Roll no: 41310
Name: Prem Vinod Bansod
Assignment: 3(SCOA)

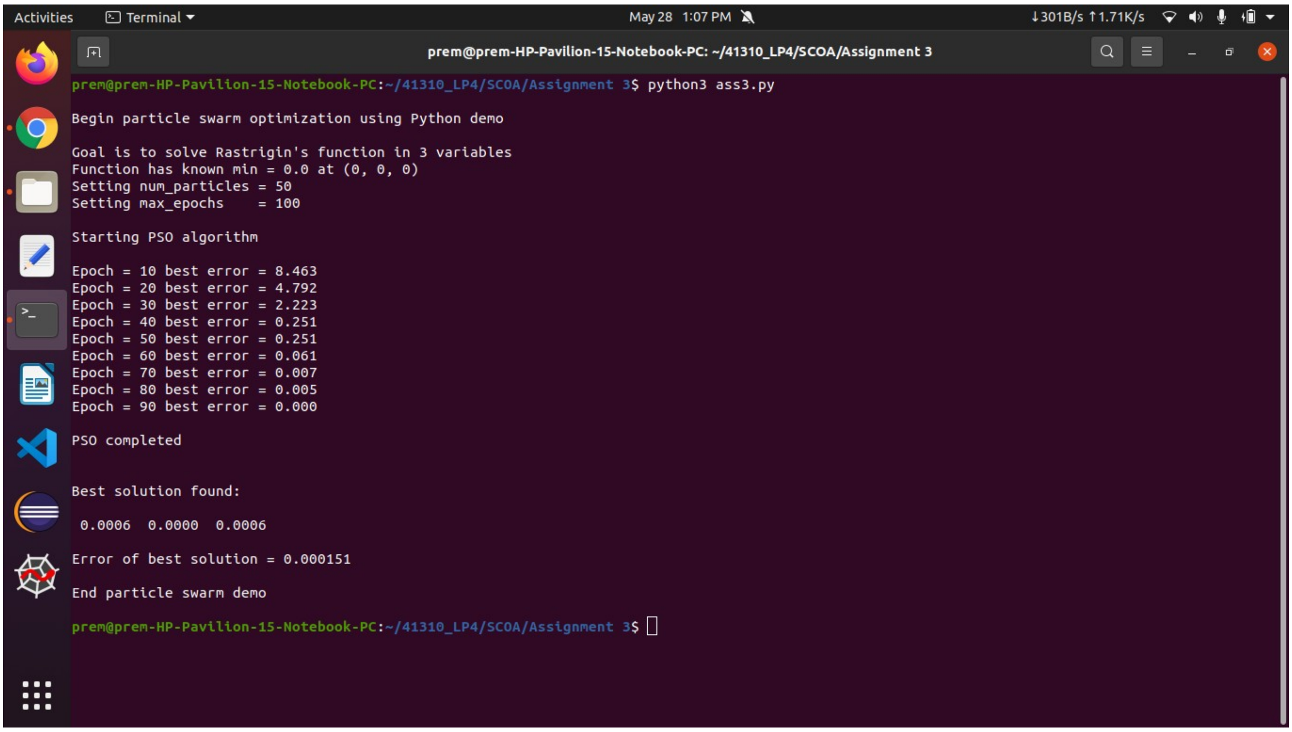
code:

```
import random
import math    # cos() for Rastrigin
import copy    # array-copying convenience
import sys     # max float

# ------------------------------------

def show_vector(vector):
  for i in range(len(vector)):
    if i % 8 == 0: # 8 columns
      print("\n", end="")
    if vector[i] >= 0.0:
      print(' ', end="")
    print("%.4f" % vector[i], end="") # 4 decimals
    print(" ", end="")
  print("\n")

def error(position):
  err = 0.0
  for i in range(len(position)):
    xi = position[i]
    err += (xi * xi) - (10 * math.cos(2 * math.pi * xi)) + 10
  return err

# ------------------------------------

class Particle:
  def __init__(self, dim, minx, maxx, seed):
    self.rnd = random.Random(seed)
    self.position = [0.0 for i in range(dim)]
    self.velocity = [0.0 for i in range(dim)]
    self.best_part_pos = [0.0 for i in range(dim)]

    for i in range(dim):
      self.position[i] = ((maxx - minx) *
        self.rnd.random() + minx)
      self.velocity[i] = ((maxx - minx) *
        self.rnd.random() + minx)

    self.error = error(self.position) # curr error
    self.best_part_pos = copy.copy(self.position)
    self.best_part_err = self.error # best error

def Solve(max_epochs, n, dim, minx, maxx):
  rnd = random.Random(0)
```

```
# create n random particles
swarm = [Particle(dim, minx, maxx, i) for i in range(n)]

best_swarm_pos = [0.0 for i in range(dim)] # not necess.
best_swarm_err = sys.float_info.max # swarm best
for i in range(n): # check each particle
  if swarm[i].error < best_swarm_err:
    best_swarm_err = swarm[i].error
    best_swarm_pos = copy.copy(swarm[i].position)

epoch = 0
w = 0.729    # inertia
c1 = 1.49445 # cognitive (particle)
c2 = 1.49445 # social (swarm)

while epoch < max_epochs:

  if epoch % 10 == 0 and epoch > 1:
    print("Epoch = " + str(epoch) +
      " best error = %.3f" % best_swarm_err)

  for i in range(n): # process each particle

    # compute new velocity of curr particle
    for k in range(dim):
      r1 = rnd.random()    # randomizations
      r2 = rnd.random()

      swarm[i].velocity[k] = ( (w * swarm[i].velocity[k]) +
        (c1 * r1 * (swarm[i].best_part_pos[k] -
        swarm[i].position[k])) +
        (c2 * r2 * (best_swarm_pos[k] -
        swarm[i].position[k])) )

      if swarm[i].velocity[k] < minx:
        swarm[i].velocity[k] = minx
      elif swarm[i].velocity[k] > maxx:
        swarm[i].velocity[k] = maxx

    # compute new position using new velocity
    for k in range(dim):
      swarm[i].position[k] += swarm[i].velocity[k]

    # compute error of new position
    swarm[i].error = error(swarm[i].position)

    # is new position a new best for the particle?
    if swarm[i].error < swarm[i].best_part_err:
      swarm[i].best_part_err = swarm[i].error
      swarm[i].best_part_pos = copy.copy(swarm[i].position)

    # is new position a new best overall?
```

```
      if swarm[i].error < best_swarm_err:
        best_swarm_err = swarm[i].error
        best_swarm_pos = copy.copy(swarm[i].position)

    # for-each particle
    epoch += 1
  # while
  return best_swarm_pos
# end Solve

print("\nBegin particle swarm optimization using Python demo\n")
dim = 3
print("Goal is to solve Rastrigin's function in " + str(dim) + " variables")
print("Function has known min = 0.0 at (", end="")
for i in range(dim-1):
  print("0, ", end="")
print("0)")

num_particles = 50
max_epochs = 100

print("Setting num_particles = " + str(num_particles))
print("Setting max_epochs    = " + str(max_epochs))
print("\nStarting PSO algorithm\n")

best_position = Solve(max_epochs, num_particles,
 dim, -10.0, 10.0)

print("\nPSO completed\n")
print("\nBest solution found:")
show_vector(best_position)
err = error(best_position)
print("Error of best solution = %.6f" % err)

print("\nEnd particle swarm demo\n")
```

output:


Activities
Terminal
May 28 1:07 PM
prem@prem-HP-Pavilion-15-Notebook-PC: ~/41310_LP4/SCOA/Assignment 3
prem@prem-HP-Pavilion-15-Notebook-PC:~/41310_LP4/SCOA/Assignment 3$ python3 ass3.py
Begin particle swarm optimization using Python demo
Goal is to solve Rastrigin's function in 3 variables
Function has known min = 0.0 at (0, 0, 0)
Setting num_particles = 50
Setting max_epochs = 100
Starting PSO algorithm
Epoch = 10 best error = 8.463
Epoch = 20 best error = 4.792
Epoch = 30 best error = 2.223
Epoch = 40 best error = 0.251
Epoch = 50 best error = 0.251
Epoch = 60 best error = 0.061
Epoch = 70 best error = 0.007
Epoch = 80 best error = 0.005
Epoch = 90 best error = 0.000
PSO completed
Best solution found:
0.0006 0.0000 0.0006
Error of best solution = 0.000151
End particle swarm demo
prem@prem-HP-Pavilion-15-Notebook-PC:~/41310_LP4/SCOA/Assignment 3$