

自动控制理论——状态空间模型

Python 综合应用与仿真考察

参考教材：《自动控制理论——状态空间》

说明：本测试旨在考察利用 Python 进行状态空间建模、系统分析（能控/能观性）、控制器设计（极点配置）及观测器设计的能力。请结合 PPT 内容完成以下题目。

题目 1：弹簧-质量-阻尼系统的状态空间建模与响应仿真

题目描述：参考 PPT 的弹簧-质量-阻尼系统，已知系统微分方程为：

$$m\ddot{x} + b\dot{x} + kx = f(t) \quad (1)$$

假设参数为 $m = 2 \text{ kg}$, $b = 1 \text{ N} \cdot \text{s/m}$, $k = 4 \text{ N/m}$ 。

1. 请定义状态变量 $z_1 = x$ (位移) 和 $z_2 = \dot{x}$ (速度)，写出该系统的状态空间矩阵 A, B, C, D (假设输出 y 为位移 x)。
2. **Python 编程任务：**使用 `scipy.integrate.odeint` 或 `scipy.signal.StateSpace` 编写代码，完成以下仿真：
 - 工况 A (零输入响应)：外力 $f(t) = 0$ ，初始状态为 $x(0) = 1\text{m}, \dot{x}(0) = 0$ 。
 - 工况 B (零状态响应)：初始状态为 0，外力 $f(t)$ 为单位阶跃信号（即 $t \geq 0$ 时 $f(t) = 1\text{N}$ ）。
3. 请在同一张图中绘制出工况 A 和工况 B 下的位移 $x(t)$ 和速度 $\dot{x}(t)$ 随时间变化的曲线，并分析系统的阻尼特性（过阻尼、欠阻尼或临界阻尼）。

题目 2：多变量系统的能控性与能观性自动化判定工具

题目描述：参考 PPT 的双小车系统，已知其状态矩阵形式如下（注：此处为示例结构，具体数值参考 PPT）：

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & -1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad (2)$$

- 1. Python 编程任务:** 编写一个通用的 Python 函数 `analyze_system(A, B, C)`，该函数需要实现以下功能：
 - 自动构建能控性矩阵 $\mathcal{C}_o = [B, AB, \dots, A^{n-1}B]$ 。
 - 自动构建能观性矩阵 $\mathcal{O} = [C^T, (CA)^T, \dots, (CA^{n-1})^T]^T$ 。
 - 使用 `numpy.linalg.matrix_rank` 计算矩阵的秩，并根据系统阶数 n 输出系统是否“完全能控”和“完全能观”的结论。
- 2. 应用该函数分析上述双小车系统。如果系统是能控的，请结合 PPT 解释其物理含义：这是否意味着我们可以通过施加在第一个小车上的力，间接控制第二个小车的位置？**

题目 3：倒立摆系统的极点配置控制器设计

题目描述：参考 PPT 中“指尖上的平衡”（倒立摆）模型。已知线性化后的状态方程矩阵为：

$$A = \begin{bmatrix} 0 & 1 \\ g/d & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (3)$$

假设物理参数 $g = 9.8, d = 1$ 。

- 1. Python 编程任务：**
 - 计算开环矩阵 A 的特征值，验证 PPT 中关于平衡点不稳定（鞍点）的结论。
 - 设计状态反馈控制器 $u = -Kz$ 。利用 Python（如 `scipy.signal.place_poles`）计算增益矩阵 $K = [k_1, k_2]$ ，使得闭环系统的极点配置在 $\lambda_{1,2} = -2 \pm 1j$ （或参考 PPT 设定为实数极点）。
- 2. 模拟该闭环系统在初始扰动 $z(0) = [0.1, 0]^T$ （即偏离平衡点 0.1 弧度）下的响应，并绘制角度 $\phi(t)$ 的收敛曲线。**
- 3. 开放性讨论：**如果将闭环极点配置得离虚轴非常远（例如 $\lambda = -10$ ），对控制器输出 $u(t)$ （即控制力矩）会有什么影响？在实际物理系统中，这种配置是否存在局限性？

题目 4：基于观测器的状态估计仿真

题目描述：参考 PPT 的线性观测器设计案例。考虑一个质量-弹簧系统，假设只能测量位移 $y = x$ ，但控制反馈需要速度 \dot{x} 。系统矩阵如下：

$$A = \begin{bmatrix} 0 & 1 \\ -1 & -0.5 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad (4)$$

- 1. Python 编程任务：**

- 构建模拟真实系统状态 $\dot{z} = Az + Bu$ 的过程。
 - 构建观测器方程 $\dot{\hat{z}} = (A - LC)\hat{z} + Bu + L(y - C\hat{z})$ 。
 - 设计观测器增益 L , 使得观测器误差的衰减速度比系统本身的动态快 (例如配置观测器极点实部为系统主导极点实部的 3 倍)。
2. 设置真实初始状态 $z(0) = [1, 0]^T$, 但观测器初始猜测 $\hat{z}(0) = [0, 0]^T$ (即存在初始估计误差)。
 3. 仿真并绘制真实速度 $z_2(t)$ 与估计速度 $\hat{z}_2(t)$ 的对比曲线, 展示估计值是如何在有限时间内追踪上真实值的。

题目 5：轨迹追踪与前馈控制实现

题目描述: PPT 提出了“轨迹追踪”问题, 目标不再是保持在零平衡点, 而是让状态 $z(t)$ 追踪期望轨迹 z_d 。控制律设计为:

$$u(t) = Fz_d + K_e(z_d - z(t)) \quad (5)$$

1. Python 编程任务:

- 定义一个期望轨迹 $z_d(t)$, 例如一个方波信号 (模拟倒立摆需要在两个设定角度之间切换)。
 - 基于题目 3 中的倒立摆模型, 计算前馈增益 F 和反馈增益 K_e 。
 - 编写仿真代码, 模拟系统在控制律 $u(t)$ 作用下的响应。
2. 绘制系统输出 $y(t)$ 与期望轨迹 $z_d(t)$ 的对比图。
 3. **分析:** 如果去掉前馈项 Fz_d , 仅使用误差反馈 $K_e(z_d - z(t))$, 系统最终能无误差地稳定在目标位置 z_d 吗?