Dual Decomposition for Marginal Inference

Justin Domke

Rochester Institute of Technology

AAAI 2011

Outline

Outline

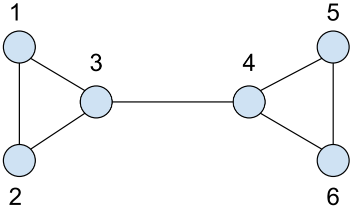
Graphical Models

- Markov Random Field / Factor Graph:

$$p(\mathbf{x}) \propto \prod_{c} \psi(\mathbf{x}_c)$$

Graphical Models

$$c_1 = \{1,2,3\},\ c_2 = \{3,4\},\ c_3 = \{4,5,6\}$$



$$\begin{aligned} p(\mathbf{x}) &\propto \prod_c \psi(\mathbf{x}_c) \\ &= \psi(x_1, x_2, x_3)\psi(x_3, x_4)\psi(x_4, x_5, x_6) \end{aligned}$$

Marginal Inference

- Want to recover $p(X_i = x_i)$.
- Brute-force sum: Define $\hat{p}(\mathbf{x}) = \prod_c \psi(\mathbf{x}_c)$

$$\begin{aligned} P(X_i = x_i) &= \frac{1}{Z} \sum_{x_1} \dots \sum_{x_{i-1}} \sum_{x_{i+1}} \dots \sum_{x_M} \hat{p}(\mathbf{x}) \\ Z &= \sum_{x_1} \dots \sum_{x_M} \hat{p}(\mathbf{x}) \end{aligned}$$

- On trees, can do sums quickly by dynamic programming.
  - Sum-product algorithm / belief propagation
- #P-hard
  - Approximate: Tree-reweighted belief propagation (TRW)
  - This paper: Same approximation as TRW, different algorithm.

Marginal Inference

- Want to recover $p(X_i = x_i)$.
- Brute-force sum: Define $\hat{p}(\mathbf{x}) = \prod_c \psi(\mathbf{x}_c)$

$$
\begin{aligned}
P(X_i = x_i) &= \frac{1}{Z} \sum_{x_1} \cdots \sum_{x_{i-1}} \sum_{x_{i+1}} \cdots \sum_{x_M} \hat{p}(\mathbf{x}) \\
Z &= \sum_{x_1} \cdots \sum_{x_M} \hat{p}(\mathbf{x})
\end{aligned}
$$

- On trees, can do sums quickly by dynamic programming.
  - Sum-product algorithm / belief propagation
- #P-hard
  - Approximate: Tree-reweighted belief propagation (TRW)
  - This paper: Same approximation as TRW, different algorithm.

Marginal Inference

- Want to recover $p(X_i = x_i)$.
- Brute-force sum: Define $\hat{p}(\mathbf{x}) = \prod_c \psi(\mathbf{x}_c)$

$$
\begin{aligned}
P(X_i = x_i) &= \frac{1}{Z} \sum_{x_1} \cdots \sum_{x_{i-1}} \sum_{x_{i+1}} \cdots \sum_{x_M} \hat{p}(\mathbf{x}) \\
Z &= \sum_{x_1} \cdots \sum_{x_M} \hat{p}(\mathbf{x})
\end{aligned}
$$

- On trees, can do sums quickly by dynamic programming.
  - Sum-product algorithm / belief propagation
- #P-hard
  - Approximate: Tree-reweighted belief propagation (TRW)
  - This paper: Same approximation as TRW, different algorithm.

Marginal Inference

- Want to recover $p(X_i = x_i)$.
- Brute-force sum: Define $\hat{p}(\mathbf{x}) = \prod_c \psi(\mathbf{x}_c)$

$$
\begin{aligned}
P(X_i = x_i) &= \frac{1}{Z} \sum_{x_1} \cdots \sum_{x_{i-1}} \sum_{x_{i+1}} \cdots \sum_{x_M} \hat{p}(\mathbf{x}) \\
Z &= \sum_{x_1} \cdots \sum_{x_M} \hat{p}(\mathbf{x})
\end{aligned}
$$

- On trees, can do sums quickly by dynamic programming.
  - Sum-product algorithm / belief propagation
- #P-hard
  - Approximate: Tree-reweighted belief propagation (TRW)
  - This paper: Same approximation as TRW, different algorithm.

Outline

Motivation

- TRW Convergence rates can be very slow.
    - If lucky, TRW = block coordate ascent on dual.
- TRW may fail to converge.
    - Damping converges in practice, slower.
    - Recent alternatives guarantee convergence. [Hazan & Shashua 2009, Globerson & Jaakkola 2007b]
    - Not claimed faster than TRW. TRW-S [Meltzer et al. 2009] is an exception.
- This paper: use a quasi-newton method on dual.
    - Line searches guarantee convergence.
    - Hopefully, faster convergence.

Motivation

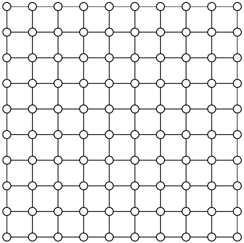
- TRW Convergence rates can be very slow.
  - If lucky, TRW = block coordate ascent on dual.
- TRW may fail to converge.
  - Damping converges in practice, slower.
  - Recent alternatives guarantee convergence. [Hazan & Shashua 2009, Globerson & Jaakkola 2007b]
  - Not claimed faster than TRW. TRW-S [Meltzer et al. 2009] is an exception.
- This paper: use a quasi-newton method on dual.
  - Line searches guarantee convergence.
  - Hopefully, faster convergence.

Motivation

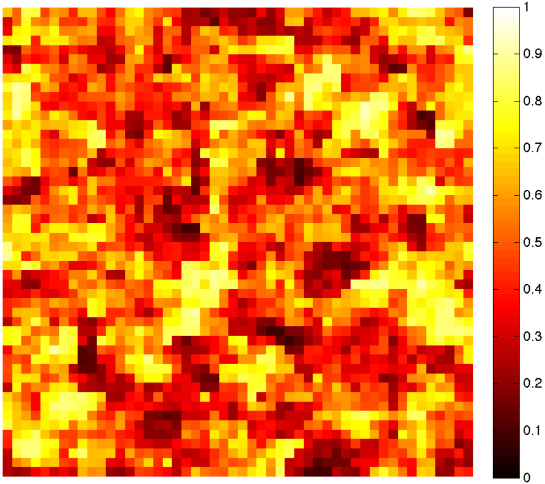
- TRW Convergence rates can be very slow.
  - If lucky, TRW = block coordate ascent on dual.
- TRW may fail to converge.
  - Damping converges in practice, slower.
  - Recent alternatives guarantee convergence. [Hazan & Shashua 2009, Globerson & Jaakkola 2007b]
  - Not claimed faster than TRW. TRW-S [Meltzer et al. 2009] is an exception.
- This paper: use a quasi-newton method on dual.
  - Line searches guarantee convergence.
  - Hopefully, faster convergence.

Ising Model

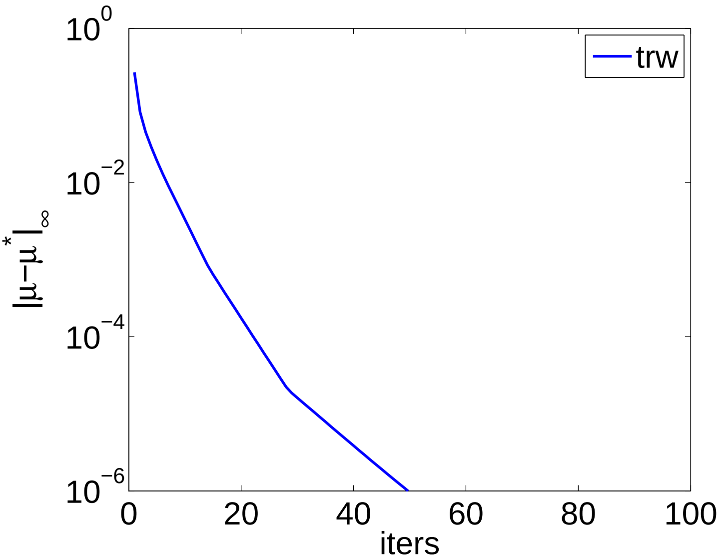
- $x_i \in \{-1, +1\}$
- $p(\mathbf{x}) \propto \prod_{ij} \exp\big(\theta(x_i, x_j)\big) \prod_i \exp(\theta(x_i))$
- $\theta(x_i) = \alpha_F x_i, \quad \alpha_F \in [-1, +1]$
- $\theta(x_i, x_j) = \alpha_I x_i x_j, \quad \alpha_I \in [0, T]$ for various T



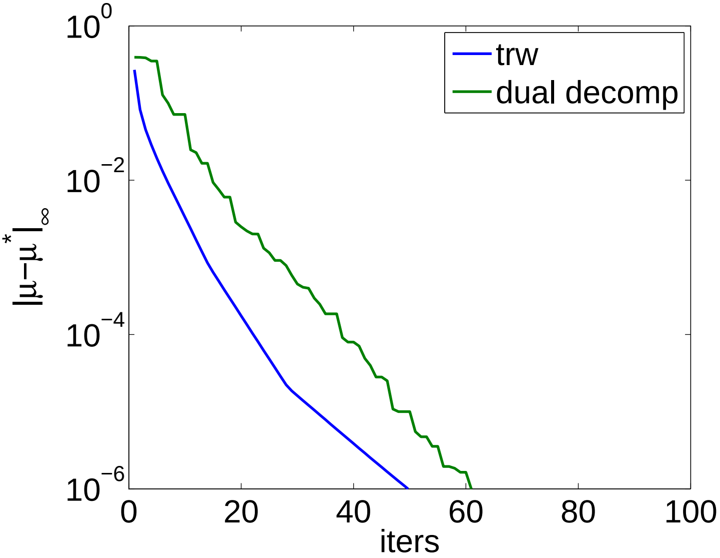
$$\theta(x_i, x_j) = \alpha_I x_i x_j, \quad \alpha_I \in [0, 1]$$



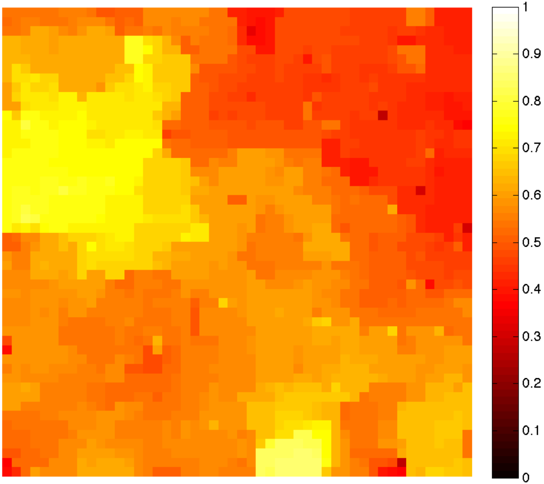
$$\theta(x_i, x_j) = \alpha_I x_i x_j, \quad \alpha_I \in [0, 1]$$



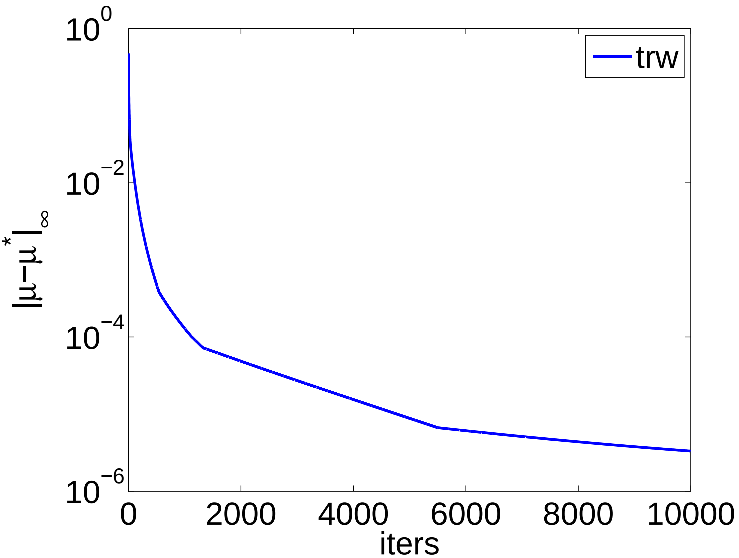
$$\theta(x_i, x_j) = \alpha_I x_i x_j, \quad \alpha_I \in [0, 1]$$



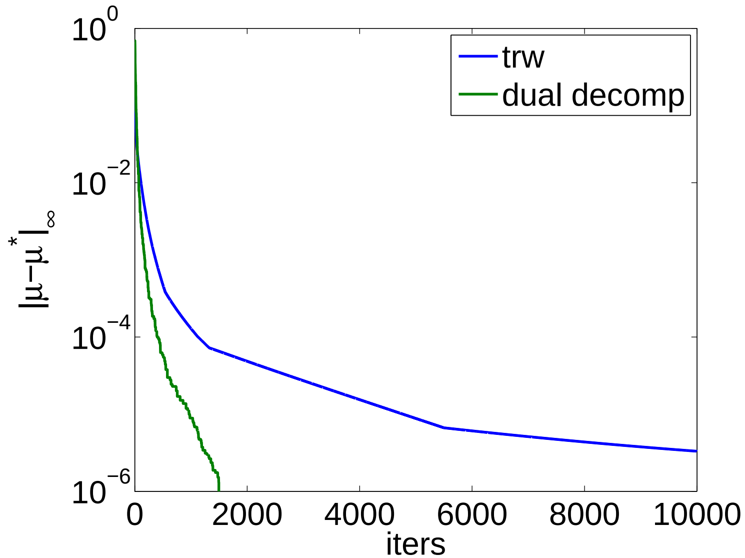
$$\theta(x_i, x_j) = \alpha_I x_i x_j, \quad \alpha_I \in [0, 3]$$



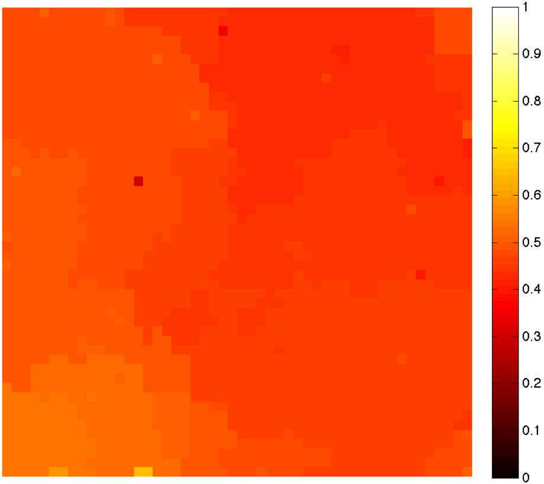
$$\theta(x_i, x_j) = \alpha_I x_i x_j, \quad \alpha_I \in [0, 3]$$



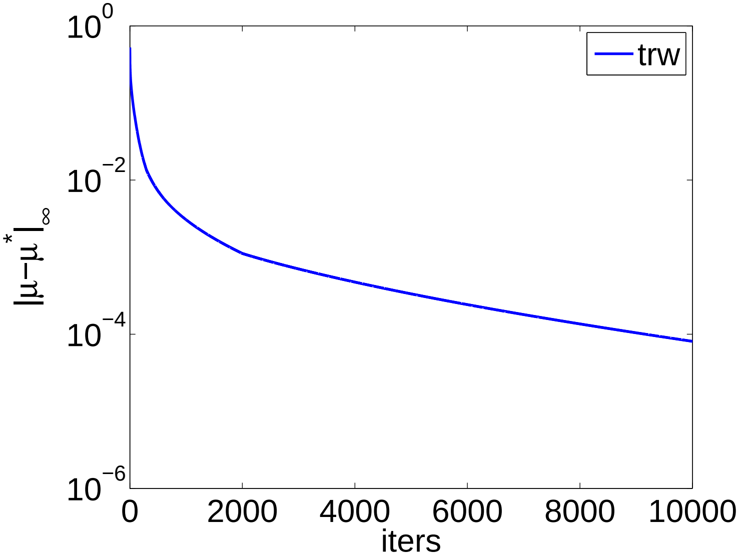
$$\theta(x_i, x_j) = \alpha_I x_i x_j, \quad \alpha_I \in [0, 3]$$



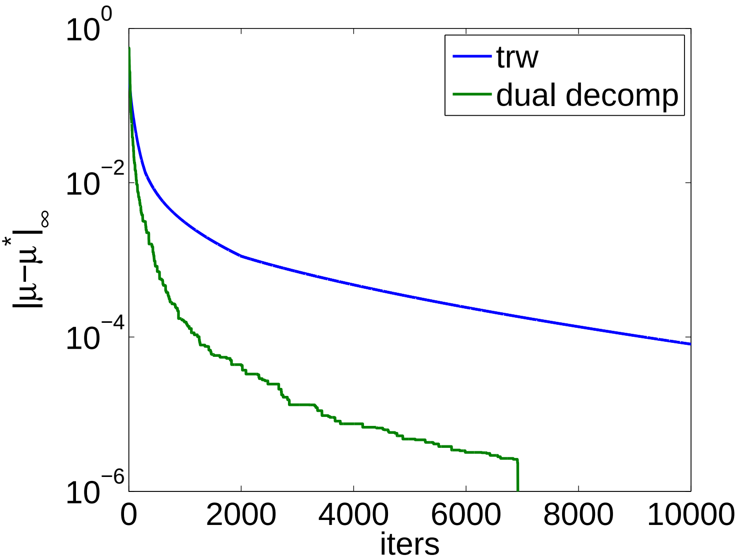
$$\theta(x_i, x_j) = \alpha_l x_i x_j, \quad \alpha_l \in [0, 5]$$



$$\theta(x_i, x_j) = \alpha_I x_i x_j, \quad \alpha_I \in [0, 5]$$



$$\theta(x_i, x_j) = \alpha_I x_i x_j, \quad \alpha_I \in [0, 5]$$



Outline

Wait a Second

Question: Why should I care about very accurately computing approximate marginals!?

Answer: You might not.

One reason to care:

- Number of iterations TRW needs for reasonable results is not easy to predict.

Wait a Second

Question: Why should I care about very accurately computing approximate marginals!?

Answer: You might not.

One reason to care:

- Number of iterations TRW needs for reasonable results is not easy to predict.

Why I Care

Want to fit a CRF with some loss $L(\theta) = M(\mu(\theta))$.

Algorithm (Domke, 2010):

1. Get μ by running TRW with parameters θ.
2. Compute $\frac{dM(\mu)}{d\mu}$
3. Get μ^+ by running TRW with parameters $\theta + r\frac{dM}{d\mu}$
4. $\frac{dL}{d\theta} \approx \frac{1}{r}(\mu^+ - \mu)$

Strong convergence needed for difference $\mu^+ - \mu$ to be meaniningful.

Why I Care

Want to fit a CRF with some loss $L(\theta) = M(\mu(\theta))$.

Algorithm (Domke, 2010):

1. Get μ by running TRW with parameters θ.
2. Compute $\frac{dM(\mu)}{d\mu}$
3. Get μ^+ by running TRW with parameters $\theta + r\frac{dM}{d\mu}$
4. $\frac{dL}{d\theta} \approx \frac{1}{r}(\mu^+ - \mu)$

Strong convergence needed for difference $\mu^+ - \mu$ to be meaniningful.

Why I Care

Want to fit a CRF with some loss $L(\theta) = M(\mu(\theta))$.

Algorithm (Domke, 2010):

1. Get μ by running TRW with parameters θ.
2. Compute $\frac{dM(\mu)}{d\mu}$
3. Get μ^+ by running TRW with parameters $\theta + r\frac{dM}{d\mu}$
4. $\frac{dL}{d\theta} \approx \frac{1}{r}(\mu^+ - \mu)$

Strong convergence needed for difference $\mu^+ - \mu$ to be meaniningful.

Outline

Dual Decomposition with Two subproblems

$$\max_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{x})$$

- Can quickly and exactly maximize $f(\mathbf{x}) + \mathbf{a} \cdot \mathbf{x}$.
- Can quickly and exactly maximize $g(\mathbf{x}) + \mathbf{b} \cdot \mathbf{x}$.

Dual Decomposition with Two subproblems

- Transform $\max_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{x})$ to a constrained problem:

$$\begin{array}{ll} \max_{\mathbf{x},\mathbf{y}} & f(\mathbf{x}) + g(\mathbf{y}) \\ \text{s.t.} & \mathbf{x} = \mathbf{y} \end{array}$$

- Leads to dual problem:

$$\min_{\lambda} h(\lambda), \quad h(\lambda) = \max_{\mathbf{x}} f(\mathbf{x}) + \lambda \cdot \mathbf{x} + \max_{\mathbf{y}} g(\mathbf{y}) - \lambda \cdot \mathbf{y}$$

Dual Decomposition with Two subproblems

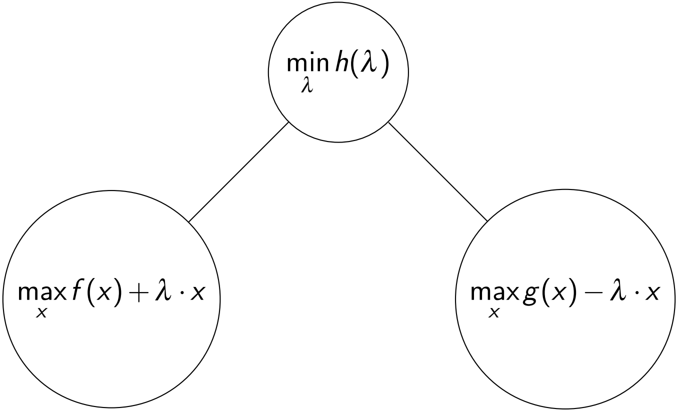
- Transform $\max_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{x})$ to a constrained problem:

$$\begin{array}{ll} \max_{\mathbf{x},\mathbf{y}} & f(\mathbf{x}) + g(\mathbf{y}) \\ \text{s.t.} & \mathbf{x} = \mathbf{y} \end{array}$$

- Leads to dual problem:

$$\min_{\lambda} h(\lambda), \quad \begin{array}{rcl} h(\lambda) & = & \max_{\mathbf{x}} f(\mathbf{x}) + \lambda \cdot \mathbf{x} \\ & + & \max_{\mathbf{y}} g(\mathbf{y}) - \lambda \cdot \mathbf{y} \end{array}$$

Dual Decomposition with Two subproblems



Dual Decomposition with N subproblems

$$\max_{\mathbf{x}} \sum_{i=1}^{N} f_i(\mathbf{x})$$

- Can quickly and exactly maximize $f_i(\mathbf{x}) + \mathbf{a}_i \cdot \mathbf{x}$, for all i.

Dual Decomposition with N subproblems

- Transform $\max_{\mathbf{x}} \sum_i f_i(\mathbf{x})$ to a constrained problem:

$$\begin{aligned} \max_{\{\mathbf{x}_i\}} \quad & \sum_i f_i(\mathbf{x}_i) \\ \text{s.t.} \quad & \mathbf{x}_i = \frac{1}{N} \sum_j \mathbf{x}_j \end{aligned}$$

- Leads to dual problem:

$$\min_{\lambda} h(\lambda), \quad h(\lambda) = \sum_i h_i(\lambda)$$

$$h_i(\lambda) = \max_{\mathbf{x}_i} f_i(\mathbf{x}_i) + (\lambda_i - \frac{1}{N} \sum_j \lambda_j) \cdot \mathbf{x}_i$$

Dual Decomposition with N subproblems

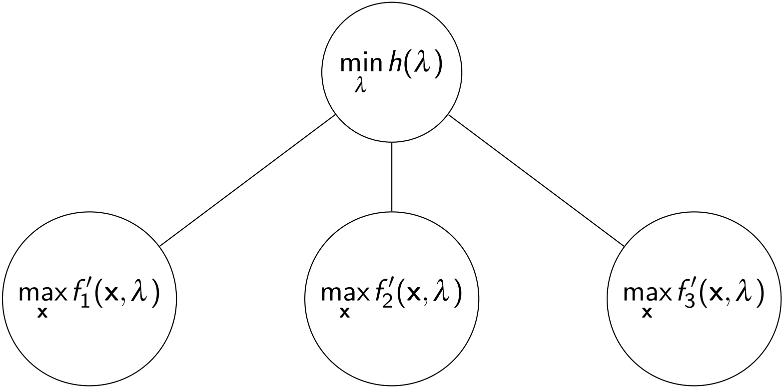
- Transform $\max_{\mathbf{x}} \sum_i f_i(\mathbf{x})$ to a constrained problem:

$$\begin{aligned} \max_{\{\mathbf{x}_i\}} \quad & \sum_i f_i(\mathbf{x}_i) \\ \text{s.t.} \quad & \mathbf{x}_i = \frac{1}{N} \sum_j \mathbf{x}_j \end{aligned}$$

- Leads to dual problem:

$$\begin{aligned} \min_{\lambda} h(\lambda), \quad h(\lambda) &= \sum_i h_i(\lambda) \\ h_i(\lambda) &= \max_{\mathbf{x}_i} f_i(\mathbf{x}_i) + (\lambda_i - \frac{1}{N} \sum_i \lambda_j) \cdot \mathbf{x}_i \end{aligned}$$

Dual Decomposition with N subproblems



$$f_i'(\mathbf{x}, \lambda) = f_i(\mathbf{x}_i) + (\lambda_i - \frac{1}{N} \sum_i \lambda_j) \cdot \mathbf{x}_i$$

Dual Decomposition with N subproblems

- Has been used extensively for MAP inference.
  - $h(\lambda)$ is non-differentiable.
- For marginal inference, $h(\lambda)$ is differentiable, convex.

Outline

Variational Inference

Can represent a graphical model in exponential family:

$$p(\mathbf{x};\theta) = \exp\big(\mathbf{f}(\mathbf{x})\cdot\theta - A(\theta)\big),\ A(\theta) = \log\sum_{\mathbf{x}}\exp\big(\mathbf{f}(\mathbf{x})\cdot\theta\big)$$

Can compute A as [Wainwright and Jordan]

$$A(\theta) = \max_{\mu\in\mathscr{M}} \theta\cdot\mu + H(\mu)$$

- $\mathscr{M}$ is marginal polytope (hard).
- H is entropy (hard).

Variational Inference

Exact inference: $A(\theta) = \max_{\mu \in \mathscr{M}} \theta \cdot \mu + H(\mu)$

TRW approximation: $B(\theta) = \max_{\mu \in \mathscr{L}} \theta \cdot \mu + \sum_T \rho_T H(\mu(T))$

- $\mathscr{L}$ - is marginal polytope (easy)
- $H(\mu(T))$ - entropy of marginals projected onto tree T (easy)

Our problem: how to compute B?

Variational Inference

Exact inference: $A(\theta) = \max_{\mu \in \mathscr{M}} \theta \cdot \mu + H(\mu)$

TRW approximation: $B(\theta) = \max_{\mu \in \mathscr{L}} \theta \cdot \mu + \sum_{T} \rho_T H(\mu(T))$

- $\mathscr{L}$ - is marginal polytope (easy)
- $H(\mu(T))$ - entropy of marginals projected onto tree T (easy)

Our problem: how to compute B?

Variational Inference

Exact inference: $A(\theta) = \max_{\mu \in \mathscr{M}} \theta \cdot \mu + H(\mu)$

TRW approximation: $B(\theta) = \max_{\mu \in \mathscr{L}} \theta \cdot \mu + \sum_T \rho_T H(\mu(T))$

- $\mathscr{L}$ - is marginal polytope (easy)
- $H(\mu(T))$ - entropy of marginals projected onto tree T (easy)

Our problem: how to compute B?

Variational Inference

Exact inference: $A(\theta) = \max_{\mu \in \mathscr{M}} \theta \cdot \mu + H(\mu)$

TRW approximation: $B(\theta) = \max_{\mu \in \mathscr{L}} \theta \cdot \mu + \sum_T \rho_T H(\mu(T))$

- $\mathscr{L}$ - is marginal polytope (easy)
- $H(\mu(T))$ - entropy of marginals projected onto tree T (easy)

Our problem: how to compute B?

Dual Decomposition for Marginal Inference

TRW approximation: $B(\theta) = \max_{\mu \in \mathscr{L}} \theta \cdot \mu + \sum_T \rho_T H(\mu(T))$

Theorem (main result):

$$B(\theta) = \min_{\{\theta^T\}} h(\{\theta^T\}) \quad \text{s.t.} \sum_{T:a \in T} \theta_a^T = \theta_a$$

$$h(\{\theta^T\}) = \sum_T B_T(\theta^T)$$

$$B_T(\theta^T) = \max_{\mu^T \in \mathscr{M}_T} \theta^T \cdot \mu^T + \rho_T H_T(\mu^T)$$

$B_T(\theta^T)$ is computable by running regular sum-product algorithm.

Dual Decomposition for Marginal Inference

TRW approximation: $B(\theta) = \max_{\mu \in \mathscr{L}} \theta \cdot \mu + \sum_T \rho_T H(\mu(T))$

Theorem (main result):

$$B(\theta) = \min_{\{\theta^T\}} h(\{\theta^T\}) \quad \text{s.t.} \sum_{T: a \in T} \theta_a^T = \theta_a$$

$$h(\{\theta^T\}) = \sum_T B_T(\theta^T)$$

$$B_T(\theta^T) = \max_{\mu^T \in \mathscr{M}_T} \theta^T \cdot \mu^T + \rho_T H_T(\mu^T)$$

$B_T(\theta^T)$ is computable by running regular sum-product algorithm.

Dual Decomposition for Marginal Inference

TRW approximation: $B(\theta) = \max_{\mu \in \mathcal{L}} \theta \cdot \mu + \sum_T \rho_T H(\mu(T))$

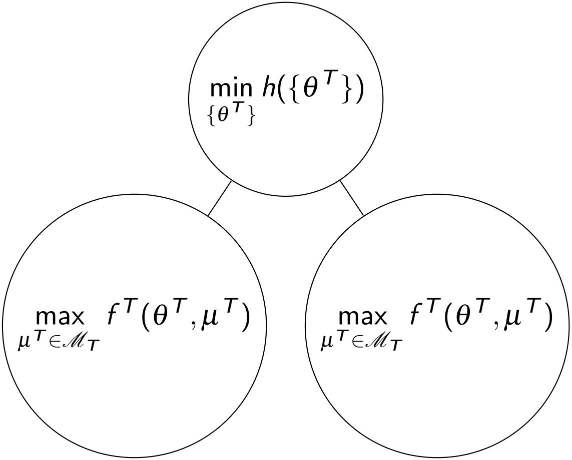
Theorem (main result):

$$B(\theta) \quad = \quad \min_{\{\theta^T\}} h(\{\theta^T\}) \quad \text{s.t.} \sum_{T:a \in T} \theta_a^T = \theta_a$$

$$\begin{aligned} h(\{\theta^T\}) &= \sum_T B_T(\theta^T) \\ B_T(\theta^T) &= \max_{\mu^T \in \mathcal{M}_T} \theta^T \cdot \mu^T + \rho_T H_T(\mu^T) \end{aligned}$$

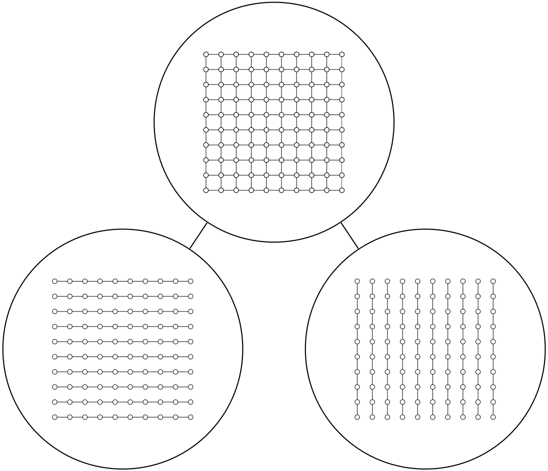
$B_T(\theta^T)$ is computable by running regular sum-product algorithm.

Dual Decomposition for Marginal Inference



$$f^T(\theta^T, \mu^T) = \theta^T \cdot \mu^T + \rho_T H_T(\mu^T)$$

Dual Decomposition for Marginal Inference



Dual Decomposition for Marginal Inference

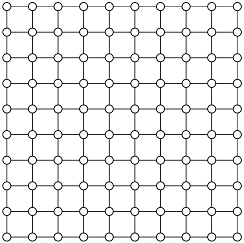
Inference: Plug $\min_{\{\theta^T\}} \sum_T B_T(\theta^T)$ into L-BFGS.

- Guarantees convergence. (Line searches)
- Fast convergence rates.

Outline

Ising Model

- $x_i \in \{-1, +1\}$
- $p(\mathbf{x}) \propto \prod_{ij} \exp\big(\theta(x_i, x_j)\big) \prod_i \exp(\theta(x_i))$
- $\theta(x_i) = \alpha_F x_i$
- $\theta(x_i, x_j) = \alpha_I x_i x_j$



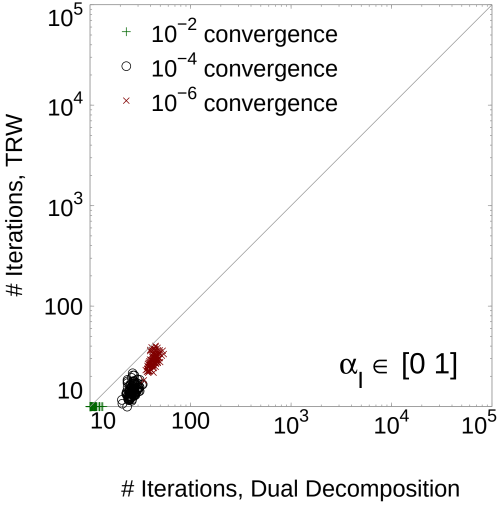
Algorithms

Algorithms Compared:

- Dual Decomposition + L-BFGS
- TRW
- TRW with damping of 1/2 in the log-domain.
- TRW-S [Meltzer et al. 2009]

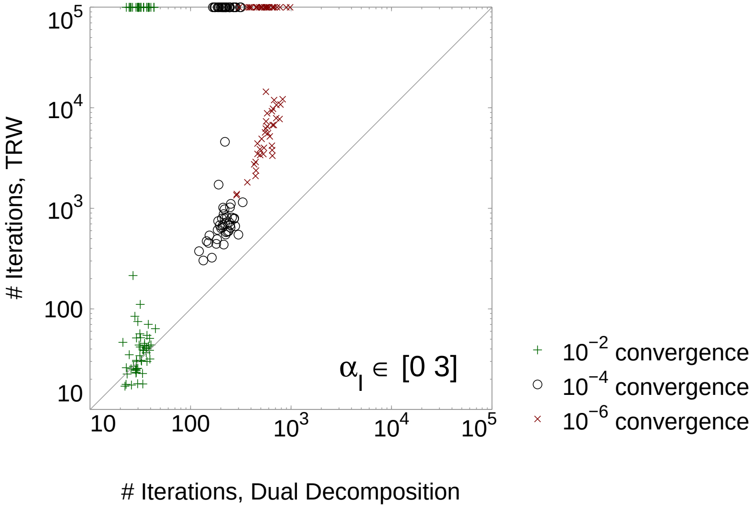
Max of 10^5 iterations allowed.

Dual Decomp vs. TRW

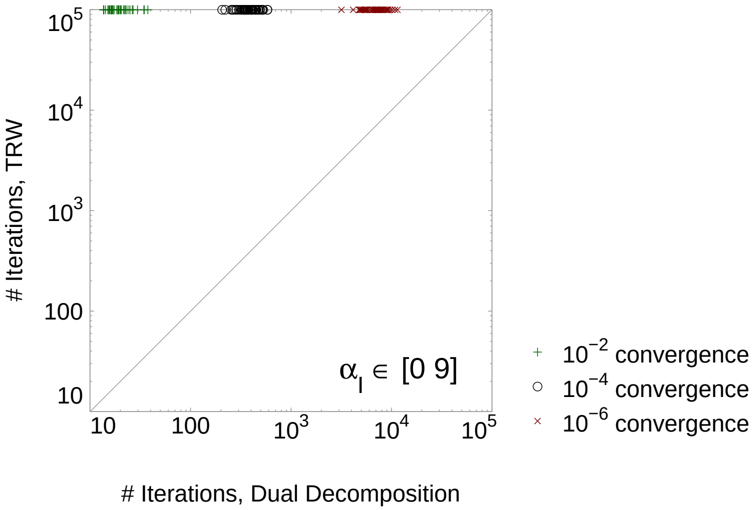


10^{-2} convergence
10^{-4} convergence
10^{-6} convergence

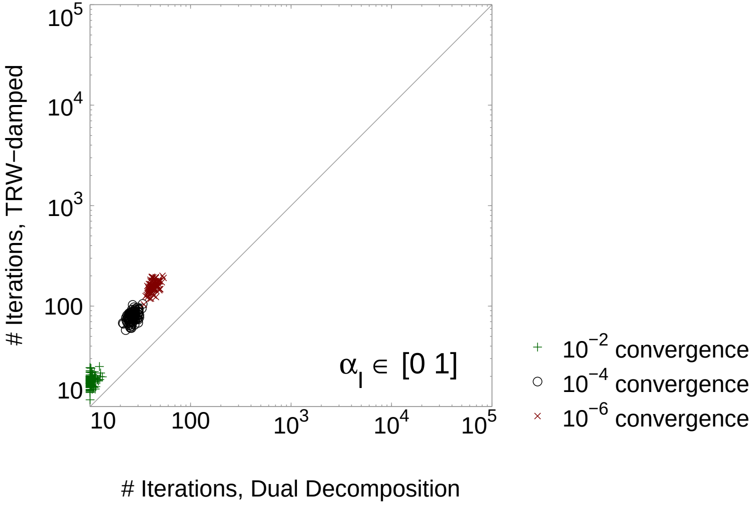
Dual Decomp vs. TRW



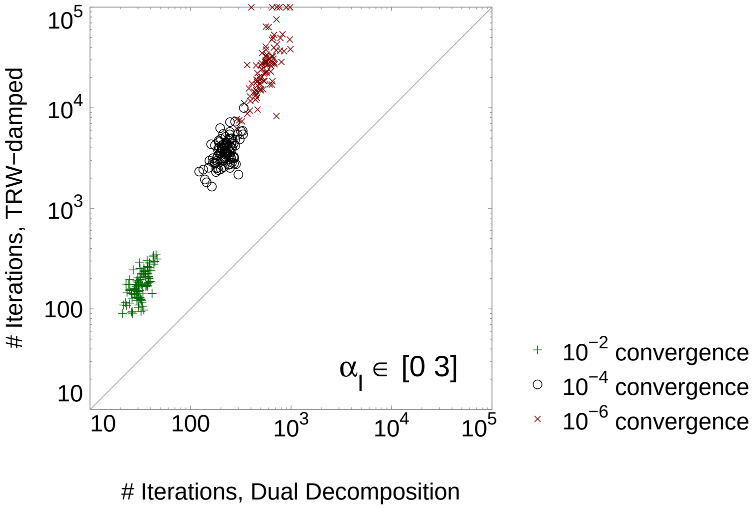
Dual Decomp vs. TRW



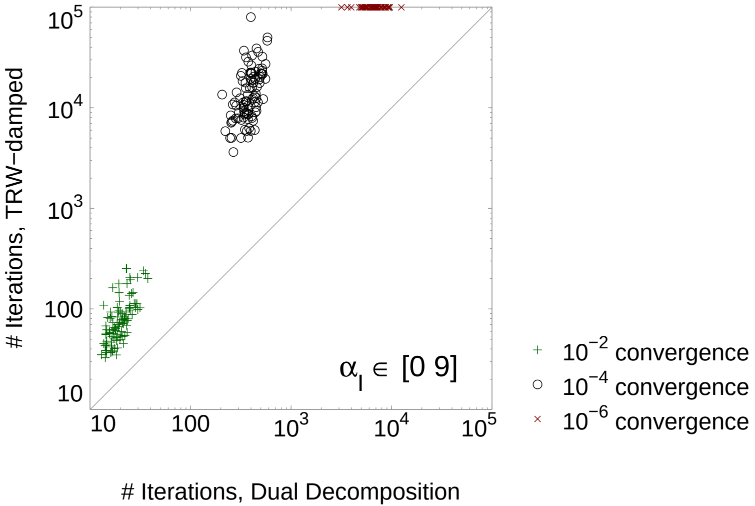
Dual Decomp vs. TRW-damped



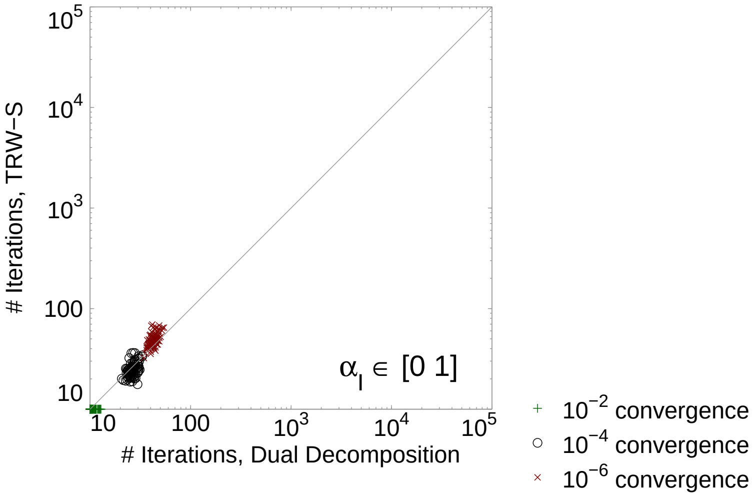
Dual Decomp vs. TRW-damped



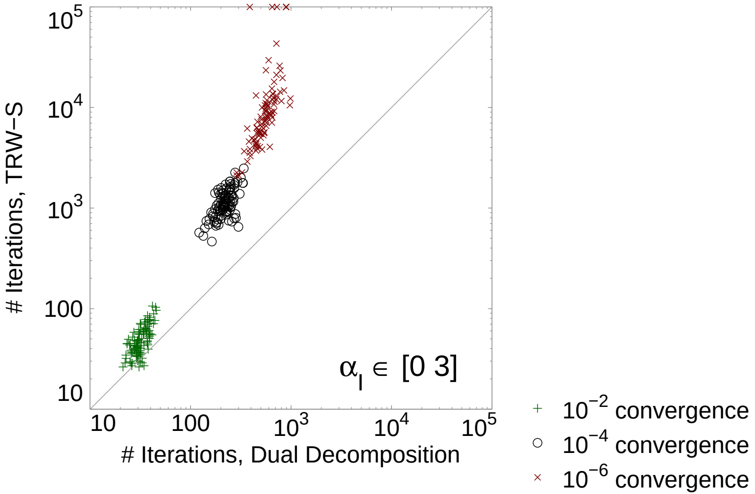
Dual Decomp vs. TRW-damped



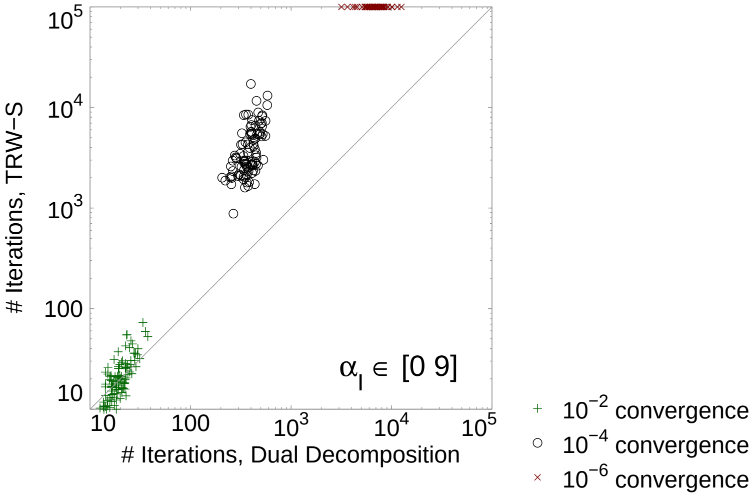
Dual Decomp vs. TRW-S



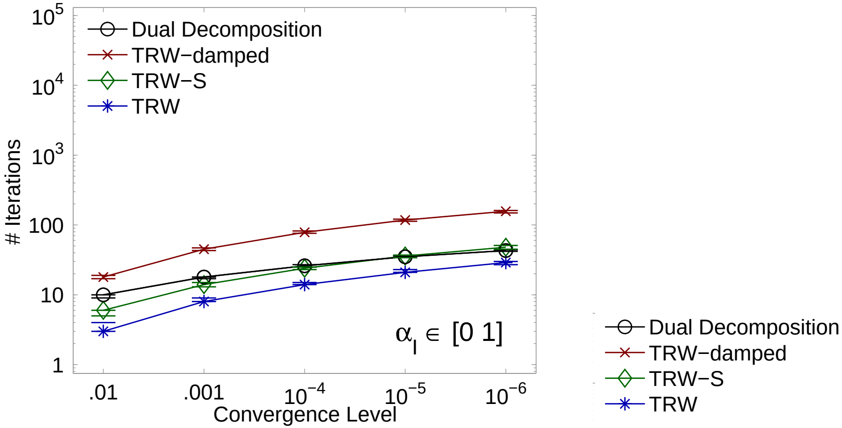
Dual Decomp vs. TRW-S



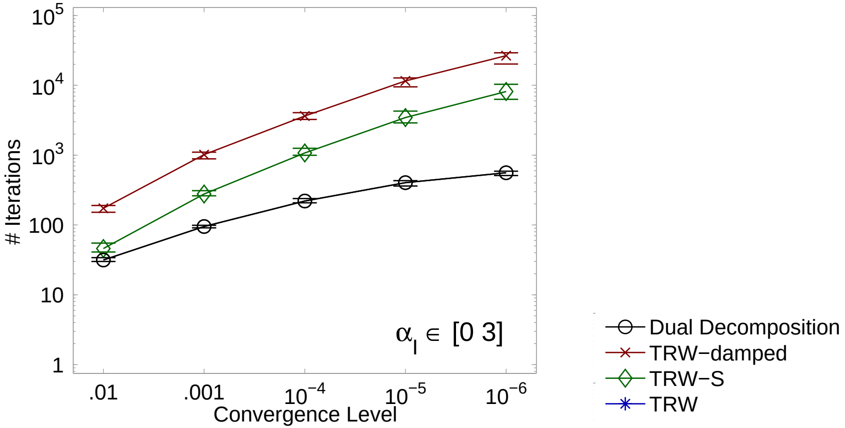
Dual Decomp vs. TRW-S



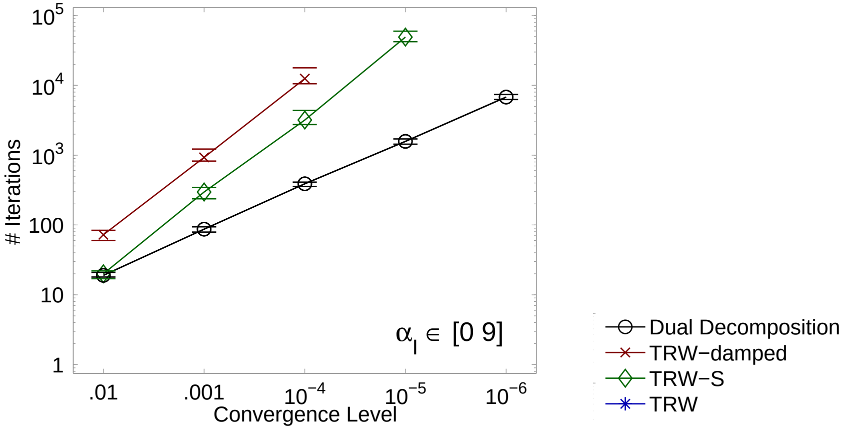
Convergence



Convergence



Convergence



Outline

Conclusions

- Dual Decomposition
  - Faster on “hard” problems or if strong convergence needed.
- Caveats
  - Not really faster on “easy” problems.
  - Restriction on tree distribution P(T).

Conclusions

- Dual Decomposition
  - Faster on "hard" problems or if strong convergence needed.
- Caveats
  - Not really faster on "easy" problems.
  - Restriction on tree distribution $P(T)$.

Conclusions

Thank you

Graphical models toolbox: `people.rit.edu/jcdicsa/`
(Dual decomposition coming soon.)