Aalto University
School of Science
and Technology

Network Lasso: Clustering and Optimization in Large Graphs

KDD'15

David Hallac, Jure Leskovec, Stephen Boyd
presented by Jonathan Strahl

Probabilistic Machine Learning Group, Aalto University
Helsinki Institute for Information and Technology (HIIT)

March 9, 2017

Main contributions

- Simultaneous clustering and optimisation using a weighted sum of norms of locally weighted difference of nodes as regularisation.
- Algorithm based on alternating direction of multipliers method (ADMM) that is distributed and scalable with guarenteed global convergence.
- A non-convex extension that worked well empirically.

Scalability an issue with bigger and bigger datasets

- Convex optimisation growing in popularity (Finance, Image processing).
- Current methods, often rely on interior-point methods, fail to scale.
- One solution is to exploit data structure for problem-specific tasks. A poor, maybe infeasible solution.
- Looking for general optimization algorithms that scale.

Present Work: Example applications

- **Policy**: where graph are state transitions and w_{jk} represents importance of neighbouring actions differing. This leads to a simpler policy.
- In the case of a Markov decision process (MDP) some states will be similar with similar actions. These can be collapsed to a single state action mapping, the Network lasso learns to find these and collapse them.

Present Work: Example applications

- **Statistical modelling**: x_i model parameters for data at (associated with) node i. f_i is model loss. Edges encourage similar (or the same) parameters across models.
- Take a large dataset, one model can be learnt for all the data. However, it could be that data would be improved if stratified. In some cases the sub-groups could be known. In the case in which they are not Network lasso can help find the groups and break this larger problem into something similar to stratified sampling for machine learning.

Present Work: Use case

- House price prediction: Linear regression coefficients, however same house features in different location can be very differently priced, unknown a-priori difficult to quantify. Global model bad. Preferable to cluster into housing “neighbourhood” sharing a common regression model.
- Network where neighbouring houses have edges, each house has own regression model. Network lasso encourages neighbourhoods to share parameters. Neighbourhoods learnt empirically.

Present Work: Formulation

- Optimisation on a graph, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, vertex set $\mathcal{V}$ and edge set $\mathcal{E}$.

$$\text{minimize} \quad \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{(j,k) \in \mathcal{E}} g_{jk}(x_j, x_k), \tag{1}$$

- where $x_1, ..., x_m \in \mathbb{R}^p$
- $m = |\mathcal{V}|$
- $f_i : \mathbb{R}^p \to \mathbb{R} \cup \{\infty\}$, convex.
- $g_{jk} : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R} \cup \{\infty\}, g_{jk}(x_j, x_k) = \lambda \sum_{(j,k) \in \mathcal{E}} w_{jk} \|x_j - x_k\|_2$
- ∞ used as constraints
- $\lambda \geq 0$, user-defined $w_{jk} \geq 0$.

Present Work: Formulation: The Network Lasso

$$\text{minimize} \quad \sum_{i \in \mathcal{V}} f_i(x_i) + \lambda \sum_{(j,k) \in \mathcal{E}} w_{jk} \|x_j - x_k\|_2. \qquad (2)$$

- Solvable with standard solvers for small problems.
- Distributed scalable solution, each vertex x_i controlled by one agent, each exchange small messages, solve iteratively.
- Global convergence. λ controls similarity: large is full consensus, small are unique solutions.
- Slightly different formulation that is non-convex shown to perform well.
- NOTE: uses norm NOT norm squared, if squared reduces to Laplacian regularisation.
- Sum-of-norms is like group lasso. As encourages $x_i = x_j$ not $x_i \simeq x_j$ (consensus).

Convex problem definition

$$\text{minimize} \quad \sum_{i \in \mathcal{V}} f_i(x_i) + \lambda \sum_{(j,k) \in \mathcal{E}} w_{jk} \|x_j - x_k\|_2. \tag{3}$$

- Convex in $x = (x_1, ..., x_m) \in \mathbb{R}^{mp}$, x^* optimal solution.
- For simplicity private variables are excluded, $f_i(x_i) = \min_{\mathcal{E}_i} \tilde{f}_i(x_i, \mathcal{E}_i)$
- $\lambda \to \infty$ gives the *consensus problem*, minimize $\quad \sum_{i \in \mathcal{V}} f_i(\tilde{x}_i)$, with *consensus solution* $x^{\text{cons}} \in \mathbb{R}^p$.
- If solution to above exists, there exists a finite $\lambda_{\text{critical}}$
- *Regularisation path* (or cluster path) is $\lambda = [0, \lambda_{\text{critical}}]$

Convex problem definition

$$\text{minimize} \quad \sum_{i \in \mathcal{V}} f_i(x_i) + \lambda \sum_{(j,k) \in \mathcal{E}} w_{jk} \|x_j - x_k\|_2. \tag{4}$$

- $\|x_j - x_k\|_2$, ℓ_2 norm defines network lasso, encourages connected node differences to be exactly zero and does not penalize outliers too severely.
- When $x_j = x_k$ they are part of a group, set or cluster. Increasing λ tends to increase cluster size, but not strictly hierarchically, fission can occur when increasing λ.
- New node x_j inferred as minimize $\sum_{k \in N(j)} w_{jk} \|x_j - x_k^*\|_2$, a weighted median of j's neighbours, the Weber problem. Solvable in linear time.

Solve with ADMM: distributed and scalable

$$\text{minimize} \quad \sum_{i \in \mathcal{V}} f_i(x_i) + \lambda \sum_{(j,k) \in \mathcal{E}} w_{jk} \|z_{jk} - z_{kj}\|_2 \tag{5}$$

$$\text{s.t.} \quad x_i = z_{ij}, \quad i = 1, ..., m, \quad j \in N(i). \tag{6}$$

- where z_{ij} is a copy of x_i at edge ij.
- Augmented Lagrangian:

$$L_\rho(x, z, u) = \sum_{i \in \mathcal{V}} f_i(x_i) + \sum_{(j,k) \in \mathcal{E}} [\lambda w_{jk} \|z_{jk} - z_{kj}\|_2 - \tag{7}$$

$$(\rho/2)(\|u_{jk}\|_2^2 + \|u_{kj}\|_2^2) + \tag{8}$$

$$(\rho/2)(\|x_j - z_{jk} + u_{jk}\|_2^2 + \|x_k - z_{kj} + u_{kj}\|_2^2]. \tag{9}$$

- u is scaled dual variable, $\rho > 0$ is the penalty parameter.

ADMM steps

$$x^{k+1} = \arg\min_x L_\rho(x, z^k, u^k) \tag{10}$$

$$z^{k+1} = \arg\min_z L_\rho(x^{k+1}, z, u^k) \tag{11}$$

$$u^{k+1} = u^k + (x^{k+1} - z^{k+1}). \tag{12}$$

ADMM steps: x-Update

$$x_i^{k+1} = \arg\min_{x_i} \left(f_i(x_i) + \sum_{j \in N(i)} (\rho/2) \| x_i - z_{ij}^k + u_{ij}^k \|_2^2 \right). \tag{13}$$

- Minimize a separable sum of functions, one per node, independent, calculate parallel.

ADMM steps: z-Update

$$z_{ij}^{k+1}, z_{ji}^{k+1} = \arg\min_{z_{ij}, z_{ji}} [\lambda w_{ij} \|z_{ij} - z_{ji}\|_2 + \tag{14}$$

$$(\rho/2)(\|x_i^{k+1} - z_{ij} + u_{ij}^k\|_2^2 + \|x_j^{k+1} - z_{ji} + u_{ji}^k\|_2^2] \tag{15}$$

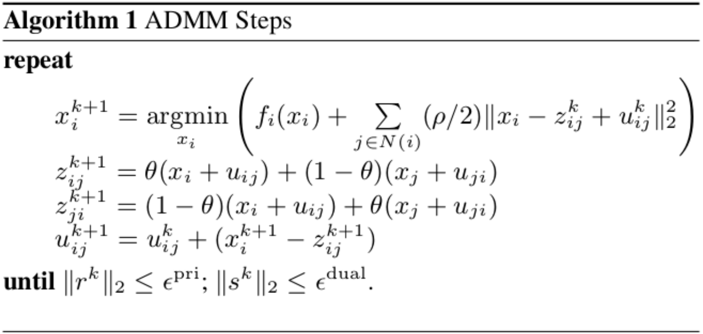
- Separable across edges, jointly update edges between same nodes.
- Close-form analytical solution.

ADMM steps: u-Update

$$u_{ij}^{k+1} = u_{ij}^{k} + (x_i^{k+1} - z_{ij}^{k+1}) \tag{16}$$

- Separable across edges.

ADMM steps: algorithm



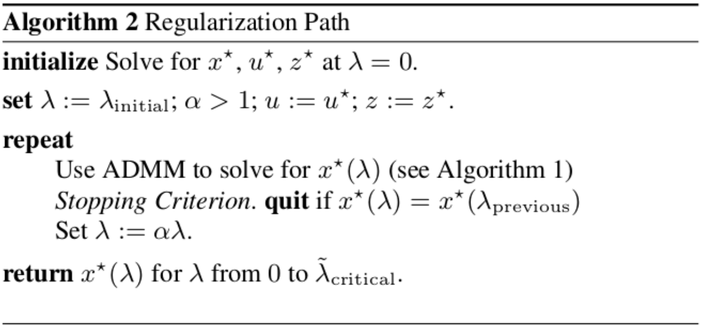
Algorithm 1 ADMM Steps

```
repeat
    $x_i^{k+1} = \underset{x_i}{\operatorname{argmin}} \left( f_i(x_i) + \sum_{j \in N(i)} (\rho/2) \|x_i - z_{ij}^k + u_{ij}^k\|_2^2 \right)$
    $z_{ij}^{k+1} = \theta(x_i + u_{ij}) + (1 - \theta)(x_j + u_{ji})$
    $z_{ji}^{k+1} = (1 - \theta)(x_i + u_{ij}) + \theta(x_j + u_{ji})$
    $u_{ij}^{k+1} = u_{ij}^k + (x_i^{k+1} - z_{ij}^{k+1})$
until $\|r^k\|_2 \leq \epsilon^{\text{pri}}; \|s^k\|_2 \leq \epsilon^{\text{dual}}$.
```

Regularization path

- Increase λ from 0 until either consensus or solution does not change more than some ϵ, $\alpha\lambda, \alpha > 1$.
- Regularisation path provides a warm start for each consecutive problem (would not be the case if run in parallel)
- Heuristic to initialise λ:
  - 1. Pick edge *ij* at random and find x_i^*, x_j^* at $\lambda = 0$.
  - 2. Evaluate the gradients of $f_i(x)$ and $f_j(x)$ at $x = (x_i^* + x_j^*)/2$.
  - 3. Set $\lambda_{\text{initial}} := 0.01 \left(\frac{\|\nabla f_i(x)\|_2 + \|\nabla f_i(x)\|_2}{2w_{ij}} \right)$.

Regularization path



Algorithm 2 Regularization Path

```
initialize Solve for $x^\star$, $u^\star$, $z^\star$ at $\lambda = 0$.
set $\lambda := \lambda_{\text{initial}}$; $\alpha > 1$; $u := u^\star$; $z := z^\star$.
repeat
    Use ADMM to solve for $x^\star(\lambda)$ (see Algorithm 1)
    Stopping Criterion. quit if $x^\star(\lambda) = x^\star(\lambda_{\text{previous}})$
    Set $\lambda := \alpha\lambda$.
return $x^\star(\lambda)$ for $\lambda$ from 0 to $\tilde{\lambda}_{\text{critical}}$.
```

Non-convex extension

- Lasso minimises non-zero differences between nodes $\|x_i - x_j\|_2$. (approximating the ℓ_0-norm)
- For a non-zero difference magnitude is not important. However, this convex problem will try to pull clusters closer together.
- Replacing the penalty with a monotonically nondecreasing concave function $\phi(u)$, where $\phi(0) = 0$ and $u \geq 0$ is closer to the ℓ_0.
- This leads to a non-convex optimization problem, without ADMM convergence guarentees, or global optimum convergence guarantee.
- It finds different solutions for different initial values x, u, z, ρ.

$$\text{minimize} \quad \sum_{i \in \mathcal{V}} f_i(x_i) + \lambda \sum_{(j,k) \in \mathcal{E}} w_{jk} \phi(\|x_j - x_k\|_2). \qquad (17)$$

Non-convex extension

$$\text{minimize} \quad \sum_{i \in \mathcal{V}} f_i(x_i) + \lambda \sum_{(j,k) \in \mathcal{E}} w_{jk} \phi(\|x_j - x_k\|_2). \qquad (18)$$

- To help with convergence a heuristic is added to ADMM that empirically performs very well.
- Simply keeping track of the iteration that had the best results and returning to that as the solution and not the most recent solution.
- As primal and dual residuals may not be 0, algorithm is run for a set number of iterations for each λ.
- More details on this in the paper.

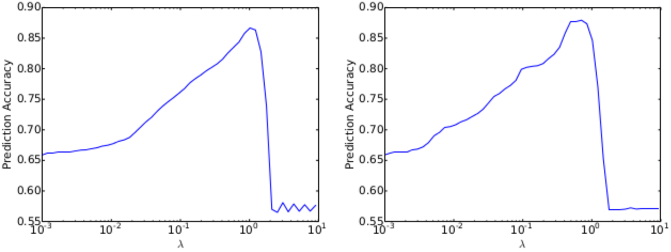
Experiments

- 1) Synthetic example gathering statistical strength from a network to improve classification accuracy
- 2) House price prediction leveraging geographic information.
- 3) Detecting outliers from a time-series dataset. *(not covered in this presentation)*

Experiments: Network-Enhanced Classification

- **Set up**: Each node has an SVM with insufficient training data.
- Clustering based on similarity of SVM model - some have the same SVM.
- Hope that similar SVMs borrow training examples to improve classification.
- **Size of problem**: 1000 nodes, 20 equally sized groups, each group with common SVM.
- In group edge p(0.5), our of group edge p(0.01). 17079 edges. 28% across clusters.
- **Timing**: with 20k unknowns found to be 100 times faster than centralized method.

Experiments: Network-Enhanced Classification



(a) Convex

(b) Non-Convex

Figure 2: SVM regularization path.

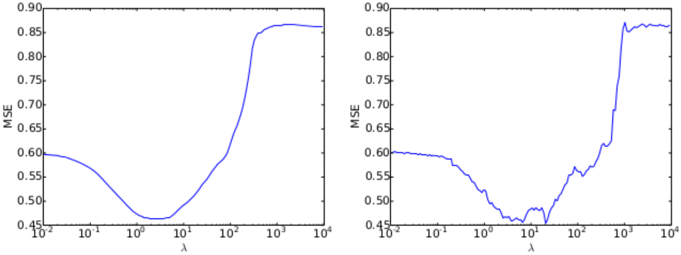
Method	Maximum Prediction Accuracy
Local SVM ($\lambda = 0$)	65.90%
Global SVM ($\lambda \geq \lambda_{\text{critical}}$)	57.10%
Convex Network Lasso	86.68%
Non-Convex Network Lasso	87.94%

Table 1: SVM test set prediction accuracy.

Experiments: Spatial Clustering with Regressors

- **Set-up:** House price prediction, with longtitude/latitude information for network construction.
- Network constructed by nearby houses having similar pricing models. Non-zero differences between different neighbourhoods.
- Neighbourhoods use common regression model and new houses are inferred using model from it's neighbourhood.
- Edge weight inversely proportional to the distance between five nearest houses.
- **Size of problem:** 985 sales, 17% missing at least one feature. 200 test set. 785 nodes, 2447 edges, diameter 61.

Experiments: Network-Enhanced Classification



(a) Convex

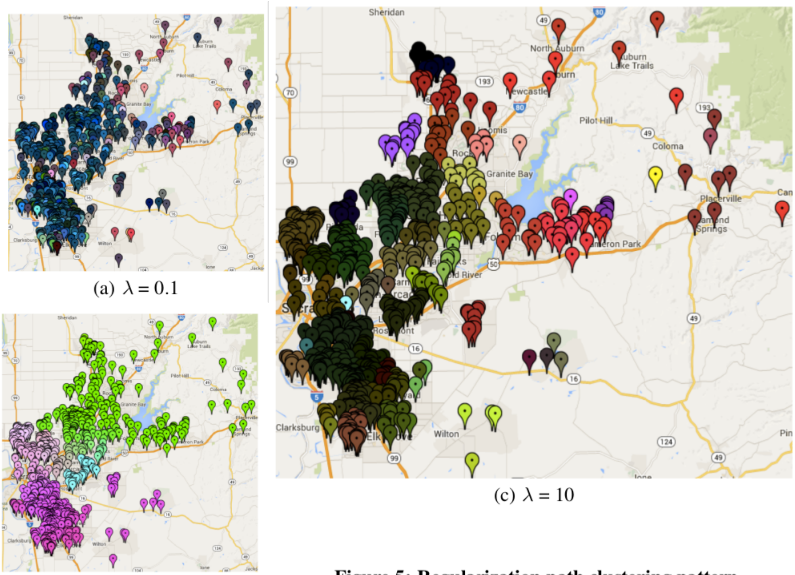
(b) Non-Convex

Figure 4: Regularization path for housing data.

Method	Mean Squared Error (MSE)
Geographic ($\lambda = 0$)	0.6013
Regularized Linear Regression ($\lambda \geq \lambda_{\text{critical}}$)	0.8611
Naive Prediction (Global Mean)	1.0245
Convex Network Lasso	0.4630
Non-Convex Network Lasso	0.4539

Table 3: MSE for housing price predictions on test set.

Experiments: Network-Enhanced Classification



(a) $\lambda = 0.1$

(b) $\lambda = 1000$

(c) $\lambda = 10$

Figure 5: Regularization path clustering pattern.

Conclusion and future work

- Network lasso is a single framework to help understand and improve many machine learning and network analysis problems.
- A useful way to represent convex optimisation problems, results show impressive improvements.
- Non-convex formulation performed well in practise. Worth investigating different non-convex functions $\phi(u)$, network sensitivity?
- Improve ADMM for speed, performance and robustness, auto-determine optimal ρ.

Future work

- There is work extending on this in my group: Professor Samuel Kaski's Probabilistic Machine Learning group, in collaboration with Makoto Yamamda [YTI^{+}16].

Questions?

- Thanks for your time.
- Were the main parts of this work clear?

Reference material

M. Yamada, K. Takeuchi, T. Iwata, J. Shawe-Taylor, and S. Kaski,
Localized Lasso for High-Dimensional Regression, ArXiv e-prints (2016).