Documentation for the computer program that finds all oriented matroids of a given rank and number of elements

Nevena Palić

September 5, 2018

1 Introduction

This is a documentation for the computer program written in the programming language C that finds all vertices of MacPhersonians, thus all oriented matroids of a given rank and on a given number of elements. The largest parameters it has been run on are rank $r = 3$ and number of elements $n = 7$. Note that this version of the code works only for parameters n and r such that $\binom{n}{r} \leq 64$.

Here we make use of representatives of reorientation classes of oriented matroids found by Finschi [2]. For every uniform oriented matroid $\mathcal{M}$ presented in [2], we find all oriented matroids $\mathcal{M}'$ such that $\mathcal{M} \rightsquigarrow \mathcal{M}'$, i.e., such that $\mathcal{M}$ weakly maps to $\mathcal{M}'$. This is done by checking for each subset of bases of $\mathcal{M}$ whether it satisfies the chirotope axioms [1, Def. 3.5.3 & Lemma 3.5.4]. In the end, we construct all oriented matorids that are obtained from such oriented matroids $\mathcal{M}'$ by permuting and reorienting its elements. As a result, we obtain all oriented matroids of rank r on n elements.

2 The code

The main object that we consider is an oriented matroid. For fixed integers r and n with properties $n \geq r \geq 2$, we work with rank r oriented matroids on the set of elements $[n] = \{1, 2, \ldots, n\}$. Note that the parameters r and n usually do not get changed during the run of one code. Every oriented matroid gets encoded as a chirotope. Since there are always two chirotopes that correspond to each oriented matroid, we work with the one whose lexicographically largest basis is positive. This is just a way to avoid storing both chirotopes for one oriented matroid, but it is not essential in the code.

In the code the rank is stored in the variable `R` and the number of elements is stored in the variable `N`.

Oriented matroids are stored as chirotopes, presented by two B-tuples whose entries can be only 0 and 1, where $B = \binom{n}{r}$, the number of bases $(a_1, a_2 \ldots, a_r)$

such that $a_1 < a_2 < \cdots < a_r$. Since chirotopes are alternating functions, these bases suffice to recover the whole chirotope. The first tuple encodes the bases that are positive and the second tuple encodes the bases that are negative in the given chirotope. In particular, it cannot happen that both tuples have at the same position the entry 1.

The first 32 entries of the tuple that encodes the positive bases are stored in `plus[0]`, the next 32 entries are stored in `plus[1]`, etc. Similarly the first 32 entries of the tuple that encode negative bases are stored in `minus[0]`, and so on.

Most of the code is written in the library `OMs.c`, which contains all necessary functions for dealing with oriented matroids. The code is given in the following files

- `OMs.c`
- `translate_Finschi_representatives.c`
- `lower_cones.c`
- `find_all_OMs.c`

2.1 `translate_Finschi_representatives.c`

We first run this code in order to store representatives of uniform oriented matroids given in [2] in the form that is used in the rest of the program. Note that the order of bases of oriented matroids given in [2] differs from the order that we use.

2.2 `lower_cones.c`

This code gets an input argument `step` (in order to be runnable in parallel), and finds all oriented matroids in the lower cone (thus weak images) of the `step`-th oriented matroid $\mathcal{M}$ from the list of uniform representatives. The function makechirotopes() checks for every subset of the bases of $\mathcal{M}$ whether it satisfies the chirotope axioms. The output are text files that contain representatives of reorientation/permutation classes of all oriented matroids of rank r on n elements. The representatives are not chosen to be unique.

2.3 `find_all_OMs.c`

This code takes representatives of oriented matroids of rank r on n elements constructed by `lower_cones.c` and makes all oriented matroids of rank r on n elements by permuting and reorienting elements. The output are text files that give all oriented matroids, sorted by the number of bases, each oriented matroid appears exactly once.

References

[1] A. Björner, M. Las Vergnas, B. Sturmfels, N. White, and G. Ziegler. *Oriented Matroids.* Encyclopedia of Mathematics and its Applications, Cambridge University Press, 1999. second edition.

[2] L. Finschi. Homepage of oriented matroids. `http://www.om.math.ethz.ch/?p=catom&filter=nondeg`.