**Technische Universiteit Delft**
**Faculteit Elektrotechniek, Wiskunde en Informatica**
**Delft Institute of Applied Mathematics**

# Het kleine Grothendieck probleem over de orthogonale groep

# (Engelse titel: The little Grothendieck problem over the orthogonal group)

Verslag ten behoeve van het
Delft Institute of Applied Mathematics
als onderdeel ter verkrijging

van de graad van

**BACHELOR OF SCIENCE**
**in**
**TECHNISCHE WISKUNDE**

**door**

**KISHAN KALICHARAN**

**Delft, Nederland**
**Juni 2014**

**BSc verslag TECHNISCHE WISKUNDE**

# "Het kleine Grothendieck probleem over de orthogonale groep"
**(Engelse titel: "The little Grothendieck problem over the orthogonal group")**


Kishan Kalicharan

**Technische Universiteit Delft**


**Begeleider**

D. de Laat

**Overige commissieleden**

Dr. D.C. Gijswijt

Dr.ir. M.C. Veraar

Dr. B. van den Dries

Juni, 2014 Delft

# Inhoudsopgave


| | | |
|---|---|---|
| 1. | Voorwoord | 2 |
| 2. | Abstract | 2 |
| 3. | Inleiding | 3 |
| 4. | Definities en notatie | 4 |
| 5. | Semidefiniet programmeren en rang beperking | 6 |
| 5.1. | Lineair programmeren | 8 |
| 5.2. | Semidefiniet programmeren | 9 |
| 5.3. | SDP algoritmen | 9 |
| 6. | Het maximale snede probleem | 9 |
| 7. | Het kleine Grothendieck probleem | 11 |
| 7.1. | Relaxatie | 11 |
| 7.2. | Algoritme | 12 |
| 7.3. | Kwaliteit | 12 |
| 7.4. | Het Grothendieck probleem | 12 |
| 7.5. | De reële Grothendieck ongelijkheid | 13 |
| 7.6. | De reële kleine Grothendieck ongelijkheid en integrality gap | 14 |
| 8. | Over de orthogonale groep | 15 |
| 8.1. | Relaxatie | 15 |
| 8.2. | Algoritme | 15 |
| 8.3. | Kwaliteit | 16 |
| 8.4. | Benaderingsconstante | 18 |
| 8.5. | Integrality gap | 19 |
| 9. | Toepassingen | 19 |
| 9.1. | Het gegeneraliseerde orthogonale Procrustesprobleem | 19 |
| 9.2. | Het globale registratieprobleem | 20 |
| 9.3. | ICP | 21 |
| 10. | Experimenten en resultaten | 22 |
| 10.1. | Experiment pizzabakkers | 22 |
| 10.2. | Experiment vierkanten | 23 |
| 10.3. | Experiment gitaar | 24 |
| Referenties | | 32 |
| Bijlagen | | 33 |

## 1. Voorwoord

Met dit onderzoek sluit ik de bachelorstudie Technische Wiskunde van de TU Delft af. In het het derde studiejaar van mijn bachelor kreeg ik in het laatste kwartaal de opdracht een bachelorscriptie te schrijven. Na onderzoek te hebben gedaan naar mogelijke onderwerpen, bleek dat mij vooral de optimalisering interesseerde. Al vrij snel kwam ik toen ik met David de Laat sprak terecht bij het onderwerp: “Het kleine Grothendieck probleem”. Dit onderwerp boeide mij vanwege de mogelijkheid om theoretisch te werken, maar toch ook toepassingen te bestuderen. Heel veel van het onderwerp was nog onduidelijk en dat zou ik later kunnen invullen. Wekelijks waren er vanuit de vakgroep presentaties van een aantal studenten dat in de optimalisering een scriptie aan het schrijven was. Hier kregen we de kans elkaar feedback geven en namen we tegelijkertijd kennis van de diversiteit binnen de wiskundige optimalisering. Om het onderzoek af te ronden heb ik op 1 juli 2014 voor de commissieleden een presentatie gehouden. Hiervoor heb ik gebruik gemaakt van de feedback die ik tijdens het derde kwartaal in het bachelorcolloquium heb gekregen en wat ik gedurende de onderzoeksperiode heb opgestoken.

## 2. Abstract

In de grafentheorie is het maximale snede probleem een bekend NP-hard probleem. Hiervoor is door Goemans en Williamson [GW95] een 0.878-benaderingsalgoritme gevonden gebaseerd op semidefiniet programmeren. Later zijn dezelfde technieken gebruikt om een benadering te vinden voor de oplossing van een algemener probleem genaamd het kleine Grothendieck probleem. In dit literatuuronderzoek worden deze problemen bestudeerd. Ook wordt er met name uitvoerig ingegaan op de uitbreiding: het kleine Grothendieck probleem over de orthogonale groep. Een van de overeenkomsten van deze problemen is dat er voor allemaal een benaderingsalgoritme is met een bepaalde benaderingsconstante. Deze algoritmen gebaseerd op het Goemans-Williamson algoritme zullen worden beschreven, zodat de overeenkomsten duidelijk worden. Het laatst genoemde probleem heeft toepassingen in de Procrustes analyse. De Procrustesproblemen in de wiskunde gaan over het vinden van orthogonale transformaties, zodat de afstand tussen de getransformeerde puntenwolken zo klein mogelijk is. Dit zou bijvoorbeeld gebruikt kunnen worden in de fotografie. Daar zal in deze scriptie een voorbeeld van worden bestudeerd en worden vergeleken met een andere methode gebaseerd op de methode van Schönemann [Sch66].

## 3. Inleiding

In deze scriptie staat het kleine Grothendieck probleem over de orthogonale groep centraal. Deze dankt zijn naam aan de wiskundige Alexander Grothendieck. Alexander Grothendieck is geboren in 1928 in Berlijn. Hij is later verhuisd naar Frankrijk waar hij een aantal papers heeft geschreven. Hiervan wordt "Produits tensoriels topologiques et espaces nucléaires" [Gro55] gezien als een artikel dat grote invloed heeft gehad op de functionaalanalyse. Daarin introduceerde en onderzocht hij de Grothendieck ongelijkheden. In de loop van de tijd zijn deze in verschillende vormen langsgekomen waarvan sommigen equivalent. De kleine grothendieck ongelijkheid, die overeenkomsten heeft met de Grothendieck ongelijkheid, is eenvoudiger te bewijzer dan de standaard Grothendieck ongelijkheid vandaar dat deze 'klein' genoemd wordt. In deze scriptie zullen problemen gerelateerd aan deze ongelijkheden behandeld worden. Naast de definities en theorie zal ook op een aantal toepassingen worden ingegaan.
De drie problemen die worden onderzocht zijn:
het kleine Grothendieck probleem met als toepassing maximale snedes in grafen,
het Grothendieck probleem met de correlatie clustering toepassing en
het kleine Grothendieck probleem over de orthogonale groep met de orthogonale Procrustes en globale registratie toepassingen.
De eerste twee genoemde problemen hebben te maken met een probleem in een graaf. De derde toepassing is vernoemd naar Procrustes. Hij komt uit de Griekse mythologie. Volgens de mythe had hij een bed met een bepaalde lengte. Hij bood voorbijgangers een slaapplaats aan. Hij vertelde dan dat iedereen perfect in zijn bed past. Volgens de mythe bond hij mensen op zijn bed, als ze te klein waren werden ze uitgestrekt en als ze te groot waren werden hun benen eraf gehakt zodat ze precies in het bed pasten. Het Procrustesprobleem in de wiskunde is minder gruwelijk. Het gaat over het vinden van een orthogonale transformatie, zodat het eerste object (puntenwolk) zo goed mogelijk naar het tweede object wordt getransformeerd. Een gegeneraliseerde variant van het Procrustesprobleem zal in dit verslag worden bekeken. Het verschil met het gewone Procrustesprobleem is dat er meerdere puntenwolken naar elkaar worden getransformeerd. Dit valt onder de statistische vormanalyse en kan eventueel in combinatie met andere theorie gebruikt worden bij het maken van twee- of drie dimensionale modellen. De laatste toepassing het globale registratieprobleem is een uitbreiding van het Procrustesprobleem waarin onder andere translaties zijn toegevoegd. [Pra05] [Ped13] [Ros04]

## 4. Definities en notatie

**Definitie 4.1.** *Een reële $n \times n$ matrix $A$ heet orthogonaal als $AA^T = A^T A = I$, met $^T$ de getransponeerde. Ofwel $A$ heeft orthonormale rijen en kolommen. Voor complexe matrices bestaat er een soortgelijke definitie. Een complexe $n \times n$ matrix $A$ heet unitair als $AA^* = A^* A = I$, met $^*$ de geadjungeerde.*

**Definitie 4.2.** *Een $n \times n$ matrix $A$ heet symmetrisch als $A = A^T$ en Hermitisch als $A = A^*$.*

**Definitie 4.3.** *Een $n \times n$ matrix $A$ heet een diagonaalmatrix als $A_{i,j} = 0$ voor alle $i \neq j$.
Een $m \times n$ matrix $A$ heet een pseudodiagonaalmatrix als $A_{i,j} = 0$ voor alle $i \neq j$.*

**Definitie 4.4.** *Een symmetrische matrix $X \in \mathbb{R}^{n \times n}$ noemen we positief semidefiniet (notatie $X \succeq 0$) als $x^T X x \geq 0$ voor alle $x \in \mathbb{R}^n$.
Een hermitische matrix $X \in \mathbb{C}^{n \times n}$ noemen we positief semidefiniet (notatie $X \succeq 0$) als $x^T X x \geq 0$ voor alle $x \in \mathbb{C}^n$*

De volgende drie stellingen worden bewezen in [Lau12].

**Stelling 4.5.** *Gegeven $X \in \mathbb{R}^{n \times n}$. $X \succeq 0$ dan en slechts dan als voor alle eigenwaarden $\lambda_i$ van $X$ geldt $\lambda_i \geq 0$.*

**Stelling 4.6.** *Gegeven $X \in \mathbb{R}^{n \times n}$. $X \succeq 0$ dan en slechts dan als er een $k \geq 1$ is zodanig dat $X = RR^T$ voor een $R \in \mathbb{R}^{n \times k}$. ($X \in \mathbb{C}^{n \times n}_{\succeq 0}$ dan en slechts dan als $X = RR^*$ voor een $R \in \mathbb{C}^{k \times n}$; voor een $k \geq 1$.) Dit noemen we de Cholesky decompositie van $X$.*

**Stelling 4.7.** *Gegeven $X \in \mathbb{R}^{n \times n}$. $X \succeq 0$ dan en slechts dan als $X = G(x_1, \ldots, x_n) = [\langle x_i, x_j \rangle]_{i,j}$ voor enkele $x_1, \ldots, x_n \in \mathbb{R}^k$; voor een $k \geq 1$. $G(x_1, \ldots, x_n)$ heet de Gram-matrix van $x_1, \ldots, x_n$ met het Euclidisch inwendig product. (Over $\mathbb{C}$ geldt deze stelling ook)*

**Definitie 4.8.** *Een verzameling $\mathcal{F}$ is een kegel als voor alle $A, B \in \mathcal{F}$ en $c \geq 0$ geldt dat $A + B \in \mathcal{F}$ en $cA \in \mathcal{F}$.*

**Definitie 4.9.** *Een verzameling $\mathcal{F}$ is convex als voor alle $A, B \in \mathcal{F}$, $\lambda \in [0, 1]$ geldt dat
$\lambda A + (1 - \lambda)B \in \mathcal{F}$.*

**Definitie 4.10.** *Laat $D$ een convexe verzameling zijn. De functie $f : D \to \mathbb{R}$ noemen we convex als voor alle $x, y \in D, \mu \in [0, 1]$ geldt dat $f(\mu x + (1 - \mu)y) \leq \mu f(x) + (1 - \mu)f(y)$.*

**Stelling 4.11.** *De verzameling positief semidefiniete matrices is een (convexe) kegel.*

*Bewijs.* Laat $A, B$ positief semidefiniete $n \times n$ matrices. Laat $c \geq 0$. Dan volgt uit de definitie eenvoudig

- $A + B$ is een $n \times n$ positief semidefiniete matrix,
- $cA$ is een $n \times n$ positief semidefiniete matrix.

□

De singulierewaardenontbinding, in het engels **s**ingular **v**alue **d**ecomposition, is een matrixontbinding met veel toepassingen in de wiskunde. In dit artikel gebruiken we de volgende definitie.

**Definitie 4.12.** *De singulierewaardenontbinding van een complexe $m \times n$ matrix $M$ is $M = U\Sigma V^*$ met $U$ een unitaire $m \times m$ matrix, $V$ een unitaire $n \times n$ matrix en $\Sigma$ een $m \times n$ pseudodiagonaalmatrix met niet-negatieve reële getallen op de diagonaal. Een reële matrix $M$ heeft* *singulierewaardenontbinding $M = U\Sigma V^T$, met $U$ een $m \times m$ orthogonale matrix, $V$ een $n \times n$ orthogonale matrix en $\Sigma$ een $m \times n$ pseudodiagonaalmatrix met niet-negatieve reële getallen op de diagonaal.*

De matrix $\Sigma$ is uniek als we bovendien de eis stellen dat $\Sigma_{1,1} \geq \Sigma_{2,2} \geq \ldots$. Deze waarden zijn de singuliere waarden van $\Sigma$. De matrix $\Sigma$ is een (pseudo)diagonaalmatrix en deze schaalt langs de assen met de getallen op de diagonaal. De matrices $U$ en $V$ kunnen gezien worden als rotatiematrices en $\Sigma$ als schalingsmatrix. In $\mathbb{R}^2$ geldt bijvoorbeeld voor een reële orthogonale matrix

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

dat

$$\begin{aligned} a^2 + b^2 = 1, &\quad a^2 + c^2 = 1, \\ c^2 + d^2 = 1, &\quad b^2 + d^2 = 1, \\ ac + bd = 0, &\quad ab + cd = 0. \end{aligned}$$

Zonder verlies van algemeenheid kunnen we vanwege de vergelijkingen nemen $a = \cos\theta$ en $c = \sin\theta$. Dan volgt ($b = \sin\theta$ en $d = -\cos\theta$) of ($b = -\sin\theta$ en $d = \cos\theta$). Hiermee krijgen we twee soorten reële orthogonale matrices. De matrix

$$\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

is de rotatiematrix over een hoek $\theta$ en deze heeft determinant 1. De matrix

$$\begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & -\cos\theta \end{pmatrix}$$

is de spiegeling over de lijn door de oorsprong met hoek $\frac{\theta}{2}$. Deze heeft determinant -1. In hogere dimensies zijn deze matrices samenstellingen van rotaties, permutaties van de assen en spiegelingen. Conform onze verwachtingen leveren orthogonale matrices in een belangrijke matrixnorm geen bijdrage aan de 'grootte' van een matrix. Voor die norm geldt namelijk $\|OA\| = \|A\| = \|AO\|$ voor alle orthogonale $n \times n$ matrices $O$ en $n \times n$ matrices $A$. Dit is te lezen verderop in dit hoofdstuk.

**Definitie 4.13.** *(spoor)*
*Zij $A$ een $n \times n$ matrix. Dan*

$$\operatorname{tr}(A) = \sum_{i=1}^{n} A_{i,i}$$

**Stelling 4.14.** *Zij $A$ en $B$ $n \times n$ matrices dan*

$$\operatorname{tr}(AB) = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{i,j}B_{j,i} = \sum_{j=1}^{n}\sum_{i=1}^{n} B_{j,i}A_{i,j} = \operatorname{tr}(BA)$$

**Stelling 4.15.** *Zij $A$ en $B$ $n \times n$ matrices dan*

$$\operatorname{tr}(A^T B) = \sum_{j=1}^{n}\sum_{i=1}^{n} A_{i,j}B_{i,j} = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{i,j}B_{i,j} = \operatorname{tr}(AB^T)$$

**Stelling 4.16.** *Zij $A$ een $n \times n$ matrix. Laat $\lambda_1, \ldots, \lambda_n$ de eigenwaarden van $A$. Dan*

$$\operatorname{tr}(A) = \sum_{i=1}^{n} \lambda_i$$

*Bewijs.* Het karakteristieke polynoom van $A$ is $p(A) = \det(A - \lambda I) = (-1)^n(\lambda^n - \mathrm{tr}(A)\lambda^{n-1} + \cdots + (-1)^n \det A)$. Alleen de term met $\lambda^{n-1}$ is van belang. Omdat $\det(A - \lambda I) = 0$ als $\lambda$ een eigenwaarde van $A$ is geldt ook $p(A) = (-1)^n \prod_{i=1}^n (\lambda - \lambda_i) = (-1)^n(\lambda^n - (\lambda_1 + \cdots + \lambda_n)\lambda^{n-1} + \cdots + (-1)^n \prod_{i=1}^n \lambda_i)$. Dus $\mathrm{tr}(A) = \sum_{i=1}^n \lambda_i$. □

Net als voor een vector is voor een matrix een inproduct gedefinieerd. Hier bewijzen dat het Frobenius inproduct een inproduct is. $\langle A, B\rangle_F = \mathrm{tr}(A^*B) = \sum_{i=1}^n \sum_{j=1}^n \overline{A_{i,j}} B_{i,j}$ over $\mathbb{C}$ is een inproduct, want:

- $\langle A, B\rangle_F = \sum_{i=1}^n \sum_{j=1}^n \overline{A_{i,j}} B_{i,j} = \sum_{i=1}^n \sum_{j=1}^n \overline{B_{i,j}\overline{A_{i,j}}} = \overline{\langle B, A\rangle}_F$
- $\langle aA, B\rangle_F = \sum_{i=1}^n \sum_{j=1}^n a\overline{A_{i,j}} B_{i,j} = a\sum_{i=1}^n \sum_{j=1}^n \overline{A_{i,j}} B_{i,j} = a\langle A, B\rangle_F$
- $\langle (A{+}C), B\rangle_F = \sum_{i=1}^n \sum_{j=1}^n \overline{(A_{i,j} + C_{i,j})} B_{i,j} = \sum_{i=1}^n \sum_{j=1}^n \overline{A_{i,j}} B_{i,j} + \sum_{i=1}^n \sum_{j=1}^n \overline{C_{i,j}} B_{i,j} = \langle A, B\rangle_F + \langle C, B\rangle_F$
- $\langle A, A\rangle_F = \sum_{i=1}^n \sum_{j=1}^n \overline{A_{i,j}} A_{i,j} \geq 0$
  $\langle A, A\rangle_F = \sum_{i=1}^n \sum_{j=1}^n \overline{A_{i,j}} A_{i,j} = 0 \Rightarrow A = 0$

Merk op beperkt tot $\mathbb{R}$ mogen we dus de getransponeerde nemen in plaats van de geadjungeerde en dan is het ook een inproduct.

Er zijn verschillende matrixnormen in gebruik om matrices met elkaar te vergelijken. De matrixnorm zou je kunnen zien als een uitbreiding van de vectornorm.

**Definitie 4.17.** *De matrixnorm $||\cdot||$ voldoet aan de volgende eisen: Zij $A, B \in \mathbb{K}^{m\times n}$*

- $||A|| \geq 0$
- $||A|| = 0$ *desda* $A = 0$
- $||\alpha A|| = |\alpha| \cdot ||A||$, *met* $\alpha \in \mathbb{K}$
- $||A + B|| \leq ||A|| + ||B||$

**Definitie 4.18.** *Een voorbeeld van een veelgebruikte matrixnorm is de Frobeniusnorm. De Frobeniusnorm van een $m\times n$ matrix $A$ is $||A||_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |A_{i,j}|^2} = \sqrt{\mathrm{tr}(A^*A)} = \sqrt{\mathrm{tr}(AA^*)}$.*

Dit is een matrixnorm omdat het wordt voortgebracht door een inproduct, $\|A\|_F = \sqrt{\langle A, A\rangle_F}$.

**Stelling 4.19.** *Zij $A$ matrix en $O_1, O_2$ unitaire matrices dan geldt*

$$\begin{aligned}
\|O_1AO_2\|_F &= \sqrt{\mathrm{tr}((O_1AO_2)^*O_1AO_2)} \\
&= \sqrt{\mathrm{tr}(O_2^*A^*O_1^*O_1AO_2)} \\
&= \sqrt{\mathrm{tr}(O_2^*A^*AO_2)} \\
&= \sqrt{\mathrm{tr}(O_2O_2^*A^*A)} \\
&= \sqrt{\mathrm{tr}(A^*A)} \\
&= \|A\|_F
\end{aligned}$$

## 5. Semidefiniet programmeren en rang beperking

**Lemma 5.1.** *Zij $A$ een $b \times c$ matrix, $B$ een $c \times d$ matrix. Dan volgt $\mathrm{rk}(AB) \leq \mathrm{rk}(A)$.*

*Bewijs.* $\mathrm{rk}(AB) = \dim\{ABx : x \in \mathbb{R}^d\} \leq \dim\{Ax : x \in \mathbb{R}^c\} = \mathrm{rk}(A)$. De $\leq$ volgt omdat $\{ABx : x \in \mathbb{R}^d\} \subseteq \{Ax : x \in \mathbb{R}^c\}$.

□

**Stelling 5.2.** *Gegeven $X \in \mathbb{R}^{n\times n}$ en $r \in \mathbb{N}$ vast. $X \succeq 0$ en $\mathrm{rk}(X) \leq r$ dan en slechts dan als $X = G(x_1, \ldots, x_n)$ voor enkele $x_1, \ldots, x_n \in \mathbb{R}^r$.*

*Bewijs.* Neem aan $X \succeq 0$ en $\mathrm{rk}(X) \leq r$. Vanwege Stelling 4.7 volgt dan $X = G(y_1, \dots, y_n)$ voor enkele $y_1, \dots, y_n \in \mathbb{R}^k$; voor een $k \geq 1$.

(i)Geval $k < r$
Definieer

$$x_i = \begin{pmatrix} y_1 \\ \vdots \\ y_k \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^r$$

voor $i = 1, \dots, n$. Dan $x_i^T x_j = y_i^T y_j$ voor alle $i, j \in \{1, \dots, n\}$. Dus $X = G(x_1, \dots, x_n)$ voor deze $x_1, \dots, x_n \in \mathbb{R}^r$.

(ii)Geval $k = r$
Dan volgt $X = G(y_1, \dots, y_n)$ voor deze $y_1, \dots, y_n \in \mathbb{R}^r$.

(iii)Geval $k > r$
Neem zonder verlies van algemeenheid aan $\mathrm{rk}(X) = r$. Zij $\{u_1, \dots, u_r\}$ een orthonormale basis voor $\mathrm{Col}(X)$. Zij $\{e_1, \dots, e_r\}$ de standaardbasis voor $\mathbb{R}^r$ (merk op dit is dus een orthonormale basis). Definieer de afbeelding $\phi : \mathbb{R}^k \to \mathbb{R}^r$ gegeven door $\phi(c_1 u_1 + \cdots + c_r u_r) = c_1 e_1 + \cdots + c_r e_r$ met $c_1, \dots, c_r \in \mathbb{R}$. Dan $\langle y_i, y_j \rangle = \sum_{i,j=1}^r a_i b_j \langle u_i, u_j \rangle = \sum_{i=1}^r a_i b_i \langle u_i, u_i \rangle = \sum_{i=1}^r a_i b_i = \sum_{i=1}^r a_i b_i \langle e_i, e_i \rangle = \sum_{i,j=1}^r a_i b_j \langle e_i, e_j \rangle = \langle \phi(y_i), \phi(y_j) \rangle$ voor alle $i, j \in \{1, \dots, n\}$. Hier is $y_i = \sum_{l=1}^r a_l u_l$ en $y_j = \sum_{l=1}^r b_l u_l$ en we hebben gebruikt dat de bases orthonormaal zijn en de bilineairiteit van het inwendig product. Dus volgt $X = G(\phi(y_1), \dots, \phi(y_n))$ voor deze $\phi(y_1), \dots, \phi(y_n) \in \mathbb{R}^r$

Voor de andere kant. Neem aan $X = G(x_1, \dots, x_n)$ voor enkele $x_1, \dots, x_n \in \mathbb{R}^r$. Wegens Stelling 4.7 volgt $X \succeq 0$. We schrijven $X = [x_1, \dots, x_n]^T [x_1, \dots, x_n]$. Dan met 5.1 volgt $\mathrm{rk}(X) \leq \mathrm{rk}([x_1, \dots, x_n]) \leq r$. □

**Gevolg 5.3.** *Gegeven $X \in \mathbb{R}^{n \times n}$. Dan geldt $X \succeq 0$ dan en slechts dan als $X = G(x_1, \dots, x_n)$ voor enkele $x_1, \dots, x_n \in \mathbb{R}^n$. Bovendien volgt dan $X = [x_1, \dots, x_n]^T [x_1, \dots, x_n]$. Merk op dit lijkt op Stelling 4.7.*

**Stelling 5.4.** *Gegeven $X \in \mathbb{R}^{dn \times dn}$. Dan geldt $X \succeq 0$ dan en slechts dan als $X = [X_i X_j^T]_{ij}$ met $X_i \in \mathbb{R}^{d \times dn}$ voor $i = 1, \dots, n$.*

*Bewijs.* Wegens Gevolg 5.3: $X \succeq 0$ dan en slechts dan als $X = G(x_1, \dots, x_{dn})$ voor enkele $x_1, \dots, x_{dn} \in \mathbb{R}^{dn}$. Dat kunnen we anders noteren

$$\begin{aligned} X &= [x_1, \dots, x_{dn}]^T [x_1, \dots, x_{dn}] \\ &= [[x_{d(i-1)+1}, \dots, x_{d(i-1)+d}]_i^T [x_{d(j-1)+1}, \dots, x_{d(j-1)+d}]_j]_{i,j} \\ &= [X_i X_j^T]_{i,j} \end{aligned}$$

, met $X_i = [x_{d(i-1)+1}, \dots, x_{d(i-1)+d}]_i^T$ voor $i = 1, \dots, n$. □

**Stelling 5.5.** *Gegeven $X \in \mathbb{R}^{dn \times dn}$. Dan geldt $X \succeq 0$ dan en slechts dan als er bestaat een $k \in \mathbb{N}$ zó dat $X = [X_i X_j^T]_{i,j}$ met $X_i \in \mathbb{R}^{d \times k}$ voor $i = 1, \dots, n$.*

*Bewijs.* Het bewijs is hetzelfde als dat van Stelling 5.4 alleen toegepast op Stelling 4.7 in plaats van Gevolg 5.3. □

**Stelling 5.6.** *Gegeven $X \in \mathbb{R}^{dn \times dn}$ en $k \in \mathbb{N}$ vast. Er geldt $X \succeq 0$ en $\text{rk}(X) \leq k$ dan en slechts dan als $X = [X_i X_j^T]_{i,j}$ met $X_i \in \mathbb{R}^{d \times k}$ voor $i = 1, \dots, n$.*

*Bewijs.* Het bewijs is hetzelfde als dat van Stelling 5.4 alleen toegepast op Stelling 5.2 in plaats van Gevolg 5.3. □

5.1. **Lineair programmeren.** Verderop in dit hoofdstuk zal geïntroduceerd worden wat semidefiniete programma's zijn, maar daarvoor kijken we eerst naar wat voor velen in de optimalisering bekend is: de lineaire programma's.

Een lineair programma (LP) is een probleem van de vorm

$$\begin{aligned} &\max c^T x \\ s.t. \qquad &a_i^T x = b_i, \quad i = 1, \dots, m \\ &x \in \mathbb{R}^n_{\geq 0} \end{aligned}$$

$c^T x$ is een lineaire functie. De voorwaarden $a_i^T x \leq b_i$, ook wel geschreven als $Ax \leq b$, zijn de (lineaire) restricties. Ieder van deze restricties creëert een hypervlak $\{x \in \mathbb{R}^n : a_i^T x = b_i\}$. De voorwaarde $x \geq 0$ geeft de volgende halfruimte $\{x \in \mathbb{R}^n : x_j \geq 0 \text{ voor } j = 1, \dots, n\}$. Dit is duidelijk een kegel. De doorsnede van die halfruimten en hypervlakken geeft een convexe polyhedron. Dat is het toegelaten gebied waarover we de lineaire functie optimaliseren. Dit probleem heeft veel toepassingen. Eerst was onbekend of dit probleem in polynomiale tijd op te lossen is; totdat werd bewezen dat het algoritme, wat de ellipsoïde methode wordt genoemd, een LP exact in polynomiale tijd kan oplossen. Theoretisch is dit algoritme sterk, maar in de praktijk wordt deze zelden gebruikt omdat hij heel traag is.

Zij $F = \{x \in \mathbb{R}^n : Cx \leq d\}$ een begrensde verzameling met bovendien de eigenschap dat zijn elementen $\mathbb{R}^n$ opspannen. Een ellipsoïde is een eventueel hoger dimensionale ellips. Het idee van het ellipsoïdealgoritme is dat er een ellipsoïde (bol) bekend is die groot genoeg is, zodat deze $F$ bevat. Vervolgens wordt nagegaan of het centrum van de ellipsoïde in het toegelaten gebied zit. Zo niet, dan schendt deze minstens een van de ongelijkheden. Vanwege de convexiteit kunnen we zeggen dat dan het toegelaten gebied ligt in de doorsnede van de halfruimte behorende bij de ongelijkheid en de ellipsoïde. Dan wordt de ellipsoïde in tweeën gedeeld met het hypervlak evenwijdig aan het hypervlak dat bij de halfruimte hoort ($\{x \in \mathbb{R}^n : a_i^T x = b_i\}$). Daarna wordt de kleinste ellipsoïde om de kant van de ellipsoïde waarin het toegelaten gebied ligt genomen. Dan herhalen we dit vanaf de stap met het centrum nemen etc. Er wordt ook in de lus een stopcriterium opgenomen voor als er geen toegelaten oplossing is. Uit dit algoritme komt dus een element van $F$.

Een manier waarop dit gebruikt kan worden om een LP op te lossen is het feit dat een LP equivalent is aan.

$$\begin{aligned} A^T x &\leq b, \\ -x &\leq 0, \\ -Ay &\leq -c, \\ -y &\leq 0, \\ -c^T x + b^T y &\leq 0. \end{aligned}$$

De derde en vierde ongelijkheid beschrijven de restricties van de duale LP en de vijfde hangt samen met de sterke dualiteitsstelling. Merk op als het primale probleem onbegrensd is, dan is de duale niet toegelaten en visa versa. Als gevolg is een groot nadeel hiervan dat er geen onderscheid wordt gemaakt tussen onbegrensde en niet toegelaten LP's. Dit kan worden verholpen door daarna ook met deze methode te kijken of er überhaupt een toegelaten oplossing is voor de primale LP. Als er zo'n oplossing is dan is het probleem onbegrensd, als er niet zo'n oplossing is dan is het probleem niet toegelaten. [Reb07]

5.2. **Semidefiniet programmeren.** Een (reëel) semidefiniet programma (SDP) heeft de vorm

$$\max \langle C, X\rangle : X \in \mathbb{R}^{n\times n}_{\succeq 0}, \langle A_k, X\rangle = b_k \text{ voor } k = 1, \dots, m$$

met $\langle C, X\rangle_F = \operatorname{tr}(C^T X) = \sum_{i,j=1}^n C_{i,j}X_{i,j}$ en $A_1, \dots, A_m, C$ reële symetrische $n \times n$ matrices en $b_1, \dots, b_m \in \mathbb{R}^n$.

Een complex semidefiniet programma ziet er hetzelfde uit alleen over $\mathbb{C}$ in plaats van $\mathbb{R}$ en met de geadjungeerde in plaats van getransponeerde bij het inproduct. Als we bovendien de restrictie opleggen dat $X$ een diagonaalmatrix moet zijn, wordt het semidefiniete programma een lineair programma met $c_i = C_{i,i}, a_i = \operatorname{diag}(A_i)$ en $b_i = b_i$. In semidefiniet programmeren optimaliseer je een lineaire functie over de doorsnede van de kegel van positief semidefiniete matrices (Stelling 4.11) en de halfruimten en hypervlakken van de verdere restricties. [Fre04]

Een SDP van beperkte rang is een semidefiniet programma met een beperking op de rang. Deze programma's zijn in het algemeen niet in polynomiale tijd op te lossen en de verzameling waarover geoptimaliseerd wordt dan niet meer convex is. Dus dan kunnen we bepaalde algoritmen niet meer gebruiken.

5.3. **SDP algoritmen.** Semidefiniet programmeren valt, net als lineair programmeren, onder convex programmeren vanwege Stelling 4.11. Een van de populairste soorten algoritmen die wordt gebruikt om de oplossing van een SDP te benaderen, is de inwendig punt methode. Zoals de naam suggereert speurt deze het inwendige van het toegelaten gebied af. Hier zijn veel varianten van. Deze werken voor veel convexe problemen zo ook voor lineair programmeren. Het is mogelijk om de oplossing van een **begrensde** SDP met een inwendig punt methode binnen een bepaalde error $\epsilon \in \mathbb{Q}$ in polynomiale tijd te benaderen. Een voorbeeld van een inwendig punt methode is Alizadeh's interior point method [Ali95]. Deze kan binnen iedere error $\epsilon$ relatief snel een benadering voor de oplossing van een SDP vinden. Het eerste algoritme voor SDP was de ellipsoïdemethode, zoals ook besproken voor lineair programmeren. De ellipsoïdemethode kan in polynomiale tijd (in $\log(\frac{1}{\epsilon})$ en de programmagrootte) een $1+\epsilon$-benadering voor de optimale oplossing van een SDP vinden. De programmagrootte is het aantal bits in de binaire representatie van $A_i, b_i$ en $C$ voor alle $i, j$. In de praktijk blijkt deze vrij langzaam, daarom wordt er meestal gebruik gemaakt van snellere algoritmen zoals de inwendig punt methode. [VB96]

## 6. Het maximale snede probleem

Goemans en Williamson kwamen in 1995 met een revolutionair artikel [GW95] over het maximale snede probleem in grafen. In een graaf $G = (V, E)$ zonder gewichten op de lijnen is een snede $\delta(U) = E[U, V \setminus U]$ met $U \subseteq V$. Informeel is dit de verzameling lijnen die tussen twee verzamelingen punten $U, W \subseteq V$ lopen waarvoor geldt dat $U \cup W = V$ en $U \cap W = \emptyset$. Dan gaat het om het maximaliseren van het aantal lijnen tussen die verzamelingen dus $\max\{|\delta(U)| : U \subseteq V\}$. Punt $x_i = 1$ als $v_i$ in de ene puntenverzameling $U$ in de snede zit en als hij in de andere puntenverzameling $V \setminus U$ zit dan $x_i = -1$. Dan volgt dat het maximale snede probleem equivalent is aan

$$\max_{x_i \in \{\pm 1\}} \sum_{i=1}^{n} \sum_{j=1}^{n} C_{i,j} x_i x_j \tag{1}$$

(de multiplicatieve factor 1/4 is weggelaten) hier is matrix $C$ de Laplacematrix van de graaf.

**Definitie 6.1.** ***(Laplacematrix)*** *Laat $G = (V, E)$ een graaf zijn. Laat de punten uit de graaf $v_1, v_2, \ldots, v_{|V|}$ zijn. Dan noemen we $L \in \mathbb{R}^{|V|\times|V|}$ de Laplacematrix van $G$ als*

$$L_{i,j} = \begin{cases} d(v_i) & \textit{als } i = j, \\ -1 & \textit{als } i \neq j \textit{ en } \{v_i, v_j\} \in E, \\ 0 & \textit{als } i \neq j \textit{ en } \{v_i, v_j\} \notin E. \end{cases}$$

Er bestaan ook andere matrices waarin informatie over een graaf zit. Zoals deze soort incidentiematrix. Laat $e_1, e_2, \ldots, e_{|E|}$ de lijnen zijn in $G$. We noemen een $|E| \times |V|$ matrix $M$ de incidentiematrix van $G$ als

$$M_{i,j} = \begin{cases} 0 & \text{als } v_j \notin e_i, \\ -1 & \text{als } v_j \in e_i \text{ en } v_1, v_2, \ldots, v_{j-1} \notin e_i, \\ 1 & \text{als } v_j \in e_i \text{ en } v_{j+1}, v_{j+2}, \ldots, v_{|V|} \notin e_i. \end{cases}$$

Merk op dat dan $L = M^T M$. Dat wil dus zeggen dat $L$ positief semidefiniet is.

Duidelijk is als in (1) voor alle $x_i = 1$ geldt dan komt er nul uit, omdat voor iedere lijn twee keer $-1$ in de matrix staat en er twee keer $+1$ op de diagonaal staat. Hetzelfde als voor alle $x_i = -1$ geldt. Dit klopt, want als alle punten in 1 van de twee puntenverzamelingen van de snede zitten dan lopen er geen lijnen tussen de twee puntenverzamelingen omdat de andere puntenverzameling leeg is. Als we dan vervolgens een van de punten $x_k$ in de andere puntenverzameling in de snede zetten, dan geldt voor alle lijnen $\{x_i, x_k\}$ waar $x_k$ een eindpunt van is; $C_{i,k} x_i x_k$ is $+1$ terwijl het eerst $-1$ was omdat $x_i$ in een andere puntenverzameling zit dan $x_k$. Voor $C_{k,i} x_k x_i$ gebeurt dat ook, dus er is voor elk zo'n lijn $+4$ bijgekomen en dit compenseren we met die factor $\frac{1}{4}$. Goemans en Williamson bedachten een algoritme om een benadering voor de maximale snede in een graaf te vinden. In hun artikel is het probleem overigens wel anders genoteerd. Ze kwamen met een gerandomiseerd algoritme, waar een toekenning voor de vector $x$ uitkwam waarvan de verwachte verkregen oplossing minstens $0.878\ldots$ van de optimale oplossing was. In het algoritme worden de volgen stappen uitgevoerd:

- Bekijk eerst de relaxatie
$$\max\{\sum_{i=1}^{n} \sum_{j=1}^{n} C_{i,j} \langle y_i, y_j \rangle : y_i \in \mathbb{R}^n \text{ met } \|y_i\|_2 = 1\}$$
- Los de relaxatie op en vind ook de vectoren $y_i$ bij de oplossing.
- Kies een willekeurige vector $r$ op de eenheidsbol in $\mathbb{R}^n$ volgens de rotatie invariante kansverdeling op die eenheidsbol.
- Definieer $x_i = \text{sign}(\langle y_i, r \rangle)$.
- Controleer of de waarde van (1) bij de oplossing van de vorige stap minstens $0.878\ldots$ van de oplossing van de relaxatie is. Als dat niet zo is herhaal vanaf stap 3, anders stop.

Er valt op te merken dat iedere oplossing voor (1) ook een oplossing voor de relaxatie is. We kunnen namelijk

$$y_i = \begin{pmatrix} x_i \\ 0 \\ \vdots \\ 0 \end{pmatrix} \tag{2}$$

definiëren en dat geeft dezelfde waarde in de relaxatie. Dus het maximum van de relaxatie is minstens zo hoog als het maximum van (1). De relaxatie is een semidefiniet programma (hierover later meer) en daardoor makkelijk op te lossen. In de derde en vierde stap wordt er eigenlijk een snede voor (1) gedefiniëerd door de bol in tweeën te snijden en te kijken welke vectoren aan de ene kant liggen en welke aan de andere kant. Het mooiste uit het algoritme is dat door dit

toe te passen een oplossing gecreëerd wordt waarvan de verwachtingswaarde minstens $0.878\ldots$ van het optimum van de relaxatie is en daarmee dus ook minstens $0.878\ldots$ van het optimum van (1).
Het artikel van Goemans en Williamson heeft ervoor gezorgd dat de interesse voor semidefiniet programmeren een enorme boost kreeg. Hun methode werkt ook voor grafen met gewichten op de lijnen, maar in plaats van dat we dan te maken hebben met de Laplacematrix $C$, is $C$ een matrix met op de diagonaal

$$C_{i,i} = \sum_{e \in E_{v_i}} w(e)$$

hier is $w : E \to \mathbb{N}$ de gewichtsfunctie en $E_{v_i} = \{e \in E : v_i \in e\}$. Voor de overige indices geldt $C_{i,j} = -w(\{v_i, v_j\})$. Voor een graaf zonder gewichten is dit de Laplacematrix. In paragraaf 10.1 wordt een experiment op basis van het maximale snede probleem uitgevoerd.

## 7. Het kleine Grothendieck probleem

Het kleine Grothendieck probleem in de combinatorische optimalisering is

$$\max_{x_i \in \{\pm 1\}} \sum_{i=1}^{n} \sum_{j=1}^{n} C_{i,j} x_i x_j, \tag{3}$$

hier is $C$ een $n \times n$ positief semidefiniete reële matrix. Dit probleem is NP-hard. Het maximale snede probleem is een toepassing van het kleine Grothendieck probleem, omdat de Laplacematrix van een graaf altijd positief semidefiniet is; zie hoofdstuk 6.

7.1. **Relaxatie.** In deze paragraaf leiden we af hoe we aan een relaxatie komen om daarmee conclusies te kunnen trekken over het kleine Grothendieck probleem. Laat $n$ een positief geheel getal zijn en zij $C \in \mathbb{R}^{n \times n}_{\succeq 0}$. Het kleine Grothendieck probleem wordt gegeven door

$$\max \sum_{i,j=1}^{n} C_{i,j} x_i x_j : x_i \in \{\pm 1\} \text{ voor } i = 1, \ldots, n. \tag{4}$$

Het blijkt handig om dit op te lossen om de voorwaarde $x \in \{\pm 1\}$ als $x^2 = 1$ te schrijven. Nu creëren we de volgende relaxatie door in plaats van 1 dimensionale variabelen, vectoren te nemen. Belangrijk is dat het bovenstaande probleem een speciaal geval is van de relaxatie door $m = 1$ te nemen. Ook als we in plaats van $\mathbb{R}^m$ een 1-dimensionale deelruimte van $\mathbb{R}^m$ nemen staat er weer het oorspronkelijke probleem. De relaxatie is

$$\sup_{m \in \mathbb{N}} \max \sum_{i,j=1}^{n} C_{i,j} X_i^T X_j : X_i \in \mathbb{R}^m; X_i^T X_i = 1 \text{ voor } i = 1, \ldots, n. \tag{5}$$

Om van (5) naar (6) gebruiken we $\mathbf{X}_{i,j} = X_i^T X_j$. (6) en (7) zijn equivalent (Stelling 4.7), we mogen in (6) zelfs m=n nemen zie Gevolg 5.3. Nu is de relaxatie geformuleerd als

$$\sup_{m \in \mathbb{N}} \max \sum_{i,j=1}^{n} C_{i,j} \mathbf{X}_{ij} : \mathbf{X} = G(x_1, .., x_n), x_i \in \mathbb{R}^m, x_i^T x_i = 1 \text{ voor } i = 1, .., n. \tag{6}$$

Tot slot passen we Stelling 4.7 toe om een semidefiniet programma te maken:

$$\max \sum_{i,j=1}^{n} C_{i,j} \mathbf{X}_{i,j} : \mathbf{X} \in \mathbb{R}^{n \times n}_{\succeq 0}, \mathbf{X}_{ii} = 1 \text{ voor } i = 1, \ldots, n. \tag{7}$$

Merk op het maximum bij (5) ligt hoger dan dat van het oorspronkelijke probleem, omdat de optimale oplossing bij (4) ook bij (5) is toegelaten. De reden dat we dit probleem hebben geconstrueerd is, omdat dit makkelijker op te lossen is en informatie geeft over (5).

7.2. **Algoritme.** Het algoritme van Goemans en Williamson geeft bij het maximale snede probleem een oplossing waarvan de verwachtingswaarde ten minste 0.878 van het optimum van de SDP is. Hetzelfde algoritme meestal genoteerd als

- Los SDP (7) op en vindt matrix $X$.
- Pas een cholesky decompositie op $X$ toe om alle $X_i$ in 5 te vinden.
- Kies een willekeurige vector $r$ met standaard normaal verdeelde entries.
- Definieer $x_i = \text{sign}(\langle X_i, r\rangle)$.
- Controleer of de waarde van (1) bij de oplossing van de vorige stap minstens $\frac{2}{\pi}$ van de oplossing van de SDP is. Als dat niet zo is herhaal vanaf stap 3, anders stop.

geeft bij het kleine Grothendieck probleem een $\frac{2}{\pi}$-benadering. Dit is minder hoog dan bij het maximale snede probleem. Er zouden in theorie gevallen kunnen zijn waarin het algoritme nooit oplossingen beter dan $\frac{2}{\pi}$ van het optimum geeft.

7.3. **Kwaliteit.**

**Lemma 7.1.** *Laat $D$ een convexe verzameling zijn en $f$ een convexe functie op $D$, dan* $\sup\{f(x) : x \in D\} = \sup\{f(x) : x \in \text{ext}(D)\}$. *Hier is* $\text{ext}(D)$ *de verzameling extreme punten van $D$.*

Voor kleine Grothendieck in het reële geval geldt: $f^{opt}(C) \geq \frac{2}{\pi} \cdot SDP^{opt}(C)$, met $f^{opt}$ de optimale oplossing voor de reële kleine Grothendieck en $SDP^{opt}$ de optimale oplossing voor de semidefiniete relaxatie. Dit bewees Nesterov [Nes97] als een van de eersten. Hier gebruikte hij dat in dit geval in plaats van $x_i, x_j \in \{-1, 1\}$ ook $x_i, x_j \in [-1, 1]$ gebruikt mag worden omdat het probleem dan toch convex is. Dit geeft informatie over hoe dicht de oplossing van de relaxatie bij dat van het oorspronkelijke probleem moet liggen. Dit ligt namelijk altijd in het interval $[\frac{2}{\pi} SDP^{opt}(C), SDP^{opt}(C)]$. Later is bekend geworden dat $\frac{2}{\pi}$ scherp is [Rie74] voor deze relaxatie, ofwel de integrality gap is $\frac{\pi}{2}$. In [SZY07] wordt met behulp van een benaderingsalgoritme de oplossing van het complexe kleine Grothendieck probleem benaderd. Dit gebruikt hetzelfde idee voor de relaxatie als Goemans en Williamson voor maxcut. Namelijk in plaats van 1 dimensionele variabelen met modulus 1 nemen ze in de relaxatie vectoren van norm 1. Het is een $\frac{\pi}{4}$-benaderingsalgoritme dat ze gebruiken en het lijkt erg op het algoritme voor maxcut van Goemans en Williamson. Ook wordt aan de hand van het algoritme bewezen dat de integrality gap precies $\frac{4}{\pi}$ is. Ze vinden namelijk problemen waarvan het optimum willekeurig dicht bij $\frac{\pi}{4} \cdot SDP^{opt}$ ligt.

Er zou in theorie voor dit kleine Grothendieck probleem een andere relaxatie kunnen zijn met een betere integrality gap. De relaxatie van Shor die gebruikt maakt van de Lagrangiaan is even goed. [Nes97]

7.4. **Het Grothendieck probleem.** Eerder namen we aan $C \succeq 0$. We kunnen ook iets zeggen over de oplossing van het Grothendieck probleem met willekeurige $n \times n$ matrices $C$.

$$\max_{x_i \in \{\pm 1\}} \sum_{i=1}^{n} \sum_{j=1}^{n} C_{i,j} x_i x_j \tag{8}$$

Er is een $(\frac{1}{4\log(n)} - \frac{8}{n})$-benaderingsalgoritme bekend voor dit soort problemen. Ook hier gebruiken we weer de relaxatie

$$\max \sum_{i,j=1}^{n} C_{i,j} X_i^T X_j : X_i \in \mathbb{R}^n; X_i^T X_i = 1 \text{ voor } i = 1, \ldots, n \tag{9}$$

Het algoritme heeft de volgende stappen.

- Los SDP (9) op en vind vectoren $X_i$.
- Pak een willekeurige vector $r$ van lengte $n$ met standaard normaal verdeelde entries.
- Definieer $z_i = \frac{\langle X_i, r\rangle}{\sqrt{4 \log n}}$.
- Als $|z_i| > 0$, dan $y_i = z_i/|z_i|$, anders $y_i = z_i$.
- Laat $x_i = \begin{cases} -1, & \text{met kans } \frac{1-y_i}{2} \\ +1, & \text{met kans } \frac{1+y_i}{2} \end{cases}$ voor alle $i$ de toekenning van de variabelen zijn.

Dit probleem heeft bredere toepassingen dan het kleine Grothendieck probleem. Een van de toepassingen is correlatie clusteren.
Zij $G = (V, E)$ een graaf met $n$ punten. Vervolgens geven we iedere lijn $\{u, v\} \in E$ een label '+' als $u$ en $v$ gelijksoortig zijn en '-' als ze niet gelijksoortig zijn. Het doel is het verdelen van $V$ in $V_1, V_2, \ldots, V_k$ disjunct, zó dat de clusters (geïnduceerde deelgrafen) $G[V_1], G[V_2], \ldots, G[V_k]$ zo veel mogelijk lijnen met label '+' bevatten en er zo veel mogelijk lijnen die tussen de clusters lopen, label '-' hebben. In plaats van dit te maximaliseren kan ook het aantal lijnen met label '-' dat zich in de clusters bevindt en het aantal lijnen met label '+' dat zich tussen de clusters bevindt geminimaliseerd worden. Hoewel deze problemen equivalent zijn, is het vinden van een benadering voor het optimum bij beiden anders. Het correlatie clusteren probleem ($corr$) is het verschil van deze problemen. In het onderstaande algoritme wordt het Grothendieck probleem gebruikt om voor het correlatie clusteren probleem een $\frac{\alpha}{\alpha+2}$-benadering te construeren. Hier is $\alpha$ de benaderingsconstante van het algoritme voor het Grothendieck probleem eerder deze paragraaf.

- Construeer matrix $A$ als volgt:
  als $i < j$ en $i$ en $j$ zijn gelijksoortig dan $A_{i,j} = 1$,
  als $i < j$ en $i$ en $j$ zijn niet gelijksoortig dan $A_{i,j} = -1$, anders 0.
- Voer het algoritme voor het Grothendieck probleem met $A$ uit en vindt variabelen $x_i$.
- Verdeel de graaf in twee clusters als $x_i = -1$ dan zit $v_i$ in cluster 1 en als $x_i = 1$ dan zit $v_i$ in cluster 2.
- Bereken de waarde $corr_{\mathcal{K}}(A)$ bij deze clustering $\mathcal{K}$.
- Bereken de waarde $corr_{\mathcal{K}_1}(A)$ van de clustering $\mathcal{K}_1$, dat is de clustering verkregen door de graaf in $n$ singleton clusters op te delen.
- Dan is de benadering voor de optimale oplossing van het correlatie clusteren probleem $\max\{corr_{\mathcal{K}}(A), corr_{\mathcal{K}_1}(A)\}$

[CW04]

7.5. **De reële Grothendieck ongelijkheid.** Sinds Alexander Grothendieck [Gro55] schreef over de Grothendieck ongelijkheden, zijn er heel veel vormen hiervan onderzocht. Ik gebruik de versie uit [KN12]. Er bestaat een universele constante $K > 0$ waarvoor het volgende geldt. Als $m, n$ positieve gehele getallen zijn en $A$ een reële $n \times m$ matrix, dan

$$\sup\{|\sum_{i=1}^{n}\sum_{j=1}^{m} A_{i,j}\langle u_i, v_j\rangle| : u_i, v_j \in \mathbb{R}^{n+m} \text{ met } \|u_i\|_2 = 1 \text{ en } \|v_j\|_2 = 1$$
$$\text{voor } i = 1, \ldots, n \text{ en } j = 1, \ldots, m\} \leq$$
$$K \cdot \sup\{|\sum_{i=1}^{n}\sum_{j=1}^{m} A_{i,j}\alpha_i\beta_j| : \alpha_i, \beta_j \in \{-1, 1\} \text{ voor } i = 1, \ldots, n \text{ en } j = 1, \ldots, m\}$$

met $B$ de eenheidsbol in $H$. Het is bewezen door Krivine en Reeds dat nu dus in het geval $\mathbb{K} = \mathbb{R}$ de beste constante $K_G^{\mathbb{R}}$ waarvoor dit geldt voldoet aan $1.676 \cdots \leq K_G^{\mathbb{R}} \leq 1.782 \ldots$. Het exacte getal is niet bekend. Krivine vermoedde dat deze constante gelijk is aan $1.782 \ldots$, maar het is later bewezen dat dit niet zo is [BMMN13]. Op analoge wijze is de complexe Grothendieck ongelijkheid gedefinieerd. In deze ongelijkheid zitten al veel overeenkomsten met het kleine Grothendieck probleem waar de kern van het artikel over gaat. Voor het (reële) kleine Grothendieck probleem kunnen we deze ongelijkheid gebruiken en deze geeft dat de integrality gap van het Grothendieck probleem met $n \times n$ matrices $C$, $\leq K_G^{\mathbb{R}}$ is en dat geldt dus ook voor het kleine Grothendieck probleem met positief semidefiniete $n \times n$ matrices $C$. Deze ongelijkheid is zoals hij in [KN12] staat en hij is daarin bewerkt ten opzicht van Lindenstrauss-Pełczyńsk [LP68]. De voorwaarde $[-1, 1]$ is namelijk veranderd in $\{-1, 1\}$, dit mag omdat $[-1, 1]$ convex is en de objectieve functie ook (Lemma 7.1). Een zelfde soort argument geeft dat in plaats van $\leq \|\cdot\|$, wordt gebruikt $= \|\cdot\|$. Ook omdat $m + n$ vectoren ten hoogste een $m + n$-dimensionale ruimte opspannen is het voldoende om in plaats van een willekeurige Hilbertruimte $\mathbb{R}^{m+n}$ te nemen met het standaard inproduct. Dat de voorwaarde $m = n$ uit Lindenstrauss-Pełczyński [LP68] is weggelaten, mag omdat iedere vierkante matrix rechthoekig is en deze ongelijkheid voor een rechthoekige matrix equivalent is aan dat voor de vierkante matrix die we uit deze rechthoekige matrix krijgen door deze aan te vullen met nullen zodat hij vierkant wordt. [Die95]

7.6. **De reële kleine Grothendieck ongelijkheid en integrality gap.** Er bestaat een universele constante $k > 0$ waarvoor het volgende geldt. Als n een positief geheel getal is en $A \succeq 0$ een reële $n \times n$ matrix, dan

$$\sup\{\sum_{i,j=1}^{n} A_{i,j}\langle u_i, u_j\rangle : u_i \in \mathbb{R}^n \text{ met } \|u_i\|_2 = 1 \text{ voor } i = 1, \dots, n\} \leq$$

$$K \cdot \sup\{\sum_{i,j=1}^{n} a_{ij}\alpha_i\alpha_j : |\alpha_i| = \{-1, 1\} \text{ voor } i = 1, \dots, n\}$$

[BKS13]

De belangrijkste reden waarom $A \succeq 0$ moet gelden is dat in de literatuur [Ped13] voor deze ongelijkheid gebruik wordt gemaakt van het feit dat $\sum_{i,j=1}^{n} A_{i,j}\alpha_i\alpha_j$ geschreven kan worden als $\langle B\alpha, B\alpha\rangle$. Dit laatste kan alleen als $A$ een Cholesky decompositie heeft dus $A = RR^T$, dan namelijk $B = R^T$. Daaruit volgt $A \succeq 0$. (Stelling 4.6) In het complexe geval wordt de ongelijkheid soortgelijk gedefinieerd. De linkerkant van deze ongelijkheid is voor de $A \succeq 0$ equivalent aan de linkerkant van de Grothendieck ongelijkheid. We zagen al bij paragraaf 7.5 over de Grothendieck ongelijkheid dat $m = n$ nemen geen invloed heeft.
Kies $x, y \in \{-1, 1\}^n$ zó dat de volgende som maximaal is

$$\sum_{i,j=1}^{n} A_{i,j}x_iy_j = x^TAy = x^TB^TBy = \langle By, Bx\rangle.$$

Cauchy-Schwarz: $|\langle x, y\rangle|^2 \leq \langle x, x\rangle \cdot \langle y, y\rangle$, dan volgt $\langle x, x\rangle \geq \langle x, y\rangle$ of $\langle y, y\rangle \geq \langle x, y\rangle$. Dus dan volgt dat het voldoende is te kijken naar $x^TAx$ (m.a.w. we nemen $\alpha_i = \beta_i$ voor $i = 1, \dots, n$) voor de rechterkant van de ongelijkheid en bovendien is $x^TAx \geq 0$ per definitie dus de absoluutstrepen laten we weg. Voor de linkerkant van de ongelijkheid bestuderen we de som

$$\begin{aligned}\sum_{i,j=1}^{n} A_{i,j}\langle b_i, c_j\rangle &= \langle A, BC^T\rangle = \langle D^TD, BC^T\rangle = \mathrm{tr}(D^TD(BC^T)^T)\\ &= \mathrm{tr}(D^TDCB^T) = \mathrm{tr}(B^TD^TDC) = \langle DB, DC\rangle.\end{aligned}$$

Hier is de cholesky decompositie gekozen zó dat $D$ een $n \times n$ matrix is, dit kan vanwege Gevolg 5.3. Weer uit Cauchy-Schwarz volgt dat het voldoende is $u_i = v_i$ te nemen aan de linkerkant van de Grothendieck ongelijkheid. We mogen de absoluutstrepen weg te laten, want

$$\sum_{i,j=1}^{n} A_{i,j}\langle b_i, b_j\rangle = \sum_{i,j=1}^{n} A_{i,j}\sum_{k=1}^{n}(b_i)_k(b_j)_k = \sum_{k=1}^{n}(b^k)^TAb^k$$

met $b^k = ((b_1)_k, (b_2)_k, \dots, (b_n)_k)^T$ en $A \succeq 0$.
De kleine Grothendieck ongelijkheid zou in zekere zin als een zwakkere versie van de Grothendieck ongelijkheid gezien kunnen worden [Ped13], omdat als de Grothendieck ongelijkheid geldt dat geldt de kleine grothendieck ongelijkheid duidelijk ook, omdat deze specifieker is want $A \succeq 0$.

Deze ongelijkheid is ook al door Grothendieck zelf bestudeerd. Het wordt de kleine Grothendieck ongelijkheid genoemd, omdat het bewijs aanzienlijk eenvoudiger is dan de 'normale' Grothendieck ongelijkheid. Alleen in zijn werk stond het in een andere vorm. Grothendieck [Gro55] heeft ook al bewezen dat voor de beste $k$'s waarvoor dit geldt respectievelijk in $\mathbb{R}$ en $\mathbb{C}$, $k_G^{\mathbb{R}} = \frac{\pi}{2}$ en $k_G^{\mathbb{C}} = \frac{4}{\pi}$ zijn. Daar kunnen we iets mee, want deze hangen samen met de integrality gap van het kleine Grothendieck probleem. De waarde $k_G^{\mathbb{R}} = \frac{\pi}{2}$ is dan de integrality gap in het reële geval en zoals we al in paragraaf 7.3 zagen is de integrality gap hoogstens $\frac{\pi}{2}$. Dit is ook af te leiden uit het algoritme, want de verwachtingswaarde van de gecreëerde oplossing met het algoritme is minstens $2/\pi \cdot SDP^{opt}$, want $\alpha_{\mathbb{R}}(1)^2 = \frac{2}{\pi}$. Nu weten we vanwege de ongelijkheid zelfs dat de integrality gap precies $\frac{\pi}{2}$ is. Rietz [Rie74] heeft hier ook een bewijs van gegeven.

## 8. Over de orthogonale groep

Het probleem dat nu zal worden onderzocht is een generalisatie van (3), waarover in [BKS13] over de theorie en toepassingen geschreven is.

$$\max_{O_1,\dots,O_n \in \mathcal{O}_d} \sum_{i=1}^{n}\sum_{j=1}^{n} \mathrm{tr}(C_{i,j}^T O_i O_j^T) \tag{10}$$

$\mathcal{O}_d$ is de groep orthogonale $d \times d$ matrices en $\mathcal{U}_d$ de groep unitaire $d \times d$ matrices. Deze gebruiken we in plaats van de voorwaarde $\{\pm 1\}$ op de variabelen. De matrix $C \in \mathbb{R}^{dn\times dn}$ is positief semidefiniet en $C_{i,j}$ is het (i,j)-de $d \times d$-blok.

8.1. **Relaxatie.** In plaats van de voorwaarde $\in \{\pm 1\}$ bekijken we nu het kleine Grothendieck probleem over de orthogonale groep en zoeken we hierbij een makkelijker probleem ($C \in \mathbb{R}^{dn\times dn}_{\succeq 0}$)

$$\max_{O_1,..,O_n \in \mathcal{O}_d} \sum_{i,j=1}^{n} \mathrm{tr}(C_{i,j}^T O_i O_j^T) \tag{11}$$

Als we kijken naar de matrix $Y = [O_i O_j^T]_{i,j}$ dan komen we tot de conclusie dat deze positief semidefiniet is en $\mathrm{rk}(Y) \leq d$ is wegens Stelling 5.6. Als we op die manier een semidefiniet programma maken krijgen we een SDP van beperkte rang. Daarom maken we eerst een relaxatie.

$$\sup_{m\in\mathbb{N}} \max \sum_{i,j=1}^{n} \mathrm{tr}(C_{i,j}^T X_i X_j^T) : X_i \in \mathbb{R}^{d\times m}; X_i X_i^T = I_{d\times d} \text{ voor } i = 1,\dots,n \tag{12}$$

$$\max \mathrm{tr}(CG) : G \in \mathbb{R}^{dn\times dn}_{\succeq 0}, G_{ii} = I_{d\times d} \text{ voor } i = 1,\dots,n \tag{13}$$

$G = [X_i X_j^T]_{i,j}$

Met behulp van de relaxatie vinden we een SDP zonder beperking op de rang en die dus daarom met de SDP algoritmen oplosbaar is. (12) is een relaxatie van (11). Met Stelling 5.4 volgt de equivalentie van (12) en (13), dus het is een semidefiniet programma en het is vanwege die stelling voldoende om $m = dn$ te bekijken. De matrices $X_i$ in een optimum van (12) kunnen worden verkregen door het semidefiniete programma (13) op te lossen en daarna een Cholesky decompositie toe te passen.

8.2. **Algoritme.** Bereken (een benadering van) de optimale oplossing voor de SDP-relaxatie. Bereken $X_1, \dots, X_n \in \mathbb{R}^{d\times nd}$ door een cholesky decompositie toe te passen op die oplossing. Zie (het bewijs van) Stelling 5.4 voor meer details hierover. Laat $R$ een $nd \times d$ willekeurige matrix zijn met reële i.i.d. $N(0, \frac{1}{d})$ entries. De toekenning van de variabelen voor de oplossing van het kleine Grothendieck probleem over de orthogonale groep wordt nu berekend door: $V_i = \mathcal{P}(X_i R)$. Hier is $\mathcal{P}(X) = \mathrm{argmin}_{Z\in\mathcal{O}_d} ||Z - X||_F$ voor een $X \in \mathbb{R}^{d\times d}$. De norm $||\cdot||_F$ is de Frobenius norm.

Soms moet er meerdere keren een nieuwe matrix $R$ worden gemaakt om een oplossing te krijgen die 'goed genoeg' is.

De polaire decompositie van een complexe $d \times d$ matrix $X$ is $X = UP$ en de linker polaire decompositie $X = P'U$ met met $U$ een unitaire matrix en $P, P'$ een positief-semidefiniete matrix. Stelling 8.1 geeft dat $\mathcal{P}(X) = U$. Merk op als we de singulierewaardenontbinding voor $X$ opschrijven $X = W\Sigma V^*$. Dan zijn $W, V$ $d \times d$ unitaire matrices en $^*$ staat voor de Hermitese getransponeerde. $\Sigma$ is een $d \times d$ diagonaalmatrix met niet-negatieve waarden op de diagonaal.

Dan geldt $U = WV^*$, want $X = W\Sigma V^* = (WV^*)(V\Sigma V^*)$ en $WV^*$ is unitair en $V\Sigma V^* \succeq 0$.

Voor $d = 1$ ronden we in het algoritme voor iedere $i$, $X_iR$ af naar het dichtsbijzijnde element van $\mathcal{O}_1$, dus 1 of -1. Voor algemene $d$ gewoon naar het dichtsbijzijnde element uit $\mathcal{O}_d$.

**Stelling 8.1.** *Zij $A$ een $n \times n$ matrix en $A = W\Sigma V^*$ de singulierewaardenontbinding van $A$. Dan geldt*

$$\operatorname{argmin}_{U \in \mathcal{U}_n} \|U - A\|_F = WV^*$$

*Bewijs.* Laat $U = WV^*$ met $V$ en $W$ gedefinieerd in de stelling. Dan geldt $UU^* = WV^*(WV^*)^* = WV^*VW^* = I$, dus $U$ is unitair. Laat $K$ een willekeurige unitaire $n \times n$ matrix. Definieer $Q = K - U$. Dus we kunnen schrijven $K = U + Q$. Er volgt dan vanwege de unitaire eigenschap $UQ^* + QU^* + QQ^* = 0$. Dan volgt verder

$$\begin{aligned}
\|A - K\|_F^2 &= \|A - U - Q\|_F^2 \\
&= \operatorname{tr}((A - U - Q)(A - U - Q)^*) \\
&= \operatorname{tr}((A - U)(A - U)^* - Q(A - U)^* + QQ^* - (A - U)Q^*) \\
&= \operatorname{tr}((A - U)(A - U)^*) + \operatorname{tr}(-Q(A - U)^* + QQ^* - (A - U)Q^*) \\
&= \|A - U\|_F^2 + \operatorname{tr}(-Q(A - U)^* + QQ^* - (A - U)Q^*) \\
&= \|A - U\|_F^2 + \operatorname{tr}(-QA^* - AQ^*) \\
&= \|A - U\|_F^2 - \operatorname{tr}(QA^* + AQ^*).
\end{aligned}$$

Nu passen we de linker polaire decompositie (gedefinieerd in paragraaf 8.2) toe op $A$. Dan krijgen we $A = PU$ met $U = WV^*$ met $V$ en $W$ zoals hierboven.

$$\begin{aligned}
\operatorname{tr}(QA^* + AQ^*) &= \operatorname{tr}(Q(PU)^* + PUQ^*) \\
&= \operatorname{tr}(QU^*P^* + PUQ^*) \\
&= \operatorname{tr}(QU^*P + PUQ^*) \\
&= \operatorname{tr}(QU^*P) + \operatorname{tr}(PUQ^*) \\
&= \operatorname{tr}(PQU^*) + \operatorname{tr}(PUQ^*) \\
&= \operatorname{tr}(PQU^* + PUQ^*) \\
&= \operatorname{tr}(P(QU^* + UQ^*)) \\
&= -\operatorname{tr}(PQQ^*)
\end{aligned}$$

Omdat $P$ en $QQ^*$ beide positief semidefiniet zijn, volgt $-\operatorname{tr}(PQQ^*)$ is niet positief. (Stelling 4.5) Daaruit concluderen we

$$\|A - K\|_F^2 = \|A - U\|_F^2 - \operatorname{tr}(QA^* + AQ^*) \geq \|A - U\|_F^2 \tag{14}$$

□

[Kel75]

## 8.3. **Kwaliteit.**

**Lemma 8.2.** *Zij $C, S \in \mathbb{K}_{\succeq 0}^{n \times n}$. Dan $\sum_{i,j} C_{i,j}S_{i,j} \geq 0$.*

*Bewijs.* Lemma 1.1.5 uit [Lau12] geeft $\sum_{i,j} C_{i,j}S_{i,j} = \operatorname{tr}(C^TS) = \operatorname{tr}(CS) \geq 0$. □

**Stelling 8.3.** *Laat $C \succeq 0$ en reëel. Laat $X_1, \ldots, X_n$ een toegelaten oplossing zijn voor (5). Laat $V_1, \ldots, V_n \in \mathcal{O}_1$ de output zijn van het algoritme toegepast op deze oplossing. Dan*

$$\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} \operatorname{tr}(C_{i,j}^T V_i V_j^T)\right] \geq \alpha_{\mathbb{R}}(1)^2 \sum_{i=1}^{n}\sum_{j=1}^{n} \operatorname{tr}(C_{i,j}^T X_i X_j^T)$$

*met* $\alpha_{\mathbb{R}}(1)^2 = 2/\pi$.

*Bewijs.* Laat $X_1, \ldots, X_n \in \mathbb{R}^{1\times n}$ een toegelaten oplossing voor (12). Laat $R$ een willekeurige $n \times 1$ matrix met i.i.d. standaard normaal verdeelde entries.

$$\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} \operatorname{tr}(C_{i,j}^T V_i V_j^T)\right] = \mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n} \operatorname{tr}(C_{i,j}^T \mathcal{P}(X_iR)\mathcal{P}(X_jR)^T)\right]$$

$\mathbb{E}[X_iRX_jR] = \mathbb{E}[\sum_{k=1}^{n}(X_i)_kR_k \sum_{m=1}^{n}(X_j)_mR_m] = \sum_{k,m}(X_i)_k(X_j)_m\mathbb{E}[R_kR_m] = \sum_k (X_i)_k(X_j)_k = X_iX_j^T$, want $\mathbb{E}[R_kR_m] = \begin{cases} 0 \text{ als } k \neq m \\ 1 \text{ als } k = m \end{cases}$

De som van standaard normaalverdeelde stochasten is ook normaal verdeeld met gemiddelde $\mu = 0$. $\operatorname{sign}(X_jR) \sim \operatorname{sign}(R_1)$, dus zonder verlies van algemeenheid $X_j = [1, 0, \ldots]$.

$$\mathbb{E}[X_iR\operatorname{sign}(X_jR)] = \mathbb{E}[((X_i)_1R_1 + (X_i)_2R_2 + \ldots)\operatorname{sign}(R_1)] =$$
$$\mathbb{E}[(X_i)_1R_1\operatorname{sign}(R_1)] + \mathbb{E}[(X_i)_2R_2\operatorname{sign}(R_1)] + \mathbb{E}[(X_i)_3R_3\operatorname{sign}(R_1)] + \cdots =$$
$$\mathbb{E}[(X_i)_1R_1\operatorname{sign}(R_1)] + \mathbb{E}[(X_i)_2R_2]\mathbb{E}[\operatorname{sign}(R_1)] + \mathbb{E}[(X_i)_3R_3]\mathbb{E}[\operatorname{sign}(R_1)] + \cdots =$$
$$\mathbb{E}[(X_i)_1R_1\operatorname{sign}(R_1)] = (X_i)_1\mathbb{E}[|R_1|] = \sqrt{\frac{2}{\pi}}X_iX_j^T = \alpha_{\mathbb{R}}(1)X_iX_j^T$$

In de laatste regel wordt gebruik gemaakt van het feit dat als $|Z|$ de absoluut standaard normale verdeling heeft dan is $\mathbb{E}[|Z|] = \sqrt{\frac{2}{\pi}}$. Dit bewijs gaat analoog voor elke andere $X_j = e_i^T$ voor een $i$, waar $e_i^T$ een $1 \times n$ matrix is met op de $i$-de plaats één en voor de rest nullen. Neem nu een willekeurige $X_j \in \mathbb{R}^{1\times n}$. Dan kunnen we die schrijven als $X_j = c_1e_1^T + c_2e_2^T + \cdots + c_ne_n^T$. Dan volgt met het bovenstaande

$$\mathbb{E}[X_iR\operatorname{sign}(X_jR)] = \alpha_{\mathbb{R}}(1)c_1(X_i)_1 + \alpha_{\mathbb{R}}(1)c_2(X_i)_2 \ldots \alpha_{\mathbb{R}}(1)c_n(X_i)_n = \alpha_{\mathbb{R}}(1)X_iX_j^T.$$

Definieer $S_{ij} = (X_iR - \frac{1}{\alpha_{\mathbb{R}}(1)}\operatorname{sign}(X_iR))(X_jR - \frac{1}{\alpha_{\mathbb{R}}(1)}\operatorname{sign}(X_jR)) \succeq 0$. Daarmee wordt bedoeld dat voor iedere vaste $R$ geldt dat $S \succeq 0$. Lemma 8.2 geeft

$$\mathbb{E}[\sum_{i,j} C_{i,j}((X_iR - \frac{1}{\alpha_{\mathbb{R}}(1)}\operatorname{sign}(X_iR))(X_jR - \frac{1}{\alpha_{\mathbb{R}}(1)}\operatorname{sign}(X_jR)))] \geq 0$$

Met de lineairiteit van de verwachtingswaarde en het bovenstaande volgt:

$$0 \leq \mathbb{E}[\sum_{i,j} C_{i,j}\frac{1}{\alpha_{\mathbb{R}}(1)^2}\operatorname{sign}(X_iR)\operatorname{sign}(X_jR) + \sum_{i,j} C_{i,j}X_iRX_jR -$$
$$\sum_{i,j} C_{i,j}\frac{1}{\alpha_{\mathbb{R}}(1)}X_iR\operatorname{sign}(X_jR) - \sum_{i,j} C_{i,j}\frac{1}{\alpha_{\mathbb{R}}(1)}X_jR\operatorname{sign}(X_iR)] =$$
$$\frac{1}{\alpha_{\mathbb{R}}(1)^2}\mathbb{E}[\sum_{i,j} C_{i,j}\operatorname{sign}(X_iR)\operatorname{sign}(X_jR)] + \sum_{i,j} C_{i,j}X_iX_j^T +$$
$$\sum_{i,j} C_{i,j}X_iX_j^T - \sum_{i,j} C_{i,j}X_iX_j^T =$$
$$\frac{1}{\alpha_{\mathbb{R}}(1)^2}\mathbb{E}[\sum_{i,j} C_{i,j}\operatorname{sign}(X_iR)\operatorname{sign}(X_jR)] - \sum_{i,j} C_{i,j}X_iX_j^T$$

Conclusie

$$\mathbb{E}[\sum_{i,j} C_{i,j}\operatorname{sign}(X_iR)\operatorname{sign}(X_jR)] \geq \alpha_{\mathbb{R}}(1)^2\sum_{i,j} C_{i,j}X_iX_j^T$$

□

**Lemma 8.4.** *Laat $M, N \in \mathcal{O}_d$ en $R \in \mathbb{R}^{nd\times d}$ een willekeurige matrix met i.i.d. $\mathcal{N}(0, \frac{1}{d})$ entries. Dan geldt $\mathbb{E}[\mathcal{P}(X_iR)\mathcal{P}(X_jR)^T] = \mathbb{E}[(MR)\mathcal{P}(NR)^T] = \alpha_{\mathbb{R}}(d)X_iX_j^T$.*

**Stelling 8.5.** *Laat $C \succeq 0$ en reëel. Laat $X_1, \dots, X_n$ een toegelaten oplossing zijn voor (12). Laat $V_1, \dots, V_n \in \mathcal{O}_d$ de output zijn van het algoritme toegepast op deze oplossing . Dan*

$$\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{tr}(C_{i,j}^TV_iV_j^T)\right] \geq \alpha_{\mathbb{R}}(d)^2\sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{tr}(C_{i,j}^TX_iX_j^T)$$

*met $\alpha_{\mathbb{R}}(d)^2$ zo gedefinieerd als in paragraaf 8.4. Het complexe geval gaat net zo.*

*Bewijs.* Laat $R \in \mathbb{R}^{nd\times d}$ een willekeurige matrix met i.i.d. $\mathcal{N}(0, \frac{1}{d})$ entries.

$$\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{tr}(C_{i,j}^TV_iV_j^T)\right] = \mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{tr}(C_{i,j}^T\mathcal{P}(X_iR)\mathcal{P}(X_jR)^T)\right]$$

De rest van het bewijs is soortgelijk aan het geval $d = 1$.
Het belangrijkste uit het bewijs is de gelijkheid uit Lemma 8.4

$$\mathbb{E}[\mathcal{P}(X_iR)\mathcal{P}(X_jR)^T] = \mathbb{E}[(X_iR)\mathcal{P}(X_jR)^T] = \alpha_{\mathbb{R}}(d)X_iX_j^T.$$

□

**Gevolg 8.6.** *Laat $C \succeq 0$ en reëel. Laat $X_1^{opt}, \dots, X_n^{opt}$ een optimale oplossing zijn voor (12). Laat $V_1, \dots, V_n \in \mathcal{O}_d$ de output zijn van het algoritme toegepast op deze oplossing . Dan*

$$\mathbb{E}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{tr}(C_{i,j}^TV_iV_j^T)\right] \geq \alpha_{\mathbb{R}}(d)^2\sum_{i=1}^{n}\sum_{j=1}^{n}\mathrm{tr}(C_{i,j}^TX_i^{opt}(X_j^{opt})^T)$$

*met $\alpha_{\mathbb{R}}(d)^2$ zo gedefinieerd als in paragraaf 8.4. Het complexe geval gaat net zo.*

Het algoritme is een $\alpha_{\mathbb{R}}(d)^2$-benaderingsalgoritme. Daarmee bedoelen we de ongelijkheid in Gevolg 8.6. Hier is $\alpha_{\mathbb{R}}(d)^2$ de constante gedefinieerd in hoofdstuk 8.4. ($\alpha_{\mathbb{C}}(d)^2$ in het complexe geval) De $\alpha_{\mathbb{R}}(d)^2$ ($\alpha_{\mathbb{C}}(d)^2$) is een constante die van $d$ afhangt. Voor $d = 1$ zijn dit $\frac{2}{\pi}$ (en $\frac{\pi}{4}$ complex) die later ook nog een aantal keer terug zullen komen. Er geldt dan in het reële geval $\frac{2}{\pi} \cdot SDP^{opt}(C) \leq \mathbb{E}[LGalg(C)] \leq LG^{opt}(C) \leq SDP^{opt}(C)$. De eerste ongelijkheid wordt in Stelling 8.5 bewezen. De tweede ongelijkheid volgt omdat de verwachtingswaarde van de oplossing nooit beter kan zijn dan de optimale oplossing. De derde ongelijkheid volgt uit de uitleg over de samenhang tussen (5) en (6). Vanwege deze ongelijkheid zien we dat om een oplossing te krijgen die ten minste $\frac{2}{\pi} \cdot SDP^{opt}(C)$ is het algoritme vanaf de stap met de willekeurige matrix $R$ herhaald moet worden als de verkregen oplossing niet binnen die grens zit. Meestal hoeft dit in de praktijk hoogstens een paar keer te gebeuren en het is ook mogelijk het algoritme in polynomiale tijd te derandomiseren zodat niet nagegaan hoeft te worden of de oplossing goed genoeg is.

8.4. **Benaderingsconstante.** In dit hoofdstuk gaan we in op de constante $\alpha_{\mathbb{K}}(d)$ voor $\mathbb{K} = \mathbb{R}$ en $\mathbb{K} = \mathbb{C}$. Voor $d = 1$ zagen we al eerder $\alpha_{\mathbb{R}}(1) = \sqrt{\frac{2}{\pi}}$ en $\alpha_{\mathbb{C}}(1) = \sqrt{\frac{\pi}{4}}$. Daarnaast is de definitie $\alpha_{\mathbb{R}}(d) = \mathbb{E}[\frac{1}{d}\sum_{j=1}^{d}\sigma_j(G_{\mathbb{R}})]$ met $G_{\mathbb{R}} \in \mathbb{R}^{d\times d}$ een willekeurige matrix met i.i.d. $\mathcal{N}(0, \frac{1}{d})$ entries en $\sigma_j(G)$ de $j$-de singuliere waarde van $G$. (analoog in het complexe geval) Deze constanten zijn ingewikkeld te berekenen en blijken dus aan de ongelijkheid in Lemma 8.4 te voldoen. Er wordt vermoed dat $\alpha_{\mathbb{R}}(d)$ als functie van $d$ stijgend en $\alpha_{\mathbb{C}}(d)$ als functie van $d$ dalend is. Bovendien is bekend dat $\lim_{d\to\infty}\alpha_{\mathbb{R}}(d) = \lim_{d\to\infty}\alpha_{\mathbb{C}}(d) = \frac{8}{3\pi}$. [BKS13]

Voor $\alpha_{\mathbb{R}}(d)$ is het handig om meer te weten te komen over de verdeling van $\frac{1}{d}\sum_{j=1}^{d}\sigma_j(G_{\mathbb{R}})$. Laat $\mu, \sigma^2$ respectievelijk de verwachtingswaarde en de variantie van deze verdeling zijn. We nemen dan een grote steekproef uit de verdeling *i.i.d.* $X_1, X_2, \ldots, X_n$. De centrale limietstelling geeft

$$\sqrt{n}\frac{\bar{X}-\mu}{\sigma} \approx_{n\to\infty} Z \sim N(0,1).$$

Bekend is dat $P(-1.96 \leq Z \leq 1.96) = 0.95$. Dat geeft

$$0.95 = P(-1.96 \leq Z \leq 1.96) \approx P(-1.96 \leq \sqrt{n}\frac{\bar{X}-\mu}{\sigma} \leq 1.96) = P(\bar{X}-\frac{1.96\sigma}{\sqrt{n}} \leq \mu \leq \bar{X}+\frac{1.96\sigma}{\sqrt{n}}).$$

Voor dit 0.95-betrouwbaarheidsinterval wordt het steekproefgemiddelde $\bar{X}$ gebruikt en $\sigma^2$ benaderen we met behulp van de steekproefvariantie $s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2$. Dit is namelijk een zuivere schatter voor $\sigma^2$. Voor $n = 10000$ is het betrouwbaarheidsinterval verkregen uit simalphr code (zie bijlage) voor verschillende waarden van $d$ gegeven in de volgende tabel.

| d | $\alpha_{\mathbb{R}}(d)$ linkergrens | $\alpha_{\mathbb{R}}(d)$ rechtergrens |
|---|---|---|
| 1 | 0.7748 | 0.7979 |
| 2 | 0.8046 | 0.8166 |
| 3 | 0.8161 | 0.8242 |
| 4 | 0.8218 | 0.8279 |
| 5 | 0.8255 | 0.8304 |

8.5. **Integrality gap.** Het is zinvol te weten hoe groot de integrality gap van de relaxatie van het kleine Grothendieck probleem over de orthogonale/unitaire groep is. Voor $d = 1$ is dat beschreven in het vorige hoofdstuk. Voor $d > 1$ is deze integrality gap moeilijker te berekenen. In [BKS13] wordt bewezen dat het $\frac{1}{\alpha_{\mathbb{R}}^*(d)^2}$ is met

$$\alpha_{\mathbb{R}}^*(d) = \max \mathbb{E}\left[\frac{1}{d}\sum_{k=1}^{d}\sigma_k(GD)\right] : D \text{ diagonaal}, ||D||_F^2 = d \text{ en } D \succeq 0.$$

Ook wordt daarin aangetoond dat $\alpha_{\mathbb{R}}^*(d) = \alpha_{\mathbb{R}}(d)$. In die zin is het algoritme dus optimaal, want het is niet mogelijk dat $\alpha_{\mathbb{R}}^*(d) < \alpha_{\mathbb{R}}(d)$. Als het wel zou gelden dan zou dat namelijk betekenen dat er problemen zijn waarvoor het algoritme een oplossing geeft die beter dan het optimum is en dat is een tegenspraak.

## 9. Toepassingen

9.1. **Het gegeneraliseerde orthogonale Procrustesprobleem.** Het Procrustesprobleem uit de literatuur bestaat uit twee puntenwolken waarvan de punten de rijen uit matrices $A$ en $B$ zijn. Bij ieder punt uit de ene wolk hoort precies een punt uit de andere wolk. Het probleem is dan een orthogonale transformatie te vinden, zodat de punten uit matrix $A$ zo goed mogelijk worden getransformeerd naar de punten uit matrix $B$;

$$\underset{O \in \mathcal{O}_d}{\text{argmin}} \quad \|AO - B\|_F$$

Schönemann heeft bewezen dat de oplossing dan $UV^*$ is met $U\Sigma V^*$ de singulierewaardenontbinding van $A^T B$. [Sch66]

Het gegeneraliseerde Procrustesprobleem dat we hier beschouwen is algemener dan het orthogonale Procrustesprobleem uit de literatuur, omdat het gaat over meerdere puntenwolken en we transformeren (bijv. spiegelen, roteren) de puntenwolken naar elkaar toe in plaats van de ene naar de andere. Praktisch gezien is die eerste toevoeging erg handig. Hierdoor is deze versie krachtiger en zou deze in bepaalde gevallen met $\geq 3$ (geperturbeerde) puntenwolken een betere oplossing hebben, dan als we paarsgewijs het gewone Procrustesprobleem oplossen. Gegeven $n$ puntenwolken van $k$ punten in $\mathbb{R}^d$ ($d$ dimensie). Dan is het doel het vinden van $n$ orthogonale transformaties $O_i$ die tegelijkertijd de puntenwolken zo goed mogelijk uitlijnen. $O_i$ is orthogonaal dan en slechts dan als $O_i^T$ dat is. Dit gebruiken we ook voor het opstellen van het probleem. Er is in dit geval al bekend welke punten uit de verschillende puntenwolken met elkaar corresponderen. We definiëren matrices $A_1, A_2, \ldots, A_n \in \mathbb{R}^{d \times k}$ als volgt. De punten van puntenwolk $i$ zijn de kolommen van matrix $A_i$. Let erop dat de volgorde dus van belang is. Het probleem is dan

$$\min_{O_1,\ldots,O_n \in \mathcal{O}_d} \sum_{i,j=1}^{n} \|O_i^T A_i - O_j^T A_j\|_F^2$$

Omdat het reële matrices zijn kunnen we zeggen

$$\begin{aligned}
\|O_i^T A_i - O_j^T A_j\|_F^2 &= \mathrm{tr}((O_i^T A_i - O_j^T A_j)^T (O_i^T A_i - O_j^T A_j)) \\
&= \mathrm{tr}(((O_i^T A_i)^T - (O_j^T A_j)^T)(O_i^T A_i - O_j^T A_j)) \\
&= \mathrm{tr}((A_i^T (O_i^T)^T - A_j^T (O_j^T)^T)(O_i^T A_i - O_j^T A_j)) \\
&= \mathrm{tr}((A_i^T O_i - A_j^T O_j)(O_i^T A_i - O_j^T A_j)) \\
&= \mathrm{tr}(A_i^T O_i O_i^T A_i - A_j^T O_j O_i^T A_i - A_i^T O_i O_j^T A_j + A_j^T O_j O_j^T A_j) \\
&= \mathrm{tr}(A_i^T A_i - A_j^T O_j O_i^T A_i - A_i^T O_i O_j^T A_j + A_j^T A_j) \\
&= \mathrm{tr}(A_i^T A_i) - \mathrm{tr}(A_j^T O_j O_i^T A_i) - \mathrm{tr}(A_i^T O_i O_j^T A_j) + \mathrm{tr}(A_j^T A_j)) \\
&= \mathrm{tr}(A_i^T A_i) - 2\mathrm{tr}(A_j^T O_j O_i^T A_i) + \mathrm{tr}(A_j^T A_j)) \\
&= \|A_i\|_F^2 + \|A_j\|_F^2 - 2\mathrm{tr}(A_j^T O_j O_i^T A_i) \\
&= \|A_i\|_F^2 + \|A_j\|_F^2 - 2\mathrm{tr}((A_i A_j^T)^T O_i O_j^T)
\end{aligned}$$

Hier gebruiken we Stelling 4.14 en 4.15. Dus het probleem is equivalent aan

$$\max_{O_1,\ldots,O_n \in \mathcal{O}_d} \sum_{i,j=1}^{n} \mathrm{tr}((A_i A_j^T)^T O_i O_j^T).$$

Wegens Stelling 5.5 is de matrix $[A_i A_j^T]_{i,j} \in \mathbb{R}^{dn \times dn}$ positief semidefiniet. Dan volgt direct dat dit probleem een speciaal geval is van het kleine Grothendieck probleem over de orthogonale groep. Door de samenhang van het maximalisatieprobleem en het minimalisatieprobleem kunnen we **niet** zeggen dat een $\frac{2}{\pi}$-benadering voor het maximalisatieprobleem ook een $\frac{2}{\pi}$-benadering voor het minimalisatieprobleem oplevert!

9.2. **Het globale registratieprobleem.** Het globale registratieprobleem is een uitbreiding van het orthogonale Procrustesprobleem. Bij het orthogonale Procrustesprobleem, zagen we dat twee punten uit twee puntenwolken bij elkaar kunnen horen. Hier beschouwen we twee punten die bij elkaar horen als één punt met mogelijk verschillende lokale coördinaten in verschillende puntenwolken. Vandaar dat we het hier gaan hebben over $n$ lokale coördinatensystemen in plaats van puntenwolken. Het is bekend welke lokale coördinaten uit de verschillende lokale coördinatensystemen bij eenzelfde punt horen. Ook nu zoeken we naar orthogonale transformaties $O_i \in \mathcal{O}_d$ ($d$ dimensie) om de punten van de lokale coördinatensystemen zo goed mogelijk op elkaar te draaien. Bovendien staan we nu translaties $t_i \in \mathbb{R}^d$ toe. Een andere toevoeging is dat een punt niet per se in ieder lokaal coördinatensysteem $i$ hoeft te zitten. Zo hoeven de lokale coördinatensystemen dus ook niet evenveel punten te bevatten. Er geldt $k \in P_i$ dan en slechts dan als punt $x_k$ in het $i$-de lokale coördinatensysteem zit. De $x_k^{(i)} \in \mathbb{R}^d$ met $k \in P_i$ zijn de lokale coördinaten van het punt $x_k$. Het globale registratieprobleem wordt dan beschreven door het minimaliseren van

$$\sum_{i=1}^{n} \sum_{k \in P_i} ||x_k - (O_i x_k^{(i)} + t_i)||_2^2 \tag{15}$$

over $x_k, t_i \in \mathbb{R}^d$ en $O_i \in \mathcal{O}_d$. Er zijn $m$ (globale) punten $x_k$. Als er precies twee puntenwolken zijn, dan zijn er ook methoden zonder semidefiniet programmeren om dit op te lossen bijvoorbeeld met een aangepaste versie van de methode van Schönemann. Er bestaan ook ICP algoritmen die dit probleem dan kunnen oplossen. De grote kracht van dat type algoritmen is dat er niet bekend hoeft te zijn welke lokale coördinaten bij welk punt horen. We geven een alternatieve formulering van bovenstaand optimalisatieprobleem waarvoor we de Moore-Penrose pseudoinverse nodig hebben.

**Definitie 9.1.** *Zij $A$ een $m \times n$ matrix. Dan is een Moore-Penrose pseudoinverse van $A$ genoteerd als $A^\dagger$. Een $n \times m$ matrix die aan de volgende eigenschappen voldoet:*
- $AA^\dagger A = A$
- $A^\dagger AA^\dagger = A^\dagger$
- $(AA^\dagger)^* = AA^\dagger$
- $(A^\dagger A)^* = A^\dagger A$

*Zo'n matrix bestaat altijd en is uniek.*

Verder gebruiken we de notatie $e_i^j$ is de kolomvector van lengte $j$ met op de $i$-de plaats een één en voor de rest nullen.

In [CKS13] wordt bewezen dat dit equivalent is aan

$$\max_{O_i \in \mathcal{O}_d} \sum_{i,j=1}^{n} \mathrm{tr}([BL^\dagger B^T]_{i,j} O_i O_j^T) \tag{16}$$

waar

$$L = \sum_{(k,i)\in F} e_{k,i} e_{k,i}^T \quad \text{en} \quad B = \sum_{(k,i)\in F} (e_i^n \otimes I_d) x_k^{(i)} e_{k,i}^T$$

met $e_{k,i} = e_k^{m+n} - e_{k+i}^{m+n}$, $\otimes$ het Kronecker product en $F = \{(k,i) : k \in P_i\}$.

Er valt op dat in (16) alleen maar geoptimaliseerd wordt over de $O_i$. Dit komt omdat er over in [CKS13] wordt geschreven over een methode om de translaties en globale coördinaten terug te krijgen. Het idee is dat er eerst geoptimaliseerd wordt om de orthogonale matrices $O_i$ te vinden. Dan kunnen daaruit globale coördinaten en translaties van ieder lokaal coördinatenstelsel worden gevonden. Voor vaste $O = [O_1, \ldots, O_n]$ zijn de optimale globale coördinaten en translaties uit (15) gegeven door de kolommen van de matrix $OBL^\dagger = [x_1^O, \ldots, x_m^O, t_1^O, \ldots, t_n^O]$. Voor optimale $O$ geldt dan ook $O^{opt}BL^\dagger = [x_1^{opt}, \ldots, x_m^{opt}, t_1^{opt}, \ldots, t_n^{opt}]$, met $O^{opt} = [O_1^{opt}, \ldots, O_n^{opt}]$. Dus zo vinden we uit de orthogonale transformaties verkregen uit het algortime de globale coördinaten en translaties terug.

9.3. **ICP.** Een iterative closest point (ICP) algoritme kan tussen tussen twee geroteerde en getransleerde puntenwolken een rotatie en translatie vinden en ook bepalen welke punten bij elkaar horen. Dit is voor twee puntenwolken dus net iets algemener dan het globale registratieprobleem met SDP, omdat de punten niet gematchd hoeven te zijn. Het nadeel is dat een beginschatting voor de rotatie en translatie bekend moet zijn. Een ICP algoritme kan paarsgewijs worden toegepast op twee puntenwolken van het globale registratieprobleem met meerdere lokale coördinatensystemen om te bepalen welke punten bij elkaar horen voor de SDP aanpak. Er zijn veel verschillende ICP algoritmen die net iets anders werken. De basisstappen zijn:

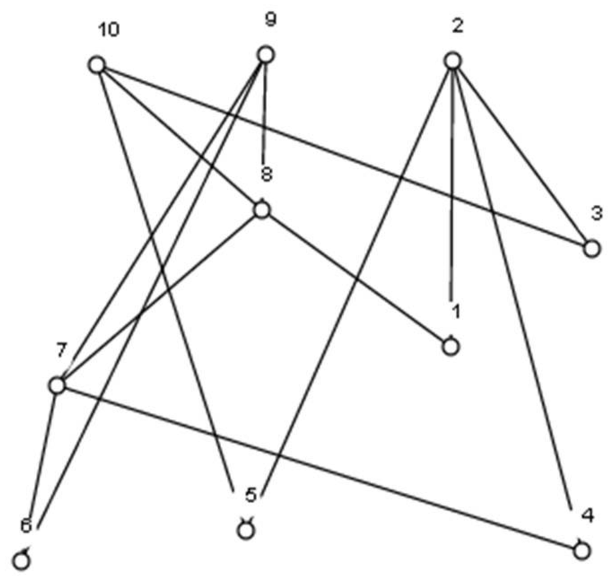
- Punten uit de puntenwolken selecteren
- Punten matchen
- Gewichten toekennen aan de punten
- Paren punten afwijzen
- Met behulp van een error metriek de afstanden van de punten bepalen
- De error metriek minimaliseren

[RL01]

## 10. Experimenten en resultaten

Met behulp van matlabpakketten, waaronder de solver sedumi, zijn voorbeelden van probleem (7) opgelost. In bijlage littlegrothendieck code staat de matlabcode. Dit is namelijk een voorbeeld van een semidefiniet programma. In bijlage goemanswilliamson code wordt beschreven hoe het algoritme uit dit artikel, gebaseerd op dat van Goemans en Williamson,voor $d = 1$ wordt toegepast om een oplossing te krijgen voor (4). Er wordt ook $\leq 4$ keer gecontroleerd of de oplossing minstens $\frac{2}{\pi}$ van de optimale oplossing van de SDP is. LGortho code en GWortho code doen hetzelfde voor de orthogonale kleine grothendieck SDP.
procrustes code is GWortho code voor het orthogonale Procrustesprobleem en globaleregistratie code is GWortho code voor het globale registratieprobleem. Gitaar2 code is speciaal voor het derde experiment hieronder.

10.1. **Experiment pizzabakkers.** In deze paragraaf zal een situatie geschetst worden waarin het maximale snede probleem zou kunnen worden gebruikt. We bevinden ons in een pittoresk stadje in Italië. Daar wonen 14 pizzabakkers. Deze krijgen hun tomaten en kaas geleverd van 10 bezorgers. De bezorgers staken omdat ze vinden dat de werkdruk te hoog is, waardoor alle voorraden kaas en tomaten bij de pizzabakkers op een dag op zijn. Gelukkig is de staking twee dagen daarna afgelopen en beginnen de bezorgers beginnen weer te bezorgen. De werkgever heeft afgesproken om de werkdruk te verlagen, dat gebeurt op twee manieren. Iedere dag hoeven de bezorgers hooguit een van beide producten te bezorgen; dus óf kaas óf tomaten. Hierdoor zijn de bezorgers korter onderweg, want deze producten zijn namelijk op andere locaties af te halen. Daarnaast heeft iedere pizzabakker nu twee vaste bezorgers. Hoeveel pizzabakkers kunnen maximaal de dag dat de staking eindigt zowel tomaten als kaas geleverd krijgen, zodat ze pizza kunnen maken?

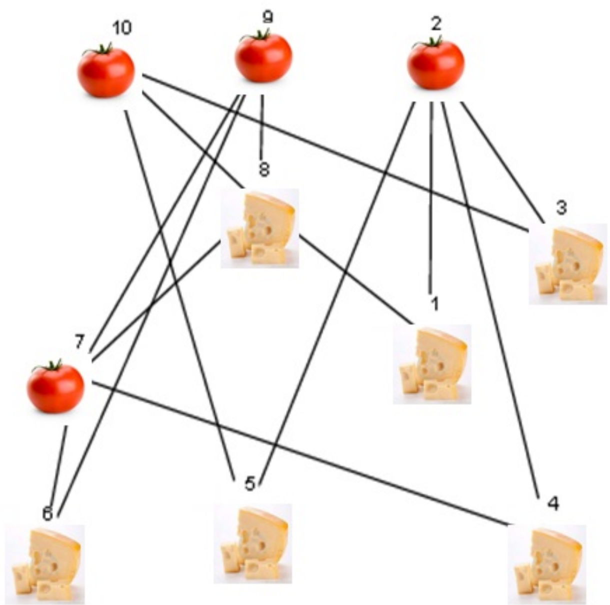



Figuur 1. Pizzagraaf

In Figuur 1 is te zien welke bezorgers welke pizzabakkers moeten bezorgen. Hier staan de lijnen voor de pizzabakkers en de punten voor de bezorgers. Het enige waar over geoptimaliseerd kan worden in dit probleem is de toekenning van kaas of tomaten aan de bezorgers. De graaf valt uiteen in twee verzamelingen punten, de ene verzameling bezorgt tomaten die dag en de andere kaas. Het probleem komt dan neer op het vinden van een maximale snede in de graaf. De Laplacematrix van de graaf is

$$\begin{pmatrix} 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ -1 & 4 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & -1 & 0 & 2 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 2 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & 3 & 0 \\ -1 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 3 \end{pmatrix}.$$

Als we de SDP van het maximale snede probleem oplossen komen we op 12. Daaruit volgt dat voor de maximale snede geldt $0.878 \cdot 12 \leq \mathrm{MC}(C) \leq 12$. Dus de maximale snede heeft grootte 11 of 12. De verdere stappen van het algoritme geven de snede $E[V_t, V_k]$ met $V_t = \{2, 7, 9, 10\}$ en $V_k = \{1, 3, 4, 5, 6, 8\}$. De grootte van deze snede is 12, dit is gelijk aan de waarde van de SDP dus is deze optimaal. In Figuur 2 staat de verkregen toewijzing van kaas en tomaten.



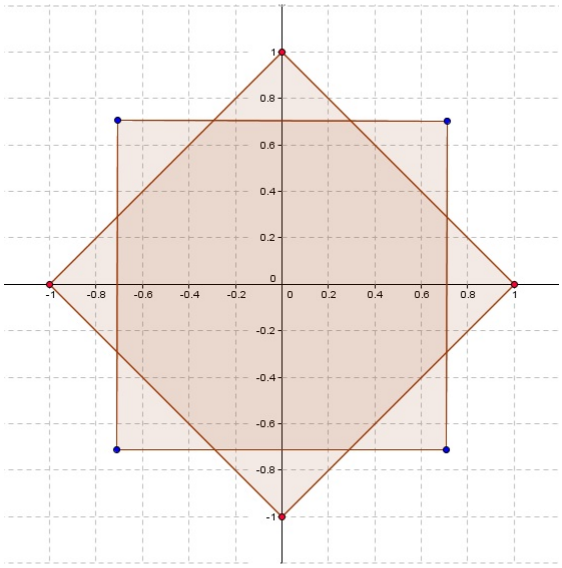

Figuur 2. Pizzagraaf resultaat

10.2. **Experiment vierkanten.** Hier passen we het gegeneraliseerde orthogonale Procrustesprobleem toe op de hoekpunten van een geroteerd vierkant (Figuur 3), waarbij deze een beetje verstoord zijn. De verstoringen worden gecreëerd door meerdere keren te trekken uit $\mathcal{N}(0, 0.05)$ en die waarden bij de coördinaten van de hoekpunten op te tellen.
De hoekpunten van het eerste vierkant zijn

$$a_1 = (1, 0), a_2 = (0, 1), a_3 = (-1, 0), a_4 = (0, -1)$$

en die van de tweede

$$b_1 = \left(\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right), b_2 = \left(-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right), b_3 = \left(-\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}\right), b_4 = \left(\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2}\right).$$



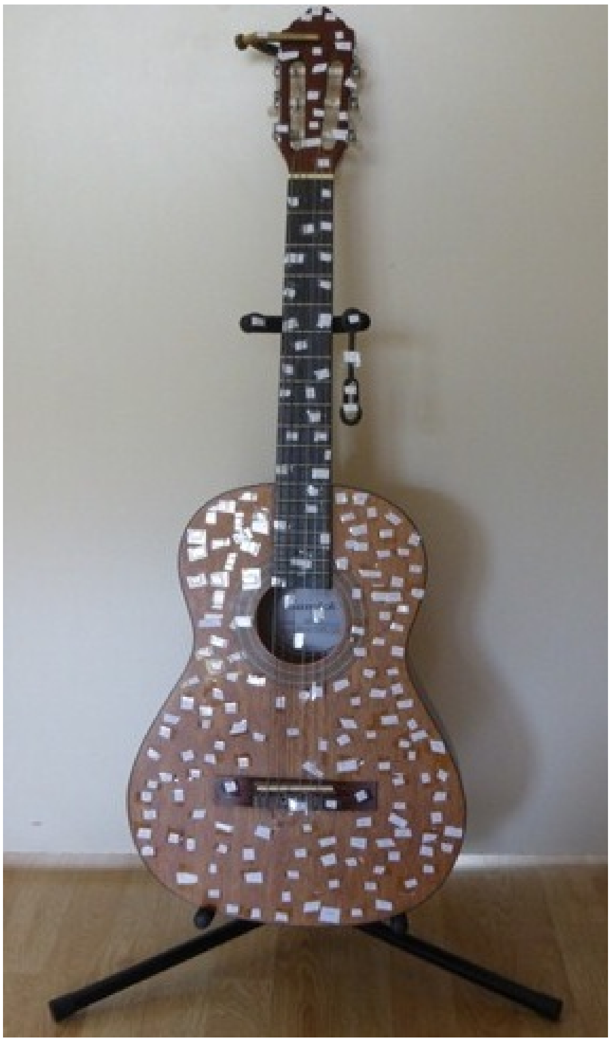

FIGUUR 3. Twee vierkanten

Hier correspondeert $a_i$ met $b_i$. Een mogelijke oplossing die uit het algoritme zou kunnen komen is

$$\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

Want dat is een rotatie van 45 graden van $A$ naar $B$. Met het normale orthogonale Procrustesprobleem zou er zonder verstoringen dit uitkomen en er is dan geen beter antwoord mogelijk omdat ze dan perfect zijn uitgelijnd. Zelfs met de verstoringen is het antwoord niet heel anders en is het zonder verlies van algemeenheid hier voldoende om het normal orthogonale Procrustesprobleem op te lossen en dat kan exact. Dus dan krijg je uit het gegeneraliseerde orthogonale Procrustesprobleem geen beter antwoord. Zoals we weten is het algoritme voor het gegeneraliseerde orthogonale Procrustesprobleem een benaderingsalgoritme en zijn er meestal meerdere combinaties van orthogonale transformaties die een soortgelijk antwoord geven, dus er kan uit het algoritme een heel ander antwoord uitkomen. Wat er na de eerste keer uitvoeren uitkwam was voor $O_A = \begin{pmatrix} 0.6508 & 0.7592 \\ 0.7592 & -0.6508 \end{pmatrix} = O_A^T$ en $O_B = \begin{pmatrix} -0.0768 & 0.9970 \\ 0.9970 & 0.0768 \end{pmatrix} = O_B^T$. Dat is voor $A$ een rotatiematrix van 49 graden en voor $B$ een rotatiematrix van 94 graden. Van te voren dachten we dat er ongeveer een rotatie van 45 graden uit zou komen en dit is ook het geval, want 94-49=45.

10.3. **Experiment gitaar.** Nu kijken we naar een experiment voor het globale registratieprobleem. Dit probleem leent zich voor de fotografie en zou gecombineerd met andere technieken gebruikt kunnen worden voor het maken van panoramafoto's. In dit experiment worden er tweedimensionale foto's gemaakt van een gitaar. Daarin moeten van bepaalde herkenningspunten duidelijk zijn welke punten op de foto's bij elkaar horen. Dan kan er met behulp van het globale registratieprobleem worden gevonden hoe de foto's naar elkaar gedraaid en verschoven kunnen worden. Hier bekijken we hoe het globale registratieprobleem gebruikt kan worden om foto's van een gitaar uit te lijnen en vervolgens vergelijken we dat met als er paarsgewijs de exacte oplosmethode op wordt toegepast. In dit probleem worden 14 foto's van delen van de voorkant van een gitaar genomen vanaf een afstand van ongeveer 20 cm, maar er zijn toch kleine verschillen in afstand opgenomen ook om te zien hoe het algoritme daarmee omgaat. Sommige foto's overlappen en bovendien zitten op de gitaar 200 labels. Het midden van elk label wordt gebruikt als punt in het probleem. De aldus verkregen punten kunnen dus in verschillende foto's verschillende lokale coördinaten hebben. De onnauwkeurigheden zitten in het bepalen waar het midden van elk label zit en het aflezen en afronden van de lokale coördinaten. De afmetingen van alle foto's zijn breedte = 69.088 cm en hoogte = 51.816 cm. Voordat het algoritme wordt toegepast schalen we dit naar breedte = 4.87 cm en hoogte = 3.65 cm. De 14 foto's staan in de bijlage.

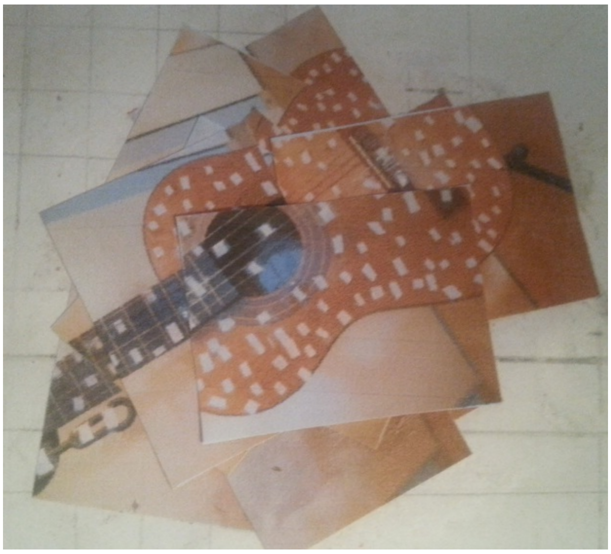


FIGUUR 4. Gitaar

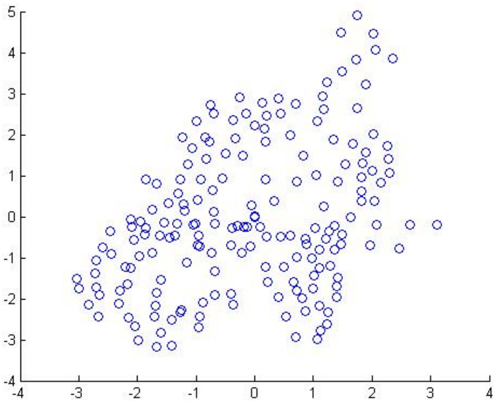
Hierboven staat een afbeelding van de hele gitaar met alle labels. Nu de lokale coördinaten bekend zijn kan de oplossing van (15) worden benaderd met behulp van het algoritme. Na het uitvoeren is de oplossing 24.0. Er valt op te merken dat er in (15) gebruik wordt gemaakt van de som van de kwadraten. Vooral de onnauwkeurigheid bij het meten en de hoeveelheid en plaatsing van de punten leiden tot dit resultaat. De verkregen orthogonale matrices en translaties van de foto's staan in de tabel.

| foto | orthogonale matrix | translatie |
|---|---|---|
| 1 | $\begin{pmatrix} 0.5286 & 0.8489 \\ 0.8489 & -0.5286 \end{pmatrix}$ | $\begin{pmatrix} -2.2058 \\ -1.7085 \end{pmatrix}$ |
| 2 | $\begin{pmatrix} -0.8362 & 0.5484 \\ 0.5484 & 0.8362 \end{pmatrix}$ | $\begin{pmatrix} 2.5103 \\ -2.2837 \end{pmatrix}$ |
| 3 | $\begin{pmatrix} 0.4712 & 0.8820 \\ 0.8820 & -0.4712 \end{pmatrix}$ | $\begin{pmatrix} -2.1977 \\ -0.4914 \end{pmatrix}$ |
| 4 | $\begin{pmatrix} 0.9228 & 0.3854 \\ 0.3854 & -0.9228 \end{pmatrix}$ | $\begin{pmatrix} -2.9239 \\ 0.0105 \end{pmatrix}$ |
| 5 | $\begin{pmatrix} -0.1527 & 0.9883 \\ 0.9883 & 0.1527 \end{pmatrix}$ | $\begin{pmatrix} -2.9146 \\ -0.3993 \end{pmatrix}$ |
| 6 | $\begin{pmatrix} 0.4917 & 0.8708 \\ 0.8708 & -0.4917 \end{pmatrix}$ | $\begin{pmatrix} -2.3105 \\ 1.1488 \end{pmatrix}$ |
| 7 | $\begin{pmatrix} -0.6157 & 0.7880 \\ 0.7880 & 0.6157 \end{pmatrix}$ | $\begin{pmatrix} -0.4052 \\ -1.8730 \end{pmatrix}$ |
| 8 | $\begin{pmatrix} 0.9248 & 0.3806 \\ 0.3806 & -0.9248 \end{pmatrix}$ | $\begin{pmatrix} -2.7242 \\ 1.5661 \end{pmatrix}$ |
| 9 | $\begin{pmatrix} -0.0653 & -0.9979 \\ 0.9979 & -0.0653 \end{pmatrix}$ | $\begin{pmatrix} -1.4286 \\ -2.6346 \end{pmatrix}$ |
| 10 | $\begin{pmatrix} 0.3017 & -0.9534 \\ 0.9534 & 0.3017 \end{pmatrix}$ | $\begin{pmatrix} -1.0875 \\ -2.1031 \end{pmatrix}$ |
| 11 | $\begin{pmatrix} -0.2676 & -0.9635 \\ 0.9635 & -0.2676 \end{pmatrix}$ | $\begin{pmatrix} -0.6162 \\ -2.3466 \end{pmatrix}$ |
| 12 | $\begin{pmatrix} -0.9748 & 0.2233 \\ 0.2233 & 0.9748 \end{pmatrix}$ | $\begin{pmatrix} 4.2459 \\ -2.1556 \end{pmatrix}$ |
| 13 | $\begin{pmatrix} 0.3789 & 0.9254 \\ 0.9254 & -0.3789 \end{pmatrix}$ | $\begin{pmatrix} -3.5973 \\ -0.0770 \end{pmatrix}$ |
| 14 | $\begin{pmatrix} -0.9850 & 0.1725 \\ 0.1725 & 0.9850 \end{pmatrix}$ | $\begin{pmatrix} 2.7938 \\ -0.8452 \end{pmatrix}$ |

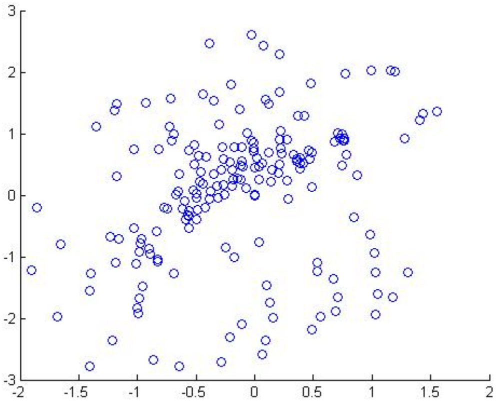
Er is in Figuur 5 zichtbaar wat er met de foto's gebeurt na de draaïngen en spiegelingen. Er valt op dat foto 9, 10 en 11 een rotatie over een bepaalde hoek en de rest een spiegeling over een lijn met een bepaalde hoek hebben gekregen. Foto 9, 10 en 11 zijn van de hals van de gitaar. Dat deze een rotatie hebben meegekregen en de rest een spiegeling, kan liggen aan het feit dat foto 9, 10 en 11 net niet met de rest overlappen. In deze figuur zijn foto 9, 10 en 11 niet opgenomen voor de overzichtelijkheid. Wel valt op te merken dat deze foto's dus duidelijk een andere richting hebben dan de rest van de gitaar en ze overlappen veel met een heel ander deel van de gitaar. Andere dingen die opvallen zijn is dat als het algoritme een aantal keer wordt uitgevoerd er twee typen oplossingen uitkomen de oplossingen rond de 30 en de oplossingen rond de 950. Het eerste type zijn de goede verkregen benaderingen. De oplossingen rond de 950 worden verkregen doordat het algoritme gerandomiseerd is. Dus de grootte van de oplossing kan bij meerdere runs erg variëren. Grafisch gezien is het eerste type oplossing (Figuur 6) een stuk beter kloppende oplossing dan het tweede type (Figuur 7). Alleen foto 9, 10 en 11 sluiten niet goed aan, omdat ze geen overlap hebben met de rest. De globale coördinaten verkregen uit het algoritme staan verderop in vectorvorm. Omdat niet alle punten op een van de foto's staan is voor de punten die niet voorkomen de vector $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ opgenomen in de tabel.



FIGUUR 5. Rotatie en translatie foto's




FIGUUR 6. Voorbeeld type 1 oplossing

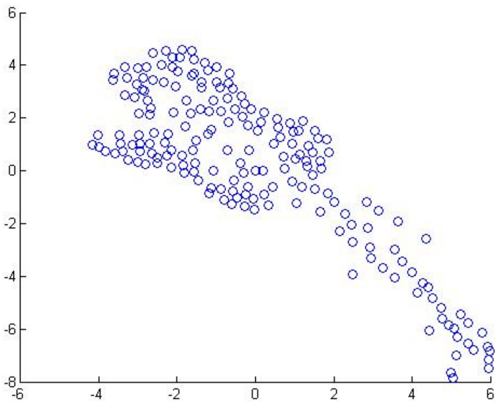



FIGUUR 7. Voorbeeld type 2 oplossing

| punt | glob. coördinaten | punt | glob. coördinaten | punt | glob. coördinaten | punt | glob. coördinaten |
|---|---|---|---|---|---|---|---|
| 1 | $\begin{pmatrix} -0.0407 \\ 1.3245 \end{pmatrix}$ | 26 | $\begin{pmatrix} 0.1566 \\ -1.1055 \end{pmatrix}$ | 51 | $\begin{pmatrix} -2.2898 \\ -0.3814 \end{pmatrix}$ | 76 | $\begin{pmatrix} -0.5905 \\ 1.6592 \end{pmatrix}$ |
| 2 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 27 | $\begin{pmatrix} -1.1986 \\ -0.2567 \end{pmatrix}$ | 52 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 77 | $\begin{pmatrix} 1.6707 \\ 0.3643 \end{pmatrix}$ |
| 3 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 28 | $\begin{pmatrix} 0.6620 \\ 0.9367 \end{pmatrix}$ | 53 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 78 | $\begin{pmatrix} -2.4073 \\ 2.2418 \end{pmatrix}$ |
| 4 | $\begin{pmatrix} 2.5899 \\ 0.5064 \end{pmatrix}$ | 29 | $\begin{pmatrix} 0.1513 \\ -1.3108 \end{pmatrix}$ | 54 | $\begin{pmatrix} 0.6593 \\ 1.5926 \end{pmatrix}$ | 79 | $\begin{pmatrix} 2.6146 \\ -0.0332 \end{pmatrix}$ |
| 5 | $\begin{pmatrix} -2.0898 \\ 1.9219 \end{pmatrix}$ | 30 | $\begin{pmatrix} 0.6367 \\ -1.5384 \end{pmatrix}$ | 55 | $\begin{pmatrix} -0.5397 \\ -0.9347 \end{pmatrix}$ | 80 | $\begin{pmatrix} 1.7584 \\ -0.2307 \end{pmatrix}$ |
| 6 | $\begin{pmatrix} 1.9062 \\ 0.9155 \end{pmatrix}$ | 31 | $\begin{pmatrix} -1.4915 \\ 0.2750 \end{pmatrix}$ | 56 | $\begin{pmatrix} 0.4882 \\ -1.9836 \end{pmatrix}$ | 81 | $\begin{pmatrix} 1.7428 \\ -0.0191 \end{pmatrix}$ |
| 7 | $\begin{pmatrix} -2.7787 \\ 2.2759 \end{pmatrix}$ | 32 | $\begin{pmatrix} -1.6213 \\ 0.3658 \end{pmatrix}$ | 57 | $\begin{pmatrix} -2.1996 \\ 2.1125 \end{pmatrix}$ | 82 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ |
| 8 | $\begin{pmatrix} 1.3652 \\ -0.2349 \end{pmatrix}$ | 33 | $\begin{pmatrix} -0.1054 \\ 0.7407 \end{pmatrix}$ | 58 | $\begin{pmatrix} 0.1676 \\ 2.3545 \end{pmatrix}$ | 83 | $\begin{pmatrix} -0.2248 \\ -0.5554 \end{pmatrix}$ |
| 9 | $\begin{pmatrix} 0.7418 \\ -1.9839 \end{pmatrix}$ | 34 | $\begin{pmatrix} -3.2810 \\ -1.0408 \end{pmatrix}$ | 59 | $\begin{pmatrix} -1.5415 \\ 0.0555 \end{pmatrix}$ | 84 | $\begin{pmatrix} 0.5758 \\ -2.3751 \end{pmatrix}$ |
| 10 | $\begin{pmatrix} 0.4639 \\ -0.3643 \end{pmatrix}$ | 35 | $\begin{pmatrix} -2.5040 \\ 2.8675 \end{pmatrix}$ | 60 | $\begin{pmatrix} -1.5672 \\ 1.5460 \end{pmatrix}$ | 85 | $\begin{pmatrix} -1.5529 \\ 0.6765 \end{pmatrix}$ |
| 11 | $\begin{pmatrix} -1.8626 \\ 0.1532 \end{pmatrix}$ | 36 | $\begin{pmatrix} -1.0875 \\ -2.1031 \end{pmatrix}$ | 61 | $\begin{pmatrix} -1.5141 \\ 0.6569 \end{pmatrix}$ | 86 | $\begin{pmatrix} -0.0194 \\ 0.2355 \end{pmatrix}$ |
| 12 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 37 | $\begin{pmatrix} 2.3389 \\ -1.5091 \end{pmatrix}$ | 62 | $\begin{pmatrix} 2.1926 \\ 0.5946 \end{pmatrix}$ | 87 | $\begin{pmatrix} 3.4661 \\ -0.1424 \end{pmatrix}$ |
| 13 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 38 | $\begin{pmatrix} -0.9973 \\ 0.1134 \end{pmatrix}$ | 63 | $\begin{pmatrix} 3.3793 \\ 0.2420 \end{pmatrix}$ | 88 | $\begin{pmatrix} 0.3484 \\ -1.4273 \end{pmatrix}$ |
| 14 | $\begin{pmatrix} 2.9941 \\ 0.3906 \end{pmatrix}$ | 39 | $\begin{pmatrix} 2.3344 \\ 0.3179 \end{pmatrix}$ | 64 | $\begin{pmatrix} 0.5390 \\ 2.1629 \end{pmatrix}$ | 89 | $\begin{pmatrix} 3.0887 \\ -0.7136 \end{pmatrix}$ |
| 15 | $\begin{pmatrix} 0.7111 \\ 0.0660 \end{pmatrix}$ | 40 | $\begin{pmatrix} -0.1373 \\ 2.3578 \end{pmatrix}$ | 65 | $\begin{pmatrix} 0.8591 \\ -2.6047 \end{pmatrix}$ | 90 | $\begin{pmatrix} 1.1416 \\ -2.4077 \end{pmatrix}$ |
| 16 | $\begin{pmatrix} -1.1011 \\ -0.4035 \end{pmatrix}$ | 41 | $\begin{pmatrix} 0.4295 \\ -0.2592 \end{pmatrix}$ | 66 | $\begin{pmatrix} 0.9196 \\ -1.1775 \end{pmatrix}$ | 91 | $\begin{pmatrix} -1.6355 \\ 0.4157 \end{pmatrix}$ |
| 17 | $\begin{pmatrix} -0.1589 \\ 2.5529 \end{pmatrix}$ | 42 | $\begin{pmatrix} -0.2424 \\ 1.9822 \end{pmatrix}$ | 67 | $\begin{pmatrix} -1.9514 \\ 0.3698 \end{pmatrix}$ | 92 | $\begin{pmatrix} -1.3516 \\ 0.0091 \end{pmatrix}$ |
| 18 | $\begin{pmatrix} 0.5355 \\ 0.9210 \end{pmatrix}$ | 43 | $\begin{pmatrix} 1.0311 \\ 1.1481 \end{pmatrix}$ | 68 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 93 | $\begin{pmatrix} 1.2567 \\ 0.4385 \end{pmatrix}$ |
| 19 | $\begin{pmatrix} -1.2445 \\ 1.8708 \end{pmatrix}$ | 44 | $\begin{pmatrix} -3.5847 \\ -0.9021 \end{pmatrix}$ | 69 | $\begin{pmatrix} -2.9160 \\ 1.9801 \end{pmatrix}$ | 94 | $\begin{pmatrix} 0.6150 \\ -0.0228 \end{pmatrix}$ |
| 20 | $\begin{pmatrix} -0.8258 \\ 1.3121 \end{pmatrix}$ | 45 | $\begin{pmatrix} 3.4812 \\ -0.4478 \end{pmatrix}$ | 70 | $\begin{pmatrix} -3.5631 \\ -0.5971 \end{pmatrix}$ | 95 | $\begin{pmatrix} 2.8672 \\ -1.2699 \end{pmatrix}$ |
| 21 | $\begin{pmatrix} -0.5153 \\ 0.7443 \end{pmatrix}$ | 46 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 71 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 96 | $\begin{pmatrix} -4.0220 \\ -1.9409 \end{pmatrix}$ |
| 22 | $\begin{pmatrix} 1.1526 \\ 1.2529 \end{pmatrix}$ | 47 | $\begin{pmatrix} 0.5608 \\ -0.8309 \end{pmatrix}$ | 72 | $\begin{pmatrix} 0.6537 \\ -1.3679 \end{pmatrix}$ | 97 | $\begin{pmatrix} 0.5275 \\ 1.8541 \end{pmatrix}$ |
| 23 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 48 | $\begin{pmatrix} 0.4226 \\ 1.1037 \end{pmatrix}$ | 73 | $\begin{pmatrix} -2.3878 \\ 1.7126 \end{pmatrix}$ | 98 | $\begin{pmatrix} -1.2062 \\ -0.4929 \end{pmatrix}$ |
| 24 | $\begin{pmatrix} 2.5691 \\ 0.8069 \end{pmatrix}$ | 49 | $\begin{pmatrix} 1.9969 \\ -1.0739 \end{pmatrix}$ | 74 | $\begin{pmatrix} 2.1212 \\ 0.2198 \end{pmatrix}$ | 99 | $\begin{pmatrix} -1.7418 \\ -0.0924 \end{pmatrix}$ |
| 25 | $\begin{pmatrix} 1.5115 \\ -1.1049 \end{pmatrix}$ | 50 | $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ | 75 | $\begin{pmatrix} -3.1420 \\ -0.6082 \end{pmatrix}$ | 100 | $\begin{pmatrix} 0.6856 \\ 1.2233 \end{pmatrix}$ |

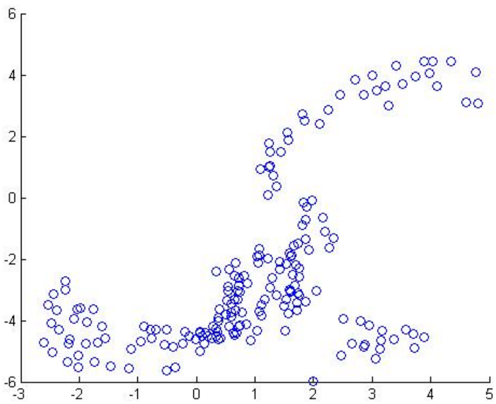
| punt | glob. coördinaten | punt | glob. coördinaten | punt | glob. coördinaten | punt | glob. coördinaten |
|---|---|---|---|---|---|---|---|
| 101 | $\begin{pmatrix}-0.7815\\0.8160\end{pmatrix}$ | 126 | $\begin{pmatrix}1.7564\\-1.0378\end{pmatrix}$ | 151 | $\begin{pmatrix}0\\0\end{pmatrix}$ | 176 | $\begin{pmatrix}0.2754\\1.8264\end{pmatrix}$ |
| 102 | $\begin{pmatrix}-2.3866\\0.4381\end{pmatrix}$ | 127 | $\begin{pmatrix}-0.0576\\-1.5837\end{pmatrix}$ | 152 | $\begin{pmatrix}-0.5126\\1.8079\end{pmatrix}$ | 177 | $\begin{pmatrix}0.3808\\1.5458\end{pmatrix}$ |
| 103 | $\begin{pmatrix}3.2290\\-0.1132\end{pmatrix}$ | 128 | $\begin{pmatrix}0\\0\end{pmatrix}$ | 153 | $\begin{pmatrix}0.8732\\0.5303\end{pmatrix}$ | 178 | $\begin{pmatrix}-1.3442\\0.9071\end{pmatrix}$ |
| 104 | $\begin{pmatrix}-2.0114\\0.6572\end{pmatrix}$ | 129 | $\begin{pmatrix}-1.8068\\0.3169\end{pmatrix}$ | 154 | $\begin{pmatrix}-0.3125\\2.0733\end{pmatrix}$ | 179 | $\begin{pmatrix}-0.7781\\1.8964\end{pmatrix}$ |
| 105 | $\begin{pmatrix}2.0972\\0.8837\end{pmatrix}$ | 130 | $\begin{pmatrix}0.5753\\-1.0228\end{pmatrix}$ | 155 | $\begin{pmatrix}-1.5755\\0.0069\end{pmatrix}$ | 180 | $\begin{pmatrix}-0.9306\\1.8107\end{pmatrix}$ |
| 106 | $\begin{pmatrix}-0.7849\\0.4011\end{pmatrix}$ | 131 | $\begin{pmatrix}1.0594\\-1.9491\end{pmatrix}$ | 156 | $\begin{pmatrix}-3.6793\\-1.5680\end{pmatrix}$ | 181 | $\begin{pmatrix}0\\0\end{pmatrix}$ |
| 107 | $\begin{pmatrix}2.7440\\-0.9141\end{pmatrix}$ | 132 | $\begin{pmatrix}-0.8656\\-0.6662\end{pmatrix}$ | 157 | $\begin{pmatrix}1.6659\\-1.3564\end{pmatrix}$ | 182 | $\begin{pmatrix}3.1197\\-0.9247\end{pmatrix}$ |
| 108 | $\begin{pmatrix}-0.5946\\2.4523\end{pmatrix}$ | 133 | $\begin{pmatrix}-0.4446\\-1.6184\end{pmatrix}$ | 158 | $\begin{pmatrix}0.7294\\-2.4707\end{pmatrix}$ | 183 | $\begin{pmatrix}0\\0\end{pmatrix}$ |
| 109 | $\begin{pmatrix}-0.2500\\-0.8307\end{pmatrix}$ | 134 | $\begin{pmatrix}0\\0\end{pmatrix}$ | 159 | $\begin{pmatrix}0.2415\\1.9829\end{pmatrix}$ | 184 | $\begin{pmatrix}0\\0\end{pmatrix}$ |
| 110 | $\begin{pmatrix}2.5780\\0.1687\end{pmatrix}$ | 135 | $\begin{pmatrix}-0.4748\\1.4051\end{pmatrix}$ | 160 | $\begin{pmatrix}-1.1913\\-0.9318\end{pmatrix}$ | 185 | $\begin{pmatrix}0.4102\\-1.6358\end{pmatrix}$ |
| 111 | $\begin{pmatrix}1.9902\\0.6129\end{pmatrix}$ | 136 | $\begin{pmatrix}-2.6092\\0.1369\end{pmatrix}$ | 161 | $\begin{pmatrix}2.7008\\-0.0867\end{pmatrix}$ | 186 | $\begin{pmatrix}-0.8321\\-0.0967\end{pmatrix}$ |
| 112 | $\begin{pmatrix}-2.9030\\3.0804\end{pmatrix}$ | 137 | $\begin{pmatrix}1.1584\\0.3404\end{pmatrix}$ | 162 | $\begin{pmatrix}-2.9233\\2.6197\end{pmatrix}$ | 187 | $\begin{pmatrix}-3.9231\\-2.6080\end{pmatrix}$ |
| 113 | $\begin{pmatrix}-1.3444\\1.3684\end{pmatrix}$ | 138 | $\begin{pmatrix}-4.2821\\-1.4697\end{pmatrix}$ | 163 | $\begin{pmatrix}1.5659\\-1.2826\end{pmatrix}$ | 188 | $\begin{pmatrix}0.9883\\-0.4682\end{pmatrix}$ |
| 114 | $\begin{pmatrix}0.9513\\0.8936\end{pmatrix}$ | 139 | $\begin{pmatrix}2.6012\\-0.7279\end{pmatrix}$ | 164 | $\begin{pmatrix}-3.9184\\-1.2073\end{pmatrix}$ | 189 | $\begin{pmatrix}-0.5459\\-0.2096\end{pmatrix}$ |
| 115 | $\begin{pmatrix}-0.2916\\-1.1609\end{pmatrix}$ | 140 | $\begin{pmatrix}2.5689\\-1.1342\end{pmatrix}$ | 165 | $\begin{pmatrix}0.0415\\-1.6917\end{pmatrix}$ | 190 | $\begin{pmatrix}2.2705\\-0.3544\end{pmatrix}$ |
| 116 | $\begin{pmatrix}1.6060\\1.0404\end{pmatrix}$ | 141 | $\begin{pmatrix}3.1522\\0.1167\end{pmatrix}$ | 166 | $\begin{pmatrix}1.6303\\0.6923\end{pmatrix}$ | 191 | $\begin{pmatrix}-2.0154\\1.3259\end{pmatrix}$ |
| 117 | $\begin{pmatrix}3.2411\\-1.2759\end{pmatrix}$ | 142 | $\begin{pmatrix}0.8232\\-0.8743\end{pmatrix}$ | 167 | $\begin{pmatrix}1.4786\\0.0434\end{pmatrix}$ | 192 | $\begin{pmatrix}2.4446\\-0.5074\end{pmatrix}$ |
| 118 | $\begin{pmatrix}-0.1350\\1.7107\end{pmatrix}$ | 143 | $\begin{pmatrix}-0.3427\\1.8555\end{pmatrix}$ | 168 | $\begin{pmatrix}-1.6182\\0.6716\end{pmatrix}$ | 193 | $\begin{pmatrix}-1.4952\\-0.2449\end{pmatrix}$ |
| 119 | $\begin{pmatrix}-2.3847\\0.0445\end{pmatrix}$ | 144 | $\begin{pmatrix}2.8756\\-1.6561\end{pmatrix}$ | 169 | $\begin{pmatrix}0\\0\end{pmatrix}$ | 194 | $\begin{pmatrix}-2.9239\\0.0105\end{pmatrix}$ |
| 120 | $\begin{pmatrix}0.9558\\-1.5090\end{pmatrix}$ | 145 | $\begin{pmatrix}2.8155\\0.6163\end{pmatrix}$ | 170 | $\begin{pmatrix}2.8812\\-0.1481\end{pmatrix}$ | 195 | $\begin{pmatrix}-1.2480\\0.3761\end{pmatrix}$ |
| 121 | $\begin{pmatrix}-2.2492\\0.1359\end{pmatrix}$ | 146 | $\begin{pmatrix}0.1667\\-1.9472\end{pmatrix}$ | 171 | $\begin{pmatrix}0.1928\\-0.4944\end{pmatrix}$ | 196 | $\begin{pmatrix}-1.6853\\1.8230\end{pmatrix}$ |
| 122 | $\begin{pmatrix}2.7476\\-1.1631\end{pmatrix}$ | 147 | $\begin{pmatrix}0.2936\\-0.9698\end{pmatrix}$ | 172 | $\begin{pmatrix}3.0599\\-0.3737\end{pmatrix}$ | 197 | $\begin{pmatrix}1.4773\\-0.5717\end{pmatrix}$ |
| 123 | $\begin{pmatrix}1.3110\\0.1798\end{pmatrix}$ | 148 | $\begin{pmatrix}-2.0039\\0.1455\end{pmatrix}$ | 173 | $\begin{pmatrix}-1.3501\\-0.8341\end{pmatrix}$ | 198 | $\begin{pmatrix}-0.4507\\0.4214\end{pmatrix}$ |
| 124 | $\begin{pmatrix}1.2503\\0.7757\end{pmatrix}$ | 149 | $\begin{pmatrix}3.2318\\0.3038\end{pmatrix}$ | 174 | $\begin{pmatrix}-2.5992\\-0.0431\end{pmatrix}$ | 199 | $\begin{pmatrix}-1.2616\\0.5874\end{pmatrix}$ |
| 125 | $\begin{pmatrix}0.1541\\1.5084\end{pmatrix}$ | 150 | $\begin{pmatrix}-0.5168\\-0.6485\end{pmatrix}$ | 175 | $\begin{pmatrix}2.2040\\-1.3171\end{pmatrix}$ | 200 | $\begin{pmatrix}3.0822\\-1.5643\end{pmatrix}$ |

Met behulp van de resultaten zien we dat extra foto's (foto 15 en 16) van het midden van de hals gewenst zijn. Met die extra foto's zou een beter kloppend resultaat kunnen worden geproduceerd. Er zijn ook nu weer twee typen oplossingen en de waarden liggen zeer ver uit elkaar. Dit kan omdat de constante $\alpha_{\mathbb{R}}(2)$ iets zegt over het maximalisatieprobleem (16), maar niet veel over (15). Er is daarin ook duidelijk dat de smalle vorm van de hals en de hoeveelheid punten op de hals ook zeker van invloed zijn op de resultaten. Dit stuk wordt namelijk zelfs bij een type 1 oplossing niet altijd goed weergegeven. Na een 30 keer runnen is het beste resultaat 26.3. Dit is te zien in Figuur 8. Hier is opvallend dat de vorm van de gitaar zeer duidelijk te herkennen is.



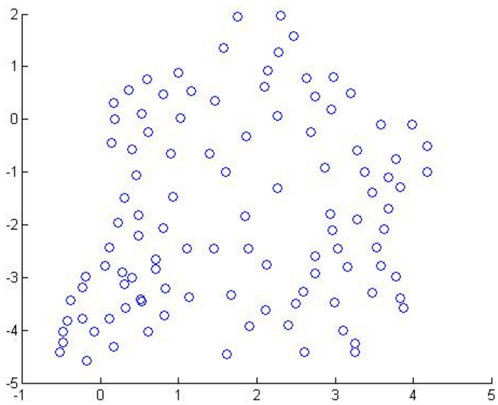

Figuur 8. Resultaat met extra foto's

Nu zal bestudeerd worden hoe goed de paarsgewijze exacte methode gebaseerd op Schönemann is. Daarvoor nemen we eerst bijvoorbeeld foto 1. Dan kijken we welke foto $i$ de meeste punten gemeenschappelijk heeft met foto 1 en kijken we uitsluitend naar de punten die in beide foto's voorkomen. Vervolgens verschuiven we het zwaartepunt van beide puntenwolken naar de oorsprong. Dan zoeken we met de exacte methode de beste rotatie. Vervolgens kijken we naar welke foto de meeste punten gemeenschappelijk heeft met óf foto 1 óf foto $i$. Dan herhalen we de stappen tussen die twee foto's, totdat dit voor alle 16 foto's is gedaan. Hier zoeken we steeds foto's die veel gemeenschappelijke punten hebben, zo wordt meer informatie benut. Merk ook op dat als er op een andere manier paarsgewijs foto's bij elkaar worden gezocht, er een betere oplossing uit zou kunnen komen. In Figuur 9 staat de oplossing. Er valt op dat een aantal




Figuur 9. Resultaat met Schönemann

punten dat zich aan het uiteinde van de gitaar bevinden apart zijn geplaatst en dat de klankkast van de gitaar niet te herkennen is. De globale coördinaten zijn gecreëerd door de gemiddelde x- en y-coördinaten te nemen van ieder punt. In deze strategie wordt niet van alle informatie gebruik gemaakt omdat er steeds maar naar twee foto's wordt gekeken en uit de figuur blijkt dat onvoldoende door alle onnauwkeurigheden bij bijvoorbeeld het meten van de coördinaten van de punten. In de figuur is foto 16 eerst genomen dat gaf de beste oplossing met deze strategie. Hier is de hals te herkennen, de klankkast is erg vervormd en de punten linksonder op de klankkast staan in de foto rechtsonder los van de rest. Het lijkt alsof de gitaar is stukgeslagen. De methode met semidefiniet programmeren geeft dus een beter resultaat. Aanvankelijk heb ik beide methoden uitgevoerd en toen bleek de gitaar vrijwel onherkenbaar met deze methode. Vervolgens heb ik nog enkele aflees- en meetfouten uit de data gecorrigeerd. Een van de redenen waarom deze methode hier niet goed werkt is omdat sommige stukken van de gitaar in veel verschillende foto's staan en het algoritme weet niet goed hoe om te gaan met meerdere foto's die overlappen. Hier maken meer foto's het resultaat dus niet altijd beter. In Figuur 10 is de oplossing te zien op basis van 3 foto's van een gedeelte van de gitaar; foto 1, 3 en 13. Hier is de gitaar veel beter te herkennen. Bij deze methode moet dus meer worden nagedacht over welke foto's overlappen om een goed resultaat te krijgen.




Figuur 10. Resultaat met Schönemann op basis van 3 foto's

## Referenties

[Ali95] Farid Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization, 1995.

[BKS13] Afonso S. Bandeira, Christopher Kennedy, and Amit Singer. Approximating the little grothendieck problem over the orthogonal and unitary groups. *arXiv:1308.5207v3 [cs.DS]*, 2013.

[BMMN13] Mark Braverman, Konstantin Makarychev, Yury Makarychev, and Assaf Naor. The grothendieck constant is strictly smaller than krivine's bound. *Forum of Mathematics,Pi. Cambridge University Press*, page e4, 2013.

[CKS13] Kunal N. Chaudhury, Yeohaw Khoo, and Amit Singer. Global registration of multiple point clouds using semidefinite programming. *arXiv 1306.5226v3 [cs.CV]*, 2013.

[CW04] Moses Charikar and Anthony Wirth. Maximizing quadratic programs: extending grothendieck's inequality. *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium*, pages 54–60, 2004.

[Die95] Joe Diestel. Absolutely summing operators vol. 43. *Cambridge university press*, 1995.

[Fre04] Robert M. Freund. Introduction to semidefinite programming (sdp). Massachusetts Institute of Technology, 2004.

[Gro55] Alexander Grothendieck. Produits tensoriels topologiques et espaces nucléaires. *No. 16. American mathematical society*, 1955.

[GW95] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, pages 1115–1145, 1995.

[Kel75] Joseph B. Keller. Closest unitary, orthogonal and hermitian operators to a given operator. *Mathematics magazine 48(4)*, pages 192–197, 1975.

[KN12] Subhash Khot and Assaf Naor. Grothendieck-type inequalities in combinatorial optimization. *Communications on Pure and Applied Mathematics*, pages 992–1035, 2012.

[Lau12] Monique Laurent. Networks and semidefinite programming, 2012.

[LP68] J Lindenstrauss and A Pełczyński. Absolutely summing operators in $L_p$-spaces and their applications. *Studia Math.*, pages 29:275–326, 1968.

[Nes97] Yu Nesterov. Semidefinite relaxation and nonconvex quadratic optimization. *Optimization methods and software*, pages 141–160, 1997.

[Ped13] Gilles Pisier and equipe d'analyse. Grothendiecks theorem, past and present uncut updated and still expanding... *Bulletin of the American Mathematical Society 49.2*, 2013.

[Pra05] Piotr Pragacz. Notes on the life and work of alexander grothendieck. *Springer*, 2005.

[Reb07] Steffen Rebennack. Ellipsoid method. *Encyclopedia of optimization*, 2007.

[Rie74] Ronald E. Rietz. A proof of the grothendieck inequality. *Israel journal of mathematics*, pages 271–276, 1974.

[RL01] Szymon RusinkieWicz and Marc Levoy. Efficient variants of the icp algorithm. *3-D Digital Imaging and Modeling*, pages 145–152, 2001.

[Ros04] Amy Ross. Procrustes analysis. Department of Computer Science and Engineering, University of South Carolina, 2004.

[Sch66] Peter H. Schönemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, pages 1–10, 1966.

[SZY07] Anthony Man-Cho So, Jiawei Zhang, and Yinyu Ye. On approximating complex quadratic optimization problems via semidefinite programming relaxations. *Mathematical programming*, pages 93–110, 2007.

[VB96] Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM review 38(1)*, pages 49–95, 1996.

# Bijlagen

### LITTLEGROTHENDIECK CODE

```
function [val, M] = littlegrothendieck(C)
    options = sdpsettings('solver', 'sedumi');
    X = sdpvar(size(C,1));
    constraints = [diag(X) == ones(size(C,1),1), X >= 0];
    objective = -trace(C*X);    %/4 als we Lapacematrix van een graaf gebruiken
    solution = solvesdp(constraints, objective, options);
    val = -double(objective);
    M = double(X);
end
```

### GOEMANSWILLIAMSON CODE

```
function [value, result] = goemanswilliamson(C)
    n = size(C,1);
    [val, M] = littlegrothendieck(C); %val is de waarde van de SDP
    R = chol(M+0.00000001*eye(n));
    k=0;
    t=0;
    while k==0
        t=t+1;
        r = randn(n,1);
        r = r / norm(r);
        result = ones(n, 1);
        for i = 1:n
            if r' * R(:, i) <= 0
                result(i) = -1;
            end
        end
        value=0;
        for i=1:n
            for j=1:n
                value=value+C(i,j)*result(i)*result(j); %/4 als we Lapacematrix van een graaf gebruiken
            end
        end
        if value>0.878*val
            k=1;
        end
        if t>3
            k=1;
            disp('het resultaat is niet goed genoeg')
        end
    end
end
```

### LGortho code

```
function [val, M] = LGortho(C,d)
    n=size(C,1)/d;
    options = sdpsettings('solver', 'sedumi');
    X = sdpvar(size(C,1));
    constraints=[X >= 0];

    for i=1:n
```

```
            constraints = constraints + [X(((i-1)*d+1):i*d,((i-1)*d+1):i*d) == eye
    (d)];
    end
    objective = -trace(C*X);
    solution = solvesdp(constraints, objective, options);
    val = -double(objective);
    M = double(X);
end
```

## GWortho code

```
function [result,W] = GWortho(C,d)
    l = size(C,1);
    n=l/d;
    [val, M] = LGortho(C,d); %val SDP waarde
    R = chol(M+0.00000001*eye(l)); %M=R'R

    for i=1:n
        for j=1:d
            Y(j,:,i)=R(:,(i-1)*d+j)';
        end
    end
    R=normrnd(0,1/d,n*d,d);
    for i=1:n
        [U,S,V]=svd(Y(:,:,i)*R);
        W(:,:,i)=U*V';
    end
    result=0;
    for i=1:n
        for j=1:n
            result=result+sum(diag(C((i-1)*d+1:i*d,(j-1)*d+1:j*d)'*W(:,:,i)*W(:,:,
    j)'));
        end
    end
end
```

## Procrustes code

```
function [v, M] = procrustes(A)
%A(:,:,i) is de matrix met punten als kolommen in het i-de lokale
%coordinatensysteem
n=size(A,3);
d=size(A,1);
k=size(A,2);

E=[];
for i=1:n
    D=[];
    for j=1:n
        D=[D,A(:,:,i)*A(:,:,j)'];
    end
    E=[E;D];
end
[val,M]=GWortho(E,d);
v=-2*val;
for f=1:n
    v=v+2*n*trace(A(:,:,f)'*A(:,:,f));
```

```
    end
end
```

GLOBALEREGISTRATIE CODE

```
function [m,Z,waarde]=globaleregistratie(A,D)
%A is de MxN matrix met op de (i,k)-de plaats een 1 als punt x_k in het
%i-de lokale coordinatenstelsel zit,
%de matrices D(:,:,j) hebben als kolommen de coordinaten van de punten in
%het j-de lokale coordinatenstelsel.
M=size(A,1);
N=size(A,2);
d=size(D,1);

L=zeros(M+N);
B=zeros(M*d,N+M);
for i=1:M
    for k=1:N
        if A(i,k)==1
            b=zeros(N+M,1);
            c=zeros(N+M,1);
            b(k,1)=1;
            c(N+i,1)=1;
            a=b-c;
            L=L+a*a';

            f=zeros(M,1);
            f(i,1)=1;
            B=B+kron(f,eye(d))*D(:,k,i)*a';
        end
    end
end
C=B*pinv(L)*B'; %pinv is de monrose pseudoinverse
[val,m]=GWortho(C,d); %de m(:,:,i) zijn de orthogonale matrices

O=[];
for i=1:M
    O=[O,m(:,:,i)];
end

Z=O*B*pinv(L); %hierin zitten de globale coordinaten en de translaties

waarde=0; %nu bereken hoe goed de oplossing is
for i=1:M
    for k=1:N
        if A(i,k)==1
            waarde=waarde+sum((Z(:,k)-m(:,:,i)*D(:,k,i)-Z(:,N+i)).^2);
        end
    end
end
end
```

SIMALPHR CODE

```
function b = simalphr(d)

    for i= 1:10000
```

```
            G=normrnd(0,1/sqrt(d),d,d);
            c(i)=mean(svd(G));
        end

        b(1)=mean(c)-1.96*sqrt(sum((c-mean(c)).^2)/9999)/sqrt(10000);
        b(2)=mean(c)+1.96*sqrt(sum((c-mean(c)).^2)/9999)/sqrt(10000);
end
```

GITAAR2 CODE

```
A=zeros(16,200); %in dit geval 16 foto's
A(1,16)=1;
A(1,132)=1;
A(1,189)=1;
A(1,150)=1;
%etc.

D=zeros(2,200,16);
D(:,16,1)=[2050;100];
D(:,132,1)=[1900;500];
D(:,189,1)=[2500;450];
%etc.
D=0.0141111111*D; %van pixels naar centimeters      (180 pix/inch)
D=D*5.5/78; %schalen
[m,z,w]=globaleregistratie(A,D)
X=z(1,1:200);
Y=z(2,1:200);
scatter(X,-Y)%plot oplossing
```

## Foto's

(a) Foto 1. (b) Foto 2. (c) Foto 3. (d) Foto 4.

(e) Foto 5. (f) Foto 6. (g) Foto 7. (h) Foto 8.

(i) Foto 9. (j) Foto 10. (k) Foto 11. (l) Foto 12.

(m) Foto 13. (n) Foto 14. (o) Foto 15. (p) Foto 16.

Figuur 11. Foto's van de gitaar