Original Paper

Human Heredity

Hum Hered 2016;82:1–15
DOI: 10.1159/000475465

Received: December 13, 2016
Accepted: April 1, 2017
Published online: July 21, 2017

Efficient Maximum Likelihood Estimation for Pedigree Data with the Sum-Product Algorithm

Alexander Engelhardt[a] Anna Rieger[a] Achim Tresch[c] Ulrich Mansmann[a, b]

[a]Institute for Medical Informatics, Biometry and Epidemiology and [b]Institute for Statistics, Ludwig Maximilian University, Munich, and [c]Institute for Medical Statistics and Computational Biology, University Clinic Cologne, Cologne, Germany

Keywords

Colorectal cancer · Personalized medicine · Cancer risk prediction · Pedigrees · EM algorithm · Factor graphs · Sum-product algorithm

Abstract

Objective: We analyze data sets consisting of pedigrees with age at onset of colorectal cancer (CRC) as phenotype. The occurrence of familial clusters of CRC suggests the existence of a latent, inheritable risk factor. We aimed to compute the probability of a family possessing this risk factor as well as the hazard rate increase for these risk factor carriers. Due to the inheritability of this risk factor, the estimation necessitates a costly marginalization of the likelihood. ***Methods:*** We propose an improved EM algorithm by applying factor graphs and the sum-product algorithm in the E-step. This reduces the computational complexity from exponential to linear in the number of family members. ***Results:*** Our algorithm is as precise as a direct likelihood maximization in a simulation study and a real family study on CRC risk. For 250 simulated families of size 19 and 21, the runtime of our algorithm is faster by a factor of 4 and 29, respectively. On the largest family (23 members) in the real data, our algorithm is 6 times faster. ***Conclusion:*** We introduce a flexible and runtime-efficient tool for statistical inference in biomedical event data with latent variables that opens the door for advanced analyses of pedigree data.

Introduction

Colorectal cancer (CRC) is one of the most prevalent cancer diseases in Europe and the United States [1], with men having a younger average age at diagnosis [2]. For a small proportion of CRC cases, genetic predispositions are known [3]. Interestingly, an additional 15–20% of CRC cases occur in familial clusters [4]. Within these clusters, family members show a higher risk of contracting CRC [5]. The cause for these clusters is unknown but assumed to be a risk factor which may be of genetic or environmental origin.

Since cancer develops earlier in these high-risk families, it is of interest to identify them in advance. Subsequently, health insurances can allow members of high-risk families to join screening programs at an earlier age. In this paper, we therefore develop an efficient risk calculator for CRC, i.e., a method for clinicians to assess the familial risk for a specific family based on the family's CRC history.

E-Mail karger@karger.com
www.karger.com/hhe

Alexander Engelhardt
Institute for Medical Informatics, Biometry and Epidemiology
Ludwig Maximilian University
Marchioninistrasse 15, DE–81377 Munich (Germany)
E-Mail engelhardt@ibe.med.uni-muenchen.de

To do this, we look at data consisting of a set of pedigrees, where each person has an inheritable latent variable, the risk factor, that influences its phenotype, the age at CRC diagnosis. Assuming an inheritance model and a penetrance model, we aim to estimate 2 parameters: the a priori probability p_1 for a founder to carry the risk factor, and the penetrance α (i.e., the multiplicative increase of the hazard rate of an individual that carries the risk factor).

A closely related subject is *complex segregation analysis*. Segregation analysis evaluates whether pedigree data of affected and unaffected offspring agree with a mendelian transmission mode and perform hypothesis tests for different models of inheritance [6]. Complex segregation analysis can go one step further and work with pedigrees of arbitrary structure instead of nuclear families and quantitative traits as well as qualitative traits [7]. We perform a kind of segregation analysis but do not test for a specific genetic model. In accordance with the argument in Houle et al. [8], we employ a phenotype-based approach to study a generic inheritance mechanism, because the details of genetic causation of CRC are still unknown and complex, and the assumptions of a genotype-based approach may not hold true.

This problem has been approached in previous work of our group [9]. Since the latent variable is unknown but influences the likelihood, a straightforward estimation procedure has to marginalize the likelihood respective to them. The inheritability of this latent variable means that observations within a family are dependent, and the marginalization cannot happen on the level of a single person, but over a whole family. Since each latent variable can assume 1 of 2 values (risk factor present: yes/no), the complexity of computing this sum is $\mathcal{O}(2^D)$, where D is the number of family members.

The runtime of this straightforward optimization over the marginalized likelihood is still reasonable when no family has an excessive number of members. However, the number of possible risk constellations within a family grows 2-fold with each new family member. As soon as even 1 family is sufficiently large, the marginalization quickly becomes unfeasible. In these situations, an alternative approach is needed.

The new aspect in this paper is the implementation of an expectation-maximization (EM) algorithm for situations when some families are too large for the marginalization procedure. The E-step is nontrivial because the latent variables within a pedigree are dependent, and a straightforward calculation of the marginal posteriors would again be of exponential runtime. For a linear dependency structure (such as in a Hidden Markov Model), the Baum-Welch algorithm [10] is an efficient method for solving the E-step. In our problem, the data instead show dependency in a tree structure. This dependency structure necessitates using the sum-product algorithm [11] to obtain the marginalized posterior probabilities for the latent variables in the E-step. A similar approach for the marginalization over hidden variables has been proposed and implemented by Failmezger et al. [12], yet in the completely different context of single cell time lapse image analysis.

We show that the runtime of our EM algorithm is linear instead of exponential in terms of the pedigree size. We also executed a simulation study to show that our algorithm correctly recovers the specified parameters. Finally, we demonstrate the runtime improvement of our algorithm on a real data set: a family study of CRC cases in Upper Bavaria.

Methods

Nomenclature

The data set is composed of families which are represented as pedigrees (Fig. 1a). We call individuals at the top of the pedigree (i.e., with unspecified parents) *founder nodes* and all other persons *nonfounders*. Individuals without any offspring (i.e., at the bottom of the pedigree) are called *final* individuals.

We denote by t_i the chronological age in years at clinical onset of CRC for each person $i = 1, \ldots, n$, if the corresponding censoring indicator c_i equals 1, and the age at censoring if $c_i = 0$. The gender of an observation is denoted by m_i, which is 1 for males and 0 for females. The observed data for 1 person is thus $x_i = (t_i, c_i, m_i)$.

Each person also has a latent variable z_i which equals 1 if this person is a risk carrier and 0 if not. We use $\male_i$ and $\female_i$ to denote the position (i.e., the value of i) of the father and mother of person i. For example, if we have a risk status z_i for a nonfounder i, his father's risk status is $z_{\male i}$. We denote the set of all i that are founder nodes by F. The complete data vectors for all patients are called x and z, respectively.

Penetrance Model

For persons where $z_i = 1$, we assume an elevated relative risk of developing CRC, which manifests itself through a hazard rate increased by a multiplicative factor α, the *penetrance* [5]. This parameter is unknown and will be estimated.

We assume a Weibull distribution for t_i, because it is a parametric distribution which fits observed CRC incidence curves quite well. The Weibull hazard rate is given by $h(t) = k\lambda^k t^{k-1}$, with the parameters $k > 0$ and $\lambda > 0$. In our relative risk model, we multiply the hazard rate by α if $z_i = 1$ and, additionally, by β if $m_i = 1$. These factors model the increased relative risk for risk carriers and males, respectively. Our hazard rate for an event (i.e., diagnosis of CRC) is then

$$h(t_i) = k\lambda^k t_i^{k-1}\, \alpha^{z_i}\, \beta^{m_i}.$$

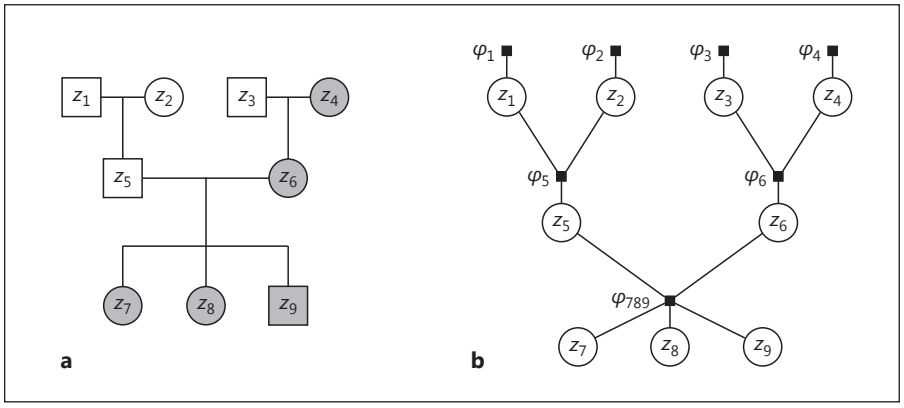

Fig. 1. A pedigree and its corresponding factor graph. **a** A sample pedigree of a family with 9 members. Squares denote males, circles females. A couple (a connected circle and square in the same row) gives rise to a set of children (the nodes connected to this couple in the row below). Persons shaded in grey are risk carriers. The 4 grandparents in the top row are the *founder nodes* in this family, the other 5 persons are *nonfounders*. **b** A factor graph visualizing the factorization of $g(z) = f(x, z)$ (Equation 2) for the family from Figure 1a. Circles represent variable nodes, and filled squares represent factor nodes (i.e., local functions). The edges show which variables are arguments to which factor. For example, the factor φ_6 has 3 arguments: $\varphi_6(z_3, z_4, z_6)$.

The survival function is defined by $S(t) = \exp(-\int_0^t h(u)\mathrm{d}u)$. With the additional relative risk factors, this becomes

$$S(t_i) = \exp(-(t_i\lambda)^k \alpha^{z_i} \beta^{m_i}).$$

The density for 1 observation i is composed of the product of the survival function and (for uncensored observations) the hazard rate:

$$f(t_i|z_i) = h(t_i)^{c_i} \cdot S(t_i).$$

The observations x_i are conditionally independent given z_i, and the density of the whole data $f(x|z, \theta)$ can be split up into a product of individual densities: $f(x|z, \theta) = \Pi_i f(x_i|z_i, \theta)$.

Heritage Model

The *founder prevalence*, i.e., the a priori probability $\mathrm{P}(Z_i = 1)$ for a founder node to carry the risk factor, is called p_1. This parameter will be estimated. The probability for a nonfounder to be a risk carrier is dependent on its parents' risk statuses and the *inheritance probability* p_H. Our model does not allow for spontaneous mutations to risk carrier. If any one of both parents passes down a risk factor $z_{\sigma i} = 1$ or $z_{\varphi i} = 1$ with the probability p_H, then the probability for the offspring to be a risk carrier is

$$\tilde{p}_i = \mathrm{P}(Z_i = 1|z_{\sigma i}, z_{\varphi i}) = p_H z_{\sigma i} + p_H z_{\varphi i} - p_H^2 z_{\sigma i} z_{\varphi i}. \quad (1)$$

We denote $\mathrm{P}(Z_i = 1|z_{\sigma i}, z_{\varphi i})$ for nonfounders by $\tilde{p}_i$ to emphasize the distinction from p_1 for founders.

A sensitivity analysis found that varying the value of p_H has a negligible effect on the final parameter estimates [9], and thus we chose $p_H = 0.5$ for all our analyses.

Given a predefined inheritance probability p_H and a founder prevalence p_1, the probability for a risk vector Z for the entire dataset becomes

$$\begin{aligned}\mathrm{P}(z) &= \prod_{i\in F} \mathrm{P}(z_i) \cdot \prod_{i\notin F} \mathrm{P}(z_i|z_{\sigma i}, z_{\varphi i}) \\ &= \prod_{i\in F} p_1^{z_i}(1-p_1)^{1-z_i} \cdot \prod_{i\notin F} \tilde{p}_1^{z_i}(1-\tilde{p}_1)^{1-z_i}\end{aligned}$$

Likelihoods

All Weibull parameters (k, λ) as well as the inheritance probability p_H and the risk increase for males (β) are assumed to be known. We set $k = 4$ and $\lambda = 0.0058$ according to Rieger and Mansmann [9], $\beta = 2$ according to Kolligs et al. [2], and $p_H = 0.5$. The complete likelihood, where both x and z are observed, is then

$$L(\theta; x, z) = f(x, z) = f(x|z)\mathrm{P}(z). \quad (2)$$

The 2 factors $f(x|z)$ and $\mathrm{P}(z)$ were defined in the penetrance model and the inheritance model, respectively. The parameter vector in our model is $\theta = (p_1, \alpha)$.

The complete log-likelihood becomes (derivation in Appendix A1)

$$\begin{aligned}l(\theta; x, z) = \text{const} &+ \left(\sum_{i\in F} z_i\right)\log p_1 + \left(|F| - \sum_{i\in F} z_i\right)\log(1-p_1) \\ &+ \sum_{i=1}^{n} c_i z_i \log\alpha - (t_i\lambda)^k \alpha^{z_i} \beta^{m_i}. \quad (3)\end{aligned}$$

We marginalize the nonreduced form of the complete likelihood to obtain the incomplete likelihood $L(\theta; x)$ [13, Equation 1.5]:

$$L(\theta; x) = \sum_z L(\theta; x, z). \quad (4)$$

To estimate the parameters p_1 and α, one could use a Nelder-Mead optimization [14] on the marginalized likelihood $L(\theta; x)$. However, for a family of size D, the sum over all z has 2^D elements. Even when splitting the sum up across all families (Appendix A3), the number of summands grows exponentially with increasing family size D. Thus, for large families, the computation of the marginalization within the likelihood evaluation quickly becomes unfeasible.

The EM Algorithm

A common approach for finding maximum likelihood estimates in the presence of latent variables is to make use of the EM algorithm. Resources on the EM algorithm are plentiful, including

a short tutorial [15], the seminal paper by Dempster et al. [16], and an entire book [17] devoted to the subject.

In short, the EM algorithm proceeds in a loop over 2 steps: in the *E-step*, one calculates the expected log-likelihood over the latent variables Z, given the observed data and the current parameter estimates. This problem reduces to computing complete-data sufficient statistics [16]. In the subsequent *M-step*, one then updates the estimates of the parameters, given the new expected sufficient statistics from the E-step.

As a convergence criterion, frequent choices include the size of the relative change of either the log-likelihood or the parameter estimates [18].

We use the size of the relative change of the parameter estimates for α and p_1 as a stopping criterion. This criterion is more conservative than using the log-likelihood [18], and we are on the safe side by letting the algorithm run a bit longer than it would have to.

To compute the expected log-likelihood $Q(\theta; \theta^{(t)})$, we introduce the *membership probabilities* [17, p. 43] $T_i^{(t)}$, i.e., the probability for *one* person's risk status Z_i, given the *whole* observed data x (derivation available in Appendix A2):

$$\begin{aligned} T_i^{(t)} &= \mathrm{E}_{Z \mid x, \theta^{(t)}}\left(Z_i\right) \\ &= \mathrm{E}_{Z_i \mid x, \theta^{(t)}}\left(Z_i\right) \\ &= \mathrm{P}\left(Z_i = 1 \mid x, \theta^{(t)}\right) \\ &= \sum_z z_i \, \mathrm{P}\left(z \mid x, \theta^{(t)}\right). \end{aligned} \tag{5}$$

Here, the summation is over all admissible combinations of z_i (i.e., $\mathrm{P}(z) > 0$ and $z_i = 1$). The condition on the entire observed data x and the summation over all z will conveniently reduce to a condition on and summation of only the respective family's data x and z (Appendix A3). The target function Q becomes (cf. Equation 3)

$$\begin{aligned} Q\left(\theta; \theta^{(t)}\right) &= E_{Z \mid x, \theta^{(t)}}\left[\, l\left(\theta; x, Z\right)\right] \\ &= \text{const} + \log\left(p_1\right)\left(\sum_{i \in F} T_i^{(t)}\right) + \log\left(1 - p_1\right)\left(\sum_{i \in F}\left(1 - T_i^{(t)}\right)\right) \\ &\quad + \sum_{i=1}^{n} T_i^{(t)} \, c_i \log \alpha - \left(t_i \, \lambda\right)^k \beta^{m_i}\left(T_i^{(t)} \, \alpha + \left(1 - T_i^{(t)}\right)\right). \end{aligned} \tag{6}$$

The M-Step

For the M-step, we maximize $Q(\theta; \theta^{(t)})$ respective to α and p_1 to obtain the new parameter estimates for iteration $t + 1$. Once the values of all $T_i^{(t)}$ are known, the maximization of Q with respect to p_1 and α is straightforward and has a closed form solution:

$$\alpha^{(t+1)} = \frac{\sum_{i=1}^{n} c_i \, T_i^{(t)}}{\sum_{i=1}^{n} T_i^{(t)}\left(t_i \, \lambda\right)^k \beta^{m_i}}. \tag{7}$$

$$p_1^{(t+1)} = \frac{\sum_{i \in F} T_i^{(t)}}{|F|}. \tag{8}$$

The E-Step

It follows from Equations 6–8 that, as in the "standard" examples of the EM algorithm, the E-step conveniently reduces to computing the complete-data sufficient statistics $T_i^{(t)}$. The reason for this simplification is the fact that the log-likelihood is linear in the latent data Z. Computing $Q(\theta; \theta^{(t)})$ thus simplifies to replacing each occurring Z_i by its conditional expectation $T_i^{(t)}$. The M-step then uses these "imputed" values of the latent data Z for the updated parameter estimates.

Marginalization of the Joint Density

In our setting, the difficulty in computing $T_i^{(t)}$ is that the probability for the risk status of 1 family member Z_i is conditioned on the observed data x of the *entire family* (Appendix A3). To compute these values, we could marginalize over all risk vectors z where $z_i = 1$, i.e., $T_i^{(t)} \sum_r \mathrm{P}(Z = r \mid x, \theta^{(t)})$, where r is a valid risk vector (i.e., with $\mathrm{P}(Z = r) > 0$ and with $z_i = 1$). However, this requires the same exponential runtime as in a Nelder-Mead optimization.

Alternatively, a pedigree can be represented as a Bayesian network [19, 20], also known as a causal probabilistic network, which in turn can be converted into a factor graph [11]. This representation is advantageous because it allows the efficient computation of marginals via the sum-product algorithm.

The sum-product algorithm [11], also known as the belief propagation algorithm, computes marginalizations of the form of $T_i^{(t)}$ in linear runtime [21, p. 290]. It does this by representing a complex "global" function $g(z)$ – here, $f(x, z \mid \theta^{(t)})$ – as a factor graph, i.e., a product of multiple "local" functions, $\Pi_j \varphi_j$, each depending on only a subset of the arguments in $g(z)$.

The sum-product algorithm then exploits this structure to efficiently compute marginalizations of $g(z)$ – here, we marginalize the joint density to obtain $f(z_i, x \mid \theta^{(t)})$. By dividing this joint density through $f(x) = f(Z_i = 1, x \mid \theta^{(t)}) + f(Z_i = 0, x \mid \theta^{(t)})$, we ultimately obtain $T_i^{(t)} = \mathrm{P}(Z_i = 1 \mid x, \theta^{(t)})$, which was our actual goal.

Factor Graphs

Factor graphs were first introduced by Kschischang et al. [11] to represent factorizations of multivariate functions.

The factor graph in Figure 1b encodes the joint density $g(z) = f(z, x)$ of the family from Figure 1a as the product of 7 factors φ_j:

$$\begin{aligned} g(z) = \varphi_1(z_1) \cdot \varphi_2(z_2) \cdot \varphi_3(z_3) \cdot \varphi_4(z_4) \cdot \varphi_5(z_1, z_2, z_5) \\ \cdot \varphi_6(z_3, z_4, z_6) \cdot \varphi_{789}(z_5, z_6, z_7, z_8, z_9). \end{aligned} \tag{9}$$

The factors are defined as

$$\varphi_J(z_J, z_{\sigma J}, z_{\varphi J}) = \prod_{j \in J} f(x_j \mid z_j) \mathrm{P}(z_j \mid z_{\sigma j}, z_{\varphi j}),$$

where J is the set of all children with the same parents, which are denoted by $z_{\sigma J}$ and $z_{\varphi J}$. If φ_J is a factor for a founder node, then $z_{\sigma J}$ and $z_{\varphi J}$ are defined as an empty set and the respective probability $\mathrm{P}(z_j)$ is unconditioned. The exemplary factors for Equation 9 are available in Appendix A4.

The factor φ_{789} (Fig. 1b) cannot be split up into 3 factors because the graph edges would then form a *cycle*, which is not allowed, or would necessitate a costly *loopy belief propagation* procedure [11, 22]. Instead, we implement a *clustering* procedure [11] and group the respective densities into 1 factor per set of parents.

The Sum-Product Algorithm

Having set up a factor graph for each family, we then apply the sum-product algorithm to compute marginalizations of $f(z, x)$ at each variable node z_i, i.e., $f(z_i, x)$. In our setting, we restrict our-

selves to family *trees*, i.e., we do not allow for consanguineous marriages (see Fig. 2 of Goddard et al. [23] for a counterexample), which would again lead to cycles in the corresponding factor graph.

Let $\mu_{z \to \varphi}(z)$ denote the message sent from a variable node z to a factor node φ, and let $\mu_{\varphi \to z}(z)$ denote the message sent from a factor node φ to a variable node z. Furthermore, let $n(v)$ denote the set of neighboring nodes of a (factor or variable) node v.

We then define the messages from a variable node to a factor node, and from a factor node to a variable node, as follows [11]:

$$\mu_{z \to \varphi}(z) = \prod_{h \in n(z) \setminus \{\varphi\}} \mu_{h \to z}(z)$$

$$\mu_{\varphi \to z}(z) = \sum_{\sim\{z\}} \left(\varphi(Z_\varphi) \prod_{y \in n(\varphi) \setminus \{z\}} \mu_{y \to \varphi}(y) \right),$$

where Z_φ is the set of arguments of the factor φ. If $h \in n(z) \setminus \{\varphi\} = \{\emptyset\}$, e.g., at a variable node of a final individual (the grandchildren z_7, z_8, and z_9 in Figure 1b), the product is defined as 1. The expression $\sum_{\sim\{z\}}$ is adapted from Kschischang et al. [11] and denotes the *not-sum* (i.e., the sum over all variables except z).

Finally, the marginalization, or *termination step*, computes the value of $g_i(z_i) = \sum_{\sim\{z_i\}} g(z)$ as the product of all incoming messages on a variable node z_i. The marginalized $g_i(z_i)$ are equal to $f(z_i, x|\theta^{(t)})$, i.e., proportional to the desired outputs $T_i^{(t)}$ from the E-step in the EM algorithm. We show some example messages and marginalizations of the sum-product algorithm in the following section. We provide a detailed computation and illustration of all possible messages in Appendix A5 and Figure 2 and give a summarized proof of correct convergence of our algorithm in Appendix A6.

We implemented a sum-product algorithm for computing the marginals of an arbitrary pedigree in R [24] and made it available on GitHub (all scripts are available on GitHub, http://github.com/AlexEngelhardt/sumproduct). The code creates 1 factor graph per family, and therein 1 factor node per founder, which contains $\varphi_i(z_i) = f(x_i|z_i) \cdot \mathrm{P}(z_i)$. Furthermore, we create 1 factor per set of parents, which contains the product of all densities of all children (but not the parents):

$$\varphi_j(Z_j) = \Pi_{i \in K_j} f(x_i|z_i) \cdot \mathrm{P}(z_i|z_{\sigma i}, z_{\varphi i}),$$

where Z_j represents all variables within the factor (parents and children), and K_j is the set of children variables connected to φ_j.

Messages from a "large" factor containing parents and many children will be summed over all neighboring variable nodes except the destination variable node. By iteratively exploiting the distributive law, this sum can be efficiently broken down from exponential to linear runtime. For an example based on Figure 1b, see Appendix A7.

Example Messages and Marginalizations of the Sum-Product Algorithm

We illustrate the sum-product algorithm by calculating 2 example messages and 1 example marginalization from the pedigree of Figure 1b.

Firstly, the message $\mu_{z_6 \to \varphi_{789}}(z_6)$ from the variable node z_6 to the factor node φ_{789} equals

$$\mu_{z_6 \to \varphi_{789}}(z_6)\, \mu_{\varphi_6 \to z_6}(z_6).$$

Secondly, the message $\mu_{\varphi_5 \to z_2}(z_2)$ from the factor node φ_5 to the variable node z_2 equals

$$\mu_{\varphi_5 \to z_2}(z_2) = \sum_{z_1} \sum_{z_5} \left(\varphi_5(z_1, z_2, z_5) \cdot \mu_{z_1 \to \varphi_5}(z_1) \mu_{z_5 \to \varphi_5}(z_5) \right).$$

Lastly, we compute the example marginalization at the variable node z_5 as

$$g_5(z_5) = f(z_5, x|\theta^{(t)}) = \mu_{\varphi_5 \to z_5}(z_5) \cdot \mu_{\varphi_{789} \to z_5}(z_5).$$

Then,

$$\begin{aligned} T_5^{(t)} &= \mathrm{P}\left(Z_5 = 1 | x, \theta^{(t)}\right) \\ &= \frac{f\left(z_5 = 1, x | \theta^{(t)}\right)}{f\left(z_5 = 0, x | \theta^{(t)}\right) + f\left(z_5 = 1, x | \theta^{(t)}\right)} \\ &= \frac{\mu_{\varphi_5 \to z_5}(1) \cdot \mu_{\varphi_{789} \to z_5}(1)}{\mu_{\varphi_5 \to z_5}(0) \cdot \mu_{\varphi_{789} \to z_5}(0) + \mu_{\varphi_5 \to z_5}(1) \cdot \mu_{\varphi_{789} \to z_5}(1)}. \end{aligned}$$

This shows that the desired values $T_i^{(t)}$ from the E-step are immediately obtained as soon as all possible messages are computed.

Appendix A5 and Figure 2 show the detailed derivation of all remaining messages.

Convergence of the EM Algorithm

A common drawback among many estimation algorithms is the danger of converging into a local maximum. This also applies to the EM algorithm. However, we will show that the marginal likelihood function (Equation 4) is concave and thus the EM algorithm will always converge to the global maximum of the marginal likelihood, irrespective of the initial parameter choice.

A twice differentiable function is concave if its Hessian matrix is negative semidefinite. We first show that the Hessian of the complete log-likelihood function (Equation 3) is negative semidefinite. Since the variables α and p_1 are separated, $\partial^2/\partial\alpha\partial p_1\, l(\theta; x, z) = 0$, and consequently the Hessian is a diagonal matrix. It suffices to show that both its diagonal entries are zero or negative. We obtain

$$\frac{\partial^2}{\partial p_1^2} l\left(\theta; x, z\right) = -\frac{\sum_{i \in F} z_i}{p_1^2} - \frac{|F| - \sum_{i \in F} z_i}{\left(1 - p_1^2\right)}. \tag{10}$$

$$\frac{\partial^2}{\partial \alpha^2} l\left(\theta; x, z\right) = -\sum_{i=1}^{n} \frac{c_i z_i}{\alpha^2}. \tag{11}$$

Note that in Equation 10, $\sum z_i \leq |F|$; hence, both terms on the right-hand side of this equation are negative or zero. This proves that the complete log-likelihood is concave. Since the exponential is strictly monotonically increasing and $l(\theta; x, z)$ is strictly concave, it follows that $L(\theta; x, z) = \exp(l(\theta; x, z))$ has only 1 unique local and hence global maximum [25]. Finally, the marginal log-likelihood, as the sum $\sum_z l(\theta; x, z)$ of concave functions, is also concave. This means that simpler but faster optimization algorithms such as Nelder-Mead can be used for our model, and there is no need for more complex algorithms such as stochastic or constrained optimizers (e.g., L-BFGS-B [26] or simulated annealing [27]).

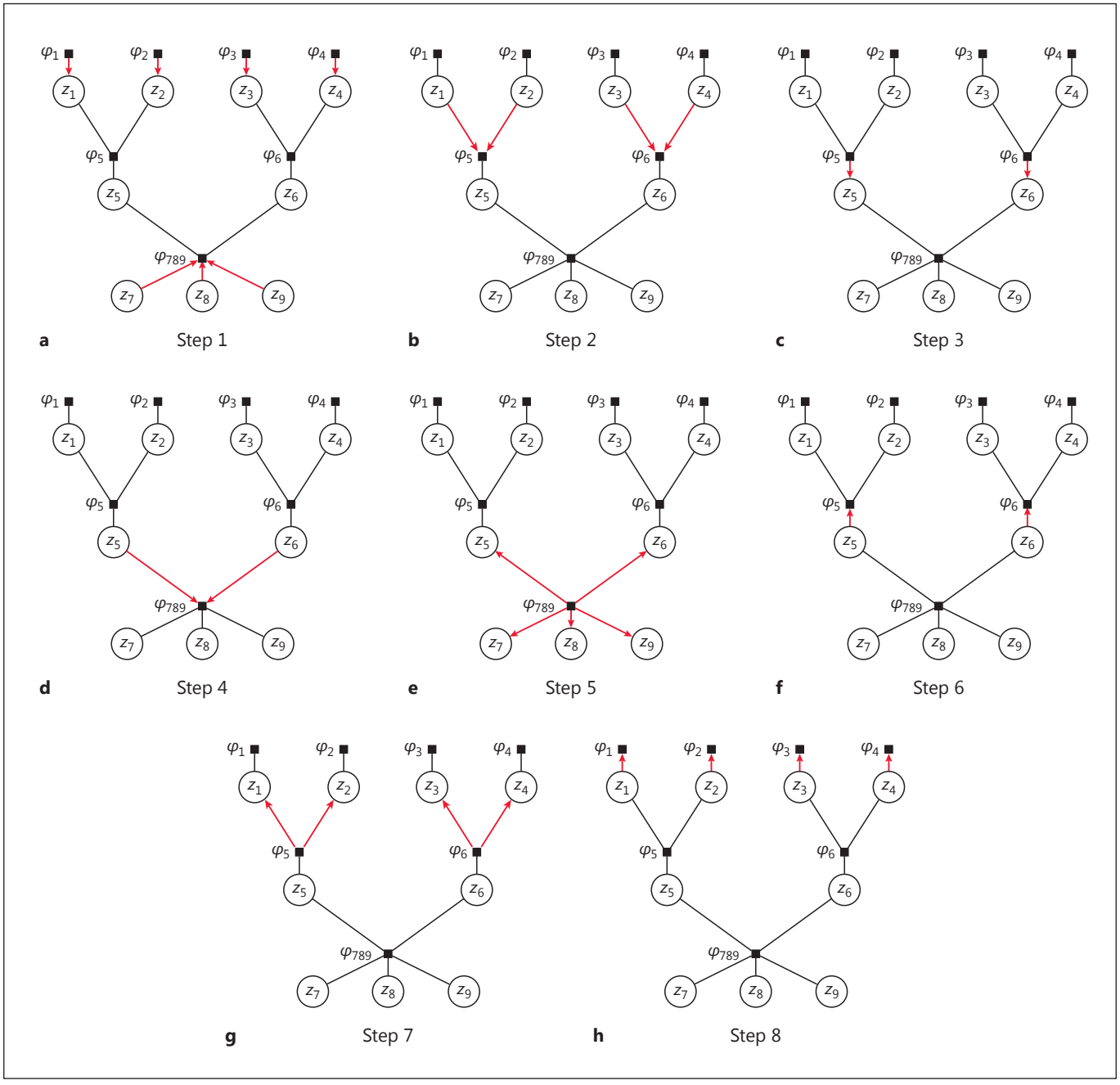

Fig. 2. Message passing during the sum-product algorithm. The messages are computed iteratively. Once all but 1 incoming messages are calculated for a given node, a message is calculated and sent across the remaining edge. Note that by Kschischang et al. [11], the actual order in which the messages are calculated is irrelevant for the result.

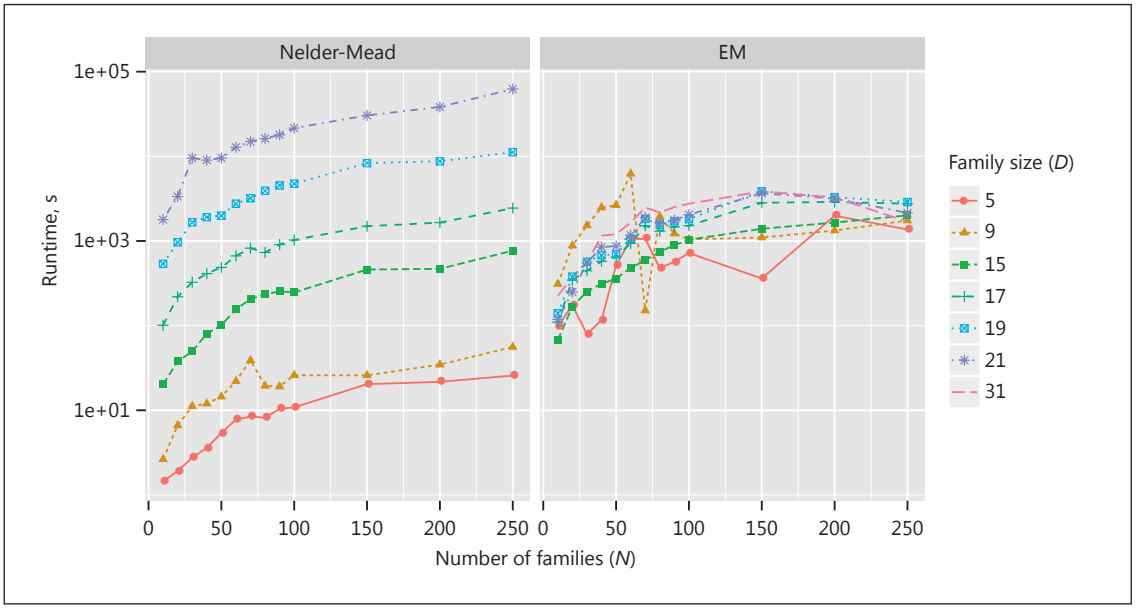

Fig. 3. Runtime comparison of Nelder-Mead optimization (left) and the EM algorithm (right). Shown is the runtime in seconds on the *y* axis (log scale) versus the number of families on the *x* axis. The effect of an increasing family size *D* is negligible with the EM algorithm, but exponential with the Nelder-Mead optimization. For the largest families of $D = 31$, only the EM algorithm could be run in a feasible time. The Nelder-Mead optimization would have taken around 2.7 years.

Application: Estimating the Probability of Being a Risk Family

Applying the sum-product algorithm directly implies a straightforward method to compute the probability of being a risk family.

After we have estimated p_1 and α, we can estimate the probability that this family carries the risk factor for a new pedigree (or a pedigree from the original study, i.e., the training data). We define a *risk family* as a family in which at least 1 member is carrying the risk factor (i.e., $z_i = 1$ for at least one i). This is exactly 1 minus the probability that *no* family member carries the risk factor. If we restrict the data x and Z to just the family in question, and define the event R as "The family is a risk family," we can then compute

$$\mathrm{P}(R) = 1 - \mathrm{P}(Z = 0 \mid x, \hat{\theta}) \tag{12}$$

$$= 1 - \prod_{i \in F} \mathrm{P}\left(Z_i = 0 \mid x, \hat{\theta}\right). \tag{13}$$

$$= 1 - \prod_{i \in F} \left(1 - T_i^{(t)}\right). \tag{14}$$

The step from Equation 12 to Equation 13 is possible because the probability that *no family member* carries the risk factor equals the probability that *no founder* carries the risk factor, since the former is true if and only if the latter is true. Then, we can split up the joint probability that no founder carries the risk factor into the individual probabilities $\mathrm{P}(Z_i = 0 \mid x, \hat{\theta})$. This step is possible because we only consider the founders, and their risk probabilities are independent of any other z_i.

Since $\mathrm{P}(Z_i = 0 \mid x, \hat{\theta})$ equals $1 - T_i^{(t)}$ by definition (Equation 5), we can simply run the E-step of the EM algorithm once on the new family to obtain these values. We then multiply over only those $T_i^{(t)}$ where $i \in F$ and obtain $\mathrm{P}(R)$, an estimator for the familial CRC risk.

Results

Simulation Study

We performed an in silico experiment by simulating data sets with a given p_H, p_1, and α and with a varying number of families (N) and pedigree size (D). The risk status z_i for each founder was randomly sampled with the probability $\mathrm{P}(z_i = 1) = p_1$, the statuses for all nonfounders were sampled according to Equation 1. The age at onset of CRC was then simulated according to a Weibull distribution with the best fitting parameters according to Rieger and Mansmann [9], $\lambda = 0.0058$ and $k = 4$, and a risk increase for males of $\beta = 2$:

$$f(t_i \mid z_i) = h(t_i) \cdot S(t_i) = [k\lambda^k t_i^{k-1} \alpha^{z_i} \beta^{m_i}] \cdot \exp(-(t_i\lambda)^k \alpha^{z_i} \beta^{m_i}).$$

We then simulated a censoring age u_i from the following Gaussian distribution: $u_i \sim \mathcal{N}(125, 100)$. The rather optimistic mean censoring age of 125 years was chosen to keep the ratio of censored subjects below 66%, since a higher

Table 1. Runtime ratio (Nelder-Mead over EM algorithm) over different family sizes D and different number of families N

	Family sizes (D)					
	5	9	15	17	19	21
Number of families (N)						
50	0.01	0.01	0.28	0.75	2.85	10.97
100	0.02	0.02	0.24	0.68	2.62	10.54
150	0.06	0.02	0.33	0.53	2.15	8.33
200	0.01	0.03	0.28	0.57	2.65	12.05
250	0.02	0.03	0.38	0.89	3.89	29.05

The EM algorithm is faster for pedigrees of size 19 and above, regardless of the number of families in the data set.

Table 2. Five-point summary and mean values for the parameter estimates of the Nelder-Mead optimization (N-M) and the EM algorithm, based on 100 simulated data sets

	Minimum	1st quartile	Median	Mean	3rd quartile	Maximum
$\hat{p}_1$, N-M	0.1478	0.1740	0.1913	0.1929	0.2078	0.2864
$\hat{p}_1$, EM	0.1476	0.1739	0.1912	0.1929	0.2078	0.2865
$\hat{\alpha}$, N-M	2.888	4.036	4.347	4.310	4.656	5.297
$\hat{\alpha}$, EM	2.890	4.037	4.345	4.310	4.660	5.287

The simulation parameters were $p_1 = 0.2$ and $\alpha = 4$.

censoring rate would just necessitate a larger simulated data set to reach the same stability. Each subject's censoring indicator c_i was then set to 1 if $t_i < u_i$ and 0 otherwise. A value of 0 therefore indicates a censored observation. If a subject is censored, t_i was replaced by u_i, the age at censoring.

Runtime Improvement

We simulated data sets with different pedigree sizes to investigate the threshold pedigree size from which the EM algorithm is faster than a Nelder-Mead optimization. Figure 3 and Table 1 show the runtime of the Nelder-Mead optimization versus the EM algorithm for different data sizes and pedigree sizes. The pedigrees used were:

- $D = 5$: two parents with 3 children
- $D = 9$: four grandparents, 2 parents, 3 children (Fig. 1a)
- $D = 15$: four generations with only 1 final individual
- $D = 17$: the same pedigree as for $D = 15$, with 1 additional parent pair for 1 founder
- $D = 19$: one more parent pair in the same generation as for $D = 17$
- $D = 21$: one more parent pair in the same generation as for $D = 19$
- $D = 31$: five generations with 1 final individual.

The results suggest that using the EM algorithm is advantageous as soon as *some* families in the data set are large (more than around 17 members). For $D = 31$, we extrapolated the runtime for the Nelder-Mead optimization to about 2.7 years, according to a log-linear regression of runtime against family size. This shows the dramatic improvement of using the EM algorithm as family sizes grow larger. A more advanced EM algorithm could even split the data into small and large pedigrees, and in the E-step use the sum-product algorithm for the larger families, and a "brute force" marginalization for smaller families.

Our Algorithm Recovers True Parameters

We simulated 100 replicated data sets of 500 families of 9 persons as in Figure 1a. In each replication, we chose $p_1 = 0.2$ and $\alpha = 4$ as the parameters and let the Nelder-Mead optimization and the EM algorithm estimate the parameters to investigate their level of agreement. Figure 4 shows scatterplots and Bland-Altman plots to compare the 2 methods and finds a strong agreement between them. Table 2 shows summary statistics on both methods' parameter estimates in the 100 replications.

The Imputation of Noninformative Parents Works

When family members were randomly removed from the data set after simulation, the imputation procedure did not affect the results – both algorithms still converged to the correct result after imputing took place. The simulation and estimation were performed as in Figure 4, but 20% of the family members were randomly removed beforehand. The preprocessing then performed an imputation of missing members. After our imputing procedure, both algorithms still recover the true parameters (data not shown; reproducible scripts available on GitHub).

Application: Estimating the Probability of Being a Risk Family

We computed the posteriori probability of being a CRC risk family for a simulated data set of 1,000 pedigrees with 9 persons each, according to Equation 14. The resulting ROC curve is shown in Figure 5. The AUC of 0.74 shows that risk families can be identified with a satisfyingly good rate.

Real Data

We applied our algorithm on a study exploring the familial CRC risk [28]. In this study, patients diagnosed with CRC in the Munich region were recruited. Subse-

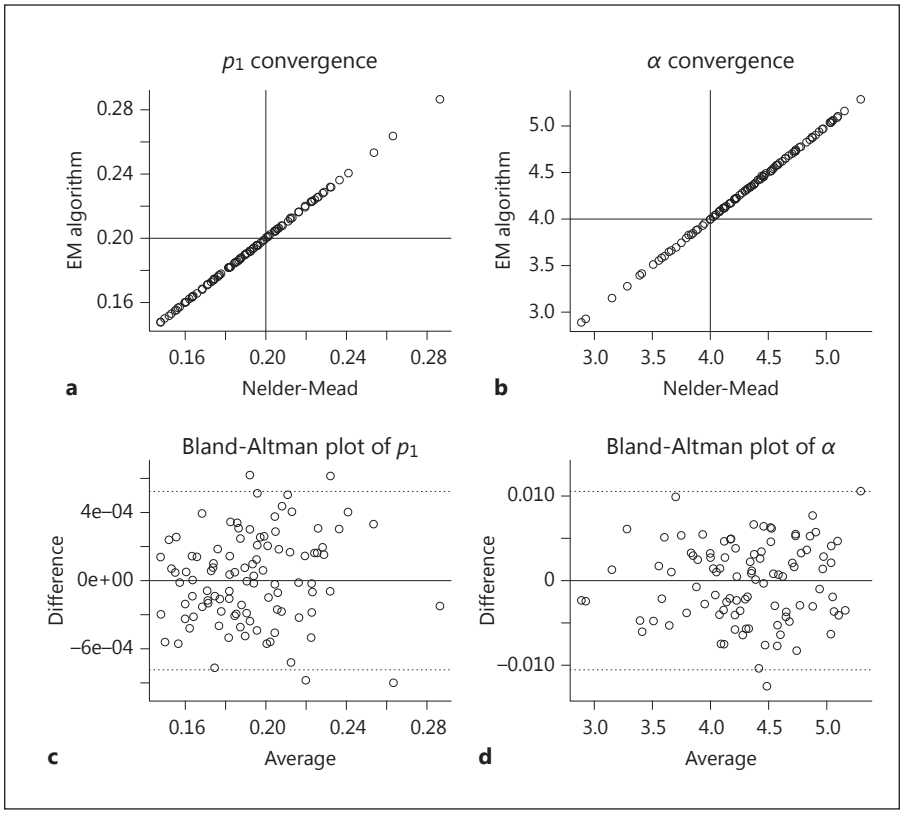

Fig. 4. Convergence of 100 replications of simulating and estimating data sets. Each replication used random uniform distributed starting values for $\theta = (p_1, \alpha)$. **a**, **b** The final parameter estimates for the Nelder-Mead optimization (x axis) and the EM algorithm (y axis). **c**, **d** Bland-Altman plots where the x axis shows the average of the parameter estimates of the 2 methods and the y axis shows their difference. Horizontal dashed lines are drawn at ±2 standard deviations of the difference. We see that both estimation methods agree with each other and converge close to the correct result of $p_1 = 0.2$ and $\alpha = 4$ regardless of starting values.

quently, each of these *index patients* was given a questionnaire with a blank pedigree to fill out data about all known relatives. With this obtained pedigree, the Munich Cancer Registry (MCR) [29] was consulted via an anonymized record-linkage procedure for any CRC diagnoses of the index patient's relatives [30]. The result was a pedigree of family data and CRC diagnoses per index patient. The study was active from September 2012 until June 2014 and resulted in a data set of 611 families, of which 181 were just individuals (a "pedigree" with only 1 person).

In the real data set, pedigrees were not always recorded in a directly useable manner. For observations with only 1 available parent, we imputed the missing parent as noninformative ($c_i = 0$, $t_i = 0$ and with the appropriate gender m_i). In cases where a family consisted only of siblings, we imputed both parents as noninformative observations to indicate the relatedness of the siblings. The remaining analysis was analogous to the in silico study described in the Methods.

We estimated a prevalence of $p_1 = 0.901$ and a risk factor increase of $\alpha = 5.723$. We then performed a 100-fold bootstrapping of families to obtain bootstrap standard er-

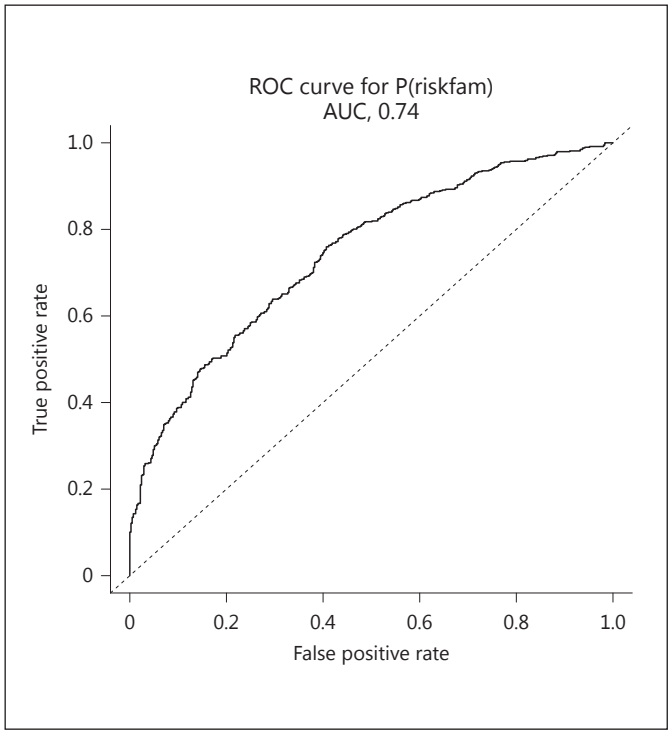

Fig. 5. ROC curve for the probability of being a risk family, based on 1,000 simulated families with 9 persons (cf. Fig. 1a).

rors. The values obtained were 0.0002 and 0.0229, respectively.

The rather high a priori probability may stem from a bias in the data set, since the collection procedure preferably selected patients and families that are already exposed to risk. A more detailed discussion on the results is given in Rieger and Mansmann [9].

The data set contained 3 families with at least 20 members. For the largest family of 23 persons, using the EM algorithm with the sum-product algorithm instead of a Nelder-Mead optimization showed a reduction of the runtime to 16%.

Discussion

Directly translating mathematical formulas into computer code often results in a formally correct but slow solution. In our case, a Nelder-Mead optimization would need multiple evaluations of the marginalized likelihood, which is unfeasible for larger families. An approach based on *peeling* [31–33], however, could have been used to reduce the time for evaluating the likelihood. The disadvantage of the Elston-Stewart peeling algorithm is that as soon as the pedigree contains loops, its runtime increases exponentially with the *cutset* (i.e., the number of members that have to be considered jointly) [13]. The EM algorithm coupled with the sum-product algorithm can be extended to pedigrees with loops by applying the "loopy belief propagation" procedure [11]. Furthermore, peeling algorithms need to find an optimal peeling order for each pedigree, a problem that still has no gold standard solution today [33]. The sum-product algorithm, on the other hand, directly implies an efficient order of computing the messages, and thus elegantly circumvents this problem. Thompson and Shaw [34] showed that the EM algorithm is a viable alternative to the peeling algorithm in polygenic models. Our approach differs from this in that we skip the detection of responsible genes and instead focus on estimating a family's probability of carrying an (unspecified) CRC risk factor.

The EM algorithm with an approximative Monte Carlo implementation of the E-step has previously been used on pedigrees for segregation analysis [35]. We saw that the EM algorithm in our setting relied on a marginalization over all possible risk vectors for each pedigree. This problem of calculating marginal densities in hierarchical data such as pedigrees has usually been tackled by Markov Chain Monte Carlo (MCMC) simulations and related sampling algorithms [36, 37]. Due to the random sampling, these methods all yield only approximative solutions and may take a long time to reach stable results. We instead use the sum-product algorithm [11] within the EM algorithm to solve the necessary marginalization in the E-step in linear instead of exponential time. This provides a fast and exact solution and allows maximum-likelihood estimation with pedigrees of arbitrary size that, furthermore, is not dependent on an arbitrarily chosen number of MCMC simulations.

In contrast to complex segregation analysis, our approach is less specific. In particular, we only model an unspecific risk "component," not necessarily a gene or multiple genes, which is passed down to offspring with a certain predefined probability. We chose this phenotype-based approach since the causes for familial occurrence of CRC are currently unknown. Our generic model is therefore not fully in line with mendelian transmission models. Only under the assumption of an autosomal dominant risk factor that is rare, so that an affected individual can be assumed to have the genotype Aa instead of AA, is a constant inheritance probability of $p_H = 0.5$ justifiable. As an advantage, environmental risk factors such as nutrition, lifestyle, or place of residence can be modeled by choosing $p_H = 1$. If one wants to account for a small probability of changing the place of residence, or a probability for children to not adopt the parents' lifestyle, inheritance probabilities less than 1 can be used as well. However, if a large enough data set were available, it should also be possible to estimate p_H robustly enough.

One limitation of the real data set in this study is the relatively small sample size. With around 600 families, the data set was not large enough to obtain stable estimates. However, since the focus of this study is methodological, the data set can still be used to show the runtime improvement of our algorithm. The moderate gain in speed in the application to the real data set can be explained by 2 facts. First, the multiplicative time constant for the sum-product algorithm is larger than that for the calculation of the marginal likelihood, since the message passing needs more elementary operations than simple summation over all possible configurations of the family members (yet, as demonstrated in Figure 3, this disadvantage is soon compensated by the exponential increase in runtime of the Nelder-Mead method). Second, the families in the real data set are mostly small (median size: 6 members), with the exception of a few larger families (maximum size: 23). Thus, the full power of the EM algorithm will unfold in applications where more, larger families are investigated. The size of the families in such studies is likely to grow in the future, with the availabil-

ity of more longitudinal data. Moreover, note that a single, large family of more than, say, 25 members, will prohibit the straightforward application of the marginalization approach.

It should also be noted that for small families, the linear runtime of the sum-product algorithm is *slower* than the exponential runtime of the marginalization, due to the overhead in setting up the factor graph (Fig. 3). In our analysis, the sum-product algorithm had significant benefits only as soon as a family consisted of more than 17 members.

It is possible to estimate further parameters in this model, such as the shape parameter k. However, in our case, these parameters were external epidemiological information, and thus already available from more robust previous studies. Moreover, if additional parameters are estimated, it should then be reevaluated whether the Hessian matrix is still negative semidefinite (i.e., whether the log-likelihood is still concave and unimodal).

Our algorithm can of course also be extended to other models. For example, other penetrance models can be used, such as a "time shift" model, where the hazard rate is not multiplied by a factor α, but instead shifted horizontally, by adding a "risk advancement" of a specific number of years [38]. It is also possible to use different response distributions besides a Weibull distribution. Furthermore, it is possible to extend our method to model true mendelian transmission, either by having $Z_i \in \{0, 1, 2\}$ model the number of affected alleles, or by specifying *two* latent variables, $Z_{\male} \in \{0, 1\}$ and $Z_{\female} \in \{0, 1\}$ per individual, 1 for each allele. One would then need a transmission matrix to specify the probability of each possible outcome for offspring given the statuses of both parents. Ghahramani [39] provides a tutorial on how to extend a Bayesian network such as a pedigree to deal with multiple latent variables.

When the number of possible genotypes (i.e., the number of possible values for Z_i) increases, both the EM algorithm and the Nelder-Mead optimization suffer from exponential runtime increase. This is the case for example when one works with multilocus genotypes. The runtime of the EM algorithm is only linear respective to the family size. However, since the genetic mechanism in our case is unknown, a dichotomized latent variable served our purpose well.

Faster algorithms such as the one presented in this paper also open the door for new analyses that were previously unfeasible. With the sum-product algorithm, we can now conduct large-scale simulation studies for power and sample size determination and extract further information such as bootstrap confidence intervals from the data.

Conclusion

In this paper, we developed an efficient algorithm for maximum likelihood estimation where the observations in the data are partially dependent. The rising size and complexity of modern data sets make it necessary to revisit popular algorithms for data analysis and develop improvements in their efficiency. Here, we considered clinical data in the form of pedigrees, where the presence of latent and inheritable genetic risk factors greatly complicated the analysis procedure. In our case, a standard implementation of the EM algorithm results in a runtime that is still exponential regarding the family sizes, due to the inheritability of the latent variables.

However, by considering the pedigree as a Bayesian network, and then factorizing it with a factor graph and reformulating the E-step by employing a sum-product algorithm, the runtime could be reduced to linear in terms of the family size. Similar to the peeling algorithm [31], the sum-product algorithm in essence breaks down complex pedigrees into *nuclear* families, each consisting of father, mother, and all children. The number of children does not cause exponential growth of runtime because the summation is again broken down between each child.

In conclusion, the combination of an EM algorithm with the sum-product algorithm removes the restrictions that exponential runtime imposes on the analysis due to large families and opens the door for maximum likelihood estimation on large pedigrees.

As a next step, we plan to make this risk prediction algorithm available as a web interface, so that clinicians can conveniently enter a family's pedigree. It will then aid in assessing familial CRC risk of individual patients.

Acknowledgments

The work is supported by the Federal Ministry for Families, Elderly, Women, and Youth (BMFSFJ, FKZ: 3911 401 001). A.E. and A.T. were supported by a BMBF e:Bio grant.

Statement of Ethics

The study was approved by the ethical review board of the Medical Faculty of the University of Munich (LMU).

Disclosure Statement

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Appendix

This section contains extended derivations of the equations used in this paper.

A1 The Complete Log-Likelihood (Equation 3)
The complete likelihood (Equation 2) becomes

$$
\begin{aligned}
&L(\theta; x, z) = f(x, z) = \mathrm{P}(z) \cdot f(x|z) \\
&= \prod_{i\in F} \mathrm{P}(z_i) \cdot \prod_{i\notin F} \mathrm{P}\left(z_i \mid z_{\mathscr{J}i}, z_{\mathscr{P}i}\right) \cdot \prod_{i=1}^{n} f\left(x_i \mid z_i\right) \\
&= \prod_{i\in F} p_i^{z_i}\left(1-p_1\right)^{1-z_i} \cdot \prod_{i\notin F} \tilde{p}_i^{z_i}\left(1-\tilde{p}_i\right)^{1-z_i} \cdot \prod_{i=1}^{n}\left[k\lambda^k t_i^{k-1}\alpha^{z_i}\beta^{m_i}\right]^{c_i} \exp\left(-\left(t_i\lambda\right)^k \alpha^{z_i}\beta^{m_i}\right).
\end{aligned}
$$

Note that the relevant part for α includes all persons, and the part for p_1 only includes the founders. The product over all $i \notin F$ is independent of $\theta = (\alpha, p_1)$ and thus becomes irrelevant in the estimation procedure.

The log-likelihood is then

$$
\begin{aligned}
l(\theta; x, z) &= \left(\sum_{i\in F} z_i\right)\log\, p_1 + \left(|F| - \sum_{i\in F} z_i\right)\log\left(1-p_1\right) \\
&\quad + \sum_{i\notin F}\left[z_i\, \log\, \tilde{p}_i + \left(1-z_i\right)\log\left(1-\tilde{p}_i\right)\right] \\
&\quad + \sum_{i=1}^{n} c_i \cdot \left[\, \log k + k \log \lambda + (k-1)\log t_i + z_i\, \log \alpha + m_i\, \log \beta\,\right] \\
&\quad - \left(t_i\lambda\right)^k \alpha^{z_i}\beta^{m_i} \\
&\quad \text{respective to } \alpha \text{ and } p_1\text{, this reduces to} \\
&= \text{const} + \left(\sum_{i\in F} z_i\right)\log\, p_1 + \left(|F| - \sum_{i\in F} z_i\right)\log\left(1-p_1\right) \\
&\quad + \sum_{i=1}^{n} c_i z_i\, \log \alpha - \left(t_i\lambda\right)^k \alpha^{z_i}\beta^{m_i}\,.
\end{aligned}
$$

A2 Derivation of Equation 5
The expected value $\mathrm{E}_{Z|x,\,\theta^{(t)}}(Z_i)$ is equal to the marginalized expected value $\mathrm{E}_{z_i|\,x,\,\theta^{(t)}}(Z_i)$ because of the following marginalization steps from Z to Z_i:

$$
\begin{aligned}
T_i^{(t)} &= \mathrm{E}_{Z|x,\theta^{(t)}}\left(Z_i\right) = \sum_z z_i\, \mathrm{P}\left(z \mid x, \theta^{(t)}\right) \\
&= \sum_{z_1}\cdots\sum_{z_n} z_i\, \mathrm{P}\left(z_1, z_2, \ldots, z_n \mid x, \theta^{(t)}\right) \\
&= \sum_{z_1} z_i \sum_{z_1}\cdots\sum_{z_{i-1}}\sum_{z_{i+1}}\cdots \underbrace{\sum_{z_n} \mathrm{P}\left(z_1, z_2, \ldots, z_n \mid x, \theta^{(t)}\right)}_{=\,\mathrm{P}\left(z_i \mid x,\,\theta^{(t)}\right)} \\
&= \mathrm{E}_{Z_i|\,x,\,\theta^{(t)}}\left(z_i\right) \\
&= \mathrm{P}\left(Z_i = 1 \mid x, \theta^{(t)}\right).
\end{aligned}
$$

A3 Efficient Computation of Likelihoods and Marginalizations
Since we assume independence between families, the complete likelihood $L(\theta; x, z)$ can be factorized into a product of N *family likelihoods* [13, Equation 1.4]. If one denotes the families in a data set by $I = 1, \ldots, N$ and their members by $d = 1, \ldots, D_I$, the index i becomes a combined index I, d from the family index and the member index. We can further define the sub-vectors x_I and z_I to be the observed and latent data for only family I. Note that using this notation, e.g., z_1 is now a vector of risk statuses for family 1, and not the risk status of just the first observation.

Equation 2 then becomes

$$
\begin{aligned}
L(\theta; x, z) &= \prod_{I=1}^{N} f\left(x_I, z_I\right) = \prod_{I=1}^{N} \mathrm{P}(z_I) \cdot f\left(x_I \mid z_I\right) \\
&= \prod_{I=1}^{N}\left\{\prod_{d\in F_I} \mathrm{P}\left(z_{I,d}\right) \cdot \prod_{d\notin F_I} \mathrm{P}\left(z_i, d \mid z_{\mathscr{J}I,d}, z_{\mathscr{P}I,d}\right) \cdot \prod_{d=1}^{D_I} f\left(x_{I,d} \mid z_{I,d}\right)\right\} \\
&= \prod_{I=1}^{N}\left\{\begin{array}{l}\displaystyle\prod_{d\in F_I} p_1^{z_{I,d}}\left(1-p_1\right)^{1-z_{I,d}} \cdot \prod_{d\notin F_I} \tilde{p}_{I,d}^{\,z_{I,d}}\left(1-\tilde{p}_{I,d}\right)^{1-z_{I,d}} \\ \displaystyle\cdot \prod_{d=1}^{D_I}\left[k\lambda^k t_{I,d}^{k-1}\alpha^{z_{I,d}}\beta^{m_{I,d}}\right]^{c_{I,d}} \exp\left(-\left(t_{I,d}\lambda\right)^k \alpha^{z_{I,d}}\beta^{m_{I,d}}\right)\end{array}\right\}.
\end{aligned}
$$

This notation now allows for computationally elegant marginalizations.

Summation in the Marginalized Likelihood in Equation 4
Keeping the notation for families and family members, we can rewrite Equation 4 into a more efficient marginalization:

$$
\begin{aligned}
L(\theta; x) &= \sum_z L(\theta; x, z) \\
&= \sum_z \prod_{I=1}^{N} L\left(\theta; x_I, z_I\right) \\
&= \sum_{z_1}\cdots\sum_{z_N} L\left(\theta; x_1, z_1 \cdots\right) L(\theta; x_N, z_N) \\
&= \sum_{z_1} L\left(\theta; x_1, z_1\right)\cdots\sum_{z_N} L\left(\theta; x_N, z_N\right) \\
&= \prod_{I=1}^{N}\sum_{z_I} L\left(\theta; x_I, z_I\right) \\
&= \prod_{I=1}^{N}\sum_{z_I} f\left(x_I \mid z_I, \theta\right)\mathrm{P}\left(z_I\right).
\end{aligned}
$$

This way, we do not sum over 2^n possible values for z, but instead

$$\prod_{I=1}^{N} 2^{D_I}$$

values, where D_I is the number of family members in family I.

Marginalizing the Risk Carrier Probability $T_i^{(t)}$ from Equation 5
The marginalization when computing $T_i^{(t)} \equiv T_{I,d}^{(t)}$ can happen more efficiently, summing only over 1 specific family. Since a $Z_{I,d}$ is independent of all x outside of its respective family's x_I, we can replace the condition on x by x_I:

$$
\begin{aligned}
T_{I,d}^{(t)} &= \mathrm{E}_{Z|\,x,\,\theta^{(t)}}\left(Z_{I,d}\right) \\
&= \mathrm{E}_{Z_{I,d}|\,x,\,\theta^{(t)}}\left(Z_{I,d}\right) \\
&= \mathrm{E}_{Z_{I,d}|\,x_I,\,\theta^{(t)}}\left(Z_{I,d}\right) \qquad \text{(Appendix A2)} \\
&= \mathrm{P}\left(Z_{I,d} = 1 \mid x_I, \theta^{(t)}\right) \\
&= \sum_{z_I} z_{I,d}\, \mathrm{P}\left(z_I \mid x_I, \theta^{(t)}\right)
\end{aligned}
$$

Therefore, the marginalizations of the factor graph can be efficiently computed for each family separately and combined at the end.

A4 Factor Functions φ_j for Equation 9

Since $g(z) \equiv f(z, x)$, the factors φ_j describe the following functions:

$$\varphi_1(z_1) = f(x_1|z_1)\ \mathrm{P}(z_1)$$
$$\varphi_2(z_2) = f(x_2|z_2)\ \mathrm{P}(z_2)$$
$$\varphi_3(z_3) = f(x_3|z_3)\ \mathrm{P}(z_3)$$
$$\varphi_4(z_4) = f(x_4|z_4)\ \mathrm{P}(z_4)$$
$$\varphi_5(z_1, z_2, z_5) = f(x_5|z_5)\ \mathrm{P}(z_5|z_1, z_2)$$
$$\varphi_6(z_3, z_4, z_6) = f(x_6|z_6)\ \mathrm{P}(z_6|z_3, z_4)$$
$$\varphi_{789}(z_5, z_6, z_7, z_8, z_9) = f(x_7|z_7)\ \mathrm{P}(z_7|z_5, z_6) \cdot f(x_8|z_8)\ \mathrm{P}(z_8|z_5, z_6) \cdot f(x_9|z_9)\ \mathrm{P}(z_9|z_5, z_6).$$

A5 All Messages of the Sum-Product Algorithm in Figure 1b

Here we show a detailed derivation of all messages for the sum-product algorithm in Figure 1b. The messages are computed iteratively in 8 steps. Figure 2 illustrates the messages computed in each step.

Step 1

In step 1, only the "outermost" messages can be computed, since only they are not depending on any other messages. In particular, the following messages are computed:

$$\mu_{\varphi_1 \to z_1}(z_1) = \varphi_1(z_1)$$
$$\mu_{\varphi_2 \to z_2}(z_2) = \varphi_2(z_2)$$
$$\mu_{\varphi_3 \to z_3}(z_3) = \varphi_3(z_3)$$
$$\mu_{\varphi_4 \to z_4}(z_4) = \varphi_4(z_4)$$
$$\mu_{z_7 \to \varphi_{789}}(z_7) = 1$$
$$\mu_{z_8 \to \varphi_{789}}(z_8) = 1$$
$$\mu_{z_9 \to \varphi_{789}}(z_9) = 1.$$

The definitions of each factor, e.g., $\varphi_1(z_1)$, are available in Appendix A4.

Step 2

In step 2 (see Fig. 2b), we have all information necessary to compute the following 4 messages:

$$\mu_{z_1 \to \varphi_5}(z_1) = \mu_{\varphi_1 \to z_1}(z_1)$$
$$\mu_{z_2 \to \varphi_5}(z_2) = \mu_{\varphi_2 \to z_2}(z_2)$$
$$\mu_{z_3 \to \varphi_6}(z_3) = \mu_{\varphi_3 \to z_3}(z_3)$$
$$\mu_{z_4 \to \varphi_6}(z_4) = \mu_{\varphi_4 \to z_4}(z_4).$$

Step 3

Now we can compute the messages *to* the parents z_5 and z_6:

$$\mu_{\varphi_5 \to z_5}(z_5) = \sum_{z_1} \sum_{z_2} \left(\varphi_5(z_1, z_2, z_5) \cdot \mu_{z_1 \to \varphi_5}(z_1) \mu_{z_2 \to \varphi_5}(z_2) \right)$$
$$\mu_{\varphi_6 \to z_6}(z_6) = \sum_{z_3} \sum_{z_4} \left(\varphi_6(z_3, z_4, z_6) \cdot \mu_{z_3 \to \varphi_6}(z_3) \mu_{z_4 \to \varphi_6}(z_4) \right).$$

Step 4

In step 4, we can only obtain 2 new messages:

$$\mu_{z_5 \to \varphi_{789}}(z_5) = \mu_{\varphi_5 \to z_5}(z_5)$$
$$\mu_{z_6 \to \varphi_{789}}(z_6) = \mu_{\varphi_6 \to z_6}(z_6).$$

Step 5

In step 5, we now have all incoming messages to ϕ_{789} available, which means we can compute all outgoing messages from φ_{789}:

$$\mu_{\varphi_{789} \to z_5}(z_5) = \sum_{z_6} \sum_{z_7} \sum_{z_8} \sum_{z_9} \left(\varphi_{789}(z_5, z_6, z_7, z_8, z_9) \cdot \mu_{z_6 \to \varphi_{789}}(z_6) \mu_{z_7 \to \varphi_{789}}(z_7) \mu_{z_8 \to \varphi_{789}}(z_8) \mu_{z_9 \to \varphi_{789}}(z_9) \right)$$

$$\mu_{\varphi_{789} \to z_5}(z_6) = \sum_{z_5} \sum_{z_7} \sum_{z_8} \sum_{z_9} \left(\varphi_{789}(z_5, z_6, z_7, z_8, z_9) \cdot \mu_{z_6 \to \varphi_{789}}(z_6) \mu_{z_7 \to \varphi_{789}}(z_7) \mu_{z_8 \to \varphi_{789}}(z_8) \mu_{z_9 \to \varphi_{789}}(z_9) \right)$$

$$\mu_{\varphi_{789} \to z_5}(z_7) = \sum_{z_5} \sum_{z_6} \sum_{z_8} \sum_{z_9} \left(\varphi_{789}(z_5, z_6, z_7, z_8, z_9) \cdot \mu_{z_5 \to \varphi_{789}}(z_5) \mu_{z_6 \to \varphi_{789}}(z_6) \mu_{z_8 \to \varphi_{789}}(z_8) \mu_{z_9 \to \varphi_{789}}(z_9) \right)$$

$$\mu_{\varphi_{789} \to z_8}(z_8) = \sum_{z_5} \sum_{z_6} \sum_{z_7} \sum_{z_9} \left(\varphi_{789}(z_5, z_6, z_7, z_8, z_9) \cdot \mu_{z_5 \to \varphi_{789}}(z_5) \mu_{z_6 \to \varphi_{789}}(z_6) \mu_{z_7 \to \varphi_{789}}(z_7) \mu_{z_9 \to \varphi_{789}}(z_9) \right)$$

$$\mu_{\varphi_{789} \to z_9}(z_9) = \sum_{z_5} \sum_{z_6} \sum_{z_7} \sum_{z_8} \left(\varphi_{789}(z_5, z_6, z_7, z_8, z_9) \cdot \mu_{z_5 \to \varphi_{789}}(z_5) \mu_{z_6 \to \varphi_{789}}(z_6) \mu_{z_7 \to \varphi_{789}}(z_7) \mu_{z_8 \to \varphi_{789}}(z_8) \right).$$

Step 6

We can now start computing the *upward* messages from φ_{789}:

$$\mu_{z_5 \to \varphi_5}(z_5) = \mu_{\varphi_{789} \to z_5}(z_5)$$
$$\mu_{z_6 \to \varphi_6}(z_6) = \mu_{\varphi_{789} \to z_6}(z_6).$$

Step 7

Here, we compute the upward messages from the factors φ_5 and φ_6:

$$\mu_{\varphi_5 \to z_1}(z_1) = \sum_{z_2} \sum_{z_5} \left(\varphi_5(z_1, z_2, z_5) \cdot \mu_{z_2 \to \varphi_5}(z_2) \mu_{z_5 \to \varphi_5}(z_5) \right)$$
$$\mu_{\varphi_5 \to z_2}(z_2) = \sum_{z_1} \sum_{z_5} \left(\varphi_5(z_1, z_2, z_5) \cdot \mu_{z_1 \to \varphi_5}(z_1) \mu_{z_5 \to \varphi_5}(z_5) \right)$$
$$\mu_{\varphi_6 \to z_3}(z_3) = \sum_{z_4} \sum_{z_6} \left(\varphi_6(z_3, z_4, z_6) \cdot \mu_{z_4 \to \varphi_6}(z_4) \mu_{z_6 \to \varphi_6}(z_6) \right)$$
$$\mu_{\varphi_6 \to z_4}(z_4) = \sum_{z_3} \sum_{z_6} \left(\varphi_6(z_3, z_4, z_6) \cdot \mu_{z_3 \to \varphi_6}(z_3) \mu_{z_6 \to \varphi_6}(z_6) \right).$$

Step 8

In the last step, we can compute the messages from the founder variables to their respective factors. For the marginalization step in the end, we do not need these messages, strictly speaking. For the sake of completeness, however, we show their computation nonetheless:

$$\mu_{z_1 \to \varphi_1}(z_1) = \mu_{\varphi_5 \to z_1}(z_1)$$
$$\mu_{z_2 \to \varphi_2}(z_2) = \mu_{\varphi_5 \to z_2}(z_2)$$
$$\mu_{z_3 \to \varphi_3}(z_3) = \mu_{\varphi_6 \to z_3}(z_3)$$
$$\mu_{z_4 \to \varphi_4}(z_4) = \mu_{\varphi_6 \to z_4}(z_4).$$

A6 Proof of Convergence of the EM Algorithm

To summarize, we showed that our EM algorithm converges correctly with the following 3 steps:

(a) The EM algorithm converges to a local maximum of the marginal likelihood. By concavity of the marginal likelihood (see

Methods), this is also the global maximum, and thus, the maximum likelihood estimator.

(b) The E-step in the EM algorithm for our problem reduces to computing the marginalized expected values $T_i^{(t)} = \mathrm{E}_{z_i|x,\,\theta^{(t)}}(Z_i)$.

(c) The sum-product algorithm delivers the marginalized densities $T_i^{(t)} = \mathrm{P}(Z_i = 1|x, \theta^{(t)})$.

(a) is clear from Dempster et al. [16], and (c) follows from Kschischang et al. [11]. (b) can be shown as follows:

By definition of the EM algorithm, the E-step consists of computing $Q(\theta; \theta^{(t)}) = \mathrm{E}_{z|x,\,\theta^{(t)}}[l(\theta; x, Z)]$. Since the complete log-likelihood $l(\theta; x, z)$ is linear in the latent data z, to obtain $Q(\theta; \theta^{(t)})$ it suffices to replace each Z_i by its conditional expectation given the observed data x and the current fit $\theta^{(t)}$ [17, p. 21].

We can see that $l(\theta; x, z)$ is linear in z from Equation 3. The factor α^{zi} can be replaced by the equivalent notation $1 + z_i(\alpha - 1)$ because $z_i \in \{0, 1\}$.

A similar approach, called the Baum-Welch algorithm, is used for estimating the parameters of Hidden Markov Models (HMMs) [40]. Since HMMs are a special case of Bayesian networks such as pedigrees, the same logic applies to our problem [17, cf. pp. 73–76].

A7 Evaluating Messages in Linear Time

For example, consider from Figure 1b the message

$$\mu_{\varphi_{789}\to z_8}(z_8) = \sum_{z_5=0}^{1}\sum_{z_6=0}^{1}\sum_{z_7=0}^{1}\sum_{z_9=0}^{1}\varphi_{789}(z_5, z_6, z_7, z_8, z_9) \cdot \mu_{z_5\to\varphi_{789}}(z_5)\,\mu_{z_6\to\varphi_{789}}(z_6)\,\mu_{z_7\to\varphi_{789}}(z_7)\,\mu_{z_9\to\varphi_{789}}(z_9).$$

Since we can decompose $\varphi_{789}(z_5, z_6, z_7, z_8, z_9)$ into the product $f(x_7|z_7)\,\mathrm{P}(z_7|z_5, z_6) \cdot f(x_8|z_8)\,\mathrm{P}(z_8|z_5, z_6) \cdot f(x_9|z_9)\,\mathrm{P}(z_9|z_5, z_6)$, the quadruple sum can be split up into

$$\mu_{\varphi_{789}\to z_8}(z_8) = \sum_{z_5=0}^{1}\sum_{z_6=0}^{1}\left\{\mu_{z_5\to\varphi_{789}}(z_5)\cdot\mu_{z_6\to\varphi_{789}}(z_6) \cdot f(x_8 \mid z_8)\cdot\mathrm{P}(z_8\mid z_5, z_6) \cdot\left[\sum_{z_7=0}^{1}\mu_{z_7\to\varphi_{789}}(z_7)\,f(x_7\mid z_7)\,\mathrm{P}(z_7\mid z_5, z_6)\right]\right\} \left[\sum_{z_9=0}^{1}\mu_{z_9\to\varphi_{789}}(z_9)\,f(x_9\mid z_9)\,\mathrm{P}(z_9\mid z_5, z_6)\right].$$

This representation of the marginalizing sum can now be evaluated in linear time respective to the number of children.

References

1 Kaatsch P, Spix C, Hentschel S, Katalinic A, Luttmann S, Stegmaier C, Caspritz S, Cernaj J, Ernst A, Folkerts J, et al: Krebs in Deutschland 2009/2010. Berlin, Robert Koch-Institut, 2013.

2 Kolligs FT, Crispin A, Munte A, Wagner A, Mansmann U, Göke B: Risk of advanced colorectal neoplasia according to age and gender. PLoS One 2011;6:e20076.

3 Half E, Bercovich D, Rozen P: Familial adenomatous polyposis. Orphanet J Rare Dis 2009; 4:22.

4 Mendelsohn RB, Markowitz AJ: Hereditary colon cancer. Eur Gastroenterol Hepatol Rev 2011;7:251–256.

5 Bermejo JL, Hemminki K: Familial risk of cancer shortly after diagnosis of the first familial tumor. J Natl Cancer Inst 2005;97: 1575–1579.

6 Lange K: Mathematical and Statistical Methods for Genetic Analysis. New York, Springer, 1997.

7 Jarvik GP: Complex segregation analyses: uses and limitations. Am J Hum Genet 1998; 63:942–946.

8 Houle D, Govindaraju DR, Omholt S: Phenomics: the next challenge. Nat Rev Genet 2010;11:855–866.

9 Rieger A, Mansmann UR: Bayesian prediction of being a colorectal cancer risk family. In preparation.

10 Baum LE: An equality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. Inequalities 1972;3:1–8.

11 Kschischang FR, Frey BJ, Loeliger HA: Factor graphs and the sum-product algorithm. IEEE Trans Inf Theory 2001;47:498–519.

12 Failmezger H, Dursun E, Schroeder T, Krug A, Tresch A: Quantification of deterministic and stochastic cell fate components using hidden factor graph models. PLoS Comp Biol. Submitted.

13 Thompson EA: Statistical inference from genetic data on pedigrees. In NSF-CBMS Regional Conference Series in Probability and Statistics. Beachwood, Institute of Mathematical Statistics, 2000, vol 6.

14 Nelder JA, Mead R: A simplex method for function minimization. Comput J 1965;7: 308–313.

15 Dellaert F: The Expectation Maximization Algorithm. Technical Report, 2002.

16 Dempster AP, Laird NM, Rubin DB: Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc Series B Stat Methodol 1977;39:1–38.

17 McLachlan G, Krishnan T: The EM Algorithm and Extensions. Hoboken, John Wiley & Sons, 2007.

18 Abbi R, El-Darzi E, Vasilakis C, Millard P: Analysis of stopping criteria for the EM algorithm in the context of patient grouping according to length of stay. In 2008 4th International IEEE Conference Intelligent Systems. IEEE, 2008, vol 1, p 3–9.

19 Nielsen TD, Jensen FV: Bayesian Networks and Decision Graphs. Berlin, Springer Science & Business Media, 2009.

20 Aliferis CF, Tsamardinos I, Statnikov AR, Brown LE: Causal explorer: a causal probabilistic network learning toolkit for biomedical discovery. In METMBS 2003;3:371–376.

21 Mezard M, Montanari A: Information, Physics, and Computation. Oxford, Oxford University Press, 2009.

22 Ihler AT, John III WF, Willsky AS: Loopy belief propagation: convergence and effects of message errors. J Mach Learn Res 2005;6:905–936.

23 Goddard K, Yu CE, Oshima J, Miki T, Nakura J, Piussan C, Martin GM, Schellenberg GD, Wijsman EM: Toward localization of the Werner syndrome gene by linkage disequilibrium and ancestral haplotyping: lessons learned from analysis of 35 chromosome 8p11.1-21.1 markers. Am J Hum Genet 1996; 58:1286.

24 R Core Team: R: A Language and Environment for Statistical Computing. Vienna, R Foundation for Statistical Computing, 2016.

25 Boyd S, Vandenberghe L: Convex Optimization. Cambridge, Cambridge University Press, 2004.

26 Byrd RH, Lu P, Nocedal J, Zhu C: A limited memory algorithm for bound constrained optimization. SIAM J Sci Comput 1995;16: 1190–1208.

27 Bélisle CJ: Convergence theorems for a class of simulated annealing algorithms on R^d. J Appl Probab 1992;29:885–895.
28 Mansmann U, Stausberg J, Engel J, Heussner P, Birkner B, Maar C: Familien schützen und stärken – Umgang mit familiärem Darmkrebs. Eine Pilotstudie zur Inzidenz von Risikoclustern und zur Möglichkeit ihrer Detektion. Der Gastroenterologe 2012;7:271–272.
29 The Munich Cancer Registry. http://www.tumorregister-muenchen.de/en/index.php, 2016 (accessed October 11, 2016).
30 Nasseh D, Engel J, Mansmann U, Tretter W, Stausberg J: Matching study to registry data: maintaining data privacy in a study on family based colorectal cancer. Stud Health Technol Inform 2014;205:808–812.
31 Elston RC, Stewart J: A general model for the genetic analysis of pedigree data. Hum Hered 1971;21:523–542.
32 Cannings C, Thompson E, Skolnick M: Probability functions on complex pedigrees. Adv Appl Probab 1978;10:26–61.
33 Belonogova NM, Axenovich TI: Optimal peeling order for pedigrees with incomplete genotypic information. Comp Biol Chem 2007;31:173–177.
34 Thompson EA, Shaw R: Pedigree analysis for quantitative traits: variance components without matrix inversion. Biometrics 1990; 46:399–413.
35 Guo SW, Thompson EA: A Monte Carlo method for combined segregation and linkage analysis. Am J Hum Genet 1992;51:1111.
36 Gelfand AE, Smith AF: Sampling-based approaches to calculating marginal densities. J Am Stat Assoc 1990;85:398–409.
37 Geyer CJ, Thompson EA: Constrained Monte Carlo maximum likelihood for dependent data. J R Stat Soc Series B Stat Methodol 1992; 54:657–699.
38 Brenner H, Hoffmeister M, Haug U: Family history and age at initiation of colorectal cancer screening. Am J Gastroenterol 2008;103: 2326–2331.
39 Ghahramani Z: An introduction to hidden Markov models and Bayesian networks. Intern J Pattern Recognit Artif Intell 2001;15: 9–42.
40 Rabiner LR: A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 1989;77:257–286.