多项式和光滑函数零点问题的多实根求解算法

周 旸　　任彦鲲
2023 级信息与计算科学　　2023 级数学与应用数学

日期：2025 年 12 月 17 日

摘　　要

本文围绕多项式与光滑函数零点问题中的多实根求解，面向实根数量判定与根区间定位这两个核心困难，系统梳理并实现了若干代表性算法。对多项式情形，分别采用单项式基底下的伴随矩阵特征值法与 Chebyshev 多项式基底下的同事矩阵特征值法，将求根转化为矩阵特征值计算，并实现基于 Sturm 序列计数与区间隔离的 Chebyshev–Sturm 方法；同时给出用于多变量多项式方程组的总次数同伦续接算法框架，以扩展到更一般的代数方程组求解。对一般光滑函数，重点讨论 Boyd–Battles 方法：在给定区间上自适应进行 Chebyshev 插值，并用 `standardChop` 截断得到 Chebyshev 代理多项式，再通过同事矩阵特征值法生成候选根并结合 Newton 精化与原函数检验输出区间内全部实根。数值实验选取含复根、重根与极近根簇的多项式算例，以及振荡衰减与近重根的光滑函数算例；在多项式实验中统一使用 MATLAB 的特征值算法并以虚部阈值 10^{-3} 进行实根筛选，在区间隔离精化中设置 10^{-12} 的截断精度，以便比较不同方法的稳定性与精度。结果表明：在实根简单且分离良好的场景下，Chebyshev–-Sturm 方法精度优势最明显，同事矩阵特征值法次之，而单项式基底下的伴随矩阵特征值法在病态多项式上误差更大；当出现重根或极近根簇时，三类方法都会受到问题病态性的显著影响，双精度下难以稳定地区分相邻根，甚至可能发生漏根。对光滑函数算例，Boyd–Battles 方法能够稳定给出残差接近机器精度量级的根，但根位置精度仍会受近重根与插值质量的限制。本文的对比结果与讨论为多实根求解中的基底选择、区间策略与数值验证提供了实践参考。

关键词： 多实根求解；多项式求根；同事矩阵特征值法；Sturm 序列方法；Boyd–Battles 方法

目录

1 引入

我们已经有非常成熟的求解连续函数零点问题单实根的方法，如二分法、不动点迭代法、Newton 方法和拟 Newton 方法。

在工程上，很多地方我们并不只是希望得到函数的单个根，我们想得到函数的所有实根。如控制系统稳定性分析的 Routh–Hurwitz 判据需要判断特征多项式所有根是否落在左半平面 [18]；连续搅拌釜式反应器（Continuous Stirred Tank Reactor，CSTR）等非线性过程可能出现多稳态点，需要找出全部稳态解以分析可操作性与失稳风险 [7]；机器人逆运动学常具有多组构型解，实际规划与避障需要系统地得到全部可行实解 [11]。

遗憾的是，我们很难简单地将上述的这些单根求解方法推广到多根求解上，因为

- 闭区间套只能收敛到唯一点，即使初始区间上可能有多个根；
- 压缩映射的不动点是唯一的，对应区间上不可能有多个根；
- Newton 方法和拟 Newton 方法都是单点迭代方法，无法得到多个根。

可见多实根求解算法的一大困难是：如何确定实数域上根的数量以及根的存在区间。

本文我们讨论的函数基于比较好的性质，构造适合求解函数多个实根的方法，

1. $f(x)$ 为多项式，设计一个稳定的方法找出 $f(x)=0$ 的全部实根；
2. $f(x)$ 为光滑函数，设计一个稳定的方法找出 $f(x)=0$ 的全部实根。

简单起见，我们将函数零点问题的解称为函数的根。

2 多项式的多实根求解方法

对于四次以下的多项式，我们有求根公式直接计算多项式的根。然而，Abel–Ruffini 定理表明：一般五次及以上代数方程不可由根式表示 [22]。因此，我们没有通用的简单公式计算高次多项式的实根。

为了在数值上找多项式的所有实根，自然地有两条实现路径，

1. 利用多项式的代数结构，找出多项式的所有根（包括实根和复根），再把复根剔除；
2. 先确定多项式的单根区间，即划分出一个区间族使得多项式在每一个区间内只有一个实根，再利用二分法高效求解对应区间内的实根。

前者的好处是，可以同时知道复根以及根的重数，而后者通常做不到；但是后者在正确划分单根区间的条件下可以将根的精度做到接近机器精度。

我们下面将分别以**伴随矩阵特征值方法**和 **Sturm 序列方法**为代表，介绍这两类算法。考虑到多项式的根关于单项式基底下的多项式系数是不稳定的，我们不得不考虑多项式空间上更好的基底，如**正交多项式**。

关于多项式方程组，我们还补充介绍了**同伦算法**的有关理论和算法，见附录 A。我们不会对其进行数值实验。

2.1 伴随矩阵特征值方法

在线性代数中，我们知道，给定一个多项式在单项式基底下的表示，不妨假设 $p(x)$ 在单项式基底下是首一的，

$$p(x) = x^n + c_{n-1}x^{n-1} + \cdots + c_0,$$

则 $p(x)$ 的所有根对应伴随矩阵（companion matrix）

$$\begin{bmatrix} 0 & 0 & \cdots & 0 & -c_0 \\ 1 & 0 & \cdots & 0 & -c_1 \\ 0 & 1 & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & -c_{n-2} \\ 0 & \cdots & 0 & 1 & -c_{n-1} \end{bmatrix}$$

的所有特征值（按重数计）。MATLAB 内置的 `roots` 函数就是通过这种方式求解多项式的根。

遗憾的是，这不是一个稳定的数值方法。虽然我们有稳定的求矩阵特征值算法（如 QR 方法），但是多项式在单项式基底下的求根问题本身是一个病态的问题（病态程度取决于具体的多项式）。定理 2.1 给出了这个问题的条件数。

定理 2.1 (Kalluci–Hoxha [10])**.** *设*

$$p(x) = \sum_{j=0}^{n} c_j x^j$$

是一个 n 次多项式，若 r 是多项式 $p(x)$ 的一个非零单根且重数为 1，并且 $c_j \neq 0$，则 r 对系数 c_j 的相对条件数为

$$\kappa = \frac{|c_j r^{j-1}|}{|p'(r)|}.$$

经典的病态算例是 Wilkinson 多项式，

$$W(x) = \prod_{i=1}^{20}(x - i),$$

Wilkinson 多项式的图像如图 1 所示，其中纵坐标用 symlog 尺度绘制，当 $|W(x)| \leq 10^{-4}$ 时坐标尺度是线性的，当 $|W(x)| > 10^{-4}$ 的部分则是对数尺度的。从图上可以看到，在根附近 Wilkinson 多项式的变化非常剧烈。

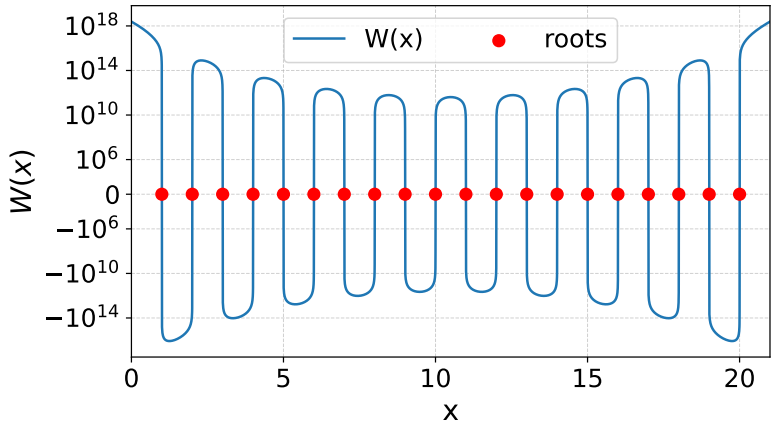


图 1: Wilkinson 多项式的图像

由定理 2.1，Wilkinson 多项式的单根 $r = 20$ 对系数 $c_{20} = 1$ 的相对条件数为

$$\kappa = \frac{20^{19}}{19!} \approx 4.31 \times 10^{7}.$$

利用 MATLAB 的 `roots` 函数，得到的 Wilkinson 多项式所有根的实部为

$$1.0000, 2.0000, 3.0000, 4.0000, 5.0000, 6.0000, 7.0001, 7.9994, 9.0027, 9.9912,$$

$$11.0225, 11.9589, 13.0627, 13.9302, 15.0593, 15.9597, 17.0185, 17.9937, 19.0013, 19.9999,$$

可以看到多项式在单项式基底下利用伴随矩阵的求根算法是不稳定的。即使双精度的多项式系数有很小的相对误差，但是对于 Wilkinson 多项式，（双精度下的）多项式系数依然有不可忽略的绝对误差，再考虑到求伴随矩阵特征值的后向误差，最终导致计算 Wilkinson 多项式的根有较大的误差。

Wilkinson 多项式是一个比较病态的例子。事实上，实际应用中的多项式更多是良态的，相应求根问题的条件数通常比较小，用这个方法也可以得到误差比较小的根（这也是 MATLAB 依然内置这个算法函数的原因）。如果希望利用在单项式基底下求解 Wilkinson 多项式这样病态多项式的根，也可以利用更高精度的计算（如符号计算或 VPA）来实现小误差。当然，当我们对一个多项式了解的很少，我们无法预估需要多少的精度才能得到小于指定误差的根。

单项式基底下多项式求根问题病态的根本原因是，单项式基底是多项式空间中正交性很差的基底，导致系数空间到多项式空间的算子范数非常大，从而根对系数高度敏感。那么，如果换一个正交性更好的基底，比如 Chebyshev 多项式基底，那么在新基底下的多项式求根问题就应该是良态的。我们先简要介绍正交多项式的相关概念，再来讨论正交多项式基底下多项式求根问题的算法以及稳定性。

2.1.1 正交多项式

对于实系数多项式空间 $\mathbb{P}_n(x) = \left\{ f(x) = \sum_{k=0}^{n} a_k x^k : a_i \in \mathbb{R} \right\}$，可以定义内积

$$\langle f, g \rangle_w = \int_a^b f(x) g(x) w(x) \, \mathrm{d}x, \tag{1}$$

其中，$a, b \in \mathbb{R} \cup \{-\infty, \infty\}$，$w(x)$ 为权函数，至少要求在区间 (a, b) 上可积、$w(x) > 0$ 且具有有限零点。这样的内积称为在区间 $[a, b]$ 上关于权函数 $w(x)$ 的内积。

由内积，自然地，可以定义多项式空间的正交基底，即正交多项式 $\{P_i\}_{i=0}^{\infty}$。最基本的要求是，对任意 $i, j \in \mathbb{N}$，

$$\langle P_i, P_j \rangle_w = k_i \delta_{ij} = \begin{cases} k_i, & i = j, \\ 0, & i \neq j, \end{cases}$$

其中 $k_i = \int_a^b P_i(x)^2 w(x)\,\mathrm{d}x$ 是只与内积和 P_i 有关的常数。为方便起见，还可以要求对任意自然数 i，$\deg P_i = i$。

下面简要介绍常见的正交多项式。

Legendre 多项式 当把区间选为 $[-1, 1]$，权函数 $w(x) = 1$ 时，所对应的正交多项式就是 Legendre 多项式。它是最常见的正交多项式之一，可以表示为

$$P_k(x) = \frac{1}{2^k} \sum_{j=0}^{\lfloor k/2 \rfloor} (-1)^j C_k^j C_{2k-2j}^k x^{k-2j} = \frac{1}{2^k k!} \frac{\mathrm{d}^k}{\mathrm{d}x^k} \big(x^2 - 1\big)^k,$$

其中 C_k^j 表示组合数。Legendre 多项式对应的三项递推公式为

$$P_{k+1}(x) = \frac{2k+1}{k+1} x P_k(x) - \frac{k}{k+1} P_{k-1}(x), \quad P_0(x) = 1, \quad P_1(x) = x.$$

Chebyshev 多项式 当区间选为 $[-1, 1]$，权函数 $w(x) = \dfrac{1}{\sqrt{1-x^2}}$ 时，所得的正交多项式就是 Chebyshev 多项式。它是定义在 $[-1, 1]$ 上的另一类常见的正交多项式。Chebyshev 多项式可以表示为

$$T_k(x) = \cos\big(k \arccos x\big).$$

Chebyshev 多项式对应的三项递推公式为

$$T_{k+1}(x) = 2x T_k(x) - T_{k-1}(x), \qquad T_0(x) = 1, \quad T_1(x) = x.$$

Chebyshev 多项式 T_k $(k \geq 1)$ 的首项系数为 2^{k-1}，且有如下内积公式

$$\langle T_n, T_n \rangle_\omega = \begin{cases} \dfrac{\pi}{2}, & n \neq 0, \\ \pi, & n = 0. \end{cases}$$

Legendre 和 Chebyshev 多项式都是 Jacobi 多项式的特例。Jacobi 多项式是权函数 $(1+x)^\alpha (1-x)^\beta$ 对应的正交多项式，其中 α 和 β 均大于 -1。当 $\alpha = \beta = 0$ 时，Jacobi 多项式就是 Legendre 多项式；当 $\alpha = \beta = -\frac{1}{2}$ 时，Jacobi 多项式就是 Chebyshev 多项式。Jacobi 多项式在物理学、概率论、逼近论等领域都有广泛的应用。

Laguerre 多项式 当区间选为 $[0, +\infty)$，权函数 $w(x) = e^{-x}$ 时，所得的正交多项式称为 Laguerre 多项式

$$L_k(x) = \frac{e^x}{k!} \frac{\mathrm{d}^k}{\mathrm{d}x^k} \big(e^{-x} x^k\big).$$

Laguerre 多项式满足三项递推公式

$$(k+1) L_{k+1}(x) = (2k + 1 - x) L_k(x) - k L_{k-1}(x), \quad L_0(x) = 1, \quad L_1(x) = 1 - x.$$

Laguerre 多项式在 Gauss–Laguerre 求积等带指数权函数的积分近似中经常出现。

Hermite 多项式 当区间选为 $(-\infty,+\infty)$，权函数 $\rho(x)=e^{-x^2}$ 时，所得的正交多项式称为 Hermite 多项式

$$H_k(x)=(-1)^k e^{x^2}\frac{\mathrm{d}^k}{\mathrm{d}x^k}e^{-x^2}.$$

Hermite 多项式的三项递推公式为

$$H_{k+1}(x)=2xH_k(x)-2kH_{k-1}(x),\quad H_0(x)=1,\quad H_1(x)=2x.$$

Hermite 多项式也有着广泛的应用。如在量子力学中是调和振动子的能量本征态，在概率论和统计学中，它们可以用于描述正态分布的概率密度函数。

下面我们着重讨论 Chebyshev 多项式作为正交基底的情形。对于其他正交多项式，也有类似的构造和结论。

2.1.2 Chebyshev 基底下的伴随矩阵特征值方法

正如单项式基底下的多项式需要借助伴随矩阵求根，Chebyshev 多项式基底下的多项式也需要找到一个相同地位的矩阵。这样的矩阵是存在的，并称之为同事矩阵（colleague matrix）。这一思想最早由 Specht [20] 提出，而 Good [8] 将这种方法系统化，并命名了同事矩阵。下面先给出同事矩阵的定义，然后通过定理阐明同事矩阵的特征值与多项式根的关系。

定义 2.1 (同事矩阵, Good [8])**.** 不妨假设多项式在 Chebyshev 多项式基底下是首一的，即

$$p(x)=T_n(x)+\sum_{k=0}^{n-1}a_kT_k(x),\quad \text{with } a_k\in\mathbb{R},\quad \text{for } k=0,1,\dots,n-1,$$

其中 $T_k(x)$ 是 k 次（第一类）Chebyshev 多项式，那么矩阵

$$C_T=\frac{1}{2}\begin{bmatrix}-a_{n-1} & -a_{n-2}+1 & -a_{n-3} & \cdots & -a_2 & -a_1 & -a_0\\ 1 & 0 & 1 & 0 & \cdots & \cdots & 0\\ 0 & 1 & 0 & 1 & \ddots & & \vdots\\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots\\ \vdots & & \ddots & \ddots & \ddots & \ddots & 0\\ 0 & \cdots & \cdots & 0 & 1 & 0 & 1\\ 0 & \cdots & \cdots & \cdots & 0 & 2 & 0\end{bmatrix}\in\mathbb{R}^{n\times n},$$

称为多项式 $p(x)$ 的同事矩阵（colleague matrix）。

定理 2.2 (Specht [20], Good [8])**.** *若 $p(x)$ 在 Chebyshev 多项式基底下首一，则同事矩阵 C_T 满足*

$$\det(xI-C_T)=\frac{1}{2^{n-1}}p(x),\tag{2}$$

从而 C_T 的特征值与多项式 $p(x)$ 的根一一对应（计重数）。

证明. 令 $A(x):=2(xI-C_T)=2xI-2C_T$，于是

$$\det(xI-C_T)=2^{-n}\det A(x).$$

因此只要证明 $\det A(x) = 2p(x)$，就能得到

$$\det(xI - C_T) = 2^{-n} \cdot 2p(x) = \frac{1}{2^{n-1}} p(x).$$

下面通过 Laplace 展开来计算 $\det A(x)$。

记 $C_{1j}(x)$ 为 $A(x)$ 第一行第 j 列的代数余子式，要证明 $C_{1j}(x) = 2T_{n-j}(x)$。一方面，令

$$M_m(x) \coloneqq \begin{bmatrix} 2x & -1 & & & \\ -1 & 2x & -1 & & \\ & \ddots & \ddots & -1 & \\ & & -1 & 2x & -1 \\ & & & -2 & 2x \end{bmatrix}, \quad D_m(x) := \det M_m(x),$$

并约定 $M_0(x) = 2$，对 $m \geq 2$，将 $M_m(x)$ 沿第一行展开，得

$$D_m(x) = 2xD_{m-1}(x) - D_{m-2}(x), \quad D_0(x) := 2, \quad D_1(x) = 2x.$$

另一方面，对 $m \geq 2$，第一类 Chebyshev 多项式满足

$$T_m(x) = 2xT_{m-1}(x) - T_{m-2}(x), \quad T_0(x) = 1, \quad T_1(x) = x,$$

所以 $D_m(x) = 2T_m(x)$。再说明 $C_{1j}(x) = D_{n-j}(x)$。对 $A(x)$，删去第 1 行与第 j 列后，得到的子矩阵 M_{1j} 可以做分块

$$M_{1j}(x) = \left[\begin{array}{c|c} U_{j-1}(x) & E_{j-1,n-j} \\ \hline 0 & H_{n-j}(x) \end{array}\right],$$

其中各个子块为

$$U_{j-1}(x) = \begin{bmatrix} -1 & 2x & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2x & -1 & \ddots & \vdots \\ 0 & 0 & -1 & 2x & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & -1 \\ 0 & 0 & \cdots & 0 & -1 & 2x \\ 0 & 0 & \cdots & \cdots & 0 & -1 \end{bmatrix},$$

$$H_{n-j}(x) = \begin{bmatrix} 2x & -1 & 0 & \cdots & 0 \\ -1 & 2x & -1 & \ddots & \vdots \\ 0 & -1 & 2x & \ddots & 0 \\ \vdots & \ddots & \ddots & 2x & -1 \\ 0 & \cdots & 0 & -2 & 2x \end{bmatrix} = M_{n-j}(x), \quad E_{j-1,n-j} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 \\ -1 & 0 & \cdots & 0 \end{bmatrix},$$

于是得到 $C_{1j}(x) = (-1)^{j+1} \cdot (-1)^{j-1} D_{n-j}(x) = 2T_{n-j}(x)$。

由 $A(x)$ 沿第一行的 Laplace 展开，

$$\begin{aligned}\det A(x) &= \sum_{j=1}^{n} A_{1j}(x)C_{1j}(x) \\ &= (2x + a_{n-1})2T_{n-1} + (a_{n-2} - 1)2T_{n-2} + \sum_{k=0}^{n-3} a_k 2T_k \\ &= 2(2xT_{n-1} - T_{n-2}) + 2\sum_{k=0}^{n-1} a_k T_k \\ &= 2\left(T_n(x) + \sum_{k=0}^{n-1} a_k T_k(x)\right) = 2p(x),\end{aligned}$$

所以

$$\det(xI - C_T) = 2^{-n} \det A(x) = 2^{-n} \cdot 2p(x) = \frac{1}{2^{n-1}} p(x). \tag{3}$$

最后讨论多项式根的重数与同事矩阵特征值的代数重数的关系。若 λ 是 $p(x)$ 的 m 重根，即

$$p(x) = (x - \lambda)^m q(x), \quad q(\lambda) \neq 0,$$

代入 (3)，得

$$\det(xI - C_T) = \frac{1}{2^{n-1}}(x - \lambda)^m q(x),$$

因此 λ 作为特征值在特征多项式中的重数正好是 m。 □

注. 可以看到，同事矩阵是一个上 Hessenberg 矩阵，有稳定的求矩阵特征值算法。

注. 事实上，若只需证明同事矩阵 C_T 的特征值对应 Chebyshev 基底下的首一多项式 $p(x)$ 的根，则有更简单且更有启发性的证明，见附录 B 定理 B.1 的证明。

定理 2.2 告诉我们，对于 Chebyshev 多项式基底下表示的多项式，只需要求对应同事矩阵的特征值，即可得到多项式的根。其中，求同事矩阵特征值的稳定算法包括 Hessenberg-QR、QZ [14] 这样的通用特征值算法，以及 Eidelman–Gemignani–Gohberg 型结构 QR [6]、Serkh–Rokhlin [19] 等利用同事矩阵特殊结构的专门算法。

在开源 MATLAB 软件包 Chebfun 上，就是在 Chebyshev 多项式基底下利用同事矩阵特征值来求解多项式的根。事实上，Chebfun 也能用类似的方法求解一般函数的根，我们将会在光滑函数的多实根求解算法中讨论。

2.1.3 稳定性分析

考虑 Chebyshev 基底 $\{T_k\}_{k\geq 0}$ 下的首一标量多项式

$$p(x) = T_n(x) + \sum_{k=0}^{n-1} a_k\, T_k(x), \qquad a_k \in \mathbb{R},$$

并令 $C_T \in \mathbb{R}^{n\times n}$ 为与 p 对应的同事矩阵（colleague matrix）。同事矩阵特征值法以 C_T 的特征值作为 p 的零点近似，因此其稳定性分析的关键在于：将特征值算法在舍入误差下产生的矩阵后向误差（扰动矩阵 E）映射为 p 的 Chebyshev 系数后向误差。

第一步给出“矩阵扰动 $\Rightarrow$ 多项式扰动”的一阶刻画。由于 $\det(xI-(C_T+E))$ 与 C_T+E 的特征值具有同一零点集合，我们引入按常数因子归一化后的多项式

$$p_E(x) := 2^{\,n-1}\det\big(xI-(C_T+E)\big),$$

其中常数因子 2^{n-1} 不改变零点，并使 p_E 在 Chebyshev 基底下保持首一形式。

定理 2.3 (Noferini–Pérez [16])**.** *设 $p(x)=T_n(x)+\sum_{k=0}^{n-1}a_k\,T_k(x)$ 为 Chebyshev 基底下的首一标量多项式，$C_T\in\mathbb{R}^{n\times n}$ 为其同事矩阵。对任意 $E\in\mathbb{R}^{n\times n}$，令*

$$p_E(x) := 2^{\,n-1}\det\big(xI-(C_T+E)\big).$$

则：

1. *C_T+E 的全部特征值（按代数重数计）恰为多项式 $p_E(x)$ 的全部零点。*
2. *令 $M_1,\dots,M_n$ 为与 p 相关的 Clenshaw 矩阵，使得*

$$\operatorname{adj}(xI-C_T)=2^{-(n-1)}\sum_{k=0}^{n-1}M_{k+1}\,T_k(x).$$

当 $\|E\|_F\to 0$ 时，有一阶展开

$$p_E(x)=p(x)\;-\;\operatorname{tr}\Big(\Big(\sum_{k=0}^{n-1}M_{k+1}T_k(x)\Big)E\Big)\;+\;\mathcal{O}(\|E\|_F^2),$$

从而

$$p_E(x)=T_n(x)+\sum_{k=0}^{n-1}\Big(a_k-\operatorname{tr}(M_{k+1}E)\Big)T_k(x)\;+\;\mathcal{O}(\|E\|_F^2).$$

等价地，存在 $\delta a_k(p,E)$ 使得

$$p_E(x)-p(x)=\sum_{k=0}^{n-1}\delta a_k(p,E)\,T_k(x)+\mathcal{O}(\|E\|_F^2),\qquad \delta a_k(p,E)=-\operatorname{tr}(M_{k+1}E).$$

3. *对每个 k，当 E 固定时，$\delta a_k(p,E)$ 关于 $(a_0,\dots,a_{n-1})$ 为仿射函数；当 p 固定时，$\delta a_k(p,E)$ 关于 E 的各元素为仿射函数。*

上述定理说明：若特征值计算等价于对 C_T 施加一个小扰动 E，则计算得到的根集合是某个首一 Chebyshev 多项式 p_E 的精确零点，并且 p_E 与原多项式 p 的差异可用 $\operatorname{tr}(M_{k+1}E)$ 在一阶上刻画。

第二步将 E 的规模与实际特征值算法的舍入误差联系起来。对一般后向稳定的特征值算法，其返回值可解释为某个 C_T+E 的精确特征值，其中 $\|E\|_F$ 与机器精度 u 同阶。结合上一结论可得到同事矩阵特征值法在 Chebyshev 系数意义下的多项式后向误差界。

定理 2.4 (Noferini–Pérez [16])**.** *设 $p(x)=T_n(x)+\sum_{k=0}^{n-1}a_kT_k(x)$，$C_T$ 为其同事矩阵。若采用一个后向稳定的特征值算法计算 C_T 的特征值，则算法返回的是某个扰动矩阵 C_T+E 的精确特征值，并满足*

$$\|E\|_F\le\varepsilon\,\|C_T\|_F,\qquad \varepsilon=\mathcal{O}(u).$$

令

$$\widetilde{p}(x) := 2^{\,n-1}\det\big(xI-(C_T+E)\big)=T_n(x)+\sum_{k=0}^{n-1}\widetilde{a}_k\,T_k(x),$$

则在忽略 $\mathcal{O}(u^2)$ 项时，计算得到的根集合恰为 $\widetilde{p}$ 的精确零点，并且存在仅依赖于 n 的多项式函数 $\kappa(n)$（系数为适度常数），使得

$$\|\widetilde{p}-p\|_2 \le \kappa(n)\,u\,\|p\|_2^2, \qquad \frac{\|\widetilde{p}-p\|_2}{\|p\|_2} \le \kappa(n)\,u\,\|p\|_2.$$

其中 $\|\cdot\|_2$ 表示 *Chebyshev* 系数向量的 *Euclidean* 范数。

最后注意到同事矩阵 C_T 具有“对称三对角矩阵 + 秩一修正”的显式结构。若特征值算法在计算过程中保持并利用该结构，则可得到比一般（非结构化）QR 更强的后向误差控制。结合 [17] 对结构化后向误差到多项式系数误差的映射，可推出 Chebyshev 系数向量的最优量级后向误差。

定理 2.5 (Noferini–Robol–Vandebril [17]; Serkh–Rokhlin [19])**.** 设 $c=(a_0,\dots,a_{n-1},1)^\top$ 为 $p(x)=\sum_{k=0}^{n} c_k T_k(x)$ 的 *Chebyshev* 系数向量（其中 $c_n=1$）。若对同事矩阵 C_T 采用保持其“对称三对角 + 秩一”结构的显式结构化 *QR* 算法，并且该算法在结构数据意义下是分量后向稳定的（例如 *[19]*），则存在系数扰动向量 δc 使得计算得到的根集合恰为

$$\widehat{p}(x)=\sum_{k=0}^{n}(c_k+\delta c_k)T_k(x)$$

的精确零点，并满足

$$\|\delta c\|_2 \lesssim u\,\|c\|_2.$$

对于其他正交多项式，也可以构造相同地位的矩阵，称为战友矩阵（comrade matrix），并与伴随矩阵、同事矩阵合称为同盟伴随矩阵（confederate matrix）[2][12]。我们将这一类方法统称为伴随矩阵特征值方法（companion-matrix eigenvalue method）。由 Nakatsukasa and Noferini [14]，其他正交多项式的伴随矩阵特征值方法应用 QZ 特征值算法也具有相应的稳定性。

如果我们知道多项式在 Chebyshev 多项式基底下的多项式系数（下面将简称为 Chebyshev 系数），那么我们可以用同事矩阵特征值方法稳定求解多项式的所有实根（事实上也包括复根）。然而在应用中，得到多项式在 Chebyshev 系数并不是显然的事情。先写出多项式在单项式基底下的多项式系数再利用多项式除法计算是不稳定的，因为这个过程已经将问题变为了单项式基底表示下的多项式求根问题。有一些手段可以直接地得到多项式在 Chebyshev 系数，如离散余弦变换。对于 Wilkinson 多项式这种病态多项式，利用缩放技巧（将根界映射到 $[-1,1]$ 上）再计算 Chebyshev 系数能显著地提高计算根的准确率。我们在数值实验中也基于此来得到一般多项式的 Chebyshev 系数。

2.2 Sturm 序列方法

Sturm 序列方法是一个典型的先分离根再精化策略的求根方法，其中最关键的是如何根分离。下面将先介绍关于 Sturm 序列的有关数学理论，解释为什么 Sturm 序列能划分单根区间；再给出 Sturm 序列的算法框架以及如何将其应用在 Chebyshev 多项式基底上以保证稳定性。

2.2.1 Sturm 序列的数学理论

定义 2.2 (Sturm 序列)**.** 对于无重根多项式 $f(x)$，考虑通过辗转相除法递归定义 Sturm 序列：

$$f_{k-1}(x)=q_k(x)f_k(x)-f_{k+1}(x), \quad \deg f_k > \deg f_{k+1} \ge 0, \tag{4}$$

其中 $f_0(x)=f(x)$，$f_1(x)=f'(x)$。由此得到一列 $\{f_k(x)\}_{k=0}^m$，我们称 $(f_0,f_1,\cdots,f_m)$ 为关于 $f(x)$ 的 Sturm 序列。

容易得到，对于无重根多项式 $f(x)$ 的 Sturm 序列 $(f_0,f_1,\cdots,f_m)$ 有 $f_m(x)$ 为非零常数。

定义 2.3 (变号数)**.** 对于一非恒为零的实数列 $(a_1,a_2,\cdots,a_n)$，考虑剔除数列中为 0 的元素得到一非零数列 $(b_1,b_2,\cdots,b_r)$ $(r\le n)$，那么 $(a_1,a_2,\cdots,a_n)$ 的变号数

$$V(a_1,\ldots,a_n):=\#\{\,i\in\{1,\ldots,r-1\}\mid b_ib_{i+1}<0\,\}.$$

特别地，对于 $V(f_0(x),f_1(x),\cdots,f_m(x))$，简记为 $V(x)$。

有了以上定义我们可以陈述 Sturm 定理。

定理 2.6 (Sturm [21])**.** *无重根多项式 $f(x)$ 在区间 $[a,\,b]$ 有 $f(a)\neq 0, f(b)\neq 0$，其 Sturm 序列为 $(f_0,f_1,\cdots,f_m)$，变号数为 $V(x)=V(f_0(x),f_1(x),\cdots,f_m(x))$，则 $f(x)$ 在 $[a,\,b]$ 上根的个数为 $V(a)-V(b)$。*

证明. 定理的证明思路分为三部分：

Step 1：$x_0\in\mathbb{R}$ 不为 $f_k(x)$ 的根 $(\forall\, k\in[0,\,m])$，则 $V(x)$ 在 x_0 的某个邻域上恒为常数；

Step 2：$x_0\in\mathbb{R}$，$\exists\, k\ge 1$ 使得 $f(x_0)\neq 0$，$f_k(x_0)=0$，则 $V(x)$ 在 x_0 的某个邻域上恒为常数；

Step 3：x_0 为 $f(x)$ 的根，则 $V(x_0^-)-V(x_0^+)=1$。

我们不妨假设 $f(x)$ 在 $[a,b]$ 上的根分别为 $x_1<x_2<\cdots<x_n$，通过证明上述三步，我们可以得到变号数 $V(x)$ 只在 $f(x)$ 的根 x_0 的附近有 $V(x_0^-)-V(x_0^+)=1$，非根区域上变号数 $V(x)$ 恒为常数，由于多项式的根具有离散性，因此 $V(x)$ 只在根的附近跃升，亦即

$$V(x)=V(x_i^+)=V(x_{i+1}^-),\quad V(a)=V(x_1^-),\quad V(b)=V(x_n^+). \tag{5}$$

因此

$$\begin{aligned}V(a)-V(b)&=V(a)-V(x_1^-)+V(x_1^-)-V(x_1^+)+\cdots+V(x_n^-)-V(x_n^+)+V(x_n^+)-V(b)\\&=\sum_{i=1}^{n}[V(x_i^-)-V(x_i^+)]\\&=\#f^{-1}(0).\end{aligned}$$

第一步的证明 由于 x_0 不为 $f_k(x)$ $(k=0,1,2,\cdots,m)$ 的根，根据 f_k 的连续性，存在的 x_0 一个邻域 U_k，使得在邻域 U_k 上非零且不变号，取 $U=\cap_{k=1}^n U_k$，于是 $(f_0(x),\,f_1(x),\cdots,\,f_m(x))$ 在 U 上不变号，因此变号数 $V(x)=V(f_0(x),\,f_1(x),\cdots,\,f_m(x))$ 在 U 上恒为常数.

第二步的证明 对于 $x_0\in\mathbb{R}$，$\exists\, k\ge 1$ 使得 $f(x_0)\neq 0$，$f_k(x_0)=0$，首先我们说明 x_0 不能使得相邻两项的函数值同时为 0。假设 $f_k(x_0)=f_{k+1}(x_0)=0$，那么根据 Sturm 序列的形式

$$f_{k-1}(x)=q_k(x)f_k(x)-f_{k+1}(x),\ \deg f_k>\deg f_{k+1}\ge 0, \tag{6}$$

于是

$$f_0(x_0)=f_1(x_0)=\cdots=f_m(x_0)=0$$

与假设矛盾. 由此对于 $f_k(x_0)=0$ 有相邻项 $f_{k-1}(x_0)$，$f_{k+1}(x_0)$ 非零，我们再说明对于 $\widetilde{V}(x)=V(f_{k-1}(x),\,f_k(x),\,f_{k+1}(x))$，存在 x_0 的邻域使得 $\widetilde{V}(x)$ 在其上不变号. 根据 Sturm 序列的形式，

我们有 $f_{k-1}(x_0) = -f_{k+1}(x_0)$，不妨设 $f_{k-1}(x_0) > 0$（对于另一情况是类似的），对于 $\widetilde{V}(x)$ 在 x_0 附近的变化，

1. $f_k(x)$ 在 x_0 的邻域 U 上不变号。若 $f_k(x) \geq 0$，显然有 $\widetilde{V}(U) = 1$；若 $f_k(x) \leq 0$，同样有 $\widetilde{V}(U) = 1$。
2. $f_k(x)$ 在 x_0 的邻域 U 上变号，其中在单侧子邻域上不变号。若 $f_k(x)$ 在 $U_0^+(x_0)$ 恒正，容易得到 $\widetilde{V}(U) = 1$；若 $f_k(x)$ 在 $U_0^+(x_0)$ 恒负，同理可得 $\widetilde{V}(U) = 1$。

根据第一步证明的方法，对每个 $f_k(x_0) \neq 0$，存在的 x_0 一个邻域 U_k，使得在邻域 U_k 上非零且不变号，取这些邻域与上述每个 $\widetilde{V}(x)$ 所得邻域的交即可。

第三步的证明 首先，由于 $f(x)$ 无重根，因此 $f(x)$ 与 $f'(x)$ 互素，那么 $f'(x_0) \neq 0$. 根据第二步中的证明可知 $f_k(x)(k \geq 1)$ 的零点附近 $\widetilde{V}(x) = V(f_{k-1}(x),\ f_k(x),\ f_{k+1}(x))$ 为常数，因此我们只需要考虑 $\widehat{V}(x) = V(f_(x_0), f_1(x_0))$ 并说明 $\widehat{V}(x)$ 在 x_0 有 $\widehat{V}(x_0^-) - \widehat{V}(x_0^+) = 1$ 即可。

由于多项式的根有离散性，因此存在 x_0 的邻域 $U(x_0)$ 使得 $f(x)$ 在 $U(x_0)$ 上变号，不妨假设 $f(x)$ 在 $U_0^+(x_0)$ 上恒正，$f(x)$ 在 $U_0^-(x_0)$ 上恒负。利用 $f'(x_0) = \lim_{h\to 0} \frac{f(x_0+h)-f(x_0)}{h}$ 可得 $f'(x_0) > 0$，于是根据 f' 的连续性，$\widehat{V}(x_0^-) = 1, \widehat{V}(x_0^+) = 0$，于是 $\widehat{V}(x_0^-) - \widehat{V}(x_0^+) = 1$. 对于 f 的另一种变号情况是完全类似的，于是我们不多赘述.

□

推论. *无重根多项式 $f(x)$ 有 Sturm 序列变号数 $V(x)$，则 $f(x)$ 在 $\mathbb{R}$ 上根的个数为*

$$\#f^{-1}(0) = V(-\infty) - V(+\infty).$$

基于上述定理，推广至一般情形是容易的。

定理 2.7. *对于多项式 $f(x)$, 令 $f_0 = f$, $f_1 = \gcd(f_0, f_0')$, $f_2 = \gcd(f_1, f_1'), \cdots, f_m = \gcd(f_{m-1}, f_{m-1}')$, 由此得到一列 $(f_0, \cdots, f_m)$，定义 $g_k = \frac{f_{k-1}}{f_k}$ $(1 \leq k \leq m)$, 其中每个 g_k 的 Sturm 序列变号数为 $V_{g_k}(x)$ 则 $f(x)$ 在 $\mathbb{R}$ 上的根的个数（计重数）为*

$$\#f^{-1}(0) = \sum_{k=1}^{m} [V_{g_k}(-\infty) - V_{g_k}(+\infty)].$$

证明. 设 $f(x) = \prod_{i=1}^{n}(x - c_i)^{m_i}$，那么容易得到

$$f_k(x) = \prod_{i=1}^{n}(x - c_i)^{\max\{m_i - k,\ 0\}}$$

于是

$$g_k(x) = \prod_{i=1}^{n}(x - c_i)^{\chi_{\mathbb{Z}^+}(m_i - k)}$$

容易知道每个 g_k 均为无重根多项式，且 $f(x) = \prod_{i=1}^{m} g_i(x)$，故自然有

$$\#f^{-1}(0) = \sum_{k=1}^{m} [V_{g_k}(-\infty) - V_{g_k}(+\infty)].$$

□

求解最大公因式总是繁琐的，我们不希望增加这部分的工作量。事实上，对于一般多项式 $f(x)$，考虑其重根 $x=a$ 附近的变号数，在 a 的充分小邻域上（即 $f(x)$ 在这上只有一个根），有

$$f(x)=(x-a)^m g(x),\quad g(a)\neq 0,$$

对于其导数 $f'(x)$ 有

$$f'(x)=m(x-a)^{m-1}g(x)+(x-a)^m g'(x)=(x-a)^{m-1}h(x),\quad h(a)\neq 0.$$

因此在 $x=a$ 的两侧，Sturm 序列变号一次，于是有

$$V(a^-)-V(a^+)=1,$$

故对于一般多项式，我们有如下 Sturm 定理。

定理 2.8. *任意多项式 $f(x)$ 有 Sturm 序列变号数 $V(x)$，则 $f(x)$ 在 $\mathbb{R}$ 上根的个数（不计重数）为*

$$\#f^{-1}(0)=V(-\infty)-V(+\infty).$$

在实际计算中，计算 $V(\infty)$ 是不容易的。为了确定包含所有根的初始搜索区间，我们可以考虑如下定理。

定理 2.9 (Cauchy 界)**.** *对于多项式 $f(z)=a_nz^n+a_{n-1}z^{n-1}+\cdots+a_1z+a_0$，$f(z)=0$ 的根 z_0 满足*

$$|z_0|<M+1,\quad M=\max_{0\le k\le n}\left|\frac{a_k}{a_n}\right|.$$

证明. 假设 $z_0=\max_{\{f(z)=0\}}|z|>0$,

$$f(z_0)=0\Longleftrightarrow a_nz_0^n=-(a_{n-1}z_0^{n-1}+\cdots+a_0).$$

等式两端取绝对值:

$$|a_n||z_0|^n=|a_{n-1}z_0^{n-1}+\cdots+a_0|\le|a_{n-1}||z_0|^{n-1}+\cdots+|a_0|.$$

设 $|z_0|=R>0$，两边同除以 $|a_n|R^{n-1}$ 得

$$R\le\left|\frac{a_{n-1}}{a_n}\right|+\frac{1}{R}\left|\frac{a_{n-2}}{a_n}\right|+\cdots+\frac{1}{R^{n-1}}\left|\frac{a_0}{a_n}\right|\le M(1+\cdots+\frac{1}{R^{n-1}})<M(1+\frac{1}{R-1})$$

若 $R\ge M+1$，则

$$R<M(1+\frac{1}{R-1})<M(1+\frac{1}{M})=M+1.$$

与假设矛盾，故 $R<M+1$. □

推论. *无重根多项式 $f(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_1x+a_0$ 有 Sturm 序列变号数 $V(x)$，其 Cauchy 界为 $1+M$, $M=\max_{0\le k\le n}|\frac{a_k}{a_n}|$，则 $f(x)$ 在 $\mathbb{R}$ 上根的个数为*

$$\#f^{-1}(0)=V(-(1+M))-V(1+M).$$

推论. *对于多项式 $f(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_1x+a_0$，其 Cauchy 界为 $1+M$, $M=\max_{0\le k\le n}|\frac{a_k}{a_n}|$，令 $f_0=f$, $f_1=\gcd(f_0,f_0')$, $f_2=\gcd(f_1,f_1')$, $\cdots$, $f_m=\gcd(f_{m-1},f_{m-1}')$, 由此得到一列 $(f_0,\cdots,f_m)$，定义 $g_k=\frac{f_{k-1}}{f_k}$ $(1\le k\le m)$, 其中每个 g_k 的 Sturm 序列变号数为 $V_{g_k}(x)$ 则 $f(x)$ 在 $\mathbb{R}$ 上的根的个数（计重数）为*

$$\#f^{-1}(0)=\sum_{k=1}^{m}[V_{g_k}(-(1+M))-V_{g_k}(1+M)].$$

对于一般多项式的根个数（不计重数），已囊括在上述定理中，因此我们不过多赘述。

2.2.2 Sturm 序列方法的算法框架

根据上述定理提供的算法的理论基础，现在简述 Sturm 序列方法求解多项式的根的算法步骤。值得注意的是，Sturm 序列方法不依赖于像二分法的端点异号，对重根一样可以实现分根；而根细化步骤也可以改进二分法（如 Sturm 计数二分）或使用 Newton 法使得端点同号的情况也能够处理。

初始化 初始化根存在区间，使得所有实根都落在这个区间内。当不了解多项式的根分布时，常用 Cauchy 根界

$$R = 1 + \max_{0 \le i \le n} \left| \frac{a_i}{a_n} \right|,$$

使得所有实根都满足 $|r| < R$。构建 Sturm 序列。

根隔离 对区间 $[l, r]$，计算 $N = V(l) - V(r)$。

1. 若 $N = 0$，则该区间无根，丢弃；
2. 若 $N = 1$，则这是一个单根区间，里面恰有一个实根；
3. 若 $N > 1$，则这个区间内有多个实根，把区间从中点分裂成 $[l, m], [m, r]$，继续递归子区间上的根隔离。

根精化 单根区间 $[l, r]$ 上恰有一个实根。利用二分法或 Newton 法（加速收敛的技巧，Sturm 序列的构造自然提供 $p'(x)$）来求得这个根。

2.2.3 Chebyshev–Sturm 方法

在之前的讨论中我们知道，多项式在单项式基底下的求根是不稳定的。所以，我们需要讨论 Sturm 序列方法是否可以与 Chebyshev 多项式基底兼容。主要是解决 Chebyshev 基底下的求导以及长除法的可行性。

考虑多项式在 Chebyshev 多项式基底下的表示

$$p(x) = \sum_{k=0}^{n} a_k T_k(x),$$

T_k 是第一类 Chebyshev 多项式，则导数为

$$p'(x) = \sum_{k=1}^{n} k a_k U_{k-1}(x),$$

$U_k(x)$ 是第二类 Chebyshev 多项式。而第二类 Chebyshev 多项式可以简单地展开成第一类 Chebyshev 多项式，

$$U_{k-1}(x) = \begin{cases} 2(T_{k-1}(x) + T_{k-3}(x) + \cdots + T_1(x)), & k \text{ 为偶数}, \\ T_0(x) + 2(T_{k-1}(x) + T_{k-3}(x) + \cdots + T_2(x)), & k \text{ 为奇数}, \end{cases}$$

也就是说，$p(x)$ 有稳定的求导格式。

考虑 Chebyshev 多项式基底下多项式的长除法。给定两个多项式

$$a(x)=\sum_{k=0}^{n_a} a_k T_k(x),\quad b(x)=\sum_{k=0}^{n_b} b_k T_k(x),$$

其中 $b_{n_b}\neq 0$，且 $n_a\geq n_b$。要求

$$q(x)=\sum_{j=0}^{n_q} q_j T_j(x)\quad r(x)=\sum_{k=0}^{n_b-1} r_k T_k(x),$$

使得 $a(x)=b(x)q(x)+r(x)$。等价于要求对任意 $k=n_b,n_b+1,\ldots,n_a$，

$$[a-bq]_{T_k}=0,$$

即 $a-bq$ 在 $T_{n_b},\ldots,T_{n_a}$ 这些高阶基函数上的系数全为零。这给出了 n_q+1 个线性方程，我们只要求出这个线性系统的矩阵，就可以求出系数向量 $q=[q_0,q_1,\ldots,q_{n_q}]^\top$。

由 Chebyshev 乘积恒等式

$$T_i(x)T_j(x)=\frac{1}{2}\left(T_{i+j}(x)+T_{|i-j|}(x)\right),$$

得到

$$b(x)T_j(x)=\sum_{i=0}^{n_b} b_i T_i(x)T_j(x)=\sum_{i=0}^{n_b} b_i\cdot\frac{1}{2}\left(T_{i+j}(x)+T_{|i-j|}(x)\right),$$

所以 $b(x)T_j(x)$ 在 T_k 的系数上的贡献为：当 $k=i+j$ 时，贡献 $\frac{1}{2}b_i$；当 $k=|i-j|$ 时，贡献为 $\frac{1}{2}b_i$。这样对每一个 j，都可以得到系数 $\{\beta_{k,j}\}_{k=0}^{n_a}$。

定义矩阵 $K\in\mathbb{R}^{(n_q+1)\times(n_q+1)}$

$$K_{\ell,j}=\beta_{n_b+\ell,j},\quad \ell,j=0,1,\ldots,n_q,$$

和右端向量 $u\in\mathbb{R}^{n_q+1}$

$$u_\ell=a_{n_b+\ell},\quad \ell=0,1,\ldots,n_q.$$

由条件

$$[a-bq]_{T_k}=0\Longleftrightarrow[bq]_{T_k}=a_k,$$

以及

$$[bq]_{T_k}=\sum_{j=0}^{n_q} q_j\beta_{k,j},$$

把 $k=n_b+\ell$ 代入就可以得到

$$\sum_{j=0}^{n_q} K_{\ell,j}q_j=u_\ell,$$

即 $Kq=u$。于是只需要求解这个线性系统就可以得到

$$q(x)=\sum_{j=0}^{n_q} q_j T_j(x),\quad r(x)=a(x)-b(x)q(x).$$

注意到我们做长除法只利用了 Chebyshev 的乘积恒等式性质，并没有将 Chebyshev 系数转化为单项式基底下的系数来处理。

于是，我们可以将单项式基底下的 Sturm 序列方法的框架应用到 Chebyshev 多项式基底的多项式上。我们将这个方法称为 Chebyshev–Sturm 方法。

2.3 数值实验

我们将使用四个算例来测试多项式求根算法的性能表现，

1. `poly_cyclotomic`：高次但实根很少的多项式
$$x^5 - 1;$$
2. `poly_wilkinson`：Wilkinson 多项式
$$W(x) = \prod_{i=1}^{20}(x - i),$$
多项式震荡剧烈，根关于单项式基底下的多项式系数不稳定；
3. `poly_quadruple_root`：多重根算例
$$x^4 - 8x^3 + 24x^2 - 32x + 16,$$
即 $(x-2)^4$ 的展开式；
4. `poly_mignotte_like`：极近根簇算例
$$x^4 - 2(10^4 x - 1)^2,$$
有两个实根距离非常近。

虽然这几个算例都没有直接地写成 Chebyshev 基底下的表示，但是我们会先用直接（不先写成单项式基底下的系数）且稳定的方法算出算例的 Chebyshev 系数（双精度），再分别用单项式基底下的伴随矩阵特征值法（记为 `poly`）、Chebyshev 基底下的同事矩阵特征值法（记为 `colleague`）以及 Chebyshev–Sturm 方法（记为 `cheb-sturm`）来计算算例多项式的根。

在实验中，`poly` 和 `colleague` 将使用 MATLAB 内置的 `eig` 函数分别求解伴随矩阵和同事矩阵的特征值（`eig` 是向后稳定的特征值算法函数）；当虚部充分小时认为是实根，阈值设置为 10^{-3}；`cheb-sturm` 的 `x_tol` 设置为 10^{-12}，即在单根区间上应用二分法或 Newton 法将根确定在长度小于 10^{-12} 的子区间上就会截断输出根；`colleague` 和 `cheb-sturm` 都会应用缩放技术，即先将算例多项式的根存在区间线性映射到 $[-1, 1]$，再计算 Chebyshev 多项式基底下的多项式系数，之后利用 `colleague` 或 `cheb-sturm` 求解到根后再映射回原始区间上。

对 `poly_cyclotomic` 算例 $x^5 - 1$，多项式有四个复根和一个实根。我们要测试算法能否正确区分实根和复根，以及复根的存在是否对实根计算的精度产生影响。计算结果如表 1 所示。

表 1: `poly_cyclotomic` 算例的求根结果与误差统计

真根	`poly`		`colleague`		`cheb-sturm`	
	根	绝对误差	根	绝对误差	根	绝对误差
1.0000	1.0000	2.2204×10^{-16}	1.0000	6.6613×10^{-16}	1.0000	4.5453×10^{-13}
统计指标						
最大绝对误差	—	2.2204×10^{-16}	—	6.6613×10^{-16}	—	4.5453×10^{-13}
平均绝对误差	—	2.2204×10^{-16}	—	6.6613×10^{-16}	—	4.5453×10^{-13}

可以看到，三种算法都找到 `poly_cyclotomic` 的唯一实根，正确剔除了复根，且计算精度

很高。

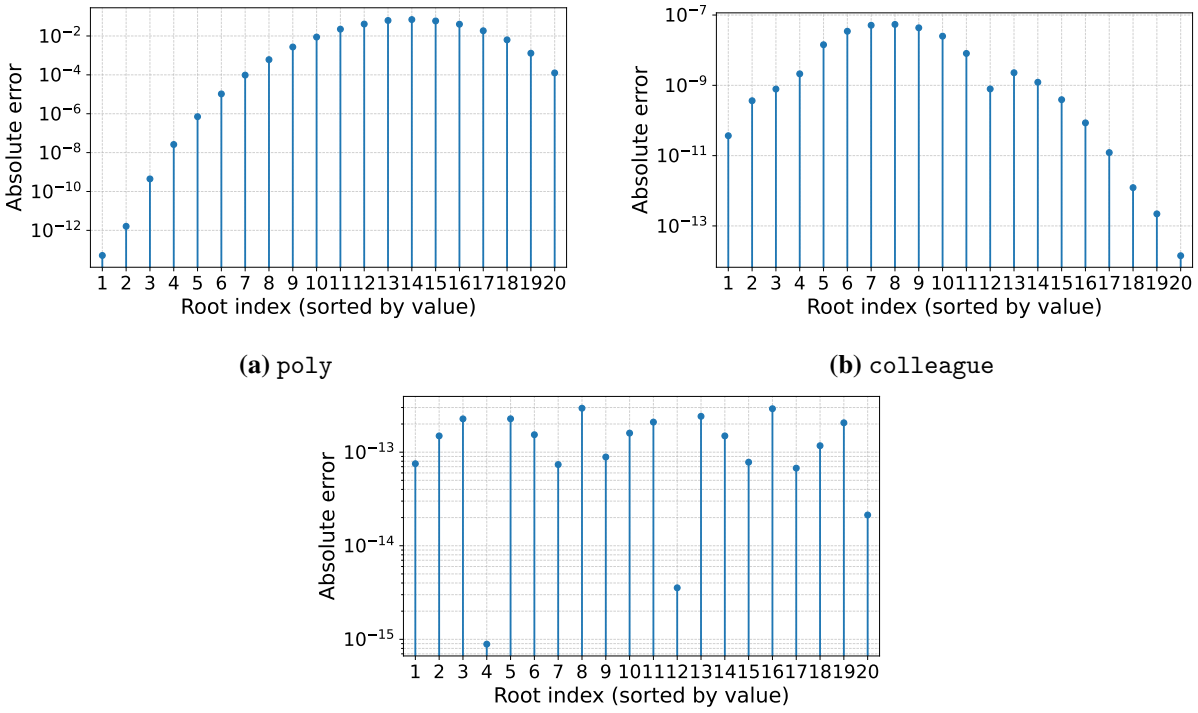
对 Wilkinson 多项式，

$$W(x) = \prod_{i=1}^{20}(x - i),$$

前面我们展示过用 `poly` 求根的结果，最大误差在 10^{-2} 数量级。虽然如此，由于 Wilkinson 多项式各个根之间的距离充分大，我们可以期待 `cheb-sturm` 可以正确分根，也想利用 Wilkinson 多项式算例来展现 `colleague` 在稳定性上的优势。计算的结果如表 2 和图 2 所示。

表 2: Wilkinson 多项式算例的求根结果与误差统计

真根	`poly`		`colleague`		`cheb-sturm`	
	根	绝对误差	根	绝对误差	根	绝对误差
1.0000	1.0000	5.1181×10^{-14}	1.0000	3.6835×10^{-11}	1.0000	7.5273×10^{-14}
2.0000	2.0000	1.6167×10^{-12}	2.0000	3.6465×10^{-10}	2.0000	1.4921×10^{-13}
3.0000	3.0000	4.4488×10^{-10}	3.0000	7.8334×10^{-10}	3.0000	2.2693×10^{-13}
4.0000	4.0000	2.6138×10^{-8}	4.0000	2.1252×10^{-9}	4.0000	8.8818×10^{-16}
5.0000	5.0000	7.0553×10^{-7}	5.0000	1.4287×10^{-8}	5.0000	2.2737×10^{-13}
6.0000	6.0000	1.0477×10^{-5}	6.0000	3.4575×10^{-8}	6.0000	1.5365×10^{-13}
7.0000	7.0001	9.6952×10^{-5}	7.0000	5.1166×10^{-8}	7.0000	7.3719×10^{-14}
8.0000	7.9994	6.0569×10^{-4}	8.0000	5.4075×10^{-8}	8.0000	2.9488×10^{-13}
9.0000	9.0027	2.7127×10^{-3}	9.0000	4.3347×10^{-8}	9.0000	8.8818×10^{-14}
10.0000	9.9912	8.8091×10^{-3}	10.0000	2.5064×10^{-8}	10.0000	1.5987×10^{-13}
11.0000	11.0220	2.2464×10^{-2}	11.0000	8.0835×10^{-9}	11.0000	2.0961×10^{-13}
12.0000	11.9590	4.1126×10^{-2}	12.0000	7.8916×10^{-10}	12.0000	3.5527×10^{-15}
13.0000	13.0630	6.2664×10^{-2}	13.0000	2.2855×10^{-9}	13.0000	2.4158×10^{-13}
14.0000	13.9300	6.9814×10^{-2}	14.0000	1.2236×10^{-9}	14.0000	1.4921×10^{-13}
15.0000	15.0590	5.9326×10^{-2}	15.0000	3.9065×10^{-10}	15.0000	7.8160×10^{-14}
16.0000	15.9600	4.0282×10^{-2}	16.0000	8.5599×10^{-11}	16.0000	2.9132×10^{-13}
17.0000	17.0190	1.8542×10^{-2}	17.0000	1.2214×10^{-11}	17.0000	6.7502×10^{-14}
18.0000	17.9940	6.3284×10^{-3}	18.0000	1.2292×10^{-12}	18.0000	1.1724×10^{-13}
19.0000	19.0010	1.2954×10^{-3}	19.0000	2.2027×10^{-13}	19.0000	2.0606×10^{-13}
20.0000	20.0000	1.2594×10^{-4}	20.0000	1.4211×10^{-14}	20.0000	2.1316×10^{-14}
统计指标						
最大绝对误差	—	6.9814×10^{-2}	—	5.4075×10^{-8}	—	2.9488×10^{-13}
平均绝对误差	—	1.6710×10^{-2}	—	1.1935×10^{-8}	—	1.4181×10^{-13}



(a) `poly` **(b)** `colleague`

(c) `cheb-sturm`

图 2: Wilkinson 多项式算例上求根的绝对误差

可以看到，由于 `cheb-sturm` 做到了正确分根并在单根区间上应用了二分法，所以精度可以做到很高（减小 `x_tol` 甚至可以将精度做到接近机器精度）；`colleague` 展示了 Chebyshev 多项式做多项式空间的基底确实可以提高计算的稳定性，得到较 `poly` 更精确的实根，即使在精确算术的意义下这两个算法是等价的。

对 `poly_quadruple_root` 算例 $(x-2)^4$，由定理 2.1 我们知道，当多项式出现重根时，计算重根的稳定性是很差的，因此我们不期待 `poly` 和 `colleague` 有很高的计算精度。虽然 `cheb_Sturm` 不计算根的重数，同时因为根的重数是偶数，不存在端点异号的性质，所以我们不能用一般二分法做根精化，而是通过 Sturm 计数二分（利用 Sturm 序列的变号数判断根处在左半区间还是右半区间）来做，这会损失一定的精度（因为 Sturm 计数二分基于 Chebyshev 多项式基底下的多项式表示，在计算上与原多项式存在误差）。计算的结果如表 3 所示。

表 3: `poly_quadruple_root` 算例的求根结果与误差统计

真根	`poly`		`colleague`		`cheb-sturm`	
	根	绝对误差	根	绝对误差	根	绝对误差
2.0000	1.9996	4.3824×10^{-4}	1.9999	1.1362×10^{-4}	1.9992	8.2129×10^{-4}
2.0000	2.0000	3.3541×10^{-8}	1.9999	1.1362×10^{-4}	—	—
2.0000	2.0000	3.3541×10^{-8}	2.0001	1.1362×10^{-4}	—	—
2.0000	2.0004	4.3830×10^{-4}	2.0001	1.1362×10^{-4}	—	—
统计指标						
最大绝对误差	—	4.3830×10^{-4}	—	1.1362×10^{-4}	—	8.2129×10^{-4}
平均绝对误差	—	2.1915×10^{-4}	—	1.1362×10^{-4}	—	8.2129×10^{-4}

可以看到，重根对我们测试的三种算法的影响比较显著，出现了预期的不稳定性，导致根出现较之 Wilkinson 多项式这样一重根算例更大的误差。在应用中，尽可能的给出除去重根的多项式是使得多项式求根更稳定的技巧。

对 `poly_mignotte_like` 算例，

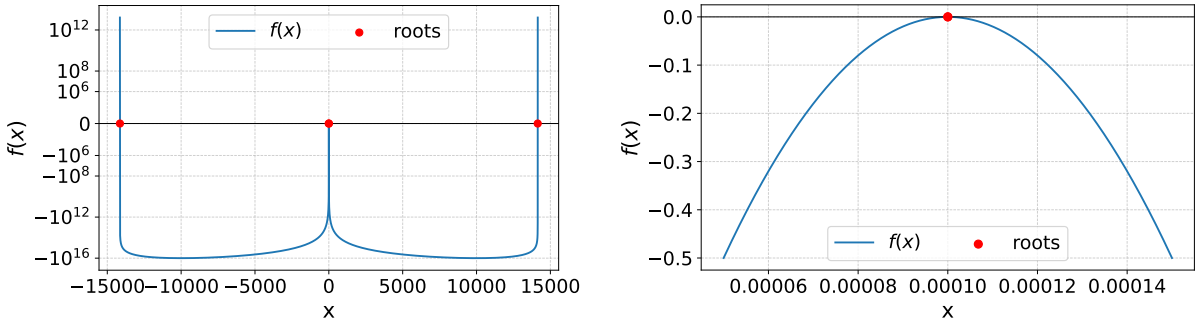
$$x^4 - 2(10^4 x - 1)^2,$$

其四个实根大致为

$$\begin{aligned} x_1 &\approx -14142.13572373094978, \\ x_2 &\approx \quad 0.00009999999929, \\ x_3 &\approx \quad 0.00010000000071, \\ x_4 &\approx \quad 14142.13552373094978. \end{aligned}$$

可以看到 x_2, x_3 的距离约为

$$|x_3 - x_2| \approx 1.42 \times 10^{-12}.$$



(a) 函数全局　　**(b)** 极近根附近

图 3: `poly_mignotte_like` 算例的函数图像

我们检验算法可以高精度地区分出这两个极近根，还是认为这两个极近根是一个二重实根。注意到四次方程有求根公式，我们可以用精确计算的结果作为真根的参考。计算结果如表 4 所示。

表 4: poly_mignotte_like 算例的求根结果与误差统计

真根	poly		colleague		cheb-sturm	
	根	绝对误差	根	绝对误差	根	绝对误差
-1.4142×10^{4}	-1.4142×10^{4}	1.8190×10^{-12}	-1.4142×10^{4}	1.8190×10^{-12}	-1.4142×10^{4}	1.8190×10^{-12}
1.0000×10^{-4}	1.0000×10^{-4}	9.0893×10^{-12}	1.0000×10^{-4}	4.1264×10^{-12}	-1.0926×10^{-3}	1.1926×10^{-3}
1.0000×10^{-4}	1.0000×10^{-4}	9.0893×10^{-12}	1.0000×10^{-4}	2.7121×10^{-12}	—	—
1.4142×10^{4}	1.4142×10^{4}	3.6380×10^{-12}	1.4142×10^{4}	1.8190×10^{-12}	1.4142×10^{4}	0
统计指标						
最大绝对误差	—	9.0893×10^{-12}	—	4.1264×10^{-12}	—	1.1926×10^{-3}
平均绝对误差	—	5.9089×10^{-12}	—	2.6191×10^{-12}	—	3.9752×10^{-4}

可以看到，poly 和 colleague 可以得到正确的根数量，然而绝对误差大于两个极近根的距离，可以认为这两个算法（在双精度下）没有分辨出极近根；而 cheb-sturm 同样认为两个极近根是一个二重根，所以返回根的数量有缺失。

从四个算例的数值结果来看，三种算法的优势与短板非常清晰。

首先在 poly_cyclotomic 算例 $x^5 - 1$ 中只有一个实根，其余为复根，三种方法都能找到该实根且精度很高，说明它们都具备正确剔除复根和保留实根的能力。不过 poly 和 colleague 的做法是先通过特征值法得到全部根，再用虚部阈值筛选实根，这一过程不可避免地依赖经验参数；而 cheb-sturm 只针对实根进行计数与隔离，不需要设置虚部阈值，因此当任务只关心实根时思路更直接。

在简单且分离的实根情形下，以 Wilkinson 多项式为例，cheb-sturm 的精度优势最明显，colleague 次之，而 poly 明显最差。这一差异与多项式表示的稳定性密切相关。当多项式的实根都是简单根且间距足够大时，实验结果支持 cheb-sturm $>$ colleague $\gg$ poly 的结论。

当多项式出现重根时，三种方法都会表现出预期的不稳定性，反映出重根问题本身对扰动极其敏感。也就是说，重根场景下不宜对任一方法的高精度抱有期待，若确实需要可靠处理重根，往往需要更高精度或额外的去重根手段配合。

极近根簇算例 poly_mignotte_like 展示了双精度下的分辨率瓶颈。该例中两根间距约 1.42×10^{-12}，poly 和 colleague 能给出数量上正确的根，但其绝对误差已大于或接近根间距，因此从数值意义上讲并不能稳定地区分这两个极近根；cheb-sturm 则表现得更脆弱，出现根缺失，这意味着计数在该尺度下被浮点误差带偏。因此，极近根簇是三种方法都难以在双精度下可靠解决的场景。要想真正分辨这类根，一般需要在系数计算、评估、计数或精化环节引入更高精度。

综合而言，poly 的优点是实现简单且一次可得到全部根（含复根），但在单项式基底下容易数值不稳；colleague 在保留特征值法便利性的同时显著提升稳定性，是比 poly 更稳健的默认选择，但仍需要虚部阈值筛实根；cheb-sturm 在“简单且分离的实根”问题上精度最好，并且天然只输出实根、不依赖阈值，但在极近根簇情形下可能出现漏根。

3 光滑函数的多实根求解方法

前面我们已经讨论过了求解多项式所有实根的两种代表性的方法。两种方法都利用了多项式良好的代数性质，并且能够处理相对良态的多项式求根问题。然而，对于一般的光滑函数，不再有这么好的性质可以利用并设计专门算法。

光滑函数的求多实根算法中，重要的一类思路与多项式求根的 Sturm 序列方法类似，先用全局策略把每个根所在子区间分离，再对每个子区间用收敛可靠的局部迭代将根逼近到所需精度。常见的根隔离做法是，在 $[a,b]$ 上递归二分生成子区间 I，用区间算术估计 $f(I)$；若 $0 \notin f(I)$ 则可直接判定 I 内无根，从而剪枝。进一步结合区间牛顿或 Krawczyk 收缩算子，可在某些子区间上给出存在唯一实根的保证并快速收缩得到互不重叠的隔离区间，随后再对每个隔离区间用任意稳定的局部迭代做精化 [13][15]。在更一般的（可解析）情形，还存在基于辐角原理（argument principle）的自适应细分算法以定位有界区域内全部零点 [5]。在仅能黑盒求值且区间信息不足时，也有人把 $|f|$ 或 f^2 视为目标做确定性全局优化（如 DIRECT 的分割采样思想）以生成候选根 [9]。

在本文中，我们更关注另一类思路。先给一般光滑函数 $f(x)$ 一个很好的多项式近似 $p(x)$，然后对近似的多项式求根作为 $f(x)$ 的近似根，而我们手上已经拥有了多项式插值或逼近，以及多项式求根的工具。这一数值算法的设计思想也在数值微分的整体估计方法、插值型求积方法等数值方法中广泛应用。这一类思路中，我们以 Boyd–Battles 方法为代表来着重介绍，这也是 Chebfun 中 `roots` 求根的实现算法 [23]。

3.1 Boyd–Battles 方法

Boyd–Battles 方法是将 f 在子区间上自适应逼近为 Chebyshev 或分段多项式，再把“找零点”转化为利用同事矩阵特征值的多项式求根。Boyd [4] 最先阐述了把 f 在目标区间做自适应 Chebyshev 插值和截断，然后用多项式求根器得到区间内全部实根（并丢弃区间外或复根）的策略；Battles [3] 将这一策略工程化，在 Chebfun 软件包上实现了 `roots` 求根算法。

下面简要介绍算法的步骤。

初始化　设目标区间为 $[a,b]$，将区间仿射变换到 $[-1,1]$，即

$$x = \frac{a+b}{2} + \frac{b-a}{2}t,$$

并令 $g(t) = f(x(t))$。

Chebyshev 插值　在第二类 Chebyshev 节点上采样，

$$t_j = \cos(\pi j/N), \quad j = 0,1,\dots,N,$$

得到 $g_j = g(t_j)$，再利用重心公式得到 Chebyshev 插值多项式，

$$p_N(t) = \frac{\sum_{j=0}^{N} \frac{w_j}{t-t_j} g_j}{\sum_{j=0}^{N} \frac{w_j}{t-t_j}}.$$

Chebyshev 展开　把 $p_N(t)$ 写成 Chebyshev 展开

$$p_N(t) = \sum_{k=0}^{N} a_k T_k(t),$$

其中系数 a_k 由余弦变换快速得到。

插值阶数截断　这一步是 Boyd–Battles 方法的关键步骤，实现了自适应地决定插值阶数，其中的超参数可能基于经验。在 Chebfun 中称为 `standardChop` [1]。具体的想法是，通过不断加密采样来生成更长的系数序列；当系数衰减到接近机器精度时，尾部往往出现由舍入误差造成的“平台”（在对数坐标下近似白噪声），此时再加密也难再提升有效精度，于是需要把级数“切掉”并保留到平台开始附近即可。

先做单调包络

$$\text{envelope}_j = \max_{j \le k \le n} |a_k|,$$

并将 envelope 归一化，令首项为 1。如果 envelope 快速下降，说明再增加插值点可以明显提高精度，不应该截断；反之，若出现 envelope 的平台，则认为再增加插值点对精度的帮助不大。具体地，对每一个候选位置 j，如果平台同时满足

1. 充分长。令

$$j_2 = \text{round}(1.25j + 5),$$

   把 $[j, j_2]$ 当作候选的平台长度；

2. 充分小和充分平。要求

$$\frac{\text{envelope}_{j_2}}{\text{envelope}_j} \ge 3\left(1 - \frac{\log(\text{envelope}_j)}{\log(\text{tol})}\right),$$

   当 $\text{envelope}_j = \text{tol}$ 时，必然通过；当 $\text{envelope}_j \approx \text{tol}^{2/3}$ 时，要求 $\text{envelope}_{j_2} \approx \text{envelope}_j$，即对平台的平坦程度有更高的要求；当 $\text{envelope}_j > \text{tol}^{2/3}$ 时，必然不通过。

则我们认为已经接近了精度的上限。若当前的 $j \le N$ 均不存在满足条件的平台，则取更大的 N 重复 Chebyshev 插值的步骤，一般 N 取 2 的幂次。最后，通过倾斜尺子（tilted ruler）来确定平台上的截断点。取截断阶数

$$m = \arg\min_k \left(\log_{10}(\text{envelope}_k) - \frac{k-1}{j_2 - 1}\left(\frac{1}{3}\log_{10}(\text{tol})\right)\right) - 1,$$

即通过偏置项权衡更小的截断项系数（对应更大的插值阶数）和更小的插值阶数（对应更大的截断项系数），兼顾较高的近似精度和较小的模型复杂度。得到截断后的多项式

$$p_m(t) = \sum_{k=0}^{m} a_k T_k(t),$$

称为 **Chebyshev 代理多项式**。注意代理多项式不是 m 次的插值多项式，而是 N 次插值多项式截断尾项后的多项式。

若区间上多项式插值的截断阶数大于 50，则会递归分裂区间使得每一个子区间上的插值多项式不超过 50 次。

多项式求根　由 Chebyshev 代理多项式直接构造同事矩阵，并求解同事矩阵的特征值得到候选根。丢弃非实根和区间外根并用原函数做必要的检查后输出。

值得注意的是，Boyd–Battles 方法要求给定搜索根区间，并返回区间内的所有实根。由于插值多项式只在目标区间上采样，所以我们无法保证多项式在目标区间外也能正确地近似函数，因此插值多项式在目标区间外的根是不可靠的。如果想在不给定目标区间地利用 Boyd–Battles 方法求解函数在全实轴上的根，需要配合其他方法来确定根界。

下面将以一个简单的算例演示 Boyd–Battles 方法的计算步骤。

令 $f(x) = \cos(5x) - x$，区间为 $[-1, 1]$。先设 $N = 16$，算出的 Chebyshev 系数 $\{a_k\}_{k=0}^{16}$，

$$|a_{14}| \approx 5.63 \times 10^{-6}, \quad |a_{16}| \approx 1.54 \times 10^{-7},$$

还不能判定为平台。加密插值阶数到 $N = 32$，得到 $\{a_k\}_{k=0}^{32}$，

$$|a_{20}| \approx 5.54 \times 10^{-11}, \quad |a_{22}| \approx 7.69 \times 10^{-13}, |a_{24}| \approx 9.18 \times 10^{-15}, \quad |a_{32}| \approx 5.55 \times 10^{-17},$$

注意到尾部已经达到机器精度量级。系数包络在 10^{-16} 附近形成平台，并截断到 $m = 24$。

对代理多项式求根，可以得到候选根

$$x \approx -0.7674934213, -0.3954766059, 0.2612880017,$$

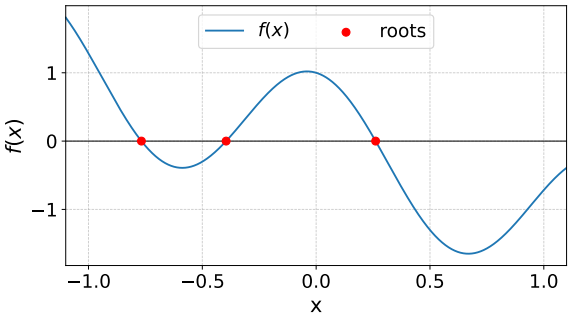
并用 Newton 迭代（导数由代理多项式提供）将根精细化，经过原函数的检查输出函数的根。

3.2 数值实验

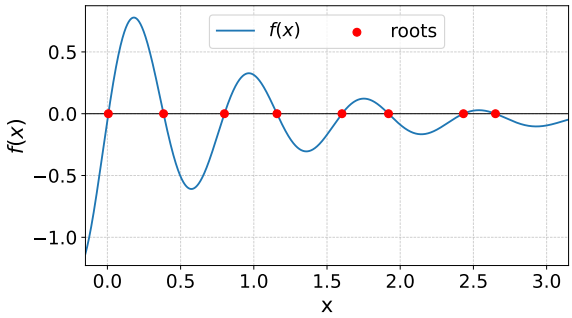
我们给出三个算例来测试 Boyd–Battles 方法，

1. $f(x) = \cos(5x) - x,\ x \in [-1, 1]$;
2. $f(x) = e^{-x}\sin(8x) - 0.05,\ x \in [0, 3]$，这是一个震荡且幅值指数衰减的函数;
3. $f(x) = (x - 0.4)^2(x + 0.8) - 10^{-6},\ x \in [-1, 1]$，这个算例在 0.4 附近有两个 “近重根”，即这两个根距离很近但不是重根。

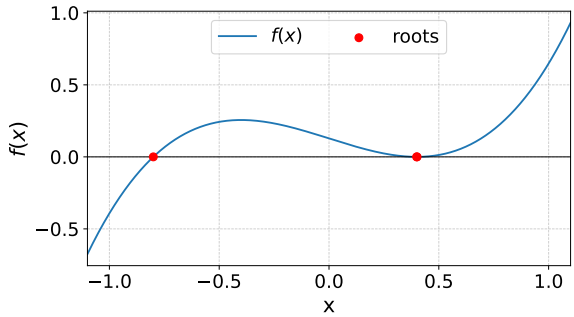
前两个算例我们均得不到显式的根，但从理论和图像（本质上是极密采样）上都比较容易判断根的数量，并且我们将通过 $f(r)$ 来评估根的质量。三个算例的图像如图 4 所示，计算结果如表 5, 6 以及 7 所示。



(a) $f(x) = \cos(5x) - x$



(b) $f(x) = e^{-x}\sin(8x) - 0.05$



(c) $f(x) = (x - 0.4)^2(x + 0.8) - 10^{-6}$

图 4: Boyd–Battles 方法三个算例的函数图像

表 5: $f(x) = \cos(5x) - x,\ x \in [-1, 1]$ 的求根结果与残差统计

根	绝对残差
-7.6749×10^{-1}	4.5519×10^{-15}
-3.9548×10^{-1}	0
2.6129×10^{-1}	1.9984×10^{-15}
统计指标	
最大绝对残差	4.5519×10^{-15}
平均绝对残差	2.1834×10^{-15}

表 6: $f(x) = e^{-x}\sin(8x) - 0.05,\ x \in [0, 3]$ 的求根结果与残差统计

根	绝对残差
6.2921×10^{-3}	7.4246×10^{-16}
3.8352×10^{-1}	1.3045×10^{-15}
7.9933×10^{-1}	5.5927×10^{-15}
1.1581	1.7347×10^{-15}
1.6021	9.7838×10^{-16}
1.9200	9.7838×10^{-16}
2.4319	5.4123×10^{-16}
2.6505	5.6205×10^{-16}
统计指标	
最大绝对残差	5.5927×10^{-15}
平均绝对残差	1.5543×10^{-15}

表 7: $f(x) = (x - 0.4)^2(x + 0.8) - 10^{-6},\ x \in [-1, 1]$ 的求根结果与残差统计

真根	根	绝对残差	绝对误差
-8.0000×10^{-1}	-8.0000×10^{-1}	9.0201×10^{-18}	0
3.9909×10^{-1}	3.9909×10^{-1}	6.7896×10^{-17}	2.6590×10^{-14}
4.0091×10^{-1}	4.0091×10^{-1}	6.7985×10^{-17}	2.6701×10^{-14}
统计指标			
最大绝对残差	6.7985×10^{-17}	最大绝对误差	2.6701×10^{-14}
平均绝对残差	4.8300×10^{-17}	平均绝对误差	1.7764×10^{-14}

三个算例的结果整体上表明，Boyd–Battles 方法在区间上能够稳定地给出根的位置，并且得到的根满足很小的残差；当根分离且函数行为温和时，残差可以轻松达到接近机器精度的水平，而在存在“近重根”的情况下仍能把根找出来，但根的位置精度会受到近重根病态性的限制。

需要强调的是，这三个算例相对都比较良态。由于 Boyd–Battles 方法基于多项式求根的同事矩阵特征值法，所以我们在求解带复杂情况的光滑函数（如重根、极近根）时，依然不可避免地出现数值不稳定的现象。额外地，如果 Chebyshev 插值的效果不好，会带来额外的求根困难。

4 总结与反思

整体来看，本文以求出多项式与光滑函数在实数域上的全部零点为目标，将多实根求解的难点归结为两点：一是根数量的可靠确定，二是根所在区间的有效定位，并据此组织了多项式与光滑函数两条主线的算法讨论。

在多项式部分，文章对比了两类思路：一类是先求全体零点再筛选实根，典型代表为单项

式基底下的伴随矩阵特征值法以及 Chebyshev 多项式基底下的同事矩阵特征值法；另一类是先根隔离再精化，以 Sturm 序列的变号计数为基础构造递归二分框架，并进一步在 Chebyshev 基底下得到 Chebyshev–Sturm 方法。四个算例的实验现象较为一致：当实根简单且间距足够大时，Chebyshev–Sturm 方法往往能在正确分根的基础上取得更高精度；在 Wilkinson 多项式等病态情形下，同事矩阵特征值法相对伴随矩阵特征值法更稳健，说明正交基底与缩放求系数对稳定性具有直接帮助；而在多重根算例与极近根簇算例中，三种方法均出现明显的精度退化，其中极近根簇两根间距约为 1.42×10^{-12}，此时特征值法的误差量级已接近甚至超过根间距，Chebyshev–Sturm 方法也可能因计数误差而漏根，这提示双精度下存在难以回避的分辨率瓶颈。

在光滑函数部分，文章选择 Boyd–Battles 方法作为核心路线，将区间求根转化为自适应 Chebyshev 插值与多项式求根：先用 `standardChop` 判定系数包络在机器精度附近形成平台并据此截断得到代理多项式，必要时递归分裂区间以控制多项式次数，然后通过同事矩阵特征值法得到候选根并进行 Newton 精化与原函数残差检验。三组算例显示，在根分离且函数行为温和时，该方法可以稳定返回残差在 10^{-15} 量级的根；而在近重根情形下虽然仍能定位到两根，但根位置的绝对误差会上升到 10^{-14} 量级，反映出病态性对精度的限制。与此同时，Boyd–Battles 方法要求预先给定搜索区间，且区间外的代理多项式根并不可靠，这意味着在全实轴场景仍需结合根界估计或区间隔离策略。

综合本文的算法梳理与数值实验，可以得到以下几点体会，

1. 多实根求解首先是一个全局信息获取问题。仅靠单点迭代难以同时解决根数与根区间定位，实际流程更应强调根计数、根隔离与精化之间的配合；
2. 基底与表示的选择会显著改变数值稳定性。Chebyshev 等正交基底配合缩放与稳定的系数计算，往往能改善特征值法与计数法的表现；
3. 重根与极近根簇是决定性难点。双精度下的分辨率瓶颈会让不同算法同时失效，因此需要更高精度、去重根技术或更强的结果认证机制；
4. 对光滑函数而言，多项式逼近路线具有很强的工程可实现性，但插值质量与区间设定是关键前提。

本文仍存在一些不足，有待后续改进和完善，主要包括，

1. 数值实验集中在一维问题，算例数量与复杂度仍偏有限；对更高次数、更高维度或真实工程模型中的多实根问题，需要更系统的验证；
2. 总次数同伦续接算法部分以理论介绍为主，缺少路径追踪与复杂度评估的数值实验，因此对其在实际多项式方程组中的可行性仍难下结论；
3. 对最困难的重根与极近根簇，仅通过双精度实验难以形成可验证的可靠求解流程，未来可引入高精度计算、去重根与区间认证等工具进行补强。

本文通过多项式求根与光滑函数求根两条线索，较为直观地展示了多实根求解中基底选择、区间策略与问题病态性之间的相互作用，对后续更稳健的算法设计与工程实践中的方法选型具有启发意义。

希望老师批评指正！

参考文献

[1] Jared L Aurentz and Lloyd N Trefethen. “Chopping a Chebyshev series”. In: *ACM Transactions on Mathematical Software (TOMS)* 43.4 (2017), pp. 1–21.

[2] Stephen Barnett. “A companion matrix analogue for orthogonal polynomials”. In: *Linear Algebra and its Applications* 12.3 (1975), pp. 197–202.

[3] Zachary Battles and Lloyd N Trefethen. “An extension of MATLAB to continuous functions and operators”. In: *SIAM Journal on Scientific Computing* 25.5 (2004), pp. 1743–1770.

[4] John P Boyd. “Computing zeros on a real interval through Chebyshev expansion and polynomial rootfinding”. In: *SIAM Journal on Numerical Analysis* 40.5 (2002), pp. 1666–1682.

[5] Michael Dellnitz, Oliver Schütze, and Qinghua Zheng. “Locating all the zeros of an analytic function in one complex variable”. In: *Journal of Computational and Applied mathematics* 138.2 (2002), pp. 325–333.

[6] Yuli Eidelman, Luca Gemignani, and Israel Gohberg. “Efficient eigenvalue computation for quasiseparable Hermitian matrices under low rank perturbations”. In: *Numerical Algorithms* 47.3 (2008), pp. 253–273.

[7] H. Scott Fogler. *Elements of Chemical Reaction Engineering*. 3rd ed. Upper Saddle River, NJ: Prentice Hall PTR, 1999.

[8] IJ Good. “The colleague matrix, a Chebyshev analogue of the companion matrix”. In: *The Quarterly Journal of Mathematics* 12.1 (1961), pp. 61–68.

[9] Donald R Jones and Joaquim RRA Martins. “The DIRECT algorithm: 25 years Later”. In: *Journal of global optimization* 79.3 (2021), pp. 521–566.

[10] Eglantina Kalluci and Fatmir Hoxha. “The Solution of Ill-Conditional Polynomial Equations”. In: *Buletini i Shkencave të Natyrës* Special (2015). Online; special issue (SPNA 2015 conference proceedings), Faculty of Natural Sciences, University of Tirana, pp. 21–29. ISSN: 2305-882X. URL: https://jns.edu.al/wp-content/uploads/2023/08/2_E_Kalluci_903387fbfb.pdf.

[11] Akshit Lunia et al. *Modeling, Motion Planning, and Control of Manipulators and Mobile Robots*. Clemson University, 2021.

[12] John Maroulas and Stephen Barnett. “Polynomials with respect to a general basis. I. Theory”. In: *Journal of Mathematical Analysis and Applications* 72.1 (1979), pp. 177–194.

[13] Ramon E Moore. *Interval analysis*. Prentice-Hall, 1966.

[14] Yuji Nakatsukasa and Vanni Noferini. “On the stability of computing polynomial roots via confederate linearizations”. In: *Mathematics of Computation* 85.301 (2016), pp. 2391–2425.

[15] Arnold Neumaier. *Interval methods for systems of equations*. 37. Cambridge university press, 1990.

[16] Vanni Noferini and Javier Pérez. “Chebyshev rootfinding via computing eigenvalues of colleague matrices: when is it stable?” In: *Mathematics of Computation* 86.306 (2017), pp. 1741–1767.

[17] Vanni Noferini, Leonardo Robol, and Raf Vandebril. “Structured backward errors in linearizations”. In: *Electronic Transactions on Numerical Analysis* 54 (2021), pp. 420–442. DOI: 10.1553/etna_vol54s420.

[18] Katsuhiko Ogata. *Modern control engineering*. Prentice hall, 2010.

[19] Kirill Serkh and Vladimir Rokhlin. *A Provably Componentwise Backward Stable $O(n^2)$ QR Algorithm for the Diagonalization of Colleague Matrices*. Feb. 2021. DOI: 10.48550/arXiv.2102.12186. arXiv: 2102.12186 [math.NA]. URL: https://arxiv.org/abs/2102.12186.

[20] Wilhelm Specht. “Die lage der nullstellen eines polynoms. iv”. In: *Mathematische Nachrichten* 21.3-5 (1960), pp. 201–222.

[21] Charles François Sturm. “Mémoire sur la résolution des équations numériques”. In: *Collected Works of Charles François Sturm*. Birkhäuser Basel, 2009, pp. 345–390.

[22] Jean-Pierre Tignol. *Galois' theory of algebraic equations*. World Scientific Publishing Company, 2015.

[23] Lloyd N Trefethen. “Computing numerically with functions instead of numbers”. In: *Communications of the ACM* 58.10 (2015), pp. 91–97.

附录

A 同伦算法

我们的动机是针对难以求解的多项式系统 $F(x) = 0$，我们考虑容易求解的多项式系统 $G(x) = 0$，我们通过将两者的解通过连续变形联系起来，通过追踪解的连续路径将 G 的解变形到 F 的解。

定义 A.1 (多项式方程组)**.** 设

$$F(x) = \begin{pmatrix} f_1(x) \\ \vdots \\ f_n(x) \end{pmatrix}, \qquad x = (x_1, \ldots, x_n) \in \mathbb{C}^n,$$

其中每个 f_i 是 $\mathbb{C}[x_1, \ldots, x_n]$ 中的多项式。称 $F(x) = 0$ 为一个 n 元 n 方程的多项式系统。

定义 A.2 (起始系统与同伦)**.** 设 $G(x) = 0$ 是一个起始系统，其所有解在 $\mathbb{C}^n$ 中已知。定义一个同伦

$$H(x, t) = (1 - t)\, G(x) + t\, F(x), \qquad t \in [0, 1]. \tag{7}$$

称满足 $H(x(t), t) = 0$ 的连续曲线 $t \mapsto x(t)$ 为从 G 到 F 的一条同伦解路径。

注. 实际上，更常用的形式是多项式同伦

$$H(x, t) = \gamma(1 - t)\, G(x) + t\, F(x), \tag{8}$$

其中 $\gamma = \mathrm{e}^{i\theta} \in \mathbb{C}$, $\theta \in [0,\, 2\pi]$. 这样的选择可以避免绝大多数奇异情况。

A.1 隐函数定理与局部解路径

同伦连续法的基本理论支撑来自隐函数定理：在雅可比矩阵非奇异处，解路径局部存在且唯一。

定理 A.1 (局部解路径存在性与唯一性)**.** 设

$$H : \mathbb{C}^n \times \mathbb{C} \longrightarrow \mathbb{C}^n$$

为一个复解析映射（或实 C^1 映射），存在点 (x_0, t_0) 满足

$$H(x_0, t_0) = 0, \qquad \det\big(J_x H(x_0, t_0)\big) \neq 0,$$

其中 $J_x H$ 表示 H 对 x 的雅可比矩阵。则存在 $\varepsilon > 0$ 以及唯一的解析函数

$$x : (t_0 - \varepsilon, t_0 + \varepsilon) \to \mathbb{C}^n$$

满足

$$x(t_0) = x_0, \qquad H(x(t), t) = 0 \quad \text{对所有} |t - t_0| < \varepsilon \text{ 成立。}$$

证明. 记 $H = (h_1, \ldots, h_n)$。由于 H 关于 (x, t) 连续可微，且在 (x_0, t_0) 处有

$$\det J_x H(x_0, t_0) \neq 0,$$

故矩阵 $J_xH(x_0,t_0)$ 可逆。由隐函数定理可知：存在邻域 $U\subset\mathbb{C}^n$（包含 x_0）与 $V\subset\mathbb{C}$（包含 t_0），以及唯一的解析函数

$$\phi : V \longrightarrow U, \qquad \phi(t_0)=x_0,$$

使得对所有 $t\in V$ 都有

$$H(\phi(t),t)=0.$$

把 $x(t)=\phi(t)$ 记为沿 t 的局部解路径，它满足 $x(t_0)=x_0$ 且在 V 上唯一确定。进一步，对 $H(x(t),t)=0$ 两边关于 t 求导得

$$J_xH(x(t),t)\frac{dx}{dt}+J_tH(x(t),t)=0,$$

由 $J_xH(x(t),t)$ 可逆性可得

$$\frac{dx}{dt}=-\big[J_xH(x(t),t)\big]^{-1}J_tH(x(t),t),$$

说明 $x(t)$ 关于 t 的导数连续存在。因此在 (x_0,t_0) 附近存在且唯一的解析路径 $t\mapsto x(t)$ 使 $H(x(t),t)=0$ 成立。 □

注. 上述定理意味着：只要路径上的点 $(x(t),t)$ 保持 正则（即 $\det J_xH\neq 0$），就可以通过求解微分方程

$$J_xH(x,t)\,\frac{dx}{dt}+J_tH(x,t)=0$$

得到解路径的切向量 $\frac{dx}{dt}$，从而在数值上用预测–校正方法追踪路径。

A.2 随机参数下的正则性

在实际的多项式同伦算法中，常使用带随机复参数的同伦

$$H_\gamma(x,t)=\gamma(1-t)\,G(x)+t\,F(x), \qquad \gamma\in\mathbb{C}\setminus\{0\},$$

其中 $F,G:\mathbb{C}^n\to\mathbb{C}^n$ 为给定多项式映射。下面的命题给出一个典型的“泛性正则”结论：除了少数坏的参数 γ 之外，任意解对 (x,t) 都是对 x 的正则点。

命题 A.2 (随机参数下的正则性)**.** *设 $F,G:\mathbb{C}^n\to\mathbb{C}^n$ 为多项式映射，定义带参数的同伦*

$$H_\gamma(x,t)=\gamma(1-t)\,G(x)+t\,F(x), \qquad (x,t,\gamma)\in\mathbb{C}^n\times\mathbb{C}\times\mathbb{C}.$$

记

$$J_xH_\gamma(x,t)=\frac{\partial H_\gamma}{\partial x}(x,t)$$

为 H_γ 对 x 的雅可比矩阵。则存在一个非零多项式 $p(\gamma)\in\mathbb{C}[\gamma]$，使得对所有满足 $p(\gamma)\neq 0$ 的 γ，都有如下性质：如果 $(x,t)\in\mathbb{C}^n\times\mathbb{C}$ 满足

$$H_\gamma(x,t)=0,$$

则

$$\det\big(J_xH_\gamma(x,t)\big)\neq 0.$$

换句话说，对于除去有限个（或至多代数意义上的“少数”）参数 γ 之外，同伦方程 $H_\gamma(x,t)=0$ 的所有解点在 x 方向上都是正则点。

证明. 考虑映射

$$\Phi : \mathbb{C}^n \times \mathbb{C} \times \mathbb{C} \longrightarrow \mathbb{C}^{n+1}, \qquad \Phi(x,t,\gamma) = \big(H_\gamma(x,t),\, \det(J_x H_\gamma(x,t))\big).$$

这里 $H_\gamma(x,t)$ 以及 $\det(J_x H_\gamma(x,t))$ 都是关于 (x,t,γ) 的多项式，因此 Φ 是一个多项式映射。

记

$$W = \Phi^{-1}(0) = \big\{(x,t,\gamma) \in \mathbb{C}^{n+2} \mid H_\gamma(x,t) = 0,\ \det(J_x H_\gamma(x,t)) = 0\big\}.$$

这是 $\mathbb{C}^{n+2}$ 中的一个代数集（即同时满足若干多项式方程的集合）。

从几何上看，W 正是“同伦方程 $H_\gamma(x,t)=0$ 的奇异解点集合”，因为在这些点上既满足方程本身，又满足对 x 的雅可比矩阵奇异。

考虑投影

$$\pi : \mathbb{C}^n \times \mathbb{C} \times \mathbb{C} \longrightarrow \mathbb{C}, \qquad \pi(x,t,\gamma) = \gamma.$$

记

$$S = \pi(W) \subset \mathbb{C},$$

即

$$S = \big\{\gamma \in \mathbb{C} \mid \exists (x,t) \in \mathbb{C}^n \times \mathbb{C} \text{ 使得} H_\gamma(x,t) = 0,\ \det(J_x H_\gamma(x,t)) = 0\big\}.$$

换句话说，S 是所有“坏参数”的集合：在这些参数下，同伦方程存在奇异解点。

根据代数几何中的消去理论（Elimination theory）或 Chevalley 定理，代数集在多项式映射下的像是构造集（constructible set），特别地，在本例中，S 是 $\mathbb{C}$ 中的一个代数构造子集。在一维情形下，这意味着存在一个多项式 $p(\gamma) \in \mathbb{C}[\gamma]$，使得 S 包含于 $p(\gamma)=0$ 的零点集合，并且 $p(\gamma)$ 可以选取为非零多项式。直观理解是：S 是由某个非零多项式的零点“描述出来”的。

于是我们得到：

$$\gamma \notin S \quad \Longrightarrow \quad \text{对所有满足} H_\gamma(x,t)=0 \text{ 的} (x,t),\ \text{都有 } \det(J_x H_\gamma(x,t)) \neq 0.$$

也就是说，对于任意 γ 不在 S 的参数，同伦方程的所有解点都是正则的。

由于 S 由非零多项式的零点给出，它在通常拓扑下是一个“稀疏”的集合：要么是有限点集，要么是某个代数曲线的零点集，但无论如何是真的代数子集。因此，对几乎所有参数 γ（例如从 $\mathbb{C}$ 中随机选取的参数），同伦方程 $H_\gamma(x,t)=0$ 的解在 x 方向上都是正则点。将这一事实写成命题开头的形式，即得所需结论。 □

A.3 Bézout 上界与路径条数

我们先给出经典的 Bézout 上界：多元多项式系统孤立解（按重数计）数量的一个乘积上界。为了叙述方便，我们在复数域 $\mathbb{C}$ 上工作。

定理 A.3 (Bézout 上界)**.** 设

$$F(x) = \begin{pmatrix} f_1(x) \\ \vdots \\ f_n(x) \end{pmatrix}, \qquad x = (x_1, \dots, x_n) \in \mathbb{C}^n,$$

其中 $f_i \in \mathbb{C}[x_1, \dots, x_n]$ 为非常数多项式，其总次数为

$$\deg(f_i) = d_i \geq 1, \qquad i = 1, \dots, n.$$

假设 $F(x) = 0$ 的所有解在 $\mathbb{C}^n$ 中都是孤立点（即不存在正维解集分支）。记这些孤立解为 $\{\xi^{(1)}, \ldots, \xi^{(N)}\}$，每个解带有代数重数 $m(\xi^{(k)})$。则有上界

$$\sum_{k=1}^{N} m(\xi^{(k)}) \;\le\; d_1 d_2 \cdots d_n. \tag{9}$$

在 $f_1, \ldots, f_n$ 的系数处于一般位置（*generic*）的情形下，上式取等号。

证明思路. 这里给出一个常见的代数几何证明框架。

第一步：射影化。 将 $\mathbb{C}^n$ 嵌入到射影空间 $\mathbb{P}^n(\mathbb{C})$ 中，通过引入齐次坐标 $[X_0 : \cdots : X_n]$，把每个 f_i 替换为一个同次数的齐次多项式

$$F_i(X_0, \ldots, X_n)$$

满足

$$f_i(x_1, \ldots, x_n) = F_i(1, x_1, \ldots, x_n).$$

这样得到射影多项式系统

$$F_1 = \cdots = F_n = 0$$

在 $\mathbb{P}^n$ 中定义的射影代数集。

第二步：射影曲线的交数。 在射影空间中，多项式 F_i 的次数仍为 d_i。若系数处于一般位置，则 n 个射影超曲面

$$V(F_i) = \{F_i = 0\} \subset \mathbb{P}^n$$

横向相交，其交集

$$V(F_1) \cap \cdots \cap V(F_n)$$

由有限个射影点组成，且没有正维分支。经典的射影交叉理论（Bézout 定理）给出：这些交点的射影交数之和恰为

$$d_1 d_2 \cdots d_n.$$

特别地，在一般位置下，交点个数（按重数计）等于这个乘积。

第三步：回到仿射空间。 每个射影解点要么坐落在仿射片 $\{X_0 \neq 0\}$ 中，对应 $\mathbb{C}^n$ 中的一个有限解 x，要么坐落在“无穷远超平面” $\{X_0 = 0\}$ 上，对应所谓无穷远方向的解。将这些射影解点限制到仿射片，可以得到 $\mathbb{C}^n$ 中的全部有限孤立解，其重数和不超过总射影交数

$$d_1 \cdots d_n.$$

这就得到不等式 (9)。

第四步：一般位置情形。 在系数“处于一般位置”时，系统的所有解均为射影中的横截交点，不落在无穷远超平面，且各点的交数（重数）为 1，因此有限解的总个数恰等于 $d_1 \cdots d_n$。这说明在 generic 情形下上界可以取到。 □

注. 严格的 Bézout 定理是在射影空间中表述的：两个射影代数簇的交数等于次数乘积，前提是没有公共分支且交点横截。上面只是把这一思想简化到 “n 个多项式在 $\mathbb{C}^n$ 中的有限孤立解数” 的情形。在数值代数几何中，我们通常直接使用结论 (9) 作为同伦路径条数的粗略上界。

A.4 总次数同伦与路径条数

Bézout 上界与同伦算法的复杂性直接相关，因为每条可能的解对应一条需要追踪的同伦路径。下面给出一个标准的总次数同伦构造及其路径条数。

定理 A.4 (总次数起始系统)**.** *在定理 A.3 的设定下，令*

$$g_i(x_1,\dots,x_n) = x_i^{d_i} - 1, \qquad i = 1,\dots,n.$$

则起始系统

$$G(x) = 0 \quad \Longleftrightarrow \quad g_1(x) = \cdots = g_n(x) = 0$$

在 $\mathbb{C}^n$ 中的解集为

$$x_i^{d_i} = 1, \quad i = 1,\dots,n,$$

因此共有

$$N_0 = d_1 d_2 \cdots d_n$$

个解，并且每个解都是非奇异的（雅可比矩阵非奇异）。

证明. 对每个 i，方程 $x_i^{d_i} - 1 = 0$ 在 $\mathbb{C}$ 中有 d_i 个不同的根，它们是 d_i 个单位根。由于各方程只涉及各自的变量 x_i，整个系统的解是各坐标根的 Descartes 积：

$$x_1 \in \{d_1 \text{ 个根}\},\dots,x_n \in \{d_n \text{ 个根}\},$$

所以解的总数为 $d_1 \cdots d_n$。

其次，$G(x)$ 的 Jacobi 矩阵为对角矩阵

$$J_x G(x) = \operatorname{diag}\left(d_1 x_1^{d_1-1},\dots,d_n x_n^{d_n-1}\right).$$

在任意解点上都有 $x_i \neq 0$，且 $d_i \geq 1$，故每个对角元都非零，从而 $\det(J_x G(x)) \neq 0$，每个解都是非奇异解点。 □

利用上述起始系统，我们可以自然构造出总次数同伦：

推论 (总次数同伦的路径条数)**.** *在定理 A.3 和 A.4 的设定下，考虑带随机复参数的同伦*

$$H_\gamma(x,t) = \gamma(1-t)\,G(x) + t\,F(x), \qquad \gamma \in \mathbb{C}\setminus\{0\}, \quad t \in [0,1].$$

则起始时刻 $t = 0$，同伦方程 $H_\gamma(x,0) = 0$ 的解恰为 $G(x) = 0$ 的 $N_0 = d_1 \cdots d_n$ 个非奇异解。对泛性选择的 γ，每个起始解都会沿着一条唯一的正则解路径

$$t \mapsto x^{(k)}(t), \qquad k = 1,\dots,N_0,$$

在 $t \in [0,1)$ 上连续存在。当 $t \to 1$ 时，每条路径要么收敛到 $F(x) = 0$ 的一个有限解，要么在射影意义下 “跑向无穷远”。

因此，总次数同伦算法在数值上需要追踪的路径条数为

$$N_{paths} = d_1 d_2 \cdots d_n,$$

而最终得到的有限孤立解个数不超过 *Bézout* 上界 (9)。

注. 从数值计算的角度看：

- 所有 N_0 条路径可以完全并行追踪；
- 真正对应于 $F(x)=0$ 有限解的路径条数通常远少于 N_0；
- 跑向无穷远或数值发散的路径可以被检测并丢弃；
- 更精细的上界（如基于混合体积的 BKK 界）可以显著减少路径数，形成所谓多面体同伦（polyhedral homotopy）。

总次数同伦因此常被视为“基准版本”的同伦算法：理论简单、实现容易，但在变量数多、次数高时往往不够高效。

下面我们给出同伦算法的步骤。

同伦算法（Homotopy Continuation）

1. 构造起始系统

$$G(x)=\begin{cases} x_1^{d_1}-1=0 \\ x_2^{d_2}-1=0 \\ \vdots \\ x_n^{d_n}-1=0 \end{cases}$$

   并求出所有解。

2. 定义同伦

$$H(x,t)=(1-t)G(x)+tF(x),\quad t\in[0,1].$$

3. 对每个起始解 $x_j^{(0)}$：
   - 预测：计算

$$\frac{dx}{dt}=-\left(\frac{\partial H}{\partial x}\right)^{-1}\frac{\partial H}{\partial t},$$

     预测 $x_j(t+\Delta t)$。
   - 校正：迭代使 $H(x_j(t+\Delta t),t+\Delta t)=0$。
   - 调整步长 Δt 并重复预测-校正，直到 $t=1$。
4. 当 $t=1$ 时，每条路径终点 $x_j(1)$ 为 $F(x)=0$ 的解。
5. 收集所有路径终点，得到系统的所有解。

B 同事矩阵特征值与多项式根关系的另一种证明

虽然定理 2.2 证明了

$$\det(xI - C_T) = \frac{1}{2^{n-1}}p(x), \tag{10}$$

但是证明过程更像是，已知同事矩阵 C_T，而通过纯粹计算的方式证明了 (10)，对构造同事矩阵的启发性并不大。但下面给出的另一个证明，使我们能够看到同事矩阵的构造过程和特殊结构。

定理 B.1. *对 Chebyshev 多项式基底下的首一多项式 $p(x)$，存在一个矩阵 C_T，使得 λ 是 $p(x)$ 的根当且仅当 λ 是同事矩阵 C_T 的特征值。*

证明. 第一类 Chebyshev 多项式 $T_k(x)$ 满足递推式

$$T_0(x) = 1, \quad T_1(x) = x, \quad T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x), \tag{11}$$

将 (11) 改写成

$$xT_k(x) = \frac{T_{k+1}(x) + T_{k-1}(x)}{2},$$

并且 $xT_0(x) = T_1(x)$。令

$$t(x) = \begin{bmatrix} T_0(x) \\ T_1(x) \\ \vdots \\ T_{n-1}(x) \end{bmatrix} \in \mathbb{R}^n, \quad e_n = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix},$$

那么存在一个固定的三对角矩阵 J，使得

$$xt(x) = Jt(x) + \frac{1}{2}T_n(x)e_n, \tag{12}$$

其中

$$J = \frac{1}{2}\begin{bmatrix} 0 & 2 & & & \\ 1 & 0 & 1 & & \\ & 1 & 0 & \ddots & \\ & & \ddots & \ddots & 1 \\ & & & 1 & 0 \end{bmatrix}.$$

若 λ 是 $p(x)$ 的根，则

$$T_n(\lambda) = -(c_{n-1}T_{n-1}(\lambda) + \cdots + c_0T_0(\lambda)) = -[c_0, \ldots, c_{n-1}]t(\lambda),$$

代入 (12)，得

$$\lambda t(\lambda) = Jt(\lambda) - \frac{1}{2}e_n[c_0, \ldots, c_{n-1}]t(\lambda),$$

令 $C_T \coloneqq J - \frac{1}{2}e_n[c_0, \ldots, c_{n-1}]$，就有

$$C_T t(\lambda) = \lambda t(\lambda),$$

即 λ 是 C_T 的特征值。

同样可以证明，若 λ 是同事矩阵 C_T 的特征值，

$$C_T v = \lambda v, \quad v \neq 0, \tag{13}$$

展开前 $n-1$ 行，得 $v_1 = \lambda v_0$，以及

$$v_{k+1} = 2\lambda v_k - v_{k-1}, \quad k = 1, 2, \ldots, n-2,$$

与 Chebyshev 多项式的递推完全相同，归纳可得

$$v_k = v_0 T_k(\lambda), \quad k = 0, 1, \ldots, n-1.$$

若 $v_0 = 0$，那么 $v_k = 0$，与 $v \neq 0$ 矛盾，所以 $v_0 \neq 0$。

再看 (13) 的最后一行，

$$\frac{1}{2} v_{n-2} - \frac{1}{2} \sum_{k=0}^{n-1} c_k v_k = \lambda v_{n-1},$$

由 $v_k = v_0 T_k(\lambda)$，得

$$\frac{1}{2} v_0 T_{n-2}(\lambda) - \frac{1}{2} v_0 \sum_{k=0}^{n-1} c_k T_k(\lambda) = \lambda v_0 T_{n-1}(\lambda),$$

又 $v_0 \neq 0$，所以

$$T_{n-2}(\lambda) - \sum_{k=0}^{n-1} c_k T_k(\lambda) = 2\lambda T_{n-1}(\lambda).$$

再利用 Chebyshev 多项式的递推式 $T_n(\lambda) = 2\lambda T_{n-1}(\lambda) - T_{n-2}(\lambda)$，得

$$T_{n-2}(\lambda) - \sum_{k=0}^{n-1} c_k T_k(\lambda) = T_n(\lambda) + T_{n-2}(\lambda),$$

即

$$T_n(\lambda) + \sum_{k=0}^{n-1} c_k T_k(\lambda) = p(\lambda) = 0,$$

所以 C_T 的特征值 λ 是 $p(x)$ 的根。 □

注. 在这里的证明中，从头构造了同事矩阵 $C_T := J - \frac{1}{2} e_n [c_0, \ldots, c_{n-1}]$。事实上，这个构造与定义 2.1 的形式有区别，前者在最后一行做了秩 1 修正，而后者在第一行。可以证明，两种构造的同事矩阵具有完全相同的特征值（计重数）。