Covariance Selection 101

Peder Olsen
IBM TJ Watson Research Center

Workshop on covariance selection and
Graphical Model Structure Learning
ICML 2014

June 26th 2014

Covariance Estimation

- Given N i.i.d. samples $\{\mathbf{x}_i\}_{i=1}^N$,, $\mathbf{x}_i \in \mathbb{R}^n$ $\mathbf{x}_i \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ estimate $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$.
- Sample mean: $\hat{\boldsymbol{\mu}} = \frac{1}{N}\sum_{i=1}^N \mathbf{x}_i$
- Sample covariance:

$$\mathbf{S} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^\top.$$

- Covariance selection: estimate $\mathbf{P} = \boldsymbol{\Sigma}^{-1}$ when $\mathbf{P}$ is a sparse matrix. Note that $\mathbf{S}^{-1}$ is not sparse even when $\mathbf{P}$ is a sparse matrix due to sample errors.

- **Problem**: Given an empirical covariance matrix $\mathbf{S}$

$$\mathbf{S} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^\top.$$

find a sparse inverse covariance matrix $\mathbf{P}$ to represent the data.

- **Approach**: Minimize the **convex** objective function

$$\min_{\mathbf{P}\succ 0} F(\mathbf{P}) \stackrel{\text{def}}{=} L(\mathbf{P}) + \lambda\|\text{vec}(\mathbf{P})\|_1, \quad L(\mathbf{P}) = -\log\det(\mathbf{P}) + \text{trace}(\mathbf{SP}).$$

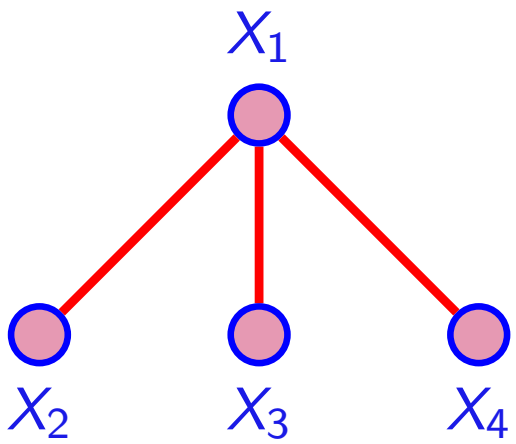
L is the negative log likelihood function and the ℓ_1 term is a sparsity inducing regularizer.

Why a sparse inverse covariance?

Why do we want to find a sparse matrix **P**?

- Understanding: The sparsity structure of **P** corresponds to the graphical model structure for a gaussian Markov random field.
- Computation: We can save both memory and computation for the log-likelihood evaluation when the matrix **P** is very sparse.
- Accuracy: Knowing where the zeros of **P** are lead to better statistical estimators.

Graphical Models



$$\Sigma^{-1} = \begin{pmatrix} \star & \star & \star & \star \\ \star & \star & 0 & 0 \\ \star & 0 & \star & 0 \\ \star & 0 & 0 & \star \end{pmatrix}$$

- When two nodes (X_2, X_3) are not connected in a graphical model they are conditionally independent given the other variables.
- For gaussian graphical models the inverse covariance matrix is zero whenever there is a missing link.

Non-gaussian graphical models

When the graphical model is not a gaussian:

- The structure of tree based graphical model can be found from the inverse covariance matrix
- Otherwise the structure can be found by looking at augmented inverse covariance matrices.

Po-Ling Loh and Martin J. Wainwright, "Structure estimation for discrete graphical models: Generalized covariance matrices and their inverses." *NIPS* (2012).

Covariance selection: the ℓ_0 approach

Optimize

$$-\log\det(\mathbf{P}) + \mathrm{trace}(\mathbf{SP}).$$

subject to $\mathrm{card}(\mathbf{P}) \leq k$.

Dempster solved the problem using a greedy forward method starting from the diagonal empirical covariance and a greedy backward elimination method starting from the full empirical covariance.

Dempster, Arthur P., "Covariance selection." *Biometrics*, pp. 157-175, (1972).

Covariance selection approach: the graphical LASSO

Graphical LASSO uses the convex relaxation

$$-\log\det(\mathbf{P}) + \mathrm{trace}(\mathbf{SP}) + \lambda\|\mathrm{vec}(\mathbf{P})\|_1$$

Onureena Banerjee, Laurent El Ghaoui, Alexandre d'Aspremont and Georges Natsoulis. "Convex optimization techniques for fitting sparse Gaussian graphical models." *ICML* pp. 89-96, (2006).

Covariance selection: Regression

Another approach is to find the set of neighbors of each node in the graphical model by regressing that variable against the remaining variables.

N. Meinshausen and P. Bühlmann, "High dimensional graphs and variable selection with the LASSO." *Annals of statistics*, **34**, pp. 1436–1462, (2006).

First Order Solvers

COVSEL A block-coordinate descent that solves the dual problem one row at a time.
Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. "Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data." *The Journal of Machine Learning Research* **9**, pp. 485-516, (2008).

GLASSO Graphical LASSO. One of the more popular solvers. It solves the primal problem one row at a time.
Alexandre d'Aspremont, Onureena Banerjee, and Laurent El Ghaoui. "First-order methods for sparse covariance selection." *SIAM Journal on Matrix Analysis and Applications* **30**(1), p. 56-66, (2008).

First Order Solvers

PSM Projected Sub-gradient Method.
J. Duchi, S. Gould, and D. Koller. "Projected subgradient methods for learning sparse Gaussians." *UAI* (2008).

SMACS Smooth Minimization Algorithm for Covariance Selection. An optimal first order ascent method.
Zhaosong Lu. "Smooth optimization approach for sparse covariance selection." *SIAM Journal on Optimization* **19**(4), pp. 1807–1827, (2009).

More Solvers

SINCO Sparse INverse COvariances. A method intended for massive parallel computation. A greedy coordinate descent method.
Katya Scheinberg and Irina Rish. "SINCO-a greedy coordinate ascent method for sparse inverse covariance selection problem." Technical Report, IBM RC24837 (2009).

ALM Alternating linear minimization. Uses an augmented Lagrangian to introduce an auxilliary variable for the non-smooth term.
Katya Scheinberg, Shiqian Ma, and Donald Goldfarb. "Sparse Inverse Covariance Selection via Alternating Linearization Methods." *NIPS* (2010).

More Solvers

IPM Interior Point Method. A second order interior point method.
Lu Li, and Kim-Chuan Toh. "An inexact interior point method for l1-regularized sparse covariance selection." *Mathematical Programming Computation* **2**(3-4), pp. 291-315, (2010).

QUIC A second order Newton method that solves the LASSO problem using a coordinate descent method.
C. J. Hsieh, M. Sustik, I. S. Dhillon, and P. Ravikumar. "Sparse inverse covariance matrix estimation using quadratic approximation." *NIPS*, (2011).

More Solvers

NL-FISTA A second order Newton method that solves the LASSO problem using a Fast Iterative Shrinkage Thresholding Algorithm (FISTA).

OBN OBN-LBFGS is an orthant based quasi Newton method and OBN-CG is an orthant based conjugate gradient method.
Peder Olsen, Figen Öztoprak, Jorge Nocedal and Steven Rennie "Newton-Like Methods for Sparse Inverse Covariance Estimation." *NIPS* 2012.

SpaRSA A generalized spectral projected gradient method that uses a spectral step length together with a nonmonotone line search to improve convergence
Jason D. Lee, Yuekai Sun, and Michael A. Saunders. "Proximal Newton-type methods for convex optimization." *NIPS* (2012).

More Solvers

DC-QUIC Divide and Conquer QUIC. A method that iteratively discovers better diagonal block approximations to the solution.
C. J. Hsieh, I. S. Dhillon, P. Ravikumar, A. Banerjee, "A Divide-and-Conquer Procedure for Sparse Inverse Covariance Estimation." *NIPS* (2012).

BIG & QUIC A solver that can handle million dimensional problems with a trillion variables.
C. J. Hsieh, M. Sustik, I. S. Dhillon, P. Ravikumar, and R. Poldrack, "BIG & QUIC: Sparse inverse covariance estimation for a million variables." *NIPS* (2013).

Problem Extensions

The penalty is a bit too simplistic. Consider the more general penalty term

$$\sum_{ij} \lambda_{ij} |P_{ij}|$$

Since $P_{ii} > 0$ is forced by the positive definite requirement we choose $\lambda_{ii} = 0$. We have found $\lambda_{ij} \propto \frac{1}{\sqrt{NS_{ii}S_{jj}}}$ to work well for $i \neq j$.

Another possible extension is to smooth towards Θ

$$\sum_{ij} \lambda_{ij} |P_{ij} - \Theta_{ij}|$$

It's also possible to consider "group LASSO" type of penalties with blocking in the covariance or other structural constraints in the sparsity of the inverse covariance.

The Exponential Family

Another avenue of extension worthy of consideration is the viewpoint of exponential families. The exponential family is characterized by the features $\phi(\mathbf{x})$ and is given by

$$\mathrm{P}(\mathbf{x}|\boldsymbol{\theta}) = \frac{e^{\boldsymbol{\theta}^\top \phi(\mathbf{x})}}{Z(\boldsymbol{\theta})}, \qquad Z(\boldsymbol{\theta}) = \int e^{\boldsymbol{\theta}^\top \phi(\mathbf{x})} \mathrm{d}\mathbf{x}.$$

$Z(\boldsymbol{\theta})$ is the partition function or normalizer. The covariance selection problem corresponds to the features $\phi(\mathbf{x}) = \mathrm{vec}(\mathbf{x}\mathbf{x}^\top)$, with parameters $\boldsymbol{\theta} = \mathrm{vec}(\mathbf{P})$ and log partition function $\log Z(\boldsymbol{\theta}) = \log \det(\mathbf{P}) + \frac{n}{2} \log(2\pi)$.

The general normal distribution

By extending the features to $\phi(\mathbf{x}) = \begin{pmatrix} \mathbf{x} \\ \mathrm{vec}(\mathbf{x}\mathbf{x}^\top) \end{pmatrix}$ we can consider the general normal distribution with parameters

$$\boldsymbol{\theta} = \begin{pmatrix} \boldsymbol{\psi} \\ \mathbf{P} \end{pmatrix} = \begin{pmatrix} \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} \\ \boldsymbol{\Sigma}^{-1} \end{pmatrix}.$$

The corresponding log likelihood function is

$$L(\boldsymbol{\theta}) = \mathbf{s}^\top \boldsymbol{\theta} - \log(Z(\boldsymbol{\theta})), \qquad \mathbf{s} = \frac{1}{T}\sum_{t=1}^{T} \phi(\mathbf{x}_t)$$

and the log partition function is

$$\log(Z(\boldsymbol{\theta})) = \frac{1}{2}\boldsymbol{\psi}^\top \mathbf{P}^{-1}\boldsymbol{\psi} - \frac{1}{2}\log\det(\mathbf{P}) + \frac{n}{2}\log(2\pi).$$

Related Optimization Problems

- **Sparse multivariate regression with covariance estimation**: LASSO + covariance selection.
  Adam J. Rothman, Elizaveta Levina, and Ji Zhu. "Sparse multivariate regression with covariance estimation." *Journal of Computational and Graphical Statistics* **19**(4), pp. 947-962 (2010).
- **Covariance constrained to a Kronecker product**: Leads to two interconnected covariance selection problems.
  Theodoros Tsiligkaridis, and Alfred O. Hero. "Covariance estimation in high dimensions via kronecker product expansions." *IEEE Transactions on Signal Processing* **61**(21) pp. 5347-5360 (2013).

Applications

Speech Better estimates of covariance for data starved situations.
Weibin Zhang and Pascale Fung "Discriminatively Trained Sparse Inverse Covariance Matrices for Speech Recognition." *IEEE Transactions on Audio, Speech and Language Processing*, **22**(5), pp. 873–882 (2014).

Clustering Clustering of sparse inverse covariances with the clusters being sparse too.
GM Chin, J Nocedal, PA Olsen and SJ Rennie, "Second Order Methods for Optimizing Convex Matrix Functions and Sparse Covariance Clustering." *IEEE Transactions on Audio, Speech, and Language Processing* **21**(11), pp. 2244-2254 (2013).

More applications

Finance Xiang Xuan and Kevin Murphy, "Modeling changing dependency structure in multivariate time series." *ICML* (2007).
Jianqing Fan, Jinchi Lv, and Lei Qi, "Sparse high dimensional models in economics." *Annual review of economics* **3**, pp. 291-317 (2011).

Other Social Co-authorship networks, Web data, climate data analysis, anomaly detection, fMRI brain analysis.

The ℓ_0 problem

- The ℓ_0 problem: Replace penalty with $\mathrm{card}(\mathbf{P}) = |\{P_{ij}|P_{ij} \neq 0\}|$.
- The true sparsity structure can be recovered under the restricted eigenvalue property and enough data. We define d as the maximum non-zero entries of a row in the true covariance matrix (maximum row-cardinality).
  - ℓ_1 problem: Need $\mathcal{O}(d^2 \log(n))$ samples.
  - ℓ_0 problem: Need $\mathcal{O}(d \log(n))$ samples.

PD Ravikumar, G Raskutti, MJ Wainwright, B Yu, "Model Selection in Gaussian Graphical Models: High-Dimensional Consistency of l1-regularized MLE.", *NIPS*, pp. 1329-1336 (2008). NIPS, 1329-1336

The problem is convex because it is the sum of two terms that are convex.

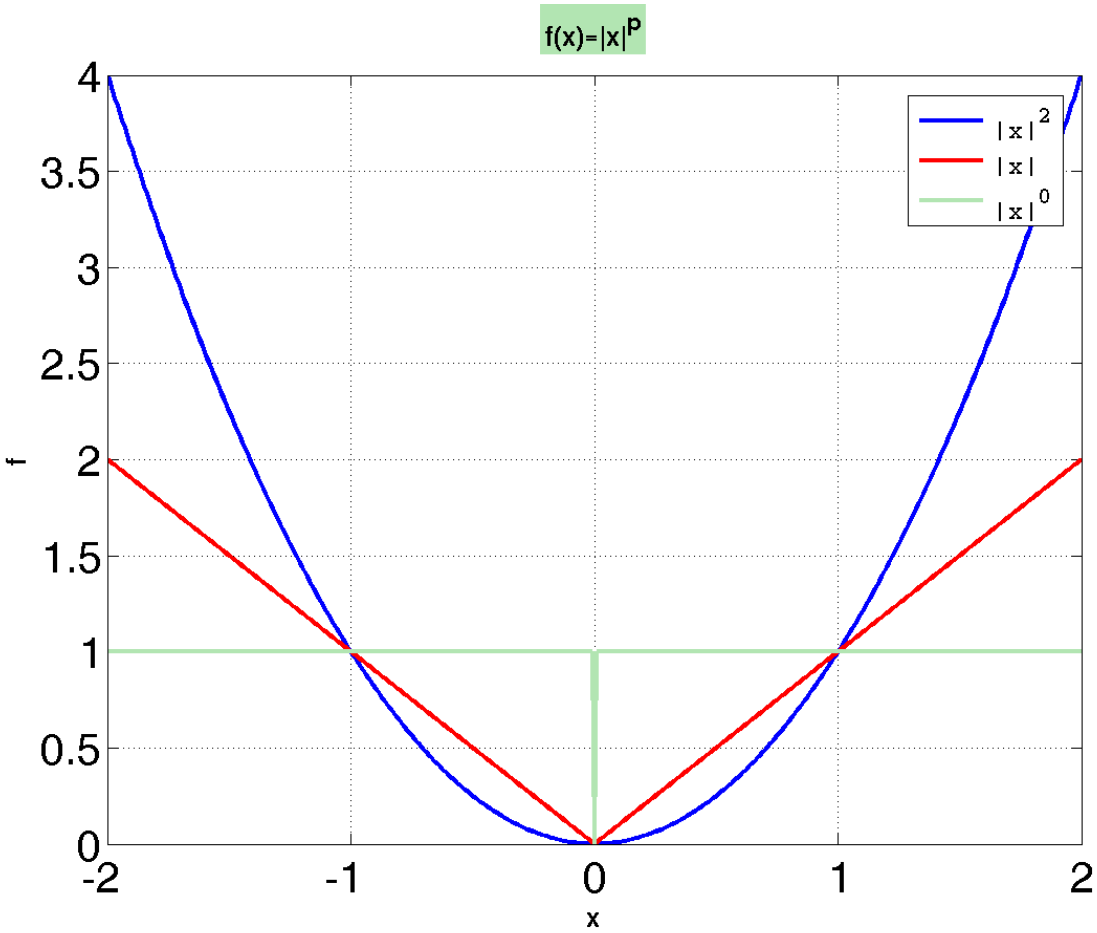
- The log-likelihood of an exponential family is convex, since

$$\frac{\partial^2}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^\top} \log Z(\boldsymbol{\theta}) = \mathrm{Var}[\boldsymbol{\phi}(\mathbf{x})].$$

  This is probably the simplest and most elegant way to prove that $-\log \det(\mathbf{P})$ is convex.

- The penalty term is convex by inspection. All norms are by definition convex.

Consider the function $|x|^p$ for $p \geq 0$. The function is convex if $p \geq 1$ and sparsity promoting if $p \leq 1$.



Norms are convex and sparsity promoting

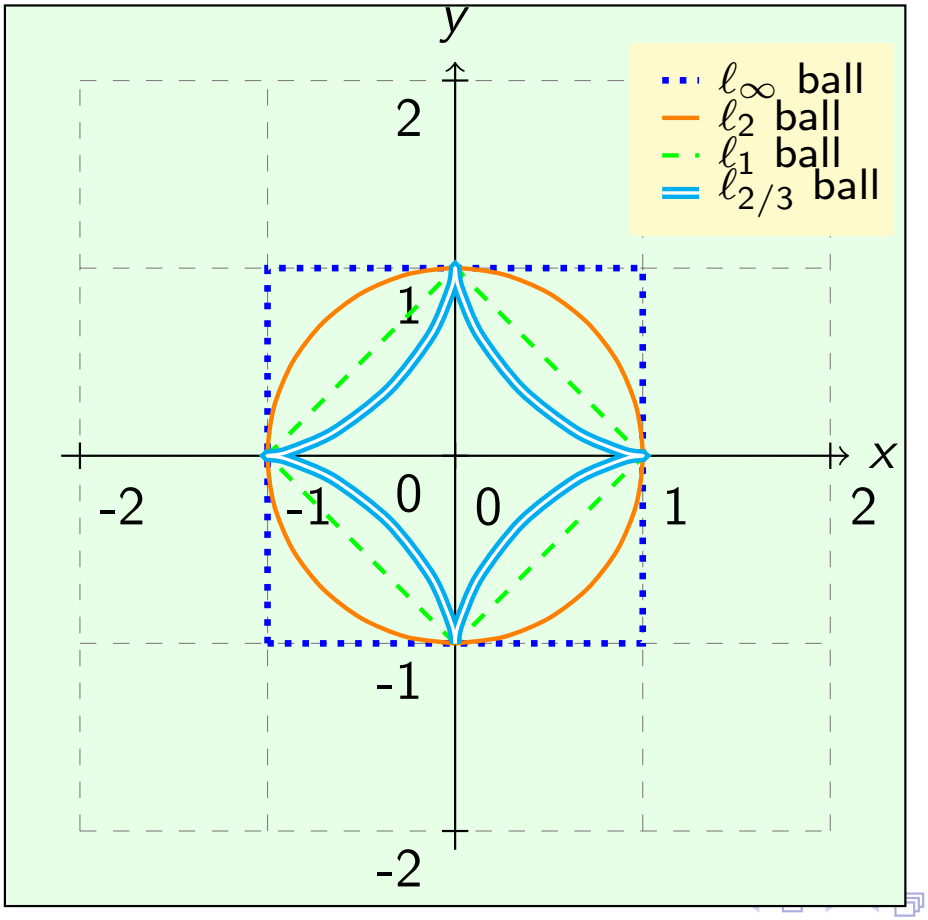
Convexity is insured by the triangle inequality. For any $0 \leq \alpha \leq 1$ with $\alpha + \beta = 1$ we have by the triangle inequality

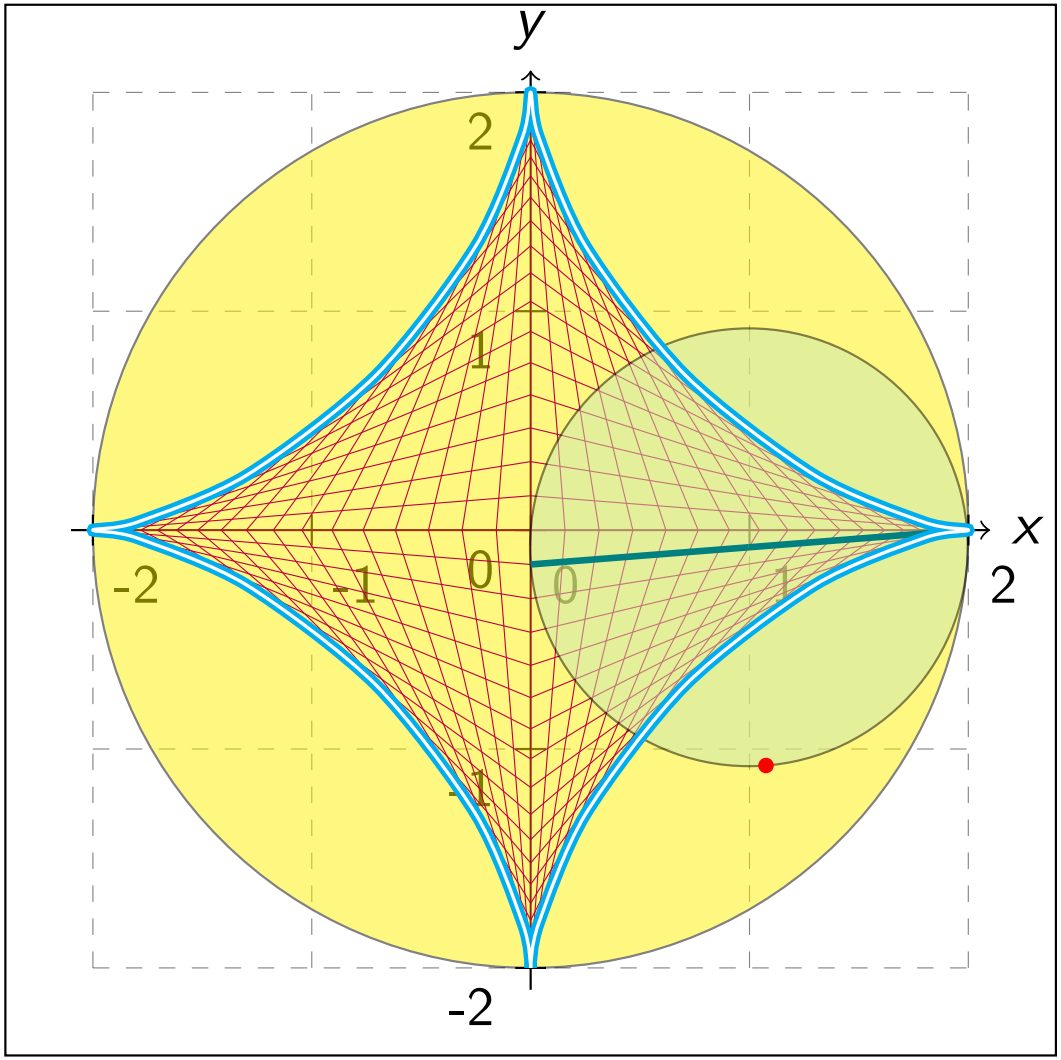
$$\|\alpha \mathbf{x} + \beta \mathbf{y}\| \leq \|\alpha \mathbf{x}\| + \|\beta \mathbf{y}\| = \alpha \|\mathbf{x}\| + \beta \|\mathbf{x}\|.$$

That any norm is sparsity inducing follows by $|x|$ being sparsity inducing, since along any direction $\mathbf{x}$ we have $\|\alpha \mathbf{x}\| = |\alpha| \|\mathbf{x}\|$. $\|\mathbf{x}\|_p$ is not a norm for $p < 1$, and convexity is lost, but it is still sparsity inducing.

How natural are ℓ_p norms?

They may seem unnatural except for $p = 1, 2$ and ∞, but consider the ℓ_p ball for $p = 2/3, 1, 2, \infty$.




y
x
2
1
0
-1
-2
-2
-1
0
1
2

When $\lambda = 0$ the problem becomes equivalent to the maximum likelihood problem, and the solution is $\mathbf{P}^* = \mathbf{S}^{-1}$. Consider the case when $\lambda \neq 0$ and the solution $\mathbf{P}^*$ is not sparse with $\mathbf{Z}^* = \mathrm{sign}(\mathbf{P}^*)$. We then have

$$\begin{aligned} F(\mathbf{P}^*) &= L(\mathbf{P}^*) + \lambda \|\mathrm{vec}(\mathbf{P}^*)\|_1 \\ &= L(\mathbf{P}^*) + \lambda \,\mathrm{trace}(\mathbf{Z}^*\mathbf{P}^*) \\ &= -\log\det(\mathbf{P}^*) + \mathrm{trace}(\mathbf{P}^*(\mathbf{S} + \lambda\mathbf{Z}^*)) \end{aligned}$$

Therefore, the solution is $\mathbf{P}^* = (\mathbf{S} + \lambda\mathbf{Z}^*)^{-1}$. In general if we know $\mathrm{sign}(\mathbf{P}^*)$ the function is smooth in all the non-zero (free) variables and therefore the solution is "easy" to find.

What is an Orthant Face?

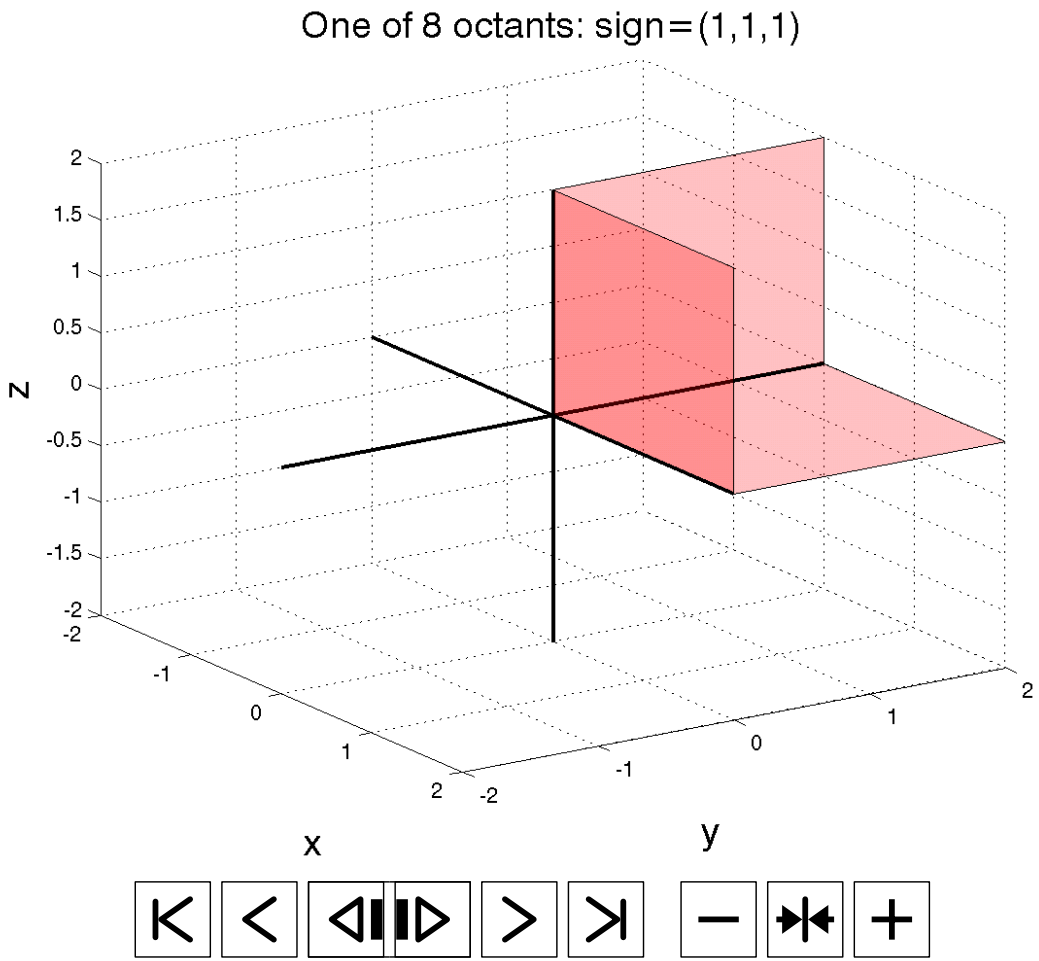
If the value

$$\mathbf{Z} = \text{sign}(\mathbf{P}^*)$$

is known then the problem is smooth for the free variables on the orthant face

$$O(\mathbf{Z}) = \{\mathbf{P} : \text{sign}(\mathbf{Z}) = \epsilon\}$$

The orthant faces are the regions where the sign of $\mathbf{P}$ does not change.



The diagonal elements

Note that the diagonal elements of **P** always have to be strictly positive to ensure the solution is positive definite. Therefore these will always be free variables. Since **P** is symmetric we need only determine the sign of $\binom{n-1}{2}$ variables.

Even if the orthant problem can be solved efficiently there are still $3^{\binom{n-1}{2}}$ orthant faces to search over. This discrete optimization problem of selecting the orthant face seems equally hard. However, if we guide the orthant face search by using the gradient on the orthant surface the discrete problem is aided by the continuous. The rest of the talk will show the structure of the problem and how to do the optimization efficiently.

Dual Formulation

$$
\begin{aligned}
\min_{\mathbf{P}\succ 0} F(\mathbf{P}) &= \min_{\mathbf{P}\succ 0} L(\mathbf{P}) + \lambda \|\mathrm{vec}(\mathbf{P})\|_1 \\
&= \min_{\mathbf{P}\succ 0} L(\mathbf{P}) + \lambda \max_{\|\mathrm{vec}(\mathbf{Z})\|_\infty \le 1} \mathrm{trace}(\mathbf{ZP}) \\
&= \min_{\mathbf{P}\succ 0} \max_{\|\mathrm{vec}(\mathbf{Z})\|_\infty \le 1} -\log\det\mathbf{P} + \mathrm{trace}(\mathbf{PS}) + \lambda\,\mathrm{trace}(\mathbf{ZP}) \\
&= \min_{\mathbf{P}\succ 0} \max_{\|\mathrm{vec}(\mathbf{Z})\|_\infty \le 1} -\log\det\mathbf{P} + \mathrm{trace}(\mathbf{P}(\mathbf{S}+\lambda\mathbf{Z})) \\
&= \max_{\|\mathrm{vec}(\mathbf{Z})\|_\infty \le 1} \min_{\mathbf{P}\succ 0} -\log\det\mathbf{P} + \mathrm{trace}(\mathbf{P}(\mathbf{S}+\lambda\mathbf{Z})) \\
&= \max_{\|\mathrm{vec}(\mathbf{Z})\|_\infty \le 1} \log\det(\mathbf{S}+\lambda\mathbf{Z}) + d
\end{aligned}
$$

At the optimum we have as shown $F(\mathbf{P}^*) = U(\mathbf{Z}^*)$ with the primal and dual variables satisfying the relation

$$\lambda \mathbf{Z}^* + \mathbf{S} - (\mathbf{P}^*)^{-1} = 0$$

and $\mathbf{P}^* \succ 0$ and $\|\mathrm{vec}(\mathbf{Z}^*)\|_\infty \leq 1$

Define the dual function to be

$$U(\mathbf{Z}) = \log\det(\mathbf{S} + \lambda \mathbf{Z}) - d$$

then we have

$$U(\mathbf{Z}) \leq U(\mathbf{Z}^*) = F(\mathbf{P}^*) \leq F(\mathbf{P})$$

so that any pair of matrices $\mathbf{P}$, $\mathbf{Z}$ satisfying $\mathbf{P} \succ 0$ and $\|\mathrm{vec}(\mathbf{Z})\|_\infty \leq 1$ yields an upper an lower bound of the objective at the optimal point.

Note that dual problem is smooth with a box constraint. Box constraint problems can be solved using projected gradients, something that has a long history.

Relationships between the primal and dual

We have that

$$\text{if } \begin{cases} [\mathbf{P}^*]_{ij} = 0 & \text{then} \quad [\mathbf{Z}^*]_{ij} \notin \{-1, 1\} \\ [\mathbf{P}^*]_{ij} > 0 & \text{then} \quad [\mathbf{Z}^*]_{ij} = 1 \\ [\mathbf{P}^*]_{ij} < 0 & \text{then} \quad [\mathbf{Z}^*]_{ij} = -1. \end{cases}$$

The corners of the box corresponds to a non-sparse solution $\mathbf{P}^*$.

The gradient at a point $\mathbf{P}$ of L is as we shall later see given by $\mathbf{G} = \mathrm{vec}(\mathbf{S} - \mathbf{P}^{-1})$. Using this we can get a good approximation to $\mathbf{Z}^*$ if we have a good approximation to $\mathbf{P}^*$. Let $\mathbf{P}$ be an approximation to $\mathbf{P}^*$ and form the value

$$[\mathbf{Z}]_{ij} = \begin{cases} 1 & \text{if } [\mathbf{P}]_{ij} > 0 \\ -1 & \text{if } [\mathbf{P}]_{ij} < 0 \\ -1 & \text{if } [\mathbf{P}]_{ij} = 0 \text{ and } [\mathbf{G}]_{ij} > \lambda \\ 1 & \text{if } [\mathbf{P}]_{ij} = 0 \text{ and } [\mathbf{G}]_{ij} < -\lambda \\ -\frac{1}{\lambda}[\mathbf{G}]_{ij} & \text{if } [\mathbf{P}]_{ij} = 0 \text{ and } |[\mathbf{G}]_{ij}| \leq \lambda. \end{cases}$$

We already know the solution for $\lambda = 0$. What other exact solutions can we find? The following is a list of solutions known to us:

- For λ large the solution is diagonal and known.
- For $n = 2$ we can give the exact solution.
- For λ sufficiently close to zero the solution is not sparse and we can give the exact solution.
- For values of λ where the solution is block-diagonal the blocking can be detected and the exact solution consists of solving each block independently.

For the LASSO problem we can guarantee some features will be zero in the solution without actually solving the problem. This is very useful in reducing the problem size and thus the computational complexity.
Laurent El Ghaoui, Vivian Viallon, and Tarek Rabbani. "Safe feature elimination for the LASSO and sparse supervised learning problems." *arXiv preprint arXiv:1009.4219* (2010).

Some of the ideas from the safe features for LASSO was transferred to covariance selection to automatically detect blocking structure in the solution at a very low computational cost.
Rahul Mazumder and Trevor Hastie, "Exact covariance thresholding into connected components for large-scale graphical lasso." *The Journal of Machine Learning Research* **13**(1), pp. 781-794 (2012).

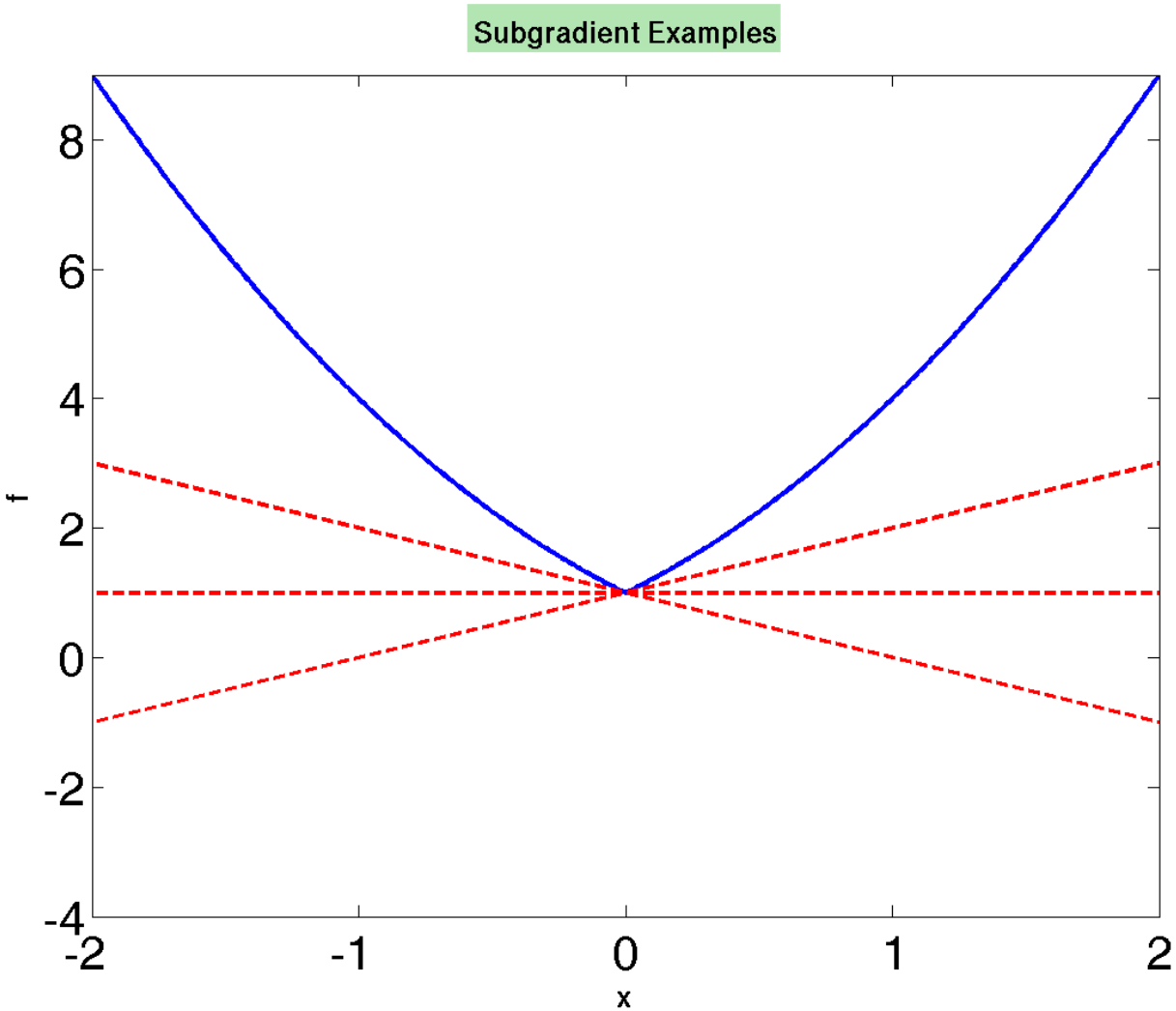
Locating Exact Solutions

The key to finding exact solutions is to use the duality relationship

$$\mathbf{S} - (\mathbf{P}^*)^{-1} + \lambda \mathbf{Z}^* = 0,$$

where $\mathbf{P} \succ 0$ and $\|\mathrm{vec}(\mathbf{Z})\|_\infty \leq 1$. $\mathbf{Z}$ will try to zero out $\mathbf{S}$, and when it can't $\mathbf{P}$ has to fill in the rest. Essentially it is easier to solve the dual problem analytically, since it is smooth, and we can simply guess the solution and verify it for the primal problem.

The key to proving that a solution is correct is the concept of the sub-gradient. A sub-gradient is the slope of a line that touches F at $\mathbf{P}$ and lies below F everywhere. If zero is a sub-gradient then this is the global minimum. The sub-differential is the set of all possible sub-gradients at a point.


Subgradient Examples
8
6
4
2
0
-2
-4
f
-2
-1
0
1
2
x

Derivatives Small and large

- **Frechet Differentiable**: The good old derivative exists (Frechet is the derivative extended to Banach spaces).
- **Gateaux Differentiable**: The **directional derivatives** exists.
- **Sub-differential**: The collection of all **sub-gradients**.
- **Clarke Derivative**: An extension to the sub-differential.
- **Bouligard Derivative**: An extension to directional derivative.
- **Pseudo-gradient**: Not quite a gradient: A few screws short of a hardware store.
- **Weak derivative**: When a function is non-differentiable the weak derivative works "under the integral sign".
- **Financial Derivatives**: The biggest scam of all...

The diagonal solution

Recall that

$$\mathbf{S} - (\mathbf{P}^*)^{-1} + \lambda \mathbf{Z}^* = 0,$$

where $\mathbf{P} \succ 0$ and $\|\mathrm{vec}(\mathbf{Z})\|_\infty \leq 1$. For $\mathbf{Z}$ to zero out the off-diagonal part we must have $\lambda \geq |S_{ij}|$ for all $i \neq j$. Since the sign of the diagonal elements must be positive we have $Z_{ii} = 1$ and we get $\mathbf{P}^* = (\mathrm{diag}(\mathbf{S}) + \lambda \mathbf{I})^{-1}$. This is the solution if and only if $\lambda \geq |S_{ij}|$ for all $i \neq j$.

The solution can be verified by computing the sub-differential and verifying that 0 is a sub-gradient. A more difficult proof uses Hadamard's inequality to verify that $\mathbf{Z}^*$ is the solution to the dual problem.

This simple method of locating zeros only worked because both $\mathbf{P}^*$ and $(\mathbf{P}^*)^{-1}$ had the same sparsity structure (diagonal). This is also the case for block-diagonal solutions.

The 2 by 2 case

A 2×2 matrix is diagonal if the only off-diagonal element is zero. Therefore, we can guess that there are only two kinds of solutions: (1) A diagonal solution when λ is large and (2) a non-sparse solution in the orthant given by $\lambda = 0$, i.e. $\mathbf{Z}^* = \text{sign}(\mathbf{S}^{-1})$. We have

$$\mathbf{S}^{-1} = \frac{1}{\det(\mathbf{S})} \begin{pmatrix} S_{22} & -S_{12} \\ -S_{12} & S_{11}. \end{pmatrix}$$

and therefore

$$\mathbf{Z}^* = \begin{pmatrix} 1 & -\text{sign}(S_{12}) \\ -\text{sign}(S_{12}) & 1 \end{pmatrix}.$$

Complete 2 by 2 solution

Assuming that $n = 2$, $\mathbf{S} \succ 0$ and $\lambda \geq 0$ then the solution to the covariance selection problem is

$$\mathbf{P}^* = \begin{cases} (\mathrm{diag}(\mathbf{S}) + \lambda \mathbf{I})^{-1} & \text{if} \quad \lambda \geq |S_{12}| \\ \begin{pmatrix} S_{11} + \lambda & S_{12}(1 - \lambda/|S_{12}|) \\ S_{12}(1 - \lambda/|S_{12}|) & S_{22} + \lambda \end{pmatrix}^{-1} & \text{if} \quad 0 \leq \lambda < |S_{12}|. \end{cases}$$

Block diagonal solutions

If λ is larger than the absolute value of all the off-block diagonal elements of $\mathbf{S}$ then the solution $\mathbf{P}^*$ is block-diagonal and each block can be found by solving a covariance selection problem.

Let $\lambda = 0.14$

$$\mathbf{S} = \begin{pmatrix} 1.06 & 0.16 & -0.03 & -0.15 & 0.00 & -0.04 & 0.01 & -0.13 & 0.02 \\ 0.16 & 0.85 & -0.11 & -0.15 & -0.01 & 0.00 & 0.03 & 0.00 & 0.01 \\ 0.03 & -0.11 & 1.03 & 0.06 & 0.11 & 0.00 & -0.04 & 0.02 & -0.05 \\ 0.15 & -0.15 & 0.06 & 0.89 & 0.02 & -0.03 & -0.01 & -0.02 & 0.20 \\ 0.00 & -0.01 & 0.11 & 0.02 & 0.93 & 0.04 & -0.01 & -0.02 & 0.14 \\ 0.04 & 0.00 & 0.00 & -0.03 & 0.04 & 1.12 & -0.12 & -0.06 & 0.00 \\ 0.01 & 0.03 & -0.04 & -0.01 & -0.01 & -0.12 & 0.87 & 0.09 & -0.09 \\ 0.13 & 0.00 & 0.02 & -0.02 & -0.02 & -0.06 & 0.09 & 1.03 & 0.02 \\ 0.02 & 0.01 & -0.05 & 0.20 & 0.14 & 0.00 & -0.09 & 0.02 & 1.06 \end{pmatrix}$$

First locate all the elements for which $|S_{ij}| > \lambda$

Let $\lambda = 0.14$

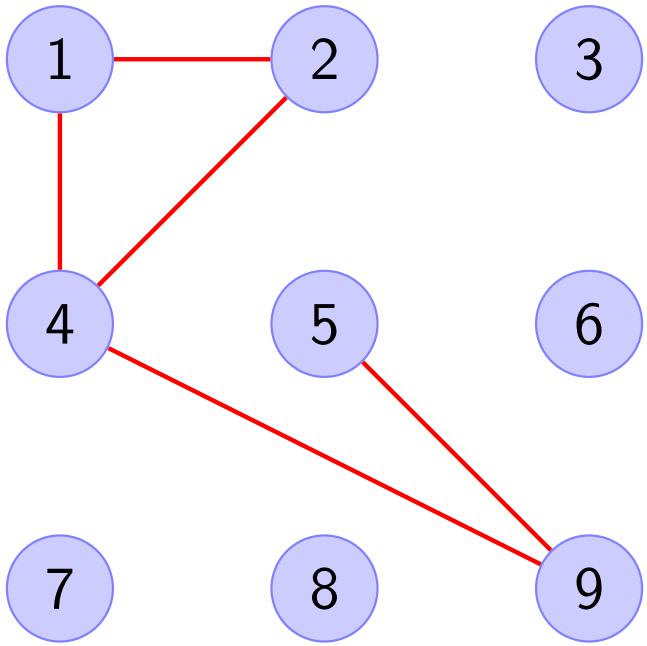
$$\mathbf{S} = \begin{pmatrix} 1.06 & 0.16 & -0.03 & -0.15 & 0.00 & -0.04 & 0.01 & -0.13 & 0.02 \\ 0.16 & 0.85 & -0.11 & -0.15 & -0.01 & 0.00 & 0.03 & 0.00 & 0.01 \\ 0.03 & -0.11 & 1.03 & 0.06 & 0.11 & 0.00 & -0.04 & 0.02 & -0.05 \\ 0.15 & -0.15 & 0.06 & 0.89 & 0.02 & -0.03 & -0.01 & -0.02 & 0.20 \\ 0.00 & -0.01 & 0.11 & 0.02 & 0.93 & 0.04 & -0.01 & -0.02 & 0.14 \\ 0.04 & 0.00 & 0.00 & -0.03 & 0.04 & 1.12 & -0.12 & -0.06 & 0.00 \\ 0.01 & 0.03 & -0.04 & -0.01 & -0.01 & -0.12 & 0.87 & 0.09 & -0.09 \\ 0.13 & 0.00 & 0.02 & -0.02 & -0.02 & -0.06 & 0.09 & 1.03 & 0.02 \\ 0.02 & 0.01 & -0.05 & 0.20 & 0.14 & 0.00 & -0.09 & 0.02 & 1.06 \end{pmatrix}$$

Let $\lambda = 0.14$

$$\mathbf{S} = \begin{pmatrix} \star & \star & & \star & & & & \\ \star & \star & & \star & & & & \\ & & \star & & & & & \\ \star & \star & & \star & & & & \star \\ & & & & \star & & & \star \\ & & & & & \star & & \\ & & & & & & \star & \\ & & & & & & & \star \\ & & & \star & \star & & & \star \end{pmatrix}$$

We find the solution blocks by locating connected components of the graph corresponding to the stars.

The graph corresponding to **S**



A glance at the graph shows that there are 4 single element blocks and one block with 5 elements. We know the solution for the single component blocks and only need to solve the remaining $n = 5$ block.

In general we might want to consider minimizing functions on the form

$$f(\mathbf{x}) + \lambda \|\mathbf{x}\|_1$$

for smooth differentiable convex functions f. In the case when f is a quadratic, this problem is known as the least absolute shrinkage and selection operator (LASSO) problem.
When f is not a quadratic we iteratively solve the LASSO problems

$$\begin{aligned} \hat{\mathbf{x}}_{k+1} &= \operatorname{argmin}_{\mathbf{x}} f(\mathbf{x}_k) + (f'(\mathbf{x}_k))^\top (\mathbf{x} - \mathbf{x}_k) \\ &\quad + \frac{1}{2}(\mathbf{x} - \mathbf{x}_k)^\top f''(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) + \lambda \|\mathbf{x}\|_1 \end{aligned}$$

We call this the Newton-LASSO method.

The line search

Not so fast. The method on the previous page won't work in general. If $f(\mathbf{x}_{k+1}) < f(\mathbf{x}_k)$ then we are OK. This will be the case when we are close to the optimum. Otherwise we need a more conservative approach. For the covariance selection problem the

solution to the quadratic may yield a matrix that is not positive definite. We consider $\mathbf{x} = \mathbf{x}_k + t(\hat{\mathbf{x}}_{k+1} - \mathbf{x}_k)$ and find a value t

that sufficiently decreases the function according to the Armijo rule

$$f(\mathbf{x}_{k+1}) - f(\mathbf{x}_k) \geq \sigma(\mathbf{x}_{k+1} - \mathbf{x}_k)^\top f'(\mathbf{x}_k)$$

and $\sigma \in (0, 1)$.

This condition is not enough to ensure quadratic convergence, but since eventually all steps will be $t = 1$ it's not an issue.

Convergence Properties

When the solution of the LASSO problem is exact the convergence is well understood and given in
P. Tseng and S. Yun. "A coordinate gradient descent method for nonsmooth separable minimization," *Mathematical Programming*, **117**(1), pp. 387-423, (2009).

Recently a proof of convergence for approximate solutions of the LASSO problem has been published:
Richard H. Byrd, Jorge Nocedal and Figen Oztoprak. "An inexact successive quadratic approximation method for convex l-1 regularized optimization," *arXiv preprint arXiv:1309.3529*, (2013).

Convergence Proofs

And here's another:
Katya Scheinberg and Xiaocheng Tang. "Complexity of Inexact Proximal Newton methods," *arXiv preprint arXiv:1311.6547* (2013).

The difficulty with the convergence proofs is that one can not gauge the convergence rate by the magnitude of the gradient/sub-gradient as the function is not continuously differentiable. One way to prove convergence/convergence rate is to compare to an ISTA step.

The first step in developing a Newton-LASSO method is to compute the gradient $\mathbf{g}_k = \mathrm{vec}(L'(\mathbf{P}_k))$ and the Hessian $\mathbf{H}_k = L''(\mathbf{P}_k)$ for our problem. Recall that $L(\mathbf{P}) = -\log\det(\mathbf{P}) + \mathrm{trace}(\mathbf{PS})$, we compute the Taylor expansion for the non-linear term $\log\det(\mathbf{P})$ around $\mathbf{P}_k$.

The Taylor Expansion

Let $\mathbf{P} = \mathbf{P}_k + \mathbf{\Delta}$, $\mathbf{\Delta} = \mathbf{P} - \mathbf{P}_k$ and $\mathbf{X} = \mathbf{P}_k^{-1/2} \mathbf{\Delta} \mathbf{P}_k^{-1/2}$. Let $\{e_i\}_{i=1}^d$ denote the eigenvalues of $\mathbf{X}$, then

$$\begin{aligned}
\log\det\mathbf{P} &= \log\det(\mathbf{P}_k + \mathbf{\Delta}) \\
&= \log\det\mathbf{P}_k + \log\det(\mathbf{I} + \mathbf{P}_k^{-1/2} \mathbf{\Delta} \mathbf{P}_k^{-1/2}) \\
&= \log\det\mathbf{P}_k + \log\det(\mathbf{I} + \mathbf{X}) \\
&= \log\det\mathbf{P}_k + \sum_{i=1}^{d} \log(1 + e_i) \\
&= \log\det\mathbf{P}_k + \sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{k} \sum_{i=1}^{d} e_i^k \\
&= \log\det\mathbf{P}_k + \sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{k} \operatorname{trace}(\mathbf{X}^k)
\end{aligned}$$

The quadratic part

Extracting the linear and quadratic parts in terms of $\mathbf{\Delta}$ we get

$$\begin{aligned}
\log\det\mathbf{P} &= \log\det\mathbf{P}_k + \mathrm{trace}(\mathbf{X}) - \frac{1}{2}\mathrm{trace}(\mathbf{X}^2) + \mathcal{O}(\mathbf{X}^3) \\
&= \log\det\mathbf{P}_k + \mathrm{trace}(\mathbf{P}_k^{-1/2}\mathbf{\Delta}\mathbf{P}_k^{-1/2}) \\
&\quad -\frac{1}{2}\mathrm{trace}(\mathbf{P}_k^{-1/2}\mathbf{\Delta}\mathbf{P}_k^{-1}\mathbf{\Delta}\mathbf{P}_k^{-1/2}) + \mathcal{O}(\mathbf{X}^3) \\
&= \log\det\mathbf{P}_k + \mathrm{trace}(\mathbf{\Delta}\mathbf{P}_k^{-1}) - \frac{1}{2}\mathrm{trace}(\mathbf{\Delta}\mathbf{P}_k^{-1}\mathbf{\Delta}\mathbf{P}_k^{-1}) + \mathcal{O}(\mathbf{X}^3) \\
&= \log\det\mathbf{P}_k + \mathrm{vec}^\top(\mathbf{\Delta})\mathrm{vec}(\mathbf{P}_k^{-1}) \\
&\quad -\frac{1}{2}\mathrm{vec}^\top(\mathbf{\Delta})\mathrm{vec}(\mathbf{P}_k^{-1}\mathbf{\Delta}\mathbf{P}_k^{-1}) + \mathcal{O}(\mathbf{X}^3) \\
&= \log\det\mathbf{P}_k + \mathrm{vec}^\top(\mathbf{\Delta})\mathrm{vec}(\mathbf{P}_k^{-1}) \\
&\quad -\frac{1}{2}\mathrm{vec}^\top(\mathbf{\Delta})(\mathbf{P}_k^{-1}\otimes\mathbf{P}_k^{-\top})\mathrm{vec}(\mathbf{\Delta}) + \mathcal{O}(\mathbf{X}^3)
\end{aligned}$$

It follows that

$$\mathbf{g}_k = \mathrm{vec}(\mathbf{S} - \mathbf{P}_k^{-1}) \qquad \mathbf{H}_k = \mathbf{P}_k^{-1} \otimes \mathbf{P}_k^{-1}.$$

The fact that the Hessian is a Kronecker product can be used to make the computation of the Newton direction and the solution to the Newton-LASSO problem more efficient. Also, we never need to explicitly instantiate the Hessian.

Definition (Kronecker Products)

For matrices $\mathbf{A} \in \mathbb{R}^{m_1 \times n_1}$ and $\mathbf{B} \in \mathbb{R}^{m_2 \times n_2}$ we define the Kronecker product $\mathbf{A} \otimes \mathbf{B} \in \mathbb{R}^{(m_1 m_2) \times (n_1 n_2)}$ to be

$$\mathbf{A} \otimes \mathbf{B} = \begin{pmatrix} a_{11}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ \vdots & \vdots & \vdots \\ a_{m1}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{pmatrix}.$$

It can be verified that this definition is equivalent to $(\mathbf{A} \otimes \mathbf{B})_{(i-1)m_2+j,(k-1)n_2+l} = a_{ik} b_{jl}$, which we simply write $(\mathbf{A} \otimes \mathbf{B})_{(ij)(kl)} = a_{ik} b_{jl}$.

Theorem

The following equations gives identities for multiplying, transposing, inverting and computing the trace of Kronecker products.

$$
\begin{aligned}
(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) &= (\mathbf{AC}) \otimes (\mathbf{BD}) \\
(\mathbf{A} \otimes \mathbf{B})^\top &= \mathbf{A}^\top \otimes \mathbf{B}^\top \\
(\mathbf{A} \otimes \mathbf{B})^{-1} &= \mathbf{A}^{-1} \otimes \mathbf{B}^{-1} \\
\mathbf{I}_m \otimes \mathbf{I}_n &= \mathbf{I}_{mn} \\
(\mathbf{B}^\top \otimes \mathbf{A})\mathrm{vec}(\mathbf{X}) &= \mathrm{vec}(\mathbf{AXB}) \\
\mathrm{trace}(\mathbf{A} \otimes \mathbf{B}) &= \mathrm{trace}(\mathbf{A})\,\mathrm{trace}(\mathbf{B})
\end{aligned}
$$

More on the linesearch

Two issues on the linesearch we have not dealt with. Both are borrowed from the creators of QUIC.

1. What line-search method do we use?
2. How do we ensure positive definiteness?

We used the backtracking linesearch with $t = 1, 1/2, 1/4, \ldots$. The reasoning is two-fold. Firstly, since $t = 1$ will be the predominant choice, the line-search is not dominating the compute time. Secondly, it is simple to implement and the accuracy of the line search is not very important as long as convergence is guaranteed.

Positive definiteness

First of all how do we check for positive definiteness?

- Find the smallest eigenvalue and check that it is positive.
- Do a Cholesky decomposition $\mathbf{P} = \mathbf{L}\mathbf{L}^\top$. If the decomposition succeeds and $L_{ii} \neq 0$ then $\mathbf{P} \succ 0$, otherwise $\mathbf{P} \not\succ 0$.

The first method is perhaps more reliable and gives more information. But the second method is by far the fastest.

The line-search interval

In the linesearch we must ensure that t is chosen so that $\mathbf{P}_{k+1} = \mathbf{P}_k + t\mathbf{V}$ is positive definite. We can do that in two ways

- Find the smallest $t > 0$ such that $\det(\mathbf{P}_k + t\mathbf{V}) = 0$ by solving the generalized eigenvalue problem $\mathbf{P}_k\mathbf{x} = \lambda\mathbf{V}\mathbf{x}$.
- If the Cholesky decomposition succeeds for a particular t then $\mathbf{P}_{k+1} \succ 0$.

Along with the line-search strategy outlined QUIC had two more important innovations

- The LASSO problem was solved using coordinate descent. Each variable can be solved very efficiently, by using the structure of the Hessian. In total it uses only $\mathcal{O}(n|\mathcal{F}|)$ operations, where $\mathcal{F}$ are the free variables, per sweep over the variables.
- By starting the process from a sparse (diagonal) matrix, really sparse solutions can be found extremely efficiently.

ISTA

The Iterative Shrinkage Thresholding Algorithm (ISTA) minimizes

$$f(\mathbf{x}) + \lambda \|\mathbf{x}\|_1$$

when f is a convex quadratic.

Ingrid Daubechies, Michel Defrise, and Christine De Mol. "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Communications on pure and applied mathematics* **57**(11), pp. 1413-1457 (2004).

ISTA

The ISTA method iterates

$$\mathbf{x}_{i+1} = S_{\lambda/c}\left(\mathbf{x}_i - \frac{1}{c}\nabla f(\mathbf{x}_i)\right),$$

where $c\mathbf{I} - f''(\mathbf{x}) \succ 0$ and S_λ is the Donoho-Johnstone shrinkage operator applied to each coordinate

$$S_\lambda(x) = \begin{cases} x - \lambda & \text{if} \quad x > \lambda \\ 0 & \text{if} \quad |x| \leq \lambda \\ x + \lambda & \text{if} \quad x < -\lambda. \end{cases}$$

FISTA

The Fast Iterative Shrinkage Thresholding Algorithm (FISTA) method is a method that converges significantly faster than ISTA at very little computational overhead.

Amir Beck and Marc Teboulle. "A fast iterative shrinkage-thresholding algorithm for linear inverse problems." *SIAM Journal on Imaging Sciences* **2**(1), pp. 183-202, (2009).

FISTA

FISTA first takes an ISTA step

$$\hat{\mathbf{x}}_i = S_{\lambda/c}\left(\mathbf{x}_i - \frac{1}{c}\nabla f(\mathbf{x}_i)\right),$$

then a Nesterov acceleration is applied to give

$$\mathbf{x}_{i+1} = \hat{\mathbf{x}}_i + \frac{t_i - 1}{t_{i+1}}(\hat{\mathbf{x}}_i - \hat{\mathbf{x}}_{i-1})$$

where

$$\hat{\mathbf{x}}_1 = \hat{\mathbf{x}}_0, t_1 = 1, t_{i+1} = \frac{1 + \sqrt{1 + 4t_i^2}}{2}.$$

ISTA for Covariance Selection

If we apply the ISTA iteration to the LASSO subproblem of the covariance selection problem we get the elegant iteration:

$$\begin{aligned} \mathbf{x}_i &= S_{\lambda/c}\left(\mathrm{vec}(\hat{\mathbf{X}}_i) - \frac{1}{c}\left(\mathbf{g}_k + \mathbf{H}_k \mathrm{vec}(\hat{\mathbf{X}}_i - \mathbf{P}_k)\right)\right) \\ &= S_{\lambda/c}\left(\frac{1}{c}\mathrm{vec}(-\mathbf{S} + 2\mathbf{P}_k^{-1} - \mathbf{P}_k^{-1}\hat{\mathbf{X}}_i\mathbf{P}_k^{-1}) + \mathrm{vec}(\hat{\mathbf{X}}_i)\right), \end{aligned}$$

The inverse matrix $\mathbf{P}_k^{-1}$ can be precomputed and stored for the iterations $\mathbf{x}_0 \to \mathbf{x}_1 \ldots \to \mathbf{x}_i \ldots$.
After the ISTA/FISTA iteration is performed we perform a back-tracking line-search. as described earlier, to determine $\mathbf{P}_{k+1}$.

Algorithm

1. Compute starting point $\mathbf{P}_0 = \mathrm{diag}^{-1}(\lambda + \mathrm{diag}(\mathbf{S}))$, $k = 0$.
2. Stop if the minimum sub-gradient norm is smaller than ϵ
3. Solve the LASSO sub problem using FISTA. Call the approximate solution $\mathbf{X}_{k+1}$
4. Find $\mathbf{P}_{k+1}$ by a backtracking line search from $\mathbf{P}_k$ to $\mathbf{X}_{k+1}$.
5. $k \leftarrow k + 1$ and go to step 2.

Active and Free Variables

We divide the variables in the following two groups

Active Variables The active constraints/variables are the variables whose values we fix at 0. We denote the set of active variables as $\mathcal{A}$.

Free Variables The free variables are the variables whose values are not fixed at 0. We denote the set of free variables as $\mathcal{F}$.

An orthant face naturally divides the variables into active and free variables, where the sign of each free variable is fixed according to the orthant face.

We claimed earlier that optimizing over an orthant face was "simple". How to do this?

Choosing the Orthant Face

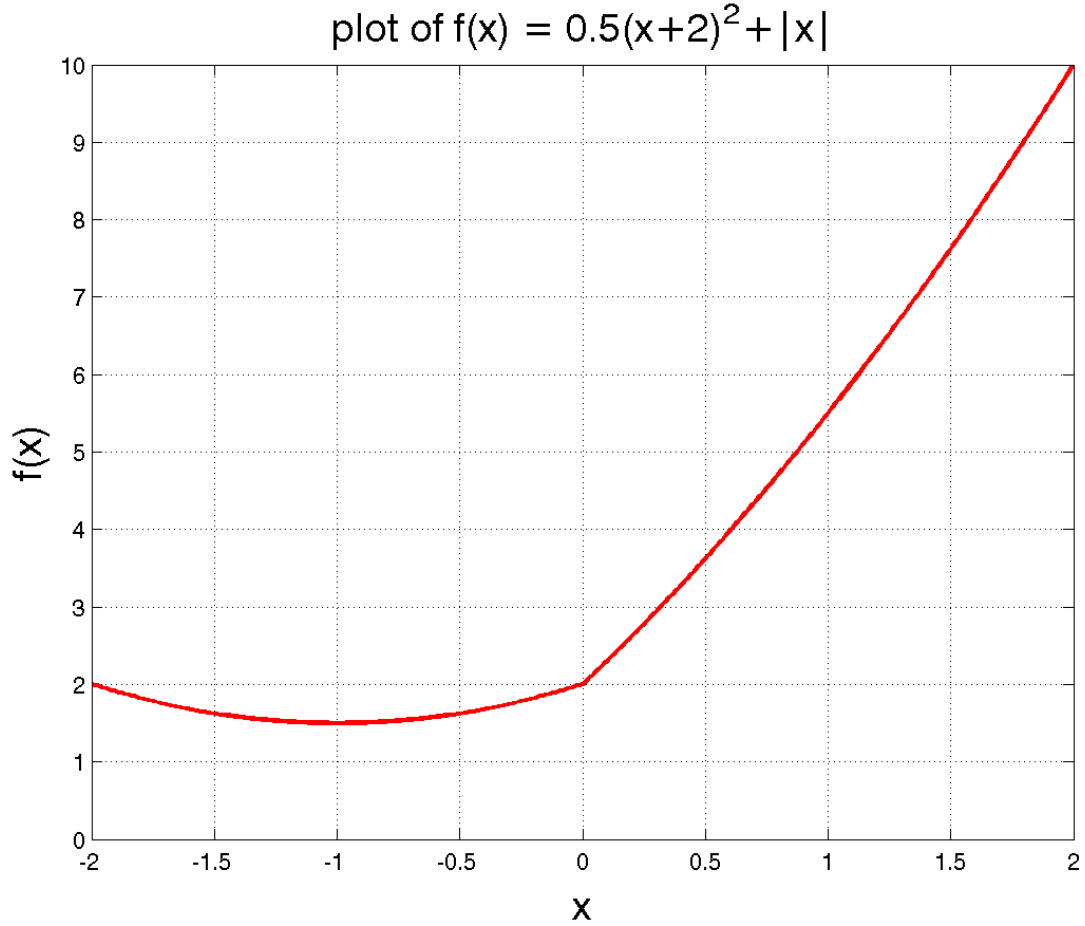
If we are at a given point $\mathbf{P}_k$ and consider the optimization in an orthant face containing $\mathbf{P}_k$ there may be several choices of orthant faces. For each value $[\mathbf{P}_k]_{ij} = 0$ we can choose the corresponding orthant-sign to be negative, zero or positive.
Consider an infinitesimal change of $[\mathbf{P}_k]_{ij}$ to decide the sign.

- If a small positive change reduces the function value, make the sign positive. This happens if $\frac{\partial L}{\partial P_{ij}} > \lambda$.
- If a small negative change reduces the function then the sign is negative. This happens if $\frac{\partial L}{\partial P_{ij}} < -\lambda$.
- Finally if neither a positive nor a negative change reduces the function value then we make the sign zero. This happens if $\left|\frac{\partial L}{\partial P_{ij}}\right| \leq \lambda$.

As an example consider the function $f(x) = \frac{1}{2}(x-a)^2 + |x|$ at $x = 0$ for various values of a:

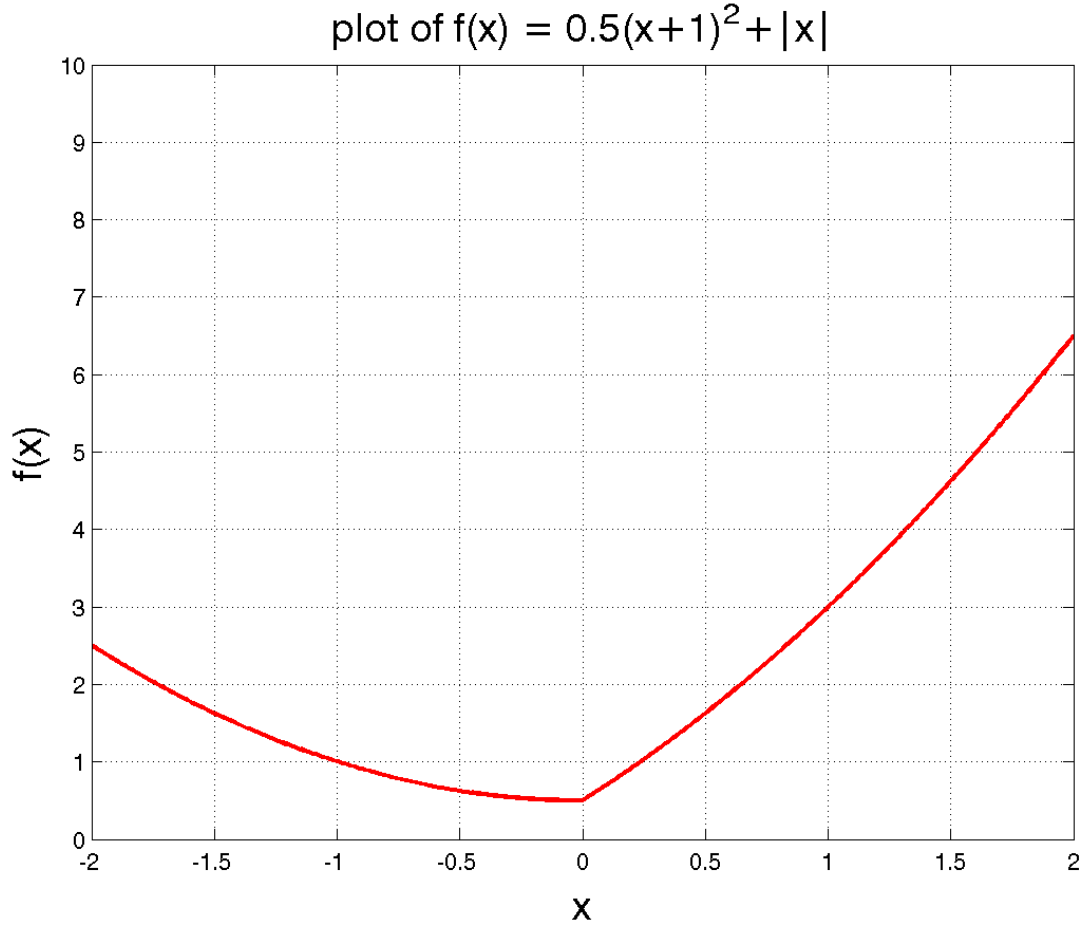
As an example consider the function $f(x) = \frac{1}{2}(x-a)^2 + |x|$ at $x = 0$ for various values of a:

For $a < -1$ the minimum occurs for $x < 0$:



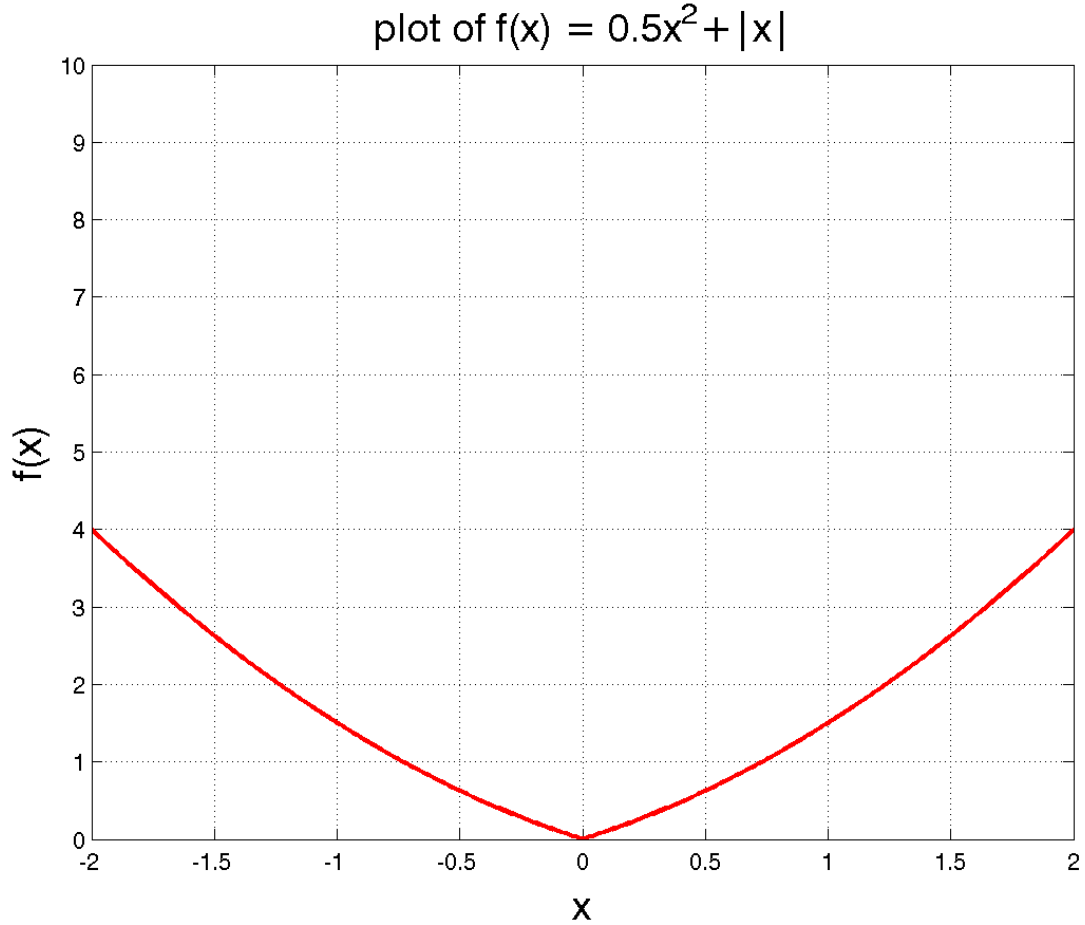
As an example consider the function $f(x) = \frac{1}{2}(x-a)^2 + |x|$ at $x = 0$ for various values of a:

For $|a| <= 1$ the minimum occurs when $x = 0$:



As an example consider the function $f(x) = \frac{1}{2}(x-a)^2 + |x|$ at $x = 0$ for various values of a:

For $|a| <= 1$ the minimum occurs when $x = 0$:

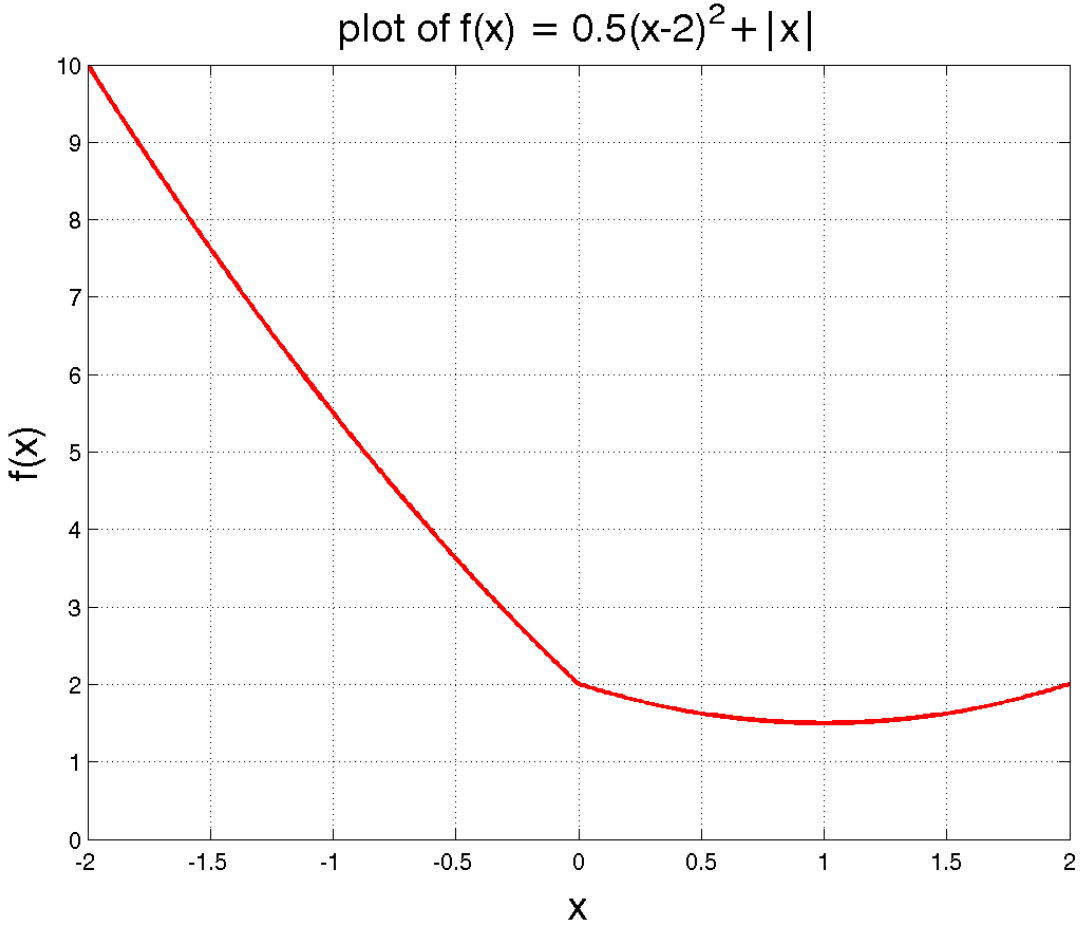


As an example consider the function $f(x) = \frac{1}{2}(x-a)^2 + |x|$ at $x = 0$ for various values of a:

For $|a| <= 1$ the minimum occurs when $x = 0$:

As an example consider the function $f(x) = \frac{1}{2}(x-a)^2 + |x|$ at $x = 0$ for various values of a:

For $a > 1$ the minimum occurs for $x > 0$:



Orthant Indicator

We defined the orthant indicator $\mathbf{Z}_k$ to be

$$[\mathbf{Z}_k]_{ij} = \begin{cases} 1 & \text{if } [\mathbf{P}_k]_{ij} > 0 \\ -1 & \text{if } [\mathbf{P}_k]_{ij} < 0 \\ -1 & \text{if } [\mathbf{P}_k]_{ij} = 0 \text{ and } [\mathbf{G}_k]_{ij} > \lambda \\ 1 & \text{if } [\mathbf{P}_k]_{ij} = 0 \text{ and } [\mathbf{G}_k]_{ij} < -\lambda \\ 0 & \text{if } [\mathbf{P}_k]_{ij} = 0 \text{ and } |[\mathbf{G}_k]_{ij}| \leq \lambda. \end{cases}$$

The 0 ensure the active variables do not move away from 0. The dual variable took a different value $\frac{1}{\lambda}[\mathbf{G}_k]_{ij}$ here.
The value $\mathbf{G}_k + \lambda \mathbf{Z}_k$ is the steepest descent direction at the point $\mathbf{P}_k$, and we refer to it as the pseudo-gradient. We consider minimizing $L(\mathbf{P}) + \lambda \operatorname{trace}(\mathbf{PZ})$ on the orthant face in place of $F(\mathbf{P})$.

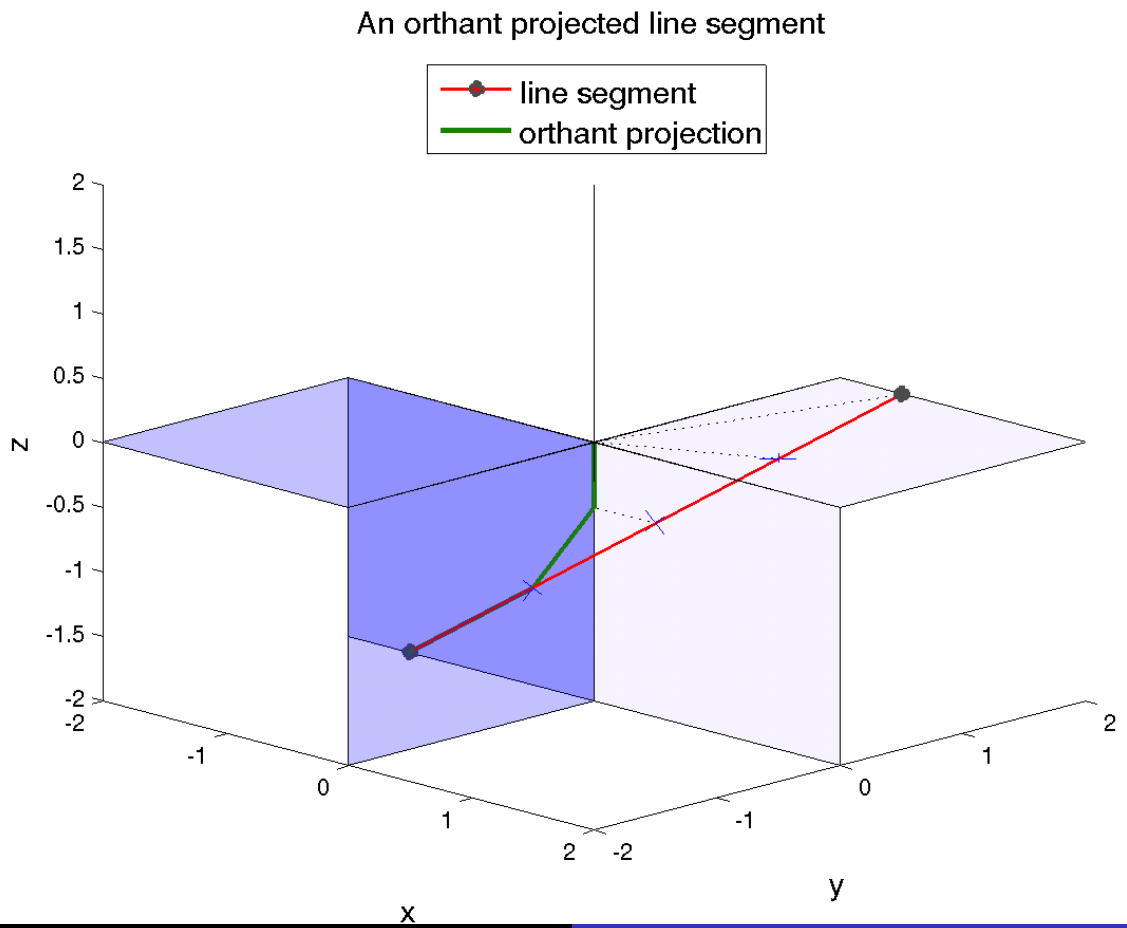
When the Newton step corresponding to the quadratic approximation on the orthant face is outside the orthant face, we must make the decision as to whether or not to allow the search to leave the orthant.

Leaving the orthant face leads to complications. The Newton direction is not guaranteed to be a descent direction anymore, but this can be fixed with something known as pseudo gradient alignment. Also, we need to ensure that we enforce sparsity whenever possible.

Not leaving the orthant face is simpler. However, only considering the line segment insde the orthant face leads to many small steps. Typically only one coordinate will be made sparse per line-step and this may mean millions of line-searches for problems where $n > 1000$. We need a strategy to allow many variables to become sparse at once – a sparsity acceleration if you will. We project the line-segment using the orthant-projection

$$\Pi(\mathbf{P}_{ij}) = \begin{cases} \mathbf{P}_{ij} & \text{if } \operatorname{sign}(\mathbf{P}_{ij}) = \operatorname{sign}(\mathbf{Z}_k)_{ij} \\ 0 & \text{otherwise.} \end{cases}$$

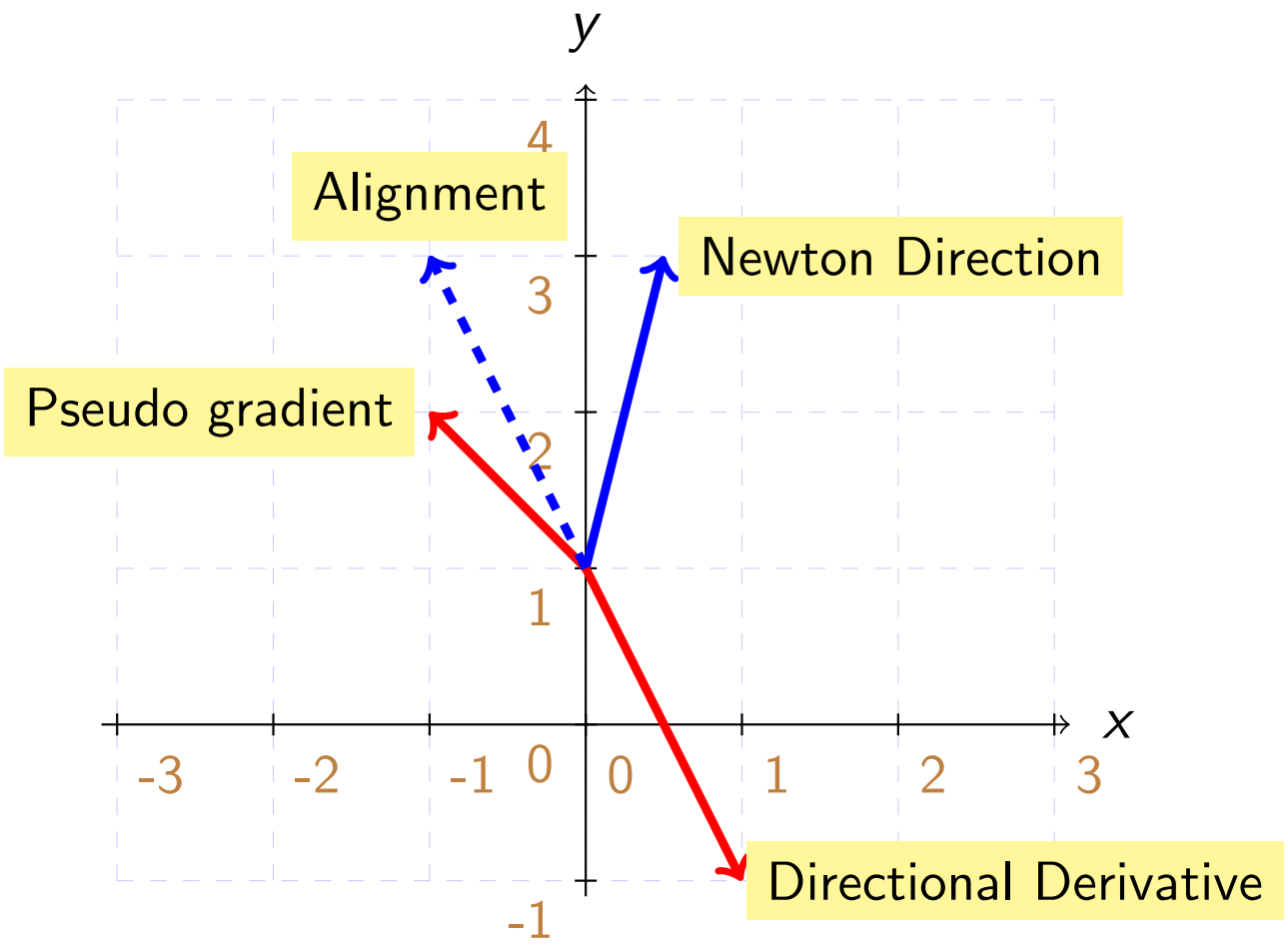
Figen tried line search strategies both confined to and not confined to the orthant. She also tried different strategies for sparsity acceleration. The orthant projection scheme was best most of the time, and also happened to be the simplest to implement!



The OWL package that optimizes functions with an ℓ_1 penalty uses a procedure called gradient alignment.
Galen Andrew and Jianfeng Gao. "Scalable training of L1-regularized log-linear models," *ICML*, (2007).

Gradient alignment is not needed with the orthant projection method.

Pseudo Gradient Alignment



The reduced quadratic

We use the notation $\mathbf{p}_k = \mathrm{vec}(\mathbf{P}_k) = \begin{pmatrix} p_{k\mathcal{F}} \\ p_{k\mathcal{A}} \end{pmatrix} = \begin{pmatrix} p_{k\mathcal{F}} \\ 0 \end{pmatrix}$. Recall that the piecewise quadratic approximation to F is

$$q_k(\mathbf{P}) = L(\mathbf{P}_k) + \mathbf{g}_k^\top(\mathbf{p} - \mathbf{p}_k) + \frac{1}{2}(\mathbf{p} - \mathbf{p}_k)^\top \mathbf{H}_k(\mathbf{p} - \mathbf{p}_k) + \lambda\|\mathbf{p}\|_1$$

If we constrain the model to the $\mathbf{Z}_k$ orthant face we get

$$q_k(\mathbf{P}) = L(\mathbf{P}_k) + \mathbf{g}_k^\top(\mathbf{p} - \mathbf{p}_k) + \frac{1}{2}(\mathbf{p} - \mathbf{p}_k)^\top \mathbf{H}_k(\mathbf{p} - \mathbf{p}_k) + \lambda\mathbf{p}^\top\mathbf{z}_k$$

subject to $\mathrm{sign}(\mathbf{p}) = \mathbf{z}_k$. Finally, if we substitute in $\mathbf{p}_{k\mathcal{A}} = 0$ and drop the constraints and the constant we get the reduced quadratic

$$Q_{\mathcal{F}}(\mathbf{p}_{\mathcal{F}}) = \mathbf{g}_{k\mathcal{F}}^\top(\mathbf{p}_{\mathcal{F}} - \mathbf{p}_{k\mathcal{F}}) + \frac{1}{2}(\mathbf{p}_{\mathcal{F}} - \mathbf{p}_{k\mathcal{F}})^\top \mathbf{H}_{k\mathcal{F}}(\mathbf{p}_{\mathcal{F}} - \mathbf{p}_{k\mathcal{F}}) + \lambda\mathbf{p}_{\mathcal{F}}^\top\mathbf{z}_{k\mathcal{F}}.$$

Here $\mathbf{H}_{k\mathcal{F}}$ equals $\mathbf{H}_k = \mathbf{P}_k^{-1} \otimes \mathbf{P}_k$ with the rows and columns corresponding to $\mathcal{A}$ removed.

The solution to the reduced quadratic can be seen to be

$$\mathbf{p}^*_{\mathcal{F}} = \mathbf{p}_{k\mathcal{F}} + \mathbf{H}^{-1}_{k\mathcal{F}}(\lambda \mathbf{z}_{k\mathcal{F}} - \mathbf{g}_{k\mathcal{F}}).$$

We need a quick way to compute $\mathbf{p}^*_{\mathcal{F}}$ without storing $\mathbf{H}^{-1}_{k\mathcal{F}}$.

For $\mathcal{A} = \emptyset$ the computation becomes trivial:
$\mathbf{P}^* = \mathbf{P}_k - \mathbf{P}_k(\lambda \mathbf{Z}_k - \mathbf{G}_k)\mathbf{P}_k$.

Observation: We can do fast multiplication ($\mathcal{O}(n|\mathcal{F}|)$) by $\mathbf{H}_{k\mathcal{F}}$ by lifting, multiplying by $\mathbf{H}_k$ and then projecting:

$$\mathbf{H}_{k\mathcal{F}}\mathbf{x}_{\mathcal{F}} = [\mathbf{H}_k \begin{pmatrix} \mathbf{x}_{\mathcal{F}} \\ \mathbf{0} \end{pmatrix}]_{\mathcal{F}} = \left[\mathbf{P}_k^{-1} \mathrm{mat} \begin{pmatrix} \mathbf{x}_{\mathcal{F}} \\ \mathbf{0} \end{pmatrix} \mathbf{P}_k^{-1}\right]_{\mathcal{F}}.$$

The conjugate gradient algorithm is an iterative procedure to find the solution to $\mathbf{Ax} = \mathbf{b}$, when $\mathbf{A} \succ 0$.

At iteration k the conjugate gradient algorithm finds the projection of the solution onto the Krylov subspace $\text{span}\{\mathbf{b}, \mathbf{Ab}, \ldots, \mathbf{A}^{k-1}\mathbf{b}\}$.

In each iteration of the conjugate gradient algorithm we compute a matrix–vector product $\mathbf{Ay}_k$. This is the most expensive step.

The conjugate Gradient Algorithm

Initialize: $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\mathbf{y}_0 = \mathbf{x}_0$, $k = 0$

`while` $r_k > \epsilon$ `do`

$$\begin{aligned}
\alpha_k &= \frac{\mathbf{r}_k^\top \mathbf{r}_k}{\mathbf{y}_k^\top \mathbf{A}\mathbf{y}_k} \\
\mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{y}_k \\
\mathbf{r}_{k+1} &= \mathbf{r}_k - \alpha_k \mathbf{A}\mathbf{y}_k \\
\beta_k &= \frac{\mathbf{r}_{k+1}^\top \mathbf{r}_{k+1}}{\mathbf{r}_k^\top \mathbf{r}_k} \\
\mathbf{y}_{k+1} &= \mathbf{r}_{k+1} + \beta_k \mathbf{y}_k \\
k &\leftarrow k+1
\end{aligned}$$

LBFGS (Limited memory Broyden–Fletcher–Goldfarb–Shannon) is considered the all-around best method to minimize non-linear functions. We used it to solve the reduced quadratic. The Hessian $\mathbf{H}_{\mathcal{F}}$ is replaced by a limited memory BFGS matrix $\mathbf{B}_{\mathcal{F}}$. Instead of using the properties of $\mathbf{H}_{\mathcal{F}}$ to efficiently compute the Newton step, we use the properties of the approximation $\mathbf{B}_{\mathcal{F}}$.

The OWL package does something similar, but used the full quadratic instead of the reduced quadratic. Which approach is best depends on the sparsity of the solution.

So Long And Thanks For All The Fish