

# Quadratic programming optimization for Newton method

R. V. Isachenko, V. V. Strijov

**Abstract:** To optimize the model parameters the Newton method is widely used. This method is second order optimization procedure that is unstable in real applications. In this paper we propose the procedure to make the optimization process robust. The idea is to select the set of model parameters which have to be optimized in the current step of optimization procedure. We show that in the case of nonlinear regression and logistic regression models the parameters selection could be performed by Quadratic Programming Feature Selection algorithm. It allows to find the set of independent parameters that are responsible for the residuals. We carried out the experiment to show how the proposed method works and compare it with other methods.

The paper proposes the robust second-order optimization algorithm. The algorithm based on the iterative Newton method, which is unstable procedure. The authors suggest to select the set of active parameters in each optimization step. The algorithm updates only parameters from this active set. Quadratic programming feature selection is used to find the active set. It maximizes the relevance of model parameters to the residuals and minimizes the redundancy. Nonlinear regression and logistic regression models are investigated. The proposed algorithm achieves the less error with comparison to the other methods.

**Keywords:** nonlinear regression, logistic regression, Newton method, quadratic programming feature selection

## Introduction

The error function models have a complex landscape with multiple local minima for models with extremely large number of parameters. In this case the optimization algorithm brings to different solutions each time.

The optimization algorithm is an iterative process. In each step it updates the current point parameters to get the next approximation. There have been developed first-order optimization algorithms, which use the first derivatives of the error function. The most famous algorithms are Gradient Descent, Nesterov Momentum [1], Adagrad [2], Adam [3]. These algorithms are used for the deep neural networks optimization [4]. The Newton algorithm is the second-order algorithm, which uses the second derivatives of the error function. It finds updates for the quadratic approximation of the error function and converges with the lesser number of iterations. The drawback of the

second-order optimization methods is the huge and ill-conditioned Hessian matrix. The optimization process in this case is computationally expensive and diverge. [5, 6] propose the approximations for the Hessian matrix and regularization to overcome this problem. The paper [7] applies the Newton method to the deep neural networks.

This paper suggests to select the set of model parameters, which is optimized in each optimization step. The authors investigate the nonlinear regression model with the squared error function and the logistic regression model with the cross-entropy error function. For the nonlinear regression the Newton method and model linearization leads to the Gauss-Newton method. Each step solves the linear regression problem. The authors use the two-layer neural network as the nonlinear model. The Newton method for logistic regression brings to Iteratively Least Squares (IRLS) algorithm. Here the optimization step is made in the direction given also by the linear regression problem solution.

The paper proposes to apply the Quadratic Programming Feature Selection (QPFS) algorithm [8, 9] to select the optimal set of model parameters. The QPFS algorithm selects features for the linear regression problem. We have the linear regression problem for both model in each step. The QPFS algorithm tries to maximize the relevances of features and minimize pairwise dependency between features [10]. In our case it allows to find independent parameters which impact the model residuals the most.

The computational experiment investigates a behaviour of the proposed algorithm near the optimal point and compared the algorithm with the other methods such as Gradient Descent, Nesterov Momentum, ADAM and Newton algorithms.

## Problem Statement

The model  $f(\mathbf{x}, \mathbf{w})$ ,  $\mathbf{w} \in \mathbb{R}^p$  predicts the target variable  $y \in \mathbb{Y}$ , given the object  $\mathbf{x} \in \mathbb{R}^n$ . The space  $\mathbb{Y}$  equals  $\{0, 1\}$  for the binary classification problem and  $\mathbb{Y} = \mathbb{R}$  for the regression problem. There are given the design matrix  $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m]^\top \in \mathbb{R}^{m \times n}$  and the target vector  $\mathbf{y} = [y_1, \dots, y_m]^\top \in \mathbb{Y}^m$ . The goal is to find the optimal parameters  $\mathbf{w}^*$ . The parameters  $\mathbf{w}$  are fitted by the minimization of the error function:

$$\mathbf{w}^* = \arg \min_{\mathbf{w} \in \mathbb{R}^p} S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f). \quad (1)$$

The investigated choices for the error function  $S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f)$  are the squared error for the regression problem:

$$S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f) = \frac{1}{2} \|\mathbf{y} - \mathbf{f}(\mathbf{X}, \mathbf{w})\|_2^2 = \frac{1}{2} \sum_{i=1}^m (y_i - f(\mathbf{x}_i, \mathbf{w}))^2, \quad (2)$$

and the cross-entropy for the binary classification problem:

$$S(\mathbf{w}|\mathbf{X}, \mathbf{y}, f) = \sum_{i=1}^m [y_i \log f(\mathbf{x}_i, \mathbf{w}) + (1 - y_i) \log(1 - f(\mathbf{x}_i, \mathbf{w}))]. \quad (3)$$

The problem (1) is solved by iterative optimization procedures. To obtain parameters in the step  $k$ , the current parameters  $\mathbf{w}^{k-1}$  is updated by the rule

$$\mathbf{w}^k = \mathbf{w}^{k-1} + \Delta \mathbf{w}^{k-1}. \quad (4)$$

This paper suggests to use the Newton optimization procedure to select parameters updates  $\Delta \mathbf{w}$ .

The Newton method is unstable and computationally hard. Each iteration inverts the Hessian matrix  $\mathbf{H} = \nabla^2 S(\mathbf{w})$ . This paper suggests the robust Newton algorithm. Before the gradient step the authors propose to select the set of active model parameters, which have the greatest impact on the error function  $S(\mathbf{w})$ . We update only the parameters with indices from a set  $\mathcal{A} \subseteq \{1, \dots, p\}$

$$\begin{aligned}\mathbf{w}_{\mathcal{A}}^k &= \mathbf{w}_{\mathcal{A}}^{k-1} + \Delta \mathbf{w}_{\mathcal{A}}^{k-1}, & \mathbf{w}_{\mathcal{A}} &= \{w_j\}_{j \in \mathcal{A}}; \\ \mathbf{w}_{\bar{\mathcal{A}}}^k &= \mathbf{w}_{\bar{\mathcal{A}}}^{k-1}, & \mathbf{w}_{\bar{\mathcal{A}}} &= \{w_j\}_{j \notin \mathcal{A}}.\end{aligned}$$

To select the set  $\mathcal{A}$  from all possible  $2^p$  subsets we introduce the quality criteria

$$\mathcal{A} = \arg \max_{\mathcal{A}' \subseteq \{1, \dots, p\}} Q(\mathcal{A}' | \mathbf{X}, \mathbf{y}, f, \mathbf{w}). \quad (5)$$

The problem (5) is solved in each step  $k$  of the optimization process for the current parameters  $\mathbf{w}^k$ .

## Quadratic programming feature selection

If there are multicollinearity between columns of the design matrix  $\mathbf{X}$  the solution of the linear regression problem

$$\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 \rightarrow \min_{\mathbf{w} \in \mathbb{R}^n}. \quad (6)$$

is unstable. The feature selection methods finds the set  $\mathcal{A} \in \{1, \dots, n\}$  of representative columns in  $\mathbf{X}$ .

The goal of the QPFS is to select non-correlated features, which are relevant to the target vector  $\mathbf{y}$ . To formalise this approach let introduce two functions:  $\text{Sim}(\mathbf{X})$  and  $\text{Rel}(\mathbf{X}, \mathbf{y})$ . The  $\text{Sim}(\mathbf{X})$  measures the redundancy between features, the  $\text{Rel}(\mathbf{X}, \mathbf{y})$  contains relevances between each feature and target vector. We want to minimize the Sim function and maximize the Rel simultaneously.

QPFS offers the explicit way to construct the functions Sim and Rel. The method minimizes the following functional

$$\underbrace{\mathbf{a}^\top \mathbf{Q} \mathbf{a}}_{\text{Sim}} - \alpha \cdot \underbrace{\mathbf{b}^\top \mathbf{a}}_{\text{Rel}} \rightarrow \min_{\substack{\mathbf{a} \in \mathbb{R}_+^n \\ \|\mathbf{a}\|_1 = 1}}. \quad (7)$$

The matrix  $\mathbf{Q} \in \mathbb{R}^{n \times n}$  entries measure the pairwise similarities between features. The vector  $\mathbf{b} \in \mathbb{R}^n$  expresses the similarities between each feature and the target vector  $\mathbf{y}$ . The normalized vector  $\mathbf{a}$  shows the importance of each feature. The functional (7) penalizes the dependent features by the function Sim and encourages features relevant to the target by the function Rel. The parameter  $\alpha$  allows to control the trade-off between the Sim and the Rel terms. The authors of the original QPFS paper suggested the way to select  $\alpha$  and make Sim and Rel terms impact are equal

$$\alpha = \frac{\overline{\mathbf{Q}}}{\overline{\mathbf{Q}} + \overline{\mathbf{b}}},$$

where  $\bar{\mathbf{Q}}$ ,  $\bar{\mathbf{b}}$  are the mean values of  $\mathbf{Q}$  and  $\mathbf{b}$  correspondingly. Apply the thresholding for  $\mathbf{a}$  to find the optimal feature subset:

$$j \in \mathcal{A} \Leftrightarrow a_j > \tau.$$

To measure similarity the authors use sample correlation coefficient between pairs of features for the Sim function and between features and target vector for the Rel function. The problem (7) is convex if the matrix  $\mathbf{Q}$  is positive semidefinite. In general it is not always true. To satisfy this condition the matrix  $\mathbf{Q}$  spectrum is shifted and the matrix  $\mathbf{Q}$  is replaced by  $\mathbf{Q} - \lambda_{\min} \mathbf{I}$ , where  $\lambda_{\min}$  is a  $\mathbf{Q}$  minimal eigenvalue.

We use the QPFS algorithm to solve the problem (5). QPFS selects the set  $\mathcal{A}$  of parameters updates  $\Delta \mathbf{w}$ , which have the greatest impact to the residuals and pairwise independent. The functional (7) corresponds to the quality criteria  $Q(\mathcal{A}|\mathbf{X}, \mathbf{y}, f, \mathbf{w})$ .

$$\mathcal{A} = \arg \max_{\mathcal{A}' \subseteq \{1, \dots, p\}} Q(\mathcal{A}'|\mathbf{X}, \mathbf{y}, f, \mathbf{w}) \Leftrightarrow \arg \min_{\mathbf{a} \in \mathbb{R}_+^p, \|\mathbf{a}\|_1=1} [\mathbf{a}^\top \mathbf{Q} \mathbf{a} - \alpha \cdot \mathbf{b}^\top \mathbf{a}]. \quad (8)$$

To construct the matrix  $\mathbf{Q}$  and the vector  $\mathbf{b}$  we show that each optimization step is equivalent to the linear regression problem (6) for the nonlinear regression model with the squared error function (2) and the logistic regression model with the cross-entropy error function (3).

## Newton method

The Newton method uses the first order optimization condition for problem (1) and linearize the gradient of  $S(\mathbf{w})$

$$\nabla S(\mathbf{w} + \Delta \mathbf{w}) = \nabla S(\mathbf{w}) + \mathbf{H} \cdot \Delta \mathbf{w} = 0,$$

$$\Delta \mathbf{w} = -\mathbf{H}^{-1} \nabla S(\mathbf{w}).$$

where  $\mathbf{H} = \nabla^2 S(\mathbf{w})$  is the Hessian matrix of the error function  $S(\mathbf{w})$ .

The iteration (4) of the Newton method is

$$\mathbf{w}^k = \mathbf{w}^{k-1} - \mathbf{H}^{-1} \nabla S(\mathbf{w}).$$

Each iteration inverts the Hessian matrix. The measure of ill-conditioning for the Hessian matrix  $\mathbf{H}$  is the condition number

$$\kappa(\mathbf{H}) = \frac{\lambda_{\max}(\mathbf{H})}{\lambda_{\min}(\mathbf{H})},$$

where  $\lambda_{\max}(\mathbf{H})$ ,  $\lambda_{\min}(\mathbf{H})$  are the maximum and minimum eigenvalues of  $\mathbf{H}$ . The large condition number  $\kappa(\mathbf{H})$  leads to instability of the optimization process. The proposed algorithm reduces the size of the Hessian matrix  $\mathbf{H}$  and makes the condition number  $\kappa(\mathbf{H})$  smaller.

## Nonlinear regression

Assume that the model  $f(\mathbf{x}, \mathbf{w})$  is close to linear in the neighborhood of the point  $\mathbf{w} + \Delta \mathbf{w}$

$$\mathbf{f}(\mathbf{X}, \mathbf{w} + \Delta \mathbf{w}) \approx \mathbf{f}(\mathbf{X}, \mathbf{w}) + \mathbf{J} \cdot \Delta \mathbf{w},$$

where  $\mathbf{J} \in \mathbb{R}^{m \times p}$  is the Jacobian matrix

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f(\mathbf{x}_1, \mathbf{w})}{\partial w_1} & \cdots & \frac{\partial f(\mathbf{x}_1, \mathbf{w})}{\partial w_p} \\ \vdots & \ddots & \vdots \\ \frac{\partial f(\mathbf{x}_m, \mathbf{w})}{\partial w_1} & \cdots & \frac{\partial f(\mathbf{x}_m, \mathbf{w})}{\partial w_p} \end{pmatrix}. \quad (9)$$

Under this assumption the gradient  $\nabla S(\mathbf{w})$  and the Hessian matrix  $\mathbf{H}$  of the error function (2) equal

$$\nabla S(\mathbf{w}) = \mathbf{J}^\top (\mathbf{y} - \mathbf{f}), \quad \mathbf{H} = \mathbf{J}^\top \mathbf{J}. \quad (10)$$

It leads to the Gauss-Newton method and the update rule (4) is

$$\mathbf{w}_k = \mathbf{w}_{k-1} + (\mathbf{J}^\top \mathbf{J})^{-1} \mathbf{J}^\top (\mathbf{f} - \mathbf{y}).$$

The updates  $\Delta \mathbf{w}$  is the solution of the linear regression problem

$$\|\mathbf{z} - \mathbf{F} \Delta \mathbf{w}\|_2^2 \rightarrow \min_{\Delta \mathbf{w} \in \mathbb{R}^p}, \quad (11)$$

where  $\mathbf{z} = \mathbf{f} - \mathbf{y}$  and  $\mathbf{F} = \mathbf{J}$ .

We consider the feed-forward two layer neural network as the nonlinear model. In this case the model  $f(\mathbf{x}, \mathbf{w})$  is given by

$$f(\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{x}^\top \mathbf{W}_1) \mathbf{w}_2.$$

Here  $\mathbf{W}_1 \in \mathbb{R}^{n \times h}$  is the weight matrix which connects the input features with  $h$  hidden units. The nonlinearity function  $\sigma(\cdot)$  applied element-wise. The weights  $\mathbf{w}_2 \in \mathbb{R}^{h \times 1}$  connect the hidden units with the output. The model parameters  $\mathbf{w}$  is a concatenation of vectorized matrices  $\mathbf{W}_1, \mathbf{w}_2$ .

## Logistic Regression

For logistic regression the model  $f(\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{x}^\top \mathbf{w})$  with the sigmoid function  $\sigma(\cdot)$ . The gradient and the Hessian of the error function (3) equal

$$\nabla S(\mathbf{w}) = \mathbf{X}^\top (\mathbf{f} - \mathbf{y}), \quad \mathbf{H} = \mathbf{X}^\top \mathbf{R} \mathbf{X}, \quad (12)$$

where  $\mathbf{R}$  is a diagonal matrix with  $f(\mathbf{x}_i, \mathbf{w}) \cdot (1 - f(\mathbf{x}_i, \mathbf{w}))$  diagonal entries.

The update rule (4) is

$$\mathbf{w}^k = \mathbf{w}^{k-1} + (\mathbf{X}^\top \mathbf{R} \mathbf{X})^{-1} \mathbf{X}^\top (\mathbf{y} - \mathbf{f}).$$

This algorithm is known as Iteratively Reweighted Least Squares (IRLS) algorithm. The updates  $\Delta \mathbf{w}$  is the solution of the linear regression problem

$$\|\mathbf{z} - \mathbf{F} \Delta \mathbf{w}\|_2^2 \rightarrow \min_{\Delta \mathbf{w} \in \mathbb{R}^p}, \quad (13)$$

where  $\mathbf{z} = \mathbf{R}^{-1/2} (\mathbf{y} - \mathbf{f})$  and  $\mathbf{F} = \mathbf{R}^{1/2} \mathbf{X}$ .

## Step size

The step size of the Newton method could be excessively large. To control the step size of the weight updates we add the parameter  $\eta$  in the update rule (4)

$$\mathbf{w}^k = \mathbf{w}^{k-1} + \eta \Delta \mathbf{w}^{k-1}, \quad \eta \in [0, 1].$$

To select the appropriate step size  $\eta$  the Armijo rule is used. We choose  $\eta$  as large as possible to satisfy the following condition

$$S(\mathbf{w}^{k-1} + \eta \Delta \mathbf{w}^{k-1}) < S(\mathbf{w}^{k-1}) + \gamma \eta \nabla S^\top(\mathbf{w}^{k-1}) \mathbf{w}^{k-1}, \quad \gamma \in [0, 0.5].$$

## Algorithm

We propose to implement the QPFS algorithm to the problems (11) and (13). QPFS selects the set  $\mathcal{A}$  of parameters updates  $\Delta \mathbf{w}$ , which have the greatest impact to the residuals and pairwise independent. The QPFS matrix  $\mathbf{Q}$  and the vector  $\mathbf{b}$  are given by

$$\mathbf{Q} = \text{Sim}(\mathbf{F}), \quad \mathbf{b} = \text{Rel}(\mathbf{F}, \mathbf{z}).$$

The sample correlation coefficient is equal to zero for the orthogonal vectors. We show that in the optimal point  $\mathbf{w}^*$  the vector  $\mathbf{z}$  is orthogonal to the columns of the matrix  $\mathbf{F}$  for the considered problems. It leads to the QPFS vector  $\mathbf{b} = \mathbf{0}$ . First order optimization condition guarantees this property for the nonlinear regression problem

$$\mathbf{F}^\top \mathbf{z} = \mathbf{J}^\top(\mathbf{f} - \mathbf{y}) = -\nabla S(\mathbf{w}^*) = \mathbf{0},$$

and the logistic regression problem

$$\mathbf{F}^\top \mathbf{z} = \mathbf{X} \mathbf{R}^{-1/2} \mathbf{R}^{1/2}(\mathbf{y} - \mathbf{f}) = \mathbf{X}^\top(\mathbf{y} - \mathbf{f}) = \nabla S(\mathbf{w}^*) = \mathbf{0}.$$

The pseudocode of the algorithm is given in the algorithm (1).

## Experiment

The goal of the computational experiment is to explore properties of the proposed algorithm and compare it with the other methods.

We investigate the dependence of the QPFS parameters for the problems (11), (13). Assume that the parameter vector  $\mathbf{w}^0$  lies near the optimal parameters vector  $\mathbf{w}^*$ . We consider the line segment

$$\mathbf{w}_\beta = \beta \mathbf{w}^* + (1 - \beta) \mathbf{w}^0; \quad \beta \in [0, 1].$$

We generate the dataset with 300 samples and 7 features for the logistic regression problem. The landscape of the error function (3) on the two random selected parameters grid is shown in the figure 1a. The surface is convex with stretched level lines along some model parameters. Add the random noise to the optimum parameters  $\mathbf{w}^*$  to get the point  $\mathbf{w}^0$ . The behaviour of the Rel term vector  $\mathbf{b}$  on the line segment between  $\mathbf{w}^0$

---

**Algorithm 1** QPFS + Newton algorithm

---

**Require:**  $\varepsilon$  – tolerance;  
 $\tau$  – QPFS solution threshold;  
 $\gamma$  – Armijo rule parameter.

**Ensure:**  $\mathbf{w}^*$ ;  
initialize  $\mathbf{w}^0$ ;  
 $k := 1$ ;  
**repeat**  
  compute  $\mathbf{z}$  and  $\mathbf{F}$  for (11) or (13) ;  
   $\mathbf{Q} := \text{Sim}(\mathbf{F})$ ,  $\mathbf{b} := \text{Rel}(\mathbf{F}, \mathbf{z})$ ,  $\alpha = \frac{\bar{\mathbf{Q}}}{\mathbf{Q} + \mathbf{b}}$ ;  
   $\mathbf{a} := \arg \min_{\mathbf{a} \geq 0, \|\mathbf{a}\|_1 = 1} \mathbf{a}^\top \mathbf{Q} \mathbf{a} - \alpha \cdot \mathbf{b}^\top \mathbf{a}$ ;  
   $\mathcal{A} = \{j \mid a_j > \tau\}$ ;  
  compute  $\nabla S(\mathbf{w}^{k-1})$ ,  $\mathbf{H}$  from (10) or (12);  
   $\Delta \mathbf{w}^{k-1} = -\mathbf{H}^{-1} \nabla S(\mathbf{w}^{k-1})$ ;  
   $\eta := \text{ArmijoRule}(\mathbf{w}^{k-1}, \gamma)$ ;  
   $\mathbf{w}_{\mathcal{A}}^k = \mathbf{w}_{\mathcal{A}}^{k-1} + \eta \Delta \mathbf{w}_{\mathcal{A}}^{k-1}$ ;  
   $k := k + 1$ ;  
**until**  $\frac{\|\mathbf{w}^k - \mathbf{w}^{k-1}\|}{\|\mathbf{w}^k\|} < \varepsilon$

---

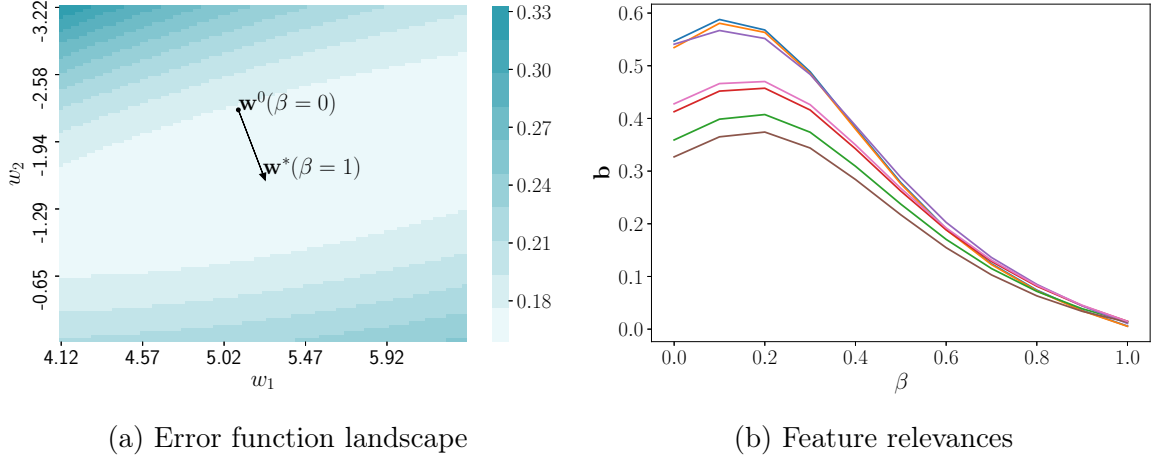


Figure 1: Logistic regression

and  $\mathbf{w}^*$  is illustrated in the figure 1b. The components of the vector  $\mathbf{b}$  starts to decrease sharply nearing the optimal point.

For nonlinear regression model we used the classical Boston Housing dataset with 506 objects and 13 features. The neural network contains two hidden neurons for simplicity. The error function landscape for the neural network model is more complex. It is not convex and could contain multiple local minimum. The two-dimensional error function landscape for this dataset is shown in the figure 2a. The grid is obtained by selecting two random weights from the matrix  $\mathbf{W}_1$ . We use the same strategy to investigate how the linear term vector  $\mathbf{b}$  is changing moving from  $\mathbf{w}^0$  to  $\mathbf{w}^*$ . The result is

shown in the figure 2b. The vector  $\mathbf{b}$  components decline near optimum. Reaching the optimum the different weights influence on the model residuals  $\mathbf{z}$ .

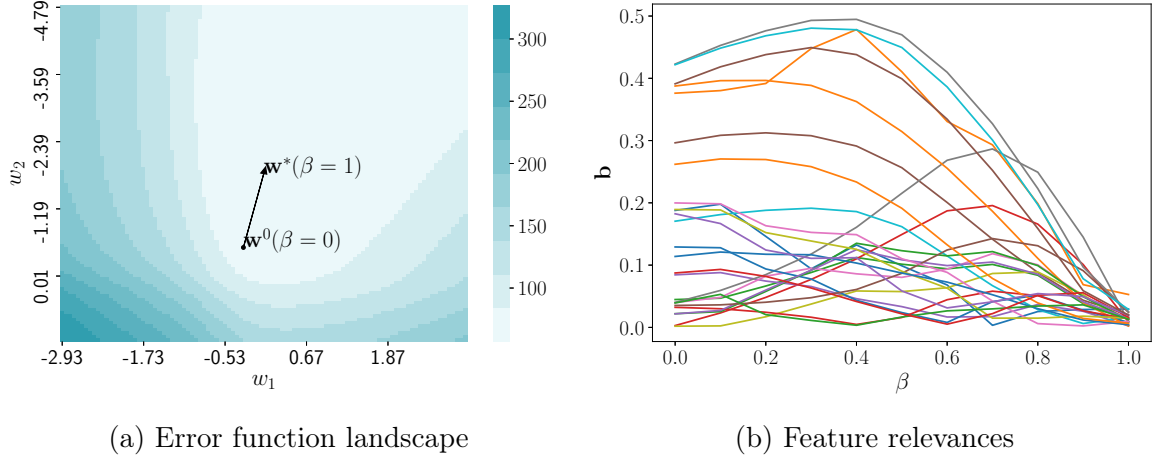


Figure 2: Neural network, first layer

Figure 3 shows the optimization process for the proposed procedure in the case of logistic regression with two model parameters. Even for two-dimensional problem the solution of Newton method is unstable and the condition number of Hessian matrix  $\mathbf{H}$  is could be extremely large. In each step of the algorithm the QPFS procedure selects the parameters that should be optimized. In this example the proposed algorithm selects and updates only one parameter per iteration in the first steps. It makes the algorithm more robust.

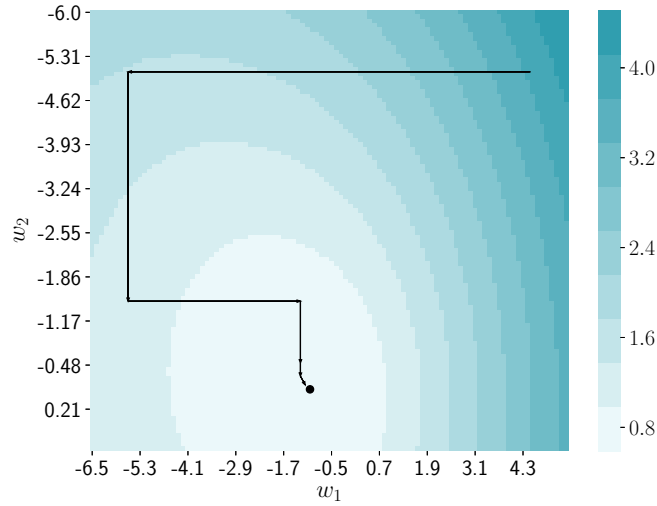


Figure 3: Optimization process for logistic regression with QPFS+Newton algorithm

Figure 4 shows the sets of active parameters over iterations for Boston housing dataset and neural network with two 2 hidden neurons. The dark cells correspond to the active parameters that we optimize.

In the considered examples the condition number  $\kappa(\mathbf{H})$  of the original Newton number in some iterations was extremely large. The selection of the active parameters



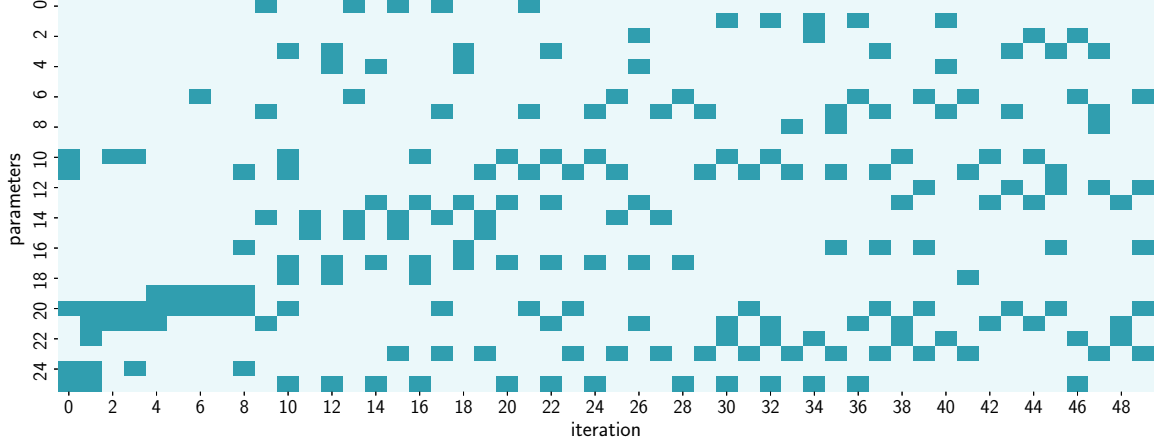


Figure 4: Active parameters sets over optimization process

allowed to reduce the condition number significantly.

We compared the proposed algorithm with the existing methods, namely Gradient Descent (GD), Nesterov Momentum, ADAM and the original Newton algorithm. Experiments were carried out for nonlinear regression problem. The datasets were chosen from the UCI achive repository [11]. The results are shown in the table 1. For each dataset there are two rows which contains mean squared error for the train (first row) and test (second row) data. We used 5-fold cross validation to find the average error and its standard deviation. The proposed algorithm shows the least error among three of four datasets. The generalization of the proposed algorithm is better than the generalization of the Newton method. It shows the difference between training and test error.

Table 1: Train and test mean squared errors for all datasets and algorithms

Datasets	#obj #feat	GD	Nesterov	ADAM	Newton	QPFS+Newton (proposed)
Boston House	506	$27.2 \pm 4.6$	$46.0 \pm 11.0$	$35.4 \pm 2.5$	$22.1 \pm 15.2$	$20.9 \pm 10.4$
Prices	13	$32.4 \pm 5.6$	$53.3 \pm 11.5$	$37.8 \pm 7.0$	$28.9 \pm 13.6$	<b><math>24.5 \pm 9.4</math></b>
Communities	1994	$48.0 \pm 6.4$	$31.4 \pm 2.8$	$23.3 \pm 3.7$	$18.3 \pm 3.4$	$26.7 \pm 3.1$
and Crime	99	$47,5 \pm 6.5$	$32.9 \pm 4.3$	$28,1 \pm 4.5$	$28.8 \pm 3.6$	<b><math>28.4 \pm 3.0</math></b>
Forest	517	$18.9 \pm 0.4$	$1.83 \pm 0.4$	$1.81 \pm 0.6$	$17.7 \pm 0.4$	$17.9 \pm 0.4$
Fires	10	<b><math>20.0 \pm 2.1</math></b>	$20.2 \pm 2.2$	<b><math>20.0 \pm 2.0</math></b>	$20.6 \pm 1.4$	$20.2 \pm 2.2$
Residential	372	$51.6 \pm 17.7$	$32.6 \pm 19.5$	$30.0 \pm 24.8$	$35.5 \pm 24.7$	$30.3 \pm 10.7$
Building	103	$53.7 \pm 13.9$	$34.1 \pm 13.6$	$34.1 \pm 19.4$	$35.0 \pm 15.6$	<b><math>30.9 \pm 5.3</math></b>

## Conclusion

The paper suggests the new approach to the second-order optimization. The selection of the set of active model parameters allows to make steps in the directions more relevant to the residuals. The experiments show the proposed algorithm improves the generalization ability and reduces the model error.

## References

- [1] Yurii Nesterov. A method of solving a convex programming problem with convergence rate  $o(1/k^2)$ . In *Soviet Mathematics Doklady*, volume 27, pages 372–376, 1983.
- [2] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for on-line learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.
- [3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [5] Mordecai Avriel. *Nonlinear programming: analysis and methods*. Courier Corporation, 2003.
- [6] Barbara Blaschke, Andreas Neubauer, and Otmar Scherzer. On convergence rates for the iteratively regularized gauss-newton method. *IMA Journal of Numerical Analysis*, 17(3):421–436, 1997.
- [7] Aleksandar Botev, Hippolyt Ritter, and David Barber. Practical gauss-newton optimisation for deep learning. *arXiv preprint arXiv:1706.03662*, 2017.
- [8] Alexandr Katrutsa and Vadim Strijov. Comprehensive study of feature selection methods to solve multicollinearity problem according to evaluation criteria. *Expert Systems with Applications*, 76:1–11, 2017.
- [9] Irene Rodriguez-Lujan, Ramon Huerta, Charles Elkan, and Carlos Santa Cruz. Quadratic programming feature selection. *Journal of Machine Learning Research*, 11(Apr):1491–1516, 2010.
- [10] Chris Ding and Hanchuan Peng. Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology*, 3(02):185–205, 2005.
- [11] Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017.