Polynomial Linear Programming with Gaussian Belief Propagation

Danny Bickson

IBM Haifa Research Lab
Mount Carmel, Haifa 31905, Israel
Email: dannybi@il.ibm.com

9/25/08

Joint work

- Yoav Tock - IBM Haifa Research Lab, Israel
- Danny Dolev - The Hebrew University, Israel
- Ori Shental - University of San Deigo

- O. Shental acknowledges the partial support of the NSF (Grant CCF-0514859)

Outline

On the relation between LP and BP

State-of-the-art results

- [Weiss *et al.*, JMLR 2006, UAI 2007]
- [Globerson *et al.* UAI 2007, JMLR 2008]
- [Hazan *et al.* UAI 2008]

- Common approach:
  - Global convex approximation of the MAP assignment problem
  - Belief propagation using discrete variables
  - Gradient descent type algorithms.

Our new approach

- Utilize Interior-point methods.
  - Faster convergence
  - Higher computational effort.
  - Main bottleneck is the Hessian inversion task.
- Approximate the (Non)Linear program around a point.
- Overcome the computational effort using the Gaussian Belief Propagation (GaBP) algorithm.
  - Using continuous variables.

Linear programming background

LP in its canonical form

$$\begin{aligned} \text{minimize} \quad & \mathbf{c}^T\mathbf{x} \\ \text{subject to} \quad & A\mathbf{x} = \mathbf{b}, \quad \mathbf{x} \geq 0. \end{aligned}$$

where $A \in \mathbb{R}^{n \times p}$ with $\mathbf{rank}\{A\} = p < n$. We assume the problem is solvable with an optimal $\mathbf{x}^*$. We also assume that the problem is strictly feasible, in other words there exists $\mathbf{x} \in \mathbb{R}^n$ that satisfies $A\mathbf{x} = \mathbf{b}$ and $\mathbf{x} > 0$.

Linear programming background

Log Barrier Method

$$\begin{array}{ll} \text{minimize}_{\mathbf{x},\mu} & \mathbf{c}^T\mathbf{x} - \mu\Sigma_{k=1}^{n} \log x_k \\ \text{subject to} & \mathbf{A}\mathbf{x} = \mathbf{b}. \end{array}$$

This is an approximation to the original problem. The quality of the approximation improves as the parameter $\mu \to 0$.

Linear programming background

The Newton method

Given		feasible starting point $\mathbf{x}_0$ and tolerance $\epsilon > 0$, $k = 1$
Repeat	1	Compute the Newton step and decrement $\Delta\mathbf{x} = f''(\mathbf{x})^{-1} f'(\mathbf{x}), \quad \lambda^2 = f'(\mathbf{x})^T \Delta\mathbf{x}$
	2	Stopping criterion. quit if $\lambda^2/2 \leq \epsilon$
	3	Line search. Choose step size t by backtracking line search.
	4	Update. $\mathbf{x}_k := \mathbf{x}_{k-1} + t\Delta\mathbf{x}, \quad k = k + 1$

From linear programming to belief propagation

- By computing derivative of the Lagrangian, and compare it to zero we get the first order optimality conditions:

$$\mathbf{AX}^2\mathbf{A}^T\mathbf{y} = \mathbf{AX}^2\mathbf{c} - \mu\mathbf{AX1}.$$

- These normal equations can be recognized as generated from the linear least-squares problem

$$\min_{\mathbf{y}} ||\mathbf{XA}^T\mathbf{y} - \mathbf{Xc} - \mu\mathbf{AX1}||_2^2.$$

From linear programming to belief propagation

- Simplifying notations we get

$$\min_{\mathbf{y}} ||\mathbf{F}\mathbf{y} - \mathbf{g}||_2^2,$$

where $\mathbf{F} \triangleq \mathbf{X}\mathbf{A}^T, \mathbf{g} \triangleq \mathbf{X}\mathbf{c} + \mu\mathbf{A}\mathbf{X}\mathbf{1}.$

- Now we define a multivariate Gaussian

$$p(\hat{\mathbf{x}}) \triangleq p(\mathbf{x}, \mathbf{y}) \propto \exp(-1/2(\mathbf{F}\mathbf{y} - \mathbf{g})^T \mathbf{I}(\mathbf{F}\mathbf{y} - \mathbf{g})).$$

- It is clear that $\hat{\mathbf{y}}$, the minimizing solution of (3), is the MAP estimator of the conditional probability

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}) = \mathcal{N}((\mathbf{F}^T\mathbf{F})^{-1}\mathbf{F}^T\mathbf{g}, (\mathbf{F}^T\mathbf{F})^{-1}).$$

Efficient distributed solution using the GaBP algorithm

- We have shown that canonical LP can be reduced to a least square problems.
- Following recent work on GaBP, we propose to use it as an iterative algorithm for distributively computing the MMSE solution over a communication network.

The GabP algorithm

#	Stage	Operation
1.	*Initialize*	Compute $P_{ii} = A_{ii}$ and $\mu_{ii} = b_i / A_{ii}$. Set $P_{ki} = 0$ and $\mu_{ki} = 0, \forall k \neq i$.
2.	*Iterate*	Propagate P_{ki} and $\mu_{ki}, \forall k \neq i \text{ such that } A_{ki} \neq 0$. Compute $P_{i\backslash j} = P_{ii} + \sum_{k \in \mathrm{N}(i) \backslash j} P_{ki}$ and $\mu_{i \backslash j} = P_{i \backslash j}^{-1} (P_{ii}\mu_{ii} + \sum_{k \in \mathrm{N}(i) \backslash j} P_{ki}\mu_{ki})$. Compute $P_{ij} = -A_{ij} P_{i \backslash j}^{-1} A_{ji}$ and $\mu_{ij} = -P_{ij}^{-1} A_{ij} \mu_{i \backslash j}$.
3.	*Check*	If P_{ij} and μ_{ij} did not converge, return to #2. Else, continue to #4.
4.	*Infer*	$P_i = P_{ii} + \sum_{k \in \mathrm{N}(i)} P_{ki}$, $\mu_i = P_i^{-1} (P_{ii}\mu_{ii} + \sum_{k \in \mathrm{N}(i)} P_{ki}\mu_{ki})$.
5.	*Output*	$x_i = \mu_i$

Extension to the primal dual algorithm

- We have shown, that the primal-dual method can be computed using GaBP by inverting the following matrix:

$$\begin{pmatrix} 0 & \mathbf{A}^T & I \\ \mathbf{A} & 0 & 0 \\ I & 0 & \mathbf{Z}^{-1}\mathbf{X} \end{pmatrix} \begin{pmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\mathbf{z} \end{pmatrix} = \begin{pmatrix} \mathbf{b} - \mathbf{A}\mathbf{x} \\ \mathbf{c} - \mathbf{A}^T\mathbf{y} - \mathbf{z} \\ \mu\mathbf{Z}^{-1}\mathbf{1} - \mathbf{X} \end{pmatrix}.$$

- In general, this method can be extended whenever the Newton method is used.

GaBP Convergence and Exactness

Theorem [Weiss and Freeman,'01,Claim 4]

If the matrix $\mathbf{A}$ is strictly diagonally dominant (*i.e.* , $|A_{ii}| > \sum_{j \neq i} |A_{ij}|, \forall i$), then the GaBP solver converges and the marginal means converge to the true solution.

Theorem [Johnson *et al.* ,'06,Proposition 2]

If the spectral radius (maximum of the absolute values of the eigenvalues) ρ of the matrix $|\mathbf{I}_n - \mathbf{A}|$ satisfies $\rho(|\mathbf{I}_n - \mathbf{A}|) < 1$, then the GaBP solver converges and the marginal means converge to the true solution.

New convergence results

Weiss' Convergence Proof

- Based on the work on Weiss which uses the pairwise potentials model

$$
\begin{aligned}
p(\mathbf{x}) &\propto \Pi_{i,j}\psi_{ij}(x_i, x_j)\Pi_i\psi_i(x_i), \\
\psi_{i,j}(x_i, x_j) &\equiv \exp(-1/2(x_i\ x_j)^T\mathbf{V}_{ij}(x_i\ x_j)), \\
\mathbf{V}_{ij} &\equiv \begin{pmatrix} \tilde{a}_{ij} & \tilde{b}_{ij} \\ \tilde{b}_{ji} & \tilde{c}_{ij} \end{pmatrix}.
\end{aligned}
$$

- Assuming the optimal solution is $\mathbf{x}^*$, for a desired accuracy $\epsilon||\mathbf{b}||_\infty$ where $||\mathbf{b}||_\infty \equiv \max_i |\mathbf{b}_i|$, and $\mathbf{b}$ is the shift vector, we need to run the algorithm for at most $t = \lceil \log(\epsilon)/\log(\beta) \rceil$ rounds to get an accuracy of $|x^* - x_t| < \epsilon||\mathbf{b}||_\infty$ where

$$\beta = \max_{ij} |\tilde{b}_{ij}/\tilde{c}_{ij}|.$$

- We use the *information form* $p(\mathbf{x}) \propto \exp(-1/2\mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{b}^T\mathbf{x})$.
- The decomposition of the matrix $\mathbf{A}$ into pairwise potentials is not unique. Any decomposition should be subject to:

$$\tilde{b}_{ij} = \mathbf{A}_{ij}, \qquad \Sigma_j \tilde{c}_{ij} = \mathbf{A}_{ii}.$$

- Assuming the matrix $\mathbf{A}$ is diagonally dominant, we define ε_i to be the non negative gap

$$\varepsilon_i \triangleq |\mathbf{A}_{ii}| - \Sigma_j |\mathbf{A}_{ij}| > 0.$$

and the following decomposition

$$\tilde{b}_{ij} = \mathbf{A}_{ij}, \quad \tilde{c}_{ij} = \mathbf{A}_{ij} + \varepsilon_i / |N(i)|,$$

where $|N(i)|$ is the number of graph neighbors of node i.

- Following Weiss, we define γ to be

$$\gamma = \max_{i,j} \frac{|\tilde{b}_{ij}|}{|\tilde{c}_{ij}|} = \frac{|a_{ij}|}{|a_{ij}| + \varepsilon_i/|N(i)|} =$$

$$= \max_{i,j} \frac{1}{1 + \varepsilon_i/(|a_{ij}||N(i)|)} < 1. \quad (3)$$

- In total, we get that for a desired accuracy of $\epsilon||\mathbf{b}||_\infty$ we need to iterate for $t = \lceil \log(\epsilon)/\log(\gamma) \rceil$ rounds. Note that this is an upper bound and in practice we indeed have observed a much faster convergence rate.
- The computation of the parameter γ can be easily done in a distributed manner: Each node locally computes ε_i, and $\gamma_i = \max_j 1/(1 + |a_{ij}|\varepsilon_i/N(i))$. Finally, one maximum operation is performed globally, $\gamma = \max_i \gamma_i$.

Computational overhead

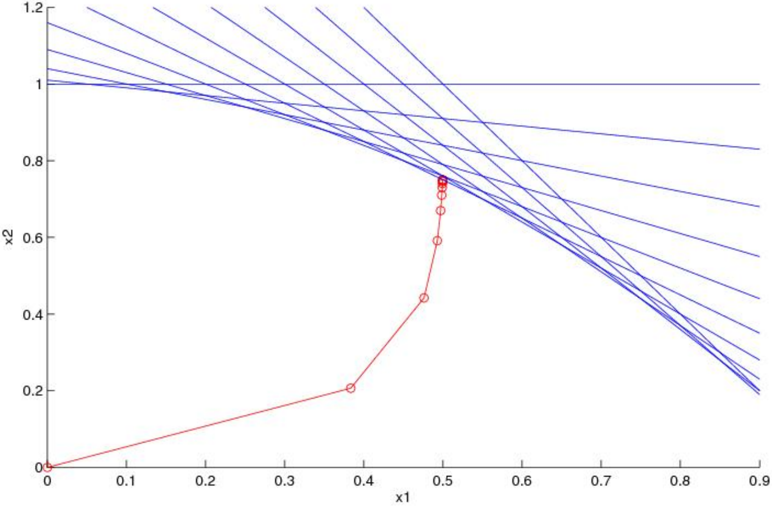
Karmarkar algorithm

- Number of rounds n.
- On a dense constraint matrix, work on each round is $O(n^{2.5})$.
- Total of $O(n^{3.5})$.

Using GaBP

- Number of rounds n.
- Work on each round $O(np\log(\epsilon)/\log(\gamma))$.
- In total we get $O(n^2p\log(\epsilon)/\log(\gamma))$

Toy linear program



Conclusion

- We have shown how to efficiently and distributively apply the GaBP algorithm for solving linear programs using interior point methods.
- Unlike previous approaches which use global convexity, using discrete variables we use local approximations using continuous variables.
- We provide a new upper bound on the convergence speed of the GaBP algorithm.

THANK YOU!