Modern Optimization Techniques

Lucas Rego Drumond

Information Systems and Machine Learning Lab (ISMLL)
Institute of Computer Science
University of Hildesheim, Germany

Stochastic Gradient Descent

Outline

Outline

1. Unconstrained Optimization

2. Stochastic Gradient Descent

3. Choosing the right step size

4. Stochastic Gradient Descent on Practice

Gradient Descent

procedure GradientDescent
input: f_0
  Get initial point $\mathbf{x}$
  repeat
    Get Step Size μ
    $\mathbf{x} := \mathbf{x} - \mu \nabla f_0(\mathbf{x})$
  until convergence
  return $\mathbf{x}$, $f_0(\mathbf{x})$
end procedure

Outline

Practical Example: Household Spending

Suppose we have the following data about different households:

- Number of workers in the household (a_1)
- Household composition (a_2)
- Region (a_3)
- Gross normal weekly household income (a_4)
- **Weekly household spending** (y)

We want to creat a model of the weekly household spending

Practical Example: Household Spending

If we have data about m households, we can represent it as:

$$A_{m,n} = \begin{pmatrix} 1 & a_{1,2} & \dots & a_{1,n} \\ 1 & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & a_{m,2} & \dots & a_{m,n} \end{pmatrix} \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$$

We can model the household consumption is a linear combination of the household features with parameters $\mathbf{x}$:

$$\hat{y}_i = \mathbf{x}^T \mathbf{a_i} = x_0 1 + x_1 a_{i,1} + x_2 a_{i,2} + x_3 a_{i,3} + x_4 a_{i,4}$$

Least Square Problem Revisited

The following least square problem

$$\text{minimize} \quad ||A\mathbf{x} - \mathbf{y}||_2^2$$

Can be rewritten as

$$\text{minimize} \quad \sum_{i=1}^{m} (\mathbf{x}^T \mathbf{a_i} - y_i)^2$$

$$A_{m,n} = \begin{pmatrix} 1 & a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ 1 & a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & a_{m,1} & a_{m,2} & a_{m,3} & a_{m,4} \end{pmatrix} \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$$

The Gradient Descent update rule

For the problem

$$\text{minimize} \quad \sum_{i=1}^{m} (\mathbf{x}^T \mathbf{a_i} - y_i)^2$$

The the gradient $\nabla f_0(\mathbf{x})$ of the objective function is:

$$\nabla_{\mathbf{x}} f_0(\mathbf{x}) = 2 \sum_{i=1}^{m} (\mathbf{x}^T \mathbf{a_i} - y_i)\mathbf{a_i}$$

The Gradient Descent update rule is then:

$$\mathbf{x} \rightarrow \mathbf{x} - \mu \left(2 \sum_{i=1}^{m} (\mathbf{x}^T \mathbf{a_i} - y_i)\mathbf{a_i} \right)$$

The Gradient Descent update rule

We need to “see“ all the data before updating $\mathbf{x}$

$$\mathbf{x} \rightarrow \mathbf{x} - \mu \left(2 \sum_{i=1}^{m} (\mathbf{x}^T \mathbf{a_i} - y_i)\mathbf{a_i} \right)$$

Can we make any progress before iterating over all the data?

Decomposing the objective function

The objective function

$$f_0(\mathbf{x}) = \sum_{i=1}^{m} (\mathbf{x}^T \mathbf{a_i} - y_i)^2$$

Can be expressed as a function of the objective on each data point $(\mathbf{a}, y)$:

$$g(\mathbf{x}, i) = (\mathbf{x}^T \mathbf{a_i} - y_i)^2$$

So that

$$f_0(\mathbf{x}) = \sum_{i=1}^{m} g(\mathbf{x}, i)$$

A simpler update rule

Now that we have

$$f_0(\mathbf{x}) = \sum_{i=1}^{m} g(\mathbf{x}, i)$$

We can define the following update rule

- Pick a random instance $i \sim \text{Uniform}(1, m)$
- Update $\mathbf{x}$

$$\mathbf{x} \rightarrow \mathbf{x} + \mu \left(-\nabla_{\mathbf{x}} g(\mathbf{x}, i)\right)$$

Stochastic Gradient Descent (SGD)

procedure StochasticGradiendDescent
input: f_0, μ
    Get initial point $\mathbf{x}$
    repeat
        for $i \in 1, \ldots, m$ **do**
            $\mathbf{x} \rightarrow \mathbf{x} - \mu \nabla g(\mathbf{x}, i)$
        end for
    until convergence
    return $\mathbf{x}$, $f_0(\mathbf{x})$
end procedure

SGD and the least squares

We have

$$f_0(\mathbf{x}) = \sum_{i=1}^{m} g(\mathbf{x}, i)$$

with

$$g(\mathbf{x}, i) = (\mathbf{x}^T \mathbf{a_i} - y_i)^2$$

The update rule is

$$\nabla_{\mathbf{x}} g(\mathbf{x}, i) = 2(\mathbf{x}^T \mathbf{a_i} - y_i)\mathbf{a_i}$$

$$\mathbf{x} \rightarrow \mathbf{x} - \mu \left(2(\mathbf{x}^T \mathbf{a_i} - y_i)\mathbf{a_i} \right)$$

SGD vs. GD

procedure SGD
input: f_0, μ
    Get initial point $\mathbf{x}$
    repeat
        for $i \in 1, \ldots, m$ **do**
            $\mathbf{x} \rightarrow \mathbf{x} - \mu \nabla g(\mathbf{x}, i)$
        end for
    until convergence
    return $\mathbf{x}$, $f_0(\mathbf{x})$
end procedure

procedure GradientDescent
input: f_0
    Get initial point $\mathbf{x}$
    repeat
        Get Step Size μ
        $\mathbf{x} := \mathbf{x} - \mu \nabla f_0(\mathbf{x})$
    until convergence
    return $\mathbf{x}$, $f_0(\mathbf{x})$
end procedure

SGD vs. GD - Least Squares

procedure SGD
input: f_0, μ
Get initial point $\mathbf{x}$
repeat
for $i \in 1, \ldots, m$ **do**
$$\mathbf{x} \rightarrow \mathbf{x} - \mu \left(2(\mathbf{x}^T \mathbf{a_i} - y_i)\mathbf{a_i}\right)$$
end for
until convergence
return $\mathbf{x}$, $f_0(\mathbf{x})$
end procedure

procedure GD
input: f_0
Get initial point $\mathbf{x}$
repeat
Get Step Size μ
$$\mathbf{x} \rightarrow \mathbf{x} - \mu \left(2 \sum_{i=1}^{m} (\mathbf{x}^T \mathbf{a_i} - y_i)\mathbf{a_i}\right)$$
until convergence
return $\mathbf{x}$, $f_0(\mathbf{x})$
end procedure

Outline

Choosing the step size for SGD

- The step size μ is a crucial parameter to be tuned
- Given the low cost of the SGD update, using line search for the step size is a bad choice
- Possible alternatives:
  - Fixed step size
  - Armijo principle
  - Bold-Driver
  - Adagrad

Real World Dataset: Body Fat prediction

We want to estimate the percentage of body fat based on various attributes:

- Age (years)
- Weight (lbs)
- Height (inches)
- Neck circumference (cm)
- Chest circumference (cm)
- Abdomen 2 circumference (cm)
- Hip circumference (cm)
- Thigh circumference (cm)
- Knee circumference (cm)
- ...

http://lib.stat.cmu.edu/datasets/bodyfat

Real World Dataset: Body Fat prediction

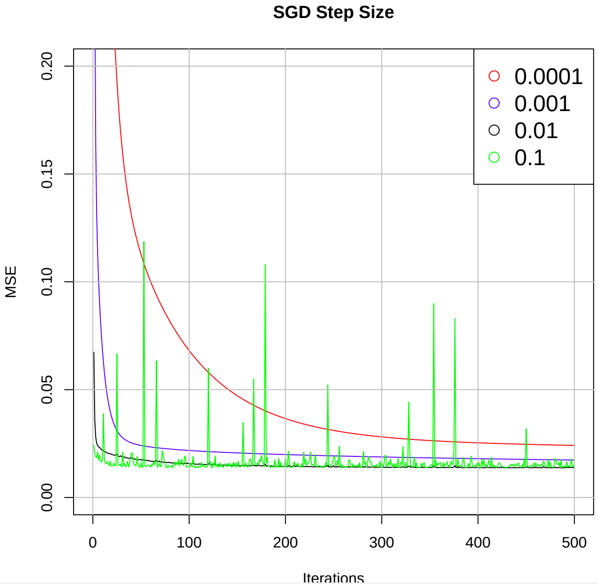
The data is represented it as:

$$A_{m,n} = \begin{pmatrix} 1 & a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ 1 & a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & a_{m,1} & a_{m,2} & \dots & a_{m,n} \end{pmatrix} \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$$

with $m = 252$, $n = 14$
We can model the percentage of body fat y is a linear combination of the body measurements with parameters $\mathbf{x}$:

$$\hat{y}_i = \mathbf{x}^T \mathbf{a_i} = x_0 1 + x_1 a_{i,1} + x_2 a_{i,2} + \ldots + x_n a_{i,n}$$

SGD - Fixed Step Size on the Body Fat dataset



Bold Driver Heuristic

- The Bold Driver Heuristic makes the assumption that smaller step sizes are needed when closer to the optimum
- It adjusts the step size based on the value of $f_0(\mathbf{x}^t) - f_0(\mathbf{x}^{t-1})$
- If the value of $f_0(\mathbf{x})$ grows, the step size must decrease
- If the value of $f_0(\mathbf{x})$ decreases, the step size can be larger for faster convergence

Bold Driver Heuristic - Update Rule

We have

$$f_0(\mathbf{x}) = \sum_{i=1}^{m} g(\mathbf{x}, i)$$

We need to define an increase factor γ and a decay factor ν

- For each epoch
- Evaluate the objective function $f_0(\mathbf{x}^{t-1})$
- Cycle through the whole data and update the parameters
- Evaluate the objective function $f_0(\mathbf{x}^{t})$
- if $f_0(\mathbf{x}^{t}) < f_0(\mathbf{x}^{t-1})$ then $\mu \rightarrow \gamma\mu$
- else $f_0(\mathbf{x}^{t}) > f_0(\mathbf{x}^{t-1})$ then $\mu \rightarrow \nu\mu$

Widely used values: $\gamma = 1.05$ and $\nu = 0.5$

SGD with Bold Driver

procedure BoldDriverSGD
input: f_0, μ, γ and ν
  Get initial point $\mathbf{x}$
  repeat
    $\epsilon^{t-1} \rightarrow f_0(\mathbf{x})$
    for $i \in 1, \ldots, m$ **do**
      $\mathbf{x} \rightarrow \mathbf{x} - \mu \nabla g(\mathbf{x}, i)$
    end for
    $\epsilon^{t} \rightarrow f_0(\mathbf{x})$
    if $\epsilon^{t} < \epsilon^{t-1}$ **then**
      $\mu \rightarrow \nu\mu$
    else
      $\mu \rightarrow \gamma\mu$
    end if
  until convergence
  return $\mathbf{x}$, $f_0(\mathbf{x})$

Considerations

- Works well for a range of problems
- The initial μ just need to be large enough
- γ and ν needs to be adusted
- May lead to faster convergence rates

AdaGrad

- Adagrad adjusts the step size for each parameter to be optimized
- It uses information about the past gradients
- Leads to faster convergence
- Less sensitive to the choice of the step size

AdaGrad - Update Rule

We have

$$f_0(\mathbf{x}) = \sum_{i=1}^{m} g(\mathbf{x}, i)$$

Update rule:

- Pick a random instance $i \sim \text{Uniform}(1, m)$
- Compute the gradient $\nabla_{\mathbf{x}} g(\mathbf{x}, i)$
- Update the gradient history $\mathbf{h} \rightarrow \mathbf{h} + \nabla_{\mathbf{x}} g(\mathbf{x}, i) \circ \nabla_{\mathbf{x}} g(\mathbf{x}, i)$
- The step size for parameter $\mathbf{x}_i$ is $\frac{\mu}{\sqrt{h_i}}$
- Update

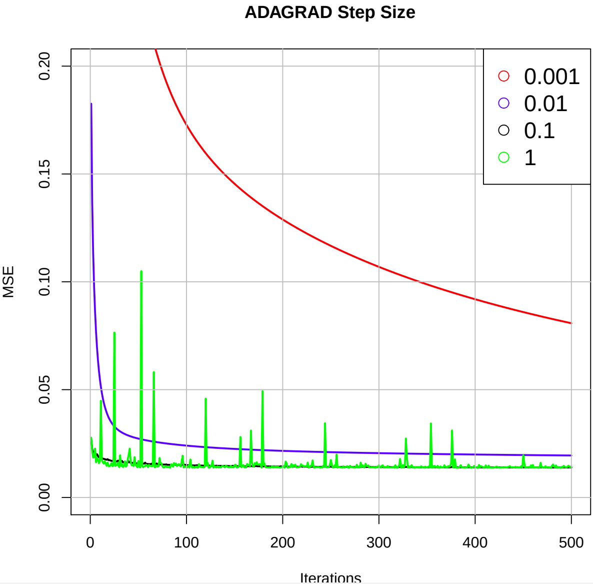
$$\mathbf{x} \rightarrow \mathbf{x} - \frac{\mu}{\sqrt{\mathbf{h}}} \circ (\nabla_{\mathbf{x}} g(\mathbf{x}, i))$$

$\circ$ denotes the elementwise product

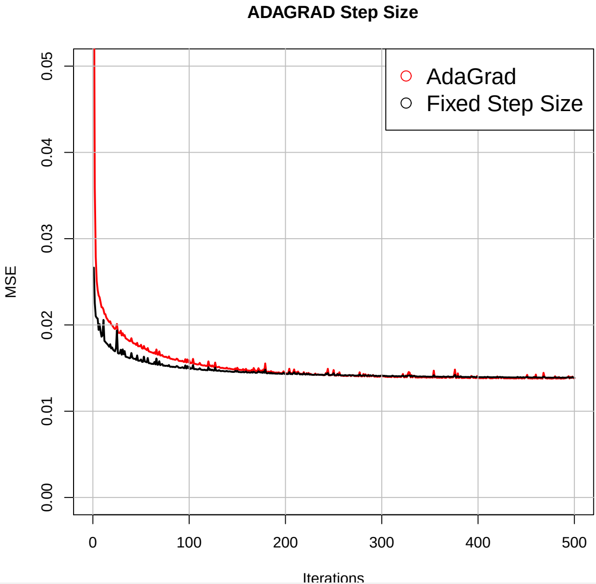
SGD with Adagrad

procedure ADAGRADSGD
input: f_0, μ
  Get initial point $\mathbf{x}$
  $\mathbf{h} \rightarrow \mathbf{0}$
  repeat
    for $i \in 1, \ldots, m$ **do**
      $\mathbf{h} \rightarrow \mathbf{h} + \nabla_{\mathbf{x}} g(\mathbf{x}, i) \circ \nabla_{\mathbf{x}} g(\mathbf{x}, i)$
      $\mathbf{x} \rightarrow \mathbf{x} - \frac{\mu}{\sqrt{\mathbf{h}}} \circ \nabla g(\mathbf{x}, i)$
    end for
  until convergence
  return $\mathbf{x}$, $f_0(\mathbf{x})$
end procedure

AdaGrad Step Size

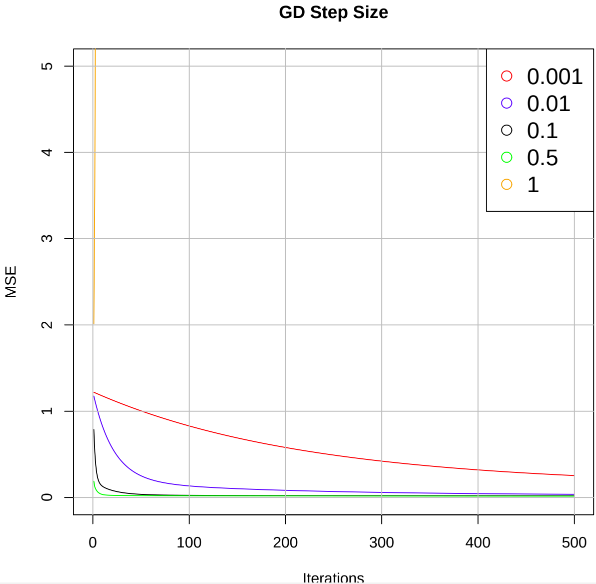


AdaGrad vs Fixed Step Size

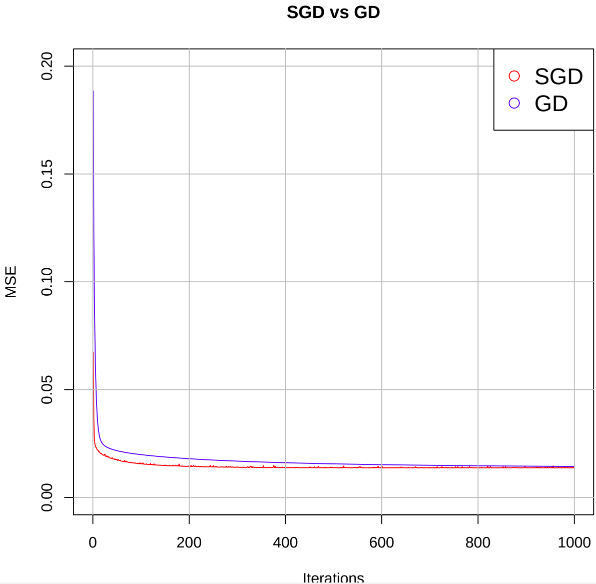


Outline

GD Step Size


GD Step Size
MSE
0
1
2
3
4
5
Iterations
0
100
200
300
400
500
0.001
0.01
0.1
0.5
1

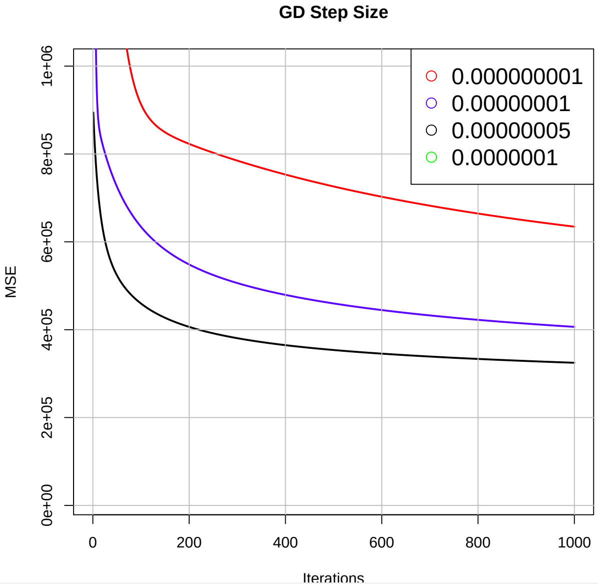
SGD vs GD - Body Fat Dataset



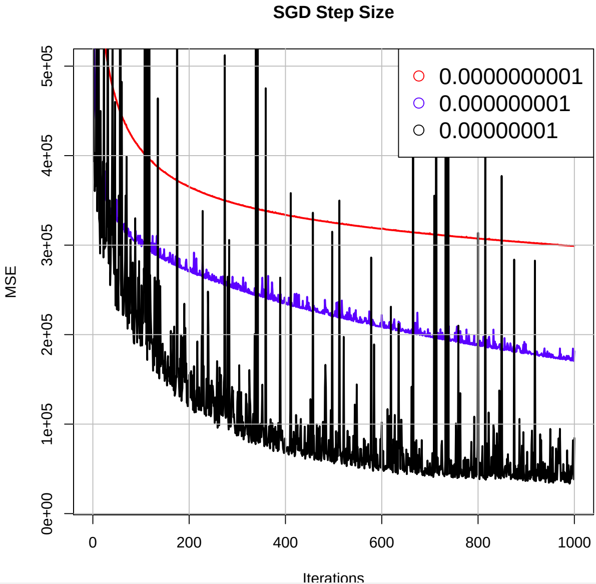
Year Prediction Data Set

- Least Squares Problem
- Prediction of the release year of a song from audio features
- 90 features
- Experiments done on a subset of 1000 instances of the data

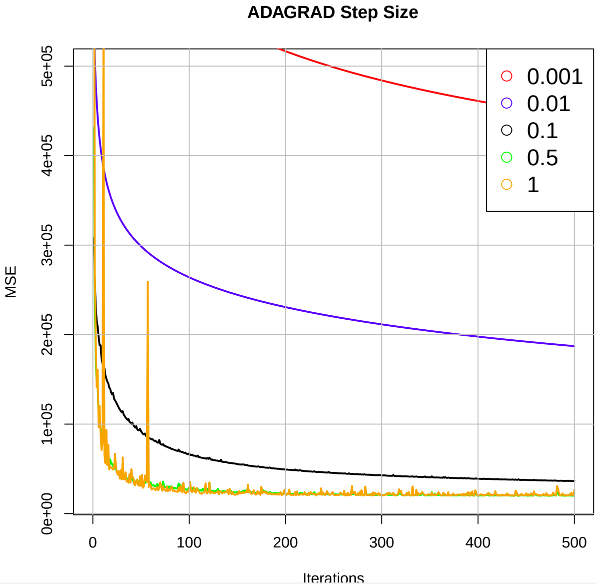
GD Step Size - Year Prediction



SGD Step Size - Year Prediction



AdaGrad Step Size - Year Prediction



AdaGrad vs SGD vs GD - Year Prediction