Mathematics for Intelligent Systems
Lecture 6 Homework
(Gradients and Hessians)

Marc Toussaint, Andrea Baisero

1 Problem 1: Finite Difference Gradient Checking

(a) Implement the following pseudo code for empirical gradient checking in the programming language of your choice:

k

Require: $x \in \mathbb{R}^n$, function $f: \mathbb{R}^n \to \mathbb{R}^d$, function $df: \mathbb{R}^n \to \mathbb{R}^{d\times n}$
  initialize $\hat{J} \in \mathbb{R}^{d\times n}$, and $\epsilon = 10^{-6}$
  for $i = 1:n$ **do**
    $\hat{J}_{\cdot i} = [f(x+\epsilon e_i) - f(x-\epsilon e_i)]/2\epsilon$ ▷ assigns the ith column of $\hat{J}$
  end for
  if $\left\|\hat{J} - df(x)\right\|_\infty < 10^{-4}$ return true; else false

Here e_i is the ith standard basis vector in $\mathbb{R}^n$.

(b) Test this for

- $f: x \mapsto Ax$, $df: x \mapsto A$, $x \sim$@randn(n,1)@ ($\mathcal{N}(0,1)$ in Matlab) and $A \sim$@randn(n,n)@
- $f: x \mapsto x^\intercal x$, $df: x \mapsto 2x^\intercal$, $x \sim$@randn(n,1)@

2 Problem 2: "Backprop" in a Neural Net

Consider the function

$$f: \mathbb{R}^{h_0} \to \mathbb{R}^{h_3},\ f(x_0) = W_2\sigma(W_1\sigma(W_0x_0))$$

where $W_i \in \mathbb{R}^{h_{i+1}\times h_i}$ and $\sigma(z) = 1/(e^{-z}+1)$ is the sigmoid function, which here is applied element-wise. Choose $(h_0, h_1, h_2, h_3) = (5, 10, 10, 2)$. Note that the function can also be written as the "chain":

$$x_0 \ \mapsto\ z_1 = W_0x_0 \ \mapsto\ x_1 = \sigma(z_1) \ \mapsto\ z_2 = W_1x_1 \ \mapsto\ x_2 = \sigma(z_2) \ \mapsto\ f = W_2x_2$$

For this exercise, let's focus on the Jacobian of f, which analytically is given as

$$\frac{\partial f}{\partial x_0} = \frac{\partial f}{\partial x_2}\,\frac{\partial x_2}{\partial z_2}\,\frac{\partial z_2}{\partial x_1}\,\frac{\partial x_1}{\partial z_1}\,\frac{\partial z_1}{\partial x_0}$$

The Jacobians in this are:

$$\frac{\partial x_l}{\partial z_l} = \text{diag}(x_l \circ (1 - x_l)),\ \frac{\partial z_{l+1}}{\partial x_l} = W_l,\ \frac{\partial f}{\partial x_2} = W_2$$

(a) Write code to implement f and ∂f for arbitrary $(W_0, .., W_2)$.

Sample random $(W_0, .., W_2)$ using @randn@ and a random x_0, and check the implemented Jacobian by comparing to the finite difference approximation.

Debugging Tip: If your first try does not work right away, the typical approach to debug is to "comment out" parts of your function f and df. For instance, start with testing $f(x) = W_0 x_0$; then test $f(x) = \sigma(W_0 x_0)$; then $f(x) = W_1 \sigma(W_0 x_0)$; then I'm sure all bugs are found.

3 Problem 3: Logistic Regression Gradient & Hessian

Consider the function

$$L : \mathbb{R}^d \to \mathbb{R} : L(\beta) = -\sum_{i=1}^{n} [y_i \log \sigma(x_i^\intercal \beta) + (1 - y_i) \log[1 - \sigma(x_i^\intercal \beta)]] + \lambda \beta^\intercal \beta \tag{1}$$

where $x_i \in \mathbb{R}^d$ is the ith row of a matrix $X \in \mathbb{R}^{n \times d}$, and $y \in \{0, 1\}^n$ is a *binary vector* (There is the word I never wanted to say! ;-))

The gradient is

$$\frac{\partial L(\beta)}{\partial \beta} = (p - y)^\intercal X + 2\lambda \beta^\intercal,\ p = \sigma(X\beta)$$

and the hessian

$$\nabla^2 L(\beta) = X^\intercal \,\text{diag}(p \circ (1 - p)) X + 2\lambda \mathbb{I}$$

(a) Implement the function, gradient and hessian. Generate a random matrix X (using @randn@) and random y (using @randi(2,n,1)-1@) and test using finite differences that ∂L is the Jacobian of L, and that $\nabla^2 L$ is the Jacobian of $\nabla L = \partial L^\intercal$.