Today

- $q$-ary codes.
- Algebraic-geometry & Codes.
- Proof of concept.
- Statement of big claim.

Hamming balls in $\mathbb{F}_q^n$

- Let $H_q(p) = p \log_q \frac{q-1}{p} + (1 - p) \log_q \frac{1}{1-p}$.
- Let $\text{Vol}_q(r, n) = \text{Volume of Hamming ball of radius } r \text{ in } \mathbb{F}_q^n$.
- Then $\text{Vol}_q(r, n) = q^{(H_q(p) + o(1))n}$.
- $q$-ary GV bound:
  Theorem: There exists an infinite family of $q$-ary codes of rate $R$ and relative distance $\delta$ satisfying
  \[ R \geq 1 - H_q(\delta) \]

Understanding $q$-ary GV bound

- To get a sense of $H_q(\delta)$, fix $0 < \delta < 1$ and let $q \to \infty$. Get
  \[ R \geq 1 - \delta - H_2(\delta)/\log q - o(1/\log q) \]
- Contrast with Singleton (Project on to first $k-1$ coordinates) upper bound on rate:
  \[ R \leq 1 - \delta \]
- I.e., GV bound approaches Singleton bound at logarithmic rate in $q$.
- Is this best possible? RS codes achieve Singleton bound and $q$ is pretty small!

Today:
Algebraic-geometry (AG) codes: Achieve
\[ R \geq 1 - \delta - \frac{1}{\sqrt{q} - 1} \]
- Needs $q$ square and prime power.
- Clearly better for large $q$.
- In fact, better for $q \geq 49$. 
Algebraic-geometry codes

- Conceived by Goppa in late 70’s - early 80’s.
- 1982 - Surprising breakthrough …
  - Due to Tsfasman, Vladuts, Zink.
  - Based on some prior work of Ihara.
  - Codes better than random for suff. large, but constant sized, alphabet.
- Almost unique in history of explicit constructions ….

Motivation: Bivariate Codes

- Consider codes obtained by evaluations of bivariate polynomials \( Q(x, y) \) of deg. \( \leq l \) in each variable.
- Gives \( \left[ q^2, l^2, \left(1 - \frac{l}{q}\right)^2\right]_q \) code.
- Contrast w. \( \left[ q^2, l^2, q^2 - l^2\right] q^2 \) RS code.
  - Bivariate alphabet smaller.
  - Distance smaller by \( 2l(q - l) \).
- Why this \( q - l \) deficit?
  - On axis-parallel line \( l \) points zero imply \( q \) points zero.
  - For every line defect of \( q - l \).

AG code idea

- Don’t evaluate poly on all points on plane.
- Ideally, don’t use more than \( l \) points on line.
- Pragmatically, don’t use much more than \( l \) points on line.
- But there exist other bad examples. Degree 2 curves, Degree 3 curves.
- So, don’t use too many points on any (low-degree) curve.
- How to find such points? Use points on some low-degree curve.

Algebraic curves in the plane

Defn: Given a bivariate polynomial \( R(x, y) \) of total degree \( D \), the set of points
\[
\{(a, b) \in \Sigma^2 \mid R(a, b) = 0\}
\]
is called an algebraic curve of degree \( D \) in the plane.

Basic result from algebraic geometry:
Nice algebraic curves don’t meet other nice algebraic curves very often.

Bezout’s Thm: Curves \( R_1, R_2 \) of deg. \( D_1, D_2 \) share at most \( D_1 D_2 \) common zeroes.
Example (stolen from Shokrollahi)

- Let $q = 13$
  \[ R(x, y) = y^2 - 2(x - 1)x(x + 1). \]
- Code obtained by evaluating (certain) polynomials at zeroes of $R$.
- Fact: There exist 19 zeroes of $R$.
- Legal polynomials: linear combinations of $\{1, x, y, x^2, xy, x^3\}$.
- If legal poly has 6 zeroes, then it is identically zero.
- Gives $[19, 6, 13]_{13}$ code.
  (RS would give $[19, 6, 14]_{19}$ code.)

Codes from Planar Curves

- Generally:
  - Evaluating polys of deg. $\leq l$
  - At zeroes of $R$, irreducible, of degree $D$, with $n$ zeroes.
  - Gives $[n, k, n - Dl]_q$ code,
    \[
    k = \begin{cases} 
    \binom{l+2}{2} & \text{if } l < D \\
    \binom{l+2}{2} - \binom{l-D+2}{2} & \text{if } l \geq D
    \end{cases}
    \]
- Distance by Bezout’s theorem.

Finding good curves

How to find $R$ with large $n$?

- No general method.
- But some well-known curves do well. e.g. Hermitian curve for $q = r^2$:
  - $x^{r+1} - y^r - y = 0$  
    - has $r^3 + 1$ points.
    - Gives $[r^3 + 1, \binom{r+2}{2}, r^3 + 1 - (r)(r+1)]_{r^2}$ code.
- Bivariate polys gave $[r^4, \binom{r+2}{2}, r^4 - r^3]_{r^2}$. 

Going to Higher Dimension

- So far, went from alphabet $n$ to (at best) $\sqrt{n}$.
- To do better need more variables.
- General AG codes:
  - Pick $m$ variables.
  - Put $m - 1$ polynomial constraints.
  - Evaluate polynomials on zeroes.
“State-of-the-art” codes

[Garcia & Stichtenoth]

- \( q = r^2 \).
- Variables \( x_1, \ldots, x_m, y_1, \ldots, y_m \).
- Constraints:
  \[
  x_1^{r+1} = y_1^r + y_1.
  x_2x_1 = y_1.
  x_2^{r+1} = y_2^r + y_2.
  \vdots
  x_m x_{m-1} = y_{m-1}.
  x_m^{r+1} = y_m^r + y_m.
  \]
- \# zeroes \( \geq (r^2 - 1)r^m \).

Keeping track of distance

- Bezout’s theorem becomes weak.
- Polynomials ordered by “order”.
  Order axioms:
  - \( \text{ord}(f + g) \leq \max\{\text{ord}(f), \text{ord}(g)\} \).
  - \( \text{ord}(f \ast g) = \text{ord}(f) + \text{ord}(g) \).
  - \( f \) has at most \( \text{ord}(f) \) zeroes.
  - Polynomials of all except \( g \) orders exist.
  - \( g = \text{genus of curve} \).
  - Genus of Garcia-Stichtenoth curve \( \leq (r+1)r^m \).
- AG codes follow.

Summary: RS vs. AG

<table>
<thead>
<tr>
<th></th>
<th>RS</th>
<th>AG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coordinates</td>
<td>( \mathbb{F}_q )</td>
<td>Points on curves</td>
</tr>
<tr>
<td>Messages</td>
<td>Polynomials ( \deg &lt; k )</td>
<td>Polynomials order ( &lt; k )</td>
</tr>
<tr>
<td>Encoding</td>
<td>Evaluations</td>
<td>Evaluations</td>
</tr>
<tr>
<td>Distance</td>
<td>( n - k + 1 )</td>
<td>( n - k + 1 )</td>
</tr>
<tr>
<td>Dimension</td>
<td>( k )</td>
<td>( k - \text{genus} )</td>
</tr>
<tr>
<td>Axioms</td>
<td>zeroes ( \leq \deg )</td>
<td>zeroes ( \leq \text{order} )</td>
</tr>
<tr>
<td></td>
<td>Sum rule</td>
<td>Sum rule</td>
</tr>
<tr>
<td></td>
<td>Product rule</td>
<td>Product rule</td>
</tr>
<tr>
<td></td>
<td>( \text{dim.} &gt; \deg )</td>
<td>( \text{dim.} &gt; \text{order} )</td>
</tr>
</tbody>
</table>

Computational requirements

- Classical AG codes computable in \( O(n^{30}) \) time.
- Newer AG codes computable in \( O(n^{17}) \) time.
- Rumors of \( O(n^2) \) time computability.
- Belief in explicit constructions.
Some best known codes

Fix $q = 2$. Given $k$ and $d/n = \frac{1}{2} - \epsilon$, what is the best known code? (Will allow $\epsilon = \epsilon(n)$).

- Random code: $n = O\left(\frac{k}{\epsilon^2}\right)$.
- RS o Hadamard: $n = \frac{k^2}{\epsilon^2}$.
- AG o Hadamard: $n = O\left(\frac{k}{\epsilon^3 \log(1/\epsilon)}\right)$.
- [ABNNR]: $n = O\left(\frac{k}{\epsilon^3}\right)$. (Polylog space constructible). 

© Madhu Sudan, Fall 2002: Essential Coding Theory, MIT 6.896