

# From an LLL-reduced Basis to another

Ivan Morel, Damien Stehlé, and Gilles Villard

ARENAIRE project, LIP  
ÉNS LYON, 46, allée d'Italie  
69364 Lyon Cedex 07,  
ivan.morel@ens-lyon.fr,  
home page: <http://perso.ens-lyon.fr/ivan.morel/>

Lattice reduction has important applications in very diverse fields of both mathematics and computer science: computer algebra, cryptology, algorithmic number theory, algorithmic group theory, MIMO communications, computer arithmetic, etc. The LLL algorithm allows one to reduce a basis to a 'good basis' (as defined by Lovász in [1]) in polynomial time. However the quality of the reduction obtained is directly related to the parameter  $\delta$  (the 3/4 factor in the original algorithm) defined below. For the purpose of this study we introduce a definition of reduction that is slightly weaker than the one described in the original article [1]:

**Definition 1** ( $(\delta, \eta, \epsilon)$ -LLL reduced basis)

Considering a basis  $(b_1, \dots, b_d)$ , let  $R$  be the  $R$ -factor of the QR-factorisation of the matrix whose columns are the  $b_i$ 's. The basis is called  $(\delta, \eta, \epsilon)$ -LLL-reduced ( $\eta \in [1/2; 1]$ ,  $\delta \in ]\eta^2; 1[$ , and  $\epsilon \geq 0$ ) if :

- $|R_{i,j}| \leq \eta R_{i,i} + \epsilon R_{j,j}$  for  $i < j$  ( $(\eta, \epsilon)$ -size reduction),
- $\delta R_{i,i}^2 \leq R_{i,i+1}^2 + R_{i+1,i+1}^2$ .

The original definition of size-reduction corresponds to the case where  $\epsilon = 0$ .

As mentioned by Cohen [2], it was suggested by LaMacchia in [3] that one could reduce the basis using  $\delta$  as small as possible (so that the reduction is as fast as possible), and then increase the value of  $\delta$  to reach the maximum quality. We present here an algorithm that improves the quality of an LLL-reduced basis by increasing the value of  $\delta$  with complexity being only a function of the dimension and total exponent the same as LLL.

More precisely we consider the problem of LLL-reducing a basis  $(b_1, \dots, b_d)$  (which we will from now on identify to the matrix  $A$  whose columns are the  $b_i$ 's) that is already LLL-reduced with a weaker  $\delta$ . The fact that  $A$  is already LLL-reduced gives us a lot of information about the lattice: the  $\|b_i\|$ 's are relatively close to the  $R_{i,i}$ 's and the  $\lambda_i$ 's (successive minima of the lattice), therefore it is possible to bound the LLL-potential ( $\prod R_{i,i}^{2 \cdot (d-i)}$ ) and to prove that there are no more than  $O(d^3)$  LLL-exchanges (instead of  $O(d^2 \cdot \log(B))$  for the usual analysis,  $B$  being a bound on the coefficients of the  $b_i$ 's). The contribution of  $B$  towards the total cost of the algorithm is lowered, and it is reasonable to think that it is possible to completely eliminate that contribution. One way to address this is to approximate the initial basis using the minimal precision required to reduce the basis, as it is done in [4]. Furthermore, in our case, the structure of the initial basis guarantees us that this precision is much smaller than the one presented in [4].

Thus we propose the following approach to reduce the basis  $A = (b_1, \dots, b_d)$ :

$$\begin{array}{ccc} A & \xrightarrow{1} & \tilde{A} \\ & & \downarrow 2 \\ AU & \xleftarrow{3} & \tilde{A}U \end{array}$$

1. At this step the goal is to approximate  $A$  by a basis  $\tilde{A}$  that can be coded on fewer bits than  $A$  while keeping as many properties of  $A$  as possible (Using a perturbation analysis of the  $QR$ -decomposition [5], it is possible to prove that only so many bits from the original matrix are needed to roughly conserve reduction in the sense of definition 1).
2. We reduce here the basis  $\tilde{A}$  and store the transformations applied to the basis in a unimodular matrix  $U$ .
3. We then apply the unimodular matrix  $U$  to the original matrix  $A$ .

The main issue of this method is to prove whether the final basis  $AU$  is LLL-reduced or not. In order to use the  $QR$  perturbation analysis [5], we need  $\tilde{A}U$  to be close enough of  $AU$ . This is directly linked to the size of  $U$ .

If both  $\tilde{A}$  and  $\tilde{A}U$  are LLL-reduced, it is possible to link the size of  $U$  to the highest ratio between the  $\lambda_i$ 's. Therefore when  $\|b_i\| > 2^{-O(d)} \max(\|b_j\|)$ , as we can prove that keeping  $O(d)$  significant bits from  $A$  to form  $\tilde{A}$  conserves LLL-reduction, the size of  $U$  can be bounded by  $2^{O(d)}$ . Therefore  $\tilde{A}U$  is close enough to  $AU$  to use the perturbation analysis. Reducing  $A$  with a higher  $\delta$  therefore is equivalent to reducing  $\tilde{A}$ , which has entry size  $O(d)$  (all entries have roughly the same number of bits and the same number of significant bits, and can therefore be scaled).

In the general case, the same method cannot be applied directly as there may be entries with completely different sizes. As the initial basis is LLL-reduced, it is however possible to manipulate the basis in a way that keeps its structure intact and reduce the overall differences in size of the entries. A form of adaptable scaling allows us to reduce the gaps between increasing  $R_{i,i}$ 's without perturbing the march of the LLL algorithm would have had on the original matrix: we create a matrix  $\tilde{A}$  close to  $A$  with entries having  $O(d)$  significant bits, and then we scale each column of  $\tilde{A}$  such that the resulting matrix  $\tilde{A}$  has the same structure than  $A$  and smaller entries, and can therefore be reduced faster. Then post-processing the output unimodular matrix  $U$  gives the matrix  $U'$  that is to be applied to the original matrix  $A$ . It is then possible to prove that  $\tilde{A}U'$  satisfies reduction in the sense of definition 1, which is enough to prove that  $AU'$  is also reduced [5].

## References

1. A.K. Lenstra, H.W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.
2. H. Cohen. *A course in computational algebraic number theory*, volume 138 of *Graduate Texts in Mathematics*. Springer-Verlag, Berlin, 1993.
3. B.A. LaMacchia. Basis Reduction Algorithms and Subset Sum Problems. SM Thesis, Dept. of Elect. Eng. and Comp. Sci., Massachusetts Institute of Technology, Cambridge, MA, 1991.
4. J. Buchmann. Reducing Lattice Bases by Means of Approximations. *First International Symposium ANTS, Springer Lecture Notes in Computer Science*, 877:160–168, 1994.
5. X.W. Chang, D. Stehlé, and G. Villard. Explicit Perturbation Analysis of the R-Factor of the QR Factorisation in the Context of LLL-Reduction. *work in progress*.