Bios 6301: Assignment 5

Ashley Spirrison

Due Thursday, 12 October, 1:00 PM

$5^{n=day}$ points taken off for each day late.

Good. 34/40

40 points total.

Submit a single knitr file (named `homework5.rmd`), along with a valid PDF output file. Inside the file, clearly indicate which parts of your responses go with which problems (you may use the original homework document as a template). Add your name as `author` to the file's metadata section. Raw R code/output or word processor files are not acceptable.

Failure to name file `homework5.rmd` or include author name may result in 5 points taken off.

Question 1

15 points

```
#writing secant function, validating, and finding iterations to convergence
secant_function <- function(x) {
  return(cos(x)-x)}
x0 <- 0
x1 <- 1
tolerance <- 1e-9
max_iterations <- 100
iterations <- 0
while(iterations < max_iterations) {
  x2 <- x1 -(secant_function(x1) * (x1-x0)) / (secant_function(x1) - secant_function(x0))
  if (abs(x2-x1) < tolerance) {
    cat("converged to root: ", x2, "\n")
    return(x2)
    break
  }
  x0 <- x1
  x1 <- x2
  iterations <- iterations + 1
}
```

```
## converged to root:  0.7390851
```

```
iterations
```

```
## [1] 5
```

```r
#writing N-R function and finding iterations to convergence to allow for secant comparison

n_r_function <-function(x) {
  return (cos(x) -x)
}
n_r_deriv <- function(x) {
  return(-sin(x) - 1)
}
x0 <- 0
tolerance <- 1e-9
max_iterations <- 100
iterations <- 0
while (iterations < max_iterations) {
  x1 <- x0 -n_r_function(x0)/ n_r_deriv(x0)
  if(abs(x1-x0) < tolerance) {
  cat("converged to root (N-R):", x1, "\n")
  return(x1)
  break
}
x0 <- x1
iterations <- iterations + 1
}
```

```
## converged to root (N-R): 0.7390851
```

```r
iterations
```

```
## [1] 4
```

```r
#N-R method is faster, coverged to root in 4 iterations versus 5 iterations for secant method
```

Question 2 (20 points)

newton method number of iterations is actually largely dependent on your initial guess. good reasoning from what you saw, but secant may be better in other cases

```r
  #using verbose to control output
#creating craps function
craps <- function(verbose = TRUE) {
  x <- sum(ceiling(6 * runif(2)))
  if (verbose) cat("First roll:", x, "\n")

  if (x %in% c(7, 11)) {
    if (verbose) cat("win\n")
    return(TRUE)
  } else {
    y <- x
    if (verbose) cat("y is", y, "\n")
    repeat {
      roll <- sum(ceiling(6 * runif(2)))
      if (verbose) cat("Next roll:", roll, "\n")
      if (roll == y) {
```

```
        if (verbose) cat("win\n")
        return(TRUE)

      } else if (roll == 7 || roll == 11) {
        if (verbose) cat("lose\n")
        return(FALSE)

      }
    }
  }
}

#setting seed and running 3 games
set.seed(100)

for (i in 1:3) {
  craps()
}
```

```
## First roll: 4
## y is 4
## Next roll: 5
## Next roll: 6
## Next roll: 8
## Next roll: 6
## Next roll: 10
## Next roll: 5
## Next roll: 10
## Next roll: 5
## Next roll: 8
## Next roll: 9
## Next roll: 9
## Next roll: 5
## Next roll: 11
## lose
## First roll: 6
## y is 6
## Next roll: 9
## Next roll: 9
## Next roll: 11
## lose
## First roll: 6
## y is 6
## Next roll: 7
## lose
```

```
#determining seed to win 10 straight games
deter_w_seed <- function() {
  seed <- 1
  while (TRUE) {
    set.seed(seed)
    wins <- 0
    consec_wins <- 0
```

```
    while (consec_wins < 10) {
      if (craps(FALSE)) {
        consec_wins <- consec_wins + 1
      } else {
        consec_wins <- 0
      }
    }
    if (consec_wins == 10) {
      cat("Seed", seed, "results in ten consecutive wins\n")
      for (i in 1:10) {
        cat("Game", i, "Result: win\n\n")
      }
      break
    }
    seed <- seed + 1
  }
}

deter_w_seed()
```

```
## Seed 1 results in ten consecutive wins
## Game 1 Result: win
##
## Game 2 Result: win
##
## Game 3 Result: win
##
## Game 4 Result: win
##
## Game 5 Result: win
##
## Game 6 Result: win
##
## Game 7 Result: win
##
## Game 8 Result: win
##
## Game 9 Result: win
##
## Game 10 Result: win
```

I am not sure what putting FALSE into your craps function does
-I think it resulted in not running the game and just spit out the first seed you gave
-the correct seed is 880

Question 3

5 points

This code makes a list of all functions in the base package:

```
objs <- mget(ls("package:base"), inherits = TRUE)
funs <- Filter(is.function, objs)
```

Using this list, write code to answer these questions.

1. Which function has the most arguments? (3 points)

```
most_argue <- 0
most_argue_function <- NULL
for(fun in funs) {
  argue <- length(formals(fun))
  if (argue > most_argue) {
    most_argue <- argue
    most_argue_function <- fun
  }
}

cat("function with most arguements:", deparse(most_argue_function), "with", most_argue, "arguements.\n")
```

should be the scan function.

```
## function with most arguements: function (file = "", what = double(), nmax = -1L, n = -1L, sep = "",
```

1. How many functions have no arguments? (2 points)

```
no_argue_functions <- 0
for(fun in funs) {
  argue <- length(formals(fun))
  if (argue == 0) {
    no_argue_functions <- no_argue_functions +1

  }
}

cat("number of functions with no arguements:", no_argue_functions, "\n")
```

```
## number of functions with no arguements: 229
```

Hint: find a function that returns the arguments for a given function.