Project Narrative

1 Introduction

Plant breeding programs are critical for addressing global food production needs as the human population approaches nine billion. Genetic improvements in yield, stress tolerance, disease resistance and post-harvest quality are required in every crop. Plant breeders have produced steady gains in all these areas over the past century (Prohens, 2011). However, current rates of yield increases in many critical species such as wheat, rice, corn and soy remain insufficient to meet the future global demands (Ray et al., 2013). Innovations in the process of plant breeding can help to fill this gap.

We believe that increasing the use of multiple-trait breeding methods can broadly accelerate crop improvement programs. Measuring multiple traits at once and modeling their relationships can help increase the rate of gain in a single target trait and is critical for improving suites of traits at once. However, the vast majority of statistical models used in plant breeding today handle only a single trait (or at most a few traits) at a time. Multiple-trait data, on the other hand, is widely available. High-throughput phenotyping technologies — such as hyperspectral imaging and molecular profiling — are becoming accessible to many breeding programs and provide new types of data to inform selection decisions. But even traditional programs collect data on many traits in every field, and when multiplied across locations or years (and considering each trait in each field as a different trait), total trait numbers can easily reach into the hundreds. **Therefore the development of powerful, efficient, and usable statistical tools that can be applied to many traits at once would have an immediate impact on a wide range of breeding programs across the country and the world.** Our proposal aims to fill this gap. Specifically, we will:

1. Develop robust, high-powered statistical models for jointly predicting plant performance from high-dimensional phenotype and genotype data sets.

2. Implement the models in flexible, computationally efficient, and user-friendly open-source software

3. Design training materials and teaching modules to demonstrate our methods in diverse contexts.

Our approach builds on the statistical framework of the linear mixed effect model (Lynch and Walsh, 1998). Linear mixed models underlie virtually all statistical tools used in plant breeding, including the most widely used genomic prediction models (Bernardo, 2010). Linear mixed models are robust, interpretable, and relatively easy to use. However, when applied to data sets with large numbers of traits, they become extremely computationally demanding and tend to lose accuracy or even fail to converge. We will address these limitations by combining recent statistical and computational innovations into a single unified

package for plant breeders to use with both large (1,000's of traits) and small (10's of traits) breeding data sets. We believe that developing such statistical tools today — while high-throughput phenotyping is still in its infancy — can help guide both technological innovation and innovations in breeding methodology to take full advantages of these new types of data.

2 Rationale and Significance

This proposal addresses the **Program Area Priority: Plant Breeding Research A1141**, specifically: *selection theory*, *applied quantitative genetics*, *phenomics*, and *the incorporation of modeling in breeding*. We intend to introduce powerful, free, and user-friendly software to support public and private breeding programs in any crop species. Specifically, our software will assist breeders to i) efficiently select on multiple aspects of quality and performance at once, ii) address local and regional adaptation through modeling gene-environment interactions, and iii) incorporate phenomics data into genomics-enabled plant breeding systems.

Statistical models are necessary for plant breeders to effectively integrate data from genomic markers, pedigrees, environmental measurements, physiological traits, and multiple performance metrics into selection decisions. Predictive models additionally allow breeders to use incomplete information to make selection decisions faster and reduce the duration of breeding cycles. Marker assisted selection and Genomic Prediction models have been widely successful at increasing rates of genetic gain in many crop species. However, these methods are not able to leverage the large-scale phenomics data becoming available in many breeding programs driven by ongoing innovations in high-throughput phenotyping. **Our first objective** is to fill this gap by developing new quantitative genetics theory applicable to high-dimensional phenotype data. **Our second objective** is to implement this theory in efficient statistical software designed for plant breeders.

We have included letters of support from five public and private breeding programs that have offered to provide data to use as tutorial and training examples, or to test the performance of our models in real breeding environments.

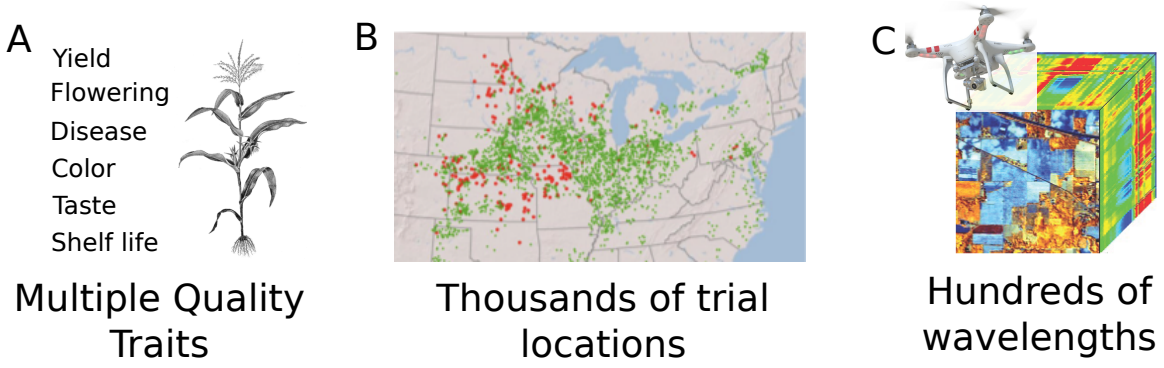
3 Background

3.1 Why is data on multiple traits useful for breeding?

Multi-trait data is both widely available and useful for breeding programs. To illustrate, we focus on three cases spanning a range of breeding goals.

A **Breeding for multiple traits at once.** Plant breeding programs are rarely focused only on improving a single trait (Figure 1A). Overall yield may be important, but without disease resistance, quality and appropriate phenology, a new cultivar will not be successful. While it is possible to select on each trait separately, this is generally inefficient if any of the traits are genetically correlated (Falconer and Mackay, 1996; Pollak et al., 1984). For example, if height is genetically correlated with yield, direct selection on yield will lead to a correlated response of height, which may not be desirable. The solution is to use the multivariate breeder's equation and calculate selection

Figure 1



indices for weighting all traits at once (Henderson and Quaas, 1976). This requires models that explicitly model the genetic and phenotypic correlations among traits.

B **Breeding for one trait across many locations.** Even if a breeder is only interested in a single trait (such as yield), the best genotypes may differ across locations. The change in relative performance of genotypes across locations is called gene-environment interactions (GEIs) (Bernardo, 2010; Granato et al., 2018). GEIs are a major challenge for plant breeders, and a major goal is predicting which genotypes will work best in different regions. One way to analyze GEIs is to treat the yield in each locations as a separate trait, rather than a single trait that changes across locations (Falconer and Mackay, 1996). By modeling the correlations in yield across locations, opportunities exist to increase selection intensity and accuracy in each environment by borrowing information on related genotypes grown in similar environments. For example, Figure 1B shows the distribution of thousands of commercial trial locations of the AQUAmax maize hybrids of Dupont Pioneer (Gaffney et al., 2015). See letter of support from Dr. Messina.

C **Leveraging secondary traits to improve genetic gains in a focal trait.** Even if a breeder is only interested in a single trait in a single environment, incorporating data from other "secondary" traits measured on the same genotypes can lead to faster genetic gains (Thompson and Meyer, 1986; Pszczola et al., 2013; Lado et al., 2018). When the focal trait has low heritability but genetically correlated secondary traits have higher heritability, the accuracy of selection on the focal trait can be improved by modeling both traits jointly (Thompson and Meyer, 1986). In addition, bigger advances are possible if the secondary trait(s) are also cheaper to measure, can be measured on more genotypes, or can be measured earlier in a growing season (or even out of season in controlled environments). In these cases, data on secondary traits can be used to both increase selection intensity and reduce cycle durations similarly to the use of genotypic data. Drone-based hyperspectral imaging can record reflectance data from hundreds of wavelengths at multiple times during the growing season (Rutkoski et al., 2016; Montesinos-López et al., 2017; Krause et al., 2019, e.g.), and can phenotype many more plots than is possible to measure for yield directly, which can be used to predict the performance of large populations of selection candidates (Figure 1C). See

letters of support from Drs. Hurr (vegetables), Crossa (wheat), Brummer (alfalfa), and Taylor (watercress).

In each case, models that estimate the correlations among traits are necessary to achieve the full breeding objectives in the most cost and time-effective way.

3.2 Data integration for cheaper, faster, and more accurate selection

Powerful and easy to use statistical modeling software is critical for optimizing plant breeding programs. Plant breeding can be an expensive, slow, and labor intensive process. Each season, a set of candidate genotypes must be grown in one or many trials, measured for a set of traits, and then the best genotypes selected to contribute to the next generation (Bernardo, 2010). The details of this process differ from one crop to the next — between annuals and perennials, self-fertilized crops and outcrossers, or plants deployed as inbreds vs hybrids vs outbred races — but the need for testing, measuring, and selecting is universal in breeding. This is encapsulated in the breeders equation: $\Delta G = \sigma^2 \times i \times r/L$ (Falconer and Mackay, 1996; Lynch and Walsh, 1998; Cobb et al., 2019) which relates genetic gains (ΔG) to the genetic variation among the genotypes (σ^2), the selection intensity (i), the accuracy of selection (r), and the number of cycles per year (L).

The earliest plant breeders made selections by simply looking for the best performing plants in a field. While sometimes successful, this phenotypic selection is often inefficient. Statistical models that integrate data from additional sources can increase selection accuracy, increase selection intensity and reduce cycle durations. The combined effect on the rate of genetic gain in breeding programs can be dramatic (Piepho et al., 2007; Crossa et al., 2010), particularly for low-heritability traits (Bernardo and Yu, 2007).

There are many reasons why the best performing plants may not actually be the best candidates: they may have ended up in particularly fertile areas of the field (microenvironmental variation); they may do particularly well in this field, but poorly in many others (gene-environment interactions); or they may carry alleles that do not combine well in crosses (gene-gene interactions) (Bernardo, 2010). Breeders can account for these possibilities by incorporating other types of data into their selections. They can look for spatial patterns to find microenvironmental variation. They can repeat their trials in multiple environments to check for gene-environment interactions. And they can use the performance of genetic relatives to assess gene-gene interactions. This data integration is performed intuitively by many breeders. But it can also be enhanced and made increasingly precise using statistical models. The most common method uses a linear mixed model to calculate Best Linear Unbiased Predictors (BLUPs) of the genetic values of genotypes or genetic effects of alleles by combining phenotype data with pedigree, environmental, geo-spatial, or other data sources (Piepho et al., 2007).

Today, massive amounts of genomic marker data are widely available and have supplemented or even replaced pedigree data in some programs (Crossa et al., 2017). While pedigrees describe the expected genetic relationships among lines, genomic markers more directly capture the *actual* patterns of relatedness due to Mendelian segregation or inaccuracies of the pedigree (Piepho et al., 2007). Marker Assisted Selection and Genomic Selection

are statistical techniques that use this additional genomic data to make BLUPs more accurate predictors of the true genetic value of candidate genotypes, which leads to even more genetic gains. However, incorporating data from genomic markers has another benefit: in some cases, actually measuring the phenotypes of every individual is not necessary if the phenotypes can be predicted directly from the genomic marker data (Crossa et al., 2017). If genomic data can be obtained more rapidly and cheaply than phenotype data, the intensity of selection can be increased by increasing the pool of candidate genotypes. The number of cycles per year can also be increased by growing and crossing plants in the offseason since their field phenotypes do not need to be measured. Genomic Prediction of genetic values can cut years out of a breeding program (Heffner et al., 2010), and has produced dramatic rates of genetic gain in a growing number of plant species (Crossa et al., 2017).

Phenotype data on the other hand remains limiting – field trials are expensive and will likely remain limited in the number of genotypes or locations that can be tested. However, the number of traits that can be measured in a single trial is increasing due to research and investments in high throughput phenotyping including proximal and remote sensors, molecular profiling, and many other new and developing technologies (Araus et al., 2018). As described above, **statistical models that integrate multi-variate phenotype data can increase the rate of gain in breeding programs in a similar way to genotype data, while also opening up new avenues for plant improvement.** This influx of phenotype data has yet to be met with similar advances in statistical models tailored to the goals of plant breeding programs, and existing models quickly run into computational limitations as the number of phenotypic observations per plant grows.

3.3 Statistical and computational challenges and proposed solutions

If it has been known for decades that multi-trait modeling can be useful for breeding, why aren't these techniques more widely applied? The extension from single-trait linear mixed models (LMMs) to multi-trait LMMs, which estimate and use trait correlations to calculate BLUPs or other predictors of genetic value, is conceptually straightforward and has been available for more than 40 years (Henderson and Quaas, 1976).

Three important bottlenecks are: 1) Multi-trait LMMs require large amounts of data (Thompson and Meyer, 1986). 2) Computational demands for fitting multi-trait LMMs increase very rapidly as the number of traits increases (Zhou and Stephens, 2014). 3) Multi-trait models are less intuitive to design, implement, and diagnose, and adequate training may not be available. **In this proposal, we will begin to address each of these limitations by developing new statistical theory, computational tools, and training modules that will make multi-trait LMMs more accessible for breeders across plant systems.** Here, we elaborate on each of these challenges and introduce our proposed solutions. Technical details will be left for the Approach section below.

3.3.1 Huge data requirements

Multi-trait models are necessarily complex with many parameters that must be estimated accurately from data. A common statistical maxim is that the number of observations should greatly exceed the number of parameters to be estimated (Huber, 2011). This is violated in

Figure 2: **The Curse of Dimensionality:** Without regularization, statistical models require many more observations than parameters. However, as the number of traits increases, the number of covariance parameters that must be estimated in a multi-trait linear mixed model increases even faster. Results shown assume data are available for each trait on 400 genotypes and the simplest LMM with one random effect and unstructured **G** and **R** covariance matrices

many typical datasets from breeding programs with more than a handful of traits, so naive applications of multi-trait LMMs are fragile, sensitive to data noise and overfitting, and exhibit convergence problems in statistical programs (Johnstone and Titterington, 2009). In the simplest LMM, calculating BLUPs to perform selection requires estimating two parameters: the *genetic variance* of the genotypes, and the *environmental variance*, or noise, of the observations. More complex models decompose the genetic variance into additive and non-additive components, or the environmental variance into separate effects of blocks, spatial variation, and residual error, each requiring an additional parameter to be estimated (Piepho et al., 2007). However, in multi-trait LMMs, not only do these variance parameters need to be estimated for *each trait*, but covariances among **all pairs of traits** for each component of variance need to be estimated, which we denote with the matrices **G** and **R** for genetic and residual covariance, respectively. The number of these covariance parameters increases quadratically with the number of traits in a model. To keep up with the number of model parameters, the number of tested genotypes in a field trial would need to be increased 25-fold to select jointly on five traits, 400-fold per location to model gene-environment interactions across 20 fields, or 1 million-fold to model the correlations among 1,000 hyperspectral bands and yield (Figure 2).

However, such sample sizes are not needed to take advantage of multi-trait data in breeding programs. The "large p / small n" setting with more model parameters (p) than observations (n) has been extensively studied in many fields including physics, economics, and biology (Bernardo et al., 2003; Johnstone and Titterington, 2009), leading to the development of robust tools for high-dimensional data that do not require excessively large sample sizes. Statistical techniques that include penalties or regularization on high-dimensional parameters make a "bet on sparsity" (Hastie et al., 2005), meaning that they focus on identifying and using only the strongest, most important signals in the data, and ignore weaker signals that are likely to cause only noise. In fact, regularization and sparsity-assuming models are already widely used in plant breeding for Genomic Prediction. Genomic data is also large p / small n; thousands to millions of genomic markers are evaluated in breeding programs for several crops, greatly exceeding the number of genotypes that can be measured. Examples of these regularized regression models for Genomic Prediction include rrBLUP (Endelman,

2011), Bayes A (Meuwissen et al., 2001), and several other Bayesian multiple regression methods, each of which make different assumptions about plausible numbers and effect sizes of causal genetic loci to increase their robustness for genetic value prediction (Meuwissen et al., 2001; Gianola, 2013). These models often work well even with very limited sample sizes.

Regularized statistical models for multi-trait prediction are much less developed than for single-trait models. For example, Meyer and Kirkpatrick (2010) proposed "bending" estimates of covariance matrices to improve robustness, and de Los Campos and Gianola (2007) and Dahl et al. (2016) use factor-analytic models to more efficiently estimate genetic covariances. While effective for small-moderate numbers of traits, these strategies are not effective for dozens of traits, and certainly not for hundreds or thousands because their regularization only applies to $\mathbf{G}$, but not $\mathbf{R}$ or $\mathbf{P}$, the residual or phenotypic covariance matrices. Stronger regularization is needed for all components of large multi-trait LMMs. Earlier we introduced a simple multi-trait LMM that could conceptually scale to thousands of traits based on a sparse-factor formulation called Bayesian Sparse Factor models for Genetics, or `BSFG` (Runcie and Mukherjee, 2013). The regularization scheme in our model was based on the idea that groups of traits tend to cluster in modules, and that these modules are likely to be co-regulated. We formulated our regularization using Bayesian priors (similar to the priors underlying Bayes A (Meuwissen et al., 2001)) that allowed the number and complexity of modules to be informed by the data, and demonstrated that the model could robustly estimate genetic covariances for hundreds of traits with realistic sample sizes. **Our proposal here is to expand the `BSFG` model into a full-fledged multi-trait LMM for use by plant breeders**, expanding the model to incorporate high-dimensional genotype and phenotype data and additional random effects to account for spatial variation, heterogeneous missing data, and non-normal observations.

3.3.2 High computational demands

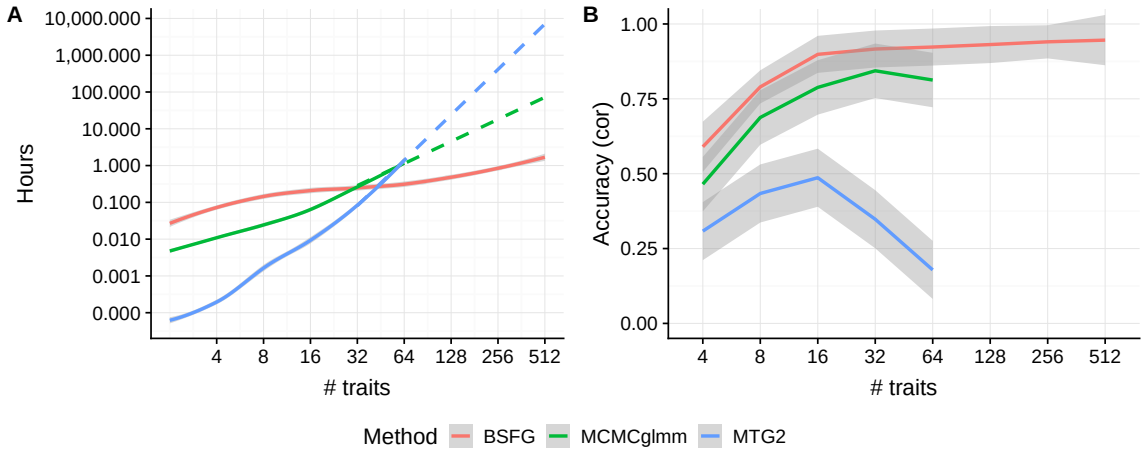
Multi-trait LMMs can be extremely computationally demanding. While needs for data increase quadratically with the number of traits (t), computational demands increase cubically because existing algorithms require repeatedly calculating the inverse of large covariance matrices — typically the $t \times t$ among-trait covariance matrix for Bayesian methods using MCMC (e.g. MCMCglmm (Hadfield, 2010)), or the full $nt \times nt$ among-observation covariance matrix for REML algorithms (e.g. `ASReml` (Gilmour, 2007)). Bayesian MCMC algorithms have to repeat these operations tens- to hundreds- of thousands of times because of poor mixing of most Gibbs sampler for LMMs (Gelman, 2006; Hadfield, 2010). These matrix inversions multiply the time required to fit a multi-trait LMM, leading to models that take days, weeks, or even years to fit with standard computers, and little opportunity exists for parallelizing the key operations. **Long wait times make it difficult for practitioners to test and optimize their models, and may preclude the rapid outputs needed for selection decisions.**

Computational issues in single-trait LMMs have been studied extensively, and more efficient algorithms have been developed for special cases including the use of sparse matrices (`lme4` (Bates et al., 2014), `MCMCglmm` (Hadfield, 2010)), eigendecomposition of covariance matrices (`EMMA` (Kang et al., 2008)), pre-caching intermediate calculations (`GEMMA` (Zhou and

Stephens, 2012), `GridLMM` (Runcie and Crawford, 2019) and improved Gibbs samplers for more efficient MCMC (Makalic and Schmidt, 2016). However, few programs have attempted to bring these algorithms to multi-trait LMMs, and those that do tend to still be limited to very small numbers of traits (`GEMMA` (Zhou and Stephens, 2014), see `phenix` for the closest example (Dahl et al., 2016).

We will combine these innovations with new algorithmic developments to greatly reduce the time required to fit multi-trait LMMs: 1) The sparse-factor formulation of `BSFG` virtually eliminates the need for large matrix inversions and permits parallelization of many calculations using efficient linear algebra C++ libraries. 2) Our discretization and pre-caching scheme for random effect covariance matrices can dramatically reduce the computational times for fitting single-trait LMMs with multiple random effects (GridLMM, (Runcie and Crawford, 2019)), and can be extended to multiple traits. 3) Re-ordering traits based on patterns of missing data can reduce the number of parameters to sample during MCMC. 4) Partially collapsed Gibbs samplers improve MCMC mixing, thereby reducing necessary run times (Runcie and Mukherjee, 2013). Our current prototype software can fit linear mixed models to datasets with 500 traits measured on each of 400 lines in less than half an hour on a typical laptop, and maintains estimation accuracy even for very large numbers of traits (Figure 3: compare the red line (`BSFG` prototype) to the REML program `MTG2` (blue) and the MCMC package `MCMCglmm` (green), which would take years or days, respectively on these data).

Figure 3: **Performance of `BSFG` prototype** We simulated data from 2-512 traits for 414 lines given known **G** and **R** matrices and used three highly efficient multi-trait mixed model programs: `MTG2` finds REML solutions and is written in fortran. `MCMCglmm` is written in C++ and is a Bayesian method. `BSFG` is our prototype software written in a combination of R and C++. **A)** Computational times to find the REML solution (`MTG2`) or collect a minimum effective sample size for **G** elements > 1000. Times for $t > 64$ were extrapolated for `MTG2` and `MCMCglmm`. **B)** Correlations between estimates of **G** and true matrix. To make the simulations realistic, we took a random set of 512 gene expression traits from 414 maize lines (Hirsch et al., 2014) and used `BSFG` to estimate **G** and **R** from these traits. We then simulated new traits using these estimated matrices as the true values. The relationship matrix is calculated from GBS data on the same lines.



3.3.3 Making statistical tools accessible to plant breeders

Widely used computational tools are powerful and effective, but also easy to learn, well-documented, and responsive to user needs (List et al., 2017). Our strategy for making multi-trait LMMs accessible to plant breeders includes the following components: 1) *Intuitive model formulation.* Linear mixed models themselves can feel abstract and difficult to understand for students and practitioners. Interpreting trait covariances and interactions on top of this in multi-trait models can easily be overwhelming. We have found that covariance matrices are difficult to interpret. One of the advantages of the sparse-factor formulation of `BSFG` is that rather than directly fitting a whole large covariance matrix among all traits, our model provides a mechanistic interpretation for why any two traits are correlated – trait correlations always have an underlying cause, whether it is a shared developmental pathway, or are limited by a shared resource, etc. `BSFG` aims to identify and model these underlying causes. This also facilitates more intuitive prior specification. 2) *Familiar implementation in widely used programming languages.* While the specification of multi-trait models is necessarily complex, and maintaining flexibility is critical for applications to diverse breeding programs, we aim to make the user interface of `BSFG` as familiar as possible. We have chosen the R language (R Core Team, 2019) to implement `BSFG`. R is widely used by biologists and increasingly used in agricultural applications. It is free and open-source meaning that it is useable by anyone anywhere in the world. We will aim to mimic the model specification syntax of popular LMM R programs such as `lme4` (Bates et al., 2015), `BGLR` (Pérez and de Los Campos, 2014), and `sommer` (Covarrubias-Pazaran, 2018) so that users can switch among tools easily. 3) *Active software maintenance and interactions with users.* We will host the software on Github `https://github.com/` and maintain an issue tracker to receive and process bug and feature requests. We will also use TravisCI `https://travis-ci.org/` to continuously test updates to ensure program stability. 4) *Statistical Education and training.* We will include multi-trait models in our annual training workshops at UC Davis.

3.4 Advantages of UC Davis

The University of California Davis campus is in the heart of the "Silicon Valley of seeds", with approximately 100 nearby seed and seed-related companies, many actively engaged in plant breeding activities. In addition, there are 15 faculty in the UC Davis Departments of Plant Sciences and Viticulture and Enology who run public-sector breeding programs, spanning local, national and international crops. The UC Davis Plant Breeding Center holds annual retreats including breeders from other University of California campuses, as well as running frequent courses through the Seed Biotechnology Center and the Plant Breeding Academy. A Graduate Academic Certificate in Plant Breeding is offered to graduate students in four graduate programs. This concentration of plant breeders and plant breeding students will facilitate active engagement of the PIs with active breeding programs, which will be important for testing and refining our methods and software, and developing training material.

4 Approach

4.1 **Objective 1**. Develop robust, high-powered statistical models for jointly predicting plant performance from high-dimensional phenotype and genotype data sets.

Our first objective is to develop a robust statistical framework for joint analyses of many traits. Our model will have the following characteristics: i) *Flexible*: adaptable to different classes of traits and experimental designs; ii) *Scalable*: maintain its effectiveness with large numbers of traits; iii) *Intuitive*: built on modules with straightforward interpretations.

As described above, PI Runcie previously introduced a model called `BSFG` which builds off the statistical framework of linear mixed models (LMMs) to fit genetic architectures to large numbers of traits (Runcie and Mukherjee, 2013). LMMs are robust, flexible and effective for a wide range of statistical problems, and much of the theoretical backbone of quantitative genetics and plant and animal breeding is based in LMMs (Bernardo and Yu, 2007; Lynch and Walsh, 1998). However, multi-trait LMMs are neither scalable nor intuitive because of the need to model covariances among all pairs of traits (Zhou and Stephens, 2014). Large covariance matrices have very many parameters, requiring massive amounts of data and computational resources, and their overall structure cannot be described by inspecting individual parameters; all relationships among parameters jointly determine the constraints that limit responses to multi-trait selection in breeding programs. We address these problems using sparse factors as a tool to reduce the complexity (i.e., the dimensionality) of multi-trait LMMs in a way that prioritizes estimating the dominant patterns in large datasets (Bernardo et al., 2003). Sparse factors effectively reduce a large multi-trait model into a set of parallel single-trait models by introducing the concept of "latent traits"; traits that were not directly observed on a plant, but can be inferred based on their effects on other traits that we do observe. Below, we first give a conceptual background on our original sparse factor model, and then provide mathematical details of our `BSFG` approach and the extensions we will make in the current project.

4.1.1 Conceptual model

Consider a single trait (say yield) measured on each of n plots of a field trial. We can represent these values in the $n \times 1$ vector $\mathbf{y}_1$. Now, say we measured a second trait (plant height) on the same set of n plots, and represent these values in the vector $\mathbf{y}_2$. Yield and height are often correlated traits. In the multi-trait LMM framework, we want to estimate this correlation, and then decompose it into the contribution from genetics (which can respond to selection), and from the environment (which we treat as noise that must be accounted for). We can describe these correlations by estimating the 2×2 genetic and residual covariance matrices $\mathbf{G}$ and $\mathbf{R}$. Each matrix has four parameters, but the off-diagonal parameters (the covariances between the two traits) within each matrix are constrained to be equal to each other and both less than the square root of the product of the two diagonal elements (the variances of each trait). These constraints make inferring $\mathbf{G}$ and $\mathbf{R}$ computationally difficult, and difficult to interpret directly.

However, from a developmental or physiological perspective, we could ask *why* these two

traits are correlated. If a plant flowers late, it will tend to be taller, and it may also have more leaves to collect light energy and so create more seed. Therefore, flowering time is an additional trait that may *cause* some of the correlation between yield and height. Better nitrogen uptake may also cause faster growth (increasing height) and increased yield, and thus also contribute to the correlation between our two focal traits (Mueller et al., 2019). If we had measured flowering times and nitrogen uptake rates for each plot, we could use path analysis to model how genetic variation in either flowering time or nitrogen uptake might explain genetic variation *and covariation* in both yield and plant height. Importantly, if flowering time and nitrogen uptake rates were the major causes of the yield-height correlations, then conditional on these values the remaining variation in yield and height would be approximately uncorrelated.

Flowering time is generally easy to measure, but nitrogen update rates are not. And it is unlikely in practice that all relevant traits during development could be measured to fully account for the correlations between yield and height, even with future developments in high-throughput phenotyping. However, we can still model these un-measured traits as causal factors in a factor model, a form of structural equation model where some of the traits are not observed (Hastie et al., 2012). The idea of a factor model is that *all correlations among observed traits* can be explained by variation in additional unobserved "latent traits", or *factors*. By including sufficient factors, and modeling both their variation and their causal paths to the observed traits, we can explain any pattern of covariance.

By itself, this factor model is actually larger and more complex than our original model with just two traits. However, using a factor model allows us to perform statistical regularization in a principled way by introducing a set of biologically reasonable assumptions:

- *A limited number of the latent traits control the majority of variation.* If these can be prioritized, the remainder can safely be ignored.
- *Each latent trait controls only a subset of the observed traits.* Large sets of traits tend to be arranged in modules such that traits inside a module are highly correlated, but their correlations with traits in other modules is relatively low.
- *Each latent trait is controlled by genetics and the environment.* Latent traits are just like any other trait (except they are not directly measured). Therefore it is reasonable to model their genetic architecture with the same terms as for any other trait.

These assumptions justify an approach that favors *sparsity* in the factors, meaning that we can prioritize estimating only a small set of the most important factors that are each responsible for controlling covariation in a small subset of all the traits. These *sparsity* assumptions are similar to those justifying the "Bayesian Alphabet" models for Genomic Prediction from high-dimensional genomic data. Further details and justification for these assumptions are provided in Runcie and Mukherjee (2013).

While our earlier work demonstrated that accurate genetic covariance estimates were possible with high-dimensional phenotype data, we propose here to expand `BSFG` to make it into a useful tool for breeders. We will focus on elaborating four aspects of the model:

- Develop a flexible observation level model to accommodate non-Gaussian responses and time-series data.

- Expand the models for individual traits (observed traits and latent factor traits) to include fixed effects and multiple random effects.
- Re-formulate the model with more intuitive prior distributions.
- Derive more efficient Gibbs updates for the MCMC algorithm used to fit the model.

4.1.2 Methods: Mathematical details

Our new model will be hierarchical with three layers. 1) An *observation layer* which relates the actual measurements on each unit (e.g. experimental plot or individual plant) to a set of modeled traits. This is similar to the link-function of a Generalized Linear (Mixed) Model. 2) A *Sparse Factor layer* which relates the observation layer parameters to a set of latent factor traits. 3) A *Genetic layer* which relates each factor trait and observation residual to genetic data (marker data, pedigree) and experimental factors (Figure 4).

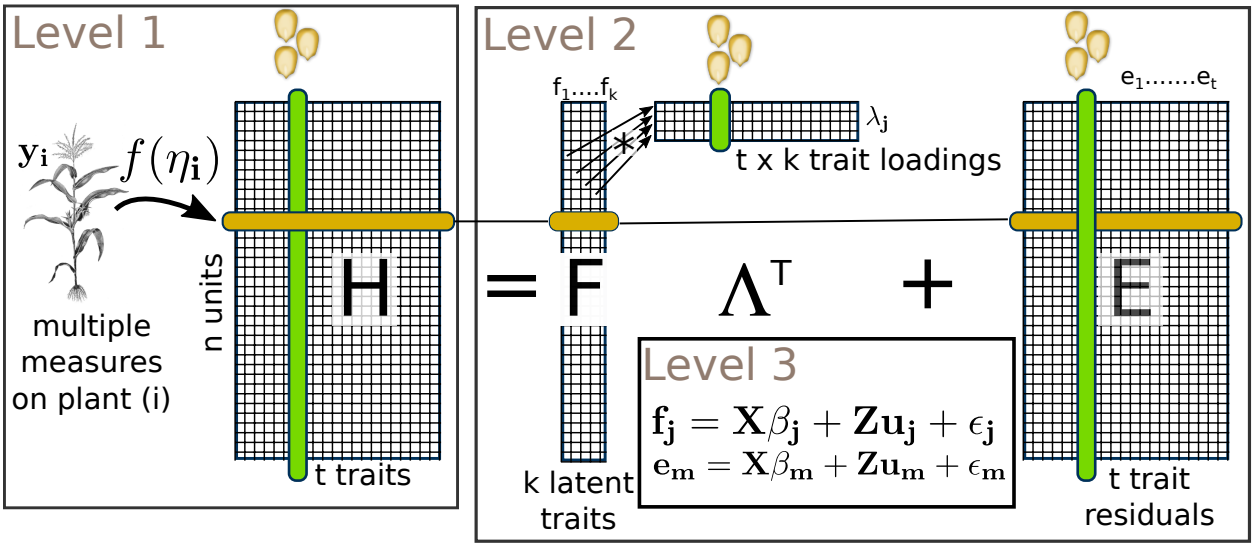


Figure 4: **Levels in the `BSFG` model:** 1) Raw data on each of the n plants ($\mathbf{y}_i$) is mapped to a common set of t traits ($\boldsymbol{\eta}_i$) by the link function $f()$. These traits are combined into a $n \times t$ trait matrix $\boldsymbol{H}$. 2) The covariance among the t traits in $\boldsymbol{H}$ is modeled using k factors $\mathbf{f}_1 \dots \mathbf{f}_k$, each of which controls variation in a subset of the original traits through the loadings matrix $\boldsymbol{\Lambda}$, with unexplained variation in each trait recorded as the residual vector $\mathbf{e}_m$. 3) Each of the factor traits $\mathbf{f}_j$ and residual vectors $\mathbf{e}_m$ are assumed to be independent and are modeled individually with single-trait linear mixed models. The same model structure is used for each of these traits. The green boxes show how the data on a single trait (e.g. seed number) across the individuals maps to the k factor loadings and one column of the $\mathbf{E}$ matrix. The orange boxes show how the trait values for one individual map to the k factor scores and a row of the $\mathbf{E}$ matrix.

Observation layer. Let $\mathbf{y}_i$ be a vector of measurements on unit i. We model these measurements as a function of t traits $\boldsymbol{\eta}_i$:

$$\mathbf{y}_i = f(\boldsymbol{\eta}_i, \boldsymbol{\theta}_y)$$

where $\boldsymbol{\theta}_y$ is a vector of observation-level noise parameters. The function $f()$ is flexible and accommodates a wide range of possible models including: i) *Identity*: $f(\boldsymbol{\eta}_i) = \boldsymbol{\eta}_i$ (original `BSFG` model) where the t traits are measured directly; ii) *Regression models*: $f(\boldsymbol{\eta}_i, \sigma_y^2) = \mathrm{N}(\mathbf{X}_y\boldsymbol{\eta}_i, \sigma_y^2)$ with $\mathbf{X}_y$ a design matrix for known covariates such as time or space, including non-parametric splines, and the t traits are the coefficients of this model. This is useful for time-series models; iii) *Non-Gaussian error*, where $f()$ is any of the common link-distributions of General Linear Models. More generally, $f()$ could be any complex non-linear function of a set of parameters $\boldsymbol{\eta}_i$, including crop growth models (Messina et al., 2018), where we consider these model parameters as traits. However this extension is beyond the scope of this proposal.

Sparse Factor Layer. Given the set of vectors $\{\boldsymbol{\eta}_i\}$ of traits for each unit, we can stack them as rows into an $n \times t$ matrix $\boldsymbol{H}$. Each column of $\boldsymbol{H}$ describes the variation in one of the traits across the n units. In the simplest case when $f()$ is the identity, $\boldsymbol{H} = \mathbf{Y}$, the matrix of t measurements on each of the n units. Our goal is to model the genetic variation and covariation among these t traits to enable joint selection on all of them or indirect selection from one subset to another. Because the number of traits t can be large (including $t >> n$), we use sparse factors to model the covariance:

$$\boldsymbol{H} = \mathbf{F}\boldsymbol{\Lambda}^T + \mathbf{E}$$

where $\mathbf{F}$ is a $n \times k$ matrix of "latent traits", i.e. unobserved characteristics of each unit that explain the covariances among the observed traits $\boldsymbol{H}$, $\boldsymbol{\Lambda}$ is a $t \times k$ matrix of trait loadings, i.e. causal paths between these latent traits and the observed traits (similar to PCA loadings), and $\mathbf{E}$ is a $n \times t$ matrix of residuals, i.e. variation in the traits that cannot be explained by the k latent traits. While the traits in $\boldsymbol{H}$ (columns) may be highly correlated, we assume that the residuals in $\mathbf{E}$ are uncorrelated; all correlation in $\boldsymbol{H}$ is accounted for by $\mathbf{F}$ and $\boldsymbol{\Lambda}$.

The key at this stage is the loadings matrix $\boldsymbol{\Lambda}$. Two general problems in factor models of this form are: i) choosing the number of factors k and ii) lack of identifiability of the factors. In our original model (Runcie and Mukherjee, 2013) we incorporated a recent innovation called *Infinite factor models* which circumvents the need to specify k by instead modeling the decrease in importance of $\boldsymbol{\lambda}_j$ relative to $\boldsymbol{\lambda}_{j-1}$ (Bhattacharya and Dunson, 2011). The most important factor, $\boldsymbol{\lambda}_1$ has the largest values and so explains the most covariation in the data. The remaining factors are arranged in decreasing order of importance until for $j^* > k$, the values are small enough that they can be safely ignored. This is convenient computationally because rather than running multiple models with different values of k, the user simply chooses k^* larger than should be needed, and then can truncate $\boldsymbol{\Lambda}$ to only the set of estimated factors that each contain at least two large values. For identifiability, and to induce sparsity in each factor, we originally used independent t-distributions as priors for each element of $\boldsymbol{\Lambda}$ (equivalent to the prior used in BayesA). We propose to switch to the horseshoe prior for these parameters (Carvalho et al., 2010). In addition to better shrinkage properties than the t-distribution prior, this has the advantage that we can specify a prior for the decline in importance of each consecutive factor in terms of the relative decrease in the effective number of traits controlled by each factor, a more specific quantity than simply "factor importance" in our original model (Piironen and Vehtari, 2017).

Genetic layer. The second advantage of the factor model formulation for a multi-trait LMM, besides providing a principled framework for regularization of covariance matrix es-

timates, is that conditional on $\mathbf{\Lambda}$, the factor traits $\mathbf{f}_j$ (each column of $\mathbf{F}$) and observed trait residuals $\mathbf{e}_m$ (each column of $\mathbf{E}$) are all assumed to be uncorrelated. Therefore, we can model each of these $k+t$ traits independently with single-trait LMMs:

$$\mathbf{f}_j = \mathbf{X}\boldsymbol{\beta}_{f_j} + \sum_{l=1}^{L} \mathbf{Z}_l \mathbf{u}_{f_j} + \boldsymbol{\epsilon}_{f_j} \qquad j = 1, \ldots, k$$

$$\mathbf{e}_m = \mathbf{X}\boldsymbol{\beta}_{e_m} + \sum_{l=1}^{L} \mathbf{Z}_l \mathbf{u}_{e_m} + \boldsymbol{\epsilon}_{e_m} \qquad m = 1, \ldots, t$$

where $\mathbf{X}$ is a design matrix for fixed effects (such as experimental design factors or marker genotypes) with coefficients $\boldsymbol{\beta}_*$ for each trait, $\mathbf{Z}_l$ are design matrices for L random effects with random effects $\mathbf{u}_*$ for each trait and *a priori* covariance matrices $\mathbf{K}_l$, and $\boldsymbol{\epsilon}_*$ are vectors of independent errors for each trait. This is a very general LMM – fixed effects are optional, and random effects such as genetic relatedness, pedigrees, or spatial effects can be specified either through $\mathbf{Z}$ or $\mathbf{K}$ – and more comprehensive than the models permitted in our original `BSFG` model. The parallel structure for both the "observed" traits and the latent factor traits facilitates model specification and is logical if we consider each of them a true characteristic of a plant or plot. Importantly, regardless of the appropriateness of the sparsity assumptions in any particular data, this is a full multi-trait LMM - given large enough k, the factors can model *any* pattern of covariance among the traits.

As in any LMM, the key parameters to estimate for each trait are the *variance components*, i.e. the weighting factors for the random effects and the residuals. We adopt the parameterization we introduced in `GridLMM` (Runcie and Crawford, 2019) in which each variance component is expressed as a proportion of the total variance (which we term h^2_l to denote the similarity to the concept of heritability), and we use as a prior distribution a discrete set of values on the L-dimensional simplex spanning all possible values of h^2. This parameterization allow for more intuitive prior specification for multiple random effect models than is typical in Bayesian multi-trait LMMs (Runcie and Crawford, 2019), and facilitates our efficient Gibbs sampling scheme which we describe below.

4.1.3 Methods: MCMC algorithm

Bayesian LMMs are typically fit by Gibbs samplers, a Markov Chain Monte Carlo (MCMC) algorithm that works by repeatedly drawing samples from the conditional posteriors of different subsets of the model parameters, conditional on current values for all others (van Dyk and Park, 2011). Gibbs samplers are relatively easy to construct for LMMs, and can be more computationally efficient than maximum likelihood (ML or REML) algorithms for multi-trait LMMs because they do not require directly calculating the likelihood of the full model (which involves inverting a $nt \times nt$ matrix). At the same time, they provide estimates of the full posterior distributions of all parameters, rather than just a point estimate. However, Gibbs samplers for LMMs typical exhibit poor "mixing", meaning that consecutive samples are not independent, and so typically 10s of thousands to millions of samples are required to adequately characterize the posterior distribution (Hadfield, 2010). And, when the number of traits is large, each iteration of existing Gibbs sampler algorithms is slow

because inverses of $t \times t$ matrices need to be calculated to draw samplers for the genetic and residual covariance matrices $\mathbf{G}$ and $\mathbf{R}$. The combination of slow sampling steps and poor mixing means that existing Gibbs sampling algorithms for mult-trait LMMs are not practical for large numbers of traits.

By specifying our multi-trait LMM as a sparse factor model with discrete priors on the variance component proportions as described above, we can design a modified MCMC algorithm with both faster per-iteration times and better mixing properties. Per-iteration computational times are reduced relative to the Gibbs samplers of `MCMCglmm` or `MTM` because direct sampling of $\mathbf{G}$ and $\mathbf{R}$ are not required. Instead, we sample the matrices $\mathbf{F}$ and $\mathbf{\Lambda}$, each of which require inverting only $k \times k$ matrices, regardless of the number of traits in the model. This allows our algorithm to scale to very large numbers of traits without loss of efficiency. In addition, since $\mathbf{\Lambda}$ accounts for all covariation among the traits, conditional on an estimate $\widehat{\mathbf{\Lambda}}$, all columns of $\mathbf{F}$ and $\mathbf{E}$ are independent in the model, so sampling draws for each column can be made independently. This permits parallelization across the multiple cores available on most modern CPU. We will additionally investigate whether parallelization can be increased using graphics processor units (Rupp et al., 2016).

Depending on the form of the observation layer model for a particular dataset, we may be able to use a Gibbs sampler (Identity or Regression models), but if not, will evaluate simple Metropolis Hastings updates, slice sampling (Hadfield, 2010), and Hamiltonian Monte Carlo (Neal et al., 2011). Missing observations in $\mathbf{Y}$ or $\boldsymbol{H}$ provide another potential bottleneck for sampling efficiency. It is straightforward in a Gibbs sampler to treat missing observations as unknown parameters and draw samples conditional on all other parameters (Hadfield, 2010). However this leads to strong autocorrelation and poor mixing. For multi-trait LMMs that are written as a single column (with traits stacked), it is more efficient to simply drop all rows with missing data. However, algorithms like `BSFG` that treat multi-trait LMMs as matrices would require dropping whole rows (i.e. all traits for a particular unit). As a compromise, when considerable amounts of data are missing, we will break up the data matrix into a set of smaller (nearly) complete-data matrices by grouping traits together with similar patterns of missing data across observations. This maintains the computational efficiency of matrix algebra, and the sampling efficiency of avoiding imputation during the Gibbs sampler.

To reduce the requirement for millions of MCMC iterations, we will incorporate recent innovations in Gibbs sampler architectures that increase mixing rates by "Partial Collapsing", i.e. ordering the steps of the Gibbs algorithm so that certain nuisance parameters can be marginalized over during sampling steps, rather than conditioned on (van Dyk and Park, 2011). One of the causes of poor mixing in Gibbs samplers for LMMs is the strong dependence between a random effects vector $\mathbf{u}$ and its variance component σ^2_u. If the current draw of σ^2_u is small, this will force the next draw of $\mathbf{u}$ to have values close to zero. In the next iteration, σ^2_u will again tend to be small because all values in $\mathbf{u}$ will be small. This causes the MCMC chain to get stuck, and require many iterations to explore other ranges of the parameter space. Instead, if we can "collapse" the Gibbs sampler by marginalizing over $\mathbf{u}$, we can directly draw values for σ^2_u unconditionally (with respect to $\mathbf{u}$), and then draw values for $\mathbf{u}$ in a second step conditional on σ^2_u. The marginalization operation in the first step breaks the dependence for σ^2_u, so mixing is improved. However, drawing samples from σ^2_u directly requires inverting a $n \times n$ covariance matrix, a computationally expensive operation. Here, our discrete prior on the variance component proportions becomes useful.

Given the current estimate of the total phenotypic variance σ^2, only a discrete set of values of σ^2_u is possible, corresponding to one of the possible h^2 values. Each h^2 value corresponds to a single covariance matrix; therefore we can pre-calculate all possible covariance matrices and their inverses prior to beginning the sampler and re-use these pre-calculated matrices for each iteration. In our GridLMM program for single-trait LMMs, we have found that a discrete grid with only 10-20 possible values for each h^2_l parameter works well for sample sizes up to several thousand individuals, so for models with $\approx 1-6$ random effects, the total number of $n \times n$ matrix operations will be less than a few hundred.

Together, these algorithmic innovations should make parameter inference of very large multi-trait LMMs computationally tractable.

4.2 **Obective 2**. Implement the models in flexible, computationally efficient, and user-friendly open-source packages for the R and Julia languages

Statistical models are not useful for breeders unless they are implemented in robust, intuitive, and accessible software packages. Our second objective is to develop an `R` package for running the `BSFG` model. `R` is a well-established language for statistical computing with $> 15,000$ freely-available, open source packages covering a very wide range of statistical topics. It is an interpreted language with well developed visualization tools for data and model exploration. However, significant computational tasks need to be written in a compiled language like C++ and then called from R.

4.2.1 Methods

Our implementation will have the following features:

- *Documentation:* We will use *RStudio* to develop the `R` implementation of `BSFG`. *RStudio* has integrated functions for helping write `Roxygen` documentation for functions and vignettes. Every exported function will be documented.

- *Interface:* The `R` implementation of `BSFG` will be constructed as an `R` package hosted on Github and CRAN. Widely used `R` packages for fitting single- and multi-trait LMMs include `lme4`, `MCMCglmm`, `BGLR`, `sommer` and `brms` (Bürkner, 2017). Although the model syntax differs slightly among these packages, most use `R`'s formula syntax to specify the response, fixed effects and random effects. Of these, the `lme4` syntax is the most straightforward, the `sommer` and `brms` syntaxes are more expressive, while `BGLR` allows nearly complete flexibility at the cost of requiring more up-front work from the users. In our `GridLMM` package, we will adopt a syntax most similar to `lme4` with additional arguments to specify *a priori* covariance matrices (similar to `lme4qtl` (Ziyatdinov et al., 2018)). For prior specification, we guide the user through the priors for each model component: the observation-level parameters, the loadings matrix $\mathbf{\Lambda}$, the fixed effects $\mathbf{X}$, and the variance component proportions, following similar parameterizations and syntax as used in `MCMCglmm` where possible.

- *Architecture:* While high-level functions will be written in native `R`, all functions that require expensive computations will be written in `C++` using the `Eigen` linear algebra

libraries and linked to `R` using `Rcpp` (Eddelbuettel and Balamuta, 2017) and `RcppEigen` (Bates and Eddelbuettel, 2013). Parallelization in `C++` code will be implemented using the `RcppParallel` package (Allaire et al., 2018). GPU computation will be implemented using `gpuR` (Rupp et al., 2016). Because `BSFG` models can contain hundreds of thousands of parameters, we will implement architectures for storing posterior samples on the disk rather than in memory.

- *Diagnostics:* We will export posterior samples in data structures compatible with the MCMC diagnostics tools available in the `Rstan` package (Stan Development Team, 2018).

- *Helper functions:* Most breeding activities require more than simply specifying and running a model. Genomic prediction models must be trained and tested using cross validation (Meuwissen et al., 2001). Cross validation involves subsetting data into non-overlapping training and testing partitions, fitting a model on the training partition, and then evaluating its prediction accuracy by comparing prediction in the testing partition to the observed data. Cross-validation is generally repeated multiple times to assess the robustness of estimated prediction accuracies. We will package this repeated partitioning, model fitting, and evaluation into a single function call and parallelized it over a computer cluster for streamlined use.

4.3 **Objective 3**. Develop training material with public datasets

4.3.1 Methods:

To demonstrate the utility of high-dimensional data for plant breeding programs and the power of our `BSFG` software, we will develop several example tutorials based on publicly available data from real breeding programs. We outline two of these examples here:

- Multi-trait / multi-environment maize breeding trials from the *Genomes2Field Initiative* `https://www.genomes2fields.org/`. The 2014 field season of the Genomes2Field project included 24 field trials in 13 states, with a core set of 11 traits measured in each field. While a total of 790 hybrids were grown across the trials, only ≈ 200 were grown in each field. If we treat each of the core traits in each field as a unique trait, and subset to traits with > 50 observations, the complete trait matrix has have 242 traits for 790 hybids, but only 26% of the cells were actually observed. In our first tutorial, we will show how genetic values for the remaining 74% of the trait:hybrid combinations can be imputed, resulting on a complete data matrix, and how the accuracy of the imputation can be estimated by cross-validation.

- Hyperspectral imaging from the CIMMYT wheat breeding program (Montesinos-López et al., 2017). 1170 wheat lines were evaluated in five environments and imaged on nine dates using 250 discrete narrow wavelengths with a hyperspectral camera on a plane. Earlier analyses used Bayesian functional regression to derive predictions of grain yield. In our second tutorial, we will implement these functional regressions in the *observation-level* model of `BSFG` to model correlations among the functional parameters

and grain yield, and develop appropriate cross-validation schemes for evaluating genetic prediction accuracy (Runcie and Cheng, 2019). The postdoctoral scholar will spend one month in years 2 and 3 at CIMMYT working with the breeders and biostatisticians to compare `BSFG` to existing methods (See letter of support from Dr. Crossa)

We will use these examples and the `BSFG` package to add a topic on advanced multi-trait modeling to our annual short course on "Modern Programming in Genomic Prediction" `http://qtl.rocks/workshops.html`, led by Co-PI Cheng. This course reaches ≈ 30 students per year and provides instruction on computational aspects of running genomic prediction models.

4.4 Project timetable

Year	Develop theory	Create R package	Develop vignettes	Host workshop
Year 1	X	X		
Year 2	X	X	X	X
Year 3		X	X	X

4.5 Expected Outcomes and Limitations

The principle outcomes of this project will be: 1) Powerful and accessible open-source software to help plant breeders implement multi-trait Genomic Prediction jointly on large numbers of traits, and 2) Practical demonstrations of the potential benefits to breeding programs in accuracy and time from such multi-trait methods. The software tools that we develop will be unique in their scope - no existing tool can handle hundreds or thousands of traits when fitting full multi-trait Linear Mixed Models, particularly with non-Gaussian data, incomplete observations, and multiple random effects.

Our preliminary data demonstrates that `BSFG` can improve prediction accuracy in real and simulated data in reasonable amounts of computational time. However, as with any method, our design choices for `BSFG` do impose limitations that may be important in real data sets. The ability of `BSFG` to efficiently scale to large numbers of traits relies on our assumptions of sparsity in the covariance structures. While there is evidence from many systems that large sets of traits nearly always cluster in modules, this may not be true in some datasets, causing `BSFG` to lose some efficiency. Also, `BSFG` is fundamentally a *linear* model – all relationships among traits are assumed to be linear, which clearly can be violated in practice. Linear models have a long history in quantitative genetics and frequently work well even when the underlying system is non-linear, but extensions to this approach that can handle non-linear relationships among traits is an area for future research. Finally, while the multi-trait LMM architecture underlying `BSFG` is flexible, our discrete prior on variance components becomes unwieldy with more than a few random effects. Extending to larger numbers of (low-rank) random effects may be possible, but would require additional theoretical and computational innovations that are beyond the scope of this proposal.