EVOLUTION STRATEGIES (ES)

agungsetiabudi@ub.ac.id

Evolution Strategies (ES)

- Evolution Strategies adalah algoritma optimisasi yang terinspirasi oleh mekanisme evolusi biologis.
- Fokus utama dari ES adalah pada operator mutasi, yang diimplementasikan secara bertahap melalui **distribusi Gaussian** untuk menghasilkan variasi pada solusi.

Langkah-langkah dalam ES

1. **Inisialisasi Populasi**: ES dimulai dengan menginisialisasi populasi solusi kandidat secara acak.
2. **Mutasi**: Solusi kandidat mengalami mutasi berdasarkan distribusi Gaussian. Misalnya, untuk solusi X dan langkah mutasi σ, vektor anak X' diperoleh dari:
   $X' = X + \sigma \cdot N(0, 1)$
   di mana $N(0, 1)$ adalah nilai acak dari distribusi Gaussian standar.

Langkah-langkah dalam ES

3. **Rekombinasi**: Dalam beberapa varian ES, seperti $(\mu + \lambda)$-ES dan (μ, λ)-ES, dua atau lebih solusi dapat dikombinasikan untuk menghasilkan solusi baru. Kombinasi ini dapat berupa rekombinasi aritmetika atau metode crossover lainnya.
4. **Seleksi**: Dalam ES, proses seleksi melibatkan memilih μ individu terbaik dari λ anak untuk membentuk populasi generasi berikutnya. Pada skema (μ, λ)-ES, hanya anak-anak yang dapat dipilih untuk generasi berikutnya, sedangkan pada $(\mu + \lambda)$-ES, baik anak-anak maupun orang tua dapat dipertimbangkan.

Langkah-langkah dalam ES

5. **Penyesuaian Parameter Mutasi**: ES sering menggunakan adaptasi parameter untuk memperbaiki strategi mutasi. Salah satu teknik yang sering digunakan adalah *self-adaptation*, di mana parameter mutasi (seperti σ) ikut beradaptasi seiring dengan evolusi populasi.

Keunggulan ES

- Efektif untuk masalah optimisasi yang melibatkan parameter kontinu.
- Mampu beradaptasi dengan kompleksitas lanskap pencarian melalui adaptasi parameter mutasi.
- Tahan terhadap noise dan variabilitas pada fungsi objektif.

Contoh

- Evolution Strategies memiliki beberapa varian, namun di sini kita akan fokus pada skema (μ, λ)-ES, yang berarti:
  - μ adalah jumlah individu di populasi induk,
  - λ adalah jumlah keturunan yang dihasilkan dari populasi induk,
  - Hanya keturunan yang berkompetisi untuk masuk ke dalam generasi berikutnya, yaitu tidak melibatkan induk.
- Misalkan kita ingin meminimalkan fungsi berikut:
  $f(x, y) = x^2 + y^2$
  dengan ruang pencarian untuk x dan y berada di antara $[-5, 5]$.

1. Inisialisasi Populasi

- Kita memulai dengan menginisialisasi populasi induk berukuran μ secara acak. Misalkan $\mu = 3$, sehingga kita memiliki tiga individu induk:
  $X_1 = [2.0, -3.0], \quad X_2 = [-1.5, 2.5], \quad X_3 = [0.5, -0.5]$
- Masing-masing individu memiliki nilai parameter x dan y. Selain itu, ES biasanya menggunakan *step size* atau parameter mutasi σ yang bisa berbeda-beda untuk setiap individu, misalnya:
  $\sigma_1 = 0.5, \quad \sigma_2 = 0.3, \quad \sigma_3 = 0.8$

Contoh program

```
def initialize_population(mu, bounds, step_size):
    population = []
    step_sizes = []
    for _ in range(mu):
        individual = [np.random.uniform(low, high) for (low, high) in bounds]
        population.append(individual)
        step_sizes.append(step_size)  # Set step size awal yang sama untuk setiap individu
    return np.array(population), np.array(step_sizes)
```

2. Mutasi

- Mutasi dilakukan dengan menambahkan gangguan acak berbasis distribusi Gaussian (Normal) pada setiap parameter individu.
- Untuk setiap individu $X_i = [x, y]$, mutasi menghasilkan keturunan X_i' sebagai berikut:
  - $X_i' = X_i + \sigma_i \cdot N(0, 1)$
- di mana $N(0, 1)$ adalah nilai acak dari distribusi Gaussian standar dengan mean 0 dan variansi 1.

2. Mutasi

Misalkan kita ingin menghasilkan $\lambda = 6$ keturunan. Dengan demikian, setiap individu induk menghasilkan dua keturunan:

1. Untuk $X_1 = [2.0, -3.0]$ dengan $\sigma_1 = 0.5$:
   - Keturunan pertama:
     $X_1' = [2.0, -3.0] + 0.5 \cdot [0.2, -1.2] = [2.1, -3.6]$
   - Keturunan kedua:
     $X_2' = [2.0, -3.0] + 0.5 \cdot [-0.4, 0.8] = [1.8, -2.6]$

2. Mutasi

2. Untuk $X_2 = [-1.5, 2.5]$ dengan $\sigma_2 = 0.3$:
   - Keturunan ketiga:
     $X'_3 = [-1.5, 2.5] + 0.3 \cdot [0.7, -0.5] = [-1.29, 2.35]$
   - Keturunan keempat:
     $X'_4 = [-1.5, 2.5] + 0.3 \cdot [-0.2, 1.1] = [-1.56, 2.83]$
3. Untuk $X_3 = [0.5, -0.5]$ dengan $\sigma_3 = 0.8$:
   - Keturunan kelima:
     $X'_5 = [0.5, -0.5] + 0.8 \cdot [1.3, -0.7] = [1.54, -1.06]$
   - Keturunan keenam:
     $X'_6 = [0.5, -0.5] + 0.8 \cdot [-0.9, 0.4] = [-0.22, -0.18]$

2. Mutasi

Sekarang, kita memiliki enam keturunan:

$$X'_1 = [2.1, -3.6], \quad X'_2 = [1.8, -2.6], \quad X'_3 = [-1.29, 2.35]$$
$$X'_4 = [-1.56, 2.83], \quad X'_5 = [1.54, -1.06], \quad X'_6 = [-0.22, -0.18]$$

Contoh Program

```
for i in range(lmbda):
    # Pilih secara acak satu induk dari populasi
    parent_index = np.random.randint(0, mu)
    parent = population[parent_index]
    step_size = step_sizes[parent_index]

    # Mutasi (self-adaptation)
    new_step_size = step_size * np.exp(np.random.normal(0, 1) * 0.1)
    child = parent + new_step_size * np.random.normal(0, 1, len(bounds))

    # Membatasi anak dalam batas yang diizinkan
    child = np.clip(child, [b[0] for b in bounds], [b[1] for b in bounds])

    offspring.append(child)
    offspring_step_sizes.append(new_step_size)
```

4. Seleksi

- Dalam skema (μ, λ)-ES, hanya keturunan yang dapat dipilih untuk generasi berikutnya.
- Kita memilih μ keturunan dengan nilai fungsi objektif terendah. Misalkan kita menetapkan $\mu = 3$, sehingga kita memilih tiga keturunan terbaik berdasarkan nilai fungsi objektif:
  - $X'_6 = [-0.22, -0.18]$ dengan $f = 0.0808$
  - $X'_5 = [1.54, -1.06]$ dengan $f = 3.4952$
  - $X'_3 = [-1.29, 2.35]$ dengan $f = 7.1866$

4. Seleksi

Populasi generasi berikutnya menjadi:

$X_1 = [-0.22, -0.18], \quad X_2 = [1.54, -1.06], \quad X_3 = [-1.29, 2.35]$

Contoh Program

```
# Evaluasi keturunan
offspring = np.array(offspring)
offspring_step_sizes = np.array(offspring_step_sizes)
scores = np.array([objective_function(ind) for ind in offspring])

# Seleksi: Memilih mu individu terbaik dari lambda keturunan
selected_indices = scores.argsort()[:mu]
population = offspring[selected_indices]
step_sizes = offspring_step_sizes[selected_indices]
```

5. Adaptasi Parameter Mutasi (Opsional)

- ES sering menggunakan adaptasi parameter mutasi (*self-adaptation*) di mana parameter mutasi (σ) juga dievolusi.
- Setiap individu memiliki nilai σ yang bervariasi berdasarkan adaptasi otomatis. Misalnya, kita dapat menggunakan pendekatan berikut:
  - $\sigma' = \sigma \cdot e^{\tau \cdot N(0,1)}$
- dengan τ sebagai faktor adaptasi. Adaptasi ini dilakukan agar parameter mutasi (σ) menjadi lebih besar atau kecil sesuai dengan perubahan lanskap pencarian di setiap generasi.

Ringkasan Iterasi

- Pada iterasi berikutnya, proses mutasi, evaluasi, dan seleksi akan diulangi menggunakan populasi baru.
- Hal ini akan terus dilakukan hingga mencapai kriteria penghentian, misalnya ketika jumlah iterasi maksimum tercapai atau ketika perbedaan antara nilai terbaik dalam populasi menjadi sangat kecil.