Advanced Algorithms (67824) – Exercise 2

Edan Orzech 322362849, Mike Greenbaum 211747639

November 2021

1. First, notice that in each iteration of the simplex algorithm, we get a set of indices. Notice that each set is of size between $0, \ldots, n$ and therefore there are at most 2^n such sets. Denote this value by M. Notice that the corresponding vectors found by the simplex algorithm in the first $M+1$ iterations $v_1, \ldots, v_{M+1} \in Vert(\mathcal{P}(A))$ and by the pigeonhole principle $\exists i \neq j$ s.t. $v_i = v_j$. It means that the algorithm cycles between $v_i, \ldots, v_j$. Denote the sets in this cycle by $I_1, \ldots, I_{k-1}, I_k = I_1$. Let t the maximal index that enters a basis (set) in the cycle, namely the maximal $i \in [n]$ such that $i \in I_j \setminus I_{j-1}$ for some $j < k$. Because it is a cycle, there is an iteration in the cycle of bases in which t leaves a basis.

   Because no $t' > t$ enters or leaves a basis in the cycle, we can look at the matrix with only the first t columns, since $Ax = A_{[t]} \cdot x_{[t]} + A_{[n]\setminus[t]} \cdot x_{[n]\setminus[t]} = b \iff A_{[t]} \cdot x_{[t]} = b - A_{[n]\setminus[t]} \cdot x_{[n]\setminus[t]}$, and $A_{[n]\setminus[t]} \cdot x_{[n]\setminus[t]}$ will remain constant according to the algorithm.

   For a basis D, define the function $c_D(x) = \left(c^T - c^T_{B(D)} \cdot A^{-1}_{B(D)} \cdot A\right) x$. Then

$$\begin{aligned}
c_D(x - x_{N(D)}) &= \left(c^T - c^T_{B(D)} \cdot A^{-1}_{B(D)} \cdot A\right)\left(x - x_{N(D)}\right) \\
&= c^T\left(x - x_{N(D)}\right) - c^T_{B(D)} \cdot A^{-1}_{B(D)} \cdot A \cdot \left(x - x_{N(D)}\right) \\
&= c^T_{B(D)} x_{B(D)} - c^T_{B(D)} \cdot A^{-1}_{B(D)} \cdot A_{B(D)} x_{B(D)} \\
&= c^T_{B(D)} x_{B(D)} - c^T_{B(D)} \cdot x_{B(D)} = 0
\end{aligned}$$

   And

$$\begin{aligned}
c_D(x - x_{B(D)}) &= \left(c^T - c^T_{B(D)} \cdot A^{-1}_{B(D)} \cdot A\right)\left(x - x_{B(D)}\right) \\
&= c^T\left(x - x_{B(D)}\right) - c^T_{B(D)} \cdot A^{-1}_{B(D)} \cdot A \cdot \left(x - x_{B(D)}\right) \\
&= c^T_{N(D)} x_{N(D)} - c^T_{B(D)} \cdot A^{-1}_{B(D)} \cdot A_{N(D)} x_{N(D)} \\
&= \left(c^T_{N(D)} - c^T_{B(D)} \cdot A^{-1}_{B(D)} \cdot A_{N(D)}\right) \cdot x_{N(D)}
\end{aligned}$$

   Notice that we got the same objective function as in class up to adding a constant and therefore we can minimize it instead. Therefore, in our algorithm, we will use this as the reduced objective function.

   Define E to be the set of indices in the iteration where we choose t to enter the basis.

   Notice that as a vector of length t (or n wlog as explained earlier), $\forall i < t,\ [c_E]_i \geq 0$, since if $i \in B(E)$ then the algorithm makes sure that $[c_E]_i = 0$, and if $i \in N(E)$ and $[c_E]_i < 0$, the Bland rule would have chosen i as the pivot instead of t which is a contradiction. Therefore, $[c_E]_i \geq 0$. In addition, $[c_E]_t < 0$ because the algorithm chose t.

   Define L to be the set of indices in the iteration where we choose t to leave the basis. Denote with s the entering variable in this iteration, define $u = A^{-1}_{B(L)} \cdot A_s$.

   Therefore the pivot column in this iteration is $\begin{pmatrix} [c_L]_s \\ \left[A^{-1}_{B(L)} \cdot A\right]_s \end{pmatrix} = \begin{pmatrix} c^T_s - c^T_{B(L)} \cdot u \\ u \end{pmatrix}$. u is indeed the matching column of A, since $A^{-1}_{B(L)}$ applies Gauss elimination on A so that the columns in $B(L)$ become standard-basis columns. Notice that $c_s - c^T_{B(L)} \cdot u \overset{\star}{<} 0$ since the algorithm chooses s in this iteration.

Define $f(i) = j$ if the j'th entry in L is i. Note that $|L| = m$.

Due to t being chosen as the leaving index, and due to Bland's rule, it means that $\forall i \in L \wedge i < t$, index i doesn't leave the basis, therefore $u_{f(i)} \leq 0$ (the i'th coordinate of the entering column), else Bland's rule would have chosen i over t.

Define $[d]_i = \mathbb{1}_{i=s} - u_{f(i)} \cdot \mathbb{1}_{i \in L}$. By d's definition, $\forall i \neq t$, $[d]_i \geq 0$ and $[d]_t < 0$.

Notice that

$$A \cdot d = A_{B(L)} \cdot d_{B(L)} + A_s = A_{B(L)} \cdot -u + A_s = A_{B(L)} \cdot -A_{B(L)}^{-1} \cdot A_s + A_s = -A_s + A_s = 0$$

Notice that $\forall i < t, [c_E]_i \geq 0, [d]_i \geq 0$ and $[d]_t, [c_E]_t < 0$. Therefore, $c_E^T \cdot d \geq 0$. Notice that

$$\begin{aligned} 0 \leq c_E^T \cdot d &= \left(c^T - c_{B(E)}^T \cdot A_{B(E)}^{-1} \cdot A\right) \cdot d = c^T \cdot d - c_{B(E)}^T \cdot A_{B(E)}^{-1} \cdot A \cdot d = c^T \cdot d - c_{B(E)}^T \cdot A_{B(E)}^{-1} \cdot 0 \\ &= c^T \cdot d = c_{B(L)}^T \cdot d_{B(L)} + c_s = c_{B(L)}^T \cdot -u + c_s = c_s - c_{B(L)}^T \cdot u \overset{\star}{<} 0 \end{aligned}$$

So we got that $0 < 0$ and it is a contradiction and therefore there can't be a cycle in the algorithm and therefore it always stops, as required.

2. For convenience we solve the question for the minimization problem, and then plug $-c$ to get the required result. First, denote via x_{opt} the optimal solution to the LP problem and v_{opt} the value of the optimal solution.

   We use the Wikipedia algorithm for the Ellipsoid Method and add that if the center of the ellipsoid is in P we add the constraint $c^T \cdot x \leq c^T \cdot x_{center}$ to P. And we return the last feasible solution found by the method.

   Let $\varepsilon > 0$. Suppose $c \neq 0$, otherwise the convergence is trivial.

   Define $\boxed{t = 2n(n+1) \cdot \left\lceil \ln\left(\frac{2R^2 \cdot \|c\|_2}{r \cdot \varepsilon}\right)\right\rceil, \alpha = \frac{\varepsilon}{2R\|c\|_2}}$

   We saw in class that if we run he ellipsoid algorithm with (P, R, r) then algorithm will reach the volume of $B(0, r)$ after $2n(n+1) \cdot \ln\left(\frac{R}{r}\right)$ iterations, therefore we need at most $2n(n+1) \cdot \left\lceil \ln\left(\frac{R}{r}\right)\right\rceil$ iterations.

   Now we run the algorithm with $(R = R, P = P, r = r \cdot \alpha)$, therefore we need at most

   $$2n(n+1) \cdot \left\lceil \ln\left(\frac{R}{r \cdot \alpha}\right)\right\rceil = 2n(n+1) \cdot \left\lceil \ln\left(\frac{R}{\frac{r \cdot \varepsilon}{2 \cdot R \cdot \|c\|_2}}\right)\right\rceil = 2n(n+1) \cdot \left\lceil \ln\left(\frac{2R^2 \cdot \|c\|_2}{r \cdot \varepsilon}\right)\right\rceil = t$$

   iterations.

   Now, we define $P_* = (1-\alpha) \cdot x_{opt} + \alpha \cdot P$. Notice that $Vol(P_*) = \alpha^n \cdot Vol(P) \geq \alpha^n \cdot Vol(B(0, r))$.

   Denote the ellipsoids in the iterations with $E_0, \ldots, E_t$.

   Notice that $Vol(E_t) < Vol(0, r\alpha) = Vol(B(0, r)) \cdot \alpha^n \leq Vol(P_*)$. Therefore $P_* \not\subseteq E_t$. Therefore, $\exists y \in P_* \setminus E_t$. Therefore, $\exists z \in P$ s.t. $y = (1-\alpha) \cdot x_{opt} + \alpha z$. Due to the convexity of P, it follows that $y \in P$. Therefore $y \in P \subseteq E_0 \wedge y \notin E_t$, therefore $0 \leq \exists i \leq t$ s.t. $y \in E_{i-1} \wedge y \notin E_i$. Denote with g_i the cutting hyperplane in the i'th iteration and x_i the i'th center, therefore due to how the algorithm works, $g_i^T \cdot y > g_i^T \cdot x_i$ (as stated in the algorithm's definition in the Wikipedia page on Ellipsoid Method).

   If $x_i \notin P$ we can oracle can choose a cutting hyper-plane that doesn't separate P. Therefore, in the case y doesn't remain in the next ellipsoid, which means that P was separated, we know that $x_i \in P$.

   Denote f to be the cost function. If $x_i \in P$, then notice that $c^T \cdot x_{opt} \leq c^T \cdot x_i$ and then the oracle can choose $c = g_i = \nabla f$.

   Denote $S = E_{i-1} \cap \left\{x \mid g_i^T \cdot x \leq g_i^T \cdot x_i\right\}$ therefore, $\forall x \in E_{i-1} \setminus S$, from convexity $f(x) \geq f(x_i) + \nabla f(x_i) \cdot (x - x_i)$, notice that $g_i(x_i) = \nabla f(x_i)^T$ and therefore $f(x) \geq f(x_i) + \nabla f(x_i)^T \cdot (x - x_i) = f(x_i) + g_i^T \cdot (x - x_i) \geq f(x_i)$. Notice that $y \in E_{i-1} \setminus S$ and therefore $f(y) > f(x_i)$.

Therefore, after this step we add the inequality that $f(x) \leq f(x_i)$ and therefore the solution that we will find will satisfy $f(x) \leq f(x_i)$.

Denote the solution of the algorithm with x_E. Notice that

$$
\begin{aligned}
f(x_E) &\leq f(x_i) \leq f(y) = f((1-\alpha)\cdot x_{opt} + \alpha z) = c^T \cdot (1-\alpha) x_{opt} + c^T \cdot \alpha z \\
&= c^T x_{opt} + \alpha \cdot c^T (z - x_{opt}) = v_{opt} + \alpha \cdot \langle c, z - x_{opt} \rangle \\
&\overset{\text{Cauchy–Schwarz–inequality}}{\leq} v_{opt} + \alpha \cdot \|c\|_2 \cdot \|z - x_{opt}\|_2 \leq v_{opt} + \alpha \cdot \|c\|_2 \cdot 2R \leq v_{opt} + \varepsilon
\end{aligned}
$$

Therefore, we showed that after t iterations that we have a feasible solution that is ε from optimal, as required.

3. For every $i \geq 1$ define $f_i := f(x_i) := f(x_{i-1}) - \frac{1}{L}\nabla f(x_{i-1})$. Similarly to the lecture, by using the Taylor expansion of f at x_{i-1} with the intermediate value theorem, we get for some $z \in [x_{i-1}, x_i]$[1]:

$$
\begin{aligned}
f_i &= f_{i-1} + \nabla f_{i-1}(x_i - x_{i-1}) + \frac{1}{2}\nabla^2 f_{i-1}(x_i - x_{i-1}, x_i - x_{i-1}) \\
&= f_{i-1} - \frac{1}{L}||\nabla f_{i-1}||^2 + \frac{1}{2L^2}\nabla^2 f_{i-1}(\nabla f_{i-1}, \nabla f_{i-1})
\end{aligned}
$$

∇f is L-Lipschitz, therefore from the lecture

$$
\leq f_{i-1} - \frac{1}{L}||\nabla f_{i-1}||^2 + \frac{1}{2L^2}L||\nabla f_{i-1}||^2 = f_{i-1} - \frac{1}{2L}||\nabla f_{i-1}||^2
$$

Now, consider $f_i - f_*$:

$$
\begin{aligned}
f_i - f_* = f_i - f_{i-1} + f_{i-1} - f_* &\leq f_{i-1} - f_* - \frac{1}{2L}||\nabla f_{i-1}||^2 \\
&\overset{\text{convexity}}{\leq} \nabla f_{i-1}(x_{i-1} - x_*) - \frac{1}{2L}||\nabla f_{i-1}||^2 \\
&= \pm\frac{L}{2}||x_{i-1} - x_*||^2 + \frac{L}{2}\cdot\left(2\frac{1}{L}\nabla f_{i-1}(x_{i-1} - x_*) - \frac{1}{L^2}||\nabla f_{i-1}||^2\right) \\
&= \frac{L}{2}(||x_{i-1} - x_*||^2 - ||x_{i-1} - \nabla f_{i-1} - x_*||^2) \\
&= \frac{L}{2}(||x_{i-1} - x_*||^2 - ||x_i - x_*||^2)
\end{aligned}
$$

Hence, since the updating rule decreases f in every iteration,

$$
\begin{aligned}
i \cdot (f_i - f_*) &\leq \sum_{t=0}^{i}(f_t - f_*) \leq \sum_{t=0}^{i}\frac{L}{2}(||x_{t-1} - x_*||^2 - ||x_t - x_*||^2) = \frac{L}{2}(||x_0 - x_*||^2 - ||x_i - x_*||^2) \\
&\leq \frac{L}{2}||x_0 - x_*||^2
\end{aligned}
$$

Therefore, $f_i - f_* \leq \frac{L}{2i}||x_0 - x_*||^2$, as required.

4. (a) In order to show that f is not self-concordant on $[0, \infty)$, we show the opposite inequality, namely $|f'''(x)| > 2|f''(x)|^{3/2}$ for any $a > 0$ and some matching $x > 0$.

$$
\begin{aligned}
|f'''(x)| = a(a+1)(a+2)x^{-a-3} &> 2(a(a+1)x^{-a-2})^{3/2} = 2|f''(x)|^{3/2} \\
\iff a + 2 &> (a(a+1)x^{-a})^{1/2}
\end{aligned}
$$

Lhs is constant and positive, and rhs $\to 0$ as $x \to \infty$. In particular there is $x > 0$ such that rhs $<$ lhs. This proves that f is not self-concordant for every $a > 0$.

[1] We assume that f is twice-differentiable. Anyway, there is a theorem that states that in this case it is twice-differentiable almost everywhere since its gradient is L-Lipschitz, and we think that the proof should hold here as well.

(b) By definition, $cone(K) = \{\sum_{i \leq N} \alpha_i k_i \mid \alpha_i \geq 0,\ k_i \in K,\ N \in \mathbb{N}\}$, and $K \subseteq cone(K)$. For such α_i, k_i, forall $x \in \mathbb{R}^n$, by their definitions, $\alpha_i \geq 0$ and $x^T k_i x \geq 0$ therefore $x^T \left(\sum_{i \leq N} \alpha_i k_i\right) x = \sum_{i \leq N} \alpha_i x^T k_i x \geq 0$, so $\left(\sum_{i \leq N} \alpha_i k_i\right) \succcurlyeq 0$. Therefore, $cone(K) \subseteq K$ and it follows that $cone(K) = K$. Hence K is a cone.

Did you know? There are 24 steps to draw the official Nepal flag, and they can be found here: https://www.mohp.gov.np/downloads/Constitution%20of%20Nepal%202072_full_english.pdf (page 221).