# Gröbner-Fächer für lineare Codes

Daniel Rembold

Technische Universität Hamburg Harburg

*daniel.rembold@tuhh.de*

August 28, 2014

# Inhaltsverzeichnis

# Motivation

- Viele Anwendungen der Gröberbasen basieren auf Gröbner-Fächer
- Nützlich für Idealzugehörigkeitsproblem und polynomiale Gleichungssysteme
- Kompletter Gröbner-Fächer nicht immer nötig

# Monome

## Monom

- Produkt von Variablen über ein endliches Feld $\mathbb{K}[X_1, X_2, \ldots, X_n]$
- Schreibweise $m = X_1^{u_1} X_2^{u_2} \cdots X_n^{u_n}$ und $u_i \in \mathbb{N}_0$

Grad eines Monoms: $deg(m) = \sum_{i=1}^{n} u_i$.

# Termordnung (1)

## Termordnung $>$

- Relation $>$ zu der Menge von allen Monomen in $\mathbb{K}[X_1, X_2, \ldots, X_n]$

## Termordnung (2)

- Lexikographische Ordnung $>_{lex}$
- grad $>_{grlex}$
- Ordnung mit Gewichtsvektor $c = (c_1, \ldots, c_n) \in \mathbb{R}_+^n$

# Leitterm

## Leitterm $\mathrm{LT}(f)$

- Polynom $p \in \mathbb{K}[X_1, X_2, \ldots, X_n]$ besitzt Term höchster Ordnung in Bezug auf $>$

## Beispiel

Sei $f = x^2 + 3xyz + y^3$

- lex-Order : $f = \underline{x^2} + 3xyz + y^3$
- grlex-Order : $f = \underline{3xyz} + y^3 + x^2$
- $(1, 2, 1)$ : $f = \underline{y^3} + 3xyz + x^2$

# Ideale

## Ideal

- Kollektion von Polynomen $f_1, \ldots, f_s$ :

$$\langle f_1, \ldots, f_s \rangle = \left\{ \sum_{i=1}^{s} h_i f_i \mid h_1, \ldots, h_s \in \mathbb{K}[X_1, \ldots, X_n] \right\}.$$

## Beispiel

Sei $I = \langle f_1, f_2 \rangle = \langle x^2 + y, x + y + 1 \rangle$ und $f = x^2y + x^2 + y^2 + xy + x$
Dann gilt $f = y \cdot f_1 + x \cdot f_2, \ f \in I$.

# Divisionsalgorithmus (1)

- Notwendig zum Lösen des Idealzugehörigkeitsproblems

## Ziel des Algorithmus

Polynom g durch Ideal $I = \langle f_1, \ldots, f_s \rangle$ teilen, so dass
$g = a_1 f_1 + \ldots + a_s f_s + r, \quad a_i, g, I, r \in \mathbb{K}[X_1, \ldots, X_n]$

Sei Polynom $p$ und Ideal $I$

- Wenn p % I = 0, dann gilt $p \in I$

# Divisionsalgorithmus (2)

**Algorithm 1** Divisionsalgorithmus (Header)

**Require:** Basis $I = \langle f_1, \ldots, f_m \rangle$ of nonzero polynomials **and** a fixed termorder $\mathrm{LT}$

**Ensure:** $r = 0$ **or** none of the terms in $r$ are divisible by $LT_{\leq}(f_1), \ldots, LT_{\leq}(f_m)$

- Reihenfolge der Polynome in $I$ beeinflusst Ergebnis
- $r \neq 0$ möglich, obwohl $p \in I$

# Gröbnerbasis (1)

## Gröbnerbasis

Sei Termordnung $>$ und Ideal $I$, dann hat Gröbnerbasis $G = \{f_1, \ldots, f_m\}$ (in Bezug auf $>$) von $I$ die Eigenschaft:

- Von jedem Polynom $p \in I$ ist $\mathrm{LT}_>(p)$ teilbar durch $\mathrm{LT}_>(f_i)$

- Divisionsrest eindeutig bestimmt und unabhängig von Reihenfolge
- Gröberbasis aus jedem Ideal mit Hilfe des Buchberger-Algorithmus und einer festen Termordnung

# Gröbnerbasis (2)

- Gröbernbasen sind nicht eindeutig
- Reduzierte Gröbnerbasen jedoch eindeutig für Ideale

**Reduzierte Gröbnerbasis**

Alle Leitterme von Gröbnerbasis G in Bezug auf Termordnung $>$ monisch und relativ prim zueinander

# Gröbner-Fächer

- Unendlich viele Termordnungen, endlich viele reduzierte Gröbnerbasen

## Gröbner-Fächer

- Vielflächiges Komplex welches Kegel im $\mathbb{R}^n_+$ enthält
- Flächen werden durch Ebenenungleichungen der Polynome bestimmt

# Beispiel Gröbner-Fächer (1)

Sei

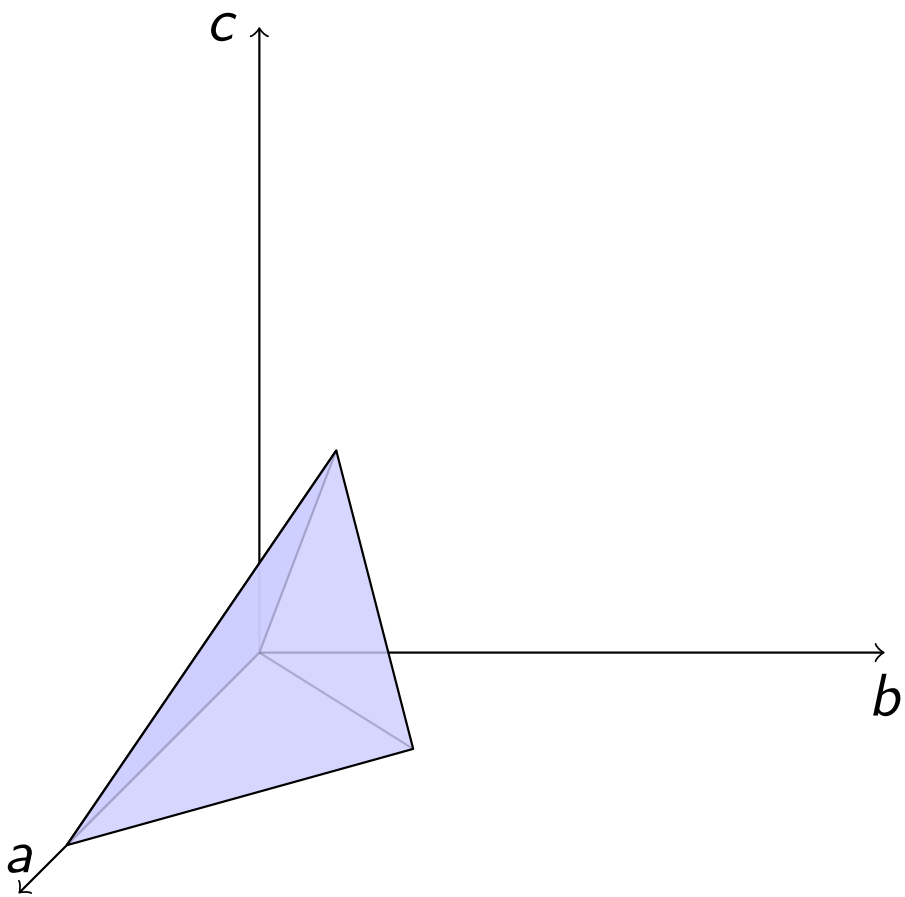
1. $I = \langle x^2 - z,\ y - x \rangle \in \mathbb{K}[x, y, z]$
2. $G_{>_{lex}} = \{\underline{y^2} - z,\ \underline{x} - y\}$
3. $\mathbf{w} = (a, b, c) \in \mathbb{R}^3_+$

- $w \in G_{>_{lex}}$ genau dann, wenn

  1. $(0, 2, 0) \cdot (a, b, c) \geq (0, 0, 1) \cdot (a, b, c) \vee 2b \geq c$
  2. $(1, 0, 0) \cdot (a, b, c) \geq (0, 1, 0) \cdot (a, b, c) \vee a \geq b$

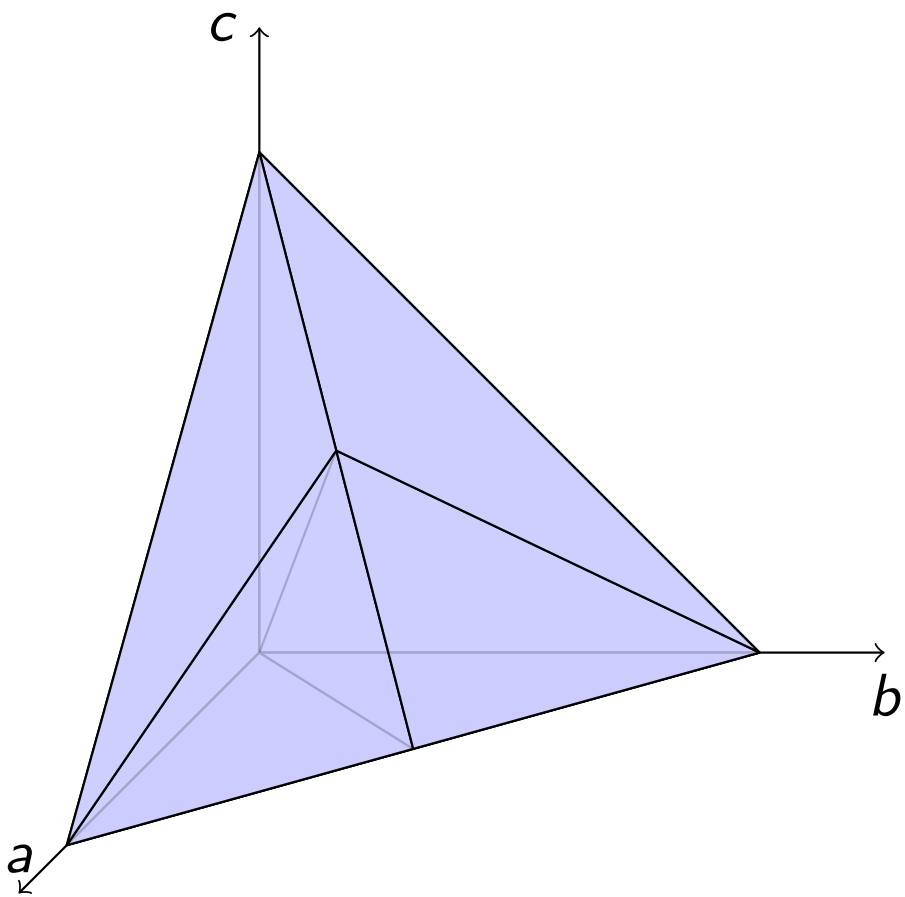
# Beispiel Gröbner-Fächer (2)

Figure: Gröbner-Kegel für $G_{>_{lex}}$



# Beispiel Gröbner-Fächer (3)

Figure: Kompletter Gröbner-Fächer



# Torische Ideale

- Obermenge von Code-Idealen
- Besteht nur aus Binomen

## Torisches Ideal

Gegeben seien $A = [a_1, \ldots, a_n] \in \mathbb{Z}^{d \times n}$ und $u \in \mathbb{Z}^n$ zerlegbar in $u^+$ und $u^-$, dann ist das torische Ideal $I_A$ definiert durch

$$\mathbf{I}_A = \langle \mathbf{x}^{u^+} - \mathbf{x}^{u^-} \mid u \in ker\,(A) \rangle.$$

# Facet Binomials

- Gröbner-Fächer als Graph
  - Gröbnerbasen als Knoten
  - Gröbnerbasen adjazent wenn im Gröbner-Fächer benachbart

## Facet Binomials

Bestimmen Grenzen zu anderen Gröbnerbasen
Sei $x^{\alpha_k} - x^{\beta_k} \in \mathcal{G}_c$ , $\mathcal{G}_c$ Gröbnerbasis mit Gewichtsvektor $c$
$x^{\alpha_k} - x^{\beta_k}$ ist Facet binomial wenn ein Vektor $u \in \mathbb{R}$ folgendes erfüllt :

- $\{\alpha_i \cdot u > \beta_i \cdot u : i = 1, \ldots, t, i \neq k\}$
- $\{\beta_k \cdot u > \alpha_k \cdot u\}$.

# Beispiel Facet Binomials

- Sei $\mathcal{G} = \{x_1 - x_5,\ x_2^2 - 1,\ x_2x_4 - x_5x_6,\ x_2x_6 - x_4x_5,\ x_3 - x_5,\ x_4^2 - 1,\ x_4x_5x_6 - x_2,\ x_5^2 - 1,\ x_6^2 - 1\}$
- Prüfe ob $x_2x_4 - x_5x_6$ ein facet binomial ist mit linearer Programmierung

$$\begin{array}{rcl} Ax & = & b \\ \text{subject to} \quad x & \geq & 0 \end{array}$$

1. $b = (0, 1, 0, 1, -1, -1)^T$
2. $$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 2 & 1 & 0 & 0 \\ -1 & 0 & -1 & 1 & 0 & 1 & 2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 2 \end{pmatrix}$$

# Traversieren von Gröbnerbasen

- Gröberbasis erhalten durch andere Gröbnerbasis
- Facet Binomial "umdrehen" (flip)

---

**Algorithm 2** Local change of reduced Gröbner bases in $I_A$

---

**Input:** Reduced Gröbner basis $\mathcal{G}$ of $I_A$ **and**
A prescribed facet binomial $\underline{x}_i^a - x_i^b \in \mathcal{G}$

**Output:** The reduced Gröbner basis adjacent to $\mathcal{G}$ in which $\underline{x}_i^b - x_i^a$ is a facet binomial.

---

- Keine expliziten Termordnung nötig

# Breitensuche

**Algorithm 3** Enumerating the edge graph of the Gröbner fan via breath-first search

**Input:** Any reduced Gröbner basis $\mathcal{G}_0$ of $I_A$
**Output:** All reduced Gröbner bases of $I_A$, (all vertices of the edge graph)

- Einfacher und intuitiver Algorithmus
- Alle Gröbnerbasen müssen für den Vergleich gespeichert werden

# Tiefensuche(1)

- Gröbner-Fächer mit umgekehrter Tiefensuche nummerieren
- Ergebnis ist ein gerichter Subgraph $\longrightarrow$ Umgekehrter Suchbaum
- Nachteil von Breitensuche kompensiert

## Umgekehrter Suchbaum $T_>(I_A)$

- Azyklischer Graph mit einer eindeutigen Senke (bzgl. einer festen Termordnung)
- Einzigartige Pfade von Gröbnerbasis zur Senke

# Tiefensuche(2)

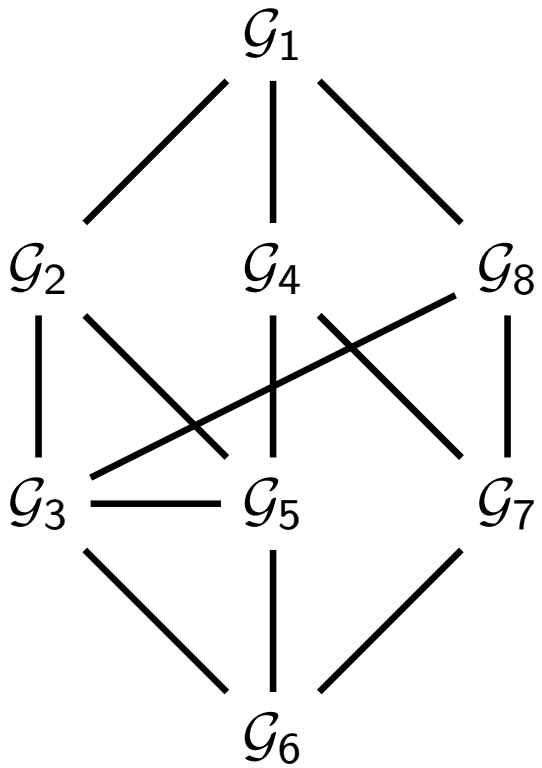
**Algorithm 4** Enumerating the edge graph of the Gröbner fan via reverse search

**Input:** Any reduced Gröbner basis $\mathcal{R}_>$ of $I_A$ and its term order $>$
**Output:** All reduced Gröbner bases of $I_A$ (all vertices of the edge graph)

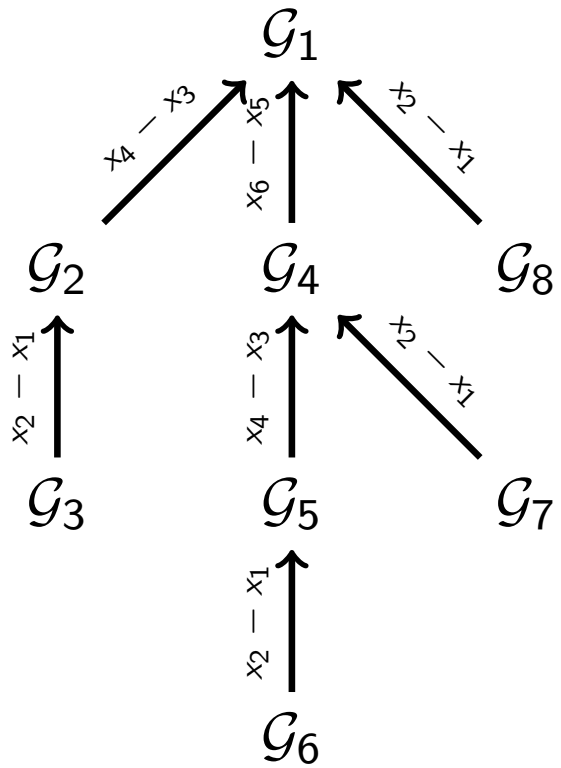
- Termordnung notwendig für Tiefensuche
- Binome die Termordung einhalten werden "umgedreht"

# Vergleich: Breitensuche & umgekehrte Tiefensuche

Sei $\mathcal{G} = \{x_1 - x_2, x_3 - x_4, x_5 - x_6, x_2^2 - 1, x_4^2 - 1, x_6^2 - 1\}$ mit lexikographischer Ordnung $>$ : $x_1 > \ldots > x_6$




(a) Ergebnis der Breitensuche




(b) Umgekehrter Suchbaum

# Gradkompatible Gröbnerbasis (1)

## Gradkompatible Gröbnerbasis

Eine reduzierte Gröbnerbasis (bzgl. zur Termordnung $>$) für ein Ideal ist gradkompatibel wenn der Vektor $\mathbf{1}$ im Gröbner-Kegel liegt.

- Leitterm muss höchsten Grad haben
- Jedes Ideal hat mindestens eine gradkompatible Gröbnerbasis

# Gradkompatible Gröbnerbasis (2)

## Einzige gradkompatible Gröbnerbasis

Eine reduzierte Gröbnerbasis (bzgl. einer Termordnung $>$) ist die einzige gradkompatible Gröbnerbasis wenn

$$deg(x^a) > deg(x^b) \; \forall \; x^a - x^b \in \mathcal{G}$$

- Breitensuche und umgekehrte Tiefensuche adaptierbar auf gradkompatible Gröbnerbasen

# Lineare Codes

## Linearer Code

Ein linearer Code der Länge n und Dimension k über $\mathbb{F}$ ist das Bild $\mathcal{C}$ einer linearen Abbildung $\phi : \mathbb{F}^k \to \mathbb{F}^n$.

- Als $[n, k]$ Code bezeichnet
- Alternativ als Generatormatrix $G \in \mathbb{F}^{k \times n}$ beschrieben
- Standardform : $G = (I_k | M)$
- Codewort $c$ vom Wort $x$ erhält man durch

$$xG = c$$

# Beispiel lineare Codes

- Sei $x = (1,0,1,0)$ und $G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$

$$(1,0,1,0) \cdot \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} = (1,0,1,0,0,0,1)$$

# Code Ideal

## Code Ideal

Sei $\mathcal{C}$ ein $[n, k]$ Code. Das Code Ideal $I(\mathcal{C})$ die Vereinigung zwischen

$$I_{\mathcal{C}} = \langle \mathbf{x}^c - \mathbf{x}^{c'} | c - c' \in \mathcal{C} \rangle + I_p,$$
$$\text{so dass } I_p = \langle x_i^p - 1 | 1 \leq i \leq n \rangle.$$

# Beispiel Code Ideal

- Sei $\mathcal{C}_1$ ein binärer $[6,3]$ code mit der Generatormatrix

$$G_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

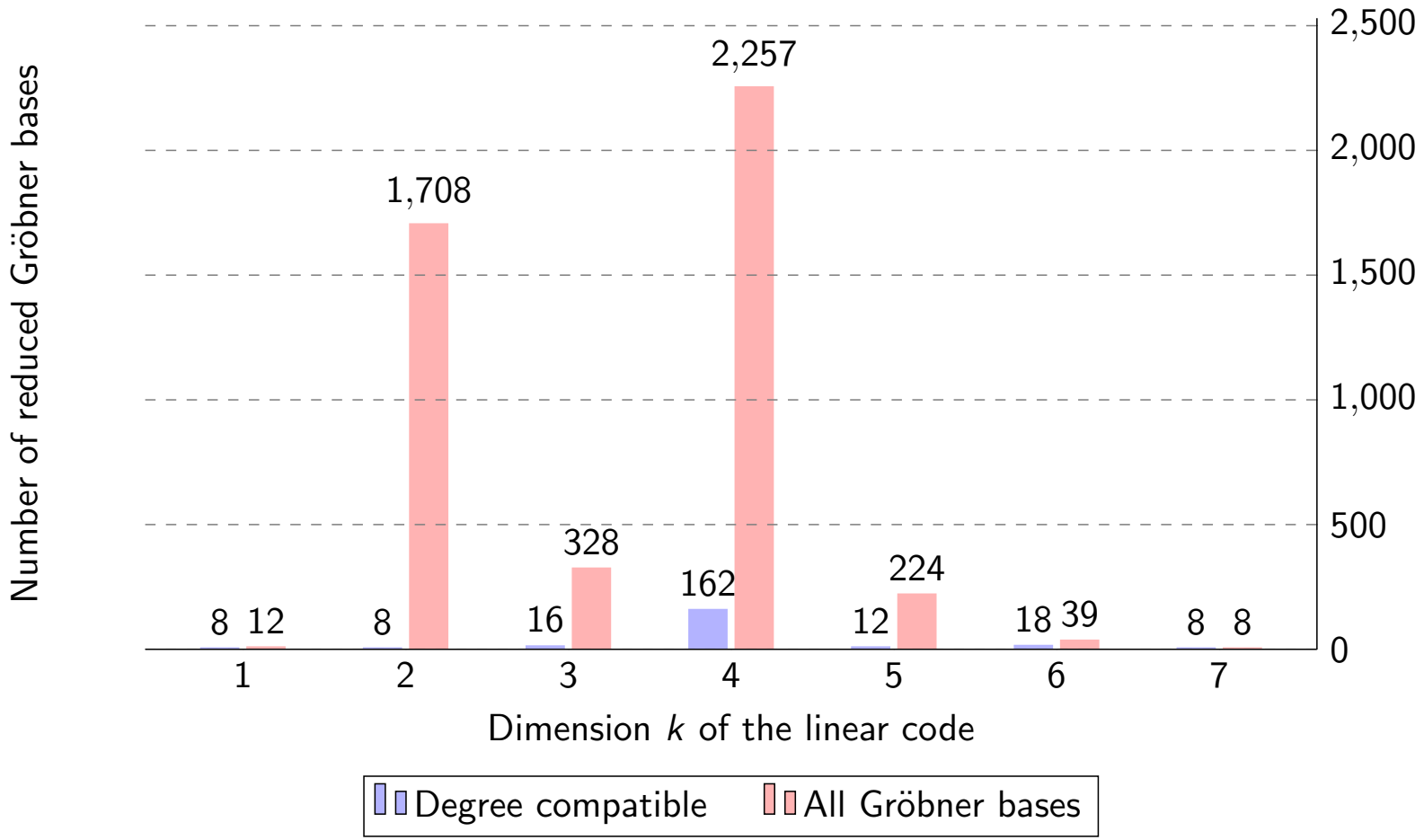
- Code Ideal $I(\mathcal{C})$ ergibt sich zu:

$$\begin{aligned} I(\mathcal{C}) = &\{x_1 - x_5,\ x_2 - x_4x_5x_6,\ x_3 - x_5\} \cup \\ &\{x_1^2 - 1,\ x_2^2 - 1,\ x_3^2 - 1,\ x_4^2 - 1,\ x_5^2 - 1,\ x_6^2 - 1\} \end{aligned}$$

- Reduzierte Gröbnerbasis mit lexikographischer Ordnung $>$ :

$$\mathcal{G}_> = \{x_1 - x_5,\ x_2 - x_4x_5x_6,\ x_3 - x_5\} \cup \{x_4^2 - 1,\ x_5^2 - 1,\ x_6^2 - 1\}$$

# Vergleich zw. gradkompatible & kompletter Gröbner-Fächer



# Zeitvergleich zwischen Gfan & Software

Table: Berechnungszeit in Sekunden

| $[n, k]$ Code | CIDGEL d.c. | CIDGEL | Gfan |
|---|---|---|---|
| $[8, 2]$ | 0.206 | 10.198 | 38.127 |
| $[8, 4]$ | 7.743 | 25.86 | 47.748 |
| $[9, 4]$ | 9.27 | 727.91 | 982.56 |
| $[9, 5]$ | 4.72 | 18.89 | 59.65 |
| $[10, 6]$ | 87.92 | 277.81 | 380.04 |

Vorführung der Software

# Mögliche Verbesserungen

1. Verwenden von externen LP-Solver
2. Parallelität

# Fazit

1. Keine (bekannte) Software berechnet gradkompatiblen Gröbnerfächer
2. Schnellere Berechnung als andere Software

Vielen Dank für eure Aufmerksamkeit!