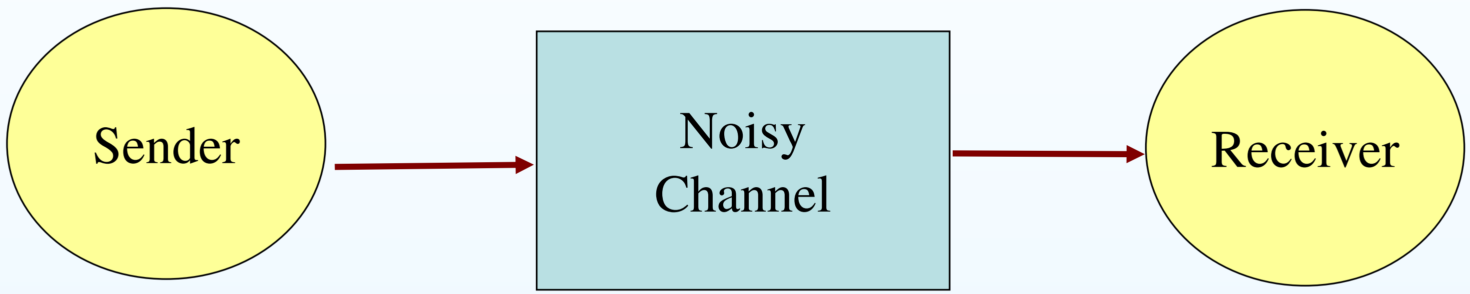
List Decoding of Reed Solomon Codes

Madhu Sudan

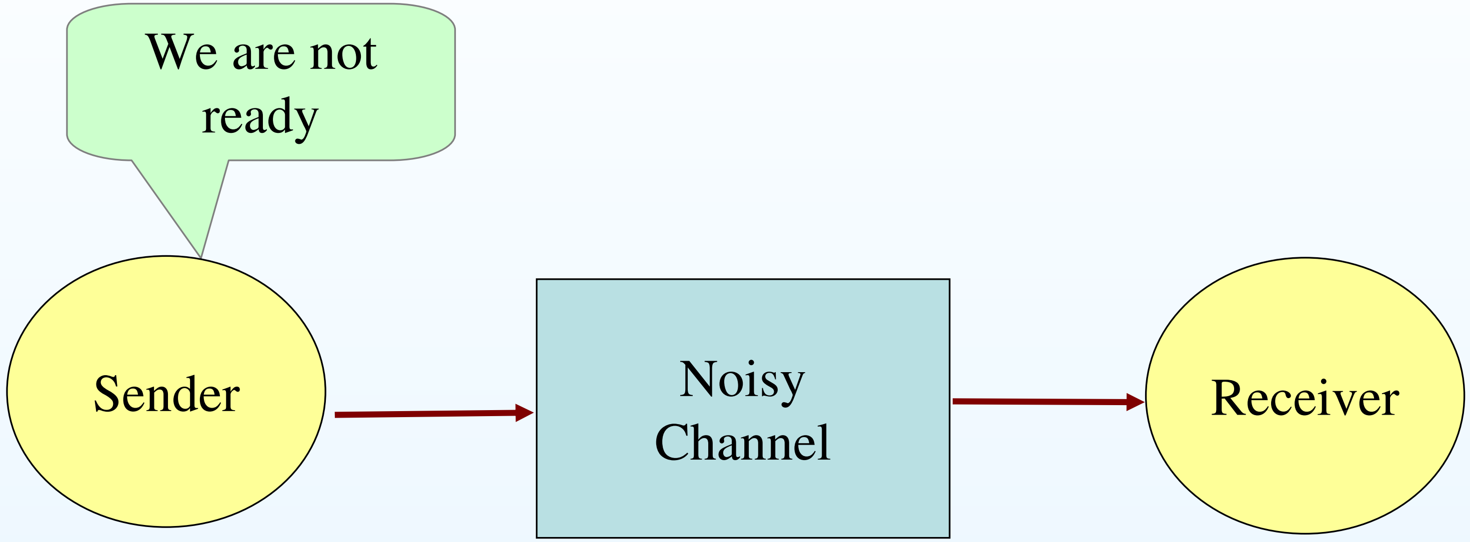
MIT CSAIL

Background: Reliable Transmission of Information

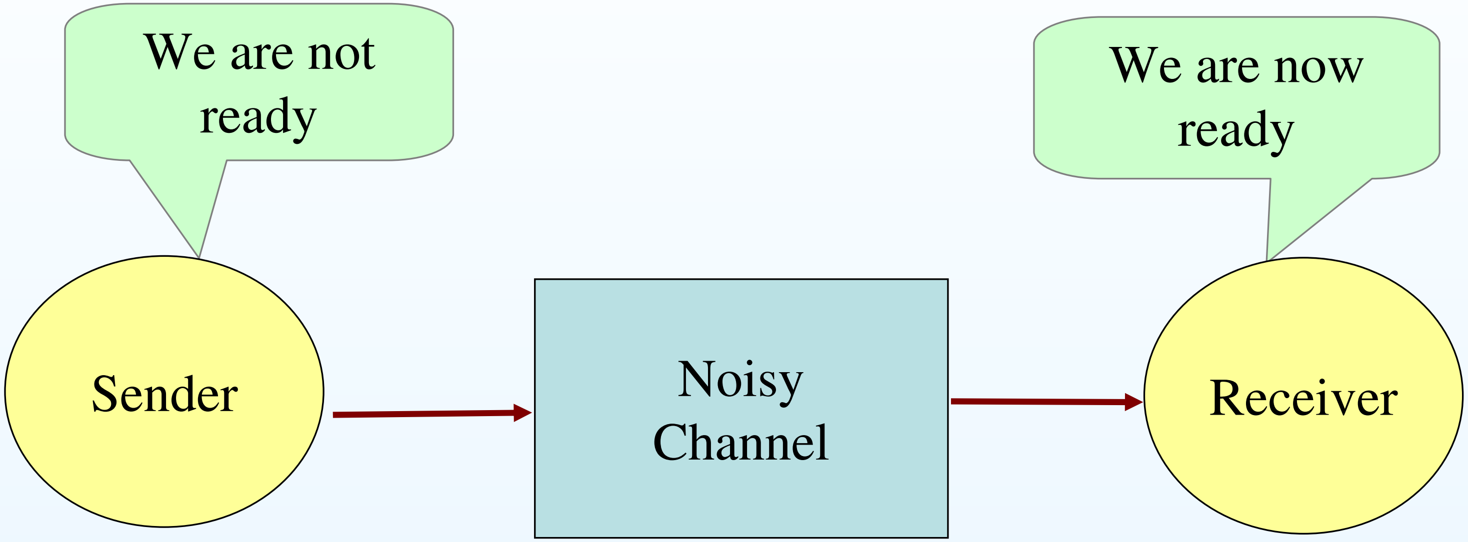
The Problem of Information Transmission



The Problem of Information Transmission



The Problem of Information Transmission



- When information is <u>digital</u>, reliability is critical.
- Need to understand <u>errors</u>, and correct them.

Shannon (1948)

- Model noise by probability distribution.
- Example: Binary symmetric channel (BSC)
  - Parameter $p \in [0, \frac{1}{2}]$.
  - Channel transmits bits.
  - With probability $1 - p$ bit transmitted faithfully, and with probability p bit flipped (independent of all other events).

Shannon's architecture

- Sender encodes k bits into n bits.
- Transmits n bit string on channel.
- Receiver decodes n bits into k bits.
- Rate of channel usage = k/n.

Shannon's theorem

- Every channel (in broad class) has a capacity s.t., transmitting at Rate below capacity is feasible and above capacity is infeasible.
- Example: Binary symmetric channel (p) has capacity $1 - H(p)$, where $H(p)$ is the binary entropy function.
  - $p = 0$ implies capacity $= 1$.
  - $p = \frac{1}{2}$ implies capacity $= 0$.
  - $p < \frac{1}{2}$ implies capacity > 0.
- Example: q-ary symmetric channel (p): On input $\sigma \in \mathbb{F}_q$ receiver receives (independently) σ', where
  - $\sigma' = \sigma$ w.p. $1 - p$.
  - σ' uniform over $\mathbb{F}_q - \{\sigma\}$ w.p. p.

  Capacity positive if $p < 1 - 1/q$.

Constructive versions

- Shannon's theory was non-constructive. Decoding takes exponential time.
- [Elias '55] gave polytime algorithms to achieve positive rate on every channel of positive capacity.
- [Forney '66] achieved any rate $<$ capacity with polynomial time algorithms (and exponentially small error).
- Modern results (following [Spielman '96]) lead to linear time algorithms.

Hamming (1950)

- Modelled errors adversarially.
- Focussed on image of encoding function (the “Code”).
- Introduced metric (Hamming distance) on range of encoding function. $d(x, y) = \#$ coordinates such that $x_i \neq y_i$.
- Noticed that for adversarial error (and guaranteed error recovery), distance of Code is important.

$$\Delta(C) = \min_{x,y \in C} \{d(x, y)\}.$$

- Code of distance d corrects $(d-1)/2$ errors.

Contrast between Shannon & Hamming

Contrast between Shannon & Hamming

[Sha48] : $\boxed{\mathcal{C}}$ probabilistic.

E.g., flips each bit independently w.p. p.

✔ Tightly analyzed for many cases e.g., q-SC(p).

✗ Channel may be too weak to capture some scenarios.

✗ Need very accurate channel model.

Contrast between Shannon & Hamming

[Sha48] : $\boxed{\mathcal{C}}$ probabilistic.

✔ Corrects many errors. ✗ Channel restricted.

Contrast between Shannon & Hamming

[Sha48] : $\boxed{\mathcal{C}}$ probabilistic.

✔ Corrects many errors. ✗ Channel restricted.

[Ham50] : $\boxed{\mathcal{C}}$ flips bits *adversarially*

✔ Safer model, “good” codes known

✗ Too pessimistic: Can only decode if $p < 1/2$ for any alphabet.

Contrast between Shannon & Hamming

[Sha48] : $\boxed{\mathcal{C}}$ probabilistic.

✔ Corrects many errors. ✗ Channel restricted.

[Ham50] : $\boxed{\mathcal{C}}$ flips bits *adversarially*

✗ Fewer errors. ✔ More general errors.

Contrast between Shannon & Hamming

[Sha48] : $\boxed{\mathcal{C}}$ probabilistic.

✔ Corrects many errors. ✗ Channel restricted.

[Ham50] : $\boxed{\mathcal{C}}$ flips bits *adversarially*

✗ Fewer errors. ✔ More general errors.

- Which model is correct? Depends on application.
  - Crudely: Small $q \Rightarrow$ Shannon. Large $q \Rightarrow$ Hamming.
- Today: New Models of error-correction + algorithms.
  - List-decoding: Relaxed notion of decoding.

Contrast between Shannon & Hamming

[Sha48] : $\boxed{\mathcal{C}}$ probabilistic.

✔ Corrects many errors. ✗ Channel restricted.

[Ham50] : $\boxed{\mathcal{C}}$ flips bits *adversarially*

✗ Fewer errors. ✔ More general errors.

- Which model is correct? Depends on application.
  - Crudely: Small $q \Rightarrow$ Shannon. Large $q \Rightarrow$ Hamming.
- Today: New Models of error-correction + algorithms.
  - List-decoding: Relaxed notion of decoding.

    ✔ More errors ✔ Strong (enough) errors.

Reed-Solomon Codes

Motivation: [Singleton] Bound

- Suppose $C \subseteq \mathbb{F}_q^n$ has q^k codewords. How large can its distance be?

Motivation: [Singleton] Bound

- Suppose $C \subseteq \mathbb{F}_q^n$ has q^k codewords. How large can its distance be?
- Bound: $\Delta(C) \leq n - k + 1$.

Motivation: [Singleton] Bound

- Suppose $C \subseteq \mathbb{F}_q^n$ has q^k codewords. How large can its distance be?
- Bound: $\Delta(C) \leq n - k + 1$.
- Proof:
  - Project code to first $k-1$ coordinates.
  - By Pigeonhole Principle, two codewords collide.
  - These two codewords thus disagree in at most $n-k+1$ coordinates.

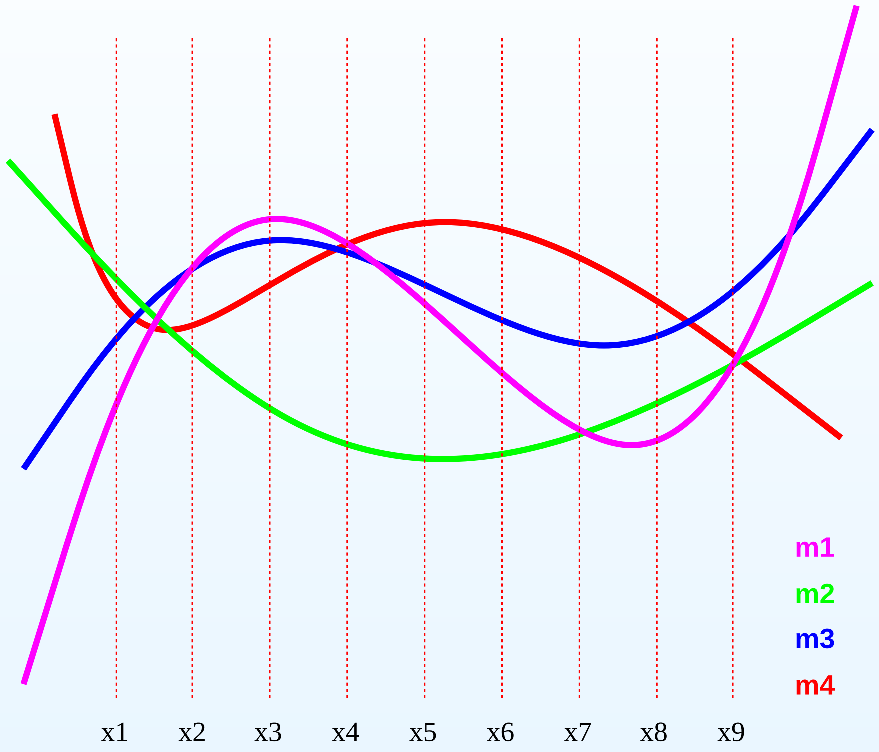
Motivation: [Singleton] Bound

- Suppose $C \subseteq \mathbb{F}_q^n$ has q^k codewords. How large can its distance be?
- Bound: $\Delta(C) \leq n - k + 1$.
- Proof:
  - Project code to first $k - 1$ coordinates.
  - By Pigeonhole Principle, two codewords collide.
  - These two codewords thus disagree in at most $n - k + 1$ coordinates.
- Surely we can do better?

Motivation: [Singleton] Bound

- Suppose $C \subseteq \mathbb{F}_q^n$ has q^k codewords. How large can its distance be?
- Bound: $\Delta(C) \leq n - k + 1$.
- Proof:
  - Project code to first $k - 1$ coordinates.
  - By Pigeonhole Principle, two codewords collide.
  - These two codewords thus disagree in at most $n - k + 1$ coordinates.
- Surely we can do better?
- Actually - No! [Reed-Solomon] Codes match this bound!

Reed-Solomon Codes



- Messages $\equiv$ Polynomial.
- Encoding $\equiv$ Evaluation at $x_1, \ldots, x_n$.
- $n >$ Degree: Injective
- $n \gg$ Degree: Redundant

Reed-Solomon Codes (formally)

- Let $\mathbb{F}_q$ be a finite field.
- Code specified by $k, n, \alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$.
- Message: $\langle c_0, \ldots, c_k \rangle \in \mathbb{F}_q^{k+1}$ coefficients of degree k polynomial $p(x) = c_0 + c_1 x + \cdots c_k x^k$.
- Encoding: $p \mapsto \langle p(\alpha_1), \ldots, p(\alpha_n) \rangle$. ($k + 1$ letters to n letters.)
- Degree k poly has at most k roots $\Leftrightarrow$ Distance $d = n - k$.
- These are the Reed-Solomon codes.
  Match [Singleton] bound!
  Commonly used (CDs, DVDs etc.).

List-Decoding of Reed-Solomon Codes

Reed-Solomon Decoding

Restatement of the problem:

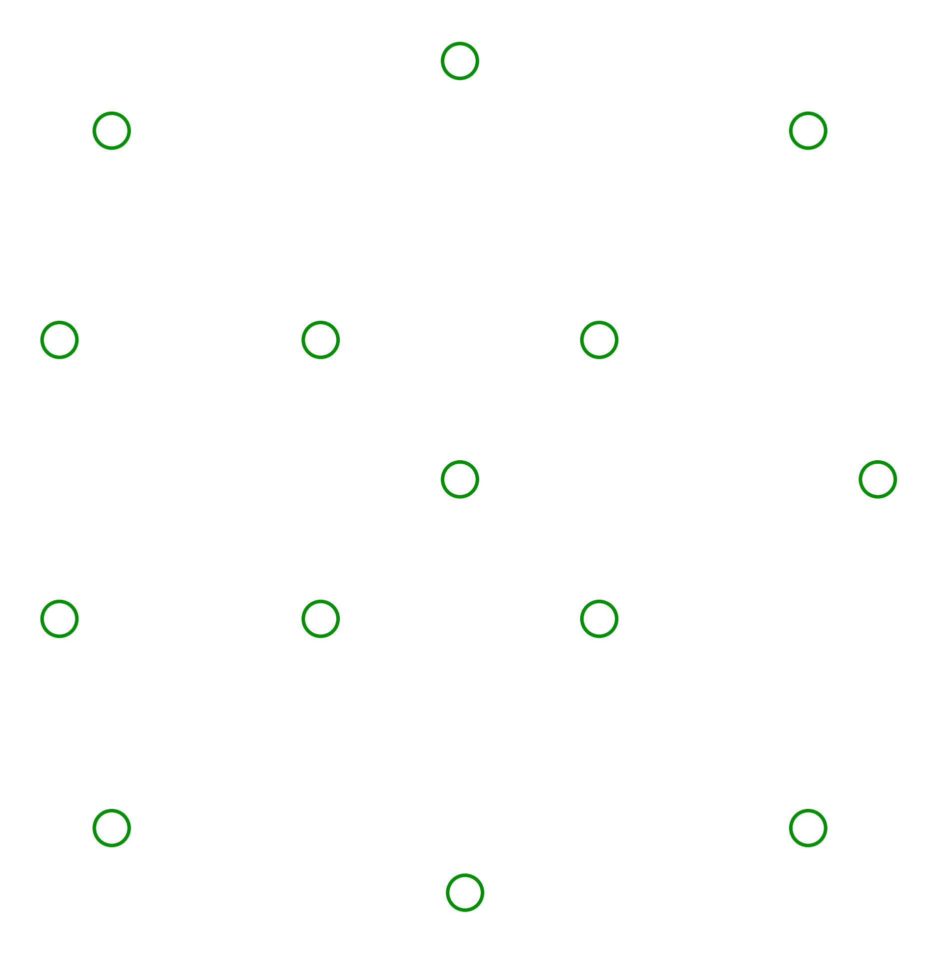
- Input: n points $(\alpha_i, y_i) \in \mathbb{F}_q^2$; agreement parameter t
- Output: **All** degree k polynomials $p(x)$ s.t. $p(\alpha_i) = y_i$ for at least t values of i.

We use $k = 1$ for illustration.

- i.e. want *all* "lines" $(y - ax - b = 0)$ that pass through $\geq t$ out of n points.

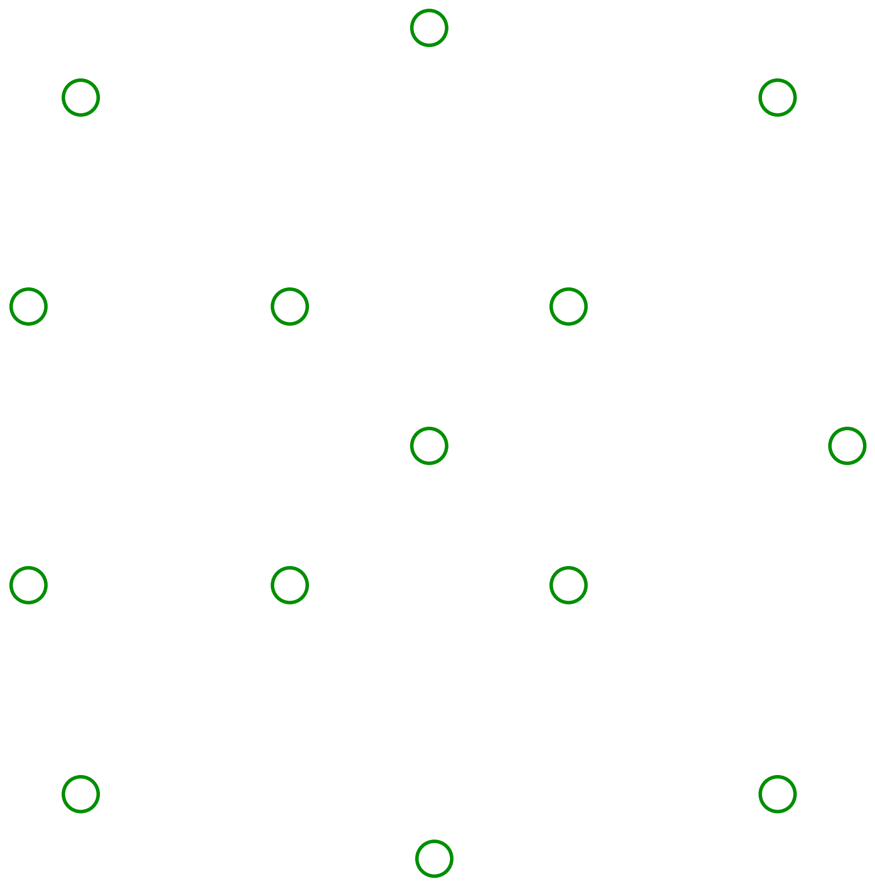
Algorithm Description [S. '96]

$n = \mathbf{14}$ points; Want all *lines* through at least **5** points.



Algorithm Description [S. '96]

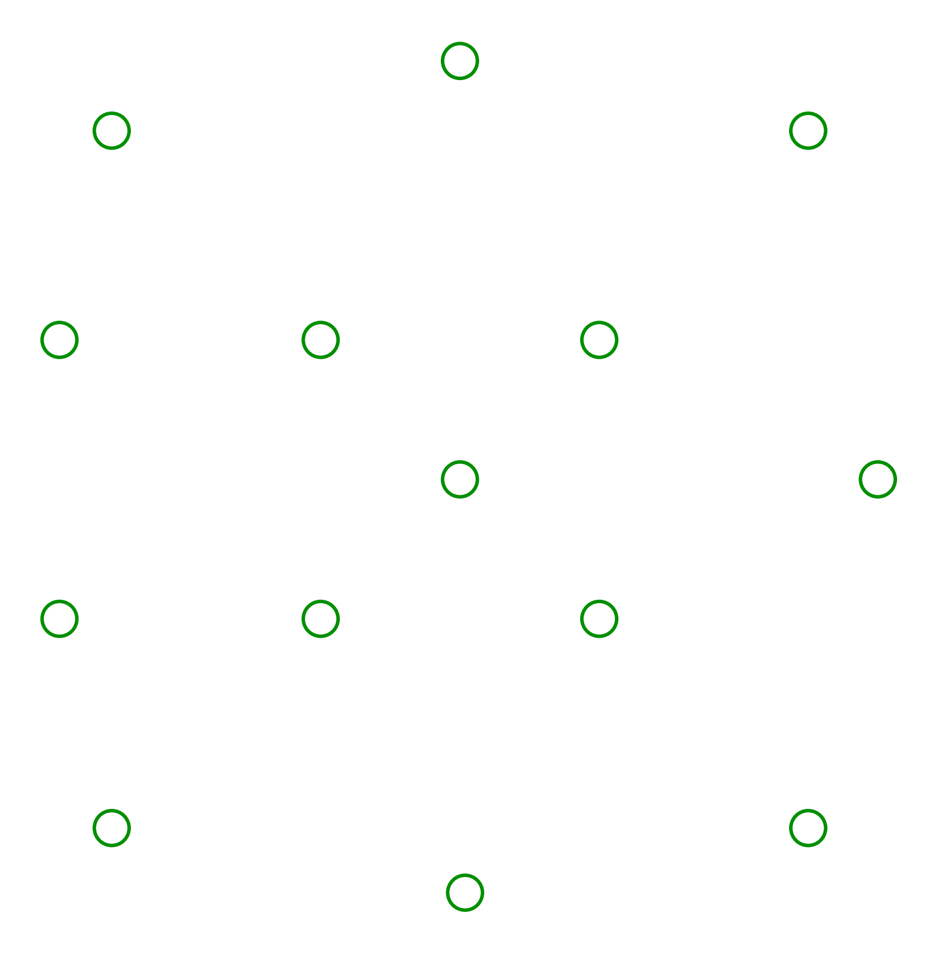
$n = \mathbf{14}$ points; Want all *lines* through at least **5** points.



Find deg. 4 poly. $Q(x, y) \neq 0$
s.t. $Q(\alpha_i, y_i) = 0$ for all points.

Algorithm Description [S. '96]

$n = \mathbf{14}$ points; Want all *lines* through at least **5** points.



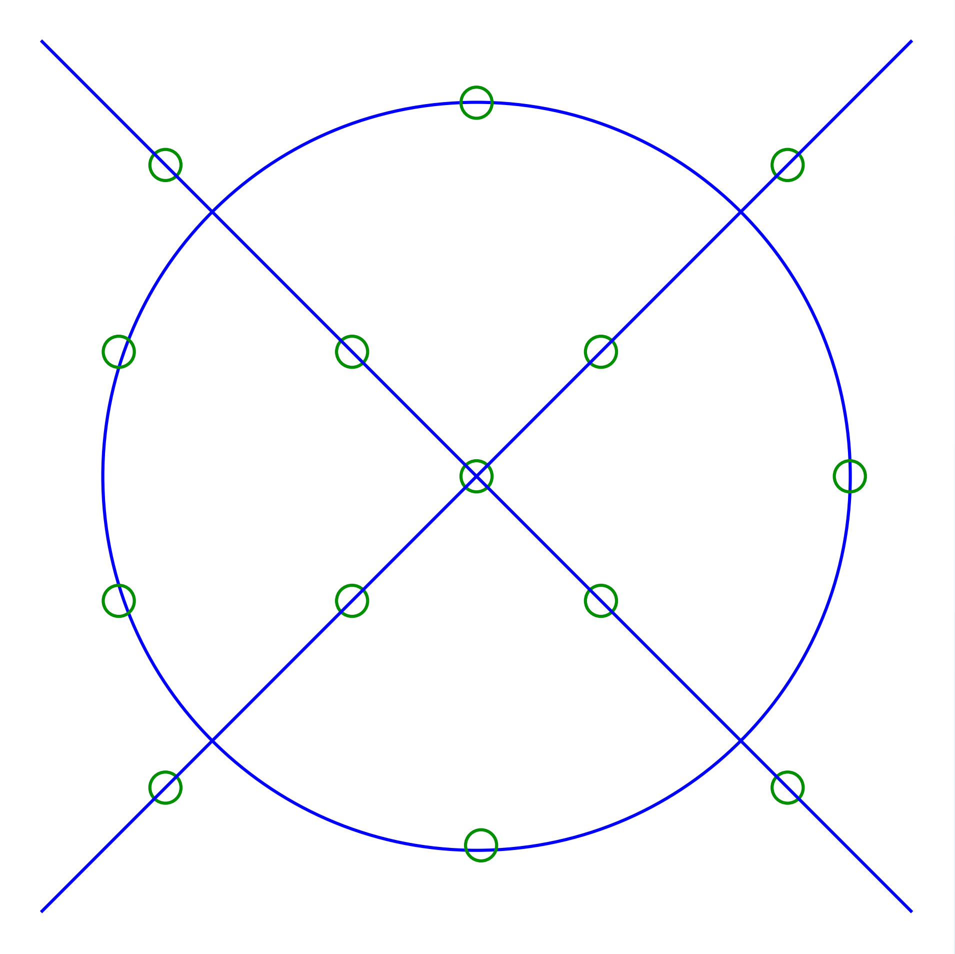
Find deg. 4 poly. $Q(x, y) \neq 0$
s.t. $Q(\alpha_i, y_i) = 0$ for all points.

$$Q(x, y) = y^4 - x^4 - y^2 + x^2$$

Let us plot all zeroes of Q ...

Algorithm Description [S. '96]

$n = \mathbf{14}$ points; Want all *lines* through at least **5** points.



Find deg. 4 poly. $Q(x, y) \not\equiv 0$
s.t. $Q(\alpha_i, y_i) = 0$ for all points.

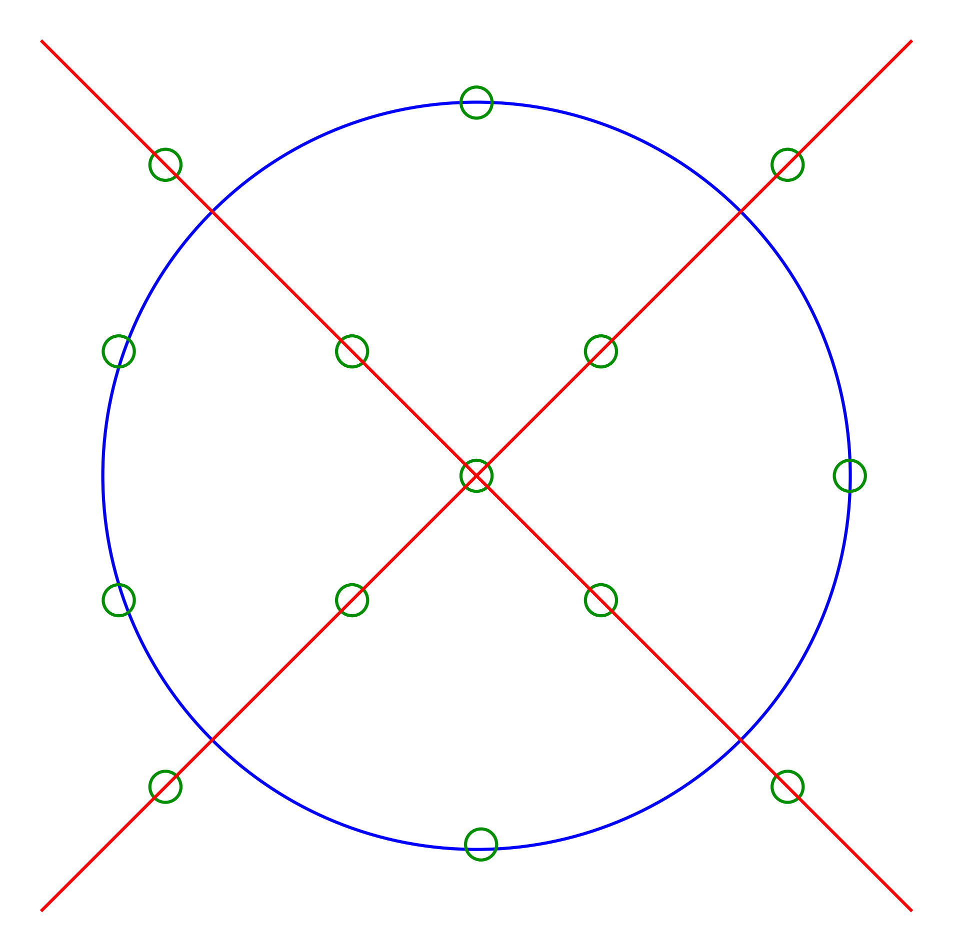
$Q(x, y) = y^4 - x^4 - y^2 + x^2$

Let us plot all zeroes of Q ...

Both relevant lines emerge !

Algorithm Description [S. '96]

$n = \mathbf{14}$ points; Want all *lines* through at least **5** points.



Find deg. 4 poly. $Q(x, y) \not\equiv 0$
s.t. $Q(\alpha_i, y_i) = 0$ for all points.

$Q(x, y) = y^4 - x^4 - y^2 + x^2$

Let us plot all zeroes of Q ...

Both relevant lines emerge !

Formally, $Q(x, y)$ factors as:
$(x^2 + y^2 - 1)(y + x)(y - x)$.

What Happened?

1. Why did degree 4 curve exist?
   - Counting argument: degree 4 gives enough degrees of freedom to pass through any 14 points.
2. Why did all the relevant lines emerge/factor out?
   - Line ℓ intersects a deg. 4 curve Q in 5 points $\Longrightarrow$ ℓ is a factor of Q

Generally

Lemma 1: $\exists Q$ with $\deg_x(Q), \deg_y(Q) \leq D = \sqrt{n}$ passing thru any n points.

Lemma 2: If Q with $\deg_x(Q), \deg_y(Q) \leq D$ intersects $y - p(x)$ with $\deg(p) \leq d$ intersect in more that $(D+1)d$ points, then $y - p(x)$ divides Q.

Efficient algorithm?

1. Can find Q by solving system of linear equations

Efficient algorithm?

1. Can find Q by solving system of linear equations
2. Fast algorithms for factorization of bivariate polynomials exist ('83-'85) [Kaltofen, Chistov & Grigoriev, Lenstra, von zur Gathen & Kaltofen]

Efficient algorithm?

1. Can find Q by solving system of linear equations
2. Fast algorithms for factorization of bivariate polynomials exist ('83-'85) [Kaltofen, Chistov & Grigoriev, Lenstra, von zur Gathen & Kaltofen]

- Immediate application:

Efficient algorithm?

1. Can find Q by solving system of linear equations
2. Fast algorithms for factorization of bivariate polynomials exist ('83-'85) [Kaltofen, Chistov & Grigoriev, Lenstra, von zur Gathen & Kaltofen]

- Immediate application:

  Theorem: Can list-decode Reed-Solomon code from $n-(k+1)\sqrt{n}$ errors.

Efficient algorithm?

1. Can find Q by solving system of linear equations
2. Fast algorithms for factorization of bivariate polynomials exist ('83-'85) [Kaltofen, Chistov & Grigoriev, Lenstra, von zur Gathen & Kaltofen]

- Immediate application:

  Theorem: Can list-decode Reed-Solomon code from $n - (k+1)\sqrt{n}$ errors.

- With some fine-tuning of parameters:

Efficient algorithm?

1. Can find Q by solving system of linear equations
2. Fast algorithms for factorization of bivariate polynomials exist ('83-'85) [Kaltofen, Chistov & Grigoriev, Lenstra, von zur Gathen & Kaltofen]

- Immediate application:

  Theorem: Can list-decode Reed-Solomon code from $n - (k+1)\sqrt{n}$ errors.

- With some fine-tuning of parameters:

  Theorem: [S. '96] Can list-decode Reed-Solomon code from $1 - \sqrt{2R}$-fraction errors.

Efficient algorithm?

1. Can find Q by solving system of linear equations
2. Fast algorithms for factorization of bivariate polynomials exist ('83-'85) [Kaltofen, Chistov & Grigoriev, Lenstra, von zur Gathen & Kaltofen]

- Immediate application:

  Theorem: Can list-decode Reed-Solomon code from $n - (k+1)\sqrt{n}$ errors.

- With some fine-tuning of parameters:

  Theorem: [S. '96] Can list-decode Reed-Solomon code from $1 - \sqrt{2R}$-fraction errors.

- Does not meet combinatorial bounds though!

Efficient algorithm?

1. Can find Q by solving system of linear equations
2. Fast algorithms for factorization of bivariate polynomials exist ('83-'85) [Kaltofen, Chistov & Grigoriev, Lenstra, von zur Gathen & Kaltofen]

- Immediate application:

  Theorem: Can list-decode Reed-Solomon code from $n - (k+1)\sqrt{n}$ errors.

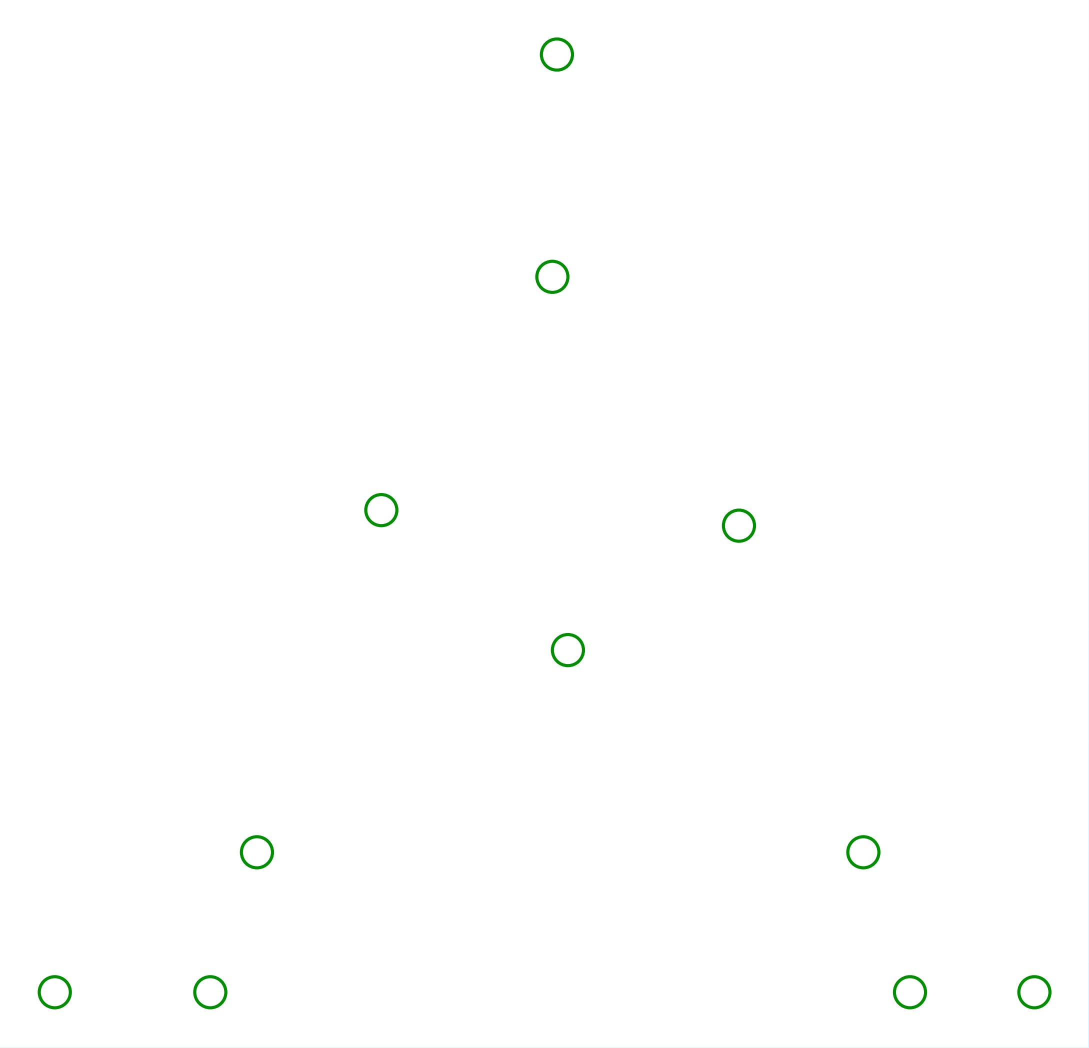
- With some fine-tuning of parameters:

  Theorem: [S. '96] Can list-decode Reed-Solomon code from $1 - \sqrt{2R}$-fraction errors.

- Does not meet combinatorial bounds though!

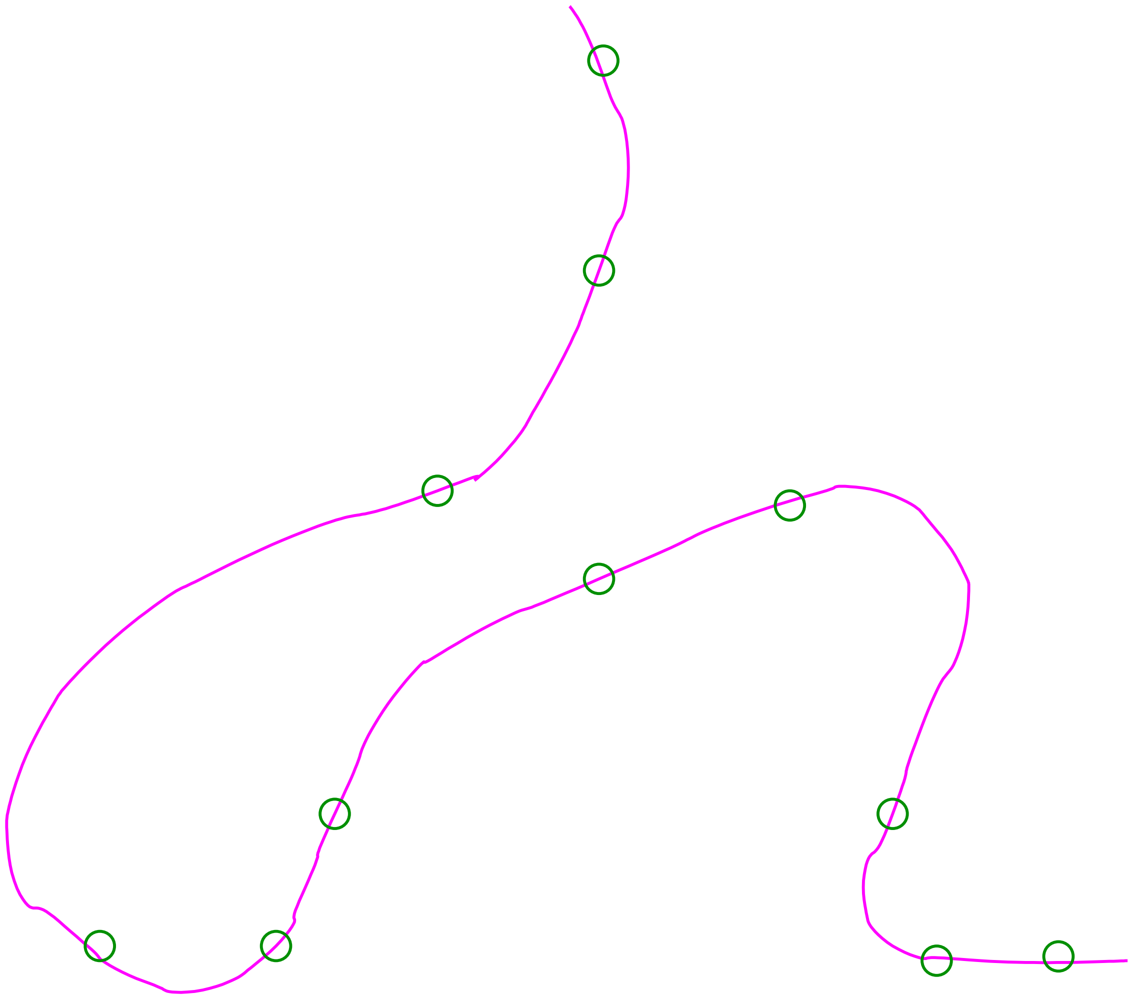
Improved List-Decoding

Going Further: Example 2 [Guruswami+S. '98]



$n = 11$ points; Want all lines through ≥ 4 pts.

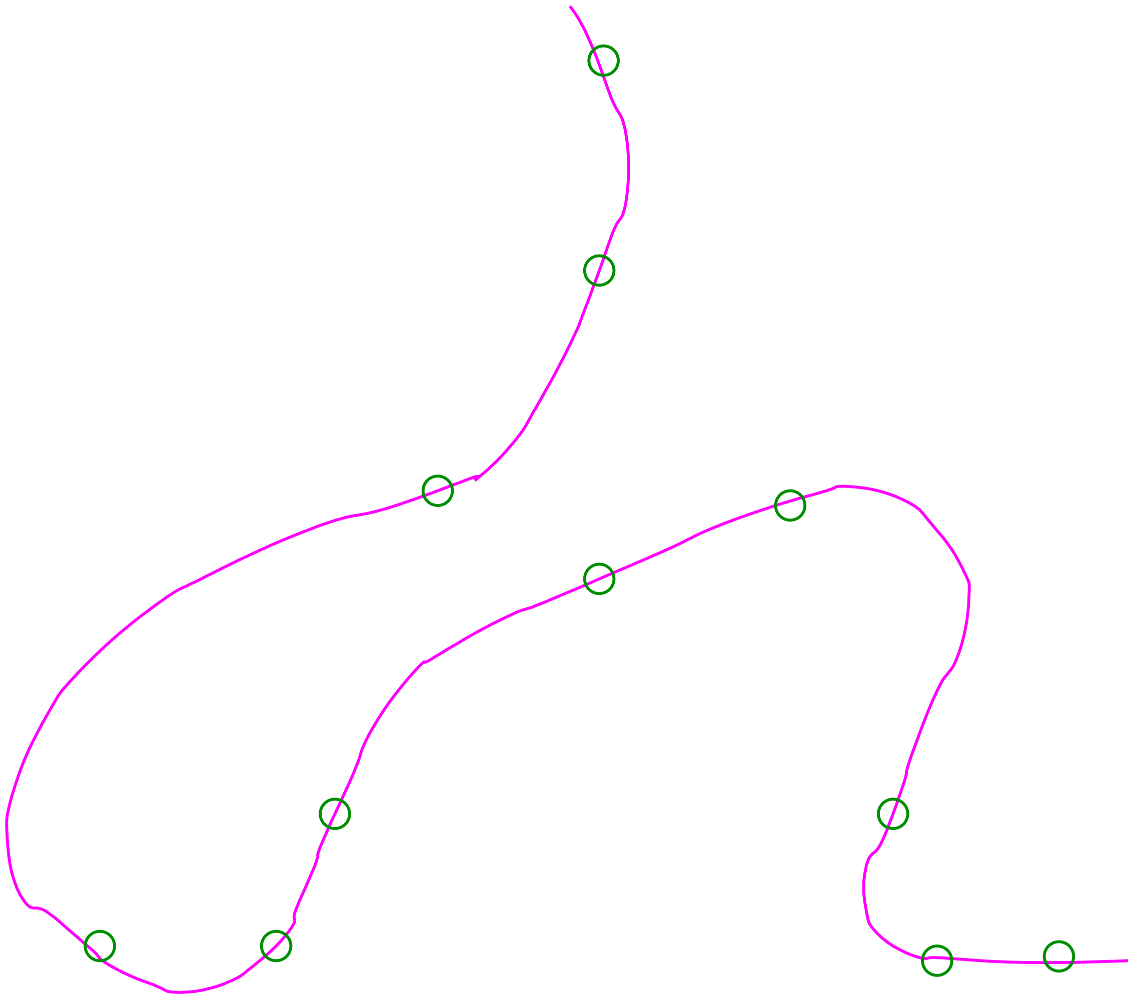
Going Further: Example 2 [Guruswami+S. '98]



$n = 11$ points; Want <u>all</u> lines through ≥ 4 pts.

Fitting degree 4 curve Q as earlier doesn't work.

Going Further: Example 2 [Guruswami+S. '98]

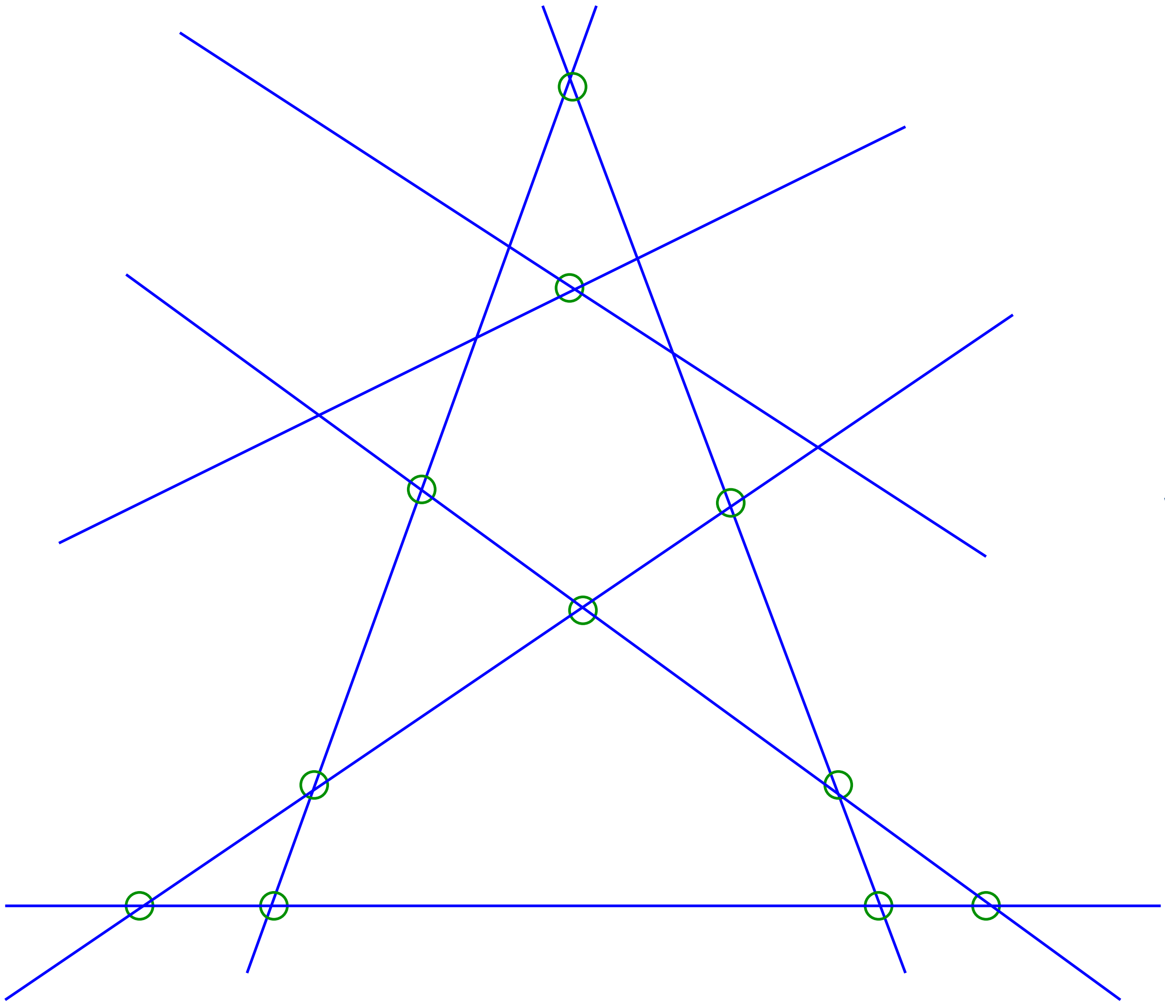


$n = 11$ points; Want all lines through ≥ 4 pts.

Fitting degree 4 curve Q as earlier doesn't work.

Why?

Going Further: Example 2 [Guruswami+S. '98]



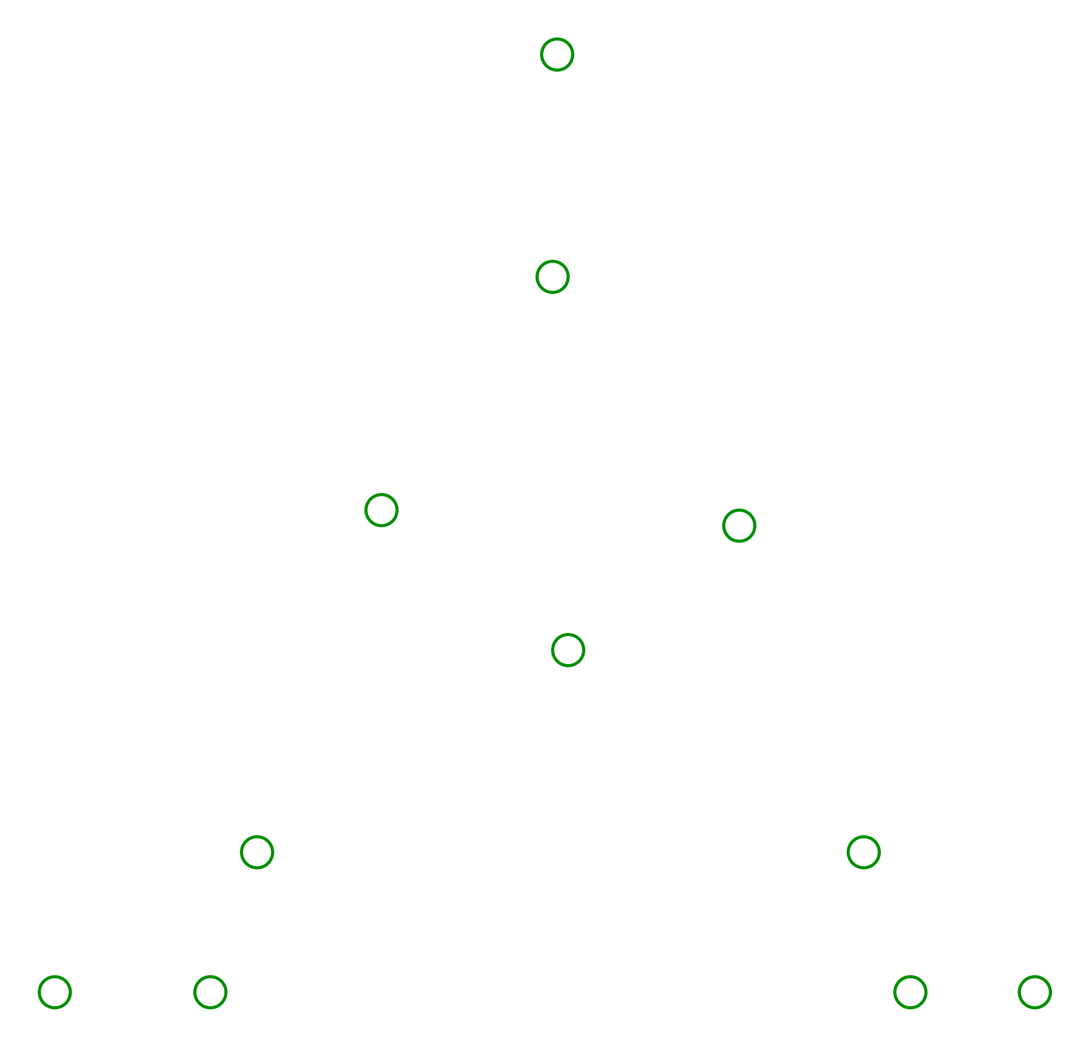
$n = 11$ points; Want <u>all</u> lines through ≥ 4 pts.

Fitting degree 4 curve Q as earlier doesn't work.

Why?

Correct answer has 5 lines. Degree 4 curve can't have 5 factors!

Going Further: Example 2 [Guruswami+S. '98]



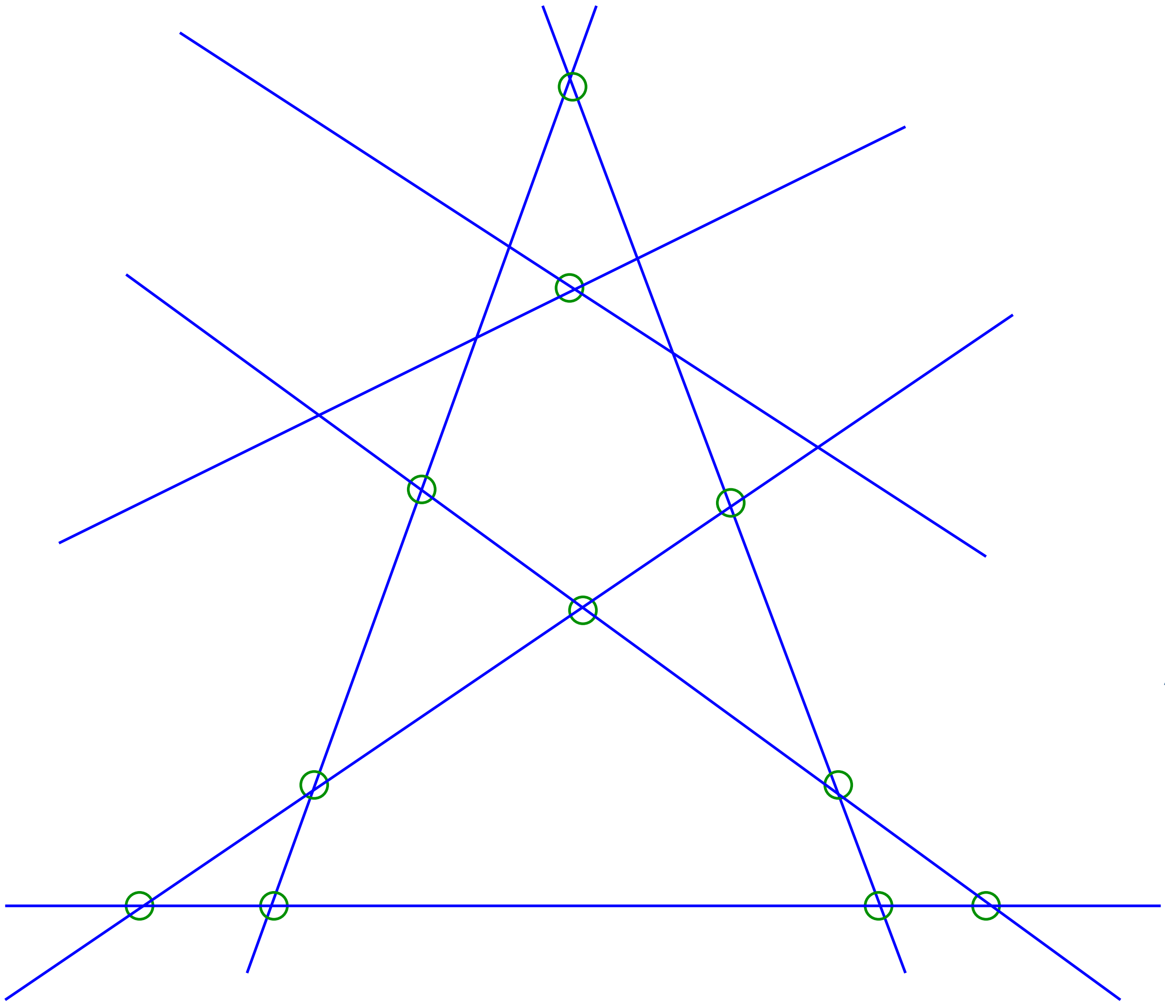
$n = 11$ points; Want all lines through ≥ 4 pts.

Fit degree 7 poly. $Q(x, y)$ passing through each point twice.

$Q(x, y) = \cdots$ (margin too small)

Plot all zeroes ...

Going Further: Example 2 [Guruswami+S. '98]



$n = 11$ points; Want all lines through ≥ 4 pts.

Fit degree 7 poly. $Q(x, y)$ passing through each point twice.

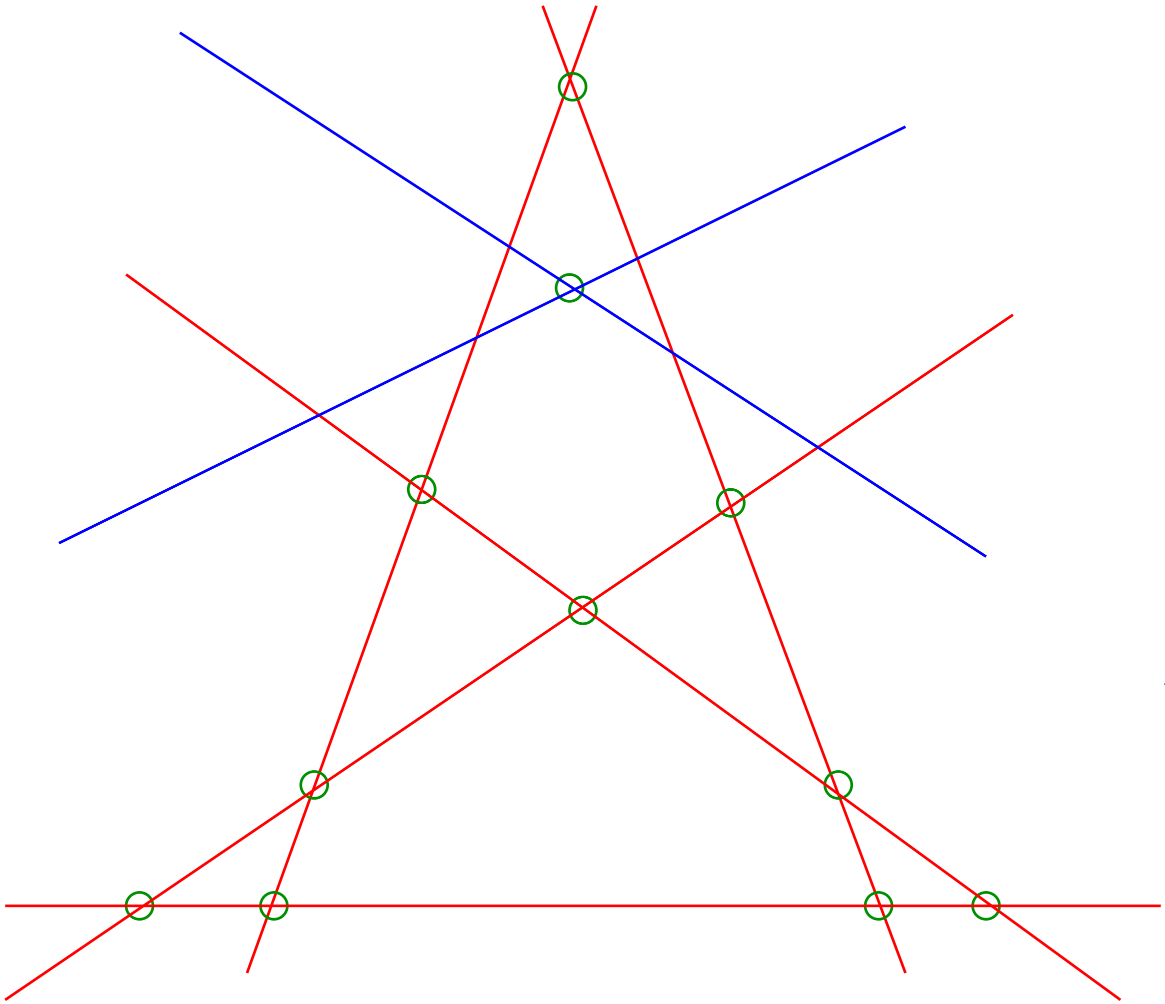
$Q(x, y) = \cdots$

(margin too small)

Plot all zeroes ...

All relevant lines emerge!

Going Further: Example 2 [Guruswami+S. '98]



$n = 11$ points; Want all
lines through ≥ 4 pts.
Fit degree 7 poly. $Q(x, y)$
passing through each
point twice.
$Q(x, y) = \cdots$
(margin too small)
Plot all zeroes ...
All relevant lines emerge!

Where was the gain?

- Requiring Q to pass through each point twice, effectively doubles the # intersections between Q and line.
  - So # intersections is now 8.
- On the other hand # constraints goes up from 11 to 33. Forces degree used to go upto 7 (from 4).
- But now # intersections is less than degree!

Can pass through each point twice with less than twice the degree!

- Letting intersection multiplicity go to ∞ gives decoding algorithm for upto $1 - \sqrt{R}$ errors.

Summary

- Can correct errors in Reed-Solomon codes well beyond "half the distance" (Hamming) barrier!

Summary

- Can correct errors in Reed-Solomon codes well beyond "half the distance" (Hamming) barrier!
- Matches best known "combinatorial" bounds on list-decodability.

Summary

- Can correct errors in Reed-Solomon codes well beyond "half the distance" (Hamming) barrier!
- Matches best known "combinatorial" bounds on list-decodability.
- Open Question: Correct more errors, or show this leads to exponentially large lists!

Summary

- Can correct errors in Reed-Solomon codes well beyond "half the distance" (Hamming) barrier!
- Matches best known "combinatorial" bounds on list-decodability.
- Open Question: Correct more errors, or show this leads to exponentially large lists!
- Techniques: The polynomial method, and the method of multiplicities!

The Polynomial Method

- **Goal:** Understand some "combinatorial parameters" of some algebraically nice set. E.g.,

The Polynomial Method

- Goal: Understand some "combinatorial parameters" of some algebraically nice set. E.g.,
  - Minimum number of points in the union of ℓ sets where each set is t points from a degree k polynomial = ?
  - Minimum number of points in $K \subseteq \mathbb{F}_q^n$ such that K contains a line in every direction.

The Polynomial Method

- Goal: Understand some "combinatorial parameters" of some algebraically nice set. E.g.,
- Method:
  - Fit low-degree polynomial Q to the set K.
  - Infer Q is zero on points outside K, due to algebraic niceness.
  - Infer lower bound on degree of Q (due to abundance of zeroes).
  - Transfer to bound on combinatorial parameter of interest.

Kakeya Sets

- Definition: $K \subseteq \mathbb{F}_q^n$ is a Kakeya set if it contains a line in every direction.
- Question: How small can K be?

Kakeya Sets

- Definition: $K \subseteq \mathbb{F}_q^n$ is a Kakeya set if it contains a line in every direction.
- Question: How small can K be?
- Bounds (till 2007):
  $\forall K, \quad |K| \geq q^{n/2}$
  $\exists K, \quad |K| \leq q^n$

Kakeya Sets

- Definition: $K \subseteq \mathbb{F}_q^n$ is a Kakeya set if it contains a line in every direction.
- Question: How small can K be?
- Bounds (till 2007):
  $\forall K, \quad |K| \geq q^{n/2}$
  $\exists K, \quad |K| \leq\approx (q/2)^n$ [Mockenhaupt & Tao]

Kakeya Sets

- Definition: $K \subseteq \mathbb{F}_q^n$ is a Kakeya set if it contains a line in every direction.
- Question: How small can K be?
- Bounds (till 2007):
  $\forall K, \quad |K| \geq q^{n/2}$
  $\exists K, \quad |K| \leq\approx (q/2)^n$ [Mockenhaupt & Tao]
- In particular, even exponent of q unknown!

Kakeya Sets

- Definition: $K \subseteq \mathbb{F}_q^n$ is a Kakeya set if it contains a line in every direction.
- Question: How small can K be?
- Bounds (till 2007):
  $\forall K, \quad |K| \geq q^{n/2}$
  $\exists K, \; |K| \leq\approx (q/2)^n$ [Mockenhaupt & Tao]
- In particular, even exponent of q unknown!
- [Dvir'08]'s breakthrough: $\forall K, \quad |K| \geq q^n/n!$

Kakeya Sets

- Definition: $K \subseteq \mathbb{F}_q^n$ is a Kakeya set if it contains a line in every direction.
- Question: How small can K be?
- Bounds (till 2007):
  $\forall K, \quad |K| \geq q^{n/2}$
  $\exists K, \ |K| \leq\approx (q/2)^n$ [Mockenhaupt & Tao]
- In particular, even exponent of q unknown!
- [Dvir'08]'s breakthrough: $\forall K, \quad |K| \geq q^n/n!$
- Subsequently [Dvir, Kopparty, Saraf, S.]
  $\forall K, \quad |K| \geq (q/2)^n$

Polynomial Method and Kakeya Sets

- [Dvir'08]'s analysis:
  - Fit low-degree polynomial Q to K. (Interpolation $\Rightarrow$ Degree not too high if K not large.)

Polynomial Method and Kakeya Sets

- [Dvir'08]'s analysis:
  - Fit low-degree polynomial Q to K. (Interpolation $\Rightarrow$ Degree not too high if K not large.)
  - Show homogenous part of Q zero at y if line in direction y contained in K.

Polynomial Method and Kakeya Sets

- [Dvir'08]'s analysis:
  - Fit low-degree polynomial Q to K. (Interpolation $\Rightarrow$ Degree not too high if K not large.)
  - Show homogenous part of Q zero at y if line in direction y contained in K.
  - Conclude homogenous part is zero too often!

Polynomial Method and Kakeya Sets

- [Dvir'08]'s analysis:
  - Fit low-degree polynomial Q to K. (Interpolation $\Rightarrow$ Degree not too high if K not large.)
  - Show homogenous part of Q zero at y if line in direction y contained in K.
  - Conclude homogenous part is zero too often!
- [Saraf + S.], [Dvir + Kopparty + Saraf + S.]:
  - Fit Q to vanish many times at each point of K.
  - Yields better bounds!

Conclusions

- Importance of model of error.

Conclusions

- Importance of model of error.
- Virtues of relaxing some notions (e.g., list-decoding vs. unique-decoding)

Conclusions

- Importance of model of error.
- Virtues of relaxing some notions (e.g., list-decoding vs. unique-decoding)
- New algorithmic insights: Can be useful outside the context of list-decoding (e.g., [Koetter-Vardy] Soft-decision decoder).

Conclusions

- Importance of model of error.
- Virtues of relaxing some notions (e.g., list-decoding vs. unique-decoding)
- New algorithmic insights: Can be useful outside the context of list-decoding (e.g., [Koetter-Vardy] Soft-decision decoder).
- Central open question:

Conclusions

- Importance of model of error.
- Virtues of relaxing some notions (e.g., list-decoding vs. unique-decoding)
- New algorithmic insights: Can be useful outside the context of list-decoding (e.g., [Koetter-Vardy] Soft-decision decoder).
- Central open question:

  Constructive list-decodable *binary* codes of rate $1 - H(\rho)$ correcting ρ-fraction errors !!
  Corresponding question for large alphabets resolved by [ParvareshVardy05, GuruswamiRudra06].

Conclusions

- Importance of model of error.
- Virtues of relaxing some notions (e.g., list-decoding vs. unique-decoding)
- New algorithmic insights: Can be useful outside the context of list-decoding (e.g., [Koetter-Vardy] Soft-decision decoder).
- Central open question:

  Constructive list-decodable *binary* codes of rate $1 - H(\rho)$ correcting ρ-fraction errors !!
  Corresponding question for large alphabets resolved by [ParvareshVardy05, GuruswamiRudra06].
- New (?) mathematical insights.

Conclusions

- Importance of model of error.
- Virtues of relaxing some notions (e.g., list-decoding vs. unique-decoding)
- New algorithmic insights: Can be useful outside the context of list-decoding (e.g., [Koetter-Vardy] Soft-decision decoder).
- Central open question:

  Constructive list-decodable *binary* codes of rate $1 - H(\rho)$ correcting ρ-fraction errors !!
  Corresponding question for large alphabets resolved by [ParvareshVardy05, GuruswamiRudra06].
- New (?) mathematical insights.
- Challenge: Apply existing insights to other practical settings.

Thank You !!