

```
In [13]: #####
#####
### Name : 01_20200676_01000 #####
### Matriculation : 2201000676 #####
```

```
In [8]: from matplotlib import pyplot as plt
import numpy as np
from matplotlib.ticker import MaxLocator
from itertools import product
```

```
In [33]: def Rosenbrock_f(x,y):
return (1 - x)**2 + 100*(y - x**2)**2

def Grad_Rosenbrock_f(x,y):
g1 = -400*x*y + 400*x**2 + 2*x -2
g2 = 200*x*y -200*y**2
return np.array([g1,g2])

def Hessian_Rosenbrock_f(x,y):
h11 = -400*y + 1200*x**2 + 2
h12 = -400 * x
h21 = -400 * x
h22 = 200
return np.array([[h11,h12],[h21,h22]])
```

```
In [9]: def Rosenbrock(X):
x, y = X
return (1 - x)**2 + 100*(y - x**2)**2

def Grad_Rosenbrock(X):
x, y = X
return np.array([
-400*x*y + 400*x**2 + 2*x -2,
200*x*y -200*x**2
])

def Hessian_Rosenbrock(X):
x, y = X
return np.matrix([
[-400*y + 1200*x**2 + 2, -400*x],
[-400*x, 200]
])
```

```
In [45]: x = np.arange(-5, 5, 0.025)
y = np.arange(-5, 5, 0.025)
X, Y = np.meshgrid(x, y)
Z = np.zeros(X.shape)
mesh_size = range(len(X))
for i, j in product(mesh_size, mesh_size):
x_coor = X[i][j]
y_coor = Y[i][j]
Z[i][j] = Rosenbrock(np.array([x_coor, y_coor]))

fig = plt.figure(figsize=(6,6))
ax = fig.gca(projection='3d')
ax.set_title('Rosenbrock Function')
ax.set_xlabel('x [s]')
ax.set_ylabel('y [s]')
ax.set_zlabel('f(x, y, z)')
ax.plot_surface(X, Y, Z, cmap='viridis')
plt.tight_layout()
plt.show()
```



```
In [46]: def ArmijoLineSearch(f, xk, pk, gfk, phi0, alpha0, rho=0.5, ci=1e-4):
"""Minimize over alpha, the function 'f(xk + alpha*pk)'.
alpha > 0 is assumed to be a descent direction.

Parameters
-----
f : callable
    Function to be minimized.
xk : array
    Current point.
pk : array
    Search direction.
gfk : array
    Gradient of 'f' at point 'xk'.
phi0 : float
    Value of 'f' at point 'xk'.
alpha0 : scalar
    Value of 'alpha' at the start of the optimization.
rho : float, optional
    Value of alpha shrinkage factor.
ci : float, optional
    Value to control stopping criterion.

Returns
-----
alpha : scalar
    Value of 'alpha' at the end of the optimisation.
phi : float
    Value of 'f' at the new point 'x_{k+1}'.
"""
derphi0 = np.dot(gfk, pk)
phi_a0 = f(xk + alpha0*pk)

while not phi_a0 <= phi0 + ci*alpha0*derphi0:
    alpha0 = alpha0 * rho
    phi_a0 = f(xk + alpha0*pk)

return alpha0, phi_a0
```

```
In [55]: def GradientDescent(f, f_grad, init, alpha=1, tol=1e-3, max_iter=10000):
"""Gradient descent method for unconstrained optimization problem.
given a starting point x ∈ R^n,
repeat
    1. Define direction: p := -∇f(x).
    2. Line search. Choose step length α using Armijo Line Search.
    3. Update: x = x + αp.
until stopping criterion is satisfied.

Parameters
-----
f : callable
    Function to be minimized.
f_grad : callable
    The first derivative of f.
init : array
    initial value of x.
alpha : scalar, optional
    the initial value of steplength.
tol : float, optional
    tolerance for the norm of f_grad.
max_iter : integer, optional
    maximum number of steps.

Returns
-----
xs : array
    x in the learning path
ys : array
    f(x) in the learning path
"""
# initialize x, f(x), and f'(x)
xk = init
fk = f(xk)
gfk = f_grad(xk)
gfk_norm = np.linalg.norm(gfk)
# initialize number of steps, save x and f(x)
num_iter = 0
curve_x = [xk]
curve_y = [fk]
print('Initial condition: y = {:.4f}, x = {}'.format(fk, xk))
# take steps
while gfk_norm > tol and num_iter < max_iter:
    # determine direction
    pk = -gfk
    gfk = np.linalg.inv(Hessian_Rosenbrock_f(x,y)) @ Grad_Rosenbrock_f(x,y)
    # calculate new x, f(x), and f'(x)
    alpha, fk = ArmijoLineSearch(f, xk, pk, gfk, fk, alpha0=alpha)
    xk = xk + alpha * pk
    gfk = f_grad(xk)
    gfk_norm = np.linalg.norm(gfk)
    # increase number of steps by 1, save new x and f(x)
    num_iter += 1
    curve_x.append(xk)
    curve_y.append(fk)
    print('Iteration: {} \t y = {:.4f}, x = {}'.format(num_iter, fk, xk, gfk_norm))
# print results
if num_iter == max_iter:
    print('\nGradient descent does not converge.')
else:
    print('\nSolution: \t y = {:.4f}, x = {}'.format(fk, xk))

return np.array(curve_x), np.array(curve_y)
```

```
In [56]: def plot(xs, ys):
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 6))
plt.suptitle('Gradient Descent Method')

ax1.plot(xs[:,0], xs[:,1], linestyle='--', marker='o', color='orange')
ax1.plot(xs[:,1:0], xs[:,1:1], 'ro')
ax1.set(
    title='Path During Optimization Process',
    xlabel='x1',
    ylabel='x2'
)
CS = ax1.contour(X, Y, Z)
ax1.clabel(CS, fontsize='smaller', fmt='%1.2f')
ax1.axis('square')

ax2.plot(ys, linestyle='--', marker='o', color='orange')
ax2.plot(len(ys)-1, ys[-1], 'ro')
ax2.set(
    title='Objective Function Value During Optimization Process',
    xlabel='Iterations',
    ylabel='Objective Function Value'
)
ax2.legend(['Armijo line search algorithm'])

plt.tight_layout()
plt.show()
```

```
In [57]: x0 = np.array([-1.2, 1])
x0, ys = GradientDescent(Rosenbrock, Grad_Rosenbrock, init=x0)
plot(xs, ys)
```





Iteration: 1	Y	4.1011,	[+0.8945313,	0.1093975,	gradient = 4.3895
Iteration: 2	Y	4.1194,	[+0.1268931,	0.0503668,	gradient = 1.7749
Iteration: 3	Y	4.1194,	[+0.0234587,	0.0474509,	gradient = 1.7749
Iteration: 4	Y	4.1126,	[+0.0236983,	0.1043468,	gradient = 1.7749
Iteration: 5	Y	4.1055,	[+0.0236983,	0.0503668,	gradient = 1.7749
Iteration: 6	Y	4.1055,	[+0.0236983,	0.0503668,	gradient = 1.7749
Iteration: 7	Y	4.1034,	[+0.0241039,	0.0567719,	gradient = 1.7761
Iteration: 8	Y	4.1034,	[+0.0241039,	0.0567719,	gradient = 1.7761
Iteration: 9	Y	4.0972,	[+0.0258238,	0.1053649,	gradient = 1.7773
Iteration: 10	Y	4.0972,	[+0.0258238,	0.1053649,	gradient = 1.7773
Iteration: 11	Y	4.0934,	[+0.0218137,	0.0502629,	gradient = 1.7773
Iteration: 12	Y	4.0934,	[+0.0218137,	0.0502629,	gradient = 1.7773
Iteration: 13	Y	4.0880,	[+0.0202923,	0.0489765,	gradient = 1.7781
Iteration: 14	Y	4.0880,	[+0.0202923,	0.0489765,	gradient = 1.7781
Iteration: 15	Y	4.0880,	[+0.0202923,	0.0489765,	gradient = 1.7781
Iteration: 16	Y	4.0880,	[+0.0202923,	0.0489765,	gradient = 1.7781
Iteration: 17	Y	4.0878,	[+0.0179967,	0.0430126,	gradient = 1.7793
Iteration: 18	Y	4.0878,	[+0.0179967,	0.0430126,	gradient = 1.7793
Iteration: 19	Y	4.0725,	[+0.0164831,	0.0418327,	gradient = 1.8001
Iteration: 20	Y	4.0725,	[+0.0164831,	0.0418327,	gradient = 1.8001
Iteration: 21	Y	4.0694,	[+0.0135922,	0.0362074,	gradient = 1.7805
Iteration: 22	Y	4.0694,	[+0.0135922,	0.0362074,	gradient = 1.7805
Iteration: 23	Y	4.0570,	[+0.0129157,	0.0344601,	gradient = 1.7821
Iteration: 24	Y	4.0570,	[+0.0129157,	0.0344601,	gradient = 1.7821
Iteration: 25	Y	4.0474,	[+0.0103264,	0.0287018,	gradient = 1.7834
Iteration: 26	Y	4.0474,	[+0.0103264,	0.0287018,	gradient = 1.7834
Iteration: 27	Y	4.0466,	[+0.0137723,	0.0371204,	gradient = 1.7834
Iteration: 28	Y	4.0466,	[+0.0137723,	0.0371204,	gradient = 1.7834
Iteration: 29	Y	4.0384,	[+0.01798179,	0.02401959,	gradient = 1.7846
Iteration: 30	Y	4.0384,	[+0.01798179,	0.02401959,	gradient = 1.7846
Iteration: 31	Y	4.0322,	[+0.0164732,	0.0258947,	gradient = 1.7858
Iteration: 32	Y	4.0322,	[+0.0164732,	0.0258947,	gradient = 1.7858
Iteration: 33	Y	4.0280,	[+0.0145053,	0.0193623,	gradient = 1.7867
Iteration: 34	Y	4.0280,	[+0.0145053,	0.0193623,	gradient = 1.7867
Iteration: 35	Y	4.0228,	[+0.0091056,	0.0162138,	gradient = 1.7867
Iteration: 36	Y	4.0228,	[+0.0091056,	0.0162138,	gradient = 1.7867
Iteration: 37	Y	4.0135,	[+0.0117637,	0.0115281,	gradient = 1.7879
Iteration: 38	Y	4.0135,	[+0.0117637,	0.0115281,	gradient = 1.7879
Iteration: 39	Y	4.0135,	[+0.0117637,	0.0115281,	gradient = 1.7879
Iteration: 40	Y	4.0072,	[+0.0021274,	0.0084601,	gradient = 1.7887
Iteration: 41	Y	4.0072,	[+0.0021274,	0.0084601,	gradient = 1.7887
Iteration: 42	Y	3.9978,	[+0.0097562,	0.0103770,	gradient = 1.7900
Iteration: 43	Y	3.9978,	[+0.0097562,	0.0103770,	gradient = 1.7900
Iteration: 44	Y	3.9884,	[+0.0095301,	0.0090285,	gradient = 1.7913
Iteration: 45	Y	3.9884,	[+0.0095301,	0.0090285,	gradient = 1.7913
Iteration: 46	Y	3.9822,	[+0.0093328,	0.0095079,	gradient = 1.7925
Iteration: 47	Y	3.9822,	[+0.0093328,	0.0095079,	gradient = 1.7925
Iteration: 48	Y	3.9780,	[+0.0093442,	0.0094397,	gradient = 1.7925
Iteration: 49	Y	3.9780,	[+0.0093442,	0.0094397,	gradient = 1.7925
Iteration: 50	Y	3.9728,	[+0.0091586,	0.009123,	gradient = 1.7934
Iteration: 51	Y	3.9728,	[+0.0091586,	0.009123,	gradient = 1.7934
Iteration: 52	Y	3.9653,	[+0.0097521,	0.0086496,	gradient = 1.7938
Iteration: 53	Y	3.9653,	[+0.0097521,	0.0086496,	gradient = 1.7938
Iteration: 54	Y	3.9605,	[+0.0098949,	0.0087508,	gradient = 1.7950
Iteration: 55	Y	3.9605,	[+0.0098949,	0.0087508,	gradient = 1.7950
Iteration: 56	Y	3.9570,	[		



[illegible]



Terration: 1919	y	-0.9925	y	-0.89624594	0.4832521,	gradient	-0.3544
Terration: 1920	y	-0.9925	y	-0.89624594	0.4832521,	gradient	-0.3544
Terration: 1921	y	-0.9922	y	-0.8964649	0.4835726,	gradient	-0.3538
Terration: 1922	y	-0.9921	y	-0.8968458	0.4837831,	gradient	-0.3535
Terration: 1923	y	-0.9919	y	-0.89724708	0.4841777,	gradient	-0.3529
Terration: 1924	y	-0.9916	y	-0.8976483	0.4845723,	gradient	-0.3525
Terration: 1925	y	-0.9916	y	-0.8976483	0.4845723,	gradient	-0.3525
Terration: 1926	y	-0.9915	y	-0.8978497	0.4849669,	gradient	-0.3522
Terration: 1927	y	-0.9915	y	-0.8978497	0.4849669,	gradient	-0.3522
Terration: 1928	y	-0.9913	y	-0.898251	0.4853615,	gradient	-0.3518
Terration: 1929	y	-0.9913	y	-0.898251	0.4853615,	gradient	-0.3518
Terration: 1930	y	-0.9911	y	-0.8986466	0.4857561,	gradient	-0.3514
Terration: 1931	y	-0.9911	y	-0.8986466	0.4857561,	gradient	-0.3514
Terration: 1932	y	-0.9909	y	-0.8988483	0.4861507,	gradient	-0.3511
Terration: 1933	y	-0.9909	y	-0.8988483	0.4861507,	gradient	-0.3511
Terration: 1934	y	-0.9907	y	-0.8992495	0.4865453,	gradient	-0.3505
Terration: 1935	y	-0.9907	y	-0.8992495	0.4865453,	gradient	-0.3505
Terration: 1936	y	-0.9905	y	-0.8996508	0.4869399,	gradient	-0.3499
Terration: 1937	y	-0.9905	y	-0.8996508	0.4869399,	gradient	-0.3499
Terration: 1938	y	-0.9903	y	-0.8998525	0.4873345,	gradient	-0.3496
Terration: 1939	y	-0.9903	y	-0.8998525	0.4873345,	gradient	-0.3496
Terration: 1940	y	-0.9902	y	-0.9002537	0.4877291,	gradient	-0.3492
Terration: 1941	y	-0.9901	y	-0.9002537	0.4877291,	gradient	-0.3492
Terration: 1942	y	-0.9899	y	-0.9006549	0.4881237,	gradient	-0.3488
Terration: 1943	y	-0.9899	y	-0.9006549	0.4881237,	gradient	-0.3488
Terration: 1944	y	-0.9896	y	-0.9010562	0.4885183,	gradient	-0.3484
Terration: 1945	y	-0.9896	y	-0.9010562	0.4885183,	gradient	-0.3484
Terration: 1946	y	-0.9894	y	-0.9014574	0.4889129,	gradient	-0.3481
Terration: 1947	y	-0.9894	y	-0.9014574	0.4889129,	gradient	-0.3481
Terration: 1948	y	-0.9890	y	-0.9018587	0.4893075,	gradient	-0.3477
Terration: 1949	y	-0.9890	y	-0.9018587	0.4893075,	gradient	-0.3477
Terration: 1950	y	-0.9889	y	-0.9022599	0.4897021,	gradient	-0.3474
Terration: 1951	y	-0.9889	y	-0.9022599	0.4897021,	gradient	-0.3474
Terration: 1952	y	-0.9887	y	-0.9026612	0.4900967,	gradient	-0.3471
Terration: 1953	y	-0.9887	y	-0.9026612	0.4900967,	gradient	-0.3471
Terration: 1954	y	-0.9885	y	-0.9030624	0.4904913,	gradient	-0.3467
Terration: 1955	y	-0.9885	y	-0.9030624	0.4904913,	gradient	-0.3467
Terration: 1956	y	-0.9883	y	-0.9034637	0.4908859,	gradient	-0.3464
Terration: 1957	y	-0.9883	y	-0.9034637	0.4908859,	gradient	-0.3464
Terration: 1958	y	-0.9881	y	-0.9038649	0.4912805,	gradient	-0.3461
Terration: 1959	y	-0.9881	y	-0.9038649	0.4912805,	gradient	-0.3461
Terration: 1960	y	-0.9879	y	-0.9042662	0.4916751,	gradient	-0.3457
Terration: 1961	y	-0.9879	y	-0.9042662	0.4916751,	gradient	-0.3457
Terration: 1962	y	-0.9878	y	-0.9046674	0.4920697,	gradient	-0.3456
Terration: 1963	y	-0.9878	y	-0.9046674	0.4920697,	gradient	-0.3456
Terration: 1964	y	-0.9876	y	-0.9050687	0.4924643,	gradient	-0.3452
Terration: 1965	y	-0.9876	y	-0.9050687	0.4924643,	gradient	-0.3452
Terration: 1966	y	-0.9874	y	-0.9054699	0.4928589,	gradient	-0.3449
Terration: 1967	y	-0.9874	y	-0.9054699	0.4928589,	gradient	-0.3449
Terration: 1968	y	-0.98					







[illegible]











[illegible]



[illegible]











[illegible]











[illegible]



