# Introduction to the R Statistical Computing Environment
# R Programming II: Exercises

**John Fox**
(McMaster University)
**ICPSR**

2021

1. * *Least-squares regression:* Write an R function for linear least-squares regression. You might call the function `ls()`, with arguments `X`, for the model matrix, and y for the response vector. Optionally include an argument `intercept`, defaulting to `TRUE`, controlling whether a column of ones should be added to the model matrix (but do, in any event, make provision for an intercept).

   If you know how to do least-squares regression by a QR or singular-value decomposition, then by all means feel free to do that, but I suggest that you simply "naïvely" use textbook formulas to compute the least-squares solution $\mathbf{b}$, fitted values $\widehat{\mathbf{y}}$, residuals $\mathbf{e}$, and the estimated covariance matrix of the coefficients $\widehat{V}(\mathbf{b})$ as

$$\begin{aligned} \mathbf{b} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} \\ \widehat{\mathbf{y}} &= \mathbf{X}\mathbf{b} \\ \mathbf{e} &= \mathbf{y} - \widehat{\mathbf{y}} \\ \widehat{V}(\mathbf{b}) &= \frac{\mathbf{e}'\mathbf{e}}{n-p}(\mathbf{X}'\mathbf{X})^{-1} \end{aligned}$$

   where $n$ is the number of rows (observations) of $\mathbf{X}$ and $p$ is the number of columns (parameters).

   Have your function return a list with elements for the coefficients (say, `$coef`), fitted values (`$fitted`), residuals (`$residuals`), and coefficient covariance matrix (`$vcov`).

   Test your function with the `longley` data set included with R. This data set was used by W. Longley (1967), "An appraisal of least-squares programs from the point of view of the user," *Journal of the American Statistical Association*, **62**, 819–841, to demonstrate the instability of then-current statistical software with highly collinear data. See `?longley` for details of the data. Using both your `ls()` function and the standard R `lm()` function, compute and compare the least-squares regression coefficients and standard errors; for example

```
mod.lm <- lm(Employed ~., data=longley)
coef(mod.lm)

X <- data.matrix(longley[, 1:6])
y <- longley[, "Employed"]
mod.ls <- ls(X, y)
```

```
mod.ls$coef

mod.ls$coef/coef(mod.lm)
sqrt(diag(mod.ls$vcov))/sqrt(diag(vcov(mod.lm)))
```

Do you get the right answer?

2. *Object-oriented programming:* Convert your linear least-squares function from Exercise 1 into an S3 generic function with `default` and (optionally) `formula` methods.[1] The `default` method for your function should return an object of class `"ls"`, and the `formula` method could call the `default` method.

   Write `print.ls()` and `summary.ls()` methods for the `"ls"` objects created by your function. Also write methods for some of the standard R modeling generic functions (unless there's a `default` method that works) — `vcov()` (covariance matrix of coefficients), `coef()` (coefficient vector), `fitted()` (fitted values), `residuals()`, and so on.

3. *Maybe the best problem:* Pick a statistical method with which you are intimately familiar and program it in R.

[1] See Section 10.10 of the *R Companion* for how to write a `formula` method.