Advanced topics in quantitative genetics

AX101317

http://alenxav.wix.com/home

MACHINE LEARNING, PHENOMICS and BIG DATA

Rationale for advanced topics in quantitative genetics

- Quantitative genetics in plant breeding has been reshaped in the post-genomic era. The tools and data are different, but the decisions to be taken remain the same: what to select, what crosses to make, how to maintain or increase genetic gains, etc.

- **Machine learning** and **data mining** are important for handling, and make good use of, information from novel *OMICs* and upcoming breeding technologies.

- In addition to that, knowing how to manage multiple traits with variable genetic architecture is crucial to deal with **phenomics**.

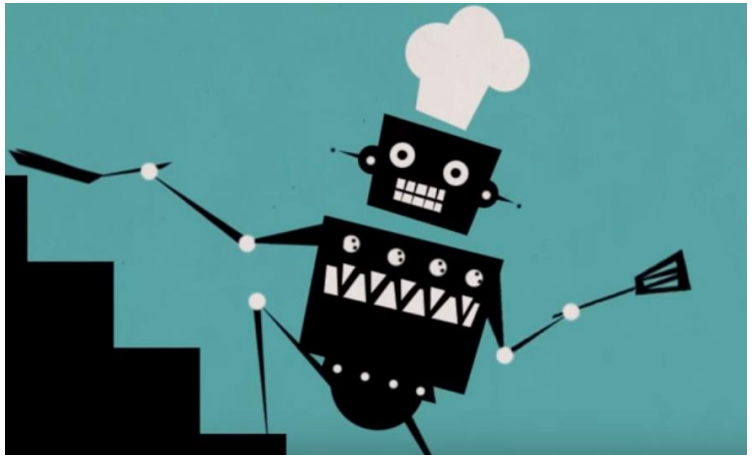
- With the abundance of data from omics and the computation burden from machine learning, it is important to get familiarized with **big data** strategies, which may come in handy to overcome computational limitations.

1. Machine learning

- Rationale
- Overfitting and Complexity-Variance tradeoff
- Parsimony (Occam's razor)
- Hierarchical principle
- ML Methods

- Machine learning is a major component of artificial intelligence (AI). ML is concerning with capturing specific patterns from data, often with the purpose of de-noising, classification and predictions for decision making.

- **COOL FACT**: **No machine can be optimally efficient in more than one task**.
  - The example below postulates that one machine that climb stairs and make pancakes will be less efficient than two machines exclusively focus on climbing start and making pancakes, respectively.



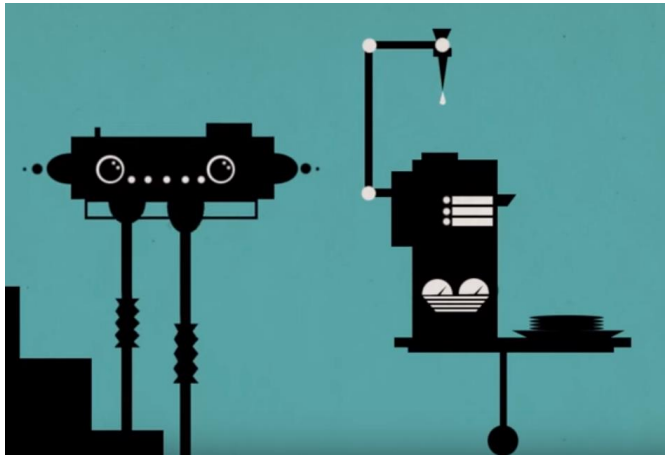
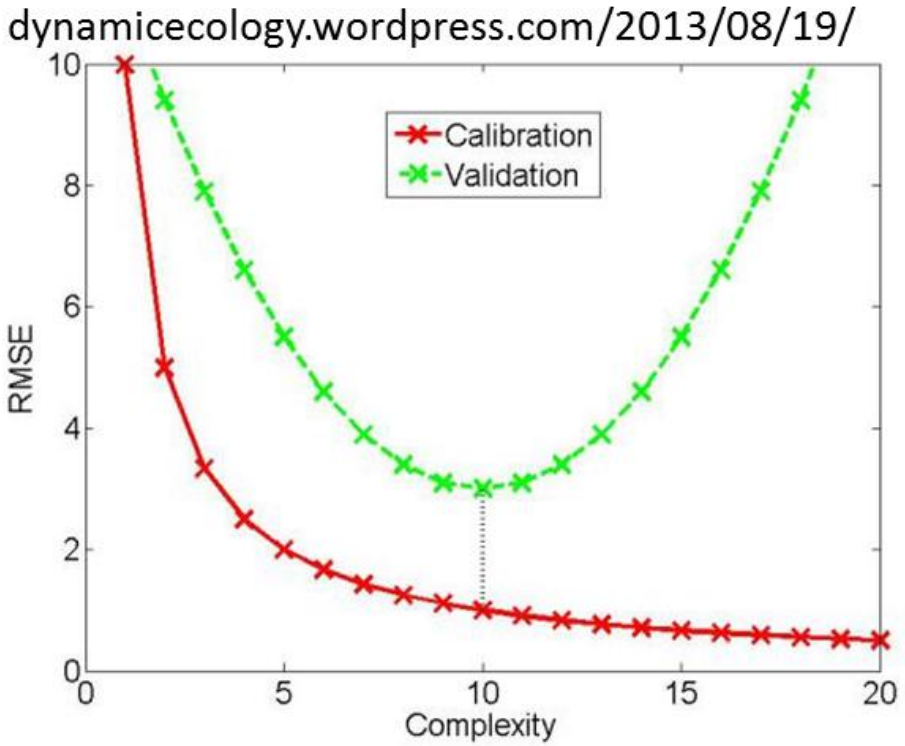


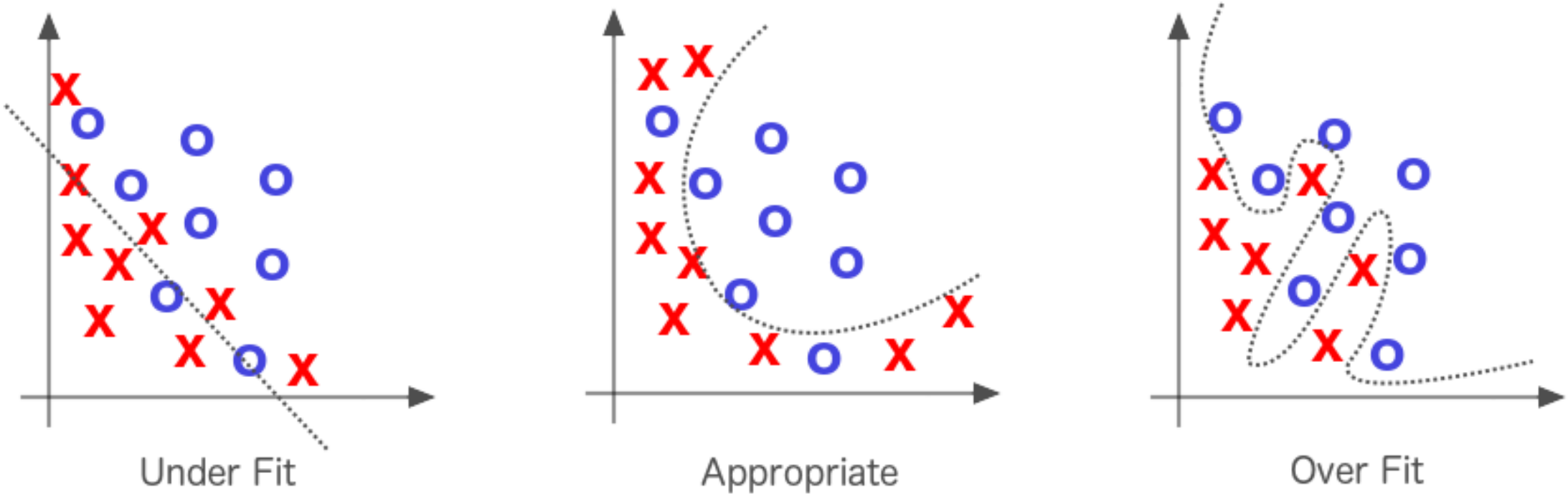
Figure source: https://www.youtube.com/watch?v=MPR3o6Hnf2g

Overfitting and Complexity-Variance tradeoff

- More variables will always increase R^2
- Overfitting is a good rationale for variable selection
- Goodness-of-fit is usually a poor indicator of quality
  - Exception - Some kernel method overfit but yield prediction
- Be suspicious of models that are too good
- Regularized method deal with overfitness
  - Example – shrinkage of a BLUP

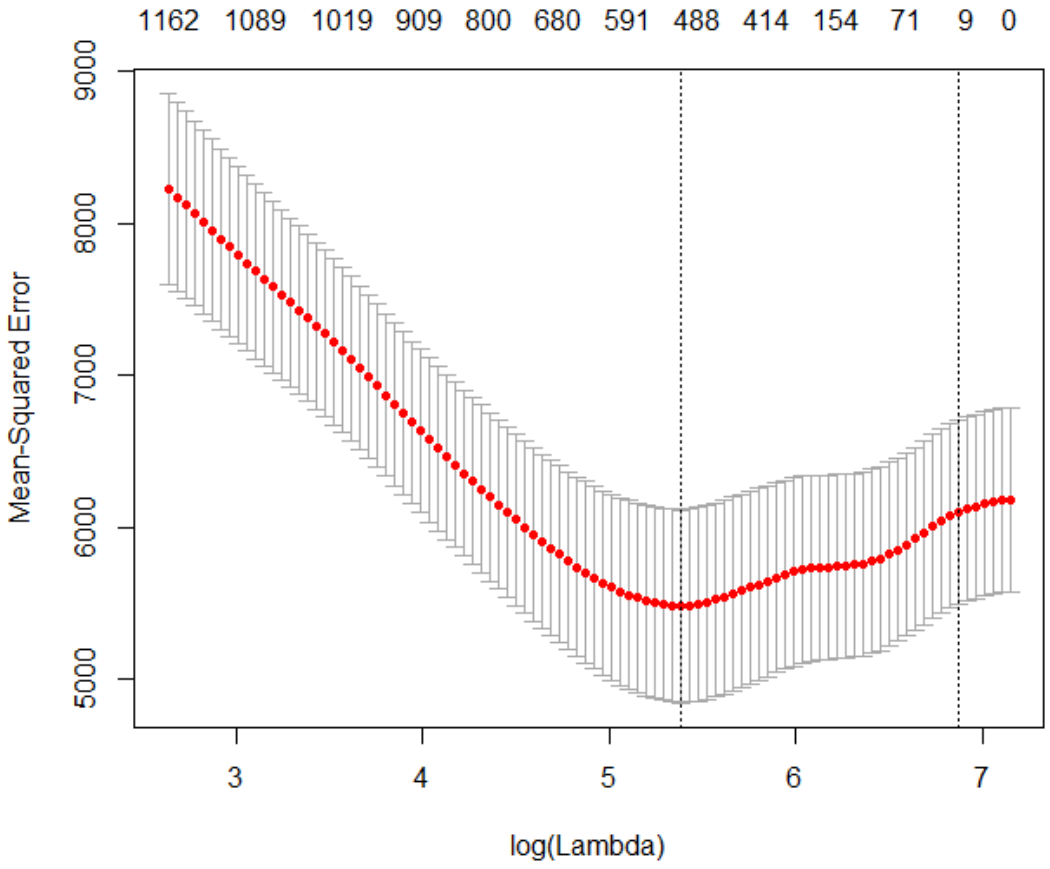


kNN example of overfitting



https://www.safaribooksonline.com/library/view/deep-learning/9781491924570/ch01.html

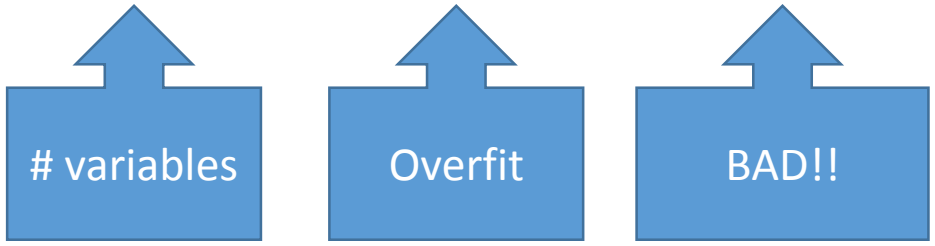
L_1L_2 example of overfitting



$$\mathbf{b} = \frac{\mathbf{x}'\mathbf{y} - \boldsymbol{\lambda}(\boldsymbol{\alpha})}{\mathbf{x}'\mathbf{x} + \boldsymbol{\lambda}(\mathbf{1} - \boldsymbol{\alpha})} = \frac{\mathbf{x}'\mathbf{y} - \boldsymbol{\lambda_1}}{\mathbf{x}'\mathbf{x} + \boldsymbol{\lambda_2}}$$

$$\lambda \uparrow = Fitness\ (R^2)\ \downarrow = Number\ of\ SNPs \downarrow = Shrinkage \uparrow$$

$$\uparrow Complexity = \downarrow Variance = \downarrow Prediction$$



Parsimony

- **Occam's razor** – A simpler explanation is better than a complex one
  - The less terms you have in your model, the better
  - An attempt to comprise most information with the least amount of factors

- **Example (model for genetic values). Check the two models:**
  1) **Yield = Block + Location + Year + Genotype + (Genotype x Year x Location) + …**
  2) **Yield = Block + Genotype**

- **NOTE 1** - In model 2), block is already a unique combination of location, year and experimental set, so it will account for all environmental parameters.
- **NOTE 2** – In model 1), higher order terms (eg. G x Y x L) do not contribute to the estimation of genetic values, rather it capture residuals – it create saturations

Occam's Razor: No more things should be presumed to exist than are absolutely necessary, i.e., the fewer assumptions an explanation of a phenomenon depends on, the better the explanation.

(William of Occam)

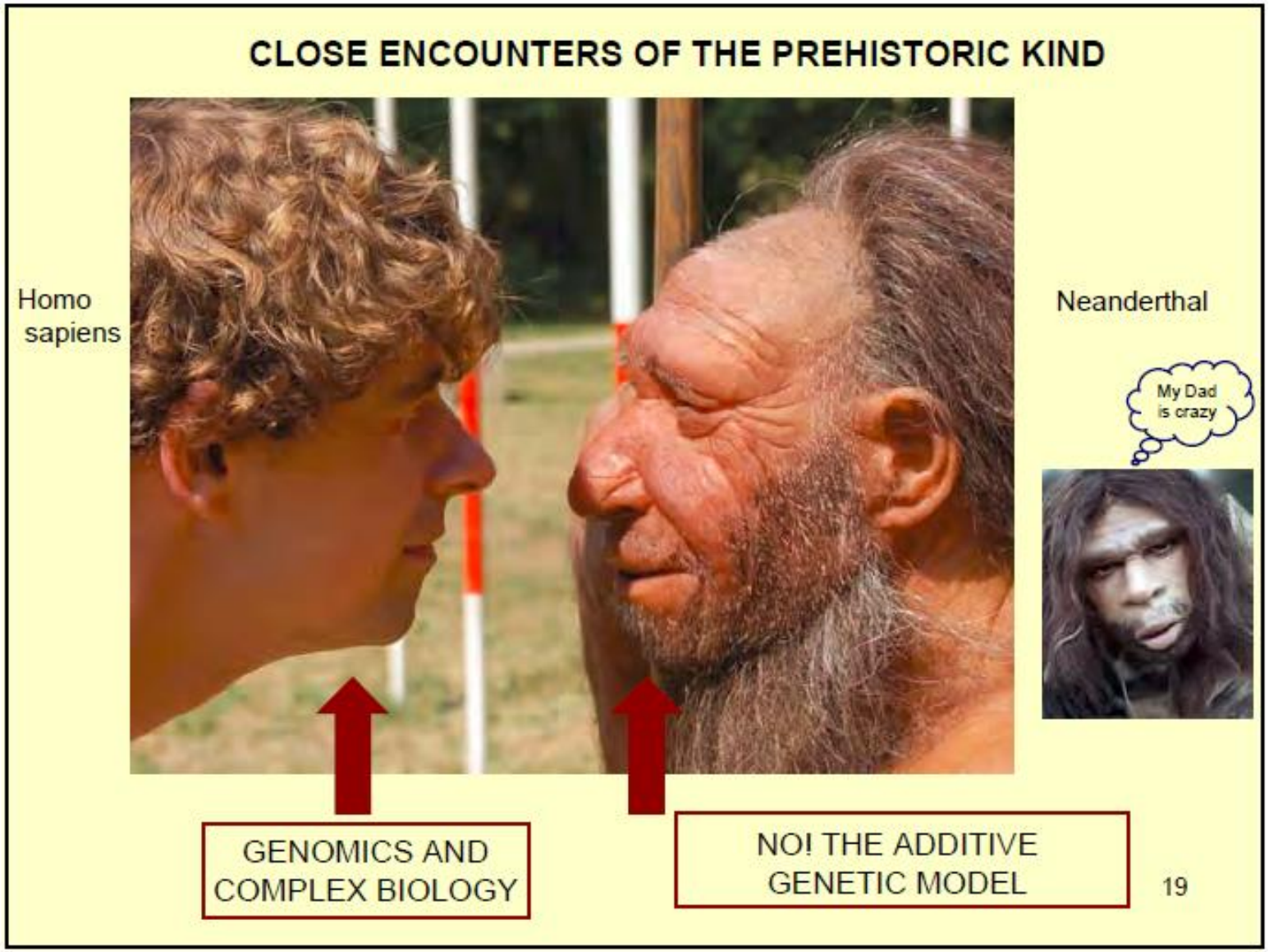
izquotes.com

Hierarchical principle

- **CLAIM**: Lower order effects more important than higher order effects
- Effects of same order equally important

(FOR STATERS: POWER IS AN ISSUE TO DETECT SIGNAL OF INTERACTIONS)

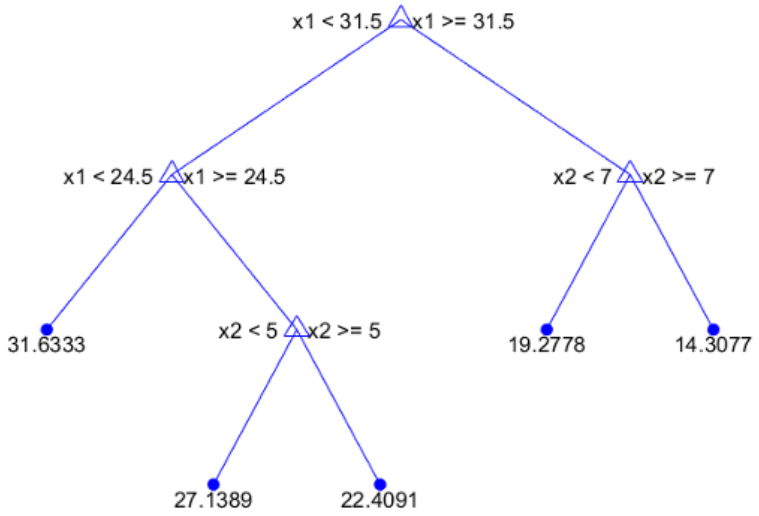
- Consider the nature of the variable (continuous or cathegorical)
- This principle makes one wonder about the relevance of **Epistasis** and **GxE**
- Higher order terms are good to **run out of degrees of freedom**

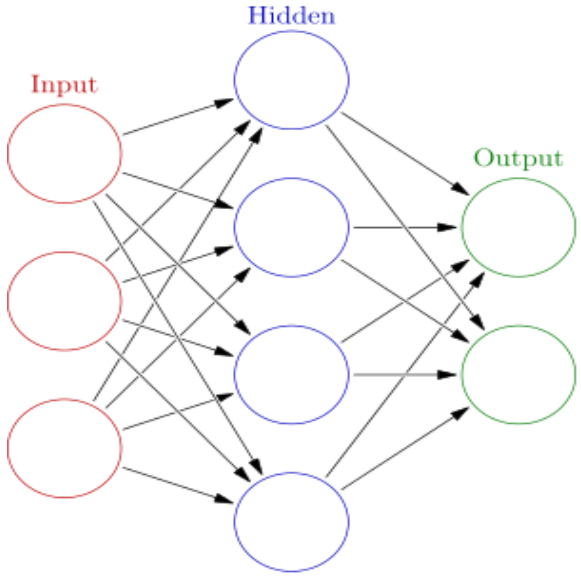


Slide by Daniel Gianola, Valencia Sep. 2010

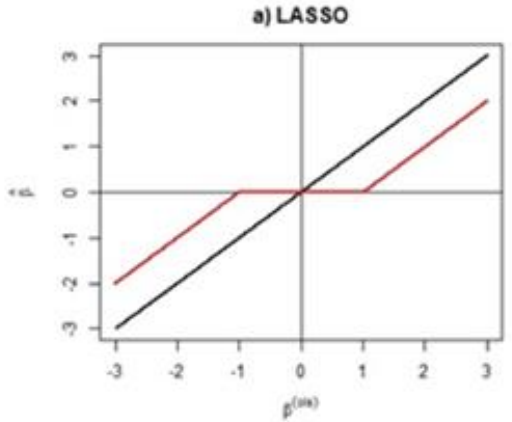
Machine Learning techniques

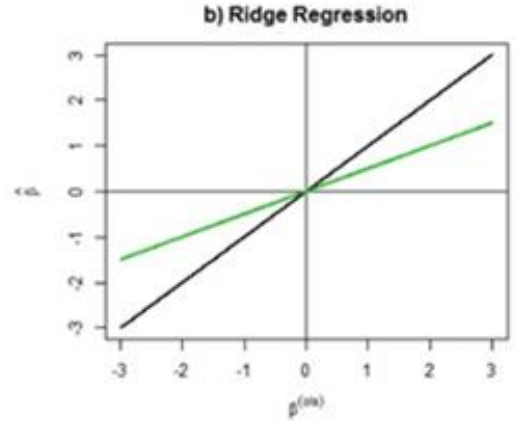
- Trees, kernels, neural nets and regression

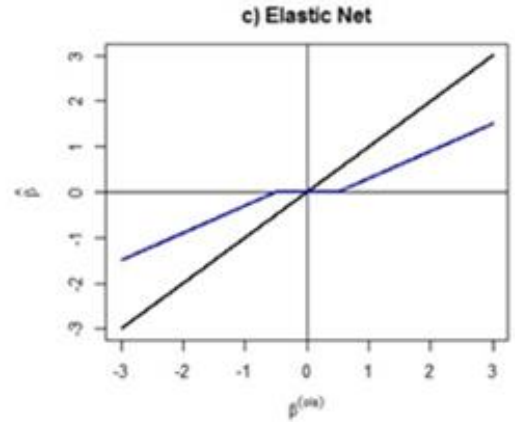


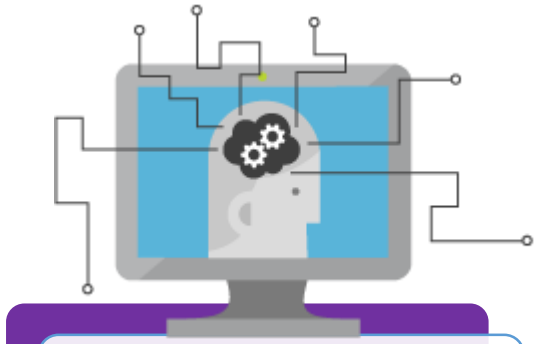


$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i K(\mathbf{x}, \mathbf{x}_i)$$









Machine Learning in Plant Breeding

- Supervised Learning
  - Prediction
    - omic-based prediction
    - Yield models
  - Classification
    - Market classes, Heterotic group
- Unsupervised Learning
  - Clustering
    - Select target environments
    - Germplasm Structure
  - Associations
    - Trade-off analysis

ML supervised methods

- Parametric
  - Regression
    - Least square
    - Least absolute
    - Stepwise regression
  - Penalized linear models
    - LASSO
    - Ridge regression
    - Elastic-net and Bridge
  - Stochastic methods
    - Stochastic search variable selection
    - Reversible jump Markov Chain Monte Carlo
    - Hierarchical Bayesian methods
  - Orthogonal transform
    - Principal components regression
    - Partial least square (PLS) regression
- Non-parametric
  - Tree
    - Tree regression
    - Random forest (Bagging)
    - Boosting
  - Kernels
    - Support vector regression
    - Reproducing Kernel Hilbert Spaces
    - Kernel ridge regression
    - k Nearest Neighbors
  - Neural networks
    - Feedforwards
    - Recurrent (forward and backward)
    - Radial basis function (kernel based)
    - Regularized NN

Kernel parametrization

An easy way to get around a large number of markers

https://graphene.limited/services--technologies/physics-of-triggering/hilbert-measurements-in/index.html

Regressing epistatic terms as ridge

$$y = \mu + \sum x_i \alpha_A + \sum x_i x_j \alpha_{AA} + \sum x_i x_j x_k \alpha_{AAA} + \cdots + \varepsilon$$

Huge number of parameters!! Below, an example of merely $p = 50$ SNPs

$$A = 50, \qquad AA = \frac{50 * 49}{2 * 1} = 1225, \qquad AAA = \frac{50 * 49 * 48}{3 * 2 * 1} = 19600$$

Regressing epistatic terms as kernels

$$y = \mu + \mathrm{u_A} + \mathrm{u_{AA}} + \mathrm{u_{AAA}} + \cdots + \varepsilon$$

$$\mathrm{u_A} \sim \mathrm{N}\left(0, \mathrm{K_A}\sigma^2_{\mathrm{AA}}\right), \qquad \mathrm{K_A} = \mathrm{MM}'c$$
$$\mathrm{u_{AA}} \sim \mathrm{N}\left(0, \mathrm{K_{AA}}\sigma^2_{\mathrm{AA}}\right), \qquad \mathrm{K_{AA}} = (\mathrm{MM}' \circ \mathrm{MM}')c$$
$$\mathrm{u_{AAA}} \sim \mathrm{N}\left(0, \mathrm{K_{AAA}}\sigma^2_{\mathrm{AAA}}\right), \qquad \mathrm{K_{AAA}} = (\mathrm{MM}' \circ \mathrm{MM}' \circ \mathrm{MM}')c$$

c = constant to normalize the trace

Ensemble learning (to fix poor learners)

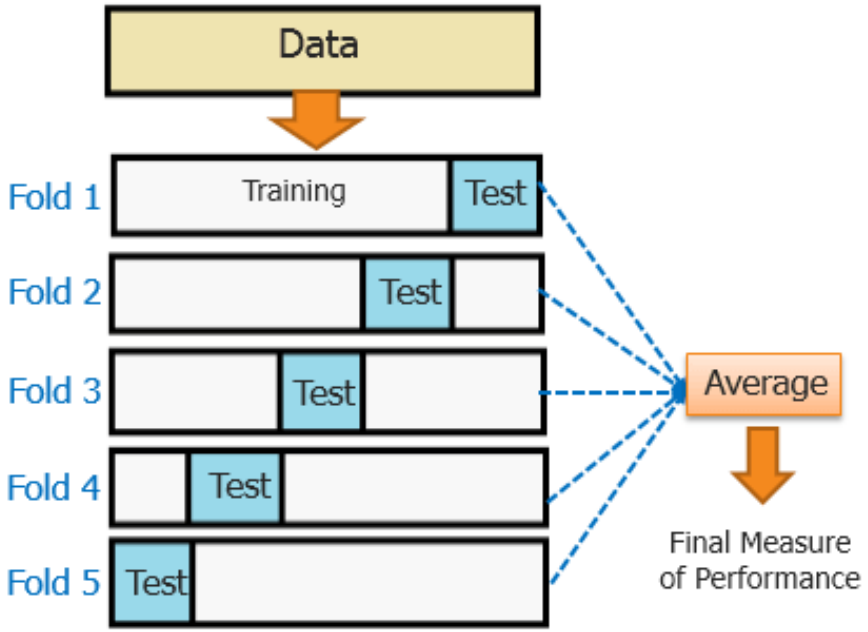
- Bagging (Bootstrapping aggregation)
  - Fit the model several times with different subsets of the data
  - Combine the models

- Boosting
  - Model is refitted continuously reweighting data points
  - More weight is given to the ‘outliers’

- Stacking
  - Models constructed from bootstrapped data
  - Residuals are modeled by the next round of bootstrapped data

Cross-validation

A practical way to assess your pipeline



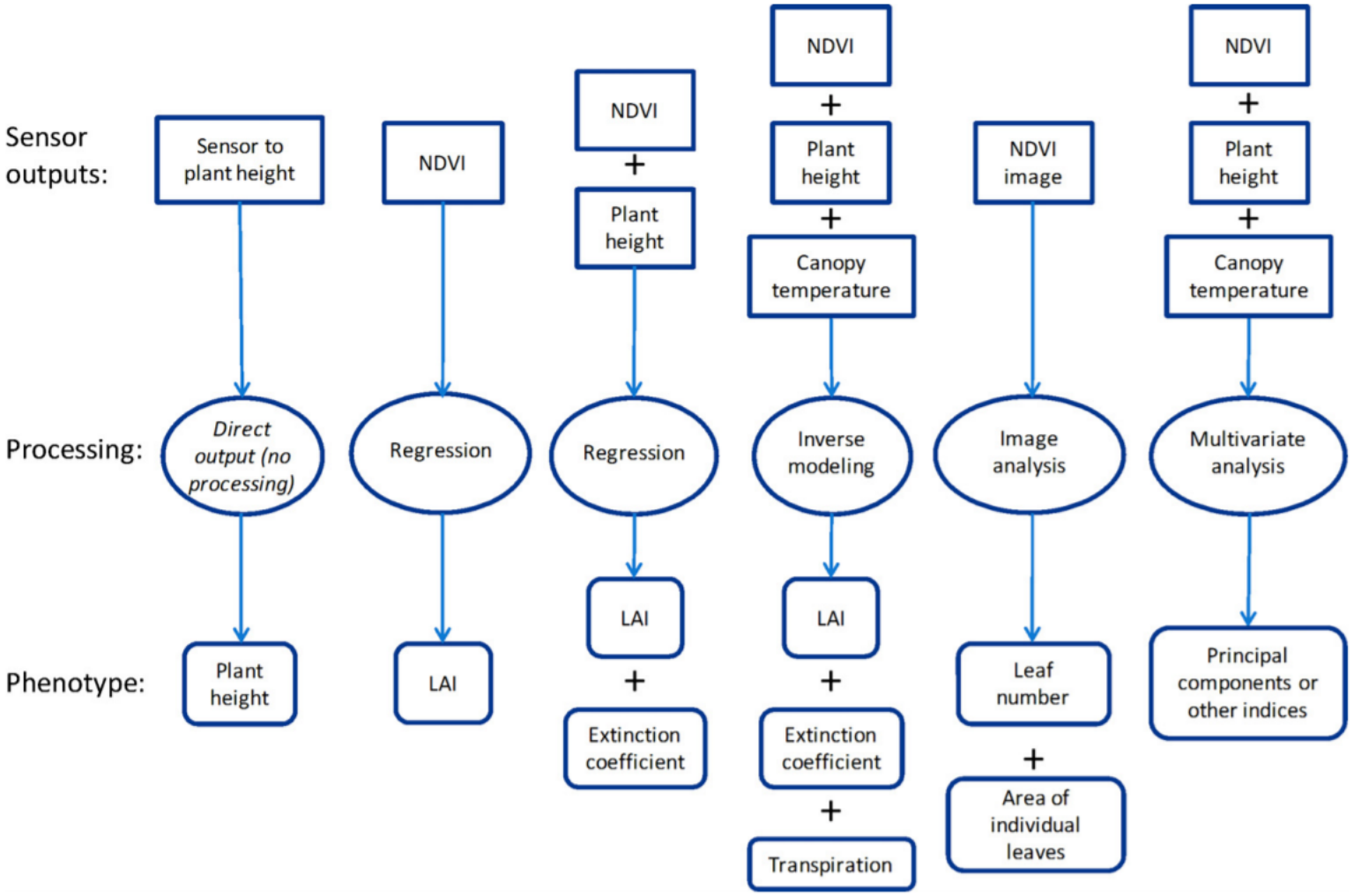
http://tomaszkacmajor.pl/index.php/2016/05/01/svm-model-selection2/

CV is used to evaluate methodologies

- Criteria
  - Correlation: Pearson or Spearman
  - % Correct selections
  - Mean squared prediction error
  - Bias
- Scheme
  - K-Fold Cross validation (within a given set)
  - Leave-one-out cross validation
    - Leave a family out, predict performance of lines from that family
    - Leave an environment out, predict performance of lines in that environment

2. Phenomic workflow

Number and quality of phenotypes increase



J.W. White et al. / Field Crops Research 133 (2012) 101–112

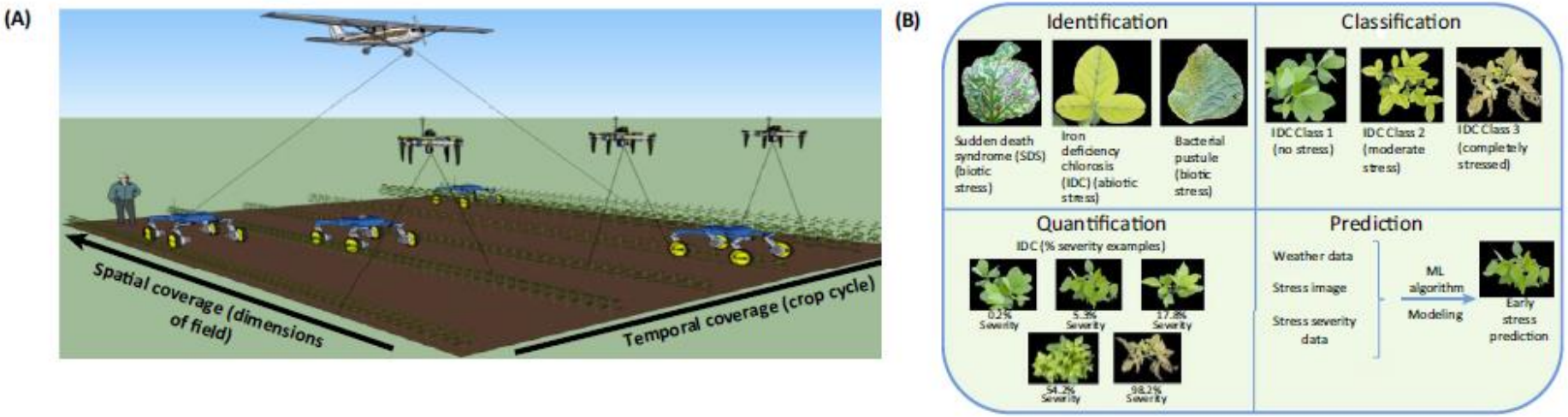
Fig. 6. Examples of possible paths of data analysis whereby field measurements are processed to provide more biologically meaningful data. Field data usually would be recorded as time series, allowing estimation of growth or developmental rates.

Pipeline complexity (highly computational)

Key Figure

Trends in Plant Science, February 2016, Vol. 21, No. 2 111

Machine Learning (ML) Tools for High-Throughput Stress Phenotyping



Phenomic data is usually tight to genomics

Review

Trends in Plant Science December 2011, Vol. 16, No. 12

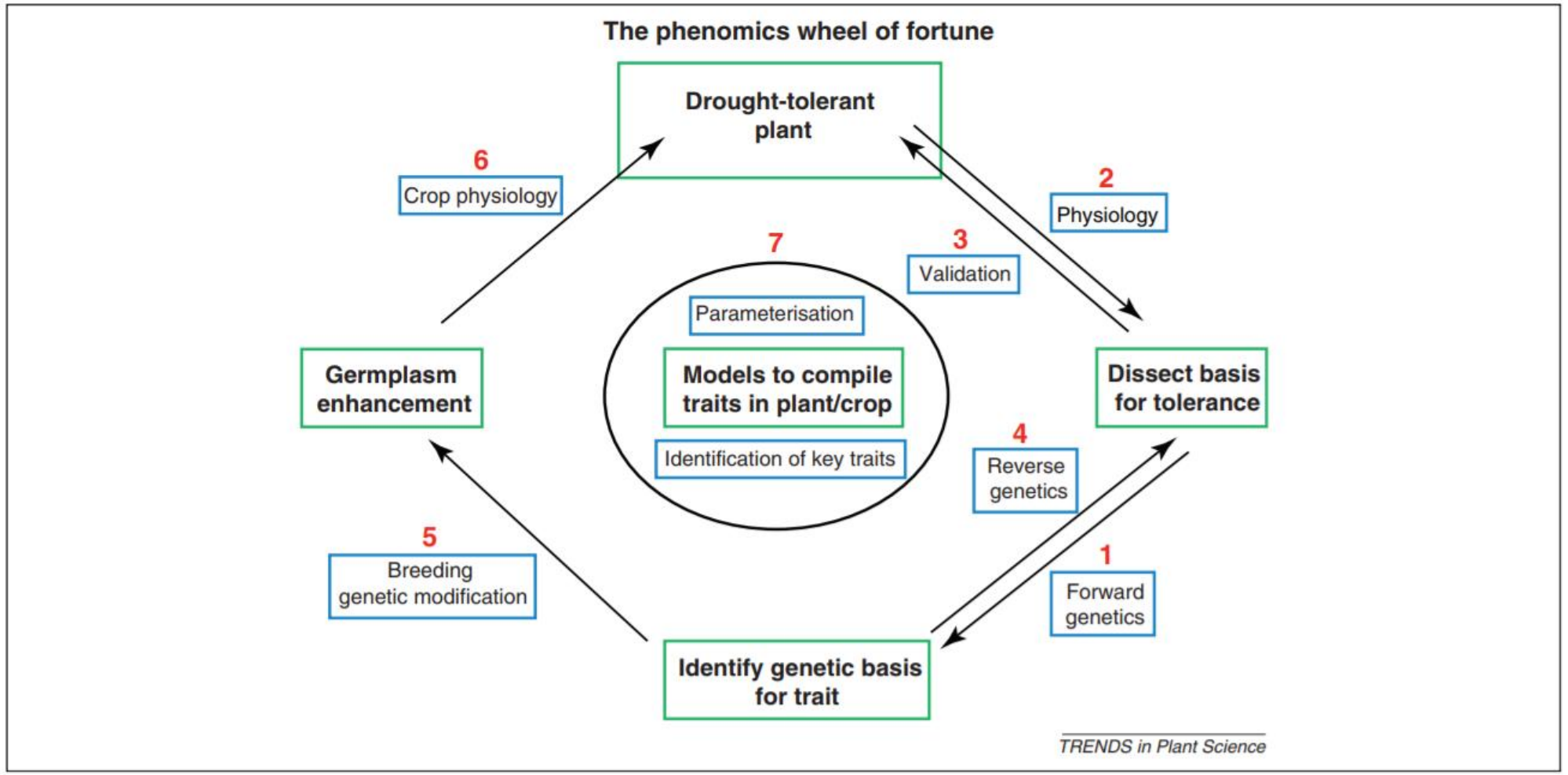


Figure 2. Closing the gene to genotype loop with phenomics.

Analysis of multiple traits

- In linear models: Covariate vs Multivariate
  - Covariate – on trait is used to predict the other. No strings attached to genetics.
  - Multivariate – modeling 2+ traits simultaneously. Genetic components connect traits through genetic correlation.

- Computational burden increases exponentially $O(k)^7$ with the number of traits

- Multicollinearity
  - Many traits are nearly identical (eg. neighbor bandwidths from hyperspec image)
  - OLS will not work due to singularities

- Modeling is often improved by accounting for time-space domains

Multivariate models

Core Ideas

- HTP platforms used to measure secondary traits across time
- Longitudinal data of secondary traits evaluated by SR, MT, and RR models, separately
- BLUPs of secondary traits used in the multivariate pedigree and genomic prediction
- Grain yield predictive ability was improved by 70%

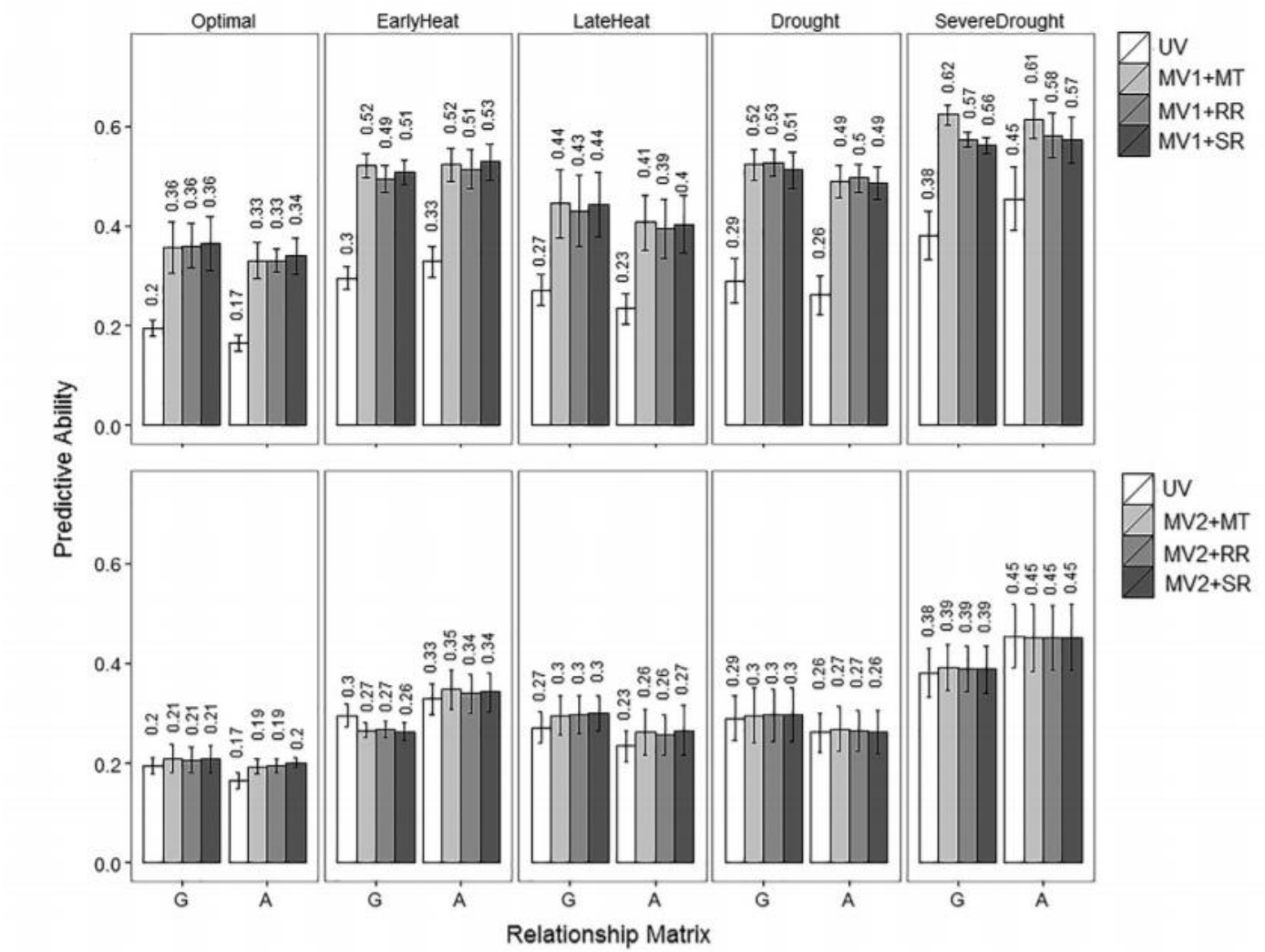
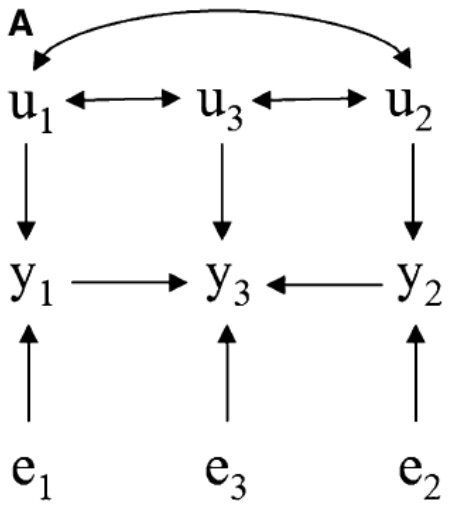


Fig. 1. Predictive ability comparison for grain yield between prediction models with secondary traits (MV1 and MV2) and without secondary trait (UV). MV1, multivariate prediction model with secondary traits in both training and testing populations; MV2, multivariate prediction model with secondary traits in training population only; UV, univariate prediction model with grain yield only; MT/RR/SR, multivariate prediction model MV1 or MV2 using best linear unbiased predictions (BLUPs) of secondary traits from multitrait (MT), random regression (RR), or simple repeatability (SR) model; *G/A*, genomic/pedigree relationship matrix.

Sun, J., Rutkoski, J. E., Poland, J. A., Crossa, J., Jannink, J. L., & Sorrells, M. E. (2017). Multitrait, Random Regression, or Simple Repeatability Model in High-Throughput Phenotyping Data Improve Genomic Prediction for Wheat Grain Yield. *The Plant Genome*.

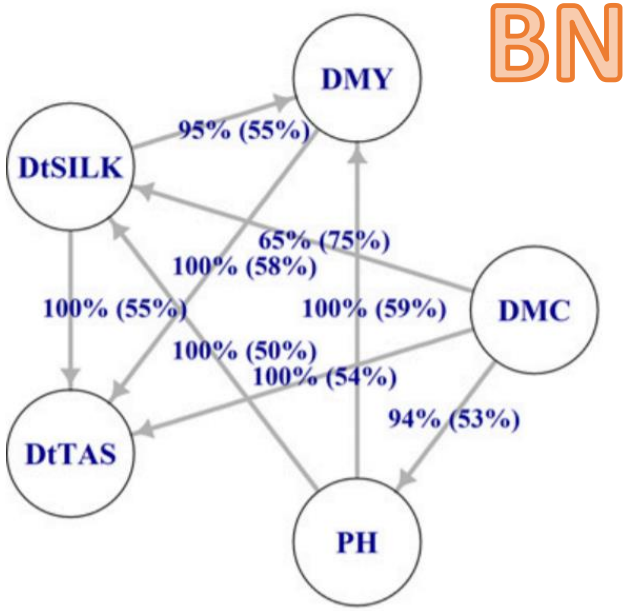
Alternatives to MTM: Structural Equation Models (SEM), Bayesian Networks (BN), and Markov Random Fields (MRF)

SEM



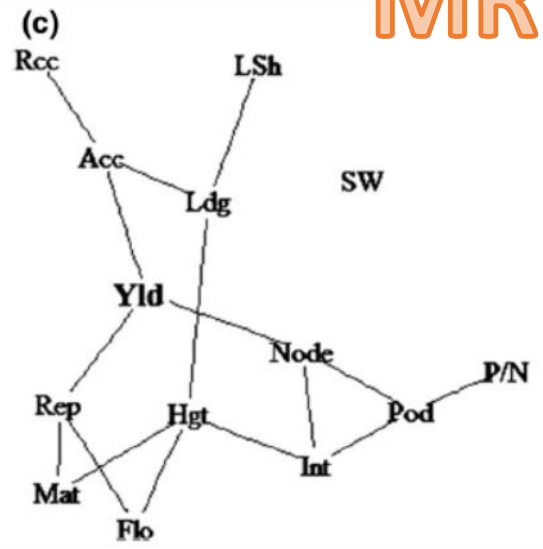
Valente, B. D., Rosa, G. J., de los Campos, G., Gianola, D., & Silva, M. A. (2010). Searching for recursive causal structures in multivariate quantitative genetics mixed models. *Genetics*, *185*(2), 633-644.

BN

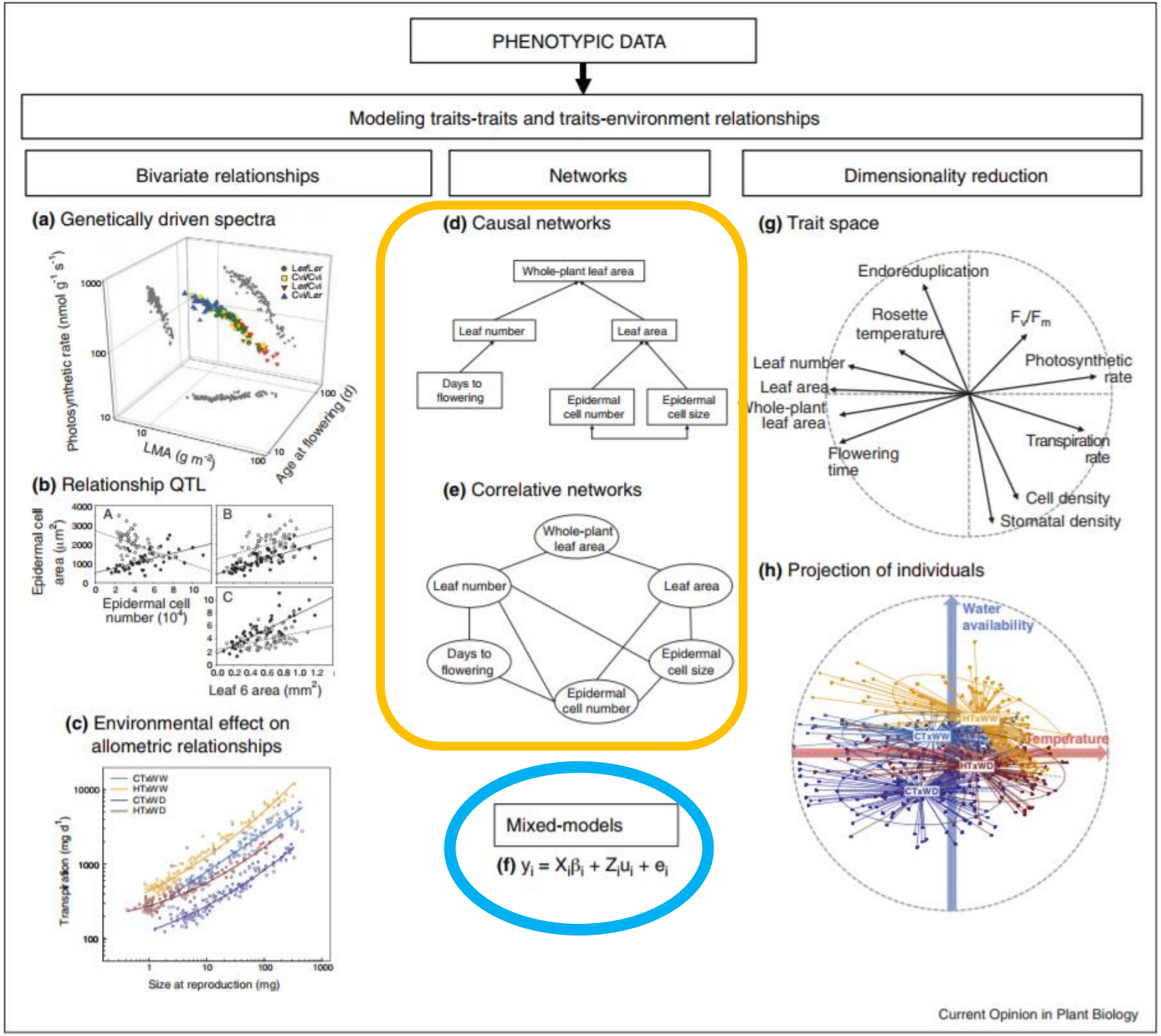


Töpner, K., Rosa, G. J., Gianola, D., & Schön, C. C. (2017). Bayesian Networks Illustrate Genomic and Residual Trait Connections in Maize (Zea mays L.). *G3: Genes, Genomes, Genetics*, *7*(8), 2779-2789.

MRF



Xavier, A., Hall, B., Casteel, S., Muir, W., & Rainey, K. M. (2017). Using unsupervised learning techniques to assess interactions among complex traits in soybeans. *Euphytica*, *213*(8), 200.



Granier, C., & Vile, D. (2014). Phenotyping and beyond: modelling the relationships between traits. *Current opinion in plant biology*, *18*, 96-102.

3. Considerations on big data, data mining and statistical computing

Number of observations and paramenters

Will Big Data Close the Missing Heritability Gap?

Hwasoon Kim,* Alexander Grueneberg,*,† Ana I. Vazquez,*,† Stephen Hsu,†,§ and Gustavo de los Campos*,†,**,§,1

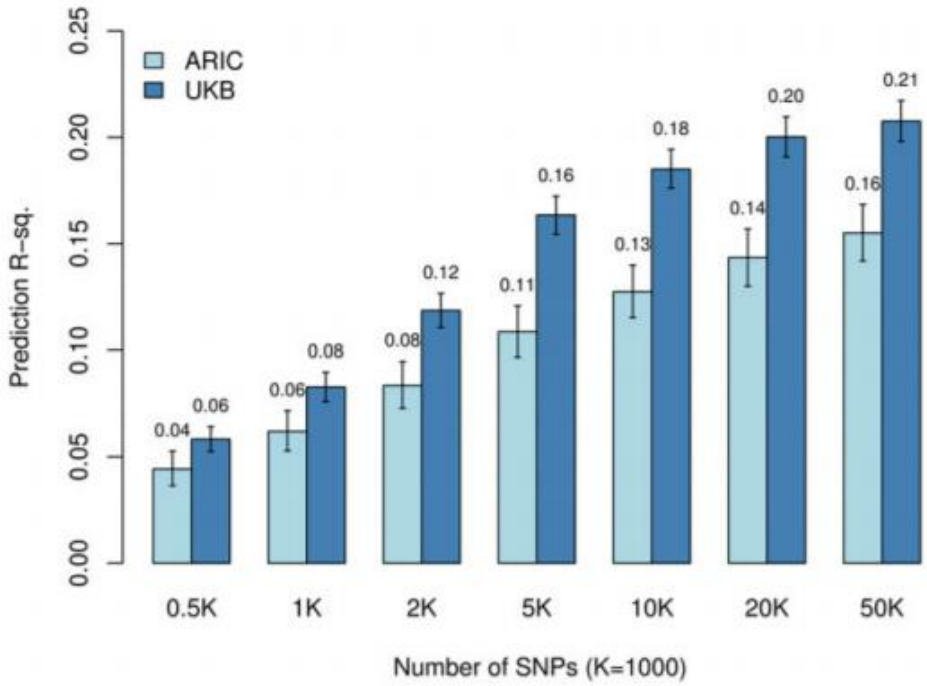


Figure 6 An external validation in a U.S. cohort yields moderately high prediction R-sq. Each bar shows the prediction R-sq. by combinations of genotypes and numbers of SNPs in ARIC (n TST = 9591) and UK Biobank (n TST = 22,221) cohorts. The vertical bars represent 95% C.I.'s. UKB, UK Biobank.

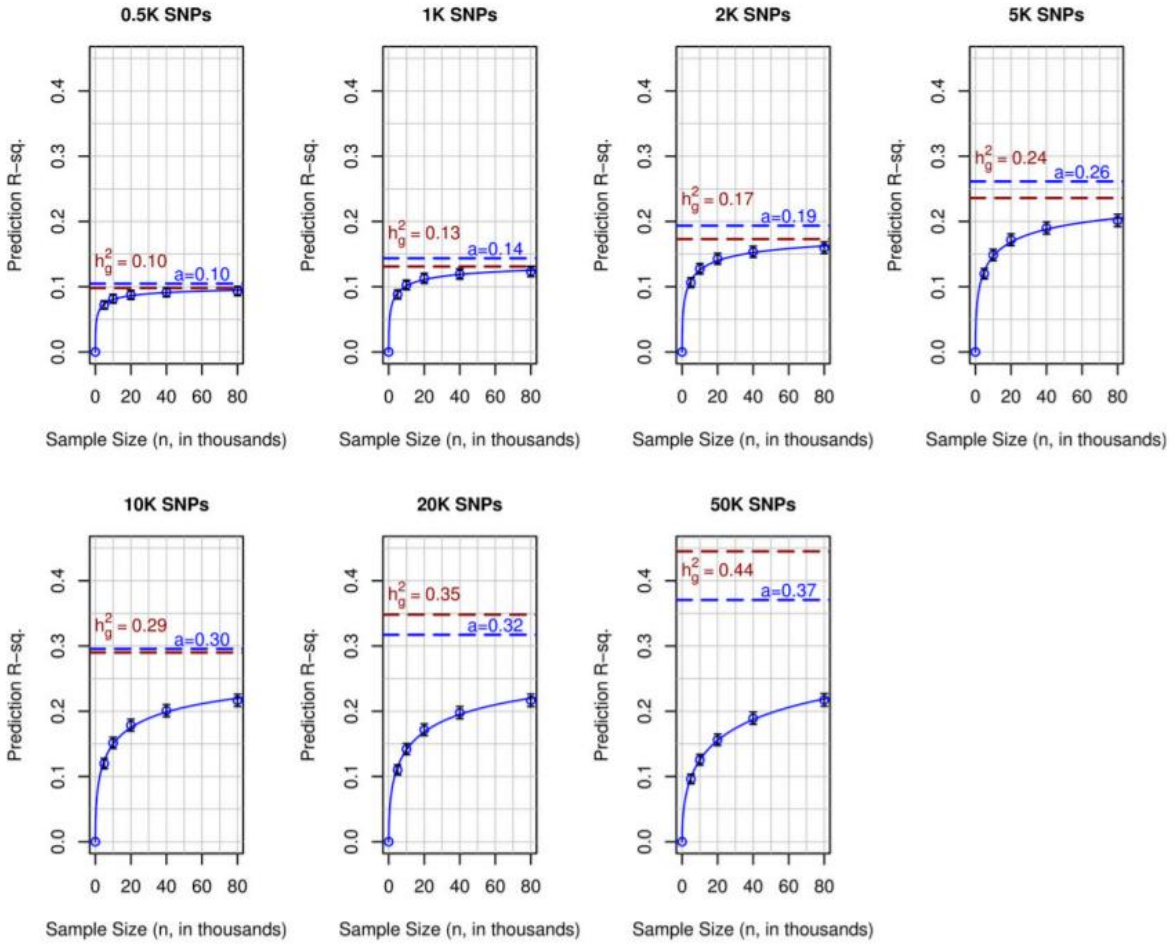


Figure 3 Polygenic prediction coupled with big data closes a sizable fraction of gap between prediction R-sq. and trait heritability. Each panel shows the average prediction R-sq. (○) achieved in the testing set (n = 22,221) by the number of SNPs used (p = 500, 1K, ⋯, 50K, K = 1000) *vs.* the size of the data set used to train models. The solid blue curve corresponds to a nonlinear function, $R^2(n) = a\sqrt{n}/\sqrt{n} + b$, fitted by least squares. The dashed blue horizontal line gives the estimated maximum prediction R-sq. (â) for each SNP set. The dashed red horizontal line gives the estimated genomic heritability of the SNP set (estimated using data from the TST set).

Reduction of dimensionality via SVD

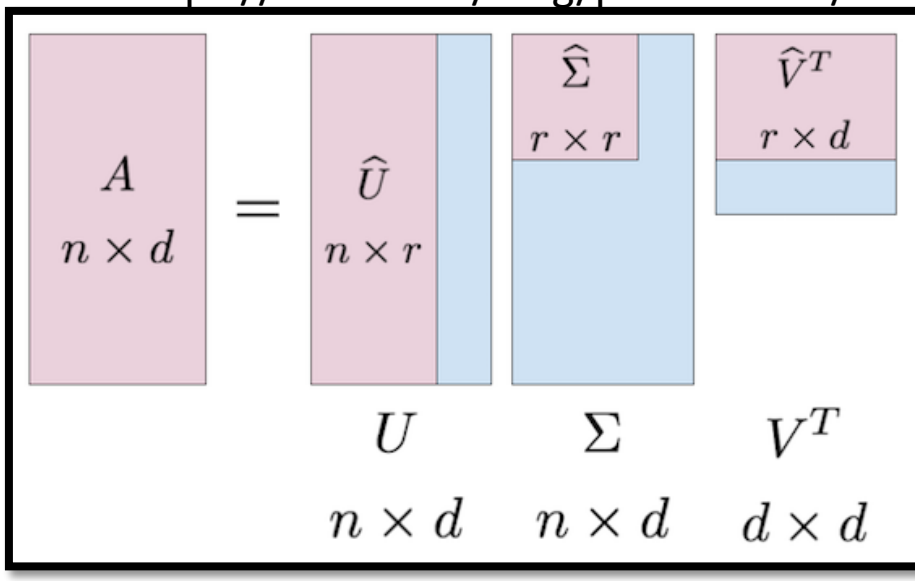
- Single value decomposition (for rectangular matrices), also known as spectral decomposition or Eigendecomposition (for square matrices)

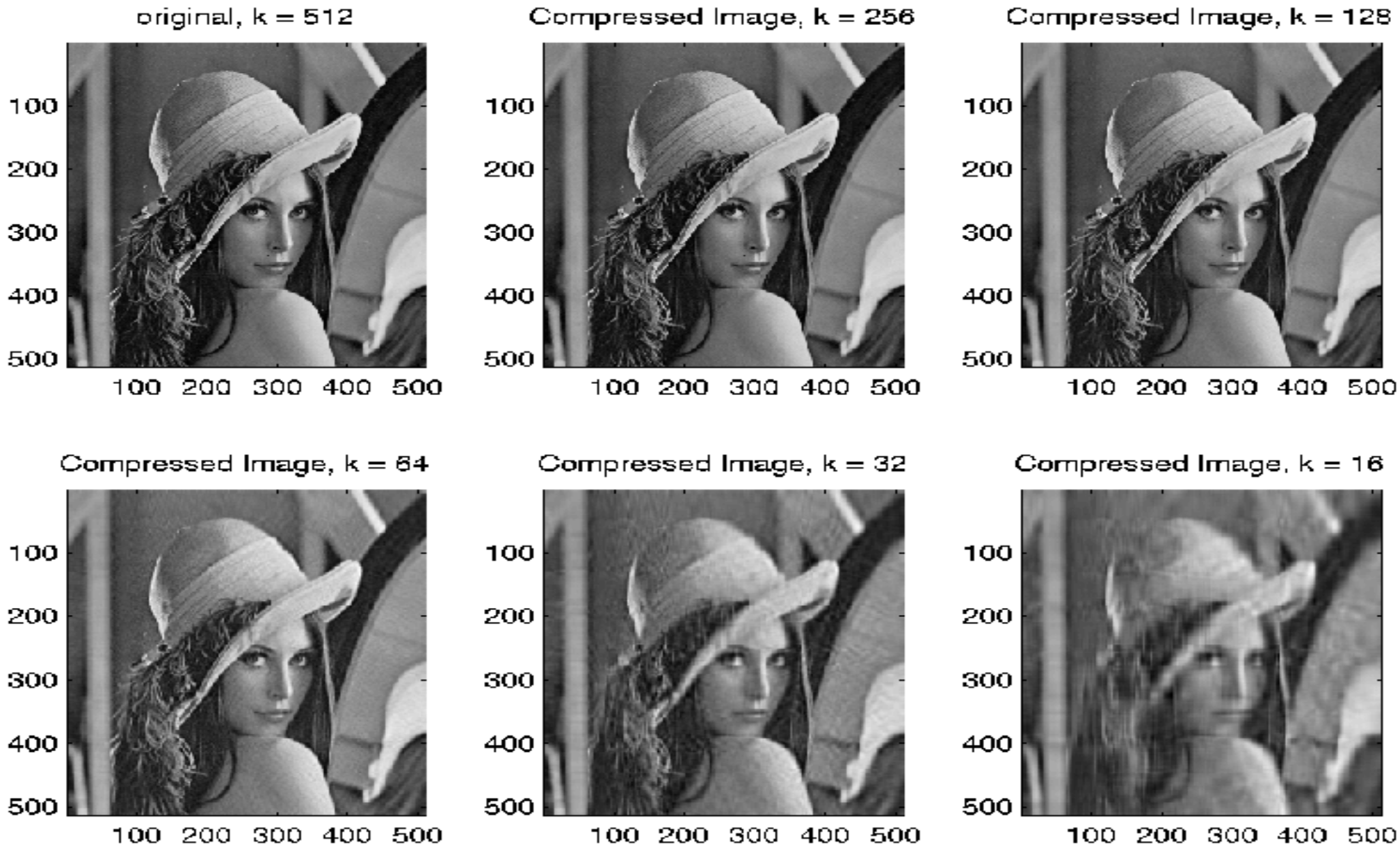
- Backbone of principal component analysis and bi-plot projections of data. Also widely used to simplify though calculations and polynomials.

- The information from a matrix is condensed into a set orthogonal components, where most information is usually compressed in the first few components

$$\mathbf{X} = \mathbf{UDV}'$$

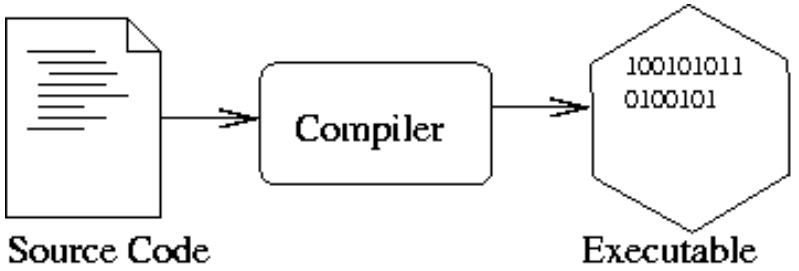
https://intoli.com/blog/pca-and-svd/





http://fourier.eng.hmc.edu/e161/lectures/svdcompression.html

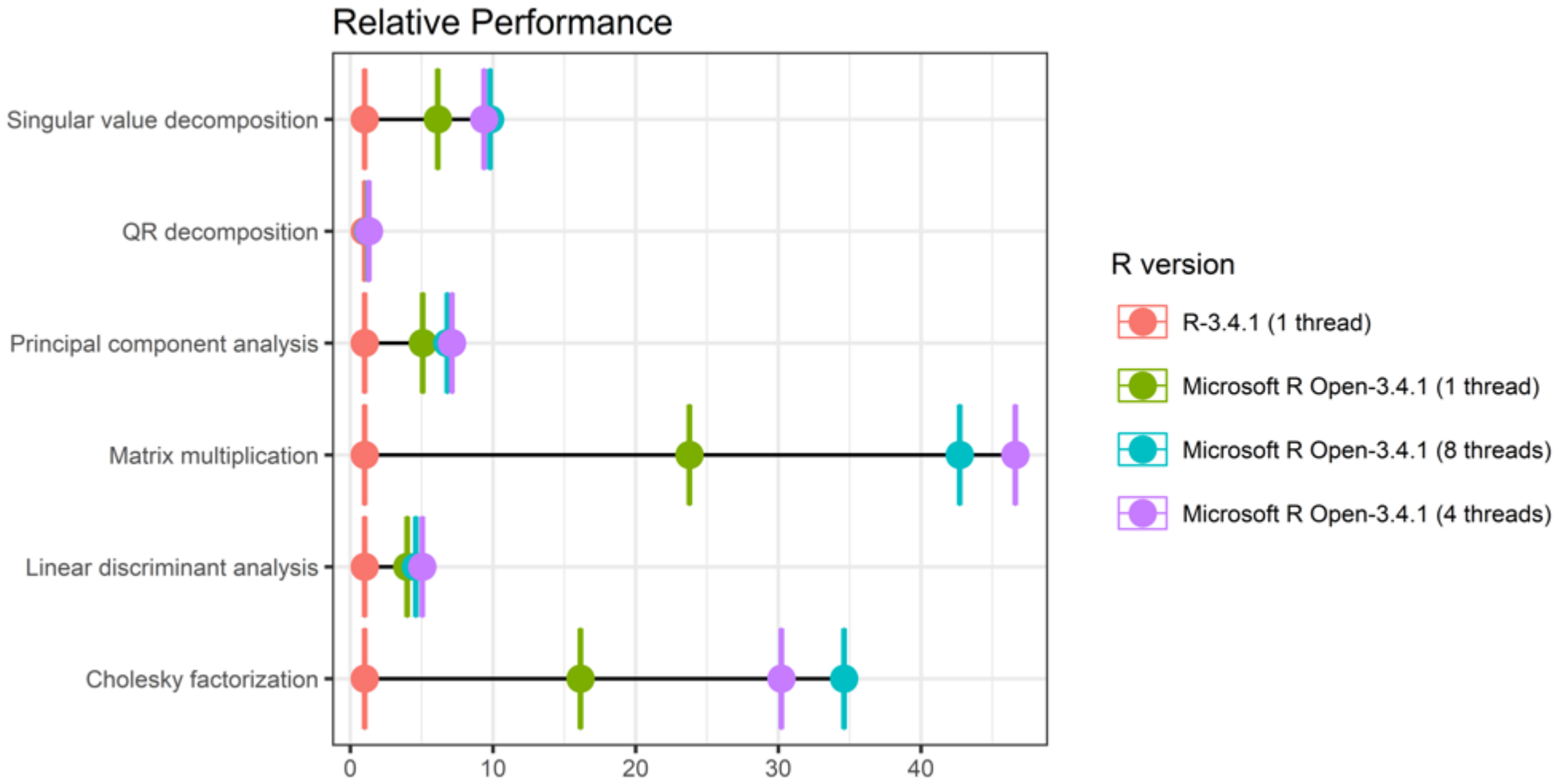
Compiler



- R is a dynamic language, you can program and execute your function on the fly. But basic function and loops are generally inefficient. When you compile your code, the function is optimized (communicate better with the machine)

- Compiling function you wrote: ***compiler::cmpfun(my_function)***
- You can write a package (packages are compiled with GNU compiler)
- Use packages compiled by **Microsoft** (mran.microsoft.com)
  - With Microsoft's free version of R, everything is much faster

Improvement by using Microsoft's compilation



Sparsity

- When matrices are not dense (all cell have numbers), they can be stored more efficiently by not saving the zero elements.
- Sparse storage also makes computation faster since it reduces the calculation to non-zero elements.
- How to do it: Use R package **Matrix** to build design matrices. The code doesn't really change (except that *matrix* becomes *Matrix*).

Sparsity – Dense vs Sparse storage

Dense

1	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	5	0
0	3	0	0	0
0	0	1	0	0
0	0	0	0	1

Sparse

1	-	-	-	-
-	-	-	-	-
-	-	-	-	-
-	-	-	5	-
-	3	-	-	-
-	-	1	-	-
-	-	-	-	1

Stored

X	Y	Val
1	1	1
2	5	3
3	6	1
4	3	5
7	5	1

Sparsity – Symmetric matrices

Dense

1	3	4	6	2	5	2
3	1	5	1	3	5	0
4	5	1	0	3	1	1
6	1	0	1	9	8	0
2	3	3	9	1	5	0
5	5	1	8	5	1	0
2	0	1	0	3	8	1

Sparse

1	-	-	-	-	-	-
3	1	-	-	-	-	-
4	5	1	-	-	-	-
6	1	0	1	-	-	-
2	3	3	9	1	-	-
5	5	1	8	5	1	-
2	0	1	0	3	8	1

Data larger than memory

- When data is larger than your computer can handle, you can load in memory only what is going to be use

- Hierarchical data format (HDF5) files is a binary designed to optimize use and storage, compatible with R, Python, Matlab, etc.
  - See R package ***h5***

- There ongoing efforts to enable parallel computation of files larger than memory in R
  - See ***bigmemory*** project (http://www.bigmemory.org/)
  - See R packages ***ff*** (for storage) and ***bigpca***

Parallel computing

- Multithread (CPU) – Perform tasks that are independent in nature:
  - Example: Analyses performed on single alleles (GWAS, FST, allele frequency)

- GPU – Very large number of computations with little memory
  - Example: Matrix inversion and multiplication, Navier-Strokes, Monte Carlo

- Cloud – Large computation, large memory
  - Example: Training neural nets, fit data with large dimensions (*p* or *n*)

Monte Carlo methods

- Monte Carlo describes a wide range of sampling and simulation methods to solve problems that are too complex or do not have analytical solution
- **Case of study**: A problem has computational cost that is an exponential of the number of observations. An example is REML variance components (EMMA algorithm):

$$\boldsymbol{VC = O(n^3)}$$

- If we solve 10x using 25% of the data, we get:

$$\boldsymbol{VC_{MC} = O\left(10 \times (n_{25\%})^3\right)}$$

- If the analysis have 100 individuals

$$VC = 100^3 = 10^6$$
$$VC_{MC} = 10 \times 25^3 = 1.56 \times 10^5$$

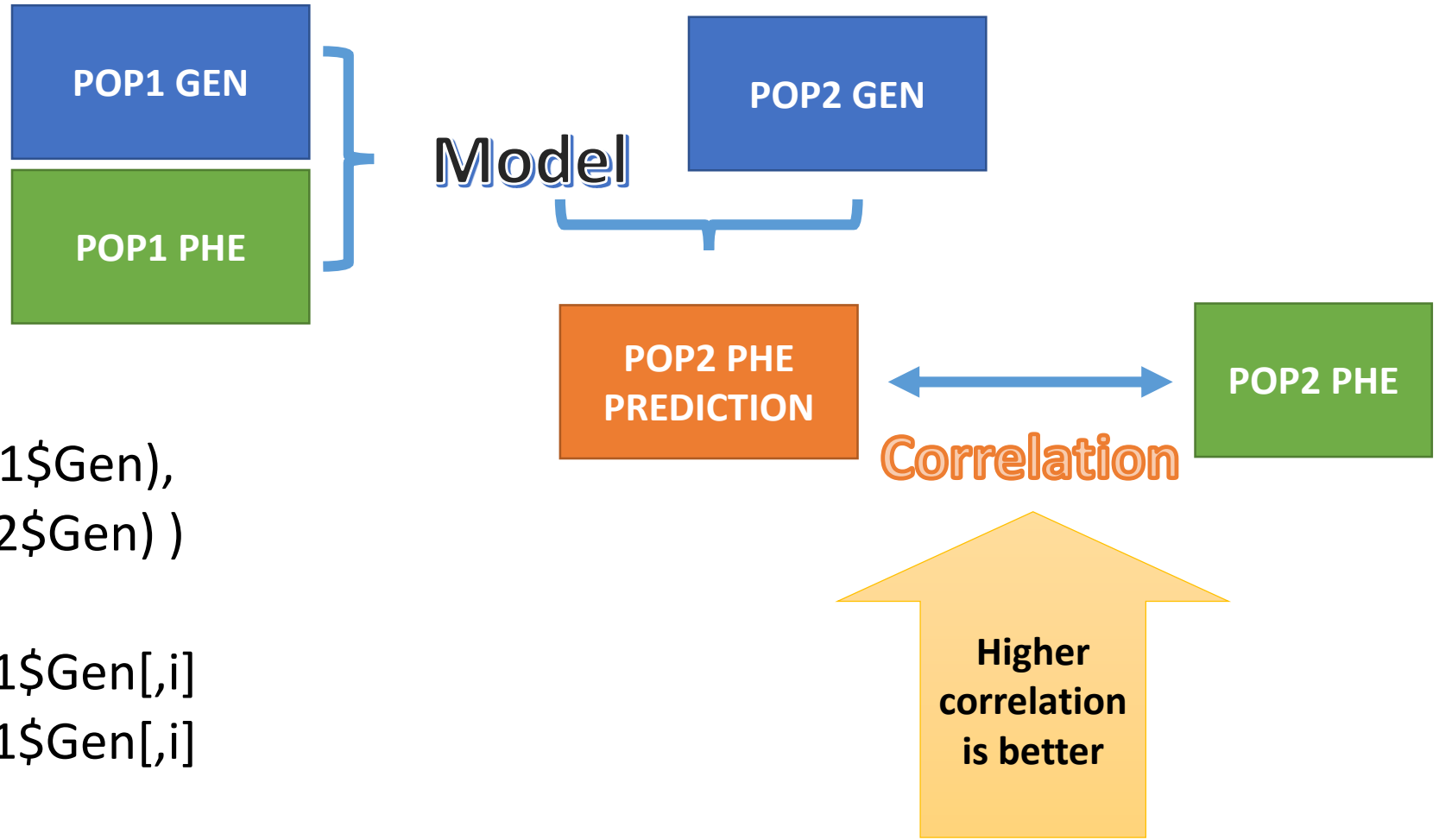
- In this case, MC is **6.4x faster** than the most efficient VC method

Nick Metropolis

4. Genomic prediction exercise (**R codes included**)

Getting training and validation datasets

```
require(SoyNAM)
# TRAINING
pop1 = BLUP(fam=3)
# VALIDATION
pop2 = BLUP(fam=4)
# Matching SNP sets
i = intersect( colnames(pop1$Gen),
               colnames(pop2$Gen) )
# Renaming objects
Y1 = pop1$Phen;  X1 = pop1$Gen[,i]
Y2 = pop2$Phen;  X2 = pop1$Gen[,i]
```

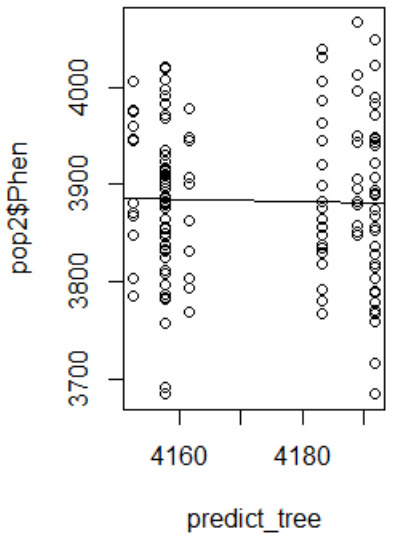


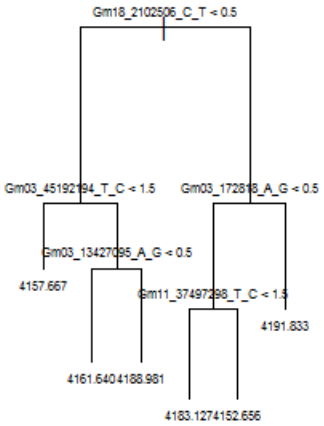
Supervised machine learning methods

- Trees and random forest
- Penalized L_1L_2 linear models
- Neural networks
- Spectral methods: PLS and PC
- Reproducing kernel in Hilbert spaces
- Support vector regressions
- Bayesian learners
- K nearest neighbors
- Bootstrap Aggregation (Bagging)

Regression tree (-0.0220)

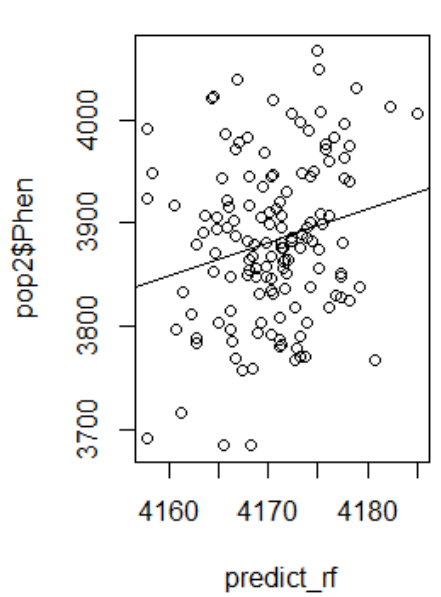
```
require(tree)
# FIT USING POP1
fit_tree = tree(y~.,data.frame(y=Y1,X1),)
fit_tree = prune.tree(fit_tree, best = 8)
# PREDICT POP2
predict_tree = predict(fit_tree, data.frame(X2) )
# PLOT
par(mfrow = c(1,2))
plot(predict_tree, Y2);
abline(lm(Y2~predict_tree))
plot(fit_tree); text(fit_tree,cex=0.5)
```

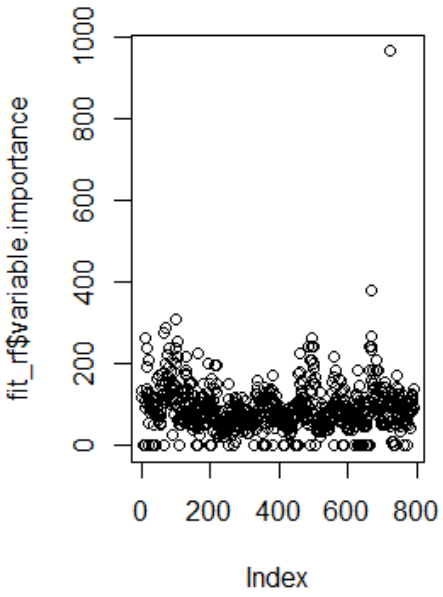




Random forest (0.2103)

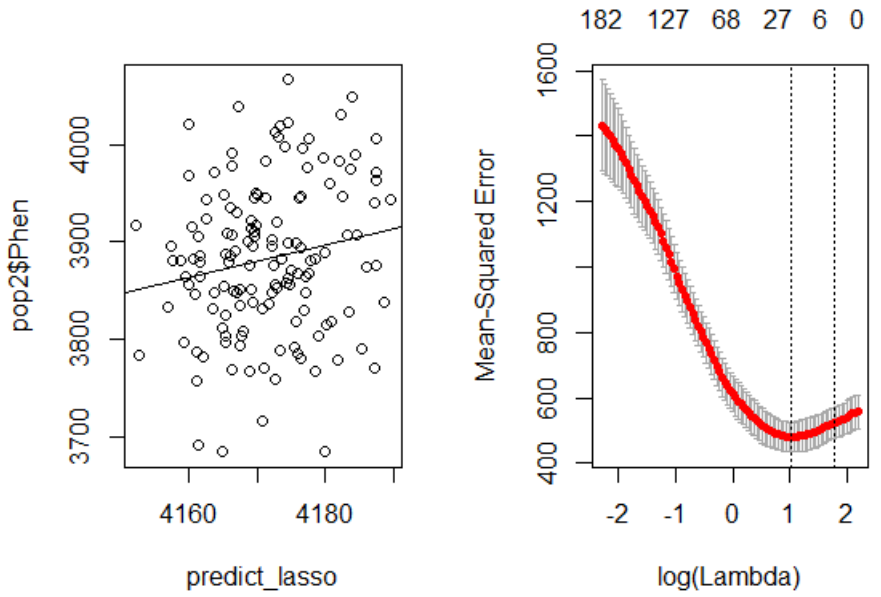
```
require(ranger)
# FIT USING POP1
fit_rf = ranger(y~.,data.frame(y=Y1,X1), importance='impurity')
# PREDICT POP2
predict_rf = predict(fit_rf, data.frame(X2) )$predictions
# PLOT
par(mfrow = c(1,2))
plot(predict_rf, Y2)
abline(lm(Y2~predict_rf))
plot(fit_rf$variable.importance)
```





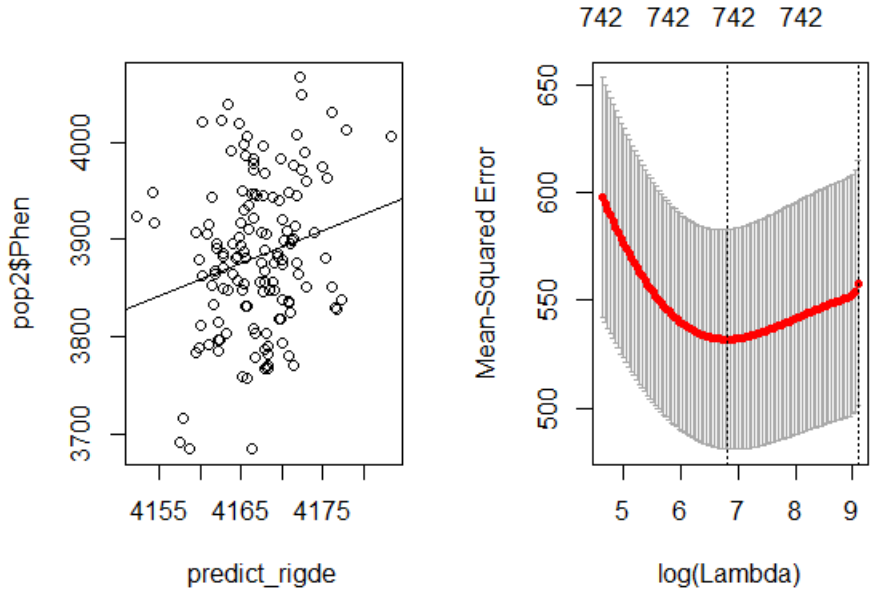
LASSO (0.1750)

```
require(glmnet)
# FIT USING POP1
cv_EN = cv.glmnet(y=Y1,x = X1, alpha = 1)
fit_EN = glmnet(y=Y1, x = X1, alpha = 1, lambda = cv_EN$lambda.min)
# PREDICT POP2
predict_EN = predict(fit_EN, X2 )
# PLOT
par(mfrow = c(1,2))
plot(predict_EN, Y2)
abline(lm(Y2~predict_EN))
plot(cv_EN);
```



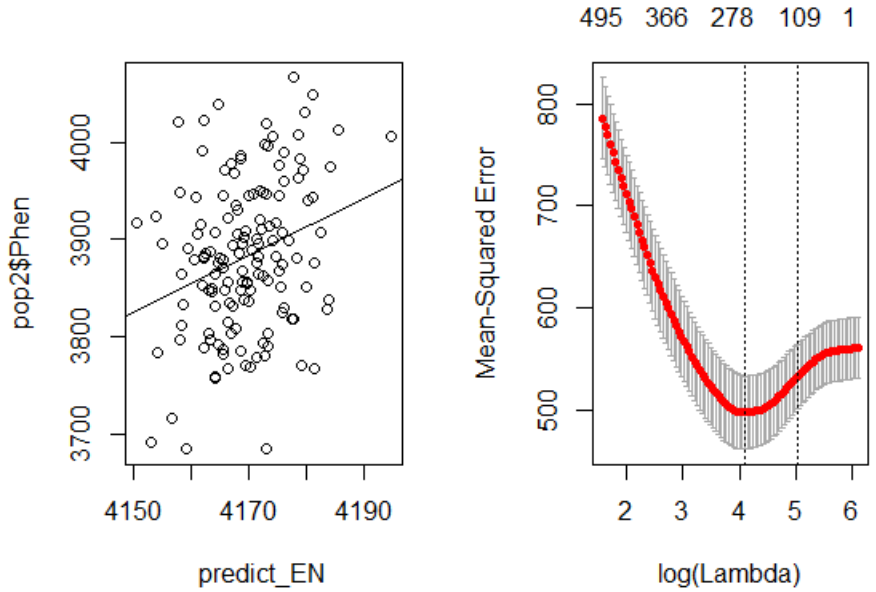
Ridge Regression (0.2137)

```
require(glmnet)
# FIT USING POP1
cv_ridge = cv.glmnet(y=Y1,x = X1, alpha = 0)
fit_ridge = glmnet(y=Y1, x = X1, alpha = 0, lambda = cv_ridge$lambda.min)
# PREDICT POP2
predict_rigde = predict(fit_ridge, X2 )
# PLOT
par(mfrow = c(1,2))
plot(predict_rigde, Y2)
abline(lm(Y2~predict_ridge))
plot(cv_ridge);
```



Elastic-Net (0.2766)

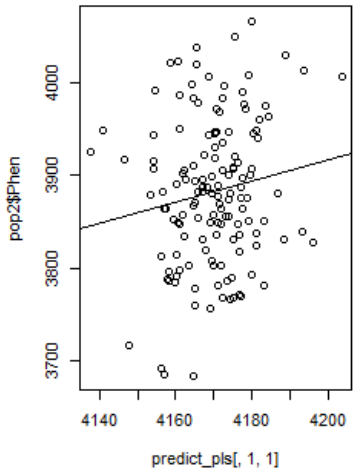
```
require(glmnet)
# FIT USING POP1
cv_EN = cv.glmnet(y=Y1,x = X1, alpha = 0.02)
fit_EN = glmnet(y=Y1, x = X1, alpha = 0.02, lambda = cv_EN$lambda.min)
# PREDICT POP2
predict_EN = predict(fit_EN, X2 )
# PLOT
par(mfrow = c(1,2))
plot(predict_EN, Y2)
abline(lm(Y2~predict_EN))
plot(cv_EN);
```



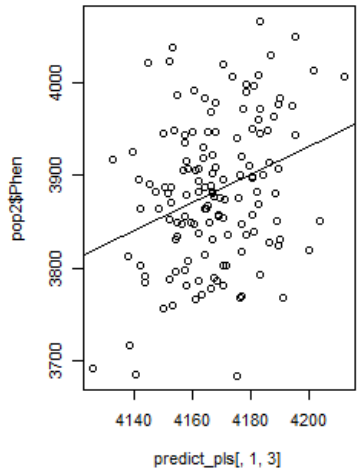
Partial Least Square (0.2793)

```
require(pls)
# FIT USING POP1
cv_pls = plsr(y~.,ncolp=5,data=data.frame(y=Y1,X1))
# PREDICT POP2
predict_pls = predict(cv_pls, X2 )
# PLOT
par(mfrow = c(1,3))
plot(predict_pls[,1,1], Y2)
plot(predict_pls[,1,3], Y2)
plot(predict_pls[,1,5], Y2)
```

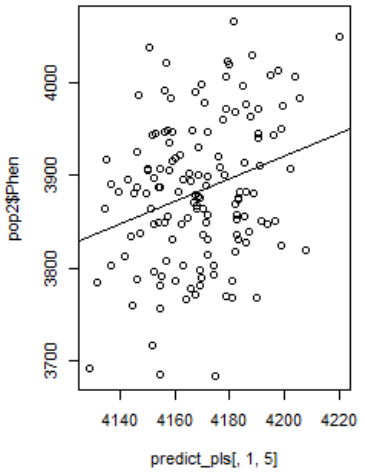
0.147



0.301

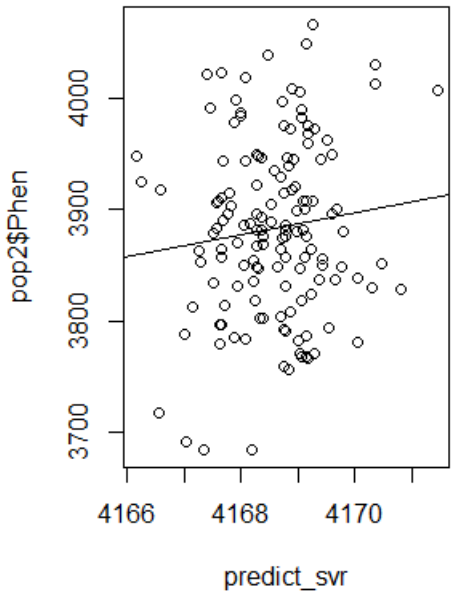


0.279



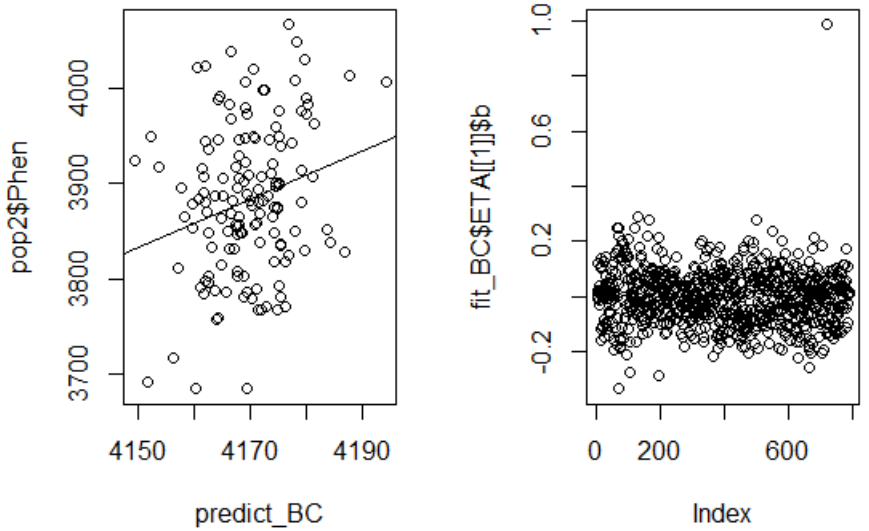
Support Vector Regression (0.1104)

```
require(kernlab)
# FIT USING POP1
fit_svr = ksvm(x = X1, y = Y1, type = "eps-svr", scaled = FALSE)
# PREDICT POP2
predict_svr = predict(fit_svr, X2)
# PLOT
par(mfrow = c(1,1))
plot(predict_svr, Y2)
abline(lm(Y2~predict_svr))
cor(predict_svr, Y2)
```



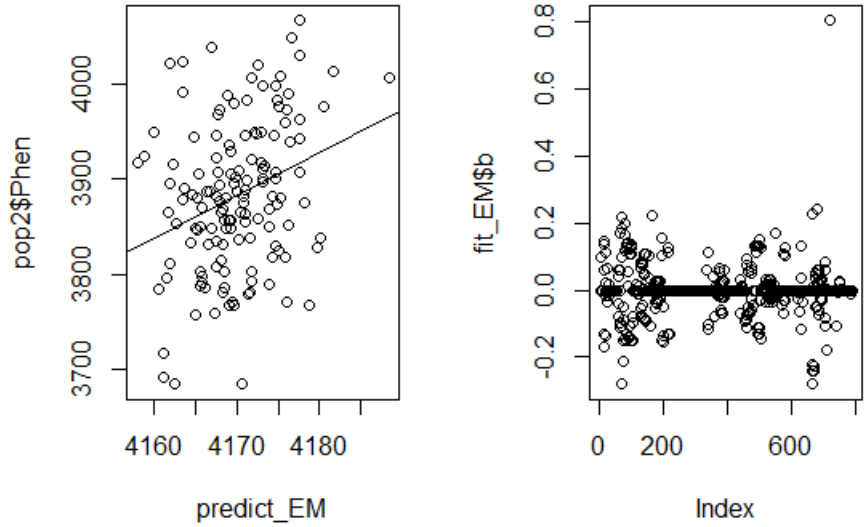
MCMC-WGR (0.2682)

```
require(BGLR)
# FIT USING POP1
fit_BC = BGLR(y=Y1, ETA = list(list(X=X1,model='BayesC')))
        # other models include: BayesA, BayesB, BL, BRR
# PREDICT POP2
predict_BC = fit_BC$mu + X2 %*% fit_BC$ETA[[1]]$b
# PLOT
par(mfrow = c(1,2))
plot(predict_BC, Y2)
abline(lm(Y2~predict_BC))
plot(fit_BC$ETA[[1]]$b)
```



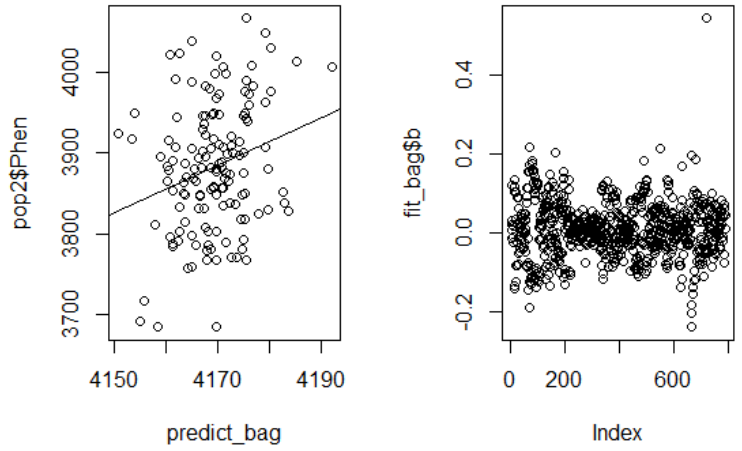
EM-WGR (0.2970)

```
require(bWGR)
# FIT USING POP1
fit_EM = emEN(y=Y1, gen =  X1)
        # other models include: emDE, emBL, emBA, emRR
# PREDICT POP2
predict_EM = fit_EM$mu + X2 %*% fit_EM$b
# PLOT
par(mfrow = c(1,2))
plot(predict_EM, Y2)
abline(lm(Y2~predict_EM))
plot(fit_EM$b)
```



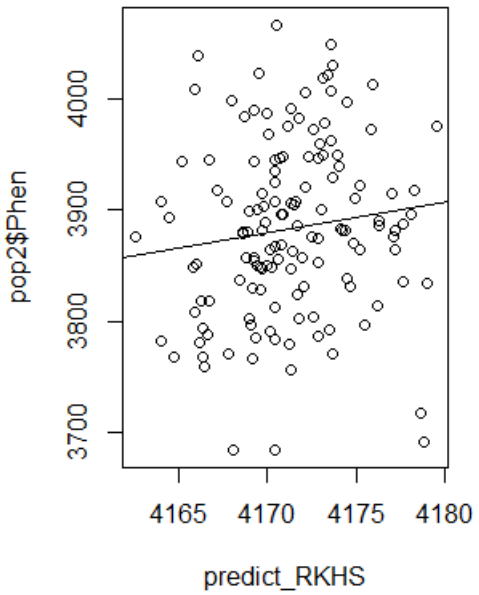
Bagging-WGR (0.2494)

```
require(bWGR)
# FIT USING POP1
fit_bag = wgr(y=Y1, X =  X1, de=T, bag = 0.75)
# PREDICT POP2
predict_bag = fit_bag$mu + X2 %*% fit_bag$b
# PLOT
par(mfrow = c(1,2))
plot(predict_bag, Y2)
abline(lm(Y2~predict_bag))
plot(fit_bag$b)
```



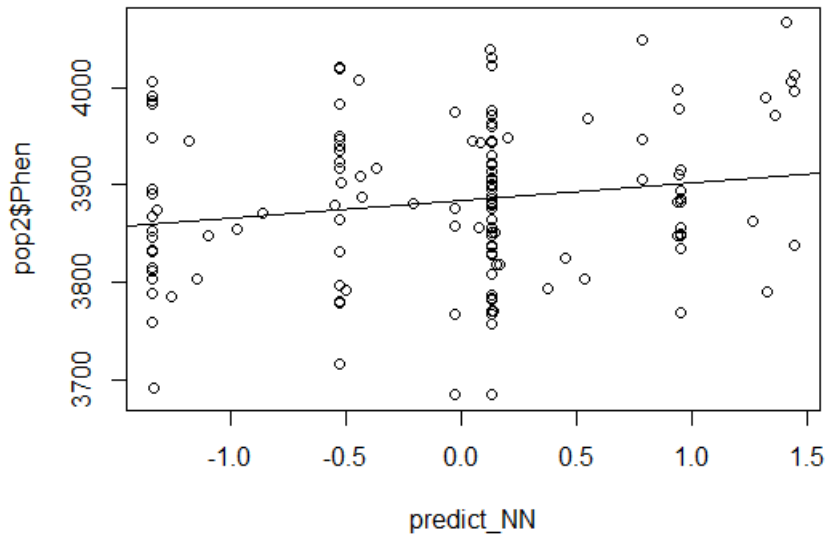
RKHS (0.1203)

```
require(BGLR)
# Building the relationship matrix
K = NAM::GAU(rbind(X2,X1))
# FIT USING POP1
fit_RKHS = BGLR(y=c(Y1,rep(NA,length(Y2))), ETA = list(list(K=K,model='RKHS')))
# PREDICT POP2
predict_RKHS = fit_RKHS$yHat[c(1:length(Y2))]
# PLOT
par(mfrow = c(1,1))
plot(predict_RKHS, Y2)
abline(lm(Y2~predict_RKHS))
cor(predict_RKHS, Y2)
```



Neural Networks (0.1568)

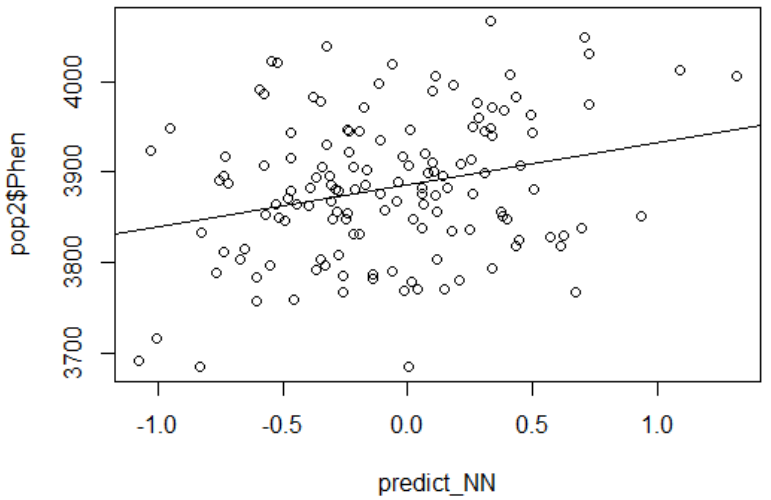
```
require(nnet)
# FIT USING POP1
y = (Y1-mean(Y1))/sd(Y1)
fit_NN = nnet(y=y,x=X1,size=3,
              decay=0.001,MaxNWts=Inf,linout=T)
# PREDICT POP2
predict_NN = predict(fit_NN,X2)
# PLOT
plot(predict_NN, Y2)
abline(lm(Y2~predict_NN))
cor(predict_NN, Y2)
```



 # The package *brnn* (0.2622) fits regularized neural nets, more suitable for GS

Neural Networks (0.2643) with Bagging

```
require(nnet)
# FIT USING POP1 and PREDICT POP2 - 100x
h = rep(0,length(Y2)); nit = 100
for(i in 1:nit){ N=sample(nrow(X1),100);P=sample(ncol(X1),100)
  y=(Y1[N]-mean(Y1[N]))/sd(Y1[N])
  fit_NN = nnet(y=y,x=X1[N,P],trace=F,size=3,decay=rbeta(1,1,100),linout=T)
  h = h + predict(fit_NN,X2[,P])}
predict_NN = h/nit
# PLOT
plot(predict_NN, Y2)
abline(lm(Y2~predict_NN))
cor(predict_NN, Y2)
```



Thanks for your attention

alenxav@gmail.com

http://alenxav.wix.com/home

Tools for sceptical thinking

- Wherever possible there must be independent confirmation of the 'facts'.
- Encourage substantive debate on the evidence by knowledgeable proponents of all points of view.
- Arguments from authority carry little weight - 'authorities' have made mistakes in the past. They will do so again in the future. Perhaps a better way to say it is that in science there are no authorities; at most, there are experts.
- Spin more than one hypothesis. If there's something to be explained, think of all the different ways in which it *could* be explained. Then think of tests by which you might systematically disprove each of the alternatives. What survives, the hypothesis that resists disproof in this Darwinian selection among 'multiple working hypotheses', has a much better chance of being the right answer than if you had simply run with the first idea that caught your fancy.*
- Try not to get overly attached to a hypothesis just because it's yours. It's only a way-station in the pursuit of knowledge. Ask yourself why you like the idea. Compare it fairly with the alternatives. See if you can find reasons for rejecting it. If you don't, others will.
- Quantify. If whatever it is you're explaining has some measure, some numerical quantity attached to it, you'll be much better able to discriminate among competing hypotheses. What is vague and qualitative is open to many explanations. Of course there are truths to be sought in the many qualitative issues we are obliged to confront, but finding *them* is more challenging.
- If there's a chain of argument, *every* link in the chain must work (including the premise) - not just most of them.
- Occam's Razor. This convenient rule-of-thumb urges us when faced with two hypotheses that explain the data *equally well* to choose the simpler.
- Always ask whether the hypothesis can be, at least in principle, falsified. Propositions that are untestable, unfalsifiable are not worth much. Consider the grand idea that our Universe and everything in it is just an elementary particle - an electron, say - in a much bigger Cosmos. But if we can never acquire information from outside our Universe, is not the idea incapable of disproof? You must be able to check assertions out. Inveterate sceptics must be given the chance to follow your reasoning, to duplicate your experiments and see if they get the same result.

Carl Sagan - The Demon Haunted World (p.197)