NANJING UNIVERSITY · 计算机学院

计算方法

Computational Methods

Lecture 10：多元微分与自动微分

Multivariable Differentiation and Automatic Differentiation

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
入群备注： 姓名、专业、学号

教材： Timothy Sauer，《数值分析》（原书第 2 版）。

最新通知以课程主页为准。



回顾：我们在哪些地方用过导数？

Lecture 1：Newton 方法

用局部线性近似代替原来的非线性方程：

$$f(x+h) \approx f(x) + f'(x)h.$$

得到迭代 $x^+ = x - f(x)/f'(x)$。

Lecture 4：最小二乘

对 $F : \mathbb{R}^n \to \mathbb{R}$，沿方向 h 的一阶变化：

$$F(x+h) = F(x) + \langle \nabla F(x), h \rangle + o(\|h\|).$$

当时得到 $\nabla \|Ax-b\|_2^2 = 2A^\top(Ax-b)$。

Lectures 6–9：线性迭代、幂迭代与 QR 迭代让我们看到：算法也由一连串矩阵运算组成。

本讲的问题：知道导数是什么，还不等于知道**如何高效计算导数**。如果一个函数由许多计算步骤组成，能否直接沿这些步骤求导？

函数由程序计算：怎样高效求导？

设一个程序通过一系列基本运算计算 $F:\mathbb{R}^n \to \mathbb{R}$：

$$x \longrightarrow \text{线性变换} \longrightarrow \text{非线性变换} \longrightarrow F(x).$$

假设计算一次 $F(x)$ 的成本可以接受，那么求出全部 n 个偏导数，是否还要重复运行程序 n 次？

朴素办法

逐个坐标求导，或一次扰动一个坐标，用函数值之差估计偏导数（有限差分）。

当 n 很大时，代价可能很高。

算法办法

利用各步运算的依赖关系，复用中间结果，逐步传递局部导数。

链式法则可以成为算法。

本讲路线：局部线性化 $\to$ 链式法则 $\to$ 计算图 $\to$ 两种求导算法 $\to$ 最小二乘与代价分析。

贯穿的问题：能否复用一次计算中得到的中间结果，而不必对每个输入变量都从头求导？

多元微分：用线性映射描述局部变化

对 $f:\mathbb{R}^n\to\mathbb{R}^m$，若在 x 处可微，就存在一个线性映射 $J_f(x):\mathbb{R}^n\to\mathbb{R}^m$，使

$$\boxed{f(x+h)=f(x)+J_f(x)h+o(\|h\|_2).}$$

Jacobian 矩阵是该线性映射在标准基下的表示：

$$J_f(x)=\begin{pmatrix}\frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n}\\ \vdots & & \vdots\\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n}\end{pmatrix}\in\mathbb{R}^{m\times n}.$$

维度要牢记：$h\in\mathbb{R}^n$ 是输入扰动，$J_f(x)h\in\mathbb{R}^m$ 是对应的一阶输出扰动。

标量输出：Jacobian 与梯度是什么关系？

若 $F : \mathbb{R}^n \to \mathbb{R}$ 可微，Lecture 4 的定义是

$$F(x+h) = F(x) + \langle \nabla F(x), h \rangle + o(\|h\|_2).$$

因此在标准欧氏内积下

$$\underbrace{J_F(x)}_{1\times n} = \underbrace{\nabla F(x)^\top}_{1\times n}, \qquad \nabla F(x) = J_F(x)^\top.$$

梯度：一个列向量

$\nabla F(x) \in \mathbb{R}^n$

与方向向量取内积，得到方向导数。

Jacobian：一个行矩阵

$J_F(x) \in \mathbb{R}^{1\times n}$

将输入方向映射为输出变化量。

先算一个 Jacobian：每一列代表什么？

取一个具体的向量值函数：

$$g(a,b,c)=\begin{pmatrix}u\\v\end{pmatrix}=\begin{pmatrix}ab\\ab+c\end{pmatrix}:\mathbb{R}^3\to\mathbb{R}^2.$$

它的 Jacobian 是

$$J_g(a,b,c)=\begin{pmatrix}b & a & 0\\ b & a & 1\end{pmatrix}.$$

在 $(a,b,c)=(2,3,1)$，沿第一个坐标方向 e_1 的**一阶变化率**为

$$\underbrace{J_g(2,3,1)}_{2\times 3}\underbrace{\begin{pmatrix}1\\0\\0\end{pmatrix}}_{e_1}=\begin{pmatrix}3\\3\end{pmatrix}.$$

30 秒检查：沿 c 的坐标正方向，g 的方向导数是多少？下一步：如果再对 g 的输出施加一个函数，导数该怎么算？

多元链式法则：按顺序相乘

设 $g:\mathbb{R}^n \to \mathbb{R}^p$ 与 $f:\mathbb{R}^p \to \mathbb{R}^m$ 可微，$F = f \circ g$。

$$\mathbb{R}^n \xrightarrow{\ g\ } \mathbb{R}^p \xrightarrow{\ f\ } \mathbb{R}^m.$$

两次局部线性化给出

$$\boxed{J_F(x) = J_f(g(x))\, J_g(x).}$$

维度检查：

$$\underbrace{m \times n}_{J_F} = \underbrace{(m \times p)}_{J_f}\,\underbrace{(p \times n)}_{J_g}.$$

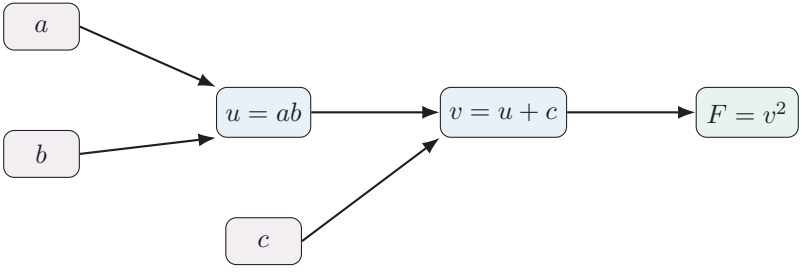
顺序不能反。
输入扰动先经过 J_g，再经过 J_f。

先判断：若 $n = 1000,\ p = 20,\ m = 1$，三个 Jacobian 各有多少行、多少列?

把表达式画成计算图（Computational Graph）

现在将前面 $u = ab,\ v = ab + c$ 的例子接上一个平方操作：

$$F(a, b, c) = (ab + c)^2, \qquad u = ab, \quad v = u + c, \quad F = v^2.$$



计算图：每个结点执行一个简单操作；箭头表示数据依赖。**共有子表达式只需要计算一次。**

计算图第一步：按顺序求出数值

给定 $a = 2,\ b = 3,\ c = 1$：

$$
\begin{aligned}
u &= ab = 6, \\
v &= u + c = 7, \\
F &= v^2 = 49.
\end{aligned}
$$

观察：计算 F 的过程中已经得到了 u 和 v。
问题：求导时能否直接利用这两个中间结果，而不用从头展开表达式？

先有输入，才能算出依赖它的结点：$u \to v \to F$。

局部求导规则：先从最简单的操作开始

既然程序由基本运算组成，我们就先研究每种运算如何传递输入的一阶变化。记 $\mathrm{d}x$ 为输入的一阶变化，$\mathrm{d}y$ 为输出的一阶变化。

操作	函数求值	一阶变化（给定输入扰动）
线性映射	$y = Ax$	$\mathrm{d}y = A\,\mathrm{d}x$
标量加法	$y = x_1 + x_2$	$\mathrm{d}y = \mathrm{d}x_1 + \mathrm{d}x_2$
标量乘法	$y = x_1 x_2$	$\mathrm{d}y = x_2\mathrm{d}x_1 + x_1\mathrm{d}x_2$
单变量函数	$y = \phi(x)$	$\mathrm{d}y = \phi'(x)\mathrm{d}x$

这些规则都来自熟悉的微积分：线性映射、乘积法则、链式法则。

我们先研究如何沿程序**向前传播**给定扰动，再研究如何把最终目标对中间结果的敏感度**反向传播**。

前向模式：沿指定方向传递变化率

沿给定方向 p 考虑参数曲线 $x(t) = x + tp$，于是 $\dot{x} = p$。上一页的一阶变化，就成为沿这条曲线的**变化率**。

$$\dot{y} \overset{\text{def}}{=} \frac{\mathrm{d}}{\mathrm{d}t} f(x(t))\Big|_{t=0} = J_f(x)p.$$

这叫做**Jacobian–vector product（JVP）**。

输入：当前位置 x，一个方向 p。

输出：f 沿该方向的变化率（方向导数）$J_f(x)p$。

不需要：
把整个 $m \times n$ 的 Jacobian 显式写出来。

如果 $p = e_i$ 是第 i 个坐标方向，那么 $J_f(x)e_i$ 就是 Jacobian 的第 i 列。

前向传播：同时计算数值和方向导数

对图中的三个操作，应用乘法法则和链式法则：

$$
\begin{aligned}
u &= ab & \dot{u} &= b\dot{a} + a\dot{b}, \\
v &= u + c & \dot{v} &= \dot{u} + \dot{c}, \\
F &= v^2 & \dot{F} &= 2v\dot{v}.
\end{aligned}
$$

前向模式的算法：
按计算图的拓扑顺序，依次求每个结点的 数值 和 一阶变化量。

局部求导规则 + 一次前向扫描。

一次前向扫描：偏导数与方向导数

仍取 $(a,b,c)=(2,3,1)$，选择方向 $(\dot a,\dot b,\dot c)=(1,0,0)$。

$$
\begin{aligned}
(u,\dot u) &= (6,\ 3),\\
(v,\dot v) &= (7,\ 3),\\
(F,\dot F) &= (49,\ 42).
\end{aligned}
$$

因此

$$\boxed{\frac{\partial F}{\partial a}(2,3,1)=42.}$$

45 秒检查：改选非坐标方向 $(\dot a,\dot b,\dot c)=(1,1,-1)$。沿这个方向，$\dot u,\dot v,\dot F$ 分别是多少？

前向模式适合解决哪些求导问题?

先核对： 方向 $(1,1,-1)$ 给出 $\dot{u}=5,\ \dot{v}=4,\ \dot{F}=56$。

设 $f:\mathbb{R}^n\to\mathbb{R}^m$，每次前向模式给出一个 $J_f(x)p$。

如果只关心一个输入方向

做一次 JVP 即可。

例如：只需分析某个指定扰动的影响。

如果需要完整 Jacobian

令 $p=e_1,\dots,e_n$，

共做 n 次 JVP，得到 n 列。

如果需要完整 Jacobian，且 n 远小于 m，前向模式往往合适。可是许多损失函数的输入维数很高，输出却只有一个标量。

能不能反过来，从这个标量出发只扫描一次?

反向模式：最终目标对中间量有多敏感？

给定一个标量目标 L，对任意中间变量 z 定义其**伴随量（adjoint）**

$$\bar{z} := \nabla_z L.$$

当 z 是标量时，$\bar{z} = \partial L / \partial z$；当 $z \in \mathbb{R}^p$ 时，$\bar{z} \in \mathbb{R}^p$。

直观理解： $\bar{z}$ 衡量最终目标 L 对中间结果 z 的敏感度。
知道输出端的敏感度后，就能沿计算图用链式法则**反向传递**到各个输入。

在终点 L 本身：$\bar{L} = 1$。这就是反向传播的初值。

反向传递的线性代数：Jacobian 转置

假设 $z = f(x)$，最终损失为 $L(z)$。对任意扰动 h，

$$\mathrm{d}L = \langle \bar{z}, \mathrm{d}z \rangle = \langle \bar{z}, J_f(x)h \rangle = \langle J_f(x)^\top \bar{z}, h \rangle .$$

由梯度定义得到

$$\boxed{\bar{x} = J_f(x)^\top \bar{z}.}$$

标准的**vector–Jacobian product（VJP）**写作行向量 $\bar{z}^\top J_f(x)$；上式是其转置后的列向量表示。

注意维度：$J_f(x) \in \mathbb{R}^{m\times n}$；输入 $\bar{z} \in \mathbb{R}^m$，输出 $\bar{x} \in \mathbb{R}^n$。
整个过程不需要显式构造 $J_f(x)$。

反向模式：每一步具体怎样传递?

假设某个标量结点的输出为 z，其伴随量 $\bar{z}$ 已经求出。对每个输入，把 $\bar{z}$ 乘以对应的局部导数，并**加到**该输入的伴随量中：

基本操作	前向求值	反向贡献（累加到输入）
加法	$z = x + y$	$\bar{x} \mathrel{+}= \bar{z}, \quad \bar{y} \mathrel{+}= \bar{z}$
乘法	$z = xy$	$\bar{x} \mathrel{+}= y\bar{z}, \quad \bar{y} \mathrel{+}= x\bar{z}$
单变量函数	$z = \phi(x)$	$\bar{x} \mathrel{+}= \phi'(x)\bar{z}$

应用于 $u = ab,\ v = u + c,\ F = v^2$**：**伴随量先置零，从 $\bar{F} = 1$ 出发，按逆求值顺序计算

$$\bar{v} = 2v, \qquad (\bar{u}, \bar{c}) = (\bar{v}, \bar{v}), \qquad (\bar{a}, \bar{b}) = (b\bar{u}, a\bar{u}).$$

$+=$ 表示“加到已有结果上”；遇到计算图分叉时尤其重要，稍后再看一个例子。

例子：一次反向扫描得到全部梯度

在 $(a,b,c)=(2,3,1)$，前向已经求出 $u=6,\ v=7,\ F=49$。

$$\begin{aligned} \bar{F} &= 1, & \bar{v} &= 2v\bar{F} = 14, \\ \bar{u} &= 14, & \bar{c} &= 14, \\ \bar{a} &= b\bar{u} = 42, & \bar{b} &= a\bar{u} = 28. \end{aligned}$$

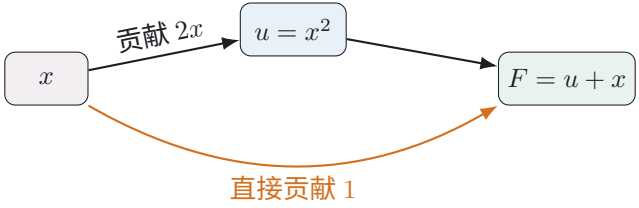
$$\boxed{\nabla F(2,3,1) = \begin{pmatrix} 42 \\ 28 \\ 14 \end{pmatrix}.}$$

复核：由第 7 页的 Jacobian 可得 $\nabla F = 2(ab+c)(b,a,1)^{\top}$。

比较：沿 a 方向前向传播得到 42；反向传播一次得到 $(42,28,14)$。两种算法使用相同的计算图，区别只是传播的信息和方向。

计算图有分叉：梯度要相加，不是覆盖

看看 $F(x) = x^2 + x$，把它写成两个运算：$u = x^2,\ F = u + x$。



$$\boxed{\bar{x} = 2x\,\bar{u} + 1 \cdot \bar{F} = 2x + 1.}$$

计算图是 DAG，不一定是一条链。某个中间结点可能影响输出的多条路径，反向传播必须**累加来自所有后继结点的贡献**。

反向传播为什么是一种 DAG 动态规划?

计算图是有向无环图（DAG）。每个结点的结果既可能被多个后续结点使用，也可能影响最终目标的多条路径。

1. **前向：**按拓扑顺序计算每个结点的数值，复用公共子表达式。
2. **初始化：**输出标量 L 的伴随量为 $\bar{L} = 1$，其余初始化为零。
3. **反向：**按逆拓扑顺序，把“局部导数 $\times$ 当前伴随量”**累加**到每个输入的伴随量。
4. **为何正确：**轮到某个结点时，所有后继都已处理，其伴随量已经**累加了全部后续贡献**。

不是神经网络专属：这就是**链式法则 + 在 DAG 上共享中间结果的动态规划**。

课堂练习：你能自己做一次反向传播吗？

令 $F(x, y) = \sin(xy) + x^2$，写成

$$u = xy, \qquad v = \sin u, \qquad w = x^2, \qquad F = v + w.$$

任务（约 6 分钟）：

1. 用约 1 分钟画出计算图，标出从 x 到 F 的两条路径。
2. 写出前向求值的拓扑顺序。
3. 从 $\bar{F} = 1$ 出发，逆序写出局部反向传播规则。
4. 在分叉处累加贡献，最后写出 $\nabla F(x, y)$。

提醒： x 同时进入乘法结点和平方结点。反向计算 $\bar{x}$ 时应把两条路径的贡献相加。

课堂练习：检查两条路径是否正确相加

前向图： $u = xy,\ v = \sin u,\ w = x^2,\ F = v + w$。**反向：** 从 $\bar{F} = 1$ 开始，在每个结点使用局部导数：

$$\bar{v} = 1, \qquad \bar{w} = 1, \qquad \bar{u} = \bar{v}\cos u = \cos(xy),$$

$$\bar{x} = \underbrace{y\bar{u}}_{\text{经过}u} + \underbrace{2x\bar{w}}_{\text{经过}w}, \qquad \bar{y} = x\bar{u}.$$

$$\boxed{\nabla F(x,y) = \begin{pmatrix} y\cos(xy) + 2x \\ x\cos(xy) \end{pmatrix}.}$$

算法要点： 除了按逆拓扑顺序遍历，还要把从同一个结点通往输出的**各条路径的贡献**累加起来。

再看链式法则：为什么有两种求导算法?

若 $F = f_k \circ \cdots \circ f_1$，设各步 Jacobian 为 $J_1, \ldots, J_k$，则

$$J_F = J_k \cdots J_2 J_1. \qquad \text{矩阵相乘可改变结合顺序，但不能交换次序。}$$

前向：给一个输入方向 p

$J_k(\cdots(J_2(J_1 p)))$

沿程序的执行顺序传递**JVP**。

得到输出沿指定输入方向的一阶变化率。

反向：给定输出端敏感度 w

$J_1^\top(\cdots(J_k^\top w))$

沿逆执行顺序传递**VJP**。

标量输出取 $w = 1$，一次得到输入梯度。

最关键的观点：利用乘法的结合律，**逐步乘向量**，不必构造完整的中间 Jacobian。给定少量输入方向时偏向前向；给定少量输出端权重时偏向反向。

矩阵与向量相乘：前向与反向

设 $y = Ax$，其中 $A \in \mathbb{R}^{m \times n}$ 是固定矩阵。
前向：令 $x(t) = x + tp$，则 y 沿该方向的变化率为

$$\dot{y} = Ap.$$

反向：最终标量目标对 y 的梯度为 $\bar{y}$，则

$$\bar{x} = A^{\top}\bar{y}.$$

回顾 Lecture 4：最小二乘中出现的 $A^{\top}$，正是这里的**反向传递算子**。
算法观点：前向计算 Ap，反向计算 $A^{\top}\bar{y}$。即使 A 很大且稀疏，也不必形成额外的巨大 Jacobian。
接下来：如果 A 也是待求导的参数，它的梯度是什么形状？

矩阵也是参数时，梯度怎么算?

仍设 $y = Ax$，其中 $A \in \mathbb{R}^{m\times n}$。最终目标是标量函数 $L(y)$，记 $\bar{y} = \nabla_y L$。

先固定 x，只让矩阵 A 变化。由于 $y_i = \sum_j A_{ij}x_j$，链式法则给出

$$\frac{\partial L}{\partial A_{ij}} = \bar{y}_i x_j, \qquad \boxed{\nabla_A L = \bar{y}x^\top.}$$

若 A 和 x 同时变化，联合增量满足

$$(A+\Delta A)(x+\Delta x) - Ax = (\Delta A)x + A\Delta x + \underbrace{(\Delta A)(\Delta x)}_{\text{二阶小量}}.$$

忽略二阶项并计算目标的一阶变化：

$$\mathrm{d}L = \bar{y}^\top(\mathrm{d}A)x + (A^\top\bar{y})^\top \mathrm{d}x, \qquad \boxed{\nabla_x L = A^\top\bar{y}.}$$

联系 Lecture 8：$\nabla_A L$ 是**秩至多为一**的外积。矩阵梯度由各元素的偏导数组成；在 Frobenius 内积 $\langle B, C\rangle_F = \sum_{ij} B_{ij}C_{ij}$ 下，它正好与 $\mathrm{d}A$ 配对。

回到最小二乘：用反向传播重新求梯度

固定 A, b，只对 x 求导。Lecture 4 用的是平方残差；这里乘上 $1/2$，仅为省去梯度中的系数 2，不改变最优解：

$$F(x) = \frac{1}{2}\,\|Ax - b\|_2^2 .$$

写成计算图的两个步骤：

$$x \xrightarrow{\;r = Ax - b\;} r \xrightarrow{\;F = \frac{1}{2} r^\top r\;} F.$$

反向传播：从 $\bar{F} = 1$ 开始，

$$\bar{r} = r, \qquad \bar{x} = A^\top \bar{r}.$$

所以

$$\boxed{\nabla F(x) = A^\top (Ax - b).}$$

回扣 Lecture 4：$\nabla F(x) = 0$ 即 $A^\top(Ax - b) = 0$，也就是残差与 A 的列空间正交。
接下来：同一个最小二乘目标，为什么不同方向上的梯度大小会相差很多？下一页用 Lecture 8 的奇异值来解释。

奇异值如何影响最小二乘的梯度?

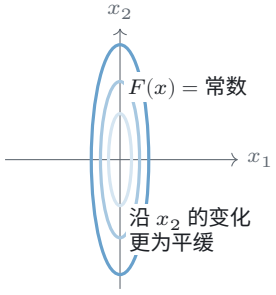
Lecture 4 从几何和微积分两方面理解最小二乘；Lecture 8 研究矩阵在不同方向上的伸缩。下面用一个二维例子把它们联系起来。

二维例子：令 $0 < \varepsilon < 1$，取

$$A = \begin{pmatrix} 1 & 0 \\ 0 & \varepsilon \end{pmatrix}, \qquad b = 0.$$

因此

$$F(x) = \tfrac{1}{2}(x_1^2 + \varepsilon^2 x_2^2), \qquad \boxed{\nabla F(x) = \begin{pmatrix} x_1 \\ \varepsilon^2 x_2 \end{pmatrix}.}$$



取 $\varepsilon = \frac{1}{4}$、$x = (1,1)^\top$，则 $\nabla F(x) = (1, \frac{1}{16})^\top$。

留给下一讲：一般地，$\nabla F(x) = (A^\top A)x - A^\top b$。
梯度算得快，不代表迭代收敛得快。那么，收敛速度取决于什么?

串联起来：线性映射与逐坐标非线性

考虑一个线性映射后接逐坐标非线性，再计算标量损失：

$$z = Ax, \qquad y = \phi(z), \qquad L = \tfrac{1}{2}\,\|y - t\|_2^2 .$$

其中 ϕ 逐坐标作用且可微，t 是给定目标；$\odot$ 表示逐坐标相乘。

前向求值： $x \to z \to y \to L$。

反向传播： 从 $\bar{L} = 1$ 出发，

$$\bar{y} = y - t, \qquad \bar{z} = \bar{y} \odot \phi'(z).$$

因此输入与参数的梯度分别为

$$\boxed{\nabla_x L = A^\top \bar{z}}, \qquad \boxed{\nabla_A L = \bar{z} x^\top} \quad \text{（若 } A \text{ 也要求导）}.$$

为什么能逐层计算？ ϕ 的 Jacobian 是对角矩阵（对角元为 $\phi'(z_i)$），因此反向作用就是逐坐标相乘；再接上线性层的反向规则。**每一层只需知道自己的局部求导规则。**

自动微分不是数值差分，也不是符号微分

方法	如何工作	主要局限 / 特点
有限差分	$(F(x+he_i)-F(x))/h$	有截断与舍入误差；全梯度常需多次求值
符号微分	把表达式改写成导数表达式	可能产生庞大的中间表达式，需做复用与化简
自动微分	在原始运算图上用局部导数规则传播	对所执行的可微基本操作应用链式法则

“自动”的意思：系统沿程序的运算步骤应用局部导数规则，无需人工展开整个函数的导数表达式。

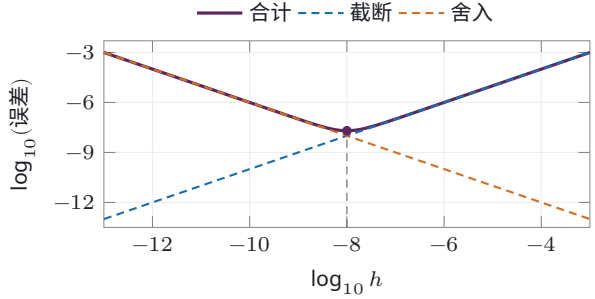
不是无限精度：自动微分仍然在浮点数上执行，也仍然会受到条件数与舍入误差影响。

有限差分：为什么步长不能一味减小？

对单变量平滑函数 F，前向差分 $D_hF(x) = (F(x+h) - F(x))/h$。在适当光滑性、函数值尺度及浮点误差假设下，常有

$$|D_hF(x) - F'(x)| \lesssim C_1 h + C_2 \frac{u}{h},$$

其中 u 是机器精度，C_1, C_2 由函数与求值过程决定。



步长较大：泰勒截断误差 $O(h)$ 占主导。

步长太小：相近浮点数相减后的误差可能被 $1/h$ 放大。

折中：平衡两项可取 $h \asymp \sqrt{C_2 u / C_1}$；常数可比时 $h \asymp \sqrt{u}$。

图仅示意：取 $u = 10^{-16}$, $C_1 = C_2 = 1$；并非每个函数的实际误差曲线都如此。

计算标量函数的梯度要花多少代价？

考虑含 T 个结点、$O(T)$ 条边的标量计算图，假设每个基本操作及其局部求导的代价为常数。

前向模式：传播输入方向

$$p \longrightarrow J_F(x)p$$

一个指定方向：$O(T)$ 时间。

求 n 个偏导数：通常做 n 次，$O(nT)$ 时间。

反向模式：传播输出敏感度

$$1 \longmapsto J_F(x)^\top 1 = \nabla F(x)$$

一次前向求值、一次反向扫描，$O(T)$ **时间**得到全部 n 个偏导数。

典型情形：目标 $L(\theta) \in \mathbb{R}$，而参数 $\theta \in \mathbb{R}^n$ 很多；逐坐标有限差分一般也要约 n 次额外函数求值。

时间与空间：反向模式通常保存中间结果，存储可达 $O(T)$；前向模式在一些图上更容易流式计算。对大型矩阵等复合算子，不能不加条件地把每个操作都视为 $O(1)$。

自动微分为什么不能替代数值稳定性分析?

回顾 Lecture 1：**等价的数学表达式可能对应不同稳定性的计算程序**。考虑

$$s(x) = \log(1 + e^x) = x + \log(1 + e^{-x}) \qquad (x > 0).$$

直接计算（$x = 1000$）

e^x 在通常的双精度浮点数中溢出。

反向传播可能计算 $e^x \cdot (1/(1 + e^x))$；浮点数中变成 $\infty \times 0$，得到 **NaN（非数）**。

稳定地重写（$x > 0$）

e^{-x} 很小；函数值可稳定求出。

导数同样可稳定计算：

$$s'(x) = 1 - \frac{e^{-x}}{1 + e^{-x}} \approx 1.$$

结论：自动微分正确应用链式法则，**却不能替你修复前向程序或局部运算的数值不稳定性**。

如何选择求导方法？还要检查什么？

先根据需要的导数信息与资源，选择传播方向：

1. **少量输入方向：**只需要一个或几个 $J_f(x)p$，通常优先考虑前向模式。
2. **标量目标的全部梯度：**通常优先考虑反向模式；若内存紧张，需要权衡保存中间结果与重新计算的代价。

无论选哪种模式，都要再检查：

3. **可微性：**绝对值、取最大值、分支等可能在某些点不可微；必须明确所用的导数规则。
4. **数值可靠性：**原问题可能病态，具体求导程序也可能不稳定；两种问题都不是 AD 自动消除的。

不能只看输入、输出维数。还要考虑所需的导数信息、可用内存，以及导数是否存在、计算是否可靠。

本讲小结：用程序计算导数

1. **局部线性化：** $J_f(x)$ 描述输入小扰动如何改变输出。
2. **链式法则：** 复合映射的 Jacobian 是局部 Jacobian 的乘积。
3. **前向模式：** 计算 $J_f(x)p$，适合少量输入方向。
4. **反向模式：** 计算 $J_f(x)^\top w$，标量输出时一次反向扫描即可得到完整梯度。
5. **算法与数值：** 计算图可以复用中间结果、累加分叉处的贡献，也可以对矩阵参数求导；存储开销与数值稳定性仍须分析。

40 秒回顾： 求 $L : \mathbb{R}^{10^5} \to \mathbb{R}$ 的全部梯度，以及 $g : \mathbb{R}^2 \to \mathbb{R}^{10^5}$ 的完整 Jacobian，分别优先用哪种模式？为什么？

下一讲： 梯度算得快，为什么迭代仍可能很慢？从二次目标与线性系统出发，我们将学习梯度下降、Krylov 子空间与共轭梯度。

课堂机动 一个两层计算图如何反向传播?

设 $h = \phi(W_1 x)$、$y = W_2 h$、$L = \frac{1}{2}\|y - t\|_2^2$，其中 ϕ 逐坐标可微。

$$
\begin{aligned}
\bar{y} &= y - t, \\
\bar{h} &= W_2^\top \bar{y}, \\
\bar{z}_1 &= \bar{h} \odot \phi'(W_1 x), \\
\nabla_{W_2} L = \bar{y} h^\top, &\qquad \nabla_{W_1} L = \bar{z}_1 x^\top, \\
\nabla_x L &= W_1^\top \bar{z}_1.
\end{aligned}
$$

结论：通常所说的**反向传播（backpropagation）**，就是把前面推导的 VJP 规则逐层应用到计算图上。
无需为神经网络另立一套求导理论。

课堂机动 省内存：保存还是重新计算？

反向传播某个结点时，往往需要前向计算保存的输入或输出。

回看 Lectures 7–9：幂迭代和同时迭代也会产生一连串中间向量；若对多步迭代程序求导，就要考虑如何保存它们。

全部保存

反向计算方便，存储中间结果多。

对很长的计算链可能不现实。

检查点（checkpointing）

只保存部分结点，在反向传播前重新计算缺失的中间值。

用额外计算换取空间。

又一次时间与空间的折中：少存一些，就可能需要多算一些。

课堂机动 更高阶的导数：Hessian–vector product

若 $F:\mathbb{R}^n\to\mathbb{R}$ 二阶可微，其 Hessian $H_F(x)=\nabla^2F(x)$ 是 $n\times n$ 矩阵。取一个方向 $p\in\mathbb{R}^n$。对已能计算的梯度函数 $g(x)=\nabla F(x)$ 再做一次前向模式：

$$\boxed{J_g(x)p=\nabla^2F(x)p.}$$

为什么有用？如果只需 Hp，无需构造和存储整个 $n\times n$ 的 Hessian。
例如下一讲的二次函数，$F(x)=\frac{1}{2}x^\top Ax-b^\top x$（$A$ 对称），其 Hessian $H_F=A$，于是 $Hp=Ap$。

注意：混合前向、反向模式时，必须正确处理中间梯度的导数规则。

补充 有方向导数，不一定有统一的线性近似

回顾：可微要求存在同一个线性映射 $J_f(x)$，对**所有**小扰动都成立

$$f(x+h) = f(x) + J_f(x)h + o(\|h\|_2).$$

仅知道每个方向上存在方向导数，还不足以保证可微。例如 $f : \mathbb{R}^2 \to \mathbb{R}$ 定义为

$$f(x,y) = \begin{cases} \frac{x^3}{x^2+y^2}, & (x,y) \neq (0,0), \\ 0, & (x,y) = (0,0). \end{cases}$$

在原点沿 (a,b) 的方向导数为 $a^3/(a^2+b^2)$（非零方向），**不是方向的线性函数**，因此原点不可微。

补充 有不可微结点时怎么办?

考虑常用逐坐标函数 $r(x) = \max\{0, x\}$。

$$r'(x) = \begin{cases} 0, & x < 0, \\ 1, & x > 0. \end{cases}$$

在 $x = 0$，这个函数**不存在普通意义下的导数**。实际计算系统可能为该点约定一个导数值；这样得到的是所选规则的计算结果，而**不是**因为原函数突然变得可微。

延伸：某些不可微的优化问题，可以借助广义导数或特殊的优化方法处理，但这些规则不能与经典链式法则混为一谈。

补充 隐式求导：有些计算通过解方程定义

设 $G(x,\theta)=0$ 定义了 $x=x(\theta)$。假设相关函数可微，且 J_xG 在该点可逆。

$$J_xG\,\frac{\mathrm{d}x}{\mathrm{d}\theta}+J_\theta G=0,\qquad \boxed{\frac{\mathrm{d}x}{\mathrm{d}\theta}=-(J_xG)^{-1}J_\theta G.}$$

例子： $A(\theta)x(\theta)=b(\theta)$，两侧求导得

$$A\dot{x}=\dot{b}-\dot{A}\,x.$$

计算启示：为了得到导数，可以**再解一个线性方程组**，而不是把求解器的每一步都展开求导。

补充 从链式法则看反向模式的正确性

设每个结点 $z_i = \phi_i(z_{p_1}, \dots, z_{p_r})$，最终输出为标量 L。如果 z_i 被若干后继结点 z_j 使用，链式法则给出

$$\boxed{\bar{z}_i = \sum_{j:\, i \to j} \left(\frac{\partial z_j}{\partial z_i}\right)^{\top} \bar{z}_j.}$$

这里局部 Jacobian 对标量结点退化为导数；对向量结点则是矩阵。

证明方式：按逆拓扑顺序归纳。所有后继的伴随量已经正确计算时，把局部贡献相加就得到当前结点的伴随量。

这解释了为何一次反向扫描足够。