Artificial Intelligence and Evolutionary Computing

Lecture 7

IT/T/414A

Winter Semester 2014

Instructor: Kartick Ch. MONDAL

Evolutionary Strategies

Evolution Strategies

- Invented early 1960s in Germany to simulating natural evolution
- Ingo Rechenberg, Hans-Paul Schwefel and Peter Bienert of the Technical University of Berlin
  - Engineering students
- Experimenting with wind tunnels
  - Optimizing jointed flat plates
- Were working on the search for the optimal shapes of bodies in a flow.
- Only intuitive methods to do this at the time

Evolution Strategies

- Unlike genetic algorithms, this approach called an **evolution strategy**
- Designed to solve technical optimization problems.
- They decided to try random changes in the parameters defining the shape following the example of natural mutation.
- As a result, the evolution strategy was born.
- Evolution strategies were developed as an alternative to the engineer's intuition.

Evolution Strategies

- Unlike GAs, evolution strategies use only a mutation operator.
- Like GP, no distinction between search and solution space
- Individuals are represented as **real-valued** vectors.
- Simple ES
  - One parent and one child
  - Child solution generated by randomly mutating the problem parameters of the parent.
- Susceptible to **stagnation** at local optima
- Rechenberg had the idea of 'mutating' the parameters and selecting good mutations
- Evolution Strategy are used for numerical parameter optimization

Evolution Strategies

- Fitness of individual is determined by how well the parameters solve the problem
- Offspring are created by mutation
- Real-numbered, normally distributed creep mutation
- Offspring replace parents only if more fit

Evolutionary Strategy vs GA’s

- ES’s are an algorithm that only uses mutation and does not use crossover

- This is not a formal definition and there is no reason why we cannot incorporate crossover (as Michalewicz, 1996 shows)

Evolutionary Strategy vs GA's

- ES's are normally applied to real numbers (continuous variables) rather than discrete values.
- Again, this is not a strict definition and work has been done on using ES's for discrete problems (Bäck, 1991) and (Herdy, 1991)

Evolutionary Strategy vs GA’s

- ES’s are a population based approach
- Originally only a single solution was maintained and this was improved upon.

Evolution Strategies

- An Example Evolution Strategy

```
Procedure ES{
  t = 0;
  Initialize P(t);
  Evaluate P(t);
  While (Not Done)
  {
    Parents(t) = Select_Parents(P(t));
    Offspring(t) = Procreate(Parents(t));
    Evaluate(Offspring(t));
    P(t+1)= Select_Survivors(P(t),Offspring(t));
    t = t + 1;
  }
```

Basic evolution strategies

- In its simplest form, termed as a (1+1)-evolution strategy, one parent generates one offspring per generation by applying *normally distributed* mutation. The (1+1)-evolution strategy can be implemented as follows:

  <u>Step 1</u>: Choose the number of parameters *N* to represent the problem, and then determine a feasible range for each parameter:

- $\{x_{1min}, x_{1max}\}, \{x_{2min}, x_{2max}\}, \ldots, \{x_{Nmin}, x_{Nmax}\}$

  Define a standard deviation for each parameter and the function to be optimized.

Basic evolution strategies

<u>Step 2</u>: Randomly select an initial value for each parameter from the respective feasible range. The set of these parameters will constitute the initial population of parent parameters: $x_1, x_2, \ldots, x_N$

<u>Step 3</u>: Calculate the solution associated with the parent parameters: $X = f(x_1, x_2, \ldots, x_N)$

Basic evolution strategies

Step 4: Create a new offspring parameter (x'_1, x'_2, . . . , x'_N) by adding a normally distributed random variable *a* with mean zero and pre-selected deviation δ to each parent parameter $x_i' = x_i + a\left(0,\ \delta\right)$

where i = 1, 2, . . . , *N*

Normally distributed mutations with mean zero reflect the natural process of evolution (smaller changes occur more frequently than larger ones).

Step 5: Calculate the solution associated with the offspring parameters: $X' = f\left(x_1', x_2', \ldots, x_N'\right)$

Basic evolution strategies

<u>Step 6</u>: Compare the solution associated with the offspring parameters with the one associated with the parent parameters.

– If the solution for the offspring is better than that for the parents, replace the parent population with the offspring population.
– Otherwise, keep the parent parameters.

<u>Step 7</u>: Go to Step 4, and repeat the process until a satisfactory solution is reached, or a specified number of generations is considered.

Evolution Strategies

- Evolution Strategy named according to number of parents and children at each generation
- 1+1 Evolution Strategy has one parent and one child

Evolution Strategies

- Later Evolution Strategy have populations
- $(\mu+\lambda)$ and (μ,λ) Evolution Strategy
- μ (mu) is the size of the parent population
- λ (lambda) is the size of the offspring population
- Offspring are created using recombination as well as mutation

Evolution Strategies

- In a $(\mu+\lambda)$ ES, the μ best survive to the next generation
- In a (μ,λ) ES, only child individuals survive to the next generation

Evolution Strategies

- There are basically 4 types of ESs
  - The Simple (1+1)-ES
  - The (μ+1)-ES (The first multimembered ES)
  - The ($\mu+\lambda$)-ES, and
  - The (μ,λ)-ES.

1+1 ES

1. Evaluated fitness of parent *P, f(P)*
2. Create child *C* by adding small normally distributed values to each parameter of *P*
3. Evaluate the fitness of *C, f(C)*
4. If *f(C) > f(P)* then
replace *P* with *C*
5. Repeat Steps 2-4 until stopping condition

Evolution Strategies: The Simple (1+1)-ES

- The simple (1+1)-ES has the following attributes:
  - Individuals are represented as follows:
    - $<x_{i,0}, x_{i,1}, \ldots, x_{i,n-1}, \sigma_i>$, where n is the number of variables
  - Offspring are created as follows:

$$\sigma_{\mu+i,j} = \sigma_{k,j} * \exp(\tau_0 * N(0,1));$$

$$x_{\mu+i,j} = x_{k,j} + \sigma_{\mu+i,j} N_{\mu+i,j}(0,1);$$

    Where j represents the jth variable.

    And where $\tau_0 \approx 1/\mathrm{sqrt}(n)$ (Global Learning Rate)
  - Uses the 1/5 Success Rule to Adapt the Step Size:
    - If more than 1/5th of the mutations cause an improvement (in the objective function) then multiply σ by 1.2,
    - If less than 1/5th of the mutations cause an improvement, then multiply σ by 0.8.

Evolution Strategies: The Simple (1+1)-ES

- ```
  Procedure simpleES{
     t = 0;
     Initialize P(t); /* μ = 1, λ = 1 */
     Evaluate P(t);
     while (t <= (4000-μ)/λ){
       for (i=0; i<1; i++){
         Create_Offspring(<x_i,y_i,σ_i>,<x_{μ+i},y_{μ+i},σ_{μ+i}>):
            σ_{μ+i} = σ_i * exp(τ_0 * N(0,1));
            x_{μ+i} = x_i + σ_{μ+i}N_{μ+i,x}(0,1);
            y_{μ+i} = y_i + σ_{μ+i}N_{μ+i,y}(0,1);
         fit_{μ+i} = Evaluate(<x_{μ+i},y_{μ+i}>);
       }
       P(t+1) = Better of 2 individuals;
       t = t + 1;
     }
  }
  ```
  ```

Evolution Strategies: The Simple (1+1)-ES

- How is a simple (1+1)-ES similar to a (1+1)-Standard EP?
- In what ways are these two different?

Evolution Strategies: The (μ+1)-ES

- Since the (μ+1)-ES is multi-membered, crossover can be used.
- According to, Bäck, T., Hoffmeister, F, and Schwefel, H.-P. (1991). "A Survey of Evolution Strategies", *The Proceedings of the 4th International Conference on Genetic Algorithms*, R. K. Belew and L. B. Booker Eds., pp. 2-9, Morgan Kaufmann.
  - Uniform Crossover (also referred to a discrete recombination) can be used on the variable values as well as the strategy parameter.
- Adaptation of the step-size is not used in the (μ+1)-ES.

Evolution Strategies: The (μ+1)-ES

```
Procedure (μ+1)-ES{
     t = 0;
     Initialize P(t); /* of μ individuals */
     Evaluate P(t);
     while (t <= (4000-μ)){
          Create_Offspring(<x_i,y_i,σ_i>,<x_μ+i,y_μ+i,σ_μ+i>):
             σ_μ+i = σ_i * exp(τ_0 * N(0,1));
             x_μ+i = x_i + σ_μ+i N_μ+i,x(0,1);
             y_μ+i = y_i + σ_μ+i N_μ+i,y(0,1);
          fit_μ+i = Evaluate(<x_μ+i,y_μ+i>);
       P(t+1) = Best μ of the μ+1 individuals;
       t = t + 1;
     }
}
```

Evolution Strategies: The $(\mu+\lambda)$-ES

- In the $(\mu+\lambda)$-ES, an individual, i, is represented as follows:
- $<x_{i,0},x_{i,1},\ldots, x_{i,n-1},\sigma_{i,0},\sigma_{i,1},\ldots,\sigma_{i,n-1}>$, where n is the number of variables
- Offspring are created by as follows:
  - $\sigma_{\mu+i,j} = \sigma_{k,j} * \exp(\tau' * N(0,1) + \tau * N_{\mu+i}(0,1))$;
    $x_{\mu+i,j} = x_{k,j} + \sigma_{\mu+i,j}N_{\mu+i,j}(0,1)$;

- Where j represents the jth variable,
- $\tau' \approx 1/\mathrm{sqrt}(2n)$ /* Global Learning Rate */
- $\tau \approx 1/\mathrm{sqrt}(2*\mathrm{sqrt}(n))$ /* Individual Learning Rate */
- The 1/5th success rule is used.

Evolution Strategies: The (μ+λ)-ES

```
Procedure (μ+λ)-ES{
      t = 0;
      Initialize P(t); /* of μ individuals */
      Evaluate P(t);
      while (t <= (4000-μ)/λ){
           for (i=0; i<λ; i++){
             Create_Offspring(<x_k,y_k,σ_{k,x},σ_{k,y}>, <x_{μ+i},y_{μ+i},σ_{μ+i,x},σ_{μ+i,y}>):

                σ_{μ+i,x} = σ_{k,x} * exp(τ' * N(0,1) + τ * N_{μ+i}(0,1));
                x_{μ+i}   = x_i + σ_{μ+i,x} N_{μ+i,x}(0,1);

                σ_{μ+i,y} = σ_{k,y} * exp(τ' * N(0,1) + τ * N_{μ+i}(0,1));
                y_{μ+i}   = y_i + σ_{μ+i,y} N_{μ+i,y}(0,1);

                fit_{μ+i} = Evaluate(<x_{μ+i},y_{μ+i}>);
           }
           P(t+1) = Best μ of the μ+λ individuals;
           t = t + 1;
      }
}
```

Evolution Strategies: The (μ,λ)-ES

```
Procedure (μ,λ)-ES{
      t = 0;
      Initialize P(t); /* of μ individuals */
      Evaluate P(t);
      while (t <= (4000-μ)/λ)
      {
           for (i=0; i<λ; i++)
           {
             Create_Offspring(<x_k,y_k,σ_k,x,σ_k,y>, <x_μ+i,y_μ+i,σ_μ+i,x,σ_μ+i,y>):

                σ_μ+i,x = σ_k,x * exp(τ' * N(0,1) + τ * N_μ+i(0,1));
                x_μ+i   = x_i + σ_μ+i,x N_μ+i,x(0,1);

                σ_μ+i,y = σ_k,y * exp(τ' * N(0,1) + τ * N_μ+i(0,1));
                y_μ+i   = y_i + σ_μ+i,y N_μ+i,y(0,1);

                fit_μ+i = Evaluate(<x_μ+i,y_μ+i>);
           }
           P(t+1) = Best μ of the λ offspring;
           t = t + 1;
      }
}
```

Evolutionary Strategies

- Slow to converge to optimal solution
- More advanced ES
  - have pools of parents and children
- Unlike GA and GP, ES have these properties:
  - ES Separates parent individuals from child individuals
  - ES Selects its parent solutions deterministically

Evolutionary Algorithms

- In summary ES's are
  - Like genetic algorithms but only use mutation and not crossover
  - They operate on real numbers
  - They are a population based approach
  - But we can break any, or all, of these rules if we wish!