Stat 610 Homework 4

Due Thursday, October 5, 11:59pm.

Assignment

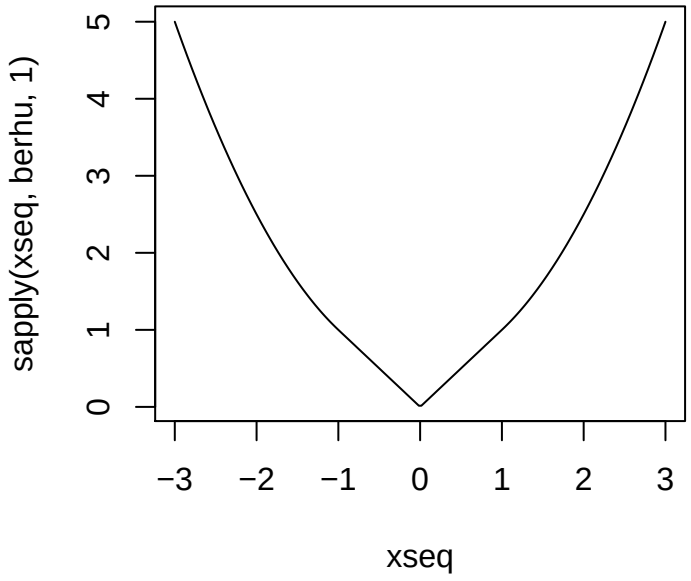
Your assignment is to debug the functions in the `homework4-buggy.R` script.

The script also has some test cases so that you know when you've gotten the right results.

1. The Berhu penalty is sometimes used for variable selection. It is defined as

$$f(x, \delta) = \begin{cases} |x| & |x| \leq \delta \\ \frac{x^2}{2\delta} + \frac{\delta}{2} & |x| > \delta \end{cases}$$

   It is a smooth function, and if $\delta = 1$, should look like this:



   The `berhu` function should compute the values of this function, but it doesn't seem to be working right, please fix it.

2. The trimmed mean is sometimes used in robust statistics, and is defined by taking the mean of a set of numbers after removing the top α and bottom α fraction of the numbers (i.e., removing all the values above the $1 - \alpha$ quantile and below the α quantile). `trimmed_mean(x,`

`trim`) function should compute the trimmed mean of `x`, trimming the top and bottom `trim` fraction of the values.

The function `trimmed_mean` in the script is not working properly: it gives NA on the test data now, when the result should be 2.75.

You can check your debugged verison against the `mean` function in R: `trimmed_mean(x, trim = alpha)` should always give the same output as `mean(x, trim = alpha)`.

3. String processing. You found some interesting data on wikipedia (https://en.wikipedia.org/wiki/Mercer_Quality_of_Living_Survey) about city quality of life, but it's not formatted in a convenient way and you need to write a function to take the string representation into a data frame. The `process_table` function tries to do this, but has several bugs that you need to fix.

   Once you've debugged `process_table`, the results should look like this (don't worry about the unicode characters).

```
> process_table(city_rankings)
  X1 Rank Old.Rank Score              City
1  1    1        1 108.6            Vienna
2  2    2        2 108.0     Z<U+00FC>rich
3  3    3        4 107.0            Munich
4  4    4        6 107.2 D<U+00FC>sseldorf
5  5    5        7 107.0         Frankfurt
```

4. In gradient descent https://en.wikipedia.org/wiki/Gradient_descent, we try to find the value $x^\star$ that minimizes a function f.

   Given a step size s and, a tolerance ε, and an initial guess at the minimizer x_0, gradient descent is the following procedure:

   - Set $i = 1$.
   - Let $x_i = x_{i-1} - sf'(x_{i-1})$.
   - If $|f(x_i) - f(x_{i-1})| < \varepsilon$, return x_i, otherwise increment i by 1 and repeat the previous step.

   The `gradient_descent` function takes a function, the function's derivative, a starting position, a step size, and a tolerance and attempts to implement the procedure described above.

   As an example,

```
gradient_descent(function(x) x^2, function(x) 2 * x,
                 start = 1, step_size = .1, tol = 1e-10)
```

   should return 0, or something close, because the function we want to minimize is $f(x) = x^2$, which has a minimum at $x = 0$.

   Unfortunately, the function as implemented now doesn't work, and you'll need to fix it.

5. There is a variant of gradient descent that uses *backtracking line search* to determine the step size. The only difference between it and the gradient descent described above is that instead of a fixed step size s, the step size is determined by backtracking line search.

   As with gradient descent, we want to find the minimizer of a function f. Given a tolerance ε and an initial guess at the minimizer x_0, gradient descent with backtracking line search is the following procedure:

   - Set $i = 1$
   - Let $x_i = x_{i-1} - \text{backtrack}(x, f, \alpha, \beta) f'(x_{i-1})$.
   - If $|f(x_i) - f(x_{i-1})| < \varepsilon$, return x_i, otherwise increment i by 1 and repeat the previous step.

   The function backtrack computes a step size t using the following procedure:

   - Set $t = 1$.
   - If $f(x - tf'(x)) > f(x) - \alpha t f'(x)^2$, set $t = \beta t$, otherwise stop and return t.

   The functions as written now are buggy, and you can check whether you have debugged correctly by trying

```
backtrack_desc(function(x) x^2, function(x) 2 * x, start = 10,
               alpha = .03, beta = .8, epsilon = 1e-10)
backtrack_desc(function(x) x^2, function(x) 2 * x, start = 1,
               alpha = .03, beta = .8, epsilon = 1e-10)
```

   which should both return something close to 0.

Submission parameters

- Submit the debugged R script. Add comments describing what you changed and what the problem was.