Lecture 16. Many predictors, model selection, multimodel inference

I've been ratcheting up the complexity of the model selection methods, and the difficulty of the model selection applications. Now we'll do the application that is probably the most difficult and most contentious. This is the scenario where we have a bunch of observational data, including a response variable and many potential predictors, and we want to ask "Which of these many predictors has support from the data, and what is their relative importance?". This case is challenging because:

- with many predictors there are many possible models
- comparing many predictors and many possible models increases the chance of finding spurious results

This kind of analysis is often called 'exploratory', as opposed to 'confirmatory'. This is because we are interested in seeing what patterns are important in the data, rather than testing a specific *a priori* hypothesis. However, the distinction between exploratory and confirmatory science is usually pretty vague. A controlled experiment with well-defined manipulations, e.g. to test whether adding nitrogen causes plant biomass to increase, is certainly a rigorous test of a predefined hypothesis. But with observational data we often have some hypotheses in mind that we want to test (in an uncontrolled setting), while also being open to any unknown relationships that may emerge from the data.

The example I will use here is the survey data on sole from the Tagus Estuary, which you analyzed in a previous homework. Previouly you used a binomial GLM to quantify how presence/absence of sole is related to salinity. In reality this survey measured many environmental predictors that could be important for sole: salinity, temperature, depth, substrate characteristics (proportion of mud, fine sand, large sand, gravel), as well as season (spring or summer), month of year, and sampling area (there were 4 stations):

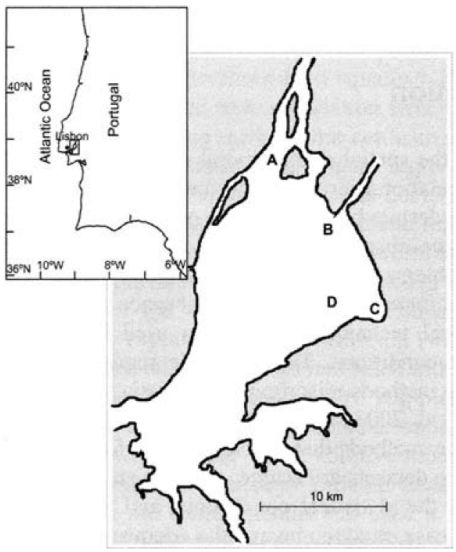


Figure 21.1. The sampling area in the Tagus estuary, Portugal. A, B, C and D indicate the location of the stations.

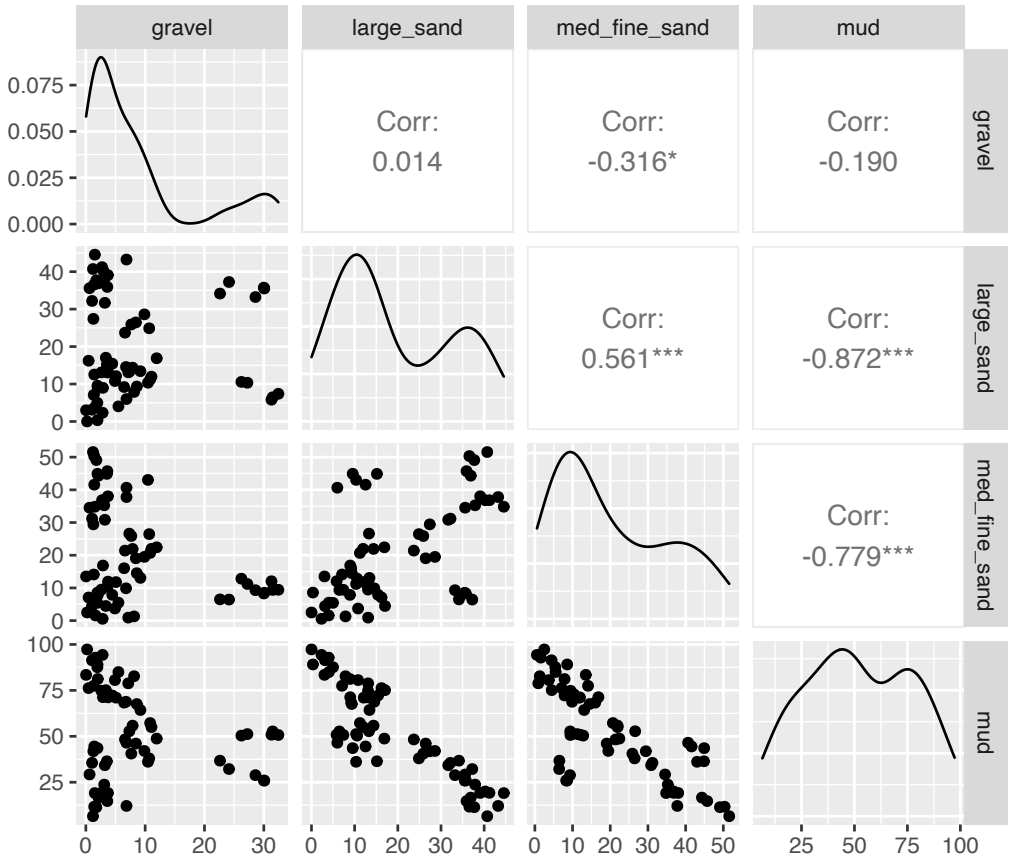
We want to know which of these variables are important for sole, and what the relationships are. We may also want to make predictions for sole abundance at other locations or in the future, so we'll think about how to choose models for prediction as well.

There are 62 samples in this dataset. That represents a good amount of work, but it is still a modest amount of data relative to the number of predictors. I just listed 10 predictors, and some of these (area, month) will include multiple parameters for different factor levels. You will see different rules of thumb for how much data you need, relative to the number of parameters in a model. I like to aim for at least 10 samples per parameter, i.e. $n/K >= 10$. For this dataset that would mean a maximum of 6 parameters for the model. How can we reduce the number of predictors to consider, before even making any models?

Preliminary choice of predictors

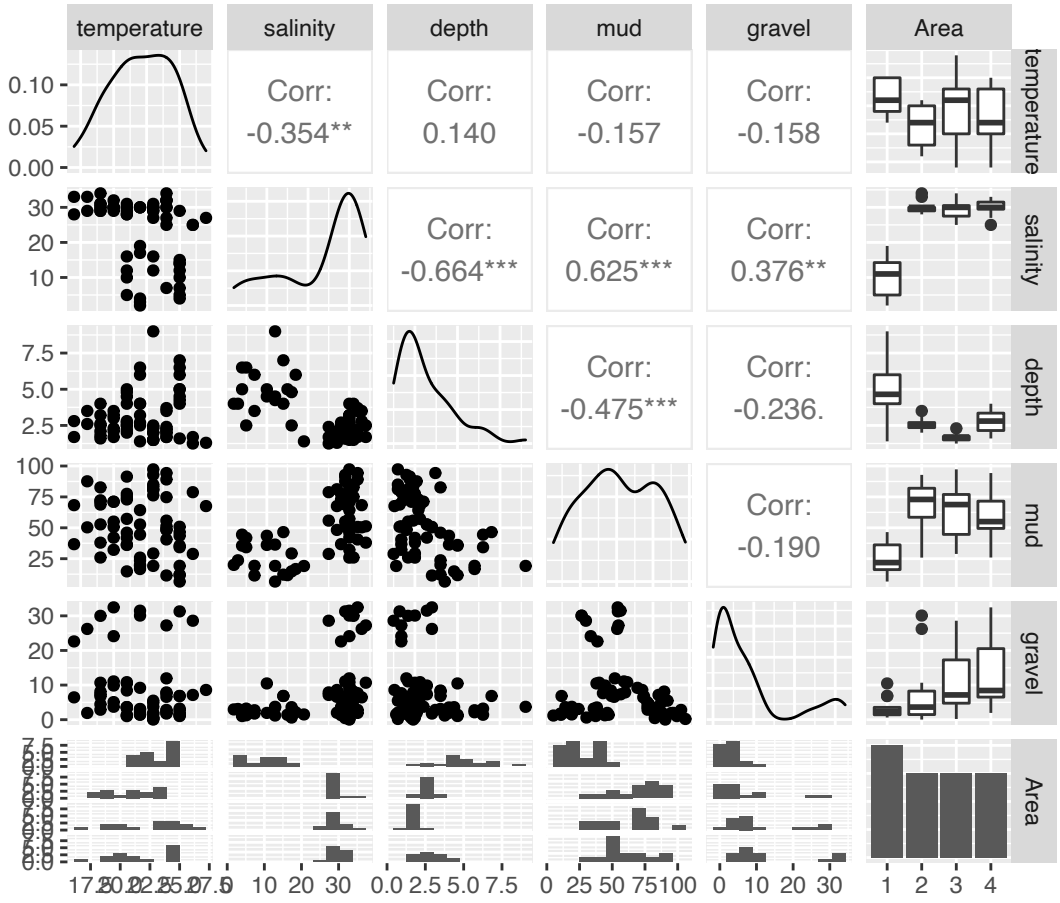
Reducing correlation among predictors by dropping some predictors would be a win-win, in the sense of avoiding collinearity and also having fewer parameters. There are only two seasons, and only four months, so these two predictors are going to be somewhat confounded. I will drop 'month', simply because it contains more levels and therefore more parameters to estimate. The predictors for substrate type (percentage of mud, fine sand, large sand, or gravel) are likely to be correlated, because if one is large then the others are by definition small.

```
ggpairs(solea, columns = 9:12)
```



It looks like the two sand variables are highly correlated with each other as well as with mud. I will drop these, leaving only mud and gravel, because these are the extremes of substrate fineness/coarseness and are not correlated with each other. Let's look at the remaining predictors:

```
ggpairs(solea, columns = c("temperature", "salinity", "depth", "mud", "
gravel", "Area"), lower = list(combo = wrap("facethist", bins = 10)))
```



These look OK, though there is still some correlation, especially among salinity and depth and mud. This is not ideal, but the correlation is not so strong that important effects cannot be estimated. Also these variables vary the most among the four Areas. I will leave them all in, though this is certainly a judgement call.

Now we have 7 predictors with a total of 9 parameters (because Area has 3 parameters). This means n/K is about 7. This is lower than I was aiming for, but since this is an exploratory analysis I'll just go with it.

Five different ways to do a regression with many predictors

Rather than proscribe a protocol for how to do this analysis, I'm going to present 5 different ways to do it. Some of these are probably better than others, but all are common in the ecological literature, so it is worthwhile to compare them and discuss the pros and cons.

1) Just fit one model with all the predictors

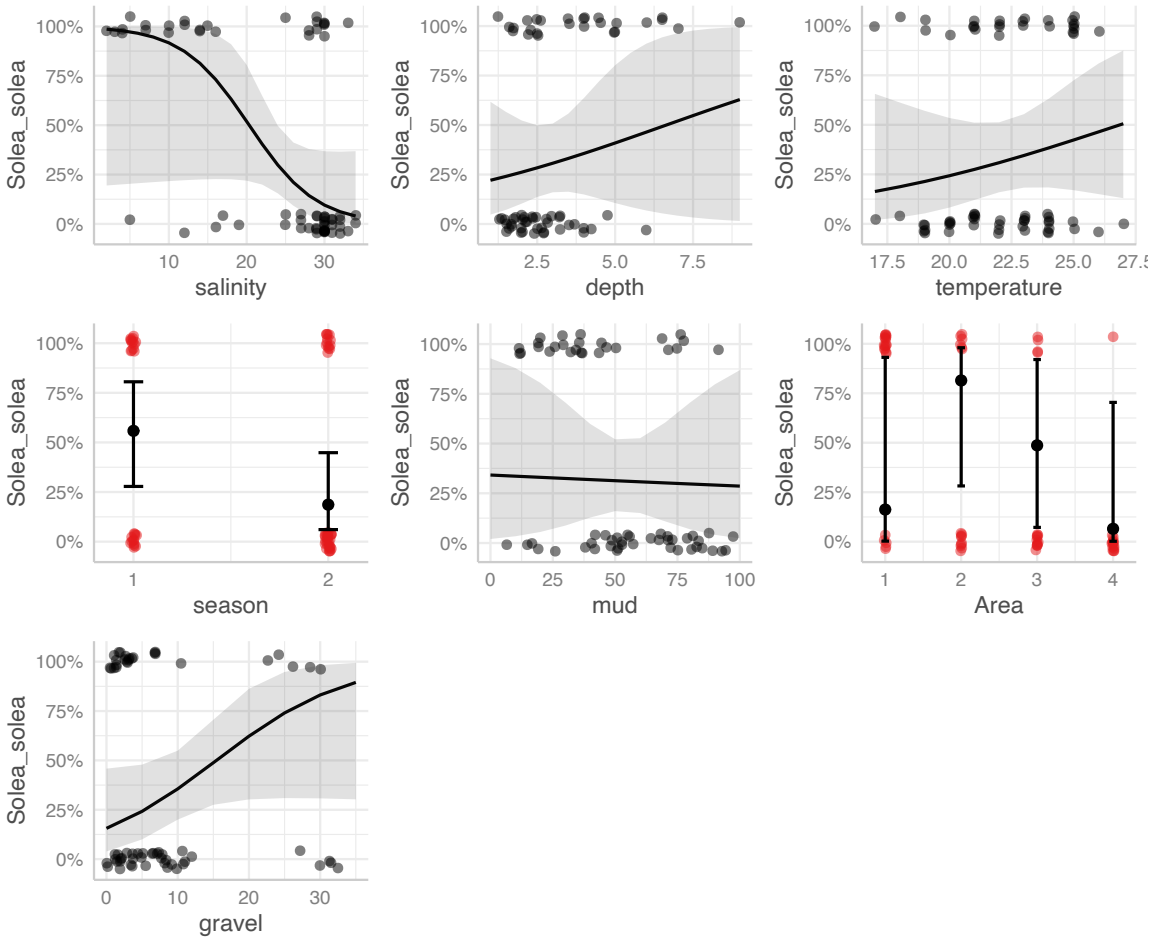
After we've done some preliminary exploration to get a reasonable number of predictors, one approach is to just fit a model that includes all these predictors. One way to motivate this approach is to say "All these effects are probably real, though some will be stronger and some weaker." Then we can test the evidence for each predictor from the data using (marginal) likelihood ratio tests. Let's see what this looks like:

```
big.model = glm(Solea_solea ~ salinity + depth + temperature + season + mud +
Area + gravel, data = solea, family = binomial)

grid.arrange(grobs = plot(ggeffect(big.model), add.data = TRUE, jitter
= 0.05, show.title = FALSE))

Anova(big.model)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: Solea_solea
##             LR Chisq Df Pr(>Chisq)
## salinity        3.91  1      0.048 *
## depth           0.35  1      0.556
## temperature     0.71  1      0.400
## season          3.67  1      0.056 .
## mud             0.01  1      0.930
## Area           10.64  3      0.014 *
## gravel          3.57  1      0.059 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



Based on the LRT results, it looks like there is significant variation between the four areas, and maybe significant effects of salinity, season, and gravel. Based on the effects plots, sole are much more likely to be present at low salinity; sole are about 3 times more likely to be present during spring (season 1); sole are about three times more likely to be present when the substrate has a lot of gravel; and sole are particularly common in area 2. For all of these effects it is clear that the confidence intervals are large, so we are not certain about the exact magnitude of these effects.

Now let's consider the advantages and disadvantages of this one-big-model approach.

Advantages: By including every predictor of interest in a single model, we bypass the model selection problem, i.e. we don't have to worry about comparing the support for different models that include/exclude predictors. In addition, we minimize the *multiple comparisons* problem.

Digression on multiple comparisons. Multiple comparisons is a Type 1 error problem. If I have a model with a single predictor, then there is a 5% chance that

the p-value will be <0.05, if the model assumptions are met. If I have a model with 10 predictors, then the chance of getting at least one 'significant' predictor due to a spurious relationship will be much larger. *If you put enough predictors into a model, you're essentially guaranteed to find at least one spurious relationship*. If I use significance tests to add/remove terms from a model, and then use significance tests to test the remaining predictors, the multiple comparisons problem is compounded even further.

By only constructing a single model, we still are making a decent number of comparisons (1 for each predictor), but at least we are not comparing many models in addition to testing each predictor. It is possible to 'correct' p-values for multiple comparisons, in order to reduce the occurrence of 'false positives', and there are many ways to do this (the Bonferroni correction, and corrections for the 'false discovery rate', are two common approaches). We may cover this later in the course, but for now we will just keep in mind that making many comparisons increases the chance of finding spurious relationships.

Disadvantages. What are the disadvantages of making a single model with every predictor? One disadvantage, maybe the primary one, is that we don't have a lot of data in this case, and so including many predictors in a single model may reduce the confidence in the estimates for all of the predictors. We may get much clearer results if we can drop more terms from the model. A second disadvantage is that the effect of one predictor may depend on which other predictors are in the model, due to correlation among the predictors. To see whether this is the case we would need to do some kind of model comparison/selection.

2) Fit one big model, and drop a few terms that seem unimportant

A common strategy is to reduce the complexity of a model by dropping a few terms that seem particularly unimportant, based on their coefficient estimates and p-values. The rationale is that these unimportant terms are just muddying the water, so to speak, especially if we have limited data. A simpler model is likely to produce more confident estimates of the remaining terms. Based on the big model fit above, we might drop 'depth' and 'mud', as these seem to explain no variation in sole presence:

```
smaller.model = glm(Solea_solea ~ salinity + temperature + season + Area + gra
vel, data = solea, family = binomial)

Anova(smaller.model)

## Analysis of Deviance Table (Type II tests)
##
## Response: Solea_solea
##             LR Chisq Df Pr(>Chisq)
## salinity        4.20  1     0.0405 *
## temperature     0.71  1     0.3980
```

```
## season          3.99  1     0.0457 *
## Area           10.70  3     0.0135 *
## gravel          7.19  1     0.0073 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The LRT results for this model are similar to what we saw before, though indeed the p-values are slightly smaller. I'm not going to plot the effect estimates because in this case they look highly similar to what was plotted above.

Advantages: The advantage of starting with a full model and dropping some terms is that with limited data we may get tighter estimates of the remaining terms.

Disadvantages: By constructing a model, doing significance tests, making a new model, and doing more significance tests, we've increased the multiple comparisons problem. Furthermore, by looking for significant terms and refining the model to include those, we are increasing the chance that these effects are spurious, because we're hunting through the data for any patterns that are there. At this point you shouldn't take the p-values at face value, and you should acknowledge when presenting the results that this is primarily an exploratory analysis.

3) Stepwise model selection

The procedure from #2 can be taken to it's logical conclusion, and terms can be removed/added until you've arrive at a model that can't be improved upon. Terms can be added/removed based on LRT results, but the more common approach these days is to use AIC to remove/add terms. The function step() in R will take a full model and do backward/forward selection based on whether removing or adding a term decreases AIC. You will often see people combining this with LRTs on the final model, e.g.:

```
step.model = step(big.model)
```

```
Anova(step.model)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: Solea_solea
##          LR Chisq Df Pr(>Chisq)
## salinity     5.42  1      0.020 *
## season       3.49  1      0.062 .
## Area         9.99  3      0.019 *
## gravel       6.60  1      0.010 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This looks pretty similar to what I found in version #2, except now temperature has been dropped as well.

Disadvantages. I can't really recommend using stepwise model selection, because with modern computational power you can just compare all possible models and find the one with the lowest AIC, if that is your goal. In contrast, stepwise model selection uses an algorithm that may or may not locate the model with the lowest AIC, depending on the data and the details of the algorithm.

4) Best AIC model

With AIC as a tool, we can compare all possible models that include/exclude each of the predictors, find the one with the lowest AIC, and call that the 'best' model. There are several packages that are helpful for this, I'll use MuMIn:

```
library(MuMIn)
big.model = glm(Solea_solea ~ salinity + depth + temperature + season + mud + Area + gravel, data = solea, family = binomial, na.action = na.pass)
```

Note we have to set 'na.action = na.pass', which is necessary for the building-all-models function.

```
sole.dredge = dredge(big.model, extra = "R^2")
```

The dredge() function will make all possible models that are subsets of the big.model that I created. I also said extra='R^2' so that it will include a psuedo-R^2 as part of the calculation for each model. We can use the model.sel() function to produce a table of the results, which in this case will be huge due to the large number of possible models:

```
> model.sel(sole.dredge)
Global model call: glm(formula = Solea_solea ~ salinity + depth + temperature +
    season + mud + Area + gravel, family = binomial, data = solea,
    na.action = na.pass)
---
Model selection table
    (Intrc)    Area depth   gravl     mud        slnty   seasn tmprt      R^2    df logLik  AICc delta weight
54   4.609e+00 +            0.108500           -0.2533 +               0.4018  7 -27.044 70.1  0.00 0.112
22   3.695e+00 +            0.092980           -0.2472                 0.3688  6 -28.789 71.0  0.97 0.069
118  6.745e-01 +            0.117500           -0.2294 +     0.1683000 0.4084  8 -26.687 71.9  1.89 0.044
56   3.505e+00 +    0.21930 0.104200           -0.2493 +               0.4051  8 -26.866 72.3  2.25 0.036
62   4.564e+00 +            0.110000 0.0010920 -0.2523 +               0.4019  8 -27.044 72.7  2.61 0.031
53   3.491e+00              0.051110           -0.1598 +               0.3025  4 -32.038 72.7  2.69 0.029
17   2.661e+00                                 -0.1299                 0.2527  2 -34.280 72.8  2.70 0.029
49   3.339e+00                                 -0.1361 +               0.2773  3 -33.191 72.8  2.72 0.029
24   2.257e+00 +    0.29620 0.088510           -0.2421                 0.3760  7 -28.417 72.8  2.75 0.028
21   2.746e+00              0.047430           -0.1509                 0.2762  3 -33.242 72.9  2.82 0.027
38   1.705e+00 +            0.094250                  +                0.3499  6 -29.752 73.0  2.90 0.026
```

Here I've abbreviated the output to only show the models with a deltaAIC < 3. The best model includes Area, gravel, salinity, and season as predictors. What do we do now? There are some different sub-options, all of which I've seen people use:

A) Say that Area, gravel, salinity, and season are all important, because they're all in the best model. Report effect sizes, R^2 from this model.

B) Use the top model to do LRTs on the predictors to see if they're 'significant'.

C) Note that there are many models with a similar Akaike weight. Note that Area, gravel, salinity, season are all in most of the best models, and so they all have support from the data. Use the best model to report coefficients / effect size plots / etc.

Out of these three options, probably (C) is the most defensible. The problem with option (A) is that it's hard to look only at the lowest-AIC model when there are so many models in the candidate set that have similar Akaike weights. Option (B) is probably the worst option: although it is used fairly commonly, it has been the object of wrath from many angles. Let's consider why.

Problems with combining information criteria and null hypothesis tests

Why shouldn't we combine the information theory approach and null hypothesis tests, by choosing the best model with AIC and then doing LRTs on that model? There are two main objections:

1. The philosophical objection. The information theory approach and null hypothesis testing are fundamentally different ways to approach inference. Combining them results in a kind of intellectual monster.

2. The data dredging objection. This is the same issue raised earlier when dropping terms from the full model. If we use AIC to sift through all possible models and find the one that best approximates the data, then it's very likely that the confidence intervals and p-values will be too small. This is because we've pre-screened or 'snooped' for any patterns in the data using AIC, and then we conduct null hypothesis tests on predictors that we already know have some pattern in the data.

What approach to use?

You may have noticed that for all the approaches I've covered, the conclusions are similar: salinity, Area, gravel, and season all have some support from the data, though all have considerable uncertainty as well. If you care a lot about the $p<0.05$ threshold, then the results will vary somewhat depending on which approach you

use. Does it really matter which method we use? In my experience, very strong effects will always pop out, and very weak effects will not get any support, regardless of how you construct the model. It's really the effects with intermediate support that may or may not seem important, depending on how you select the model and which other predictors are in the model.

For theoretical/philosophical reasons it's probably best to avoid 1) stepwise methods, and 2) selecting the best AIC model and then doing null hypothesis tests on it. Nonetheless, you will see these approaches often in publications, so I can't in good faith say that these approaches are generally agreed upon to be "wrong".

Personally I prefer to either make one large model and do marginal tests (#1), if I have a lot of data, or look at the full set of models with AIC and evaluate support for predictors across the models (#4C). The former approach avoids too much data snooping, and is consistent with the idea that effects are unlikely to be truly null in most cases. The latter approach acknowledges that if we are comparing many models, we may be uncertain about which is the best. This *model selection uncertainty* can be quantitatively accounted for using an approach called multi-model inference, which we will cover in the next lecture (this is approach #5, if you're counting.

Multimodel inference

To wrap up our material on model selection, I want to talk briefly about multi-model inference. When we compare models using AIC, it is often the case that no single model has overwhelming support from the data (overwhelming support would be something like an Akaike weight > 0.9). This is particularly common when we are comparing models that overlap in the predictors they contain; the most extreme case is when we compare all possible models constructed from a set of predictors. In this kind of situation, adding or removing weak predictor(s) from a model will result in small changes of AIC, resulting in many models that have some support from the data. This was the case for the sole example from the last lecture; when we construct all possible models and look at the AIC table, it looks like this:

```
> model.sel(sole.dredge)
Global model call: glm(formula = Solea_solea ~ salinity + depth + temperature +
    season + mud + Area + gravel, family = binomial, data = solea,
    na.action = na.pass)
---
Model selection table
    (Intrc)    Area depth   gravl     mud        slnty   seasn tmprt      R^2    df logLik  AICc delta weight
54   4.609e+00 +            0.108500           -0.2533 +              0.4018  7 -27.044 70.1  0.00 0.112
22   3.695e+00 +            0.092980           -0.2472                0.3688  6 -28.789 71.0  0.97 0.069
118  6.745e-01 +            0.117500           -0.2294 +     0.1683000 0.4084  8 -26.687 71.9  1.89 0.044
56   3.505e+00 +    0.21930 0.104200           -0.2493 +              0.4051  8 -26.866 72.3  2.25 0.036
62   4.564e+00 +            0.110000 0.0010920 -0.2523 +              0.4019  8 -27.044 72.7  2.61 0.031
53   3.491e+00              0.051110           -0.1598 +              0.3025  4 -32.038 72.7  2.69 0.029
17   2.661e+00                                 -0.1299                0.2527  2 -34.280 72.8  2.70 0.029
49   3.339e+00                                 -0.1361 +              0.2773  3 -33.191 72.8  2.72 0.029
24   2.257e+00 +    0.29620 0.088510           -0.2421                0.3760  7 -28.417 72.8  2.75 0.028
21   2.746e+00              0.047430           -0.1509                0.2762  3 -33.242 72.9  2.82 0.027
38   1.705e+00 +            0.094250                  +              0.3499  6 -29.752 73.0  2.90 0.026
86   5.523e+00 +            0.091220           -0.2607      -0.0707900 0.3709  7 -28.683 73.3  3.28 0.022
102 -3.741e+00 +            0.106900                  +     0.2539000 0.3689  7 -28.786 73.5  3.48 0.020
30   3.622e+00 +            0.095550 0.0018740 -0.2456                0.3689  7 -28.787 73.5  3.48 0.020
58   5.805e+00 +                     -0.0335600 -0.2647 +              0.3666  7 -28.907 73.8  3.73 0.017
18   3.531e+00 +                                -0.2114                0.3160  5 -31.403 73.8  3.77 0.017
26   4.742e+00 +                     -0.0284600 -0.2508                0.3397  6 -30.256 74.0  3.91 0.016
50   4.193e+00 +                                -0.2136 +              0.3379  6 -30.344 74.1  4.08 0.015
120 -2.867e-01 +    0.22140 0.112800           -0.2298 +     0.1642000 0.4115  9 -26.516 74.3  4.25 0.013
81   5.218e+00                                 -0.1412      -0.1033000 0.2594  3 -33.988 74.4  4.32 0.013
6    8.696e-01 +            0.077390                                   0.3095  5 -31.710 74.4  4.38 0.013
85   5.371e+00              0.047860           -0.1633      -0.1055000 0.2828  4 -32.942 74.6  4.50 0.012
126  7.153e-01 +            0.115800 -0.0014140 -0.2308 +     0.1692000 0.4084  9 -26.686 74.6  4.59 0.011
```

This is an abbreviated version of the table. With 7 predictors there are $2^7 = 128$ possible models. If I had considered interactions, the number would be even greater. From the table we can see that the best model only has a weight of 0.112, and there are 11 models within delta-AIC of 3, and 17 within a delta-AIC of 4.

How can we deal with all this information, and the fact that no model is overwhelmingly supported? If we care primarily about individual predictors, we could focus on which predictors occur regularly in the models at the top of the list, and use variable importances (based on Akaike weights) to get a sense for the relative support of variables. We would still be left with the issue of which model to use for reporting the effect sizes and standard errors / confidence intervals for those effects. If the estimates don't vary much across models, using the best model is one option.

An alternative approach is to explicitly account for *model selection uncertainty*, i.e. the fact that we aren't certain which model is the best, and if we took a new sample from the same biological processes then the relative ranking of the models by AIC might change. This approach is called *multimodel inference*. There are different approaches to multimodel inference, but the basic logic is the same: we want to average the model predictions and coefficient estimates across models, accounting for the amount of support each model has. Probably the most theoretically sound way to do this is with a fully Bayesian approach, but that is difficult to implement and outside the scope of this course. A relatively easy

alternative that approximates a Bayesian approach is to use AIC and Akaike weights to average across models.

Multimodel inference with AIC

I will not get into the theoretical justification for these multimodel inference procedures; you can refer to Burnham & Anderson chapter 4. The basic thing we want to do is to use the coefficient estimates for each model to get an average coefficient estimate across models. To account for the fact that different models have different support, we will weight each estimate using the Akaike weight, which approximates the probability that that model is the best model. So for a parameter β, the model-averaged estimate is

$$\bar{\hat{\beta}} = \frac{\sum w_i \hat{\beta}_i}{\sum w_i}$$

where $\hat{\beta}_i$ is the coefficient estimate from model *i*, and w_i is the Akaike weight for model *i*. There are two ways to do this average:

1. Average over all models that contain the parameter β
2. Average over all models, and set $\beta = 0$ in models where β is absent

The first way is effectively assuming that the effect of β is real, and asking what is the mean estimate for that effect across the models that include β. The second way is more conservative; it is assuming that if a model where β is excluded has support, then that is evidence that $\beta = 0$. The second estimator is referred to as a 'shrinkage' estimator, because the estimate for all parameters will be shrunk towards zero.

So now we have a way (or two ways) to get a model-averaged estimate for a predictor. The second part is to get a model-averaged estimate for the uncertainty around that predictor. Now that we are acknowledging that we are uncertain about the 'best' model, there are two sources of uncertainty to include:

1. Parameter uncertainty *within* each model (i.e. the standard error / confidence interval in that model)
2. Parameter uncertainty *across* models

These two sources of uncertainty can be combined using the following approximation:

$$\widehat{se}\left(\bar{\hat{\beta}}\right) = \sum w_i \sqrt{\widehat{se}\left(\hat{\beta}_i \middle| \text{model}_i\right)^2 + \left(\hat{\beta}_i - \bar{\hat{\beta}}\right)^2}$$

This is saying that the standard error for the model-averaged β, $\widehat{se}\left(\bar{\hat{\beta}}\right)$, is determined both by the standard error within each model, $\widehat{se}\left(\hat{\beta}_i \middle| \text{model}_i\right)$, and the the variation in β across models, $\left(\hat{\beta}_i - \bar{\hat{\beta}}\right)^2$. This estimate of uncertainty is called the *unconditional* standard error, because the uncertainty is not conditional on using a particular model. Because we are accounting for model selection uncertainty, *this standard error will tend to be larger than the standard error for any particular model*. That makes sense because we are now properly accounting for an extra source of uncertainty. This standard error can be used to create a 95% confidence interval, by assuming that the model-averaged parameter has an approximately normal sampling distribution.

Fortunately for us, we live in an opulent age of abundant and free software. We can use the package MuMIn (or glmulti, or AICcmodavg) to automate the process of getting model-averaged estimates and standard errors. For the sole example, we've already decided to look at all possible models. So the main question for multimodel inference is 'which models should we average over?'. There's no automatic answer to this question. Some possibilities include:

- Average over the models that cumulatively receive 95% of the Akaike weight
- Average over the models that are within some threshold for delta-AIC, e.g. 4 or 5 or 6
- Average over the models that have an evidence ratio, relative to the best model, within a threshold like 0.1.

These different choices will produce similar results in most cases. For a situation like the sole example, there are a large number of models that receive some Akaike weight, and so we might want to be a little restrictive in how many models we use for multimodel inference. Here I'll use all models with delta-AIC < 4.

```
library(MuMIn)

big.model = glm(Solea_solea ~ salinity + depth + temperature + season + mud + Area +
gravel, data = solea, family = binomial, na.action = na.pass)

sole.dredge = dredge(big.model)

sole.avg = model.avg(sole.dredge, delta < 4)

summary(sole.avg)
```

```
##
## Call:
## model.avg.model.selection(object = sole.dredge, subset = delta <
##     4)
##
## Component models:
##       df logLik  AICc Delta Weight
## 1356   7 -27.04 70.05  0.00   0.20
## 135    6 -28.79 71.03  0.97   0.12
## 13567  8 -26.69 71.95  1.89   0.08
```

```
## 12356  8 -26.87 72.30  2.25   0.06
## 13456  8 -27.04 72.66  2.61   0.05
## 356    4 -32.04 72.74  2.69   0.05
## 5      2 -34.28 72.75  2.70   0.05
## 56     3 -33.19 72.77  2.72   0.05
## 1235   7 -28.42 72.80  2.75   0.05
## 35     3 -33.24 72.88  2.82   0.05
## 136    6 -29.75 72.95  2.90   0.05
## 1357   7 -28.68 73.33  3.28   0.04
## 1367   7 -28.79 73.54  3.48   0.03
## 1345   7 -28.79 73.54  3.48   0.03
## 1456   7 -28.91 73.78  3.73   0.03
## 15     5 -31.40 73.82  3.77   0.03
## 145    6 -30.26 73.96  3.91   0.03
##
## Term codes:
##        Area       depth      gravel         mud    salinity      season
##           1           2           3           4           5           6
## temperature
##           7
##
## Model-averaged coefficients:
##             Estimate Std. Error Adjusted SE z value Pr(>|z|)
## (Intercept)   3.2894     2.9769      3.0168    1.09    0.276
## Area2         2.3193     2.6889      2.7290    0.85    0.395
## Area3         0.9609     2.6663      2.7047    0.36    0.722
## Area4        -1.2299     3.0336      3.0782    0.40    0.690
## gravel        0.0961     0.0497      0.0505    1.90    0.057 .
## salinity     -0.2248     0.1176      0.1197    1.88    0.060 .
## season2      -1.3301     0.8039      0.8189    1.62    0.104
## temperature   0.1268     0.2215      0.2249    0.56    0.573
## depth         0.2530     0.3771      0.3851    0.66    0.511
## mud          -0.0116     0.0299      0.0304    0.38    0.703
##
## Full model-averaged coefficients (with shrinkage):
##             Estimate Std. Error Adjusted SE z value Pr(>|z|)
## (Intercept)  3.28943    2.97686     3.01683    1.09     0.28
## Area2        1.85552    2.57780     2.61130    0.71     0.48
## Area3        0.76873    2.41564     2.44953    0.31     0.75
## Area4       -0.98395    2.75758     2.79688    0.35     0.72
## gravel       0.07796    0.05845     0.05907    1.32     0.19
## salinity    -0.20677    0.12825     0.13007    1.59     0.11
## season2     -0.79975    0.90152     0.90960    0.88     0.38
## temperature  0.01884    0.09656     0.09770    0.19     0.85
## depth        0.02867    0.15015     0.15244    0.19     0.85
## mud         -0.00169    0.01214     0.01231    0.14     0.89
##
## Relative variable importance:
##                      salinity gravel Area season temperature mud  depth
## Importance:          0.92     0.81   0.80 0.60   0.15        0.15 0.11
## N containing models:   15       12     13    9      3           4    2
```

The process is dangerously easy. I make a full model that contains all the terms I'm interested in, I used dredge() to create all possible models, and I used model.avg() to get model-averaged parameter estimates. model.avg() allows you to give it logical conditions just like the subset() function, and I said 'delta < 4' to use all the models with delta-AIC < 4.

The function returns info on all the models used, the model-averaged coefficient estimates and standard errors, the shrinkage version of the model averages, and the variable importances (in this subset of models). It's probably a good idea to ignore the p-values, if you want to avoid the intellectual horror of combining information theory with null hypothesis testing. However you can get confidence intervals for the parameters with confint(), which tells you pretty much the same thing:

```
confint(sole.avg)
```

```
##                  2.5 %   97.5 %
## (Intercept) -2.623443 9.202307
## Area2       -3.029463 7.668148
## Area3       -4.340228 6.262013
## Area4       -7.263096 4.803290
## gravel      -0.002937 0.195208
## salinity    -0.459488 0.009831
## season2     -2.935154 0.274916
## temperature -0.313905 0.567601
## depth       -0.501826 1.007855
## mud         -0.071152 0.047981
```

What do these results tell us? In the other approaches we tried, the predictors Area, salinity, gravel, and season all received some support, though the effects had substantial uncertainty. The model-averaged estimates tell a similar story. The confidence intervals for gravel and salinity slightly overlap zero, while the effect of season overlaps it more. One downside of using model-averaged parameters is that there is not a good way to ask whether a factor with >2 levels is important (e.g. Area). For this you will have to rely on variable importances and the model selection table. Area receives a variable importance of 0.71 across all models, which argues that it is important though not overwhelmingly so.

Unfortunately we can't use convenient plotting functions like effects() to plot the model-averaged results, but the equivalent plots could be created using the predict() function, which will return predicted relationships and standard errors around those predictions. And there may be other packages that can do it. We can compare the model-averaged results to the results from the lowest-AIC model:

```
best.model = get.models(sole.dredge, subset = 1)[[1]]
confint(best.model)
```

```
## Waiting for profiling to be done...
```

```
##                 2.5 %   97.5 %
## (Intercept)  1.65635  8.81871
## Area2       -1.66910  7.33051
## Area3       -3.10537  5.57853
## Area4       -6.46517  3.80099
## gravel       0.02484  0.21080
## salinity    -0.53597 -0.03678
## season2     -2.85382  0.06253
```

The results are similar, but the confidence intervals are a little narrower. This is consistent with the fact that we are now assuming that the best model is the only model, i.e. we are not accounting for model selection uncertainty.

When should you use multimodel inference, as opposed to other approaches we've looked at? This is still a pretty subjective decision, and will depend on your goals. Multimodel inference seems particularly apt if you are interested in making predictions, because accounting for model selection uncertainty should lead to more robust predictions. It also seems apt for truly exploratory analyses where you want to see what patterns emerge, and you aren't particularly interested in isolating a specific hypothesized relationship. If you have a small amount of data relative to the number of predictors you want to test, then multi-model inference may result in larger uncertainty in the estimated effects, compared to an approach that reduces model complexity.