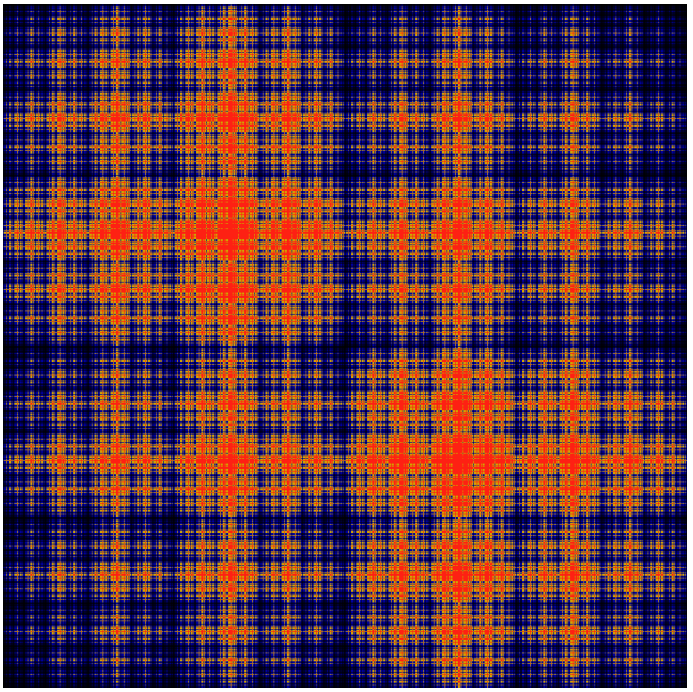
Applied Mathematics
and
Computing

Volume I



List of Contributors

J. Humpherys
Brigham Young University

J. Webb
Brigham Young University

R. Murray
Brigham Young University

J. West
University of Michigan

R. Grout
Brigham Young University

K. Finlinson
Brigham Young University

A. Zaitzeff
Brigham Young University

Preface

This lab manual is designed to accompany the textbook *Foundations of Applied Mathematics* by Dr. J. Humpherys.

Lab 19

Algorithms: Multivariate Finite Difference Schemes

Lesson Objective: *Understand how to compute numerically the Jacobian and Hessian of a function.*

The Jacobian is our generalization of the derivative in many dimensions. We can think of it, intuitively, as a tangent plane at a point. The Jacobian is of critical importance in a variety of areas, and we will use it in lab 23 to find zeros of multivariate functions.

Formally, the Jacobian of a function $f : \mathbb{R}^n \to \mathbb{R}^m$ is an $m \times n$ matrix. It is defined by the formula:

$$J_{i,j} = \frac{\partial f_i}{\partial x_j}$$

We can use finite difference approximations to find partial derivatives in the natural manner:

$$\frac{\partial f}{\partial x_i}(x) \approx \frac{f(x + he_i) - f(x)}{h}$$

where e_i is a unit vector in the ith coordinate (the direction x_i). Higher order approximations and centered and backwards differences follow by naturally extending this definition.

Problem 1 Write a function `Jacobian(func,inputs,h,o)` that accepts a function handle, inputs to the function, step size h, and option o. Based on the option have the function output the Jacobian using either a centered, forward or backward difference.

Test your function on the following $f : \mathbb{R}^2 \to \mathbb{R}^2$:

$$f(x_1, x_2) = \begin{pmatrix} e^{x_1} sin(x_2) + x_2^3 \\ 3x_2 - cos(x_1) \end{pmatrix}$$

Compare your `Jacobian` function against the analytically computed derivative on the square $[-1,1]\times[-1,1]$ using ten thousand grid points (100 per side). Which method is faster? What is the maximum error of your function? Make sure to test the different options of your function to maximize performance (including values for h, and different types of schemes).

Given a function from $\mathbb{R}^n \to \mathbb{R}$ sometimes the mixed partial derivatives are useful. In particular the mixed partials will be useful when we study optimization in Volume 2. This information is contained in the Hessian matrix, which is defined as

$$H_{i,j} = \frac{\partial^2 f}{\partial x_i \partial x_j}$$

We can use the following formula to approximate mixed partial derivatives

$$\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{f(x+(e_i+e_j)h) - f(x+(e_i-e_j)h) - f(x+(e_j-e_i)h) + f(x-(e_i+e_j)h)}{4h^2}$$

Problem 2 Write a Python function that numerically calculates the Hessian of a given function. Test it on the following function

$$f(x,y) = (1-x)^2 + 100(y-x^2)^2$$

This function is known as the Rosenbrock Banana function, or Rosenbrock's Valley. It is a common test function for optimization algorithms because it is non-convex and the global minimum is hard to find from certain starting points. A graph is shown in figure 19.1. Compare the output of your function with the analytic solution on the region $[-2,2]\times[0,2]$, using ten thousand points. What is the maximum error of your function?

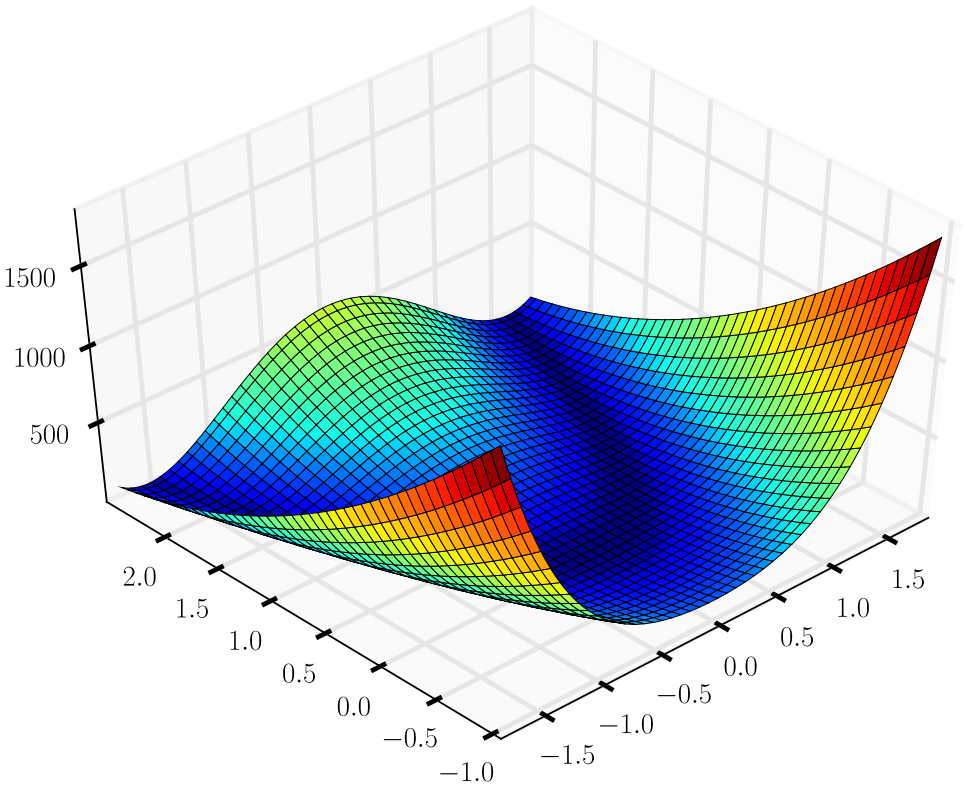


Figure 19.1: The Rosenbrock Banana Function, a common test function in optimization algorithms