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 the proposed method TODO

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

Introduction

The modern machine learning models have extremely large number of parameters. The error function for such models have ф complex landscape with multiple local minima. The optimization procedure in this case brings to different solutions each time.

The optimization procedure is an iterative process. In each step it updates the current point parameters to get the next approximation. There have been developed a lot of first-order optimization algorithms, which use the first derivatives of the error function. The most famous algorithms are SGD, Nesterov momentum [1], Adagrad [2], Adam [3]. These algorithms are widely used for deep neural networks optimization [4]. The Newton algorithm is the second-order algorithm which use the second derivatives of the error function. It finds more reasonable updates and converges with much less number of iterations. The drawback of the second-order optimization methods huge and ill-conditioned Hessian matrix. The optimization in this case could be computationally expensive and could diverge. To overcome this problem the approximations for the Hessian matrix and regularization are applied [5, 6]. The paper [7] implements the Newton method to the deep neural networks.

This paper suggests to select the set of model parameters which we optimize in each optimization step. We consider nonlinear regression model with squared error function and logistic regression model with cross-entropy error function. For 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.

In the computational experiment we investigate TODO

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 binary classification problem and $\mathbb{Y} = \mathbb{R}$ for regression problem. There are given the design matrix $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_m]^\mathsf{T} \in \mathbb{R}^{m \times n}$ and the target vector $\mathbf{y} = [y_1, \dots, y_m]^\mathsf{T} \in \mathbb{Y}^m$. The goal is to find the optimal parameters $\mathbf{w}^*$. The parameters $\mathbf{w}$ are fitted by the minimization of the error function:

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

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

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

and cross-entropy for the binary classification problem:

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

The number of model parameters p could be extremely huge. 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

$$\mathbf{w}^k = \mathbf{w}^{k-1} + \Delta\mathbf{w}^{k-1}. \tag{4}$$

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

Newton method

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

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

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

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

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

The Newton method is unstable and computationally hard. Each iteration inverts the Hessian matrix. The measure of ill-conditioning for the Heassian matrix $\mathbf{H}$ is the condition number

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

where $\lambda_{\text{max}}(\mathbf{H}), \lambda_{\text{min}}(\mathbf{H})$ are the maximum and minimum eigenvalues of $\mathbf{H}$.

This paper suggests the robust Newton algorithm. Before the gradient step we propose to select the set of model parameters, which have the greatest impact on the error function $S(\mathbf{w})$. It reduces the size of the Hessian matrix $\mathbf{H}$ and makes the condition number $\varkappa(\mathbf{H})$ smaller. The proposed algorithm is implemented to the nonlinear regression problem and binary classification problem.

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

$$\mathbf{J} = \begin{pmatrix} \frac{\partial f(\mathbf{x}_1|\mathbf{w})}{\partial w_1} & \dots & \frac{\partial f(\mathbf{x}_1|\mathbf{w})}{\partial w_p} \\ \dots & \dots & \dots \\ \frac{\partial f(\mathbf{x}_m|\mathbf{w})}{\partial w_1} & \dots & \frac{\partial f(\mathbf{x}_m|\mathbf{w})}{\partial w_p} \end{pmatrix}. \tag{5}$$

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

$$\nabla S(\mathbf{w}) = \mathbf{J}^{\mathsf{T}}(\mathbf{y} - \mathbf{f}); \quad \mathbf{H} = \mathbf{J}^{\mathsf{T}}\mathbf{J}. \tag{6}$$

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

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

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

$$\|\mathbf{z} - \mathbf{F}\Delta\mathbf{w}\|_2^2 \to \min_{\Delta\mathbf{w}\in\mathbb{R}^p}, \tag{7}$$

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

$$f(\mathbf{x}|\mathbf{w}) = \sigma(\mathbf{x}^{\mathsf{T}}\mathbf{W}_1)\mathbf{w}_2.$$

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

Logistic Regression

For logistic regression problem the model $f(\mathbf{x}|\mathbf{w}) = \sigma(\mathbf{x}^{\mathsf{T}}\mathbf{w})$, where σ is a sigmoid function. The gradient and the Hessian of the error function (3) equal

$$\nabla S(\mathbf{w}) = \mathbf{X}^{\mathsf{T}}(\mathbf{f} - \mathbf{y}); \quad \mathbf{H} = \mathbf{X}^{\mathsf{T}}\mathbf{R}\mathbf{X}, \tag{8}$$

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

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

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

$$\|\mathbf{z} - \mathbf{F}\Delta\mathbf{w}\|_2^2 \to \min_{\Delta\mathbf{w} \in \mathbb{R}^p}, \tag{9}$$

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

Quadratic programming feature selection

Consider linear regression problem

$$\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 \to \min_{\mathbf{w} \in \mathbb{R}^n}. \tag{10}$$

If there are multicollinearity between columns of the design matrix $\mathbf{X}$ the solution of (10) is unstable. The feature selection methods finds the set $\mathcal{A} \in \{1, \dots, n\}$ of representative $\mathbf{X}$ columns.

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: Sim and Rel. The former measures the redundancy between features, the latter contains relevances between each feature and target vector. We want to minimize the Sim function and maximize the Rel simultaneously.

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

$$\underbrace{\mathbf{a}^{\mathsf{T}}\mathbf{Q}\mathbf{a}}_{\text{Sim}} - \alpha \cdot \underbrace{\mathbf{b}^{\mathsf{T}}\mathbf{a}}_{\text{Rel}} \to \min_{\substack{\mathbf{a} \geqslant 0 \\ \|\mathbf{a}\|_1 = 1}}. \tag{11}$$

The first term is associated with the Sim function and the second with the Rel. 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 (11) penalizes the dependent features by the function Sim and encourages features relevant to the target by the function Rel. The parameter α 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 α and make Sim and Rel terms impact are equal

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

where $\overline{\mathbf{Q}}$, $\overline{\mathbf{b}}$ 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 mutual information coefficient between pairs of features for the Sim function and between features and target vector for the Rel function. The problem (11) is convex if the matrix $\mathbf{Q}$ is positive semidefinite. In general it is not always true. To satisfy this condition we shift the matrix $\mathbf{Q}$ spectrum and replace the matrix $\mathbf{Q}$ by $\mathbf{Q} - \lambda_{\min}\mathbf{I}$, where $\lambda_{\min}$ is a $\mathbf{Q}$ minimal eigenvalue.

Proposal

To make the optimization process of the Newton algorithm is robust and stable we propose to implement the QPFS algorithm to the problems (7) and (9). The QPFS selects the set $\mathcal{A}$ of weight updates $\Delta\mathbf{w}$, which have the greatest impact to the residuals and pairwise independent and which are needed to optimize in the current optimization step. We update only the weights with indices from the set $\mathcal{A}$

$$\mathbf{w}^k_{\mathcal{A}} = \mathbf{w}^{k-1}_{\mathcal{A}} + \Delta\mathbf{w}^{k-1}_{\mathcal{A}}, \quad \mathbf{w}_{\mathcal{A}} = \{w_j\}_{j \in \mathcal{A}}.$$

The sample correlation coefficient and mutual information coefficient are 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

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

and the logistic regression problem

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

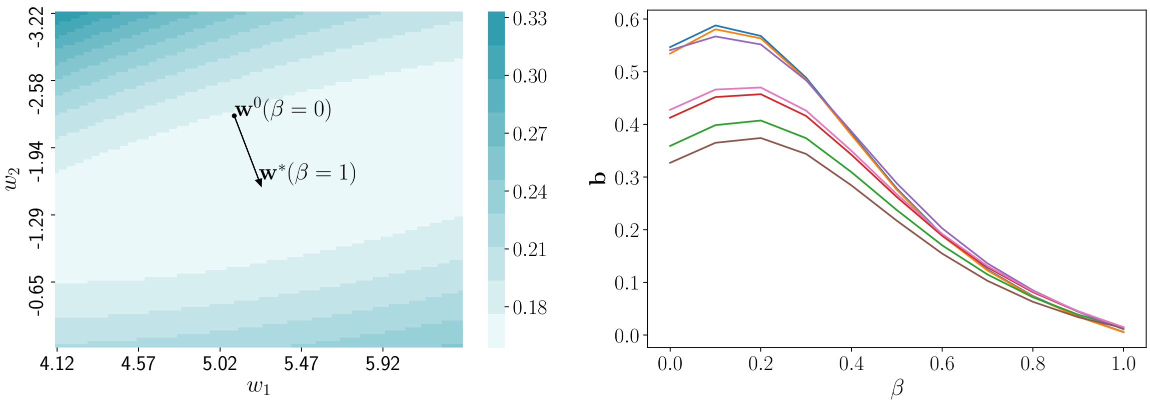
Step size

The step size of the Newton method could be excessively large. To control the step size of the weight updates we add the parameter η 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 η the Armijo rule is used. We choose η as large as possible to satisfy the following condition

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



(a) Error landscape

(b) Feature relevances

Figure 1: Logistic regression

Experiment

The goal of the computational experiment is TODO

We investigate the dependence of the QPFS parameters for the problems (7), (9). Assume that weight vector $\mathbf{w}^0$ lies near the optimal weight vector $\mathbf{w}^*$. We consider the line segment

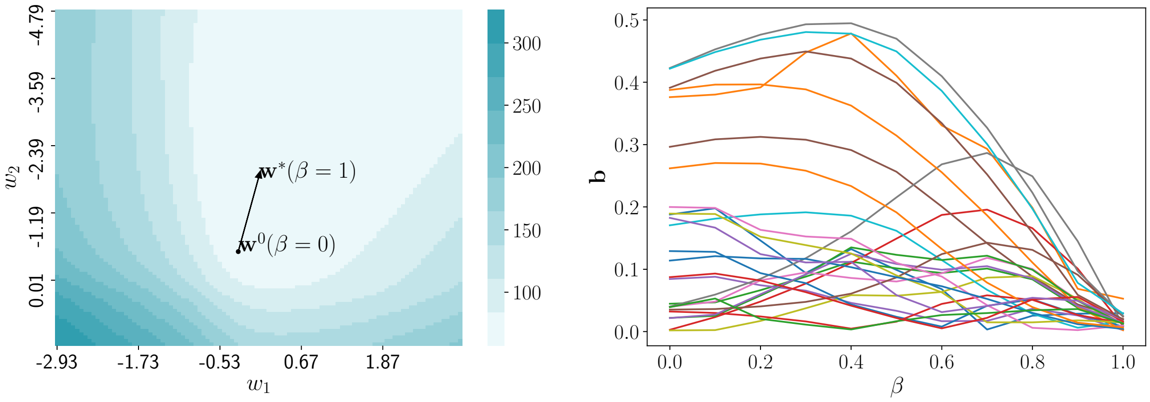
$$\mathbf{w}_\beta = \beta\mathbf{w}^* + (1-\beta)\mathbf{w}^0;\ \beta \in [0, 1].$$

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 weights. We add the random noise to the optimum weights $\mathbf{w}^*$ to get the point $\mathbf{w}^0$. The behaviour of the Rel term vector $\mathbf{b}$ on the line segment between $\mathbf{w}^0$ 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 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 weigths that should be optimized. In this example the proposed algorithm selects and updates only one weight per iteration in the first steps. It makes the algorithm more robust.

Figure 4 shows the sets of active parameters over iterations for Boston housing



(a) Error landscape

(b) Feature relevances

Figure 2: Neural network, first layer

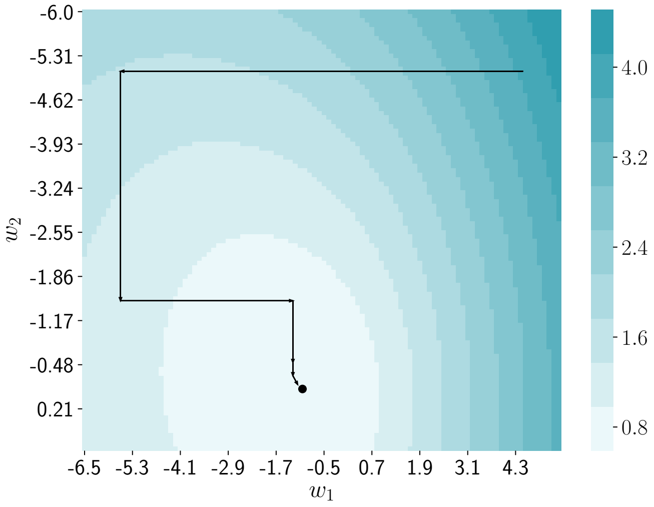


Figure 3: Optimization process for logistic regression with QPFS+Newton algorithm

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 (7) or (9) ;
    $\mathbf{Q} := \text{Sim}(\mathbf{F}),\ \mathbf{b} := \text{Rel}(\mathbf{F}, \mathbf{z}),\ \alpha = \frac{\overline{\mathbf{Q}}}{\overline{\mathbf{Q}} + \overline{\mathbf{b}}}$;
    $\mathbf{a} := \underset{\mathbf{a} \geqslant 0,\, \|\mathbf{a}\|_1 = 1}{\arg\min}\ \mathbf{a}^{\mathsf{T}}\mathbf{Q}\mathbf{a} - \alpha \cdot \mathbf{b}^{\mathsf{T}}\mathbf{a}$;
    $\mathcal{A} = \{j \mid a_j > \tau\}$;
    compute $\nabla S(\mathbf{w}_{k-1})$, $\mathbf{H}$ from (6) or (8);
    $\Delta\mathbf{w}^{k-1} = -\mathbf{H}^{-1}\nabla S(\mathbf{w}^{k-1})$;
    $\eta := \text{ArmijoRule}(\mathbf{w}_{k-1}, \gamma)$;
    $\mathbf{w}^k_{\mathcal{A}} = \mathbf{w}^{k-1}_{\mathcal{A}} + \eta\Delta\mathbf{w}^{k-1}_{\mathcal{A}}$;
    $k := k + 1$;
  until $\frac{\|\mathbf{w}^k - \mathbf{w}^{k-1}\|}{\|\mathbf{w}^k\|} < \varepsilon$
```

dataset and neural network with two 2 hidden neurons. The dark cells correspond to the active parameters that we optimize.

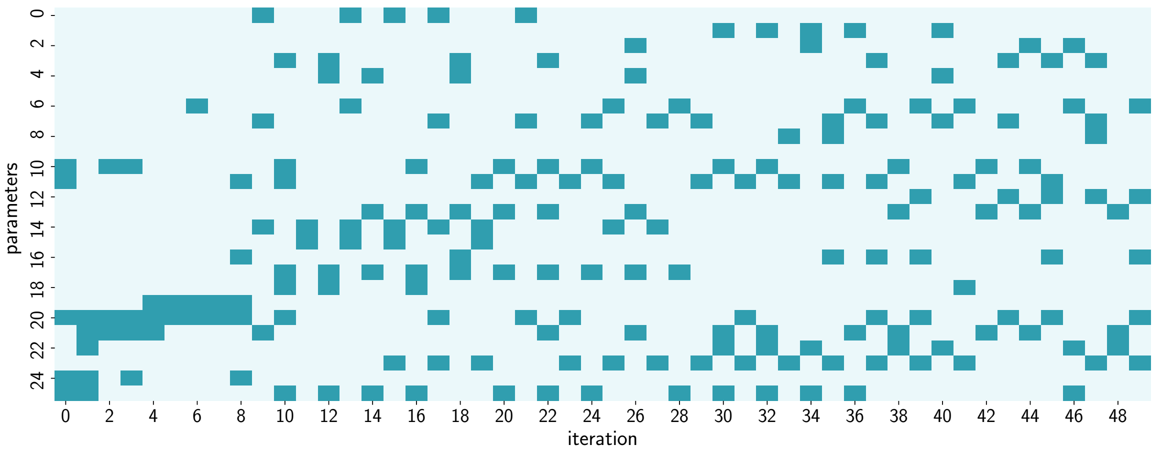


Figure 4: Active parameters sets over optimization process

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

Conclusion

References

[1] Yurii Nesterov. A method of solving a convex programming problem with convergence rate o (1/k2). In *Soviet Mathematics Doklady*, volume 27, pages 372–376, 1983.

[2] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online 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.