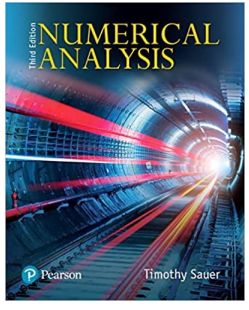
计算方法

刘景铖

计算机软件新技术国家重点实验室
南京大学

课程基本信息

- 教师：刘景铖
- Email：liu@nju.edu.cn
- Office hour：周四 14:00-16:00，计算机系516
- 课程主页：https://tcs.nju.edu.cn/wiki
- 数值分析 （Numerical Analysis） （原书第2版） Timothy Sauer, 机械工业出版社



参考：

- Numerical Algorithms: Methods for Computer Vision, Machine Learning, and Graphics. Justin Solomon. CRC Press
- Lx=b,拉普拉斯方程求解以及它的应用（Lx=b, Laplacian Solver and Their Algorithmic Applications） Nisheeth K. Vishnoi

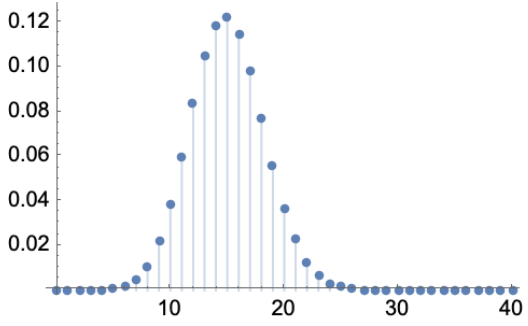
课程QQ群

- **作业每两周发布一次**
- **第二次作业将于今天下午发布**
- **作业Email: cm_nju_2023@163.com**
- **QQ群可以讨论课程相关的问题**
- **作业可以小组讨论（<=3人）**
  - 清晰标明合作者，及各自的贡献
  - 使用了的参考资料必须注明
  - 但写作必须由自己独立完成，不可照抄
  - 类似地，别人提供的想法也必须注明
- 分辨合作与作弊之间的区别
  - 参与讨论之前，请先花时间自己进行思考
  - 避免参与你不能提供贡献的讨论
  - 作弊不仅让你丧失一次学习的机会
  - 还会影响你以后的自信心
- 对学术不诚信的行为零容忍

对作弊的认定参照http://www.acm.org/publications/policies/plagiarism_policy

回顾

- 方程求根：给定f，求x使得$f(x) = 0$
  - 连续函数：二分法
  - 1-Lipschitz: 转换为不动点迭代方程后使用不动点迭代法
  - 根的敏感性
- 这节课：
  - 可导且导函数也已知：牛顿法，二次收敛
  - 插值(Interpolation)：给定一些不完整的观察值，想要推理出全景图像



牛顿法

给定f可导且导函数f'也已知，求x使得$f(x) = 0$

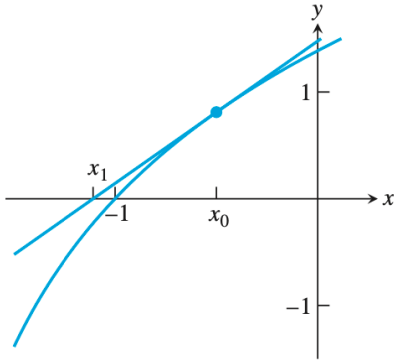


Figure 1.8 One step of Newton's Method. Starting with x_0, the tangent line to the curve $y = f(x)$ is drawn. The intersection point with the x-axis is x_1, the next approximation to the root.

牛顿方法：

x_0:*初始估计*

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, k = 0,1,2 \ldots$$

牛顿法的二次收敛 (Quadratic convergence)

记$e_i = |x_i - r|$为i次迭代的误差

- **局部收敛**：在某个足够小的邻域里面，任意的初始值都会收敛
- **二次收敛**：如果$M = \lim_{i\to\infty} \frac{e_{i+1}}{e_i^2} < \infty$，则称该迭代方法二次收敛。

定理. 如果f二次可导且$f'(r) \neq 0$，其中r满足$f(r) = 0$，那么牛顿法在局部二次收敛到r，且

$$\lim_{i\to\infty} \frac{e_{i+1}}{e_i^2} = \left|\frac{f''(r)}{2f'(r)}\right|$$

牛顿法的二次收敛分析：不动点方程

定理. 如果f二次连续可导且$f'(r) \neq 0$，其中r满足$f(r) = 0$，那么牛顿法在局部二次收敛到r，且

$$\lim_{i\to\infty} \frac{e_{i+1}}{e_i^2} = \left|\frac{f''(r)}{2f'(r)}\right|$$

证明： 首先注意到牛顿法等价于以下不动点方程$x_{k+1} = g(x_k)$，其中$g(x) = x - \frac{f(x)}{f'(x)}$. g的不动点对应于f的根。

$$g'(x) = 1 - \frac{f'(x)^2 - f(x)f''(x)}{f'(x)^2} = \frac{f(x)f''(x)}{f'(x)^2}.$$

在g的不动点处，有$g'(r) = 0$. 因此牛顿法是局部收敛的。

牛顿法的二次收敛分析

定理. 如果f二次可导且$f'(r) \neq 0$，其中r满足$f(r) = 0$，那么牛顿法在局部二次收敛到r，且

$$\lim_{i\to\infty} \frac{e_{i+1}}{e_i^2} = \left|\frac{f''(r)}{2f'(r)}\right|$$

证明： 考虑第i次迭代，在r处进行Taylor展开有：

$$f(r) = f(x_i) + (r - x_i)f'(x_i) + \frac{(r - x_i)^2}{2} f''(\xi_i), \text{for some } \xi_i$$

因为r满足$f(r) = 0$，化简得到

$$x_i - \frac{f(x_i)}{f'(x_i)} - r = \frac{(r - x_i)^2}{2} \frac{f''(\xi_i)}{f'(x_i)}$$

回顾牛顿法的迭代方程：

$$e_{i+1} = |x_{i+1} - r| = e_i^2 \left|\frac{f''(\xi_i)}{2f'(x_i)}\right|$$

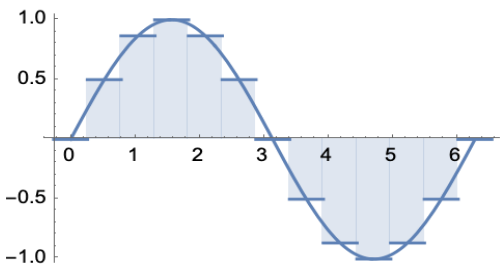
由局部收敛可知，只要足够的接近r， 则有$x_i \to r$且$\xi_i \to r$。由f''和f'的连续性可知

$$\lim_{i\to\infty} \frac{e_{i+1}}{e_i^2} = \lim_{i\to\infty} \left|\frac{f''(\xi_i)}{2f'(x_i)}\right| = \left|\frac{f''(r)}{2f'(r)}\right|$$

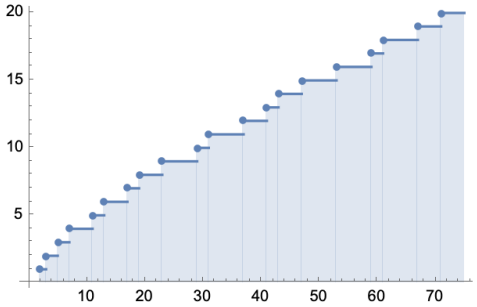
牛顿法的变种

- 牛顿法只用到了一阶导数的信息
  - 注：二次收敛的分析也可以在只假设f'是C-Lipschitz的情况下进行
  - 如果使用更高阶的导数呢？例如，并不只是考虑切线，而是在局部做一个二次曲线的近似
- 牛顿法更容易推广到高维，可以解决多变量的求根问题/最优化问题
- 然而导数可能并不容易求解：割线法 (secant method)，quasi-newton method
  - Secant method: $x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}, k = 1,2, \ldots$
  - Convergence rate of Secant method: $\frac{1+\sqrt{5}}{2} \approx 1.618$
  - quasi-newton method非常常见的一个实现是Broyden-Fletcher-Goldfarb-Shanno (BFGS)算法
- 现实中，往往会结合二分法和牛顿/割线法（如Dekker/Brent方法）
  - 二分法可以保证全局收敛
  - 牛顿/割线法可以保证在足够接近解的时候，更快速地收敛到需要的精度

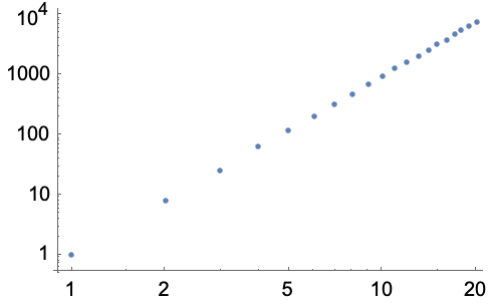
插值是为了什么？



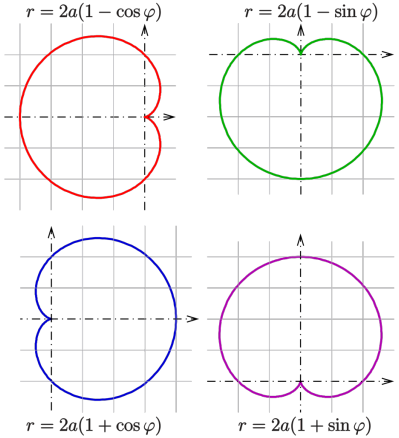
$\sin(x)$



素数的个数



Log-Log plot of $f(x) = x^3$



心形线(Cardioid)

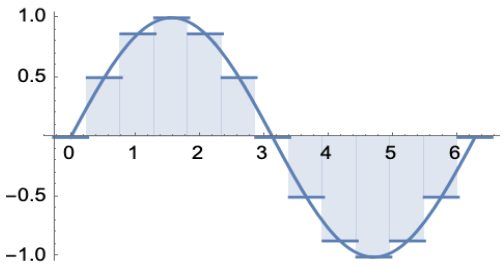
Source:Wikipedia CC-BY-SA 4.0

旋转照片

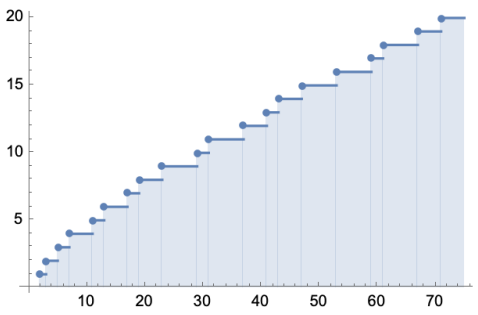
旋转360度后照片一样吗？

$$y' = \ln f(x) = 3 \ln x$$
$$x' = \ln x$$
$$y' = 3x'$$

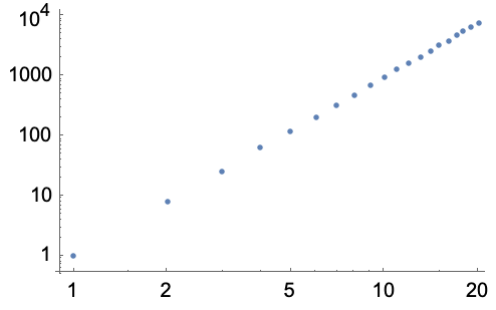
插值是为了什么？



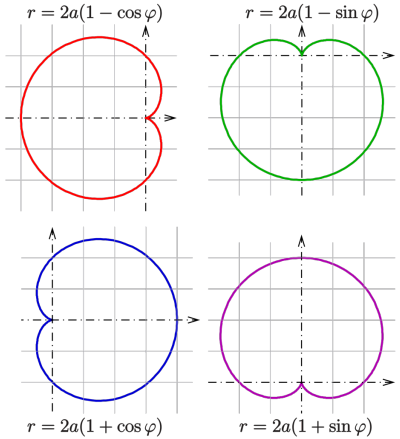
$\sin(x)$



素数的个数



Log-Log plot of $f(x) = x^3$



心形线(Cardioid)

Source:Wikipedia CC-BY-SA 4.0

旋转照片

旋转360度后照片一样吗？

$$y' = \ln f(x) = 3 \ln x$$
$$x' = \ln x$$
$$y' = 3x'$$

为什么想要连续性？好看？

插值的连续性

- 回顾一下不连续函数的**数值稳定性**：
- 给定函数f
- 输入： $x + \Delta x$
- 计算：$f(x)$

$$f(x + \Delta x) - f(x) \approx \Delta x \, f'(x)$$

在函数不连续性的点上，数值可能非常不稳定！

更进一步的处理，往往需要更高阶的导数

用什么插值？

- 多项式定义：

$$p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \cdots + a_n x^n$$

- $a_0, a_1, \ldots, a_n$称为多项式的系数
- 若$a_n \neq 0$, 则称n为多项式的次数
- 常用表达形式（代数基本定理）：

$$p(x) = a_n (x - r_1)(x - r_2) \ldots (x - r_n)$$

- 一般来说$r_1, r_2, \ldots, r_n$可能是复数，复数根总是成对出现的

- 为什么使用多项式？如果使用物理电路插值呢？神经网络呢？
  - 回顾Taylor展开，也是多项式
  - Taylor展开一般只关注一个点上的近似多项式
  - Weierstrass 定理: 对于连续函数，多项式可以任意地逼近。

Lagrange Interpolation

拉格朗日插值

- 任意给定n个不同的数据点$(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- 满足$p(x_i) = y_i, \forall i$的，次数为$n-1$次（或更低次）多项式有且仅有一个
- 存在性的构造证明：
  - 固定k，试找出$L_k(x_k) = 1, L_k(x_j) = 0, \forall j \neq k$
    $L_k(x) = A(x - x_1) \ldots (x - x_{k-1})(x - x_{k+1}) \ldots (x - x_n)$
  - 注意到$L_k(x_j) = 0, \forall j \neq k$
  - 而$L_k(x_k) = A(x_k - x_1) \ldots (x_k - x_{k-1})(x_k - x_{k+1}) \ldots (x_k - x_n)$
  - 可令$A = \frac{1}{(x_k - x_1) \ldots (x_k - x_{k-1})(x_k - x_{k+1}) \ldots (x_k - x_n)}$

Lagrange Interpolation

存在性的构造证明：

- 对任意k，找出$L_k(x_k) = 1, L_k(x_j) = 0, \forall j \neq k$

$$L_k(x) = A(x - x_1) \dots (x - x_{k-1})(x - x_{k+1}) \dots (x - x_n)$$

- 把它们加起来：

$$P_{n-1}(x) = y_1 L_1(x) + \cdots + y_n L_n(x)$$

- 则 $P_{n-1}(x_k) = 0 + \cdots + 0 + y_k L_k(x_k) + 0 + \cdots + 0 = y_k$
- **注意多项式的次数最多为n-1次**
- **思考**：是否可能小于n-1次？

- **练习**：尝试用Lagrange interpolation找出通过$(1, 1), (2,1)$这两点的多项式，并化简

Lagrange Interpolation

唯一性证明：

- 反证法，假设有$P(x)$和$Q(x)$两个多项式，至多$n-1$次且都通过给定的n个点
- 考虑$H(x)=P(x)-Q(x)$，H的次数最多为$n-1$次
- $H(x_1)=H(x_2)=H(x_3)=\cdots=H(x_n)=0$
- H有n个零点
- 代数基本定理说，$n-1$次多项式有且仅有$n-1$个零点，除非H是零多项式
- H必须恒等于零
- $P(x)$恒等于$Q(x)$

- 所以这样的多项式只有一个

Lagrange Interpolation

拉格朗日插值：

- 对任意k，找出$L_k(x_k) = 1, L_k(x_j) = 0, \forall j \neq k$

$$L_k(x) = A_k(x - x_1) \dots (x - x_{k-1})(x - x_{k+1}) \dots (x - x_n)$$

$$令 A_k = \frac{1}{(x_k - x_1) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)}$$

- 把它们加起来：

$$P_{n-1}(x) = y_1 L_1(x) + \cdots + y_n L_n(x)$$

- **思考**：计算复杂性？
  - 每给定一个新的x,需要多少步算术运算？
  - 可以加速至O(n):
    - 通过Newton basis（牛顿差商形式）
    - 或者Barycentric Lagrange interpolation（利用上除法）

Lagrange Interpolation Error Analysis

拉格朗日插值的误差分析

- 误差公式：对连续n次可导函数f，在$x_1, \ldots, x_n$上插值

$$f(x) - P(x) = \frac{(x - x_1)(x - x_2)\cdots(x - x_n)}{n!} f^{(n)}(c),$$

- 与Taylor展开的余项相似，但是结论更强
- 证明：不断地在newton basis下面应用罗尔定理：一阶可导函数的两个零点之间，存在一个导数为零的点。详见书本

龙格现象

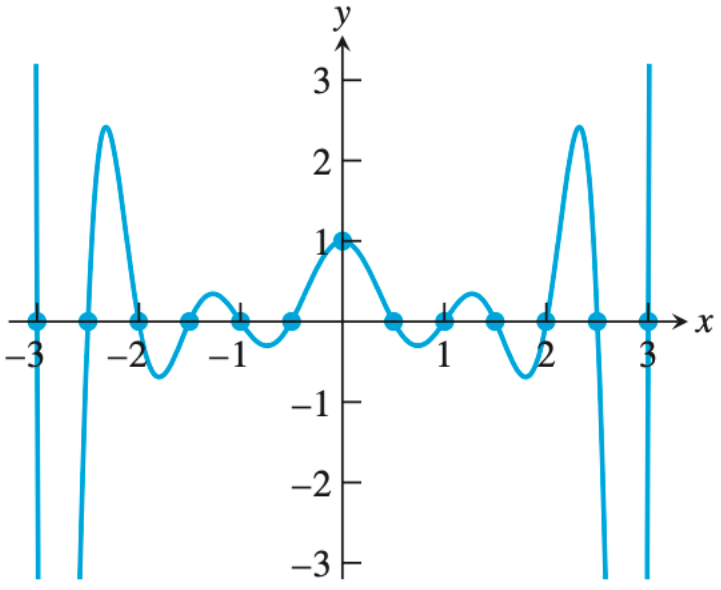


Figure 3.5 Interpolation of triangular bump function. The interpolating polynomial wiggles much more than the input data points.

高次多项式插值固然可以让得到的函数更加smooth，但是也更容易出现龙格现象

插值的应用 I：分享秘密

多项式插值的原理看似简单，但是简单的原理也可以有非凡的应用

- 11个科学家参与了一个高度保密的项目。科学家们希望把档案锁起来，保密的要求是：当且仅当其中6位或者更多的科学家在场的时候，才能打开所有的锁（进而读取档案）。
- 问题：最少需要多少把锁？每个科学家最少需要同时携带多少把钥匙？

- 这个组合问题的答案是：**462**把锁，每个科学家需要带**252**把钥匙！
- 试想，如果有更多的科学家呢？

- Adi Shamir在1979年提出了一个全新的解法：使用多项式插值！

插值的应用 I：分享秘密

多项式插值的原理看似简单，但是简单的原理也可以有非凡的应用

- 11个科学家参与了一个高度保密的项目。科学家们希望把档案锁起来，保密的要求是：当且仅其中6位或者更多的科学家在场的时候，才能打开所有的锁（进而读取档案）。
- 问题：最少需要多少把锁？每个科学家最少需要同时携带多少把钥匙？

- Adi Shamir在1979年提出了全新的解法：使用多项式插值！
  - **把密码**/密文藏进一个5次多项式p里面
  - **每个人手握多项式的一个点值**$(x_i, p(x_i))$，点值互不相同
  - **任意**6个人（或更多），就有了6个数据点
  - 通过拉格朗日插值，即可唯一地还原多项式p

注：实际的算法使用的是有限域而非实数域；
需要确保任意5个人的点值，在信息论意义下面，跟一个点值都没有是一样的。

Polynomials over Finite field（选讲）

$$p(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + \cdots + a_k x^k \ (mod\ q)$$
$$p: \{0, \ldots, q-1\} \to \{0, \ldots, q-1\}$$
$$a_i \in \{0, \ldots, q-1\}$$

- Coefficient representation
- Value representation: $p(1), p(2), \ldots, p(k), p(k+1)$
  - Interpolation: Lagrange, or as a system of linear equations

$$L(x) := \sum_{j=0}^{k} y_j \prod_{0 \le m \le k, m \ne j} \frac{x - x_m}{x_j - x_m}.$$

  - There is a unique degree $\le k$ polynomial passing through $k+1$ points
  - Division in modular arithmetic: multiplicative inverses

Shamir's secret sharing scheme（选讲）

For illustration, let's take n=10, k=2

Want: divide a secret among 10 people, so that any 2 people can recover the secret, but any one person knows nothing about the secret

Shamir's scheme:

Say secret $s \in \{0, \dots, q-1\}$

Sample another number $a \in \{0, \dots, q-1\}$, u.a.r.

Consider the polynomial $p(x) = ax + s \ (mod\ q)$

Hand out $p(1), p(2), \dots, p(10)$

Correctness: knowing two points determine a unique line (e.g. by interpolation)

Secrecy: knowing say $p(1)$ tells you **nothing** about $p(0)$

- If $a \in \{0, \dots, q-1\}$ is drawn u.a.r., then $a + s \ (mod\ q)$ is independent from s

Shamir's secret sharing scheme

Shamir's scheme for general n,k:

Say secret $s \in \{0, \dots, q-1\}$

Sample $a_1, a_2, \dots, a_{k-1}$, u.a.r.

Consider $p(x) = a_{k-1}x^{k-1} + \cdots + a_1 x + s \ (mod\ q)$

Hand out $p(1), p(2), \dots, p(n)$

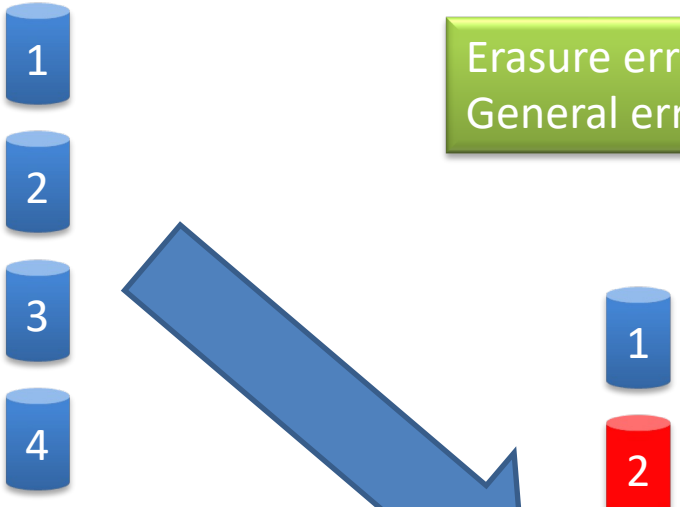
Note that $q > \max(k, n)$

Correctness: Any k knows the secret

Secrecy: Any k-1 knows nothing about the secret

- Any k-1 points, together with any choice of $p(0)$ determine a unique polynomial
- To prove independence: use bijection!

Error correcting code I: Erasure codes



Erasure errors: packet loss
General errors: packet corruption

TCP/IP: resend packets, change data rate etc.
But this can be wasteful!
Can we reconstruct packets?

Another example: RAID storage levels

- Imagine you have 4 disks, each of 4TB
- To tolerate 1 disk failure, a common practice is to use RAID 5
- You have 3 disks of available storage, and 1 disk for parity

- Say you lose B, you can rebuild $B = D \oplus A \oplus C$

- However, at current unrecoverable read error (URE) rate of 1e-14, the success rate of rebuilding is less than 40%
  - RAID 6: One common implementation is via Reed-Solomon codes
- Cloud storage providers are already using more general ECCs

Erasure codes

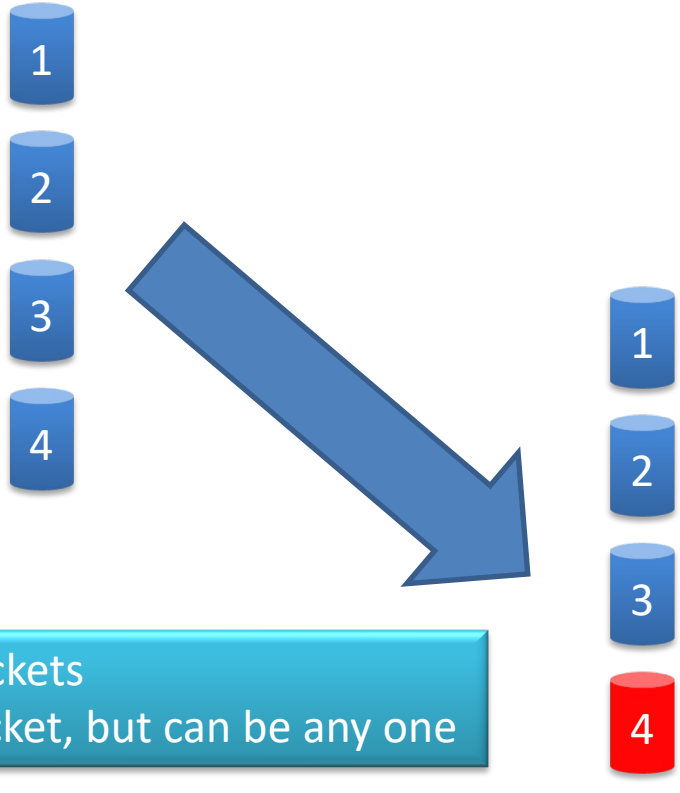
Imagine you want to send n packets through a lossy channel
Lossy channel: can lose up to k packets
Question: can you send n+k packets and recover the message?
Hint: what do we know about polynomials?

Answer: Yes!
Any n points uniquely determine a degree $\leq n-1$ polynomial.

Say we are given messages $a_0, a_2, \dots, a_{n-1}$
Consider $p(x) = a_{n-1}x^{n-1} + \cdots + a_1 x + a_0 \ (mod\ q)$
Send $p(1), p(2), \dots, p(n+k)$

Any n packets allow us to recover all. This is optimal!

Error correcting code II: General errors



Imagine we want to send 2 packets
Channel: corrupt at most 1 packet, but can be any one

Reed-Solomon codes

Imagine we want to send 2 packets
Channel: corrupts at most 1 packet, but can be any one

2 points -> a line?
What if we send other points on the line as well?

Say we send 4 points in total.
How do we determine which one is corrupted?

At most 1 error -> there is a line passing through 3 points
If we found such a line, it must be the correct line:

- At most 1 error, so 2 out of the 3 points must be correct
- 2 points determine a unique line, which must be the correct line

Reed-Solomon codes

Imagine we want to send n packets
Channel: corrupts at most k packet, but can be any one

Reed-Solomon codes
Say we are given messages $a_0, a_2, \dots, a_{n-1}$
Consider $p(x) = a_{n-1}x^{n-1} + \cdots + a_1 x + a_0 \ (mod\ q)$
Send $p(1), p(2), \dots, p(n+2k)$

At most k errors -> ∃ a poly. of deg n-1 passing through n+k points
If we found such a poly., it must be the correct poly.:

- At most k error, so n out of the n+k points must be correct
- There is a unique poly. of deg n-1 going through n points
- So, if our poly. goes through them, it's the correct one

Reed-Solomon codes

Many applications:

- HDFS-RAID in Facebook
- GFS II in Google
- Covid-19 Pool testing
- …

Q: Why mod q?

- Numerical instability or genuine error

Q: Choice of q? Need $q > n + 2k$.

- Not ideal, but OK in practice.
- Often times, errors are bursty. If a bit is corrupted, all nearby symbols are often unreliable

Efficient decoding:

- $\binom{n+2k}{n}$ interpolations to try? Brute-force would take exponential time in general.
- Berlekamp-Welch: one polynomial interpolation suffices – error locating polynomial

ECCs and distance properties

Hamming distance $d(s,t) := \sum_i 1(s_i \neq t_i)$

Minimum distance of a code:= $\min_{m \neq m'} d(c(m), c(m'))$

To handle k erasure errors, need minimum distance k+1

To handle k general errors, need minimum distance > 2k

Theorem: The Reed-Solomon code mapping n messages to a codeword of n+2k messages has minimum distance 2k+1

下节课

Chebyshev多项式插值

函数逼近与正交多项式

最小二乘法与最佳平方逼近