# CMSC 510 – L16
# Regularization Methods for Machine Learning

Instructor:
Dr. Tom Arodz

# Recap: Lovasz extension

- **Lovasz extension of Ω: $\Omega^L$** $\Omega^L(w) = \sum_{i=0}^{F} \lambda_i \Omega(S_i)$

  where: $\emptyset = S_0 \subset S_1 \subset S_2 \subset \ldots \subset S_F = V$

  $\sum_{i=0}^{F} \lambda_i 1_{S_i} = w, \sum_{i=0}^{F} \lambda_i = 1, \lambda_i \geq 0$

- **How to evaluate it for vector w in $[0,1]^F$?**
  - Order elements in V in decreasing order of w's:
    - $V=\{v_1,v_2,\ldots,v_F\}$ such that $w_1 \geq w_2 \geq \ldots \geq w_F$
  - Set: $S_0=\emptyset$, $\lambda_0=1-w_1$

    $S_i=S_{i-1}+\{v_i\}=\{v_1,\ldots,v_i\}$, $\lambda_i = w_i - w_{i+1}$

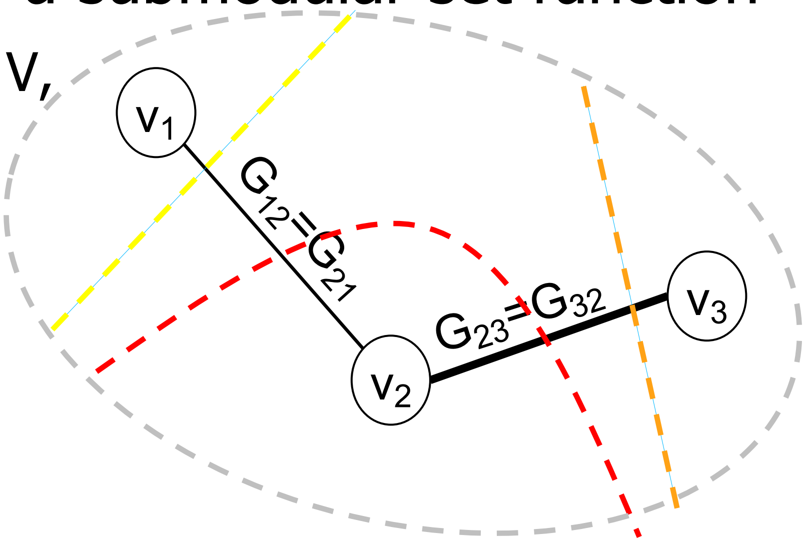
    $S_F=V$ $\lambda_F=w_F$
  - Two alternative formulas:

$$\Omega^L(w) = (1-w_1)\Omega(S_0) + w_F\Omega(S_F) + \sum_{i=1}^{F-1}(w_i - w_{i+1})\Omega(S_i)$$

$$= \Omega(S_0) + \sum_{i=1}^{F} w_i \left[\Omega(S_i) - \Omega(S_{i-1})\right]$$

# Recap: Graph cut capacity

- **Undirected graph cut capacity** is a submodular set function
  - We have a graph G over vertices in set V, with undirected, weighted edges with weight $G_{jk}$ between element $v_j$ and $v_k$




  - Set function $\Omega(S) = \sum_{j \in S} \sum_{k \notin S} G_{jk}$

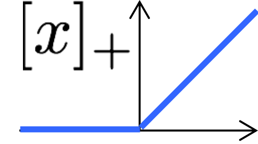
$$= \frac{1}{2} \sum_{j=1}^{F} \sum_{k=1}^{F} G_{j,k} \left| (1_S)_j - (1_S)_k \right|$$

is submodular.

Notation: $(\mathbf{1_S})_j$=1 if j is in S
=0 if not
$1_S$ – vector of 0's/1's representing S

- Interpretation of Ω in machine learning:
  - Ω([w]) = number of edges in graph of features that link features in the model represented by vector w to features not in the model

# Recap: Graph cut capacity




$$[x]_+ = \max(x, 0)$$

- **Directed graph cut capacity** is a submodular set function
  - Ω(S) = total weights of edges **from S to V-S**

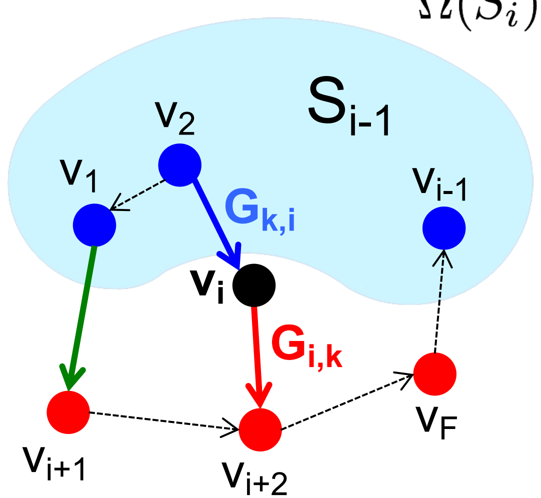
$$\Omega(S) = \sum_{j \in S} \sum_{k \notin S} G_{jk}$$

- Start from: $\Omega^L(w) = \sum_{i=1}^{F} w_i \left[\Omega(S_i) - \Omega(S_{i-1})\right]$
- Final formula is:

$$\Omega^L(w) = \sum_{i=1}^{F} \sum_{k=1}^{F} G_{i,k} [w_i - w_k]_+$$

$$\Omega(S_i) - \Omega(S_{i-1}) = \underbrace{\sum_{k=i+1}^{F} G_{i,k}}_{\text{red edges}} - \underbrace{\sum_{k=1}^{i-1} G_{k,i}}_{\text{blue edges}}$$

$S_i = S_{i-1} + \{v_i\}$
$= \{v_1, \ldots, v_i\}$




Move from $S_{i-1}$ to $S_i$

Red edges start playing a role

Blue edges stop playing a role

Green edges: no change

Black edges: play no role

# Graph cut: Lovasz extension

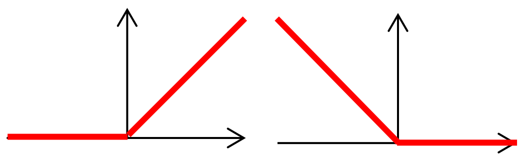
- Directed graph cut has Lovasz extension:

$$\Omega^L(w) = \sum_{i=1}^{F} \sum_{k=1}^{F} G_{i,k}[w_i - w_k]_+$$

- Undirected graph cut:
  - Replace each undirected edge by two edges, one in each direction
  - Apply the *directed cut* formula twice (for S=S, and S=V-S), divide by 2

$$\Omega^L(w) = \frac{1}{2} \sum_{i=1}^{F} \sum_{k=1}^{F} G_{i,k}([w_i - w_k]_+ + [w_k - w_i]_+)$$

  - Use the formula: $|x| = [x]_+ + [-x]_+$

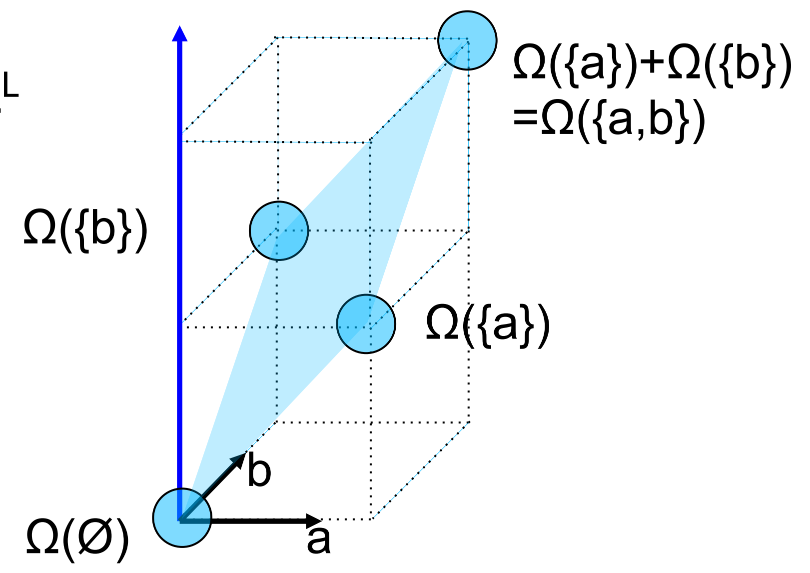


  - Final formula for **Lovasz extension of undirected graph cut**:

$$\Omega^L(w) = \frac{1}{2} \sum_{i=1}^{F} \sum_{k=1}^{F} G_{i,k}|w_i - w_k|$$

# Recap: "ID Card" for Set cardinality

- **Set cardinality** is a modular (and thus submodular) set function
  - $\Omega(S)=|S|$
- Interpretation of $\Omega$ in machine learning:
  - $\Omega([w])$ = number of features in the model represented by vector w
  - Model preferred by $\Omega^L(w)$: lost of feature weights $w_f$ are 0
- Lovasz extension on $[0,1]^F$ :
  - $\Omega^L(w) = \Sigma_f w_f$ (we derived it previously)
  - We have derived it from definition of $\Omega^L$
- Extension to $R^F$:
  - $\Omega^{L\infty}(w) = ||w||_1 = \Sigma_f |w_f|$
  - $L_1$ penalty
- Minimum of $prox_{\Omega^L}(v)$ :
  - soft thresholding of v



# Graph cut capacity

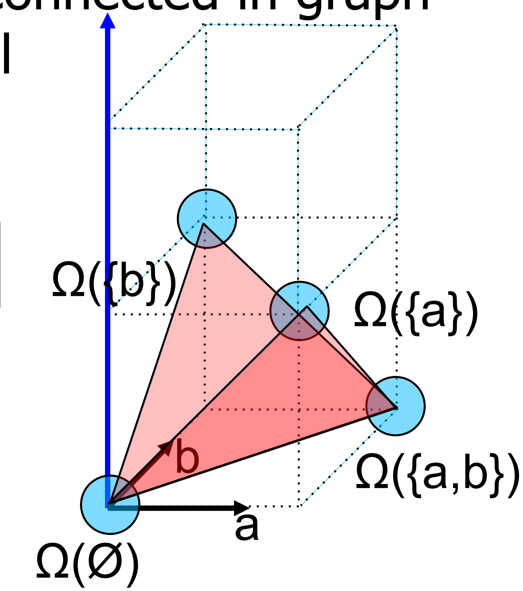
- **Undirected graph cut capacity** is a submodular set function
  - For undirected graph with weights $G_{jk}$

$$\Omega(S) = \frac{1}{2}\sum\sum_{j,k=1}^{F} G_{j,k}\left|(1_S)_j - (1_S)_k\right|$$

- Interpretation of Ω in machine learning:
  - Ω([w]) = number of edges that link features in the model w to features not in the model
  - Model preferred by $\Omega^L(w)$: weights of features $w_f$ connected in graph are similar or identical
- Lovasz extension on $[0,1]^F$ = extension to $R^F$:

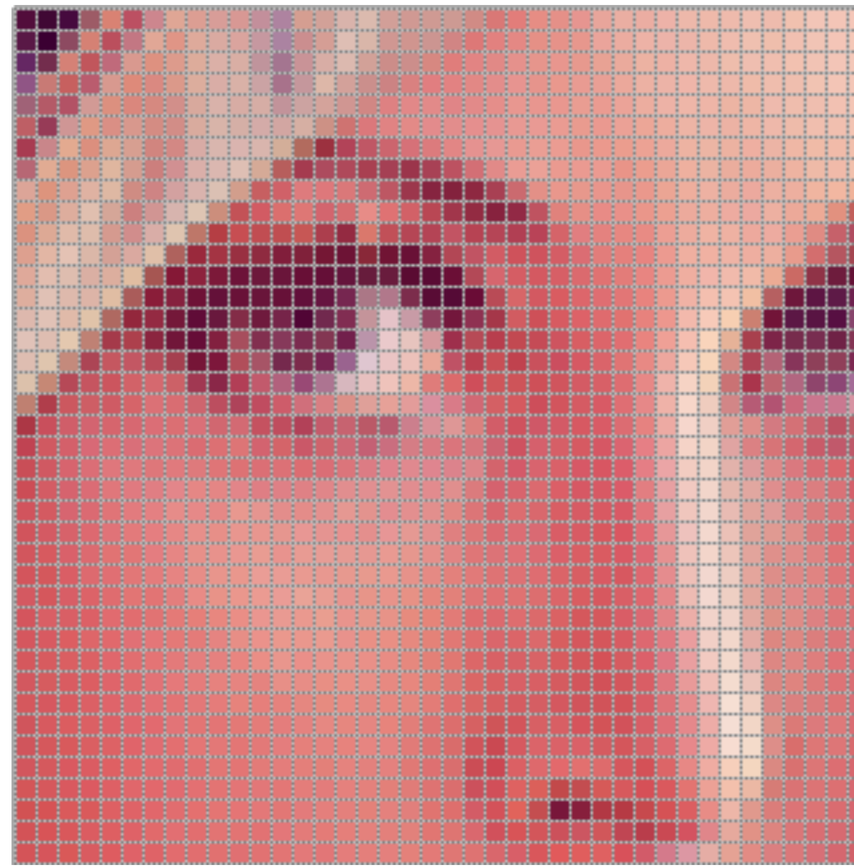
$$\Omega^L(w) = \frac{1}{2}\sum\sum_{j,k=1}^{F} G_{j,k}\left|w_j - w_k\right|$$

- Minimum of $\mathrm{prox}_{\Omega^L}(v)$:
  - Can be solved by QP, *max trick*
    e.g. |x-y|=max( x-y, -(x-y) )



# Applications: graph cut penalty

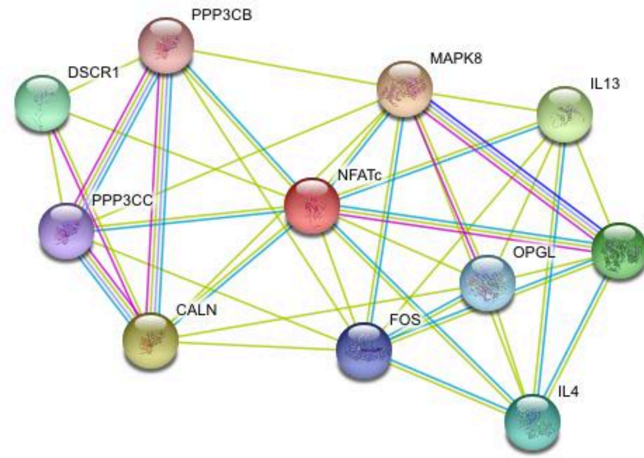
- **Images (and lattice signals in general, e.g. MRI (3D) or gene copy number alterations (1D))**
- We can build a graph where each pixel is linked to its neighbors with a certain weight
  - And second-degree neighbors with lower weight, etc.
- Neighboring pixels often describe the same "feature"
  - E.g. corner of the mouth, we're classifying smiling vs sad
- If a "feature" is important it shows up in a group of connected (i.e. neighboring) pixels

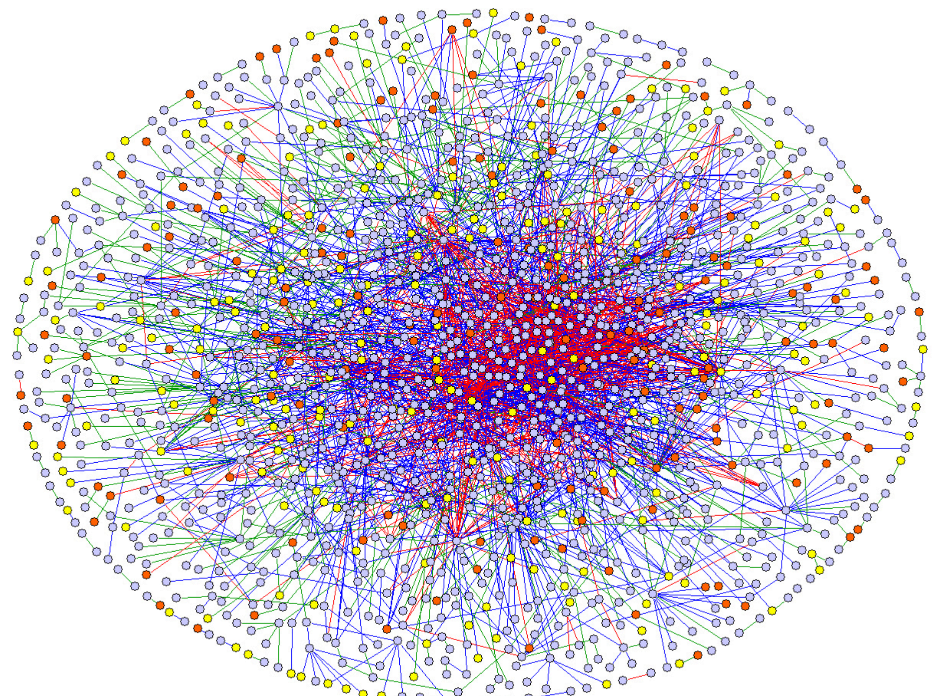


- Graph cut: minimize the border of the neighborhood

# Applications: graph cut penalty

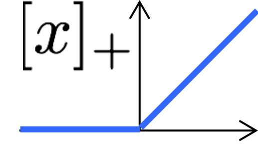
- **Biology:** Molecular entities (proteins, genes) are linked by a network of molecular reactions
  - E.g. protein A modifies (phosphorylates) protein B
  - Gene A turns on/off gene B
- When we have e.g. a mutation in gene A, it’s consequences show up downstream (in genes C, where A->->C)
- Inter-class change will show in whole connected subgraphs



- Graph cut: minimize the border of the connected subgraph

# Submodular set functions




$[x]_+ = \max(x,0)$

- **Directed graph cut capacity** is a submodular set function
  - Ω(S) = total weights of edges **from S to V-S** **j->k**

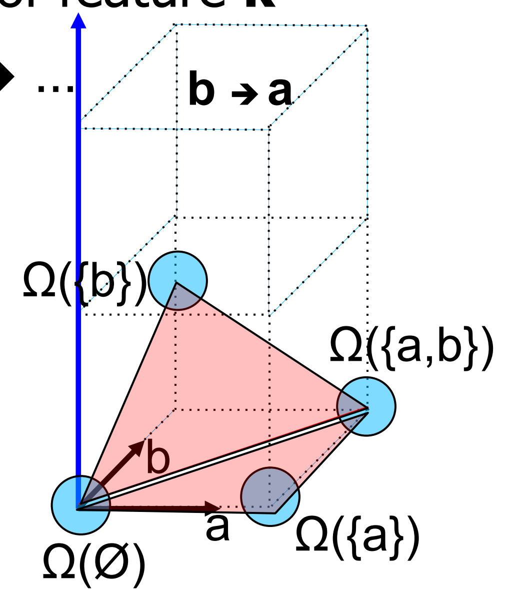
$$\Omega(S) = \sum_{j\in S}\sum_{k\notin S} G_{jk} = \sum_{j=1}^{F}\sum_{k=1}^{F} G_{j,k}\left[(1_S)_j - (1_S)_k\right]_+$$

- Interpretation of Ω in machine learning:
  - Ω([w]) = number of edges *from* features in the model *to* features not in the model
  - Model preferred by $\Omega^L(w)$ = for every **j->k** edge in G, weight of feature **j** doesn't exceed weight of feature **k**
    - e.g. in chain graph of features **a** ➔ **b** ➔ **c** ➔ **d** ➔ … we prefer: $w_a \le w_b \le w_c \le \mathbf{w_d} \le$
- Lovasz extension on $[0,1]^F$ = extension to $R^F$:

$$\Omega^L(w) = \sum_{j=1}^{F}\sum_{k=1}^{F} G_{j,k}\left[w_j - w_k\right]_+$$

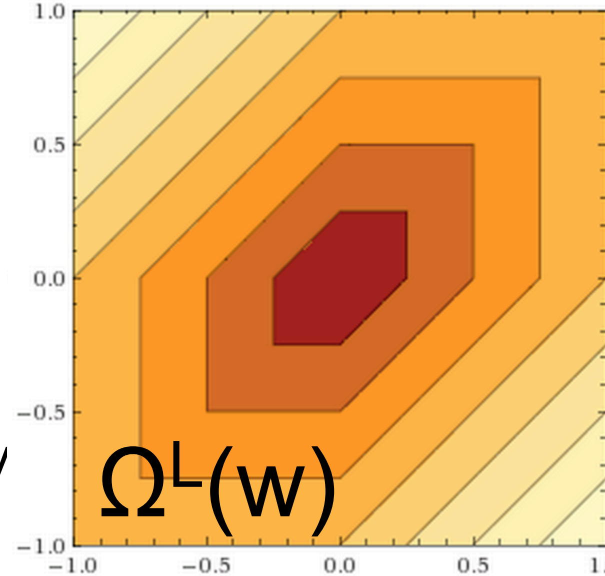
**Null if $w_j$ <= $w_k$**

- Minimum of $\mathrm{prox}_{\Omega^L}(v)$:
  - Can be solved by QP, *max trick* e.g. $|x|_+ = \max(x,0)$



# Conical combination

- We have seen some submodular set functions
- Can we get other submodular function from them?
  - Conical (i.e. nonnegative-weight linear) combination
    - If functions $g_i : 2^V \to \mathbb{R}$ are submodular, and $\alpha_i \geq 0$
      Then: $f(S) = \sum_{i=1}^{n} \alpha_i g_i(S)$ is submodular
    - **Their Lovasz extensions add up**
  - Example:
    - Undirected graph cut capacity
      \+ set cardinality
    - Preferred model: connected features should have similar weights, small boundary to features that are not selected ($w_i$=0), many weights should be zero




$$\Omega^L(w) = \lambda_1 \sum \sum_{j,k=1}^{F} G_{j,k} |w_j - w_k| + \lambda_2 \sum_{j=1}^{F} |w_j|$$

# Composition with concave

- We have seen some submodular set functions
- Can we get other submodular function from them?

- Composition with concave non-decreasing funct.:
  - If Ω(S) is a **non-decreasing** submodular function, i.e.,

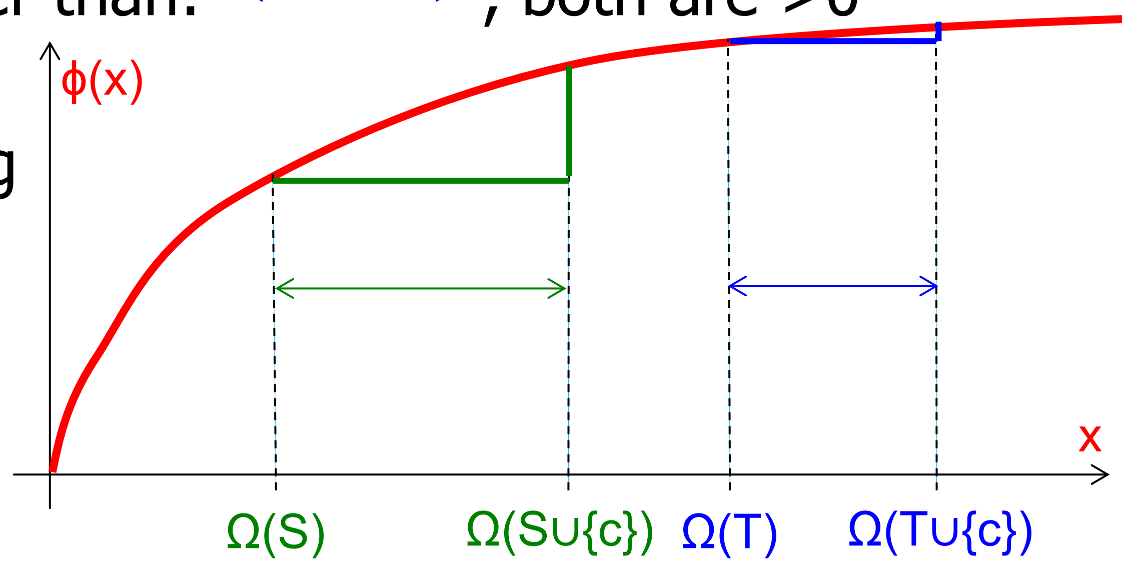
    $$A, B \subseteq V, A \subseteq B \implies \Omega(A) \leq \Omega(B)$$

    and $\phi : \mathbb{R} \to \mathbb{R}$ is (at least on region [Ω(Ø),Ω(V)]) concave and non-decreasing (we assume also ϕ(0)=0)

  - Then: $f(S) = \phi(\Omega(S))$ is submodular

# Composition with concave

- Composition with concave non-decreasing funct.:
  - If Ω(S) is a **non-decreasing** submodular function, i.e.,
    $$A, B \subseteq V, A \subseteq B \implies \Omega(A) \leq \Omega(B)$$
    and $\phi : \mathbb{R} \to \mathbb{R}$ is (at least on region [Ω(Ø),Ω(V)]) concave and non-decreasing, with ϕ(0)=0
  - Then: $f(S) = \phi(\Omega(S))$ is submodular
- Why?
  $$\Omega(S \cup \{c\}) - \Omega(S) \geq \Omega(T \cup \{c\}) - \Omega(T)$$
  - From Ω submodular and non-decreasing we get:
    $\longleftrightarrow$ longer than: $\longleftrightarrow$ , both are >0
  - From ϕ concave and non-decreasing we get that **vertical difference** is also smaller for blue than for green



# Concave f. of set cardinality

- **Composition of a concave function and set cardinality** is a submodular set function: |S| is nondecreasing submodular
  - Ω(S)=g(|S|)
    where g:R->R is a concave, non-decreasing, with g(0)=0
- Lovasz extension on $[0,1]^F$ : $\Omega^L(w) = \sum_{i=1}^{F} w_i \left[\Omega(S_i) - \Omega(S_{i-1})\right]$
  - Define a sequence
    of variable indices: $\{f_k : w_{f_k} \geq w_{f_{k+1}} \quad \forall k = 1, \ldots, F-1\}$
    That is, $\{f_k\}$ impose a decreasing order on the coordinates of w:

$$w_{f_1} \geq w_{f_2} \geq \ldots \geq w_{f_F}$$

set $S_k$ has k elements!

  - Then:

$$\Omega^L(w) = \sum_{k=1}^{F} w_{f_k} \left(g(k) - g(k-1)\right)$$

- Extension to $R^F$:

$$\Omega^L(w) = \sum_{k=1}^{F} |w_{f_k}| \left(g(k) - g(k-1)\right)$$

# Non-emptiness

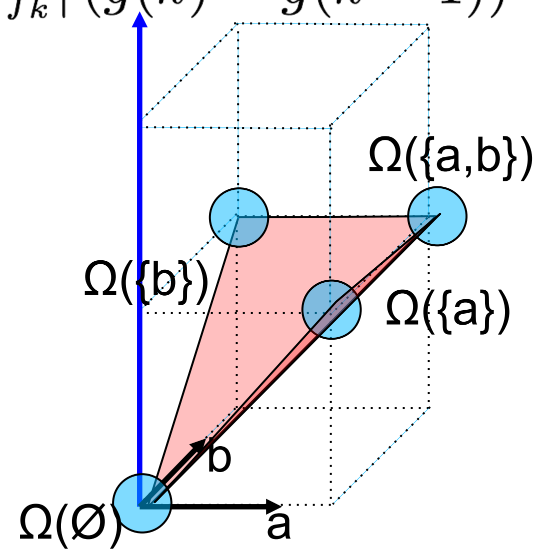
- **Composition of a concave function and set cardinality** is a submodular set function
  - Ω(S)=g(|S|) where g:R->R is a concave function, g(0)=0
  - g(s)=min(s,1) => Ω(S)=min(|S|,1)      g(0)=0, g(1)=1, g(2)=1,…
- Interpretation of Ω in machine learning:
  - Ω([w]) = does the model represented by **w** contain any feature?
  - Model preferred by $\Omega^L(w)$: maximum weight $w_f$ of any feature is 0
- Lovasz extension on $[0,1]^F$ : $\Omega^L(w) = \sum_{k=1}^{F} |w_{f_k}| \left(g(k) - g(k-1)\right)$

$$\Omega^L(w) = ||w||_\infty = \max_{f=1}^{F} w_f$$

- Extension to $R^F$:

$$\Omega^L(w) = ||w||_\infty = \max_{f=1}^{F} |w_f|$$

- Minimum of $\mathrm{prox}_{\Omega^L}(v)$:
  - Can be solved by QP, *max trick* (twice):
    e.g. max(|x|,|y|)=max(x,-x,y,-y)



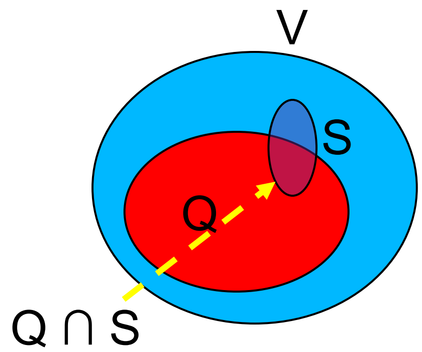
# Non-emptiness

  - $\Omega(S)=\min(|S|,1)$
- Interpretation of $\Omega$ in machine learning:
  - $\Omega([w])$ = does the model represented by **w** contain any feature?
  - Model preferred by $\Omega^L(w)$: maximum weight $w_f$ of any feature is 0
- That doesn't seem to make much sense: we want some features to have non-zero weights
- But it will be useful building block in defining a more useful submodular set function

# Extension of submodular f.

- We have seen some submodular set functions
- Can we get other submodular function from them?
  - **Extension**:
    - if we have a fixed set $Q \subset V$,
      and we have a submodular set function $\Gamma:2^Q \rightarrow R$
      defined on subsets of Q

      Then we can define a function $\Omega:2^V \rightarrow R$
      as $\Omega(S)=\Gamma(Q \cap S)$
      and $\Omega$ is submodular




      - Basically, to get $\Omega(S)$, we restrict S to Q, and use $\Gamma(Q \cap S)$
    - E.g. $\Omega_Q(S)=\min(|Q \cap S|,1)$ is submodular

# Proof (omit)

- **Extension**:
  - if we have a fixed set Q ⊂ V, and we have a submodular set function $\Gamma:2^Q \to R$ defined on subsets of Q
    Then we can define a function $\Omega:2^V \to R$
    as $\Omega(S)=\Gamma(Q \cap S)$ and $\Omega$ is submodular
- Why? Definition of submodularity:

$$\forall S, T, \{c\} \subseteq V,\ S \subseteq T,\ \Omega(S \cup \{c\}) - \Omega(S) \geq \Omega(T \cup \{c\}) - \Omega(T)$$

  - If c not in Q, both sides equal to 0. We consider cases "c in Q"
  - If c in Q:
    - If T in Q, then S in Q, and we have c in Q, so submodularity of Ω follows directly from submodularity of Γ

# Proof (omit)

- **Extension**:
  - if we have a fixed set Q ⊂ V, and we have a submodular set function $\Gamma:2^Q \rightarrow R$ defined on subsets of Q
    Then we can define a function $\Omega:2^V \rightarrow R$
    as $\Omega(S)=\Gamma(Q \cap S)$ and Ω is submodular
- Why? Definition of submodularity:

$$\forall S, T, \{c\} \subseteq V,\ S \subseteq T,\ \Omega(S \cup \{c\}) - \Omega(S) \geq \Omega(T \cup \{c\}) - \Omega(T)$$

  - If "c in Q", but not "T in Q"
  - we construct $T'=Q\cap T$, $S'=Q\cap S$, we have: T' contains S'
  - since c is in Q we have $T'\cup\{c\} = Q\cap(T\cup\{c\})$
    $S'\cup\{c\} = Q\cap(S\cup\{c\})$
  - We have $\Omega(S\cup\{c\})-\Omega(S)=\Gamma(S'\cup\{c\})-\Gamma(S')$
    $\geq \Gamma(T'\cup\{c\})-\Gamma(T')=\Omega(T\cup\{c\})-\Omega(T)$
    where the inequality comes from submodularity of Γ.
    So Ω is submodular

# Group absence

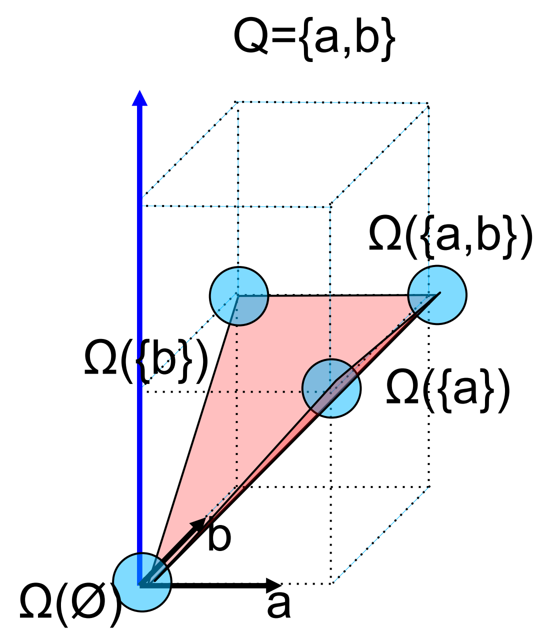
- For a group of features Q, $\boldsymbol{\Omega_Q(S)=\min(|Q\cap S|,1)}$ is a submodular set function
- Interpretation of Ω in machine learning:
  - Ω([w]) = does the model represented by **w** contain any feature from Q?
  - Model preferred by $\Omega^L(w)$: max. weight $w_f$ of any feature from Q is 0
- Lovasz extension on $[0,1]^F$ :

$$\Omega^L(w) = \max_{f\in Q} w_f = ||w_Q||_\infty$$

- Extension to $R^F$:

$$\Omega^L(w) = \max_{f\in Q} |w_f| = ||w_Q||_\infty$$

- Minimum of $\mathrm{prox}_{\Omega^L}(v)$:
  - Can be solved by QP, *max trick* : e.g. max(|x|,|y|)=max(max(x,-x),max(y,-y))



# Group absence ($L_1/L_\infty$ norm)

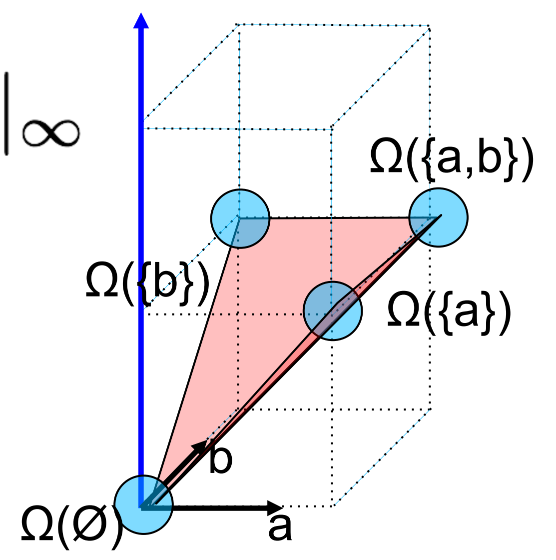
- For a collection of groups of features $\{Q_i\}$, $\Omega(S)=\Sigma_i \min(|Q_i \cap S|,1)$ is a submodular set function
  - Because sum of submodular is submodular
- Interpretation of Ω in machine learning:
  - Ω([w]) = does the model represented by **w** contain any feature from any $Q_i$?
  - Model preferred by $\Omega^L(w)$: max. weight $w_f$ of features from any $Q_i$ is 0 (or: make as many groups $Q_i$ as possible absent from the model)
- Lovasz extension on $[0,1]^F$ :

$$\Omega^L(w) = \sum_i \max_{f\in Q_i} w_f = \sum_i ||w_{Q_i}||_\infty$$

- Extension to $R^F$:

$$\Omega^L(w) = \sum_i \max_{f\in Q_i} |w_f| = \sum_i ||w_{Q_i}||_\infty$$

- Minimum of $\mathrm{prox}_{\Omega^L}(v)$:
  - Can be solved by QP, *max trick* (twice): e.g. max(|x|,|y|)=max(max(x,-x),max(y,-y))



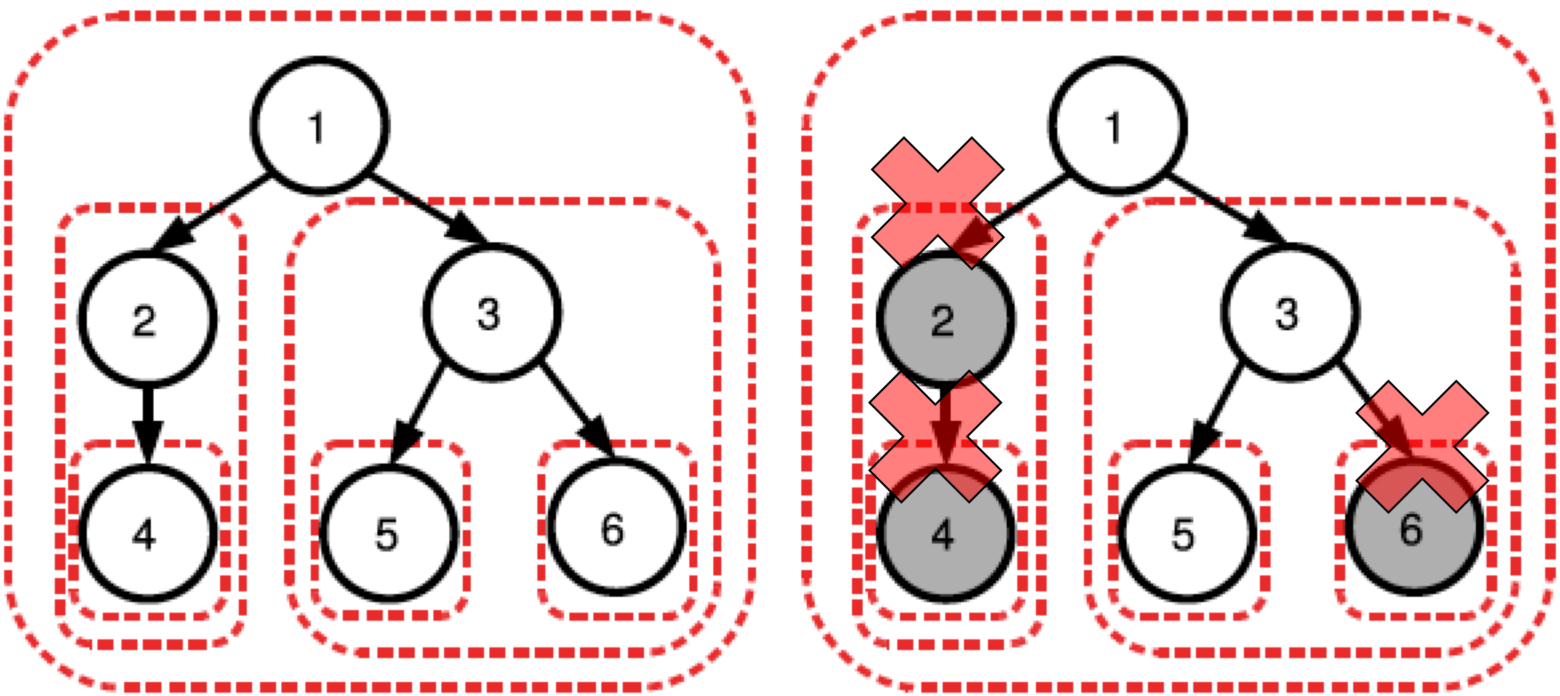
# $L_1/L_\infty$ norm

- A potential use case:
  - Features form a tree
    - Our desire is that
      a feature should be selected
      only
      if its parent features in the tree
      are selected

- How to design a submodular set function promoting this?

# $L_1/L_\infty$ norm

- Ideally, a feature should be selected only if its parent features are selected
- Eliminating only 3 does not reduce penalty
- Eliminating only 5 does reduce penalty

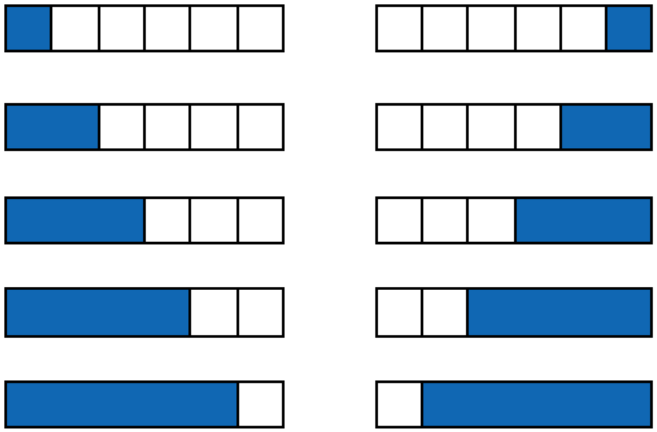



Groups $Q_i$

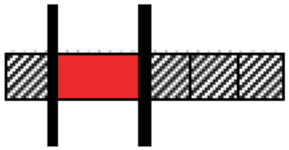
Example of selected features: 1,3,5 – penalty reduced by 3

# $L_1/L_\infty$ norm

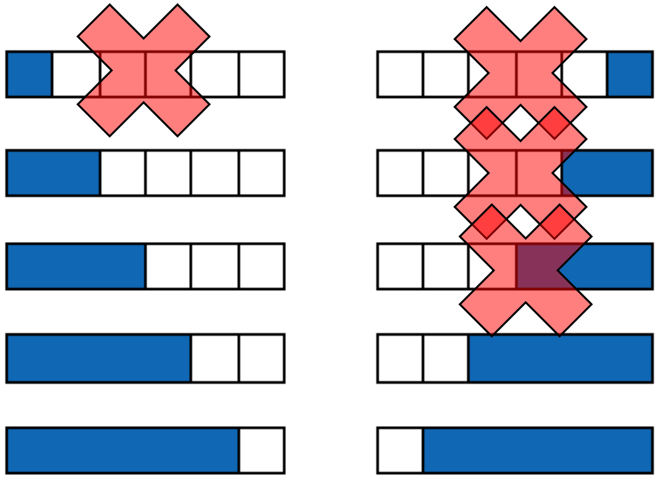
- Eliminating border of a 1D signal



Groups $Q_i$

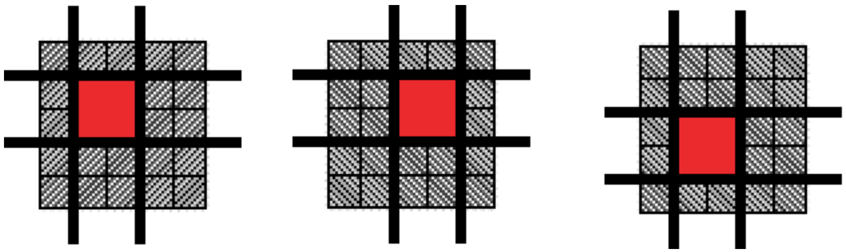


Example of selected features:
2,3

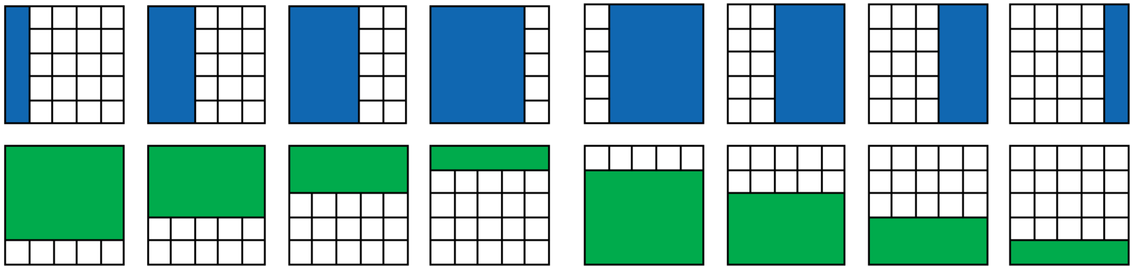


# $L_1/L_\infty$ norm $\Omega^L(w) = \sum_i \max_{f \in Q_i} |w_f| = \sum_i ||w_{Q_i}||_\infty$

- Eliminating borders of a 2D signal



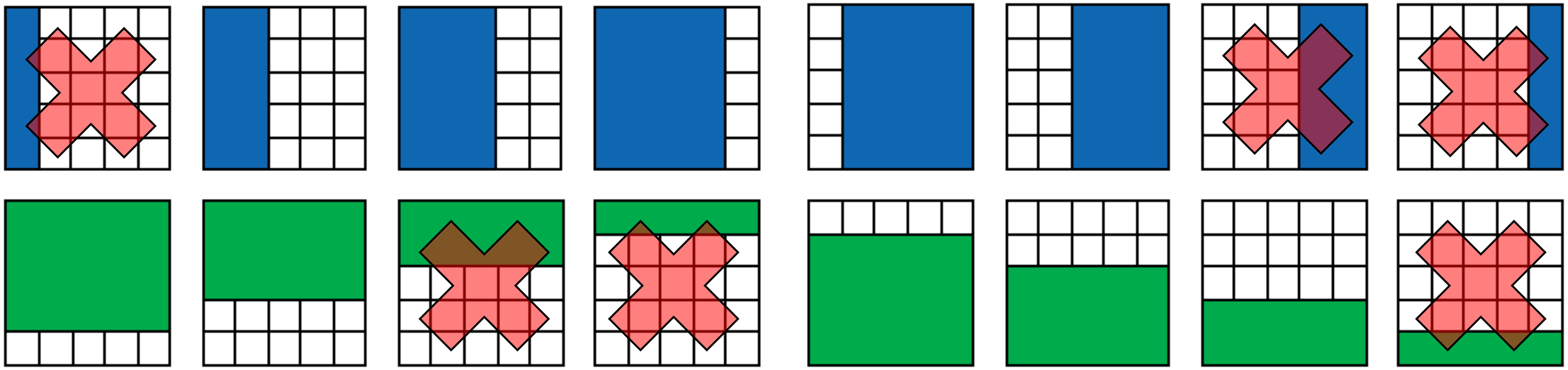
- To achieve squares, we find out what is the pattern of zeros (features outside of rectangle) that we want, to get rectangles
  - That's how we come up with the definition of groups $Q_i$
  - We see that the zeros are always a union of rectangles that touch borders of the image
  - If any combination of these gets zero'ed out during training,



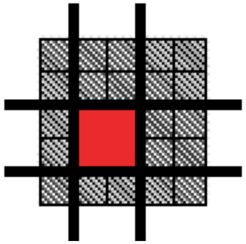
(and those are the only zero-weight features – but we can't guarantee that) we end up with the non-zero features forming a rectangle

# $L_1/L_\infty$ norm

- For example, if during training these groups are pushed to zeros (through minimizing the penalty + risk):



- Then we get a classifier that uses these features (maybe only some of them, we're not penalizing for that):

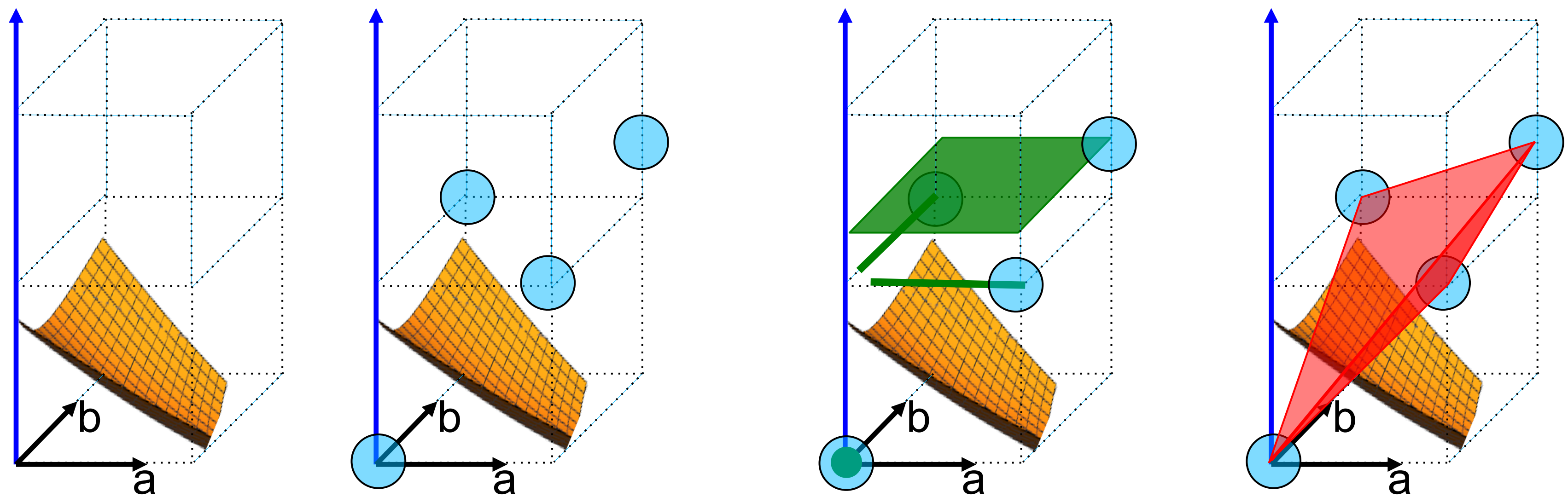


# Big picture

- We can solve problems of the form:
  - **Differentiable risk**
    **+ Lovasz extension of a submodular set function Ω**
  - Solution: e.g. proximal gradient descent
- **Examples: penalties to be minimized:**
  - Set cardinality: $L_1$ norm
    $$\Omega^L(w) = \sum_{f=1}^{F} |w_f| = ||w||_1$$
  - Undirected graph cut: total variance/fused lasso
    $$\Omega^L(w) = \frac{1}{2} \sum \sum_{j,k=1}^{F} G_{j,k} |w_j - w_k|$$
  - Directed graph cut
    $$\Omega^L(w) = \sum_{j=1}^{F} \sum_{k=1}^{F} G_{j,k} \left[w_j - w_k\right]_+$$
  - Group absence: $L_1/L_\infty$ mixed norm
    $$\Omega^L(w) = \sum_i \max_{f \in Q_i} |w_f| = \sum_i ||w_{Q_i}||_\infty$$

# Big picture

- Example: two features: $x_a$ and $x_b$
- Any linear classifier is $y = \text{sign}(w_a x_a + w_b x_b)$
- What are the weights $w = (w_a, w_b)$?
  - To find out, we optimize Risk(w)+Penalty(w)




Risk(w)

We can't just add penalty on the set (of features) $\Omega(S)$ to risk over vectors (of feature weights).

What we want:
**Risk(w)+Penalty(w)**
Penalty(w)=$\Omega([w])$
tough to solve!

What we can:
**Risk(w)+Penalty(w)**
Penalty(w)=$\Omega^{-}(w)$
both convex, so often easy to solve

# Big picture

- We can solve problems of the form:
  - **Differentiable risk**
    **+ Lovasz extension of a submodular set function Ω**
  - Solution: e.g. proximal gradient descent

- Potential problem:
  - Not all classifiers have differentiable risk
  - Notable exception: Support Vector Machines

# Alternatives

- In literature, we can also see convex differentiable penalties:
  - **Risk + quadratic penalty**
  - These typically don't lead to sparse solutions (e.g. $L_2$ vs $L_1$ norm)
- **Examples: penalties to be minimized:**
  - $L_2$ norm (an extension of set cardinality, but not pointwise highest)

$$\Omega_2(w) = \sum_{f=1}^{F} w_f^2 = ||w||_2^2$$

  - Graph Laplacian penalty
    (a convex extension of graph cut, but not pointwise highest)

$$\Omega_2(w) = \frac{1}{2} \sum \sum_{j,k=1}^{F} G_{j,k}(w_j - w_k)^2 = w^T L_G w$$

  - Group Lasso: $L_1/L_2$ mixed norm (not an extension of *group absence*)

$$\Omega_2(w) = \sum_i \sqrt{\sum_{f \in Q_i} w_f^2} = \sum_i ||w_{Q_i}||_2$$