An Incremental Subgradient Algorithm for MAP Estimation in Graphical Models

Jeremy Jancsary  Gerald Matz\textsuperscript{1}  Harald Trost\textsuperscript{2}

jeremy.jancsary@ofai.at

December 8, 2010

\textsuperscript{1}Vienna University of Technology
\textsuperscript{2}Medical University of Vienna
A Few Things to Take Away From This Talk

• Why you might be interested in our algorithm:
  • It is efficient, both computationally and memory-wise.
  • It finds better solutions than the methods we compared it to.
  • It converges to the global optimum of the first-order LP relaxation of the MAP problem (which is tight in some cases).
  • You get a certificate of optimality w.r.t. the discrete problem.

• How it all works:
  • Start out with tree-reweighted upper bound (Wainwright et al., 2005).
  • The upper bound is developed until it assumes a degenerate form involving a large number of easy problems.
  • The tightest bound can then be found very efficiently using incremental methods, solving one easy problem at a time.
  • Equivalent to maximizing the LP relaxation of the MAP problem.
Outline

1. The Problem
2. Towards a Solution
3. Some Properties
4. Experimental Results
5. Conclusion
Outline

1. The Problem
2. Towards a Solution
3. Some Properties
4. Experimental Results
5. Conclusion
Maximum-a-Posteriori (MAP) Estimation

Consider an undirected graphical model (e.g. ) with vertex set $\mathcal{V}$ and edge set $\mathcal{E}$ defined over discrete random variables with pairwise interactions. Potential of a particular variable state $x \in \mathcal{X}^n$:

$$P(x; \theta) = \sum_{s \in \mathcal{V}} \theta_s(x_s) + \sum_{(s,t) \in \mathcal{E}} \theta_{st}(x_s, x_t).$$

**MAP Estimation (Discrete Problem) (OP1)**

$$\tilde{P}(\theta) = \max_{x \in \mathcal{X}^n} P(x; \theta),$$

$$\tilde{x} = \arg\max_{x \in \mathcal{X}^n} P(x; \theta).$$

Computation of these quantities is NP-hard in general (notable exceptions: trees; binary variables + submodular energies).
Outline

1. The Problem

2. Towards a Solution

3. Some Properties

4. Experimental Results

5. Conclusion
Relaxing the Discrete Problem

**Tightest Tree-Reweighted Upper Bound**  
\[ \min_{\{\theta^T\} \in C(\theta)} \sum_T \rho^T \bar{P}(\theta^T), \]

\[ C(\theta) = \left\{ \{\theta^T\} \mid \sum_T \rho^T \theta^T = \theta \right\} \text{ and } \{\rho^T\} \in \{\rho \geq 0 \mid \sum_T \rho^T = 1\} . \]

is connected through strong duality to

**First-Order LP Relaxation**  
\[ \max_{\mu \in L} \sum_s \mu_s \cdot \theta_s + \sum_{(s,t)} \mu_{st} \cdot \theta_{st}, \]

\[ L = \left\{ \mu \geq 0 \mid \begin{array}{l} \sum_{x_s} \mu_s(x_s) = 1 \text{ for all } s \\ \sum_{x_t'} \mu_{st}(x_s', x_t) = \mu_t(x_t), \sum_{x_t'} \mu_{st}(x_s, x_t') = \mu_s(x_s) \end{array} \right\} . \]
Simplifying the Upper Bound (1)

• **Curious fact #1** (Wainwright et al., 2005): Choice of $\rho$ irrelevant as long as all edges are covered (otherwise, $C(\theta)$ is empty). Minimization of tree-reweighted upper bound $\equiv$ maximization of LP relaxation, which does not depend on $\rho$.

• Can pick small set $S(T)$ of trees needed to cover all edges and set $\rho^T = \rho \overset{\text{def}}{=} 1/|S(T)|$ if $T \in S(T)$, and $\rho^T = 0$ otherwise.

• Exploit linearity of $P(\cdot)$, move $\rho$ into params—viz. $\lambda^T = \rho \theta^T$:

```
“Dual Decomposition”-like Formulation (OP4)

\[
\min_{\{\lambda^T\} \in S(\theta)} \sum_{T \in S(T)} \bar{P}(\lambda^T) \quad \text{with} \quad S(\theta) = \left\{ \{\lambda^T\} \mid \sum_{T \in S(T)} \lambda^T = \theta \right\}.
\]
```
Simplifying the Upper Bound (2)

- **Curious fact #2** (Kolmogorov, 2006): Trees in tree-reweighted upper bound need not be spanning (→ no impact on **tightest** bound).
- **Central idea #1**: Choose each tree $T$ as single edge $E = (s, t)$.
- Determines almost all parameters:
  \[ \lambda^E_{st} = \begin{cases} \theta_{st} & \text{if } E = (s, t), \\ 0 & \text{otherwise} \end{cases}, \quad \lambda^E_s = 0 \text{ if } s \notin E. \]
- Remaining parameters of an edge $E = (s, t)$: $\lambda^E = \{\lambda^E_s, \lambda^E_t\}$.

**Tightest Degenerate Upper Bound** (OP5)

\[
\min_{\lambda \in \mathcal{Q}(\theta)} D(\lambda; \theta) \overset{\text{def}}{=} \sum_E \max_{(x_s, x_t)} \{\lambda^E_s(x_s) + \lambda^E_t(x_t) + \theta_{st}(x_s, x_t)\},
\]

\[
\mathcal{Q}(\theta) = \left\{ \{\lambda^E\} \mid \sum_{E : s \in E} \lambda^E_s = \theta_s \text{ for all } s \right\}.
\]
Tightening the Upper Bound

**Subgradient**

Objective $D(\lambda; \theta)$ is non-differentiable, but $g \in \mathbb{R}^{2|\mathcal{X}|\mathcal{E}}$ is given by:

$$g^E_s(x_s) = [x_s = \bar{x}^E_s], \quad g^E_t(x_t) = [x_t = \bar{x}^E_t]$$

for all $E = (s, t), x_s, x_t$, where we use $(\bar{x}^E_s, \bar{x}^E_t)$ to refer to the edge MAP state (cf. OP5).

**Projection**

We need to solve $\arg\min_{\lambda \in \mathcal{Q}(\theta)} \|\lambda - \lambda'\|_2^2$. Solution obtained as:

$$\mathcal{P}_\theta(\lambda') = \left\{ \lambda^E_s(x_s) \leftarrow \lambda^E'_s(x_s) - \left( \sum_{E' \in \mathcal{E}_s} \lambda^{E'}_s(x_s) - \theta_s(x_s) \right) / |\mathcal{E}_s| \right\},$$

which distributes amount of change uniformly over adjacent edges.

**Central idea #2**: Separable, non-diff. problem, cheap projection $\rightarrow$ use incremental subgradient method (Nedić and Bertsekas, 2001).
The Algorithm

**Input**: Graph $G$, target parameters $\theta$, initial feasible point $\lambda$

**Output**: Feasible primal solution $\tilde{x}$ that is an approximation to $\bar{x}$

choose initial feasible primal solution $\tilde{x}$ arbitrarily;

repeat

pick next step size $\alpha$ and shuffle the set of edges $\mathcal{E}$;

foreach $E = (s, t) \in \mathcal{E}$ do

find MAP state: $(\bar{x}_s^E, \bar{x}_t^E)$;

subtract subgradient: $\lambda_s^E(\bar{x}_s^E) \leftarrow \lambda_s^E(\bar{x}_s^E) - \alpha$, $\lambda_t^E(\bar{x}_t^E) \leftarrow \lambda_t^E(\bar{x}_t^E) - \alpha$;

foreach $E' \in \mathcal{E}_s$ do project: $\lambda_s^{E'}(\bar{x}_s^E) \leftarrow \lambda_s^{E'}(\bar{x}_s^E) + \alpha/|\mathcal{E}_s|$;

foreach $E' \in \mathcal{E}_t$ do project: $\lambda_t^{E'}(\bar{x}_t^E) \leftarrow \lambda_t^{E'}(\bar{x}_t^E) + \alpha/|\mathcal{E}_t|$;

foreach $s \in \mathcal{V}$ do build candidate $\tilde{c}$: $\tilde{c}_s \leftarrow$ at random from $\{\bar{x}_s^E | E \in \mathcal{E}_s\}$;

if $P(\tilde{c}; \theta) > P(\tilde{x}; \theta)$ then

accept best primal solution so far: $\tilde{x} \leftarrow \tilde{c}$;

if $D(\lambda; \theta) = P(\tilde{x}; \theta)$ then

optimal primal solution found: return $\tilde{x}$;

until converged;

approximate primal solution found: return $\tilde{x}$;
Outline

1. The Problem
2. Towards a Solution
3. Some Properties
4. Experimental Results
5. Conclusion
Formal Guarantees

Proposition (Global Convergence)

For an appropriately chosen sequence of step sizes \( \{\alpha^{(k)}\} \), convergence to the global optimum of (OP5) is guaranteed as \( k \to \infty \).

The choice of \( \{\alpha^{(k)}\} \) is discussed in detail in the paper.

Proposition (Optimality of Primal Solutions)

Assume that at an outer iteration, \( P(\tilde{x}; \theta) = D(\lambda; \theta) \). It follows that \( \tilde{x} \) maximizes \( P(\cdot) \) and \( \lambda \) minimizes \( D(\cdot) \). This happens precisely if for each node, the edge MAP states agree on a common node MAP state.

We thus obtain a certificate of optimality for our primal solution.
Comparison to Related Approaches

Several methods have been devised that all aim at solving the first-order linear programming relaxation (OP3).

<table>
<thead>
<tr>
<th>Method</th>
<th>Converg.</th>
<th>Global</th>
<th>Rate</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>IncMP (Our Method)</td>
<td>yes</td>
<td>yes</td>
<td>sublinear</td>
<td>$O(</td>
</tr>
<tr>
<td>DDSUB (Komodakis et al., 2007)</td>
<td>yes</td>
<td>yes</td>
<td>sublinear</td>
<td>$O(</td>
</tr>
<tr>
<td>TRWMP (Wainwright et al., 2005)</td>
<td>no</td>
<td>no</td>
<td>?</td>
<td>$O(</td>
</tr>
<tr>
<td>TRW-S (Kolmogorov, 2006)</td>
<td>yes</td>
<td>no</td>
<td>?</td>
<td>$O(</td>
</tr>
<tr>
<td>MPLP (Globerson et al., 2007)</td>
<td>yes</td>
<td>no</td>
<td>?</td>
<td>$O(</td>
</tr>
<tr>
<td>PROXLP (Ravikumar et al., 2010)</td>
<td>yes</td>
<td>yes</td>
<td>superlinear</td>
<td>$O(</td>
</tr>
<tr>
<td>DDPROX (Jojic et al., 2010)</td>
<td>yes</td>
<td>yes</td>
<td>linear</td>
<td>$O(</td>
</tr>
</tbody>
</table>

The convergence rates and working memory requirements listed above are asymptotic and do not necessarily reveal a lot about real-world performance (cost of an iteration is crucial).
Outline

1. The Problem
2. Towards a Solution
3. Some Properties
4. Experimental Results
5. Conclusion
Experimental Setup

We compared three solvers (IncMP, DDSub and TrwMP) on three different types of graphs, averaged over 20 runs.

**GridIsingUni**: A $50 \times 50$ grid with binary variables ($\mathcal{X} = \{-1, +1\}$) and potentials given by $\theta_s(x_s) = \gamma x_s$ and $\theta_{st}(x_s, x_t) = \gamma x_s x_t$ with $\gamma \sim U(-1, +1)$ drawn independently for each node and edge.

**GridMultiGauss**: A $20 \times 20$ grid with variables of arity $|\mathcal{X}| = 16$ and potentials chosen as $\theta_s(x_s) = 0$ and $\theta_{st}(x_s, x_t) \sim \mathcal{N}(0, 15)$ independently.

**ComplIsingUni**: A complete graph of 50 binary variables with potentials chosen akin to GridIsingUni.
Results

• Measured the score $P(\tilde{x}; \theta)$ of the best primal solution $\tilde{x}$ found so far, as a function of running time (seconds).
• For IncMP and DDSUB, constructed $\tilde{x}$ randomly from the edge and tree MAP states, respectively (at each iteration).
• For TrwMP, used the maximizers of the node beliefs.

GridIsingUni  
GridMultiGauss  
CompIsingUni
Outline

1 The Problem

2 Towards a Solution

3 Some Properties

4 Experimental Results

5 Conclusion
Future Work

• Step size $\alpha$ can be determined analytically so as not to increase the dual objective $P(\lambda, \theta)$.
  • Turns algorithm into a “dual descent method” (Bertsekas, 1999).
  • Open question: Can global convergence still be guaranteed?
  • Most likely, can get stuck in a “corner” (akin to MPLP).

• Use as computational core in a branch-and-bound scheme.
  • Low working memory requirements, ideal for parallelization.
  • Constraints like “$X_s \overset{!}{=} x_s$” can easily be added, warm-starting should work rather well.

• Release source code as part of the PhiWeave package for approximate training of discriminative graphical models.
Some References

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky.
*MAP estimation via agreement on (hyper)trees: Message passing and linear-programming approaches.*

V. Kolmogorov.
*Convergent tree-reweighted message passing for energy minimization.*
Pattern Analysis and Machine Intelligence, Vol. 28(10), 2006.

A. Nedić and D. P. Bertsekas.
*Incremental subgradient methods for nondifferentiable optimization.*

N. Komodakis, N. Paragios, and G. Tziritas.
*MRF optimization via dual decomposition: Message-passing revisited.*

A. Globerson and T. Jaakkola.
*Fixing max-product: Convergent message passing algorithms for MAP-LP relaxations.*

P. Ravikumar, A. Argarwal, and M. J. Wainwright.
*Message-passing for graph-structured linear programs: proximal methods and rounding schemes.*

V. Jojic, S. Gould, and D. Koller.
*Accelerated dual decomposition for MAP inference.*
Consider an undirected graphical model (e.g. 
\[ \begin{array}{cccccc}
\text{●} & \text{●} & \text{●} & \text{●} & \text{●} \\
\text{●} & \text{●} & \text{●} & \text{●} & \text{●} \\
\text{●} & \text{●} & \text{●} & \text{●} & \text{●} \\
\text{●} & \text{●} & \text{●} & \text{●} & \text{●} \\
\text{●} & \text{●} & \text{●} & \text{●} & \text{●} \\
\end{array} \] 
) with vertex set \( V \) and edge set \( E \) defined over discrete random variables with pairwise interactions. Potential of a particular variable state \( x \in X^n \):

\[ P(x; \theta) = \sum_{s \in V} \theta_s(x_s) + \sum_{(s, t) \in E} \theta_{st}(x_s, x_t). \]

MAP Estimation (Discrete Problem) (OP1)

\[ \bar{P}(\theta) = \max_{x \in X^n} P(x; \theta), \quad \bar{x} = \arg\max_{x \in X^n} P(x; \theta). \]

Computation of these quantities is NP-hard in general (notable exceptions: trees; binary variables + submodular energies).