SQUARE ROOTS IN FINITE FIELDS AND QUADRATIC NONRESIDUES

JEREMY BOOHER

The law of quadratic reciprocity is a well-known and beautiful result about when a number is a quadratic residue (non-zero square) modulo a prime p. The problem of actually finding a square root is harder, and its solutions are not as widely known. We will discuss two simple randomized algorithms to do so. The Tonelli-Shanks algorithm is group-theoretic in nature, using the structure of $\mathbb{F}_p^\times$ to inductively find better and better approximations to the square root. It needs a quadratic non-residue modulo p to run, which can most efficiently be found through randomness. On the other hand, Cipolla's algorithm is field-theoretic in nature, and tries to produce, using randomness, a special quadratic polynomial that will allow the square root to be calculated in $\mathbb{F}_{p^2}$. Both algorithms run in polynomial time, but need randomness. No deterministic polynomial time algorithm is known: we will show such an algorithm is equivalent to a deterministic polynomial time algorithm for finding quadratic non-residues.

Next, we briefly discuss how finding square roots modulo a composite number is related to factoring, and how this allows the construction of the Rabin cryptosystem. Although not often used in practice, this cryptosystem is theoretically simple to describe and provably as secure as the factoring problem.

We finally turn to the question of deterministically finding quadratic non-residues. This leads to the question of bounding the size of the smallest quadratic non-residue modulo p as a function of p. It is elementary to get a bound of the form $O(\sqrt{p})$, which we will do, but much stronger results seem to be true. It is conjectured that the smallest quadratic non-residue is bounded by $O(\log(p)^2)$. We will prove this assuming the generalized Riemann hypothesis.

Throughout, we will discuss finite fields of prime order for simplicity. Everything generalizes to general finite field, usually at no extra cost. The general results are presented in *Algorithmic Number Theory* by Bach and Shallit [1], which is a good source to learn more.

1. Calculating Square Roots in $\mathbb{F}_p$

Let p be a prime and a be a quadratic residue modulo p. We would like an efficient algorithm to find an integer b such that $b^2 = a \mod p$.

As a first step, suppose that $p \equiv 3 \mod 4$. One of the simplest potential ways to find a square root for a would be to raise a to some power. We are looking for a power n such that $(a^n)^2 \equiv a \mod p$. Since $a^{\frac{p-1}{2}} = 1 \mod p$ as a is a quadratic residue, we need $2n \equiv 1 \mod \frac{p-1}{2}$. As $p \equiv 3 \mod 4$, we can take $n = \frac{p+1}{4}$.

Algorithm 1. *Let $p \equiv 3 \mod 4$ be prime and a a quadratic residue modulo p. A square root of a is given by $a^{\frac{p+1}{4}}$. This can be calculated in polynomial time.*

Proof. Let $a = u^2$. Then

$$\left(a^{\frac{p+1}{4}}\right)^2 \equiv u^{p+1} \equiv u^2 \equiv a \mod p.$$

Using repeated squaring, the power can be computed in time polynomial in $\log(p)$. □

The assumption that $p \equiv 3 \mod 4$ is essential - something more complicated is needed for the general case.

Date: July 13, 2012.

1.1. **The Tonelli-Shanks Algorithm.** We will first describe the algorithm, and then explain it conceptually.

Given a prime p, a quadratic residue a, and a quadratic non-residue v, we want to find a square root of a. Write $p-1=2^r s$, where s is odd. Start by letting

$$x_0 := a^{\frac{s+1}{2}} \mod p \quad \text{and} \quad w := v^s.$$

Note that w has order 2^r, and x_0 is an approximation to the square root in the sense that

$$\left(\frac{x_0^2}{a}\right)^{2^{r-1}} \equiv a^{s2^{r-1}} \equiv a^{\frac{p-1}{2}} \equiv 1 \mod p$$

since a is a quadratic residue. If $p \equiv 3 \mod 4$, $r=1$ and we have recovered the method in the previous section. Since we know the order is a power of two, define

$$2^{t_0} := \operatorname{ord}\left(\frac{x_0^2}{a}\right).$$

We now inductively improve the approximation: set

$$x_{i+1} := x_i \cdot w^{2^{r-t_i-1}} \quad \text{and} \quad 2^{t_{i+1}} := \operatorname{ord}\left(\frac{x_i^2}{a}\right)$$

Note that as $\frac{x_i^2}{a}$ has order 2^{t_i}, $\left(\frac{x_i^2}{a}\right)^{2^{t_i-1}} \equiv -1 \mod p$. Likewise, as w has order 2^r, we know $w^{2^{r-1}} \equiv -1 \mod p$. Then

$$\left(\frac{x_{i+1}^2}{a}\right)^{2^{t_i-1}} \equiv \left(\frac{x_i^2}{a}\right)^{2^{t_i-1}} (w^{2\cdot 2^{r-t_i-1}})^{2^{t_i-1}} \equiv -1 \cdot -1 \equiv 1 \mod p.$$

Therefore the order of $\frac{x_{i+1}^2}{a}$ is a power of 2, so t_{i+1} is defined and $t_{i+1} < t_i$.

When $t_n = 0$, we see that $x_n^2 \equiv a \mod p$ so we have found a square root for a.

Algorithm 2. *With the definitions given above, given a quadratic non-residue v, calculating the x_i and t_i until $t_n = 0$ gives a polynomial time algorithm for computing the square root.*

Proof. It is clear from above that $t_0 < r$ so as $2^r | p$, we see $t_0 < \log_2 p$. As t_i decreases by at least one at each step, this algorithm repeats at most $\log_2(p)$ times. Each step requires arithmetic modulo p which can be done quickly using repeated squaring. Thus this runs in polynomial time. □

Example 1. Let us calculate a square root of $a=8$ modulo 41. 3 is a quadratic non-residue as $\left(\frac{3}{41}\right) = \left(\frac{41}{3}\right) = \left(\frac{2}{3}\right) = -1$. As $41-1 = 2^3 \cdot 5$, $r=3$ and $s=5$. Therefore $x_0 = 8^{\frac{5+1}{2}} \equiv 20 \mod 41$ and $w = 3^5 \equiv -3 \mod 41$.

Now $x_0^2/a \equiv 8^5 \equiv 9 \mod 41$, and 9 has order 4, so $t_0 = 2$. Then $x_1 = x_0 v^{2^{3-2-1}} = 20 \cdot -3 \equiv -19 \mod 41$.

Furthermore, $x_1^2/a \equiv x_0^2/a \cdot w^2 \equiv 9 \cdot 9 \equiv -1 \mod 41$. Thus $t_1 = 1$. Then we have $x_2 = -19 \cdot 3^{2^{3-2}} = -19 \cdot 9 \equiv -7 \mod 41$. Therefore we have that ± 7 are square roots of 8 modulo 41.

It remains to give a conceptual explanation for what this algorithm is doing. This algorithm exploits the group theoretic structure of the cyclic group $G = \mathbb{F}_p^\times$. Write $|G| = p-1 = 2^r s$, and consider $G_i = \{g \in G : \operatorname{ord}(g) | 2^i\}$. As G is Abelian, these are a subgroups. Then filter G by a sequence of subgroups

$$G \supset G_r \supset \ldots \supset G_0 = \{1\}.$$

As G is cyclic of order $p-1$, $|G_i| = 2^i$. The quotient G/G_r has size s. Since s is odd, $\frac{s+1}{2}$ is a multiplicative inverse for 2 modulo s, and a square root of a in G/G_r can be calculated by raising $a = u^2$ to the $\frac{s+1}{2}$ power. This is exactly what is done to obtain x_0, which we now interpret as a square root for a in G/G_r (in fact, the analysis showed it is a square root in G/G_{r-1}).

We now want to use a square root $x_i \in G/G_{t_i}$ for a to obtain a square root in G/G_{t_i-1}. By definition of the t_i, x_i is a square root of a in G/G_{t_i} but not in G/G_{t_i-1}. Therefore x_i^2/a is the non-identity element in $G_{t_i}/G_{t_i-1} \simeq \mathbb{Z}/2\mathbb{Z}$. On the other hand, $w^{2^{r-t_i-1}}$ has order exactly 2^{t_i+1}, so $\left(w^{2^{r-t_i-1}}\right)^2$ reduces to the non-trivial element in G_{t_i}/G_{t_i-1} as well. Therefore

$$\frac{x_i^2\left(w^{2^{r-t_i-1}}\right)^2}{a} \in G_{t_i-1}$$

so $x_i \cdot w^{2^{r-t_i-1}}$ is a square root for a in G/G_{t_i-1}. The inductive process described in the algorithm runs until it produces a square root for a in $G/G_0 \simeq G = \mathbb{F}_p^\times$.

Remark 2. To implement this algorithm, one needs a quadratic non-residue modulo p. Since half of the non-zero elements of $\mathbb{F}_p$ are quadratic non-residues, simply choosing elements at random and checking whether they are quadratic residues using Euler's criteria (or quadratic reciprocity) will make this into a randomized polynomial time algorithm. There is no known unconditional way to do this deterministically.

1.2. **Cipolla's Algorithm.** We now present a different algorithm to compute square roots. It is still a randomized algorithm, and uses extensions of $\mathbb{F}_p$ instead of the group structure of $\mathbb{F}_p^\times$.

Algorithm 3. *Given a quadratic residue a in $\mathbb{F}_p$, to calculate a square root choose $t \in \mathbb{F}_p$ at random until $t^2 - 4a$ is not a square in $\mathbb{F}_p$. Then calculate $x^{\frac{p+1}{2}}$ in $\mathbb{F}_p[x]/(x^2 - tx + a)$. This gives a randomized polynomial time algorithm for calculating square roots.*

Proof. Given such a t, the polynomial $x^2 - tx + a$ is irreducible as the discriminant is not a square in $\mathbb{F}_p$. Thus $\mathbb{F}_p[x]/(x^2 - tx + a)$ is a field with p^2 elements. Then the element x has minimal polynomial $x^2 - tx + a$, and

$$\left(x^{\frac{p+1}{2}}\right)^2 = x^{p+1} = N_{\mathbb{F}_{p^2}/\mathbb{F}_p}(x) = a$$

since $\mathrm{Gal}(\mathbb{F}_{p^2}/\mathbb{F}_p)$ is generated by the Frobenius map. Note that $x^{\frac{p+1}{2}}$ does in fact lie in $\mathbb{F}_p$, since there are only two square roots of a in $\mathbb{F}_{p^2}$, and we assumed a is a quadratic residue in $\mathbb{F}_p$. This is easy to calculate using repeated squaring.

If t is chosen at random, what is the chance that $x^2 - tx + a$ is irreducible ($t^2 - 4a$ is not a square)? We will count polynomials of the form $x^2 - tx + a$ that factor as $(x - \alpha)(x - \beta)$ over $\mathbb{F}_p$. As $t = \alpha + \beta$ and $\alpha\beta = a$, if we know β then $\alpha = \frac{a}{\alpha}$ and hence we know t. There are $p - 1$ choices for α, which determines β and t. Since α and β are interchangeable, if they are distinct we double count these polynomials. $\alpha = \beta$ happens only if α is a square root of a, so there are $\frac{p-3}{2}$ polynomials with distinct roots and 2 polynomials with double roots. Thus $\frac{p+1}{2}$ of the polynomials of this form factor, so $\frac{p-1}{2}$ are irreducible. Therefore a random t has about a fifty percent chance of working. Thus we obtain a randomized polynomial algorithm. □

Although this avoids needing a quadratic non-residue as input, we still find one in the process of running this algorithm and needed to use randomness. In fact, there is no escaping this.

Proposition 3. *The problem of finding a quadratic non-residue in deterministic polynomial time is equivalent to the problem of extracting square roots in deterministic polynomial time.*

Proof. The Tonelli-Shanks algorithm gives a way to extract square roots given a quadratic non-residue. Conversely, given an algorithm to extract square roots write $p - 1 = 2^r s$ and compute a primitive 2^rth root of unity by repeatedly taking square roots starting with -1. This takes at most $\log_2(p)$ steps, and the resulting root of unity has order 2^r and hence is a quadratic non-residue. □

2. The Rabin Cryptosystem

Given a square root algorithm for $\mathbb{F}_p$, it is easy to generalize it to composite moduli. Let a be a square in $\mathbb{Z}/n\mathbb{Z}$. Factor $n = p_1^{r_1} \dots p_m^{r_m}$, and compute a square root of $a \in \mathbb{Z}/p_i\mathbb{Z}$ for each i.[1] Then use Hensel's lemma to lift it to a square root modulo $p_i^{r_i}$, and combine them using the Chinese Remainder Theorem. This is conceptually simple, but in practice the factorization step is inefficient. There is no known polynomial time algorithm that will factor general integers. Much of modern public key cryptography is based on the assumption that factorization cannot be done efficiently, so this is a serious impediment. In fact, the Rabin cryptosystem is built around the intractability of this problem.

The goal of the Rabin cryptosystem is for Alice to be able to send a message to Bob that cannot be understood by Eve who is monitoring their communications. Furthermore, Alice and Bob need to be able to communicate without having any shared secret (which would require them to meet in person, for example).[2] The idea is to arrange for the information necessary to encrypt a message (called the public key) to be different than the information necessary to decrypt the message (called the private key), which is kept secret by Bob. Only the first piece of information is made publicly available.

2.1. Encryption and Decryption.

To set up Rabin encryption, Bob picks two large prime numbers p and q.[3] He publishes $n = pq$, the public key. p and q are the private decryption keys, so he keeps them secret.

For Alice to send a message M, it needs to be an integer satisfying $0 \leq M < n$. (If the message is text, it can be converted into a number using ASCII and then split into digit blocks of the appropriate size.) Alice then calculates $E(M) = M^2 \mod n$, and sends $E(M)$ to Bob.

Bob can decrypt the cipher-text $E(M) = C$ by finding the square roots of C. Since Bob knows the factorization of n already, he (but no one else) can use the square root algorithm described above. Using the Chinese Remainder Theorem, he obtains the four square roots of $E(M)$ in $\mathbb{Z}/n\mathbb{Z}$. One of these is the message which Alice sent him. However, this poses a problem: how does Bob tell which square root is the correct one? Without help, he can't.

There are various clever ways to solve the disambiguation problem. The simplest involves padding the message with a prearranged sequence of bits. For example, Alice and Bob agree that at the start of every message they'll put the word Rabin. (So instead of saying "Hello Bob", the message would be "RabinHello Bob". When Bob sees all four square roots, it's unlikely that any of the 3 extraneous square roots will start with Rabin, so Bob can figure out which of the four choices is Alice's message. However, this complication means that RSA encryption is more often used in practice, despite it requiring substantially more modular arithmetic than Rabin encryption.

2.2. Security.

The other important question is whether Eve can read Alice's message. In theory she can: all she needs to do is factor n, find its two prime factors, and follow the same procedure Bob did. However, if both p and q were chosen to have 800 digits each, the fastest known methods of factoring n would require centuries. Is there any other way? Rabin encryption is nice because it is possible to prove there is no other way.

Proposition 4. *If an adversary Eve can decode a positive proportion of the messages Alice sends to Bob, then Eve can factor n.*

Proof. Suppose Eve had a magic algorithm that decrypts a positive fraction of the messages encrypted using Bob's public key n. Using this, Eve can factor n. To do so, Eve picks a random

[1] If $p_i = 2$, start working modulo 4.

[2] If they can arrange to share a secret, for example by meeting in person or by trusted third parties, they could use a one-time pad which is provably secure.

[3] In practice, these is no reason to not pick primes which are $\equiv 3 \mod 4$ to simplify the decryption calculations.

integer x from the range $[0, n)$. Then she calculates $x^2 \mod n$, and repeats this until she finds one that her algorithm works on. Since x was chosen randomly, there is a one-half chance that the decryption algorithm produces $\pm x \mod n$. In that case, try again. Otherwise, Eve has found x and y with $x^2 = y^2 \mod n$ and $x \neq \pm y \mod n$. Then $(x - y)(x + y) = 0 \mod n$, but neither factor is 0. This means that $x - y$ or $x + y$ is a multiple of p, while the other is a multiple of q. Using the Euclidean algorithm (which is very efficient) produces the greatest common factor of $x - y$ and n, which is one of the prime factors. Since the algorithm works on a positive proportion on possible messages, and when it works there is a one half chance it fails, the probability that this fails t trials is c^t for some constant $c < 1$. This goes to 0 exponentially fast, giving Eve an effective way to factor n. □

Thus conditional on the widely believed fact that factoring is hard, Rabin encryption is unbreakable. This is in contrast to the situation with RSA, for which no one has published a way to break it but no one can prove how secure it is.

Remark 5. There are other notions of security that require modifications to this algorithm. For example, we might want this to be semantically secure in the sense that an adversary who knows the message is unable to determine significant information about the encryption. If there were a limited number of possible messages (for example, "Sell" or "Hold"), this protects against the possibility of Eve simply encrypting the possible messages and checking which one matches the encrypted message. A crude way to defend against this is to pad the message with a sequence of random bits. The algorithm as first described is also vulnerable to chosen cipher-text attacks.

Remark 6. The idea that extracting square roots modulo n is possible if and only if the factorization for n is known is used elsewhere in cryptography. For example, it provides one method of implementing oblivious transfer, and provides a protocol for coin flipping.

3. Smallest Quadratic Non-residues

We now return to the problem of finding a quadratic non-residue in $\mathbb{F}_p$. As one can be found easily at random, this is primarily of theoretical interest. The simplest idea to derandomize it is to generate elements of $\mathbb{F}_p$ in some deterministic order, and check them until a quadratic non-residue is found. There are many ways to do this: the way most tractable to analyze is to just look at $2, 3, 4, \ldots$. This leads us to investigate the size of the smallest quadratic non-residue modulo p. In particular, we are interested in results that say the smallest quadratic non-residue is $O(f(p))$, by which we mean that there is a constant C such that there is always a quadratic non-residue in $\{2, 3, \ldots, [Cf(p)]\}$ for any prime p. It is elementary to prove that that the smallest quadratic non-residue is $O(\sqrt{p})$, and we will also sketch a proof assuming the generalized Riemann hypothesis that the smallest quadratic non-residue is $O(\log(p)^2)$.

3.1. The $O(\sqrt{p})$ Bound.

Since we know $\left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right)$, if we know that the first $\sqrt{p}$ elements are quadratic residues and that the products of these elements were distinct, there would be $\sqrt{p}^2 > \frac{p-1}{2}$ quadratic residues in $\mathbb{F}_p$. This is not true, but we can salvage it by considering the first $2\sqrt{p}$ elements. The following beautiful proof comes from footnote 1 of [5].

Proposition 7. *There exists a quadratic non-residue in* $(0, 2\sqrt{p})$.

Proof. If $p \equiv 3 \mod 4$, then $p - [\sqrt{p}]^2$ is a quadratic non-residue in $\mathbb{F}_p$ as $\left(\frac{-1}{p}\right) = -1$, and $p - [\sqrt{p}]^2 \leq 2\sqrt{p}$.

If $p \equiv 1 \mod 4$, then $\left(\frac{-1}{p}\right) = 1$. Suppose there is no quadratic non-residue in $(0, 2\sqrt{p})$. Then there is no quadratic non-residue in $(-2\sqrt{p}, 2\sqrt{p})$. Let n be any element of $\mathbb{F}_p$. Under this assumption, we will show that n is a quadratic residue, a contradiction.

Consider expressions of the form $a + bn$, where $0 \leq a, b \leq \sqrt{p}$. There are at least $\sqrt{p}^2 = p$ such expressions, each taking on a value in $\mathbb{F}_p$, so by the pigeonhole principle there exist $0 \leq a, b, c, d \leq \sqrt{p}$ such that $a + bn = c + dn$ and $(a, b) \neq (c, d)$. Note that if $b = d$ then $a = c$. Therefore we see that

$$n = \frac{c - a}{b - d}$$

and by our assumption on all the elements in $(-2\sqrt{p}, 2\sqrt{p})$ being quadratic residues, n is a quadratic residue. □

There is an alternate approach to obtain an almost-equivalent bound of $O(\sqrt{p}\log(p))$ using the Pólya-Vinagradov inequality bounding character sums. It states that if $\chi : (\mathbb{Z}/n\mathbb{Z})^\times \to \mathbb{C}^\times$ is a non-trivial character then

$$\left| \sum_{x=m}^{m+r} \chi(x) \right| \leq 2\sqrt{n}\log(n).$$

To apply this to the problem of the smallest quadratic non-residue, let χ be the Legendre symbol $(\mathbb{Z}/p\mathbb{Z}) \to \{\pm 1\}$. The bound says that there is a constant C so that

$$\left| \sum_{x=1}^{q} \left(\frac{x}{p}\right) \right| \leq C\sqrt{p}\log(p).$$

Taking q to be one less than the smallest quadratic non-residue, since $\left(\frac{x}{p}\right) = 1$ for $1 \leq x \leq q$ we obtain $q \leq C\sqrt{p}\log(p)$. Thus the smallest quadratic non-residue is $O(\sqrt{p}\log(p))$.

Remark 8. This is just the crudest of the unconditional bounds that can be obtained from analytic number theory. Burgess strengthened the result to show the smallest quadratic non-residue is $O(p^{\frac{1}{4\sqrt{e}}+\epsilon})$ which up to improving the ϵ factor is the best unconditional bound. [3] A more detailed discussion, and proofs of many of these facts, are found on Terry Tao's blog. [7]

3.2. **The Conditional $O(\log(p)^2)$ Bound.** Using the generalized Riemann hypothesis, much sharper bounds are possible. We will prove the following, assuming substantially more background than in the previous sections.

Theorem 9. *Assuming the generalized Riemann hypothesis, the smallest quadratic non-residue is $O(\log(p)^2)$.*

Recall that the generalized Riemann hypothesis (GRH) states that for the any Dirichlet character $\chi : (\mathbb{Z}/n\mathbb{Z})^\times \to \mathbb{C}^\times$, all of the non-trivial zeros of the L-function $L(s, \chi)$ have real part $\frac{1}{2}$. Taking χ to be the trivial character, this is the usual Riemann hypothesis.

Let $N_\chi(t)$ denote the number of zeros of $L(s, \chi)$ with $0 < \mathrm{Re}(s) < 1$ and $|\mathrm{Im}(s)| \leq t$. The following is a standard fact from analytic number theory [4, Section 16]:

$$N_\chi(t) = O(t\log(nt)). \tag{1}$$

The idea is to use the argument principal to express the number of zeroes in this region as an integral and then to bound it.

The generalized Riemann hypothesis gives considerable information about the distribution of zeros and the growth of the logarithmic derivative of the L-function.

Lemma 10. *Assuming the generalized Riemann hypothesis we have*

$$\left| \frac{L'}{L}(\frac{1}{4} + it, \chi) \right| = O(\log(n(|t| + 2))).$$

This is a technical analytic fact. The proof will be discussed in an appendix. We use this to establish the following proposition, whose proof uses some of the same techniques as the proof of the prime number theorem.

Proposition 11. *Assuming GRH, let χ be a non-trivial Dirichlet character. Then*

$$\sum_{p\leq x}\Lambda(p)\chi(p)(x-p)=O(x^{3/2}\log(n)).$$

Proof. Recall that by differentiating the logarithm of the Euler product, we obtain

$$-\frac{L'}{L}(s,\chi)=\sum_{m=1}^{\infty}\chi(m)\Lambda(m)m^{-s}.$$

Therefore we calculate that

$$\begin{aligned}\frac{-1}{2\pi i}\int_{2-i\infty}^{2+i\infty}\frac{x^{s+1}}{s(s+1)}\frac{L'}{L}(s,\chi)ds&=\frac{1}{2\pi i}\int_{2-i\infty}^{2+i\infty}\frac{x^{s+1}}{s(s+1)}\sum_{m=1}^{\infty}\frac{\chi(m)\Lambda(m)}{m^s}ds\\&=\sum_{m\leq x}\chi(m)\Lambda(m)(x-m)\end{aligned}$$

using the standard fact that

$$\frac{1}{2\pi i}\int_{c-i\infty}^{c+i\infty}\frac{a^s}{s(s+1)}ds=\begin{cases}0 & \text{if } 0<a\leq 1\\ 1-\frac{1}{a} & \text{if } 1\leq a\end{cases}$$

On the other hand, we can shift the contour of integration to the line $\mathrm{Re}(s)=\frac{1}{4}$, obtaining that

$$\frac{-1}{2\pi i}\int_{2-i\infty}^{2+i\infty}\frac{x^{s+1}}{s(s+1)}\frac{L'}{L}(s,\chi)ds=-\sum_{\rho}\frac{x^{\rho+1}}{\rho(\rho+1)}-\frac{1}{2\pi i}\int_{\frac{1}{4}-i\infty}^{\frac{1}{4}+i\infty}\frac{x^{s+1}}{s(s+1)}\frac{L'}{L}(s,\chi)ds$$

where ρ runs over the non-trivial zeros of $L(s,\chi)$. The residue of $\frac{L'}{L}(s,\chi)$ at a zero of $L(s,\chi)$ is its multiplicity, so the residue of $\frac{x^{s+1}}{s(s+1)}\frac{L'}{L}(s,\chi)$ is $\frac{x^{\rho}}{\rho(\rho+1)}$. Now GRH implies that the real part of each zero is $\frac{1}{2}$, so

$$\left|\sum_{\rho}\frac{x^{\rho+1}}{\rho(\rho+1)}\right|\leq x^{\frac{3}{2}}\left(\frac{4}{3}N_\chi(2)+\int_2^{\infty}\frac{dN_\chi(t)}{t^2}\right)$$

Integrating by parts, we see that

$$\left|\int_2^{\infty}\frac{dN_\chi(t)}{t^2}\right|\leq C+C'\int_2^{\infty}\frac{N_\chi(t)}{t^3}dt=O\left(\int_2^{\infty}\frac{t\log(nt)}{t^3}dt\right)=O(\log(n))$$

using (1) to control $N_\chi(t)$. Therefore we conclude that

$$\left|\sum_{\rho}\frac{x^{\rho+1}}{\rho(\rho+1)}\right|=O(x^{3/2}\log(n)).$$

Now we can use the lemma to see that

$$\left|\int_{\frac{1}{4}-i\infty}^{\frac{1}{4}+i\infty}\frac{x^{s+1}}{s(s+1)}\frac{L'}{L}(s,\chi)ds\right|=O\left(x^{5/4}\int_0^{\infty}\frac{\log(n(t+2))}{(t+\frac{1}{4})^2}dt\right)=O(x^{\frac{5}{4}}\log(n))$$

This completes the proof of the proposition. □

We can now prove a general theorem about $\mathbb{F}_p^\times$ being generated by small elements. Looking at the squares in $\mathbb{F}_p^\times$ will give our desired result about quadratic non-residues.

Theorem 12. *Assuming GRH, there is a constant C such that no proper subgroup of $\mathbb{F}_p^\times$ contains every element in $(0, C\log(p)^2)$.*

Proof. Let H be a proper subgroup, and lift a non-trivial Dirichlet character of $\mathbb{F}_p^\times/H$ to a non-trivial Dirichlet character χ on $\mathbb{F}_p^\times$. Let x be the smallest element of $\mathbb{F}_p^\times$ not in H, so $\chi(a) = 1$ on $(0, x)$. The Proposition gives the bound

$$\sum_{q<x} \Lambda(q)\chi(q)(x-q) = O(x^{\frac{3}{2}} \log(p)).$$

However, we also obtain the lower bound

$$\begin{aligned}\sum_{q<x} \Lambda(q)\chi(q)(x-q) &= \sum_{q<x} (x-q)\log(q) \\ &\geq \sum_{q<\frac{1}{2}x} \frac{1}{2}x \log(q) \\ &\geq cx^2\end{aligned}$$

as $\sum_{q<\frac{1}{2}x} \log(q) \sim \frac{1}{2}x$ by the prime number theorem. Therefore $x^2 \leq Cx^{\frac{3}{2}} \log(p)$ and hence

$$x = O(\log(p))^2.$$

In other words, the smallest element element not in a proper subgroup is of size $O(\log(p)^2)$. □

Corollary 13. *Assuming the generalized Riemann hypothesis, the smallest quadratic non-residue is $O(\log(p)^2)$.*

Proof. Apply the theorem to the subgroup of squares. □

Remark 14. This is almost the best possible bound. It is possible to show that there is a constant c such that for infinitely many primes the smallest quadratic non-residue is at least $c\log(p)$. The idea, attributed to Chowla, is to choose primes where congruence conditions and quadratic reciprocity force all of the small primes to be quadratic residues. To carry out this argument, one uses Linnik's theorem to bound the size of the smallest prime satisfying a congruence condition. Details are discussed in [2, Theorem 3.10].

Appendix A.

We sketch a proof of Lemma 10 using GRH. Recall that χ is a Dirichlet character with conductor n, so has a functional equation. The exact form does not matter: all that is relevant is the values of $L(s,\chi)$ on the line with real part $\frac{1}{4}$ are related to the values on the line with real part $\frac{3}{4}$. Since we are interested in the size of $\log|L(s,\chi)|$, we need to understand $\log|\Gamma(z)|$, which is easily done with Stirling's formula. Therefore we can bound the ratio of the Γ functions appearing in the functional equation, and see that

$$\log|L(\frac{1}{4}+it,\chi)| = \log|L(\frac{3}{4}+it,\chi)| + O(\log(|t|n)).$$

The next question is to investigate the growth of $|L(c+it,\chi)|$ as $t\to\infty$ in terms of c. According to the Lindelöf hypothesis, for $\frac{1}{2} \leq c \leq 1$, this is $O(|t|^\epsilon)$. This does follow from GRH, but the unconditional results along these lines are sufficiently strong for our application, so we will use them instead. The only subtlety is that we must make the dependence on the conductor n clear. Theorem 12.9 of [6] makes this dependence clear, so we will cite it, but be aware that simpler arguments along these lines that do not use the Burgess bound would give an equally usefull statement. It says that for $\operatorname{Re} s \geq \frac{1}{2}$

$$|L(s,\chi)| = O(|s|n^{\frac{3}{16}+\epsilon})$$

with the implied constant depending only on ϵ. Note that this is stated only for $\operatorname{Re} s = \frac{1}{2}$, but the convexity bound, or the Phragmén-Lindelöf principle directly, extends it to the right. All we care about is that this says

$$\log(|L(s,\chi)|) = O(\log(|t|n)). \tag{2}$$

Now we wish to obtain a bound on the logarithmic derivate of L. The key is to use Caratheodory's inequality, which is a clever application of the Schwarz Lemma. It states that if $f(z)$ is a holomorphic function on $|z| \leq R$ and $0 < r < R$, then

$$\max_{|z|=r} |f'(z)| \leq \frac{8}{(R-r)^2} \left(f(0) + \max_{|z|=R} \operatorname{Re} f(z) \right).$$

This follows from the standard version of the inequality, involving f instead of f', using the Cauchy integral formula [8, 5.51]. What this says is that if $|z| < \frac{15}{16}R$ then

$$|f'(z)| \leq C_R \left(f(0) + \max_{|z|=R} \operatorname{Re} f(z) \right).$$

where C_R is a constant depending on R but not the function.

Proof. Using the functional equation, it suffices to show that

$$\log|L(\frac{3}{4} + it, \chi)| = O(\log|t|n).$$

By GRH, $L(s,\chi)$ has no zeroes for $\operatorname{Re}(s) > \frac{1}{2}$. Therefore for any t, we may consider the disc $|z - (2+it)| \leq \frac{11}{8}$ on which $\log L(s,\chi)$ is holomorphic. Applying the bound from Caratheodory's inequality, we see that

$$\left| \frac{L'}{L}(\frac{3}{4} + it, \chi) \right| \leq C \left(L(2+it,\chi) + \max_{|z-(2+it)|=\frac{11}{8}} \operatorname{Re} \log(L(z,\chi) \right).$$

We know that $L(2+it,\chi) = O(1)$ (bound the Dirichlet series directly). Furthermore, as $\operatorname{Re}\log(L(z,\chi)) = \log|L(z,\chi)|$, the bound in (2) implies that

$$\left| \frac{L'}{L}(\frac{3}{4} + it, \chi) \right| = O(\log(|t|n)).$$

This gives the desired bound. □

References

1. E. Bach and J. Shallit, *Algorithmic number theory, volume 1: Efficient algorithms*, Algorithmic number theory, MIT Press, 1996.
2. Carmen Bruni, *Least quadratic non-residue and least primitive root*, `http://www.math.ubc.ca/~gerg/teaching/613-Winter2011/LeastQuadraticNonResidue.pdf`.
3. D. A. Burgess, *On character sums and primitive roots*, Proceedings of the London Mathematical Society **s3-12** (1962), no. 1, 179–192.
4. H. Davenport and H.L. Montgomery, *Multiplicative number theory*, Graduate Texts in Mathematics, Springer, 2000.
5. Noam Elkies, *Math 229: Introduction to analytic number theory: Incomplete character sums I*, `www.math.harvard.edu/~elkies/M229.09/short.pdf`.
6. H. Iwaniec and E. Kowalski, *Analytic number theory*, American Mathematical Society Colloquium Publications, no. v. 53, American Mathematical Society, 2004.
7. Terence Tao, *The least quadratic nonresidue, and the square root barrier*, `http://terrytao.wordpress.com/2009/08/18/the-least-quadratic-nonresidue-and-the-square-root-barrier/`.
8. E.C. Titchmarsh, *The theory of functions*, Oxford science publications, Oxford University Press, 1939.