**FarmTest: An R Package for Factor-Adjusted Robust Multiple Testing**

by Koushiki Bose, Jianqing Fan, Yuan Ke, Xiaou Pan and Wen-Xin Zhou

**Abstract** We provide a publicly available library FarmTest in the R programming system. This library implements a factor-adjusted robust multiple testing principle proposed by Fan et al. (2019) for large-scale simultaneous inference on mean effects. We use a multi-factor model to explicitly capture the dependence among a large pool of variables. Three types of factors are considered: observable, latent, and a mixture of observable and latent factors. The non-factor case, which corresponds to standard multiple mean testing under weak dependence, is also included. The library implements a series of adaptive Huber methods integrated with fast data-driven tuning schemes to estimate model parameters and to construct test statistics that are robust against heavy-tailed and asymmetric error distributions. Extensions to two-sample multiple mean testing problems are also discussed. The results of some simulation experiments and a real data analysis are reported.

**Introduction**

In the era of big data, large-scale multiple testing problems arise from a wide range of fields, including biological sciences such as genomics and neuroimaging, social science, signal processing, marketing analytics, and financial economics. When testing multidimensional statistical hypotheses simultaneously, researchers appreciate statistically significant evidence against the null hypothesis with a guarantee of controlled false discovery rate (FDR) (Benjamini and Hochberg, 1995). Since the seminal work of Benjamini and Hochberg (1995), multiple testing with FDR control has been extensively studied and successfully used in many applications. Most of the existing testing procedures are tailored to independent or weakly dependent hypotheses or tests. See, Storey (2002), Genovese and Wasserman (2004) and Lehmann and Romano (2005), to name a few. The independence assumption, however, is restricted in real applications as correlation effects are ubiquitous in high dimensional measurements. Ignoring such strong dependency and directly applying standard FDR controlling procedures can lead to inaccurate false discovery control, loss of statistical power, and unreliable scientific conclusions.

Over the past decade, a multi-factor model has proven to be an effective tool for modeling cross-sectional dependence, with applications in genomics, neuroscience, and financial economics. Related references in the context of multiple testing include Leek and Storey (2008), Fritguet et al. (2009), Fan et al. (2012), Desai and Storey (2011) and Fan and Han (2017). A common thread of the aforementioned works is that the construction of test statistics and p-values heavily relies on the assumed joint normality of factors and noise, which is arguably another folklore regarding high dimensional data. Therefore, it is imperative to develop large-scale multiple testing tools that adjust cross-sectional dependence properly and are robust to heavy-tailedness at the same time.

Recently, Fan et al. (2019) developed a Factor-Adjusted Robust Multiple Test (FarmTest) procedure for large-scale simultaneous inference with highly correlated and heavy-tailed data. Their emphasis is on achieving robustness against both strong cross-sectional dependence and heavy-tailed sampling distribution. Specifically, let $X = (X_1, \ldots, X_p)^T$ be a random vector with mean $\mu = (\mu_1, \ldots, \mu_p)^T$. We are interested in testing the $p$ hypotheses $H_{0j}: \mu_j = 0$, and wish to find a multiple comparison procedure to test individual hypotheses while controlling the FDR. The FarmTest method models the dependency among $X_j$’s through an approximate multi-factor model, namely $X_j = \mu_j + b_j f + u_j$, where $f$ is a zero-mean random vector capturing the dependence structure of $X$. The method applies to either observable or unobservable factor $f$. The former includes the non-factor case which corresponds to the standard multiple mean testing problem. For the latter, we estimate the factors in a data-driven way. Test statistics are then calculated by subtracting out the realized common factors. Multiple comparisons are then applied to these weakly dependent factor-adjusted test statistics. Also, adjusting the factors before testing reduces signal-to-noise ratios, which enhances statistical power. Since a data-driven eigenvalue ratio method is used to estimate the number of (latent) factors, the testing procedure still works when the dependence is weak and therefore is rather flexible.

This article describes an R library named FarmTest, which implements the FarmTest procedure(s) developed in Fan et al. (2019). It is a user-friendly tool to conduct large-scale hypothesis testing, especially when one or several of the following scenarios are present: the dimensionality is far larger than the sample size available; the data is heavy-tailed and/or asymmetric; there is strong cross-sectional dependence among the data. FarmTest is implemented using the Armadillo library (Sanderson and Curtin, 2016) with Rcpp interfaces (Eddelbuettel and Francois, 2011; Eddelbuettel and Sanderson, 2014). A simple call of FarmTest package only requires the input of a data matrix and the
null hypotheses to be tested. It outputs the hypotheses that are rejected, along with the p-values and some estimated parameters which may be of use in further analysis. Testing can be carried out for both one-sample and two-sample problems.

Another key feature of our package is that it implements several recently developed robust methods for fitting regression models (Zhou et al., 2018; Sun et al., 2020) and covariance estimation (Ke et al., 2019). When data is generated from a heavy-tailed distribution, test statistics that are based on the least-squares method are sensitive to outliers, which often causes significant false discoveries and suboptimal power (Zhou et al., 2018). The effect of heavy-tailedness is amplified by high dimensionality; even moderate-tailed distributions can generate very large outliers by chance, making it difficult to separate the true signals from spurious variables. As a result, large-scale multiple testing based on non-robust statistics may engender an excessive false discovery rate, which arguably is one of the causes of the current crisis in reproducibility in science. Moreover, to choose the multiple tuning parameters in robust regression and covariance estimation, we employ the recently developed data-driven procedures (Wang et al., 2020; Ke et al., 2019), which are particularly designed for adaptive Huber regression and are considerably faster than the cross-validation method used in Fan et al. (2019).

We further remark that most existing multiple testing R packages do not address the robustness against both heavy-tailed distribution and strong dependence. The hypothesis testing function in R, named t.test, neither adjusts for strong dependence in the data nor estimates the parameters in focus robustly. The built-in function p.adjust or the package qvalue (Storey, 2002) only adjust user-input p-values for multiple testing and do not address the problem of estimating the p-values themselves. The package multcomp (Hothorn et al., 2008) provides simultaneous testing tools for general linear hypotheses in parametric models under the assumptions that the central limit theorem holds. The package mutltest (Pollard et al., 2005) is developed to implement non-parametric bootstrap and permutation resampling-based multiple testing procedures. The mutltest can calculate test statistics based on ranked data which is robust against outliers but yields biased mean estimators. In addition, mutltest cannot explicitly model the dependence structure in data. The package mutoss is designed to apply many existing multiple hypothesis testing procedures with FDR control and p-value correction. Nevertheless, none of the tools in mutoss is suitable to deal with both strong dependency and heavy-tailedness. Moreover, existing packages are often difficult to navigate since users need to combine many functions to perform multiple tests.

**Factor-adjusted robust multiple testing**

In this section, we revisit the problem of simultaneous inference on the mean effects under a factor model and discuss the main ideas behind the FarmTest method developed by Fan et al. (2019).

**Multiple testing with false discovery rate control**

Suppose we observe \( n \) independent data vectors \( X_1, \ldots, X_n \) from a \( p \)-dimensional random vector \( X = (X_1, \ldots, X_p)^T \). Further, let \( \mu = (\mu_1, \ldots, \mu_p)^T \) and \( \Sigma = \left( \sigma_{jk} \right)_{1 \leq j, k \leq p} \) denote the mean vector and covariance matrix of \( X \), respectively. In the language of hypothesis testing, we are interested in one of the following three types of hypotheses:

\[
\begin{align*}
H_{0j} & : \mu_j = h^0_j \quad \text{versus} \quad H_{1j} : \mu_j \neq h^0_j; \\
H_{0j} & : \mu_j \leq h^0_j \quad \text{versus} \quad H_{1j} : \mu_j > h^0_j; \\
H_{0j} & : \mu_j \geq h^0_j \quad \text{versus} \quad H_{1j} : \mu_j < h^0_j;
\end{align*}
\]

for \( j = 1, \ldots, p \). In the default setting, \( h^0_j = 0 \) for all \( j \).

Here we take the two-sided test (1) as an example to discuss the false discovery rate (FDR) control. For \( 1 \leq j \leq p \), let \( T_j \) be a generic test statistic for the \( j \)th hypothesis. Given a prespecified threshold \( z > 0 \), we reject the \( j \)th null hypothesis if \( |T_j| \geq z \). The FDR is defined as the expected value of the false discovery proportion (FDP): \( \text{FDR}(z) = \mathbb{E} \{ \text{FDP}(z) \} \) with \( \text{FDP}(z) = V(z)/\max \{ R(z), 1 \} \), where \( R(z) = \sum_{j=1}^{p} 1 \left( |T_j| \geq z \right) \) is the number of total rejections and \( V(z) = \sum_{j=1}^{p} 1 \left( |T_j| \geq z \right) \) is the number of false discoveries. If the FDP \( (z) \) were known, the rejection threshold will be \( z_{0} = \inf \{ z \geq 0 : \text{FDP}(z) \leq a \} \) in order to achieve FDP control. Notice that \( R(z) \) is observable given the data while \( V(z) \) is an unobserved random quantity that needs to be estimated.

Assume that there are \( p_0 = \pi_0 p \) true nulls and \( p_1 = (1 - \pi_0) p \) true alternatives. Suppose the constructed test statistic \( T_j \) is close in distribution to standard normal for every \( j = 1, \ldots, p \), if the test statistics are weakly dependent. Heuristically the number of false discoveries \( V(z) \) is close to
2p_0\Phi(-z) for any z \geq 0. A conservative way is to replace V(z) by 2p\Phi(-z). Assuming the normal approximation is sufficiently accurate, 2p\Phi(-z) provides an overestimate of the number of false discoveries, resulting in an underestimate of the FDP(z). A more accurate method is to estimate the unknown proportion of null hypotheses \( \pi_0 = p_0/p \) from the data. Let \( \{ P_j = 2\Phi(-|T_j|) \}_{j=1}^p \) be the approximate p-values. For a predetermined \( \lambda \in [0,1) \), Storey (2002) suggest to estimate \( \pi_0 \) by \( \hat{\pi}_0(\lambda) = \{(1-\lambda)p\}^{-1}\sum_{j=1}^p 1\{P_j > \lambda\} \), because larger p-values are more likely to come from the true null hypotheses. Consequently, a data-driven rejection threshold is \( \hat{\varepsilon}_n = \inf \{z > 0: F\text{FDP}(z) \leq \alpha\} \), where\( F\text{FDP}(z) = 2\hat{\pi}_0(\lambda)p\Phi(-z)/R(z) \).

Factor-adjusted test statistics

In this section, we discuss the construction of test statistics under strong cross-sectional dependency captured by common factors. Specifically, we allow the \( p \) coordinates of \( X \) to be strongly correlated through an approximate factor model of the form \( X = \mu + Bf + u \), where \( B = (b_1, \ldots, b_p)^T \in \mathbb{R}^{p \times K} \) represents the factor loading matrix, \( f = (f_1, \ldots, f_K)^T \in \mathbb{R}^K \) is the common factor, and \( u = (u_1, \ldots, u_p)^T \in \mathbb{R}^p \) denotes a vector of idiosyncratic errors uncorrelated with \( f \). The observed samples thus follow

\[
X_i = \mu + Bf_i + u_i, \quad i = 1, \ldots, n, \tag{4}
\]

where \((f_i, u_i)'s\) are independent copies of \((f, u)\). Assume that both \( f \) and \( u \) have zero means. Further, denote by \( \Sigma_f \) and \( \Sigma_u = \{\sigma_{u,jk}\}_{1 \leq j,k \leq p} \) the covariance matrices of \( f \) and \( u \), respectively.

Our package allows the common factor \( f \) to be either observable or unobservable. In the former case, we observe \((X_i, f_i)_{i=1}^n\) so that model (4) is reduced to a multi-response linear regression problem; for the latter, we only observe \( \{X_i\}_{i=1}^n \) and therefore need to recover the latent factors. The latent factor model has identifiability issues; see Bai and Li (2012) for a set of possible solutions. For simplicity, we assume that \( \Sigma_f = I_K \) and \( B^TB \) is diagonal.

Robust estimation

As another key feature, the FarmTest method is robust against heavy-tailed sampling distributions. Under such scenarios, the ordinary least squares estimators can be suboptimal. Recently, Fan et al. (2017) and Sun et al. (2020) proposed the adaptive Huber regression method, the core of which is Huber’s M-estimator (Huber, 1964) with a properly calibrated robustification parameter that adapts to the sample size, dimensionality and noise level. They showed that the adaptive Huber estimator admits a sub-Gaussian-type deviation bound under mild moment conditions. This package exploits this approach to estimate the unknown parameters and to construct test statistics.

Algorithms

In this section, we formally describe the algorithms for the FarmTest procedure. We revisit and discuss procedures for the two scenarios with observable and unobservable/latent factors (Zhou et al., 2018; Fan et al., 2019). Notice that the two scenarios are inherently different in terms of estimating unknown parameters and constructing test statistics. Moreover, the selection of tuning parameters is based on the recent methods proposed by Ke et al. (2019) and Wang et al. (2020).

Observable factors

Suppose we observe independent data vectors \( \{(X_i, f_i)\}_{i=1}^n \) from model (4). The testing procedure for the hypotheses in (1)–(3) is described in Algorithm 1. Algorithm 1 automatically selects the robustification parameters \( \{\tau_j, \nu\}_{j=1}^p \) following the data-driven method proposed by Ke et al. (2019). See the Section of Selection of tuning parameters for more details. To enhance the finite sample performance, alternatively we can use the weighted/multiplier bootstrap (Zhou et al., 2018; Chen and Zhou, 2019) to compute p-values for all the marginal hypotheses. For \( b = 1, \ldots, B \), we obtain the corresponding bootstrap draw of \( (\hat{r}_j, \hat{b}_j) \) via \( \left( \hat{r}_{b,j}, \hat{b}_{b,j} \right) = \arg\min_{\mu, b} \sum_{i=1}^n w_{b,ij} \ell_\eta (X_{ij} - \mu - f_j^T b) \), where \( \{w_{b,ij}\}_{i=1}^n, j = 1, \ldots, p \) are independent and identically distributed (iid) random variables that are independent from the data and satisfy \( \mathbb{E}(w_{b,ij}) = 1 \) and \( \text{var}(w_{b,ij}) = 1 \). To retain convexity
Algorithm 1 FarmTest with known factors (Zhou et al., 2018)

**Input:** Data \( \{ (X_i, f_i) \}_{i=1}^n \), null hypotheses \( \{ h_0 \} \), and \( \alpha, \lambda \in (0, 1) \)

1. For \( j = 1, \ldots, p \), obtain the Huber estimators
   \( (\hat{\mu}_j, \hat{b}_j) \in \text{argmin}_{\mu,b} \sum_{i=1}^n \ell_{\tau_j} (X_{ij} - \mu - f_i b) \).
   
2. Estimation of residual variances \( \sigma_{u,j}^2 \): compute
   
   (i) \( \hat{\Sigma}_j = (1/n) \sum_{i=1}^n f_i f_i^\top, \hat{\theta}_j \in \text{argmin}_{\theta} \sum_{i=1}^n \ell_{\tau_j} (X_{ij}^2 - \theta) \) for \( j = 1, \ldots, p \);
   
   (ii) \( \check{\sigma}_{u,j} = \hat{\theta}_j - \hat{\theta}_j^2 - \hat{\theta}_j^2 \hat{\Sigma}_j \hat{b}_j \) if \( \hat{\theta}_j > \hat{\theta}_j^2 + \hat{\theta}_j^2 \hat{b}_j^\top \hat{b}_j^j \); otherwise \( \check{\sigma}_{u,j} = \hat{\theta}_j \).
   
3. Construct test statistics \( T_j = \sqrt{n/\check{\sigma}_{u,j}} \left( \hat{\mu}_j - h_0 \right) \) for \( j = 1, \ldots, p \).
4. Compute p-values \( \{ P_j \}_{j=1}^p = \left\{ \begin{array}{ll} 2\Phi(-|T_j|) & \text{for (1)}, \\ \Phi(-T_j) & \text{for (2)}, \\ \Phi(T_j) & \text{for (3)}. \end{array} \right. \)
5. Estimate the proportion of true alternatives: \( \hat{\pi}_0(\lambda) = \frac{\text{Card} \{ P_j > \lambda \}}{(1-\alpha)p} \).
6. Order the p-values as \( P(1) \leq \cdots \leq P(p) \).
   
   Compute the rejection threshold \( t := \max \{ 1 \leq j \leq p : P(j) \leq \frac{\alpha}{\hat{\pi}_0(\lambda)p} \} \).
7. Reject each hypothesis in the set \( \{ 1 \leq j \leq p : P_j \leq P(t) \} \).

**Output:** Rejected hypotheses, p-values, other estimated parameters

of the loss function, nonnegative random weights are preferred, such as \( w_{u,ij} \sim \text{Exp} (1) \)—exponential distribution with rate 1, or \( w_{u,ij} \sim 2 \text{Ber}(1/2) - \text{P} (w_{u,ij} = 0) = \text{P} (w_{u,ij} = 2) = 1/2 \). For two-sided alternatives, the bootstrap p-values are then defined as \( P_j^B = (1/B) \sum_{b=1}^B I \left( |\hat{\mu}_{u,j} - \hat{\mu}_j| \geq |\hat{\mu}_j| \right) \) followed by Steps 5–7 in Algorithm 1.

An extension of Algorithm 1 to the two-sample problem is also included in the library. Suppose we observe two independent samples \( \{ X_i - \mu_i \} \) and \( \{ Y_i - \mu_i \} \) from the models

\[
X = \mu + B f + u \quad \text{and} \quad Y = \mu + B' f' + u'.
\]

We are interested in the \( p \) hypotheses \( H_{0j} : \mu_X - \mu_Y = h_0 \) versus \( H_{1j} : \mu_X - \mu_Y \neq h_0 \) or versus some one-sided alternatives. To begin with, applying Step 1 in Algorithm 1 separately to each dataset to obtain the estimates \( \{ (\hat{\mu}_X, \hat{\mu}_Y) \}_{j=1}^p \) and \( \{ (\hat{\nu}_{u,j}, \hat{\nu}_{u,j}) \}_{j=1}^p \). Next, define the two-sample counterparts of the test statistics in Step 2 as \( T_j = \left( \hat{\mu}_X - \hat{\mu}_Y - h_0 \right)/\sqrt{\hat{\Sigma}_{u,j}^X/n_1 + \hat{\Sigma}_{u,j}^Y/n_2} \) for \( j = 1, \ldots, p \). After that, we follow Steps 3–7 as in Algorithm 1 to obtain the p-values and rejected hypotheses.

### Latent factors

In this section, suppose we are given independent observations \( \{ X_i \}_{i=1}^n \). The strong dependency among the coordinates of \( X_i \) is captured by a latent factor \( f_i \) (Leek and Storey, 2008). Due to the need of recovering latent factors from the data, the corresponding testing procedure is more involved. We summarize the major steps in Algorithm 2. All the tuning parameters required for Algorithm 2, \( \{ \tau_p, \nu_{lij} \}_{j=1}^p \) and \( \{ v_{jk} \} \), are automatically selected from the data; see Selection of tuning parameters.

An extension of Algorithm 2 to the two-sample problem is also included in the library. Suppose we observe two independent samples \( \{ X_i \}_{i=1}^n \) and \( \{ Y_i \}_{i=1}^m \), and wish to test the hypotheses \( H_{0j} : \mu_X - \mu_Y = h_0 \) versus \( H_{1j} : \mu_X - \mu_Y \neq h_0 \) or some one-sided alternatives. In this case, Steps 1–5 in Algorithm 2 are applied separately to each dataset to obtain the estimates \( \{ (\hat{\mu}_X, \hat{\mu}_Y) \}_{j=1}^p \).
Algorithm 2 FarmTest with latent factors (Fan et al., 2019)

**Input:** Data \( \{X_i\}_{i=1}^n \), null hypotheses \( \{h_0^j\}_{j=1}^p \), and \( \alpha, \lambda \in (0, 1) \)

1. For \( j = 1, \ldots, p \), compute
   - \( \hat{\mu}_j = \arg\min_\mu \sum_{i=1}^n \ell_{ij} (X_{ij} - \mu) \), \( \hat{\theta}_j = \arg\min_\theta \sum_{i=1}^n \ell_{ij} \left( \frac{X_{ij}^2 - \theta}{\theta} \right) \),
   - \( \hat{\sigma}_{jj} = \begin{cases} \hat{\theta}_j - \hat{\mu}_j^2 & \text{if } \hat{\theta}_j > \hat{\mu}_j^2 \\ \hat{\theta}_j & \text{otherwise.} \end{cases} \)

2. Define the paired data \( \{Y_1, Y_2, \ldots, Y_N\} = \{X_1 - X_2, X_1 - X_3, \ldots, X_{n-1} - X_n\} \), where \( N = n(n-1)/2 \). For \( 1 \leq j < k \leq p \), compute
   - \( \hat{\sigma}_{jk} = \arg\min_\theta \sum_{i=1}^n \ell_{ij} \left( \frac{Y_{ij} Y_{jk} / 2 - \theta}{\theta} \right) \), and \( \hat{\sigma}_{kj} = \hat{\sigma}_{jk} \).

3. Define the covariance matrix estimator \( \hat{\Sigma} = \left( \hat{\sigma}_{jk} \right)_{1 \leq j, k \leq p} \).
   - Let \( \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_p \) be the ordered eigenvalues of \( \hat{\Sigma} \) and denote by \( \nu_1, \nu_2, \ldots, \nu_p \) the corresponding eigenvectors.
   - Calculate \( K = \arg\max_{1 \leq k \leq \min(n, p)/2} \frac{\lambda_k}{\lambda_{k+1}} \). This step is omitted if \( K \) is user-specified.
   - Calculate \( \hat{B} = \left( \hat{b}_1, \ldots, \hat{b}_p \right)^T = \left( \lambda_1^{1/2} \nu_1, \ldots, \lambda_K^{1/2} \nu_K \right) \in \mathbb{R}^{p \times K} \).

4. \( \hat{f} = \arg\min_{f \in \mathbb{R}^K} \sum_{j=1}^p \ell_f \left( \frac{X_{ij} - \hat{b}_j^T f}{\hat{h}_0^j} \right) \), where \( X_{ij} = (1/n) \sum_{i=1}^n X_{ij} \).

5. For \( j = 1, \ldots, p \), compute \( \hat{\sigma}_{u, jj} = \begin{cases} \hat{\sigma}_{jj} - \|\hat{b}_j\|_2^2 & \text{if } \hat{\sigma}_{jj} > \|\hat{b}_j\|_2^2 \\ \hat{\sigma}_{jj} & \text{otherwise.} \end{cases} \)

6. Construct test statistics \( T_j = \sqrt{n/\hat{\sigma}_{u, jj}} \left( \hat{\mu}_j - \hat{b}_j^T f - h_0^j \right) \), \( j = 1, \ldots, p \).

7. Compute p-values \( P_j = \begin{cases} \Phi \left( -|T_j| \right) & \text{for (1)}, \\ \Phi \left( T_j \right) & \text{for (2)}, \\ \Phi \left( -T_j \right) & \text{for (3)}. \end{cases} \)

8. Estimate the proportion of true alternatives: \( \hat{\pi}_0 (\lambda) = \frac{\text{Card}\{P_j > \lambda\}}{(1-\lambda) p} \).

9. Compute the rejection threshold \( t = \max \left\{ 1 \leq j \leq p : P_j \leq \frac{\alpha_j}{\hat{\pi}_0 (\lambda) p} \right\} \).

10. Reject each hypothesis in the set \( \left\{ 1 \leq j \leq p : P_j \leq P(t) \right\} \).

**Output:** Rejected hypotheses, p-values, other estimated parameters

\[ \left\{ \left( \hat{\sigma}_{u, jj}, \hat{\sigma}_{i, jj}^X, \hat{\sigma}_{i, jj}^Y \right) \right\}_{j=1}^p, \hat{B}^X, \hat{B}^Y, \hat{f}^X \text{ and } \hat{f}^Y. \]

After replacing the test statistics in Step 6 with

\[ T_j = \frac{\left( \hat{\mu}_j - \hat{b}_j^T f - h_0^j \right)}{\sqrt{\hat{\sigma}_{u, jj}/\pi_1 + \hat{\sigma}_{i, jj}^X/\pi_2}}, \quad j = 1, \ldots, p, \]

one can follow Steps 7–10 to obtain the p-values and rejected hypotheses.

**Partially observable factors**

Motivated by applications to comparative microarray experiments (Leek and Storey, 2008; Friguet et al., 2009) and mutual fund selection (Lan and Du, 2019), we further discuss the case where both explanatory variables and latent factors are present. The statistical model is of the form

\[ X_i = \mu + B f_i + C g_i + u_i, \quad i = 1, \ldots, n, \]
where $f_i \in \mathbb{R}^K$ is a vector of explanatory variables and $g_i \in \mathbb{R}^L$ represents the latent factor. Here $L \geq 1$ may be user-specified or unknown. For multiple comparison of the mean effects under this model, the FarmTest package can be used in a two-stage fashion. In the first stage, apply Algorithm 1 to fit model (4) with observed data $\{ (X_i, f_i) \}_{i=1}^{n}$ and compute fitted residuals $X_i^{\text{res}} = X_i - B f_i$; in the second stage, run Algorithm 2 on $\{ X_i^{\text{res}} \}_{i=1}^{n}$ to conduct factor-adjusted multiple testing.

Selection of tuning parameters

The FarmTest procedure involves multiple tuning parameters, including the number of factors $K$ (if not specified by the user) and robustification parameters for fitting factor models. For the former, we apply the eigenvalue ratio method (Lam and Yao, 2012; Ahn and Horenstein, 2013) to estimate $K$, that is, $\hat{K} = \text{argmax}_{1 \leq k \leq K_{\text{max}}} \lambda_k (\hat{\Sigma}) / \lambda_{k+1} (\hat{\Sigma})$, where $\hat{\Sigma}$ is a generic covariance matrix estimator with eigenvalues $\lambda_1 (\hat{\Sigma}) \geq \cdots \geq \lambda_p (\hat{\Sigma})$, and $K_{\text{max}}$ be a prescribed upper bound. In the library, we take $K_{\text{max}} = \min (n, p) / 2$. This method is chosen as it does not involve other hyperparameters (except $K_{\text{max}}$). When the factors are unobservable, the estimation of $K$ is essentially an un-supervised learning problem. We choose $K$ to be the smallest nonnegative integer such that the residuals $X_i - B f_i$ are weakly correlated. Therefore, slight overestimation of $K$ does not affect much of the testing results. If $K$ is set to be zero, the FarmTest library directly applies a robust multiple testing procedure based on Huber’s $M$-estimation partnered with multiplier bootstrap. See Zhou et al. (2018) for more details.

The robustification parameter in the Huber loss plays an important role in controlling the bias-robustness tradeoff. According to the theoretical analysis in Zhou et al. (2018), the optimal choice of $\tau$ in Algorithm 1 depends on the variance of $X_i$. Due to heterogeneity, we have $p$ different $\tau$’s that need to be selected from the data. Furthermore, the covariance estimation step in Algorithm 2 entails as many as $p (p - 1) / 2$ parameters $\nu_{jk}$. Cross-validation is therefore computationally expensive when the dimension is large. Recently, Ke et al. (2019) and Wang et al. (2020) proposed fast data-driven methods, which estimate the regression coefficients/covariances and calibrate the tuning parameter simultaneously by solving a system of equations. Numerical studies therein suggest that this data-driven method is considerably faster than cross-validation while performs equally as well.

Package overview

The FarmTest package is publicly available from the Comprehensive R Archive Network (CRAN) and its GitHub page https://github.com/XiaoouPan/FarmTest. It contains four core functions. The main function farm.test carries out the entire FarmTest procedure, and outputs the testing results along with several useful estimated model parameters. User-friendly summary, print, and plot functions that summarize and visualize the test outcome are equipped with farm.test. The other three functions, huber.mean, huber.cov and huber.reg implement data-driven robust methods for estimating the mean vector and covariance matrix (Ke et al., 2019) as well as the regression coefficients (Wang et al., 2020). In particular, the huber.reg function uses the gradient descent algorithm with Barzilai and Borwein step size (Barzilai and Borwein, 1988). In this section, we focus primarily on introducing the farm.test function, and demonstrate its usage with numerical experiments.

A showcase example

We first present an example by applying the package to a synthetic dataset. To begin with, we use the rstiefel package (Hoff, 2012) to simulate a uniformly distributed random orthonormal matrix as the loading matrix $B$ after rescaling. With sample size $n = 120$, dimension $p = 400$ and number of factors $K = 5$, we generate data vectors $\{ X_i \}_{i=1}^{n}$ from model (4), where the factors $f_i \in \mathbb{R}^K$ follow a standard multivariate normal distribution and the noise vectors $u_i \in \mathbb{R}^p$ are drawn from a multivariate $t$ distribution with zero mean and identity covariance matrix. For the mean vector $\mu = (\mu_1, \ldots, \mu_p)^\top$, we set the first $p_1 = 100$ coordinates to be 1 and the rest to be 0.

```r
library(FarmTest)
library(rstiefel)
library(mvtnorm)

n <- 120
p <- 400
K <- 5
set.seed(100)
B <- rstiefel(p, K) %>% diag(rep(sqrt(p), K))
FX <- rmvnorm(n, rep(0, K), diag(K))
```
p1 <- 100
strength <- 1
mu <- c(rep(strength, p1), rep(0, p - p1))
U <- rmvt(n, diag(p), 3)
X <- rep(1, n) %*% t(mu) + FX %*% t(B) + U

Function call with default parameters

Using the data generated above, let us call the main function farm.test with all default optional parameters, and then print the outputs.

output <- farm.test(X)
output

One-sample FarmTest with unknown factors
n = 120, p = 400, nFactors = 5
FDR to be controlled at: 0.05
Alternative hypothesis: two.sided
Number of hypotheses rejected: 104

As shown in the snapshot above, the function farm.test correctly estimates the number of factors, and rejects 104 hypotheses with 4 false discoveries. For this individual experiment, the FDP and power are 0.038 and 1, respectively as calculated below. Here the power is referred to as the ratio between the number of correct rejections and the number of nonnulls p1.

FDP <- sum(output$reject > p1) / length(output$reject)
FDP

[1] 0.03846154

globalPower <- sum(output$reject <= p1) / p1
globalPower

[1] 1

All the outputs are incorporated into a list, which can be quickly examined by names() function. See Table 1 for detailed descriptions of the outputs.

names(output)

[1] "means" "stdDev" "loadings" "eigenVal" "eigenRatio" "nFactors"
[7] "tStat" "pValues" "pAdjust" "significant" "reject" "type"
[13] "n" "p" "h0" "alpha" "alternative"

We can present the testing results using the affiliated summary function.

head(summary(output))

means p-values p-adjusted significance
1 1.0947056 1.768781e-18 8.936997e-17 1
2 0.8403608 3.131733e-09 1.157817e-08 1
3 0.8668348 1.292850e-11 6.532295e-11 1
4 0.9273998 2.182485e-12 1.350281e-11 1
5 0.7257105 7.699350e-08 2.593465e-07 1
6 0.9473088 1.180288e-13 1.192712e-12 1

To visualize the testing results, in Figure 1 we present several plots based on the outputs. From the histograms of estimated means and test statistics, we see that data are generally categorized into two groups, one of which has \( \hat{\mu}_j \) concentrated around 1 and test statistics bounded away from 0. It is therefore relatively easy to identify alternatives/signals from the nulls. From the eigenvalue ratio plot, we see that the fifth ratio (highlighted as a red dot) is evidently above the others, thus determining the number of factors. The scree plot, on the other hand, reveals that the top 5 eigenvalues (above the red dashed line) together explain the vast majority of the variance, indicating that the proportion of common variance (due to common factors) is high.
Table 1: Objects in the output list of \texttt{farm.test} function with their implications, and description of data type and class in R language.

<table>
<thead>
<tr>
<th>Output</th>
<th>Implication</th>
<th>Data type</th>
<th>R class</th>
</tr>
</thead>
<tbody>
<tr>
<td>means</td>
<td>estimated means</td>
<td>(p)-vector</td>
<td>matrix</td>
</tr>
<tr>
<td>stdDev</td>
<td>estimated standard deviations</td>
<td>(p)-vector</td>
<td>matrix</td>
</tr>
<tr>
<td>loadings</td>
<td>estimated loading matrix</td>
<td>((p \times K))-matrix</td>
<td>matrix</td>
</tr>
<tr>
<td>eigenVal</td>
<td>eigenvalues of estimated covariance</td>
<td>(p)-vector</td>
<td>matrix</td>
</tr>
<tr>
<td>eigenRatio</td>
<td>eigenvalue ratios of estimated covariance</td>
<td>((\min {n, p}/2))-vector</td>
<td>matrix</td>
</tr>
<tr>
<td>nFactors</td>
<td>(estimated) number of factors</td>
<td>positive integer</td>
<td>integer</td>
</tr>
<tr>
<td>tStat</td>
<td>test statistics</td>
<td>(p)-vector</td>
<td>matrix</td>
</tr>
<tr>
<td>pValues</td>
<td>adjusted p-values</td>
<td>(p)-vector</td>
<td>matrix</td>
</tr>
<tr>
<td>pAdjust</td>
<td>indicators of significance</td>
<td>boolean (p)-vector</td>
<td>matrix</td>
</tr>
<tr>
<td>significant</td>
<td>indices of rejected hypotheses</td>
<td>vector</td>
<td>integer</td>
</tr>
<tr>
<td>type</td>
<td>whether factor is known</td>
<td>string</td>
<td>character</td>
</tr>
<tr>
<td>n</td>
<td>sample size</td>
<td>positive integer</td>
<td>integer</td>
</tr>
<tr>
<td>p</td>
<td>data dimension</td>
<td>positive integer</td>
<td>integer</td>
</tr>
<tr>
<td>h(0)</td>
<td>null hypothesis</td>
<td>(p)-vector</td>
<td>numeric</td>
</tr>
<tr>
<td>alpha</td>
<td>nominal FDR level</td>
<td>numerical number</td>
<td>numeric</td>
</tr>
<tr>
<td>alternative</td>
<td>alternative hypothesis</td>
<td>string</td>
<td>character</td>
</tr>
</tbody>
</table>

Figure 1: Upper panel: histograms of estimated means and test statistics. Lower panel: eigenvalue ratio plot with the largest ratio highlighted and scree plot of the eigenvalues of the estimated covariance matrix.

Function call with options

In this section, we illustrate \texttt{farm.test} function with other options that allow us to call it more flexibly. When the factors are observable, we can simply put the \(n \times K\) factor matrix into argument \(fX\), and the output is formatted the same as before. As a remark, among all the items listed in Table 1, \texttt{eigenVal} and \texttt{eigenRatio}, which are eigenvalues and eigenvalue ratios of estimated covariance matrix, are not available in this case; see Algorithm 1.

```r
output <- farm.test(X, fX = FX)
output
```

One-sample FarmTest with known factors
\(n = 120\), \(p = 400\), \(nFactors = 5\)
Consider one-sided alternatives $H_{1j}: \mu_j \geq 0, j = 1, \ldots, p$ with a nominal FDR level 1%. We modify the arguments alternative and alpha as follows:

```r
output <- farm.test(X, alternative = "greater", alpha = 0.01)
```

Users can specify null hypotheses by passing any vector with length $p$ into argument $h0$. In the next example, we consider the $p$ null hypotheses as all the means are equal to 1, so that the number of true nonnulls becomes 300.

```r
output <- farm.test(X, h0 = rep(1, p), alpha = 0.01)
```

When the factors are unknown, users can also specify the number of factors based on some subjective grounds. In this case, Step 3 in Algorithm 2 is avoided. For example, we run the function with the number of factors chosen to be $K = 2$, which is less than the true parameter 5. This misspecification results in a loss of power with two true alternatives unidentified.

```r
output <- farm.test(X, K = 2)
power <- sum(output$reject <= p1) / p1
power
[1] 0.98
```

As a special case, if we declare $K = 0$ in the function, a robust multiple test without factor-adjustment is conducted.

```r
output <- farm.test(X, K = 0)
```

Finally, we present an example of two-sample FarmTest. Using the same sampling distributions for the factor loading matrix, factors and noise vectors, we generate another sample $\{Y_i\}_{i=1}^m$ from model (5) with $m = 150$.

```r
m <- 150
set.seed(200)
BY <- rustiefel(p, K) %*% diag(rep(sqrt(p), K))
FY <- rmvnorm(m, rep(0, K), diag(K))
uY <- rmvt(m, diag(p), 3)
Y <- FY %*% t(BY) + uY
```
Then the `farm.test` function can be called with an additional argument Y.

```r
output <- farm.test(X, Y = Y)
output
```

Two-sample FarmTest with unknown factors  
X.n = 120, Y.n = 150, p = 400, X.nFactors = 5, Y.nFactors = 5  
FDR to be controlled at: 0.05  
Alternative hypothesis: two.sided  
Number of hypotheses rejected: 105

The output is formatted similarly as in Table 1, except that `means`, `stdDev`, `loadings`, `eigenVal`, `eigenRatio`, `nFactors` and `n` now consist of two items for samples X and Y.

```r
names(output$means)
```

[1] "X.mean" "Y.mean"

Simulations

In this section, we assess and compare the performance of the `farm.test` function in the `FarmTest` package with the following methods:

- t-test using the R built-in function `t.test`;
- WMW-test (Wilcoxon-Mann-Whitney) using the `onesamp.marginal` function in the `mutoss` package;
- RmTest (Robust Multiple test) without factor-adjustment by claiming `KX = 0` in the `farm.test` function.

For t-test and WMW-test, the functions we call produce vectors of p-values, to which the method proposed in Storey (2002) is applied, see Steps 5–7 in Algorithm 1 or Steps 8–10 in Algorithm 2.

In all the numerical experiments, we consider two-sided alternatives with a nominal FDR level $\alpha = 5\%$. The true number of factors is 5. Factors and loadings are generated the same way as in a showcase example Section. To add dependency among idiosyncratic errors, the covariance matrix of $u$, denoted by $\Sigma_u$, is taken to be a block-diagonal symmetric matrix with block size $5 \times 5$. Within each block, the diagonal entries are all equal to 3 and the off-diagonal entries are generated from $U[0, 1]$. In the simulations, we drop the case where the generated $\Sigma_u$ is not positive-definite. The distribution of $u$ is specified in two models as follows.

- **Model 1.** $u \sim N(0, \Sigma_u)$: centered multinormal distribution with covariance matrix $\Sigma_u$;  
- **Model 2.** $u \sim t_3(0, \Sigma_u)$: multivariate t-distribution with degrees of freedom 3 and covariance matrix $\Sigma_u$.

For each model, we consider various combinations of sample size $n$ and dimensionality $p$, specifically, $n \in \{60, 80, 100, 120, 140\}$ and $p \in \{200, 400, 600, 800, 1000\}$. The number of true alternatives $p_1$ is taken to be 0.2 $p$, and the signal strength is set as $4 \sqrt{\log(p)/n}$.

Figures 2 and 3 depict the FDR and power curves for either "fixed $n$ growing $p$" or "fixed $p$ growing $n$" based on 200 simulations. Across various settings, FarmTest consistently maintains high empirical power with FDR well controlled around the nominal level. In contrast, the competing methods may lose as many as 10% to 30% powers, which can be ascribed to not accounting for the common factors. In summary, we conclude that the `FarmTest` package provides an efficient implementation of the FarmTest method, which carries out multiple testing for multivariate data with heavy-tailed distribution and a strong dependency structure.

Real data example

In this section, we apply the `FarmTest` package to test the mean effects of stock returns. In capital asset pricing theory, the stock's risk-adjusted mean return or "alpha" is a quantity of interest since it indicates the excessive return incurred from investing in a particular stock. If the efficient equity market hypothesis holds, we expect "alpha" to be zero. Hence, detecting non-zero alphas can help investors to identify market inefficiencies, that is, whether certain stocks exhibit an abnormal rate...
of return or are mispriced. As discussed in Cont (2001), both cross-sectional dependency and heavy tailedness are silent features of stock returns.

In this study, we test the annual mean effects of stocks in the S&P500 index. The data is available on COMPUSTAT and CRSP databases. We find that most of the stocks with continuous membership in the S&P500 index from 2008 to 2016 have excess kurtosises greater than zero, indicating tails heavier than that of a normal distribution. Also, more than 33% of the stocks are severely heavy-tailed as their

---

**Figure 2:** Comparison of FarmTest with three other methods in terms of FDR and power under Model 1 (multivariate normal distribution). In the upper panel, $p$ is fixed at 600 and $n$ grows from 60 to 140; in the lower panel, $n$ is fixed at 100 and $p$ ranges from 200 to 1000.

**Figure 3:** Comparison of FarmTest with three other methods in terms of FDR and power under Model 2 (multivariate $t$-distribution). In the upper panel, $p$ is fixed at 600 and $n$ grows from 60 to 120; in the lower panel, $n$ is fixed at 100 and $p$ ranges from 200 to 1000.
excess kurtosises exceed 6, which is the excess kurtosis of $t_5$-distribution. We collect monthly returns of stocks from the S&P500 index over rolling windows: for each month between 2008 and 2016, we collect monthly returns of stocks who have continuous records over the past year. The average number of stocks collected each year is 598. For each rolling window, we conduct multiple testing using the four methods considered in the previous section, that is, FarmTest, $t$-test, WMW-test, and RmTest.

The nominal FDR level is set as $\alpha = 1\%$. Within each rolling time window, we have $p \approx 600$ and $n = 12$. The numbers of discoveries of each method are depicted chronologically in Figure 4, and Table 2 displays several key summary statistics. Since the $t$-test barely discovers any stock throughout the whole procedure, we only present the results for the other three methods in Figure 4. It is interesting to observe that across different time rolling windows, the testing outcomes of the WMW-test are relatively stable and time-insensitive. FarmTest, on the other hand, selects much fewer stocks in the year of 2009, coinciding to some extent with the financial crisis during which the market volatility is much higher. RmTest typically selects the most stocks, which is partly due to the lack of FDR control under strong dependency. A major, noticeable impact of dependence is that it results in clusters of rejections: if a test is rejected, then there are likely to be further rejections for tests that are highly correlated with this one. This phenomenon is in accord with our simulation results, showing that FarmTest simultaneously controls the FDR and maintains high power while the other methods either make too many false discoveries or fail to detect true signals.

<table>
<thead>
<tr>
<th>Method</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>Median</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>FarmTest</td>
<td>14.477</td>
<td>11.070</td>
<td>12</td>
<td>0</td>
<td>52</td>
</tr>
<tr>
<td>WMW-test</td>
<td>10.991</td>
<td>1.005</td>
<td>11</td>
<td>8</td>
<td>12</td>
</tr>
<tr>
<td>RmTest</td>
<td>8.147</td>
<td>14.414</td>
<td>3</td>
<td>0</td>
<td>68</td>
</tr>
</tbody>
</table>

Table 2: Summary statistics of the number of discoveries via FarmTest, WMW-test and RmTest between 2008 and 2016 using rolling windows of size 12 (months).

**Summary**

We provide an R package to implement FarmTest, a flexible large-scale multiple testing method that is robust against strongly dependent and heavy-tailed data. The factor-adjustment procedure helps to construct weakly dependent test statistics, and also enhances statistical power by reducing the signal-to-noise ratio. Moreover, by exploiting the idea of adaptive Huber regression, the testing procedure is robust against heavy-tailed noise. The efficacy of our package is demonstrated on both real and simulated datasets.
Bibliography


