Stat.585X: Project Report

Lindsay Rutter

April 25, 2014

Project Topic

The project focused on the construction and interactive visualization of phylogenetic representation of soybean varieties.

Project Motivation

The motivation of the project stems from my interest in bioinformatics and computational biology. Ideally, the product could be used by biologists, geneticists, and agronomists interested in studying how soybean varieties are related. By referencing the interactive visualization of the phylogenetic tree, these scientists may better understand genetic testing results - in this particular dataset, in terms of copy number variants, single nucleotide polymorphisms, protein content, and yield - and use that knowledge in future breeding.

Available Data

The available data consisted of a data frame structure that contains 412 direct child-parent relationships between pairs of soybean varieties. These data were collected from frield trials, genetic studies, and United States Department of Agriculture (USDA) bulletins, and date as early as the first decade of the 1900s.

Data Collection and Processing

The data frame format of the soybean varieties has been represented as an interactive phylogenetic tree, in Shiny software produced by Susan VanderPlas:

http://gsoja.agron.iastate.edu:3838/Soybeans/

In the "Genealogy" tab of the Shiny application, the user may choose one or more varieties in the left-panel menu, and immediately view the phylogenetic tree representation of the selected varieties in the right-panel plot as per reactive programming.

However, currently, the right-panel plots are plotted independently for each soybean variety selected, showing a user-selected number of generations surrounding that variety, regardless of its relationship to any other varieties selected.

Instead, it may be useful for biologists to obtain one large plot in the right-panel that merges the selected soybean varieties, thereby determining not only whether or not there exists a relationship between the varieties, but also showing the relationship as a path in the graphical phylogetntic structure, with possibly the varieties (nodes) and the path (edges) between them highlighted for emphasis.

In total, there are about 230 unique soybean varieties present in the tree data frame. The package igraph might be used to determine any pathway relationships between selected varities:

http://cran.r-project.org/web/packages/igraph/index.html

Final Product: Functions Explanation

The final product is a R package titled `phyViz` that includes 23 main functions. The almost 1000 lines of code for all 23 functions is included in the `R` directory of the `phyViz` package. Below is a brief description of the individual functions, which are separately included as `.Rd` files in the `man` file of the `phyViz` package:

Function 1 is simply for data reformatting.

1. `processTreeGraph.Rd`: Converts the example `tree.rda` file into an igraph object.

Functions 2-9 are used to gather immediate information about a particular variety. Some of these functions are called as helper function in other more-complex functions that are described later

2. `isParent.Rd`: Determines if a variety is a parent of another variety.

3. `isChild.Rd`: Determines if a variety is a child of another variety.

4. `getparent.Rd`: Returns the parents of a variety.

5. `getchild.Rd`: Returns the children of a variety.

6. `getancestors.Rd`: Returns a nested list of the ancestors of a variety.

7. `getdescendants.Rd`: Returns a nested list of the descendants of a variety.

8. `getYear.Rd`: Determines the year of a variety.

9. `getDegree.Rd`: Determines the degree between two varieties, where each edge is weighted unity for a child-parent relationship.

Function 10 makes use of commands in the iGraph package to report information about the entire tree (as opposed to individual varieties as in the functions above).

10. `getBasicStatistics.Rd`: Returns the basic statistics of the tree (number of nodes, number of edges, whether or not the whole graph is connected, number of components, average path length, graph diameter, etc.)

Functions 11-13 must all three be used, in the respective order, to construct a simplified image of the path between two varieties `(See Figure 1)`. The x-axis position of each variety label indicates the year of the variety, and the y-axis position of each variety label simply indicates its index in the path.

11. `getPath.Rd`: Determines the shortest path between the two inputted vertices, and takes into account whether or not the graph is directed. If there is a path, the list of vertices of the path will be returned. Note: For a directed graph, the direction matters. However, this function will check both directions and return the path if it.

12. `buildPathDF.Rd`: Builds a dataframe of information about the path object created in the function above that can later be used for visualization. The dataframe includes `label` (name of each variety) of each node, `x` (the year of the variety, the x-axis value for which the label and incoming/outgoing edges are centered), `y` (the y-axis value, which is the index of the path, incremented by unity), `xstart` (the x-axis position of theoutgoing edge (leaving to connect to the node at the next largest y-value)), `xend` (the x-axis position of the outgoing edge (connected to the node at the next largest y-value)), `ystart` (the y-axis position of the outgoing edge (leaving to connect to the node at the next largest y-value), `yend` (the y-axis position of the outgoing edge (connected to the node at the next largest y-value))).

13. `generatePathPlot.Rd`: Generates an ggplot2 object of the path dataframe build in the function above. The image will correctly position the node labels with x-axis representing the node year, and y-axis representing the node path index. Edges between two nodes represent parent-child relationships between those nodes. For visual appeal, there is a grey box that outlines the node label, as well as an underline and overline for each label.

Functions 14-18 must all five be used, in the respective order, to construct an image of the path between two varieties superimposed on the entire tree `(See Figure 2)`. Specifically, functions 14-17 all create data frames that must be fed into function 18 to generate the image. In the final image, The x-axis position of each variety label indicates the year of the variety, and the y-axis position of each variety label simply indicates its index in the path `(See Figure 2)`.

14. `buildSpreadTotalDF.Rd`: Constructs a data frame object that mitigates overlap in text labeling of varieties, while maintaining that the x-axis position will represent years. (This was a difficult process in this tree dataset because hundreds of varieties were all centered in a small window of years).

15. `buildMinusPathDF.Rd`: Constructs a data frame object that uses the information on how to spread the text labelings of variietes (from the function above) to include the label, x-axis position, and y-axis position of all nodes in the tree. However, the labels of the path varietes are not included in this data frame, as they will be created in a separate data frame, so that they can be fed into the image-generation function (Function 18) separately and allow for separate sizes and colors.

16. `buildEdgeTotalDF.Rd`: Constructs a data frame object of all the parent-child relationships in the graph, taking into account the optimized mitigation of text overlap from two functions above.

17. `buildPlotTotalDF.Rd`: Constructs a data frame object of text label positions and edges only for the varieties in the path, taking into the account the optimized mitigation of text overlap from two functions above.

18. `generateTotalPlot.Rd`: Constructs a ggplot2 object using the four data frames constructed in the previous four functions. The image will correctly position the node labels with x-axis representing the node year, and y-axis representing the node path index. Light grey edges between two nodes represent parent-child relationships between those nodes. To enhance the visual understanding of how the path-of-interest fits into the entire graph structure, the nodes within the path are labelled in boldface, and connected with light-green boldfaced edges.

Functions 19-22 must all four be used, in the respective order, to construct an image of the ancestors and descendents of a given variety `(See Figure 3)`. Specifically, functions 19-21 all create data frames that must be fed into function 22 to generate the image.

19. `node.to.data.frame.Rd`: This is a helper function that the function below (plotcoords) will use. It converts all the ancestor and descendent nodes of a variety to a data frame format.

20. `plotcoords.Rd`: Returns a data frame that contains the coordinates to plot all ancestors and descendanst of a variety on a tree. The x and y values describe the coordinates of the label, while the xstart, ystart, xend, and yend values describe the edges of the label.

21. `buildGenDF.Rd`: Returns the data frame that includes labels and plot coordinates of all ancestors and descendants of a variety. Users can specify the maximum number of ancestors and descendants to display.

22. `generateGenPlot.Rd`: Returns the image object to show the ancestors (to the left) and descendants (to the right) of a variety, with the variety highlighted in orange.

Function 23 can be used to construct a matrix that shows the degree between all pairwise combination of all varieties inputted.

23. `buildDegMatrix.Rd`: Returns an image showing the degree between all pairwise combinations of all varietes inputted `(See Figure 4)`.

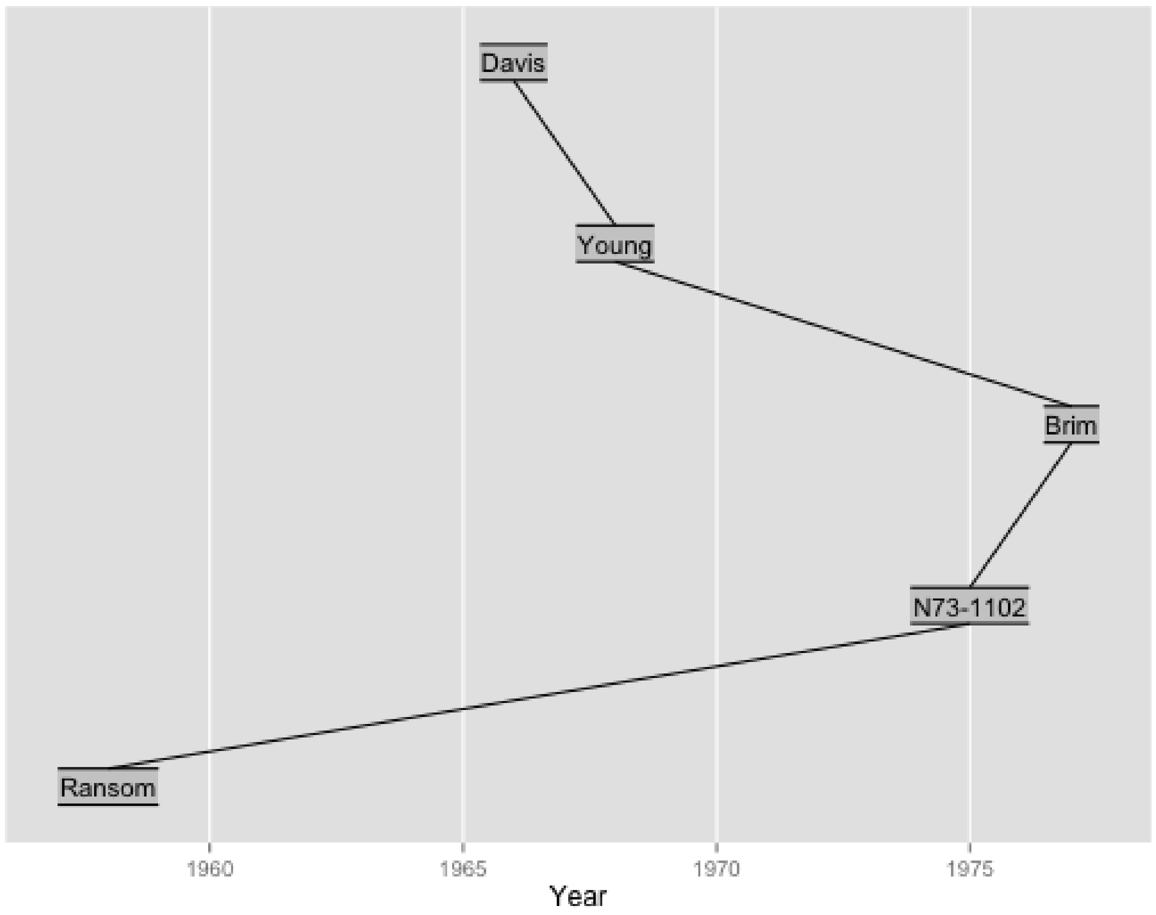

Figure 1: Phylogenetic representation of the path between two example varieties, Davis and Ransom. Here, the x-axis values represent the year of the varieties

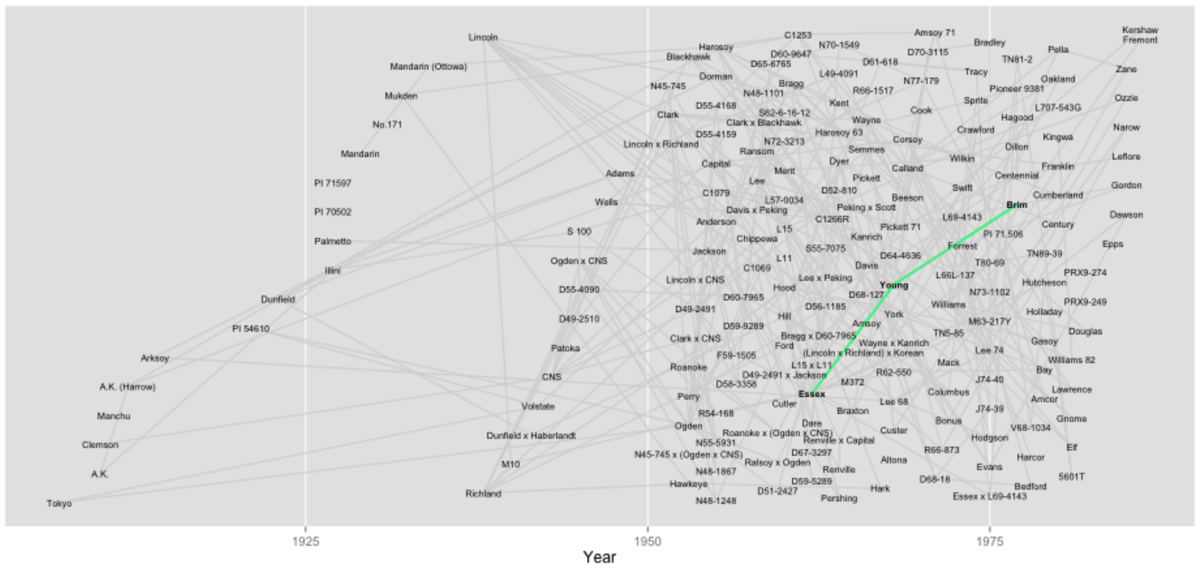

Figure 2: Phylogenetic representation of the path between two example varieties, Essex and Brim, superimposed on the entire phylogenetic tree. For visual enhancement, the path edges are highlighted in green and the path nodes are boldfaced. Here, the x-axis values represent the year of the varieties.

Final Product: R Package Explanation

As stated previously, the R package is called `phyViz.` It is composed of the following files and directories:

- `NAMESPACE`: This file exports all 23 functions
- `DESCRIPTION`: This file describes the package
- `R`: This directory contains one file called `allFunctions.R`
  `allFunctions.R`: This file contains the code for all 23 functions, along with documentation.
- `man`: This directory contains 23 `.Rd` files, one for each function
- `inst`: This directory contains one subdirectory called `doc`
  `doc`: This directory three files for phyViz documention with extensions .R, .html, and .rmd
- `vignette`: This directory contains one file for phyViz vignette .rmd file

Final Product: Package Location

The phyViz package is located on my gitHub webpage at:

https://github.com/lrutter/Stat585Project

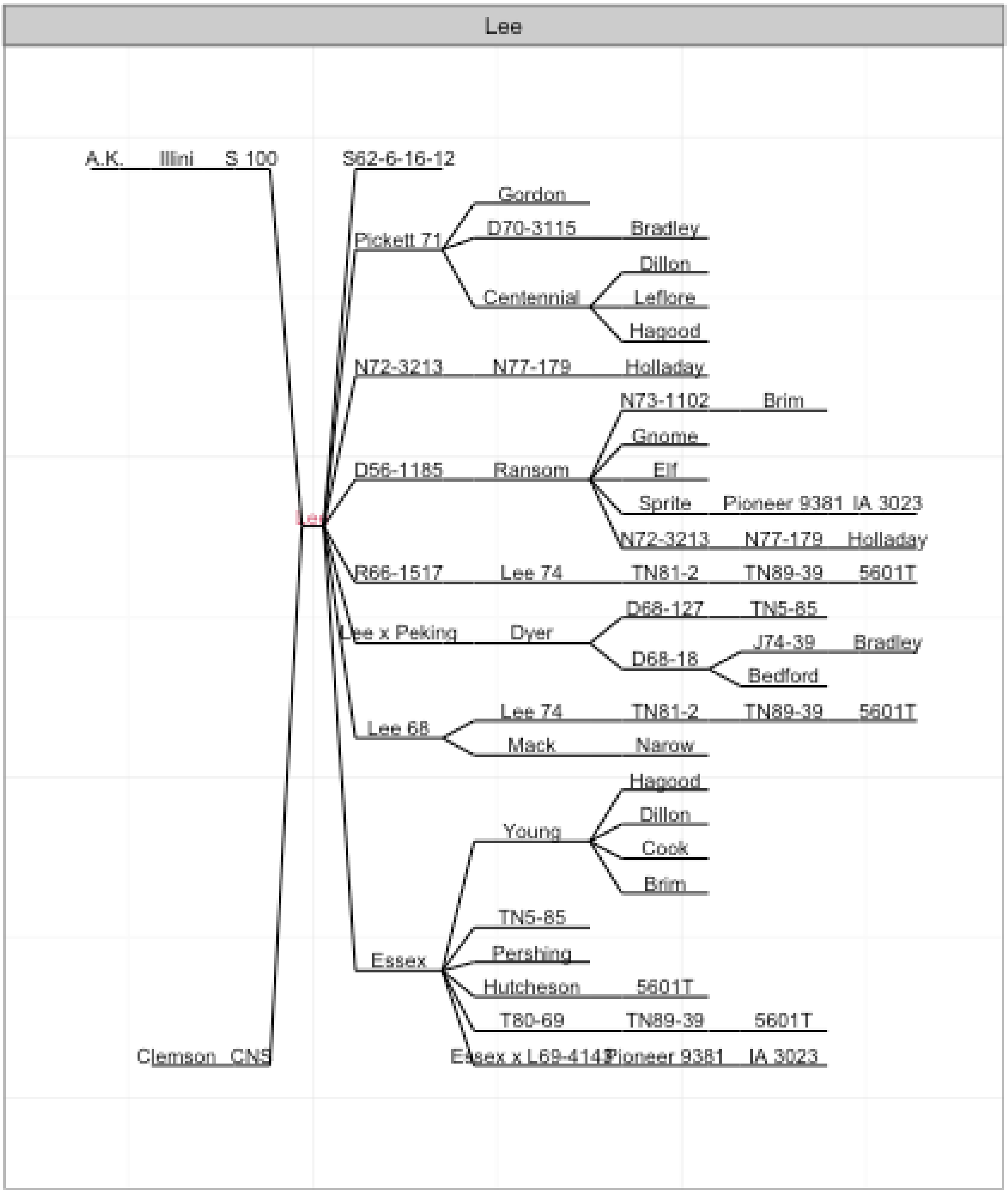

Figure 3: Phylogenetic representation of ancestors and descendants of an example variety called Lee. The x-axis represents the generation number for ancestors (to the left of Lee) and descendants (to the right of Lee), not the years of each variety.

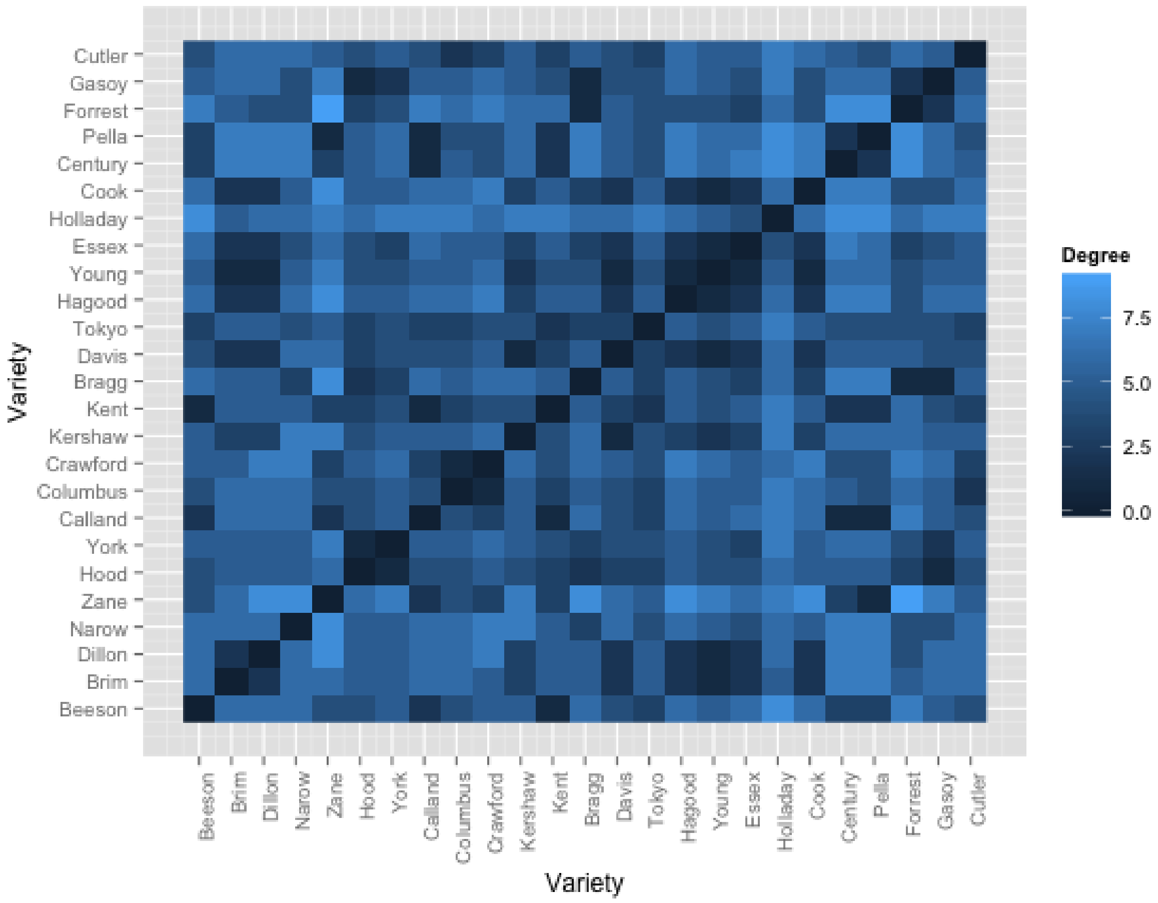

Figure 4: Matrix representation of the degree between all pairwise combinations of an example set of 25 varieties. The degrees are calculated as an unweighted graph, where a parent-child relationship has a value of unity

Final Product: Future Direction

One future avenue would be to incorporate the Shiny application into the R package, if possible. That would allow users to view the four main types of images demonstrated above in a more interactive way, so that reactive programming could save them the time of using command-line for each change of input as well as the inefficiency of rerunning code that does not need to be rerun.

In the future, the package can also be tested on other examples than the `tree.rda` file. This could help fix any remaining bugs, and provide further insight into how to best develop the graphics for different types of trees.