勉強会形式ゼミ資料②
L. Ducas, L. N. Pulles and M. Stevens Towards a modern LLL implementation[1]

佐藤 新

July 7, 2025

本セミナーで用いられる記号など

- $\boldsymbol{A} \in M_{n,m}(\mathbb{R})$ に対して，$\|\boldsymbol{A}\|_{\max} := \max_{i,j} |a_{i,j}|$
- $\boldsymbol{A} \in M_{n,m}(\mathbb{R})$ に対して，$\boldsymbol{A}_{[r,s]\times[t,u]} := [a_{i,j}]_{r\leq i\leq s, t\leq j\leq u}$

[1]の貢献

- BLASter の提案
  - 高速かつモダンな LLL の実装
  - [2] の分割手法を利用
  - サイズ簡約を Seysen 簡約に置き換え
  - Cholesky 分解を QR 分解に置き換え
  - 更に deep や BKZ も実装

[2] の分割手法

定義 1 (specific dimension)

specific dimension n_k を以下で定める

$$n_1 \coloneqq 2, n_k \coloneqq \max\left\{ pn_{k-1} \;\middle|\; pn_{k-1} \leq \sqrt{2}^{4-k} L^{k(k-1)/4} \right\}$$

但し $L = \mathcal{O}(\log(n\beta/\eta))$

Algorithm Reduce[2]

Require: 基底行列 $\boldsymbol{B} \in M_n(\mathbb{Z})$，$0 < \eta \leq 1$
Ensure: 簡約された基底 $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$

$n_r \leftarrow n$ 以下最大の specific dimension
computes GSO $B \leftarrow [\|\boldsymbol{b}_i\|^\star], \boldsymbol{U} \leftarrow [\mu_{i,j}]$
`sizeReduce`$(\boldsymbol{B})$
$\overline{\beta} \leftarrow 2 + \lceil \log_2 (\sqrt{n}(\max_{i \leq n} \|\boldsymbol{b}_i\|)^n) \rceil$
for $t = 1$ to $(n/n_r)^2 \lceil \log(\overline{\beta}/\eta) \rceil$ **do**
    for $k = 0$ to $n - n_r$ **do**
        $B' \leftarrow B_{[k+1, n_r+k]}$
        $\boldsymbol{U}' \leftarrow \boldsymbol{U}_{[k+1, n_r+k]}$
        $\boldsymbol{V} \leftarrow$ `recReduce`$(B', \boldsymbol{U}', \eta, \overline{\beta})$
        $\boldsymbol{V}' \leftarrow \boldsymbol{E}_n$
        for $k + 1 \leq i, j \leq k + n_r$ **do**
            $v'_{i,j} \leftarrow v_{i-k, j-k}$
        $\boldsymbol{B} \leftarrow \boldsymbol{V}'\boldsymbol{B}$
        `sizeReduce`$(\boldsymbol{B})$

Algorithm Recursive Reduce[2]

Require: $B, \boldsymbol{U}, \eta, \overline{\beta}$
Ensure: 簡約された基底 $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$

if $n = 2$ **then** Schonhage のアルゴリズム [3] **endif**
$\boldsymbol{V} \leftarrow \boldsymbol{E}_n$
for $c = 1$ to$(2n_r/n_{r-1})^2 \lceil \log(8\overline{\beta}/\eta) \rceil$ **do**
    for $k = 2(n_r/n_{r-1})^2$ downto 0 **do**
        $t \leftarrow kn_{r-1}/2$
        $B' \leftarrow B_{[t+1, t+n_{r-1}]}$
        $\boldsymbol{U}' \leftarrow \boldsymbol{U}_{[t+1, t+n_{r-1}]}$
        $\boldsymbol{V}' =$ `recReduce`$(B', \boldsymbol{U}', \eta, \overline{\beta})$
        $\boldsymbol{V}'' \leftarrow \boldsymbol{E}_{n_r}$
        for $t + 1 \leq i, j \leq t + n_{r-1}$ **do**
            $v'_{i,j} \leftarrow v_{i-t, j-t}$
        $\boldsymbol{V} \leftarrow \boldsymbol{V}''\boldsymbol{V} \bmod 2^{\overline{\beta}}$
        `sizeReduce`$(B, \boldsymbol{U})$

Seysen 簡約（1/3）

- $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$: 基底
- $\boldsymbol{B} = (\boldsymbol{b}_1^\top, \ldots, \boldsymbol{b}_n^\top)^\top$: 基底行列
- $\boldsymbol{B} = \boldsymbol{R}\boldsymbol{Q}$ ($\boldsymbol{R}$： 下三角行列, $\boldsymbol{Q}$： 直交行列)
- $\boldsymbol{R} = \begin{bmatrix} \boldsymbol{R}_{1,1} & \boldsymbol{O}_{\lfloor n/2 \rfloor, n-\lfloor n/2 \rfloor} \\ \boldsymbol{R}_{2,1} & \boldsymbol{R}_{2,2} \end{bmatrix}$

定義 2 (Seysen 簡約)

$\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ が Seysen 簡約されているとは

$$n = 1 \vee \left(\boldsymbol{R}_{1,1},\ \boldsymbol{R}_{2,2}\text{が Seysen 簡約されている} \wedge \left\| \boldsymbol{R}_{2,1} \boldsymbol{R}_{1,1}^{-1} \right\|_{\max} \leq \frac{1}{2} \right)$$

Seysen 簡約（2/3）

Seysen 簡約は大まかには次のようなことを $\boldsymbol{B} = \boldsymbol{RQ}$ なる $\boldsymbol{R}$ に対して再帰的に行う．

1. $\boldsymbol{R}$ が 1 次正方なら何もしない
2. $\boldsymbol{R}$ を $\begin{bmatrix} \boldsymbol{R}_{1,1} & \boldsymbol{O} \\ \boldsymbol{R}_{2,1} & \boldsymbol{R}_{2,2} \end{bmatrix}$ とブロックに分ける
3. $\boldsymbol{R}_{1,1}, \boldsymbol{R}_{2,2}$ をそれぞれ Seysen 簡約

Seysen 簡約（3/3）

Algorithm Seysen 簡約 [1]

Require: 下三角行列 $\boldsymbol{R} \in M_n(\mathbb{R})$ s.t. $\boldsymbol{B} = \boldsymbol{R}\boldsymbol{Q}$
Ensure: $\boldsymbol{U}\boldsymbol{B}$ が簡約基底行列となるような unimodular 行列 $\boldsymbol{U} \in M_n(\mathbb{R})$

if $\boldsymbol{R} \in M_1(\mathbb{R})$ **then**
  return $[1]$
$\begin{bmatrix} \boldsymbol{R}_{1,1} & \boldsymbol{O} \\ \boldsymbol{R}_{2,1} & \boldsymbol{R}_{2,2} \end{bmatrix} \leftarrow \boldsymbol{R}$ /* $\boldsymbol{R}_{1,1} \in M_{\lfloor n \rfloor}(\mathbb{R})$, $\boldsymbol{R}_{2,1} \in M_{n-\lfloor n \rfloor, \lfloor n \rfloor}(\mathbb{R})$, $\boldsymbol{R}_{2,2} \in M_{n-\lfloor n \rfloor, n-\lfloor n \rfloor}(\mathbb{R})$ */
$\boldsymbol{U}_{1,1} \leftarrow \texttt{seysenReduce}(\boldsymbol{R}_{1,1})$; $\boldsymbol{U}_{2,2} \leftarrow \texttt{seysenReduce}(\boldsymbol{R}_{2,2})$
$\boldsymbol{R}_{2,1} \leftarrow \boldsymbol{U}_{2,2}\boldsymbol{R}_{2,1}$
$\boldsymbol{U}_{2,1} \leftarrow \left\lfloor -\boldsymbol{R}_{2,1}\boldsymbol{R}_{1,1}^{-1} \right\rceil$
$\boldsymbol{R}_{2,1} \leftarrow \boldsymbol{U}_{2,1}\boldsymbol{R}_{1,1} + \boldsymbol{R}_{2,1}$
return $\begin{bmatrix} \boldsymbol{U}_{1,1} & \boldsymbol{O} \\ \boldsymbol{U}_{2,1}\boldsymbol{U}_{1,1} & \boldsymbol{U}_{2,2} \end{bmatrix}$

BLASter LLL(1/3)

weakly-LLL 簡約

定義 3 (weakly-LLL 簡約基底)

$\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ が weakly-LLL 簡約されているとは

$$\|\boldsymbol{b}_k^\star\|^2 \geq (\delta - \mu_{k,k-1}^2)\left\|\boldsymbol{b}_{k-1}^\star\right\|^2$$

なるときをいう.

- Lovász 条件のみを満たす

BLASter LLL(2/3)

BLASter LLL では，大まかに次のことを行っている．

- **$B = RQ$ と下三角行列と直交行列の積に分解**
  - [1] では，`Numpy` の組み込み関数 `numpy.linalg.qr` を利用
- **B をブロックに分けて LLL を施す**
  - ブロックはすべて disjoint なので，並列処理
- **$B = RQ$ と下三角行列と直交行列の積に分解**
  - R, Q の更新より再計算の方が数値的に安定
- **Seysen アルゴリズムを用いてサイズ簡約**

Algorithm BLASter LLL アルゴリズム [1]

Require: 下三角行列 $\boldsymbol{R} \in M_n(\mathbb{R})$ s.t. $\boldsymbol{B} = \boldsymbol{RQ}$
Ensure: $\boldsymbol{UB}$ が簡約基底行列となるような unimodular 行列 $\boldsymbol{U} \in M_n(\mathbb{R})$

$i_0 \leftarrow 0$; $\boldsymbol{U} \leftarrow \boldsymbol{E}_n$
do
  $\boldsymbol{R} \leftarrow \boldsymbol{B} = \boldsymbol{RQ}$ なる下三角行列 $\boldsymbol{R}$ ($\boldsymbol{Q}$: 直交行列)
  $i_0 \leftarrow \ell/2 - i_0$
  $\mathcal{I} \leftarrow \{(i_0 + k\ell + 1, \min\{n, i_0 + k\ell + \ell\}) \mid 0 \le k < (n - i_0)/\ell\}$
  for $(i, j) \in \mathcal{I}$ **do** $\boldsymbol{V}_i \leftarrow \mathrm{LLL}(\boldsymbol{R}_{[i,j]\times[i,j]}, \delta)$ **endfor** /* $\boldsymbol{V}_i$ は LLL 基底を与えるユニモジラ行列 */
  for $(i, j) \in \mathcal{I}$ **do**
    $\boldsymbol{B}_{[i,j]\times[1,n]} \leftarrow \boldsymbol{V}_i \boldsymbol{B}_{[i,j]\times[1,n]}$; $\boldsymbol{U}_{[i,j]\times[1,n]} \leftarrow \boldsymbol{V}_i \boldsymbol{U}_{[i,j]\times[1,n]}$
  $\boldsymbol{R} \leftarrow \boldsymbol{B} = \boldsymbol{RQ}$ なる下三角行列 $\boldsymbol{R}$ ($\boldsymbol{Q}$: 直交行列)
  $\boldsymbol{W} \leftarrow \mathtt{seysenReduce}(\boldsymbol{R})$; $\boldsymbol{B} \leftarrow \boldsymbol{WB}$; $\boldsymbol{U} \leftarrow \boldsymbol{WU}$
  $f \leftarrow \mathtt{true}$ /* $\boldsymbol{B}$ が weakly-LLL 簡約されているか */
  for $i = 1 \mathrm{to} n - 1$ **do**
    if $\delta r_{i,i}^2 > r_{i,i+1}^2 + r_{i+1,i+1}^2$ **then** $f \leftarrow \mathtt{false}$; **break endif**
while not f

BLASter BKZ（1/2）

BLASter BKZ では大まかに次のことを行う

- B **を下三角行列と直交行列の積に分解**
- B **をブロックに分けて BKZ を施す**
- **BLASter DeepLLL を用いて簡約**

Algorithm BLASter BKZ アルゴリズム [1]

Require: 下三角行列 $\boldsymbol{R} \in M_n(\mathbb{R})$ s.t. $\boldsymbol{B} = \boldsymbol{R}\boldsymbol{Q}$
Ensure: $\boldsymbol{U}\boldsymbol{B}$ が簡約基底行列となるような unimodular 行列 $\boldsymbol{U} \in M_n(\mathbb{R})$

if $n \leq 40$ **then return** $\texttt{BLASterDeepLLL}(\boldsymbol{B}, \delta, \ell, 4)$
$i_0 \leftarrow 0$; $(\boldsymbol{B}, \boldsymbol{U}) \leftarrow \texttt{BLASterBKZ}(\boldsymbol{B}, \delta, \ell, \ell', \beta - P, 1, P)$
while $t > 0$ **do**
    $\boldsymbol{R} \leftarrow \boldsymbol{B} = \boldsymbol{R}\boldsymbol{Q}$ なる下三角行列 $\boldsymbol{R}$ ($\boldsymbol{Q}$: 直交行列)
    $i_0 \leftarrow i_0 \bmod \ell'$
    $\mathcal{I} \leftarrow \{(i'_0 + k\ell' + 1, \min\{n, i'_0 + k\ell' + \ell'\}) \mid 0 \leq k < (n - i'_0)/\ell'\}$ **for** $(i, j) \in \mathcal{I}$ **do** $\boldsymbol{V}_i \leftarrow \texttt{BKZReduce}(\boldsymbol{R}_{[i,j]\times[i,j]}, \delta, \beta)$
    for $(i, j) \in \mathcal{I}$ **do**
        $\boldsymbol{B}_{[i,j]\times[1,n]} \leftarrow \boldsymbol{V}_i \boldsymbol{B}_{[i,j]\times[1,n]}$; $\boldsymbol{U}_{[i,j]\times[1,n]} \leftarrow \boldsymbol{V}_i \boldsymbol{U}_{[i,j]\times[1,n]}$
    if $i_0 + \beta > n$ **then** $i_0 \leftarrow 0$; $t \leftarrow t - 1$ **else** $i_0 \leftarrow i_0 + \ell' - \beta + 1$
    $(\boldsymbol{B}, \boldsymbol{V}) \leftarrow \texttt{BLASterDeepLLL}(\boldsymbol{B}, \delta, \ell, 4)$
    $\boldsymbol{U} \leftarrow \boldsymbol{V}\boldsymbol{U}$

参考文献 I

[1] Léo Ducas, Ludo N. Pulles, and Marc Stevens. Towards a modern LLL implementation. Cryptology ePrint Archive, Paper 2025/774, 2025.

[2] Arnold Neumaier and Damien Stehle. Faster LLL-type reduction of lattice bases. Cryptology ePrint Archive, Paper 2016/852, 2016.

[3] Arnold Schönhage. Fast reduction and composition of binary quadratic forms. In Proceedings of the 1991 International Symposium on Symbolic and Algebraic Computation, ISSAC '91, page 128–133. Association for Computing Machinery, 1991.