FTDS // Calculus // Partial Derivative

Hacktiv8 DS Curriculum Team

Phase 0 Learning Materials

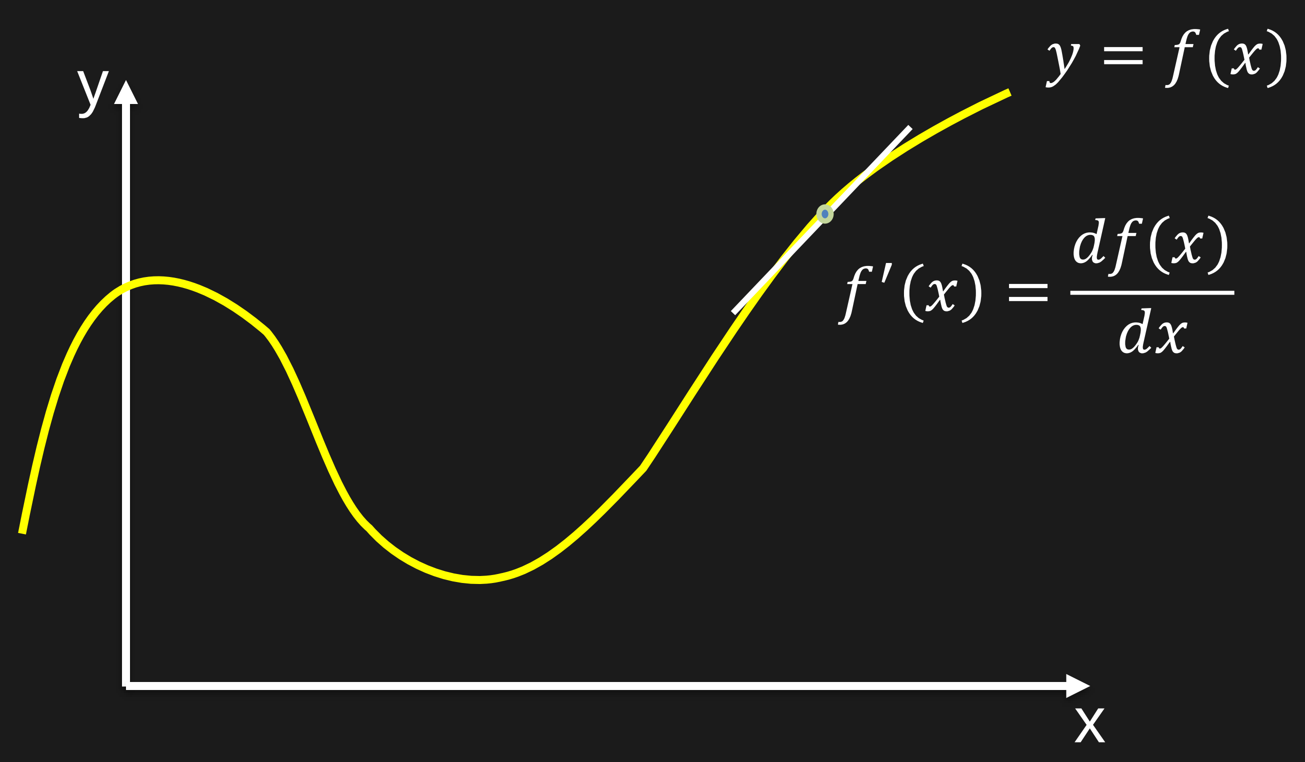
Hacktiv8 DS Curriculum Team

Contents

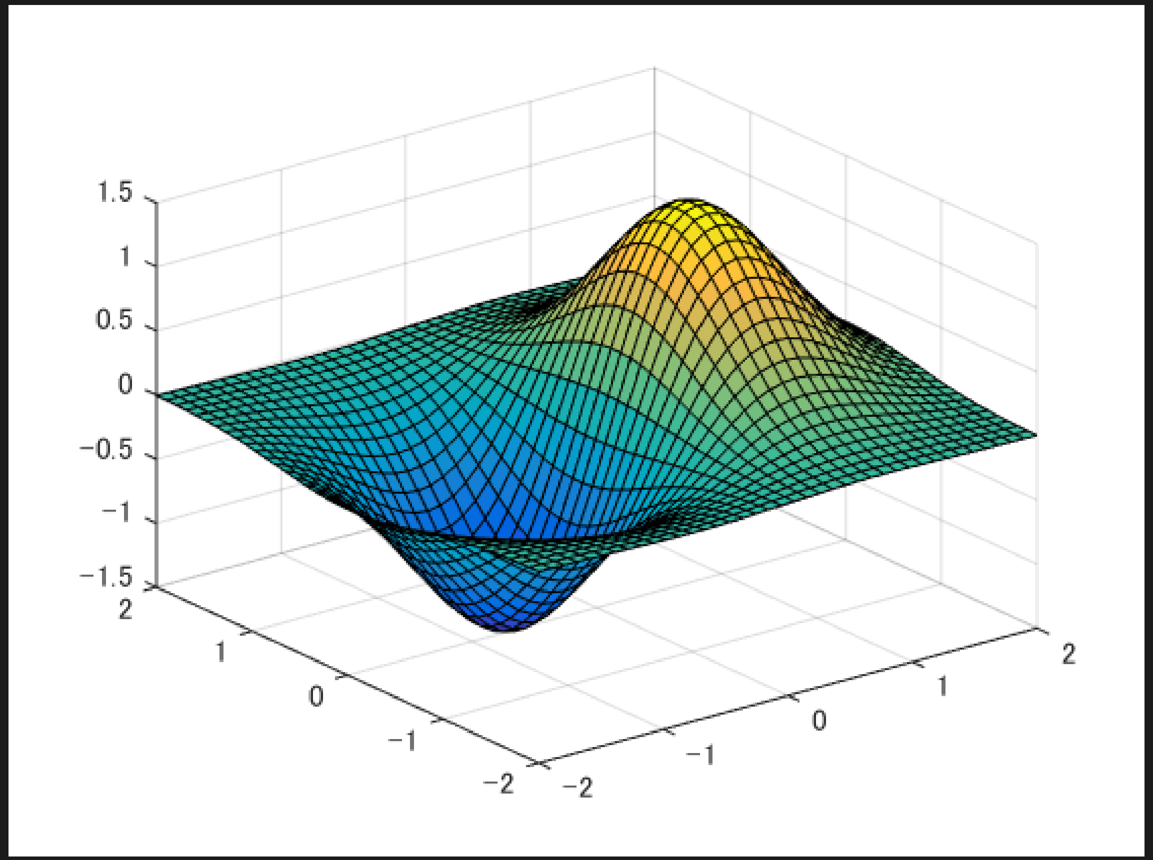
Objectives

- Basic understanding of Partial Derivative
- Basic understanding of Gradient, Gradient Descent, Jacobian, and Hessian
- Able to calculate the Partial Derivative of a function
- Implementation on Python

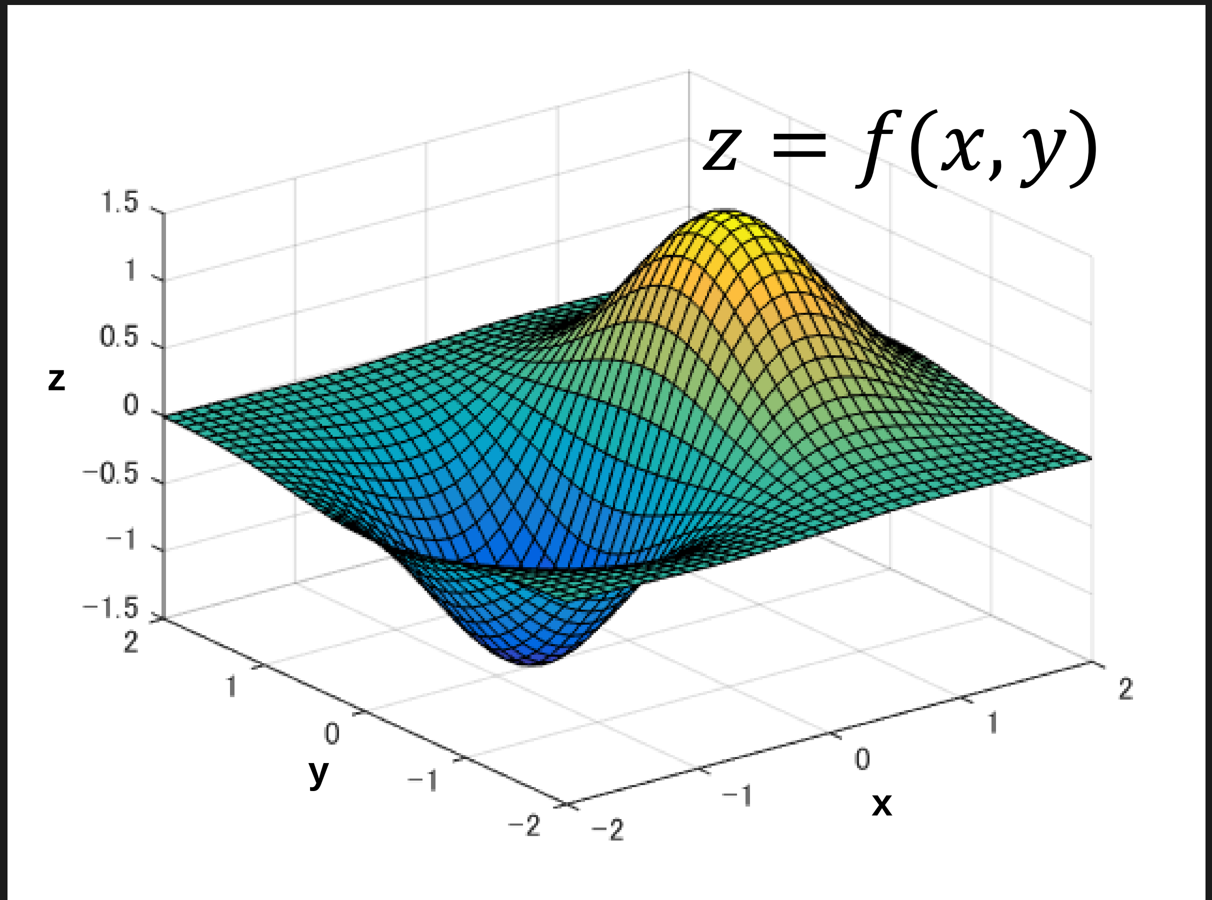
Partial Derivative works with multivariable function



In the prior lesson, we learn how to calculate the rate of change of a curve.



But how do we calculate the rate of change of a surface?



To measure a rate of change of a surface, we can calculate it based on each the axis direction. Let we have a surface z=f(x,y), the partial derivatives are:

$$\frac{\partial f(x,y)}{\partial x} \qquad \frac{\partial f(x,y)}{\partial y}$$

Example:

$$z = f(x, y) = xy^2 + x^3$$

$$\partial_x f = \frac{\partial f}{\partial x} = y^2 + 2x^2$$

$$\partial_y f = \frac{\partial f}{\partial y} = 2xy$$

$$z = f(x, y)$$

$$\frac{\partial^2 f(x,y)}{\partial x^2} = \frac{\partial}{\partial x}\left(\frac{\partial f(x,y)}{\partial x}\right)$$

$$\frac{\partial^2 f(x,y)}{\partial y^2} = \frac{\partial}{\partial y}\left(\frac{\partial f(x,y)}{\partial y}\right)$$

$$\frac{\partial f(x,y)}{\partial x \partial y} = \frac{\partial}{\partial x}\left(\frac{\partial f(x,y)}{\partial y}\right)$$

$$\frac{\partial f(x,y)}{\partial y \partial x} = \frac{\partial}{\partial y}\left(\frac{\partial f(x,y)}{\partial x}\right)$$

Example:

$$z = f(x, y) = xy^2 + x^3$$

$$\partial_x f = \frac{\partial f}{\partial x} = y^2 + 2x^2$$

$$\partial_y f = \frac{\partial f}{\partial y} = 2xy$$

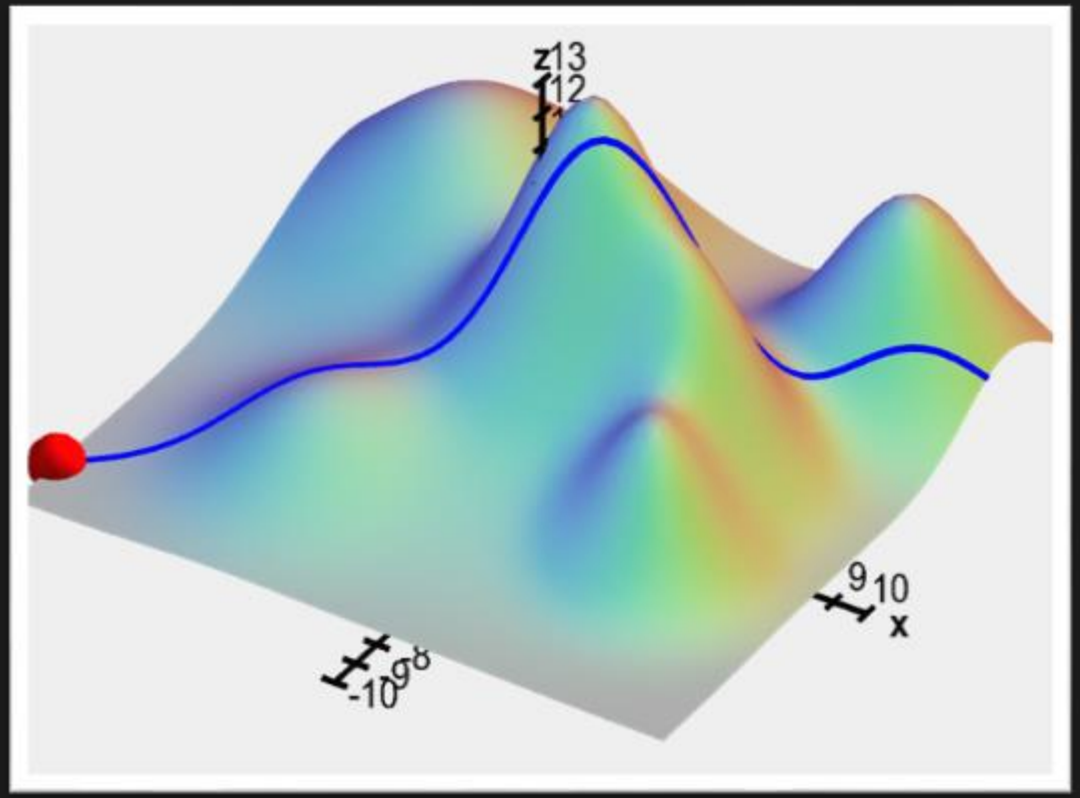
$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x}(\partial_x f) = 4x$$

$$\frac{\partial^2 f}{\partial y \partial x} = \frac{\partial}{\partial y}(\partial_x f) = 2y$$

$$\frac{\partial^2 f}{\partial y^2} = \frac{\partial}{\partial y}(\partial_y f) = 2x$$

$$\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial}{\partial x}(\partial_y f) = 2y$$

When you calculate the partial derivative of a multivariate function, you will get more than one results. Gradient simply store your results into a vector. Yet mathematicaly, gradient is a change rate on the surface.



$$\nabla f(x,y) = \begin{pmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \end{pmatrix} \qquad \nabla f(x,y,z) = \begin{pmatrix} \dfrac{\partial f}{\partial x} \\ \dfrac{\partial f}{\partial y} \\ \dfrac{\partial f}{\partial z} \end{pmatrix}$$

Example:

$$z = f(x,y) = xy^2 + x^3$$

$$\partial_x f = \frac{\partial f}{\partial x} = y^2 + 2x^2$$

$$\partial_y f = \frac{\partial f}{\partial y} = 2xy$$

$$\nabla f(x,y) = \begin{pmatrix} y^2 + 2x^2 \\ 2xy \end{pmatrix}$$

Sometimes you will have a multi-dimensional function, also the input and the output are multi-dimensional. So, you will work with Jacobian. In brief, Jacobian is a matrix that store the partial derivative of the functions.

$$J_f = J_{i,j} = \frac{\partial}{\partial x_j} f(x)_i = \begin{bmatrix} \nabla f_1 \\ \nabla f_2 \\ \vdots \\ \nabla f_i \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_j} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_j} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_i}{\partial x_1} & \frac{\partial f_i}{\partial x_2} & \cdots & \frac{\partial f_i}{\partial x_j} \end{bmatrix}$$

Example:

$$f(x,y) = \begin{bmatrix} x^2y + y \\ 2xy - 2 \end{bmatrix}$$

$$J_f = \begin{bmatrix} \nabla f_1 \\ \nabla f_2 \end{bmatrix} = \begin{bmatrix} \dfrac{\partial f_1}{\partial x} & \dfrac{\partial f_1}{\partial y} \\ \dfrac{\partial f_2}{\partial x} & \dfrac{\partial f_2}{\partial y} \end{bmatrix} = \begin{bmatrix} 2xy & x^2 + 1 \\ 2y & 2x \end{bmatrix}$$

Like gradient for first order partial derivative, we can also store the second partial derivative results in a matrix. The matrix is called Hessian. Hessian also represents second order gradient.

$$H_f = \nabla^2 f = H_{i,j} = \frac{\partial^2}{\partial x_i x_j} f(x)_i = \begin{bmatrix} \frac{\partial^2 f_1}{\partial x_1^2} & \frac{\partial^2 f_1}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f_1}{\partial x_1 \partial x_j} \\ \frac{\partial^2 f_2}{\partial x_2 \partial x_1} & \frac{\partial^2 f_2}{\partial x_2^2} & \cdots & \frac{\partial f_2}{\partial x_j} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f_i}{\partial x_i \partial x_1} & \frac{\partial^2 f_i}{\partial x_i \partial x_2} & \cdots & \frac{\partial^2 f_i}{\partial x_j^2} \end{bmatrix}$$

Example:

$$z = f(x, y) = xy^2 + x^3$$

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial}{\partial x}(\partial_x f) = 4x$$

$$\frac{\partial^2 f}{\partial x \partial y} = \frac{\partial}{\partial x}(\partial_y f) = 2y$$

$$\frac{\partial^2 f}{\partial y^2} = \frac{\partial}{\partial y}(\partial_y f) = 2x$$

$$\frac{\partial^2 f}{\partial y \partial x} = \frac{\partial}{\partial y}(\partial_x f) = 2y$$

$$H_f = \nabla^2 f(x, y) = \begin{bmatrix} \frac{\partial}{\partial x}(\partial_x f) & \frac{\partial}{\partial x}(\partial_y f) \\ \frac{\partial}{\partial y}(\partial_x f) & \frac{\partial}{\partial y}(\partial_y f) \end{bmatrix} = \begin{bmatrix} 4x & 2y \\ 2y & 2x \end{bmatrix}$$

Partial Derivative on Code // Symbolic// First Order

$$f(x) = 4xy + x\sin z + x^3 + z^8 y$$

This method the input or the output as symbols even also the function.

```
import sympy as sy

x,y,z = sy.symbols('x y z')
f = 4*x*y + x*sy.sin(z) + x**3 + z**8*y
```

`sy.diff(f,x)` ***Output:*** $3x^2 + 4y + \sin z$

`sy.diff(f,y)` ***Output:*** $4x + z^8$

`sy.diff(f,z)` ***Output:*** $x\cos z + 8yz^7$

Partial Derivative on Code // Symbolic// Gradient

$$f(x) = 4xy + x\sin z + x^3 + z^8 y$$

This method the input or the output as symbols even also the function.

```
import sympy as sy
from sympy.tensor.array import derive_by_array

x,y,z = sy.symbols('x y z')
f = 4*x*y + x*sy.sin(z) + x**3 + z**8*y

derive_by_array(f, (x,y,z))
```

Output:

$$\begin{pmatrix} 3x^2 + 4y + \sin z \\ 4x + z^8 \\ x\cos z + 8yz^7 \end{pmatrix}$$

Partial Derivative on Code // Symbolic// Jacobian

$$f(x,y) = \begin{bmatrix} x^2y + y \\ 2xy - 2 \end{bmatrix}$$

This method the input or the output as symbols even also the function.

```
import sympy as sy

x,y,z = sy.symbols(‘x y z’)

f =sy.Matrix([x**y+y, 2*x*y-2])

sy.hessian(f, (x,y))
```

Output:

$$\begin{bmatrix} 2xy & x^2+1 \\ 2y & 2x \end{bmatrix}$$

Partial Derivative on Code // Symbolic// Hessian

$$z = f(x, y) = xy^2 + x^3$$

This method the input or the output as symbols even also the function.

```
import sympy as sy

x,y,z = sy.symbols('x y')

f =sy.Matrix([x**y+y, 2*x*y-2])
X=sy.Matrix([x,y])

f.jacobian(X)
```

Output:

$$\begin{bmatrix} 4x & 2y \\ 2y & 2x \end{bmatrix}$$

Partial Derivative on Code // Numeric//

$$z = f(x, y) = xy^2 + x^3$$

Same as the derivative, we can use numpy gradient to compute partial derivative numerically. But, we need to define a matrix or tensor to store the f(x,y) values.

```
import numpy as np

def f(x,y):
        return x**2*y+2*x**3*y+y**4

x=np.linspace(1,10)
y=np.linspace(1,10)

z=np.array([[f(i,j) for i in x] for j in y])

dx,dy=np.gradient(z)
```

Hacktiv8 DS
Curriculum
Team

External References

Colab Link — Visit Here