# Discussion 5A Recap

Tyler Zhu

February 28, 2020

## 1 Lagrange Interpolation

Suppose we have points $(x_1, y_1), \dots, (x_{d+1}, y_{d+1})$. Then there is a unique polynomial of degree at most $d$ passing through these points. Lagrange's Interpolation Formula tells us exactly what this is. If we let

$$\Delta_i = \frac{\prod_{j\neq i}(x - x_j)}{\prod_{j\neq i}(x_i - x_j)} = \frac{(x - x_1)\dots(x - x_{i-1})(x - x_{i+1})\dots(x - x_{d+1})}{(x_i - x_1)\dots(x_i - x_{i-1})(x_i - x_{i+1})\dots(x_i - x_{d+1})},$$

then our polynomial is

$$P(x) = \sum_{i=1}^{d+1} y_i\Delta_i = y_1\Delta_1 + y_2\Delta_2 + \dots + y_{d+1}\Delta_{d+1}.$$

The intuition is that each of the $\Delta_i$'s is a basis for generating our polynomial. Notice that $\Delta_i(x_j) = 0$ for all $j \neq i$, and $\Delta_i(x_i) = 1$, so we can combine these $\Delta_i$'s in a smart way to force this polynomial to pass through all of our desired points.

## 2 Error Correcting Codes

There are two types of errors we will investigate in this course: *erasure* errors and *general*, or corruption, errors. The difference is knowing where the errors are; in the former you know, in the latter you don't.

- If I have up to $k$ erasures, then I need to send $n + k$ packets. I can create a polynomial $P(x)$ of degree at most $n - 1$ where $P(i) = m_i$, so that even if $k$ packets drop, I have $n$ points left to recover the original polynomial, and any potentially dropped packets in my original message.
- If I have up to $k$ corruptions, then I need to send $n + 2k$ packets if we use the Berlekamp-Welch algorithm. Note that I can't just send $n+k$ packets, since I can't tell which packets were corrupted.