Part III — Modern Statistical Methods

Based on lectures by R. D. Shah

Notes taken by Dexter Chua

Michaelmas 2017

These notes are not endorsed by the lecturers, and I have modified them (often significantly) after lectures. They are nowhere near accurate representations of what was actually lectured, and in particular, all errors are almost surely mine.

The remarkable development of computing power and other technology now allows scientists and businesses to routinely collect datasets of immense size and complexity. Most classical statistical methods were designed for situations with many observations and a few, carefully chosen variables. However, we now often gather data where we have huge numbers of variables, in an attempt to capture as much information as we can about anything which might conceivably have an influence on the phenomenon of interest. This dramatic increase in the number variables makes modern datasets strikingly different, as well-established traditional methods perform either very poorly, or often do not work at all.

Developing methods that are able to extract meaningful information from these large and challenging datasets has recently been an area of intense research in statistics, machine learning and computer science. In this course, we will study some of the methods that have been developed to analyse such datasets. We aim to cover some of the following topics.

- Kernel machines: the kernel trick, the representer theorem, support vector machines, the hashing trick.
- Penalised regression: Ridge regression, the Lasso and variants.
- Graphical modelling: neighbourhood selection and the graphical Lasso. Causal inference through structural equation modelling; the PC algorithm.
- High-dimensional inference: the closed testing procedure and the Benjamini–Hochberg procedure; the debiased Lasso

Pre-requisites

Basic knowledge of statistics, probability, linear algebra and real analysis. Some background in optimisation would be helpful but is not essential.

Contents

0 Introduction

In recent years, there has been a rather significant change in what sorts of data we have to handle and what questions we ask about them, witnessed by the popularity of the buzzwords "big data" and "machine learning". In classical statistics, we usually have a small set of parameters, and a very large data set. We then use the large data set to estimate the parameters.

However, nowadays we often see scenarios where we have a very large number of parameters, and the data set is relatively small. If we tried to apply our classical linear regression, then we would just be able to tune the parameters so that we have a perfect fit, and still have great freedom to change the parameters without affecting the fitting.

One example is that we might want to test which genomes are responsible for a particular disease. In this case, there is a huge number of genomes to consider, and there is good reason to believe that most of them are irrelevant, i.e. the parameters should be set to zero. Thus, we want to develop methods that find the "best" fitting model that takes this into account.

Another problem we might encounter is that we just have a large data set, and doing linear regression seems a bit silly. If we have so much data, we might as well try to fit more complicated curves, such as polynomial functions and friends. Perhaps more ambitiously, we might try to find the best *continuously differentiable function* that fits the curve, or, as analysts will immediately suggest as an alternative, weakly differentiable functions.

There are many things we can talk about, and we can't talk about all of them. In this course, we are going to cover 4 different topics of different size:

- Kernel machines
- The Lasso and its extensions
- Graphical modeling and causal inference
- Multiple testing and high-dimensional inference

The four topics are rather disjoint, and draw on different mathematical skills.

1 Classical statistics

This is a course on *modern* statistical methods. Before we study methods, we give a brief summary of what we are *not* going to talk about, namely classical statistics.

So suppose we are doing regression. We have some *predictors* $x_i \in \mathbb{R}^p$ and *responses* $Y_i \in \mathbb{R}$, and we hope to find a model that describes Y as a function of x. For convenience, define the vectors

$$X = \begin{pmatrix} x_1^T \\ \vdots \\ x_n^T \end{pmatrix}, \quad Y = \begin{pmatrix} Y_1^T \\ \vdots \\ Y_n^T \end{pmatrix}.$$

The linear model then assumes there is some $\beta^0 \in \mathbb{R}^p$ such that

$$Y = X\beta^0 + \varepsilon,$$

where ε is some (hopefully small) error random variable. Our goal is then to estimate β^0 given the data we have.

If X has full column rank, so that $X^T X$ is invertible, then we can use *ordinary least squares* to estimate β^0, with estimate

$$\hat{\beta}^{OLS} = \operatorname*{argmin}_{\beta \in \mathbb{R}^p} \|Y - x\beta\|_2^2 = (X^T X)^{-1} X^T Y.$$

This assumes nothing about ε itself, but if we assume that $\mathbb{E}\varepsilon = 0$ and $\operatorname{var}(\mathcal{E}) = \sigma^2 I$, then this estimate satisfies

- $\mathbb{E}_\beta \hat{\beta}^{OLS} = (X^T X)^{-1} X^T X \beta^0 = \beta^0$
- $\operatorname{var}_\beta(\hat{\beta}^{OLS}) = (X^T X^{-1}) X^T \operatorname{var}(\varepsilon) X (X^T X)^{-1} = \sigma^2 (X^T X)^{-1}$.

In particular, this is an unbiased estimator. Even better, this is the best linear unbiased estimator. More precisely, the Gauss–Markov theorem says any other linear estimator $\tilde{\beta} = AY$ has $\operatorname{var}(\tilde{\beta}) - \operatorname{var}(\hat{\beta}^{OLS})$ positive semi-definite.

Of course, ordinary least squares is not the only way to estimate β^0. Another common method for estimating parameters is maximum likelihood estimation, and this works for more general models than linear regression. For people who are already sick of meeting likelihoods, this will be the last time we meet likelihoods in this course.

Suppose we want to estimate a parameter θ via knowledge of some data Y. We assume Y has density $f(y; \theta)$. We define the *log-likelihood* by

$$\ell(\theta) = \log f(Y, \theta).$$

The *maximum likelihood estimator* then maximize $\ell(\theta)$ over θ to get $\hat{\theta}$.

Similar to ordinary least squares, there is a theorem that says maximum likelihood estimation is the "best". To do so, we introduce the *Fisher information matrix*. This is a family of $d \times d$ matrices indexed by θ, defined by

$$I_{jk}(\theta) = -E_\theta \left[\frac{\partial^2}{\partial \theta_j \partial \theta_j} \ell(\theta) \right].$$

The relevant theorem is

Theorem (Cramér–Rao bound)**.** If $\tilde{\theta}$ is an unbiased estimator for θ, then $\mathrm{var}(\tilde{\theta}) - I^{-1}(\theta)$ is positive semi-definite.

Moreover, asymptotically, as $n \to \infty$, the maximum likelihood estimator is unbiased and achieves the Carmér–Rao bound.

Another wonderful fact about the maximum likelihood estimator is that asymptotically, it is normal distributed, and so it is something we understand well.

This might seem very wonderful, but there are a few problems here. The results we stated are asymptotic, but what we actually see in real life is that as $n \to \infty$, the value of p also increases. In these contexts, the asymptotic property doesn't tell us much. Another issue is that these all talk about unbiased estimators. In a lot of situations of interest, it turns out biased methods do much much better than these methods we have.

Another thing we might be interested is that as n gets large, we might want to use more complicated models than simple parametric models, as we have much more data to mess with. This is not something ordinary least squares provides us with.

2 Kernel machines

We are going to start a little bit slowly, and think about our linear model $Y = X\beta^0 + \varepsilon$, where $\mathbb{E}(\varepsilon) = 0$ and $\mathrm{var}(\varepsilon) = \sigma^2 I$. Ordinary least squares is an unbiased estimator, so let's look at biased estimators.

For a biased estimator, $\tilde{\beta}$, we should not study the variance, but the *mean squared error*

$$\begin{aligned}\mathbb{E}[(\tilde{\beta} - \beta^0)(\tilde{\beta} - \beta^0)^T] &= \mathbb{E}(\tilde{\beta} - \mathbb{E}\tilde{\beta} + E\tilde{\beta} - \beta^0)(\tilde{\beta} - \mathbb{E}\tilde{\beta} + E\tilde{\beta} - \beta^0)^T\\ &= \mathrm{var}(\tilde{\beta}) + (\mathbb{E}\tilde{\beta} - \beta^0)(\mathbb{E}\tilde{\beta} - \beta^0)^T\end{aligned}$$

The first term is, of course, just the variance, and the second is the squared bias. So the point is that if we pick a clever biased estimator with a tiny variance, then this might do better than unbiased estimators with large variance.

2.1 Ridge regression

Ridge regression was introduced in around 1970. The idea of Ridge regression is to try to shrink our estimator a bit in order to lessen the variance, at the cost of introducing a bias. We would hope then that this will result in a smaller mean squared error.

Definition (Ridge regression)**.** *Ridge regression* solves

$$(\hat{\mu}_\lambda^R, \hat{\beta}_\lambda^R) = \operatorname*{argmin}_{(\mu,\beta)\in\mathbb{R}\times\mathbb{R}^p} \{\|Y - \mu\mathbf{1} - X\beta\|_2^2 + \lambda\|\beta\|_2^2\},$$

where $\mathbf{1}$ is a vector of all 1's. Here $\lambda \geq 0$ is a tuning parameter, and it controls how much we penalize a large choice of β.

Here we explicitly have an intercept term. Usually, we eliminate this by adding a column of 1's in X. But here we want to separate that out, since we do not want give a penalty for large μ. For example, if the parameter is temperature, and we decide to measure in degrees Celsius rather than Kelvins, then we don't want the resulting $\hat{\mu}, \hat{\beta}$ to change.

More precisely, our formulation of Ridge regression is so that if we make the modification

$$Y' = c\mathbf{1} + Y,$$

then we have

$$\hat{\mu}_\lambda^R(Y') = \hat{\mu}_\lambda^R(Y) + c.$$

Note also that the Ridge regression formula makes sense only if each entry in β have the same order of magnitude, or else the penalty will only have a significant effect on the terms of large magnitude. Standard practice is to subtract from each column of X its mean, and then scale it to have ℓ_2 norm $\sqrt{n}$. The actual number is not important here, but it will in the case of the Lasso.

By differentiating, one sees that the solution to the optimization problem is

$$\begin{aligned}\hat{\mu}_\lambda^R &= \bar{Y} = \frac{1}{n}\sum Y_i\\ \hat{\beta}_\lambda^R &= (X^T X + \lambda I)^{-1} X^T Y.\end{aligned}$$

Note that we can always pick λ such that the matrix $(X^T X + \lambda I)$ is invertible. In particular, this can work even if we have more parameters than data points. This will be important when we work with the Lasso later on.

If some god-like being told us a suitable value of λ to use, then Ridge regression always works better than ordinary least squares.

Theorem. Suppose $\operatorname{rank}(X) = p$. Then for $\lambda > 0$ sufficiently small (depending on β^0 and σ^2), the matrix

$$\mathbb{E}(\hat{\beta}^{OLS} - \beta^0)(\hat{\beta}^{OLS} - \beta^0)^T - \mathbb{E}(\hat{\beta}^R_\lambda - \beta^0)(\hat{\beta}^R_\lambda - \beta^0)^T \tag{$*$}$$

is positive definite.

Proof. We know that the first term is just $\sigma^2(X^T X)^{-1}$. The right-hand-side has a variance term and a bias term. We first look at the bias:

$$\begin{aligned}\mathbb{E}[\hat{\beta} - \beta^0] &= (X^T X + \lambda I)^{-1} X^T X \beta^0 - \beta^0\\ &= (X^T X + \lambda I)^{-1}(X^T X + \lambda I - \lambda I)\beta^0 - \beta^0\\ &= -\lambda(X^T X + \lambda I)^{-1}\beta^0.\end{aligned}$$

We can also compute the variance

$$\operatorname{var}(\hat{\beta}) = \sigma^2(X^T X + \lambda I)^{-1} X^T X (X^T X + \lambda I)^{-1}.$$

Note that both terms appearing in the squared error look like

$$(X^T X + \lambda I)^{-1}\text{something}(X^T X + \lambda I)^{-1}.$$

So let's try to write $\sigma^2(X^T X)^{-1}$ in this form. Note that

$$\begin{aligned}(X^T X)^{-1} &= (X^T X + \lambda I)^{-1}(X^T X + \lambda I)(X^T X)^{-1}(X^T X + \lambda I)(X^T X + \lambda I)^{-1}\\ &= (X^T X + \lambda I)^{-1}(X^T X + 2\lambda I + \lambda^2 (X^T X)^{-1})(X^T X + \lambda I)^{-1}.\end{aligned}$$

Thus, we can write $(*)$ as

$$\begin{aligned}&(X^T X + \lambda I)^{-1}\Big(\sigma^2(X^T X + 2\lambda I + \lambda^2 (X^T X)^{-1})\\ &\qquad\qquad - \sigma^2 X^T X - \lambda^2 \beta^0 (\beta^0)^T\Big)(X^T X + \lambda I)^{-1}\\ &= \lambda (X^T X + \lambda I)^{-1}\Big(2\sigma^2 I + \lambda (X^T X)^{-1} - \lambda \beta^0(\beta^0)^T\Big)(X^T X + \lambda I)^{-1}\end{aligned}$$

Since $\lambda > 0$, this is positive definite iff

$$2\sigma^2 I + \sigma^2 \lambda (X^T X)^{-1} - \lambda \beta^0 (\beta^0)^T$$

is positive definite, which is true for $0 < \lambda < \frac{2\sigma^2}{\|\beta^0\|_2^2}$. □

While this is nice, this is not really telling us much, because we don't know how to pick the correct λ. It also doesn't tell us when we should expect a big improvement from Ridge regression.

To understand this better, we need to use the *singular value decomposition.*

Theorem (Singular value decomposition)**.** Let $X \in \mathbb{R}^{n \times p}$ be any matrix. Then it has a *singular value decomposition* (SVD)

$$\underset{n\times p}{X} = \underset{n\times n}{U} \underset{n\times p}{D} \underset{p\times p}{V^T},$$

where U, V are orthogonal matrices, and $D_{11} \geq D_{22} \geq \cdots \geq D_{mm} \geq 0$, where $m = \min(n, p)$, and all other entries of D are zero.

When $n > p$, there is an alternative version where U is an $n \times p$ matrix with orthogonal columns, and D is a $p \times p$ diagonal matrix. This is done by replacing U by its first p columns and D by its first p rows. This is known as the *thin singular value decomposition*. In this case, $U^T U = I_p$ but UU^T is not the identity.

Let's now apply try to understand Ridge regressions a little better. Suppose $n > p$. Then using the thin SVD, the fitted values from ridge regression are

$$\begin{aligned} X\hat{\beta}^R_\lambda &= X(X^TX + \lambda I)^{-1}X^TY \\ &= UDV^T(VD^2V^T + \lambda I)^{-1}VDU^TY. \end{aligned}$$

We now note that $VD^2V^T + \lambda I = V(D^2 + \lambda I)V^T$, since V is still orthogonal. We then have

$$(V(D^2 + \lambda I)V^T)^{-1} = V(D^2 + \lambda I)^{-1}V^T$$

Since D^2 and λI are both diagonal, it is easy to compute their inverses as well. We then have

$$X\hat{\beta}^R_\lambda = UD^2(D^2 + \lambda I)^{-1}U^TY = \sum_{j=1}^p U_j \frac{D_{jj}^2}{D_{jj}^2 + \lambda} U_j^T Y$$

Here U_j refers to the jth column of U.

Now note that the columns of U form an orthonormal basis for the column space of X. If $\lambda = 0$, then this is just a fancy formula for the projection of Y onto the column space of X. Thus, what this formula is telling us is that we look at this projection, look at the coordinates in terms of the basis given by the columns of U, and scale accordingly.

We can now concretely see the effect of our λ. The shrinking depends on the size of D_{jj}, and the larger D_{jj} is, the less shrinking we do.

This is not very helpful if we don't have a concrete interpretation of the D_{jj}, or rather, the columns of U. It turns out the columns of U are what are known as the *normalized principal components* of X.

Take $u \in \mathbb{R}^p$, $\|u\|_2 = 1$. The sample variance of $Xu \in \mathbb{R}^n$ is then

$$\frac{1}{n}\|Xu\|_2^2 = \frac{1}{n}u^TX^TXu = \frac{1}{n}u^TVD^2V^tu.$$

We write $w = V^Tu$. Then $\|w\|_2 = 1$ since V is orthogonal. So we have

$$\frac{1}{n}\|Xu\|_2^2 = \frac{1}{n}w^TD^2w = \frac{1}{n}\sum_j w_j^2 D_{jj^2} \leq \frac{1}{n}D_{11}^2,$$

and the bound is achieved when $w = e_1$, or equivalently, $u = Ve_1 = V_1$. Thus, V_1 gives the coefficients of the linear combination of columns of X that has largest sample variance. We can then write the result as

$$XV_1 = UDV^TV_1 = U_1D_{11}.$$

We can extend this to a description of the other columns of U, which is done in the example sheet. Roughly, the subsequent principle components obey the same optimality conditions with the added constraint of being orthogonal to all earlier principle components.

The conclusion is that Ridge regression works best if $\mathbb{E}Y$ lies in the space spanned by the large principal components of X.

2.2 v-fold cross-validation

In practice, the above analysis is not very useful, since it doesn't actually tell us what λ we should pick. If we are given a concrete data set, how do we know what λ we should pick?

One common method is to use *v-fold cross-validation*. This is a very general technique that allows us to pick a regression method from a variety of options. We shall explain the method in terms of Ridge regression, where our regression methods are parametrized by a single parameter λ, but it should be clear that this is massively more general.

Suppose we have some data set $(Y_i, x_i)_{i=1}^n$, and we are asked to predict the value of Y^* given a new predictors x^*. We may want to pick λ to minimize the following quantity:

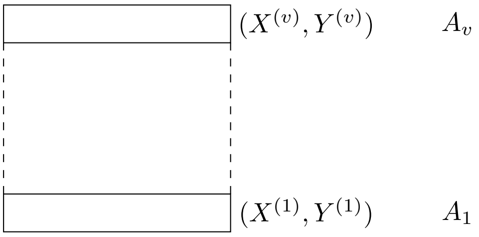
$$\mathbb{E}\left\{(Y^* - (x^*)^T \hat{\beta}_\lambda^R(X, Y))^2 \mid X, Y\right\}.$$

It is difficult to actually do this, and so an easier target to minimize is the expected prediction error

$$\mathbb{E}\left[\mathbb{E}\left\{(Y^* - (x^*)^T \hat{\beta}_\lambda^R(\tilde{X}, \tilde{Y}))^2 \mid \tilde{X}, \tilde{Y}\right\}\right]$$

One thing to note about this is that we are thinking of $\tilde{X}$ and $\tilde{Y}$ as arbitrary data sets of size n, as opposed to the one we have actually got. This might be a more tractable problem, since we are not working with our actual data set, but general data sets.

The method of cross-validation estimates this by splitting the data into v folds.



A_i is called the *observation indices*, which is the set of all indices j so that the jth data point lies in the ith fold.

We let $(X^{(-k)}, Y^{(-k)})$ be all data except that in the kth fold. We define

$$CV(\lambda) = \frac{1}{n} \sum_{k=1}^v \sum_{i \in A_k} (Y_i - x_i^T \hat{\beta}_\lambda^R(X^{(-k)}, Y^{(-k)}))^2$$

We write λ_{CV} for the minimizer of this, and pick $\hat{\beta}^R_{\lambda_{CV}}(X, Y)$ as our estimate. This tends to work very will in practice.

But we can ask ourselves — why do we have to pick a single λ to make the estimate? We can instead average over a range of different λ. Suppose we have computed $\hat{\beta}^R_\lambda$ on a grid of λ-values $\lambda_1 > \lambda_2 > \cdots > \lambda_L$. Our plan is to take a good weighted average of the λ. Concretely, we want to minimize

$$\frac{1}{n}\sum_{k=1}^{v}\sum_{i \in A_k}\left(Y_i - \sum_{i=1}^{L} w_i x_i^T \hat{\beta}^R_{\lambda_i}(X^{(-k)}, Y^{(-k)})\right)^2$$

over $w \in [0, \infty)^L$. This is known as *stacking*. This tends to work better than just doing v-fold cross-validation. Indeed, it must — cross-validation is just a special case where we restrict w_i to be zero in all but one entries. Of course, this method comes with some heavy computational costs.

2.3 The kernel trick

We now come to the actual, main topic of the chapter. We start with a very simple observation. An alternative way of writing the fitted values from Ridge regression is

$$(X^T X + \lambda I)X^T = X^T(XX^T + \lambda I).$$

One should be careful that the λI are different matrices on both sides, as they have different dimensions. Multiplying inverses on both sides, we have

$$X^T(XX^T + \lambda I)^{-1} = (X^T X + \lambda I)^{-1}X^T.$$

We can multiply the right-hand-side by Y to obtain the $\hat{\beta}$ from Ridge regression, and multiply on the left to obtain the fitted values. So we have

$$XX^T(XX^T + \lambda I)^{-1}Y = X(X^T X + \lambda I)^{-1}X^T Y = X\hat{\beta}^R_\lambda.$$

Note that $X^T X$ is a $p \times p$ matrix, and takes $O(np^2)$ time to compute. On the other hand, XX^T is an $n \times n$ matrix, and takes $O(n^2 p)$ time to compute. If $p \gg n$, then this way of computing the fitted values would be much quicker. The key point to make is that the fitted values from Ridge regression only depends on $K = XX^T$ (and Y). Why is this important?

Suppose we believe we have a quadratic signal

$$Y_i = x_i^T \beta + \sum_{k,\ell} x_{ik} x_{i\ell} \theta_{k\ell} + \varepsilon_i.$$

Of course, we can do Ridge regression, as long as we add the products $x_{ik}x_{i\ell}$ as predictors. But this requires $O(p^2)$ many predictors. Even if we use the XX^T way, this has a cost of $O(n^2p^2)$, and if we used the naive method, it would require $O(np^4)$ operations.

We can do better than this. The idea is that we might be able to compute K directly. Consider

$$(1 + x_i^T x_j)^2 = 1 + 2x_i^T x_j + \sum_{k,\ell} x_{ik} x_{i\ell} x_{jk} x_{j\ell}.$$

This is equal to the inner product between vectors of the form

$$(1, \sqrt{2}x_{i1}, \ldots, \sqrt{2}x_{ip}, x_{i1}x_{i1}, \ldots, x_{i1}x_{ip}, x_{i2}x_{i1}, \ldots, x_{ip}x_{ip}) \qquad (*)$$

If we set $K_{ij} = (1+x_i^T x_j)^2$ and form $K(K+\lambda I)^{-1}Y$, this is equivalent to forming ridge regression with $(*)$ as our predictors. Note that here we don't scale our columns to have the same ℓ_2 norm. This is pretty interesting, because computing this is only $O(n^2 p)$. We managed to kill a factor of p in this computation. The key idea here was that the fitted values depend only on K, and not on the values of x_{ij} itself.

Consider the general scenario where we try to predict the value of Y given a predictor $x \in \mathcal{X}$. In general, we don't even assume $\mathcal{X}$ has some nice linear structure where we can do linear regression.

If we want to do Ridge regression, then one thing we can do is that we can try to construct some map $\phi : \mathcal{X} \to \mathbb{R}^D$ for some D, and then run Ridge regression using $\{\phi(x_i)\}$ as our predictors. If we want to fit some complicated, non-linear model, then D can potentially be very huge, and this can be costly. The observation above is that instead of trying to do this, we can perhaps find some magical way of directly computing

$$K_{ij} = k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle.$$

If we can do so, then we can simply use the above formula to obtain the fitted values (we shall discuss the problem of making new predictions later).

Since it is only the function k that matters, perhaps we can specify a "regression method" simply by providing a suitable function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. If we are given such a function k, when would it arise from a "feature map" $\phi : \mathcal{X} \to \mathbb{R}^D$?

More generally, we will find that it is convenient to allow for ϕ to take values in infinite-dimensional vector spaces instead (since we don't actually have to compute ϕ!).

Definition (Inner product space)**.** an inner product space is a real vector space $\mathcal{H}$ endowed with a map $\langle \cdot, \cdot \rangle : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ and obeys

- Symmetry: $\langle u, v \rangle = \langle v, u \rangle$
- Linearity: If $a, b \in \mathbb{R}$, then $\langle au + bw, v \rangle = a\langle u, v \rangle + b\langle w, v \rangle$.
- Positive definiteness: $\langle u, u \rangle \geq 0$ with $\langle u, u \rangle = 0$ iff $u = 0$.

If we want k to come from a feature map ϕ, then an immediate necessary condition is that k has to be symmetric. There is also another condition that corresponds to the positive-definiteness of the inner product.

Proposition. Given $\phi : \mathcal{X} \times \mathcal{X} \to \mathcal{H}$, define $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ by

$$k(x, x') = \langle \phi(x), \phi(x') \rangle.$$

Then for any $x_1, \ldots, x_n \in \mathcal{X}$, the matrix $K \in \mathbb{R}^n \times \mathbb{R}^n$ with entries

$$K_{ij} = k(x_i, x_j)$$

is positive semi-definite.

Proof. Let $x_1, \ldots, x_n \in \mathcal{X}$, and $\alpha \in \mathbb{R}^n$. Then

$$\begin{aligned}\sum_{i,j} \alpha_i k(x_i, x_j) \alpha_j &= \sum_{i,j} \alpha_i \langle \phi(x_i), \phi(x_j)\rangle \alpha_j \\ &= \left\langle \sum_i \alpha_i \phi(x_i), \sum_j \alpha_j \phi(x_j)\right\rangle \\ &\geq 0\end{aligned}$$

since the inner product is positive definite. □

This suggests the following definition:

Definition (Positive-definite kernel)**.** A *positive-definite kernel* (or simply *kernel*) is a symmetric map $k\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that for all $n \in \mathbb{N}$ and $x_1, \ldots, x_n \in \mathcal{X}$, the matrix $K \in \mathbb{R}^n \times \mathbb{R}^n$ with entries

$$K_{ij} = k(x_i, x_j)$$

is positive semi-definite.

We will prove that every positive-definite kernel comes from a feature map. However, before that, let's look at some examples.

Example. Suppose $k_1, k_2, \ldots, k_n$ are kernels. Then

- If $\alpha_1, \alpha_2 \geq 0$, then $\alpha_1 k_1 + \alpha_2 k_2$ is a kernel. Moreover, if

  $$k(x, x') = \lim_{m \to \infty} k_m(x, x')$$

  exists, then k is a kernel.

- The pointwise product $k_1 k_2$ is a kernel, where

  $$(k_1 k_2)(x, x') = k_1(x, x') k_2(x, x').$$

Example. The *linear kernel* is $k(x, x') = x^T x'$. To see this, we see that this is given by the feature map $\phi = \mathrm{id}$ and taking the standard inner product on $\mathbb{R}^p$.

Example. The *polynomial kernel* is $k(x, x') = (1 + x^T x')^d$ for all $d \in \mathbb{N}$.

We saw this last time with $d = 2$. This is a kernel since both 1 and $x^T x'$ are kernels, and sums and products preserve kernels.

Example. The *Gaussian kernel* is

$$k(x, x') = \exp\left(-\frac{\|x - x'\|_2^2}{2\sigma^2}\right).$$

The quantity σ is known as the *bandwidth*.

To show that it is a kernel, we decompose

$$\|x - x'\|_2^2 = \|x\|_2^2 + \|x'\|_2^2 - 2x^T x'.$$

We define

$$k_1(x, x') = \exp\left(-\frac{\|x\|_2^2}{2\sigma^2}\right) \exp\left(-\frac{\|x'\|_2^2}{2\sigma^2}\right).$$

This is a kernel by taking $\phi(\,\cdot\,) = \exp\left(-\frac{\|\cdot\|_2^2}{2\sigma^2}\right)$.

Next, we can define

$$k_2(x, x') = \exp\left(\frac{x^T x'}{\sigma^2}\right) = \sum_{r=0}^\infty \frac{1}{r!}\left(\frac{x^T x'}{\sigma^2}\right)^r.$$

We see that this is the infinite linear combination of powers of kernels, hence is a kernel. Thus, it follows that $k = k_1 k_2$ is a kernel.

Note also that any feature map giving this k must take values in an infinite dimensional inner product space. Roughly speaking, this is because we have arbitrarily large powers of x and x' in the expansion of k.

Example. The *Sobolev kernel* is defined as follows: we take $\mathcal{X} = [0, 1]$, and let

$$k(x, x') = \min(x, x').$$

A slick proof that this is a kernel is to notice that this is the covariance function of Brownian motion.

Example. The *Jaccard similarity* is defined as follows: Take $\mathcal{X}$ to be the set of all subsets of $\{1, \ldots, p\}$. For $x, x' \in \mathcal{X}$, define

$$k(x, x') = \begin{cases} \frac{|x \cap x'|}{|x \cup x'|} & x \cup x' \neq \emptyset \\ 1 & \text{otherwise} \end{cases}.$$

Theorem (Moore–Aronszajn theorem)**.** For every kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, there exists an inner product space $\mathcal{H}$ and a feature map $\phi : \mathcal{X} \to \mathcal{H}$ such that

$$k(x, x') = \langle \phi(x), \phi(x')\rangle.$$

This is not actually the full Moore–Aronszajn theorem, but a simplified version of it.

Proof. Let $\mathcal{H}$ denote the vector space of functions $f : \mathcal{X} \to \mathbb{R}$ of the form

$$f(\,\cdot\,) = \sum_{i=1}^n \alpha_i k(\,\cdot\,, x_i) \tag{$*$}$$

for some $n \in \mathbb{N}$, $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ and $x_1, \ldots, x_n \in \mathcal{X}$. If

$$g(\,\cdot\,) = \sum_{i=1}^m \beta_j k(\,\cdot\,, x_j') \in \mathcal{H},$$

then we tentatively define the inner product of f and g by

$$\langle f, g\rangle = \sum_{i=1}^n \sum_{j=1}^m \alpha_i \beta_j k(x_i, x_j').$$

We have to check that this is an inner product, but even before that, we need to check that this is well-defined, since f and g can be represented in the form $(*)$ in multiple ways. To do so, simply observe that

$$\sum_{i=1}^n \sum_{j=1}^m \alpha_i \beta_j k(x_i, x_j') = \sum_{i=1}^n \alpha_i g(x_i) = \sum_{j=1}^m \beta_j f(x_j'). \tag{$\dagger$}$$

The first equality shows that the definition of our inner product does not depend on the representation of g, while the second equality shows that it doesn't depend on the representation of f.

To show this is an inner product, note that it is clearly symmetric and bilinear. To show it is positive definite, note that we have

$$\langle f, f\rangle = \sum_{i=1}^n \sum_{j=1}^m \alpha_i k(x_i, x_j)\alpha_j \geq 0$$

since the kernel is positive semi-definite. It remains to check that if $\langle f, f\rangle = 0$, then $f = 0$ as a function. To this end, note the important *reproducing property*: by (†), we have

$$\langle k(\,\cdot\,, x), f\rangle = f(x).$$

This says $k(\cdot, x)$ represents the evaluation-at-x linear functional.

In particular, we have

$$f(x)^2 = \langle k(\,\cdot\,, x), f\rangle^2 \leq \langle k(\,\cdot\,, x), k(\,\cdot\,, x)\rangle\langle f, f\rangle = 0.$$

Here we used the Cauchy–Schwarz inequality, which, if you inspect the proof, does not require positive definiteness, just positive semi-definiteness. So it follows that $f \equiv 0$. Thus, we know that $\mathcal{H}$ is indeed an inner product space.

We now have to construct a feature map. Define $\phi : \mathcal{X} \to \mathcal{H}$ by

$$\phi(x) = k(\,\cdot\,, x).$$

Then we have already observed that

$$\langle \phi(x), \phi(x')\rangle = \langle k(\,\cdot\,, x), k(\,\cdot\,, x')\rangle = k(x, x'),$$

as desired. □

We shall boost this theorem up to its full strength, where we say there is a "unique" inner product space with such a feature map. Of course, it will not be unique unless we impose appropriate quantifiers, since we can just throw in some random elements into $\mathcal{H}$.

Recall (or learn) from functional analysis that any inner product space $\mathcal{B}$ is a normed space, with norm

$$\|f\|^2 = \langle f, f\rangle_{\mathcal{B}}.$$

We say sequence (f_m) in a normed space $\mathcal{B}$ is *Cauchy* if $\|f_m - f_n\|_{\mathcal{B}} \to 0$ as $n, m \to \infty$. An inner product space is *complete* if every Cauchy sequence has a limit (in the space), and a complete inner product space is called a *Hilbert space*.

One important property of a Hilbert space $\mathcal{B}$ is that if V is a closed subspace of $\mathcal{B}$, then every $f \in \mathcal{B}$ has a unique representation as $f = v + w$, where $v \in V$ and

$$w \in V^\perp = \{u \in \mathcal{B} : \langle u, y\rangle = 0 \text{ for all } y \in V\}.$$

By adding limits of Cauchy sequences to $\mathcal{H}$, we can obtain a Hilbert space. Indeed, if (f_m) is Cauchy in $\mathcal{H}$, then

$$|f_m(x) - f_n(x)| \leq k^{1/2}(x, x)\|f_m - f_n\|_{\mathcal{H}} \to 0$$

as $m, n \to \infty$. Since every Cauchy sequence in $\mathbb{R}$ converges (i.e. $\mathbb{R}$ is complete), it makes sense to define a limiting function

$$f(x) = \lim_{n \to \infty} f_n(x),$$

and it can be shown that after augmenting $\mathcal{H}$ with such limits, we obtain a Hilbert space. In fact, it is a special type of Hilbert space.

Definition (Reproudcing kernel Hilbert space (RKHS))**.** A Hilbert space $\mathcal{B}$ of functions $f : \mathcal{X} \to \mathbb{R}$ is a reproducing kernel Hilbert space if for each $x \in \mathcal{X}$, there exists a $k_x \in \mathcal{B}$ such that

$$\langle k_x, f \rangle = f(x)$$

for all $x \in \mathcal{B}$.

The function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ given by

$$k(x, x') = \langle k_x, k'_x \rangle = k_x(x') = k_{x'}(x)$$

is called the *reproducing kernel* associated with $\mathcal{B}$.

By the Riesz representation theorem, this condition is equivalent to saying that pointwise evaluation is continuous.

We know the reproducing kernel associated with an RKHS is a positive-definite kernel, and the Moore–Aronszajn theorem can be extended to show that to each kernel k, there is a unique representing kernel Hilbert space $\mathcal{H}$ and feature map $\phi : \mathcal{X} \to \mathcal{H}$ such that $k(x, x') = \langle \phi(x), \phi(x') \rangle$.

Example. Take the linear kernel

$$k(x, x') = x^T x'.$$

By definition, we have

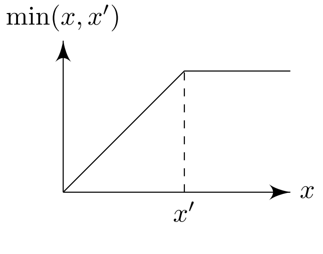
$$\mathcal{H} = \left\{ f : \mathbb{R}^p \to \mathbb{R} \mid f(x) = \sum_{i=1}^n \alpha_i x_i^T x \right\}$$

for some $n \in \mathbb{N}$, $x_1, \ldots, x^n \in \mathbb{R}^p$. We then see that this is equal to

$$\mathcal{H} = \{ f : \mathbb{R}^p \to \mathbb{R} \mid f(x) = \beta^T x \text{ for some } \beta \in \mathbb{R}^p \},$$

and if $f(x) = \beta^T x$, then $\|f\|_{\mathcal{H}}^2 = k(\beta, \beta) = \|\beta\|_2^2$.

Example. Take the Sobolev kernel, where $\mathcal{X} = [0, 1]$ and $k(x, x') = \min(x, x')$. Then $\mathcal{H}$ includes all linear combinations of functions of the form $x \mapsto \min(x, x')$, where $x' \in [0, 1]$, and their pointwise limits. These functions look like



Since we allow arbitrary linear combinations of these things and pointwise limits, this gives rise to a large class of functions. In particular, this includes all Lipschitz functions that are 0 at the origin.

In fact, the resulting space is a Sobolev space, with the norm given by

$$\|f\| = \left(\int_0^1 f'(x)^2 \;\mathrm{d}x\right)^{1/2}.$$

For example, if we take $f = \min(\,\cdot\,, x)$, then by definition we have

$$\|\min(\,\cdot\,, x)\|_{\mathcal{H}}^2 = \min(x, x) = x,$$

whereas the formula we claimed gives

$$\int_0^x 1^2 \;\mathrm{d}t = x.$$

In general, it is difficult to understand the RKHS, but if we can do so, this provides a lot of information on what we are regularizing in the kernel trick.

2.4 Making predictions

Let's now try to understand what exactly we are doing when we do ridge regression with a kernel k. To do so, we first think carefully what we were doing in ordinary ridge regression, which corresponds to using the linear kernel. For the linear kernel, the L^2 norm $\|\beta\|_2^2$ corresponds exactly to the norm in the RKHS $\|f\|_{\mathcal{H}}^2$. Thus, an alternative way of expressing ridge regression is

$$\hat{f} = \operatorname*{argmin}_{f \in \mathcal{H}} \left\{\sum_{i=1}^n (Y_i - f(x_i))^2 + \lambda \|f\|_{\mathcal{H}}^2\right\}, \tag{$*$}$$

where $\mathcal{H}$ is the RKHS of the linear kernel. Now this way of writing ridge regression makes sense for an arbitrary RKHS, so we might think this is what we should solve in general.

But if $\mathcal{H}$ is infinite dimensional, then naively, it would be quite difficult to solve $(*)$. The solution is provided by the representer theorem.

Theorem (Representer theorem)**.** Let $\mathcal{H}$ be an RKHS with reproducing kernel k. Let c be an arbitrary loss function and $J : [0, \infty) \to \mathbb{R}$ any strictly increasing function. Then the minimizer $\hat{f} \in \mathcal{H}$ of

$$Q_1(f) = c(Y, x_1, \ldots, x_n, f(x_1), \ldots, f(x_n)) + J(\|f\|_{\mathcal{H}}^2)$$

lies in the linear span of $\{k(\,\cdot\,, x_i)\}_{i=1}^n$.

Given this theorem, we know our optimal solution can be written in the form

$$\hat{f}(\,\cdot\,) = \sum_{i=1}^n \hat{\alpha}_i k(\,\cdot\,, x_i),$$

and thus we can rewrite our optimization problem as looking for the $\hat{\alpha} \in \mathbb{R}^n$ that minimizes

$$Q_2(\alpha) = c(Y, x_1, \ldots, x_n, K\alpha) + J(\alpha^T K \alpha),$$

over $\alpha \in \mathbb{R}^n$ (with $K_{ij} = k(x_i, x_j)$).

For an arbitrary c, this can still be quite difficult, but in the case of Ridge regression, this tells us that $(*)$ is equivalent to minimizing

$$\|Y - K\alpha\|_2^2 + \lambda \alpha^T K \alpha,$$

and by differentiating, we find that this is given by solving

$$K\hat{\alpha} = K(K + \lambda I)^{-1} Y.$$

Of course $K\hat{\alpha}$ is also our fitted values, so we see that if we try to minimize $(*)$, then the fitted values is indeed given by $K(K + \lambda I)^{-1}Y$, as in our original motivation.

Proof. Suppose $\hat{f}$ minimizes Q_1. We can then write

$$\hat{f} = u + v$$

where $u \in V = \operatorname{span}\{k(\,\cdot\,, x_i) : i = 1, \ldots, n\}$ and $v \in V^\perp$. Then

$$\hat{f}(x_i) = \langle f, k(\,\cdot\,, x_i)\rangle = \langle u + v, k(\,\cdot\,, x_i)\rangle = \langle u, k(\,\cdot\,, x_i)\rangle = u(x_i).$$

So we know that

$$c(Y, x_1, \ldots, x_n, f(x_1), \ldots, f(x_n)) = c(Y, x_1, \ldots, x_n, u(x_1), \ldots, u(x_n)).$$

Meanwhile,

$$\|f\|_{\mathcal{H}}^2 = \|u + v\|_{\mathcal{H}}^2 = \|u\|_{\mathcal{H}}^2 + \|v\|_{\mathcal{H}}^2,$$

using the fact that u and v are orthogonal. So we know

$$J(\|f\|_{\mathcal{H}}^2) \geq J(\|u\|_{\mathcal{H}}^2)$$

with equality iff $v = 0$. Hence $Q_1(f) \geq Q_1(u)$ with equality iff $v = 0$, and so we must have $v = 0$ by optimality.

Thus, we know that the optimizer in fact lies in V, and then the formula of Q_2 just expresses Q_1 in terms of α. □

How well does our kernel machine do? Let $\mathcal{H}$ be an RKHS with reproducing kernel k, and $f^0 \in \mathcal{H}$. Consider a model

$$Y_i = f^0(x_i) + \varepsilon_i$$

for $i = 1, \ldots, n$, and assume $\mathbb{E}\varepsilon = 0$, $\operatorname{var}(\varepsilon) = \sigma^2 I$.

For convenience, We assume $\|f^0\|_{\mathcal{H}}^2 \leq 1$. There is no loss in generality by assuming this, since we can always achieve this by scaling the norm.

Let K be the kernel matrix $K_{ij} = k(x_i, x_j)$ with eigenvalues $d_1 \geq d_2 \geq \cdots \geq d_n \geq 0$. Define

$$\hat{f}_\lambda = \operatorname*{argmin}_{f \in \mathcal{H}} \left\{ \sum_{i=1}^n (Y_i - f(x_i))^2 + \lambda \|f\|_{\mathcal{H}}^2 \right\}.$$

Theorem. We have

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^n \mathbb{E}(f^0(x_i) - \hat{f}_\lambda(x_i))^2 &\leq \frac{\sigma^2}{n}\sum_{i=1}^n \frac{d_i^2}{(d_i+\lambda)^2} + \frac{\lambda}{4n}\\ &\leq \frac{\sigma^2}{n}\frac{1}{\lambda}\sum_{i=1}^n \min\left(\frac{d_i}{4}, \lambda\right) + \frac{\lambda}{4n}.\end{aligned}$$

Given a data set, we can compute the eigenvalues d_i, and thus we can compute this error bound.

Proof. We know from the representer theorem that

$$(\hat{f}_\lambda(x_i), \ldots, \hat{f}_\lambda(x_n))^T = K(K+\lambda I)^{-1} Y.$$

Also, there is some $\alpha \in \mathbb{R}^n$ such that

$$(f^0(x_1), \ldots, f^0(x_n))^T = K\alpha.$$

Moreover, on the example sheet, we show that

$$1 \geq \|f^0\|_{\mathcal{H}}^2 \geq \alpha^T K \alpha.$$

Consider the eigen-decomposition $K = UDU^T$, where U is orthogonal, $D_{ii} = d_i$ and $D_{ij} = 0$ for $i \neq j$. Then we have

$$\mathbb{E}\sum_{i=1}^n (f^0(x_i) - \hat{f}_\lambda(x_i))^2 = \mathbb{E}\|K\alpha - K(K+\lambda I)^{-1}(K\alpha + \varepsilon)\|_2^2$$

Noting that $K\alpha = (K+\lambda I)(K+\lambda I)^{-1}K\alpha$, we obtain

$$\begin{aligned}&= \mathbb{E}\left\|\lambda(K+\lambda I)^{-1}K\alpha - K(K+\lambda I)^{-1}\varepsilon\right\|_2^2\\ &= \underbrace{\lambda^2\left\|(K+\lambda I)^{-1}K\alpha\right\|_2^2}_{\text{(I)}} + \underbrace{\mathbb{E}\left\|K(K+\lambda I)^{-1}\varepsilon\right\|_2^2}_{\text{(II)}}.\end{aligned}$$

At this stage, we can throw in the eigen-decomposition of K to write (I) as

$$\begin{aligned}\text{(I)} &= \lambda^2\left\|(UDU^T + \lambda I)^{-1}UDU^T\alpha\right\|_2^2\\ &= \lambda^2\|U(D+\lambda I)^{-1}\underbrace{DU^T\alpha}_{\theta}\|_2^2\\ &= \sum_{i=1}^n \theta_i^2 \frac{\lambda^2}{(d_i+\lambda)^2}\end{aligned}$$

Now we have

$$\alpha^T K\alpha = \alpha^T UDU^T\alpha = \alpha^T UDD^+DU^T,$$

where D^+ is diagonal and

$$D_{ii}^+ = \begin{cases} d_i^{-1} & d_i > 0\\ 0 & \text{otherwise}\end{cases}.$$

We can then write this as

$$\alpha^T K \alpha = \sum_{d_i > 0} \frac{\theta_i^2}{d_i}.$$

The key thing we know about this is that it is ≤ 1.

Note that by definition of θ_i, we see that if $d_i = 0$, then $\theta_i = 0$. So we can write

$$\text{(II)} = \sum_{i : d_i \geq 0} \frac{\theta_i^2}{d_i} \frac{d_i \lambda^2}{(d_i + \lambda)^2} \leq \lambda \max_{i = 1, \ldots, n} \frac{d_i \lambda}{(d_i + \lambda)^2}$$

by Hölder's inequality with $(p, q) = (1, \infty)$. Finally, use the inequality that

$$(a + b)^2 \geq 4ab$$

to see that we have

$$\text{(I)} \leq \frac{\lambda}{4}.$$

So we are left with (II), which is a bit easier. Using the trace trick, we have

$$\begin{aligned}
\text{(II)} &= \mathbb{E}\varepsilon^T (K + \lambda I)^{-1} K^2 (K + \lambda I)^{-1} \varepsilon\\
&= \mathbb{E} \operatorname{tr}\left(K (K + \lambda I)^{-1} \varepsilon\varepsilon^T (K + \lambda I)^{-1} K\right)\\
&= \operatorname{tr}\left(K (K + \lambda I)^{-1} \mathbb{E}(\varepsilon\varepsilon^T) (K + \lambda I)^{-1} K\right)\\
&= \sigma^2 \operatorname{tr}\left(K^2 (K + \lambda I)^{-2}\right)\\
&= \sigma^2 \operatorname{tr}\left(U D^2 U^T (U D U^T + \lambda I)^{-2}\right)\\
&= \sigma^2 \operatorname{tr}\left(D^2 (D + \lambda I)^{-2}\right)\\
&= \sigma^2 \sum_{i=1}^n \frac{d_i^2}{(d_i + \lambda)^2}.
\end{aligned}$$

Finally, writing $\frac{d_i^2}{(d_i + \lambda)^2} = \frac{d_i}{\lambda} \frac{d_i \lambda}{(d_i + \lambda)^2}$, we have

$$\frac{d_i^2}{(d_i + \lambda)^2} \leq \min\left(1, \frac{d_i}{4\lambda}\right),$$

and we have the second bound. □

How can we interpret this? If we look at the formula, then we see that we can have good benefits if the d_i's decay quickly, and the exact values of the d_i depend only on the choice of x_i. So suppose the x_i are random and idd and independent of ε. Then the entire analysis is still valid by conditioning on $x_1, \ldots, x_n$.

We define $\hat{\mu}_i = \frac{d_i}{n}$, and $\lambda_n = \frac{\lambda}{n}$. Then we can rewrite our result to say

$$\frac{1}{n} \mathbb{E} \sum_{i=1}^n (f^0(x_i) - \hat{f}_\lambda(x_i))^2 \leq \frac{\sigma^2}{\lambda_n} \frac{1}{n} \sum_{i=1}^n \mathbb{E} \min\left(\frac{\hat{\mu}_i}{4}, \lambda_n\right) + \frac{\lambda_n}{4} \equiv \mathbb{E}\delta_n(\lambda_n).$$

We want to relate this more directly to the kernel somehow. Given a density $p(x)$ on $\mathcal{X}$, *Mercer's theorem* guarantees the following eigenexpansion

$$k(x, x') = \sum_{j=1}^\infty \mu_j e_j(x) e_j(x'),$$

where the eigenvalues μ_j and eigenfunctions e_j obey

$$\mu_j e_j(x') = \int_{\mathcal{X}} k(x, x') e_j(x) p(x)\;\mathrm{d}x$$

and

$$\int_{\mathcal{X}} e_k(x) e_j(x) p(x)\;\mathrm{d}x = \mathbf{1}_{j=k}.$$

One can then show that

$$\frac{1}{n}\mathbb{E}\sum_{i=1}^{n} \min\left(\frac{\hat{\mu}_i}{4}, \lambda_n\right) \leq \frac{1}{n}\sum_{i=1}^{\infty} \min\left(\frac{\mu_i}{4}, \lambda_n\right)$$

up to some absolute constant factors. For a particular density $p(x)$ of the input space, we can try to solve the integral equation to figure out what the μ_j's are, and we can find the expected bound.

When k is the Sobolev kernel and $p(x)$ is the uniform density, then it turns out we have

$$\frac{\mu_j}{4} = \frac{1}{\pi^2(2j-1)^2}.$$

By drawing a picture, we see that we can bound $\sum_{i=1}^{\infty} \min\left(\frac{\mu_i}{4}, \lambda_n\right)$ by

$$\sum_{i=1}^{\infty} \min\left(\frac{\mu_i}{4}, \lambda_n\right) \leq \int_0^{\infty} \lambda_n \wedge \frac{1}{\pi^2(2j-1)^2}\;\mathrm{d}j \leq \frac{\sqrt{\lambda_n}}{\pi} + \frac{\lambda_n}{2}.$$

So we find that

$$\mathbb{E}(\delta_n(\lambda_n)) = O\left(\frac{\sigma^2}{n\lambda_n^{1/2}} + \lambda_n\right).$$

We can then find the λ_n that minimizes this, and we see that we should pick

$$\lambda_n \sim \left(\frac{\sigma^2}{n}\right)^{2/3},$$

which gives an error rate of $\sim \left(\frac{\sigma^2}{n}\right)^{2/3}$.

2.5 Other kernel machines

Recall that the representer theorem was much more general than what we used it for. We could apply it to any loss function. In particular, we can apply it to classification problems. For example, we might want to predict whether an email is a spam. In this case, we have $Y \in \{-1, 1\}^n$.

Suppose we are trying to find a hyperplane that separates the two sets $\{x_i\}_{y_i=1}$ and $\{x_i\}_{i:Y_i=-1}$. Consider the really optimistic case where they can indeed be completely separated by a hyperplane through the origin. So there exists $\beta \in \mathbb{R}^p$ (the normal vector) with $y_i x_I^T \beta > 0$ for all i. Moreover, it would be nice if we can maximize the minimum distance between any of the points and the hyperplane defined by β. This is given by

$$\max \frac{y_i x_i^T \beta}{\|\beta\|_2}.$$

Thus, we can formulate our problem as

$$\text{maximize } M \text{ among } \beta \in \mathbb{R}^p, M \geq 0 \text{ subject to } \tfrac{y_i x_i^T \beta}{\|\beta\|_2} \geq M.$$

This optimization problem gives the hyperplane that maximizes the *margin* between the two classes.

Let's think about the more realistic case where we cannot completely separate the two sets. We then want to impose a penalty for each point on the wrong side, and attempt to minimize the distance.

There are two options. We can use the penalty

$$\lambda \sum_{i=1}^n \left(M - \frac{Y_i x_i^T \beta}{\|\beta\|_2}\right)_+$$

where $t_+ = t\mathbf{1}_{t \geq 0}$. Another option is to take

$$\lambda \sum_{i=1}^n \left(1 - \frac{Y_i x_i^T \beta}{M\|\beta\|_2}\right)_+$$

which is perhaps more natural, since now everything is "dimensionless". We will actually use neither, but will start with the second form and massage it to something that can be attacked with our existing machinery. Starting with the second option, we can write our optimization problem as

$$\max_{M \geq 0, \beta \in \mathbb{R}^p} \left(M - \lambda \sum_{i=1}^n \left(1 - \frac{Y_i x_i^T \beta}{M\|\beta\|_2}\right)_+\right).$$

Since the objective function is invariant to any positive scaling of β, we may assume $\|\beta\|_2 = \frac{1}{M}$, and rewrite this problem as maximizing

$$\frac{1}{\|\beta\|_2} - \lambda \sum_{i=1}^n (1 - Y_i x_i^T \beta)_+.$$

Replacing $\max \frac{1}{\|\beta\|_2}$ with minimizing $\|\beta\|_2^2$ and adding instead of subtracting the penalty part, we modify this to say

$$\min_{\beta \in \mathbb{R}^p} \left(\|\beta\|_2^2 + \lambda \sum_{i=1}^n (1 - Y_i x_i^T \beta)_+ \right).$$

The final change we make is that we replace λ with $\frac{1}{\lambda}$, and multiply the whole equation by λ to get

$$\min_{\beta \in \mathbb{R}^p} \left(\lambda\|\beta\|_2^2 + \sum_{i=1}^n (1 - Y_i x_i^T \beta)_+ \right).$$

This looks much more like what we've seen before, with $\lambda\|\beta\|_2^2$ being the penalty term and $\sum_{i=1}^n (1 - Y_i x_i^T \beta)_+$ being the loss function.

The final modification is that we want to allow planes that don't necessarily pass through the origin. To do this, we allow ourselves to translate all the x_i's by a fixed vector $\delta \in \mathbb{R}^p$. This gives

$$\min_{\beta \in \mathbb{R}^p, \delta \in \mathbb{R}^p} \left(\lambda\|\beta\|_2^2 + \sum_{i=1}^n \left(1 - Y_i (x_i - \delta)^T \beta\right)_+ \right)$$

Since $\delta^T\beta$ always appears together, we can simply replace it with a constant μ, and write our problem as

$$(\hat{\mu}, \hat{\beta}) = \operatorname*{argmin}_{(\mu,\beta)\in\mathbb{R}\times\mathbb{R}^p} \sum_{i=1}^n \left(1 - Y_i(x_i^T\beta + \mu)\right)_+ + \lambda\|\beta\|_2^2. \tag{$*$}$$

This gives the *support vector classifier*.

This is still just fitting a hyperplane. Now we would like to "kernelize" this, so that we can fit in a surface instead. Let $\mathcal{H}$ be the RKHS corresponding to the linear kernel. We can then write $(*)$ as

$$(\hat{\mu}_\lambda, \hat{f}_\lambda) = \operatorname*{argmin}_{(\mu,f)\in\mathbb{R}\times\mathcal{H}} \sum_{i=1}^n (1 - Y_i(f(x_i) + \mu))_+ + \lambda\|f\|_{\mathcal{H}}^2.$$

The representer theorem (or rather, a slight variant of it) tells us that the above optimization problem is equivalent to the *support vector machine*

$$(\hat{\mu}_\lambda, \hat{\alpha}_\lambda) = \operatorname*{argmin}_{(\mu,\alpha)\in\mathbb{R}\times\mathbb{R}^n} \sum_{i=1}^n (1 - Y_i(K_i^T\alpha + \mu))_+ + \lambda\alpha^T K\alpha$$

where $K_{ij} = k(x_i, x_j)$ and k is the reproducing kernel of $\mathcal{H}$. Predictions at a new x are then given by

$$\operatorname{sign}\left(\hat{\mu}_\lambda + \sum_{i=1}^n \hat{\alpha}_{\lambda,i} k(x, x_i)\right).$$

One possible alternative to this is to use *logistic regression*. Suppose that

$$\log\left(\frac{\mathbb{P}(Y_i = 1)}{\mathbb{P}(Y_i = -1)}\right) = x_i^T\beta^0,$$

and that $Y_1, \ldots, Y_n$ are independent. An estimate of β^0 can be obtained through maximizing the likelihood, or equivalently,

$$\operatorname*{argmin}_{b\in\mathbb{R}^p} \sum_{i=1}^n \log(1 + \exp(-Y_i x_i^T\beta)).$$

To kernelize this, we introduce an error term of $\lambda\|\beta\|_2^2$, and then kernelize this. The resulting optimization problem is then given by

$$\operatorname*{argmin}_{f\in\mathcal{H}} \left(\sum_{i=1}^n \log(1 + \exp(-Y_i f(x_i))) + \lambda\|f\|_{\mathcal{H}}^2\right).$$

We can then solve this using the representer theorem.

2.6 Large-scale kernel machines

How do we apply kernel machines at large scale? Whenever we want to make a prediction with a kernel machine, we need $O(n)$ many steps, which is quite a pain if we work with a large data set, say a few million of them. But even before that, forming $K \in \mathbb{R}^{n\times n}$ and inverting $K + \lambda I$ or equivalent can be computationally too expensive.

One of the more popular approaches to tackle these difficulties is to form a randomized feature map $\hat{\phi} : \mathcal{X} \to \mathbb{R}^b$ such that

$$\mathbb{E}\hat{\phi}(x)^T\hat{\phi}(x') = k(x, x')$$

To increase the quality of the approximation, we can use

$$x \mapsto \frac{1}{\sqrt{L}}(\hat{\phi}_1(x), \ldots, \hat{\phi}_L(x))^T \in \mathbb{R}^{Lb},$$

where the $\hat{\phi}_i(x)$ are iid with the same distribution as $\hat{\phi}$.

Let Φ be the matrix with ith row $\frac{1}{\sqrt{L}}(\hat{\phi}_1(x), \ldots, \hat{\phi}_L(x))$. When performing, e.g. kernel ridge regression, we can compute

$$\hat{\theta} = (\Phi^T\Phi + \lambda I)^{-1}\Phi^T Y.$$

The computational cost of this is $O((Lb)^3 + n(Lb)^2)$. The key thing of course is that this is linear in n, and we can choose the size of L so that we have a good trade-off between the computational cost and the accuracy of the approximation.

Now to predict a new x, we form

$$\frac{1}{\sqrt{L}}(\hat{\phi}_1(x), \ldots, \hat{\phi}_L(x))^T\hat{\theta},$$

and this is $O(Lb)$.

In 2007, Rahimi and Recht proposed a random map for shift-invariant kernels, i.e. kernels k such that $k(x, x') = h(x - x')$ for some h (we work with $\mathcal{X} = \mathbb{R}^p$). A common example is the Gaussian kernel.

One motivation of this comes from a classical result known as *Bochner's theorem.*

Theorem (Bochner's theorem)**.** Let $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{R}$ be a continuous kernel. Then k is shift-invariant if and only if there exists some distribution F on $\mathbb{R}^p$ and $c > 0$ such that if $W \sim F$, then

$$k(x, x') = c\mathbb{E}\cos((x - x')^T W).$$

Our strategy is then to find some $\hat{\phi}$ depending on W such that

$$c\cos((x - x')^T W) = \hat{\phi}(x)\hat{\phi}(x').$$

We are going to take $b = 1$, so we don't need a transpose on the right.

The idea is then to play around with trigonometric identities to try to write $c\cos((x - x')^T W)$. After some work, we find the following solution:

Let $u \sim U[-\pi, \pi]$, and take $x, y \in \mathbb{R}$. Then

$$\mathbb{E}\cos(x + u)\cos(y + u) = \mathbb{E}(\cos x\cos u - \sin x\sin u)(\cos y\cos u - \sin y\sin u)$$

Since u has the same distribution as $-u$, we see that $\mathbb{E}\cos u\sin u = 0$.

Also, $\cos^2 u + \sin^2 u = 1$. Since u ranges uniformly in $[-\pi, \pi]$, by symmetry, we have $\mathbb{E}\cos^2 u = \mathbb{E}\sin^2 u = \frac{1}{2}$. So we find that

$$2\mathbb{E}\cos(x + u)\cos(y + u) = (\cos x\cos y + \sin x\sin y) = \cos(x - y).$$

Thus, given a shift-invariant kernel k with associated distribution F, we set $W \sim F$ independently of u. Define

$$\hat{\phi}(x) = \sqrt{2c}\cos(W^T x + u).$$

Then

$$\begin{aligned}\mathbb{E}\hat{\phi}(x)\hat{\phi}(x') &= 2c\mathbb{E}[\mathbb{E}[\cos(W^T x + u)\cos(W^T x' + u) \mid W]]\\ &= c\mathbb{E}\cos(W^T(x - x'))\\ &= k(x, x').\end{aligned}$$

In order to get this to work, we must find the appropriate W. In certain cases, this W is actually not too hard to find:

Example. Take

$$k(x, x') = \exp\left(-\frac{1}{2\sigma^2}\|x - x'\|_2^2\right),$$

the Gaussian kernel. If $W \sim N(0, \sigma^{-2}I)$, then

$$\mathbb{E}e^{it^T W} = e^{-\|t\|_2^2/(2\sigma^2)} = \mathbb{E}\cos(t^2 W).$$

3 The Lasso and beyond

We are interested in the situation where we have a very large number of variables compared to the size of the data set. For example the data set might be all the genetic and environmental information about patients, and we want to predict if they have diabetes. The key property is that we expect that most of the data are not useful, and we want to select a small subset of variables that are relevant, and form prediction models based on them.

In these cases, ordinary least squares is not a very good tool. If there are more variables than the number of data points, then it is likely that we can pick the regression coefficients so that our model fits our data exactly, but this model is likely to be spurious, since we are fine-tuning our model based on the random fluctuations of a large number of irrelevant variables. Thus, we want to find a regression method that penalizes non-zero coefficients. Note that Ridge regression is not good enough — it encourages small coefficients, but since it uses the ℓ_2 norm, it's quite lenient towards small, non-zero coefficients, as the square of a small number is really small.

There is another reason to favour models that sets a lot of coefficients to zero. Consider the linear model

$$Y = X\beta^0 + \varepsilon,$$

where as usual $\mathbb{E}\varepsilon = 0$ and $\operatorname{var}(\varepsilon) = \sigma^2 I$. Let $S = \{k : \beta^0_k \neq 0\}$, and let $s = |S|$. As before, we suppose we have some a priori belief that $s \ll p$.

For the purposes of this investigation, suppose X has full column rank, so that we can use ordinary least squares. Then the prediction error is

$$\begin{aligned}
\frac{1}{n}\mathbb{E}\|X\beta^0 - X\hat{\beta}^{OLS}\|_2^2 &= \frac{1}{n}\mathbb{E}(\hat{\beta}^{OLS} - \beta^0)^T X^T X(\hat{\beta}^{OLS} - \beta^0)\\
&= \frac{1}{n}\mathbb{E}\operatorname{tr}(\hat{\beta}^{OLS} - \beta^0)(\hat{\beta}^{OLS} - \beta^0)^T X^T X\\
&= \frac{1}{n}\operatorname{tr}\mathbb{E}(\hat{\beta}^{OLS} - \beta^0)(\hat{\beta}^{OLS} - \beta^0)^T X^T X\\
&= \frac{1}{n}\operatorname{tr}\mathbb{E}\sigma^2(X^T X)^{-1}X^T X\\
&= \sigma^2\frac{p}{n}.
\end{aligned}$$

Note that this does not depend on what what β^0 is, but only on σ^2, p and n.

In the situation we are interested in, we expect $s \ll p$. So if we can find S and find ordinary least squares just on these, then we would have a mean squared prediction error of $\sigma^2\frac{s}{n}$, which would be much much smaller.

We first discuss a few classical model methods that does this.

The first approach we may think of is the *best subsets method*, where we try to do regression on all possible choices of S and see which is the "best". For any set M of indices, we let X_M be the submatrix of X formed from the columns of X with index in M. We then regress Y on X_M for every $M \subseteq \{1, \ldots, p\}$, and then pick the best model via cross-validation, for example. A big problem with this is that the number of subsets grows exponentially with p, and so becomes infeasible for, say, $p > 50$.

Another method might be *forward selection.* We start with an intercept-only model, and then add to the existing model the predictor that reduces the RSS the most, and then keep doing this until a fixed number of predictors have been added. This is quite a nice approach, and is computationally quite fast even if p is large. However, this method is greedy, and if we make a mistake at the beginning, then the method blows up. In general, this method is rather unstable, and is not great from a practical perspective.

3.1 The Lasso estimator

The Lasso (Tibshirani, 1996) is a seemingly small modification of Ridge regression that solves our problems. It solves

$$(\hat{\mu}_\lambda^L, \hat{\beta}_\lambda^L) = \operatorname*{argmin}_{(\mu,\beta)\in\mathbb{R}\times\mathbb{R}^p} \frac{1}{2n}\|Y - \mu 1 - X\beta\|_2^2 + \lambda\|\beta\|_1.$$

The key difference is that we use an ℓ_1 norm on β rather than the ℓ_2 norm,

$$\|\beta\|_1 = \sum_{k=1}^p |\beta_k|.$$

This makes it drastically different from Ridge regression. We will see that for λ large, it will make all of the entries of β exactly zero, as opposed to being very close to zero.

We can compare this to best subset regression, where we replace $\|\beta\|_1$ with something of the form $\sum_{k=1}^p \mathbf{1}_{\beta_k > 0}$. But the beautiful property of the Lasso is that the optimization problem is now continuous, and in fact convex. This allows us to solve it using standard convex optimization techniques.

Why is the ℓ_1 norm so different from the ℓ_2 norm? Just as in Ridge regression, we may center and scale X, and center Y, so that we can remove μ from the objective. Define

$$Q_\lambda(\beta) = \frac{1}{2n}\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1.$$

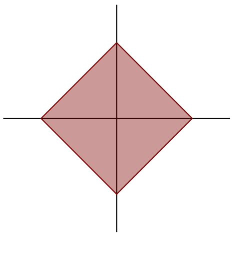
Any minimizer $\hat{\beta}_\lambda^L$ of $Q_\lambda(\beta)$ must also be a solution to

$$\min \|Y - X\beta\|_2^2 \text{ subject to } \|\beta\|_1 \leq \|\hat{\beta}_\lambda^L\|_1.$$

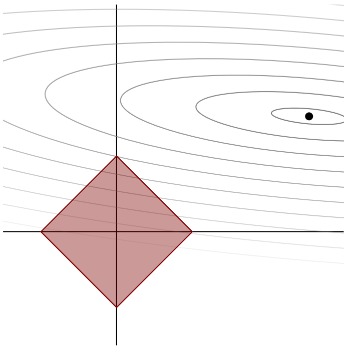
Similarly, $\hat{\beta}_\lambda^R$ is a solution of

$$\min \|Y - X\beta\|_2^2 \text{ subject to } \|\beta\|_2 \leq \|\hat{\beta}_\lambda^R\|_2.$$

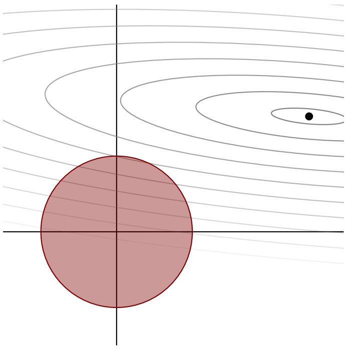
So imagine we are given a value of $\|\hat{\beta}_\lambda^L\|_1$, and we try to solve the above optimization problem with pictures. The region $\|\beta\|_1 \leq \|\hat{\beta}_\lambda^L\|_1$ is given by a rotated square



On the other hand, the minimum of $\|Y - X\beta\|_2^2$ is at $\hat{\beta}^{OLS}$, and the contours are ellipses centered around this point.



To solve the minimization problem, we should pick the smallest contour that hits the square, and pick the intersection point to be our estimate of β^0. The point is that since the unit ball in the ℓ^1-norm has these corners, this β^0 is likely to be on the corners, hence has a lot of zeroes. Compare this to the case of Ridge regression, where the constraint set is a ball:

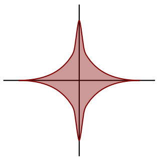


Generically, for Ridge regression, we would expect the solution to be non-zero everywhere.

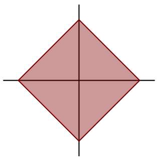
More generally, we can try to use the ℓ_q norm given by

$$\|\beta\|_q = \left(\sum_{k=1}^p \beta_k^q\right)^{1/q}.$$

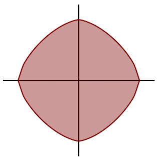
We can plot their unit spheres, and see that they look like



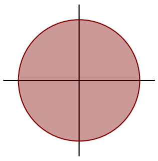
$q = 0.5$



$q = 1$



$q = 1.5$



$q = 2$

We see that $q = 1$ is the smallest value of q for which there are corners, and also the largest value of q for which the constraint set is still convex. Thus, $q = 1$ is the sweet spot for doing regression.

But is this actually good? Suppose the columns of X are scaled to have ℓ_2 norm $\sqrt{n}$, and, after centering, we have a normal linear model

$$Y = X\beta^0 + \varepsilon - \bar{\varepsilon}1,$$

with $\varepsilon \sim N_n(0, \sigma^2 I)$.

Theorem. Let $\hat{\beta}$ be the Lasso solution with

$$\lambda = A\sigma\sqrt{\frac{\log p}{n}}$$

for some A. Then with probability $1 - 2p^{-(A^2/2-1)}$, we have

$$\frac{1}{n}\|X\beta^0 - X\hat{\beta}\|_2^2 \leq 4A\sigma\sqrt{\frac{\log p}{n}}\|\beta^0\|_1.$$

Crucially, this is proportional to $\log p$, and not just p. On the other hand, unlike what we usually see for ordinary least squares, we have $\frac{1}{\sqrt{n}}$, and not $\frac{1}{n}$.

We will later obtain better bounds when we make some assumptions on the design matrix.

Proof. We don't really have a closed form solution for $\hat{\beta}$, and in general it doesn't exist. So the only thing we can use is that it in fact minimizes $Q_\lambda(\beta)$. Thus, by definition, we have

$$\frac{1}{2n}\|Y - X\hat{\beta}\|_2^2 + \lambda\|\hat{\beta}\|_1 \leq \frac{1}{2n}\|Y - X\beta^0\|_2^2 + \lambda\|\beta^0\|_1.$$

We know exactly what Y is. It is $X\beta^0 + \varepsilon - \bar{\varepsilon}1$. If we plug this in, we get

$$\frac{1}{2n}\|X\beta^0 - X\hat{\beta}\|_2^2 \leq \frac{1}{n}\varepsilon^T X(\hat{\beta} - \beta^0) + \lambda\|\beta^0\|_1 - \lambda\|\hat{\beta}\|_1.$$

Here we use the fact that X is centered, and so is orthogonal to 1.

Now Hölder tells us

$$|\varepsilon^T X(\hat{\beta} - \beta^0)| \leq \|X^T\varepsilon\|_\infty \|\hat{\beta} - \beta^0\|_1.$$

We'd like to bound $\|X^T\varepsilon\|_\infty$, but it can be arbitrarily large since ε is a Gaussian. However, with high probability, it is small. Precisely, define

$$\Omega = \left\{\frac{1}{n}\|X^T\varepsilon\|_\infty \leq \lambda\right\}.$$

In a later lemma, we will show later that $\mathbb{P}(\Omega) \geq 1 - 2p^{-(A^2/2-1)}$. Assuming Ω holds, we have

$$\frac{1}{2n}\|X\beta^0 - X\hat{\beta}\|_2^2 \leq \lambda\|\hat{\beta} - \beta^0\| - \lambda\|\hat{\beta}\| + \lambda\|\beta^0\| \leq 2\lambda\|\beta^0\|_1. \qquad \square$$

3.2 Basic concentration inequalities

We now have to prove the lemma we left out in the theorem just now. In this section, we are going to prove a bunch of useful inequalities that we will later use to prove bounds.

Consider the event Ω as defined before. Then

$$\begin{aligned}\mathbb{P}\left(\frac{1}{n}\|X^T\varepsilon\|_\infty > \lambda\right) &= \mathbb{P}\left(\bigcup_{j=1}^p \left\{\frac{1}{n}|X_j^T\varepsilon| > \lambda\right\}\right)\\ &\leq \sum_{j=1}^p \mathbb{P}\left(\frac{1}{n}|X_j^T\varepsilon| > \lambda\right).\end{aligned}$$

Now $\frac{1}{n}X_j^T\varepsilon \sim N(0, \frac{\sigma^2}{n})$. So we just have to bound tail probabilities of normal random variables.

The simplest tail bound we have is Markov's inequality.

Lemma (Markov's inequality)**.** Let W be a non-negative random variable. Then

$$\mathbb{P}(W \geq t) \leq \frac{1}{t}\mathbb{E}W.$$

Proof. We have

$$t\mathbf{1}_{W \geq t} \leq W.$$

The result then follows from taking expectations and then dividing both sides by t. □

While this is a very simple bound, it is actually quite powerful, because it assumes almost nothing about W. This allows for the following trick: given any strictly increasing function $\varphi : \mathbb{R} \to (0, \infty)$ and any random variable W, we have

$$\mathbb{P}(W \geq t) = \mathbb{P}(\varphi(W) \geq \varphi(t)) \leq \frac{\mathbb{E}\varphi(W)}{\varphi(t)}.$$

So we get a bound on the tail probability of W for *any* such function. Even better, we can minimize the right hand side over a class of functions to get an even better bound.

In particular, applying this with $\varphi(t) = e^{\alpha t}$ gives

Corollary (Chernoff bound)**.** For any random variable W, we have

$$\mathbb{P}(W \geq t) \leq \inf_{\alpha > 0} e^{-\alpha t}\mathbb{E}e^{\alpha W}.$$

Note that $\mathbb{E}e^{\alpha W}$ is just the *moment generating function* of W, which we can compute quite straightforwardly.

We can immediately apply this when W is a normal random variable, $W \sim N(0, \sigma^2)$. Then

$$\mathbb{E}e^{\alpha W} = e^{\alpha^2\sigma^2/2}.$$

So we have

$$\mathbb{P}(W \geq t) \leq \inf_{\alpha > 0} \exp\left(\frac{\alpha^2\sigma^2}{2} - \alpha t\right) = e^{-t^2/(2\sigma^2)}.$$

Observe that in fact this tail bound works for any random variable whose moment generating function is bounded above by $e^{\alpha^2\sigma^2/2}$.

Definition (Sub-Gaussian random variable)**.** A random variable W is *sub-Gaussian* (with parameter σ) if

$$\mathbb{E}e^{\alpha(W-\mathbb{E}W)} \leq e^{\alpha^2\sigma^2/2}$$

for all $\alpha \in \mathbb{R}$.

Corollary. Any sub-Gaussian random variable W with parameter σ satisfies

$$\mathbb{P}(W \geq t) \leq e^{-t^2/2\sigma^2}. \qquad \square$$

In general, bounded random variables are sub-Gaussian.

Lemma (Hoeffding's lemma)**.** If W has mean zero and takes values in $[a, b]$, then W is sub-Gaussian with parameter $\frac{b-a}{2}$. □

Recall that the sum of two independent Gaussians is still a Gaussian. This continues to hold for sub-Gaussian random variables.

Proposition. Let $(W_i)_{i=1}^n$ be independent mean-zero sub-Gaussian random variables with parameters $(\sigma_i)_{i=0}^n$, and let $\gamma \in \mathbb{R}^n$. Then $\gamma^T W$ is sub-Gaussian with parameter

$$\left(\sum (\gamma_i\sigma_i)^2\right)^{1/2}.$$

Proof. We have

$$\begin{aligned}\mathbb{E}\exp\left(\alpha\sum_{i=1}^n \gamma_i W_i\right) &= \prod_{i=1}^n \mathbb{E}\exp\left(\alpha\gamma_i W_i\right)\\ &\leq \prod_{i=1}^n \exp\left(\frac{\alpha^2}{2}\gamma_i^2\sigma_i^2\right)\\ &= \exp\left(\frac{\alpha^2}{2}\sum_{i=1}^n \sigma_i^2\gamma_i^2\right). \qquad \square\end{aligned}$$

We can now prove our bound for the Lasso, which in fact works for any sub-Gaussian random variable.

Lemma. Suppose $(\varepsilon_i)_{i=1}^n$ are independent, mean-zero sub-Gaussian with common parameter σ. Let

$$\lambda = A\sigma\sqrt{\frac{\log p}{n}}.$$

Let X be a matrix whose columns all have norm $\sqrt{n}$. Then

$$\mathbb{P}\left(\frac{1}{n}\|X^T\varepsilon\|_\infty \leq \lambda\right) \geq 1 - 2p^{-(A^2/2-1)}.$$

In particular, this includes $\varepsilon \sim N_n(0, \sigma^2 I)$.

Proof. We have

$$\mathbb{P}\left(\frac{1}{n}\|X^T\varepsilon\|_\infty > \lambda\right) \leq \sum_{j=1}^p \mathbb{P}\left(\frac{1}{n}|X_j^T\varepsilon| > \lambda\right).$$

But $\pm\frac{1}{n}X_j^T\varepsilon$ are both sub-Gaussian with parameter

$$\sigma\left(\sum_i \left(\frac{X_{ij}}{n}\right)^2\right)^{1/2} = \frac{\sigma}{\sqrt{n}}.$$

Then by our previous corollary, we get

$$\sum_{j=1}^p \mathbb{P}\left(\frac{1}{n}|X_j^T\varepsilon|_\infty > \lambda\right) \leq 2p\exp\left(-\frac{\lambda^2 n}{2\sigma^2}\right).$$

Note that we have the factor of 2 since we need to consider the two cases $\frac{1}{n}X_j^T\varepsilon > \lambda$ and $-\frac{1}{n}X_j^T\varepsilon > \lambda$.

Plugging in our expression of λ, we write the bound as

$$2p\exp\left(-\frac{1}{2}A^2\log p\right) = 2p^{1-A^2/2}.$$ □

This is all we need for our result on the Lasso. We are going to go a bit further into this topic of concentration inequalities, because we will need them later when we impose conditions on the design matrix. In particular, we would like to bound the tail probabilities of products.

Definition (Bernstein's condition)**.** We say that a random variable W satisfies Bernstein's condition with parameters (σ, b) where $a, b > 0$ if

$$\mathbb{E}[|W - \mathbb{E}W|^k] \leq \frac{1}{2}k!\,\sigma^2 b^{k-2}$$

for $k = 2, 3, \ldots$.

The point is that these bounds on the moments lets us bound the moment generating function of W.

Proposition (Bernstein's inequality)**.** Let $W_1, W_2, \ldots, W_n$ be independent random variables with $\mathbb{E}W_i = \mu$, and suppose each W_i satisfies Bernstein's condition with parameters (σ, b). Then

$$\mathbb{E}e^{\alpha(W_i - \mu)} \leq \exp\left(\frac{\alpha^2\sigma^2/2}{1 - b|\alpha|}\right) \text{ for all } |\alpha| < \frac{1}{b},$$

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^n W_i - \mu \geq t\right) \leq \exp\left(-\frac{nt^2}{2(\sigma^2 + bt)}\right) \text{ for all } t \geq 0.$$

Note that for large t, the bound goes as e^{-t} instead of e^{-t^2}.

Proof. For the first part, we fix i and write $W = W_i$. Let $|\alpha| < \frac{1}{b}$. Then

$$\begin{aligned}
\mathbb{E}e^{\alpha(W_i - \mu)} &= \sum_{k=0}^\infty \mathbb{E}\left[\frac{1}{k!}\alpha^k|W_i - \mu|^k\right] \\
&\leq 1 + \frac{\sigma^2\alpha^2}{2}\sum_{k=2}^\infty |\alpha|^{k-2}b^{k-2} \\
&= 1 + \frac{\sigma^2\alpha^2}{2}\frac{1}{1 - |\alpha|b} \\
&\leq \exp\left(\frac{\alpha^2\sigma^2/2}{1 - b|\alpha|}\right).
\end{aligned}$$

Then

$$\begin{aligned}\mathbb{E}\exp\left(\frac{1}{n}\sum_{i=1}^n \alpha(W_i - \mu)\right) &= \prod_{i=1}^n \mathbb{E}\exp\left(\frac{\alpha}{n}(W_i - \mu)\right)\\ &\leq \exp\left(n\frac{\left(\frac{\alpha}{n}\right)^2 \sigma^2/2}{1 - b\left|\frac{\alpha}{n}\right|}\right),\end{aligned}$$

assuming $\left|\frac{\alpha}{n}\right| < \frac{1}{b}$. So it follows that

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^n W_i - \mu \geq t\right) \leq e^{-\alpha t}\exp\left(n\frac{\left(\frac{\alpha}{n}\right)^2 \sigma^2/2}{1 - b\left|\frac{\alpha}{n}\right|}\right).$$

Setting

$$\frac{\alpha}{n} = \frac{t}{bt + \sigma^2} \in \left[0, \frac{1}{b}\right)$$

gives the result. □

Lemma. Let W, Z be mean-zero sub-Gaussian random variables with parameters σ_W and σ_Z respectively. Then WZ satisfies Bernstein's condition with parameter $(8\sigma_W\sigma_Z, 4\sigma_W\sigma_Z)$.

Proof. For any random variable Y (which we will later take to be WZ), for $k > 1$, we know

$$\begin{aligned}\mathbb{E}|Y - \mathbb{E}Y|^k &= 2^k\mathbb{E}\left|\frac{1}{2}Y - \frac{1}{2}\mathbb{E}Y\right|^k\\ &\leq 2^k\mathbb{E}\left|\frac{1}{2}|Y| + \frac{1}{2}|\mathbb{E}Y|\right|^k.\end{aligned}$$

Note that

$$\left|\frac{1}{2}|Y| + \frac{1}{2}|\mathbb{E}Y|\right|^k \leq \frac{|Y|^k + |\mathbb{E}Y|^k}{2}$$

by Jensen's inequality. Applying Jensen's inequality again, we have

$$|\mathbb{E}Y|^k \leq \mathbb{E}|Y|^k.$$

Putting the whole thing together, we have

$$\mathbb{E}|Y - \mathbb{E}Y|^k \leq 2^k\mathbb{E}|Y|^k.$$

Now take $Y = WZ$. Then

$$\mathbb{E}|WZ - \mathbb{E}WZ| \leq 2^k\mathbb{E}|WZ|^k \leq 2^k\left(\mathbb{E}W^{2k}\right)^{1/2}(\mathbb{E}Z^{2k})^{1/2},$$

by the Cauchy–Schwarz inequality.

We know that sub-Gaussians satisfy a bound on the tail probability. We can then use this to bound the moments of W and Z. First note that

$$W^{2k} = \int_0^\infty \mathbf{1}_{x < W^{2k}}\;\mathrm{d}x.$$

Taking expectations of both sides, we get

$$\mathbb{E}W^{2k} = \int_0^\infty \mathbb{P}(x < W^{2k})\ \mathrm{d}x.$$

Since we have a tail bound on W instead of W^{2k}, we substitute $x = t^{2k}$. Then $\mathrm{d}x = 2kt^{2k-1}\ \mathrm{d}t$. So we get

$$\begin{aligned}\mathbb{E}W^{2k} &= 2k \int_0^\infty t^{2k-1}\mathbb{P}(|W| > t)\ \mathrm{d}t\\ &= 4k \int_0^\infty t^{2k} \exp\left(-\frac{t^2}{2\sigma_N^2}\right)\ \mathrm{d}t.\end{aligned}$$

where again we have a factor of 2 to account for both signs. We perform yet another substitution

$$x = \frac{t^2}{2\sigma_N^2},\quad \mathrm{d}x = \frac{t}{\sigma_W^2}\ \mathrm{d}t.$$

Then we get

$$\mathbb{E}W^{2k} = 2^{k+1}\sigma_W^{2k} k\sigma_W^2 \int_0^\infty x^{k-1}e^{-x}\ \mathrm{d}x = 4\cdot k!\sigma_W^2.$$

Plugging this back in, we have

$$\begin{aligned}\mathbb{E}|WZ - \mathbb{E}WZ|^k &\leq 2^k 2^{k+1} k! \sigma_W^k \sigma_Z^k \sigma_Z^k\\ &= \frac{1}{2}k!2^{2k+2}\sigma_W^k\sigma_Z^k\\ &= \frac{1}{2}k!(8\sigma_W\sigma_Z)^2(4\sigma_W\sigma_Z)^{k-2}.\end{aligned}$$

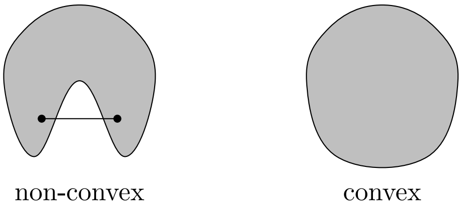
□

3.3 Convex analysis and optimization theory

We'll leave these estimates aside for a bit, and give more background on convex analysis and convex optimization. Recall the following definition:

Definition (Convex set)**.** A set $A \subseteq \mathbb{R}^d$ is convex if for any $x, y \in A$ and $t \in [0, 1]$, we have

$$(1-t)x + ty \in A.$$



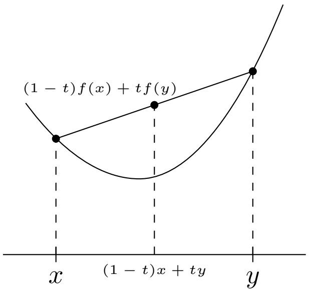
We are actually more interested in convex functions. We shall let our functions to take value ∞, so let us define $\bar{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$. The point is that if we want our function to be defined in $[a, b]$, then it is convenient to extend it to be defined on all of $\mathbb{R}$ by setting the function to be ∞ outside of $[a, b]$.

Definition (Convex function)**.** A function $f : \mathbb{R}^d \to \bar{\mathbb{R}}$ is *convex* iff

$$f((1-t)x + ty) \le (1-t)f(x) + tf(y)$$

for all $x, y \in \mathbb{R}^d$ and $t \in (0,1)$. Moreover, we require that $f(x) < \infty$ for at least one x.

We say it is *strictly convex* if the inequality is strict for all x, y and $t \in (0,1)$.



Definition (Domain)**.** Define the *domain* of a function $f : \mathbb{R}^d \to \bar{\mathbb{R}}$ to be

$$\operatorname{dom} f = \{x : f(x) < \infty\}.$$

One sees that the domain of a convex function is always convex.

Proposition.

(i) Let $f_1, \ldots, f_m : \mathbb{R}^d \to \bar{\mathbb{R}}$ be convex with $\operatorname{dom} f_1 \cap \cdots \cap \operatorname{dom} f_m \neq \emptyset$, and let $c_1, \ldots, c_m \ge 0$. Then $c_1 + \cdots + c_m f_m$ is a convex function.

(ii) If $f : \mathbb{R}^d \to \mathbb{R}$ is twice continuously differentiable, then

(a) f is convex iff its Hessian is positive semi-definite everywhere.

(b) f is strictly convex if its Hessian positive definite everywhere. □

Note that having a positive definite Hessian is not necessary for strict convexity, e.g. x^4 is strictly convex but has vanishing Hessian at 0.

Now consider a constrained optimization problem

$$\text{minimize } f(x) \text{ subject to } g(x) = 0$$

where $x \in \mathbb{R}^d$ and $g : \mathbb{R}^d \to \mathbb{R}^b$. The *Lagrangian* for this problem is

$$L(x, \theta) = f(x) + \theta^T g(x).$$

Why is this helpful? Suppose c^* is the minimum of f. Then note that for any θ, we have

$$\inf_{x \in \mathbb{R}^d} L(x, \theta) \le \inf_{x \in \mathbb{R}^d, g(x) = 0} L(x, \theta) = \inf_{x \in \mathbb{R}^d : g(x) = 0} f(x) = c^*.$$

Thus, if we can find some θ^*, x^* such that x^* minimizes $L(x, \theta^*)$ and $g(x^*) = 0$, then this is indeed the optimal solution.

This gives us a method to solve the optimization problem — for each fixed θ, solve the unconstrained optimization problem $\operatorname{argmin}_x L(x, \theta)$. If we are doing this analytically, then we would have a formula for x in terms of θ. Then seek for a θ such that $g(x) = 0$ holds.

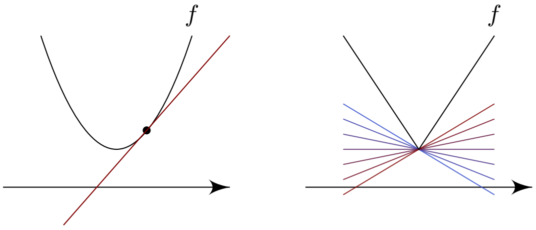
Subgradients

Usually, when we have a function to optimize, we take its derivative and set it to zero. This works well if our function is actually differentiable. However, the ℓ_1 norm is not a differentiable function, since $|x|$ is not differentiable at 0. This is not some exotic case we may hope to avoid most of the time — when solving the Lasso, we actively want our solutions to have zeroes, so we really want to get to these non-differentiable points.

Thus, we seek some generalized notion of derivative that works on functions that are not differentiable.

Definition (Subgradient)**.** A vector $v \in \mathbb{R}^d$ is a *subgradient* of a convex function at x if $f(y) \geq f(x) + v^T(y - x)$ for all $y \in \mathbb{R}^d$.

The set of subgradients of f at x is denoted $\partial f(x)$, and is called the *subdifferential*.



This is indeed a generalization of the derivative, since

Proposition. Let f be convex and differentiable at $x \in \mathrm{int}(\mathrm{dom}\, f)$. Then $\partial f(x) = \{\nabla f(x)\}$. □

The following properties are immediate from definition.

Proposition. Suppose f and g are convex with $\mathrm{int}(\mathrm{dom}\, f) \cap \mathrm{int}(\mathrm{dom}\, g) \neq \emptyset$, and $\alpha > 0$. Then

$$\begin{aligned} \partial(\alpha f)(x) &= \alpha \partial f(x) = \{\alpha v : v \in \partial f(x)\} \\ \partial(f + g)(x) &= \partial g(x) + \partial f(x). \end{aligned}$$ □

The condition (for convex differentiable functions) that "x is a minimum iff $f'(x) = 0$" now becomes

Proposition. If f is convex, then

$$x^* \in \operatorname*{argmin}_{x \in \mathbb{R}^d} f(x) \Leftrightarrow 0 \in \partial f(x^*).$$

Proof. Both sides are equivalent to the requirement that $f(y) \geq f(x^*)$ for all y. □

We are interested in applying this to the Lasso. So we want to compute the subdifferential of the ℓ_1 norm. Let's first introduce some notation.

Notation. For $x \in \mathbb{R}^d$ and $A \subseteq \{1 \ldots, d\}$, we write x_A for the sub-vector of x formed by the components of x induced by A. We write $x_{-j} = x_{\{j\}^c} = x_{\{1,\ldots,d\} \setminus j}$. Similarly, we write $x_{-jk} = x_{\{jk\}^c}$ etc.

We write

$$\operatorname{sgn}(x_i) = \begin{cases} -1 & x_i < 0 \\ 1 & x_i > 0 \\ 0 & x_i = 0 \end{cases},$$

and $\operatorname{sgn}(x) = (\operatorname{sgn}(x_1), \ldots, \operatorname{sgn}(x_d))^T$.

First note that $\|\cdot\|_1$ is convex, as it is a norm.

Proposition. For $x \in \mathbb{R}^d$ and $A \in \{j : x_j \neq 0\}$, we have

$$\partial \|x\|_1 = \{v \in \mathbb{R}^d : \|v\|_\infty \leq 1, v_A = \operatorname{sgn}(x_A)\}.$$

Proof. It suffices to look at the subdifferential of the absolute value function, and then add them up.

For $j = 1, \ldots, d$, we define $g_j : \mathbb{R}^d \to \mathbb{R}$ by sending x to $|x_j|$. If $x_j \neq 0$, then g_j is differentiable at x, and so we know $\partial g_j(x) = \{\operatorname{sgn}(x_j) e_j\}$, with e_j the jth standard basis vector.

When $x_j = 0$, if $v \in \partial g_j(x_j)$, then

$$g_j(y) \geq g_j(x) + v^T(y - x).$$

So

$$|y_j| \geq v^T(y - x).$$

We claim this holds iff $v_j \in [-1, 1]$ and $v_{-j} = 0$. The $\Leftarrow$ direction is an immediate calculation, and to show $\Rightarrow$, we pick $y_{-j} = v_{-j} + x_{-j}$, and $y_j = 0$. Then we have

$$0 \geq v_{-j}^T v_{-j}.$$

So we know that $v_{-j} = 0$. Once we know this, the condition says

$$|y_j| \geq v_j y_j$$

for all y_j. This is then true iff $v_j \in [-1, 1]$. Forming the set sum of the $\partial g_j(x)$ gives the result. □

3.4 Properties of Lasso solutions

Let's now apply this to the Lasso. Recall that the Lasso objective was

$$Q_\lambda(\beta) = \frac{1}{2n}\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1.$$

We know this is a convex function in β. So we know $\hat{\beta}_\lambda^L$ is a minimizer of $Q_\lambda(\beta)$ iff $0 \in \partial Q_\lambda(\hat{\beta})$.

Let's subdifferentiate Q_λ and see what this amounts to. We have

$$\partial Q_\lambda(\hat{\beta}) = \left\{-\frac{1}{n}X^T(Y - X\beta)\right\} + \lambda\left\{\hat{\nu} \in \mathbb{R}^d : \|\hat{\nu}\|_\infty \leq 1, \hat{\nu}|_{\hat{\rho}_\lambda^L} = \operatorname{sgn}(\hat{\beta}^L_{\lambda, \hat{\rho}_\lambda^L})\right\},$$

where $\hat{\rho}_\lambda^L = \{j : \hat{\beta}^L_{\lambda,j} \neq 0\}$.

Thus, $0 \in \partial Q_\lambda(\hat{\beta}_\lambda)$ is equivalent to

$$\hat{\nu} = \frac{1}{\lambda} \cdot \frac{1}{n} X^T (Y - X\hat{\beta}_\lambda)$$

satisfying

$$\|\hat{\nu}\|_\infty \leq 1, \quad \hat{\nu}_{\hat{\rho}^L_\lambda} = \operatorname{sgn}(\hat{\beta}^L_{\lambda, \hat{\rho}^L_\lambda}).$$

These are known as the *KKT conditions* for the Lasso.

In principle, there could be several solutions to the Lasso. However, at least the fitted values are always unique.

Proposition. $X\hat{\beta}^L_\lambda$ is unique.

Proof. Fix $\lambda > 0$ and stop writing it. Suppose $\hat{\beta}^{(1)}$ and $\hat{\beta}^{(2)}$ are two Lasso solutions at λ. Then

$$Q(\hat{\beta}^{(1)}) = Q(\hat{\beta}^{(2)}) = c^*.$$

As Q is convex, we know

$$c^* = Q\left(\frac{1}{2}\hat{\beta}^{(1)} + \frac{1}{2}\hat{\beta}^{(2)}\right) \leq \frac{1}{2} Q(\hat{\beta}^{(1)}) + \frac{1}{2} Q(\hat{\beta}^{(2)}) = c^*.$$

So $\frac{1}{2}\hat{\beta}^{(1)} + \frac{1}{2}\hat{\beta}^{(2)}$ is also a minimizer.

Since the two terms in $Q(\beta)$ are individually convex, it must be the case that

$$\begin{aligned} \left\| \frac{1}{2}(Y - X\hat{\beta}^{(1)}) + \frac{1}{2}(Y - X\hat{\beta}^{(2)}) \right\|_2^2 &= \frac{1}{2} \left\| Y - X\hat{\beta}^{(1)} \right\|_2^2 + \frac{1}{2} \left\| Y - X\hat{\beta}^{(2)} \right\|_2^2 \\ \left\| \frac{1}{2}(\hat{\beta}^{(1)} + \hat{\beta}^{(2)}) \right\|_1 &= \frac{1}{2} \|\hat{\beta}^{(1)}\|_1 + \frac{1}{2} \|\hat{\beta}^{(2)}\|_1. \end{aligned}$$

Moreover, since $\|\cdot\|_2^2$ is strictly convex, we can have equality only if $X\hat{\beta}^{(1)} = X\hat{\beta}^{(2)}$. So we are done. □

Definition (Equicorrelation set)**.** Define the *equicorrelation set* $\hat{E}_\lambda$ to be the set of k such that

$$\frac{1}{n} |X_k^T (Y - X\hat{\beta}^L_\lambda)| = \lambda,$$

or equivalently, the k with $\nu_k = \pm 1$, which is well-defined since it depends only on the fitted values.

By the KKT conditions, $\hat{E}_\lambda$ contains the set of non-zeroes of Lasso solution, but may be strictly bigger than that.

Note that if $\operatorname{rk}(X_{\hat{E}_\lambda}) = |\hat{E}_\lambda|$, then the Lasso solution must be unique since

$$X_{\hat{E}_\lambda}(\hat{\beta}^{(1)} - \hat{\beta}^{(2)}) = 0.$$

So $\hat{\beta}^{(1)} = \hat{\beta}^{(2)}$.

3.5 Variable selection

So we have seen that the Lasso picks out some "important" variables and discards the rest. How well does it do the job?

For simplicity, consider a noiseless linear model

$$Y = X\beta^0.$$

Our objective is to find the set

$$S = \{k : \beta_k^0 \neq 0\}.$$

We may wlog assume $S = \{1, \ldots, s\}$, and $N = \{1 \ldots p\} \setminus S$ (as usual X is $n \times p$). We further assume that $\operatorname{rk}(X_S) = s$.

In general, it is difficult to find out S even if we know $|S|$. Under certain conditions, the Lasso can do the job correctly. This condition is dictated by the ℓ_∞ norm of the quantity

$$\Delta = X_N^T X_S (X_S^T X_S)^{-1} \operatorname{sgn}(\beta_S^0).$$

We can understand this a bit as follows — the kth entry of this is the dot product of $\operatorname{sgn}(\beta_S^0)$ with $(X_S^T X_S)^{-1} X_S^T X_k$. This is the coefficient vector we would obtain if we tried to regress X_k on X_S. If this is large, then this suggests we expect X_k to look correlated to X_S, and so it would be difficult to determine if k is part of S or not.

Theorem.

(i) If $\|\Delta\|_\infty \leq 1$, or equivalently

$$\max_{k \in N} |\operatorname{sgn}(\beta_S^0)^T (X_S^T X_S)^{-1} X_S^T X_k| \leq 1,$$

and moreover

$$|\beta_k^0| > \lambda \left| \operatorname{sgn}(\beta_S^0)^T \left(\frac{1}{n} X_j^T X_j\right)_k^{-1} \right|$$

for all $k \in S$, then there exists a Lasso solution $\hat{\beta}_\lambda^L$ with $\operatorname{sgn}(\hat{\beta}_\lambda^L) = \operatorname{sgn}(\beta^0)$.

(ii) If there exists a Lasso solution with $\operatorname{sgn}(\hat{\beta}_\lambda^L) = \operatorname{sgn}(\beta^0)$, then $\|\Delta\|_\infty \leq 1$.

We are going to make heavy use of the KKT conditions.

Proof. Write $\hat{\beta} = \hat{\beta}_\lambda^L$, and write $\hat{S} = \{k : \hat{\beta}_k \neq 0\}$. Then the KKT conditions are that

$$\frac{1}{n} X^T (\beta^0 - \hat{\beta}) = \lambda \hat{\nu},$$

where $\|\hat{\nu}\|_\infty \leq 1$ and $\hat{\nu}_{\hat{S}} = \operatorname{sgn}(\hat{\beta}_{\hat{S}})$.

We expand this to say

$$\frac{1}{n} \begin{pmatrix} X_S^T X_S & X_S^T X_N \\ X_N^T X_S & XN^T X_N \end{pmatrix} \begin{pmatrix} \beta_S^0 - \hat{\beta}_S \\ -\hat{\beta}_N \end{pmatrix} = \lambda \begin{pmatrix} \hat{\nu}_S \\ \hat{\nu}_N \end{pmatrix}.$$

Call the top and bottom equations (1) and (2) respectively.

It is easier to prove (ii) first. If there is such a solution, then $\hat{\beta}_N = 0$. So from (1), we must have

$$\frac{1}{n} X_S^T X_S (\beta_S^0 - \hat{\beta}_S) = \lambda \hat{\nu}_S.$$

Inverting $\frac{1}{n} X_S^T X_S$, we learn that

$$\beta_S^0 - \hat{\beta}_S = \lambda \left(\frac{1}{n} X_S^T X_S\right)^{-1} \operatorname{sgn}(\beta_S^0).$$

Substitute this into (2) to get

$$\lambda \frac{1}{n} X_N^T X_S \left(\frac{1}{n} X_S^T X_S\right)^{-1} \operatorname{sgn}(\beta_S^0) = \lambda \hat{\nu}_N.$$

By the KKT conditions, we know $\|\hat{\nu}_N\|_\infty \leq 1$, and the LHS is exactly $\lambda \Delta$.

To prove (1), we need to exhibit a $\hat{\beta}$ that agrees in sign with $\hat{\beta}$ and satisfies the equations (1) and (2). In particular, $\hat{\beta}_N = 0$. So we try

$$\begin{aligned} (\hat{\beta}_S, \hat{\nu}_S) &= \left(\beta_S^0 - \lambda \left(\frac{1}{n} X_S^T X_S\right)^{-1} \operatorname{sgn}(\beta_S^0), \operatorname{sgn}(\beta_S^0)\right) \\ (\hat{\beta}_N, \nu_N) &= (0, \Delta). \end{aligned}$$

This is by construction a solution. We then only need to check that

$$\operatorname{sgn}(\hat{\beta}_S) = \operatorname{sgn}(\beta_S^0),$$

which follows from the second condition. □

Prediction and estimation

We now consider other question of how good the Lasso functions as a regression method. Consider the model

$$Y = X\beta^0 + \varepsilon - \bar{\varepsilon} 1,$$

where the ε_i are independent and have common sub-Gaussian parameter σ. Let S, s, N be as before.

As before, the Lasso doesn't always behave well, and whether or not it does is controlled by the compatibility factor.

Definition (Compatibility factor)**.** Define the *compatibility factor* to be

$$\phi^2 = \inf_{\substack{\beta \in \mathbb{R}^p \\ \|\beta_N\|_1 \leq 3 \|\beta_S\|_1 \\ \beta_S \neq 0}} \frac{\frac{1}{n}\|X\beta\|_2^2}{\frac{1}{s}\|\beta_S\|_1^2} = \inf_{\substack{\beta \in \mathbb{R}^p \\ \|\beta_S\| = 1 \\ \|\beta_N\|_1 \leq 3}} \frac{s}{n} \|X_S \beta_S - X_N \beta_N\|_2^2.$$

Note that we are free to use a minus sign inside the norm since the problem is symmetric in $\beta_N \leftrightarrow -\beta_N$.

In some sense, this ϕ measures how well we can approximate $X_S \beta_S$ just with the noise variables.

Definition (Compatibility condition)**.** The *compatibility condition* is $\phi^2 > 0$.

Note that if $\hat{\Sigma} = \frac{1}{n}X^TX$ has minimum eigenvalue $c_{min} > 0$, then we have $\phi^2 \geq c_{min}$. Indeed,

$$\|\beta_S\|_1 = \operatorname{sgn}(\beta_S)^T\beta_S \leq \sqrt{s}\|\beta_S\|_2 \leq \sqrt{s}\|\beta\|_2,$$

and so

$$\phi^2 \geq \inf_{\beta \neq 0} \frac{\frac{1}{n}\|X\beta\|_2^2}{\|\beta\|_2^2} = c_{min}.$$

Of course, we don't expect the minimum eigenvalue to be positive, but we have the restriction in infimum in the definition of ϕ^2 and we can hope to have a positive value of ϕ^2.

Theorem. Assume $\phi^2 > 0$, and let $\hat{\beta}$ be the Lasso solution with

$$\lambda = A\sigma\sqrt{\log p/n}.$$

Then with probability at least $1 - 2p^{-(A^2/8-1)}$, we have

$$\frac{1}{n}\|X(\beta^0 - \hat{\beta})\|_2^2 + \lambda\|\hat{\beta} - \beta^0\|_1 \leq \frac{16\lambda^2 s}{\phi^2} = \frac{16A^2 \log p}{\phi^2}\frac{s\sigma^2}{n}.$$

This is actually two bounds. This simultaneously bounds the error in the fitted values, and a bound on the error in predicting $\hat{\beta} - \beta^0$.

Recall that in our previous bound, we had a bound of $\sim \frac{1}{\sqrt{n}}$, and now we have $\sim \frac{1}{n}$. Note also that $\frac{s\sigma^2}{n}$ is the error we would get if we magically knew which were the non-zero variables and did ordinary least squares on those variables.

This also tells us that if β^0 has a component that is large, then $\hat{\beta}$ must be non-zero in that component as well. So while the Lasso cannot predict exactly which variables are non-zero, we can at least get the important ones.

Proof. Start with the basic inequality $Q_\lambda(\hat{\beta}) \leq Q_\lambda(\beta^0)$, which gives us

$$\frac{1}{2n}\|X(\beta^0 - \hat{\beta})\|_2^2 + \lambda\|\hat{\beta}\|_1 \leq \frac{1}{n}\varepsilon^T X(\hat{\beta} - \beta^0) + \lambda\|\beta^0\|_1.$$

We work on the event

$$\Omega = \left\{\frac{1}{n}\|X^T\varepsilon\|_\infty \leq \frac{1}{2}\lambda\right\},$$

where after applying Hölder's inequality, we get

$$\frac{1}{n}\|X(\beta^0 - \hat{\beta})\|_2^2 + 2\lambda\|\hat{\beta}\|_1 \leq \lambda\|\hat{\beta} - \beta^0\|_1 + 2\lambda\|\beta^0\|_1.$$

We can move $2\lambda\|\hat{\beta}\|_1$ to the other side, and applying the triangle inequality, we have

$$\frac{1}{n}\|X(\hat{\beta} - \beta^0)\|_2^2 \leq 3\lambda\|\hat{\beta} - \beta^0\|.$$

If we manage to bound the RHS from above as well, so that

$$3\lambda\|\hat{\beta} - \beta^0\| \leq c\lambda\frac{1}{\sqrt{n}}\|X(\hat{\beta} - \beta^0)\|_2$$

for some c, then we obtain the bound

$$\frac{1}{n}\|X(\beta - \beta^0)\|_2^2 \leq c^2\lambda^2.$$

Plugging this back into the second bound, we also have

$$3\lambda\|\hat{\beta} - \beta^0\|_1 \leq c^2\lambda^2.$$

To obtain these bounds, we want to apply the definition of ϕ^2 to $\hat{\beta} - \beta^0$. We thus need to show that the $\hat{\beta} - \beta^0$ satisfies the conditions required in the infimum taken.

Write

$$a = \frac{1}{n\lambda}\|X(\hat{\beta} - \beta^0)\|_2^2.$$

Then we have

$$a + 2(\|\hat{\beta}_n\|_1 + \|\hat{\beta}_S\|_1) \leq \|\hat{\beta}_S - \beta_S^0\|_1 + \|\hat{\beta}_N\|_1 + 2\|\beta_S^0\|_1.$$

Simplifying, we obtain

$$a + \|\hat{\beta}_N\|_1 \leq \|\hat{\beta}_S - \beta_S^0\|_1 + 2\|\beta_S^0\|_1 - 2\|\hat{\beta}_S\|_1.$$

Using the triangle inequality, we write this as

$$a + \|\hat{\beta}_N - \beta^0\|_N \leq 3\|\hat{\beta}_S - \beta_S^0\|_1.$$

So we immediately know we can apply the compatibility condition, which gives us

$$\phi^2 \leq \frac{\frac{1}{n}\|X(\hat{\beta} - \beta^0)\|_2^2}{\frac{1}{s}\|\hat{\beta}_S - \beta_S^0\|_1^2}.$$

Also, we have

$$\frac{1}{n}\|X(\hat{\beta} - \beta^0)\|_2^2 + \lambda\|\hat{\beta} - \beta^0\|_1 \leq 4\lambda\|\hat{\beta}_S - \beta_S^0\|_1.$$

Thus, using the compatibility condition, we have

$$\frac{1}{n}\|X(\hat{\beta} - \beta^0)\|_2^2 + \lambda\|\hat{\beta} - \beta^0\| \leq \frac{4\lambda}{\phi}\sqrt{\frac{s}{n}}\|X(\hat{\beta} - \beta^0)\|_2.$$

Thus, dividing through by $\frac{1}{\sqrt{n}}\|X(\hat{\beta} - \beta^0)\|_2$, we obtain

$$\frac{1}{\sqrt{n}}\|X(\hat{\beta} - \beta^0)\|_2 \leq \frac{4\lambda\sqrt{s}}{\phi}. \tag{$*$}$$

So we substitute into the RHS $(*)$ and obtain

$$\frac{1}{n}\|X(\hat{\beta} - \beta^0)\|_2^2 + \lambda\|\hat{\beta} - \beta^0\|_1 \leq \frac{16\lambda^2 s}{\phi^2}. \qquad \square$$

If we want to be impressed by this result, then we should make sure that the compatibility condition is a reasonable assumption to make on the design matrix.

The compatibility condition and random design

For any $\Sigma \operatorname{im} \mathbb{R}^{p \times p}$, define

$$\phi_\Sigma^2(S) = \inf_{\beta : \|\beta_N\|_1 \leq 3\|\beta_S\|_1,\ \beta_s \neq 0} \frac{\beta^T \Sigma \beta}{\|\beta_S\|_1^2 / |S|}$$

Our original ϕ^2 is then the same as $\phi_{\hat{\Sigma}}^2(S)$.

If we want to analyze how $\phi_\Sigma^2(S)$ behaves for a "random" Σ, then it would be convenient to know that this depends continuously on Σ. For our purposes, the following lemma suffices:

Lemma. Let $\Theta, \Sigma \in \mathbb{R}^{p \times p}$. Suppose $\phi_\Theta^2(S) > 0$ and

$$\max_{j,k} |\Theta_{jk} - \Sigma_{jk}| \leq \frac{\phi_\Theta^2(S)}{32|S|}.$$

Then

$$\phi_\Sigma^2(S) \geq \frac{1}{2}\phi_\Theta^2(S).$$

Proof. We suppress the dependence on S for notational convenience. Let $s = |S|$ and

$$t = \frac{\phi_\Theta^2(S)}{32s}.$$

We have

$$|\beta^T(\Sigma - \Theta)\beta| \leq \|\beta\|_1 \|(\Sigma - \Theta)\beta\|_\infty \leq t\|\beta\|_1^2,$$

where we applied Hölder twice.

If $\|\beta_N\| \leq 3\|\beta_S\|_1$, then we have

$$\|\beta\|_1 \leq 4\|\beta_S\|_1 \leq 4\frac{\sqrt{\beta^T\Theta\beta}}{\phi_\Theta/\sqrt{s}}.$$

Thus, we have

$$\beta^T\Theta\beta - \frac{\phi_\Theta^2}{32s} \cdot \frac{16\beta^T\Theta\beta}{\phi_\Theta^2/s} = \frac{1}{2}\beta^T\Theta\beta \leq \beta^T\Sigma\beta. \qquad \square$$

Define

$$\phi_{\Sigma,s}^2 = \min_{S:|S|=s} \phi_\Sigma^2(S).$$

Note that if

$$\max_{jk} |\Theta_{jk} - \Sigma_{jk}| \leq \frac{\phi_{\Theta,s}^2}{32s},$$

then

$$\phi_\Sigma^2(S) \geq \frac{1}{2}\phi_\Theta^2(S).$$

for all S with $|S| = s$. In particular,

$$\phi_{\Sigma,s}^2 \geq \frac{1}{2}\phi_{\Theta,s}^2.$$

Theorem. Suppose the rows of X are iid and each entry is sub-Gaussian with parameter v. Suppose $s\sqrt{\log p/n} \to 0$ as $n \to \infty$, and $\phi^2_{\Sigma^0, s}$ is bounded away from 0. Then if $\Sigma^0 = \mathbb{E}\hat{\Sigma}$, then we have

$$\mathbb{P}\left(\phi^2_{\hat{\Sigma}, s} \geq \frac{1}{2}\phi^2_{\Sigma^0, s}\right) \to 1 \text{ as } n \to \infty.$$

Proof. It is enough to show that

$$\mathbb{P}\left(\max_{jk} |\hat{\Sigma}_{jk} - \Sigma^0_{jk}| \leq \frac{\phi^2_{\Sigma^0, s}}{32s}\right) \to 0$$

as $n \to \infty$.

Let $t = \frac{\phi^2_{\Sigma^0, s}}{32s}$. Then

$$\mathbb{P}\left(\max_{j,k} |\hat{\Sigma}_{jk} - \Sigma^0_{jk}| \geq t\right) \leq \sum_{j,k} \mathbb{P}(|\hat{\Sigma}_{jk} - \Sigma^0_{jk}| \geq t).$$

Recall that

$$\hat{\Sigma}_{jk} = \frac{1}{n}\sum_{i=1}^n X_{ij}X_{ik}.$$

So we can apply Bernstein's inequality to bound

$$\mathbb{P}(|\hat{\Sigma}_{jk} - \Sigma^0_{jk}) \leq 2\exp\left(-\frac{nt^2}{2(64v^4 + 4v^2 t)}\right),$$

since $\sigma = 8v^2$ and $b = 4v^2$. So we can bound

$$\mathbb{P}\left(\max_{j,k} |\hat{\Sigma}_{jk} - \Sigma^0_{jk}| \geq t\right) \leq 2p^2 \exp\left(-\frac{cn}{s^2}\right) = 2\exp\left(-\frac{cn}{s^2}\left(c - \frac{2s^2}{n\log p}\right)\right) \to 0$$

for some constant c. □

Corollary. Suppose the rows of X are iid mean-zero multivariate Gaussian with variance Σ^0. Suppose Σ^n has minimum eigenvalue bounded from below by $c_{min} > 0$, and suppose the diagonal entries of Σ^0 are bounded from above. If $s\sqrt{\log p/n} \to 0$, then

$$\mathbb{P}\left(\phi^2_{\hat{\Sigma}, s} \geq \frac{1}{2}c_{min}\right) \to 1 \text{ as } n \to \infty.$$

3.6 Computation of Lasso solutions

We have had enough of bounding things. In this section, let's think about how we can actually run the Lasso. What we present here is actually a rather general method to find the minimum of a function, known as *coordinate descent*.

Suppose we have a function $f : \mathbb{R}^d \to \mathbb{R}$. We start with an initial guess $x^{(0)}$ and repeat for $m = 1, 2, \ldots$

$$\begin{aligned}
x_1^{(m)} &= \operatorname*{argmin}_{x_1} f(x_1, x_2^{(m-1)}, \ldots, x_d^{(m-1)})\\
x_2^{(m)} &= \operatorname*{argmin}_{x_2} f(x_1^{(m)}, x_2, x_3^{(m-1)}, \ldots, x_d^{(m-1)})\\
&\vdots\\
x_d^{(m)} &= \operatorname*{argmin}_{x_d} f(x_1^{(m)}, x_2^{(m)}, \ldots, x_{d-1}^{(m)}, x_d)
\end{aligned}$$

until the result stabilizes.

This was proposed for solving the Lasso a long time ago, and a Stanford group tried this out. However, instead of using $x_1^{(m)}$ when computing $x_2^{(m)}$, they used $x_1^{(m-1)}$ instead. This turned out to be pretty useless, and so the group abandoned the method. After trying some other methods, which weren't very good, they decided to revisit this method and fixed their mistake.

For this to work well, of course the coordinatewise minimizations have to be easy (which is the case for the Lasso, where we even have explicit solutions). This converges to the global minimizer if the minimizer is unique, $\{x : f(x) \leq f(x^{(0)})\}$ is compact, and if f has the form

$$f(x) = g(x) + \sum_j h_j(x_j),$$

where g is convex and differentiable, and each h_j is convex but not necessarily differentiable. In the case of the Lasso, the first is the least squared term, and the h_j is the ℓ_1 term.

There are two things we can do to make this faster. We typically solve the Lasso on a grid of λ values $\lambda_0 > \lambda_1 > \cdots > \lambda_L$, and then picking the appropriate λ by v-fold cross-validation. In this case, we can start solving at λ_0, and then for each $i > 0$, we solve the $\lambda = \lambda_i$ problem by picking $x^{(0)}$ to be the optimal solution to the λ_{i-1} problem. In fact, even if we already have a fixed λ value we want to use, it is often advantageous to solve the Lasso with a larger λ-value, and then use that as a warm start to get to our desired λ value.

Another strategy is an *active set* strategy. If p is large, then this loop may take a very long time. Since we know the Lasso should set a lot of things to zero, for $\ell = 1, \ldots, L$, we set

$$A = \{k : \hat{\beta}^L_{\lambda_{\ell-1},k} \neq 0\}.$$

We then perform coordinate descent only on coordinates in A to obtain a Lasso solution $\hat{\beta}$ with $\hat{\beta}_{A^c} = 0$. This may not be the actual Lasso solution. To check this, we use the KKT conditions. We set

$$V = \left\{k \in A^c : \frac{1}{n}|X_k^T(Y - X\hat{\beta})| > \lambda_\ell\right\}.$$

If $V = \emptyset$, and we are done. Otherwise, we add V to our active set A, and then run coordinate descent again on this active set.

3.7 Extensions of the Lasso

There are many ways we can modify the Lasso. The first thing we might want to change in the Lasso is to replace the least squares loss with other log-likelihoods. Another way to modify the Lasso is to replace the ℓ_1 penalty with something else in order to encourage a different form of sparsity.

Example (Group Lasso)**.** Given a partition

$$G_1 \cup \cdots G_q = \{1, \ldots, p\},$$

the *group Lasso* penalty is

$$\lambda \sum_{j=1}^q m_j \|\beta_{G_j}\|_2,$$

where $\{m_j\}$ is some sort of weight to account for the fact that the groups have different sizes. Typically, we take $m_j = \sqrt{|G_j|}$.

If we take $G_i = \{i\}$, then we recover the original Lasso. If we take $q = 1$, then we recover Ridge regression. What this does is that it encourages the entire group to be all zero, or all non-zero.

Example. Another variation is the *fused Lasso*. If β^0_{j+1} is expected to be close to β^0_j, then a *fused Lasso* penalty may be appropriate, with

$$\lambda_1 \sum_{j=1}^{p-1} |\beta_{j+1} - \beta_j| + \lambda_2 \|\beta\|_1.$$

For example, if

$$Y_i = \mu_i + \varepsilon_i,$$

and we believe that $(\mu_i)_{i=1}^n$ form a piecewise constant sequence, we can estimate μ^0 by

$$\operatorname*{argmin}_{\mu} \left\{ \|Y - \mu\|_2^2 + \lambda \sum_{i=1}^{n-1} |\mu_{i+1} - \mu_i| \right\}.$$

Example (Elastic net)**.** We can use

$$\hat{\beta}^{EN}_{\lambda,\alpha} = \operatorname*{argmin}_{\beta} \left\{ \frac{1}{2n} \|Y - X\beta\|_2^2 + \lambda(\alpha\|\beta\|_1 + (1-\alpha)\|\beta\|_2^2) \right\}.$$

for $\alpha \in [0, 1]$. What the ℓ_2 norm does is that it encourages highly positively correlated variables to have similar estimated coefficients.

For example, if we have duplicate columns, then the ℓ_1 penalty encourages us to take one of the coefficients to be 0, while the ℓ_2 penalty encourages the coefficients to be the same.

Another class of variations try to reduce the bias of the Lasso. Although the bias of the Lasso is a necessary by-product of reducing the variance of the estimate, it is sometimes desirable to reduce this bias.

The *LARS-OLS hybrid* takes the $\hat{S}_\lambda$ obtained by the Lasso, and then re-estimate $\beta^0_{\hat{S}_\lambda}$ by OLS. We can also re-estimate using the Lasso on $X_{\hat{S}_\lambda}$, and this is known as the *relaxed Lasso*.

In the *adaptive Lasso*, we obtain an initial estimate of β^0, e.g. with the Lasso, and then solve

$$\hat{\beta}^{\mathrm{adapt}}_\lambda = \operatorname*{argmin}_{\beta : \beta_{\hat{S}} = 0} \left\{ \frac{1}{2n} \|Y - X\beta\|_2^2 + \lambda \sum_{k \in \hat{S}} \frac{|\beta_k|}{|\hat{\beta}_k|} \right\}.$$

We can also try to use a non-convex penalty. We can attempt to solve

$$\operatorname*{argmin}_{\beta} \left\{ \frac{1}{2n} \|Y - X\beta\|_2^2 + \sum_{k=1}^n p_\lambda(|\beta_k|) \right\},$$

where $p_\lambda : [0, \infty) \to p[0, \infty)$ is a non-convex function. One common example is the *MCP*, given by

$$p'_\lambda(u) = \left(\lambda - \frac{u}{\gamma} \right)_+,$$

where γ is an extra tuning parameter. This tends to give estimates even sparser than the Lasso.

4 Graphical modelling

4.1 Conditional independence graphs

So far, we have been looking at prediction problems. But sometimes we may want to know more than that. For example, there is a positive correlation between the wine budget of a college, and the percentage of students getting firsts. This information allows us to make predictions, in the sense that if we happen to know the wine budget of a college, but forgot the percentage of students getting firsts, then we can make a reasonable prediction of the latter based on the former. However, this does not suggest any causal relation between the two — increasing the wine budget is probably not a good way to increase the percentage of students getting firsts!

Of course, it is unlikely that we can actually figure out causation just by looking at the data. However, there are some things we can try to answer. If we gather more data about the colleges, then we would probably find that the colleges that have larger wine budget and more students getting firsts are also the colleges with larger endowment and longer history. If we condition on all these other variables, then there is not much correlation left between the wine budget and the percentage of students getting firsts. This is what we are trying to capture in conditional independence graphs.

We first introduce some basic graph terminology. For the purpose of conditional independence graphs, we only need undirected graphs. But later, we need the notion of direct graphs as well, so our definitions will be general enough to include those.

Definition (Graph)**.** A *graph* is a pair $\mathcal{G} = (V, E)$, where V is a set and $E \subseteq (V, V)$ such that $(v, v) \notin E$ for all $v \in V$.

Definition (Edge)**.** We say there is an *edge* between j and k and that j and k are *adjacent* if $(j, k) \in E$ or $(k, j) \in E$.

Definition (Undirected edge)**.** An edge (j, k) is *undirected* if also $(k, j) \in E$. Otherwise, it is *directed* and we write $j \to k$ to represent it. We also say that j is a *parent* of k, and write $\mathrm{pa}(k)$ for the set of all parents of k.

Definition ((Un)directed graph)**.** A graph is *(un)directed* if all its edges are (un)directed.

Definition (Skeleton)**.** The *skeleton* of $\mathcal{G}$ is a copy of $\mathcal{G}$ with every edge replaced by an undirected edge.

Definition (Subgraph)**.** A graph $\mathcal{G}_1 = (V, E)$ is a *subgraph* of $\mathcal{G} = (V, E)$ if $V_1 \subseteq V$ and $E_1 \subseteq E$. A *proper subgraph* is one where either of the inclusions are proper inclusions.

As discussed, we want a graph that encodes the conditional dependence of different variables. We first define what this means. In this section, we only work with undirected graphs.

Definition (Conditional independence)**.** Let X, Y, Z be random vectors with joint density f_{XYZ}. We say that X is *conditionally independent* of Y given Z, written $X \amalg Y \mid Z$, if

$$f_{XY\mid Z}(x, y \mid z) = f_{X\mid Z}(x \mid z) f_{Y\mid Z}(y \mid z).$$

Equivalently,

$$f_{X|YZ}(x \mid y, z) = f_{X|Z}(x \mid z)$$

for all y.

We shall ignore all the technical difficulties, and take as an assumption that all these conditional densities exist.

Definition (Conditional independence graph (CIG))**.** Let P be the law of $Z = (Z_1, \ldots, Z_p)^T$. The *conditional independent graph* (CIG) is the graph whose vertices are $\{1, \ldots, p\}$, and contains an edge between j and k iff Z_j and Z_k are conditionally dependent given Z_{-jk}.

More generally, we make the following definition:

Definition (Pairwise Markov property)**.** Let P be the law of $Z = (Z_1, \ldots, Z_p)^T$. We say P satisfies the *pairwise Markov property* with respect to a graph $\mathcal{G}$ if for any distinct, non-adjacent vertices j, k, we have $Z_j \amalg Z_k \mid Z_{-jk}$.

Example. If $\mathcal{G}$ is a complete graph, then P satisfies the pairwise Markov property with respect to $\mathcal{G}$.

The conditional independence graph is thus the minimal graph satisfying the pairwise Markov property. It turns out that under mild conditions, the pairwise Markov property implies something much stronger.

Definition (Separates)**.** Given a triple of (disjoint) subsets of nodes A, B, S, we say S *separates* A from B if every path from a node in A to a node in B contains a node in S.

Definition (Global Markov property)**.** We say P satisfies the *global Markov property* with respect to $\mathcal{G}$ if for any triple of disjoint subsets of V (A, B, S), if S separates A and B, then $Z_A \amalg Z_B \mid Z_S$.

Proposition. If P has a positive density, then if it satisfies the pairwise Markov property with respect to $\mathcal{G}$, then it also satisfies the global Markov property.

This is a really nice result, but we will not prove this. However, we will prove a special case in the example sheet.

So how do we actually construct the conditional independence graph? To do so, we need to test our variables for conditional dependence. In general, this is quite hard. However, in the case where we have Gaussian data, it is much easier, since independence is the same as vanishing covariance.

Notation ($M_{A,B}$)**.** Let M be a matrix. Then $M_{A,B}$ refers to the submatrix given by the rows in A and columns in B.

Since we are going to talk about conditional distributions a lot, the following calculation will be extremely useful.

Proposition. Suppose $Z \sim N_p(\mu, \Sigma)$ and Σ is positive definite. Then

$$Z_A \mid Z_B = z_B \sim N_{|A|}(\mu_A + \Sigma_{A,B}\Sigma_{B,B}^{-1}(z_B - \mu_B), \Sigma_{A,A} - \Sigma_{A,B}\Sigma_{B,B}^{-1}\Sigma_{B,A}).$$

Proof. Of course, we can just compute this directly, maybe using moment generating functions. But for pleasantness, we adopt a different approach. Note that for any M, we have

$$Z_A = MZ_B + (Z_A - MZ_B).$$

We shall pick M such that $Z_A - MZ_B$ is independent of Z_B, i.e. such that

$$0 = \mathrm{cov}(Z_B, Z_A - MZ_B) = \Sigma_{B,A} - \Sigma_{B,B}M^T.$$

So we should take

$$M = (\Sigma_{B,B}^{-1}\Sigma_{B,A})^T = \Sigma_{A,B}\Sigma_{B,B}^{-1}.$$

We already know that $Z_A - MZ_B$ is Gaussian, so to understand it, we only need to know its mean and variance. We have

$$\begin{aligned} \mathbb{E}[Z_A - MZ_B] &= \mu_A - M\mu_B = \mu_A - \Sigma_{AB}\Sigma_{BB}^{-1}\mu_B \\ \mathrm{var}(Z_A - MZ_B) &= \Sigma_{A,A} - 2\Sigma_{A,B}\Sigma_{B,B}^{-1}\Sigma_{B,A} + \Sigma_{A,B}\Sigma_{B,B}^{-1}\Sigma_{B,B}\Sigma_{B,B}^{-1}\Sigma_{B,A} \\ &= \Sigma_{A,A} - \Sigma_{A,B}\Sigma_{B,B}^{-1}\Sigma_{B,A}. \end{aligned}$$

Then we are done. □

Neighbourhood selection

We are going to specialize to $A = \{k\}$ and $B = \{1, \ldots, n\} \setminus \{k\}$. Then we can separate out the "mean" part and write

$$Z_k = M_k + Z_{-k}^T\Sigma_{-k,-k}^{-1}\Sigma_{-k,k} + \varepsilon_k,$$

where

$$\begin{aligned} M_k &= \mu_k - \mu_{-k}^T\Sigma_{k,-k}^{-1}\Sigma_{-k,k}, \\ \varepsilon_k \mid Z_{-k} &\sim N(0, \Sigma_{k,k} - \Sigma_{k,-k}\Sigma_{-k,-k}^{-1}\Sigma_{-k,k}). \end{aligned}$$

This now looks like we are doing regression.

We observe that

Lemma. Given k, let j' be such that $(Z_{-k})_j = Z_{j'}$. This j' is either j or $j+1$, depending on whether it comes after or before k.

If the jth component of $\Sigma_{-k,-k}^{-1}\Sigma_{-k,k}$ is 0, then $Z_k \amalg Z_{j'} \mid Z_{-kj'}$.

Proof. If the jth component of $\Sigma_{-k,-k}^{-1}\Sigma_{-k,k}$ is 0, then the distribution of $Z_k \mid Z_{-k}$ will not depend on $(Z_{-k})_j = Z_{j'}$ (here j' is either j or $j+1$, depending on where k is). So we know

$$Z_k \mid Z_{-k} \overset{d}{=} Z_k \mid Z_{-kj'}.$$

This is the same as saying $Z_k \amalg Z_{j'} \mid Z_{-kj'}$. □

Neighbourhood selection exploits this fact. Given $x_1, \ldots, x_n$ which are iid $\sim Z$ and

$$X = (x_1^T, \cdots, x_n^T)^T,$$

we can estimate $\Sigma_{-k,-k}^{-1}\Sigma_{-k,k}$ by regressing X_k on X_{-k} using the Lasso (with an intercept term). We then obtain selected sets $\hat{S}_k$. There are two ways of estimating the CIG based on these:

- OR rule: We add the edge (j,k) if $j \in \hat{S}_k$ or $k \in \hat{S}_j$.
- AND rule: We add the edge (j,k) if $j \in \hat{S}_k$ and $k \in \hat{S}_j$.

The graphical Lasso

Another way of finding the conditional independence graph is to compute $\operatorname{var}(Z_{jk} \mid Z_{-jk})$ directly. The following lemma will be useful:

Lemma. Let $M \in \mathbb{R}^{p\times p}$ be positive definite, and write

$$M = \begin{pmatrix} P & Q \\ Q^T & R \end{pmatrix},$$

where P and Q are square. The *Schur complement* of R is

$$S = P - QR^{-1}Q^T.$$

Note that this has the same size as P. Then

(i) S is positive definite.

(ii)
$$M^{-1} = \begin{pmatrix} S^{-1} & -S^{-1}QR^{-1} \\ -R^{-1}Q^TS^{-1} & R^{-1} + R^{-1}Q^TS^{-1}QR^{-1} \end{pmatrix}.$$

(iii) $\det(M) = \det(S)\det(R)$

We have seen this Schur complement when we looked at $\operatorname{var}(Z_A \mid Z_{A^c})$ previously, where we got

$$\operatorname{var}(Z_A \mid Z_{A^c}) = \Sigma_{A,A} - \Sigma_{A,A^C}\Sigma^{-1}_{A^c,A^c}\Sigma_{A^c,A} = \Omega^{-1}_{A,A},$$

where $\Omega = \Sigma^{-1}$ is the *precision matrix*.

Specializing to the case where $A = \{j,k\}$, we have

$$\operatorname{var}(Z_{\{j,k\}} \mid Z_{-jk}) = \frac{1}{\det(\Omega_{A,A})}\begin{pmatrix} \Omega_{k,k} & -\Omega_{j,k} \\ -\Omega_{j,k} & \Omega_{j,j}. \end{pmatrix}$$

This tells us $Z_k \amalg Z_j \mid Z_{-kj}$ iff $\Omega_{jk} = 0$. Thus, we can approximate the conditional independence graph by computing the precision matrix Ω.

Our method to estimate the precision matrix is similar to the Lasso. Recall that the density of $N_p(\mu, \Sigma)$ is

$$P(z) = \frac{1}{(2\pi)^{p/2}(\det\Sigma)^{1/2}}\exp\left(-\frac{1}{2}(z-\mu)^T\Sigma^{-1}(z-\mu)\right).$$

The log-likelihood of (μ, Ω) based on an iid sample $(X_1, \ldots, X_n)$ is (after dropping a constant)

$$\ell(\mu,\Omega) = \frac{n}{2}\log\det\Omega - \frac{1}{2}\sum_{i=1}^n (x_i - \mu)^T\Omega(x_i - \mu).$$

To simplify this, we let

$$\bar{x} = \frac{1}{n}\sum_{i=1}^n x_i, \quad S = \frac{1}{n}\sum_{i=1}^n (x_i - \bar{x})(x_i - \bar{x})^T.$$

Then

$$\begin{aligned}\sum (x_i - \mu)^T \Omega (x_i - \mu) &= \sum (x_i - \bar{x} + \bar{x} - \mu)^T \Omega (x_i - \bar{x} + \bar{x} - \mu)\\ &= \sum (x_i - \bar{x})^T \Omega (x_i - \bar{x}) + n(\bar{X} - \mu)^T \Omega (\bar{X} - \mu).\end{aligned}$$

We have

$$\sum (x_i - \bar{x})^T \Omega (x_i - \bar{x}) = \sum \operatorname{tr}\Big((x_i - \bar{x})^T \Omega (x_i - \bar{x})\Big) = n \operatorname{tr}(S\Omega).$$

So we now have

$$\ell(\mu, \Omega) = -\frac{n}{2}\Big(\operatorname{tr}(S\Omega) - \log\det\Omega + (\bar{X} - \mu)^T \Omega (\bar{X} - \mu)\Big).$$

We are really interested in estimating Ω. So we should try to maximize this over μ, but that is easy, since this is the same as minimizing $(\bar{X} - \mu)^T \Omega (\bar{X} - \mu)$, and we know Ω is positive-definite. So we should set $\mu = \bar{X}$. Thus, we have

$$\ell(\Omega) = \max_{\mu \in \mathbb{R}^p} \ell(\mu, \Omega) = -\frac{n}{2}\Big(\operatorname{tr}(S\Omega) - \log\det\omega\Big).$$

So we can solve for the MLE of Ω by solving

$$\min_{\Omega : \Omega \succ 0}\Big(-\log\det\Omega + \operatorname{tr}(S\Omega)\Big).$$

One can show that this is convex, and to find the MLE, we can just differentiate

$$\frac{\partial}{\partial \Omega_{jk}} \log\det\Omega = (\Omega^{-1})_{jk}, \quad \frac{\partial}{\partial \Omega_{jk}} \operatorname{tr}(S\Omega) = S_{jk},$$

using that S and Ω are symmetric. So provided that S is positive definite, the maximum likelihood estimate is just

$$\Omega = S^{-1}.$$

But we are interested in the high dimensional situation, where we have loads of variables, and S cannot be positive definite. To solve this, we use the *graphical Lasso*.

The *graphical Lasso* solves

$$\operatorname*{argmin}_{\Omega : \Omega \succ 0}\Big(-\log\det\Omega + \operatorname{tr}(S\Omega) + \lambda\|\Omega\|_1\Big),$$

where

$$\|\Omega\|_1 = \sum_{jk} \Omega_{jk}.$$

Often, people don't sum over the diagonal elements, as we want to know if off-diagonal elements ought to be zero. Similar to the case of the Lasso, this gives a sparse estimate of Ω from which we may estimate the conditional independence graph.

4.2 Structural equation modelling

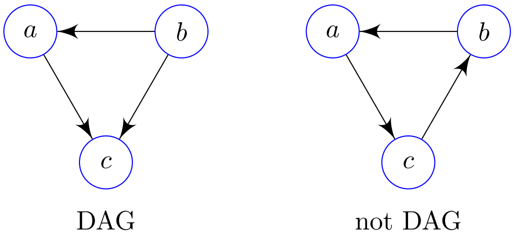
The conditional independence graph only tells us which variables are related to one another. However, it doesn't tell us any causal relation between the different variables. We first need to explain what we mean by a causal model. For this, we need the notion of a *directed acyclic graph*.

Definition (Path)**.** A *path* from j to k is a sequence $j = j_1, j_2, \ldots, j_m = k$ of (at least two) distinct vertices such that j_ℓ and $j_{\ell+1}$ are adjacent.

A path is *directed* if $j_\ell \to j_{\ell+1}$ for all ℓ.

Definition (Directed acyclic graph (DAG))**.** A *directed cycle* is (almost) a directed path but with the start and end points the same.

A *directed acyclic graph (DAG)* is a directed graph containing no directed cycles.



We will use directed acyclic graphs to encode causal structures, where we have a path from a to b if a "affects" b.

Definition (Structural equation model (SEM))**.** A *structural equation model* $\mathcal{S}$ for a random vector $Z \in \mathbb{R}^p$ is a collection of equations

$$Z_k = h_k(Z_{p_k}, \varepsilon_k),$$

where $k = 1, \ldots, p$ and $\varepsilon_1, \ldots, \varepsilon_p$ are independent, and $p_k \subseteq \{1, \ldots, p\} \setminus \{k\}$ and such that the graph with $pa(k) = p_k$ is a directed acyclic graph.

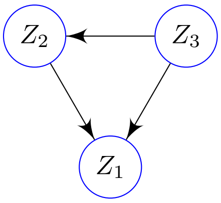
Example. Consider three random variables:

- $Z_1 = 1$ if a student is taking a course, 0 otherwise
- $Z_2 = 1$ if a student is attending catch up lectures, 0 otherwise
- $Z_3 = 1$ if a student heard about machine learning before attending the course, 0 otherwise.

Suppose

$$\begin{aligned} Z_3 &= \varepsilon_3 \sim \text{Bernoulli}(0.25)\\ Z_2 &= \mathbf{1}_{\{\varepsilon_2(1+Z_3) > \frac{1}{2}\}}, \quad \varepsilon_2 \sim U[0,1]\\ Z_1 &= \mathbf{1}_{\{\varepsilon_1(Z_2+Z_3) > \frac{1}{2}\}}, \quad \varepsilon_1 \sim U[0,1]. \end{aligned}$$

This is then an example of a structural equation modelling, and the corresponding DAG is



Note that the structural equation model for Z determines its distribution, but the converse is not true. For example, the following two distinct structural equation give rise to the same distribution:

$$\begin{aligned} Z_1 &= \varepsilon & \qquad Z_1 &= Z_2 \\ Z_2 &= Z_1 & \qquad Z_2 &= \varepsilon \end{aligned}$$

Indeed, if we have two variables that are just always the same, it is hard to tell which is the cause and which is the effect.

It would be convenient if we could order our variables in a way that Z_k depends only on Z_j for $j < k$. This is known as a *topological ordering*:

Definition (Descendant)**.** We say k is a *descendant* of j if there is a directed path from j to k. The set of descendant of j will be denoted $\mathrm{de}(j)$.

Definition (Topological ordering)**.** Given a DAG $\mathcal{G}$ with $V = \{1, \ldots, p\}$ we say that a permutation $\pi : V \to V$ is a *topological ordering* if $\pi(j) < \pi(k)$ whenever $k \in \mathrm{de}(j)$.

Thus, given a topological ordering π, we can write Z_k as a function of $\varepsilon_{\pi^{-1}(1)}, \ldots, \varepsilon_{\pi^{-1}(\pi(k))}$.

How do we understand structural equational models? They give us information that are not encoded in the distribution itself. One way to think about them is via *interventions*. We can modify a structural equation model by replacing the equation for Z_k by setting, e.g. $Z_k = a$. In real life, this may correspond to forcing all students to go to catch up workshops. This is called a *perfect intervention*. The modified SEM gives a new joint distribution for Z. Expectations or probabilities with respect to the new distribution are written by adding "$\mathsf{do}(Z_k = a)$". For example, we write

$$\mathbb{E}(Z_j \mid \mathsf{do}(Z_k = a)).$$

In general, this is different to $\mathbb{E}(Z_j \mid Z_k = a)$, since, for example, if we conditioned on $Z_2 = a$ in our example, then that would tell us something about Z_3.

Example. After the intervention $\mathsf{do}(Z_2 = 1)$, i.e. we force everyone to go to the catch-up lectures, we have a new SEM with

$$\begin{aligned} Z_3 &= \varepsilon_3 \sim \mathrm{Bernoulli}(0.25) \\ Z_2 &= 1 \\ Z_1 &= \mathbf{1}_{\varepsilon_1(1+Z_3)>\frac{1}{2}}, \quad \varepsilon_1 \sim U[0,1]. \end{aligned}$$

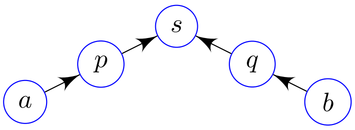
Then, for example, we can compute

$$\mathbb{P}(Z_1 = 1 \mid \mathsf{do}(Z_2 = 1)) = \frac{1}{2} \cdot \frac{3}{4} + \frac{3}{4} + \frac{1}{4} = \frac{9}{16},$$

and by high school probability, we also have

$$\mathbb{P}(Z_1 = 1 \mid Z_2 = 1) = \frac{7}{12}.$$

To understand the DAGs associated to structural equation models, we would like to come up with Markov properties analogous to what we had for undirected graphs. This, in particular, requires the correct notion of "separation", which we call d-separation. Our notion should be such that if S d-separates A and B in the DAG, then Z_A and Z_B are conditionally independent given Z_S. Let's think about some examples. For example, we might have a DAG that looks like this:



Then we expect that

(i) Z_a and Z_s are not independent;

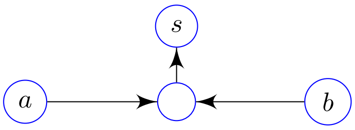
(ii) Z_a and Z_s are independent given Z_q;

(iii) Z_a and Z_b are independent;

(iv) Z_a and Z_b are not independent given Z_s.

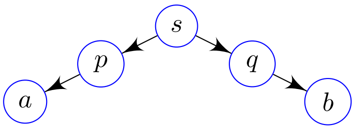
We can explain a bit more about the last one. For example, the structural equation model might telling us $Z_s = Z_a + Z_b + \varepsilon$. In this case, if we know that Z_a is large but Z_s is small, then chances are, Z_b is also large (in the opposite sign). The point is that both Z_a and Z_b both contribute to Z_s, and if we know one of the contributions and the result, we can say something about the other contribution as well.

Similarly, if we have a DAG that looks like



then as above, we know that Z_a and Z_b are not independent given Z_s.

Another example is

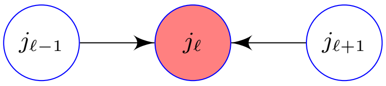


Here we expect

- Z_a and Z_b are not independent.
- Z_a and Z_b are independent given Z_s.

To see (i), we observe that if we know about Z_a, then this allows us to predict some information about Z_s, which would in turn let us say something about Z_b.

Definition (Blocked)**.** In a DAG, we say a path $(j_1, \ldots, j_m)$ between j_1 and j_m is *blocked* by a set of nodes S (with neither j_1 nor j_m in S) if there is some $j_\ell \in S$ and the path is *not* of the form



or there is some j_ℓ such that the path *is* of this form, but neither j_ℓ nor any of its descendants are in S.

Definition (d-separate)**.** If $\mathcal{G}$ is a DAG, given a triple of (disjoint) subsets of nodes A, B, S, we say S *d-separates* A from B if S blocks every path from A to B.

For convenience, we define

Definition (v-structure)**.** A set of three nodes is called a *v-structure* if one node is a child of the two other nodes, and these two nodes are not adjacent.

It is now natural to define

Definition (Markov properties)**.** Let P be the distribution of Z and let f be the density. Given a DAG $\mathcal{G}$, we say P satisfies

(i) the *Markov factorization criterion* if

$$f(z_1, \ldots, z_p) = \prod_{k=1}^{p} f(z_k \mid z_{\mathrm{pa}(k)}).$$

(ii) the *global Markov property* if for all disjoint A, B, S such that A, B is d-separated by S, then $Z_A \amalg Z_B \mid Z_S$.

Proposition. If P has a density with respect to a product measure, then (i) and (ii) are equivalent.

How does this fit into the structural equation model?

Proposition. Let P be the structural equation model with DAG $\mathcal{G}$. Then P obeys the Markov factorization property.

Proof. We assume $\mathcal{G}$ is topologically ordered (i.e. the identity map is a topological ordering). Then we can always write

$$f(z_1, \ldots, z_p) = f(z_1) f(z_2 \mid z_1) \cdots z(z_p \mid z_1, z_2, \ldots, z_{p-1}).$$

By definition of a topological order, we know $\mathrm{pa}(k) \subseteq \{1, \ldots, k-1\}$. Since Z_k is a function of $Z_{\mathrm{pa}(k)}$ and independent noise ε_k. So we know

$$Z_k \amalg Z_{\{1,\ldots,p\} \setminus \{k \cup \mathrm{pa}(k)\}} \mid Z_{\mathrm{pa}(k)}.$$

Thus,

$$f(z_k \mid z_1, \ldots, z_{k-1}) = f(z_k \mid z_{\mathrm{pa}(k)}).$$ □

4.3 The PC algorithm

We now want to try to find out the structural equation model given some data, and in particular, determine the causal structure. As we previously saw, there is no hope of determining this completely, even if we know the distribution of the Z completely. Let's consider the different obstacles to this problem.

Causal minimality

If P is generated by an SEM with DAG $\mathcal{G}$, then from the above, we know that P is Markov with respect to $\mathcal{G}$. The converse is also true: if P is Markov with respect to a DAG $\mathcal{G}$, then there exists a SEM with DAG $\mathcal{G}$ that generates P. This immediately implies that P will be Markov with respect to many DAGs. For example, a DAG whose skeleton is complete will always work. This suggests the following definition:

Definition (Causal minimality)**.** A distribution P satisfies *causal minimality* with respect to $\mathcal{G}$ but not any proper subgraph of $\mathcal{G}$.

Markov equivalent DAGs

It is natural to aim for finding a causally minimal DAG. However, this does not give a unique solution, as we saw previously with the two variables that are always the same.

In general, two different DAGs may satisfy the same set of d-separations, and then a distribution is Markov with respect to one iff its Markov with respect to the other, and we cannot distinguish between the two.

Definition (Markov equivalence)**.** For a DAG $\mathcal{G}$, we let

$$\mathcal{M}(\mathcal{G}) = \{\text{distributions } P \text{ such that } P \text{ is Markov with respect to } \mathcal{G}\}.$$

We say two DAGs $\mathcal{G}_1, \mathcal{G}_2$ are are *Markov equivalent* if $\mathcal{M}(\mathcal{G}_1) = \mathcal{M}(\mathcal{G}_2)$.

What is nice is that there is a rather easy way of determining when two DAGs are Markov equivalent.

Proposition. Two DAGs are Markov equivalent iff they have the same skeleton and same set of v-structure.

The set of all DAGs that are Markov equivalent to a given DAG can be represented by a *CPDAG* (completed partial DAG), which contains an edge (j, k) iff some member of the equivalence class does.

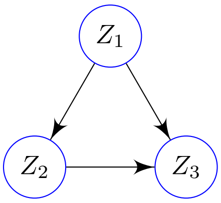
Faithfulness

To describe the final issue, consider the SEM

$$Z_1 = \varepsilon_1, \quad Z_2 = \alpha Z_1 + \varepsilon_2, \quad Z_3 = \beta Z_1 + \gamma Z_2 + \varepsilon_3.$$

We take $\varepsilon \sim N_3(0, I)$. Then we have $Z = (Z_1, Z_2, Z_3) \sim N(0, \Sigma)$, where

$$\Sigma = \begin{pmatrix} 1 & \alpha & \beta + \alpha\gamma \\ \alpha & \alpha^2 + 1 & \alpha(\beta + \alpha\gamma) + \gamma \\ \beta + \alpha\gamma & \alpha(\beta + \alpha\gamma) + \gamma & \beta^2 + \gamma^2(\alpha^2 + 1) + 1 + 2\alpha\beta\gamma \end{pmatrix}.$$



Now if we picked values of α, β, γ such that

$$\beta + \alpha\gamma = 0,$$

then we obtained an extra independence relation $Z_1 \amalg Z_3$ in our system. For example, if we pick $\beta = -1$ and $\alpha, \gamma = 1$, then

$$\Sigma = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix}.$$

While there is an extra independence relation, we cannot remove any edge while still satisfying the Markov property. Indeed:

- If we remove 1 → 2, then this would require $Z_1 \amalg Z_2$, but this is not true.
- If we remove 2 → 3, then this would require $Z_2 \amalg Z_3 \mid Z_1$, but we have

  $$\mathrm{var}((Z_2, Z_3) \mid Z_1) = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix},$$

  and this is not diagonal.
- If we remove 1 → 3, then this would require $Z_1 \amalg Z_3 \mid Z_2$, but

  $$\mathrm{var}((Z_1, Z_3) \mid Z_2) = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix} - \frac{1}{2} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \end{pmatrix},$$

  which is again non-diagonal.

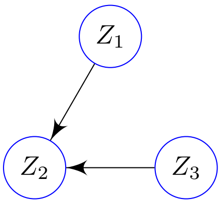
So this DAG satisfies causal minimality. However, P can also be generated by the structural equation model

$$\tilde{Z}_1 = \tilde{\varepsilon}_1, \quad \tilde{Z}_2 = \tilde{Z}_1 + \frac{1}{2}\tilde{Z}_3 + \tilde{\varepsilon}_2, \quad \tilde{Z}_3 = \tilde{\varepsilon}_3,$$

where the $\tilde{\varepsilon}_i$ are independent with

$$\tilde{\varepsilon}_1 \sim N(0,1), \quad \tilde{\varepsilon}_2 \sim N(0,2), \quad \tilde{\varepsilon}_3 \sim N(0, \tfrac{1}{2}).$$

Then this has the DAG



This is a strictly smaller DAG in terms of the number of edges involved. It is easy to see that this satisfies causal minimality.

Definition (Faithfulness)**.** We say P is *faithful* to a DAG $\mathcal{G}$ if it is Markov with respect to $\mathcal{G}$ and for all A, B, S disjoint, $Z_A \amalg Z_B \mid Z_S$ implies A, B are d-separated by S.

Determining the DAG

We shall assume our distribution is faithful to some $\mathcal{G}_0$, and see if we can figure out $\mathcal{G}_0$ from P, or even better, from data.

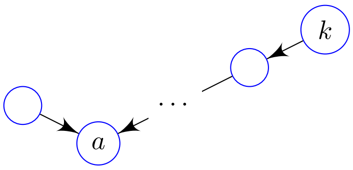
To find $\mathcal{G}$, the following proposition helps us compute the skeleton:

Proposition. If nodes j and k are adjacent in a DAG $\mathcal{G}$, then no set can d-separate them.

If they are not adjacent, and π is a topological order for $\mathcal{G}$ with $\pi(j) < \pi(k)$, then they are d-separated by $\mathrm{pa}(k)$.

Proof. Only the last part requires proof. Consider a path $j = j_1, \ldots, j_m = k$. Start reading the path from k and go backwards. If it starts as $j_{m-1} \to k$, then j_{m-1} is a parent of k and blocks the path. Otherwise, it looks like $k \to j_{m-1}$.

We keep going down along the path until we first see something of the form



Thus must exist, since j is not a descendant of k by topological ordering. So it suffices to show that a does not have a descendant in $\mathrm{pa}(k)$, but if it did, then this would form a closed loop. □

Finding the v-structures is harder, and at best we can do so up to Markov equivalence. To do that, observe the following:

Proposition. Suppose we have $j - \ell - k$ in the skeleton of a DAG.

(i) If $j \to \ell \leftarrow k$, then no S that d-separates j can have $\ell \in S$.

(ii) If there exists S that d-separates j and k and $\ell \not\in S$, then $j \to \ell \leftarrow k$. □

Denote the set of nodes adjacent to the vertex k in the graph $\mathcal{G}$ by $\mathrm{adj}(\mathcal{G}, k)$.

We can now describe the first part of the *PC algorithm*, which finds the skeleton of the "true DAG":

(i) Set $\hat{\mathcal{G}}$ to be the complete undirected graph. Set $\ell = -1$.

(ii) Repeat the following steps:

- (a) Set $\ell = \ell + 1$:
- (b) Repeat the following steps:
    - i. Select a new ordered pair of nodes j, k that are adjacent in $\hat{\mathcal{G}}$ and such that $|\mathrm{adj}(\hat{\mathcal{G}}, j) \setminus \{k\}| \geq \ell$.
    - ii. Repeat the following steps:
        - A. Choose a new $S \subseteq \mathrm{adj}(\hat{\mathcal{G}}, j) \setminus \{k\}$ with $|S| = \ell$.
        - B. If $Z_j \perp\!\!\!\perp Z_k \mid Z_S$, then delete the edge jk, and store $S(k, j) = S(j, k) = S$
        - C. Repeat until $j - k$ is deleted or all S chosen.
    - iii. Repeat until all pairs of adjacent nodes are inspected.

(c) Repeat until $\ell \geq p - 2$.

Suppose P is faithful to a DAG $\mathcal{G}^0$. At each stage of the algorithm, the skeleton of $\mathcal{G}^0$ will be a subgraph of $\hat{\mathcal{G}}$. On the other hand, edges (j, k) remaining at termination will have

$$Z_j \amalg Z_k \mid Z_S \text{ for all } S \subseteq (\hat{\mathcal{G}}, k), \quad S \subseteq (\hat{\mathcal{G}}, j).$$

So they must be adjacent in $\mathcal{G}^0$. Thus, $\hat{\mathcal{G}}$ and $\mathcal{G}_0$ have the same skeleton.

To find the v-structures, we perform:

(i) For all $j - l - k$ in $\hat{\mathcal{G}}$, do:

(a) If $\ell \notin S(j, k)$, then orient $j \to \ell \leftarrow k$.

This gives us the Markov equivalence class, and we may orient the other edges using other properties like acyclicity.

If we want to apply this to data sets, then we need to apply some conditional independence tests instead of querying our oracle to figure out if things are conditional dependence. However, errors in the algorithm propagate, and the whole process may be derailed by early errors. Moreover, the result of the algorithm may depend on how we iterate through the nodes. People have tried many ways to fix these problems, but in general, this method is rather unstable. Yet, if we have large data sets, this can produce quite decent results.

5 High-dimensional inference

5.1 Multiple testing

Finally, we talk about high-dimensional inference. Suppose we have come up with a large number of potential drugs, and want to see if they are effective in killing bacteria. Naively, we might try to run a hypothesis test on each of them, using a $p < 0.05$ threshold. But this is a terrible idea, since each test has a 0.05 chance of giving a false positive, so even if all the drugs are actually useless, we would have incorrectly believed that a lot of them are useful, which is not the case.

In general, suppose we have some null hypothesis $H_1, \ldots, H_m$. By definition, a p value p_i for H_i is a random variable such that

$$\mathbb{P}_{H_i}(p_i \leq \alpha) \leq \alpha$$

for all $\alpha \in [0, 1]$.

Let $m_0 = |I_0|$ be the number of true null hypothesis. Given a procedure for rejecting hypothesis (a *multiple testing procedure*), we let N be the number of false rejections (false positives), and R the total number of rejections. One can also think about the number of false negatives, but we shall not do that here.

Traditionally, multiple-testing procedures sought to control the *family-wise error rate* (*FWER*), defined by $\mathbb{P}(N \geq 1)$. The simplest way to minimize this is to use the *Bonferroni correction*, which rejects H_i if $p_i \leq \frac{\alpha}{m}$. Usually, we might have $\alpha \sim 0.05$, and so this would be very small if we have lots of hypothesis (e.g. a million). Unsurprisingly, we have

Theorem. When using the Bonferroni correction, we have

$$\mathrm{FWER} \leq \mathbb{E}(N) \leq \frac{m_0 \alpha}{m} \leq \alpha.$$

Proof. The first inequality is Markov's inequality, and the last is obvious. The second follows from

$$\mathbb{E}(N) = \mathbb{E}\left(\sum_{i \in I_0} \mathbf{1}_{p_i \leq \alpha/m}\right) = \sum_{i \in I_0} \mathbb{P}\left(p_i \leq \frac{\alpha}{m}\right) \leq \frac{m_0 \alpha}{m}. \qquad \square$$

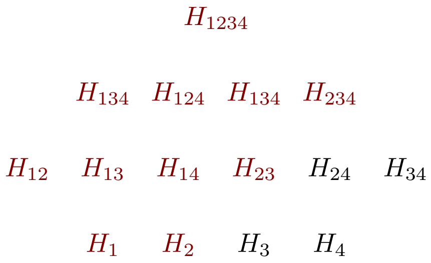
The Bonferroni is a rather conservative procedure, since all these inequalities can be quite loose. When we have a large number of hypotheses, the criterion for rejection is very very strict. Can we do better?

A more sophisticated approach is the *closed testing procedure*. For each non-empty subset $I \subseteq \{1, \ldots, m\}$, we let H_I be the null hypothesis that H_i is true for all $i \in I$. This is known as an *intersection hypothesis*. Suppose for each $I \subseteq \{1, \ldots, m\}$ non-empty, we have an α-level test ϕ_I for H_I (a *local test*), so that

$$\mathbb{P}_{H_I}(\phi_I = 1) \leq \alpha.$$

Here Φ_I takes values in $\{0, 1\}$, and $\phi_I = 1$ means rejection. The closed testing procedure then rejects H_I iff for all $J \supseteq I$, we have $\phi_J = 1$.

Example. Consider the tests, where the red ones are the rejected one:



In this case, we reject H_1 but not H_2 by closed testing. While H_{23} is rejected, we cannot tell if it is H_2 or H_3 that should be rejected.

This might seem like a very difficult procedure to analyze, but it turns out it is extremely simple.

Theorem. Closed testing makes no false rejections with probability $\geq 1 - \alpha$. In particular, FWER $\leq \alpha$.

Proof. In order for there to be a false rejection, we must have falsely rejected H_{I_0} with the local test, which has probability at most α. □

But of course this doesn't immediately give us an algorithm we can apply to data. Different choices for the local test give rise to different multiple testing procedures. One famous example is *Holm's procedure*. This takes ϕ_I to be the Bonferroni test, where $\phi_I = 1$ iff $p_i \leq \frac{\alpha}{|I|}$ for some $i \in I$.

When m is large, then we don't want to compute all ϕ_I, since there are 2^I computations to do. So we might want to find a shortcut. With a moment of thought, we see that Holm's procedure amounts to the following:

- Let (i) be the index of the ith smallest p-value, so $p_{(1)} \leq p_{(2)} \leq \cdots \leq p_{(m)}$.
- Step 1: If $p_{(1)} \leq \frac{\alpha}{m}$, then reject $H_{(1)}$ and go to step 2. Otherwise, accept all null hypothesis.
- Step i: If $p_{(i)} \leq \frac{\alpha}{m-i+1}$, then reject $H_{(i)}$ and go to step $i+1$. Otherwise, accept $H_{(i)}, H_{(i+1)}, \ldots, H_{(m)}$.
- Step m: If $p_{(m)} \leq \alpha$, then reject $H_{(m)}$. Otherwise, accept $H_{(m)}$.

The interesting thing about this is that it has the same bound on FWER as the Bonferroni correction, but the conditions here are less lenient.

But if m is very large, then the criterion for accepting $p_{(1)}$ is still quite strict. The problem is that controlling FWER is a very strong condition. Instead of controlling the probability that there is one false rejection, when m is large, it might be more reasonable to control the proportion of false discoveries. Many modern multiple testing procedures aim to control the *false discovery rate*

$$\mathrm{FDR} = \mathbb{E}\left(\frac{N}{\max\{R, 1\}}\right).$$

The funny maximum in the denominator is just to avoid division by zero. When $R = 0$, then we must have $N = 0$ as well, so what is put in the denominator doesn't really matter.

The *Benjamini–Hochberg procedure* attempts to control the FDR at level α and works as follows:

- Let $\hat{k} = \max\left\{i : p_{(i)} \leq \frac{\alpha i}{m}\right\}$. Then reject $M_{(1)}, \ldots, H_{(\hat{k})}$, or accept all hypothesis if $\hat{k}$ is not defined.

Under certain conditions, this does control the false discovery rate.

Theorem. Suppose that for each $i \in I_0$, p_i is independent of $\{p_j : j \neq i\}$. Then using the Benjamini–Hochberg procedure, the false discovery rate

$$FDR = \mathbb{E}\frac{N}{\max(R,1)} \leq \frac{\alpha M_0}{M} \leq \alpha.$$

Curiously, while the proof requires p_i to be independent of the others, in simulations, even when there is no hope that the p_i are independent, it appears that the Benjamini–Hochberg still works very well, and people are still trying to understand what it is that makes Benjamini–Hochberg work so well in general.

Proof. The false discovery rate is

$$\begin{aligned}\mathbb{E}\frac{N}{\max(R,1)} &= \sum_{r=1}^{M} \mathbb{E}\frac{N}{r}\mathbf{1}_{R=r}\\ &= \sum_{r=1}^{m} \frac{1}{r}\mathbb{E}\sum_{i \in I_0} \mathbf{1}_{p_i \leq \alpha r/M}\mathbf{1}_{R=r}\\ &= \sum_{i \in I_0}\sum_{r=1}^{M} \frac{1}{r}\mathbb{P}\left(p_i \leq \frac{\alpha r}{m}, R = r\right).\end{aligned}$$

The brilliant idea is, for each $i \in I_0$, let R_i be the number of rejections when applying a modified Benjamini–Hochberg procedure to $p^{\setminus i} = \{p_1, \ldots, p_M\} \setminus \{p_i\}$ with cutoff

$$\hat{k}_i = \max\left\{j : p_{(j)}^{\setminus i} \leq \frac{\alpha(j+1)}{m}\right\}$$

We observe that for $i \in I_0$ and $r \geq 1$, we have

$$\begin{aligned}\left\{p_i \leq \frac{\alpha r}{m},\, R = r\right\} &= \left\{p_i \leq \frac{ar}{m},\, p_{(r)} \leq \frac{\alpha r}{m},\, p_{(s)} > \frac{\alpha s}{m} \text{ for all } s \geq r\right\}\\ &= \left\{p_i \leq \frac{\alpha r}{m},\, p_{(r-1)}^{\setminus i} \leq \frac{\alpha r}{m},\, p_{(s-1)}^{\setminus i} > \frac{\alpha s}{m} \text{ for all } s > r\right\}\\ &= \left\{p_i \leq \frac{\alpha r}{m},\, R_i = r-1\right\}.\end{aligned}$$

The key point is that $R_i = r - 1$ depends only on the other p-values. So the FDR is equal to

$$\begin{aligned}\text{FDR} &= \sum_{i \in I_0}\sum_{r=1}^{M} \frac{1}{r}\mathbb{P}\left(p_i \leq \frac{\alpha r}{M}, R_i = r-1\right)\\ &= \sum_{i \in I_0}\sum_{r=1}^{M} \frac{1}{r}\mathbb{P}\left(p_i \leq \frac{\alpha r}{m}\right)\mathbb{P}(R_i = r-1)\end{aligned}$$

Using that $\mathbb{P}(p_i \leq \frac{\alpha r}{m}) \leq \frac{\alpha r}{m}$ by definition, this is

$$\begin{aligned}&\leq \frac{\alpha}{M} \sum_{i \in I_0} \sum_{r=1}^{m} \mathbb{P}(R_i = r - 1)\\&= \frac{\alpha}{M} \sum_{i \in I_0} \mathbb{P}(R_i \in \{0, \ldots, m-1\})\\&= \frac{\alpha M_0}{M}.\end{aligned}$$ □

This is one of the most used procedures in modern statistics, especially in the biological sciences.

5.2 Inference in high-dimensional regression

We have more-or-less some answer as to how to do hypothesis testing, given that we know how to obtain these p-values. But how do we obtain these in the first place?

For example, we might be trying to do regression, and are trying figure out which coefficients are non-zero. The the low dimension setting, with the normal linear model $Y = X\beta^0 + \varepsilon$, where $\varepsilon \sim N_n(0, \sigma^2 I)$. In the low-dimensional setting, we have $\sqrt{n}(\hat{\beta}^{OLS} - \beta^0) \sim N_p(0, \sigma^2(\frac{1}{n}X^T X)^{-1})$. Since this does not depend on β^0, we can use this to form confidence intervals and hypothesis tests.

However, if we have more coefficients than there are data points, then we can't do ordinary least squares. So we need to look for something else. For example, we might want to replace the OLS estimate with the Lasso estimate. However, $\sqrt{n}(\hat{\beta}^L_\lambda - \beta^0)$ has an intractable distribution. In particular, since $\hat{\beta}^L_\lambda$ has a bias, the distribution will depend on β^0 in a complicated way.

The recently introduced *debiased Lasso* tries to overcomes these issues. See *van de Geer, Bühlmann, Ritov, Dezeure* (2014) for more details. Let $\hat{\beta}$ be the Lasso solution at $\lambda > 0$. Recall the KKT conditions that says $\hat{\nu}$ defined by

$$\frac{1}{n} X^T (Y - X\hat{\beta}) = \lambda \hat{\nu}$$

satisfies $\|\hat{\nu}\|_\infty \leq 1$ and $\hat{\nu}_{\hat{S}} = \operatorname{sgn}(\hat{\beta}_{\hat{S}})$, where $\hat{S} = \{k : \hat{\beta}_k \neq 0\}$.

We set $\hat{\Sigma} = \frac{1}{n} X^T X$. Then we can rewrite the KKT conditions as

$$\hat{\Sigma}(\hat{\beta} - \beta^0) + \lambda \hat{\nu} = \frac{1}{n} X^T \varepsilon.$$

What we are trying to understand is $\hat{\beta} - \beta^0$. So it would be nice if we can find some sort of inverse to $\hat{\Sigma}$. If so, then $\hat{\beta} - \beta^0$ plus some correction term involving $\hat{v}$ would then be equal to a Gaussian.

Of course, the problem is that in the high dimensional setting, that $\hat{\Sigma}$ has no hope of being invertible. So we want to find some approximate inverse $\hat{\Theta}$ so that the error we make is not too large. If we are equipped with such a $\hat{\Theta}$, then we have

$$\sqrt{n}(\hat{\beta} + \lambda \hat{\Theta} \hat{\nu} - \beta^0) = \frac{1}{\sqrt{n}} \hat{\Theta} X^T \varepsilon + \Delta,$$

where

$$\Delta = \sqrt{n}(\hat{\Theta}\hat{\Sigma} - I)(\beta^0 - \hat{\beta}).$$

We hope we can choose $\hat{\Theta}$ so that δ is small. We can then use the quantity

$$b = \hat{\beta} + \lambda\hat{\Theta}\hat{\nu} = \hat{\beta} + \frac{1}{n}\hat{\Theta}X^T(Y - X\hat{\beta})$$

as our modified estimator, called the *debiased Lasso*.

How do we bound Δ? We already know that (under compatibility and sparsity conditions), we can make the ℓ_1 norm of $\|\beta^0 - \hat{\beta}\|$ small with high probability. So if the ℓ_∞ norm of each of the rows of $\hat{\Theta}\hat{\Sigma} - 1$ is small, then Hölder allows us to bound Δ.

Write $\hat{\theta}_j$ for the jth row of $\hat{\Theta}$. Then

$$\|(\hat{\Sigma}\hat{\Theta}^T)_j - I\|_\infty \leq \eta$$

is equivalent to $|(\hat{\Sigma}\hat{\Theta}^T)_{kj}| \leq \eta$ for $k \neq j$ and $|(\hat{\Sigma}\hat{\Theta}^T)_{jj} - 1| \leq \eta$. Using the definition of $\hat{\Sigma}$, these are equivalent to

$$\frac{1}{n}|X_k^T X\hat{\theta}_j| \leq \eta, \quad \left|\frac{1}{n}X_j^T X\hat{\theta}_j - 1\right| \leq \eta.$$

The first is the same as saying

$$\frac{1}{n}\|X_{-j}^T X\hat{\theta}_j\|_\infty \leq \eta.$$

This is quite reminiscent of the KKT conditions for the Lasso. So let us define

$$\begin{aligned} \hat{\gamma}^{(j)} &= \operatorname*{argmin}_{\gamma \in \mathbb{R}^{p-1}} \left\{\frac{1}{2n}\|X_j - X_{-j}\gamma\|_2^2 + \lambda_j\|\gamma\|_1\right\} \\ \hat{\tau}_j^2 &= \frac{1}{n}X_j^T(X_j - X_{-j}\hat{\gamma}^{(j)}) = \frac{1}{n}\|X_j - X_{-j}\hat{\gamma}^{(j)}\|_2^2 + \lambda_j\|\hat{\gamma}^{(j)}\|_1. \end{aligned}$$

The second equality is an exercise on the example sheet.

We can then set

$$\hat{\theta}_j = -\frac{1}{\hat{\tau}_j^2}(\hat{\gamma}_1^{(j)}, \ldots, \hat{\gamma}_{j-1}^{(j)}, -1, \hat{\gamma}_j^{(j)}, \ldots, \hat{\gamma}_{p-1}^{(j)})^T.$$

The factor is there so that the second inequality holds.

Then by construction, we have

$$X\hat{\theta}_j = \frac{X_j - X_{-j}\hat{\gamma}^{(j)}}{X_j^T(X - X_{-j}\hat{\gamma}^{(j)})/n}.$$

Thus, we have $X_j^T X\hat{\theta}_j n = 1$, and by the KKT conditions for the Lasso, we have

$$\frac{\hat{\tau}_j}{n}\|X_{-j}^T X\hat{\theta}_j\|_\infty \leq \lambda_j.$$

Thus, with the choice of $\hat{\Theta}$ above, we have

$$\|\Delta\|_\infty \leq \sqrt{n}\|\hat{\beta} - \beta^0\|_1 \max_j \frac{\lambda_j}{\hat{\tau}_j^2}.$$

Now this is good as long as we can ensure $\frac{\lambda_j}{\hat{\tau}_j^2}$ to be small. When is this true?

We can consider a random design setting, where each row of X is iid $N_p(0, \Sigma)$ for some positive-definite Σ. Write $\Omega = \Sigma^{-1}$.

Then by our study of the neighbourhood selection procedure, we know that for each j, we can write

$$X_j = X_{-j}\gamma^{(j)} + \varepsilon^{(j)},$$

where $\varepsilon_i^{(j)} \mid X_{-j} \sim N(0, \Omega_{jj}^{-1})$ are iid and $\gamma^{(j)} = -\Omega_{jj}^{-1}\Omega_{-j,j}$. To apply our results, we need to ensure that $\gamma^{(j)}$ are sparse. Let use therefore define

$$s_j = \sum_{k \neq j} \mathbf{1}_{\Omega_{jk} \neq 0},$$

and set $s_{max} = \max(\max_j s_j, s)$.

Theorem. Suppose the maximum eigenvalue of Σ is always at least $c_{min} > 0$ and $\max_j \Sigma_{jj} \leq 1$. Suppose further that $s_{max}\sqrt{\log(p)/n} \to 0$. Then there exists constants A_1, A_2 such that setting $\lambda = \lambda_j = A_1\sqrt{\log(p)/n}$, we have

$$\begin{aligned}\sqrt{n}(\hat{b} - \beta^0) &= W + \Delta\\ W \mid X &\sim N_p(0, \sigma^2\hat{\Theta}\hat{\Sigma}\hat{\Theta}^T),\end{aligned}$$

and as $n, p \to \infty$,

$$\mathbb{P}\left(\|\Delta\|_\infty > A_2 s\frac{\log(p)}{\sqrt{n}}\right) \to 0.$$

Note that here X is not centered and scaled.

We see that in particular, $\sqrt{n}(\hat{b}_j - \beta_j^0) \sim N(0, \sigma^2(\hat{\Theta}\hat{\Sigma}\hat{\Theta}^T)_{jj})$. In fact, one can show that

$$d_j = \frac{1}{n}\frac{\|X_j - X_{-j}\hat{\gamma}^{(j)}\}_2^2}{\hat{\tau}_j^2}.$$

This suggests an approximate $(1-\alpha)$-level confidence interval for β_j^0,

$$\mathrm{CI} = \left(b_j - Z_{\alpha/2}\sigma\sqrt{d_j/n}, \hat{b}_j + Z_{\alpha/2}\sigma\sqrt{d_j/n}\right),$$

where Z_α is the upper α point of $N(0,1)$. Note that here we are getting confidence intervals of width $\sim \sqrt{1/n}$. In particular, there is no $\log p$ dependence, if we are only trying to estimate β_j^0 only.

Proof. Consider the sequence of events Λ_n defined by the following properties:

- $\phi_{\hat{\Sigma},s} \geq c_{min}/2$ and $\phi^2_{\hat{\Sigma}_{-j,-j},s_j} \geq c_{min}/2$ for all j
- $\frac{2}{n}\|X^T\Sigma\|_\infty \leq \lambda$ and $\frac{2}{n}\|X_{-j}^T\varepsilon^{(j)}\|_\infty \leq \lambda$.
- $\frac{1}{n}\Sigma^{(j)}\|_2^2 \geq (\Omega_{jj})^{-1}(1 - 4\sqrt{(\log p)/n})$

Question 13 on example sheet 4 shows that $\mathbb{P}(\Lambda_n) \to 1$ for A_1 sufficiently large. So we will work on the event Λ_n.

By our results on the Lasso, we know

$$\|\beta^0 - \hat{\beta}\|_1 \leq c_1 s \sqrt{\log p/n}.$$

for some constant c_1. We now seek a lower bound for $\hat{\tau}_j^2$. Consider linear models

$$X_j = X_{-j}\gamma^{(j)} + \varepsilon^{(j)},$$

where the sparsity of $\gamma^{(j)}$ is s_j, and $\varepsilon_i^{(j)}|X_{-j} \sim N(0, \Omega_{jj}^{-1})$. Note that

$$\Omega_{jj}^{-1} = \mathrm{var}(X_{ij} \mid X_{i,-j}) \leq \mathrm{var}(X_{ij}) = \Sigma_{ij} \leq A.$$

Also, the maximum eigenvalue of Ω is at most c_{min}^{-1}. So $\Omega_{jj} \leq c_{min}^{-1}$. So $\Omega_{jj}^{-1} \geq c_{min}$. So by Lasso theory, we know

$$\|\gamma^{(j)} - \hat{\gamma}^{(j)}\|_1 \leq c_2 s_j \sqrt{\frac{\log p}{n}}$$

for some constant c_2. Then we have

$$\begin{aligned}
\hat{\tau}_j^2 &= \frac{1}{n}\|X_j - X_{-j}\hat{\gamma}^{(j)}\|_2^2 + \lambda\|\hat{\gamma}^{(j)}\|_1 \\
&\geq \frac{1}{n}\|\varepsilon^{(j)} + X_{-j}(\gamma^{(j)} - \hat{\gamma}^{(j)})\|_2^2 \\
&\geq \frac{1}{n}\|\varepsilon^{(j)}\|_2^2 - \frac{2}{n}\|X_{-j}^T\varepsilon^{(j)}\|_\infty \|\gamma^{(j)} - \hat{\gamma}^{(j)}\|_1 \\
&\geq \Omega_{jj}^{-1}\left(1 - 4\sqrt{\frac{\log p}{n}}\right) - c_2 s_j \sqrt{\frac{\log p}{n}} + A_1\sqrt{\frac{\log p}{n}}
\end{aligned}$$

In the limit, this tends to Ω_{jj}^{-1}. So for large n, this is $\geq \frac{1}{2}\Omega_{jj}^{-1} \geq \frac{1}{2}c_{min}$.

Thus, we have

$$\|\Delta\|_\infty \leq 2\lambda\sqrt{n}c_1 s\sqrt{\frac{\log p}{n}}c_{min}^{-1} = A_2 s\frac{\log p}{\sqrt{n}}. \qquad \square$$

Index