系统辨识 -- MATLAB辨识

前置知识

拉普拉斯变换

> 把时域中的微分方程，变成复频域的代数方程（拉普拉斯域形式），不用解微分就能分析系统

定义

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

- $s = \sigma + j\omega$ ，“复频率”“拉普拉斯算子” $j = \sqrt{-1}$ 是虚数单位
  - 实部σ：决定信号的衰减/增幅特性，单位 s^{-1}
  - 虚部 ω：决定信号的“振荡频率”，单位rad/s。

性质

时域运算	频域对应
微分 $\frac{d}{dt}$	乘以（s）
积分 $\int$	除以（s）

即 $\mathcal{L}\{\dot{x}(t)\} = sX(s) - x(0)$ 当初值为0时 $\mathcal{L}\{\dot{x}(t)\} = sX(s)$

在时域中稳态即 $\frac{df(t)}{dt} = 0$ 在拉普拉斯域中 $\mathcal{L}\{\dot{f}(t)\} = sF(s)$

若稳态中 $\dot{f}(t) = 0$ 那么对应拉普拉斯运算中所有一阶微分项都消失，相当于s=0，系统仅剩常数项

Laplace基本对

$$\mathcal{L}\{1\}=\frac{1}{s},\quad \mathcal{L}\{e^{-at}\}=\frac{1}{s+a},$$
$$\mathcal{L}\{\sin\omega t\}=\frac{\omega}{s^2+\omega^2},\quad \mathcal{L}\{\cos\omega t\}=\frac{s}{s^2+\omega^2}$$

运算实例

对于一阶惯性系统 $G(s)=\dfrac{1}{Ts+1}$ 可以得到这样的结论：频率越高，幅值越小、相位越滞后。

分析：代入唯一变量 $s=\sigma+j\omega$ 得 $G(s)=\dfrac{1}{T(\sigma+j\omega)+1}=\dfrac{1}{(T\sigma+1)+jT\omega}$

- 复数的相位是其在复平面上与实轴的夹角

对于复数 $G(s)=\dfrac{1}{m+jn}$ 其中$m=T\sigma+1, n=T\omega$

相位角 ϕ 满足 $\phi=$ 相位$(1)-$ 相位$(m+jn)=0-\arctan(\dfrac{a}{b})=-\arctan(\dfrac{T\omega}{T\sigma+1})$

> 别忘了 j 是虚数单位， σ 表示幅值 ω 表示频率

辨识

目标：得到被控对象的参数化模型

简述系统辨识方法

> 建立数学模型的两种方法
>
> - 解析法：通过内部运动规律等，利用有关定理用数学方法建立数学模型
> - 试验法：对输入输出数据进行处理，获取其特性从而得到数学模型
> - 试验法即系统辨识

系统辨识方法建立起的被控对象的数学模型可分为**参数模型**（如传递函数、状态空间方程等）和**非参数模型**（如阶跃响应、脉冲响应），非参数模型可以通过变换转化为参数模型。

系统辨识的方法可以分为**经典和现代**两大类。现代辨识法有**最小二乘法、极大似然法、随机逼近法**等。

经典辨识法：

- 时域法：试验测出响应曲线，确定传递函数种类后求参
  - 阶跃/脉冲响应法：当系统处于稳态时，快速施加一个阶跃/脉冲扰动，并记录输入输出数据
  - 脉冲响应曲线需要换算成阶跃响应曲线才能和标准传递函数的响应曲线比较
- 频率响应法：**在系统的输入端施加不同频率的正弦信号**，然后测量系统输出的稳态响应，再根据幅值比和相位差作出系统的对数频率特性曲线，再将对数幅频曲线用斜率为0dB/dec、±20dB/dec、±40dB/dec…直线分段近似，获得对数幅频渐进特性曲线，并确定最小相位条件下系统的传递函数。
- 相关辨识法：采用伪随机信号作为相关辨识法的输入信号
  - 用M序列作为输入信号，通过输入输出信号求互相关函数，进一步求出脉冲响应函数
  - M序列时用N个以为寄存器所能得到的周期最长的二元序列信号

辨识的对象模型--传递函数

传递函数定义

运算定义（输入输出关系视角）

$$G(s)=\frac{Y(s)}{U(s)}$$

- $U(s)$ ： 输入信号 $u(t)$ 的拉普拉斯变换，输入即系统的“激励”
- $Y(s)$ ： 输出信号 $y(t)$ 的拉普拉斯变换，输出即系统的“响应”
- $G(s)$ ： 传递函数，是系统本身的动态属性
  - 在**零初始条件**下，输出信号的拉普拉斯变换与输入信号的拉普拉斯变换之比
  - 只适用于线性定常系统

> 飞控内环"角速度控制环“
>
> - 输入 $u(t)$:电机力矩差分命令 $\tau(s)$
> - 输出 $y(t)$:机体角速度 $\omega(t)$

极点–零点形式（结构表达式视角）

表达系统在复平面（s平面)中的极点–零点结构特征

$$G(s) = K \cdot \frac{(s - z_1)(s - z_2)\cdots(s - z_m)}{(s - p_1)(s - p_2)\cdots(s - p_n)} \cdot e^{-Ls}$$

项	名称	含义
K	系统增益	系统的比例放大倍数
$(s - z_i)$	零点	当 *s=zi* 时，输出为 0
$(s - p_i)$	极点	当 *s=pi* 时，系统响应发散（无界）
e^{-Ls}	延迟项	表示系统有纯时间延迟 *L* 秒

> 具体极点和零点是什么好复杂看不懂，是复频域的事情了
>
> 阶次n = 极点个数（含重根）
>
> 相对阶n−m
>
> 极点含在原点（p=0)积分型，多个原点极点−>多重积分

传递函数分类

按动态特征分类

类型	典型传递函数形式	主要特征	物理意义/常见场景
比例环节（零阶）	$G(s) = K$	无惯性，无延迟；输出立即变化	放大器、信号缩放
一阶惯性环节	$G(s) = \frac{K}{Ts + 1}$	平滑单调响应，无振荡；时间常数 T 决定速度	电机转速、热系统、飞控内环
二阶振荡环节	$G(s) = \frac{K\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}$	有振荡/超调；阻尼比 ξ 决定形态	弹簧–质量、姿态外环
积分环节	$G(s) = \frac{K}{s}$	对输入积分；响应无稳态	角速度→角度、速度→位置
比例积分环节（PI）	$G(s) = K_p + \frac{K_i}{s}$	零稳态误差、响应更慢	控制系统常用补偿

比例微分环节（PD）	$G(s) = K_p(1 + T_d s)$	相位提前、抑制超调	PID控制前馈部分
纯微分环节	$G(s) = Ks$	对输入变化敏感，放大噪声	理论环节，不可直接实现
纯延迟环节	$G(s) = e^{-Ls}$	输出延后 L 秒开始变化	信号传输、采样、滤波延迟
惯性+延迟	$G(s) = \frac{Ke^{-Ls}}{Ts + 1}$	常见于执行器或飞控内环	电机+采样延迟、四轴飞行器角速度环建模
二阶+延迟	$G(s) = \frac{K\omega_n^2 e^{-Ls}}{s^2 + 2\xi\omega_n s + \omega_n^2}$	慢响应或振荡滞后	姿态外环、机械振动系统

按数学结构特征分类

1. 按阶次（order）

阶次	特征	含义
0阶（比例环节）	分母无（s）	无惯性
1阶（惯性环节）	分母为（Ts + 1）	一种储能元件（惯量、电容等）
2阶	分母为二次多项式	两个能量存储元件（质量+弹簧）
n阶	分母为（n）次多项式	多惯性系统、复合系统

2. 按极点/零点分布（略）
3. 按延迟与非延迟（有无乘 e^{-Ls} ）
4. 按惯性与非惯性

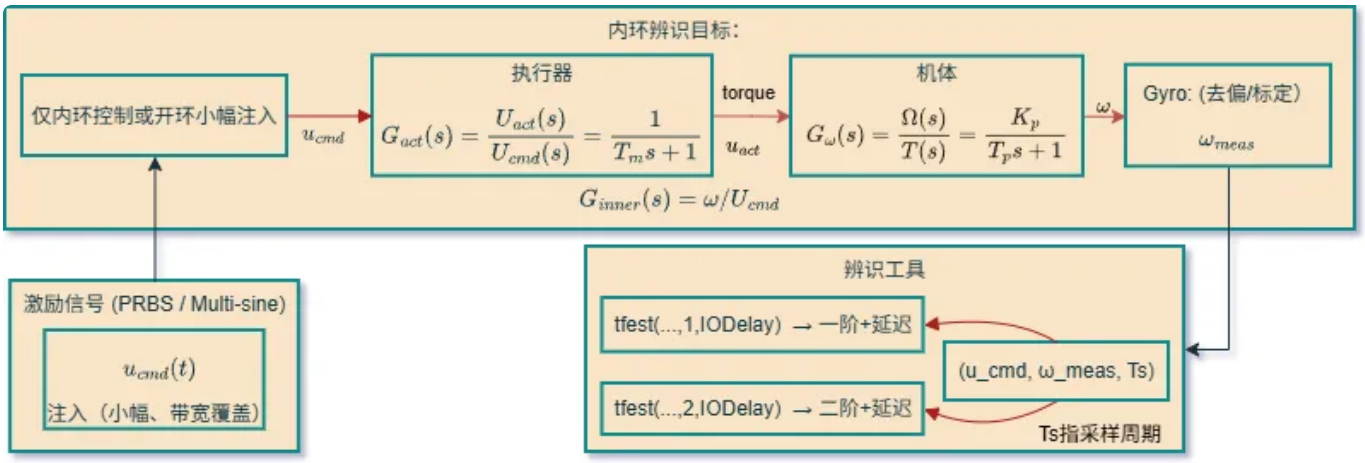
特征	判断	含义
有惯性	传递函数分母含（s）	系统响应随时间逐渐变化，无法瞬间跟随输入
无惯性	传递函数分母为常数	系统输出无延迟、无滞后，随输入即时变化

5. 按能量特性（积分/微分性质）

类型	传递函数	输出特征
积分型系统	$G(s) \propto \frac{1}{s}$	输出随输入的累积持续增长（例如：由角速度积分得到位置）
微分型系统	$G(s) \propto s$	对输入的变化率即时响应，输入变化越剧烈，输出响应越显著
复合型系统	传递函数含多种项（如 $G(s) = K_p + \frac{K_i}{s} + K_d s$ ）	综合多种特性（如PID控制器，同时具备比例、积分、微分环节的功能）

飞控内环--角速度环的传递函数

准确来说是二阶延迟模型，可以简化为一阶延迟模型



对四轴飞行器任一转动通道来说，刚体动力学的基本方程是： $J\dot{\omega} = \tau - D\omega$

符号	含义	单位
J	转动惯量（惯性大小）	kg·m²
ω	角速度（Rate）	rad/s
$\dot{\omega}$	角加速度	rad/s²

τ	电机产生的控制力矩（输入）	N·m
$D\omega$	阻尼力矩（空气阻力、摩擦力、反电势等）	N·m

对上式两边进行**拉普拉斯变换（Laplace Transform）**，并假设初始角速度为零（ $\omega(0)=0$ ）：

$Js\omega(s)=\tau(s)-D\omega(s)$

整理得：$\frac{\omega(s)}{\tau(s)}=\frac{1}{Js+D}$ 即 $G(s)$

如果定义两个参数：$T=\frac{J}{D}$ （时间常数），$K=\frac{1}{D}$ （稳态增益）

> $\frac{\text{单位 of } J}{\text{单位 of } D}=\frac{\mathrm{N\cdot m\cdot s^2/rad}}{\mathrm{N\cdot m\cdot s/rad}}=s$

则可改写成常见的一阶惯性形式 $G(s)=\frac{K}{Ts+1}$

实际中，除了物理惯性还有额外的延迟效应，比如电机响应时间，滤波器延迟，通信延迟，采样和计算延迟。这些延迟总体用一个**纯滞后项**近似表示 e^{-Ls} ,L为延迟时间

最终 $G(s)=\frac{Ke^{-Ls}}{Ts+1}$ 一阶惯性+延迟（FOPDT:First-Order+Dead time）

MATLAB中的辨识手段

理论在MATLAB中对应的具体方法和典型函数

1）非参数模型（先看数据的“外形”）

非参数＝不先假设极点/零点结构，直接从数据得到脉冲/阶跃/频响，再需要时“再转参数化”。

常用函数与方法

- 频率响应估计（经典“频率响应法”）
  - `etfe`：经验传函估计（Empirical Transfer Function Estimate）
  - `spa`：谱分析频响估计（对噪声更稳健）
  - 输出是 `idfrd`（频域数据对象），可**再拟合**为传递函数
- 脉冲/阶跃响应（经典“时域法”）
  - `impulseest`、`stepest`：直接从输入输出数据估计非参数脉冲/阶跃响应

- 相关法（经典“相关辨识法”）
  - `cra`：输入/输出相关性分析，估计冲激响应趋势

把非参数转为参数（关键桥）

- `tfest`：可吃 `idfrd`（频域数据）或 `iddata`（时域数据），拟合到**传递函数（参数模型）**
- `procest`：直接把数据拟合成**工艺过程模型**（如一阶+延迟 P1D、二阶+延迟 P2D……）

简明示例（频响→传函）：

```
Plain Text
```

```
data = iddata(y, u, Ts);
Gnp  = spa(data);                    % 非参数频响
sys  = tfest(Gnp, 1, 0);             % 拟合为 1 极 0 零（可改阶数/延迟）
bode(Gnp, sys), grid on
```

2）参数模型（直接得到极点/零点/延迟）

参数＝直接输出可用于控制器设计的传递函数/状态空间模型。

A. 传递函数（经典 + 现代）

- `tfest`：预测误差法 PEM（对应“极大似然/最小二乘的等价情形”，现代辨识）
- `procest`：过程模型（P1D/P2D/P3D 等），更贴合**一阶+延迟**这类工程对象（经典“过程辨识法”的现代实现）
- `delayest`：估计纯延迟

B. 盒式结构（ARX/ARMAX/OE/BJ）（经典统计建模）

- `arx`：最小二乘，入门稳健
- `armax`：在 arx 基础上加噪声模型
- `oe`（Output–Error）：只拟合确定性部分，适合执行器噪声小、测量噪声主导
- `bj`（Box–Jenkins）：对噪声/干扰建模最丰富

C. 状态空间（现代子空间/极大似然）

- `n4sid`：**子空间法**（现代辨识），强项是中高阶与多变量
- `ssest`：PEM（极大似然）估计连续/离散**状态空间**模型
- `ssregest`：带正则的状态空间（高维/稀疏先验时更稳）

D. 灰箱/先验结构（现代）

- `greyest` + `idgrey`：把你的物理微分方程（例如欧拉方程 + 执行器一阶 + 阻尼）放进去，让

算法只估未知参数（惯量/阻尼/延迟）

> 内环实现顺序
>
> - `procest(data,'P1D')` 或 `tfest(data,1,0)`（让 `delayest` 或 `tfest` 自动带延迟）
> - 先 `spa/etfe` 看频响走向，再用 `tfest(Gnp,1,0)` 拟合，验证 Bode/Step 是否一致
> - 如果数据噪声强、测量占主导，用 `oe(data,[nb nf nk])`（常比 arx 更贴对象）
> - 有明确先验（执行器一阶/阻尼），用 `greyest` 做灰箱最稳

tfest如何工作

两种辨识路线

激励信号设计

类型	波形示意	主要特性	优点	缺点	是否适合内环辨识
阶跃信号（Step）	┌────────	单次突变，频谱包含多频成分	简单、直观；易于分析时域特征（如 T、K）	频谱稀疏；难区分高频动态；激烈输入可能导致系统饱和	✅ 可用（初步估计参数）
脉冲信号（Impulse）	尖峰脉冲	极短高幅冲击，理论上激励所有频率	理论频谱宽度广	实际工程中难实现；信噪比差	❌ 不适用于电机 / 四轴
斜坡信号（Ramp）	/（线性上升线）	连续线性变化	平稳，输入强度易控制	激励频率低，包含系统动态的信息量不足	❌ 不推荐
正弦波扫描（Sine sweep /chirp）	~~~~（频率渐变正弦波）	频率随时间连续变化	能全面获得系统频率响应，覆盖带宽可控	实验耗时较长	✅ 推荐用于频响辨识

多正弦信号（Multi-sine）	多频率正弦波叠加组合波形	同时激励多个特定频点	快速获取宽频信息；功率在各频点分布均匀	信号设计复杂；需满足系统线性假设	✅ 很适合内环辨识
伪随机二进制序列（PRBS）	0100110...（高低电平随机切换）	频谱近似白噪声，仅 0/1 两电平切换	易实现（数字 PWM 输出）；频谱均匀；不易触发系统饱和	响应含一定噪声；需较长时间信号平均以降噪	✅✅ 推荐！最常用！
白噪声（White noise）	无规则随机波动	理论上激励全频段	理论最理想；可直接计算频响	实际实现带宽有限；高强度随机输入对电机不安全	⚠️ 不建议在实体机上直接使用
随机正弦叠加（Sum-of-sines）	Σsin（多随机相位正弦波叠加）	受控随机性，激励平稳	输入平滑，对系统线性度辨识效果好	信号生成逻辑复杂	✅ 推荐替代白噪声

（1）激励频谱要覆盖系统带宽

系统辨识的目标是估计 G(jω)；
若激励信号的频率分布（功率谱密度）没有覆盖系统的工作频段，
→ 得到的模型只反映部分动态。

举例：
四轴内环带宽通常在 20–40 Hz；
那么激励频率范围最好覆盖 1 Hz ~ 80 Hz。

（2）信号能量要足够

输入信号太小 → 输出信号被噪声淹没；
输入信号太大 → 驱动器饱和、非线性失真。
通常选取：

振幅 ≈ 最大可用输入的 20% ~ 50%

路线	流程	优点	缺点	何时优先
直接参数化	`iddata(y,u,Ts) → tfest/procest/oe/arx ...`	1. 一步输出传递函数，可直接用于控制器整定； 2. 实现逻辑简单，操作门槛低； 3. 与 pidtune 工具无缝衔接	1. 若预设的模型结构（阶数/延迟）偏差大，易出现模型“退化”或过拟合； 2. 对数据质量、激励信号的频率覆盖度更敏感	已明确对象结构（如内环近似为“一阶+延迟”），需快速得到可用控制器时
非参数→参数	`spa/etfe/step` 得到频响/阶跃特性 → `tfest/procest` 拟合	1. 先观察数据“外形”（频响/阶跃曲线），更易选择匹配的阶数与延迟； 2. 对噪声的稳健性更强； 3. 便于发现系统异常（如共振、测量滤波干扰）	1. 步骤较直接参数化更多，流程稍复杂； 2. 需理解 idfrd 频域数据格式及频域拟合逻辑； 3. 整体操作稍“厚重”	数据含噪声、未知系统结构（不确定阶数/延迟），或需排查系统异常时

> - 内环≈FOPDT：一阶 + 纯延迟非常适合**直接参数化**，用 `tfest` 或 `procest('P1D')` 就能稳稳拿到模型并喂给 `pidtune` 。
> - 当怀疑“延迟被吃成慢极点、或出现奇怪相位/高频尖峰、或不确定阶数”，再走**非参数→参数**更保险。

脚本中内环辨识部分解析

脚本做的事情就是：**给系统“好”激励 → 记录** u, y **→ 用** `tfest` **拟合** K, T, L ，并自动避免/修复“退化模型”。

1. 基本设置与“物理合理化”

代码要点

```MATLAB
Ts   = 0.002;            % 采样 2ms=500Hz
Jx=0.022; Jy=0.022; Jz=0.044;   % 惯量（F450量级）
L_arm=0.225; k_th=1.6; k_yaw=0.03;

% 执行器一阶滞后（电机/ESC/桨）
T_motor = 0.020;
alpha_m = exp(-Ts/T_motor);

% 转动阻尼 tau_damp = -D*w （轻微即可）
D = diag([0.02, 0.02, 0.03]);

use_noise = true; gyro_std = deg2rad(0.05); % 轻噪声
```

为什么要这样做

- 纯刚体 $J\dot{\omega} = \tau$ 会**非常接近积分**；在小激励下 `tfest` 容易把它“看成”**直流增益或极点≈0**，导致**退化**（出现 $K_p \sim 10^{15}$ 那种）。
- 加**执行器一阶滞后**和**微小阻尼**，让被辨识对象**自带一阶特性**，更接近真实硬件，从源头上减少退化。
- **轻噪声**能打破“完美拟合”，使估计更稳健。

可调建议

- `T_motor`：仿真阶段 10—30 ms；实机时可从 10 ms 起试。
- `D`：从 0.01—0.05（N·m·s/rad）之间试，一点点即可，太大系统会过慢。
- 若估计仍退化，提高噪声或增大激励幅值/时长（后述）。

---

2. 激励设计与 I/O 生成（辨识的地基）

代码要点

```MATLAB
% 多频正弦或 PRBS，保证幅值、频带和时长
switch lower(exct)
  case 'multisine'
    for f in fset: u_cmd += amp*sin(2*pi*f*t + 随机相位)
    u_cmd 限幅到 ±0.6
  case 'prbs'
    baseN=4095; reps=ceil(N/baseN)
    seq = idinput(baseN,'prbs',[0 1],[-amp amp]);
    u_cmd = repmat(seq,reps,1); u_cmd = u_cmd(1:N)
end

% 执行器一阶：u_act = alpha_m*u_act + (1-alpha_m)*u_cmd
% 力矩映射：roll: [k_th*L*u_act;0;0] 等
% 刚体欧拉 + 阻尼：J*wdot = tau - D*w - w×(Jw)
% 选定轴的角速度作为 y，u 用“指令侧”（把执行器也并入对象）
```

为什么要这样做

- **多频正弦**：频率点可控（1/2/3/5/8/12…Hz），覆盖目标带宽×5~10 倍，适合看频响走势。
- **PRBS**：宽频、均匀，经典辨识激励；**必须保证满周期**（4095 点），否则 MATLAB 会警告“只取了前 4001 个值”，频率覆盖不完整→估计偏差。
- **u 用指令侧**：把执行器一阶一起当作对象的一部分辨识（符合“控制器→对象”的实际链路）。

可调建议

- `amp`：0.3–0.6；太小 SNR 低，太大易触发非线性。
- 多频频点上限：≳ 目标带宽的 5–10 倍（内环带宽 4–8 Hz，可把最高点拉到 20 Hz 左右）。
- 时长：PRBS 至少 1 周期；多频正弦≥ 8–12 s。

---

3. 退化检测（早发现、早止损）

代码要点

```MATLAB
function bad = is_degenerate(sys, data)
  p = pole(sys); z = zero(sys);
  if any(abs(p)<1e-4) or any(abs(z)<1e-4): bad=true
  G0 = dcgain(sys); if ~finite(G0) or |G0|>1e6: bad=true
  [~,fit,~] = compare(data, sys); if fit>99.9: bad=true
```

这段在“守护”什么

- **极点/零点 ≈ 0**：伪积分/伪微分 → 典型退化征兆。
- **直流增益爆炸**：模型数值失真。
- **拟合率 99.9%**：在无噪环境常常意味着“把噪声（或伪迹）也拟合了”，大概率不物理。

为什么要这样做

- 遇到过“完美拟合 + 巨大 Kp/Ki”的典型退化；有了这一关，脚本能**自动换策略**而不是给出“好看但不可用”的结果。

---

4. 自动重试流水线（把经验写进代码）

代码要点（示意）

```MATLAB
retry = {
  % 先多正弦，一阶+自动延迟
  {'multisine', amp=0.40, np=1, nz=0, delay='auto', T=8.2}
  % 增幅
  {'multisine', amp=0.50, np=1, nz=0, delay='auto', T=8.2}
  % 换 PRBS
  {'prbs',      amp=0.50, np=1, nz=0, delay='auto', T=8.2}
  % PRBS + 延长
  {'prbs',      amp=0.60, np=1, nz=0, delay='auto', T=10.0}
  % 多正弦 + 增阶（1零1极）
  {'multisine', amp=0.60, np=1, nz=1, delay='auto', T=10.0}
  % PRBS + 固定无延迟（必要时）
  {'prbs',      amp=0.60, np=1, nz=0, delay=0,      T=10.0}
};

for 每个方案:
  生成 u,y → data
  sys_tmp = tfest(data, np, nz, delay=auto/0)
  if ~is_degenerate(sys_tmp): sys_est=它; break;
end
```

为什么要这样做

- **换激励/幅值/时长**：保障频域覆盖和 SNR。
- **换模型结构**：有些数据里延迟被“吃成慢极点”，或确实需要 1 个零点来匹配高频；自动切换，避免人工试错。

- **固定 delay=0**：用于“强行不估延迟”的兜底（某些数据更稳定）。

你可以怎么定制

- 把你“最信任”的结构排在前面（比如 `procest('P1D')` 也可加入作为第一候选）。
- 把 `np,nz` 、amp、Tsec 调成你自己的默认习惯。

---

5. `tfest` 调用与参数估计（核心一步）

代码要点

```MATLAB
opt = tfestOptions('InitializeMethod','all'); % 允许多种初始化
sys_est = tfest(data, np, nz, opt);           % 或 tfest(data,np,nz,delay,opt)
```

发生了什么

- `tfest` 用**预测误差法（PEM）最小化 Σ(y−y^)2\sum (y-\hat y)^2Σ(y−y^)2，直接给出连续时间传递函数**（可带纯延迟）。
- `'InitializeMethod','all'`：多起点/多策略初始化，**更不容易卡坏局部极小**，对有延迟的对象很重要。
- 你也可以先 `delayest(data)` 固定 LLL，再 `tfest(..., L)`，避免延迟被“吃成慢极点”。

质量评估要看什么

```MATLAB
bode(sys_est); step(sys_est); compare(data, sys_est)
resid(data, sys_est)    % 残差是否近似白噪声
pole(sys_est)           % 极点是否全在左半平面
dcgain(sys_est)         % 增益是否合理
```

一阶对象的“像样”迹象

- Bode 幅频：低频平坦，高频 −20 dB/dec 滚降；相位平滑接近 −90°；
- Step：指数型单调上升，无持续振荡；
- compare 拟合 ⩾ 80%；
- 极点远离 0（不应该是 1e−10 这种量级）。