Differential Gene Expression and Developmental Stages in Date Palm Fruit Varieties

Final Report: Transcriptomics
NYU-Tandon School of Engineering
By Michael Dhar

ABSTRACT

The date palm (*Phoenix dactylifera*) fruit is one of the most economically important food crops in the Middle East and North Africa, but faces ongoing breeding and cultivation challenges. The more than 2,000 varieties of date palm are valued primarily for differences in their fruit traits. However, there has been little study until recently concerning the genetic basis for differing varietal traits. The current report attempts to add to this knowledge-base by investigating RNA-Seq data from five varieties of date palm fruits, each sampled at five different time points in the fruit's development (45, 75, 105, 120, and 135 days post-pollination). The report describes differential expression, clustering and GO-term enrichment analyses of this RNA-seq data. These analyses attempt to elucidate the expression patterns characteristic of the five varieties, particularly their developmental stages. Results showed that the time-point replicates largely clustered together within varieties, as was expected Not all varieties followed this pattern, however. Moreover, time-point and gene-expression clustering together suggest broader developmental stages that span time periods. It was also found that developmental stages may have sub-stages. These stages show different patterns among the 5 varieties. GO-term analysis further showed that the most significantly enriched GO terms differed among the varieties, while their broader GO-term profiles were similar. The terms tend to be general, or related to the cell cycle and/or fruit development. Finally, a hypothesis was made linking a trait of the ChiChi variety to the GO terms of genes differentially expressed in ChiChi but not in Khalas.

INTRODUCTION

Date palm (*Phoenix dactylifera*) is a dioecious (male/female), perennial diploid (2n = 36) tree in the *Arecaceae* family, or palm trees (Hazzouri et al., 2015). It produces the date fruit, described as "One of the most economically important fruit trees in the Middle East and North Africa" (Mokhtar et al., 2016) and a major food and income source in that region (Chao and Krueger, 2007). Production of the popular fruit is expected to continue to grow, in response to the increasing (human) population in the Middle East (Chao and Kreuger, 2007).

The date palm fruit comes in more than 2,000 different cultivars, which vary in shape, color, size, sugar content, flowering time, and other traits (Mokhtar et al., 2016)(Hazzouri et al., 2015). These different varieties or genotypes are valued primarily due to these fruit traits (Hazzouri et al., 2015). However, until recently, the detection of genetic differences among these many varieties depended on morphological markers alone, so "the pace of progress was bound to be slow" for breeding programs (Mokhtar et al., 2016).

The last decade has seen more genetic-marker and genome-sequencing studies of date palm, including analyses of simple sequence repeats (SSRs) and sequencing of the mitochondrial and nuclear genomes (Mokhtar et al., 2016). This and further genetic studies of date palm are necessary to "improve date palm breeding for yield and other agronomic traits" and also to

answer questions about the plant's diversity (Hazzouri et al., 2015). Date palm continues to present breeding challenges due to its long life-cycle and juvenile period, as well as its dioecism (Chao and Krueger, 2007). However, even after recent genetic studies, "very little is known about the genomic diversity" of date palm, and further studies of this species' genetic variations are needed to ensure its continued production in the face of cultivation challenges (Hazzouri et al., 2015).

RNA-Seq Data on Fruit Development

One of the authors in the recent Hazzouri study on single nucleotide polymorphisms in date palm (2015) was Jonathan Flowers, Ph.D., at NYU Biology. His lab recently completed an RNA-Seq experiment on date palm fruit that sequenced five different varieties (or cultivars/genotypes) of date palm fruit, each at five different time points in the fruit's development.

The goal of that experiment, and of the ongoing analysis of the sequences obtained, was to understand changes in the metabolism of the fruit varieties and determine if the varieties have similar gene-expression patterns (Flowers J, personal communication, 2016). The sequences obtained in that experiment (unpublished data) were previously aligned to the reference genome under a bioinformatics pipeline written and run by the author of the current report. That pipeline produced a count table of the gene expression for the samples from the five varieties (see details in Materials below). No differential expression analysis had been done on this count table, however, until the work described in the current report.

Objective and Importance of Current Analysis

The current analysis was intended to examine the differential expression of the five varieties in order to elucidate their differential genetic expression and developmental patterns. To do so, differential expression, clustering and GO-term-enrichment analyses were performed. It was hoped that several questions could be answered, while also providing information to shape further analysis. Investigators wanted to know:

- Do time points within varieties cluster together?
- What are the developmental stages? Do these stages span time points?
- How does the full library (all varieties and times) cluster, by time or variety?
- How do genes cluster—i.e., what are the varieties' expression patterns?
- How similar are the GO-term profiles of the varieties?
- What types of GO terms characterize the differentially expressed genes of the varieties?

An attempt to answer these questions through the differential expression analysis of the developing-fruit sequence data is the focus of the current report. This analysis aims to further elucidate the genetic basis for differences in date palm fruit varieties, and ultimately, potentially contribute to improvements in breeding programs and production of this important food crop, amid ongoing challenges in cultivation (Hazzouri et al., 2016)(Flowers J, personal communication, 2016).

MATERIALS

Hardware

The previous bioinformatics analysis pipeline, for alignment of the FastQ files, was completed using the following systems: Personal laptop: 2012 MacBook Pro; NYU Mercer cluster HPC, accessed via SSH; NYU-Abu Dhabi Butinah cluster HPC, accessed via SSH. The bioinformatics analysis for this Transcriptomics course final project was done using the following hardware: Personal computer: 2015 MacBook Pro; NYU Mercer cluster HPC, accessed via SSH.

Software

The previous alignment pipeline used the following software: R, R packages (Bioconductor, easyRNASeq), Perl, STAR, and BASH scripting. The current Transcriptomics course final project used the following software and packages: R, R packages (Bioconductor, DESeq2, Pheatmap, RColorBrewer, xlsx, and GOStats/GSEABase), BASH scripting, and InterProScan.

Beginning Data

The previous, alignment pipeline aligned 64 FastQ files derived from Illumina paired-end (2 x 100bp) RNA-seq experiments done on date palm (*Phoenix dactylifera*) samples. The samples came from the fruit of five different varieties (cultivars, or genotypes) of date palm, all collected from trees in the United Arab Emirates. The varieties, all common commercial varieties in the Arabian Peninsula, are: ChiChi, Kenezi, Khalas, Lulu, and Nebeit Seif. The date palm GFF and protein FASTA files were obtained from the NCBI FTP site (ftp://ftp.ncbi.nlm.nih.gov/genomes/Phoenix_dactylifera/).

The varieties were all sampled at different times, measured in days post-pollination: 45 days, 75 days, 105 days, 120 days, and 135 days. From two to four biological replicates were sequenced for each time point for each variety, as follows:

Table 1. Replicate counts for varieties/time points

Variety	T. 45	T. 75	T. 105	T. 120	T. 135
ChiChi	2	3	3	3	2
Kenezi	3	3	3	3	3
Khalas	3	3	3	4	3
Lulu	2	2	2	2	2
Nebeit Seif	2	2	2	2	2

Alignments/Count Table

The sequencing experiment produced 64 FastQ files. These were then indexed and aligned on the NYU-Abu Dahbi Butinah HPC system using a BASH/Perl-coded pipeline written by M. Dhar in 2015/2016 and executed in spring 2016, producing 64 associated BAM alignment files. The

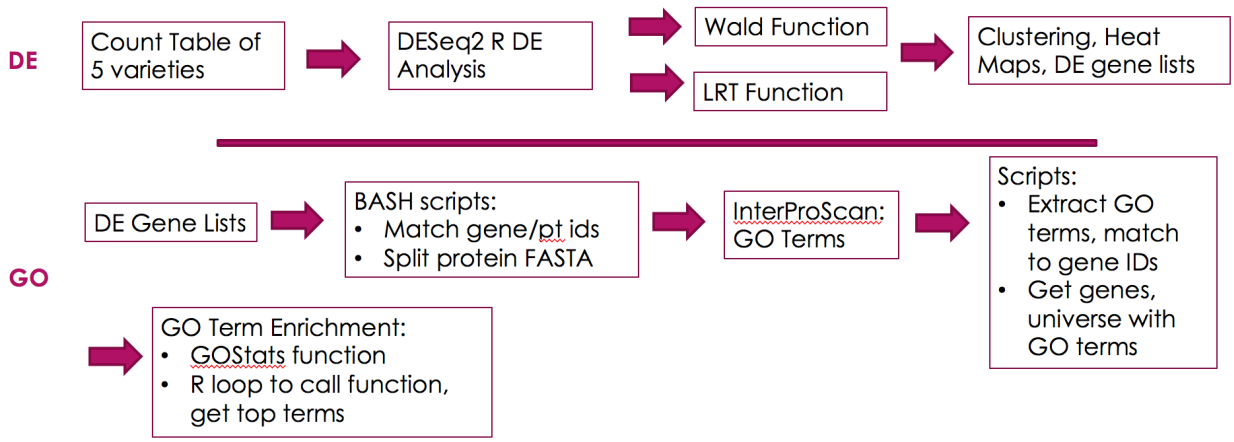
same pipeline then extracted a count table from these BAM files using the simpleRNASeq function of the easyRNASeq R package. No differential expression analysis was done prior to this course on that count table.

METHODS/RESULTS

Project Workflow Diagram

The overall workflow of this final project (including differential expression/clustering and GO-term-enrichment steps) is summarized below:

Diagram 1. Project Workflow



Methods: Differential Expression - DESeq2

The count table, produced on 12 May 2016, was the starting point for the differential expression and Gene Ontology enrichment analyses completed for this project. Differential expression was found for each variety, with time points as the factor. This was the primary differential-expression analysis of interest to the principal investigator, as it was predicted that the differing developmental patterns and time-courses among the five varieties would make an overall comparison of all libraries less biological meaningful. However, it was also of interest whether the entire library (all varieties, all times) would cluster primarily by time or variety. Therefore, differential expression was also determined for the overall library as a multifactor experiment, in terms of both time and variety.

Differential expression was calculated using the R package DESeq2. This program was chosen because hypothesis tests for differential expression that require continuously varying data, as is found in microarrays for example, would not be appropriate for discrete, count data. Thus, t-test and ANOVA would be inappropriate for this project. The DESeq2 package, by contrast, is designed to deal with the discrete count data resulting from an RNA-Seq experiment. DESeq2 checks for differential expression using negative binomial generalized models (Love et al.,

2016), meaning linear models that can handle non-normal error distributions and distributions of discrete yes/no values (as is characteristic of gene counts).

DESeq2 actually offers several options for setting up the hypothesis test used in the program's differential expression analysis. The basic design uses the Wald test, and sets one level of the factor as the reference. The Wald test is a parametric test (Wald test, Wikipedia) that evaluates if the estimated standard error of log2 fold change is zero (Love et al., 2016). This test was done on all varieties and on the overall library.

DESeq2 also allows for time-series hypothesis testing. Instead of comparing all levels of a factor against a reference level, this design tests each level of the factor (as a time series) against one another. This is done using the likelihood-ratio test (LRT), which is similar to ANOVA and can be used to "test for any differences over multiple time points" (Love et al., 2016). It does so by evaluating a "full model," then a "reduced model," with the time levels removed, and then checking for differences (Love et al., 2016). This test design was also conducted on all varieties with time as the factor, and on the comprehensive library, with both time and variety as factors. It was chosen to perform both Wald and LRT tests because a) the PI suggested using Wald with a reference level of 45 to provide a basic analysis, while b) the LRT model seemed the most relevant of DESeq2's hypothesis designs for this project's time-series data.

Methods: Normalization

DESeq2's basic differential-expression function expects non-normalized count tables as input. This is because the package does internal normalization for library size (Love et al., 2016). DESeq2's results function, applied to the differential-expression object, "automatically performs independent filtering based on the mean of normalized counts for each gene, optimizing the number of genes which will have an adjusted p value below a given FDR cutoff, alpha" (Love et al., 2016). The adjusted p-values (padj) in the results table can thus be used to filter by a given FDR cutoff value (Love et al., 2016). An alpha of 0.05 was used in this analysis.

The package recommends that other transformations be conducted on the data for use in hierarchical clustering and other visualization functions. These transformations include regularized-log (RLD) and variance-stabilized (VSD) transformations. RLD transforms data to log2 scale, and is recommended for clustering (Flowers, personal communication, 2016)(Love et al., 2016). It was used for hierarchical clustering and heat maps in the current project. VSD relies on a nonparametric fit for dispersion, using a "closed-form expression" instead of statistical parameters (Love et al., 2016). It was used for heat maps.

Methods: DESeq2 Function - Differential Expression/Clustering

R functions were written to perform differential expression analysis steps and produce a DESeqDataSet (dds) object for each variety in terms of the time factor. These steps — estimating size factors, estimating dispersions, and performing the hypothesis test — are wrapped into a single DESeq2 command (Love et al., 2016). One function was written to do the above using R's basic (Wald) test (Appendix 1), and one to do the time-series (LRT) test

(Appendix 2). Each function also accepted a Boolean argument to switch it to a multi-factor test that included factors for both time and the different varieties. This switch was used to perform a multi-factor test on the entire library (all varieties and times) under both the Wald and LRT hypothesis tests.

Results: Differential Expression

Lists of differentially expressed genes were produced for each sample and for the multifactor all-varieties, all-times test. These lists were printed out by the DESeq2 functions described above, and the lists saved for later use in GO analysis. Numbers of differentially expressed genes (padj > 0.05) were obtained for each variety and for the all-variety experiment, under both the Wald and LRT hypothesis designs (Table 2).

Table 2. Differentially expressed gene counts for all varieties

Variety	**# Differentially Expressed genes (p < 0.05) Wald Test**	**# Differentially expressed genes (p < 0.05) Time Series (LRT)**
ChiChi	14,930	18,761
Kenezi	16,352	18,861
Khalas	10,766	14,227
Lulu	17,464	21,024
Nebeit Seif	17,417	21,226
All Varieties	18,523	22,142

Methods: Clustering

The DESeq2 functions also extracted the results from the dds objects created, and then produced the following output, conducting hierarchical clustering and producing various visualizations:

- A hierarchical clustering graph. It was found that there was no difference in this clustering between the Wald and LRT outputs. Therefore, when silhouette plots were made of each variety in terms of time, this was done only with the Wald test results, which were quicker. Clustering was also done for the full library. (Clustering was done using the RLD transformation, as this transformation was recommended by both DESeq2 and the principal investigator as most appropriate for hierarchical clustering.)
- Hierarchical clustering, silhouettes and boxplots for differentially expressed genes. This was done in LRT, because this method produced a greater number of differentially expressed genes. It was desired to capture all of them in the gene clustering.
- A heat map of the expression for the variety's genes using RLD transformation (also done for full library) using the Pheatmap R package
- A heat map of the distances between the samples for the variety (also done for full library)

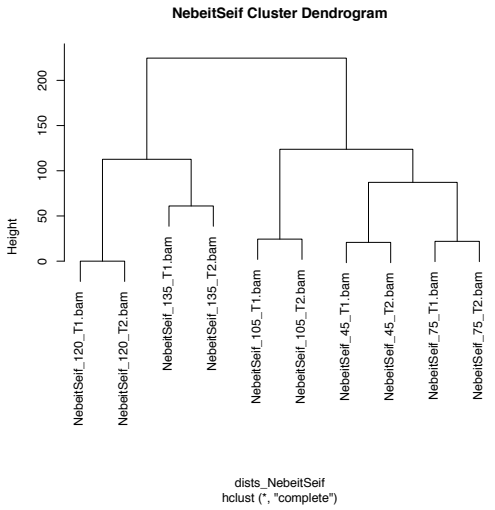
Results: Clustering - Finding Developmental Stages

Results: Clustering of times within varieties

One of the primary questions investigators wanted to address in this project was how the time points within varieties clustered. Hierarchical clustering of the differentially expressed genes in terms of sample (here, meaning the time points for each variety) provides some answers to that question. (Sample hierarchical clustering was done using both the Wald test and LRT designs, but no differences were found in the results from the two methods.)

The sample clustering of Nebeit Seif (Graph 1) gives an example of an "expected" grouping of time periods: Replicates from separate time periods form clusters together, and the closer time periods then form larger clusters. For example, the 45-day replicates cluster together at the lowest clustering level, then form a larger cluster with the 75-day replicates; the two then cluster with day-105 replicates. Day-120 replicates similarly cluster together, then join the day-135 cluster.

Graph 1. Sample (time-point) hierarchical clustering for Nebeit Seif variety

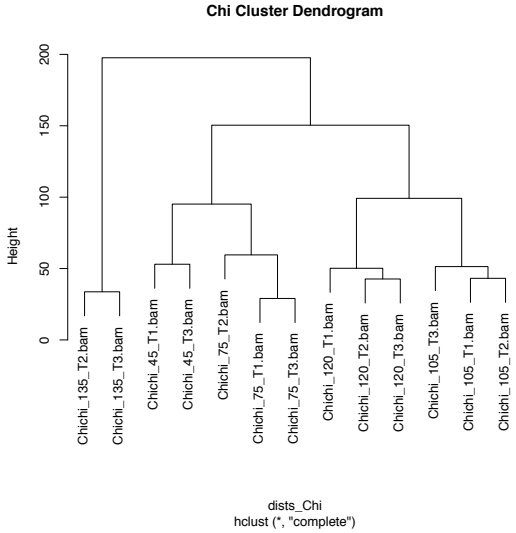


Not all of the varieties clustered as neatly into time periods, however. Overall, h-clustering of times for the varieties showed three main patterns of clustering:

Pattern 1 - By Time: The time periods cluster together, and then into larger clusters that make sense chronologically, as in Nebeit Seif, above. The Lulu sample follows a similar pattern, differing only slightly, in how the higher groups are organized: For example, the day-75 cluster groups into a larger cluster with the day-105 cluster, instead of with the day-45 cluster. (Interestingly, these two varieties were the only ones with less than three replicates for all time periods, which may warrant further investigation.)

Pattern 2 - Outlier: The time periods cluster as might be expected, as above, but there is a notable outlier. This was true for the ChiChi variety, in which the day-135 replicates clustered very distantly from all other groups (Graph 2).

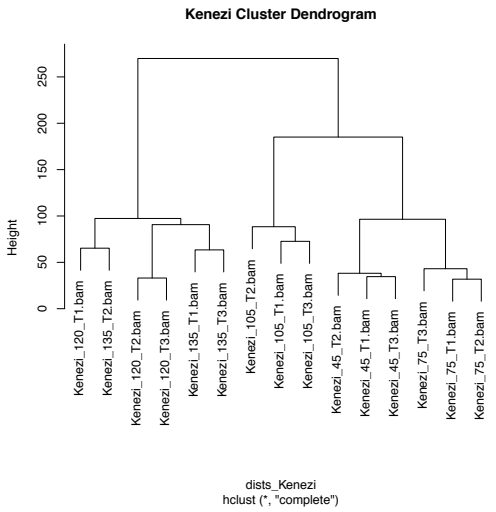
Graph 2. Sample (time-point) hierarchical clustering for ChiChi variety



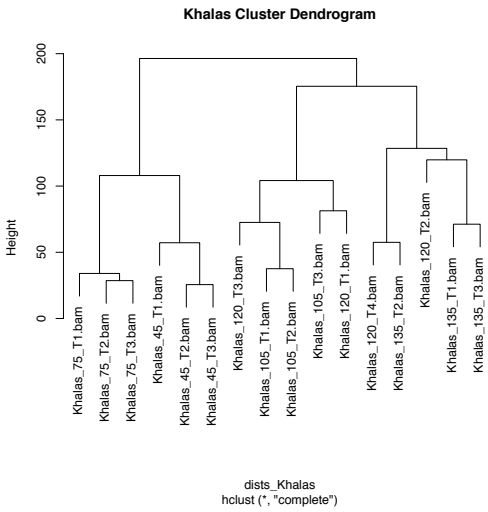
Pattern 3-"Umbrella" Stages: Some of the individual groups of time replicates fail to group together at the lowest clustering level. Instead, they can be found grouping with replicates from other time periods. This suggests that the time periods in these varieties do not represent distinct developmental stages. Instead, there may be "umbrella" developmental stages that encompass more than one of the sampled time periods. This pattern characterized the Kenezi and Khalas varieties:

- **Kenezi**: The 135-day and 120-day replicates can be found interspersed together in the lowest level of clustering (Graph 3).
- **Khalas**: Three of the time periods' replicates can be found interspersed at the lowest cluster level. Some 120-day replicates are interspersed with 105-day replicates, while the remaining 120-day replicates are interspersed with the 135-day replicates. This suggests two umbrella stages (105-120 and 120-135) or an overall umbrella developmental stage spanning 105 to 135 days (Graph 4).

Graph 3. Sample H-cluster for Kenezi



Graph 4. Sample H-cluster for Khalas



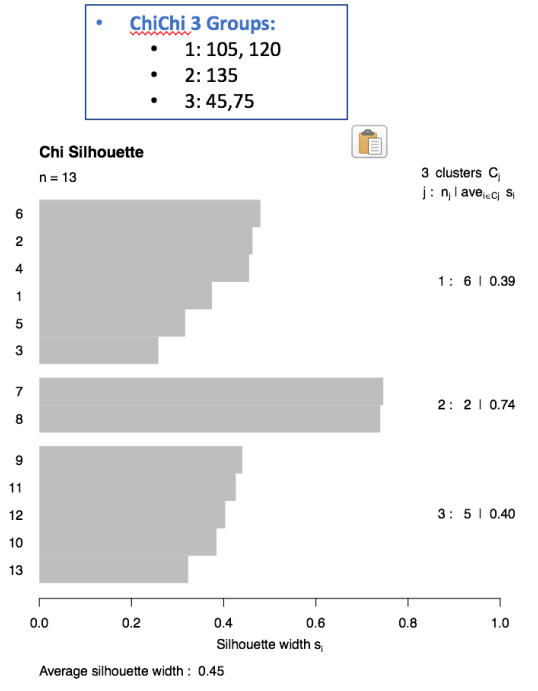
Results: Developmental Stages via K Grouping

Despite the clustering of time-period replicates together in most varieties, developmental stages could still span multiple time periods in those varieties. For example, though day-45 and day-75 replicates cluster separately at the lowest levels, their groups may be close enough to label them one developmental stage. To further elucidate the developmental stages, the variety clusters were divided into k numbers of groups. Different values of k were evaluated, based on a visual inspection of the h-clustering dendrograms, and the best fit decided by looking at silhouette plots, comparing average silhouette width and the size of groups that failed to fit into any of the proposed k groups.

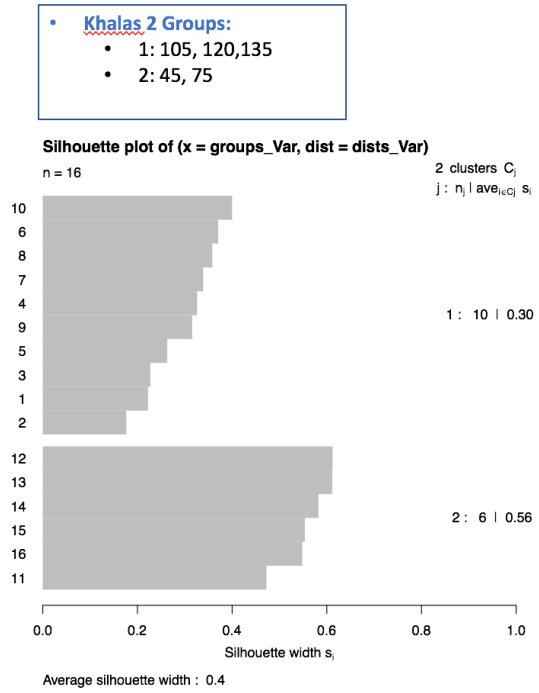
The most obvious grouping for ChiChi would be into the five time periods, which separate so neatly in the dendrogram. A k=5 grouping indeed worked well, producing an average silhouette width of 0.44, and the groups are as expected: splitting neatly on the five time periods. Even better, however, was 3 groups for ChiChi, with an average width of 0.45, and again showing the strong outlier of time 135 (Graph 5).

Unsurprisingly, 5 groups did not work as well for Khalas, producing an average width of 0.37, with some replicates in their own clusters and one group showing a width of 0. Echoing the h-cluster dendrogram, time periods in the k=5 grouping did not necessarily cluster together. A split of k=2 groups worked best for Khalas, splitting between the early times and the later, interspersed times, and giving an average width of 0.4 (Graph 6).

Graph 5. Silhouette plot for ChiChi sample groups



Graph 6. Silhouette for Khalas sample groups

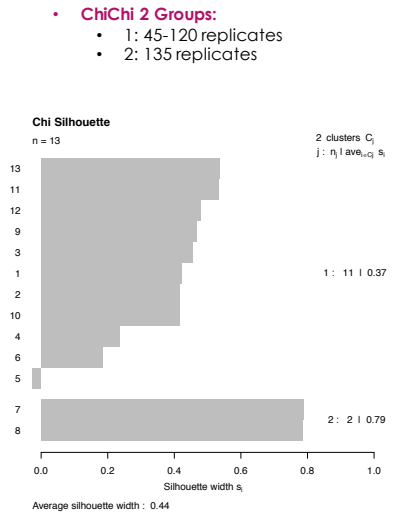


Note, here and elsewhere, the groups ChiChi and Khalas are used to represent the range of clustering patterns. Khalas epitomizes the "umbrella" clustering of time periods, while ChiChi represents both the expected time groupings and the occurrence of an outlier group.

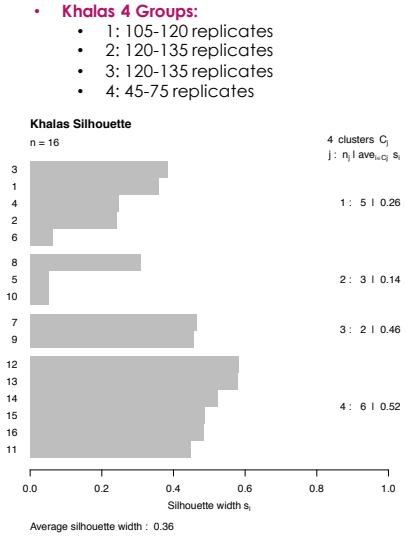
Results: Alternative time-period clustering

Alternative time-period clusterings were explored (Graphs 7-8). ChiChi clustered nearly as well into two groups, with an average silhouette width of 0.44, just 0.01 less than with 3 groups. Khalas clustered reasonably well into 4 groups, with an average silhouette width of 0.36. This is slightly less than for five groups, but this grouping does not involve any clusters with width 0 or that consist of only 1 replicate. Moreover, these alternatives will become important later on.

Graph 7. Silhouette for alt ChiChi sample grouping



Graph 8. Silhouette for alt Khalas sample grouping



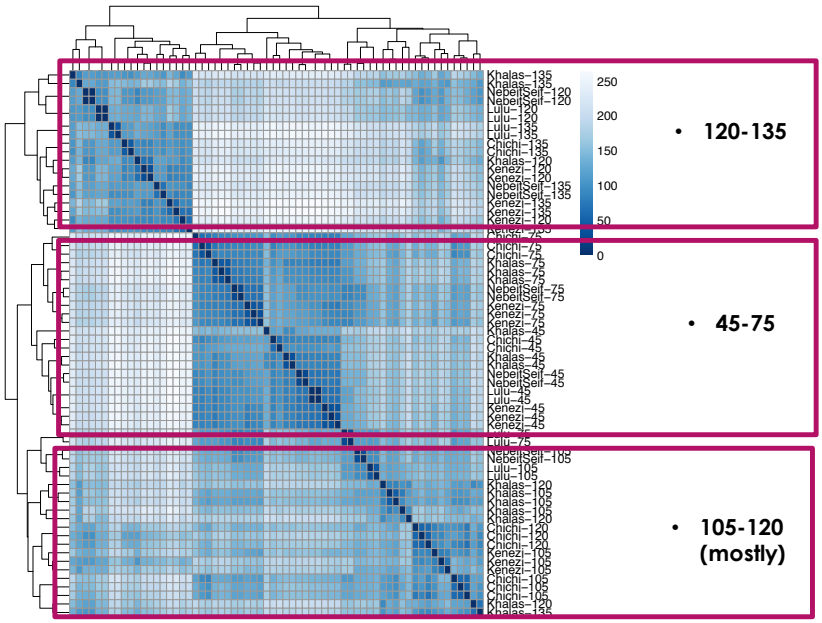
Results: Multifactor experiment sample clustering

A multifactor differential expression analysis was also applied to all times and all varieties. Results showed that time was, as predicted, a more important factor in the clustering than was variety. Reading through the dendrogram of the multifactor experiment shows that the samples cluster by time into three main groupings:

- **1) Late: 120-135:** Contains day-135 replicates (from Khalas, Lulu, and Kenezi) and day-120 replicates (from Nebeit Seif, Khals, and Kenezi).
- **2) Early: 45-75:** Contains day-45 replicates (from all five varieties) and day-75 replicates (from ChiChi, Khalas, and Nebeit Seif).
- **3) Mostly Middle: 105-120:** Contains day-105 replicates (from all five varieties) and day-120 replicates (from Khalas and ChiChi), with stray replicates from two other times (Lulu 75 days and Khalas 135 days).

This division is best visualized in a sample-distance heat map (Graph 9), as the dendrogram is quite dense.

Graph 9. Sample-Distance Heat Map: All Varieties, All Times



Methods: Clustering of Genes, Interpreted by Developmental Stage

Clustering of the genes was also performed, and the two types of clustering (by time period and by genes) help make sense of one another. That is, the individual time periods and umbrella clusters of time periods may represent developmental stages. If so, one would expect genes to turn on or off in patterns consistent with those stages; thus, genes might be expected to cluster into groups that are over-expressed during one development stage and underexpressed in other stages — and vice versa. Heat maps of gene expression provide a first look into how gene expression may be overlaying the developmental stages in this way. Further evaluations of these conclusions are provided via hierarchical clustering and silhouette plots.

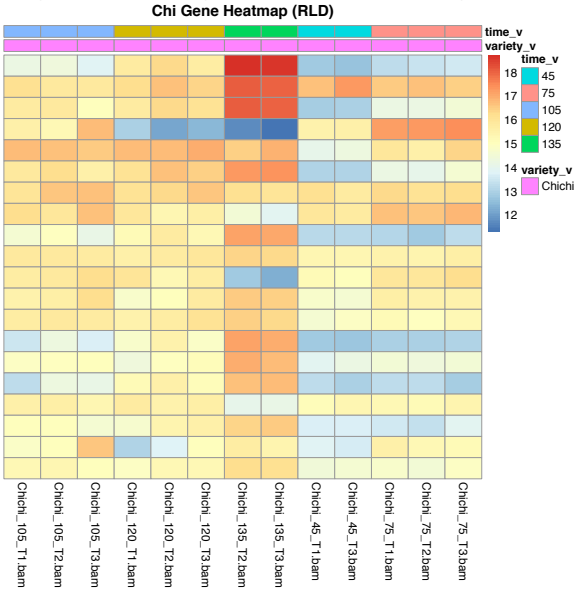
Methods: Gene expression heat map

The gene expression heat maps show gene expression data from the count matrix. DESeq2 provides procedures for producing these maps using various transformations of the expression data. Gene heat maps were produced for this experiment using both the regularized-log (RLD) and variance-stabilized (VSD) transformations (see Methods) of the expression data. The two transformations showed mostly the same patterns, with slightly higher overall expression portrayed using VSD. The RLD versions are shown here, because they better highlight the contrast between expression levels in different time periods.

Results: ChiChi Gene Clustering

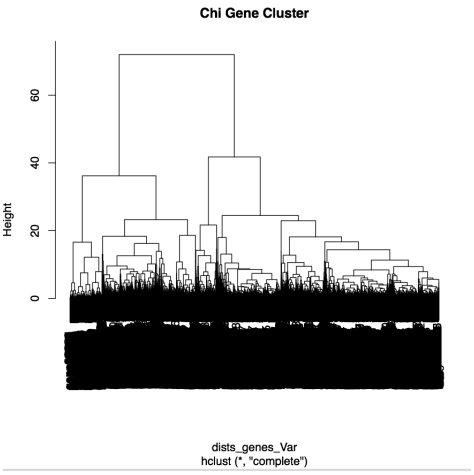
The gene-expression heat map for variety ChiChi (Graph 10) shows marked increase in expression for a particular set of genes in the 135-day group. This heat map also shows a marked "cool" area, of low gene expression, covering the 135-day period, bleeding over somewhat into day-120. The rest of the genes seem to show lower levels of the same pattern: either higher expression in 135, with relatively lower expression elsewhere, or lower expression in 135, with relatively higher expression elsewhere.

Graph 10. Gene expression (RLD) heat map for variety ChiChi



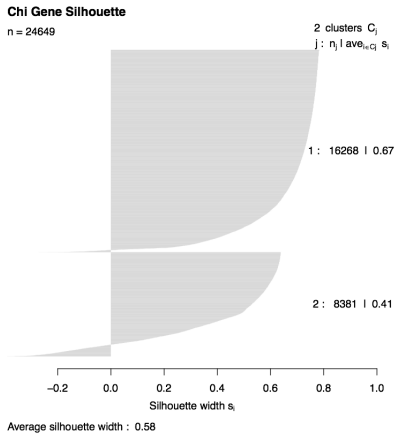
This suggests that the genes for variety ChiChi could cluster into two groups: 1) those with high expression in time 135 and low expression elsewhere, and 2) those with lower expression in 135 and relatively higher expression elsewhere. That is, one might predict two major developmental stages: time 135 and everything preceding it. In fact, hierarchical clustering of the genes bears out a 2-group clustering of ChiChi genes (Graph 11):

Graph 11. Hierarchical clustering for differentially expressed genes in ChiChi



A silhouette plot of ChiChi's gene cluster into k=2 groups (Graph 12) provides further support for this interpretation. This is the best fit of several k-values investigated, providing the highest average silhouette width (0.58) and smallest ungrouped tails (left-pointing parts of the plot):

Graph 12. Silhouette for k=2 grouping of differentially expressed genes, ChiChi



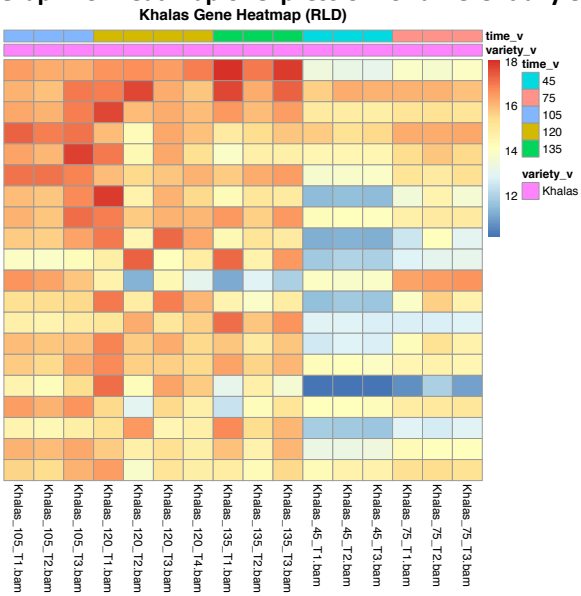
Results: ChiChi's 2 developmental stages

The clustering of genes, in combination with the time-period clustering above demonstrating an outlier 135-day grouping, suggests two major developmental stages for the ChiChi variety: 135 days and everything else. That is, though the earlier time-point replicates cluster into 5 separate groups, more significant genetic changes happen in the 135-day period than occur between any of those earlier clusters. Thus, the alternative time-period clustering of ChiChi into 2 groups above (Graph 7) fits with the gene-clustering data.

Results: Khalas gene clustering

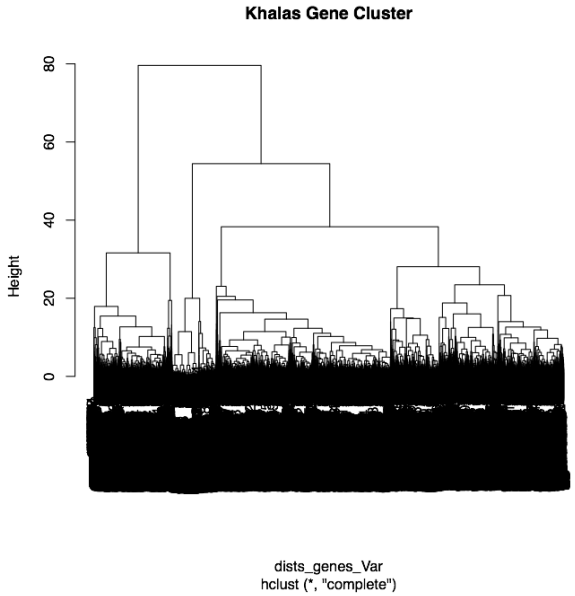
At first glance, the gene-expression heat map for Khalas (Graph 13) also shows evidence for two groupings of genes, in concordance with the time-period clusterings described earlier. That is, it suggests the umbrella group of 105-135 may divide the genes into two expression patterns: 1) those that are overexpressed in the 105-135 umbrella and underexpressed elsewhere (in the lower times of 45-75), and 2) those that are underexpressed in the umbrella and overexpressed elsewhere:

Graph 13. Heat map of expression for differentially expressed genes (RLD), Khalas variety

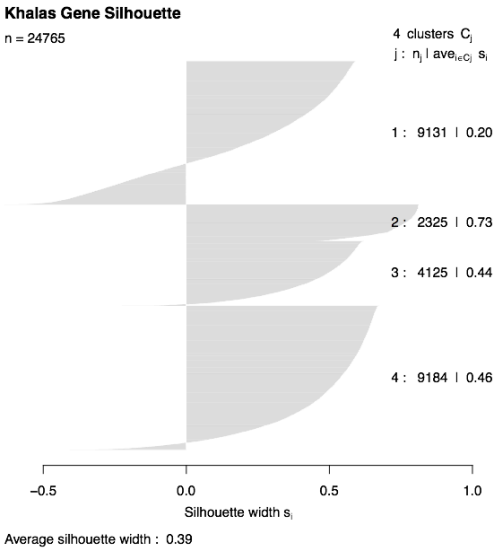


Clustering and k=2 grouping of the genes for Khalas, however, do not support this interpretation, producing a silhouette with low average width and large tails of non-grouping genes. An inspection of the h-clustering dendrogram and experimentation with different k-values (from 2 to 8) pointed to k=4 as the best grouping of Khalas genes, with an average silhouette width of 0.39 (Graphs 14-15).

Graph 14. Silhouette for k=2 grouping, Khalas genes



Graph 15. Silhouette for k=4 grouping Khalas genes



Results: Re-evaluating Khalas gene heat map

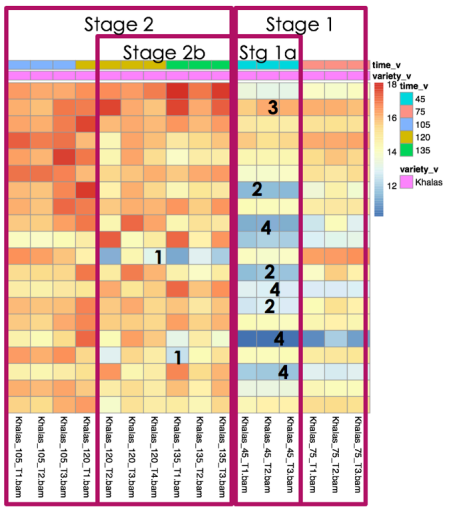
With this information, the Khalas gene heat map may be viewed in a different light, and a hypothesis made about how the developmental stages and how genes are clustering. A division of the Khalas gene heat map into either four overlapping stages or two large stages that contain sub-stages makes sense. The latter is described here (visualized in Graph 18):

- Stage 1) 45-75 Days
- Stage 1a) 45 Days
- Stage 2) 105-135 Days
- Stage 2b) 120-135 Days

Groupings of genes then overlay these developmental stages, splitting into the following sets of genes:

- Group 1) Genes underexpressed in Stage 2b, but not elsewhere
- Group 2) Genes underexpressed in Stage 1a, but not elsewhere
- Group 3) Genes highly overexpressed in Stage 2, expressed at medium level in Stage 1
- Group 4) Genes underexpressed in Stage 1, expressed at medium-to-high levels in 2

These gene groupings and developmental stages can be seen in the annotated heat map for Khalas (Graph 18). This set of overlapping stages doesn't perfectly match with the k=4 grouping of time periods for Khalas found earlier, likely due to the overlaps. In fact, it fits better with the k=2 grouping of Khalas time periods, with Stage 1 referring to the earlier time periods and Stage 2 referring to the umbrella 105-135 period.



Graph 18. Khalas Gene Heat Map Re-Visited

Proposed developmental stages are outlined and labeled above heat map. Proposed gene groups are numbered on the heat map, representing horizontal slices of the map.

Methods: Gene Ontology Enrichment

The lists of differentially expressed genes for each variety, and for all varieties and times in the multi-factor design, provided the starting point for the Gene Ontology (GO) term enrichment analysis. Because no GO annotation was available for the date palm organism, the GO terms for each gene were obtained using the InterProScan program on the NYU HPC Mercer cluster. Interproscan is a Java-based software that will search a given protein sequence against family and domain databases. It can also return the associated GO terms for the proteins (Jones et al., 2014)

Several BASH scripts were written to obtain these GO terms using InterProScan. First, because InterProScan returns GO terms identified by protein id's, a BASH script was written to match the gene id's from the date palm count table to protein id's (Appendix 4). This was done using the date palm GFF. Because the GFF lists gene, rna and protein ids in different lines, genes were matched to their children rnas, and then rnas to their children proteins to make a list of gene ids paired with their corresponding protein id's.

Because of the size of the date palm protein FASTA file, another BASH script split the FASTA file into 100 sub-files to submit to InterProScan (Appendix 5) as a PBS array job (Appendix 6). Another script extracted the protein id's and GO terms from the InterProScan results, found the matching gene id's using the gene-protein list produced as described above, and produced a file readable by R as a GOStats goFrameData object (Appendix 7). Finally, a BASH script used the above file to produce universe and gene lists that consisted of only those genes that had corresponding GO terms (Appendix 8).

These lists and the goFrameData object were used as input for the GO term enrichment R script (Appendix 9). In R, a function was written that would be able to run the GOstats GO-term enrichment workflow on each variety of date palm. The function called the GOstats hypergeometric test to find enriched GO terms, as the hypergeometric tests evaluates the chances of finding a given number of GO term matches in a given set (universe) by chance (Falcon and Gentelman, 2007). The R function that applies the GOstats method was written to run on either LRT- or Ward-produced gene lists, depending on a Boolean argument.

Methods: Calling GO-enrichment function and contrasting GO-term profiles

A loop was written that would call the function on each date palm variety's gene list. That same loop also extracted the top 10 GO terms for each variety and the top 50 GO terms for each variety. Here, "top 10" and related terms refer to the first "n" terms returned by the GO term enrichment function, which are those with the "n" lowest p-vals. These were compiled into comprehensive vectors (one containing all the top 10s for all 5 varieties, and one containing all the top 50s for all 5 varieties). Then, using R's table and sort functions, the most common occurrences in those two lists were found (Appendix 9). This was done in an effort to characterize how similar or different the GO term profiles were for the five varieties.

Methods: Finding Gene Ontology Enrichment for Non-Overlapping Variety Genes

Finally, based on feedback on the presentation for this project, an attempt was made to find the Gene Ontology terms that characterized those differentially expressed genes that were non-overlapping between two date palm varieties. That is, the differentially expressed genes for Variety A and Variety B were compared using the R command "setdiff" to find those genes that appeared in Variety A's list of differentially expressed genes but not in Variety B's list, and vice versa (Appendix 10). GO-term enrichment using the GOStats hypergeometric test was then applied to those non-overlapping gene lists. The resulting top enriched terms for several varieties in comparison with Khalas were obtained (Appendix 10). (Khalas was chosen as a primary comparison it is a well-known and popular variety, which seemed to have more information available on fruit traits.) For example, top-10 lists of GO terms were found for "ChiChiNotKhalas" (those genes differentially expressed in ChiChi but not in Khalas) and for "KhalasNotChiChi" (the reverse).

With these non-overlapping GO-term lists in hand, a search was made for published studies of date palm fruit traits. This was done in the hopes that the non-overlapping GO-term lists might help explain some of the salient traits of the different fruit varieties studied here. A hypothesis was made linking a differential trait of ChiChi (fruit weight) to the non-overlapping GO-term list for ChiChi in comparison to Khalas (see Results below). The corresponding gene list for this GO-term list was then produced. This was a list of genes that both a) are associated with a term in the non-overlapping GO term list and b) are present in the ChiChi-not-Khalas gene list. These were found by searching through the original goFrameData object file for all genes associated with those GO-term IDs ("GOgenedata_d_3rdrun.txt") and then choosing only those resulting genes that appeared in the original non-overlapping gene list ("ChiChiNotKhalas"). This was all done via a BASH script (Appendix 11).

Results: Gene Ontology Enrichment

GO term enrichment profiles show that the different varieties are characterized by different top 10 GO term profiles. However, the broader collections of GO terms associated with each sample are more similar. First, a look at the top 10 enriched GO terms for ChiChi and Khalas (Table 3), the two varieties used in this paper to best represent the range of clustering patterns among the varieties. These terms were derived from the experiment using the Wald test.

Table 3. Top 10 enriched GO terms for ChiChi, Khalas (Wald hypothesis test)

ChiChi Top 10: Wald Test

Term	Pvalue
single-organism cellular process	4.96783E-09
small molecule metabolic process	5.56742E-07
generation of precursor metabolites and energy	1.97427E-06
single-organism process	2.10939E-06
hydrogen ion transmembrane transport	8.61672E-06
ATP hydrolysis coupled proton transport	1.71759E-05
energy coupled proton transmembrane transport, against electrochemical gradient	1.71759E-05
nucleobase-containing small molecule metabolic process	1.88027E-05
phosphorus metabolic process	3.20313E-05
phosphate-containing compound metabolic process	3.91603E-05

Khalas Top 10: Wald Test

Term	Pvalue
regulation of catalytic activity	4.03805E-05
response to organic substance	5.26204E-05
response to hormone	5.26204E-05
response to endogenous stimulus	5.26204E-05
regulation of transferase activity	6.52281E-05
ATP hydrolysis coupled proton transport	9.10824E-05
energy coupled proton transmembrane transport, against electrochemical gradient	9.10824E-05
regulation of molecular function	9.25346E-05
regulation of cyclin-dependent protein serine/threonine kinase activity	0.00014656
regulation of cyclin-dependent protein kinase activity	0.00014656

The experiment that used the LRT time-series design produced different top 10 lists of GO terms (Tables 4).

Table 4. Top 10 enriched GO terms for ChiChi, Khalas (Likelihood-ration hypothesis test)

ChiChi Top 10 GO (LRT)

Term	Pvalue
single-organism cellular process	5.01297E-10
small molecule metabolic process	8.05627E-09
organonitrogen compound metabolic process	3.42595E-08
generation of precursor metabolites and energy	1.17317E-06
organonitrogen compound biosynthetic process	4.91325E-06
DNA packaging	7.682E-06
organophosphate metabolic process	1.69909E-05
chromatin assembly or disassembly	1.76848E-05
nucleosome organization	2.17559E-05
nucleosome assembly	2.17559E-05

Khalas Top 10 GO (LRT)

Term	Pvalue
single-organism process	1.97099E-06
single-organism cellular process	0.000289568
small molecule biosynthetic process	0.000321363
carboxylic acid biosynthetic process	0.000323663
organic acid biosynthetic process	0.000323663
disaccharide metabolic process	0.000340105
regulation of catalytic activity	0.000432053
cellular amino acid biosynthetic process	0.000436525
oligosaccharide metabolic process	0.000522966
regulation of molecular function	0.001082269

While these lists show individual similarities, more interesting is how similar or different the top 10 profiles are for all the varieties. The table and sorting technique described in Methods above was performed to provide this information (Table 5).

Table 5.

Most common terms in all top 10s

Term	No. Occurrences
single-organism cellular process	5
organonitrogen compound biosynthetic process	3
organonitrogen compound metabolic process	3
small molecule metabolic process	3
amide biosynthetic process	2
cellular amide metabolic process	2
peptide biosynthetic process	2
peptide metabolic process	2
protein localization	2
single-organism process	2

The top 10 terms, then, in each variety differ significantly from one another.

While the top 10 terms for each variety may differ, it is possible, and even likely, that a broader collection of "top" terms from each variety would show more similarity. Finding the commonality among *all* terms for each variety would likely prove of little use, since a term could easily appear in all five lists simply by occurring at position 1 in some groups and position 171 in others, for example. Therefore, a more meaningful, but still broader method was chosen: Find the most commonly occurring terms in the collection of top 50 terms from each variety (Table 6).

Table 6.

Most common terms in all top 50s

Term	No. Occurrences
single-organism biosynthetic process	5
single-organism cellular process	5
small molecule metabolic process	5
carboxylic acid metabolic process	4
establishment of protein localization	4
nucleobase-containing small molecule metabolic process	4
nucleoside phosphate metabolic process	4
nucleotide metabolic process	4
organic acid metabolic process	4
oxoacid metabolic process	4

Results: Gene Ontology Enrichment for Non-Overlapping Variety Genes

Enriched Gene Ontology terms were found for the non-overlapping gene lists of several varieties in comparison with the Khalas variety. Table 7 shows the resulting GO list for those differentially expressed genes found in ChiChi but not Khalas; Table 8 shows the list for those differentially expressed genes found in Khalas but not ChiChi.

Table 7. Top 10 enriched GO terms for 'ChiChi-Not-Khals'

ChiChi-Not-Khalas Go Terms

Term	Pvalue
peptide metabolic process	1.46914E-29
peptide biosynthetic process	1.58708E-29
translation	2.15046E-29
cellular amide metabolic process	2.51567E-27
amide biosynthetic process	3.37244E-27
organonitrogen compound biosynthetic process	1.79589E-21
organonitrogen compound metabolic process	1.32622E-17
photosynthesis	6.50616E-14
cellular protein metabolic process	2.60792E-10
gene expression	4.75947E-10

Table 8. Top 10 Go Terms for 'Khals-Not-ChiChi'

Khalas-Not-ChiChi Go Terms

Term	Pvalue
L-methionine salvage	0.01347049
amino acid salvage	0.01347049
L-methionine biosynthetic process from methylthioadenosine	0.01347049
protein folding	0.014197433
tRNA processing	0.022858757
L-methionine biosynthetic process	0.025719093
lipid A metabolic process	0.025719093
tryptophanyl-tRNA aminoacylation	0.025719093
gas transport	0.025719093
lipooligosaccharide metabolic process	0.025719093

The search for specific data on differential traits among the fruit varieties studied here identified a promising study by Al-Abdoulhadi et al. (2011). This analysis of three varieties — Khalas, ChiChi (or "Sheshi"), and Reziz — found the following comparative traits for two varieties of interest in the current project:

Khalas:

- Maximum length
- Least fruit moisture
- Lightest color (yellow)

ChiChi (Sheshi)

- Highest fruit breadths
- Highest weight

Examining these traits, in light of the non-overlapping GO terms found for ChiChi vs. Khalas, at least one hypothesis can be made for a connection between GO terms and differential traits. This can be done for the greater weight found in ChiChi (<8 grams for small fruits, > 11 grams for larger fruits) vs. Khalas (<7 grams for small fruits, >10 grams for large fruits) (Al-Abdoulhadi et al., 2011).

As Table 7 shows, ChiChi's non-overlapping GO terms, in relation to Khalas, are enriched in terms related to protein synthesis and metabolism ("translation," "peptide biosynthetic process", "translation," etc.) and amide synthesis and metabolism. It may be proposed, therefore, that the increased weight of ChiChi fruits relative to Khalas is related to a greater dedication to production of protein tissues. Furthermore, ChiChi also shows non-overlapping GO enrichment in nitrogen synthesis and metabolism (e.g., "organonitrogen compound biosynthetic process"). Nitrogen has been shown to play a role in cell division in developing fruit tissues (Yara, "Role of Nitrogen"), so this relative GO enrichment may also help explain the greater weight of ChiChi, in that this variety devotes more resources to generating additional tissue. Finally, ChiChi's non-overlapping terms of gene expression and photosynthesis may, in general, play a contributing role in the greater relative production of tissue in the ChiChi variety, compared to Khalas.

With this finding, the genes associated with those traits for ChiChi were compiled into a file, "go_gene_b_cnk.txt," for potential use in further analysis.

DISCUSSION

Differential Expression

As shown in Table 2, the time-series LRT approach produced greater numbers of differentially expressed genes than the Wald test. This is to be expected, as the LRT design compares differences in multiple time points against one another, not just against a single reference time. Both hypothesis designs, however, produced large numbers of differentially expressed genes. This is partly due to the number of factor levels: five different time points for each variety. A test-run of DESeq2 on only two time points for the ChiChi variety (105 and 120) produced a relatively smaller gene list of about 5,000 genes, which is still large. Given that these are developing fruit, then, it is likely that a large number of genes are over- and under-expressed as development proceeds. That is, development of fruit is a very active period genetically.

That large number, combined with the developmental-stage and gene clustering results found, suggests that this data reflects the expression patterns of important developmental genes for the date palm fruit. This supports the conclusion that the differentially expressed genes examined here do in fact represent the genetic basis for the differential fruit traits among the five varieties considered.

Time Point Clustering/Developmental Stages

This analysis of gene expression data in developing date palm fruit answered many of the questions of interest about time-period clustering and developmental stages. Clustering of time points showed that, in general, replicates for different time points do group together. However, this clustering also showed evidence for developmental stages that spanned multiple time points, what this paper has referred to as "umbrella" stages.

Overall, when clustering both time points and cultivar variety in a multi-factor experiment, the samples cluster mostly by time. Here, again, however, the time points do not necessarily neatly group, with 120-day replicates, in particular, appearing in multiple groups.

Clustering of genes showed that gene groups can further elucidate the developmental stages. Examining gene clusters and heat maps shows which time periods, and spans of time periods, likely experience increased and decreased expression of different sets of genes over a fruit's developmental process. This analysis revealed the likely presence of 2 major developmental stages for variety ChiChi and 2 subdivided developmental stages for variety Khalas.

The examination of Khalas, in particular, shows how clustering of genes into expression groups, in combination with an examination of the clustering of time periods, can elucidate developmental stages. In summary, as Graph 18 shows, the developmental timeline of Khalas fruit may be divided into two overarching stages, which reflect the clustering information obtained for Khalas' time periods. These two stages, as shown by the information obtained about gene groupings, divide into early and late periods: One group of genes is expressed at a moderate level over Stage 1, while another group is suppressed in the early part of Stage 1, but not later. One group of genes is overexpressed over the course of Stage 2, relative to Stage 1, while another group is suppressed in the second half of Stage 2, but not the first half of Stage 2.

That characterization of the Khalas developmental timeline differs markedly from the one proposed here for ChiChi, which shows two major developmental periods (time 135-days and everything earlier) and two gene groups (those overexpressed at 135 days and relatively underexpressed elsewhere, and those underexpressed in 135 and relatively overexpressed elsewhere). These proposed, differing developmental stages demonstrate one way in which the genetics of fruit development may help explain differences in date palm fruit variety.

GO Term Enrichment

The GO term enrichment analysis demonstrated differences in output from the Wald vs. LRT designs. This is to be expected, as the time-series experiment a) produced a greater number of differentially expressed genes and b) showed difference in expression among all time periods, not just in reference to Day 45.

GO-term-enrichment analysis also demonstrated further expression differences among the 5 varieties. In fact, the top 10 GO terms for the varieties appear fairly dissimilar. The list of "top 10 top 10s" (Table 5) shows that the most significantly enriched GO terms are quite different among the different varieties. That is, only one term (the very general "single organism cellular process") occurs in the top 10 lists of all five varieties. Three terms, including two involving nitrogen, occur in only 3 varieties' top 10 lists. The remaining "common" terms each occur in only 2 varieties' top 10 lists.

However, more comprehensive lists of enriched GO terms for the different varieties show greater similarity. In the list of most common terms among all varieties' top-50 lists (Table 6), all terms occur at least 4 times. This demonstrates that while the very most significant GO terms for each variety may differ, the varieties are more similarly enriched in GO terms in a broader sense. These differing top terms may be those most important in producing the different fruit traits found among the varieties. The similarity in the broader list of terms likely reflects the fact that these are the same general developmental processes in the same species.

The lists of enriched GO terms consist largely of terms that would be expected to be important in developing fruit tissue: those related to cell division; those related specifically to fruit-tissue development, such as acid levels; and general terms for organismal processes. A reading of the top 10 lists for individual varieties, as well as the lists of most common terms among the varieties ("top top-10s" and "top top-50s") characterizes what types of terms appear. The enriched terms tend to belong to three main groups:

- **General Terms:** Very broad terms — like "single-organism cellular process" and "single-organism biosynthetic process" — appear frequently, particularly in the collections of the most common terms among all varieties' lists. These terms do not provide much valuable information, other than the trivial observation that this is a single organism, but neither do they contradict what might be expected in a developing fruit.

- **Cell Division/Growth:** Many of the enriched terms relate to cell division and growth, particularly in fruit development. This would be expected as development of the date palm fruit tissue would clearly involve cell division and growth. Terms involving nitrogen, such as "organonitrogen compound metabolic process," make sense as nitrogen is known to play an important role in cell division in young fruit tissue (Yara, "Role of Nitrogen").

  Terms involving cyclin-dependent protein kinases also appear multiple times, for example "regulation of cyclin-dependent protein serine/threonine kinase activity."

These types of kinases have been shown to play a role in the cell cycle, and have been associated with dividing tissues in developing fruit (Joubes et al., 2001). Finally, many terms involved with DNA replication during the cell cycle appear, such as "DNA packaging," "chromatin assembly or disassembly," and "nucleosome assembly."

- **Fruit Development:** In addition to some of the cell-cycle-related terms above, several other frequently appearing GO terms relate to fruit development. Changes in proton transport and associated changes in acidity have been shown to play a role in ripening in many fruits (Terrier et al., 2001). This helps explain why terms like "ATP hydrolysis coupled proton transport" and its parent term "Energy coupled proton transmembrane transport, against electrochemical gradient" appear in the GO term lists.

Above, a hypothesis was also made linking the "non-overlapping" ChiChi GO terms, in reference to Khalas, to at least one trait found for ChiChi fruit. The non-overlapping GO terms for ChiChi were found to be strongly related to protein synthesis and tissue synthesis in general. This was linked to the greater weight found in ChiChi fruit by Al-Abdoulhadi et al. (2011).

It must be admitted that this hypothesis is very speculative, and that it is difficult to draw a straight line from the non-overlapping GO terms found for ChiChi to these specific differential varietal traits. In fact, the difference in weight reported for the fruit varieties may not be large enough that it would explain the difference in expression of protein-synthesis genes. That is, the GO-term differences and trait differences may not necessarily be linked, though the hypothesis is suggestive and could point to further avenues for research.

In general, it may be difficult to align GO terms with specific fruit traits, even when those traits can be identified (Flowers J, personal communication, 2016). Therefore, the PI for the original RNA-seq experiment for this project suggested an alternative method to link differential expression to varietal fruit traits, by searching for differential expression among genes related to a pathway involved in fruit color. A proposed outline for this additional/supplemental experiment is included in "Next Steps," below.

Significance

The types of GO terms found suggest that the patterning and differential expression analyzed in this project do indeed refer to the genes important in date palm fruit development. This is important, because the goal of this analysis was to increase knowledge of the genetic basis for the developmental differences of date palm fruit varieties that are characterized by different traits. Ultimately, this analysis could add needed information on the gene-expression basis for different varietal traits in an important food crop, one that faces ongoing cultivation challenges.

Next Steps/Future Research

This analysis suggests several additional examinations and refinements. Differential expression and clustering might be done for individual time points against other time points. This could be done, for example, for times 120-135 in Khalas, to find if they form one or two umbrella

developmental stages, or by comparing times in ChiChi excluding 135 to find if any significant changes occur in those earlier time periods. DESeq2's "Contrasts" workflow could be used to investigate these questions, as it allows for simultaneous comparisons among three different factor levels. "Contrasts" might also be used to investigate differences among sets of three varieties, as an alternative method of finding those differentially expressed genes that do not overlap between varieties (in place of using setdiff on the results from each variety).

The observation that the varieties with lower numbers of time replicates appeared to cluster more "neatly" could be further investigated by running the analysis on only 2 replicates per time point for all varieties. GO term enrichment could be analyzed in a more granular fashion against the different clustering patterns seen in the varieties.

Proposed workflow for gene-color differential expression analysis

Based on PI feedback, an additional or supplemental differential expression analysis could be performed to identify genes related to a specific fruit trait: color. The lab that produced this RNA-seq analysis has discovered a link between alleles in the flavonoid/anthocyanin pathway and color in date palm, responsible for producing yellow and red varieties (Hazzouri et al., 2015)(Flowers, personal communication, 2016). The proposed new analysis would identify the genes in the date palm associated with the key enzymes in this pathway, then look for differential expression of those genes between the yellow-colored (Kenezi, possibly Khalas) and red-colored varieties (all others). Here are the proposed steps and associated methods in such an analysis, which may be performed in the future:

Proposed fruit-color differential expression analysis work-flow

- Find differentially expressed genes for Kenezi variety vs. other, red varieties
    - Do one-factor differential expression analysis (variety as factor) of all samples
    - Use Kenezi as reference level
    - Produce differentially expressed gene list
- Get date palm gene list for flavonoid/anthocyanin pathway
    - Find genes for key enzymes in apple flavenoid/anthocyanin pathway in Espley et al. (2007)
    - Identify date palm homologs for those genes using date palm flavenoid biosynthesis map (KEGG, "Flavanoid biosynthesis")
    - Match the NCBI gene IDs thus found to the gene-list ids from differential expression analysis (using Perl or BASH scripting)
- Print list of differentially expressed genes associated with fruit color via the flavonoid/anthocyanin pathway
- Potentially: Perform GO-term analysis on those genes

Additional PI Feedback

Finally, based on feedback from the principal investigator, the findings in this report could be used to shape further analysis of the date palm fruit RNA-seq data obtained in the original experiment.

References

Al-Abdoulhadi IA, Al-Ali S, Khurshid K, Al-Shryda F, Al-Jabr AM, Ben Abdalah A. 2011. Assessing fruit characteristics to standardize quality norms in date cultivars of Saudi Arabia. Indian Journal of Science and Technology [Internet] [cited 14 Dec 2016]; 4(10):1262-1266. Available from: http://www.moa.gov.sa/files/alhasa/26.pdf

Alberts B, Johnson A, Lewis J, et al. 2002. Molecular Biology of the Cell. 4th edition. Chromosomal DNA and Its Packaging in the Chromatin Fiber. New York: Garland Science. [Internet] [cited 6 Dec 2016]. Available from: https://www.ncbi.nlm.nih.gov/books/NBK26834/

AmiGO 2. [Internet]. ATP hydrolysis coupled proton transport. [cited 5 Dec 2016]. Available from: http://amigo.geneontology.org/amigo/term/GO:0015991

Binomial distribution. [Internet] [cited 6 Dec 2016]. Available from: https://en.wikipedia.org/wiki/Binomial_distribution

Carlson M and Pages H. 2016. AnnotationForge: Code for Building Annotation Database Packages. R package version 1.16.0.

Chao C, Krueger R. 2007. The Date Palm (Phoenix dactylifera L.): Overview of Biology, Uses, and Cultivation. HortScience [Internet] [cited 2016 Dec 5]; 42(5):1077-1082. Available from: http://hortsci.ashspublications.org/content/42/5/1077.full

D'Alessandro M, Melandri B. 2010. ATP hydrolysis in ATP synthases can be differently coupled to proton transport and modulated by ADP and phosphate: A structure based model of the mechanism. Biochimica et Biophysica Acta (BBA) – Bioenergetics [Internet] [cited Dec 6 2016];1797(6-7):755-762. Available from: http://www.sciencedirect.com/science/article/pii/S0005272810001131

Date palm. [Internet] [cited 6 Dec 2016]. Available from: https://en.wikipedia.org/wiki/Date_palm

Deniz O, Flores O, Aldea M, Soler-Lopez M, Orozco M. 2015. Nucleosome architecture throughout the cell cycle. Nature Scientific Reports [Internet] [cited Dec 6 2016]; 6. Available from: http://www.nature.com/articles/srep19729

Espley R, Hellens R, Putterill J, Stevenson D, Kutty-Amma S, Allan A. 2007. Red colouration in apple fruit is due to the activity of the MYB transcription factor, MdMYB10. The Plant Journal; 49:414-427.

Falcon S, Gentleman R. 2007. Using GOstats to test gene lists for GO term association. Bioinformatics [Internet] [cited 5 Dec 2016]; 23(2):257-8. Available from: https://www.ncbi.nlm.nih.gov/pubmed/17098774

Generalized linear model. [Internet] [cited 6 Dec 2016]. Available from: https://en.wikipedia.org/wiki/Generalized_linear_model

Hazzouri K, Flowers J, Visser H, Khierallah H, Rosas U, Pham G, Meyer R, Johansen C, Fresquez Z, Masmoudi K, et al. 2015. Whole genome re-sequencing of date palms yields insights into diversification of a fruit tree crop. Nature Communications [Internet] [cited 2016 Dec 5]; 6. Available from: http://www.nature.com/articles/ncomms9824

Jones P, Binns D, Chang HY, Fraser M, Li W, McAnulla C, McWilliam H, Maslen J, Mitchell A, Nuka G, et al. 2014. InterProScan 5: genome-scale protein function classification. Bioinformatics [Internet] [cited 7 Dec 2016]. Available from: https://bioinformatics.oxfordjournals.org/content/early/2014/01/29/bioinformatics.btu031.full

Joubes J, Lemaire-Chamley M, Delmas F, Walter J, Hernould M, Mouras A, Raymond P, Chevalier C. 2001. A New Cy-Type Cyclin-Dependent Kinase from Tomato Expressed in Dividing Tissues Does Not Interact with Mitotic and G1 Cyclins. Plant Physiol [Internet] [cited 5 Dec 2016]; 126(4):1403-1415. Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC117141/

KEGG. [Internet]. Flavonoid biosynthesis – Phoenix dactylifera (date palm). [cited 14 Dec 2016]. Available from: http://www.genome.jp/kegg-bin/show_pathway?org_name=pda&mapno=00941&mapscale=&show_description=hide

Lawrence M, Huber W, Pages H, Aboyoun P, Carlson M, Gentelman R, Morgan M, Carey V. 2013. Software for Computing and Annotating Genomic Ranges. PLoS Computational Biology [Internet] [cited 5 Dec 2016]. Available from: http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1003118 [Bioconductor]

Love M, Anders S, Huber W. 2016. Differential analysis of count data – the DESeq2 package. [Internet] [cited 5 Dec 2016]. Available from: https://www.bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.pdf

Love M, Huber W, Anders S. 2014. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biology [Internet] [cited Dec 6 2016]; 15:550. Available from: http://genomebiology.biomedcentral.com/articles/10.1186/s13059-014-0550-8

Mokhtar M, Adawy S, El-Assal S, Hussein E. 2016. Genic and Intergenic SSR Database Generation, SNPs Determination and Pathway Annotations, in Date Palm (Phoenix dactylifera L.). PLoS ONE [Internet] [cited 2016 Dec 5]; 11(7). Available from: http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0159268

Terrier N, Sauvage FX, Ageorges A, Romieu C. 2001. Changes in acidity in proton transport at the tonoplast of grape berries during development. Planta [Internet] [cited 5 Dec 2016]; 2013(1):20-8. Available from: https://www.ncbi.nlm.nih.gov/pubmed/11523652

Wald test. [Internet] [cited 6 Dec 2016]. Available from: https://en.wikipedia.org/wiki/Wald_test

Yara. [Internet]. Role of Nitrogen in Apple Production. [cited 5 Dec 2016]. Available from: http://libguides.willamette.edu/c.php?g=56954&p=367143

Appendices

DIFFERENTIAL EXPRESSION/CLUSTERING CODE
(Clustering is performed in DESeq2 differential expression functions)
(Normalization is included in DESeq2 code)

Appendix 1. R Prep Steps: Prep_Steps_Final.R

```
#This script takes performs the necessary preparatory steps to run the DESeq2
#R scripts for this project: installing packages, reading in files, and setting up factors.

#Intalling necessary Packages

#bioconductor: for necessary bioconductor packages (DESeq2, GOstats)
source("https://bioconductor.org/biocLite.R")
biocLite()

#DESeq2: for differential expression, clustering, visualization
biocLite("DESeq2")
library(DESeq2)

#GOstats, GSEABase, AnnotationForge: for GO-enrichment analyses
biocLite("GOstats"):
  library(GOstats)
library(cluster)

biocLite("GSEABase")
library(GSEABase)

biocLite("AnnotationForge")
library(AnnotationForge)

#Pheatmap: For heatmap visualizations
install.packages("pheatmap")
library(pheatmap)
library("RColorBrewer")

#xlsx: to print files to Excel
install.packages("xlsx")
library(xlsx)

################

#Load data
#load full Date palm count matrix from 5-12-16 pipeline run
```

```
countData<-as.matrix(read.csv("count.table_full-pipeline-5-12-16.txt",sep="\t",row.names=1))

#Split column data
#get sample names (which contain exp design info: sample variety and time)
expdesign=colnames(countData)
expdesign[55] #-->this is where Nebeit_Seif starts
#get two parts: second part is for Nebeit_Seif, which contains internal "_" in name
expdesign_pt1=colnames(countData[,1:54])
expdesign_pt2=colnames(countData[,55:ncol(countData)])

#split to get levesl and factors

#for part 1
#examine result on line one
strsplit(expdesign[1],"_")

unlist(strsplit(expdesign_pt1,"_"))->dplevels_pt1
matrix(dplevels_pt1,ncol=5,byrow=T)->dpfactors_pt1

#for part 2
#examine result on line one
strsplit(expdesign[55],"_")

unlist(strsplit(expdesign_pt2,"_"))->dplevels_pt2
matrix(dplevels_pt2,ncol=6,byrow=T)->dpfactors_pt2
#paste Nebeit Seif name together
dpfactors_pt2_paste<-(cbind(dpfactors_pt2[,1:2],
                    paste0(dpfactors_pt2[,3],dpfactors_pt2[,4]),
                    dpfactors_pt2[,5:6]))

#pate two parts of factors together
dpfactors_all<-rbind(dpfactors_pt1,dpfactors_pt2_paste)

#set as factors
var_fact_all<-as.factor(dpfactors_all[,3])
time_fact_all<-as.factor(dpfactors_all[,4])

#set factors separated
var_fact_pt1<-as.factor(dpfactors_pt1[,3])
time_fact_pt1<-as.factor(dpfactors_pt1[,4])
var_fact_pt2<-as.factor(dpfactors_pt1[,3])
time_fact_pt2<-as.factor(dpfactors_pt1[,4])

#get shortened colnames (remove "Sample_sample") for countData,
#assign to new countData_shortCols object
```

```
shortCols<-paste0(dpfactors_all[,3],"_",dpfactors_all[,4],"_",dpfactors_all[,5])
countData_shortCols<-countData
colnames(countData_shortCols)<-shortCols

###END SCRIPT#####
```

Appendix 2. R Wald DESeq2 Function: "DESeq_OneVar_Function_Final.R"

```
#This script defines a function to run DESeq2 differential expression analysis
#on a variety in terms of time using the Wald hypothesis test.
#It then conducts clustering of the variety by time and creates
#clustering dendrogram, silhouette plots, heat maps for genes,
#and heat maps for sample-to-sample distances.
#It can be run as a mutliple-factor experiment on both variety and time (MF=True).

#define DESeq One Sample Function
deseq_one<-function(countData,r1,r2,var_name,variety_v,time_v,MF=FALSE) {

 #get sample count data subset for variety from args
 countData_Var<-countData[,r1:r2]

 #get subset of factors for variety
 variety_v=variety_v[r1:r2]
 time_v=time_v[r1:r2]

 #define column data and set row names
 colData_Var<-data.frame(variety_v,time_v)
 rownames(colData_Var)<-colnames(countData_Var)

 #test if col data rownames match count data col names
 if(!all(rownames(colData_Var) == colnames(countData_Var))) {
  #if not, exit function with error
  return("Error: Col data row names do not match count matrix column names")
 }


 ##################
 #DESeq data set object

 #Construct DESeqDataSet for Variety sample in terms of time
 dds_Var <- DESeqDataSetFromMatrix(countData = countData_Var,
                                   colData = colData_Var,
                                   design = ~time_v)

 dds_Var
```

```
#set default text size for single-factor experiment
text_size=1

#Run multi-factor design if MF true (for all-variety run)
if(MF) {
  design(dds_Var)<-formula(~variety_v + time_v)
  text_size=0.5 #decrease text size for multifactor dendrogram
}

#pre-filtering low-count reads
dds_Var <- dds_Var[ rowSums(counts(dds_Var)) > 1, ]

#set reference levels
dds_Var$time_v<-factor(dds_Var$time_v,levels=c("45","75","105","120","135"))

###############
#DIFFERENTIAL EXPRESSION

#Run differential expression test for variety
dds_Var <- DESeq(dds_Var)

#get results of DE test for variety
res_Var <- results(dds_Var, alpha=0.05)

#Preview summary of results
summary(res_Var)

#get # of adjusted p-values less than 0.05
sum(res_Var$padj < 0.05, na.rm=TRUE)
#print number to text file
pval_df<-data.frame()
pval_df[1,1]<-("Variety:")
pval_df[1,2]<-(var_name)
pval_df[2,1]<-("No. genes with DE padj<0.05:")
pval_df[2,2]<-(sum(res_Var$padj < 0.05, na.rm=TRUE))

write.table(pval_df,file=paste0("padj#_",var_name,".txt"),col.names=FALSE,row.names=FALSE,
quote=FALSE)

###############
#Rlog NORMALIZATION for clustering

#Get rlog transformation of dds object (rld)
rld_Var <- rlog(dds_Var)
```

```
 #get transformed count values
 rld_Var.counts<-assay(rld_Var)

 ###############

 #CLUSTERING

 #get distances and plot hcluster using rld
 dists_Var <-dist(t(rld_Var.counts))
 hclusts_Var <-hclust(dists_Var)
 pdf(paste0("hclust_",var_name,".pdf"))
 plot(hclusts_Var,cex=text_size,main=paste0(var_name," Cluster
Dendrogram"),xlab=paste0("dists_",var_name))
 dev.off()

 ############################
 #Check silhouette

 #split in # groups on basis of dendrogram --> adjust number for variety as needed
 groups_Var<-cutree(hclusts_Var, k=4)

 #get names of groups --> adjust number for specific variety as needed
 names(which(groups_Var==1))->group1_Var
 names(which(groups_Var==2))->group2_Var
 names(which(groups_Var==3))->group3_Var
 #names(which(groups_Var==4))->group4_Var
 #names(which(groups_Var==5))->group5_Var

 #print group names --> adjust number for specific variety as needed

write.table(as.data.frame(group1_Var),file=paste0("group1names_",var_name,".txt"),col.name
s=FALSE,row.names=FALSE,quote=FALSE)

write.table(as.data.frame(group2_Var),file=paste0("group2names_",var_name,".txt"),col.name
s=FALSE,row.names=FALSE,quote=FALSE)

write.table(as.data.frame(group3_Var),file=paste0("group3names_",var_name,".txt"),col.name
s=FALSE,row.names=FALSE,quote=FALSE)

write.table(as.data.frame(group4_Var),file=paste0("group4names_",var_name,".txt"),col.name
s=FALSE,row.names=FALSE,quote=FALSE)

#write.table(as.data.frame(group5_Var),file=paste0("group5names_",var_name,".txt"),col.nam
es=FALSE,row.names=FALSE,quote=FALSE)
```

```
#get silouhette of k grouping
silhouette(groups_Var,dist=dists_Var)->Var_sil

#print silhouette to PDF
pdf(paste0("sil_",var_name,".pdf"))
plot(Var_sil, main=paste0(var_name, " Silhouette"))
dev.off()

##########################

#Other visualizations of the data

#HeatMaps

#Get heatmap of count matrix of differentially expressed genes
select<-order(rowMeans(counts(dds_Var,normalized=TRUE)),
          decreasing=TRUE)[1:20]

#get col data as dataframe
df<-as.data.frame(colData(dds_Var)[,c("variety_v","time_v")])

#heatmap using rld transform
pdf(paste0("gene_heatmap_rld_",var_name,".pdf"))
pheatmap(assay(rld_Var)[select,],cluster_rows=FALSE,show_rownames=FALSE,
      cluster_cols=FALSE,annotation_col=df,main=paste0(var_name," Gene Heatmap (RLD)"))
dev.off()

#heatmap using varianceStabilizingTransformation
vsd_Var <- varianceStabilizingTransformation(dds_Var, blind=FALSE)
pdf(paste0("gene_heatmap_vsd_",var_name,".pdf"))
pheatmap(assay(vsd_Var)[select,],cluster_rows=FALSE,show_rownames=FALSE,
      cluster_cols=FALSE,annotation_col=df, main=paste0(var_name,"Gene Heatmap (VSD)"))
dev.off()
###

#Heatmap of sample-to-sample distance

#get distances for Variety samples
sampleDists_Var <- dist(t(assay(rld_Var)))

#make map
sampleDistMatrix_Var <- as.matrix(sampleDists_Var)
rownames(sampleDistMatrix_Var)<-paste(rld_Var$variety_v,rld_Var$time_v,sep="-")
colnames(sampleDistMatrix_Var)<-NULL
colors<-colorRampPalette(rev(brewer.pal(9,"Blues")))(255)
```

```
 #print map to PDF
 pdf(paste0("sampletimes_heatmap_",var_name,".pdf"))
 pheatmap(sampleDistMatrix_Var,
        clustering_distance_rows=sampleDists_Var,
        clustering_distance_cols=sampleDists_Var,
        col=colors,main=paste0(var_name," Sample Heatmap"))
 dev.off()

  ##GETTING GENE LIST####

 #GET DIFFERENTIALLY EXPRESSED GENES

 #remove nas
 res_Var.nona <- res_Var[complete.cases(res_Var),]
 #get variety genes with padj < 0.05
 gene.list.Var<-row.names(res_Var.nona[res_Var.nona$padj<0.05,])

 #printing out
 head(as.data.frame(gene.list.Var))

write.table(as.data.frame(gene.list.Var),file=paste0("gene.list_",var_name,".txt"),col.names=FA
LSE,row.names=FALSE,quote=FALSE)

 ############

 #Get expression matrix for DE genes (gene.list.**)
 expmatrix.gene.list.Var<-countData_Var[gene.list.Var,]

} #END OF FUNCTION###

#function calls
#on ChiChi
deseq_one(countData_shortCols,1,13,"Chi",var_fact_all,time_fact_all)
#on Kenezi (14:28)
deseq_one(countData_shortCols,14,28,"Kenezi",var_fact_all,time_fact_all)
#on Khalas
deseq_one(countData_shortCols,29,44,"Khalas",var_fact_all,time_fact_all)
#on Lulu
deseq_one(countData_shortCols,45,54,"Lulu",var_fact_all,time_fact_all)
#on NebeitSeif
deseq_one(countData_shortCols,55,64,"NebeitSeif",var_fact_all,time_fact_all)

#function call on all samples, times
```

```
deseq_one(countData_shortCols,1,ncol(countData_shortCols),"All_Vars",var_fact_all,time_fact
_all,MF=TRUE)

###END SCRIPT#####
```

Appendix 3. R LRT DESeq2 Function: "DESeq_OneVar_LRT_Function_Final.R"

```
#This script defines a function to run DESeq2 differential expression analysis
#on a variety in terms of time using the LRT hypothesis test (for time series).
#It then conducts clustering of the variety by time and creates
#clustering dendrograms as well and heat maps for sample-to-sample distances.
#It also conducts clustering by gene, creating gene cluster dendrogram,
#silhouette plots for gene cluster groups, box plots, and gene expression
#heat maps.
#It can be run as a mutliple-factor experiment on both variety and time (MF=True).

#define DESeq One Sample LRT Function
deseq_one_lrt<-function(countData,r1,r2,var_name,variety_v,time_v,MF=FALSE) {

 var = var_name

 #get sample count data subset for variety
 countData_Var<-countData[,r1:r2]

 #get subset of factors for variety
 variety_v=variety_v[r1:r2]
 time_v=time_v[r1:r2]

 #define column data and set row names
 colData_Var<-data.frame(variety_v,time_v)
 rownames(colData_Var)<-colnames(countData_Var)

 #test if col data rownames match count data col names
 if(!all(rownames(colData_Var) == colnames(countData_Var))) {
  #if not, exit function with error
  return("Error: Col data row names do not match count matrix column names")
 }


 ##################
 #DESeq data set object

 #Construct DESeqDataSet for Variety sample in terms of time
 dds_Var <- DESeqDataSetFromMatrix(countData = countData_Var,
                                   colData = colData_Var,
```

```
                        design = ~time_v)

dds_Var

#set text size to default for non-MF test
text_size=1

#Run multi-factor design if MF true (for all-variety run)
if(MF) {
  design(dds_Var)<-formula(~variety_v + time_v)
  text_size=0.5 #set smaller text size for MF dendrogram
}

#pre-filtering low-count reads
dds_Var <- dds_Var[ rowSums(counts(dds_Var)) > 1, ]

#set reference levels
dds_Var$time_v<-factor(dds_Var$time_v,levels=c("45","75","105","120","135"))


###############
#DIFFERENTIAL EXPRESSION

#TIME SERIES experimental design: LRT

#Differential expression test

#LRT option for multifactor
if(MF) {
  dds_Var <- DESeq(dds_Var,test="LRT",reduced=~variety_v)
} else {
  dds_Var <- DESeq(dds_Var,test="LRT",reduced=~1)
}

#get results of DE test for variety
res_Var <- results(dds_Var, alpha=0.05)

#get # of adjusted p-values less than 0.05
sum(res_Var$padj < 0.05, na.rm=TRUE)
#print number to text file
pval_df<-data.frame()
pval_df[1,1]<-("Variety:")
pval_df[1,2]<-(var_name)
pval_df[2,1]<-("No. genes with DE padj<0.05:")
pval_df[2,2]<-(sum(res_Var$padj < 0.05, na.rm=TRUE))
```

```
write.table(pval_df,file=paste0("padj#_",var_name,"_LRT.txt"),col.names=FALSE,row.names=FA
LSE,quote=FALSE)

 ###############
 #NORMALIZATION for clustering

 #Get rlog transformation of dds object (rld)
 rld_Var <- rlog(dds_Var)

 #get transformed count values
 rld_Var.counts<-assay(rld_Var)

 ###############

 #CLUSTERING

 #clustering of SAMPLES
 #get distances and plot hcluster using rld
 dists_Var <-dist(t(rld_Var.counts))
 pdf(paste0("hclust_",var_name,"_LRT.pdf"))
 plot(hclust(dists_Var),cex=text_size)
 dev.off()

 #clustering of GENES
 #get distances for genes, and plot hcluster
 dists_genes_Var <-dist(rld_Var.counts)
 hclusts_genes_Var<-hclust(dists_genes_Var)
 pdf(paste0("hclust_genes_",var_name,"_LRT.pdf"))
 plot(hclusts_genes_Var,main=paste0(var_name," Gene Cluster"))
 dev.off()

 #get silhouette for gene clusters

 #split gene cluster in # groups on basis of dendrogram; set number for variety
 groups_genes_Var<-cutree(hclusts_genes_Var, k=2)

 #get names of groups, set number for variety
 names(which(groups_genes_Var==1))->group1_genes_Var
 names(which(groups_genes_Var==2))->group2_genes_Var
 #names(which(groups_genes_Var==3))->group3_genes_Var
 #names(which(groups_genes_Var==4))->group4_genes_Var
 #names(which(groups_genes_Var==5))->group5_genes_Var
 #names(which(groups_genes_Var==6))->group6_genes_Var
 #names(which(groups_genes_Var==7))->group7_genes_Var
```

```
 #print group names; set number for variety

write.table(as.data.frame(group1_genes_Var),file=paste0("group1names_genes",var_name,".tx
t"),col.names=FALSE,row.names=FALSE,quote=FALSE)

write.table(as.data.frame(group2_genes_Var),file=paste0("group2names_genes",var_name,".tx
t"),col.names=FALSE,row.names=FALSE,quote=FALSE)

#write.table(as.data.frame(group3_genes_Var),file=paste0("group3names_genes",var_name,".
txt"),col.names=FALSE,row.names=FALSE,quote=FALSE)

#write.table(as.data.frame(group4_genes_Var),file=paste0("group4names_genes",var_name,".
txt"),col.names=FALSE,row.names=FALSE,quote=FALSE)

#write.table(as.data.frame(group5_genes_Var),file=paste0("group5names_genes",var_name,".
txt"),col.names=FALSE,row.names=FALSE,quote=FALSE)

#write.table(as.data.frame(group6_genes_Var),file=paste0("group6names_genes",var_name,".
txt"),col.names=FALSE,row.names=FALSE,quote=FALSE)

#write.table(as.data.frame(group7_genes_Var),file=paste0("group7names_genes",var_name,".
txt"),col.names=FALSE,row.names=FALSE,quote=FALSE)

 #get silouhette
 silhouette(groups_genes_Var,dist=dists_genes_Var)->Var_genes_sil

 #print silhouette to PDF
 pdf(paste0("sil_genes_",var_name,".pdf"))
 plot(Var_genes_sil,main=paste0(var_name," Gene Silhouette"))
 dev.off()

 #return()

 ######BELOW: BOXPLOT FOR VARIETY GENE GRPS###

 #plotting AVERAGES: get average gene vals across time factor for gene groups, plot log2
 #using count table
 #set number according to variety

 #grp1
 avgs_grp1_genes_Var<-apply(countData_Var[group1_genes_Var,],1,tapply,time_v,mean)
 pdf(paste0("box_grp1genes_",var_name,"_LRT.pdf"))
 boxplot(t(log2(avgs_grp1_genes_Var)),main=paste0(var_name,": Group 1"),
ylab="log2(mean(expression))",xlab="Time Factor") #THIS WORKS
```

```
 dev.off()

 #grp 2
 avgs_grp2_genes_Var<-apply(countData_Var[group2_genes_Var,],1,tapply,time_v,mean)
 pdf(paste0("box_grp2genes_",var_name,"_LRT.pdf"))
 boxplot(t(log2(avgs_grp2_genes_Var)),main=paste0(var_name,": Group 2"),
ylab="log2(mean(expression))",xlab="Time") #THIS WORKS
 dev.off()

 #grp 3
 #avgs_grp3_genes_Var<-apply(countData_Var[group3_genes_Var,],1,tapply,time_v,mean)
 #pdf(paste0("box_grp3genes_",var_name,"_LRT.pdf"))
 #boxplot(t(log2(avgs_grp3_genes_Var)),main=paste0(var_name,": Group 3"),
ylab="log2(mean(expression))",xlab="Time") #THIS WORKS
 #dev.off()

 #grp 4
 #avgs_grp4_genes_Var<-apply(countData_Var[group4_genes_Var,],1,tapply,time_v,mean)
 #pdf(paste0("box_grp4genes_",var_name,"_LRT.pdf"))
 #boxplot(t(log2(avgs_grp4_genes_Var)),main=paste0(var_name,": Group 4"),
ylab="log2(mean(expression))",xlab="Time") #THIS WORKS
 #dev.off()


 #######END CHI GENE CLUST BOX PLOT####


 ############################

 #Other visualizations of the data

 #########
 #HeatMaps

 #load package
 library("pheatmap")

 #Get heatmap of count matrix of differentially expressed genes
 select<-order(rowMeans(counts(dds_Var,normalized=TRUE)),
          decreasing=TRUE)[1:20]

 #get col data as dataframe
 df<-as.data.frame(colData(dds_Var)[,c("variety_v","time_v")])

 #heatmap using rld transform
```

```
pdf(paste0("gene_heatmap_rld_",var_name,"_LRT.pdf"))
pheatmap(assay(rld_Var)[select,],cluster_rows=FALSE,show_rownames=FALSE,
      cluster_cols=FALSE,annotation_col=df)
dev.off()

#heatmap using varianceStabilizingTransformation
vsd_Var <- varianceStabilizingTransformation(dds_Var, blind=FALSE)
pdf(paste0("gene_heatmap_vsd_",var_name,"_LRT.pdf"))
pheatmap(assay(vsd_Var)[select,],cluster_rows=FALSE,show_rownames=FALSE,
      cluster_cols=FALSE,annotation_col=df)
dev.off()
###

#Heatmap of sample-to-sample distance

#get distances for Chi samples
sampleDists_Var <- dist(t(assay(rld_Var)))

#make map
sampleDistMatrix_Var <- as.matrix(sampleDists_Var)
rownames(sampleDistMatrix_Var)<-paste(rld_Var$variety_v,rld_Var$time_v,sep="-")
colnames(sampleDistMatrix_Var)<-NULL
colors<-colorRampPalette(rev(brewer.pal(9,"Blues")))(255)

#print map to PDF
pdf(paste0("sampletimes_heatmap_",var_name,"_LRT.pdf"))
pheatmap(sampleDistMatrix_Var,
      clustering_distance_rows=sampleDists_Var,
      clustering_distance_cols=sampleDists_Var,
      col=colors)
dev.off()

################

#GET DIFFERENTIALLY EXPRESSED GENES
#get differentially expressed genes
#Variety genes with padj < 0.05

#remove nas
res_Var.nona <- res_Var[complete.cases(res_Var),]
gene.list.Var<-row.names(res_Var.nona[res_Var.nona$padj<0.05,])

#printing out
head(as.data.frame(gene.list.Var))
```

```
write.table(as.data.frame(gene.list.Var),file=paste0("gene.list_",var_name,"_LRT.txt"),col.names=FALSE,row.names=FALSE,quote=FALSE)

 ###########

 #Get expression matrix for DE genes (gene.list.**)
 expmatrix.gene.list.Var<-countData_Var[gene.list.Var,]


 #############

} #END OF FUNCTION###

#LRT function calls on varieties
#on ChiChi
deseq_one_lrt(countData_shortCols,1,13,"Chi",var_fact_all,time_fact_all)
#on Kenezi (14:28)
deseq_one_lrt(countData_shortCols,14,28,"Kenezi",var_fact_all,time_fact_all)
#on Khalas
deseq_one_lrt(countData_shortCols,29,44,"Khalas",var_fact_all,time_fact_all)
#on Lulu
deseq_one_lrt(countData_shortCols,45,54,"Lulu",var_fact_all,time_fact_all)
#on NebeitSeif
deseq_one_lrt(countData_shortCols,55,64,"NebeitSeif",var_fact_all,time_fact_all)


#function call on all samples, times
deseq_one_lrt(countData_shortCols,1,ncol(countData_shortCols),"All_Vars",var_fact_all,time_fact_all,MF=TRUE)

###END SCRIPT#####
```

GO TERM ENRICHMENT SCRIPTS

Appendix 4. BASH Gene-Protein Hasher: "gene_pt_hasher.pbs"

```
#!/bin/bash

#PBS -l nodes=1:ppn=8,walltime=12:00:00,mem=6gb
#PBS -N Gene_Pt_Hasher
#PBS -M mid224@nyu.edu
#PBS -m a
#PBS -j eo
```

```
#this script takes differentially expresseed gene list and creates
#gene-to-protein list from date palm gff

cd ${PBS_O_WORKDIR}

#loop through gene list to get rnas that are children to gene id
while read -r line
do
	gene="$line"
	echo "gene = $gene"

	    #get rnas that list gene id as parent, write to temp file
	rna="$(grep -i "Parent=${gene};" DPRefs/ref_DPV01_scaffolds.gff3 | grep -o '\brna\w*')"
	echo "$rna" > rna_temp.txt

	    #read through temp file and add each listed rna, with parent gene id to rna list
	while read -r rline
	do
	      echo -e "${rline}\t${gene}" >> rna_list_tab.txt
	done < "rna_temp.txt"

done < "GeneLists/full_dp_genelist.txt"

#now loop through gene/rna list to get protein ids, print gene-pt hash
while IFS=$'\t' read rna gene;do

	echo "rna = $rna gene = $gene"

	#get protein id

	#below command takes all protein id matches, places in temp file
	ptid="$(grep -i "Parent=${rna};" DPRefs/ref_DPV01_scaffolds.gff3 | grep 'ID=cds' | grep -o
'\protein_id=.*$')"
	echo "ptid = $ptid"

	#place ptid in temp file
	echo "$ptid" > ptid_temp.txt

	#read through ptid_temp.txt and place each protein id, with associated gene id, in file
	while read -r pt_exp
	do
	      #trim 'protein_id=' from pt_exp
	      pt_exp="${pt_exp#'protein_id='}"
	      pt_exp=${pt_exp%;*} #remove any trailing ';' and associated info
	      #print to tab-delimted file
```

```
        #echo -e "${rna}\t${gene}\t${pt_exp}" >> ptid_list_tab.txt #rnas not needed in print
        echo -e "${gene}\t${pt_exp}" >> ptid_list_tab.txt
    done < "ptid_temp.txt"

done < "rna_list_tab.txt"

#remove duplicates
sort ptid_list_tab.txt | uniq > gene_pt_hash_notail.txt
```

Appendix 5. BASH FASTA Protein Splitter: fasta100_full.pbs

```
#!/bin/bash

#PBS -l nodes=1:ppn=12,walltime=12:00:00,mem=8gb
#PBS -N Fasta100_Full
#PBS -M mid224@nyu.edu
#PBS -m a
#PBS -j eo

#this script steps through fasta ids of datepalm protein fasta
#to make 100 fasta-sequence files
#It then calls PBS array to run interproscan on each fasta sequence file

cd ${PBS_O_WORKDIR}

#get all fasta ids
#search for id symbol (>) and print out first word of line into file
grep -i ">" protein.fa  | awk '{print $1}' > fastaids_all.txt

#get number of ids in list
length="$(wc -l < fastaids_all.txt)"

#set to split into n groups
groups=100
#create n groups by setting steps to: # fasta ids/n
step=$((${length}/${groups}))

#create directory for split FASTA files
mkdir "files_div_${groups}"
#create directory for interproscan output
mkdir "files_interpro_div_${groups}"

#create fasta id list that goes by steps
g=0 #set counter for number of groups
#loop through fasta ids by step, print to file
```

```
for ((i=1;i<=$length;((i=i+${step})))); do
	echo "i = $i"
	sed -n ${i}p fastaids_all.txt >> fastaids_all_steps.txt
	((g++))
	if [ $g -eq $groups ] #quit when reach n groups
	then
		break
	fi
done

#get group sequences by stepped fasta ids

#initialize ID1 to first line of step fasta
ID1="$(sed -n 1p fastaids_all_steps.txt)"

#remove first line of step fasta and add eof line
sed '1d' fastaids_all_steps.txt > fastaids_all_stepsmod.txt
#add eof line to end of step fasta mod
echo "ENDOFFILE" >> fastaids_all_stepsmod.txt

#set counter for number of files
n=0
#read through protein fasta by step ids, write sequences to n files
while read -r line
do
	((n++)) #count group number
	ID2="$line"

	#read seq from ID1 to ID2 into nth fa file
	sed -n "/$ID1/,/$ID2/p" protein_dp_full.fa >> "files_div_${groups}/protein_pt${n}.fa"
	#remove trailing ID2 line (no sequences follow it)
	sed -i '$d' "files_div_${groups}/protein_pt${n}.fa"

	#set ID1 to ID2 for next iteration
	ID1="$ID2"

done < "fastaids_all_stepsmod.txt"


#interpro pbs array call on split fastas
qsub -t 1-${groups} interpro_arr.pbs
```

Appendix 6. BASH InterProScan Array Script: interpro_arr.pbs

```
#!/bin/bash
```

```
#PBS -l nodes=1:ppn=12,walltime=24:00:00,mem=12gb
#PBS -N Interpro_Arr
#PBS -M mid224@nyu.edu
#PBS -m a
#PBS -j eo

# This script calls PBS array job to run interproscan on n groups

cd ${PBS_O_WORKDIR}

#set number of fasta files
group=100

#interproscan calls
#load module
module load interproscan-5.4-47.0

#call interpro job for nth fasta file in nth instance of array job
#include go terms in interpro analysis
#save files to appropriate directory
/share/apps/interproscan/interproscan-5.4-47.0/interproscan.sh -i
files_div_${group}/protein_pt${PBS_ARRAYID}.fa -goterms -d files_interpro_div_${group} -f tsv
```

Appendix 7. BASH GOFrameData Object Writer: GO_Pt_writer_multi.pbs

```
#!/bin/bash

#PBS -l nodes=1:ppn=8,walltime=12:00:00,mem=6gb
#PBS -N GO_Pt_Writer
#PBS -M mid224@nyu.edu
#PBS -m a
#PBS -j eo

#This script reads through interproscan output files in a directory,
#and produces final csv file of GO term, ND, and gene name

cd ${PBS_O_WORKDIR}

#read through all interproscan .tsv files in directory
#create list of protein ids and GO terms (piped) as ptid_GO.txt

for f in protein_pt*
do
    echo "Processing $f"
```

```
    #get protein id and GO term columns and append to ptid_GO.txt list
    #remove lines if no GO terms
    cut -f 1,14 ${f} | grep '[^[:blank:]]' | awk 'NF >1' >> ptid_GO.txt

done

#expand piped GO term lines (GOterm|GOterm), save as ptid_GO_expand.txt
while IFS=$'\t' read ptid GO; do

    IFS='|' read -ra GOArr <<< "$GO"
    for i in "${GOArr[@]}"; do
        #echo "$i"
        echo -e "${ptid}\t${i}" >> ptid_GO_expand.txt
    done
done < "ptid_GO.txt"

#remove duplicates
sort ptid_GO_expand.txt | uniq > ptid_GO_expand_d.txt

#Now, match pt IDs to genes, write GOgenedata file(s) (for R reading)
#Remove duplicate lines from that file

while IFS=$'\t' read ptid GO; do

      #get matching gene ids, print to temp file
    gene="$(grep -i "$ptid" gene_pt_hash_notail.txt | cut -f 1)"
    echo "$gene" > gene_temp.txt

    #loop for proteins that match to multiple genes, print to GO data file
    while read -r gene_exp
    do
        echo -e "${GO}\tND\t${gene_exp}" >> GOgenedata.txt
    done < "gene_temp.txt"

done < "ptid_GO_expand.txt"

#remove duplicates
sort GOgenedata.txt | uniq | grep '[^[:blank:]]' | awk 'NF >2'> GOgenedata_d.txt
```

Appendix 8. BASH GO-Gene Matcher: genelist_GOcomp.sh

```
#!/bin/bash

#This script finds those gene ids in a list
#that have GO terms, saves those to new file
```

```
#file for-loop
#read through GOdata object, check if gene name there, print to new file if so
for f in gene.list_*
do
     fbase=${f%.*} #remove '.txt' to get file base name
          echo "File = $f and file_GO = ${fbase}_GO.txt"

          while IFS=$'\t' read gene; do

      #check if gene (only whole word -w) is present in GOdata object
      #if so, print to GOgene list
      if grep --quiet -w "$gene" GOgenedata_d_3rdrun.txt; then
            echo $gene >> "${fbase}_GO.txt"
      fi

          done < "$f"

done
```

Appendix 9. R GO Term Enrichment Script: "GO_Function_Final.R"

```
#This script defines a function to run the GOstats hypergeometric test
#for GO-term enrichment. It also reads in the necessary goFrameData, gene list, and
#universe files. It runs a loop that calls the GOstats function on each variety and
#extracts top terms from those varieties. The script then uses table and sort to
#calculate the most common terms among all varieties' concatenated top-10 and top 50-lists.


#####GO_on FUNCTION STARTS#########
#define function to run GO-term analysis on variant gene lists
GO_on<-function(gsc,universe,variant_g,LRT=FALSE) {
 #function takes in gsc object, universe and datepalm variant name

 #read in variant's genes with GO terms
 #gene GO lists of the form: gene.list_NebeitSeif_GO.txt

 #if check for LRT=TRUE; change base name if reading in LRT gene lists
 if (LRT) {
  file_base=paste0("gene.list_",variant_g,"_LRT")
 } else {
  file_base=paste0("gene.list_",variant_g)
 }

 #read in gene list
```

```
 genes = read.csv(paste0(file_base,"_GO.txt"),sep="\t",header=FALSE)
 genes<-levels(genes$V1) #->get gene names from levels of object factor

 #set paramters for hyperG test
 params <- GSEAGOHyperGParams(name="My Custom GSEA Var Params",
                      geneSetCollection = gsc,
                      geneIds = genes,
                      universeGeneIds = universe,
                      ontology = "BP",
                      pvalueCutoff = 0.05,
                      conditional = FALSE,
                      testDirection = "over")

 #run the hyper g test
 Over <- hyperGTest(params)

 #print top ten terms for var to Excel file

write.xlsx(head(summary(Over),n=10L),file=paste0("GO10_",file_base,".xlsx"),col.names=TRUE,
row.names=TRUE)

 #return summary object with data
 return(summary(Over))

} #END GO_on FUNCTION########

##########

#GET OVERALL GSC, UNIVERSE OBJECTS
#read in goframeData object for use in all GO_on function runs
#read in GO term list (GO terms, evidence, gene ids) for full genome
goframeData <- read.csv("GOgenedata_d_3rdrun.txt", sep="\t",header=FALSE)
colnames(goframeData)<-c("frame.go_id","frame.Evidence","frame.gene_id")

#prepare GO to gene mappings
goFrame=GOFrame(goframeData,organism="Phoenix dactylifera")
goAllFrame=GOAllFrame(goFrame)

#define gsc object #
gsc <- GeneSetCollection(goAllFrame,setType = GOCollection())

#set universe as all genes with GO terms
universe<-levels(goframeData$frame.gene_id)

#make vector of variety base names
```

```
varieties<-c("Chi","Kenezi","Khalas","Lulu","NebeitSeif")

#######
#FUNCTION CALLS ON NON-lrt##########

#initialize char vector to contain all vars' top 10s, all vars' terms
#!re-initialize before every for-loop run!
all_top10<-vector(mode="character")
all_terms<-vector(mode="character")

#loop through varieties and call GO function on them
for (v in varieties) {
  os<-GO_on(gsc, universe, v)
  assign(paste0("GOsum_",v),os)

  #in this loop also add top 10 terms for each to a vector, and for all terms
  all_top10<-c(all_top10,os[1:10,7])
  all_terms<-c(all_terms,os[,7])
}

all_top10
head(all_terms)

#find most common occurrences in all_top10, write to file
GO10_All10<-head(sort(table(all_top10), decreasing = TRUE),n=10L)
write.xlsx(GO10_All10,file="GO10_All10.xlsx",col.names=TRUE,row.names=TRUE)
#find most common occurrences in all_terms, write to file
GO10_AllTerms<-head(sort(table(all_terms), decreasing = TRUE),n=10L)
write.xlsx(GO10_AllTerms,file="GO10_AllTerms.xlsx",col.names=TRUE,row.names=TRUE)

#####
#function call on All Vars
GOsum_All_Vars<-GO_on(gsc, universe, "All_Vars")


###########

#FUNCTION CALLS ON LRT

#initialize char vector to contain all vars' top 10s, all top 50s, all vars' terms
#!!re-initialize before every for-loop run!
all_top10_LRT<-vector(mode="character")
all_top50_LRT<-vector(mode="character")
all_terms_LRT<-vector(mode="character")
```

```
#loop through varieties and call GO function on them
for (v in varieties) {
 os<-GO_on(gsc, universe, v,TRUE)
 assign(paste0("GOsum_LRT_",v),os)

 #in this loop also add top 10, top 50 terms for each to a vector, and for all terms
 all_top10_LRT<-c(all_top10_LRT,os[1:10,7])
 all_top50_LRT<-c(all_top50_LRT,os[1:50,7])
 all_terms_LRT<-c(all_terms_LRT,os[,7])
}


#find 10 most common occurrences in all_top10_LRT; write to file
GO10_All10_LRT<-head(sort(table(all_top10_LRT), decreasing = TRUE),n=10L)
write.xlsx(GO10_All10_LRT,file="GO10_All10_LRT.xlsx",col.names=TRUE,row.names=TRUE)
#find 10 most common occurrences in all_top50_LRT, write to file
GO10_All50_LRT<-head(sort(table(all_top50_LRT), decreasing = TRUE),n=10L)
write.xlsx(GO10_All50_LRT,file="GO10_All50_LRT.xlsx",col.names=TRUE,row.names=TRUE)
#find 10 most common occurrences in all_terms_LRT, write to file
GO10_AllTerms_LRT<-head(sort(table(all_terms_LRT), decreasing = TRUE),n=10L)
write.xlsx(GO10_AllTerms_LRT,file="GO10_AllTerms_LRT.xlsx",col.names=TRUE,row.names=TR
UE)

#####
#LRT function call on All Vars
GOsum_LRT_All_Vars<-GO_on(gsc, universe, "All_Vars",TRUE)

####END SCRIPT###
```

Appendix 10. R GO Gene Non-Overlap Script: "GO_GeneComps.R"

```
#GO TERM GENE-COMPARISONS
#This script checks what genes differ between varities, and runs GO-term-enrichment
#on the different genes.
#Uses LRT gene lists that have GO terms

#define function to run GO-term analysis on variant gene lists
GO_Comp<-function(gsc,universe,gene_comp_list,name_base) {
 #function takes in gsc object, universe and compared-genes list

 #set paramters for hyperG test
 params <- GSEAGOHyperGParams(name="My Custom GSEA Var Params",
                              geneSetCollection = gsc,
                              geneIds = gene_comp_list,
                              universeGeneIds = universe,
```

```
                    ontology = "BP",
                    pvalueCutoff = 0.05,
                    conditional = FALSE,
                    testDirection = "over")

 #run the hyper g test
 Over <- hyperGTest(params)

 #print top ten terms for var to Excel file

write.xlsx(head(summary(Over),n=10L),file=paste0("GO10_",name_base,".xlsx"),col.names=TR
UE,row.names=TRUE)
 #print just top 10 term column to text file

write.table(head(summary(Over)$GOBPID,n=10L),file=paste0("GO10IDs_",name_base,".txt"),co
l.names=FALSE,row.names=FALSE,quote=FALSE)


 #return summary object with data
 return(summary(Over))

} #END GO_Comp FUNCTION########



#find genes that differ between DE gene lists

#READ IN GENE LISTS
#Chi
genes_Chi = read.csv("gene.list_Chi_LRT_GO.txt",sep="\t",header=FALSE)
genes_Chi <- levels(genes_Chi$V1)
#Khalas
genes_Khalas = read.csv("gene.list_Khalas_LRT_GO.txt",sep="\t",header=FALSE)
genes_Khalas <- levels(genes_Khalas$V1)
#Lulu
genes_Lulu = read.csv("gene.list_Lulu_LRT_GO.txt",sep="\t",header=FALSE)
genes_Lulu <- levels(genes_Lulu$V1)
#Kenezi
genes_Kenezi = read.csv("gene.list_Kenezi_LRT_GO.txt",sep="\t",header=FALSE)
genes_Kenezi <- levels(genes_Kenezi$V1)
#Nebeit Seif
genes_NebeitSeif = read.csv("gene.list_NebeitSeif_LRT_GO.txt",sep="\t",header=FALSE)
genes_NebeitSeif <- levels(genes_NebeitSeif$V1)

#COMPARE GENE LISTS
```

```
#compare ChiChi and Khalas
genes_ChiNotKhalas <- setdiff(genes_Chi,genes_Khalas)
#compare Khalas and ChiChi
genes_KhalasNotChi <- setdiff(genes_Khalas,genes_Chi)

#write gene lists to text file
write.table(genes_ChiNotKhalas,file="genes_ChiNotKhalas.txt",col.names=FALSE,row.names=F
ALSE,quote=FALSE)


#Compare Lulu and Khalas
genes_LuluNotKhalas <- setdiff(genes_Lulu,genes_Khalas)
#Compare Khalas and Lulu
genes_KhalasNotLulu <- setdiff(genes_Khalas,genes_Lulu)

#Compare Kenezi and Khalas
genes_KeneziNotKhalas <- setdiff(genes_Kenezi,genes_Khalas)
#Compare Khalas and Kenezi
genes_KhalasNotKenezi <- setdiff(genes_Khalas,genes_Kenezi)


#FUNCTION CALLS
#Chi vs. Khalas
GO_ChiNotKhalas<-GO_Comp(gsc, universe, genes_ChiNotKhalas,"ChiNotKhalas")
GO_KhalasNotChi<-GO_Comp(gsc, universe, genes_KhalasNotChi,"KhalasNotChi")

#Lulu vs. Khalas
GO_LuluNotKhalas<-GO_Comp(gsc, universe, genes_LuluNotKhalas,"LuluNotKhalas")
GO_KhalasNotLulu<-GO_Comp(gsc, universe, genes_KhalasNotLulu,"KhalasNotLulu")

#Kenezi vs. Khalas
GO_KeneziNotKhalas<-GO_Comp(gsc, universe, genes_KeneziNotKhalas,"KeneziNotKhalas")
GO_KhalasNotKenezi<-GO_Comp(gsc, universe, genes_KhalasNotKenezi,"KhalasNotKenezi")

#############END SCRIPT##########################
```

Appendix 11. BASH GOIDs to Genes: "GOIDs_to_genes.sh"

```
#!/bin/bash

#This script finds the date palm gene IDs that connect to given GO IDs,
#then checks which of those gene ids exist in given VarVsVar gene list,
#and prints the passing genes to file.

#read through GOID terms and find associated gene ids in GOgenedata_d_3rdrun.txt
```

```
while read -r line
do
	BPID="$line"
	echo "$BPID"

	#get gene ids that match to each GOID, print to  file
	gene="$(grep -i "$BPID" GOgenedata_d_3rdrun.txt | cut -f 3)"
	echo "$gene" >> go_gene.txt

done < "GO10IDs_ChiNotKhalas.txt"

#remove blank lines, will be no duplicates
grep '[^[:blank:]]' go_gene.txt > go_gene_b.txt

#####################

#Now check which genes in go_gene_b.txt are present in orignal genes_ChiNotKhalas.txt

while read -r gline
do
	if grep --quiet "$gline" genes_ChiNotKhalas.txt; then
		echo "$gline" >> go_gene_b_cnk.txt
	fi


done < "go_gene_b.txt"

#######END SCRIPT############
```