234 Final Project: Stability Selection on High Dimensioinal Setting

Jinglin Yang

2025-12-09

Introduction

In this project, I would examine the enhanced feature selection techniques on high dimensional setting, where there are more variables than observations, i.e., $p >> n$. Since high dimensional problem suffers from unstable variable selection as there are too few samples to generalize the pattern, stability selection on resampling the limited observations helps improve the model by selecting the true, relevant dependent variables that account for the response. In addition, using regularization method, such as lasso, is even challenging because it is hard to choose the perfect or the right λ. And even small change of λ can result in completely different set of variables.

The dataset being implemented is the riboflavin dataset from `hdi` package, which contains 4088 gene expression variables and 71 observations. The response variable is the logarithmic production rate of riboflavin, or vitamin B2.

```
data("riboflavin")
X <- riboflavin$x
y <- riboflavin$y
p <- ncol(X)
n <- nrow(X)

cat("Dimension of predictors: ", paste(dim(X), collapse = " x "), "
")
```

```
Dimension of predictors:  71 x 4088
```

```
cat("Number of individual sample: ", length(y), "
")
```

```
Number of individual sample:  71
```

Mathematics Procedure

1. Define $q = \sqrt{0.8p}$ that specify maximum number of selected variables at each regularization level.
2. The regularization region $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_l\}$ has the inverval $[\lambda_{min}, \lambda_{max}]$ and is defined such that for all $\lambda \in \Lambda, |S^\lambda| \leq q$, which ensures that no more than q distinct variables contribute to the model across all regularization parameters.
3. For $b = 1, ..., B$,

- Draw a subsample $I_b \subset 1, ..., n, |I_b| = \frac{n}{2}$
- Let $X^{(b)} = X_{I_b}, y^{(b)} = y_{I_b}$

4. For each value $\lambda \in \Lambda$:

- For each $b \in \{1, ..., B\}$,
  - Fit a lasso estimator on the bootstrap sample with regularization parameter λ:

    $\hat{\beta}^{(b)}(\lambda) = \min_\beta \left\{ \|y^{(b)} - X^{(b)}\beta\|_2^2 + \lambda\|\beta\|_1 \right\}$

- Given the selection set $S^{\lambda,b} = \{k : \hat{\beta}_k^{(b)} \neq 0\}$ from each subsample, calculate the selection probability for each signal predictors:

  $\hat{\Pi_k^\lambda} = P(K \subset S^{\lambda,b}) = \frac{1}{B}\sum_{b=1}^{B} \mathbf{I}_{\{k \in \hat{S}^{\lambda,b}\}}$
- The selection probability for predictor k is its probability of having nonzero coefficients under regularization level λ over B samples.

5. Given a predefined threshold $\pi_{thr} \in \ (0, 1)$ and the selection probabilities $\hat{\Pi_k^\lambda}$ for each predictor and for each value of λ, construct that stable set as:

$$\hat{S_{stable}} = \{k : max_{\lambda \in \Lambda}\hat{\Pi_k^\lambda} \geq \pi_{thr}\}$$

   These are predictors that are selected frequently across subsamples and across different regularization strengths
6. (Optional) Randomized Lasso Weighting step

- Each subsample uses randomized penalty wights

$$w_j^{(b)} \sim U(\alpha, 1),$$

  where $\alpha \in (0, 1]$ is the predefined weakness level.

  the randomized lasso estimator becomes:

$$\hat{\beta^{(b)}} = \min_\beta \left\{ \|y^{(b)} - X^{(b)}\beta\|_2^2 + \lambda\sum_{k=1}^{p}\frac{|\beta_k|}{w_k^{(b)}} \right\}$$

Reproducing Procedure

In this section, I am going to reproduce figure 1 in the original paper that plot the regularization path, lasso stability path, and randomized lasso stability path on riboflavin dataset.

Below is the permutation process that the author set up to randomly select 6 gene expression variable out of 200 most correlated variables to the response, and the rest 4082 variables are permuted to serve as noise variables.

Permutation helps us examine the performance of stability selection whether it can correctly select the 6 signal variables other than over 4000 noise variables.

Regularization grid selection

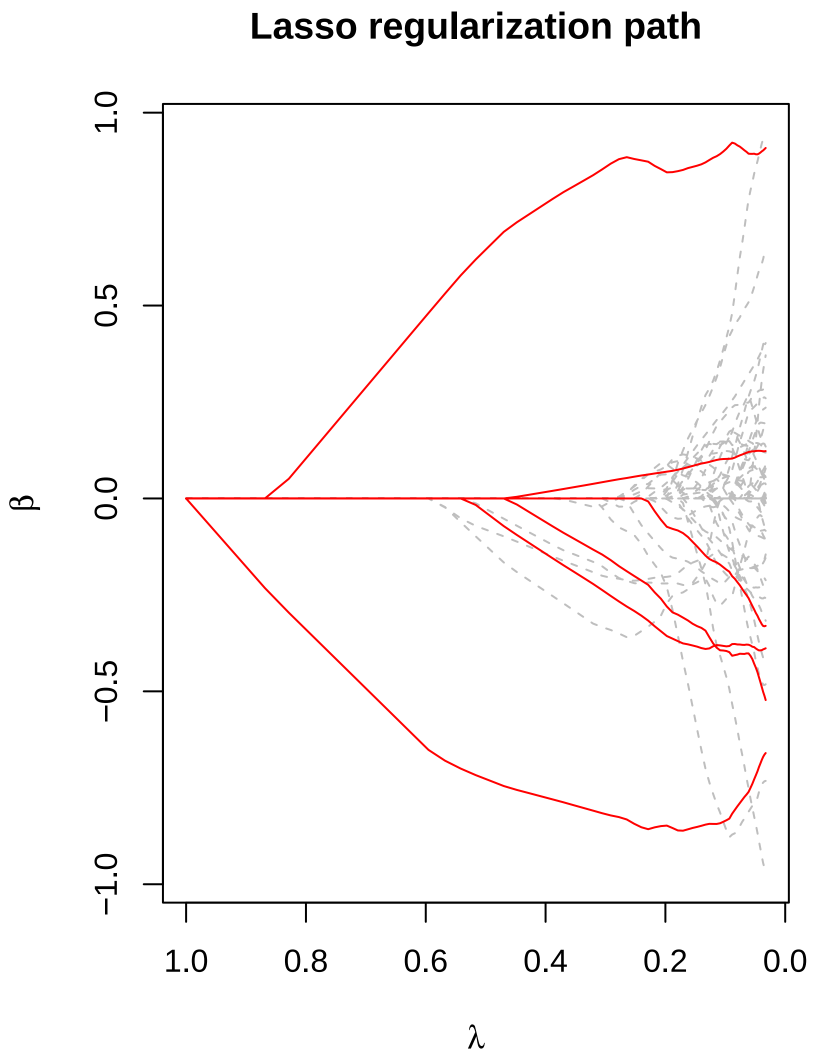
So one of the advantages of stability selection is that we do not need to manually select λ. Instead, we pre-define $q = \sqrt{0.8p}$, which specify the maximum number of non-zero variables to enter the model among all $\lambda \in \Lambda$, controlling the model complexity. The regularization region Λ is defined such that for all $\lambda \in \Lambda$, the number of non-zero coefficients of that fit on λ cannot exceed q.

After we obtain Λ, all values are rescaled so it has the interval [0, 1], where no variables will be selected at $\lambda = 1$ and q variables will be selected when $\lambda = 0$

```r
# error control to define regularization region of lambda
q <- sqrt(0.8*p)
coefs <- coef(fit_glm)[-1, ]  # remove intercept row
selected_counts <- apply(coefs, 2, function(col) sum(col != 0))
idx <- which(selected_counts >= q)[1]
lambda_region <- fit_glm$lambda[1:idx]  # from largest to chosen lambda

# cat("Variables selected at chosen lambda:", selected_counts[idx], "\n")
```

```r
# lambda rescale to [0, 1]
idx_null <- which(selected_counts == 0)[1]
lambda_null <- fit_glm$lambda[idx_null]      # null model lambda (max penalty)
lambda_min_full <- min(fit_glm$lambda)       # minimal lambda in full path

lambda_scaled <- (lambda_region - lambda_min_full)/(lambda_null - lambda_min_full)
lasso_lambda <- lambda_scaled

beta_mat <- as.matrix(fit_glm$beta)[, 1:idx] # p x l
beta_mat <- t(beta_mat) # transpose, l x p
combined_mat <- cbind(lasso_lambda, beta_mat)
```

The lasso regularization plot indicates that only 2 of the 6 unpermuted variables stands out while the rest are lost among the noise variables. It is showing that even though some variables are relevant ones, they would not be selected for some values of λs, such as $\lambda = 0.8$, visualizing the instability on feature selection.



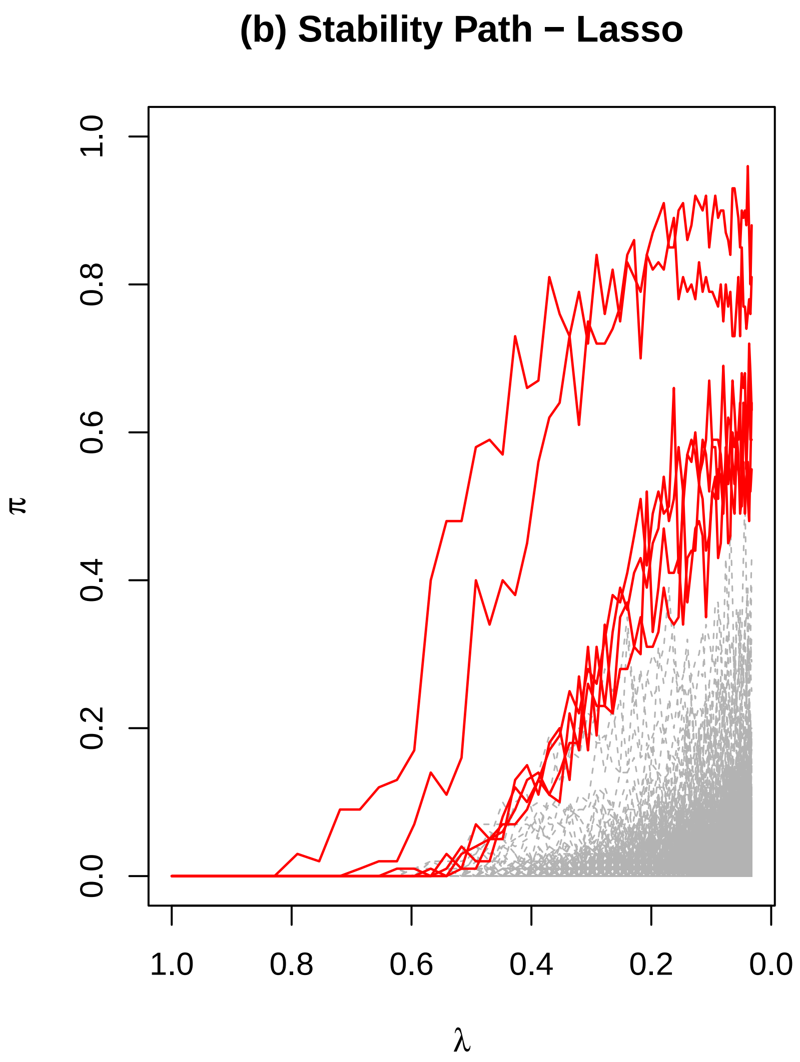
Stability Selection with LASSO

Here, we define `stability_lasso` to first loop through all λ and then loop through all subsamples I_b, where $|I_b| = \frac{n}{2}$ without replacement and perform lasso regression on I_b with λ.

The total number of fit is $p \times l$, where p is number of variables and $l = |\Lambda|$. Then the selection probabilities of k^{th} variable at λ is calculated, represent by π_k^{λ}, and stored in a `stab`$\in \mathbb{R}^{p\times l}$.

```r
stability_lasso <- function(X, y, lambda_seq, B = 100) {
  # X: n x p
  n <- nrow(X)
  p <- ncol(X)
  stab <- matrix(0, nrow = p, ncol = length(lambda_seq))
  # p x l, each row is coef of each gene of a lambda

  for (j in seq_along(lambda_seq)) { #traverse through each lambda
    lam <- lambda_seq[j]
    for (b in 1:B) { # loop through all subsamples
      idx <- sample(1:n, n/2, replace = FALSE)
      fit <- glmnet(X[idx,], y[idx], alpha = 1, lambda = lam,
                    standaridze= TRUE)
      stab[, j] <- stab[, j] + as.numeric(coef(fit)[-1] != 0)
      # update non-zero coef of all gene per lambda per sample
    }
  }
  return(stab / B)
}
```

The lasso stability path demonstrates the selection probabilities of variables at different regularization λ. From the plot, we can inspect that all 6 relevant genes stood out on top of all noise variables.



Stability selection with Randomzied Lasso

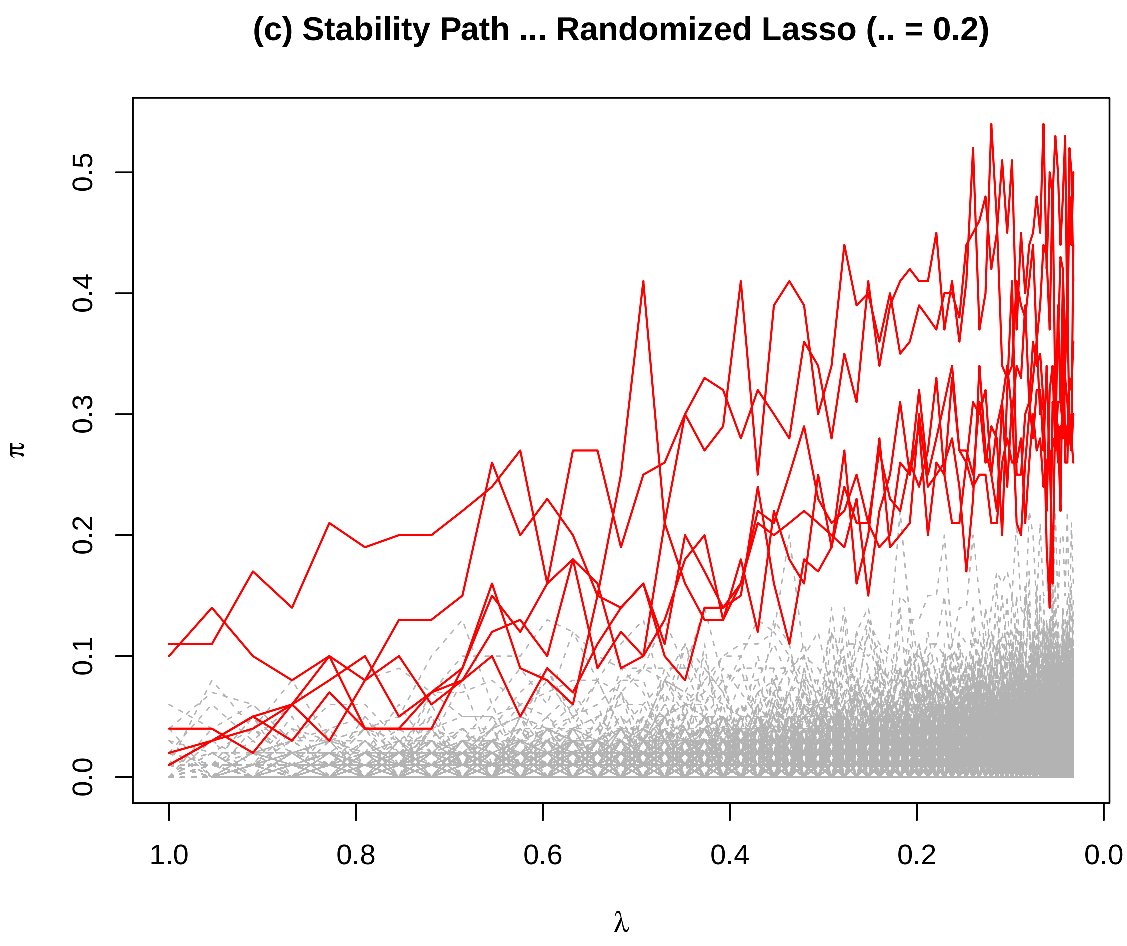
Instead of fitting a standard lasso regression, randomized lasso apply additional penalty weights, w_j^b on each variable j when fitting the model in each I_b. The penalty weights are randomly distributed in $Unif(\alpha, 1)$, and α is the predefined weakness level between 0 and 1, excluding 1. When $\alpha = 1$, it is just the standard lasso, because all variables are penalized by 1, which does not change anything.

When α is small, for example $\alpha = 0.2$, strong randomization is implemented, where variables are perturbed more significantly. This tends to break symmetry among correlated variables and prevents one variable from dominating. A large α (closer to 1) results in weak randomization and predictors are barely perturbed. Thus, correlated variables may compete and one may consistently outperform the others. In practice, α is usually chosen between 0.2 and 0.5.

Similarily, selection probabilities of each variable at different λ values are calculated once all subsamples are fitted.

```r
randomized_stability <- function(X, y, lambda_seq, B = 100, alpha = 0.2) {
  n <- nrow(X)
  p <- ncol(X)
  stab <- matrix(0, nrow = p, ncol = length(lambda_seq))

  for (j in seq_along(lambda_seq)) {
    lam <- lambda_seq[j]
    for (b in 1:B) {
      idx <- sample(1:n, n/2, replace = FALSE)

      # Random weights each predictors
      w <- runif(p, alpha, 1)
      fit <- glmnet(X[idx,], y[idx], alpha = 1, lambda = lam,
                    standardize = TRUE, penalty.factor = w)
      stab[, j] <- stab[, j] + as.numeric(coef(fit)[-1] != 0)
    }
  }
  return(stab / B)
}
```

The randomized lasso stability path further separates the signal variables from noise variables and only unpermuted variables are included in the model at the very beginning of regularization grid. This highlights that the model can accurately locate the signal variables very quickly, indicating increased accuracy and stability for feature selections.



Differences in reproducing results

Even though I am using the same dataset as the paper used, the sample size of the dataset in paper is larger than mine. In addition, it may also due to software differences because the software setup is not provided.

Extension study on numerical results

In reality, it is nearly impossible to find perfect dataset that satisfy all regression assumptions and exchangeability assumption, such as equal variance for all variables and zero correlation among all variables. The violation on assumption would negatively impact the performance of lasso regression since it penalizes coefficients rather than standardized effects. Thus, I want to examine the robustness of stability selection of lasso and randomized lasso on misspecified simulated data that is imperfect with correlated predictors X_1 and X_2.

Simulation setting

- $p = 2000$, $n = 50$
- Relevant predictors: X_1, X_2, X_3, X_4, Noise predictors: $X_5, ..., X_{2000}$
- $\beta = [1, 1, 1, 1, 0, ..., 0]$, $\epsilon \sim N(0, \frac{1}{4})$,
- $Y = X\beta + \epsilon$
- $var(X_1) = 1.2 \neq var(X_2) = 0.5 \neq var(X_3) = 1 = ... = var(X_{2000}) = 1$
- $corr(X_1, X_2) = \rho = 0.8, cov(X_1, X_2) = 0.6196773 \approx 0.62$

Due to above predefined setting, the covariance matrix Σ can be derived and fit into the simulation function.

$$\Sigma = \begin{bmatrix} 1.2 & 0.62 & 0 & 0 & ... & 0 \\ 0.62 & 0.5 & 0 & 0 & ... & 0 \\ 0 & 0 & 1 & 0 & ... & 0 \\ 0 & 0 & 0 & 1 & ... & 0 \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

X is simulated using `mvrnorm` function with `mns` vector containing all means of variables and `sigma` is a covariance matrix defined by `make_sigma`. Y is a vector of all response values $Y = X\beta + \epsilon$.

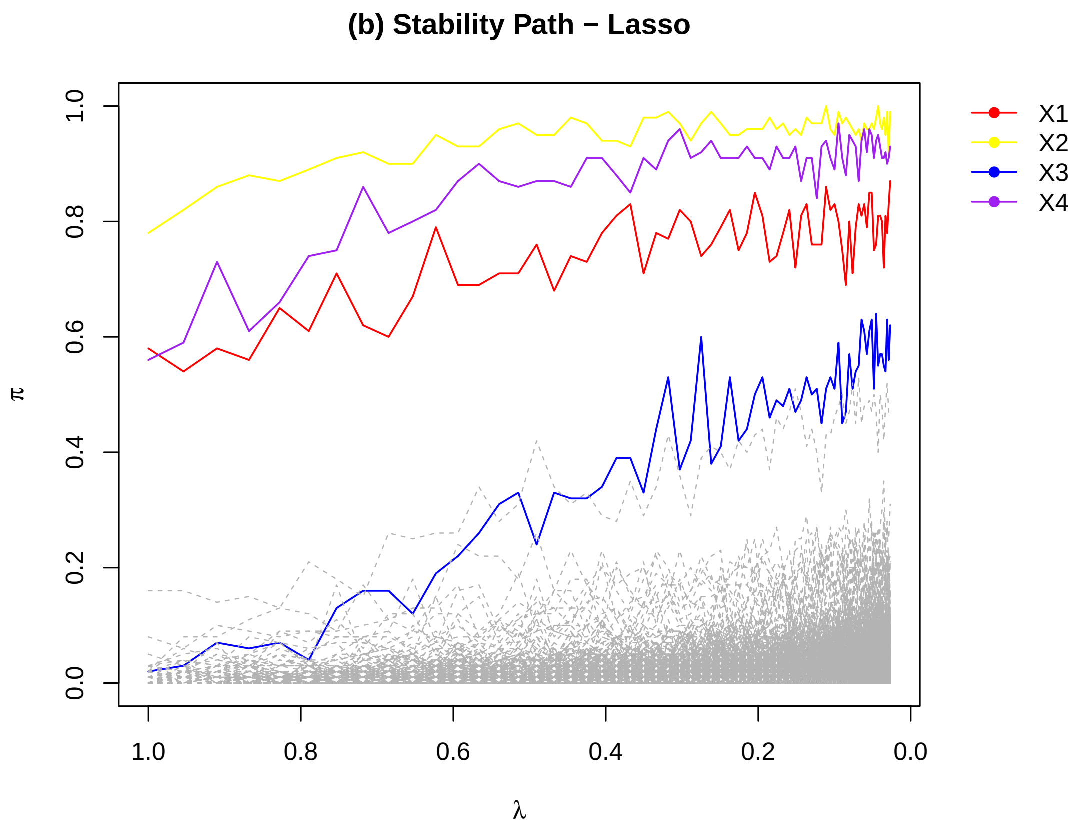
In order to regenerate different simulated data for different scenarios, `simulate_data` is a function that takes $n, p, \rho, var(X_1), Var(X_2)$.

In this section, I want to examine the impact of 1) value of correlation $\rho \in [-1, 1]$ and 2) the magnitude of variances on correlated variables on stability selection performance, as lasso is sensitive to the variances.

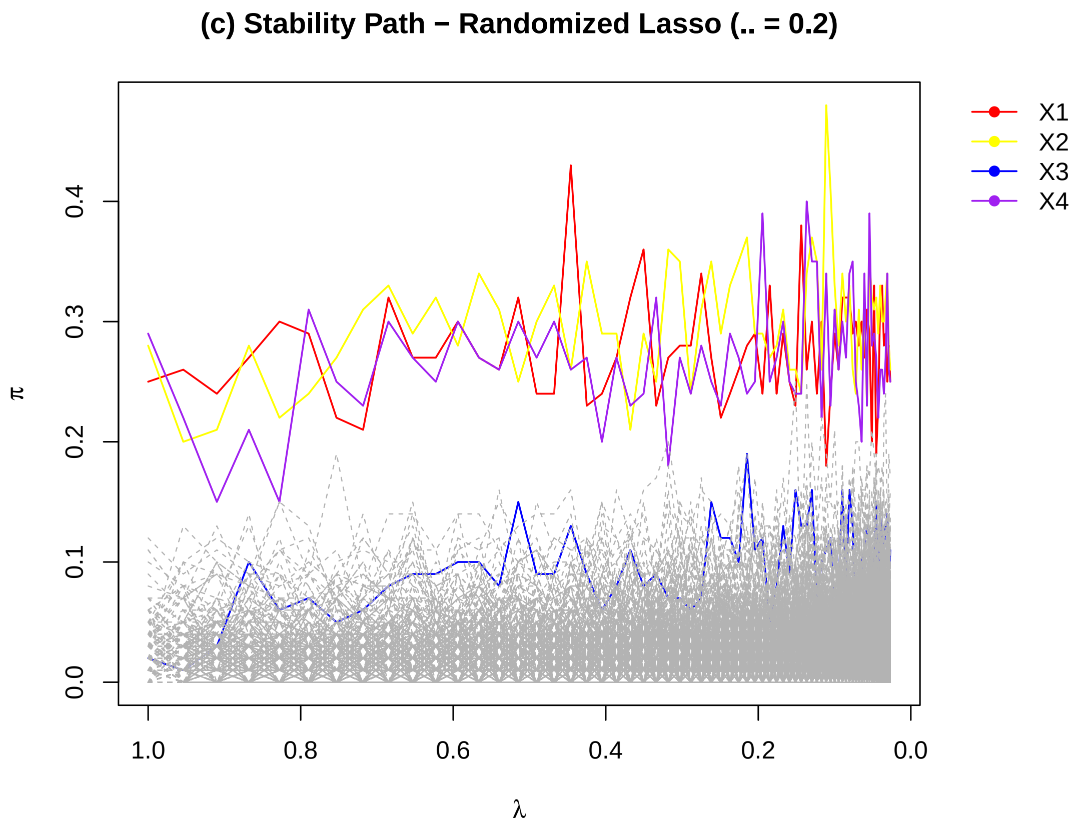
```r
make_sigma <- function(p, rho, v1 = 1, v2 = 1){
  corr <- diag(p)
  corr[1,2] <- rho * sqrt(v1*v2)
  corr[2,1] <- corr[1,2]
  corr[1,1] <- v1
  corr[2,2] <- v2
  return(corr)
}
```

```r
simulate_data <- function(n, p, rho_, var_x1, var_x2, signals){
    # epsilon, noise simulation, epsilon - N(0, 1/4), var = 1/2
    eps <- rnorm(n, 0, sd = 1/4)

    # beta simulation
    beta <- rep(0, p)
    for (i in 1:signals) { beta[i] = 1 }

    # predictor values simulation
    mns <- rep(0, p) # mean of each predictor
    var_x1 <- 1.2; var_x2 <- 0.5
    sigma <- make_sigma(p, rho = rho_, v1 = var_x1, v2 = var_x2)
    # covariance matrix of predictors
    sim_X <- mvrnorm(n, mu = mns, Sigma = sigma) # n x p
    colnames(sim_X) <- paste0("X", 1:p)

    # response values simulation
    sim_y <- as.numeric(sim_X %*% beta + eps)

    # first fit
    fit_sim <- glmnet(sim_X, sim_y, alpha = 1, standardize = FALSE)

    # error control to define regularization region of lambda
    sim_q <- sqrt(0.8*p)
    sim_coefs <- coef(fit_sim)[-1, ]  # remove intercepts
    sim_selected_counts <- apply(sim_coefs, 2, function(col) sum(col != 0))
    sim_idx <- which(sim_selected_counts >= sim_q)[1]
    sim_lambda_region <- fit_sim$lambda[1:sim_idx]

    # lambda rescale to [0, 1]
    sim_idx_null <- which(sim_selected_counts == 0)[1]
    sim_lambda_null <- fit_sim$lambda[sim_idx_null]# max penalty
    sim_lambda_min_full <- min(fit_sim$lambda) # minimal lambda in full path

    sim_lambda_scaled <- (sim_lambda_region - sim_lambda_min_full)/(sim_lambda_null - sim_la
    sim_lambda_seq <- sim_lambda_scaled

    return(list(X = sim_X, y = sim_y, lambda_sq = sim_lambda_seq))
}
```

```
set.seed(123)
p_sim <- 1000 # num of predictors
n_sim <- 50 # num of samples
rho_ <- 0.8 # correlation value between x1 and x2, range from -1 to 1
signals <- 4 # number of signal varaible, aka relevant variable
var_x1 <- 1.2
var_x2 <- 0.5
results1 <- simulate_data(n_sim, p_sim, rho_, var_x1, var_x2, signals)
sim_X <- results1$X; sim_y <- results1$y
sim_lambda_seq <- results1$lambda_sq # regularization region
```



In the above lasso stability path, all four relevant variables stands out and is included as part of the model at very early stage of regularization path. However, the high correlation between X_1 and X_2 reduced the stability on one of the variables which is X_1 in this run. The results presented here is different from the result I presented in the slide because standardization is applied but not in previous setting. In addition, there is one noise variable that has the similar path as X_3, meaning that this variable is consistently included in the model which is not expected. The rest of the noise variables behaves expectedly with low selection probabilities along the regularization path.

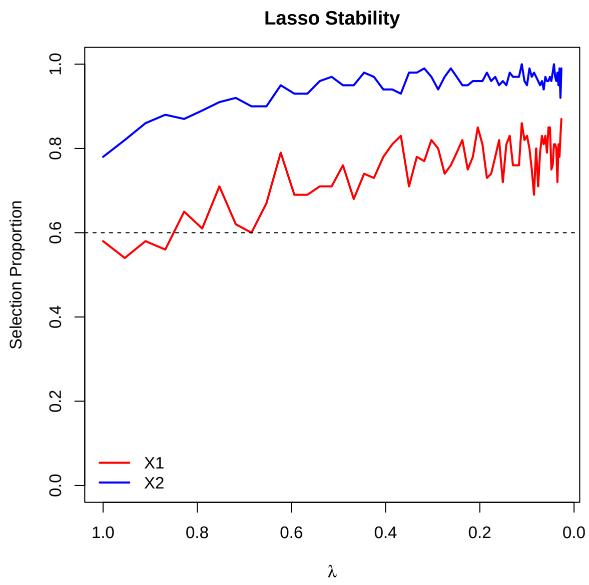


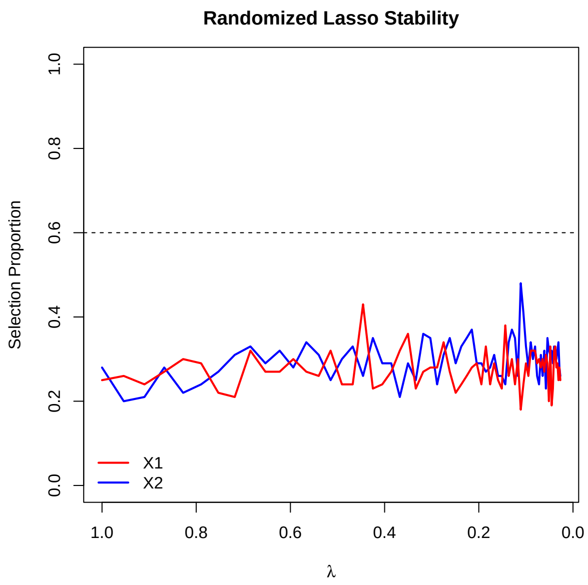
In the above randomized lasso stability path, the selection probabilities are low overall but more stable and smoother among the regularization path, while X_3 is blended among the noise variables, indicating that the effect of X_3 is relatively weak compared to other 3 signal variables. Also, the gap between X_1 and X_2 reduces significantly, alleviating the correlational relationship between these two variables.

Proportion X_1 selected vs X_2 selected

The below code and graph specifically looks at stabilities path on X_1 and X_2, and the reduction in gap is more obvious, where in randomized lasso, two paths essentially overlap each other.

```r
par(mfrow = c(1, 2),mar = c(4, 4, 3, 1))
# proportion in standard lasso
x1_prop_lasso <- sim_stab_lasso[1,]; x2_prop_lasso <- sim_stab_lasso[2,]

# visualization
plot(sim_lambda_seq, x2_prop_lasso, type = "l",ylab = "Selection Proportion",
     ylim = c(0, 1), lwd = 2, col = "blue", xlab = expression(lambda),
     xlim = rev(range(sim_lambda_seq)),
     main = "Lasso Stability")

lines(sim_lambda_seq, x1_prop_lasso,lwd = 2, col = "red")
abline(h = 0.6, lty = 2)
legend("bottomleft", legend = c("X1", "X2"),col = c("red", "blue"),
       lwd = 2, bty = "n")

# proportion in randomized lasso
x1_prop_rand_lasso <- sim_stab_rand[1,]; x2_prop_rand_lasso <- sim_stab_rand[2,]

# visualization
plot(sim_lambda_seq, x2_prop_rand_lasso, type = "l",ylab = "Selection Proportion",
     ylim = c(0, 1), lwd = 2, col = "blue", xlab =expression(lambda),
     xlim = rev(range(sim_lambda_seq)),
     main = "Randomized Lasso Stability")

# Add X1 (flat line at 1)
lines(sim_lambda_seq, x1_prop_rand_lasso,lwd = 2, col = "red")
# Add stability threshold line
abline(h = 0.6, lty = 2)
legend("bottomleft", legend = c("X1", "X2"), col = c("red", "blue"),
       lwd = 2, bty = "n")
```





Comparison of different ρ

In this subsection, I vary the values of correlation $\rho = 0.2, 0.6, 1$. A smaller ρ values indicate weak correlation between the correlated variables, 0.6 indicate moderate correlation, and 1 indicate perfect correlation. When $\rho = 1$, the encoded information in X_1 and X_2 is overlapping, suggesting that they are likely to have the same effect on response variables.

Because ρ is changed, the dataset are re-simulated, which will result in different result compared to previous conclusion.

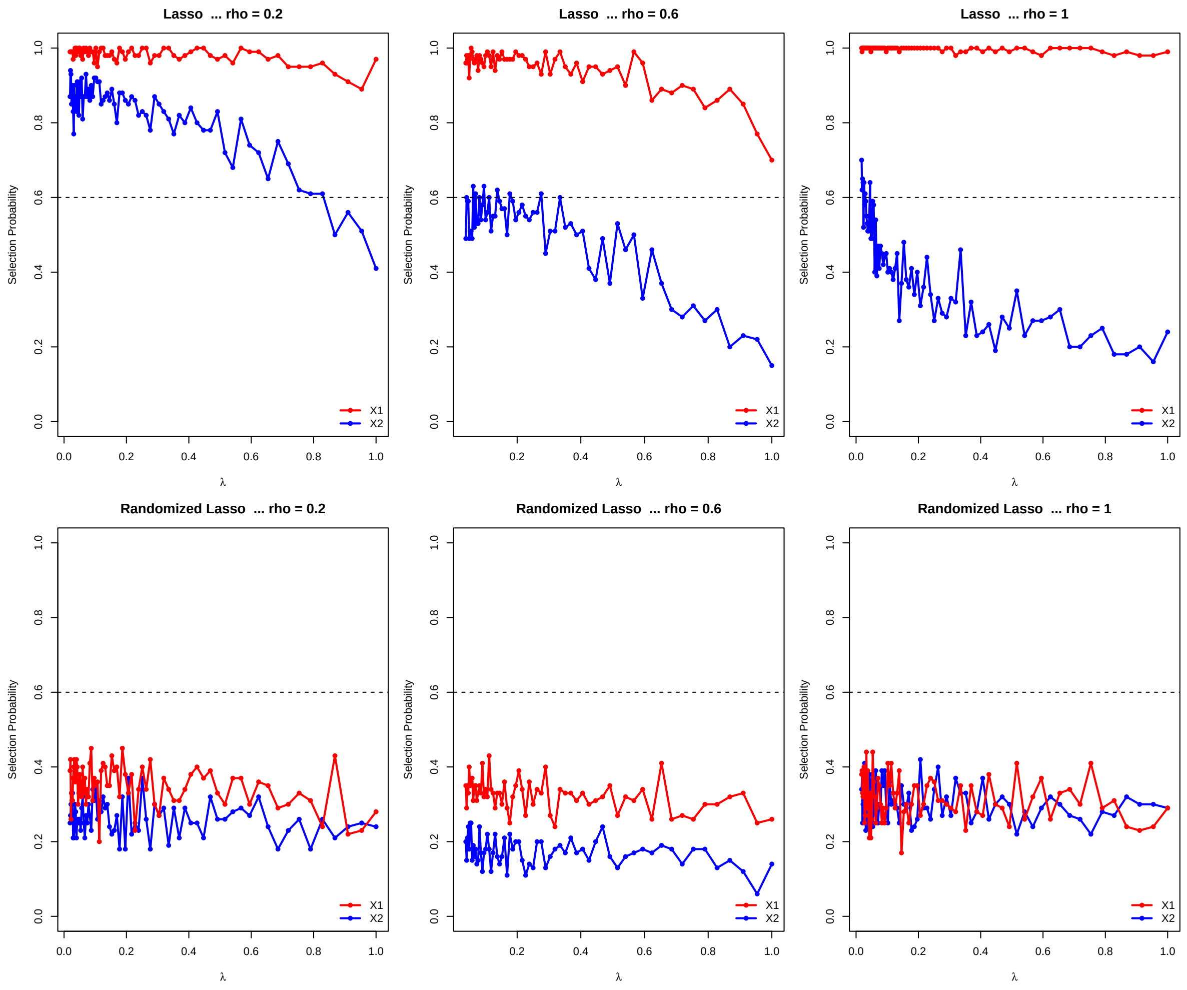
```r
rho_vals <- c(0.2, 0.6, 1)
simulation_results <- list()

for (r in rho_vals) {
  results <- simulate_data(n = n_sim, p = p_sim, rho_ = r, var_x1 = var_x1,
                           var_x2 = var_x2, signals = signals)

  X_sim <- results$X
  y_sim <- results$y
  lambda_seq0 <- results$lambda_sq

  stab_lasso <- stability_lasso(X_sim, y_sim, lambda_seq0)
  stab_random <- randomized_stability(X_sim, y_sim, lambda_seq0, alpha = 0.2)

  simulation_results[[paste0("rho_", r)]] <- list(
    X = X_sim,
    y = y_sim,
    lambda_seq1 = lambda_seq0,
    stab_lasso = stab_lasso,
    stab_random = stab_random
  )
}

res02 <- simulation_results$rho_0.2
stab_lasso_02 <- res02$stab_lasso
stab_random_02 <- res02$stab_random

res06 <- simulation_results$rho_0.6
stab_lasso_06 <- res06$stab_lasso
stab_random_06 <- res06$stab_random

res1 <- simulation_results$rho_1
stab_lasso_1 <- res1$stab_lasso
stab_random_1 <- res1$stab_random
```

In the below figure where three different ρ are tested on both standard lasso and randomized lasso, the first row are stability paths from the lasso and the second row presents stability path fitted on the randomized lasso. Throughout all lasso path, X_1 has dominate selection probabilities over the regularization values while X_2's selection probabilities is very unstable, and the gap of selection probabilities exaggerates as $\rho \to 1$. This is likely caused by the increases in overlapping representation on response variable, and thus, Lasso would tend to select only one variable in the correlated pair.

Not surprisingly, in the randomized lasso stability path, the gap between X_1 and X_2 is much smaller than that in standard lass due to the additional randomized penalty. The penalty weights equalize the selection probabilities and minimize the dominance effect of one variable. In addition, there is a trade-off between selecting X_1 and X_2 that when X_1 is chosen with higher percentage, X_2 is chosen less along the regularization grid. As X_1 and X_2 have more common information, randomized lasso just arbitrarily choose one of them.

```
par(mfrow = c(2, 3), mar = c(4,4,3,1))

# lasso
plot_stability_two_vars(lambdas = res02$lambda_seq1, stab = stab_lasso_02,
                        rho_value = 0.2, method_name = "Lasso")
plot_stability_two_vars(lambdas = res06$lambda_seq1, stab = stab_lasso_06,
                        rho_value = 0.6,method_name = "Lasso")
```

```r
plot_stability_two_vars(lambdas = res1$lambda_seq1, stab = stab_lasso_1,
                        rho_value = 1, method_name = "Lasso")
# randomized lasso
plot_stability_two_vars(lambdas = res02$lambda_seq1, stab = stab_random_02,
                        rho_value = 0.2, method_name = "Randomized Lasso")
plot_stability_two_vars(lambdas = res06$lambda_seq1,stab = stab_random_06,
                        rho_value = 0.6, method_name = "Randomized Lasso")
plot_stability_two_vars(lambdas = res1$lambda_seq1, stab = stab_random_1,
                        rho_value = 1,method_name = "Randomized Lasso")
```



Comparison of magnitude of variances

In the last section, I want to examine whether the magnitude of variances would affect the stability selection performance, by only changing the variance of X_1 while $var(X_2) = 0.5$ among three test cases.

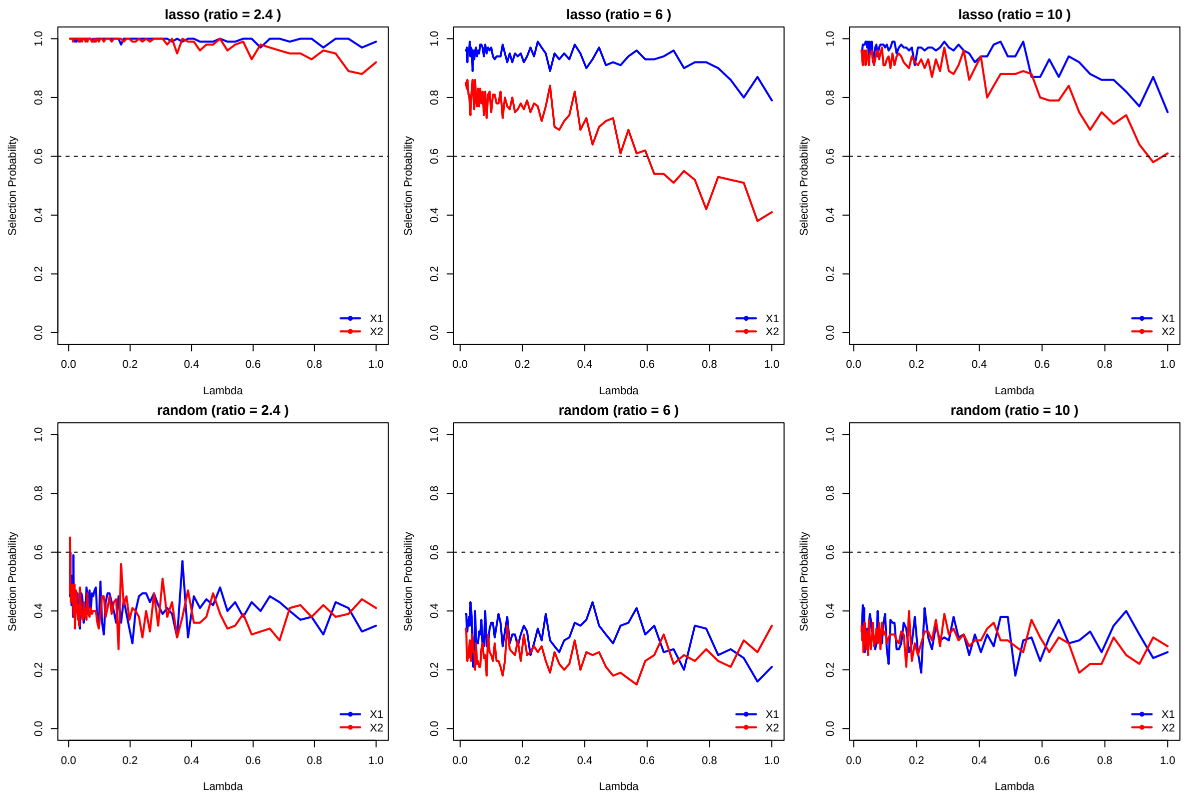
- $var(X_1) = 1.2, var(X_2) = 0.5, ratio = 2.4$
- $var(X_1) = 3, var(X_2) = 0.5, ratio = 6$
- $var(X_1) = 5, var(X_2) = 0.5, ratio = 10$

The variance ration is equal to $\frac{var(X_1)}{var(X_2)}$ and all ratios are greater than 1 that $var(X_1) > var(X_2)$. The data are also re-simulated.

```r
#specify different variances to x1 and x2; focus on the ratio x1/x2 only
var_set <- list(c(1.2, 0.5), c(3.0, 0.5), c(5.0, 0.5))
results_var <- list()
rho1 <- 0.8
i <- 1
for (vars in var_set) {
  vx1 <- vars[1]; vx2 <- vars[2]
  res <- simulate_data(n = n_sim, p = p_sim, rho_ = rho1, var_x1 = vx1,
                       var_x2 = vx2, signals = signals)
  X_sim <- res$X
  y_sim <- res$y
  lambda_seq2 <- res$lambda_sq
  stab_lasso <- stability_lasso(X_sim, y_sim, lambda_seq2)
  stab_random <- randomized_stability(X_sim, y_sim, lambda_seq2, alpha = 0.2)

  results_var[[i]] <- list(lambda_ = lambda_seq2, var_ratio = vx1/vx2,
                           stab_lasso = stab_lasso, stab_random = stab_random)
  i <- i + 1
}
```

In the below figure present six different plots that test different variance ratio on both standard lasso and randomized lasso, at $\rho = 0.8$. In standard lasso, as the variance ratio increases, both signal variables can be included in the stable set but selection probabilities are less stable for both variables; in the topleft plot, the two paths are stable across the regularization grid, whereas in the middle and right plots of the first row, the selection path has a downward trend over the regularization values. Specifically, when ratio = 6, the gap is the largest, in correlated pair, indicating that high-variance variable are more stable as they are penalized less heavily, leading to biased selection.

Randomized lasso shows better consistency in selecting both variables, alleviating the effect from variance magnitude; however, except in ratio $= 2.4$ that X_1 can be included in the stable set, none of the correlated pair can pass π_{thr} to enter the model and be considered to the signal variable even though they are true signals.



Conclusion

Stability selection is a powerful framework for identifying true signal variables in high-dimensional settings, demonstrating robustness across varying correlation structures and variance imbalances. The implementation on the riboflavin dataset reveals that both standard lasso and randomized lasso maintain high selection probabilities (j 1 j 1) for true signal variables across different regularization parameters, effectively distinguishing signals from noise. In the last extended comparison, when variables have heterogeneous variances, randomized lasso outperforms standard lasso by keeping both true signals more equally selected, whereas standard lasso exhibits a larger gap between signal variables due to its scale-sensitive penalty. This advantage becomes particularly apparent at extreme variance ratios, suggesting that randomized lasso can mitigate bias caused by variable heterogeneity via randomized penalty weights.

For practitioners analyzing complex biological datasets, especially gene expression datasets, stability selection coupled with randomized penalties is an effective approach to identify relationships between responses and signal variables while maintaining strict control over false discoveries. Future work should explore the interplay with optimal threshold selection to further refine stability selection for robust feature selection.

However, some limitations should be considered. The application is computationally expensive: when p is large, a single lasso fit on that data takes longer to run, and repeated subsampling would make it even worse. Another noticeable issue is that weak signals can fail to be included in the stable set even when they do not exceed the threshold, and thus are missed even if they are relevant. This technique also relies on the exchangeability assumption that the selection probabilities of all noise variables are equal, which is hard to satisfy. When this assumption is violated, meaning that some noise variables have higher selection probabilities, it might lead to biased selections. Finally, the standard lasso struggles with correlated variables, leading to unstable selections.

Citation

Nicolai Meinshausen, Peter Bühlmann, Stability Selection, Journal of the Royal Statistical Society Series B: Statistical Methodology, Volume 72, Issue 4, September 2010, Pages 417–473, https://doi.org/10.1111/j.1467-9868.2010.00740.x

Bühlmann, P., Kalisch, M. and Meier, L. (2014) *High-dimensional statistics with a view towards applications in biology.* Annual Review of Statistics and its Applications **1**, 255–278