Gradient methods for parameter optimization

Exercise T4.1: Multilayer perceptron (tutorial)

(a) Recap the optimization of the MLP parameters (via the backpropagation algorithm).

(b) Outline the weight space symmetries giving rise to $\Pi_{v=1}^{L} N_v! \cdot 2^{N_v}$ equivalent solutions where L is the number of hidden layers and N_v the respective number of neurons in layer $v \implies$ no unique global minimum but a large equivalence class of (best) solutions.

Exercise T4.2: Linear neuron for regression (tutorial)

To prepare for the homework, we discuss a simple connectionist neuron with linear output function for a real one-dimensional input $x \in \mathbb{R}$ and output $y \in \mathbb{R}$.

(a) Describe the output function $y(x; \underline{\mathbf{w}})$ of the neuron in vector notation.

(b) Derive the gradient and Hessian matrix of the quadratic error function.

(c) Solve the optimization of the quadratic error function for a data set $\{(x^{(\alpha)}, y_T^{(\alpha)})\}_{\alpha=1,\ldots,p}$ analytically in matrix form.

(d) Calculate the solution when the objective includes the quadratic training cost E^T plus a "weight decay" regularization term as used in *ridge regression*, i.e.

$$R_{[\underline{\mathbf{w}}]} = E^T_{[\underline{\mathbf{w}}]} + \lambda \, ||\underline{\mathbf{w}}||^2$$

Exercise T4.3: Conjugate gradient (tutorial)

(a) How does the convergence speed of *gradient descent* depend on the learning rate η?

(b) Describe how *line search* speeds up convergence.

(c) What is a *conjugate direction* and how can it improve convergence speed?

Exercise H4.1: Line search (homework, 4 points)

In this exercise you will analyze line search based on the simple example of a linear neuron with quadratic cost function $E^T_{[\underline{\mathbf{w}}]}$. Here we optimize the cost function along a given direction $\underline{\mathbf{d}}_t$ (that can be but is not necessarily identical to the gradient):

$$\underline{\mathbf{w}}_{t+1} = \underline{\mathbf{w}}_t - \eta_t \, \underline{\mathbf{d}}_t \, .$$

(a) (1 point) Derive the 2nd order Taylor approximation of an *arbitrary* $E^T_{[\underline{\mathbf{w}}_{t+1}]}$ around $\underline{\mathbf{w}}_t$.

(b) (1 point) Derive a bound on the step size η_t using the above approximation in
$E^T_{[\underline{\mathbf{w}}_{t+1}]} \overset{!}{\leq} E^T_{[\underline{\mathbf{w}}_t]}$.

(c) (1 point) Derive the optimal step size η_t^* for the quadratic cost function

$$E^T_{[\underline{\mathbf{w}}]} = \frac{1}{2}(\underline{\mathbf{w}} - \underline{\mathbf{w}}^*)^\top \underline{\mathbf{H}}(\underline{\mathbf{w}} - \underline{\mathbf{w}}^*)$$

by minimizing the cost function w.r.t. η. Make sure your solution depends only on known quantities like the weight vector $\underline{\mathbf{w}}_t$, the gradient $\underline{\nabla} E^T_{[\underline{\mathbf{w}}_t]}$ and/or the Hessian $\underline{\mathbf{H}}$ of $E^T_{[\underline{\mathbf{w}}_t]}$.

(d) (1 point) Prove that the gradient $\underline{\nabla} E^T_{[\underline{\mathbf{w}}_{t+1}]}$ after one update step with *line search* is orthogonal to the optimized direction $\underline{\mathbf{d}}_t$.

Exercise H4.2: Comparison of gradient descent methods (homework, 6 points)

In this exercise we compare the performance of three learning procedures applied to a simple connectionist neuron with a linear output function. All procedures will compute the gradient using the entire training set (batch gradient descent). The procedures are: (i) Gradient (or steepest) descent with constant learning rate, (ii) steepest descent combined with a line search method to determine the learning rate, and (iii) the conjugate gradient method.

Training Data: The training data set consists of three points ($p = 3$):

$$\{(x^{(\alpha)}, y_T^{(\alpha)})\} = \{(-1, -0.1), (0.3, 0.5), (2, 0.5)\},$$

i.e. for a given data point, both input and output are scalar values.

Cost function: The gradient for the *quadratic error* function is given by

$$\underline{\mathbf{g}} = \frac{\partial E^T}{\partial \underline{\mathbf{w}}} = \underline{\mathbf{H}}\,\underline{\mathbf{w}} + \underline{\mathbf{b}}, \qquad \text{with} \quad \underline{\mathbf{H}} = \underline{\mathbf{X}}\,\underline{\mathbf{X}}^\top \quad \text{and} \quad \underline{\mathbf{b}} = -\underline{\mathbf{X}}\,\underline{\mathbf{y}}^\top,$$

where $\underline{\mathbf{X}} = \begin{pmatrix} 1 & 1 & \dots & 1 \\ x^{(1)} & x^{(2)} & \dots & x^{(p)} \end{pmatrix} \in \mathbb{R}^{2,p}$ and $\underline{\mathbf{y}} = \left(y_T^{(1)}, y_T^{(2)}, \dots, y_T^{(p)}\right) \in \mathbb{R}^{1,p}$.

Initialization: Use the following initialization for all three (batch) gradient methods:

$$\underline{\mathbf{w}}_1 = (w_0, w_1)_1^\top = (-0.45, 0.2)^\top$$

(a) (2 points) *Gradient Descent:* Implement a steepest descent procedure where the weights at iteration $t + 1$ are calculated using the weights and the gradient at iteration t

$$\underline{\mathbf{w}}_{t+1} = \underline{\mathbf{w}}_t - \eta\, \underline{\mathbf{g}}_t,$$

with an adequate learning rate η and where $\underline{\mathbf{g}}_t = \underline{\mathbf{g}}(\underline{\mathbf{w}}_t)$. Plot the resulting weight vectors from all iterations as a scatter plot (w_0 vs. w_1), and in an additional plot (w_i vs. iterations t), to show the development of each parameter during gradient descent.

(b) (2 points) *Line Search:* Implement a line search procedure

$$\underline{\mathbf{w}}_{t+1} = \underline{\mathbf{w}}_t - \eta\, \underline{\mathbf{g}}_t, \qquad \text{with optimal step size} \qquad \eta = \frac{\underline{\mathbf{g}}_t^\top \underline{\mathbf{g}}_t}{\underline{\mathbf{g}}_t^\top \underline{\mathbf{H}} \underline{\mathbf{g}}_t}.$$

Plot the resulting weight vectors from all iterations as a scatter plot (w_0 vs. w_1), and in an additional plot (w_i vs. iterations t), to show the development of the parameters during line search.

(c) (2 points) *Conjugate Gradient:* Implement a conjugate gradient procedure:

Initialize: $\underline{\mathbf{w}}_1, \underline{\mathbf{d}}_1 = -\underline{\mathbf{g}}_1$

while *stopping criterion not satisfied* **do**

minimize E along $\underline{\mathbf{d}}_t$: $\underline{\mathbf{w}}_{t+1} = \underline{\mathbf{w}}_t + \eta_t \underline{\mathbf{d}}_t$ with step size $\eta_t = -\frac{\underline{\mathbf{d}}_t^\top \underline{\mathbf{g}}_t}{\underline{\mathbf{d}}_t^\top \underline{\mathbf{H}}\, \underline{\mathbf{d}}_t}$

calculate new gradient $\underline{\mathbf{g}}_{t+1} = \underline{\mathbf{H}}\, \underline{\mathbf{w}}_{t+1} + \underline{\mathbf{b}}$

calculate new conjugate direction $\underline{\mathbf{d}}_{t+1} = \underline{\mathbf{g}}_{t+1} + \beta_t \underline{\mathbf{d}}_t$ with "momentum"

$$\beta_t = -\frac{\underline{\mathbf{g}}_{t+1}^\top \underline{\mathbf{g}}_{t+1}}{\underline{\mathbf{g}}_t^\top \underline{\mathbf{g}}_t}. \qquad \text{(Fletcher-Reeves form)}$$

increase $t \leftarrow t + 1$

end

Plot the resulting weight vectors from all iterations as a scatter plot (w_0 vs. w_1), and in an additional plot (w_i vs. iterations t), to show the development of the parameters during conjugate gradient descent.

Compare the different methods in terms of convergence behaviour.

Total 10 points.