A Normalizing flows

Normalizing flows (Papamakarios et al., 2021) represent a general framework for density estimation of a multi-dimensional distribution with arbitrary dependencies. Briefly, suppose $X \sim \mathcal{P}_X$ is a random variable in $\mathbb{R}^d$. Now, let $Z \sim \mathcal{N}(0, I_d)$ be a multivariate standard normal distribution. We assume there exists a mapping G that is triangular, increasing, and differentiable such that

$$G(X) = Z.$$

A formal treatment of when such a G exists can be found in Bogachev et al. (2005). However, a sufficient condition is that the density of X is greater than 0 on $\mathbb{R}^d$ and the cumulative density function of X_j, conditional on the previous components $X_{\leq j}$, is differentiable with respect to $X_j, X_{\leq j}$ (Papamakarios et al., 2021):

$$U_i = G_i(X) \equiv F_i(X_i | X_{\leq i})$$

From this construction, each U_i is independent of all previous U_i and has distribution $\text{Unif}[0, 1]$. From there, we simply set $Z_i = \Phi^{-1}(U_i)$, where Φ is the CDF of the standard normal.

Since $G_i(X)$ depends only on the elements in X up to i, it is triangular. Because $p_X > 0$, the conditional cdfs are strictly increasing, so G is an increasing map. Finally, since each cdf is differentiable, the entire map G is differentiable, and its Jacobian is non-zero.

Because of the inverse mapping theorem, G is invertible and we can write

$$X = G(Z).$$

Normalizing flows are a collection of distributions that parameterize a family of invertible, differentiable transformations G_θ from a fixed base distribution Z to an unknown distribution X. Using the change-of-variables theorem, we can express the distribution of X in terms of the base distribution density p_Z and the transformation G_θ:

$$p_\theta(X) = p(G_\theta(X)) \left| \det \left(\frac{\partial G_\theta(X)}{\partial X} \right) \right|$$

where $\frac{\partial G_\theta(X)}{\partial X}$ is the Jacobian of G. The goal is to find a parameter value $\hat{\theta}$ that maximizes the likelihood of the observed X:

$$\hat{\theta} = \arg\max_\theta p_\theta(X).$$

A key feature of normalizing flows is that they are composable.

A.1 Flow Architecture

In experiments, the first layer G is a Gaussianization flow (Meng et al., 2020) applied elementwise:

$$G_j(X_j) = \Phi^{-1} \left(\sum_{m=1}^{M} \sigma \left(\frac{X_j - \mu_{j,m}}{s_{j,m}} \right) \right),$$

where Φ^{-1} is the standard normal inverse CDF. With sufficiently large M, this Gaussianization layer can approximate any univariate distribution. This is composed with a Masked Autoregressive Flow (MAF) F (Papamakarios et al., 2017), which consists of MADE layers interspersed with batch normalization and reverse permutation layers:

$$\begin{aligned}
\text{MADE}_{j,k} &= (X_j - \mu_{j,k}) \exp(-\alpha_{j,k}) \\
\text{where } \mu_j &= f_{\mu_{j,k}}(X_{<j}) \\
\alpha_j &= f_{\alpha_{j,k}}(X_{<j}) \\
F &= \text{MADE}_{j,K} \circ \text{BatchNorm} \circ \text{Reverse} \circ \text{MADE}_{j,K-1} \circ \cdots \text{BatchNorm} \circ \text{Reverse} \circ \text{MADE}_{j,1}
\end{aligned}$$

Here, f_{μ_j} and f_{α_j} are fully connected neural networks.

B Proof of convergence

Theorem 1. *Let $X \in \mathbb{R}_{N\times D}$ be a random feature matrix, where each row $X_{i,\cdot}$ is independent and identically distributed; $x \in \mathbb{R}_{N\times D}$ be the observed feature matrix; and α_j be the p-value as defined in Equation* (4) *with test statistic $T_j(X)$. Suppose there exists a sequence of functions $(G^n)_{n=1}^{\infty}$ and a base random variable Z satisfying the following conditions:*

1. *Each G^n is continuously differentiable and invertible.*
2. *$G^n \to G$ pointwise for some map G that is triangular, increasing, continuously differentiable, and satisfies $G(X_{i,\cdot}) \overset{D}{=} Z$.*

For $n = 1, 2, \dots$, let X^n be the random feature matrix where each row i is independent and has distribution $X^n_{i,\cdot} = (G^n)^{-1}(Z)$. Then, the p-value in Equation (5) *calculated using K MCMC samples targeting $X^n_{\cdot,j} \mid X^n_{\cdot,-j} = x_{\cdot,-j}$ converges to the correct p-value α_j with probability* 1.

The assumption that $G^n \to G$ depends on the universality of the family of normalizing flows chosen. Universality has been shown for a wide variety of normalizing flows (Huang et al., 2018; Meng et al., 2020; Kobyzev et al., 2020).

Proof of Theorem 1. Without loss of generality, we consider the first feature, which is indexed by $j = 1$. Let p_X be the density of each row of the matrix $X_{i,\cdot}$ and p_Z the density of the base variable Z. For each i.i.d observation at $i = 1, \dots, N$, we define F to be the cumulative distribution function of $X_{i,1}$ conditional on the other features $X_{i,-1} = x_{i,-1}$:

$$\begin{aligned} F(x_1) \triangleq \mathcal{P}(X_{i,1} \leq x_1 | X_{i,-1} = x_{i,-1}) &= \frac{\int_{-\infty}^{x_1} p_X(x_1', x_{i,-1}) dx_1'}{\int_{-\infty}^{\infty} p_X(x_1', x_{i,-1}) dx_1'} \\ &= \frac{\int_{-\infty}^{x_1} p_Z(G(x_1', x_{i,-1})) |\partial G(x_1', x_{i,-1})| dx_1'}{\int_{-\infty}^{\infty} p_Z(G(x_1', x_{i,-1})) |\partial G(x_1', x_{i,-1})| dx_1'}. \end{aligned} \tag{7}$$

For a particular mapping G^n, we define F^n analogously:

$$F^n(x_1) \triangleq \frac{\int_{-\infty}^{x_1} p_Z(G^n(x_1', x_{i,-1})) |\partial G^n(x_1', x_{i,-1})| dx_1'}{\int_{-\infty}^{\infty} p_Z(G^n(x_1', x_{i,-1})) |\partial G^n(x_1', x_{i,-1})| dx_1'}. \tag{8}$$

Since G^n and G are continuously differentiable,

$$p_Z(G^n(x_1', x_{i,-1})) |\partial G^n(x_1', x_{i,-1})| \to p_Z(G(x_1', x_{i,-1})) |\partial G(x_1', x_{i,-1})| \text{ as } n \to \infty. \tag{9}$$

Then, by the dominated convergence theorem, $F^n \to F$ pointwise.

Let $X^n_{i,1} \sim F^n$. Since $F^n \to F$ pointwise, and F is a distribution function, $X^n_{i,1}$ converges in distribution to $X_{i,1} \mid X_{i,-1} = x_{i,-1}$. Likewise, the joint distribution across all independent observations, written $X^n_{\cdot,1}$, converges in distribution to $X_{\cdot,1} \mid X_{\cdot,-1} = x_{\cdot,-1}$.

Now, let $\tilde{X}^n_{\cdot,1}$ be equal in distribution to $X^n_{\cdot,1}$, but sampled such that it is independent of the outcome Y. It follows from the reasoning above that $\tilde{X}^n_{\cdot,1}$ converges to the desired null distribution $\tilde{X}_{\cdot,1} | X_{\cdot,-1}$ as $n \to \infty$. Define $g_1(\tilde{x}_{\cdot,1}) \triangleq 1[T_1(X) < T_1([\tilde{x}_{\cdot,1}, X_{\cdot,-1}])]$. With the regularity condition that T_1 is discontinuous on a set of measure zero, the expectation converges:

$$\lim_{n\to\infty} \mathbb{E}_{\tilde{X}^n_{\cdot,1}}(g_1) \to \mathbb{E}_{\tilde{X}_{\cdot,1} | X_{\cdot,-1} = x_{\cdot,-1}}(g_1) = \alpha_1. \tag{10}$$

The Cesaro average of g calculated over MCMC samples that target the distribution of $\tilde{X}^n_{\cdot,1}$ under the probability law of G_n converges almost surely to $\mathbb{E}_{\tilde{X}^n_{\cdot,1}}(g_1)$ (Smith & Roberts, 1993). That is,

$$\lim_{K\to\infty} \hat{\alpha}_{j,K,n} = \lim_{K\to\infty} \frac{1}{K} \sum_{k=1}^{K} g_1(\tilde{X}_{\cdot,1,k}) = \mathbb{E}_{\tilde{X}^n_{\cdot,1}}(g_1) \; w.p.1. \tag{11}$$

Combining Equation (10) and Equation (11) gives the desired result. ☐

C Feature datasets

Name	Covariate	Response	N	D	# Relevant	Source
Gaussian Mixture	Synthetic	Synthetic	$100,000$	100	20	-
scRNA-seq	Real	Synthetic	$100,000$	100	10	10x Genomics (2017)
Soybean	Real	Real	$5,128$	$4,236$	-	Xavier et al. (2019)

Licensing All of the data used is available for personal use. Terms for the scRNA-seq data can be found here: `https://www.10xgenomics.com/terms-of-use`. The scRNA-seq data was accessed using scvi-tools (Gayoso et al., 2021), distributed under the BSD 3-Clause license. The soybean data is part of the SoyNAM R package (Xavier et al., 2019), distributed under the GPL-3 license.

D Architecture and training details for synthetic experiments

D.1 FlowSelect

For FlowSelect , the joint distribution was fitted with a GaussMAF normalizing flow as described in Appendix A. The first Gaussianization layer consisted of $M = 6$ clusters, followed by 5 layers of MAF. Within each MAF layer, the neural network consisted of three masked fully connected residual layers with 100 hidden units, followed by a BatchNorm layer.

We trained the Gaussianization layer first with 100 epochs and learning rate 1×10^{-3} within the ADAM optimizer. This allowed the Gaussianization layer to learn the marginal distribution of each feature. Then, we jointly trained the whole architecture with 100 epochs and learning rate 1×10^{-3} using ADAM.

MCMC We draw 1000 samples using a Metropolis-Hastings procedure. The proposal distribution is a random walk:

$$X^*_{i,j,k} \sim \mathcal{N}(\tilde{X}_{i,j,k-1}, \hat{\sigma}^2_j),$$

where $\hat{\sigma}^2_j$ is the sample conditional variance:

$$\begin{aligned} \hat{\sigma}^2_j &= \hat{\Sigma}_{j,j} - \hat{\Sigma}_{j,-j}\hat{\Sigma}^{-1}_{-j,-j}\hat{\Sigma}^T_{j,-j} \\ \text{where } \hat{\Sigma}_j &= \widehat{\text{Var}}(X) \end{aligned}$$

D.2 Variable selection methods

Linear For the linear response, we estimate a linear model with an L1 penalty (aka the LASSO) on training data:

$$\hat{\beta} = \arg\min_{\beta} \frac{1}{N}\|X\beta - Y\|^2_2 + \lambda \sum_{j=1}^{D} |\beta_j| \tag{12}$$

The penalization term λ is selected via 5-fold cross-validation.

Nonlinear For the nonlinear response, we fit a random forest on the training data. The hyperparameters are the defaults in the scikit-learn implementation.

Feature statistic If $\hat{f}(X)$ is the fitted regression function, then the feature statistic is the negative mean-squared error:

$$T(X, Y) = -\frac{1}{N}\|\hat{f}(X) - Y\|^2_2.$$

D.3 Competing methods

For DDLK (Sudarshan et al., 2020), KnockoffGAN (Jordon et al., 2019), and DeepKnockoffs, (Romano et al., 2020), we used the exact architecture and hyperparameter settings from their respective papers. For the ablation study in Section 5.3, we use the exact implementation in Tansey et al. (2021). For these methods, we used the code that the researchers graciously made publicly available:

Method	Link
DDLK	`https://github.com/rajesh-lab/ddlk/`
DeepKnockoffs	`https://github.com/msesia/deepknockoffs/`
HRT (MDN)	`https://github.com/tansey/hrt/`
KnockoffGAN	`https://github.com/firmai/tsgan/tree/master/alg/knockoffgan`

For MASS (Gimenez et al., 2019), we followed their described procedure and fit a mixture of Gaussians to the feature distribution using scikit-learn, selecting the number of components via the Akiake Information Criterion (AIC). We then used the `knockoffs` R package, available on CRAN, to sample knockoffs using the estimated parameters for each component.

For RANK (Fan et al., 2020), we estimate the sparse precision matrix using the Graphical LASSO (Friedman et al., 2008) implemented in sci-kit learn, using cross-validation to tune the regularization parameter. We then use the `knockoffs` R package to sample the knockoffs with this covariance.

E Architecture and training details for soybean GWAS

Discrete flows For the discrete flows in the soybean example, we use a single layer of MADE which outputs a dimension of size 4. μ is then set equal to the argmax of this output.

For training the flows, we use a relaxation of argmax with temperature equal to 0.1.

Discrete MCMC Each feature has $K = 4$ values, so we can enumerate all four possible states for each proposal and sample in proportional to these probabilities via a Gibbs Sampling procedure. Setting the probabilities leads to an acceptance rate of 1, and the samples are uncorrelated since the previous sample doesn't enter into the proposal distribution

Predictive model For the predictive model of each trait conditional on the SNPs, we use a fully connected neural network. This network has three hidden layers of size 128, 256, and 128. ReLU activations are used between each fully connected layer. Dropout is used on both the input layer and after each hidden layer with $p = 0.2$. The learning rate in ADAM was set to 1×10^{-5}, with early stopping implemented using a held-out validation set.

The feature statistic for each sample is the negative mean-squared error (MSE) for each observation.

Runtime To obtain sufficient resolution on roughly 4200 simultaneous tests, we drew 100,000 samples from our model. The runtime was 10 hours using a single NVIDIA 2080 Ti.

Selected SNPs Table 1 shows the SNPs selected by FLOWSELECT that are associated with oil content in soybeans.

Chromosome	SNP	p-value
4	Gm04_42203141	1.60e-04
5	Gm05_37467797	1.90e-04
8	Gm08_15975626	2.10e-04
14	Gm14_1753922	9.00e-05
14	Gm14_1799390	1.60e-04
14	Gm14_1821662	2.90e-04
18	Gm18_1685024	5.00e-05

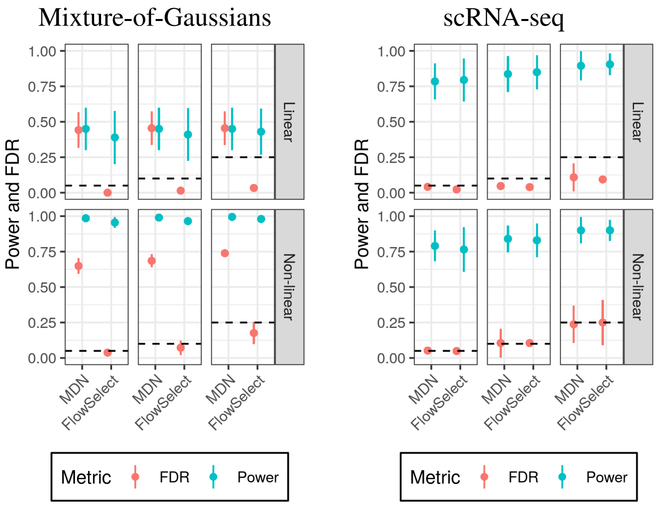
Table 1: Selected SNPs for soybean GWAS experiment.

F Runtime comparison of each controlled feature selection method

Method	Runtime (min)
DeepKnockoff	3.0
KnockoffGAN	3.73
MASS	12.6
DDLK	91.9
FlowSelect	59.4

Table 2: The median runtime for each method on the scRNA-seq data with $D = 100$ features and $N = 100,000$ observations. All experiments were implemented using PyTorch, except for KnockoffGAN, which was implemented in Tensorflow, and MASS, which we implemented using scikit-learn and the knockoffs R package. The experiments were conducted using an Intel Xeon Gold 6130 CPU and an NVIDIA GeForce RTX 2080 Ti GPU.

G Comparison to Holdout Randomization Test of Tansey et al. (2021)


Mixture-of-Gaussians

scRNA-seq

Figure 4: Comparison of FlowSelect to the HRT procedure in Tansey et al. (2021) which samples the complete conditionals using multiple mixture-density-networks (MDNs). Each column shows the power and observed false discovery rate (FDR) at targeted FDRs of 0.05, 0.1, and 0.25 (indicated by the dashed lines). The experimental settings for each dataset are the same as in Figure 2.

H Oracle Model-X

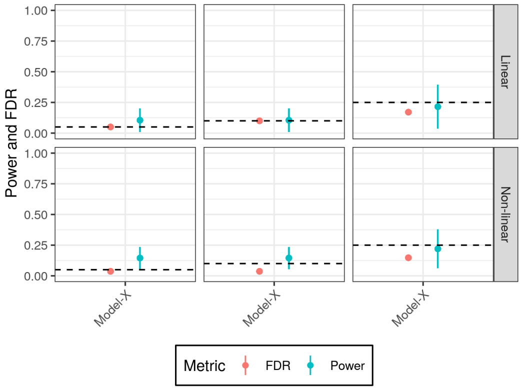


Figure 5: FDR control and power of Oracle Model-X knockoffs on the mixture-of-Gaussians dataset (compare to Figure 2).

I DDLK with true joint distribution

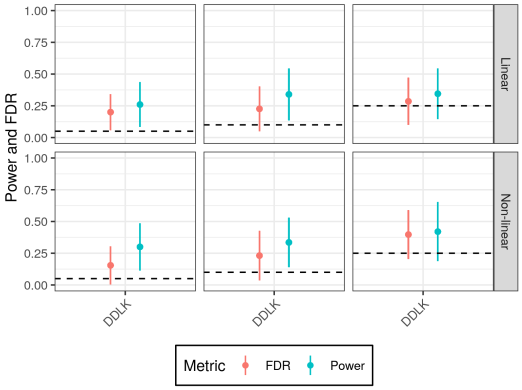


Figure 6: FDR control and power of DDLK on the mixture-of-Gaussians dataset using the ground truth feature density in training (compare to Figure 2).

J Observed Power and FDR control for given number of MCMC samples

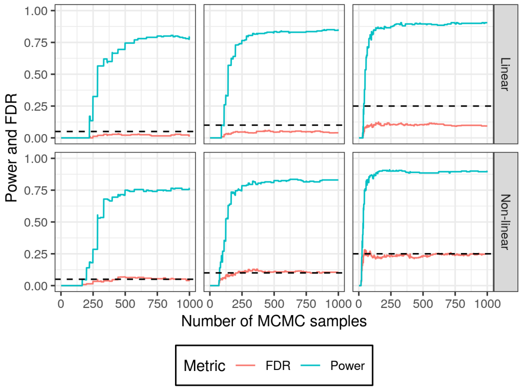


Figure 7: Power and FDR control of FLOWSELECT on the scRNA-seq dataset as a function of the number of MCMC samples at targeted FDRs of 0.05, 0.1, and 0.25 (indicated by the dashed lines). This suggests that the consequence of terminating the MCMC chain prematurely leads to a drop in power but FDR control is still maintained.

K Mixture-of-Gaussians results for FDR and Power under Sudarshan et al. (2020) settings

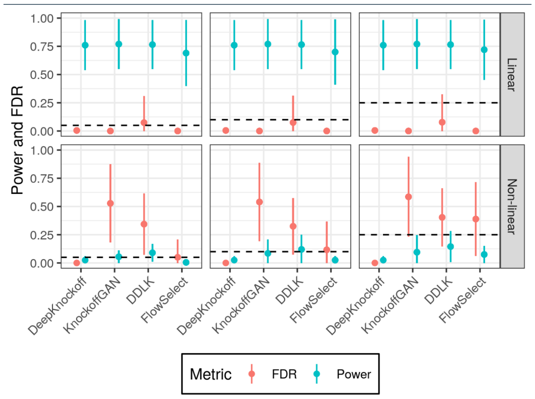


Figure 8: Mixture-of-gaussians setup with $\rho = (0.6, 0.4, 0.2)$ and $N = 2000$ to match the settings in Sudarshan et al. (2020). In the linear response setting, which matches the data-generating process of Sudarshan et al. (2020), all competing knockoff-based methods (i.e., DDLK, KnockoffGAN, and DeepKnockoff) as well as FlowSelect control the FDR at 5%, 10% and 25% levels and achieve a power of about 0.75. In the non-linear response setting, none of the methods control FDR, except for DeepKnockoffs which had nearly zero power. The good performance in the linear setting can be explained by the LASSO feature statistic shrinking most null features to zero since they have relatively low correlation. Since FDR control should hold for any response setting, these findings suggest that none of the methods do well in modeling the underlying distribution with $N = 2000$ observations.

L Learned normalizing flow mapping on mixture-of-Gaussians and scRNA-seq datasets

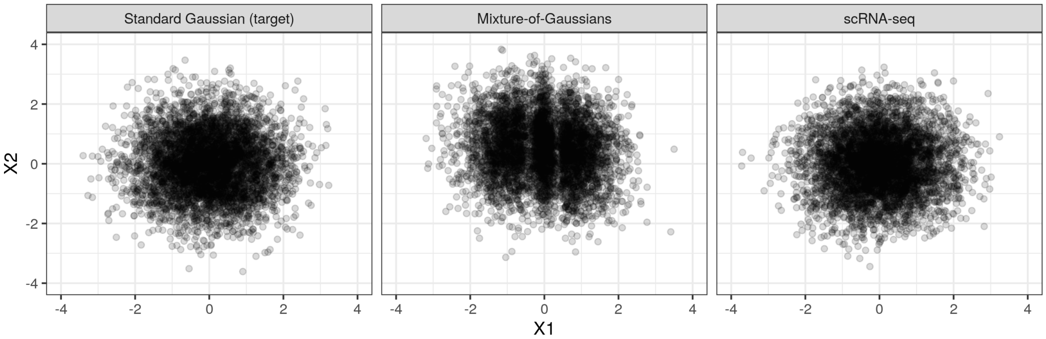


Figure 9: Plot of features mapped to flow space by the learned normalizing flow within FLOWSELECT with $j = 1$ on the x-axis and $j = 2$ on the y-axis. Mapped features are shown for the mixture-of-Gaussians and scRNA-seq datasets, and they are compared to samples from a true standard Gaussian distribution.