The four squares problem and its application in operator approximation

Xiaoning Bian

Dalhousie University

December 1, 2020

$O(n^2)$ vs $O((\log n)^2 \log\log n)$

$O((\log n)^2 / \log\log n)$

Abstract

I will first walk through an algorithm of Rabin and Shallit [2, 1] that efficiently solves the four-square Diophantine equation $n = x^2 + y^2 + z^2 + w^2$. The efficiency comes from the use of randomness — there are enough "good seed number", so by randomly choosing a number, it is likely that we can hit a good seed that will grow into a solution. Then, I will explain how this algorithm can be adapted to solve the problem of approximating any 2x2 unitary using matrices of a certain kind. The resulting algorithm is a "baby" version of Ross and Selinger's algorithm[4, 3].

Abstract

I will first walk through an algorithm of Rabin and Shallit [2, 1] that efficiently solves the four-square Diophantine equation $n = x^2 + y^2 + z^2 + w^2$. The efficiency comes from the use of randomness — there are enough "good seed number", so by randomly choosing a number, it is likely that we can hit a good seed that will grow into a solution. Then, I will explain how this algorithm can be adapted to solve the problem of approximating any 2x2 unitary using matrices of a certain kind. The resulting algorithm is a "baby" version of Ross and Selinger's algorithm[4, 3].

$x \xmapsto{f}$ a solution

Abstract

I will first walk through an algorithm of Rabin and Shallit [2, 1] that efficiently solves the four-square Diophantine equation $n = x^2 + y^2 + z^2 + w^2$. The efficiency comes from the use of randomness — there are enough "good seed number", so by randomly choosing a number, it is likely that we can hit a good seed that will grow into a solution. Then, I will explain how this algorithm can be adapted to solve the problem of approximating any 2x2 unitary using matrices of a certain kind. The resulting algorithm is a "baby" version of Ross and Selinger's algorithm[4, 3].

Outline

Randomized algorithms
- Solving simple equations randomly
- Two-square algorithm
- Four-square algorithm

Operator approximation
- Dense subsets of $\mathbb{C}^n$
- Unit vector approximation
- Unitary operator approximation

Solving simple equations randomly

Question

Let p be a prime. Design an algorithm that finds *one* solution of $x^{p-1} \equiv 1 \pmod{p}$ (besides the trivial solution 1).

Notation

Let $\mathbb{Z}_p$ be the usual finite field, $\mathbb{Z}_p^* = \mathbb{Z}_p \setminus \{0\}$ the multiplicative group of $\mathbb{Z}_p$.

Answer

By Fermat's little theorem, every number in $\mathbb{Z}_p^*$ is a solution.
Algorithm:

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. return a, finish.

Complexity: $O(1)$.

Solving simple equations randomly

Question

Let p be a prime. Design an algorithm that finds *one* solution of $x^{p-1} \equiv 1 \pmod{p}$ (besides the trivial solution 1).

Notation

Let $\mathbb{Z}_p$ be the usual finite field, $\mathbb{Z}_p^* = \mathbb{Z}_p \setminus \{0\}$ the multiplicative group of $\mathbb{Z}_p$.

Answer

By Fermat's little theorem, every number in $\mathbb{Z}_p^*$ is a solution.
Algorithm:

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. return a, finish.

Complexity: $O(1)$.

Solving simple equations randomly

Question

Let p be a prime. Design an algorithm that finds *one* solution of $x^{p-1} \equiv 1 \pmod{p}$ (besides the trivial solution 1).

Notation

Let $\mathbb{Z}_p$ be the usual finite field, $\mathbb{Z}_p^* = \mathbb{Z}_p \setminus \{0\}$ the multiplicative group of $\mathbb{Z}_p$.

Answer

By Fermat's little theorem, every number in $\mathbb{Z}_p^*$ is a solution.
Algorithm:

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. return a, finish.

$x - x = 0$

Complexity: $O(1)$.

Solving simple equations randomly

Question

Let p be a prime such that $p \equiv 1 \pmod 4$. Design an algorithm that finds *one* solution (besides the trivial 1) of $x^2 \equiv 1 \pmod p$.

Answer

Let $p = 4k + 1$. By Fermat's little theorem, for $x \in \mathbb{Z}_p^*$, we have

$$x^{4k} \equiv 1 \pmod p, \text{i.e., } (x^{2k})^2 \equiv 1 \pmod p.$$

Algorithm:

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. calculate $b \equiv a^{2k} \pmod p$, using "repeated squaring".
3. return b, finish.

Complexity: $O(1) + O(\log(2k)) = O(\log p)$.

Solving simple equations randomly

Question

Let p be a prime such that $p \equiv 1 \pmod 4$. Design an algorithm that finds *one* solution (besides the trivial 1) of $x^2 \equiv 1 \pmod p$.

Answer

Let $p = 4k + 1$. By Fermat's little theorem, for $x \in \mathbb{Z}_p^*$, we have

$$x^{4k} \equiv 1 \pmod p, \text{i.e., } (x^{2k})^2 \equiv 1 \pmod p.$$

Algorithm:

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. calculate $b \equiv a^{2k} \pmod p$, using "repeated squaring".
3. return b, finish.

Complexity: $O(1) + O(\log(2k)) = O(\log p)$.

Solving simple equations randomly

Question

Let p be a prime such that $p \equiv 1 \pmod 4$. Design an algorithm that finds *one* solution (besides the trivial 1) of $x^2 \equiv 1 \pmod p$.

Answer

Let $p = 4k + 1$. By Fermat's little theorem, for $x \in \mathbb{Z}_p^*$, we have

$$x^{4k} \equiv 1 \pmod p, \text{i.e., } (x^{2k})^2 \equiv 1 \pmod p.$$

Algorithm: $a^1\ a^2\ a^3 \ldots\ a^{2k}$

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. calculate $b \equiv a^{2k} \pmod p$, using "repeated squaring".
3. return b, finish.

$a^1\ a^2\ a^4\ a^8\ a^{16}$

Complexity: $O(1) + O(\log(2k)) = O(\log p)$.

$2k = 10$

Solving simple equations randomly

$\in O(n^2)$

$\in O((\log_2 n)^2 \log_2 \log_2 n)$

Question

Let p be a prime such that $p \equiv 1 \pmod 4$. Design an algorithm that finds *one* solution (besides the trivial 1) of $x^2 \equiv 1 \pmod p$.

Answer

Let $p = 4k + 1$. By Fermat's little theorem, for $x \in \mathbb{Z}_p^*$, we have

$$x^{4k} \equiv 1 \pmod p, \text{i.e., } (x^{2k})^2 \equiv 1 \pmod p.$$

O

$\cancel{O((2k))}$

Algorithm:

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. calculate $b \equiv a^{2k} \pmod p$, using "repeated squaring".
3. return b, finish.

Complexity: $O(1) + O(\log(2k)) = O(\log p)$.

Solving simple equations randomly

Question

Let $p = 4k + 1$ be a prime. Design an algorithm that finds *one* solution of $x^2 = -1$ in $\mathbb{Z}_p$.

Answer

By Fermat's little theorem, for $x \in \mathbb{Z}_p$, we have

$$x^{4k} = 1, \text{i.e.}, (x^{2k})^2 = 1, \text{i.e.}, x^{2k} = (x^k)^2 = 1 \text{ or } -1.$$

Claim: Half of $\mathbb{Z}_p^*$ satisfy $(x^k)^2 = -1$. Will show later.
Algorithm:

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. calculate $b = a^k$, using "repeated squaring".
3. check if $b^2 = -1$. If yes return b, finish. If not, go to step 1.

Solving simple equations randomly

Question

Let $p = 4k + 1$ be a prime. Design an algorithm that finds *one* solution of $x^2 = -1$ in $\mathbb{Z}_p$.

Answer

By Fermat's little theorem, for $x \in \mathbb{Z}_p$, we have

$$x^{4k} = 1, \text{i.e., } (x^{2k})^2 = 1, \text{i.e., } x^{2k} = (x^k)^2 = 1 \text{ or } -1.$$

$\sqrt{1}$

Claim: Half of $\mathbb{Z}_p^*$ satisfy $(x^k)^2 = -1$. Will show later.

Algorithm:

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. calculate $b = a^k$, using "repeated squaring".
3. check if $b^2 = -1$. If yes return b, finish. If not, go to step 1.

Solving simple equations randomly

Question

Let $p = 4k + 1$ be a prime. Design an algorithm that finds *one* solution of $x^2 = -1$ in $\mathbb{Z}_p$.

Answer

By Fermat's little theorem, for $x \in \mathbb{Z}_p$, we have

$$x^{4k} = 1, \text{i.e., } (x^{2k})^2 = 1, \text{i.e., } x^{2k} = (x^k)^2 = 1 \text{ or } -1.$$

Claim: Half of $\mathbb{Z}_p^*$ satisfy $(x^k)^2 = -1$. Will show later.

Algorithm:

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. calculate $b = a^k$, using "repeated squaring".
3. check if $b^2 = -1$. If yes return b, finish. If not, go to step 1.

Solving simple equations randomly

Question

Let $p = 4k + 1$ be a prime. Design an algorithm that finds *one* solution of $x^2 = -1$ in $\mathbb{Z}_p$.

Answer

By Fermat's little theorem, for $x \in \mathbb{Z}_p$, we have

$$x^{4k} = 1, \text{i.e., } (x^{2k})^2 = 1, \text{i.e., } x^{2k} = (x^k)^2 = 1 \text{ or } -1.$$

Claim: Half of $\mathbb{Z}_p^*$ satisfy $(x^k)^2 = -1$. Will show later.
Algorithm:

1. randomly pick a number a in $\mathbb{Z}_p^*$.
2. calculate $b = \underline{a^k}$, using "repeated squaring".
3. check if $b^2 = -1$. If yes return b, finish. If not, go to step 1.

Complexity analysis

$1 \quad 1 \quad 1 \quad 1$

$\frac{1}{2} \quad \frac{1}{4} \quad \frac{1}{8} \quad \ldots$

Expected Complexity:

$$O(\log p)\left(\sum_{i=0}^{\infty}\left(\frac{1}{2}\right)^{i}\right) = 2O(\log p) = O(\log p).$$

Or compute in an easier way: succeed with probality 0.5, on average, we need try $\frac{1}{0.5} = 2$ times to succeed.

Note

Step 3 only takes constant time. In general, randomized algorithms are suitable for solving NP problems — problems that may or may not need polynomial time to solve, but only need polynomial time to check. But randomized algorithms cannot solve NP-hard problems.

Complexity analysis

Expected Complexity:

$$O(\log p)\left(\sum_{i=0}^{\infty}\left(\frac{1}{2}\right)^i\right) = 2O(\log p) = O(\log p).$$

Or compute in an easier way: succeed with probality 0.5, on average, we need try $\frac{1}{0.5} = 2$ times to succeed.

Note

Step 3 only takes constant time. In general, randomized algorithms are suitable for solving NP problems — problems that may or may not need polynomial time to solve, but only need polynomial time to check. But randomized algorithms cannot solve NP-hard problems.

Complexity analysis

Expected Complexity:

$$O(\log p)\left(\sum_{i=0}^{\infty}\left(\frac{1}{2}\right)^i\right) = 2O(\log p) = O(\log p).$$

Or compute in an easier way: succeed with probality 0.5, on average, we need try $\frac{1}{0.5} = 2$ times to succeed.

Note

$b^2 = -1$

Step 3 only takes constant time. In general, randomized algorithms are suitable for solving NP problems — problems that may or may not need polynomial time to solve, but only need polynomial time to check. But randomized algorithms cannot solve NP-hard problems.

Complexity analysis

Expected Complexity:

$$O(\log p)\left(\sum_{i=0}^{\infty}\left(\frac{1}{2}\right)^i\right) = 2O(\log p) = O(\log p).$$

Or compute in an easier way: succeed with probality 0.5, on average, we need try $\frac{1}{0.5} = 2$ times to succeed.

Note

Step 3 only takes constant time. In general, randomized algorithms are suitable for solving NP problems — problems that may or may not need polynomial time to solve, but only need polynomial time to check. But randomized algorithms cannot solve NP-hard problems.

Proof of the claim

Claim

Half of $\mathbb{Z}_p^*$ satisfy $(x^k)^2 = -1$.

Proof.

Recall Fermat's little theorem implies

$$x^{4k} = 1, \text{i.e., } (x^{2k})^2 = 1, \text{i.e., } x^{2k} = 1 \text{ or } -1.$$

Since $\mathbb{Z}_p$ is a field, the equations $x^{2k} = 1$ and $x^{2k} = -1$ each has at most $2k$ solutions. And there are $4k$ elements in $\mathbb{Z}_p^*$. $\square$

Proof of the claim

Claim

Half of $\mathbb{Z}_p^*$ satisfy $(x^k)^2 = -1$.

Proof.

Recall Fermat's little theorem implies

$$x^{4k} = 1, \text{i.e., } (x^{2k})^2 = 1, \text{i.e., } x^{2k} = 1 \text{ or } -1.$$

Since $\mathbb{Z}_p$ is a field, the equations $x^{2k} = 1$ and $x^{2k} = -1$ each has at most $2k$ solutions. And there are $4k$ elements in $\mathbb{Z}_p^*$. □

Solving $x^2 + y^2 = p$ randomly

Question

Let $p = 4k + 1$ be a prime. Design an algorithm that finds *one* solution of $x^2 + y^2 = p$ in $\mathbb{Z}$.

Answer

Let $\mathbb{Z}[i]$ be the set of Gaussian Integers, which is a Euclidean Domain, and the norm $N(a + bi) = a^2 + b^2$ is the rank function. Let $u \in \mathbb{Z}_p$ be a solution of $x^2 = -1$ obtained from the last algorithm. Then in $\mathbb{Z}[i]$, we have $(u + i)(u - i) = u^2 + 1 = mp$. Let $a + bi = \gcd(u + i, p)$. We claim $a^2 + b^2 = p$.

Algorithm:

1. run the last algorithm to obtain u s.t. $u^2 \equiv -1 \pmod{p}$.
2. calculate $a + bi = \gcd(u + i, p)$.
3. return (a, b), finish.

Complexity: Step 1 and 2, $O(\log p)$. Total, $O(\log p)$.

Solving $x^2 + y^2 = p$ randomly

Question

Let $p = 4k + 1$ be a prime. Design an algorithm that finds *one* solution of $x^2 + y^2 = p$ in $\mathbb{Z}$.

Answer

Let $\mathbb{Z}[i]$ be the set of Gaussian Integers, which is a Euclidean Domain, and the norm $N(a + bi) = a^2 + b^2$ is the rank function.
Let $u \in \mathbb{Z}_p$ be a solution of $x^2 = -1$ obtained from the last algorithm. Then in $\mathbb{Z}[i]$, we have $(u + i)(u - i) = u^2 + 1 = mp$.
Let $a + bi = \gcd(u + i, p)$. We claim $a^2 + b^2 = p$.
Algorithm:

1. run the last algorithm to obtain u s.t. $u^2 \equiv -1 \pmod{p}$.
2. calculate $a + bi = \gcd(u + i, p)$.
3. return (a, b), finish.

Complexity: Step 1 and 2, $O(\log p)$. Total, $O(\log p)$.

Solving $x^2 + y^2 = p$ randomly

Question

Let $p = 4k + 1$ be a prime. Design an algorithm that finds *one* solution of $x^2 + y^2 = p$ in $\mathbb{Z}$.

Answer

Let $\mathbb{Z}[i]$ be the set of Gaussian Integers, which is a Euclidean Domain, and the norm $N(a + bi) = a^2 + b^2$ is the rank function. Let $u \in \mathbb{Z}_p$ be a solution of $x^2 = -1$ obtained from the last algorithm. Then in $\mathbb{Z}[i]$, we have $(u + i)(u - i) = u^2 + 1 = mp$.

Let $a + bi = \gcd(u + i, p)$. We claim $a^2 + b^2 = p$.

Algorithm:

1. run the last algorithm to obtain u s.t. $u^2 \equiv -1 \pmod{p}$.
2. calculate $a + bi = \gcd(u + i, p)$.
3. return (a, b), finish.

Complexity: Step 1 and 2, $O(\log p)$. Total, $O(\log p)$.

Solving $x^2 + y^2 = p$ randomly

Question

Let $p = 4k + 1$ be a prime. Design an algorithm that finds *one* solution of $x^2 + y^2 = p$ in $\mathbb{Z}$.

Answer

Let $\mathbb{Z}[i]$ be the set of Gaussian Integers, which is a Euclidean Domain, and the norm $N(a + bi) = a^2 + b^2$ is the rank function. Let $u \in \mathbb{Z}_p$ be a solution of $x^2 = -1$ obtained from the last algorithm. Then in $\mathbb{Z}[i]$, we have $(u + i)(u - i) = u^2 + 1 = mp$. Let $a + bi = \text{gcd}(u + i, p)$. We claim $a^2 + b^2 = p$.

$O(\log N(p))$

Algorithm:

1. run the last algorithm to obtain u s.t. $u^2 \equiv -1 \pmod{p}$.
2. calculate $a + bi = \text{gcd}(u + i, p)$.
3. return (a, b), finish.

Complexity: Step 1 and 2, $O(\log p)$. Total, $O(\log p)$.

Solving $x^2 + y^2 = p$ randomly

Question

Let $p = 4k + 1$ be a prime. Design an algorithm that finds *one* solution of $x^2 + y^2 = p$ in $\mathbb{Z}$.

Answer

Let $\mathbb{Z}[i]$ be the set of Gaussian Integers, which is a Euclidean Domain, and the norm $N(a + bi) = a^2 + b^2$ is the rank function. Let $u \in \mathbb{Z}_p$ be a solution of $x^2 = -1$ obtained from the last algorithm. Then in $\mathbb{Z}[i]$, we have $(u + i)(u - i) = u^2 + 1 = mp$. Let $a + bi = \gcd(u + i, p)$. We claim $a^2 + b^2 = p$.

Algorithm:

1. run the last algorithm to obtain u s.t. $u^2 \equiv -1 \pmod{p}$.
2. calculate $a + bi = \gcd(u + i, p)$.
3. return (a, b), finish.

Complexity: Step 1 and 2, $O(\log p)$. Total, $O(\log p)$.

Solving $x^2 + y^2 = p$ randomly

$8 = 2^2 + 2^2$

Question

Let $p = 4k + 1$ be a prime. Design an algorithm that finds *one* solution of $x^2 + y^2 = p$ in $\mathbb{Z}$.

$7 = 2^2 + 1^2 + 1^2 + 1^2$

Answer

Let $\mathbb{Z}[i]$ be the set of Gaussian Integers, which is a Euclidean Domain, and the norm $N(a + bi) = a^2 + b^2$ is the rank function. Let $u \in \mathbb{Z}_p$ be a solution of $x^2 = -1$ obtained from the last algorithm. Then in $\mathbb{Z}[i]$, we have $(u + i)(u - i) = u^2 + 1 = mp$. Let $a + bi = \gcd(u + i, p)$. We claim $a^2 + b^2 = p$.

Algorithm:

1. run the last algorithm to obtain u s.t. $u^2 \equiv -1 \pmod{p}$.
2. calculate $a + bi = \gcd(u + i, p)$.
3. return (a, b), finish.

Complexity: Step 1 and 2, $O(\log p)$. Total, $O(\log p)$.

Proof of the claim

Claim

Let $a + bi = \gcd(u + i, p)$. We claim $a^2 + b^2 = p$.

Proof.

$\{0 \cdots p-1\}$

Recall we have $(u + i)(u - i) = u^2 + 1 = mp$. Since $u \in \mathbb{Z}_p$, we have $mp = u^2 + 1 < p^2$, i.e., $m < p$.

$$a + bi \mid u + i \implies N(a + bi) \mid N(u + i), \text{i.e., } a^2 + b^2 \mid mp \text{ in } \mathbb{Z} \quad (*)$$

$$a + bi \mid p \implies N(a + bi) \mid N(p), \text{i.e., } a^2 + b^2 \mid p^2 \text{ in } \mathbb{Z}.$$

which implies $a^2 + b^2 = 1$ or p or p^2. We can exclude p^2 by $(*)$. To exclude 1, suppose not, then $a + bi$ is a unit, so $u + i$ and p are relatively prime. Then $u + i \mid m$, which means $N(u + i) \mid N(m)$, i.e., $mp \mid m^2$, a contradiction. □

Proof of the claim

Claim

Let $a + bi = \gcd(u + i, p)$. We claim $a^2 + b^2 = p$. $\mathbb{Z}[i] \to \mathbb{Z}$

Proof.

Recall we have $(u + i)(u - i) = u^2 + 1 = mp$. Since $u \in \mathbb{Z}_p$, we have $mp = u^2 + 1 < p^2$, i.e., $m < p$.

$<p^2$

$$a + bi \mid u + i \implies N(a + bi) \mid N(u + i), \text{i.e., } a^2 + b^2 \mid mp \text{ in } \mathbb{Z} \quad (*)$$

$$a + bi \mid p \implies N(a + bi) \mid N(p), \text{i.e., } a^2 + b^2 \mid p^2 \text{ in } \mathbb{Z}.$$

which implies $a^2 + b^2 = 1$ or p or p^2. We can exclude p^2 by $(*)$.
To exclude 1, suppose not, then $a + bi$ is a unit, so $u + i$ and p are relatively prime. Then $u + i \mid m$, which means $N(u + i) \mid N(m)$, i.e., $mp \mid m^2$, a contradiction. □

Proof of the claim

Claim

Let $a + bi = \gcd(u + i, p)$. We claim $a^2 + b^2 = p$.

Proof.

Recall we have $(u + i)(u - i) = u^2 + 1 = mp$. Since $u \in \mathbb{Z}_p$, we have $mp = u^2 + 1 < p^2$, i.e., $m < p$.

$$a + bi \mid u + i \implies N(a + bi) \mid N(u + i), \text{i.e., } a^2 + b^2 \mid mp \text{ in } \mathbb{Z} \quad (*)$$

$$a + bi \mid p \implies N(a + bi) \mid N(p), \text{i.e., } a^2 + b^2 \mid p^2 \text{ in } \mathbb{Z}.$$

which implies $a^2 + b^2 = 1$ or p or p^2. We can exclude p^2 by $(*)$. To exclude 1, suppose not, then $a + bi$ is a unit, so $u + i$ and p are relatively prime. Then $u + i \mid m$, which means $N(u + i) \mid N(m)$, i.e., $mp \mid m^2$, a contradiction. □

$m < p$

Solving $x^2 + y^2 + z^2 + w^2 = \underbrace{4k + 2}_{n}$ randomly

Question

Let $n = 4k + 2$ be a non-negative integer. Design an algorithm that finds *one* solution of $x^2 + y^2 + z^2 + w^2 = n$ in $\mathbb{Z}$.

Answer

The idea is that by randomly choosing x and y, we might have $p := n - x^2 - y^2$ is a prime of the form $4a + 1$. Then using the two-square algorithm, we can find z, w s.t. $n - x^2 - y^2 = z^2 + w^2$.

First notice that, if p is a prime, then $p = 2$ or p odd and $p = n - x^2 - y^2 \equiv 2 - x^2 - y^2 \pmod 4 \implies p \equiv 1 \pmod 4$. This means we only need p to be a prime. For the abundance of such x and y, I need to use a theorem in [2] which says for a certain constant $A > 0$, there exists n_0 such that for $n > n_0$, the number of x, y's such that $n - x^2 - y^2$ is prime is greater than $\frac{A \cdot n}{(\log n) \log \log n}$, out of $\sqrt{n} \cdot \sqrt{n}$ choices for x, y.

Solving $x^2 + y^2 + z^2 + w^2 = 4k + 2$ randomly

Question

Let $n = 4k + 2$ be a non-negative integer. Design an algorithm that finds *one* solution of $x^2 + y^2 + z^2 + w^2 = n$ in $\mathbb{Z}$.

Answer

The idea is that by randomly choosing x and y, we might have $p := n - x^2 - y^2$ is a prime of the form $4a + 1$. Then using the two-square algorithm, we can find z, w s.t. $n - x^2 - y^2 = z^2 + w^2$.

First notice that, if p is a prime, then $p = 2$ or p odd and $p = n - x^2 - y^2 \equiv 2 - x^2 - y^2 \pmod 4 \implies p \equiv 1 \pmod 4$. This means we only need p to be a prime. For the abundance of such x and y, I need to use a theorem in [2] which says for a certain constant $A > 0$, there exists n_0 such that for $n > n_0$, the number of x, y's such that $n - x^2 - y^2$ is prime is greater than $\frac{A \cdot n}{(\log n) \log \log n}$, out of $\sqrt{n} \cdot \sqrt{n}$ choices for x, y.

Solving $x^2 + y^2 + z^2 + w^2 = 4k + 2$ randomly

Question

Let $n = 4k + 2$ be a non-negative integer. Design an algorithm that finds *one* solution of $x^2 + y^2 + z^2 + w^2 = n$ in $\mathbb{Z}$.

Answer

The idea is that by randomly choosing x and y, we might have $p := n - x^2 - y^2$ is a prime of the form $4a + 1$. Then using the two-square algorithm, we can find z, w s.t. $n - x^2 - y^2 = z^2 + w^2$.

$x^2 \equiv 0, 1 \pmod 4$

First notice that, if p is a prime, then $p = 2$ or p odd and $p = n - x^2 - y^2 \equiv 2 - x^2 - y^2 \pmod 4 \implies p \equiv 1 \pmod 4$. This means we only need p to be a prime. For the abundance of such x and y, I need to use a theorem in [2] which says for a certain constant $A > 0$, there exists n_0 such that for $n > n_0$, the number of x, y's such that $n - x^2 - y^2$ is prime is greater than $\frac{A \cdot n}{(\log n) \log \log n}$, out of $\sqrt{n} \cdot \sqrt{n}$ choices for x, y.

Solving $x^2 + y^2 + z^2 + w^2 = 4k + 2$ randomly

Question

Let $n = 4k + 2$ be a non-negative integer. Design an algorithm that finds *one* solution of $x^2 + y^2 + z^2 + w^2 = n$ in $\mathbb{Z}$.

Answer

The idea is that by randomly choosing x and y, we might have $p := n - x^2 - y^2$ is a prime of the form $4a + 1$. Then using the two-square algorithm, we can find z, w s.t. $n - x^2 - y^2 = z^2 + w^2$.

First notice that, if p is a prime, then $p = 2$ or p odd and $p = n - x^2 - y^2 \equiv 2 - x^2 - y^2 \pmod 4 \implies p \equiv 1 \pmod 4$. This means we only need p to be a prime. For the abundance of such x and y, I need to use a theorem in [2] which says for a certain constant $A > 0$, there exists n_0 such that for $n > n_0$, the number of x, y's such that $n - x^2 - y^2$ is prime is greater than $\frac{A \cdot n}{(\log n) \log \log n}$, out of $\sqrt{n} \cdot \sqrt{n}$ choices for x, y.

Solving $x^2 + y^2 + z^2 + w^2 = 4k + 2$ randomly

Question

Let $n = 4k + 2$ be a non-negative integer. Design an algorithm that finds *one* solution of $x^2 + y^2 + z^2 + w^2 = n$ in $\mathbb{Z}$.

Answer

$u^2 = -1$

Algorithm:

1. randomly choose $0 \le x, y \le \sqrt{n}$.
2. run two-square algorithm but only finite many steps, with input $n - x^2 - y^2$. If a solution z, w is found. return (x, y, z, w) and finish, otherwise go to step 1.

Complexity: First, notice the "density" of such x, y is $\frac{A \cdot n}{(\log n) \log\log n} \cdot \frac{1}{\sqrt{n}\sqrt{n}} = \frac{A}{(\log n) \log\log n}$, so on average we need to run $\frac{(\log n) \log\log n}{A}$ times to get a good pair of x, y such that $n - x^2 - y^2$ is a prime. For each such try, running finite many steps of the two-square algorithm costs $O(\log n)$. Total $O((\log n)^2 \log\log n)$.

Solving $x^2 + y^2 + z^2 + w^2 = 4k + 2$ randomly

Question

Let $n = 4k + 2$ be a non-negative integer. Design an algorithm that finds *one* solution of $x^2 + y^2 + z^2 + w^2 = n$ in $\mathbb{Z}$.

Answer

Algorithm:

1. randomly choose $0 \le x, y \le \sqrt{n}$.
2. run two-square algorithm but only finite many steps, with input $n - x^2 - y^2$. If a solution z, w is found, return (x, y, z, w) and finish, otherwise go to step 1.

Complexity: First, notice the "density" $\frac{1}{2}$ of such x, y is $\frac{A\cdot n}{(\log n)\log\log n} \cdot \frac{1}{\sqrt{n}\sqrt{n}} = \frac{A}{(\log n)\log\log n}$, so on average we need to run 1000 $\frac{(\log n)\log\log n}{A}$ times to get a good pair of x, y such that $n - x^2 - y^2$ is a prime. For each such try, running finite many steps of the two-square algorithm costs $O(\log n)$. Total $O((\log n)^2 \log\log n)$.

Solving $x^2 + y^2 + z^2 + w^2 = n$ randomly

Question

Let n be a non-negative integer. Design an algorithm that finds *one* solution of $x^2 + y^2 + z^2 + w^2 = n$ in $\mathbb{Z}$.

Answer

- $n = 4k$. Solve $n/4 = a^2 + b^2 + c^2 + d^2$, then $n = (2a)^2 + (2b)^2 + (2c)^2 + (2d)^2$.
- $n = 4k + 1$ or $4k + 3$. Solve $2n = 4(2k) + 2$ or $2n = 8k + 6 = 4(2k + 1) + 2$, say $2n = a^2 + b^2 + c^2 + d^2$, then $a^2 + b^2 + c^2 + d^2 \equiv 2 \pmod 4$, so two of a, b, c, d are odd, two are even. WLOG, say a, b odd, c, d even, then we have $n = \left(\frac{1}{2}(a+b)\right)^2 + \left(\frac{1}{2}(a-b)\right)^2 + \left(\frac{1}{2}(c+d)\right)^2 + \left(\frac{1}{2}(c-d)\right)^2$
- $n = 4k + 2$. Solved using last algorithm.

Solving $x^2 + y^2 + z^2 + w^2 = n$ randomly

Question

Let n be a non-negative integer. Design an algorithm that finds *one* solution of $x^2 + y^2 + z^2 + w^2 = n$ in $\mathbb{Z}$.

Answer

- $n = 4k$. Solve $n/4 = a^2 + b^2 + c^2 + d^2$, then $n = (2a)^2 + (2b)^2 + (2c)^2 + (2d)^2$.
- $n = 4k + 1$ or $4k + 3$. Solve $2n = 4(2k) + 2$ or $2n = 8k + 6 = 4(2k + 1) + 2$, say $2n = a^2 + b^2 + c^2 + d^2$, then $a^2 + b^2 + c^2 + d^2 \equiv 2 \pmod 4$, so two of a, b, c, d are odd, two are even. WLOG, say a, b odd, c, d even, then we have $n = \left(\frac{1}{2}(a+b)\right)^2 + \left(\frac{1}{2}(a-b)\right)^2 + \left(\frac{1}{2}(c+d)\right)^2 + \left(\frac{1}{2}(c-d)\right)^2$
- $n = 4k + 2$. Solved using last algorithm.

Solving $x^2 + y^2 + z^2 + w^2 = n$ randomly

Question

Let n be a non-negative integer. Design an algorithm that finds *one* solution of $x^2 + y^2 + z^2 + w^2 = n$ in $\mathbb{Z}$.

Answer

- $n = 4k$. Solve $n/4 = a^2 + b^2 + c^2 + d^2$, then $n = (2a)^2 + (2b)^2 + (2c)^2 + (2d)^2$.
- $n = 4k + 1$ or $4k + 3$. Solve $2n = 4(2k) + 2$ or $2n = 8k + 6 = 4(2k + 1) + 2$, say $2n = a^2 + b^2 + c^2 + d^2$, then $a^2 + b^2 + c^2 + d^2 \equiv 2 \pmod 4$, so two of a, b, c, d are odd, two are even. WLOG, say a, b odd, c, d even, then we have $n = \left(\frac{1}{2}(a+b)\right)^2 + \left(\frac{1}{2}(a-b)\right)^2 + \left(\frac{1}{2}(c+d)\right)^2 + \left(\frac{1}{2}(c-d)\right)^2$
- $n = 4k + 2$. Solved using last algorithm.

Solving $x^2 + y^2 + z^2 + w^2 = n$ randomly

Question

Let n be a non-negative integer. Design an algorithm that finds *one* solution of $x^2 + y^2 + z^2 + w^2 = n$ in $\mathbb{Z}$.

Answer

- $n = 4k$. Solve $n/4 = a^2 + b^2 + c^2 + d^2$, then $n = (2a)^2 + (2b)^2 + (2c)^2 + (2d)^2$.
- $n = 4k + 1$ or $4k + 3$. Solve $2n = 4(2k) + 2$ or $2n = 8k + 6 = 4(2k + 1) + 2$, say $2n = a^2 + b^2 + c^2 + d^2$, then $a^2 + b^2 + c^2 + d^2 \equiv 2 \pmod 4$, so two of a, b, c, d are odd, two are even. WLOG, say a, b odd, c, d even, then we have $n = \left(\frac{1}{2}(a+b)\right)^2 + \left(\frac{1}{2}(a-b)\right)^2 + \left(\frac{1}{2}(c+d)\right)^2 + \left(\frac{1}{2}(c-d)\right)^2$
- $n = 4k + 2$. Solved using last algorithm.

Solving $x^2 + y^2 + z^2 + w^2 = n$ randomly

Answer

Complexity: At most $O(\log n)$ arithmetic operations are needed before reaching the form $n = 4k + 2$. Then one run of the last algorithm costs $O((\log n)^2 \log\log n)$. Total $O((\log n)^2 \log\log n)$.

Outline

Dense subsets of $\mathbb{C}^n$

Notation

- $\mathbb{D} := \mathbb{Z}[\frac{1}{2}] = \{\frac{a}{2^k} \mid a \in \mathbb{Z}, k \in \mathbb{N}\}$, the ring of dyadic fractions.
- $\mathbb{D}[i] = \{a + bi \mid a, b \in \mathbb{D}\}$, a ring with no name yet. $\frac{1}{2^k}(a+bi)$
- $S^{n-1} = \{(x_1, x_2, \ldots, x_n) \in \mathbb{C}^n \mid \sum |x_i|^2 = 1\}$, unit sphere in $\mathbb{C}^n$.

Note that $\mathbb{D}[i]^n = \cup_{k\in\mathbb{N}}(1/2^k)\mathbb{Z}[i]^n$.

$$\frac{1}{2^k}\begin{bmatrix} a+bi \\ c+di \\ e+fi \end{bmatrix} \rightsquigarrow \begin{bmatrix} e^{i\theta} \\ 0 \\ 0 \end{bmatrix}$$

Claim

1. $\mathbb{D}[i]$ is dense in $\mathbb{C}$.
2. $\mathbb{D}[i] \cap S^0$ is *not* dense in $\mathbb{C} \cap S^0$.
3. $\mathbb{D}[i]^2 \cap S^1$ is dense in $((\mathbb{C} \oplus 0) \cup \mathbb{D}[i]^2) \cap S^1$, under unproven hypothesis.
4. $\mathbb{D}[i]^3 \cap S^2$ is dense in $((\mathbb{C} \oplus 0 \oplus 0) \cup \mathbb{D}[i]^3) \cap S^2$.

Proof of claim 1, 2, and 4

1. $\mathbb{D}[i]$ is dense in $\mathbb{C}$. Since any $x \in \mathbb{R}$ can be approximated by $\lfloor 2^i x \rfloor / 2^i$, $\mathbb{D}$ is dense in $\mathbb{R}$. Then $\mathbb{D}^n$ is dense in $\mathbb{R}^n$, hence $\mathbb{D}[i]$ $\cong \mathbb{D}^2$ is dense in $\mathbb{C}$. $\cong \mathbb{R}^2$

2. Unit vectors in $\mathbb{D}[i]$ does *not* approximate unit vectors in $\mathbb{C}$. Because for $u = a/2^k + b/2^l i \in \mathbb{D}[i]$ (in reduced form, wlog, $k \geq l$) to be of unit length, $a^2 + b^2 4^{k-l} = 4^k$. If $k > 0$, the right is congruent to 0 modulo 4, but the left 1 or 2. For $k = 0$, we only have finite many such u, but S is infinite.

4. Unit vectors in $\mathbb{D}[i]^3$ approximates unit vectors in $(\mathbb{C} \oplus 0 \oplus 0)$.
$$(1/2^k)\begin{bmatrix} \lfloor 2^k \cos\theta \rfloor + \lfloor 2^k \sin\theta \rfloor i \\ a + bi \\ c + di \end{bmatrix} \text{ approximates } \begin{bmatrix} e^{i\theta} \\ 0 \\ 0 \end{bmatrix},$$
where a, b, c, d satisfy
$a^2 + b^2 + c^2 + d^2 = (2^k)^2 - (\lfloor 2^k \cos\theta \rfloor)^2 - (\lfloor 2^k \sin\theta \rfloor)^2$.

Proof of claim 1, 2, and 4

1. $\mathbb{D}[i]$ is dense in $\mathbb{C}$. Since any $x \in \mathbb{R}$ can be approximated by $\lfloor 2^i x \rfloor / 2^i$, $\mathbb{D}$ is dense in $\mathbb{R}$. Then $\mathbb{D}^n$ is dense in $\mathbb{R}^n$, hence $\mathbb{D}[i]$ is dense in $\mathbb{C}$.

$\frac{1}{2^k}(a+bi)$ $\quad k=0$

2. Unit vectors in $\mathbb{D}[i]$ does *not* approximate unit vectors in $\mathbb{C}$. Because for $u = a/2^k + b/2^l i \in \mathbb{D}[i]$ (in reduced form, wlog, $k \geq l$) to be of unit length, $a^2 + b^2 4^{k-l} = 4^k$. If $k > 0$, the right is congruent to 0 modulo 4, but the left 1 or 2. For $k = 0$, we only have finite many such u ($\pm 1, \pm i$), but S is infinite.

4. Unit vectors in $\mathbb{D}[i]^3$ approximates unit vectors in $(\mathbb{C} \oplus 0 \oplus 0)$.

$$(1/2^k) \begin{bmatrix} \lfloor 2^k \cos\theta \rfloor + \lfloor 2^k \sin\theta \rfloor i \\ a + bi \\ c + di \end{bmatrix} \text{ approximates } \begin{bmatrix} e^{i\theta} \\ 0 \\ 0 \end{bmatrix},$$

where a, b, c, d satisfy
$a^2 + b^2 + c^2 + d^2 = (2^k)^2 - (\lfloor 2^k \cos\theta \rfloor)^2 - (\lfloor 2^k \sin\theta \rfloor)^2.$

Proof of claim 1, 2, and 4

$x^2+y^2+z^2+w^2=n$ $\mathbb{D}$ $\mathbb{Z}$ $\underline{U_4(\mathbb{D})}$

1. $\mathbb{D}[i]$ is dense in $\mathbb{C}$. Since any $x \in \mathbb{R}$ can be approximated by $\lfloor 2^i x \rfloor / 2^i$, $\mathbb{D}$ is dense in $\mathbb{R}$. Then $\mathbb{D}^n$ is dense in $\mathbb{R}^n$, hence $\mathbb{D}[i]$ is dense in $\mathbb{C}$.
2. Unit vectors in $\mathbb{D}[i]$ does *not* approximate unit vectors in $\mathbb{C}$. Because for $u = a/2^k + b/2^l i \in \mathbb{D}[i]$ (in reduced form, wlog, $k \geq l$) to be of unit length, $a^2 + b^2 4^{k-l} = 4^k$. If $k > 0$, the right is congruent to 0 modulo 4, but the left 1 or 2. For $k = 0$, we only have finite many such u, but S is infinite.
4. Unit vectors in $\mathbb{D}[i]^3$ approximates unit vectors in $(\mathbb{C} \oplus 0 \oplus 0)$.

$$(1/2^k) \begin{bmatrix} \lfloor 2^k \cos\theta \rfloor + \lfloor 2^k \sin\theta \rfloor i \\ a + bi \\ c + di \end{bmatrix} \text{ approximates } \begin{bmatrix} e^{i\theta} \\ 0 \\ 0 \end{bmatrix},$$

where a, b, c, d satisfy
$a^2 + b^2 + c^2 + d^2 = (2^k)^2 - (\lfloor 2^k \cos\theta \rfloor)^2 - (\lfloor 2^k \sin\theta \rfloor)^2.$

Unit vector approximation — Proof of claim 3

Question

Let $u = \begin{bmatrix} e^{i\theta} \\ 0 \end{bmatrix} \in \mathbb{C}^2$ be a unit vector. Design an algorithm that finds unit $(1/2^k)\begin{bmatrix} a+bi \\ c+di \end{bmatrix} \in \mathbb{D}[i]^2 \cap S^1$ that approximates u.

Answer

Recall in four-square problem "The idea is that by randomly choosing x and y, we might have $p := n - x^2 - y^2$ is a prime of the form $4k+1$ ...."

But here: we randomly choose x and y in smaller region determined by $e^{i\theta}$. And we need a strong hypothesis about the abundance of such x and y.

Unit vector approximation — Proof of claim 3

Question

Let $u = \begin{bmatrix} e^{i\theta} \\ 0 \end{bmatrix} \in \mathbb{C}^2$ be a unit vector. Design an algorithm that finds unit $(1/2^k)\begin{bmatrix} a+bi \\ c+di \end{bmatrix} \in \mathbb{D}[i]^2 \cap S^1$ that approximates u.

Answer

Recall in four-square problem "The idea is that by randomly choosing x and y, we might have $p := n - x^2 - y^2$ is a prime of the form $4k+1$ ...."

But here: we randomly choose x and y in smaller region determined by $e^{i\theta}$. And we need a strong hypothesis about the abundance of such x and y.

Unit vector approximation — Proof of claim 3

Question

Let $u = \begin{bmatrix} e^{i\theta} \\ 0 \end{bmatrix} \in \mathbb{C}^2$ be a unit vector. Design an algorithm that finds unit $(1/2^k) \begin{bmatrix} a+bi \\ c+di \end{bmatrix} \in \mathbb{D}[i]^2 \cap S^1$ that approximates u.

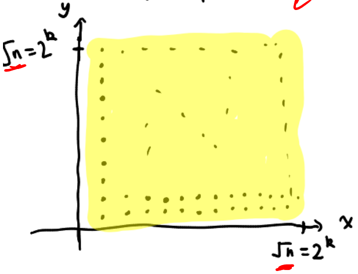
Answer

Recall in four-square problem "The idea is that by randomly choosing x and y, we might have $p := n - x^2 - y^2$ is a prime of the form $4k+1$ ...."

But here: we randomly choose x and y in smaller region determined by $e^{i\theta}$. And we need a strong hypothesis about the abundance of such x and y.

Smaller region

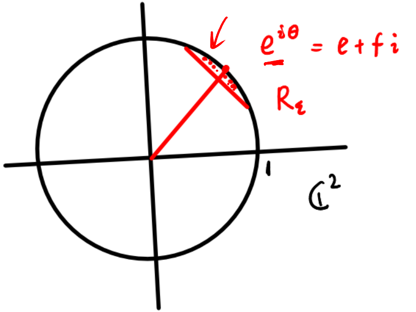
Four-square problem



Let $n=4^k$

Pool : all the integer pts in this region.

unit vector approximation

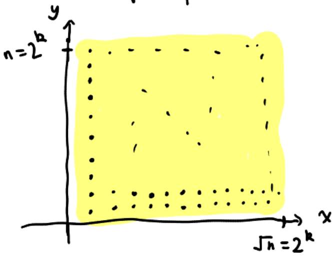


Pool : all the "Dyadic fraction" pts in this region. e.g. For $a+bi \in R_\varepsilon$

$$\left\| \begin{pmatrix} a+bi \\ 0 \end{pmatrix} - \begin{pmatrix} e+fi \\ 0 \end{pmatrix} \right\| < \varepsilon$$

Smaller region

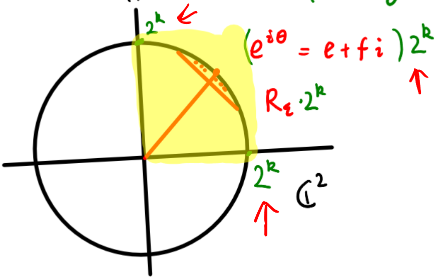
Four-square problem



Let $n=4^k$

Pool : all the integer pts in this region. i.e. this square

unit vector approximation — equivalently



Equivalently : $\frac{1}{2^k}(a+bi) \in R_\varepsilon$, $a,b \in \mathbb{Z}$

$\Longleftrightarrow$ $a+bi \in 2^k R_\varepsilon$.

Pool : all the interger pts in the crescent

Bigger hypothesis

Note

$2^k R_\epsilon$ contains many points for sufficient large k.

Recall the proven density of good x, y's in the square

$$\frac{A}{(\log n) \log\log n}$$

Bigger hypothesis

Good points x, y are even distributed, i.e. the density of good x, y's in the crescent is same as the density in the square.

Consequences

For large k, we can find x, y such that $4^k - x^2 - y^2$ is a prime of the form $4j + 1$. That is the four-square problem with extra constraint can be efficiently solved.

Bigger hypothesis

Note

$2^k R_\epsilon$ contains many points for sufficient large k.

Recall the proven density of good x, y's in the square

$\frac{A}{(\log n)\log\log n}$

Bigger hypothesis

Good points x, y are even distributed, i.e. the density of good x, y's in the crescent is same as the density in the square.

Consequences

For large k, we can find x, y such that $4^k - x^2 - y^2$ is a prime of the form $4j + 1$. That is the four-square problem with extra constraint can be efficiently solved.

Bigger hypothesis

Note

$2^k R_\epsilon$ contains many points for sufficient large k.

Recall the proven density of good x, y's in the square

$\frac{A}{(\log n) \log \log n}$

Bigger hypothesis

Good points x, y are even distributed, i.e. the density of good x, y's in the crescent is same as the density in the square.

Consequences

For large k, we can find x, y such that $4^k - x^2 - y^2$ is a prime of the form $4j + 1$. That is the four-square problem with extra constraint can be efficiently solved.

Bigger hypothesis

$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$

$\frac{1}{2}\begin{bmatrix} 1-i & 1-i \\ 1-i & -1+i \end{bmatrix}$

$\begin{bmatrix} 1 & i \end{bmatrix}$

Note

$2^k R_\epsilon$ contains many points for sufficient large k.

Recall the proven density of good x, y's in the square

$$\frac{A}{(\log n) \log \log n}$$

Bigger hypothesis

Good points x, y are even distributed, i.e. the density of good x, y's in the crescent is same as the density in the square.

Consequences

For large k, we can find x, y such that $4^k - x^2 - y^2$ is a prime of the form $4j + 1$. That is the four-square problem with extra constraint can be efficiently solved.

Unit vector approximation

Question

Find a unit vector $(1/2^k)\begin{bmatrix} a+bi \\ c+di \end{bmatrix}$ that approximates $\begin{bmatrix} e^{i\theta} \\ 0 \end{bmatrix}$.

Answer

Algorithm:

1. For given ϵ, calculate k. (not hard, not expensive, omitted)
2. Solve restricted four-square problem $4^k = x^2 + y^2 + y^2 + z^2$ with $(x, y) \in 2^k R_\epsilon$ get solution a, b, c, d.
3. return $(1/2^k)\begin{bmatrix} a+bi \\ c+di \end{bmatrix}$, finish.

Complexity: depends on the area of R_ϵ. We omit the calculation and only state the result. Polynomial in $\log(1/\epsilon)$.

Note

c, d are relatively small compared to 2^k, it does't affect much.

Unit vector approximation

Question

Find a unit vector $(1/2^k)\begin{bmatrix} a+bi \\ c+di \end{bmatrix}$ that approximates $\begin{bmatrix} e^{i\theta} \\ 0 \end{bmatrix}$.

Answer

Algorithm:

1. For given ϵ, calculate k. (not hard, not expensive, omitted)
2. Solve restricted four-square problem $4^k = x^2 + y^2 + y^2 + z^2$ with $(x, y) \in 2^k R_\epsilon$ get solution a, b, c, d.
3. return $(1/2^k)\begin{bmatrix} a+bi \\ c+di \end{bmatrix}$, finish.

$\frac{1}{\epsilon} \sim n = 4^k$

$\Rightarrow k \sim \log \frac{1}{\epsilon}$

Complexity: depends on the area of R_ϵ. We omit the calculation and only state the result. Polynomial in $\log(1/\epsilon)$.

$O((\log n)^2 \log\log n)$

Note

c, d are relatively small compared to 2^k, it does't affect much.

Unit vector approximation

Question

Find a unit vector $(1/2^k)\begin{bmatrix} a+bi \\ c+di \end{bmatrix}$ that approximates $\begin{bmatrix} e^{i\theta} \\ 0 \end{bmatrix}$.

Answer

Algorithm:

1. For given ϵ, calculate k. (not hard, not expensive, omitted)
2. Solve restricted four-square problem $4^k = x^2 + y^2 + y^2 + z^2$ with $(x, y) \in 2^k R_\epsilon$ get solution a, b, c, d.
3. return $(1/2^k)\begin{bmatrix} a+bi \\ c+di \end{bmatrix}$, finish.

Complexity: depends on the area of R_ϵ. We omit the calculation and only state the result. Polynomial in $\log(1/\epsilon)$.

Note

c, d are relatively small compared to 2^k, it does't affect much.

z-rotation approximation

z - rotation

Question

Find unitary $(1/2^k)\begin{bmatrix} a+bi & e+fi \\ c+di & g+hi \end{bmatrix}$ that is close to $\begin{bmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{bmatrix}$, where $a, b, c, d, e, f, g, h, k$ are integers.

Answer

Basically, we first approximate $\begin{bmatrix} e^{i\theta} \\ 0 \end{bmatrix}$, then through some algebraic manipulation, we get a solution. We claim, if $(1/2^k)\begin{bmatrix} a+bi \\ c+di \end{bmatrix}$ approximates $\begin{bmatrix} e^{i\theta} \\ 0 \end{bmatrix}$, then $(1/2^k)\begin{bmatrix} a+bi & -c+di \\ c+di & a-bi \end{bmatrix}$ is unitary and close to $\begin{bmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{bmatrix}$.

z-rotation approximation

Question

Find unitary $(1/2^k)\begin{bmatrix} a+bi & e+fi \\ c+di & g+hi \end{bmatrix}$ that is close to $\begin{bmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{bmatrix}$, where $a, b, c, d, e, f, g, h, k$ are integers.

Answer

Basically, we first approximate $\begin{bmatrix} e^{i\theta} \\ 0 \end{bmatrix}$, then through some algebraic manipulation, we get a solution. We claim, if $(1/2^k)\begin{bmatrix} a+bi \\ c+di \end{bmatrix}$ approximates $\begin{bmatrix} e^{i\theta} \\ 0 \end{bmatrix}$, then $(1/2^k)\begin{bmatrix} a+bi & -c+di \\ c+di & a-bi \end{bmatrix}$ is unitary and close to $\begin{bmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{bmatrix}$.

Unitary approximation

Question

Find unitary $(1/2^k)\begin{bmatrix} a+bi & e+fi \\ c+di & g+hi \end{bmatrix}$ that approximates $\begin{bmatrix} p & q \\ r & s \end{bmatrix}$ an arbitary unitary, where $a, b, c, d, e, f, g, h, k$ are integers.

Answer

An x-rotation is of the form $K\begin{bmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{bmatrix}K^\dagger$, where $K = \frac{1}{2}\begin{bmatrix} 1-i & 1-i \\ 1-i & -1+i \end{bmatrix}$, and $\bullet^\dagger$ means taking conjugate transpose.

Any 2x2 unitary can written as a product ABC, where A, C are z-rotations and B is an x-rotation.

Unitary approximation

Question

Find unitary $(1/2^k)\begin{bmatrix} a+bi & e+fi \\ c+di & g+hi \end{bmatrix}$ that approximates $\begin{bmatrix} p & q \\ r & s \end{bmatrix}$ an arbitary unitary, where $a, b, c, d, e, f, g, h, k$ are integers.

Answer

An x-rotation is of the form $K\begin{bmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{bmatrix}K^\dagger$, where

$K = \frac{1}{2}\begin{bmatrix} 1-i & 1-i \\ 1-i & -1+i \end{bmatrix}$, and $\bullet^\dagger$ means taking conjugate transpose.

Any 2x2 unitary can written as a product ABC, where A, C are z-rotations and B is an x-rotation. , up to global phase

References

$O\left((\log n)^2 / \log\log n\right)$

Paul Pollack and Enrique Trevino.
Finding the four squares in lagrange's theorem.
Integers, 18:A15, 2018.

Michael O. Rabin and Jeffery O. Shallit.
Randomized algorithms in number theory.
Communications on Pure and Applied Mathematics, 39(S1):S239–S256, 1986.

Neil J. Ross and Peter Selinger.
Optimal ancilla-free Clifford+T approximation of z-rotations.
Quantum Information and Computation, 16(11–12):901–953, 2016.

$\mathbb{Z}[\omega], \mathbb{D}[\omega]$ $\quad \omega^8 = 1$

$\mathbb{Z}[i], \mathbb{D}[i]$ $\quad i^4 = 1$

Peter Selinger.
Efficient Clifford+T approximation of single-qubit operators.
Quantum Information and Computation, 15(1–2):159–180, 2015.