1. Lemma: A μ-strongly convex function $f: R^n \to R$ has $\lim_{\|x\| \to \infty} f(x) = \infty$.

5/10

Proof: By the definition of μ-strongly convex, we have

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

$$= f(x) + \nabla f^T(x)(y-x) + \frac{\mu}{2}\|y\|^2 - \mu x^T y + \mu\|x\|^2 - \frac{\mu}{2}\|x\|^2$$

$$= \frac{\mu}{2}\|y\|^2 + (\nabla f(x) - \mu x)^T (y-x) + f(x) - \frac{\mu}{2}\|x\|^2$$

$$= \frac{\mu}{2}\|y\|^2 + (\nabla f(x) - \mu x)^T y + f(x) - \frac{\mu}{2}\|x\|^2 - (\nabla f(x) - \mu x)^T x$$

Using Cauchy-Schwarz, $(\nabla f(x) - \mu x)^T y \geq -\|\nabla f(x) - \mu x\| \|y\|$.

$$\geq \frac{\mu}{2}\|y\|^2 - \|\nabla f(x) - \mu x\|\|y\| + \underbrace{f(x) - \frac{\mu}{2}\|x\|^2 - (\nabla f(x) - \mu x)^T x}_{\text{Constant only depends on } x}$$

$\therefore$ When $\|y\| \to \infty$, $f(y) \to \infty$. (Q.E.D).

Let's suppose the f doesn't have a minimizer, and define $f^* = \inf_{x \in R^n} f(x)$ and assume f is upper bounded. By the definition of infimum, there exists a sequence $\{x_k\}$ s.t. $f(x_k) \to f^*$. We now have 2 mutually exclusive cases:

this is not a valid assumption

① $\sup \|x_k\| = b < \infty$. Then all $x_k \in \{R^n \cap B\}$ where $B := \{x \in R^n \mid \|x\| < b\}$. Since B is compact, a subsequence of $\{x_k\}$ conveges to x^* s.t. $f(x^*) = f^*$, yielding contradiction.

② $\sup \|x_k\| = \infty$. Then because of the Lemma above, $\|x_k\| \to \infty$ and $f(x) \to \infty$ contradicting the assumption that $f(x)$ is upper bounded.

Therefore, f must have a minimizer x^*.

7/10

Exercise 4.2. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex, L-smooth function that has a minimizer $x^\star$. Suppose we use random coordinate choice as a stochastic gradient oracle, with step lengths $\alpha_k = \frac{1}{nL}$. Let the (random) iterates be $x_0, x_1, x_2, \ldots$. Assume that the level set $\mathcal{L} := \{x \in \mathbb{R}^n : f(x) \le f(x_0)\}$ is bounded and has diameter R, i.e., $\|x - y\| \le R$ for all $x, y \in \mathcal{L}$. Show that for any $T \ge 1$,

$$\mathbb{E}[f(x_T) - f^\star] \le \frac{2LnR^2}{T}.$$

[Hint: Try to adapt the deterministic analysis (Theorem 2.1 in the "Smooth Convex Optimization" lecture notes) and take expectations in an appropriate way.]

Ans:

Notice we use random coordinate choice as a stochastic gradient oracle with step length $\alpha_k = \frac{1}{nL}$, we know that the update rule would be:

$$a_k = n \frac{\partial f}{\partial x_{ik}} e^{ik}$$

$$x_{k+1} = x_k - \alpha_k a_k = x_k - \frac{1}{L} \frac{\partial f}{\partial x_{ik}} e^{ik}$$

Notice that $f: \mathbb{R}^n \to \mathbb{R}$ is a L-smooth function, we will have following property:

$$f(x_{k+1}) \le f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k \rangle + \frac{L}{2} \|x_{k+1} - x_k\|^2$$

Since $x_{k+1} - x_k = -\frac{1}{L} \frac{\partial f}{\partial x_{ik}} e^{ik}$

$$f(x_{k+1}) \le f(x_k) + \langle \nabla f(x_k), -\frac{1}{L} \frac{\partial f}{\partial x_{ik}} e^{ik} \rangle + \frac{L}{2} \left\| \frac{1}{L} \frac{\partial f}{\partial x_{ik}} e^{ik} \right\|^2$$

$$= f(x_k) - \frac{1}{L} \left(\frac{\partial f}{\partial x_{ik}} \right)^2 + \frac{1}{2L} \left(\frac{\partial f}{\partial x_{ik}} \right)^2$$

$$= f(x_k) - \frac{1}{2L}\left(\frac{\partial f}{\partial x_{ik}}\right)^2$$

we define $\Delta_k = f(x_k) - f^* = f(x_k) - f(x^*)$ where x^* is the minimizer of f. Thus $f(x_k) - f(x^*) \geq 0 \quad \forall k$. subtract $f(x^*)$ from both side:

$$f(x_{k+1}) - f(x^*) \leq f(x_k) - f(x^*) - \frac{1}{2L}\left(\frac{\partial f}{\partial x_{ik}}\right)^2$$

$$\Rightarrow \quad \Delta_{k+1} \leq \Delta_k - \frac{1}{2L}\left(\frac{\partial f}{\partial x_{ik}}\right)^2$$

we now take a conditional expectation on both side, conditioned on x_k

$$\mathbb{E}[\Delta_{k+1} | x_k] \leq \Delta_k - \frac{1}{2L}\mathbb{E}_{ik}\left[\left(\frac{\partial f}{\partial x_{ik}}\right)^2\right]$$

Since $P(e^i) = \frac{1}{n}$ and $\mathbb{E}_{ik}\left[\left(\frac{\partial f}{\partial x_{ik}}\right)^2\right] = \frac{1}{n}\sum_{i=0}^{n}\left(\frac{\partial f}{\partial x_{ik}}\right)^2$

$$\mathbb{E}[\Delta_{k+1} | x_k] \leq \Delta_k - \frac{1}{2Ln}\sum_{i=0}^{n}\left(\frac{\partial f}{\partial x_{ik}}\right)^2$$

$$= \Delta_k - \frac{1}{2Ln}\|\nabla f(x_k)\|^2$$

We now take an expectation over x_k

$$\mathbb{E}[\Delta_{k+1}] \leq \mathbb{E}[\Delta_k] - \frac{1}{2Ln}\mathbb{E}[\|\nabla f(x_k)\|^2]$$

since $\mathrm{Var}[z] = \mathbb{E}[z^2] - \mathbb{E}[z]^2 \geq 0$

thus $\mathbb{E}[\|\nabla f(x_k)\|^2] \geq \mathbb{E}[\|\nabla f(x_k)\|]^2$

$$\Rightarrow \mathbb{E}[\Delta_{k+1}] \leq \mathbb{E}[\Delta_k] - \frac{1}{2Ln}\mathbb{E}[\|\nabla f(x_k)\|]^2 \quad -(1)$$

Notice that $f: \mathbb{R}^n \to \mathbb{R}$ is a convex function, it implies the following inequality:

$$f(y) - f(x) \leq \langle \nabla f(y) - \nabla f(x), y - x \rangle$$

if we set $y = x_k$, $x = x^*$, then we can get

$$\Delta_k := f(x_k) - f(x^*) \leq \langle \nabla f(x_k), x_k - x^* \rangle$$

$$\Rightarrow \Delta_k \leq \langle \nabla f(x_k), x_k - x^* \rangle \leq \|\nabla f(x_k)\| \, \|x_k - x^*\|$$

Notice that level set $\mathcal{L}$ is bounded with diameter R Thus $\|x_k - x^*\| \leq R$, then we can get the further relationship for Δ_k (-3)

$$\mathbb{E}[\Delta_k] \leq R\,\mathbb{E}[\|\nabla f(x_k)\|]$$

R is the diameter for the set $\{x : f(x) \leq f(x_0)\}$.

You need to carefully argue it in expectation by connecting x_k to x_0.

$$\Rightarrow -\frac{\mathbb{E}[\Delta_k]^2}{R^2} \geq -\mathbb{E}[\|\nabla f(x_k)\|]^2 \quad \text{— ②}$$

Combine ① and ②, we can rewrite ① as

$$\mathbb{E}[\Delta_{k+1}] \leq \mathbb{E}[\Delta_k] - \frac{1}{2Ln R^2}\mathbb{E}[\Delta_k]^2$$

Divide both side by $\mathbb{E}[\Delta_{k+1}]\mathbb{E}[\Delta_k]$, we get

$$\frac{1}{\mathbb{E}[\Delta_{k+1}]} \geq \frac{1}{\mathbb{E}[\Delta_k]} + \frac{1}{2Ln R^2}\frac{\mathbb{E}[\Delta_k]}{\mathbb{E}[\Delta_{k+1}]}$$

Since $\mathbb{E}[\Delta_{k+1}] \leq \mathbb{E}[\Delta_k] - \frac{1}{2Ln R^2}\mathbb{E}[\Delta_k]^2$

$$\Rightarrow \frac{\mathbb{E}[\Delta_k]}{\mathbb{E}[\Delta_{k+1}]} \geq 1 + \frac{1}{2Ln R^2}\frac{\mathbb{E}[\Delta_k]^2}{\mathbb{E}[\Delta_{k+1}]} \geq 1$$

Thus

$$\frac{1}{\mathbb{E}[\Delta_{k+1}]} \geq \frac{1}{\mathbb{E}[\Delta_k]} + \frac{1}{2Ln R^2}$$

Sum from $k=0$ to $k=T-1$, we get

$$\frac{1}{\mathbb{E}[\Delta_T]} \geq \frac{1}{\mathbb{E}[\Delta_0]} + \frac{T}{2Ln R^2}$$

Notice $\Delta_0 = f(x_0) - f(x^*)$ and x_0 is arbitrary point $\mathbb{E}[\Delta_0] = \Delta_0$

Thus

$$\frac{1}{\mathbb{E}[\Delta_T]} \geq \frac{1}{\Delta_0} + \frac{T}{2Ln R^2}$$

Since $\Delta_0 = f(x_0) - f(x^*) > 0$

thus

$$\frac{1}{\mathbb{E}[\Delta_T]} \geq \frac{1}{\Delta_0} + \frac{T}{2Ln R^2} \geq \frac{T}{2Ln R^2}$$

$$\Rightarrow \mathbb{E}[f(x_T) - f(x^*)] \leq \frac{2Ln R^2}{T}$$

/6 /10

4.3 (ii)

When the dimension n and the condition number of H are fixed, the performance of each method is as follows. **Cyclic order with exact linesearch** method takes the smallest number of iterations to get to the minimum. This is because at each coordinate direction, the step size is "optimal" in that it results in maximum function value decrease. **Cyclic order with fixed step size** and **random order with fixed step size** have very much similar number of iterations which is much larger than that of the **cyclic order with exact linesearch** method. For **cyclic order with fixed step size**, because its complexity is $O(n^2)$, it is expected to be larger. But for **random order with fixed step size**, its complexity is just $O(n)$, indicating it should take less iterations than the **cyclic order with fixed step size**. However, this is probably all theoretical and expectational, while in practice, when running the Matlab program, the performance of this method is not that optimal as the theory. **Gauss-Southwell with fixed step size** takes smaller number of iterations than the previous two. This is because in each iteration, this method goes towards the steepest descent coordinate and its complexity is $O(n)$.

When the dimension n increases, the number of iterations of these 4 methods all increase because their convergence rate depends on n and increasing dimension makes the optimization problem much more complicated to solve.

When the condition number of H increases, the number of iterations of these 4 methods also all increase because the larger the condition number of H is, the more singular the H is, indicating the less convex the f is. Therefore, more iterations are needed.

5/5

```
function [x, F, G, H, iter] = cyclic_exact(H, x0)

n = size(H,1);
iter = 0;
x = x0;
G = H * x0;
tol = 1e-6 * max(norm(G),1);

while norm(G) > tol

    i = mod(iter,n) + 1;

    temp = zeros(n,1); % column
    temp(i) = 1;
    e_i = temp;

    alpha = ((x')*H*e_i) / ((e_i')*H*e_i);

    x = x - alpha * e_i;

    iter = iter + 1;

    F = 0.5 * x' * H * x;
    G = H * x;
end
```

```
Not enough input arguments.

Error in cyclic_exact (line 3)
n = size(H,1);
```

Published with MATLAB® R2018b

5 / 5

```
function [x, F, G, H, iter] = cyclic_fixed_step(H, x0)

n = size(H,1);

iter = 0;

x = x0;
G = H * x0;
tol = 1e-6 * max(norm(G),1);

alpha = 1/normest(H);

while norm(G) > tol

    i = mod(iter,n) + 1;

    temp = zeros(n,1);
    temp(i) = 1;
    e_i = temp;

    x = x - alpha*G(i)*e_i;

    iter = iter + 1;

    F = 0.5 * x' * H * x;
    G = H * x;
end
```

```
Not enough input arguments.

Error in cyclic_fixed_step (line 3)
n = size(H,1);
```

Published with MATLAB® R2018b

5 / 5

```
function [x, F, G, H, iter] = random_fixed_step(H, x0)

n = size(H,1);

iter = 0;

x = x0;
G = H * x0;
tol = 1e-6 * max(norm(G),1);

H_norm = normest(H);
alpha = 1/H_norm;

while norm(G) > tol

    i = randi(n);

    temp = zeros(n,1);
    temp(i) = 1;
    e_i = temp;

    x = x - alpha*G(i)*e_i;

    iter = iter + 1;

    F = 0.5 * x' * H * x;
    G = H * x;
end
```

```
Not enough input arguments.

Error in random_fixed_step (line 3)
n = size(H,1);
```

Published with MATLAB® R2018b

5/5

```
function [x, F, G, H, iter] = gs_fixed_step(H, x0)

n = size(H,1);

iter = 0;

x = x0;
G = H * x0;
tol = 1e-6 * max(norm(G),1);

H_norm = normest(H);
alpha = 1/H_norm;

while norm(G) > tol

    G_abs = abs(G);
    i = find(G_abs == max(G_abs));

    temp = zeros(n,1);
    temp(i) = 1;
    e_i = temp;

    x = x - alpha*G(i)*e_i;

    iter = iter + 1;

    F = 0.5 * x' * H * x;
    G = H * x;
end
```

[~, i] = max(abs(G))

✓

```
Not enough input arguments.

Error in gs_fixed_step (line 3)
n = size(H,1);
```

Published with MATLAB® R2018b

```
% for n = [10, 100, 1000]
n = 10;

x0 = ones(n,1);

for condnum = [10,100,1000,10000]
    H = sprandsym(n, 1, 1/condnum, 1);

    fprintf('The efficiency of coordinate minimization algorithms at n
 = %d and cond(H) = %d: \n', n, condnum)

    [x, F, G, H, iter] = cyclic_exact(H, x0);
    fprintf('Cyclic order and exact linesearch: %d iterations.\n',
 iter)

    [x, F, G, H, iter] = cyclic_fixed_step(H, x0);
    fprintf('Cyclic order and fixed step: %d iterations.\n', iter)

    [x, F, G, H, iter] = random_fixed_step(H, x0);
    fprintf('Random order and fixed step: %d iterations.\n', iter)

    [x, F, G, H, iter] = gs_fixed_step(H, x0);
    fprintf('Gauss-Southwell Rule and fixed step: %d iterations.\n\n
\n', iter)

end
% end
```

```
The efficiency of coordinate minimization algorithms at n = 10 and
 cond(H) = 10:
Cyclic order and exact linesearch: 166 iterations.
Cyclic order and fixed step: 898 iterations.
Random order and fixed step: 1077 iterations.
Gauss-Southwell Rule and fixed step: 516 iterations.


The efficiency of coordinate minimization algorithms at n = 10 and
 cond(H) = 100:
Cyclic order and exact linesearch: 1086 iterations.
Cyclic order and fixed step: 6588 iterations.
Random order and fixed step: 7000 iterations.
Gauss-Southwell Rule and fixed step: 5327 iterations.


The efficiency of coordinate minimization algorithms at n = 10 and
 cond(H) = 1000:
Cyclic order and exact linesearch: 3274 iterations.
Cyclic order and fixed step: 69943 iterations.
Random order and fixed step: 71570 iterations.
Gauss-Southwell Rule and fixed step: 25245 iterations.
```

```
The efficiency of coordinate minimization algorithms at n = 10 and
 cond(H) = 10000:
Cyclic order and exact linesearch: 7329 iterations.
Cyclic order and fixed step: 432720 iterations.
Random order and fixed step: 443738 iterations.
Gauss-Southwell Rule and fixed step: 169043 iterations.
```

Published with MATLAB® R2018b

```
% for n = [10, 100, 1000]
n = 100;

x0 = ones(n,1);

for condnum = [10,100,1000,10000]
    H = sprandsym(n, 1, 1/condnum, 1);

    fprintf('The efficiency of coordinate minimization algorithms at n
 = %d and cond(H) = %d: \n', n, condnum)

    [x, F, G, H, iter] = cyclic_exact(H, x0);
    fprintf('Cyclic order and exact linesearch: %d iterations.\n',
 iter)

    [x, F, G, H, iter] = cyclic_fixed_step(H, x0);
    fprintf('Cyclic order and fixed step: %d iterations.\n', iter)

    [x, F, G, H, iter] = random_fixed_step(H, x0);
    fprintf('Random order and fixed step: %d iterations.\n', iter)

    [x, F, G, H, iter] = gs_fixed_step(H, x0);
    fprintf('Gauss-Southwell Rule and fixed step: %d iterations.\n\n
\n', iter)

end
% end
```

```
The efficiency of coordinate minimization algorithms at n = 100 and
 cond(H) = 10:
Cyclic order and exact linesearch: 1847 iterations.
Cyclic order and fixed step: 9476 iterations.
Random order and fixed step: 11922 iterations.
Gauss-Southwell Rule and fixed step: 4926 iterations.


The efficiency of coordinate minimization algorithms at n = 100 and
 cond(H) = 100:
Cyclic order and exact linesearch: 8327 iterations.
Cyclic order and fixed step: 72659 iterations.
Random order and fixed step: 77193 iterations.
Gauss-Southwell Rule and fixed step: 36739 iterations.


The efficiency of coordinate minimization algorithms at n = 100 and
 cond(H) = 1000:
Cyclic order and exact linesearch: 88133 iterations.
Cyclic order and fixed step: 602161 iterations.
Random order and fixed step: 638892 iterations.
Gauss-Southwell Rule and fixed step: 310352 iterations.
```

```
The efficiency of coordinate minimization algorithms at n = 100 and
 cond(H) = 10000:
Cyclic order and exact linesearch: 276164 iterations.
Cyclic order and fixed step: 3986100 iterations.
Random order and fixed step: 4234998 iterations.
Gauss-Southwell Rule and fixed step: 1938563 iterations.
```

Published with MATLAB® R2018b

```
% for n = [10, 100, 1000]
% n = 1000;

x0 = ones(n,1);

for condnum = [10,100,1000,10000]
    H = sprandsym(n, 1, 1/condnum, 1);

    fprintf('The efficiency of coordinate minimization algorithms at n
 = %d and cond(H) = %d: \n', n, condnum)

    [x, F, G, H, iter] = cyclic_exact(H, x0);
    fprintf('Cyclic order and exact linesearch: %d iterations.\n',
 iter)

    [x, F, G, H, iter] = cyclic_fixed_step(H, x0);
    fprintf('Cyclic order and fixed step: %d iterations.\n', iter)

    [x, F, G, H, iter] = random_fixed_step(H, x0);
    fprintf('Random order and fixed step: %d iterations.\n', iter)

    [x, F, G, H, iter] = gs_fixed_step(H, x0);
    fprintf('Gauss-Southwell Rule and fixed step: %d iterations.\n\n
\n', iter)

end
```

It takes extremely long time to get the publishable PDF file so I just ran it and paste the output here.

The efficiency of coordinate minimization algorithms at n = 1000 and cond(H) = 10:
Cyclic order and exact linesearch: 32302 iterations.
Cyclic order and fixed step: 129969 iterations.
Random order and fixed step: 154874 iterations.
Gauss-Southwell Rule and fixed step: 69132 iterations.

The efficiency of coordinate minimization algorithms at n = 1000 and cond(H) = 100:
Cyclic order and exact linesearch: 154195 iterations.
Cyclic order and fixed step: 1185919 iterations.
Random order and fixed step: 1251630 iterations.
Gauss-Southwell Rule and fixed step: 551524 iterations.

The efficiency of coordinate minimization algorithms at n = 1000 and cond(H) = 1000:
Cyclic order and exact linesearch: 562179 iterations.
Cyclic order and fixed step: 5836555 iterations.
Random order and fixed step: 6505786 iterations.
Gauss-Southwell Rule and fixed step: 2127534 iterations.

The efficiency of coordinate minimization algorithms at n = 1000 and cond(H) = 10000:
Cyclic order and exact linesearch: 4682629 iterations.
Cyclic order and fixed step: 41569904 iterations.
Random order and fixed step: 44364648 iterations.
Gauss-Southwell Rule and fixed step: 20278002 iterations.