SI211: Numerical Analysis
Project

Prof. Boris Houska

Deadline: Oct 28, 2020

The goal of this project is to implement algorithmic differentiation in forward and backward mode by using operator overloading. Requirements are:

- Think about an efficient storage format and use a programming that supports operator overloading (such as Matlab, Julia, C++, etc).

- Implement algorithmic differentiation in forward mode (similar to what we discussed in the lecture) and provide a user interface of the form

$$g = \text{ADforward}(f) .$$

Here, the user provides the function $f : \mathbb{R}^m \to \mathbb{R}^n$. The function "ADforward" should then return the function

$$g(x,d) = \frac{\partial f(x)}{\partial x} \cdot d$$

to the user in order to evaluate the forward derivative of f with respect to the given direction $d \in \mathbb{R}^n$. Make sure that your code works for all combination of the atom operations

$$+, \ -, \ *, \ / \ , \sin, \ \cos \ .$$

- Implement algorithmic differentiation in backward mode and provide a user interface of the form

$$h = \text{ADbackward}(f) .$$

As above, the use provides the function $f : \mathbb{R}^m \to \mathbb{R}^n$. The function "ADbackward" should then return the function

$$h(x,d) = d^\intercal \frac{\partial f(x)}{\partial x}$$

to the user in order to evaluate the backward derivative of f with respect to the backward seed vector $d \in \mathbb{R}^m$. As for the foward mode, make sure that your code works for all combination of the atom operations

$$+, \ -, \ *, \ / \ , \sin, \ \cos \ .$$

- Implement the function

```
function f(x)
  a = 1;
  b = 1;
  for i=1:length(x)
    y = 0.3*sin(a)+0.4*b;
    z = 0.1*a+0.3*cos(b)+x[i];
    a = y;
    b = z;
  end
  return [a;b];
end
```

  and compute the gradient of $f : \mathbb{R}^{2020} \to \mathbb{R}^2$ at $x = [1; 1; 1 \ldots; 1] \in \mathbb{R}^{2020}$ with

  1. Numerical differentiation using finite differences
  2. Using your AD forward code (with 2020 seeds)
  3. Using your AD backward code (with 2 seeds)

  and compare your results in terms of accuracy and run-time (in milliseconds). Which of the three implementations is the fastest / most accurate ?

The project consists of a report ($\geq$ 6 pages) and a software. Please submit both to our TAs before the above mentioned deadline.

1 Requirements on the Project Report

Write a short report (preferably in Latex) containing the following sections:

1. *Title and Authors of your report*
2. *Introduction* (Briefly explain how AD works)
3. *Code Design* (explain what you have implemented and how your code works. Which data format do you use? How did you implement the AD backward routine by using operator overloading)
4. *User Manual* (briefly summarize how to run your code; such that at least the TAs can reproduce your results).
5. *Numerical Results* (plot/visualize and explain your numerical results)
6. *Conclusion* (summarize the highlights of your results and outline what could still be improved)