ABO-gene allele frequency estimation using Phenotypic Blood-Type data.

Introduction:

We have a theoretical frame work where we know that pairing combinations of the 3 alleles (antigens: A, B, O) on the ABO-gene (ABO locus) found on chromosome 9, lead to four phenotypic manifestations in the form of 4 blood types: A, B, AB, O.

Now, there is an interest in estimating the frequency of the 3 alleles A, B, O in the population. One way to do this would be to take blood samples of a big sample of people from the population of interest and using DNA sequencing to determine the allele type (A or B or O) on the ABO-gene on chromosome 9. However, to get reliable frequency estimates one needs to do to DNA sequency for a large number of people, which will be extremely expensive and time consuming.

A more practical and more efficient method for estimating the 3 allele frequencies of a population would be to collect the phenotypic data, the blood type A, B, AB or O of people, which would be much more cheaper and easier to do at a large scale and using the theoretical framework to estimate the 3 allele frequencies in the population using the phenotypic blood type data.

In this paper we will layout the theoretical framework for estimating the allele frequency using phenotypic blood type data and use two separate algorithms and approaches for the estimation. Finally, we will use the actual blood type data gathered from a sample of 2114 people to demonstrate and estimate the allele frequency using the two algorithms.

# Method Section

In the ideal case we could sample n people and find out how many people possess allele A (nA), how many possess allele B (nB) and how many possess allele (nO). In such a case it would be very easy to calculate the allele frequency in the population: freq(A) = nA/n, freq(B) = nB/n, and freq(O) = nO/n. And we would be done. However, getting nA, nB, nO directly from DNA sequencing is very expensive and time consuming. It is much easier to collect the blood type of each individual from the sample of n people. This will give us a numeric count of how many people have each blood type, namely, n\_A: number of people with blood type A, n\_B: number of people with blood type B, n\_AB: number of people with blood type AB and n\_O: number of people with blood type O.

We use genetics to link the number of individuals with alleles nA, nB, and nO with number of people with phenotypes n\_A, n\_B, n\_AB, and n\_O (blood-type).

## Theoretical Framework.

### Genetical Theory

From genetic theory we know that alleles A, B are dominant to allele O; alleles A, B are co-dominant; and allele O is recessive to alleles A, B. Using this below is a mapping from genotype to phenotype.

Genotype Phenotype

AA or AO -🡪 A

BB or BO -🡪 B

AB -🡪 AB

OO -🡪 O

### Hardy–Weinberg equilibrium.

Going from allele frequency to genotypic frequency and going from genotypic frequency to phenotypic frequency.

How reasonable is it to assume HWE.

### Statistical Theory:

We have our log likelihood function:

We can find the maxima’s of this by taking the first derivative, equating to zero and then solving for p and q. However, this is very difficult to do analytically. Hence, we will use two different algorithms and approaches to estimate p and q. Namely, the Estimation-Maximization algorithm and the Newton-Raphson algorithm.

## Estimation-Maximization Algorithm Method

One method to solving this is to think of the allele frequencies as latent variables or missing variables. In this approach we can cast the problem in the framework of EM algorithm.

## Newton-Raphson Algorithm Method

The other method is a more direct method where we directly find the maxima’s of the loglikelihood function using numeric computational method called Newton-Raphson algorithm.

We have the log-likelihood function

$ln(L) ~ n\_\text{a}\ln\left(p^2+2\left(-p q+1\right)p\right)+n\_\text{ab}\ln\left(2qp\right)+2n\_\text{o}\ln\left(-p-q+1\right)+n\_\text{b}\ln\left(2q\left(-p-q+1\right)+q^2\right)$

<!-- https://webdemo.myscript.com/views/math/index.html# hand written to latex and code-->

<!-- https://www.derivative-calculator.net/ all the derivatives -->

Using the properties of log we can be simplified to the following:

$ln(L) ~ n\_\text{b}\ln\left(-q\left(2p+q-2\right)\right)+n\_\text{a}\ln\left(-p\left(p+2q-2\right)\right)+n\_\text{ab}\ln\left(2qp\right)+2n\_\text{o}\ln\left(-p-q+1\right)$

We need to get the full first derivative of this with respect to p and q.

We find that the derivative with respect to p is:

$ \dfrac{2n\_\text{b}}{2p+q-2}+\dfrac{n\_\text{a}\left(2p+2q-2\right)}{p\left(p+2q-2\right)}+\dfrac{n\_\text{ab}}{p}-\dfrac{2n\_\text{o}}{-p-q+1}$

The first derivative with respect to q is:

$$

The second derivatives which are the components of the hessian are as follows:

Ddfpp

$ -\dfrac{4n\_\text{b}}{\left(2p+q-2\right)^2}+\dfrac{2n\_\text{a}}{p\left(p+2q-2\right)}+\dfrac{n\_\text{a}\left(-2p-2q+2\right)}{p^2\left(p+2q-2\right)}+\dfrac{n\_\text{a}\left(-2p-2q+2\right)}{p\left(p+2q-2\right)^2}-\dfrac{n\_\text{ab}}{p^2}-\dfrac{2n\_\text{o}}{\left(-p-q+1\right)^2}$

## Data

# Analysis and Results

# Discussion

1. Trying different starting values for both algorithms. The Newton-Raphson algorithm gets stuck in local minima. With the EM algorithm we also need to be careful with the starting values, as .1 for p0 and q0 are problematic but using 0.6 converges to the correct values.
2. How reasonable is it to assume HWE in our case and what would be the repercussions in case it is not true.

We notice that the Newton-Raphson

Further, the NR algorithm is more computationally intensive compared to the EM algorithm. This is because we are calculating the Hessian Matrix and inverting it.

In terms of speed, the EM algorithm is very fast. It completed 100 iterations in less than a second, whereas the NR algorithm took longer.

In terms of efficiency, the NR algorithm took fewer iterations to converge than the EM algorithm (12 iterations vs. 100 iterations).

## Conclusion

with the assumptions of HWE and random mating and maximum likelihood method estimate.