FINDING 5D POLYTOPES USING A GENETIC ALGORITHM

Eldar Sultanow and Alexander Rahn

Abstract. Five-dimensional polytopes play a role in string theory, especially when they are reflexive. It is not trivial to find reflexive polytopes and there exists genetic approaches to finding them. We are implementing a genetic algorithm in Python and have found a new five-dimensional polytope.

1. Introduction

First of all: When we talk about polytopes, we mean lattice polytopes. The reflective property of a polytope is given if the polytope has only one interior point and if the lattice distances to this interior point are each 1.

Berglund, He, Heyes, Hirst, Jejjala, and Lukas [1] have demonstrated how a genetic approach can be used to identify new reflexive polytopes. The authors have found various polytopes and published their data set on GitHub [2]. We take up this idea and embark on a search for new five-dimensional polytopes.

2. Implementing the genetic algorithm

To develop a solid algorithm with the possibility of adjustments on several ends, no Genetic Algorithm Library was used. Instead, the whole structure was built up from the scratch.

The current first version, does not implement specific mutation or selection methods which will be available in future versions. However, the fact of a first successfull finding of a reflexive polytope proofs the functionality of the underlying approach.

To be able to use a Genetic Algorithm for solving a Problem, a proper representation needs to be defined. For this, the choice fell on a matrix, which holds the information about the position of the vertices of the polytope.

A proper structure is provided in form of objects. The representation-matrix is part of the Chromosome object. A Generation consists of n Chromosomes. m Generations build the Population.

2010 *Mathematics Subject Classification.* 52B05.

Key words and phrases. Polytopes, Calabi-Yau Manifold, Genetic Algorithm.

This structure is the base of the algorithm and builds up the overall solution in a top-down approach.

After defining basic information like the number of Generations, the number of Chormosomes per Generation, the dimension of the polytope and some more numbers, the Population gets initialized. Case one: A first Generation is handed over while initializing the object, new Generations are generated through crossover of the existing polytopes. Case two no first Generation is handed over, so n representation-matrixes of polytopes are randomly created and used as first Generation. This randomly generated Generation marks the start Generation of the Genetic Algorithm and is used to create new Generations through crossover. After the first Generation is set, all Chromosomes are sorted by their Fitness.

The fitness is based on the idea of Berglund et al. [1]. We penalize the fitness value if the polytope has more than one interior point and if the lattice distance of a vertex to the polytope's interior point is not equal to 1, see Listing 1.

```
def calc_fitness(vertices):
    ip_count = len(enumerate_integral_points(qhull(vertices)))
    distances = compute_distances(vertices)
    result = 0
    if ip_count > 1:
        result -= 1
    for d in distances:
        result -= abs(d-1)
    return result
```

Listing 1. Fitness function for searching reflexive polytopes

To create a new Generation through crossover, the previous Generation must be copied with deepcopy. After deleting the last half with worse fitness values than the first half, a random selection out of the remaining half is performed to get two parents, which create one new child through crossover. For the crossover a random one-point crossover function is used which generates a random split point and combines the two chromosomes in a new way. The new created solutions are added to the reduced Generation until it is filled up again.

This process is repeated, untill the set number of Generations is created.

3. Preliminary results

Without any optimization of our genetic algorithm we already found a new reflexive polytope with 7 vertices, which is not included in the data set [2] as of today 2023-12-29.

The matrix containing the vertices is given below.

$$\begin{pmatrix} -2 & 1 & 4 & 1 & -3 & 1 & -2 \\ 3 & -2 & -4 & 2 & 3 & 2 & -1 \\ 1 & -1 & 1 & 3 & -1 & 3 & -3 \\ -3 & 2 & 2 & 0 & -2 & -3 & 0 \\ 0 & 0 & -2 & -1 & 2 & -1 & 2 \end{pmatrix} \tag{1}$$

Its interior point is $\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \end{pmatrix}$. Figure 1 displays the new found polytope, projected from 5D into 3D.

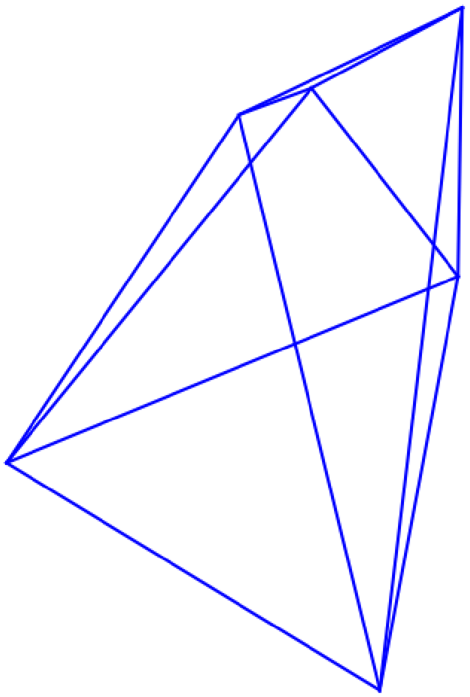

FIGURE 1. Found new polytope given by matrix (1)

The reflexivity feature of this polytope can be verified using Sage as demonstrated by Listing 2 or using the CYTools framework [3] as shown by Listing 3, both given in the appendix A. The complete code is available at GitHub [4].

4. Conclusion and outlook

In the next steps, we optimize the genetic search algorithm and increase the search space to larger polytopes, that is, we expand the search for polytopes that have larger coordinates.

We are also expanding the visualization framework to provide a better and unambiguous graphical representation of the results.

Appendix A. Listings

```
p = LatticePolytope([(-2, 3, 1, -3, 0), (1, -2, -1, 2, 0), (4, -4, 1,
    2, -2), (1, 2, 3, 0, -1), (-3, 3, -1, -2, 2), (1, 2, 3, -3, -1),
    (-2, -1, -3, 0, 2)])
p.is_reflexive()
```

Listing 2. Verify the polytope's reflexivity using Sage

```
vertices = [[-2, 3, 1, -3, 0],
 [1, -2, -1, 2, 0],
 [4, -4, 1, 2, -2],
 [1, 2, 3, 0, -1],
 [-3, 3, -1, -2, 2],
 [1, 2, 3, -3, -1],
 [-2, -1, -3, 0, 2]]
p = Polytope(vertices)
print(p.is_reflexive())
```

Listing 3. Verify the polytope's reflexivity using CYTools

References

[1] Per Berglund, Yang-Hui He, Elli Heyes, Edward Hirst, Vishnu Jejjala, and Andre Lukas. New calabi-yau manifolds from genetic algorithms. *arXiv preprint arXiv:2306.06159*, 2023.

[2] Elli Heyes. Polytope-generation. https://github.com/elliheyes/Polytope-Generation/tree/main/Data, 2023. Accessed: 2023-12-29.

[3] Mehmet Demirtas, Andres Rios-Tascon, and Liam McAllister. Cytools: A software package for analyzing calabi-yau manifolds. *arXiv preprint arXiv:2211.03823*, 2022.

[4] Eldar Sultanow. Polytopes. https://github.com/Sultanow/polytopes, 2023. Accessed: 2023-12-29.

Eldar Sultanow, Capgemini, Bahnhofstrasse 30, 90402 Nuremberg, Germany
Email address: eldar.sultanow@capgemini.com

Email address: alrahn@gmx.de