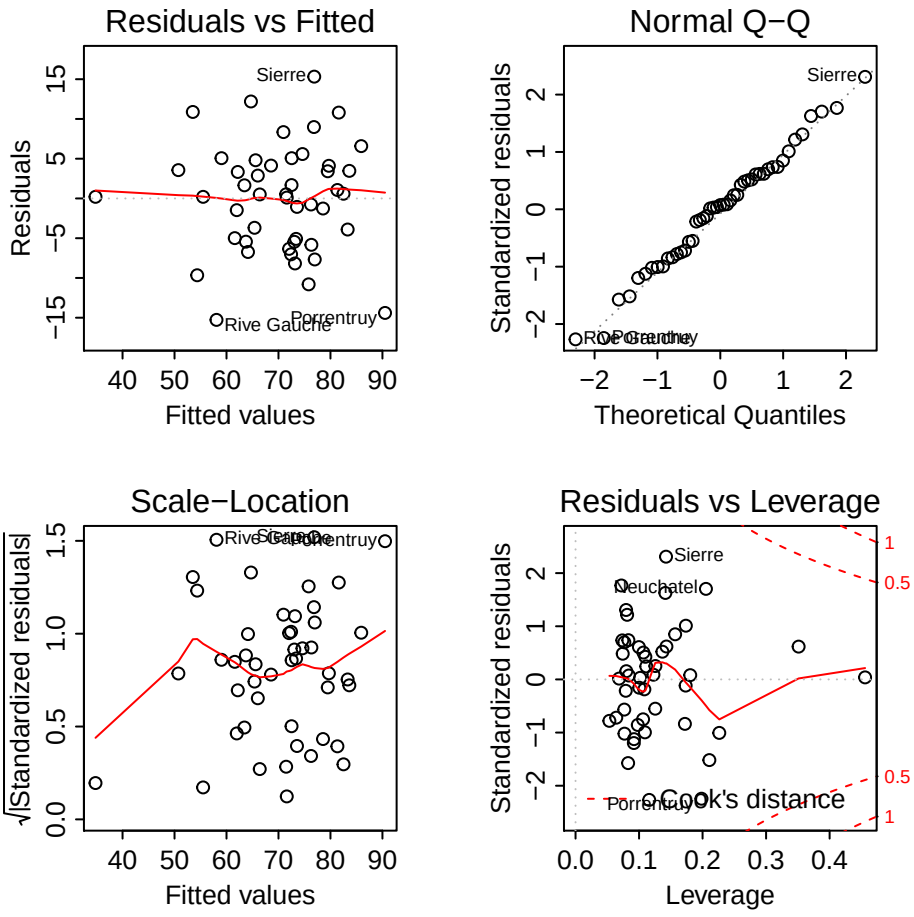
Contents

1 Multivariable LM: checking and selection

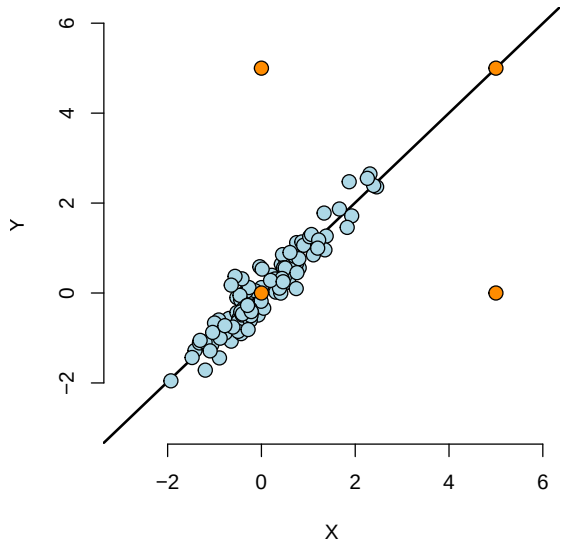
1.1 Model checking (residuals again)

We can generalize this idea to the vertical distances between the observed data and the fitted surface in multivariable settings.

```
data(swiss)
fit <- lm(Fertility ~ . , data = swiss); plot(fit)
```



1.1.1 Outlying, high leverage and influential points



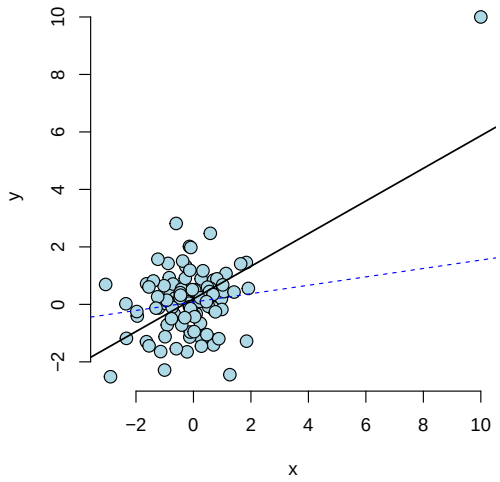
Outliers

- can be the result of spurious or real processes.
- can conform to the regression relationship (i.e being marginally outlying in X or Y, but not outlying given the regression relationship).
- can have varying degrees of influence.

Do `?influence.measures` to see the full suite of influence measures in stats.

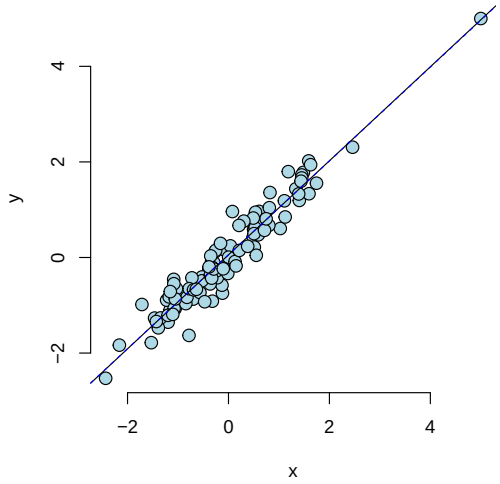
Be wary of simplistic rules for diagnostic plots and measures. The use of these tools is context specific.

- Patterns in your **residual plots** generally indicate some poor aspect of model fit. These can include:
  - Heteroskedasticity.
  - Missing model terms.
  - Temporal patterns.
- Residual **QQ plots** investigate normality of the errors.
- **Leverage measures** can be useful for diagnosing data entry errors.
- **Influence measures**: 'how does deleting or including this point impact a particular aspect of the model'.



The point `c(10, 10)` (the 1st one in the order) has created a strong regression relationship where there shouldn't be one.

```
round(hatvalues(fit)[1 : 10], 3); round(dfbetas(fit)[1 : 10, 2], 3)
#>     1     2     3     4     5     6     7     8     9    10
#> 0.512 0.010 0.016 0.054 0.016 0.035 0.013 0.013 0.017 0.013
#>      1      2      3      4      5      6      7      8      9     10
#>  6.024 -0.020 -0.027 -0.439  0.079  0.007  0.020 -0.029  0.018 -0.085
```



Looking at some of the diagnostics

```
round(hatvalues(fit2)[1 : 10], 3); round(dfbetas(fit2)[1 : 10, 2], 3)
#>     1     2     3     4     5     6     7     8     9    10
#> 0.227 0.019 0.032 0.010 0.013 0.010 0.011 0.010 0.012 0.014
#>      1      2      3      4      5      6      7      8      9     10
#>  0.052  0.028  0.208  0.020  0.062  0.004  0.008 -0.002 -0.017  0.050
```

1.2 Model selection

How do we chose what variables to include in a regression model?

No single easy answer exists and the most reasonable answer would be "It depends.".

- We'll focus on modeling now.
  - In modeling, our interest lies in parsimonious, interpretable representations of the data that enhance our understanding of the phenomena under study.
  - In prediction, we're less concerned with interpretability, what we want is to come up with the best prediction with respect to a specific loss function (then, models with lots of variables, automated search algorithms).

- Finding the simplest model to explain what we're looking at, so as simple as possible, but no simpler.
  - What happens if we omit a variable that we should have included? What about including variables that are unimportant?
- A model is a lense through which to look at our data, to teach us something true about your data set.
- George Box: *all models are wrong, some models are useful.*
- Good modeling decisions are context dependent: a good model for prediction versus one for studying mechanisms versus one for trying to establish causal effects may not be the same.

1.2.1 General rules

- Omitting important variables (**underfitting**) results in bias in the coeficients of interest - unless their regressors are uncorrelated with the omitted ones.
  - Randomization can help. But,If there's too many unobserved confounding variables, even randomization won't help you.
- Including unnecessary variables (**overfitting**) increases (actual) standard errors of other regressors - unless these are uncorrelated with the included ones.
- R^2 increases monotonically as more regressors are included.

Variance inflation 1 (you can skip this and the next simulation)

```
n <- 100; nosim <- 1000
x1 <- rnorm(n); x2 <- rnorm(n); x3 <- rnorm(n);
betas <- sapply(1 : nosim, function(i){
  y <- x1 + rnorm(n, sd = .3)
  c(coef(lm(y ~ x1))[2],
    coef(lm(y ~ x1 + x2))[2],
    coef(lm(y ~ x1 + x2 + x3))[2])
})
dim(betas)
#> [1]    3 1000
round(apply(betas, 1, sd), 5)
#>      x1      x1      x1
#> 0.03146 0.03148 0.03147
```

Variance inflation 2

```
n <- 100; nosim <- 1000
x1 <- rnorm(n); x2 <- x1/sqrt(2) + rnorm(n) /sqrt(2)
x3 <- x1 * 0.95 + rnorm(n) * sqrt(1 - 0.95^2);
betas <- sapply(1 : nosim, function(i){
  y <- x1 + rnorm(n, sd = .3)
  c(coef(lm(y ~ x1))[2],
    coef(lm(y ~ x1 + x2))[2],
    coef(lm(y ~ x1 + x2 + x3))[2])
})
round(apply(betas, 1, sd), 5)
#>      x1      x1      x1
#> 0.03025 0.04222 0.10177
```

Variance inflation factors

- We don't know σ, so we can't actually calculate the exact variance inflation
- However, σ drops out of the relative standard errors.
- The **variance inflation factor** (VIF) is the increase in the variance for the ith regressor compared to the ideal setting where it is orthogonal to the other regressors.
  - (The square root of the VIF is the increase in the sd ...)
- Remember, variance inflation is only part of the picture. We want to include certain variables, even if they dramatically inflate our variance.

Revisting our previous simulation (you can skip)

```
## doesn't depend on which y you use
y <- x1 + rnorm(n, sd = .3)
a <- summary(lm(y ~ x1))$cov.unscaled[2,2]
c(summary(lm(y ~ x1 + x2))$cov.unscaled[2,2],
  summary(lm(y ~ x1 + x2 + x3))$cov.unscaled[2,2]) / a
#> [1]  1.925878 11.907132
temp <- apply(betas, 1, var); temp[2 : 3] / temp[1]
#>       x1       x1
#>  1.947512 11.316833
```

Swiss data (you can skip, use simply the R function `vif`)

```
data(swiss)
fit1 <- lm(Fertility ~ Agriculture, data = swiss)
a <- summary(fit1)$cov.unscaled[2,2]
fit2 <- update(fit, Fertility ~ Agriculture + Examination, data=swiss)
fit3 <- update(fit, Fertility ~ Agriculture + Examination + Education, data=swiss)
  c(summary(fit2)$cov.unscaled[2,2],
    summary(fit3)$cov.unscaled[2,2]) / a
#> [1] 1.891576 2.089159
as.numeric(c(vif(fit2)[1], vif(fit3)[1]))
#> [1] 1.891576 2.089159
```

Swiss data VIFs,

```
library(car)
fit <- lm(Fertility ~ . , data = swiss)
vif(fit)
#>      Agriculture      Examination        Education         Catholic
#>         2.284129         3.675420         2.774943         1.937160
#> Infant.Mortality
#>         1.107542
sqrt(vif(fit))
#>      Agriculture      Examination        Education         Catholic
#>         1.511334         1.917138         1.665816         1.391819
#> Infant.Mortality
#>         1.052398
```

What about residual variance estimation?

- If we underfit the model, the variance estimate is biased.
- If we correctly or overfit the model, including all necessary covariates and/or unnecessary covariates, the variance estimate is unbiased, however, the variance of the variance is larger if we include unnecessary variables.

1.2.2 Covariate selection

- If the models of interest are nested and without lots of parameters differentiating them, it's fairly uncontroversial to use **nested likelihood ratio tests**. (Example to follow.)
- **Automated covariate selection** can be a choice if the covariate space one wants to explore is quite rich.
- for the purposes of prediction, there are many modern methods for traversing large model spaces.
- **Principal components** or factor analytic models on covariates are often useful for reducing complex covariate spaces.
- *Good design* can often eliminate the need for complex model searches at analyses; though often control over the design is limited.

How to do nested model testing in R

```
fit1 <- lm(Fertility ~ Agriculture, data = swiss)
fit3 <- update(fit, Fertility ~ Agriculture + Examination + Education)
fit5 <- update(fit, Fertility ~ Agriculture + Examination + Education + Catholic + Infant.Mortality)
anova(fit1, fit3, fit5)
#> Analysis of Variance Table
#>
#> Model 1: Fertility ~ Agriculture
#> Model 2: Fertility ~ Agriculture + Examination + Education
#> Model 3: Fertility ~ Agriculture + Examination + Education + Catholic +
#>     Infant.Mortality
#>   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
```

```
#> 1     45 6283.1
#> 2     43 3180.9  2    3102.2 30.211 8.638e-09 ***
#> 3     41 2105.0  2    1075.9 10.477 0.0002111 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Nested model testing cannot be applied if model are not nested.

You get into the harder world of automated model selection with things like information criteria and resampling methods.

Recall the linear model

$$Y = \beta_0 + \beta_1 + X_1 + \ldots + \beta_p X_p + \epsilon$$

Despite its simplicity, the *linear model is often competitive* in relation to non-linear methods: it has distinct advantages in terms of its interpretability and often shows good predictive performance.

How can we improve it in interpretability and/or predictive performance?

- Model Interpretability: by removing irrelevant features— i.e., by setting the corresponding coefficient estimates to zero— we can obtain a model that is more easily interpreted. (e.g., by some approaches for automatically performing *variable* or *feature selection*).
- Prediction Accuracy: especially when $p > n$, to control the variance.

Three classes of methods

- **Subset Selection** We identify a subset of the p predictors that we believe to be related to the response. We then fit a model using least squares on the reduced set of variables.
- **Shrinkage** or **regularization** We fit a model involving all p predictors, but the estimated coefficients are shrunken towards zero relative to the least squares estimates. This shrinkage has the effect of reducing variance and can also perform variable selection.
- **Dimension Reduction** We project the p predictors into a M-dimensional subspace, where $M < p$. This is achieved by computing M different linear combinations, or *projections*, of the variables. Then these M projections are used as predictors to fit a linear regression model by least squares.

1.2.3 Subset Selection

1.2.3.1 Best Subset Selection

- Let M_0 denote the *null model*, which contains no predictors. This model simply predicts the sample mean for each observation.
- For $k = 1, 2, \ldots, p$:
  (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.
  (b) Pick the best among these $\binom{p}{k}$ models, and call it M_k. Here best is defined as having the smallest RSS, or equivalently largest R^2.
- Select a single best model from among $M_0, \ldots, M_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

- Although we have presented best subset selection here for least squares regression, the same ideas apply to other types of models, such as logistic regression.
- The **deviance**—negative two times the maximized log-likelihood—plays the role of RSS for a broader class of models.

1.2.3.2 Stepwise Selection

- For computational reasons, best subset selection cannot be applied with very large p. *Why not?*
- Best subset selection may also suffer from statistical problems when p is large: larger the search space, the higher the chance of finding models that look good on the training data, even though they might not have any predictive power on future data (*study before the bias-variance trade-off topic*).
- Thus an enormous search space can lead to *overfitting* and high variance of the coefficient estimates.
- Given the foregoing, **stepwise** methods, which explore a far more restricted set of models, are attractive alternatives to best subset selection.

Forward Stepwise Selection

- Forward stepwise selection begins with a model containing no predictors, and then adds predictors to the model, one-at-a-time, until all of the predictors are in the model.
- In particular, at each step the variable that gives the greatest **additional** improvement to the fit is added to the model.

In Detail

Let M_0 denote the **null model**, which contains no predictors.

- For $k = 0, 1, \ldots, p-1$:
  (a) Consider all $p-k$ models that augment the predictors in M_k with one additional predictor.
  (b) Pick the best among these $p-k$ models, and call it M_{k+1}. Here best is defined as having the smallest RSS, or largest R^2.
- Select a single best model from among $M_0, \ldots, M_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Forward Stepwise Selection: characteristics

- Computational advantage over best subset selection.
- It is not guaranteed to find the best possible model out of all 2^p models containing subsets of the p predictors. *Why not?* Give an example.

Backward Stepwise Selection

- Like forward stepwise selection, backward stepwise selection provides an efficient alternative to best subset selection.
- However, unlike forward stepwise selection, it begins with the full least squares model containing all p predictors, and then iteratively removes the least useful predictor, one-at-a-time.

In Detail

- Let M_p denote the **full model**, which contains all p predictors.
- For $k = p, p-1, \ldots, 1$:
  (a) Consider all k models that contain all but one of the predictors in M_k, for a total of $k-1$ predictors.
  (b) Pick the best among these k models, and call it M_{k-1}. Here best is defined as having the smallest RSS, or largest R^2.
- Select a single best model from among $M_0, \ldots, M_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Backward Stepwise Selection: characteristics

- Like forward stepwise selection, the backward selection approach searches through only $1 + p(p+1)/2$ models, and so can be applied in settings where p is too large to apply best subset selection
- Like forward stepwise selection, backward stepwise selection is not guaranteed to yield the best model containing a subset of the p predictors.
- Backward selection requires that the number of samples n is larger than the number of variables p (so that the full model can be fit). In contrast, forward stepwise can be used even when $n < p$, and so is the only viable subset method when p is very large.

1.2.4 Choosing the Optimal Model

- The model containing all of the predictors will always have the smallest RSS and the largest R^2, since these quantities are related to the training error.
- We wish to choose a model with low test error, not a model with low training error. Recall that training error is usually a poor estimate of test error.
- Therefore, RSS and R^2 are not suitable for selecting the best model among a collection of models with different numbers of predictors.

Estimating test error: two approaches

- We can *indirectly* estimate test error by making an adjustment to the training error to account for the bias due to overfitting.
- We can *directly* estimate the test error, using e.g. either a validation set approach or a cross-validation approach.

1.2.5 Selection criteria

C_p, AIC, BIC, and Adjusted R^2

These techniques adjust the training error for the model size, and can be used to select among a set of models with different numbers of variables.

- Mallow's C_p

$$C_p = \frac{1}{n}(RSS + 2p\hat{\sigma}^2)$$

where p is the total # of parameters used and $\hat{\sigma}^2$ is an estimate of the variance of the error ϵ.

- The AIC criterion is defined for a large class of models fit by maximum likelihood:

$$AIC = -2 \log L + 2p$$

where L is the maximized value of the likelihood function for the estimated model.

- In the case of the linear model with Gaussian errors, maximum likelihood and least squares are the same thing, and C_p and AIC are equivalent.

- The BIC criterion:

$$BIC = \frac{1}{n}(RSS + \log(n) p \hat{\sigma}^2)$$

  - Like C_p, the BIC will tend to take on a small value for a model with a low test error, and so generally we select the model that has the lowest BIC value.
  - Since $\log n > 2$ for any $n > 7$, the BIC statistic generally places a heavier penalty on models with many variables, and hence results in the selection of smaller models than C_p.

- The adjusted R^2

$$\text{Adjusted } R^2 = 1 - \frac{RSS/(n-p-1)}{TSS/(n-1)}$$

where TSS is the total sum of squares.

  - Unlike C_p, AIC, and BIC, for which a small value indicates a model with a low test error, a large value of adjusted R^2 indicates a model with a small test error.
  - Maximizing the adjusted R^2 is equivalent to minimizing $\frac{RSS}{n-p-1}$ which may increase or decrease as the number of variables in the model increases.
  - Unlike the R^2 statistic, the adjusted R^2 statistic pays a price for the inclusion of unnecessary variables in the model.

1.2.6 Resampling procedures

- Each of the previous procedures returns a sequence of models M_k indexed by model size $k = 0, 1, \ldots, p$. Our job here is to select $\hat{k}$. Once selected, we will return model M_k.
- We compute the validation set error or the cross-validation error for each model M_k under consideration, and then select the k for which the resulting estimated test error is smallest.
- This procedure has an advantage relative to C_p, AIC, BIC, and Adjusted R^2, in that it provides a direct estimate of the test error.

1.2.7 Regularization

Shrinkage or Regularization Methods

- The subset selection methods use least squares to fit a linear model that contains a subset of the predictors.

- As an alternative, we can fit a model containing all p predictors using a technique that **constrains** or **regularizes** the coefficient estimates, or equivalently, that *shrinks* the coefficient estimates towards zero.
- II may not be immediately obvious why such a constraint should improve the fit, but it turns out that shrinking the coefficient estimates can significantly reduce their variance.

1.2.7.1 Ridge regression

Penalized RSS

- Recall that the least squares fitting procedure estimates $\beta_0, \ldots, \beta_p$ using the values that minimize

$$\text{RSS} = \sum_{i=1}^{n} \left(u_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 .$$

- In contrast, the ridge regression coefficient estimates $\hat{\beta}^R$ are the values that minimize

$$\text{RSS} + \lambda \sum_{j=1}^{p} \beta_j^2,$$

  where $\lambda \geq 0$ is a **tuning parameter**, to be determined separately.

Ridge regression: characteristics

- As with least squares, ridge regression seeks coefficient estimates that fit the data well, by making the RSS small
- However, the second term, $\lambda \sum_{j=1}^{p} \beta_j^2$, called a **shrinkage penalty**, is small when $\beta_1, \ldots, \beta_p$ are close to 0, and so it has the effect of shrinking the estimates of β_j towards zero.
- The tuning parameter λ serves to control the relative impact of these two terms on the regression coefficient estimates.
- Selecting a good value for λ is critical; cross-validation is used for this.

1.2.7.2 Lasso

Another shrunken approach

- Ridge regression does have one obvious disadvantage: unlike subset selection, which will generally select models that involve just a subset of the variables, ridge regression will include all p predictors in the final model
- The **Lasso** is a relatively recent alternative to ridge regression that overcomes this disadvantage. The lasso coeffs. β_λ^L, minimize the quantity

$$\text{RSS} + \lambda \sum_{j=1}^{p} |\beta_j|,$$

- The lasso uses an l_1 penalty instead of an l_2 penalty.

The Lasso: characteristics

- As with ridge regression, the lasso shrinks the coefficient estimates towards zero.
- However, in the case of the lasso, the l_1 penalty has the effect of forcing some of the coeffs estimates to be exactly equal to zero when the tuning parameter λ is sufficiently large.
- Hence, much like best subset selection, the lasso performs **variable selection**.
- We say that the lasso yields **sparse** models—that is, models that involve only a subset of the variables.
- As in ridge regression, selecting a good value of λ for the lasso is critical; cross-validation is again the method of choice.