Ridge Regularization: An Essential Concept in Data Science

Trevor Hastie

Taylor & Francis
Taylor & Francis Group

Ridge Regularization: An Essential Concept in Data Science

Trevor Hastie

Department of Statistics, Department of Biomedical Data Science, Stanford University, Stanford, CA

ABSTRACT
Ridge or more formally ℓ_2 regularization shows up in many areas of statistics and machine learning. It is one of those essential devices that any good data scientist needs to master for their craft. In this brief *ridge fest*, I have collected together some of the magic and beauty of ridge that my colleagues and I have encountered over the past 40 years in applied statistics.

1. Ridge and Linear Regression

We first learn of ridge when we study linear regression. Ridge provides a remedy for an ill-conditioned $X^\top X$ matrix. If our $n \times p$ regression matrix X has column rank less than p (or nearly so in terms of its condition number, the ratio of largest to smallest singular value), then the usual least-squares regression equation is in trouble:

$$\hat{\beta} = (X^\top X)^{-1} X^\top y. \qquad (1)$$

The poor (large) condition number of X is inherited by $X^\top X$, which is either singular or nearly so, and here we try to invert it. The problem is that $X^\top X$ has some eigenvalues of zero or nearly zero, so inverting is not a good idea. So what we do is add a *ridge* on the diagonal—$X^\top X + \lambda I$ with $\lambda > 0$—which increases all the eigenvalues by λ and takes the problem away:

$$\hat{\beta}_\lambda = (X^\top X + \lambda I)^{-1} X^\top y. \qquad (2)$$

This is the ridge regression solution proposed by Hoerl and Kennard (1970) 50 years ago, and as we will see is alive and strong today.

We can write out the optimization problem that ridge is solving,

$$\underset{\beta}{\text{minimize}} \, \|y - X\beta\|_2^2 + \lambda \|\beta\|_2^2, \qquad (3)$$

where $\| \cdot \|_2$ is the ℓ_2 (Euclidean) norm. Some simple matrix calculus gets us from (3) to (2).

What value of $\lambda > 0$ shall we use? If our main concern is resolving the numerical issues with solving the least squares equation, then a small value suffices—say $\lambda = 0.001$ or perhaps that fraction of the largest eigenvalue of $X^\top X$.

The ridge remedy (2) comes with consequences. Under a true linear model, the ridge estimate is biased toward zero. It also has smaller variance than the OLS estimate. Selecting λ amounts to a bias-variance trade-off. We address this further in Section 3.

The ridge modification works in many situations where we fit linear models, and the effect is not as transparent as in (2).

- With GLMs we model $\eta(x) = \beta^\top x$ and $E(y|x) = g(\eta(x))$, and fit by maximum likelihood. If X is ill conditioned, this will lead to a Hessian that is flat (near zero) in some directions, and the Newton algorithm will be unstable. We can instead maximize

$$\ell(\beta; X, y) - \lambda \|\beta\|_2^2, \qquad (4)$$

which, as in (2), adds λI to the Hessian, and removes the problem. One example is logistic regression. Here even if X is well behaved, if the classes are separated in x-space, the usual maximum-likelihood estimator is undefined—some coefficients are infinite. A little bit of ridge resolves the issue.
- The same is true in the Cox model, multiclass logistic regression, and any other model linear in the parameters.
- In *wide-data* situations where $p \gg n$; for example, in genomics where the variables are single nucleotide polymorphisms (SNPs) and can number in the millions, and in document classification in the tens of thousands. Here regularization is essential, and λ requires careful tuning.

Typically we do not penalize the intercept in the linear model; in the case of least squares, we can center X and y upfront, and ignore the intercept. In other GLMs, it is handled not quite as simply, but is a detail which for now we will ignore.

We have expressed the ridge problem in *Lagrange* form in (3), as opposed to the more evocative *bound* form

$$\underset{\beta}{\text{minimize}} \, \|y - X\beta\|_2 \text{ subject to } \|\beta\|_2 \leq c. \qquad (5)$$

The two problems are of course equivalent: every solution $\hat{\beta}_\lambda$ to problem (3) is a solution to (5) with $c = \|\hat{\beta}_\lambda\|_2$. The lasso (Tibshirani 1996) uses an ℓ_1 rather than an ℓ_2 penalty in linear models to achieve sparsity:

$$\underset{\beta}{\text{minimize}} \, \|y - X\beta\|_2^2 + \lambda \|\beta\|_1. \qquad (6)$$

This penalty both shrinks coefficients like ridge, but also sets some to zero and thus selects. Figure 1 compares their constraint
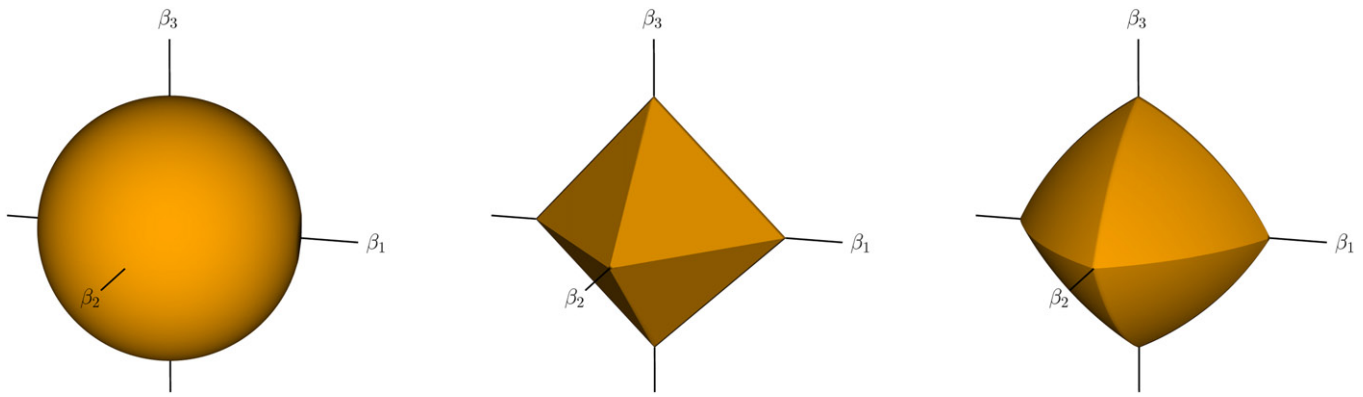
CONTACT Trevor Hastie ✉ hastie@stanford.edu ▣ Department of Statistics, Department of Biomedical Data Science, Stanford University, Stanford, CA 94305.

Figure 1. Constraint balls for ridge, lasso, and elastic-net regularization. The sharp edges and corners of the latter two allow for variable selection as well as shrinkage.

regions, which explains the ability of lasso to set coefficients to zero. The lasso has inspired ridge-like generalizations, such as the two mentioned here:

- *elastic net* (Zou and Hastie 2005) which mixes the ridge and lasso penalty

$$\lambda \left[(1 - \alpha) \|\beta\|_2^2 + \alpha \|\beta\|_1 \right]. \tag{7}$$

It still selects variables like the lasso, but deals more gracefully with correlated variables.

- *group lasso* (Yuan and Lin 2007) that does lasso-like selection for predefined groups of variables

$$\lambda \sum_{j=1}^{J} \|\theta_j\|_2; \tag{8}$$

here each θ_j is a vector of parameters. Notice the lack of the square on the ℓ_2 norms, which changes what would be an overall ridge penalty to a group-wise lasso. The *overlap group lasso* (Jacob, Obozinski, and Vert 2009; Hastie, Tibshirani, and Wainwright 2015) allows for hierarchy in selection, and has been used (by us, among others) for selecting interactions (Lim and Hastie 2015) and additive models (Chouldechova and Hastie 2015).

There is a Bayesian view of ridge regression. We assume $y_i|\beta$, $X = x_i \sim x_i^\top \beta + \epsilon_i$, with ϵ_i iid $N(0, \sigma_\epsilon^2)$. Here we think of β as random as well, and having a prior distribution $\beta \sim N(0, \sigma_\beta^2 I)$. Then the negative log posterior distribution is proportional to (3) with $\lambda = \sigma_\epsilon^2/\sigma_\beta^2$, and the posterior mean is the ridge estimator (2). The smaller the prior variance parameter σ_β^2, the more the posterior mean is shrunk toward zero, the prior mean for β. The Bayesian version of the lasso uses a Laplace prior, which puts much more mass at and around zero, and has wider tails than the Gaussian, per unit variance. Ridge therefore expects more variables in the model, and shrinks them all to stabilize variance. Lasso expects fewer variables, and hence is able to shrink those included less to stabilize variance.

2. Ridge Computations and the SVD

In many wide-data and other ridge applications, we need to treat λ as a tuning parameter, and select a good value for the problem at hand. For this task, we have a number of approaches available for selecting λ from a series of candidate values:

- With a validation dataset separate from the training data, we can evaluate the prediction performance at each value of λ.
- Cross-validation does this efficiently using just the training data, and leave-one-out (LOO) CV is especially efficient; see Section 7.
- C_p and unbiased risk estimates, where we correct the bias in the training risk at each λ.

Whatever the approach, they all require computing a number of solutions $\hat{\beta}_\lambda$ at different values of λ: the *ridge regularization path*. With squared-error loss as in (3), we can achieve great efficiency via the singular-value decomposition (SVD): $X = UDV^\top$. If X is $n \times p$, we use the *full* form of the SVD, with U $n \times n$ orthogonal, V $p \times p$ orthogonal and D $n \times p$ diagonal, with diagonal entries $d_1 \geq d_2 \geq \cdots \geq d_m \geq 0$, where $m = \min(n, p)$. Plugging this into (2), we get

$$\hat{\beta}_\lambda = V(D^\top D + \lambda I)^{-1} D^\top U^\top y$$
$$= \sum_{d_j > 0} v_j \frac{d_j}{d_j^2 + \lambda} \langle u_j, y \rangle. \tag{9}$$

So once we have the SVD of X, we have the ridge solution for *all* values of λ.

One can use this representation to show that if two or more variables are identical (a serious problem for linear regression), their ridge coefficients are identical (and sum to what would be the ridge coefficient if just one were included in the model). Similarly, correlated variables have their coefficients shrunk toward each other—the mechanism exploited by the elastic net.

It is also illuminating to look at the fitted values using (9):

$$\hat{y}_\lambda = \sum_{d_j > 0} u_j \frac{d_j^2}{d_j^2 + \lambda} \langle u_j, y \rangle. \tag{10}$$

The u_j form an orthonormal basis for the column space of X. If X has its column means removed (which we would have done if there were an unpenalized intercept in the model), then the u_j are the principal components of X. If $\lambda = 0$, then (10) says the fit is the projection onto the column space of X, using the principal components as a basis ($\langle u_j, y \rangle$ is the coordinate of y on the jth principal component). With $\lambda > 0$ the coordinates
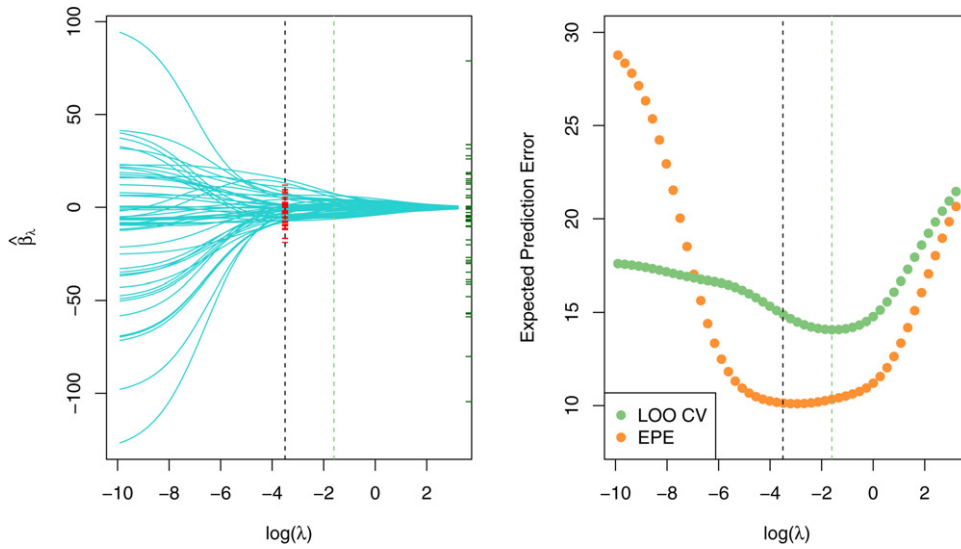
Figure 2. Simulation from a linear model with $n = 100$, $p = 54$, and SNR $= 3.3$. Left panel: Coefficients profiles $\hat{\beta}_\lambda$ versus log λ. The OLS coefficients are on the far left. The left vertical broken line is at the optimal EPE, and the red bars are the true coefficients. The second vertical line corresponds to the minimum LOO CV error. The dashes on the right axis are the James–Stein (uniformly shrunk) estimates. Right panel: The EPE of the fitted model on an infinite test dataset (orange): EPE$(\lambda) = \sigma^2 + E_X(\hat{f}_\lambda(X) - f(X))^2$, and the LOO CV curve estimated from the 100 training points.

of $\hat{y}_\lambda$ are shrunk by an increasing amount as we go through the succession of principal components.

When $n > p$ the ridge solution with $\lambda = 0$ is simply the OLS solution for β. When $p > n$, there are infinitely many least squares solutions for β, all leading to a zero-residual solution. This is because the least-squares estimating equations

$$\nabla_\beta \text{RSS}(\beta) = -2X^\top(y - X\beta) = 0 \quad (11)$$

are under determined (more unknowns than equations). But evidently from (9), we can get a unique solution

$$\hat{\beta}_{mn} = \sum_{d_j > 0} v_j \frac{1}{d_j} \langle u_j, y \rangle. \quad (12)$$

This is the least-squares solution with minimum ℓ_2 norm.

To see this, we go back to the SVD of X in the $p > n$ setting, and partition

$$D_{n \times p} = [\tilde{D}_{n \times n} : 0_{n \times (p-n)}] \text{ and } V_{p \times p} = [\tilde{V}_{p \times n} : \tilde{V}^\perp_{p \times (p-n)}]. \quad (13)$$

Reparameterizing $\beta = V\theta$ and plugging into (11), we get after some simplification

$$U^\top(y - U\tilde{D}\theta_1) = 0, \quad (14)$$

where $\theta = [\theta_1 : \theta_2]$ is partitioned like V. This defines $\hat{\theta}_1 = \tilde{D}^{-1}U^\top y$, and $\hat{\beta} = \tilde{V}\hat{\theta}_1 + \tilde{V}^\perp\theta_2$ for *any* θ_2 is a solution to (11). The minimum norm solution is obtained by setting $\theta_2 = 0$. We have for simplicity assumed that X has full row rank, and hence $\tilde{d}_j > 0$, $j = 1, \ldots, n$; if not we use a generalized inverse $\tilde{D}^+$ above, which inserts zeros in the inverse where $\tilde{d}_j = 0$.

3. Ridge and the Bias-Variance Trade-Off

The coefficients of ridge regression are explicitly shrunk toward the origin. If the data arise from a linear model

$$y_i = x_i^\top \beta + \varepsilon_i, \quad i = 1, \ldots, n \quad (15)$$

with iid zero-mean errors ε_i, then $\hat{\beta}_\lambda$ will be a biased estimate of β. If the x_i are assumed fixed, $n > p$, and X has full column rank, we can get an explicit expression for this bias from (9)

$$\begin{aligned} \text{Bias}(\hat{\beta}_\lambda) &= E\hat{\beta}_\lambda - \beta \\ &= \sum_{j=1}^p v_j \frac{\lambda}{d_j^2 + \lambda} \langle v_j, \beta \rangle. \end{aligned} \quad (16)$$

So the coordinates of the true coefficient along different principal components (of the training data) are differentially shrunk toward zero—the smaller the PC, the more the shrinkage. The bias increases in all components as λ increases. Note, when $p > n$, $\hat{\beta}_\lambda$ lies in the at most n-dimensional row space of X, and any component of β in the orthogonal component will contribute to the bias as well.

Similarly there is a nice expression for the covariance matrix under the sampling model (15):

$$\text{var}(\hat{\beta}_\lambda) = \sigma^2 \sum_{j=1}^p \frac{d_j^2}{(d_j^2 + \lambda)^2} v_j v_j^\top, \quad (17)$$

with $\sigma^2 = \text{var}(\varepsilon_i)$. With $\lambda = 0$ this is the usual OLS covariance $\sigma^2(X^\top X)^{-1}$ in factored form. Here we see that the covariance decreases uniformly (in a PSD sense) as λ increases.

These play off each other when we make predictions at new locations x_0, leading to a bias-variance trade-off.

$$\begin{aligned} \text{MSE}(x_0, \lambda) &= E(x_0^\top \hat{\beta}_\lambda - x_0^\top \beta)^2 \\ &= x_0^\top \text{var}(\hat{\beta}_\lambda)x_0 + [x_0^\top \text{Bias}(\hat{\beta}_\lambda)]^2. \end{aligned} \quad (18)$$

For the expected prediction error (EPE) we add σ^2, the variance of the target response. Even when the model is correct, a shrunken estimate can perform considerably better than the OLS solution if the bias and variances are traded off correctly. This is especially true in cases when p/n is large and/or the signal-to-noise ratio (SNR) is small. Figure 2 shows the results
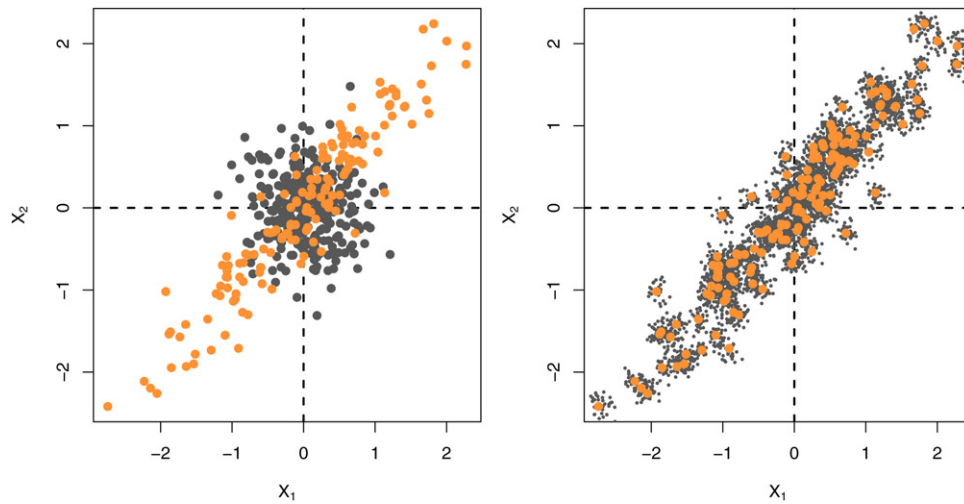
Figure 3. Examples of data augmentation. Left: A mass of fake points at the origin fattens the data cloud, and stabilizes the coefficient estimates in the direction of the smaller principal axis. Right: Many perturbed versions of the original data points are presented, all with the same response. If the perturbations are zero-mean, scalar covariance with the right scalar, the result is approximate ridge.

of a linear-model simulation with $n = 100$ and $p = 54$ and the SNR $= 3.3$ (this is a population R^2 of 77%). The OLS coefficients (left panel, far left in plot) are wild but unbiased. The ridge-shrunken versions at $\log(\lambda) \approx -4$ are much closer to the true coefficients (red marks), and predictions using them achieve minimum mean-squared error from the true linear function $E_X(\hat{f}_\lambda(X) - f(X))^2$ (right panel). Of course we don't know the best λ, and typically use a left-out dataset or cross-validation to estimate the prediction-error curve empirically. Included in the plot is the LOO CV curve, discussed in Section 7, which finds a reasonable value for λ.

A general form of shrinkage for any multivariate estimator is given by the celebrated James–Stein formula (James and Stein 1961). In the context of linear regression we have

$$\hat{\beta}_{JS} = \left[1 - \frac{(p-2)\sigma^2}{\hat{\beta}^\top X^\top X \hat{\beta}} \right] \hat{\beta}, \tag{19}$$

where $\hat{\beta}$ is the OLS estimator, and σ^2 can be estimated in the usual fashion via the residual sum-of-squares for linear regression (Efron and Hastie 2016, chap. 7). The left panel of Figure 2 includes the James–Stein estimates on the right axis. The corresponding MSE is above 15, and so is not that good on this example.

4. Ridge and Data Augmentation

There are some interesting tricks for fitting a ridge regression using standard linear-model software. Suppose we augment X and y in the following way:

$$\widetilde{X} = \begin{bmatrix} X \\ \sqrt{\lambda} I_p \end{bmatrix} \qquad \tilde{y} = \begin{bmatrix} y \\ 0. \end{bmatrix} \tag{20}$$

Think of the case where X and y are centered. We have added p additional points around the origin, each a distance $\sqrt{\lambda}$ along a coordinate axis. It is straightforward to check that the OLS coefficient is $\hat{\beta}_\lambda$. Another way to do this approximately is to augment X in a similar way with n_a random draws X_a from

a $N(0, \lambda I)$ distribution (or any multivariate distribution with mean zero and covariance λI). y is again padded with n_a zeros. If we fit the model by weighted least squares, giving weight 1 to the original points, and $1/n_a$ to the augmentation points, we approximately achieve $\hat{\beta}_\lambda$. Approximate because $\frac{1}{n_a} X_a^\top X_a \approx \lambda I$. See the left plot in Figure 3 for an example. Another way to achieve this is to perturb each observed data vector x_i by a random amount. We make m perturbed copies of each x_i: $x'_{ij} = x_i + \varepsilon_{ij}$, $j = 1, \ldots, m$, where $\varepsilon_{ij} \sim N(0, \frac{\lambda}{n}I)$. Each of the m x'_{ij} vectors gets the same response y_i. Then

$$\sum_{i=1}^{n} \sum_{j=1}^{m} x'_{ij} x'^{\top}_{ij} \approx m \left(X^\top X + \lambda I \right), \tag{21}$$

because the zero-mean cross-terms get averaged away as m gets large.

Here the augmentation is somewhat artificial. A more realistic augmentation is used in image classification with deep neural networks (e.g., Chollet and Allaire 2018). We have a limited number of labeled training images. The idea is to take each image and apply some natural deformations in such a way that humans would have no trouble making the classification. For example, suppose we have an image of a poodle. By making small rotations, scale and location changes, hue and contrast changes, it will still look like a poodle. We can now augment our training set with all these additional versions, creating a cloud of images around the original, as in Figure 3(right panel). This kind of regularization prevents the model from over-training on the original image, and by analogy is a form of ridge regularization.

5. Ridge and Dropout Regularization

Two forms of regularization have arisen in recent years in the statistical and machine learning communities. They are similar in spirit to each other, and rather closely aligned with ridge regression.

Random forests (Breiman 2001) have an option whereby each time a tree is grown, and a split is contemplated at a node, only a

subset m of the p variables available are considered as candidates for splitting. This causes variables to stand in for each other, or share the predictive burden—a characteristic shared by ridge regression. The smaller m, the more regularized the fit.

Modern deep neural networks employ a similar mechanism: *dropout learning* (Srivastava et al. 2014). Neural networks have layers of activations or transformations that feed forward into successive layers through a series of linear models followed by nonlinear transformations. For example, going from layer $k-1$ with p_{k-1} units to layer k with p_k units, consider computing the activation $a_\ell^{(k)}$ for a single observation during the feed-forward stage of training.

$$z_\ell^{(k)} = w_{\ell 0}^{(k-1)} + \sum_{j=1}^{p_{k-1}} w_{\ell j}^{(k-1)} a_j^{(k-1)},$$
$$a_\ell^{(k)} = g(z_\ell^{(k)}). \tag{22}$$

The idea is to randomly set each of the p_{k-1} activations $a_j^{(k-1)}$ to zero with probability ϕ, and inflate the remaining ones by a factor $1/(1-\phi)$. Hence, for this observation, those nodes that survive have to *stand in* for those omitted. This is done independently for each observation, and can also be seen as be a form of ridge regularization, and when done correctly improves performance. The fraction ϕ omitted is a tuning parameter, and for deep networks it appears to be better to use different values at different layers.

We illustrate using a simple version of dropout for linear regression (see also Wager, Wang, and Liang 2013). For simplicity we assume all variables have mean zero, so we can ignore intercepts. Consider the following random least-squares criterion:

$$L_I(\beta) = \frac{1}{2} \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} x_{ij} I_{ij} \beta_j \right)^2. \tag{23}$$

Here the I_{ij} are iid variables for all i, j with

$$I_{ij} = \begin{cases} 0 & \text{with probability } \phi, \\ 1/(1-\phi) & \text{with probability } 1-\phi, \end{cases} \tag{24}$$

(this particular form is used so that $E[I_{ij}] = 1$). Using simple probability it can be shown that the expected score equations can be written

$$E\left[\frac{\partial L_I(\beta)}{\partial \beta} \right] = -X^\top y + X^\top X \beta + \frac{\phi}{1-\phi} D\beta = 0, \tag{25}$$

with $D = \text{diag}\{\|x_1\|^2, \|x_2\|^2, \ldots, \|x_p\|^2\}$, where x_j is the jth column of X. Hence, the solution is given by

$$\hat{\beta} = \left(X^\top X + \frac{\phi}{1-\phi} D \right)^{-1} X^\top y, \tag{26}$$

a generalized ridge regression. If the variables are standardized, the term D becomes a scalar, and the solution is identical to ridge regression.

6. Ridge and the Kernel Trick

We start with the ridge problem (3) for the case $p \gg n$ and write out the score equation (up to a factor 2):

$$-X^\top(y - X\beta) + \lambda\beta = 0. \tag{27}$$

For $\lambda > 0$, it is easy to see that the solution must satisfy $\beta = X^\top \alpha$ for some n vector α. In other words, the solution $\hat{\beta}_\lambda$ lies in the row space of X, an n-dimensional subspace of $\mathbb{R}^p$ (we assume X has full row rank, for ease of exposition).

From this we can easily derive

$$\hat{\alpha}_\lambda = (K + \lambda I_n)^{-1} y, \tag{28}$$

where $K = XX^\top$ is the $n \times n$ *gram* matrix of pairwise inner products. Likewise the fit vector is given by

$$\begin{aligned} \hat{y}_\lambda &= X\hat{\beta}_\lambda \\ &= XX^\top \hat{\alpha}_\lambda \\ &= K(K + \lambda I_n)^{-1} y. \end{aligned} \tag{29}$$

So even though p can be very large, all we need to do are n-dimensional calculations to compute the solution (although the computations needed to produce K here are $O(pn^2)$). This is the *kernel trick* in its simplest form.

This is also the case for GLMs, and in fact any linear model fit obtained by optimizing a quadratically penalized objective. We will use a GLM as an example. Denoting the n-vector of fits by $\eta = X\beta$, we can write the quadratically penalized log-likelihood problem as

$$\underset{\beta}{\text{maximize}}\, \ell(y, X\beta) - \lambda\|\beta\|_2^2. \tag{30}$$

Again it can be shown (Hastie and Tibshirani 2004) that the solution has the form $\beta = X^\top \alpha$, the fit vector $\eta = K\alpha$, and the optimization in α becomes

$$\underset{\beta}{\text{maximize}}\, \ell(y, K\alpha) - \lambda\alpha^\top K\alpha. \tag{31}$$

Here the *linear* kernel matrix corresponds to ridged linear models, but the formulation opens the door to the rich world of function fitting in reproducing kernel Hilbert spaces (RKHS). For any positive definite bivariate function $\mathcal{K}(x, x')$, we can think of it computing inner products in an implicit feature space $h(x)$: $\mathcal{K}(x, x') = \langle h(x), h(x') \rangle$. We end up fitting functions of the form $f(x) = \sum_{i=1}^{n} \mathcal{K}(x, x_i)\alpha_i$, by solving problems of the form (31) where $K_{ii'} = \mathcal{K}(x_i, x_{i'})$ (see Hastie, Tibshirani, and Friedman 2009, chap. 6 for details). The support-vector machine for two-class classification is of this form. With $y_i \in \{-1, +1\}$ and $\eta = K\alpha$, the optimization problem is

$$\underset{}{\text{minimize}} \sum_{i=1}^{n} (1 - y_i\eta_i)_+ + \lambda\alpha^\top K\alpha. \tag{32}$$

For GLMs, we can solve (31) using standard ridge software. We compute the Cholesky decomposition of $K = R^\top R$, and reparameterize via $\theta = R\alpha$ (again an n-vector). Then the objective in θ becomes

$$\underset{\beta'}{\text{maximize}}\, \ell(y, R^\top\theta) - \lambda\|\theta\|_2^2, \tag{33}$$

another ridged linear GLM, but in n rather $p \gg n$ dimensions. For the linear kernel K, it is easy to see that the solution for the original β is $\hat{\beta}_\lambda = Q\hat{\theta}_\lambda$, where $X^\top = QR$ is the QR decomposition of $X^\top$ (same R). The beauty here is that we reduce our $n \times p$ wide matrix X to an $n \times n$ matrix $R^\top$, and then perform our ridge MLE with it instead. We can perform CV to select λ in this space as well, and need only map back if we want to make predictions on new wide vectors x_0.

7. Ridge and Leave-One-Out Cross-Validation

We have already talked of using the SVD to compute the ridge path of solutions efficiently. This eases the burden when computing the solution paths k times during k-fold cross-validation. For n-fold (LOO) CV, we have another beautiful result for ridge and other linear operators.

$$\text{LOO}_\lambda = \sum_{i=1}^{n}(y_i - x_i^\top \hat{\beta}_\lambda^{(-i)})^2 = \sum_{i=1}^{n} \frac{(y_i - x_i^\top \hat{\beta}_\lambda)^2}{(1 - R_{ii}^\lambda)^2}. \quad (34)$$

Here $\hat{\beta}_\lambda^{(-i)}$ is the ridge estimate computed using the $(n-1)$-observation dataset with the pair (x_i, y_i) omitted, and

$$R^\lambda = X(X^\top X + \lambda I)^{-1} X^\top \quad (35)$$

is the $n \times n$ ridge operator matrix for the original n-observation X matrix. The equation says we can compute all the LOO residuals for ridge from the original residuals, each scaled up by $1/(1 - R_{ii}^\lambda)$. From (10), we see we can obtain R^λ efficiently for all λ via

$$R^\lambda = US(\lambda)U^\top, \quad (36)$$

with $S(\lambda)$ the diagonal shrinkage matrix with elements $d_j^2/(d_j^2 + \lambda)$.

One can derive this result using the Sherman–Morrison–Woodbury identity, but a more general and elegant derivation due to Golub, Heath, and Wahba (1979) is as follows. For each pair (x_i, y_i) left out we are required to solve

$$\underset{\beta}{\text{minimize}} \sum_{\ell \neq i}(y_\ell - x_\ell^\top \beta)^2 + \lambda\|\beta\|^2 \quad (37)$$

with solution $\hat{\beta}_\lambda^{(-i)}$. Let $y_i^* = x_i^\top \hat{\beta}_\lambda^{(-i)}$. If we insert the pair (x_i, y_i^*) back into the size $n-1$ dataset, it will not change the solution to (37), since this point is on the solution surface (and hence has zero loss at $\beta = \hat{\beta}_\lambda^{(-i)}$). Back at a full n dataset, and using the linearity of the ridge operator, we have

$$\begin{aligned} y_i^* &= \sum_{\ell \neq i} R_{i\ell}^\lambda y_\ell + R_{ii}^\lambda y_i^* \\ &= \sum_{\ell=1}^{n} R_{i\ell}^\lambda y_i - R_{ii}^\lambda y_i + R_{ii}^\lambda y_i^*, \end{aligned} \quad (38)$$

from which we see that $(y_i - y_i^*) = (y_i - \hat{y}_i)/(1 - R_{ii}^\lambda)$ with $\hat{y}_i = x_i^\top \hat{\beta}_\lambda$.

The LOO formula (34) appears to break down when $p > n$ in the limit as $\lambda \downarrow 0$ toward the minimum-norm solution. It turns out we can get an equally elegant solution in this case as well (Hastie et al. 2019).

For $\lambda > 0$ from (29) we get

$$\begin{aligned} r_\lambda &= (I_n - K(K + \lambda I_n)^{-1})y \\ &= \lambda(K + \lambda I_n)^{-1}y. \end{aligned} \quad (39)$$

Hence, the summands in formula (34) can be written as

$$y_i - x_i^\top \hat{\beta}_\lambda^{(-i)} = \frac{\{(K + \lambda I_n)^{-1}y\}_i}{\{(K + \lambda I_n)^{-1}\}_{ii}} \quad (40)$$

(the λ multiplier from (39) in the numerator and denominator cancel). Now we can set $\lambda = 0$ to obtain

$$\text{LOO}_{mn} = \sum_{i=1}^{n}\left(y_i - \hat{x}_i^\top \beta_{mn}^{(-i)}\right)^2 = \sum_{i=1}^{n}\left(\frac{\{K^{-1}y\}_i}{\{K^{-1}\}_{ii}}\right)^2. \quad (41)$$

This formula does not apply if there is an unpenalized intercept in the model. One can show in this case that the corresponding formula is

$$\text{LOO}_{mn} = \sum_{i=1}^{n} \frac{\{\widetilde{K}^+ y\}_i}{\{\widetilde{K}^+\}_{ii}}, \quad (42)$$

where $\widetilde{K}$ is the doubly-centered kernel matrix $\widetilde{K} = (I_n - M)K(I_n - M)$ (with rank at most $n-1$), $M = 1_n 1_n^\top/n$ is the mean projection operator, and the $\widetilde{K}^+$ is a pseudo inverse.

8. Ridge, Minimum Norm, and Double Descent

There has been a burst of activity in the machine learning community over some surprising behavior of a particular class of estimators. Before we get into detail, a bit of context is in order. Deep learning models are dominant in certain high SNR domains, such as image classification. Practitioners have found that fitting a deep convolutional network with many layers by gradient descent all the way down to zero training error performs very well on test data. These networks typically have more parameters than training observations and so this would normally be classed as severe *overfitting*. What has also been observed is that *increasing* the number of parameters (via more hidden units) can improve performance in some instances (Zhang et al. 2016; Belkin et al. 2018; Hastie et al. 2019).

We first make the simple observation (Zhang et al. 2016) that with $p \gg n$ and squared-error loss, gradient descent starting at $\beta = 0$ down to zero training error gives the minimum ℓ_2-norm solution. This is because the gradient (11) is in the row space of X, and hence all the gradient updates remain there. This characterizes the minimum-norm solution (see end of Section 2).

Figure 4 is a simple demonstration of this *double descent* behavior. We generated data from a model $y = f(x) + \varepsilon$ with $x \in \mathbb{R}^9$. The true function f is nonlinear and nonadditive, x is distributed $N(0, I_9)$, ε is $N(0, \sigma^2)$, and σ is chosen so the SNR $\text{var}(f(x))/\sigma^2 = 3$. The sample size is $n = 100$, and we use a very large test set (10K) to measure out-of-sample error. For our estimation method we use an additive model

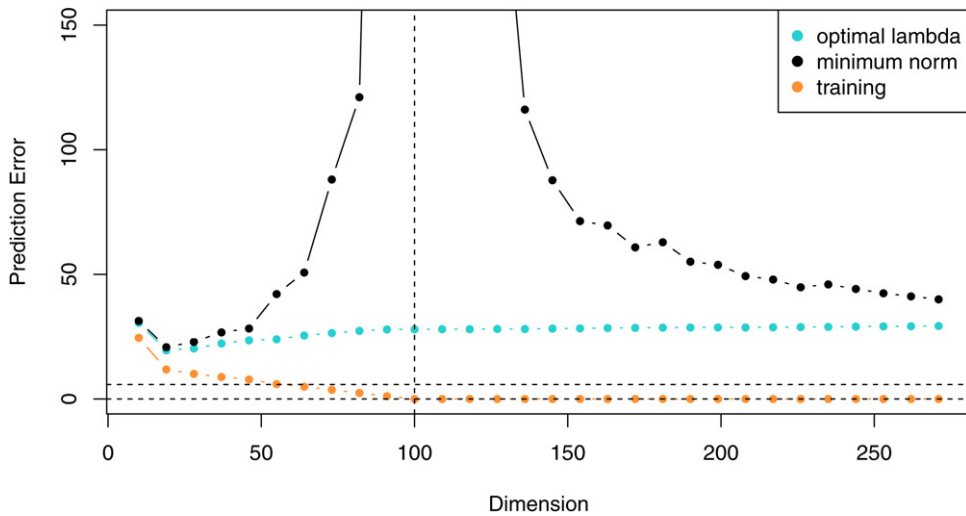$$g(x; d) = \theta_0 + \sum_{j=1}^{9} \theta_j^\top h_j(x_j; d), \quad (43)$$

Figure 4. Double descent in the generalization error of minimum-norm estimation as the dimension increases.
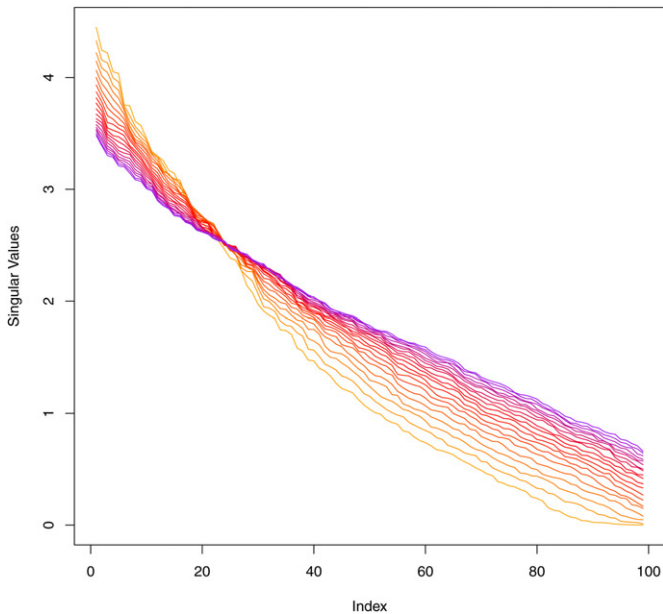

Figure 5. Singular values for the natural spline bases on the training data, as the dimension increases from 100 (orange) to 270 (purple).

where each $h_j(\cdot; d)$ is a d-vector of natural-spline basis functions, with knots chosen at uniform quantiles of the training values for variable x_j (Hastie, Tibshirani, and Friedman 2009, chap. 5). The total number of parameters or dimension of the model is $9d + 1$, with d stepping from 1 thru 30. The dimension reaches the training sample size of $n = 100$ when $d = 11$.

The black curve shows the test prediction error of the minimum-norm least squares fits, and the orange curve their training error. The training error behaves as expected, reaching and staying at zero as the dimension passes 100. Before 100, the fits are OLS and overfit almost immediately, and increase to a dramatic level (around 2000) before descending down again as the dimension increases. The blue curve corresponds to the optimally tuned ridge estimator which is fairly flat, with a minimum around dimension 20.

The apparent dilemma here is that the black curve does not show the usual bias-variance trade-off as the dimension increases. The explanation for the increase around 100 is that the model has to interpolate the training data to achieve zero training error. As a result the prediction curve has to wiggle a lot between training points, and the ℓ_2 norm of $\hat{\theta}$ gets large. This does not bode well for out-of-sample prediction, since the test features fall in this in-between region. But as the dimension grows beyond 100, zero error can be achieved more smoothly with decreasing norm, and leads to improved out-of-sample prediction. So the complexity of the model is not determined by dimension alone; the potential for smaller ℓ_2 norm of the solution at each dimension plays a role as well. The "objective" that is optimized at each dimension combines loss and complexity (OLS fit with minimum norm), which clouds the usual bias-variance picture.

It is interesting to note that the minimum-norm fitting method is linear in y (see (12)). However, as the number of columns (dimension) of the basis matrix H increases, the small singular values increase leading to the potential for more stable and smaller ℓ_2-norm solutions. Figure 5 illustrates this for our example.

9. Ridge + Ridge = Lasso

Suppose X is an $m \times n$ matrix. Consider the optimization problem

$$\underset{M}{\text{minimize}} \|X - M\|_F^2 \quad \text{s.t. } \text{rank}(M) \leq q. \tag{44}$$

Even though the rank constraint makes this problem nonconvex, the solution is well known and easy to compute. We compute the SVD of $X = UDV^\top$, set all but the top q singular values of D to zero, and reconstruct. There is a convex relaxation of this problem (Fazel 2002)

$$\underset{M}{\text{minimize}} \|X - M\|_F^2 + \lambda \|M\|_*, \tag{45}$$

where $\|M\|_*$ is the *nuclear norm* of M—the sum of the singular values. The solution is again via the SVD of X. Now we replace

the singular values by their soft-thresholded values $\tilde{d}_i = (d_i - \lambda)_+$, and reconstruct. This is the lasso version of rank selection for matrices. Suppose λ is such that all but q of the $\tilde{d}_i$ are greater than zero, and hence the solution $\widetilde{M}_\lambda$ has rank q. Now consider the doubly ridged problem (Srebro, Rennie, and Jaakkola 2005)

$$\underset{A,B}{\text{minimize}} \, \|X - AB^\top\|_F^2 + \frac{\lambda}{2}\|A\|_F^2 + \frac{\lambda}{2}\|B\|_F^2, \quad (46)$$

over matrices $A_{m \times q}$ and $B_{n \times q}$. This biconvex problem has a solution that coincides with that of (45): $\widetilde{A}\widetilde{B}^\top = \widetilde{M}$. Quite remarkable that a biconvex ℓ_2-regularized problem is equivalent to a convex ℓ_1 regularized problem! At this point these connections may seem somewhat academic, since the SVD of X provides solutions to all the problems above. These connections show their strength in variations of these problems.

- If X is massive and sparse, we can compute a low-rank matrix approximation by alternating ridge regressions. Given A, we obtain B via $B^\top = (A^\top A + \frac{\lambda}{2}I_q)A^\top X$, which is a dense skinny matrix multiplying a sparse matrix. Likewise for A given B.
- When X has missing values, we can solve the *matrix completion* objective

$$\underset{M}{\text{minimize}} \, \|P_\Omega(X - M)\|_F^2 + \lambda\|M\|_*, \quad (47)$$

where P_Ω projects onto the observed values of X (i.e., the Frobenius norm ignores the entries corresponding to missing values in X). Objective (47) is convex in M, and for large X solutions can be obtained again using the alternating ridged version (46) (Hastie et al. 2015).

10. Discussion

We see through these examples that ridge regularization and its extensions are pervasive in applied statistics. Although the lasso enjoys widespread popularity in wide-data scenarios, both ridge and elastic-net claim some of the territory. In document and web-page classification using bag-of-words and n-grams, synonyms create problems for methods that select variables. Ridge includes them all, and suffers less. Furthermore, classifiers are hurt less by bias than quantitative regressors. What started off as a simple fix for wayward linear regression models has evolved into a large collection of tools for data modeling. It would be hard to imagine the life of a data scientist without them.

Acknowledgments

Funding

References

Belkin, M., Hsu, D., Ma, S., and Mandal, S. (2018), "Reconciling Modern Machine Learning and the Bias-Variance Trade-Off," arXiv no. 1812.11118. [431]

Breiman, L. (2001), "Random Forests," *Machine Learning*, 45, 5–32. [429]

Chollet, F., and Allaire, J. J. (2018), *Deep Learning With R* (1st ed.), Shelter Island, NY: Manning Publications Co. [429]

Chouldechova, A., and Hastie, T. (2015), "Generalized Additive Model Selection," arXiv no. 1506.03850. [427]

Efron, B., and Hastie, T. (2016), *Computer Age Statistical Inference; Algorithms, Evidence and Data Science*, New York: Cambridge University Press. [429]

Fazel, M. (2002), "Matrix Rank Minimization With Applications," PhD thesis, Department of Electrical Engineering, Stanford University. [432]

Golub, G., Heath, M., and Wahba, G. (1979), "Generalized Cross-Validation as a Method for Choosing a Good Ridge Parameter," *Technometrics*, 21, 215–224. [431]

Hastie, T., Mazumder, R., Lee, J., and Zadeh, R. (2015), "Matrix Completion and Low-Rank SVD via Fast Alternating Least Squares," *Journal of Machine Learning Research*, 16, 3367–3402. [433]

Hastie, T., Montanari, A., Rosset, S., and Tibshirani, R. J. (2019), "Surprises in High-Dimensional Ridgeless Least Squares Interpolation," arXiv no. 1903.08560. [431]

Hastie, T., and Tibshirani, R. (2004), "Efficient Quadratic Regularization for Expression Arrays," *Biostatistics*, 5, 329–340. [430]

Hastie, T., Tibshirani, R., and Friedman, J. (2009), *The Elements of Statistical Learning: Prediction, Inference and Data Mining* (2nd ed.), New York: Springer-Verlag. [430,432]

Hastie, T., Tibshirani, R., and Wainwright, M. (2015), *Statistical Learning With Sparsity: The Lasso and Generalizations*, Boca Raton, FL: Chapman and Hall/CRC Press. [427]

Hoerl, A. E., and Kennard, R. W. (1970), "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *Technometrics*, 12, 55–67. [426]

Jacob, L., Obozinski, G., and Vert, J.-P. (2009), "Group Lasso With Overlap and Graph Lasso," in *Proceeding of the 26th International Conference on Machine Learning*, Montreal, Canada. [427]

James, W., and Stein, C. (1961), "Estimation With Quadratic Loss," in *Proc. 4th Berkeley Symposium on Mathematical Statistics and Probability* (Vol. I), Berkeley, CA: University of California Press, pp. 361–379. [429]

Lim, M., and Hastie, T. (2015), "Learning Interactions via Hierarchical Group-Lasso Regularization," *Journal of Computational and Graphical Statistics*, 24, 627–654. [427]

Srebro, N., Rennie, J., and Jaakkola, T. (2005), "Maximum-Margin Matrix Factorization," in *Advances in Neural Information Processing Systems* (Vol. 17), Cambridge, MA: MIT Press, pp. 1329–1336. [433]

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014), "Dropout: A Simple Way to Prevent Neural Networks From Overfitting," *Journal of Machine Learning Research*, 15, 1929–1958. [430]

Tibshirani, R. (1996), "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society*, Series B, 58, 267–288. [426]

Wager, S., Wang, S., and Liang, P. S. (2013), "Dropout Training as Adaptive Regularization," in *Advances in Neural Information Processing Systems* (Vol. 26), eds. C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Red Hook, NY: Curran Associates, Inc., pp. 351–359, available at *http://papers.nips.cc/paper/4882-dropout-training-as-adaptive-regularization.pdf*. [430]

Yuan, M., and Lin, Y. (2007), "Model Selection and Estimation in Regression With Grouped Variables," *Journal of the Royal Statistical Society*, Series B, 68, 49–67. [427]

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2016), "Understanding Deep Learning Requires Rethinking Generalization," arXiv no. 1611.03530. [431]

Zou, H., and Hastie, T. (2005), "Regression Shrinkage and Selection via the Elastic Net," *Journal of the Royal Statistical Society*, Series B, 67, 301–320. [427]