List Decoding of Error Correcting Codes

Madhu Sudan

Laboratory for Computer Science
MIT

The Problem of Information Transmission

- Want to transmit digital information over noisy channel.
- How to overcome noise and achieve reliable communication?

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 (recovers message with exponentially small error), 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.

  (q-ary channel error threshold $= 1 - \frac{1}{q}$.)

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 $x_i \neq y_i$.

Hamming (contd.)

- 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.
  - $\exists$ binary codes mapping k bits to $n = O(k)$ bits, with $\Delta(C)/n \to \frac{1}{2}$ [Gilbert '50s].
  - $\Delta(C)/n > \frac{1}{2}$ implies $k = \log n$ [Plotkin '50s].

Gap between Hamming & Shannon

- Shannon's theory: Can deal with channels that err with probability less than 50%.
- Hamming theory:
  - Can correct $d/2$ errors, with code of distance d.
  - Binary code of positive rate has $d/n < 1/2$.
  - Conclude: Can correct less than 25% errors.

List-decoding

- Extend notion of decoding to allow "list" of the ℓ most likely candidates.

- Code C is (p, ℓ)-error-correcting, if it has decoding algorithm (of unbounded computational power) correcting pn errors with lists of size ℓ. (Formally, $C \subseteq \Sigma^n$ is (p, ℓ)-error-correcting-code if $\forall\ r \in \Sigma^n$, at most ℓ codewords $c \in C$ satisfy $\Delta(r, c) \leq p \cdot n$.)

- Notion due to [Elias '57, Wozencraft '58]. C of distance d is $((\frac{1}{2} \cdot \frac{d}{n}), 1)$-error-correcting.

- How many errors can we correct in this relaxed setting?

List-decoding and the gap between Hamming & Shannon

- [Zyablov-Pinsker '70s] $\forall \epsilon > 0, \exists \ell < \infty$ and codes of rate $1 - H(p) - \epsilon$ that are (p, ℓ)-error-correcting.

- Narrows gap between probabilistic and adversarial models:
  - Either transmit at rate $1 - H(p)$ and recover message, where p-fraction of error introduced by random noise.
  - Or transmit at rate $1 - H(p)$ and recover small list including message, where adversary introduces p-fraction error.

- Catch: [ZP]-result non-constructive.

List-decoding: Algorithmic Results

- Problem highlighted by [Goldreich-Levin].
- First interesting poly-time algorithm in [S' 96]. Since then lots of work [Shokrollahi-Wasserman '97], [Guruswami-S.'98-00] etc.

Theorem 1 $\forall \epsilon > 0$, exists a binary code of rate $O(\epsilon^4)$ with polynomial time encoding algorithm and polynomial time list-decoding algorithm to decode from $\frac{1}{2} - \epsilon$ errors. (Generalizes to q-ary alphabet.)

(Analogous to [Elias] result in Shannon model.)

Reed-Solomon Codes & List-decoding

Aside: What to do with a list of candidates?

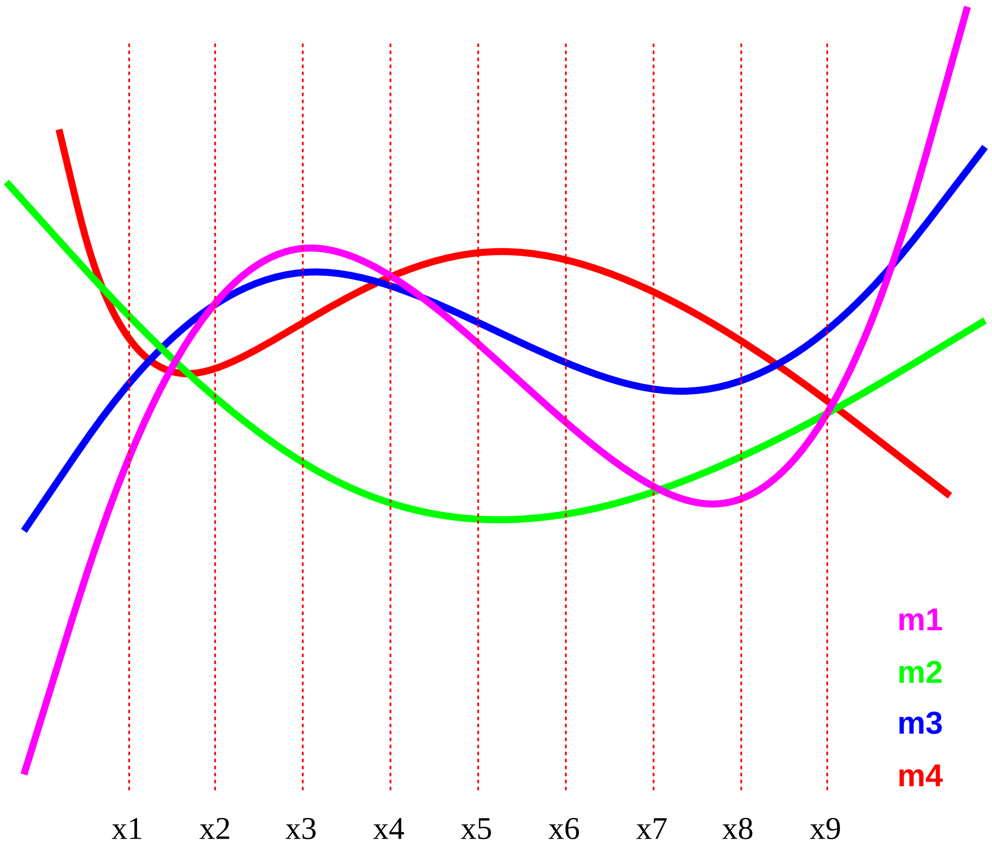
- Answer 1: If noise is probabilistic, probability list contains two elements is very low, for reasonable noise models! (Note: Algorithm independent of model!)

- Answer 2: Can disambiguate elements of list, using small amount of extra information, if a second, more reliable channel is available. [Guruswami '03].

- Answer 3: If messages are appropriately encrypted, then error-channel has to solve hard problems to create confusion. [Lipton et al., Micali et al.]

Reed-Solomon Codes & List-decoding

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 Σ be a finite field.
- Code specified by $k, n, \alpha_1, \ldots, \alpha_n \in \Sigma$.
- Message: $\langle c_0, \ldots, c_k \rangle \in \Sigma^{k+1}$ 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. Best possible! Commonly used (CDs, DVDs etc.).

Reed-Solomon Decoding

Restatement of the problem:

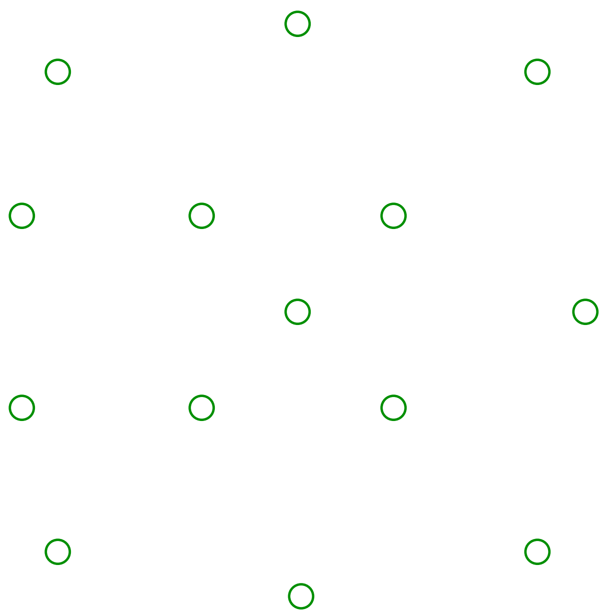
- Input: n points $(\alpha_i, y_i) \in \mathbb{F}_q^2$; agreement parameter t
- Output: **All** degree k polynomials $p(x)$ s.t. $p(\alpha_i) = y_i$ for at least t values of i.

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: Example 1

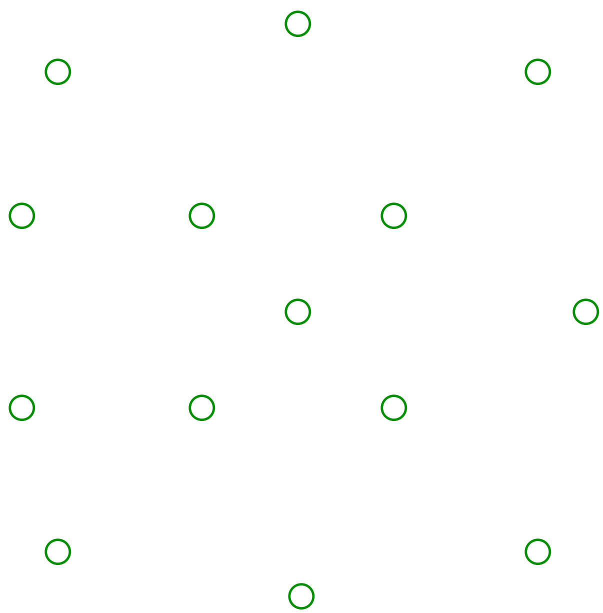
$n = \mathbf{14}$ points; Want all *lines* through at least **5** points.



Algorithm Description: Example 1

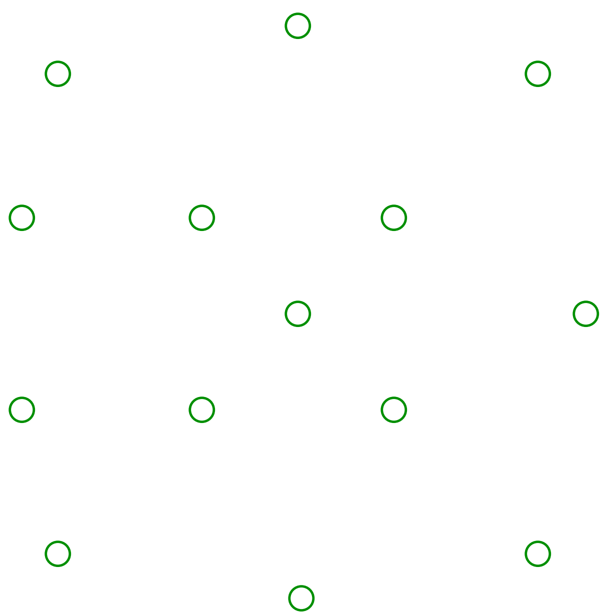
$n = \mathbf{14}$ points; Want all *lines* through at least **5** points.

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



Algorithm Description: Example 1

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



Find deg. 4 poly. $Q(x, y) \not\equiv 0$

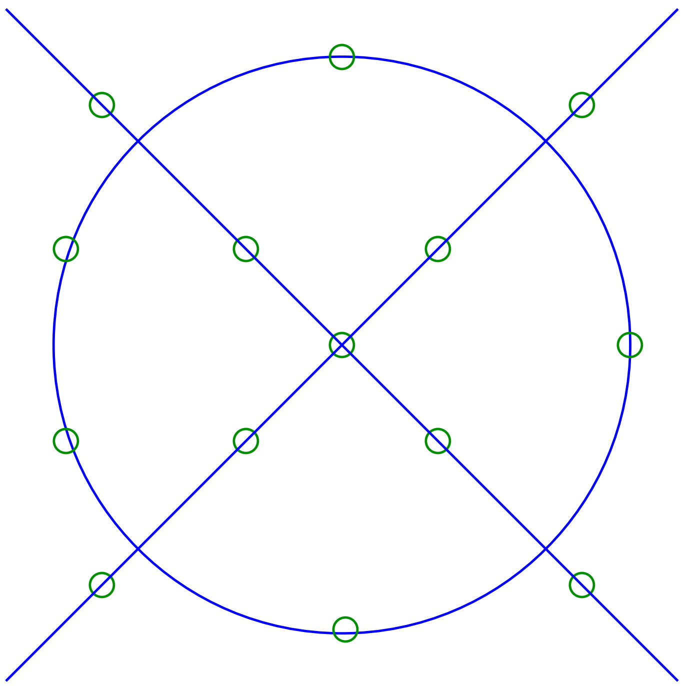
s.t. $Q(\alpha_i, y_i) = 0$ for all points.

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

Let us plot all zeroes of Q ...

Algorithm Description: Example 1

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



Find deg. 4 poly. $Q(x, y) \not\equiv 0$

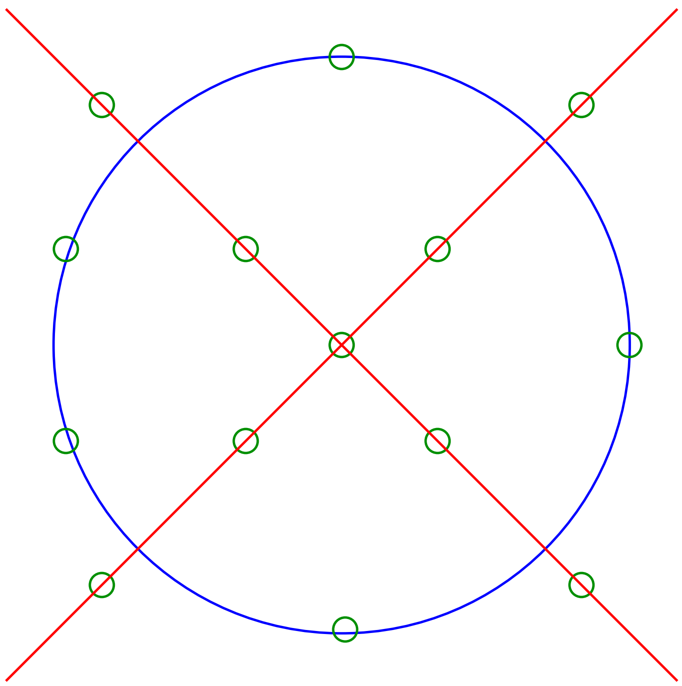
s.t. $Q(\alpha_i, y_i) = 0$ for all points.

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

Let us plot all zeroes of Q ...

Algorithm Description: Example 1

$n = \mathbf{14}$ 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 $\Longrightarrow \ell$ is a factor of Q

Generally

Lemma 1: $\exists Q$ with $\mathsf{deg}_x(Q), \mathsf{deg}_y(Q) \leq D = \sqrt{n}$ passing thru any n points.

Lemma 2: If Q with $\mathsf{deg}_x(Q), \mathsf{deg}_y(Q) \leq D$ intersects $y - p(x)$ with $\mathsf{deg}(p) \leq d$ 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
2. Fast algorithms for factorization of bivariate polynomials exist.

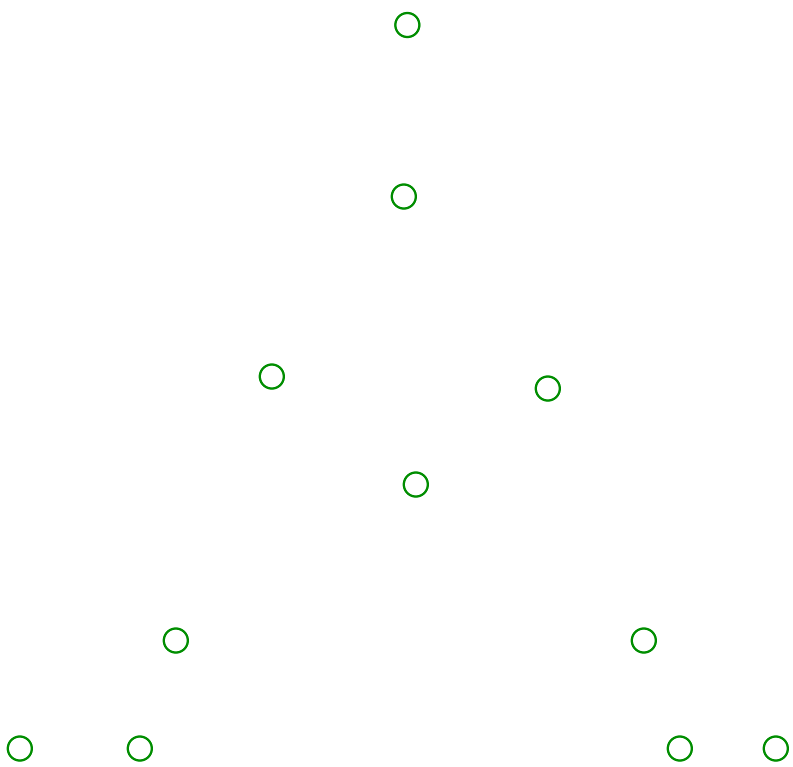
Thm: Can find polynomials having agreement $t \geq (k+1)\sqrt{n}$.

Can fine tune parameters a bit to get:

Thm: Can find polynomials having agreement $t \geq \sqrt{2kn}$.

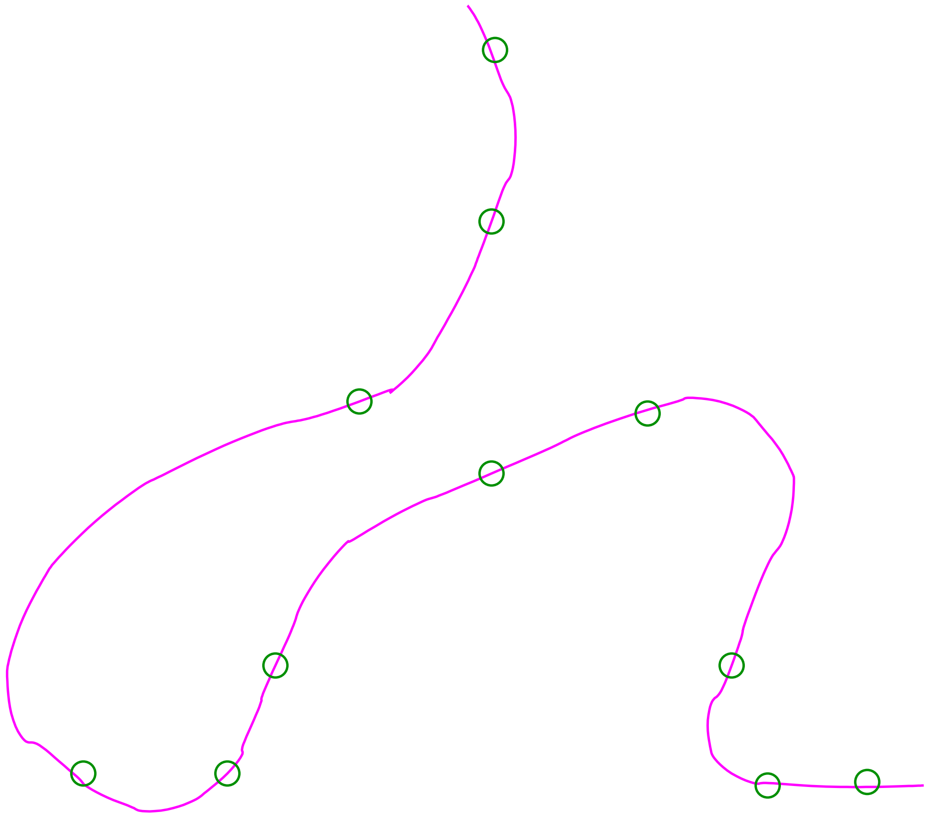
Does not meet combinatorial bounds!

Going Further: Example 2



$n = 11$ points; Want **all** lines through $\geq$ **4** pts.

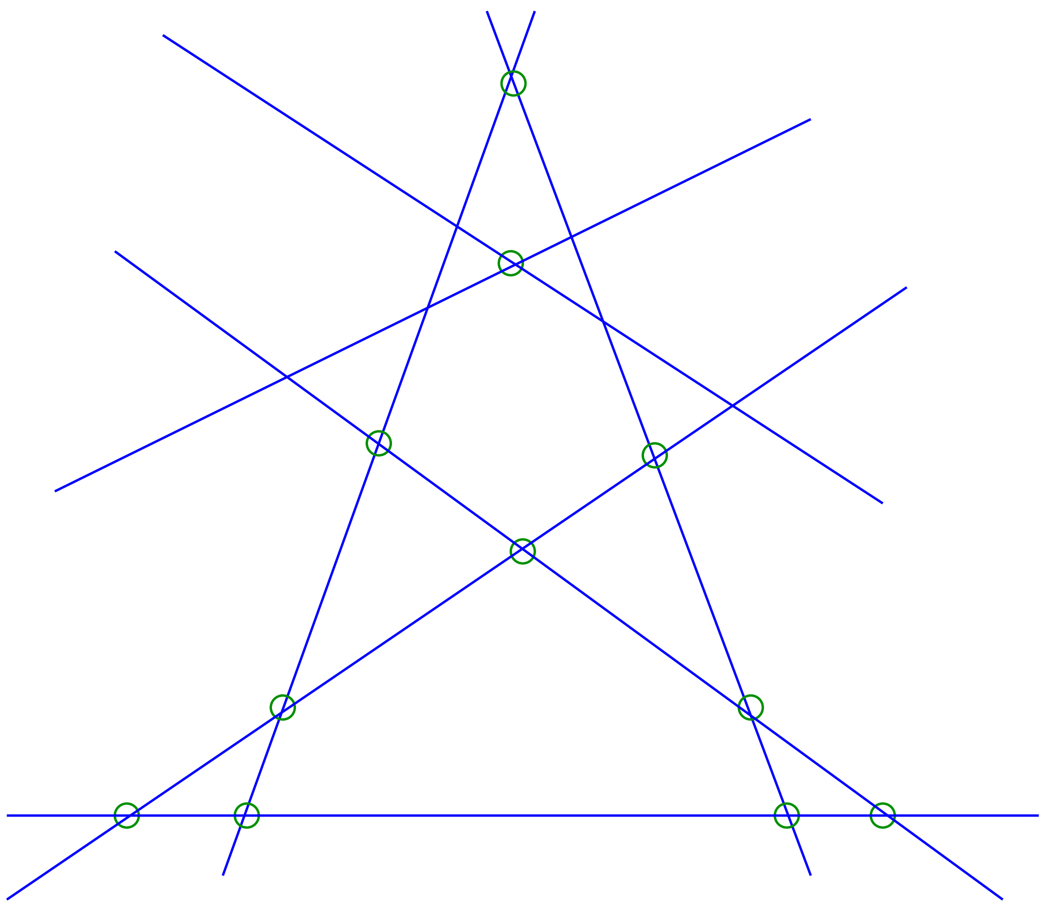
Going Further: Example 2



$n = 11$ points; Want **all** lines through $\geq$ **4** pts.

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

Going Further: Example 2



$n = 11$ points; Want **all** lines through $\geq$ **4** pts.

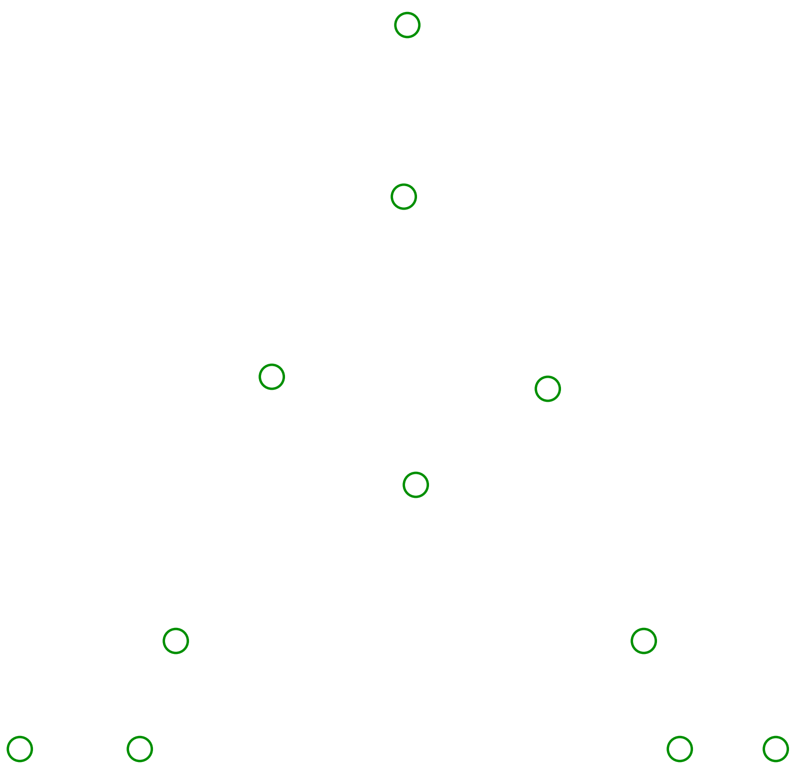
Fitting degree 4 curve Q as earlier doesn't work.

Why?

Correct answer has 5 lines.

Degree 4 curve can't have 5 factors!

Going Further: Example 2

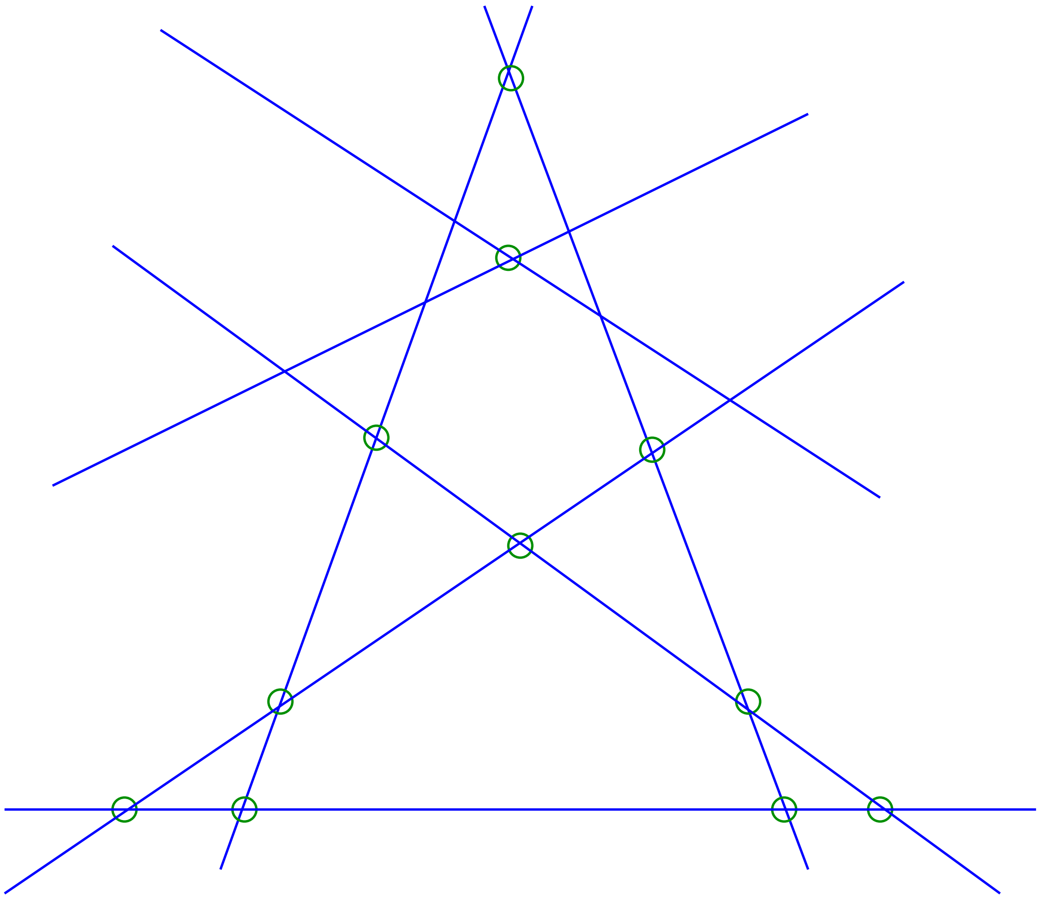


$n = 11$ points; Want **all** lines through $\geq$ **4** pts.

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

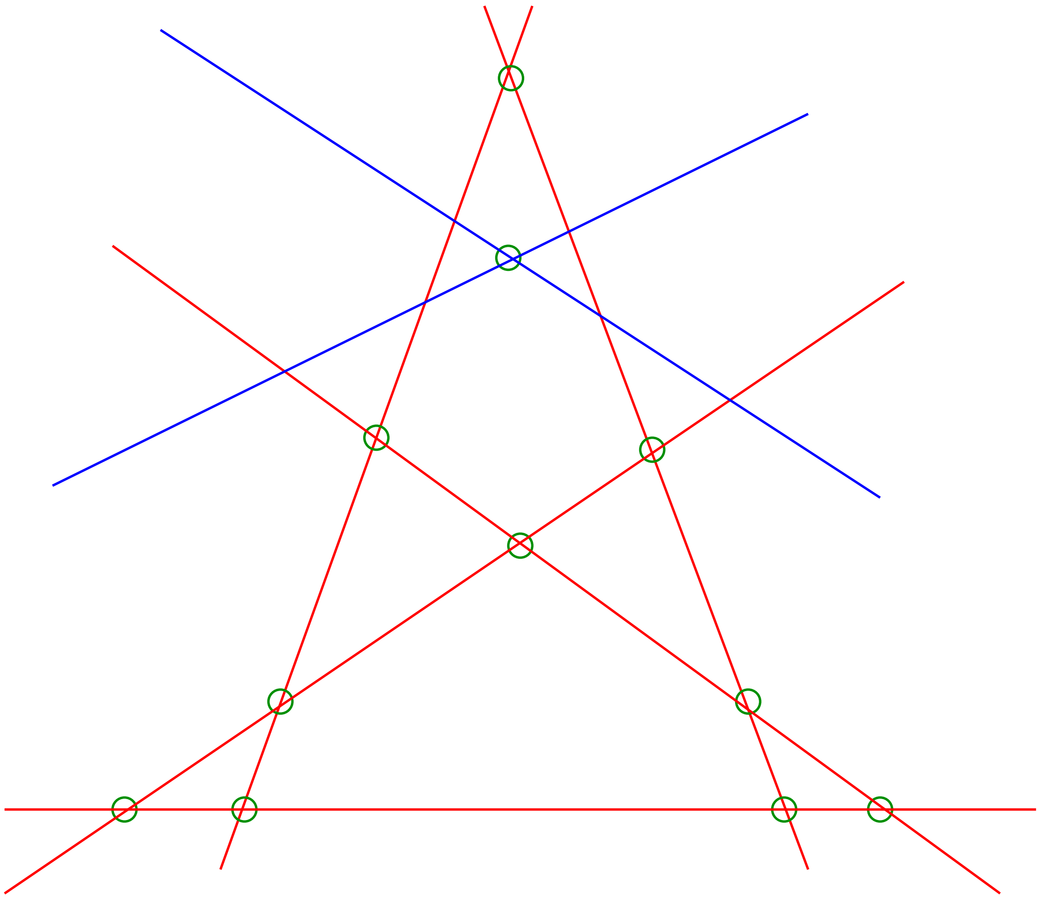


$n = 11$ points; Want **all** lines through $\geq$ **4** pts.

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



$n = 11$ points; Want **all** lines through $\geq$ **4** pts.

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

$Q(x, y) = \cdots$

Plot all zeroes ....

All lines emerge!

Where was the gain?

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

In previous example: Passing through each point twice

- increased degree of Q from 4 to 7
- doubled # intersections between "target" line ℓ and Q.
- $8 > 7$ so any such ℓ must factor out of Q. QED.

Decoding Algorithm Summary

Task: Find deg. k polys. p s.t. $p(\alpha_i) = y_i$ for $\geq t$ values of i.

- Pick suitable parameters, namely "degree" D of Q and multiplicity r of each point, such that $D \simeq \sqrt{knr(r+1)}$.
- Fit a "degree" D polynomial $Q(x, y)$ that passes through each point (α_i, y_i) at least r times.
- Factor $Q(x, y)$ and look for candidate polynomials p among factors of form $y - p(x)$.

  (If $t > D/r$, this will find all relevant polynomials p.)

Summary

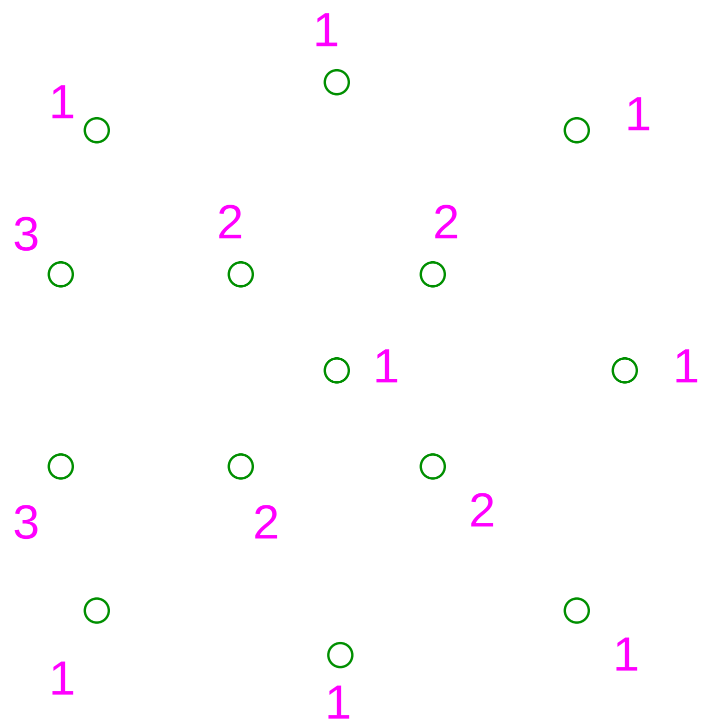
With appropriate multiplicity, and appropriate "degree", get

Theorem: [Guruswami-S.'98]: Can solve decoding problem in polynomial time if # agreements $> \sqrt{kn}$.

Generalizations: Leads to nice definition of codes based on ideals in commutative rings.

An Additional Benefit

Can handle weighting of points (**not conceivable earlier ...**)



Sample question:
Find lines through points
of total weight ≥ 7.

Solution strategy:
Fit curve that passes thru pts
in *proportion* to their weights

Weighted Reed-Solomon Decoding

- Given: n points $(\alpha_i, y_i) \in \mathbb{F}_q^2$; and ***weights*** w_i;
  Agreement parameter W
- Task: Find **all** deg. k poly's p s.t. $\sum_{i:p(\alpha_i)=y_i} w_i \geq W$.

Thm [Guruswami-S.'98]: Can solve above efficiently if $W > \sqrt{k \sum_i w_i^2}$.

- Left open in [GS'98]: When do weights make sense? What weights to use?

[Koetter-Vardy'01]: Algebraic Soft-Decision Decoding

- Starting Point: Weighted Reed-Solomon Decoding.
- Major Issues Addressed:
  - How to assign weights for specific channels? Non-trivial, even for "trivial" channels.
  - How to improve runtime (when running this complex procedure)?
- Consequence: Dramatic improvement in performance of RS codes, in some cases; "WSJT" (publicly available Ham Radio software) reports "3dB" gain using KV algorithm.

Summary

- List decoding is meaningful, useful, and feasible.
- Demonstrates that a little extra computational power can cope with much more errors.
- Already has changed the theoretical perspective on ability to cope with errors.
- Challenge ahead: Any more practical uses?