# Identifiability and Unmixing of Latent Parse Trees

**Daniel Hsu**
Microsoft Research

**Sham M. Kakade**
Microsoft Research

**Percy Liang**
Stanford University

## Abstract

This paper explores unsupervised learning of parsing models along two directions. First, which models are identifiable from infinite data? We use a general technique for numerically checking identifiability based on the rank of a Jacobian matrix, and apply it to several standard constituency and dependency parsing models. Second, for identifiable models, how do we estimate the parameters efficiently? EM suffers from local optima, while recent work using spectral methods [1] cannot be directly applied since the topology of the parse tree varies across sentences. We develop a strategy, unmixing, which deals with this additional complexity for restricted classes of parsing models.

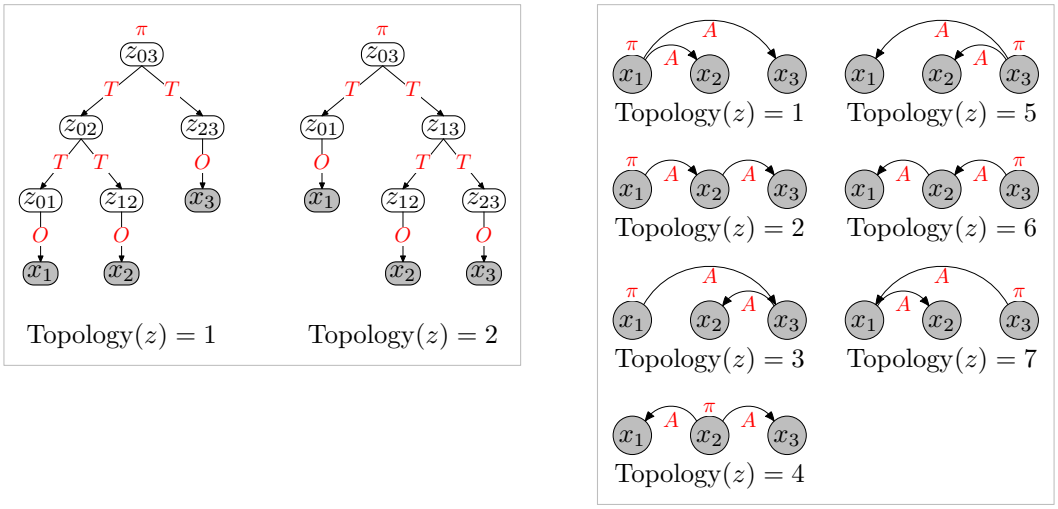
## 1 Introduction

Generative parsing models, which define joint distributions over sentences and their parse trees, are one of the core techniques in computational linguistics. We are interested in the unsupervised learning of these models [2–6], where the goal is to estimate the model parameters given only examples of sentences. Unsupervised learning can fail for a number of reasons [7]: model misspecification, non-identifiability, estimation error, and computation error. In this paper, we delve into two of these issues: identifiability and computation. In doing so, we confront a central challenge of parsing models—that the topology of the parse tree is unobserved and varies across sentences. This is in contrast to standard phylogenetic models [8] and other latent tree models for which there is a single fixed global tree across all examples [9].

A model is identifiable if there is enough information in the data to pinpoint the parameters (up to some trivial equivalence class); establishing the identifiability of a model is often a highly non-trivial task. A classic result of Kruskal [10] has been employed to prove the identifiability of a wide class of latent variable models, including hidden Markov models and certain restricted mixtures of latent tree models [11–13]. However, these techniques cannot be directly applied to parsing models since the tree topology varies over an exponential set of possible topologies. Instead, we turn to techniques from algebraic geometry [14–17]; we show that a simple numerical procedure can be used to check identifiability for a wide class of models in NLP. Using this tool, we discover that probabilistic context-free grammars (PCFGs) are non-identifiable, but that simpler PCFG variants and dependency models are identifiable.

The most common way to estimate unsupervised parsing models is by using local techniques such as EM [18] or MCMC sampling [19], but these methods can suffer from local optima and slow mixing. Meanwhile, recent work [1,20–23] has shown that spectral methods can be used to estimate mixture models and HMMs with provable guarantees. These techniques express low-order moments of the observable distribution as a product of matrix parameters and use eigenvalue decomposition to recover these matrices. However, these methods are not directly applicable to parsing models because the tree topology again varies non-trivially. To address this, we propose a new technique, *unmixing*. The main idea is to express moments of the observable distribution as a mixture over the possible topologies. For restricted parsing models, the moments for a fixed tree structure can

E-mail: dahsu@microsoft.com, skakade@microsoft.com, pliang@cs.stanford.edu




(a) Constituency (PCFG-IE)

(b) Dependency (DEP-IE)

Figure 1: The two constituency trees and seven dependency trees over $L = 3$ words, $x_1, x_2, x_3$. (a) A constituency tree consists of a hierarchical grouping of the words with a latent state $z_v$ for each node $v$. (b) A dependency tree consists of a collection of directed edges between the words. In both cases, we have labeled each edge from $i$ to $j$ with the parameters used to generate the state of node $j$ given $i$.

be "unmixed", thereby reducing the problem to one with a fixed topology, which can be tackled using standard techniques [1]. Importantly, our unmixing technique does not require the training sentences be annotated with the tree topologies *a priori*, in contrast to recent extensions of [21] to learning PCFGs [24] and dependency trees [25, 26], which work on a fixed topology.

## 2 Notation

For a positive integer $n$, define $[n] \stackrel{\text{def}}{=} \{1, \ldots, n\}$ and $\langle n \rangle = \{e_1, \ldots, e_n\}$, where $e_i$ is the vector which is 1 in component $i$ and 0 elsewhere. For integers $a, b \in [n]$, let $a \otimes_n b = (a-1)n + b \in [n^2]$ be the integer encoding of the pair $(a, b)$. For a pair of matrices, $A, B \in \mathbb{R}^{m \times n}$, define the columnwise tensor product $A \otimes^{\mathrm{c}} B \in \mathbb{R}^{m^2 \times n}$ to be such that $(A \otimes^{\mathrm{c}} B)_{(i_1 \otimes_m i_2) j} = A_{i_1 j} B_{i_2 j}$. For a matrix $A \in \mathbb{R}^{m \times n}$, let $A^\dagger$ denote the Moore-Penrose pseudoinverse.

## 3 Parsing models

A sentence is a sequence of $L$ words, $\mathbf{x} = (x_1, \ldots, x_L)$, where each word $x_i \in \langle d \rangle$ is one of $d$ possible word types. A (generative) *parsing model* defines a joint distribution $\mathbb{P}_\theta(\mathbf{x}, z)$ over a sentence $\mathbf{x}$ and its parse tree $z$ (to be made precise later), where $\theta$ are the model parameters (a collection of multinomials). Each parse tree $z$ has a *topology* $\mathrm{Topology}(z) \in \mathrm{Topologies}$, which is both unobserved and varying across sentences. The learning problem is to recover $\theta$ given only samples of $\mathbf{x}$.

Two important classes of models of natural language syntax are constituency models, which represent a hierarchical grouping and labeling of the phrases of a sentence (e.g., Figure 1(a)), and dependency models, which represent pairwise relationships between the words of a sentence (e.g., Figure 1(b)).

### 3.1 Constituency models

A *constituency tree* $z = (V, s)$ consists of a set of nodes $V$ and a collection of hidden states $s = \{s_v\}_{v \in V}$. Each state $s_v \in \langle k \rangle$ represents one of $k$ possible syntactic categories. Each node $v$ has the form $[i : j]$ for $0 \leq i < j \leq L$ corresponding to the phrase between positions $i$ and $j$ of the sentence. These nodes form a binary tree as follows: the root node is $[0 : L] \in V$, and for each node $[i : j] \in V$ with $j - i > 1$, there exists a unique $m$ with $i < m < j$ defining the two children nodes $[i : m] \in V$ and $[m : j] \in V$. Let $\mathrm{Topology}(z)$ be an integer encoding of $V$.

**PCFG.** Perhaps the most well-known constituency parsing model is the probabilistic context-free grammar (PCFG). The parameters of a PCFG are $\theta = (\pi, B, O)$, where $\pi \in \mathbb{R}^k$ specifies the initial state distribution, $B \in \mathbb{R}^{k^2 \times k}$ specifies the binary production distributions, and $O \in \mathbb{R}^{d \times k}$ specifies the emission distributions.

A PCFG corresponds to the following generative process (see Figure 1(a) for an example): choose a topology $\mathrm{Topology}(z)$ uniformly at random; generate the state of the root node using $\pi$; recursively generate pairs of children states given their parents using $B$; and finally generate words $x_i$ given their parents using $O$. This generative process defines a joint probability over a sentence $\mathbf{x}$ and a parse tree $z$:

$$\mathbb{P}_\theta(\mathbf{x}, z) = |\,\mathrm{Topologies}\,|^{-1} \pi^\top s_{[0:L]} \prod_{[i:m],[m:j] \in V} (s_{[i:m]} \otimes_k s_{[m:j]})^\top B s_{[i:j]} \prod_{i=1}^{L} x_i^\top O s_{[i-1:i]}, \quad (1)$$

We will also consider two variants of the PCFG with additional restrictions:

**PCFG-I.** The left and right children states are generated independently—that is, we have the following factorization: $B = T_1 \otimes^{\mathrm{c}} T_2$ for some $T_1, T_2 \in \mathbb{R}^{k \times k}$.

**PCFG-IE.** The left and the right productions are independent and equal: $B = T \otimes^{\mathrm{c}} T$.

### 3.2 Dependency tree models

In contrast to constituency trees, which posit internal nodes with latent states, dependency trees connect the words directly. A *dependency tree* $z$ is a set of directed edges $(i, j)$, where $i, j \in [L]$ are distinct positions in the sentence. Let $\mathrm{Root}(z)$ denote the position of the root node of $z$. We consider only *projective* dependency trees [27]: $z$ is projective if for every path from $i$ to $j$ to $k$ in $z$, we have that $j$ and $k$ are on the same side of $i$ (that is, $j - i$ and $k - i$ have the same sign). Let $\mathrm{Topology}(z)$ be an integer encoding of $z$.

**DEP-I.** We consider the simple dependency model of [4]. The parameters of this model are $\theta = (\pi, A_{\swarrow}, A_{\searrow})$, where $\pi \in \mathbb{R}^d$ is the initial word distribution and $A_{\swarrow}, A_{\searrow} \in \mathbb{R}^{d \times d}$ are the left and right argument distributions. The generative process is as follows: choose a topology $\mathrm{Topology}(z)$ uniformly at random, generate the root word using $\pi$, and recursively generate argument words to the left to the right given the parent word using $A_{\swarrow}$ and $A_{\searrow}$, respectively. The corresponding joint probability distribution is as follows:

$$\mathbb{P}_\theta(\mathbf{x}, z) = |\,\mathrm{Topologies}\,|^{-1} \pi^\top x_{\mathrm{Root}(z)} \prod_{(i,j) \in z} x_j^\top A_{\mathrm{dir}(i,j)} x_i, \quad (2)$$

where $\mathrm{dir}(i, j) = \swarrow$ if $j < i$ and $\searrow$ if $j > i$.

We also consider the following two variants:

**DEP-IE.** The left and right argument distributions are equal: $A = A_{\swarrow} = A_{\searrow}$.

**DEP-IES.** $A = A_{\swarrow} = A_{\searrow}$ and $\pi$ is the stationary distribution of $A$ (that is, $\pi = A\pi$).

Usually a PCFG induces a topology via a state-dependent probability of choosing a binary production versus an emission. Our model is a restriction which corresponds to a state-independent probability.

# 4 Identifiability

Our goal is to estimate model parameters $\theta_0 \in \Theta$ given only access to sentences $\mathbf{x} \sim \mathbb{P}_{\theta_0}$. Specifically, suppose we have an *observation function* $\phi(\mathbf{x}) \in \mathbb{R}^m$, which is the only lens through which an algorithm can view the data. We ask a basic question: in the limit of infinite data, is it information-theoretically possible to identify $\theta_0$ from the *observed moments* $\mu(\theta_0) \stackrel{\text{def}}{=} \mathbb{E}_{\theta_0}[\phi(\mathbf{x})]$?

To be more precise, define the *equivalence class* of $\theta_0$ to be the set of parameters $\theta$ that yield the same observed moments:

$$\mathcal{S}_\Theta(\theta_0) = \{\theta \in \Theta : \mu(\theta) = \mu(\theta_0)\}. \tag{3}$$

It is impossible for an algorithm to distinguish among the elements of $\mathcal{S}_\Theta(\theta_0)$. Therefore, one might want to ensure that $|\mathcal{S}_\Theta(\theta_0)| = 1$ for all $\theta_0 \in \Theta$. However, this requirement is too strong for two reasons. First, models often have natural symmetries—e.g., the $k$ states of any PCFG can be permuted without changing $\mu(\theta)$, so $|\mathcal{S}_\Theta(\theta_0)| \geq k!$. Second, $|\mathcal{S}_\Theta(\theta_0)| = \infty$ for some pathological $\theta_0$'s—e.g., PCFGs where all states have the same emission distribution $O$ are indistinguishable regardless of the production distributions $B$. The following definition of identifiability accommodates these two exceptional cases:

**Definition 1** (Identifiability)**.** *A model family with parameter space $\Theta$ is* (globally) identifiable from *$\phi$ if there exists a measure zero set $\mathcal{E}$ such that $|\mathcal{S}_\Theta(\theta_0)|$ is finite for every $\theta_0 \in \Theta \backslash \mathcal{E}$. It is* locally identifiable from *$\phi$ if there exists a measure zero set $\mathcal{E}$ such that, for every $\theta_0 \in \Theta \backslash \mathcal{E}$, there exists an open neighborhood $N(\theta_0)$ around $\theta_0$ such that $\mathcal{S}_\Theta(\theta_0) \cap N(\theta_0) = \{\theta_0\}$.*

**Example of non-identifiability.** Consider the DEP-IE model with $L = 2$ with the full observation function $\phi(\mathbf{x}) = x_1 \otimes x_2$. The corresponding observed moments are $\mu(\theta) = 0.5A\,\mathrm{diag}(\pi) + 0.5\,\mathrm{diag}(\pi)A^\top$. Note that $A\,\mathrm{diag}(\pi)$ is an arbitrary $d \times d$ matrix whose entries sum to 1, which has $d^2 - 1$ degrees of freedom, whereas $\mu(\theta)$ is a symmetric matrix whose entries sum to 1, which has $\binom{d+1}{2} - 1$ degrees of freedom. Therefore, $\mathcal{S}_\Theta(\theta)$ has dimension $\binom{d}{2}$ and therefore the model is non-identifiable.

**Parameter counting.** It is important to compute the degrees of freedom correctly—simple parameter counting is insufficient. For example, consider the PCFG-IE model with $L = 2$. The observed moments with respect to $\phi(\mathbf{x}) = x_1 \otimes x_2$ is a $d \times d$ matrix, which places $d^2$ constraints on the $k^2 + (d-1)k$ parameters. When $d \geq 2k$, there are more constraints than parameters, but the PCFG-IE model with $L = 2$ is actually non-identifiable (as we will see later). The issue here is that the number of constraints does not reveal the fact that some of these constraints are redundant.

## 4.1 Observation functions

An observation function $\phi(\mathbf{x})$ and its associated observed moments $\mu(\theta_0) = \mathbb{E}_{\theta_0}[\phi(\mathbf{x})]$ reveals aspects of the distribution $\mathbb{P}_{\theta_0}(\mathbf{x})$. For example, $\phi(\mathbf{x}) = x_1$ would only reveal the marginal distribution of the first word, whereas $\phi(\mathbf{x}) = x_1 \otimes \cdots \otimes x_L$ reveals the entire distribution of $\mathbf{x}$. There is a tradeoff: Higher-order moments provide more information, but are harder to estimate reliably given finite data, and are also computationally more expensive. In this paper, we consider the following intermediate moments:

$$\phi_{12}(\mathbf{x}) \stackrel{\text{def}}{=} x_1 \otimes x_2 \qquad \phi_{**}(\mathbf{x}) \stackrel{\text{def}}{=} \big(x_i \otimes x_j : i, j \in [L]\big)$$

$$\phi_{123}(\mathbf{x}) \stackrel{\text{def}}{=} x_1 \otimes x_2 \otimes x_3 \qquad \phi_{***}(\mathbf{x}) \stackrel{\text{def}}{=} \big(x_i \otimes x_j \otimes x_k : i, j, k \in [L]\big)$$

$$\phi_{123\eta}(\mathbf{x}) \stackrel{\text{def}}{=} (x_1 \otimes x_2)(\eta^\top x_3) \qquad \phi_{***\eta}(\mathbf{x}) \stackrel{\text{def}}{=} \big((x_i \otimes x_j)(\eta^\top x_k) : i, j, k \in [L]\big)$$

$$\phi_{\text{all}}(\mathbf{x}) \stackrel{\text{def}}{=} x_1 \otimes \cdots \otimes x_L$$

Above, $\eta \in \mathbb{R}^d$ denotes a unit vector in $\mathbb{R}^d$ (e.g., $e_1$) which picks out a linear combination of matrix slices from a third-order $d \times d \times d$ tensor.

## 4.2 Automatically checking identifiability

One immediate goal is to determine which models in Section 3 are identifiable from which of the observed moments (Section 4.1). A powerful analytic tool that has been succesfully applied in

previous work is Kruskal's theorem [10, 11], but (i) it is does not immediately apply to models with random topologies, and (ii) only gives sufficient conditions for identifiability, and cannot be used to determine non-identifiability. Furthermore, since it is common practice to explore many different models for a given problem in rapid succession, we would like to check identifiability quickly and reliably. In this section, we develop an automatic procedure to do this.

To establish identifiability, let us examine the algebraic structure of $\mathcal{S}_\Theta(\theta_0)$ for $\theta_0 \in \Theta$, where we assume that the parameter space $\Theta$ is an open subset of $[0,1]^n$. Recall that $\mathcal{S}_\Theta(\theta_0)$ is defined by the moment constraints $\mu(\theta) = \mu(\theta_0)$. We can write these constraints as $h_{\theta_0}(\theta) = 0$, where

$$h_{\theta_0}(\theta) \stackrel{\text{def}}{=} \mu(\theta) - \mu(\theta_0)$$

is a vector of $m$ polynomials in $\theta$.

Let us now compute the number of degrees of freedom of $h_{\theta_0}$ around $\theta_0$. The key quantity is $J(\theta) \in \mathbb{R}^{m \times n}$, the Jacobian of $h_{\theta_0}$ at $\theta$ (note that the Jacobian of $h_{\theta_0}$ does not depend on $\theta_0$; it is precisely the Jacobian of $\mu$). This Jacobian criterion is well-established in algebraic geometry, and has been adopted in the statistical literature for testing model identifiability and other related properties [14–17].

Intuitively, each row of $J(\theta_0)$ corresponds to a direction of a constraint violation, and thus the row space of $J(\theta_0)$ corresponds to all directions that would take us outside the equivalence class $\mathcal{S}_\Theta(\theta_0)$. If $J(\theta_0)$ has less than rank $n$, then there is a direction orthogonal to all the rows along which we can move and still satisfy all the constraints—in other words, $|\mathcal{S}_\Theta(\theta_0)|$ is infinite, and therefore the model is non-identifiable. This intuition leads to the following algorithm:

CHECKIDENTIFIABILITY:
1. Choose a point $\tilde{\theta} \in \Theta$ uniformly at random.
2. Compute the Jacobian matrix $J(\tilde{\theta})$.
3. Return "yes" if the rank of $J(\tilde{\theta}) = n$ and "no" otherwise.

The following theorem asserts the correctness of CHECKIDENTIFIABILITY. It is largely based on techniques in [16], although we have not seen it explicitly stated in this form.

**Theorem 1** (Correctness of CHECKIDENTIFIABILITY)**.** *Assume the parameter space $\Theta$ is a non-empty open connected subset of $[0,1]^n$; and the observed moments $\mu\colon \mathbb{R}^n \to \mathbb{R}^m$, with respect to observation function $\phi$, is a polynomial map. Then with probability 1,* CHECKIDENTIFIABILITY *returns "yes" iff the model family is locally identifiable from $\phi$. Moreover, if it returns "yes", then there exists $\mathcal{E} \subset \Theta$ of measure zero such that the model family with parameter space $\Theta \setminus \mathcal{E}$ is identifiable from $\phi$.*

The proof of Theorem 1 is given in Appendix A.

### 4.3 Implementation of CHECKIDENTIFIABILITY

**Computing the Jacobian.** The rows of $J$ correspond to $\partial \mathbb{E}_\theta[\phi_j(\mathbf{x})]/\partial\theta$ and can be computed efficiently by adapting dynamic programs used in the E-step of an EM algorithm for parsing models. There are two main differences: (i) we must sum over possible values of $\mathbf{x}$ in addition to $z$, and (ii) we are not computing moments, but rather gradients thereof. Specifically, we adapt the CKY algorithm for constituency models and the algorithm of [27] for dependency models. See Appendix C.1 for more details.

**Numerical issues.** Because we implemented CHECKIDENTIFIABILITY on a finite precision machine, the results are subject to numerical precision errors. However, we verified that our numerical results are consistent with various analytically-derived identifiability results (e.g., from [11]).

While we initially defined $\theta$ to be a tuple of conditional probability matrices, we will now use its non-redundant vectorized form $\theta \in \mathbb{R}^n$.

<table>
<tr><th>Model \ Observation function</th><th>$\phi_{12}$</th><th>$\phi_{**}$</th><th>$\phi_{123e_1}$</th><th>$\phi_{123}$</th><th>$\phi_{***e_1}$</th><th>$\phi_{***}$</th></tr>
<tr><td>PCFG</td><td colspan="6">No, even from $\phi_{\text{all}}$ for $L \in \{3, 4, 5\}$</td></tr>
<tr><td>PCFG-I / PCFG-IE</td><td>No</td><td>Yes iff $L \geq 4$</td><td colspan="4">Yes iff $L \geq 3$</td></tr>
<tr><td>DEP-I</td><td>No</td><td colspan="5">Yes iff $L \geq 3$</td></tr>
<tr><td>DEP-IE / DEP-IES</td><td colspan="6">Yes iff $L \geq 3$</td></tr>
</table>

Figure 2: Local identifiability of parsing models. These findings are given by CHECKIDENTIFIABILITY have the semantics from Theorem 1. These were checked for $d \in \{2, 3, \ldots, 8\}$, $k \in \{2, \ldots, d\}$ (applies only for PCFG models), $L \in \{2, 3, \ldots, 9\}$.

### 4.4 Identifiability of constituency and dependency tree models

We checked the identifiability status of various constituency and dependency tree models using our implementation of CHECKIDENTIFIABILITY. We focus on the regime where $d \geq k$ for PCFGs; additional results for $d < k$ are given in Appendix B.

The results are reported in Figure 2. First, we found that the PCFG is not identifiable from $\phi_{\text{all}}$ (and therefore not identifiable from any $\phi$) for $L \in \{3, 4, 5\}$; we believe that the same holds for all $L$. This negative result motivates exploring restricted subclasses of PCFGs, such as PCFG-I and PCFG-IE, which factorize the binary productions. For these classes, we found that the sentence length $L$ and choice of observation function can influence identifiability: Both models are identifiable for large enough $L$ (e.g., $L \geq 3$) and with a sufficiently rich observation function (e.g., $\phi_{123\eta}$).

The dependency models, DEP-I and DEP-IE, were all found to be identifiable for $L \geq 3$ from second-order moments $\phi_{**}$. The conditions for identifiability are less stringent than their constituency counterparts (PCFG-I and PCFG-IE), which is natural since dependency models are simpler without the latent states. Note that in all identifiable models, second-order moments suffice to determine the distribution—this is good news because low-order moments are easier to estimate.

## 5 Unmixing algorithms

Having established which parsing models are identifiable, we now turn to parameter estimation for these models. We will consider algorithms based on moment matching—those that try to find a $\theta$ satisfying $\mu(\theta) = u$ for some $u$. Typically, $u$ is an empirical estimate of $\mu(\theta_0) = \mathbb{E}_{\theta_0}[\phi(\mathbf{x})]$ based on samples $\mathbf{x} \sim \mathbb{P}_{\theta_0}$.

In general, solving $\mu(\theta) = u$ corresponds to finding solutions to systems of multivariate polynomials, which is NP-hard [28]. However, $\mu(\theta)$ often has additional structure which we can exploit. For instance, for an HMM, the sliced third-order moments $\mu_{123\eta}(\theta)$ can be written as a product of parameter matrices in $\theta$, and each matrix can be recovered by decomposing the product [1].

For parsing models, the challenge is that the topology is random, so the moments is not a single product, but a mixture over products. To deal with this complication, we propose a new technique, which we call *unmixing*: We "unmix" the products from the mixtures, essentially reducing the problem to one with a fixed topology.

We will first present the general idea of unmixing (Section 5.1) and then apply it to the PCFG-IE model (Section 5.2) and the DEP-IES model (Section 5.3).

### 5.1 General case

We assume the observation function $\phi(\mathbf{x})$ consists of a collection of observation matrices $\{\phi_o(\mathbf{x})\}_{o \in \mathcal{O}}$ (e.g., for $o = (i, j)$, $\phi_o(\mathbf{x}) = x_i \otimes x_j$). Given an observation matrix $\phi_o(\mathbf{x})$ and a topology $t \in \text{Topologies}$, consider the mapping that computes the observed moment conditioned on

Note that these subclasses occupy measure zero subsets of the PCFG parameter space, which is expected given the non-identifiability of the general PCFG.

We will develop our algorithms assuming true moments ($u = \mu(\theta_0)$). The empirical moments converge to the true moments at $O_p(n^{-\frac{1}{2}})$, and matrix perturbation arguments (e.g., [1]) can be used derive sample complexity arguments for the parameter error.

that topology: $\Psi_{o,t}(\theta) = \mathbb{E}_\theta[\phi_o(\mathbf{x}) \mid \text{Topology} = t]$. As we range $o$ over $\mathcal{O}$ and $t$ over Topologies, we will enounter a finite number of such mappings. We call these mappings *compound parameters*, denoted $\{\Psi_p\}_{p\in\mathcal{P}}$.

Now write the observed moments as a weighted sum:

$$\mu_o(\theta) = \sum_{p\in\mathcal{P}} \underbrace{\mathbb{P}(\Psi_{o,\text{Topology}} = \Psi_p)}_{\stackrel{\text{def}}{=} M_{op}} \Psi_p \quad \text{for all } o \in \mathcal{O}, \tag{4}$$

where we have defined $M_{op}$ to be the probability mass over tree topologies that yield compound parameter $\Psi_p$. We let $\{M_{op}\}_{o\in\mathcal{O},p\in\mathcal{P}}$ be the *mixing matrix*. Note that (4) defines a system of equations $\mu = M\Psi$, where the variables are the compound parameters and the constraints are the observed moments. In a sense, we have replaced the original system of polynomial equations (in $\theta$) with a system of linear equations (in $\Psi$).

The key to the utility of this technique is that the number of compound parameters can be polynomial in $L$ even when the number of possible topologies is exponential in $L$. Previous analytic techniques [13] based on Kruskal's theorem [10] cannot be applied here because the possible topologies are too many and too varied.

Note that the mixing equation $\mu = M\Psi$ holds for each sentence length $L$, but many compound parameters $p$ appear in the equations of multiple $L$. Therefore, we can combine the equations across all observed sentence lengths, yielding a more constrained system than if we considered the equations of each $L$ separately.

The following proposition shows how we can recover $\theta$ by unmixing the observed moments $\mu$:

**Proposition 1** (Unmixing)**.** *Suppose that there exists an efficient base algorithm to recover $\theta$ from some subset of compound parameters $\{\Psi_p(\theta) : p \in \mathcal{P}_0\}$, and that $e_p^\top$ is in the row space of $M$ for each $p \in \mathcal{P}_0$. Then we can recover $\theta$ as follows:*

UNMIX($\mu$):
1. Compute the mixing matrix $M$ (4).
2. Retrieve the compound parameters $\Psi_p(\theta) = (M^\dagger\mu)_p$ for each $p \in \mathcal{P}_0$.
3. Call the base algorithm on $\{\Psi_p(\theta) : p \in \mathcal{P}_0\}$ to obtain $\theta$.

For all our parsing models, $M$ can be computed efficiently using dynamic programming (Appendix C.2). Note that $M$ is data-independent, so this computation can be done once in advance.

### 5.2 Application to the PCFG-IE model

As a concrete example, consider the PCFG-IE model over $L = 3$ words. Write $A = OT$. For any $\eta \in \mathbb{R}^d$, we can express the observed moments as a sum over the two possible topologies in Figure 1(a):

$$\begin{aligned}
\mu_{123\eta} &\stackrel{\text{def}}{=} \mathbb{E}[x_1 \otimes x_2(\eta^\top x_3)] = 0.5\Psi_{1;\eta} + 0.5\Psi_{2;\eta}, & \Psi_{1;\eta} &\stackrel{\text{def}}{=} A\,\text{diag}(T\,\text{diag}(\pi)A^\top\eta)A^\top,\\
\mu_{132\eta} &\stackrel{\text{def}}{=} \mathbb{E}[x_1 \otimes x_3(\eta^\top x_2)] = 0.5\Psi_{3;\eta} + 0.5\Psi_{2;\eta}, & \Psi_{2;\eta} &\stackrel{\text{def}}{=} A\,\text{diag}(\pi)T^\top\,\text{diag}(A^\top\eta)A^\top,\\
\mu_{231\eta} &\stackrel{\text{def}}{=} \mathbb{E}[x_2 \otimes x_3(\eta^\top x_1)] = 0.5\Psi_{3;\eta} + 0.5\Psi_{1;\eta}, & \Psi_{3;\eta} &\stackrel{\text{def}}{=} A\,\text{diag}(A^\top\eta)T\,\text{diag}(\pi)A^\top,
\end{aligned}$$

or compactly in matrix form:

$$\underbrace{\begin{pmatrix}\mu_{123\eta}\\ \mu_{132\eta}\\ \mu_{231\eta}\end{pmatrix}}_{\text{observed moments } \mu_\eta} = \underbrace{\begin{pmatrix}0.5I & 0.5I & 0\\ 0 & 0.5I & 0.5I\\ 0.5I & 0 & 0.5I\end{pmatrix}}_{\text{mixing matrix } M} \underbrace{\begin{pmatrix}\Psi_{1;\eta}\\ \Psi_{2;\eta}\\ \Psi_{3;\eta}\end{pmatrix}}_{\text{compound parameters } \Psi_\eta}.$$

Let us observe $\mu_\eta$ at two different values of $\eta$, say at $\eta = \mathbf{1}$ and $\eta = \tau$ for some random $\tau$. Since the mixing matrix $M$ is invertible, we can obtain the compound parameters $\Psi_{2;\mathbf{1}} = (M^{-1}\mu_1)_2$ and $\Psi_{2;\tau} = (M^{-1}\mu_\tau)_2$.

Now we will recover $\theta$ from $\Psi_{2;\mathbf{1}}$ and $\Psi_{2;\tau}$ by first extracting $A = OT$ via an eigenvalue decomposition, and then recovering $\pi$, $T$, and $O$ in turn (all up to the same unknown permutation) via elementary matrix operations.

For the first step, we will use the following tool (adapted from Algorithm A of [1]), which allow us to decompose two related matrix products:

**Lemma 1** (Spectral decomposition)**.** *Let $M_1, M_2 \in \mathbb{R}^{d\times k}$ have full column rank and $D$ be a diagonal matrix with distinct diagonal entries. Suppose we observe $X = M_1 M_2^\top$ and $Y = M_1 D M_2^\top$. Then* DECOMPOSE$(X, Y)$ *recovers $M_1$ up to a permutation and scaling of the columns.*

DECOMPOSE$(X, Y)$:
1. Find $U_1, U_2 \in \mathbb{R}^{d\times k}$ such that $\mathrm{range}(U_1) = \mathrm{range}(X)$ and $\mathrm{range}(U_2) = \mathrm{range}(X^\top)$.
2. Perform an eigenvalue decomposition of $(U_1^\top Y U_2)(U_1^\top X U_2)^{-1} = VSV^{-1}$.
3. Return $(U_1^\top)^\dagger V$.

First, run DECOMPOSE$(X = \Psi_{2;\mathbf{1}}^\top, Y = \Psi_{2;\tau}^\top)$ (Lemma 1), which corresponds to $M_1 = A$ and $M_2 = A\,\mathrm{diag}(\pi)T^\top$. This produces $A\Pi S$ for some permutation matrix $\Pi$ and diagonal scaling $S$. Since we know that the columns of $A$ sum to one, we can identify $A\Pi$.

To recover the initial distribution $\pi$ (up to permutation), take $\Psi_{2;\mathbf{1}}\mathbf{1} = A\pi$ and left-multiply by $(A\Pi)^\dagger$ to get $\Pi^{-1}\pi$. For $T$, put the entries of $\pi$ in a diagonal matrix: $\Pi^{-1}\,\mathrm{diag}(\pi)\Pi$. Take $\Psi_{2;\mathbf{1}}^\top = AT\,\mathrm{diag}(\pi)A^\top$ and multiply by $(A\Pi)^\dagger$ on the left and $((A\Pi)^\top)^\dagger(\Pi^{-1}\,\mathrm{diag}(\pi)\Pi)^{-1}$ on the right, which yields $\Pi^{-1}T\Pi$. (Note that $\Pi$ is orthogonal, so $\Pi^{-1} = \Pi^\top$.) Finally, multiply $A\Pi = OT\Pi$ and $(\Pi^{-1}T\Pi)^{-1}$, which yields $O\Pi$.

The above algorithm identifies the PCFG-IE from only length 3 sentences. To exploit sentences of different lengths, we can compute a mixing matrix $M$ which includes constraints from sentences of length $1 \le L \le L_{\max}$ up to some upper bound $L_{\max}$. For example, $L_{\max} = 10$ results in a $990 \times 2376$ mixing matrix. We can retrieve the same compound parameters ($\Psi_{2;\mathbf{1}}$ and $\Psi_{2;\tau}$) from the pseudoinverse of $M$ and as proceed as before.

### 5.3 Application to the DEP-IES model

We now turn to the DEP-IES model over $L = 3$ words. Our goal is to recover the parameters $\theta = (\pi, A)$. Let $D = \mathrm{diag}(\pi) = \mathrm{diag}(A\pi)$, where the second equality is due to stationarity of $\pi$.

$$\begin{aligned}
\mu_1 &\stackrel{\text{def}}{=} \mathbb{E}[x_1] = \pi,\\
\mu_{12} &\stackrel{\text{def}}{=} \mathbb{E}[x_1 \otimes x_2] = 7^{-1}(DA^\top + DA^\top + DA^\top A^\top + AD + ADA^\top + AD + DA^\top),\\
\mu_{13} &\stackrel{\text{def}}{=} \mathbb{E}[x_1 \otimes x_3] = 7^{-1}(DA^\top + DA^\top A^\top + DA^\top + ADA^\top + AD + AAD + AD),\\
\tilde{\mu}_{12} &\stackrel{\text{def}}{=} \tilde{\mathbb{E}}[x_1 \otimes x_2] = 2^{-1}(DA^\top + AD),
\end{aligned}$$

where $\tilde{\mathbb{E}}[\cdot]$ is taken with respect to length 2 sentences. Having recovered $\pi$ from $\mu_1$, it remains to recover $A$. By selectively combining the moments above, we can compute $AA + A = [7(\mu_{13} - \mu_{12}) + 2\tilde{\mu}_{12}]\,\mathrm{diag}(\mu_1)^{-1}$. Assuming $A$ is generic position, it is diagonalizable: $A = Q\Lambda Q^{-1}$ for some diagonal matrix $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_d)$, possibly with complex entries. Therefore, we can recover $\Lambda^2 + \Lambda = Q^{-1}(AA + A)Q$. Since $\Lambda$ is diagonal, we simply have $d$ independent quadratic equations in $\lambda_i$, which can be solved in closed form. After obtaining $\Lambda$, we retrieve $A = Q\Lambda Q^{-1}$.

## 6 Discussion

In this work, we have shed some light on the identifiability of standard generative parsing models using our numerical identifiability checker. Given the ease with which this checker can be applied, we believe it should be a useful tool for analyzing more sophisticated models [6], as well as developing new ones which are expressive yet identifiable.

There is still a large gap between showing identifiability and developing explicit algorithms. We have made some progress on closing it with our unmixing technique, which can deal with models where the tree topology varies non-trivially.

## References


[1] A. Anandkumar, D. Hsu, and S. M. Kakade. A method of moments for mixture models and hidden Markov models. In *COLT*, 2012.

[2] F. Pereira and Y. Shabes. Inside-outside reestimation from partially bracketed corpora. In *ACL*, 1992.

[3] G. Carroll and E. Charniak. Two experiments on learning probabilistic dependency grammars from corpora. In *Workshop Notes for Statistically-Based NLP Techniques, AAAI*, pages 1–13, 1992.

[4] M. A. Paskin. Grammatical bigrams. In *NIPS*, 2002.

[5] D. Klein and C. D. Manning. Conditional structure versus conditional estimation in NLP models. In *EMNLP*, 2002.

[6] D. Klein and C. D. Manning. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *ACL*, 2004.

[7] P. Liang and D. Klein. Analyzing the errors of unsupervised learning. In *HLT/ACL*, 2008.

[8] J. T. Chang. Full reconstruction of Markov models on evolutionary trees: Identifiability and consistency. *Mathematical Biosciences*, 137:51–73, 1996.

[9] A. Anandkumar, K. Chaudhuri, D. Hsu, S. M. Kakade, L. Song, and T. Zhang. Spectral methods for learning multivariate latent tree structure. In *NIPS*, 2011.

[10] J. B. Kruskal. Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and Applications*, 18:95–138, 1977.

[11] E. S. Allman, C. Matias, and J. A. Rhodes. Identifiability of parameters in latent structure models with many observed variables. *Annals of Statistics*, 37:3099–3132, 2009.

[12] E. S. Allman, S. Petrovi, J. A. Rhodes, and S. Sullivant. Identifiability of 2-tree mixtures for group-based models. *Transactions on Computational Biology and Bioinformatics*, 8:710–722, 2011.

[13] J. A. Rhodes and S. Sullivant. Identifiability of large phylogenetic mixture models. *Bulletin of Mathematical Biology*, 74(1):212–231, 2012.

[14] T. J. Rothenberg. Identification in parameteric models. *Econometrica*, 39:577–591, 1971.

[15] L. A. Goodman. Exploratory latent structure analysis using both identifiabile and unidentifiable models. *Biometrika*, 61(2):215–231, 1974.

[16] D. Bamber and J. P. H. van Santen. How many parameters can a model have and still be testable? *Journal of Mathematical Psychology*, 29:443–473, 1985.

[17] D. Geiger, D. Heckerman, H. King, and C. Meek. Stratified exponential families: graphical models and model selection. *Annals of Statistics*, 29:505–529, 2001.

[18] K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35–56, 1990.

[19] M. Johnson, T. Griffiths, and S. Goldwater. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *HLT/NAACL*, 2007.

[20] E. Mossel and S. Roch. Learning nonsingular phylogenies and hidden Markov models. *Annals of Applied Probability*, 16(2):583–614, 2006.

[21] D. Hsu, S. M. Kakade, and T. Zhang. A spectral algorithm for learning hidden Markov models. In *COLT*, 2009.

[22] S. M. Siddiqi, B. Boots, and G. J. Gordon. Reduced-rank hidden Markov models. In *AISTATS*, 2010.

[23] A. Parikh, L. Song, and E. P. Xing. A spectral algorithm for latent tree graphical models. In *ICML*, 2011.

[24] S. B. Cohen, K. Stratos, M. Collins, D. P. Foster, and L. Ungar. Spectral learning of latent-variable PCFGs. In *ACL*, 2012.

[25] F. M. Luque, A. Quattoni, B. Balle, and X. Carreras. Spectral learning for non-deterministic dependency parsing. In *EACL*, 2012.

[26] P. Dhillon, J. Rodue, M. Collins, D. P. Foster, and L. Ungar. Spectral dependency parsing with latent variables. In *EMNLP-CoNLL*, 2012.

[27] J. Eisner. Three new probabilistic models for dependency parsing: An exploration. In *COLING*, 1996.

[28] S. Sahni. Computationally related problems. *SIAM Journal on Computing*, 3:262–279, 1974.

[29] J. Eisner. Bilexical grammars and their cubic-time parsing algorithms. In *Advances in Probabilistic and Other Parsing Technologies*, pages 29–62, 2000.