Lab 5: Planning Production Continued

2025/06/30

Reproducibility and Functions

Some of the lectures have included examples of planning production for a factory that turns steel and labor into cars and trucks. Below is a piece of code that optimizes the factory's output (roughly) given the available resources, using a `repeat` loop. It's embedded in a function to make it easier for you to run.

```
factory.function <- function (cars.output=10, trucks.output=5) {
  factory <- matrix(c(40,1,60,3),nrow=2,
    dimnames=list(c("labor","steel"),c("cars","trucks")))
  available <- c(1600,70); names(available) <- rownames(factory)
  slack <- c(8,1)/5; names(slack) <- rownames(factory)
  output <- c(cars.output, trucks.output); names(output) <- colnames(factory)

  passes <- 0 # How many times have we  been around the loop?
  repeat {
     passes <- passes + 1
     needed <- factory %*% output # What do we need for that output level?
     # If we're not using too much, and are within the slack, we're done
     if (all(needed <= available) &&
         all((available - needed) <= slack)) {
       break()
     }
     # If we're using too much of everything, cut back by 10%
     if (all(needed > available)) {
       output <- output * 0.9
       next()
     }
     # If we're using too little of everything, increase by 10%
     if (all(needed < available)) {
       output <- output * 1.1
       next()
     }
     # If we're using too much of some resources but not others, randomly
     # tweak the plan by up to 10%
      # runif == Random number, UNIFormly distributed, not "run if"
     output <- output * (1+runif(length(output),min=-0.1,max=0.1))
  }

  return(list(output=output,pass=passes))
}
```

5. Run the function above with the command

```
factory.function()
```

```
## $output
##     cars   trucks
## 10.18018 19.87624
##
## $pass
## [1] 300048
```

to obtain a default output value, starting from a very low initial planned output. What is the final output capacity obtained?

6. Repeat this four more times to obtain new output values. Do these answers differ from each other? If so why? If not, why not?
7. Right now, the number of `passes` is a value held within the function itself and not shared. Change the code so that the number of `passes` will be returned at the end of the function, as well as the final `output`.
8. Now, set the initial output levels to 30 cars and 20 trucks and run the code. What is the final output plan (`output`)? What is the final demand for resources (`needed`)? Is the plan within budget and within the slack? How many iterations did it take to converge (`passes`)? For all but `output` you will need to either print this message out deliberately, or return an object that contains all the quantities you want.