Fast Stochastic Quadrature for Approximate Maximum-Likelihood Estimation

Nico Piatkowski
Department of Computer Science
AI Group, TU Dortmund
44221 Dortmund, Germany

Katharina Morik
Department of Computer Science
AI Group, TU Dortmund
44221 Dortmund, Germany

Abstract

Recent stochastic quadrature techniques for undirected graphical models rely on near-minimax degree-k polynomial approximations to the model's potential function for inferring the partition function. While providing desirable statistical guarantees, typical constructions of such approximations are themselves not amenable to efficient inference. Here, we develop a class of Monte Carlo sampling algorithms for efficiently approximating the value of the partition function, as well as the associated pseudo-marginals. More precisely, for pairwise models with n vertices and m edges, the complexity can be reduced from $\mathcal{O}(d^k)$ to $\mathcal{O}(k^4 + kn + m)$, where $d \geq 4m$ is the parameter dimension. We also consider the uses of stochastic quadrature for the problem of maximum-likelihood (ML) parameter estimation. For completely observed data, our analysis gives rise to a probabilistic bound on the log-likelihood of the model. Maximizing this bound yields an approximate ML estimate which, in analogy to the moment-matching of exact ML estimation, can be interpreted in terms of pseudo-moment-matching. We present experimental results illustrating the behavior of this approximate ML estimator.

1 INTRODUCTION

The major source of complexity in the course of parameter estimation for undirected graphical models is the #P-hardness of the partition function (Valiant, 1979; Bulatov and Grohe, 2004). This quantity plays a fundamental role in various contexts, including approximate inference, maximum-likelihood (ML) parameter estimation, and large deviations analysis—to mention just a few. For a general undirected model, exact computation of this partition function is intractable; therefore, developing approximations and bounds is an important problem. The dominant approaches in this area are Markov Chain Monte Carlo (MCMC) sampling approaches (Andrieu et al., 2003) and variational inference (Wainwright and Jordan, 2008). While both directions work very well in practice, theoretical quality guarantees cannot be asserted. Some of the existing techniques indeed deliver error bounds, but the error cannot be quantified without making assumptions that go beyond the ordinary variational principle or sampling procedures.

Our recent stochastic quadrature technique (Piatkowski and Morik, 2016) for undirected graphical models relies on a near-minimax degree-k polynomial approximation to the model's potential function for inferring the partition function. While providing desirable statistical guarantees, typical constructions of such approximations are themselves not amenable to efficient inference. Here, we develop a class of Monte Carlo sampling algorithms for efficiently approximating the value of the partition function, as well as the associated pseudo-marginals.

Our contributions can be summarized as follows:

- We provide a Monte Carlo sampling procedure that reduces the complexity of the stochastic quadrature inference method from $\mathcal{O}(d^k)$ to $\mathcal{O}(k^4 + kn + m)$ when certain combinatorial quantities are precomputed. An empirical evaluation shows that our new method is several orders of magnitude faster than the existing approach.
- We provide the first stochastic quadrature based algorithm for marginal inference, and thus, for approximate maximum-likelihood parameter estimation. Experimental results show that approximate log-likelihood and predicted marginal probabilities are close to their exact counterparts.
- We explain how the stochastic quadrature can be ap-

plied to models with continuous random variables.

- Our results are derived from first-principles and work with any discrete and some continuous exponential family members.

2 NOTATION AND BACKGROUND

Let us summarize the notation and background necessary for subsequent development. The set that contains the first n strictly positive integers is denoted by $[n] = \{1, 2, \dots, n\}$.

Graphical Models: An undirected graph $G = (V, E)$ consists of $n = |V|$ vertices, connected via edges $(v, w) \in E$. For each vertex $v \in V$, we denote the set of adjacent vertices by $\mathcal{N}(v)$. A clique C is a fully-connected subset of vertices, i.e., $\forall v, w \in C : (v, w) \in E$. The set of all cliques of G is denoted by $\mathcal{C}$. Here, any undirected graph represents the conditional independence structure of an undirected graphical model or Markov random field (MRF). To this end, we identify each vertex $v \in V$ with a random variable $\boldsymbol{X}_v$ taking values in the state space $\mathcal{X}_v$. The random vector $\boldsymbol{X} = (\boldsymbol{X}_v : v \in V)$ contains the joint state of all vertices in some arbitrary but fixed order, taking values $\boldsymbol{x}$ in the Cartesian product space $\mathcal{X} = \bigotimes_{v \in V} \boldsymbol{X}_v$. Moreover, we allow to access these quantities for any proper subset of variables $S \subset V$, i.e., $\boldsymbol{X}_S = (\boldsymbol{X}_v : v \in S)$, $\boldsymbol{x}_S$, and $\mathcal{X}_S$, respectively.

Exponential Families: Markov random fields with strictly positive density can be represented via exponential family members, which have been studied extensively during the last century, e.g. (Pitman, 1936; Hammersley and Clifford, 1971; Besag, 1975; Wainwright and Jordan, 2008). The probability density of $\boldsymbol{X}$ w.r.t. some probability measure $\mathbb{P}_{\boldsymbol{\theta}}$ can hence be written as

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \exp(\langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle - A(\boldsymbol{\theta})) \tag{1}$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ is the d-dimensional parameter vector, and $\phi(\boldsymbol{x})$ is a statistic, sufficient for $\boldsymbol{\theta}$—it captures all properties of $\boldsymbol{X}$ which are relevant for inferring $\boldsymbol{\theta}$, i.e., $\mathbb{P}(\boldsymbol{\theta} \in \Omega \mid \phi(\boldsymbol{x})) = \mathbb{P}(\boldsymbol{\theta} \in \Omega \mid \boldsymbol{x})$ for all $\Omega \subseteq \mathbb{R}^d$. Normalization of $p_{\boldsymbol{\theta}}$ is guaranteed via $A(\boldsymbol{\theta}) = \ln Z(\boldsymbol{\theta}) = \ln \int_{\mathcal{X}} \exp(\langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle) \,\mathrm{d}\, \nu(\boldsymbol{x})$ w.r.t. some base measure ν. Different base measures allow for either discrete of continuous random variables $\boldsymbol{X}$ (Wainwright and Jordan, 2008). When $\mathcal{X}$ is discrete, the statistic $\phi(\boldsymbol{x})$, given via $\phi(\boldsymbol{x})_{C=\boldsymbol{y}} = \prod_{v \in V} \delta_{\{\boldsymbol{x}_C = \boldsymbol{y}\}}$ with $\boldsymbol{y} \in \mathcal{X}_C$, is always sufficient for $\boldsymbol{\theta}$. Here, $\delta_{\{\text{expression}\}}$ is the indicator function that evaluates to 1 if and only if the expression is true, and 0 otherwise. Note that each dimension of ϕ, say $\phi(\boldsymbol{x})_i$, corresponds to $\phi(\boldsymbol{x})_{C=\boldsymbol{y}}$. That is, we have an equivalence between indices $i \in [d]$ and pairs of clique $C \in \mathcal{C}$ and clique-state $\boldsymbol{y} \in \mathcal{X}_C$, in short: $i \equiv (C, \boldsymbol{y})$. Thus, we have $d = \sum_{C \in F} |\mathcal{X}_C|$ dimensions in total. This kind of sufficient statistic is also called *overcomplete*. In various applications (Ising, 1925; Sutton and McCallum, 2011), the dimensionality of the model is reduced by assuming a pairwise factorization. Only cliques of size ≤ 2 are considered in this case, which implies $d \leq \sum_{v \in V} |\mathcal{X}_v| + \sum_{\{v,w\} \in E} |\mathcal{X}_v||\mathcal{X}_w|$.

Quadrature: Whenever integrating a function f is not tractable, one may resort to numerical methods in order to approximate the definite integral $\mathcal{I}[f] = \int_l^u f(z) \,\mathrm{d}\, z$. A different way of performing numeric integration are general quadrature rules. There, the basic idea is to replace the integrand f by an approximation $h \approx f$, that admits tractable integration. It turns out, that choosing $h = h_k$ to be a degree-k Chebyshev polynomial approximation of f, delivers highly accurate results, due to the equioscillation property implied by near-minimax optimality. The general quadrature procedure can be summarized as

$$\int_l^u f(x) \mathrm{d}x \approx \int_l^u h_k(x) \mathrm{d}x = \sum_{i=0}^{k} w_i f(x_i) = \mathcal{I}_k[f]$$

where w_i are certain coefficients and x_i are certain abscissae in $[l, u]$ (all to be determined) (Mason and Handscomb, 2002).

In general, any polynomial approximation works. It can be shown that an optimal (w.r.t. the l_p-norm) degree-k polynomial approximation h_k of any function f on a fixed interval $[l; u]$ always exists and is uniquely characterized by the equioscillation property (Mason and Handscomb, 2002). That is, the error function $E(z) = f(z) - h_k(z)$ oscillates on $[l; u]$ and has exactly $k + 2$ extrema (Jr., 1966).

Due to their oscillation property, Chebyshev polynomials are an important building block in the construction and analysis of minimax optimal approximations. Chebyshev polynomials are specified by the fundamental recurrence relation

$$T_0(z) = 1, T_1(z) = z, T_k(z) = 2zT_{k-1}(z) - T_{k-2}(z)\ .$$

They have an extraordinary large variety of convenient properties, like rapidly decreasing and individually converging coefficients (Gautschi, 1985), which make them ubiquitous in virtually any field of numerical analysis. An excellent discussion of Chebyshev polynomials in general, can be found in (Mason and Handscomb, 2002). Depending on the choice of interpolation points and different kinds of orthogonality properties, Chebyshev polynomial based quadrature rules are termed Gauss-Chebyshev quadrature, Fejér quadrature or Clenshaw-Curtis quadrature (Clenshaw and Curtis, 1960).

Putting all together, the (deterministic) quadrature approximation to the partition function $Z(\boldsymbol{\theta})$ is

$$\begin{aligned} Z(\boldsymbol{\theta}) &= \int_{\mathcal{X}} \exp(\langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle) \,\mathrm{d}\,\nu(\boldsymbol{x}) \\ &\approx \int_{\mathcal{X}} \hat{\exp}^k_{\boldsymbol{\zeta}}(\langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle) \,\mathrm{d}\,\nu(\boldsymbol{x}) = \hat{Z}^k_{\boldsymbol{\zeta}}(\boldsymbol{\theta}) \ , \end{aligned} \tag{2}$$

where $\hat{\exp}^k_{\boldsymbol{\zeta}}$ is a degree-k Chebyshev approximation to the exponential function on the interval $[l;u]$, and $\boldsymbol{\zeta}$ are the corresponding coefficients. Chebyshev approximations yield the best uniform approximation on $[l;u]$. $\boldsymbol{\zeta}$ can be approximated numerically via discrete cosine transformation or the Remez exchange algorithm (Fraser, 1965). It can be shown that the approximation error ε is bounded and exponentially small in $k \ln k$ (Xiang et al., 2010) when $l \leq \min_{\boldsymbol{x}} \langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle$ and $u \geq \max_{\boldsymbol{x}} \langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle$.

3 FAST STOCHASTIC QUADRATURE

In this section, we present the stochastic Clenshaw-Curtis quadrature that yields an $(1 \pm \varepsilon)$-approximation to the partition function (Piatkowski and Morik, 2016). We then develop a class of Monte Carlo algorithms designed to perform the actual estimation of $A(\boldsymbol{\theta})$ efficiently.

k-Integrable Statistics: Let ϕ denote the d-dimensional statistic of the exponential family representation (1) of some undirected graphical model for $\boldsymbol{X}$. Of particular interest are statistics which are *k-integrable*—that is, the function

$$\chi^k_\phi(\boldsymbol{j}) = \int_{\mathcal{X}} \prod_{i=1}^{k} \phi(\boldsymbol{x})_{\boldsymbol{j}_i} \,\mathrm{d}\,\nu(\boldsymbol{x}) \tag{3}$$

admits a polynomial time computable closed-form expression for all index tuples $\boldsymbol{j} \in [d]^k$. It can be shown (Piatkowski and Morik, 2016) that overcomplete sufficient statistics of discrete Markov random fields are always k-integrable. In this case,

$$\chi^k_\phi(\boldsymbol{j}) = \begin{cases} \frac{|\mathcal{X}|}{|\mathcal{X}_{\bigcup_{i=1}^k C(\boldsymbol{j}_i)}|} & , \boldsymbol{j} \text{ is realizable} \\ 0 & , \text{ otherwise .} \end{cases} \tag{4}$$

Here, $C(\boldsymbol{j}_i)$ denotes the clique that corresponds to the i-th entry of $\boldsymbol{j}$, i.e., $\boldsymbol{j}_i \equiv (C(\boldsymbol{j}_i), \boldsymbol{y}(\boldsymbol{j}_i))$. An index tuple $\boldsymbol{j}$ is not realizable, if two (or more) indices imply that the same vertex is in two (or more) different states at the same time.

Let us extend this result by showing that various sufficient statistics for continuous random variables are as well k-integrable.

Lemma 1 (Continuous k-integrability) *Let $\boldsymbol{X}$ be an n-dimensional continuous random vector. Any statistic $\phi(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}^d$ whose coordinate-wise statistics $\phi(\boldsymbol{x})_i$ are (linear transformations of)*

$$\phi(\boldsymbol{x})_i = \boldsymbol{x}_j^c, \ \textit{or} \ \ \phi(\boldsymbol{x})_i = \tfrac{1}{\boldsymbol{x}_j^c}, \ \textit{or} \ \ \phi(\boldsymbol{x})_i = \ln(\boldsymbol{x}_j)^c,$$

with $c \in \mathbb{N}, j \in [n]$, is k-integrable for all $k \in \mathbb{N}$.

Details on the integration of elementary functions can be found in (Bronstein, 1990). In fact, the sufficient statistics of the Gaussian, Poisson, exponential, beta, Dirichlet, Pareto, Weibull with known shape, chi-squared, log-normal, beta, and gamma distributions, restricted to the interval $(0;u]$, consist only of terms of the form $1/x^c$, x^c, and $\ln(x)^c$ which implies their k-integrability. E.g., assume that $\phi(x)_1 = x$ and $\phi(x)_2 = \ln_2(x)^2$ with $x \in (0;u]$, then, for $\boldsymbol{j} = (1,2,1)$, we have $\chi^k_\phi(\boldsymbol{j}) = u^3(9\ln(u)^2 - 6\ln(u) + 2)/(27\ln(2)^2)$, which is indeed a polynomial time computable closed-form.

One may extend Lemma 1 to include statistics of the form $|x - m|^c$, which appear in the density of the Laplace distribution. Closed-form expressions exist, but we excluded them here due to the notational clutter that arises when products of such functions are integrated.

Stochastic Clenshaw-Curtis Quadrature (SCCQ): The major ingredient of the stochastic quadrature is a specific probability mass function (pmf) over index tuples $\boldsymbol{j} \in [d]^i$ of length $0 \leq i \leq k$. For ease of notation, we assume that indices of $(k+1)$-dimensional objects start at 0. Suppose $\phi : \mathcal{X} \to \mathbb{R}^d_+$ is a non-negative, k-integrable statistic. Let $\|\chi^i_\phi\|_1$ denote the 1-norm of the function χ^i_ϕ. Moreover, for any $(k+1)$-dimensional real-valued vector $\boldsymbol{\zeta}$, let $|\boldsymbol{\zeta}|$ denote the element-wise absolute value of $\boldsymbol{\zeta}$, i.e., $\|\chi^i_\phi\|_1 = \sum_{\boldsymbol{j} \in [d]^i} |\chi^i_\phi(\boldsymbol{j})|$.

Let further $(\boldsymbol{J}, I)$ be the discrete random variable with state space $[d]^k \otimes ([k] \cup \{0\})$ and pmf $\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j}, I = i) = \mathbb{P}_\phi(\boldsymbol{J} = \boldsymbol{j} \mid I = i)\mathbb{P}_{\boldsymbol{\zeta},\phi}(I = i)$ with

$$\mathbb{P}_{\boldsymbol{\zeta},\phi}(I = i) = \frac{|\boldsymbol{\zeta}_i| \|\chi^i_\phi\|_1}{\sum_{j=0}^k |\boldsymbol{\zeta}_j| \|\chi^j_\phi\|_1} \tag{5}$$

and

$$\mathbb{P}_\phi(\boldsymbol{J} = \boldsymbol{j} \mid I = i) = \frac{\chi^i_\phi(\boldsymbol{j})}{\|\chi^i_\phi\|_1} \ . \tag{6}$$

We call $\mathbb{P}_{\boldsymbol{\zeta},\phi}$ the *tuple mass* with parameter $(\boldsymbol{\zeta}, \phi)$.

Now, we define the random variable which constitutes the core of SCCQ.

Algorithm 1: Stochastic Clenshaw-Curtis Quadrature

```
input $\boldsymbol{\theta}$, $\boldsymbol{\zeta}$, $k$, $N$
output Approximate partition function $\hat{Z}^{N,k}_{\boldsymbol{\zeta}}(\boldsymbol{\theta})$
1: $S \leftarrow 0$
2: for $l = 1$ to $N$ do
3:     $(\boldsymbol{j}, i) \sim \mathbb{P}_{\boldsymbol{\zeta},\phi}$
4:     $S \leftarrow S + \hat{Z}^k_{\boldsymbol{j},i}(\boldsymbol{\theta})$
5: end for
6: $\hat{Z}^{N,k}_{\boldsymbol{\zeta}}(\boldsymbol{\theta}) \leftarrow \frac{1}{N} S$
```

Definition 1 (1-SCCQ) *Let $k \in \mathbb{N}$, $\boldsymbol{\theta} \in \mathbb{R}^d$, and let $\boldsymbol{J}$ be a random index tuple of random length I, both having joint tuple mass $\mathbb{P}_{\boldsymbol{\zeta},\phi}$. The random variable*

$$\hat{Z}^k_{\boldsymbol{J},I}(\boldsymbol{\theta}) = \tau \,\mathrm{sgn}(\boldsymbol{\zeta}_I) \prod_{r=0}^{I} \boldsymbol{\theta}_{\boldsymbol{J}_r}$$

with $\tau = \sum_{j=0}^{k} |\boldsymbol{\zeta}_j| \|\chi^j_\phi\|_1$ is called 1-SCCQ.

Surprisingly, this random variable is closely related to the quadrature approximation to $Z(\boldsymbol{\theta})$ from equation (2).

Theorem 1 (Unbiasedness of SCCQ) *Let $\boldsymbol{\zeta}$ be the coefficient vector $\boldsymbol{\zeta}$ of a degree-k polynomial approximation to* exp *over some arbitrary but fixed interval $[l; u]$, and let ϕ be a non-negative and k-integrable statistic. The random variable $\hat{Z}^k_{\boldsymbol{J},I}(\boldsymbol{\theta})$ is an unbiased estimator for $\hat{Z}^k_{\boldsymbol{\zeta}}(\boldsymbol{\theta}) = \int_{\boldsymbol{x}} \hat{\exp}^k_{\boldsymbol{\zeta}}(\langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle) \,\mathrm{d}\,\nu(\boldsymbol{x})$.*

Proof. Using equations (3), (5), and (6), as well as Definition 1, it follows that

$$\begin{aligned}
&\mathbb{E}\left[\hat{Z}^k_{\boldsymbol{J},I}(\boldsymbol{\theta})\right] \\
&= \sum_{i=0}^{k} \sum_{\boldsymbol{j} \in [d]^k} \mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j}, I = i) \tau \,\mathrm{sgn}(\boldsymbol{\zeta}_i) \prod_{r=0}^{i} \boldsymbol{\theta}_{\boldsymbol{j}_r} \\
&= \sum_{i=0}^{k} \boldsymbol{\zeta}_i \sum_{\boldsymbol{j} \in [d]^i} \prod_{r=0}^{i} \boldsymbol{\theta}_{\boldsymbol{j}_r} \int_{\mathcal{X}} \prod_{r=0}^{i} \phi(\boldsymbol{x})_{\boldsymbol{j}_r} \,\mathrm{d}\,\nu(\boldsymbol{x}) \\
&= \int_{\mathcal{X}} \sum_{i=0}^{k} \boldsymbol{\zeta}_i \langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle^i \,\mathrm{d}\,\nu(\boldsymbol{x}) \;=\; \hat{Z}^k_{\boldsymbol{\zeta}}(\boldsymbol{\theta}) \;,
\end{aligned}$$

where the last line stems from the fact that

$$\langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle^i = \sum_{\boldsymbol{j}_1=1}^{d} \sum_{\boldsymbol{j}_2=1}^{d} \cdots \sum_{\boldsymbol{j}_i=1}^{d} \prod_{l=0}^{i} \boldsymbol{\theta}_{\boldsymbol{j}_l} \prod_{r=0}^{i} \phi(\boldsymbol{x})_{\boldsymbol{j}_r} \;.$$

■

Based on the theorem, we devise a Monte Carlo procedure, called *N-SCCQ* or simply *SCCQ*, shown in Algorithm 1. By combining the error ε that is introduced by the polynomial approximation with the error that is introduced by the sampling procedure, an overall error bound can be derived[1].

Theorem 2 (SCCQ Error Bound) *Let $\boldsymbol{\zeta}$ be the coefficient vector of a degree-k Chebyshev approximation to* exp *on $[l; u] = [-\|\boldsymbol{\theta}\|_1; +\|\boldsymbol{\theta}\|_1]$ with worst-case error ε. Let $\hat{Z}^{N,k}_{\boldsymbol{\zeta}}(\boldsymbol{\theta})$ be the output of Algorithm 1. Furthermore, let $\delta \in (0, 1]$, $\epsilon > 0$, $N = (\ln 2/\delta)\tau^2 2\|\boldsymbol{\theta}\|_\infty^{2k'} \varepsilon^{-2} |\mathcal{X}|^{-2}$, with $(k-1)\,k! \geq 8 \exp(2\|\boldsymbol{\theta}\|_1)/(\pi\epsilon)$, and $k' = 1$ if $\|\boldsymbol{\theta}\|_\infty < 1$ or otherwise $k' = k$. Then,*

$$\mathbb{P}[|\hat{Z}^{N,k}_{\boldsymbol{\zeta}}(\boldsymbol{\theta}) - Z(\boldsymbol{\theta})| < \epsilon Z(\boldsymbol{\theta})] \geq 1 - \delta \;.$$

3.1 COMPUTATIONAL COMPLEXITY

While SCCQ enjoys a bounded error and an apparently simple algorithm, the actual sampling of index tuples from $\mathbb{P}_{\boldsymbol{\zeta},\phi}$ (line 3 in Algorithm 1) and the computation of $\hat{Z}^k_{\boldsymbol{J},I}(\boldsymbol{\theta})$ (line 4 in Algorithm 1) turn out to be computationally hard. Computing $\hat{Z}^k_{\boldsymbol{J},I}(\boldsymbol{\theta})$ requires the $\|\chi^i_\phi\|_1$ values. In (Piatkowski and Morik, 2016), the authors assume that the values of $\|\chi^i_\phi\|_1$ for all $0 \leq i \leq k$ are precomputed, which requires $\mathcal{O}(d^k)$ additions. While rather small polynomial degrees ($k \approx 8$) suffice to achieve reasonable results, the overcomplete dimension d of a 10×10 binary Ising grid model is 720. Hence, at least $d^k = 720^8 > 2^{75}$ additions are required to compute $\|\chi^i_\phi\|_1$. In our initial work on SCCQ (Piatkowski and Morik, 2016), rejection sampling was used to generate the samples from $\mathbb{P}_{\boldsymbol{\zeta},\phi}$ with a uniform proposal $\mathbb{Q}$ on $[d]^k \otimes ([k] \cup \{0\})$. Since the ratio $\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{j}, i)/\mathbb{Q}(\boldsymbol{j}, i) = (k+1)d^k \mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{j}, i)$ is large, one shall expect that many samples will be rejected.

3.2 NORMALIZING THE TUPLE MASS

To alleviate the complexity issues of SCCQ, we now present a closed-form expression for $\|\chi^i_\phi\|_1$. Our result relies on the closed-form of k-integrable statistics, which is given by equation (4) for discrete state space models. We restrict ourselves to discrete models, since a general closed-form that covers all continuous state space models does not exist. However, the general methodology can be transferred to the continuous case.

The forthcoming results make heavy use of equivalence classes of index tuples $\boldsymbol{j} \in [d]^i$ and their cardinalities. In

[1] An earlier result can be found in (Piatkowski and Morik, 2016). There, the bound on the error of the polynomial approximation uses an inequality which is originally designed for complex-valued functions. Here, we apply a recent inequality due to Trefethen (Trefethen, 2008). Both results are qualitatively equivalent w.r.t. N and k. Nevertheless, the new proof is simplified.

this context, it is important to recall that any index $j \in [d]$ corresponds to a pair of clique and state: $i \equiv (C, \boldsymbol{y})$. Consequently, a tuple of indices corresponds to a tuple of cliques and states.

Definition 2 (Sub-Alphabets) *Let $\mathcal{A}$ be some set of objects or symbols—$\mathcal{A}$ is an* alphabet*—and let $\mathcal{P}(\mathcal{A})$ be its power set. The set $\mathcal{P}(\mathcal{A}, n) \subseteq \mathcal{P}(\mathcal{A})$ contains all subsets of $\mathcal{A}$ with at most n elements, i.e.,*

$$\mathcal{P}(\mathcal{A}, n) = \{S \in \mathcal{P}(\mathcal{A}) \mid |S| \leq n\} .$$

The size of $\mathcal{P}(\mathcal{A}, n)$ is thus

$$|\mathcal{P}(\mathcal{A}, n)| = \sum_{i=1}^{n} \binom{|\mathcal{A}|}{i} .$$

Definition 3 (Tuple Classes) *Let $i \in \mathbb{N}$, and denote the clique tuple that corresponds to an index tuple $\boldsymbol{j} \in [d]^i$ by $\boldsymbol{C}(\boldsymbol{j}) \in \mathcal{C}^i$. Two or more index tuples $\boldsymbol{j}, \boldsymbol{j}'$ may correspond to the same clique tuple, i.e., $\boldsymbol{C}(\boldsymbol{j}) = \boldsymbol{C}(\boldsymbol{j}')$. The equivalence class of all index tuples that correspond to the same clique tuple is denoted by*

$$\llbracket \boldsymbol{j} \rrbracket = \{\boldsymbol{j}' \in [d]^i \mid \boldsymbol{C}(\boldsymbol{j}) = \boldsymbol{C}(\boldsymbol{j}')\} .$$

Similarly, two or more clique tuples $\boldsymbol{C}, \boldsymbol{C}'$ may correspond to the same set of cliques. The equivalence class of clique tuples that correspond to the same set of cliques is denoted by

$$\llbracket \boldsymbol{C} \rrbracket = \left\{ \boldsymbol{C}' \in \mathcal{C}^i \;\middle|\; \bigcup_{c \in \boldsymbol{C}} \{c\} = \bigcup_{c' \in \boldsymbol{C}'} \{c'\} \right\} .$$

Combining both, the equivalence class of all index tuples, whose corresponding clique tuples are in the same equivalence class, is denoted by

$$\llbracket \boldsymbol{j} \rrbracket^* = \{\boldsymbol{j}' \in [d]^i \mid \boldsymbol{C}(\boldsymbol{j}') \in \llbracket \boldsymbol{C}(\boldsymbol{j}) \rrbracket\} .$$

Note that all members of a specific clique tuple equivalence class $\llbracket \boldsymbol{C} \rrbracket$ are determined by a unique set of cliques which come from the alphabet $\mathcal{C}$. Hence, we identify each class $\llbracket \boldsymbol{C} \rrbracket$ with this unique set of cliques and treat each $\llbracket \boldsymbol{C} \rrbracket$ as an element of $\mathcal{P}(\mathcal{C}, i)$. Moreover, there are $|\mathcal{P}(\mathcal{C}, i)|$ distinct size-i clique tuple equivalence classes.

In the remainder, it will be important to know how large these equivalence classes are.

Lemma 2 (Counting Tuples) *Let $i, j \in \mathbb{N}$, $\boldsymbol{j} \in [d]^j$, $\boldsymbol{C} \in \mathcal{C}^i$, and consider the equivalence classes defined above. Then,*

$$|\llbracket \boldsymbol{j} \rrbracket| = \prod_{l=1}^{i} |\mathcal{X}_{\boldsymbol{C}(\boldsymbol{j})_l}|, \quad |\llbracket \boldsymbol{C} \rrbracket| = h(\boldsymbol{C})! \left\{ i \atop h(\boldsymbol{C}) \right\},$$
$$|\llbracket \boldsymbol{j} \rrbracket^*| = |\llbracket \boldsymbol{C}(\boldsymbol{j}) \rrbracket| |\llbracket \boldsymbol{j} \rrbracket|$$

where $h(\boldsymbol{C})$ is the number of distinct cliques which appear in the tuple $\boldsymbol{C}$, $n!$ is the factorial, and $\{n\ k\}^\top$ is the Stirling number of second kind.

It will be helpful to define equivalence classes of index tuples w.r.t. some k-integrable statistics. Here, equivalence w.r.t. χ^i_ϕ is established by the value that each member of an equivalence class contributes to $\|\chi^i_\phi\|_1$.

Definition 4 (Tuple Classes and k-integrability) *Let ϕ be a k-integrable statistic, $i \in \mathbb{N}$, and $\boldsymbol{j} \in [d]^i$. The equivalence class of all index tuples which correspond to the same clique tuple and have non-zero χ^i_ϕ-value is denoted by*

$$\llbracket \boldsymbol{j} \rrbracket_\phi = \{\boldsymbol{j}' \in [d]^i \mid \boldsymbol{j}' \in \llbracket \boldsymbol{j} \rrbracket \wedge \chi^i_\phi(\boldsymbol{j}') \neq 0\} .$$

The corresponding extension to equivalence classes of clique tuples, is denoted by

$$\llbracket \boldsymbol{j} \rrbracket^*_\phi = \{\boldsymbol{j}' \in [d]^i \mid \boldsymbol{j}' \in \llbracket \boldsymbol{j} \rrbracket^* \wedge \chi^i_\phi(\boldsymbol{j}') \neq 0\} .$$

Up to now, we made no use of the fact that our state space is discrete. The above definitions and lemmas are valid for any exponential family model with positive k-integrable statistic. However, the proof of the next lemma makes use of equation (4). In order to extend our results to continuous random variables, one has to invoke Lemma 1 to derive a closed-form for χ^i_ϕ.

Lemma 3 (Counting Realizable Tuples) *Suppose ϕ is the binary, overcomplete sufficient statistic of discrete MRFs. Then,*

$$|\llbracket \boldsymbol{j} \rrbracket_\phi| = |\mathcal{X}_{\boldsymbol{C}(\boldsymbol{j})}|, \quad \textit{and} \quad |\llbracket \boldsymbol{j} \rrbracket^*_\phi| = |\llbracket \boldsymbol{C}(\boldsymbol{j}) \rrbracket| |\llbracket \boldsymbol{j} \rrbracket_\phi| ,$$

with $\mathcal{X}_{\boldsymbol{C}(\boldsymbol{j})} = \mathcal{X}_{\cup_{l=1}^{i} \boldsymbol{C}(\boldsymbol{j})_l}$ and $\boldsymbol{C}(\boldsymbol{j}) \in \mathcal{C}^i$.

Now, we have gathered all terms and definitions to devise an improved procedure for the normalization of the index tuple mass.

Theorem 3 (Tuple Mass Normalization) *Suppose ϕ is the binary, overcomplete sufficient statistic of discrete MRFs. The conditional index tuple mass $\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j} \mid I = i)$ (equation (6)) can be normalized in $\mathcal{O}(1)$ steps. More precisely,*

$$\|\chi^i_\phi\|_1 = |\mathcal{X}| \sum_{l=0}^{i} \left\{ i \atop l \right\} \binom{|\mathcal{C}|}{l} l! = |\mathcal{X}||\mathcal{C}|^i . \tag{7}$$

The complexity $\mathcal{O}(1)$ provided in the theorem is an overwhelming improvement, compared to the naive summation, i.e., $\mathcal{O}(d^k)$. Since we need the normalization $\|\chi^i_\phi\|_1$ for all $1 \leq i \leq k$ tuple lengths, $\hat{Z}^k_{\boldsymbol{J},I}(\boldsymbol{\theta})$ can be computed in $\mathcal{O}(k)$ steps when a pair $(\boldsymbol{j}, i)$ is given.

Algorithm 2: Fast Index Tuple Sampler

```
input  Tuple length $i$
output  Sample $\boldsymbol{j} \mid I = i$ from $\mathbb{P}_{\boldsymbol{\zeta},\phi}$
$l \sim \mathbb{P}(l \mid i)$ // See Theorem 4
$a \sim \mathbb{U}(1; \textbf{binom}(|\mathcal{C}|, l))$
$b \sim \mathbb{U}(1; \textbf{Stirling2}(i, l) \times \textbf{factorial}(l))$
$[\![\boldsymbol{C}]\!] \leftarrow$ compute $a$-th $l$-combination of
    $\{1, 2, \ldots, |\mathcal{C}|\}$ // via (Buckles and Lybanon, 1977)
$\boldsymbol{C} \leftarrow$ compute $b$-th composition of $\{1, 2, \ldots, i\}$
    with $l$ subsets // via (Ehrlich, 1973)
$S \leftarrow \bigcup_{h=1}^{i} \boldsymbol{C}_h$
$c \sim \mathbb{U}(1; \prod_{v \in S} |\mathcal{X}_v|)$
$\boldsymbol{y} \leftarrow$ compute $c$-th joint state of all vertices in $S$
return $\boldsymbol{j}$ that corresponds to $\boldsymbol{C} = \boldsymbol{y}$
```

3.3 FAST INDEX TUPLE SAMPLER

Based on the insights that we gained so far, we derive a direct sampling scheme for index tuples that circumvents any rejection step (Algorithm 2).

Given our results from the last subsection, drawing a random tuple length from $\mathbb{P}_{\boldsymbol{\zeta}}(I)$ can be done efficiently—it is a draw from a categorical distribution with state space size k (which is rather small). Sampling from the tuple mass $\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j} \mid I = i)$ can be more involved, which motivates the derivation of a specialized sampling scheme. Our algorithm is motivated by inversion sampling: For any fixed i, inversion sampling of j then consists of drawing a uniform random number u in $(0; 1)$, and finding the smallest $L \in \mathbb{N}$, such that the sum of the first L tuple masses exceeds u. The L-th tuple is then the sample. The worst-case runtime complexity is then $\mathcal{O}(d^k)$ per sample, which can be prohibitively expensive whenever the dimension d of the model is large. Based on the equivalence classes that we exploited already for the normalization of the tuple mass, we derive a factorization of $\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j} \mid I = i)$ which in turn implies an efficient stagewise sampling procedure.

To this end, let $\prec$ be an any arbitrary but fixed strict total ordering on the equivalence classes of clique tuples. I.e., $\forall \boldsymbol{A}, \boldsymbol{B} \in \mathcal{P}(\mathcal{C}, i)$ with $\boldsymbol{A} \neq \boldsymbol{B}$, either $[\![\boldsymbol{A}]\!] \prec [\![\boldsymbol{B}]\!]$ or $[\![\boldsymbol{B}]\!] \prec [\![\boldsymbol{A}]\!]$—by definition, each element of $\mathcal{P}(\mathcal{C}, i)$ corresponds to a unique equivalence class. This order induces an order on clique tuples and index tuples, i.e., $\boldsymbol{j}, \boldsymbol{j}' \in [d]^i$, $\boldsymbol{j} \leq \boldsymbol{j}' \Leftrightarrow [\![\boldsymbol{C}(\boldsymbol{j})]\!] \preceq [\![\boldsymbol{C}(\boldsymbol{j}')]\!]$. Within each equivalence class, we assume that tuples are ordered lexicographically.

Theorem 4 (Tuple Mass Factorization) *Suppose that ϕ is the binary, overcomplete sufficient statistic of a discrete state MRF. The tuple mass of any $\boldsymbol{j}$ factorizes:*

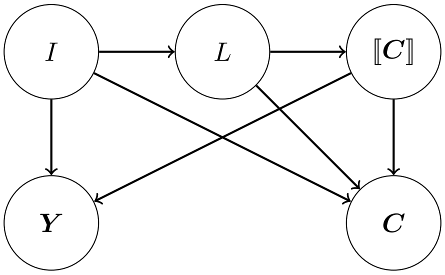

Figure 1: Directed graphical model for the factorization of the tuple mass $\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j}, I = i)$. Any index tuple $\boldsymbol{j}$ can be identified with some pair $(\boldsymbol{C}, \boldsymbol{y})$ of clique tuple and state tuple.

$$\begin{aligned}&\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j} \mid I = i)\\ =&\mathbb{P}(\boldsymbol{C} \mid [\![\boldsymbol{C}]\!], l, i)\mathbb{P}(\boldsymbol{y} \mid [\![\boldsymbol{C}]\!], i)\mathbb{P}([\![\boldsymbol{C}]\!] \mid l)\mathbb{P}(l \mid i)\end{aligned}$$

with

$$\mathbb{P}(l \mid i) = |\mathcal{C}|^{-i} \left\{ {i \atop l} \right\} \binom{|\mathcal{C}|}{l} l!$$

$$\mathbb{P}([\![\boldsymbol{C}]\!] \mid l) = \frac{1}{\binom{|\mathcal{C}|}{l}} \quad \mathbb{P}(\boldsymbol{C} \mid [\![\boldsymbol{C}]\!], l, i) = \frac{1}{\left\{ {i \atop l} \right\} l!}$$

$$\mathbb{P}(\boldsymbol{y} \mid [\![\boldsymbol{C}]\!], i) = \begin{cases} \frac{1}{|\mathcal{X}_{[\![\boldsymbol{C}]\!]}|} & , \boldsymbol{y} \in \mathcal{X}_{[\![\boldsymbol{C}]\!]} \\ 0 & , \textit{otherwise} \end{cases},$$

where l denotes the number of distinct cliques in the clique tuple $\boldsymbol{C}$, $[\![\boldsymbol{C}]\!]$ denotes the equivalence class that contains $\boldsymbol{C}$, and $\boldsymbol{y}$ is the joint state of all cliques in the tuple $\boldsymbol{C}$.

While the proof is rather simple, it is not obvious how to come up with this factorization. The idea is to first draw the equivalence class $[\![\boldsymbol{C}]\!]$, then a uniform member $\boldsymbol{C}$ of this class, then a uniform joint state $\boldsymbol{y}$ of all cliques in $\boldsymbol{C}$. Notice that the sampling steps for $[\![\boldsymbol{C}]\!]$, $\boldsymbol{C}$ and $\boldsymbol{y}$ are uniform, while the probability mass of the number l of distinct cliques that will appear in the tuple is a function of l. Let us now investigate the complexity of our new method.

Theorem 5 (Complexity of Tuple Sampling) *Algorithm 2 samples an index tuple $\boldsymbol{j}$ of given length i from $\mathbb{P}_{\boldsymbol{\zeta},\phi}$ in*

$$\mathcal{O}(k^4 + kn + |\mathcal{C}| + \{i \; l\}^{\top} + l!)$$

steps. When permutations and partitions are precomputed, the runtime reduces to

$$\mathcal{O}(k^4 + kn + |\mathcal{C}|)$$

per sample. Here, k is the polynomial degree, $l \leq i$ is the number of distinct cliques in the generated tuple, and $n = |V|$.

Thus, we found a Monte Carlo algorithm to sample from $\mathbb{P}_{\boldsymbol{\zeta},\phi}$ without any rejection step. Since the algorithm does not use a Markov chain, the generated samples are truly independent. Any number of samples can thus be generated in parallel. Because no data has to be exchanged, the overall runtime scales linearly with the number of processors. This is a superior property compared to MCMC methods, where sampling cannot be parallelized and consecutive samples are not independent. Moreover, the theorem tells us how the complexity of stochastic quadrature is related to the graphical structure and the polynomial degree. The runtime is independent of the parameter dimension d and the state space sizes. In contrast, the runtime of loopy belief propagation (Pearl, 1988; Kschischang et al., 2001) and similar variational techniques (like TRW-BP (Wainwright et al., 2003)) is at least quadratic in the vertex state space sizes.

4 APPROXIMATE ML ESTIMATION

An important feature of maximum-likelihood parameter estimation is that the solution is specified by moment-matching. To illustrate this notion, suppose that we are given an i.i.d. data set $\mathcal{D} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, \dots, \boldsymbol{x}^N\}$ from some unknown measure $\mathbb{P}_{\boldsymbol{\theta}^*}$. By using an exponential family model (which is exact whenever the state space $\mathcal{X}$ is discrete), the log-likelihood of $\boldsymbol{\theta}$ on $\mathcal{D}$ is given by:

$$\ell(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \ln \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}^i) = \langle \boldsymbol{\theta}, \tilde{\boldsymbol{\mu}} \rangle - A(\boldsymbol{\theta})$$

with $\tilde{\boldsymbol{\mu}} = (1/N)\sum_{i=1}^{N} \phi(\boldsymbol{x}^i)$. Taking the derivatives of ℓ w.r.t. some $\boldsymbol{\theta}_i$, we find that $(1/N)\sum_{i=1}^{N} \phi(\boldsymbol{x}^i) = \mathbb{E}_{\boldsymbol{\theta}}[\phi(\boldsymbol{X})_i]$ at any critical point $\boldsymbol{\theta}$ where $\mathbb{E}_{\boldsymbol{\theta}}$ denotes the expectation under $\mathbb{P}_{\boldsymbol{\theta}}$. That is, the maximum-likelihood solution has its moments matched to the empirical average $\tilde{\boldsymbol{\mu}}$. In this section, we show how SCCQ can be used to develop a method for approximate ML estimation that, in analogy to this exact moment-matching, performs a type of pseudo-moment matching. To this end, a means of computing $\nabla A(\boldsymbol{\theta}) = \nabla \ln Z(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\theta}}[\phi(\boldsymbol{X})]$ via SCCQ is required. Recalling that $i \equiv (C, \boldsymbol{y})$ and that $\phi(\boldsymbol{y})$ is binary in discrete models reveals that $\mathbb{E}_{\boldsymbol{\theta}}[\phi(\boldsymbol{X})_i] = \mathbb{P}_{\boldsymbol{\theta}}(\phi(\boldsymbol{X})_i = 1) = \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{X}_C = \boldsymbol{y})$. Since $\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{X}_C = \boldsymbol{y})$ is the marginal probability mass of the event $\{C = \boldsymbol{y}\}$, the problem of computing $\mathbb{E}_{\boldsymbol{\theta}}[\phi(\boldsymbol{X})_i]$ is also called *marginal inference*.

4.1 MARGINAL INFERENCE

For any subset $U \subseteq V$ of variables, and any joint state $\boldsymbol{x}_U$, the marginal density is defined by

$$\begin{aligned}\mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{X}_U = \boldsymbol{x}_U) &= \int_{\mathcal{X}_{V\setminus U}} \mathbb{P}_{\boldsymbol{\theta}}(\boldsymbol{x}_U, \boldsymbol{x}_{V\setminus U}) \,\mathrm{d}\,\nu(\boldsymbol{x}) \\ &= \frac{1}{Z(\boldsymbol{\theta})} \int_{\mathcal{X}_{V\setminus U}} \exp(\langle \boldsymbol{\theta}, \phi(\boldsymbol{x}) \rangle) \,\mathrm{d}\,\nu(\boldsymbol{x}) \ ,\end{aligned}$$

with $\boldsymbol{x} = (\boldsymbol{x}_U, \boldsymbol{x}_{V\setminus U})$. Especially the last integral is reminiscent of the partition function. In fact, it can be interpreted as the partition function of another model with state space $\mathcal{X}_{V\setminus U}$. It is this sum that will be approximated via SCCQ to estimate the marginal. To formalize this idea, we provide adjusted definitions of the SCCQ core concepts. First, we adapt the notion of k-integrability to marginal densities. In accordance to equation (3), we call ϕ marginally k-integrable, if

$$\chi^k_{\phi,U}(\boldsymbol{j}, \boldsymbol{x}_U) = \int_{\mathcal{X}_{V\setminus U}} \prod_{i=1}^{k} \phi(\boldsymbol{x}_U, \boldsymbol{x}_{V\setminus U})_{\boldsymbol{j}_i} \,\mathrm{d}\,\nu(\boldsymbol{x}_{V\setminus U})$$

admits a polynomial time computable closed-form expression for all $\boldsymbol{j} \in [d]^k$, for all $U \subseteq V$, and for all $\boldsymbol{x}_U \in \mathcal{X}_{V\setminus U}$. The difference to ordinary k-integrability is merely symbolical. In fact, all k-integrable statistics that are mentioned in this paper are also marginally k-integrable. Moreover, marginally k-integrable statistics give rise to the marginal tuple mass $\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j}, \boldsymbol{X}_U = \boldsymbol{x}_U, I = i)$ in the same way how the ordinary tuple mass from equation (6) arises from ordinary k-integrability. Moreover, the marginal tuple mass factorizes.

Corollary 1 (Marginal Tuple Mass Factorization) *Suppose that ϕ is the binary, overcomplete sufficient statistic . The marginal tuple mass factorizes:*

$$\begin{aligned}&\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j}, I = i, \boldsymbol{X}_U = \boldsymbol{x}_U) \\ =&\mathbb{P}(\boldsymbol{C} \mid \llbracket \boldsymbol{C} \rrbracket, l, i)\mathbb{P}(\boldsymbol{y}, \boldsymbol{x}_U \mid \llbracket \boldsymbol{C} \rrbracket, i)\mathbb{P}(\llbracket \boldsymbol{C} \rrbracket \mid l)\mathbb{P}(l \mid i)p(i)\end{aligned}$$

where $\mathbb{P}(l \mid i)$, $\mathbb{P}(\llbracket \boldsymbol{C} \rrbracket \mid l)$, and $\mathbb{P}(\boldsymbol{C} \mid \llbracket \boldsymbol{C} \rrbracket, l, i)$ are given by Theorem 4, and

$$\mathbb{P}(\boldsymbol{y}, \boldsymbol{x}_U \mid \llbracket \boldsymbol{C} \rrbracket, i) = \begin{cases} \frac{1}{|\mathcal{X}_{\llbracket \boldsymbol{C} \rrbracket \cup U}|} & , \boldsymbol{y} \in \mathcal{X}_{\llbracket \boldsymbol{C} \rrbracket} \\ & \wedge \nexists v \in U : \boldsymbol{x}_v \neq \boldsymbol{y}_v \\ 0 & , \textit{otherwise} \ . \end{cases}$$

The ordinary tuple mass $\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j}, I = i)$ and the marginal tuple mass $\mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j}, \boldsymbol{X}_U = \boldsymbol{x}_U, I = i)$ differ only in the factor $\mathbb{P}(\boldsymbol{y}, \boldsymbol{x}_U \mid \llbracket \boldsymbol{C} \rrbracket, i)$. We may hence use their quotient as importance weight to convert SCCQ samples for the partition function into SCCQ samples for

Algorithm 3: SCCQ Marginal Inference

```
input $\boldsymbol{\theta}, k, \boldsymbol{\zeta}, N$
output Pseudo marginals $\hat{\boldsymbol{\mu}}$
 1: $\boldsymbol{S} \leftarrow \boldsymbol{0}, \boldsymbol{m} \leftarrow \boldsymbol{0}$, done $\leftarrow$ false
 2: while $\exists i : \boldsymbol{m}_i < N$ do
 3:     $(\boldsymbol{j}, i) \sim \mathbb{P}_{\boldsymbol{\zeta},\phi}(\cdot, \boldsymbol{x}_C)$
 4:     for $C \in \mathcal{C}$ do
 5:         for $\boldsymbol{x}_C \in \mathcal{X}_C$ do
 6:             if $\mathrm{agree}(\boldsymbol{j}, i, C, \boldsymbol{x}) \wedge \boldsymbol{m}_i < N$ then
 7:                 $\boldsymbol{S}_l \leftarrow \boldsymbol{S}_l + \hat{Z}^k_{\boldsymbol{j},i}(\boldsymbol{\theta}) \frac{p(\boldsymbol{y}(\boldsymbol{j}),\boldsymbol{x}_C | [\![\boldsymbol{C}(\boldsymbol{j})]\!],i)}{p(\boldsymbol{y}(\boldsymbol{j}) | [\![\boldsymbol{C}(\boldsymbol{j})]\!],i)}$
 8:                 $\boldsymbol{m}_i \leftarrow \boldsymbol{m}_i + 1$
 9:             end if
10:         end for
11:     end for
12: end while
13: for $C \in \mathcal{C}$ do
14:     for $\boldsymbol{x}_C \in \mathcal{X}_C$ do
15:         $\hat{\boldsymbol{\mu}}_{C=\boldsymbol{x}_C} \leftarrow \frac{\boldsymbol{S}_{C=\boldsymbol{x}_C}}{\sum_{\boldsymbol{x}_C \in \mathcal{X}_C} \boldsymbol{S}_{C=\boldsymbol{x}_C}}$
16:     end for
17: end for
```

marginal probabilities. We have

$$
\begin{aligned}
&\mathbb{E}_{\boldsymbol{J},I}\left[\frac{p(\boldsymbol{y}(\boldsymbol{J}), \boldsymbol{x}_U \mid [\![\boldsymbol{C}(\boldsymbol{J})]\!], I)}{p(\boldsymbol{y}(\boldsymbol{J}) \mid [\![\boldsymbol{C}(\boldsymbol{J})]\!], I)} \hat{Z}^k_{\boldsymbol{J},I}(\boldsymbol{\theta})\right] \\
=&\sum_{i=0}^{k} \sum_{\boldsymbol{j} \in [d]^i} \mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j}, I = i) w_{\boldsymbol{j},i,U} \hat{Z}^k_{\boldsymbol{j},i}(\boldsymbol{\theta}) \\
=&\sum_{i=0}^{k} \sum_{\boldsymbol{j} \in [d]^i} \mathbb{P}_{\boldsymbol{\zeta},\phi}(\boldsymbol{J} = \boldsymbol{j}, I = i, \boldsymbol{X}_U = \boldsymbol{x}_U) \hat{Z}^k_{\boldsymbol{j},i}(\boldsymbol{\theta}) \\
=&\mathbb{E}_{\boldsymbol{J},I,\boldsymbol{X}_U=\boldsymbol{x}_U}\left[\hat{Z}^k_{\boldsymbol{J},I}(\boldsymbol{\theta})\right] ,
\end{aligned}
$$

with importance weight $w_{\boldsymbol{j},i,U} = \frac{p(\boldsymbol{y}(\boldsymbol{j}),\boldsymbol{x}_U | [\![\boldsymbol{C}(\boldsymbol{j})]\!],i)}{p(\boldsymbol{y}(\boldsymbol{j}) | [\![\boldsymbol{C}(\boldsymbol{j})]\!],i)}$.

Now, we gathered all parts which are required for marginal inference. The corresponding inference procedure is provided in Algorithm 3. While the main idea is to perform d separate runs of Algorithm 1, such a naive approach would result in an unnecessary high runtime. Instead, we make use of Corollary 1 to propose an importance sampling approach, in which each SCCQ sample is shared among all marginals. For each marginal $p(\boldsymbol{X}_C = \boldsymbol{x}_C)$, we validate if the pair $(\boldsymbol{j}, i)$ that is sampled in line 3 agrees with the assignment $\boldsymbol{x}_C$ (line 6)—otherwise, its marginal tuple mass is zero. If they agree, we reweigh the sample, perform the summation and count the number of successes in lines 7 and 8. In lines 13–17, the estimated sums are normalized and written to $\hat{\boldsymbol{\mu}}$.

4.2 PARAMETER ESTIMATION

With Algorithm 3, we can compute the log-likelihood's gradient $\nabla \ell(\boldsymbol{\theta})$, and employ any first-order method to estimate the parameters. To measure the progress of parameter estimation, it is convenient to estimate the log-likelihood of the model, which inherits its computational complexity from the log-partition function. Before we proceed to some experimental results, we close this section by translating the SCCQ error bound from Theorem 2 to an error bound on the log-likelihood.

Theorem 6 (SCCQ Log-Likelihood Error) *Assume that the preconditions of Theorem 2 hold. Let $\hat{\ell}(\boldsymbol{\theta}) = \langle \boldsymbol{\theta}, \tilde{\boldsymbol{\mu}} \rangle - \ln \hat{Z}^{N,k}_{\boldsymbol{\zeta}}(\boldsymbol{\theta})$ be the SCCQ approximation to the log-likelihood. Whenever the outcome $\hat{Z}^{N,k}_{\boldsymbol{\zeta}}(\boldsymbol{\theta})$ of Algorithm 1 is positive, we have*

$$
\mathbb{P}\left[|\hat{\ell}(\boldsymbol{\theta}) - \ell(\boldsymbol{\theta})| < \frac{\epsilon Z(\boldsymbol{\theta})}{\min\{\hat{Z}^{N,k}_{\boldsymbol{\zeta}}(\boldsymbol{\theta}), Z(\boldsymbol{\theta})\}}\right] \geq 1 - \delta \ .
$$

That is, with probability of at least $1 - \delta$, the absolute error in the approximated log-likelihood is roughly ϵ when $\hat{Z}^{N,k}_{\boldsymbol{\zeta}}(\boldsymbol{\theta})$ and $Z(\boldsymbol{\theta})$ have the same order of magnitude.

5 EXPERIMENTAL DEMONSTRATION

Theoretical insights from the previous sections do provably reduce the computational complexity. Moreover, pseudo marginals, based on unbiased estimates of the quadrature approximation to the partition function, facilitate approximate maximum-likelihood estimation. We conduct a small set of experiments to assess our methods empirically and answer the following questions:

Q1 What is the runtime improvement when $\|\chi^k_\phi\|$ is computed via Theorem 3 instead of naive summation?

Q2 What is the runtime improvement when index tuples are sampled with Algorithm 2 instead of rejection sampling?

Q3 Does SCCQ-based approximate maximum-likelihood estimation work in practice?

To answer **Q1**, we measure the runtime in nanoseconds for computing $\|\chi^k_\phi\|$ via Theorem 3 and via naive summation on a 4×4 binary Ising grid for polynomial degree $k \in \{1, 2, 3, 4\}$. The results are depicted in the leftmost plot of Figure 2. All results are averaged over 10 independent runs and error-bars show the standard deviation (if any). The runtime is shown in log-scale. Normalizing the tuple mass via Theorem 3 is several orders

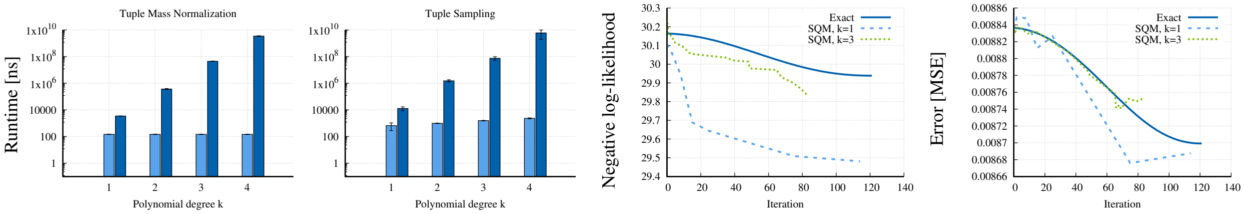

Figure 2: From left to right: (1) Runtime in log-scale for computing $\|\chi_\phi^k\|$ with Theorem 3 (light blue) and naive summation (dark blue). Runtime in log-scale for drawing a single index tuple via Algorithm 2 (light blue) and rejection sampling (dark blue). (3) and (4): Progress of (approximate) negative log-likelihood $-\ell(\boldsymbol{\theta})$ and mean squared error (MSE) between predicted and empirical marginals during parameter estimation on the mushroom data set. The solid line indicates the exact outcome, while the dashed lines represent SCCQ results with $N = 10^4$ samples and $k \in \{1, 3\}$.

of magnitude faster than the standard approach, as expected. Regarding **Q2**, the situation looks similar. The corresponding results are depicted in the second plot of Figure 2. We see that increasing the polynomial degree and thus the maximal tuple length increases the runtime of rejection sampling. Clearly, the proportion of rejected samples increases when the state space size of the random tuples increases. On the other hand, the runtime of Algorithm 2 is almost constant in practice. To answer **Q3**, a regularized maximum-likelihood estimation on the mushroom data set[2] is conducted. The set contains 5644 fully observed training instances. Each data point $\boldsymbol{x}$ consists of 23 categorical features with up to 9 different states, representing properties of mushrooms. In total, $|\mathcal{X}| \approx 2^{43}$. To facilitate exact computation of likelihood and marginals, we use the Chow-Liu tree (Chow and Liu, 1968) as the conditional independence structure of the model. Note, however, that SCCQ is completely oblivious of the graphical structure. Hence, the reported results are valid for intractable non-tree-structured models as well. To prevent the model parameters from becoming too large, l_1-regularization with $\lambda = {}^{1}\!/_{2}$ is applied. The actual parameter estimation is carried out via the fast iterative shrinkage-thresholding algorithm (FISTA) (Beck and Teboulle, 2009) with stepsize $1/L$ where L is an upper bound on the log-likelihood's gradient's Lipschitz constant. We run SCCQ with $N = 10^4$ Monte Carlo samples. In each training iteration, we assess the (approximate) negative log-likelihood and the mean squared error (MSE) between predicted and empirical marginal probabilities. The last two plots of Figure 2 show the corresponding results. Each line corresponds to one parameter estimation. Since the runs converge in different iterations, the three lines have slightly different lengths. The results show that even the very coarse linear ($k = 1$) approximation yields a reasonable approximate log-likelihood and approximate marginals. The learning process evolves similar to the exact computation. When the polynomial degree is increased to $k = 3$, the approximation is even closer to the exact outcome as predicted by the theory. Especially the SCCQ marginal probabilities are often indistinguishable from the exact marginals.

6 CONCLUSION

We presented the first complete framework for SCCQ-based parameter learning for undirected graphical models. Quadrature-based inference provides bounds on the partition partition. However, the complexity of existing algorithms is exponential in the degree of the underlying polynomial approximation and polynomial in the dimension of the model's parameter vector—the accompanying computational complexity is not practical. We provide accelerated SCCQ algorithms whose complexity is independent of the dimension. Our empirical evaluation shows that the new algorithms are several orders of magnitude faster. In addition, we provide the first algorithm for SCCQ-based marginal inference whose practical speed and accuracy are sufficient to be used for approximate maximum-likelihood estimation. Hence, SCCQ is a highly parallel drop-in replacement for MCMC and message-passing whenever the parameter norm is bounded (e.g., via regularization). Finally, we explained how the stochastic quadrature can be applied to models with continuous random variables, which opens new research opportunities, e.g., inference in exponential family models with mixed domains, where some dimensions are discrete and others are continuous.

Acknowledgements

This work has been supported by Deutsche Forschungsgemeinschaft (DFG) within the Collaborative Research Center SFB 876 "Providing Information by Resource-Constrained Analysis", project A1.

[2] `https://archive.ics.uci.edu/ml/datasets/mushroom`

References

Christophe Andrieu, Nando de Freitas, Arnaud Doucet, and Michael I. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50(1-2):5–43, 2003. doi: 10.1023/A:1020281327116.

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1): 183–202, 2009. doi: 10.1137/080716542.

Julian Besag. Statistical analysis of non-lattice data. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 24(3):179–195, 1975. ISSN 00390526, 14679884. doi: 10.2307/2987782.

Manuel Bronstein. Integration of elementary functions. *Journal of Symbolic Computation*, 9(2):177–173, 1990. doi: 10.1016/S0747-7171(08)80027-2.

Bill P. Buckles and M. Lybanon. Algorithm 515: Generation of a vector from the lexicographical index [g6]. *ACM Transactions on Mathematical Software*, 3(2):180–182, June 1977. ISSN 0098-3500. doi: 10.1145/355732.355739.

Andrei Bulatov and Martin Grohe. The complexity of partition functions. In Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella, editors, *Automata, Languages and Programming*, volume 3142 of *Lecture Notes in Computer Science*, pages 294–306. Springer, Heidelberg, Germany, 2004.

C.K. Chow and C.N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14 (3):462–467, 1968. ISSN 0018-9448. doi: 10.1109/TIT.1968.1054142.

C.W. Clenshaw and A.R. Curtis. A method for numerical integration on an automatic computer. *Numerische Mathematik*, 2(1):197–205, 1960.

Gideon Ehrlich. Loopless algorithms for generating permutations, combinations, and other combinatorial configurations. *Journal of the ACM*, 20(3):500–513, 1973. doi: 10.1145/321765.321781.

W. Fraser. A survey of methods of computing minimax and near-minimax polynomial approximations for functions of a single independent variable. *Journal of the ACM*, 12(3):295–314, July 1965.

W. Gautschi. Questions of numerical condition related to polynomials. *Studies in Numerical Analysis*, (24): 140–177, 1985.

John Michael Hammersley and Peter Clifford. Markov fields on finite graphs and lattices. *Unpublished manuscript*, 1971.

Ernst Ising. Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift für Physik*, 31:253–258, 1925.

Elliott Ward Cheney Jr. *Introduction to Approximation Theory*. Amer Mathematical Society, 2nd edition, 1966. ISBN 978-0821813744.

Frank R. Kschischang, Brendan J. Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

J.C. Mason and David C. Handscomb. *Chebyshev polynomials*. Chapman and Hall/CRC, 1st edition, 2002. ISBN 9780849303555.

Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers, Burlington, MA, USA, 1988.

Nico Piatkowski and Katharina Morik. Stochastic discrete clenshaw-curtis quadrature. In *Proceedings of the 33rd International Conference on Machine Learning*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 3000–3009. JMLR.org, 2016.

Edwin James George Pitman. Sufficient statistics and intrinsic accuracy. *Mathematical Proceedings of the Cambridge Philosophical Society*, 32:567–579, 1936.

Charles Sutton and Andrew McCallum. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373, 2011.

Lloyd N. Trefethen. Is gauss quadrature better than clenshaw-curtis? *SIAM Review*, 50(1):67–87, 2008. doi: 10.1137/060659831.

Leslie Gabriel Valiant. The complexity of enumeration and reliability problems. *SIAM Journal on Computing*, 8(3):410–421, 1979.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1 (1–2):1–305, 2008.

Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. Tree-reweighted belief propagation algorithms and approximate ML estimation by pseudo-moment matching. In Christopher M. Bishop and Brendan J. Frey, editors, *9th Workshop on Artificial Intelligence and Statistics*. Society for Artificial Intelligence and Statistics, Key West, FL, 2003.

Shuhuang Xiang, Xiaojun Chen, and Haiyong Wang. Error bounds for approximation in Chebyshev points. *Numerische Mathematik*, 116(3):463–491, 2010.