Using Probabilistic Knockoffs of Binary Variables to Control the False Discovery Rate

Aaron Maurer

July 27, 2015

1 Introduction

Variable selection is an essential problem to fitting a regression model. For linear regression, the knockoff filter (Barber and Candès, 2014) offers a method of exact FDR control. However, the method as originally formulated does not extend to other generalized linear models such as logistic regression. This paper offers an extension of this method which will work with binary features in the context of linear regression, but also will extend to other GLMs.

1.1 Knockoff Filter

Let us consider the consider the usual setting for linear regression, where n observations of a variable of interest y arise from the model

$$\mathbf{y} = X\beta + \mathbf{z}$$

where $\mathbf{y} \in \mathbb{R}^n$ are the observed values of y, $X \in \mathbb{R}^{n\times p}$ is the matrix of predictor variables, $\beta \in \mathbb{R}^p$ are the unknown coefficients, and $\mathbf{z}$ is Gaussian noise. It is important to note that a intercept vector $\mathbf{1}$ is not included in X. For the purpose of this paper, only the case where $2p \leq n$ will be considered, though the methods can be extended to $p \leq n$. I will to refer the jth column of X as X_j, which is n observations of the random variable x_j. The random variable x_j is in turn jth entry of the random vector valued variable $\mathbf{x}$. In the situation where β is sparse and there is reason to expect that x_j and y are uncorrelated for many j, the knockoff filter is a method to select a subset of the x_j which are likely correlated with y.

The method specifically seeks to control the false discovery rate (FDR), which is the expected portion of some selection of x_j which are, in fact, uncorrelated with y. For a procedure which chooses a subset of variables $\hat{S} \subseteq \{1, \dots, p\}$, this is defined as

$$\text{FDR} = \text{E}\left[\frac{|\{j : \beta_j = 0 \;\&\; j \in \hat{S}\}|}{\max\{|\hat{S}|, 1\}}\right]$$

FDR is controlled at level q if FDR is less than q irrespective of the coefficients β.

The knockoff filter achieves FDR control in several steps, the first of which is creating a set of 'knockoff' features $\tilde{X}_j$ which imitate the original X_j while being less correlated with y. In particular, the matrix of knockoffs $\tilde{X}$ has the same internal correlation structure as X, the knockoff feature $\tilde{X}_j$ has the same correlation with other X_i as X_j does, but the correlation between $\tilde{X}_j$ and X_j is minimized. In other words, where $G := X^T X$,

$$\tilde{X}^T \tilde{X} = G \text{ and } \tilde{X}^T X = G - \text{diag}\{\mathbf{s}\}$$

for some vector $s \in \mathbb{R}^p$ such that, where $\text{diag}\{G\}$ is the vector made from the main diagonal of G, $\text{diag}\{G\} - \mathbf{s}$ is small. Since $\tilde{X}_j$ and X_j have relatively low correlation, as long as X_j is created independently of $\mathbf{y}$, $\tilde{X}_j$ will have lower correlation with $\mathbf{y}$ than X_j. In the case where X_j is uncorrelated with $\mathbf{y}$, $\tilde{X}_j$ will also be uncorrelated with $\mathbf{y}$. How these knockoffs are actually generated will be described shortly.

The second step is to fit a series of Lasso model of $\mathbf{y}$ on the combined design matrix $X_L = [X \ \tilde{X}]$ so as to determine the largest value λ at which the coefficient for each X_j and $\tilde{X}_j$ is nonzero. The subscript L refers to the large design matrix with both features and knockoffs. [1] Recall, for a given λ, the estimated coefficient from the Lasso regression will be

$$\beta(\lambda) = \arg\min_{\mathbf{b}} \left\{ \frac{1}{2} \|\mathbf{y} - X_L \mathbf{b}\|_2^2 + \lambda \|b\|_1 \right\}$$

In the general case, the coefficient for one feature in a Lasso regression being first nonzero for a larger value of λ than another feature is an indication that the first feature is a stronger predictor of the outcome than the second. Thus, since, by construction, the knockoff features are weaker predictors than the originals, we would expect the original features to have nonzero coefficients sooner than the knockoff features when the original is a valid predictor. On the other hand, since $\beta(\lambda)$ only depends on X_L through the sufficient statistics $G_L = X_L^T X_L$ and $X_L^T \mathbf{y}$, for null predictors, the coefficients of the original features won't enter any sooner on average than knockoff feature. This can be seen by switching a null X_j with $\tilde{X}_j$; by construction, G_L will be unaltered, while $\text{E}\left[X_L^T \mathbf{y}\right]$ will also be unchanged, since $\text{E}[X_j^T \mathbf{y}] = \text{E}[\tilde{X}_j^T \mathbf{y}] = 0$.

This observation leads to the final step. Let Z_j be the largest λ such that feature j has a nonzero coefficient and $\tilde{Z}_j$ the largest λ such that knockoff j has a nonzero coefficient. Then, define W_j as

$$W_j = \begin{cases} Z_j & \text{if } Z_j > \tilde{Z}_j \\ -\tilde{Z}_j & \text{if } Z_j < \tilde{Z}_j \\ 0 & \text{if } Z_j = \tilde{Z}_j \end{cases}$$

When X_j is a null predictor, W_j will be symmetrically distributed around zero, and when X_j is true predictor, it will have a distribution skewed positive. Since W_j with large positive values are more likely to be true predictors, the knockoff filter selects the variables $\{j : W_j \geq T\}$, where T is a threshold defined as

$$T = \min \left\{ t > 0 \ : \ \frac{|\{j : W_j \leq -t\}|}{\max\{|\{j : W_j \geq t\}|, 1\}} \leq q \right\}$$

[1] Other statistics besides the λs can also be used in the knockoff method, but are not considered in this paper

Basic logic of this method is fairly simple. Let $N_h(t)$ be the random variable for the number of null predictors with $W_j \leq t$, $N_l(t)$ be the number of null predictors with $W_j \geq t$, and $V_l(t)$ & $V_h(t)$ be the similar number of true predictors. Since the null W_j are symmetrically distributed around 0, $\mathrm{E}[N_l(t)] = \mathrm{E}[N_h(t)]$. Thus,

$$\begin{aligned}
q &\geq \mathrm{E}\left[\frac{|\{j : W_j \leq -t\}|}{\max\{|\{j : W_j \geq t\}|, 1\}}\right] \\
q &\geq \mathrm{E}\left[\frac{N_l(t) + V_l(t)}{\max\{N_h(t) + V_h(t), 1\}}\right] \\
q &\geq \mathrm{E}\left[\frac{N_l(t)}{\max\{N_h(t) + V_h(t), 1\}}\right] \\
q &\geq \mathrm{E}\left[\frac{N_h(t)}{\max\{N_h(t) + V_h(t), 1\}}\right] \\
q &\geq \mathrm{FDR}
\end{aligned}$$

So FDR is controlled, and since $V_l(t)$ will tend to be small, it should be controlled fairly tightly.

1.2 Original Knockoff Features

The original formulation of the method offers two similar methods of constructing knockoffs. Both of these will, by construction, have exactly the property that

$$\tilde{X}^T\tilde{X} = G \text{ and } \tilde{X}^T X = G - \mathrm{diag}\{\mathbf{s}\}.$$

For both methods, the first step is to normalize the matrix X such that $X_j^T X_j = 1$ for all j. Then, let the Gram matrix of X_L is

$$G_L = [X\ \tilde{X}]^T[X\ \tilde{X}] = \begin{bmatrix} G & G - \mathrm{diag}\{\mathbf{s}\} \\ G - \mathrm{diag}\{\mathbf{s}\} & G \end{bmatrix}$$

A, the Schur complement of G in G_L can be calculated as

$$A = 2\,\mathrm{diag}\{\mathbf{s}\} - \mathrm{diag}\{\mathbf{s}\}G^{-1}\mathrm{diag}\{\mathbf{s}\}$$

For G_L to exist, G_L must be positive semi-definite, which happens if and only if A is positive semi-definite, which happens in turn if and only if[2]

$$\mathrm{diag}\{\mathbf{s}\} \succeq 0 \quad\text{and}\quad 2G - \mathrm{diag}\{\mathbf{s}\} \succeq 0$$

Given this is true, A can be factored as $A = C^T C$. Combining this with a satisfactory $\mathbf{s}$ and an orthonormal matrix $\tilde{U} \in \mathbb{R}^{n\times p}$ such that $\tilde{U}^T X = 0$, a $\tilde{X}$ fulfilling the desired properties can be calculated as

$$\tilde{X} = X(I - G^{-1}\,\mathrm{diag}\{\mathbf{s}\}) + \tilde{U}C$$

As mentioned above, $\mathbf{s}$ should be chosen so as to make $\mathrm{diag}\{G\} - \mathbf{s}$ small. Since each X_j has been normalized, this means that $\mathrm{diag}\{G\} = \mathbf{1}$, so $\mathbf{1} - \mathbf{s}$ should be minimized in accordance with the restrictions on $\mathbf{s}$. The two methods differ in how $\mathbf{s}$ is chosen:

[2] A fleshed out version of this argument can be found in the original knockoff paper

- *Equi-correlated knockoffs*: Each original features is set to have the same correlation with its knockoff by setting $\mathbf{s} = 2\min\{\lambda_{min}(G), 1\}\mathbf{1}$. For all simulations in this paper, this was the method used.

- *SDP knockoffs*: The $\mathbf{s}$ which minimizes the average correlation between knockoff and original features can be found via a semi-definite programing problem:

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{1} - \mathbf{s}\|_1 \\ \text{subject to} \quad & 0 \preceq \operatorname{diag}\{\mathbf{s}\} \preceq 2G \end{aligned}$$

  This method is significantly more computationally intensive.

1.3 Binary Knockoffs

Though the "original" knockoff method, described above, achieves the exact desired knockoff properties, the individual values in the vector X_j will have little relation to the individual values in X_j. In particular, these values will often have very different empirical distributions. This effect is particularly noticeable when the random variable x_j is discrete but $\tilde{X}_j$ does not even consist of the same set of values. The end result is that for variable selection for other generalized linear models, which do not have the same sufficient statistics as linear regression which knockoffs are designed to hit, the original knockoffs will fail to control FDR.

Thus, this paper offers a new method of generating knockoffs which should offer superior performance with other GLMs. In this new method, the matrix of knockoff features $\tilde{X}$ will be generated by row from a random vector valued variable $\tilde{\mathbf{x}}$, each entry $\tilde{x}_j$ of which will have the property $\tilde{x}_j \sim x_j$. This random variable will also abide by a relaxation of the original knockoff condition:

$$\mathrm{E}[\tilde{\mathbf{x}}^T\tilde{\mathbf{x}}] = G \;\text{ and }\; \mathrm{E}[\tilde{\mathbf{x}}^T\mathbf{x} \mid \mathbf{x}] = G - \operatorname{diag}\{\mathbf{s}\}$$

This will ensure that, at the very least, these knockoffs are a suitable replacement for the original knockoffs for linear regression.

Generating such knockoffs for a completely general case of any $\mathbf{x}$ is likely infeasible. However, in the specific case where $\mathbf{x}$ is a random binary vector variable, this paper will demonstrate a method to generate such knockoff variables. Since binary data is extremely common in many areas, this will hopefully serve as a useful tool.

1.4 Paper Organization

The rest of this paper will be organized in the following fashion:

- Section 2 will go into more detail about how the original knockoffs breakdown with generalized linear models. In particular, several simulations will demonstrate how they breakdown.

- Section 3 will develop the method and theory for generating binary knockoffs.

- Section 4 will discuss tests of Binary knockoffs in simulation, both in comparison to the original knockoffs with Lasso and for logistic regression by itself.
- Section 5 is the final section, and will contain discussion of the results as well as areas for further work.

2 Issues With Deterministic Knockoffs

The knockoff filter and W statistics have a very natural extension to generalized linear models. Here, where $l(\beta \mid X, \tilde{X})$ is the log likelihood of coefficients β given the model, the $L1$ regularized regression model will have estimated coefficient vector

$$\beta(\lambda) = \arg\min_{\mathbf{b}} \left\{ l(\beta \mid X, \tilde{X}) + \lambda \|b\|_1 \right\}$$

The Z_j and $\tilde{Z}_j$ are then, once again, the largest λ such that the given original or knockoff feature has a positive coefficient, and then W_j is constructed exactly as before. The problem with this is that if the given GLM doesn't have the same sufficient statistics as linear regression, there is no theoretical guarantee that the knockoff filter will control FDR. This is since, for null X_j, there is no theory to guarantee the Z_j and $\tilde{Z}_j$ are exchangeable, and thus W_j is not guaranteed to be symmetrically distributed around 0.

To test how bad this problem is, I tested the method using original style knockoffs in logistic regression. The results can be seen in figure 1, which shows cumulatively, starting with the largest absolute value, what portion of the W_j are positive. For the method to work, for the null predictors, this should be a consistent 50%, indicative of the desired symmetric distribution. This is particularly important for high values, which play the largest roll in selecting a threshold. We see that in the top left corner, when all features are null and normally distributed, that we almost achieve this, with only a small, but consistent, deviation from 50%. However, in the top right, when the predictors are binary, the null originals enter first far more often, which will result in the FDR being too high.

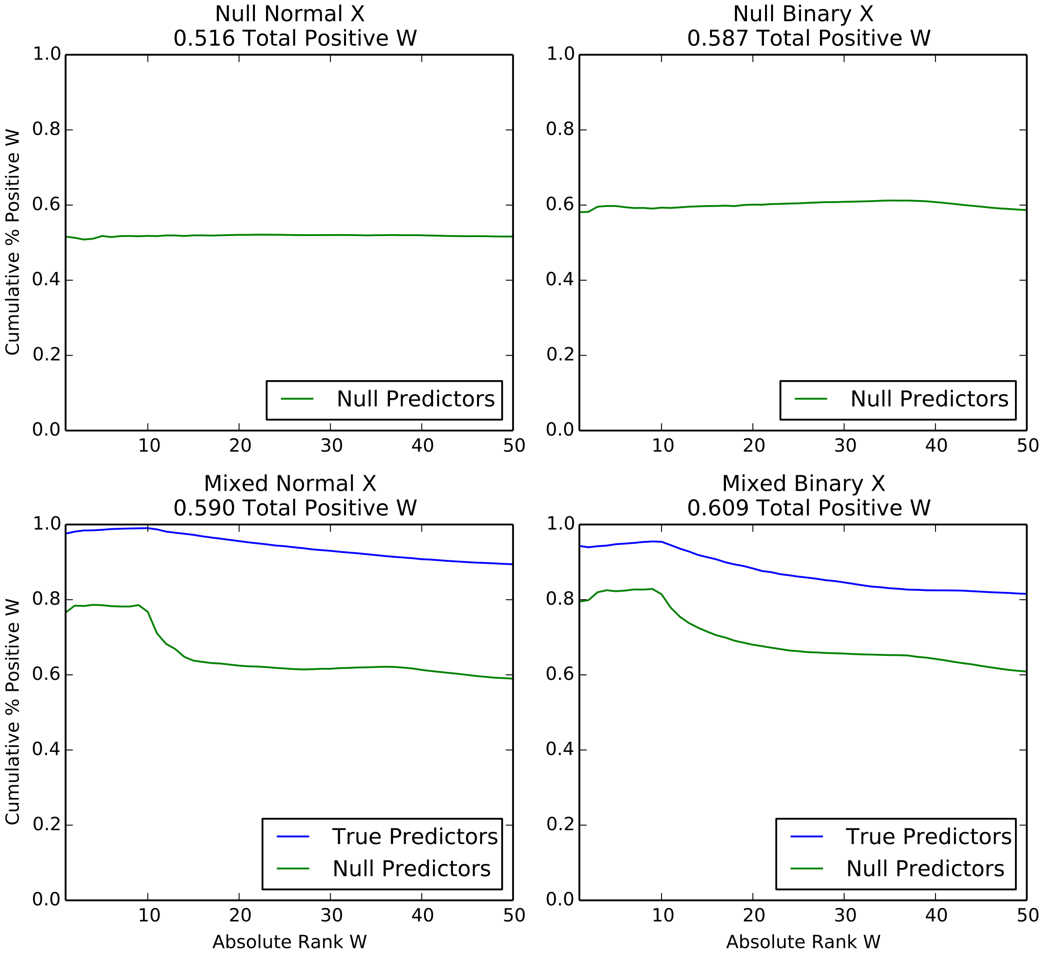

Figure 1: The mean cumulative portion of W statistic which are positive by absolute rank of the value. The mean is over 1000 simulations. On the left side, the features are drawn from a random standard normal distribution. On the right side, they are drawn as Bernoulli with probability .25 of being 1. The top has 50 features uncorrelated with the outcome, while the bottom has 40 features uncorrelated with the outcome and 10 correlated with it.

This gets much worse when some true predictors are included, as shown in the bottom row of the figure. For both normal and binary predictors, the originals come in first far more often in the beginning. This will completely ruin FDR control, since the portion of null predictors will be assumed far lower than it actually is. It is worth noting that the particular shape of the curve in the diagram, with the plateau for the null predictors through 10, is partially an artifact of how the totals were tabulated. The mean through the first ten is the mean for null predictors that came in first out of both null and true predictors; since the largest ten W_j will tend to be true predictors, the average over the null predictors represent the few extreme W_j which managed to beat out some number of true predictors.

Upon examination, the issue seems to be that the original style knockoff features have a very different empirical distribution than the features they are imitating. This is obviously true with the binary features, but even with the normal features, the distributions differ a lot. In figure 2, the distribution of four feature and the original style

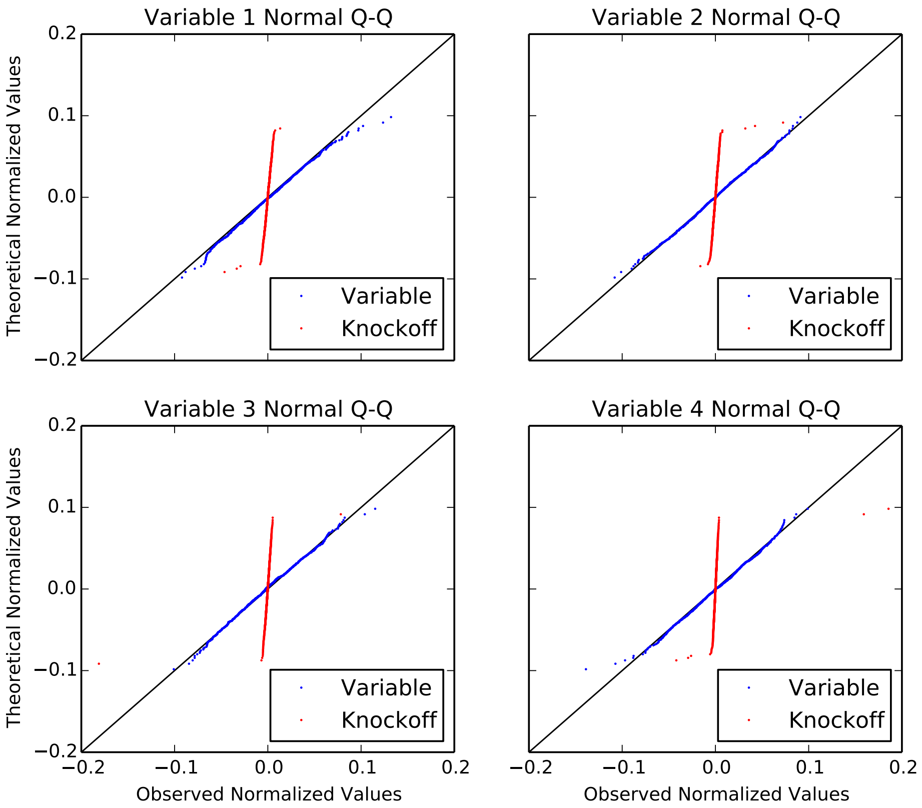

Figure 2: Normal Q-Q of original variables and knockoffs for simulation with 4 variables and 1,000 observations

knockoff are plotted against a normal distribution in Q-Q plots. As you can see, they don't match at all. This is suggestive of a solution; if the knockoff variables match the marginal distribution of the original features better, the result may be more suitable for FDR control in GLMs.

3 Random Binary Knockoffs

Random binary knockoffs $\tilde{\mathbf{x}}$, which maintain the correlation condition in expectation, fit the bill in this regard. The method to produce them is based heavily on more general methodology to random binary vector variables with internal correlation.

3.1 Random Binary Vector Generation

The full class of random binary vector variables on $\{0,1\}^p$ can be specified as multinomial on the 2^p elements of the set. However, in practice, it is generally impractical to specify or estimate all 2^p necessary parameters, so it is useful to, as is desirable for knockoffs, pick a method to match the first two moments. These are the mean vector $\mathrm{E}(\mathbf{x}) = \mathbf{m} \in \mathbf{(0,1)^p}$ and the cross-moment matrix $\mathrm{E}(\mathbf{x}\mathbf{x}^\mathbf{T}) = \mathbf{M} \in \mathbf{(0,1)^{p\times p}}$. Obviously, $m_i = \mathrm{P}(x_i = 1)$, $M_{ij} = \mathrm{P}(x_i = x_j = 1)$, and $\mathbf{m} = \mathrm{diag}\{M\}$. For an arbitrary symmetric M to be valid cross-moment matrix for some random binary vector variable, $M - mm^T \succeq 0$, and

$$\max\{0, m_i + m_j - 1\} \leq M_{ij} \leq \min\{m_i, m_j\}$$

for all $i \neq j$. Given a qualifying M, or observed X, there are then a few methods for generating additional random values.

3.1.1 Gaussian Copula Family

Since multivariate normal distributions are easy to randomly draw from and defined by their first two moments, an obvious choice is to use them to generate random binary vectors. This might be done by finding μ and Σ such that for $\mathbf{z} \sim N_p(\mu, \Sigma)$, x_i can be generated as $x_i = \text{sign}(z_i)$, and have the desired first two moments. However, this is only guaranteed to be feasible in the bivariate case. For higher dimensions, this method can generally only provide random binary vectors with approximately the right moments, and the quality of the approximation quickly degenerates as p increases.

3.1.2 μ-Conditionals family

There exists a more flexible family which will always work for arbitrary M called μ-conditionals. The basic idea is that the X is generate sequentially as

$$\mathrm{P}(x_i = 1 \mid x_1, ..., x_{i-1}) = \mu\left(a_{ii} + \sum_{k=1}^{i-1} a_{ik} x_i\right)$$

for some monotone function $\mu : \mathbb{R} \to (0, 1)$. This is essentially a binomial family GLM for a link function μ. If one takes all of the a_{kj}, they can form a lower triangular matrix A, and then the joint density can be expressed as

$$\mathrm{P}(\mathbf{x} = \gamma) \propto \mu(\gamma^T A \gamma)$$

If μ is chosen such that it is a bijection and differentiable, there is a unique A such that $\mathrm{E}(\mathbf{x}\mathbf{x}^T) = M$ when generated from this model. The natural choice for μ is the logistic link function $L(x) = \frac{1}{1-e^{-x}}$, which yields the Ising model, the "binary analogue of the multivariate normal distribution which is the maximum entropy distribution on $\mathbb{R}^p$ having a given covariance matrix." Additionally, it has the usual benefit that the coefficients can be viewed as a log odds ratio:

$$a_{ij} = \log\left(\frac{\mathrm{P}(x_j = x_k = 1)\mathrm{P}(x_j = x_k = 0)}{\mathrm{P}(x_j = 0, x_k = 1)\mathrm{P}(x_j = 1, x_k = 0)}\right)$$

when $i \neq j$. When $\mathbf{x}$ is generated from this model with $a_{ij} = 0$, then x_i and x_j are conditionally independent.

There is no closed form to calculate the entries in A if $p > 1$, but they can be derived numerically two ways. The first of these, when one is attempting to replicate the empirical cross-moments from a data matrix X, a_{1i} to a_{ii} is by fitting successive binomial family GLM regressions. The link function is the given function μ, and X_i is regressed on $X_1 \ldots X_{i-1}$ using maximum likelihood. a_{ji} for $i \neq j$ will then be the coefficient on X_j while a_{ii} is the intercept of the regression.

Otherwise, if one is just working with a desired cross-moment matrix M, the successive rows of A can be fit via Newton-Raphson. This is performed successively on each row of A. If the first $i-1$ rows have been fit, then the upper left $(i-1) \times (i-1)$ sub matrix A_{-i} of A has already been filled. Next, $\mathbf{a}_i$, the first i entries of the ith row of A must be fit, while the rest of the row will be 0. This corresponds to $\mathbf{m}_i$, the first i entries of the ith row of M. In other words, $\mathbf{m}_i = [\mathrm{E}(x_i x_j)]_{j=1}^{i}$. Finally, let us say that $\mathbf{x}_{-i}$ is the first $i-1$ entries of $\mathbf{x}$. $\mathbf{a}_i$ must satisfy

$$\begin{aligned}
\mathbf{m}_i &= \mathrm{E}\left(x_i \begin{bmatrix} \mathbf{x}_{-i} \\ x_i \end{bmatrix} \right) \\
\mathbf{m}_i &= \mathrm{E}\left(\mathrm{E}\left(x_i \begin{bmatrix} \mathbf{x}_{-i} \\ x_i \end{bmatrix} \middle| \mathbf{x}_{-i} \right) \right) \\
\mathbf{m}_i &= \sum_{\mathbf{x}_{-i} \in \{0,1\}^{i-1}} \mathrm{P}(\mathbf{x}_{-i}) \mathrm{P}(x_i = 1 \,|\, \mathbf{x}_{-i}) \begin{bmatrix} \mathbf{x}_{-i} \\ 1 \end{bmatrix}
\end{aligned}$$

When μ is the logistic link function L, this is

$$\mathbf{m}_i = \sum_{\mathbf{x}_{-i} \in \{0,1\}^{i-1}} \frac{1}{c} L\left(\mathbf{x}_{-i}^T A_{-i} \mathbf{x}_{-i}\right) L\left(\mathbf{a}_i^T \begin{bmatrix} \mathbf{x}_{-i} \\ 1 \end{bmatrix} \right) \begin{bmatrix} \mathbf{x}_{-i} \\ 1 \end{bmatrix}$$

Where c is the appropriate normalizing constant. Let us define the quantity on the right in the last line as $f(\mathbf{a}_i)$. This equation $f(\mathbf{a}_i) = \mathbf{m}_i$ can be solved by successive Newton-Raphson iterations defined by

$$\mathbf{a}_i^{(k+1)} = \mathbf{a}_i^{(k)} - \left[J\left(\mathbf{a}_i^{(k)} \right) \right]^{-1} \left[f\left(\mathbf{a}_i^{(k)} \right) - \mathbf{m}_i \right]$$

where J is the Jacobian of f. To do so, the Jacobian matrix is calculated as

$$J\left(\mathbf{a}_i\right) = \sum_{\mathbf{x}_{-i} \in \{0,1\}^{i-1}} \frac{1}{c} L\left(\mathbf{x}_{-i}^T A_{-i} \mathbf{x}_{-i}\right) L'\left(\mathbf{a}_i^T \begin{bmatrix} \mathbf{x}_{-i} \\ 1 \end{bmatrix} \right) \begin{bmatrix} \mathbf{x}_{-i} \\ 1 \end{bmatrix} \begin{bmatrix} \mathbf{x}_{-i}^T & 1 \end{bmatrix}$$

With 2^{i-1} possible values for $\mathbf{x}_{-i}$, this can quickly become computationally expensive. Instead, with a series of values $\mathbf{x}_{-i}^{(k)} \sim \mathbf{x}_{-i}$, the simulated values

$$f\left(\mathbf{a}_i\right) \approx \frac{1}{K} \sum_{k=1}^{K} L\left(\mathbf{a}_i^T \begin{bmatrix} \mathbf{x}_{-i}^{(k)} \\ 1 \end{bmatrix} \right) \begin{bmatrix} \mathbf{x}_{-i}^{(k)} \\ 1 \end{bmatrix}$$

and

$$J\left(\mathbf{a}_i\right) \approx \frac{1}{K} \sum_{k=1}^{K} L'\left(\mathbf{a}_i^T \begin{bmatrix} \mathbf{x}_{-i}^{(k)} \\ 1 \end{bmatrix} \right) \begin{bmatrix} \mathbf{x}_{-i}^{(k)} \\ 1 \end{bmatrix} \begin{bmatrix} [x_{-i}^{(k)}]^T & 1 \end{bmatrix}$$

can be used.

Though, in theory, A should always exist, in practice limited accuracy may dictate that Newton-Raphson method won't converge. In the simulation method in particular, if K is not large enough, the equations can be very difficult or impossible to solve. When the intial problem can't be solved, besides increasing the simulation size or numerical accuracy, one can instead solve the relaxed problem $f(\mathbf{a}) = \mathbf{m}_i^*(\tau)$, where, for $\tau \in [0,1]$

$$\mathbf{m}_i^*(\tau) = (1-\tau)\mathbf{m}_i + \tau \begin{bmatrix} 0 & \dots & 0 & M_{ii} \end{bmatrix}^T$$

When $\tau = 0$, this yields the original problem, while when $\tau = 1$, it is treats x_i as independent of $\mathbf{x}_{-i}$. The latter will always have the solution

$$\mathbf{a}_i = \begin{bmatrix} 0 & \dots & 0 & \log\left(\frac{M_{ii}}{1-M_{ii}}\right) \end{bmatrix}^T$$

The hope is that for some τ close to 0, convergence can be achieved, only causing a slight distortion from the desired cross-moments.

3.2 Generating Binary Knockoffs

This theory provides the tools to generate random binary knockoffs. This is done is done sequentially as two steps, the first of which is to come up with the target covariance matrix Σ_L for $\mathbf{x}_L = [\mathbf{x}\ \tilde{\mathbf{x}}]$. This is extremely similar, though not identical, to the method with the original knockoffs, where $X_L^T X_L$ after normalization was targeted. However, much the same method as before can be used. Let the vector of empirical means of the columns of X is $\mathbf{m} = \frac{1}{n}X^T\mathbf{1}$ and the empirical covariance matrix of X is $\Sigma = \frac{1}{n}X^TX - \mathbf{m}\mathbf{m}^T$. Much as before with the original knockoffs, both the equi-correlated and SDP method can be used to select $\mathbf{s}$ such that $\text{diag}\{\Sigma\} - \mathbf{s}$ is small and

$$\Sigma_L = \begin{bmatrix} \Sigma & \Sigma - \text{diag}\{\mathbf{s}\} \\ \Sigma - \text{diag}\{\mathbf{s}\} & \Sigma \end{bmatrix} \succeq 0$$

While with the original knockoffs, the result was small values for $X_i^T\tilde{X}_i$, now the result is low covariance between x_i and $\tilde{x}_i$.

Additionally though, the moment condition must hold for Σ_L to correspond to a proper random binary vector. If $\mathbf{m}_L = \text{E}\left(\mathbf{x}_L\right) = [\mathbf{m}\ \mathbf{m}]$, then the desired cross-moment matrix of the joint distribution is

$$M_L = \Sigma_L + \mathbf{m}_L\mathbf{m}_L^T$$

So it must be the case that

$$\max\{0, \mathbf{m}_{L,i} + \mathbf{m}_{L,j} - 1\} \leq \Sigma_{L,ij} + \mathbf{m}_i\mathbf{m}_j \leq \min\{\mathbf{m}_{L,i}, \mathbf{m}_{L,j}\}$$

However, since Σ and $\mathbf{m}$ are from an actual random binary vector, the entries of Σ satisfy this property by default. Only off diagonal entries which have had $\mathbf{s}$ subtracted from them could possibly violate this condition. Furthermore, for these entries, the goal is to make $\Sigma_{L,ij}$ as close to 0 as possible. Since the condition is always satisfied in a neighborhood of $M_{L,ij} = \mathbf{m}_{L,j}\mathbf{m}_{L,i}$, this implies that this method should normally be satisfied. This is the principle advantage minimizing the $\text{cor}(x_i, \tilde{x}_i)$ rather than $\text{E}[x_i\tilde{x}_i]$. For the SDP method though, the restriction can be built directly into the optimization problem. If $\sigma = \text{diag}\{\Sigma\}^{\frac{1}{2}}$ is the vector of standard deviations, the updated problem is

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{1} - \text{diag}\{\sigma\}^{-1}\mathbf{s}\|_1 \\ \text{subject to} \quad & 0 \preceq \text{diag}\{\mathbf{s}\} \preceq 2\Sigma \\ & 2\mathbf{m}_i - 1 \leq \Sigma_{ii} - \mathbf{s}_i + \mathbf{m}_i^2 \leq \mathbf{m}_i \quad \forall i \end{aligned}$$

the factor of $\text{diag}\{\sigma\}^{-1}$ in the objective is to account for the x_i not being normalized, as was the case with the original knockoffs.

3.2.1 Fitting via μ-conditionals

Once a proper cross-moment matrix M_L has been selected, the most obvious approach to generate knockoffs is to apply the Newton-Raphson method to fit the matching A matrix for the Ising model. This can be used to generate $\tilde{\mathbf{x}}$ sequentially conditioned on x, as $\tilde{x}_i \,\big|\, \mathbf{x}, \tilde{x}_1, \ldots, \tilde{x}_{i-1}$. However, there are a few variations of this process which are preferable depending on the situation.

1. If the whole A matrix is being fit, then the upper half of the matrix, corresponding to $\mathbf{x}$, can be fit via the regression method, since they arise from real data X. This will tend to be quicker, since the calculation lends itself to parallel computing, and the data set will often be smaller than the space of all binary vectors or a simulation of that space. The second half of the matrix still must be fit via Newton-Raphson.

2. One can avoid fitting the first half of the A matrix entirely though if performing Newton-Raphoson with by using the actual observed values in X to simulate the first half of $\mathbf{x}_L$. This can be done by either bootstrapping rows from X or using some number of copies of X exactly. The advantage of this is the saved computation and that $\tilde{x}$ should be built to match the correlation condition more exactly. The downside is that, since these values won't be from the Ising model, the theoretical guarantee of a suitable A may no longer hold. As well, if n isn't big enough, the equations will quickly become difficult to solve, since the observed samples will provide a noisy and incomplete sample of the probability space.

3. As A is being fit, even the parts of $\mathbf{x}_{-i}$ corresponding to knockoffs need not be redrawn for each Newton-Raphson iteration or each successive i. To fit $\mathbf{a}$ for feature i, the $\mathbf{x}_{-i}^{(k)}$ will remain constant; then, once $\mathbf{a}$ has been fit, the $x_i^{(k)}$ need only be drawn once. This saves computation, and the multiplication

$$\begin{bmatrix} \mathbf{x}_{-i}^{(k)} \\ 1 \end{bmatrix} \begin{bmatrix} [\mathbf{x}_{-i}^{(k)}]^T & 1 \end{bmatrix}$$

needs only be partially recalculated for each Newton-Raphson iteration since the upper left portion, $\mathbf{x}_{-i}^{(k)}[\mathbf{x}_{-i}^{(k)}]^T$, won't change. This may result in slight deviations in any particular draw compounding into a more severe error over time.

4. As p grows large, and K necessarily along with it, the cost of calculating the Jacobian matrix grows far quicker than the cost of calculating $f(\mathbf{a})$. This is since it has $O(Kp^2)$ complexity compared to $O(Kp)$ complexity for $f(\mathbf{a})$, quickly becoming prohibitive. One might instead estimate the full Jacobian based on first differences, but in practice an even simpler method called Anderson mixing, which avoids additional computation of $f(\mathbf{a})$, works well. Anderson mixing is one of several related quasi-Newtonian methods, which performs Newton-Raphson iteration k with an approximation of the inverted inverted Jacobian labeled G_k. Thus, to solve for a variable $\mathbf{z}$ such that $f(\mathbf{z}) = \mathbf{0}$, the best approximation for the zero is updated as $\mathbf{z}_{k+1} = \mathbf{z}_k - G_k f(\mathbf{z}_k)$ at each iteration.

   Anderson mixing does this implicitly, with no extra recalculation of f, by approximating f with the affine space spanning the previous m evaluations of f. These values are $f(\mathbf{z}_k)$ through $f(\mathbf{z}_{k-m+1})$. The best approximation for $f(\mathbf{z}) = 0$ in the affine space which spans them is defined by

$$\alpha = \underset{\mathbf{w}:\, \sum_{i=1}^{m} w_i = 1}{\arg\min} \left\| \sum_{i=1}^{m} w_i f(\mathbf{z}_{k-i+1}) \right\|_2$$

   This leads to the updated approximation

$$\mathbf{z}_{k+1} = \sum_{i=1}^{m} \alpha_i \mathbf{z}_{k-i+1}$$

   G_k doesn't need to be calculated here, and the quickest implementation finds α without calculating it. However, it's presence can be seen through an equivalent formulation. Let $\Delta\mathbf{f}_i = f(\mathbf{z}_k) - f(\mathbf{z}_{k-i})$, $\Delta\mathbf{z}_i = \mathbf{z}_k - \mathbf{z}_{k-i}$,

$$\mathcal{F} = [\Delta\mathbf{f}_1 \ldots \Delta\mathbf{f}_{m-1}]$$

   and

$$\mathcal{Z} = [\Delta\mathbf{z}_1 \ldots \Delta\mathbf{z}_{m-1}]$$

   This means that α has been replaced by θ such that

$$\theta \;=\; \underset{\mathbf{w}}{\arg\min} \, \|f(\mathbf{z}_k) - \mathcal{F}\mathbf{w}\|_2 \;=\; (\mathcal{F}^T\mathcal{F})^{-1}\mathcal{F}^T f(\mathbf{z}_k)$$

   so

$$\mathbf{z}_{k+1} \;=\; \mathbf{z}_k - \mathcal{Z}\theta \;=\; \mathbf{z}_k - \mathcal{Z}(\mathcal{F}^T\mathcal{F})^{-1}\mathcal{F}^T f(\mathbf{z}_k)$$

   Which yields the approximated inverse Jacobian $G_k = \mathcal{Z}(\mathcal{F}^T\mathcal{F})^{-1}\mathcal{F}^T$

In practice, it seemed that combining the first, third, and fourth of these variations leads to the most successful technique. However, should n be large enough to provide a good approximation of the probability space, the second variation also saves a lot of time.

3.2.2 Fitting via Quadratic Programing

Random binary knockoffs which produce the desired cross-moment matrix M_L need not arise from a generative model for $\tilde{\mathbf{x}}$ however. It is sufficient to have a random matrix $\tilde{X} \in \{0,1\}^{n \times p}$ such that

$$\mathrm{E}_{\tilde{X}}(X^T \tilde{X}) = \Sigma - \mathrm{diag}\{s\} \quad \& \quad \mathrm{E}_{\tilde{X}}(\tilde{X}^T \tilde{X}) = \Sigma$$

The simplest way to do this is based on a probability matrix $P \in (0,1)^{n \times p}$ where each X_{ij} is drawn independently as $X_{ij} \sim \text{Bernoulli}(P_{ij})$. This requirement can be stated as the quadratic feasibility problem

$$\begin{aligned} \text{exists} \quad & P \\ \text{subject to} \quad & X^T P = \Sigma - \mathrm{diag}\{s\} \\ & P^T P = \Sigma \\ & 0 \leq P \leq 1 \end{aligned}$$

If such a matrix exists and can be found, then $\tilde{X}$ can easily be generated. However, the constraints likely result in infeasibility much of the time. Instead, one can try to minimize how far the expected moments given P would be from those desired for M_L. This would take the form

$$\begin{aligned} \text{minimize} \quad & \|X^T P - (\Sigma - \mathrm{diag}\{s\})\|_{fro}^2 + \|P^T P - \Sigma\|_{fro}^2 \\ \text{subject to} \quad & \mathbf{1}^T P = \mathbf{m}_L \\ & 0 \leq P \leq 1 \end{aligned}$$

Here, the marginal distribution of $\tilde{X}_i$ will be the same as X_i, and the cross-moments shouldn't deviate to far from M_L. This problem can be stated as the quadratic program via the introduction of slack variables:

$$\begin{aligned} \text{minimize} \quad & \|W\|_{fro}^2 + \|V\|_{fro}^2 \\ \text{subject to} \quad & -W \leq X^T P - (\Sigma - \mathrm{diag}\{s\}) \leq W \\ & -V \leq P^T P - \Sigma \leq V \\ & \mathbf{1}^T P = \mathbf{m}_L \\ & 0 \leq P \leq 1 \end{aligned}$$

Though there are nice tools and theory for solving quadratic programs, $n \times p$ would have to be fairly small for this optimization problem to be computationally reasonable in practice. Ideally, there would be a method to breakdown this problem into a series of smaller optimization problems which would yield the same solution. An optimist might hope that iterating through the columns of P, optimizing each to hit its desired cross-moments individually conditioned on the estimate of the rest of P, would achieve this. In practice, this does seem to yield reasonable results, and is far quicker than other methods for large p. In particular, the following method seemed to work well:

1. Optimize initial P_i as such that $X_j^T P_i = M_{ij}$ if $i \neq j$ and $X_i^T P_i = \sigma_i(1 - s)$.

$$\begin{aligned} \text{minimize} \quad & \big(\tfrac{1}{n}X_i^T P_i - (M_{ij} - \sigma_i(1-s))\big)^2 + \textstyle\sum_{i \neq j}(\tfrac{1}{n}X_j^T P_i - M_{ij})^2 \\ \text{subject to} \quad & \tfrac{1}{n}\mathbf{1}^T P_i = M_{ii} \\ & 0 \leq P_i \leq 1 \end{aligned}$$

2. Draw initial $\tilde{X}_i$ as independent Bernoulli with probability P_i.

3. Iterate through $1 \leq i \leq p$, fitting new P_i such that

$$\begin{aligned} \text{minimize} \quad & \big(\tfrac{1}{n}X_i^T P_i - (M_{ij} - \sigma_i(1-s))\big)^2 + \textstyle\sum_{i \neq j}(\tfrac{1}{n}X_j^T P_i - M_{ij})^2 + \textstyle\sum_{i \neq j}(\tfrac{1}{n}\tilde{X}_j^T P_i - M_{ij})^2 \\ \text{subject to} \quad & \tfrac{1}{n}\mathbf{1}^T P_i = M_{ii} \\ & 0 \leq P_i \leq 1 \end{aligned}$$

   After each P_i is fit, redraw $\tilde{X}_i$ as independent Bernoulli with probability P_i.

Drawing the $\tilde{X}_i$ after each step, rather than just leaving the P_i, seems counter intuitive, but under experimentation, it seemed to provide better results. As well, though the third step may be repeated several times, in practice this didn't lead to improvement.

4 Binary Knockoff Performance

While the theory for original knockoffs and random binary variable generation suggests at how the binary knockoffs should perform, seeing their performance in practice still provides invaluable insight to their uses and limitations. There are three major areas in which this is useful; how well do the binary knockoffs' empirical cross-moment matrices converge to their expectation, how well is FDR controlled for linear regression, and how well, if at all, is FDR controlled for other GLMs?

4.1 Convergence to Theoretical cross-moments

In a certain sense, the question of the convergence of the binary knockoffs' empirical cross-moments is the least interesting of these questions. Assuming the Ising model coefficient A or the probability matrix P was successfully fit, the cross-moment matrices' will converge to their expectation in probability for increasingly large values of n. However, especially with the limited numerical accuracy available to fit A or P, the rate at which this convergence occurs and the size of the deviation is an issue of paramount importance. As is seen in Figure 3, this convergence occurs at similar rates irrespective of p, with fairly meaningful deviation until n has reached a few thousand.

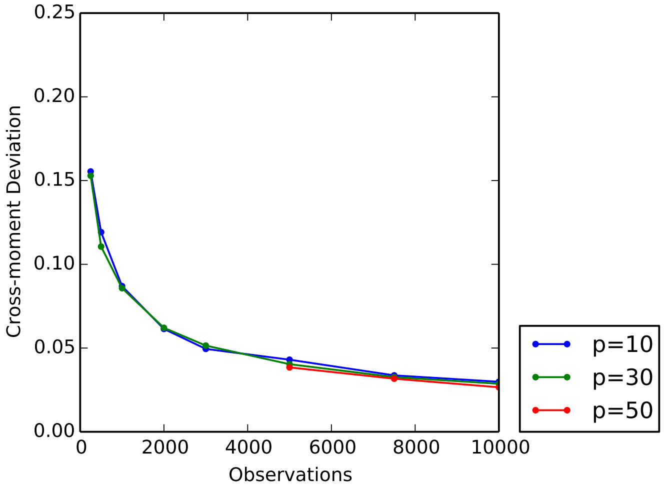

Figure 3: Measures how much far the empirical cross-moment matrix $\frac{1}{n}X_L^T X_L$ is from the desired cross moment matrix M_L for random binary knockoffs for various n and p, averaged over 50 simulations of X from the Ising model. Where $W = \Sigma - \text{diag}\{\mathbf{s}\} + \mathbf{m}\mathbf{m}^T$ is the top quadrant of M_L, deviation is calculated as $(\|X^T\tilde{X} - W\|_{fro} + \|\tilde{X}^T\tilde{X} - M\|_{fro})/(\|W\|_{fro} + \|M\|_{fro})$

4.2 Performance in Linear Regression

When the knockoff filter is used in linear regression, we see that the original knockoffs are still superior to the binary knockoffs. As figure 4 shows, while they both fail to control the FDR when there are only a few true predictors[3], the original knockoffs properly control FDR for more valid predictors. That it does so rather conservatively is a reflection of the relatively low number of total features compared to the sparsity. On the other hand, though mirroring the trend with the original knockoffs, the binary knockoffs fail to properly control FDR. They select too many variables, which gives higher power at the expense of proper FDR control. As figure 5 shows, both methods make related, but no where near identical variable selections.

This is disappointing, but shouldn't be surprising. The binary knockoffs only replicate the correlation structure of the original features in expectation, rather than exactly. Thus, for any given value n, the empirical correlation of the combined design matrix X_L won't exactly be Σ_L, and the method won't work perfectly. However, even here, the actual FDR is only somewhat hire than the desired .2, and for larger values of n, the empirical correlation will get closer to Σ_L, and the FDR will be controlled better.

[3] This issue is addressed in the original paper; a slightly more conservative filter called knockoff+ avoids this problem

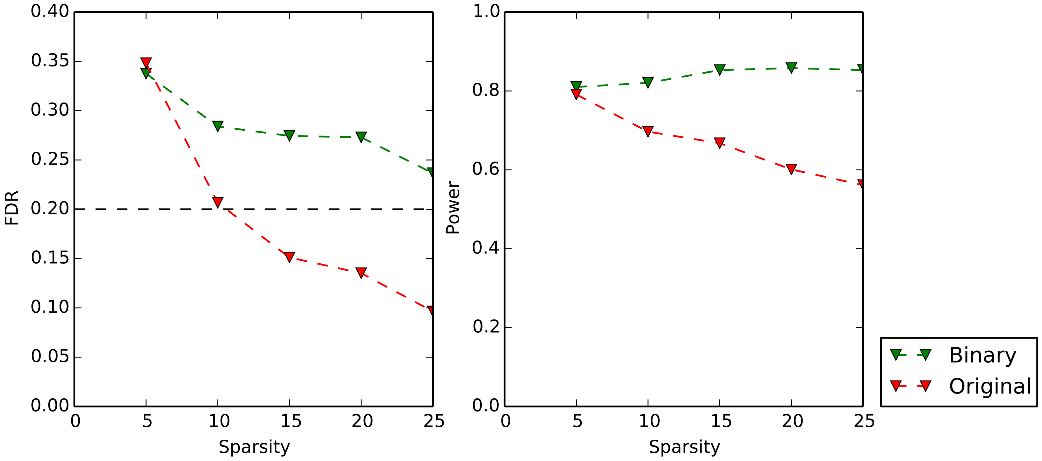

Figure 4: The mean FDR and power of binary and original knockoffs. Taken over 200 simulations of linear regression on $n = 1000$, $p = 50$, targeted FDR $q = .2$, and varying sparsity, or number of true predictors. X is generated from Ising model while y is normally distributed about $X\beta$.

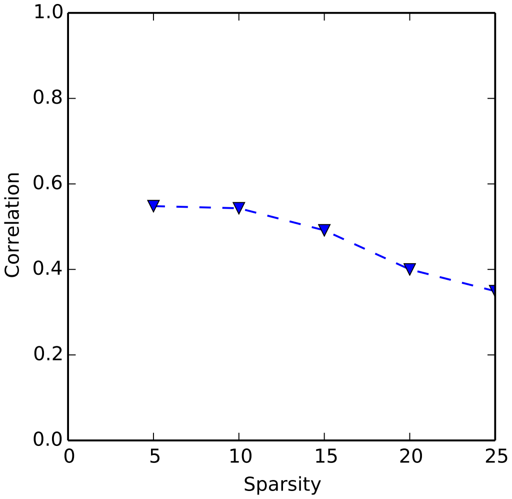

Figure 5: The mean correlation between variables selected using binary and original knockoffs. Taken over 200 simulations of linear regression on $n = 1000$, $p = 50$, targeted FDR $q = .2$, and varying sparsity, or number of true predictors. X is generated from Ising model while y is normally distributed about $X\beta$.

4.3 Performance in Logistic Regression

In logistic regression, the binary knockoffs do seem to provide an improvement over the original knockoffs, as can be seen in figure 6. Though neither control FDR properly, using binary knockoffs result both in higher power and lower FDR. As well, while the original knockoffs do considerably worse in logistic regression than in linear regression, the binary knockoffs perform only a little worse, having similar FDR and somewhat lower power. This suggests that much of the excess FDR is due the deviation from the desired correlation matrix. It should be noted though that this simulation is very favorable towards the binary knockoffs; the design matrix X is generated from the Ising model, which is also the model used to generate the binary knockoffs. It is quite likely that for X generated with significant high order interactions, the binary knockoffs wouldn't perform as well, though still likely better than the original knockoffs.

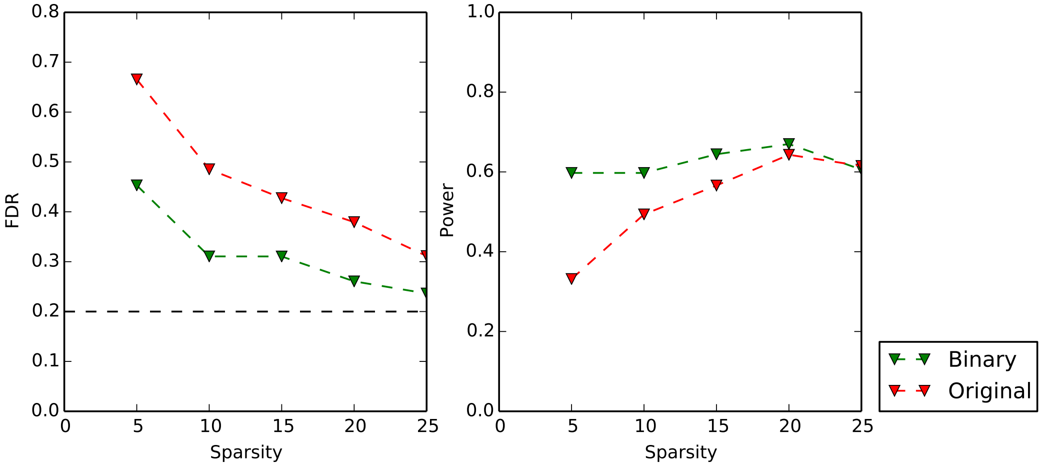

Figure 6: The mean FDR and power of binary and original knockoffs. Taken over 200 simulations of logistic regression on $n = 1000$, $p = 50$, targeted FDR $q = .2$, and varying sparsity, or number of true predictors. X is generated from Ising model while y is Bernoulli with $p = L(X\beta)$.

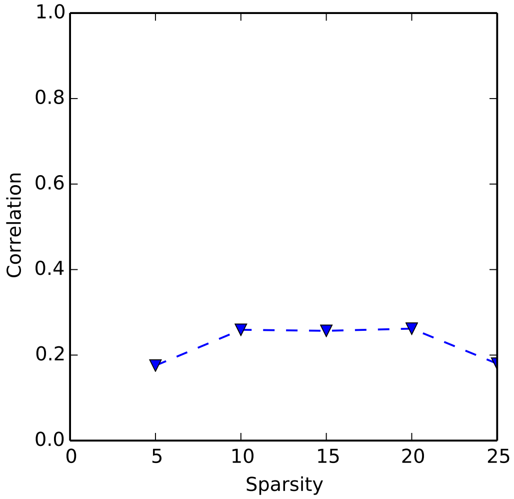

Figure 7: The mean correlation between variables selected using binary and original knockoffs. Taken over 200 simulations of logistic regression on $n = 1000$, $p = 50$, targeted FDR $q = .2$, and varying sparsity, or number of true predictors. X is generated from Ising model while y is Bernoulli with $p = L(X\beta)$.

5 Discussion

Binary knockoffs, as a practical tool for statistical inference, are still a work in progress. Currently, they work best for high values of n, for which the empirical cross-moment matrix deviates little, and relatively small values of p, for which the computational time is not prohibitive. Considering that FDR control in feature selection is most essential when p is large, this is not a edifying state of affairs. However, the approach suggests paths forward to a much more complete tool.

5.1 Dealing With Small n

With random generation of knockoffs, random variation from desired cross-moments is necessarily going to be an issue. However, there are likely ways to mitigate this problem by using multiple draws of $\tilde{X}$. For instance, if the A or P matrices were fit once, several draws of $\tilde{X}$ could be made from it, and a variable selection could be made for each of the corresponding X_L. Then these could be combined into a single variable selection, most obviously by selecting the variables which were selected for the majority of the X_L.

5.2 Computational Issues

The biggest drawback of this method is its computational requirements. As p increases, a larger and larger matrix A must be fit for the Ising model as well as a larger P for the quadratic program. For the Ising model, K must similarly grow to provide accurate estimates of the probability space. This causes an explosion in computation time. Some of this can likely be mitigated by improvements in implementation, and the most time intensive issue, calculating $f(\mathbf{a})$, can be easily parallelized. A simpler method would still be desirable though. A method based on optimizing each column of P individually is thus appealing. Certainly, in cases such as the EM algorithm, methods have been developed which have strong guarantees for global optimization using such piecewise methods. If a similar development could be made here, or merely a method that hits the desired cross-moments more closely in practice, the result would be far more useful tool.

5.3 Random Binary Variables With Higher Order Interactions