



计算方法

刘景铨

计算机软件新技术国家重点实验室
南京大学

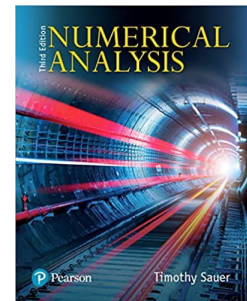


课程基本信息

- 教师：刘景铎
- Email: liu@nju.edu.cn
- Office hour: 周二 15:00-17:00?, 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
- 第一次作业已经发布
- 作业Email: nm_nju_2024@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
- 迟交作业需要提前预约

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



计算方法 nju 2024

群号: 855212527



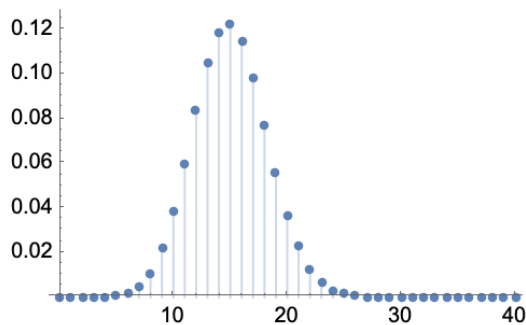
扫一扫二维码, 加入群聊。





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 $f(x) = 0$

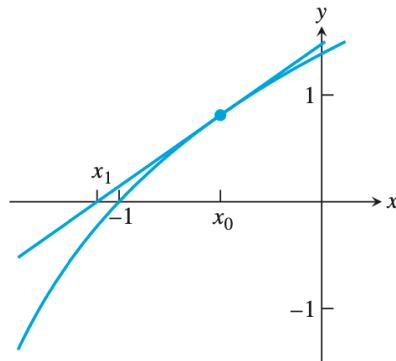


Figure 1.8 One step of Newton's Method. Starting with x_0 , the tangent line to the curve $y=f(x)$ is drawn. The intersection point with the x -axis is x_1 , the next approximation to the root.

牛顿方法:

x_0 :初始估计

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



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 $M = \lim_{i \rightarrow \infty} \frac{e_{i+1}}{e_i^2} < \infty$, 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

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



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

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

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

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

证明: 考虑第 i 次迭代, 在 r 处进行Taylor展开有:

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

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 $x_i \rightarrow r$ 且 $\xi_i \rightarrow r$ 。由 f'' 和 f' 的连续性可知

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

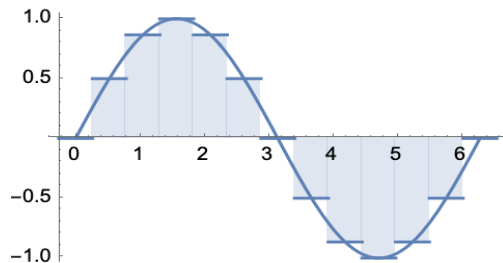


课外阅读：牛顿法的变种

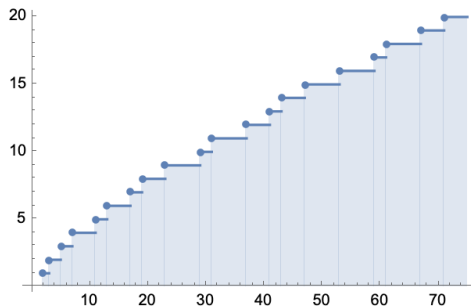
- 牛顿法只用到了二阶导数的信息
 - 注：二次收敛的分析也可以在只假设 f' 是C-Lipschitz的情况下进行
 - 如果使用更高阶的导数呢？例如，并不只是考虑切线，而是在局部做一个二次曲线的近似
- 牛顿法更容易推广到高维，可以解决多变量的求根问题/最优化问题
- 然而导数可能并不容易求解：割线法 (secant method), quasi-newton method
 - Secant method:
$$x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}, k = 1, 2, \dots$$
 - Convergence 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



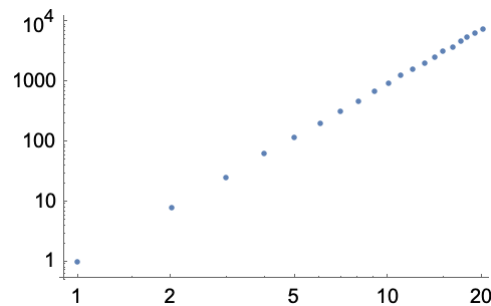
插值是为了什么?



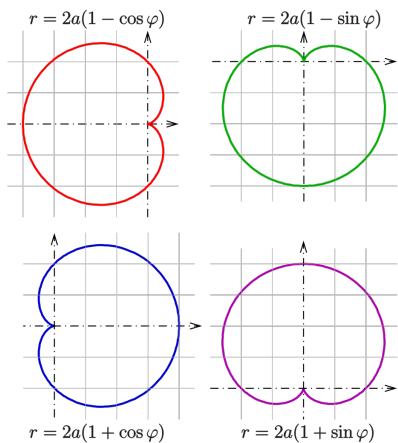
$\sin(x)$



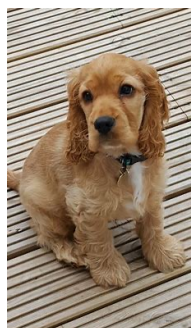
素数的个数



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



心形线(Cardiod)



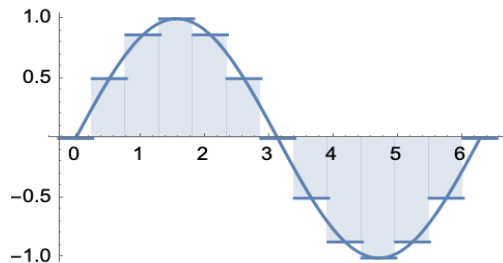
旋转照片

旋转360度后照片一样吗?

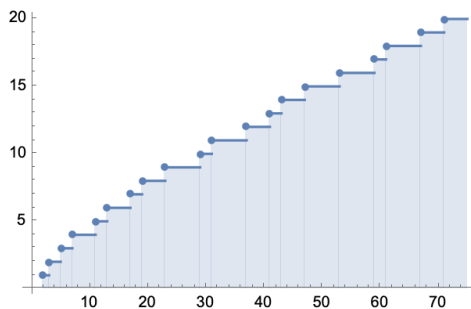
$$y' = \ln f(x) = 3 \ln x$$
$$x' = \ln x$$
$$y' = 3x'$$



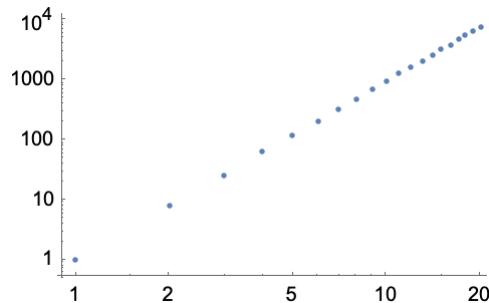
插值是为了什么?



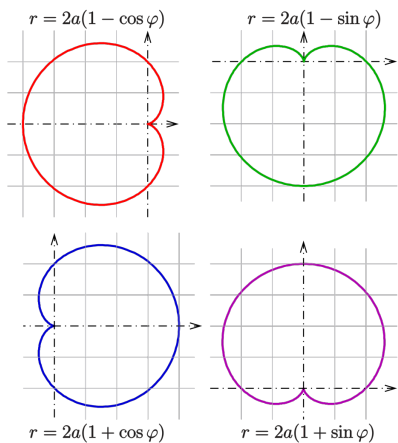
sin(x)



素数的个数



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



心形线(Cardiod)



为什么想要连续性? 好看?

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：

$$p(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \dots + a_nx^n$$

- a_0, a_1, \dots, 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）：

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

- 一般来说 r_1, r_2, \dots, 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
 - 课外阅读：Stone-Weierstrass 定理和神经网络的Universal approximation



Lagrange Interpolation 拉格朗日插值

- 任意给定 n 个不同的数据点 $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$
- 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 $L_k(x_k) = 1, L_k(x_j) = 0, \forall j \neq k$
$$L_k(x) = A(x - x_1) \dots (x - x_{k-1})(x - x_{k+1}) \dots (x - x_n)$$
 - 注意到 $L_k(x_j) = 0, \forall j \neq k$
 - 而 $L_k(x_k) = A(x_k - x_1) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)$
 - 可令 $A = \frac{1}{(x_k - x_1) \dots (x_k - x_{k-1})(x_k - x_{k+1}) \dots (x_k - x_n)}$



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:

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

- 则 $P_{n-1}(x_k) = 0 + \dots + 0 + y_k L_k(x_k) + 0 + \dots + 0 = y_k$
- 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
 - $H(x_1) = H(x_2) = H(x_3) = \dots = H(x_n) = 0$
 - 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 $L_k(x_k) = 1, L_k(x_j) = 0, \forall j \neq k$

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

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

- 把它们加起来:

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

- **思考:** 计算复杂性?

- 每给定一个新的 x , 需要多少步算术运算?
- 每个 $L_k(x)$ 需要 $O(n)$, 共 $O(n)$ 个 $L_k(x)$, 所以需要 $O(n^2)$ 次
- 课外阅读: 加速至 $O(n)$ 次运算:
 - 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, \dots, 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

证明思路：

- 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
- 详见书本第三章



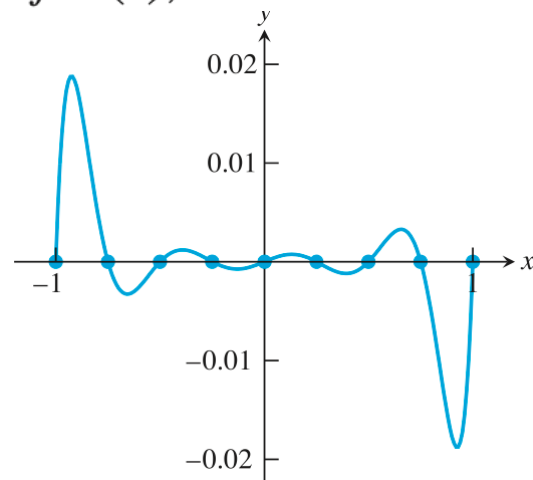
Lagrange Interpolation Error Analysis (选讲)

对连续 n 次可导函数 f ，在 x_1, \dots, x_n 上拉格朗日插值的误差

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

证明：令 $q(t)$ 为 x_1, \dots, x_n, x 上插值的多项式

- 则 $q(t) = P(t) + \lambda \prod_{j=1}^n (t - x_j)$ ，其中 $\lambda = \frac{f(x) - P(x)}{\prod_{j=1}^n (x - x_j)}$
- 考虑 $\phi(t) := f(t) - q(t)$ ，则 $\phi(t)$ 在 $n + 1$ 个点处为零
- 由Rolle定理， $\phi'(t)$ 在 n 个点处为零
- 反复应用Rolle定理， $\phi^{(n)}(t)$ 在区间上某个点为零
- 在该点 $\phi^{(n)}(c) = 0 \Rightarrow f^{(n)}(c) = q^{(n)}(c) = \lambda n!$
- 整理即得误差公式



使用等距采样插值时的误差函数

如何选点 x_1, \dots, x_n ，才能最小化误差？插值的表现可否媲美函数逼近？

- 采用更多靠近边界的点



龙格现象

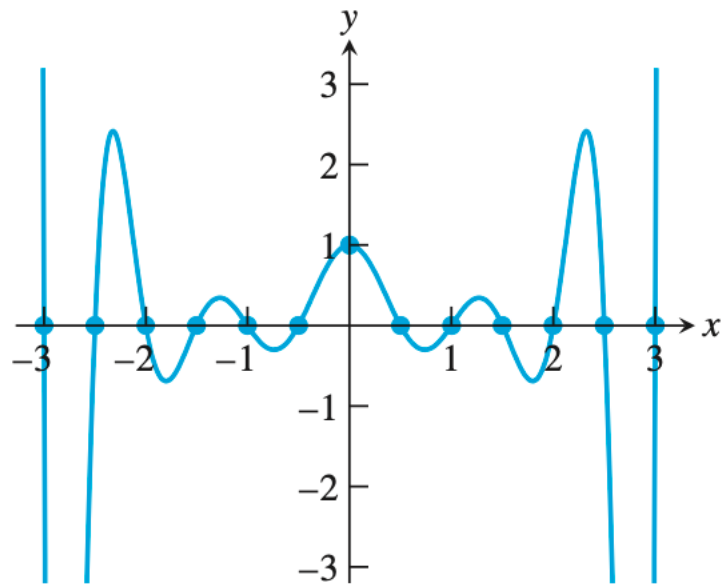


Figure 3.5 Interpolation of triangular bump function. The interpolating polynomial wiggles much more than the input data points.

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？

组合问题答案： $\binom{11}{5} = 462$ 把锁，每个科学家需要带 $\binom{10}{5} = 252$ 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。



有限域多项式

$$p(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots + a_kx^k \pmod{q}$$

$$p: \{0, \dots, q-1\} \rightarrow \{0, \dots, q-1\}$$

$$a_i \in \{0, \dots, q-1\}$$

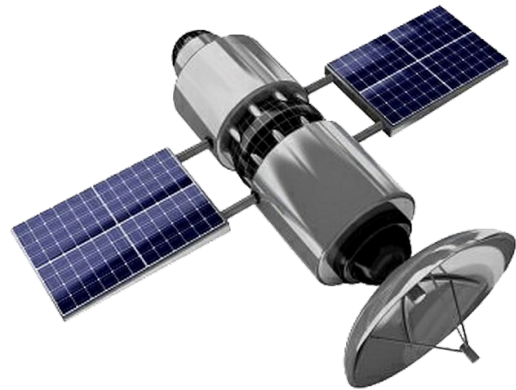
- 系数表示
- 取值表示: $p(1), p(2), \dots, p(k), p(k+1)$
 - 插值: Lagrange, 或解线性方程组

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

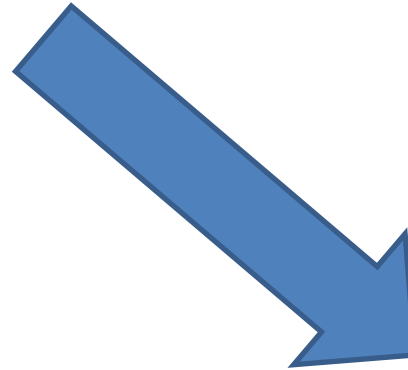
- 经过 $k+1$ 个点的次数 $\leq k$ 的多项式有且只有一个
- 取模下的除法: 乘法的逆



Error correcting code I: Erasure codes



[This Photo](#) by Unknown Author is licensed under [CC BY-NC](#)



Erasure errors: packet loss
General errors: packet corruption



[This Photo](#) by Unknown Author is licensed under [CC BY-SA](#)

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_1, \dots, a_{n-1}

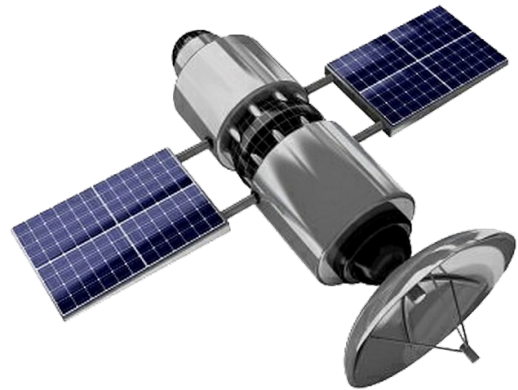
Consider $p(x) = a_{n-1}x^{n-1} + \dots + a_1x + a_0 \pmod{q}$

Send $p(1), p(2), \dots, p(n+k)$

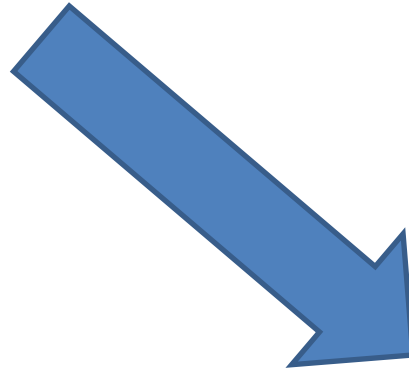
Any n packets allow us to recover all. This is optimal!



Error correcting code II: General errors



This Photo by Unknown Author is licensed under CC BY-NC



This Photo by Unknown Author is licensed under CC BY-SA

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 \rightarrow 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 \rightarrow 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_1, \dots, a_{n-1}

Consider $p(x) = a_{n-1}x^{n-1} + \dots + a_1x + a_0 \pmod{q}$

Send $p(1), p(2), \dots, p(n + 2k)$

At most k errors $\rightarrow \exists$ 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:

- Voyager I for colored photos
- 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



课外阅读: Berlekamp-Welch算法

Sent: $p(1), p(2), \dots, p(n), p(n+1), \dots, p(n+2k)$

Received: $r_1, r_2, \dots, r_n, r_{n+1}, \dots, r_{n+2k}$

Berlekamp Welch Theorem

There is a polynomial $Q(x)$ of deg. $n+k-1$, $E(x)$ of deg. k , such that

$p(x) = \frac{Q(x)}{E(x)}$, and $\forall i, Q(i) = r_i E(i)$. In other words,

$$\forall i, p(i)E(i) = r_i E(i)$$

To understand this, let $Q(x) = b_{n+k-1}x^{n+k-1} + \dots + b_0$, and $E(x) = x^k + e_{k-1}x^{k-1} + \dots + e_0$.

Total number of unknowns = $n+2k$.

For each received value r_i , there is a linear equation $Q(i) = r_i E(i)$.

Total number of linear equations = $n+2k$.



课外阅读: Error locating polynomial

Sent: $p(1), p(2), \dots, p(n), p(n+1), \dots, p(n+2k)$

Received: $r_1, r_2, \dots, r_n, r_{n+1}, \dots, r_{n+2k}$

Say the location of errors are f_1, \dots, f_k

Consider the error locating polynomial $E(x) = (x - f_1) \dots (x - f_k)$

Ideally $p(i) = r_i$

Instead we ask that $p(i)E(i) = r_iE(i)$

Case 1: location i is an error, $0=0$

Case 2: location i is correct, $p(i) = r_i$

Let $Q(x) = p(x)E(x)$. Then, the system of linear equations have a solution! (Consistency)

Uniqueness of $p(x)$: Say $Q(x), E(x)$ is one solution, and $Q'(x), E'(x)$ is another solution.

Notice that $Q(x)E'(x) = Q'(x)E(x)$, because they agree on $n+2k$ points, and they are of degree at most $n+2k$.

So, $p(x) = \frac{Q(x)}{E(x)} = \frac{Q'(x)}{E'(x)}$ is unique.

Note that $Q(x), E(x)$ may not be unique themselves. Consider the case where there are no errors.



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 $2k + 1$



Distance properties

Claim 1: The Reed-Solomon code mapping n messages to a codeword of $n + 2k$ messages has minimum distance **at most** $2k + 1$

It suffices to show that, $\exists m \neq m', d(c(m), c(m')) \leq 2k + 1$

Say we choose codeword $c_0, c_1, \dots, c_{n-2}, c_{n-1}$, we interpolate to get $p_1(x)$

Choose another codeword $c_0, c_1, \dots, c_{n-2}, e_{n-1}$, we interpolate to get $p_2(x)$

$p_1(x)$ agrees with $p_2(x)$ on at least $n - 1$ points

$p_1(x)$ and $p_2(x)$ encode messages $m \neq m'$ in their coefficients

Reed-Solomon code will send $c(m), c(m')$ that agrees on at least $n - 1$ points

Therefore, $d(c(m), c(m')) \leq 2k + 1$

Side note: this is essentially the Singleton/Projection bound.



Distance properties

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

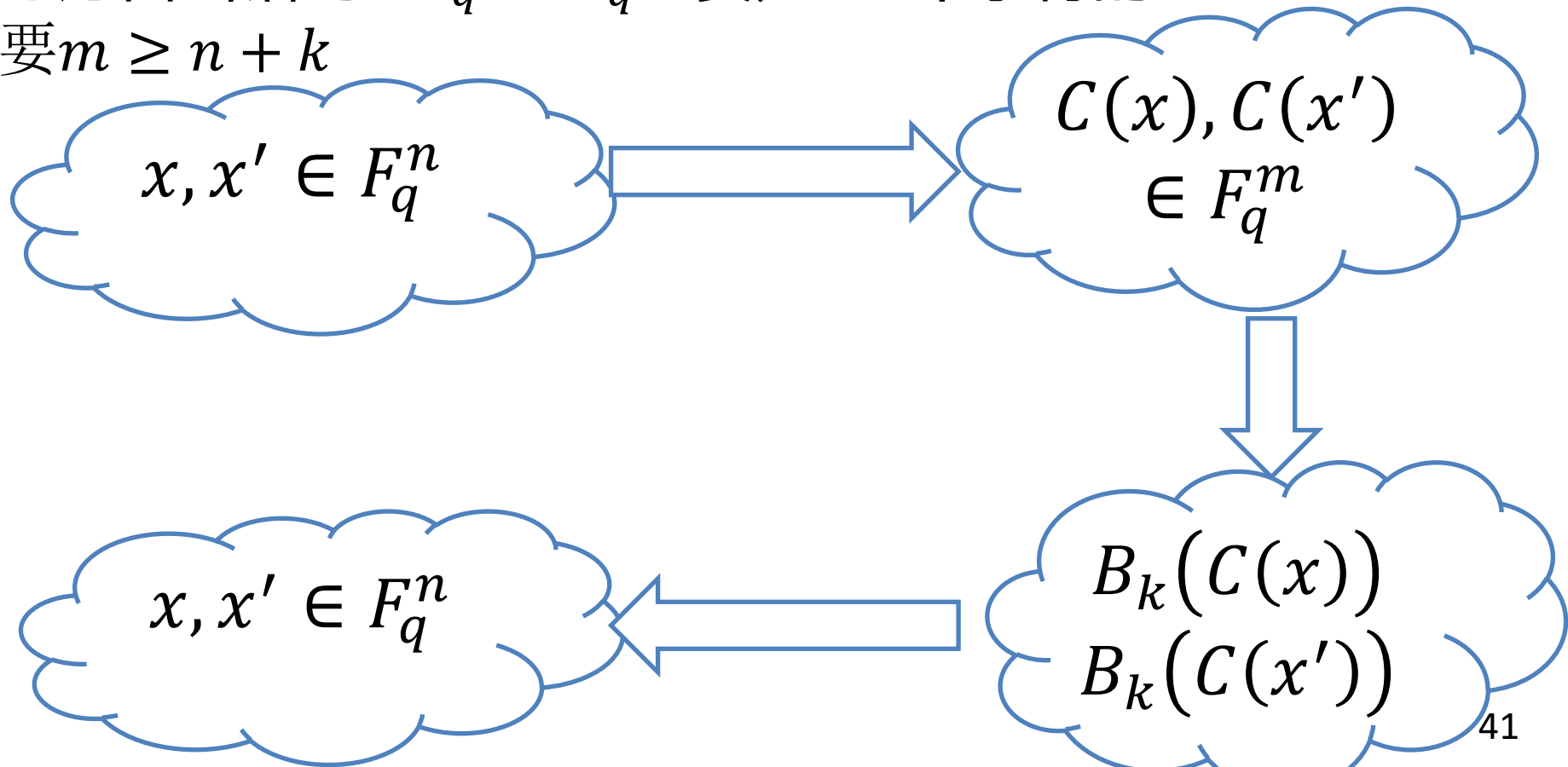
This essentially follows from decoding. If the minimum distance is smaller, then decoding will fail.

See the following slides on Hamming bound.



Hamming bound

考虑自纠错码 $C: F_q^n \rightarrow F_q^m$. 要处理 k 个字符的 erasure, 那么需要 $m \geq n + k$





Hamming bound

考虑自纠错码 $C: F_q^n \rightarrow F_q^m$. 要处理 k 个字符的 erasure, 那么需要 $m \geq n + k$

反证法: 假设 $m \leq n + k - 1$

- 考虑所有的 codeword 的集合: $\{C(x): x \in F_q^n\}$
- 去掉这些 codeword 的后面 k 位 (投影到前面 $n - 1$ 位)
- 剩下每个 codeword 的长度最多为 $n - 1$
- 由 Pigeon-hole principle, 一定存在两个 codeword 在前 $n - 1$ 位是相同的, 记为 x, x'
- 这意味着, 存在 $x, x' \in F_q^n$, $x \neq x'$, 使得 $C(x)$ 和 $C(x')$ 在去掉了最后的 k 位后相等
- 这与能处理 k 个字符的 erasure 的假设矛盾。



Hamming bound

考虑自纠错码 $C: F_q^n \rightarrow F_q^m$. 要处理 k 个字符的 corruption, 那么需要 $m \geq n + 2k$

反证法: 假设 $m \leq n + 2k - 1$

- 考虑所有的 codeword 的集合: $\{C(x): x \in F_q^n\}$
- 去掉这些 codeword 的后面 $2k$ 位 (投影到前面 $n - 1$ 位)
- 剩下每个 codeword 的长度最多为 $n - 1$
- 由 Pigeon-hole principle, 一定存在两个 codeword 在前 $n - 1$ 位是相同的, 记为 x, x'
- 这意味着, 存在 $x, x' \in F_q^n, x \neq x'$, 使得 $C(x)$ 和 $C(x')$ 在去掉了最后的 $2k$ 位后相等
- 因此, 可以从 $C(x)$ 修改最多 k 个字符, 再从 $C(x')$ 修改最多 k 个字符, 将得到一样的信息
- 从接收者的角度看, 将无法区分这是由 $C(x)$ 修改最多 k 个字符得到的, 还是从 $C(x')$ 修改最多 k 个字符得到的
- 这与能处理 k 个字符的 corruption 矛盾



Polynomial interpolation and Boosting (选讲)

If you can compute a polynomial on sufficiently many points, with decent amount of confidence

And the polynomial has a known degree bound

- Then, you can compute the polynomial correctly, everywhere, by interpolation!
- Say you want to compute some function that can be “approximated”, or “represented” by a polynomial
- And someone promise to help by correctly computing an answer most of the time
- Then you can just encode the function using polynomials (or ECCs to be precise)
- Ask for enough answers
- And decode the function that you really wanted to compute

课外阅读: [random self-reducibility of the PERMANENT](#)



函数的一致性逼近 (选讲)

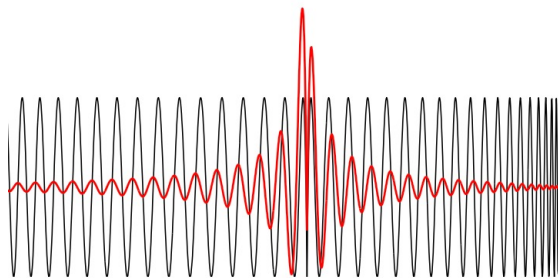
如果不要多项式恰好经过每一个点，只要求尽可能地近似呢？

函数 f 的 ∞ -范数 (∞ -norm):

$$\|f\|_{\infty} := \sup |f|$$

最佳一致逼近: 找出多项式 p ，使得 $\|f - p\|_{\infty}$ 为最小

回顾Weierstrass 定理的内容: 对于连续函数，多项式可以任意地逼近





一些概率论的回顾 (选讲)

期望(Expectation)

$$E[X] = \sum_x x \cdot \Pr[X = x]$$

方差(Variance)

$$\text{Var}[X] = E[(X - E[X])^2]$$

Chebyshev不等式

$$\Pr[|X - E[x]| \geq k] \leq \frac{\text{Var}[X]}{k^2}$$



Bernstein Polynomial (选讲)

给定：连续函数 $f: [0,1] \rightarrow R$

目标：构造多项式 p ，使得在 $x \in [0,1]$ 这一点上， $p(x)$ 尽可能地接近 $f(x)$

设 K 是这样的一个随机变量，定义为 n 次独立的 Bernoulli 试验中成功的次数，每次成功的概率是 x

Bernoulli 试验：成功的概率是 x ，不成功的概率是 $1 - x$

K 服从二项分布 (Binomial distribution)，期望 $E[K] = nx$
方差 $\text{Var}[K] = nx(1 - x)$



Bernstein Polynomial (选讲)

设 K 是这样的一个随机变量，定义为 n 次独立的Bernoulli试验中成功的次数，每次成功的概率是 x ， K 服从二项分布 (Binomial distribution)

- 期望 $E[K] = nx$
- 方差 $\text{Var}[K] = nx(1 - x)$
- $\Pr[K = i] = \binom{n}{i}x^i(1 - x)^{n-i}$

Bernstein多项式的一组基: $b_{n,i}(x) := \binom{n}{i}x^i(1 - x)^{n-i}$

$\lim_{n \rightarrow \infty} \Pr \left[\left| \frac{K}{n} - x \right| > \delta \right] = 0$ 对所有 x 一致地成立

- 注: 可从Chebyshev不等式证明



Bernstein Polynomial (选讲)

- 期望 $E[K] = nx$
- 方差 $\text{Var}[K] = nx(1-x)$
- $\Pr[K = i] = \binom{n}{i} x^i (1-x)^{n-i}$, Bernstein多项式的一组基

$\lim_{n \rightarrow \infty} \Pr \left[\left| \frac{K}{n} - x \right| > \delta \right] = 0$ 对所有 x 一致地成立

对于闭区间 $[0,1]$, f 是连续的意味着 f 也是一致连续的

因此 $\lim_{n \rightarrow \infty} \Pr \left[\left| f\left(\frac{K}{n}\right) - f(x) \right| > \delta \right] = 0$ 对 x 一致地成立

进而 $\lim_{n \rightarrow \infty} E \left[\left| f\left(\frac{K}{n}\right) - f(x) \right| \right] = 0$ 对 x 一致地成立



Bernstein Polynomial (选讲)

- 期望 $E[K] = nx$
- 方差 $\text{Var}[K] = nx(1-x)$
- $\Pr[K = i] = \binom{n}{i} x^i (1-x)^{n-i}$, Bernstein多项式的一组基

$\lim_{n \rightarrow \infty} \Pr \left[\left| \frac{K}{n} - x \right| > \delta \right] = 0$ 对所有 x 一致地成立

对于闭区间 $[0,1]$, f 是连续的意味着 f 也是一致连续的

因此 $\lim_{n \rightarrow \infty} \Pr \left[\left| f\left(\frac{K}{n}\right) - f(x) \right| > \delta \right] = 0$ 对 x 一致地成立

进而 $\lim_{n \rightarrow \infty} E \left[\left| f\left(\frac{K}{n}\right) - f(x) \right| \right] = 0$ 对 x 一致地成立

$$E \left[f\left(\frac{K}{n}\right) \right] = \sum_{i=0}^n f\left(\frac{i}{n}\right) \Pr(K = i) = \sum_{i=0}^n f\left(\frac{i}{n}\right) b_{n,i}(x)$$



Bernstein Polynomial (选讲)

- 期望 $E[K] = nx$
- 方差 $\text{Var}[K] = nx(1-x)$
- $\Pr[K = i] = \binom{n}{i} x^i (1-x)^{n-i}$, Bernstein多项式的一组基

$\lim_{n \rightarrow \infty} \Pr \left[\left| \frac{K}{n} - x \right| > \delta \right] = 0$ 对所有 x 一致地成立因此

$\lim_{n \rightarrow \infty} \Pr \left[\left| f\left(\frac{K}{n}\right) - f(x) \right| > \delta \right] = 0$ 对 x 一致地成立

进而 $\lim_{n \rightarrow \infty} E \left[\left| f\left(\frac{K}{n}\right) - f(x) \right| \right] = 0$ 对 x 一致地成立

$$E \left[f\left(\frac{K}{n}\right) \right] = \sum_{i=0}^n f\left(\frac{i}{n}\right) \Pr(K = i) = \sum_{i=0}^n f\left(\frac{i}{n}\right) b_{n,i}(x)$$

- Bernstein多项式, 一致逼近 f
- 具体的误差亦可以使用Chebyshev不等式得到



Bernstein Polynomial (选讲)

误差的定量分析

- 注意到 $\left| f(x) - \mathbb{E} \left[f \left(\frac{K}{n} \right) \right] \right| \leq \mathbb{E} \left[\left| f(x) - f \left(\frac{K}{n} \right) \right| \right]$
 - 使用了三角不等式 $|a + b| \leq |a| + |b|$

- 后者分开两项

$$\begin{aligned} & \mathbb{E} \left[\left| f(x) - f \left(\frac{K}{n} \right) \right| \right] \\ & \leq 2 \Pr \left[\left| \frac{K}{n} - x \right| > \epsilon \right] \cdot \|f\|_\infty + \Pr \left[\left| \frac{K}{n} - x \right| \leq \epsilon \right] \cdot \sup_{|x-y| \leq \epsilon} |f(x) - f(y)| \end{aligned}$$

回忆: $\mathbb{E}[X] = \sum_x x \cdot \Pr[X = x]$

对 $\Pr \left[\left| \frac{K}{n} - x \right| > \epsilon \right]$ 使用Chebyshev不等式, 即可得到:

- $\forall x \in [0,1], \forall \epsilon > 0,$

$$\left| f(x) - \mathbb{E} \left[f \left(\frac{K}{n} \right) \right] \right| \leq \frac{\|f\|_\infty}{2n\epsilon^2} + \sup_{|x-y| \leq \epsilon} |f(x) - f(y)|$$



Beyond Polynomial (选讲)

Weierstrass 定理: 对于连续函数 f , 多项式可以任意地逼近

如果不是多项式呢?

推广 (Stone-Weierstrass 定理): 考虑连续函数的任意一个子代数(subalgebra) A , A 能任意地逼近连续函数当且仅当 A 能分离点: $\forall x \neq y, \exists p \in A, s.t., p(x) \neq p(y)$

因此更一般的, 如神经网络等, 往往也可以通过Stone-Weierstrass 定理证明其相关的逼近定理。



下节课

Chebyshev多项式与插值

函数逼近与正交多项式

最小二乘法与最佳平方逼近