Index Calculus Attack for Hyperelliptic Curves of Small Genus

Nicolas Thériault
nicolast@exp-math.uni-essen.de

University of Toronto /

IEM – Universität Duisburg–Essen

Slide index

The Discrete Log Problem

Let C be a nonsingular hyperelliptic curve of genus g with a single point at infinity defined over the finite field $\mathbb{F}_q$.

Let D_1, D_2 be two elements of $Jac(C)(\mathbb{F}_q)$ such that $D_2 \in \langle D_1 \rangle$.

The ***discrete log problem*** for the pair (D_1, D_2) on $Jac(C)(\mathbb{F}_q)$ consist in computing the smallest integer $\lambda \in \mathbb{N}$ such that

$$D_2 = \lambda D_1 \quad .$$

Hyperelliptic Jacobians

- C is of the form

$$C : Y^2 + h(X)Y = f(X)$$

with $\deg(h) \leq g$ and $\deg(f) = 2g + 1$.

- $Jac(C)(\mathbb{F}_q)$ is the divisor class group, which is isomorphic to the ideal class group.

- $\left(\sqrt{q} - 1\right)^{2g} \leq |Jac(C)(\mathbb{F}_q)| \leq \left(\sqrt{q} + 1\right)^{2g}$, i.e.

$$|Jac(C)(\mathbb{F}_q)| = q^g + O\left(gq^{g-1/2}\right) \quad .$$

- Reduced divisors in $Jac(C)(\mathbb{F}_q)$ can be added in $O\left(g^2(\log q)^2\right)$ bit operations (Cantor).

- To a point $P \in C(\overline{\mathbb{F}_q})$ we associate the divisor
$$D(P) = P - \infty.$$
- Every reduced divisor $D \in Jac(C)(\overline{\mathbb{F}_q})$,
$$D = \sum_{i=1}^{k} D(P_i) \ ,$$
can be represented uniquely by a pair of polynomials $[a(x), b(x)]$, $a(x), b(x) \in \overline{\mathbb{F}}_q[x]$, with
$$a(x) = \prod_{i=1}^{k} (x - x_i) \quad \text{and} \quad b(x_i) = y_i$$
such that $\deg(b) < \deg(a)$ and $a(x)$ divides $b(x)^2 + h(x)b(x) - f(x)$.

Hyperelliptic Jacobians

- A reduced divisor $D = [a(x), b(x)]$ is in $Jac(C)(\mathbb{F}_q)$ if and only if $a(x), b(x) \in \mathbb{F}_q[x]$.

Hyperelliptic Jacobians

- A reduced divisor $D = [a(x), b(x)]$ is in $Jac(C)(\mathbb{F}_q)$ if and only if $a(x), b(x) \in \mathbb{F}_q[x]$.

- To know if the points P_i associated to a reduced divisor are in $C(\mathbb{F}_q)$, we can check if $a(x)$ splits completely in $\mathbb{F}_q[x]$.

- To find the points P_i associated to a reduced divisor, we need to completely factor $a(x)$.

Hyperelliptic Jacobians

- A reduced divisor $D = [a(x), b(x)]$ is in $Jac(C)(\mathbb{F}_q)$ if and only if $a(x), b(x) \in \mathbb{F}_q[x]$.
- To know if the points P_i associated to a reduced divisor are in $C(\mathbb{F}_q)$, we can check if $a(x)$ splits completely in $\mathbb{F}_q[x]$.
- To find the points P_i associated to a reduced divisor, we need to completely factor $a(x)$.
- $D(-P) = -D(P)$.

Generic attacks

- Three main types of attack:
  - Shank's Baby Step - Giant Step algorithm;
  - Pollard's ρ method;
  - Pollard's λ (kangaroo) method.

- They work for every abelian group.

- They require

$$O\left(\sqrt{\text{group order}}\right)$$

  group operations to solve the discrete log.

Attacks for hyperelliptic curves

- Weil descent attack:
  - Frey / Gaudry, Hess and Smart,
  - for some curves defined over field extensions.
- Index calculus attack for large genus:
  - Adleman, DeMarrais and Huang

Attacks for hyperelliptic curves

- Weil descent attack:
  - Frey / Gaudry, Hess and Smart,
  - for some curves defined over field extensions.
- Index calculus attack for large genus:
  - Adleman, DeMarrais and Huang
- Index calculus attack for small genus:
  - Gaudry,
  - for curves of genus > 4,
  - variation (Harley) for curves of genus > 3,

Attacks for hyperelliptic curves

- Weil descent attack:
  - Frey / Gaudry, Hess and Smart,
  - for some curves defined over field extensions.
- Index calculus attack for large genus:
  - Adleman, DeMarrais and Huang
- Index calculus attack for small genus:
  - Gaudry,
  - for curves of genus > 4,
  - variation (Harley) for curves of genus > 3,
  - can be improved for curves of genus > 2.

We want to find a good set of "points" (the factor base)

$$P_1, P_2, \ldots, P_t$$

and "random" linear combinations

$$\alpha_i D_1 + \beta_i D_2 = \sum_{j=1}^{t} c_{ij} P_j.$$

We want to find a good set of "points" (the factor base)

$$P_1, P_2, \ldots, P_t$$

and "random" linear combinations

$$\alpha_i D_1 + \beta_i D_2 = \sum_{j=1}^{t} c_{ij} P_j.$$

We then find γ_i's such that for every j

$$\sum_{i=1}^{s} \gamma_i c_{ij} = 0.$$

This gives us

$$0 = \sum_{j=1}^{t} \left(\sum_{i=1}^{s} \gamma_i c_{ij} \right) P_j$$

This gives us

$$\begin{aligned} 0 &= \sum_{j=1}^{t} \left(\sum_{i=1}^{s} \gamma_i c_{ij} \right) P_j \\ &= \sum_{i=1}^{s} \gamma_i \left(\sum_{j=1}^{t} c_{ij} P_j \right) \end{aligned}$$

This gives us

$$\begin{aligned} 0 &= \sum_{j=1}^{t} \left(\sum_{i=1}^{s} \gamma_i c_{ij} \right) P_j \\ &= \sum_{i=1}^{s} \gamma_i \left(\sum_{j=1}^{t} c_{ij} P_j \right) \\ &= \sum_{i=1}^{s} \gamma_i \left(\alpha_i D_1 + \beta_i D_2 \right) \end{aligned}$$

This gives us

$$
\begin{aligned}
0 &= \sum_{j=1}^{t} \left(\sum_{i=1}^{s} \gamma_i c_{ij} \right) P_j \\
&= \sum_{i=1}^{s} \gamma_i \left(\sum_{j=1}^{t} c_{ij} P_j \right) \\
&= \sum_{i=1}^{s} \gamma_i \left(\alpha_i D_1 + \beta_i D_2 \right) \\
&= \left(\sum_{i=1}^{s} \gamma_i \alpha_i \right) D_1 + \left(\sum_{i=1}^{s} \gamma_i \beta_i \right) D_2
\end{aligned}
$$

This gives us

$$
\begin{aligned}
0 &= \sum_{j=1}^{t} \left(\sum_{i=1}^{s} \gamma_i c_{ij} \right) P_j \\
&= \sum_{i=1}^{s} \gamma_i \left(\sum_{j=1}^{t} c_{ij} P_j \right) \\
&= \sum_{i=1}^{s} \gamma_i \left(\alpha_i D_1 + \beta_i D_2 \right) \\
&= \left(\sum_{i=1}^{s} \gamma_i \alpha_i \right) D_1 + \left(\sum_{i=1}^{s} \gamma_i \beta_i \right) D_2 \\
&= \alpha D_1 + \beta D_2
\end{aligned}
$$

If $\beta \neq 0$, we can solve for D_2 :

$$D_2 = \frac{-\alpha}{\beta} D_1 \quad ,$$

i.e.

$$\begin{aligned} \lambda &= \frac{-\alpha}{\beta} \\ &= \frac{-\sum_{i=1}^{s} \gamma_i \alpha_i}{\sum_{i=1}^{s} \gamma_i \beta_i} \quad . \end{aligned}$$

Smooth divisors

Let $\mathcal{P} = C(\mathbb{F}_q)$, i.e. $\mathcal{P}$ is the set of points of C over $\mathbb{F}_q$. Let B be a subset of $\mathcal{P}$.

Let $\mathcal{P} = C(\mathbb{F}_q)$, i.e. $\mathcal{P}$ is the set of points of C over $\mathbb{F}_q$. Let B be a subset of $\mathcal{P}$.

A divisor is ***smooth relative to*** B if it is reduced and it can be written in the form

$$\sum_{i=1}^{k} D(P_i)$$

with the P_i's in B and $k \leq g$.

Smooth divisors

Let $\mathcal{P} = C(\mathbb{F}_q)$, i.e. $\mathcal{P}$ is the set of points of C over $\mathbb{F}_q$. Let B be a subset of $\mathcal{P}$.

A divisor is ***smooth relative to*** B if it is reduced and it can be written in the form

$$\sum_{i=1}^{k} D(P_i)$$

with the P_i's in B and $k \leq g$.

In this case, B is called the ***factor base***.

Smooth divisors

Let $\mathcal{P} = C(\mathbb{F}_q)$, i.e. $\mathcal{P}$ is the set of points of C over $\mathbb{F}_q$. Let B be a subset of $\mathcal{P}$.

A divisor is ***smooth relative to*** B if it is reduced and it can be written in the form

$$\sum_{i=1}^{k} D(P_i)$$

with the P_i's in B and $k \leq g$.

In this case, B is called the ***factor base***.

A ***potentially smooth*** divisor is smooth relative to $\mathcal{P}$.

Working with the factor base

- Make use of the equality $D(-P) = -D(P)$.

Working with the factor base

- Make use of the equality $D(-P) = -D(P)$.
- If P is in the factor base, $-P$ is also in the factor base, but we use only P for the factorization.
- Example of representation:

$$D(P_1)+D(-P_{29})+D(-P_{103}) = D(P_1)-D(P_{29})-D(P_{103})$$

Working with the factor base

- Make use of the equality $D(-P) = -D(P)$.
- If P is in the factor base, $-P$ is also in the factor base, but we use only P for the factorization.
- Example of representation:

  $$D(P_1)+D(-P_{29})+D(-P_{103}) = D(P_1)-D(P_{29})-D(P_{103})$$

- The "size" of the factor base is $|B|/2$ for the linear algebra.
- This decreases the running time for the search by 50% and time for the linear algebra by 75%.

Large primes

Given a factor base $B \subset \mathcal{P}$, a point $P \in \mathcal{P}$ is called a ***large prime*** if $P \notin B$.

Given a factor base $B \subset \mathcal{P}$, a point $P \in \mathcal{P}$ is called a ***large prime*** if $P \notin B$.

A reduced divisor

$$D = \sum_{i=1}^{k} D(P_i)$$

is said to be ***almost-smooth*** if:

- all but one of the P_i's are in B;
- the remaining P_i is a large prime.

Intersections

- Let T_i be an almost-smooth divisor with the large prime P.
- T_i is called an ***intersection*** if one of the previous almost-smooth divisor (T_j) has large prime $\pm P$.

Intersections

- Let T_i be an almost-smooth divisor with the large prime P.
- T_i is called an ***intersection*** if one of the previous almost-smooth divisor (T_j) has large prime $\pm P$.
- We use the intersection of T_i with T_j to build a (non-reduced) divisor that factors over the factor base.

Intersections

- Let T_i be an almost-smooth divisor with the large prime P.
- T_i is called an ***intersection*** if one of the previous almost-smooth divisor (T_j) has large prime $\pm P$.
- We use the intersection of T_i with T_j to build a (non-reduced) divisor that factors over the factor base.
- Intersections are used to decrease the time required to build the linear algebra system.

Intersections

- Let T_i be an almost-smooth divisor with the large prime P.
- T_i is called an ***intersection*** if one of the previous almost-smooth divisor (T_j) has large prime $\pm P$.
- We use the intersection of T_i with T_j to build a (non-reduced) divisor that factors over the factor base.
- Intersections are used to decrease the time required to build the linear algebra system.
- T_i is an intersection with at most one of the previous almost-smooth T_j's.

Cancelling large primes

If T_1, T_2 are two almost-smooth divisors who share the same large prime P,
i.e. T_1, T_2 can be represented in the form

$$T_1 = D(P) + \sum_{i=1}^{k_1-1} D(P_{1,i}) \quad \text{and} \quad T_2 = D(P) + \sum_{i=1}^{k_2-1} D(P_{2,i})$$

with $P_{1,i}, P_{2,i} \in B$, then we use the divisor

$$T' = T_1 - T_2 = \sum_{i=1}^{k_1-1} D(P_{1,i}) - \sum_{i=1}^{k_2-1} D(P_{2,i}).$$

Cancelling large primes

If T_1, T_2 are two almost-smooth divisors such that T_1 has large prime P and T_2 has large prime $-P$,
i.e. T_1, T_2 can be represented in the form

$$T_1 = D(P) + \sum_{i=1}^{k_1-1} D(P_{1,i}) \quad \text{and} \quad T_2 = -D(P) + \sum_{i=1}^{k_2-1} D(P_{2,i})$$

with $P_{1,i}, P_{2,i} \in B$, then we use the divisor

$$T' = T_1 + T_2 = \sum_{i=1}^{k_1-1} D(P_{1,i}) + \sum_{i=1}^{k_2-1} D(P_{2,i}).$$

Using a smaller factor base:

1. Search for the elements of the factor base
2. Initialization of the random walk
3. Search (random walk)
   - Search for potentially smooth divisors
   - Factorization of the potentially smooth divisors
   - Construction of the linear algebra system
4. Solution of the linear algebra system
5. Final solution

Using a smaller factor base:

1. Search for the elements of the factor base
2. Initialization of the random walk
3. Search (random walk)
   - Search for potentially smooth divisors
   - Factorization of the potentially smooth divisors
   - Construction of the linear algebra system
4. Solution of the linear algebra system
5. Final solution

Using large primes:

1. Search for the elements of the factor base
2. Initialization of the random walk
3. Search (random walk)
   - Search for potentially smooth divisors
   - Factorization of the potentially smooth divisors
   - Cancellation of the large primes (for intersections)
   - Construction of the linear algebra system
4. Solution of the linear algebra system
5. Final solution

Running time analysis

- Assume classical arithmetic.
- Assume $q > g!$.

Running time analysis

- Assume classical arithmetic.
- Assume $q > g!$.
- Assume the size of the factor base is q^r, $\frac{2}{3} < r < 1$.
- Find the expected running time with a factor base of that size.
- Choose r to "minimize" the running time.

Running time analysis

- Assume classical arithmetic.
- Assume $q > g!$.
- Assume the size of the factor base is q^r, $\frac{2}{3} < r < 1$.
- Find the expected running time with a factor base of that size.
- Choose r to "minimize" the running time.
- When using large primes, also assume $q^r < \frac{|C(\mathbb{F}_q)|}{2}$.

Factor base

We try values of $x_i \in \mathbb{F}_q$ to see if they correspond to x-coordinates of points of $C(\mathbb{F}_q)$.

We add points of $C(\mathbb{F}_q)$ in B until the factor base has the desired size.

This can be done in $O\left(g^2 q (\log q)^2\right)$ bit operations.

We choose the state function

$$\begin{aligned} \mathcal{R} : Jac(C)(\mathbb{F}_q) \times \{1, 2, \ldots, n\} &\rightarrow Jac(C)(\mathbb{F}_q) \\ (D, i) &\mapsto D + T^{(i)}. \end{aligned}$$

We take $n = O\left(\log(|Jac(C)(\mathbb{F}_q)|)\right)$.

We choose n random $\alpha^{(i)}$'s and $\beta^{(i)}$'s and compute

$$T^{(i)} = \alpha^{(i)} D_1 + \beta^{(i)} D_2.$$

This can be done in $O\left(g^4 (\log q)^4\right)$ bit operations.

We need a nonzero vector in the kernel of the matrix M.

The matrix is sparse with weigth $O\left(gq^r\right)$.

Operations are done modulo $|Jac(C)(\mathbb{F}_q)|$.

Using algorithms by Lanczos or Wiedemann, this can be done in

$$O\left(g^3 q^{2r} (\log q)^2\right)$$

bit operations.

We compute

$$\begin{aligned} \alpha &= \sum_i \gamma_i \alpha_i, \\ \beta &= \sum_i \gamma_i \beta_i \end{aligned}$$

and

$$\lambda = -\frac{\alpha}{\beta}.$$

The computations are done modulo $|J_q|$.

This can be done in $O\left(g^2 q^r (\log q)^2\right)$ bit operations.

Potentially smooth divisors

Proposition: *There are* $\frac{q^g}{g!} + O\left(\frac{gq^{g-\frac{1}{2}}}{g!}\right)$ *potentially smooth divisors in* $Jac(C)(\mathbb{F}_q)$.

We expect to have a potentially smooth divisor for every

$$O(g!)$$

divisors computed in the search.

Proposition: *For* $\frac{2}{3} < r < 1$, *there are* $\frac{q^{rg}}{g!} + O\left(\frac{g^2 q^{r(g-1)}}{g!}\right)$ *smooth divisors in* $Jac(C)(\mathbb{F}_q)$.

We expect to have to look at

$$O\left(g! q^{(1-r)g}\right)$$

divisors for each smooth divisor found in the search.

Search

- We need $O(q^r)$ smooth divisors.
- We expect to look at $O\left(g!q^{(1-r)g+r}\right)$ divisors, each taking (in bit operations):
  - $O(g^2(\log q)^2)$ to compute the reduced divisor;
  - $O(g \log q)$ to compute α_i and β_i;
  - $O(g^2(\log q)^2)$ to check if $a(x)$ splits completely.
- Of these, we expect $O\left(q^{(1-r)g+r}\right)$ to be potentially smooth (and must be factorized);
  - factorization take $O(g^2(\log q)^2)$ bit operations.
- Total of $O\left(g^2 g!q^{g-(g-1)r}(\log q)^2\right)$ bit operations.

Proposition: *For* $\frac{2}{3} < r < 1$, *there are* $\frac{q^{rg+1-r}}{(g-1)!} + O\left(\frac{q^{rg}}{(g-1)!}\right)$ *almost-smooth divisors in* $Jac(C)(\mathbb{F}_q)$.

For each almost-smooth divisors found during the search, we can expect to look at

$$O\left((g-1)!q^{(1-r)(g-1)}\right)$$

divisors.

Let $Q_n(s,i)$ be the probability of having i intersections out of a sample of size s drawn with replacement from a set of n elements.

Let $E_{n,s}$ be the expected number of intersections after s draws from a set of n elements, i.e.

$$E_{n,s} = \sum_{i=0}^{s-1} iQ_n(s,i).$$

Theorem: *If* $3 \leq s < n/2$, *then* $E_{n,s}$ *is between* $\frac{2s^2}{3n}$ *and* $\frac{s^2}{n}$.

In our case, n is the number of large primes (i.e. $n = q - q^r + O(\sqrt{q})$) and

$$E_{n,s} = O\left(\frac{s^2}{q}\right).$$

We want $E_{n,s} \approx q^r$, so we take $s = O\left(q^{(r+1)/2}\right)$.

It will then take

$$O\left(s(g-1)!q^{(g-1)(1-r)}\right) = O\left((g-1)!q^{(g-1)(1-r)+\frac{r+1}{2}}\right)$$

steps of random walk to build the linear algebra system.

Search

- We expect to look at $O\left((g-1)!q^{(g-1)(1-r)+\frac{r+1}{2}}\right)$ divisors;
  - each divisor takes $O(g^2(\log q)^2)$ bit operations.
- Of these, we expect $O\left(q^{(g-1)(1-r)+\frac{r+1}{2}}/g\right)$ to be potentially smooth each taking an extra $O(g^2(\log q)^2)$ bit operations.
- We also expect to get $O\left(q^{r-\frac{1-r}{2}}/g\right)$ smooth divisors.
- Total of $O\left(gg!q^{(g-1)(1-r)+\frac{r+1}{2}}(\log(q))^2\right)$ bit operations.

Using a smaller factor base:

1. $O\left(g^2 q (\log q)^2\right)$
2. $O\left(g^4 (\log q)^4\right)$
3. $O\left(g^2 g! q^{g-(g-1)r} (\log q)^2\right)$
4. $O\left(g^3 q^{2r} (\log q)^2\right)$
5. $O\left(g^2 q^r (\log q)^2\right)$

Using a smaller factor base:

1. $O\left(g^2 q (\log q)^2\right)$
2. $O\left(g^4 (\log q)^4\right)$
3. $O\left(g^2 g! q^{g-(g-1)r} (\log q)^2\right)$
4. $O\left(g^3 q^{2r} (\log q)^2\right)$
5. $O\left(g^2 q^r (\log q)^2\right)$

The total running time is then

$$O\left(g^2 g! q^{g-(g-1)r} (\log(q))^2\right) + O\left(g^3 q^{2r} (\log(q))^2\right).$$

bit operations.

Running times

For the original index calculus attack by Gaudry, $q^r = |C(\mathbb{F}_q)|$, which gives a running time of

$$O\left(g^3 q^{2+\epsilon}\right) + O\left(g^2 g! q^{1+\epsilon}\right)$$

bit operations.

To optimize the running time, we choose

$$r = \frac{g + \log_q((g-1)!)}{g+1},$$

which gives us

$$O\left(g^5 q^{2-\frac{2}{g+1}+\epsilon}\right)$$

bit operations.

Running times

Using large primes:

1. $O\left(g^2 q (\log q)^2\right)$
2. $O\left(g^4 (\log q)^4\right)$
3. $O\left(g g! q^{(g-1)(1-r)+\frac{r+1}{2}} (\log(q))^2\right)$
4. $O\left(g^3 q^{2r} (\log q)^2\right)$
5. $O\left(g^2 q^r (\log q)^2\right)$

Running times

Using large primes:

1. $O\left(g^2 q(\log q)^2\right)$
2. $O\left(g^4(\log q)^4\right)$
3. $O\left(gg!q^{(g-1)(1-r)+\frac{r+1}{2}}(\log(q))^2\right)$
4. $O\left(g^3 q^{2r}(\log q)^2\right)$
5. $O\left(g^2 q^r(\log q)^2\right)$

The total running time is then

$$O\left(gg!q^{(g-1)(1-r)+\frac{r+1}{2}}(\log(q))^2\right)+O\left(g^3q^{2r}(\log(q))^2\right).$$

bit operations.

To optimize the running time, we choose

$$r = \frac{g - \frac{1}{2} + \log_q((g-1)!/g)}{g + \frac{1}{2}},$$

which gives us

$$O\left(g^5 q^{2-\frac{4}{2g+1}+\epsilon}\right)$$

bit operations.

For small genus, we have:

g	square root attacks			
3	$q^{3/2}$			
4	q^2			
5	$q^{5/2}$			
6	q^3			

For small genus, we have:

g	square root attacks	original index calculus		
3	$q^{3/2}$	q^2		
4	q^2	q^2		
5	$q^{5/2}$	q^2		
6	q^3	q^2		

For small genus, we have:

g	square root attacks	original index calculus	smaller factor base	
3	$q^{3/2}$	q^2	$q^{3/2}$	
4	q^2	q^2	$q^{8/5}$	
5	$q^{5/2}$	q^2	$q^{5/3}$	
6	q^3	q^2	$q^{12/7}$	

Comparison

For small genus, we have:

g	square root attacks	original index calculus	smaller factor base	with large primes
3	$q^{3/2}$	q^2	$q^{3/2}$	$q^{10/7}$
4	q^2	q^2	$q^{8/5}$	$q^{14/9}$
5	$q^{5/2}$	q^2	$q^{5/3}$	$q^{18/11}$
6	q^3	q^2	$q^{12/7}$	$q^{22/13}$

Memory

One of the biggest problems of the index calculus attack is the memory requirement.

Memory

One of the biggest problems of the index calculus attack is the memory requirement.

- For the original index calculus: $O\left(gq^{1+\epsilon}\right)$ bits.
  - For the linear algebra.

Memory

One of the biggest problems of the index calculus attack is the memory requirement.

- For the original index calculus: $O\left(gq^{1+\epsilon}\right)$ bits.
  - For the linear algebra.
- Using a smaller factor base: $O\left(g^2 q^{\frac{g}{g+1}+\epsilon}\right)$ bits.
  - For the linear algebra.

Memory

One of the biggest problems of the index calculus attack is the memory requirement.

- For the original index calculus: $O\left(gq^{1+\epsilon}\right)$ bits.
  - For the linear algebra.
- Using a smaller factor base: $O\left(g^2 q^{\frac{g}{g+1}+\epsilon}\right)$ bits.
  - For the linear algebra.
- Using large primes: $O\left(g^2 q^{\frac{2g}{2g+1}+\epsilon}\right)$ bits.
  - For the storage of the almost-smooth divisors.

Memory

One of the biggest problems of the index calculus attack is the memory requirement.

- For the original index calculus: $O\left(gq^{1+\epsilon}\right)$ bits.
  - For the linear algebra.
- Using a smaller factor base: $O\left(g^2 q^{\frac{g}{g+1}+\epsilon}\right)$ bits.
  - For the linear algebra.
- Using large primes: $O\left(g^2 q^{\frac{2g}{2g+1}+\epsilon}\right)$ bits.
  - For the storage of the almost-smooth divisors.
  - The linear algebra requires $O\left(g^2 q^{\frac{2g-1}{2g+1}+\epsilon}\right)$ bits.