Midterm Exam

36-350, Statistical Computing, Fall 2013

11 October 2013

INSTRUCTIONS: Provide all answers in your bluebook. Only work in the bluebook will be graded. Clearly indicate which problem each answer goes with.

The exam will be curved.

Explain your reasoning when possible, even for the multiple choice questions; this will help us to give partial credit.

You can use any notes you like, or books. You cannot access the Internet in any way, or run R.

Benchmarking

When we do surveys, totals or averages often do not match the values we know they ought to add up to — e.g., if we survey people about their income, the survey average will often not add up to the known national average income. This can be an especially big problem if the survey needs to be broken down into results for many small areas (such as individual cities in a national survey). One approach to dealing with this is *benchmarking*, adjusting survey results to match known national averages and variances, while distorting the data as little as possible. Distortion is typically measured by the mean squared error between the original measurements and the values after benchmarking.

Here is some code for benchmarking. `x` is a vector of length n, each component of which is a survey measurement for a different small area.

```
benchmark <- function(x,mu,v,l1=1000,l2=1000,method="BFGS",...) {        #  1
  obj <- function(par) {                                                 #  2
    l1 <- par[1]                                                         #  3
    l2 <- par[2]                                                         #  4
    y <- par[-(1:2)]                                                     #  5
    mse <- mean((y-x)^2)                                                 #  6
    constraint1 <- abs(mean(y)-mu)                                       #  7
    constraint2 <- abs(var(y)-v)                                         #  8
    return(mse+l1*constraint1+l2*constraint2)                            #  9
  }                                                                      # 10
  fit <- optim(par=c(l1,l2,x),fn=obj,method=method,...)                  # 11
  return(y=fit$par[-(1:2)],distortion=mse)                               # 12
}                                                                        # 13
```

All of the questions on this exam refer to this code.

1. (15) What function does `benchmark` minimize?

   (a)
   $$\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2 + \ell_1 \left|\mu - \frac{1}{n}\sum_{i=1}^{n} x_i\right| + \ell_2 \left|v - \frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \frac{1}{n}\sum_{j=1}^{n} x_j\right)^2\right|$$

   (b)
   $$\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2 + \ell_1 \left|\mu - \frac{1}{n}\sum_{i=1}^{n} y_i\right| + \ell_2 \left|v - \frac{1}{n-1}\sum_{i=1}^{n}\left(y_i - \frac{1}{n}\sum_{j=1}^{n} x_j\right)^2\right|$$

   (c)
   $$\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2 + \ell_1 \left|\mu - \frac{1}{n}\sum_{i=1}^{n} y_i\right| + \ell_2 \left|v - \frac{1}{n-1}\sum_{i=1}^{n}\left(y_i - \frac{1}{n}\sum_{j=1}^{n} y_j\right)^2\right|$$

   (d)
   $$\sum_{i=1}^{n}(y_i - x_i)^2 + \ell_1 \left|\mu - \frac{1}{n}\sum_{i=1}^{n} y_i\right| + \ell_2 \left|v - \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \mu)^2\right|$$

   (e) The code does not actually define any proper mathematical function to optimize.

2. (10) What are `mu` and `v`?

   (a) The mean and variance (respectively) of `x`

   (b) Scaling factors which control how hard `optim` tries to adjust the mean and variance.

   (c) The mean and variance (respectively) of `y`

   (d) The benchmark values of the mean and variance (respectively)

   (e) `mu` and `v` are not defined in this code

3. (10) Why does `obj` not need to be passed `mu` and `v` as arguments?

   (a) Actually, `obj` does need to be passed them; this is a bug.

   (b) `obj` is defined inside `benchmark`, which has `mu` and `v` as arguments.

   (c) `mu` and `v` are global variables, so `obj` can find them in the global environment; this may lead to weird behavior if other parts of the code use variables with those names.

   (d) `obj` finds `mu` and `v` from components of its argument `par`.

   (e) `obj` does not use `mu` and `v`, because its only argument is `par`.

4. (10) Why does `benchmark` not return the first two components of `fit$par`?

   (a) They are always equal to the arguments `l1` and `l2`, so there's no need to return them.

   (b) They are always equal to 1000, so there's no need to return them.

   (c) They are the mean and variance of what it does return, so there's no need to return them.

   (d) They just help ensure the benchmark constraints are obeyed, without any substantive meaning.

   (e) Not returning them is a bug.

5. (20) Alas! `benchmark` contains a buggy line.

   (a) (5) What line is the bug on?

   (b) (10) Explain what's wrong.

   (c) (5) How might the line be fixed?

6. (5) What will R do here?

```
identical(x,benchmark(x,mu=mean(x),v=var(x)))
```

   (a) Return FALSE, because the whole point of benchmarking is to change the values in `x`.

   (b) Return TRUE, because benchmarking to the sample mean and variance should do nothing.

   (c) Probably return FALSE, because benchmarking to the sample mean and variance should do nothing, but finite-precision arithmetic will usually introduce round-off error.

   (d) `benchmark` will generate an error message about self-referential arguments.

   (e) It depends on how the bug is fixed.

7. (15) Using mean squared error treats each of the small areas equally, but some of them might be measured more precisely than others. One way to do this would be to replace the mean squared error,

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - x_i)^2$$

with a weighted squared error,

$$\frac{1}{n}\sum_{i=1}^{n} w_i(y_i - x_i)^2$$

where we have a vector w of positive weights, putting more weight on the more-precisely-measured areas.

Explain how to modify the code for `benchmark` so that it could, optionally, use weighted squared error, but its default would still be to use mean squared error. Be as concrete as possible (e.g., "change `foo` in line 6 to `bar`" is better than "make sure the code does `baz` somewhere").

8. (15) Suppose that `x` is a very long vector that contains measurements for each county in the country, but we want to benchmark each state separately, so that `mu` is a vector of 51 state-level means and `v` is a vector of 51 state-level variances. You can also presume that `states` is a vector which says which state each county is in (in the same order as `x`). Sketch out how to use `benchmark` to benchmark each state separately, and get back a single vector of adjusted measurements. Be as concrete as possible.