Estimating the Inverse Covariance Matrix of Independent Multivariate Normally Distributed Random Variables

Dominique Brunet, Hanne Kekkonen, Vitor Nunes, Iryna Sivak

Fields-MITACS Thematic Program on Inverse Problems and Imaging
Group on Bayesian/Statistical Inverse Problems

July 27, 2012

Menu

Bayes' formula brings a priori information and measured data together

Statistical Inversion

We consider the linear measurement model

$$m = Ax + \varepsilon$$

where $x \in \mathbf{R}^p$ and $m \in \mathbf{R}^p$ are treated as random variables.

Bayes' formula

Bayes' formula gives us the posterior distribution $\pi(x \mid m)$:

$$\pi(x \mid m) \propto \pi(m \mid x)\pi(x)$$

We recover x as a point estimate from $\pi(x \mid m)$.

Likelihood distribution describes the measurement

If we assume that the noise ε is Gaussian

$$\varepsilon_k \sim \mathcal{N}_p(\mu, \sigma^2), \qquad 1 \leq k \leq p$$

we can write the likelihood distribution which measures data misfit as

$$\pi(m \mid x) = \pi_\varepsilon(\|Ax - m\|) = \frac{1}{\sigma\sqrt{(2\pi)}} \exp\left(- \frac{1}{2\sigma^2} \|Ax - m\|_2^2 \right).$$

Prior distribution contains all other information about unknown x

- The prior distribution shouldn't depend on the measurement.
- The prior distribution should assign a clearly higher probability to those values of x that we expect to see than to unexpected x.
- Designing a good prior distributions is one of the main difficulties in statistical inversion.

Gaussian prior distribution

Gaussian model

If we assume that also x is Gaussian

$$x \sim \mathcal{N}_p(\mu, \Sigma)$$

we can write

$$\pi(x) = \frac{(\det(\Sigma^{-1}))^{1/2}}{(2\pi)^{p/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

Estimating inverse covariance matrix Σ^{-1}

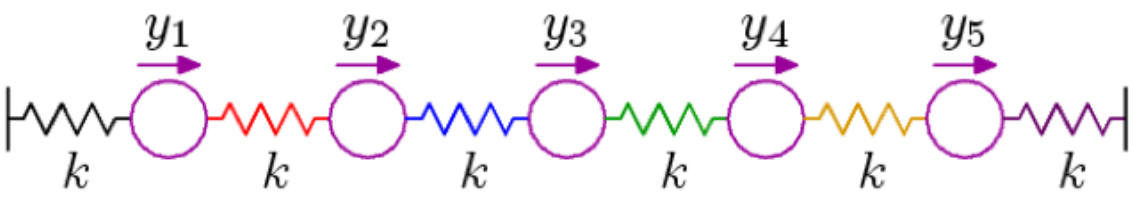
We consider the problem of finding a good estimator for inverse covariance matrix Σ^{-1} with a constraint that certain given pairs of variables are conditionally independent.

Conditional independence constraints describe the sparsity pattern of the inverse covariance matrix Σ^{-1}, zeros showing the conditional independence between variables.

Why inverse covariance matrix?

Simple example

We can think a line of objects which are connected by a spring.



If we move the leftmost object y_1 also the rightmost object y_5 of the line will move so clearly y_1 and y_5 are not independent.

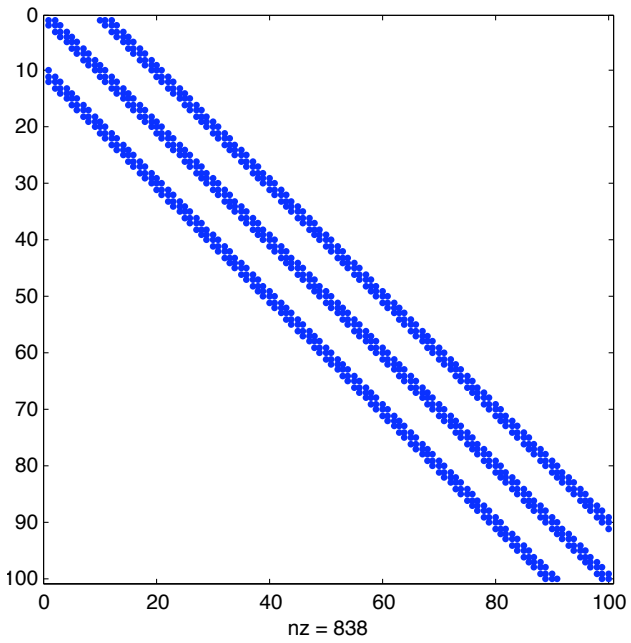
However y_1 and y_5 are conditionally independent given all the other variables. This is because given the information of how y_2 moves knowing y_5 gives us no further information about y_1. In fact y_i is conditionally dependent only of its neighbours.

$$\mathbf{K}^{-1} = \frac{k}{T}\begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix} \qquad \mathbf{K} = \frac{T}{k}\begin{bmatrix} 0.83 & 0.67 & 0.50 & 0.33 & 0.17 \\ 0.67 & 1.33 & 1.00 & 0.67 & 0.33 \\ 0.50 & 1.00 & 1.50 & 1.00 & 0.50 \\ 0.33 & 0.67 & 1.00 & 1.33 & 0.67 \\ 0.17 & 0.33 & 0.50 & 0.67 & 0.83 \end{bmatrix}$$

What does this have to do with pictures?

Neighbourhood -Rule

We are interested in exploring the assumption that every pixel is conditionally dependent only of its neighbours.



In this case the inverse covariance matrix has non zero elements only on nine 'diagonals'.

Log-likelihood function

Likelihood function

We want to estimate parameters Σ^{-1} and μ of $\mathcal{N}_p(\mu, \Sigma)$, based on N independent samples x_i. The likelihood function is

$$\prod_{i=1}^{N} \pi(x_i \mid \Sigma, \mu) = \frac{\det \Sigma^{-\frac{N}{2}}}{(2\pi)^{\frac{Np}{2}}} \exp\left\{ -\frac{1}{2} \sum_{j=1}^{N} (x_j - \mu)^T \sigma^{-1} (x_j - \mu) \right\}$$

Log-likelihood function

Log-likelihood function of data is, up to a constant

$$L(\Sigma^{-1}, \mu) = N \log \det \Sigma^{-1} - \sum_{j=1}^{N} (x_j - \mu)^T \Sigma^{-1} (x_j - \mu)$$

How to obtain sparse Σ^{-1}

maximum likelihood estimate

Maximizing the log-likelihood with respect to Σ^{-1} leads to the maximum likelihood estimate $\widehat{\Sigma}^{-1} = S^{-1}$ which isn't usually sparse (here S is the sample covariance obtained from the data). Also when $p > N$, S will be singular and so the maximum likelihood estimate cannot be computed.

Penalised log-likelihood function

To find a sparse Σ^{-1} we use penalised log-likelihood function

$$\text{argmin}_{\Sigma^{-1} \succ 0}\{-\log\det\Sigma^{-1} + (S\Sigma^{-1}) + ||P * \Sigma^{-1}||_1\}$$

where where S is the sample covariance obtained from the data and P is our sparsity constraint.

In a nutshell

$\pi(x \mid m) \propto \pi(m \mid x)\pi(x)$

Expect that

$$x \sim \mathcal{N}_p(\mu, \Sigma).$$

What is Σ^{-1}?

Penalised log-likelihood function

Estimating $\widehat{\Sigma}^{-1}$

Use algorithms to find $\widehat{\Sigma}^{-1}$

Back to original inverse problem

Solve the original inverse problem using $\widehat{\Sigma}^{-1}$

$\text{argmin}_x \|Ax - m\|^2 - (x-\mu)^T \widehat{\Sigma}^{-1}(x-\mu)$

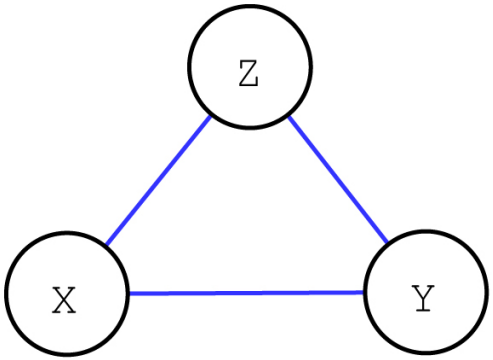
Graphical Lasso

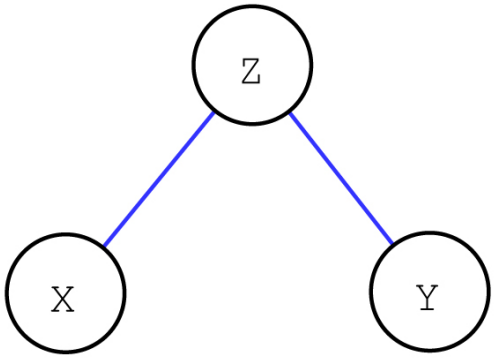
$$\Sigma_1^{-1} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{12} & a_{22} & a_{23} \\ a_{13} & a_{23} & a_{33} \end{pmatrix} \qquad \Sigma_2^{-1} = \begin{pmatrix} b_{11} & 0 & b_{13} \\ 0 & b_{22} & b_{23} \\ b_{13} & b_{23} & b_{33} \end{pmatrix}$$

Graphical Lasso

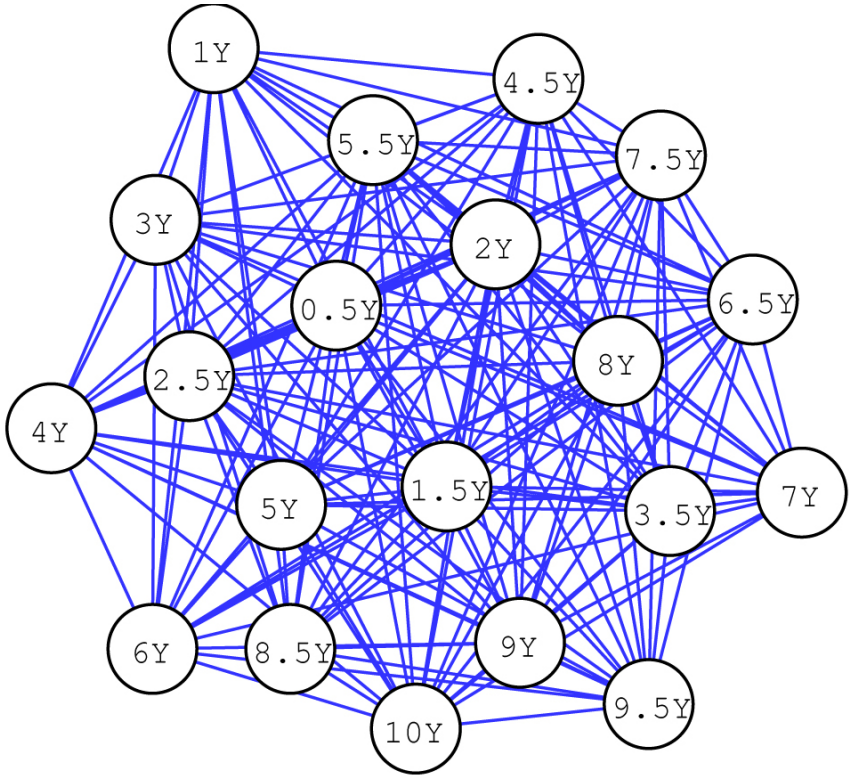
$$\Sigma_1^{-1} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{12} & a_{22} & a_{23} \\ a_{13} & a_{23} & a_{33} \end{pmatrix}$$

$$\Sigma_2^{-1} = \begin{pmatrix} b_{11} & 0 & b_{13} \\ 0 & b_{22} & b_{23} \\ b_{13} & b_{23} & b_{33} \end{pmatrix}$$

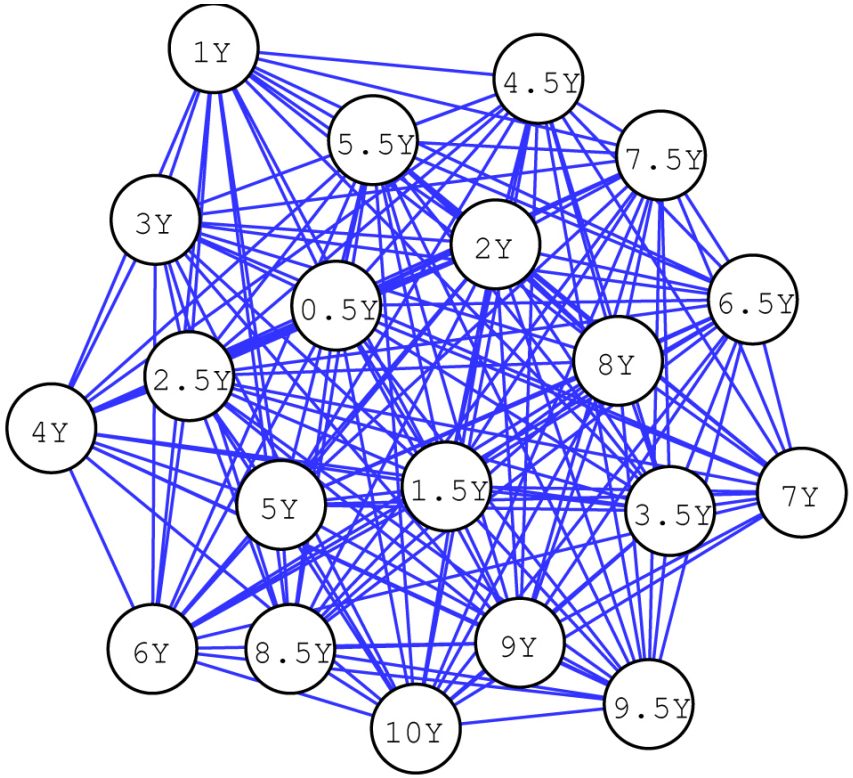


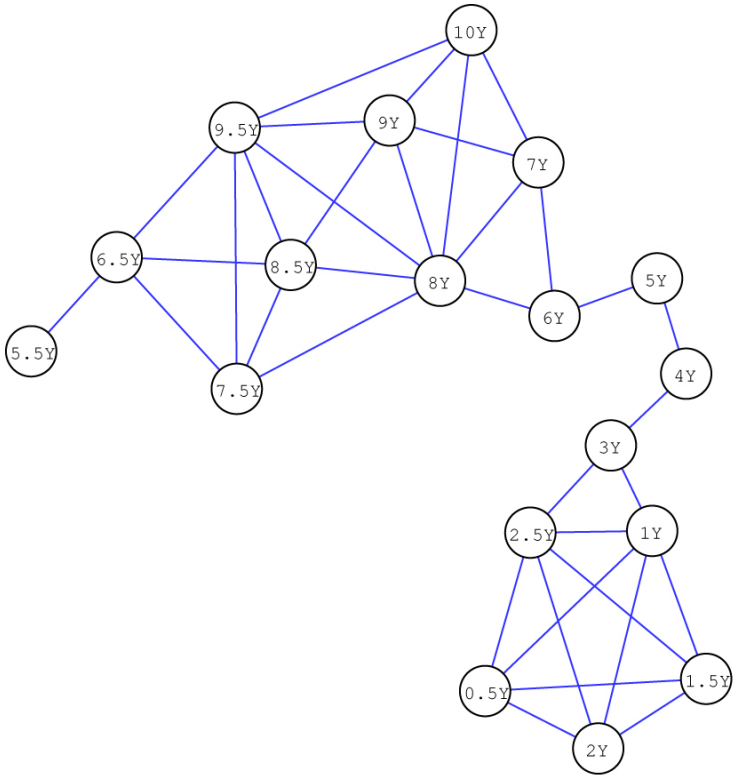


Graphical Lasso



Graphical Lasso





GLASSO

$\Theta := \Sigma^{-1}$

$$\hat{\Theta} = \underset{\Theta \succ 0}{\operatorname{argmin}} - \log\det(\Theta) + tr(S\Theta) + \lambda||\Theta||_1$$

$$\Updownarrow$$

$$\hat{\Theta}^{-1} - S - \lambda\Gamma(\hat{\Theta}) = 0,$$

where $(\Gamma(\hat{\Theta}))_{ij} = \Gamma(\hat{\Theta}_{ij})$ the subgradient of $|\hat{\Theta}_{ij}|$:

$$\Gamma(\hat{\Theta}_{ij}) = sign(\hat{\Theta}_{ij}) \text{ if } \hat{\Theta}_{ij} \neq 0,$$

$$\Gamma(\hat{\Theta}_{ij}) \in (-1, 1) \text{ if } \hat{\Theta}_{ij} = 0$$

GLASSO

$\Theta := \Sigma^{-1}, \; W := \Theta^{-1}$

$$\Theta = \begin{pmatrix} \Theta_{11} & \theta_{12} \\ \theta_{21} & \theta_{22} \end{pmatrix}, \quad S = \begin{pmatrix} S_{11} & s_{12} \\ s_{21} & s_{22} \end{pmatrix}, \quad W = \begin{pmatrix} W_{11} & w_{12} \\ w_{21} & w_{22} \end{pmatrix}$$

Start with $W = S + \lambda I$. We also added Thresholding.
Cycle around rows/columns till convergence:

- solve QP:

$$\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^{p-1}} \frac{1}{2}\beta' W_{11}\beta + \beta' s_{12} + \lambda ||\beta||_1, \; W_{11} \succ 0$$

  using as warm starts the solution from the previous round for this column
- update $\hat{w}_{12} = -W_{11}\hat{\beta}$
- save $\hat{\beta}$ for this column in the matrix B

For every row/column compute $\hat{\theta}_{22} = (s_{22} + \lambda - \hat{\beta}'\hat{w}_{12})^{-1}$, $\hat{\theta}_{12} = \hat{\beta}\hat{\theta}_{22}$.

[J. Friedman, T. Hastie, R. Tibshirani, 2007]

GLASSO 1.7

Theorem. *Solution to the graphical lasso problem to be diagonal with blocks* $C_1, C_2, \ldots C_K \Leftrightarrow |S_{ij}| \leq \lambda$ *for all* $i \in C_k,\ j \in C_l,\ k \neq l$.

$$\Theta = \begin{pmatrix} \Theta_1 & & & \\ & \Theta_2 & & \\ & & \ddots & \\ & & & \Theta_K \end{pmatrix}$$

where Θ_k solves the graphical lasso problem applied to submatrix of S consisting of the entries whose indices are in C_k.

[D.M. Witten, J. Friedman, N. Simon, 2011]

Problem Formulation

Cost Functional Definition

Our goal is to minimize the following functional, $f(v)$, under the set of definite positive matrices.

$$\min_{\Sigma^{-1} \succ 0} f(\Sigma^{-1}) = -\log(\det \Sigma^{-1}) + tr(\Sigma^{-1} S) + \frac{\rho}{2} \|R(\Sigma^{-1})\|_2^2 \tag{1}$$

Minimizing the previous is equivalent to minimize :

$$\min_{\Theta \succ 0} f(\Theta) = -\log(\det \Theta) + tr(\Theta S) + \frac{\rho}{2} \|R(\Theta)\|_2^2 \tag{2}$$

Model Reduction- Closest Neighbor

Neighborhood -Rule

Reduce the model by assuming that each pixel is only related to itself and to those which it shares a boundary or a corner. As the following figure shows:

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	32	33	34	35

Neighborhood -Rule

This rule permits to reduced the problem dimension from $n = (n_x n_y)^2$ to $m = 5n_x n_y - 3n_x - 3n_y + 2$, where n_x and n_y are the number of pixels on the x-axis and y-axis respectively.

Model Reduction-Problem Reformulation

Transformation

Once $g : \mathbf{R}^m \to \mathbf{R}^{n,n}$ is defined, the problem becomes an unconstrained optimization problem on $\mathbf{R}^m$. Therefore the minimizing problem (2) is written as:

$$\min_{v \in \mathbf{R}^m} J(v) = f(g(v)) = -\log(\det g(v)) + tr(g(v)S) + \frac{\rho}{2}\|R(g(v))\|_2^2 \qquad (3)$$

Numerical Approach

Quasi-Newton's Method

Expanding $J(v)$ to a quadratic function we get :

$$J(v+\Delta) \approx J(v) + \nabla J(v) \cdot \Delta + \Delta^t H(v) \Delta \tag{4}$$

Therefore the Δ that minimizes $J(v+\Delta)$ must be the solution of $\Delta = -H^{-1}(v)\, \nabla J(v)$. So given an initial guess v_0 one can solve the iterative sequence:

$$v_k = v_{k-1} - \alpha_k H_k^{-1}(v_{k-1}) J'(v_{k-1}) \tag{5}$$

Gradient Computation:

The gradient of J(v) is computed exactly by:

$$J'(v) = \nabla f(g(v)) \cdot g'(v) \in \mathbf{R}^m \tag{6}$$

Where $\nabla f(\Theta)$ is given by

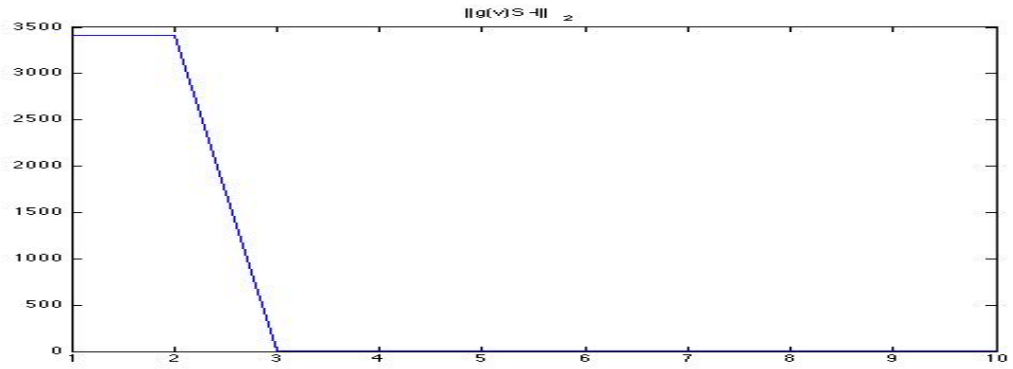
$$(\nabla f)_{i,j} = -\Theta_{j,i}^{-1} + S_{j,i} + +\rho[R'(\Theta)\frac{\partial \Theta}{\partial \sigma_{i,j}}]_{i,j} \tag{7}$$

Hessian Computation:

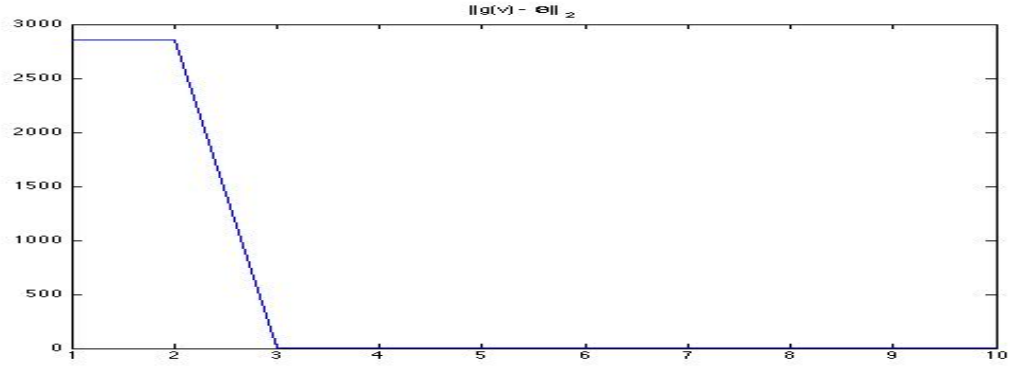
Since is computational expensive to compute the Hessian, one should use an approximation method such as BFGS.

Numerical Results, n=100

Error Analysis $\|g(v)S - I\|_2$



Error Analysis $\|g(v) - \Theta\|_2$



QUIC Algorithm, Cho Jui Hsieh University of Texas:

Cost Function

$$\min_{\Theta} J(\Theta) = -\log(\det \Theta) + tr(\Theta S) + \frac{\rho}{2}\|\Theta\|_1 \tag{8}$$

Split Into a Differentiable Function and a Non-Differentiable

$$g(\Theta) = -\log(\det \Theta) + tr(\Theta S) \tag{9}$$

$$h(\Theta) = \frac{\rho}{2}\|\Theta\|_1 \tag{10}$$

QUIC Algorithm, Cho Jui Hsieh University of Texas:

Second Order Approximation of g(.)

$$G(\Delta) = g(\Theta + \Delta) = g(\Theta) + \nabla g(\Theta + \Delta) + \Delta^t H \Delta \tag{11}$$

New Cost functional

$$\min_{\Delta} F(\Delta) = G(\Delta) + h(\Theta + \Delta) \tag{12}$$

Choice of Δ

$$\Delta = \mu[e_j e_i^t + e_i e_j^t] \tag{13}$$

$$\min_{\mu} F(\mu) \tag{14}$$

Overview / Future Work

Overview

- Model Reduction
- Numerical Approach
- Numerical Results
- QUIC Algorithm

Future Work

- Improve Memory Efficiency
- Better Choice of x_0
- Different Rules of Reduction
- Incorporate the Model Reduction to QUIC

Simulation

Inverse Covariance Test Matrix

$\Sigma^{-1} =$ 

Multivariate Normal Sampling

$x_1, x_2, \ldots, x_N$

Sample Covariance

$\mathbf{S} = xx^T/N$

Inverse Covariance Estimation

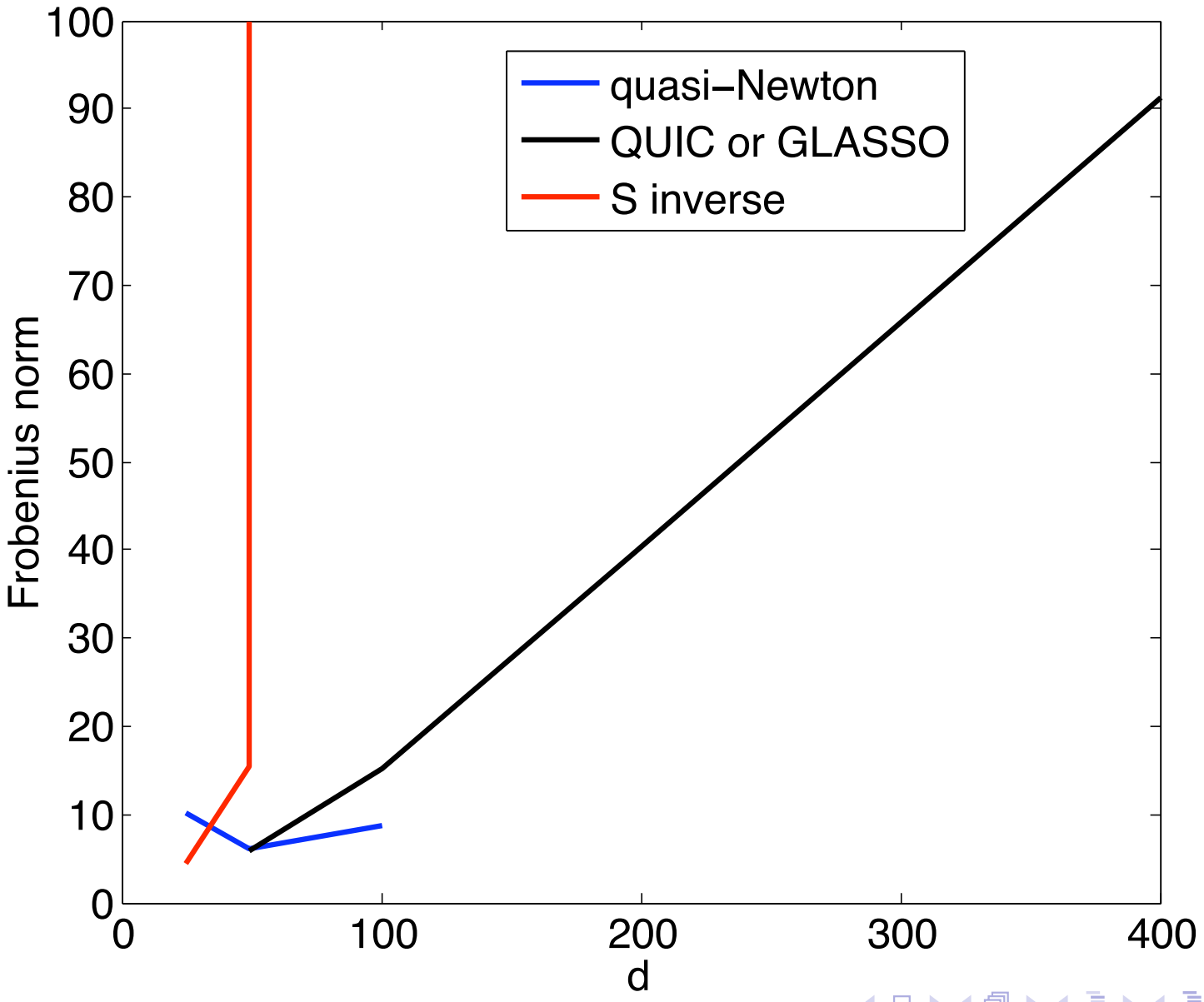
$\hat{\Sigma}^{-1}$

Timing (s)

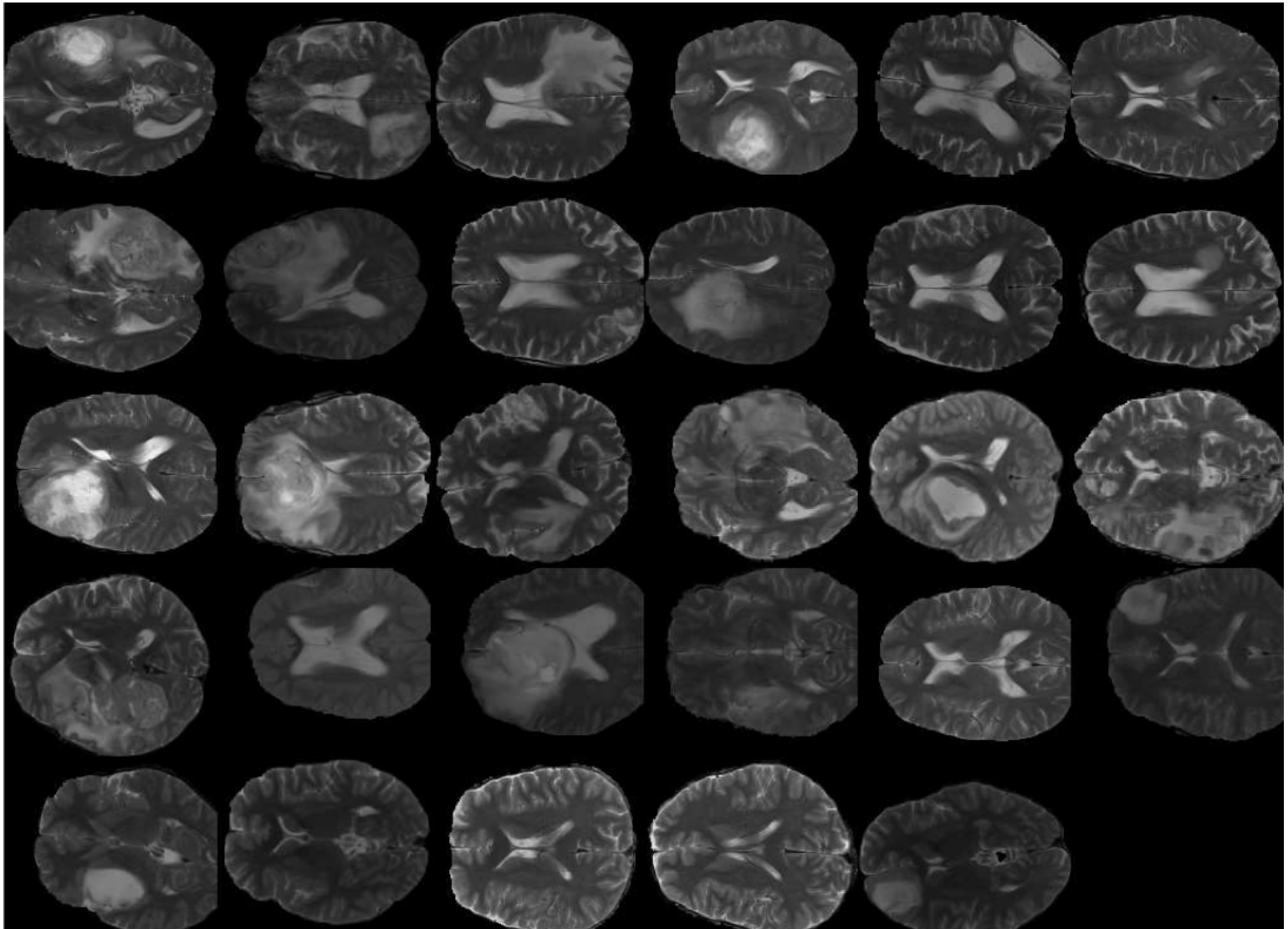
Sample data: $X \in \mathbb{R}^{d \times N}$ with $N = 100$.
Covariance matrix $S \in \mathbb{R}^{d \times d}$.

Algorithm	$d = 49$	$d = 100$	$d = 200$	$d = 400$
S^{-1}	0.0004	0.0018	0.0083	0.0477
GLASSO	0.01 ± 0.01	0.17 ± 0.01	2.03 ± 0.03	23.51 ± 0.22
QUIC	0.04 ± 0.02	0.39 ± 0.03	4.72 ± 0.11	77.15 ± 2.46
quasi-Newton	10.5	56.2	$*$	$*$

Error Analysis ($N = 100$)

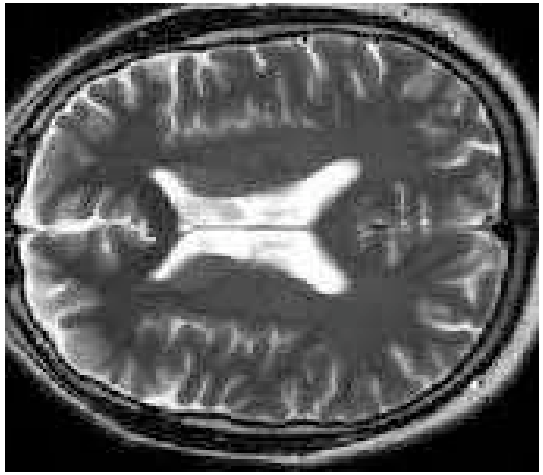


Training Set extracted from BraTS Multimodal Brain Tumor Segmentation Challenge

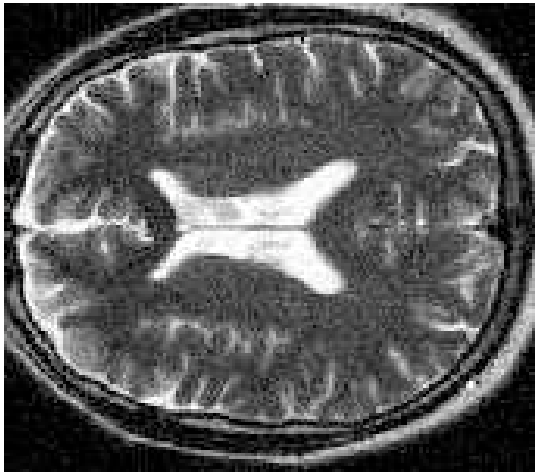


Results

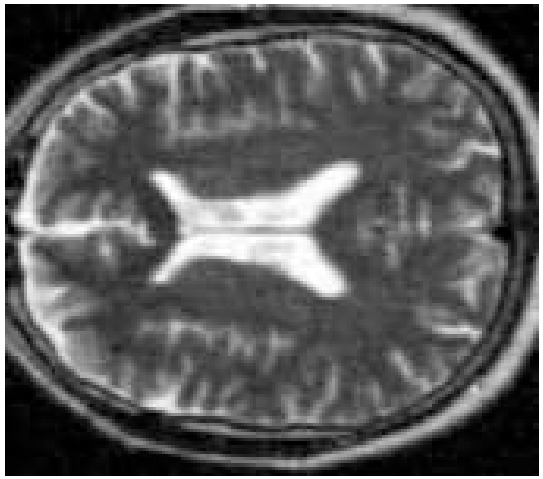
original



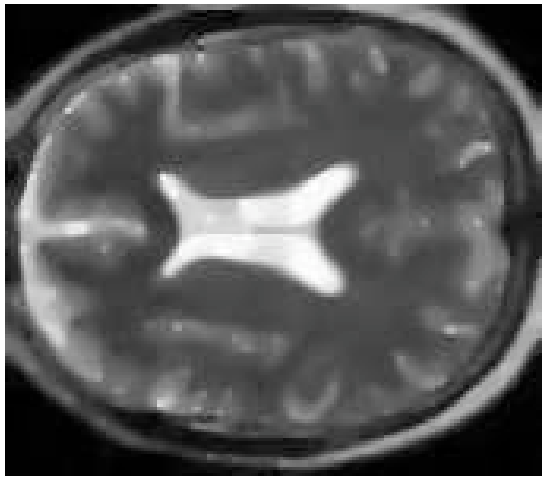
noisy (σ = 5%)



GRMF denoising



TV denoising



Remarks

1. Computed Inverse Covariance Matrix on 15×17 subblocks and aggregated blocks;
2. Assumed Stationarity;
3. Assumed Gaussianity of the image pixels;
4. Performed very basic (poor) alignment of training set;
5. Had a small (29) (and unhealthy) training set;

Nevertheless, it performs better than TV denoising!

Future Challenges

1. Computational Challenge: computing inverse covariance matrix on full images (e.g. $40,000 \times 40,000$)
   - Could Iteratively Re-weighted Least Squares Minimization for Sparse Recovery help?
2. Parameter selection: “warm start” and generalized cross-validation;
3. Non-stationary model: use bigger and better aligned database;
4. Try Gaussianity model in transform domain or higher order models;

THANK YOU FOR YOUR TIME!!!