Applications of Computational Commutative Algebra in Statistics

Jocelyn Baker
Cidmarie Odiott
Alex Simao
PREMUR 2007

August 6, 2007

Abstract

Recently, the application of computational commutative algebra and algebraic geometry in statistics has played an important role in bioinformatics and genomics. One application is through *maximum likelihood estimation* of parameters in discrete probability models. One of the most basic, but most important, questions is how many critical points the likelihood function has, and whether there is an efficient way of counting them. The focus of this project was centered around an example of a mixture model with three unknown parameters. Through this project we were able to apply techniques of computational commutative algebra such as Gröbner bases with those of algebraic geometry to form a procedure to solve the maximum likelihood equations, and then developed a symbolic method to count the number of critical points.

1 Our Mixing Problem

A large part of this paper is focused on solving a mixing problem dealing with the probability of picking and flipping two coins in a series of four flips. This mixing problem was taken directly from [2].

Suppose a man has two biased coins, one in each of his sleeves, and he picks the coin to be used at random with probabilities π and $1 - \pi$.

The model of the above scenario is the mixture of a pair of four-times repeated Bernoulli trials. The mixing parameter π is the probability that the man picks the coin in his left sleeve. The bias of the left coin is s, and the bias of the right coin is t. The model stipulates that the probabilities of the five outcomes are given by the parametric equations:

$$\begin{aligned}
p_0 &= \pi(1-s)^4 + (1-\pi)(1-t)^4,\\
p_1 &= 4\pi s(1-s)^3 + 4(1-\pi)t(1-t)^3,\\
p_2 &= 6\pi s^2(1-s)^2 + 6(1-\pi)t^2(1-t)^2,\\
p_3 &= 4\pi s^3(1-s) + 4(1-\pi)t^3(1-t),\\
p_4 &= \pi s^4 + (1-\pi)t^4.
\end{aligned}$$

Let $\vec{u} = (u_0, u_1, u_2, u_3, u_4) \in \mathbb{N}^5$, where the u_i is the number of trials that had i heads as the outcome. The overall goal, therefore, is to figure out the best way to predict values for π, s, and t given a weight vector $\vec{u}$. The best way of doing this is by finding the maximum of the likelihood function. The likelihood function can be defined as the following:

Definition 1 *Given any weight vector $\vec{u} = (u_0, u_1, u_2, u_3, u_4)$, and its corresponding $\vec{p} = (p_0, p_1, p_2, p_3, p_4)$, let the likelihood function L be defined as,*

$$L = p_0^{u_0} p_1^{u_1} p_2^{u_2} p_3^{u_3} p_4^{u_4} \tag{1}$$

Then the maximum of L is simply the critical point $\vec{p}$ that results in the maximum value for L.

After successfully setting up the parametric equations that represent this mixing model, the computations involved using the parametric equations were too large for Mathematica to complete. The system of equations contained eight different variables, slowing down the process to the point where a Gröbner basis for the parametric model could not be computed. In order to alleviate the problem, it is beneficial to look at the elimination ideal $J = I \cap \mathbb{C}[p_0, ..., p_4]$, where I is the ideal generated by the system of parametric equations. J is generated by two polynomials g_1 and g_2 given below. This turns the focus on $p_0, \ldots, p_4$, making the computations more manageable. We then must solve the following constrained optimization problem.

Constrained Optimization Problem

Maximize (1) subject to

$$g_1 = p_0 + p_1 + p_2 + p_3 + p_4 - 1 = 0$$

and

$$g_2 = \det(P) = 0,$$

where P is the implicit matrix that models the parametric equations:

$$P = \left(\begin{array}{ccc} 12p_0 & 3p_1 & 2p_2 \\ 3p_1 & 2p_2 & 3p_3 \\ 2p_2 & 3p_3 & 12p_4 \end{array} \right).$$

We use the method of Lagrange Multipliers to find the critical points, while using $\ln L$ to simplify the system of equations in the following form:

$$0 = u_i + p_i\lambda_1\frac{\partial g_1}{\partial p_i} + p_i\lambda_2\frac{\partial g_2}{\partial p_i},\ i = 0, ..., 4. \tag{2}$$

By computing the Gröbner Basis of the elimination ideal, it is possible to eliminate λ_1 and λ_2, and find critical points of the form $\vec{p} = (p_0, p_1, p_2, p_3, p_4)$ that solve (2). The only critical points of interest are those in which all p_i's are positive, real numbers in the range $[0, 1]$. For any $\vec{u}$, if $\vec{p}$ is a solution of (2), then $\vec{p}$ is also a solution of (1). Thus $\vec{p}$ is the critical point of L.

After setting up this new method of finding the critical points of (1), it was worthwhile to write a package in Mathematica that used this new implicit model method. The package *Compute* takes in any $\vec{u}$ as input and returns all real critical points in the range $[0, 1]$. This code can be used to efficiently look at how different weight vectors affect the critical points of the likelihood equation. This package led to the generation of several theorems regarding the critical points of the likelihood equation which are to follow.

2 Theorems Related to the Critical Points of the Likelihood Equation

Let $L = p_0^{u_0}p_1^{u_1}...p_n^{u_n}$ be the likelihood function. It is necessary to solve the constrained optimization problem for L with the two constraints g_1 and g_2 where

$$\begin{aligned} g_1 &= p_0 + p_1 + \ldots + p_n - 1 \\ g_2 &= \det(P). \end{aligned}$$

Theorem 1 (Effect of Scalar Multiplication) *For the constrained optimization problem for L as above, the critical points derived from the weight vector $\vec{u}$ are the same as the critical points for the weight vector $c\vec{u}$, where $c \in \mathbb{N}$. Moreover, If $\vec{p}$ gives the maximum likelihood for L, then $\vec{p}$ gives the maximum likelihood for L^c.*

Proof. Use the method of Lagrange Multipliers to solve, while using $\ln L$ to simplify the system of equations. The application of these techniques lead to a system of equations in the following form:

$$0 = u_i \frac{1}{p_i} + \lambda_1 \frac{\partial g_1}{\partial p_i} + \lambda_2 \frac{\partial g_2}{\partial p_i} \qquad \forall i \in [0, n] \tag{3}$$

and for some $\lambda_1, \lambda_2 \in \mathbb{R}$.

It is equivalent to say that a solution to (3) is a critical point of L. This implies $\vec{p}$ solves (3).

Consider the same problem, replacing $\vec{u}$ by $c\vec{u}$, where $c \in \mathbb{N}$. So

$$0 = c\left(u_i \frac{1}{p_i} + \lambda_1 \frac{\partial g_1}{\partial p_i} + \lambda_2 \frac{\partial g_2}{\partial p_i}\right) \qquad \forall i \in [0, n]. \tag{4}$$

Since $u_i, \lambda_1, \lambda_2 \in \mathbb{R}$, $\lambda_i = c\overline{\lambda_i}$ for some $\overline{\lambda_i}$, so (4) becomes

$$0 = c\left(u_i \frac{1}{p_i} + \overline{\lambda_1} \frac{\partial g_1}{\partial p_i} + \overline{\lambda_2} \frac{\partial g_2}{\partial p_i}\right) \qquad \forall i \in [0, n]. \tag{5}$$

Furthermore, since $\vec{p}$ is a solution of (1), $\vec{p}$ is a solution of (2), which implies $\vec{p}$ is a critical point of L^c. Now assume $\vec{p}$ is a critical point of L such that $\vec{p}$ makes L a maximum. Consider L^c. By the algebraic property that $x > y \Rightarrow x^c > y^c$ and the assumption that $\vec{p}$ is a critical point of L such that $\vec{p}$ maximizes L, it is clear that $\vec{p}$ maximizes L^c. □

Theorem 2 *For any $\vec{u}$ in the form $(u_0, u_1, u_2, u_3, u_4)$ with critical points $\vec{p_i}$ in the form $(p_0, p_1, p_2, p_3, p_4)$, if $\vec{u'} = (u_4, u_3, u_2, u_1, u_0)$ ($\vec{u}$ in the reverse order), then $\vec{u'}$ has critical points of the form $\vec{p_i'} = (p_4, p_3, p_2, p_1, p_0)$ for all critical points of $\vec{u}$.*

Proof. Consider u_0, the number of times zero heads occur in a series of four coin flips. If $\vec{u}$ is inverted, then u_0 becomes the last element of $\vec{u'}$. So in the new case, u_0 now shows the number of times four heads in a row occur. There is another way to look at the problem. Instead of looking at $\vec{u'}$ as being $\vec{u}$ inverted, look at $\vec{u'}$ as being the vector for the corresponding 'tail' values for the likelihood function. So u_0 not only displays how many times zero heads appeared, it also shows how many times four tails appeared. This correspondance with the entire vector can be continued, resulting in the same likelihood function, but this time for tails. This problem can also be thought about in terms of the system of parametric equations seen earlier.

Again the idea of corresponding 'tail' values can be used to see how it affects the system of equations. Recall, initial system:

$$\begin{aligned}
p_0 =& \quad \pi(1-s)^4 + (1-\pi)t^4 \\
p_1 =& \quad 4\pi s(1-s)^3 + 4(1-\pi)t(1-t)^3 \\
p_2 =& \quad 6\pi s^2(1-s)^2 + 6(1-\pi)t^2(1-t)^2 \\
p_3 =& \quad 4\pi s^3(1-s) + 4(1-\pi)t^3(1-t) \\
p_4 =& \quad \pi s^4 + (1-\pi)t^4.
\end{aligned}$$

Now to switch these equations to be in terms of tails, replace s by $1-s$ and replace t by $1-t$. The resulting equations are

$$\begin{aligned}
p_0' =& \quad \pi s^4 + (1-\pi)t^4 = p_4 \\
p_1' =& \quad 4\pi s^3(1-s) + 4(1-\pi)t^3(1-t) = p_3 \\
p_2' =& \quad 6(1-\pi)t^2(1-t)^2 + 6\pi s^2(1-s)^2 = p_2 \\
p_3' =& \quad 4\pi s(1-s)^3 + 4(1-\pi)t(1-t)^3 = p_1 \\
p_4' =& \quad \pi(1-s)^4 + (1-\pi)t^4 = p_0.
\end{aligned}$$

This results in the same critical points, which can then be inverted to get them in terms of 'heads'. □

Theorem 3 *If $\vec{u}$ has the form $(n,0,0,0,0)$ where $n \in \mathbb{N}$, then L has exactly one critical point $\vec{p} = (1,0,0,0,0)$.*

Proof. Using the previously outlined procedure to determine the critical points given any $\vec{u}$, compute a Gröbner Basis for the ideal generated by polynomials in the Lagrange Multiplier equations, eliminating λ_1 and λ_2. Because Theorem 1 has been proved, it is sufficient to show that this theorem works for $\vec{u'} = (1,0,0,0,0)$. So using the Gröbner basis, create a Gröbner basis for the elimination ideal that contains only the variable p_0, which gives

$$1 + 14p_0 + 44p_0^2 - 38p_0^3 - 45p_0^4 + 24p_0^5. \tag{6}$$

Solving for all the roots of (6) gives all possible values of p_0. The only solutions for p_0 that are of interest are positive, real, solutions that are less than or equal to one. Those conditions follow as a direct result of p_0 itself being a probability. Upon solving, we get the following values for p_0:

$$p_0 = -1., -0.154701, -0.125, 1., 2.1547.$$

The only solution that satisfies the needed conditions is when $p_0 = 1$. This automatically implies that the rest of the p values must be zero. Scalar multipication by n can then be applied to $\vec{u'}$ giving $\vec{p}$ as the only solution for our $\vec{u}$. □

Theorem 4 *Given a $\vec{u} = (u_0, u_1, u_2, u_1, u_0)$, then there always exists a critical point $\vec{p}$, such that $\vec{p} = (p_0, p_1, p_2, p_1, p_0)$.*

Proof. Given the symmetric weight vector $\vec{u}$, the system of Lagrange Multiplier equations can be constructed in the form of:

$$-\lambda_1 p_0 - \lambda_2 p_0(-108p_3^2 + 288p_2p_4) + u_0 \tag{7}$$

$$-\lambda_1 p_1 - \lambda_2 p_1(36p_2p_3 - 216p_1p_4) + u_1 \tag{8}$$

$$-\lambda_1 p_2 - \lambda_2 p_2(-24p_2^2 + 36p_1p_3 + 288p_0p_4) + u_2 \tag{9}$$

$$-\lambda_1 p_3 - \lambda_2 p_3(36p_1p_2 - 216p_0p_3) + u_1 \tag{10}$$

$$-\lambda_1 p_4 - \lambda_2 p_4(-108p_1^2 + 288p_0p_2) + u_0 \tag{11}$$

Subtract equations (7) - (11) and (8) - (10). Then remaining are the three following equations which only contain u_2:

$$\begin{gathered} \lambda_1 p_0 - 108\lambda_2 p_0 p_3^2 - \lambda_1 p_4 + 108\lambda_2 p_1^2 p_4 \\ \lambda_1 p_1 - \lambda_1 p_3 + 216\lambda_2 p_0 p_3^2 - 216\lambda_2 p_1^2 p_4 \\ -\lambda_1 p_2 - \lambda_2 p_2(-24p_2^2 + 36p_1p_3 + 288p_0p_4) + u_2 \end{gathered}$$

Evaluate the Gröbner basis for elimination ideal to eliminate λ_1 and λ_2, to get the following polynomial:

$$\begin{gathered} 2p_0^2p_3^2u_2 + p_0p_1p_3^2u_2 - p_0p_3^3u_2 - 2p_0p_1^2p_4u_2 - \\ p_1^3p_4u_2 + p_1^2p_3p_4u_2 - 2p_0p_3^2p_4u_2 + 2p_1^2p_4^2u_2. \end{gathered}$$

Next it is necessary to find the roots of this equation in order to find the critical points for $\vec{u}$. First, factor out u_2, which leaves:

$$u_2(2p_0^2p_3^2 + p_0p_1p_3^2 - p_0p_3^3 - 2p_0p_1^2p_4 - p_1^3p_4 + p_1^2p_3p_4 - 2p_0p_3^2p_4 + 2p_1^2p_4^2)$$

So if $u_2 = 0$ then there is always a symmetric solution. Now, look at the case where $u_2 \neq 0$ and see if a symmetric $\vec{p}$ is a root of the equation.

$$2p_0^2p_3^2 + p_0p_1p_3^2 - p_0p_3^3 - 2p_0p_1^2p_4 - p_1^3p_4 + p_1^2p_3p_4 - 2p_0p_3^2p_4 + 2p_1^2p_4^2. \tag{12}$$

If $p_0 = p_4$,$p_1 = p_3$, then (12) is always equal to zero. Thus there is a $\vec{p} = (p_0, p_1, p_2, p_1, p_0)$, a symmetric critical point, for any symmetric $\vec{u}$. □

Theorem 5 *If $\vec{u} = (m, n, 0, 0, 0)$ for $m, n \in \mathbb{N}$, then the critical point $\vec{p} = (\frac{m}{n+m}, \frac{n}{n+m}, 0, 0, 0)$ results in a maximum for L.*

Proof. First to prove a critical point of the form $\vec{p} = (\frac{m}{m+n}, \frac{n}{m+n}, 0, 0, 0)$ exists, use the method of Lagrange Multipliers to compute the Gröbner basis of the ideal generated by the polynomials from the Lagrange Multiplier equations. Substituting into the Gröbner basis $p_0 = \frac{m}{m+n}, p_1 = \frac{n}{m+n}$, $p_2 = 0$, $p_3 = 0$ and $p_4 = 0$ results in zeros for all equations of the Gröbner basis. This shows that there is always a solution of the form $\vec{p} = (\frac{m}{m+n}, \frac{n}{m+n}, 0, 0, 0)$.

Now it is necessary show that $\vec{p} = (\frac{m}{m+n}, \frac{n}{m+n}, 0, 0, 0)$ is always the maximum of L. Recall that $p_0 + p_1 + p_2 + p_3 + p_4 = 1$ so either $p_0 + p_1 = 1$ if $p_2 = p_3 = p_4 = 0$ or $p_0 + p_1 < 1$ if any $p_2, p_3, p_4 \neq 0$. Since $\vec{u} = (m, n, 0, 0, 0)$, the values of p_2, p_3, p_4 have no effect on the likelihood function, $L = p_0^m p_1^n p_2^0 p_3^0 p_4^0$ because they are raised to the zero power and therefore $p_2^0 p_3^0 p_4^0 = 1$. Therefore, the best way to maximize the likelihood equation is to let $p_2 = p_3 = p_4 = 0$ so that the largest possible values can be given to p_0 and p_1 to maximize L. So $p_0 + p_1 = 1$ since we have let $p_2 = p_3 = p_4 = 0$. Again using the method of Lagrange Multipliers, compute the Gröbner basis of the ideal generated by (2). This time, substitute in $p_2 = p_3 = p_4 = 0$. This results in a Gröbner basis of two equations

$$\begin{aligned} 0 &= -1 + p_0 + p_1 \\ 0 &= -n + mp_1 + np_1 = 0. \end{aligned}$$

Solve for p_0 and p_1 to find that $p_0 = \frac{m}{m+n}$ and $p_1 = \frac{n}{m+n}$. Thus it is true that $\vec{p} = (\frac{m}{n+m}, \frac{n}{m+n}, 0, 0, 0)$ is always the maximum of L. □

3 A Better Way to Count

Although the computations for the critical points are successful, they rely heavily upon decimal approximations when computing roots of the generators for each univariate elimination ideal. It would be very beneficial to have a symbolic way of determining the number of critical points for any given weight vector $\vec{u}$. The process of symbolically counting critical points involves many steps, but luckily, it starts with the important fact that we are only concerned about $\mathbf{V}(I)$ when it is finite. From here, a few additional facts can be applied to set up our case.

3.1 Application of Ring Theory

Knowing that it is only important to consider $\mathbf{V}(I)$ when it is finite, the following statement can be applied.

Let G be a Gröbner basis for I with respect to any monomial order, then:

Theorem 6 ([3], Theorem 6 of Chapter 5, §3) $\mathbf{V}(I)$ *is finite in* $\mathbb{C}^n$ *if and only if for each* i*, there exists* $g \in G$ *such that* $LT(g) = x_i^{m_i}$ *for some* $m_i \geq 0$.

Thus the number of x^α that are not in $\langle LT(I) \rangle$ is in fact finite. Next it is necessary to look at the vector space spanned by those x^α. Looking at a Gröbner basis $G = \{g_1, \ldots, g_n\}$ of the ideal I, note that any remainder upon division by G results in polynomials that only contain x^α's. Define the remainder upon division by G on a polynomial f to be $\overline{f}^G$. It can be seen that the set of $\overline{f}^G$ has naturally defined addition and multiplication operations, in which remainders are the result.

The above remainders are closely related to the idea of quotient rings from abstract algebra. Recall that for a quotient ring $k[x_1, \ldots, x_n]/I$, the coset, $[f]$, for some $f \in k[x_1, \ldots, x_n]$ can be defined as:

$$[f] = f + I = \{f + h : h \in I\}.$$

From the idea of cosets, a very important one-to-one correspondence between the cosets and the remainders can be seen. This means that one can think of $\overline{f}^G$ as a representation of the coset $[f]$. Again looking at the quotient rings, recall that $k[x_1, \ldots, x_n]/I$ also has the structure of a vector space, or is an algebra.

It follows that one can consider the set of x^α as a basis of the algebra. So it is possible to denote the basis B as

$$B = \{x^\alpha : x^\alpha \notin \langle LT(I) \rangle\}.$$

Given any Gröbner basis G, the first step is to find the basis of non-leading terms, B. Once this basis exists, it is easy to compute a multiplication table for the elements of B. Before continuing, look at the following example from [4].

Example Let G be the Gröbner basis, and let B be the basis formed by the non-leading term monomials of G, such that

$$\begin{aligned} G &= \{x^2 + 3xy/2 + y^2/2 - 3x/2 - 3y/2, xy^2 - x, y^3 - y\} \\ B &= \{1, x, y, xy, y^2\}. \end{aligned}$$

Then the multiplication table that results for the elements of the basis B is:

$\cdot$	1	x	y	xy	y^2
1	1	x	y	xy	y^2
x	x	α	xy	β	x
y	y	xy	y^2	x	y
xy	xy	β	x	α	xy
y^2	y^2	x	y	xy	y^2

where

$$\begin{aligned} \alpha &= -3xy/2 - y^2/2 + 3x/2 + 3y/2 \\ \beta &= 3xy/2 + 3y^3/2 - 3x/3 - y/2. \end{aligned}$$

3.2 Multiplication Matrices and S_h Function

Since a one to one correspondence exists between cosets and the remainders upon division, the multiplication table can be used to form multiplication matrices, m_f for $f \in B$. Each matrix is a n by n matrix, where n is the number of elements of the basis B. The entries of each matrix are the coefficient values from the entries of the multiplication table we developed earlier. Now to continue the above example:

Example cont. Using G and B from before, a multiplication matrix can be constructed for x, m_x, by taking the coefficients from the second row (or column). This gives:

$$m_x = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & 3/2 & 0 & -3/2 & 1 \\ 0 & 3/2 & 0 & -1/2 & 0 \\ 0 & -3/2 & 1 & 3/2 & 0 \\ 0 & -1/2 & 0 & 3/2 & 0 \end{pmatrix}.$$

Now given any two multiplication matrices, define the function S as

$$S(f, g) = Tr(m_f \cdot m_g) = Tr(m_{fg})$$

Next, define a different polynomial $h \in k[x_1, \ldots, x_n]$, and note that S can be modified to take into account h

$$S_h(f, g) = Tr(m_{hf} \cdot m_g) = Tr(m_{hfg}).$$

Since $S_h(f, g)$ is simply the trace of a matrix, it is always a numerical value, so it is possible to denote S_h as a matrix whose entries are $S_h(f, g)$ for all combinations of $f, g \in B$. Furthermore, this implies that the matrix form of S_h is symmetric. This is the final condition needed to apply the following theorem.

Theorem 7 ([4], Chapter 2, Theorem (5.2)) *Let I be a zero-dimensional ideal generated by polynomials in $k[x_1, \ldots, x_n]$ where $k \subset \mathbb{R}$, so that $\boldsymbol{V}(I) \subset \mathbb{C}^n$ is finite. Then, for $h \in k[x_1, \ldots, x_n]$, the signature ($\sigma$) and rank ($\rho$) of the bilinear form S_h satisfy:*

$$\begin{aligned} \sigma(S_h) &= \#\{a \in \boldsymbol{V}(I) \cap \mathbb{R}^n : h(a) > 0\} - \#\{a \in \boldsymbol{V}(I) \cap \mathbb{R}^n : h(a) < 0\} \\ \rho(S_h) &= \#\{a \in \boldsymbol{V}(I) : h(a) \neq 0\}. \end{aligned}$$

So given h, one can determine the number of points that fall within a specific region. The next consideration is how to determine a value for h or a system of h_i's that would isolate the regions containing the critical points of interest.

4 Finding h

The goal is to construct h in such a way that the region that satisfies the conditions of the original mixing problem (the coin flipping problem)is isolated. Since the original mixing problem is in 5-dimensions, it is a good idea to first look at a simplified version of the problem in 2-dimensions.

4.1 Two-Dimensional Case

Let the conditions of the two-dimensional problem be that $0 \leq x \leq 1$ and $0 \leq y \leq 1$. To satisfy this condition, let $h_1 = x(x-1)$ and $h_2 = y(y-1)$. Define C_{**} to be the number of solutions within each region where $* = 0, +, -$. For example,

$$C_{--} = \#\{a \in \mathbf{V}(I) \cap \mathbb{R}^n : h_1(a) \leq 0 \text{ and } h_2(a) \leq 0\}.$$

In this case, it is desired to find the number of solutions in the area where $0 \leq x \leq 1$ and $0 \leq y \leq 1$, so one should solve for C_{--}, C_{-0}, C_{0-}, C_{00}. In

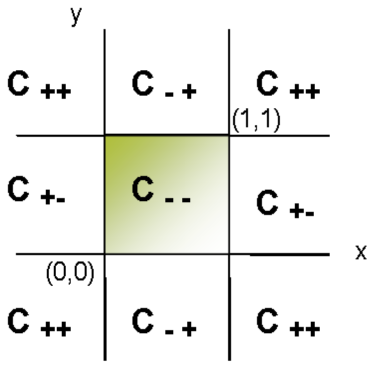

Figure 1: 2-D Regions

general, it is possible to find any C_{**} in terms of the signatures because recall that $\sigma(S_h) = (\#\text{ of times } \; h(a) > 0) - (\#\text{ of times } \; h(a) < 0)$. Using this theorem, the signatures can be written in terms of the C_{**}'s.

$$\begin{aligned}
\sigma(S_1) &= \sum_* C_{**} \\
\sigma(S_{h_1}) &= \sum_* C_{+*} - \sum_* C_{-*} \\
\sigma(S_{h_2}) &= \sum_* C_{*+} - \sum_* C_{*-} \\
\sigma(S_{h_1 h_2}) &= C_{++} + C_{--} - C_{+-} - C_{-+} \\
\sigma(S_{h_1^2}) &= \sum_* C_{+*} + \sum_* C_{-*} \\
\sigma(S_{h_2^2}) &= \sum_* C_{*+} + \sum_* C_{*-} \\
\sigma(S_{h_1^2 h_2}) &= (C_{++} + C_{-+}) - (C_{--} + C_{+-}) \\
\sigma(S_{h_1 h_2^2}) &= (C_{++} + C_{+-}) - (C_{--} + C_{-+}) \\
\sigma(S_{h_1^2 h_2^2}) &= C_{++} + C_{--} + C_{-+} + C_{+-}
\end{aligned}$$

One method to solve for the C_{**} is to manipulate the above signatures to isolate the C_{**} of interest. For example, $\sigma(S_{h_1^2 h_2^2}) + \sigma(S_{h_1 h_2}) - \sigma(S_{h_1^2 h_2}) - \sigma(S_{h_1 h_2^2})$ equals

$$2C_{++} + 2C_{--} - 2C_{++} + 2C_{--} = 4C_{--}.$$

However, there is a more efficient way to find any C_{**}. Let M be a matrix whose entries represent the coefficients of the signatures of all possible combinations of h_1 and h_2, then it is true that

$$\begin{aligned} M\vec{C} &= \sigma(S_{h_1^* h_2^*}) \\ \vec{C} &= M^{-1}\sigma(S_{h_1^* h_2^*}) \end{aligned}$$

$$\begin{pmatrix} C_{--} \\ \vdots \\ C_{++} \end{pmatrix} = \left(M^{-1}\right) \begin{pmatrix} \sigma(S_1) \\ \vdots \\ \sigma(S_{h_1^2 h_2^2}) \end{pmatrix}.$$

So the inverse of the matrix M can be used to find any C_{**}. Using the depiction of the signatures as the combination of the C_{**}'s, for the two dimensional case the coefficient matrix M can be developed of the following form.

$$M_{2D} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ -1 & 0 & 1 & -1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & -1 & -1 & -1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & 1 & -1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & -1 & 0 & -1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$

4.2 Three-Dimensions and Generalization

The above example can be extended to the three-dimensional case and then a generalization can be used to extend to the five-dimensional case which corresponds to the original mixing problem. Let the new system of h_i's for the three-demensional case be $h_1 = x(x-1)$, $h_2 = y(y-1)$ and $h_3 = z(z-1)$. Much like in the two-dimensional case, we are interested in solving for C_{---}, which can be expressed in terms of the signatures.

Using the same approach as before, a matrix can be formed by the coefficients (-1, 0, or 1) that describes the signatures as polynomials of the different C_{***}s. Again, the matrix is invertible and any C_{***} can be written in terms of different signature combinations. The most valuable result from the three-dimensional example is a generalized way to form the coefficient matrix.

In the three-dimensional case, the coefficient matrix is a 27 by 27 matrix, and by arranging the rows and columns, it can be written in the form:

$$M_{3D} = \begin{pmatrix} M_{2D} & M_{2D} & M_{2D} \\ 0 & M_{2D} & -M_{2D} \\ 0 & M_{2D} & M_{2D} \end{pmatrix},$$

where "0" is a nine by nine matrix containing all zeros.

This arrangement led to the realization of the proper ordering of the rows and columns to write the coefficient matrix for any dimension $n \geq 3$.

Theorem 8 (Coefficient Matrix) *For any dimension $n \geq 3$ the coefficient matrix M_{nD} for all possible signatures and $C_{*\cdots*}$'s will always have the form of:*

$$M_{nD} = \begin{pmatrix} M_{(n-1)D} & M_{(n-1)D} & M_{(n-1)D} \\ 0 & M_{(n-1)D} & -M_{(n-1)D} \\ 0 & M_{(n-1)D} & M_{(n-1)D} \end{pmatrix}$$

Proof It is easy to achieve a matrix of this form for n-dimensions provided that one maintains the same ordering for the rows and columns. In the case of the $C_{*\cdots*}$, the columns, one must maintain the ordering that $0 < + < -$. As for the rows, one always needs the ordering of $h_n < h_{n-1} < \cdots < h_1$ and where the powers of the h_i's go from zero to two. When this ordering is maintained the coefficient matrix will always have the form of M_{nD}. □

Now that the coefficient matrix has been generalized, the next step is to look at the signature combinations that led to the desired $C_{*\cdots*}$ to find a generalization.

5 h-Polynomial Equations

Using the matrices developed for the 2-D and 3-D cases, and the proper ordering of the $C_{*\cdots*}$ and signatures, a trend appears amongst the $C_{*\cdots*}$ of importance. The number of critical points that are not on the boundary are of the most interest (those on the boundary are easy enough to find), so consider the example where all the h_i's are positive. Using the matrix, each $C_{+\cdots+}$ can be expressed as a linear combination of the different signature combinations. This can be represented as a polynomial of the different combinations of the h_i's needed. So for example, for the 2-D and 3-D cases, the polynomials are:

$$\begin{aligned}
C_{++} &= \frac{1}{4}\left(\sigma(S_{h_1h_2}) + \sigma(S_{h_1^2h_2}) + \sigma(S_{h_1h_2^2}) + \sigma(S_{h_1^2h_2^2})\right) \\
C_{+++} &= \frac{1}{8}\left(\sigma(S_{h_1h_2h_3}) + \sigma(S_{h_1^2h_2h_3}) + \sigma(S_{h_1h_2^2h_3}) + \sigma(S_{h_1^2h_2^2h_3}) + \right. \\
&\quad \left. \sigma(S_{h_1h_2h_3^2}) + \sigma(S_{h_1h_2^2h_3^2}) + \sigma(S_{h_1^2h_2h_3^2}) + \sigma(S_{h_1^2h_2^2h_3^2})\right)
\end{aligned}$$

From those two examples and some simple factorization, we concluded that for a system with $n \in \mathbb{N}$ number of h_i's, then one can always write out the h_i pairings needed as the terms of the polynomial H:

$$H = (h_1 + h_1^2)(h_2 + h_2^2)...(h_n + h_n^2)$$

So in other words, $C_{+\cdots+}$ can always be expressed as a polynomial of signatures of the S matrices for all combination of h_i where the power of h_i is at least one (no zero powers). Then $C_{+\cdots+}$ will always have the form:

$$C_{+\cdots+} = \frac{1}{2^n}\left(\sigma(S_{h_1\cdots h_n}) + \sigma(S_{h_1^2\cdots h_n}) + ... + \sigma(S_{h_1^2\cdots h_n^2})\right) \tag{13}$$

Before this can be used, it must be proven that this pattern will always simplify to $C_{+...+}$ for any n.

Theorem 9 *For any $n \in \mathbb{N}$ and any $h_1, \ldots, h_n$, the number of critical points $C_{+\cdots+}$ is given by the equation (13).*

Proof To see that the polynomial equation always simplifies to just the $C_{+\cdots+}$ region, refer to the 2-D case. Recall that the polynomial equation for 2-D is,

$$C_{++} = \frac{1}{4}\left(\sigma(S_{h_1h_2}) + \sigma(S_{h_1^2h_2}) + \sigma(S_{h_1h_2^2}) + \sigma(S_{h_1^2h_2^2})\right)$$

To see that this works, evaluate the signatures for each monomial pairing in the equation. This results in:

$$\begin{aligned}
&(C_{++} + C_{--} - C_{+-} - C_{-+}) + (C_{++} + C_{-+} - C_{+-} - C_{--}) + \\
&\quad (C_{++} + C_{+-} - C_{-+} - C_{--}) + (C_{++} + C_{-+} + C_{+-} + C_{--})
\end{aligned}$$

	C_{-----}	C_{----+}			
None	$-$	$+$	1	$+$	$-$
5	$+$	$+$	$1, 5$	$-$	$-$
4	$+$	$-$	$1, 4$	$-$	$+$
$4, 5$	$-$	$-$	$1, 4, 5$	$+$	$+$
3	$+$	$-$	$1, 3$	$-$	$+$
$3, 5$	$-$	$-$	$1, 3, 5$	$+$	$+$
$3, 4$	$-$	$+$	$1, 3, 4$	$-$	$+$
$3, 4, 5$	$+$	$+$	$1, 3, 4, 5$	$-$	$-$
2	$+$	$-$	$1, 2$	$-$	$+$
$2, 5$	$-$	$-$	$1, 2, 5$	$+$	$+$
$2, 4$	$-$	$+$	$1, 2, 4$	$-$	$+$
$2, 4, 5$	$+$	$+$	$1, 2, 4, 5$	$-$	$-$
$2, 3$	$-$	$+$	$1, 2, 3$	$+$	$-$
$2, 3, 5$	$+$	$+$	$1, 2, 3, 5$	$-$	$-$
$2, 3, 4$	$+$	$-$	$1, 2, 3, 4$	$-$	$+$
$2, 3, 4, 5$	$-$	$-$	*All*	$+$	$+$
			$(\#+)/(\#-)$	$16/16$	$16/16$

Figure 2: The figure above shows the sign of the coefficient for both C_{-----} and C_{----+} as they appear in the signature of the S matrix for each combination of h_i's. The columns of numbers represent which, if any, h_i values are squared.

From this form, it can be seen that for any n, there will always be 2^n $C_{+\cdots+}$'s in the equation, since $C_{+\cdots+}$ will always be positive for any combination of h_i's. This justifies the coefficient $\frac{1}{2^n}$, and that $C_{+\cdots+}$ will always be in the solution. Now it must be shown that the rest of the $C_{*\cdots*}$'s cancel out.

Since the equation yields only powers of h_i greater than zero, then it is true that each signature combination will contain the same $C_{*\cdots*}$'s; the only difference between them will be the coefficient values (1 or -1). So one way to show that the other $C_{*\cdots*}$'s cancel is by showing that the equation results in the same number of positive and negative coefficients for each $C_{*\cdots*}$. In fact, this does work out for the 2-D, 3-D, and 5-D cases, and will in fact work for any case. Since the coefficient is dependent upon the number of squares and the number of negative signs, this can easily be represented for every $C_{*\cdots*}$. Figure 2 looks at our 5-D case for C_{-----} and C_{----+}. $\square$

This process can be continued this for all $C_{*\cdots*}$s to see that this holds true. This means that the polynomial (13) is sufficient to count the number of critical points. Now that we have this, we can implement a procedure to determine the number of points for a given weight vector $\vec{u}$.

6 Examples of Mathematica Coding

For all of the operations we have discussed above, it was advantageous to create commands in Mathematica that aided the computations in finding the critical points, the number of critical points, and then to solve for the parameters of π,s, and t. This section will feature some example outputs of this coding package. For further detail on the coding, one can see the Appendix following the paper for the complete writeup of the coding procedures. Many of the following codes rely upon the use of another Mathematica package, the Mathematica Gröbner Basis Package (MGBP) [5]. This package is used in operations which are necessary to determine leading terms and to form the monomial basis B of non-leading term monomials.

```
<< "/MGPB.m"
<< "/Implicit.m"
```

6.1 Compute Command

For any weight vector $\vec{u} = (u_0, u_1, u_2, u_3, u_4)$, this command returns the solutions to the original mixing problem.

```
In[1]:= u={11,31,0,31,11}
Out[1]:={11,31,0,31,11}

In[2]:=Compute[u]
Out[2]:={{0.03273809523809523809523809523809523809523809 5238095,
    0.3690476190476190476190476190476190476190 4761905,
    0.1964285714285714285714285714285714285714286,
    0.369047619047619047619047619047619047619 04761905,
    0.0327380952380952380952380952380952380952 38095238095},
    {0.195469970684762919026500577292725374339 5242,
    0.2218106497231094059155694354665987490838,
    0.165438759184255350115859974481351753153 3826,
    0.2218106497231094059155694354665987490838,
    0.1954699706847629190265005772927253743395242}}
```

6.2 Critical Point Test

This command returns all $\vec{u}$ that have critical points that meet the constraints, for any given $n \in \mathbb{N}$ value where, $n = u_0 + u_1 + u_2 + u_3 + u_4$.

```
In[1]:=CPTest[1]
Out[1]:=

{1,0,0,0,0} {{1.,0.,0.,0.,0.}}

{0,1,0,0,0} {{0.,1.,0.,0.,0.},
             {0.314459,0.42265,0.211325,0.0515668,0.},
             {0.,0.5,0.,0.5,0.},
             {0.329505,0.414214,0.21967,0.,0.0366117}}

{0,0,1,0,0} {{0.25,0.,0.75,0.,0.},
             {0.,0.242641,0.514719,0.242641,0.},
             {0.125,0.,0.75,0.,0.125},
             {0.,0.,0.75,0.,0.25}}

{0,0,0,1,0} {{0.,0.5,0.,0.5,0.},
             {0.,0.,0.,1.,0.},
             {0.,0.0515668,0.211325,0.42265,0.314459},
             {0.0366117,0.,0.21967,0.414214,0.329505}}

{0,0,0,0,1} {{0.,0.,0.,0.,1.}}
```

6.3 Finding π, s, t

Given any weight vector $\vec{u}$, this command produces the estimates for the three coin probabilities (here expressed as r, s, and t).

```
In[1]:=U={2,4,5,3,2}
Out[1]:={2,4,5,3,2}

In[2]:=FindRST[U]

Out[2]:={0.0832709,0.22943,0.412121,0.216885,0.0582932}
         {{s -> 0.484656,r -> 0.999902,t -> -2.37445},
         {s -> -2.37445,r -> 0.0000982976,t -> 0.484656}}

         {0.110711,0.28244,0.288948,0.194439,0.123462}
         {{s -> 0.387052,r -> 0.783215,t -> 0.835989},
         {s -> 0.835989,r -> 0.216785,t -> 0.387052}}
```

6.4 Multiplication Matrix for Grevlex Order

Given any Gröbner basis G for the ideal I, this command returns all of the multiplication matrices for all $f \in V$, the basis formed by the non-leading term monomials of G.

```
In[1]:=G={x^2+3x y/2+y^2/2-3x/2-3y/2, x y^2-x, y^3-y}
Out[1]:={x^2+3x y/2+y^2/2-3x/2-3y/2, x y^2-x, y^3-y}

In[2]:=MultMatrixGrevlex[G, {x, y}]
Out[2]:=
```

$$\{ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 1 & \frac{3}{2} & 0 & 1 & -\frac{3}{2} \\ 0 & \frac{3}{2} & 0 & 0 & -\frac{1}{2} \\ 0 & -\frac{1}{2} & 0 & 0 & \frac{3}{2} \\ 0 & -\frac{3}{2} & 1 & 0 & \frac{3}{2} \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{3}{2} & 1 & 0 & \frac{3}{2} \\ 0 & -\frac{1}{2} & 0 & 0 & \frac{3}{2} \\ 0 & \frac{3}{2} & 0 & 0 & -\frac{1}{2} \\ 1 & \frac{3}{2} & 0 & 1 & -\frac{3}{2} \end{pmatrix} \}$$

6.5 S_{hf} Multiplication Matrix

Given the non-leading term basis V, command returns the S_{fh} matrix for any $f \in V$ and for any polynomial $h \in k[x_1, \ldots, x_n]$.

```
In[1]:=V={1,x,y,x y,y^2}
Out[1]:={1,x,y,x y,y^2}
In[2]:=h=x
Out[2]:=x
In[3]:=f=y
Out[3]:=y
In[4]:=hfMultMatrixGrevlex[h,f,V,G,{x,y}]
Out[4]:={{0, 0, 0, 0, 0}, {0, -3/2, 1, 3/2, 0},
         {0, -1/2, 0, 3/2, 0}, {1, 3/2, 0, -3/2, 1},
         {0, 3/2, 0, -1/2, 0}}
```

6.6 S_h Matrix

Using the applications above for the multiplication matrices, and the functions for S, the $S_h Matrix$ command repeats the S function for a given h for all combinations $f, g \in V$.

```
In[1]:=G={x^2 + 3 x y/2 +y^2/2 -3 x/2 - 3y/2, x y^2-x, y^3-y};
In[2]:=VD=VSDimension[G, {x,y}];
In[3]:=V=VD[[1]]
Out[3]:={1, x, y, y^2, xy}
In[4]:=h=x^2+1;
In[5]:=M=shMatrix[V,G,h,{x,y}];
In[6]:=MatrixForm[M]
Out[6]:=
```

$$\begin{pmatrix} 12 & 12 & -3 & 11 & -12 \\ 12 & 26 & 12 & -12 & 18 \\ -3 & -12 & 11 & -3 & 12 \\ 11 & 12 & -3 & 11 & -12 \\ -12 & -18 & 12 & -12 & 26 \end{pmatrix}$$

6.7 Number of Solutions

Given a weight vector, $\vec{u}$, NumSols[$\vec{u}$] returns the number of real solutions in the range $(0, 1)$ to the likelihood function.

```
In[1]:=u = {3, 0, 4, 0, 5};
In[2]:=NumSols[u]
Out[2]:=1
```

7 Summary and Future Goals

Overall the initial goals of the project were successful. We were able to understand both how to count and solve for the critical points of L, by using methods from algebra and algebraic geometry. Furthermore, we were able to take this logic and actually apply it using our programming package from Mathematica. With this package, it is now fairly simple to examine all aspects of our original mixing problem. Beyond this, it gives us a good framework in which we can now apply these techniques to more complex mixing problems, and hence problems more closely related to those witnessed in bioinformatics.

8 Appendix

8.1 Tes[n]

Given a set of real numbers, n, Tes[n] returns a set where the entry i is $true$ if $n_i \geq 0$ and $false$ if $n_i < 0$.

```
Tes[n_]:=Module[
{M,i},
M={};
For[i=1, i<=Length[n],
If[n[[i]]==0 ||Positive[n[[i]]]==True, AppendTo[M,True], AppendTo[M,False]];
i++];
Return[M];
];
```

8.2 Compute[$\vec{u}$]

Given a weight vector $\vec{u} = (u_0, u_1, u_2, u_3, u_4)$, Compute[$\vec{u}$] uses the method of Lagrange Multipliers to compute the critical point(s) $\vec{p} = (p_0, p_1, p_2, p_3, p_4)$ of the likelihood equation, $L = p_0^{u_0} p_1^{u_1} p_2^{u_2} p_3^{u_3} p_4^{u_4}$.

```
Compute[u_] := Module[
{g1, g2, Id,GB, GB0,GB1,GB2,GB3,GB4,A0,A1,A2,A3,A4,A00,A11,A22,A33,A44,SS,
MXT,UU,i,j,k},
Clear[p0,p1,p2,p3,p4];
M ={{12p0, 3 p1, 2 p2}, {3 p1, 2 p2, 3 p3}, {2 p2, 3 p3, 12 p4}};
g2=Det[M];
g1=p0+p1+p2+p3+p4-1;
Id= {u[[1]]- C*p0*D[g1,p0]- D*p0*D[g2, p0], u[[2]]- C*p1*D[g1,p1]-
D*p1*D[g2,p1],u[[3]]-C*p2*D[g1,p2]- D*p2*D[g2, p2], u[[4]]-C*p3*D[g1,p3]-
D*p3*D[g2,p3], u[[5]]-C*p4*D[g1,p4]- D*p4*D[g2, p4], g1, g2};
GB= GroebnerBasis[Id,{p0,p1,p2,p3,p4},{C, D}, MonomialOrder-> EliminationOrder];
 GB0=GroebnerBasis[GB,p0,{p1,p2,p3,p4},MonomialOrder->EliminationOrder];
A0=NSolve[GB0[[1]], 45];
 GB1=GroebnerBasis[GB,p1,{p0,p2,p3,p4},MonomialOrder->EliminationOrder];
A1=NSolve[GB1[[1]],45];
 GB2=GroebnerBasis[GB,p2,{p0,p1,p3,p4},MonomialOrder->EliminationOrder];
A2=NSolve[GB2[[1]],45];
 GB3=GroebnerBasis[GB,p3,{p0,p1,p2,p4},MonomialOrder->EliminationOrder];
A3=NSolve[GB3[[1]],45];
 GB4=GroebnerBasis[GB,p4,{p0,p1,p2,p3},MonomialOrder->EliminationOrder];
A4=NSolve[GB4[[1]],45];
A00={};
A11={};
A22={};
A33={};
A44={};
For[i=1,i<=Length[A0],If[Head/@{p0/.A0[[i]]}=={Real},
                 AppendTo[A00,p0/.A0[[i]]]];i++];
For[i=1,i<=Length[A1],If[Head/@{p1/.A1[[i]]}=={Real},
                 AppendTo[A11,p1/.A1[[i]]]];i++];
For[i=1,i<=Length[A2],If[Head/@{p2/.A2[[i]]}=={Real},
                 AppendTo[A22,p2/.A2[[i]]]];i++];
For[i=1,i<=Length[A3],If[Head/@{p3/.A3[[i]]}=={Real},
                 AppendTo[A33,p3/.A3[[i]]]];i++];
For[i=1,i<=Length[A4],If[Head/@{p4/.A4[[i]]}=={Real},
                 AppendTo[A44,p4/.A4[[i]]]];i++];
SS={};
MXT=Flatten[Table[Table[Table[Table[
                 Table[{A00[[i]],A11[[j]],A22[[k]],A33[[l]],
A44[[m]]},{i,1,Length[A00]}],{j,1,Length[A11]}],{k,1,Length[A22]}],
                 {l,1,Length[A33]}],{m,1,Length[A44]}],4];
For[i=1, i<=Length[MXT],
            If[Abs[MXT[[i,1]]+MXT[[i,2]]+MXT[[i,3]]+MXT[[i,4]]+MXT[[i,5]]
          -1]<=(1*10^-11), AppendTo[SS,MXT[[i]]]];i++];
TT={};
For[k=1,k<=Length[SS], If[Tes[SS[[k]]]==
               {True,True,True,True,True},
AppendTo[TT,SS[[k]]]];k++];
UU={};
For[i=1,i<=Length[TT],{p0,p1,p2,p3,p4}=TT[[i]];
                If[Det[M]<=(1*10^-11), AppendTo[UU,TT[[i]]]];i++];
If[Length[UU]!=0,Return[UU],Return[{}]];
];
```

8.3 CPTest$[n]$

Given $n \in \mathbb{N}$, CPTest$[n]$ returns the critical point(s) $\vec{p} = (p_0, p_1, p_2, p_3, p_4)$ for each $\vec{u} = (u_0, u_1, u_2, u_3, u_4)$ such that $u_0 + u_1 + u_2 + u_3 + u_4 = n$.

```
CPTest[n_]:=Module[
{MM, KK,i,h},
Clear[i,j,k,l,m,KK,MM];
MM = Flatten[Table[Table[Table[Table[Table[{i, j, k, l, m},
             {i, 0, n}],
{j, 0, n}], {k, 0, n}], {l, 0, n}], {m, 0, n}], 4];
KK = {};
For[i = 1, i <= Length[MM],If[MM[[i,1]]+MM[[i,2]]+MM[[i,3]]+
       MM[[i, 4]]+MM[[i, 5]] == n, AppendTo[KK, MM[[i]]]]; i++];
For[h = 1, h <= Length[KK], L = Compute[KK[[h]]];
           If[L != {}, Print[KK[[h]], L]]; h++];
];
```

8.4 FindRST$[\vec{u}]$

Given a weight vector, $\vec{u}$, FindRST$[\vec{u}]$ solves for π, s, t in the original parametric equations for each critical point, $\vec{p}$.

```
FindRST[u_]:=Module[
{SS, Eq,m},
SS=Compute[u];
For[m = 1, m <= Length[SS],
    Clear[r,s,t,p0,p1,p2,p3,p4];
    Eq = {r (1 - s)^4 + (1 - r) (1 - t)^4 - p0,
    4 r s (1 - s)^3 + 4 (1 - r) t (1 - t)^3 - p1,
    6 r s^2 (1 - s)^2 + 6 (1 - r) t^2 (1 - t)^2 - p2,
    4 r s^3 (1 - s) + 4 (1 - r) t^3 (1 - t) - p3,
    r s^4 + (1 - r) t^4 - p4}//.
          {p0->SS[[m,1]],p1->SS[[m,2]],p2->SS[[m,3]],p3->SS[[m, 4]],
           p4->SS[[m,5]]}};
        Print[NSolve[Eq, {r, s, t}]];
m++];
];
```

8.5 Remainders$[V, G, vlist]$

Given the basis, V, the Grob̈ner Basis, G, and the variable(s), *vlist*, the procedure call Remainders$[V, G, vlist]$ returns the remainder upon division of V by G with respect to the variable(s) *vlist*.

```
Remainders[V_, G_, vlist_]:=Module[
{T, l,k,i,j},
l={};
m={};
For[j=1, j<=Length[V],m={};
For[i=1, i<=Length[V],
Clear[T];
T=V[[j]]*V;
k=PRemainder[T[[i]],G,vlist];
AppendTo[m, k];
i++];
AppendTo[l, m];
j++];
Return[l];
];
```

8.6 MultMatrixGrevlex$[G, vlist]$

Let V be the basis formed by all the non-leading term monomials in the Gröbner Basis G, and let f be any element in V. Given the Gröbner Basis, G, and the variable(s), *vlist*, MultMatrixGrevlex$[G, vlist]$ returns multiplication matrices with respect to f, whose terms are the coefficents of the remainders of fV divided by G. This division follows the monomial order Graded Reverse Lexicographic.

```
MultMatrixGrevlex[G_, vlist_]:=Module[
{V, J, M,LT, subs, k, LC, LM,q,j,i,m},
MonOrder[Grevlex];
V=VSDimension[G,vlist];
V=V[[1]];
R=Remainders[V, G, vlist];
J={};
For[q = 1, q <= Length[R], M = Table[Table[0, {i, 1, Length[V]}] 0,
    {j, 1, Length[V]}];
   For[i = 1, i <= Length[R],
LT = {};
subs = {};
     While[R[[q, i]] =!= 0,
                AppendTo[LT, k = GrevlexLT[R[[q, i]], vlist]];
     R[[q, i]] = R[[q, i]] - k;];
For[j = 1, j <= Length[vlist],
                subs = AppendTo[subs, vlist[[j]] -> 1]; j++];
LC = LT //. subs;
LM = LT/LC;
For[j = 1, j <= Length[LT],
   For[m = 1, m <= Length[V],
                  If[LM[[j]] == V[[m]], M[[i, m]] = LC[[j]] ];
       m++];
            j++];
i++];
AppendTo[J, Transpose[M]];
q++];
Return[J];
];
```

8.7 hfMultMatrixGrevlex$[h, f, V, G, vlist]$

Given a Gröbner Basis, G, a basis, V, an element f in the basis V, a polynomial, h, and variables, $vlist$, hfMultMatrixGrevlex$[h, f, V, G, vlist]$ computes the multiplication matrix with respect to fh. It follows the monomial ordering Graded Reverse Lexicographic.

```
hfMultMatrixGrevlex[h_,f_,V_,G_,vlist_]:=Module[
{P,NN,J,LT,LC,LM,M,k, subs,j,m,i},
      subs = {};
      For[j = 1, j <= Length[vlist],
          subs = AppendTo[subs, vlist[[j]] -> 1];
          j++];
P = Expand[h*f*V];
NN={};
For[i = 1, i <= Length[P],
          AppendTo[NN, PRemainder[P[[i]], G, vlist]];
        i++];
J = {};
M = Table[Table[0, {i, 1, Length[V]}] 0, {j, 1, Length[V]}];
For[i = 1, i <= Length[NN],
     LT = {};
     While[NN[[i]] =!= 0,
               AppendTo[LT, k = GrevlexLT[NN[[i]], vlist]];
        NN[[i]] = NN[[i]] - k;];
        LC = LT //. subs;
        LM = LT/LC;
            For[j = 1, j <= Length[LT],
              For[m = 1, m <= Length[V],
                    If[LM[[j]] == V[[m]],
                        M[[i, m]] = LC[[j]]];
                  m++];
                j++];
    i++];
Return[Transpose[M]];
];
```

8.8 **shTr**$[f, g, h, V, G, vlist]$

Given a Gröbner basis, G, a basis V, two elements, f and g, in the basis V, a polynomial, h, and variable(s), *vlist*, the procedure call shTr$[f, g, h, V, G, vlist]$ returns the trace of the multiplication matrix with respect to fh multiplied by the multiplication matrix with respect to g.

```
shTr[f_,g_,h_,V_,G_,vlist_]:=Module[
{M1, l},
M1=hfMultMatrixGrevlex[h,f*g,V,G,vlist];
l=Tr[M1];
Return[l];
];
```

8.9 **shMatrix**$[V, G, h, vlist]$

Given a Gröbner basis, G, a basis, V, a polynomial, h, and variable(s), *vlist*, the procedure call shMatrix$[V, G, h, vlist]$ returns a matrix whose entries are the traces of the multiplication matrices with respect to hfg, where f, g are any two elements in the basis V.

```
shMatrix[V_,G_,h_,vlist_]:=Module[
{T,MM,LL,i,j},
T=Flatten[Table[Table[{V[[j]],V[[i]]},{i,1,Length[V]}],
       {j,1,Length[V]}],1];
MM=Table[Table[0,{i,1,Length[V]}]0,{j,1,Length[V]}];
LL={};
While[T=!={},
Clear[f,g];
{f,g}=T[[1]];
AppendTo[LL,shTr[f,g,h,V,G,vlist]];
T=Drop[T,1] ];
For[i=1,i<=Length[MM],
For[j=1,j<=Length[MM[[1]]],
MM[[i,j]]=LL[[1]]; LL=Drop[LL,1]; j++]; i++];
Return[MM];
];
```

8.10 Rank[m]

Given a matrix, m, Rank[m] returns the rank of the matrix m.

```
Rank[m_]:=Length[Transpose[m]]-Length[NullSpace[m]];
```

8.11 Sig[m]

Given a matrix, m, Sig[m] returns the signature of the matrix m.

```
Sig[m_]:=Module[
{nc,r,i,subs,LT,k,DD,LC},
DD = Expand[Det[m - x IdentityMatrix[Length[m]]]];
subs = {x -> 1};
LT = {};
While[DD =!= 0,
k = GrevlexLT[DD,{x}];
AppendTo[LT, k];
DD = DD - k;];
LC = LT //. subs;
nc=0;
For[i=2,i<=Length[LC],
If[Sign[LC[[i]]] != Sign[LC[[i-1]]], nc=nc+1];
 i++];
       r = Rank[m];
Return[2*nc-r];
];
```

8.12 NumSols[$\vec{u}$]

Given a weight vector, $\vec{u}$, NumSols[$\vec{u}$] returns the number of critical points of the likelihood function $L = p_0^{u_0} p_1^{u_1} p_2^{u_2} p_3^{u_3} p_4^{u_4}$.

```
NumSols[u_]:=Module[
{Combo,H,M,g1,g2,Id,GB,vlist,mV,V,Mh,k,S,MM,T,i,j,l,m,a},
Combo = {};
H={p0,p1,p2,p3,p4};
For[i = 1, i <= 2,
   For[j = 1, j <= 2,
For[k = 1, k <= 2,
For[l = 1, l <= 2,
For[m = 1, m <= 2,
            AppendTo[Combo,
                        H[[1]]^i*H[[2]]^j*H[[3]]^k*
                        H[[4]]^l*H[[5]]^m];
```

```
 m++];
          l++];
 k++];
 j++];
 i++];
M = {{12p0, 3p1, 2p2}, {3p1, 2p2, 3p3}, {2p2, 3p3, 12p4}};
g1 = Det[M];
g2 = p0 + p1 + p2 + p3 + p4 - 1;
Id = {u[[1]] - C p0 D[g1,p0] - D p0 D[g2,p0],
          u[[2]] - C p1 D[g1,p1] - D p1 D[g2,p1],
          u[[3]] - C p2 D[g1,p2] - D p2 D[g2,p2],
          u[[4]] - C p3 D[g1,p3] - D p3 D[g2,p3],
          u[[5]] - C p4 D[g1, p4] - D p4 D[g2, p4], g1, g2};
GB = GroebnerBasis[Id, {p0, p1, p2, p3, p4}, {C, D},
            MonomialOrder -> EliminationOrder];
vlist={p0,p1,p2,p3,p4};
mV = MultMatrixGrevlex[GB, vlist];
V = VSDimension[GB, {p0, p1, p2, p3, p4}];
V = V[[1]];
Mh = {};
For[k = 1, k <= Length[Combo],
     Clear[i, j];
AppendTo[Mh,
           hfMultMatrixGrevlex[Combo[[k]], 1, V, GB, vlist]];
k++];
Clear[k];
S = {};
For[a = 1, a <= Length[Mh],
MM = Table[Table[0, {i, 1, Length[V]}]0, {j, 1, Length[V]}];
For[i = 1, i <= Length[mV],
For[j = i, j <= Length[mV[[1]]],
T = Tr[Dot[mV[[i]],Dot[mV[[j]],Mh[[a]]]]];
                MM[[i, j]] = T;
MM[[j, i]] = T;
 j++];
i++];
     AppendTo[S, Sig[MM]];
a++];

Return[Sum[S[[i]], {i, 1, Length[S]}]/32];
];
```

References

[1] Pachter, Lior; Sturmfels, Bernd: *Algebraic Statistics for Computational Biology*, Cambridge, Cambridge, U.K. 2005.

[2] Hosten, Khetan, & Sturmfels. "Solving the Likelihood Equations," *Foundations of Computational Mathematics* **5** (2005), 389-407.

[3] Cox, Little, & O'Shea. *Ideals, Varieties, and Algorithms*, 3rd ed., Springer, New York, 2006.

[4] Cox, Little, & O'Shea. *Using Algebraic Geometry*, 2nd ed. Springer, New York, 2004.

[5] Cox, David, et al. Mathematica Gröbner Basis Package for *Ideals, Varieties, and Algorithms,* available at URL `http://www.cs.amherst.edu/~dac/iva/groebner40.m`