CPSC 540: Machine Learning

Group L1-Regularization, Structured Sparsity,
Projected-Gradient

Admin

- Room: Next week, we will be moving to CHEM B150.
- Assignment 1:
  - You can use 2 of your 3 late days to hand it in before Tuesday’s class.
- Assignment 2:
  - Due February 2^{nd}.
  - Start early!

Last Time: Gradient Descent Theory and Practice

- We discussed further properties of gradient descent:
  - "Strong-smoothness" weakened to "gradient is L-Lipschitz continuous".
    - And only along the line segments between x^t and x^{t+1}.
  - No need to know 'L':
    - Adaptive step-size, Armijo line-search, or exact step-size.
  - "Strong-convexity" is implied if we have $f(x) + \lambda||x||^2$ and 'f' is convex.
    - If 'f' is not convex, convergence rate only holds near solution.
- We overviewed methods with better performance:
  - Nesterov's accelerated-gradient method.
  - Approximations to Newton's method.

Last Time: L1-Regularization

- We considered regularization by the L1-norm:

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \; f(x) + \lambda \|x\|_1$$

  - Encourages solution x* to be sparse.
- Convex approach to regularization and pruning irrelevant features.
  - Not perfect, but very fast.
  - Could be used as filter, or to initialize NP-hard solver.
- Non-smooth, but "simple" regularizer allows special methods:
  - Proximal-gradient methods for general differentiable 'f' (today).
  - Coordinate optimization for some special cases of 'f'.

Why not just threshold ‘w’?

- Why not just compute least squares ‘w’ and threshold?
  - You can show some nice properties of this, but it does some silly things:
    - Let feature 1 be an irrelevant feature, and assume feature 2 is a copy of feature 1.
    - Without regularization, could have $w_1 = -w_2$ with both values arbitrarily large.
- Why not just compute L2-regularized ‘w’ and threshold?
  - Fixes the above problem, but still does weird things:
    - Let feature 1 is irrelevant and feature 2 is relevant.
    - Assume feature 3 is also relevant, and features 4:d are copies of feature 3.
    - For ‘d’ large enough, L2-regularization prefers irrelevant feature ‘1’ or relevant 3:d. (L1-regularization should pick at least one among 3:d for any ‘d’.)
- (I’m not saying L1-regularization doesn’t do weird things, too.)
- If features are orthogonal, thresholding and L1 are equivalent.
  - But feature selection is not interesting in this case.

Last Time: Coordinate Optimization

- For gradient descent we assume gradient is Lipschitz continuous:

$$\|\nabla f(x) - \nabla f(y)\| \leq L_f \|x - y\| \qquad \nabla^2 f(x) \preceq L_f I$$

- For coordinate optimization we assume coordinate-wise L-Lipschitz:

$$|\nabla_j f(x + \alpha e_j) - \nabla_j f(x)| \leq L|\alpha| \qquad \nabla^2_{ii} f(x) \leq L$$

- Note that neither of these is stronger:
  - If gradient is L_f-Lipschitz, then its coordinate-wise L_f-Lipschitz, so $L \leq L_f$.
  - If coordinate-wise L-Lipschitz, then gradient is dL-Lipschitz, so $L_f \leq dL$.
- Gradient descent requires $O((L_f/\mu)\log(1/\varepsilon))$ iterations.
- Coordinate optimization requires $O(d(L/\mu)\log(1/\varepsilon))$ iterations.
  - This is slower because $L_f \leq dL$.
  - But if iterations are 'd' times cheaper, this is faster because $L \leq L_f$.

Since $\nabla_j f$ is coordinate-wise L-Lipschitz continuous, we have:

$$f(y) \leq f(x) + \nabla_j f(x)(y-x)_j + \frac{L}{2}(y-x)_j^2$$ for any and any 'x' and 'y' that equal in all coordinates except 'j'!

As with gradient descent proof, let $x = x^t$ and $y = x^{t+1}$ and $j = j_t$:

$$f(x^{t+1}) \leq f(x^t) + \nabla_{j_t} f(x^t)(x^{t+1} - x^t)_{j_t} + \frac{L}{2}(x^{t+1} - x^t)_{j_t}^2$$

Now assume we take a step of $x^{t+1} = x^t - \frac{1}{L}\nabla_{j_t} f(x^t) e_{j_t}$ where we use the convention $e_j = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \leftarrow j$

This means $(x^{t+1} - x^t)_{j_t} = -\frac{1}{L}\nabla_{j_t} f(x^t)$ so we have:

$$\begin{aligned} f(x^{t+1}) &\leq f(x^t) + \nabla_{j_t} f(x^t)\left(-\frac{1}{L}\nabla_{j_t} f(x^t)\right) + \frac{L}{2}\left(-\frac{1}{L}\nabla_{j_t} f(x^t)\right)^2 \\ &= f(x^t) - \frac{1}{L}\left(\nabla_{j_t} f(x^t)\right)^2 + \frac{1}{2L}\left(\nabla_{j_t} f(x^t)\right)^2 \\ &= f(x^t) - \frac{1}{2L}\left(\nabla_{j_t} f(x^t)\right)^2 \end{aligned}$$

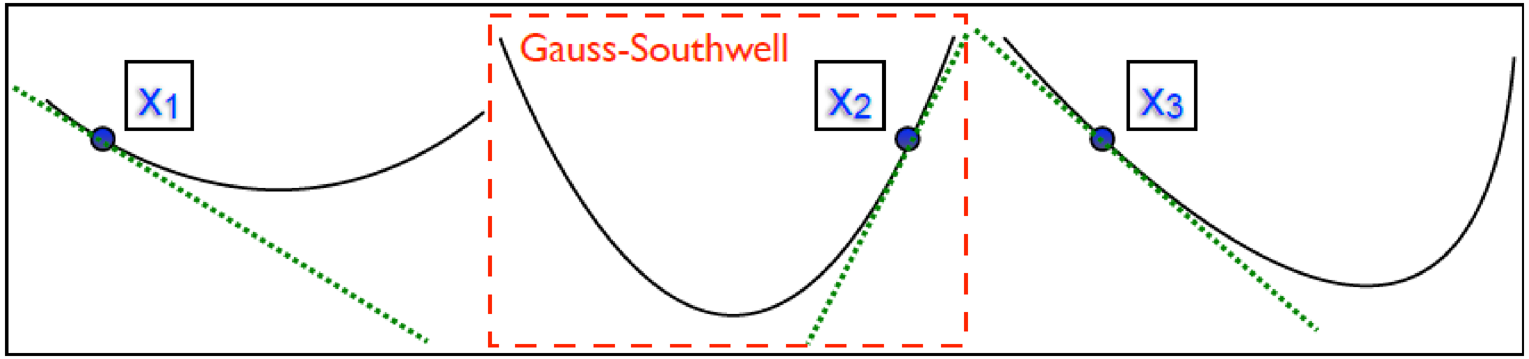
This is the same bound we got for the gradient method, except now we are taking into account what happens if you only update one variable.

Gauss-Southwell Selection Rule

- Our bound for any coordinate: $f(x^{t+1}) \leq f(x^t) - \frac{1}{2L} |\nabla_{j_t} f(x^t)|^2$
- The "best" coordinate to update is:

$$j_t \in \underset{j}{\operatorname{argmax}} \{ |\nabla_j f(x^t)| \}$$

  – Called the 'Gauss-Southwell' or greedy rule.



  – You can derive a convergence rate by using that $|\nabla_{j_t} f(x^t)|^2 = \|\nabla f(x^t)\|_\infty^2$
  – Typically, this can't be implemented 'd' times faster than gradient method.

Random Selection Rule

- Our bound for any coordinate: $f(x^{t+1}) \leq f(x^t) - \frac{1}{2L}|\nabla_{j_t} f(x^t)|^2$
- Let's consider random selection of each 'j' with probability 1/d:

$$E[f(x^{t+1})] \leq E\left[f(x^t) - \frac{1}{2L}|\nabla_{j_t} f(x^t)|^2\right] \quad \text{(expectation with respect to } j_t\text{)}$$

$$E[\alpha X + \beta f(Y)] = \alpha E[X] + \beta E[f(Y)]$$

$$= E[f(x^t)] - \frac{1}{2L} E[|\nabla_{j_t} f(x^t)|^2] \quad \text{(expectation is linear)}$$

$$= f(x^t) - \frac{1}{2L}\sum_{j=1}^{d} p(j)|\nabla_j f(x^t)|^2 \quad \text{(definition of expectation)}$$

$$= f(x^t) - \frac{1}{2L}\sum_{j=1}^{d} \frac{1}{d}|\nabla_{j_t} f(x^t)|^2 \quad \left(\text{using } p(j) = \frac{1}{d}\right)$$

$$= f(x^t) - \frac{1}{2Ld}\sum_{j=1}^{d}|\nabla_{j_t} f(x^t)|^2$$

$$= f(x^t) - \frac{1}{2Ld}\|\nabla f(x^t)\|^2 \quad \left(\|v\|^2 = \sum_{j=1}^{d}|v_j|^2\right)$$

Analysis of Coordinate Optimization

- If 'f' is μ-strongly-convex, then we get a linear convergence rate:

$$E[f(x^{t+1}) - f(x^*)] \leq f(x^t) - f(x^*) - \frac{1}{2Ld}\|\nabla f(x^t)\|^2$$

(subtract $f(x^*)$ from both sides)

$$\leq f(x^t) - f(x^*) - \frac{\mu}{Ld}[f(x^t) - f(x^*)]$$

$(\leftarrow \|\nabla f(x^t)\|^2 \leq 2\mu(f(x^t) - f(x^0)))$

from strong-convexity

$$= \left(1 - \frac{\mu}{Ld}\right)[f(x^t) - f(x^*)]$$

Analysis of Coordinate Optimization

- If ‘f’ is μ-strongly-convex, then we get a linear convergence rate:

$$E[f(x^{t+1}) - f(x^*)] \leq f(x^t) - f(x^*) - \frac{1}{2Ld}\|\nabla f(x^t)\|^2$$

(subtract $f(x^*)$ from both sides)

$$\leq f(x^t) - f(x^*) - \frac{\mu}{Ld}[f(x^t) - f(x^*)]$$

($-\|\nabla f(x^t)\|^2 \leq 2\mu(f(x^t) - f(x^0))$)
from strong-convexity

$$= \left(1 - \frac{\mu}{Ld}\right)[f(x^t) - f(x^*)]$$

$$E[E[f(x^{t+1}) - f(x^*)]] = E\left[\left(1 - \frac{\mu}{Ld}\right)[f(x^t) - f(x^*)]\right]$$

(expectation with respect to j_{t-1})

$$E[f(x^{t+1}) - f(x^*)] = \left(1 - \frac{\mu}{Ld}\right)E[f(x^t) - f(x^*)]$$

iterated expectation
($E_Y[E_{X|Y}[X|Y]] = E[X]$)

Analysis of Coordinate Optimization

- If 'f' is μ-strongly-convex, then we get a linear convergence rate:

$$E[f(x^{t+1}) - f(x^*)] \leq f(x^t) - f(x^*) - \frac{1}{2Ld}\|\nabla f(x^t)\|^2$$

(subtract $f(x^*)$ from both sides)

$$\leq f(x^t) - f(x^*) - \frac{\mu}{Ld}[f(x^t) - f(x^*)]$$

$(-\|\nabla f(x^t)\|^2 \leq 2\mu(f(x^t) - f(x^0)))$ from strong-convexity

$$= \underline{\left(1 - \frac{\mu}{Ld}\right)[f(x^t) - f(x^*)]}$$

$$E[E[f(x^{t+1}) - f(x^*)]] = E\left[\left(1 - \frac{\mu}{Ld}\right)[f(x^t) - f(x^*)]\right]$$

(expectation with respect to j_{t-1})

iterated expectation ($E_Y[E_{X|Y}[X|Y]] = E[X]$)

$$E[f(x^{t+1}) - f(x^*)] = \left(1 - \frac{\mu}{Ld}\right)E[f(x^t) - f(x^*)]$$

apply recursively

$$\leq \left(1 - \frac{\mu}{Ld}\right)^2 [f(x^{t-1}) - f(x^*)]$$

$$\vdots$$

Finally giving $\underline{E[f(x^k) - f(x^*)] \leq \left(1 - \frac{\mu}{Ld}\right)^t [f(x^0) - f(x^*)]}$

This implies we need $O(d\frac{L}{\mu}\log(\frac{1}{\epsilon}))$ iterations until $E[f(x^k) - f(x^*)] \leq \epsilon$

Lipschitz Sampling and Gauss-Southwell-Lipschitz

- You can go even faster if you have an L_j for each coordinate:

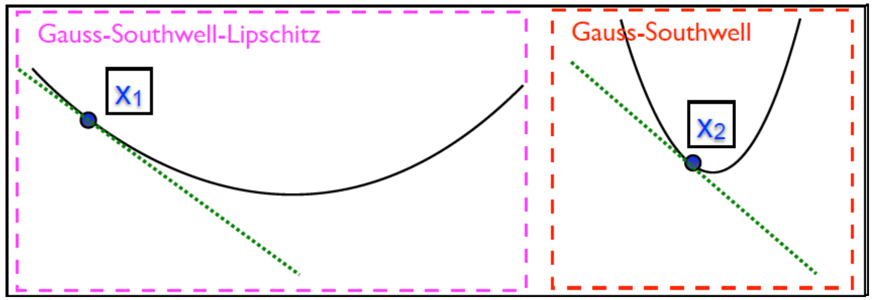
$$|\nabla_j f(x+\alpha e_j) - \nabla_j f(x)| \leq L_j |\alpha|$$

- If you sample j_t proportional to L_j, you can get a rate of:

$$\mathbb{E}[f(x^t) - f(x^*)] \leq \left(1 - \frac{\mu}{\bar{L} d}\right)^t [f(x^0) - f(x^*)] \quad \text{where } \bar{L} = \frac{1}{d}\sum_{j=1}^{d} L_j$$

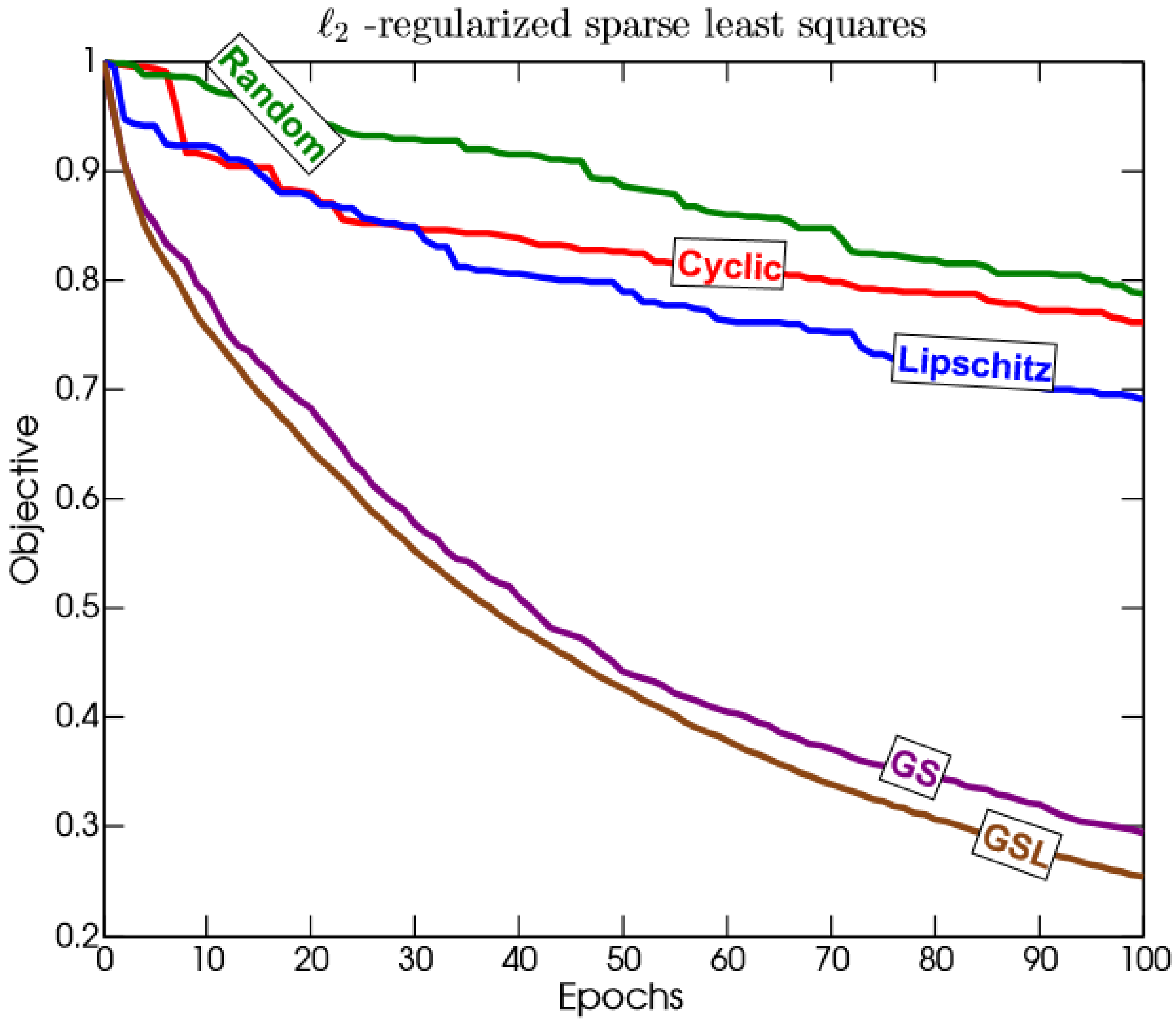
  – Depends on average L_j instead of maximum L_j.

- The Gauss-Southwell-Lipschitz rule: $j_t \in \underset{j}{\operatorname{argmax}} \left\{ \frac{|\nabla_j f(x^t)|^2}{L_j} \right\}$



  – Even faster, and optimal for quadratic functions.

Comparison of Coordinate Selection Rules



Coordinate Optimization for Non-Smooth Objectives

- Consider an optimization problem of the form:

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \underbrace{f(x)}_{\text{smooth}} + \underbrace{\sum_{i=1}^{d} g_i(x_i)}_{\text{"separable"}}$$

- Assume:
  – 'f' is coordinate-wise L-Lipschitz continuous and μ-strongly convex.
  – 'h_i' are general convex functions (could be non-smooth).
  – You do exact coordinate optimization.
- For example, L1-regularized least squares: $\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \overbrace{\frac{1}{2}\|Xw - y\|^2}^{f(x)} + \overbrace{\lambda \sum_{j=1}^{d} |x_j|}^{g_j(x_j) = \lambda|x_j|}$
- Linear convergence rate still holds (proof more complicated):

$$E[f(x^t) - f(x^*)] \leq \left(1 - \frac{\mu}{Ld}\right)^t [f(x^0) - f(x^*)]$$

- We can solve these non-smooth problems much faster than O(1/ε).

Motivation: Group Sparsity

- More general case: we want sparsity in 'groups' of variables.
  - E.g., we represent categorical/numeric variables as set of binary variables,

City	Age
Vancouver	22
Burnaby	35
Vancouver	28

Vancouver	Burnaby	Surrey	$\text{Age} \le 20$	$20 < \text{Age} \le 30$	$\text{Age} > 30$
1	0	0	0	1	0
0	1	0	0	0	1
1	0	0	0	1	0

and we want to select original variables ("city" and "age")

- We can address this problem with group L1-regularization:
  - 'Group' is all binary variables that came from same original variable.

Group L1-Regularization

Remember that for <u>any</u> norm $\|\cdot\|_p$ we have $\|x\|_p = 0$ iff $x = 0$.

- Minimizing a function 'f' with group L1-regularization:

$$\underset{x \in \mathbb{R}^d}{\text{argmin}}\ f(x) + \lambda \sum_{g \in G} \|x_g\|_p$$

some norm

individual group

set of groups

- Think of this as L1-regularization of the group norms:
  - Encourages group norms to be exactly zero: all group variables become 0.
  - Sometimes written as 'mixed' norm: $\|x\|_{1,p} = \sum_{g \in G} \|x_g\|_p$
- Typical choices of norm:

L_2-norm: $\|x_g\|_2 = \sqrt{\sum_{j \in g} x_j^2}$

L_∞-norm: $\|x_g\|_\infty = \max_{j \in g} |x_j|$

L_1-norm does <u>not</u> work:

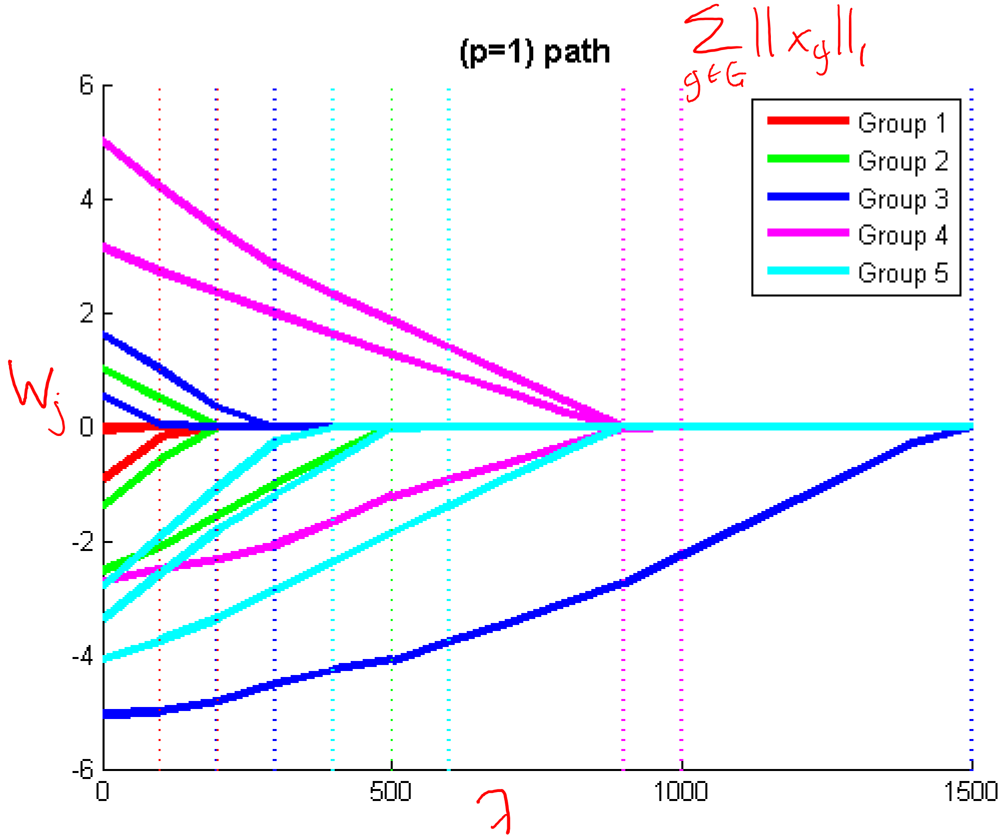
no grouping effect

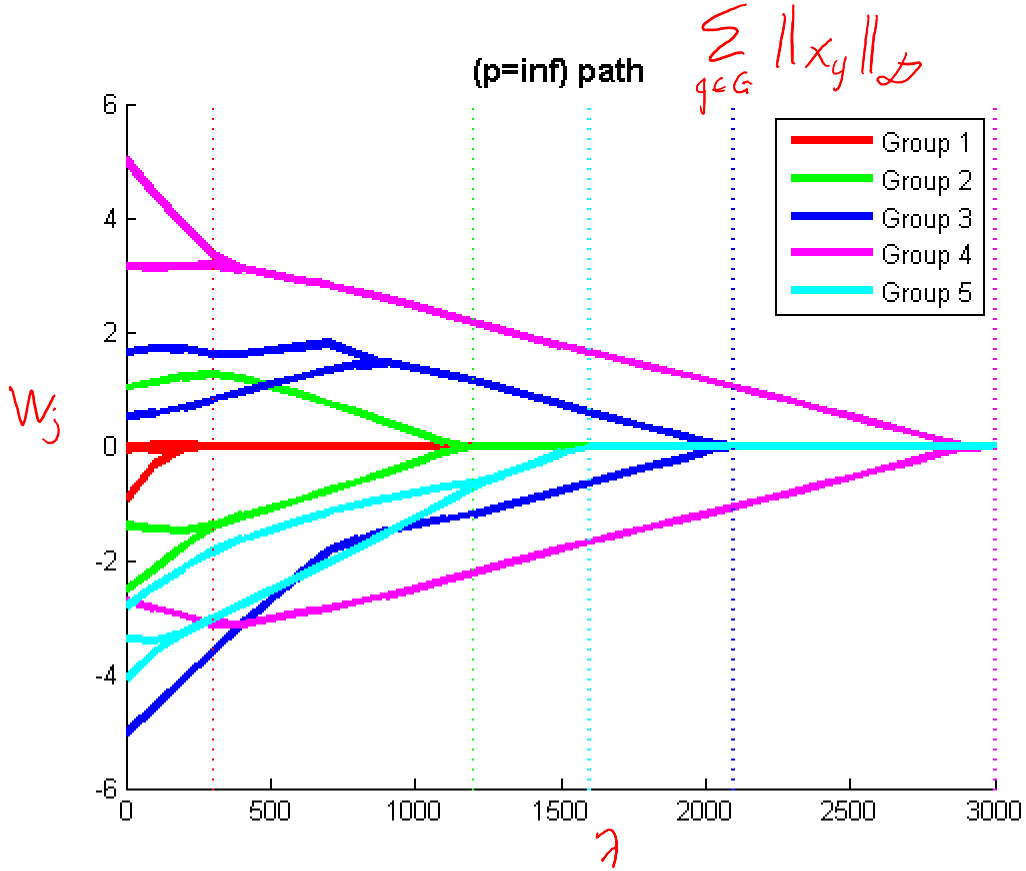
$$\|x\|_{1,1} = \sum_{g \in G} \|x_g\|_1 = \sum_{g \in G} \sum_{j \in g} |x_j| = \sum_{j=1}^{d} |x_j| = \|x\|_1$$

Group L1-Regularization

- Minimizing a function 'f' with group L1-regularization:

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} f(x) + \lambda \sum_{g \in G} \|x_g\|_p$$

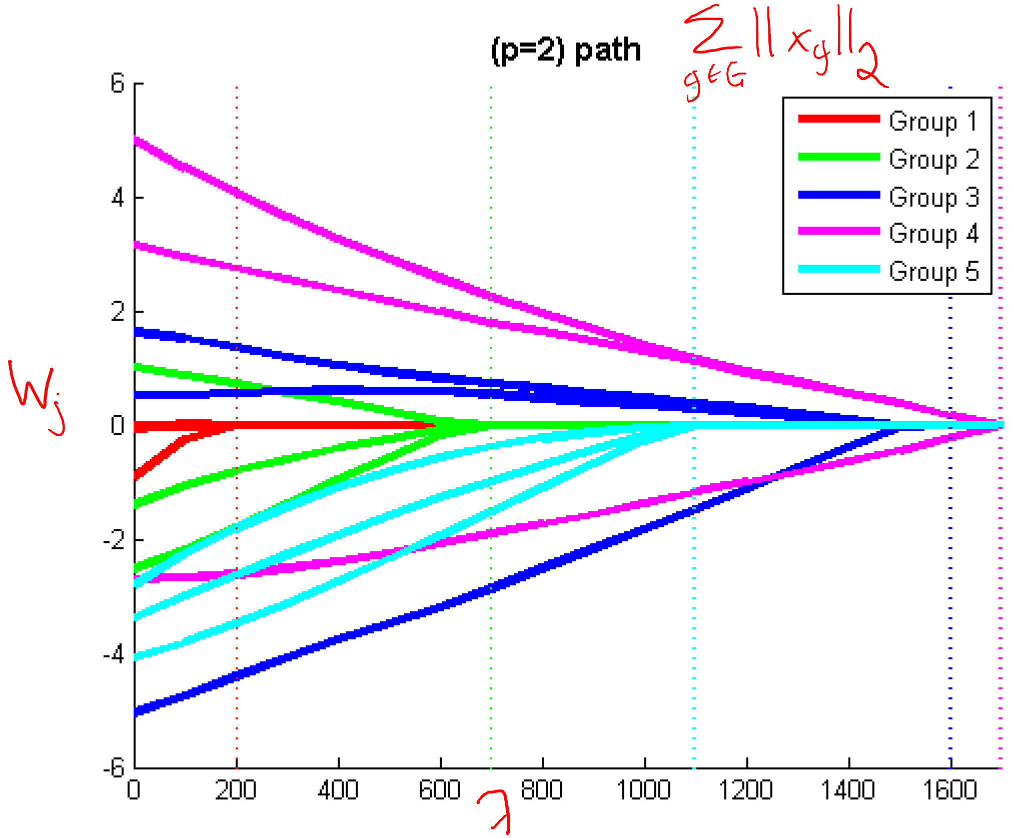


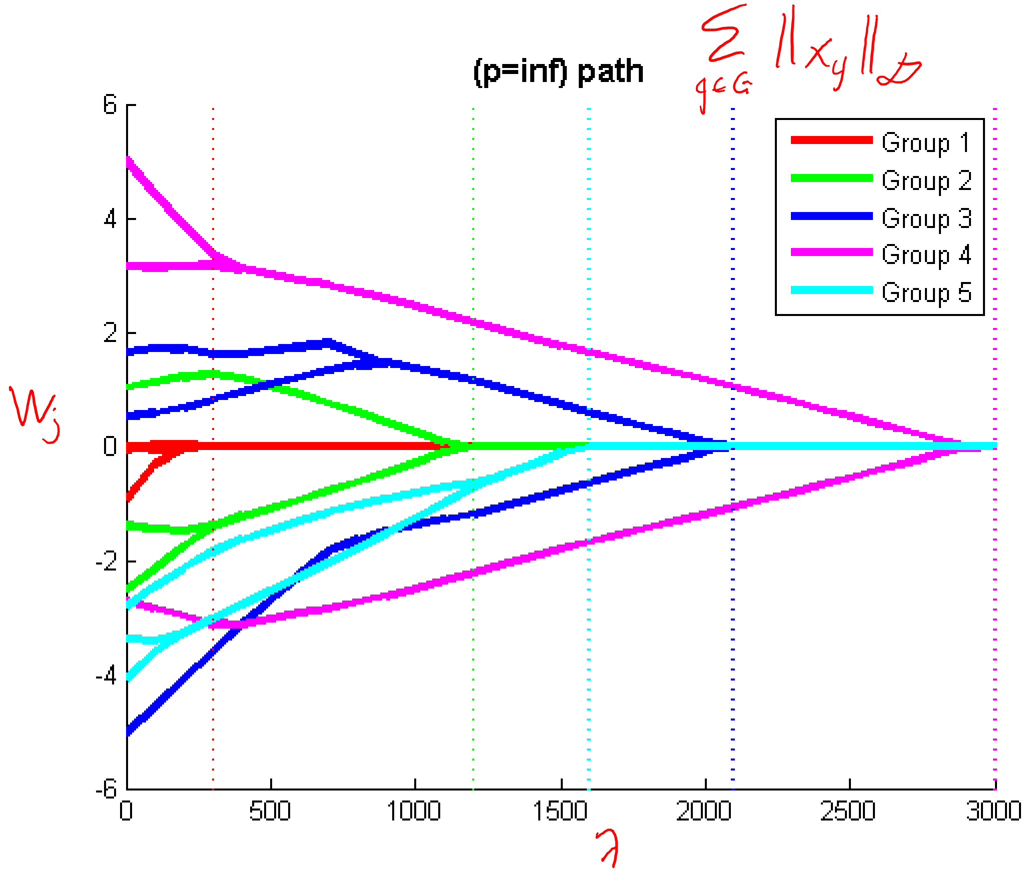


Group L1-Regularization

- Minimizing a function 'f' with group L1-regularization:

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} f(x) + \lambda \sum_{g \in G} \|x_g\|_p$$

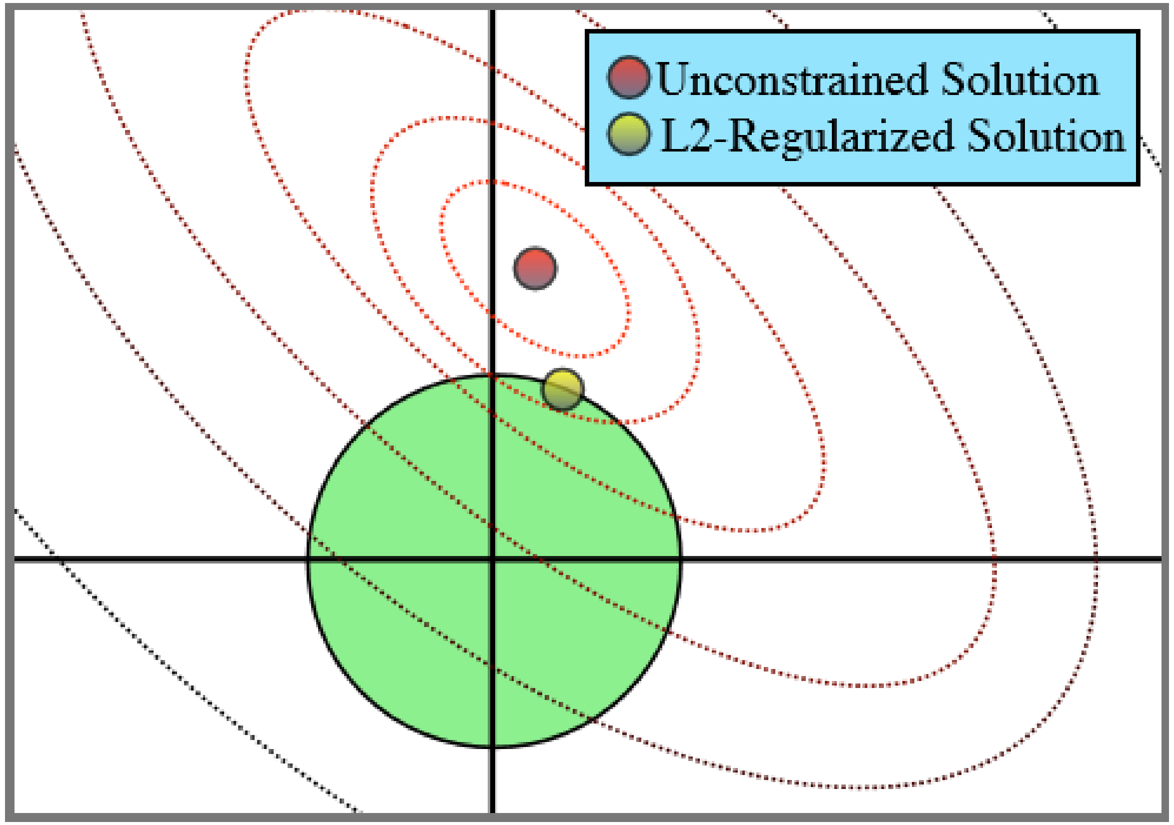


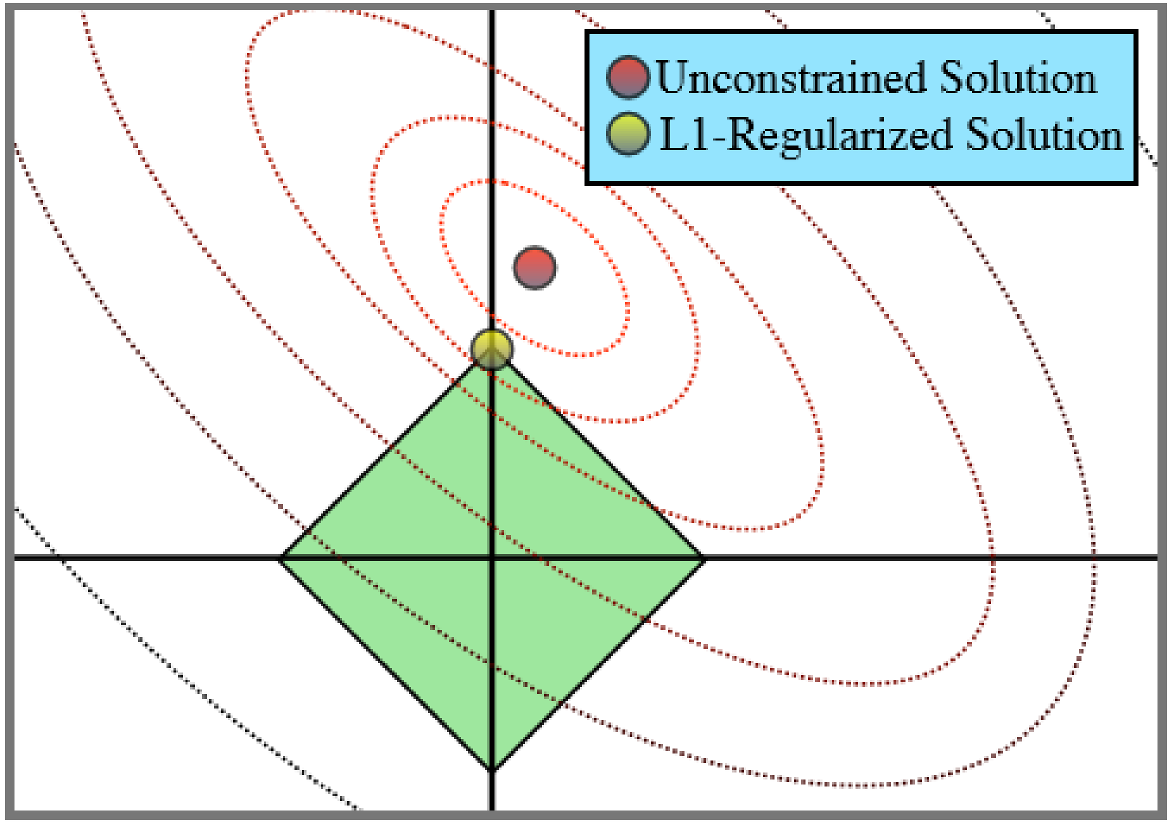


Group L1-Regularization

- Minimizing a function ‘f’ with group L1-regularization:

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} f(x) + \lambda \sum_{g \in G} \|x_g\|_p$$



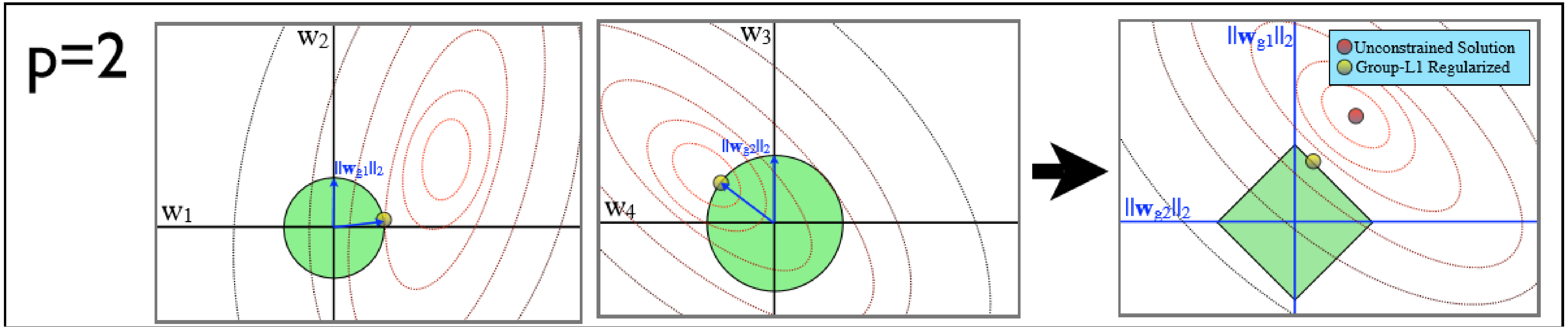


Group L1-Regularization

- Minimizing a function ‘f’ with group L1-regularization:

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} f(x) + \lambda \sum_{g \in G} \|x_g\|_p$$

$$G = \{\underbrace{\{1,2\}}_{g_1}, \underbrace{\{3,4\}}_{g_2}\}$$

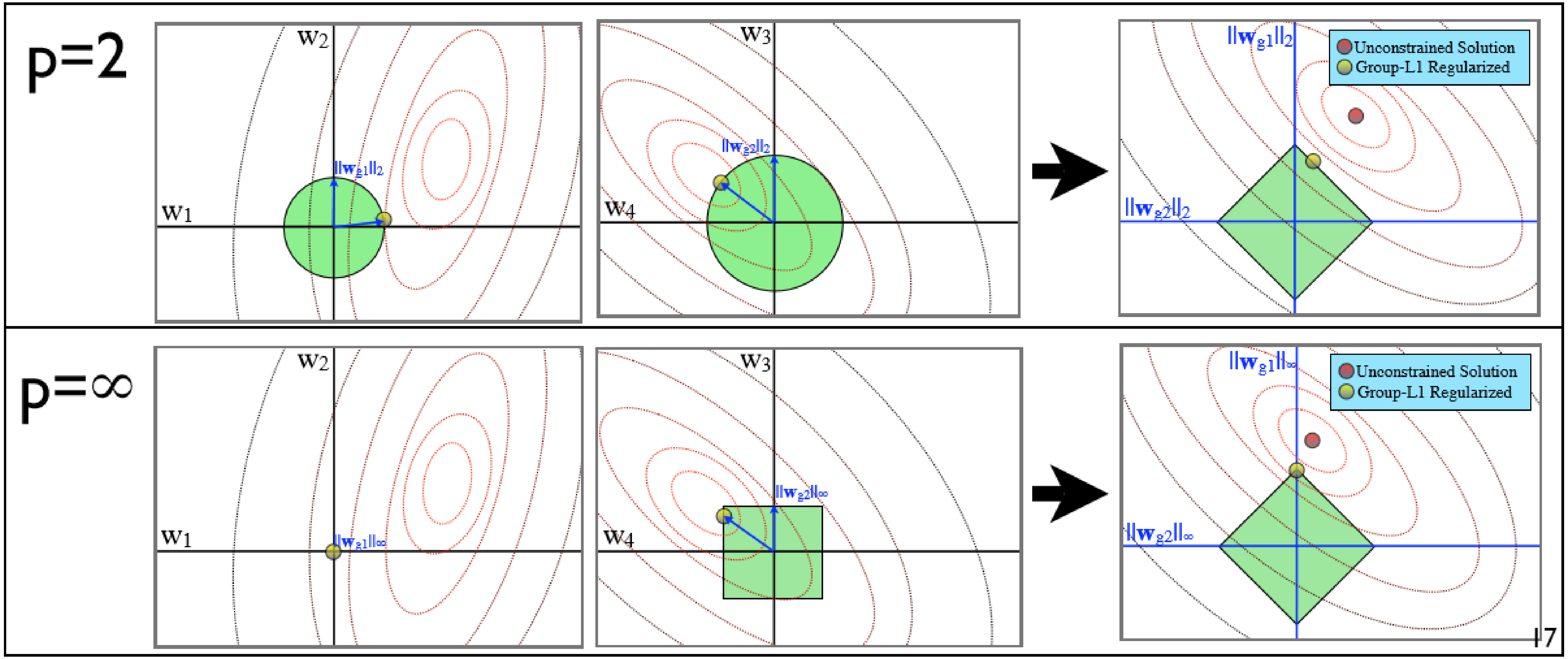


Group L1-Regularization

- Minimizing a function ‘f’ with group L1-regularization:

$$\underset{x \in \mathbb{R}^d}{\arg\min}\ f(x) + \lambda \sum_{g \in G} \|x_g\|_p$$

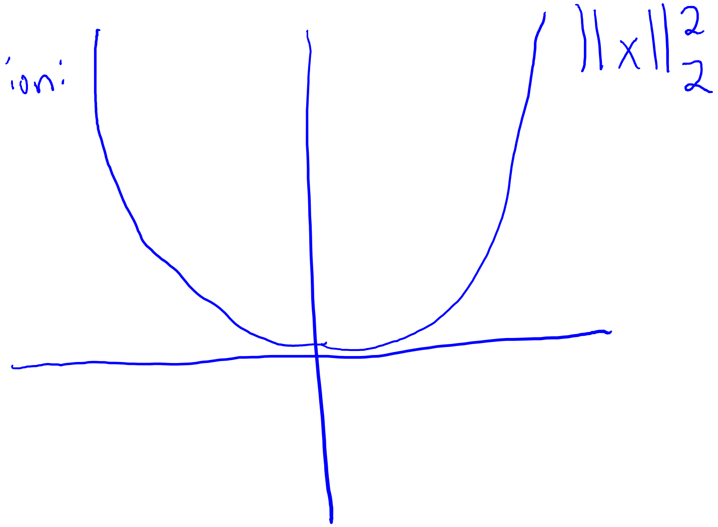
$$G = \{ \underbrace{\{1,2\}}_{g_1}, \underbrace{\{3,4\}}_{g_2} \}$$

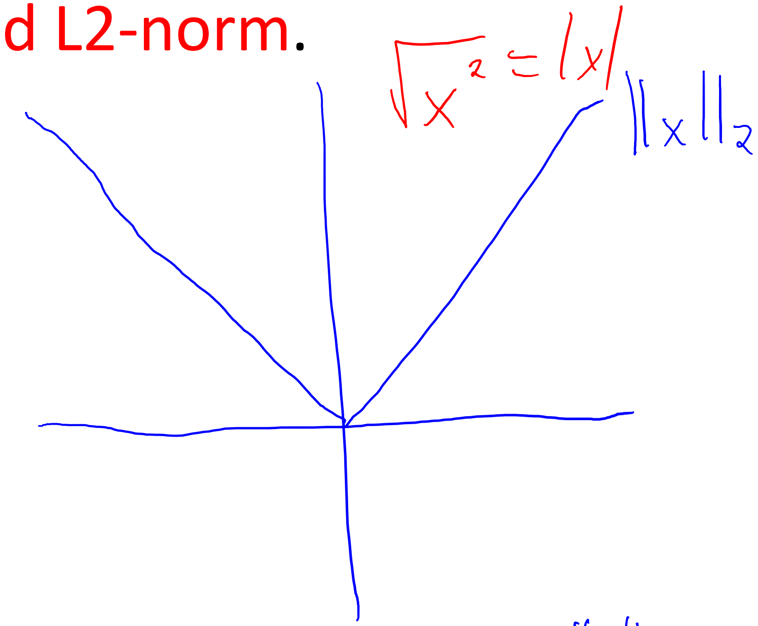


Sparsity from the L2-norm?

- Didn't we say that sparsity comes from L1-norm and not L2-norm?
  - Yes, but we were using squared L2-norm.

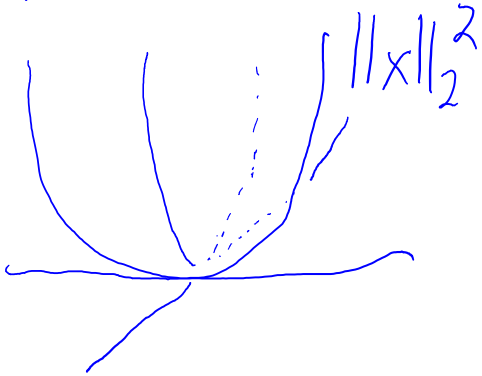
One-dimension:

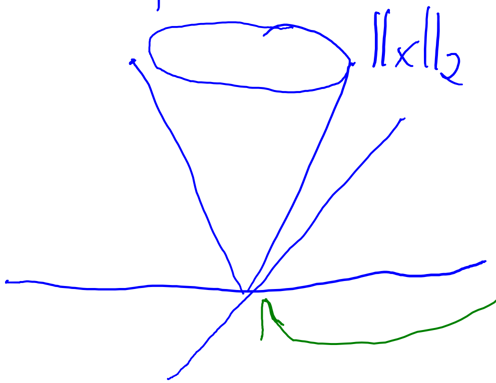




If you regularize by $\lambda\|w\|_2$ then for some finite λ it sets all variables to zero.

Two-dimensions





If you regularize by $\lambda\|w\|_1^2$ there may be no finite λ that sets all variables to 0.

non-differentialbe when $\|x\|_2 = 0$.

Other Applications of Group Sparsity

- Recall that multi-class logistic regression uses:

$$\hat{y}_i = \underset{c}{\operatorname{argmax}}\{w_c^T x_i\}$$

- We can write our parameters as a matrix:

$$W = \begin{bmatrix} | & | & | & & | \\ w_1 & w_2 & w_3 & \cdots & w_k \\ | & | & | & & | \end{bmatrix}$$

all parameters in this row correspond to same original feature.

- To 'select' a feature 'j', we need 'w_{cj} = 0' for all 'j'.
  - If any element of row is non-zero, we still use feature.
  - We need a row of zeroes.

Other Applications of Group Sparsity

- In multiple regression we have multiple targets y_{ic}:

$$\hat{y}_{i1} = w_1^T x_i$$
$$\hat{y}_{i2} = w_2^T x_i$$
$$\vdots$$
$$\hat{y}_{ik} = w_k^T x_i$$

- We can write our parameters as a matrix:

$$W = \begin{bmatrix} | & | & | & & | \\ w_1 & w_2 & w_3 & \cdots & w_k \\ | & | & | & & | \end{bmatrix}$$

← all parameters in this row correspond to same original feature.

- To 'select' a feature 'j', we need 'w_{cj} = 0' for all 'j'.
- Same pattern also arises in multi-label and multi-task classification.

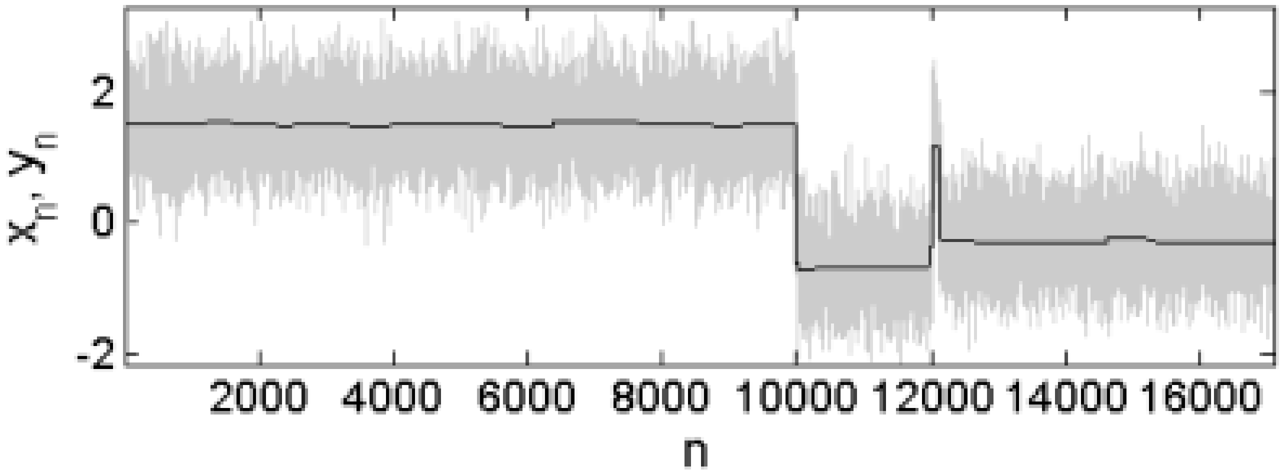
(pause)

Structured Sparsity

- There are many other patterns that regularization can encourage:
  - Total-variation regularization ('fused' LASSO):

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \; f(x) + \lambda \sum_{j=1}^{d-1} |x_j - x_{j+1}|$$

  - Encourages consecutive parameters to have same value.
  - Often used for time-series data:
  - 2D version is popular for image denoising.
  - Can also define for general graphs between variables.



http://statweb.stanford.edu/~bjk/regreg/examples/fusedlassoapprox.html

Structured Sparsity

- There are many other patterns that regularization can encourage:
  - Nuclear-norm regularization:

$$\underset{X \in \mathbb{R}^{d \times k}}{\operatorname{argmin}} \; f(X) + \lambda \underbrace{\|X\|_*}_{\text{sum of singular values}}$$

  - Encourages parameter matrix to have low-rank representation.
  - E.g., consider multi-label classification with huge number of labels.

$$W = \begin{bmatrix} | & | & & | \\ w_1 & w_2 & \cdots & w_k \\ | & | & & | \end{bmatrix} = UV^T \quad \text{with} \quad U = \underbrace{\begin{bmatrix} \; \end{bmatrix}}_{\text{small}} \Big\} d \quad \text{and} \quad V^T = \underbrace{\begin{bmatrix} & & \end{bmatrix}}_{k} \Big\} \text{small}$$

Structured Sparsity

- There are many other patterns that regularization can encourage:
  - Overlapping Group L1-Regularization:

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \; f(x) + \sum_{g \in G} \lambda_g \|w_g\|_p$$

  - Same as group L1-regularization, but groups overlap.
  - Can be used to encourage any intersection-closed sparsity pattern.

"Intersection-closed" set 'A' means if you take finite number of set $a_i \in A$ then $\bigcap_i a_i \in A$

(There is also a variant that does unions)

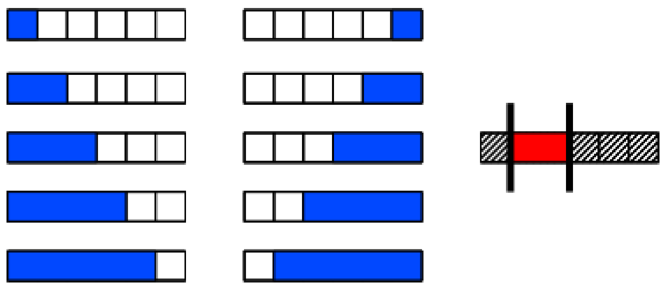


Fig 3: (Left) The set of blue groups to penalize in order to select contiguous patterns in a sequence. (Right) In red, an example of such a nonzero pattern with its corresponding zero pattern (hatched area).

http://arxiv.org/pdf/1109.2397v2.pdf

Structured Sparsity

- There are many other patterns that regularization can encourage:
  - Overlapping Group L1-Regularization:

$$\underset{x \in \mathbb{R}^d}{\text{argmin}} \; f(x) + \sum_{g \in G} \lambda_g \|w_g\|_p$$

  - How does this work?
    - Consider the case of two groups {1} and {1,2}:

$$\underset{x \in \mathbb{R}^d}{\text{argmin}} \; f(x) + \lambda\left(|x_1| + \sqrt{x_1^2 + x_2^2}\right)$$

If $x_1 \neq 0$ the objective is smooth with respect to x_2 and so it moves away from zero.

If $x_2 \neq 0$ the objective is still non-smooth with respect to x_1 so it's encouraged to be zero.

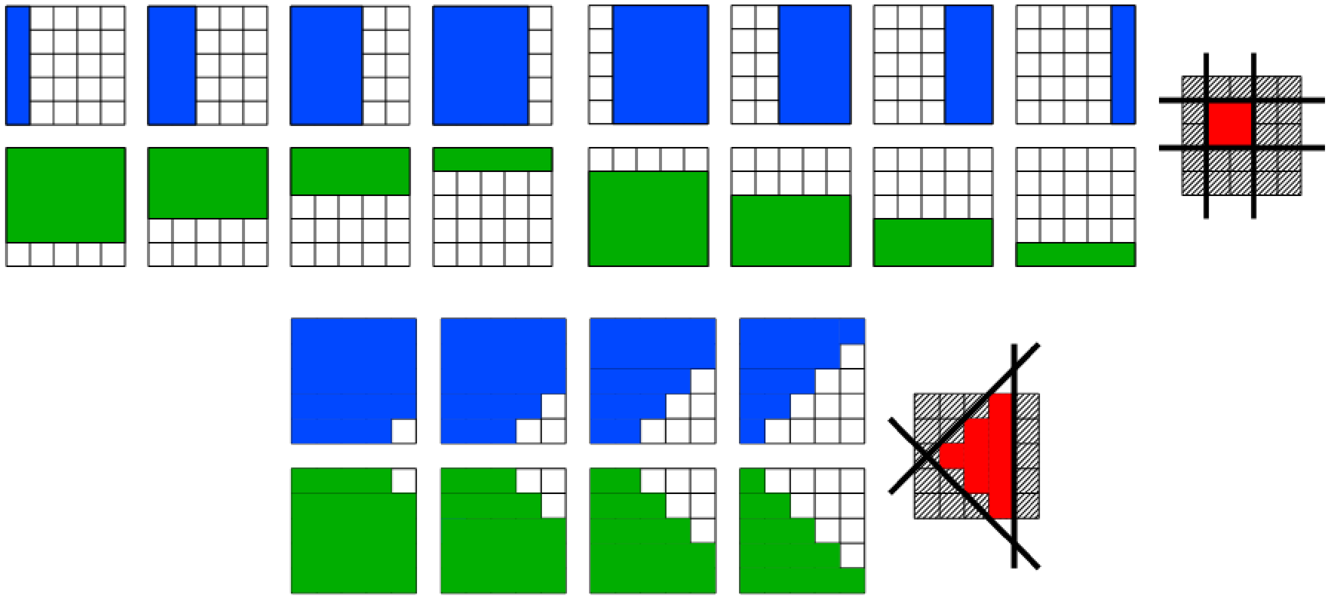
So possible non-zero patterns are $\{2\}$ or $\{1,2\}$.

Structured Sparsity

- There are many other patterns that regularization can encourage:
  - Overlapping Group L1-Regularization:

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \; f(x) + \sum_{g \in G} \lambda_g \|w_g\|_p$$

  - Enforcing convex non-zero patterns:



Structured Sparsity

- There are many other patterns that regularization can encourage:
  - Overlapping Group L1-Regularization:

$$\underset{x \in \mathbb{R}^d}{\arg\min} \; f(x) + \sum_{g \in G} \lambda_g \|w_g\|_p$$

  - Enforcing convex non-zero patterns:

http://arxiv.org/pdf/1109.2397v2.pdf

Structured Sparsity

- There are many other patterns that regularization can encourage:
  - Overlapping Group L1-Regularization:

$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \; f(x) + \sum_{g \in G} \lambda_g \|w_g\|_p$$

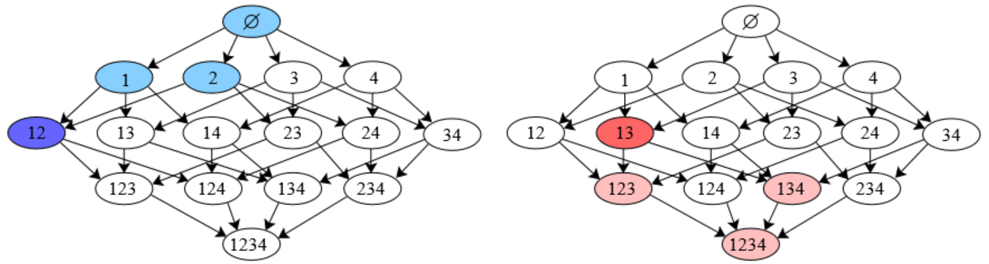


Fig 9: Power set of the set $\{1, \ldots, 4\}$: in blue, an authorized set of selected subsets. In red, an example of a group used within the norm (a subset and all of its descendants in the DAG).

  - Enforcing a hierarchy:

$$\hat{y}_i = w_0 + w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} + w_{12} x_{i1} x_{i2} + w_{13} x_{i1} x_{i3} + w_{23} x_{i2} x_{i3} + w_{123} x_{i1} x_{i2} x_{i3}$$

    - We only allow w_S non-zero is $w_{S'}$ is non-zero for all subsets S' of S.
    - E.g., we only consider $w_{123} \neq 0$ if we have $w_{12} \neq 0$, $w_{13} \neq 0$, and $w_{23} \neq 0$.
    - For certain bases, you can solve this problem in polynomial time.

http://arxiv.org/pdf/1109.2397v2.pdf

Fitting Models with Structured Sparsity

- These objectives typically have the form:

$$\underset{x \in \mathbb{R}^d}{\mathrm{argmin}} \; \underbrace{f(x)}_{\text{smooth}} + \underbrace{r(x)}_{\text{non-smooth}}$$

- It's the non-differentiable regularizer that leads to the sparsity.
- We can't always apply coordinate descent:
  - 'f' might not allow cheap updates.
  - 'r' might not be separable.
- But general non-smooth methods have slow O(1/ε) rate.
- Are there faster methods for the above structure?

Converting to Constrained Optimization

- Re-write non-smooth problem as constrained problem.
- The problem

$$\min_x f(x) + \lambda \|x\|_1,$$

is equivalent to the problem:

$$\min_{x^+ \geq 0, x^- \geq 0} \quad f(x^+ - x^-) + \lambda \sum_i (x_i^+ + x_i^-),$$

nice because only constraints are that variables are non-negative

or the problems

$$\underbrace{\min_{-y \leq x \leq y} \quad f(x) + \lambda \sum_i y_i}_{\text{comes from usual 'max' trick}}, \qquad \min_{\|x\|_1 \leq \gamma} \quad f(x) + \lambda\gamma$$

- These are a smooth objectives with 'simple' constraints.

$$\min_{x \in C} f(x).$$

Optimization with Simple Constraints

- Recall: gradient descent minimizes quadratic approximation:

$$x^{t+1} = \underset{y}{\operatorname{argmin}} \left\{ f(x^t) + \nabla f(x^t)^T (y - x^t) + \frac{1}{2\alpha_t} \|y - x^t\|^2 \right\}.$$

- Consider minimizing subject to simple constraints:

where constraints are satisfied.

$$x^{t+1} = \underset{y \in \mathcal{C}}{\operatorname{argmin}} \left\{ f(x^t) + \nabla f(x^t)^T (y - x^t) + \frac{1}{2\alpha_t} \|y - x^t\|^2 \right\}.$$

minimize the same bound but restricted to the "feasible" set.

- We can re-write this as: $x^{t+1} = \underset{y \in C}{\operatorname{argmin}} \left\{ \alpha_t \nabla f(x^t)^T (y - x^t) + \frac{1}{2}\|y - x^t\|^2 \right\}$ (drop constant $f(x^t)$, multiply by α_t)

(add constant $\frac{\alpha_t^2}{2}\|\nabla f(x^t)\|^2$) $= \underset{y \in C}{\operatorname{argmin}} \left\{ \frac{\alpha_t^2}{2}\|\nabla f(x^t)\|^2 + \alpha_t \nabla f(x^t)^T (y - x^t) + \frac{1}{2}\|y - x^t\|^2 \right\}$

"complete the square":

$\frac{1}{2}\|a+b\|^2 = \frac{1}{2}(a+b)^T(a+b)$

$= \frac{1}{2}(a^Ta + a^Tb + b^Ta + b^Tb)$

$= \frac{1}{2}\|a\|^2 + a^Tb + \frac{1}{2}\|b\|^2$

$= \underset{y \in C}{\operatorname{argmin}} \left\{ \|(y - x^t) + \alpha_t \nabla f(x^t)\|^2 \right\}$

$= \underset{y \in C}{\operatorname{argmin}} \left\{ \|y - (x^t - \alpha_t \nabla f(x^t))\|^2 \right\}$

Projected-Gradient

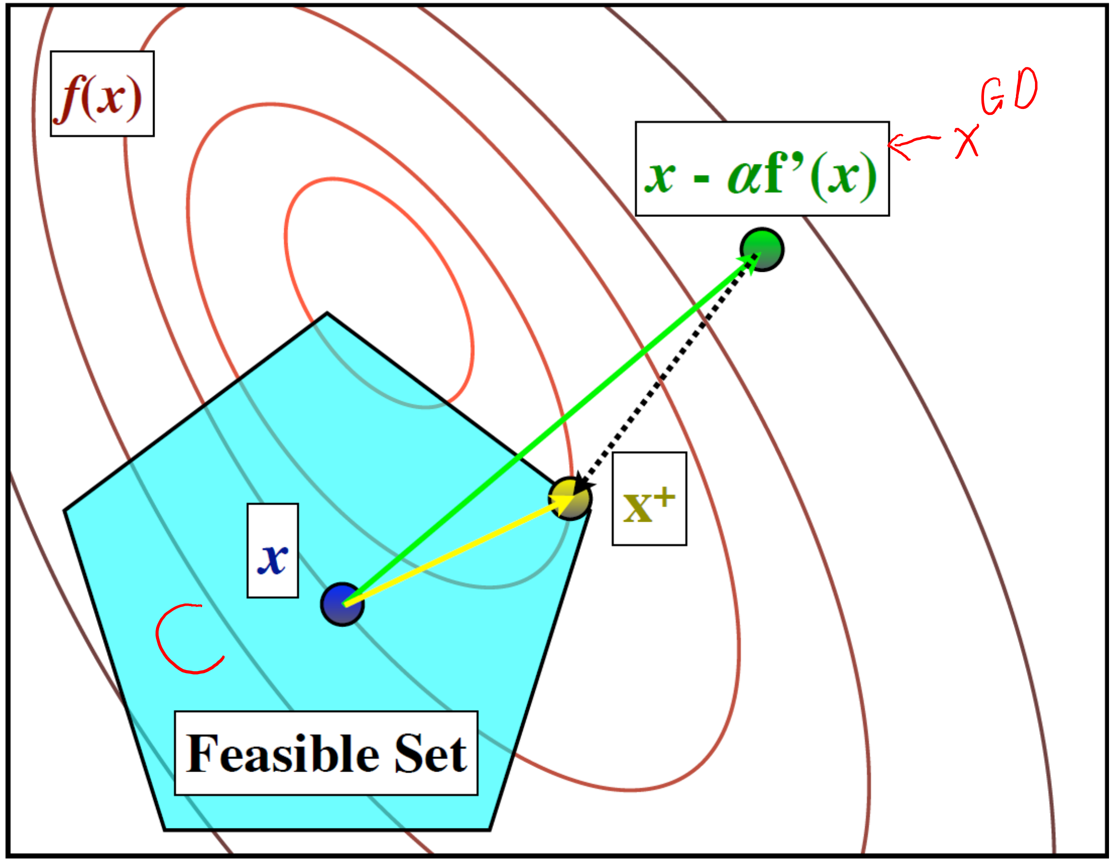
- This is called projected-gradient:

$$x_t^{GD} = x^t - \alpha_t \nabla f(x^t),$$

E.g., $\alpha_t = \frac{1}{L}$

$$x^{t+1} = \underset{y \in \mathcal{C}}{\text{argmin}} \left\{ \|y - x_t^{GD}\| \right\},$$

A set is 'simple' if we can efficiently compute projection.



Projection Onto Simple Sets

Projections onto simple sets:

- $\arg\min_{y \geq 0} \|y - x\| = \max\{x, 0\}$

Projection Onto Simple Sets

Projections onto simple sets:

- $\arg\min_{y \geq 0} \|y - x\| = \max\{x, 0\}$
- $\arg\min_{l \leq y \leq u} \|y - x\| = \max\{l, \min\{x, u\}\}$
- $\arg\min_{a^T y = b} \|y - x\| = x + (b - a^T x)a/\|a\|^2.$
- $\arg\min_{a^T y \geq b} \|y - x\| = \begin{cases} x & a^T x \geq b \\ x + (b - a^T x)a/\|a\|^2 & a^T x < b \end{cases}$

Could have <u>small</u> number of linear constraints

Projection Onto Simple Sets

Projections onto simple sets:

- $\arg\min_{y \geq 0} \|y - x\| = \max\{x, 0\}$
- $\arg\min_{l \leq y \leq u} \|y - x\| = \max\{l, \min\{x, u\}\}$
- $\arg\min_{a^T y = b} \|y - x\| = x + (b - a^T x)a/\|a\|^2.$
- $\arg\min_{a^T y \geq b} \|y - x\| = \begin{cases} x & a^T x \geq b \\ x + (b - a^T x)a/\|a\|^2 & a^T x < b \end{cases}$
- $\arg\min_{\|y\| \leq \tau} \|y - x\| = \tau x/\|x\|.$
- Linear-time algorithm for ℓ_1-norm $\|y\|_1 \leq \tau$.
- Linear-time algorithm for probability simplex $y \geq 0, \sum y = 1.$
- Intersection of simple sets: Dykstra's algorithm.

Could have small number of linear constraints

Possible for many other norm balls and cones

Discussion of Projected-Gradient

- Convergence rates are the same for projected versions:

f convex and non-smooth $\quad O(1/\varepsilon^2)$

f convex and ∇f Lipschitz $\quad O(1/\varepsilon)$

f strongly-convex and non-smooth $\quad O(1/\varepsilon)$

f strongly-convex and ∇f Lipschitz $\quad O(\log(1/\varepsilon))$

- Having 'simple' constraints is as easy as having no constraints.
- We won't prove these, but some simple properties proofs use:

Projection is a <u>contraction</u>

$$\|P_c(x) - P_c(y)\| \leq \|x - y\|$$

(moves x and y <u>closer</u>)

Solution x^* is a fixed point:

$$x^* = P_c[x^* - \alpha \nabla f(x^*)]$$

for any α.

Projected-Gradient for L1-Regularization

- We've considered writing our L1-regularization problem

$$\min_x f(x) + \lambda \|x\|_1,$$

as a problem with simple constraints:

$$\min_{x^+ \geq 0, x^- \geq 0} f(x^+ - x^-) + \lambda \sum_i (x_i^+ + x_i^-),$$

and then applying projected-gradient.

- But this problem might be hard to solve.
  - The transformed problem is never strongly-convex.
- Can we develop a method that works with the original problem?

If

$$F(x^+, x^-) = f(x^+ - x^-) + \lambda \sum_{j=1}^{d} (x_j^+ - x_j^-)$$

then

$$\nabla^2 F(x^+, x^-) = \begin{bmatrix} \nabla^2 f(x^+ - x^-) & -\nabla^2 f(x^+ - x^-) \\ -\nabla^2 f(x^+ - x^-) & \nabla^2 f(x^+ - x^-) \end{bmatrix}$$

which has at least d eigenvalues of 0: never strongly-convex.

Summary

- Coordinate optimization convergence rate analysis.
- Group L1-regularization encourages sparsity in variable groups.
- Structured sparsity encourages other patterns in variables.
- Projected-gradient allows optimization with simple constraints.

- Next time: what if the number of training examples ‘n’ is huge?