SECOND MOMENT/ORDER APPROXIMATIONS BY KERNEL SMOOTHERS WITH APPLICATION TO VOLATILITY ESTIMATION

L. Beleña[1,2], *E. Curbelo*[3], *L. Martino*[1], *V. Laparra*[4]

[1] Universidad Rey Juan Carlos, Madrid, Spain.

[2] Universidad Francisco de Vitoria, Madrid, Spain.

[3] Universidad Carlos III de Madrid (UC3M), Madrid, Spain.

[4]Image Processing Lab, Universitat de Valencia, Paterna, Spain.

ABSTRACT

Volatility estimation and quantile regression are relevant active research areas in statistics, machine learning and econometrics. In this work, we propose two procedures to estimate local variances in generic regression problems by using of kernel smoothers. The proposed schemes can be applied in multidimesional scenarios (not just for time series analysis) and easily in a multi-output framework, as well. Moreover, they allow the possibility of providing uncertainty estimation using a generic kernel smoother technique. Several numerical experiments show the benefits of the proposed methods, even comparing with benchmark techniques. One of these experiment involves a real dataset analysis.

Index Terms— Quantile regression, kernel smoothers, times series, heteroscedasticity, nearest neighbours.

1. INTRODUCTION

Volatility estimation (intended as local variance or local standard deviation) and quantile regression analysis are currently important tasks in statistics, machine learning and econometrics. The problem of the estimation of the volatility arised in financial time series analysis, where the volatility represents the degree of variation of a trading price series over time, usually measured by the standard deviation of logarithmic returns. It is still an important active area of research [11]. Quantile regression models study the relationship between a set of predictor (independent) vector $\mathbf{x}$ and specific percentiles (or "quantiles") of a output (dependent) variable y. Therefore, in quantile regression analysis the goal is to estimate higher order quantiles of the response/output variable. For instance, considering also second order moments, we can also express the local/instant variance how confident you are in that current trend prediction [4, 10, 16]. It is important to remark that most of the standard and/or advance regression methods consider the variance constant, that does vary with the input variable $\mathbf{x}$ [23].

In this work, we propose two procedures for estimating the local variance in a regression problem and using a kernel smoother approach. Note that the kernel smoother schemes contain several well-known techniques as special cases, such as the *fixed radius nearest neighbours*, as an example [5]. The resulting

solution is *non-parametric* method, hence both complexity and flexibility of the solution grows with the number of data N. This ensures to have the adaquate flexibility to analyze the data.
The first proposed method is based on the Nadaraya-Watson derivation of a linear kernel smoother [6, 23]. The second proposed approach draws inspiration from *divisive normalization*, a function grounded in the activity of brain neurons [9]. This function aims to standardize neuron activity by dividing it by the activity of neighboring neurons. It has demonstrated favorable statistical properties [3, 17, 21] and has been utilized in various applications [15, 18].
The proposed methods can be applied to time series analysis and/or in more general regression problems where the input variables are multidimensional. Therefore, our approach can be implemented in spatial statistical modelling and any other inverse problems [22, 28]. Furthermore, another advantage of the proposed scenario, is that the generalization for a multi-output scenarios can be easily designed.
From another point of view, this work gives another relevant contribution. The proposed schemes allow to perform uncertainty estimation with a generic kernel smoother technique. Indeed, a generic kernel smoother method is not generally supported by a generative probabilistic derivation that yields also an uncertainty estimation. The Gaussian processes (GPs) and relevance vector machines (RVMs) regression methods are relevant well-known and virtually unique exceptions [23, 27]).
We have tested the proposed methods in five numerical experiments. Four of them involve the application of the different schemes to time series analyis, and the comparison with GARCH models which are considered benchmark techniques for volatility estimation in time series [7]. The last numerical experiment addresses a more general regression problem, with a multidimensional input variable $\mathbf{x}$ of dimension 122, considering a real (emo-soundscapes) database [12, 13].

2. APPROXIMATING THE TREND

Let us consider a set of N data pairs, such as $\{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^D$, with $D \geq 1$, and $y_i \in \mathbb{R}$. First of all, we are interested in obtaining a regression function (a.k.a.,"local mean"– trend), i.e., removing the noise in the signal obtaining an estimator $\widehat{f}(\mathbf{x})$ for all possible $\mathbf{x}$ in the input space.
One possibility is to employ a *linear kernel smoother*. More specifically, we consider the Nadaraya-Watson estimator [6, 23] that has the following form,

$$E[y|\mathbf{x}] \approx \widehat{f}(\mathbf{x}) = \tag{1}$$
$$= \sum_{n=1}^{N} \frac{h(\mathbf{x}, \mathbf{x}_n)}{\sum_{j=1}^{N} h(\mathbf{x}, \mathbf{x}_j)} y_n = \sum_{n=1}^{N} \varphi(\mathbf{x}, \mathbf{x}_n) y_n,$$

where $\varphi(\mathbf{x}, \mathbf{x}_n) = \frac{h(\mathbf{x}, \mathbf{x}_n)}{\sum_{j=1}^{N} h(\mathbf{x}, \mathbf{x}_j)}$. Note that, by this definition, the nonlinear weights $\varphi(\mathbf{x}, \mathbf{x}_n)$ are normalized, i.e.,

$$\sum_{n=1}^{N} \varphi(\mathbf{x}, \mathbf{x}_n) = 1, \ \forall \mathbf{x}. \tag{2}$$

As an example, we could consider

$$h(\mathbf{x}, \mathbf{z}) = h(\mathbf{x}, \mathbf{z}|\lambda) = \exp\left(-\frac{||\mathbf{x} - \mathbf{z}||^2}{\lambda}\right),$$

where λ is a parameter that should be tuned. Clearly, we have also

$$\varphi(\mathbf{x}, \mathbf{z}) = \varphi(\mathbf{x}, \mathbf{z}|\lambda).$$

The form of this estimator above is quite general, and contains different other well-known methods as special cases. For instance, it contains the k-nearest neighbors algorithm (kNN) for regression as a specific case (to be more specific, the fixed-radius near neighbors algorithm). Indeed, with a specific choice of $h(\mathbf{x}, \mathbf{x}_n|\lambda)$ (as a rectangular function), the expression above can represent the fixed-radius near neighbors estimator [5, 23].

Remark. The resulting regression function is flexible non-parametric method. Both complexity and flexibility of the solution grows with the number of data N.

Remark. The input variables $\mathbf{x}_n \in \mathbb{R}^D$ are vectors ($D \geq 1$), in general. Therefore, the described methods have a much wider applicability than the techniques that can be employed only for time series (where the time index is a scalar number, $x = t \in \mathbb{R}$). Clearly, the methodologies described here can be also employed for analyzing time series. Moreover, even in a time series framework, we can obtain a prediction between two consecutive time instants. For instance, if we have a time series with daily data, with a kernel smoother we can have a prediction at each hour (or minute) within a specific day.

Learning λ. One possibility for tuning the parameters of the kernel functions is to use a cross-validation (CV) procedure. In this work, we have employed a *leave-one-out* cross-validation (LOO-CV) [6].

3. VARIANCE ESTIMATION PROCEDURES

Let assume that we have already computed the trend (a.k.a., "local mean"), i.e., the regression function $\widehat{f}(\mathbf{x})$. Here we present two methods to obtain an estimation of the local variance (or volatility) at each point $\mathbf{x}$.

METHOD-1 (M1). If the weights $\varphi(\mathbf{x}, \mathbf{x}_n)$ are adequate for linearly combined the outputs y_i and obtaining an proper approximation of $E[y|\mathbf{x}]$, one can extend this idea for approximating the second non-central moment $E[y^2|x]$ as

$$E[y^2|\mathbf{x}] \approx \widehat{q}(\mathbf{x}) = \sum_{n=1}^{N} \varphi(\mathbf{x}, \mathbf{x}_n) y_n^2. \tag{3}$$

hence

$$\widehat{v}(\mathbf{x}) = \widehat{q}(\mathbf{x}) - \left(\widehat{f}(\mathbf{x})\right)^2 \approx \text{var}[y|\mathbf{x}], \tag{4}$$

which is an estimator of the *instant variance*. Note that $\widehat{q(\mathbf{x})}$ and $\widehat{f}(\mathbf{x})$ and obtained with the same weights

$$\varphi(\mathbf{x}, \mathbf{x}_n) = \varphi(\mathbf{x}, \mathbf{x}_n|\lambda),$$

with the same value of λ (obtained using a LOO-CV procedure). Thus, a variant of this procedure consists into learning another value of λ_2, i.e., obtaining other coefficients $\varphi_2(\mathbf{x}, \mathbf{z}) = \varphi_2(\mathbf{x}, \mathbf{z}|\lambda_2)$, in the Eq. (3) considering the signal y_n^2 (instead of y_n).

METHOD-2 (M2). Let us define the signal obtained as the estimated square errors

$$v_n = \left(y_n - \widehat{f}(\mathbf{x})\right)^2, \quad n = 1, ..., N.$$

If $\widehat{f}(\mathbf{x}) \approx E[y|\mathbf{x}]$ as assumed, v_n is *a one-sample estimate* of the variance at $\mathbf{x}$. Then, the goal is to approximate the trend of this new signal (new output) s_n,

$$\widehat{v}(\mathbf{x}) = \sum_{n=1}^{N} \varphi_2(\mathbf{x}, \mathbf{x}_n) v_n,$$

where we consider another parameter λ_2, i.e.,

$$\varphi_2(\mathbf{x}, \mathbf{z}) = \varphi_2(\mathbf{x}, \mathbf{z}|\lambda_2),$$

that is tuned again with LOO-CV but considering the new signal v_n. Note that $\widehat{v}(\mathbf{x})$ can be interpreted as estimation of the *instant variance* of the underlying signal. As alternative, also completely different kernel functions (as φ_2) can be applied (that differs for the functional form with respect to φ, instead just for the choice of λ).

Remark. Again, as for estimating the trend, note that the two procedures above can be applied for multivariate inputs $\mathbf{x}_n \in \mathbb{R}^D$, and not just for scalar inputs (as in the time series).

Remark. If one divides $\mathbf{x}$ by $\widehat{v}(\mathbf{x})$ we get a signal with uniform local variance, i.e. we have removed the (possible) heteroscedasticity. In [17], this procedure was used to define the relation kernels in the divisive normalization, and thus equalize locally the energy of neuron responses.

4. EXTENSIONS AND VARIANTS

The use of linear kernel smoothers as regression methods is not mandatory. Indeed, the ideas previously described can be employed even applying different regression methods:

1. Choose a regression method. Obtain a trend function $\widehat{f}(\mathbf{x}) \approx E[y|\mathbf{x}]$ given the dataset $\{\mathbf{x}_i, y_i\}_{i=1}^N$.
2. Choose one method for estimating the instance variance (between the two below):
   - **M1**. Choose a regression method (the same of the previous step or a different one). Considering the dataset the dataset $\{\mathbf{x}_i, y_i^2\}_{i=1}^N$ and obtain $\widehat{q}(\mathbf{x}) \approx E[y^2|\mathbf{x}]$. Then, compute $\widehat{v}(\mathbf{x}) = \widehat{q}(\mathbf{x}) - (\widehat{f}(\mathbf{x}))^2$.
   - **M2**. Choose a regression method (the same of the previous step or a different one). Considering the dataset the dataset $\{\mathbf{x}_i, v_i\}_{i=1}^N$ where $v_n = (y_n - \widehat{f}(\mathbf{x}))^2$, obtain the function $\widehat{v}(\mathbf{x})$.

4.1. Multi-output scenario and other extensions

In a multi-output framework, we have N data pairs, such as $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^D$, but in this case also the outputs are vectors $\mathbf{y}_i \in \mathbb{R}^{D_Y}$. Hence, for the local trend we have also a vectorial hidden function $\widehat{\mathbf{f}}(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}^{D_Y}$ for all possible $\mathbf{x}$ in the input space. Then, we could easily write

$$\widehat{\mathbf{f}}(\mathbf{x}) = \sum_{n=1}^{N} \varphi(\mathbf{x}, \mathbf{x}_n) \mathbf{y}_n. \tag{5}$$

Regarding the estimations of local variances follow the same procedures, defining the following vectorial quantities: $\widehat{\mathbf{q}}(\mathbf{x}) \approx E[\mathbf{y}^2|\mathbf{x}]$ for M1, $\widehat{\mathbf{v}}(\mathbf{x}) = \widehat{\mathbf{q}}(\mathbf{x}) - (\widehat{\mathbf{f}}(\mathbf{x}))^2$ and $\mathbf{v}_n = (\mathbf{y}_n - \widehat{\mathbf{f}}(\mathbf{x}))^2$ for M2.
Furthermore, we could also consider different local parameters λ, for instance λ_n with $n = 1, ..., N$, or more generally, with $\lambda = \lambda(\mathbf{x})$. In this case, we would have different coefficient functions $\varphi_n(\mathbf{x}, \mathbf{x}_n)$ (one for each input $\mathbf{x}_n$), so o that trend would be

$$\widehat{\mathbf{f}}(\mathbf{x}) = \sum_{n=1}^{N} \varphi_n(\mathbf{x}, \mathbf{x}_n)\mathbf{y}_n. \tag{6}$$

4.2. Example of alternative to the kernel smoothers for time series analysis

Let us consider a time series framework, i.e., the input is a scalar time instance, $x = t \in \mathbb{R}$, then the dataset is formed by the following pairs $\{t_i, y_i\}_{i=1}^N$. Let also consider that the intervals, $t_i - t_{i-1}$, are constant, in this case, we can skip the t index, and consider the dataset $\{t_i, y_i\}_{i=1}^N$. In this scenario, as an alternative, one can be use an autoregressive (AR) model,

$$y_i = a_1 y_{i-1} + a_2 y_{i-2} + ... + a_W y_{i-W} + \epsilon_y, \tag{7}$$

where ϵ_y is a noise perturbation and the coefficients a_i, for $i = 1, ..., W$, are obtained by a least squares (LS) minimization and the length of the temporal window W (i.e., the order of the AR filter) is obtained using a spectral information criterion (SIC) [24, 26]. Then, considering the estimated coefficients $\widehat{a}_i$ by LS, and $\widehat{W}$ by SIC, we have

$$\widehat{f}(t_i) = \widehat{f}_i = \widehat{a}_1 y_{i-1} + \widehat{a}_2 y_{i-2} + ... + \widehat{a}_{\widehat{W}} y_{i-\widehat{W}}. \tag{8}$$

Just as an example, in order to apply M2 for the estimating the instant variance, we can set $v_i = (y_i - \widehat{f}_i)^2$, and assume another AR model over v_i with window length now denoted as H,

$$v_i = b_1 v_{i-1} + b_2 v_{i-2} + ... + b_H v_{i-H} + \epsilon_v, \tag{9}$$

where again the estimations $\widehat{b}_i$ can be obtained by LS, and $\widehat{H}$ by SIC. The resulting instant variance function is

$$v(t_i) = v_i = \widehat{b}_1 v_{i-1} + \widehat{b}_2 v_{i-2} + ... + \widehat{b}_{\widehat{H}} v_{i-\widehat{H}}. \tag{10}$$

Clearly, the application of M1 could be performed in a similar way.

Remark. The resulting estimators in this section are still a linear combination of (a portion) of the outputs. Therefore, we still have linear smoothers (or better *linear filters*, since only considering combinations of past samples). However, note that in Eqs. (8)-(10) the coefficients of the linear combinations are obtained by least squares (LS) method. Hence, we have a window of length W (and then H) but this differs from the use of a rectangular kernel function for two reasons: (a) the window only considers past samples; (b) the coefficients are not all equals (to the ratio 1 over the number of samples included in the window) but they are tuned by LS.

Table 1: Proposed methods (specific implementations).

Method	**KS-1-1**	**KS-1-2**	**KS-2-1**	**KS-2-2**	**KS-10-1**	**KS-10-2**	**AR-based**
For trend estimation	Eqs. (1)-(12), $\alpha = 1$		Eqs. (1)-(12), $\alpha = 2$		Eqs. (1)-(12), $\alpha = 10$		AR - Section 4.2
For variance estimation	M1	M2	M1	M2	M1	M2	AR and M2 - Section 4.2

5. NUMERICAL EXPERIMENTS

5.1. Applications to time series

In this section, we consider 5 different numerical experiments with different (true) generating models. Here, we focus on time series ($x = t \in \mathbb{R}$) for two main reasons: (a) first of all, for the sake of simplicity (we can easily show a realization of the data in one plot), (b) and last but not the least, we have relevant benchmark competitors such as the GARCH models [8, 14, 29], that we can also test. Note that GARCH models have been specifically designed for handling volatility in time series.
More specifically, in this section, we test our methodologies and different GARCH models in four easy reproducible examples. These four examples differ for the data generating model. Indeed, the data are generated as the sum of a mean function, $f(x)$, and an error term, $s(x)\varepsilon(x)$, i.e.,

$$\begin{aligned} y(x) &= f(x) + s(x)\varepsilon(x) \\ y(t) &= f(t) + s(t)\varepsilon(t), \quad \varepsilon(t) \sim \mathcal{N}(0,1). \end{aligned} \tag{11}$$

where we have used the fact that we are analyzing time series, i.e., $x = t \in \mathbb{R}$. The functions $f(t)$ and $s(t)$ are deterministic, and represent for the trend and the standard deviation of the observations $y(t)$. Whereas $\varepsilon(t) \sim \mathcal{N}(0,1)$ represent a standard Gaussian random noise. Note also that $v(t) = s(t)^2$. In order to keep the notation of the rest of the paper, note that the dataset $\{t_n, y_n\}_{n=1}^N$ will be formed by the N pairs,

$$x_n = t_n, \quad y_n = y(t_n),$$

where y_n will be generated according to the model above. The four experiments consider:

1. $f(t) = \sin(0.5t), \quad s(t) = 0.2.$
2. $f(t) = t, \quad s(t) = 2|t|.$
3. $f(t) = 0.5t^2, \quad s(t) = 5\sin(0.3t).$
4. $f(t) = 0$ and $s(t)$ generated by a GARCH model.

In all cases, the generated data $y(t)$ and the undelying standard deviation $s(t)$ (or variance $v(t) = s(t)^2$) are known, hence we can compute errors and evaluate the performance of the used algorithms. For the application of the proposed methods we consider a nonlinear kernel function as

$$h(t, t_n) = \exp\left(-\frac{(t - t_n)^\alpha}{\lambda}\right), \tag{12}$$

with $\alpha \in \{1, 2, 10\}$. The parameter λ is tuned by LOO-CV. We test 7 different combinations, each one mixing a specific trend approximation method with a specific variance estimation technique:

- **Based on kernel smoothers**: we consider different regression methods based in Eqs. (1) and (12). Furthermore, for the variance estimation we consider both procedures M1 and M2 described in Section 3. More specifically:

  - **KS-1-1**: $\alpha = 1$ and M1,
  - **KS-1-2**: $\alpha = 1$ and M2,
  - **KS-2-1**: $\alpha = 2$ and M1,
  - **KS-2-2**: $\alpha = 2$ and M2,
  - **KS-10-1**: $\alpha = 10$ and M1,
  - **KS-10-2**: $\alpha = 10$ and M2.

- **Based on AR models**: we apply the method described in Section 4.2, that employs two auto-regressive models, one for the trend and one for the variance, jointly with procedure M2.

Table 1 summarizes all the seven considered specific implementations. Moreover, we test several GARCH(p,q) models: GARCH(1,1), GARCH(2,2), GARCH(5,5) and GARCH(10,10) (using a Matlab implementation of GARCH models).

Remark. The GARCH models work directly with the signal without trend, i.e., $|y_n - \widehat{f}_n|$. In order to have a fair comparison, we use a kernel smoother with $\alpha = 2$, i.e., applying Eqs. (1)-(12) with $\alpha = 2$ (as in the KS-2 methods).

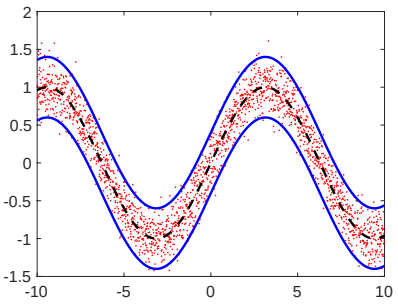
We recall that GARCH models have been specific designed for making inference in heteroscedastic time series.

5.1.1. Experiment 1

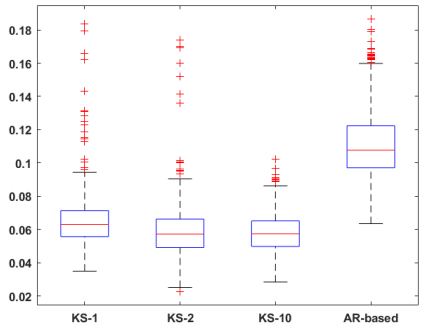
As previously said, in this first example we consider

$$y(t) = \sin(0.5t) + 0.2\,\varepsilon(t), \quad \varepsilon(t) \sim \mathcal{N}(0,1),$$

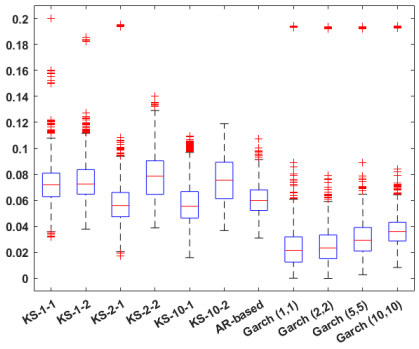
i.e., where $f(t) = \sin(0.5t)$ and $s(t) = 0.2$ (in this case we have homoscedastic series). We generate $N = 10^4$ data with $t \in [-10, 10]$. We apply all the algorithms, compute the mean absolute error (MAE), and repeat and average the results over 10^4 independent runs. Figure 1(a) depicts one realization of the generated data with the model above.
Results: Figures 1(b)-1(c) provide the results in this example. Figure 1(b) shows the MAE obtained in estimating the trend, and Figure 1(c) depicts the MAE in estimating the standard deviation. In this case, regarding the trad, we see that the AR-based has an MAE higher than the rest when estimating the trend, but the other MAEs are quite similar. Similar outcomes are also obtained for approximating with the standard deviation, although the smaller MAE are provided by the GARCH models. Regarding the estimating of the standard deviation, the M1 seems to provide better results than M2, in this example.



(a) Realization of the data in the example 5.1.1.



(b) MAE in trend estimation.



(c) MAE in standard deviation estimation.

Fig. 1: **(a)** A realization of the data (red dots) generated by the model in Section 5.1.1; the function $f(t)$ is shown dashed black line and the two continue blue lines depict $f(t) \pm 2s(t)$ (containing approx. 95% of the probability mass). **(b)-(c)** Box-plots of the MAEs of the different algorithms of the experiment in Section 5.1.1.

5.1.2. Experiment 2

in this second example, we consider

$$y(t) = t + 2|t|\,\varepsilon(t), \quad \varepsilon(t) \sim \mathcal{N}(0,1),$$

where $f(t) = t$ and $s(t) = 2|t|$ (i.e., in this case, we are heteroscedastic scenario). As previously, we generate $N = 10^4$ data with $t \in [-10, 10]$. We apply all the algorithms, compute the mean absolute error (MAE), and again we repeat and average the results over 10^4 independent runs. Figure 2(a) depicts one realization of the generated data according to the model above.

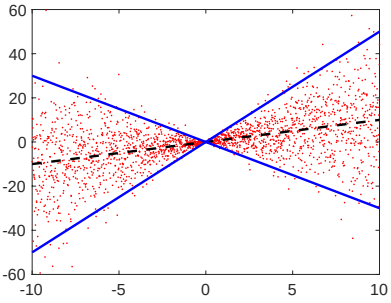
Results. The results are presented in Figures 2(b)-2(c). Again the MAE for the AR-based method seems to be higher than those for the rest of the algorithms when estimating the trend and standard deviation of the time series. Regarding the estimation of the standard deviation the best results are provided by the kernel methods which use the M2 (the MAE is particularly small with $\alpha = 10$). Thus, it is remarkable that, even in this *heteroscedastic scenario*, the proposed methods provide similar and, in same cases, better results than GARCH models.

5.1.3. Experiment 3

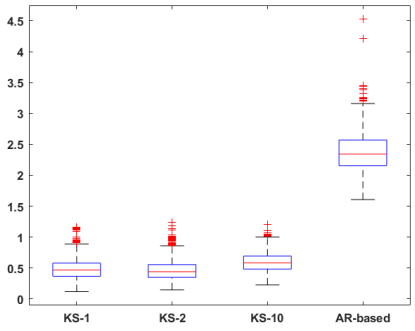
In this third example, we consider a standard deviation that varies periodically, and a second order polynomial as a trend function, i.e.,

$$y(t) = 0.5t^2 + 5\sin(0.3t)\,\varepsilon(t), \quad \varepsilon(t) \sim \mathcal{N}(0,1),$$

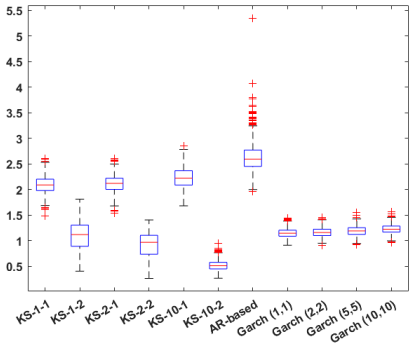
where $f(t) = 0.5t^2$ and $s(x) = 5\sin(0.3t)$. Again, we have heterostodastic scenario. As previously, we generate $N = 10^4$ data with $t \in [-10, 10]$. We compute the mean absolute error (MAE), and again we repeat and average the results over 10^4 independent runs. Figure 3(a) depicts one realization of the generated data according to the described model.



(a) Realization of the data in the example 5.1.2.



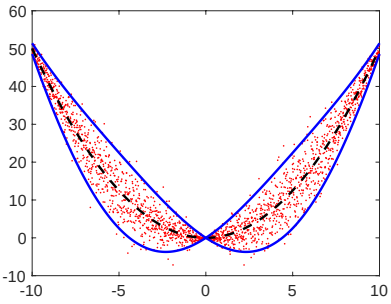
(b) MAE in trend estimation.



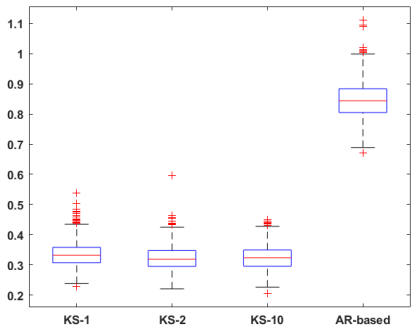
(c) MAE in trend standard deviation estimation.

Fig. 2: **(a)** A realization of the data (red dots) generated by the model in Section 5.1.2; the function $f(t)$ is shown dashed black line and the two continue blue lines depict $f(t) \pm 2s(t)$ (containing approx. 95% of the probability mass). **(b)-(c)** Box-plots of the MAEs of the different algorithms of the experiment in Section 5.1.2.

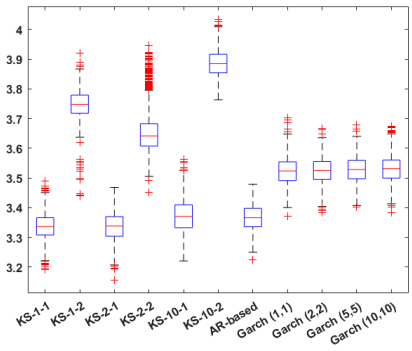
Results. Figures 3(b)-3(c) depict the box-plots of the MAE in estimating the trend and the standard deviation. As with the rest of the examples, the AR-based gives the worst results when calculating the trend of the time series, meanwhile the rest of the algorithms have very similar error values (recall that the GARCH models employ the KS-2 method for the trend estimation). However, the AR-based method provides good performance in the approximation of the standard deviation. The AR-based technique uses M2 for the variance. Moreover, for the standard deviation estimation, the methods KS-1-1-1, KS-2-1, KS-10-1 gives the smaller errors. Hence, it seems that in this case M1 works better even than GARCH models.



(a) Realization of the data in the example 5.1.3.



(b) MAE in trend estimation.



(c) MAE in standard deviation estimation.

Fig. 3: **(a)** A realization of the data (red dots) generated by the model in Section 5.1.3; the function $f(t)$ is shown dashed black line and the two continue blue lines depict $f(t) \pm 2s(t)$ (containing approx. 95% of the probability mass). **(b)-(c)** Box-plots of the MAEs of the different algorithms of the experiment in Section 5.1.3.

5.1.4. Experiment 4

Let us consider

$$y(t) = s(t)\,\varepsilon(t), \quad \varepsilon(t) \sim \mathcal{N}(0,1),$$
$$y_t = s_t\epsilon_t = z_t,$$

i.e., $f(t) = f_t = 0$, and we have denoted $z_t = s_t\epsilon_t$. The process for the standard deviation $s_t = \sqrt{v_t}$ will be a GARCH(P, Q), i.e.,

$$\begin{aligned} v_t =& \kappa + \gamma_1 v_{t-1} + \ldots + \gamma_P v_{t-P} + \\ &+ \alpha_1 z_{t-1}^2 + \ldots + \alpha_Q z_{t-Q}^2. \end{aligned}$$

In this example, we generate the variance from a GARCH$(2, 3)$

$$\begin{aligned} v_t =& 2 + 0.2 \cdot v_{t-1} + 0.1 \cdot v_{t-2} + 0.2 \cdot z_{t-1}^2 + \\ &+ 0.2 \cdot z_{t-2}^2 + 0.2 \cdot z_{t-3}^2 \end{aligned} \tag{13}$$

Clearly, Again, it is a heterostodastic scenario. we generate $N = 10^4$ data and apply the different techniques. We compute the mean absolute error (MAE), and we repeat and average the results over 10^4 independent runs. One realization of the generated data is given in Figure 4(a). Note that the trend is linear but the volatility presents very fast changes since it generated by a GARCH(2,3) model (see blue line in Figure 4(a)).

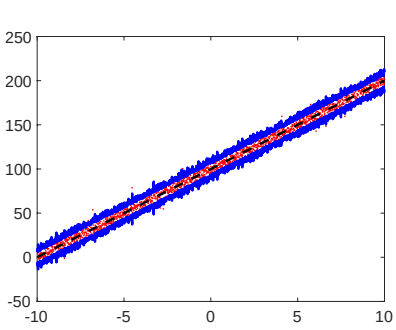
Results. The results for this example are presented in Figures 4(b)-4(c). The best performance in the estimation of $s(t)$ are obtained by KS-1-2 and all the GARCH models (as expected in this case). The AR-based and KS-2-2 also provided close values of MAE. In this example, clearly M2 outperforms M1. We remark that, even in this scenario that is particularly favorable for the GARCH models, all the proposed methods provide competitive results. The highest (hence worst) MAEs values are given by the KS with $\alpha = 10$ which is a quite extreme tuning/choice of this parameter, specially when the volatility follows an autoregressive rule. Indeed, a Laplacian kernel with $\alpha = 1$ is more adequate in this framework. For further considerations see [23, Section 10].

5.1.5. Final comments about the applications to time series

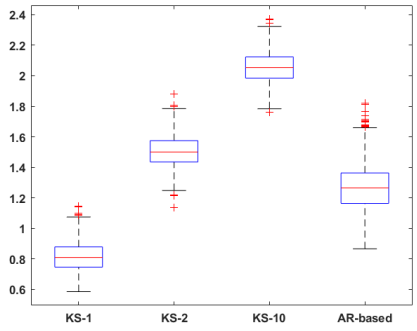
All the proposed methods based on kernel smoothers (KS) and M1, M2 always provide competitive results (with closer or smaller MAE values) with respect to the benchemark algorithms such as GARCH methods, which have been specifically designed for the estimation of the standard deviation in time series.
These numerical experiments show that The proposed M1 and M2 can even outperform GARCH models the estimation of the standard deviation in time series. The GARCH models seems to provide more robust results when the signal has been generated truly by a GARCH model in Section 5.1.4 and, surprisingly, in the homoscedasticscenario in Section 5.1.1. The methods based on kernel smoothers virtually outperform the AR-based due to the fact that they incorporate the "future samples" in their estimators. A rectangular window, including both past and future samples, should improve the performance of the AR-based method.

5.2. Application in higher input dimension: real data on soundscape emotions

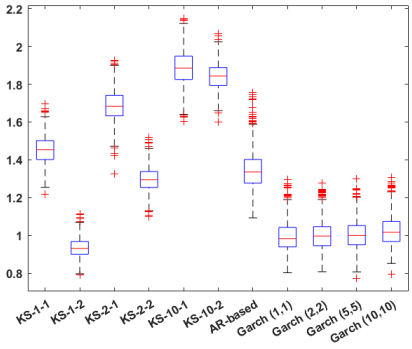
GARCH models have been specifically designed for estimating the volatility in time series, hence $x = t \in \mathbb{R}$. However, the methods proposed in this work can be applied to problems with multidimensional inputs



(a) Realization of the data in the example 5.1.4.



(b) MAE in trend estimation.



(c) MAE in standard deviation estimation.

Fig. 4: **(a)** A realization of the data (red dots) generated by the model in Section 5.1.4. The linear trend $f(t) = t$ is shown dashed black line. Whereare the two continue blue lines depict $f(t) \pm 2s(t)$ where $s(t)$ is generated by a GARCH model. **(b)-(c)** Box-plots of the MAEs of the different algorithms of the experiment in Section 5.1.4.

$\mathbf{x} \in \mathbb{R}^D$ with $D \geq 1$.

More specifically, here we focus on analyzing a soundscape emotion database. In the last decade, soundscapes have become one of the most active topics in acoustics. In fact, the number of related research projects and scientific articles grows exponentially [1,19,20]. In urban planning and environmental acoustics, the procedure consists of (a) soundscapes recording, (b) calculation of acoustic and psychoacoustic indicators of the signals, (c) collecting other context indicators (e.g. visual information [2]), and (d) ranking of soundscapes audio signals employing emotional descriptors. Finally, the model can be developed [25].

Here, we consider the emo-soundscapes database (EMO) [12, 13], where $N = 1200$ and $D = 122$, i.e., $\mathbf{x} \in \mathbb{R}^{122}$. that might be considered as the largest publicly available soundscape database with annotations of emotion labels, and the most bench-marked up to now [12]. EMO contains 1213 audio clips which are released under creative commons license from the freesound collaborative audio platform [13]. Among the 122 audio features that form the inputs $\mathbf{x}$, we can distingush three main groups of features of the audio signals: psychoacoustic features, time-domain features and frequency-domain features. The output y that we consider is called *arousal*. For further information, the complete database can be found at `https://metacreation.net/emo-soundscapes/`.

We employ again a kernel function of type

$$h(\mathbf{x}, \mathbf{z}) = h(\mathbf{x}, \mathbf{z}|\lambda) = \exp\left(-\frac{||\mathbf{x} - \mathbf{z}||^\alpha}{\lambda}\right),$$

with $\alpha = 2$. Therefore, considering the EMO database $\{\mathbf{x}_i, y_i\}_{i=1}^{1200}$ We apply M1 and M2 and CV-LOO for tuning λ.The relative mean square error in estimating the trend is 0.0388 obtained with an optimal parameter $\lambda^* = 0.12$. Moreover, in both cases, we find a sensible increase of the volatility of the output in the last half of the samples, for i from 621 to 1213. This could be an interesting result, that should be discussed with experts on the field and in the EMO database.

6. CONCLUSIONS

In this work, we have proposed two methods for estimating the instant/local variance in regression problems based on kernel smoothers. From another point of view, this work can be also seen as a way to perform quantile regression (at least, a second order quantile regression) using kernel smoothers.
With respect to other procedures given in the literature, the proposed methods have a much wider applicability than other techniques, since they can be employed only for time series, as the well-known GARCH models (where the time index is a scalar number, $x = t \in \mathbb{R}$). Indeed, the proposed schemes can be applied with multidimensional input variables $\mathbf{x}_n \in \mathbb{R}^D$. Moreover, the proposed techniques can be also applied to time series ($D = 1$) and also can obtain a prediction between two consecutive time instants (i.e., with an higher resolution that the received data). More generally, estimation of the local variance can given at any the training data $\mathbf{x}_i$ but also in any generic test input $\mathbf{x}$.
Furthermore, analyzing time series, the numerical simulations have shown that the proposed schemes provides competitive results even with respect to GARCH models (which are specifically design for analyzing time series). An application to real data and multidimensional inputs (with dimension 122) have been also provided. As a future research line, we plan to design kernel smoother functions with local parameters λ, or more generally with $\lambda = \lambda(\mathbf{x})$, to better handle the heteroscedasticity. Moreover, the application to multioutput scenarios can be designed and considered.

acknowledgments

This work was partially supported by the Young Researchers R&D Project, ref. num. F861 (AUTO-BAGRAPH) funded by Community of Madrid and Rey Juan Carlos University, and by Agencia Estatal de Investigación AEI (project SP-GRAPH, ref. num. PID2019-105032GB-I00).

7. REFERENCES

[1] F. Aletta and J. Xiao. What are the current priorities and challenges for (urban) soundscape research? *Challenges*, 9(1):16, 2018.

[2] O. Axelsson, M. E. Nilsson, and B. Berglund. A principal components model of soundscape perception. *The Journal of the Acoustical Society of America*, 128(5):2836–2846, 2010.

[3] J. Ballé, V. Laparra, and E. Simoncelli. Density Modeling of Images using a Generalized Normalization Transformation. pages 1–14, 2015.

[4] D. G. Baur and T. Dimpfl. A quantile regression approach to estimate the variance of financial returns. *Journal of Financial Econometrics*, 17(4):616–644, 2019.

[5] J. L. Bentley, D. F. Stanat, and E. Hollins Williams. The complexity of finding fixed-radius near neighbors. *Information Processing Letters*, 6(6):209–212, 1977.

[6] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2006.

[7] T. Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327, 1986.

[8] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327, April 1986.

[9] M. Carandini and D. Heeger. Normalization as a canonical neural computation. *Nat. Rev. Neurosci.*, 13(1):51–62, 2012.

[10] I. C. Chronopoulos, A. Raftapostolos, and G. Kapetanios. Forecasting value-at-risk using deep neural network quantile regression. *Journal of Financial Econometrics*, 2023.

[11] R. Engle. Risk and volatility: Econometric models and financial practice. *American economic review*, 94(3):405–420, 2004.

[12] Jianyu Fan, Miles Thorogood, and Philippe Pasquier. Emo-soundscapes: A dataset for soundscape emotion recognition. In *2017 Seventh international conference on affective computing and intelligent interaction (ACII)*, pages 196–201. IEEE, 2017.

[13] E. Fonseca, J. Pons Puig, X. Favory, F. Font Corbera, Dm. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra. Freesound datasets: a platform for the creation of open audio datasets. In *Hu X, Cunningham SJ, Turnbull D, Duan Z, editors. Proceedings of the 18th ISMIR Conference; 2017 oct 23-27; Suzhou, China.[Canada]: International Society for Music Information Retrieval; 2017. p. 486-93.* International Society for Music Information Retrieval (ISMIR), 2017.

[14] P. R. Hansen and A. Lunde. A forecast comparison of volatility models: does anything beat a garch (1, 1)? *Journal of applied econometrics*, 20(7):873–889, 2005.

[15] Pablo HernÃąndez-CÃąmara, Valero Laparra, and JesÃžs Malo. Neural networks with divisive normalization for image segmentation, 2023.

[16] Q. Huang, H. Zhang, J. Chen, and M. He. Quantile regression models and their applications: a review. *Journal of Biometrics & Biostatistics*, 8(3):1–6, 2017.

[17] V Laparra, J Ballé, A Berardino, and E. Simoncelli. Perceptual image quality assessment using a normalized laplacian pyramid. In *Proc. Human Vis. Elect. Im.*, volume 2016, pages 1–6, 2016.

[18] V. Laparra, A. Berardino, J. BallÃl', and E. Simoncelli. Perceptually optimized image rendering. *Journal of the Optical Society of America A*, 34(9):1511, Aug 2017.

[19] M. Lionello, F. Aletta, and J. Kang. A systematic review of prediction models for the experience of urban soundscapes. *Applied Acoustics*, 170:107479, 2020.

[20] P. Lundén and M. Hurtig. On urban soundscape mapping: A computer can predict the outcome of soundscape assessments. In *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*, volume 253, pages 2017–2024. Institute of Noise Control Engineering, 2016.

[21] J. Malo and V. Laparra. Psychophysically tuned divisive normalization approximately factorizes the pdf of natural images. *Neural computation*, 22(12):3179–3206, 2010.

[22] L. Martino, F. Llorente, E. Curbelo, J. López-Santiago, and J. Míguez. Automatic tempered posterior distributions for bayesian inversion problems. *Mathematics*, 9(7), 2021.

[23] L. Martino and J. Read. Joint introduction to Gaussian Processes and Relevance Vector Machines with connections to Kalman filtering and other kernel smoothers. *Information Fusion*, 74:17–38, 2021.

[24] L. Martino, R. San Millan-Castillo, and E. Morgado. Spectral information criterion for automatic elbow detection. *Expert Systems with Applications*, 231:120705, 2023.

[25] R. San Millan-Castillo, L. Martino, E. Morgado, and F. Llorente. An exhaustive variable selection study for linear models of soundscape emotions: Rankings and gibbs analysis. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:2460–2474, 2022.

[26] E. Morgado, L. Martino, and R. San Millan-Castillo. Universal and automatic elbow detection for learning the effective number of components in model selection problems. *Digital Signal Processing*, 140:104103, 2023.

[27] C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. MIT Press, 2006.

[28] D. Heestermans Svendsen, L. Martino, and G. Camps-Valls. Active emulation of computer codes with Gaussian processes - application to remote sensing. *Pattern Recognition*, 100:107103, 2020.

[29] J. R. Trapero, M. Cardos, and N. Kourentzes. Empirical safety stock estimation based on kernel and garch models. *Omega*, 84:199–211, 2019.