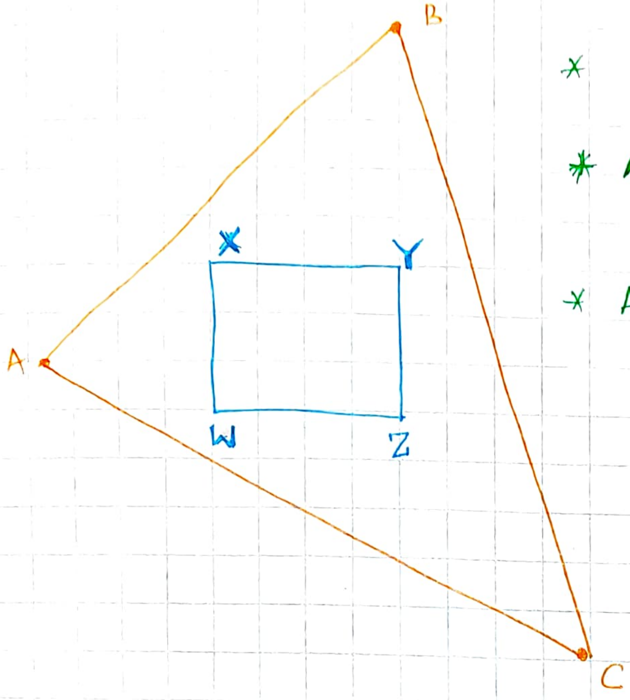
Q5 (a) Not a matroid.



* Clearly A B C are affinely independent.

* Also clearly WXYZ are affinely independent

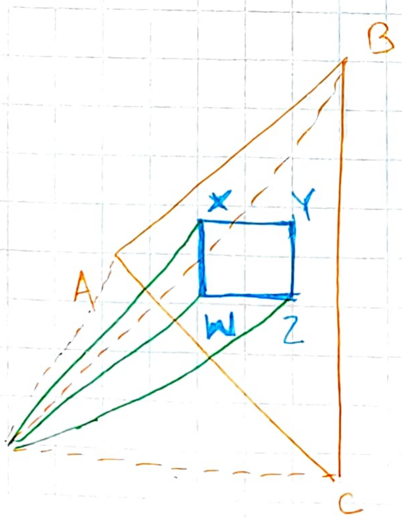
* Also clearly $\#\{W, X, Y, Z\} = 4$

$\#\{A, B, C\} = 3$

$4 > 3$

* But we can't add anything from $\{W, X, Y, Z\}$ to $\{A, B, C\}$ that still keeps $\{A, B, C,$ whatever from $WXYZ\}$ affinely independent.

(c) Not a matroid: Almost same arguement as above.



$\{A, B, C\}$ are conically independent.

$\{X, Y, Z, W\}$ are conically independent.

Clearly no element from $\{W, X, Y, Z\}$ can be added to $\{A, B, C\}$ while still keeping the new set conically independent.

(b) * Let A & B be 2 sets of points. [This is a Matroid]

* Let points in A be affinely independent.
* Let points in B be affinely independent.

* Clearly affine combination of nothing (ϕ) is ~~nothing~~ $\{0\}$ & hence independent.

* If S is affinedly independent, then any subset of S is affinely independent too.

<u>Proof</u>: Let $S = \{x_1, x_2, x_3, x_4\}$ be affinely indep.

Let $P \subseteq S$ & $P = \{x_1, x_2, x_3\}$ be affinely dep.

Then $\alpha_1 x_1 + \alpha_2 x_2 = x_3$ where $\alpha_1 + \alpha_2 = 1$

$$\Rightarrow \alpha_1 x_1 + \alpha_2 x_2 + 0 \cdot x_4 = x_3$$

$$\Rightarrow \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_4 = x_3$$

$\Rightarrow \{x_1, x_2, x_3, x_4\}$ are not affinely indep.

[Contradiction]

* <u>Fact</u> :- $N+1$ affinely indep. points span an N-dimensional affine subspace.

If $\# A = n+1$ & $\# B = m+1$ with $n > m$,

then $\text{Affine}(A)$ has $\dim = n$

$\text{Affine}(B)$ has $\dim = m$

~~Let $\dim(\text{Affine}(A) \cap \text{Affine}(B)) = k \leq \min(m, n)$~~

~~or $k \leq m$~~

~~Let C = $\text{Affine}(A) \cap \text{Affine}(B)$~~

- ~~Clearly we can choose points from A that span C.~~
~~Same goes for B i.e. we can select~~

Proof by contradiction

→ Let there be no element $a \in A$ such that $B \cup \{a\}$ is affinely independent.

⇒ All elements of A lie in the affine hull of B or Affine(B).

⇒ ~~$\dim(\text{Affine}(A)) = \dim$~~

$$\underset{\text{Hull}}{\text{Affine}}(A) = \underset{\text{Hull}}{\text{Affine}}(B)$$

⇒ $\dim(\text{Affine Hull}(A)) = \dim(\text{Affine Hull}(B))$

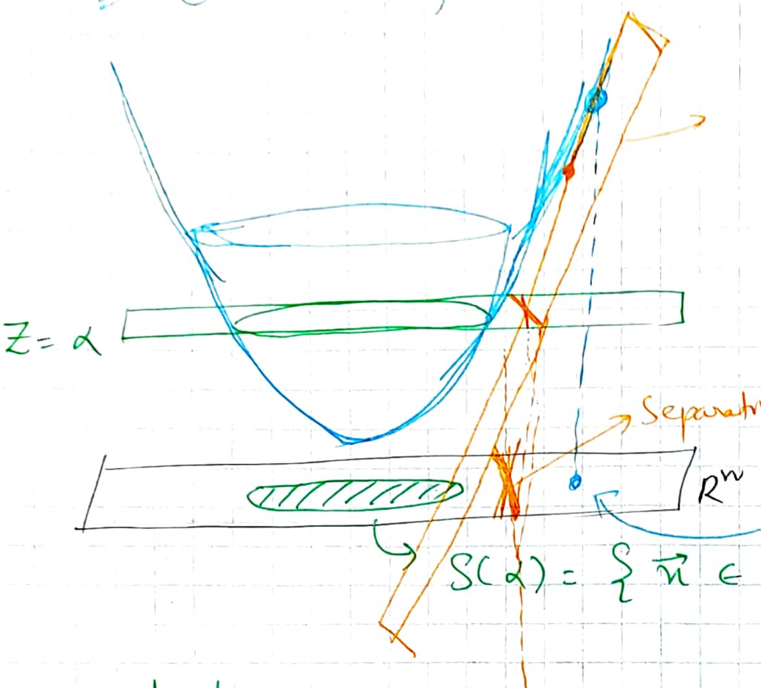
⇒ $\underline{\underline{n}} = m$ which is a contradiction since we started with $n < m$

we know that A is affinely independent & must span affinely n dimension.

But ~~since~~ ~~now~~, ~~we~~ we found that it spans m dimension, which is a contradiction.

* So this is a Matroid.

Q1. (a) Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex.



CASE-1 : $\vec{x}_0 \notin S(\alpha)$

$g(\vec{x}) \approx f(\vec{x})$ = tangent at x_0

$g(\vec{x}) = f(\vec{x}_0) + \vec{\nabla} f(\vec{x}_0) \times (\vec{x} - \vec{x}_0)$

set $g(\vec{x}) = \alpha$.

$\vec{x}_0$ [given point for which we want to find a separating hyperplane from $S(\alpha)$]

$S(\alpha) = \{ \vec{x} \in \mathbb{R}^n : f(\vec{x}) \leq \alpha \}$

we can check $f(\vec{x}_0) > \alpha$ or $f(\vec{x}_0) \leq \alpha$

If $f(\vec{x}_0) > \alpha$, then find the separating hyperplane

Setting $g(\vec{x}) = \alpha$

$\Rightarrow f(\vec{x}_0) + \vec{\nabla} f(\vec{x}_0) (\vec{x} - \vec{x}_0) = \alpha$

$\Rightarrow \vec{\nabla} f(\vec{x}_0) \cdot (\vec{x} - \vec{x}_0) = \alpha - f(\vec{x}_0)$

which is the eq^n of the separating hyperplane.

$\vec{\nabla} f(\vec{x}_0) \cdot (\vec{x} - \vec{x}_0) = \alpha - f(\vec{x}_0)$

$\Rightarrow \vec{\nabla} f(\vec{x}_0) \cdot \vec{x} = [\alpha - f(\vec{x}_0)] + \nabla f(\vec{x}_0) \cdot \vec{x}_0$

normal to the separating hyperplane

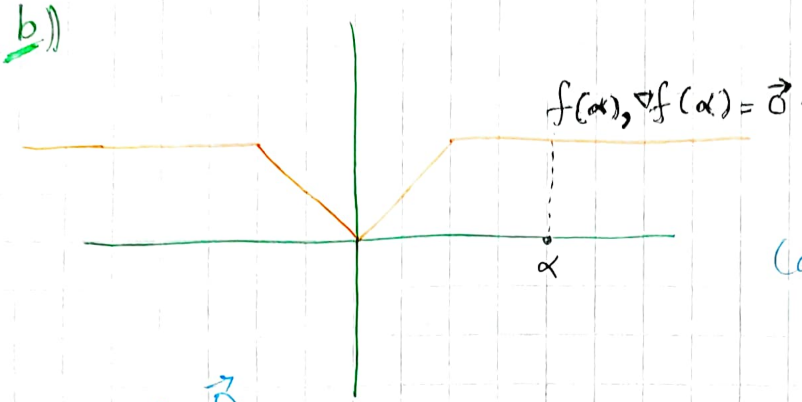
If we just need the normal, then we can just return $\vec{\nabla} f(\vec{x}_0)$.

this describes the hyperplane in $\mathbb{R}^n$ that separates $\vec{x}_0$ from $S(\alpha)$.

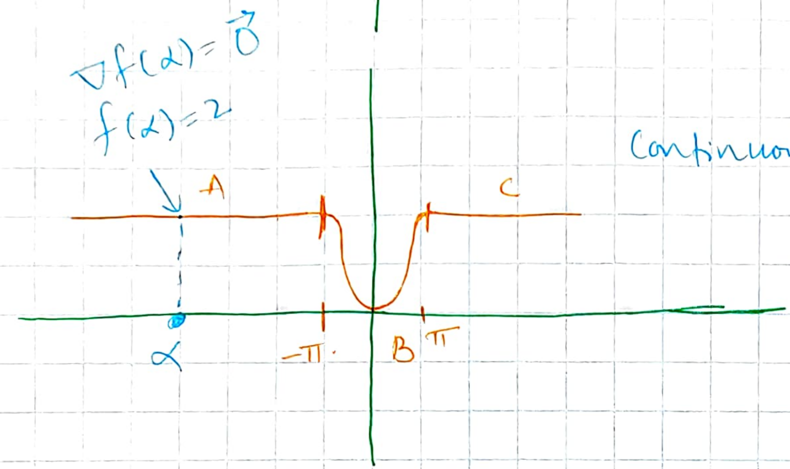
CASE: 2 $\vec{x}_0 \in S(\alpha)$

Just get $f(\vec{x}_0)$ & see if $f(\vec{x}_0) = \alpha$ & if so we can return that $\vec{x}_0 \in S(\alpha)$.

Q1 b)



Continuous, not smooth.



Continuous & smooth

$A \equiv f(x) = 2$

$B \equiv f(x) = 1 - \cos x$

$C \equiv f(x) = 2$

So $f(x) = \begin{cases} 1 - \cos x, & x \in [-\pi, \pi] \\ 2, & \text{Otherwise} \end{cases}$

In $\mathbb{R}^2$

$f(x,y) = \begin{cases} 1 - \cos\left(\sqrt{x^2+y^2}\right), & \sqrt{x^2+y^2} \in [0, \pi] \\ 2, & \text{otherwise} \end{cases}$

in $\mathbb{R}^n$

or $f(\vec{r}) = \begin{cases} 1 - \cos(|\vec{r}|), & |\vec{r}| \in [0, \pi] \\ 2, & \text{otherwise.} \end{cases}$

Q1 c) We can encode the unique SAT into searching over the vertices of a hypercube in $\mathbb{R}^n$ where there are $n - \{+1, -1\}$ ~~boolean~~ variables in the unique SAT.

* In this encoding, since there is only one setting that solves the unique SAT, there is only one vertex that will solve the problem.

* The vertices of the hypercube are of the form $\underbrace{(\pm 1, \pm 1 \ldots\ldots \pm 1)}_{n\text{-times}}$

We define a smooth continuous function that is zero at the ~~soln~~ vertex that gives the sol^n & greater than zero everywhere else.

$$f(\vec{\gamma}) = \begin{cases} 1 - \cos|\vec{\gamma} - \vec{\gamma}_0| & \text{if } |\vec{\gamma} - \vec{\gamma}_0| \leq \pi \\ 2, & \text{Otherwise} \end{cases}$$

* Here $\vec{\gamma}_0$ is the vertex ~~which~~ which solves the unique SAT.

→ Clearly the above function is continuous & differentiable.

→ It is also a convex function.

→ Since the function above is difficult to evaluate, without knowing $\vec{\gamma}_0$ apriori, we will implement an algorithm that can easily evaluate

(this is a catch 22, we ~~can't~~ can't know $\vec{\gamma}_0$ apriori & hence can't construct or evaluate the convex function above.)

* We could have set $g(\vec{\gamma}) = \|\vec{\gamma} - \vec{\gamma}_0\|^2$ which is continuous, differentiable & convex. But g is hard to evaluate. We will show that f is easy to evaluate.

def $f(\vec{\gamma})$:

→ Let $x_1, x_2, \ldots\ldots x_n = \vec{\gamma}$ # components of $\vec{\gamma}$.

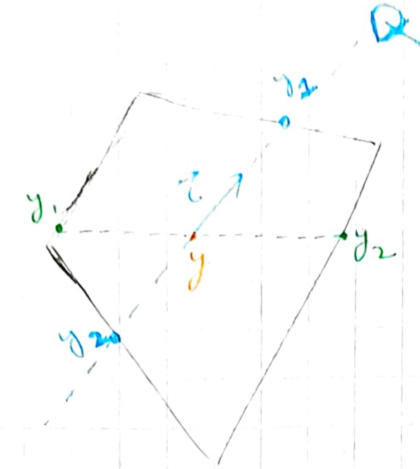
→ Set $y_i = \text{Sign}(x_1) = \frac{|x_1|}{x_1} = \frac{x_1}{|x_1|}$

easy to evaluate & check.

→ Check if $(y_1, y_2, \ldots y_n)$ is ~~equal~~ the vertex that solves the unique SAT.

→ If solves, return $f(\vec{\gamma})$ & $\nabla f(\vec{\gamma})$ since $\vec{\gamma}_0$ is known now. If not, return $f = 2$ & $\nabla f = 0$.

Q2



Pick a random direction $\vec{c}$.

Method - I : Using 2 linear programs

L.P1

$\max \alpha$

s.t.

$y_1 \in P$

$\vec{y}_1 - \alpha\vec{c} = \vec{y}$

L.P2

$\min \alpha$

s.t.

$y_2 \in P$

$\vec{y}_2 - \alpha\vec{c} = \vec{y}$

* The above 2 L.P's can be solved since we have a separation oracle over P & we can easily come up with a separation oracle over the line Q.

* So basically we want to search over the intersection of P & the line Q, both of which are convex & for both we have a separation oracle. We know that the intersection of 2 convex sets can be identified by using 2 separating oracles i.e. if both oracles return true for feasibility, we know the point is in the intersection which is also a convex set. If anyone returns not feasible, then use the separating hyperplane returned by it, if both return not feasible, then use any of the separating hyperplanes returned by them.

Method - II : Find the longest line segment via y

$$\max \ \|y_1 - y_2\|^2 \leftarrow \text{convex}$$

s.t.

$$y_1 \in P \leftarrow \text{Convex}$$
$$y_2 \in P \leftarrow \text{Convex}$$

$$\alpha_1 y_1 + \alpha_2 y_2 = y \leftarrow \text{non-convex.}$$

However, from HW3, we know how to convert ~~problems~~ optimizations with this kind of constraint into a optimization/problem which is is convex using

Change of variables by perspective transforms.

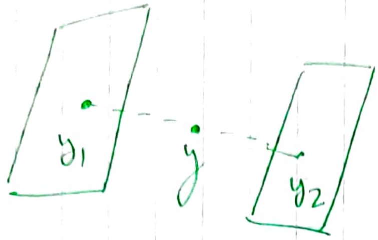
(b) Since we know y_1 & y_2 (lie on the boundary), ask the oracle to give us separating hyperplane/violated constraint

$$\left.\begin{aligned} \text{for} \quad & y = y_1 + \epsilon(y_1 - y_2) \\ \& \quad & y = y_2 + \epsilon(y_2 - y_1) \end{aligned}\right\} \epsilon \to 0^+.$$

* Note that the violated constraint eq^n is a valid and weakly separating hyperplane.
* Also note that even if the oracle gave us an ~~random~~ arbitrary separating hyperplane instead of the violated constraint, we will still get the approx. eq^n for the violated constraint if we make $\epsilon \to 0$.

Q2

(c)



$y = \alpha_1 y_1 + \alpha_2 y_2$

$\alpha_1 + \alpha_2 = 1$

$y_1 = \beta_1 z_1 + \beta_2 z_2 \quad \{ \beta_1 + \beta_2 = 1$

$y_2 = \gamma_1 w_1 + \gamma_2 w_2 \quad \{ \gamma_1 + \gamma_2 = 1$

$y = \alpha_1 (y_1) + \alpha_2 y_2$

$= \alpha_1 (\beta_1 z_1 + \beta_2 z_2) + \alpha_2 (\gamma_1 w_1 + \gamma_2 w_2)$

$= \alpha_1 \beta_1 z_1 + \alpha_1 \beta_2 z_2 + \alpha_2 \gamma_1 w_1 + \alpha_2 \gamma_2 w_2$

$= \delta_1 z_1 + \delta_2 z_2 + \delta_3 w_1 + \delta_4 w_2$

Clearly, $\delta_1 + \delta_2 + \delta_3 + \delta_4$

$= \alpha_1 \beta_1 + \alpha_1 \beta_2 + \alpha_2 \gamma_1 + \alpha_2 \gamma_2$

$= \alpha_1 (\beta_1 + \beta_2) + \alpha_2 (\gamma_1 + \gamma_2)$

$= \alpha_1 * 1 + \alpha_2 \times 1$

$= \alpha_1 + \alpha_2$

$= 1.$

So y is a convex combination of z_1, z_2, w_1 & w_2.

Algorithm

(1) find y_1, y_2 using part (a)

(2) find tight constraints at y_1 & y_2. Let call these 2 hyperplanes from the tight constraints H_1 & H_2.

(3) Using (a), Solve for z_1 & z_2 where $z = y_1$ on the intersection of H_1 & P.

* Do the same for $w = y_2$.

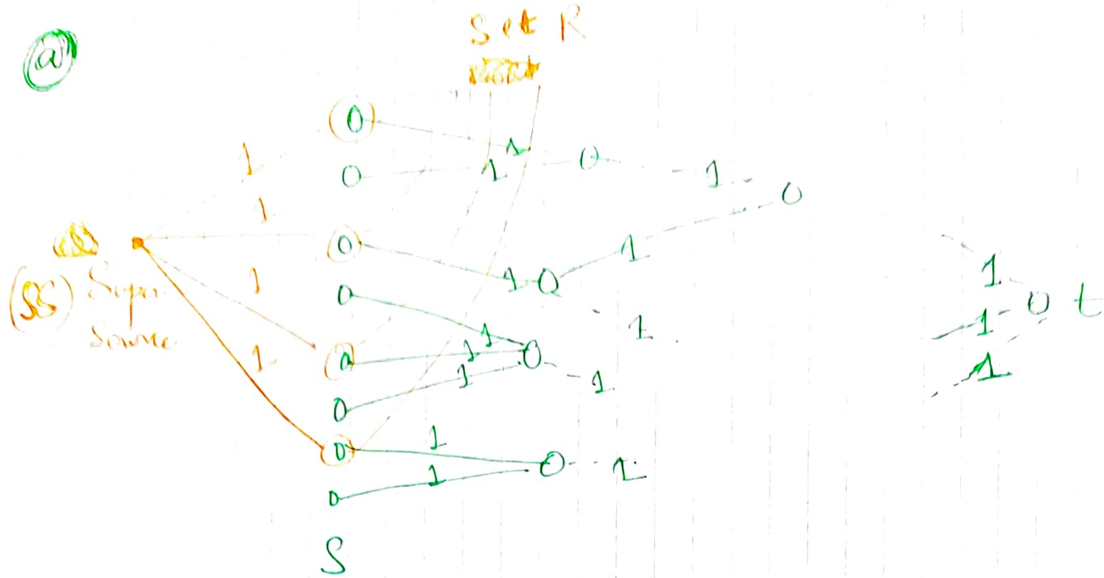
(4) Recurse till we get 'n' tight constraints which is when we have arrived at the

$\downarrow$ $\boxed{\log_2(n+1)\ \text{poly}(n, B, \langle y \rangle)}$

Since we are in $\mathbb{R}^n$, & Since y is inside the polytope, by Caratheodory's theorem, we just will end up with $n+1$ points at max to represent y as a convex combination of

So at max, we will iterate $\log_2(n+1)$ times to get to the $n+1$ points (at max $n+1$ points).

Q.4. (a)



→ Connect the Sources from R to a supersource.

→ Set capacity of all edges to 1.

→ Use Ford-Fulkerson to find max-flow.

If max-flow $\neq$ #R, then R not routable

else R is routable.

(b) Let $\vec{x}_R$ be an indicator variable. $\vec{x}_R \in \{0,1\}^{|S|}$.

$$x_R[i] = \begin{cases} 1 & \text{if } i \in R \\ 0 & \text{if } i \in S/R \quad i \in S-R \end{cases}$$

If R is routable, then $\#R \leq$ Capacity of any SS-t cut.

as other-wise R cannot be routable.

So $$\mathbb{1}^T \vec{x}_R \leq \sum c_e$$

$e \in$ edges cut by arbitrary SS-t cut.

Q4 (c) maximize $w^T x_{B}$

$$\text{s.t } 1^T x \le \sum c_e$$

$e \in$ edges cut by arbitrary s-t cut.

* We already have a separation oracle from (a) for the sets that are routable in polynomial time since the capacities are integers.

* ~~Since we~~ So we can run the above convex maximization problem in polynomial time.

Q3 (a) Dual LP: $\min b^T y$

$$A^T y = c$$

$$y \geq 0$$

For primal LP We know that the soln occurs at a vertex of $Ax \le b$

So exactly 'n' constraints are tight.

So, due to complementary slackness, exactly n entries in y are non-zero. We can keep track of which constraints are tight in the primal & then optimize over only the values of y that correspond to those constraints while setting / holding / constraining the other entries of y to zero.

* We can represent the soln of dual L.P. using only the non-zero entries of $\vec{y}$.

(b) From HW2, we know that the dual of an L.P is dependent on the algebraic form of the primal L.P.

for eg., the dual L.P for the 2 primal L.Ps are different

$$\max c^T x \quad \Big\| \quad \max c^T x$$
$$Ax \le b \quad \Big\| \quad 2Ax \le 2b$$

although the feasible region is the same.

Since the arbitrary separation oracle has no idea of the algebraic form of the feasible region, the oracle cannot be used to get a soln of the dual L.P as the oracle has no idea of the exact algebraic form of our representation of primal L.P.'s feasible region

(c) * Let V^* be the optimal for primal.

* In HW2 we showed c/V^* is in the polar of the normalized primal's feasible region (which is convex combination of rows of A')

* Using Q2 from this HW4, we can write c/V^* as a convex combination of rows of A'

* Let the weights for the convex combinations be $\frac{y_i}{V^*} = \alpha_i$

Such that $\sum_i \frac{y_i}{V_*} = 1 = \sum_i \alpha_i$

Now we got the α_i's from the algo in Q3.

Now just get $y_i = V^* \times \alpha_i$.

So we get the optimal setting for dual of the normalised primal L.P.