

程序员与 AI 的数学对话

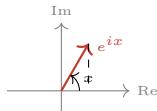
A Dialogue on Math: From Engineering to Truth

复变函数：降维打击

程序员：复变函数是什么？解决什么实际问题？

Gemini：你可以把它当成算法优化的捷径。复变函数的本质是“降维打击”。

$$e^{ix} = \cos x + i \sin x$$



在实数域极难解决的问题，一旦映射到复平面（Complex Plane），利用欧拉公式 e^{ix} 的旋转特性，复杂度会瞬间崩塌。

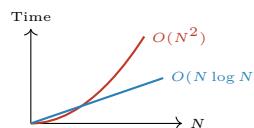
实战价值：

1. 算法加速：没有复数理论，就没有 FFT。它利用复数单位根的对称性，把信号处理复杂度从 $O(N^2)$ 降到了 $O(N \log N)$ 。
2. 稳定性判断：在控制系统或金融模型中，利用留数定理或奈奎斯特判据，画个图就能知道系统会不会崩溃。

算力时代的价值

程序员：在现代计算机算力如此强大的背景下，这些工具还有那么大的价值吗？

Gemini：这是一个非常深刻且切中要害的问题。在 19 世纪，复变函数确实会用来“手算”那些算不出来的积分。但在计算机算力爆炸的今天，它的价值非但没有降低，反而变成了底层核心竞争力。



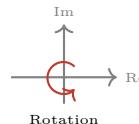
在算力强大的今天，复变函数的角色发生了转变：

- 以前：它是计算工具（帮人算）。
- 现在：它是算法设计的蓝图（教计算机怎么算）。

核心思想：升维与投影

程序员：那我能不能这样理解：正弦波计算比较复杂，所以把正弦波升维映射到三维空间中容易计算的螺旋线，算完后再投影回二维空间？

Gemini：您的这个理解非常精准，你可以把这种解题思路称为数学界的“降维打击”：



1. 升维（Mapping）：从“震荡”到“旋转”
2. 计算（Computation）：在螺旋线上做运算
3. 投影（Projection）：回到现实世界

如果非要用一句话来概括工程数学中复变函数的灵魂，那就是：

“现实世界太难走，不如去复平面借个道。”

常规武器：工程必备

程序员：除了复变函数，还有哪些需要了解的数学工具？

Gemini：对于 99% 的通用工程场景，你需要这四把“常规武器”：

$$Ax = \lambda x \quad \Delta \quad P(A|B) \quad H(X)$$

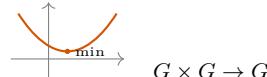
- **线性代数**：空间的变换。它是 AI 的血液（矩阵运算），也是推荐系统（向量相似度）和图形学（3D 变换）的基石。
- **图论**：关系与结构。解决依赖管理（Maven/Pip）、死锁检测、网络路由和任务调度问题。
- **概率统计**：处理不确定性。用于负载均衡（泊松分布）、A/B 测试、反垃圾邮件（贝叶斯）和风控系统。
- **信息论**：数据的量化。理解了“熵”，你就懂了数据压缩（Zip）、密码强度和机器学习中的损失函数（交叉熵）。

* 掌握这些，你已经超越了 80% 的工程师。

进阶逻辑：底层架构

程序员：还有哪些进阶的数学工具？

Gemini：那你需要触碰更底层的逻辑：



- **抽象代数**（群/环/域）：密码学的基石。RSA、ECC 加密、RAID 纠错码，本质上都是在有限域上定义的运算规则。
- **凸优化**：AI 训练的引擎。解决“在资源受限下找全局最优解”的问题，比如梯度下降。
- **数值分析**：管理浮点数误差。只要涉及物理引擎、金融模型，必须懂这个，否则 $0.1 + 0.2 \neq 0.3$ 会让系统崩溃。
- **范畴论**：架构设计的终极抽象。函数式编程（Monad）、Rust 的类型系统、异步编程模型的理论源头。它教你如何优雅地组合代码。

高阶视界：时空与博弈

程序员：还有吗？

Gemini：接下来的这 4 个数学工具，属于理论物理学家、顶尖 AI 科学家和金融风控的领域。它们描述的不再是简单的“计算”，而是“时空的形状”、“混沌的动态”和“智能的博弈”。

$$\chi(M) = \sum (-1)^k b_k \quad dX_t = \mu dt + \sigma dW_t$$

- **拓扑学**（Topology）：超越几何学。拓扑学完全不在乎长短、大小，它只关心“连续性”和“洞”。
- **微分几何**（Differential Geometry）：在“弯曲”的世界里做微积分。微积分和线性代数默认世界是“平”的。微分几何研究的就是在弯曲的空间（流形）上如何求导、如何测量距离。
- **随机微积分**（SDE）：生成模型的原理。Stable Diffusion 本质上是在解一个随机微分方程，学习如何逆转“把图片变成噪音”的过程。
- **博奕论**：智能对抗。GAN（生成对抗网络）、区块链共识机制，都是在寻找多智能体博奕下的纳什均衡。

上帝视角：物理与真理

程序员：太好了，还有更多的或者更高级的数学工具介绍吗？

Gemini：有，那是物理学家和语言设计者的领域：

$$i\hbar \frac{\partial}{\partial t} \Psi = \hat{H} \Psi$$

$$R_{\mu\nu} - \frac{1}{2} R g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$$

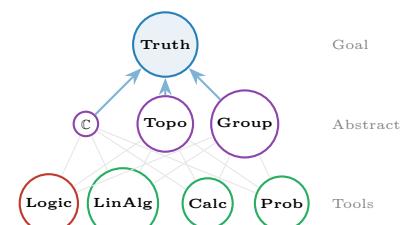
- **泛函分析**：无限维空间。量子计算的底层语言（希尔伯特空间），也是核方法（SVM）的原理。
- **张量微积分**：物理法则的容器。深度学习借用了它的名字，但在物理中，它描述了坐标变换下不变的真理（如广义相对论）。
- **辛几何**：能量守恒。用于高保真物理引擎，保证模拟过程中能量不发散。
- **类型论**：代码即真理。“写程序”和“写数学证明”是等价的。这是形式化验证（保证代码 0 Bug）和编译器设计的理论源。

总结：数学武器库

程序员：那我们能不能把今天聊过的数学工具总结成一套 Cheat Sheet？

Gemini：当然可以。从基础的线性代数到上帝视角的泛函分析，按图索骥，你就能找到解决问题的数学钥匙。

- **建立索引**：遇到瓶颈时，先查表寻找数学工具。
- **洞察本质**：透过框架，看到不变的数学结构。
- **心中有剑**：不背细节，但深知原理。



高阶程序员数学武器库 Cheat Sheet

Math for Advanced Developers: From Engineering to Truth

极简决策树 (Decision Tree)

快速定位技术瓶颈的数学解法

- 有波或周期? → 傅里叶 / 复变
- 有网络或依赖? → 图论
- 系统不稳或发散? → 拉普拉斯 / 极点
- 数据存不下或算太慢? → 线代 / 信息论
- 资源分配或调度? → 凸优化 (规划)
- 协议设计或校验? → 抽象代数 / 形式化
- 要搞加密或纠错? → 有限域 (GF)
- 要搞架构或编译器? → 范畴论 / 类型论

1. 基础与描述 (Foundation)

描述世界的基石，日常开发最常用

线性代数 (Linear Algebra)

- 核心: 空间的变换 (Space Transformation)
- 关键: 矩阵, 特征值, SVD, 秩
- 应用: AI 训练, 推荐系统, PCA 降维

概率统计 (Probability)

- 核心: 处理不确定性
- 关键: 分布 (高斯/泊松), 贝叶斯 $P(A|B)$
- 应用: 负载均衡, A/B 测试, 风控

图论 (Graph Theory)

- 核心: 关系与结构
- 关键: DAG, 拓扑排序, 最短路径
- 应用: 依赖管理 (Pip), 死锁检测, 调度

信息论 (Information Theory)

- 核心: 数据的量化
- 关键: 熵 (Entropy), 互信息, 交叉熵
- 应用: 压缩 (Huffman), 密码强度

2. 信号与变换 (Transforms)

解决“波动”、“周期”与“稳定性”

复变函数 (Complex Analysis)

- 核心: 旋转与降维
- 公式: $e^{ix} = \cos x + i \sin x$
- 应用: 交流电, 流体力学 (保角映射)

傅里叶变换 (Fourier)

- 核心: 频域视角 (Time vs Freq)
- 关键: 卷积定理, FFT ($O(N \log N)$)
- 应用: 音频处理, 图像压缩, 大数乘法

拉普拉斯 / Z 变换

- 核心: 系统稳定性 (S 域 / Z 域)
- 关键: 极点 (Poles), 收敛域
- 应用: 控制系统 (PID), 数字滤波 (DSP)

3. 计算与博弈 (Computation)

解决“怎么算最快”和“怎么选最优”

数值分析 (Numerical Analysis)

- 痛点: 浮点误差 ($0.1 + 0.2 \neq 0.3$)
- 解法: 迭代法, 条件数分析
- 场景: 物理引擎, 高频交易, 科学计算

凸优化 (Convex Optimization)

- 痛点: 在万千可能中找唯一最优
- 解法: 梯度下降, 拉格朗日对偶
- 场景: 深度学习训练, 物流调度

博弈论 (Game Theory)

- 痛点: 对手也在变, 无静态最优
- 解法: 纳什均衡 (Nash Equilibrium)
- 场景: 区块链共识, GAN 对抗网络

4. 结构与逻辑 (Logic)

架构师与安全专家的底层内功

抽象代数 (Abstract Algebra)

- 本质: 定义运算规则 (群/环/域)
- 有限域: $GF(2^8) \rightarrow$ 计算机完美算术
- 应用: AES 加密, RAID 纠错, ECC

范畴论 (Category Theory)

- 本质: 组合与解耦 (Morphism)
- 概念: Functor, Monad (单子)
- 应用: 函数式编程, 异步模型

类型论 (Type Theory)

- 本质: 代码即真理
- 原理: Curry-Howard 同构
- 应用: 形式化验证, 编译器 (Rust)

5. 高维与动态 (Dynamics)

AI 和前沿物理的核心

流形 (Manifold)

- 定义: 高维空间中卷曲的低维曲面
- 洞察: 数据是有形状的 (Embedding)
- 应用: 深度学习原理, 降维

随机微积分 (Stochastic Calc)

- 公式: $dX_t = \mu dt + \sigma dW_t$
- 特征: 确定性趋势 + 随机噪声
- 应用: 生成式 AI (Diffusion), 定价

拓扑学 (Topology)

- 关注: 连通性与洞 (忽略距离)
- 关键: 同胚, 贝蒂数 (Betti Numbers)
- 应用: 拓扑数据分析 (TDA), 鲁棒性

6. 上帝视角 (The Endgame)

触碰物理法则与计算边界

泛函分析 (Functional)

- 无限维线性代数; 希尔伯特空间
- \hookrightarrow 量子计算 / 核方法 (SVM)

辛几何 (Symplectic)

- 能量与相空间体积守恒 (哈密顿)
- \hookrightarrow 高保真物理仿真 (HMC)

代数几何 (Algebraic Geom)

- 方程与形状的统一; 概形 (Scheme)
- \hookrightarrow 零知识证明 (ZKP), 隐私计算

张量分析 (Tensor Calculus)

- 坐标无关的物理真理; 协变/逆变
- \hookrightarrow 广义相对论 / AI 底层架构

核心心法 (Philosophy)

“手中无剑 (不背细节), 心中有剑 (深知原理).”

数学不是枯燥的计算题, 而是你认知世界的显微镜与望远镜。这个武器库存在的意义, 是从工程的“How”上升到数学的“What”:

- 建立索引: 遇到瓶颈时, 不要暴力试错, 先查表寻找是否存在降维打击的数学工具。
- 洞察本质: 代码是逻辑的载体, 工程是数学的近似。透过复杂的框架, 看到背后不变的数学结构。
- 识别同构: 发现不同系统间 (如电路与流体、逻辑与几何) 的数学同构性, 实现知识迁移。
- 第一原理: 基于公理推演而非经验堆砌, 用数学思维构建反脆弱的系统架构。

当你凝视深渊 (底层原理) 时, 深渊也在为你提供无尽的算力。

线性代数 Linear Algebra Cheat Sheet

The Engine of Modern AI: From Matrix Transformations to SVD

场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 数据降维/压缩 → SVD / PCA
- 推荐系统/相似度 → 余弦相似度 (点积)
- 解线性方程组 → 逆矩阵 / 伪逆
- 图像旋转/缩放 → 线性变换矩阵
- 系统稳定性 → 特征值 (Eigenvalues)
- 概率分布变换 → 行列式 (Jacobian)
- 深度学习优化 → 梯度下降 / Hessian
- 数据投影/回归 → 最小二乘法 / QR 分解

1. 空间与向量 (Spaces)

万物皆向量, 计算即变换

基本概念

- 线性无关: $\sum c_i v_i = 0 \iff \forall c_i = 0$ 。
- 基 (Basis): 描述空间的最小坐标系。
- 秩 (Rank): 独立维度的数量。
- 张成 (Span): 所有线性组合构成的空间。

范数与度量 (Metric)

- L₁: $\sum |x_i|$ (Lasso, 稀疏解)
- L₂: $\sqrt{\sum x_i^2}$ (欧氏距离)
- 点积: $a \cdot b = |a||b|\cos\theta$ (投影)
- 正交: $a \cdot b = 0 \iff a \perp b$

2. 矩阵与映射 (Matrices)

矩阵不是数表, 而是函数的描述

四大子空间 (The Big Four)

- 列空间 $C(A)$: 值域 (Image)。
- 零空间 $N(A)$: 核 (Kernel), 解 $Ax = 0$ 。
- 行空间 $C(A^T) \perp$ 零空间。

秩-零化度定理 (Rank-Nullity)

$$\dim(N(A)) + \text{Rank}(A) = n \text{ (列数)}$$

直觉: 输入维度 = 损失的维度 + 保留的维度

3. 特特征值与特征向量 (Eigen)

寻找变换中的“不动轴”

定义: A 作用在向量 x 上, 仅伸缩不旋转。

$$Ax = \lambda x$$

物理意义

- 特征向量 (x): 矩阵的主轴、共振态。
- 特征值 (λ): 沿该轴伸缩的倍数。
- 迹 (Trace): $\sum \lambda_i = \text{tr}(A)$ 。
- 行列式 (Det): $\prod \lambda_i = \det(A)$ (体积)。

对角化 (Diagonalization) 若 A 有 n 个独立特征向量, 则 $A = SAS^{-1}$ 。本质: 换个基底, 让变换变得纯粹 (只伸缩)。

4. 奇异值分解 (SVD)

线性代数的“皇帝”公式

任意矩阵 $A (m \times n)$ 均可分解为:

$$A = U\Sigma V^T$$

- U : 左奇异向量 (正交), 输出空间基。
- Σ : 奇异值 (对角), $\sigma_i \geq 0$, 伸缩强度。
- V^T : 右奇异向量 (正交), 输入空间基。

几何直觉 任何线性变换 = 旋转 (V^T) → 拉伸 (Σ) → 旋转 (U)。

应用: 低秩近似 (PCA) $A \approx \sum_{i=1}^k \sigma_i u_i v_i^T$ 保留前 k 个最大的 σ , 即保留了主成分。

5. 正定矩阵 (Positive Definite)

矩阵里的“正数”

定义: $\forall x \neq 0, x^T Ax > 0$ 。

- 性质: 所有特征值 $\lambda_i > 0$ 。
- 几何: 碗状函数 (凸函数), 有全局最小值。
- 应用: 优化问题 (Hessian)、协方差。

6. 线性方程组 (Solving $Ax = b$)

从精确解到最优解

解的情况

- 唯一解: A 可逆 ($\det \neq 0$)。
- 无解: $b \notin C(A)$ (方程矛盾)。
- 无穷解: $N(A)$ 非空 (有自由变量)。

矩阵分解求解 (LU) $A = LU$ (下三角 \times 上三角)。高斯消元的矩阵形式, 计算机解方程的标准做法。

数值稳定性 条件数: $\kappa(A) = \|A\| \|A^{-1}\| = \sigma_{\max}/\sigma_{\min}$ 。 κ 越大, 矩阵越“病态”, 解对误差越敏感。

最小二乘法 (Least Squares) 当方程无解时, 求误差最小的解:

$$A^T A \hat{x} = A^T b$$

几何意义: 将 b 投影到 A 的列空间上。

7. Python / Numpy 实战

代码即数学

基础运算

- 乘法: $A @ B$ 或 $\text{np.dot}(A, B)$
- 逆/伪逆: $\text{inv}(A)$ / $\text{pinv}(A)$
- 爱因斯坦求和: $\text{np.einsum('ij,jk->ik', A, B)}$

SVD 与 PCA

- $U, S, V^T = \text{np.linalg.svd}(A)$
- S 是奇异值数组, 降维时取 $S[:k]$ 。

求解方程

- 精确解: $x = \text{np.linalg.solve}(A, b)$
- 最小二乘: $x, \dots, \dots, \dots = \text{np.linalg.lstsq}(A, b)$

正交化 (QR) $Q, R = \text{np.linalg.qr}(A)$ (Gram-Schmidt)

8. 关键定理 (Theorems)

支撑理论大厦的支柱

谱定理 (Spectral Thm) 任何实对称矩阵 ($A = A^T$) 都可以被正交对角化。

$$A = Q\Lambda Q^T$$

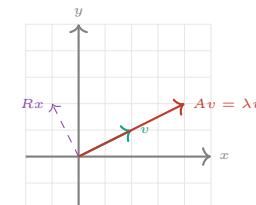
意义: 对称矩阵极其完美, 特征向量互相垂直。

Cholesky 分解 正定矩阵 $A = LL^T$ 。意义: 矩阵的“开平方”, 生成相关随机数。

Cayley-Hamilton 矩阵满足其自身的特征方程: $p(A) = 0$ 。

核心直觉 (Intuition)

“矩阵是空间的折叠椅。”



线性代数不仅仅是解方程, 它是描述空间变换的语言。

三大核心视角

矩阵即变换: 一个 $m \times n$ 矩阵是将 n 维空间扭曲、旋转、投射到 m 维空间的操作。行列式告诉我们体积是扩大了还是缩小了 ($\det = 0$ 意味着空间被拍扁了)。

特征值即主轴: 在混乱的变换中, 寻找那些“只伸缩不旋转”的特殊方向。它们构成了系统的“骨架”(如人脸识别中的特征脸)。

SVD 即数据压缩: 任何复杂的数据矩阵, 都可以拆解为: 旋转 → 缩放 → 旋转。扔掉小的缩放系数, 就做到了去噪和压缩。

数据是新时代的石油, 而线性代数是提炼石油的炼油厂。

概率统计 Probability & Statistics Cheat Sheet

Managing Uncertainty: From Bayes' Theorem to MCMC

场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 垃圾邮件/风控 → 素朴贝叶斯 (Naive Bayes)
- 流量/排队预测 → 泊松分布 (Poisson)
- AB 测试/效果评估 → 假设检验 (P-value)
- SLA/P99 延迟 → 正态分布/长尾分布
- 复杂系统模拟 → 蒙特卡洛 (Monte Carlo)
- 模型参数估计 → MLE / MAP

1. 概率公理 (Axioms)

量化不确定性的规则

基本公式

- 条件概率: $P(A|B) = \frac{P(AB)}{P(B)}$
- 全概率公式: $P(A) = \sum P(A|B_i)P(B_i)$
- 独立性: $P(AB) = P(A)P(B)$

贝叶斯定理 (Bayes' Rule)

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- 后验 \propto 似然 \times 先验
- 直觉: 用新的证据 B 更新对 A 的旧看法。

2. 随机变量 (Random Vars)

将事件映射为数字

期望与方差

- 期望 (Mean): $E[X] = \sum xp(x)$ (重心)
- 方差 (Var): $Var(X) = E[(X - \mu)^2]$ (波动)
- 协方差: $Cov(X, Y) = E[(X - \mu_x)(Y - \mu_y)]$

大数定律 (LLN) 样本越多, 平均值越接近真实期望。 $\bar{X}_n \rightarrow \mu$

切比雪夫不等式 (Chebyshev) 无论分布形状如何, 数据偏离均值 k 个 σ 的概率:

$$P(|X - \mu| \geq k\sigma) \leq \frac{1}{k^2}$$

3. 离散分布 (Discrete)

人数头、抛硬币、服务器请求

伯努利 (Bernoulli) / 二项 (Binomial)

- 场景: 抛硬币、点击率 (CTR)。
 $P(k) = \binom{n}{k} p^k (1-p)^{n-k}$

泊松分布 (Poisson)

$$P(k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

- 场景: 单位时间内到达的请求数、Bug 数。
- 特征: 均值 = 方差 = λ 。事件是独立的。

几何分布 (Geometric)

- 场景: 第 k 次尝试才成功的概率 (重试机制)。

多项分布 (Multinomial) 二项分布的推广。
 n 次试验, k 个类别。场景: NLP 词频统计 (*Bag of Words*)。

4. 连续分布 (Continuous)

身高、误差、延迟、寿命

高斯/正态分布 (Gaussian)

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- 中心极限定理 (CLT): 独立随机变量之和, 最终都趋向于正态分布。这是误差通常服从正态分布的原因。
- 法则: $\pm 1\sigma$ (68%), $\pm 2\sigma$ (95%), $\pm 3\sigma$ (99.7%)。

指数分布 (Exponential) $f(x) = \lambda e^{-\lambda x}$

- 场景: 事件发生的时间间隔、粒子寿命。
- 无记忆性: 无论等了多久, 下一秒发生的概率不变。

对数正态 (Log-Normal) 若 $\ln X \sim N(\mu, \sigma^2)$ 。描述乘性误差累积。场景: 用户收入、API 响应延迟 (长尾分布)。

5. 统计推断 (Inference)

从数据反推规律

参数估计

- MLE (极大似然): 找参数 θ , 让数据出现的概率最大。 $\max P(D|\theta)$ 。
- MAP (最大后验): 加上先验知识。 $\max P(D|\theta)P(\theta)$ 。

假设检验 (Hypothesis Testing)

- H0 (零假设): 世界是无聊的 (无效果/无差异)。
- P-value: 在 H0 成立下, 看到当前数据 (或更极端) 的概率。P 越小, H0 越不可信。
- Type I/II 错误: 误报 (False Positive) vs 漏报。

A/B 测试核心 双样本均值检验 (T-test / Z-test)。判断两个版本的转化率是否有显著差异。

6. 随机过程与 MCMC

动态的随机性

马尔可夫链 (Markov Chain) 无记忆性:
 $P(X_{t+1}|X_t, \dots) = P(X_{t+1}|X_t)$ 。应用: PageRank, 文本生成。

蒙特卡洛 (Monte Carlo) 用随机采样来解决计算问题 (如求积分、算 π)。

MCMC (马尔可夫链蒙特卡洛) 构造一个马链, 使其平稳分布等于目标分布。

- 应用: 贝叶斯推断中, 当后验概率 $P(\theta|D)$ 无法直接计算积分时, 用 MCMC 采样近似。

7. Python / Scipy 实战

代码工具箱

- 分布: `scipy.stats.norm(loc, scale)`
- 采样: `rvs(size=1000)`
- 概率密度: `pdf(x)` / 累积: `cdf(x)`
- 检验: `ttest_ind(a, b)`

8. 信息论视角 (Information)

概率与编码的桥梁

熵 (Entropy) 衡量分布的混乱程度/信息量:

$$H(X) = -\sum p(x) \log p(x)$$

KL 散度 (相对熵) 衡量两个分布 P 和 Q 的距离:

$$D_{KL}(P||Q) = \sum P(x) \log \frac{P(x)}{Q(x)}$$

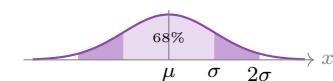
应用: 机器学习中的 Cross-Entropy Loss 就是它。

互信息 (Mutual Information)

$I(X; Y) = H(X) - H(X|Y)$ 。含义: 知道 Y 后, 消除了多少关于 X 的不确定性。

核心直觉 (Intuition)

“世界不是非黑即白, 而是灰度的。”



概率论是程序员处理不完全信息的操作系统。

三大核心视角

• **频率 vs 贝叶斯:** 频率派认为参数是固定的 (真理存在); 贝叶斯派认为参数也是随机变量 (我们对真理的信心)。大数据时代, 贝叶斯思维 (Prior + Data → Posterior) 更符合 AI 的学习过程。

• **分布即形状:** 不必死记公式。泊松是“稀疏事件”, 高斯是“误差累积”, 指数是“无记忆等待”, 幕律是“赢家通吃”。看到数据形状, 就选什么分布。

• **相关 ≠ 因果:** 统计只能告诉你 A 和 B 同时发生 (Correlation), 不能告诉你 A 导致了 B (Causality)。这是数据分析最大的陷阱。

在上帝掷骰子的宇宙里, 概率是我们唯一的导航仪。

图论 Graph Theory Cheat Sheet

The Structure of Relationships: From Social Networks to Dependency Trees

场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 任务调度/依赖 → 拓扑排序 (Topological Sort)
- 地图导航/路由 → 最短路径 (Dijkstra / A*)
- 社交推荐/六度 → 随机游走 / PageRank
- 死锁检测 → 环检测 (Cycle Detection)
- 网络最大流量 → 最大流最小割 (Max-Flow)
- 聚类/社群发现 → 谱聚类 (Spectral)

1. 基础定义 (Foundations)

点与线的抽象世界

图的构成 $G = (V, E)$

- V (Vertex): 节点 (实体)。 $|V| = n$ 。
- E (Edge): 边 (关系)。 $|E| = m$ 。
- 度 (Degree): 连接边的数量。 $\sum \deg(v) = 2m$ (握手定理)。

存储方式

- 邻接矩阵: $A_{ij} = 1$ (稠密图, $O(1)$ 查询)。
- 邻接表: $\text{List}[u] = [v_1, v_2]$ (稀疏图, 省空间)。

连通性 (Connectivity)

- 强连通 (SCC): 有向图中任意两点互通。
- 弱连通: 忽略方向后连通。

2. 遍历与搜索 (Traversal)

不重不漏地访问每个角落

DFS (深度优先)

- 栈/递归: 一条路走到黑, 撞墙回溯。
- 应用: 连通性、拓扑排序、环检测。
- 时间: $O(n + m)$ 。

BFS (广度优先)

- 队列: 层层推进, 像水波纹。
- 应用: 无权图最短路径、Web 爬虫。
- 性质: 首次访问即最短距离 (无权)。

3. 核心算法 (Algorithms)

图论皇冠上的明珠

最短路径 (Shortest Path)

- Dijkstra: 贪心 + 优先队列。 $O(m \log n)$ 。限制: 边权非负。
- Bellman-Ford: 动态规划。 $O(mn)$ 。能力: 处理负权边, 检测负环。
- Floyd-Warshall: 所有点对。 $O(n^3)$ 。

最小生成树 (MST) 连接所有点且总权值最小 (造价最低的网络)。

- Prim: 从一点出发长树 (切分定理)。
- Kruskal: 并查集加边 (贪心)。

拓扑排序 (Topo Sort)

- 对象: DAG (有向无环图)。
- 算法: 不断移除入度为 0 的点。
- 应用: 编译依赖 (Makefile), 任务流。

4. 谱图论 (Spectral Graph)

用线性代数研究图结构

拉普拉斯矩阵 (Laplacian)

$$L = D - A$$

(D: 度矩阵, A: 邻接矩阵)。

代数连通度 (Fiedler Value) L 的第二小特征值 λ_2 。

- $\lambda_2 > 0 \iff$ 图连通。
- λ_2 越大 \rightarrow 图越难被切断 (鲁棒)。
- Cheeger 不等式: λ_2 约束了瓶颈宽窄。

谱聚类 (Spectral Clustering) 利用特征向量将图嵌入低维空间, 再 K-Means。原理: 相似的点在特征空间距离近。

图的嵌入与表示

- 图同构: 判断两图结构是否相同 (NP 问题)。
- 谱半径: 最大特征值, 反映扩散速度。

5. 网络流 (Network Flow)

管道、运输与瓶颈

最大流最小割定理

$$\text{MaxFlow} = \text{MinCut}$$

- 最大流: 源点 s 到汇点 t 能通过的最大流量。
- 最小割: 割断最少 (容量) 的边使 $s - t$ 不连通。
- 直觉: 系统的最大产能取决于瓶颈。

二分图匹配 转化为最大流问题 (添加源汇点)。场景: 任务分配、相亲配对。

6. 复杂网络 (Complex Networks)

真实世界的网络特征

小世界效应 (Small World)

- 大部分节点不相邻, 但任意两点平均路径短。
- 六度分隔: 任何人之间只隔约 6 个人。

无标度网络 (Scale-Free)

- 幂律分布: $P(k) \sim k^{-\gamma}$ 。
- Hub 节点: 少数节点拥有海量连接 (KOL)。
- 鲁棒性: 随机攻击不怕, 针对 Hub 攻击脆弱。

PageRank

$$PR(u) = \frac{1-d}{N} + d \sum_{v \in B(u)} \frac{PR(v)}{L(v)}$$

核心: 被重要节点引用的节点更重要 (随机游走稳态)。

7. Python / NetworkX 实战

代码工具箱

- `G = nx.Graph(); G.add_edge(1, 2)`
- 最短路: `nx.shortest_path(G, src, dst)`
- 连通分量: `nx.connected_components(G)`
- 绘图: `nx.draw(G, with_labels=True)`
- PageRank: `nx.pagerank(G)`

8. 高阶理论 (Advanced)

图论的深水区

图神经网络 (GNN)

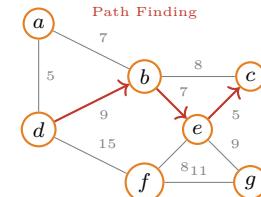
- 消息传递: 聚合邻居信息更新节点 Embedding。
- $h_v^{(k)} = \sigma(W \cdot \text{AGG}(\{h_u^{(k-1)}\}))$
- 应用: 推荐系统、分子性质预测。

随机图 (Random Graph) Erdos-Renyi 模型 $G(n, p)$ 。研究相变 (Phase Transition)。

平面图 (Planar Graph) 能画在平面上且边不交叉的图。欧拉公式: $V - E + F = 2$ 。

核心直觉 (Intuition)

“关系比实体更重要。”



图论是研究结构的科学。在图论的视角下, 个体是谁并不重要, 重要的是它和谁连接, 以及它在网络中的位置。

三大核心视角

- 路径与流: 信息、能量或物质如何在网络中流动? 最短路径是效率, 最大流是容量。
- 中心性 (Centrality): 谁是网络的中心? 是连接最多的 (Degree), 还是处于交通要道的 (Betweenness), 还是被大咖关注的 (PageRank)?
- 谱分析 (Spectrum): 图的“指纹”藏在矩阵的特征值里。听图的“声音”(频谱), 就能知道它的结构特征 (如同步能力、聚类倾向)。

万物互联, 而图论是理解连接的语法。

信息论 Information Theory Cheat Sheet

The Physics of Data: From Entropy to Channel Capacity

场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 数据压缩 → 熵 / 霍夫曼编码
- 模型评估 → 交叉熵 / KL 散度
- 特征选择 → 互信息 (Mutual Info)
- 通信带宽 → 信道容量 (Shannon Limit)
- 决策树分裂 → 信息增益 (Information Gain)

1. 熵与不确定性 (Entropy)

信息的度量单位

香农熵 (Shannon Entropy) 衡量分布 P 的混乱程度 (平均惊奇度)。

$$H(X) = - \sum_x p(x) \log_2 p(x)$$

- 单位: Bit (底数为 2), Nat (底数为 e)。
- 性质: $H(X) \geq 0$ 。均匀分布时最大。

联合熵与条件熵

- $H(X, Y) = H(X) + H(Y|X)$ (链式法则)
- $H(Y|X)$: 知道 X 后 Y 的不确定性

2. 距离与相关性 (Measures)

分布之间的“距离”

KL 散度 (Relative Entropy) 用分布 Q 拟合真实分布 P 时的信息损失。

$$D_{KL}(P||Q) = \sum p(x) \log \frac{p(x)}{q(x)}$$

- 非负: $D_{KL} \geq 0$ (Gibbs 不等式)。
- 非对称: $D_{KL}(P||Q) \neq D_{KL}(Q||P)$ 。

互信息 (Mutual Information) X 和 Y 共享的信息量。

$$I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

直觉: 知道 X 后, Y 的不确定性减少了多少。

3. 信源编码 (Source Coding)

如何压得更小 (无损压缩)

香农第一定理 最优码长 L 的下界是熵 $H(X)$ 。

$$L \geq H(X)$$

霍夫曼编码 (Huffman)

- 策略: 概率大的用短码, 概率小的用长码。
- 算法: 自底向上构建二叉树 (贪心)。
- 最优性: 前缀码中最优。
- 平均码长: $H(X) \leq L < H(X) + 1$ 。

算术编码 (Arithmetic) 将整个消息映射为 $[0, 1]$ 区间内的一个小数。优势: 突破整数位限制, 近逼熵极限。

Lempel-Ziv 编码 动态构建字典, 适用于未知分布。应用: ZIP, GZIP, PNG。

渐进等价性 (AEP) 长序列的典型概率 $\approx 2^{-nH(X)}$, 大多数序列都是典型序列。

4. 信道编码 (Channel Coding)

如何在噪声中传输 (纠错)

香农第二定理 只要传输速率 $R < C$ (信道容量), 就存在一种编码能实现任意低的误码率。

信道容量 (Capacity)

$$C = \max_{p(x)} I(X; Y)$$

AWGN 信道公式:

$$C = B \log_2(1 + \text{SNR})$$

(B : 带宽, SNR: 信噪比)。启示: 增加带宽或信噪比都能提升网速。

纠错码

- 汉明码: 单比特纠错。
- LDPC/Turbo: 接近香农极限的实用码。
- Reed-Solomon: 纠突发错误 (光盘/二维码)。

5. 连续熵 (Differential)

连续变量的世界

微分熵

$$h(X) = - \int f(x) \log f(x) dx$$

- 注意: 可以是负数! (不再代表比特数)。
- 最大熵: 给定方差 σ^2 , 高斯分布熵最大。

AEP (渐进均分性) 大数定律的信息论版本。绝大多数序列都是“典型序列”, 其概率 $\approx 2^{-nH(X)}$ 。

6. 机器学习视角 (ML View)

AI 的损失函数之源

交叉熵 (Cross Entropy)

$$\begin{aligned} H(P, Q) &= H(P) + D_{KL}(P||Q) \\ &= - \sum p(x) \log q(x) \end{aligned}$$

- 分类任务: 真实标签 P 是 One-hot, 预测 Q 是 Softmax。
- 最小化交叉熵 \iff 最小化 KL 散度。

最大熵原理 (MaxEnt) 在满足约束的条件下, 选择熵最大的分布 (最无偏见的假设)。例子: 没有任何信息时, 假设为均匀分布。

信息增益 (Information Gain) 决策树分裂标准: $IG = H(Y) - H(Y|X)$ 。选择信息增益最大的特征进行分裂。

7. Python / Scipy 实战

代码工具箱

- from scipy.stats import entropy
- 熵: `entropy(pk, base=2)`
- KL 散度: `entropy(pk, qk)`
- 互信息: `sklearn.metrics.mutual_info_score`

8. 高阶概念 (Advanced)

前沿探索

信息瓶颈 (Information Bottleneck) 深度学习理论: 压缩输入 $I(X; T)$, 保留输出 $I(T; Y)$ 。

科尔莫哥洛夫复杂度 描述对象所需的最短程序长度 (熵的算法版本)。不可计算, 但可近似估计。

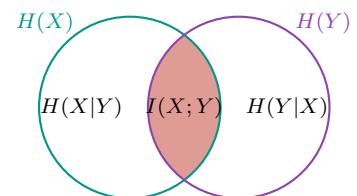
数据处理不等式 如果 $X \rightarrow Y \rightarrow Z$ 构成马尔可夫链:

$$I(X; Z) \leq I(X; Y)$$

信息只会减少, 不会增加。

核心直觉 (Intuition)

“比特是宇宙的原子。”



$$H(X, Y) = H(X) + H(Y) - I(X; Y)$$

信息论告诉我们, 信息是可以被精确度量的物理量, 就像质量和能量一样。

三大核心视角

- 惊奇度 (Surprise): 太阳从东边升起不包含信息 (概率 = 1, 熵 = 0)。太阳没升起才包含巨大信息。概率越小, 信息量越大。
- 相关性即共享: 互信息 $I(X; Y)$ 衡量变量共享比特数。独立则互信息为 0。
- 编码即理解: 如果你能把数据压缩得很小, 说明你理解了它的规律。压缩率是智能的度量。

宇宙由原子组成, 但其本质是比特。

复变函数 Complex Analysis Cheat Sheet

Complex Analysis: The Art of Rotation from Euler's Formula to Conformal Mapping

场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 处理震荡/周期 → 欧拉公式 (旋转)
- 解微分积分 → 留数定理 (数奇点)
- 不规则边界流体 → 保角映射 (变圆)
- 分析系统发散 → 极点/零点 (稳定性)
- 解微分方程 → 拉普拉斯变换 (代数)
- 快速乘法/卷积 → FFT (单位根)

1. 基础 (Foundations)

把“震荡”变为“旋转”

定义与表示

- 形式: $z = x + iy = re^{i\theta}$
- 核心: 模相乘 ($r_1 r_2$), 角相加 ($\theta_1 + \theta_2$)

上帝公式 (Euler's Formula)

$$e^{i\theta} = \cos \theta + i \sin \theta$$

- 物理意义: 单位圆上的旋转算子
- 旋转操作: $z \cdot e^{i\phi}$ = 逆时针转 ϕ 度

根与多值性

- 单位根: $w_k = e^{i\frac{2\pi k}{n}}$ (FFT 基础)
- 对数: $\ln z = \ln r + i(\theta + 2k\pi)$

2. 全纯与解析 (Analyticity)

比实数微积分更“完美”的平滑

柯西-黎曼方程 (C-R) $f = u + iv$ 可导 $\iff u_x = v_y, u_y = -v_x$

全纯性质 (Holomorphic)

- 无穷可导: 一阶可导 → 无穷阶
- 保角性: 保持局部角度不变 (流体)
- 调和性: $\Delta u = 0$ (天然满足拉普拉斯方程)
- 刘维尔定理: 有界整函数必为常数。

3. 积分与留数 (Integration)

积分与路径无关, 只看“奇点”

柯西积分定理 (Cauchy Thm)

$$\oint_C f(z) dz = 0 \quad (\text{解析区内闭路积分})$$

柯西公式 (The Scanner)

边界值定内部值:

$$f(a) = \frac{1}{2\pi i} \oint \frac{f(z)}{z-a} dz$$

留数定理 (Residue Thm)

围住 n 个奇点 z_k :

$$\oint_C f(z) dz = 2\pi i \sum \text{Res}(f, z_k)$$

- 实积分: $\int_{-\infty}^{\infty} \rightarrow$ 上半平面大圆弧闭合。
- 约当引理: 若 $f \rightarrow 0$, 则 $\int f(z) e^{iaz} dz \rightarrow 0$ 。
- 柯西主值: 奇点在路径上时, 挖去奇点计算。

4. 变换与映射 (Transforms)

换个空间解决问题: 时域 \rightarrow 频域

傅里叶变换 (Fourier)

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt$$

本质: 信号与螺旋线 $e^{-i\omega t}$ 的内积

拉普拉斯变换 (Laplace)

$$F(s) = \int_0^{\infty} f(t) e^{-st} dt$$

本质: 引入衰减因子 σ 处理发散信号

Z 变换 (Discrete) $X(z) = \sum x[n] z^{-n}$ (极点在单位圆内稳定)

保角映射 (Conformal)

- 茹科夫斯基: 圆 \rightarrow 机翼; 默比乌斯: 圆 \rightarrow 直线
- 应用: 复杂几何场 (热/电/流) \rightarrow 映射为单位圆

5. 工程应用 (Engineering)

从数学理论到物理实战

信号处理 (DSP)

- 相位: $\arg(z)$ 即波形延迟; 群延迟 $-\frac{d\phi}{d\omega}$
- 滤波: 极点在单位圆内 \rightarrow 稳定 (BIBO)
- 希尔伯特变换: 构造解析信号 $f(t) + i\mathcal{H}[f]$ 。

电路与控制 (Circuits)

- 相量: $V_m \cos(\omega t + \phi) \rightarrow V_m e^{j\phi}$
- 奈奎斯特: $1 + G(s)H(s)$ 绕原点圈数 \rightarrow 稳定性
- 波特图: 利用 $H(j\omega)$ 分析幅频/相频特性。
- 史密斯圆图: 反射系数 Γ 与阻抗 Z 的保角映射。

物理学 (Physics)

- 流体: 复势 $w = \phi + i\psi$, 速度 $V = \overline{w'(z)}$
- 量子: $\psi = Re^{i(kx - \omega t)}$ 。薛定谔方程核心 $i\hbar\partial_t$ 。
- 升力: 库塔-儒可夫斯基定理计算机翼升力。
- 流体复势: $w(z) = \phi + i\psi$, $V = \overline{w'(z)}$

7. 高阶视界 (Riemann Surfaces)

破解“多值函数”的拓扑迷宫

多值性的困境 \sqrt{z} 或 $\ln z$ 绕原点一圈回不到起点 (变负或加 $2\pi i$)。复平面上的这些函数是不连续的。

黎曼的解决方案

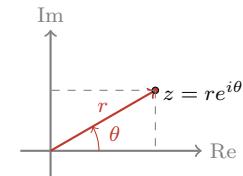
- 螺旋楼梯: 不要把平面看作纸, 而是一层层楼。
- 支割线 (Cut): 楼梯边缘, 跨过即换层。
- 亏格 (Genus): 曲面上“洞”的数量 \rightarrow 拓扑学。

分形 (Mandelbrot) 复平面上最复杂的图形 (混沌边缘):

$$z_{n+1} = z_n^2 + c$$

核心直觉 (Intuition)

“复数是连接几何与代数的虫洞。”



如果说实数是一根直线, 复数就是一个平面。引入复数不是为了把问题变复杂, 而是为了利用二维空间的对称性来降维打击一维问题。

四大核心视角

- 旋转即乘法:** $e^{i\theta}$ 是永不停歇的齿轮。所有波动本质上都是圆周运动的投影。
- 解析即全息 (Holographic):** 只要知道了一个小邻域的信息 (泰勒级数), 理论上就能推导出整个定义域的函数值。(解析延拓)
- 物理的统一性:** 量子力学的波函数、流体力学的势流、交流电的相量, 底层共享同一套数学结构。

现实世界是实数的, 但理解它的钥匙往往是复数的。

傅里叶变换 Fourier Transforms Cheat Sheet

The Prism of Mathematics: Decomposing the Universe into Waves

场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 音频降噪/均衡 → 频域滤波 (Low/High Pass)
- 图像压缩 (JPEG) → DCT (离散余弦变换)
- 大整数乘法 → 卷积定理 + FFT
- 解微分方程 → 频域代数化

1. 连续傅里叶 (CTFT)

时域与频域的桥梁

正变换 (Analysis)

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t} dt$$

含义: 信号 $f(t)$ 中包含多少频率为 ω 的成分。

逆变换 (Synthesis)

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega)e^{i\omega t} d\omega$$

狄拉克 δ 函数

- $\delta(t) \iff 1$ (脉冲包含所有频率)
- $e^{i\omega_0 t} \iff 2\pi\delta(\omega - \omega_0)$

3. 离散傅里叶 (DFT)

计算机里的傅里叶

定义 对于长度为 N 的序列 $x[n]$:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-i\frac{2\pi}{N}kn}$$

k : 频率索引, $W_N = e^{-i\frac{2\pi}{N}}$ 为 N 次单位根。

采样定理 (Nyquist) 采样频率必须大于信号最高频率的 2 倍 ($f_s > 2f_{max}$)。

频谱泄露 (Leakage) 信号周期不是 N 的整数倍时, 能量会泄露。解法: 加窗 (Hamming)。

2. 核心性质 (Properties)

在频域操作更简单

卷积定理 (Convolution) 时域卷积 = 频域相乘

$$f(t) * g(t) \iff F(\omega) \cdot G(\omega)$$

威力: $O(N^2)$ 的卷积变为 $O(N \log N)$ 的乘法。

微分性质

$$f'(t) \iff i\omega F(\omega)$$

威力: 微分方程变为代数方程。

帕塞瓦尔定理 (Parseval) 能量守恒: 时域总能量 = 频域总能量。

$$\int |f(t)|^2 dt = \frac{1}{2\pi} \int |F(\omega)|^2 d\omega$$

其他重要性质

- 平移: $f(t - t_0) \iff e^{-i\omega t_0} F(\omega)$
- 调制: $f(t)e^{i\omega_0 t} \iff F(\omega - \omega_0)$
- 尺度: $f(at) \iff \frac{1}{|a|} F(\frac{\omega}{a})$

4. 快速傅里叶变换 (FFT)

20 世纪最重要的算法

Cooley-Tukey 算法 利用 W_N^k 的对称性和周期性, 分治求解。

$$DFT(N) \rightarrow 2 \times DFT(N/2) + O(N)$$

- 复杂度: $O(N^2) \rightarrow O(N \log N)$ 。
- 蝶形运算: 基本计算单元。

应用: 多项式乘法 两个 N 次多项式相乘。

- 系数转点值 (FFT)。
- 点值相乘 ($O(N)$)。
- 点值转系数 (IFFT)。

5. 相关变换 (Relatives)

傅里叶家族

短时傅里叶 (STFT) 加窗, 看局部频率随时间的变化 (声纹图)。

$$X(m, \omega) = \sum x[n]w[n-m]e^{-i\omega n}$$

缺陷: 窗口大小固定, 时频分辨率不可兼得。

小波变换 (Wavelet) 用不同尺度的波 (小波) 去拟合。优势: 高频处时间分辨率高, 低频处频率分辨率高。

离散余弦变换 (DCT) 只用实数余弦 (丢掉虚部)。应用: JPEG 压缩 (能量集中在低频)。

希尔伯特变换 (Hilbert) 计算信号的解析形式, 提取瞬时包络 (幅度和相位)。应用: FM/AM 调制解调。

6. 物理直觉 (Physics)

自然界的语言

不确定性原理

$$\Delta t \cdot \Delta \omega \geq \frac{1}{2}$$

信号在时域越窄 (短), 在频域就越宽 (Γ)。量子力学: $\Delta x \Delta p \geq \hbar/2$ 本质上就是傅里叶变换性质。

衍射与成像 透镜的焦平面就是物体的傅里叶变换平面。晶体衍射图样就是晶格的倒易空间 (频域)。

波动方程 电磁波、声波、量子波函数都是傅里叶模式的叠加。

7. Python / Numpy 实战

代码工具箱

- `np.fft.fft(x)`: 计算 FFT。
- `np.fft.fftfreq(n, d)`: 获取频率轴坐标。
- `np.fft.fftshift(X)`: 将零频移到中心。
- 滤波: `X[high_freqs] = 0; ifft(X)`。

8. 高阶技巧 (Advanced)

魔改与优化

分数阶傅里叶 (FRFT) 旋转时频平面 α 角度。应用: 线性调频信号 (Chirp) 处理。

数论变换 (NTT) 在有限域 $GF(p)$ 上做 FFT。优势: 无浮点误差。应用: 大数乘法库, 密码学。质数 p 需满足 $p = k \cdot 2^n + 1$ 形式。

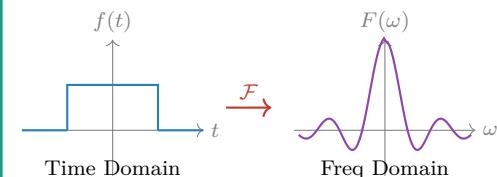
压缩感知 (Compressed Sensing) 信号在频域稀疏时, 可以低于奈奎斯特频率采样并恢复。

$$\min \|x\|_1 \quad \text{s.t.} \quad y = \Phi x$$

应用: MRI 成像加速、单像素相机。

核心直觉 (Intuition)

“万物皆震荡。”



傅里叶变换不仅仅是一个数学公式, 它是一种世界观。

三大核心视角

正交分解: 正弦波 $e^{i\omega t}$ 是函数空间的一组正交基。傅里叶变换就是把信号投影到这组基上, 求坐标 (系数)。

全局性: 频域上的一个点, 对应时域上的无限长正弦波。修改频域的一点, 会影响时域的所有时刻。

对偶性: 时域的卷积等于频域的乘法。时域的离散对应频域的周期。时域的压缩对应频域的扩展。两个世界完美对称。

给我足够多的正弦波, 我可以构造出整个宇宙。

拉普拉斯与 Z 变换 Laplace & Z Transforms Cheat Sheet

System Analysis: Stability, Control, and Feedback

场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 解微分方程 (ODE) → 拉普拉斯变换
- 解差分方程 (递推) → Z 变换
- 控制系统设计 (PID) → 传递函数 $H(s)$
- 数字滤波器 (DSP) → 传递函数 $H(z)$
- 系统稳定性分析 → 极点位置 (Poles)

1. 拉普拉斯变换 (Laplace)

连续系统的代数化

定义 (单边)

$$F(s) = \int_0^{\infty} f(t)e^{-st} dt, \quad s = \sigma + i\omega$$

- 收敛域 (ROC): 使得积分收敛的 s 区域 ($\text{Re}(s) > \sigma_0$)。
- 本质: 傅里叶变换 + 指数衰减因子 $e^{-\sigma t}$ (处理发散信号)。

核心性质

- 微分: $f'(t) \iff sF(s) - f(0)$ 。
- 积分: $\int f(t)dt \iff \frac{1}{s}F(s)$ 。
- 终值定理: $\lim_{t \rightarrow \infty} f(t) = \lim_{s \rightarrow 0} sF(s)$ 。

2. Z 变换 (Z-Transform)

离散系统的代数化

定义 (单边)

$$X(z) = \sum_{n=0}^{\infty} x[n]z^{-n}, \quad z = re^{i\omega}$$

- 关系: $z = e^{sT}$ (从 s 平面映射到 z 平面)。
- 单位圆: 对应 s 平面的虚轴。

核心性质

- 移位: $x[n - k] \iff z^{-k}X(z)$ (延迟 k 拍)。
- 卷积: $x[n] * h[n] \iff X(z)H(z)$ 。

3. 传递函数与稳定性 (Stability)

极点决定命运

传递函数 (Transfer Function)

$$H(s) = \frac{Y(s)}{X(s)} = \frac{N(s)}{D(s)} = K \frac{\prod(s - z_i)}{\prod(s - p_i)}$$

- 零点 (Zeros, z_i): 让输出为 0 的点 (分子根)。
- 极点 (Poles, p_i): 让输出无穷大的点 (分母根)。

稳定性判据 系统冲激响应 $h(t)$ 是否收敛?

- S 域 (连续): 所有极点实部 < 0 (左半平面)。
- Z 域 (离散): 所有极点模 < 1 (单位圆内)。

直觉: 极点是系统的“本征模态”，实部正意味着指数增长爆炸。

4. 控制理论基础 (Control)

让系统听话

反馈控制 (Feedback) 闭环传递函数:

$$T(s) = \frac{G(s)}{1 + G(s)H(s)}$$

目标: 通过设计控制器 $H(s)$ 移动闭环极点到左半平面。

频率响应 (Bode Plot)

- 幅频: $20 \log |H(i\omega)|$ (dB)。
- 相频: $\angle H(i\omega)$ 。
- 带宽: 增益下降 3dB 的频率。

PID 控制器

$$C(s) = K_p + \frac{K_i}{s} + K_d s$$

- K_p : 比例 (现在的偏差)
- K_i : 积分 (历史的累积)
- K_d : 微分 (未来的趋势)

奈奎斯特判据 (Nyquist) 通过开环传递函数的频率响应判断闭环稳定性。

5. 常用变换对 (Pairs)

查表神器

信号	Laplace ($F(s)$)	Z ($X(z)$)
$\delta(t), \delta[n]$	1	1
$u(t), u[n]$	$1/s$	$\frac{z}{z-1}$
e^{-at}, a^n	$\frac{1}{s+a}$	$\frac{z}{z-a}$
$\sin(\omega t)$	$\frac{\omega}{s^2 + \omega^2}$	$\frac{z \sin \omega}{z^2 - 2z \cos \omega + 1}$
t, n	$1/s^2$	$\frac{z}{(z-1)^2}$

注: Z 变换中通常 a 对应 e^{-aT} 。

6. 滤波器设计 (Filters)

雕刻频率

模拟滤波器

- 低通: $H(s) = \frac{\omega_c}{s + \omega_c}$ (一阶)。
- 巴特沃斯: 通带最平坦。

数字滤波器

- IIR (无限冲激): 有反馈, 有极点, 可能不稳定。
- FIR (有限冲激): 无反馈, 全零点, 永远稳定, 线性相位。
- 双线性变换: $s = \frac{2}{T} \frac{1-z^{-1}}{1+z^{-1}}$ (模拟转数字)。

窗函数设计法 FIR 滤波器设计: 理想频率响应 \rightarrow IFFT \rightarrow 加窗。常用窗: Hamming, Hanning, Blackman。

切比雪夫滤波器 等波纹特性: 通带或阻带有均匀的纹波。优势: 过渡带更陡峭, 阶数更低。

7. Python / Scipy 实战

代码工具箱

- `sys = scipy.signal.lti([num], [den])`
- 阶跃响应: `step(sys)`
- 波特图: `bode(sys)`
- 极零点: `sys.zeros, sys.poles`
- Z 变换: `scipy.signal.dlsim`

8. 高阶视角 (Advanced)

深入系统本质

状态空间 (State Space) 现代控制理论的核心。

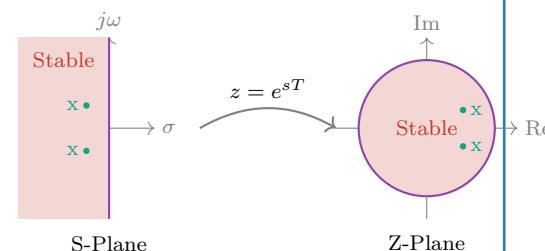
$$\dot{x} = Ax + Bu, \quad y = Cx + Du$$

- A: 系统矩阵 (特征值即极点)。
- 可控性: 能否把状态推到任意值?
- 可观性: 能否通过输出推断状态?

根轨迹 (Root Locus) 随增益 K 变化, 闭环极点在复平面上的移动轨迹。

核心直觉 (Intuition)

“S 平面是系统的地形图。”



拉普拉斯变换将随时间变化的微分方程变成了静态的代数方程。

三大核心视角

- 极点即寿命: 极点的实部决定了信号是衰减 (稳定) 还是增长 (爆炸)。极点离虚轴越远, 衰减/增长越快。
- 虚部即频率: 极点的虚部决定了震荡的频率。共轭极点对产生正弦震荡。
- 映射即离散化: Z 变换就是把 S 平面的左半平面“卷”成了一个单位圆。虚轴变成了圆周。这就是为什么离散系统的稳定性看单位圆。

给我极点的位置, 我就能预言系统的未来。

数值分析 Numerical Analysis Cheat Sheet

The Art of Approximation: Solving the Unsolvable

场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 解非线性方程 → 牛顿法 (Newton's)
- 解大型线性方程组 → 迭代法 (GMRES / CG)
- 函数拟合/插值 → 样条插值 (Spline)
- 数值积分 → 高斯求积 (Gaussian Quad)
- 解微分方程 → 龙格-库塔 (Runge-Kutta)
- 特征值计算 → 幂法 / QR 算法

1. 误差与浮点数 (Errors)

计算机不是数学家

IEEE 754 浮点数

$$x = (-1)^s \times (1.m) \times 2^{e-127}$$

- 机器精度 (ϵ): $1 + \epsilon \neq 1$ 的最小正数 ($\approx 10^{-16}$ for double)。
- 大数吃小数: $10^{16} + 1 = 10^{16}$ 。

稳定性 (Stability) 算法对输入微小扰动的敏感度。条件数: $\kappa = \frac{|\delta \text{Output}|}{|\delta \text{Input}|}$ 。

2. 根查找 (Root Finding)

求 $f(x) = 0$

二分法 (Bisection) 区间减半。绝对收敛, 但慢 (线性)。

牛顿法 (Newton-Raphson) 利用切线逼近。

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

- 收敛: 二次收敛 (每次有效位翻倍)。
- 风险: 初值选不好会发散。

割线法 (Secant) 不需要导数, 用两点的割线逼近。

$$x_{n+1} = x_n - f(x_n) \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$$

3. 线性代数求解 (Linear Sys)

解 $Ax = b$ (N 很大)

直接法 (Direct)

- LU 分解: $O(N^3)$ 。适合 $N < 10000$ 。
- Cholesky: 针对正定矩阵, 快一倍。

迭代法 (Iterative)

- 适合稀疏矩阵。
- Jacobi / Gauss-Seidel: 简单但慢。
 - 共轭梯度 (CG): 针对对称正定, 收敛快。
 - GMRES: 针对非对称矩阵。

条件数 (Condition Number) $\kappa(A) = \|A\| \|A^{-1}\|$ 。 κ 很大 \implies 矩阵病态 (Ill-conditioned), 解不可信。

- Hilbert 矩阵: $\kappa \sim 10^{13}$ (经典病态矩阵)
- 单位矩阵: $\kappa = 1$ (完美条件)

预条件 (Preconditioning) 改变矩阵结构降低条件数: 解 $M^{-1}Ax = M^{-1}b$ 。目标: 让 $M^{-1}A$ 接近单位矩阵。常用方法: Jacobi 预条件 ($M = \text{diag}(A)$)、ILU 分解。

4. 插值与拟合 (Fitting)

连接数据点

插值 (Interpolation) 必须经过所有点。

- 拉格朗日: 多项式震荡 (龙格现象)。
- 样条 (Spline): 分段低阶多项式, 平滑且稳定。

最小二乘拟合 (Least Squares) 不经过所有点, 误差最小。

$$\min \|Ax - b\|^2 \implies A^T Ax = A^T b$$

正则化 (Regularization) 防止过拟合: $\min \|Ax - b\|^2 + \lambda \|x\|^2$ (Ridge / Lasso)。 λ 控制模型复杂度。

特征值计算 幂法: 迭代 $x_{n+1} = Ax_n / \|Ax_n\|$ 找最大特征值。QR 算法: 分解 $A = QR$, 收敛至所有特征值。

5. 数值积分与微分 (Calc)

求面积与斜率

数值积分 (Quadrature)

$$\int_a^b f(x) dx \approx \sum w_i f(x_i)$$

- 梯形法则: 线性逼近, $O(h^2)$ 。
- 辛普森法则: 抛物线逼近, $O(h^4)$ 。
- 高斯求积: 选取特定节点, 精度最高。

数值微分 中心差分:

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

注意: h 太小会引发舍入误差, 太大会有截断误差。

6. 微分方程求解 (ODE)

模拟动态系统

欧拉法 (Euler) $y_{n+1} = y_n + h f(t_n, y_n)$ 。一阶精度, 误差累积快, 不稳定。

龙格-库塔 (RK4) 工业标准。四阶精度。

$$y_{n+1} = y_n + \frac{h}{6} (k_1 + 2k_2 + 2k_3 + k_4)$$

探测四个斜率取加权平均。

刚性方程 (Stiff) 时间尺度差异巨大 (既有快变又有慢变)。必须用隐式方法 (Implicit Methods)。例子: 化学反应动力学、电路仿真。

7. Python / Scipy 实战

代码工具箱

- 求根: `scipy.optimize.root(f, x0)`
- 积分: `scipy.integrate.quad(f, a, b)`
- ODE: `scipy.integrate.solve_ivp`
- 插值: `scipy.interpolate.interp1d`

8. 高阶技巧 (Advanced)

加速与优化

蒙特卡洛积分 (Monte Carlo) 高维积分神器。

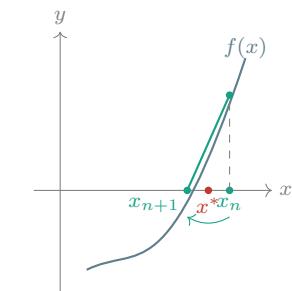
$$\int f(x) dx \approx \frac{V}{N} \sum f(x_i)$$

收敛速度 $O(1/\sqrt{N})$, 与维数无关。

多重网格法 (Multigrid) 在粗网格和细网格间切换, 快速消除不同频率的误差。应用: 求解偏微分方程 (PDE), 计算流体力学 (CFD)。

核心直觉 (Intuition)

“连续是幻觉, 离散才是真实。”



数值分析是数学与计算机妥协的产物。我们放弃了“解析解”的完美, 换取了“数值解”的通用与高效。

三大核心视角

- 迭代即逼近:** 既然无法一步登天, 那就步步为营。只要误差在缩小, 我们终将到达真理的彼岸 (或足够近的地方)。
- 离散即近似:** 微分变成差分, 积分变成求和。微积分的极限定义在计算机里变成了微小的 Δx 。
- 条件即命运:** 如果问题本身是病态的 (条件数大), 再好的算法也救不了。输入的一点点噪音, 会导致输出面目全非。

所有的计算都是近似, 问题在于你允许多少误差。

凸优化 Convex Optimization Cheat Sheet

The Science of Best Choices: From Gradient Descent to Duality

场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 模型训练/拟合 → 梯度下降 (SGD / Adam)
- 资源分配/调度 → 线性规划 (LP)
- 稀疏解/特征选择 → Lasso (L_1 正则)
- 投资组合优化 → 二次规划 (QP)
- 约束满足问题 → 拉格朗日乘子法
- 超大规模求解 → ADMM

1. 凸性基础 (Convexity)

碗状世界，绝无局部陷阱

凸集 (Convex Set) 集合内任意两点的连线仍在集合内。

$$\theta x + (1 - \theta)y \in C, \quad \forall \theta \in [0, 1]$$

凸函数 (Convex Function) 函数图像上方的区域是凸集 (弦在弧之上)。

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y)$$

- 一阶条件: $f(y) \geq f(x) + \nabla f(x)^T(y - x)$ (切线在下)。
- 二阶条件: $\nabla^2 f(x) \succeq 0$ (Hessian 半正定)。

核心性质 局部最优 \Rightarrow 全局最优。这是凸优化最强大的保证。常见凸函数: $e^x, x^2, -\log x, \|x\|_p$ ($p \geq 1$)。

2. 标准形式 (Standard Form)

把问题装进笼子

$$\begin{aligned} \min \quad & f_0(x) \\ \text{s.t.} \quad & f_i(x) \leq 0, \quad i = 1, \dots, m \\ & h_i(x) = 0, \quad i = 1, \dots, p \end{aligned}$$

- f_0, f_i 必须是凸函数。
- h_i 必须是仿射函数 ($Ax + b$)。

3. 对偶理论 (Duality)

从另一个角度看问题

拉格朗日函数 (Lagrangian) 把约束加权放到目标函数里:

$$L(x, \lambda, \nu) = f_0(x) + \sum \lambda_i f_i(x) + \sum \nu_i h_i(x)$$
$$(\lambda_i \geq 0)$$

对偶函数 (Dual Function) $g(\lambda, \nu) = \inf_x L(x, \lambda, \nu)$ 。

- g 永远是凹函数 (即使原问题非凸)。
- 弱对偶: $d^* \leq p^*$ (下界)。
- 强对偶: $d^* = p^*$ (Slater 条件满足时)。

KKT 条件 (KKT Conditions) 最优解的必要条件 (强对偶下也是充分条件):

1. 平稳性: $\nabla_x L = 0$ (梯度为 0)。
2. 原始可行: 满足原约束。
3. 对偶可行: $\lambda_i \geq 0$ 。
4. 互补松弛: $\lambda_i f_i(x) = 0$ (关键!)。

直觉: 要么约束没起作用 ($\lambda = 0$), 要么约束紧绷 ($f = 0$)。

4. 典型问题类 (Classes)

常见的优化模型

线性规划 (LP) 目标和约束都是线性的。解法: 单纯形法 (*Simplex*), 内点法。

二次规划 (QP) 目标是二次的 ($x^T P x$), 约束是线性的。场景: SVM, 最小二乘。

半定规划 (SDP) 变量是矩阵, 约束是半正定 ($X \succeq 0$)。场景: 控制理论, 组合优化松弛。

锥优化 (Cone Programming) 约束是锥: $x \in K$ (如二阶锥 SOC)。统一框架: LP, QP, SDP 都是特殊情况。

5. 求解算法 (Algorithms)

如何一步步走到山谷

梯度下降 (Gradient Descent)

$$x_{k+1} = x_k - \alpha \nabla f(x_k)$$

- SGD: 每次只用一个样本算梯度 (快但抖)。
- Momentum: 加惯性, 冲过鞍点。

牛顿法 (Newton's Method) 利用二阶信息 (Hessian) 调整方向和步长。

$$x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$$

收敛极快 (二次收敛), 但求逆矩阵贵 ($O(n^3)$)。

拟牛顿法 (BFGS) 近似 Hessian 矩阵, 避免求逆。

8. 高阶技巧 (Tricks)

化腐朽为神奇

松弛 (Relaxation) 把难解的非凸问题 (如整数规划) 放宽为凸问题 (如 LP), 求下界。

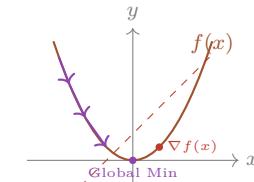
坐标下降 (Coordinate Descent) 每次只优化一个变量, 固定其他。适合 Lasso 等非光滑问题。

ADMM 交替方向乘子法。将大问题拆分为小问题分布式求解。

投影梯度法 (Projected Gradient) 处理约束: $x_{k+1} = \Pi_C(x_k - \alpha \nabla f(x_k))$ 。 Π_C 将点投影回可行域 C 。

核心直觉 (Intuition)

“万物皆可优化。”



凸优化的美妙之处在于确定性。如果你能把工程问题建模为凸问题, 那么你一定能找到最优解, 而且很快。

三大核心视角

- **梯度即方向:** 在黑暗的山脉中, 梯度是唯一的指路明灯, 告诉我们往哪里走下降最快。
- **对偶即定价:** 拉格朗日乘子 λ 不仅仅是数学辅助量, 它是资源的影子价格。如果约束放松一点, 目标函数能优化多少? λ 告诉你答案。
- **稀疏即选择:** L_1 正则化不仅仅是数学技巧, 它是一种哲学: 在纷繁复杂的因素中, 只有少数几个是关键的 (Occam's Razor)。

既然无法遍历所有可能, 那就沿着梯度滑向谷底。

7. Python / CVXPY 实战

声明式编程求解

- ```
• x = cp.Variable(n)
• prob = cp.Problem(cp.Minimize(obj),
 [constraints])
• prob.solve()
• Scipy: minimize(fun, x0, method='BFGS')
```

# 博弈论 Game Theory Cheat Sheet

Strategic Decision Making: From Nash Equilibrium to Mechanism Design

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 多方竞争/合作 → 纳什均衡 (Nash Eq)
- 拍卖/广告竞价 → 机制设计 (VCG)
- 对抗生成网络 → 极小极大 (Minimax)
- 区块链共识 → 拜占庭容错 (BFT)
- 资源调度 → 合作博弈 (Shapley Value)
- 演化策略 → 演化博弈 (ESS)

## 1. 基础定义 (Foundations)

玩家、策略与收益

### 博弈三要素

- Players ( $N$ ): 参与者集合。
- Strategies ( $S_i$ ): 每个玩家的可选动作。
- Payoffs ( $u_i$ ): 收益函数  $u_i(s_1, \dots, s_n)$ 。

**纳什均衡 (Nash Equilibrium)** 策略组合  $s^*$ , 使得没有任何玩家可以通过单方面改变策略来获益。

$$u_i(s_i^*, s_{-i}^*) \geq u_i(s_i, s_{-i}^*) \quad \forall s_i \in S_i$$

含义: 稳定的僵局, 没人想动。

## 2. 经典模型 (Classics)

人性的囚笼

### 囚徒困境 (Prisoner's Dilemma)

|    |          |          |
|----|----------|----------|
|    | 抵赖       | 坦白       |
| 抵赖 | (-1, -1) | (-3, 0)  |
| 坦白 | (0, -3)  | (-2, -2) |

结论: 个体理性导致集体非理性 (帕累托次优)。

**协调博弈 (Coordination)** (左, 左) 和 (右, 右) 都是均衡。问题: 如何达成共识? (Focal Point)

**零和博弈 (Zero-Sum)** 一方收益 = 另一方损失。 $\sum u_i = 0$ 。解法: Minimax 定理。

## 3. 混合策略 (Mixed Strategy)

随机应变

**定义** 玩家以概率分布  $\sigma_i$  选择策略。例: 石头剪刀布, 最优策略是  $(1/3, 1/3, 1/3)$ 。

**纳什存在性定理** 任何有限博弈至少存在一个混合策略纳什均衡。证明用到不动点定理 (Brouwer)。

**期望效用** 混合策略下, 玩家  $i$  的期望效用:

$$u_i(\sigma) = \sum_{s \in S} \left( \prod_{j=1}^n \sigma_j(s_j) \right) u_i(s)$$

直觉: 不可预测性本身是一种武器。

## 4. 动态博弈 (Dynamic)

时间与顺序

**扩展式 (Extensive Form)** 博弈树 (Game Tree)。

- 信息集: 玩家不知道自己在树的哪个节点 (不完全信息)。

**子博弈精炼均衡 (SPE)** 在博弈树的每一个子树上都是纳什均衡。解法: 逆向归纳法 (Backward Induction)。应用: 排除不可信的威胁。

## 5. 机制设计 (Mechanism Design)

反向博弈论: 制定规则

**目标** 设计规则 (游戏), 使得自私玩家的均衡结果符合整体利益 (如社会福利最大化)。

**VCG 机制** 支付 = 给他人造成的“负外部性”。特性: 说真是占优策略 (Truthful)。应用: 谷歌/百度广告竞价 (GSP 是 VCG 的变体)。

**激励兼容 (Incentive Compatible)** 说真是每个玩家的最优策略。Revelation Principle: 任何机制都等价于某个激励兼容机制。直觉: 不需要猜心, 直说即可。

## 6. 合作博弈 (Cooperative)

结盟与分赃

**夏普利值 (Shapley Value)** 公平分配收益的唯一解。玩家  $i$  的收益:

$$\phi_i(v) = \sum_{S \subseteq N \setminus \{i\}} w(|S|) \cdot [v(S \cup \{i\}) - v(S)]$$

其中  $w(|S|) = \frac{|S|!(n-|S|-1)!}{n!}$ 。直觉: 贡献越大, 拿得越多 (边际贡献的期望)。应用: 机器学习特征重要性解释 (SHAP)。

## 9. 高阶前沿 (Advanced)

复杂系统

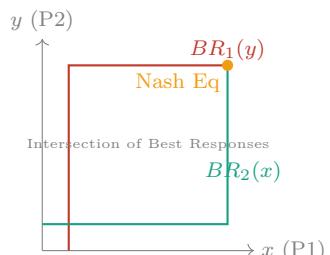
**算法博弈论 (AGT)** 关注均衡的计算复杂度 (PPAD Complete) 和效率损失。无政府代价 (PoA):

$$PoA = \frac{\text{最差均衡的社会福利}}{\text{全局最优社会福利}}$$

**平均场博弈 (Mean Field)** 玩家数量  $N \rightarrow \infty$ 。个体只与“平均分布”交互。

## 核心直觉 (Intuition)

“预测对手的预测。”



博弈论不是教你怎么赢, 而是教你理性对手面前如何不输。

## 三大核心视角

- 均衡即稳定:** 物理系统趋向于能量最低, 博弈系统趋向于纳什均衡。一旦达到均衡, 系统就锁死了, 除非改变规则。
- 承诺即力量:** 在动态博弈中, 限制自己的选择权 (如破釜沉舟) 反而能增加谈判筹码。不可信的威胁是无效的。
- 规则优于策略:** 如果你不喜欢均衡的结果 (如内卷), 不要试图改变玩家的策略 (说教), 而要改变游戏的规则 (机制设计)。

每一个选择, 都是对未来世界的一次投票。

# 抽象代数 Abstract Algebra Cheat Sheet

The Rules of Structure: From Groups to Cryptography

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 数据加密/签名 → 群论 / 椭圆曲线 (ECC)
- 数据校验/纠错 → 有限域 (Galois Field)
- 冗余编码 (RAID) → 多项式环
- 对称性分析 → 置换群
- 类型系统设计 → 范畴论 / 同态
- 零知识证明 → 算术电路 / 多项式承诺

## 1. 群论 (Groups)

对称性的数学描述

**定义**  $(G, \cdot)$  集合  $G$  和运算  $\cdot$  满足:

- 封闭性:  $a \cdot b \in G$ 。
- 结合律:  $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ 。
- 单位元:  $\exists e, a \cdot e = a$ 。
- 逆元:  $\forall a, \exists a^{-1}, a \cdot a^{-1} = e$ 。

### 关键概念

- 阿贝尔群: 交换律成立 ( $ab = ba$ )。
- 循环群: 由一个元素生成 ( $g^k$ )。
- 阶 (Order): 元素个数  $|G|$ 。
- 拉格朗日定理: 子群的阶整除群的阶。
- 正规子群: 左右陪集相等, 商群构造的基础。
- 凯莱定理: 任何群都同构于某个置换群。

## 2. 环与域 (Rings & Fields)

加法与乘法的游乐场

**环 (Ring)** 有加法和乘法 (如整数  $\mathbb{Z}$ )。注意: 乘法不一定有逆元 (不能除)。 整环: 无零因子 ( $ab = 0 \implies a = 0 \vee b = 0$ )。

**域 (Field)** 加减乘除都完备 (如  $\mathbb{R}, \mathbb{C}, \mathbb{Q}$ )。

- 有限域 (Galois Field): 元素个数有限。
- 记作  $GF(p^n)$  或  $\mathbb{F}_{p^n}$ 。
- 计算机最爱:  $GF(2^8)$  (Byte)。

## 3. 同态与同构 (Morphisms)

结构之间的映射

**同态 (Homomorphism)** 保持运算结构的映射  $\phi$ :

$$\phi(a \cdot b) = \phi(a) * \phi(b)$$

直觉: 在  $A$  世界运算后再过去, 等于先过去再在  $B$  世界运算。

**同构 (Isomorphism)** 双射同态。两个结构本质相同, 只是名字不同。例: 对数函数是  $(\mathbb{R}^+, \times) \rightarrow (\mathbb{R}, +)$  的同构。

**核 (Kernel)** 映射到单位元的元素集合。 $\text{Ker}(\phi) = \{x | \phi(x) = e'\}$

**第一同构定理**  $G/\text{Ker}(\phi) \cong \text{Im}(\phi)$ 。意义: 压缩掉核, 剩下的结构就是像。

**自同构 (Automorphism)** 到自身的同构。 $\text{Aut}(G)$  构成群。例: 复数共轭  $z \mapsto \bar{z}$ 。

## 4. 密码学应用 (Crypto)

基于“难解问题”的安全

**RSA 算法** 基于大数分解困难。

- 公钥  $(n, e)$ , 私钥  $d$ 。
- 加密  $c = m^e \pmod{n}$ 。
- 解密  $m = c^d \pmod{n}$ 。
- 原理: 欧拉定理  $m^{\phi(n)} \equiv 1 \pmod{n}$ 。

**Diffie-Hellman 密钥交换** 基于离散对数困难 ( $g^x \pmod{p}$ ) 求  $x$ 。

- Alice:  $A = g^a$ , Bob:  $B = g^b$ 。
- 共享密钥:  $S = B^a = (g^b)^a = g^{ab} = A^b$ 。

**椭圆曲线 (ECC)** 群运算定义在如下曲线的点集上:

$$y^2 \equiv x^3 + ax + b \pmod{p}$$

优势: 相同安全强度下, 密钥更短。

**ElGamal 加密** 基于离散对数, 由 DH 扩展而来。特点: 引入随机数, 同一明文对应不同密文。

## 5. 编码理论 (Coding Theory)

对抗噪声与错误

**汉明距离 (Hamming Dist)** 两个码字不同位的个数。纠错能力: 若最小距离为  $d$ , 能纠正  $(d-1)/2$  个错误。

**CRC (循环冗余校验)** 把数据看作多项式  $M(x)$ , 除以生成多项式  $G(x)$ , 余数即校验码。本质: 多项式环  $\mathbb{F}_2[x]$  上的模运算。

### 里德-所罗门码 (Reed-Solomon)

- 把数据点视为多项式系数。
- 过  $k$  点可确定  $k-1$  次多项式。
- 只要收到任意  $k$  个分片, 就能恢复原数据。
- 应用: 二维码, RAID 6, 卫星通信。

## 8. 高阶概念 (Advanced)

通向现代数学

**同调代数 (Homology)** 用代数方法研究拓扑空间的“洞”。应用: 拓扑数据分析 (TDA)。

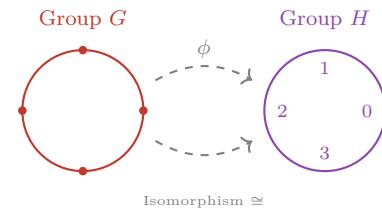
**李群与李代数 (Lie Group)** 既是群又是流形 (光滑连续)。应用: 机器人旋转控制 ( $SO(3)$ ), 粒子物理。

**格 (Lattice)** 偏序集合, 任意两元素有上确界和下确界。应用: 形式化验证, 密码分析。

**表示论 (Representation)** 把群元素映射为矩阵, 用线性代数研究群。

## 核心直觉 (Intuition)

“结构决定性质。”



抽象代数让你忘掉数字的具体大小, 只关注运算规则。

## 三大核心视角

**群即对称:** 研究一个物体怎么转动、翻转后保持不变。物理定律的守恒性 (诺瑟定理) 本质上就是群的对称性。

**域即算术:** 如果你想在计算机里做完美的加减乘除 (没有浮点误差), 你就需要有限域。它是现代编码和密码学的基石。

**同态即翻译:** 把一个复杂系统映射到一个简单系统, 如果结构保持不变, 我们就可以在简单系统里解决问题, 再翻译回去。

所谓理解, 就是发现不同事物背后的同一模式。

# 范畴论 Category Theory Cheat Sheet

The Mathematics of Mathematics: Patterns, Composition, and Abstraction

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 函数式编程 (FP) → Monad / Functor
- 程序架构设计 → 组合性 (Composition)
- 跨领域映射 → 函子 (Functor)
- 最优解/通用解 → 泛性质 (Universal Property)
- 逻辑与类型 → Curry-Howard 同构
- 数据库查询 → Monad Comprehension

## 1. 基础定义 (The Basics)

对象与态射

范畴 (Category)  $\mathcal{C}$  由以下构成:

- Objects ( $Ob(\mathcal{C})$ ): 点 (如集合、群、类型)。
- Morphisms ( $Hom(A, B)$ ): 箭头  $f : A \rightarrow B$ 。
- 公理:
  - 组合:  $f \circ g$  存在。
  - 结合律:  $(h \circ g) \circ f = h \circ (g \circ f)$ 。
  - 单位元: 每个对象有  $id_A$ , 使得  $f \circ id = f$ 。

常见范畴

- Set: 集合与函数。
- Grp: 群与群同态。
- Hask: Haskell 类型与函数。

## 2. 泛性质 (Universal Property)

通过关系定义对象

积 (Product)  $A \times B$  最“好”的包含  $A$  和  $B$  的对象。任何其他包含  $A, B$  的对象  $Z$  都有唯一的映射指向  $A \times B$ 。编程: Tuple / Struct。

上积 (Coproduct)  $A + B$  最“好”的被  $A$  和  $B$  包含的对象。编程: Union / Enum / Sum Type。

极限 (Limit) / 上极限 (Colimit) 积与上积的推广。

## 3. 单子 (Monad)

自函子的么半群

定义  $(M, \eta, \mu)$

- $M$ : 函子  $M : \mathcal{C} \rightarrow \mathcal{C}$ 。
- $\eta$  (Unit):  $id \rightarrow M$  (把值放入上下文)。
- $\mu$  (Join):  $M(M(x)) \rightarrow M(x)$  (展平上下文)。

编程中的 Monad 处理副作用 (Side Effects) 的设计模式。

- return:  $\eta$  (Wrap)。
- bind ( $>>=$ ): 组合带副作用的函数。

常见 Monad:

- Maybe: 处理空值。
- List: 处理非确定性。
- IO: 处理输入输出。
- State: 处理可变状态。

## 4. 函子 (Functor)

范畴间的映射

定义  $F : \mathcal{C} \rightarrow \mathcal{D}$

- 映射对象:  $A \mapsto F(A)$ 。
- 映射态射:  $f \mapsto F(f)$ 。

保结构:

$$F(f \circ g) = F(f) \circ F(g), \quad F(id_A) = id_{F(A)}$$

编程直觉: map 函数。把函数  $f$  提升 (lift) 到容器内部操作。

## 自然变换 (Natural Transformation)

$$\alpha : F \rightarrow G$$

编程直觉: 泛型函数。如 `head :: [a] -> Maybe a`。

应用函子 (Applicative) 介于 Functor 和 Monad 之间。允许将包装在上下文中的函数应用到包装在上下文中的值。

## 5. 伴随 (Adjunction)

范畴论的核心概念

定义  $F \dashv G$  函子  $F : \mathcal{C} \rightarrow \mathcal{D}$  和  $G : \mathcal{D} \rightarrow \mathcal{C}$  是一对伴随, 如果:

$$Hom_{\mathcal{D}}(F(A), B) \cong Hom_{\mathcal{C}}(A, G(B))$$

直觉: 最优解与最自由解的对偶。

- 左伴随  $F$ : 保留上极限 (Colimits)。
- 右伴随  $G$ : 保留极限 (Limits)。

## 例子

- 自由群  $\dashv$  遗忘函子: 构造最自由的群 vs 忘记群结构只看集合。
- Currying:  $Hom(A \times B, C) \cong Hom(A, C^B)$ 。

## 6. 么半范畴 (Monoidal)

自带乘法的范畴

定义  $(\mathcal{C}, \otimes, I)$

- $\otimes$ : 张量积 (Tensor Product)。
- $I$ : 单位对象。

例子:  $(Set, \times, \{\ast\})$  或  $(Vect, \otimes, k)$ 。

应用: 字符串图 (String Diagram) 用线代表对象, 盒子代表态射。量子计算线路图本质上就是么半范畴的图示。

闭范畴 (Closed Category) 拥有内部 Hom 对象 (指数对象  $B^A$ )。对应函数类型  $A \rightarrow B$ 。

## 7. Haskell / Scala 实战

代码工具箱

- Functor: `fmap (+1) [1,2,3]`
- Applicative: `Just (+1) <*> Just 2`
- Monad: `Just 1 >>= \x -> Just (x+1)`
- Lens: 也就是 Profunctor Optics。

## 8. 高阶前沿 (Advanced)

深水区

### Yoneda 引理

$$Hom(Hom(-, A), F) \cong F(A)$$

含义: 一个对象完全由它与其他对象的关系决定。  
编程: Continuation Passing Style (CPS)。

Topos (拓扑斯) 像集合范畴一样的范畴。拥有内部逻辑 (Internal Logic)。连接逻辑、几何与代数的桥梁。

层 (Sheaf) 将局部数据粘合为全局数据的工具。  
应用: 代数几何, 微分几何。上同调 (Cohomology)  
测量粘合的障碍。

## 核心直觉 (Intuition)

“结构重于实质。”

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ h \downarrow & & \downarrow g \\ C & \xrightarrow{k} & D \end{array}$$
$$g \circ f = k \circ h$$

范畴论不关心对象内部是什么 (是数字还是香蕉), 它只关心对象之间如何连接 (态射)。

## 三大核心视角

- 组合性 (Compositionality): 如果一个系统不能被分解为子系统并重新组合, 它就是不可理解的。范畴论是关于组合的科学。
- 抽象的抽象: 算术抽象了苹果, 代数抽象了算术, 范畴论抽象了代数。它是数学的高阶语言。
- 同一性 (Sameness): 两个东西什么时候“一样”? 相等 (Equality)  $\rightarrow$  同构 (Isomorphism)  $\rightarrow$  等价 (Equivalence)。范畴论提供了更灵活的相等观。

所有的数学都是范畴论, 剩下的只是细节。

# 类型论 Type Theory Cheat Sheet

Proofs as Programs: The Foundation of Correctness

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 编译期查错 → 静态类型系统
- 形式化验证 → 依赖类型 (Coq/Agda)
- 泛型编程 → System F / 多态
- 逻辑证明 → Curry-Howard 同构
- 内存安全 → 线性类型 (Rust)
- 自动推导 → Hindley-Milner 算法

## 1. $\lambda$ 演算 (Lambda Calculus)

计算的原子模型

语法

$$e ::= x \mid \lambda x.e \mid e_1 e_2$$

- 变量 ( $x$ ): 占位符。
- 抽象 ( $\lambda x.e$ ): 函数定义。
- 应用 ( $e_1 e_2$ ): 函数调用。

规约 (Reduction)  $\beta$ -规约: 计算的核心。

$$(\lambda x.e)v \longrightarrow e[v/x]$$

含义: 把函数体里的  $x$  替换成参数  $v$ 。 $\alpha$ -变换: 变量重命名 ( $\lambda x.x \equiv \lambda y.y$ )。

## 2. 简单类型 $\lambda$ 演算 (STLC)

给变量贴标签

类型 ( $\tau$ )

$$\tau ::= \text{Bool} \mid \text{Int} \mid \tau_1 \rightarrow \tau_2$$

## 定型规则 (Typing Rules)

$$\frac{\Gamma, x : \tau_1 \vdash e : \tau_2}{\Gamma \vdash \lambda x.e : \tau_1 \rightarrow \tau_2}$$

含义: 如果在假设  $x$  是  $\tau_1$  的情况下能推出  $e$  是  $\tau_2$ , 那么函数  $\lambda x.e$  的类型就是  $\tau_1 \rightarrow \tau_2$ 。

## 3. Curry-Howard 同构

程序 = 证明

| 逻辑 (Logic)          | 编程 (Programming)         |
|---------------------|--------------------------|
| 命题 (Proposition)    | 类型 (Type)                |
| 证明 (Proof)          | 程序 (Program)             |
| 蕴含 $P \implies Q$   | 函数 $P \rightarrow Q$     |
| 合取 $P \wedge Q$     | 积类型 $P \times Q$ (Tuple) |
| 析取 $P \vee Q$       | 和类型 $P + Q$ (Union)      |
| 全称 $\forall x.P(x)$ | 泛型 / 依赖积 $\Pi$           |
| 存在 $\exists x.P(x)$ | 依赖和 $\Sigma$             |

结论: 编写一个通过类型检查的程序, 就是证明了一个数学定理。

## 4. 系统 F (System F)

多态的起源

参数多态 (Polymorphism) 引入类型变量  $\alpha$ 。

$$\Lambda \alpha. \lambda x : \alpha. x : \forall \alpha. \alpha \rightarrow \alpha$$

应用: Java/C# 的 Generics, C++ Templates。

Hindley-Milner (HM) System F 的一个受限子集, 支持完全类型推导。应用: Haskell, OCaml, Rust (部分)。

## 5. 依赖类型 (Dependent Types)

类型可以依赖于值

定义 类型中包含数值。例:  $\text{Vec}\langle \text{Int}, 3 \rangle$  (长度为 3 的数组)。

$$\Pi(x : A).B(x)$$

返回值的类型  $B$  取决于输入参数  $x$  的值。

应用 消除数组越界错误。证明排序算法的输出确实是有序的。语言: Idris, Agda, Coq, Lean。

精制类型 (Refinement Types) 基础类型 + 谓词。 $\{x : \text{Int} \mid x > 0\}$ 。

## 6. 线性类型 (Linear Types)

资源管理

规则 每个变量必须且只能被使用一次。

应用

- 内存管理: Rust 的 Ownership/Borrowing 机制深受其影响。
- 量子计算: 量子态不能被复制 (No-cloning)。
- 并发编程: Session Types 保证通信协议正确。

变体 仿射类型: 最多一次。相关类型: 至少一次。

## 7. 常见类型系统特性

术语表

- 协变 (Covariance): 子类数组是父类数组。
- 逆变 (Contravariance): 函数参数类型相反。
- 结构化类型 (Duck Typing): 长得像就是。
- 名义类型 (Nominal): 名字一样才是。
- 子类型 (Subtyping):  $S <: T$ , S 可用在 T 处。

## 8. 同伦类型论 (HoTT)

数学的新基础

Univalence Axiom 等价的类型是相等的。

$$(A \simeq B) \simeq (A = B)$$

把拓扑学中的“路径”概念引入类型论。类型是空间, 程序是路径。

立方类型论 (Cubical Type Theory) HoTT 的计算性解释。使得单价公理具有计算意义, 可用于编程。

$$\text{Path } A \ a \ b$$

高阶归纳类型 (HITs) 不仅定义点, 还定义路径。例: 圆  $S^1$  由基点  $\text{base}$  和路径  $\text{loop}$  构成。

## 9. 实践中的类型

TypeScript / Rust

代数数据类型 (ADT)

- Sum: enum Option { Some(T), None }
- Product: struct Point { x: i32, y: i32 }

模式匹配 (Pattern Matching) 解构 ADT 的利器。match val { Some(x) => ... , None => ... } 编译器检查穷尽性 (Exhaustiveness Check)。

类型推导 (Type Inference) 编译器自动推断类型。let x = 10; // 自动推断为 int

高阶类型 (HKT) 类型构造器的类型。例: Monad m 中的 m 是  $* \rightarrow *$ 。

## 核心直觉 (Intuition)

“类型是程序的守门人。”

$$\frac{\Gamma \vdash e_1 : \tau_1 \rightarrow \tau_2 \quad \Gamma \vdash e_2 : \tau_1}{\Gamma \vdash e_1 e_2 : \tau_2}$$

类型不仅仅是用来查错的, 它是设计文档, 是逻辑规范。

## 三大核心视角

• 良构即正确: “Well-typed programs cannot go wrong.” (Milner)。类型系统保证了程序在运行时不会发生某些特定的错误 (如把整数当函数调用)。

• 类型驱动开发 (TDD): 先写函数签名 (类型), 再写实现。让编译器指导你补全代码 (Type Holes)。

• 表达力 vs 可判定性: 类型系统越强大 (如依赖类型), 能证明的定理越多, 但编译器的自动推导越难, 甚至不可判定 (图灵停机问题)。

动态类型是一时之爽, 静态类型是一世之安。

# 流形与微分几何 Manifold Cheat Sheet

Calculus on Curved Spaces: From Maps to Metrics

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 机器人运动规划 → 构型空间 (SE(3))
- 广义相对论 → 黎曼流形 / 曲率
- 数据降维 → 流形学习 (t-SNE, UMAP)
- 优化约束 → 黎曼梯度下降
- 计算机图形学 → 曲面参数化 / 法向量
- 李群控制 → 李代数 ( $\mathfrak{so}(3)$ )

## 1. 基础定义 (Foundations)

局部像欧几里得空间

**拓扑流形 (Manifold M)** 一个豪斯多夫空间，且每一点都有一个邻域同胚于  $\mathbb{R}^n$ 。

**坐标卡 (Chart)  $(U, \phi)$**

- $U \subset M$ : 开集。
- $\phi: U \rightarrow \mathbb{R}^n$ : 坐标映射 (地图)。

**图册 (Atlas)  $\mathcal{A}$**  一组覆盖整个  $M$  的坐标卡  $\{(U_\alpha, \phi_\alpha)\}$ 。相容性: 重叠区域的坐标变换  $\phi_\beta \circ \phi_\alpha^{-1}$  必须是光滑的 ( $C^\infty$ )。

## 2. 切空间 (Tangent Space)

曲面上的线性化

**切向量**  $v_p \in T_p M$

- 几何视角: 过点  $p$  的曲线  $\gamma(t)$  的速度向量  $\gamma'(0)$ 。
- 代数视角: 作用在函数上的导数算子 (方向导数)。

**基底 (Basis)** 在局部坐标  $(x^1, \dots, x^n)$  下, 基底为偏导数算子:

$$\left\{ \frac{\partial}{\partial x^1}, \dots, \frac{\partial}{\partial x^n} \right\}$$

任意向量  $v = v^i \frac{\partial}{\partial x^i}$  (爱因斯坦求和)。

## 3. 黎曼度量 (Riemannian Metric)

测量长度与角度

**度量张量**  $g$  在每一点  $p$ , 定义切空间上的内积:

$$g_p: T_p M \times T_p M \rightarrow \mathbb{R}$$

局部坐标表示:  $g_{ij} = \langle \frac{\partial}{\partial x^i}, \frac{\partial}{\partial x^j} \rangle$ 。

**长度与体积**

- 线长:  $L(\gamma) = \int \sqrt{g_{ij} \dot{x}^i \dot{x}^j} dt$
- 体积元:  $dV = \sqrt{\det(g)} dx^1 \dots dx^n$

应用: 在曲面上计算距离 (测地线)。

**诱导度量 (Induced Metric)** 子流形从环境空间继承的度量。例: 球面作为  $\mathbb{R}^3$  子集的度量。

**音乐同构 (Musical Isomorphisms)** 利用度量  $g$  在  $TM$  和  $T^*M$  之间转换。升号  $\sharp$  (Sharp) 与降号  $\flat$  (Flat)。

## 4. 联络与曲率 (Curvature)

空间弯曲的本质

**列维-奇维塔联络**  $\nabla$  定义了向量场如何沿曲线“平移”。

$$\nabla_X Y$$

直觉: 在球面上平行移动切向量, 回到原点会转一个角度。克里斯托费尔符号:

$$\Gamma_{ij}^k = \frac{1}{2} g^{kl} (\partial_i g_{jl} + \partial_j g_{il} - \partial_l g_{ij})$$

**测地线 (Geodesic)** “直线”的推广。自身平移的曲线。

$$\nabla_{\dot{\gamma}} \dot{\gamma} = 0$$

**曲率张量**  $R$  衡量平移交换性的破坏程度。

$$R(X, Y)Z = \nabla_X \nabla_Y Z - \nabla_Y \nabla_X Z - \nabla_{[X, Y]} Z$$

**截面曲率**: 决定了测地线是汇聚 (正) 还是发散 (负)。**里奇曲率 (Ricci)**: 体积元的扭曲程度。

## 5. 微分形式 (Differential Forms)

可以积分的对象

**余切空间**  $T_p^* M$  切空间的对偶空间。基底:  $\{dx^1, \dots, dx^n\}$ 。

**k-形式**  $\omega$  全反对称的  $k$ -重线性映射。外积 ( $\wedge$ ):  $dx \wedge dy = -dy \wedge dx$ 。

**外微分 (Exterior Derivative)  $d$**

$$d(fd\lambda) = \frac{\partial f}{\partial y} dy \wedge dx + \dots$$

- $d^2 = 0$  (核心性质)。
- 统一了梯度、旋度、散度。

**德拉姆上同调 (De Rham)** 闭形式 ( $d\omega = 0$ ) 与恰当形式 ( $\omega = d\eta$ ) 的商空间。

$$H_{dR}^k(M) = \frac{\text{Ker}(d_k)}{\text{Im}(d_{k-1})}$$

## 6. 斯托克斯定理 (Stokes)

微积分基本定理的终极版

$$\int_M d\omega = \int_{\partial M} \omega$$

- $\omega$ :  $(n-1)$ -形式。
- $\partial M$ : 流形的边界。

涵盖了: 牛顿-莱布尼茨、格林公式、高斯公式、斯托克斯公式。

## 7. Python / Geomstats 实战

代码工具箱

- **Geomstats**: 黎曼几何库。
- **Hypersphere(dim=2)**: 定义球面。
- **metric.dist(p, q)**: 计算测地线距离。
- **metric.exp(v, p)**: 指数映射 (沿切向量走)。
- **metric.log(q, p)**: 对数映射 (求切向量)。

## 8. 李群与李代数 (Lie Groups)

光滑的群

**李群**  $G$  既是群又是流形, 且群运算光滑。例:  $SO(3)$  (旋转群),  $GL(n)$  (一般线性群)。

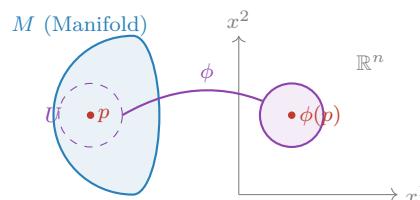
**李代数**  $\mathfrak{g}$  单位元处的切空间  $T_e G$ 。指数映射:

$$\exp: \mathfrak{g} \rightarrow G$$

应用: 3D 旋转通常在李代数 ( $\mathfrak{so}(3)$ , 向量) 上插值, 再映射回李群 ( $SO(3)$ , 矩阵)。李括号:  $[X, Y]$  封闭且满足雅可比恒等式。伴随表示:  $Ad_g(x) = gxg^{-1}$ 。

## 核心直觉 (Intuition)

“地球是圆的, 但地图是平的。”



流形是连接局部简单 (像  $\mathbb{R}^n$ ) 与全局复杂 (拓扑结构) 的桥梁。

## 三大核心视角

**局部线性化**: 切空间  $T_p M$  是流形在一点处的最佳线性逼近。微积分的所有威力 (导数、积分) 都在这里施展。

**坐标无关性**: 真正的物理定律 (如  $F = ma$ , 麦克斯韦方程) 不应该依赖于你选择直角坐标还是球坐标。张量分析就是为了书写这种“坐标无关”的真理。

**内蕴几何**: 高斯绝妙定理: 蚂蚁不需要跳出球面, 只需要测量三角形内角和, 就能知道自己生活在弯曲的空间里。

所有的空间都是平的, 只要你看得足够近。

# 随机微积分 Stochastic Calculus Cheat Sheet

Calculus for the Unpredictable: Ito's Lemma and Beyond

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 股票期权定价 → Black-Scholes 方程
- 噪声系统建模 → 随机微分方程 (SDE)
- 强化学习 (RL) → 策略梯度 / 随机探索
- 扩散模型 (AI) → Langevin 动力学
- 风险管理 → Value at Risk (VaR)
- 信号处理 → 卡尔曼滤波

## 1. 布朗运动 (Brownian Motion)

随机游走  $W_t$

### 定义 (维纳过程)

- $W_0 = 0$ 。
- 独立增量:  $W_{t+s} - W_t$  独立于过往。
- 正态增量:  $W_{t+s} - W_t \sim N(0, s)$ 。
- 路径性质: 处处连续, 但处处不可导。

### 二次变差 (Quadratic Variation)

$$[W, W]_t = t$$

直觉:  $(dW_t)^2 = dt$ 。这是随机微积分与普通微积分的核心区别。协变差:  $[W^i, W^j]_t = \delta_{ij}t$  (独立时为 0)。

## 2. 伊藤积分 (Ito Integral)

对随机过程积分

### 定义

$$I(t) = \int_0^t \Delta_s dW_s$$

取区间左端点进行黎曼和逼近。

- 鞅性质 (Martingale): 期望不变  $E[I(t)] = 0$ 。
- 伊藤等距:  $E[(\int \Delta dW)^2] = E[\int \Delta^2 dt]$ 。

**Stratonovich 积分** 取区间中点。符合普通微积分链式法则, 但不是鞅。物理中常用 Stratonovich, 金融中常用 Ito。

## 3. 伊藤引理 (Ito's Lemma)

随机微积分的链式法则

**公式** 如果  $X_t$  服从  $dX_t = \mu dt + \sigma dW_t$ , 且  $f(t, x)$  是光滑函数, 则  $Y_t = f(t, X_t)$  服从:

$$df = (\frac{\partial f}{\partial t} + \mu \frac{\partial f}{\partial x} + \frac{1}{2}\sigma^2 \frac{\partial^2 f}{\partial x^2})dt + \sigma \frac{\partial f}{\partial x} dW_t$$

### 直觉推导 (泰勒展开)

$$df \approx f_t dt + f_x dX + \frac{1}{2}f_{xx}(dX)^2$$

代入  $(dX)^2 = \sigma^2 dt$  (忽略  $dt^2$  和  $dtdW$ )。关键项:  $\frac{1}{2}\sigma^2 f_{xx} dt$  是普通微积分没有的“凸性修正”。乘积法则:

$$d(X_t Y_t) = X_t dY_t + Y_t dX_t + dX_t dY_t$$

## 4. 随机微分方程 (SDE)

带噪声的 ODE

### 一般形式

$$dX_t = \mu(t, X_t)dt + \sigma(t, X_t)dW_t$$

- $\mu$ : 漂移项 (Drift, 确定性趋势)。
- $\sigma$ : 扩散项 (Diffusion, 随机波动)。

### 几何布朗运动 (GBM) 股票价格模型:

$$dS_t = \mu S_t dt + \sigma S_t dW_t$$

解:  $S_t = S_0 \exp((\mu - \frac{1}{2}\sigma^2)t + \sigma W_t)$ 。注意:  $\frac{1}{2}\sigma^2$  的修正项。

### Ornstein-Uhlenbeck 过程 (OU) 均值回归过程 (Mean-Reverting)。

$$dX_t = \theta(\mu - X_t)dt + \sigma dW_t$$

- $\theta$ : 回归速度。 $\mu$ : 长期均值。
- 应用: 利率模型 (Vasicek), 配对交易。
- 解是高斯过程, 方差有界。

## 5. Black-Scholes 方程

金融界的  $E = mc^2$

**偏微分方程 (PDE)** 期权价格  $V(S, t)$  满足:

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS \frac{\partial V}{\partial S} - rV = 0$$

**费曼-卡茨公式 (Feynman-Kac)** 建立了 PDE 和 SDE 的桥梁。PDE 的解等于 SDE 路径的期望。

$$V(S, t) = E[e^{-r(T-t)} \text{Payoff}(S_T)]$$

启示: 求解 PDE 可以通过蒙特卡洛模拟 SDE 来实现。希腊字母:  $\Delta = \partial V / \partial S$  (对冲比率),  $\Gamma = \partial^2 V / \partial S^2$  (凸性)。

## 8. 高阶应用 (Advanced)

AI 与物理

### 朗之万动力学 (Langevin)

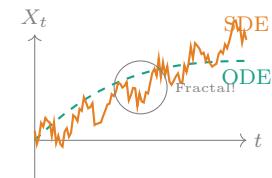
$$dX_t = -\nabla U(X_t)dt + \sqrt{2}dW_t$$

平稳分布是玻尔兹曼分布  $p(x) \propto e^{-U(x)}$ 。应用: 贝叶斯采样 (MCMC), 扩散模型 (Stable Diffusion)。

**随机控制 (Stochastic Control)** HJB 方程 (Hamilton-Jacobi-Bellman)。在随机环境下寻找最优策略。

## 核心直觉 (Intuition)

“噪音也是信号。”



随机微积分处理的是不可微的函数。普通微积分假设曲线无限放大后是直线, 而布朗运动无限放大后依然是锯齿。

### 三大核心视角

• **二阶效应不可忽略:** 在普通微积分中,  $(dx)^2$  是高阶无穷小, 可以扔掉。但在随机微积分中,  $(dW)^2 = dt$ , 是一阶无穷小, 必须保留。这就是伊藤项的来源。

• **鞅 (Martingale):** 公平的赌局。如果你无法预测未来的增量 (期望为 0), 那么这个过程就是鞅。有效市场假说认为股价本质上是鞅。

• **噪声驱动:** SDE 告诉我们, 系统的演化 = 确定性的物理定律 (Drift) + 随机的热涨落 (Diffusion)。

上帝掷骰子, 伊藤算出了骰子的轨迹。

# 拓扑学 Topology Cheat Sheet

The Geometry of Rubber Sheets: Continuity and Connectivity

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 数据形状分析 → 持续同调 (TDA)
- 网络连通性 → 连通分量 / Betti 数
- 函数极值存在性 → 紧致性 (Compactness)
- 机器人路径规划 → 同伦 (Homotopy)
- 向量场奇点 → 庞加莱-霍普夫定理
- 扭结分类 → 琼斯多项式

## 1. 拓扑空间 (Topological Space)

抛弃距离, 只谈“接近”

**定义**  $(X, \mathcal{T})$   $\mathcal{T}$  是  $X$  的子集族 (称为开集), 满足:

- $\emptyset, X \in \mathcal{T}$ 。
- 任意并集仍在  $\mathcal{T}$  中。
- 有限交集仍在  $\mathcal{T}$  中。

**连续函数 (Continuity)**  $f: X \rightarrow Y$  是连续的, 当且仅当: 开集的原像还是开集。

$$V \in \mathcal{T}_Y \implies f^{-1}(V) \in \mathcal{T}_X$$

这是微积分中  $\epsilon - \delta$  定义的终极推广。

## 2. 核心性质 (Properties)

拓扑不变量

**紧致性 (Compactness)** “有限”的推广。任意开覆盖都有有限子覆盖。定理: 紧集上的连续实函数必有最大最小值。

**连通性 (Connectedness)** 不能被分成两个不相交的非空开集。定理: 介值定理 (Intermediate Value Theorem)。

**豪斯多夫 (Hausdorff,  $T_2$ )** 任意两点都能被不相交的开集分开。大多数“好”的空间都是 Hausdorff 的。

## 3. 代数拓扑 (Algebraic Topology)

用群来算洞

**同伦 (Homotopy)** 两个映射  $f, g$  可以连续变形互相转化。

**基本群 (Fundamental Group)**  $\pi_1(X)$  所有从基点出发又回来的回路 (Loop) 的同伦类构成的群。

- $\pi_1(S^1) = \mathbb{Z}$  (绕圈次数)。
- $\pi_1(S^2) = 0$  (球面上所有圈都能缩成一点)。
- $\pi_1(T^2) = \mathbb{Z} \times \mathbb{Z}$  (环面)。

**同调群 (Homology)**  $H_n(X)$  比同伦群更容易计算 (阿贝尔群)。

- $H_0$ : 连通分量个数。
- $H_1$ : 孔 (Holes) / 环的个数。
- $H_2$ : 空腔 (Voids) 的个数。

**贝蒂数 (Betti Numbers)**  $k$  维洞的个数  $\beta_k = \text{rank}(H_k(X))$ 。

$$\chi = \sum_k (-1)^k \beta_k$$

## 4. 欧拉示性数 (Euler Char)

$$V - E + F$$

$$\chi = V - E + F$$

对于曲面:

$$\chi = 2 - 2g$$

( $g$  是亏格 Genus, 即洞的个数)。

- 球面:  $\chi = 2$  ( $g = 0$ )。
- 环面 (甜甜圈):  $\chi = 0$  ( $g = 1$ )。
- 双环面:  $\chi = -2$  ( $g = 2$ )。

这是唯一的整数拓扑不变量。

**庞加莱-霍普夫定理** 光滑向量场零点指数之和等于  $\chi(M)$ 。

$$\sum \text{index}(v) = \chi(M)$$

## 5. 拓扑数据分析 (TDA)

大数据的形状

**单纯复形 (Simplicial Complex)** 点、线、面、体构成的组合结构, 用来逼近数据流形。

**持续同调 (Persistent Homology)** 随着“分辨率”(半径  $\epsilon$ ) 增大, 洞的产生与消失。**条形码 (Barcode)**: 长条代表显著的拓扑特征 (信号), 短条代表噪声。

**Vietoris-Rips 复形** 若  $d(x_i, x_j) < \epsilon$ , 则连边。若子集中任意两点距离都小于  $\epsilon$ , 则构成高维单纯形。稳定性定理: 持续图对噪声是稳定的。

## 6. 不动点定理 (Fixed Point)

必然的存在

**Brouwer 不动点定理** 从圆盘到自身的任意连续映射, 必有一个不动点。直觉: 搅拌咖啡, 总有一个分子停在原来的位置。

**毛球定理 (Hairy Ball)** 偶数维球面上不存在处处非零的连续切向量场。直觉: 你永远无法把椰子上的毛完全梳平, 总会有一个旋。

**博苏克-乌拉姆定理 (Borsuk-Ulam)**

$f: S^n \rightarrow \mathbb{R}^n$  必有一对对径点映射到同一点。地球上总有一对对径点气温气压完全相同。

## 7. Python / Gudhi 实战

代码工具箱

- Gudhi / Ripser**: TDA 库。
- `RipsComplex(points).create_simplices()`
- `st.dimension() / st.num_simplices()`
- `persistence()`: 计算持续同调。
- `plot_persistence_barcode()`: 画条形码。
- Bottleneck**: `bottleneck_distance(d1, d2)`

## 8. 高阶前沿 (Advanced)

深不可测

**扭结理论 (Knot Theory)** 研究圆圈在三维空间中的嵌入。应用: DNA 拓扑结构, 量子场论。

**微分拓扑** 研究光滑流形的拓扑。莫尔斯理论 (Morse): 通过研究流形上函数的临界点来推导拓扑结构。登山者视角: 山峰、鞍点、山谷决定了地形的拓扑。

**庞加莱猜想 (Poincaré Conjecture)** 单连通的闭 3-流形同胚于  $S^3$ 。由 Perelman 证明 (Ricci Flow)。

**高阶同伦群 (Homotopy Groups)**

$\pi_n(X) = [S^n, X]$  (球面到空间的映射类)。 $\pi_k(S^n)$  即使  $k > n$  也可能非平凡。Hopf 纤维丛:  $\pi_3(S^2) \cong \mathbb{Z}$ 。

## 核心直觉 (Intuition)

“咖啡杯就是甜甜圈。”



Torus ( $g = 1$ )

Mug ( $g = 1$ )

拓扑学是橡皮泥几何学。它不关心长度、角度、面积, 只关心洞的个数和连通性。

## 三大核心视角

- 连续形变**: 只要不撕破、不粘合, 物体就是一样的 (同胚)。在拓扑学家眼里, 正方体和球体是一样的, 但和圆环不一样。
- 局部与全局**: 流形在局部都是平的 (欧几里得), 但全局可能有洞、扭曲 (莫比乌斯带)。代数拓扑就是用代数工具 (群) 来探测这些全局特征。
- 降维打击**: 欧拉示性数  $\chi$  把复杂的几何形状压缩成了一个整数。这是数学抽象的极致体现。

洞, 定义了实体的本质。

# 泛函分析 Functional Analysis Cheat Sheet

Infinite Dimensional Linear Algebra: Spaces of Functions

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 量子力学 → 希尔伯特空间 / 算子谱
- 偏微分方程 (PDE) → 索伯列夫空间 / 弱解
- 机器学习核方法 → RKHS (再生核希尔伯特空间)
- 信号压缩 → 巴拿赫空间 / 小波基
- 变分法 → 欧拉-拉格朗日方程
- 分布理论 → 广义函数 ( $\delta$  函数)

## 1. 空间层级 (Hierarchy)

从松散到严格

1. 线性空间 (Vector Space) 只有加法和数乘。

2. 范数空间 (Normed Space) 有长度  $\|x\|$ 。Banach 空间: 完备的范数空间 (柯西序列收敛)。例:  $L^p$  空间 ( $p \neq 2$ ),  $C[a, b]$ 。

3. 内积空间 (Inner Product) 有角度  $\langle x, y \rangle$ 。Hilbert 空间: 完备的内积空间。例:  $L^2$  空间,  $\mathbb{R}^n$ 。

$$\|x\| = \sqrt{\langle x, x \rangle}$$

## 2. 核心不等式 (Inequalities)

分析的武器

### Cauchy-Schwarz

$$|\langle x, y \rangle| \leq \|x\| \|y\|$$

内积空间最重要的不等式。

### Hölder 不等式

$$\|fg\|_1 \leq \|f\|_p \|g\|_q \quad (1/p + 1/q = 1)$$

### Minkowski 不等式

$$\|f + g\|_p \leq \|f\|_p + \|g\|_p$$

## 3. 算子理论 (Operator Theory)

无限维矩阵

### 有界线性算子

$$\|T\| = \sup_{\|x\|=1} \|Tx\| < \infty$$

定理: 线性算子有界  $\iff$  连续。

**对偶空间 (Dual Space)**  $X^*$  所有连续线性泛函  $f : X \rightarrow \mathbb{R}$  构成的空间。Riesz 表示定理: 在 Hilbert 空间中, 任意泛函  $f(x)$  都可以写成内积形式  $\langle x, y \rangle$ 。

**自反空间 (Reflexive Space)** 若典范嵌入  $J : X \rightarrow X^{***}$  是满射 ( $X \cong X^{***}$ )。例: Hilbert 空间和  $L^p$  ( $1 < p < \infty$ ) 是自反的。

### 伴随算子 (Adjoint) $T^*$

$$\langle Tx, y \rangle = \langle x, T^*y \rangle$$

自伴算子 (Hermitian):  $T = T^*$ 。特征值必为实数 (量子力学可观测量的基础)。

**紧算子 (Compact Operator)** 将有界集映射为相对紧集 (表现像有限维矩阵)。Fredholm 选择定理: 紧算子谱理论的基础。

## 4. 三大基本定理 (The Big Three)

Banach 空间的基石

1. **Hahn-Banach 定理** (延拓) 局部定义的线性泛函可以保范数地延拓到全空间。保证了“足够多”的泛函存在。

2. **开映射定理** (逆) 满射的连续线性算子将开集映为开集。推论: 双射连续算子的逆也是连续的。

3. **一致有界原理 (Banach-Steinhaus)** (共鸣) 一族算子如果点点有界, 则一致有界。

4. **闭图像定理 (Closed Graph)** 若  $T : X \rightarrow Y$  的图像在  $X \times Y$  中闭, 则  $T$  连续。

## 5. 谱理论 (Spectral Theory)

特征值的推广

**谱 (Spectrum)**  $\sigma(T)$  使得  $T - \lambda I$  不可逆的  $\lambda$  的集合。预解集:  $\rho(T) = \mathbb{C} \setminus \sigma(T)$  (算子可逆)。

- 点谱: 真正的特征值 ( $Tx = \lambda x$ )。
- 连续谱: 类似  $Tf(x) = xf(x)$  这种算子。
- 剩余谱: 像空间不稠密。

**Fredholm 二择一** 紧算子扰动  $I - K$  要么可逆, 要么有非零核。

**谱定理 (Spectral Theorem)** 紧自伴算子可以对角化:

$$T = \sum \lambda_i \langle \cdot, e_i \rangle e_i$$

这是线性代数中对称矩阵对角化的无穷维推广。

## 6. 广义函数 (Distributions)

导数的解放

测试函数空间  $\mathcal{D}$  光滑且紧支集的函数 ( $C_c^\infty$ )。

广义函数  $T \in \mathcal{D}$  上的连续线性泛函。

$$\langle T, \phi \rangle = \int T(x) \phi(x) dx$$

Dirac  $\delta$ :  $\langle \delta, \phi \rangle = \phi(0)$ 。弱导数:  $\langle T', \phi \rangle = -\langle T, \phi' \rangle$ 。这让不可导函数 (如阶跃函数) 也有了导数 ( $\delta$  函数)。

**卷积 (Convolution)** 广义函数与光滑函数的卷积是光滑函数 (正则化)。

## 7. Python / Scikit-Learn 实战

代码工具箱

- SVM (核方法):** SVC(kernel='rbf') 隐式地将数据映射到无限维 Hilbert 空间 (RKHS) 中找超平面。
- 小波变换:** PyWavelets  $L^2$  空间的一组正交基。

## 8. 高阶应用 (Advanced)

现代物理语言

**索伯列夫空间 (Sobolev)**  $W^{k,p}$  函数本身及其弱导数都在  $L^p$  空间中。

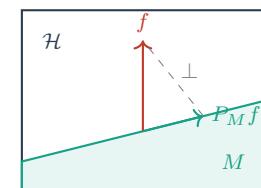
**$C^*$ -代数** 算子代数的抽象。

**阿蒂亚-辛格指标定理** 分析 (算子核的维数) 与拓扑 (流形的示性类) 的深刻联系。

**弱拓扑 (Weak Topology)**  $x_n \rightharpoonup x \iff \forall f \in X^*, f(x_n) \rightarrow f(x)$ . Banach-Alaoglu: 对偶单位球是弱 \* 紧的。

## 核心直觉 (Intuition)

“函数是空间中的点。”



泛函分析把函数看作无穷维向量空间中的点。

## 三大核心视角

• **完备性 (Completeness):** 为什么我们要用 Lebesgue 积分而不是 Riemann 积分? 因为  $L^2$  空间是完备的 (Hilbert), 而  $C[a, b]$  不是。完备性保证了柯西序列一定有极限 (不会跑出空间外)。

• **对偶性 (Duality):** 通过研究作用在物体上的测量 (泛函), 来研究物体本身。就像通过投影来重构三维物体。

• **致密性 (Compactness):** 在无穷维空间, 有界闭集不一定是紧的 (单位球不紧)。这导致了无穷维分析的种种困难, 但也催生了弱拓扑等深刻概念。

有限维是特例, 无穷维才是常态。

# 辛几何 Symplectic Geometry Cheat Sheet

The Geometry of Phase Space: Mechanics and Area Preservation

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 经典力学 → 哈密顿方程 / 相空间
- 数值积分 (保能量) → 辛积分器 (Symplectic Integrator)
- 量子化 → 几何量子化 / 形变量子化
- 光学 → 费马原理 / 辛矩阵
- 镜像对称 → 凯勒流形 (Kähler)
- 控制理论 → 庞特里亚金极值原理

## 1. 辛向量空间 (Linear)

反对称的内积

辛形式  $\omega$  向量空间  $V$  上的双线性形式, 满足:

- 反对称:  $\omega(u, v) = -\omega(v, u)$ 。
- 非退化:  $\forall u \neq 0, \exists v, \omega(u, v) \neq 0$ 。

推论:  $V$  的维数必须是偶数  $2n$ 。

标准型 存在基底  $\{q_i, p_i\}$  使得:

$$\omega = \sum_{i=1}^n dp_i \wedge dq_i$$

物理意义:  $q$  是位置,  $p$  是动量。 $\omega$  测量“有向面积”。

## 2. 辛流形 (Manifold)

弯曲的相空间

定义 ( $M, \omega$ )  $M$  是偶数维流形,  $\omega$  是闭的非退化 2-形式。

$$d\omega = 0$$

达布定理 (Darboux) 辛流形在局部都和标准辛空间  $(\mathbb{R}^{2n}, \sum dp_i \wedge dq_i)$  同构。对比: 黎曼流形局部有曲率, 辛流形局部都是平的。辛几何是全局几何。

余切丛 (Cotangent Bundle)  $M = T^*Q$  具有典范辛形式  $\omega = d\theta$  (刘维尔形式)。

## 3. 哈密顿力学 (Hamiltonian)

物理的几何化

哈密顿向量场  $X_H$  给定能量函数 (哈密顿量)  $H: M \rightarrow \mathbb{R}$ , 定义  $X_H$ :

$$\iota_{X_H} \omega = dH$$

即  $\omega(X_H, \cdot) = dH(\cdot)$ 。

运动方程

$$\dot{q} = \frac{\partial H}{\partial p}, \quad \dot{p} = -\frac{\partial H}{\partial q}$$

这正是  $X_H$  的积分曲线。

刘维尔定理 (Liouville) 哈密顿流保持辛形式  $\omega$  不变 (也保持体积形式  $\omega^n$  不变)。物理意义: 相空间体积守恒 (信息不灭)。

诺特定理 (Noether) 若  $H$  在辛变换  $\phi_t$  下不变, 则生成元  $f$  是守恒量。

$$\{f, H\} = 0 \implies f \text{ 是运动常数}$$

## 4. 泊松括号 (Poisson Bracket)

代数结构

定义

$$\{f, g\} = \omega(X_f, X_g)$$

在标准坐标下:

$$\{f, g\} = \sum \left( \frac{\partial f}{\partial q_i} \frac{\partial g}{\partial p_i} - \frac{\partial f}{\partial p_i} \frac{\partial g}{\partial q_i} \right)$$

演化方程

$$\frac{df}{dt} = \{f, H\}$$

量子力学对应:  $\frac{d\hat{A}}{dt} = \frac{1}{i\hbar} [\hat{A}, \hat{H}]$ 。

雅可比恒等式 (Jacobi)

$$\{f, \{g, h\}\} + \{g, \{h, f\}\} + \{h, \{f, g\}\} = 0$$

这使得光滑函数空间构成无穷维李代数。

## 5. 辛积分器 (Symplectic Integrator)

数值计算的魔法

为什么不用 RK4? 普通积分器 (如 Runge-Kutta) 不保持辛结构, 长期积分会导致能量漂移 (系统发散或耗散)。

Verlet / Leapfrog 算法

$$p_{n+1/2} = p_n - \frac{\Delta t}{2} \nabla V(q_n)$$

$$q_{n+1} = q_n + \Delta t M^{-1} p_{n+1/2}$$

$$p_{n+1} = p_{n+1/2} - \frac{\Delta t}{2} \nabla V(q_{n+1})$$

特性: 显式、时间可逆、长期保持能量稳定。

影子哈密顿量 (Shadow Hamiltonian)

辛算法精确求解的是一个被扰动的哈密顿量  $\tilde{H} = H + O(\Delta t^k)$ 。

## 6. 拉格朗日流形 (Lagrangian)

半维子流形

定义  $L \subset M$   $L$  的维数是  $M$  的一半 ( $n$ ), 且  $\omega|_L = 0$ 。例子: 构型空间 (只包含  $q$  的空间) 是相空间的一个拉格朗日流形。

应用 WKB 近似、焦散线 (Caustics)、几何光学。

生成函数 (Generating Function) 辛变换可以通过生成函数  $S(q, P)$  隐式定义。

$$p = \frac{\partial S}{\partial q}, \quad Q = \frac{\partial S}{\partial P}$$

## 7. Python / SciPy 实战

代码工具箱

- Hamiltonian Monte Carlo (HMC): 利用哈密顿力学在概率分布上高效采样 (PyMC3 / Stan)。
- Rebound: 天体物理辛积分库。

## 8. 高阶前沿 (Advanced)

深水区

格罗莫夫非挤压定理 (Non-squeezing)

辛骆驼定理: 你不能把一个大球通过保辛变换塞进一个细管子里, 除非管子够粗。

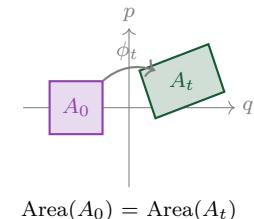
Floer 同调 无穷维莫尔斯理论。用于证明阿诺德猜想 (哈密顿系统不动点个数)。

接触几何 (Contact Geometry) 奇数维的辛几何对应物。

深谷范畴 (Fukaya Category) 对象是拉格朗日子流形, 态射是拉格朗日交。

## 核心直觉 (Intuition)

“面积是有方向的。”



辛几何是相空间的几何学。它描述了守恒系统如何随时间演化。

## 三大核心视角

- 反对称性: 黎曼几何基于对称的度量  $g_{ij}$  (长度), 辛几何基于反对称的形式  $\omega_{ij}$  (面积)。
- 守恒律: 辛变换不仅保持体积 (Liouville), 还保持每个 2D 投影面的有向面积。这比单纯的体积守恒要强得多。
- 刚性 (Rigidity): 辛流形像是一种“不可压缩的流体”, 但比流体更硬。有些形状变换在拓扑上允许, 在体积上允许, 但在辛几何中被禁止 (辛骆驼)。

动量与位置共舞, 面积永恒不变。

# 代数几何 Algebraic Geometry Cheat Sheet

Geometry of Equations: Varieties, Schemes, and Cohomology

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 机器人运动学 → 理想/Gröbner 基
- 密码学 (ECC) → 椭圆曲线
- 纠错码 → Goppa 码 / 代数曲线
- 计算机视觉 → 射影几何 / 基础矩阵
- 系统生物学 → 化学反应网络理论
- 字符串理论 → Calabi-Yau 流形

## 1. 仿射簇 (Affine Varieties)

多项式的零点集

**定义**  $V(S)$  给定多项式集合  $S \subset k[x_1, \dots, x_n]$ ,

$$V(S) = \{x \in k^n \mid f(x) = 0, \forall f \in S\}$$

**理想 (Ideal)**  $I(V)$  在  $V$  上全为 0 的多项式集合。坐标环:  $k[V] = k[x_1, \dots, x_n]/I(V)$ 。

**希尔伯特零点定理 (Nullstellensatz)** 代数与几何的完美对应 (在代数闭域上):

$$I(V(J)) = \sqrt{J}$$

几何对象 (簇) 与代数对象 (理想) 一一对应。

**扎里斯基拓扑 (Zariski)** 闭集即代数集。非常粗糙 (非 Hausdorff)。

## 2. 射影几何 (Projective)

引入无穷远点

**射影空间**  $\mathbb{P}^n \cong k^{n+1} \setminus \{0\}$  中过原点的直线集合。坐标  $(x_0 : x_1 : \dots : x_n)$ , 差一个缩放因子。

**齐次多项式** 只有齐次多项式在射影空间上有定义 (零点位置不变)。

**贝祖定理 (Bézout)** 两条  $m$  次和  $n$  次的平面曲线, 在  $\mathbb{P}^2$  中恰好有  $mn$  个交点 (计入重数)。完美! 没有“平行线不相交”这种例外。

## 3. 概型 (Schemes)

格罗滕迪克的革命

**为什么需要概型?** 簇无法处理重根 (如  $x^2 = 0$ ) 和数论问题 (整数环  $\mathbb{Z}$ )。

**仿射概型 Spec(R)** 环  $R$  的所有素理想构成的空间。包含了几何点 (极大理想) 和“肥点” (非极大素理想)。

**层 (Sheaf)**  $\mathcal{O}_X$  在每个开集上定义“函数”。概型 = 拓扑空间 + 结构层 (局部像 Spec(R))。

**态射 (Morphisms)** 概型间的映射  $f: X \rightarrow Y$  由连续映射和层同态组成。代数几何中的函数不仅是点到点的映射, 还拉回函数。

**纤维积 (Fiber Product)**  $X \times_S Y$ : 概型范畴中的“交集”操作。

## 4. 维数理论 (Dimension)

大小的度量

**Krull 维数** 素理想链的最大长度:  $P_0 \subsetneq P_1 \subsetneq \dots \subsetneq P_d$ . 对应几何对象的维数 (点 = 0, 线 = 1, 面 = 2)。

**切空间 (Tangent Space)** 在点  $p$  处的线性逼近。

$$T_p(V) = (I(V)/\mathfrak{m}_p^2)^*$$

## 5. Gröbner 基 (Gröbner Basis)

多项式的“高斯消元”

**问题** 如何判断  $f$  是否属于理想  $I = \langle f_1, \dots, f_m \rangle$ ? 如何解多项式方程组?

**Buchberger 算法** 计算理想的一组“好”的生成元  $G$ 。

- 任意多项式除以  $G$  的余式唯一。
- 可以用来消元、解方程。

计算代数几何的核心算法。

## 6. 椭圆曲线 (Elliptic Curves)

代数几何的明珠

**方程**

$$y^2 = x^3 + ax + b$$

**群结构** 曲线上的点构成一个阿贝尔群。**加法法则:** 三点共线  $P + Q + R = 0$  (几何作图)。**Mordell-Weil 定理:**  $\mathbb{Q}$  上的有理点群是有限生成的。

**黎曼-罗赫定理 (Riemann-Roch)** 计算具有特定极点的函数空间的维数。

$$l(D) - l(K - D) = \deg(D) - g + 1$$

代数几何中最重要的计数公式。

**应用**

- ECC: 离散对数问题难解。
- 费马大定理: 证明的核心。
- j-invariant:  $j(E)$  唯一确定同构类。

## 9. 高阶前沿 (Advanced)

深空探索

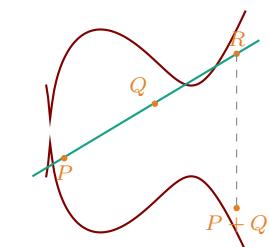
**模空间 (Moduli Space)** 参数化一族几何对象的空间。例: 所有亏格为  $g$  的黎曼面的集合  $\mathcal{M}_g$ 。

**平展上同调 (Etale Cohomology)** 为了证明韦伊猜想 (Weil Conjectures) 而发明的。在代数几何中定义了类似拓扑学的“孔”。

**Motive (动机)** 格罗滕迪克的终极梦想: 统一所有上同调理论。寻找代数簇的“原子”分解。**Chow Ring:** 代数闭链构成的环。

## 核心直觉 (Intuition)

“代数是几何的影子。”



代数几何在方程 (代数) 和形状 (几何) 之间建立了字典。

## 三大核心视角

- **刚性:** 代数函数 (多项式) 非常坚硬。如果你知道它在局部的一小段, 你就知道了它在全宇宙的样子 (解析延拓)。这与拓扑学的柔软形成鲜明对比。
- **内蕴代数:** 坐标环  $k[V]$  包含了簇  $V$  的所有几何信息。研究几何变成了研究环论。
- **无穷远点:** 射影几何告诉我们, 平行线在无穷远处相交。这修复了欧几里得几何中的许多“例外”, 让定理变得完美 (如 Bézout 定理)。

每一个素理想都是一个点, 每一个环都是一个世界。

# 张量分析 Tensor Calculus Cheat Sheet

The Language of Physics: Invariance and Covariance

## 场景导航 (Use Cases)

遇到什么问题 → 用什么工具

- 广义相对论 → 黎曼曲率张量 / 爱因斯坦方程
- 流体力学 → 应力张量 / 纳维-斯托克斯
- 深度学习 → 多维数组运算 (PyTorch)
- 连续介质力学 → 应变张量
- 电磁学 → 麦克斯韦张量  $F_{\mu\nu}$
- 微分几何 → 形式与向量场

## 1. 基础定义 (Foundations)

坐标变换的规律

爱因斯坦求和约定 上下指标重复即求和。

$$v^i e_i = \sum_{i=1}^n v^i e_i$$

逆变向量 (Contravariant)  $v^i$  像坐标微分  $dx^i$  一样变换 (上标)。

$$\bar{v}^i = \frac{\partial \bar{x}^i}{\partial x^j} v^j$$

协变向量 (Covariant)  $u_i$  像梯度  $\frac{\partial \phi}{\partial x^i}$  一样变换 (下标)。

$$\bar{u}_i = \frac{\partial x^j}{\partial \bar{x}^i} u_j$$

直觉: 逆变是箭头, 协变是等高线。

## 2. 度量张量 (Metric Tensor)

升降指标的梯子

定义  $g_{ij}$

$$ds^2 = g_{ij} dx^i dx^j$$

- 降指标:  $v_i = g_{ij} v^j$ 。
- 升指标:  $v^i = g^{ij} v_j$  ( $g^{ij}$  是  $g_{ij}$  的逆矩阵)。

内积

$$\mathbf{u} \cdot \mathbf{v} = g_{ij} u^i v^j = u^i v_i$$

## 3. 协变导数 (Covariant Derivative)

弯曲空间怎么求导?

问题 普通偏导  $\partial_i v^j$  不是张量 (因为基底也在变)。

定义  $\nabla_i v^j$

$$\nabla_i v^j = \partial_i v^j + \Gamma_{ik}^j v^k$$

- $\Gamma_{ik}^j$ : 克里斯托费尔符号 (Christoffel Symbols)。
- 修正项描述了坐标系的弯曲/扭曲。

对于标量  $\nabla_i \phi = \partial_i \phi$  (标量没有方向, 无需修正)。

度量相容性  $\nabla_\gamma g_{\alpha\beta} = 0$  (度量在平移下不变)。

## 4. 测地线 (Geodesics)

弯曲空间中的直线

测地线方程

$$\frac{d^2 x^\mu}{d\tau^2} + \Gamma_{\nu\lambda}^\mu \frac{dx^\nu}{d\tau} \frac{dx^\lambda}{d\tau} = 0$$

物理意义: 自由落体运动的轨迹 (无外力)。

平移 (Parallel Transport) 向量沿曲线移动时保持“平行” ( $\nabla_{\dot{\gamma}} v = 0$ )。测地线就是切向量沿自身平移的曲线。

## 5. 曲率 (Curvature)

平移一圈回到原点

黎曼曲率张量  $R_{\sigma\mu\nu}^\rho$

$$[\nabla_\mu, \nabla_\nu] v^\rho = R_{\sigma\mu\nu}^\rho v^\sigma$$

衡量协变导数的不可交换性。

对称性  $R_{\rho\sigma\mu\nu} = -R_{\sigma\rho\mu\nu} = -R_{\rho\sigma\nu\mu} = R_{\mu\nu\rho\sigma}$

里奇张量 (Ricci)  $R_{\mu\nu} R^{\mu\nu}_{\lambda\nu} = R_{\mu\nu}^\lambda$  (缩并)。  
里奇标量  $R$ :  $R = g^{\mu\nu} R_{\mu\nu}$ 。物理意义: 描述体积的扭曲。

## 5. 物理应用 (Physics)

宇宙的方程

爱因斯坦场方程 (GR)

$$R_{\mu\nu} - \frac{1}{2} R g_{\mu\nu} = \frac{8\pi G}{c^4} T_{\mu\nu}$$

几何 (左边) = 物质 (右边)。时空告诉物质如何运动, 物质告诉时空如何弯曲。

麦克斯韦方程组

$$\partial_\mu F^{\mu\nu} = \mu_0 J^\nu$$

$F^{\mu\nu}$  是电磁场张量。四个方程合而为一。

## 9. 高阶前沿 (Advanced)

超越张量

旋量 (Spinor) “半个”向量。旋转 360 度不变号, 旋转 720 度才复原。描述电子等费米子。

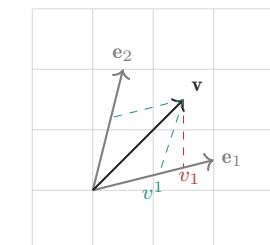
李导数 (Lie Derivative)  $\mathcal{L}_X$  不需要度量, 只需要流形结构。描述张量场沿流线的变化。

外微分形式 (Forms) 反对称的张量。 $d, \wedge, *$  (Hodge Star)。比指标记法更优雅、更几何。

挠率 (Torsion)  $\Gamma_{\mu\nu}^\lambda \neq \Gamma_{\nu\mu}^\lambda$  时存在挠率。 $T_{\mu\nu}^\lambda = \Gamma_{\mu\nu}^\lambda - \Gamma_{\nu\mu}^\lambda$  ( $GR$  中通常假设为 0)。

## 核心直觉 (Intuition)

“物理定律对坐标系免疫。”



张量不仅仅是多维数组, 它是几何实体。

## 三大核心视角

• 坐标无关性: 张量方程 (如  $T_{\mu\nu} = 0$ ) 在一个坐标系下成立, 则在所有坐标系下成立。这是广义相对论的基石 (广义协变性原理)。

• 指标体操: 上标和下标的缩并 (Contraction) 本质上是矩阵乘法或求迹。学会熟练地升降指标, 你就能在弯曲时空中自由穿梭。

• 几何修正:  $\Gamma$  (联络) 不是张量, 它是坐标系弯曲产生的“惯性力”(如离心力、科里奥利力)。把它加上去, 导数就变成了真正的几何导数 (协变导数)。

无论你怎么扭曲时空, 张量依然是张量。

## 8. Python / EinsteinPy 实战

代码工具箱

- Einstein Summation:  
`np.einsum('ij,jk->ik', A, B)` (矩阵乘法)  
`np.einsum('ii', A)` (迹)
- EinsteinPy: 广义相对论库。
- SymPy.diffgeom: 符号计算张量。