INVESTIGATING THE ML DEGREE

YUSEN LIU, XINYANG HU

Abstract. In this report, we first introduce the MLE problem and restate the problem with an optimization problem. We compute the ML degree of 3×3 and 4×4 symmetric matrices with different quasi-symmetric models as constraints. We also compute the prediction models $M_{3\times3}$ that cover most possibilities of ML degrees. For 3×3 and 4×4 symmetric matrices, we generate the distribution of ML degree based on the result that we have. Next, we investigate the algebraic complexity of critical points, and the explicit formula of them under special input data. With the definitions and conjectures we derive, we give a formulation of the critical points for the special data matrix.

1. MLE Problem Overview

We consider the problem of finding all critical points of the likelihood function, with the aim of identifying all local maxima. The defining equations of the critical points are the likelihood equations. The number of complex solutions to the likelihood equations (for generic data) is called the *maximum likelihood (ML) degree* of the model[4].

2. Methodology

2.1. **Problem restatement.** The problem described in section 1 is an unconstrained optimization problem, which is equivalent to the following constrained optimization problem:

Maximize $p_0^{u_0}p_1^{u_1}p_2^{u_2}p_3^{u_3}p_4^{u_4}$ subject to $det(M)=0$ and $p_0+...+p_4=1$,

$$\text{where M} = \begin{bmatrix} c_{11}p_0 & c_{12}p_1 & c_{13}p_2 \\ c_{21}p_1 & c_{22}p_2 & c_{23}p_3 \\ c_{31}p_2 & c_{32}p_3 & c_{33}p_4 \end{bmatrix}$$

2.2. **Example.** Consider the case where $\text{M} = \begin{bmatrix} 12p_0 & 3p_1 & 2p_2 \\ 3p_1 & 2p_2 & 3p_3 \\ 2p_2 & 3p_3 & 12p_4 \end{bmatrix}$

In order to find the ML degree of a specific determinant variety, we take matrix M as an example. Maximizing $p_0^{u_0}p_1^{u_1}p_2^{u_2}p_3^{u_3}p_4^{u_4}$ is equivalent to maximizing the logarithm of it, i.e. $\log p_0^{u_0}p_1^{u_1}p_2^{u_2}p_3^{u_3}p_4^{u_4}$.

Since

$$\log p_0^{u_0} p_1^{u_1} p_2^{u_2} p_3^{u_3} p_4^{u_4} = \sum_{i=0}^{4} u_i \log p_i \tag{1}$$

The problem is now equivalent to maximizing $\sum_{i=0}^{4} u_i \log p_i$, subject to $det(M) = 0$ and $p_0 + ... + p_4 - 1 = 0$.

Next, we need to introduce two Lagrange Multipliers, λ_1 and λ_2, so that we can transform the problem into maximizing $\phi = \sum_{i=0}^{4} u_i \log p_i + \lambda_1(p_0 + ... + p_4 - 1) + \lambda_2 det(M)$. Hence, we need to let

$$\frac{\partial \phi}{\partial p_i} = 0$$

i.e.

$$\frac{u_i}{p_i} + \lambda_1 + \lambda_2 \frac{\partial det(M)}{\partial p_i} = 0$$

Let $d_i = \frac{\partial det(M)}{\partial p_i}$, then the above equation can be rewritten as

$$f_i^* = \frac{u_i}{p_i} + \lambda_1 + \lambda_2 d_i = 0 \tag{2}$$

From (2), we can see that the variable p_i is the denominator. Since $p_i > 0$, we can multiply p_i on both sides:

$$p_i f_i^* = u_i + \lambda_1 p_i + \lambda_2 d_i p_i = 0 \tag{3}$$

Let

$$f_i = p_i f_i^* = u_i + \lambda_1 p_i + \lambda_2 d_i p_i = 0 \tag{4}$$

Since i's range is from 0 to 4, there will be 5 equations. With the two constraints, there will be 7 linearly independent equations with 7 variables. Hence, the polynomial system is solvable.

In order to solve the polynomial system described above, we need to introduce a computation tool - Macaulay2, which is a software system that could do mathematical computations in algebraic geometry[2].

The code in Macaulay 2 is shown below:

```
needsPackage "PHCpack"
R = CC[x1,x2,xx0,xx1,xx2,xx3,xx4]
```

I use $x1$, $x2$ to represent λ_1 and λ_2, and xx_i to represent p_i (i = 0,1,2,3,4). Using other variables will cause errors here. The following code is as follows

```
M = matrix{{12*xx0,3*xx1,2*xx2},{3*xx1,2*xx2,3*xx3},{2*xx2,3*xx3,12*xx4}}
detM = det(M)
#Next, compute the partial derivatives with respect to xx0...xx4.
d0=diff(xx0,detM)
d1=diff(xx1,detM)
```

```
d2=diff(xx2,detM)
d3=diff(xx3,detM)
d4=diff(xx4,detM)
```

Since $u_0 ... u_4$ is the observation data, we can assign them values randomly, with $u_0 + u_1 + u_2 + u_3 + u_4 = 1000$. The code is as follows:

```
u0 = random 250
u1 = random 250
u2 = random 250
u3 = random 250
u4 = 1000-u0-u1-u2-u3
```

Computation of f_i is as follows:

```
f0 = x1*xx0 + x2*d0*xx0 +u0
f1 = x1*xx1 + x2*d1*xx1 +u1
f2 = x1*xx2 + x2*d2*xx2 +u2
f3 = x1*xx3 + x2*d3*xx3 +u3
f4 = x1*xx4 + x2*d4*xx4 +u4
f5 = xx0+xx1+xx2+xx3+xx4-1
```

Finally, we need to solve the polynomial system, and the ML degree is the number of the solutions.

```
S = {f0,f1,f2,f3,f4,f5,detM}
sol = solveSystem(S)
length(sol)
```

The result is shown below

```
i27 : sol = solveSystem(S)
*** variables in the ring : {x1, x2, xx0, xx1, xx2, xx3, xx4} ***

o27 = {{-1000, -68.1101-34.3685*ii, .0901454-.0370969*ii, .0899334+.0807658*ii,
      ------------------------------------------------------------------------
      .0747324-.0096973*ii, .342153-.0745151*ii, .403035+.0405436*ii}, {-1000,
      ------------------------------------------------------------------------
      7.87914, .754153, -1.16371, .739807, .116909, .552843}, {-1000, -3.03123,
      ------------------------------------------------------------------------
      .0599081, .145414, .141221, .163684, .489773}, {-1000,
      ------------------------------------------------------------------------
      -194.881-81.1503*ii, .00579787-.00188128*ii, .0503142+.0586777*ii,
      ------------------------------------------------------------------------
      .400776-.0694786*ii, .146313-.0295506*ii, .396798+.0422329*ii}, {-1000,
      ------------------------------------------------------------------------
      -194.881+81.1503*ii, .00579787+.00188128*ii, .0503142-.0586777*ii,
      ------------------------------------------------------------------------
      .400776+.0694786*ii, .146313+.0295506*ii, .396798-.0422329*ii}, {-1000,
      ------------------------------------------------------------------------
      -2.88809, -.0316948, -2.12272, 3.47686, .663232, -.985671}, {-1000,
```

```
      ------------------------------------------------------------------------
      -4.81285, -1.2171, .288666, 3.08041, -1.03641, -.115565}, {-1000,
      ------------------------------------------------------------------------
      -.495463, -.0993614, -5.39818, 10.5249, -3.33575, -.691577}, {-1000,
      ------------------------------------------------------------------------
      -27.6142, -.538365, .114175, 1.4124, .108125, -.0963396}, {-1000,
      ------------------------------------------------------------------------
      -68.1101+34.3685*ii, .0901454+.0370969*ii, .0899334-.0807658*ii,
      ------------------------------------------------------------------------
      .0747324+.0096973*ii, .342153+.0745151*ii, .403035-.0405436*ii}, {-1000,
      ------------------------------------------------------------------------
      -2189.67, .00877598, .0523465, .262491, .404875, .271512}, {-1000,
      ------------------------------------------------------------------------
      11.0189, .22069, .066706, 1.02302, -2.00692, 1.6965}}

o27 : List

i28 : length(sol)

o28 = 12
```

Hence, we now can conclude that the ML degree of this polynomial system is 12.

2.3. **ML-Degree of 3×3 and 4×4 symmetric matrix with different quasi-symmetric models as constraints.** In a Maximum Likelihood Estimation problem, we are given a set of observations, and a probability matrix corresponding to each outcome's probability. For simplicity, we denote each observation as matrix $U_{n \times n}$, and the probability matrix $P_{n \times n}$. For each observation U, we can use the prediction model, denoted by $M_{n \times n}$, to solve for the likelihood equations.

ML degree[7] is the number of solutions to the likelihood equations, it gives a measure on the difficulty of the problem. The reason that we want to compute the ML-Degree is to find which prediction model $M_{n \times n}$ would give us some specific number of solutions, i.e. the ML degree. After that, with the models that we find, we can find the best model for the prediction problem by using the testing data.

In this section, we first set n to be 3, and the probability matrix P is set to be a symmetric matrix. We want to figure out which models would give us some specific ML degrees and trying to see if there's some pattern between the models that would give us the same ML degree.

Now, we are given a 3×3 symmetric probability matrix P and the corresponding 3×3 observation matrix U, where u_{ij} represents the observation number of p_{ij}. We want to compute the ML degree of matrix P.

Polynomial system outline:

$$\sum p_{ij} = 1 \tag{5}$$

$$\det M = 0 \tag{6}$$

Similar to what we have discussed previously, since we want to compute the maximum for $p_{11}^{u_{11}} p_{12}^{u_{12}} p_{13}^{u_{13}} p_{21}^{u_{21}} p_{22}^{u_{22}} p_{23}^{u_{23}} p_{31}^{u_{31}} p_{32}^{u_{32}} p_{33}^{u_{33}}$ with the upper two constraints, we need to introduce Lagrange Multiplier, and form some new constraints of the form: $u_{ij} + \lambda_1 p_{ij} + \lambda_2 d_{ij} p_{ij}$, where d_{ij} is the partial derivative of $\det M$ with respect to p_{ij}. Hence, step (3) is as follows:

$$u_{ij} + \lambda_1 p_{ij} + \lambda_2 d_{ij} p_{ij} \tag{7}$$

Before investigating into it, let's first see 5 quasi-symmetric 3×3 models[1], which are shown below:

$$\mathrm{M}_1 = \begin{bmatrix} 2p_{11} & p_{12} & p_{13} \\ p_{12} & 2p_{22} & p_{23} \\ p_{13} & p_{23} & 2p_{33} \end{bmatrix} M_2 = \begin{bmatrix} p_{11} & 1/2p_{12} & 1/2p_{13} \\ 1/2p_{12} & p_{22} & 1/2p_{23} \\ 1/2p_{13} & 1/2p_{23} & p_{33} \end{bmatrix} M_3 = \begin{bmatrix} p_{11} & 1/2p_{12} & p_{13} \\ 1/2p_{12} & p_{22} & p_{23} \\ p_{13} & p_{23} & p_{33} \end{bmatrix}$$
$$M_4 = \begin{bmatrix} p_{11} & p_{12} & p_{13} \\ p_{12} & p_{22} & p_{23} \\ p_{13} & p_{32} & p_{33} \end{bmatrix} M_5 = \begin{bmatrix} p_{11} & 1/2p_{12} & 1/2p_{13} \\ 1/2p_{12} & p_{22} & p_{23} \\ 1/2p_{13} & p_{32} & p_{33} \end{bmatrix}$$

Instead of using Macaulay 2, we use Julia to solve this problem.

The result showed that:
If we pick the quasi-symmetric model M_1, the number of real solutions will be 6.
If we pick the quasi-symmetric model M_2, the number of real solutions will be 6.
If we pick the quasi-symmetric model M_3, the number of real solutions will be 11.
If we pick the quasi-symmetric model M_4, the number of real solutions will be 13.
If we pick the quasi-symmetric model M_5, the number of real solutions will be 10.

Based on the result that we have, in order to find more M's that will give us different ML degrees, we need to firstly find the domain of the ML degree, i.e. the least and the largest possible values of ML degree. According to Bézout's theorem, the theorem states that the number of common points of two curves is at most equal to the product of their degrees, and equality holds if one counts points at infinity and points with complex coordinates. According to the Fundamental Theorem of Algebra, equation (5) has degree 1, equation (6) has degree 3, equation (7) has a total number of 6 equations, with each equation of degree 4. Therefore, by Bézout's theorem, this polynomial system has degree $4^6 \times 3 \times 1 = 12288$. This number is exactly equal to the number of paths tracked each time we solve the polynomial system.

Next, what we want to do is to try to find as much models M's that will give us some specific ML degrees as possible, and to find some patterns according to the models M's. The first step that we do is to randomly choosing from numbers ranging from 1 to 10, and assign them as coefficients to the model M. However, it takes a lot of loops that could give us one result with ML degree less than 24. So we look back to the quasi-symmetric models and find that the models have coefficients of p_{ij}'s 1,2 and 1/2. Therefore, we decide to narrow down the domain of the numbers we choose from: $\pm 1/4$,$\pm 1/2$,$\pm 3/4$,± 1,$\pm 5/4$. Combining the results of two sets of choices of coefficient matrix C, we pick some of the results and aggregate them into table 2.3 in Appendix.

We pick different scales of M to see if it would give us the same result. For instance, for the ML degree of 11, the corresponding matrix M is:

$$\begin{bmatrix} -0.25p_{11} & 0.5p_{12} & 0.25p_{13} \\ 0.25p_{12} & p_{22} & -0.5p_{23} \\ 0.25p_{13} & -p_{23} & -p_{33} \end{bmatrix};$$

If we scale it by 10000, we will have a new matrix $M' =$

$$\begin{bmatrix} -2500p_{11} & 5000p_{12} & 2500p_{13} \\ 2500p_{12} & 10000p_{22} & -5000p_{23} \\ 2500p_{13} & -10000p_{23} & -10000p_{33} \end{bmatrix};$$

The number of non-singular solutions will vary with different observation data matrix U. Therefore, we cannot scale the coefficients of the matrix M. What we need to do next is to find some patterns based on the models that we've got. Our first approach is to compute the determinant of each M and check if they are equal. Our second approach is to compute the sum of the $(n-1) \times (n-1)$-minors to see if they are equal.

Consider the coefficient matrices with ML degree equal to 23. We choose the following coefficient matrices:

$$M_1 = \begin{bmatrix} 1 & 2 & 2 \\ 2 & 1 & 1 \\ 1 & 2 & 2 \end{bmatrix}, M_2 = \begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 1 \\ 2 & 2 & 2 \end{bmatrix}, M_3 = \begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 1 & 1 \end{bmatrix}, M_4 = \begin{bmatrix} 2 & 2 & 1 \\ 2 & 1 & 2 \\ 2 & 2 & 2 \end{bmatrix}, M_5 = \begin{bmatrix} 2 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix},$$
$$M_6 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 2 & 1 \end{bmatrix}, M_7 = \begin{bmatrix} 2 & 2 & 2 \\ 1 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix}.$$

Firstly, we compute the determinant of the matrices. The result shows that all of the determinants are 0 except for M_4, whose determinant is -2. Secondly, we compute the sum of the 2×2-minors. The result shows that for M_1, M_2, M_6, M_7, the sum of the 2×2-minors are 0, while for M_3, M_4, M_5 the sum of the 2×2-minors are -8, 7 and 6 respectively. Thirdly, we try to take the reciprocal of the entries of the matrices. The determinant of the new matrices are all 0 except that of M_4, which is -0.125. We then compute the sum of the 2×2-minors of them. The result shows that for M_1, M_2, M_6, M_7, the sum of the 2×2-minors are 0, while for M_3,M_4,M_5 the sum of the 2×2-minors are 1, -0.125 and -1.5 respectively. What we do next is to take the inverse of the matrices and do the same. However, we do not find any pattern.

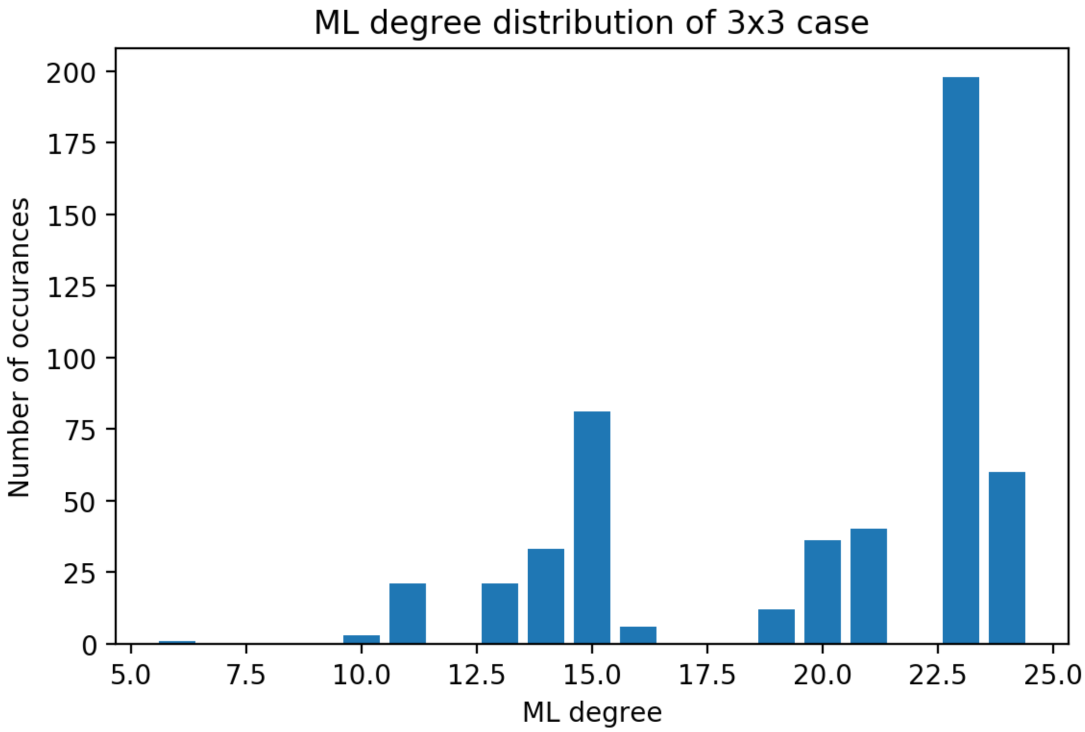

FIGURE 1. ML degree distribution (3×3 case)

We then focus on finding the distribution of the ML degrees of 3×3 and 4×4 symmetric cases. We restrict the coefficients of the prediction matrix M to be within $\{1, 2\}$. Firstly, we consider the 3×3 case. We create a list of permutations of 1 and 2 of 9 digits. Next, we assign random floating numbers to the observation matrix U and solve the system. We store all the ML degrees and the distribution is shown in Figure 1.

Next, we focus on the 4×4 case. The setup of the 4×4 case is similar to the 3×3 case and the constraints are similar.

Polynomial system outline:

$$\sum p_{ij} = 1, i, j = 1, 2, 3, 4 \tag{8}$$

$$\det M = 0 \tag{9}$$

Similar to what we have discussed in section 1.4, since we want to compute the maximum for $p_{11}^{u_{11}} p_{12}^{u_{12}} p_{13}^{u_{13}} p_{14}^{u_{14}} p_{21}^{u_{21}} p_{22}^{u_{22}} p_{23}^{u_{23}} p_{24}^{u_{24}} p_{31}^{u_{31}} p_{32}^{u_{32}} p_{33}^{u_{33}} p_{34}^{u_{34}} p_{41}^{u_{41}} p_{42}^{u_{42}} p_{43}^{u_{43}} p_{44}^{u_{44}}$ with the upper two constraints, we need to introduce Lagrange Multiplier, and form some new constraints of the form: $u_{ij} + \lambda_1 p_{ij} + \lambda_2 d_{ij} p_{ij}$, where d_{ij} is the partial derivative of $\det M$ with respect to p_{ij}. Hence, step (6) is as follows:

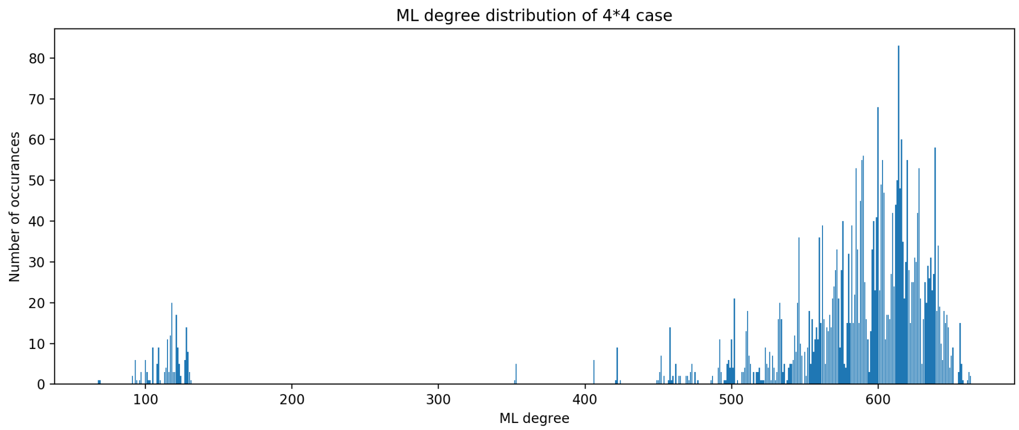

FIGURE 2. ML degree distribution ($4\times 4 case$)

$$u_{ij} + \lambda_1 p_{ij} + \lambda_2 d_{ij} p_{ij}, i, j = 1, 2, 3, 4 \tag{10}$$

What we do first is using HomotopyContinuation in Julia to solve the system with entries of M randomly chosen. However, it takes approximately two days to solve the system. We optimize the solving process by using the idea of polyhedral homotopy[5], and the implementation is Anders Jensen's algorithm[6], through which, the number of paths tracked that required to solve the system is significantly reduced. During the computing process, we find that for some specific prediction model M_i, different observation data matrix U will lead to different ML degrees. In order to solve this, for each prediction model M_i, we compute three times, and only if we get three identical ML degrees for a specific prediction model M_i can we assign the ML degree to M_i. Instead of randomly choosing the entries of matrix M from the set $1, 2$, we first list all the permutations of $\{1, 2\}$ of 16 digits long, and then solve each polynomial system one after one.

What we have so far is two arrays. One array is called MLDegreeArray which records the ML degree with respect to different prediction models M. The other is called NumOfRealSolns which records the number of real solutions with respect to different prediction models M. Since there are a lot of duplicates in MLDegreeArray, we have to use the method unique() to obtain an array containing only the unique elements of MLDegreeArray. Based on the result that we have, we can conclude it using a bar chart, as shown in Figure 2. The x-axis is the ML degrees and the y-axis is the number of prediction models M that would give us some specific ML degrees. Observing the figure, we can conclude that based on what we have computed so far, the models would give us ML degrees ranging from 93 to 128 and from 459 to 656. For the array NumOfRealSolns, it is all zeros.

2.4. **Conclusions of computational result.** Different 3×3 coefficient matrices C_i of prediction matrices M_i's with its corresponding ML degrees are shown in table 2.3. If we use HomotopyContinuation in Julia, then in each computation, the time cost to solve the 3×3 case is approximately 12 seconds, while the time cost to solve the 4×4 case is approximately 2 days. The number of paths tracked each time we solve the 3×3 cases is identical, which is 12288. 12288 is the number of complex solutions according to Bézout's theorem. If we use polyhedral homotopy, then in each computation, the time cost to solve the 3×3 case is strictly less than 1 second, while the time cost to solve the 4×4 case is approximately 30 seconds. The number of paths tracked each time we solve the 4×4 cases is identical, which is 6036. Polyhedral homotopy is much more efficient than HomotopyContinuation, and it has significantly less number of paths tracked than that of using HomotopyContinuation. Based on what we have computed so far, the models would give us ML degrees ranging from 68 to 663. There are two portions $[68, 132]$ and $[351, 663]$ that are more likely to be the values of ML degrees. For the array NumOfRealSolns, it is all zeros, which means that we have not found a prediction model M, with coefficients chosen from $\{1, 2\}$ that will give us any complex solution yet.

3. ML degree of symmetric rank 2 matrices

In this section, we investigate the algebraic complexity of critical points, and the explicit formula of them under special input data.
We introduce the definitions of ML degree and MLS degree in section 3.1, and make conjectures about their algebraic value. In 3.2, we look at the explicit (positive real) solutions of a type of special matrix $U(a, b)$ in terms of equivalent classes. Section 3.3 discusses the missing solutions base on the local kernel formulation when applying to the specified data. In 3.4 we connect all the previous subsections to give a formulation of the critical points for the special data matrix. And we will discuss the properties of the global maximizer out of the critical points in 3.5.

3.1. **ML degree under special data matrix.**

Definition 3.1 (ML(n))**.** For $n \geq 2$, let $ML(n)$ denotes the amount of critical points of the likelihood function with respect to generic samples on the set of rank 2 symmetric $n \times n$ matrices.

To perform computation of $ML(n)$, we use the symmetric local kernel formulation proposed by Hauenstein et al.[3]. As defined in (2.12) and (2.13) in the paper, we define the following 5 variables:

P_1: A full rank $r \times r$ symmetric matrix, with diagonal entries multiplied by 2.
L_1: A $(n-r) \times r$ variable matrix.
λ: A symmetric $(n-r) \times (n-r)$ matrix.

$$L = (L_1 - I_{m-r})$$
$$P = \begin{bmatrix} P_1 & P_1 L_1^T \\ L_1 P_1 & L_1 P_1 L_1^T \end{bmatrix}$$

Let D denotes a $n \times n$ matrix with 2 on the diagonal and 1 else.
We find the critical points by solving the following system:

$$P \star (L^T \cdot \lambda \cdot L) + \sum_{i \leq j} u_{ij} \cdot P = D \star U$$

where U is the data matrix, and $\star$ represents the Hadamard product.

Theorem 2.1 in Hauenstein et al. [3] states that for a generic data matrix, the number of critical points found by the local kernel formulation coincides with ML(n).
However, for specially chosen data, we will miss some of the zeros.
For example, let $a > b > 0$, $n \geq 2$, and consider the $n \times n$ data matrix with a on the diagonal, and b everywhere else, write this matrix as $U(a, b)$.

Definition 3.2 (MLS(n))**.** For $U(a, b)$ type of matrices, the number of critical points found by the local kernel formulation is less than ML(n). We denote this number by MLS(n).

We will discuss relationship associating with $ML(n)$ and $MLS(n)$ in the following conjectures. Furthermore, we will look into the closed-form expression of the missing solutions in later sections.

Conjecture 3.3. *For* $n \geq 2$,

$$ML(n-1) = ML(n) - MLS(n)$$

.

Example 3.4. This is a table illustrating the relationship in the previous conjecture.

n=	2	3	4	5	6	7
ML(n)=	1	6	37	270	2341	23646
MLS(n)=	1	5	31	233	2071	21305
ML(n)-MLS(n)=		1	6	37	270	2341

Conjecture 3.5. *For* $n \geq 2$,

$$MLS(n) = \sum_{k=1}^{n} \left\{ {n \atop k} \right\} \cdot k \cdot k!$$

$$\left\{ {n \atop k} \right\} = Stirling2(n, k) = \frac{1}{k!} \sum_{i=1}^{k} (-1)^i \binom{k}{i} (k-i)^n$$

Example 3.6.

$$MLS(2) = 1 \cdot 1 \cdot 1! + 1 \cdot 2 \cdot 2! = 5$$
$$MLS(3) = 1 \cdot 1 \cdot 1! + 3 \cdot 2 \cdot 2! + 1 \cdot 3 \cdot 3! = 31$$

Conjecture 3.7. *For* $n \geq 2$,

$$ML(n) = \sum_{k=2}^{n} T(n, k) \cdot 2^{k-2}$$

where $T(n, k)$ *denotes the Eulerian numbers,* $1 \leq k \leq n$.

Example 3.8.

$$ML(2) = 1 \cdot 2^0 = 1$$

$$ML(3) = 4 \cdot 2^0 + 1 \cdot 2^1 = 6$$

$$ML(4) = 11 \cdot 2^0 + 11 \cdot 2^1 + 1 \cdot 2^2 = 37$$

3.2. **Closed-form positive real solutions for $U(a, b)$ matrices.**
In this section, we take a closer look at the explicit expressions of the critical points. We will introduce some properties of the local extremes. Next we will define the classification of the positive real solutions, and then discuss the recursive formulation going from the zeros of $n \times n$ case to the $(n+1) \times (n+1)$ case.

Lemma 3.9. *Write the normalized $n \times n$ probability matrix as*

$$P = \begin{bmatrix} e_{11} & e_{12} & \dots & e_{1n} \\ e_{21} & e_{22} & \dots & e_{2n} \\ \vdots & \vdots & \dots & \\ e_{n1} & e_{n2} & \dots & e_{nn} \end{bmatrix}$$

Then if p is a critical point, $\forall 1 \leq i \leq n, \sum_{j=1}^{n} e_{ij} = \frac{1}{n}, \sum_{j=1}^{n} e_{ji} = \frac{1}{n}$.

This statement follows from Lemma 1 in Zhu et al.[8]*. Note that although Zhu et al. assumed the variable pair (a, b) to be $(2, 1)$ in their problem setting, the authors did not use the specific value in the proof of the Lemma 1, therefore we can do such a generalization.*

Lemma 3.10. *WLOG, let $(a, b) = (k, 1)$. Write the normalized $n \times n$ probability matrix*

$$P = \begin{bmatrix} e_{11} & e_{12} & \dots & e_{1n} \\ e_{21} & e_{22} & \dots & e_{2n} \\ \vdots & \vdots & \dots & \\ e_{n1} & e_{n2} & \dots & e_{nn} \end{bmatrix}$$

Then for an optimal solution, we need for each $1 \leq i \leq n$:

$$\frac{k-1}{e_{ii}} + \sum_{j=1}^{n} \frac{1}{e_{ij}} = n^2(k + n - 1)$$

Proof. This proof is adapted from the paper by Zhu et al.[8]. Scale the probability matrix by n^2, we rewrite the matrix as

$$X = n^2 P \begin{bmatrix} 1 + a_1 b_1 & 1 + a_1 b_2 & \dots & 1 + a_1 b_n \\ 1 + a_2 b_1 & 1 + a_2 b_2 & \dots & 1 + a_2 b_n \\ \vdots & \vdots & \dots & \\ 1 + a_n b_1 & 1 + a_n b_2 & \dots & 1 + a_n b_n \end{bmatrix} \tag{11}$$

And our normalized data matrix is

$$U = \begin{bmatrix} k & 1 & \dots & 1 \\ 1 & k & \dots & 1 \\ \vdots & \vdots & \dots & \\ 1 & 1 & \dots & k \end{bmatrix}$$

Then our problem becomes

$$\text{Maximize: } l(X) = \textstyle\sum_{i \neq j} ln(1 + a_i b_j) + k \sum_{i=1}^{n} ln(1 + a_i b_i)$$
$$\text{Subject to: } 1 + a_i b_j \geq 0, \textstyle\sum_{1 \leq i,j \leq n} 1 + a_i b_j = n^2$$

By $\sum_{1 \leq i,j \leq n} 1 + a_i b_j = n^2$, we need

$$\sum_{i=1}^{n} a_i \sum_{i=1}^{n} b_i = 0$$

By *Lemma 1* in Zhu. et al., we have

$$\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} b_i = 0$$

The Lagrange multiplier gives us

$$f(X, t) = l(X) + t \sum_{i=1}^{n} a_i \sum_{i=1}^{n} b_i, t \in \mathbb{R}$$

$$\frac{\partial f}{\partial a_i} = (k-1) \frac{b_i}{1 + a_i b_i} + \sum_{j=1}^{n} \frac{b_j}{1 + a_i b_j} = 0 \tag{12}$$

$$\frac{\partial f}{\partial b_i} = (k-1) \frac{a_i}{1 + a_i b_i} + \sum_{j=1}^{n} \frac{a_j}{1 + a_j b_i} = 0 \tag{13}$$

Note that

$$(k-1) \frac{1 + a_i b_i}{1 + a_i b_i} + \sum_{j=1}^{n} \frac{1 + a_i b_j}{1 + a_i b_j} = n + k - 1 = a_i \times (1) + \frac{k-1}{1 + a_i b_i} + \sum_{j=1}^{n} \frac{1}{1 + a_i b_j} \tag{14}$$

$$(k-1) \frac{1 + a_i b_i}{1 + a_i b_i} + \sum_{j=1}^{n} \frac{1 + a_j b_i}{1 + a_j b_i} = n + k - 1 = b_i \times (1) + \frac{k-1}{1 + a_i b_i} + \sum_{j=1}^{n} \frac{1}{1 + a_j b_i} \tag{15}$$

Since $1 + a_i b_j = n^2 e_{i,j}, \forall 1 \leq i, j \leq n$, we obtained the desired equality.

□

As we can see in section 2.1, the number of critical points grow rapidly as the dimension increases. Therefore, it is helpful to divide them into subgroups. In the following subsections, we classify the zeroes into equivalence classes, and study the behaviour within each class, and among different classes.

Lemma 3.11 (Type 1 equivalence class). *Suppose P is a positive real matrix of the following form,*

$$\begin{bmatrix} M_1 & \dots & H_i & \dots & H_N \\ \vdots & \ddots & & & \\ H_i^T & & M_i & & \\ \vdots & & & \ddots & \\ H_N^T & & & & M_N \end{bmatrix}$$

where for $1 \leq i \leq N$, $N \geq 1$, M_i is a block matrix with uni-valued diagonal and off diagonal. Different M_i have different diagonal value m_i, i.e.

$$M_i = \begin{bmatrix} m_i & m_i^* & ... & m_i^* \\ m_i^* & m_i & \ddots & m_i^* \\ m_i^* & \ddots & ... & m_i^* \\ m_i^* & m_i^* & m_i^* & m_i \end{bmatrix}, m_i, m_i^* > 0; m_i \neq m_j \forall i \neq j$$

Furthermore, for $2 \leq i \leq N$, H_i is a $|M_{i-1}| \times |M_i|$ matrix with equal positive real entries. If P is a critical point, then there is a equivalence class of cardinality S such that all elements are critical points

$$S = \frac{(\sum_{i=1}^{N} |M_i|)!}{\prod_{i=1}^{N} |M_i|!}$$

and the elements in the class is generated by the permutation of the diagonal entries and corresponding exchange of the rows and column. We will show a example next.

Proof.
Recalling the formulation (1) in the previous Lemma. We can obtain new matrices by reordering the position of the value a_i and b_j obtained from the existing critical point. The new matrices would still satisfy the Langrange multiplier equations because the permutation of sequences $(a_i)_{i \leq n}$ or $(b_i)_{i \leq n}$ is irrelevant to the equation. Therefore, the existence of a critical point implies the existence of the rest of the equivalence class. And the size of the equivalence class is determined by the amount of repeated value in the sequences $(a_i)_{i \leq n}$ and $(b_i)_{i \leq n}$.

□

Lemma 3.12 (Type 2 equivalence class). *In Type 1, we need the diagonal value of M_i to be distinct for different i. For an matrix of Type 2, we allow different M_i to hold the same diagonal value while satisfying the rest of the restrictions in Type 1.*

Example 3.13 (Type 1 equivalence class). $X(a, b)$ is a solution for $4 \times 4 U(a, b)$

$$X(a,b) = \frac{1}{32(a+b)} \begin{bmatrix} 4(a+b) & 4(a+b) & 4(a+b) & 4(a+b) \\ 4(a+b) & 12a & 6b & 6b \\ 4(a+b) & 6b & 3(2a+b) & 3(2a+b) \\ 4(a+b) & 6b & 3(2a+b) & 3(2a+b) \end{bmatrix}$$

then Y is also a critical point

$$Y(a,b) = \frac{1}{32(a+b)} \begin{bmatrix} 3(2a+b) & 6b & 4(a+b) & 3(2a+b) \\ 6b & 12a & 4(a+b) & 6b \\ 4(a+b) & 4(a+b) & 4(a+b) & 4(a+b) \\ 3(2a+b) & 6b & 4(a+b) & 3(2a+b) \end{bmatrix}$$

This matrix is obtained by exchanging entry e_{11} and e_{33}, and also the rows and columns,

$$y_{1i} = e_{i3}, y_{i3} = e_{1i}, 1 < i < 3$$

$$y_{j1} = e_{3j}, y_{3j} = e_{j1}, 1 < j < 3$$

Example 3.14 (Type 2 equivalence class)**.** X_4^4 is a critical point of Type 2 equivalence class.

$$X_4^4(a,b) = \frac{1}{4(2a+3b)} \begin{bmatrix} 2a+b & 2a+b & 2b & 2b \\ 2a+b & 2a+b & 2b & 2b \\ 2b & 2b & 2a+b & 2a+b \\ 2b & 2b & 2a+b & 2a+b \end{bmatrix}$$

Conjecture 3.15 (Equivalence classes grouping)**.** $\forall n \geq 2$, *the set of positive real critical points of* n*-dimensional* $U(a,b)$ *consists only of elements from Type* 1 *and Type* 2 *equivalence classes.*

Notation 3.16. For simplicity, we will let X_i^n denote a equivalence class, where i is the index and n is the dimension.

Example 3.17. Let $n = 3$, $a > b > 0$, consider the data matrix $U(a,b)$:

$$U(a,b) = \begin{bmatrix} a & b & b \\ b & a & b \\ b & b & a \end{bmatrix}$$

Under our model, $MLS(3) = 5$, where $ML(3) = 6$. The solutions fall into the following 2 symmetry classes, and each class has 3, 2 elements respectively. The first group is complete, and the second class has 1 element missing,

$$X_1^3(a,b) = \frac{1}{9(2a+b)} \begin{bmatrix} 2(2a+b) & 2(2a+b) & 2(2a+b) \\ 2(2a+b) & 8a & 4b \\ 2(2a+b) & 4b & 8a \end{bmatrix}$$

$$X_2^3(a,b) = \frac{1}{6(a+b)} \begin{bmatrix} 4a & 2b & 2b \\ 2b & 2a+b & 2a+b \\ 2b & 2a+b & 2a+b \end{bmatrix}$$

Example 3.18. Let $n = 4$, $a > b > 0$, consider the data matrix $U(a,b)$:

$$U(a,b) = \begin{bmatrix} a & b & b & b \\ b & a & b & b \\ b & b & a & b \\ b & b & b & a \end{bmatrix}$$

Under our model, we get 19 positive real solutions (out of 31 complex in total), where the true degree is 25. The solutions fall into the following 4 symmetry classes, and each class has 5, 10, 2, 2 elements, with 1, 2, 2, 1 terms missing, respectively.

$$X_1^4(a,b) = \frac{1}{8(2a+b)} \begin{bmatrix} 2a+b & 2a+b & 2a+b & 2a+b \\ 2a+b & 2a+b & 2a+b & 2a+b \\ 2a+b & 2a+b & 4a & 2b \\ 2a+b & 2a+b & 2b & 4a \end{bmatrix}$$

$$X_2^4(a,b) = \frac{1}{32(a+b)} \begin{bmatrix} 4(a+b) & 4(a+b) & 4(a+b) & 4(a+b) \\ 4(a+b) & 12a & 6b & 6b \\ 4(a+b) & 6b & 3(2a+b) & 3(2a+b) \\ 4(a+b) & 6b & 3(2a+b) & 3(2a+b) \end{bmatrix}$$

$$X_3^4(a,b) = \frac{1}{6(2a+3b)} \begin{bmatrix} 6a & 3b & 3b & 3b \\ 3b & 2(a+b) & 2(a+b) & 2(a+b) \\ 3b & 2(a+b) & 2(a+b) & 2(a+b) \\ 3b & 2(a+b) & 2(a+b) & 2(a+b) \end{bmatrix}$$

$$X_4^4(a,b) = \frac{1}{4(2a+3b)} \begin{bmatrix} 2a+b & 2b & 2a+b & 2b \\ 2b & 2a+b & 2b & 2a+b \\ 2a+b & 2b & 2a+b & 2b \\ 2b & 2a+b & 2b & 2a+b \end{bmatrix}$$

Theorem 3.19 (Recursively defined critical points). *Let $n \geq 3$, write X_i^n, $1 \leq i \leq N$ denotes all the critical points of $n \times n$ $U(a,b)$ data matrix. Then the $(n+1) \times (n+1)$ $U(a,b)$ sample matrix has at least the following N equivalent classes of critical points.*

$$X_m^{n+1}(a,b) = \frac{n^2}{(n+1)^2} \begin{bmatrix} \frac{2}{n^2} & \frac{2}{n^2} & \cdots & \frac{2}{n^2} & \frac{2}{n^2} \\ \frac{2}{n^2} & & & & \\ \vdots & & X_m^n(a,b) & & \\ \frac{2}{n^2} & & & & \\ \frac{2}{n^2} & & & & \end{bmatrix}, 1 \leq m \leq N$$

Proof. Scale the probability matrix $X_m^{n+1}(a,b)$ by $(n+1)^2$, we get:

$$(n+1)^2 X_m^{n+1}(a,b) = \begin{bmatrix} 2 & 2 & \cdots & 2 & 2 \\ 2 & & & & \\ \vdots & & n^2 X_m^n(a,b) & & \\ 2 & & & & \\ 2 & & & & \end{bmatrix}, 1 \leq i \leq N$$

Note that our matrices have sum of entries equals to 2, we scale the formulation in the previous lemma by 2. Let's express $(n+1)^2 X_m^{n+1}(a,b)$ in terms of $2n$ variables a_i, bj as

$$= \begin{bmatrix} 2+a_1b_1 & 2+a_1b_2 & \dots & 2+a_1b_n \\ 2+a_2b_1 & 2+a_2b_2 & \dots & 2+a_2b_n \\ \vdots & \vdots & \dots & \\ 2+a_nb_1 & 2+a_nb_2 & \dots & 2+a_nb_n \end{bmatrix}, 1 \le i \le N$$

Since X_m^n is a critical point, it satisfies the Lagrange equation (1) and (2) indicated in previous Lemma. Write each entry e_{ij} from X_m^n as $\frac{2+x_iyj}{n^2}$.
Therefore, $a_1 = b_1 = 0$; for $2 \le i \le n$, $x_i = a_{i+1}$, $y_i = b_{i+1}$ constitutes the unique solution for $X_m^{n+1}(a,b)$.
Now to verify the Lagrange condition for the $(n+1)\times(n+1)$ matrix.

$$(k-1)\frac{b_i}{1+a_ib_i} + \sum_{j=1}^{n+1} \frac{b_j}{1+a_ib_j} \tag{16}$$

if $i = 1$,

$$\frac{\partial f}{\partial a_i} = 0 + \sum_{j=1}^{n+1} b_j = 0$$

if $2 \le i \le n$,

$$\frac{\partial f}{\partial a_i} = (k-1)\frac{x_{i-1}}{1+x_{i-1}y_{i-1}} + \frac{b_1}{1+x_{i-1}b_1} + \sum_{j=2}^{n+1} \frac{y_{j-1}}{1+x_{i-1}y_{j-1}} = 0$$

We can apply the same procedure to $\frac{\partial f}{\partial b_i}$ to prove that

$$(k-1)\frac{a_i}{1+a_ib_i} + \sum_{j=1}^{n+1} \frac{a_j}{1+a_jb_i} = 0$$

Thus, we conclude $X_m^{n+1}(a,b)$ is a representative of a equivalent class of critical points of $(n+1)\times(n+1)U(a,b)$ matrix. □

Example 3.20 (solutions for n=3,4)**.**
Example 2.17 and 2.18 represent the equivalence classes for 3-dimension and 4-dimension cases, respectively.
As we can see, we can achieve the representative of the class of X_1^4 from the representative of X_1^3 by scaling the matrix by 9/16, and adding a row to the top and a column to the left with all entries equaling to $\frac{1}{8}$. The same rule also holds for X_2^4 and X_2^3.

$$X_j^4(a,b) = \frac{3^2}{4^2} \begin{bmatrix} \frac{1}{8} & \frac{1}{8} & \frac{1}{8} & \frac{1}{8} \\ \frac{1}{8} & & & \\ \frac{1}{8} & & X_j^3(a,b) & \\ \frac{1}{8} & & & \\ \frac{1}{8} & & & \end{bmatrix}, 1 \le j \le 2$$

3.3. **The missing solutions.** Recalling Conjecture 2.3, there is a bijection between the missing solutions under $n \times n$ matrix and the ML degree of $(n-1) \times (n-1)$ generic data. Since in previous sections we establish the explicit forms of the missing solutions for special data, we can discuss the relationship between the complete set of $(n-1) \times (n-1)$ critical points and these missing solutions.

Remark 3.21 (P_1 and the missing solutions)**.** The missing positive real solutions for $n \times n$ special case are the matrices with $rank(P_1) = 1$.
This follows from our model setting, where we set the variable P_1, the upper left 2×2 minor, as a rank 2 matrix. Although this set-up makes us miss some solutions, the omitted matrices can be easily recovered by other critical points in the same equivalence class.

Example 3.22 (Missing solutions)**.** The following 3 matrices are the complete solution elements inside a equivalence class for the 3×3 case. Note that under our model, we will have the last one missing because it has P_1, the upper left 2×2 minor having $rank(P_1) = 1$.

$$X^3_{2,1}(a,b) = \frac{1}{6(a+b)} \begin{bmatrix} 4a & 2b & 2b \\ 2b & 2a+b & 2a+b \\ 2b & 2a+b & 2a+b \end{bmatrix}$$

$$X^3_{2,2}(a,b) = \frac{1}{6(a+b)} \begin{bmatrix} 2a+b & 2b & 2a+b \\ 2b & 4a & 2b \\ 2a+b & 2a+b & 2a+b \end{bmatrix}$$

$$Missing := \frac{1}{6(a+b)} \begin{bmatrix} 2a+b & 2a+b & 2b \\ 2a+b & 2a+b & 2b \\ 2b & 2b & 4a \end{bmatrix}$$

Conjecture 3.23. *There exists a bijective correspondence between each missing solution obtained from our model for $n \times n$ special matrix and a critical point in the complete solution set for the $(n-1) \times (n-1)$ special data.*

The conjecture above is quite useful. If proved, it can be used to determine the lower bound of $ML(n)$.

3.4. **Formulation of the critical points for $U(a,b)$ matrix.** As mentioned in the earlier section, we construct our rank 2, $n \times n$ symmetric model P using a symmetric 2×2 matrix P_1, and a $(n-2) \times 2$ matrix L_1.

$$P = \begin{bmatrix} P_1 & P_1 L_1^T \\ L_1 P_1 & L_1 P_1 L_1^T \end{bmatrix}$$

We will see conjectures about the submatrix L_1.

Conjecture 3.24 (Type A and Type B equivalent classes)**.** *Each equivalent class with respect to the $n \times n$ $U(a,b)$ matrix is either a Type A or Type B class:*

Type A: An equivalent class such that all elements are recursively generated from the zeros of $(n-1) \times (n-1)$ case.

Type B: An equivalent class with all elements having L_1 consists only of 0 *and* 1*.*

Note: the set of Type A class and Type B class are mutually exclusive.

Example 3.25. Recalling the solutions for 4×4 case, we can see that classes X_1^4 and X_2^4 are the Type A points: they are generated from the 3×3 case. The critical points in classes X_3^4 and X_4^4 has the following values of L_1, thus they are the Type B classes.

$$\begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

3.5. **The global maximum.**

Conjecture 3.26 (Type of the global maximizer)**.** *The global maximizer is a critical point of Type B.*

In this section, we took a closer look at the positive real solutions for rank 2 special data input. In the future, we hope to prove the conjectures stated in this section, and to generalize our findings to higher rank and less special matrix.

References

[1] C. Améndola, N. Bliss, I. Burke, C. R. Gibbons, M. Helmer, S. Hoşten, E. D. Nash, J. I. Rodriguez, and D. Smolkin. The maximum likelihood degree of toric varieties. *Journal of Symbolic Computation*, 92:222–242, May 2019. 5

[2] D. R. Grayson and M. E. Stillman. Macaulay2, a software system for research in algebraic geometry, 2002. 2

[3] J. Hauenstein, J. I. Rodriguez, and B. Sturmfels. Maximum likelihood for matrices with rank constraints. *J. Algebr. Stat.*, 5(1):18–38, 2014. 9, 10

[4] S. Hoşten, A. Khetan, and B. Sturmfels. Solving the likelihood equations. *Found. Comput. Math.*, 5(4):389–407, 2005. 1

[5] B. Huber and B. Sturmfels. A polyhedral method for solving sparse polynomial systems. *Mathematics of Computation*, 64(212):1541–1555, 1995. 8

[6] A. N. Jensen. Tropical homotopy continuation, 2016. 8

[7] J. I. Rodriguez and B. Wang. The maximum likelihood degree of mixtures of independence models, 2015. 4

[8] M. Zhu, G. Jiang, and S. Gao. Solving the 100 swiss francs problem. *Math.Comput. Sci*, 5:195–207, 2011. 11

4. Appendix

Coefficient Matrix C_i of Prediction Matrix M_i	ML Degree
$C=\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$	6
$C=\begin{bmatrix} -2 & 1 & 1 \\ 1 & -2 & 1 \\ 1 & 1 & -2 \end{bmatrix}$	7
$C=\begin{bmatrix} 2 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}, C=\begin{bmatrix} 2 & 1 & 2 \\ 1 & 2 & 1 \\ 2 & 1 & 2 \end{bmatrix}, C=\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 2 \\ 1 & 2 & 2 \end{bmatrix}$	10
$\begin{bmatrix} -0.25 & 0.5 & 0.25 \\ 0.25 & 1 & -0.5 \\ 0.25 & -1 & -1 \end{bmatrix}, \begin{bmatrix} 2 & 2 & 2 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}, \begin{bmatrix} 2 & 1 & 1 \\ 2 & 2 & 2 \\ 1 & 1 & 2 \end{bmatrix}, \begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 2 & 2 & 2 \end{bmatrix}$	11
$\begin{bmatrix} 2 & 3 & 3 \\ -1 & -2 & 1 \\ 1 & -1 & 2 \end{bmatrix}, \begin{bmatrix} 2 & 3 & 3 \\ -1 & -2 & 1 \\ 1 & -1 & 2 \end{bmatrix}$	12
$\begin{bmatrix} -1 & -0.5 & 0.75 \\ -0.25 & 0.5 & -0.25 \\ -0.75 & 0.5 & -1 \end{bmatrix}, \begin{bmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{bmatrix}, \begin{bmatrix} n & 1 & 1 \\ 1 & n & 1 \\ 1 & 1 & n \end{bmatrix}$ for $3 \leq n \leq 10$	13
$\begin{bmatrix} 1.25 & -1 & -1.25 \\ -1 & -1.25 & 0.5 \\ 1.25 & -0.5 & -1.25 \end{bmatrix}, \begin{bmatrix} 1 & -1.25 & -1.25 \\ -1 & -1 & -0.5 \\ 0.5 & 0.25 & -0.75 \end{bmatrix}, \begin{bmatrix} 1 & 1 & -0.5 \\ 0.5 & 1 & 0.5 \\ 0.25 & -0.5 & 0.5 \end{bmatrix} ...$	15
$\begin{bmatrix} -1.25 & -0.5 & -1 \\ -1 & -1.25 & -1 \\ -1 & -0.5 & 0.5 \end{bmatrix}, \begin{bmatrix} 1.25 & -0.5 & 0.5 \\ 0.25 & 1.25 & -0.25 \\ 0.25 & 0.25 & -0.75 \end{bmatrix}, \begin{bmatrix} 0.5 & 1.25 & 0.25 \\ 1.25 & -1.25 & 1 \\ -0.25 & -1 & -0.75 \end{bmatrix} ...$	16
$\begin{bmatrix} -1 & 1 & 1 \\ 1.25 & 1.25 & 1.25 \\ -0.75 & 0.75 & -0.5 \end{bmatrix}, \begin{bmatrix} -1 & 0.25 & 0.25 \\ -1.25 & -0.25 & 0.25 \\ -1 & 1.25 & 1 \end{bmatrix}, \begin{bmatrix} -0.5 & -1.25 & 1 \\ -0.75 & 0.75 & 0.75 \\ -1 & 1.25 & -0.5 \end{bmatrix} ...$	19
$\begin{bmatrix} 0.75 & -0.5 & -0.75 \\ 1 & -0.75 & 0.25 \\ -1 & 0.75 & -1 \end{bmatrix}, \begin{bmatrix} 1.25 & 1.25 & -1.25 \\ -0.5 & -0.25 & 0.75 \\ 0.25 & -0.5 & -1 \end{bmatrix}, \begin{bmatrix} 1 & 1.25 & 1.25 \\ -1.25 & 1 & 1 \\ -1.25 & -0.25 & -1 \end{bmatrix}, ...$	20
$\begin{bmatrix} -3 & -3 & -3 \\ 2 & -3 & 2 \\ 2 & -3 & -3 \end{bmatrix}, \begin{bmatrix} 1 & -3 & -2 \\ 2 & -1 & 2 \\ -3 & -3 & 1 \end{bmatrix}, \begin{bmatrix} 2 & 3 & -2 \\ 1 & -1 & -1 \\ 3 & -3 & 2 \end{bmatrix}, ...$	21
$\begin{bmatrix} -0.75 & -0.75 & -0.25 \\ 0.75 & 0.25 & -0.25 \\ -1 & -0.5 & -0.25 \end{bmatrix}, \begin{bmatrix} 0.5 & -1 & -1 \\ 1 & -0.5 & 0.5 \\ -0.25 & 0.25 & -0.25 \end{bmatrix}, \begin{bmatrix} -0.25 & 0.5 & 1 \\ -0.75 & -1.25 & -0.75 \\ -0.75 & 0.5 & -0.25 \end{bmatrix}, ...$	22
$\begin{bmatrix} 0.5 & 0.25 & 0.25 \\ -1 & -0.75 & -0.5 \\ -0.5 & 1.25 & 1 \end{bmatrix} ...$	23

table 2.3

Below is the Julia code for computing the ML degrees of 4×4 symmetric matrices.

```
using DynamicPolynomials
using LinearAlgebra
using Random
using HomotopyContinuation
using PolynomialTestSystems
import Base.iterate,Base.length

@polyvar p11 p12 p13 p14 p22 p23 p24 p33 p34 p44 l1 l2
#P is the symmetric matrix
P = [[p11, p12, p13, p14] [p12, p22, p23, p24] [p13, p23, p33, p34] [p14,
p24, p34, p44]]
pFunc1 = P[1,1]+P[1,2]+P[1,3]+P[1,4]+P[2,1]+P[2,2]+P[2,3]+P[2,4]+P[3,1]+P[3,2]+
P[3,3]+P[3,4]+P[4,1]+P[4,2]+P[4,3]+P[4,4]-1

struct Combinations{T}
    itr::Vector{T}
    count::Int64
    itrsize::Int64
    function Combinations(itr::Vector{T},count::Int) where T
        new{T}(itr,Int64(count),length(itr))
    end
end

function iterate(c::Combinations,state::Int64=0)
    if state>=length(c)
        return nothing
    end
    indices=digits(state,base=c.itrsize,pad=c.count)
    [c.itr[i] for i in (indices .+1)],state+1
end

function length(c::Combinations)
    length(c.itr) ^ c.count
end

listOfPermutations = collect(Combinations([1,2],16))
M = rand(Float64,(4,4))

MLdegreeArray = zeros(0)
NumOfRealSolArray = zeros(0)
for i in 1:length(listOfPermutations)
    M[1,1] = listOfPermutations[i][1]
    M[1,2] = listOfPermutations[i][2]
```

```
M[1,3] = listOfPermutations[i][3]
M[1,4] = listOfPermutations[i][4]
M[2,1] = listOfPermutations[i][5]
M[2,2] = listOfPermutations[i][6]
M[2,3] = listOfPermutations[i][7]
M[2,4] = listOfPermutations[i][8]
M[3,1] = listOfPermutations[i][9]
M[3,2] = listOfPermutations[i][10]
M[3,3] = listOfPermutations[i][11]
M[3,4] = listOfPermutations[i][12]
M[4,1] = listOfPermutations[i][13]
M[4,2] = listOfPermutations[i][14]
M[4,3] = listOfPermutations[i][15]
M[4,4] = listOfPermutations[i][16]

M_random = [
[M[1,1]*P[1,1], M[1,2]*P[1,2], M[1,3]*P[1,3], M[1,4]*P[1,4]]
[M[2,1]*P[2,1], M[2,2]*P[2,2], M[2,3]*P[2,3], M[2,4]*P[2,4]]
[M[3,1]*P[3,1], M[3,2]*P[3,2], M[3,3]*P[3,3], M[3,4]*P[3,4]]
[M[4,1]*P[4,1], M[4,2]*P[4,2], M[4,3]*P[4,3], M[4,4]*P[4,4]]]
detM_random = det(M_random)
matrixOfDif_random = differentiate(det(M_random), P)

while true
    checkArray = zeros(0)
    for j in 1:3
        U = rand(Complex{Float64},(4,4)) #4x4 matrix of complex numbers
        f11 = U[1,1] + l1*P[1,1] + l2*matrixOfDif_random[1,1]*P[1,1]
        f12 = U[1,2] + l1*P[1,2] + l2*matrixOfDif_random[1,2]*P[1,2]
        f13 = U[1,3] + l1*P[1,3] + l2*matrixOfDif_random[1,3]*P[1,3]
        f14 = U[1,4] + l1*P[1,4] + l2*matrixOfDif_random[1,4]*P[1,4]
        f22 = U[2,2] + l1*P[2,2] + l2*matrixOfDif_random[2,2]*P[2,2]
        f23 = U[2,3] + l1*P[2,3] + l2*matrixOfDif_random[2,3]*P[2,3]
        f24 = U[2,4] + l1*P[2,4] + l2*matrixOfDif_random[2,4]*P[2,4]
        f33 = U[3,3] + l1*P[3,3] + l2*matrixOfDif_random[3,3]*P[3,3]
        f34 = U[3,4] + l1*P[3,4] + l2*matrixOfDif_random[3,4]*P[3,4]
        f44 = U[4,4] + l1*P[4,4] + l2*matrixOfDif_random[4,4]*P[4,4]

        result = solve([pFunc1, detM_random,f11,f12,f13,f14,f22,f23,
        f24,f33,f34,f44];start_system = :polyhedral)
        global result
        push!(checkArray, length(result))
    end

    if length(unique(checkArray)) == 1
```

```
            push!(MLdegreeArray, length(result))
            push!(NumOfRealSolArray, length(real_solutions(result)))
            break
        else
            continue
        end
    end
end
```

The Hessian matrix computed in the examples above were computed using the following code:

```
-- Note: This method works any input function which is differentiable.
-- Note 2: The order of the values in the declared ring R changes the
order of the Hessian matrix output.
Hessian := (f) -> (
    J = flatten entries jacobian ideal f;
    output = matrix {flatten entries jacobian ideal J#0};
    for i from 1 to (length J - 1) do (
r = transpose jacobian ideal J#i;
output = output || r;
    );
    return output;
)

-- Save the output into a variable
hessL = Hessian (L)
```