MATH 697
ITERATIVE PROPORTIONAL SCALING

REN BETTENDORF

1. Abstract

The purpose of this project is to show observations taken from the Iterative Proportional Scaling algorithm when used with data taken from Hierarchical Log-Linear Models. We show with empirical proof that the difference between decomposable and non-decomposable graph models used is not exceedingly different. For the purpose of this project we looked at the properties of the two, three, and four variable case of graph models.

2. Introduction

For many years, statisticians and probabilists struggled to find an effective way to maximize likelihood estimators. It was through this work that these mathematicians discovered a way to maximize these likelihood estimates. The Iterative Proportional Scaling algorithm (IPS) was first introduced in 1940 by W. Edwards Deming and Frederick Stephan. Deming and Stephan used it mainly to maximize likelihood estimators given a finite discrete set of data with fixed marginal totals. After its introduction, Darroch and Ratcliff gave an algorithm for generalized log-linear models when the expected frequencies are restricted by equalities. It was from this result that the current version of the algorithm exists and was used for this project.

The goal of using the Iterative Proportional Scaling algorithm is to maximize a set of likelihood estimators in the form of a contingency table subject to the marginal probabilities. This is achieved through solving a linear system by taking the data from the table and putting it into the form of a data vector and a matrix generated from a graph or graphical model.

3. Hierarchical Log-Linear Models

In this project, we specifically use graph models that come from hierarchical log-linear models. These models are parametrized by the different levels that each vertex can take and generate a joint distribution given the set of edges $\{X_1, X_2, \cdots, X_n\}$ that have the following levels $[1, 2, \cdots, d_i]$:

$$p_i = p_{i_1,\ldots,i_n} = X_1 * X_2 * \cdots * X_n$$

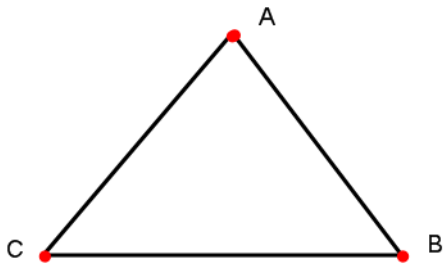
We can also look at this model through the following parametrization

$$\prod_{i=1}^{n} p_i^{u_i}$$

Date: May 28, 2012.

Where u_i is the parametrization.

Example 1. *Consider the following triangle model:*



We will call the edge connecting 1 *and* 2 *to be* α *which takes values* i, j*, the edge connecting* 2 *and* 3 *to be* β *which takes values* j, k*, and the edge connecting* 1 *and* 3 *to be* γ *which takes values* i, k*. So the following model is:*

$$p_i = \alpha_{ij}\beta_{jk}\gamma_{ik}$$

Now if we fix a matrix, A whose columns all sum to the same value k. The *log-linear model* associated with A is the set of probability tables taking values of 0 or 1.

$$\mathcal{M}_{\mathcal{A}} = \{p = (p_i) : log(p) \in rowspan(A)\}$$

where $rowspan(A)$ is the linear space spanned by the rows of A and also $log(p)$ denotes the vector whose i-th coordinate is the logarithm of the positive real number p_i. It should be noted also that *toric model* is used to denote the model used.

Example 2. *A* $d \times k$ *probability table* $p = (p_{ij})$ *can be generated from the following graph model, where the vertices both take values from* $\{1, 2, 3\}$*. Therefore as long as each* p_{ij} *can be factored into a product of marginal probabilities,* p_{i+} *and* p_{+j} *then the following is the log-linear model*

$$\log(p_{ij}) = \log(p_{i+}) + \log(p_{+j})$$

Therefore we have the following matrix generated from the row span of the vectors that generate A*.*

$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccccccccc} 11 & 12 & 13 & 21 & 22 & 23 & 31 & 32 & 33 \end{array} \\ \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

The first three numbers to the left of the matrix refer to the first number on top of the matrix and the second set of three refers to the second number. One important part to recognize is that the dimensions of A *is a* $(d+k) \times dk$ *matrix. It is from this aspect that it is easy to develop an algorithm that generates this matrix and is given*

in the following pseudocode after this example. Another important observation is that in this case we used two random variables i and j and the sum of every column is 2 which is the same as the cardinality of the set of random variables.

The following is the algorithm that was used to generate all the A matrices for this project.

Algorithm 1.
Input: Array B of edge connections, Array C with levels of all vertices Output: Matrix A

(1) Determine number of columns, $N = \prod_{i=1}^{k} C_i$

(2) Determine rows for every edge through multiplying cardinalities together then summing $\{r_1, r_2, \cdots, r_l\}$

(3) Set A with these dimensions with every entry 0.

(4) Set counter variables for number of vertices

(5) For $i = 1 : n$
For $j = 1 : r_1$
if counters are same as i and j, $A(i, j) = 1$*, else* $A(i, j) = 0$
Repeat until through every edge
Increase counter variables starting with the "last"

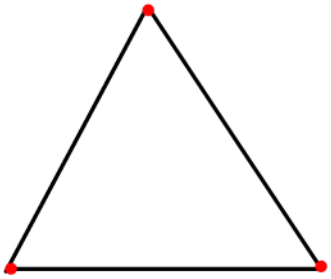
For this project, we used graph models using two, three, or four variables that generated the A matrix. These models could either be described as decomposable or indecomposable. In the case of graph models, a model is considered decomposable if it upholds the following properties when given a graph $G = (V, E)$

(1) G is complete
(2) We can express V as $V = A \cup B \cup C$
(3) A,B, and C are disjoint
(4) A and C are non-empty
(5) B is complete
(6) B separates A and C in G
(7) $A \cup B$ and $B \cup C$ are decomposable

From this definition we were able to label the models that we used as being either decomposable or indecomposable. The following are pictures of the models that we used and a description if they are decomposable or not.

Model 1.A

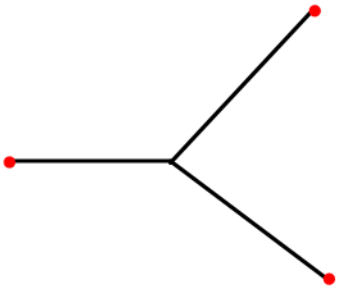
Model 1.B



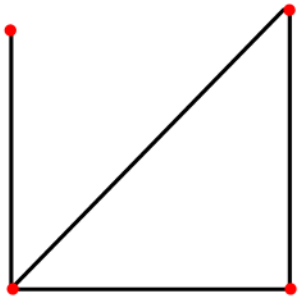
Model 2.A

Model 2.D

Model 2.B



Model 2.C



4. Iterative Proportional Scaling Algorithm

We know from *Lectures on Algebraic Statistics*, that if we are given a matrix A with row and column dimensions d, k and $u \in \mathbb{N}^k$ is a vector of positive counts. The maximum likelihood estimate of the frequencies of $\hat{u}$ in the log-linear model $\mathcal{M}_{\mathcal{A}}$ is the unique non-negative solution to the simultaneous system of equations

$$A\hat{u} = Au$$

We know that this has a solution due to Birch's Theorem. In the next section we will show how this system can be expressed using this definition.

The Iterative Proportional Scaling algorithm is defined through the following: Let there be n random variables denoted as such $S = \{\boldsymbol{X_1}, \boldsymbol{X_2}, \cdots, \boldsymbol{X_n}\}$ where these take a value from the set $D = \{1, 2, \cdots, C\}$. As well each element in S combines with each other to create a set of probabilities $P_{i_1,\cdots,i_n} = P(X_1 = i_1, \cdots, X_n = i_n)$ where each p_i has a corresponding frequency u_i. Therefore, we can denote the *likelihood function* as we did at the beginning of the Hierarchical Log-Linear Models section.

$$p_i = P_{i_1,\cdots,i_n} = P(X_1 = i_1, \cdots, X_n = i_n) = x_1 * x_2 * \cdots * x_n$$

This leads to the following problem when we parametrize our model as a product of each p_i using u_i as the parametrization:

Maximize: $\prod_{i=1}^{n} p_i^{u_i}$

Subject to $\sum_{i=1}^{n} p_i = 1$

$$p = \prod_{i=1}^{n} p_i^{u_i}$$

Where using log-likelihood function is equivalent to the following

$$ln(p) = \sum_{i=1}^{n} u_i log(p_i)$$

Now if we let each p_i can be expressed as the unique combination of values taken from S and take n partial derivatives we see that we have n linear equations which can be expressed as a matrix $A\hat{\boldsymbol{p}}$. Given that A is the matrix generated from the log-linear graph model and each column sums α. Now introducing Lagrangian multipliers from the boundary condition, these are set equal to $A\hat{p}$ and are denoted as $\dfrac{A\boldsymbol{u}}{n}$. Which gives us the definition we sought.

$$nA\hat{\boldsymbol{p}} = A\boldsymbol{u}$$

Now we wish to show how the algorithm solves this system. We start by looking at the ith row and multiplying both sides by $\dfrac{1}{\hat{p}}$

$$\frac{(A\mathbf{u})_i}{\hat{p}_i} = \frac{n(A\hat{\mathbf{p}})_i}{\hat{p}_i}$$

Where $(A\mathbf{u})_i$ refers to the ith row in the multiplication which gives us the marginal total and the same for $(A\hat{\mathbf{p}})_i$. As well we are going to let the right $\hat{p}_i$ be the

kth iteration and the left will be the $k+1$st iteration. Therefore, multiplying both sides by $\hat{p}_i^k \hat{p}_i^{k+1}$.

$$(A\mathbf{u})_i \hat{p}_i^{(k)} = n(A\hat{\mathbf{p}})_i \hat{p}_i^{(k+1)}$$
$$\Rightarrow \hat{p}_i^{k+1} = \hat{p}_i^k \frac{(A\mathbf{u})_i}{n(A\hat{\mathbf{p}})_i}$$

Therefore we have defined the main part of the algorithm, but are not complete because we need to take the α-th root because we are multiplying these numbers α times. Now all we have to do is run this through a loop until the difference between $\hat{p}^{(k+1)} - \hat{p}^{(k)} \leq \epsilon$ where ϵ is defined by the user.

Rewriting the above into pseudocode

Algorithm 2.
Input: The matrix $A \in \mathbb{N}^{dxk}$, a table of counts $u \in \mathbb{N}^k$, and a tolerance $\epsilon > 0$.
Output: Expected counts $\hat{p}$.

(1) Initialize $p_i = \|u\|_1 * \frac{1}{k}$

(2) Initialize $\alpha =$ sum of one column.

(3) While $\|Av - Au\|_1 > \epsilon$ do:
For all $i \in [k]$, set $p_i := p_i \left(\frac{(A\mathbf{u})_i}{n(A\hat{\mathbf{p}})_i}\right)^{\frac{1}{\alpha}}$

(4) Output $\hat{p}$.

5. Observations

At the end of this project we found four observations that defined how the Iterative Proportional Scaling algorithm works. These four observations are the difference between absolute and relative error make a huge difference, expansions along one variable does have a maximum limit such as in the case $\{3; 3; k\}$ case, the convergence between decomposable and indecomposable models, and how tolerance affects the algorithms convergence. The following iteration data was found through generating a random table of integers ranging from 0 to 100. This process was done 100 times and then the algorithm proceeded to take the average of these iterations. Before the average iterations though, the A matrix was created through considering the number of levels for each edge in the graph models. The random data table was then used for each model to see the difference.

1) When I first started using the Iterative Proportional algorithm, I followed the pseudocode from *Lectures on Algebraic Statistics* but I found that at a tolerance of 0.005 the algorithm was failing to converge while using absolute error in combination with a basic two independent variable model. By making the switch we were able to decrease the algorithm from failing to converge at 50 to converging in 7. So for the rest of the project, I continued to use the relative error which led to much faster convergences.

2) The next observation of this project came from looking at what happens when all but one level of a vertex of a graph model is kept constant. We then tracked

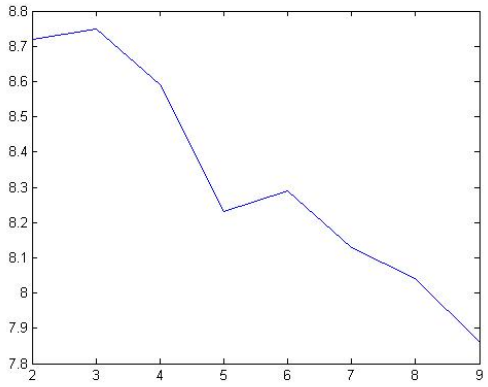
what happened as we increased the level by 1 in 2 and 3 vertex models. In the following data table we looked at what happened when we kept the number of edges the same where we saw interesting behaviour in both decomposable and indecomposable graph models. After looking at this behaviour, we compared the average iterations for models that had the same number of vertices.

TABLE 1. Indecomposable Line Graph Model

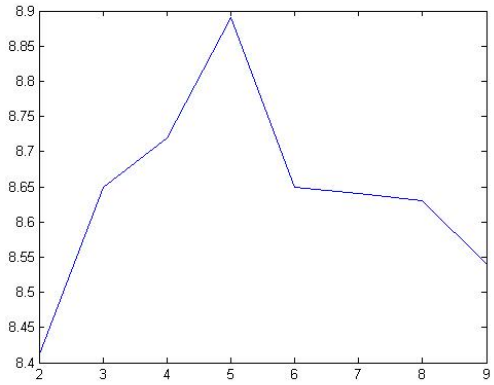
k	1.A	1.B
2	8.41	8.72
3	8.65	8.75
4	8.72	8.59
5	8.89	8.23
6	8.65	8.29
7	8.64	8.13
8	8.63	8.04
9	8.54	7.86

As we see the right model is decomposable as it follows the 7 properties defined in the hierarchical log-linear model section and we see that as k increases we get the following graph which shows the difference between the two models.

Decomposable 3 Edge Model



Indecomposable 3 Edge Model

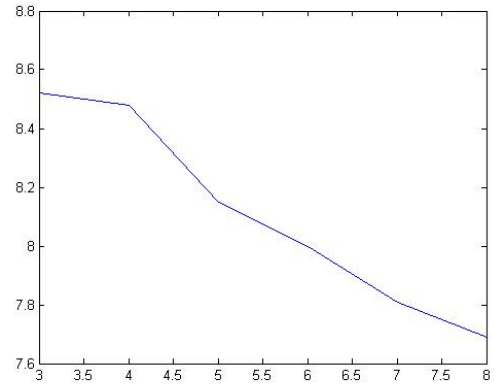


From the graphs we see a bump in the indecomposable model which follows from a theorem that there is a point in which no new models are generated after a certain point. This point empirically is shown through the model at the $\{2; 2; 5\}$ and $\{2; 2; 2; 5\}$. In the case of the decomposable case, we see that it also reaches this point, but at a quicker rate. We then recreated this affect by changing the data to $\{3; 3; k\}$ and $\{3; 3; 3; k\}$.

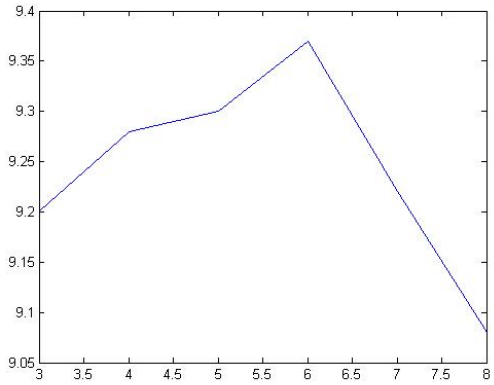
TABLE 2. Indecomposable Line Graph Model

k	1,2;1,3;2,3	1,2;2,3;3,4
3	9.20	8.52
4	9.28	8.48
5	9.30	8.15
6	9.37	8.00
7	9.22	7.81
8	9.08	7.69

Decomposable 3 Edge Model



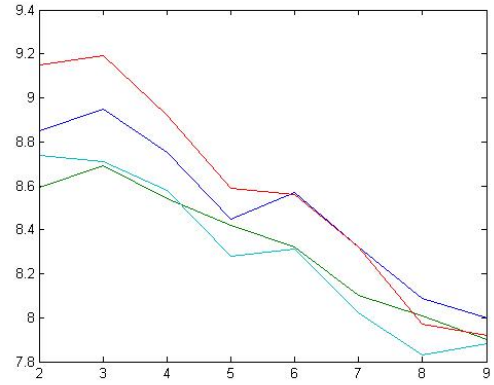
Indecomposable 3 Edge Model



3) Similarly to the previous case, we see that there is a bump in the indecomposable model, but for this case we see that when $k = 6$ we get the bump and then slopes back down. Again in the decomposable case, we see a gentle slope until $k = 4$ then a more aggressive downward slope which follows from the previous theorem. This shows that there is a fundamental difference between the decomposable and indecomposable case, but when we compare number of variables we see more of a difference.

TABLE 3. Indecomposable Line Graph Model

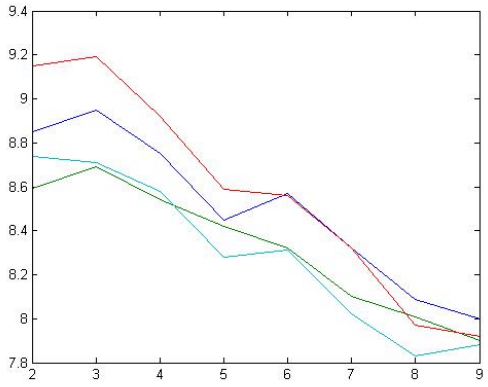
k	1,2;1,3;2,3	1,2;2,3;3,4		
2	8.85	8.59	9.15	8.74
3	8.95	8.69	9.19	8.71
4	8.75	8.54	8.92	8.58
5	8.45	8.42	8.59	8.28
6	8.57	8.32	8.56	8.31
7	8.32	8.10	8.32	8.02
8	8.09	8.01	7.97	7.83
9	8.00	7.90	7.92	7.88



4)We see that the models which are decomposable follow the same rules as above, as do the indecomposable models. It is easy to see from these three cases whether comparing number of edges or number of vertices, we see that the difference is not that different. It has a maximum difference of 0.20 iterations. This difference is because using the relative error means that we are looking at the relative difference which means that by comparing similar number of edges or vertices is irrelevant. While it is easy to see that as we increase the levels the number of iterations seems to scale equally for the different cases depending on decomposable or not. If we move onto the tolerance then we see that as the tolerance increases we see an asymptotic boundary which approaches 10 which is shown in the following graph and table.

TABLE 4. Indecomposable Line Graph Model

k	1,2;1,3;2,3	1,2;2,3;3,4		
5e-3	8.72	8.99	9.30	8.74
2.5e-3	10.33	10.76	11.69	11.10
1e-3	12.78	13.20	14.95	14.21
5e-4	14.58	15.07	17.52	16.64
2.5e-4	16.40	16.90	20.00	19.08
1e-4	18.85	19.44	23.37	22.23
5e-5	20.71	21.32	25.92	24.78
2.5e-5	22.55	23.16	28.51	27.18
1e-5	24.96	24.84	31.94	30.38
5e-6	26.81	24.99	34.55	32.86
2.5e-6	28.64	29.94	37.14	35.34
1e-6	31.13	32.00	40.58	38.58



As we have shown in this graph and data table the difference between decomposable and indecomposable models is not that different as the tolerance approaches 1. Yet, as the tolerance decreases we see a linear increase in the number of iterations for each model. This increase appears to be about 3 iterations for every increase in toleration. I'm not sure why this increase happens linearly, but it does suggest that there might be a linear relationship between the two.

6. References

(1) Bartlett, Peter. *Undirected Graphical Models: Chordal Graphs, Decomposable Graphs, Junction Trees, and Factorizations*. October 2003.

(2) Drton, Mathias, Bernd Sturmfels, and Seth Sullivant. *Lectures on Algebraic Statistics*. Basel: Birkhäuser, 2009