- ggplot2
- dplyr
- knitr
- pryr

Let's look at `vapply()`. It's used as such (run this code):

```
vapply(mtcars, class, character(1))
vapply(list(matrix(1:9, nrow=3), matrix(1:20, nrow=5)), dim, numeric(2))
```

Exercise. Let `trims = seq(0, 0.5, 0.1)` and `x = rnorm(100)`. Rewrite the expression `lapply(trims, function(trim) mean(x, trim=trim))` to not need an anonymous function.[1]

In general, when calling `vapply(args, func, example)`, *each time `func()` is called on an element of `args`*, the output must have the same type and length as `example`. Otherwise, `vapply()` stops with an error. Also, when returning multiple numeric vectors, `vapply()` will add appropriate dimensions to the output.

Exercise. With a variety of different functions, test the behavior of `vapply()` and `sapply()` when the list of arguments is an empty list (`list()`). How would the behavior of `vapply()` help you write code robust to errors and bugs?[^bugs]

Exercise. What happens when `sapply(args, func)` is called in a situation where `func()` returns vectors of different lengths for different elements of `args`? How can `vapply()` be used to detect unexpected instances of this situation?

Exercise. Experiment with lists containing `Sys.time()`. In particular, what happens when you use an `*apply()` function to determine the class of every element in a list containing `Sys.time()` for one of its entries?

=============

Regional-level analysis

We'll also sometimes want to take a step back and group some of our observations together to do data analysis at a different level.

- Aggregate at the level of regions using the `aggregate()` function. (*Hint:* Pass in `FUN=median`.)
- Compute the correlations between the resulting columns.
- How do these compare with the correlations you calculated at the state level? What do you think explains the difference?

[1] Try `lapply(trims, mean, x=x)`.

Adding interaction terms

We can add *interaction terms* to a linear regression very easily: in the list of predictors we pass in to the `lm()` function, we can include the interaction of `var1` with `var2` by including `var1:var2` or `var1*var2`.

- What's the difference between including `var1:var2` or `var1*var2`? (*Hint:* Try regressing against nothing aside from the interaction term.)
- How much additional predictive power can you get by including well-chosen interaction terms in your regression? Which interaction terms help the most?

More *n*-dominoes

Suppose that you have a single copy of every unique n-domino for some value of n.

- Write a function `make_circle(n)` that tries to construct a valid circle of n-dominoes from a *single copy* of every unique n-domino.
  - In the process of doing so, keep track of your various approaches.
  - Are there values of n for which no approach seems to work?
  - If so, can you make an argument about why you can't make a valid circle of n-dominoes for those values of n (using a single copy of every n-domino)? It may be instructive to look at the intermediate steps of your algorithm and how it fails.
  - Give a proof of your heuristic results.
- Construct several arbitrary square matrices of identical size but differing contents (try to make them interesting, with lots of nonzero numbers), and consider the trace of their *product*. If you change the order in which they're multiplied, does the trace necessarily stay the same? Characterize the permutations of the order of matrices in the product which leave the trace of the product unchanged. (It may be instructive to consider all permutations of 3 different matrices.) What about the case where the matrices are symmetric?