# Lenstra–Lenstra–Lovász lattice basis reduction algorithm

The **Lenstra–Lenstra–Lovász** (LLL) **lattice basis reduction algorithm** is a polynomial time lattice reduction algorithm invented by Arjen Lenstra, Hendrik Lenstra and László Lovász in 1982.[1] Given a basis $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_d\}$ with *n*-dimensional integer coordinates, for a lattice L (a discrete subgroup of $\mathbf{R}^n$) with $d \leq n$, the LLL algorithm calculates an *LLL-reduced* (short, nearly orthogonal) lattice basis in time

$$\mathcal{O}(d^5 n \log^3 B)$$

where $B$ is the largest length of $\mathbf{b}_i$ under the Euclidean norm, that is, $B = \max\left(\|\mathbf{b}_1\|_2, \|\mathbf{b}_2\|_2, \dots, \|\mathbf{b}_d\|_2\right)$.[2][3]

The original applications were to give polynomial-time algorithms for factorizing polynomials with rational coefficients, for finding simultaneous rational approximations to real numbers, and for solving the integer linear programming problem in fixed dimensions.

## LLL reduction

The precise definition of LLL-reduced is as follows: Given a basis

$$\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n\},$$

define its Gram–Schmidt process orthogonal basis

$$\mathbf{B}^* = \{\mathbf{b}_1^*, \mathbf{b}_2^*, \dots, \mathbf{b}_n^*\},$$

and the Gram-Schmidt coefficients

$$\mu_{i,j} = \frac{\langle \mathbf{b}_i, \mathbf{b}_j^* \rangle}{\langle \mathbf{b}_j^*, \mathbf{b}_j^* \rangle},$$

for any $1 \leq j < i \leq n$.

Then the basis $B$ is LLL-reduced if there exists a parameter $\delta$ in $(0.25, 1]$ such that the following holds:

1. (size-reduced) For $1 \leq j < i \leq n$: $|\mu_{i,j}| \leq 0.5$. By definition, this property guarantees the length reduction of the ordered basis.
2. (Lovász condition) For k = 2,3,..,n : $\delta \|\mathbf{b}_{k-1}^*\|^2 \leq \|\mathbf{b}_k^*\|^2 + \mu_{k,k-1}^2 \|\mathbf{b}_{k-1}^*\|^2$.

Here, estimating the value of the $\delta$ parameter, we can conclude how well the basis is reduced. Greater values of $\delta$ lead to stronger reductions of the basis. Initially, A. Lenstra, H. Lenstra and L. Lovász demonstrated the LLL-reduction algorithm for $\delta = \frac{3}{4}$. Note that although LLL-reduction is well-defined for $\delta = 1$, the polynomial-time complexity is guaranteed only for $\delta$ in $(0.25, 1)$.

The LLL algorithm computes LLL-reduced bases. There is no known efficient algorithm to compute a basis in which the basis vectors are as short as possible for lattices of dimensions greater than 4.[4] However, an LLL-reduced basis is nearly as short as possible, in the sense that there are absolute bounds $c_i > 1$ such that the first basis vector is no more than $c_1$ times as long as a shortest vector in the lattice, the second basis vector is likewise within $c_2$ of the second successive minimum, and so on.

# Applications

An early successful application of the LLL algorithm was its use by Andrew Odlyzko and Herman te Riele in disproving Mertens conjecture.[5]

The LLL algorithm has found numerous other applications in MIMO detection algorithms[6] and cryptanalysis of public-key encryption schemes: knapsack cryptosystems, RSA with particular settings, NTRUEncrypt, and so forth. The algorithm can be used to find integer solutions to many problems.[7]

In particular, the LLL algorithm forms a core of one of the integer relation algorithms. For example, if it is believed that *r*=1.618034 is a (slightly rounded) root to an unknown quadratic equation with integer coefficients, one may apply LLL reduction to the lattice in $\mathbf{Z}^4$ spanned by $[1, 0, 0, 10000r^2], [0, 1, 0, 10000r]$, and $[0, 0, 1, 10000]$. The first vector in the reduced basis will be an integer linear combination of these three, thus necessarily of the form $[a, b, c, 10000(ar^2 + br + c)]$; but such a vector is "short" only if *a*, *b*, *c* are small and $ar^2 + br + c$ is even smaller. Thus the first three entries of this short vector are likely to be the coefficients of the integral quadratic polynomial which has *r* as a root. In this example the LLL algorithm finds the shortest vector to be [1, -1, -1, 0.00025] and indeed $x^2 - x - 1$ has a root equal to the golden ratio, 1.6180339887....

# Properties of LLL-reduced basis

Let $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n\}$ be a $\delta$-LLL-reduced basis of a lattice $\mathcal{L}$. From the definition of LLL-reduced basis, we can derive several other useful properties about $\mathbf{B}$.

1. The first vector in the basis cannot be much larger than the shortest non-zero vector: $\|\mathbf{b}_1\| \le (2/(\sqrt{4\delta - 1}))^{n-1} \cdot \lambda_1(\mathcal{L})$. In particular, for $\delta = 3/4$, this gives $\|\mathbf{b}_1\| \le 2^{(n-1)/2} \cdot \lambda_1(\mathcal{L})$.[8]
2. The first vector in the basis is also bounded by the determinant of the lattice: $\|\mathbf{b}_1\| \le (2/(\sqrt{4\delta - 1}))^{(n-1)/2} \cdot (\det(\mathcal{L}))^{1/n}$. In particular, for $\delta = 3/4$, this gives $\|\mathbf{b}_1\| \le 2^{(n-1)/4} \cdot (\det(\mathcal{L}))^{1/n}$.
3. The product of the norms of the vectors in the basis cannot be much larger than the determinant of the lattice: let $\delta = 3/4$, then $\prod_{i=1}^n \|\mathbf{b}_i\| \le 2^{n(n-1)/4} \cdot \det(\mathcal{L})$.

# LLL algorithm pseudocode

The following description is based on (Hoffstein, Pipher & Silverman 2008, Theorem 6.68), with the corrections from the errata.[9]

```
INPUT
    a lattice basis b0, b1, ..., bn in Z^m
    a parameter δ with 1/4 < δ < 1, most commonly δ = 3/4
PROCEDURE
```

```
    B* <- GramSchmidt({b0, ..., bn}) = {b0*, ..., bn*};  and do not normalize
    μi,j <- InnerProduct(bi, bj*)/InnerProduct(bj*, bj*);   using the most current values of bi
and bj*
    k <- 1;
    while k <= n do
        for j from k−1 to 0 do
            if |μk,j| > 1/2 then
                bk <- bk − ⌊μk,j⌉bj;
               Update B* and the related μi,j's as needed.
               (The naive method is to recompute B* whenever bi changes:
                B* <- GramSchmidt({b0, ..., bn}) = {b0*, ..., bn*})
            end if
        end for
        if InnerProduct(bk*, bk*) > (δ − μ2k,k−1) InnerProduct(bk−1*, bk−1*) then
            k <- k + 1;
        else
            Swap bk and  bk−1;
            Update B* and the related μi,j's as needed.
            k <- max(k−1, 1);
        end if
    end while
    return B the LLL reduced basis of {b0, ..., bn}
OUTPUT
    the reduced basis b0, b1, ..., bn in Zm
```

# Examples

## Example from $Z^3$

Let a lattice basis $\mathbf{b_1}, \mathbf{b_2}, \mathbf{b_3} \in \mathbf{Z}^3$, be given by the columns of

$$\begin{bmatrix} 1 & -1 & 3 \\ 1 & 0 & 5 \\ 1 & 2 & 6 \end{bmatrix}$$

then the reduced basis is

$$\begin{bmatrix} 0 & 1 & -1 \\ 1 & 0 & 0 \\ 0 & 1 & 2 \end{bmatrix},$$

which is size-reduced, satisfies the Lovász condition, and is hence LLL-reduced, as described above. See W. Bosma.[10] for details of the reduction process.

## Example from $Z[i]^4$

Likewise, for the basis over the complex integers given by the columns of the matrix below,

$$\begin{bmatrix} -2+2i & 7+3i & 7+3i & -5+4i \\ 3+3i & -2+4i & 6+2i & -1+4i \\ 2+2i & -8+0i & -9+1i & -7+5i \\ 8+2i & -9+0i & 6+3i & -4+4i \end{bmatrix},$$

then the columns of the matrix below give an LLL-reduced basis.

$$\begin{bmatrix} -6+3i & -2+2i & 2-2i & -3+6i \\ 6-1i & 3+3i & 5-5i & 2+1i \\ 2-2i & 2+2i & -3-1i & -5+3i \\ -2+1i & 8+2i & 7+1i & -2-4i \end{bmatrix}.$$

## Implementations

LLL is implemented in

- Arageli (http://www.arageli.org/) as the function `lll_reduction_int`
- fpLLL (https://github.com/fplll/fplll) as a stand-alone implementation
- FLINT as the function `fmpz_lll`
- GAP as the function `LLLReducedBasis`
- Macaulay2 as the function `LLL` in the package `LLLBases`
- Magma as the functions `LLL` and `LLLGram` (taking a gram matrix)
- Maple as the function `IntegerRelations[LLL]`
- Mathematica as the function `LatticeReduce`
- Number Theory Library (NTL) (https://github.com/libntl/ntl) as the function `LLL`
- PARI/GP as the function `qflll`
- Pymatgen (http://pymatgen.org/) as the function `analysis.get_lll_reduced_lattice`
- SageMath as the method `LLL` driven by fpLLL and NTL
- Isabelle/HOL in the 'archive of formal proofs' entry `LLL_Basis_Reduction`. This code exports to efficiently executable Haskell.[11]

## See also

- Coppersmith method

## Notes

1. Lenstra, A. K.; Lenstra, H. W., Jr.; Lovász, L. (1982). "Factoring polynomials with rational coefficients". *Mathematische Annalen*. **261** (4): 515–534. CiteSeerX 10.1.1.310.318 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.310.318). doi:10.1007/BF01457454 (https://doi.org/10.1007%2FBF01457454). hdl:1887/3810 (https://hdl.handle.net/1887%2F3810). MR 0682664 (https://mathscinet.ams.org/mathscinet-getitem?mr=0682664). S2CID 5701340 (https://api.semanticscholar.org/CorpusID:5701340).
2. Galbraith, Steven (2012). "chapter 17" (https://www.math.auckland.ac.nz/~sgal018/crypto-book/crypto-book.html). *Mathematics of Public Key Cryptography*.
3. Nguyen, Phong Q.; Stehlè, Damien (September 2009). "An LLL Algorithm with Quadratic Complexity" (https://dl.acm.org/citation.cfm?id=1655318). *SIAM J. Comput*. **39** (3): 874–903. doi:10.1137/070705702 (https://doi.org/10.1137%2F070705702). Retrieved 3 June 2019.

4. Nguyen, Phong Q.; Stehlé, Damien (1 October 2009). "Low-dimensional lattice basis reduction revisited". *ACM Transactions on Algorithms*. **5** (4): 1–48. doi:10.1145/1597036.1597050 (https://doi.org/10.1145%2F1597036.1597050). S2CID 10583820 (https://api.semanticscholar.org/CorpusID:10583820).
5. Odlyzko, Andrew; te Reile, Herman J. J. "Disproving Mertens Conjecture" (http://www.dtc.umn.edu/~odlyzko/doc/arch/mertens.disproof.pdf) (PDF). *Journal für die reine und angewandte Mathematik*. **357**: 138–160. doi:10.1515/crll.1985.357.138 (https://doi.org/10.1515%2Fcrll.1985.357.138). S2CID 13016831 (https://api.semanticscholar.org/CorpusID:13016831). Retrieved 27 January 2020.
6. D. Wübben et al., "Lattice reduction," IEEE Signal Processing Magazine, Vol. 28, No. 3, pp. 70-91, Apr. 2011.
7. D. Simon (2007). "Selected applications of LLL in number theory" (https://simond.users.lmno.cnrs.fr/maths/lll25_Simon.pdf) (PDF). *LLL+25 Conference*. Caen, France.
8. Regev, Oded. "Lattices in Computer Science: LLL Algorithm" (https://cims.nyu.edu/~regev/teaching/lattices_fall_2004/ln/lll.pdf#page=3) (PDF). New York University. Retrieved 1 February 2019.
9. Silverman, Joseph. "Introduction to Mathematical Cryptography Errata" (http://www.math.brown.edu/~jhs/MathCrypto/MathCryptoErrata.pdf) (PDF). *Brown University Mathematics Dept*. Retrieved 5 May 2015.
10. Bosma, Wieb. "4. LLL" (http://www.math.ru.nl/~bosma/onderwijs/voorjaar07/compalg7.pdf) (PDF). *Lecture notes*. Retrieved 28 February 2010.
11. Divasón, Jose (2018). "A Formalization of the LLL Basis Reduction Algorithm" (https://doi.org/10.1007%2F978-3-319-94821-8_10). *Conference Paper*. Lecture Notes in Computer Science. **10895**: 160–177. doi:10.1007/978-3-319-94821-8_10 (https://doi.org/10.1007%2F978-3-319-94821-8_10). ISBN 978-3-319-94820-1.

# References

- Napias, Huguette (1996). "A generalization of the LLL algorithm over euclidean rings or orders" (http://www.numdam.org/item?id=JTNB_1996__8_2_387_0). *Journal de Théorie des Nombres de Bordeaux*. **8** (2): 387–396. doi:10.5802/jtnb.176 (https://doi.org/10.5802%2Fjtnb.176).
- Cohen, Henri (2000). *A course in computational algebraic number theory*. GTM. Vol. 138. Springer. ISBN 3-540-55640-0.
- Borwein, Peter (2002). *Computational Excursions in Analysis and Number Theory*. ISBN 0-387-95444-9.
- Luk, Franklin T.; Qiao, Sanzheng (2011). "A pivoted LLL algorithm" (https://doi.org/10.1016%2Fj.laa.2010.04.003). *Linear Algebra and Its Applications*. **434** (11): 2296–2307. doi:10.1016/j.laa.2010.04.003 (https://doi.org/10.1016%2Fj.laa.2010.04.003).
- Hoffstein, Jeffrey; Pipher, Jill; Silverman, J.H. (2008). *An Introduction to Mathematical Cryptography*. Springer. ISBN 978-0-387-77993-5.