Západočeská univerzita v Plzni
FAKULTA PEDAGOGICKÁ
KATEDRA TECHNICKÉ VÝCHOVY, FYZIKY A MATEMATIKY
ODDĚLENÍ MATEMATIKY

GRAM-SCHMIDTŮV ORTOGONALIZAČNÍ PROCES A LLL ALGORITMUS

DIPLOMOVÁ PRÁCE

Eva Mašková
Učitelství pro 2. stupeň ZŠ, obor Ma-Bi

Vedoucí práce: *Doc. RNDr. Jaroslav Hora, CSc.*

Plzeň, 1. leden 2012

Prohlašuji, že jsem diplomovou práci vypracovala samostatně s použitím uvedené literatury a zdrojů informací.

Plzeň, 1. leden 2012

……
vlastnoruční podpis

OBSAH

ÚVOD

Objevení LLL algoritmu pro matematiky znamenalo možnost objevení nových vět a vzorců za pomoci počítače, o jejichž existenci neměli doposud žádné povědomí. Další přínos je ve faktorizaci (rozkladu) polynomů, kde by mohl pomoci právě LLL algoritmus. V neposlední řadě je jeho výhodnost patrná také z hlediska výpočetní složitosti.

Hlavním cílem práce je čtenáři představit doposud, v České republice, nedostatečně zpracované téma LLL algoritmu. Jedná se především o záměr, uvést toto poměrně nové téma v českém prostředí, ilustrovat příklady výpočtů a demonstrovat jeho variabilní přínos a využití v matematické oblasti.

LLL algoritmus publikovali v roce 1982 A. K. Lenstra, H. W. Lenstra a L. Lovász jako algoritmus pro faktorizaci polynomů s racionálními koeficienty v článku *Factoring polynomials with rational coefficients*, který spolu s *Počítačovou algebrou* od Stanovského a *Lineární algebrou* od Drábka tvoří hlavní zdroje, ze kterých budu v této práci čerpat. Při zpracování práce budou rovněž využity počítačové programy Geogebra, ke grafickým ilustracím, a Mathematica, která je určená k výpočtu LLL algoritmu. Do práce bude vloženo i dostatečné množství odkazů, aby si mohl čtenář dohledat další potřebné informace, které není tato práce schopna obsáhnout.

Toto téma jsem si vybrala proto, že je poměrně nové a ne mnohokrát zpracované.

Vzhledem k tomu, že pro LLL algoritmus je nutné znát Gram-Schmidtův ortogonalizační proces, osvojit si potřebné informace o mřížkách a o hledání krátké báze dané mřížky, je nutné v práci uvést kapitoly, které se výše zmíněnými oblastmi budou zabývat, a po jejichž přečtení by měl mít čtenář dobrý základ pro pochopení LLL algoritmu.

Základ pro algoritmus tvoří zobecnění Gram-Schmidtova ortogonalizačního procesu, který lze provádět v prostorech se skalárním součinem. Teorii těchto prostorů, ortogonální a ortonormální báze bych chtěla připomenout v části 1. kapitoly, přičemž vyberu především základní věty a definice, které jsou důležité pro porozumění LLL algoritmu. Pominu zde některé důkazy a i některé souvislosti mezi definovanými pojmy a to především proto, že k tématu existuje dostatek kvalitní literatury (např. *Lineárna algebra a geometria* od Zlatoše, *Lineární algebra* od Bicana, atd.). Na závěr této kapitoly bych ráda uvedla, co je to Gram-Schmidtův ortogonalizační proces, jeho postup a samozřejmě

příklady na Gram-Schmidtovu ortogonalizaci a nápovědu, jakým způsobem se dá ověřit správnost výsledků těchto příkladů.

V druhé části bych chtěla představit mřížky, jejich základní vlastnosti a ukázat hledání krátké báze dané mřížky v dimenzi 2, kde je dokonce možné najít nejkratší bázi. V této práci se omezím pouze na mřížky nad vektorovým prostorem $\mathbb{R}^n$ a to především proto, že je tato definice jednodušší, není tedy nutné definovat další pojmy a z hlediska aplikace tento rozdíl není natolik podstatný.

Třetí kapitola se bude týkat právě LLL algoritmu, který dokáže najít v polynomiálním čase vcelku krátkou bázi dané mřížky. Zde si popíšeme LLL redukovanou bázi mřížky, dále uvedeme samotný LLL algoritmus a na závěr si ukážeme, jak tento algoritmus funguje na konkrétních příkladech.

Čtvrtá část bude o aplikaci LLL algoritmu, kde bych chtěla ukázat, že i tento algoritmus má mnoho využití nejen pro odborníky, jak v práci stručně uvedu, ale například i pro učitele v praxi. Ty by mohlo zaujmout, jak při troše štěstí lze za pomoci tohoto algoritmu najít ze zaokrouhleného reálného kořene určité polynomiální rovnice n – tého stupně právě tento polynom za předpokladu, že se jim zadání rovnice ztratilo.

Ke každé kapitole se pokusím uvést příklady, aby čtenář viděl, jak se daný postup provádí krok po kroku a zároveň bych chtěla ukázat, jak si lze ušetřit práci využitím matematických programů, zde konkrétně programu Mathematica 8.

Motivací ke zpracování tohoto tématu byl předmět Lineární algebra, kde jsme probírali Gram-Schmidtův ortogonalizační proces, který tvoří základ pro Lenstra-Lenstra-Lovászův algoritmus, kterým se ve své práci se zabývám. Tento algoritmus má velmi mnoho využití, které v práci stručně uvedu, ale vzhledem k mému oboru bych ráda přiblížila především jedno z nich a to hledání ztraceného polynomu ze zaokrouhleného čísla. Rozebírání dalších aplikací tohoto algoritmu již přesahuje rámec této práce.

1 GRAM – SCHMIDTŮV ORTOGONALIZAČNÍ PROCES

V této kapitole si řekneme co je skalární součin, co je vektorový prostor se skalárním součinem a jeho modely. Dále si budeme definovat ortogonální a ortonormální báze a Gram-Schmidtův ortogonalizační proces, ke kterému uvedeme několik příkladů.

1.1 VEKTOROVÉ PROSTORY SE SKALÁRNÍM SOUČINEM

Nejprve si ukážeme co je to skalární součin, několik jeho základních vlastností a modely vektorového prostoru se skalárním součinem.

Vektory z vektorového prostoru $\mathbb{R}^n$, budeme psát sloupcově. Matici, která má sloupce $b_1, b_2, \dots, b_k \in \mathbb{R}^n$ budeme zapisovat jako $(b_1|b_2|\dots|b_k)$. Lineární obal vektorů $b_1, b_2, \dots, b_k$ budeme značit $\langle b_1, b_2, \dots, b_k \rangle$.

Máme-li báze $B = (b_1, b_2, \dots, b_n)$ a $C = (c_1, c_2, \dots, c_n)$ prostoru $\mathbb{R}^n$, pak jednoznačně určenou maticí X rozumíme matici přechodu od báze B k bázi C, pro kterou platí $\bar{C} = \bar{B}X$, kde $\bar{B} = (b_1|b_2|\dots|b_n)$ a $\bar{C} = (c_1|c_2|\dots|c_n)$. Tedy i-tý sloupec matice X je roven vyjádření vektoru c_i v bázi B. Jen dodám, že matice přechodu od báze C k bázi B je rovna X^{-1}.

Skalární součin vektorů $\vec{x}$ a $\vec{y}$, kde $\vec{x} = (x_{1,}x_{2,\dots,}x_n)^T$ a $\vec{y} = (y_{1,}y_{2,\dots,}y_{n)}{}^T$ lze vyjádřit jako

$$\vec{x}.\vec{y} = \vec{x}^T\vec{y} = x_1y_1 + x_2y_2 + \cdots + x_ny_n.$$

^{T} v druhém výrazu znamená transponování, součin chápeme maticově, tedy jako matic typů $1 \times n$ a $n \times 1$.

Definice 1.1.1. Nechť V je vektorový prostor nad tělesem T, pak skalárním součinem na prostoru V nazveme každé zobrazení f množiny $V \times V$ do tělesa T, které má následující vlastnosti

1) $\forall x, y \in V \qquad f(x,y) = \overline{f(y,x)}$,
2) $\forall x, y, z \in V \quad f(x+y,z) = f(x,z) + f(y,z)$,
3) $\forall x, y \in V \qquad \forall a \in T \quad f(ax,y) = a.f(x,y)$,
4) $\forall x \in V, \quad x \neq 0 \qquad f(x,x) > 0$.

Nyní odvodíme základní vlastnosti skalárního součinu v prostoru A^2.

1) Vzhledem k tomu, že platí $\vec{x}.\vec{x} = x_1^2 + x_2^2$, usoudíme, že platí $\vec{x}.\vec{x} \geq 0 \wedge \vec{x}.\vec{x} = 0 \Leftrightarrow \vec{x} = \vec{o}$.

2) Komutativnost skalárního součinu je evidentní, tedy $\vec{x}.\vec{y} = \vec{y}.\vec{x}$.

3) Nyní budeme zkoumat, čemu se rovná výraz $\vec{x}.(\vec{y} + \vec{z})$. Podle definice skalárního součinu v A^2 bude platit

$$\vec{x}.(\vec{y} + \vec{z}) = x_1(y_1 + z_1) + x_2(y_2 + z_2) = (x_1y_1 + x_2y_2) + (x_1z_1 + x_2z_2) = \vec{x}.\vec{y} + \vec{x}.\vec{z}.$$

Platí tedy $\vec{x}.(\vec{y} + \vec{z}) = \vec{x}.\vec{y} + \vec{x}.\vec{z}$, což znamená, že je skalární součin distributivní vůči sčítání vektorů.

4) Nakonec upravujeme výraz $(\lambda.\vec{x}).\vec{y}$.

Postupně platí

$$(\lambda.\vec{x}).\vec{y} = [\lambda.(x_1, x_2)].\left(y_1, y_2\right) = (\lambda.x_1, \lambda.x_2).\left(y_1, y_2\right) = (\lambda.x_1).y_1 + (\lambda.x_2).y_2 = \lambda.(x_1.y_1) + \lambda.(x_2.y_2) = \lambda.(x_1.y_1 + x_2\,.y_2) = \lambda.(\vec{x}.\vec{y}).$$

Tím jsme získali další základní vlastnost v aritmetickém vektorovém prostoru A^2 a to asociativnost vnějšího násobení a skalárního součinu

$$(\lambda.\vec{x}).\vec{y} = \lambda.(\vec{x}.\vec{y}).$$

Vlastnosti, které jsme si nyní odvodili, jsou vlastnostmi eukleidovského vektorového prostoru a tento prostor axiomaticky vymezují.

Definice 1.1.2. Vektorový prostor V nazveme vektorovým prostorem se skalárním součinem resp. eukleidovským vektorovým prostorem právě tehdy, když je v tomto prostoru definován skalární součin dvou vektorů $\vec{u}, \vec{v}$, který budeme značit $\vec{u}.\vec{v}$, přičemž platí že $\vec{u}.\vec{v} \in \mathbb{R}$ a splňuje tyto axiomy

$(S_1)(\forall \vec{u} \in V)\vec{u}.\vec{u} \geq 0 \wedge \vec{u}.\vec{u} = 0 \Leftrightarrow \vec{u} = \vec{o}$.

$(S_2)(\forall \vec{u}, \vec{v} \in V)\vec{u}.\vec{v} = \vec{v}.\vec{u}$. (Skalární součin je komutativní.)

$(S_3)(\forall \vec{u}, \vec{v}, \vec{w} \in V)\vec{u}.(\vec{v} + \vec{w}) = \vec{u}.\vec{v} + \vec{u}.\vec{w}$. (Skalární součin je distributivní vůči sčítání vektorů.)

$(S_4)(\forall \vec{u}, \vec{v} \in V)(\lambda \in R)(\lambda.\vec{u}).\vec{v} = \lambda.(\vec{u}.\vec{v})$.

Nyní si ukážeme modely eukleidovského vektorového prostoru.

1. Aritmetický vektorový prostor. Skalární součin v aritmetickém vektorovém prostoru A^n je definován

$$\vec{u}.\vec{v} = u_1v_1 + u_2v_2 + \cdots + u_nv_n,$$

kde $\vec{u} = (u_1, u_2, \dots, u_n), \vec{v} = (v_1, v_2, \dots, v_n)$.

2. Geometrický vektorový prostor. Skalární součin v geometrickém vektorovém prostoru G^2 (resp. G^3) je definován

$$\vec{u}.\vec{v} = |\vec{u}|.|\vec{v}|.\cos\varphi,$$

kde $|\vec{u}|, |\vec{v}|$ jsou velikosti geometrických vektorů $\vec{u}, \vec{v}$ a kde φ je jejich odchylka.

3. Funkční model. Skalární součin ve funkčním modelu $\Phi_{\langle 0,1\rangle}$ je definován

$$\vec{f}(x).\vec{g}(x) = \int_0^1 f(x).g(x).$$

Lze dokázat, že skalární součiny, které jsme právě definovali, splňují axiomy eukleidovského vektorového prostoru. Ale to již v této práci dokazovat nebudeme.

Nyní si ukážeme několik základních vlastností skalárního součinu.

Upravíme dvojím způsobem následující výraz

$$\vec{u}.(\vec{u} + \vec{o}).$$

S použitím S_3 platí

$$\vec{u}.(\vec{u} + \vec{o}) = \vec{u}.\vec{u} + \vec{u}.\vec{o}.$$

Dále platí

$$\vec{u}.(\vec{u} + \vec{o}) = \vec{u}.\vec{u}.$$

Z těchto dvou rovností dostaneme

$$\vec{u}.\vec{u} + \vec{u}.\vec{o} = \vec{u}.\vec{u},$$

ze které dostáváme

$$\vec{u}.\vec{u} + \vec{u}.\vec{o} = \vec{u}.\vec{u}$$

$$\Downarrow$$

$$\vec{u}.\vec{u} + \vec{u}.\vec{o} = \vec{u}.\vec{u} + 0$$

$$\Downarrow$$

$$\vec{u}.\vec{o} = 0.$$

Tím jsme dokázali následující větu.

Věta 1.1.1. $(\forall \vec{u} \in V)\vec{u}.\vec{o} = 0.$

Dále budeme počítat skalární součin $(\vec{u} + \vec{v}).(\vec{w} + \vec{z})$. Označíme-li si $\vec{u} + \vec{v}$ jako vektor $\vec{x}$, bude platit za použití S_3

$$\vec{x}.(\vec{w} + \vec{z}).\vec{w} = \vec{x}.\vec{w} + \vec{x}.\vec{z} = (\vec{u} + \vec{v}).\vec{w} + (\vec{u} + \vec{v}).\vec{z} = \vec{u}.\vec{w} + \vec{v}.\vec{w} + \vec{u}.\vec{z} + \vec{v}.\vec{z}.$$

Tím jsme dokázali další větu.

Věta 1.1.2. $(\forall \vec{u}, \vec{v}, \vec{w}, \vec{z})\ (\vec{u} + \vec{v}).(\vec{w} + \vec{z}) = \vec{u}.\vec{w} + \vec{v}.\vec{w} + \vec{u}.\vec{z} + \vec{v}.\vec{z}.$

A nyní budeme počítat skalární součin $(\lambda.\vec{u}).(\eta.\vec{v})$.

Postupnými úpravami s využitím axiomů S_2 a S_4 dojdeme k

$$(\lambda.\vec{u}).(\eta.\vec{v}) = \lambda.[\vec{u}.(\eta.\vec{v})] = \lambda.[(\eta.\vec{v}).\vec{u}] = \lambda.[\eta.(\vec{v}.\vec{u})] = \lambda.[\eta.(\vec{u}.\vec{v})] = (\lambda.\eta).(\vec{u}.\vec{v}).$$

Tím dostáváme tuto větu.

Věta 1.1.3. $(\forall \vec{u}, \vec{v} \in V)(\forall \lambda, \eta \in R)\ (\lambda.\vec{u}).(\eta.\vec{v}) = (\lambda.\eta).(\vec{u}.\vec{v}).$

Jsou dány dva vektory $\vec{u}, \vec{v}$, přičemž vektor $\vec{v}$ je nenulový a λ, což je libovolné reálné číslo. Budeme počítat skalární součin

$$(\vec{u} - \lambda.\vec{v}).(\vec{u} - \lambda.\vec{v}),$$

který bude podle axiomu S_1 nezáporný, takže musí platit

$$(\vec{u} - \lambda.\vec{v}).(\vec{u} - \lambda.\vec{v}) \geq 0.$$

Podle vět 1.1.2. a 1.1.3. platí, že

$$\vec{u}.\vec{u} - 2\lambda.(\vec{u}.\vec{v}) + \lambda^2.(\vec{v}.\vec{v}) \geq 0.$$

Vzhledem k tomu, že je tato nerovnost platná pro libovolné reálné číslo λ, musí platit i pro

$$\lambda = \frac{\vec{u}.\vec{v}}{\vec{v}.\vec{v}}.$$

Dosazením λ z této rovnice do předchozí nerovnice dostaneme po snadné úpravě

$$\vec{u}.\vec{u} - 2.\frac{(\vec{u}.\vec{v})^2}{\vec{v}.\vec{v}} + \frac{(\vec{u}.\vec{v})^2}{\vec{v}.\vec{v}} \geq 0.$$

Pokud nerovnici vynásobíme skalárním součinem $\vec{v}.\vec{v}$, který bude jistě kladné reálné číslo, dostaneme nerovnost

$$(\vec{u}.\vec{u}).(\vec{v}.\vec{v}) - (\vec{u}.\vec{v})^2 \geq 0.$$

Po drobné úpravě dostaneme nerovnost

$$(\vec{u}.\vec{u}).(\vec{v}.\vec{v}) \geq (\vec{u}.\vec{v})^2,$$

která je základní nerovností na eukleidovských vektorových prostorech. Nazývá se Cauchy-Bunjakovského nerovnost.

Věta 1.1.4. (Cauchy-Bunjakovského nerovnost.)

$$(\forall \vec{u}, \vec{v} \in V)(\vec{u}.\vec{u}).(\vec{v}.\vec{v}) \geq (\vec{u}.\vec{v})^2.$$

Tato věta platí i pro nulový vektor $\vec{v}$.

Dále uvedeme definici velikosti vektoru.

Definice1.1.3. Velikostí vektoru $\vec{u}$ v eukleidovském vektorovém prostoru, kterou značíme $|\vec{u}|$, rozumíme druhou odmocninu ze skalárního součinu vektoru se sebou samým. Tedy

$$|\vec{u}| = \sqrt{\vec{u}.\vec{u}}.$$

K této definici si uvedeme několik poznámek.

1. Definice má smysl, protože podle axiomu S_1 je pod odmocninou nezáporné číslo.
2. $|\vec{u}| = 0 \Leftrightarrow \vec{u} = \vec{o}$.
3. $\vec{u} \neq \vec{o} \Rightarrow |\vec{u}| > 0$.
4. Vektor $\vec{u}$ nazveme jednotkovým vektorem právě tehdy, když $|\vec{u}| = 1$.
5. Z vektoru $\vec{u}$ dostaneme jednotkový vektor tak, že ho z vnějšku vynásobíme převrácenou hodnotou jeho velikosti. Takže

$\frac{1}{|\vec{u}|}.\vec{u}$ představuje jednotkový vektor.

Tato operace se nazývá normování vektoru.

Dalším pojmem je odchylka dvou nenulových vektorů.

Definice 1.1.4. Odchylkou φ dvou nenulových vektorů $\vec{u}, \vec{v}$ rozumíme úhel φ v

intervalu $\langle 0, \pi \rangle$, pro který platí

$$\cos \varphi = \frac{\vec{u}.\vec{v}}{|\vec{u}|.|\vec{v}|}.$$

Opět si uvedeme několik poznámek.

1. Víme, že tento vztah je smysluplný, protože z nerovnosti

$$(\vec{u}.\vec{u}).(\vec{v}.\vec{v}) \geq (\vec{u}.\vec{v})^2$$

vyplývá nerovnost

$$\frac{(\vec{u}.\vec{v})^2}{(\vec{u}.\vec{u}).(\vec{v}.\vec{v})} \leq 1.$$

Tuto nerovnost přepíšeme pomocí definice velikosti vektoru a dostaneme

$$\frac{(\vec{u}.\vec{v})^2}{|\vec{u}|^2.|\vec{v}|^2} \leq 1.$$

Tato nerovnost již vede k nerovnostem

$$-1 \leq \frac{\vec{u}.\vec{v}}{|\vec{u}||\vec{v}|} \leq 1,$$

které konečně zaručují smysluplnost úvodního vzorce.

2. Odchylka φ bude rovna $\frac{\pi}{2}$ právě tehdy, když je skalární součin $\vec{u}.\vec{v}$ roven nule. Takovéto vektory nazveme kolmé neboli ortogonální. Definici uvedeme v následující kapitole.

1.2 ORTOGONÁLNÍ A ORTONORMÁLNÍ BÁZE

Definice 1.2.1. Dva nenulové vektory nazveme ortogonální (kolmé) právě tehdy, když se jejich skalární součin rovná nule. Zapíšeme

$$\vec{u} \perp \vec{v} \Leftrightarrow \vec{u}.\vec{v} = 0.$$

Předpokládáme, že je dána skupina vektorů $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_n$, které jsou vzájemně ortogonální, takže platí

$$(\forall i, j) i \neq j \Rightarrow \vec{u}_i \perp \vec{u}_j.$$

Nyní utvoříme lineární kombinaci těchto vektorů a položíme ji rovnu nulovému vektoru.

$$\lambda_1\vec{v}_1 + \lambda_2\vec{v}_2 + \cdots + \lambda_i\vec{v}_i + \cdots + \lambda_n\vec{v}_n = \vec{o}.$$

Tuto vektorovou rovnost skalárně vynásobíme vektorem $\vec{v}_i$. Pokud použijeme axiomy S_2 a S_4, dostaneme rovnost

$$\lambda_1(\vec{v}_1.\vec{v}_i) + \lambda_2(\vec{v}_2.\vec{v}_i) + \cdots + \lambda_i(\vec{v}_i.\vec{v}_i) + \cdots + \lambda_n(\vec{v}_n.\vec{v}_i) = \vec{o}.\vec{v}_i.$$

Skalární součiny $\vec{v}_j.\vec{v}_i$, kde $i \neq j$ se rovnají nule, skalární součin $\vec{v}_i.\vec{v}_i$ je podle axiomu S_1 kladné reálné číslo ρ, skalární součin $\vec{o}.\vec{v}_i$ je roven nule (podle věty 1.1.1). Tímto dostáváme z výše uvedené rovnosti rovnost

$$\lambda_i.\rho = 0.$$

Vzhledem k tomu, že $\rho > 0$ bude $\lambda_i = 0$ a tím jsme dokázali, že

$$(\forall i)\lambda_i = 0.$$

Skupinu vektorů $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_n$ tvoří lineárně nezávislé vektory.

Věta 1.2.1. Pokud skupina vektorů představuje skupinu vzájemně ortogonálních vektorů (každé dva různé vektory patřící do této skupiny jsou na sebe kolmé), potom vektory patřící do této skupiny jsou lineárně nezávislé a tvoří ortogonální bázi. Toto můžeme říci ještě jinak: ortogonalita je příčinou nezávislosti.

Nyní si budeme definovat bázi ortonormální.

Věta 1.2.2. Bázi $[\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_n]_B^{V^n}$ nazýváme ortonormální bází vektorového prostoru V^n právě tehdy, když splňuje tyto podmínky:

1. Báze musí být ortogonální.

2. Vektory $\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_n$ jsou jednotkové vektory.

Ortonormalitu báze $[\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_n]_B^{V^n}$ můžeme zapsat za pomoci skalárních součinů:

1. Jestliže $i \neq j$, pak $\vec{e}_i.\vec{e}_j = 0$.

2. Jestliže $i = j$, pak $\vec{e}_i.\vec{e}_j = 1$.

Lehce shledáme, že vektory

$\vec{e}_1 = (1, 0, \ldots, 0), \vec{e}_2 = (0, 1, \ldots, 0), \vec{e}_n = (0, 0, \ldots, 1)$ tvoří ortonormální bázi aritmetického vektorového prostoru.

1.3 GRAM-SCHMIDTŮV ORTOGONALIZAČNÍ PROCES

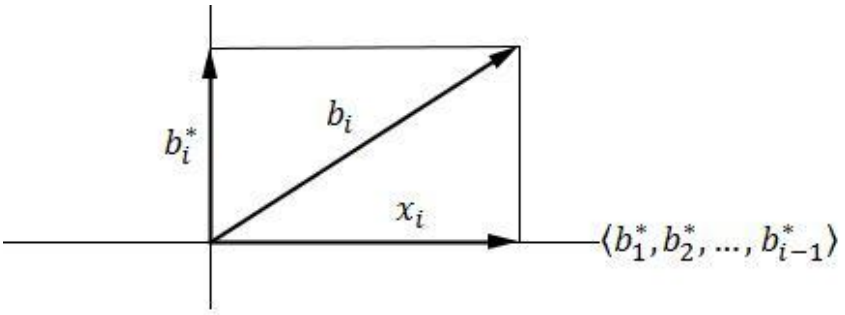
Tento proces dostal název podle Jorgena Pedersena Grama a Erharda Schmidta.[1]

Gram-Schmidtův ortogonalizační proces je proces, který pro danou bázi $b_1, b_2, \dots, b_n$ prostoru $\mathbb{R}^n$ najde bázi $b_1^*, b_2^*, \dots, b_n^*$ takovou, že

(1) $b_i^* . b_j^* = 0$ pro všechna $1 \leq i \leq j \leq n$,

(2) $b_i^* = b_i - x_i$, kde $x_i \in \langle b_1, \dots, b_{i-1} \rangle$, pro všechna $1 \leq i \leq n$.

V (1) se uvádí, že vektory $b_1^*, b_2^*, \dots, b_n^*$ jsou na sebe kolmé. Vektor x_i, který je uveden v (2) je ortogonální projekcí vektoru b_i na podprostor $\langle b_1, \dots, b_{i-1} \rangle$ a tudíž i nejlepší aproximace vektoru x_i v tomto prostoru. Vektor b_i^* je kolmicí na tento podprostor. Z (2) vyplývá, že $\langle b_1, \dots, b_i \rangle = \langle b_1^*, b_2^*, \dots, b_i^* \rangle$ pro všechna i.



Obrázek 1

[1] uvádí, že si lze Gram-Schmidtův ortogonalizační proces představit jako postupné narovnávání vstupních vektorů do kolmé plochy a hýbání i-tým vektorem v podprostoru, který je určen prvními i vektory, a to v tom směru, aby se neměnil objem rovnoběžnostěnu, který je určen danými vektory.

Protože vektory b_i^* a x_i jsou na sebe kolmé, platí, že $\|b_i\|^2 = \|b_i^*\|^2 + \|x_i\|^2$. Speciálně

$$\|b_i\| \geq \|b_i^*\|.$$

Vektor x_i leží v podprostoru $\langle b_1^*, b_2^*, \dots, b_{i-1}^* \rangle$, tedy

(3) $$x_i = \sum_{j=1}^{i-1} \mu_{ij} \, b_j^*$$

pro určitá reálná čísla μ_{ij}, která získáme z (1), (2) tak, že $b_i^* . b_j^* = (b_i - x_i) . b_j^* = 0$ a z (3) dosazením vyjádření vektoru x_i dostaneme

[1] Měli bychom však dodat, že Laplace představil tento proces dříve než Gram a Schmidt, viz [10].

$$\mu_{ij} = \frac{b_i b_j^*}{b_j^* b_j^*} = \frac{b_i b_j^*}{\left\| b_j^* \right\|^2}.$$

Vektory $b_1^*, b_2^*, \dots, b_n^*$ jsou tedy určeny vektory $b_1, \dots, b_n$ tímto způsobem:

$b_1^* = b_1$,

$b_2^* = b_2 - \mu_{21} b_1^*$, kde $\mu_{21} = \frac{b_2 b_1^*}{\left\| b_1^* \right\|^2}$,

...

$b_i^* = b_i - \sum_{j=1}^{i-1} \mu_{ij} b_j^*$, kde $\mu_{ij} = \frac{b_i b_j^*}{\left\| b_j^* \right\|^2}$,

...

Tím, že vyjádříme vektory $b_1, b_2, \dots, b_n$, dostaneme maticový zápis

$$(b_1 | b_2 | \dots | b_n) = (b_1^* | b_2^* | \dots | b_n^*) \begin{pmatrix} 1 & \mu_{21} & \mu_{31} & \cdots & \mu_{n1} \\ 0 & 1 & \mu_{32} & \cdots & \mu_{n2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Nyní si ukážeme další možný způsob, jak lze vypočíst ortogonální projekci x_i vektoru b_i na podprostor $\langle b_1, b_2, \dots, b_{i-1} \rangle$. Hledáme vektor $x_i = x_1 b_1 + x_2 b_2 + \cdots + x_{i-1} b_{i-1}$ takový, že vektor $b_i - x_i$ je kolmý na všechny vektory $b_1, b_2, \dots, b_{i-1}$. Výsledkem je soustava rovnic

$$G_{b_1, b_2, \dots, b_{i-1}} . (x_1, x_2, \dots, x_{i-1})^T = (b_i . b_1, b_i . b_2, \dots, b_i . b_{i-1})^T,$$

kde $G_{b_1, b_2, \dots, b_{i-1}}$ je tzv. Gramova matice[2] k vektorům $b_1, b_2, \dots, b_{i-1}$, která je daná předpisem

$$G_{b_1, b_2, \dots, b_{i-1}} = (b_k . b_l)_{k,l=1}^{i-1} = (b_1 | b_2 | \dots | b_{i-1})^T (b_1 | b_2 | \dots | b_{i-1}).$$

Tyto dva způsoby výpočtu se liší v tom, zda vektor x_i vyjadřujeme v bázi $b_1^*, b_2^*, \dots, b_{i-1}^*$ nebo v bázi $b_1, b_2, \dots, b_{i-1}$. První postup je výhodnější v tom, že vzniklá soustava rovnic má diagonální matici, a tudíž lze její řešení (což jsou koeficienty μ_{ij}) vyjádřit přímo.

Tvrzení 1.3.1 Pro libovolné $1 \leq k \leq n$ platí

[2] Gramova matice – více viz [4] str. 388

$$\det\left(G_{b_1,b_2,\dots,b_k}\right) = \|b_1^*\|^2 . \|b_2^*\|^2 \dots \|b_k^*\|^2 \leq \|b_1\|^2 . \|b_2\|^2 \dots \|b_k\|^2 .$$ [3]

Důkaz (podle [1]). Nerovnost v tvrzení vyplývá z nerovnosti $\|b_i^*\| \leq \|b_i\|$.

Označíme si matice

$$A = (b_1|b_2| \dots |b_k|b_{k+1}^*|b_{k+2}^*|\dots|b_n^*) \text{ a } B = (b_1^*|b_2^*|\dots|b_n^*).$$

Z maticového zápisu Gram-Schmidtovy ortogonalizace vyplývá, že $A = BX$ pro určitou horní trojúhelníkovou matici X, která má na diagonále samé jedničky. Pro $k = n$ je to přímo vztah, který jsme odvodili výše, pro $k < n$ je nutné nahradit posledních $n - k$ sloupců vektory kanonické báze. Podle věty o determinantu součinu matic je $det(A) = det(B)\, det(X) = det\,(B)$, takže tím, že využijeme $det\,(A^T) = det\,(A)$ a $det(B^T) = det\,(B)$ dostaneme

$$det(A^T A) = det(A)^2 = det(B)^2 = det(B^T B).$$

Matice $B^T B$ je diagonální (vzhledem k tomu, že vektory $b_1^*, b_2^*, \dots, b_n^*$ jsou na sebe kolmé) a na diagonále jsou prvky $\|b_1^*\|^2, \|b_2^*\|^2, \dots, \|b_n^*\|^2$ a tudíž

$$det(A^T A) = \|b_1^*\|^2 . \|b_2^*\|^2 \dots \|b_n^*\|^2.$$

Protože pro všechna $1 \leq i < j \leq n$ platí, že je vektor b_j^* kolmý na vektor b_i, je matice $A^T A$ blokově diagonální. Prvním blokem je matice $G_{b_1,b_2,\dots,b_k}$, zbylé bloky mají velikost 1 a obsahují čísla $\|b_{k+1}^*\|^2, \|b_{k+2}^*\|^2, \dots, \|b_n^*\|^2$. Z toho vyplývá, že

$$det(A^T A) = \det\left(G_{b_1,b_2,\dots,b_k}\right) \|b_{k+1}^*\|^2 \, \|b_{k+2}^*\|^2 \dots \, \|b_n^*\|^2.$$

Pokud porovnáme oba odvozené vztahy, získáme požadovanou rovnost.

Vzhledem k tomu, že absolutní hodnota determinantu vlastně udává n-rozměrný objem rovnoběžnostěnu, který je určen sloupcovými (řádkovými) vektory dané matice, tak je z tohoto důkazu patrný geometrický význam determinantu Gramovy matice, tedy, že udává druhou mocninu *k*-rozměrného objemu rovnoběžnostěnu určeného vektory $b_1, b_2, \dots, b_k$.

1.4 PŘÍKLADY VÝPOČTŮ

Na příkladech si ukážeme několik různých způsobů, jak lze provést Gram-Schmidtův ortogonalizační proces.

[3] Této nerovnosti se říká Hadamardova nerovnost a její platnost je zřejmá i bez důkazu: objem rovnoběžnostěnu je jistě menší nebo roven objemu kvádru se stejně dlouhými hranami.

Příklad 1.4.1. Nalezněte ortogonální bázi modulu M, který je částí aritmetického vektorového prostoru A^4, jestliže báze tohoto modulu je dána vektory $\vec{b}_1 = (1,1,1,1), \vec{b}_2 = (1,1,1,-1), \vec{b}_3 = (-1,1,1,1)$.

Snadno se přesvědčíme, že vektory jsou lineárně nezávislé (stačí vytvořit matici, která má tyto vektory za své řádkové vektory a převést ji na trojúhelníkový tvar - viz příklad 1.4.5.).

1. Nejprve položíme $\vec{b}_1^* = \vec{b}_1$.
2. Další vektor ortogonální báze $\vec{b}_2^*$ se pokusíme vyjádřit jako lineární kombinaci vektorů $\vec{b}_1^*, \vec{b}_2$. Vektorově to vyjádříme jako

$$\vec{b}_2^* = \lambda_1 \vec{b}_1^* + \lambda_2 \vec{b}_2 \text{ s podmínkou, že } \vec{b}_2^* \perp \vec{b}_1^*.$$

Tuto nerovnost skalárně vynásobíme vektorem $\vec{b}_1^*$ a dostaneme

$$\vec{b}_2^* . \vec{b}_1^* = \lambda_1 (\vec{b}_1^* . \vec{b}_1^*) + \lambda_2 (\vec{b}_2 . \vec{b}_1^*).$$

Vzhledem k tomu, že $\vec{b}_2^* \perp \vec{b}_1^*$, musí platit $\vec{b}_2^* . \vec{b}_1^* = 0$. Dále platí $\vec{b}_1^* . \vec{b}_1^* = (1,1,1,1).(1,1,1,1) = 4, \vec{b}_2 . \vec{b}_1^* = (1,1,1,-1).(1,1,1,1) = 2$.

Takže z výše uvedené rovnice dostaneme rovnici

$$0 = 4\lambda_1 + 2\lambda_2.$$

Jedním z možných řešení rovnice je

$$\lambda_1 = -2 \wedge \lambda_2 = 4,$$

a tedy

$$\vec{b}_2^* = (-2).(1,1,1,1) + 4.(1,1,1,-1) = (2,2,2,-6).$$

Můžeme provést kontrolu kolmosti a to tak, že spočteme skalární součin vektorů $\vec{b}_1^*, \vec{b}_2^*$, který je evidentně roven nule.

3. Poslední hledaný vektor $\vec{b}_3^*$ se pokusíme najít jako lineární kombinaci vektorů $\vec{b}_1^*,\ \vec{b}_2^*,\ \vec{b}_3$ a zapíšeme to takto

$$\vec{b}_3^* = \lambda_1 \vec{b}_1^* + \lambda_2 \vec{b}_2^* + \lambda_3 \vec{b}_3 \text{ s podmínkou, že } \vec{b}_3^* \perp \vec{b}_1^* \text{ a } \vec{b}_3^* \perp \vec{b}_2^*.$$

Tuto vektorovou rovnost postupně skalárně vynásobíme vektory $\vec{b}_1^*,\ \vec{b}_2^*$ a dostaneme

$$\vec{b}_3^*.\vec{b}_1^* = \lambda_1(\vec{b}_1^*.\vec{b}_1^*) + \lambda_2(\vec{b}_2^*.\vec{b}_1^*) + \lambda_3(\vec{b}_3.\vec{b}_1^*),$$

$$\vec{b}_3^*.\vec{b}_2^* = \lambda_1(\vec{b}_1^*.\vec{b}_2^*) + \lambda_2(\vec{b}_2^*.\vec{b}_2^*) + \lambda_3(\vec{b}_3.\vec{b}_2^*).$$

Nyní vyčíslíme z těchto rovnic příslušné skalární součiny. Nejprve díky výše uvedeným podmínkám je $\vec{b}_3^*.\vec{b}_1^* = 0 \wedge \vec{b}_3^*.\vec{b}_2^* = 0$ a již dříve jsme uvedli v 2. kroku, že $\vec{b}_2^*.\vec{b}_1^* = 0$. Dříve byl vyčíslen také skalární součin $\vec{b}_1^*.\vec{b}_1^* = 4$. Nyní provedeme zbývající skalární součiny.

$\vec{b}_2^*.\vec{b}_2^* = (2,2,2,-6).(2,2,2,-6) = 48,$ $\vec{b}_3.\vec{b}_1^* = (-1,1,1,1).(1,1,1,1) = 2,$

$\vec{b}_3.\vec{b}_2^* = (-1,1,1,1).(2,2,2,-6) = -4.$

Rovnice jsou tedy ve tvaru

$$4\lambda_1 + 2\lambda_3 = 0,$$

$$48\lambda_2 - 4\lambda_3 = 0.$$

Řešením rovnic je například $\lambda_3 = 12 \wedge \lambda_2 = 1 \wedge \lambda_1 = -6$. Tím tedy dostáváme vyjádření hledaného vektoru $\vec{b}_3^*$ a to

$$\vec{b}_3^* = -6.(1,1,1,1) + (2,2,2,-6) + 12.(-1,1,1,1) = (-16,8,8,0).$$

Opět si můžeme ověřit, že platí $\vec{b}_3^*.\vec{b}_1^* = 0 \wedge \vec{b}_3^*.\vec{b}_2^* = 0.$

Ortogonální báze je tedy ve tvaru

$$[(1,1,1,1),(2,2,2,-6),(-16,8,8,0)\ ].$$

Ještě bychom mohli provést ortonormalizaci této ortogonální báze.

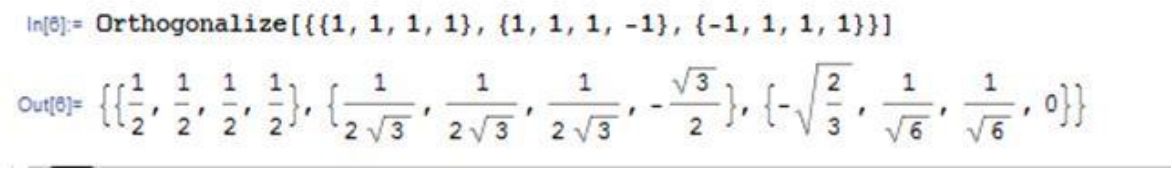
Postupně vyčíslíme velikosti vektorů báze a to

$$\|\vec{b}_1^*\| = \sqrt{4}, \|\vec{b}_2^*\| = \sqrt{48}, \|\vec{b}_3^*\| = \sqrt{384}.$$

Ortonormální báze je tedy ve tvaru

$$\left[\frac{1}{2}(1,1,1,1), \frac{1}{\sqrt{48}}\ (2,2,2,-6), \frac{1}{\sqrt{384}}(-16,8,8,0)\ \right].$$

V Mathematice vyjde



Po drobné úpravě zjistíme, že se oba výsledky rovnají.

Příklad 1.4.2. Nalezněte ortogonální bázi modulu M, který je částí aritmetického vektorového prostoru A^4, jestliže báze tohoto modulu je dána vektory

$$\vec{b}_1 = (1, 1, -1, -2), \vec{b}_2 = (5, 8, -2, -3), \vec{b}_3 = (3, 9, 3, 8).$$

Nejprve položíme $\vec{b}_1 = \vec{b}_1^*$, tedy

$$\vec{b}_1^* = (1, 1, -1, -2).$$

Dále vypočteme vektor $\vec{b}_2^*$ jako

$$\vec{b}_2^* = \vec{b}_2 + \lambda\vec{b}_1^* = (5, 8, -2, -3) + \lambda(1, 1, -1, -2).$$

Víme, že skalární součin $\vec{b}_1^*.\vec{b}_2^* = 0$, tedy

$$\vec{b}_1^*.\vec{b}_2^* = 21 + 7\,\lambda = 0.$$

Odtud dostáváme $\lambda = -3$ a

$$\vec{b}_2^* = (2, 5, 1, 3).$$

Nyní vyjádříme $\vec{b}_3^*$ jako

$$\vec{b}_3^* = \vec{b}_3 + \eta\vec{b}_1^* + \mu\vec{b}_2^* = (3, 9, 3, 8) + \eta(1, 1, -1, -2) + \mu(2, 5, 1, 3).$$

Jelikož má být vektor $\vec{b}_1^*$ kolmý na $\vec{b}_3^*$ a zároveň $\vec{b}_2^*$ má být kolmý na $\vec{b}_3^*$, musí být jejich skalární součin roven nule. Takže platí

$$\vec{b}_1^*.\vec{b}_3^* = -7 + 7\,\eta = 0 \Rightarrow \eta = 1,$$

$$\vec{b}_2^*.\vec{b}_3^* = 78 + 39\mu = 0 \Rightarrow \mu = -2.$$

Když dosadíme do výše uvedené rovnice pro vyjádření vektoru $\vec{b}_3^*$, dostaneme

$$\vec{b}_3^* = (2, 5, 1, 3).$$

Zjistili jsme, že $\vec{b}_2^* = \vec{b}_3^*$ a že jejich skalární součin není roven nule. To znamená, že původní báze byla lineárně závislá, což jsme mohli zjistit hned na začátku, pokud bychom

závislost ověřili a ušetřit si tím práci. Proto je důležité nezapomenout lineární nezávislost vektorů na začátku vždy ověřit.

Ortogonální báze je tedy ve tvaru

$$[(1,1,-1,-2),(2,5,1,3)].$$

Příklad 1.4.3. Nalezněte ortogonální bázi modulu M, který je částí aritmetického vektorového prostoru A^4, jestliže báze tohoto modulu je dána vektory $\vec{b}_1 = (1,2,2,-1), \vec{b}_2 = (1,1,-5,3), \vec{b}_3 = (3,2,8,-7)$.

Nejprve ověříme lineární nezávislost vektorů a poté položíme $\vec{b}_1 = \vec{b}_1^*$, tedy

$$\vec{b}_1^* = (1,2,2,-1).$$

Dále vypočteme vektor $\vec{b}_2^*$ jako

$$\vec{b}_2^* = \vec{b}_2 + \lambda\vec{b}_1^* = (1,1,-5,3) + \lambda(1,2,2,-1).$$

Víme, že skalární součin $\vec{b}_1^*.\vec{b}_2^*$ musí být roven nule, tedy

$$\vec{b}_1^*.\vec{b}_2^* = -10 + 10\,\lambda = 0.$$

Odtud dostáváme $\lambda = 1$ a

$$\vec{b}_2^* = (2,3,-3,2).$$

Nyní vyjádříme $\vec{b}_3^*$ jako

$$\vec{b}_3^* = \vec{b}_3 + \eta\vec{b}_1^* + \mu\vec{b}_2^* = (3,2,8,-7) + \eta(1,2,2,-1) + \mu(2,3,-3,2).$$

Vzhledem k tomu, že vektor $\vec{b}_1^*$ má být kolmý na $\vec{b}_3^*$ a stejně tak $\vec{b}_2^*$ má být kolmý na $\vec{b}_3^*$, musí být jejich skalární součin roven nule, takže

$$\vec{b}_1^*.\vec{b}_3^* = 30 + 10\,\eta = 0 \Rightarrow \eta = -3,$$

$$\vec{b}_2^*.\vec{b}_3^* = -26 + 26\mu = 0 \Rightarrow \mu = 1.$$

Když dosadíme do výše uvedené rovnice pro vyjádření vektoru $\vec{b}_3^*$, dostaneme

$$\vec{b}_3^* = (2,-1,-1,-2).$$

Ortogonální báze je tedy ve tvaru

$$[(1,2,2,-1),(2,3,-3,2),(2,-1,-1,-2)].$$

Ortogonalitu si opět můžeme ověřit tak, že skalární součiny vektorů budou rovny nule.

Příklad 1.4.4. Nalezněte ortogonální bázi modulu M, který je částí aritmetického vektorového prostoru A^3, jestliže báze tohoto modulu je dána vektory

$$\vec{b}_1 = (1,0,0), \vec{b}_2 = (-1,3,2), \vec{b}_3 = (2,-2,5).$$

Nejprve opět ověříme nezávislost vektorů a dále položíme $\vec{b}_1 = \vec{b}_1^*$. Dostaneme

$$\vec{b}_1^* = (1,0,0).$$

Dále vypočteme vektor $\vec{b}_2^*$ jako

$$\vec{b}_2^* = \vec{b}_2 - \mu_{21}\vec{b}_1^*.$$

Nyní musíme vypočítat μ_{21} jako

$$\mu_{21} = \frac{\vec{b}_2 . \vec{b}_1^*}{\left\|\vec{b}_1^*\right\|^2} = \frac{-1}{1} = -1$$

a teď již lze dosadit do vzorce pro $\vec{b}_2^*$, tudíž

$$\vec{b}_2^* = \vec{b}_2 - \mu_{21}\vec{b}_1^* = (-1,3,2) + (1,0,0) = (0,3,2).$$

Vektor $\vec{b}_3^*$ dostaneme dosazením do vzorce

$$\vec{b}_3^* = b_3 - \mu_{31}\vec{b}_1^* - \mu_{32}\vec{b}_2^*.$$

Nejprve musíme vypočíst μ_{31} a μ_{32}, takže

$$\mu_{31} = \frac{\vec{b}_3 . \vec{b}_1^*}{\left\|\vec{b}_1^*\right\|^2} = \frac{2}{1} = 2,$$

$$\mu_{32} = \frac{\vec{b}_3 . \vec{b}_2^*}{\left\|\vec{b}_2^*\right\|^2} = \frac{4}{13}.$$

Po dosazení do výše uvedeného vzorce dostaneme

$$\vec{b}_3^* = (2,-2,5) - 2(1,0,0) - \frac{4}{13}(0,3,2) = \left(0, -\frac{38}{13}, \frac{57}{13}\right).$$

Ortogonální báze je tedy ve tvaru

$$\left[(1,0,0),(0,3,2),\left(0,-\frac{38}{13},\frac{57}{13}\right)\right].$$

Ortonormální bázi lze nalézt také pomocí programu Mathematica 8 a příkazu Orthogonalize.[4]

```
In[18]:= Orthogonalize[{{1, 0, 0}, {-1, 3, 2}, {2, -2, 5}}]
Out[18]= {{1, 0, 0}, {0, 3/Sqrt[13], 2/Sqrt[13]}, {0, -2/Sqrt[13], 3/Sqrt[13]}}
```

Příklad 1.4.5. Nalezněte ortogonální bázi modulu M, který je částí aritmetického vektorového prostoru A^4, jestliže báze tohoto modulu je dána vektory

$$\vec{b}_1 = (2,1,3,-1), \vec{b}_2 = (7,4,3,-3), \vec{b}_3 = (1,1,-6,0), \vec{b}_4 = (5,7,7,8).$$

Opět nejprve ověřujeme lineární nezávislost, což si u tohoto příkladu podrobně ukážeme. Nejprve vytvoříme matici, která má zadané vektory za své řádkové vektory.

$$\begin{pmatrix} 2 & 1 & 3 & -1 \\ 7 & 4 & 3 & -3 \\ 1 & 1 & -6 & 0 \\ 5 & 7 & 7 & 8 \end{pmatrix}.$$

Nyní prohodíme vektory tak, aby to pro nás bylo co nevýhodnější a postupně matici převádíme na trojúhelníkový tvar pomocí ekvivalentních úprav, takže

$$\begin{pmatrix} 2 & 1 & 3 & -1 \\ 7 & 4 & 3 & -3 \\ 1 & 1 & -6 & 0 \\ 5 & 7 & 7 & 8 \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & -6 & 0 \\ 2 & 1 & 3 & -1 \\ 5 & 7 & 7 & 8 \\ 7 & 4 & 3 & -3 \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & -6 & 0 \\ 0 & -1 & 15 & -1 \\ 0 & 2 & 37 & 8 \\ 0 & -3 & 45 & -3 \end{pmatrix}.$$

Vidíme, že vektory ve 2. a 4. řádku jsou stejné, takže můžeme jeden vyškrtnout a zůstane nám

$$\begin{pmatrix} 1 & 1 & -6 & 0 \\ 0 & -1 & 15 & -1 \\ 0 & 2 & 37 & 8 \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & -6 & 0 \\ 0 & -1 & 15 & -1 \\ 0 & 0 & 67 & 6 \end{pmatrix}.$$

Tyto vektory jsou již lineárně nezávislé, tudíž nyní hledáme ortogonální bázi modulu M, k bázi, jež je dána vektory

$$\vec{b}_1 = (2,1,3,-1), \vec{b}_2 = (1,1,-6,0), \vec{b}_3 = (5,7,7,8).$$

[4] Více o této funkci viz. příloha

Nyní již položíme $\vec{b}_1 = \vec{b}_1^*$. Dostaneme

$$\vec{b}_1^* = (2, 1, 3, -1).$$

Dále vyjádříme vektor $\vec{b}_2^*$ jako

$$\vec{b}_2^* = \vec{b}_2 - \mu_{21}\vec{b}_1^*.$$

Nyní musíme vypočítat μ_{21} jako

$$\mu_{21} = \frac{\vec{b}_2 . \vec{b}_1^*}{\left\|\vec{b}_1^*\right\|^2} = \frac{-15}{15} = -1$$

a nyní můžeme dosadit do vzorce pro $\vec{b}_2^*$

$$\vec{b}_2^* = \vec{b}_2 - \mu_{21}\vec{b}_1^* = (1, 1, -6, 0) + (2, 1, 3, -1) = (3, 2, -3, -1).$$

Vektor $\vec{b}_3^*$ dostaneme dosazením do vzorce

$$\vec{b}_3^* = b_3 - \mu_{31}\vec{b}_1^* - \mu_{32}\vec{b}_2^*.$$

Nejprve musíme vypočíst μ_{31} a μ_{32}, takže

$$\mu_{31} = \frac{\vec{b}_3 . \vec{b}_1^*}{\left\|\vec{b}_1^*\right\|^2} = \frac{30}{15} = 2,$$

$$\mu_{32} = \frac{\vec{b}_3 . \vec{b}_2^*}{\left\|\vec{b}_2^*\right\|^2} = 0.$$

Po dosazení do výše uvedeného vzorce dostaneme

$$\vec{b}_3^* = (5, 7, 7, 8) - 2(2, 1, 3, -1) = (1, 5, 1, 10).$$

Ortogonální báze je tedy ve tvaru

$$[(2, 1, 3, -1), (3, 2, -3, -1), (1, 5, 1, 10)].$$

2 MŘÍŽKY V $\boldsymbol{R^n}$

V této kapitole si ukážeme základní vlastnosti mřížek a podrobněji rozebereme speciální případ mřížek v dimenzi 2 (pro zjednodušení budu již značit vektory, bez značky vektoru, tudíž $\vec{b}_1 = b_1$).

2.1 ZÁKLADNÍ VLASTNOSTI

Mřížkou v prostoru $\mathbb{R}^n$ se rozumí množina všech celočíselných lineárních kombinací dané báze, tj. množina

$$\left\{\sum_{i=1}^{n} x_i b_i : x_1, \dots, x_n \in \mathbb{Z}\right\},$$

kde $b_1, \dots, b_n$ je právě daná báze $\mathbb{R}^n$.

Zapíšeme definici mřížky dle [1]:

Definice 2.1.1. Podmnožina $M \subseteq \mathbb{R}^n$ se nazývá mřížka, pokud existuje báze $b_1, b_2, \dots, b_n$ vektorového prostoru $\mathbb{R}^n$ taková, že

$$M = \sum_{i=1}^{n} \mathbb{Z} b_i = \left\{\sum_{i=1}^{n} x_i b_i : x_1, x_2, \dots, x_n \in \mathbb{Z}\right\}.$$

Vektory $b_1, \dots, b_n$ nazýváme bází této mřížky. Mřížku M nazveme celočíselnou, pokud $M \subseteq \mathbb{Z}^n$.

Ve své práci budu používat převážně celočíselné mřížky. Pokud na začátku vektory vynásobíme společným násobkem jmenovatelů a na konci je tímto číslem vydělíme, zobecníme tak snadno algoritmy právě na případ mřížek $M \subseteq \mathbb{Q}^n$.

Každá mřížka má pro $n \geq 2$ nekonečně mnoho bází, například dvojice vektorů $(1,0),(0,1)$ a $(123,124),(124,125)$ jsou bázemi stejné mřížky $M = \mathbb{Z}^2$. Pokusíme se nalézt relativně krátkou bázi dané celočíselné mřížky. Úplně nejlepší by bylo najít bázi složenou z co nejkratších možných vektorů. Tedy například pro $M = \mathbb{Z}^2$ je takovou bází zřejmě kanonická báze $(1,0),(0,1)$. Nejkratší bázi je ovšem (kromě malých hodnot n) těžké nalézt.[5]

[5] Neznáme žádný efektivní algoritmus na nalezení nejkratšího nenulového vektoru v dané mřížce. Daniele Micciancio dokázal, že je těžké najít vektor nejvýše $\sqrt{2}$-krát delší než nejkratší. V roce 2004 ukázal Subhash Khot, že je těžké najít vektor t-krát delší než nejkratší pro libovolnou konstantu t, viz.[1].

LLL algoritmus je v tomto případě kompromisem, jelikož je schopen najít v polynomiálním čase bázi, která není o mnoho horší než ta optimální. Ve speciálním případě, nejkratší vektor nalezené báze bude nejvýše $\left(2^{\frac{n-1}{2}}\right)$-krát delší než nejkratší nenulový vektor mřížky.

Nyní si uvedeme tvrzení, které budeme používat v dalším textu.

Tvrzení 2.1.1. Dvě báze prostoru $\mathbb{R}^n$ jsou bází stejné mřížky právě tehdy, když je matice přechodu od jedné ke druhé celočíselná s determinantem ± 1.

Důkaz (dle [1]). Nechť $B = (b_1, b_2, \dots, b_n)$ a $C = (c_1, c_2, \dots, c_n)$ jsou báze prostoru $\mathbb{R}^n$, označíme příslušné matice $\bar{B}, \bar{C}$, označíme X matici přechodu od B k C a Y matici přechodu od C k B. Tedy $\bar{C} = \bar{B}X$ a $\bar{B} = \bar{C}Y$ a platí, že $Y = X^{-1}$.

a) ($\Rightarrow$) Budeme předpokládat, že B a C jsou báze stejné mřížky. Platí, že X je celočíselná, protože každý vektor v bázi C je celočíselnou lineární kombinací vektorů z báze B. A podobně Y je celočíselná, protože každý vektor v bázi B je celočíselnou lineární kombinací vektorů z báze C. Navíc je XY jednotková matice, tudíž podle věty o determinantu součinu je $\det \mathrm{X} \det \mathrm{Y} = 1$, takže $\det X = \pm 1$.

b) ($\Leftarrow$) Budeme předpokládat, že X je celočíselná a $|\det X| = 1$. Pak je také Y celočíselná, jak plyne například z vyjádření inverzní matice jako podílu adjungované a determinantu. Tedy každý vektor z B je celočíselnou lineární kombinací vektorů z C a naopak, čili B a C jsou bázi stejné mřížky.

Determinant je důležitým parametrem mřížky, takže si uvedeme jeho definici.

Definice 2.1.2. Determinantem mřížky M s bází $b_1, b_2, \dots, b_n$ rozumíme číslo

$$d(M) = |\det(b_1|b_2| \dots |b_n)|.$$

Determinant tedy vlastně určuje n-rozměrný objem rovnoběžnostěnu, který je určen bázovými vektory. Jedním z důsledků tvrzení 2.1.1 je, že determinant není závislý na volbě báze.

Tvrzení 2.1.2. Determinant mřížky nezávisí na volbě báze a platí

$$d(M) = \|b_1^*\| \, \|b_2^*\| \dots \|b_n^*\| = \sqrt{G_{b_1, b_2, \dots, b_n}} \leq \|b_1\| \, \|b_2\| \dots \|b_n\|.$$

Důkaz. Máme-li dvě matice $\bar{B}, \bar{C}$, u kterých jejich sloupce tvoří báze mřížky M, pak je $\bar{B} = \bar{C}X$ (podle tvrzení 2.1.1), kde $\det X = \pm 1$. Pokud použijeme větu o determinantu součinu, je $|\det(\bar{B})| = |\det(\bar{C})|$. Zbytek již vyplývá z tvrzení 1.3.1.

Když to velmi obecně shrneme, tak čím je determinant nižší, tím kratší vektory v mřížce existují.

2.2 MŘÍŽKY V DIMENZI $n = 2$ A GAUSSOVA REDUKCE MŘÍŽKY

V případě takové mřížky, můžeme efektivně najít přímo nejkratší bázi celočíselné mřížky.

Definice 2.2.1. Bázi (b_1, b_2) celočíselné mřížky $M \subseteq \mathbb{Z}^2$ nazveme nejkratší, pokud

1) b_1 je nejkratší nenulový vektor z M a
2) b_2 je nejkratší vektor $M \backslash \langle b_1 \rangle$.

Příklad 2.2.1[převzato z [1]]. $(12, 2), (13, 4)$; $(1, 2), (11, 0)$ a $(1, 2), (9, -4)$ jsou tři báze stejné mřížky M (zde je $d(M) = 22$). Uvidíme, že třetí báze je nejkratší bází této mřížky.

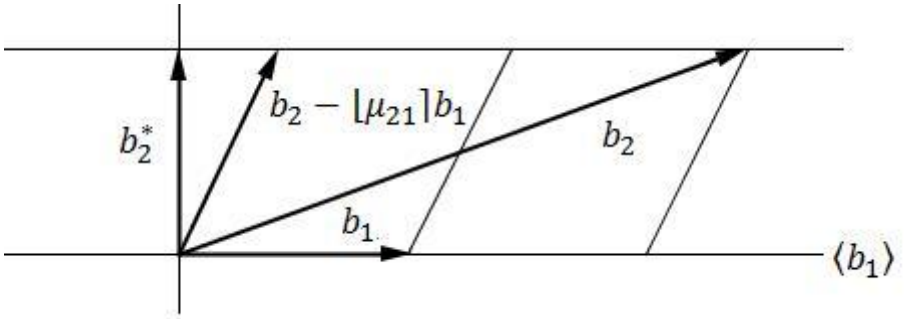
Algoritmu, který použijeme na hledání nejkratší báze, se říká Gaussova redukce mřížky[6]. Předpoklad je takový, že vektory $(12, 2), (13, 4)$ svírají příliš malý úhel, zato vektory $(1, 2), (9, -4)$ svírají úhel přibližně 87 stupňů, takže jsou „téměř kolmé“. Stačí si uvědomit, že pokud jsou vektory báze na sebe kolmé, je tato báze skutečně nejkratší (případně po prohození vektorů).

Tento předpoklad nás vede k provedení celočíselné aproximace Gram-Schmidtovy ortogonalizace. V první řadě uspořádáme vektory takovým způsobem, aby vektor b_1 nebyl delší než vektor b_2, a pak položíme

$$b_2' = b_2 - \lfloor \mu_{21} \rceil b_1,$$

kde $\lfloor \mu_{21} \rceil$ je nejbližší celé číslo k číslu μ_{21}. Je patrné, že buď $b_2' = b_2$, nebo vektory b_1, b_2' svírají úhel, který je bližší $\frac{\pi}{2}$ než vektory b_1, b_2. Tento postup stále opakujeme, dokud dochází k nějaké změně. Na začátku provádíme prohození, protože číslo $\lfloor \mu_{21} \rceil$ bude spíše nenulové, pokud je b_1 kratší než b_2 (dle definice nebo obrázku)

[6] dle některých zdrojů je toto pojmenování mylné, protože se stejným objevem přišel dříve Lagrange ,viz[1].



Obrázek 2

Nyní si ukážeme algoritmus pro hledání nejkratší mřížky.

Algoritmus (Gaussova redukce mřížky)

vstup: báze (b_1, b_2) mřížky $M \subseteq \mathbb{Z}^2$

výstup: nejkratší báze mřížky M

1. repeat

   if$\|b_1\| > \|b_2\|$ then prohoď b_1, b_2

   x:= $\lfloor \mu_{21} \rceil = \left\lfloor \frac{b_1 . b_2}{\|b_1\|^2} \right\rceil$

   b_2:= $b_2 - x b_1$

   until $x = 0$

2. return (b_1, b_2)

To, že algoritmus skončí, je patrné z toho, že při každém průběhu cyklem se delší z vektorů b_1, b_2 zkrátí a mřížka obsahuje pouze konečné množství bodů s menší velikostí než je předem dané číslo. Nyní si musíme dokázat, že tento algoritmus funguje.

Důkaz [podle [1]]. Nechť b_1, b_2 je báze mřížky M vrácená algoritmem a uvažujme libovolný nenulový vektor v v mřížce M, tzn.

$$v = x b_1 + y b_2, \qquad x, y \in \mathbb{Z}, \qquad xy \neq 0.$$

Druhá mocnina jeho normy vyjde

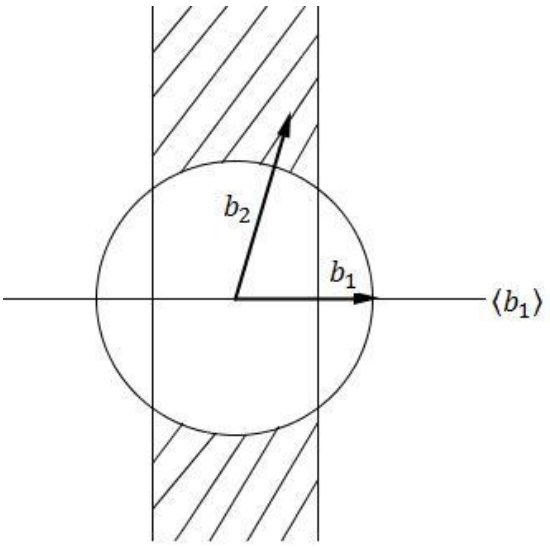
$$\|v\|^2 = (x b_1 + y b_2).(x b_1 + y b_2) = x^2 \|b_1\|^2 + 2xy(b_1 . b_2) + y^2 \|b_2\|^2.$$

Protože $|\mu_{21}| \leq \frac{1}{2}$, platí $|b_1 . b_2| \leq \frac{1}{2}\|b_1\|^2$, takže

$$\|v\|^2 \geq x^2 \|b_1\|^2 - xy\|b_1\|^2 + y^2 \|b_2\|^2.$$

Když je $x = 0$, pak je tento výraz roven alespoň $\|b_2\|^2$. Když je $y = 0$, pak je tento výraz roven alespoň $\|b_1\|^2$. Dále, pokud je $0 < |y| \leq |x|$, pak $x^2 - xy \geq 0$, z čehož plyne, že součet prvních dvou výrazů je kladný a výsledek bude alespoň $\|b_2\|^2$. A pokud bude $0 < |x| < |y|$, pak $y^2 - xy \geq 1$ a součet druhého a třetího členu je alespoň $\|b_2\|^2$. Ověřili jsme si, že b_1 je opravdu nejkratším vektorem mřížky a všechny vektory z $M \setminus \langle b_1 \rangle$ mají délku alespoň $\|b_2\|$.

Nyní se zaměříme na vzájemnou polohu výstupních vektorů b_1, b_2. Vektor b_2 je nejméně tak dlouhý jako b_1 a ortogonální projekcí b_2 na podprostor $\langle b_1 \rangle$ je vektor $\mu_{21} b_1$, kdy $|\mu_{21}| \leq \frac{1}{2}$. Vektor b_2 tedy leží v části roviny znázorněné na obrázku. Z něho lze vidět, že vektory b_1, b_2 svírají úhel v intervalech $\pm\langle 60^\circ, 120^\circ \rangle$ a že $\|b_2^*\| \geq \frac{\sqrt{3}}{2}\|b_1^*\|$. Tento vztah je motivací definice LLL-redukované báze a o které bude řeč v následující kapitole.



Obrázek 3

2.3 PŘÍKLADY

Příklad 2.3.1. Najdeme nejkratší bázi mřížky dané bázovými vektory

$$b_1 = (2, 8), b_2 = (5, 25).$$

Podle výše uvedeného algoritmu vypočítáme

$$x = \lfloor \mu_{21} \rceil = \left\lfloor \frac{b_1 . b_2}{\|b_1\|^2} \right\rceil$$

a tedy

$$x = \left\lfloor \frac{210}{68} \right\rceil = 3.$$

Takže položíme

$$b_2 = b_2 - x b_1$$

$$b_2 = (5, 25) - 3.(2, 8) = (-1, 1).$$

Nyní prohodíme b_1, b_2, jelikož $\|b_2\| \not> \|b_1\|$. Takže dostáváme $b_1 = (-1, 1)$, $b_2 = (2, 8)$.

Dále opakujeme stejný algoritmus, takže nyní je

$$x = \left\lfloor \frac{6}{2} \right\rceil = 3,$$

a tedy

$$b_2 = (2, 8) - 3(-1, 1) = (5, 5).$$

Nyní již platí $\|b_2\| > \|b_1\|$, takže v dalším kroku bude $x = 0$ a výstupem je dvojice vektorů $b_1 = (-1, 1)$, $b_2 = (5, 5)$.

Kontrolu správnosti řešení provedeme v Mathematice.

```
In[19]:= LatticeReduce[{{2, 8}, {5, 25}}]
Out[19]= {{-1, 1}, {5, 5}}
```

Příklad 2.3.2 [převzato z [1]]. Najdeme nejkratší bázi mřížky dané bázovými vektory

$$b_1 = (1, 5), b_2 = (6, 21).$$

Podle stejného algoritmu jako u předchozího příkladu vyjde

$$x = \left\lfloor \frac{111}{26} \right\rceil = 4,$$

tudíž položíme

$$b_2 = (6, 21) - 4.(1, 5) = (2,1).$$

V dalším kroku prohodíme vektory b_1, b_2, takže nyní $b_1 = (2, 1)$, $b_2 = (1, 5)$ a podle stejného algoritmu vyjde

$$x = \left\lfloor \frac{7}{5} \right\rceil = 1$$

a tedy

$$b_2 = (1,5) - (2,1) = (-1,4).$$

Nyní již je $\|b_2\| > \|b_1\|$, takže ve třetím kroku bude $x = 0$ a výstupem bude dvojice vektorů $(2,1)$, $(-1,4)$.

V Mathematice vyjde

```
In[20]:= LatticeReduce[{{1, 5}, {6, 21}}]
Out[20]= {{2, 1}, {-1, 4}}
```

Příklad 2.3.3 Najdeme nejkratší bázi mřížky dané bázovými vektory

$$b_1 = (-1,3), b_2 = (4,7).$$

Opět podle stejného algoritmu jako u předchozích příkladů vyjde

$$x = \left\lfloor \frac{17}{10} \right\rceil = 2,$$

tudíž položíme

$$b_2 = (4,7) - 2.(-1,3) = (6,1).$$

V dalším kroku zjistíme, že $\|b_2\| > \|b_1\|$, takže vektory neprohazujeme, x nyní vyjde rovno nule a výstupem bude dvojice vektorů $(-1,3)$, $(6,1)$.

Pro kontrolu vyjde v Mathematice

```
In[21]:= LatticeReduce[{{-1, 3}, {4, 7}}]
Out[21]= {{-1, 3}, {6, 1}}
```

Příklad 2.3.4. Najdeme nejkratší bázi mřížky dané bázovými vektory

$$b_1 = (50,35), b_2 = (21,14).$$

Podle stejného algoritmu jako u předchozích příkladů vyjde

$$x = \left\lfloor \frac{1540}{3725} \right\rceil = 0,$$

Takže vektory zůstanou stejné, ale protože $\|b_2\| \not> \|b_1\|$, musíme je prohodit, takže dostáváme

$$b_1 = (21, 14), b_2 = (\,50, 35).$$

Nyní opět vypočteme

$$x = \lfloor \mu_{21} \rceil = \left\lfloor \frac{b_1 . b_2}{\|b_1\|^2} \right\rceil,$$

tudíž

$$x = \left\lfloor \frac{1540}{637} \right\rceil = 2,$$

a po dosazení do příslušného vzorce získáme

$$b_2 = (50, 35) - 2.(21, 14) = (8, 7).$$

Protože opět $\|b_2\| \not> \|b_1\|$, prohodíme vektory a máme

$$b_1 = (8, 7), b_2 = (\,21, 14).$$

Opakujeme stejný algoritmus a vyjde

$$x = \left\lfloor \frac{266}{113} \right\rceil = 2$$

a tedy

$$b_2 = (21, 14) - 2.(8, 7) = (5, 0).$$

Opět prohodíme vektory, tudíž nyní máme

$$b_1 = (5, 0), b_2 = (\,8, 7)$$

a znovu použijeme algoritmus, takže

$$x = \left\lfloor \frac{40}{25} \right\rceil = 2,$$

$$b_2 = (8, 7) - 2.(5, 0) = (-2, 7).$$

Nyní již platí $\|b_2\| > \|b_1\|$, takže vektory již dále neprohazujeme, x nyní vyjde rovno nule a výstupem bude dvojice vektorů $(5, 0)$, $(-2, 7)$.

3 LENSTRA-LENSTRA-LOVÁSZŮV ALGORITMUS

Tento algoritmus byl objeven Arjenem Lenstrou, Heindrikem Lenstrou a László Lovászem roku 1982 a nejprve sloužil k návrhu prvního polynomiálního algoritmu na rozklad celočíselných polynomů, na hledání celočíselných závislostí mezi čísly, na hledání simultánních diofantických aproximací a na řešení problému celočíselného lineárního programování v pevné dimenzi. Později se tento algoritmus aplikoval také například v kryptologii[7], nebo při testování různých hypotéz v teorii čísel. Některé aplikace si ukážeme později.

Nejprve si zavedeme pojem LLL-redukované báze mřížky a poté algoritmus, který takovou bázi najde.

3.1 LLL-REDUKOVANÁ BÁZE

Představa redukované báze je poměrně stará, ale dlouhou dobu nebyl znám žádný algoritmus, který by redukovanou bázi dokázal najít v rozumném čase pro dimenzi $n > 2$.[8] S opravdovým převratem přišli ovšem v roce 1982 A. K. Lenstra, H. W. Lenstra a L. Lovász[9], kteří zavedli novou definici redukované báze a zároveň předložili i algoritmus, který pro libovolnou dimenzi najde redukovanou bázi v polynomiálním čase.

Již jsme si v předchozí části ukázali, že pokud chceme najít ne příliš dlouhou bázi nějaké mřížky, musíme nejprve zajistit, aby její vektory na sebe byly dostatečně kolmé. Toho, že všechny koeficienty μ_{ij} budou v absolutní hodnotě menší než $\frac{1}{2}$, docílíme pomocí celočíselné aproximace Gram-Schmidtovy ortogonalizace. Tato báze se někdy nazývá *redukovaná báze vzhledem k velikosti* (viz. podmínka P_1, kterou uvedeme níže).

Kolmost je zaručena redukovaností vzhledem k velikosti jen v případě, že velikosti vektorů příliš neklesají. Musíme tedy najít dostatečně silnou podmínku na to, aby byla báze dost krátká pro aplikace, ale zároveň dostatečně slabou na to, abychom takovou bázi mohli najít v rozumném (polynomiálním) čase. Takováto podmínka je uvedena v následující definici (viz. podmínka P_2).

Definice 3.1.1. Báze $b_1, \dots, b_n$ mřížky $M \subseteq \mathbb{R}^n$ se nazývá LLL-redukovaná, jestliže

[7] Kryptologií zde rozumíme nauku o šifrování, která zahrnuje kryptografii - vědu o vývoji šifrovacích systémů i kryptoanalýzu - umění prolomit šifrovací systémy.
[8] Pro dimenzi 2 popsal algoritmus s kvadratickou časovou složitostí na počátku 19. století Gauss a mnohem později ještě objevil kubický algoritmus pro dimenzi 3 Vallée.
[9] Více v [11].

(P_1) $|u_{ij}| \leq \frac{1}{2}$ pro všechna $1 \leq j < i \leq n$

(P_2) $\|b_i^*\|^2 \geq (\frac{3}{4} - u_{i\ i-1}^2)\|b_{i-1}^*\|^2$ pro všechna $1 < i \leq n$.

Podmínku (P_2) můžeme chápat jako zeslabenou podmínku $\|b_i^*\|^2 \geq \frac{3}{4}\|b_1^*\|^2$, která je platná po té, co skončí algoritmus na redukci mřížky v dimenzi 2. Když podmínku (P_2) upravíme, dostaneme

$$\|b_i^*\|^2 + u_{i\ i-1}^2\|b_{i-1}^*\|^2 \geq \frac{3}{4}\|b_{i-1}^*\|^2.$$

Pokud dále využijeme kolmost vektorů b_i^* a b_{i+1}^* dostaneme ekvivalentní podmínku

(P_2') $\|b_i^* + u_{i\ i-1}b_{i-1}^*\|^2 \geq \frac{3}{4}\|b_{i-1}^*\|^2$ pro všechna $1 < i \leq n$.

Můžeme si všimnout, že vektory $b_i^* + u_{i\ i-1}b_{i-1}^*$ a b_{i-1}^* jsou kolmice vektorů b_i a b_{i-1} na podprostor $\langle b_1, \dots, b_{i-2}\rangle$, tudíž podmínku (P_2') lze chápat tak, že pokud promítneme dvourozměrnou mřížku generovanou vektory b_i a b_{i-1} na ortogonální doplněk prostoru $\langle b_1, \dots, b_{i-2}\rangle$, pak je téměř splněna podmínka z výše uvedeného algoritmu na uspořádání vektorů podle velikosti, až na faktor $\frac{3}{4}$.

Nyní si dokážeme několik vlastností LLL-redukovaných bází.

Začneme vztahy mezi velikostmi vektorů b_i^* pro různá i. Z podmínek (P_1) a (P_2) vidíme, že $\|b_i^*\|^2 \geq \frac{1}{2}\|b_{i-1}^*\|^2$. Z toho pomocí indukce plyne, že

$$\left\|b_j^*\right\|^2 \geq 2^{i-j}\|b_i^*\|^2$$

pro všechna $1 \leq i \leq j \leq n$. Dále provedeme odhad velikosti vektorů b_i pomocí b_j^*.

Lemma 3.1.1. Pro libovolnou LLL-redukovanou bázi a každé $1 \leq i \leq j \leq n$ platí

$$\|b_i\|^2 \leq 2^{j-1}\left\|b_j^*\right\|^2.$$

Důkaz. Vzhledem k tomu, že

$$b_i = b_i^* + \sum_{k=1}^{i-1} \mu_{ik} b_k^*,$$

a protože vektory $b_1^*, b_2^*, \dots, b_i^*$ jsou vzájemně kolmé, dostaneme

$$\|b_i\|^2 = \|b_i^*\|^2 + \sum_{k=1}^{i-1} \mu_{ik}^2 \|b_k^*\|^2.$$

Když použijeme podmínku (P_1) a nerovnosti mezi vektory $b_1^*, b_2^*, \ldots$, které jsme odvodili výše, dostaneme

$$\|b_i\|^2 \leq \|b_i^*\|^2 + \sum_{k=1}^{i-1} \frac{1}{4} 2^{i-k} \|b_i^*\|^2 = \|b_i^*\|^2 (1 + \frac{1}{4}(2^i - 2)) \leq 2^{i-1} \|b_i^*\|^2 \leq$$

$$\leq 2^{i-1} 2^{j-i} \left\|b_j^*\right\|^2 = 2^{j-1} \left\|b_j^*\right\|^2.$$

Důsledkem je odhad součinu velikostí vektorů LLL-redukované báze a velikosti vektoru, který je v této bázi první, v závislosti na determinantu mřížky.

Tvrzení 3.1.1. Pro libovolnou LLL-redukovanou bázi $b_1, b_2, \ldots, b_n$ mřížky M platí

$$d(M) \leq \|b_1\| \|b_2\| \ldots \|b_n\| \leq 2^{\frac{n(n-1)}{4}} d(M) \quad a \quad \|b_1\| \leq 2^{\frac{n-1}{4}} \sqrt[n]{d(M)}.$$

Důkaz. Nerovnost $d(M) \leq \|b_1\| \|b_2\| \ldots \|b_n\|$ jsme již uvedli a dokázali v tvrzení 2.1.2. Další nerovnost lze odvodit z předchozího lemmatu:

$$\prod_{i=1}^{n} \|b_i\| \leq \prod_{i=1}^{n} 2^{\frac{i-1}{2}} \|b_i^*\| = 2^{\frac{n(n-1)}{4}} d(M).$$

Pokud tuto nerovnost vynásobíme nerovností $\|b_1\|^2 \leq 2^{i-1} \|b_i^*\|^2$ přes všechna $1 \leq i \leq n$, dostaneme

$$\|b_1\|^{2n} \leq \prod_{i=1}^{n} 2^{i-1} \|b_i^*\|^2 = 2^{\frac{n(n-1)}{2}} d(M)^2.$$

Pokud tuto nerovnost odmocníme, získáme výsledek.

Číslo

$$\frac{\|b_1\| \|b_2\| \ldots \|b_n\|}{d(M)}$$

se nazývá *defekt kolmosti* báze $b_1, b_2, \ldots, b_n$.

V předchozím tvrzení jsme dokázali, že u LLL-redukované báze je defekt kolmosti roven nejvýše $2^{\frac{n(n-1)}{4}}$. Pokud bychom chtěli stanovit nejmenší defekt kolmosti báze zadané mřížky, tak nejbližší dosud známý odhad je $\mathcal{O}\left(n^{\frac{1}{4}}(0{,}97n)^n\right)$.[10]

Nyní si ukážeme odvození dolního odhadu velikosti nejkratšího nenulového vektoru mřížky za pomoci velikosti prvního vektoru LLL-redukované báze.

Tvrzení 3.1.2. Pro libovolnou LLL-redukovanou bázi $b_1, b_2, \dots, b_n$ mřížky M a libovolný vektor $0 \neq v \in M$ platí

$$\|b_1\| \leq 2^{\frac{n-1}{2}}\|v\|.$$

Důkaz. Vzhledem k tomu, že $v \in M$, existují taková celá čísla $x_1, x_2, \dots, x_n$, že $v = \sum_{i=1}^{n} x_i b_i$. Označíme k největší index takový, že $x_k \neq 0$, tudíž $\sum_{i=1}^{k} x_i b_i$, $x_k \neq 0$. Protože $b_i = b_i^* + \sum_{j=1}^{i-1} \mu_{ij}\, b_j^*$, tak platí $v = x_k b_k^* + \sum_{j=1}^{k-1} v_j\, b_j^*$ pro určitá reálná čísla v_j. Takže dostaneme

$$\|v\|^2 = x_k^2\|b_k^*\|^2 + \sum_{j=1}^{k-1} v_j^2 \left\|b_j^*\right\|^2 \geq \|b_k^*\|^2.$$

Nyní již pouze stačí použít lemma 3.1.1 a důkaz je hotov.

V tvrzeních 3.1.1 a 3.1.2 i v lemmatu 3.1.1 jsme vlastně místo podmínky (P_2) využili slabší podmínku $\|b_i^*\|^2 \geq \frac{1}{2}\|b_{i-1}^*\|^2$, kterou se někdy redukovaná báze zavádí.

3.2 LLL ALGORITMUS

LLL algoritmus nám umožňuje najít LLL-redukovanou bázi celočíselné mřížky, která je zadaná její libovolnou bází.

Jeho kostra je následující: nejprve provedeme Gram-Schmidtův ortogonalizační proces, dále provedeme celočíselnou aproximaci tohoto procesu a dostaneme novou bázi, která splňuje podmínku (P_1) a v posledním kroku vyzkoušíme podmínku (P_2). Pokud tato podmínka není pro nějaké i splněna, prohodíme vektory b_i a b_{i-1}, vrátíme se zpět na začátek a celý cyklus opakujeme.

[10] Uvedeno v [1] str. 169.

LLL algoritmus

vstup: báze $b_1, \dots, b_n$ mřížky $M \subseteq \mathbb{Z}^n$

výstup: LLL-redukovaná báze mřížky M

1. pomocí Gram-Schmidtovy ortogonalizace spočítej $b_1^*, \dots, b_n^*$ a μ_{ij}, $1 \leq j < i \leq n$
2. for $i = 2, \dots, n$ do

   for $j = i - 1, \dots, 1$ do

   $x \coloneqq \lfloor \mu_{ij} \rceil$

   $b_i \coloneqq b_i - x b_j$

   $\mu_{ij} \coloneqq \mu_{ij} - x$

   for $l = 1, \dots, j - 1$ do $\mu_{il} \coloneqq \mu_{il} - x\mu_{jl}$
3. for $i = 2, \dots, n$ do

   if $\|b_i^*\|^2 < (\frac{3}{4} - \mu_{i\ i-1}^2)\|b_{i-1}^*\|^2$ then

   prohoď b_i a b_{i-1}

   goto 1.
4. return $b_1, b_2, \dots, b_n$.

Když ukážeme, že hodnoty μ_{ij} jsou průběžně správně aktualizovány, pak je zřejmé, že po provedení 2. kroku bude splněna podmínka (P_1), což si nyní dokážeme.

Lemma 3.2.1. Po provedení kroku 2. je splněna podmínka (P_1).

Důkaz. Nové hodnoty pro b_k^*, b_k, μ_{kl} po odečtení xb_j od vektoru b_i označíme c_k^*, c_k, v_{kl}. Takže $c_i = b_i - xb_j$ a $c_k = b_k$ pro $k \neq i$. Protože c_i vznikl tak, že jsme k b_i přičetli vektor z prostoru $\langle b_1, \dots, b_{i-1} \rangle$, platí $c_k^* = b_k^*$ pro každé $1 \leq k \leq n$. Pro k, l taková, že platí $l < k \neq i$, je

$$v_{kl} = \frac{c_k c_l^*}{\|c_l^*\|^2} = \frac{b_k b_l^*}{\|b_l^*\|^2} = \mu_{kl}.$$

Vektor b_l^* je kolmý na $b_j$pro takové l, že platí $j < l < i$, tudíž

$$v_{il} = \frac{c_i c_l^*}{\|c_l^*\|^2} = \frac{(b_i - xb_j)b_l^*}{\|b_l^*\|^2} = \frac{b_i b_l^*}{\|b_l^*\|^2} = \mu_{il}.$$

Vzhledem k tomu, že $b_j b_j^* = b_j^* b_j^*$, platí

$$v_{ij} = \frac{c_i c_j^*}{\|c_j^*\|^2} = \frac{(b_i - xb_j)b_j^*}{\|b_j^*\|^2} = \mu_{ij} - x.$$

Na konec pro $l < j$ máme

$$v_{il} = \frac{c_i c_l^*}{\|c_l^*\|^2} = \frac{(b_i - xb_j)b_l^*}{\|b_l^*\|^2} = \mu_{il} - x\mu_{jl}.$$

Tím jsme ukázali, že jsou hodnoty μ_{kl} pro i, j aktualizovány správně a nová hodnota pro μ_{ij} je maximálně $\frac{1}{2}$. K tomu jsou hodnoty jiné než $\mu_{il}, l \leq j$ neměnné, čím máme díky pořadí provádění cyklů zaručeno, že po 2. kroku jsou μ_{kl} aktualizovány správně.

Nyní víme, že když LLL algoritmus skončí, jsou u výsledné báze splněny obě podmínky (P_1) i (P_2).

Dále si musíme dokázat, že LLL-algoritmus skončí v polynomiálním čase vzhledem k velikosti vstupu.[11]

K tomu budeme potřebovat hodnoty D a $d_i, 1 \leq i \leq n$:

$$D = \prod_{i=1}^{n} d_i, \quad kde \quad d_i = \prod_{j=1}^{i} \|b_j^*\|^2.$$

Číslo d_i se tedy rovná druhé mocnině objemu rovnoběžnostěnu, který je určen prvními i vektory dané báze, $d_n = d(M)^2$. Vrátíme-li se k tvrzení 1.3.1, pak dle toho tvrzení platí

$$d_i = \det\left(G_{b_1, b_2, \ldots, b_i}\right).$$

Nyní si ukážeme, jak je zaručeno, že se do 1. kroku nevrátíme mnohokrát. Je to díky tomu, že hodnota D není vzhledem k velikosti vstupu příliš velká, v druhém kroku je neměnná a ve třetím kroku se minimálně $\frac{3}{4}$-krát zmenší, což si nyní musíme dokázat.

[11] Více lze najít v [7].

Lemma 3.2.2. Velikost D i počet návratů do kroku 1. je polynomiální vzhledem k velikosti vstupu.

Důkaz. Velikost vstupu lze odhadnout zdola hodnotou

$$R = \max\left(n, \log(\max_j \|b_j\|)\right),$$

protože každý vektor potřebuje alespoň jeden bit a vektor normy r potřebuje alespoň $\log r$ bitů.

Číslo d_i je zřejmě menší než $(\max_j \|b_j\|)^i$, číslo D tedy na začátku splňuje

$$D \leq \left(\max_j \|b_j\|\right)^{n(n-1)}.$$

Jak jsme již dokázali u předchozího lemmatu, druhý krok nemění hodnoty b_i^*, tedy nemění ani hodnoty D.

Když prohodíme vektory b_i a b_{i-1}, tak se vektory $b_1^*, \dots, b_{i-2}^*$ nezmění, a tudíž se nezmění ani hodnoty $d_1, \dots, d_{i-2}$. To, že se nezmění ani čísla $d_i, \dots, d_n$ vidíme z vyjádření $d_j = \det(G_{b_1, b_2, \dots, b_j})$. Dále si označme c_{i-1}^* novou hodnotu b_{i-1}^*. Vektor c_{i-1}^* je kolmicí vektoru b_i k podprostoru $\langle b_1, \dots, b_{i-2}\rangle$, takže $c_{i-1}^* = b_i^* + \mu_{i\,i-1} b_{i-1}^*$. Že norma tohoto vektoru je $\leq \frac{3}{4}\|b_{i-1}^*\|$ vidíme z ekvivalence (P_2) a (P'_2), kterou jsme již odvodili dříve. Z toho důvodu se číslo d_i alespoň $\frac{3}{4}$-krát zmenší a tudíž i hodnota D se alespoň $\frac{3}{4}$-krát zmenší. Protože je číslo D zdola omezeno jedničkou, je podmínka z 3. kroku splněna nejvýše tolikrát:

$$\log_{\frac{4}{3}} D \leq n(n-1) \log_{\frac{4}{3}}\left(\max_j \|b_j\|\right).$$

Toto číslo je pro jistou konstantu C jistě menší než CR^3.

Je zřejmé, že během průběhu jednoho cyklu přes kroky 1., 2., 3. probíhá jen polynomiálně mnoho operací sčítání, odčítání, násobení a dělení čísel μ_{ij} a složek vektorů b_i a b_i^*. Tato čísla jsou všechna racionální. K dokončení důkazu, že LLL algoritmus pracuje v polynomiálním čase, zbývá odhadnout velikost čitatelů a jmenovatelů zúčastněných čísel. Nyní si omezíme velikost jmenovatelů a rovněž velikost vektorů b_i^* (tím samozřejmě omezíme i velikost jejich složek).

Lemma 3.2.3. Pro každé $1 \leq j \leq i \leq n$ je

$$d_{i-1}b_i^* \in \mathbb{Z}^n, \qquad \|b_i^*\| \leq D \quad a \quad d_j\mu_{ij} \in \mathbb{Z}.$$

Důkaz. Zvolíme libovolné $1 \leq i \leq n$. Výše u Gram-Schmidtova ortogonalizačního procesu jsme uvedli, že vektor b_i^* lze napsat jako $b_i^* = b_i - \sum_{j=1}^{i-1} a_j b_j$, kde $(a_1, a_2, \dots, a_{i-1})$ je řešením soustavy lineárních rovnic

$$G_{b_1,b_2,\dots,b_{i-1}}(a_1, a_2, \dots, a_{i-1})^T = (b_i.b_1, b_i.b_2, \dots, b_i.b_{i-1})^T.$$

Podle Cramerova pravidla je každé a_j podílem determinantu určité celočíselné matice a determinantu matice $G_{b_1,b_2,\dots,b_{i-1}}$. Tento podíl je roven d_{i-1}. Z toho plyne, že

$$d_{i-1}b_i^* = d_{i-1}b_i - \sum_{j=1}^{i-1} d_{i-1}\, a_j b_j \in \mathbb{Z}^n.$$

Vzhledem k tomu, že pro libovolné j je $d_{j-1}b_j^* \in \mathbb{Z}^n$, platí $\left\|d_{i-1}b_j^*\right\| \geq 1$, tudíž $\left\|b_j^*\right\| \geq \frac{1}{d_{j-1}}$. Z této nerovnosti a definice čísel d_j dostaneme

$$\|b_i^*\|^2 = \frac{d_{i+1}}{d_i} = \frac{d_{i+1}}{\|b_1^*\|^2\|b_2^*\|^2 \dots \left\|b_i^*\right\|^2} \leq d_{i+1}{d_1}^2 \dots {d_i}^2 \leq D^2$$

a tedy

$$d_j\mu_{ij} = d_j \frac{b_i \,.\, b_j^*}{\left\|b_j^*\right\|^2} = d_j \frac{b_i \,.\, b_j^*}{\frac{d_j}{d_{j-1}}} = d_{j-1}\left(b_i \,.\, b_j^*\right) = b_i \,.\left(d_{j-1} \,.\, b_j^*\right) \in \mathbb{Z}\,.$$

Lemma 3.2.4. Po provedení 2. kroku platí $\|b_i\|^2 \leq nD^2$ pro každé $1 \leq i \leq n$.

Důkaz. Vzhledem k tomu, že jsou vektory b_i^* na sebe kolmé a protože

$b_i = b_i^* + \sum_{j=1}^{i-1} \mu_{ij} b_j^*$, platí

$$\|b_i\|^2 = \|b_i^*\|^2 + \sum_{j=1}^{i-1} u_{ij}^2 \left\|b_j^*\right\|^2 .$$

Když využijeme nerovnost $\mu_{ij} \leq \frac{1}{2}$ a předchozí lemma dostaneme

$$\|b_i\|^2 \leq D^2 + \sum_{j=1}^{i-1} \frac{1}{4} D^2 \leq D^2 + \frac{n}{4} D^2 \leq nD^2.$$

Pokud využijeme všechny tyto poznatky, dostaneme polynomiální mez pro časovou složitost LLL algoritmu, což si shrneme v důkazu následujícího tvrzení.

Tvrzení 3.2.5. LLL algoritmus pracuje v polynomiálním čase v závislosti na velikosti vstupu.

Důkaz. V lemmatu 3.2.2 jsme dokázali, že počet návratů do 1. kroku je polynomiálně omezený shora. Polynomiálně mnoho operací provádíme v 1., 2. a 3. kroku. Všechny operace provádíme s racionálními čísly, které mají polynomiálně omezeného jmenovatele (což jsme ukázali v lemmatu 3.2.3) a jejichž velikost je také polynomiálně omezená, tedy i čitatel je polynomiálně omezen. Další skutečnost jsme dokázali v lemmatech 3.2.3 a 3.2.4, ale jen po provedení 2. kroku. Zbytek můžeme snadno dokázat ze vzorců, které vystupují v Gram-Schmidtově ortogonalizaci a v 2. kroku.

3.3 PŘÍKLADY

Příklad 3.3.1.[podle [1]]. Najdeme LLL-redukovanou bázi mřížky dané bází

$$b_1 = (1,1,1), b_2 = (-1,0,2), b_3 = (3,5,6).$$

Vektory v příkladech budeme psát do řádků. Nejprve provedeme Gram-Schmidtovu ortogonalizaci.

$b_1^* = b_1 = (1,1,1),$

$$b_2^* = b_2 - \mu_{21} b_1^* = (-1,0,2) - \frac{1}{3}(1,1,1) = \left(-\frac{4}{3}, -\frac{1}{3}, \frac{5}{3}\right),$$

$$b_3^* = b_3 - \mu_{31} b_1^* - \mu_{32} b_2^* = (3,5,6) - \frac{14}{3}(1,1,1) - \frac{13}{14}\left(-\frac{4}{3}, -\frac{1}{3}, \frac{5}{3}\right) =$$

$$= \left(-\frac{6}{14}, \frac{9}{14}, -\frac{3}{14}\right).$$

Nyní provedeme 2. krok. Pro $i = 2, j = 1$ je $x = \lfloor \mu_{21} \rceil = 0$, takže se nic nezmění. Pro $i = 3, j = 2$ vyjde

$$x = \lfloor \mu_{32} \rceil = 1, \quad b_3 = b_3 - x b_2 = (3,5,6) - (-1,0,2) = (4,5,4),$$

$$\mu_{32} = \mu_{32} - 1 = -\frac{1}{14}, \quad \mu_{31} = \mu_{31} - \mu_{21} = \frac{14}{3} - \frac{1}{3} = \frac{13}{3}.$$

Pro $i = 3, j = 1$ vyjde

$$x = \lfloor \mu_{31} \rceil = 4, \quad b_3 = (4,5,4) - 4(1,1,1) = (0,1,0),$$

$$\mu_{31} = \mu_{31} - 4 = \frac{13}{3} - \frac{12}{3} = \frac{1}{3}.$$

Po 2. kroku máme

$$b_1 = (1,1,1), b_2 = (-1,0,2), b_3 = (0,1,0),$$

$$\mu_{21} = \frac{1}{3}, \quad \mu_{31} = \frac{1}{3}, \quad \mu_{32} = -\frac{1}{14}.$$

Nyní přejdeme ke kroku 3., kde porovnáváme $\|b_i^*\|^2$ a $\left(\frac{3}{4} - \mu_{i\ i-1}^2\right)\|b_{i-1}^*\|^2$, takže

$$\|b_2^*\|^2 = \frac{14}{3} > \frac{69}{36} = \left(\frac{3}{4} - \mu_{21}^2\right)\|b_1^*\|^2,$$

ale

$$\|b_3^*\|^2 = \frac{9}{14} < \frac{73}{21} = \left(\frac{3}{4} - \mu_{32}^2\right)\|b_2^*\|^2,$$

takže musíme prohodit vektory b_2 a b_3 a dostáváme

$$b_1 = (1,1,1), b_2 = (0,1,0), b_3 = (-1,0,2).$$

Nyní se vrátíme na začátek a opět provedeme Gram-Schmidtovu ortogonalizaci

$b_1^* = b_1 = (1,1,1),$

$b_2^* = b_2 - \mu_{21}b_1^* = (0,1,0) - \frac{1}{3}(1,1,1) = \left(-\frac{1}{3}, \frac{2}{3}, -\frac{1}{3}\right),$

$b_3^* = b_3 - \mu_{31}b_1^* - \mu_{32}b_2^* = (-1,0,2) - \frac{1}{3}(1,1,1) + \frac{1}{2}\left(-\frac{1}{3}, \frac{2}{3}, -\frac{1}{3}\right) == \left(-\frac{3}{2}, 0, \frac{3}{2}\right).$

Krok 2. proběhne beze změn, když si vybereme, že $\frac{1}{2}$ zaokrouhlíme dolů.

Nyní provedeme 3. krok, kde opět porovnáváme

$$\|b_2^*\|^2 = \frac{2}{3} < \frac{23}{12} = \left(\frac{3}{4} - \mu_{21}^2\right)\|b_1^*\|^2,$$

a

$$\|b_3^*\|^2 = \frac{9}{2} > \frac{1}{3} = \left(\frac{3}{4} - \mu_{32}^2\right)\|b_2^*\|^2.$$

Zjistili jsme, že nyní musíme prohodit vektory b_1, b_2, tedy dostaneme

$$b_1 = (0,1,0), b_2 = (1,1,1), b_3 = (-1,0,2)$$

a opět provádíme Gram-Schmidtův ortogonalizační proces, takže dostaneme

$b_1^* = b_1 = (0,1,0),$

$b_2^* = b_2 - \mu_{21} b_1^* = (1,1,1) - 1(0,1,0) = (1,0,1),$

$b_3^* = b_3 - \mu_{31} b_1^* - \mu_{32} b_2^* = (-1,0,2) - \frac{1}{2}(1,0,1) = \left(-\frac{3}{2}, 0, \frac{3}{2}\right).$

V 2. kroku dostaneme

pro $i = 2, j = 1$

$$x = \lfloor \mu_{21} \rceil = 1, \qquad b_2 = b_2 - x b_1 = (1,1,1) - (0,1,0) = (1,0,1),$$

$$\mu_{21} = \mu_{21} - 1 = 0$$

Pro $i = 3, j = 2$ se nic nezmění, stejně jako pro $i = 3, j = 1.$

Takže po 2. kroku máme

$$b_1 = (0,1,0), b_2 = (1,0,1), b_3 = (-1,0,2),$$

$$\mu_{21} = 0, \quad \mu_{31} = 0, \quad \mu_{32} = \frac{1}{2}.$$

Ve 3. kroku porovnáme

$$\|b_2^*\|^2 = 2 > \frac{3}{4} = \left(\frac{3}{4} - \mu_{21}^2\right) \|b_1^*\|^2,$$

a

$$\|b_3^*\|^2 = \frac{9}{2} > 1 = \left(\frac{3}{4} - \mu_{32}^2\right) \|b_2^*\|^2,$$

čímž jsme zjistili, že je splněná podmínka (P_2), takže jsme nalezli LLL-redukovanou bázi.

Ověření správnosti výsledku lze provést opět v programu Mathematica, tedy

```
LatticeReduce[{{1, 1, 1}, {-1, 0, 2}, {3, 5, 6}}]
{{0, 1, 0}, {1, 0, 1}, {-1, 0, 2}}
```

Příklad 3.3.2. Najdi LLL-redukovanou bázi mřížky, která je daná bází

$$b_1 = (-1,5,0), b_2 = (2,5,0), b_3 = (8,6,16).$$

Nejprve provedeme Gram-Schmidtovu ortogonalizaci.

$b_1^* = b_1 = (-1, 5, 0)$,

$b_2^* = b_2 - \mu_{21} b_1^* = (2, 5, 0) - \frac{23}{26}(-1, 5, 0) = \left(\frac{75}{26}, \frac{15}{26}, 0\right)$,

$b_3^* = b_3 - \mu_{31} b_1^* - \mu_{32} b_2^* = (8, 6, 16) - \frac{11}{13}(-1, 5, 0) - \frac{46}{15}\left(\frac{75}{26}, \frac{15}{26}, 0\right) = (0, 0, 16)$.

Nyní provedeme krok 2.

Pro $i = 2, j = 1$ je

$$x = \lfloor \mu_{21} \rceil = 1, \qquad b_2 = b_2 - x b_1 = (2, 5, 0) - (-1, 5, 0) = (3, 0, 0),$$

$$\mu_{21} = \mu_{21} - 1 = \frac{23}{26} - 1 = -\frac{3}{26}$$

Pro $i = 3, j = 2$ vyjde

$$x = \lfloor \mu_{32} \rceil = 3, \qquad b_3 = b_3 - x b_2 = (8, 6, 16) - 3(3, 0, 0) = (-1, 6, 16),$$

$$\mu_{32} = \mu_{32} - 1 = \frac{1}{15}, \qquad \mu_{31} = \mu_{31} - \mu_{21} = \frac{11}{3} + \frac{3}{26} = \frac{25}{26}.$$

Pro $i = 3, j = 1$ vyjde

$$x = \lfloor \mu_{31} \rceil = 1, \qquad b_3 = (-1, 6, 16) - (1, 5, 0) = (0, 1, 16),$$

$$\mu_{31} = \mu_{31} - 1 = \frac{25}{26} - 1 = -\frac{1}{26}.$$

Po 2. kroku máme

$$b_1 = (-1, 5, 0), b_2 = (3, 0, 0), b_3 = (0, 1, 16),$$

$$\mu_{21} = -\frac{3}{26}, \quad \mu_{31} = -\frac{1}{26}, \quad \mu_{32} = \frac{1}{15}.$$

V kroku 3. porovnáváme $\|b_i^*\|^2$ a $\left(\frac{3}{4} - \mu_{i\ i-1}^2\right)\|b_{i-1}^*\|^2$, takže dostaneme

$$\|b_2^*\|^2 = \frac{225}{26} < \frac{249}{13} = \left(\frac{3}{4} - \mu_{21}^2\right)\|b_1^*\|^2,$$

a

$$\|b_3^*\|^2 = 256 > \frac{671}{104} = \left(\frac{3}{4} - \mu_{32}^2\right)\|b_2^*\|^2,$$

z čehož je patrné, že musíme prohodit vektory b_1, b_2, takže dostaneme

$$b_1 = (3,0,0), b_2 = (-1,5,0), b_3 = (0,1,16),$$

vrátíme se na začátek cyklu a znovu provedeme Gram-Schmidtovu ortogonalizaci

$b_1^* = b_1 = (3,0,0)$,

$b_2^* = b_2 - \mu_{21} b_1^* = (-1,5,0) + \frac{1}{3}(3,0,0) = (0,5,0)$,

$b_3^* = b_3 - \mu_{31} b_1^* - \mu_{32} b_2^* = (0,1,16) - \frac{1}{5}(0,5,0) = (0,0,16)$.

Krok 2 proběhne beze změn.

Po 2. Kroku tedy máme

$$b_1 = (3,0,0), b_2 = (-1,5,0), b_3 = (0,1,16),$$

$$\mu_{21} = -\frac{1}{3}, \quad \mu_{31} = 0, \quad \mu_{32} = \frac{1}{5}.$$

Znovu porovnáme

$$\|b_2^*\|^2 = 25 > \frac{9}{2} = \left(\frac{3}{4} - \mu_{21}^2\right)\|b_1^*\|^2,$$

a

$$\|b_3^*\|^2 = 256 > \frac{71}{4} = \left(\frac{3}{4} - \mu_{32}^2\right)\|b_2^*\|^2.$$

Ve 3. kroku jsme zjistili, že jsme již nalezli LLL-redukovanou bázi.

Příklad 3.3.3. Najdeme LLL-redukovanou bázi mřížky dané bází

$$b_1 = (0,3,4), b_2 = (-1,3,3), b_3 = (5,4,-7).$$

Krok 1.

$b_1^* = b_1 = (0,3,4)$,

$b_2^* = b_2 - \mu_{21} b_1^* = (-1,3,3) - \frac{21}{25}(0,3,4) = \left(-1, \frac{12}{25}, -\frac{9}{25}\right)$,

$$b_3^* = b_3 - \mu_{31} b_1^* - \mu_{32} b_2^* = (5,4,-7) + \frac{16}{25}(0,3,4) + \frac{7}{17}\left(-1, \frac{12}{25}, -\frac{9}{25}\right)$$
$$= \left(\frac{78}{17}, \frac{104}{17}, -\frac{78}{17}\right).$$

Krok 2.

Pro $i = 2, j = 1$ je

$$x = \lfloor \mu_{21} \rceil = 1, \qquad b_2 = b_2 - x b_1 = (-1, 3, 3) - (0, 3, 4) = (-1, 0, -1),$$

$$\mu_{21} = \mu_{21} - 1 = \frac{21}{25} - 1 = -\frac{4}{25}.$$

Pro $i = 3, j = 2$ se nestane nic, pouze $\mu_{31} = \mu_{31} - \mu_{21} = -\frac{12}{25}$

Pro $i = 3, j = 1$ vyjde

$$x = \lfloor \mu_{31} \rceil = -1, \quad b_3 = (5, 4, -7) + (0, 3, 4) = (5, 7, -3),$$

$$\mu_{31} = \mu_{31} + 1 = -\frac{12}{25} + 1 = \frac{13}{25}.$$

Po 2. kroku máme

$$b_1 = (0, 3, 4), b_2 = (-1, 0, -1), b_3 = (5, 7, -3),$$

$$\mu_{21} = -\frac{4}{25}, \quad \mu_{31} = \frac{13}{25}, \quad \mu_{32} = -\frac{7}{17}.$$

Krok 3.

$$\|b_2^*\|^2 = \frac{850}{625} < \frac{1811}{100} = \left(\frac{3}{4} - \mu_{21}^2\right) \|b_1^*\|^2,$$

$$\|b_3^*\|^2 = \frac{22984}{289} > \frac{1342}{612} = \left(\frac{3}{4} - \mu_{32}^2\right) \|b_2^*\|^2.$$

takže prohodíme vektory b_1, b_2 a dostaneme

$$b_1 = (-1, 0, -1), b_2 = (0, 3, 4), b_3 = (5, 7, -3)$$

a vrátíme se na začátek ke Gram-Schmidtově ortogonalizaci

$b_1^* = b_1 = (-1, 0, -1)$,

$b_2^* = b_2 - \mu_{21} b_1^* = (0, 3, 4) + 2(-1, 0, -1) = (-2, 3, 2)$,

$$b_3^* = b_3 - \mu_{31} b_1^* - \mu_{32} b_2^* = (5, 7, -3) + (-1, 0, -1) - \frac{5}{17}(-2, 3, 2) = \left(\frac{78}{17}, \frac{104}{17}, -\frac{78}{17}\right)$$

Krok 2.

Pro $i = 2, j = 1$ je

$$x = \lfloor \mu_{21} \rceil = -2, \qquad b_2 = b_2 - xb_1 = (0,3,4) + 2(-1,0,-1) = (-2,3,2),$$

$$\mu_{21} = \mu_{21} + 2 = 0.$$

Pro $i = 3, j = 2$ se nestane nic.

Pro $i = 3, j = 1$ vyjde

$$x = \lfloor \mu_{31} \rceil = -1, \quad b_3 = (5,4,-3) + (-1,0,-1) = (4,7,-4),$$

$$\mu_{31} = \mu_{31} + 1 = 0.$$

Po 2. kroku máme

$$b_1 = (-1,0,-1), b_2 = (-2,3,2), b_3 = (4,7,-4),$$

$$\mu_{21} = 0, \quad \mu_{31} = 0, \quad \mu_{32} = \frac{5}{17}.$$

Ve 3. kroku porovnáme

$$\|b_2^*\|^2 = 17 > -\frac{13}{2} = \left(\frac{3}{4} - \mu_{21}^2\right)\|b_1^*\|^2,$$

$$\|b_3^*\|^2 = \frac{22984}{289} > \frac{51}{4} = \left(\frac{3}{4} - \mu_{32}^2\right)\|b_2^*\|^2.$$

Nalezli jsme tedy LLL-redukovanou bázi.

Příklad 3.3.4. Najdeme LLL-redukovanou bázi mřížky dané bází

$$b_1 = (1,0,3,2), b_2 = (0,1,-1,3), b_3 = (0,0,-1,2).$$

Provedeme Gram-Schmidtovu ortogonalizaci.

$$b_1^* = b_1 = (1,0,3,2),$$

$$b_2^* = b_2 - \mu_{21}b_1^* = (0,1,-1,3) - \frac{3}{14}(1,0,3,2) = \left(-\frac{3}{14}, 1, -\frac{23}{14}, \frac{36}{14}\right),$$

$$b_3^* = b_3 - \mu_{31}b_1^* - \mu_{32}b_2^* = (0,0,-1,2) - \frac{1}{14}(1,0,3,2) - \frac{19}{29}\left(-\frac{3}{14}, 1, -\frac{23}{14}, \frac{36}{14}\right) =$$

$$= \left(\frac{14}{203}, -\frac{133}{203}, -\frac{28}{203}, \frac{35}{203}\right).$$

Nyní provedeme 2. krok. Pro $i = 2, j = 1$ je $x = \lfloor \mu_{21} \rceil = 0$, takže se nic nezmění. Pro $i = 3, j = 2$ vyjde

$$x = \lfloor \mu_{32} \rceil = 1, \quad b_3 = b_3 - xb_2 = (0,0,-1,2) - (0,1,-1,3)$$
$$= (0,-1,0,-1),$$

$$\mu_{32} = \mu_{32} - 1 = -\frac{10}{29}, \quad \mu_{31} = \mu_{31} - \mu_{21} = \frac{1}{14} - \frac{3}{14} = -\frac{1}{7}.$$

Pro $i = 3, j = 1$ vyjde $x = \lfloor \mu_{31} \rceil = 0$, takže po 2. kroku máme

$$b_1 = (1,0,3,2), b_2 = (0,1,-1,3), b_3 = (0,-1,0,-1),$$

$$\mu_{21} = \frac{3}{14}, \quad \mu_{31} = -\frac{2}{14}, \quad \mu_{32} = -\frac{10}{29}.$$

Dále provedeme krok 3., kde porovnáváme $\|b_i^*\|^2$ a $\left(\frac{3}{4} - \mu_{i\ i-1}^2\right)\|b_{i-1}^*\|^2$

$$\|b_2^*\|^2 = \frac{2030}{196} > \frac{138}{14} = \left(\frac{3}{4} - \mu_{21}^2\right)\|b_1^*\|^2,$$

ale

$$\|b_3^*\|^2 = \frac{19894}{41209} < \frac{307835}{47096} = \left(\frac{3}{4} - \mu_{32}^2\right)\|b_2^*\|^2,$$

Z čehož vyplývá, že musíme prohodit vektory b_2 a b_3 a dostáváme

$$b_1 = (1,0,3,2), b_2 = (0,-1,0,-1), b_3 = (0,1,-1,3).$$

Nyní se musíme vrátit na začátek a opět provést Gram-Schmidtovu ortogonalizaci

$$b_1^* = b_1 = (1,0,3,2),$$

$$b_2^* = b_2 - \mu_{21}b_1^* = (0,-1,0,-1) + \frac{1}{3}(1,0,3,2) = \left(\frac{1}{7}, -1, \frac{3}{7}, -\frac{5}{7}\right),$$

$$b_3^* = b_3 - \mu_{31}b_1^* - \mu_{32}b_2^* = (0,1,-1,3) - \frac{3}{14}(1,0,3,2) + \frac{25}{12}\left(\frac{1}{7}, -1, \frac{3}{7}, -\frac{5}{7}\right) =$$
$$= \left(\frac{1}{12}, -\frac{13}{12}, -\frac{9}{12}, \frac{13}{12}\right).$$

Dále provedeme 2. krok.

Pro $i = 2, j = 1$ je $x = \lfloor \mu_{21} \rceil = 0$, takže beze změny.

Pro $i = 3, j = 2$ vyjde

$$x = \lfloor \mu_{32} \rceil = -2, \quad b_3 = b_3 - xb_2 = (0,1,-1,3) + 2(0,-1,0,-1) = (0,-1,-1,1),$$

$$\mu_{32} = \mu_{32} + 2 = -\frac{1}{12}, \quad \mu_{31} = \mu_{31} - \mu_{21} = \frac{3}{14} + \frac{2}{14} = \frac{5}{14}.$$

Pro $i = 3, j = 1$ vyjde $x = \lfloor \mu_{31} \rceil = 0$, takže po 2. kroku dostaneme

$$b_1 = (1, 0, 3, 2), b_2 = (0, -1, 0, -1), b_3 = (0, -1, -1, 1).$$

$$\mu_{21} = -\frac{1}{7}, \quad \mu_{31} = -\frac{1}{12}, \quad \mu_{32} = \frac{5}{14}.$$

Nyní provedeme 3. krok, kde opět porovnáváme

$$\|b_2^*\|^2 = \frac{84}{49} < \frac{143}{14} = \left(\frac{3}{4} - \mu_{21}^2\right)\|b_1^*\|^2,$$

a

$$\|b_3^*\|^2 = \frac{420}{144} > \frac{107}{84} = \left(\frac{3}{4} - \mu_{32}^2\right)\|b_2^*\|^2,$$

takže nyní musíme prohodit vektory b_1, b_2. Dostaneme

$$b_1 = (0, -1, 0, -1), b_2 = (1, 0, 3, 2), b_3 = (0, -1, -1, 1)$$

a opět provádíme Gram-Schmidtův ortogonalizační proces, takže dostaneme

$b_1^* = b_1 = (0, -1, 0, -1)$,

$b_2^* = b_2 - \mu_{21} b_1^* = (1, 0, 3, 2) + (0, -1, 0, -1) = (1, -1, 3, 1)$,

$b_3^* = b_3 - \mu_{31} b_1^* - \mu_{32} b_2^* = (0, -1, -1, 1) + \frac{1}{12}(1, -1, 3, 1) = \left(\frac{1}{12}, -\frac{13}{12}, -\frac{9}{12}, \frac{13}{12}\right)$.

V 2. kroku dostaneme

pro $i = 2, j = 1$

$$x = \lfloor \mu_{21} \rceil = -1, \qquad b_2 = b_2 - x b_1 = (1, 0, 3, 2) + (0, -1, 0, -1) = (1, -1, 3, 1),$$

$$\mu_{21} = \mu_{21} + 1 = 0$$

Pro $i = 3, j = 2$ a pro $i = 3, j = 1$ se nic nezmění.

Takže po 2. kroku máme

$$b_1 = (0, -1, 0, -1), b_2 = (1, -1, 3, 1), b_3 = (0, -1, -1, 1),$$

$$\mu_{21} = 0, \quad \mu_{31} = 0, \quad \mu_{32} = -\frac{1}{12}.$$

Ve 3. kroku porovnáme

$$\|b_2^*\|^2 = 12 > \frac{6}{4} = \left(\frac{3}{4} - \mu_{21}^2\right)\|b_1^*\|^2,$$

a

$$\|b_3^*\|^2 = \frac{420}{144} < 9 = \left(\frac{3}{4} - \mu_{32}^2\right)\|b_2^*\|^2,$$

takže opět musíme prohodit 2. a 3. vektor a máme

$$b_1 = (0,-1,0,-1), b_2 = (0,-1,-1,1), b_3 = (1,-1,3,1)$$

a opakujeme cyklus od začátku.

$b_1^* = b_1 = (0,-1,0,-1)$,

$b_2^* = b_2 - \mu_{21}b_1^* = (0,-1,-1,1)$,

$b_3^* = b_3 - \mu_{31}b_1^* - \mu_{32}b_2^* = (1,-1,3,1) + \frac{1}{3}(0,-1,-1,1) = \left(1,-\frac{4}{3},\frac{8}{3},\frac{4}{3}\right)$.

Krok 2. proběhne beze změny a ve třetím kroku zjistíme, že

$$\|b_2^*\|^2 = 3 > \frac{6}{4} = \left(\frac{3}{4} - \mu_{21}^2\right)\|b_1^*\|^2,$$

a

$$\|b_3^*\|^2 = \frac{105}{9} > \frac{23}{12} = \left(\frac{3}{4} - \mu_{32}^2\right)\|b_2^*\|^2.$$

Tím jsme nalezli LLL redukovanou bázi.

4 APLIKACE LLL REDUKCE

Nejprve si stručně uvedeme oblasti, ve kterých lze LLL algoritmus využít, a dále si krátce přiblížíme vybrané aplikace, které nebudu blíže rozebírat, vzhledem k tomu, že to už by přesahovalo rozsah této práce. Na závěr si ukážeme pár příkladů na aplikaci LLL algoritmu v programu Mathematica.

LLL algoritmus se využívá především v těchto oblastech:

- Faktorizace polynomů nad celými nebo racionálními čísly, případně nad dalšími tělesy.

- Problémy teorie mřížek: Problém nejmenší báze mřížky, problém nejkratšího vektoru mřížky a problém nejuzavřenějšího vektoru mřížky (více viz [3]).

- Nalezení minimálního polynomu, jehož kořenem je zadané algebraické číslo (dané aproximací).

- Celočíselné lineární programování, které je odvětvím optimalizace. Základním problémem celočíselného lineárního programování je rozhodnutí, zda existuje celočíselné řešení *r* racionálních nerovnic o *s* neznámých.

- Pro danou posloupnost reálných čísel $(x_1, \dots, x_n)$ najít posloupnost celých čísel $(a_1, \dots, a_n)$, tak aby platilo: $\sum_{i=1}^{n} a_i x_i = 0 \wedge \exists i : a_i \neq 0$.

- Široké využití v kryptologii. Nejdříve byl LLL algoritmus používán v kryptoanalýze jako nástroj útoků na různé systémy - jako první byly prolomeny různé systémy založené na principu batohu (více viz [3]).

4.1 DIOFANTICKÉ APROXIMACE

Věta z teorie diofantických aproximací říká, že pro libovolná reálná čísla $\beta_1, \dots, \beta_n$ a $0 < \varepsilon < 1$ existují celá čísla $p_1, \dots, p_n, q$, která splňují

$$\left|\frac{p_i}{q} - \beta_i\right| \leq \frac{\varepsilon}{q} \qquad pro\ i = 1, \dots, n \quad a \quad 1 \leq q \leq \varepsilon^{-n}.$$

Aproximaci, která je jen o málo horší, můžeme nalézt v polynomiálním čase pomocí LLL algoritmu.

Tvrzení 4.2.1. Existuje polynomiální algoritmus, který pro zadaná racionální čísla $\beta_1, \dots, \beta_n$ a $0 < \varepsilon < 1$ najde celá čísla $p_1, \dots, p_n, q$ splňující

$$\left|\frac{p_i}{q} - \beta_i\right| \leq \frac{\varepsilon}{q} \qquad pro\ i = 1, \dots, n \quad a \quad 1 \leq q \leq 2^{\frac{n(n+1)}{4}} \varepsilon^{-n}.$$

Důkaz. Mějme mřížku $M \subseteq \mathbb{Q}^{n+1}$, která má bázi

$$(1, 0, \dots, 0), (0, 1, \dots, 0), \dots, (0, \dots, 1, 0) \left(-\beta_1, -\beta_2, \dots, -\beta_n, 2^{-\frac{n(n+1)}{4}} \varepsilon^{n+1}\right).$$

Poslední složku nejprve zaokrouhlíme na dostatečně přesné racionální číslo, protože nemusí být racionální. Dále označíme b_1 aproximaci nejkratšího nenulového vektoru mřížky M kterou jsme nalezli pomocí LLL algoritmu, tedy

$$b_1 = p_1(1, 0, \dots, 0) + \dots + p_n\ (0, \dots, 1, 0) + q\left(-\beta_1, -\beta_2, \dots, -\beta_n, 2^{-\frac{n(n+1)}{4}} \varepsilon^{n+1}\right)$$
$$= \left(p_1 - \beta_1 q, \dots, p_n - \beta_n q, 2^{-\frac{n(n+1)}{4}} \varepsilon^{n+1}\right)$$

pro celá čísla $p_1, \dots, p_n, q$, která se dají spočítat v polynomiálním čase z vektoru b_1 jako řešení soustavy lineárních rovnic. S použitím tvrzení 3.1.1 dostaneme

$$\|b_1\| \leq 2^{\frac{n}{4}} \sqrt[n+1]{2^{-\frac{n(n+1)}{4}} \varepsilon^{n+1}} = \varepsilon.$$

Vzhledem k tomu, že velikost vektoru b_1 je nejvýše ε, jsou absolutní hodnoty všech složek menší než ε, což jsou po úpravě nerovnosti z tvrzení.

4.2 HLEDÁNÍ CELOČÍSELNÝCH VZTAHŮ MEZI ČÍSLY

Celočíselná závislost mezi reálnými čísly $\beta_1, \dots, \beta_n$ je rovnost

$$s_1\beta_1 + \dots + s_n\beta_n = 0,$$

kde $s_1, \dots, s_n$ jsou celá čísla. Abychom nalezli takový vztah, zkusíme najít krátký vektor b_1 v mřížce $M \subseteq \mathbb{Z}^{n+1}$, která je daná bází

$$(1, 0, \dots, 0, \lfloor N\beta_1 \rceil), (0, 1, \dots, 0, \lfloor N\beta_2 \rceil), \dots, (0, 0, \dots, 1, \lfloor N\beta_n \rceil),$$

kde N je dostatečně velké číslo. Vektor b_1 leží v mřížce M, tudíž

$$b_1 = s_1(1, 0, \dots, 0, \lfloor N\beta_1 \rceil) + \dots + s_n(0, 0, \dots, 1, \lfloor N\beta_n \rceil) \doteq$$
$$\doteq (s_1, s_2, \dots, s_n, N(\beta_1 s_1 + \beta_2 s_2 + \dots + \beta_n s_n))$$

pro celá čísla $s_1, \dots, s_n$ (chyby způsobené zaokrouhlováním ve 2. řádku pomineme). Vzhledem k tomu, že je vektor b_1 krátký, nebudou čísla $s_1, \dots, s_n$ příliš velká a tedy číslo $\beta_1 s_1 + \beta_2 s_2 + \dots + \beta_n s_n$ bude poměrně malé (v nejlepším případě bude rovno nule).

Takto lze dojít k mnoha zajímavým vztahům, k nimž například patří i tzv. Machinův vzorec

$$-\pi + 16 \operatorname{arctg}\left(\frac{1}{5}\right) - 4arctg\left(\frac{1}{239}\right) = 0\,,$$

pro který stačí hodnota N okolo 10000. Dalším příkladem je hledání minimálních polynomů algebraických čísel. Pro dané reálné číslo γ položíme $\beta_i = \gamma^i$ a existuje-li takový polynom, tak v nejlepším případě najdeme celočíselný polynom, jehož je γ kořenem.

Pokud ovšem chceme hledat celočíselné závislosti, existují na to speciální algoritmy. Např. významným výsledkem algoritmu PSLQ bylo nalezení formule

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k}\left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6}\right),$$

díky které lze za použití dalších triků nalézt n-tou cifru π v šestnáctkové soustavě, aniž bychom znali předchozí.[12]

4.3 PŘÍKLADY NA APLIKACI LLL ALGORITMU

V této kapitole si pokusíme uvést několik příkladů, v nichž lze využít LLL algoritmu.

Příklad 4.3.1: Pokusíme se pomocí LLL algoritmu nalézt racionální aproximaci čísla $\pi \doteq 3{,}1416$.

Řešení: Vezmeme vektory $\vec{u}_1 = (1, 0, 3{,}146), \vec{u}_2 = (0, 1, 1000)$ a provedeme hledání redukované báze. Ta bude obsahovat vektor, který bude lineární kombinací těchto dvou vektorů s celočíselnými koeficienty a, b, tj. vektor ve tvaru $(a, b, 1000(a\pi + b))$. Očekáváme, že tento vektor bude mít „nevelkou“ velikost v a, b a tím spíše v $1000(a\pi + b)$.

Celý výpočet provedeme v programu Mathematica 8. Musíme vzít v úvahu, že tento programový balík vyžaduje při použití povelu LatticeReduce[13] jakožto souřadnice

[12] Vzorec objevil v roce 1995 Simon Plouffe. Byl pojmenován podle autorů dokumentu, ve kterém byl zveřejněn Davida H. Baileyho, Petera Borweina a Simona Plouffea, více viz Bailey-Borwein-Plouffe formula na Wikipedii.

[13] O tomto povelu je více v příloze.

vektorů jen racionální čísla. Budeme proto pracovat s racionální aproximací čísla π ve tvaru $\pi \doteq 3{,}1416$, kterou zapíšeme ve tvaru zlomku $31416/10000$:

```
pi =31416/10000
   3927/1250
```

a použijeme LLL algoritmus na vektory $\vec{u}_1, \vec{u}_2$, jak jsme rozmysleli výše. Dostaneme

```
LatticeReduce[{{1,0,1000pi},{0,1,1000}}]
{{-7,22,44/5},{15,-47,124}}
```

a teď si důkladně prohlédneme první vektor. Ten vlastně říká, že pokud vezmeme $a = -7$, $b = 22$, pak je rozdíl $1000(-7pi + 22)$ „nevelký“ a tedy $-7pi + 22$ je „téměř“ nula, neboli

$\pi \doteq \frac{22}{7}$ je „dobrá“ aproximace čísla π.

Z historie matematiky je známo, že tato racionální aproximace čísla π byla známa již Archimédovi (ten odvodil odhady $\frac{223}{71} < \pi < \frac{22}{7}$) a je ostatně i dnes využívána na ZŠ.

Příklad 4.3.2: Student řešil jakousi kvadratickou rovnici s celočíselnými koeficienty a nakonec si jeden její kořen vyčíslil na papírek. Vyšlo mu $x \doteq 3{,}26795$. Jenže původní zadání a výpočet ztratil. Dokážeme najít koeficienty té původní kvadratické rovnice?

Řešení: Označme $r = 3{,}26795$ přibližnou hodnotu hledaného kořene a směřujeme k nalezení vhodných celočíselných koeficientů a, b, c takových, že $ar^2 + br + c$ by mělo být „malé“. Vezměme vektory $\vec{u}_1 = (1, 0, 0, 10\,000r^2), \vec{u}_2 = (0, 1, 1, 10\,000r), \vec{u}_3 = (0, 0, 1, 10\,000)$ a použijme na tyto vektory LLL algoritmus. Očekáváme, že získáme vektor ve tvaru lineární kombinace těchto tří vektorů ve tvaru $a\vec{u}_1 + b\vec{u}_2 + c\vec{u}_3$, tj. ve tvaru $(a, b, c, 10\,000(ar^2 + br + c))$. Čekáme zároveň, že tento vektor bude „malý“ a zejména jeho poslední souřadnice by měla být „malá“.

V Mathematica 8 začněme racionální aproximací *r*:

```
r = 326795/10000
65359/2000
```

a nyní užijeme LLL algoritmus:

```
LatticeReduce[{{1,0,0,100000r^2},{0,1,0,100000r},
{0,0,1,100000}}]

{{1,-10,22,-(1119/4000)},{-29,81,45,-(587549/4000)},
{80,-229,-106,-(3869/50)}}
```

Teď nám koeficienty prvního vektoru určují kvadratickou rovnici $x^2 - 10x + 22 = 0$. Snadno zjistíme, že jeden její kořen je $x = 5 - \sqrt{3}$:

```
Solve[x^2-10x+22=0,x]

{{x→5-√3},{x→5+√3}}
```

a určíme si ještě přibližnou hodnotu:

```
N[5-√3,6]

3.26795
```

Může se říci, že k získání podobného výsledku je třeba mít štěstí a také umět trochu experimentovat (povšimněme si, že je třeba vhodně zvolit násobek N čísla r). My jsme tentokrát vzali $N = 100\,000$, jindy může pomoci jiná volba, zde se trochu přibližujeme experimentování běžnému spíše v přírodovědných oborech.

Příklad 4.3.3: Podobně jako u předchozího příkladu, můžeme hledat původní zadání kvadratické rovnice s celočíselnými koeficienty, jejíž kořen vyšel $x \doteq 0{,}645\,751$.

Řešení: Označme $s = 0{,}645\,751$ přibližnou hodnotu hledaného kořene a opět směřujeme k nalezení vhodných celočíselných koeficientů a, b, c, stejně jako u předchozího. Vezměme vektory $\vec{u}_1 = (1, 0, 0, 10\,000r^2), \vec{u}_2 = (0, 1, 1, 10\,000r), \vec{u}_3 = (0, 0, 1, 10\,000)$ a použijme na tyto vektory LLL algoritmus.

V Mathematica 8 začněme racionální aproximací r a dále použijeme LLL algoritmus a příkazy *Solve* pro zjištění kořenů rovnice a příkazem *N* pro zaokrouhlení ověříme správnost výpočtu:

```
In[8]:= s = 645751 / 1000000;

In[9]:= LatticeReduce[{{1, 0, 0, 10000000 s^2}, {0, 1, 0, 10000000 s}, {0, 0, 1, 10000000}}]

Out[9]= {{1, 4, -3, -1645999/100000}, {-284, -371, 358, -4384071/25000}, {-880, 799, -149, 218489/1250}}

In[13]:= Solve[y^2 + 4 y - 3 == 0, y]

Out[13]= {{y → -2 - Sqrt[7]}, {y → -2 + Sqrt[7]}}

In[15]:= N[-2 + Sqrt[7], 7]

Out[15]= 0.6457513
```

Příklad 4.3.4: Povzbuzeni předchozími příklady popíšeme postup na „výrobu“ reklamních příkladů, které jsou podobné předchozímu.

Naplánujeme si rozumné hodnoty kořenů kvadratické rovnice, řekněme

a) $x_1 = 7 - 2\sqrt{2} \doteq 4,17157$

b) $x_1 = 1 - \sqrt{6} \doteq -1,44949.$

Vzhledem k Viètovým vzorcům je jasné, že v případě a) je druhý kořen kvadratické rovnice s celočíselnými koeficienty nutně $7 + 2\sqrt{2}$ a tato rovnice se získá z rozkladu $\left(x - 7 + 2\sqrt{2}\right)\left(x - 7 - 2\sqrt{2}\right) = (x-7)^2 - 8 = x^2 - 14x + 41 = 0.$

Věc je tedy naprosto jasná, pokud známe exaktně zapsaný kořen kvadratické rovnice s celočíselnými koeficienty. Celá hra je o tom, že když známe jen nepřesnou, tedy již zaokrouhlenou hodnotu jednoho kořene, jako v daném případě a) $x_1 = 7 - 2\sqrt{2} \doteq 4,17157$, tak se přesto můžeme s pomocí počítače pokusit najít původní kvadratickou rovnici. Získat nám ji pomůže výše popsaný LLL algoritmus a samozřejmě můžeme užít i programy počítačové algebry. Budeme potřebovat vhodnou hodnotu čísla N a trochu štěstí, ale zato zkoušíme věci netradiční, které by přesto mohly být zajímavé i pro budoucí učitele.

V Mathematica 8 si uložíme vhodnou racionální aproximaci kořene x_1 jako r:

```
r = 417157/100000

417157/100000
```

a po nám již známém povelu

```
LatticeReduce[{{1,0,0,100000r^2},{0,1,0,100000r},
{0,0,1,100000}}]
```

dostaneme trojici vektorů

```
{{1,-14,41,162649/100000},{-20,76,31,-
(302649/5000)},{57,-221,-70,-(31829007/100000)}},
```

z nichž první obsahuje koeficienty námi hledané kvadratické rovnice.

V případě b) jen změníme zápis. Uložíme si hodnotu

```
r = -144949/100000
```

a nám již známý povel `LatticeReduce` poskytne výsledek

```
{{1,-2,-5,12601/100000},{-61,-76,18,-
(10568661/100000)},{-151,-183,52,24597249/100000}}.
```

Snadno se přesvědčíme, že kvadratická rovnice $x^2 - 2x - 5 = 0$ má kořen $x_1 = 1 - \sqrt{6} \doteq -1,44949$.

Z předchozího by mohl vzniknout dojem, že celý LLL algoritmus sice poskytuje jakousi „zábavu“, ale je vcelku neužitečný. Není tomu tak. LLL algoritmus patří mezi algoritmy hledající celočíselné vztahy mezi čísly (integer finding algorithm). Obecně jde o algoritmy, které hledají, zda by se k dané množině reálných čísel $\{x_1, x_2, \dots, x_n\}$ nedala najít taková množina celých čísel $\{a_1, a_2, \dots, a_n\}$ nikoli vesměs rovných nule taková, že $a_1x_1 + a_2x_2 + \cdots + a_nx_n = 0$.

Těchto algoritmů je dnes již známa celá řada. Některé vedly k objevu nebo k „znovuobjevení počítačem“ některých užitečných vztahů, které například umožnily výpočet čísla π na mnoho desetinných míst, obecně tedy nikoli jen ke „věcem na hraní“, ale i k relacím, které byly matematikům neznámé a vlastně byly „objevené za pomoci počítače“.

5 ZÁVĚR

Cílem mé práce bylo čtenáři představit nedostatečně zpracované téma LLL algoritmu v České republice. Jednalo se o záměr, uvést toto poměrně nové téma v českém prostředí, ilustrovat příklady výpočtů a demonstrovat jeho variabilní přínos a využití v matematické oblasti.

Vzhledem ke struktuře a obsáhlosti práce, která se zabývala LLL algoritmem a tématy s ním bezprostředně souvisejícími, byl tento cíl naplněn. Zpracovaná práce by měla českému čtenáři dostatečně představit téma LLL algoritmu a jeho přínos včetně praktického využití.

Ve své práci jsem se zabývala Gram-Schmidtovým ortogonalizačním procesem, který nám dokáže poskytnout bázi, která obsahuje jen ortogonální vektory. Další zkoumanou oblastí byly mřížky a jejich redukce. Stěžejní kapitolou pak byla část zabývající se LLL algoritmem, který dokáže v relativně krátkém čase najít poměrně krátkou bázi dané mřížky, ovšem na úkor totální ortogonality. Tyto vektory jsou „téměř“ ortogonální, zato vcelku krátké. Díky tomuto algoritmu, mohou matematici objevit nové věty a vzorce za pomoci počítače, o kterých dosud vůbec nevěděli, že existují. Další použití je ve faktorizaci (rozkladu) polynomů, kde by mohl pomoci právě LLL algoritmus. Toto téma jsem ovšem pouze zmínila, jelikož jeho přiblížení by bylo na další diplomovou práci. Více o těchto aplikacích lze nalézt např. v bakalářské práci *LLL algoritmus a jeho aplikace* od Forbelské.

Zpracování této diplomové práce znamenalo nesporný přínos pro moji osobu i pro potencionální čtenáře. Díky této práci jsem se seznámila s programem Geogebra, s jehož pomocí jsem vytvořila všechny ilustrační obrázky. Seznámení s tímto programem bylo velkým přínosem a domnívám se, že by s ním mohl pracovat každý, nebo každý učitel, který by program mohl využít jako pomůcku při vyučování pro názornou představu studentů.

Další program, který jsem zde využila je Mathematica. Mathematica dokáže provést LLL algoritmus v mnohem kratším čase, než jsem jej dokázala provést ručním výpočtem bez použití počítače. Tento program dokáže ušetřit mnoho času a lze jej využít právě při aplikaci LLL algoritmu na hledání celočíselného polynomu, pokud známe jeho

kořen. Mathematica lze mimo jiné využít také pro hledání ortonormální báze a při redukci mřížky, jak je ukázáno na některých příkladech.

Vzhledem k tomu, že tento algoritmus, u nás nebyl ještě mnohokrát popsán, bude přínosem pro čtenáře popis algoritmu, jeho využití i podrobně vypočtené příklady. Práce rovněž zahrnuje ukázku výpočtů v programu Mathematica. Hlavní přínos shledávám v deskripci a názorné demonstraci v Mathematice, jakým způsobem může nalézt čtenář ztracenou rovnici, pokud zná zaokrouhlený kořen této rovnice, což je určitou „hříčkou“, kterou mohou využít jak studenti, tak učitelé použitím jednoduchého příkazu LatticeReduce v tomto programu.

6 SEZNAM OBRÁZKŮ

7 SEZNAM LITERATURY

[1] STANOVSKÝ, D., BARTO, L.:*Počítačová algebra.*Praha:Matfyzpres, 2011.ISBN 978-80-7378-167-5

[2] DRÁBEK, J.: *Lineární algebra: Eukleidovský prostor* [online].Elektronické texty přednášek [cit. 2012-01-9]. Dostupné z: http://www.kmt.zcu.cz/subjects/la.html

[3] FORBELSKÁ, J.: *LLL algoritmus a jeho aplikace*. Brno, 2006. Dostupné z: http://is.muni.cz/th/98916/prif_b/thesis_click.pdf. Bakalářská práce. Masarykova univerzita v Brně.

[4] BEČVÁŘ, J.: *Lineární algebra.* Praha: Matfyzpres, 2010. ISBN 978-80-7378-135-4

[5] Harvey Mudd College Math Tutorial: *The Gram-Schmidt Algorithm.* Dostupné na http://www.math.hmc.edu/calculus/tutorials/gramschmidt/gramschmidt.pdf

[6] LENSTRA, A. K.; LENSTRA, H. W., Jr.; LOVÁSZ, L.: *Factoring polynomials with rational coefficients*. Mathematische Annalen 261: 515–534, 1982. Dostupné na https://openaccess.leidenuniv.nl/bitstream/handle/1887/3810/346_050.pdf?sequence=1

[7] REGEV, O.: *LLL Algorithm (Lattices in Computer Science: Lecture 5).* Učební text Tel Aviv University, 2004. Dostupný na http://www.cs.tau.ac.il/~odedr/teaching/lattices_fall_2004/ln/lll.pdf

[8] ZLATOŠ, P.: *Lineárna algebra a geometria.* Elektronický učební text FMFI UK, Bratislava, 2003. Dostupné na http://thales.doa.fmph.uniba.sk/zlatos/la/LAG_A4.pdf

[9]CASSELS, J. W. S.: *An introduction to the Geometry of Numbers.*Springer Classics in Mathematics, Springer-Verlag, 1997. ISBN 3-540-61788-4. Částečně dostupné na http://books.google.cz/books?id=FEb_4fo6T64C&pg=PA9&hl=cs&source=gbs_toc_r&cad=4#v=onepage&q&f=false

[10] Schmidt biography [online]. Scotland: School of Mathematics and Statistics, University of St. Andrews, c1996, December 1996 [cit. 2011-11-20]. Dostupné na http://www-history.mcs.st-andrews.ac.uk/Biographies/Schmidt.html

[11] Lovász biography [online]. Scotland: School of Mathematics and Statistics, University of St. Andrews, c1996, December 1996 [cit. 2011-11-20]. Dostupné na http://www-history.mcs.st-andrews.ac.uk/Biographies/Lovasz.html

[12] WOLFRAM RESEARCH, Inc. *Wolfram Mathematica 8*: *Documentation center* [online]. Oxfordshire, 2012 [cit. 2012-02-19]. Dostupné z: http://www.wolfram.com/company/contact.cgi

[13] PROSKURJAKOV, I. V.:*Sbornik zadač po linejnoj algebre.* Moskva: Nauka, 1970.

[14] BICAN, L.: *Lineární algebra.* Praha: SNTL, 1979.

8 RESUMÉ

This master thesis will be concerned with LLL algorithm. The target of the thesis is to introduce LLL algorithm to Czech readers and demonstrate contribution of algorithm in mathematical science.

My thesis is divided into 4 chapters. The first chapter deals with the Gram–Schmidt process. This is a method for orthonormalising a set of vectors in an inner product space, most commonly the Euclidean space $\mathbb{R}^n$.

In mathematics, the goal of lattice basis reduction is given an integer lattice basis as input, to find a basis with short, nearly orthogonal vectors. This is realized by using different algorithms, whose running time is usually at least exponential in the dimension of the lattice. The second chapter is just about lattices and their reduction.

In the third chapter, I finally defined the LLL algorithm, which can be found in polynomial time quite short based on the lattice. The fourth chapter includes application of LLL algorithm.

Each chapter involves amount of practical examples for better understanding, supplemented by calculations in the computer program Mathematica 8. Illustration images are created in the program GeoGebra.

9 PŘÍLOHY

9.1 FUNKCE ORTHOGONALIZE

Zde je popsaná funkce Orthogonalize, kterou jsme využili při hledání ortonormální báze (převzato z [12]).

Orthogonalize

`Orthogonalize[{v1, v2, ...}]`
gives an orthonormal basis found by orthogonalizing the vectors v_i.

`Orthogonalize[{e1, e2, ...}, f]`
gives a basis for the e_i orthonormal with respect to the inner product function f.

▶ MORE INFORMATION

▼ EXAMPLES OPEN ALL

▼ **Basic Examples** (1)

Find an orthonormal basis for two 3D vectors:

```
In[1]:= Orthogonalize[{{1, 0, 1}, {1, 1, 1}}]
```

$$\text{Out[1]}= \left\{\left\{\frac{1}{\sqrt{2}}, 0, \frac{1}{\sqrt{2}}\right\}, \{0, 1, 0\}\right\}$$

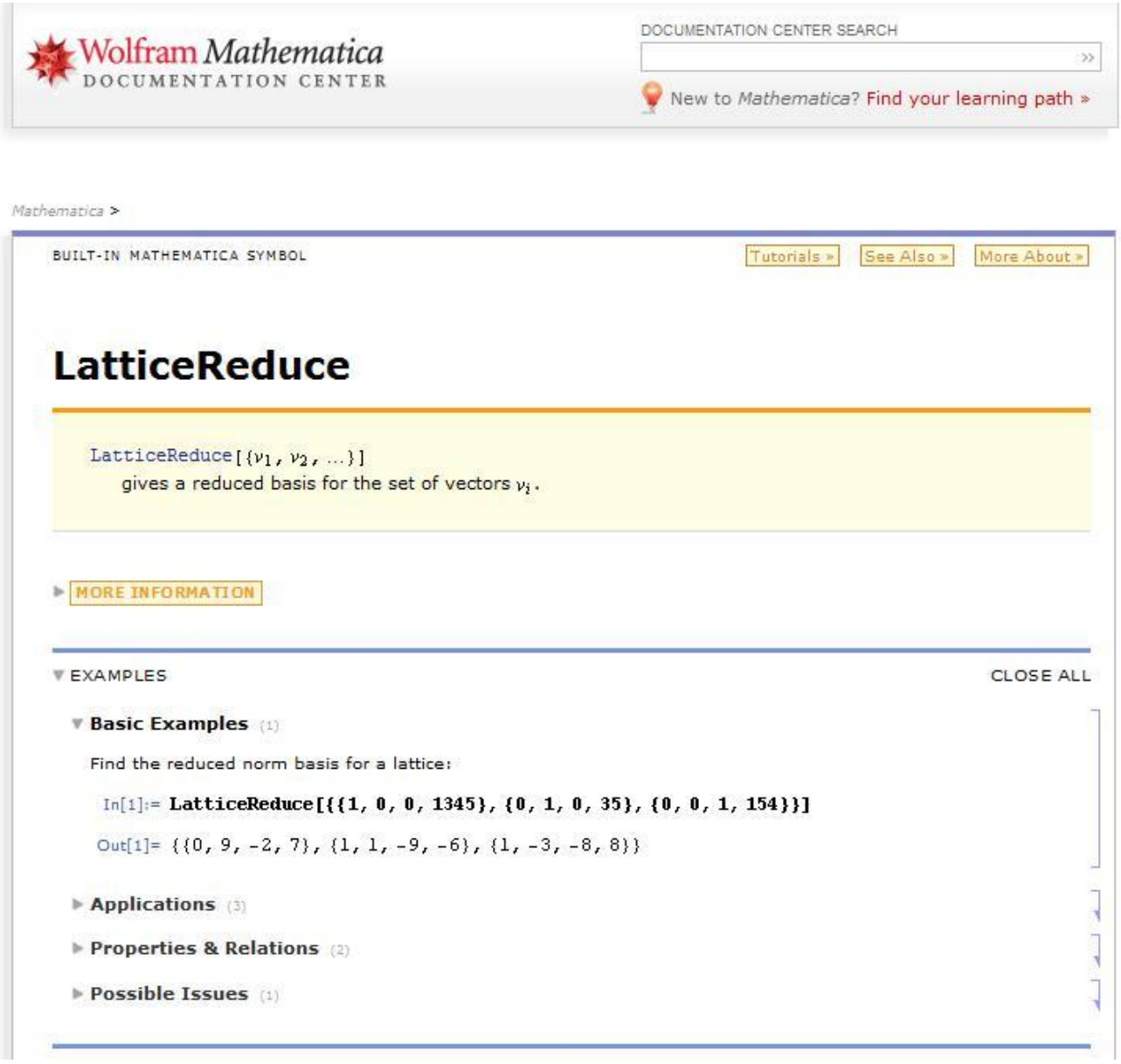
Find the coefficients of a general vector with respect to this basis:

```
In[2]:= %.{x, y, z}
```

$$\text{Out[2]}= \left\{\frac{x}{\sqrt{2}} + \frac{z}{\sqrt{2}}, y\right\}$$

9.2 FUNKCE LATTICEREDUCE

Zde máme popsánu funkci *LatticeReduce* v Mathematice a jednoduchý příklad na vysvětlení jejího použití (převzato z [12]).



Wolfram Mathematica
DOCUMENTATION CENTER

DOCUMENTATION CENTER SEARCH

New to *Mathematica*? Find your learning path »

Mathematica >

BUILT-IN MATHEMATICA SYMBOL

Tutorials » See Also » More About »

LatticeReduce

LatticeReduce[{v_1, v_2, ...}]
gives a reduced basis for the set of vectors v_i.

▶ MORE INFORMATION

▼ EXAMPLES CLOSE ALL

▼ **Basic Examples** (1)

Find the reduced norm basis for a lattice:

```
In[1]:= LatticeReduce[{{1, 0, 0, 1345}, {0, 1, 0, 35}, {0, 0, 1, 154}}]
Out[1]= {{0, 9, -2, 7}, {1, 1, -9, -6}, {1, -3, -8, 8}}
```

▶ **Applications** (3)

▶ **Properties & Relations** (2)

▶ **Possible Issues** (1)

9.3 PŘÍKLADY APLIKACE FUNKCE LATTICEREDUCE

Zde si ukážeme 3 příklady aplikace této funkce podle nápovědy v programu Mathematica (převzato z [12]).

Applications (3)

Starting with trivial integer linear relationships, LatticeReduce can produce more interesting ones:

In[1]:=
$$\underbrace{\begin{pmatrix} 1 & 0 & \cdots & 0 & -a_0 \\ 0 & 1 & \cdots & 0 & -a_1 \\ 0 & & \ddots & & \vdots \\ 0 & & & 1 & -a_n \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}}_{a} \cdot \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \\ 1 \end{pmatrix} == \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}$$

Find integer linear relationships for $a_1 = 2$ and $a_2 = 3$ of the form $x_0 + a_1 x_1 + a_2 x_2 = 0$:

```
In[2]:= a = {{1, 0, 0, -1}, {0, 1, 0, -2}, {0, 0, 1, -3}};

In[3]:= a.{1, 2, 3, 1}

Out[3]= {0, 0, 0}
```

LatticeReduce preserves linear relationships, and the third row provides $x_0 = -1$, $x_1 = -1$, and $x_2 = 1$:

```
In[4]:= b = LatticeReduce[a]

Out[4]= {{1, 0, 0, -1}, {-1, 1, 0, -1}, {-1, -1, 1, 0}}

In[5]:= b.{1, 2, 3, 1}

Out[5]= {0, 0, 0}
```

Find polynomial relationships $x_4 t^4 + x_3 t^3 + x_2 t^2 + x_1 t + x_0 = 0$ for $t = \sqrt[3]{3}$:

```
In[1]:= {a0, a1, a2, a3, a4} = Table[Round[10^7 Power[3, 1/3]^i], {i, 0, 4}]

Out[1]= {10 000 000, 14 422 496, 20 800 838, 30 000 000, 43 267 487}
```

The trivial initial relationships:

```
In[2]:= a = {{1, 0, 0, 0, -a0}, {0, 1, 0, 0, -a1},
            {0, 0, 1, 0, -a2}, {0, 0, 0, 1, -a3}, {0, 0, 0, 0, -a4}};
```

The reduced relationships:

```
In[3]:= b = LatticeReduce[a]

Out[3]= {{-3, 0, 0, 1, 0}, {0, -3, 0, 0, 1},
         {34, -12, -10, 98, -39}, {1, 50, -95, 4, 148}, {3, 26, 213, 9, 76}}
```

The first relationship:

```
In[4]:= b[[1]].t^Range[0,4]

Out[4]= -3 + t^3

In[5]:= % /. t -> Power[3, 1/3] // FullSimplify

Out[5]= 0
```

Find linear relationships $x_0+x_1\,\text{ArcTan}[1]+x_2\,\text{ArcTan}[1/5]+x_3\,\text{ArcTan}[1/239]==0$:

```
In[1]:= v = {1, ArcTan[1], ArcTan[1/5], ArcTan[1/239], 1};

In[2]:= {a0, a1, a2, a3, a4} = Round[10^20 v]

Out[2]= {100 000 000 000 000 000 000, 78 539 816 339 744 830 962,
  19 739 555 984 988 075 837, 418 407 600 207 472 386, 100 000 000 000 000 000 000}
```

Initial trivial relationships:

```
In[3]:= a = {{1, 0, 0, 0, -a0}, {0, 1, 0, 0, -a1}, {0, 0, 1, 0, -a2}, {0, 0, 0, 1, -a3}}

Out[3]= {{1, 0, 0, 0, -100 000 000 000 000 000 000}, {0, 1, 0, 0, -78 539 816 339 744 830 962},
  {0, 0, 1, 0, -19 739 555 984 988 075 837}, {0, 0, 0, 1, -418 407 600 207 472 386}}
```

Reduced relationships:

```
In[4]:= b = LatticeReduce[a]

Out[4]= {{0, 1, -4, 1, 0}, {-325 302, 315 725, 367 312, 1 153 518, 928 458},
  {-381 213, 314 234, 633 857, 2 221 192, -2 330 529},
  {-3 210 817, 4 041 574, 249 764, -3 042 512, 306 976}}
```

The first relationship:

```
In[5]:= b[[1]].v

Out[5]= Pi/4 + ArcTan[1/239] - 4 ArcTan[1/5]

In[6]:= N[%]

Out[6]= -1.11022 × 10^-16
```