NANJING UNIVERSITY · 计算机学院

计算方法

Computational Methods

Lecture 1：误差、条件性与牛顿法
Error, Conditioning, and Newton's Method

刘景铖
南京大学 · 计算机学院

课程信息

计算方法 Fall 2026

授课教师 刘景铖 `liu@nju.edu.cn`

上课 周四 5–6 节 · 1–18 周 · 仙 I-319

答疑 周四下午 4 点

助教 周宇恒 · 侯哲

作业 `nm_nju_2026@163.com`

课程主页 TCS Wiki · Fall 2026

QQ 群 1054896504 加入时注明**姓名、专业、学号**

最新公告与完整课程规则以课程主页为准。

作业与合作：可以讨论，但答案必须是自己的

可以做什么？

- 最多 **3 人**组成学习小组，讨论思路与方法；
- 可以查阅教材、论文或在线资料；
- 可以从同学那里获得提示或重要想法。

必须做到什么？

- **独立完成书面解答**，不得照抄；
- 在作业首页注明所有合作者与重要帮助；
- 使用外部资料必须注明来源；
- 不得把自己的解答公开给其他同学看到。

Homework 1　9 月 10 日 23:59 截止　　**迟交申请原则上需要提前提出。**

提交：`nm_nju_2026@163.com`　　文件名：学号 _ 姓名 _A1.pdf

先自己想，再讨论。合作应该增加学习，而不是替代思考。

这一学期，会反复遇到哪些计算问题？

插值、逼近与拟合
从有限观测恢复连续对象：天体轨迹、工业设计、计算机动画。
interpolation · approximation · least squares

线性代数与压缩
大型线性系统、SVD/PCA、低秩近似：数据很多，结构可能很少。
projection · conditioning · low rank

特征值与随机游走
动力系统稳定性、PageRank、局部聚类、MCMC、图上的谱方法。
eigenvalues · power iteration · Markov chains

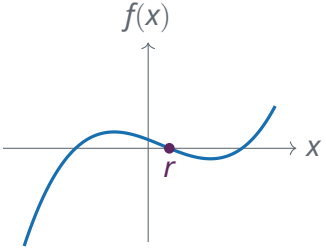
优化与对偶
机器学习、网络优化、线性规划、零和博弈。
gradient · convexity · duality

看起来完全不同，却会反复遇到同样的工具：**近似**、**误差**、**条件性**、**迭代**、**结构**。

第一个模型问题：求根

给定 f，求 r 使得

$$f(r) = 0$$



预言机：先查询 $f(x)$；稍后再允许查询 $f'(x)$。

Accuracy 输出不是无限展开的 r，而是满足某种精度要求的 $\hat{r}$。

Conditioning f 稍微改变，正确的根会移动多少？

Information 只知道符号、知道函数值、再知道导数，会有什么区别？

Cost 达到精度 ε，需要多少次查询、多少次迭代？

今天会看到两种极端： **二分法：全局可靠** **牛顿法：局部极快**。

它们的差别来自**信息、结构和局部几何**，而不只是公式不同。

热身：这段程序会输出什么？

```
double x = 0.3 * 3 + 0.1;
double y = 1.0;

if (x == y)
    cout << "Equal";
else
    cout << "Not equal";
```

先猜一猜。

数学上当然有

$$0.3 \times 3 + 0.1 = 1.$$

请先不要打开 Python。

热身：这段程序会输出什么？

```
double x = 0.3 * 3 + 0.1;
double y = 1.0;

if (x == y)
    cout << "Equal";
else
    cout << "Not equal";
```

Not equal

一种常见的双精度计算结果是

$$x \approx 0.9999999999999999.$$

那以后统一检查 $|x - y| < \varepsilon$ 就够了吗？ 也不够。

同一个 ε 对 10^{-12} 和 10^{12} 的意义完全不同。

绝对误差与相对误差：差一个“尺度”

设真实值为 x，计算结果为 $\hat{x}$。

绝对误差 / Absolute error

$$E_{\text{abs}} = |\hat{x} - x|.$$

直接度量**偏差有多大**，但数值大小依赖问题的单位和尺度。

相对误差 / Relative error

$$E_{\text{rel}} = \frac{|\hat{x} - x|}{|x|}, \qquad x \neq 0.$$

把误差除以答案本身的尺度，更适合比较不同量级的问题。

$x = 10^{-8}$, $\hat{x} = 2 \times 10^{-8}$
$E_{\text{abs}} = 10^{-8}$ 很小，
但 $E_{\text{rel}} = 1$：**相对误差** 100%。

$x = 10^{8}$, $\hat{x} = 10^{8} + 1$
$E_{\text{abs}} = 1$ 看似不小，
但 $E_{\text{rel}} = 10^{-8}$：**相对精度很好**。

关键：“绝对 / 相对”说明**如何归一化误差**；下一页的“前向误差 / 残差”说明**误差测量的是哪个对象**。当真值接近 0 时，相对误差本身也可能不是合适的尺度。

前向误差与残差：测量对象不同

根的前向误差

$$|x - r| \le \varepsilon.$$

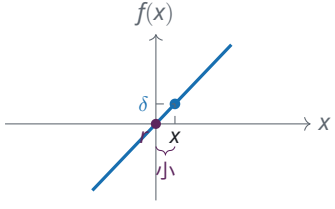
计算出的根在**水平方向**离真根有多远。

残差 / Residual

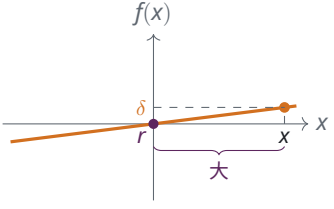
$$|f(x)| \le \varepsilon.$$

把近似解代回方程后，在**函数值方向**偏离零多少。

小残差不一定意味着小前向误差：关键在根附近的斜率。



根附近斜率较大



根附近斜率较小

误差从哪里来？先把三层分开

Problem / data	Algorithm	Machine
测量、建模、输入扰动	截断、停止、近似子问题	有限精度与舍入
$x \longrightarrow x + \Delta x$	$y \longrightarrow \widehat{y}$	$z \longrightarrow \mathrm{fl}(z)$
问题本身会不会放大扰动?	算法是否保留已有精度?	单步误差如何累积?

conditioning $\neq$ **algorithmic stability** $\neq$ **roundoff.**

本课程会反复问：**误差从哪一层进入？经过计算后被放大多少?**

浮点数：只需要记住一个理想化模型

在没有溢出 / 下溢等异常情形时，对基本运算 $\circ \in \{+, -, \times, /\}$，常用理想化模型是

$$\boxed{\mathrm{fl}(a \circ b) = (a \circ b)(1 + \delta), \qquad |\delta| \le u.}$$

好的消息

每一步的**相对**舍入误差通常很小：双精度浮点数的 $u \approx 10^{-16}$。

坏的消息

很多个很小的误差，经过不良条件的问题或不稳定算法，可以变成很大的答案误差。

今天不会做完整的浮点误差分析；我们先学习比 IEEE 754 更普遍的**稳定性语言**。

Fun fact. IEEE 754 的重要奠基者 William Kahan 获得了 **1989 年图灵奖**。 Source: ACM A.M. Turing Award.

37 秒：Ariane 5 的一次昂贵教训

发生了什么?

Ariane 5 首飞沿用了惯性参考系统中的旧软件。一个与水平速度相关的量超出了原设计假设的范围；将 **64 位浮点值转换为 16 位有符号整数**时触发异常，主、备两套惯性参考系统随后失效。

真正的教训： 数值范围、软件假设、异常处理、测试覆盖面都是**计算模型的一部分**。
一个假设在旧系统里“从未出错”，并不会因此变成定理。

Source: ESA/CNES, Ariane 501 Inquiry Board Report (1996).

一个实用的稳定化：Softmax 平移技巧

$$\mathrm{softmax}(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}}.$$

Naive representation

取 $z = (1000, 999, 998)$。
在双精度浮点数中直接形成

$$e^{1000}$$

会 溢出；后面的归一化甚至可能得到 `NaN`。

Shift before exponentiating

令 $m = \max_j z_j = 1000$。整体平移不改变 softmax：

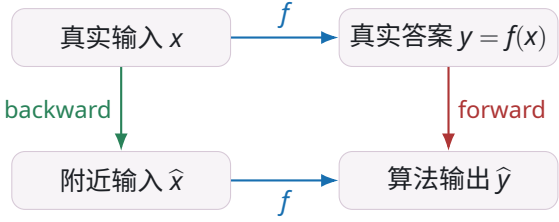
$$\mathrm{softmax}(z)_i = \frac{e^{z_i - m}}{\sum_j e^{z_j - m}}.$$

现在指数只有 $0, -1, -2$。

同一个技巧给出 **log-sum-exp**：$\log \sum_j e^{z_j} = m + \log \sum_j e^{z_j - m}$。

同一种思路还会出现在：范数计算先缩放 · compensated summation · 稳定的二次公式。

前向误差与后向误差



- **前向误差**: $\hat{y}$ 离 $f(x)$ 多远?
- **后向误差**: 最小需要把输入改动多少，才能让 $\hat{y} = f(\hat{x})$?

后向稳定 $\approx$
“精确解决了一个几乎相同的问题”。

在算法之前：先检查问题是否适定

Hadamard 的经典问题意识可以翻译成三个问题：

Existence

解存在吗？

$Z(f) \neq \varnothing$?

没有解，算法无从谈起。

Uniqueness / target

解唯一吗？若不唯一，输出目标是否明确？

$|Z(f)| = 1$?

多解也可以算，但目标必须说清楚。

Stability

输入稍变，解是否稍变？

$\Delta f \Rightarrow \Delta r$?

连续依赖才允许可靠计算。

数值分析从选择算法之前就已经开始。

先问：**有解？目标明确？稳定？**

条件数：问题本身有多敏感？

考虑标量映射 $y = f(x)$。若输入变成 $x + \Delta x$，局部线性化给出

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

因此

$$\frac{|\Delta y|}{|y|} \approx \underbrace{\left|\frac{x f'(x)}{f(x)}\right|}_{\kappa_f(x)} \frac{|\Delta x|}{|x|}, \qquad \boxed{\kappa_f(x) = \left|\frac{x f'(x)}{f(x)}\right|}.$$

当 $x \neq 0$、$f(x) \neq 0$；这是一个**局部相对**条件数。

How to read it

$\kappa \ll 1$：扰动被压缩。
$\kappa \gg 1$：扰动被放大。

Three-second example

$$f(x) = x^p \quad \Longrightarrow \quad \kappa_f(x) = |p|.$$

导数不仅是切线斜率，也是局部的误差放大器。

把问题敏感度与算法误差接起来

Problem conditioning

输入扰动 $\Delta x \quad \Longrightarrow \quad$ 真解变化 Δy

$$\frac{|\Delta y|}{|y|} \approx \kappa \frac{|\Delta x|}{|x|}.$$

这是**问题**的属性。

Algorithmic stability

计算得到 $\widehat{y}$，能否解释为

$$\widehat{y} = f(x + \Delta x), \qquad |\Delta x| \text{ small?}$$

这是**算法**的属性。

前向误差 $\lesssim$ **条件数** $\times$ **后向误差**

(local / first-order; use compatible error scales)

病态问题限制任何算法；不稳定算法则会浪费本来可以保留的精度。

局部线性化：今天真正的主角

一个可微函数在局部看起来像线性函数：

$$f(x+h) = f(x) + f'(x)h + O(h^2).$$

用来分析

输入误差 h

$\Downarrow$

输出误差 $\approx f'(x)h$

用来设计算法

把 f 换成切线

$\Downarrow$

解一个简单的线性问题

条件数与牛顿法，其实来自同一个泰勒展开。

根的条件性：从残差到位置误差

设 $f(r) = 0$，而我们只能把方程满足到一个小残差尺度 η：

$$f(r + \Delta r) = \eta.$$

在简单根 r 附近做一阶展开：

$$\eta = f(r + \Delta r) \approx f(r) + f'(r)\Delta r = f'(r)\Delta r.$$

因此

$$\boxed{\Delta r \approx \frac{\eta}{f'(r)}} \quad \Longrightarrow \quad \boxed{\kappa_{\text{root}}^{\text{abs}}(r) \approx \frac{1}{|f'(r)|}}.$$

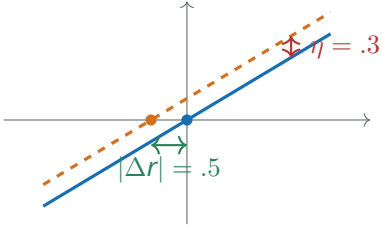
Steep crossing

$|f'(r)|$ 大：小残差通常意味着小根误差。

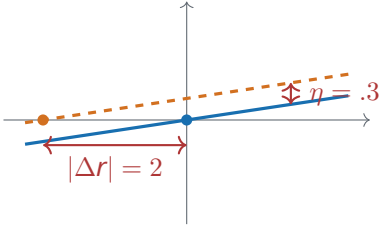
Flat crossing

$|f'(r)|$ 小：同样的残差可能对应很大的位置误差。

同样的纵向扰动，根可以移动完全不同的距离



steep: error stays local



flat: error becomes displacement

同样的 $\eta = .3$，根的位移正好按 $1/|f'(r)|$ 放大。

残差是可观测的；前向误差是否小，则取决于条件性。

一个算法问题需要完整的定义

$$\text{find } \widehat{r} \quad \text{s.t.} \quad \operatorname{dist}\big(\widehat{r}, Z(f)\big) \le \varepsilon, \qquad Z(f) = \{x : f(x) = 0\}.$$

Promise	Oracle	Accuracy	Cost
continuous? monotone? smooth? interval?	sign only? $f(x)$? $f'(x)$? higher order?	$\lvert\widehat{r} - r\rvert$? relative error? residual? probability?	queries? arithmetic? 精度所需的比特数? memory?

Promise + Information + Accuracy + Cost 才组成一个算法问题。

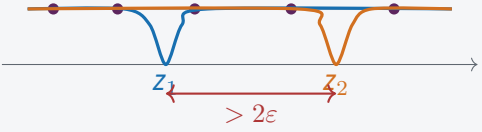
为什么需要结构假设？一个对手思维实验

算法只查询有限多个点 $x_1, \ldots, x_T$，而每次都得到同一个回答 $f(x_i) = 1$。

算法看到的查询记录

算法最后只能给出**同一个**输出 $\widehat{r}$。

但同一查询记录背后可以是两个不同实例



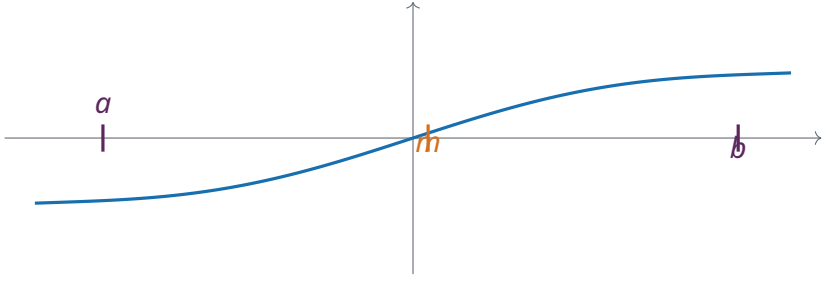
同样的查询结果，却可以有根在 z_1 或 z_2。算法不可能对两者都做到 $|\widehat{r} - r| \le \varepsilon$。

结构假设 = 信息：它排除了与已知数据不可区分的坏实例。

二分法：用拓扑信息换取全局保证

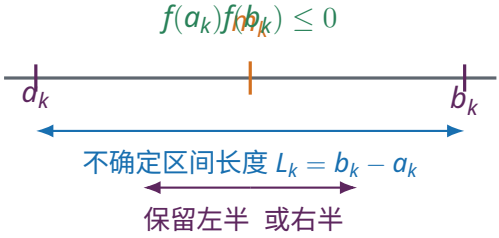
介值定理

若 f 连续且 $f(a)f(b) < 0$，则 $[a, b]$ 内至少有一个根。



每一步只问一个问题： **根在哪一半？**

二分法 = 不变量 + 不确定性收缩



One iteration

1. 查询中点 m_k。
2. 检查 $f(m_k)$ 的符号。
3. 保留仍然异号的那一半区间。

Invariant

$$f(a_k)f(b_k) \le 0.$$

由连续性，区间内始终至少保留一个根。

Progress

$$L_{k+1} = \frac{1}{2} L_k.$$

每次查询都将不确定区间恰好缩短一半。

二分法的收敛速度 = 一行复杂度分析

初始区间长度 $L_0 = b - a$。k 次后

$$L_k = \frac{b-a}{2^k}.$$

输出中点 x_k，则某个被保留的根 r 满足

$$|x_k - r| \le \frac{b-a}{2^{k+1}}.$$

要达到 $|x_k - r| \le \varepsilon$，只需

$$\boxed{k \ge \left\lceil \log_2 \frac{b-a}{2\varepsilon} \right\rceil}.$$

一个值得记住的算法学规律

每一步把不确定性缩小一个常数因子 $\rho < 1$，通常意味着达到精度 ε 需要 $O(\log(1/\varepsilon))$ 步。

在单比特预言机中，二分法已经最优

甚至只考虑 $f_r(x) = x - r$，未知 $r \in [a, b]$。令预言机每次只返回一个比特：

$$Q_r(x) = \mathbf{1}[r \le x],$$

也就是带固定规则处理相等情形的符号 / 阈值查询。

Decision tree

k 次自适应单比特查询

$\Downarrow$

最多 2^k 种查询记录

必须区分多少种答案?

可选出

$$N = \Theta\left(\frac{b-a}{\varepsilon}\right)$$

个两两距离 $> 2\varepsilon$ 的候选根。

$$2^k \ge N \quad \Longrightarrow \quad \boxed{k = \Omega\left(\log \frac{b-a}{\varepsilon}\right).}$$

二分法在这个 **单比特预言机模型** 中已最优；要更快，必须利用更丰富的局部信息。

其实就是连续版的“猜数字 / 20 Questions”：每问一次，只得到 left / right 一个 bit。

从求根到不动点：方程 $\neq$ 算法

把 $f(x)=0$ 改写成 $x=g(x)$ 后，可以迭代 $x_{k+1}=g(x_k)$。
但**同一个方程**可以对应完全不同的动力学。以 $x^2=2$ 为例：

一个合法但糟糕的改写

$$g_1(x)=\frac{2}{x}.$$

从 $x_0=1$：

$$1\to 2\to 1\to 2\to\cdots$$

而且 $g_1'(\sqrt{2})=-1$。

一个精心设计的改写

$$g_2(x)=\frac{1}{2}\left(x+\frac{2}{x}\right).$$

从 $x_0=1$：

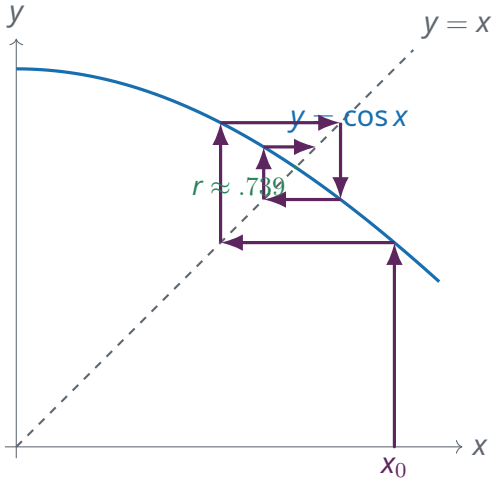
$$1\to 1.5\to 1.4167\to\cdots$$

而且 $g_2'(\sqrt{2})=0$。

选择更新映射 g，本身就是算法设计。

Cobweb 图：一次迭代到底在做什么？

以 $x_{k+1} = \cos x_k$、$x_0 = 1$ 为例。



横着碰到 $y = x$，就是把 $g(x_k)$ 重新解释为下一次输入 x_{k+1}。

局部稳定性：导数决定收敛还是发散

设 $r = g(r)$，误差 $e_k = x_k - r$。由中值定理

$$e_{k+1} = g(x_k) - g(r) = g'(\xi_k) e_k.$$

若 $x_k \to r$，则局部地

$$\frac{|e_{k+1}|}{|e_k|} \to |g'(r)|.$$

稳定不动点

$$|g'(r)| < 1$$

误差被局部**压缩**。

不稳定不动点

$$|g'(r)| > 1$$

误差被局部**放大**。

压缩映射：把局部直觉变成全局保证

压缩映射定理（一维版本）

若 $g:[a,b]\to[a,b]$，且 $|g(x)-g(y)|\le q|x-y|$ 对所有 x,y 成立，其中 $q<1$，则存在唯一不动点 r，且

$$\boxed{|x_k-r|\le q^k|x_0-r|.}$$

一维里怎么检查?

若

$$\sup_{x\in[a,b]}|g'(x)|\le q<1,$$

中值定理直接给出压缩性。

证明模板

$$|e_{k+1}|\le q|e_k|$$

反复迭代即可；唯一性也只需一行。

如何比较“收敛得有多快”？

设 $e_k = |x_k - r| \to 0$。若对某个 $p \geq 1$，

$$\lim_{k\to\infty} \frac{e_{k+1}}{e_k^p} = C,$$

其中 $0 < C < \infty$（当 $p = 1$ 时还要求 $C < 1$），则称收敛阶为 p。

p	名称	计算上的含义
1	linear	误差每步乘一个固定因子 $C < 1$
> 1	superlinear	越接近，误差缩得越快
2	二次	正确位数大约每步翻倍

能不能设计一个固定点映射 g，让 $g'(r) = 0$?

牛顿法的想法：在当前位置把世界“线性化”

在 x_k 处，用切线近似 f：

$$f(x) \approx f(x_k) + f'(x_k)(x - x_k).$$

不直接解难方程 $f(x) = 0$，而是先解这个线性近似：

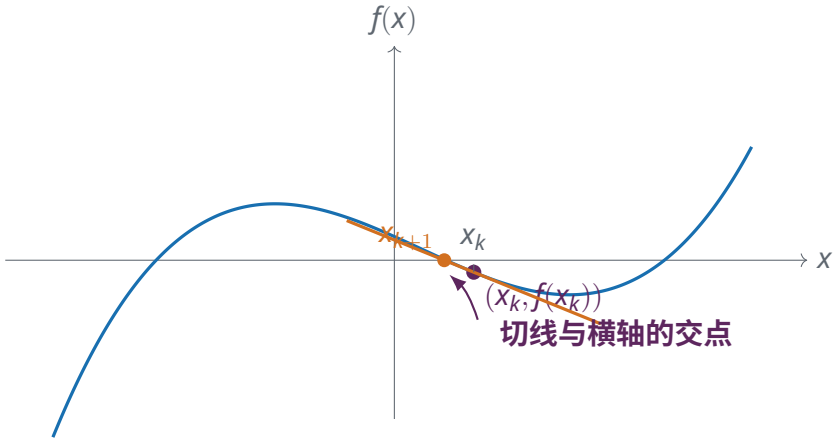
$$0 = f(x_k) + f'(x_k)(x_{k+1} - x_k).$$

于是

$$\boxed{x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.}$$

局部近似 $\rightarrow$ 精确求解局部模型 $\rightarrow$ 重复迭代。

牛顿步的几何图像



切线不是“画图技巧”：它是**一阶局部模型**。

牛顿法也是固定点迭代——而且是“精心设计”的固定点

定义

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

简单根 r 满足 $f(r) = 0$，因此 $g(r) = r$。
更关键的是

$$g'(x) = \frac{f(x)f''(x)}{[f'(x)]^2}, \qquad \Longrightarrow \qquad \boxed{g'(r) = 0.}$$

设计原则

普通固定点迭代的一阶误差项是 $g'(r)e_k$。牛顿法选择 g，使这个一阶项在根处自动消失。

牛顿法的局部定理：误差会被平方

局部二次收敛

设 $f \in C^2$，$f(r) = 0$ 且 $f'(r) \neq 0$。若 x_0 足够接近 r，牛顿迭代定义良好并收敛到 r，而且

$$\boxed{|e_{k+1}| \leq C|e_k|^2 \qquad \text{当} k \text{ 足够大时,}} \qquad e_k = x_k - r.$$

更精确地，

$$\frac{|e_{k+1}|}{|e_k|^2} \longrightarrow \left| \frac{f''(r)}{2f'(r)} \right|.$$

若 $f''(r) \neq 0$：精确收敛阶为 2。
若 $f''(r) = 0$：可能比二次收敛更快。

机制：一阶项被消掉，二阶余项主导。

证明只需要一次泰勒展开

在 x_k 附近对根 r 展开：存在 ξ_k 位于 x_k, r 之间，使

$$0 = f(r) = f(x_k) + (r - x_k) f'(x_k) + \frac{(r - x_k)^2}{2} f''(\xi_k).$$

整理：

$$\underbrace{x_k - \frac{f(x_k)}{f'(x_k)}}_{x_{k+1}} - r = \frac{f''(\xi_k)}{2f'(x_k)} (x_k - r)^2.$$

因此

$$|e_{k+1}| = \left| \frac{f''(\xi_k)}{2f'(x_k)} \right| |e_k|^2.$$

当 $x_k \to r$ 时，系数趋向 $|f''(r)/(2f'(r))|$。

牛顿步恰好消掉泰勒展开的常数项和一阶项，只留下二阶余项。

二次收敛：把收敛速度翻译成复杂度

一旦进入局部收敛域，可把常数吸收到误差尺度里，粗略看成

$$e_{k+1} \lesssim e_k^2.$$

于是经过 t 步，误差指数本身反复翻倍：

$$10^{-2} \to 10^{-4} \to 10^{-8} \to 10^{-16}, \qquad e_{k+t} \text{ 大致为 } e_k^{2^t}.$$

Iteration complexity

固定已经进入收敛域的起点，要达到 $e \le \varepsilon$，局部只需

$$\boxed{t = O\left(\log\log\frac{1}{\varepsilon}\right)}$$

次牛顿迭代。

为什么不违反二分法的下界？

二分法的下界只允许**单比特符号信息**；牛顿法使用 f, f'，而且这里只是**局部**保证。

例子：$\sqrt{2}$、巴比伦开方法与牛顿法

求 $f(x) = x^2 - 2$ 的根：

$$x_{k+1} = x_k - \frac{x_k^2 - 2}{2x_k} = \frac{1}{2}\left(x_k + \frac{2}{x_k}\right).$$

从 $x_0 = 1$ 开始：

k	x_k	$\lvert x_k - \sqrt{2}\rvert$
0	1	4.14×10^{-1}
1	1.5	8.58×10^{-2}
2	1.4166666667	2.45×10^{-3}
3	1.4142156863	2.12×10^{-6}
4	1.4142135624	$\approx 1.6 \times 10^{-12}$

这正是巴比伦 / Heron 开平方迭代。

牛顿法很快，但它不是“无条件正确”

考虑

$$f(x) = x^3 - 2x + 2.$$

从 $x_0 = 0$：

$$x_1 = 0 - \frac{2}{-2} = 1, \qquad x_2 = 1 - \frac{1}{1} = 0.$$

$$0 \longleftrightarrow 1 \longleftrightarrow 0 \longleftrightarrow \cdots$$

牛顿法进入了 2-周期，根一个也没找到。

局部收敛的真正含义

牛顿法的二次收敛定理只承诺：**当你已经足够接近一个简单根时，**它会极快收敛。

重根：算法变慢，问题本身也变脆弱

若

$$f(x) = (x - r)^m, \qquad m > 1,$$

则牛顿更新直接化成

$$x_{k+1} = r + \left(1 - \frac{1}{m}\right)(x_k - r), \qquad \boxed{e_{k+1} = \left(1 - \frac{1}{m}\right) e_k.}$$

Dynamics

$g'(r) = 1 - 1/m \neq 0$：从二次收敛降成线性收敛。

Conditioning

例如对 $\eta > 0$，微小常数扰动

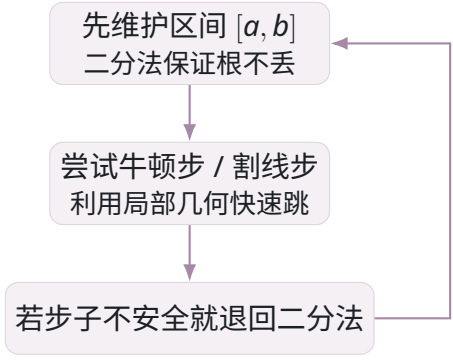
$$(x - r)^m = \eta$$

会产生尺度 $|\Delta r| = \eta^{1/m}$，远大于 η。

同一个退化 $f'(r) = 0$，同时破坏条件性与牛顿迭代的动力学。

因此：算法收敛快不等于问题条件良好；两件事必须分别分析。

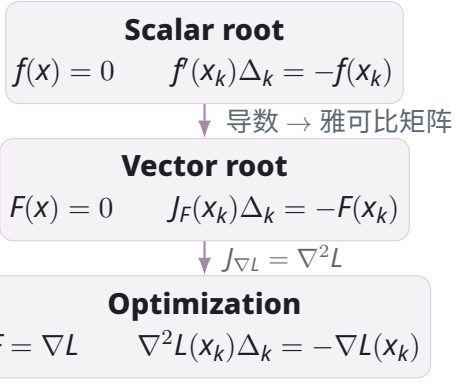
实践中的经典策略：全局 + 局部



全局可靠性 + **局部高速**。

Dekker / Brent 等求根方法正是这种设计哲学。

牛顿法的同一个模板：标量 → 向量 → 优化



建立局部线性模型 → 解线性方程 → 更新迭代点。
高维牛顿法每一步的主要代价，往往正是那个线性方程求解。

今天真正值得带走的四个观念

思想	今天的表现
Approximation	用 ε 定义问题；用简单对象逼近复杂对象
Stability	区分条件性 / 前向误差 / 后向误差
Dynamics	把算法看成 $x_{k+1} = g(x_k)$ 的迭代系统
Linearization	用导数分析敏感性，也用导数设计牛顿步

同一个导数 f'，既告诉我们“误差如何传播”，也告诉我们“算法该往哪走”。

一分钟检查：你应该能回答什么？

1. 为什么残差很小，仍可能离真实根很远？哪一个量控制这种放大？
2. 条件性与算法稳定性分别属于“问题”还是“算法”？
3. 为什么二分法需要 $\Theta(\log((b-a)/\varepsilon))$ 次符号查询，而且这个量级无法改进？
4. 同一个方程为什么可以有收敛、发散、周期等完全不同的固定点迭代？
5. 牛顿法为什么在简单根附近至少二次收敛？一句话解释，不要背证明。

第 5 题的理想答案： **牛顿法精确解掉局部一阶模型，所以误差只从二阶余项开始。**

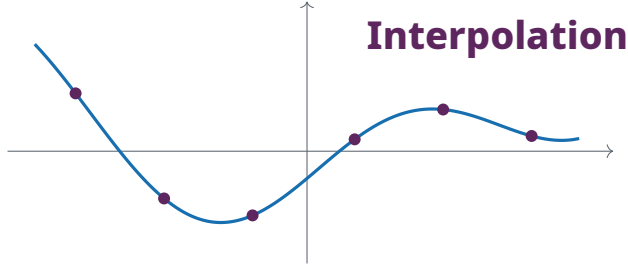
下节课：从“局部线性”到“全局多项式”

今天我们用一条切线近似函数：

$$f(x) \approx f(x_0) + f'(x_0)(x - x_0).$$

下次换一个问题：

给定$(x_0, f(x_0)), \ldots, (x_n, f(x_n))$，　能否构造一个全局近似$p(x)$?



课后练习

四个值得做的练习

1. 对 $x^2 = 2$ 比较 $g_1(x) = 2/x$ 与 $g_2(x) = (x + 2/x)/2$ 的局部动力学。
2. 把单比特符号预言机的二分法下界写成完整证明。
3. 对 $f(x) = (x - r)^m$ 分析牛顿法；若已知重数 m，怎样修改更新式？
4. 实现带保护的牛顿法：牛顿步离开当前区间时退回二分法。

四题分别对应：迭代动力学、信息下界、重根与条件性、全局保证与局部加速。

Lecture 1 takeaway

近似决定我们问什么，**条件性**决定信息能有多可靠；
迭代把算法变成动力学，**线性化**把局部几何变成速度。

导数既刻画**误差放大**，也指导**算法设计**。