Counting points on curves

Edgar Costa

Dartmouth College

UNSW, 12th July 2016

(joint work with David Harvey, UNSW)

Setup

$p =$ prime

$C \subset \mathbb{P}^n_{\mathbb{F}_p} =$ algebraic curve defined over $\mathbb{F}_p$

The Hasse–Weil zeta function of C is the generating function

$$Z_C(t) := exp\left(\sum_{a>0} \frac{\#C(\mathbb{F}_{p^a})t^a}{a}\right)$$

Example

Let $p = 5$ and C be genus 3 curve given in $\mathbb{P}^2$ by

$$x^4 + xy^3 + 2y^4 - z^4 = 0$$

By naive point enumeration, we find that

$$\begin{aligned} \#C(\mathbb{F}_p) &= 9, & \#C(\mathbb{F}_{p^4}) &= 581, \\ \#C(\mathbb{F}_{p^2}) &= 29, & \#C(\mathbb{F}_{p^5}) &= 3309, \\ \#C(\mathbb{F}_{p^3}) &= 156, & \#C(\mathbb{F}_{p^6}) &= 15212, \\ &\dots \end{aligned}$$

thus:

$$Z_C(t) = 1 + 9t + 55t^2 + 304t^3 + 1579t^4 + 8029t^5 + 40404t^6 + \cdots$$

Weil conjectures for smooth curves

Indeed, $\#C(\mathbb{F}_p), \#C(\mathbb{F}_{p^2}), \#C(\mathbb{F}_{p^3})$ is sufficient to deduce $Z(t)$!

If C is smooth, we have:

$$Z_C(t) := exp\left(\sum_{a>0} \frac{\#C(\mathbb{F}_{p^a})t^a}{a}\right)$$
$$= \frac{L_C(t)}{(1-t)(1-pt)}$$

where $L_C(t) \in \mathbb{Z}[t]$ has degree $2g$, and

$$L_C(t) = 1 + a_1 t + a_2 t^2 + \cdots + a_2 p^{g-2} t^{2g-2} + a_1 p^{g-1} t^{2g-1} + p^g t^{2g}.$$

In our example,

$$L_C(t) = 1 + 3t + 6t^2 + 19t^3 + 6 \cdot 5t^4 + 3 \cdot 5^2 t^5 + 5^3 t^6$$

Naive algorithm

Naively computing

$$\#C(\mathbb{F}_p), \#C(\mathbb{F}_{p^2}), \cdots, \#C(\mathbb{F}_{p^g})$$

is not practical!

Looping over $\mathbb{F}_{p^g}$ takes at least $O(p^g)$ time.

Typical algorithm

Can we rewrite $\#C(\mathbb{F}_{p^a}) = \#\{x \in \mathbb{P}^n_{\mathbb{F}_{p^a}} : F(x) = 0\}$?

$$\cdots = \#\{x \in C(\overline{\mathbb{F}_p}) : \mathrm{Frob}^a(x) = x\}$$

One can them compute the last line with variety of methods.

- the group structure on the Jacobian
- ℓ-adic cohomology
- p-adic cohomology

Monsky–Washnitzer cohomology

For curves there is a wide range of methods using Monsky–Washnitzer cohomology, a flavour of p-adic cohomology:

- Elliptic curves (Kato–Lubkin, 1982)
- Hyperelliptic curves (Kedlaya, 2001)
- Superelliptic curves, C_{ab} curves, ...
- Smooth hypersurfaces (Abbott–Kedlaya–Roe, 2005)
- Nondegenerate curves (Denef–Castryck–Vercauteren, 2006)
- very generic variant (Tuitman, 2014)

New method

Today I will overview a new method to compute $L_C(t)$.

- Cohomology free!
- Simple
- Practical

New method - Input/Output

Input: $F(x, y, z)$ homogeneous polynomial of degree d

Output: $L_C(t)$, where C is the desingularization of the zero locus of $F(x, y, z)$ in $\mathbb{P}^2$

Assumption: The polynomials $F(0, y, z)$, $F(x, 0, y)$ and $F(x, y, 0)$ have no repeated factors.

Geometrically, the curve given by F intersects the coordinate axes ($x = 0$, $y = 0$, and $z = 0$) transversally.

New method - Ingredients

C curve of genus g that is the desingularization of $\{F = 0\} \subset \mathbb{P}^2$

Goal: compute $\#C(\mathbb{F}_{p^a})$

Approach:

- Compute $\#\{F(x, y, z) = 0 : (x, y, z) \in \mathbb{P}^2(\mathbb{F}_{p^a}),\ xyz \neq 0\}$
- Compute $\#\{F(x, y, z) = 0 : (x, y, z) \in \mathbb{P}^2(\mathbb{F}_{p^a}),\ xyz = 0\}$
- Resolve the singularities of $F(x, y, z) = 0$ (over $\overline{\mathbb{F}_p}$)

Altogether, we can deduce $\#C(\mathbb{F}_{p^a})$.

New method - Trace formula

Let $B_\ell = \{u \in \mathbb{N}^3 : \sum_i u_i = \ell\}$.

For $u \in B_{dn}$, let $(F^n)_u$ be the coefficient of $x^{u_0} y^{u_1} z^{u_2}$ in F^n.

$$(M_s)_{v,u} := \left(F^{s(p-1)}\right)_{pv-u} \quad \text{for } v, u \in B_{ds}.$$

Harvey (2015)

For p not small:

$$\#\{F(x,y,z) = 0 : (x,y,z) \in \mathbb{P}^2(\mathbb{F}_{p^a}),\ xyz \neq 0\}$$
$$= (p^a - 1)^2 \sum_{s=0}^{\lambda} \binom{\lambda}{s} \mathrm{Tr}(M_s^a) \bmod p^\lambda$$

$$M_0, M_1, \ldots, M_\lambda \rightsquigarrow \#C(\mathbb{F}_{p^a}) \bmod p^\lambda \quad \forall_a$$

New method - M_s

$$(M_s)_{v,u} := \left(F^{s(p-1)}\right)_{pv-u} \qquad v, u \in B_{ds}.$$

Harvey's trace formula gives us:

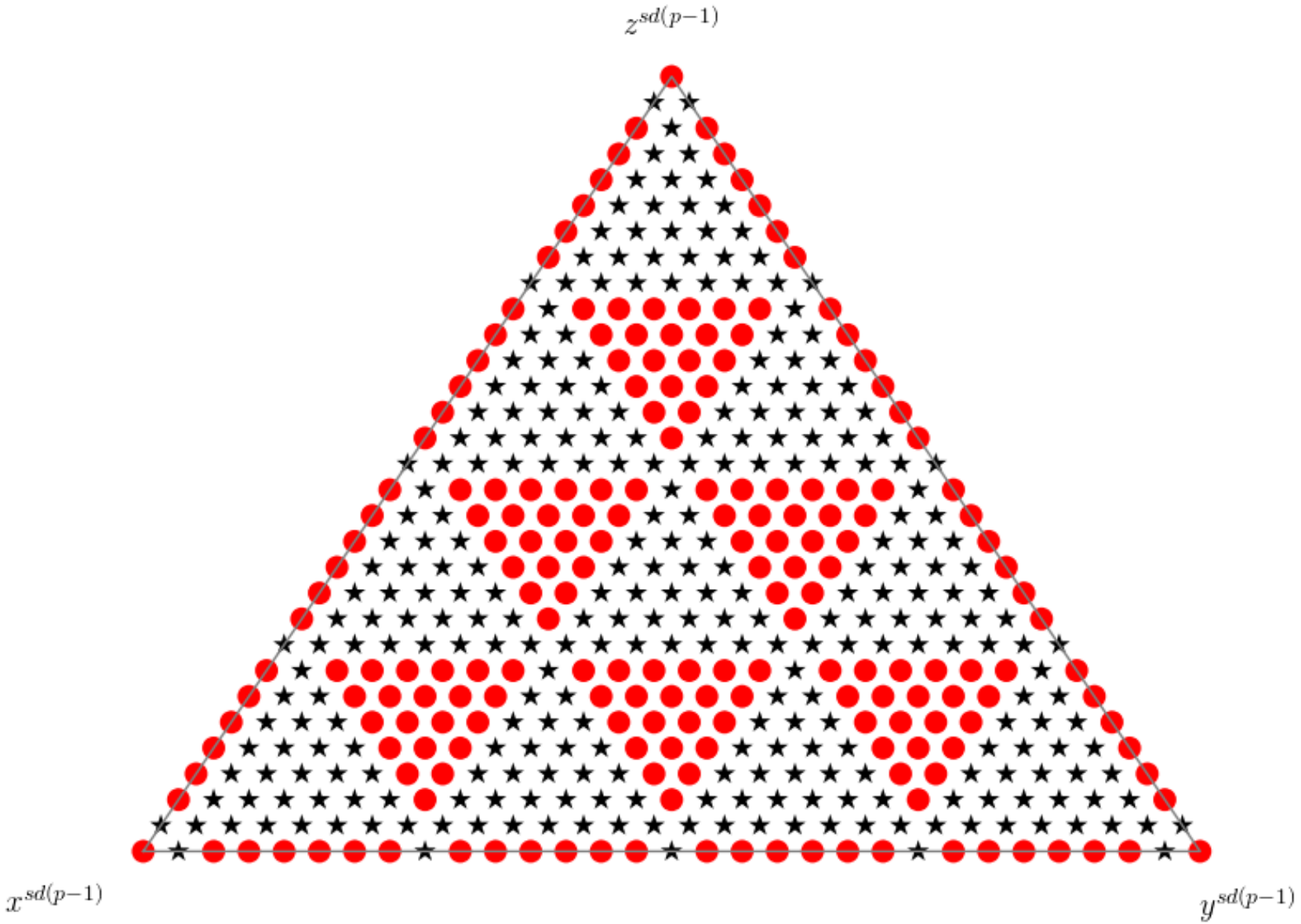
$$M_0, M_1, \ldots, M_\lambda \rightsquigarrow \#C(\mathbb{F}_{p^a}) \bmod p^\lambda$$

We can deduce $L_C(t)$ from $\#C(\mathbb{F}_{p^a}) \bmod p^{\lfloor g/2 \rfloor + 1}$, i.e., we need to compute

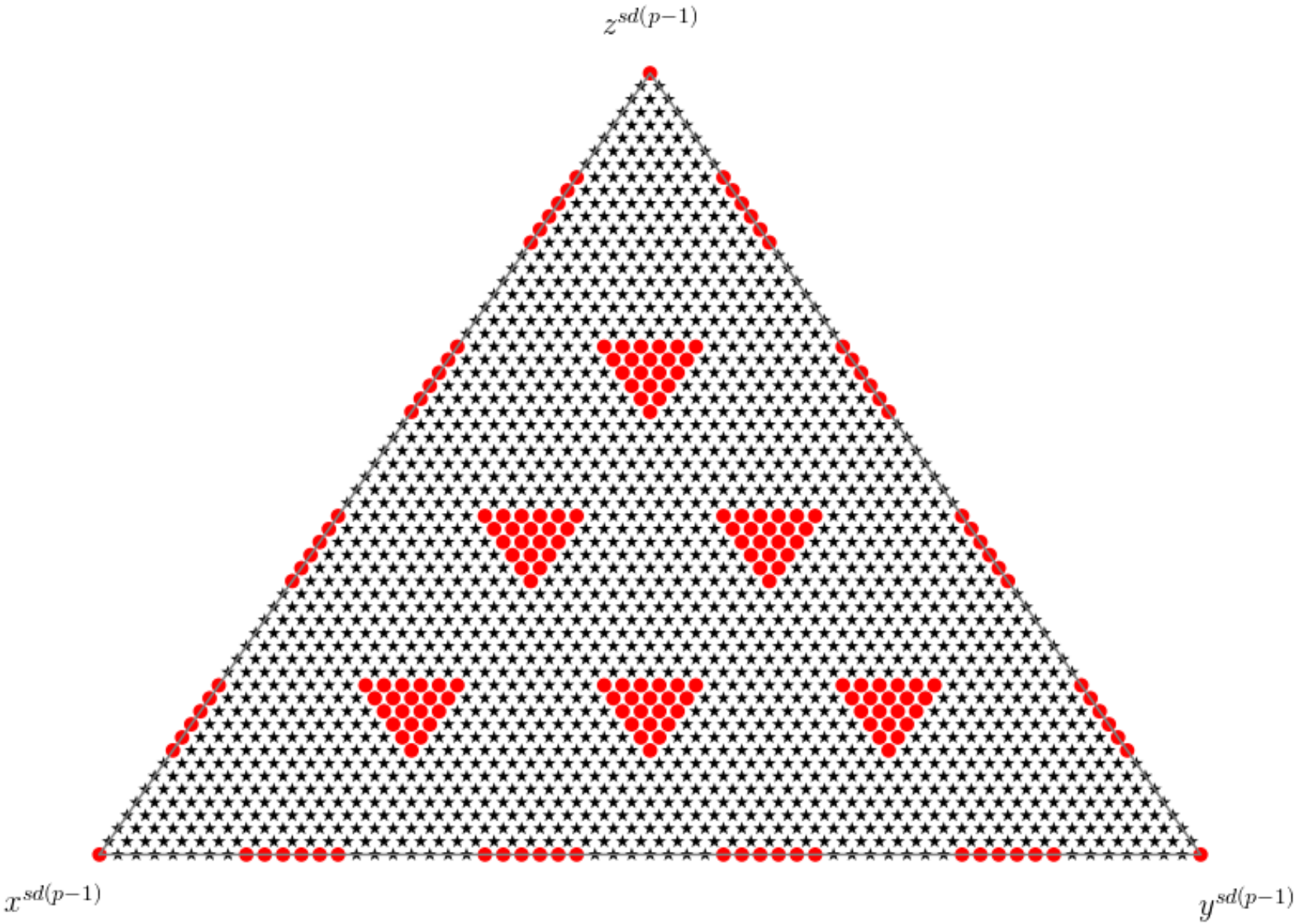
$$M_0, M_1, \ldots, M_{\lfloor g/2 \rfloor + 1}.$$

The coefficients from $F^{s(p-1)}$ needed to compute M_s are clustered around a "sublatice" of index p^2.

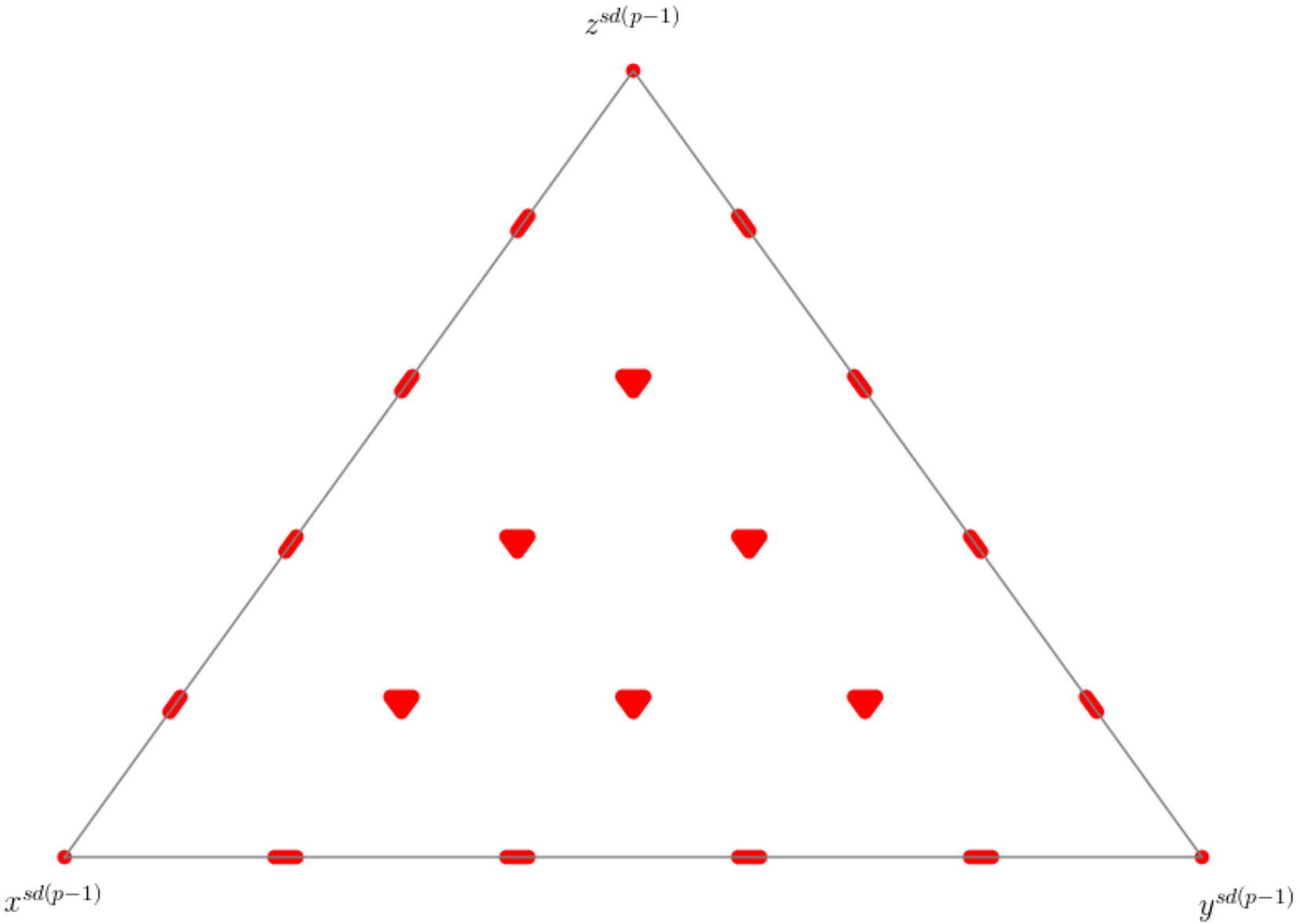
M_1 for $p = 7, d = 5$



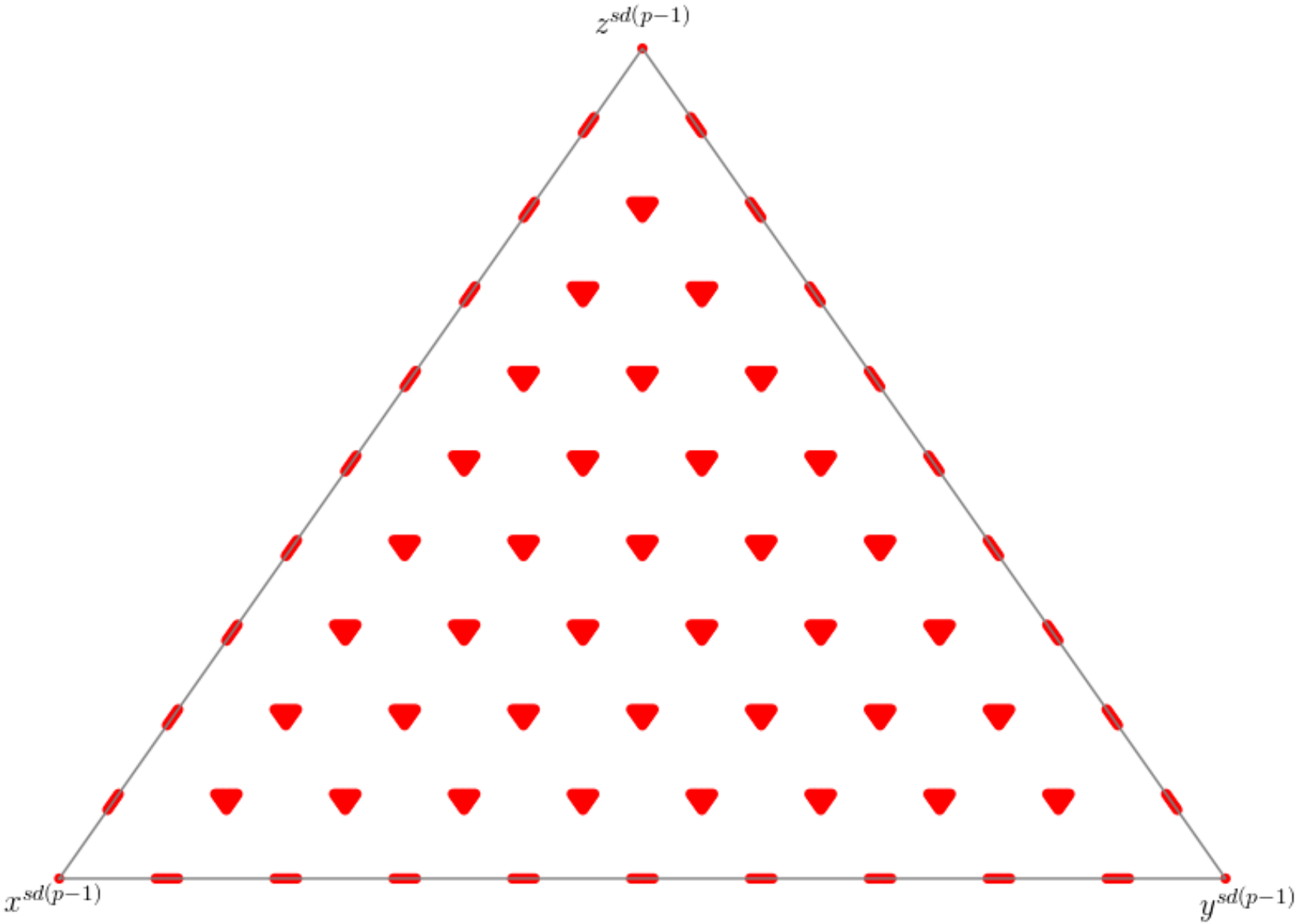
M_1 for $p = 13, d = 5$



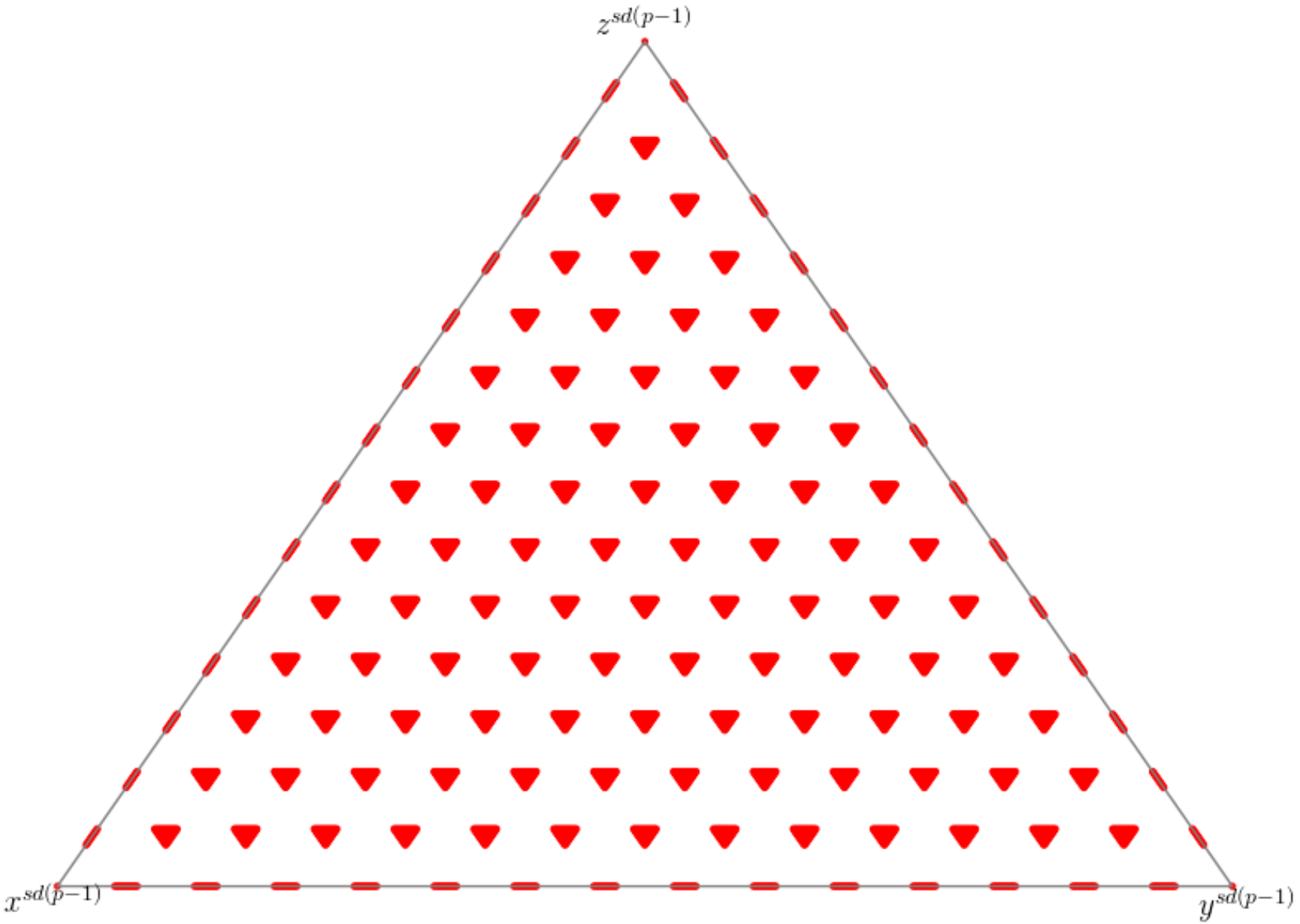
M_1 for $p = 53, d = 5$



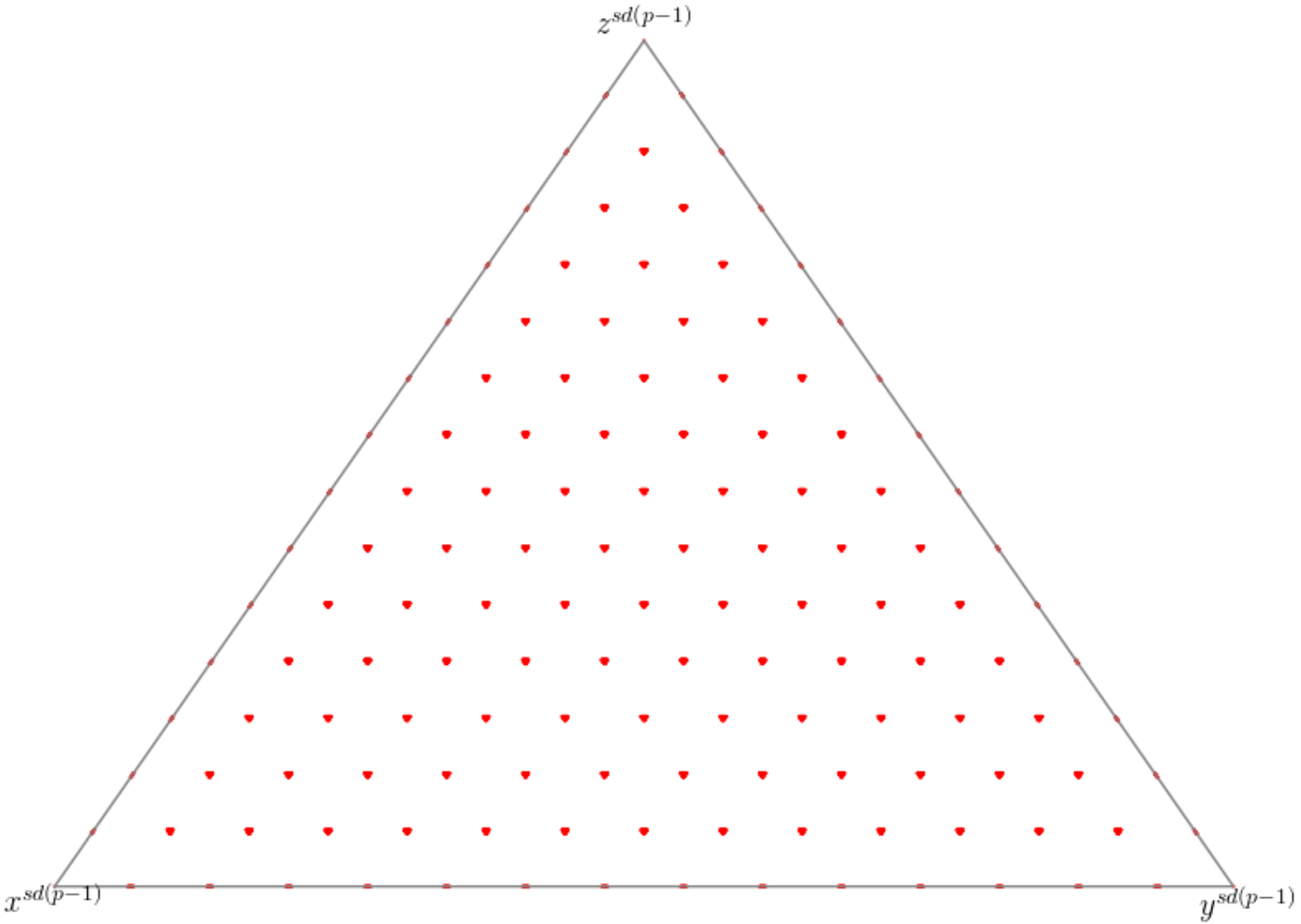
M_2 for $p = 53, d = 5$



M_3 for $p = 53, d = 5$



M_3 for $p = 199, d = 5$



New method - Key idea

There are relations between neighbouring coefficients of F^n.

Let us sketch how to derive these relations.

$$F^{n+1} = F \cdot F^n$$

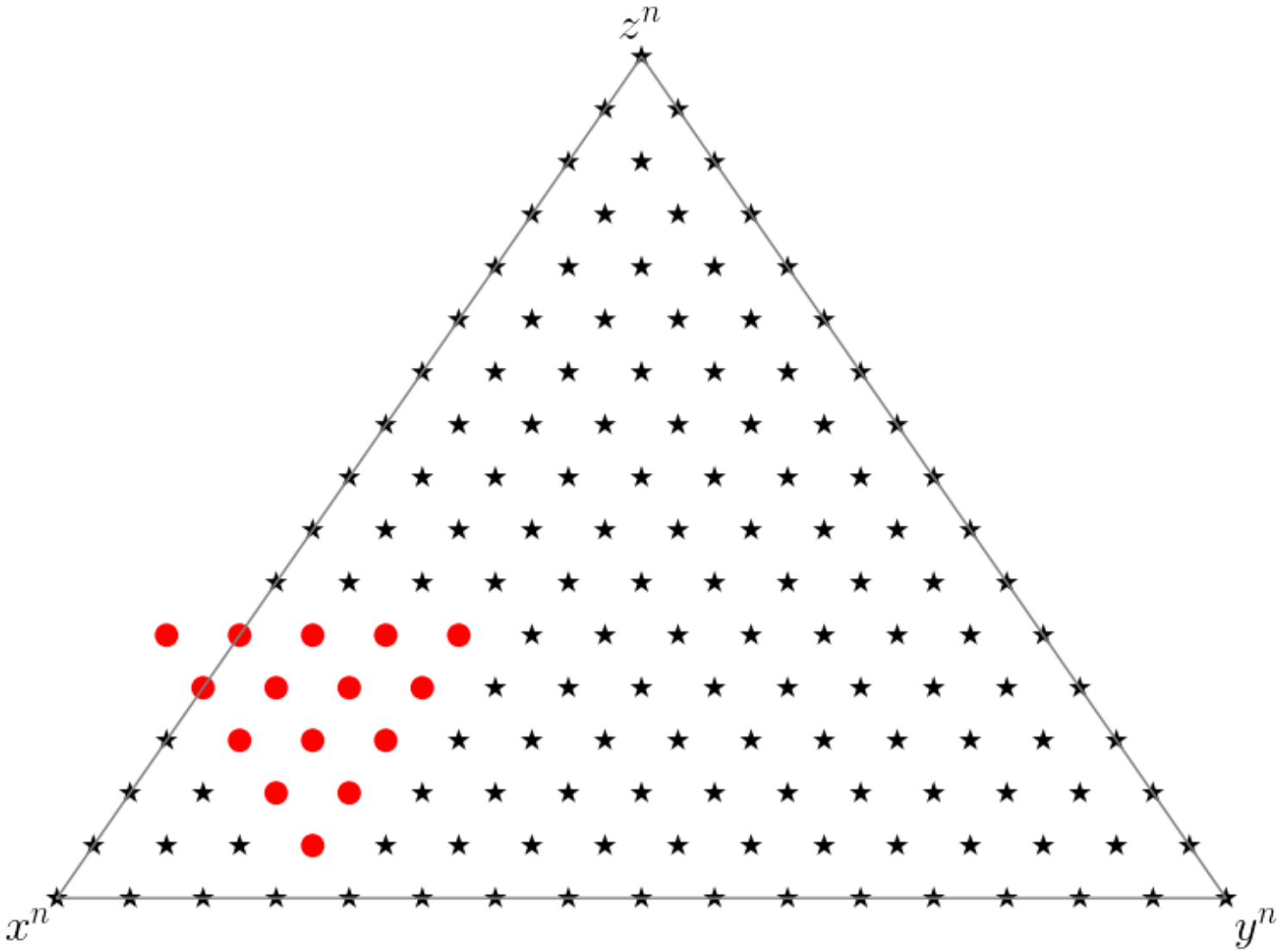
in terms of coefficients

$$(F^{n+1})_u = \sum_{t \in B_d} F_t (F^n)_{u-t}$$

this expresses a given coefficient of F^{n+1} as a linear combination of a bunch of nearby coefficients of F^n.

Example, $d = 4$ and $n = 4$

The coefficient $(F^{n+1})_{(12,3,5)}$ is known if given F^n on:



New method - Key idea

There are more relations.

Take $\partial_x = x\frac{\partial}{\partial x}$, then

$$\partial_x F^{n+1} = (n+1)\partial_x F \cdot F^n$$

in terms of coefficients

$$u_0(F^{n+1})_u = (n+1)\sum_{t\in B_d} t_0 F_t (F^n)_{u-t}$$

New method - Key idea

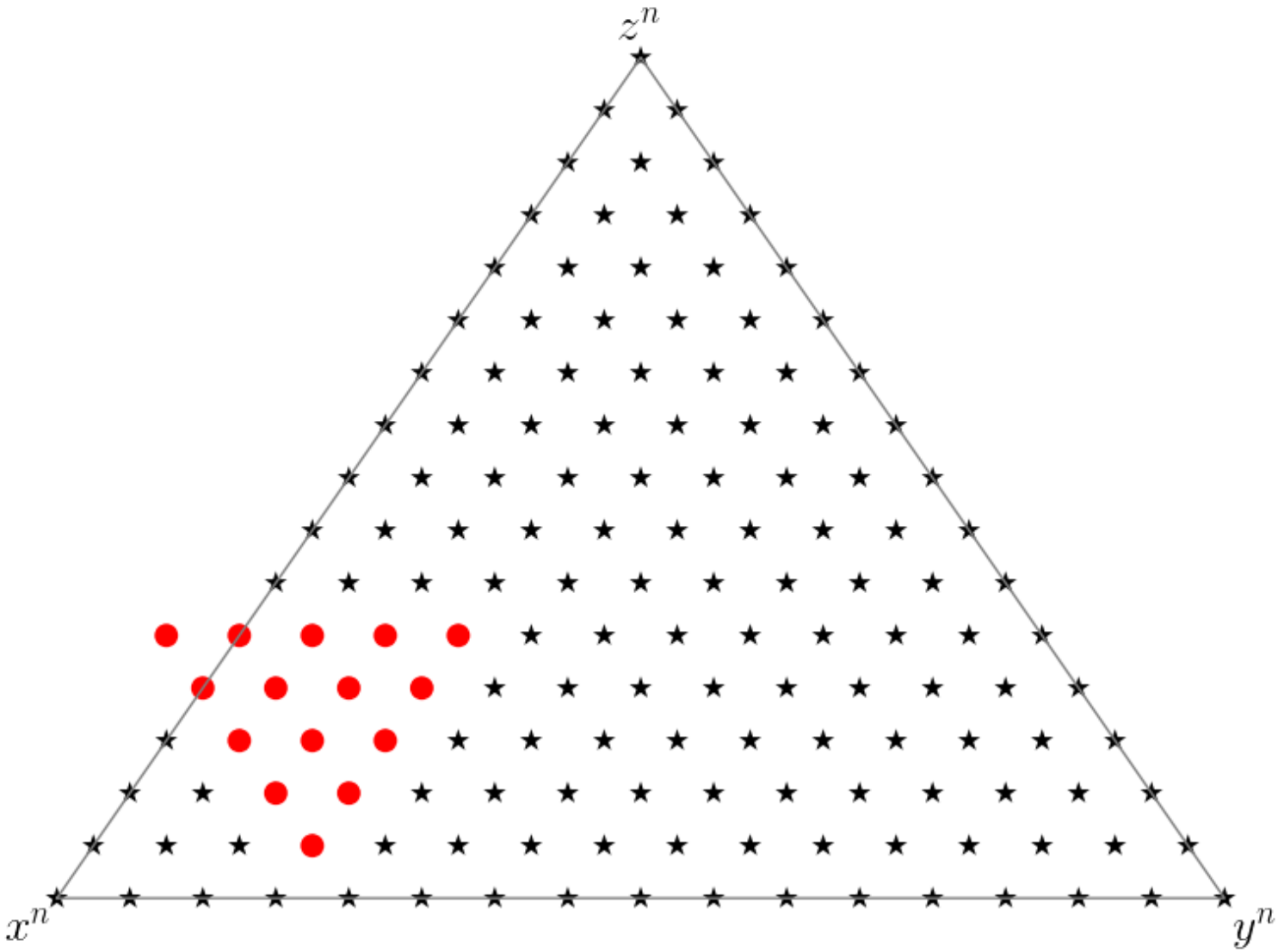
Combining both coefficient equations we can eliminate the coefficient $(F^{n+1})_u$ and we get

$$u_0 \left(\sum_{t \in B_d} F_t (F^n)_{u-t} \right) = (n+1) \sum_{t \in B_d} t_0 F_t (F^n)_{u-t}$$

for each $u \in B_{(n+1)d}$, involving the coefficients at $u - t$ for $t \in B_d$

Example, $d = 4$ and $n = 4$

There are 2 independent equations involving the red dots



New method - Key idea

Combining enough neighbouring relations we have the following.

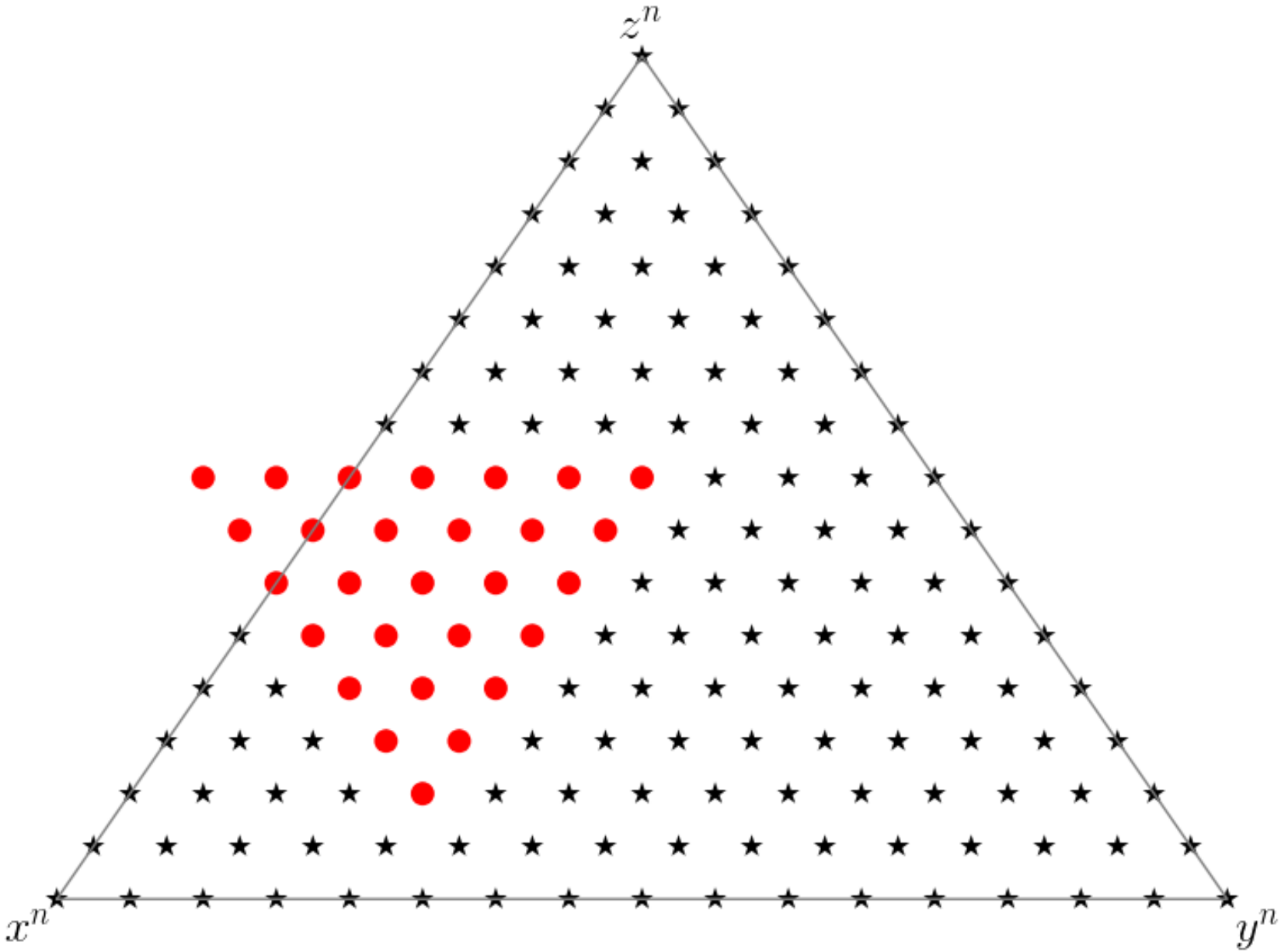
Given

$$(F^n)_{u-t} \quad t \in B_{2d-2}$$

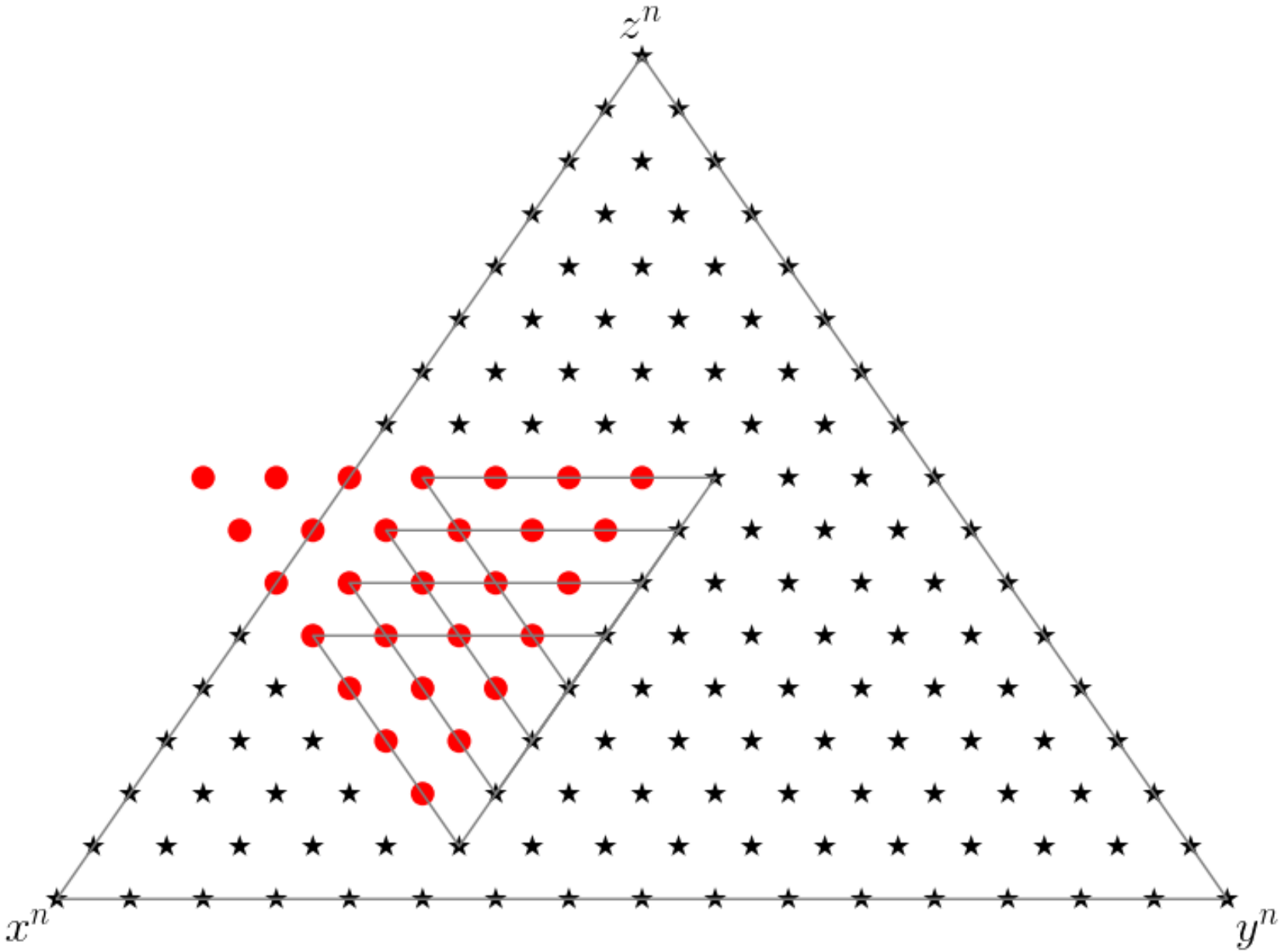
We can deduce

$$(F^n)_{(u+e_i-e_j)-t} \quad t \in B_{2d-2}$$

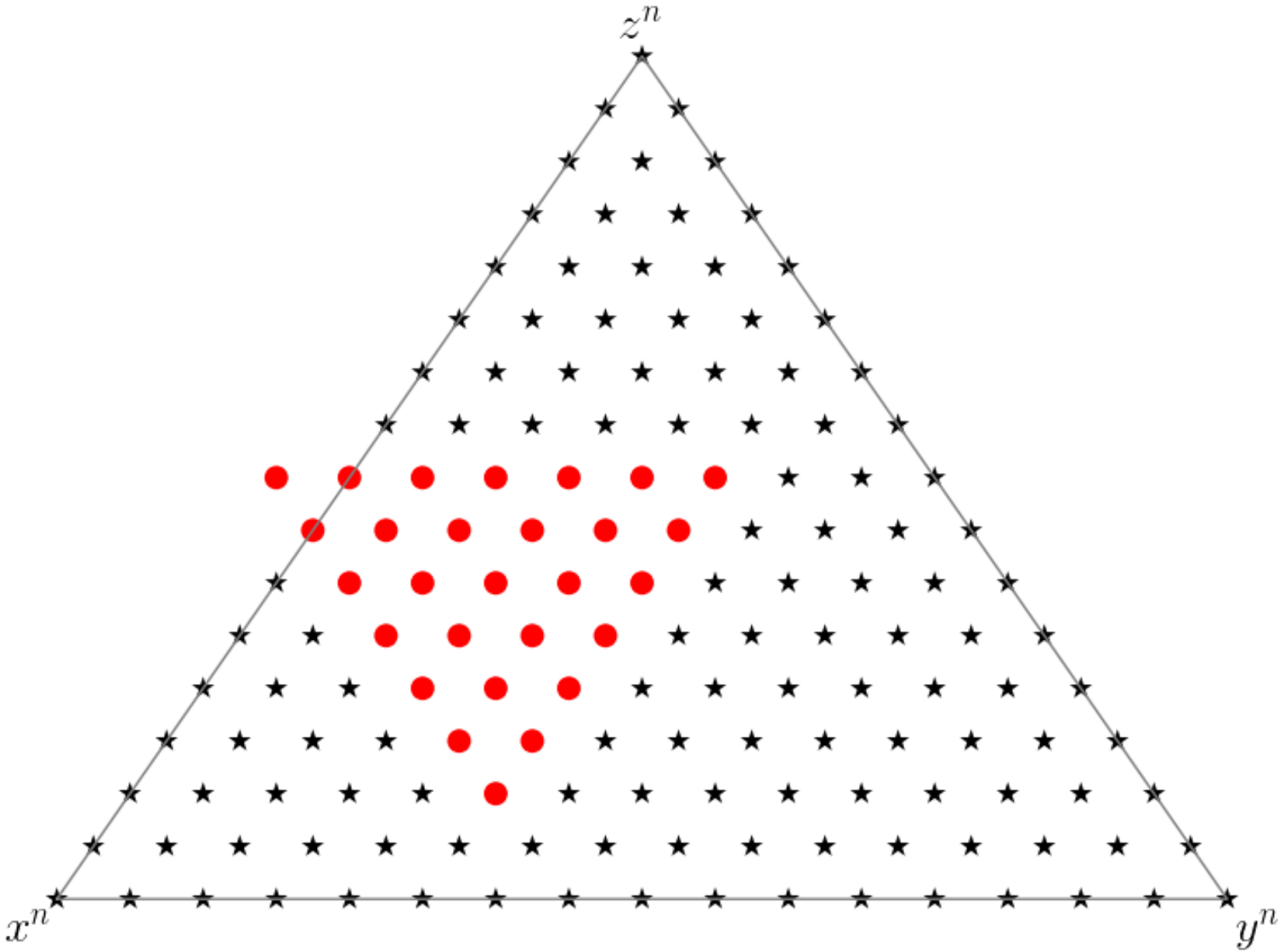
Example, $d = 4$ and $n = 4$



Example, $d = 4$ and $n = 4$



Example, $d = 4$ and $n = 4$

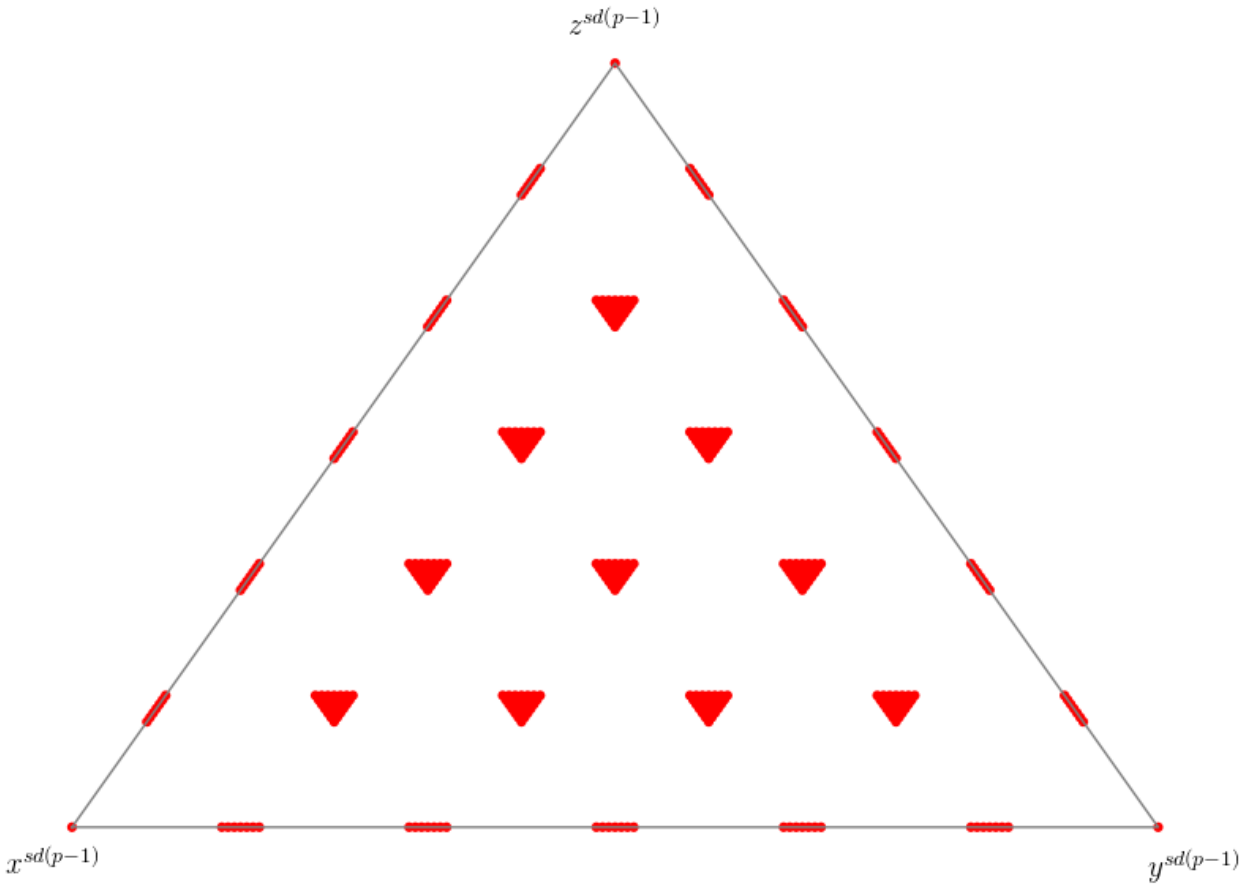


New method - Altogether

Start with a triangle at one of the vertices, where the coefficients of $F^{s(p-1)}$ are trivial to compute.

Move it around until we have computed by all the target coefficients.

Assemble M_s.



New method - Running time

Moving a triangle one step $O(d^3)$ operations.

There are $O(s^2d^2)$ clusters.

Going between clusters involves $O(p)$ steps.

Altogether, computing M_s involves $O(ps^2d^5 + s^4d^6)$ ring operations

The second factor it is for computing all the coefficients in each cluster.

New method - Precision

Computing M_s involves dividing by p numerous times.

However, for p not small, we experimentally observe that we only need work with two spare digits.

We have not yet been able to prove this.

There are also some (expensive) alternatives to avoid dividing by p.

Toy implementation in sage

Setup: $d = 6$ and genus $= 5$

$p = 1009$: 7.5 minutes

$p = 1999$: 14 minutes

Tuitman's algorithm in Magma: 8 and 24 minutes respectively.

Dislcaimer: this is NOT a fair comparison.

Future

Short term:

- C++ implementation
- precision bounds

Long term, perhaps consider the two obvious improvements:

- reduce the running time to $p^{1/2+o(1)}$.
- average polynomial time