Homework 1

Bisection method and regula falsi

Suppose we have a real function of one variable $f : \mathbb{R} \to \mathbb{R}$ continuous on the interval $[a, b] \in \mathbb{R}$, for which the function values in the extreme points of this interval have the opposite sign. The bisection method is an iterative algorithm for finding the root of such a function on a specified interval. In each iteration, the bisection method performs the following two steps:

1. **Finding the center point:** the method finds the center of the given interval
$$c = \frac{a + b}{2}.$$
2. **Update of the search interval:** if the function values $f(a)$ and $f(c)$ have the same sign, then the new interval for the root search is $[c, b]$. Otherwise, the new interval is $[a, c]$.

The above two steps are repeated until the desired accuracy is reached or the maximum number of iterations is reached. As a stopping criterion, we can use the functional value at point c, i.e. the algorithm will terminate if $|f(c)| < \varepsilon$ for the specified tolerance $\varepsilon \in \mathbb{R}$.

The regula falsi method differs from the bisection method only in how it selects the c point. The following formula is used for this method
$$c = \frac{a \cdot f(b) - b \cdot f(a)}{f(b) - f(a)}.$$

Input

Implement a `findroot` function that finds the root of a given function on a given interval. The `findroot` function must have the following input arguments (in the order listed):

- `method` : the method that will be used to find the root,
- `f` : function of one variable whose root we want to find,
- `a` : lower limit of the interval,
- `b` : upper limit of the interval.

Choose the appropriate types for all input parameters of the `findroot` function. Use the following type hierarchy to differentiate the root search method.

```
abstract type BracketingMethod end

struct Bisection <: BracketingMethod end
struct RegulaFalsi <: BracketingMethod end
```

Additionally, the `findroot` function must accept the following keyword arguments (values after `=` are values):

- `atol = 1e-8` : algorithm tolerance,
- `maxiter = 1000` : maximum number of iterations.

When implementing, note that the bisection method and the regula falsi method differ only in the selection of a new point. Write a general `findroot` function for both methods. Use multiple-dispatch and write a `midpoint` function that will return a new point based on the method used. This function must have the following input arguments (in the order listed):

- `method` : the method that will be used to find the root,
- `f` : function of one variable whose root we want to find,
- `a` : lower limit of the interval,
- `b` : upper limit of the interval.

The `findroot` function must also meet the following properties:

- The function must check that `a < b` holds, and if not, it must swap variables to satisfy this inequality.
- If the function is given a root to look for, the function must return this root without any further calculation.
- If the function values at the endpoints of the specified interval have the same sign, the function must return DomainError [https://docs.julialang.org/en/v1/base/base/#Core.DomainError] with with a meaningful error message.

Use the function to test

$$f(x) = x^3 - x - 2$$

and the interval $[1, 2]$.

courses/b0b36jul/en/hw/hw1.txt · Last modified: 2022/10/24 15:04 by adamluk3