# Dixon's Factoring Algorithm

## Basic (Important) Idea (Not Just For Dixon's Algorithm)

- Suppose
$$n = p_1^{e_1} p_2^{e_2} \cdots p_r^{e_r}$$
with $p_j$ "odd" distinct primes and $e_j \in \mathbb{Z}^+$.

- Then $x^2 \equiv 1 \pmod{p_j^{e_j}}$ has two solutions which implies $x^2 \equiv 1 \pmod{n}$ has $2^r$ solutions.

- If $x$ and $y$ are random and $x^2 \equiv y^2 \pmod{n}$, then with probability $(2^r - 2)/2^r$ we can factor $n$ (nontrivially) by considering $\gcd(x + y, n)$.

# Dixon's Factoring Algorithm

1. Randomly choose a number $a > \sqrt{n}$ and compute $s(a) = a^2 \mod n$.

2. A bound $B = B(n)$ is chosen (specified momentarily). Determine if $s(a)$ has a prime factor $> B$. We choose a new $a$ if it does. Otherwise, we obtain a complete factorization of $s(a)$.

3. Let $p_1, \ldots, p_t$ denote the primes $\leq B$. We continue steps (1) and (2) until we obtain $t+1$ different $a$'s, say $a_1, \ldots, a_{t+1}$.

4. From the above, we have the factorizations

$$s(a_i) = p_1^{e(i,1)} p_2^{e(i,2)} \cdots p_t^{e(i,t)} \quad \text{for } i \in \{1, 2, \ldots, t+1\}.$$

# Dixon's Factoring Algorithm

4. From the above, we have the factorizations

$$s(a_i) = p_1^{e(i,1)} p_2^{e(i,2)} \cdots p_t^{e(i,t)} \quad \text{for } i \in \{1, 2, \ldots, t+1\}.$$

For $i \in \{1, 2, \ldots, t+1\}$, compute the vectors

$$\vec{v}_i = \langle e(i,1), e(i,2), \ldots, e(i,t) \rangle \mod 2.$$

These vectors are linearly dependent modulo 2. Use Gaussian elimination (or something better) to find a non-empty set $S \subseteq \{1, 2, \ldots, t+1\}$ such that $\sum_{i \in S} \vec{v}_i \equiv \vec{0} \pmod 2$. Calculate $x \in [0, n-1] \cap \mathbb{Z}$ (in an obvious way) satisfying

$$\prod_{i \in S} s(a_i) \equiv x^2 \pmod n.$$

4. **From the above, we have the factorizations**

$$s(a_i) = p_1^{e(i,1)} p_2^{e(i,2)} \cdots p_t^{e(i,t)} \quad \text{for } i \in \{1, 2, \ldots, t+1\}.$$

**For $i \in \{1, 2, \ldots, t+1\}$, compute the vectors**

$$\vec{v}_i = \langle e(i,1), e(i,2), \ldots, e(i,t) \rangle \mod 2.$$

**These vectors are linearly dependent modulo 2. Use Gaussian elimination (or something better) to find a non-empty set $S \subseteq \{1, 2, \ldots, t+1\}$ such that $\sum_{i \in S} \vec{v}_i \equiv \vec{0} \pmod 2$. Calculate $x \in [0, n-1] \cap \mathbb{Z}$ (in an obvious way) satisfying**

$$\prod_{i \in S} s(a_i) \equiv x^2 \pmod n.$$

5. **Calculate $y = \prod_{i \in S} a_i \mod n$. Then $x^2 \equiv y^2 \pmod n$. Compute $\gcd(x + y, n)$. Hopefully, a nontrivial factorization of $n$ results.**

The number of different $a$'s we expect to consider before we get enough good $s(a)$'s in the algorithm is

$$(\pi(B)+1)\exp\left((1+o(1))\log n \log u / \log B\right).$$

We also expect $\leq B$ steps to factor each value of $s(a)$. This means we should take

$$B = \exp\left(\sqrt{\log n}\sqrt{\log u}/\sqrt{2}\right) = \exp\left(\sqrt{\log n}\sqrt{\log\log n}/2\right),$$

and the expected running time for Dixon's Algorithm is about

$$\exp\left(2\sqrt{\log n}\sqrt{\log\log n}\right),$$

including Gaussian elimination.

Comment: This is a rough estimate. A closer analysis would give a running time of

$$\exp\left((2\sqrt{2}+o(1))\sqrt{\log n}\sqrt{\log\log n}\,\right).$$

With some more work, Pomerance and later Vallée reduced the constant $2\sqrt{2}$ so that now we know it can be replaced to $\sqrt{4/3}$.

# The CFRAC Algorithm

Every real number $\alpha$ can be written uniquely as a *simple continued fraction*

$$\alpha = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \ddots}}}$$

where $a_0 \in \mathbb{Z}$ and $a_j \in \mathbb{Z}^+$ for $j \geq 1$. The *convergents* obtained by truncating the above give approximations $a/b$ to $\alpha$ satisfying

$$\left|\alpha - \frac{a}{b}\right| < \frac{1}{b^2}.$$

# The CFRAC Algorithm

$$\left|\alpha - \frac{a}{b}\right| < \frac{1}{b^2}$$

$$\left|\alpha^2 - \frac{a^2}{b^2}\right| \ll \frac{\alpha}{b^2}$$

$$\left|b^2\alpha^2 - a^2\right| \ll \alpha$$

Comment: Every convergent $a/b$ of $\sqrt{n}$ satisfies

$$\left|b^2 n - a^2\right| < 2\sqrt{n}.$$

**EXAMPLE**

# The CFRAC Algorithm

$$\sqrt{7} = 2 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{4 + \cfrac{1}{1 + \ddots}}}}}$$

$$2\sqrt{7} = 5.2915\ldots$$

Convergents: $\frac{2}{1}, \frac{3}{1}, \frac{5}{2}, \frac{8}{3}, \frac{37}{14}, \ldots$

Comment: Every convergent $a/b$ of $\sqrt{n}$ satisfies

$$\left|b^2 n - a^2\right| < 2\sqrt{n}.$$

# The CFRAC Algorithm

**Compute the numerators $a_j$ of the convergents of $\sqrt{n}$. If the corresponding denominators are $b_j$, then $|a_j^2 - nb_j^2| < 2\sqrt{n}$. Recall $s(a) = a^2 \mod n$. Repeat Dixon's algorithm but now**

- **Define $s(a)$ to be in $(-n/2, n/2]$ with $s(a) \equiv a^2 \pmod{n}$. Then $|s(a_j)| < 2\sqrt{n}$.**
- **Treat $-1$ (the possible negative sign in $s(a)$) as a prime.**

**The chance that $a_j$ has the property that all its prime divisors are $\leq B$ is $\psi(2\sqrt{n}, B)$ instead of $\psi(n, B)$. The expected running time is**

$$O\left(\exp(\sqrt{2}\sqrt{\log n}\sqrt{\log\log n})\right).$$

**Comment: Brillhart and Morrison (1970) used the CFRAC algorithm to factor $F_7 = 2^{2^7} + 1$ (having 39 digits).**

# A Further Idea

**An "early abort" strategy can be combined with the above ideas to reduce the running time of the algorithms. Given $a$, one stops trying to factor $s(a)$ if it has no "small" prime factors. This leads to a running time of the form**

$$O\left(\exp\left(\sqrt{3/2}\sqrt{\log n}\sqrt{\log\log n}\,\right)\right).$$

# The Quadratic Sieve Algorithm

**Consider** $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n.$

# The Quadratic Sieve Algorithm

**Consider $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n$. Then $|F(x)| \ll |x|\sqrt{n}$ for $0 < |x| \leq \sqrt{n}$.**

# The Quadratic Sieve Algorithm

**Consider $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n$. Then $|F(x)| \ll |x|\sqrt{n}$ for $0 < |x| \le \sqrt{n}$. The idea of the quadratic sieve algorithm is to consider $a$ in Dixon's Algorithm to be of the form $a = x + \lfloor\sqrt{n}\rfloor$ with $|x|$ small.**

# The Quadratic Sieve Algorithm

Consider $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n$. Then $|F(x)| \ll |x|\sqrt{n}$ for $0 < |x| \le \sqrt{n}$. The idea of the quadratic sieve algorithm is to consider $a$ in Dixon's Algorithm to be of the form $a = x+\lfloor\sqrt{n}\rfloor$ with $|x|$ small. Here, we allow for $s(a)$ to be negative as in the CFRAC Algorithm.

# The Quadratic Sieve Algorithm

Consider $F(x) = (x + \lfloor \sqrt{n} \rfloor)^2 - n$. Then $|F(x)| \ll |x|\sqrt{n}$ for $0 < |x| \le \sqrt{n}$. The idea of the quadratic sieve algorithm is to consider $a$ in Dixon's Algorithm to be of the form $a = x + \lfloor \sqrt{n} \rfloor$ with $|x|$ small. Here, we allow for $s(a)$ to be negative as in the CFRAC Algorithm. Thus, $|s(a)| \ll |x|\sqrt{n}$.

Why would this be better than the CFRAC Algorithm?

# Why would this be better than the CFRAC Algorithm?

**For $n$ fixed, $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n$ is a fixed quadratic polynomial that is used to obtain $a$'s to apply the approach in Dixon's algorithm.**

## Why would this be better than the CFRAC Algorithm?

**For $n$ fixed, $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n$ is a fixed quadratic polynomial that is used to obtain $a$'s to apply the approach in Dixon's algorithm. For $p$ a small prime, solve**

$$(*) \qquad F(x) \equiv 0 \pmod{p}.$$

## Why would this be better than the CFRAC Algorithm?

**For $n$ fixed, $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n$ is a fixed quadratic polynomial that is used to obtain $a$'s to apply the approach in Dixon's algorithm. For $p$ a small prime, solve**

$$(*) \qquad F(x) \equiv 0 \pmod{p}.$$

**If $(*)$ has one solution, typically it has two, say $x_1$ and $x_2$.**

## Why would this be better than the CFRAC Algorithm?

For $n$ fixed, $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n$ is a fixed quadratic polynomial that is used to obtain $a$'s to apply the approach in Dixon's algorithm. For $p$ a small prime, solve

$$(*) \qquad F(x) \equiv 0 \pmod{p}.$$

If $(*)$ has one solution, typically it has two, say $x_1$ and $x_2$. If $x \equiv x_1$ or $x_2$ modulo $p$ and $a = x + \lfloor\sqrt{n}\rfloor$, then $p|s(a)$.

## Why would this be better than the CFRAC Algorithm?

For $n$ fixed, $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n$ is a fixed quadratic polynomial that is used to obtain $a$'s to apply the approach in Dixon's algorithm. For $p$ a small prime, solve

$$(*) \qquad F(x) \equiv 0 \pmod{p}.$$

If $(*)$ has one solution, typically it has two, say $x_1$ and $x_2$. If $x \equiv x_1$ or $x_2$ modulo $p$ and $a = x + \lfloor\sqrt{n}\rfloor$, then $p|s(a)$. Otherwise, $p \nmid s(a)$.

## Why would this be better than the CFRAC Algorithm?

For $n$ fixed, $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n$ is a fixed quadratic polynomial that is used to obtain $a$'s to apply the approach in Dixon's algorithm. For $p$ a small prime, solve

$$(*) \qquad F(x) \equiv 0 \pmod{p}.$$

If $(*)$ has one solution, typically it has two, say $x_1$ and $x_2$. If $x \equiv x_1$ or $x_2$ modulo $p$ and $a = x + \lfloor\sqrt{n}\rfloor$, then $p | s(a)$. Otherwise, $p \nmid s(a)$. Therefore, one knows the different $x$'s and hence the different $a$'s for which $p | s(a)$ when $p$ is small and can partially factor the $s(a)$'s quickly.

The expected running time is

$$O\left(\exp\left(\sqrt{9/8}\sqrt{\log n}\sqrt{\log\log n}\,\right)\right)$$

for the Quadratic Sieve Algorithm.

# Why would this be better than the CFRAC Algorithm?

For $n$ fixed, $F(x) = (x + \lfloor\sqrt{n}\rfloor)^2 - n$ is a fixed quadratic polynomial that is used to obtain $a$'s to apply the approach in Dixon's algorithm. For $p$ a small prime, solve

$$(*) \qquad F(x) \equiv 0 \pmod{p}.$$

If $(*)$ has one solution, typically it has two, say $x_1$ and $x_2$. If $x \equiv x_1$ or $x_2$ modulo $p$ and $a = x + \lfloor\sqrt{n}\rfloor$, then $p|s(a)$. Otherwise, $p \nmid s(a)$. Therefore, one knows the different $x$'s and hence the different $a$'s for which $p|s(a)$ when $p$ is small and can partially factor the $s(a)$'s quickly.

Comment: A variation of the Quadratic Sieve Algorithm developed by Peter Montgomery reduces the running time to

$$O\left(\exp\left((1/2)\sqrt{\log n}\sqrt{\log\log n}\,\right)\right).$$

# The Number Field Sieve

**Preliminaries: Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$.**

# The Number Field Sieve

**Preliminaries:** Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

for some positive integer $d$ and each $c_j \in \{0, 1, \ldots, b-1\}$.

# The Number Field Sieve

**Preliminaries:** Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

for some positive integer $d$ and each $c_j \in \{0, 1, \ldots, b-1\}$. Set $f(x) = \sum_{j=0}^{d} c_j x^j$.

# The Number Field Sieve

**Preliminaries:** Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

for some positive integer $d$ and each $c_j \in \{0, 1, \ldots, b-1\}$. Set $f(x) = \sum_{j=0}^{d} c_j x^j$. Then one of the following holds:

(i) The polynomial $f(x)$ is irreducible over $\mathbb{Q}[x]$.

(ii) The polynomial $f(x) = g(x)h(x)$ for $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, and $n = g(b)h(b)$ is a non-trivial factorization of $n$.

**Comment 1:** The conclusion (ii) holds for every non-trivial factorization $f(x) = g(x)h(x)$.

**Comment 2:** What does this mean if $n$ is a prime?

**Theorem (F., Gross)** ***Let $f(x)$ be a polynomial with non-negative integer coefficients with $f(10)$ prime. If each of the coefficients of $f(x)$ is***

$$\le 49598666989151226098104244512918,$$

***then $f(x)$ is irreducible. Furthermore, if each coefficient is***

$$\le 8592444743529135815769545955936773$$

***and $f(x)$ is reducible, then $f(x)$ is divisible by $x^2-20x+101$.***

**Comment: The result is sharp. If either of the big numbers above is increased by 1, the theorem is no longer true.**

# The Number Field Sieve

Preliminaries: Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

for some positive integer $d$ and each $c_j \in \{0, 1, \ldots, b-1\}$. Set $f(x) = \sum_{j=0}^{d} c_j x^j$. Then one of the following holds:

(i) The polynomial $f(x)$ is irreducible over $\mathbb{Q}[x]$.

(ii) The polynomial $f(x) = g(x)h(x)$ for $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, and $n = g(b)h(b)$ is a non-trivial factorization of $n$.

Comment 1: The conclusion (ii) holds for every non-trivial factorization $f(x) = g(x)h(x)$.

Comment 2: What does this mean if $n$ is a prime?

Preliminaries: Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

for some positive integer $d$ and each $c_j \in \{0, 1, \ldots, b-1\}$. Set $f(x) = \sum_{j=0}^{d} c_j x^j$. Then one of the following holds:

(i) The polynomial $f(x)$ is irreducible over $\mathbb{Q}[x]$.

(ii) The polynomial $f(x) = g(x)h(x)$ for $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, and $n = g(b)h(b)$ is a non-trivial factorization of $n$.

Idea: The polynomial $f(x)$ cannot have a root in

$$\mathcal{D} = \{z \in \mathbb{C} : |z - b| \leq 1\}.$$

If $f(x) = g(x)h(x)$ with $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, then both $|g(b)| > 1$ and $|h(b)| > 1$.

**Preliminaries:** Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

for some positive integer $d$ and each $c_j \in \{0, 1, \ldots, b-1\}$. Set $f(x) = \sum_{j=0}^{d} c_j x^j$. Then one of the following holds:

(i) The polynomial $f(x)$ is irreducible over $\mathbb{Q}[x]$.

(ii) The polynomial $f(x) = g(x)h(x)$ for $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, and $n = g(b)h(b)$ is a non-trivial factorization of $n$.

**Idea:** The polynomial $f(x)$ cannot have a root in

$$\mathcal{D} = \{z \in \mathbb{C} : |z - b| \leq 1\}.$$

If $f(x) = g(x)h(x)$ with $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, then both $|g(b)| > 1$ and $|h(b)| > 1$.

**Preliminaries:** Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

for some positive integer $d$ and each $c_j \in \{0, 1, \ldots, b-1\}$. Set $f(x) = \sum_{j=0}^{d} c_j x^j$. Then one of the following holds:

(i) The polynomial $f(x)$ is irreducible over $\mathbb{Q}[x]$.

(ii) The polynomial $f(x) = g(x)h(x)$ for $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, and $n = g(b)h(b)$ is a non-trivial factorization of $n$.

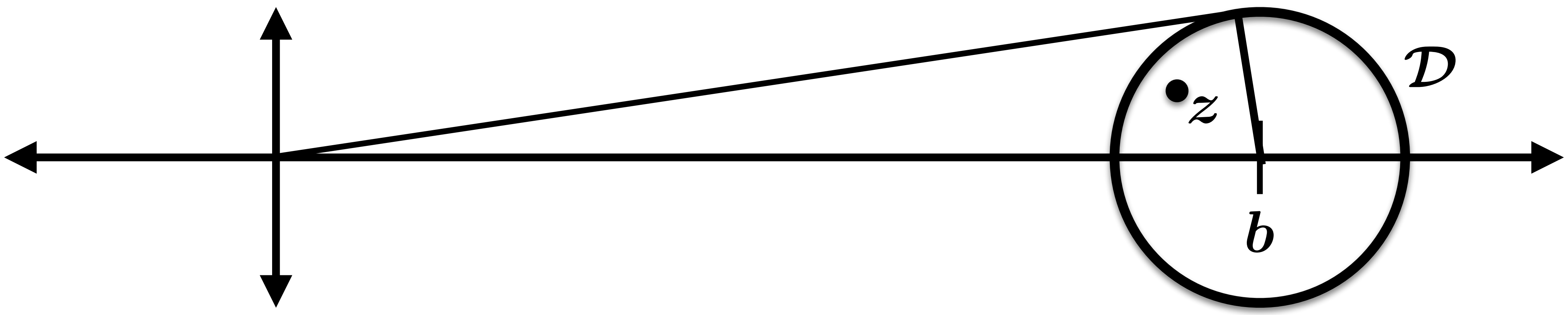
**Idea:** The polynomial $f(x)$ cannot have a root in

$$\mathcal{D} = \{z \in \mathbb{C} : |z - b| \leq 1\}.$$

If $f(x) = g(x)h(x)$ with $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, then both $|g(b)| > 1$ and $|h(b)| > 1$.

The polynomial $f(x)$ cannot have a root in

$$\mathcal{D} = \{z \in \mathbb{C} : |z - b| \leq 1\}.$$



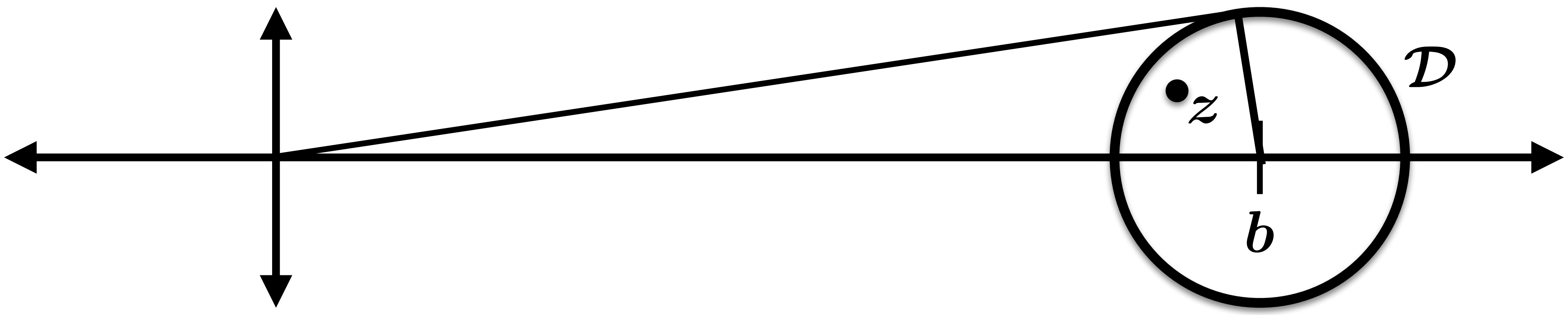

$$z = re^{i\theta}, \quad r \geq b - 1, \quad 0 < \theta < \sin^{-1}(1/b) < \pi/4$$

$$\left|\frac{f(z)}{z^d}\right|$$

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0$$

**The polynomial $f(x)$ cannot have a root in**

$$\mathcal{D} = \{z \in \mathbb{C} : |z - b| \leq 1\}.$$



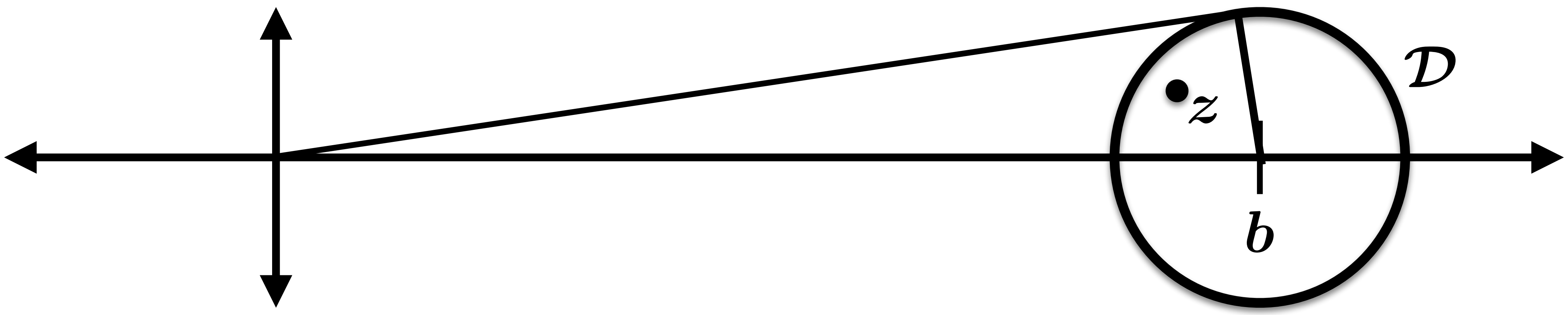

$$z = re^{i\theta}, \quad r \geq b - 1, \quad 0 < \theta < \sin^{-1}(1/b) < \pi/4$$

$$\left| \frac{f(z)}{z^d} \right| \geq \left| c_d + \frac{c_{d-1}}{z} \right| - \sum_{j=2}^{d} \frac{b-1}{(b-1)^j}$$

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0$$

**The polynomial $f(x)$ cannot have a root in**

$$\mathcal{D} = \{z \in \mathbb{C} : |z - b| \leq 1\}.$$

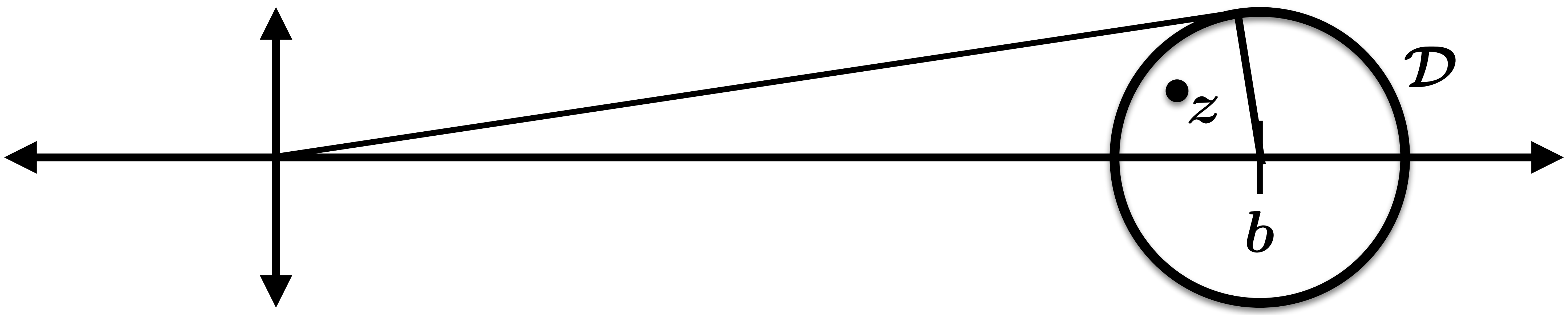



$$z = re^{i\theta}, \quad r \geq b - 1, \quad 0 < \theta < \sin^{-1}(1/b) < \pi/4$$

$$\left|\frac{f(z)}{z^d}\right| \geq \left|c_d + \frac{c_{d-1}}{z}\right| - \sum_{j=2}^{d} \frac{b-1}{(b-1)^j} > 1 - \frac{1}{b-2}$$

**The polynomial $f(x)$ cannot have a root in**

$$\mathcal{D} = \{z \in \mathbb{C} : |z - b| \leq 1\}.$$




$$z = re^{i\theta}, \quad r \geq b - 1, \quad 0 < \theta < \sin^{-1}(1/b) < \pi/4$$

$$\left|\frac{f(z)}{z^d}\right| \geq \left|c_d + \frac{c_{d-1}}{z}\right| - \sum_{j=2}^{d} \frac{b-1}{(b-1)^j} > 1 - \frac{1}{b-2} \geq 0$$

**Preliminaries:** Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

for some positive integer $d$ and each $c_j \in \{0, 1, \ldots, b-1\}$. Set $f(x) = \sum_{j=0}^{d} c_j x^j$. Then one of the following holds:

(i) The polynomial $f(x)$ is irreducible over $\mathbb{Q}[x]$.

(ii) The polynomial $f(x) = g(x)h(x)$ for $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, and $n = g(b)h(b)$ is a non-trivial factorization of $n$.

**Idea:** The polynomial $f(x)$ cannot have a root in

$$\mathcal{D} = \{z \in \mathbb{C} : |z - b| \leq 1\}.$$

If $f(x) = g(x)h(x)$ with $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, then both $|g(b)| > 1$ and $|h(b)| > 1$.

# The Number Field Sieve

**Let $f$ be an irreducible monic polynomial in $\mathbb{Z}[x]$.**

# The Number Field Sieve

Let $f$ be an irreducible monic polynomial in $\mathbb{Z}[x]$. Let $\alpha$ be a root of $f$. Let $m$ be an integer for which $f(m) \equiv 0 \pmod{n}$.

Preliminaries: Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

for some positive integer $d$ and each $c_j \in \{0, 1, \ldots, b-1\}$. Set $f(x) = \sum_{j=0}^{d} c_j x^j$. Then one of the following holds:

(i) The polynomial $f(x)$ is irreducible over $\mathbb{Q}[x]$.

(ii) The polynomial $f(x) = g(x)h(x)$ for $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, and $n = g(b)h(b)$ is a non-trivial factorization of $n$.

Comment: We can use $f(x)$ above and $m = b = \lfloor n^{1/d} \rfloor$.

# The Number Field Sieve

Let $f$ be an irreducible monic polynomial in $\mathbb{Z}[x]$. Let $\alpha$ be a root of $f$. Let $m$ be an integer for which $f(m) \equiv 0 \pmod{n}$.

Preliminaries: Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

for some positive integer $d$ and each $c_j \in \{0, 1, \dots, b-1\}$. Set $f(x) = \sum_{j=0}^{d} c_j x^j$. Then one of the following holds:

(i) The polynomial $f(x)$ is irreducible over $\mathbb{Q}[x]$.

(ii) The polynomial $f(x) = g(x)h(x)$ for $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, and $n = g(b)h(b)$ is a non-trivial factorization of $n$.

Comment: We can use $f(x)$ above and $m = b = \lfloor n^{1/d} \rfloor$.

Is $f(x)$ monic?

# The Number Field Sieve

**Let $f$ be an irreducible monic polynomial in $\mathbb{Z}[x]$. Let $\alpha$ be a root of $f$. Let $m$ be an integer for which $f(m) \equiv 0 \pmod{n}$.**

**Preliminaries: Let $n$ be a large positive integer, and let $b$ be an integer $\geq 3$ smaller than $n$. Suppose we write $n$ in base $b$, so**

$$n = c_d b^d + c_{d-1} b^{d-1} + \cdots + c_1 b + c_0,$$

**for some positive integer $d$ and each $c_j \in \{0, 1, \ldots, b-1\}$. Set $f(x) = \sum_{j=0}^{d} c_j x^j$. Then one of the following holds:**

**(i) The polynomial $f(x)$ is irreducible over $\mathbb{Q}[x]$.**

**(ii) The polynomial $f(x) = g(x)h(x)$ for $g(x)$ and $h(x)$ in $\mathbb{Z}[x]$, and $n = g(b)h(b)$ is a non-trivial factorization of $n$.**

**How long does it take to factor $f(x)$?**

# The Number Field Sieve

**Let $f$ be an irreducible monic polynomial in $\mathbb{Z}[x]$. Let $\alpha$ be a root of $f$. Let $m$ be an integer for which $f(m) \equiv 0 \pmod{n}$. The mapping $\phi : \mathbb{Z}[\alpha] \to \mathbb{Z}_n$ with $\phi(g(\alpha)) = g(m) \bmod n$ for all $g(x) \in \mathbb{Z}[x]$ is a homomorphism. (Recall what $\mathbb{Z}[\alpha]$ is.)**

# The Number Field Sieve

**Let $f$ be an irreducible monic polynomial in $\mathbb{Z}[x]$. Let $\alpha$ be a root of $f$. Let $m$ be an integer for which $f(m) \equiv 0 \pmod{n}$. The mapping $\phi : \mathbb{Z}[\alpha] \rightarrow \mathbb{Z}_n$ with $\phi(g(\alpha)) = g(m) \bmod n$ for all $g(x) \in \mathbb{Z}[x]$ is a homomorphism. (Recall what $\mathbb{Z}[\alpha]$ is.) The idea is to find a set $S$ of polynomials $g(x) \in \mathbb{Z}[x]$ such that both of the following hold:**

**(i)** $\prod_{g \in S} g(m) = y^2$ **for some** $y \in \mathbb{Z}$

**(ii)** $\prod_{g \in S} g(\alpha) = \beta^2$ **for some** $\beta \in \mathbb{Z}[\alpha]$.

**Taking $x = \phi(\beta)$, we deduce**

$$x^2 \equiv \phi(\beta)^2 \equiv \phi(\beta^2) \equiv \phi\left(\prod_{g \in S} g(\alpha)\right) \equiv \prod_{g \in S} g(m) \equiv y^2 \pmod{n}.$$

**Thus, we can hope to factor $n$ by computing $\gcd(x + y, n)$.**