CS 260B Homework 1

Hanna Co

Collaborators: Isha Gonugunta

Due: April 13, 2022

(1a)

$f(x, y) = x^2 - y^2$ and $x_0 = (0, 0)$

We can compute the gradient of f: $\nabla f(x, y) = \langle 2x, -2y \rangle$

At x_0, the gradient is zero: $\nabla f(0, 0) = \langle 0, 0 \rangle$

However, x_0 is neither a local minimum or local maximum, and we can prove this through the second partial derivative test. We compute $H = f_{xx}(x, y) * f_{yy}(x, y) - f_{xy}(x, y)^2$. If $H > 0$ then (x, y) is either a local maximum or minimum; if $H < 0$, then (x, y) is a saddle point. We compute the second order partial derivatives:

$f_{xx} = 2$

$f_{yy} = -2$

$f_{xy} = 0$

$H = (2)(-2) + 0^2 = -4$, and $-4 < 0$, so x_0 is not a local maximum or minimum. Thus, for the function $f(x, y) = x^2 - y^2$ and point $x_0 = (0, 0)$, the gradient at x_0 is 0, but x_0 is not necessarily a local max or min, therefore the condition that $\nabla f(x, y) = 0$ is necessary but not sufficient for a point to be a local max or min.

(1b)

The gradient descent algorithm gives us $w_{i+1} = w_i + t\nabla f(w_i)$

Thus, $\|w_{i+1} - w_1\|$ is equivalent to $\|w_i + t\nabla f(w_i) - w_i\|$

$\|w_{i+1} - w_1\| = t\|\nabla f(w_i)\|$

This means that, each iteration, the decrease is proportional to the magnitude of the gradient. As we also know, the further away we are from a minimum or maximum, the larger the gradient is at that point.We are able to deduce this from the monotonicity of gradient descent.Thus, the closer we get to a minimum or maximum, the smaller steps we take. Therefore, with each GD iteration, we take smaller and smaller steps, until we reach a point where $\|\nabla f(w)\| \leq \epsilon$.

From monotonicity, we know that $f(x_{i+1}) \leq f(x_i) - \frac{t}{2}\|\nabla f(x_i)\|^2$.

As we run iterations of GD, we get

$f(w_1) \leq f(w_0) - \frac{t}{2}\|\nabla f(w_0)\|^2$

$f(w_2) \leq f(w_1) - \frac{t}{2}\|\nabla f(w_1)\|^2$

$f(w_3) \leq f(w_2) - \frac{t}{2}\|\nabla f(w_2)\|^2$

and so on.

We add up these inequalitites to get

$f(w_1) + f(w_2) + f(w_3) \leq f(w_0) - \frac{t}{2}\|\nabla f(w_0)\|^2 + f(w_1) - \frac{t}{2}\|\nabla f(w_1)\|^2 + f(w_2) - \frac{t}{2}\|\nabla f(w_2)\|^2$

$f(w_3) \leq f(w_0) - \frac{t}{2}(\|\nabla f(w_0)\|^2 + \|\nabla f(w_1)\|^2 + \|\nabla f(w_2)\|^2)$

We generalize this for i iterations.

$f(w_i) \leq f(w_0) - \frac{t}{2}(\sum_{n=0}^{i=i-1}(\|\nabla f(w_n)\|^2)$

$f(w_i) - f(w_0) \leq -\frac{t}{2}(\sum_{n=0}^{i=i-1}(\|\nabla f(w_n)\|^2)$
$\sum_{n=0}^{i=i-1}\|\nabla f(w_n)\|^2 \leq \frac{2}{t}(f(w_0) - f(w_i))$
The minimum value that $\|\nabla f(w_n)\|$ can take on is ϵ, so
$i * \epsilon^2 \leq \frac{2}{t}(f(w_0) - f(w_i))$
Thus
$i \leq \frac{2(f(w_0)-f(w_i))}{t*\epsilon^2}$
We want i such that $f(w_i)$ is essentially the minimum, so
$i \leq \frac{2(f(w_0)-f(w))}{t*\epsilon^2}$
Thus, GD would take $\frac{2(f(w_0)-f(w))}{t*\epsilon^2}$ iterations to find such a w.

(2)

From lecture, we know that for a β-smooth and convex function,

$f(x_{k+1}) - f(x^*) \leq \frac{\beta}{2}(\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2)$

$f(x_k) - f(x^*) \leq \frac{\beta}{2}(\|x_{k-1} - x^*\|^2 - \|x_k - x^*\|^2)$

We are given

$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\alpha}{2}\|y - x\|_2^2$

Substituting in $y = x_k$ and $x = x^*$:

$f(x_k) \geq f(x^*) + \langle \nabla f(x^*), x_k - x^* \rangle + \frac{\alpha}{2}\|x_k - x^*\|_2^2$

$f(x_k) - f(x^*) \geq \frac{\alpha}{2}\|x_k - x^*\|_2^2$

$\frac{\alpha}{2}\|x_k - x^*\|_2^2 \leq f(x_k) - f(x^*)$

We substitute in the equation from lecture:

$\frac{\alpha}{2}\|x_k - x^*\|_2^2 \leq \frac{\beta}{2}(\|x_{k-1} - x^*\|^2 - \|x_k - x^*\|^2)$

$\frac{\alpha}{\beta}\|x_k - x^*\|_2^2 \leq \|x_{k-1} - x^*\|^2 - \|x_k - x^*\|^2$

$\frac{\alpha}{\beta}\|x_k - x^*\|_2^2 + \|x_k - x^*\|^2 \leq \|x_{k-1} - x^*\|^2$

$(1 + \frac{\alpha}{\beta})\|x_k - x^*\|_2^2 \leq \|x_{k-1} - x^*\|^2$

$\|x_k - x^*\|_2^2 \leq (1 + \frac{\alpha}{\beta})^{-1}\|x_{k-1} - x^*\|^2$

As we iterate GD:

$\|x_k - x^*\|_2^2 \leq (1 + \frac{\alpha}{\beta})^{-1}\|x_{k-1} - x^*\|^2$

$\|x_k - x^*\|_2^2 \leq (1 + \frac{\alpha}{\beta})^{-1}(1 + \frac{\alpha}{\beta})^{-1}\|x_{k-2} - x^*\|^2$

...

$\|x_k - x^*\|_2^2 \leq \|x_0 - x^*\|^2 \prod_{i=0}^{k-1}(1 + \frac{\alpha}{\beta})^{-1}$

Giving us

$\|x_k - x^*\|_2^2 \leq (1 + \frac{\alpha}{\beta})^{-k}\|x_0 - x^*\|^2$

as desired.

(3)

Suppose f is convex, and x^* is a local minimum in some interval W of f. This means for all $x \in W$, $f(x) \geq f(x^*)$. Now say we have some point y such that $f(y) < f(x^*)$. From convexity, we know that

$f(\lambda x^* + (1-\lambda)y) \leq \lambda f(x^*) + (1-\lambda)f(y)$.

Since $f(y) < f(x^*)$ must be true,

$f(\lambda x^* + (1-\lambda)y) \leq \lambda f(x^*) + (1-\lambda)f(y) < f(\lambda x^* + (1-\lambda)x^*) \leq \lambda f(x^*) + (1-\lambda)f(x^*)$.

We simplify this expression to get the following:

$f(\lambda x^* + (1-\lambda)y) < f(x^*)$.

But, since $\lambda \in [0,1]$, and x^* is a local minimum, $f(\lambda x^* + (1-\lambda)y) > f(x^*)$ must hold. Therefore, we have a contradiction, so x^* must also be a global minimum.

(4a)

Taking the derivative for $\sigma(\langle w, x_i \rangle)$:

$\sigma(\langle w, x_i \rangle) = (1 + e^{-\langle w, x_i \rangle})^{-1}$

$\sigma'(\langle w, x_i \rangle) = -(1 + e^{-\langle w, x_i \rangle})^{-2}(-x_i e^{-\langle w, x_i \rangle})$

$\sigma'(\langle w, x_i \rangle) = \frac{x_{ik} e^{-\langle w, x_i \rangle}}{(1+e^{-\langle w, x_i \rangle})^2}$

$\sigma'(\langle w, x_i \rangle) = (x_{ik})(\sigma(\langle w, x_i \rangle)) * \frac{e^{-\langle w, x_i \rangle}}{1+e^{-\langle w, x_i \rangle}}$

$\sigma'(\langle w, x_i \rangle) = (x_{ik})(\sigma(\langle w, x_i \rangle))(1 - \sigma(\langle w, x_i \rangle))$

We can now find the gradient of $L(w)$:

$L(w) = (\frac{1}{n}) \sum_{i=1}^{n} -y_i log(\sigma(\langle w, x_i \rangle)) - (1 - y_i) log(1 - \sigma(\langle w, x_i \rangle))$

$\nabla L(w) = (\frac{1}{n}) \sum_{i=1}^{n} -y_i \frac{\sigma'(\langle w, x_i \rangle)}{\sigma(\langle w, x_i \rangle)} - (1 - y_i) \frac{-\sigma'(\langle w, x_i \rangle)}{1-\sigma(\langle w, x_i \rangle)}$

$\nabla L(w) = (\frac{1}{n}) \sum_{i=1}^{n} -y_i x_{ik}(1-\sigma(\langle w, x_i \rangle)) + (1-y_i)(x_{ik}\sigma(\langle w, x_i \rangle))$

$\nabla L(w) = (\frac{1}{n}) \sum_{i=1}^{n} -y_i x_{ik}(1-\sigma(\langle w, x_i \rangle)) + x_{ik}\sigma(\langle w, x_i \rangle) + y_i x_{ik}\sigma(\langle w, x_i \rangle)$

$\nabla L(w) = (\frac{1}{n}) \sum_{i=1}^{n} x_{ik}(-y_i + y_i\sigma(\langle w, x_i \rangle) + \sigma(\langle w, x_i \rangle) - y_i\sigma(\langle w, x_i \rangle))$

$\nabla L(w) = (\frac{1}{n}) \sum_{i=1}^{n} x_{ik}(\sigma(\langle w, x_i \rangle) - y_i)$

(4b)

The code below was used for running the gradient descent algorithms.

```
def gradient_descent(xinit,steps,gradient):
    """Run gradient descent.
    Return an array with the rows as the iterates.
    """
    xs = [xinit]
    x = xinit
    for step in steps:
        x = x - step*gradient(x)
        xs.append(x)
    return np.array(xs)

def nagd(winit,gradient,eta=0.1,nsteps=100):
    """Run Nesterov's accelrated graident descent.
    Return an array with the rows as the iterates.
    """
    ws = [winit]
    u = v = w = winit
    for i in range(nsteps):
        etai = (i+1)*eta/2
        alphai = 2/(i+3)
        w = v - eta*gradient(v)
        u = u - etai*gradient(v)
        v = alphai*u + (1-alphai)*w
        ws.append(w)
    return np.array(ws)
```

```
def lr_cost(X,y,w):
    n,d = X.shape
    return (1/n)*sum(-y*np.log(sigmoid(X.dot(w)))-((1-y)*np.log(1-sigmoid(X.dot(w)))))

def lr_gradient(X,y,w):
    n,d = X.shape
    return (1/n) * X.T.dot(sigmoid(X.dot(w))-y)
```

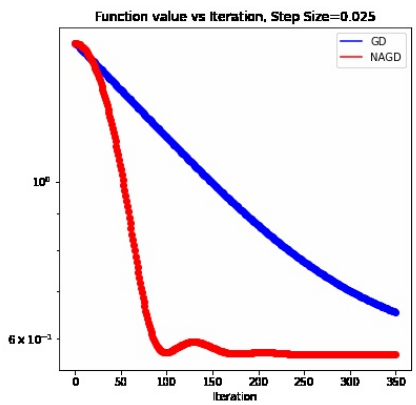
```python
# use step sizes 0.01, 0.05, 0.1 for 350 iterations
X = dataset[0]
y = dataset[1].flatten()

objective = lambda w: lr_cost(X, y, w)
gradient = lambda w: lr_gradient(X, y, w)

w0 = np.random.normal(0, 1, d)
```

```python
gd_1 = gradient_descent(w0, [0.025]*350, gradient)
nagd_1 = nagd(w0,gradient,0.01,350)

gd_2 = gradient_descent(w0, [0.05]*350, gradient)
nagd_2 = nagd(w0,gradient,0.05,350)

gd_3 = gradient_descent(w0, [0.075]*350, gradient)
nagd_3 = nagd(w0,gradient,0.1,350)
```

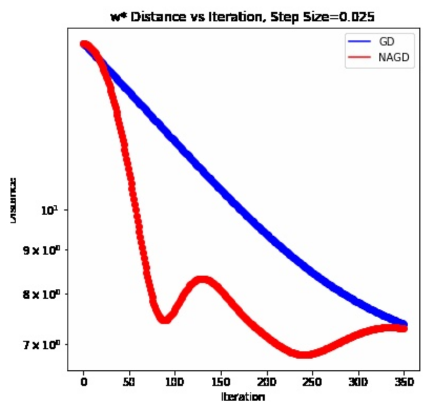
The following portion of code was used to generate plots for both GD and NAGD, with a separate plot for each step size.

```python
animated_lplot(Ys=[[objective(w) for w in gd_1],[objective(w) for w in nagd_1]],labels=['GD','NAGD'], \
               title='Function value vs Iteration, Step Size=0.025', name='gd_1_fVal.gif')
animated_lplot(Ys=[[np.linalg.norm(dataset[2]-w) for w in gd_1],[np.linalg.norm(dataset[2]-w) for w in nagd_1]],\
               labels=['GD','NAGD'], ylabel="Distance", title='w* Distance vs Iteration, Step Size=0.025', \
               name='gd_1_w.gif')

animated_lplot(Ys=[[objective(w) for w in gd_2],[objective(w) for w in nagd_2]],labels=['GD','NAGD'], \
               title='Function value vs Iteration, Step Size=0.05', name='gd_2_fVal.gif')
animated_lplot(Ys=[[np.linalg.norm(dataset[2]-w) for w in gd_2],[np.linalg.norm(dataset[2]-w) for w in nagd_2]],\
               labels=['GD','NAGD'], ylabel="Distance", title='w* Distance vs Iteration, Step Size=0.05', \
               name='gd_2_w.gif')

animated_lplot(Ys=[[objective(w) for w in gd_3],[objective(w) for w in nagd_3]],labels=['GD','NAGD'], \
               title='Function value vs Iteration, Step Size=0.075', name='gd_3_fVal.gif')
animated_lplot(Ys=[[np.linalg.norm(dataset[2]-w) for w in gd_3],[np.linalg.norm(dataset[2]-w) for w in nagd_3]],\
               labels=['GD','NAGD'], ylabel="Distance", title='w* Distance vs Iteration, Step Size=0.075', \
               name='gd_3_w.gif')
```

It produced the following plots.

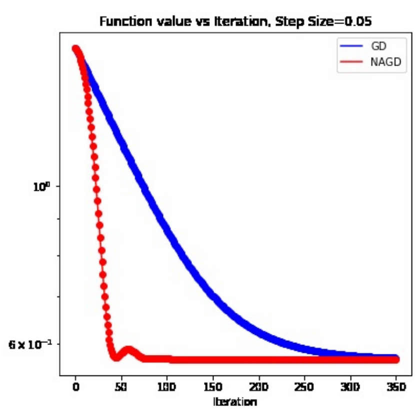


(a) Iteration vs f

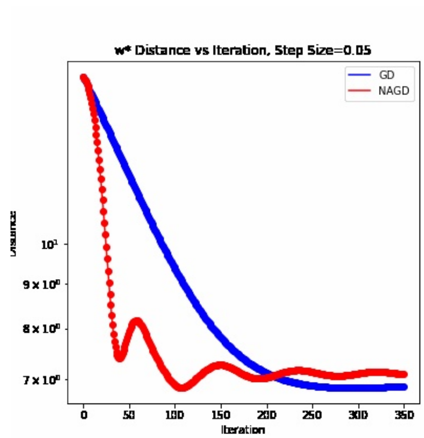


(b) Distance from w*

Figure 1: Plots for Step Size 0.025

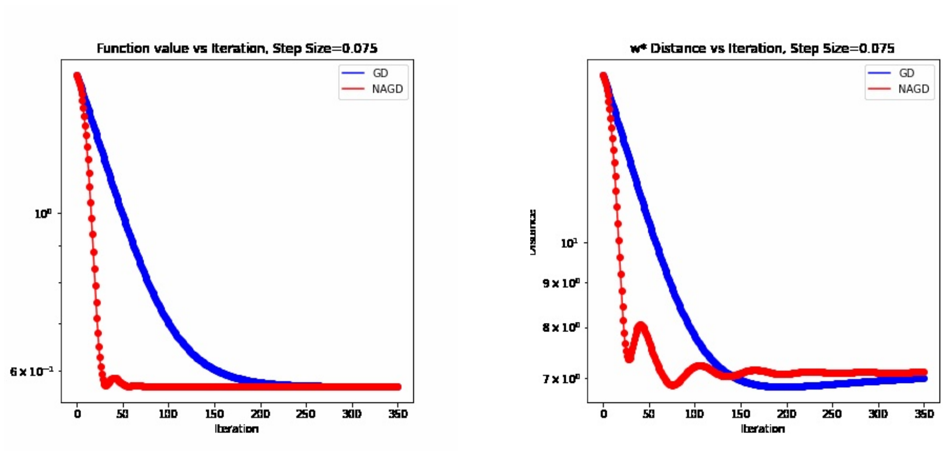


(a) Iteration vs f



(b) Distance from w*

Figure 2: Plots for Step Size 0.05



(a) Iteration vs f

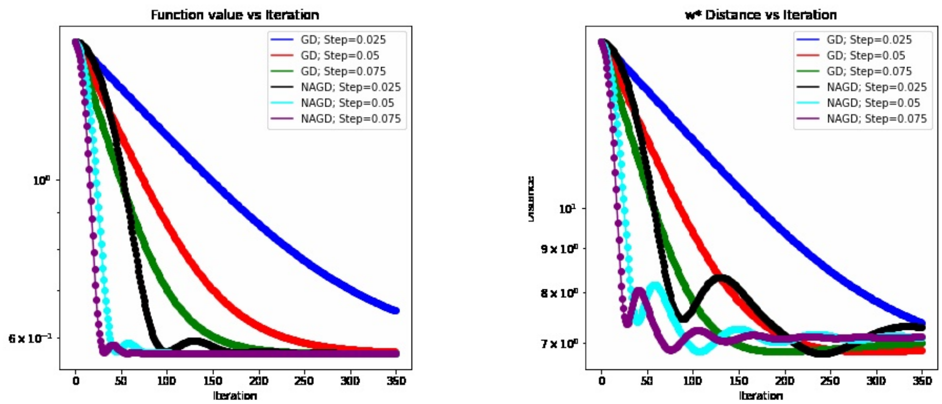
(b) Distance from w*

Figure 3: Plots for Step Size 0.075

For visualization purposes, I plotted all NAGD and GD iterations on the same plot with the following code.

```
animated_lplot(Ys=[[objective(w) for w in gd_1],[objective(w) for w in gd_2],[objective(w) for w in gd_3],\
                   [objective(w) for w in nagd_1],[objective(w) for w in nagd_2],[objective(w) for w in nagd_3]],\
               labels=['GD; Step=0.025','GD; Step=0.05','GD; Step=0.075','NAGD; Step=0.025','NAGD; Step=0.05',\
                       'NAGD; Step=0.075'], title='Function value vs Iteration', name='All_fVal.gif')

animated_lplot(Ys=[[np.linalg.norm(dataset[2]-w) for w in gd_1],[np.linalg.norm(dataset[2]-w) for w in gd_2],\
                   [np.linalg.norm(dataset[2]-w) for w in gd_3],[np.linalg.norm(dataset[2]-w) for w in nagd_1],\
                   [np.linalg.norm(dataset[2]-w) for w in nagd_2],[np.linalg.norm(dataset[2]-w) for w in nagd_3]],\
               labels=['GD; Step=0.025','GD; Step=0.05','GD; Step=0.075','NAGD; Step=0.025','NAGD; Step=0.05',\
                       'NAGD; Step=0.075'], ylabel="Distance", title='w* Distance vs Iteration', name='All_w.gif')
```

This resulted in the following plots.



(a) Iteration vs f

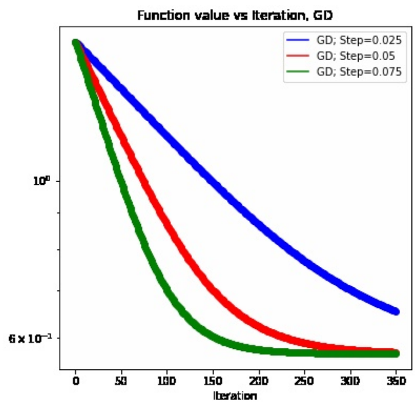
(b) Distance from w*

Figure 4: GD and NAGD for various step sizes

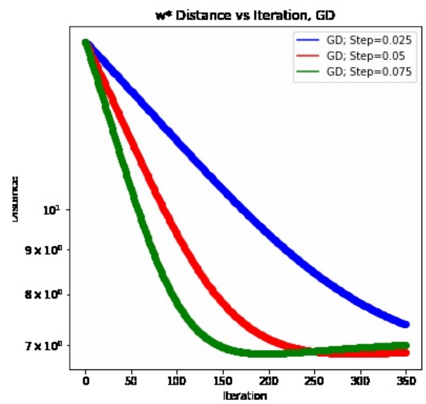
This was very busy too look at, so I separated the GD and NAGD iterations into separate graphs.

```
animated_lplot(Ys=[[objective(w) for w in gd_1],[objective(w) for w in gd_2],[objective(w) for w in gd_3]],\
               labels=['GD; Step=0.025','GD; Step=0.05','GD; Step=0.075'], title='Function value vs Iteration, GD', \
               name='gd_All_fVal.gif')

animated_lplot(Ys=[[objective(w) for w in nagd_1],[objective(w) for w in nagd_2],[objective(w) for w in nagd_3]],\
               labels=['NAGD; Step=0.025','NAGD; Step=0.05','NAGD; Step=0.075'], \
               title='Function value vs Iteration, NAGD', name='nagd_All_fVal.gif')

animated_lplot(Ys=[[np.linalg.norm(dataset[2]-w) for w in gd_1],[np.linalg.norm(dataset[2]-w) for w in gd_2],\
                   [np.linalg.norm(dataset[2]-w) for w in gd_3]],labels=['GD; Step=0.025','GD; Step=0.05',\
                                                                       'GD; Step=0.075'], ylabel="Distance", \
               title='w* Distance vs Iteration, GD', name='gd_All_w.gif')

animated_lplot(Ys=[[np.linalg.norm(dataset[2]-w) for w in nagd_1],[np.linalg.norm(dataset[2]-w) for w in nagd_2],\
                   [np.linalg.norm(dataset[2]-w) for w in nagd_3]],labels=['NAGD; Step=0.025','NAGD; Step=0.05',\
                                                                         'NAGD; Step=0.075'], ylabel="Distance", \
               title='w* Distance vs Iteration, NAGD', name='nagd_All_w.gif')
```

This resulted in the following plots.

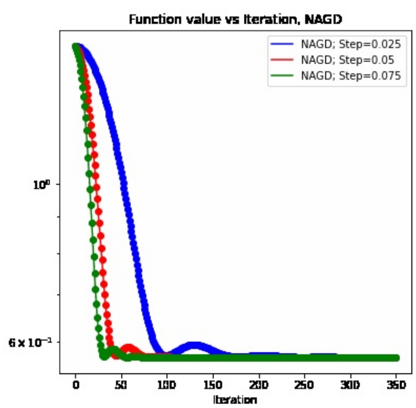


(a) Iteration vs f

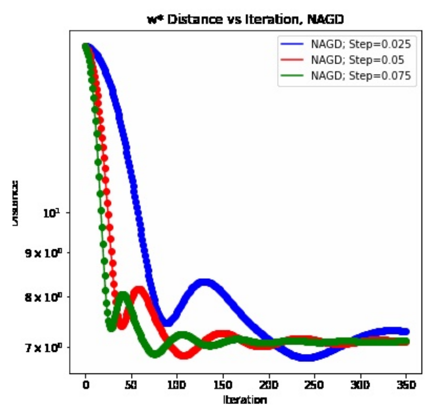


(b) Distance from w*

Figure 5: GD Plots



(a) Iteration vs f



(b) Distance from w*

Figure 6: NAGD Plots