SVAN 2016 Mini Course: Stochastic Convex Optimization Methods in Machine Learning

Mark Schmidt

University of British Columbia, May 2016

www.cs.ubc.ca/~schmidtm/SVAN16

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:
  - Coordinate optimization for special 'f', separable regularizers (last lecture).
  - Proximal-gradient methods for general 'f' and regularizers (this lecture).

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	Age ≤ 20	20 < Age ≤ 30	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

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

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

some norm

individual group

set of groups

- Encourages sparsity in terms of groups 'g'.
  - E.g., if G = { {1,2}, {3,4} } then we have:

$$\sum_{g \in G} \|x_g\|_2 = \sqrt{x_1^2 + x_2^2} + \sqrt{x_3^2 + x_4^2}$$

Variables x_1 and x_2 will either be both zero or both non-zero.

Variables x_3 and x_4 will either be both zero or both non-zero.

Group L1-Regularization

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

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

- Why is it called group "L1"-regularization?
  - If 'v' is a vector containing norms of the groups, it's the L1-norm of 'v'.

$$\text{E.g.,} \quad v = \begin{bmatrix} \sqrt{x_1^2 + x_2^2} \\ \sqrt{x_3^2 + x_4^2} \end{bmatrix} \quad \text{then} \quad \sum_{g \in G} \|x_g\|_2 = \sum_{k=1}^{|G|} v_k = \sum_{k=1}^{|G|} |v_k| = \|v\|_1$$

- Typical choices of norm:

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

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

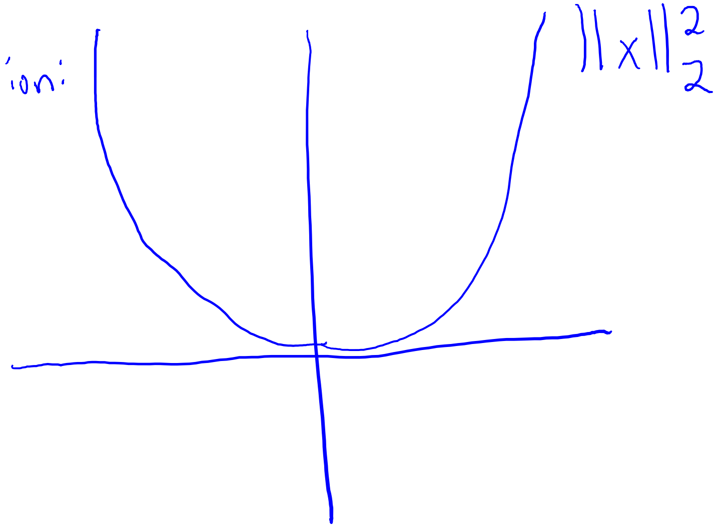
L_1-norm does not work: no grouping effect

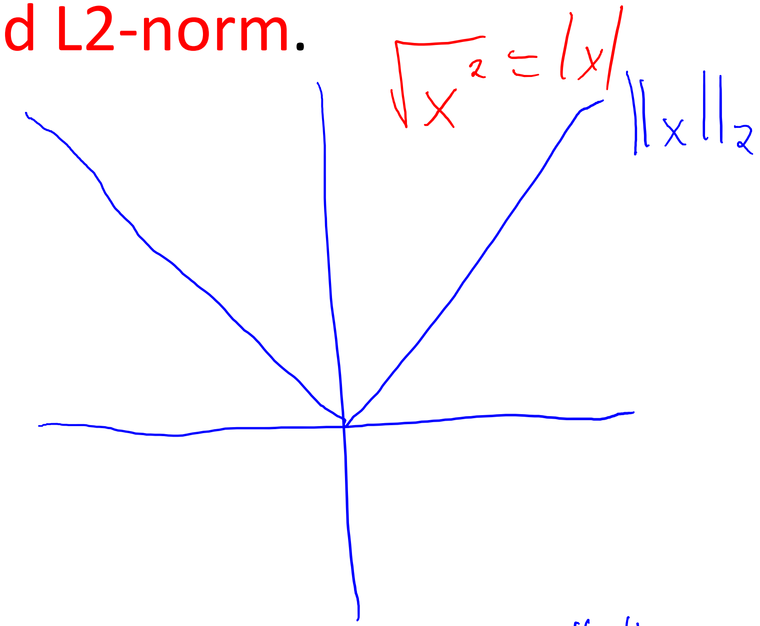
$$\|x\|_{1,1} = \sum_{g \in G} \|x_g\|_1 = \sum_{g \in G} \sum_{j \in g} |x_j| = \sum_{j=1}^{d} |x_j| = \|x\|_1$$

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.

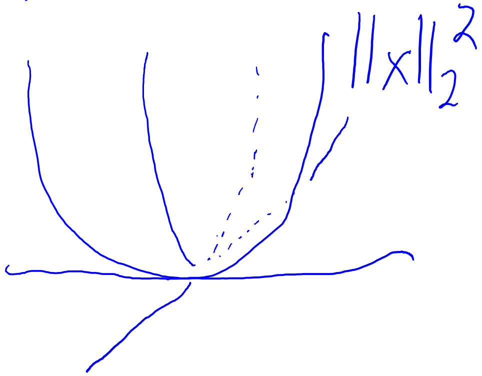
One-dimension:

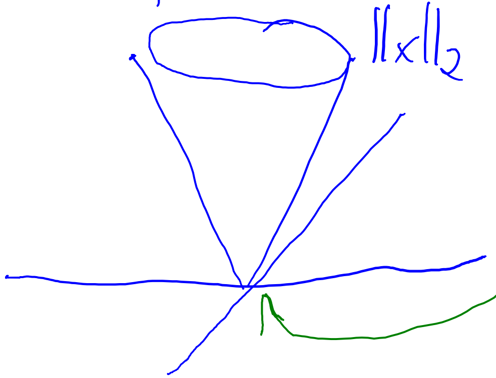




If you regularize by $\lambda\|w\|_2$ then for some finite λ it sets all variables to zero.

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 $\|x\|_2 = 0$.

Regularization Paths

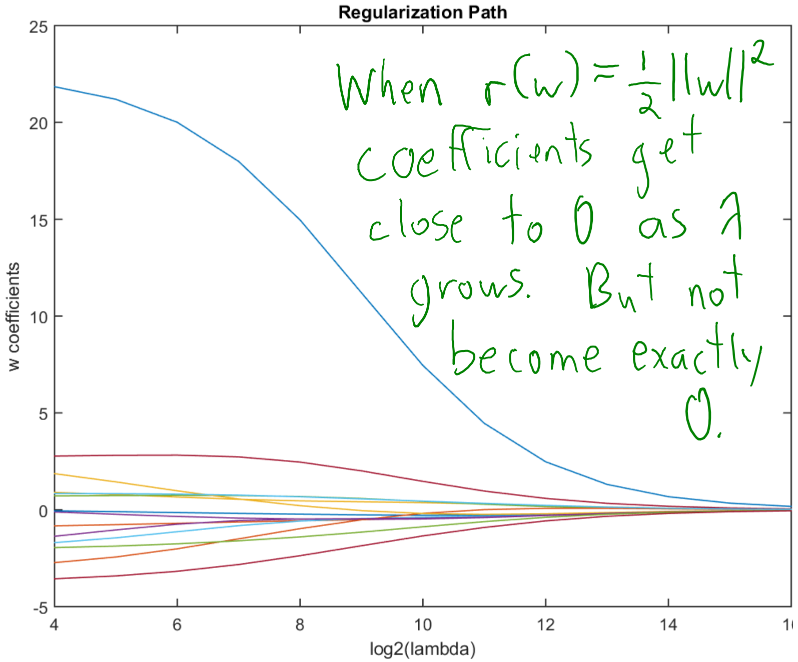
- The regularization path is the set of ‘w’ values as ‘λ’ varies:

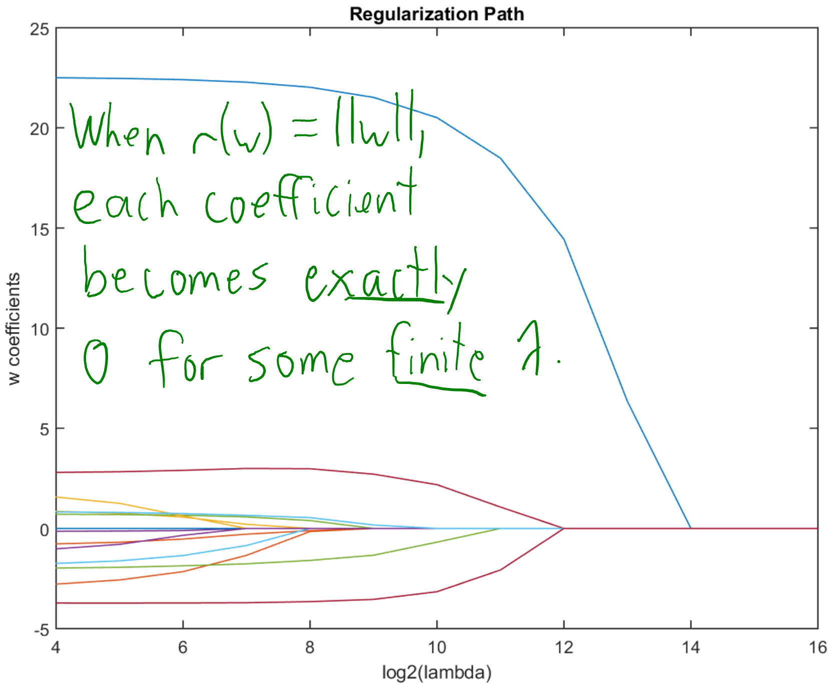
$$w^{\lambda_1} = \underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \; f(w) + \lambda_1 r(w)$$

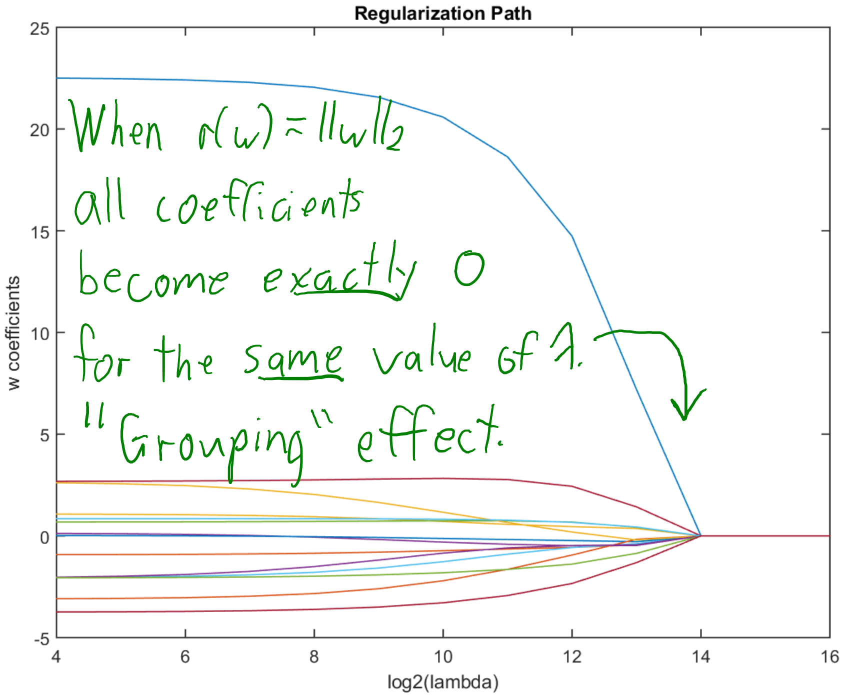
$$w^{\lambda_2} = \underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \; f(w) + \lambda_2 r(w)$$

$$w^{\lambda_3} = \underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \; f(w) + \lambda_3 r(w)$$

$$\vdots$$

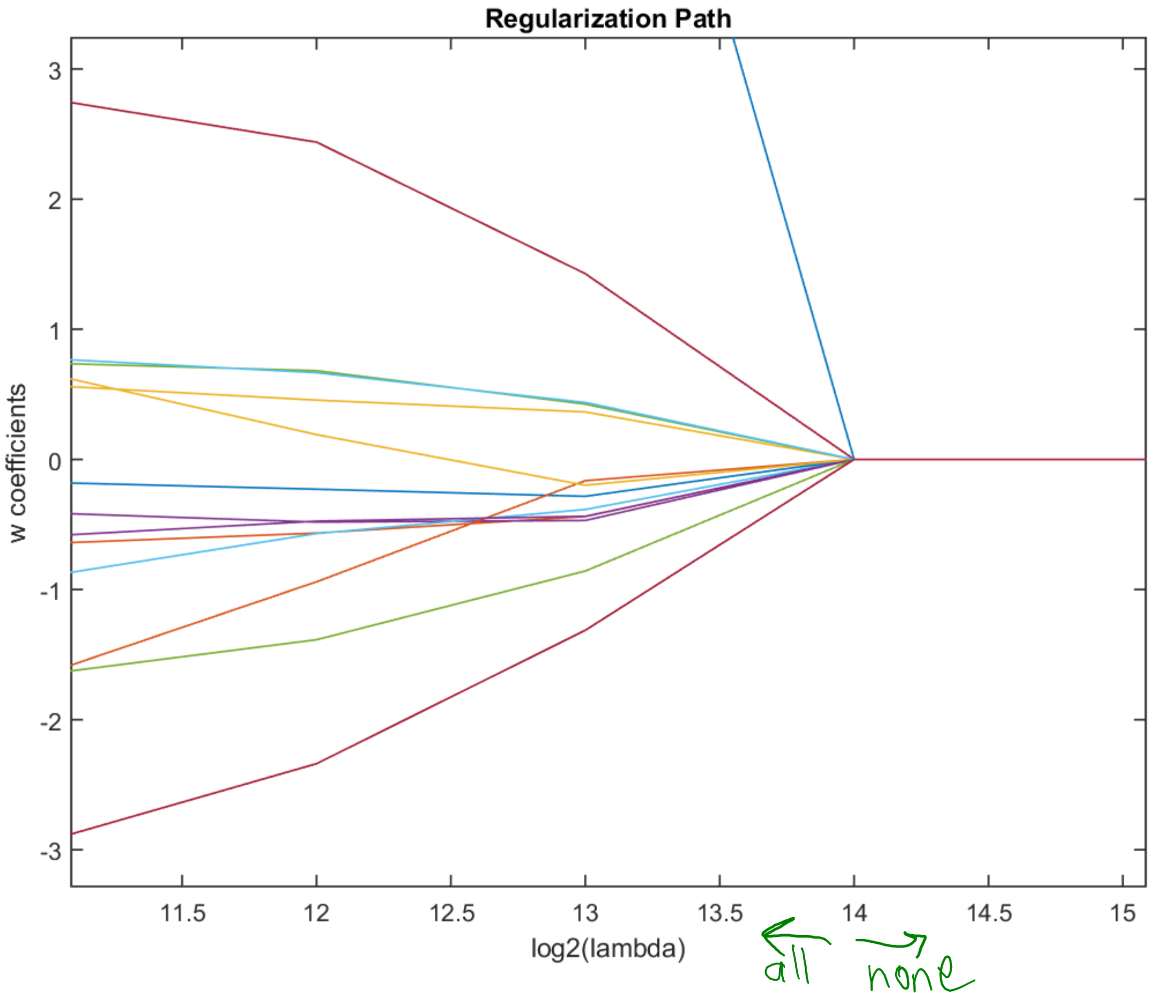






Regularization Paths

- The regularization path is the set of 'w' values as 'λ' varies:

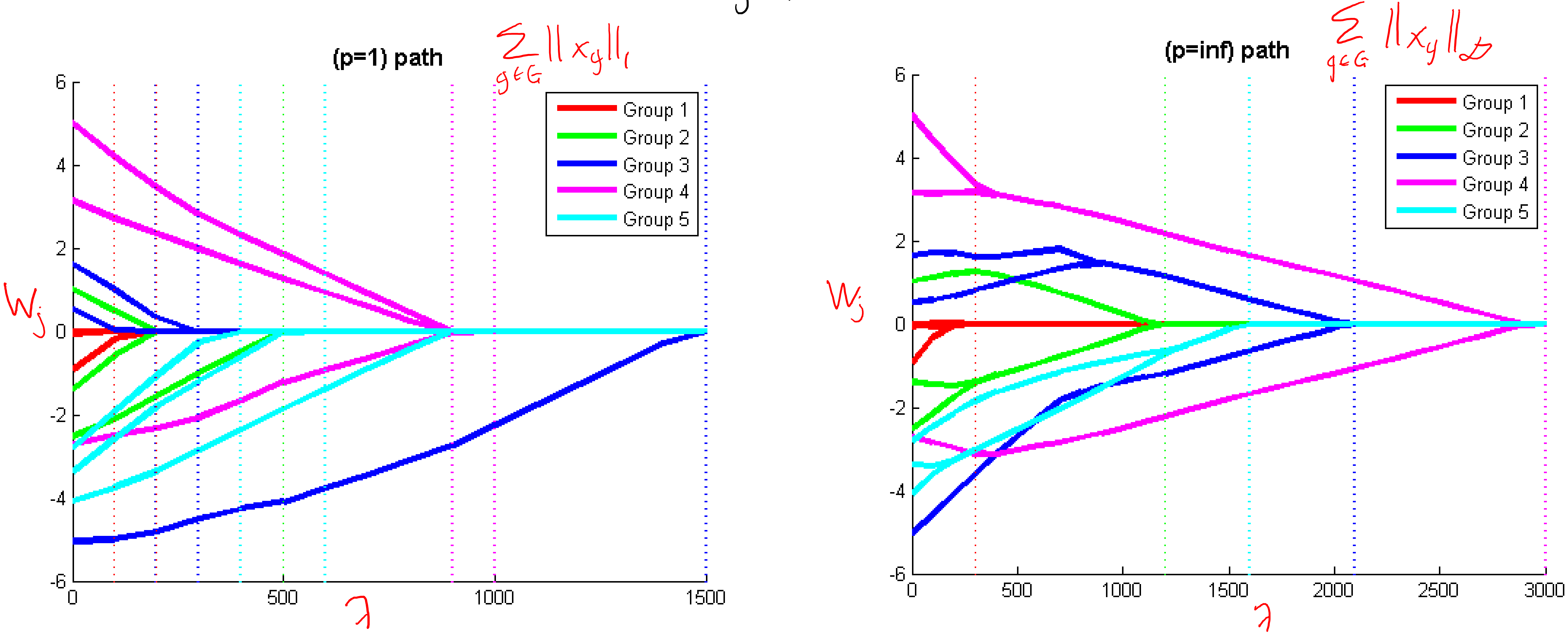


With non-squared L2-regularization, you tend to get all regularized variables or none of them.

Group L1-Regularization

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

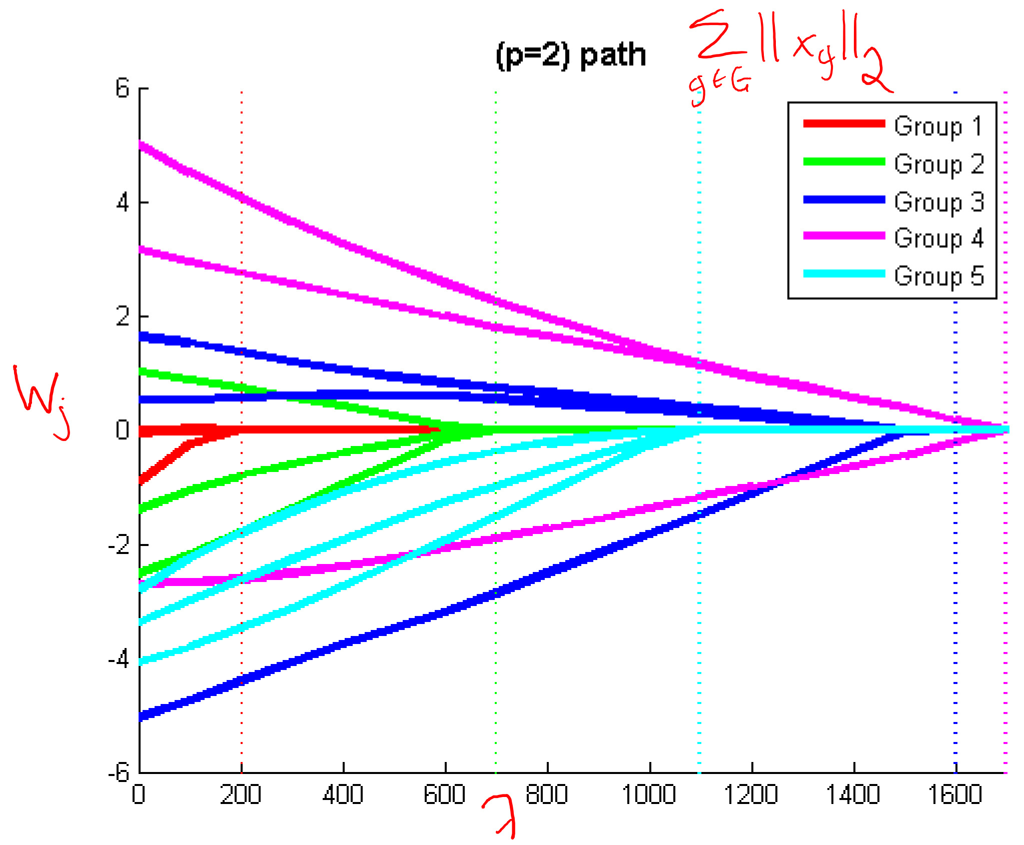
$$\underset{x \in \mathbb{R}^d}{\operatorname{argmin}} f(x) + \lambda \sum_{g \in G} \|x_g\|_p$$

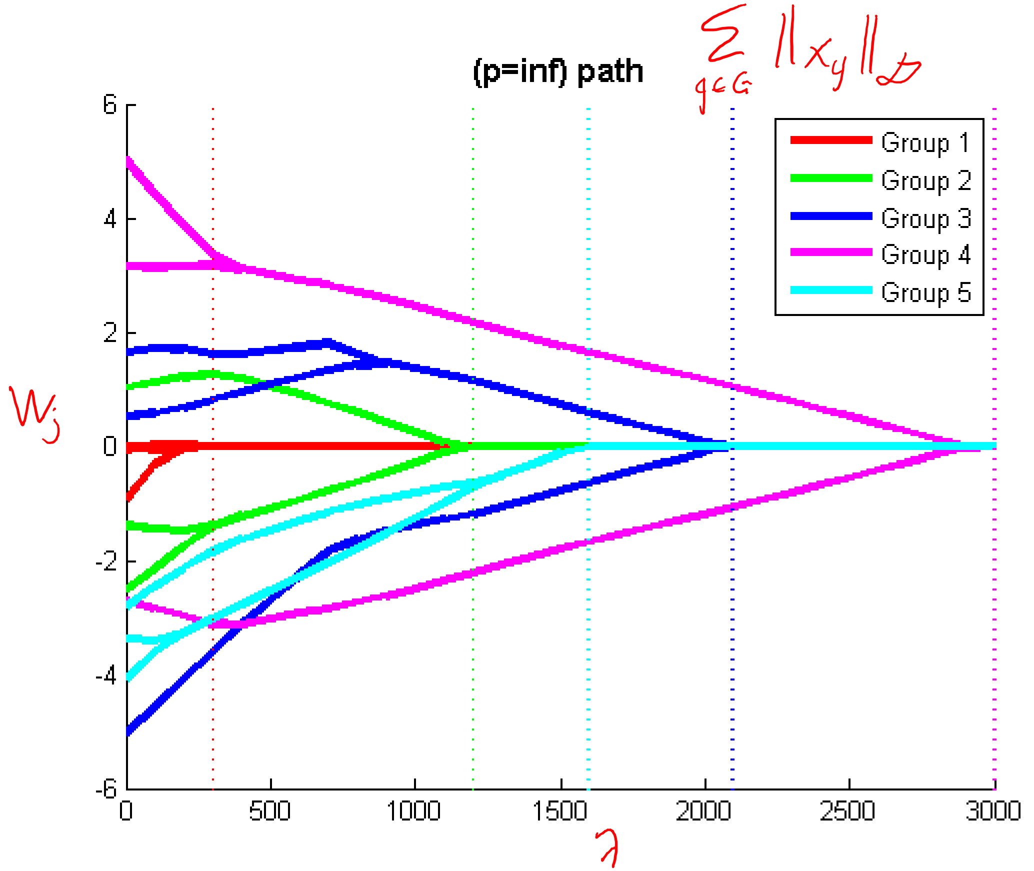


Group L1-Regularization

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

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

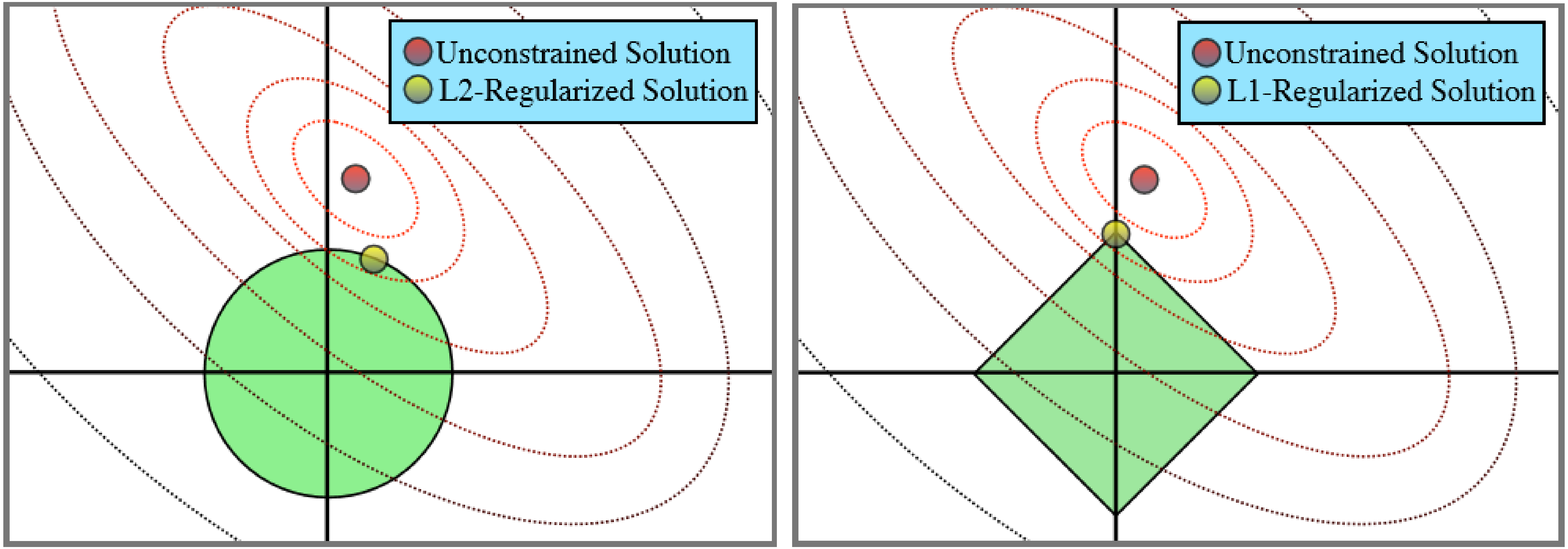




Group L1-Regularization

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

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

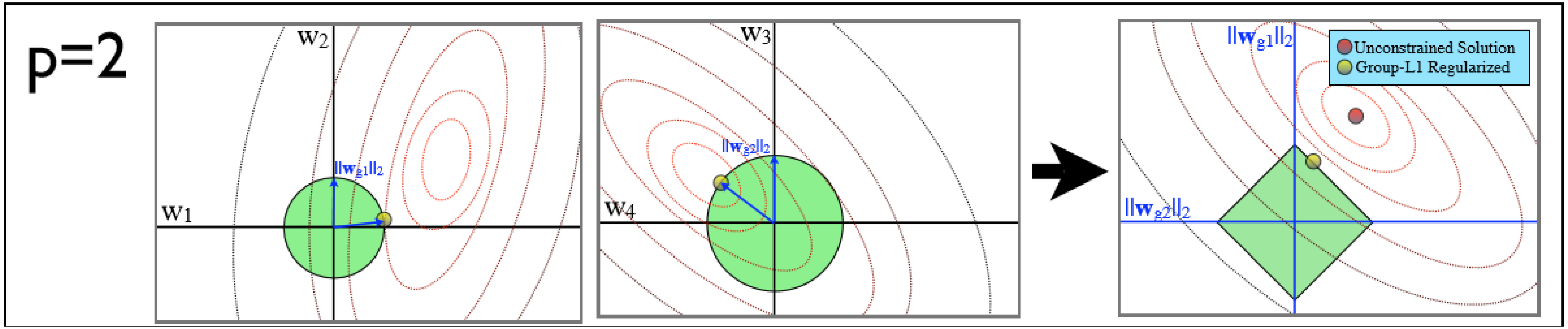


Group L1-Regularization

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

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

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

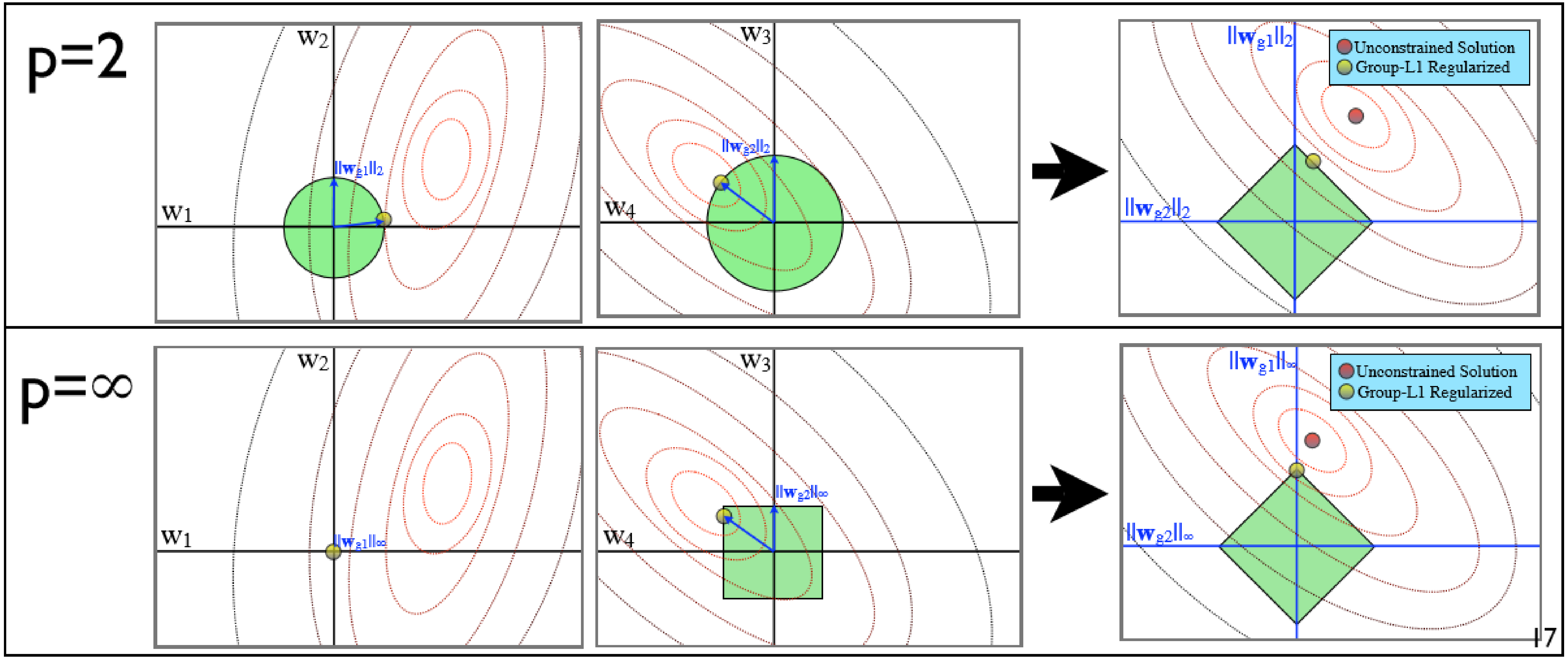


Group L1-Regularization

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

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

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



Other Applications of Group Sparsity

- Recall that multi-class logistic regression uses:

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

- 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.

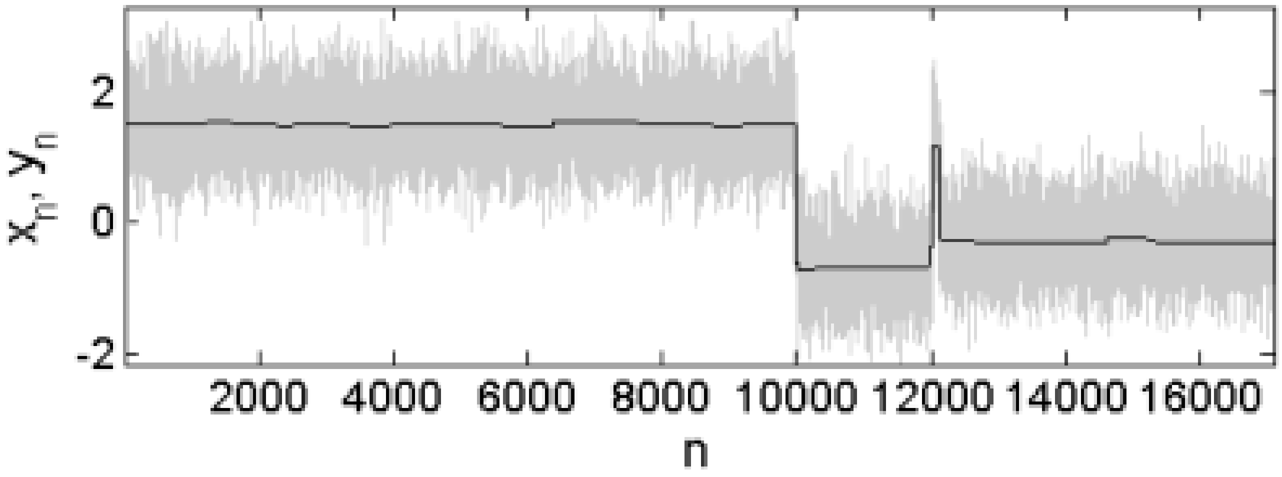
- To 'select' a feature 'j', we need 'w_{cj} = 0' for all 'j'.
- Same pattern also arises in multi-label and multi-task classification.

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.

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

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

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)

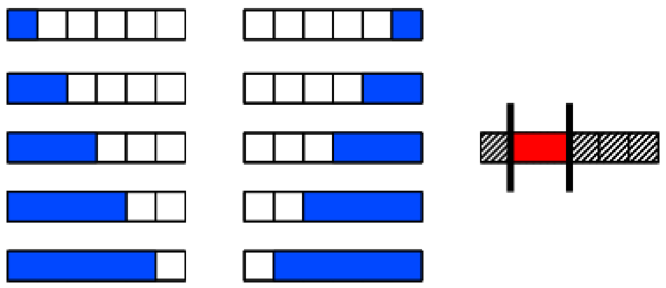


Fig 3: (Left) The set of blue groups to penalize in order to select contiguous patterns in a sequence. (Right) In red, an example of such a nonzero pattern with its corresponding zero pattern (hatched area).

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

Structured Sparsity

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

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

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

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

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.

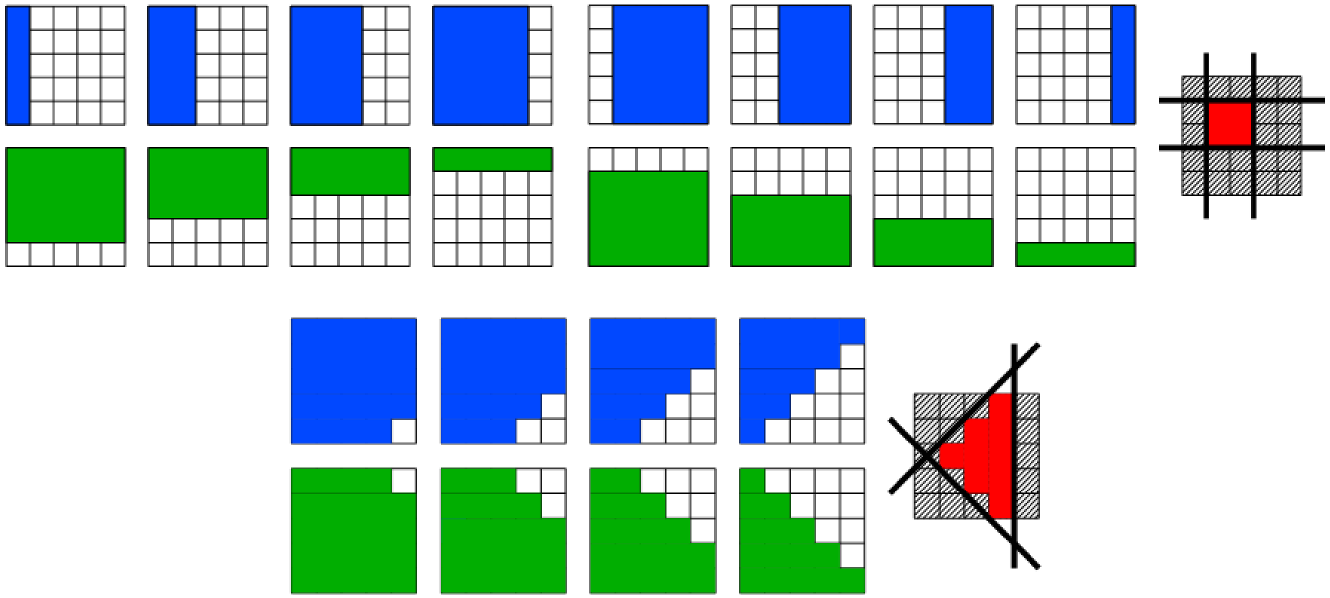
So possible non-zero patterns are $\{2\}$ or $\{1,2\}$.

Structured Sparsity

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

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

  - 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:

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

  - 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:

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

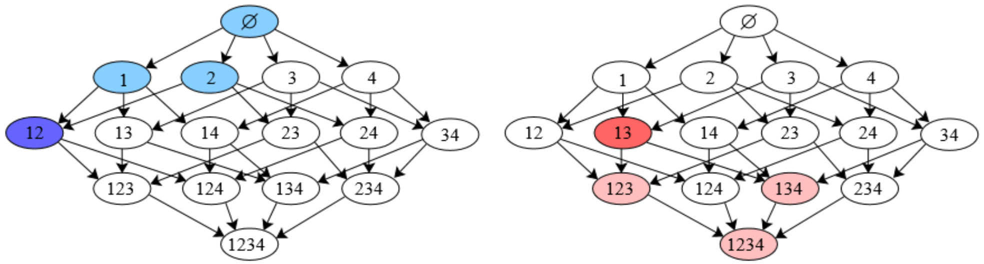


Fig 9: Power set of the set $\{1, \dots, 4\}$: in blue, an authorized set of selected subsets. In red, an example of a group used within the norm (a subset and all of its descendants in the DAG).

  - 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

- Structured sparsity objectives typically have the form:

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

- 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 $O(1/\varepsilon)$ 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:

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

nice because only constraints are that variables are non-negative.

or the problems

$$\underbrace{\min_{-y \leq x \leq y} f(x) + \lambda \sum_i y_i}, \qquad \min_{\|x\|_1 \leq \gamma} f(x) + \lambda \gamma$$

comes from usual 'max' trick

- These are smooth objectives with 'simple' constraints.

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

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

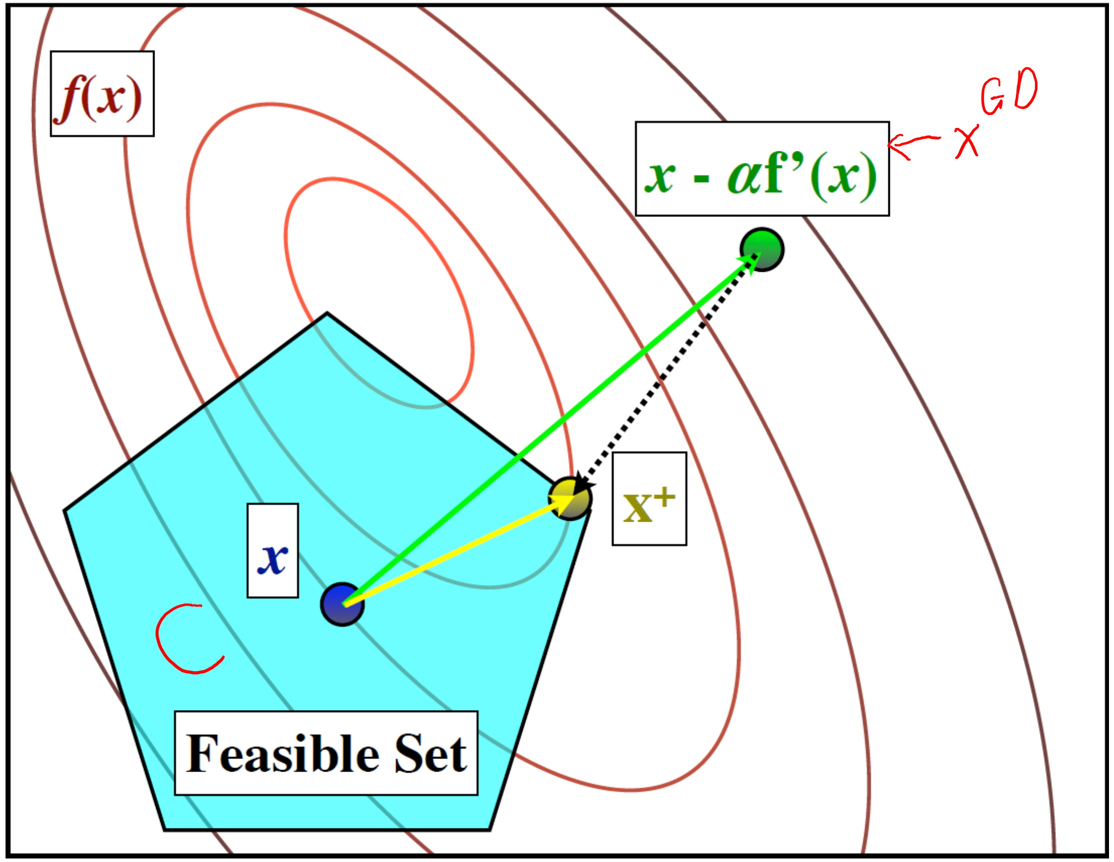
- This is called projected-gradient:

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

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

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

A set is ‘simple’ if we can efficiently compute projection.



Discussion of Projected-Gradient

- Convergence rates are the same for projected versions:

  f convex and non-smooth $O(1/\varepsilon^2)$
  f convex and ∇f Lipschitz $O(1/\varepsilon)$
  f strongly-convex and non-smooth $O(1/\varepsilon)$
  f strongly-convex and ∇f Lipschitz $O(\log(1/\varepsilon))$

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

Projection is a contraction

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

(moves x and y closer)

Solution x^* is a fixed point:

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

for any α.

"Simple" Convex Sets

- There are several "simple" sets that allows efficient projection:
  - Non-negative constraints (projection sets negative values to 0).
  - General lower/upper bounds on variables (projection sets to bound).
  - Small number of linear equalities (small linear system).
  - Small number of linear inequalities (small quadratic program).
  - Probability simplex (non-negative and sum-to-one).
  - Many norm-balls and norm-cones (L1, L2, L_∞).
- Dykstra's algorithm:
  - Compute projection onto intersection of simple sets.

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.