Journal of the Nigerian Association of Mathematical Physics
***Volume* 24 (July, 2013), pp 563 – 568**

Higher-Order Gaussian Kernel in Meshsize Boosting Algorithm

[1]Ishiekwene, C. C. and [2]Afere, B.A.E

[1]Department of Mathematics
University of Benin, Nigeria
[2]Department of Mathematics & Statistics
Federal Polytechnic Idah, Nigeria

Abstract

In this paper, the higher-order Gaussian kernel in meshsize boosting algorithm is presented. The algorithm is a bias reduction scheme but uses the higher-order Gaussian kernel instead of the regular fixed kernels. A comparative study for this scheme is conducted and the findings reveal bias reduction for higher order Gaussian kernels when compared with existing regular fixed kernels.

Keywords: Boosting, kernel density estimates, bias reduction, higher-order Gaussian kernel, meshsize, fixed kernels.
AMS classification: 62F40, 62G08

1.0 Introduction

Schapire [1] first proposed Boosting in kernel density estimation. Other authors [2 – 4] but to mention a few have also made contributions. Boosting is a means of improving the performance of a ‘weak learner’ in the sense that given sufficient data, it would be guaranteed to produce an error rate which is better than “random guessing”.

Weak Learner’ is typically a decision tree algorithm in the context of classification.

. It is applied in this context using the higher-order Gaussian kernel. Boosting does not only guarantee an error rate that is better than random guessing but also deals with the correction of ‘noises’ at the tails of the distribution or where we have sparse cluster of data within a given region.

Mazio and Taylor [5] proposed an algorithm in which a kernel density classifier is boosted by suitably re-weighting the data. This weight placed on the kernel estimator, is a ratio of a log function in which the denominator is a leave-one-out estimate of the density function. A theoretical explanation is also given by [5] to show how boosting is a bias reduction technique i.e a reduction of the bias term in the expression for the popular asymptotic mean integrated squared error (AMISE) (See [3]).

2.0 METHODS

(Existing Mazio and Taylor’s leave-one-out Algorithm)

Step 1: Given $\{x_i, i = 1,2,...,n\}$, initialize $W_1(i) = 1/n$

Step 2: Select *h* (the smoothing parameter).

Step 3: For m =1, 2, … M, obtain a weighted kernel estimate

$$\hat{f}_m(x) = \sum_{i=1}^{n} \frac{W_m(i)}{h} k\left(\frac{x - x_i}{h}\right) \tag{1}$$

where k is the kernel function and w is a weight function, m is the number of boosting steps ; and then update the weights according to

$$W_{m+1}(i) = W_m(i) + \log\left\{\frac{\hat{f}_m(x_i)}{\hat{f}_m^{(-1)}(x_i)}\right\} \tag{2}$$

Corresponding author: ***Ishiekwene, C. C.*** E-mail: cycigar@yahoo.co.uk, Tel.: +2348023348430

step 4: Provide output as

$$\prod_{m=1}^{M} \hat{f}_m(x) \text{ renormalized to integrate to unity}$$

Normalization is done by summing up the m-step density estimates and dividing this sum by each estimate.
We shall see how the leave-one-out estimator of [5] in the weight function can be replaced by a meshsize estimator due to the time complexity involved.
In the leave-one-out estimator of equation (2), we require (n+(n-1)).n function evaluations of the density for each boosting step (where n is the sample size). Thus, we are using a meshsize in its place. The only limitation on this meshsize algorithm is that we must first determine the quantity $1/nh$ so as to know what the meshsize that would be placed on the weight function would be [6]. The need to use a meshsize in place of the leave-one-out lies on the fact that boosting is like the steepest-descent algorithm in unconstrained optimization and thus the meshsize is a good substitute that approximates the leave-one-out estimate of the function in updating the weight function [7,8,9]. The new meshsize algorithm is stated as:
Meshsize boosting Algorithm for Higher-order Gaussian Kernel

STEP 1: Given { x_i , i =1,2,…n} initialize $W_1(i) = 1/n$

STEP 2: Select h(the smoothing parameter)

STEP 3: For m=1,2,…,M

(i) Get

$$\hat{f}_m(x) = \sum_{1}^{n} \frac{W_m(i)}{h} \frac{1}{2\sqrt{2\pi}} \left[3-(t-t_i)^2\right] \exp- \frac{(t-t_i)^2}{2} \tag{3}$$

(ii) Update

$$W_{m+1}(i) = W_m(i) + \text{mesh} \tag{4}$$

STEP 4: Provide output

$$\prod_{1}^{M} \hat{f}_m (x)$$

and renormalize to integrate to unity.

The weight function in equation (4) of the algorithm uses a meshsize instead of the leave-one-out log ratio function of equation (2). Also, the kernel function used is the higher-order Gaussian kernel unlike the fixed used in [6].
The fixed kernel of [6] is given as follows:

$$k(t) = \frac{1}{\sqrt{2\pi}} \exp^{-\frac{1}{2}t^2} \tag{5}$$

While the n-dimension higher-order Gaussian kernel is

$$k(t) = \frac{1}{(2\pi)^{\frac{1}{n}}} \exp - \sum_{i=1}^{n} \frac{t_i^2}{2} \quad \text{, i=1,2,…,n and zero elsewhere.} \tag{6}$$

The idea of higher-order kernels via bias reduction dates back to [10,11]. Schucany and Summers [12] also applied the generalized jackknife to bias reduction in kernel density estimation and showed that it is equivalent to using higher-order kernels [13]. Ishiekwene et al [6] first introduced the meshsize boosting algorithm and has been used in several other areas of boosting just like the bootstrap boosting algorithm of [14]. See [15,16].
An expression for the asymptotic mean integrated squared error (AMISE) is given as (See [17])

$$AMISE\ \hat{f}_h(x) = \frac{1}{((2m+2)!)^2} h^{4m+4} (K_{2m+2})^2 \left\| f^{(2m+2)} \right\|_2^2 + \left(nh^{-1}\right)\left(\|K\|_2^2\right) \tag{7}$$

Where m=1,2,… , h is the smoothing parameter, k is the kernel, n is the sample size and f is the distribution.

3.0 DISCUSSION

In this section, we shall use three sets of data to illustrate our algorithm and BASIC programming language is used. Table 1 is a sample of size forty and is the lifespan of car batteries in years. Table 2 is a sample of size sixty-four and is the number of written words without mistakes in every 100 words by a set of students in a written essay. Table 3 is the scar length of patients randomly selected in millimeters [18,19].

Implementation of this algorithm is done in BASIC programming language using two boosting steps. The results are shown in Figures 3.1a – 3.3b. Figure 3.1a is the graph for Table 1 showing the bias reduction while Figure 3.1b for Table 1 shows the MISE. Figure 3.2a is the graph for Table 2 showing the bias reduction while Figure 3.2b for Table 2 shows the MISE. Figure 3.3a is the graph for Table 3 showing the bias reduction while Figure 3.3b for Table 3 shows the MISE.

In all three data sets used in this paper, we can clearly see the bias reduction which in turn translates to a reduction in the MISE. Table 4 shows the various window widths, bias^2,variance and the MISE for all three data sets.

4.0 CONCLUSION

We have shown that the higher-order Gaussian kernel can be used in place of the classical fixed kernel in boosting in kernel density estimation. The charts- Figs. 3.1a – 3.3b and table 4 reveals that the higher-order Gaussian kernel method of orders not greater than ten does better than the classical fixed kernel method in kernel density estimation. It is therefore recommended for use in place of the classical fixed kernel method in boosting in KDE having exhibited the qualities of bias reduction which translates to a reduction in the MISE.

Table 1

2.2	4.1	3.5	4.5	3.2
3.7	3.0	2.6	3.4	1.6
3.1	3.3	3.8	3.1	4.7
3.7	2.5	4.3	3.4	3.6
2.9	3.3	3.9	3.1	3.3
3.1	3.7	4.4	3.2	4.1
1.9	3.4	4.7	3.8	3.2
2.6	3.9	3.0	4.2	3.5

Table 2

88	58	92	77	81	86	90	67
69	84	85	78	79	72	86	94
70	68	69	84	88	89	82	75
74	79	67	68	96	90	66	69
70	75	81	80	77	79	80	91
86	83	79	69	83	73	75	85
76	93	97	87	75	83	81	76
74	78	83	69	91	88	82	80

Table 3

1.2	1.4	2.6	2.0	1.4	1.7	1.6	1.5	1.48	1.6
2.2	1.35	1.35	1.2	1.6	1.2	1.6	1.2	2.0	1.4
1.7	1.6	2.0	2.4	1.8	1.6	1.64	1.3	2.0	1.9
1.4	2.0	1.4	1.7	1.9	1.6	2.0	2.4	1.8	1.6
1.64	1.3	1.4	2.4	1.6	2.4	2.0	1.4	1.6	1.8
1.2	2.0	2.2	1.8	1.9	2.0	2.3	1.4	1.8	1.64
2.0	2.3	1.2	1.3	1.9	2.0	2.4	2.0	2.6	1.3
1.7	1.6	1.5	1.9	2.4	2.1	2.3	1.8	1.4	1.9
2.0	1.3	1.9	1.42	1.47	1.4	1.9	2.0	2.0	2.4
1.9	2.0	2.4	2.0	1.98	2.2	1.6	2.4	2.6	2.0
1.6	1.7	1.9	2.2	1.86	1.4	1.9	1.7	1.6	2.3

Table 4: Showing the various higher-order window widths, bias, variance and MISE for three data sets

m	n = 40				n = 64				n = 110			
	h_{opt}^m	$\int \text{Bias}^2 dx$	$\int \text{Var} dx$	MISE	h_{opt}^m	$\int \text{Bias}^2 dx$	$\int \text{Var} dx$	MISE	h_{opt}^m	$\int \text{Bias}^2 dx$	$\int \text{Var} dx$	MISE
2	0.508056	0.00273569	0.0234243	0.0261542	5.81862	0.00015979	0.00127832	0.00143811	0.233918	0.00231255	0.0185004	0.020813
4	0.511906	0.00213714	0.0232481	0.0253764	5.8684	0.000105623	0.00126747	0.0013731	0.240328	0.00150058	0.018007	0.0195076
6	0.523919	0.00141969	0.0227151	0.0241348	6.10333	0.0000761679	0.00121869	0.00129485	0.252412	0.00107156	0.017145	0.0182165
8	0.540169	0.00110159	0.0220318	0.0231333	6.32586	0.0000587908	0.00117582	0.00123461	0.263207	0.000822088	0.0164418	0.0172639
10	0.55443	0.000894378	0.0214651	0.0223595	6.51615	0.0000475616	0.00114148	0.00118904	0.272246	0.000662328	0.0158959	0.0165582
12	0.566618	0.000750119	0.0210033	0.0217535	6.67669	0.0000397868	0.00111403	0.00115382	0.279788	0.000552406	0.0154674	0.0160198
14	0.577034	0.000644507	0.0206242	0.0212687	6.8128	0.000034118	0.00109178	0.00112589	0.286139	0.000472628	0.0151241	0.0155967
16	0.585995	0.000564134	0.0203088	0.020873	6.92926	0.0000298174	0.00107343	0.00110324	0.291547	0.00041232	0.0148435	0.0152559
18	0.593773	0.00050107	0.0200428	0.0205439	7.02994	0.0000264513	0.00105805	0.0010845	0.296206	0.000365252	0.0146101	0.0149753
20	0.600582	0.000450354	0.0198156	0.0202659	7.1178	0.0000237498	0.00104499	0.00106874	0.30026	0.000327564	0.0144128	0.0147404

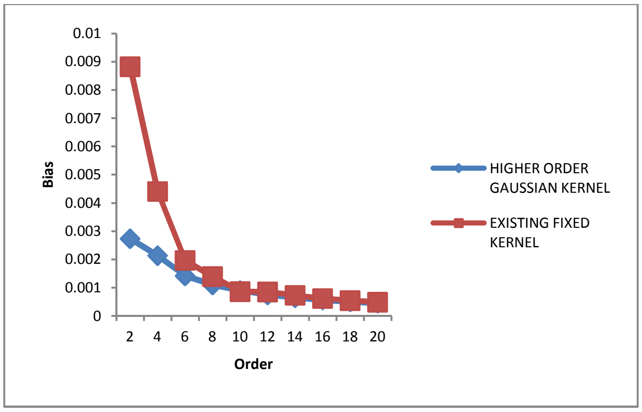

Fig 3.1a: Graph Showing the Bias for Table 1

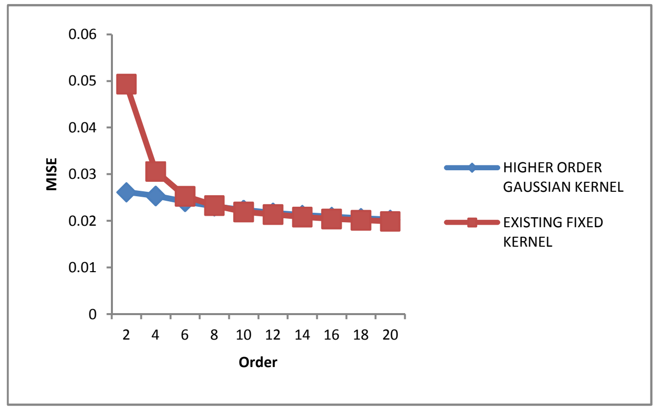

Fig 3.1b: Graph Showing the MISE for Table 1

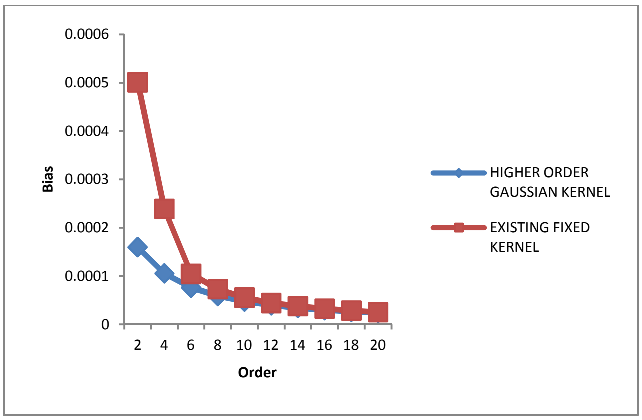

Fig 3.2a: Graph Showing the Bias for Table 2

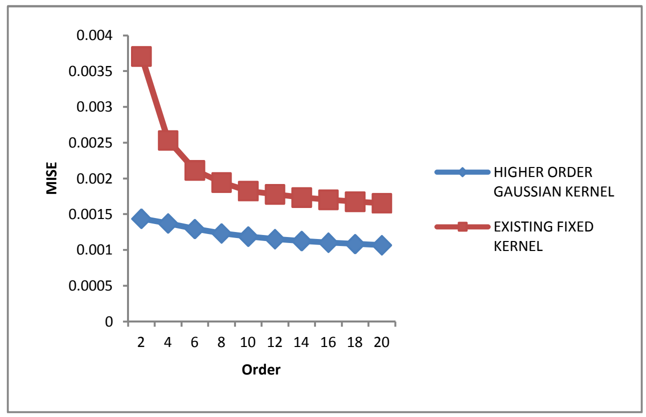

Fig 3.2b: Graph Showing the MISE for Table 2

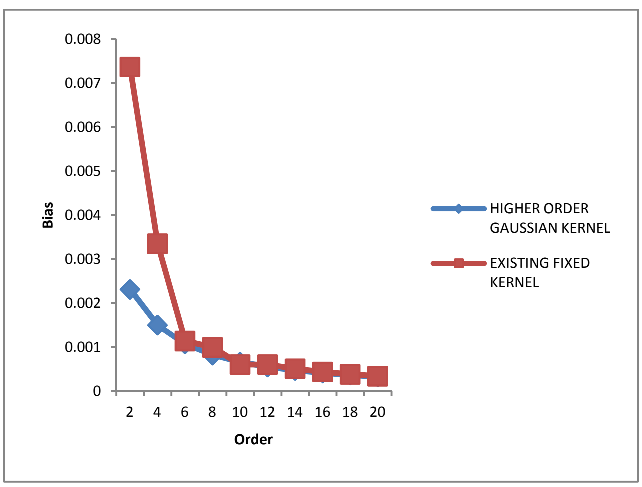

Fig 3.3a: Graph Showing the Bias for Table 3

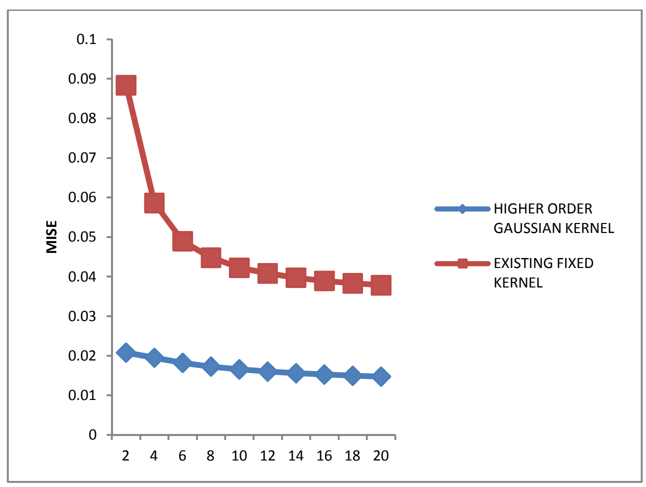

Fig 3.3b: Graph Showing the MISE for Table 3

References

[1]. Schapire, R. (1990) – The strength of weak learnability. Machine learn. **5**,pg 313 – 321.

[2]. Schapire, R. and Singer, Y. (1999) – Improved boosting algorithm using confidence rated prediction. Machine learn. **37**,pg 297 – 336.

[3]. Silverman, B. W. (1986). Density Estimation For Statistics and Data Analysis. London: Chapman and Hall.

[4]. Freund, Y. (1995). Boosting a weak learning algorithm by majority. Info. Comp. **121**,pg 256 - 285.

[5]. Mazio, D. M. and Taylor, C.C.(2004) – Boosting Kernel Density estimates: A bias reduction technique? Biometrika **91**, pg 226 - 233.

[6]. Ishiekwene, C.C; Ogbonmwan, S.M and Osemwenkhae, J.E. (2008) – A Meshsize Boosting Algorithm in Kernel Density Estimation. Journal of Science & Technology, Ghana. Vol **28** # 2,pg 69 – 74.

[7]. Duffy, N. and Hemlbold, D. (2000) – Potential bosters? Advances in Neural info. Proc. Sys. **12**, 258 – 264.

[8]. Ratsch, G. Mika, S., Scholkopf, B. and Muller, K. K. (2000) – Construction boosting algorithms from SVM's: An application to-one-class classification. GMD tech report no. 1.

[9]. Hazelton, M.L & Turlach,B.A (2007) – Reweighted KDE. Computational Statistics & Data Analysis,Vol **51**,issue 6,March 2007,pg 3057 – 3069.

[10]. Parzen, E. (1962) – On the estimation of a probability function and the nodes. Annals of Mathematical Statistics. **33**,pg 1065 – 1076.

[11]. Bartlett, M.S. (1963) – Statistical estimation of density functions. Sankhyā series A, **25**, pg 245 – 254.

[12]. Schucany, W.R. and Sommers, J.P. (1977) – Improvement of kernel-type density estimators. Journal of American Statistical Assoc. **72**, pg 420 -423.

[13]. Birke, Melanie (2009) – Shape constrained KDE. Journal of Statistical planning & inference, vol **139,** issue 8 , August 2009, pg 2851 – 2862.

[14]. Ishiekwene, C.C; Odiase, J.I and Ogbonmwan, S.M. (2007) - Bootstrap in Boosting Kernel Density Estimates. Int'l Journal of Natural & Applied Sciences,Vol 3 # 4, 531 – 536.

[15]. Ishiekwene, C.C and Nwelih, E. (2011a) – Adaptive Kernel in Meshsize Boosting Algorithm in KDE. African Research Review, Ethiopia. Vol 5(1), pg 438-446.

[16]. Ishiekwene, C.C and Nwelih, E. (2011b) – Adaptive Kernel in Bootstrap Boosting Algorithm in KDE. Journal of the Nig. Assoc. of Mathematical Physics. Vol 18, pg 363 – 366.

[17]. Afere, B.A.E (2010). A family of Multivariate Higher Order Hybrid Polynomial Kernels in Kernel Density Estimation. An unpublished Ph.D thesis submitted to the School of Postgraduate Studies, University of Benin, Benin City, Nigeria.

[18]. Ishiekwene, C.C and Afere, B.A.E. (2001) – Higher Order Window Width Selectors for Empirical Data. Journal of Nigerian Statistical Association (J.N.S.A), **14**, 69 – 82.

[19]. Ishiekwene, C. C. and Osemwenkhae, J. E. (2006) – A comparison of fourth order window width selectors in Kernel Density Estimation (A Univarate case), ABACUS, **33**; 14 - 20.