Funcția de Fitness – Concept și vizualizare în Python

Asist.univ.dr Mihai Tudor

Obiective

- Definirea ueni funcții de fitness pentru o problemă simplă de optimizare,
- Bazele bibliotecii NumPy pentru lucrul cu date numerice.
- Vizualizarea grafică a funcției folosind biblioteca Matplotlib,

1 Biblioteca NumPy

NumPy (Numerical Python) este o bibliotecă care permite:

- lucrul eficient cu tablouri (arrays) numerice;
- operații matematice vectorizate (fără bucle for);
- generarea de date numerice pentru grafice.

Implementare

1. Operaţii cu vectori (arrays)

```python
from numpy import*

# Crearea unui array
a = array([1, 2, 3, 4, 5])
print("Array a:", a)

# Operatii vectoriale
print("a + 10:", a + 10)
print("a * 2:", a * 2)

# Proprietati de baza
print("Dimensiune:", a.ndim)
print("Lungime:", len(a))
```

2. Generarea de date numerice

```
from numpy import*
# Secventa de numere de la 0 la 10 cu pas de 1
x = arange(0, 11, 1)
print("x =", x)

# 100 de valori intre 0 si 2pi (util pentru functii
    trigonometrice)
x = linspace(0, 2 * pi, 100)
print("Primele 5 valori din x:", x[:5])
```

2 Biblioteca Mathplotlib

Matplotlib este o bibliotecă Python folosită pentru vizualizarea datelor numerice sub formă de grafice bidimensionale.

Este extrem de versatilă și permite reprezentarea:

- datelor discrete – seturi de puncte izolate;
- datelor continue – funcții sau relații matematice reprezentate prin curbe;
- personalizarea completă a graficelor (culori, titluri, legende, grile etc.).

Implementare

1. Reprezentarea discretă a datelor

```
from matplotlib.pyplot import*

# date discrete
x = [1, 2, 3, 4, 5]
y = [2, 3.5, 3, 4.5, 5]

plot(x, y, 'o', color='red')   # 'o' = doar marcatori
title("Reprezentare discreta (puncte izolate)")
xlabel("x")
ylabel("y")
grid(True)
show()
```

2. Reprezentarea continuă a datelor

```
from numpy import*
from matplotlib.pyplot import*

# date continue: functia f(x) = sin(x) pe [0, 2pi]
x = linspace(0, 2 * np.pi, 200)  # 200 de puncte "apropiate"
y = sin(x)
```

```
plot(x, y, color='blue')  # linie continua
title("Reprezentare continua: y = sin(x)")
xlabel("x")
ylabel("y")
grid(True)
show()
```

3. Reprezentarea discretă şi continuă a datelor în acelaşi grafic

```
from numpy import*
from matplotlib.pyplot import*

# reprezentam functia f(x) = x^2

x_discret = [0, 1, 2, 3, 4, 5]
y_discret = [xi**2 for xi in x_discret]

x_continuu = linspace(0, 5, 200)
y_continuu = x_continuu**2

plot(x_discret, y_discret, 'ro', label='Puncte discrete')
plot(x_continuu, y_continuu, 'b-', label='Curba continua')
title("Reprezentare discreta vs. continua")
xlabel("x")
ylabel("y")
legend()
grid(True)
show()
```

4. Reprezentarea datelor sub forma de bar char

```
from matplotlib.pyplot import*
categorii = ['A', 'B', 'C', 'D']
valori = [23, 45, 12, 36]

bar(categorii, valori, color='orange')
title("Bar char")
xlabel("Categorie")
ylabel("Valoare")
show()
```

5. Reprezentarea datelor sub forma de pie char

```
from matplotlib.pyplot import*
etichete = ['Apa', 'Suc', 'Cafea', 'Ceai']
procentaje = [40, 25, 20, 15]

pie(procentaje, labels=etichete)
title("Preferinte bauturi")
show()
```

Pentru reprezentarea seturilor de date mai complexe si o vizualizare a acestora într-un mediu mai avansat putem utiliza biblioteca **Seaborn**.

```
from seaborn import*
from matplotlib.pyplot import*

# incarcam datasetul "tips"
tips = load_dataset("tips")

# grafic discret intre nota totala de plata si bacsis
scatterplot(data=tips, x="total_bill", y="tip", color="teal")

title("Relatia intre nota totala ai bacaia")
xlabel("Nota totala ($)")
ylabel("Bacaia ($)")
grid(True)
show()
```

3 Funcția de fitness (funcţia obiectiv)

Funcția de fitness (funcţia obiectiv) este o aplicație $f : \mathcal{X} \to \mathbb{R}$ care asociază fiecărei soluții candidate o valoare numerică reprezentând "calitatea" acesteia, unde $\mathcal{X}$ este spaţiul de căutare. Într-un algoritm genetic, funcția de fitness evaluează calitatea unei soluții candidate. Cu alte cuvinte, ea răspunde la întrebarea: „Cât de bună este această soluție față de celelalte?”

Există două moduri de a defini o funcţie de fitness:

1. Direct din problemă.

   Exemplu - maximizarea profitului: $f(x) = profit(x)$

2. Propusă de noi.

   Exemplu - minimizarea costului: $f(x) = \frac{1}{1+cost(x)}$

Implementare

1. Funcţie de fitness unimodală

```
from numpy import*
from matplotlib.pyplot import*

# Definirea functiei de fitness
def fitness(x):
    return x**2

# Domeniul functiei
x = linspace(-5, 5, 400)
y = fitness(x)

```

```
# Reprezentarea grafica
plot(x, y, label="f(x) = x^2")
title("Exemplu de functie de fitness (maximizare)")
xlabel("x")
ylabel("Fitness")
legend()
grid(True)
show()
```

2. Funcţie de fitness multimodale

```
from numpy import*
from matplotlib.pyplot import*
def fitness2(x):
    return x * sin(10 * pi * x) + 1

x = linspace(0, 1, 400)
y = fitness2(x)

plot(x, y, color='orange', label="f(x) = x*sin(10pix) + 1")
title("Functie de fitness cu multiple maxime locale")
xlabel("x")
ylabel("Fitness")
legend()
grid(True)
show()
```