Universidad Nacional Autónoma de México

Facultad de Ciencias

Tarea 06:

Estrategias evolutivas

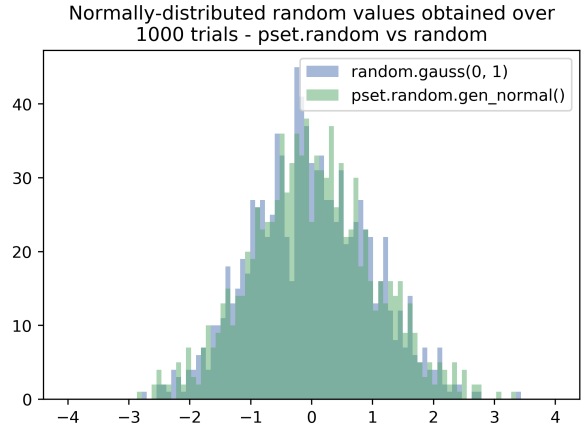
Pablo A. Trinidad Paz - 419004279

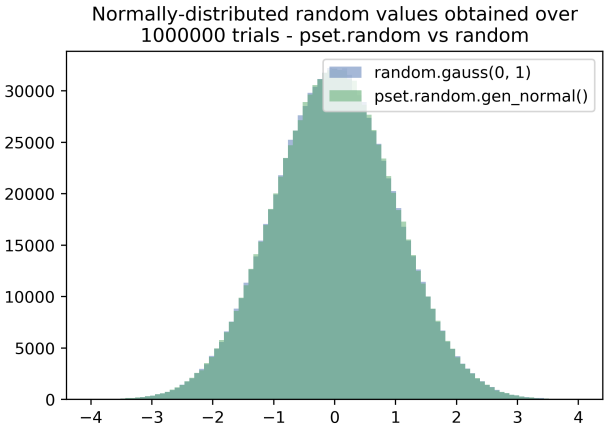
Trabajo presentado como parte del curso de **Cómputo Evolutivo** impartido por el profesor **Mario Iván Jaen Márquez**.

Fecha de entrega: **Jueves 4 de Abril de 2019**.

1. **[Ejercicio de programación]** Escribe una función que genere números pseudo-aleatorios de las distribución normal estándar $N(0,1)$ a partir de números uniformemente distribuidos. Indica el método usado.

Solución: Se implementó el método de muestreo de números pseudo-aleatorios descrito por Box-Muller[1]. A continuación se presentan los resultados de la implementación comparados con el método random.gauss$(0,1)$ de la librería estándar de Python.





[1] https://en.wikipedia.org/wiki/Box-Muller_transform

2. **[Ejercicio de programación]** Implementa el algoritmo (1+1)-ES. Prueba tu algoritmo sobre la función *Sphere*, la cuál es una función unimodal d-dimensional definida como:

$$f(\vec{x}) = \sum_{i=1}^{d} x_i^2$$

Donde cada $x_i \in [-100, 100]$. Utiliza un parámetro $\sigma = 1$ y un punto inicial $\vec{x} = (x_1, ..., x_d) = (-99, ..., -99)$

a) Ejecuta tu algoritmo para $d = 10$ y para $d = 100$. ¿Qué tan cerca del óptimo converge y qué tan rápido?

Respuesta: Para $d = 10$ con una precisión de 0.01 y un máximo de 10^6 iteraciones, el algoritmo concluye después 413 mutaciones exitosas debido a haber alcanzado el número máximo de iteraciones con un valor promedio para cada x_i de -0.068264 y un fitness de 0.34359 (el cual no cumple con la precisión). Para $d = 100$ con la misma precisión, el algoritmo concluye después de 1296 mutaciones exitosas con un valor promedio para cada x_i de 0.03281650 y fitness de 125.16605 (nuevamente lejano al óptimo y terminando debido al número máximo de iteraciones).

b) Implementa la regla del 1/5 y vuelve a ejecutar tu algoritmo. ¿Qué diferencias observas respecto a la ejecución anterior?

Respuesta: La diferencia es fundamental en el sentido de que los algoritmos sí convergieron y en menos iteraciones que el límite permitía (10^6). Aproximadamente el $\sim 20\,\%$ de las mutaciones fueron exitosas en ambos casos a comparación del caso anterior donde menos del $0.001\,\%$ de las mutaciones fueron exitosas. En ambos casos se logró obtener fitness bajo la precisión solicitada: para $d = 10$ se obtuvo 0.00907 y para $d = 100$ se obtuvo 0.00987 con 515 y 22,867 iteraciones respectivamente y 120 y 5,715 mutaciones exitosas respectivamente.

```
Starting simulation with d=10 (fifth rule disabled)
Objective reached
        Generations: 1000000
        Successful mutations: 413
        Chromosome mean: -0.06826455987971958
        Fitness: 0.34359743285720606
Starting simulation with d=100 (fifth rule disabled)
Objective reached
        Generations: 1000000
        Successful mutations: 1296
        Chromosome mean: 0.032816509152443094
        Fitness: 125.16605998855266
Starting simulation with d=10 (fifth rule enabled)
Objective reached
        Generations: 515
        Successful mutations: 120
        Chromosome mean: -0.005919947575676352
        Fitness: 0.009073898608427272
Starting simulation with d=100 (fifth rule enabled)
Objective reached
        Generations: 22867
        Successful mutations: 4715
        Chromosome mean: 0.0003964242388768172
        Fitness: 0.009875960400695284
```