

自动控制原理 II：线性系统分析与设计

课内实验

系统设计部分

概述：本部分实验主要内容

- 系统设计的一般流程
- 镇定设计：实现系统满足渐近稳定
- 极点配置：实现系统满足给定性能条件
- 状态观测：利用状态观测器估计状态、利用观测估计值设计状态反馈

概述：重点&预备知识

➤ 重点

- ❑ 了解系统设计一般流程
- ❑ 典型综合问题：镇定、极点配置、状态估计、基于状态观测器的状态反馈
- ❑ 常用控制器增益计算方法：LMI、极点配置

➤ 预备知识

- ❑ MATLAB基础：控制结构图搭建、曲线绘制、等常规操作
- ❑ 线性系统基础：线性定常系统综合相关理论知识

课内实验：设计部分

3.1 系统设计一般流程

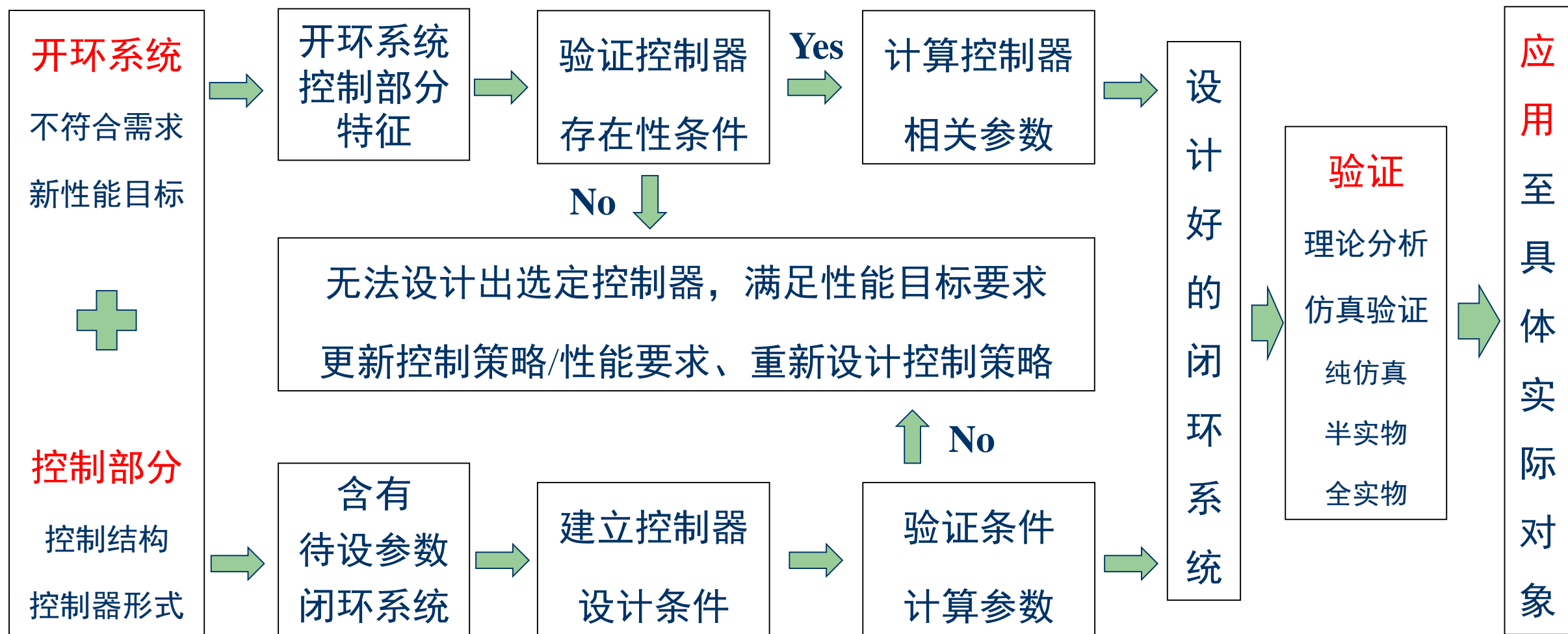
3.2 系统镇定设计

3.3 系统极点配置

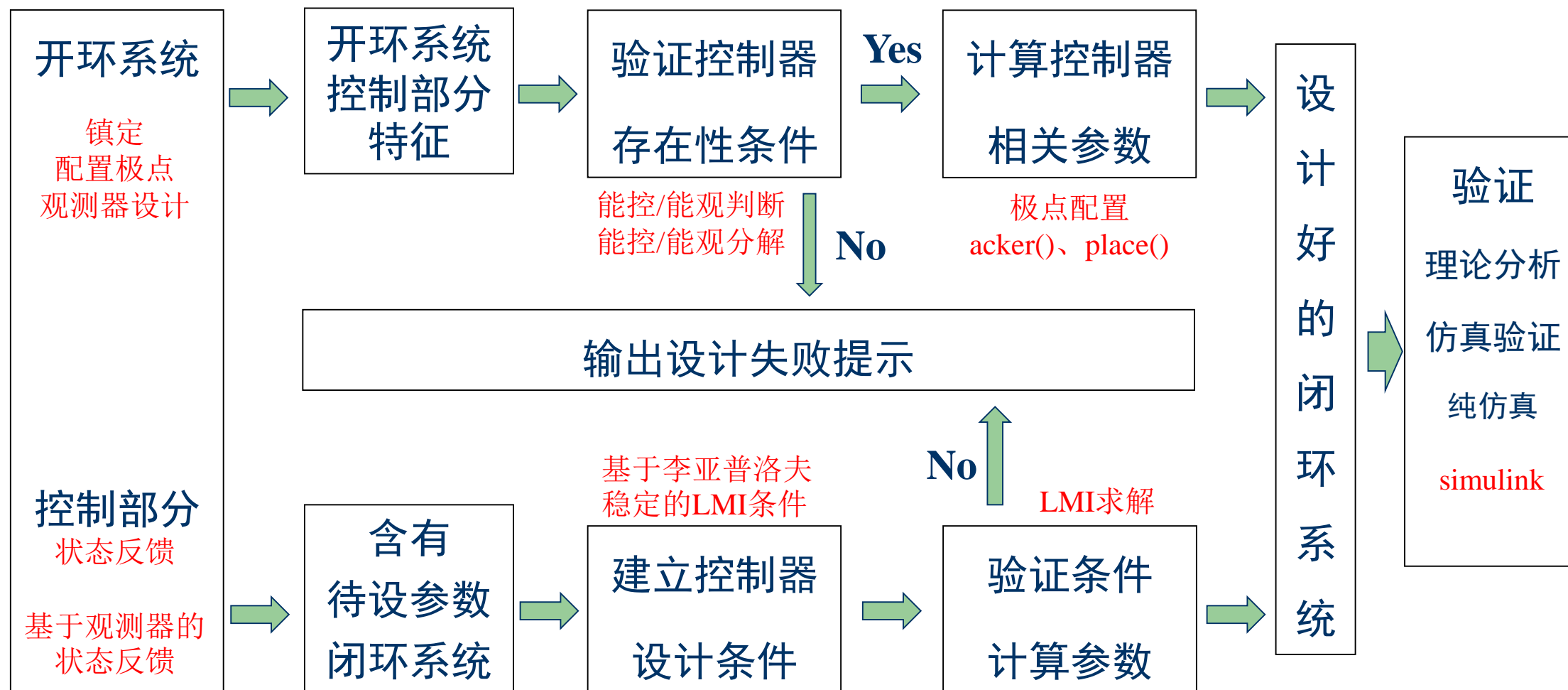
3.4 系统状态观测器设计

3.5 基于状态观测器的状态反馈

系统设计一般流程



系统设计实验内容



课内实验：设计部分

3.1 系统设计一般流程

3.2 系统镇定设计

3.3 系统极点配置

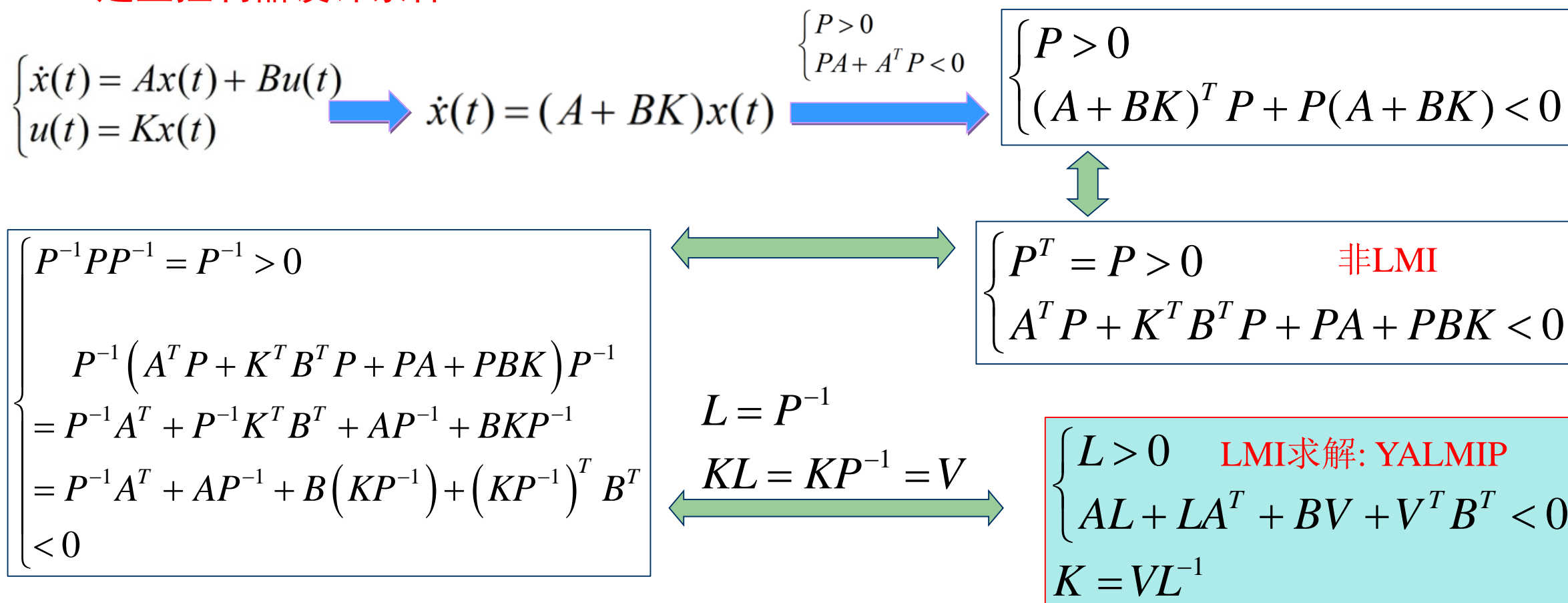
3.4 系统状态观测器设计

3.5 基于状态观测器的状态反馈

系统镇定设计：渐近稳定

□ 内容：状态反馈（控制策略结构） + LMI（增益求解方法） + 验证（simulink仿真）

➤ 建立控制器设计条件



系统镇定设计：渐近稳定

□ 基于LMI设计镇定控制器

$$\begin{cases} L > 0 \\ AL + LA^T + BV + V^T B^T < 0 \end{cases} \Rightarrow K = VL^{-1}$$

➤ YALMIP工具箱

例3-1：设计镇定控制器

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} 1 & 2 & 4 \\ 1 & 1 & 1 \\ 0 & 2 & 1 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} u(t) \\ u(t) = Kx(t) \end{cases}$$

结果与验证（略），自己完成

```
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 %%% Chuan-Ke Zhang
3 %%% 2018-01-19
4 %%% 设计系统镇定控制器程序
5 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
6
7 clc; clear
8
9 %%% 系统参数
10 A = [1 2 4; 1 1 1; 0 2 1];
11 B = [1; 2; 1];
12
13 %%% 描述待求的LMI
14 L = sdpvar(3,3,'symmetric'); % 给出待求矩阵
15 V = sdpvar(1,3,'full'); % 给出待求矩阵
16 Fcond = [L>0, A*L+L*A'+B*V+V'*B'<0]; % 列出所有待求LMI
17
18 %%% 求解LMI
19 ops = sdpsettings('verbose',0,'solver','sedumi'); % 设置求解环境
20 diagnostics = solvesdp(Fcond,[],ops); % 迭代求解
21 [m,p] = checkset(Fcond); % 返回求解结果
22 tmin = min(m); % 验证是否满足
23
24 if tmin > 0
25     Vh = double(V);
26     Lh = double(L);
27     disp('System can be stabilized, and the control gain is given as') % 结论输出
28     K = Vh*inv(Lh)
29 else
30     disp('System cannot be stabilized using state-feedback controller') % 结论输出
31 end
```

课内实验：设计部分

3.1 系统设计一般流程

3.2 系统镇定设计

3.3 系统极点配置

3.4 系统状态观测器设计

3.5 基于状态观测器的状态反馈

系统极点配置：动态性能

□ 内容：状态反馈（控制结构） + 理论方法/内嵌算法（增益求解） + 验证（仿真/理论）

➤ 课堂方法计算增益：期望特征方程 VS 实际特征方程（自学，MATLAB符号运算）

➤ 内嵌算法计算增益

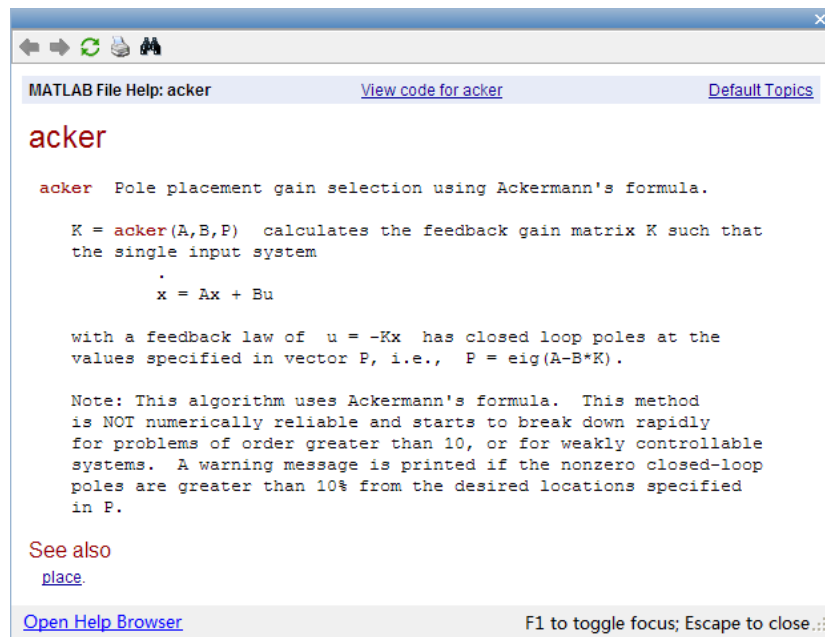
acker()函数、place()函数

功能：计算给定极点下的
状态反馈增益。格式：

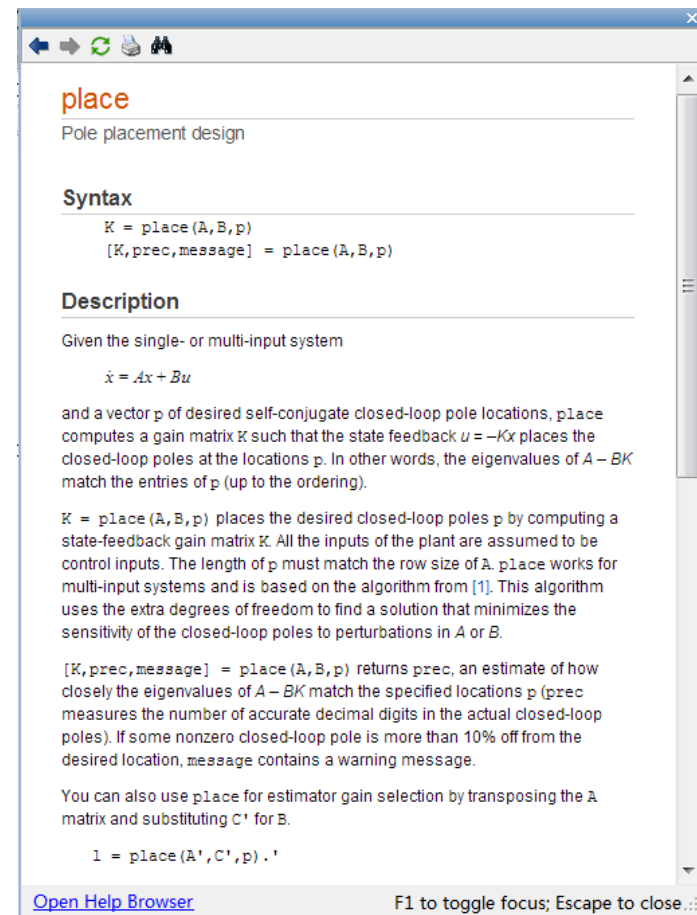
$K = \text{acker}(A, b, P)$

$K = \text{place}(A, B, P)$

其中，P为目标极点
K为反馈增益



The image shows the MATLAB File Help window for the `acker` function. The title bar says "MATLAB File Help: acker". There are links for "View code for acker" and "Default Topics". The main content area has the heading "acker" and a description: "Pole placement gain selection using Ackermann's formula." It includes the syntax: `K = acker(A,B,P)` calculates the feedback gain matrix K such that the single input system $\dot{x} = Ax + Bu$ with a feedback law of $u = -Kx$ has closed loop poles at the values specified in vector P, i.e., $P = \text{eig}(A-B*K)$. A note states: "Note: This algorithm uses Ackermann's formula. This method is NOT numerically reliable and starts to break down rapidly for problems of order greater than 10, or for weakly controllable systems. A warning message is printed if the nonzero closed-loop poles are greater than 10% from the desired locations specified in P." There is a "See also" section with a link to `place`. At the bottom, there is a link "Open Help Browser" and a status bar "F1 to toggle focus; Escape to close..".



The image shows the MATLAB File Help window for the `place` function. The title bar says "MATLAB File Help: place". There are links for "View code for place" and "Default Topics". The main content area has the heading "place" and a description: "Pole placement design". It includes the syntax: `K = place(A,B,p)` and `[K,prec,message] = place(A,B,p)`. A description states: "Given the single- or multi-input system $\dot{x} = Ax + Bu$ and a vector p of desired self-conjugate closed-loop pole locations, place computes a gain matrix K such that the state feedback $u = -Kx$ places the closed-loop poles at the locations p. In other words, the eigenvalues of $A - BK$ match the entries of p (up to the ordering)." It also explains that `K = place(A,B,p)` places the desired closed-loop poles p by computing a state-feedback gain matrix K. All the inputs of the plant are assumed to be control inputs. The length of p must match the row size of A. place works for multi-input systems and is based on the algorithm from [1]. This algorithm uses the extra degrees of freedom to find a solution that minimizes the sensitivity of the closed-loop poles to perturbations in A or B. The `[K,prec,message] = place(A,B,p)` returns prec, an estimate of how closely the eigenvalues of $A - BK$ match the specified locations p (prec measures the number of accurate decimal digits in the actual closed-loop poles). If some nonzero closed-loop pole is more than 10% off from the desired location, message contains a warning message. It also mentions that you can use place for estimator gain selection by transposing the A matrix and substituting C' for B. At the bottom, there is a link "Open Help Browser" and a status bar "F1 to toggle focus; Escape to close..".

$$u = Kx \xleftarrow{\text{教材}} \text{SF} \xrightarrow{\text{MATLAB}} u = -Kx$$

系统极点配置：动态性能

例3-2：利用SF，将系统极点配置到P

$$\dot{x} = \begin{bmatrix} -2 & -1 & 1 \\ 1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} x + \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} u$$

尝试：

- ✓ 测试并比较程序运行结果
- ✓ 解释程序代码作用
- ✓ 找出代码中的问题
- ✓ 编写正确且较完整代码
- ✓ 将极点改为(-1,-1,-1)

```
A=[-2, -1, 1; 1, 0, 1; -1, 0, 1];  
b=[1; 1; 1];  
Qc=ctrb(A, b);  
rc=rank(Qc);  
P=[-1, -2, -3];  
K=acker(A, b, P)
```

```
A=[-2, -1, 1; 1, 0, 1; -1, 0, 1];  
b=[1; 1; 1];  
Qc=ctrb(A, b);  
rc=rank(Qc);  
P=[-1, -2, -3];  
K=place(A, b, P)
```

思考并完成：

- ✓ 理论验证结果正确性，仿真验证
- ✓ 给定超调量和过渡时间，设计状态反馈控制（选）

课内实验：设计部分

3.1 系统设计一般流程

3.2 系统镇定设计

3.3 系统极点配置

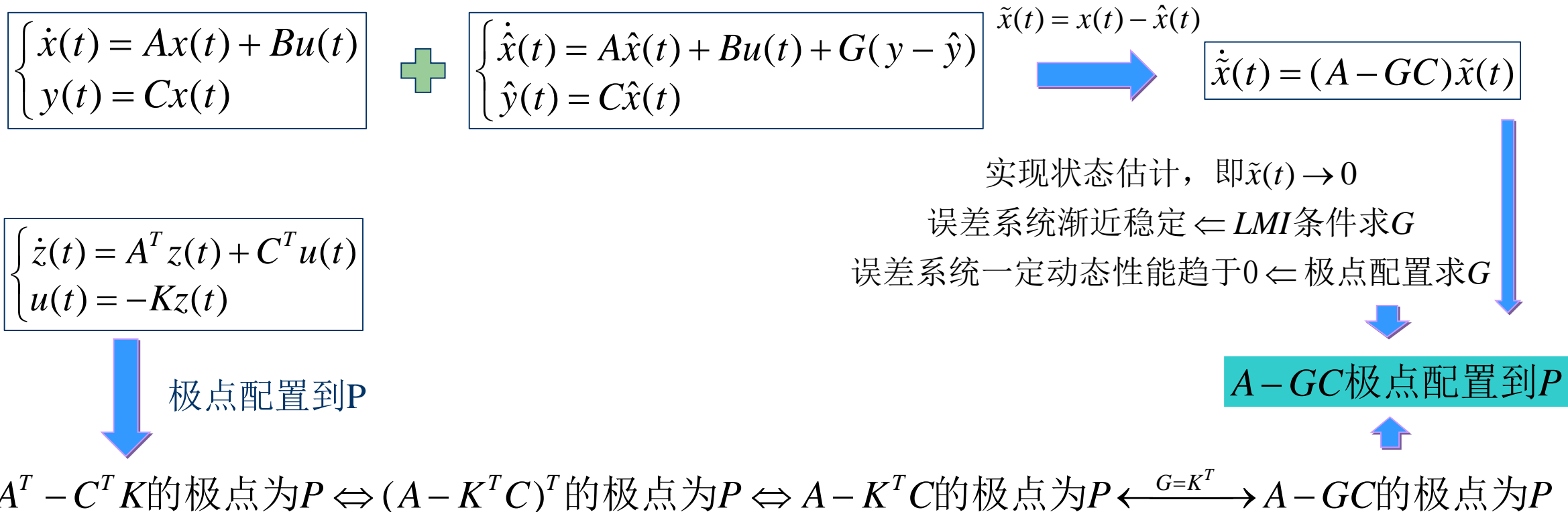
3.4 系统状态观测器设计

3.5 基于状态观测器的状态反馈

系统全维观测器设计

□ 内容：观测器结构 + 极点配置（增益求解）+ 验证（simulink仿真）

➤ 建立观测器设计思路



➤ 求解G步骤: $K = \text{acker/place}(A^T, C^T, P) \rightarrow G = K^T$

系统全维观测器设计

例3-3：设计状态观测器，极点配置到P

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 0 & -2 \\ 1 & 0 & 9 \\ 0 & 1 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix} \mathbf{u}$$
$$\mathbf{y} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \mathbf{x}$$

尝试：

- ✓ 调试程序，获得结果
- ✓ 解释程序代码作用
- ✓ 找出代码可能存在的问题，并更正
- ✓ 利用simulink观察状态估计误差

```
A=[0 0 2;1 0 9;0 1 0];
B=[3;2;1];
C=[0 0 1];
n=3;
Qo=obsv(A,C);
ro=rank(Qo);
if (ro==n)
    disp('系统是可观测的')
    P=[-3 -4 -5]; %状态观测器的设计
    A1=A';
    B1=C';
    K=acker(A1,B1,P);
    G=K';
    AGC=A-G*C;
elseif (ro~=n)
    disp('系统是不可观测的，不能进行观测器的设计')
end
```

课内实验：设计部分

3.1 系统设计一般流程

3.2 系统镇定设计

3.3 系统极点配置

3.4 系统状态观测器设计

3.5 基于状态观测器的状态反馈

基于状态观测器的状态反馈

□ 内容：基于观测器的状态反馈结构 + 极点配置（增益求解）+ 验证（simulink仿真）

➤ 设计思路：分离原理，分步设计

✓ 开环系统

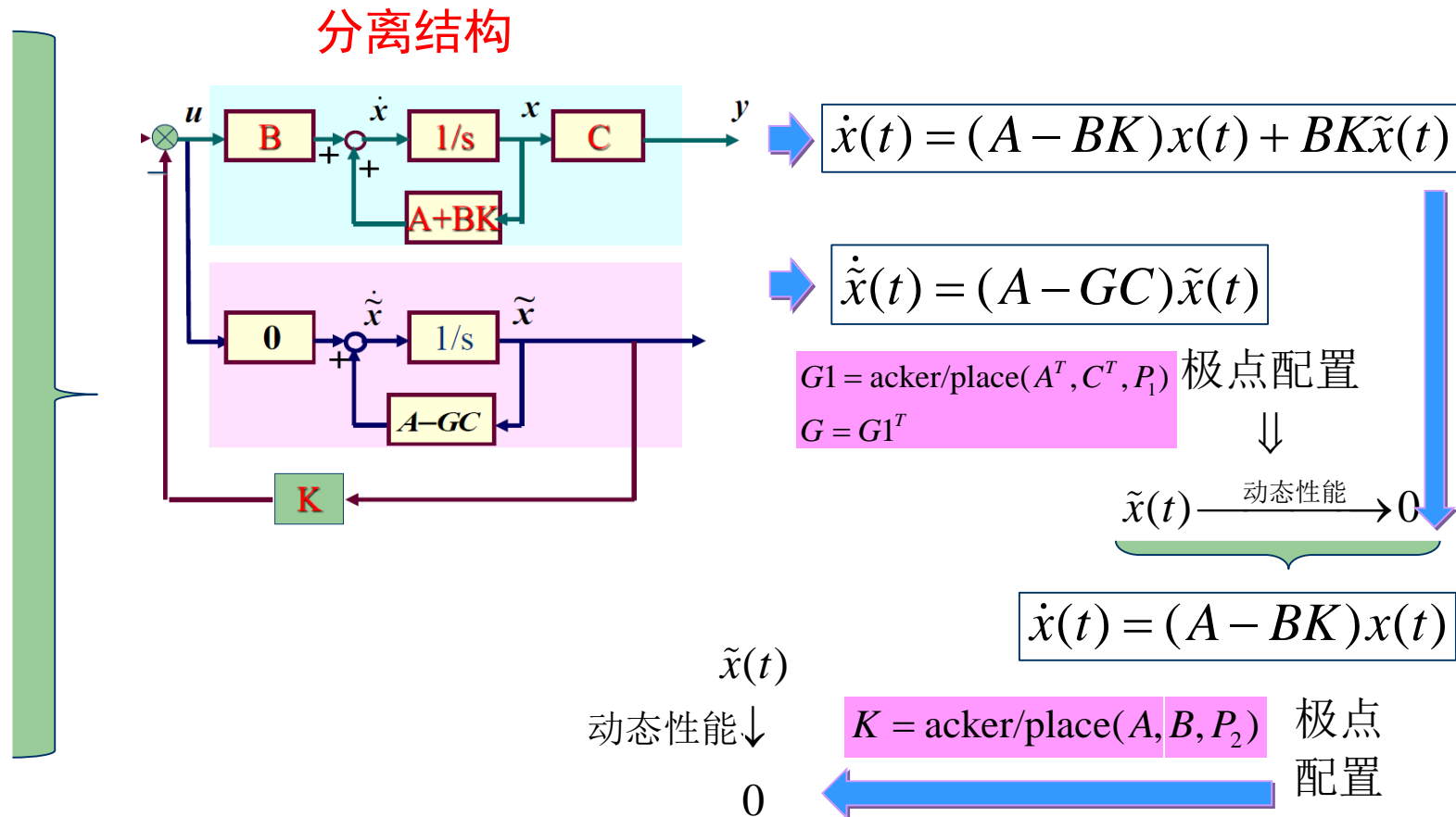
$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases}$$

✓ 观测器系统

$$\begin{cases} \dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + G(y - \hat{y}) \\ \hat{y}(t) = C\hat{x}(t) \end{cases}$$

✓ 反馈控制器

$$u(t) = -K\hat{x}(t)$$



基于状态观测器的状态反馈

例3-4：针对如下开环系统

$$\begin{cases} \dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 20.6 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{u} \\ \mathbf{y} = \begin{bmatrix} 1 & 0 \end{bmatrix} \mathbf{x} \end{cases}$$

1. 设计状态观测器，极点 $P_1: -1.8 \pm 2.4i$
2. 设计基于状态观测器的状态反馈控制器，
极点 $P_2: -8, -8$

$$\begin{cases} \dot{\hat{\mathbf{x}}} = \begin{bmatrix} 0 & 1 \\ 20.6 & 0 \end{bmatrix} \hat{\mathbf{x}} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \mathbf{u} + \begin{bmatrix} 16 \\ 84.6 \end{bmatrix} (y - \hat{y}) \\ \hat{y} = \begin{bmatrix} 1 & 0 \end{bmatrix} \hat{\mathbf{x}} \\ \mathbf{u} = -\begin{bmatrix} 29.6 & 3.6 \end{bmatrix} \hat{\mathbf{x}} \end{cases}$$

思考：验证状态观测器、状态反馈控制器的效果？

Command Window

系统能观，可任意配置观测器系统极点

G =

16.0000

84.6000

系统能控，可通过状态反馈任意配置极点

K =

29.6000 3.6000

fx >>

Editor - C:\Users\hp\Desktop\张传科\20210901-张传科&何老师 线

ex3_4.m

```
1 %%%%%%%%%%%
2 %% Chuan-Ke Zhang
3 %% 2021-10-20
4 %% Example 3-4
5 %% 设计基于状态观测器的状态反馈，实现极点配置
6 %%%%%%%%%%%
7 clc
8 clear
9
10 A = [0 1; 20.6 0];
11 B = [0; 1];
12 C = [1 0];
13
14 n = size(A);
15
16 if rank(observ(A,C))==n
17     disp('系统能观，可任意配置观测器系统极点')
18     P2 = [-8,-8]; % 观测器系统极点
19     G = (acker(A',C',P2))'
20 else
21     disp('系统不完全能观，不可以对观测器系统进行极点任意配置')
22 end
23
24 if rank(ctrb(A,B))==n
25     disp('系统能控，可通过状态反馈任意配置极点')
26     P1 = [-1.8+2.4*i,-1.8-2.4*i]; % 反馈系统极点
27     K = acker(A,B,P1)
28 else
29     disp('系统不完全能控，不可以通过状态反馈实现极点任意配置')
30 end
```

本部分实验小结

□ 系统分析

- 熟悉仿真结果的呈现（主要为响应曲线）

□ 系统分析

- 熟悉设计的一般过程
- 如何用MATLAB设计镇定控制器、极点配置（基于状态反馈）
- 如何用MATLAB设计状态观测器（全维）
- 如何用MATLAB设计基于状态观测器的状态反馈（实现某极点要求）
- 如何用MATLAB验证控制器效果、及相互比较

本部分实验练习题

□ 作业一（综合性题）：

- 设计状态反馈控制器，极点P1
- 设计状态观测器，极点P2
- 设计基于状态观测器的状态反馈，极点P1
- P1, P2在合理条件下任意选
- 验证状态反馈控制效果
- 验证状态观测器效果
- 验证基于状态观测器的状态反馈控制效果
- 比较上述两类状态反馈的效果

$$\begin{cases} \dot{\mathbf{x}} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 2 & -3 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 4 & -1 & 2 & -4 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 2 \end{bmatrix} u \\ y = [3 \ 0 \ 1 \ 0] \mathbf{x} \end{cases}$$

本部分实验练习题

□ 作业二（简单题）

- 尝试利用MATLAB完成教材第5章习题，5-1、5-2、5-3、5-5、5-6、5-10
- 选做1-2题

谢谢！