CONVEX AND CONIC OPTIMIZATION

Homework 1

NILAVA METYA
nilava.metya@rutgers.edu
nm8188@princeton.edu

February 27, 2024

Problem 1

Let $S \subseteq \mathbb{R}^n$.

1. The convex hull of S is the intersection of all convex sets that contain S.
2. If S is closed, then the convex hull of S is closed.
3. If S is bounded, then the convex hull of S is bounded.
4. If S is compact, then the convex hull of S is compact.
5. The sum of two quasiconvex functions is quasiconvex.
6. A quadratic function $f(x) = x^\top Qx + b^\top x + c$ is convex if and only if it is quasiconvex.
7. Any closed convex set $\Omega \subseteq \mathbb{R}^n$ can be written as $\Omega = \{x \in \mathbb{R}^n \mid g(x) \leq 0\}$ for some convex function $g : \mathbb{R}^n \to \mathbb{R}$.
8. If $f : \mathbb{R}^n \to \mathbb{R}$ is convex on a convex set $S \subseteq \mathbb{R}^n$, then f is continuous on S.
9. Suppose $P \in \mathbb{R}^{n\times n}$ is a matrix with nonnegative entries whose columns each sum up to one. Then, there exists $x \in \mathbb{R}^n$ such that $Px = x, x \geq 0$,and $\sum_i x_i = 1$.
10. A continuous function $f : \mathbb{R}^n \to \mathbb{R}$ satisfying the midpoint convexity property

$$f\left(\frac{x+y}{2}\right) \leq \frac{f(x)+f(y)}{2} \forall x, y \in \mathbb{R}^n$$

is convex.

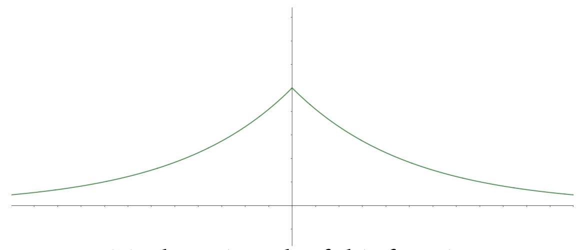
Solution

1. **True**.

   Let $\mathcal{C} = \text{conv}(S)$ as defined in class (collection of convex combinations) and let $\mathcal{X}$ be the intersection of all convex sets that contain S.
   It is clear that $S \subseteq \mathcal{C}$ whence $\boxed{\mathcal{X} \subseteq \mathcal{C}}$ by the description of $\mathcal{X}$.
   Now, if T is a convex set containing S, then any convex combination x of points in S is a convex combination of points in T, so $x \in T$. This shows that $\mathcal{C} \subseteq T$. Again by description of $\mathcal{X}$, we have $\boxed{\mathcal{C} \subseteq \mathcal{X}}$.

2. **False**.

Consider $S = \left\{(x,y)\in\mathbb{R}^2 \;\middle|\; y \geq e^{-|x|}\right\}$. We'll show that $\mathcal{C} := \operatorname{conv}(S)$ is $\mathcal{X} := \left\{(x,y)\in\mathbb{R}^2 \;\middle|\; y > 0\right\}$. Clearly $S \subseteq \mathcal{X}$ because exponential takes only positive values, so $\boxed{\mathcal{C} = \operatorname{conv}(S) \subseteq \operatorname{conv}(\mathcal{X}) = \mathcal{X}}$, where the last equality is true because $\mathcal{X}$ is convex (if two points have positive $y-$coordinate, so do their convex combinations).
On the other hand, say $(a,b)\in\mathcal{X}$, pick a large enough θ such that $\min(|a+\theta|, |a-\theta|) > \ln\frac{1}{b}$. This ensures that $e^{-|a\pm\theta|} < b$ so that $(a\pm\theta, b)\in S$ and thus their average $(a,b)\in\mathcal{C}$. This proves that $\boxed{\mathcal{X}\subseteq\mathcal{C}}$.
So $\mathcal{X} = \operatorname{conv}(S)$ where S is closed but $\mathcal{X}$ (the strict upper half plane) is not.



S is the epigraph of this function

3. **True**.

   S is bounded, so $\exists\delta > 0$ such that $S \subseteq B(0,\delta)$. Recall that $B(0,\delta)$ is convex. This implies that $\operatorname{conv}(S) \subseteq \operatorname{conv}(B(0,\delta)) = B(0,\delta)$, whence $\operatorname{conv}(S)$ is bounded.

4. **True**.

   We'll use the fact that a set in Euclidean space is compact iff it is sequentially compact. We say that a set T is sequentially compact if every sequence of points in T has a convergent subsequence (with limit in T). $\boxed{\text{We will use a } \bullet \text{ in subscript to suppress the lower index of a sequence.}}$
   Assume S is compact (hence sequentially compact). It is also important to note that $[0,1]$ is (sequentially) compact. Consider a sequence of points $\{a_k\}_{k\in\mathbb{N}}$ in $\operatorname{conv}(S)$. By Caratheodory's theorem, for each a_k, there are $n+1$ non-negative reals $\lambda_1^{(k)}, \cdots, \lambda_{n+1}^{(k)} \in [0,1]$ adding to 1 and $n+1$ points $x_1^{(k)}, \cdots, x_1^{(k)} \in S$ such that $a_k = \displaystyle\sum_{i=1}^{n} \lambda_i^{(k)} x_i^{(k)}$ (I want to emphasize again that this is for each a_k).
   Consider the sequence $\left\{\lambda_k^{(1)}\right\}_{k\in\mathbb{N}}$. This is a sequence in the compact unit interval and thus has a convergent subsequence, say $\left\{\lambda_{d_k^{(1)}}^{(1)}\right\}_{k\in\mathbb{N}}$. Let the limit of this subsequence be $\lambda^{(1)}$. Next note that $\left\{x_{d_k^{(1)}}^{(1)}\right\}_{k\in\mathbb{N}}$ is a sequence in the compact set S and thus has a convergent subsequence, say $\left\{x_{\tilde{d}_k^{(1)}}^{(1)}\right\}_{k\in\mathbb{N}}$. So $\left\{\tilde{d}_k^{(1)}\right\}_{k\in\mathbb{N}}$ is a subsequence of $\left\{d_k^{(1)}\right\}_{k\in\mathbb{N}}$, and thus does not affect the convergence of $\lambda_{\tilde{d}_k^{(1)}}^{(1)}$. Let the limit of $\left\{x_{\tilde{d}_k^{(1)}}^{(1)}\right\}_{k\in\mathbb{N}}$ be $x^{(1)}\in S$. Notice that $\lim\limits_{k\to\infty} \lambda_{\tilde{d}_k^{(1)}}^{(1)} x_{\tilde{d}_k^{(1)}}^{(1)} = \lambda^{(1)}x^{(1)}$, which happens because

$$\begin{aligned}\left\|\lambda_{\tilde{d}_k^{(1)}}^{(1)} x_{\tilde{d}_k^{(1)}}^{(1)} - \lambda^{(1)}x^{(1)}\right\| &\leq \left\|\lambda_{\tilde{d}_k^{(1)}}^{(1)} x_{\tilde{d}_k^{(1)}}^{(1)} - \lambda_{\tilde{d}_k^{(1)}}^{(1)} x^{(1)}\right\| + \left\|\lambda_{\tilde{d}_k^{(1)}}^{(1)} x^{(1)} - \lambda^{(1)}x^{(1)}\right\| \\ &= \lambda_{\tilde{d}_k^{(1)}}^{(1)} \underbrace{\left\|x_{\tilde{d}_k^{(1)}}^{(1)} - x^{(1)}\right\|}_{\text{can be made arbitrarily small}} + \underbrace{\left|\lambda_{\tilde{d}_k^{(1)}}^{(1)} - \lambda^{(1)}\right|}_{\text{can be made arbitrarily small}} \left\|x^{(1)}\right\|.\end{aligned}$$

   By following a similar procedure, one can further extract a subsequence $\left\{d_k^{(2)}\right\}_{k\in\mathbb{N}}$, which makes $\lambda_\bullet^{(2)}$ converge to $\lambda^{(2)}$, and a further sub-subquence $\left\{\tilde{d}_k^{(2)}\right\}_{k\in\mathbb{N}}$ which makes $x_\bullet^{(2)}$ converge to $x^{(2)}$. Note that this smaller sequence does not affect the convergence of $\lambda_\bullet^{(1)}$ and $x_\bullet^{(1)}$ (and also their product). By the same argument as above, $\lim\limits_{k\to\infty} \lambda_{\tilde{d}_k^{(2)}}^{(2)} x_{\tilde{d}_k^{(2)}}^{(2)} = \lambda^{(2)}x^{(2)}$.

We do this (extracting subsequences in turn for convergence of subsequences of $\lambda_\bullet^{(1)}, x_\bullet^{(1)}, \lambda_\bullet^{(2)}, x_\bullet^{(2)}, \lambda_\bullet^{(3)}, \cdots$) for a total of $n+1$ times, and finally get a subsequence $\left\{\tilde{d}_k^{(n+1)}\right\}_{k\in\mathbb{N}}$ which ensures convergence of $\lambda^{(i)}_{\tilde{d}_k^{(n+1)}} \xrightarrow{k\to\infty} \lambda^{(i)}$ and $x^{(i)}_{\tilde{d}_k^{(n+1)}} \xrightarrow{k\to\infty} x^{(i)}$ for every $i = 1, \cdots, n+1$. It thus stands that $\lambda^{(i)} \in [0,1]$, $x^{(i)} \in S \forall i$ and $\sum_{i=1}^{n+1} \lambda^{(i)} = \sum_{i=1}^{n+1} \lim_{k\to\infty} \lambda^{(i)}_{\tilde{d}_k^{(n+1)}} = \lim_{k\to\infty} \sum_{i=1}^{n+1} \lambda^{(i)}_{\tilde{d}_k^{(n+1)}} = \lim_{k\to\infty} 1 = 1$ where the sum and limit could be exchanged because each limit exists. Thus $\left\{a_{\tilde{d}_k^{(n+1)}}\right\}_{k\in\mathbb{N}}$ is a subsequence of $\{a_k\}_{k\in\mathbb{N}}$ which converges to $a := \sum_{i=1}^{n+1} \lambda^{(i)} x^{(i)}$. $a \in \mathrm{conv}(S)$ because each $x^{(i)} \in S$ by construction and the $\lambda^{(i)}$'s form a convex weight for the $x^{(i)}$'s.

5. **False**.

   Consider $f(x) = x^3, g(x) = x^2$. Then f is quasiconvex with sublevel sets $S_\alpha(f) = \left(-\infty, \alpha^{\frac{1}{3}}\right]$ and g is quasiconvex with sublevel sets $S_\alpha(g) = \begin{cases} [-\sqrt{\alpha}, \sqrt{\alpha}] & \text{if } \alpha \geq 0 \\ \varnothing & \text{otherwise} \end{cases}$. But $(f+g)(x) = x^3 + x^2$ is not quasiconvex. Indeed, $f + g \leq 0 \iff x^2(x+1) \leq 0 \iff x = 0$ or $x \leq -1$, so $S := S_0(f+g) = (-\infty, -1] \cup \{0\}$ which is not convex because $\frac{-1+0}{2} = \frac{-1}{2} \notin S$ even though $0, -1 \in S$.

6. **True**.

   We already know convexity implies quasiconvexity. We'll prove the (contrapositive of) converse for quadratic polynomals (that is, not convex implies not quasiconvex). Note here that convexity is same as midpoint convexity by problem 1(10) because quadratic polynomials are continuous. Problem 1(10) can be used because this probem has not been used there and thus there is no circular argument.
   $f(x) = x^\top Q x + b^\top x + c$, where Q is symmetric (WLOG). Assume that f is not convex. So $\exists x, y \in \mathbb{R}^n$ such that

$$
\begin{aligned}
& f\left(\frac{x+y}{2}\right) > \frac{f(x)+f(y)}{2} \\
\implies & \frac{1}{2}(x+y)^\top Q(x+y) + b^\top(x+y) > x^\top Qx + y^\top Qy + b^\top(x+y) \\
\implies & x^\top Qx + y^\top Qy + 2x^\top Qy > 2\left(x^\top Qx + y^\top Qy\right) \\
\implies & x^\top Qx + y^\top Qy + 2x^\top Qy < 0 \\
\implies & (x-y)^\top Q(x-y) < 0
\end{aligned}
$$

   Let $z = \begin{cases} x - y & \text{if } b^\top(x-y) \geq 0 \\ y - x & \text{otherwise} \end{cases}$. So this ensures that $B := b^\top z \geq 0$ and $A := z^\top Q z < 0$. Note that $B^2 - 4A = B^2 + 4|A| > 0$ so that $\lambda_\pm := \dfrac{-B \pm \sqrt{B^2 - 4A}}{2A}$ are well defined real numbers. These are precisely the roots of the polynomial $At^2 + Bt + 1 = 0$. We thus note that $f(\lambda_\pm z) = \lambda_\pm^2 A + \lambda_\pm B + c = c - 1$. Consider the sublevel set $S := S_{c-1} = \{q \in \mathbb{R}^n \mid f(q) \leq c - 1\}$. We've shown that $p_1 := \lambda_+ z, p_2 := \lambda_- z \in$

S because $f(p_1) = f(p_2) = c - 1$. But $\dfrac{p_1 + p_2}{2} = \dfrac{\lambda_+ z + \lambda_- z}{2} = \dfrac{-B}{2A} z$ whence

$$\begin{aligned}
f\left(\frac{p_1 + p_2}{2}\right) &= f\left(\frac{-B}{2A} z\right) \\
&= \frac{B^2}{4A^2} A - \frac{B}{2A} B + c \\
&= \frac{B}{4(-A)} + c \geq c \\
&> c - 1 \\
\implies \frac{p_1 + p_2}{2} &\notin S.
\end{aligned}$$

This proves that S is not convex. So f is not quasiconvex.

7. **True**.

   Ω is closed and convex. For each $x \in \mathbb{R}^n$ there is a point $p^* \in \Omega$ such that $\inf_{p \in \Omega} \|x - p\| = \|x - p^*\|$. This is because the optimal value of $\inf_{p \in \Omega} \|x - p\|$, for a fixed x, is same as that obtained by optimizing over $\Omega \cap B(x, R)$ for some large enough radius R such that $B(x, R) \cap \Omega$ is nontrivial (this intersection is closed and bounded and thus there is an optimal solution).
   Consider the function $g : \mathbb{R}^n \to \mathbb{R}$ given by $g(x) = \inf_{p \in \Omega} \|x - p\|$. Then g is convex. Indeed, if $u, v \in \mathbb{R}^n$ and $\lambda \in [0, 1]$ then there are points $a, b \in \Omega$ such that $g(u) = \|u - a\|, g(v) = \|v - b\|$, which means that

$$\begin{aligned}
\lambda g(u) + (1 - \lambda) g(v) &= \lambda \|u - a\| + (1 - \lambda) \|v - b\| \\
&\geq \left\| \lambda u + (1 - \lambda) v - \underbrace{[\lambda a + (1 - \lambda) b]}_{\in \Omega} \right\| \\
&\geq \inf_{p \in \Omega} \|\lambda u + (1 - \lambda) v - p\| = g(\lambda u + (1 - \lambda) v).
\end{aligned}$$

   We will show that $\Omega = \{x \in \mathbb{R}^n \mid g(x) \leq 0\}$. By definition, $g(x) = 0 \forall x \in \Omega$, so $\Omega \subseteq \{x \in \mathbb{R}^n \mid g(x) \leq 0\}$. For the other inclusion, observe that if $g(x) \leq 0$ then $g(x) = 0$ because g is the infimum over norms, hence non-negative, so that $\|x - p^*\| = 0$ (where $p^* \in \Omega$ is the optimal solution for the abovementioned optimization problem). This means $x = p^* \in \Omega$. So the other inclusion is also true.

8. **False**.

   The counterexample is obtained by taking $S = [0, 1] \subseteq \mathbb{R}$ and $f(x) = \mathbf{1}_{x=1} = \begin{cases} 0 & \text{if } 0 \leq x < 1 \\ 1 & \text{otherwise} \end{cases}$. In each dimension, there is a counterexample, namely $S = [0, 1]^n \subseteq \mathbb{R}^n$ and $f = \mathbf{1}_{x=(1,\cdots,1)}$. S is clearly convex. The function is convex because it is constant 0 except for one point and if $p = (1, \cdots, 1)$ and $q \in S \setminus \{p\}$ then for any $\lambda \in (0, 1]$ we have $\lambda p + (1 - \lambda) q \neq (1, \cdots, 1)$ whence $f(\lambda p + (1 - \lambda) q) = 0 \leq \lambda = \lambda f(p) + (1 - \lambda) f(q)$. But clearly f is not continuous.

9. **True**.

   First solution: (using Farkas lemma)
   Consider $A = \begin{bmatrix} & P & \\ 1 & 1 & \cdots & 1 \end{bmatrix} - \begin{bmatrix} & \mathbf{1}_{n \times n} & \\ 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{(n+1) \times n}$, where $\mathbf{1}_{n \times n}$ is the $n \times n$ identity matrix,

and $b = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} = f_{n+1} \in \mathbb{R}^{n+1}$ the $(n+1)^{\text{st}}$ basic vector in $\mathbb{R}^{n+1}$. Let $\tilde{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \\ t \end{bmatrix} \in \mathbb{R}^n$ be an arbitrary real column vector. Consider the index $k \in \{1, \cdots, n\}$ such that $y_k = \min\limits_{i \le i \le n} y_i$. Then

$$\begin{aligned}
\left(A^\top \tilde{y}\right)_k &= \sum_{i=1}^n p_{ik} y_i - y_k + t \\
&= \sum_{i=1}^n p_{ik} y_i - \sum_{i=1}^n p_{ik} y_k + t && [\because \text{column sum } = 1] \\
&= \sum_{i=1}^n p_{ik}(y_i - y_k) + t \\
&\ge t && [\because p_{ik} \ge 0, y_i - y_k \ge 0] \\
&= b^\top \tilde{y}
\end{aligned}$$

This means $\left\{\tilde{y} \ \middle|\ A^\top \tilde{y} \le 0, b^\top \tilde{y} > 0\right\} = \varnothing$ because $0 \ge A^\top \tilde{y} \ge b^\top \tilde{y} > 0$ is infeasible. By Farkas lemma, $\{x \in \mathbb{R}^n \mid Ax = b, x \ge 0\} \ne \varnothing$. This simply means that there is a vector $x \in \mathbb{R}^n$ such that $(P - \mathbf{1}_{n\times n})x = 0$ (using the first n rows of A) with the properties that $\sum\limits_i x_i = 1$ (using last row of A) and $x \ge 0$

Second solution:

We know that $P^\top \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$ because each column of P sums to 1. Recall that row rank is same as column rank, whence $\operatorname{rank}(P - \mathbf{1}_{n\times n}) = \operatorname{rank}\left((P - \mathbf{1}_{n\times n})^\top\right) = \operatorname{rank}(P^\top - \mathbf{1}_{n\times n}) < n$. This means $\ker(P - \mathbf{1}_{n\times n}) \ne \{0\}$ and so there is some $v \in \mathbb{R}^n \setminus \{0\}$ such that $Pv = v$. If each $v_i \ge 0$, our required vector is $\frac{v}{\|v\|_1}$. If each $v_i \le 0$ we can take $x = \frac{-v}{\|v\|_1}$ to ensure that $x_i \ge 0 \forall i$ and $\sum x_i = 1$.
Assume otherwise, that is, some indices are positive, some are negative. Then consider $v^+ = (v_1^+, \cdots, v_n^+)$ where $v_i^+ = \max(0, v_i) \ge 0$. This means $(Av^+)_k - (v)_k = (Av^+ - v)_k = (A(v^+ - v))_k \ge 0$ because $v^+ - v \ge 0$ and A has nonnegative entries, so $Av^+ \ge v$. But $Av^+ \ge 0$ because A, v^+ all have non-negative entries. This proves $Av^+ - v^+ \ge 0$. But $\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}^\top Av^+ = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}^\top v^+$ by associativity, whence $\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}^\top (Av^+ - v^+) = 0$ where the LHS is the sum of entries of the non-negative vector $Av^+ - v^+$. This can only happen when each entry of $Av^+ - v^+$ is 0. This shows $Av^+ = v^+$. So we are done by taking $x = \frac{v^+}{\|v^+\|_1}$ because v^+ is a nonzero vector because of our assumptions.

10. **True**.

    Say f is midpoint convex (that is, satisfies the given condition).
    Consider epi$(f) = \{(x,t) \in \mathbb{R}^n \times \mathbb{R} \mid f(x) \le t\}$. Consider a sequence $\{(x_n, t_n)\}_{n\in\mathbb{N}}$ of points in epi(f) that converge to $(a, s) \in \mathbb{R}^n \times \mathbb{R}$. This implies $\lim\limits_{n\to\infty} x_n = a$, $\lim\limits_{n\to\infty} t_n = s$. By choice of points, $t_n \ge f(x_n)$ whence $\lim\limits_{n\to\infty} t_n \ge \lim\limits_{n\to\infty} f(x_n) \overset{f \text{ continuous}}{=} f(a)$. By definition, $(a,s) \in$ epi(f). Thus, epi(f) is closed.
    Say $(x,t), (y,u) \in$ epi(f). So $u \ge f(y), t \ge f(x)$. Then by midpoint convexity of f, we have $f\left(\frac{x+y}{2}\right) \le \frac{f(x)+f(y)}{2} \le \frac{u+t}{2}$ and, by definition of epi(f), this implies $\left(\frac{x+y}{2}, \frac{u+t}{2}\right) \in$ epi(f). So epi(f) is mid-point convex.
    Since epi(f) is closed and midpoint convex, epi(f) is convex. This implies f is convex.

Problem 2

Solution

Explanation:

First note that $x(N) = \sum_{i=0}^{N-1} u(i) A^{N-1-i} b = \underbrace{\begin{bmatrix} A^{N-1}b & A^{N-2}b & \cdots & Ab & b \end{bmatrix}}_{M} \begin{bmatrix} u(0) \\ \vdots \\ u(N-1) \end{bmatrix}$. Thus one constraint must be $\boxed{Mu = x_{\text{des}}}$, which is linear.

We want to solve

$$\begin{aligned} &\min_u \sum_{i=0}^{N-1} f(u(i)) \\ &\text{s.t. } Mu = x_{\text{des}} \end{aligned}$$

Next note that $f(a) = \max(|a|, 2|a| - 1)$. There are two issues to address while forming affine constraint(s): one is the absolute value, another is the $\max$ function. The first one is addressed by considering a variable $v(i) \geq 0$ for each $i \in \{0, 1, \cdots, N-1\}$, such that $\boxed{u(i) \leq v(i)}$ and $\boxed{-u(i) \leq v(i)}$ and taking the tightest such $v(i)$, so the objective function might be written as a minimization of these $\sum_{i=0}^{N-1} f(v(i)) = \sum_{i=0}^{N-1} \max(v(i), 2v(i) - 1)$. But now the objective isn't affine. We can further introduce a variable $w(i)$ for each i that satisfies $\boxed{w(i) \geq v(i)}$ and $\boxed{w(i) \geq 2v(i) - 1}$, which ensures that $w(i) \geq f(v(i))$ and minimizing $w(i)$ ensures that the optimal value is indeed $f(v(i))$.

The LP formulation:

So we propose the following LP:

$$\begin{aligned} &\min_{u,v,w} \sum_{i=0}^{N-1} w(i) \\ &\text{s.t. } Mu = x_{\text{des}} \\ &\quad -v + u \leq 0 \\ &\quad -v - u \leq 0 \\ &\quad -w + v \leq 0 \\ &\quad 2v - \boldsymbol{c} - w \leq 0 \end{aligned}$$

where $\boldsymbol{c}$ is the column vector of length N containing all 1's.

See code next page onwards

Fuel Optimization

First we import the necessary packages: `cvxpy` for solving optimization problems, `numpy` for linear algebra and `matplotlib` for using the `stairs()` function to graph the signal.

[1]:
```
import cvxpy as cp
import numpy as np
import matplotlib.pyplot as plt
```

Next we enter the data required to form the optimization problem, namely A, b, x_{des}, N.

[2]:
```
A = np.matrix([[-1, 0.4, 0.8],[1, 0, 0],[0, 1, 0]])
b = [1, 0, 0.3]
x_des = [7, 2, -6]
N = 30
```

Solving the original problem

The original optimization problem is

$$\begin{aligned} \min_u \sum_{i=0}^{N-1} f(u(i)) \\ \text{s.t. } Mu = x_{\text{des}} \end{aligned}$$

where

$$M = \begin{bmatrix} A^{N-1}b & A^{N-2}b & \cdots & Ab & b \end{bmatrix}.$$

In the next chunk of code, we compute M by iteratively acting A on b and pushing them as columns of M. We also initialize a list `c = [1,1,...,1]` which essentially does the job of an $N \times 1$ column vector comprising all 1's.

[3]:
```
M = []
temp = np.matrix([[1,0,0],[0,1,0],[0,0,1]])
for i in range(N):
    M.insert(0, (temp @ b).tolist()[0])
    temp = A * temp
M = np.matrix(M).T
```

```
c = [1] * N
```

Now we want to focus on the function $f(a) = \begin{cases} |a| & \text{if } |a| \leq 1 \\ 2|a| - 1 & \text{otherwise} \end{cases}$. It is easy to see that this expression is exactly $\max(|a|, 2|a| - 1)$. So each summand in the abovementioned objective is precisely $\max(|u(i)|, 2|u(i)| - 1)$. This is the expression `f` (so `f[i] = max(abs(u[i]), abs(2u[i])-1)`) in the following block of code. The objective `obj` is simply the sum of `f[i]`'s and the only constraint `cons` is $Mu = x_{\text{des}}$.

[4]:
```
u = cp.Variable(N, 'u')
f = cp.maximum(cp.abs(u), cp.abs(u+u)-c)
obj = cp.sum(f)
cons = [M @ u == x_des]
problem = cp.Problem(cp.Minimize(obj), cons)
problem.solve(verbose = True, solver = cp.ECOS)
```

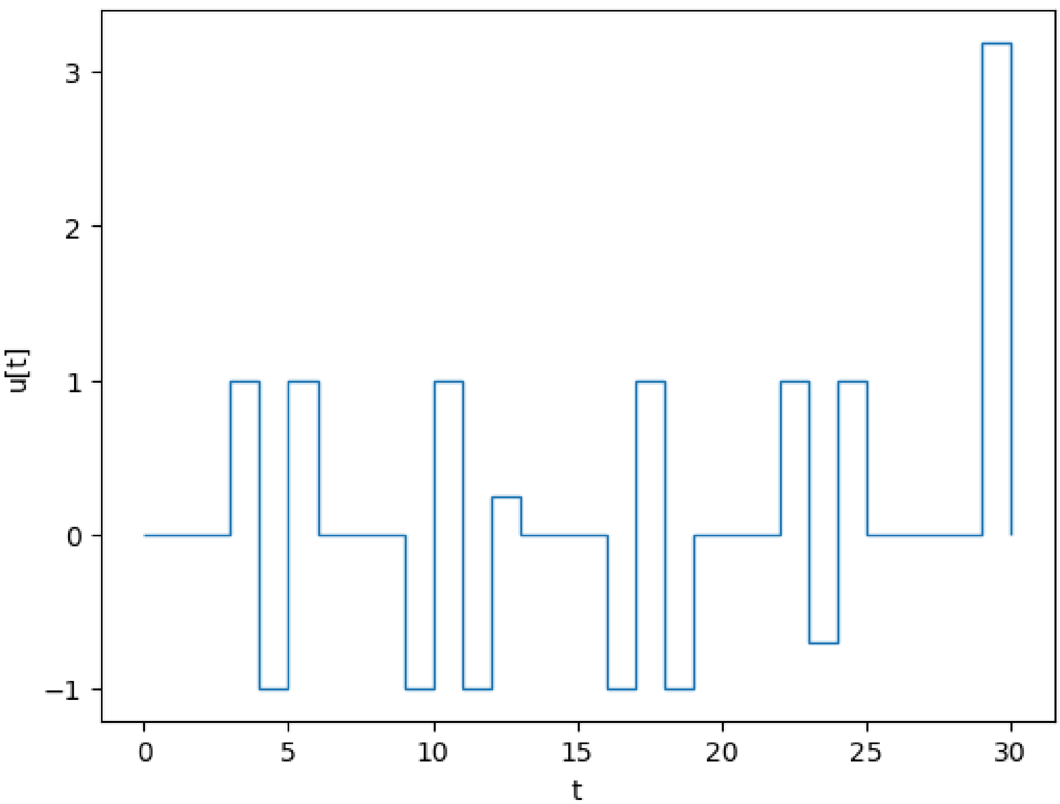
```
===============================================================================
                                     CVXPY
                                     v1.4.2
===============================================================================
(CVXPY) Feb 24 04:11:56 PM: Your problem has 30 variables, 1 constraints, and 0
parameters.
(CVXPY) Feb 24 04:11:56 PM: It is compliant with the following grammars: DCP,
DQCP
(CVXPY) Feb 24 04:11:56 PM: (If you need to solve this problem multiple times,
but with different data, consider using parameters.)
(CVXPY) Feb 24 04:11:56 PM: CVXPY will first compile your problem; then, it will
invoke a numerical solver to obtain a solution.
(CVXPY) Feb 24 04:11:56 PM: Your problem is compiled with the CPP
canonicalization backend.
-------------------------------------------------------------------------------
                                  Compilation
-------------------------------------------------------------------------------
(CVXPY) Feb 24 04:11:56 PM: Compiling problem (target solver=ECOS).
(CVXPY) Feb 24 04:11:56 PM: Reduction chain: Dcp2Cone -> CvxAttr2Constr ->
ConeMatrixStuffing -> ECOS
(CVXPY) Feb 24 04:11:56 PM: Applying reduction Dcp2Cone
(CVXPY) Feb 24 04:11:56 PM: Applying reduction CvxAttr2Constr
(CVXPY) Feb 24 04:11:56 PM: Applying reduction ConeMatrixStuffing
(CVXPY) Feb 24 04:11:56 PM: Applying reduction ECOS
(CVXPY) Feb 24 04:11:56 PM: Finished problem compilation (took 1.030e-02
seconds).
-------------------------------------------------------------------------------
                                Numerical solver
-------------------------------------------------------------------------------
```

```
(CVXPY) Feb 24 04:11:56 PM: Invoking solver ECOS  to obtain a solution.
-------------------------------------------------------------------------------
                                    Summary
-------------------------------------------------------------------------------
(CVXPY) Feb 24 04:11:56 PM: Problem status: optimal
(CVXPY) Feb 24 04:11:56 PM: Optimal value: 1.732e+01
(CVXPY) Feb 24 04:11:56 PM: Compilation took 1.030e-02 seconds
(CVXPY) Feb 24 04:11:56 PM: Solver (including time spent in interface) took
1.340e-03 seconds
```

[4]: 17.32356785630167

[5]:

```
plt.stairs(u.value, range(N+1))
plt.xlabel('t')
plt.ylabel('u[t]')
plt.show()
```



Solving the proposed LP

The above solution was based on directly solving the optimization problem, without writing it as an LP. We now solve the following LP which is mentioned in my solution:

$$\begin{aligned} \min_{q,v,w} \ & \sum_{i=0}^{N-1} w(i) \\ \text{s.t.} \ & Mq = x_{\text{des}} \\ & -v + q \leq 0 \\ & -v - q \leq 0 \\ & -w + v \leq 0 \\ & 2v - \boldsymbol{c} - w \leq 0. \end{aligned}$$

Note that the variable u has been replaced with q because we used u above.

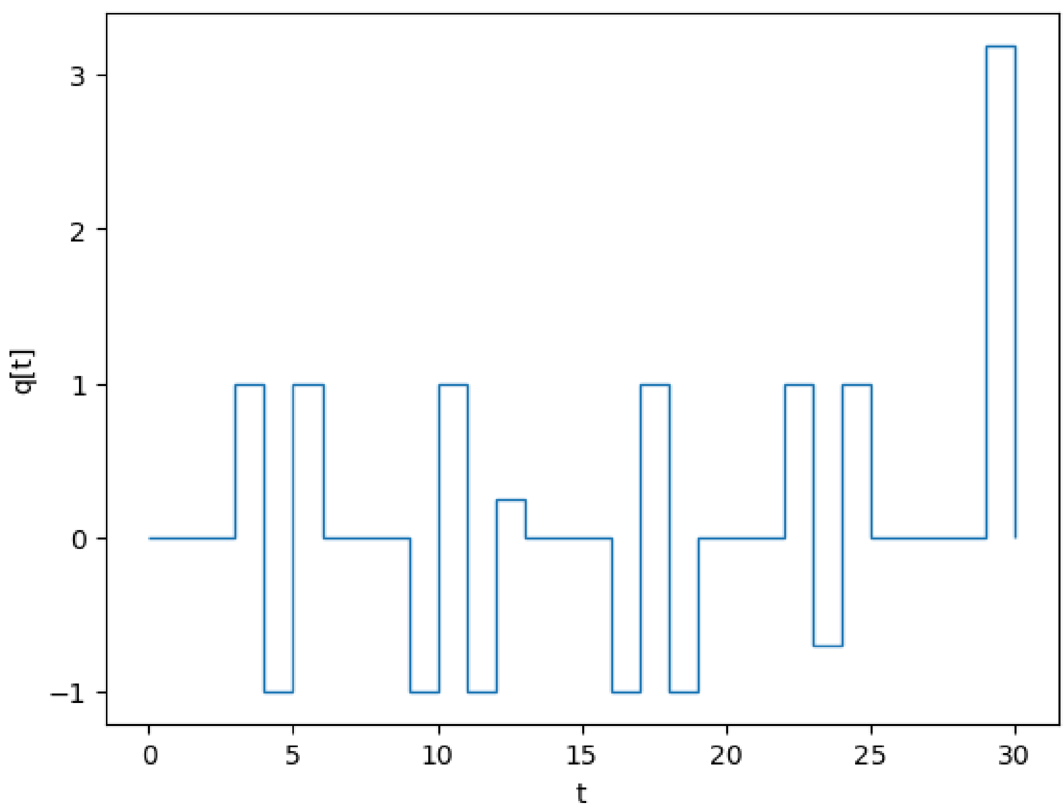
[6]:
```
q = cp.Variable(N, 'q')
v = cp.Variable(N, 'v')
w = cp.Variable(N, 'w')
ob = cp.sum(w)
con = [M @ q == x_des, q - v <= 0, q + v >= 0, v - w <= 0, v + v - c - w <= 0]
pb = cp.Problem(cp.Minimize(ob), con)
pb.solve(verbose = True, solver = cp.ECOS)
```

```
===============================================================================
                                     CVXPY
                                     v1.4.2
===============================================================================
(CVXPY) Feb 24 04:11:56 PM: Your problem has 90 variables, 5 constraints, and 0
parameters.
(CVXPY) Feb 24 04:11:56 PM: It is compliant with the following grammars: DCP,
DQCP
(CVXPY) Feb 24 04:11:56 PM: (If you need to solve this problem multiple times,
but with different data, consider using parameters.)
(CVXPY) Feb 24 04:11:56 PM: CVXPY will first compile your problem; then, it will
invoke a numerical solver to obtain a solution.
(CVXPY) Feb 24 04:11:56 PM: Your problem is compiled with the CPP
canonicalization backend.
-------------------------------------------------------------------------------
                                  Compilation
-------------------------------------------------------------------------------
(CVXPY) Feb 24 04:11:56 PM: Compiling problem (target solver=ECOS).
(CVXPY) Feb 24 04:11:56 PM: Reduction chain: Dcp2Cone -> CvxAttr2Constr ->
ConeMatrixStuffing -> ECOS
(CVXPY) Feb 24 04:11:56 PM: Applying reduction Dcp2Cone
(CVXPY) Feb 24 04:11:56 PM: Applying reduction CvxAttr2Constr
(CVXPY) Feb 24 04:11:56 PM: Applying reduction ConeMatrixStuffing
(CVXPY) Feb 24 04:11:56 PM: Applying reduction ECOS
```

```
(CVXPY) Feb 24 04:11:56 PM: Finished problem compilation (took 1.040e-02
seconds).
-------------------------------------------------------------------------------
                                Numerical solver
-------------------------------------------------------------------------------
(CVXPY) Feb 24 04:11:56 PM: Invoking solver ECOS  to obtain a solution.
-------------------------------------------------------------------------------
                                    Summary
-------------------------------------------------------------------------------
(CVXPY) Feb 24 04:11:56 PM: Problem status: optimal
(CVXPY) Feb 24 04:11:56 PM: Optimal value: 1.732e+01
(CVXPY) Feb 24 04:11:56 PM: Compilation took 1.040e-02 seconds
(CVXPY) Feb 24 04:11:56 PM: Solver (including time spent in interface) took
8.488e-04 seconds
```

[6]: 17.323567854988987

[7]:
```
plt.stairs(q.value, range(N+1))
plt.xlabel('t')
plt.ylabel('q[t]')
plt.show()
```



The LP formulation indeed gives the same solution as the original formulation. The following line determines the ℓ_1 error between the optimal solutions of the above two problems, which is $< 10^{-8}$, so the solutions are practically the same.

[8]:
```
print(sum([abs(q.value[i] - u.value[i]) for i in range(N)]))
```

```
7.965044064222433e-09
```