CPSC 540: Machine Learning

Structured Regularization

Mark Schmidt

University of British Columbia

Winter 2019

Last Time: Group L1-Regularization

- Last time we discussed group L1-regularization:

$$\operatorname*{argmin}_{w \in \mathbb{R}^d} f(w) + \lambda \sum_{g \in G} \|w_g\|_2.$$

- Encourages sparsity in terms of groups g.
  - For example, if $G = \{\{1,2\},\{3,4\}\}$ then we have:

$$\sum_{g \in G} \|w_g\|_2 = \sqrt{w_1^2 + w_2^2} + \sqrt{w_3^2 + w_4^2}.$$

    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.

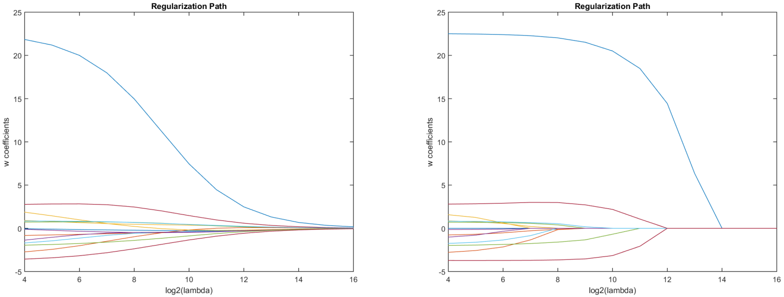
- Relevant for feature selection when each feature affects multiple parameters.
- It's important that we are using non-squared L2-norm.
  - Non-squared L2-norm is non-differentiable at zero.

L2 and L1 Regularization Paths

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

$$w^\lambda = \underset{w\in\mathbb{R}^d}{\text{argmin}}\, f(w) + \lambda r(w),$$

- Squared L2-regularization path vs. L1-regularization path:



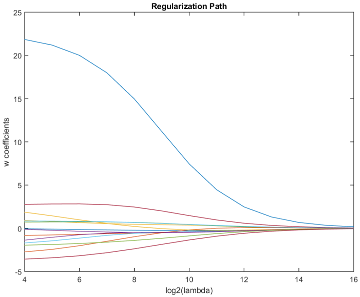
- With $r(w) = \|w\|^2$, each w_j gets close to 0 but is never exactly 0.
- With $r(w) = \|w\|_1$, each w_j gets set to exactly zero for a finite λ.

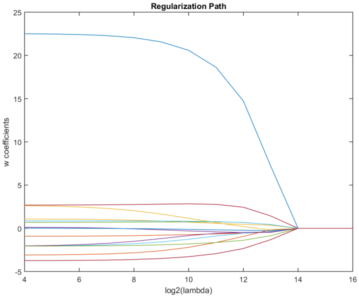
$L2^2$ and L2 Regularization Paths

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

$$w^\lambda = \underset{w \in \mathbb{R}^d}{\text{argmin}}\, f(w) + \lambda r(w),$$

- Squared L2-regularization path vs. non-squared path:

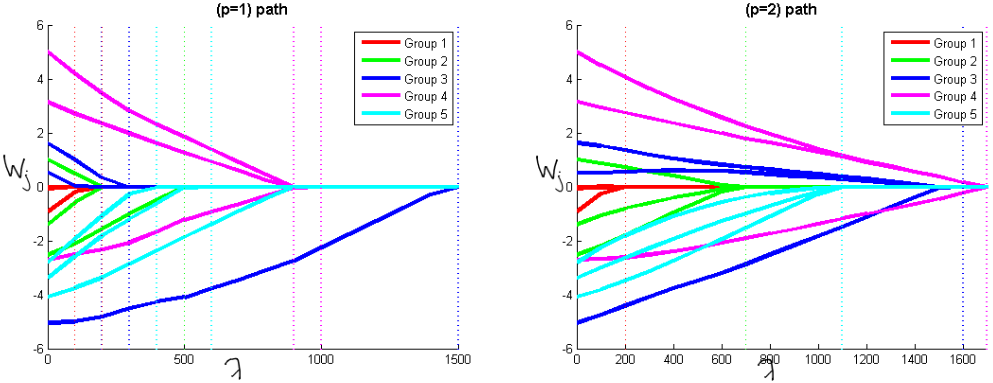




- With $r(w) = \|w\|^2$, each w_j gets close to 0 but is never exactly 0.
- With $r(w) = \|w\|_2$, all w_j get set to exactly zero for same finite λ.

Group L1-Regularization Paths

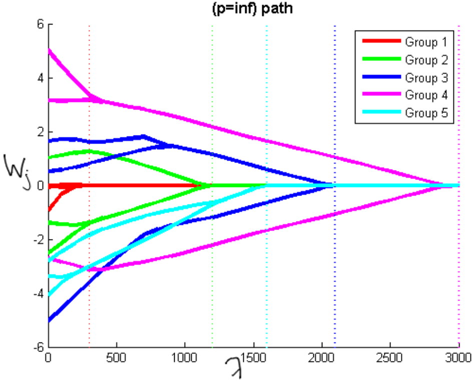
- The regularization path for group L1-regularizaiton for different p values:

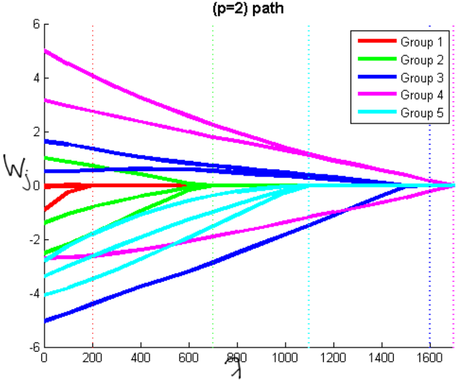


- With $p = 1$ there is no grouping effect.
- With $p = 2$ the groups become zero at the same time.

Group L1-Regularization Paths

- The regularization path for group L1-regularizaiton for different p values:





- With $p = 1$ there is no grouping effect.
- With $p = 2$ the groups become zero at the same time.
- With $p = \infty$ the groups converge to same magnitude which then goes to 0.

Last Time: Proximal-Gradient

- We discussed proximal-gradient methods for problems of the form

$$\operatorname*{argmin}_{w \in \mathbb{R}^d} \underbrace{f(w)}_{\text{smooth}} + \underbrace{r(w)}_{\text{simple}},$$

  where specifically $f \in C^1$ and r is convex.
- These methods use the iteration

$$w^{k+\frac{1}{2}} = w^k - \alpha_k \nabla f(w^k) \qquad \text{(gradient step)}$$

$$w^{k+1} \in \operatorname*{argmin}_{v \in \mathbb{R}^d} \left\{ \frac{1}{2}\|v - w^{k+\frac{1}{2}}\|^2 + \alpha_k r(v) \right\} \qquad \text{(proximal step)}$$

- Examples of simple functions include:
    - L1-regularization.
    - Group L1-regularization.
- Proximal operators for these cases are soft-thresholds: sets variables/groups to 0.

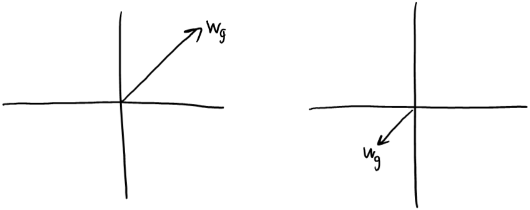
Proximal-Gradient for Group L1-Regularization

- The proximal operator for group L1-regularization,

$$\underset{v\in\mathbb{R}^d}{\operatorname{argmin}}\left\{\frac{1}{2}\|v-w\|^2+\alpha_k\lambda\sum_{g\in G}\|v\|_2\right\},$$

applies a soft-threshold group-wise,

$$w_g \leftarrow \frac{w_g}{\|w_g\|_2}\max\{0,\|w_g\|_2-\alpha_k\lambda\}.$$



- So we can solve group L1-regularization problems as fast as smooth problems.

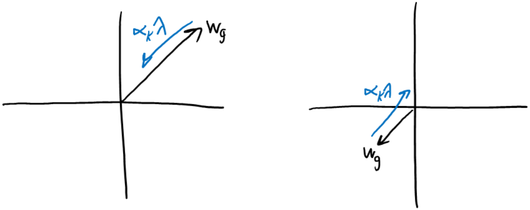
Proximal-Gradient for Group L1-Regularization

- The proximal operator for group L1-regularization,

$$\mathop{\mathrm{argmin}}_{v \in \mathbb{R}^d} \left\{ \frac{1}{2}\|v - w\|^2 + \alpha_k \lambda \sum_{g \in G} \|v\|_2 \right\},$$

applies a soft-threshold group-wise,

$$w_g \leftarrow \frac{w_g}{\|w_g\|_2} \max\{0, \|w_g\|_2 - \alpha_k \lambda\}.$$



- So we can solve group L1-regularization problems as fast as smooth problems.

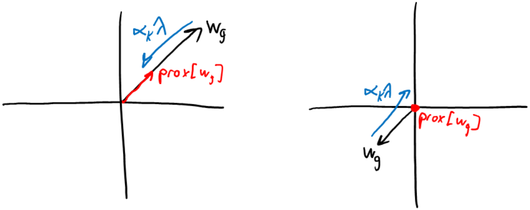
Proximal-Gradient for Group L1-Regularization

- The proximal operator for group L1-regularization,

$$\operatorname*{argmin}_{v\in\mathbb{R}^d}\left\{\frac{1}{2}\|v-w\|^2+\alpha_k\lambda\sum_{g\in G}\|v\|_2\right\},$$

applies a soft-threshold group-wise,

$$w_g \leftarrow \frac{w_g}{\|w_g\|_2}\max\{0,\|w_g\|_2-\alpha_k\lambda\}.$$



- So we can solve group L1-regularization problems as fast as smooth problems.

Outline

Structured Regularization

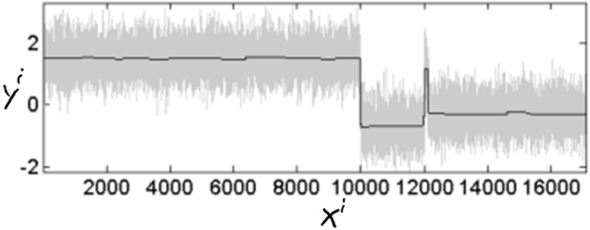
- There are many other patterns that regularization can encourage.
  - We'll call this structured regularization.
- The three most common cases:
  - Total-variation regularization encourages slow/sparse changes in w.
  - Nuclear-norm regularization encourages sparsity in rank of matrices.
  - Structured sparsity encourages sparsity in variable patterns.

Total-Variation Regularization

- 1D total-variation regularization ("fused LASSO") takes the form

$$\operatorname*{argmin}_{w\in\mathbb{R}^d} f(w) + \lambda \sum_{j=1}^{d-1} |w_j - w_{j+1}|.$$

- Encourages consecutive parameters to have same value.
- Often used for time-series or sequence data.



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

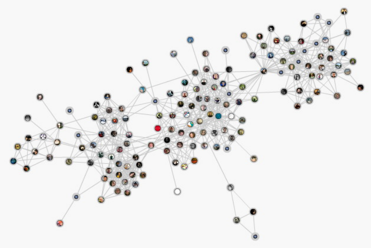
Here we're trying to estimate de-noised w_i of y^i at each time x^i.

Total-Variation Regularization

- More generally, we could penalizes differences on general graph between variables.
- An example is social regularization in recommeder systems:
  - Penalizing the difference between your parameters and your friends' parameters.

$$\underset{W \in \mathbb{R}^{d \times k}}{\operatorname{argmin}} \; f(W) + \lambda \sum_{(i,j) \in \text{Friends}} \|w_i - w_j\|^2.$$

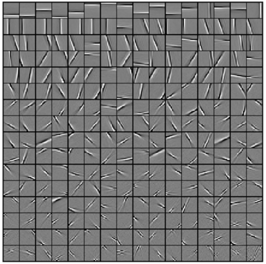
  - Typically use L2-regularization (we aren't aiming for identical parameters).

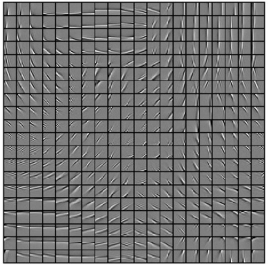


http://asawicki.info/news_1453_social_graph_-_application_on_facebook.html

Total-Variation Regularization

- Consider applying latent factor models (from 340) on image patches.
  - Similar to learning first layer of convolutional neural networks.
- Latent-factors discovered on patches with/without TV regularization.
  - Encouraging neighbours in a spatial grid to have similar filters.



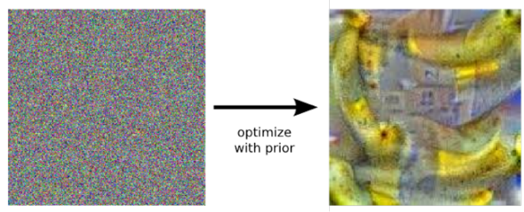


http://lear.inrialpes.fr/people/mairal/resources/pdf/review_sparse_arxiv.pdf

- Similar to “cortical columns” theory of visual cortex.

Total-Variation Regularization

- Another application is inceptionism.



https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html

- Find image x that causes strongest activation of class c in neural network.

$$\operatorname*{argmin}_x f(v_c^\top h(W^{(m)} h(W^{(m-1)} \cdots h(W^{(1)} x) + \lambda \sum_{(x_i, x_j) \in \text{neigh.}} (x_i - x_j)^2,$$

- Total variation based on neighbours in image (needed to get interpretable images).

Nuclear Norm Regularization

- With matrix parameters an alternative is nuclear norm regularization,

$$\underset{W \in \mathbb{R}^{d \times k}}{\text{argmin}}\ f(W) + \lambda \|W\|_*,$$

 where $\|W\|_*$ is the sum of singular values.

- "L1-regularization of the singular values".
  - Encourages parameter matrix to have low-rank.

- Consider a multi-class logistic regression with a huge number of features/labels,

$$W = \begin{bmatrix} | & | & & | \\ w_1 & w_2 & \cdots & w_k \\ | & | & & | \end{bmatrix} = UV^\top, \quad \text{with} \quad U = \begin{bmatrix} | & | \\ u_1 & u_2 \\ | & | \end{bmatrix}, V = \begin{bmatrix} | & | \\ v_1 & v_2 \\ | & | \end{bmatrix},$$

 U and V can be much smaller, and $XW = (XU)V^\top$ can be computed faster:
  - $O(ndk)$ cost reduced to $O(ndr + nkr)$ for rank r, much faster if $r < \min\{d, k\}$.

Structured Sparsity

- Structured sparsity is variation on group L1-regularization,

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

  where now the groups g can overlap.

- Why is this interesting?
  - Consider the case of two groups, $\{1\}$ and $\{1,2\}$,

$$\underset{w \in \mathbb{R}^d}{\text{argmin}}\, f(w) + \lambda_1 |w_1| + \lambda_2 \sqrt{w_1^2 + w_2^2}.$$

  - This encourages 3 non-zero "patterns": $\{\}$, $\{w_2\}$, $\{w_1, w_2\}$.
    - "You can only take w_1 if you've already taken w_2."
  - If $w_1 \neq 0$, the third term is smooth and doesn't encourage w_2 to be zero.
  - If $w_2 \neq 0$, we still pay a λ_1 penalty for making w_1 non-zero.
  - We can use this type of "ordering" to impose patterns on our sparsity.

Structured Sparsity

- Consider a problem with matrix parameters W.
- We want W to be "band-limited":
  - Non-zeroes only on the main diagonals.

$$W = \begin{bmatrix} w_{11} & w_{12} & w_{13} & 0 & 0 & 0 & 0 \\ w_{21} & w_{22} & w_{23} & w_{24} & 0 & 0 & 0 \\ w_{31} & w_{32} & w_{33} & w_{34} & w_{35} & 0 & 0 \\ 0 & w_{42} & w_{43} & w_{44} & w_{45} & w_{46} & 0 \\ 0 & 0 & w_{53} & w_{54} & w_{55} & w_{56} & w_{57} \\ 0 & 0 & 0 & w_{64} & w_{65} & w_{66} & w_{67} \\ 0 & 0 & 0 & 0 & w_{75} & w_{76} & w_{77} \end{bmatrix}.$$

  - This makes many computations much faster.
- We can enforce this with structured sparsity:
  - Only allow non-zeroes on ± 1 diagonal if you are non-zero on main diagonal.
  - Only allow non-zeroes on ± 2 diagonal if you are non-zero on ± 1 diagonal.
  - Only allow non-zeroes on ± 3 diagonal if you are non-zero on ± 2 diagonal.

Structured Sparsity

- Consider a linear model with higher-order terms,

$$\hat{y}^i = w_0 + w_1 x_1^i + w_2 x_2^i + w_3 x_3^i + w_{12} x_1^i x_2^i + w_{13} x_1^i x_3^i + w_{23} x_2^i x_3^i + w_{123} x_1^i x_2^i x_3^i.$$

  - If d is non-trivial, then the number of higher-order terms is too large.

- We can use structured sparsity to enforce a hierarchy.
  - We only allow $w_{12} \neq 0$ if $w_1 \neq 0$ and $w_2 \neq 0$.
    - You can enforce this using the groups $\{\{w_{12}\}, \{w_1, w_{12}\}, \{w_2, w_{12}\}\}$:

$$\operatorname*{argmin}_w f(w) + \lambda_{12}|w_{12}| + \lambda_1\sqrt{w_1^2 + w_{12}^2} + \lambda_2\sqrt{w_2^2 + w_{12}^2}.$$

Structured Sparsity

- We can use structured sparsity to enforce a hierarchy.
  - We only allow $w_{12} \neq 0$ if $w_1 \neq 0$ and $w_2 \neq 0$.
  - We only allow $w_{123} \neq 0$ if $w_{12} \neq 0$, $w_{13} \neq 0$, and $w_{23} \neq 0$.
  - We only allow $w_{1234} \neq 0$ if all threeway interactions are present.

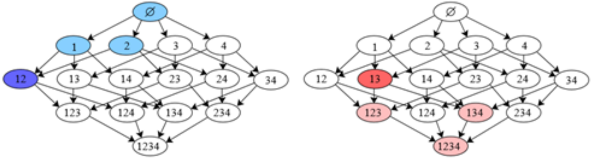


Fig 9: Power set of the set $\{1, \ldots, 4\}$: in blue, an authorized set of selected subsets. In red, an example of a group used within the norm (a subset and all of its descendants in the DAG).

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

- For certain bases, you can work with the full hierarchy in polynomial time.
  - Otherwise, a heuristic is to gradually "grow" the set of allowed bases.

Structured Sparsity

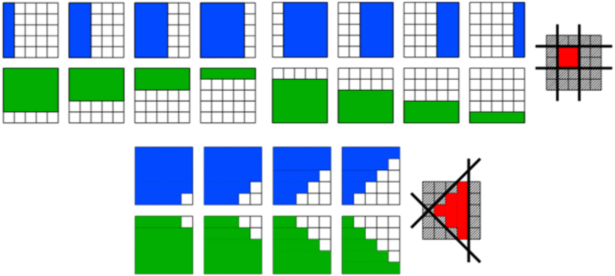
- Structured sparsity encourages zeroes to be any intersections of groups.
  - Possible non-zeroes are given by $\cap_{g \in \mathcal{G}'} g^c$ for all $\mathcal{G}' \subseteq \mathcal{G}$.
    - Equivalently, the set of zeroes is any $\cup_{g \in \mathcal{G}'} g$.
  - Our first example used $\{1\}$ and $\{1, 2\}$ so possible non-zeroes $\{\}$, $\{2\}$, or $\{1, 2\}$.
    - E.g., $\{2\}$ is $\{1, 2\} \cap \{1\}^c = \{1, 2\} \cap \{2\}$.
- Example is enforcing convex non-zero patterns:

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).

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

Structured Sparsity

- Structured sparsity encourages zeroes to be any intersections of groups.
  - Possible non-zeroes are given by $\cap_{g \in \mathcal{G}'} g^c$ for all $\mathcal{G}' \subseteq \mathcal{G}$.
    - Equivalently, the set of zeroes is any $\cup_{g \in \mathcal{G}'} g$.
  - Our first example used $\{1\}$ and $\{1,2\}$ so possible non-zeroes $\{\}$, $\{2\}$, or $\{1,2\}$.
    - E.g., $\{2\}$ is $\{1,2\} \cap \{1\}^c = \{1,2\} \cap \{2\}$.
- Example is enforcing convex non-zero patterns:



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

Structured Sparsity

- Structured sparsity encourages zeroes to be any intersections of groups.
  - Possible non-zeroes are given by $\cap_{g \in \mathcal{G}'} g^c$ for all $\mathcal{G}' \subseteq \mathcal{G}$.
    - Equivalently, the set of zeroes is any $\cup_{g \in \mathcal{G}'} g$.
  - Our first example used $\{1\}$ and $\{1,2\}$ so possible non-zeroes $\{\}$, $\{2\}$, or $\{1,2\}$.
    - E.g., $\{2\}$ is $\{1,2\} \cap \{1\}^c = \{1,2\} \cap \{2\}$.
- Example is enforcing convex non-zero patterns:

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

- Left-to-right: data, NMF, sparse PCA, and PCA with structured sparsity.

Structured Sparsity

- Structured sparsity encourages zeroes to be any intersections of groups.
  - Possible non-zeroes are given by $\cap_{g \in \mathcal{G}'} g^c$ for all $\mathcal{G}' \subseteq \mathcal{G}$.
    - Equivalently, the set of zeroes is any $\cup_{g \in \mathcal{G}'} g$.
  - Our first example used $\{1\}$ and $\{1,2\}$ so possible non-zeroes $\{\}$, $\{2\}$, or $\{1,2\}$.
    - E.g., $\{2\}$ is $\{1,2\} \cap \{1\}^c = \{1,2\} \cap \{2\}$.
- Example is enforcing convex non-zero patterns:

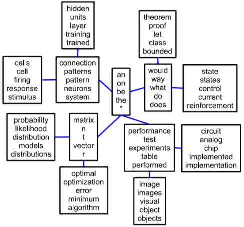


Figure 4. Example of a topic hierarchy estimated from 1714 NIPS proceedings papers (from 1988 through 1999). Each node corresponds to a topic whose 5 most important words are displayed. Single characters such as n, t, r are part of the vocabulary and often appear in NIPS papers, and their place in the hierarchy is semantically relevant to children topics.

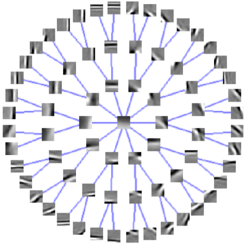


Figure 3. Learned dictionary with tree structure of depth 4. The root of the tree is in the middle of the figure. The branching factors are $p_1 = 10$, $p_2 = 2$, $p_3 = 2$. The dictionary is learned on 50,000 patches of size 16×16 pixels.

www.di.ens.fr/~fbach/icml2010a.pdf

- There is also a variant ("over-LASSO") that considers unions of groups.

Outline

Structured Regularization

- The three most common cases of structured regularization:
  - Total-variation regularization encourages slow/sparse changes in w.
  - Nuclear-norm regularization encourages sparsity in rank of matrices.
  - Structure sparsity encourages sparsity in variable patterns.
- Unfortunately, these regularizers are not "simple".
- But we can efficiently approximate the proximal operator in all these cases.

Inexact Proximal-Gradient Methods

- For total-variation and overlapping group-L1, we can use Dykstra's algorithm
  - Iterative method that computes proximal operator for sum of "simple" functions.
- For nuclear-norm regularization, many method approximate top singular vectors.
  - Krylov subspace methods, randomized SVD approximations.
- Inexact proximal-gradient methods:
  - Proximal-gradient methods with an approximation to the proximal operator.
  - If approximation error decreases fast enough, same convergence rate:
    - To get $O(\rho^t)$ rate, error must be in $o(\rho^t)$.

Alternating Direction Method of Multipliers

- ADMM is also popular for structured sparsity problems

- Alternating direction method of multipliers (ADMM) solves:

$$\min_{Aw+Bv=c} f(w) + r(v).$$

- Alternates between proximal operators with respect to f and r.
  - We usually introduce new variables and constraints to convert to this form.
- We can apply ADMM to L1-regularization with an easy prox for f using

$$\min_w \frac{1}{2}\|Xw - y\|^2 + \lambda\|w\|_1 \quad \Leftrightarrow \quad \min_{v=Xw} \frac{1}{2}\|v - y\|^2 + \lambda\|w\|_1,$$

- For total-variation and structured sparsity we can use

$$\min_w f(w) + \|Aw\|_1 \quad \Leftrightarrow \quad \min_{v=Aw} f(w) + \|v\|_1.$$

- If prox can not be computed exactly: linearized ADMM.
  - But ADMM rate depends on tuning parameter(s) and iterations aren't sparse.

Frank-Wolfe Method

- In some cases the projected gradient step

$$w^{k+1} = \operatorname*{argmin}_{y \in \mathcal{C}} \left\{ f(w^k) + \nabla f(w^k)^\top (v - w^k) + \frac{1}{2\alpha_k}\|v - w^k\|^2 \right\},$$

may be hard to compute.

- Frank-Wolfe step is sometimes cheaper:

$$w^{k+\frac{1}{2}} = \operatorname*{argmin}_{v \in \mathcal{C}} \left\{ f(w^k) + \nabla f(w^k)^\top (v - w^k) \right\},$$

requires compact $\mathcal{C}$, algorithm takes convex combination of w^k and $w^{k+\frac{1}{2}}$.

https://www.youtube.com/watch?v=24e08AX9Eww

- $O(1/t)$ rate for convex objectives, some linear results for strongly-convex.

Summary

- Structured regularization encourages more-general patterns in variables.
- Total-variation penalizes differences between variables.
- Structured sparsity can enforce sparsity hierarchies.
- Inexact proximal-gradient methods are a common approach to solving these.

- Next time: finding all the cat videos on YouTube.