**Assignment 3**

By : Malka Abramovitch, Maor Betser

**The problem:**

Given quadratic family functions with mixed variables, we had to optimize the integer part of the solution matrix in the range of [-100,100] where the first 32 numbers are real numbers and the latter 32 numbers are integers.

**Path to solution:**

At first, we tried implementing a **Genetic Algorithm (GA)** to optimize the **integer part** of the solution after running 1**+1 ES** on the real part. The idea was that since the real part was already optimized, we could focus entirely on refining the integer values.

However, we ran into **runtime issues**—running **10⁶ iterations with 30 NRUNS** was costly. Our results were **~10 on HCigar and RotatedHEllipse**, while for **Hadamard**, we reached **~500**.

At that point, we felt **stuck** and started doubting our GA approach. Instead of forcing it, we decided to **talk to classmates, compare ideas, and look for inspiration** on a better way to tackle the problem. That brainstorming session helped us rethink our strategy and eventually led to our second approach.

For our second attempt, we wanted something simple, fast, and effective. Instead of overcomplicating things, we went back to the 1+1 ES from class to handle the real part of the vector. Once that was optimized, we focused on the integer part using mutation.

At first, we kept the mutation basic randomly picking a step from [-1, 0, 1] to allow small jumps. Later, we improved it by switching to a **Gaussian-based mutation**, where:

* Small jumps happened more often.
* Larger jumps were rare but possible, helping the search escape local minima ("dips").

This approach worked **really well**—we saw **scores drop to 0** on the first two functions. But Hadamard? **That one was tough.** Our best result was **~57**, and despite our improvements, we couldn’t push it lower.

Outputs taken from 030.ipynb:

genHcigar | Run 10: Best Combined Fitness = 9.164359451072592e-10 | Best solution = [ 7.00029887 -6.99999996 6.9999993 -7.00000022 7.00000052 -7.00000018  
 7.00000049 -7.00000136 7.0000004 -7.00000042 6.99999877 -7.00000145  
 6.9999995 -6.9999988 7.00000115 -6.99999975 6.99999846 -6.99999821  
 6.99999862 -6.99999907 6.99999979 -7.00000048 6.99999943 -6.99999943  
 6.99999918 -7.00000016 7.00000051 -6.99999936 7.00000095 -6.99999968  
 6.99999882 -7.00000042 7. -7. 7. -7.  
 7. -7. 7. -7. 7. -7.  
 7. -7. 7. -7. 7. -7.  
 7. -7. 7. -7. 7. -7.  
 7. -7. 7. -7. 7. -7.  
 7. -7. 7. -7. ]

**genRotatedHellipse** | Run 29: Best Combined Fitness = 9.917684940356375e-07 | Best solution = [ 6.99234341 -6.99969282 7.00141066 -7.00020068 7.00062212 -7.00010197  
 7.00001591 -6.99999754 7.00027108 -7.00030247 7.00009227 -7.00020855  
 7.00020467 -7.00015642 7.00036259 -7.00017031 7.00029702 -6.999943  
 7.0002107 -7.00013039 7.00036073 -7.00005952 7.00002376 -7.00008649  
 7.00025615 -7.00017909 6.99996681 -7.0001505 7.00010171 -7.00020887  
 6.99998808 -7.00002225 7. -7. 7. -7.  
 7. -7. 7. -7. 7. -7.  
 7. -7. 7. -7. 7. -7.  
 7. -7. 7. -7. 7. -7.  
 7. -7. 7. -7. 7. -7.  
 7. -7. 7. -7. ]

**genHadamardHellipse** | Run 26: Best Combined Fitness = 57.0838955736759 | Best solution = [ -1.51685005 -6.09616108 0.40093612 -13.77349842 -0.71430614  
 -5.2813 1.5109152 -9.9566662 0.40127872 -5.41077176  
 1.08967328 -13.40032349 -1.31098022 -4.29423538 3.61663172  
 -9.23000237 -3.2162443 -5.21453675 0.3066451 -13.97066138  
 -1.12529974 -4.8230191 1.44062789 -7.94045564 -0.53172954  
 -4.53189776 0.62818344 -12.59539899 -0.62494219 -4.58929151  
 2.80922747 -9.00636634 -1. -6. 1.  
 -13. -1. -6. 1. -10.  
 0. -5. 2. -12. 0.  
 -5. 2. -9. -2. -6.  
 0. -13. -1. -6. 1.  
 -8. 0. -6. 2. -12.  
 0. -5. 2. -9. ]

**Final Thoughts**

This assignment was a rollercoaster. We started with **high expectations for GA**, but got humbled by **runtime issues**, and ultimately found a **faster, simpler approach** through mutation. Consulting with classmates in between was a super important for us it helped us step back and rethink our direction.

**First run with Uniform Distribution (100,000 iterations)**



**Run with Gaussian Distribution (50,000 iterations)** we can see an improvement with half of the calculations.

