Vector Rational Number Reconstruction
Version 2

Curtis Bright

August 28, 2009
Rational Number Reconstruction

- Given an integer residue \( a \in \mathbb{Z}_M \) and a size bound \( N \), the rational number reconstruction problem is to solve

\[
da \equiv n \pmod{M}, \quad d, n \leq N
\]

for \( d, n \in \mathbb{Z} \).
• If $M > 2N^2$ then there is at most one rational number $n/d$ solution.

• For example, consider $a = 25 \in \mathbb{Z}_{97}$ and $N = 6$.

> iratrecon(25, 97);

    3/4

• Lo and behold, $4 \cdot 25 \equiv 3 \pmod{97}$. 
Vector Rational Number Reconstruction

- Given an integer residue vector \( \mathbf{a} \in \mathbb{Z}_M^n \) and a size bound \( N \), the vector rational number reconstruction problem is to solve

\[
d\mathbf{a} \equiv \mathbf{n} \pmod{M}, \quad \| [d \ | \ n] \| \leq N
\]

for \( d \in \mathbb{Z} \) and \( n \in \mathbb{Z}^n \).

- For example, consider

\[
\mathbf{a} = \begin{bmatrix} -23677 & -49539 & 74089 & -21989 & 63531 \end{bmatrix} \in \mathbb{Z}_{195967}^5
\]

and \( N = 10^4 \).
• This has the unique nonzero solution

\[ d = 3137 \quad \text{and} \quad \mathbf{n} = [-3256, -2012, 331, 891, -1692], \]

i.e.,

\[ \mathbf{a} \equiv [-3256, -2012, 331, 891, -1692] / 3137 \pmod{195967}. \]

• Even though the solution is unique, Maple can’t find it because \( M \) isn’t sufficiently larger than \( N \) to ensure entrywise uniqueness.
\[
\text{\texttt{a := \{-23677, -49539, 74089, -21989, 63531\}}:}
\]
\[
\text{\texttt{map(iratrecon, a, 195967);}}
\]
\[
\begin{array}{ll}
-235 & 211 \\
\text{[FAIL, ----, FAIL, ---, FAIL]}
\end{array}
\]
\[
\begin{array}{ll}
269 & 303 \\
\end{array}
\]

\[
\text{\texttt{map(iratrecon, a, 195967, 3256, 3137);}}
\]
\[
\begin{array}{llll}
2527 & -2245 & -957 \\
\text{[----, -2189/4, -----, -1934/9, ----]}
\end{array}
\]
\[
\begin{array}{ll}
33 & 37 \\
37 & 37 \\
\end{array}
\]

- Finding a common denominator, we see that

\[
\text{\texttt{a \equiv \{-53814, 16340, 90815, -13080, 12962\} / 14652 \ (mod 195967),}}
\]

but this solution vector has norm greater than $10^5$, and we wanted one less than $10^4$. 
Lattices

- Given a set of vectors, the lattice generated by them is the set of all integer linear combinations of those vectors:

- A set of linearly independent vectors which generate the same lattice is known a basis of the lattice.
Lattice Bases

- Not all bases are created equal, some have needlessly long vectors:

![Diagram showing lattice points and vectors](image-url)
• Many problems, including rational reconstruction, can be posed in the form, ‘given this lattice basis with long vectors, find a short nonzero vector in the lattice’.

• The LLL Algorithm finds a vector within a factor of $2^d$ of the shortest nonzero vector in a $d$-dimensional lattice, and it runs in polynomial time in $d$. 
LLL Algorithm

- LLL is based around the concept of size reduction of a vector $b_i$ against a set of vectors $b_1, b_2, \ldots, b_{i-1}$.
- To do the size reduction against $b_j$, we replace $b_i$ with

$$b_i' := b_i - rb_j$$

for the $r \in \mathbb{Z}$ which minimizes $\|\text{proj}_{b_j^*}(b_i')\|$. 

![Diagram showing vector operations related to the LLL algorithm](image)
• We want short vectors in the set we are size reducing against. If we had instead...

\[ b_2 \]

we can’t size reduce \( b_2 \) against \( b_1 \).

• In a case like this we would want to swap \( b_1 \) and \( b_2 \), and then size-reduce.

• Roughly, the Lovász condition is satisfied when \( b_i \) and \( b_{i-1} \) aren’t in a case like this.
LLL Pseudocode

for $i := 2$ to $n$ do
    size reduce $b_i$ against $b_1, b_2, \ldots, b_{i-1}$
    if Lovász condition not satisfied then
        swap $b_{i-1}$ and $b_i$
        $i := \max(i - 2, 1)$

• At the conclusion of the loop, the first $i$ vectors are *LLL reduced.*
Rational Reconstruction: Lattice Reformulation

- Consider the lattice generated by the rows of the following \((n + 1) \times (n + 1)\) integer matrix:

\[
\begin{bmatrix}
  M & & & \\
  & M & & \\
  & & M & \\
  1 & a_1 & a_2 & a_3 & \cdots & a_n
\end{bmatrix}
\]

- For all \(d \in \mathbb{Z}\), the vector

\[
\begin{bmatrix}
  d & d a_1 & d a_2 & d a_3 & \cdots & d a_n
\end{bmatrix}
\]

is in this lattice. Because of the first \(n\) rows,

\[
\begin{bmatrix}
  d & \text{rem}_M(da_1) & \text{rem}_M(da_2) & \text{rem}_M(da_3) & \cdots & \text{rem}_M(da_n)
\end{bmatrix}
\]

is also in this lattice.
• Recall we want to solve

\[ da \equiv n \pmod{M}, \quad \|[d \mid n]\| \leq N \]

for \( d \) and \( n \). Equivalently, we can solve

\[ \|[[d \mid \text{rem}_M(da)]]\| \leq N \]

for \( d \).

• As noted, vectors of this form are in the lattice just considered.
• Therefore, the problem is equivalent to finding vectors shorter than \( N \) in the specific lattice we just saw.
Applying LLL Straightforwardly

• The problem instance we saw previously gives raise to the basis matrix

\[
\begin{bmatrix}
195967 & 195967 & 195967 \\
195967 & 195967 & 195967 \\
1 & -23677 & -49539 74089 -21989 63531
\end{bmatrix}.
\]

• Running LLL on this lattice gives the new basis matrix

\[
\begin{bmatrix}
-3137 & 3256 & 2012 & -331 & -891 & 1692 \\
-3600 & -8445 & 10430 & -9313 & -10268 & -18111 \\
-4047 & -7044 & 10092 & -8673 & 20465 & -1253 \\
241 & -23114 & 15088 & 22452 & -8240 & 25545 \\
28082 & 18517 & 15535 & -14341 & -3081 & -6026 \\
-11836 & 8162 & 10340 & 34921 & 17628 & -27537
\end{bmatrix},
\]

from which the first vector gives a solution to our problem.

• In fact, it is not hard to show that the other vectors do not contribute to a vector shorter than \( N \), using the Gram-Schmidt orthogonalization.
Problems with LLL

- Too expensive; running LLL on the previous lattice requires $O(n^6 \log^3 M)$ bit operations.
- LLL approximation factor is $2^n$, much too large for large $n$. 
Iterative Reduction

- However, the structure of the lattice permits a kind of iterative reduction.

- For example, consider only reducing the lower-left $2 \times 2$ submatrix:

  \[
  \begin{pmatrix}
  0 & 195967 \\
  1 & -23677
  \end{pmatrix}
  \xrightarrow{\text{LLL}}
  \begin{pmatrix}
  -389 & -96 \\
  -149 & 467
  \end{pmatrix}
  \]

- We can use this to help us reduce the lower-left $3 \times 3$ submatrix.
• We can tell what the third column would have been, had we kept it around. Note the third column starts out as $-49539$ times the first column:

$$\begin{bmatrix} 1 & 195967 & -49539 \\ -23677 & 195967 & 1 \\ -49539 & -23677 & 1 \end{bmatrix}$$

and this is always preserved by size reduction and swaps.

• It follows that we have a basis for the lattice generated by the lower-left $3 \times 3$ matrix:

$$\begin{bmatrix} 1 & 195967 & 195967 \\ 1 & 195967 & 195967 \\ 1 & -23677 & -49539 \end{bmatrix} \xleftrightarrow{\text{same lattice}} \begin{bmatrix} -389 & -96 & 195967 \\ -149 & 467 & 19270671 \\ -149 & 467 & 7381311 \end{bmatrix}$$
• We can now run LLL again:

\[
\begin{bmatrix}
-389 & -96 & 195967 \\
-149 & 467 & 7381311
\end{bmatrix}
\xrightarrow{\text{LLL}}
\begin{bmatrix}
-538 & 371 & 470 \\
91 & 1030 & -808 \\
27089 & 13738 & 20045
\end{bmatrix}
\]

• The last vector can now be thrown away, because the last GSO vector has norm larger than \( N \).
Main Contributions

- If $M > 2^{(c+1)/2}N^{1+1/c}$, for $c \in \mathbb{Z}_{>0}$ a small constant which can be chosen, then continuing in this way the row dimension of the matrices will be bounded by $c + 1$.

- For $c = O(1)$ the bit complexity is $O(n^2 \log^3 M)$.

- The column dimension of the matrices are bounded by $n$, but in fact we can get away with only storing the first column. This improves the bit complexity to $O(n \log^3 M)$. 
• The last step of Dixon’s algorithm for linear system solving is to reconstruct a rational vector $\mathbf{x} \in \mathbb{Q}^n$ from its modular image $\text{rem}_M(\mathbf{x})$ when $M = p^i$.
  • Usual elementwise reconstruction requires $i \approx 2 \log N$.
  • This lattice technique requires $i \approx (1 + \frac{1}{c}) \log N$. 