Gradient Descent

© Malay K. Das, 210 Southern Lab, ph-7359, mkdas@iitk.ac.in

Office hours: W 1030-1130, SL-210

Previously:

Steepest descent: theory

Today:

Gradient descent, Newton's method: example

HW due:

September 03, 2024

Computing quiz:

September 04, 2024

Malay K. Das, mkdas@iitk.ac.in

Unconstrained Optimization

We wish to minimize $y = f(\mathbf{x})$ $\quad \mathbf{x} \in \mathbb{R}^n; y \in \mathbb{R}$

search for minimum — search in calculated direction with optimum step **(Gradient Descent Methods)**

- steepest descent, Newton's method etc.

random search **(Metaheuristic Methods)**

- simulated annealing, genetic algorithm etc.

Both the above approaches include variety of techniques

1D optimization

Random Search

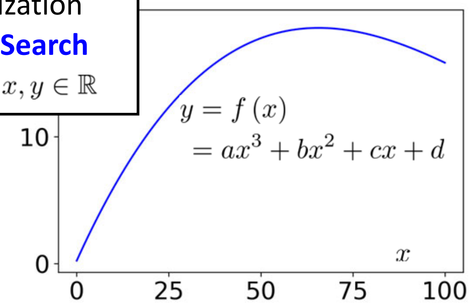
$y = f(x) \qquad x, y \in \mathbb{R}$

```
import numpy as np

n=10000             #no. of trials
a =1.918e-5;b =-6.77e-3
c =.6413;d =.2302
f = lambda x: a*x**3+b*x**2+c*x+d

x1=10.;x2=100.
l=x2-x1             #interval
f2=f(x1)

for i in range(n):
    x3=x1+l*np.random.rand()
    f3=f(x3)
    if f3>f2:
        xstar=x3
        ystar=f3
        f2=f3
    print(xstar,ystar)
```



$a = 1.918 \times 10^{-5} \quad b = -6.77 \times 10^{-3}$

$c = 0.6413 \quad d = 0.2302$

Program to find maximum of $y = f(x) = ax^3 + bx^2 + cx + d$ → $\boxed{18.58}$ (same as analytical)

The program may be improved further to automate the number of trials

Malay K. Das, mkdas@iitk.ac.in

Gradient Descent Methods: methods using gradient information for optimization

To find $\mathbf{x}_\star = \arg\min_{\mathbf{x}} f(\mathbf{x})$ starting from an initial guess $\mathbf{x}^{(0)}$

we iteratively update $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + h_k \mathbf{d}_k$

such that $f\left(\mathbf{x}^{(k+1)}\right) < f\left(x^{(k)}\right)$

$\mathbf{d}_k$: descent (search) direction

h_k : step size (+ve scalar)

To select a good $\mathbf{d}$, we expand in (multivariable) Taylor series

$$f\left(\mathbf{x}^{(k+1)}\right) = f\left(\mathbf{x}^{(k)}\right) + h_k \mathbf{d}_k^T \nabla f\left(\mathbf{x}^{(k)}\right) + \mathcal{O}\left(h_k^2\right)$$

neglecting higher-order terms, change in the function f $\quad \delta f \approx h_k \mathbf{d}_k^T \nabla f\left(\mathbf{x}^{(k)}\right)$

for minimization of f, we enforce $\quad \delta f < 0 \quad \boxed{\Rightarrow \mathbf{d}_k^T \nabla f\left(\mathbf{x}^{(k)}\right) < 0}$

fundamental theory gradient descent

Different choice of $\mathbf{d}_k, h_k$ leads to different methods

directional derivative in the direction $\mathbf{d}_k$

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

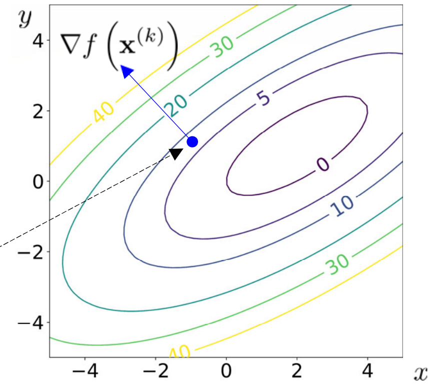
$$\nabla f = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\[2ex] \dfrac{\partial f}{\partial y} \end{bmatrix} = \begin{bmatrix} 2x - 2y - 2 \\ 4y - 2x \end{bmatrix}$$

at k-th iteration

$$\mathbf{x}^{(k)} = \begin{bmatrix} -1 & 1 \end{bmatrix}^T$$

$$f(-1,1) = 7$$

$$\nabla f\left(\mathbf{x}^{(k)}\right) = 6\begin{bmatrix} -1 & 1 \end{bmatrix}^T$$



$f(x,y)$ contour

for descent direction $\mathbf{d}_k \qquad \mathbf{d}_k^T \nabla f\left(\mathbf{x}^{(k)}\right) < 0$

let us assume $\mathbf{d}_k = \begin{bmatrix} d_1 & d_2 \end{bmatrix}^T \Rightarrow \mathbf{d}_k^T \nabla f\left(\mathbf{x}^{(k)}\right) = 6(d_2 - d_1)$

For $d_2 < d_1 \qquad \mathbf{d}_k^T \nabla f\left(\mathbf{x}^{(k)}\right) < 0$ and, therefore, $f\left(\mathbf{x}^{(k+1)}\right) < f\left(x^{(k)}\right)$

Therefore, all $\mathbf{d}_k = \begin{bmatrix} d_1 & d_2 \end{bmatrix}^T$ with $d_2 < d_1$ are valid descent directions

Malay K. Das, mkdas@iitk.ac.in

Steepest Descent method

$\mathbf{d}_k^T \nabla f\left(\mathbf{x}^{(k)}\right) < 0 \Rightarrow \|\mathbf{d}_k\| \left\| \nabla f\left(\mathbf{x}^{(k)}\right) \right\| \cos\theta < 0 \qquad \theta$ is the angle between $\mathbf{d}_k, \nabla f\left(\mathbf{x}^{(k)}\right)$

$\Rightarrow \cos\theta < 0$ minimum $\cos\theta = \cos(\pi) = -1$ (easy to visualize in 2D)

in steepest descent method, the angle between $\mathbf{d}_k$ and $\nabla f\left(\mathbf{x}^{(k)}\right)$ is π

$$\mathbf{x}_\star = \arg\min_{\mathbf{x}} f(\mathbf{x})$$

$$\mathbf{x}^{(k+1)} = x^{(k)} + h_k \mathbf{d}_k$$

$$\mathbf{d}_k^T \nabla f\left(\mathbf{x}^{(k)}\right) < 0 \qquad h_k > 0$$

Thus $\boxed{\mathbf{d}_k = -\nabla f\left(\mathbf{x}^{(k)}\right)}$

The 'rate of minimization' is at its highest at this direction

Rate of minimization refers to the reduction in the value of the function over unit distance

Now h_k may be obtained from any suitable **Line Search** technique

Steepest Descent method

$$\mathbf{x}^{(k+1)} = x^{(k)} + h_k \mathbf{d}_k \quad \mathbf{d}_k = -\nabla f\left(\mathbf{x}^{(k)}\right)$$

$$\mathbf{x}_\star = \arg\min_{\mathbf{x}} f\left(\mathbf{x}\right)$$

$\mathbf{d}_k$: descent (search) direction

h_k : step size (+ve scalar)

Line Search to find h_k

since we now know $x^{(k)}, \mathbf{d}_k$

$$f\left(\mathbf{x}^{(k+1)}\right) = f\left(x^{(k)} + h_k \mathbf{d}_k\right) = \phi\left(h_k\right)$$ is a function of h_k only

We define $h_k = \arg\min_h \phi\left(h\right)$ and minimize $\phi\left(h\right)$ using 1D optimization techniques

When minimization of $\phi\left(h\right)$ becomes computationally expensive, alternatives methods are used

alternatives methods partially minimizes $\phi\left(h\right)$

Malay K. Das, mkdas@iitk.ac.in

Example: minimize $f\left(x, y\right) = x^2 + 2y^2 - 2xy - 2x$

initial guess $\mathbf{x}^{(0)} = \left[-1 \quad 1\right]^T$ $\quad \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

steepest descent

$$\mathbf{d}_0 = -\nabla f\left(\mathbf{x}^{(0)}\right) = -6\left[-1 \quad 1\right]^T \qquad \mathbf{x}^{(k+1)} = x^{(k)} + h_k \mathbf{d}_k$$

$$\phi\left(h\right) = f\left(\mathbf{x}^{(0)} + h\mathbf{d}_0\right) \qquad \mathbf{x}^{(0)} + h\mathbf{d}_0 = \left(1 - 6h\right)\begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

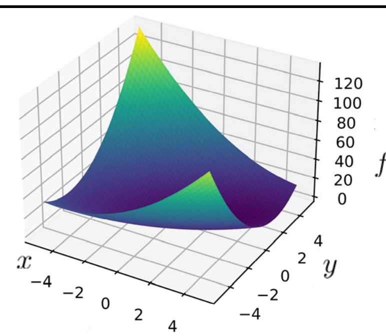
$$= 5\left(1 - 6h\right)^2 + 2\left(1 - 6h\right)$$

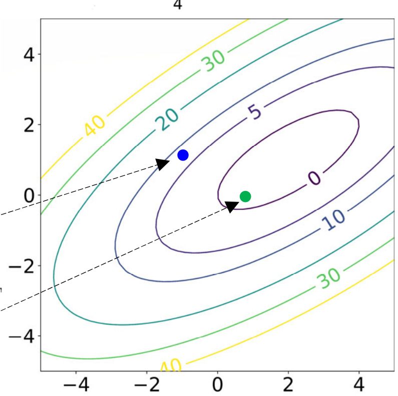
$$\phi'\left(h\right) = -60\left(1 - 6h\right) - 12 \qquad \phi'\left(h = h_0\right) = 0 \Rightarrow h_0 = 0.2$$

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + h_0 \mathbf{d}_0 = \left[0.2 \quad -0.2\right]^T$$

$$f\left(\mathbf{x}^{(0)}\right) = 7 \qquad f\left(\mathbf{x}^{(1)}\right) = -0.36$$

solution converges when $\left\|\mathbf{d}_k\right\| < \epsilon$ ← tolerance





$\mathbf{x}^{(0)} = \left[-1 \quad 1\right]^T$

$\mathbf{x}^{(1)} = \left[0.2 \quad -0.2\right]^T$

Steepest Descent Method: Summary

To find $\mathbf{x}_\star = \arg\min_{\mathbf{x}} f(\mathbf{x})$ starting from an initial guess $\mathbf{x}^{(0)}$

we iteratively update $\mathbf{x}^{(k+1)} = x^{(k)} + h_k \mathbf{d}_k$

$\mathbf{d}_k$: descent (search) direction

h_k : step size (+ve scalar)

in steepest descent method $\mathbf{d}_k = -\nabla f\left(\mathbf{x}^{(k)}\right)$

h_k may be obtained by various **Line Search** techniques, such as

$h_k = \arg\min_{h} \phi(h)$ where $\phi(h) = f\left(\mathbf{x}^{(k)} + h\mathbf{d}_k\right)$

Such step size calculations are difficult to implement, sometimes constant step size is used in computer program

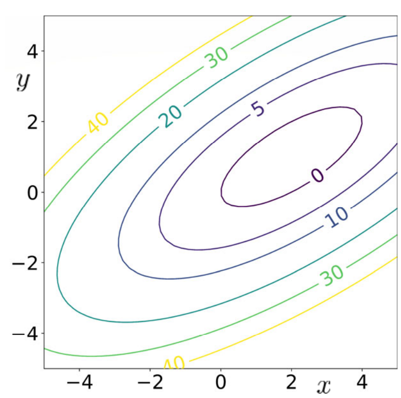
Malay K. Das, mkdas@iitk.ac.in

Exercise

Minimize $f(x, y) = x^2 + 2y^2 - 2xy - 2x$

$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + h_k \mathbf{d}_k$ $\quad \mathbf{d}_k = -\nabla f\left(\mathbf{x}^{(k)}\right)$ $\quad \mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$

$$\nabla f = \begin{bmatrix} \dfrac{\partial f}{\partial x} \\[2ex] \dfrac{\partial f}{\partial y} \end{bmatrix} = \begin{bmatrix} 2x - 2y - 2 \\ 4y - 2x \end{bmatrix}$$



$f(x, y)$ contour

Let's first try using a constant step size $h_k = 0.01$

we know the analytical solution

$$\min f(x, y) = -2 \quad \text{at} \quad \begin{bmatrix} x_\star \\ y_\star \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

```python
import numpy as np

maxit=10000;epsilon=1.e-6;stepsize=.01
objective_function= lambda x: x[0]**2 + 2*x[1]**2-2*x[0]*x[1]-2*x[0]
gradient_function=  lambda x: np.array([2 * x[0]-2*x[1]-2, 4 * x[1]-2*x[0]])

x=np.array([-1.,1.])        #initial guess
for i in range(maxit):
   gradient=gradient_function(x);b=np.linalg.norm(gradient)
   if b < epsilon:
      break
   x  -=  stepsize * gradient
   print(i,x,b)

minimum_value = objective_function(x)
print("Minimum value:", minimum_value)
print("Minimum location:", x)
```

constant step size

objective function

gradient of objective function

norm of gradient; used as stopping criterion

Program output: Minimum value: -1.999999, Minimum location: [1.999999,.999999]

Analytical: Minimum value: -2, Minimum location: [2,1]

Malay K. Das, mkdas@iitk.ac.in

Backtracking Line Search: Armijo condition

we enforce $f\left(x^{(k)}+h\mathbf{d}_k\right) < f\left(x^{(k)}\right)$ but do not minimize $f\left(x^{(k)}+h\mathbf{d}_k\right)$

Armijo condition suggests

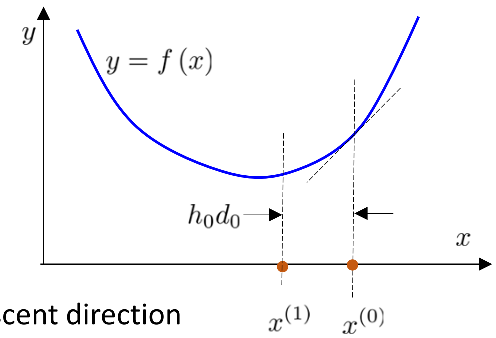
$$f\left(x^{(k)}+h\mathbf{d}_k\right) \le f\left(x^{(k)}\right) + \beta h \mathbf{d}_k^T \nabla f\left(x^{(k)}\right)$$ for some constant $\beta \in (0,1)$

With guess small values of $\beta \in (0,1)$ $\tau \in (0,1)$ initial guess of $h=1$ (usually)

1. check the Armijo condition is satisfied or not
2. if not satisfied $h \leftarrow \tau h$ $\tau \in (0,1)$
3. continue until the Armijo condition is satisfied

typical value of $\tau = 0.5$

Recall 1D Newton-Raphson Method



To minimize $f(x)$

we begin with an initial guess $x^{(0)}$

$x^{(1)} = x^{(0)} + h_0 d_0 \qquad d_0 = -f'\left(x^{(0)}\right)$

The step size h_0 hopefully minimizes $f(x)$

$\Rightarrow f'\left(x = x^{(1)}\right) = 0$

d_0 : descent direction

h_0 : step size

Expanding in Taylor series

$$f'\left(x = x^{(1)}\right) = 0 = f'\left(x^{(0)} + h_0 d_0\right) = f'\left(x^{(0)}\right) + h_0 d_0 f''\left(x^{(0)}\right) + \cdots$$

$$\Rightarrow h_0 = -\frac{f'\left(x^{(0)}\right)}{d_0 f''\left(x^{(0)}\right)} = \frac{1}{f''\left(x^{(0)}\right)}$$

$$\Rightarrow x^{(1)} = x^{(0)} - \frac{f'\left(x^{(0)}\right)}{f''\left(x^{(0)}\right)}$$

in general

$$\boxed{x^{(k+1)} = x^{(k)} - \left(\frac{f'}{f''}\right)_{x = x^{(k)}}}$$

Multidimensional Newton's (or Newton-Raphson) Method

Extension of 1D Newton-Raphson method

Recall 1D $x^{(k+1)} = x^{(k)} - \left(\frac{f'}{f''}\right)_{x = x^{(k)}}$

Following the same argument

$$\boxed{\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \left[\mathbf{H}f\left(\mathbf{x}^{(k)}\right)\right]^{-1} \nabla f\left(\mathbf{x}^{(k)}\right)}$$

$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + h_k \mathbf{d}_k$

$\mathbf{d}_k$: descent direction

h_k : step size

As in 1D technique, the method may not converge for certain initial guesses, and for highly nonlinear functions

As in 1D technique, when converges, shows rapid convergence otherwise

To improve convergence, the method is often modified using Armijo condition

Backtracking Line Search

Armijo condition suggests

$$f\left(x^{(k+1)}\right)-f\left(x^{(k)}\right)\leq \beta h\left[\mathbf{H}f\left(\mathbf{x}^{(k)}\right)\right]^{-1}\nabla f\left(\mathbf{x}^{(k)}\right)$$

for some constant $\beta\in(0,1)$

With a guess small values of $\beta\in(0,1)$ $\tau\in(0,1)$ initial guess of $h=1$

1. check the Armijo condition is satisfied or not
2. if not satisfied $h\leftarrow\tau h$ $\tau\in(0,1)$ typical value of $\tau=0.5$
3. continue until the Armijo condition is satisfied

Once Armijo condition is satisfied, update

$$\boxed{\mathbf{x}^{(k+1)}=\mathbf{x}^{(k)}-h\left[\mathbf{H}f\left(\mathbf{x}^{(k)}\right)\right]^{-1}\nabla f\left(\mathbf{x}^{(k)}\right)}$$

Malay K. Das, mkdas@iitk.ac.in

Exercise

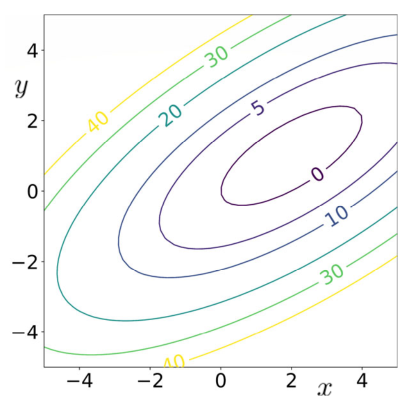
Minimize $f(x,y)=x^2+2y^2-2xy-2x$ $\quad\mathbf{x}=\begin{bmatrix}x\\y\end{bmatrix}$

$$\mathbf{x}^{(k+1)}=\mathbf{x}^{(k)}-h\left[\mathbf{H}f\left(\mathbf{x}^{(k)}\right)\right]^{-1}\nabla f\left(\mathbf{x}^{(k)}\right)$$

$$\nabla f=\begin{bmatrix}\dfrac{\partial f}{\partial x} & \dfrac{\partial f}{\partial y}\end{bmatrix}^T=\begin{bmatrix}2x-2y-2\\4y-2x\end{bmatrix}\qquad \mathbf{H}f=\begin{bmatrix}\dfrac{\partial^2 f}{\partial x^2} & \dfrac{\partial^2 f}{\partial x\partial y}\\[2mm] \dfrac{\partial^2 f}{\partial x\partial y} & \dfrac{\partial^2 f}{\partial y^2}\end{bmatrix}=\begin{bmatrix}2 & -2\\-2 & 4\end{bmatrix}$$

$$(\mathbf{H}f)^{-1}=\frac{1}{\det(\mathbf{H}f)}\begin{bmatrix}4 & 2\\2 & 2\end{bmatrix}=\frac{1}{12}\begin{bmatrix}4 & 2\\2 & 2\end{bmatrix}$$

$$(\mathbf{H}f)^{-1}\nabla f=\frac{1}{12}\begin{bmatrix}4 & 2\\2 & 2\end{bmatrix}\begin{bmatrix}2x-2y-2\\4y-2x\end{bmatrix}=\frac{1}{3}\begin{bmatrix}x-2\\y-1\end{bmatrix}$$



$f(x,y)$ contour

analytical solution

$\min f(x,y)=-2$

at $\begin{bmatrix}x_\star\\y_\star\end{bmatrix}=\begin{bmatrix}2\\1\end{bmatrix}$