A LIMS-less System for Genotyping Data in Marker-Assisted Selection

Alex Rios
San Jose State University

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_projects

Part of the Bioinformatics Commons

Recommended Citation
DOI: https://doi.org/10.31979/etd.7xme-5utn
https://scholarworks.sjsu.edu/etd_projects/1202

This Master's Project is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Projects by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.
A LIMS-less System for Genotyping Data in Marker-Assisted Selection

A Project

Presented to
Department of Computer Science
San Jose State University

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science

By

Alex Rios
December 2022
ABSTRACT

Climate change and feeding the growing human population are two tightly intertwined problems we face now. Innovations across several agricultural sectors are needed to meet these challenges as we proceed into the future. Seed companies and plant breeders are at the forefront of producing new plant varieties that must survive under the unique stresses of extreme weather patterns caused by changing climates. This project aims to provide breeders with faster marker results by streamlining the data process of marker-assisted selection (MAS). Manually managing genotyping data in MAS can take several hours or days to compile results for breeders. Although some solutions have been deploying laboratory information management systems (LIMS), they are often excessive, expensive, and add more steps to workflows in MAS labs. MASLGenerator and SPLi4MKGTR are two programs created in this project as an alternative to commercial LIMS systems. These tools facilitate and increase the rate of data generation for MAS.

Keywords – LIMS, Marker Assisted Selection, Genotyping
Many thanks and gratitude to everyone listed below for the support, advice, and encouragement throughout this project, especially:

- Dr. Wendy Lee, for advising, for all the python coding I learned from you, and for the continuous positive energy you provided to move forward on this project.
- Dr. William Andreopoulos for serving on my committee and providing feedback on this body of work.
- Dr. John Lindbo, for allowing me to pursue this project and trial the development versions on live projects coming into the lab.
- The students’ past, present, and future in the MS Bioinformatics program for pursuing this line of interdisciplinary study and learning new things from each other.
- My family for making the space and time for me to pursue this endeavor.
# TABLE OF CONTENTS

1 INTRODUCTION ........................................................................................................ 1
   1.1 The Environment ........................................................................................................ 1
   1.2 The Business ............................................................................................................... 1
   1.3 Plant Breeding ........................................................................................................... 1
   1.4 Breeding tools ........................................................................................................... 2
   1.5 History of Markers in MAS. .................................................................................... 3
   1.6 MAS Data Processing Past to Present ................................................................... 4
   1.7 LIMS systems .......................................................................................................... 4
   1.8 Intro to LIMS-less system ....................................................................................... 5
      1.8.1 Introducing MASLGenerator ......................................................................... 5
      1.8.2 Introducing SPLi4MKGTR ......................................................................... 6
   1.9 Back to the Breeder Cycle ....................................................................................... 6

2 METHODS AND MATERIALS .................................................................................. 7
   2.1 Sampling .................................................................................................................. 7
   2.2 DNA Extraction ....................................................................................................... 8
   2.3 Standardized Plant List ........................................................................................... 9
   2.4 MASLGenerator ...................................................................................................... 10
      2.4.1 MASLGenerator Backend .......................................................................... 11
   2.5 Barcoding with Bartender ...................................................................................... 17
   2.6 Loading DNA plates, Mastermix and Thermocycling ........................................... 22
   2.7 Klustercaller Genotyping ....................................................................................... 26
   2.8 SPLi4MKGTR ......................................................................................................... 30
      2.8.1 Processing the genotyping file ...................................................................... 31
      2.8.2 Processing the Plant List file ......................................................................... 36
      2.8.3 Merging the Polished Plant List and Genotyping file ............................... 37
      2.8.4 Constructing a heatmap of the data .............................................................. 38

3 RESULTS .................................................................................................................. 39
   3.1 Results for Each Program ..................................................................................... 39

4 DISCUSSION .......................................................................................................... 42
   4.1 Faster Results Reporting ...................................................................................... 42
   4.2 Example of Process .............................................................................................. 48
   4.3 Customization with programming ........................................................................ 54
   4.4 Debugging .............................................................................................................. 56
   4.5 Limitations .............................................................................................................. 57
   4.6 Future ...................................................................................................................... 58

5 CONCLUSION ........................................................................................................ 58
   5.1 Conclusion .............................................................................................................. 58

REFERENCES .......................................................................................................... 59
LIST OF TABLES
Table 1 Plant List Submission Form ................................................................. 10
Table 2 Loading Patterns .................................................................................. 22
Table 3 Loading Marker Mix ............................................................................ 24
Table 4 KASP ..................................................................................................... 25
Table 5 Cleaned Plant List File ........................................................................ 37
Table 6 Genotyping Results Report Output for SPLi4MKGTR ............................ 40
Table 7 Genotyping Results Report Output for SPLi4MKGTR with Ranked Markers 41
Table 8 Summary of Demo-Project .................................................................... 48
Table 9 Genotyping Results Report Output for Demo-Project ............................ 53
Table 10 Project Summary output for MASLGenerator3 .................................... 55
Table 11 Project Summary output for MASLGenerator3 – per Marker ............... 56
Table 12 Debugging Program Crashes ............................................................... 57
A LIMS-LESS SYSTEM FOR GENOTYPING DATA IN MARKER-ASSISTED SELECTION

LIST OF FIGURES

Figure 1 Breeder Selection Cycle .............................................................. 7
Figure 2 Sampling .................................................................................... 8
Figure 3 MASLGenerator GUI ................................................................. 11
Figure 4 MASLGenerator Backend Step 1 ............................................... 12
Figure 5 MASLGenerator Backend Step 2 ............................................... 13
Figure 6 MASLGenerator Backend Step 3 ............................................... 13
Figure 7 MASLGenerator Backend Step 4 ............................................... 14
Figure 8 MASLGenerator Backend Step 5 ............................................... 16
Figure 9 MASLGenerator Backend Step 6 ............................................... 17
Figure 10 Editing Output Labels ............................................................. 19
Figure 11 BarTender Software Interface ............................................... 20
Figure 12 TipSpacing and Loading DNA ............................................... 20
Figure 13 Label Pattern and Loading DNA ........................................... 21
Figure 14 DNA Stamping ........................................................................ 23
Figure 15 Control Locations .................................................................. 24
Figure 16 Microplate Scanning and Output File .................................... 25
Figure 17 Importing Plate Reader File to KlusterCaller ...................... 27
Figure 18 Calling Clusters ...................................................................... 28
Figure 19 Linking SNP Labels to Clusters ............................................. 29
Figure 20 KlusterCaller Genotyping Export File ................................... 30
Figure 21 SPLi4MKGTR GUI ................................................................. 31
Figure 22 SPLi4MKGTR Backend Flowchart ....................................... 33
Figure 23 Config Files Used in SPLi4MKGTR ....................................... 34
Figure 24 Genotype Calls Updated ......................................................... 35
Figure 25 Sorting KlusterCaller Genotyping File .................................. 35
Figure 26 Removing Controls ................................................................. 36
Figure 27 Updating Plant List with Genotyping Data ............................ 38
Figure 28 Result Output For MASLGenerator ...................................... 40
Figure 29 Fixedness Results Output for SPLi4MKGTR ......................... 41
Figure 30 Fixedness Results Output for SPLi4MKGTR with Ranked Markers 42
Figure 31 Manual Data Handling Process Step 1 ................................. 44
Figure 32 Manual Data Handling Process Step 2 ................................. 45
Figure 33 Manual Data Handling Process Step 3 ................................. 46
Figure 34 Manual Data Handling Process Repeat ............................... 47
Figure 35 Manual Data Handling Process Step 4 ................................. 46
Figure 36 MASLGenerator Output for Demo-Project ............................ 49
Figure 37 Printing Labels for Demo-Project ........................................ 49
Figure 38 Labeling Plates for Demo-Project ......................................... 50
Figure 39 Exporting Genotyping Data From KlusterCaller for Demo-Project 51
Figure 40 Processing Demo-Project through SPLi4MKGTR .................. 52
Figure 41 Fixedness Results Output for Demo-Project .......................... 54
Figure 42 Modified Plant List Submission Form ..................................... 55
Figure 43 384-Well Plate to 1536-Well Plate ....................................... 58
1 INTRODUCTION

1.1 The Environment
The world population has just recently reached 8 billion people [1]. At the same time, the evidence of anthropogenic-induced climate change is bolder than ever before [2]–[4]. Climate change has created a daunting challenge and questions of how it will be possible to feed this new world as it adapts to the changing climate. It is more apparent than ever that food security is a global concern, and climate change exacerbates an existing problem [5]–[8].Growing food under these new circumstances will continue to require better use of natural resources, like space and water, for producing crops. Innovations across different sectors of agriculture will be needed.

1.2 The Business
Fossil fuels account for over 75% of carbon dioxide emissions; however, agriculture is still a contributor to the effects of climate change accounting for an estimated 11% of greenhouse gas emissions just in the US [9]. The business aspects of agriculture need to seek more sustainable practices, even at reasonable costs, to mitigate the contributions to climate change through sustainable methods [10]–[12]. One solution has been genetics in creating crops better suited to specific environments that can produce higher yields in smaller areas [12]–[14], reducing space and possibly water requirements. Having better traits suited for a particular environment is a task for plant breeders.

1.3 Plant Breeding
A commercial plant breeders' job is ultimately to create a product that will generate revenue for the company [10]. At the same time, that crop must be able to survive in several areas to be sold, so naturally, important traits are in the context of the current global situation. Plant breeders are working in the background of many seed companies in the research portions of companies. Still, they are also at the forefront of this fight creating new varieties that must survive under new environmental stresses worldwide [15]. Simultaneously Plant breeders are also combating time;
time to grow plants to make selections, time to create a new product for the market, and time of cycles to fit in a growing season. Yield is a typical trait of interest in many crops, but the yield is affected by many factors [15], [16]. Some external factors are plant pathogens, extreme weather events, nutrients, and water inputs. All of these factors need consideration when creating new cultivars. By creating new genetic combinations breeders can create varieties that under a variety of biotic and abiotic stresses.

1.4 Breeding tools
The breeder aims to develop a good product as quickly as possible. The usual turnaround for a new cultivar can take up to 10 years to build [17]–[19]. Since breeders are fighting time, anything that cuts down waiting time reduces their space footprint and produces "better" crops benefits the breeding program. Some innovative tools in modern breeding are the use of global growing sites, Speed breeding, genomic selection, breeding management systems, and Marker-Assisted Selection (MAS) [18], [20]–[25]. Growing crops worldwide increases their ability to grow crops and make crosses year-round instead of waiting for a season to return. Speed breeding is another tool breeders have deployed in many crops by controlling plant growth within a completely controlled environment with the use of growing lights and chambers to trick plants into developing by changing their environment [17]. Single nucleotide polymorphisms (SNP's) are single base change differences between two DNA sequences. Genomic selection allows the development of cultivars with many complex traits by coupling statistical models and predictive SNP's effects on the model to advance plants closer to the end of the breeding cycle [20], [26]. Breeding management systems have also provided a great deal of support in modern breeding to store, sort, and analyze breeding data such as pedigrees, harvest information, and notes and recipes for developing hybrids[22], [23]. Marker-assisted selection is also a frequently used tool by breeders to track known or predicted traits in their crosses to ensure the plants they are developing carry
specific characteristics. A major objective in plant breeding research is to discover genetic loci associated with specific phenotypes. Rapid tests for these loci can then be used to develop markers to track these genetic areas of interest across different crops for improvement in the cultivars [13], [27]–[30]

1.5 History of Markers in MAS.

Before the advent of molecular breeding, traditional plant breeding used phenotypes to decide the following crosses in the breeding cycles [31]. Traditional breeding takes longer because plants must be grown until they exhibit the traits of interest. A laboratory technique used early on by plant breeders was the use of Isozymes [32]. Isozymes use enzymatic reactions that researchers carried out in laboratory procedures showing different variants in the samples when put through gel electrophoresis. Although Isozymes provided some level of distinction between plants, it was limited to changes in the amino acid sequence and would not detect anything at the DNA level [33]. Restriction Fragment Length Polymorphisms (RFLPs) was the first turning point in efficiently using molecular markers for breeding [34]. Using RFLP markers allowed breeders to save some time by enabling the selection of plants at the seedling stage to make new varieties of crops. Polymerase chain reaction (PCR) allowed the development of more simplified markers, further speeding up the selection process. Some of the types of simple markers used were Random-amplified polymorphic DNA (RAPDs), amplified fragment length polymorphisms (AFLPs), and simple sequence repeats (SSRs). Although RAPDs, AFLPs, and SSRs were helpful, they lacked easy readability when visualized on gel electrophoresis. Markers such as cleaved amplified polymorphic sequences (CAPS) and sequence characterized amplified regions (SCARs) allowed easier visualization and readability of gels than their predecessor markers. The next challenge that needed improvement was that RAPDs, AFLPs, S.S.R.s, CAPS, and SCARs still required electrophoresis, quickly becoming more tedious and inefficient as sample sizes grew. Fluorescent
DNA labeling coupled with single nucleotide polymorphisms (SNPs) paved the way for modern genotyping systems. Fluorescent technology allowed for the advent of modern high-throughput genotyping techniques by using SNP markers with fluorescent labels to detect different alleles and eliminating the need for gel visualization [35]–[37].

1.6 MAS Data Processing Past to Present

In the early days of reading gels for isozymes, data could be recorded on paper and then transferred manually onto a spreadsheet. As sample sizes grew with PCR techniques, data was still managed in spreadsheets using excel and clever ways of sorting files for 96-well plates, but the process of running gels was still slow enough to limit the amount of data processed daily. Managing this amount of data in excel left room for many mistakes in copying and pasting columns and slowed down the ability to return results to breeders due to data management taking days to create results reports to return to breeders. When high throughput systems came into use, sample sizes remained manageable, adapting excel spreadsheets with sorting patterns. However, an extreme influx of data became apparent as time progressed and automation in laboratory processes became evident [25].

1.7 LIMS systems

Laboratory Information Management Systems (LIMS) have managed data processing through laboratories. Some significant benefits of using LIMS systems are the ability to track individual samples from arrival to reporting results [38]. Some limitation to LIMS for genotyping data is that free methods for this specific application are limited, specialized for specific equipment, and applicable to particular chemistries [24]. Some free LIMS systems are available for specific chemistry, such as Taqman; it is rare to find a free LIMS system for the KASP chemistry widely used in plant genotyping labs [39]. The most relevant LIMS system used in genotyping applications for KASP chemistry is Kraken LIMS system [40]–[42]. Kraken provides sample
tracking from start to finish and generates results for the processed genotyping data. The benefit of using the Kraken LIMS system is the ability to integrate LGC Ltd automation equipment for increased throughput and manage the equipment from the system. One disadvantage of Kraken is that you are limited to incorporating equipment from LGC. There is also a high price tag for many commercial LIMS systems. This cost can be prohibitively expensive for some labs, and customization is limited to what the system offers [43]. This project aims to provide a user-friendly alternative to the Kraken LIMS system for genotyping data from KASP chemistry. This alternative system uses the output from KlusterCaller genotyping software and marries this data to list plant sample data from breeders to create an excel file with the merged reportable results.

1.8 Intro to LIMS-less system

A LIMS-less system for processing genotyping data requires simple inputs and outputs. This system introduces a basic structure to allow further customization with some programming skills. The significant contribution of this LIMS-less system is the flexibility to move samples through the physical process of DNA Extraction on arrival and to piece the data related to the plates in a later step which is different from a LIMS system. LIMS systems require project and data setup before anything starts. The LIMS-less system designed for this project has two parts; One is a label-generating program executable with a user-friendly graphic user interface (GUI) to produce the labels to process DNA samples through the lab. The second program aims to merge the plant list files provided by breeders with the genotyping data generated by the MAS lab with the DNA samples.

1.8.1 Introducing MASLGenerator

Marker Assisted Selection Label Generator (MASLGenerator) is appropriately named, with the purpose of this program being to create labels that are needed downstream in the lab process. Typically DNA extraction is carried out in a 96-well plate format. In high throughput
genotyping, 96-well DNA plates get stamped into a 384-well plate format or a 1536-well plate format by robotics or semi-automatic pipettes. This program explicitly handles the data for stamping into 384-well formats, creating labels that follow a deliberate naming convention to allow the user to know in what position every 96-well block will go into the 384-well plate. An additional feature of this program is that PCR reagent measurements for the project are also output, saving time in prepping assays and master mix.

1.8.2 Introducing SPLi4MKGTR

KlusterCaller is a pay-to license genotyping analysis software that is widely used in genotyping labs and used to export genotyping results for use with the program created in this project. Simple Plant List for Merging KlusterCaller Genotyping Results (SPLi4MKGTR), another appropriately named program packaged as an executable with a user-friendly GUI. This program focuses on taking exported genotyping data from KlusterCaller software, sorting the values to its 96-well sample pattern then merging the genotyping calls to their appropriate plate in the breeder-provided plant list. The output of this program is an excel file containing all the results for their submitted samples. Some key features of this program are the ability to handle ranked markers when different plates require different markers and provide a visual overview of the project showing the fixedness of the breeding lines sampled in a heatmap. Fixedness is a measure of a group of plants, how many are homozygous, represented with a 1 or 0, and heterozygous individuals represented as 0.5, taking the average, giving a measure of which direction plants are being "fixed".

1.9 Back to the Breeder Cycle

With the ultimate goal of providing better crops for a growing population, the laboratory processes and data management contributes back to the ability of breeders to make selections in a much faster time frame. The goal to automate the data process of marker-assisted selection
allows them to squeeze more cycles per season and create better cultivars faster, continuously feeding into the cycle of selecting better plants until commercialization is ready (Figure 1).

![Breeder Selection Cycle](image)

Figure 1 Breeder Selection Cycle

The process of breeding new cultivars involves a cycle of selecting traits repeatedly with the help of Marker Assisted selection over the years, growing the next generations, and selecting again until a desired set of characteristics gets incorporated into the cultivar for commercialization.

2 Methods and Materials

2.1 Sampling

In this project, the breeders' teams sampled using a typical deliberate sampling pattern for harvesting tissue into deep well plates (Figure 2). Depending on the crop, harvesters collected cotyledon or true leaf into a 96-well deep block. Harvesting with tweezers requires taking about 50mm x 50mm pieces or two 6mm hole punches of tissue from each plant. The first harvested sample goes into the first well of the 96-well plate, with each following sample put into the well below it, filling up the block by column. In the final column of each 96-well plate sampled, the last four cells located at positions E12, F12, G12, and H12 are left empty to allow space for controls when processing in the lab. The breeders' team of harvesters ship the sampled plates sealed with
breathable tape and placed with a silica sachet bag placed on top to dry the samples and packaged into a ziplock bag for shipping.

Figure 2 Sampling

Sampling plants into 96-deep well plates for DNA extraction is carried out by taking two 6mm hole punches and transferring them to the deep well plate with each successive sample going below the previous, then starting at the top of the next column once a column gets filled.

2.2 DNA Extraction

The lab used a standard 96-well plate DNA extraction method with a high-salt extraction buffer and alcohol precipitation. The lab first receives plates and spins them down in a centrifuge at 3396 RCF to unstick samples from the breathable tape. Each well gets loaded with a high salt extraction buffer, and samples get macerated using a SPEX Geno/Grinder at 1400RPM for 4 minutes. Samples get centrifuged for 2 minutes at 3396 RCF and then placed in a water bath at 65°C for 45 minutes. After the heat treatment, 150µl of ammonium acetate is added to each well and spun down for 20 minutes at 3396 RCF. 100µl of the supernatant gets transferred to a 200µl cell culture plate, followed by 100µl of Isopropyl alcohol. The cell culture plates get spun at 2683 RCF to pellet the DNA, then dumped, followed by a wash step with ethanol, spinning down once again, and then discarding the supernatant again. The plates get left on the bench to dry until ready to be
hydrated with TE buffer. This extraction method yields a range of concentrations between 1 and 7 ng/ul, dependent on consistent sample sizes and the quality of the tissue harvested.

2.3 Standardized Plant List

Plant samples must be traced back to their origin for the breeder to select which plants they want to keep. As a part of this project, I developed a standardized plant list submission form in Microsoft excel for breeders to submit their marker requests (Table 1). The plant list template gets populated with the breeder's information to trace a specific plant sampled back to a position in a greenhouse tray or a particular plant in a field. The required fields in this excel workbook are 'Plate #', 'Well', 'FEID', 'Stake', 'Plant', and 'Markers'. The 'Plate #' column contains the information for the 96-well plate number sampled. This column is pre-filled in the template and provides positions for up to 400 plate projects. The 'Well' column contains 96-well locations for each block, indicating an x and y coordinate on each plate designated by a letter for the row and a number for each column. The 'Well' column in the excel file is pre-filled since it pertains to a location of the 96-well plate format that is physically bound to a rigid plate and cannot change. 'FEIDs' are relevant for breeders working in the Phenome database. The FEID is an identifier of the plant in a database allowing the breeder to see more information about the plant. This column gets passed through the data process; if the breeder does not use a database in their downstream analysis, it gets filled as a null value of 'None'. This form's 'Stake' field refers to the information the breeder requires to know which breeding line in the greenhouse or field to trace back. An example of what the 'Stake' column represents is the plastic stakes seen in (Figure 2) defining a breeding line or, more simply, a group of plants. Lastly, the 'Plant' field contains the sample plant number. This field, coupled with the 'Stake' column, gives the location of a specific plant; for example, using (Figure 2 and Table 1) Stake + Plant gives you the place on the tray of plants by locating the particular stake and counting down the number of cells in the tray. The last critical field is 'Markers'; this contains a list of DNA
markers requested to run on each sample of a submitted project. Once the breeder populates the excel file with the necessary information, it gets submitted to the genotyping lab for processing. The To_csv tab in this excel file compiles the tabs for the plate information stacking plates 1 to 400 in a project to easily save the data as a .csv file to run through the label-generating program.

Table 1 Plant List Submission Form

Plate # is the DNA extraction plate the harvester samples it in. Well is the coordinate position on a 96-well block where the sample goes. FEID, if available, refers to more information on the sample stored in a Database. The ’Stake’ is the breeding line or group of plants, the Plant column contains the plant number within each group, and the Markers column will have all the desired markers for each sample. The breeder must track plant samples digitally to know where each plant is in physical space when the marker results get reported to make selections. Each row in the table represents an individual sample with all the information necessary to trace it back to its corresponding tray in a greenhouse or field.

<table>
<thead>
<tr>
<th>Plate #</th>
<th>Well</th>
<th>FEID</th>
<th>Stake</th>
<th>Plant</th>
<th>Markers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A01</td>
<td>None</td>
<td>ex.Line1</td>
<td>2</td>
<td>M1,M2,M3</td>
</tr>
<tr>
<td>1</td>
<td>B01</td>
<td>None</td>
<td>ex.Line1</td>
<td>3</td>
<td>M1,M2,M3</td>
</tr>
<tr>
<td>1</td>
<td>C01</td>
<td>None</td>
<td>ex.Line1</td>
<td>4</td>
<td>M1,M2,M3</td>
</tr>
<tr>
<td>1</td>
<td>D01</td>
<td>None</td>
<td>ex.Line1</td>
<td>5</td>
<td>M1,M2,M3</td>
</tr>
<tr>
<td>1</td>
<td>E01</td>
<td>None</td>
<td>ex.Line1</td>
<td>6</td>
<td>M1,M2,M3</td>
</tr>
<tr>
<td>1</td>
<td>F01</td>
<td>None</td>
<td>ex.Line1</td>
<td>7</td>
<td>M1,M2,M3</td>
</tr>
<tr>
<td>1</td>
<td>G01</td>
<td>None</td>
<td>ex.Line1</td>
<td>8</td>
<td>M1,M2,M3</td>
</tr>
<tr>
<td>1</td>
<td>H01</td>
<td>None</td>
<td>ex.Line1</td>
<td>9</td>
<td>M1,M2,M3</td>
</tr>
<tr>
<td>1</td>
<td>A02</td>
<td>None</td>
<td>ex.Line2</td>
<td>10</td>
<td>M1,M2,M3</td>
</tr>
<tr>
<td>1</td>
<td>B02</td>
<td>None</td>
<td>ex.Line2</td>
<td>1</td>
<td>M1,</td>
</tr>
<tr>
<td>1</td>
<td>C02</td>
<td>None</td>
<td>ex.Line2</td>
<td>2</td>
<td>M1,</td>
</tr>
<tr>
<td>1</td>
<td>D02</td>
<td>None</td>
<td>ex.Line2</td>
<td>3</td>
<td>M1,</td>
</tr>
<tr>
<td>1</td>
<td>E02</td>
<td>None</td>
<td>ex.Line2</td>
<td>4</td>
<td>M1,</td>
</tr>
<tr>
<td>1</td>
<td>F02</td>
<td>None</td>
<td>ex.Line2</td>
<td>5</td>
<td>M1,</td>
</tr>
<tr>
<td>1</td>
<td>G02</td>
<td>None</td>
<td>ex.Line2</td>
<td>6</td>
<td>M1,</td>
</tr>
<tr>
<td>1</td>
<td>H02</td>
<td>None</td>
<td>ex.Line2</td>
<td>7</td>
<td>M1,</td>
</tr>
<tr>
<td>1</td>
<td>A03</td>
<td>None</td>
<td>ex.Line2</td>
<td>8</td>
<td>M1,</td>
</tr>
<tr>
<td>1</td>
<td>B03</td>
<td>None</td>
<td>ex.Line2</td>
<td>9</td>
<td>M1,</td>
</tr>
<tr>
<td>1</td>
<td>C03</td>
<td>None</td>
<td>ex.Line2</td>
<td>10</td>
<td>M1,</td>
</tr>
<tr>
<td>1</td>
<td>D03</td>
<td>None</td>
<td>ex.Line3</td>
<td>1</td>
<td>M2,</td>
</tr>
</tbody>
</table>

2.4 MASLGenerator

MASLGenerator is an easy, user-friendly tool to create labels in a deliberate pattern to work downstream with SPLI4MKGTR. The first step in processing data is to take the .csv file created
from the breeder-provided standardized plant list and trim the end of the file by deleting everything after the last plate information. For example, if the breeder only has 24 plates in their project, you delete plates 25 to 400. Once this file gets trimmed, it can be saved and run through MASLGenerator. MASLGenerator is an executable file with a simple GUI shown in (Figure 3) asking the user to browse to the location of the plant list .csv file. The second user input of the GUI is the project name that will be attached to every label created. After clicking submit, a file titled labels.txt file is created or updated in the directory of the MASLGenerator.exe file. The labels.txt file will list all the labels needed to run the project on 384-well PCR plates, and reagent measurement requirements will also be listed. The contents of the labels.txt file get copied and pasted into the standardized plant list file the breeder submits for the project on separate tabs for the labels and one for reagents.

2.4.1 MASLGenerator Backend

The backend of the MASLGenerator program includes six significant steps. The program pulls from the .csv input all the unique values in the 'Markers' column and all the unique plate numbers from the 'Plate #' column and stores them as lists (Figure 4). The list of unique plates gets transformed into an array to start a new DataFrame. Within a for loop, looking at the original input
DataFrame in the 'Markers' column, the list of unique markers is taken per plate, creating a list of markers per plate. This list is then added to the DataFrame with the unique plate numbers in the following column (Figure 5). The list of unique markers for the entire project pulled earlier can now get added to the DataFrame build, with each marker as a header of an empty array with placeholders for the length of the DataFrame (Figure 6). Now using a for loop and some conditional statements, the array for each marker header is filled out, asking if the header value is in the 'Markers' column list for the row, fill in the value for that row at the header column with an X (Figure 6).
The unique plate numbers pulled from the first step start a new DataFrame (shown on the right). The unique Markers for each plate shown in the circles get added to the new DataFrame with the unique plate numbers.

for loop:

If Header M(i) in Markers for plate (i):

Add an “X”

else: Pass

This figure shows the following pieces added to the new DataFrame. The unique Markers across all plates pulled in the first step are now used and added individually as an array with placeholders creating a grid. This grid is populated by X’s looking at each row and seeing if the marker in the header is in the Markers column for that row filling in X’s where this condition is satisfied.
for loop:
If in column M(i) there is an X:
add current plate number to a list

<table>
<thead>
<tr>
<th>Plate #</th>
<th>Markers</th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>M1,M2,M3</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>M1,M3,M6</td>
<td>x</td>
<td>-</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>3</td>
<td>M1</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>M1,M2,M6</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>x</td>
</tr>
</tbody>
</table>

Figure 7 MASLGenerator Backend Step 4
With the grid of X's, another new DataFrame that contains the unique Markers as the index is made. The plates for each marker are put into a list by a conditional statement checking if there is an x for a particular marker and adding the plate number to a list creating a list of plate numbers later combined with its respective marker in the new DataFrame.

Iterations of the for loop conditional creates a grid with markers running along the top and the DataFrame and an X value for each plate requiring that marker. Another DataFrame with unique markers as the index gets constructed. Then a plate numbers list is built in a second for-loop conditional. Asking every iteration: Is there an X value in the plate row for the header column? If it returns a true, add the plate number to the list for the marker name (Figure 7). At this point, you have all the information you need to create labels stored in a DataFrame of all the plates that require a particular marker. To construct the string for each label, a nested for loop takes each list of plates and chunks it into four. Making the label string takes the user's input project name, separated by a dash, then the four plates separated by dashes, and then the marker name from the index of the DataFrame holding the list of plates for each marker (Figure 8). A list of plate labels stores all the labels after the nested for loop. The list of labels gets edited by using a regular expression to detect labels that contain only one plate. If a label has one plate, it is removed from the list, edited to repeat the plate twice, and then added back to the list of labels. Once all the labels are ready, the
next step is determining the volumes of master mix and assay needed for each marker. Calculating reagent needs begins by taking an intermediate action earlier in the label-creating process, counting the number of instances a marker occurs for each label, and storing this information in a dictionary with markers as keys and counts as values, giving the number of reactions necessary. An external config file contains the measurements of the master mix and assay for a single reaction, the volume needed for one full 384-well plate of master mix, and the average difference of needed overage as the number of plates increases. With the values from the config file, calculating the master mix needs gets done in a conditional statement. If there is a single 384-well plate needed, use the master mix needs of one plate from the config file, divide it by 2.5ul and multiply by the measurement of the master mix to get the required master mix reagent and by the assay measurement for one reaction for the assay needs. If there is more than one plate, multiply the number of plates in the dictionary for the marker times the average difference divided by 2.5 and multiply the measurement of master mix for one reaction separately by .07 to find the volume of assay needed (Figure 9). Finally, the remaining step is writing the output and the labels to the labels.txt file. Going back to the Plant breeder submission form, the information in the text file gets copied into the excel file on new tabs for printing the labels with a corresponding barcode.
Marker Labels = []
List of lists of Plates = []
for i in index:
    for j in chunks(list of plates, 4):
        add i to Marker Labels list
        add j the list of 4 plates to List of lists of Plates

Project Name Input + (List of lists of Plates, Marker Labels)
Replace brackets, commas, and spaces appropriately to make labels

Figure 8 MASLGenerator Backend Step 5

Using the plate list DataFrame, the label information is pieced together in a nested for loop. With all the information in the correct order, it is converted to a string, replacing brackets, commas, and spaces.
A LIMS-LESS SYSTEM FOR GENOTYPING DATA IN MARKER-ASSISTED SELECTION

for loop:
- Count the number of labels with marker i
- if i is only one then use volume for one full plate in calculation
- if i is greater than one use the average difference value in the calculation

<table>
<thead>
<tr>
<th>Description</th>
<th>Abbreviation</th>
<th>ul</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vol Mix Needed for one full 384-Well Plate</td>
<td>Mix Full Plate</td>
<td>1150</td>
</tr>
<tr>
<td>Average Difference in Mix Needed for More than one plate</td>
<td>Avg Diff Multiple Plates</td>
<td>1058</td>
</tr>
<tr>
<td>Mastermix and assay required in one reaction</td>
<td>MM+asy</td>
<td>2.5</td>
</tr>
<tr>
<td>Vol Mastermix for one reaction</td>
<td>MM</td>
<td>2.43</td>
</tr>
<tr>
<td>Vol Assay for one reaction</td>
<td>asy</td>
<td>0.07</td>
</tr>
<tr>
<td>Mastermix Volume for Marker i in this project</td>
<td>Marker MM for Project</td>
<td>?</td>
</tr>
<tr>
<td>Assay Volume for Marker i in this project</td>
<td>Assay for Project</td>
<td>?</td>
</tr>
</tbody>
</table>

Marker MM for Project = \[
\frac{\text{Mix Full Plate (or Avg Diff Multiple Plates)}}{\text{MM+asy}}\] *MM

Assay for Project = \[
\frac{\text{Mix Full Plate (or Avg Diff Multiple Plates)}}{\text{MM+asy}}\] *asy

Figure 9 MASLGenerator Backend Step 6

Counting the labels for each marker allows the calculation of the necessary mastermix and assay needed. A conditional statement is required because more overage is needed when only one 384-well plate is loaded. However, as the number of plates with the same marker increases to more than one, the average difference in overage for many plates can be used to minimize reagent waste.

2.5 Barcoding with Bartender

BarTender software is a label printing program that takes information from different databases, including excel workbooks, and pulls information to print labels. The BarTender can link to the excel sheet where the labels are stored, importing the data to design a printable label. Before importing labels into BarTender software, they must first be stored in the master Plant List file and slightly edited where plates with the same cycling can be combined (Figure 10). The software has several barcode formats, but for this project, it is necessary to use code 128 barcode, which returns ASCI text. The barcode returns the exact label when scanned (Figure 11). Each 384-well PCR
plate gets two labels, one with readable text and the second is a barcode. The readable label adheres to the front of the 384-well plate, and its corresponding barcode is on the left side. The user loading plates translate the readable label to the physical placement of the correct 96-well DNA plate in the proper position of the 384-well plate quadrant (Figure 12). Four 96-well plates stamp in a quadrant style due to the spacing of tips and the smaller well size of 384-well PCR plates.

This quadrant loading will have the plate loaded in position one in the alternating wells starting at A1, skip the next row in the 384-well plate, and the second row of the same 96-well plate will begin in C1 alternating again. The alternating pattern continues for the remaining 96-well plates going into the 384-wells. The starting positions will differ with A2 for the second plate, B1 for the third, and B2 for the fourth plate. Designating the positions as quadrants q1, q2, q3, and q4, as shown in (Figure 13), the corresponding label will appear as 'project-name-q1-q2-q3-q4-marker'. The position of the q's allows the user loading plates to load correctly, which is critical when sorting the data in the downstream process. There are a few labeling scenarios outlined in (Table 2). Loading the 96-well DNA follows a specific order into the four quadrants ordered as q1=A1, q2=A2, q3=B1, and q4=B2. The label's structure depends on how to fit the 96 samples into a 384 format. Some ways are two 96-well plates in the 384 format, three 96-well plates in a 384-format, or four in the 384-well format. Also, if there are two markers in one 384-well container with quadrants, A1 and A2 are filled with plates that need one marker, and B1 and B2 get filled with plates needing a different marker.
Figure 10 Editing Output Labels

The labels generated in MASLGenerator are output as a .txt file, so the information in the text file must be copied and pasted into the master .xlsx Plant List file as separate tabs, one for the labels and one for the mastermix volumes. The labels with two plates get condensed by including two more plates with one different marker. Manual editing of this label is optional but saves the use of another dish by not leaving empty wells. This step remains manual due to the different thermocycling protocols required for some markers.
Figure 11 BarTender Software Interface

Above is the interface of the barcoding software. The red boxes point out critical portions needed to print the labels. The software can link to an excel file and pull the information from a particular tab. In this case, it grabs the labels tab and uses the data stored in this location to populate the barcode and the user-readable label.

Figure 12 Tip Spacing and Loading DNA

Multichannel pipettes or 96-well heads are a must to transfer 96-well DNA plates into a 384-well format in high-throughput settings. The spacing between standard multichannel pipettes and 96-well heads of tips is larger than 384-well plate spacings resulting in quadrant-style loading. Each tip for one 96-well pipette tip head loads in alternating well positions on the 384-well plate.
The user can read the printed labels by interpreting the position on the label about a particular quadrant in the 384-well plate. In the image above, plate one is loaded into q1, corresponding to the black color in the diagram representing the 384-well plate. The other corresponding color designations are by quadrant q, with q2 being red, q3 blue, and q4 green.
2.6 Loading DNA plates, Mastermix and Thermocycling

The digital process gets translated to the physical actions of carrying out the instructions with the labels created to stamp DNA into the 384-well plates according to the labels using a multichannel liquid handler (Figure 14). Alternatively, a semi-automatic plate stamper such as an Apricot system or a 96-well pipette will work as long as the DNA can go into the necessary pattern. Note that the last four tips of column twelve are left empty to save space for controls. The last sixteen wells of the 384-well PCR are empty after the DNA is stamped to its particular pattern according to the label, leaving room for controls for each marker (Figure 15). The master mix for this project was loaded using a Meridian liquid handler; However, this can be loaded manually with a multichannel repeater, but in high-throughput processes, liquid handlers are much faster.
For 384-well plates with only one marker on the entire plate, the master mix goes into all four quadrants adding 2.5 ul in each well. If the plate requires two markers, the first goes according to the label in quadrants 1 and 2, and the second is into quadrants 3 and 4 (Table 3). When all the plates are loaded with DNA, Mastermix, and sealed, they are ready for PCR. Competitive allele-specific PCR (KASP) detects different alleles' via SNPs to allow sample genotyping [37]. KASP uses fluorescent labeling of the different SNPs in the sample to see which allele is present in a sample. Standard KASP conditions (Table 4) or a slightly modified version (increasing from 30 cycles to 35 or 40 in the annealing step) are the protocols used. The PCR happens in a high-throughput thermocycler system like a Hydrocycler or single-unit PCR machine. The KASP PCR program runs for about an hour and a half; after this, a microplate reader detects the fluorescent signals of the KASP reactions after being scanned. The plate reader used for this project was a Pherastar microplate reader with a stacker. The stacker has a barcode reader to check the plates and name the output reader file according to the label of the plate (Figure 16). After scanning each plate with the microplate reader, the generated files get imported into the KlusterCaller software for genotyping.

Figure 14 DNA Stamping

The use of robotics allows efficient DNA stamping for high throughput. The first image (left) is a fully automated DNA stamping robot transferring DNA from the 96-well DNA plates to 384-well PCR plates. The second image (middle) is a semi-automatic plate stamper requiring the user to switch tips and move the plate to specific spaces for loading. The image on the right is also semi-automatic but requires more manual aspirating and dispensing steps.
Figure 15 Control Locations

In every 384-well PCR plate, the last sixteen wells noted above in the black box are reserved to load with DNA controls and negative controls and are left empty in the DNA stamping process.

Table 3 Loading Marker Mix

The DNA-loaded plates have a marker mix added to each PCR plate according to its label. The appropriate marker information is at the end of each label, telling the user what marker pairs with what quadrant. In the first label example, the marker M1 is loaded into all the wells since each DNA loaded into this PCR plate requires M1. The second label designates only three quadrants that will require marker M6. The following plate packs the mix into quadrants 1 and 2 only, leaving half the plate empty. The fourth label has two markers in the PCR Plate where marker M3 is into quadrants q1 and q2 and M6 filled for q3 and q4.
Table 4 KASP

KASP Thermocycling protocol

<table>
<thead>
<tr>
<th>Temperature °C</th>
<th>Time</th>
<th>Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>94</td>
<td>15 min</td>
<td>1</td>
</tr>
<tr>
<td>94</td>
<td>20 s</td>
<td>10</td>
</tr>
<tr>
<td>61 (-0.6 per cycle)</td>
<td>60 s</td>
<td>30 (*)</td>
</tr>
<tr>
<td>94</td>
<td>20 s</td>
<td>1</td>
</tr>
<tr>
<td>55</td>
<td>60 s</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>hold</td>
<td>1</td>
</tr>
</tbody>
</table>

* some markers use an increased number of cycles at 35 or 40

Figure 16 Microplate Scanning and Output File

A Pherestar reader with a stacker capability and barcode scanner is helpful for the high throughput of samples. A plate reader measures the fluorescence values corresponding to the different alleles in each sample. The file generated from the plate reader scans the barcode, names the file with the barcode name, and inputs this name into the output as ID1. The organization of the output file is a general header with the parameters used in scanning the plate, followed by the chromatic readings below.
2.7 KlusterCaller Genotyping

The microplate reader files will correspond to each 384-well plate scanned with the label name as the file name. In the KlusterCaller software, the files imported belong to one project in a project folder (Figure 17). The user inspects and analyzes the clusters for each imported plate scan by lassoing uncalled points corresponding to where the controls cluster (Figure 18). The next step is linking each plate file to a SNP from an excel file storing the SNP allele designations. The SNP allele call file is imported into the software to connect each marker with its corresponding allele calls (Figure 19) to designate the clusters’ X, X:Y, and Y coordinates. Exporting results as a .csv file is done after the plates have been checked and linked to their SNP calls (Figure 20). Note that the genotyping data has no ties to the plant list file provided by the breeder since each plate is in a 384-well format, and the wells do not correspond to the 96-well plates at this time. With the genotyping results exported, SPLi4MKGTR developed in this project is used to create the final results reportable to the breeders.
Figure 17 Importing Plate Reader File to KlusterCaller

The user first creates a project in KlusterCaller software (left), then imports the plate reader files containing the fluorescence values (right).
Figure 18 Calling Clusters

Cluster calling by position, the user uses red to represent the Y allele position, blue to represent the X allele position, and Heterozygotes as green.
Figure 19 Linking SNP Labels to Clusters

The end user must link each plate read file to its appropriate marker in the software by first uploading a table of the labels for the clusters. Users can then link each plate to its suitable marker by selecting the linked SNP box when viewing a specific plate and selecting the appropriate marker indicated by the plate filename. A marker is attached when the plate read file is followed by [SNP <marker>] in brackets.
Figure 20 KlusterCaller Genotyping Export File

The Genotyping file exported from KlusterCaller is structured as shown above. The blue box shows the sections of the file not used for the downstream processing but provides general details of the project. The portion in the green box is used for processing by the SPLi4MKGTR program. The main columns pulled from this data are the DaughterPlate, MasterWell, Call, and SNPID columns. The MasterPlate is not used in the genotyping process but would be used in a LIMS system if plates were being tracked at each step.

2.8 SPLi4MKGTR

Like MASLGenerator, SPLi4MKGTR has an easy-to-use interface with a GUI where the user can provide the input files and what they would like their output file named (Figure 21). This program asks first for a Genotyping file, the exported csv file generated by KlusterCaller software; the second file it prompts for is the plant list in csv format, which is the same file used in MASLGenerator to create the labels. The user can then hit submit, and the compiled results excel file and a heatmap showing the project's fixedness level shows up in the directory of the SPLi4MKGTR executable file.
2.8.1 Processing the genotyping file

The goal of SPLi4MKGTR is to piece together the genotyping data and the plant list together. Figure 22 shows a schematic demonstration of the processes carried out by this program. First, the genotyping file organization has to fit the plant list to begin pulling pieces together. The program takes in the genotyping file to find the start position of the genotyping data using a function that detects where the "Daughter plate" column begins in the csv file returning the start position to input when the genotyping data gets imported into a pandas DataFrame. Next, a config file containing the sorting pattern of the 384-well plate to a stacked 96-well plate format is imported and merged with the genotyping data set by the 384-well column positions. At this point, several iterative tasks add information to the DataFrame. First, the entire DataFrame gets chunked out into portions of 384 rows and sorted by the sample id column of the imported sorting config file information (Figure 23). Next, for each chunk, conditional statements coupled with functions created to detect the plates in the plate label are used to retrieve the 96-well plate used for each 384-well piece. The functions use regular expression detecting -q1-q2-q3-q4- and pulling out the plate numbers and inserting them with a second function checking if the plate in the sorting config file corresponds to the correct position. If the conditional is true, then the 96-well plate number is
added to a list, and once an iteration is complete, the list of 96-well plates gets added as a column to the 384-well chunk (Figure 24). A list of DataFrames after the for loop completes now contain a new column showing the 96-well plate tied to each 384-well sample. Each 384-well chunk in the list of DataFrames is stacked back together into a big DataFrame ready for the following process. Next, transcribing the genotyping calls according to the genotyping calls config file allows the SNPID column to change from ratio calls to readable calls for the breeder (Figure 23 and 25). Genotype calls in the program get translated by mapping the SNPID column with a function, taking the dictionary created using the data from the genotyping calls config file, and renaming the values according to the dictionary. The final version of the genotyping DataFrame is almost complete except for one detail. Recall that the data contains 16 samples holding controls; luckily, the sorting config file pushes these data points to the end of each 384-well chunk when sorting was carried out. The genotyping DataFrame gets broken up again into 384-well chunks and positions 0 to 368 for each piece added to a new stacked DataFrame with no controls (Figure 26).
A LIMS-LESS SYSTEM FOR GENOTYPING DATA IN MARKER-ASSISTED SELECTION

Figure 22 SPLi4MKGTR Backend Flowchart

Above is a Schematic of the overall backend process of the SPLi4MKGTR program.
The image above shows a piece of the three config files used in SPLi4MKGTR. The sorting config file is used to carry out sorting the data from the 384-well plate format to the 96-well plate format for the data contained in one 384 piece chunk. The Genotyping Calls Config file is used for combining calls to their respective reporting designation. The Genotype Value Config File is used to convert the calls into values of 0, 1 depending on homozygous calls and 0.5 if the call was heterozygous.
The lab reports the data to breeders as a single call representing the trait of interest. The Calls output from the genotyping export in KlusterCaller report as allele calls. So as an example, for disease markers R:R indicates resistance to a disease but must report as R and S:S represents a susceptibility to the particular disease but must report as an S, and H is if the plant is heterozygous for the marker.
2.8.2 Processing the Plant List file

The program's next portion is doing some simple edits to the imported plant list file. It checks the length of the file first to see if the empty wells of each 96-well sample plate persist in the file. It checks this using a conditional statement if there is a remainder when the file length divides by 96, and the unique values for samples 0 to 96 equal one unique value. When the conditional is true, the file breaks into pieces of 96 and removes the four empty wells in each plate from the end of the file chunk. If the file imported does not contain the blank wells, then the first condition fails and moves to the next if statement checking if the plant list file length divided by 92 has no remainder. And if this is true, it will pass the file to the next portion of the code. At this point, it pulls in the information from the plant list to create a DataFrame adding empty arrays with the unique SNPID markers from the genotyping file as columns to this DataFrame (Table 5).
Table 5 Cleaned Plant List File

The cleaned-up plant list file will look like the table below, depending on how many plates are in a project. The unique markers pulled from the genotyping file SNPID column create placeholders where the data for each sample will be populated using the cleaned genotyping DataFrame.

<table>
<thead>
<tr>
<th>Plate #</th>
<th>Well</th>
<th>FEID</th>
<th>Stake</th>
<th>Plant</th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A01</td>
<td>None</td>
<td>ex.Line1</td>
<td>2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>B01</td>
<td>None</td>
<td>ex.Line1</td>
<td>3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>C01</td>
<td>None</td>
<td>ex.Line1</td>
<td>4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>4</td>
<td>B12</td>
<td>None</td>
<td>ex.Line140</td>
<td>3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>C12</td>
<td>None</td>
<td>ex.Line140</td>
<td>4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>D12</td>
<td>None</td>
<td>ex.Line140</td>
<td>5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

2.8.3 Merging the Polished Plant List and Genotyping file

Now there are two main polished DataFrames one has a plant list with all the headers of the markers ran for the project containing empty placeholders. The second polished file is a DataFrame of the genotyping results containing all the information needed to fill in the blank placeholders in the other. An empty list is created and filled with 92 chunk DataFrames by a nested for-loop first chunking out the plant list DataFrame in chunks of 92. Second, it appends the piece of 92 to the list. In a second for loop, it chunks the genotyping DataFrame by 92 and checks a conditional statement. First, if the unique plate number value in the 92-chunk plant list DataFrame is equal to the unique value in the 96-well plate number in the genotyping-chunk DataFrame. Second, the Genotype column unique value is not equal to ND: fill in the column of the plant list with the genotyped data from the genotyping file (Figure 27). If this is not satisfied, it will simply pass and continue the nested loop. The idea here is that it is iteratively checking for the matching header with the appropriate data from the genotyping file. Now you have a list of DataFrames chunked out in 92, but all the genotyping data is now populated. To recompile the list of DataFrames into one, a for loop appends each DataFrame in the list to one big data frame. The
constructed data frame is used as the results output to excel named with the users' desired output filename asked in the beginning.

for loop 1:
   - Chunk Plant List DataFrame by 92 and append current chunk to a list of DataFrames
for loop 2:
   - Chunk the genotyping results DataFrame by 92
   Condition:
   if the Plate # in the Plant List = Plate # in the genotyping results in both 92 chunks:
      - Update the Plant List with marker information from the genotyping results
   else:
      - Continue to next 92 chunk

Figure 27 Updating Plant List with Genotyping Data

The Pseudocode above shows how a nested for loop updates the plant list file with the corresponding marker data from the genotyping results. Both DataFrames are broken into pieces of 92 and compared to each other by the Plate #; if the plate number matches and there is data (i.e., not full of ND values), then it updates the plant list for a specific marker from the genotyping 92-chunk. If the two Plate #s do not match, nothing happens, and the for loop continues to the next 92-chunk.

2.8.4 Constructing a heatmap of the data

Constructing a heat map of the data requires an allelic_values config file import. The purpose of this file is to convert the genotyping calls into an expected numerical value for an allele classification. An example is for a disease marker; one would indicate resistance and susceptibility as a zero. For heterozygous samples, they classify as 0.5. This config file is imported as a dictionary and updates the results DataFrame used previously to export the results and drops all the columns that are not marker data except for the stake column, which is the breeding lines or the groups of the material run on the project. The data grouping by the mean of the sample genotypes groups on the assigned values in the 'Stake' column. A heat map construction shows an overview of the project with which lines are fixed for a particular marker or still need more work.
depending on the breeder's goals. The heat map reports as PDFs showing the results for every group run, and the color represents the fixedness of the group running in the project.

3 Results

3.1 Results for Each Program

The output for the MASLGenerator program produces labels for processing plates shown in (Figure 28). The output for SPLi4MKGTR reports results in an excel file with the plant list and the marker results tied to each sample. This output can look slightly different depending on the structure of the project. Results for a project with all the same markers on every plate will have every cell for each marker filled in with genotyping results (Table 6). In cases where the breeder has selected ranked markers for particular groups of plants and not for others, there will be cells with a dash taking the place of the genotyping data (Table 7). In each case, the output heat map will mirror the results files condensed by group (Figure 29) and (Figure 30). Once again, the heat map for a project running all the same markers on all samples will fill the entire heat map, whereas the project with ranked markers will only contain the groups for the requested markers. A critical focus of this project was being able to compile results automatically vs. manually managing data involving several copy-and-paste steps of results. With the two programs shown here, the time to compile results reduced from days and hours, depending on project size, to compiling data in a matter of seconds regardless of project size.
Figure 28 Result Output For MASLGenerator

The image above is the output for MASLGenerator for a small synthetic dataset to demonstrate the entirety of an output.

Table 6 Genotyping Results Report Output for SPLi4MKGTR

Results.xlsx output from SPLi4MKGTR with the same markers run across all samples.
Table 7 Genotyping Results Report Output for SPLi4MKGTR with Ranked Markers

Results.xlsx output from SPLi4MKGTR with only selected ranked markers chosen.

<table>
<thead>
<tr>
<th>Plate #</th>
<th>Well</th>
<th>FEID</th>
<th>Stake</th>
<th>Plant</th>
<th>Ph2</th>
<th>Ph3</th>
<th>Sft</th>
<th>Ty1</th>
<th>Ty3</th>
<th>Ty6</th>
<th>hp-1</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1 A1</td>
<td>6312071</td>
<td>5</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>1 B1</td>
<td>6312072</td>
<td>5</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>1 C1</td>
<td>6312073</td>
<td>5</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>-</td>
<td>-</td>
<td>H</td>
<td>-</td>
<td>-</td>
<td>...</td>
</tr>
<tr>
<td>3</td>
<td>1 D1</td>
<td>6312074</td>
<td>5</td>
<td>S</td>
<td>S</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>...</td>
</tr>
<tr>
<td>4</td>
<td>1 E1</td>
<td>6312075</td>
<td>5</td>
<td>S</td>
<td>S</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>...</td>
</tr>
<tr>
<td>5</td>
<td>1 F1</td>
<td>6312076</td>
<td>5</td>
<td>S</td>
<td>S</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>...</td>
</tr>
<tr>
<td>3021</td>
<td>33 F10</td>
<td>6312475</td>
<td>9</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>S</td>
<td>R</td>
<td>...</td>
</tr>
<tr>
<td>3022</td>
<td>33 G10</td>
<td>6312476</td>
<td>9</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>H</td>
<td>H</td>
<td>...</td>
</tr>
<tr>
<td>3023</td>
<td>33 H10</td>
<td>6312477</td>
<td>9</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>H</td>
<td>S</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>3024</td>
<td>33 A11</td>
<td>6312478</td>
<td>12</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
</tr>
<tr>
<td>3025</td>
<td>33 B11</td>
<td>6312479</td>
<td>13</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
</tr>
<tr>
<td>3026</td>
<td>33 C11</td>
<td>6312480</td>
<td>14</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
</tr>
<tr>
<td>4134</td>
<td>45 G11</td>
<td>6314743</td>
<td>176b</td>
<td>44</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>non-hp-1</td>
<td>...</td>
</tr>
<tr>
<td>4135</td>
<td>45 H11</td>
<td>6314744</td>
<td>176b</td>
<td>41</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>H</td>
<td>...</td>
</tr>
<tr>
<td>4136</td>
<td>45 A12</td>
<td>6314745</td>
<td>176b</td>
<td>42</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>H</td>
<td>...</td>
</tr>
<tr>
<td>4137</td>
<td>45 B12</td>
<td>6314746</td>
<td>176b</td>
<td>43</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>H</td>
<td>...</td>
</tr>
<tr>
<td>4138</td>
<td>45 C12</td>
<td>6314747</td>
<td>176b</td>
<td>45</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>H</td>
<td>...</td>
</tr>
</tbody>
</table>

Figure 29 Fixedness Results Output for SPLi4MKGTR

The above heatmap is the second output from the SPLi4MKGTR program as a .pdf showing an overview of the submitted project, reporting a level of fixedness for plants that belong to the same group.
4.1 Faster Results Reporting

The ability to return results to the breeder with a faster turnaround raises opportunities for the breeder to focus resources on growing and phenotyping plants to those lines with the most genetic potential. When the results turn around in sync with other tools, such as speed breeding, it can dramatically affect the productivity of moving the candidate breeding lines to the next commercialization stage in an even faster time. One important thing to note is that breeders will also be working on multiple projects simultaneously, so saving time on one gives them time to move forward on several projects. The process of manually handling the data is outlined in figure 31 to figure 36. Using this LIMS-less system to process plates eliminates the redundancies in the
workflow when exporting the information manually, saving time, especially when the amount of plates processed is considerable. When data files are handled manually across different excel files, there is a risk of copying data into the wrong columns or pasting it into the incorrect cells shifting the data, and having the wrong data matched to the incorrect sample. When this kind of mistake occurs, it has the opposite effect of helping breeders. It can cause delays in pushing a product forward since the data may indicate it is carrying a trait not really in that particular sample leading to longer times in fixing traits in inbred lines. With automation using MASLGenerator and SPLi4MKGTR, the data management process is robust and makes the mistakes in manual data management obsolete.
The process of manually exporting is very similar to the LIMS-less system, with one exception. Instead of exporting a complete project (all plate information) at once, it has to be done one plate reader file at a time, as indicated by the single green check on one plate file above.
Figure 32 Manual Data Handling Process Step 2

The user spends more time manually managing different excel to export data manually, first parsing out the export to read the marker designations outlined above.
A LIMS-LESS SYSTEM FOR GENOTYPING DATA IN MARKER-ASSISTED SELECTION

Separate Sorting excel file

Paste in Sorting template

Sort by Sample ID

Figure 34 Manual Data Handling Process Step 3

The user has to manage a separate template excel file storing a sorting pattern to regroup plates by their 96-well plate designation. This template is pasted into the exported data from KlusterCaller to sort it into the correct 96-well pattern shown above.

Copy and paste genotyping data into appropriate place in the plant list excel file

Figure 33 Manual Data Handling Process Step 4

The sorted data can then be copied and pasted at the particular spot; it needs to go into the plant list file for the exported marker.
The user must repeat the process shown in Figures 16.1-16.4 again to get the information for the next plate and then again for the following plate until all the plate files get analyzed. This task quickly grows tedious as the project sizes increase.
4.2 Example of Process

This section discusses an example project demonstrating the workflow of how these two programs fit into the laboratory process—beginning with (Table 8) a summary of a submitted project containing the data for eleven 96-well plates. This project includes 11 sample plates with 16 markers requested on each sample. The first step in processing this example is creating labels, so clicking on the MASLGenerator.exe launches the GUI interface (Figure 36). In the user input for Project Name, we will use Demo-Project for this example. Clicking on the browse button will open a file where a user can browse to the input file. Clicking on submit, you will see a labels.txt file containing the label information shown in the directory. A copy of this information gets pasted into the master excel file for the project to print on BarTender software. In BarTender software, the labels link from the excel file and barcodes for each plate created (Figure 37). The printed labels adhere to the 384-well plates shown in (Figure 38). The plates go through processing in the lab for extraction, PCR, scanning, and analysis in KlusterCaller to export the data to use in SPLi4MKGTR to join the plant list file to the genotyping data (Figure 39). Executing the program file prompts us to browse the genotyping file and the plant list .csv file and to type in the desired filename; for this example, we will continue with the name Demo-Project (Figure 40). Once submit is clicked, two files show up in the executable's directory, a Demo-Project_Results file, and a Demo_Project_Fixedness.pdf file, to report the results of the project (Table 9 and Figure 41).

Table 8 Summary of Demo-Project

This table shows an overview summary of the project used and a few lines from the plant list request form.

<table>
<thead>
<tr>
<th>Project Name</th>
<th>Number of plates</th>
<th>Number of Markers</th>
<th>384-well Plates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Demo-Project</td>
<td>11</td>
<td>16</td>
<td>42</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Plate #</th>
<th>Well</th>
<th>FEID</th>
<th>Stake</th>
<th>Plant</th>
<th>Markers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A01</td>
<td>omitted</td>
<td>74001</td>
<td>1</td>
<td>ACL,Bs2,Bs5,CCS,L1,SMXL1,Pc059514,Pc066896,Pc066921,PMR10275c,PMR19742,Pn1,pvrl,pvrl2,Tsw,</td>
</tr>
<tr>
<td>1</td>
<td>B01</td>
<td>omitted</td>
<td>74001</td>
<td>2</td>
<td>ACL,Bs2,Bs5,CCS,L1,SMXL1,Pc059514,Pc066896,Pc066921,PMR10275c,PMR19742,Pn1,pvrl,pvrl2,Tsw,</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>11</td>
<td>G12</td>
<td>none</td>
<td>empty</td>
<td>empty</td>
<td>ACL,Bs2,Bs5,CCS,L1,SMXL1,Pc059514,Pc066896,Pc066921,PMR10275c,PMR19742,Pn1,pvrl,pvrl2,Tsw,</td>
</tr>
<tr>
<td>11</td>
<td>H12</td>
<td>none</td>
<td>empty</td>
<td>empty</td>
<td>ACL,Bs2,Bs5,CCS,L1,SMXL1,Pc059514,Pc066896,Pc066921,PMR10275c,PMR19742,Pn1,pvrl,pvrl2,Tsw,</td>
</tr>
</tbody>
</table>
A LIMS-LESS SYSTEM FOR GENOTYPING DATA IN MARKER-ASSISTED SELECTION

Figure 36 MSLGenerator Output for Demo-Project

The GUI uses the input for this Demo-Project as the string to make the labels on the right. Only a subset of the labels is displayed to show the areas of the Labels.txt output, including the mix requirements.

Figure 37 Printing Labels for Demo-Project

The image above shows the BarTender Software interface for printing labels for the Demo-Project.
Figure 38 Labeling Plates for Demo-Project

The labels adhere to each 384-well plate processed for the Demo-Project.
Figure 39 Exporting Genotyping Data From KlusterCaller for Demo-Project

The image above shows the genotyping process by calling clusters, linking to the SNP labels, and exporting the results for use in SPLI4MKGTR.
The SPLi4MKGTR program takes the genotyping file generated by KlusterCaller and the PlantList file used to create the labels as inputs and outputs two files with your desired output file name, which in this case is Demo-Project_Results. The output files appear in the directory of the executable program. In this case, the Demo-Project_Results.xlsx and Demo-Project_Results_Fixedness.pdf are the output files.
<table>
<thead>
<tr>
<th>Plate #</th>
<th>Well</th>
<th>FEID</th>
<th>Stake</th>
<th>Plant</th>
<th>ACL</th>
<th>Bs2</th>
<th>Bs5</th>
<th>CCS</th>
<th>L1</th>
<th>...</th>
<th>Pun1</th>
<th>SMX-L1</th>
<th>Tsw</th>
<th>pvr1</th>
<th>pvr1-1-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>A01</td>
<td>omitted</td>
<td>74001</td>
<td>1</td>
<td>P</td>
<td>R</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
<td>NP</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>B01</td>
<td>omitted</td>
<td>74001</td>
<td>2</td>
<td>P</td>
<td>R</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
<td>NP</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>C01</td>
<td>omitted</td>
<td>74002</td>
<td>1</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>D01</td>
<td>omitted</td>
<td>74002</td>
<td>3</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>E01</td>
<td>omitted</td>
<td>74002</td>
<td>5</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>F01</td>
<td>omitted</td>
<td>74002</td>
<td>6</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>G01</td>
<td>omitted</td>
<td>74002</td>
<td>7</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>H01</td>
<td>omitted</td>
<td>74002</td>
<td>8</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>A02</td>
<td>omitted</td>
<td>74002</td>
<td>9</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>S</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>S</td>
<td>ND</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>B02</td>
<td>omitted</td>
<td>74003</td>
<td>10</td>
<td>Y</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>RH</td>
<td>S</td>
<td>ND</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>C02</td>
<td>omitted</td>
<td>74003</td>
<td>11</td>
<td>Y</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>RH</td>
<td>S</td>
<td>ND</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>11</td>
<td>B11</td>
<td>omitted</td>
<td>74247</td>
<td>1</td>
<td>Y</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>ND</td>
</tr>
<tr>
<td>1002</td>
<td>11</td>
<td>C11</td>
<td>omitted</td>
<td>74247</td>
<td>3</td>
<td>Y</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>ND</td>
</tr>
<tr>
<td>1003</td>
<td>11</td>
<td>D11</td>
<td>omitted</td>
<td>74248</td>
<td>4</td>
<td>H</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>S</td>
</tr>
<tr>
<td>1004</td>
<td>11</td>
<td>E11</td>
<td>omitted</td>
<td>74248</td>
<td>5</td>
<td>H</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>S</td>
</tr>
<tr>
<td>1005</td>
<td>11</td>
<td>F11</td>
<td>omitted</td>
<td>74248</td>
<td>6</td>
<td>H</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>S</td>
</tr>
<tr>
<td>1006</td>
<td>11</td>
<td>G11</td>
<td>omitted</td>
<td>74249</td>
<td>7</td>
<td>Y</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>S</td>
</tr>
<tr>
<td>1007</td>
<td>11</td>
<td>H11</td>
<td>none</td>
<td>74249</td>
<td>8</td>
<td>Y</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>S</td>
</tr>
<tr>
<td>1008</td>
<td>11</td>
<td>A12</td>
<td>none</td>
<td>74249</td>
<td>9</td>
<td>Y</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>S</td>
</tr>
<tr>
<td>1009</td>
<td>11</td>
<td>B12</td>
<td>none</td>
<td>74249</td>
<td>10</td>
<td>Y</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>S</td>
</tr>
<tr>
<td>1010</td>
<td>11</td>
<td>C12</td>
<td>none</td>
<td>74249</td>
<td>11</td>
<td>Y</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>S</td>
</tr>
<tr>
<td>1011</td>
<td>11</td>
<td>D12</td>
<td>none</td>
<td>74249</td>
<td>12</td>
<td>Y</td>
<td>S</td>
<td>S</td>
<td>R</td>
<td>R</td>
<td>...</td>
<td>P</td>
<td>S</td>
<td>R</td>
<td>S</td>
</tr>
</tbody>
</table>
4.3 Customization with programming

One of the benefits of using this alternative to a LIMS system is it leaves room for customization with programming. The raw code on git-hub can be used as a framework to expand the programs for other features. An example of this is from the original MASLGenerator I created a more detailed version that caters to collecting specific metadata from a modified submission sheet (Figure 41). In the MASLGenerator3 version, the program outputs other data, such as the number of markers run per plate and how many samples need processing for the project. This

Figure 41 Fixedness Results Output for Demo-Project

The .pdf output is a heatmap showing an overview of the project with the plants condensed by their group (Stake column). This output is a 5-page .pdf with results for the groups submitted, giving fixedness measures for the plants in each group for each marker.
version of MASLGenerator stores the information in an excel file called Project_Summary, collecting specifics for each project processed through the program (Tables 10 and 11).

Figure 42 Modified Plant List Submission Form

The image above is a modified plant list for submission to work with a further developed version MASLGenerator called MASLGenarator 3 to add the specific task of collecting more information about the processed projects each time the application runs.

Table 10 Project Summary output for MASLGenerator3

The table below is sample output for MASLGenerator3, which outputs separately from the labels a new line in a file named Project_Summary, which is updated after every time the application executes.
Table 13 Project Summary output for MASLGenerator3 – per Marker

The second tab in the Project_Summary output of MASLGenerator3 is the projects further broken down into what markers were requested and for what specific plates, calculating the number of samples run per marker.

<table>
<thead>
<tr>
<th>PID</th>
<th>Markers</th>
<th>Plates</th>
<th>96-well plates per marker</th>
<th># of samples per assay</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>2</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>3</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>4</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>5</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>6</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>7</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>8</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>9</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>10</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>11</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
<tr>
<td>12</td>
<td>P067</td>
<td>[1, 2, 3]</td>
<td>3</td>
<td>276</td>
</tr>
</tbody>
</table>

4.4 Debugging

Initially, the code handled all presented situations until a project was received where this issue arose. Some resolved issues in this project were when checking the length of the plant list files, where the amount of plates to process is divisible by 96 and 92 (Table 12). A secondary part of the conditional statement checking the file length fixed this issue. The secondary portion also checked if the number of unique values for the first 96 samples in the plate number column was equal to one. The thought behind this strategy was that if there are 92 samples, the number of unique values will be two, but if the first 96 samples are all on the same plate, the number of unique values will be one, indicating the empty wells in the plant list. A second debugging issue resolved was computing the mean for the heatmaps. In cases where samples contain ND values assigning a numerical is nonsense and skews the data values. A method of ignoring the ND values and only compiling the actual results was critical. The resolution was to use the mean function substituted for the manual calculation to ignore ND values.
4.5 Limitations

Although this LIMS-less system provides a quick way to compile results, there are still limitations. One limitation is the loading DNA pattern; the empty control wells must remain constant, or the sorting file will not work. Another limitation is the use of 384-well PCR plates. Although this provides high-throughput genotyping, there are options to scale up the PCR plate format to 1536-well plates for hyper-throughput genotyping (Figure 42). If a 1536-well plate format is needed, the structure of this project serves as a framework for adapting a higher-capacity plate.

Table 16 Debugging Program Crashes

One of the debugging issues was when the SPL4MKGTR crashed for an unknown reason. When verifying the PlantList length, particular lengths were divisible by 96 and 92 when checking if controls needed removing. Below is a short list of 96-well blocks that breeders could submit to a project that would cause the program to fail.

<table>
<thead>
<tr>
<th>Number of Samples</th>
<th>Divided by 92</th>
<th>Divided by 96</th>
</tr>
</thead>
<tbody>
<tr>
<td>2208</td>
<td>23</td>
<td>24</td>
</tr>
<tr>
<td>4416</td>
<td>46</td>
<td>48</td>
</tr>
<tr>
<td>6634</td>
<td>69</td>
<td>72</td>
</tr>
<tr>
<td>8832</td>
<td>92</td>
<td>96</td>
</tr>
<tr>
<td>11040</td>
<td>113</td>
<td>120</td>
</tr>
<tr>
<td>13248</td>
<td>138</td>
<td>144</td>
</tr>
<tr>
<td>15456</td>
<td>161</td>
<td>168</td>
</tr>
<tr>
<td>17664</td>
<td>184</td>
<td>192</td>
</tr>
<tr>
<td>19872</td>
<td>207</td>
<td>216</td>
</tr>
<tr>
<td>22080</td>
<td>216</td>
<td>230</td>
</tr>
<tr>
<td>24288</td>
<td>233</td>
<td>246</td>
</tr>
<tr>
<td>26496</td>
<td>276</td>
<td>288</td>
</tr>
<tr>
<td>28704</td>
<td>299</td>
<td>312</td>
</tr>
<tr>
<td>30912</td>
<td>322</td>
<td>336</td>
</tr>
<tr>
<td>33120</td>
<td>345</td>
<td>360</td>
</tr>
<tr>
<td>35328</td>
<td>368</td>
<td>384</td>
</tr>
<tr>
<td>37536</td>
<td>391</td>
<td>408</td>
</tr>
<tr>
<td>39744</td>
<td>414</td>
<td>432</td>
</tr>
<tr>
<td>41952</td>
<td>437</td>
<td>456</td>
</tr>
<tr>
<td>44160</td>
<td>460</td>
<td>480</td>
</tr>
</tbody>
</table>
4.6 Future
Some future improvements for this project include recognizing single plates in the SPLi4MKGTR program. When I first began developing this program, the microplate reader initially used had poorer optics than most new readers, so clusters had a wider spread of data points for some markers. One of the ways to overcome this scatter in the groups was to have repeat plates in each 384well plate allowing the clusters to look much nicer. A newer plate reader began being used later in this project with better optics providing tighter groups, so repeated plates are unnecessary. Adding the recognition of single plates with more markers in this program would help run four markers on a 384-well plate. Another future improvement on this project is addressing time issues linking markers. In the genotyping platform clicking through to link every marker to a plate can consume time as the number of plates grows, a possible improvement would be to connect the SNPs to the data in the post-processing in SPLi4MKGTR using the marker at the end of each label.

5 CONCLUSION
5.1 Conclusion
This LIMS-less system for reporting genotyping results provides an easy-to-use method for high-throughput genotyping labs to report results without needing a LIMS system. In plant
breeding programs, this system of the two programs created in this project allows for results to be written in less time on the scale of days allowing the breeder to carry on their work to continue improving cultivars. Incorporating new traits into cultivars for better adaptability to climate change is needed, and the population will continue to grow. Providing solutions to these problems is not a single fix but the collection of many advancements and improvements of tools like the one described in this project.

REFERENCES


A LIMS-LESS SYSTEM FOR GENOTYPING DATA IN MARKER-ASSISTED SELECTION


A LIMS-LESS SYSTEM FOR GENOTYPING DATA IN MARKER-ASSISTED SELECTION


