Introduction to PALP

Johanna Knapp, Emanuel Scheidegger

University of Melbourne and Peking University, respectively

The Unreasonable Effectiveness of Toric Geometry
ESI, 18-19/06/2026

Resources

- PALP stands for **P**ackage for **A**nalysing **L**attice **P**olytopes.
- The source code (written in C) can be found at:

  `https://hep.itp.tuwien.ac.at/~kreuzer/CY/CYpalp.html`
- The original paper is:

  `https://arxiv.org/abs/math/0204356`
- There is a user manual:

  `https://arxiv.org/abs/1205.4147`
- Online documentation

  `https://hep.itp.tuwien.ac.at/~kreuzer/CY/palpwiki/PALP_online_documentation`

Package structure

- PALP consists of five executables.
- `class.x` and `cws.x` for classification of reflexive lattice polytopes – not this presentation
- `poly.x` for analysis of (reflexive) lattice polytopes and associated toric varieties and Calabi-Yau hypersurfaces, in particular computation of Hodge numbers
- `nef.x` to compute nef partitions and to compute Hodge numbers of Calabi-Yau complete intersections
- `mori.x` to deal with Mori cones and to compute intersection rings and second Chern classes using polytope triangulations
  - Some options need `Singular`:

    `https://www.singular.uni-kl.de/`

Plan for the rest of the talks

- Part 1
  - General input/output
  - Basic functions of `poly.x`, `nef.x`, `mori.x`
  - Focus on reflexive polytopes and Calabi-Yaus
- Part 2
  - More advanced applications such as:
  - Extracting specific nef partitions
  - Quotients by discrete groups
  - Fibration analysis
  - Landau-Ginzburg models

Global.h

- PALP works with global variables defined in `Global.h`
- This values can be modified prior to compilation.
- `POLY_Dmax` fixes the maximal dimension of a polytope:

```
#define  POLY_Dmax  6  /* max dim of polytope*/
```

- Optimal numbers for maximal numbers of points, vertices, etc for a given polytope dimension.

```
#if (POLY_Dmax <= 3)
#define   POINT_Nmax 40  /* max number of points */
#define   VERT_Nmax  16  /* max number of vertices */
#define   FACE_Nmax  30  /* max number of faces */
#define   SYM_Nmax   88  /* cube: 2^D*D! plus extra */
```

- Needs adjustment for complete intersections of higher codimension.

Using PALP today

- The original PALP is almost 25 years old.
- PALP has been optimised for efficiency and limited computing resources.
- With today's computing resources, this leads to some drawbacks, such as making an optimal choice for global parameters prior to compilation.
- This leads to limitations with PALP integration into other computer algebra systems.
- By now, there are other tools that have functionalities overlapping with PALP, eg. CYtools, cohomCalg, FTheoryTools,...
- There are still many reasons to use PALP:
  - Fast and reliable algorithms to deal with lattice polytopes.
  - Functionalities for CY complete intersections.
  - Handling of large datasets.
  - Output that can easily be integrated into other software.

General input I

- `poly.x`, `nef.x`, and `mori.x` take a polytope or a weight system as input.

```
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
```

- Polytope input
  - Enter number of lines and columns.
  - The smaller number is interpreted as the dimension of the polytope.
  - Enter the points.
- Example:

```
./poly.x
Degrees and weights 'd1 w11 w12 ... d2 w21 w22 ...'
or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
2 3
Type the 6 coordinates as dim=2 lines with #pts=3 columns:
2 0 0
0 2 0
M:6 3 F:3
```

General input II

- Weight input
  - For each weight vector, enter the degree, followed by the weights.
- Example (Quintic)

```
  ./poly.x
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
5 1 1 1 1 1
5 1 1 1 1 1 M:126 5 N:6 5 H:1,101 [-200]
```

- Errors
  - Wrong input will lead to an error message and will end the program.
  - Example:

    ```
    3 3
    The number of points must be larger than the dimension!
    ```

- Except for `mori.x`, the input is interpreted as defining the M-lattice polytope.

Useful input options

- PALP makes it easy to pipe results into other programs or other PALP options.
- The option `-f` suppresses the input prompts.
  - This is important for entering large datasets via files.
- The text after the input of the polytope dimensions is ignored.
  - This helps with piping PALP output back into PALP.
- Example:

```
./poly.x -f
4 5  Vertices of P
   -1    4   -1   -1   -1
   -1   -1    4   -1   -1
   -1   -1   -1    4   -1
   -1   -1   -1   -1    4
M:126 5 N:6 5 H:1,101 [-200]
```

poly.x

- `poly.x` can be called with various options. For a full list, type

  ```
  ./poly.x -h
  ```

- General Output (no option set, or option -g)

```
  ./poly.x
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
8 1 1 2 2 2
8 1 1 2 2 2 M:105 5 N:7 5 H:2,86 [-168]
```

  - Points and vertices of the M-lattice polytope
  - Points and vertices of the N-lattice polytope
  - Hodge numbers $h^{1,i}$
  - Euler number
  - Note: $\mathbb{P}^4_{11222}[8]$ is singular. PALP completes the polytope by adding points on facets.

Listing points and vertices

- `poly.x -v` lists vertices of the M-lattice polytope

```
8 1 1 2 2 2
4 5  Vertices of P
   -1    7   -1   -1   -1
   -1   -1    3   -1   -1
   -1   -1   -1    3   -1
   -1   -1   -1   -1    3
```

- `poly.x -p` lists points of the M-lattice polytope

```
8 1 1 2 2 2
105 4  Points of P
-1 -1 -1 -1
0 -1 -1 -1
1 -1 -1 -1
2 -1 -1 -1
3 -1 -1 -1
4 -1 -1 -1
[...]
```

Points of the dual polytope

- `poly.x -e` lists vertices of the N-lattice polytope

```
8 1 1 2 2 2
5 4  Vertices of P-dual <-> Equations of P
   1   0   0   0
   0   0   1   0
   0   1   0   0
   0   0   0   1
  -1  -2  -2  -2
```

- `poly.x -d` lists points of the N-lattice polytope

```
8 1 1 2 2 2
4 7  Points of P-dual
   1   0   0   0  -1   0   0
   0   0   1   0  -2  -1   0
   0   1   0   0  -2  -1   0
   0   0   0   1  -2  -1   0
```

  - Vertices are listed first, then points, the interior point is last.

Comparing polytopes

- For classification purposes, it is important to know if weight systems/polytopes are the same up to lattice automorphisms.
- `poly.x -N` provides an upper triangular normal form of the polytope.
- Example: Two weight systems with $(h^{1,1}, h^{2,1}) = (491, 11)$:

```
  3402 40 41 486 1134 1701
4 5  Normal form of vertices of P    perm=43210
   1   0   0   0 -42
   0   1   0   0 -28
   0   0   1   0 -12
   0   0   0   1  -1
[...]
3486 41 42 498 1162 1743
4 5  Normal form of vertices of P    perm=43210
   1   0   0   0 -42
   0   1   0   0 -28
   0   0   1   0 -12
   0   0   0   1  -1
```

Weight system from a polytope

- This happens to be an option of `cws.x`:

  ```
  cws.x -N
  ```

  - The input is the N-lattice polytope.
- Example: Octic

```
  ./cws.x -N
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
4 5
Type the 20 coordinates as dim=4 lines with #pts=5 columns:
    0   -1    0    0    1
    0   -2    1    0    0
    0   -2    0    1    0
    1   -2    0    0    0
8 2 1 2 2 1
```

nef.x

- `nef.x` computes nef-partitions and Hodge numbers for complete intersection Calabi-Yaus.
- Recall:

$M_{\mathbb{R}}$ $\qquad\qquad$ $N_{\mathbb{R}}$

$$\Delta = \Delta_0 + \ldots + \Delta_{r-1} \qquad \Delta^* = \langle \nabla_0, \ldots, \nabla_{r-1} \rangle$$

$$(\Delta_l, \nabla_{l'}) \geq -\delta_{l\,l'}$$

$$\nabla^* = \langle \Delta_0, \ldots, \Delta_{r-1} \rangle \qquad \nabla = \nabla_0 + \ldots + \nabla_{r-1}$$

- Hodge numbers can be computed combinatorially.

[Batyrev-Borisov 95][Kreuzer-Riegler-Sahakyan 01]

Functionality of `nef.x`

- The input is a higher dimensional toric ambient variety given in terms of points is a polytope or a weight system.
- `nef.x` computes all inequivalent nef partitions and Hodge numbers.
  - Use option `-p` to suppress Hodge number calculation.
- The default setting is codimension 2.
  - Use option `-c#` for codimension #.
  - Option `-c1` is the hypersurface case. This calls `poly.x` but provides some nice display options of `nef.x`
- `nef.x` computes Gorenstein polytopes.
- Limitations:
  - Slows down considerably at higher codimension.
  - `Global.h` needs to be optimised for the problem.
  - Slow Hodge number calculation.

Basic output

- Example: codimension 2 complete intersections in $\mathbb{P}^5$

```
./nef.x
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
6 1 1 1 1 1 1
6 1 1 1 1 1 1 M:462 6 N:7 6  codim=2 #part=3
H:1 73 [-144] P:0 V:3 4 5       0sec  0cpu
H:1 89 [-176] P:1 V:4 5        0sec  0cpu
np=2 d:0 p:1    0sec     0cpu
```

  - Point and vertex number of the ambient M- and N-lattice polytope.
  - 2 nef partitions `P:#` with $(h^{1,1}, h^{2,1}, [\chi])$.
  - Since we are in codimension 2, only N-lattice vertex labels of one element of the partition are given.
  - Would be nice to have the vertices explicitly...

-Lv: N-lattice vertices and relations

- Example

```
    ./nef.x -Lv
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
6 1 1 1 1 1 1
6 1 1 1 1 1 1 M:462 6 N:7 6  codim=2 #part=3
5 6 Vertices in N-lattice:
   -1    0    0    0    0    1
   -1    0    1    0    0    0
   -1    0    0    1    0    0
   -1    0    0    0    1    0
   -1    1    0    0    0    0
----------------------------
    1    1    1    1    1    1  d=6  codim=0
H:1 73 [-144] P:0 V:3 4 5   (3 3)     0sec  0cpu
H:1 89 [-176] P:1 V:4 5   (2 4)     0sec  0cpu
np=2 d:0 p:1    0sec     0cpu
```

  - This also provides the hypersurface degrees as (d_1, d_2): $\mathbb{P}^5[3,3]$, $\mathbb{P}^5[2,4]$.

-Lp: N-lattice points and relations

- Calling `nef.x` with options -Lp and -c1 gives a nice display of all relations.
- Example: Octic

```
./nef.x -Lp -c1
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
8 1 1 2 2 2
8 1 1 2 2 2 M:105 5 N:7 5  codim=1 #part=1
4 7  Points of Poly in N-Lattice:
    0   -1    0    0    1    0    0
    0   -2    1    0    0   -1    0
    0   -2    0    1    0   -1    0
    1   -2    0    0    0   -1    0
-----------------------------------
    2    1    2    2    1    0  d=8  codim=0
    1    0    1    1    0    1  d=4  codim=1
H:2 86 [-168] P:0  (8) (4)     0sec  0cpu
np=1 d:0 p:0    0sec     0cpu
```

Higher codimension

- Example: $\mathbb{P}^6[2,2,3]$

```
      ./nef.x -Lp -c3
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
7 1 1 1 1 1 1 1
7 1 1 1 1 1 1 1 M:1716 7 N:8 7  codim=3 #part=4
6 8  Points of Poly in N-Lattice:
   -1    0    0    0    0    0    1    0
   -1    0    1    0    0    0    0    0
   -1    0    0    1    0    0    0    0
   -1    0    0    0    1    0    0    0
   -1    0    0    0    0    1    0    0
   -1    1    0    0    0    0    0    0
----------------------------------------
    1    1    1    1    1    1    1  d=7  codim=0
H:1 73 [-144] P:0 V0:3 4  V1:5 6   (2 2 3)     51sec  50cpu
np=1 d:0 p:3    0sec     0cpu
```

Hodge numbers

- The option -H displays the full Hodge diamond (also for hypersurfaces)
- `echo "10 1 1 1 1 2 2 2" | ./nef.x -c1 -H -f`
  `10 1 1 1 1 2 2 2 M:1806 7 N:8 7  codim=1 #part=1`

```
                    h 0 0

                h 1 0     h 0 1

            h 2 0     h 1 1     h 0 2

        h 3 0     h 2 1     h 1 2     h 0 3

    h 4 0     h 3 1     h 2 2     h 1 3     h 0 4

h 5 0     h 4 1     h 3 2     h 2 3     h 1 4     h 0 5

    h 5 1     h 4 2     h 3 3     h 2 4     h 1 5

        h 5 2     h 4 3     h 3 4     h 2 5

            h 5 3     h 4 4     h 3 5

                h 5 4     h 4 5

                    h 5 5
```

Hodge numbers

- The option `-H` displays the full Hodge diamond (also for hypersurfaces)
- `echo "10 1 1 1 1 2 2 2" | ./nef.x -c1 -H -f`
  `10 1 1 1 1 2 2 2 M:1806 7 N:8 7  codim=1 #part=1`

```
                    1
                0       0
            0       1       0
        0       0       0       0
    0       0       2       0       0
1       1751    19252   19252   1751    1
    0       0       2       0       0
        0       0       0       0
            0       1       0
                0       0
                    1
```

Gorenstein cone

- Options -g# and -d# give the points of the Gorenstein polytope for every nef partition in the N- and M-lattice, respectively.
  - The number #$\in \{0, 1, 2\}$ gives output options for the cone data.
- Example

```
./nef.x -g2
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
6 1 1 1 1 1 1
6 1 1 1 1 1 1 M:462 6 N:7 6  codim=2 #part=3
7 8 Points of PG: (nv=6)
    1    1    1    0    0    0    0    1
    0    0    0    1    1    1    1    0
   -1    0    0    0    0    1    0    0
   -1    0    1    0    0    0    0    0
   -1    0    0    1    0    0    0    0
   -1    0    0    0    1    0    0    0
   -1    1    0    0    0    0    0    0
[...]
```

Dual Gorenstein cone I

- Determine the polytope $\nabla^* \subset M_{\mathrm{R}}$ for c. i. X of codimension 2 in $\mathbb{P}^2 \times \mathbb{P}^1 \times \mathbb{P}^2$ of multidegrees $(3,1,0),(0,1,3)$:
- Step 1: Find the relevant partition:

```
./nef.x -N -Lv
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
5 8
Type the 40 coordinates as dim=5 lines with #pts=8 columns:
 1  0 -1  0  0  0  0  0
 0  1 -1  0  0  0  0  0
 0  0  0  1 -1  0  0  0
 0  0  0  0  0  1  0 -1
 0  0  0  0  0  0  1 -1
M:300 18 N:9 8  codim=2 #part=15
5 8 Vertices in N-lattice:
    1    0   -1    0    0    0    0    0
    0    1   -1    0    0    0    0    0
    0    0    0    1   -1    0    0    0
    0    0    0    0    0    1    0   -1
    0    0    0    0    0    0    1   -1
```

Dual Gorenstein cone II

```
-----------------------------------------
    1    1    1    0    0    0    0    0  d=3  codim=3
    0    0    0    1    1    0    0    0  d=2  codim=4
    0    0    0    0    0    1    1    1  d=3  codim=3
H:3 51 [-96] P:0 V:2 3 4 7   (1 2) (2 0) (1 2)     0sec  0cpu
H:3 51 [-96] P:1 V:2 4 6 7   (1 2) (1 1) (2 1)     0sec  0cpu
H:3 60 [-114] P:2 V:2 4 7   (1 2) (1 1) (1 2)     0sec  0cpu
H:3 51 [-96] P:3 V:2 6 7   (1 2) (0 2) (2 1)     0sec  0cpu
H:3 69 [-132] P:4 V:2 7   (1 2) (0 2) (1 2)     0sec  0cpu
H:9 27 [-36] P:5 V:3 4 6 7   (0 3) (2 0) (2 1)     0sec  0cpu
H:3 75 [-144] P:6 V:3 4 7   (0 3) (2 0) (1 2)     0sec  0cpu
H:19 19 [0] P:8 V:4 5 6 7   (0 3) (1 1) (3 0)     1sec  0cpu
H:6 51 [-90] P:9 V:4 6 7   (0 3) (1 1) (2 1)     0sec  0cpu
H:3 75 [-144] P:10 V:4 7   (0 3) (1 1) (1 2)     0sec  0cpu
H:3 75 [-144] P:13 V:6 7   (0 3) (0 2) (2 1)     0sec  0cpu
np=11 d:2 p:2    0sec     0cpu
```

- The relevant partition is `P:8` which is the one in the 8th row

Dual Gorenstein cone III

- Step 2: Compute the dual Gorenstein cone

```
nef.x -N -d1
[...]
5 8
Type the 40 coordinates as dim=5 lines with #pts=8 columns:
 1  0 -1  0  0  0  0  0
 0  1 -1  0  0  0  0  0
 0  0  0  1 -1  0  0  0
 0  0  0  0  0  1  0 -1
 0  0  0  0  0  0  1 -1
[...]
M:300 18 N:9 8  codim=2 #part=15
[...]
7 40 Points of dual PG: (nv=12)
   1   1   0   0   0   1   1   1   0   0   0   1   [...]
   0   0  -1   2  -1   0   0   0  -1   2  -1   0   [...]
   0   0   2  -1  -1   0   0   0   2  -1  -1   0   [...]
   0   1   0   0   0   1   1   0  -1  -1  -1   0   [...]
  -1  -1   0   0   0   2  -1  -1   0   0   0   2   [...]
  -1   2   0   0   0  -1  -1   2   0   0   0  -1   [...]
```

Dual Gorenstein cone IV

- There are 11 dual Gorenstein cones with the corresponding lists of points. We pick the eighth one. We check that it is the correct one by e.g. computing the Hodge numbers $(h^{11}, h^{21}) = (19, 19)$ with `nef.x -N -G` as explained below.
- ∇^* has 39 points and 12 vertices. We read off the dual nef partition $\nabla^* = \langle \Delta_1, \Delta_2 \rangle$

$$\{D_0, D_1, D_5, D_6, D_7, D_{11}\}, \{D_2, D_3, D_4, D_8, D_9, D_{10}\}$$

- The points are useful in order to get the explicit equations of X according to

$$F_\ell = \sum_{m \in \Delta_\ell \cap M} a_{\ell,m} \prod_{\ell'=1}^{r} \prod_{\rho_i \in \nabla_{\ell'} \cap N} z_i^{\langle m, \rho_i \rangle + \delta_{\ell\ell'}}$$

mori.x

- mori.x computes the Mori cone of toric varieties given by star triangulations of reflexive polytopes.
- It can also be used to compute some topological invariants of the associate Calabi–Yau complete intersection.
- Two distinct modes of operation:
  1. Option -M: The input can be arbitrary reflexive polytopes of any dimension, but a triangulation is required as input
  2. Without this option there are strong restrictions on the reflexive polytopes but triangulations are computed by mori.x

  We only discuss option 1.
- The default input is a weight system. Use option -D for a polytope as input. In either case the input is interactive: After specifying the polytope one needs to enter the number of triangulations, then for each triangulation a line starting with the number of simplices followed by the simplices encoded by a bit sequence.

Triangulations and bit sequences

- We want to work with the following example `poly.x -d` :

```
8 1 1 2 2 2
4 7  Points of P-dual
    1    0    0    0   -1    0    0
    0    0    1    0   -2   -1    0
    0    1    0    0   -2   -1    0
    0    0    0    1   -2   -1    0
```

- Label the lattice points by $1, \ldots, 6$ and the origin by 0. It is easy to see that the polytope is a simplex and point 6 lies on the edge between points 1 and 5. The unique maximal fine regular star triangulation consists of the 8 simplices

$$\{0,1,2,3,4\}, \{0,1,2,3,6\}, \{0,1,2,4,6\}, \{0,1,3,4,6\},$$
$$\{0,2,3,4,5\}, \{0,2,3,5,6\}, \{0,2,4,5,6\}, \{0,3,4,5,6\}$$

- For each simplex generate a bit sequence by setting 1 at position i if the point i is in the simplex and 0 otherwise (ignoring the origin):

```
111100 111001 110101 101101 011110 011011 010111 001111
```

The Stanley–Reisner ideal I

It is recommended to use -M only together with -D

```
mori.x -MD
'#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
4 6
Type the 24 coordinates as dim=4 lines with #pts=6 columns:
1 0 0 0 -1 0
0 0 1 0 -2 -1
0 1 0 0 -2 -1
0 0 0 1 -2 -1
4 7
    1    0    0    0   -1    0    0
    0    0    1    0   -2   -1    0
    0    1    0    0   -2   -1    0
    0    0    0    1   -2   -1    0
'#triangulations':
1
1 triangulations:
8 111100 111001 110101 101101 011110 011011 010111 001111
2 SR-ideal
100010 011101
```

The Stanley–Reisner ideal II

- The last two lines say that the Stanley–Reisner ideal has two generators, given in bit sequence form.

```
100010 011101
```

  There is a 1 in position i if the toric divisor D_i is a factor in the generator and 0 otherwise.

$$I_{SR} = \langle D_1 D_5, D_2 D_3 D_4 D_6 \rangle \subset \mathbb{Z}[D_1, \ldots, D_6]$$

The Mori cone of the ambient toric variety

- The option -m yields the Mori cone
- ```
  mori.x -MDm
  '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
  4 6
  Type the 24 coordinates as dim=4 lines with #pts=6 columns:
  1 0 0 0 -1 0
  0 0 1 0 -2 -1
  0 1 0 0 -2 -1
  0 0 0 1 -2 -1
  [...]
  8 111100 111001 110101 101101 011110 011011 010111 001111

  2 MORI GENERATORS / dim(cone)=2
    0  1  1  1  0  1   I:10
    1  0  0  0  1 -2   I:01
  ```
- The incidence structure between the generators of the Mori cone and its facets is displayed
- To get Stanley–Reisner ideal and Mori cone, use -MDgm
  ```

Choosing a specific nef partition

- By default, PALP computes all nef partitions. Sometimes, one is interested in a specific complete intersection given by a nef partition.
- `nef.x -G` allows for input of a Gorenstein cone.
  - As a weight system. See PALP user manual for examples and input format.
  - As a Gorenstein polytope.
    - M-lattice input is default, combine with `-N` for N-lattice input.
    - The expected format can be generated with `nef.x -d1` (M-lattice) and `nef.x -g1`
- Example: Suppose we only want to compute the Hodge numbers of $\mathbb{P}^5[3,3]$.

Example: Hodge numbers of $\mathbb{P}^5[3,3]$ I

- Step 1: Generate all nef partitions without computing Hodge numbers.

```
./nef.x -Lv -p
[...]
6 1 1 1 1 1 1 M:462 6 N:7 6  codim=2 #part=3
5 6 Vertices in N-lattice:
   -1    0    0    0    0    1
   -1    0    1    0    0    0
   -1    0    0    1    0    0
   -1    0    0    0    1    0
   -1    1    0    0    0    0
-----------------------------
    1    1    1    1    1    1  d=6  codim=0
 P:0 V:3 4 5   (3 3)      0sec  0cpu
 P:1 V:4 5   (2 4)      0sec  0cpu
np=2 d:0 p:1    0sec      0cpu
```

- The partition `P:0`, i.e. the first one in the list, is the one we want.

Example: Hodge numbers of $\mathbb{P}^5[3,3]$ II

- Step 2: Generate the Gorenstein polytope in the N-lattice.

```
./nef.x -g1
[...]
6 1 1 1 1 1 1 M:462 6 N:7 6  codim=2 #part=3
6 8 Points of PG: (nv=6)
    0    0    0    1    1    1    1    0
   -1    0    0    0    0    1    0    0
   -1    0    1    0    0    0    0    0
   -1    0    0    1    0    0    0    0
   -1    0    0    0    1    0    0    0
   -1    1    0    0    0    0    0    0
6 8 Points of PG: (nv=6)
    0    0    0    0    1    1    1    0
   -1    0    0    0    0    1    0    0
[...]
```

- We take the first one.

Example: Hodge numbers of $\mathbb{P}^5[3,3]$ III

- Step 3: Run the `-G -N` option.

```
./nef.x -G -N -f
6 8 Points of PG: (nv=6)
    0    0    0    1    1    1    1    0
   -1    0    0    0    0    1    0    0
   -1    0    1    0    0    0    0    0
   -1    0    0    1    0    0    0    0
   -1    0    0    0    1    0    0    0
   -1    1    0    0    0    0    0    0
M:112 12 N:8 8 H:1 73 [-144]
```

- Note the option `-f` to suppress the input prompts for easy copy/paste.

Quotients

- PALP can deal with quotients by discrete groups induced by refinements of the N-lattice.
- General input:

```
weight system /Z#: w1 w2 w3 [...] /Z#: w1 w2 w3 [...]
```

- Example 1: Quintic (reminder)

```
5 1 1 1 1 1 M:126 5 N:6 5 H:1,101 [-200]
```

- Example 2: Quintic twin

```
./poly.x
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
5 1 1 1 1 1 /Z5: 0 1 2 3 4
5 1 1 1 1 1 /Z5: 0 1 2 3 4 M:26 5 N:6 5 H:1,21 [-40]
```

- Example 3: Mirror quintic

```
5 1 1 1 1 1 /Z5: 1 4 0 0 0 /Z5: 1 0 4 0 0 /Z5: 1 0 0 4 0
M:6 5 N:126 5 H:101,1 [200]
```

Finding quotients with cws.x

- One can use `cws.x -N` to obtain weight systems and quotient groups given a polytope.
- Example: Mirror quintic

```
./cws.x -N
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
4 5
Type the 20 coordinates as dim=4 lines with #pts=5 columns:
-1 -1 -1 -1  4
-1 -1 -1  4 -1
-1 -1  4 -1 -1
-1  4 -1 -1 -1
5 1 1 1 1 1 /Z5: 4 1 0 0 0 /Z5: 4 0 1 0 0 /Z5: 4 0 0 1 0
```

Fibrations with `poly.x`

- General input:

```
./poly.x -#

./poly.x -##
```

  - # or ## specifies the maximal codimension of the fibre polytope
- Output: Points of the N-lattice polytope and a list of fibrations
  - determined a single IP simplex if # is used
  - determined all IP simplices if ## is used

  in the following format:

```
------------------------------ #fibrations=#
v    p    _    v    _    _    p  cd=#  m: # # n: # #
```

  - Number of fibrations
  - Points, vertices of fibre polytope.
  - Codimension.
  - M- and N-lattice data of fibre polytope.

Fibrations with poly.x II

- Example 1: The octic is a K3 fibration

```
./poly.x -2
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
8 1 1 2 2 2
4 7  points of P-dual and IP-simplices
    1    0    0    0   -1    0    0
    0    0    1    0   -2   -1    0
    0    1    0    0   -2   -1    0
    0    0    0    1   -2   -1    0
-----------------------------    #fibrations=1
    _    v    v    v    _    v  cd=1  m:35 4 n:5 4
```

Fibrations with poly.x III

- Example 2: An elliptic fibration

```
  ./poly.x -2
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
18 1 1 1 6 9
4 10  points of P-dual and IP-simplices
    1    0    0    0   -1    0    0    0    0    0
    0    0    1    0   -1    0    0    0    0    0
    0    1    0    0   -6   -2   -1    0   -1    0
    0    0    0    1   -9   -3   -2   -1   -1    0
-----------------------------------------    #fibrations=1
    _    v    _    v    _    v    p    p    p  cd=2  m:7 3 n:7 3
```

Fibrations with poly.x IV

- Example 2: A K3 fibration and elliptic fibration such that the K3 fiber is also elliptically fibered

```
  ./poly.x -12
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
24 1 1 2 8 12
4 10  Em:7 3 n:7 3  Km:39 4 n:9 4  M:335 5 N:11 5  p=1367892504
   1    0   -2   -1    0   -1    0   -4    0   -8
   0    1   -3   -2   -1   -1    0   -6    0  -12
   0    0    0    0    0    0    1   -1    0   -2
   0    0    0    0    0    0    0    0    1   -1
```

Fibrations with `nef.x`

- General input:

  `./nef.x -F#`

  - # specifies the maximal codimension of the fibre polytope
  - The option should be called in conjunction with `-Lv` or `-Lp` to display the vertices/points of the N-lattice polytope.
  - Caution when determining the correct codimension for complete intersections. This will depend on the nef partition.

- Construct genus one fibrations in codimension 2 toric complete intersections using combined weight systems from dimension 2 and 3:
- `./nef.x -c2 -Lp -F3`

```
Degrees and weights  'd1 w11 w12 ... d2 w21 w22 ...'
  or '#lines #columns' (= 'PolyDim #Points' or '#Points PolyDim'):
3 1 1 1 0 0 0 0  4 0 0 0 1 1 1 1
3 1 1 1 0 0 0 0  4 0 0 0 1 1 1 1 M:350 12 N:8 7  codim=2 #part=9
5 8  Points of Poly in N-Lattice:
    0    0    0    1    0    0   -1    0
[...]
---------------------------------- #fibrations=2
    v    v    _    _    v    v    _  cd=2  m: 35  4 n: 5 4
    _    _    v    v    _    _    v  cd=3  m: 10  3 n: 4 3
H:8 35 [-54] P:0 V:0 1 4   (3 1) (0 3)     0sec  0cpu
H:2 59 [-114] P:2 V:3 5 6   (1 3) (2 1)     0sec  0cpu
H:2 86 [-168] P:3 V:3 6   (0 4) (2 1)     0sec  0cpu
H:2 62 [-120] P:4 V:4 5 6   (2 2) (1 2)     0sec  0cpu
H:3 75 [-144] P:5 V:4 5   (2 2) (0 3)     0sec  0cpu
H:2 77 [-150] P:6 V:5 6   (1 3) (1 2)     0sec  0cpu
np=6 d:1 p:2    0sec     0cpu
```

Analysis of a few partitions

- The vertices of the cd=3 fiber polytope are $2, 3, 6$. For partitions `P:0,5` these vertices lie in the same nef partition, hence one equation cuts down the fiber dimension by 1, the other the base dimension by 1. Hence, we get a genus one fibration by $\mathbb{P}^2[3]$ over a hypersurface in $\mathbb{P}^3$.
- For the partition `P:0` the vertices $0, 1, 4, 5$ of the cd=2 fiber polytope have an induced nef partition $\{0, 1, 4\}, \{5\}$, hence the two equations cut down the fiber dimension by 2, and we get a second genus one fibration with fiber $\mathbb{P}^2[3]$ on the same Calabi-Yau variety. For the partition `P:5` the induced nef partition is $\{0, 1\}, \{4, 5\}$, so the fiber of the second fibration is $\mathbb{P}^3[2, 2]$.
- For partition `P:3` the vertices $0, 1, 4, 5$ of the cd=2 fiber polytope lie in the same nef partition, hence the fiber is related to $\mathbb{P}^3[4]$. But since the base is a quadric in $\mathbb{P}^2$, the fiber actually consists of two copies of $\mathbb{P}^3[4]$.

Landau-Ginzburg models I

- `poly.x` can deal with Landau-Ginzburg orbifolds
- Landau-Ginzburg orbifolds, specified by a quasihomogeneous function (superpotential) can appear in the stringy Kähler moduli space of Calabi-Yaus.
- There is a full classification. [Kreuzer-Skarke 92]
  - Total number: $10,839$.
  - $7,555$ can be associated to Calabi-Yau hypersurfaces in weighted $\mathbb{P}^4$.
- State spaces of LG orbifolds are labeled by group elements of the orbifold group: twisted sectors
- Dimensions of each twisted sector can be computed.

[Vafa 89][Intriligator-Vafa 90]

- Dimensions match the Hodge numbers of the associated Calabi-Yau hypersurface.

poly.x -L: Landau-Ginzburg spectra

- Example 1: Quintic Landau-Ginzburg Model ($\mathbb{Z}_5$ orbifold)

```
./poly.x -L
type degree and weights  [d  w1 w2 ...]: 5 1 1 1 1 1
sec[0] th= 0  0  0  0  0  QL= 0/5 dQ= 0  q00+=1 q11+=101
                                         q22+=101 q33+=1
sec[1] th= 1  1  1  1  1  QL= 0/5 dQ= 3  q03+=1
sec[2] th= 2  2  2  2  2  QL= 5/5 dQ= 1  q12+=1
sec[3] th= 3  3  3  3  3  QL=10/5 dQ=-1  q21+=1
sec[4] th= 4  4  4  4  4  QL=15/5 dQ=-3  q30+=1
WittenIndex=-200, Trace=208
5 1 1 1 1 1 M:126 5 N:6 5 V:1,101 [-200]
```

  - Contributing sectors.
  - Orbifold action on chirals.
  - Left R-charge and difference between left and right R-charge.
  - State space dimension/contribution to Hodge numbers.
  - Witten index (Euler Number)
  - Polytope data

Landau-Ginzburg spectra II

- Example 2: Octic Landau-Ginzburg model ($\mathbb{Z}_8$-orbifold)

```
type degree and weights  [d  w1 w2 ...]: 8 1 1 2 2 2
sec[0] th= 0  0  0  0  0  QL= 0/8 dQ= 0  q00+=1 q11+=83
                                                q22+=83 q33+=1
sec[1] th= 1  1  2  2  2  QL= 0/8 dQ= 3  q03+=1
sec[2] th= 2  2  4  4  4  QL= 8/8 dQ= 1  q12+=1
sec[3] th= 3  3  6  6  6  QL=16/8 dQ=-1  q21+=1
sec[4] th= 4  4  0  0  0  QL= 6/8 dQ= 0  q11+=3 q22+=3
sec[5] th= 5  5  2  2  2  QL= 8/8 dQ= 1  q12+=1
sec[6] th= 6  6  4  4  4  QL=16/8 dQ=-1  q21+=1
sec[7] th= 7  7  6  6  6  QL=24/8 dQ=-3  q30+=1
WittenIndex=-168, Trace=180
8 1 1 2 2 2 M:105 5 N:7 5 V:2,86 [-168]
```

  - Twisted sectors contribute to the even cohomology.

Landau-Ginzburg spectra III

- Example 3: Mirror quintic via quotient.

```
type degree and weights  [d  w1 w2 ...]: 5 1 1 1 1 1
/Z5: 1 4 0 0 0 /Z5: 1 0 4 0 0 /Z5: 1 0 0 4 0
sec[0:0,0,0] th= 0  0  0  0  0  QL= 0/5 dQ= 0  q00+=1 q11+=1
                                               q22+=1 q33+=1
sec[1:0,0,0] th= 1  1  1  1  1  QL= 0/5 dQ= 3  q03+=1
sec[1:2,0,0] th= 3  4  1  1  1  QL= 5/5 dQ= 1  q12+=1
sec[1:3,0,0] th= 4  3  1  1  1  QL= 5/5 dQ= 1  q12+=1
sec[1:0,2,0] th= 3  1  4  1  1  QL= 5/5 dQ= 1  q12+=1
[...]
sec[4:0,2,3] th= 4  4  2  1  4  QL=10/5 dQ=-1  q21+=1
sec[4:2,2,3] th= 1  2  2  1  4  QL= 5/5 dQ= 1  q12+=1
sec[4:3,2,3] th= 2  1  2  1  4  QL= 5/5 dQ= 1  q12+=1
sec[4:1,3,3] th= 1  3  1  1  4  QL= 5/5 dQ= 1  q12+=1
sec[4:2,3,3] th= 2  2  1  1  4  QL= 5/5 dQ= 1  q12+=1
sec[4:3,3,3] th= 3  1  1  1  4  QL= 5/5 dQ= 1  q12+=1
WittenIndex=200, Trace=208
5 1 1 1 1 1 /Z5: 1 4 0 0 0 /Z5: 1 0 4 0 0 /Z5: 1 0 0 4 0
M:6 5 N:126 5 V:101,1 [200]
```

Papers using PALP

- Klemm, Kreuzer, Riegler, ES: "Topological string amplitudes, complete intersection Calabi-Yau spaces and threshold corrections" `hep-th/0410018`
- Braun, Kreuzer, Ovrut, ES: "Worldsheet Instantons and Torsion Curves, Part B: Mirror Symmetry" `0704.0449[hep-th]`
- Chen, JK, Kreuzer, Mayrhofer "Global SO(10) F-theory GUTs" `1005.5735[hep-th]`
- JK, Kreuzer, Mayrhofer, Walliser "Toric Construction of Global F-Theory GUTs" `1101.4908[hep/th]`
- JK, Romo, ES: "D-Brane Central Charge and Landau–Ginzburg Orbifolds" `2003.00182[hep-th]`