

## Multivariate Optimization 3

## Exercise 1: Stochastic Gradient Descent

- (a) •  $\mathbb{E}_{\mathbf{x},y} \nabla_{\boldsymbol{\theta}} \|\boldsymbol{\theta}^\top \mathbf{x} - y\|_2^2 = \mathbb{E}_{\mathbf{x}} \mathbb{E}_{y|\mathbf{x}} \nabla_{\boldsymbol{\theta}} \|\boldsymbol{\theta}^\top \mathbf{x} - y\|_2^2 = \mathbb{E}_{\mathbf{x}} \mathbb{E}_{y|\mathbf{x}} 2\mathbf{x}\mathbf{x}^\top \boldsymbol{\theta} - 2\mathbf{x}y = \mathbb{E}_{\mathbf{x}} 2\mathbf{x}\mathbf{x}^\top \boldsymbol{\theta} - 2\mathbf{x}\mathbf{x}^\top \boldsymbol{\theta}^* = 2\boldsymbol{\Sigma}_{\mathbf{x}}(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$
- $\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x},y} \|\boldsymbol{\theta}^\top \mathbf{x} - y\|_2^2 = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x}} \mathbb{E}_{y|\mathbf{x}} \boldsymbol{\theta}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{\theta} - 2\boldsymbol{\theta}^\top \mathbf{x}y + y^2 = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x}} \boldsymbol{\theta}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{\theta} - 2\boldsymbol{\theta}^\top \mathbf{x}\mathbf{x}^\top \boldsymbol{\theta}^* + \boldsymbol{\theta}^{*\top} \mathbf{x}\mathbf{x}^\top \boldsymbol{\theta}^*$   
 $= \nabla_{\boldsymbol{\theta}} (\boldsymbol{\theta}^\top \boldsymbol{\Sigma}_{\mathbf{x}} \boldsymbol{\theta} - 2\boldsymbol{\theta}^\top \boldsymbol{\Sigma}_{\mathbf{x}} \boldsymbol{\theta}^* + \boldsymbol{\theta}^{*\top} \boldsymbol{\Sigma}_{\mathbf{x}} \boldsymbol{\theta}^*) = 2\boldsymbol{\Sigma}_{\mathbf{x}}(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$
- (b) We can estimate  $\mathbb{E}_{\mathbf{x},y} \nabla_{\boldsymbol{\theta}} \|\boldsymbol{\theta}^\top \mathbf{x} - y\|_2^2$  without bias via SGD since we have access to realizations of  $\nabla_{\boldsymbol{\theta}} \|\boldsymbol{\theta}^\top \mathbf{x} - y\|_2^2$ . From a) it follows that this estimate is also an unbiased estimate of the gradient of our objective  $\nabla_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x},y} \|\boldsymbol{\theta}^\top \mathbf{x} - y\|_2^2$ . Hence, SGD can be successfully applied in this situation.

```
(c) library(ggplot2)
library(gridExtra)

set.seed(123)

sigma_x = 0.5
sigma_y = 0.1

n = 10000
x = sort(rnorm(n, sd = sigma_x))
theta_star = 0.5
y = theta_star * x + rnorm(n, sd = sigma_y)

theta = 0.9
mean(2*(x*x*theta - y*x))

## [1] 0.2015163

compute_conf <- function(theta, n){
  x = rnorm(n, sd = sigma_x)
  y = theta_star * x + rnorm(n, sd = sigma_y)
  # mean of squared differences between the sampled gradients and
  # the gradient of the objective
  return(mean((2*(x*x*theta - y*x) - 2*sigma_x^2*(theta - theta_star))^2))
}

# compute confusions for m = 100

confs = c()
m = 100
reps = 200
thetas = seq(from=0, to=1, length.out = 21)
for(i in 1:reps){
  for(theta in thetas){
    confs = c(confs, compute_conf(theta, m))
  }
}

p_batch100 = ggplot(data.frame(thetas = rep(thetas, reps), confs = confs),
```

```

      aes(x = thetas, y = confs)) +
geom_point() + xlab(expression(theta)) + ylim(0, 0.4) + ggtitle("m = 100") +
ylab("confusion")

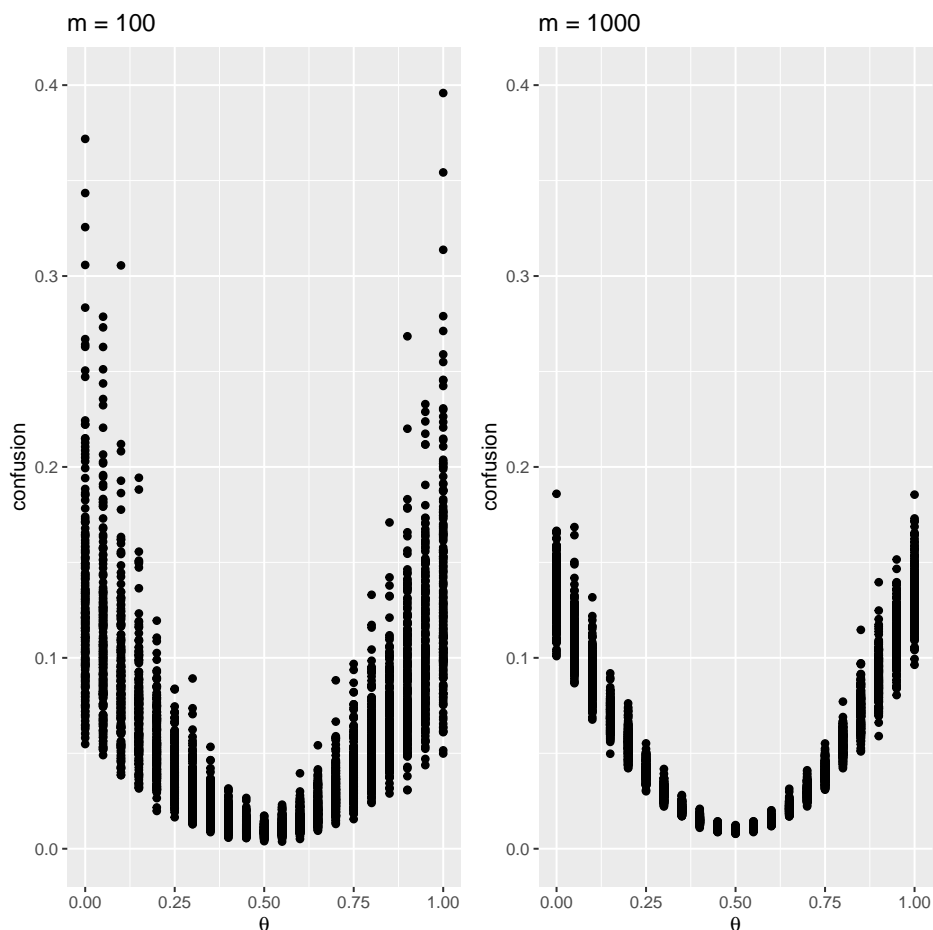
# compute confusions for m = 1000

confs = c()
m = 1000
reps = 200
thetas = seq(from=0, to=1, length.out = 21)
for(i in 1:reps){
  for(theta in thetas){
    confs = c(confs, compute_conf(theta, m))
  }
}

p_batch1000 = ggplot(data.frame(thetas = rep(thetas, reps), confs = confs),
  aes(x = thetas, y = confs)) +
  geom_point() + xlab(expression(theta)) + ylim(0, 0.4) + ggtitle("m = 1000") +
  ylab("confusion")

# plot all
grid.arrange(p_batch100, p_batch1000, ncol = 2)

```



- (d) Qualitatively, we observe for both settings that the mean and the variance of the confusion rise symmetrically around  $\theta^*$ . As expected, the mean and the variance of the confusion is smaller for the larger batch size  $m = 1000$  than for  $m = 100$ .

(e) `set.seed(123)`

```
# SGD
thetas = NULL
alpha = 0.3
m = 10
for(j in 1:200){
  theta = 0
  for(i in 1:20){
    x = rnorm(m, sd = sigma_x)
    y = theta_star * x + rnorm(n, sd = sigma_y)

    theta = theta - alpha * mean(2*(x*x*theta - y*x))
    thetas = rbind(thetas, theta)
  }
}

plot_sgd = ggplot(data.frame(thetas = thetas, it = rep(1:20, 200)),
  aes(x = it, y = thetas)) +
  geom_point() + ylab(expression(theta)) + xlab("iteration") +
  ggtitle("SGD with m=10 (200 runs)")

# GD
theta = 0
thetas = theta
alpha = 0.3

for(i in 1:20){
  theta = theta - alpha * 2*sigma_x^2*(theta - theta_star)
  thetas = rbind(thetas, theta)
}

plot_gd = ggplot(data.frame(thetas = thetas, it = 1:21),
  aes(x = it, y = thetas)) +
  geom_point() + ylab(expression(theta)) + xlab("iteration") + ggtitle("GD")

# plot all
grid.arrange(plot_sgd, plot_gd, ncol=2)
```

