OPTIMIZATION. HOMEWORK 5

OSCAR DALMAU

(1) Implement the line search steepest descent algorithm using the following methods for computing the step size: backtracking, quadratic interpolation and cubic interpolation. Apply the previous implementations to the following functions and compare the results with respect to: number of iterations, norm of the gradient $\|\nabla f(\boldsymbol{x}_k)\|$ and the error $|f(\boldsymbol{x}_k) - f(\boldsymbol{x}^*)|$.

- Rosembrock function, for $n = 2$ and $n = 100$

$$
\begin{aligned}
f(\boldsymbol{x}) &= \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right] \\
\boldsymbol{x}^0 &= [-1.2, 1, 1, \cdots, 1, -1.2, 1]^T \\
\boldsymbol{x}^* &= [1, 1, \cdots, 1, 1]^T \\
f(\boldsymbol{x}^*) &= 0
\end{aligned}
$$

- Wood function

$$
\begin{aligned}
f(\boldsymbol{x}) &= 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 \\
&\quad 10.1[(x_2 - 1)^2 + (x_4 - 1)^2] + 19.8(x_2 - 1)(x_4 - 1) \\
\boldsymbol{x}^0 &= [-3, -1, -3, -1]^T \\
\boldsymbol{x}^* &= [1, 1, 1, 1]^T \\
f(\boldsymbol{x}^*) &= 0
\end{aligned}
$$

(2) The dataset mnist.pkl.gz can be read using

```
import cPickle, gzip, numpy
# Load the dataset
f = gzip.open('mnist.pkl.gz', 'rb')
train_set, valid_set, test_set = cPickle.load(f)
f.close()
```

- train_set[0] is a matrix of size (50000, 784) where $n = 50000$ is the number of observations and each row represents an observation $\boldsymbol{x}_i \in \mathbb{R}^{784}$
- train_set[1] is a vector of size (50000) where each entry $y_i \in \{0, 1, \cdots, 9\}$

- test_set[0] is a matrix of size (10000, 784) where $n = 10000$ is the number of observations and each row represents an observation $\boldsymbol{x}_i \in \mathbb{R}^{784}$
- test_set[1] is a vector of size (10000) where each entry $y_i \in \{0, 1, \cdots, 9\}$
- Additionally, we include the previous information in *.csv files, that you can download from the website.

Select from train_set[0] and train_set[1] the set of observations $\mathcal{S} = \{(\boldsymbol{x}_i, y_i)\}$ with $\boldsymbol{x}_i \in \mathbb{R}^{784}$ and $y_i \in \{0, 1\}$ and estimate the parameters $\hat{\boldsymbol{\beta}}, \hat{\beta}_0$ that maximizes the function

$$h(\boldsymbol{\beta}, \beta_0) = \sum_{i=1}^{n} y_i \log \pi_i + (1 - y_i) \log(1 - \pi_i) \tag{1}$$

$$\pi_i := \pi_i(\boldsymbol{\beta}, \beta_0) = \frac{1}{1 + \exp\left(-\boldsymbol{x}_i^T \boldsymbol{\beta} - \beta_0\right)} \tag{2}$$

using the set $\mathcal{S}$. Select one optimization method implemented in the homework for computing $\hat{\boldsymbol{\beta}}, \hat{\beta}_0$.

Select from test_set[0] and test_set[1] the set $\mathcal{T} = \{(\boldsymbol{x}_i, y_i)\}$ such that and $y_i \in \{0, 1\}$ and compute the error

$$error = \frac{1}{|\mathcal{T}|} \sum_{(\boldsymbol{x}_i, y_i) \in \mathcal{T}} |\mathbf{1}_{\pi_i(\hat{\boldsymbol{\beta}}, \hat{\beta}_0) > 0.5}(\boldsymbol{x}_i) - y_i|$$

where $|\mathcal{T}|$ represents the number of elements of the set $\mathcal{T}$.

Note:

- Each row $\boldsymbol{x}_i$ of train_set[0] and/or test_set[0] can be shown as an image as follows:

```
import matplotlib.pyplot as plt
idx = 1 # index of the image
im = train_set[0][idx].reshape(28, -1)
plt.imshow(im, cmap=plt.cm.gray)
print('Label: ', train_set[1][idx])
```

- The equations (1)-(2) correspond to the log-likelihood of the logistic regression model for two-class classification.
- You can find more information available on line at: http://yann.lecun.com/exdb/mnist/