EXPLORING THE IMPACT OF POST-TRAINING ROUNDING IN REGRESSION MODELS

Jan Kalina, Praha

Received April 21, 2023. Published online February 15, 2024.

Abstract. Post-training rounding, also known as quantization, of estimated parameters stands as a widely adopted technique for mitigating energy consumption and latency in machine learning models. This theoretical endeavor delves into the examination of the impact of rounding estimated parameters in key regression methods within the realms of statistics and machine learning. The proposed approach allows for the perturbation of parameters through an additive error with values within a specified interval. This method is elucidated through its application to linear regression and is subsequently extended to encompass radial basis function networks, multilayer perceptrons, regularization networks, and logistic regression, maintaining a consistent approach throughout.

Keywords: supervised learning; trained model; perturbations; effect of rounding; low-precision arithmetic

MSC 2020: 68Q87, 62M45, 62H12

1. Introduction

This study is dedicated to theoretical examinations of the influence of rounding parameters in fitted (trained) regression models within the domains of statistics and machine learning. Special emphasis is placed on the research of rounding (quantization) during the training process of neural networks [4]. Quantization of weights aims to represent the weights by an efficient low-precision encoding and reducing the numerical precision of the weights (i.e., rounding) represents one of popular ways of reducing the energy and/or latency of neural networks [10]. Various sophisticated adaptive approaches to rounding have been proposed for the context of low-energy

This research was financially supported by the Czech Science Foundation project 22-02067S ("AppNeCo: Approximate Neurocomputing").
Open access publishing supported by the National Technical Library in Prague.

DOI: 10.21136/AM.2024.0090-23

computation [13], which has also allowed to use specific tailor-made hardware accelerators with explicit support for sparse computations [24].

Post-training quantization (PTQ) of pre-trained statistical learning algorithms has acquired much less attention compared to rounding within the training [28]. Still, PTQ is very important e.g., for computation on mobile phones with limited energy resources. More generally, the effect of rounding is omnipresent in computational intelligence, because only a finite number of digits may be stored in computer memory replacing any real numbers by rounded or truncated approximations. A complete state of the art in this field was presented in [15]. The effect on the predictive abilities of the methods was investigated in numerical experiments, which revealed PTQ to be much less harmful compared to quantization within the training [26]. The rounding in the experiments corresponds to using a fixed-point arithmetic, which is known to be more effective than floating-point arithmetic [4]. PTQ is also one of important approaches to model compression (sparsification) of a trained neural network [16]. Nevertheless, no theoretical results seem available for evaluating the effect of PTQ.

Low-precision arithmetic has recently been studied (only) empirically in various machine learning algorithms. To give some examples, it was investigated within an efficient implementation of Gaussian processes in [14] or in optimization tools for Bayesian deep learning based on stochastic gradient Markov chain Monte Carlo [29]. Floating-point arithmetic has already been implemented in hardware accelerators produced by NVIDIA, which is the world's dominant supplier of graphics processing units (GPU), and its rounding effects were empirically explored, e.g., in [6].

Our work belongs to the context of low-energy (approximate) computing [4], [24] obtained by rounding or by random perturbations of the estimated parameters. Theoretical expressions for the influence of rounding the estimated parameters are presented for various common models of statistics and machine learning. This work aims to evaluate the effect of rounding the parameters on the results (predictions) of various trained models, starting with explaining the ideas on linear regression in Section 2. Section 3 considers some common types of neural networks (radial basis function networks, multilayer perceptrons, or regularization networks) for nonlinear regression. Section 4 investigates the effect of rounding on the parameters of logistic regression and Section 5 brings conclusions.

2. Linear regression

Firstly, perturbations of parameters will be studied in the standard linear regression model

$$Y_i = \beta_1 X_{i1} + \ldots + \beta_p X_{ip} + e_i, \quad i = 1, \ldots, n, \tag{2.1}$$

which may be expressed in the matrix notation as $\mathbf{Y} = \mathbb{X}\boldsymbol{\beta} + \mathbf{e}$. The errors $e_1, \ldots, e_n$ are independent identically distributed random variables, the regressors (predictors) are fixed p-variate vectors, and $Y_1, \ldots, Y_n$ is a continuous random variable. The ith row of $\mathbb{X}$ will be denoted as $\mathbf{X}_i = (X_{i1}, \ldots, X_{ip})^\top$ for $i = 1, \ldots, n$. The estimation of $\boldsymbol{\beta} \in \mathbb{R}^p$ is usually considered under the assumptions of the classical linear regression model [8], i.e., with $\mathsf{E}\mathbf{e} = 0$, $\operatorname{var}\mathbf{e} = \sigma^2 I$ for a $\sigma > 0$, $\mathsf{E}\mathbb{X}^\top\mathbf{e} = 0$, and $\operatorname{rank}(X) = p$. Here, we assume to already have a fixed estimate $\widehat{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}$ so that these assumptions related to the estimation are not relevant. We stress that $\widehat{\boldsymbol{\beta}}$ may be any estimate, i.e., not necessarily the least squares estimate.

Let us assume that we make predictions for an observation with known regressors $\mathbf{Z} \in \mathbb{R}^p$, which is not necessarily in the training dataset, with a rounded (or modified in general) version of $\widehat{\boldsymbol{\beta}}$. The rounding is connected to approximate computing and may save substantial computational demands especially for a large p, see [9]. Thus, we consider an alternative vector of parameters in the form $\widetilde{\boldsymbol{\beta}} = \widehat{\boldsymbol{\beta}} + \boldsymbol{\tau}$, where $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_p)^\top$ with $\tau_j \in [-\varepsilon, \varepsilon]$, where $j = 1, \ldots, p$ and $\varepsilon > 0$ is fixed. The fitted value (predicted value) of the response in the model with $\widehat{\boldsymbol{\beta}}$ is denoted as $\widehat{Y}(\mathbf{Z}) = \mathbf{Z}^\top\widehat{\boldsymbol{\beta}} \in \mathbb{R}$ and in the model with $\widetilde{\boldsymbol{\beta}}$ is denoted as $\widetilde{Y}(\mathbf{Z}) = \mathbf{Z}^\top\widetilde{\boldsymbol{\beta}}$. We are interested in expressing $\widetilde{Y}(\mathbf{Z})$ in dependence on $\widehat{Y}(\mathbf{Z})$; the result immediately follows from Lemma 6.1 presented in Appendix.

Lemma 2.1. *In the standard linear regression model* (2.1)*, let* $\widehat{\boldsymbol{\beta}}$ *represent a given estimate of* $\boldsymbol{\beta}$*. Let us have* $\widetilde{\boldsymbol{\beta}} = \widehat{\boldsymbol{\beta}} + \boldsymbol{\tau}$*, where* $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_p)^\top$ *with* $\tau_j \in [-\varepsilon, \varepsilon]$*. Let us denote* $\boldsymbol{\varepsilon}_p = (\varepsilon, \ldots, \varepsilon)^\top \in \mathbb{R}^p$*. Then the fitted values of the response* $\widetilde{Y}(\mathbf{Z})$ *for* $\mathbf{Z} \in \mathbb{R}^p$ *fulfil*

$$\widetilde{Y}(\mathbf{Z}) \in [\mathbf{Z}^\top\widehat{\boldsymbol{\beta}} - |\mathbf{Z}|^\top\boldsymbol{\varepsilon}_p,\ \mathbf{Z}^\top\widehat{\boldsymbol{\beta}} + |\mathbf{Z}|^\top\boldsymbol{\varepsilon}_p]. \tag{2.2}$$

We note that all the results of the paper are formulated without assuming ε to be infinitesimally small. In addition, it is worth noting that the result (just like all the results throughout the paper) are derived using deterministic steps, i.e., they are valid surely. This is because the random variable $\widetilde{Y}(\mathbf{Z}) = Z^\top\widehat{\boldsymbol{\beta}}$ is contained also in the bounds of the interval of (2.2). The same would be true if the fixed regressors are replaced by random ones. Let us also state explicitly that estimates with a hat (as in $\widehat{\boldsymbol{\beta}}$) denote given estimates and estimates with a tilde (as in $\widetilde{\boldsymbol{\beta}}$) denote rounded estimates throughout this whole paper.

2.1. Illustrative example. Lemma 2.1 will be illustrated on a dataset with a single regressor, which can be easily visualized. The data shown in Figure 1 are levels of two proteins in the blood of hypothyroidism patients analyzed previously in [12]. The model

$$Y_i = \beta_0 + \beta_1 X_i + e_i, \quad i = 1, \dots, n, \tag{2.3}$$

with $n = 22$ is considered here. The least squares fit is shown as the black line in Figure 1 and the bounds (2.2) computed for $\varepsilon = 0.02$ are the red lines. We can see the bounds to become wider together with the increasing regressor.

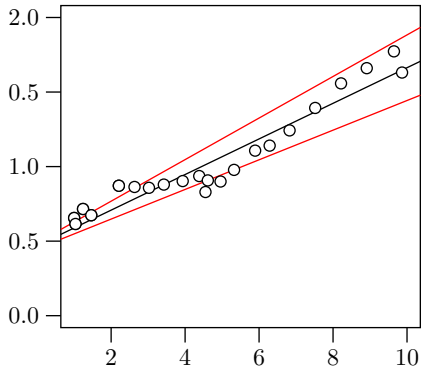

Figure 1. The least squares fit for the dataset of Section 2.1 together with the lower and upper bounds given by formula (2.2). The value $\varepsilon = 0.020$ was used here.

To express (2.2) for the specific model (2.3), let us take an observation with the value of the regressor equal to $z \in \mathbb{R}$. Because of the intercept in (2.3), we have $\mathbf{Z} = (1, z)^\top$, $\mathbf{Z}^\top \widehat{\beta} = \widehat{\beta}_0 + \widehat{\beta}_1 z$, and $|\mathbf{Z}|^\top \varepsilon_p = (1 + |z|)\varepsilon$. Using simple algebra, the result (2.2) can be expressed as

$$\widetilde{Y}(\mathbf{Z}) \in [\widehat{\beta}_0 - \varepsilon + z(\widehat{\beta}_1 - \varepsilon \operatorname{sgn}(z)),\ \widehat{\beta}_0 + \varepsilon + z(\widehat{\beta}_1 + \varepsilon \operatorname{sgn}(z))]. \tag{2.4}$$

The interval is the narrowest for $z = 0$ with the width 2ε and increases for data with large absolute values of the regressor. Thus, the observations that are outlying on the horizontal axis could be very influential in the rounded model. This brings a strong argument in favor of using robust estimators in the rounded models.

The value of $\varepsilon = 0.020$ used in Figure 1 corresponds to quite severe rounding and is used here to illustrate how the interval of (2.2) gets wider with the increasing regressor. The result of Lemma 2.1 is not primarily intended to reveal the robustness of linear regression but is rather aimed at evaluating the effect of perturbations as such. While the effect of rounding on predictions will often be quite small in applications, the theoretical result (2.2) is evaluated for the most extreme situation and presents

the widest possible interval in the given context. The width of the interval (2.2) remains exactly the same if the least squares estimator is replaced by an alternative estimator, for example by the least weighted squares (LWS) estimator [25], [11], which is highly robust to outliers and at the same time efficient for noncontaminated models with normally distributed errors.

In addition, we present a simulation revealing the effect of rounding the estimates on the prediction ability of the regression estimators. Such effect is not studied in Lemma 2.1, but prediction is often the very aim of the regression analysis. Using the same data and model (2.3), the estimate $(\widehat{\beta}_1, \widehat{\beta}_2)^\top$ was 1000-times replaced by $(\widehat{\beta}_1+\tau_1, \widehat{\beta}_2+\tau_2)^\top$, where τ_1 and τ_2 are independent random variables generated from uniform distribution $\mathsf{U}(-\varepsilon, \varepsilon)$. Leave-one-out cross-validation was performed for each situation and the resulting mean square error (MSE) averaged across the individual choices of τ_1 and τ_2 is presented for the least squares and the LWS with linearly decreasing weights in Table 1. The least squares estimator is based on minimizing MSE, but it is still outperformed here by the LWS as a consequence of using the cross-validation and because of the rounding. On the whole, the results are not in contradiction with general recommendations to use robust estimators for real data; if p is large, regularized estimators (such as the lasso or a robust version of [23]) may be suitable and the bounds of Lemma 2.1 are valid also for them.

	Estimator	
ε	LS	LWS
0.000	0.011	0.012
0.002	0.011	0.013
0.004	0.012	0.013
0.006	0.013	0.014
0.008	0.015	0.015
0.010	0.019	0.018
0.020	0.032	0.027

Table 1. Prediction mean square error (MSE) evaluated in a simulation in a leave-one-out cross validation for the least squares (LS) and the LWS in the example of Section 2.1 with different values of ε.

3. Model of nonlinear regression

Let us consider several popular machine learning methods for the task of nonlinear regression modeling. Estimating and predicting a nonlinear trend of an observed continuous variable represents an important task with various applications in signal

and image processing, engineering, biomedicine, economics, etc. [18]. The aim is to model (estimate and predict) the unknown continuous nonlinear regression function

$$Y_i = f(\mathbf{X}_i) + e_i \quad \text{for } i = 1, \dots, n, \tag{3.1}$$

based on the given observed data, where $e_1, \dots, e_n$ are independent and identically distributed random variables (errors). We assume a continuous response variable $\mathbf{Y} = (Y_1, \dots, Y_n)^\top$ and fixed p-variate regressors, where the regressor corresponding to the ith observation is denoted as $\mathbf{X}_i = (X_{i1}, \dots, X_{ip})^\top$. The model (3.1) encompasses various types of artificial neural networks as special cases. For several common types, we study the effect of perturbing the estimated parameters on the fitted (predicted) value of $\mathbf{Z} \in \mathbb{R}^p$ denoted as $\hat{f}(\mathbf{Z})$. Nevertheless, methods of Sections 3.1 and 3.2 work also for the classification task and the obtained results are valid for such context as well.

3.1. Radial basis function networks. Radial basis function (RBF) networks for the model (3.1) contain an input layer with p inputs, a single hidden layer with the total number N of RBF units (neurons), and a linear output layer. Using the notation of [2], the user chooses N together with a radially symmetric function (kernel) $\varrho\colon\ \mathbb{R}_0^+ \to \mathbb{R}_0^+$, where $\mathbb{R}_0^+ := \{x \in \mathbb{R};\ x \geqslant 0\}$. The model for the RBF network has the form

$$f(\mathbf{x}) = \sum_{j=1}^{N} a_j \varrho((\mathbf{x} - \mathbf{c}_j)^\top (\mathbf{x} - \mathbf{c}_j)), \quad \mathbf{x} \in \mathbb{R}^p, \tag{3.2}$$

with parameters $\mathbf{c}_1, \dots, \mathbf{c}_N \in \mathbb{R}^p$ and $a_1, \dots, a_N \in \mathbb{R}$, and possibly with other parameters corresponding to the (typically Gaussian) kernel ϱ. The presented property is also valid for robust, regularized, or robust regularized versions of RBF networks [20].

Lemma 3.1. *In the nonlinear regression model* (3.1)*, let* $\widehat{a}_1, \dots, \widehat{a}_N$ *and* $\widehat{\mathbf{c}}_1, \dots, \widehat{\mathbf{c}}_N$ *be given estimates of the parameters of a given trained RBF network with a kernel* ϱ *that is nonincreasing on* $\mathbb{R}_0^+$*. Let* $\hat{f}(\mathbf{Z})$ *be the predicted value of the response obtained using these estimates for* $\mathbf{Z} \in \mathbb{R}^p$*. Let us have* $\widetilde{a}_j = \widehat{a}_j + \theta_j$ *and* $\widetilde{\mathbf{c}}_j = \widehat{\mathbf{c}}_j + \boldsymbol{\tau}_j$ *for* $j = 1, \dots, N$*, where* $\boldsymbol{\tau}_j = (\tau_{j1}, \dots, \tau_{jp})^\top \in \mathbb{R}^p$*,* $\tau_{jk} \in [-\varepsilon, \varepsilon]$ *and* $\theta_j \in [-\varepsilon, \varepsilon]$ *for* $j = 1, \dots, N$*,* $k = 1, \dots, p$*, and* $\varepsilon > 0$*. Let us denote* $\boldsymbol{\chi}_j = |\mathbf{Z}| + |\mathbf{c}_j| + \boldsymbol{\varepsilon}_p$*. For* $\mathbf{Z} \in \mathbb{R}^p$*, the fitted value* $\tilde{f}(\mathbf{Z})$ *obtained with the RBF network with parameters* $\widetilde{a}_j$ *and* $\widetilde{\mathbf{c}}_j$ *for* $j = 1, \dots, N$ *fulfils*

$$\tilde{f}(\mathbf{Z}) \in \left[\sum_{j=1}^{N} (\widehat{a}_j - \varepsilon) \varrho((\mathbf{Z} - \mathbf{c}_j)^\top (\mathbf{Z} - \mathbf{c}_j) - \boldsymbol{\chi}_j^\top \boldsymbol{\varepsilon}_p) \sum_{j=1}^{N} (\widehat{a}_j + \varepsilon) \varrho((\mathbf{Z} - \mathbf{c}_j)^\top (\mathbf{Z} - \mathbf{c}_j) + \boldsymbol{\chi}_j^\top \boldsymbol{\varepsilon}_p) \right]. \tag{3.3}$$

3.2. Multilayer perceptrons. Multilayer perceptrons (MLPs) contain an input layer, one or more hidden layers with a fixed number of neurons, and an output layer. Here, f in model (3.1) is estimated using an MLP with a single hidden layer and using the notation following [2], although other more or less different versions of notation may be used in this context as well. The MLP estimates the response Y_i of the ith observation of the training set by

$$\widehat{Y}_i = g\Bigg(\sum_{k=1}^{N} \gamma_k h\Bigg(\sum_{j=1}^{p} \omega_{kj} X_{ij} + \omega_{k0}\Bigg) + \gamma_0\Bigg) + c, \quad i = 1, \dots, n, \tag{3.4}$$

where g and h must be specified (possibly nonlinear) functions. For a trained MLP, let us have the estimates of all the parameters

$$(c, \gamma_0, \gamma_1, \dots, \gamma_N, \omega_{10}, \dots, \omega_{N0}, \omega_{11}, \dots, \omega_{N1}, \dots, \omega_{1p}, \dots, \omega_{Np})^\top. \tag{3.5}$$

It is common to use nondecreasing activation functions and to choose g as the identity. The formula (3.4) for computing the fitted values of the response can be generalized for networks with more layers in a straightforward way. Essentially, the linear combination of the features also represents a fundamental element of convolutional neural networks (CNNs) [27].

Lemma 3.2. *In the nonlinear regression model* (3.1)*, let us consider a trained multilayer perceptron* (3.4) *with nondecreasing functions g and h. Let us denote its estimated parameters by*

$$(\widehat{c}, \widehat{\gamma}_0, \widehat{\gamma}_1, \dots, \widehat{\gamma}_N, \widehat{\omega}_{10}, \dots, \widehat{\omega}_{N0}, \widehat{\omega}_{11}, \dots, \widehat{\omega}_{N1}, \dots, \widehat{\omega}_{1p}, \dots, \widehat{\omega}_{Np})^\top \tag{3.6}$$

and the predicted value for $\mathbf{Z} \in \mathbb{R}^p$ by $\hat{f}(\mathbf{Z})$. Let us have

$$\begin{aligned} \widetilde{\gamma}_k &= \widehat{\gamma}_k + \tau_k, \quad k = 0, 1, \dots, N, \\ \widetilde{c} &= \widehat{c} + \tau_{-1}, \\ \widetilde{\omega}_{kj} &= \widehat{\omega}_{kj} + \theta_{kj}, \quad k = 1, \dots, N,\ j = 1, \dots, p, \end{aligned} \tag{3.7}$$

where

$$-\varepsilon \leqslant \tau_k \leqslant \varepsilon, \quad k = -1, 0, \dots, N, \quad -\varepsilon \leqslant \theta_{kj} \leqslant \varepsilon, \quad k = 1, \dots, N,\ j = 1, \dots, p, \tag{3.8}$$

with $\varepsilon > 0$. The fitted value $\tilde{f}(\mathbf{Z})$ obtained with the MLP with parameters (3.7) *then fulfils*

$$\tilde{f}(\mathbf{Z}) \in \Bigg[g\Bigg(\sum_{k=1}^{N} (\widehat{\gamma}_k - \varepsilon) h(\widehat{\delta}_{ik}^{-}) + \widehat{\gamma}_0 - \varepsilon\Bigg) + \widehat{c} - \varepsilon,\ g\Bigg(\sum_{k=1}^{N} (\widehat{\gamma}_k + \varepsilon) h(\widehat{\delta}_{ik}^{+}) + \widehat{\gamma}_0 + \varepsilon\Bigg) + \widehat{c} + \varepsilon\Bigg], \tag{3.9}$$

where

$$\widehat{\delta}_{ik}^{+} = \sum_{j=1}^{p} \widehat{\omega}_{kj} Z_j + \varepsilon \sum_{j=1}^{p} |Z_j| + \widehat{\omega}_{k0} + \varepsilon, \quad i = 1, \ldots, n, \tag{3.10}$$

and

$$\widehat{\delta}_{ik}^{-} = \sum_{j=1}^{p} \widehat{\omega}_{kj} Z_j - \varepsilon \sum_{j=1}^{p} |Z_j| + \widehat{\omega}_{k0} - \varepsilon, \quad i = 1, \ldots, n. \tag{3.11}$$

If specifically g is the identity and h is the ReLU function, (3.9) *reduces to*

$$\tilde{f}(\mathbf{Z}) \in \Big[\sum_{k=1}^{N} (\widehat{\gamma}_k - \varepsilon) \widehat{\delta}_{ik}^{-} + \widehat{\gamma}_0 - 2\varepsilon, \sum_{k=1}^{N} (\widehat{\gamma}_k + \varepsilon) \widehat{\delta}_{ik}^{+} + \widehat{\gamma}_0 + 2\varepsilon \Big]. \tag{3.12}$$

To recall, the ReLU (rectified linear unit) function considered in (3.12) is defined as $h(x) = \max\{0, x\}$ for $x \in \mathbb{R}$. The presented results are valid also for robust, regularized, or robust regularized versions of MLPs [5].

3.3. Regularization networks. Regularization networks [17] may be characterized as tools for nonlinear regression (3.1) with a clear interpretation and a straightforward computation. They are distinct from regularized neural networks [2], where the latter can be described simply as regularized versions of any models of neural networks. Regularization networks can be derived in the context of reproducing Hilbert kernel spaces (RKHS). Briefly, the method exploits a chosen reproducing kernel, which is a function $\kappa\colon\ \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ aimed to produce the symmetric matrix $\mathbb{K} \in \mathbb{R}^{n \times n}$ with values $K_{ij} = \kappa(\mathbf{X}_i, \mathbf{X}_j)$ for $i, j = 1, \ldots, n$. Most commonly, the function κ is chosen as the Gaussian kernel so that

$$K_{ij} = \kappa(\mathbf{X}_i, \mathbf{X}_j) := \exp\Big(-\frac{\|\mathbf{X}_i - \mathbf{X}_j\|^2}{2\sigma^2}\Big), \tag{3.13}$$

where $\|\cdot\|$ denotes the Euclidean distance. The fixed parameter $\sigma > 0$ can be estimated from the data.

The fitted value of the response for a new observation $\mathbf{Z} \in \mathbb{R}^p$ is obtained using

$$\hat{f}(\mathbf{Z}) = \sum_{i=1}^{n} \widehat{\alpha}_i \kappa(\mathbf{Z}, \mathbf{X}_i), \tag{3.14}$$

where $\widehat{\boldsymbol{\alpha}} \in \mathbb{R}^n$ is an estimate of the parameter $\boldsymbol{\alpha} \in \mathbb{R}^n$. A standard estimate of $\boldsymbol{\alpha}$ allows an explicit expression

$$\widehat{\boldsymbol{\alpha}} = (\mathbb{K}^\top \mathbb{K} + \lambda \mathbb{K})^{-1} \mathbb{K}^\top \mathbf{Y} = (\mathbb{K} + \lambda \mathbb{I}_n)^{-1} \mathbf{Y}, \tag{3.15}$$

where $\mathbb{I}_n \in \mathbb{R}^{n\times n}$ is the unit matrix. The regularization parameter $\lambda > 0$ may be found by cross-validation. The formula (3.15) corresponds to the ridge estimator for the linear regression model (2.1) in the form $\mathbf{Y} = \mathbb{K}\boldsymbol{\alpha} + \mathbf{e}$ and the estimator (3.15) is therefore commonly denoted as the generalized ridge estimator.

Lemma 3.3. *In the nonlinear regression model* (3.1)*, let* (3.14) *represent the regularization network prediction of* $f(\mathbf{Z})$*, where* $\mathbf{Z} \in \mathbb{R}^p$ *and* $\widehat{\boldsymbol{\alpha}}$ *is the estimate* (3.15) *of* $\boldsymbol{\alpha}$*. Let us have* $\widetilde{\boldsymbol{\alpha}} = \widehat{\boldsymbol{\alpha}}+\boldsymbol{\tau}$*, where* $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_n)^\top$ *with* $\tau_i \in [-\varepsilon, \varepsilon]$ *for* $i = 1, \ldots, n$ *and* $\varepsilon > 0$*. Then the fitted values obtained with* $\widetilde{\boldsymbol{\alpha}}$ *fulfil*

$$\tilde{f}(\mathbf{Z}) \in \left[\hat{f}(\mathbf{Z}) - \varepsilon \sum_{i=1}^{n} \kappa(\mathbf{Z}, \mathbf{X}_i),\ \hat{f}(\mathbf{Z}) + \varepsilon \sum_{i=1}^{n} \kappa(\mathbf{Z}, \mathbf{X}_i)\right]. \tag{3.16}$$

4. Logistic regression

Logistic regression is commonly used for models with a binary response variable [1]. Let $\mathbf{Y} = (Y_1, \ldots, Y_n)^\top$ denote values of a random variable following Bernoulli distribution, i.e., a binary variable with values 1 or 0. The probability of $Y_i = 1$, denoted as $\pi_i(\mathbf{X}_i)$, is explained by $\mathbf{X}_i = (X_{i1}, \ldots, X_{ip})^\top$; the regressors are assumed to be fixed and may include continuous as well as discrete variables. The expectation of Y_i, which is dependent on the nonrandom regressors, is obtained as $\mathsf{E}Y_i = \mathsf{P}(Y_i = 1) = \pi_i(\mathbf{X}_i)$ for $i = 1, \ldots, n$ and the model considers

$$\log \frac{\pi(\mathbf{X}_i)}{1 - \pi(\mathbf{X}_i)} = \mathbf{X}_i^\top \boldsymbol{\beta}, \quad i = 1, \ldots, n. \tag{4.1}$$

Let us now have a given (arbitrary) estimate $\widehat{\boldsymbol{\beta}} = (\widehat{\beta}_1, \ldots, \widehat{\beta}_p)^\top$ of the parameter $\boldsymbol{\beta} \in \mathbb{R}^p$. For an observation with regressors $\mathbf{Z} \in \mathbb{R}^p$, which is not necessarily in the training data, the model estimates $\pi(\mathbf{Z}) := P(\mathbf{Z} = 1)$ by

$$\widehat{\pi}(\mathbf{Z}) = \frac{\exp\{\mathbf{Z}^\top \widehat{\boldsymbol{\beta}}\}}{1 + \exp\{\mathbf{Z}^\top \widehat{\boldsymbol{\beta}}\}}. \tag{4.2}$$

We are interested in the lower and upper bounds for the estimated probabilities $\widetilde{\pi}(\mathbf{Z})$ obtained for $\widetilde{\boldsymbol{\beta}} = \widehat{\boldsymbol{\beta}} + \boldsymbol{\tau}$, where $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_p)^\top$ with $\tau_j \in [-\varepsilon, \varepsilon]$. A generalization of the logistic function to more than 2 outcomes, i.e., the multinomial logistic regression, is known as the softmax function, which is often used in the output layer of classification neural networks [7].

Lemma 4.1. *In the logistic regression model* (4.1)*, let an estimate* $\widehat{\boldsymbol{\beta}}$ *of* $\boldsymbol{\beta}$ *be considered in the form* $\widetilde{\boldsymbol{\beta}} = \widehat{\boldsymbol{\beta}} + \boldsymbol{\tau}$*, where* $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_p)^\top$ *with* $\tau_j \in [-\varepsilon, \varepsilon]$ *for* $j = 1, \ldots, p$ *and a fixed* $\varepsilon > 0$*. Let us denote* $\boldsymbol{\varepsilon}_p = (\varepsilon, \ldots, \varepsilon)^\top \in \mathbb{R}^p$*. Then it holds that*

$$\widetilde{\pi}(\mathbf{Z}) \in \Big[\frac{\exp\{\mathbf{Z}^\top\widehat{\boldsymbol{\beta}} - |\mathbf{Z}|^\top\boldsymbol{\varepsilon}_p\}}{1+\exp\{\mathbf{Z}^\top\widehat{\boldsymbol{\beta}} + |\mathbf{Z}|^\top\boldsymbol{\varepsilon}_p\}}, \frac{\exp\{\mathbf{Z}^\top\widehat{\boldsymbol{\beta}} + |\mathbf{Z}|^\top\boldsymbol{\varepsilon}_p\}}{1+\exp\{\mathbf{Z}^\top\widehat{\boldsymbol{\beta}} - |\mathbf{Z}|^\top\boldsymbol{\varepsilon}_p\}}\Big]. \tag{4.3}$$

5. Conclusion

This paper explores the impact of small perturbations in estimated parameters on predictions across various regression methods. Our focus on perturbations in the parameters is different from that of the measurement error theory, which is based on error-in-variables (EIV) models with perturbations in the data [3], [21].

The results in the form of derived lower and upper bounds (typically) for the predicted values are derived for selected regression models, without using any properties of the given estimates of the parameters. As a consequence, the obtained formulas are valid not only for the most standard estimates, but for every possible alternatives estimates including estimates that are robust due to the presence of measurement errors and/or outliers in the data [19]. In other words, there turns out to be no opposition between the robustness to outliers and the effect of perturbations. It is argued in the illustration of Section 2.1 that robust estimators are actually much preferable for rounded models because the impact of rounding may be very large for observations that are outlying in the regressors. Besides, the lower and upper bounds are not affected by multicollinearity; this is an appealing property for machine learning models, where the level of multicollinearity is typically very severe.

The predictions depend on the perturbations in a continuous way, i.e., are obtained as continuous functions of ε. Here, we assume the random perturbations to be within an interval $[-\varepsilon, \varepsilon]$ for some $\varepsilon > 0$. Such approach corresponds to rounding the estimates, e.g., if the model is trained using some low-precision arithmetic. The obtained results may be exploited within the task to evaluate the effect of perturbations within convolutional neural networks (CNNs) [22], where the effect of rounding was intensively studied in numerical experiments. For example, tested pipelines for different approaches to rounding in the training of neural networks were provided in [15] together with experiments evaluating their accuracy degradation. Practical suggestions for saving the multiplier demands of CNNs using efficient floating-point formats and low-precision representations were compared in [4]. These papers (just like the current paper) are not focused on robustness issues of different rounding

schemes, but rather on the average effect of rounding, which is quite small, although the (theoretical) effect in the worst case may be quite severe. As a topic for future theoretical work, it remains to derive analogous relationships for CNNs; their fully connected layers correspond precisely to MLPs, but the convolutional layers represent very complex hierarchical structures [22]. There is void of theoretical investigations of the effect of rounding also on classification methods of multivariate statistics and machine learning.

6. Appendix

We start here by formulating a lemma helpful for deriving some of the expressions throughout the whole paper, and proceed with proving other lemmas.

Lemma 6.1. *Let us assume* $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_p)^\top \in \mathbb{R}^p$ *fulfilling* $\tau_j \in [-\varepsilon, \varepsilon]$ *for* $j = 1, \ldots, p$ *with* $\varepsilon > 0$. *Let us denote* $\boldsymbol{\varepsilon}_p = (\varepsilon, \ldots, \varepsilon)^\top \in \mathbb{R}^p$.

(a) *For fixed* $\mathbf{r} \in \mathbb{R}^p$ *and* $\mathbf{s} \in \mathbb{R}^p$, *it holds that*

$$\mathbf{r}^\top \mathbf{s} - |\mathbf{r}|^\top \boldsymbol{\varepsilon}_p \leqslant \mathbf{r}^\top (\mathbf{s} + \boldsymbol{\tau}) \leqslant \mathbf{r}^\top \mathbf{s} + |\mathbf{r}|^\top \boldsymbol{\varepsilon}_p. \tag{6.1}$$

(b) *For fixed* $\mathbf{r} \in \mathbb{R}^p$ *and* $\mathbf{s} \in \mathbb{R}^p$, *it holds that*

$$\begin{aligned}(\mathbf{r} - \mathbf{s})^\top (\mathbf{r} - \mathbf{s}) - (|\mathbf{r}|^\top + |\mathbf{s}|^\top + \boldsymbol{\varepsilon}_p^\top)\boldsymbol{\varepsilon}_p &\leqslant (\mathbf{r} - (\mathbf{s} + \boldsymbol{\tau}))^\top (\mathbf{r} - (\mathbf{s} + \boldsymbol{\tau})) \\ &\leqslant (\mathbf{r} - \mathbf{s})^\top (\mathbf{r} - \mathbf{s}) + (|\mathbf{r}|^\top + |\mathbf{s}|^\top + \boldsymbol{\varepsilon}_p^\top)\boldsymbol{\varepsilon}_p.\end{aligned} \tag{6.2}$$

(c) *For a fixed* $u \in \mathbb{R}$ *and* $v \in \mathbb{R}$, *it holds that*

$$(u - \varepsilon \operatorname{sgn}(v))(v - \varepsilon \operatorname{sgn}(u)) \leqslant (u + \tau)(v + \tau) \leqslant (u + \varepsilon \operatorname{sgn}(v))(v + \varepsilon \operatorname{sgn}(u)). \tag{6.3}$$

Proof of Lemma 6.1. The first two parts are straightforward. To obtain the third part, one can start by realizing that

$$\begin{aligned}(u + \tau)v &\geqslant (u + \varepsilon) \quad \text{if } v < 0, \\ (u + \tau)v &\geqslant (u - \varepsilon) \quad \text{if } v > 0.\end{aligned} \tag{6.4}$$

This can be expressed as $(u + \tau)v \geqslant (u - \varepsilon \operatorname{sgn}(v))v$ and analogous reasoning leads to $(u + \tau)v \leqslant (u + \varepsilon \operatorname{sgn}(v))v$. Combining these two expressions yields

$$(u - \varepsilon \operatorname{sgn}(v))v \leqslant (r + \tau)v \leqslant (u + \varepsilon \operatorname{sgn}(v))v, \tag{6.5}$$

which finally leads to (5.3) if combined with an analogous result

$$u(v - \varepsilon \operatorname{sgn}(v)) \leqslant u(v + \tau) \leqslant u(v + \varepsilon \operatorname{sgn}(u)). \tag{6.6}$$

□

P r o o f of Lemma 3.1. First, if only $\widehat{\mathbf{c}}_1, \dots, \widehat{\mathbf{c}}_N$ are replaced by $\widetilde{\mathbf{c}}_1, \dots, \widetilde{\mathbf{c}}_N$ retaining $\widehat{a}_1, \dots, \widehat{a}_N$ unchanged, we use (5.2) and the nonincreasing property of ϱ to obtain

$$\tilde{f}(\mathbf{Z}) \in \Big[\sum_{j=1}^{N} \widehat{a}_j \varrho((\mathbf{Z}-\widehat{\mathbf{c}}_j)^\top(\mathbf{Z}-\widehat{\mathbf{c}}_j)-\boldsymbol{\chi}_j^\top \boldsymbol{\varepsilon}_p), \sum_{j=1}^{N} \widehat{a}_j \varrho((\mathbf{Z}-\widehat{\mathbf{c}}_j)^\top(\mathbf{Z}-\widehat{\mathbf{c}}_j)+\boldsymbol{\chi}_j^\top \boldsymbol{\varepsilon}_p)\Big]. \tag{6.7}$$

Secondly, let us also prepare

$$\sum_{j=1}^{N} \theta_j \varrho((\mathbf{Z}-\widehat{\mathbf{c}}_j)^\top(\mathbf{Z}-\widehat{\mathbf{c}}_j)) \leqslant \varepsilon \sum_{j=1}^{N} \varrho((\mathbf{Z}-\widehat{\mathbf{c}}_j)^\top(\mathbf{Z}-\widehat{\mathbf{c}}_j)). \tag{6.8}$$

We are now interested in

$$\begin{aligned} \tilde{f}(\mathbf{Z}) &= \sum_{j=1}^{N} (\widehat{a}_j + \theta_j) \varrho((\mathbf{Z}-\widehat{\mathbf{c}}_j-\boldsymbol{\tau}_j)^\top(\mathbf{Z}-\widehat{\mathbf{c}}_j-\boldsymbol{\tau}_j)) \\ &\leqslant \sum_{j=1}^{N} (\widehat{a}_j + \varepsilon) \varrho((\mathbf{Z}-\widehat{\mathbf{c}}_j-\boldsymbol{\tau}_j)^\top(\mathbf{Z}-\widehat{\mathbf{c}}_j-\boldsymbol{\tau}_j)), \end{aligned} \tag{6.9}$$

which exploits (5.8), and using the first preparatory step (5.7) leads us now to the upper bound given in the lemma. Analogous reasoning leads to the lower bound for $\tilde{f}(\mathbf{Z})$. □

P r o o f of Lemma 3.2. First, if only values of $\widehat{\omega}_{kj}$ are replaced by $\widehat{\omega}_{kj} + \theta_{kj}$ for $k = 1, \dots, N$ and $j = 0, 1, \dots, p$, we obtain

$$\begin{aligned} \tilde{f}(\mathbf{Z}) &\leqslant g\Big(\sum_{k=1}^{N} \widehat{\gamma}_k h\Big(\sum_{j=1}^{p} (\widehat{\omega}_{kj} + \theta_{kj}) Z_j + \widehat{\omega}_{k0} + \theta_{k0}\Big) + \widehat{\gamma}_0\Big) + \widehat{c} \\ &\leqslant g\Big(\sum_{k=1}^{N} \widehat{\gamma}_k h\Big(\sum_{j=1}^{p} \widehat{\omega}_{kj} Z_j + \sum_{j=1}^{p} |Z_j| \varepsilon\Big) + \widehat{\omega}_{k0} + \varepsilon + \widehat{\gamma}_0\Big) + \widehat{c}, \end{aligned} \tag{6.10}$$

because f and g are nondecreasing. Second, if only values of $\widehat{\gamma}_k$ are replaced by $\widehat{\gamma}_k + \tau_k$ for $k = 0, 1, \dots, N$, we obtain

$$\begin{aligned} \tilde{f}(\mathbf{Z}) &= g\Big(\sum_{k=1}^{N} \widehat{\gamma}_k h(\widehat{\xi}_j) + \sum_{k=1}^{N} \tau_k h(\widehat{\xi}_j) + \widehat{\gamma}_0 + \tau_0\Big) + \widehat{c} \\ &\leqslant g\Big(\sum_{k=1}^{N} \widehat{\gamma}_k h(\widehat{\xi}_j) + \varepsilon \sum_{k=1}^{N} h(\widehat{\xi}_j) + \widehat{\gamma}_0 + \tau_0\Big) + \widehat{c} \\ &\leqslant g\Big(\sum_{k=1}^{N} (\widehat{\gamma}_k + \varepsilon) h(\widehat{\xi}_j) + \widehat{\gamma}_0 + \tau_0\Big) + \widehat{c}, \end{aligned} \tag{6.11}$$

where $\widehat{\xi}=\sum_{j=1}^{p}\widehat{\omega}_{kj}Z_j+\widehat{\omega}_{k0}$. Finally, using the notation

$$\widehat{\zeta}_k=\sum_{j=1}^{p}(\widehat{\omega}_{kj}+\theta_{kj})Z_j+\widehat{\omega}_{k0}+\theta_{k0},\quad k=1,\dots,N, \tag{6.12}$$

we can use the combination of (5.10) and (5.11) to obtain

$$\begin{aligned}\tilde{f}(\mathbf{Z}) &\leqslant g\Big(\sum_{k=1}^{N}\widehat{\gamma}_k h(\widehat{\zeta}_k)+\sum_{k=1}^{N}\tau_k h(\widehat{\zeta}_k)+\widehat{\gamma}_0+\varepsilon\Big)+\widetilde{c}\\ &\leqslant g\Big(\sum_{k=1}^{N}\widehat{\gamma}_k h(\widehat{\delta}_{ik}^{+})+\varepsilon\sum_{k=1}^{N}h(\widehat{\delta}_{ik}^{+})+\widehat{\gamma}_0+\varepsilon\Big)+\widehat{c}+\varepsilon,\end{aligned} \tag{6.13}$$

where the right-side is equal to the upper bound of (3.9). The lower bound is obtained by analogous steps. □

A c k n o w l e d g e m e n t. The author would like to thank Barbora Peštová for discussion and the anonymous referee for constructive advice.

References

[1] *A. Agresti*: Foundations of Linear and Generalized Linear Models. Wiley Series in Probability and Statistics. John Wiley & Sons, Hoboken, 2015. zbl MR

[2] *G. Blokdyk*: Artificial Neural Network: A Complete Guide. 5STARCooks, Toronto, 2021.

[3] *R. J. Carroll, D. Ruppert, L. A. Stefanski, C. M. Crainiceanu*: Measurement Error in Nonlinear Models: A Modern Perspective. Monographs on Statistics and Applied Probability 105. Chapman & Hall/CRC, Boca Raton, 2006. zbl MR doi

[4] *M. Croci, M. Fasi, N. J. Higham, T. Mary, M. Mikaitis*: Stochastic rounding: Implementation, error analysis and applications. R. Soc. Open Sci. *9* (2022), Article ID 211631, 25 pages. doi

[5] *E. Egrioglu, E. Bas, O. Karahasan*: Winsorized dendritic neuron model artificial neural network and a robust training algorithm with Tukey's biweight loss function based on particle swarm optimization. Granul. Comput. *8* (2023), 491–501. doi
[6] *M. Fasi, N. J. Higham, M. Mikaitis, S. Pranesh*: Numerical behavior of NVIDIA tensor cores. PeerJ Computer Sci. *7* (2021), Article ID e330, 19 pages. doi
[7] *F. Gao, B. Li, L. Chen, Z. Shang, X. Wei, C. He*: A softmax classifier for high-precision classification of ultrasonic similar signals. Ultrasonics *112* (2021), Article ID 106344, 8 pages. doi
[8] *W. H. Greene*: Econometric Analysis. Pearson Education, Harlow, 2018.
[9] *T. Hastie, R. Tibshirani, R. Wainwright*: Statistical Learning with Sparsity: The Lasso and Generalizations. Monographs on Statistics and Applied Probability 143. CRC Press, Boca Raton, 2015. zbl MR doi
[10] *T. Hoefler, D. Alistarh, T. Ben-Nun, N. Dryden, A. Peste*: Sparsity in deep learning: Pruning and growth for efficient inference and training in neural networks. J. Mach. Learn. Res. *22* (2021), Article ID 241, 124 pages. zbl MR
[11] *J. Kalina, J. Tichavský*: On robust estimation of error variance in (highly) robust regression. Measurement Sci. Rev. *20* (2020), 6–14. doi
[12] *J. Kalina, P. Vidnerová, L. Soukup*: Modern approaches to statistical estimation of measurements in the location model and regression. Handbook of Metrology and Applications. Springer, Singapore, 2023, pp. 2355–2376. doi
[13] *C. Louizos, M. Reisser, T. Blankevoort, E. Gavves, M. Welling*: Relaxed quantization for discretized neural networks. Available at `https://arxiv.org/abs/1810.01875` (2018), 14 pages. doi
[14] *W. J. Maddox, A. Potapczynski, A. G. Wilson*: Low-precision arithmetic for fast Gaussian processes. Proc. Mach. Learn. Res. *180* (2022), 1306–1316.
[15] *M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, T. Blankevoort*: A white paper on neural network quantization. Available at `https://arxiv.org/abs/2106.08295` (2021), 27 pages. doi
[16] *J.-H. Park, K.-M. Kim, S. Lee*: Quantized sparse training: A unified trainable framework for joint pruning and quantization in DNNs. ACM Trans. Embedded Comput. Syst. *21* (2022), Article ID 60, 22 pages. doi
[17] *G. Pillonetto*: System identification using kernel-based regularization: New insights on stability and consistency issues. Automatica *93* (2018), 321–332. zbl MR doi
[18] *H. Riazoshams, H. Midi, G. Ghilagaber*: Robust Nonlinear Regression with Applications Using R. John Wiley & Sons, Hoboken, 2019. zbl MR doi
[19] *A. K. M. E. Saleh, J. Picek, J. Kalina*: R-estimation of the parameters of a multiple regression model with measurement errors. Metrika *75* (2012), 311–328. zbl MR doi
[20] *A.-K. Seghouane, N. Shokouhi*: Adaptive learning for robust radial basis function networks. IEEE Trans. Cybernetics *51* (2021), 2847–2856. doi
[21] *K. S. Shultz, D. Whitney, M. J. Zickar*: Measurement Theory in Action: Case Studies and Exercises. Routledge, New York, 2020. doi
[22] *J. Šíma, P. Vidnerová, V. Mrázek*: Energy complexity model for convolutional neural networks. Artificial Neural Networks and Machine Learning – ICANN 2023. Lecture Notes in Computer Science 14263. Springer, Cham, 2023, pp. 186–198. doi
[23] *E. Smucler, V. J. Yohai*: Robust and sparse estimators for linear regression models. Comput. Stat. Data Anal. *111* (2017), 116–130. zbl MR doi
[24] *V. Sze, Y.-H. Chen, T.-J. Yang, J. S. Emer*: Efficient processing of deep neural networks: A tutorial and survey. Proc. IEEE *105* (2295–2329), 2017. doi
[25] *J.Á. Víšek*: Consistency of the least weighted squares under heteroscedasticity. Kybernetika *47* (2011), 179–206. zbl MR

[26] *N. Wang, J. Choi, D. Brand, C.-Y. Chen, K. Gopalakrishnan*: Training deep neural networks with 8-bit floating point numbers. NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems. Curran Associates, New York, 2018, pp. 7686–7695. doi

[27] *W. Q. Yan*: Computational Methods for Deep Learning: Theory, Algorithms, and Implementations. Texts in Computer Science. Springer, Singapore, 2023. zbl MR doi

[28] *J. Yu, M. Anitescu*: Multidimensional sum-up rounding for integer programming in optimal experimental design. Math. Program. *185* (2021), 37–76. zbl MR doi

[29] *R. Zhang, A. G. Wilson, C. De Sa*: Low-precision stochastic gradient Langevin dynamics. Proc. Mach. Learn. Res. *162* (2022), 26624–26644.

Author's address: *Jan Kalina*, The Czech Academy of Sciences, Institute of Computer Science, Pod Vodárenskou věží 271/2, 182 00 Praha 8, Czech Republic, e-mail: `kalina@cs.cas.cz`.