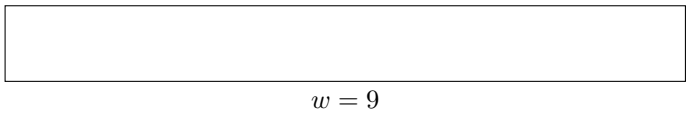
CSC 191B: Lab #4: Squaring a Rectangle

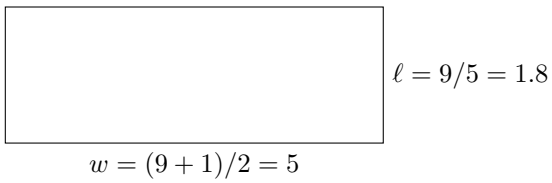
Learning Outcomes

- Use of nested, local, and anonymous functions
- More practice with loops and numerical error
- Use of Newton's method and MATLAB's general nonlinear equation solver `fzero`

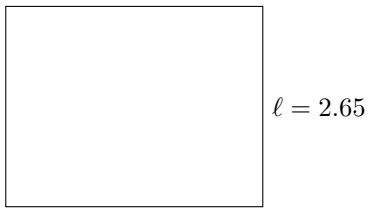
Background. In this lab we'll write our own function to compute the square root of a positive number. Given an input a, let's consider it to represent the area of a rectangle. If we can make that rectangle a square, then the side length of the square is the square root of a. For example, suppose $a = 9$: we can draw a rectangle that has width 9 and length 1:



To make the rectangle more square (to make w closer to ℓ), we can set the new width to be the average of w and ℓ: $(w + \ell)/2$. To maintain the same area, we can set the new length to be a divided by the new width. This yields a new rectangle:



Another step yields:



Continuing this process will lead w and ℓ to converge to $\sqrt{a}$. If we focus on width and let w_i be the width at iteration i, we have the formula

$$w_{i+1} = (w_i + \ell_i)/2 = (w_i + a/w_i)/2.$$

It turns out that this iteration corresponds to Newton's method applied to the function

$$f(x) = x^2 - a$$

which has a zero at $\sqrt{a}$. For general functions, Newton's method uses the following update function

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}.$$

For our function to compute the square root, we have $f'(x) = 2x$, which yields the formula

$$x_{i+1} = x_i - \frac{x^2 - a}{2x} = x_i - (x_i - a/x_i)/2 = (x_i + a/x_i)/2,$$

which matches our formula above.

Problem Statement. In this lab, we'll write our function to compute the square root of a number, and we'll compare it to MATLAB's general nonlinear equation solver. In order to display information about each iteration, we'll make use of a local function and a nested function. MATLAB has a general equation solver that we can use to compute square roots as well (though the built-in `sqrt` is more fine-tuned for problem). To use the general solver, we'll need to use an anonymous function.

1. Write a function `mysqrt` that takes a single positive input `a` and returns its approximate positive square root `x`. Your function should iterate until the relative error $|x^2 - a|/a$ is smaller than `1e-14`. Include code to print the current approximate square root as well the the relative error for every iteration. You must make use of the following two functions (one should be used as a local function and one should be used as a nested function):

```
% prints iteration info
function print_values()
    fprintf('%2d   %21.15f   %.2e\n', i, x, comp_err(x,a));
end

% compute the relative error
function e = comp_err(s,b)
    e = abs(s*s-b) / b;
end
```

2. Write a script called `use_fzero` to use MATLAB's `fzero` function to compute the square root of a number (`help fzero` and `doc fzero` will be useful). The `fzero` function will accept as an optional parameter `optimset('Display','iter')`, which will ask it to display per-iteration information similar to `mysqrt`.

3. Compare `mysqrt` and `fzero` in computing $\sqrt{2}$ and $\sqrt{23456}$. Copy the results of each into the respective MATLAB files.

Discussion.

1. How do you decide whether to write a custom function in its own MATLAB file or to write a nested or local function in the same file as another function?

2. The `fzero` function requires an extra input for starting value. How does the choice of starting value affect convergence behavior?

If you're interested in the inner workings of `fzero`, you can read the relevant chapter in Moler's *Numerical Computing with MATLAB* online: `https://www.mathworks.com/content/dam/mathworks/mathworks-dot-com/moler/zeros.pdf`.

What to turn in.

- Two MATLAB files `mysqrt.m` and `use_fzero.m` that contain the code and the results of $\sqrt{2}$ and $\sqrt{23456}$. Include the answers to the discussion questions in `use_fzero.m`.

Grading rubric

- Code: 30 points for `mysqrt`, plus 10 points each for correct use of `print_values` and `comp_err`
- Code: 20 points for `use_fzero`
- Results: 10 points total for results of `mysqrt` and `fzero`
- Discussion: 10 points each for the discussion questions