Simulating Correlated Multivariate Normal Data

Alison Kosel
Advisor: Jo Hardin

April 3, 2009

Contents

1 Motivation

Microarrays are currently used to collect genetic data, but, due to the large number of genes examined at once (often in the thousands) and the noisiness of the data, they are difficult to analyze. One very effective way to look for co-regulation is to apply clustering, which sorts related genes into groups. The massive amount of data generated by the microarray is then turned into smaller groups of genes that biologists can focus on. Clustering requires a good algorithm for partitioning the data, as the groups of genes are only as reliable as the algorithm used to generate the groups. Because no one wants to waste time and money examining a group of genes that doesn't really belong together, developing new clustering algorithms and refining current algorithms are important topics of statistical research. Unfortunately, it is difficult to test new clustering algorithms on real data, as the true relationship between genes is generally unknown and thus the efficacy of the algorithm cannot be determined. Instead, we simulate data whose structure resembles that of real data, thereby creating datasets where we actually know the underlying groups. Simulated datasets can then be used to examine new clustering algorithms that, if they work well, can be used to analyze real microarray data in the future. Thus, our project focuses on the creation of simulated data. While the motivation is genetic data, these techniques could be applied equally well to other large data sets (e.g., economic data). Rather than the generic *samples* and *variables*, we will refer to *subjects* and *genes*.

2 Analysis of Simulated Data - Techniques

2.1 Distance Metrics

Many clustering algorithms partition genes based on the distances between each pair of genes. Hence, the first question when clustering data is what distance metric will be used to assess relationships between pairs of genes. In this project, when considering the definition of distance between genes, we are not interested in similarities of magnitude of expression level but rather in similarities of patterns of expression between genes. In other words, we are more interested in two genes whose expression levels rise and fall in tandem than we are in two genes that just so happen to be expressed at the same level. (This is true for many large data sets – if we are, for example, tracking stocks, we also wish to know which ones rise and fall in tandem rather than those that have similar prices.) Euclidean distance

is therefore completely unsuitable, as it measures magnitude rather than behavior.

Correlation, on the other hand, measures linear relationships – similarity of behavior – which makes it a good candidate for our distance metric. (Unfortunately, there is no metric that can measure all types of relationships.) Correlation is defined as

$$cor(x, y) = \frac{\Sigma(x_i - \bar{x})(y_i - \bar{y})}{(n-1)s_x s_y}$$

Note that correlation is symmetric (i.e., $cor(x, y) = cor(y, x)$). However, correlation is not a true distance metric. Distance metrics must meet the following criteria for all x, y, z in the metric space (in this case, $\mathbb{R}^n$):

- $d(x, y) \geq 0$
- $d(x, y) = d(y, x)$
- $d(x, y) = 0 \iff x = y$
- $d(x, y) \leq d(x, z) + d(y, z)$

Correlation itself, while it measures what we want, is a similarity instead of a distance measure. As the linear relationship between genes becomes stronger, the correlation between them increases; additionally, correlation itself clearly violates the first criterion. We would prefer the distance to decrease as the strength of the relationship increases, as this creates a more intuitive and reasonable system. We will thus define our distance to be $d(x, y) = 1 - cor(x, y)$, though $1 - |cor(x, y)| = d(x, y)$ or $1 - cor(x, y)^2 = d(x, y)$ could also be used with little change in the theory. The choice simply depends on whether one is interested only in genes that are positively correlated or in genes that are positively and negatively correlated.

Consider two pairs of genes: genes a and b, with correlation ρ, and genes c and d, with correlation $-\rho$. We can then calculate the distances between these pairs of genes using each potential distance metric as follows:

$$d(a, b) = 1 - \rho \neq 1 - (-\rho) = d(c, d)$$
$$d(a, b) = 1 - |\rho| = 1 - |-\rho| = d(c, d)$$
$$d(a, b) = 1 - \rho^2 = 1 - (-\rho)^2 = d(c, d)$$

Observe that using $d(x, y) = 1 - cor(x, y)$ we do not obtain $d(a, b) = d(c, d)$, whereas using the other distances we do. Also observe that $1 - cor(x, y) = d(x, y)$

runs from 0 to 2 (the other two potential distances run from 0 to 1), thereby clearly obeying the positivity requirement for distance metrics.

Because correlation is symmetric, we have that $1 - cor(x, y) = 1 - cor(y, x)$, and therefore that $d(x, y) = d(y, x)$. In other words, our distance is also symmetric, meaning that it obeys the second criterion for distance metrics. However, our distance metric fails to meet the other two requirements for distance metrics. While there are restrictions on the possible sets of correlations between three vectors, they are not quite as stringent as the triangle inequality. Fortunately, the triangle inequality is not often violated in practice and does not prevent us from using correlation as our distance metric.

Finally, two vectors can have a distance of zero (equivalent to a correlation of one) even if they are not identical. If one vector is simply a scaled or shifted version of the other (i.e., x and $ax + b$ where $b = \beta * (1 \dots 1)$ are our vectors), then the distance between them is zero, which is, in fact, precisely why we chose to use correlation – we want identically behaving genes to have a distance of zero, regardless of their actual expression levels. To see this, we can first consider x and ax such that $E(x) = E(ax) = 0$, which allows us to obtain

$$cor(x, ax) = \frac{\langle x, ax \rangle}{||x||||ax||} = \frac{\langle x, x \rangle}{||x||||x||} = 1$$

We can then consider $cor(x, ax + b)$ and see that, when $cov(x, ax + b)$ is the covariance of x and $ax + b$ and $var(x)$ is the variance of x,

$$cor(x, ax+b) = \frac{cov(x, ax + b)}{\sqrt{var(x)var(ax + b)}} = \frac{cov(x, ax + b)}{\sqrt{var(x)var(ax)}} = \frac{cov(x, ax) + cov(x, b)}{\sqrt{var(x)var(ax)}}$$

$$= \frac{cov(x, ax)}{\sqrt{var(x)var(ax)}} + \frac{cov(x, b)}{\sqrt{var(x)var(ax)}} = \frac{cov(x, ax)}{\sqrt{var(x)var(ax)}} = cor(x, ax)$$

We have already shown that $cor(x, ax) = 1$, so this implies that $cor(x, ax + b) = 1$. Clearly one could re-center x as well with the same outcome. Correlation also suffers from other limitations. First and foremost, it is only a measure of linear relationships. Non-linear relationships can and will be overlooked. Additionally, correlation is not a robust measure. Even one outlier can easily overshadow a genuine linear relationship or make it appear that a strong linear relationship exists where one does not.

2.2 Clustering

Now that we have set our distance, we can use it to cluster data. PAM (partitioning around medoids) is a well-known clustering algorithm. Thus, PAM will be used to cluster the data after it is simulated. While we do not expect perfect clusters, if the simulated data clusters terribly with PAM, there is probably something wrong with the data, not with the algorithm. Thus, PAM is a method not only of examining the effects of various types of simulation on clustering but also of evaluating the data. PAM partitions the data based on a dissimilarity matrix, which simply tells us how related each pair of objects is; the dissimilarity matrix for simulated data was created by using the correlation distance metric described above. If we have genes $1, \ldots, p$, our dissimiliarity matrix will be

$$\begin{bmatrix} 0 & d(g_1, g_2) & \ldots & d(g_1, g_p) \\ d(g_1, g_2) & 0 & \ldots & d(g_2, g_p) \\ \vdots & \vdots & \ddots & \vdots \\ d(g_1, g_p) & d(g_2, g_p) & \ldots & 0 \end{bmatrix}$$

which in this case is

$$\begin{bmatrix} 0 & 1 - cor(g_1, g_2) & \ldots & 1 - cor(g_1, g_p) \\ 1 - cor(g_1, g_2) & 0 & \ldots & 1 - cor(g_2, g_p) \\ \vdots & \vdots & \ddots & \vdots \\ 1 - cor(g_1, g_p) & 1 - cor(g_2, g_p) & \ldots & 0 \end{bmatrix}$$

PAM can search for the optimal number of clusters but, for this project, the number was simply set to k, the actual number of clusters created in the simulated data. PAM selects k representative genes, called medoids (hence the name, partitioning around medoids), and every other gene in the data set is assigned to a group containing the closest medoid, based on the dissimiliarity matrix. The overall goal of the algorithm is to minimize the sum of dissimilarities between genes and the medoids to which they are assigned. The first medoid is selected by choosing the gene for which the sum of dissimilarities between it and all other genes (i.e., the sum of the distances between each gene and the proposed medoid) is minimized.

The second medoid is selected in much the same manner. We consider an unselected gene as the proposed medoid and calculate a difference score that measures its representative ability. This difference score is calculated by taking into account the distance between all unselected genes and the proposed medoid. If

an unselected gene is closer to its current medoid than to the proposed medoid, it adds nothing (zero) to the difference score because it would not move if the proposed medoid were selected. If an unselected gene is closer to the proposed medoid than to its current medoid, then the difference in the distance between the gene and the proposed medoid and the distance between the gene and the current medoid is added to the difference score (for example, if an unselected gene is a distance of .3 from its current medoid and .1 from the proposed medoid, we add .2 to the difference score of the proposed medoid). We sum up the contributions of each unselected gene to obtain the overall difference score for the proposed medoid. The proposed medoid that has the largest difference score out of all unselected genes is then chosen to be the new medoid. This process is repeated until k medoids have been selected.

PAM then begins the swap stage. PAM examines the sum of dissimiliarities when a representative gene (medoid) is swapped with an unselected gene. If there exists a pair such that swapping them decreases the sum of dissimiliarities, the swap is made. If there exists more than one such pair of genes, the swap that decreases the sum of dissimilarities the most is selected. This swapping continues until the sum of dissimilarities cannot be further minimized [2]. PAM has then clustered the data as well as it is able. However, we wish to know the degree of accuracy from the PAM clustering output. Thus, we look at measures of clustering efficacy.

2.3 Measures of Clustering Efficacy

When clustering data, we would like genes that belong together to be placed in the same cluster and genes that do not belong together to be placed in different clusters. Both of these are good things, but bad things can also happen: genes that don't belong together can wind up in the same cluster and genes that do belong together can wind up in different clusters. One method of measuring clustering efficacy is to count up the number of pairs of genes in each of these four possible scenarios,

$$\begin{bmatrix} & \text{Clustered Together} & \text{Clustered Apart} \\ \text{Actually Together} & a & b \\ \text{Actually Apart} & c & d \end{bmatrix}$$

Note that each of these numbers represents a count of pairs, since we are observing every gene paired with all other genes examined. For p genes, we then have $a + b + c + d = \binom{p}{2}$. One method of measuring the ratio of desirable to

undesirable pairs is the Rand index, which simply takes the number of desirable pairs over the total number of pairs (i.e., Rand index $= \frac{a+d}{a+b+c+d}$) [4]. The Rand index clearly increases as the partition improves, but there is unfortunately no expected value for the Rand index of a random partition. Thus, while we know that a Rand index of 1 represents a perfect partition and a Rand index of 0 represents the worst possible partition, we have no idea what a value of, for example, .3 means.

In order for our index to have a definitive expected value, we can use the hypergeometric distribution to measure the sampling distribution of pairs of genes under the hypothesis that genes are randomly allocated. The hypergeometric distribution is used because we are essentially sampling without replacement – each and every pair of genes is either correctly placed or not. We then have the adjusted Rand index, in which the expected value is subtracted from the numerator and the denominator of the Rand index. If we consider two partitions of p genes, $u_1, ..., u_i$, representing the true partition of the data, and $v_1, ..., v_j$, representing the partition of the data created by the clustering algorithm, with n_{ij} representing the genes in both class u_i and cluster v_j, then the adjusted Rand index is

$$\frac{\Sigma_{i,j}\binom{n_{ij}}{2} - [\Sigma_i \binom{n_{i.}}{2} \Sigma_j \binom{n_{.j}}{2}]/\binom{p}{2}}{\frac{1}{2}[\Sigma_i \binom{n_{i.}}{2} + \Sigma_j \binom{n_{.j}}{2}] - [\Sigma_i \binom{n_{i.}}{2} \Sigma_j \binom{n_{.j}}{2}]/\binom{p}{2}}$$

The adjusted Rand index then gives us 1 as a perfect score and additionally has 0 as the expected adjusted Rand index of a random partition. With these as guides, we can now interpret intermediate scores. The adjusted Rand index is also more sensitive, as it varies over a wider range – unlike the Rand index, it is not bounded below by zero [4].

3 Simulation of Data

The goal of our project is to simulate k clusters, each containing n samples of p genes with correlation ρ. The simulation will be conducted using a p-dimensional multivariate normal distribution, which has the following probability density function:

$$\frac{1}{(2\pi)^{p/2}\ |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$$

where μ represents the p x 1 mean vector, Σ represents the p x p covariance matrix, and x represents the p x 1 random variable. The p x 1 mean vector, μ, was set to

the zero vector and variances were set to 1, which can be done without loss of generality. Observe that if x is a vector, a is a scalar, and β is a scalar such that $b = \beta * (1 \dots 1)$, the correlation between x and $ax + b$ is 1. In other words, $d(x, y) = d(ax + b, y)$ for all vectors y, as noted previously. Thus, we may scale and shift the vectors at any point without affecting the underlying correlation structure of the data. The correlation structure is the subject of interest, so there is no sense in clouding the picture by introducing shifts and scaling at this point.

Σ must be positive definite. One can observe that the probability contours of the distribution only exist if all eigenvalues are positive; as the determinant is the product of the eigenvalues, clearly this implies that the determinant is positive. More rigorously, one can see that because $cor(x, y) = cor(y, x)$, Σ is symmetric. As a symmetric matrix, Σ can be rewritten as AA^T for some matrix A. Thus, when v is any p x 1 vector,

$$\langle \Sigma v, v \rangle = \left\langle AA^T v, v \right\rangle = \langle Av, Av \rangle = ||Av||^2 \geq 0$$

thereby implying that $|\Sigma| \geq 0$.

3.1 Basic Simulation

The initial simulation was performed in order to provide a general understanding of the problem. We let all genes in a cluster be correlated at the same level, so Σ_a, the p x p covariance matrix for cluster a, will be

$$\begin{bmatrix} 1 & \rho_a & \cdots & \rho_a \\ \rho_a & 1 & \cdots & \rho_a \\ \vdots & \vdots & \ddots & \rho_a \\ \rho_a & \rho_a & \cdots & 1 \end{bmatrix}$$

where ρ_a is the correlation between genes in cluster a. Because the variance of all genes has been set to 1, this covariance matrix is equivalent to the correlation matrix; one can observe this from the fact that correlation is defined as

$$\rho_{ij} = \frac{\sigma_{ij}}{\sigma_{ii}\sigma_{jj}}$$

and we have set $\sigma_{ii} = \sigma_{jj} = 1$. To simulate multiple clusters at once, the correlation matrix for each cluster was included as a diagonal partition of a larger covariance matrix. If we wish to simulate k clusters, labeled $1, 2, \dots, k$, then the overall correlation matrix Σ will be

$$\begin{bmatrix} \Sigma_1 & 0 & 0 & 0 \\ 0 & \Sigma_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \Sigma_k \end{bmatrix}$$

where 0 is not a scalar but rather a matrix of zeroes (inter-cluster correlations will not be introduced at this point). Thus, while Σ_a is a correlation matrix in and of itself, it can also be thought of as an element of a larger correlation matrix.

From there, we generate by row a matrix of data. For one cluster, we will have the following data matrix:

$$\begin{bmatrix} g_{1,1} & g_{2,1} & \cdots & g_{p,1} \\ g_{1,2} & g_{2,2} & \cdots & g_{p,2} \\ \vdots & \vdots & \ddots & \vdots \\ g_{1,n} & g_{2,n} & \cdots & g_{p,n} \end{bmatrix}$$

The above matrix represents n independent subjects drawn from the p-dimensional multivariate normal distribution and becomes an n x p matrix where each subject forms a row and each gene forms a column. While the entries of the matrix are generated in rows, they are analyzed in columns, as it is the correlation between genes rather than the correlation between subjects that is of interest. Indeed, the correlation between subjects is assumed to be zero since they are independent. The expected value of the correlation between any two genes in cluster a is ρ_a, regardless of the value of n. And the means specified in the distribution for each dimension (gene) represent columns, so that any shift in the mean vector would affect each element in a column equally, as we expected previously. In other words, all elements of any given column are generated from the same univariate normal distribution and so a non-zero mean would simply increase or decrease the expected value of each element by the same amount.

If a larger value of n is desired, adding more samples is trivial, as any number of 1 x p vectors can be easily generated from the distribution. Another sample would simply cause the data matrix to become:

$$\begin{bmatrix} g_{1,1} & g_{2,1} & \cdots & g_{p,1} \\ g_{1,2} & g_{2,2} & \cdots & g_{p,2} \\ \vdots & \vdots & \ddots & \vdots \\ g_{1,n} & g_{2,n} & \cdots & g_{p,n} \\ g_{1,n+1} & g_{2,n+1} & \cdots & g_{p,n+1} \end{bmatrix}$$

Adding another correlated gene is possible but more difficult. We must add a value for gene $(p+1)$ for each of the n subjects. For each of the n samples, we are considering a $(p+1)$-dimensional multivariate normal distribution where the first p elements are fixed, leaving us with one dimension (which represents gene $(p+1)$) unspecified. In other words, we are generating a value for the $(p+1)$th gene from a conditional univariate normal distribution. To generate values for gene $(p+1)$, we must use the conditional distribution function $f(g_{p+1}|g_1, g_2, \ldots, g_p)$ because each gene is correlated with all of the other genes, so the values of the first p genes will alter the probability function of gene $(p+1)$.

Because there are n samples, we must generate n values for the $(p+1)$th gene. Each of these values will be conditional on the values of the other genes from the same subject (and only on the values of the other genes from the same subject). In essence, we maintain both the intra-subject dependence of genes (based on the correlations we've specified) and the inter-subject independence, meaning that the conditional distributions from which we generate our $(p+1)$th gene readings are independent.

Recall that the mean of all genes is set to zero. Then

$$g_{p+1,i}|g_1, g_2, \ldots, g_p \sim N(\Sigma_{p+1,a}\Sigma_{a,a}^{-1}X_i, \Sigma_{p+1,p+1} - \Sigma_{p+1,a}\Sigma_{a,a}^{-1}\Sigma_{a,p+1})$$

where $X_i = [g_{1,i}, g_{2,i}, \ldots, g_{p,i}]^T$ [1]. Also recall that each of the values of g_{p+1} is independent; hence, the correlation between any two subjects – and therefore between the values of the $(p+1)$th gene for each subject – is zero. Therefore, the distribution of g_{p+1} is an n-dimensional multivariate normal distribution with mean

$$\begin{bmatrix} \Sigma_{p+1,a}\Sigma_{a,a}^{-1}X_1 \\ \Sigma_{p+1,a}\Sigma_{a,a}^{-1}X_2 \\ \vdots \\ \Sigma_{p+1,a}\Sigma_{a,a}^{-1}X_n \end{bmatrix}$$

and n x n covariance matrix

$$\begin{bmatrix} \Sigma_{p+1,p+1} - \Sigma_{p+1,a}\Sigma_{a,a}^{-1}\Sigma_{a,p+1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \Sigma_{p+1,p+1} - \Sigma_{p+1,a}\Sigma_{a,a}^{-1}\Sigma_{a,p+1} \end{bmatrix}$$

The new $(p+1)$ x $(p+1)$ covariance matrix is written as

$$\begin{bmatrix} \Sigma_{a,a} & \Sigma_{a,p+1} \\ \Sigma_{p+1,a} & \Sigma_{p+1,p+1} \end{bmatrix}$$

where $\Sigma_{a,a}$ is the p x p covariance matrix for the p genes in cluster a, $\Sigma_{p+1,p+1}$ is the 1 x 1 covariance matrix of g_{p+1} (i.e., [1]), $\Sigma_{p+1,a}$ is the 1 x p covariance vector between g_{p+1} and the other p genes (i.e., [ρ_a, ρ_a, $\ldots$, ρ_a], as all genes have the same correlation), and $\Sigma_{a,p+1} = (\Sigma_{p+1,a})^T$.

Observe that there are several parameters, ρ, k, n, and p, in the data whose values can be adjusted. By altering these parameters, it is possible to create a wide variety of simulated data sets in the manner described above. By examining several values of each parameter, the effect of changing each and the interactions between such changes can be seen. Therefore, this is the next step in order to determine the ideal parameters for any given situation and better understand their effects. Once the effect of these variables on the data is determined, more complicated simulations can be examined.

Data were generated for several values of each parameter: $\rho = .2, .5, .8$; $k = 2, 3, 5$; $n = 5, 10, 20, 50$; and $p = 20, 50, 100$. Each box seen in Figures 1, 2, and 3 represents 1000 samples drawn from a distribution with the specified parameters. The data matrix was generated, PAM was used to cluster the data, and the adjusted Rand value of the resulting partition was then calculated. The adjusted rand value of each sample is plotted. While large correlation and large sample size have a positive effect on clustering, a large number of clusters has a negative effect. Number of genes has little to no effect.

When correlation is large, other parameters are very nearly irrelevant; even with small sample sizes and a large number of clusters, good partitions are obtained. However, when the relationship is weak, very large sample sizes are needed to obtain a good partition. For example, even when we have two clusters, 20 genes, and 50 samples, the mean adjusted Rand index of the partition we obtain is only .705 when correlation is .2. When dealing with real data, correlation unfortunately is the parameter we have least control over. While we can always take more samples or examine a different number of genes, we cannot change the innate relationship between genes. Weaker relationships are simply more difficult to see and it is difficult to imagine any algorithm that can overcome this fact.

3.2 Further Simulation

In the simulations conducted in earlier sections, no problems with the correlation matrices were encountered. However, the correlation matrix itself began to cause significant problems in more complex simulations. Specifically, large correlations and large dimensions cause the determinant of the correlation matrix to go to zero, thereby compromising its invertibility and stability. Eventually, as the dimension of the matrix goes to infinity, the determinant becomes close enough to zero that software cannot distinguish it from zero; the determinant goes to zero regardless of the correlation, but it occurs more quickly for larger correlations. As the off-diagonal values begin to approach the diagonal ones, the matrix comes dangerously close to losing its linear independence. In Figure 4, one can see that the determinant decreases exponentially as matrix size increases. Even with relatively small matrices – with no more than 100 genes – the determinants are already incredibly small. This can be seen numerically if we consider a p x p correlation matrix Σ such that $\rho_{ij} = \rho$ for all i, j. We can then write

$$\Sigma = (1 - \rho)I + \rho M$$

where M is a p x p matrix of ones and I is the p x p identity matrix. Because M has rank one, $p - 1$ of its eigenvalues are zero; the last is p. We then know that the eigenvalues of Σ are $1 + (p - 1)\rho$ and $p - 1$ copies of $1 - \rho$. Because the determinant is the product of the eigenvalues, $|\Sigma| = (1 + (p - 1)\rho)(1 - \rho)^{p-1}$, which explains the exponential decrease in determinant size as p increases and the fact that a larger value of ρ causes this to occur more quickly.

Adding additional randomness also causes problems, as the matrix that results from even a slight perturbation often has a negative determinant. One way around this is to consider a sample correlation matrix as the actual correlation matrix for our distribution. In other words, we decide where we would like our correlations centered, create Σ, draw a sample from the distribution, and use the correlation matrix obtained from this sample in the distribution function. Because this data exists, it clearly will come from a correlation matrix that is positive definite. Thus, we obtain additional randomness without the problem of negative determinants.

3.3 Conclusion

We have explored methods of analyzng and simulating correlated data, which can be useful in a variety of situations, particularly when working with microarray data. Using the same methodology, we can simulate a wide variety of data with

which we can test clustering algorithms, though problems with the simulation can arise when dealing with very large amounts of data. These problems are interesting subjects for future study; examining additional methods of introducing randomness into the data as well as determining more precisely the nature of the restrictions on the correlation structure would be particularly useful.

4 References and Acknowledgments

References

[1] Richard A. Johnson and Dean W. Wichern. *Applied Multivariate Statistical Analysis*. Pearson Education, Inc., 2007.

[2] Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley and Sons, 1990.

[3] Brian D. Ripley. *Stochastic Simulation*. John Wiley and Sons, 1987.

[4] Ka Yee Yeung and Walter L. Ruzzo. Details of the adjusted rand index and clustering algorithms: Supplement to the paper 'an empirical study on principal component analysis for clustering gene expression data'. *Bioinformatics*, 17(9), 2001.

Acknowledgments

Thanks to Jo Hardin, Stephan Garcia, and Ghassan Sarkis!

5 Figures and Tables

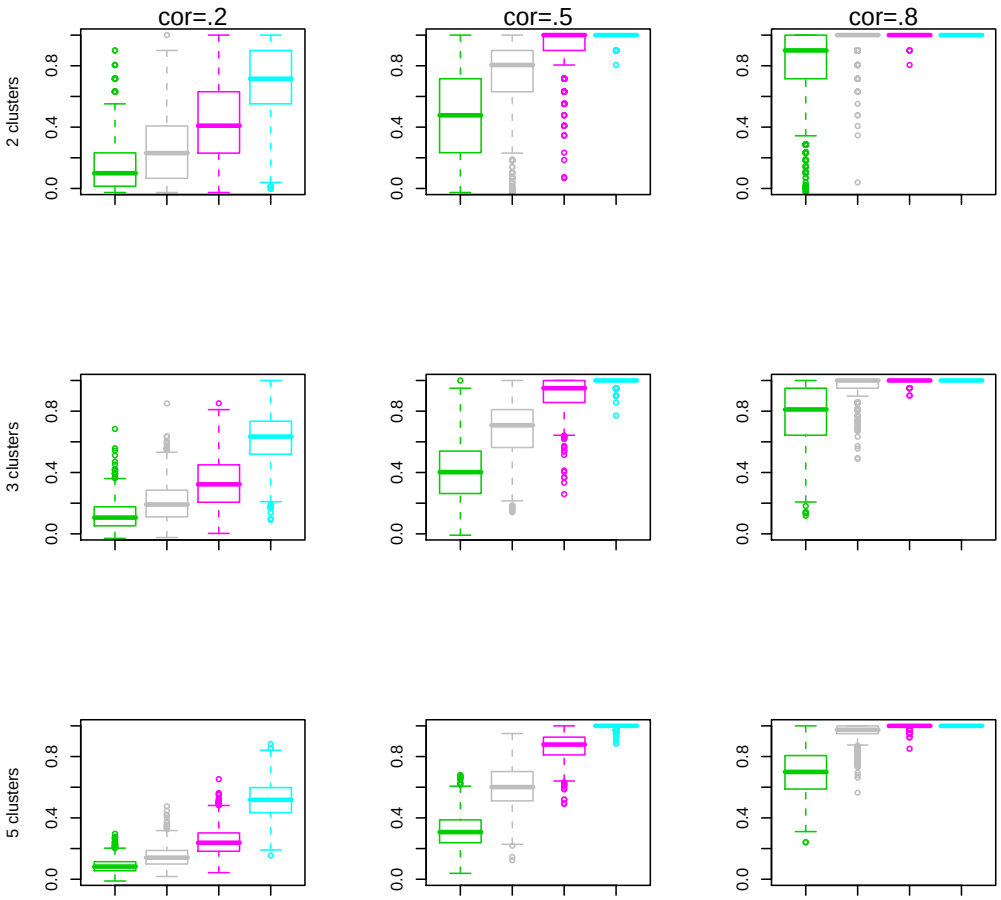

Figure 1: The above plots show the distribution of adjusted rand values when there are 20 genes. The number of subjects are varied across each plot: 5, 10, 20, and 50, from left to right within a plot. Correlation is varied by column and number of clusters by row.

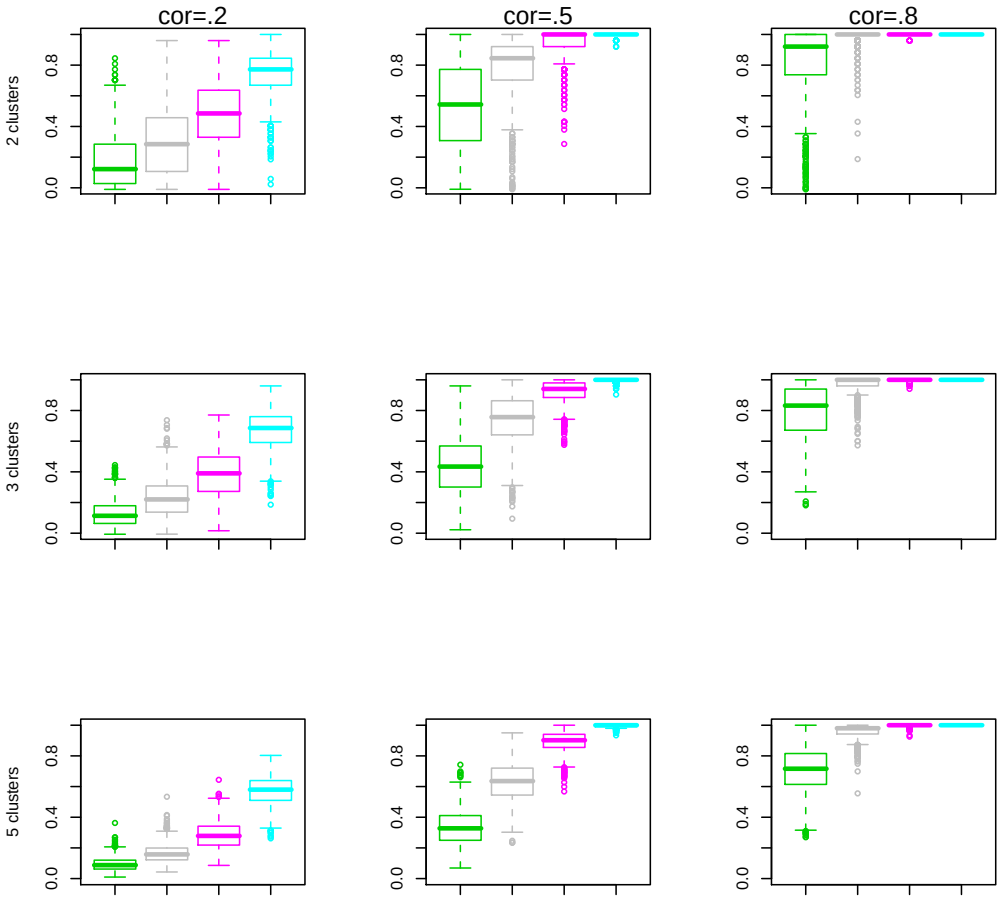

Figure 2: The above plots show the distribution of adjusted rand values when there are 50 genes. The number of subjects are varied across each plot: 5, 10, 20, and 50, from left to right within a plot. Correlation is varied by column and number of clusters by row.

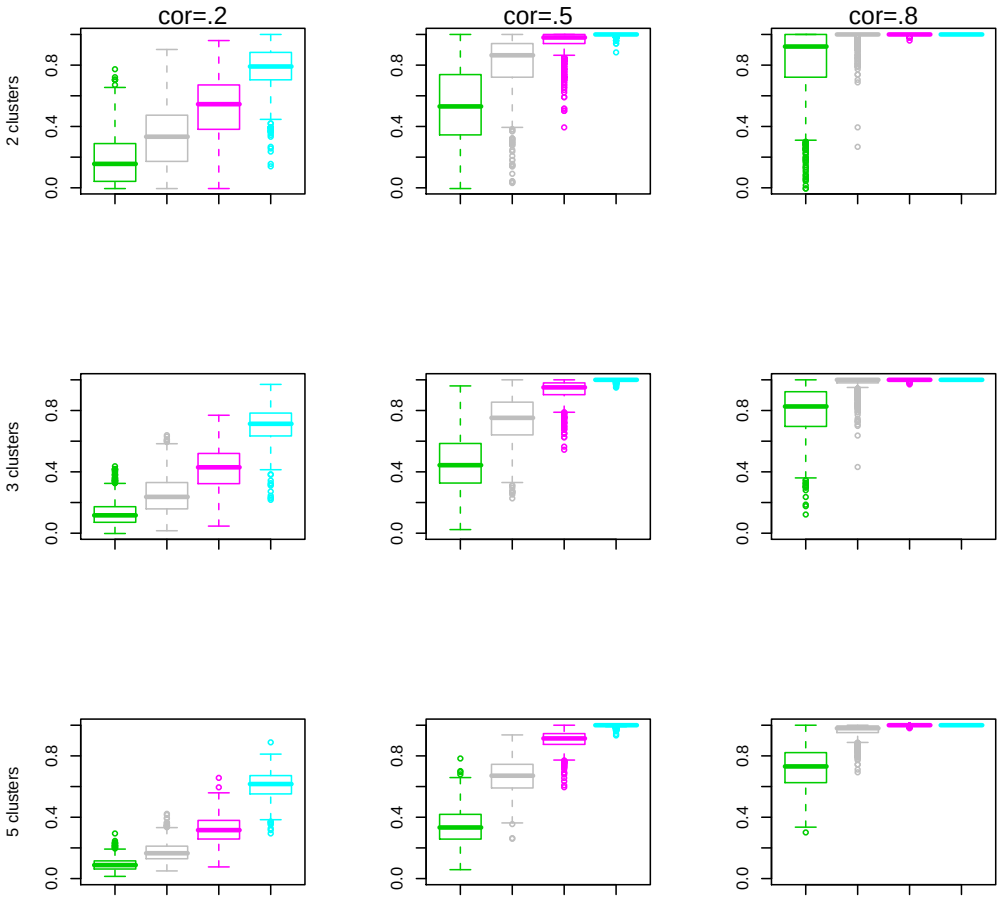

Figure 3: The above plots show the distribution of adjusted rand values when there are 100 genes. The number of subjects are varied across each plot: 5, 10, 20, and 50, from left to right within a plot. Correlation is varied by column and number of clusters by row.

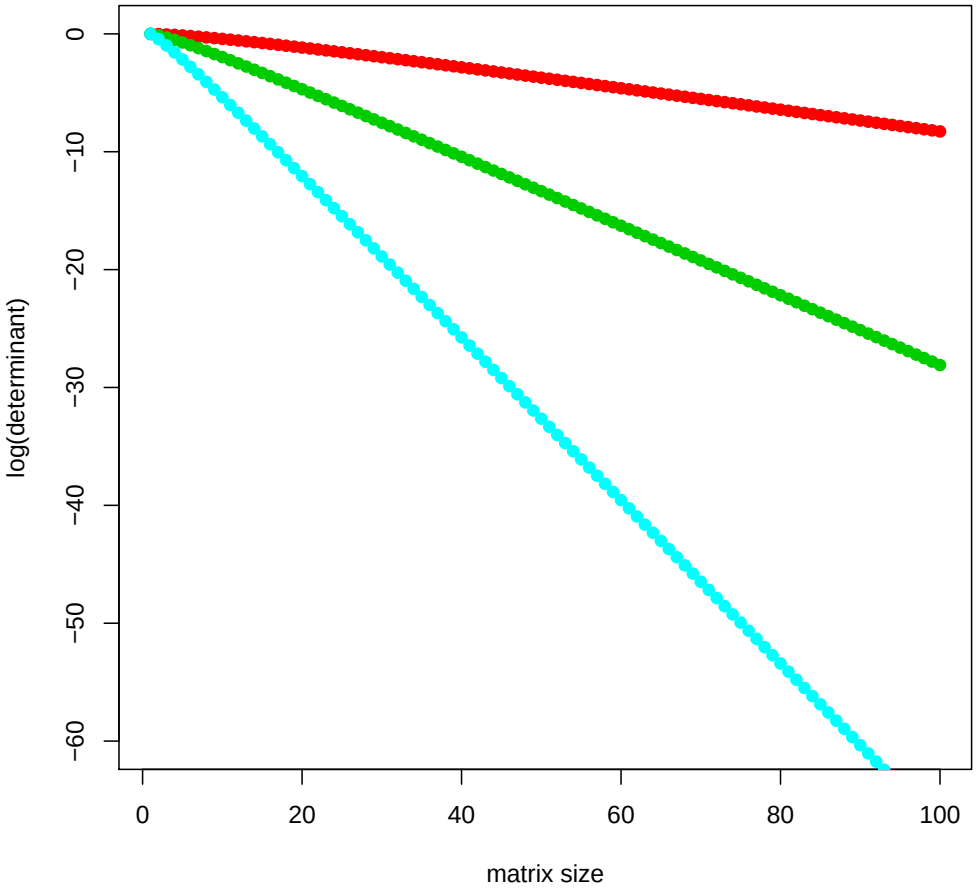

Figure 4: The above plot shows the log of the determinant plotted against matrix size for correlations of .2 (red), .5 (green), and .8 (cyan). Matrix size ranges from 1 to 100. Note that the larger the correlation, the more quickly the determinant goes to zero.