Statistical Machine Learning
Finding Minima Algorithms

Horacio Gómez-Acevedo
Department of Biomedical Informatics
University of Arkansas for Medical Sciences

March 16, 2022

UAMS

Case $n = 1$

Let's suppose we have a real valued function which is smooth $f : \mathbb{R} \to \mathbb{R}$. Then, we can approximate the function in a vicinity of $x = c$ by the so-called Taylor expansion

$$f(x) = f(c) + \frac{df}{dx}(c)(x-c) + \frac{1}{2!}\frac{d^2 f}{dx^2}(c)(x-c)^2 + \frac{1}{3!}\frac{d^3 f}{dx^3}(c)(x-c)^3 + \cdots$$

Notice that when x is very close to c, then $(x-c)^p$ for $p \geq 3$ starts getting exceedingly small. Then, we write

$$f(x) \approx f(c) + \frac{df}{dx}(c)(x-c) + \frac{1}{2!}\frac{d^2 f}{dx^2}(c)(x-c)^2$$

If at the point $x = c$ the function reaches a (local) minimum, then $f'(c) = 0$.
Recall that in the threshold logic, once we receive the input, a decision must be made to *fire* or *suppress* the output.
Activation functions generalize the concept of activation given a specific input.

Case $n \geq 2$

Let's suppose we have a real valued function which is smooth $f : \mathbb{R}^n \to \mathbb{R}$. Then, we can approximate the function in a vicinity of $x = c$ by the so-called Taylor expansion

$$
\begin{aligned}
f(\mathbf{x}) &= f(\mathbf{c}) + \sum_i \frac{\partial f}{\partial x_i}(c)(x_i - c_i) + \frac{1}{2!}\sum_{i,j} \frac{\partial^2 f}{\partial x_i \partial x_j}(c)(x_i - c_i)(x_j - c_j) + \cdot \\
&\approx f(\mathbf{c}) + (\mathbf{x} - \mathbf{c})^T \nabla f(\mathbf{c}) + \frac{1}{2}(\mathbf{x} - \mathbf{c})^T H_f(\mathbf{c})(\mathbf{x} - \mathbf{c})
\end{aligned}
$$

Where $H_f(\mathbf{c})$ is the **Hessian matrix** defined as

$$
H_f(\mathbf{c}) = \begin{pmatrix}
\frac{\partial^2 f}{\partial x_1 \partial x_1}(\mathbf{c}) & \frac{\partial^2 f}{\partial x_1 \partial x_2}(\mathbf{c}) & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n}(\mathbf{c}) \\
\frac{\partial^2 f}{\partial x_2 \partial x_1}(\mathbf{c}) & \frac{\partial^2 f}{\partial x_2 \partial x_2}(\mathbf{c}) & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n}(\mathbf{c}) \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial^2 f}{\partial x_n \partial x_1}(\mathbf{c}) & \frac{\partial^2 f}{\partial x_n \partial x_2}(\mathbf{c}) & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_n}(\mathbf{c})
\end{pmatrix}
$$

Minimum Condition

For the case $n = 1$, the point $x = c$ if $f'(c) = 0$ and $f''(c) < 0$, then the function f has a **local minimum**.
For the case when $n \geq 2$, if $\nabla f(\mathbf{c}) = 0$ and that H_f satisfies for points close to $\mathbf{c}$

$$v^T H_f v > 0 \; \forall v \in \mathbb{R}^n - \{0\}$$

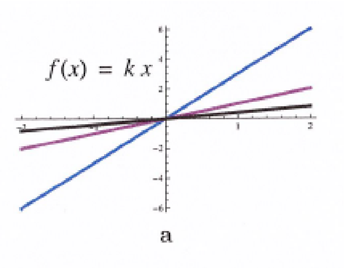
Then, $\mathbf{c}$ is a **local minimum** of f.

Linear Functions

These functions are used to model the firing rate of a neuron.

$$f(x) = kx \quad k \text{ is a positive constant}$$

Pros: It is continuous and differentiable.



References

Materials and some of the pictures are from (Calin, 2019).

Calin, O. (2019). *Deep Learning Architectures*. Springer Series in the Data Sciences. Springer. ISBN: 978-3-030-36723-7.

I have used some of the graphs by hacking TiKz code from StakExchange, Inkscape for more aesthetic plots and other old tricks of TEX