A Derivation of the u-updates in DD-ADMM

By equating to zero $\nabla_u \mathcal{A}_\rho$, we obtain

$$u^{t+1}(r) = \frac{1}{\delta(r)} \sum_{s:r\in\bar{\mathcal{R}}_s} \left(z_s^{t+1}(r) - \rho^{-1}\lambda_s^t(r)\right). \quad (22)$$

Second, note that if we initialize λ to zero, it will always satisfy $\sum_{s:r\in\bar{\mathcal{R}}_s} \lambda_s(r) = 0, \forall r \in \mathcal{R}$. (This can be easily proved by induction by inspecting the updates in Eqs. 10 and 22.) Those conditions are simply the constraints of problem D (Eq. 7); hence, each iterate of the ADMM is guaranteed to produce a dual feasible solution. Furthermore, this property allows us to simplify the u-updates (Eq. 22) to:

$$u^{t+1}(r) = \frac{1}{\delta(r)} \sum_{s:r\in\bar{\mathcal{R}}_s} z_s^{t+1}(r); \quad (23)$$

this is precisely the average operation used in the subgradient algorithm (cf. Eq. 9).

B Pairwise Factors

Define $a_{12} = \rho^{-1}\theta_{12}$, and assume that $a_{12} \geq 0$, without loss of generality (if $a_{12} < 0$, we recover this case by redefining $a'_1 = a_1 + a_{12}$, $a'_2 = 1 - a_2$, $a'_{12} = -a_{12}$, $z'_2 = 1 - z_2$, $z'_{12} = z_1 - z_{12}$). Then, the lower bound constraints $z_{12} \geq z_1 + z_2 - 1$ and $z_{12} \geq 0$ are always inactive and can be ignored. By inspecting the KKT conditions we obtain the following closed-form solution: $z_{12}^\star = \min\{z_1^\star, z_2^\star\}$ and $\langle z_1^\star, z_2^\star\rangle =$

$$\begin{cases} ([a_1]_{\mathbb{U}}, [a_2 + a_{12}]_{\mathbb{U}}) & \text{if } a_1 > a_2 + a_{12} \\ ([a_1 + a_{12}]_{\mathbb{U}}, [a_2]_{\mathbb{U}}) & \text{if } a_2 > a_1 + a_{12} \\ ([(a_1 + a_2 + a_{12})/2]_{\mathbb{U}}, & \\ \quad [(a_1 + a_2 + a_{12})/2]_{\mathbb{U}}) & \text{otherwise,} \end{cases} \quad (24)$$

where $[x]_{\mathbb{U}} = \min\{\max\{x, 0\}, 1\}$ denotes the projection (clipping) onto the unit interval. Hence, for this case, Eq. 14 can be solved in constant time.

C Uniqueness Quantification and XOR

The algorithm for computing a projection onto the simplex is depicted as Alg. 2 (Duchi et al., 2008).

D Existential Quantification and OR

The following procedure computes this projection:

1. For $i = 1, \ldots, n$, set $z_i = [a_i]_{\mathbb{U}}$.

Algorithm 2 Projection onto simplex

```
Input: $\langle a_1, \ldots, a_n\rangle$
Sort $\langle a_1, \ldots, a_n\rangle$ into $\langle b_1, \ldots, b_n\rangle$: $b_1 \geq \ldots \geq b_n$
Find $\rho = \max\left\{j \in [n] \mid b_j - \frac{1}{j}\left(\sum_{r=1}^j b_r - 1\right) > 0\right\}$
Define $\tau = \frac{1}{\rho}\left(\sum_{r=1}^\rho b_r - 1\right)$
Output: $\langle z_1, \ldots, z_n\rangle$ with $z_i = \max\{a_i - \tau, 0\}$.
```

2. If $\sum_{i=1}^n z_i \geq 1$, return $\langle z_1, \ldots, z_n\rangle$. Else, project $\langle a_1, \ldots, a_n\rangle$ onto the simplex.

The runtime is also $O(n \log n)$.

To see that this procedure is correct, we need the following

Lemma 1. *Consider a problem of the form*

$$P: \quad \min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}) \quad \text{s.t.} \quad g(\boldsymbol{x}) \leq 0, \quad (25)$$

where $\mathcal{X}$ is nonempty convex subset of $\mathbb{R}^d$ and $f : \mathcal{X} \to \mathbb{R}$ and $g : \mathcal{X} \to \mathbb{R}$ are convex functions. Suppose that the problem (25) *is feasible and bounded below, and let $\mathcal{A}$ be the set of solutions of the* relaxed problem $\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x})$*, i.e.* $\mathcal{A} = \operatorname{Argmin}_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x})$. *Then:*

1. *if for some $\tilde{\boldsymbol{x}} \in \mathcal{A}$ we have $g(\tilde{\boldsymbol{x}}) \leq 0$, then $\tilde{\boldsymbol{x}}$ is also a solution of the original problem P;*
2. *otherwise (if for all $\tilde{\boldsymbol{x}} \in \mathcal{A}$ we have $g(\tilde{\boldsymbol{x}}) > 0$), the inequality constraint is necessarily active in P, i.e., problem P is equivalent to* $\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x})$ *s.t.* $g(\boldsymbol{x}) = 0$.

Proof. Let f^* be the optimal value of P. The first statement is obvious: since $\tilde{\boldsymbol{x}}$ is a solution of a relaxed problem we have $f(\tilde{\boldsymbol{x}}) \leq f^*$; hence if $\tilde{\boldsymbol{x}}$ is feasible this becomes an equality. For the second statement, assume that $\exists \boldsymbol{x} \in \mathcal{X}$ s.t. $g(\boldsymbol{x}) < 0$ (otherwise, the statement holds trivially). The nonlinear Farkas' lemma (Prop. 3.5.4, p. 204, of Bertsekas et al. (2003)) implies that there exists some $\lambda^* \geq 0$ s.t. $f(\boldsymbol{x}) - f^* + \lambda^* g(\boldsymbol{x}) \geq 0$ holds for all $\boldsymbol{x} \in \mathcal{X}$. In particular, this also holds for an optimal $\boldsymbol{x}^*$ (i.e., such that $f^* = f(\boldsymbol{x}^*)$), which implies that $\lambda^* g(\boldsymbol{x}^*) \geq 0$. However, since $\lambda^* \geq 0$ and $g(\boldsymbol{x}^*) \leq 0$ (since $\boldsymbol{x}^*$ has to be feasible), we also have $\lambda^* g(\boldsymbol{x}^*) \leq 0$, *i.e.*, $\lambda^* g(\boldsymbol{x}^*) = 0$. Now suppose that $\lambda^* = 0$. Then we have $f(\boldsymbol{x}) - f^* \geq 0, \forall \boldsymbol{x} \in \mathcal{X}$, which implies that $\boldsymbol{x}^* \in \mathcal{A}$ and contradicts the assumption that $g(\tilde{\boldsymbol{x}}) > 0, \forall \tilde{\boldsymbol{x}} \in \mathcal{A}$. Hence we must have $g(\boldsymbol{x}^*) = 0$. □

Hence, the validity of the second step stems from the fact that, if the relaxed problem in the first step does not return a feasible point, then the constraint $\sum_{i=1}^{n} z_i \geq 1$ has to be active, *i.e.*, we must have $\sum_{i=1}^{n} z_i = 1$. This, in turn, implies that $z_i \leq 1, \forall i$, hence the problem reduces to the XOR case.

E Logical Variable Assignments

Note that $\mathcal{Z}_{\text{OR-OUT}} = \mathcal{Z}' \cap \mathcal{Z}''$ where $\mathcal{Z}' = \{\langle z_0, \ldots, z_n \rangle \mid z_0 \leq z_i,\ \forall i = 1, \ldots, n\}$ and $\mathcal{Z}'' = \{\langle z_0, \ldots, z_n \rangle \in [0,1]^n \mid z_0 \geq \sum_{i=1}^{n} z_i\}$.

The following procedure computes the desired projection:

1. Set $\langle z'_0, \ldots, z'_n \rangle$ as the projection of $\langle a_0, \ldots, a_n \rangle$ onto $\mathcal{Z}'$. This can be done with a sort in $O(n \log n)$, via Alg. 3.
2. Clip onto the unit cube: for $i = 1, \ldots, n$, set $z_i = [z'_i]_{\mathbb{U}}$. If the result lies in $\mathcal{Z}''$, return $\langle z_0, \ldots, z_n \rangle$. Otherwise, go to step 3.
3. Project $\langle a_0, \ldots, a_n \rangle$ onto $\{\langle z_0, \ldots, z_n \rangle \in [0,1]^n \mid z_0 = \sum_{i=1}^{n} z_i\}$. Note that this corresponds to the slave subproblem of the XOR-WITH-OUTPUT factor, hence can be solved in $O(n \log n)$ by projecting onto the simplex.

The total runtime is $O(n \log n)$.

To prove the correctness of this procedure, we will show that steps 1–2 are computing a projection onto the set: $\tilde{\mathcal{Z}} = \{\langle z_0, \ldots, z_n \rangle \in [0,1]^n \mid z_0 \leq z_i,\ \forall i = 1, \ldots, n\}$. If that is true, then Lemma 1 ensures that the procedure is correct. Note that steps 1–2 are a composition of two projections. In general, the composition of individual projections is not equivalent to projecting onto the intersection. In particular, commuting the two steps would make our procedure incorrect. However, it turns out that the sequence of these two projections correspond to the first iteration of Dykstra's projection algorithm (Boyle and Dykstra, 1986) applied to sets $\mathcal{Z}'$ and $[0,1]^n$; and that Dykstra's converges in one iteration for this particular case (Martins et al., 2011, supplementary material).

F Linear Program for the Head Automaton

In this section, we derive the linear program associated with the sibling-based head automaton used by Koo et al. (2010). Let $\langle t_0, \ldots, t_n \rangle$ be a sentence with n words, where t_0 is a dummy root symbol. For each word i, we will cast the head automaton problem as an LP. Without loss of generality, assume that $i = 0$ and the sequence of siblings lie on the right side of the root; the general case follows easily. Let:

- s_j be the score associated with word t_j being a modifier of t_0,
- s_{0j} be the score associated with word t_j being the *first* modifier of t_0,
- s_{jk} be the score associated with words t_j and t_k being *consecutive* siblings.

The problem is equivalent to that of finding a Viterbi path $\langle y_1, \ldots, y_n \rangle$ for a chain model whose possible states for Y_j are $\{0, \ldots, j\}$; the event $Y_j = a$ means that "the last modifier, up to t_j, is t_a." Between words t_j and t_{j+1}, only two transitions may occur: either $Y_{j+1} = y_j$ (which happens if t_{j+1} is not a modifier), or $Y_{j+1} = j + 1$ (which happens otherwise). Since this is a chain model, the marginal polytope is exactly characterized by *local consistency* constraints (Wainwright and Jordan, 2008) and *hard* constraints. Let $\mu_i(a)$ be the posterior marginal for the event $Y_i = a$, and $\mu_{i,i+1}(a, b)$ the posterior marginal for the event $Y_i = a \wedge Y_{i+1} = b$. Local consistency constraints assert that

$$\textstyle\sum_{a=0}^{i} \mu_i(a) = 1, \quad i \in [n] \tag{26}$$

$$\textstyle\sum_{b=0}^{i+1} \mu_{i,i+1}(a, b) = \mu_i(a), \quad i \in [n], a \in [i] \tag{27}$$

$$\textstyle\sum_{a=0}^{i} \mu_{i,i+1}(a, b) = \mu_{i+1}(b), \quad i \in [n], b \in [i+1] \tag{28}$$

$$\mu_i(a) \geq 0, \quad i \in [n], a \in [i] \tag{29}$$

$$\mu_{i,i+1}(a, b) \geq 0, \quad i \in [n], a \in [i], b \in [i+1]. \tag{30}$$

Algorithm 3 Projection onto $\mathcal{Z}'$

```
**Input:** $\langle a_0, \ldots, a_n \rangle$
Sort $a_1, \ldots, a_n$ into $b_1 \geq \ldots \geq b_n$
Find $\rho = \min \left\{ j \in [n] \mid \frac{1}{j} \left( a_0 + \sum_{r=1}^{j-1} b_r \right) > b_j \right\}$
Define $\tau = \frac{1}{\rho} \left( a_0 + \sum_{r=1}^{\rho-1} b_r \right)$
**Output:** $\langle z_0, \ldots, z_n \rangle$ with $z_0 = \tau$ and $z_i = \min\{a_i, \tau\}, i = 1, \ldots, n$.
```

Hard constraints assert that impossible configurations must receive zero marginals:

$$\mu_{i,i+1}(a,b) = 0, \quad i \in [n], a \in [i], b \notin \{a, i+1\}. \tag{31}$$

Plugging (31) in (27)–(28) yields:

$$\mu_{i,i+1}(a,a) + \mu_{i,i+1}(a,i+1) = \mu_i(a), \quad i \in [n], a \in [i] \tag{32}$$

$$\mu_{i,i+1}(b,b) = \mu_{i+1}(b), \quad i \in [n], b \in [i+1] \tag{33}$$

$$\textstyle\sum_{a=0}^{i} \mu_{i,i+1}(a,i+1) = \mu_{i+1}(i+1), \quad i \in [n], \tag{34}$$

and plugging further (33) in (32) yields:

$$\mu_{i+1}(a) + \mu_{i,i+1}(a,i+1) = \mu_i(a), \quad i \in [n], a \in [i]. \tag{35}$$

The marginal polytope is thus characterized by (26), (35), (34), and (29)–(30). We next make the variable replacements

- $z_i \triangleq \mu_i(i)$, the posterior marginal for the event that t_i is a modifier;
- $z_{a(i+1)} \triangleq \mu_{i,i+1}(a,i+1)$, the posterior marginal for the event that t_a and t_{i+1} are consecutive siblings;
- $\omega_{ai} \triangleq \mu_i(a)$, the posterior marginal for the event that, up to t_i, the last modifier is t_a.

The overall optimization problem becomes that of maximizing

$$\sum_{j=1}^{n} s_j z_j + \sum_{j=0}^{n} \sum_{k=j+1}^{n} s_{jk} z_{jk} \tag{36}$$

subject to:

$$\omega_{ii} = z_i, \quad i \in [n] \tag{37}$$

$$\sum_{a=0}^{i} \omega_{ai} = 1, \quad i \in [n] \tag{38}$$

$$\omega_{a(i+1)} + z_{a(i+1)} = \omega_{ai}, \quad i \in [n], a \in [i] \tag{39}$$

$$\sum_{a=0}^{i} z_{a(i+1)} = z_{i+1}, \quad i \in [n] \tag{40}$$

$$\omega_{ai} \geq 0, \quad i \in [n], a \in [i] \tag{41}$$

$$z_{a(i+1)} \geq 0, \quad i \in [n], a \in [i]. \tag{42}$$

Introducing head automata for each word $t_0, \ldots, t_n$ yields $O(n^3)$ variables and constraints. Therefore this formulation is as costly as the one employed in Martins et al. (2009a), while much simpler and, unlike the latter, *exact*.

Examining constraints (38) and (41), we recognize the linear equations that define the marginal polytope $\mathcal{Z}_{\text{XOR}}$ (cf. Eq. 19). Similarly, constraints (39) and (41–42) define the marginal polytope of a XOR-WITH-OUTPUT factor, and so do (40) and (42). Writing the corresponding logical constraints yields the expressions in the fifth row of Table 1.

We now show that, if we take the multi-commodity flow formulation of Martins et al. (2009a) and replace the consecutive-sibling constraints there by the ones in (37)–(42), then the resulting LP has *exactly* the same solution that is found by the dual decomposition method of Koo et al. (2010) with sibling head automata.[16] This is done by showing that the two *local polytope approximations* are the same. Moreover, both are tighter approximations than the one resulting from the single commodity flow formulation of Martins et al. (2009a).

The local polytope in Koo et al. (2010) is of the form

$$\bar{\mathcal{Z}} = \left\{ (z_{\text{tree}}, z_{\text{head}}) \;\middle|\; \begin{array}{l} z_{\text{tree}} \in \mathcal{Z}_{\text{tree}} \\ z_{\text{head}} \in \mathcal{Z}_{\text{head}} \\ z_{\text{tree}} \sim z_{\text{head}} \end{array} \right\}, \tag{43}$$

where $\mathcal{Z}_{\text{tree}}$ is the directed spanning tree polytope, and $\mathcal{Z}_{\text{head}}$ is the marginal polytope associated with the head automata, which is characterized by (37)–(42);[17] the requirement for overlap agreement is the same in Martins et al. (2009a) and Koo et al. (2010), hence we only need to concern about the directed spanning tree polytope $\mathcal{Z}_{\text{tree}}$. The multi-commodity flow formulation of Martins et al. (2009a) adds extra variables for *flows* in the arcs, and it is shown that for an arc-factored model the formulation is exact. This means that by projecting out the flow variables, the constraint space in Martins et al. (2009a)

[16]However, this does not apply to the head automata model of Koo et al. (2010) with siblings and grandparents, whose relaxation appears to be tighter than just adding grandparent variables and constraints as in Martins et al. (2009a).

[17]To be precise, (37)–(42) define a polytope in a larger space (with extra dimensions for the auxiliary ω-variables). However the projection of this polytope onto the subspace where the z-variables live equals $\mathcal{Z}_{\text{head}}$.

becomes exactly the spanning tree polytope $\mathcal{Z}_{\text{tree}}$. Therefore, the local polytope $\bar{\mathcal{Z}}_{\text{mc}}$ in Martins et al. (2009a) *with the corrections above* equals the one in (43). In contrast, the formulation with single commodities is not exact for the arc-factored model, and as a consequence, the polytope $\bar{\mathcal{Z}}_{\text{sc}}$ is an outer bound of $\mathcal{Z}$. In sum, we have the chain:

$$\mathcal{Z} \subseteq \bar{\mathcal{Z}} = \bar{\mathcal{Z}}_{\text{mc}} \subseteq \bar{\mathcal{Z}}_{\text{sc}} \tag{44}$$

where $\mathcal{Z}$ is the true (intractable) marginal polytope for this problem.

G Error Analysis

Fig. 5 shows examples of parses that were correctly predicted by the full model, but not by the G+CS model.

In (A), the G+CS model has predicted *1987* as the head of *about*. The full model got it right, arguably due to the nonprojectivity features that find the non-projective arc *lesson*$\rightarrow$*about* to be likely.

In (B), the G+CS model attached *further* to *retailers*. The word *further* forms an adverbial phrase which enjoys considerable freedom about where it can be placed in a sentence. Hence, features that look at *all siblings* attached to a head word (rather than just consecutive ones) may help parsing sentences without a rigid word ordering.

(C) is an example where *path* features seem to help. The G+CS model predicted 3 arcs incorrectly, because it assumed that *policy won't become clear for months* was a phrase (hence it predicted *$\rightarrow$*translate*$\rightarrow$*wo(n't)*$\rightarrow$*policy*). The full model may have found unlikely the long path that would descend from *translate*, and prefered a more horizontal parse.

Example (D) seems simple to parse; the word *snowball*, however, was incorrectly tagged as a verb by the part-of-speech tagger and confused the G+CS model, which predicted *to*$\rightarrow$*snowball*$\rightarrow$*effect*. The full model got it right, arguably because of the bigram features, which give a low score to configurations in which two consecutive words (in this case *a* and *snowball*) have crossing dependency attachments in opposite sides. This shows that a parser with many features may gain robustness to errors in the pipeline.

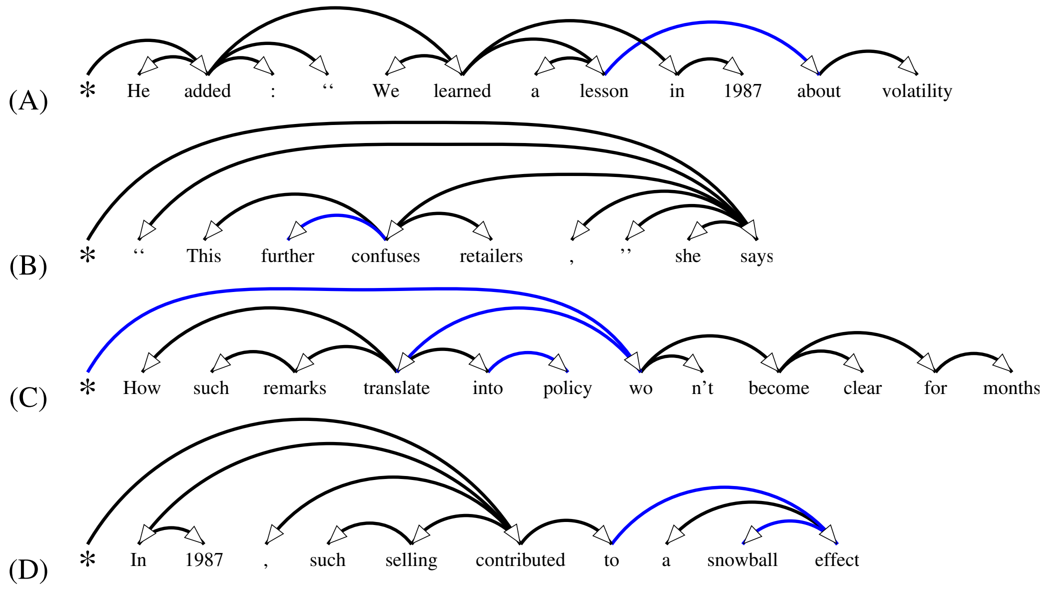
(A)
* He added : ‘‘ We learned a lesson in 1987 about volatility
(B)
* ‘‘ This further confuses retailers , ’’ she says
(C)
* How such remarks translate into policy wo n’t become clear for months
(D)
* In 1987 , such selling contributed to a snowball effect

Figure 5: Sentences of the English non-projective dataset (CoNLL 2008) that were correctly parsed by the full model, but not by the G+CS model. Arcs shown in blue are those that were missed by the G+CS model. See text for an explanation.