

第5回

- コンソールの利用
- 零点・極と時間応答の関係
- 安定性
- 過渡応答の特性

コンソールでの計算

The screenshot shows the Jamox 0.9.5 software interface. The main workspace contains a control block diagram with a transfer function $s^2 + 1.38s + 40$ and a gain of 28.2, connected to a 'jpit' block. The console window at the bottom displays the following commands and results:

```
>lambda = 0.5;
>Tp = 0.5;
>Amax = 0.5;
>zeta = abs(log(lambda))/sqrt(1+4*PI^2)
0.109
>wn = PI/(Tp*sqrt(1-zeta^2))
6.32
>K = Amax*exp(PI*zeta/sqrt(1-zeta^2))
0.706
>
```

変数テーブル

名前	値
Real wn	6.320809271986...
Real Tp	0.5
Real K	0.705511767731...
Real Amax	0.5
Real lambda	0.5
Real zeta	0.108946604114...

実数・複素数の演算

ベクトル・行列の演算

変数の利用

ブロック内での変数参照

The screenshot shows the Jamox 0.9.5 interface. The main workspace contains a block diagram with a transfer function $s^2 + 1.38s + 40$ and a gain of 28.2, connected to a 'jpit' block. A red arrow points from the transfer function to the parameter dialog box. The dialog box is titled 'パラメータ' and has tabs for '基本', 'システム', and 'ポート'. The 'システム' tab is active, showing fields for '分子係数(numerator)' and '分母係数(denominator)'. The numerator field contains $[\$K*\$wn^2]$ and the denominator field contains $[1\ 2*\$zeta*\$wn\ \$wn^2]$. A yellow callout box with a red border contains the text '変数の頭に\$を付ける' (Add \$ to the start of the variable). The console window at the bottom left shows the following commands and outputs:

```
>lambda = 0.5;
>Tp = 0.5;
>Amax = 0.5;
>zeta = abs(log(lam
0.109
>wn = PI/(Tp*sqrt(1
6.32
>K = Amax*exp(PI*ze
0.706
>
```

名前	値
Real wn	6.320809271986...
Real Tp	0.5
Real K	0.705511767731...
Real Amax	0.5
Real lambda	0.5
Real zeta	0.108946604114...

スカラー関数(1/2)

関数名	説明	関数名	説明
abs(x)	絶対値	atanh(x)	逆双曲線正接
acos(x)	逆余弦	ceil(x)	+ ∞ 方向丸め
acosh(x)	逆双曲線余弦	conj(x)	複素共役
arg(x)	位相角	cos(x)	余弦
asin(x)	逆正弦	cosh(x)	双曲線余弦
asinh(x)	逆双曲線正弦	exp(x)	指数関数
atan(x)	逆正接	fact(x)	階乗
atan2(x,y)	逆正接(4象限)	fix(x)	ゼロ方向丸め

スカラー関数(2/2)

関数名	説明	関数名	説明
floor(x)	$-\infty$ 方向丸め	round(x)	最も近い整数
Im(x)	虚部	sgn(x)	符号関数
inv(x)	逆数	sin(x)	正弦
log(x)	自然対数	sinh(x)	双曲線正弦
log10(x)	常用対数	sqrt(x)	平方根
pow(x,y)	べき乗	tan(x)	正接
Re(x)	実部	tanh(x)	双曲線正接
rem(x)	除算の余り		

コンソールに関するTIPS

- コマンドライン編集

左右カーソル(矢印)キー(CTRL+F,CTRL+B)
コマンド上を左右に移動できる

- コマンド履歴参照

上下カーソル(矢印)キー(CTRL+N,CTRL+P)
以前に入力したコマンドを再利用できる

- 変数の値

変数名を入力すると、変数の値が表示される

- 計算結果の非表示

行末にセミコロン「;」を付けると
計算結果を非表示にできる

コンソール利用上の注意

The screenshot shows the Jamox 0.9.5 (2009.4.9) software interface. The title bar reads "Jamox 0.9.5 (2009.4.9) Copyright (C) 2000-2009, mklab.org". The menu bar includes "ファイル(F)", "編集(E)", "ブロック(B)", "シミュレーション(S)", "線形解析(L)", "ウインドウ(W)", and "ヘルプ(H)". The toolbar contains various icons for file operations and simulation. The main window is divided into several panes:

- 名称未設定**: A tabbed pane for variable names, currently empty.
- 変数テーブル**: A table showing variable names and their values.

名前	値
Real wn	6.320809271986...
Real zeta	0.108946604114...
Real a	1
- コンソール**: A console window showing the following text:

```
>a = (2+(1 + 2)/2  
org.mklab.jmatx.ParseException: Encountered " ;"  
>  
>  
b = 3*4  
org.mklab.jmatx.ParseException: Encountered " ="  
>  
>
```

Annotations in the image highlight specific issues in the console:

- 分法エラー**: A yellow box with a red border highlights the error message "org.mklab.jmatx.ParseException: Encountered ';'".
- プロンプト > が無い状態で入力**: A yellow box with a red border highlights the input "b = 3*4" which was entered without a prompt.
- リターンキーを入力すると > が現れる**: A yellow box with a red border highlights the prompt ">" that appears after pressing the return key.

極と零点

$$G(s) = \frac{b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0}$$

$$= \frac{K(s - z_1)(s - z_2) \cdots (s - z_m)}{(s - p_1)(s - p_2) \cdots (s - p_n)} \quad \begin{array}{l} z_i : \text{零点} \\ p_i : \text{極} \end{array}$$

$$= \frac{b_m s^m + b_{m-1} s^{m-1} + \cdots + b_1 s + b_0}{\prod_{i=1}^M (s + \sigma_i) \prod_{j=1}^N ((s + \alpha_j)^2 + \omega_j^2)}$$

$-\sigma_i$: 実極 $-\alpha_j \pm \omega_j$: 複素極

一般システムのステップ応答

$$y(t) = \mathcal{L}^{-1} \left[G(s) \frac{1}{s} \right]$$

$$= \mathcal{L}^{-1} \left[\frac{A_0}{s} + \sum_{i=1}^M \frac{A_i}{s + \sigma_i} + \sum_{j=1}^N \frac{B_j \omega_j}{(s + \alpha_j)^2 + \omega_j^2} \right]$$

$$= A_0 + \sum_{i=1}^M A_i e^{-\sigma_i t} + \sum_{j=1}^N \frac{B_j}{\omega_j} e^{-\alpha_j t} \sin \omega_j t$$

= 各極のインパルス応答の和

線形システムの安定性

任意の有界入力に対して出力が有界

$$|u(t)| < \infty \Rightarrow |y(t)| < \infty \quad \Rightarrow \text{システムは安定}$$

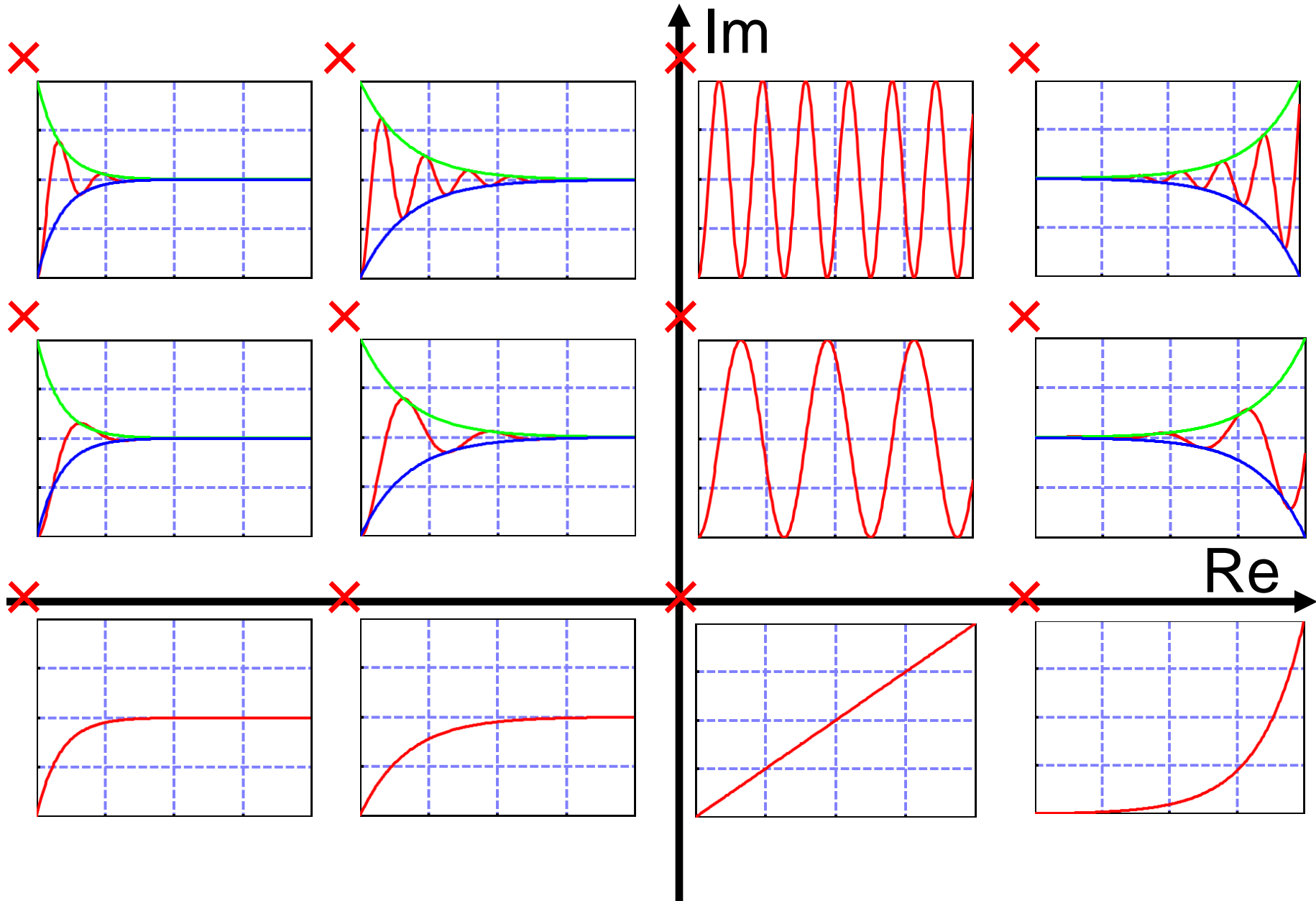
ステップ応答が一定値 ($\neq \infty$) に収束

$$e^{-\sigma_i t} \rightarrow 0, e^{-\alpha_j t} \sin \omega_j t \rightarrow 0 \quad (\text{全ての } i \text{ と } j)$$

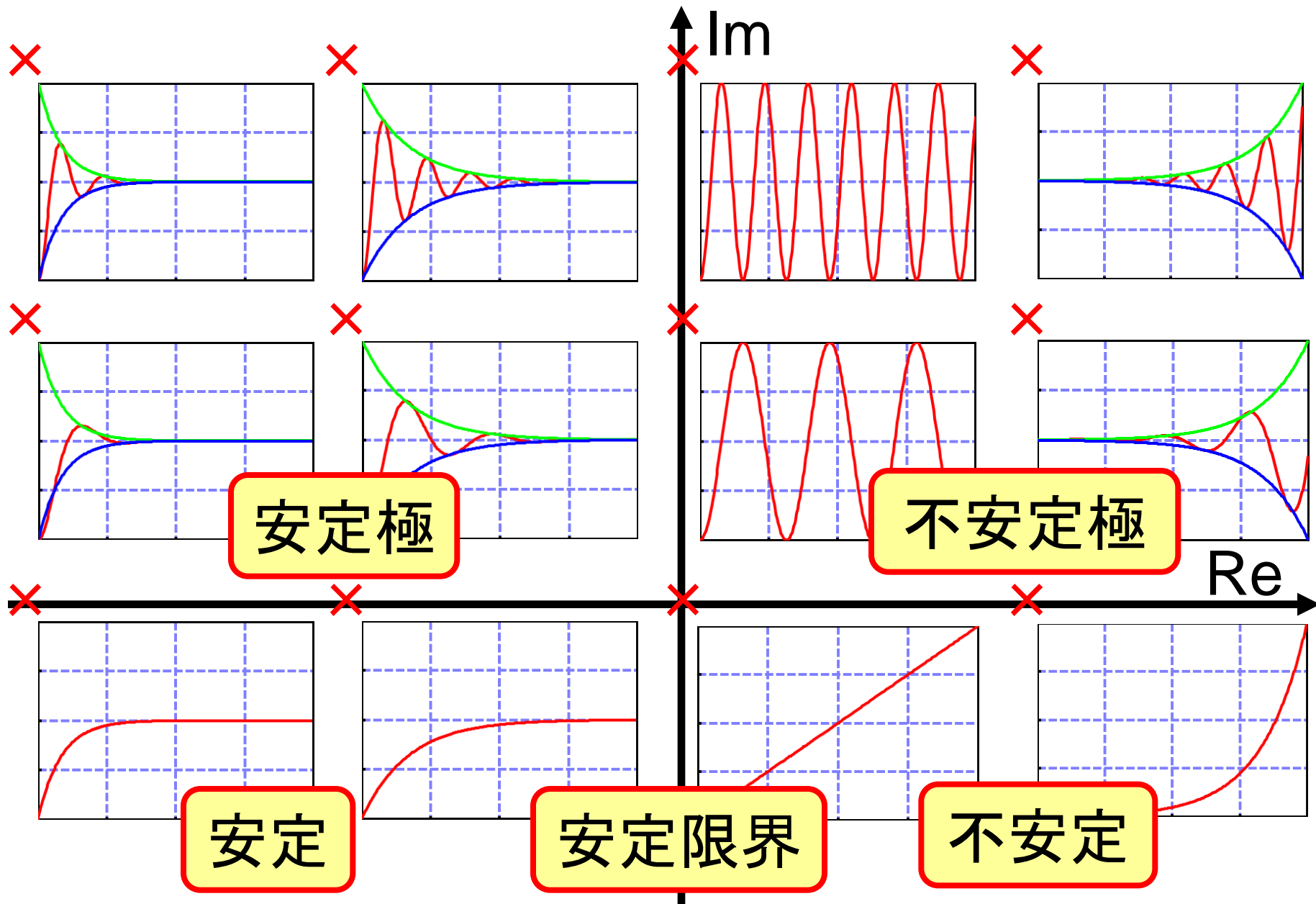
$$-\sigma_i < 0, -\alpha_i < 0 \quad (\text{全ての } i \text{ と } j)$$

全ての極の実部が負

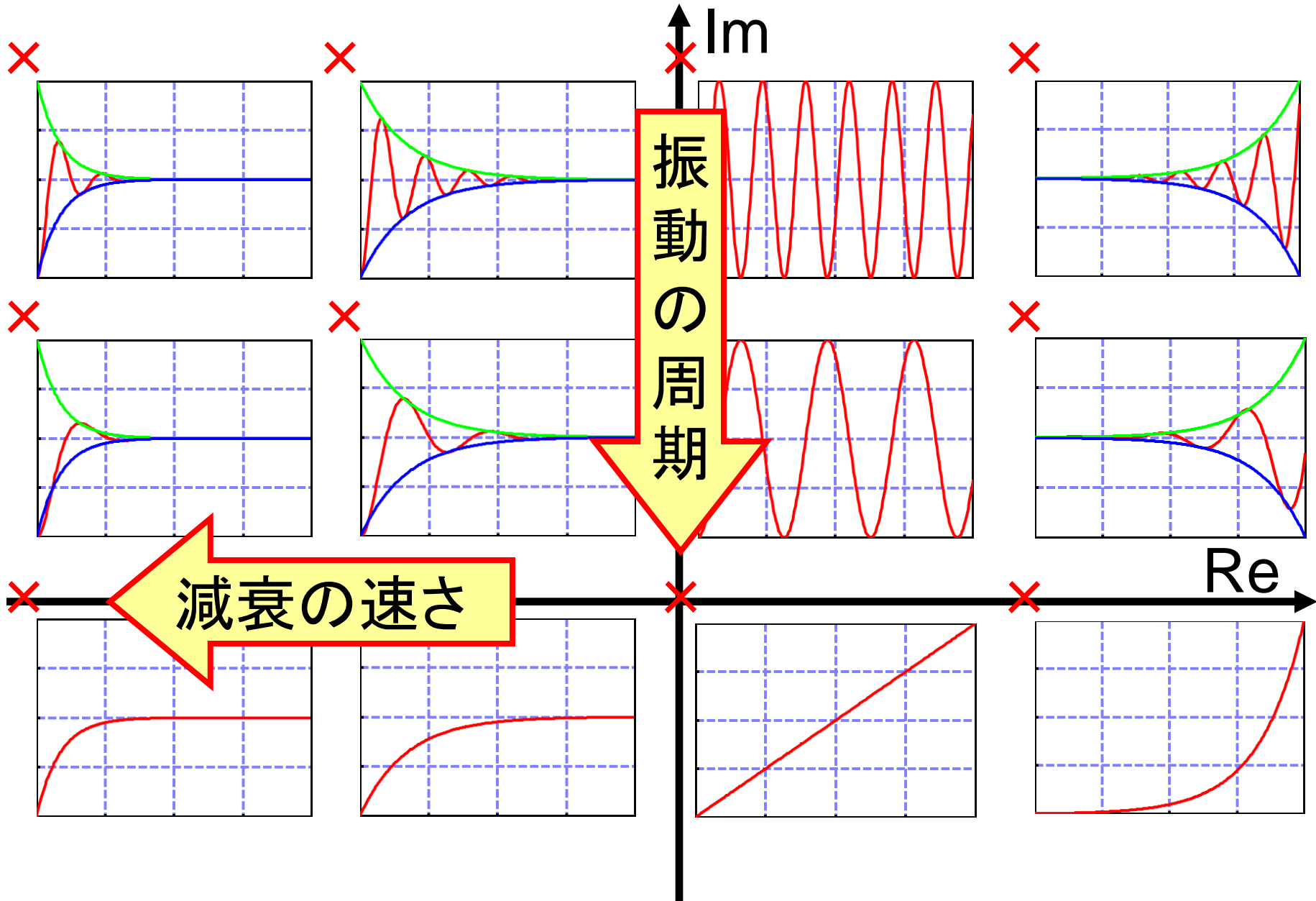
極の位置とステップ応答



極の位置と安定性



極の位置と振動性

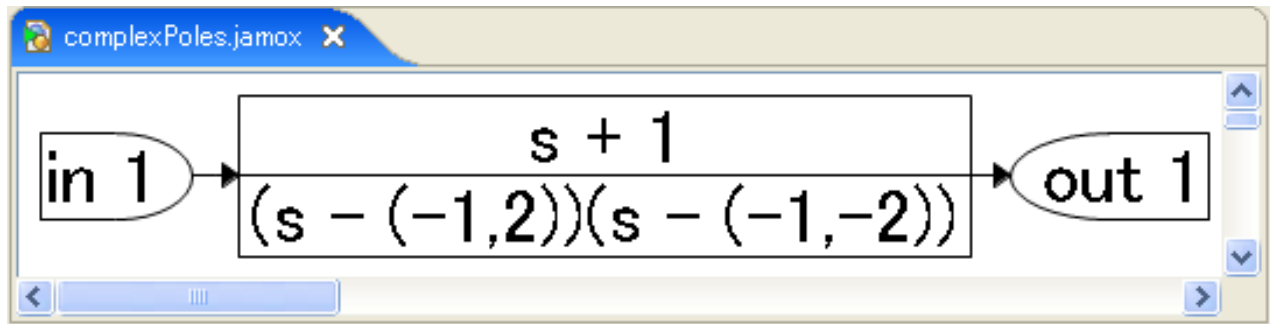


零点・極指定による伝達関数

The image illustrates the process of defining a transfer function in a control system software. It consists of three main components:

- Block Library:** On the left, a library of blocks is shown. The '零点・極表現' (Zero/Pole Representation) block is highlighted with a red box.
- Block Configuration:** The central window shows the transfer function
$$\frac{10(s+1)}{(s-1)(s+2)}$$
 entered into the block. The numerator $10(s+1)$ is enclosed in a green box, and the denominator $(s-1)(s+2)$ is enclosed in a blue box. A red arrow points from the parameter window to the $(s+1)$ term, and a blue arrow points from the parameter window to the denominator terms.
- Parameter Window:** On the right, the 'パラメータ...' (Parameters...) dialog box is open, showing the following settings:
 - 零点(zeros): $[-1]$ (highlighted with a red box)
 - 極(poles): $[1, -2]$ (highlighted with a blue box)
 - ゲイン(gain): $[10]$ (highlighted with a green box)

複素極をもつ伝達関数



コンソール 問題

>伝達関数 :

$$\frac{s + 1}{s^2 + 2s + 5}$$

複素極:

$$p_1 = -1 + 2j$$

$$p_2 = -1 - 2j$$

パラメータ設定:

基本 システム 図 ポート

零点(zeros) [-1]

極(poles) [(-1, 2), (-1, -2)]

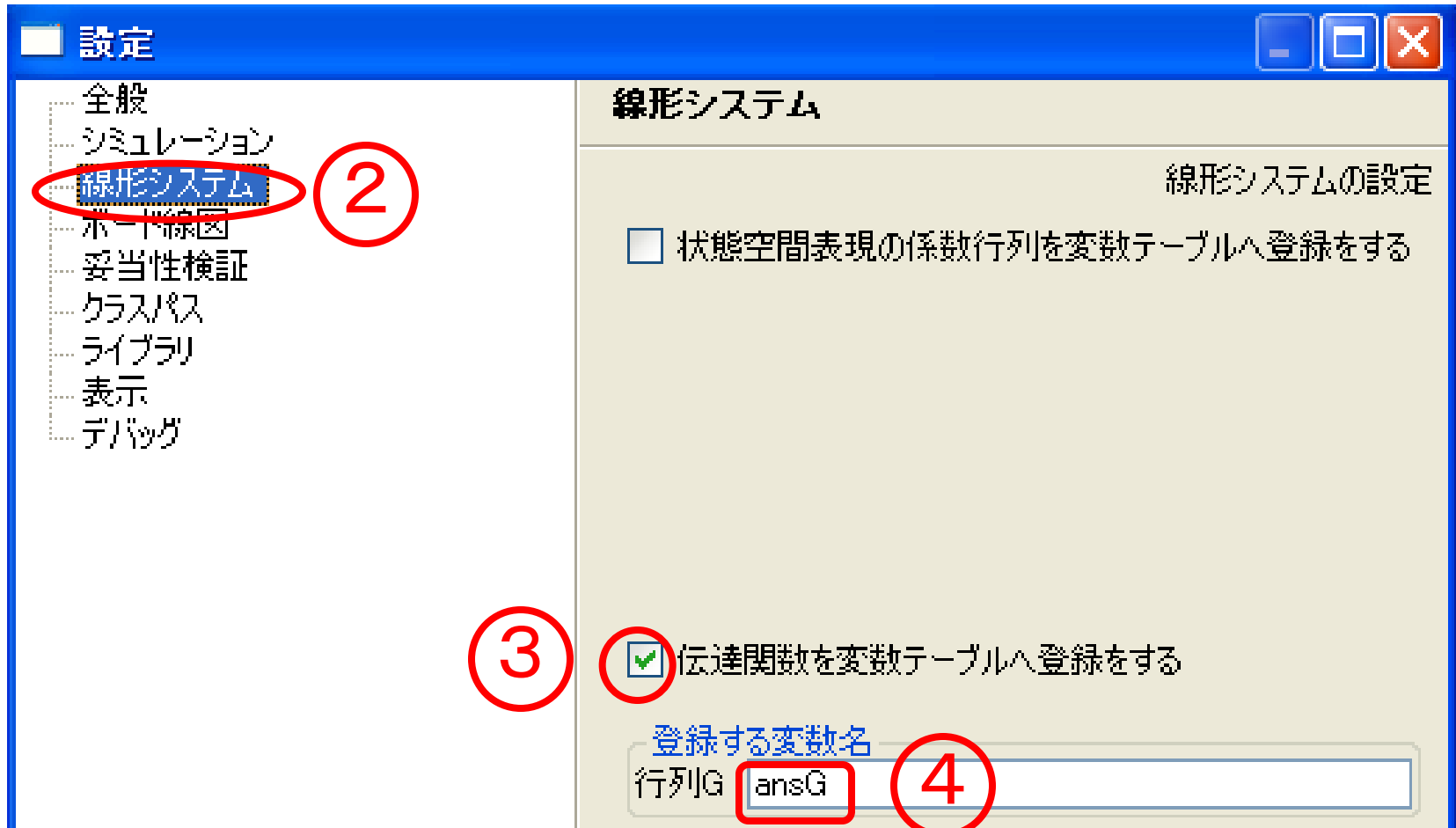
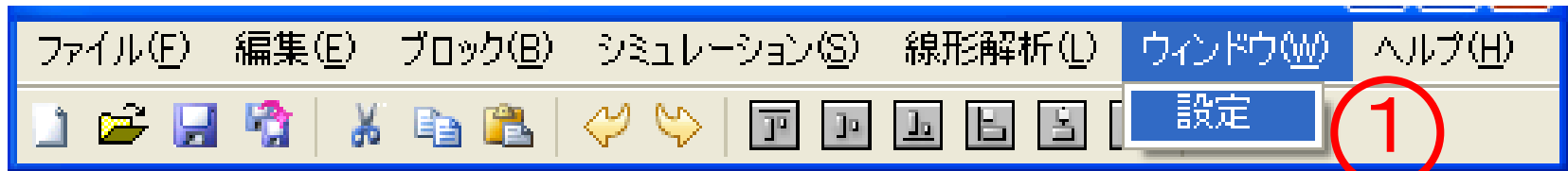
ゲイン(gain) [1]

了解 キャンセル 適用

複素数

The dialog box shows the parameter settings for the transfer function. The poles are set to [(-1, 2), (-1, -2)], which are complex conjugates. A red box labeled "複素数" (Complex number) points to these poles. The zeros are set to [-1] and the gain is set to [1].

伝達関数を変数へ保存する設定



変数に保存した伝達関数の表示

The screenshot shows the Jamox 0.9.5 interface. At the top, the title bar reads "Jamox 0.9.5 (2009.5.27) Copyright (C) 2000-2009. mklab.org". The menu bar includes "ファイル(F)", "編集(E)", "ブロック(B)", "シミュレーション(S)", "線形解析(L)", and "ウインドウ(W)". The toolbar contains various icons for file operations and simulation.

The main workspace displays a block diagram with an input "in 1" connected to a transfer function block $\frac{s-1}{(s+1)(s+2)}$, which is connected to an output "out 1".

On the right, the "変数テーブル" (Variable Table) window shows a table with two columns: "名前" (Name) and "値" (Value). The entry "ansG" is highlighted with a red box, and its value is $(s-1)/(s^2+3s+2)$. A red arrow points from the text "変数に保存される" (Saved to variable) to this entry.

At the bottom, the "コンソール" (Console) window shows the command `>ansG` entered, with "ansG" highlighted by a red box. A red arrow points from the text "変数名を入力" (Enter variable name) to this box. Below the command, the transfer function is displayed as a fraction:
$$\frac{s-1}{s^2+3s+2}$$

A context menu is open over the console, listing several options: "伝達関数(T)", "状態空間表現(数値)(S)", "状態空間表現(数式)(E)", "状態空間表現(キャンセル無し)", "全システムの状態空間表現(数式)(E)", "ボード線図(B)", "ナイキスト線図", "シグマプロット(G)", and "ファイルへ保存". A red arrow points from the text "変数に保存される" to the "ファイルへ保存" option.

伝達関数の極の計算

The screenshot shows a software interface for calculating the poles of a transfer function. At the top, a block diagram shows an input 'in 1' entering a transfer function block $\frac{s-1}{(s+1)(s+2)}$, which outputs 'out 1'. To the right, a '変数テーブル' (Variable Table) lists the transfer function as 'ansG' with the value $(s-1)/(s^2+3s+2)$. Below, a 'コンソール' (Console) window shows the command `>poles(ansG)` and its output: a CoMatrix with two rows. The first row shows the real parts of the poles: (1) and (2) . The second row shows the imaginary parts: -1 and -2 . A red box labeled '極を求める関数' (Function to find poles) points to the `poles` command. Two yellow boxes at the bottom, labeled '実部' (Real part) and '虚部' (Imaginary part), point to the respective columns of the output matrix.

伝達関数: $\frac{s-1}{(s+1)(s+2)}$

変数テーブル:

名前	値
ansG	$(s-1)/(s^2+3s+2)$

コンソール:

```
>poles(ansG)
=== ( 2 x 1) CoMatrix ===
 [ ( 1)-Real ( 1)-Imag ]
 ( 1) -1 0
 ( 2) -2 0
```

極を求める関数

実部

虚部

伝達関数の零点の計算

The screenshot shows a software interface with three main panels:

- Block Diagram:** A block labeled $\frac{s-1}{(s+1)(s+2)}$ with an input **in 1** and an output **out 1**.
- 変数テーブル (Variable Table):** A table with two columns: 名前 (Name) and 値 (Value). It contains one entry: ansG with the value $(s-1)/(s^2+3s+2)$.
- コンソール (Console):** A command prompt window showing the command `>zeros (ansG)` and its output:

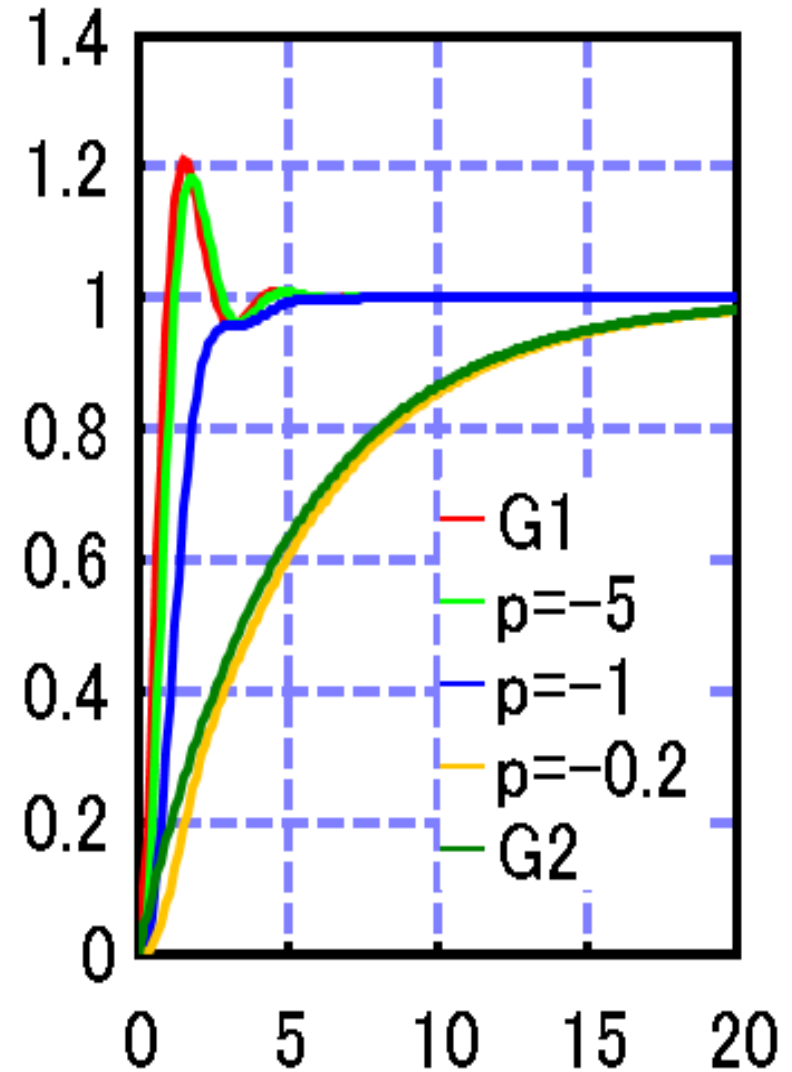
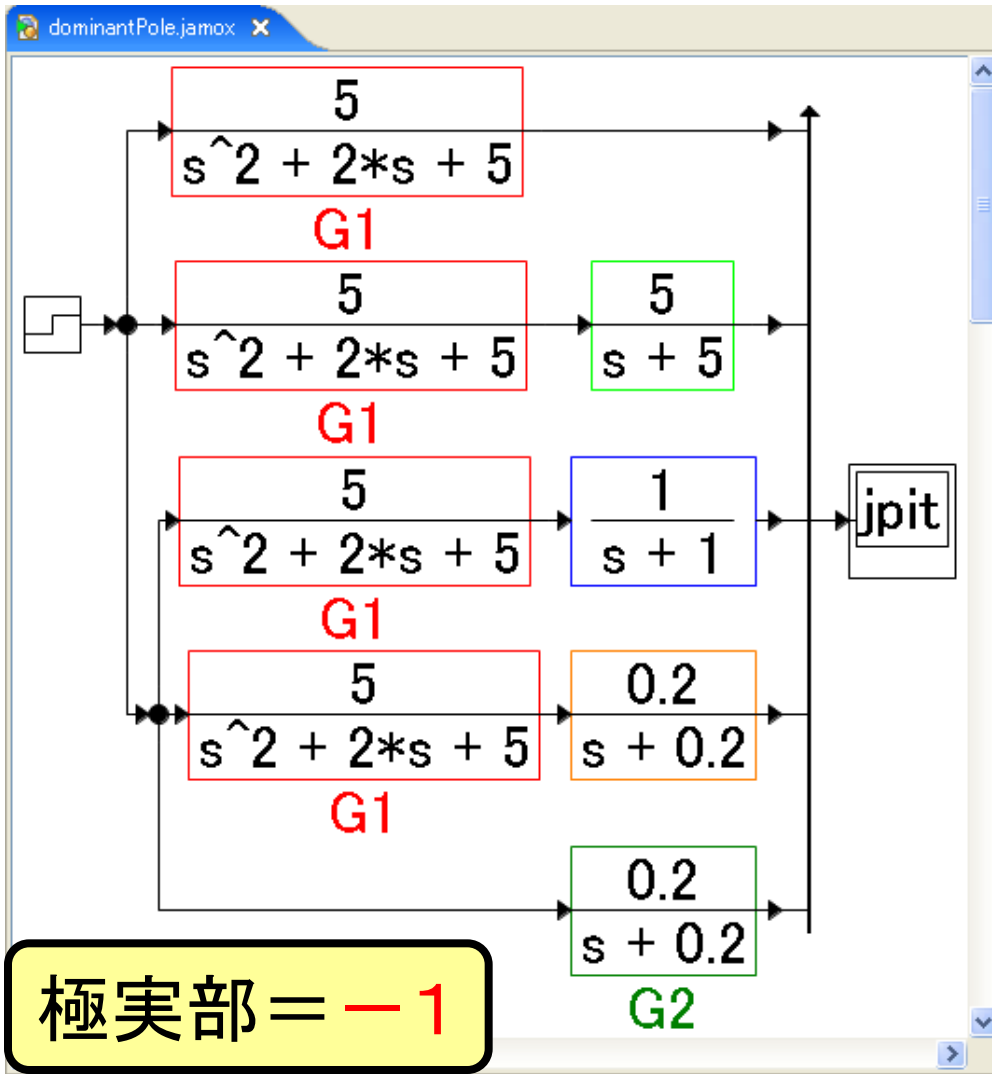
```
==== ( 1 x 1) CoMatrix ====  
[ ( 1)-Real ( 1)-Imag ]  
( 1) 1 0
```

A red box highlights the text "零点を求める関数(新バージョンのみ)" (Zero-finding function (new version only)) with an arrow pointing to the `zeros` command in the console.

Two yellow boxes with red borders are positioned at the bottom of the console output:

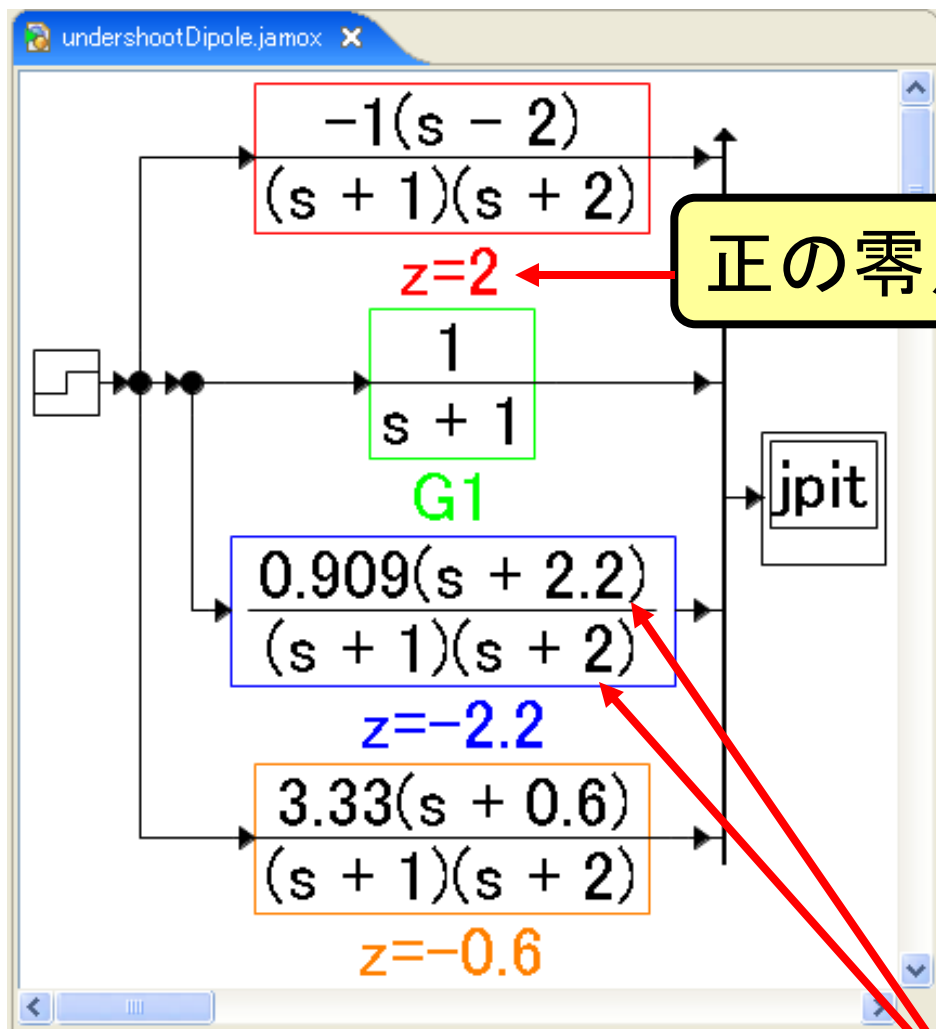
- 実部 (Real part):** A box containing the number 1, corresponding to the real part of the zero.
- 虚部 (Imaginary part):** A box containing the number 0, corresponding to the imaginary part of the zero.

代表極

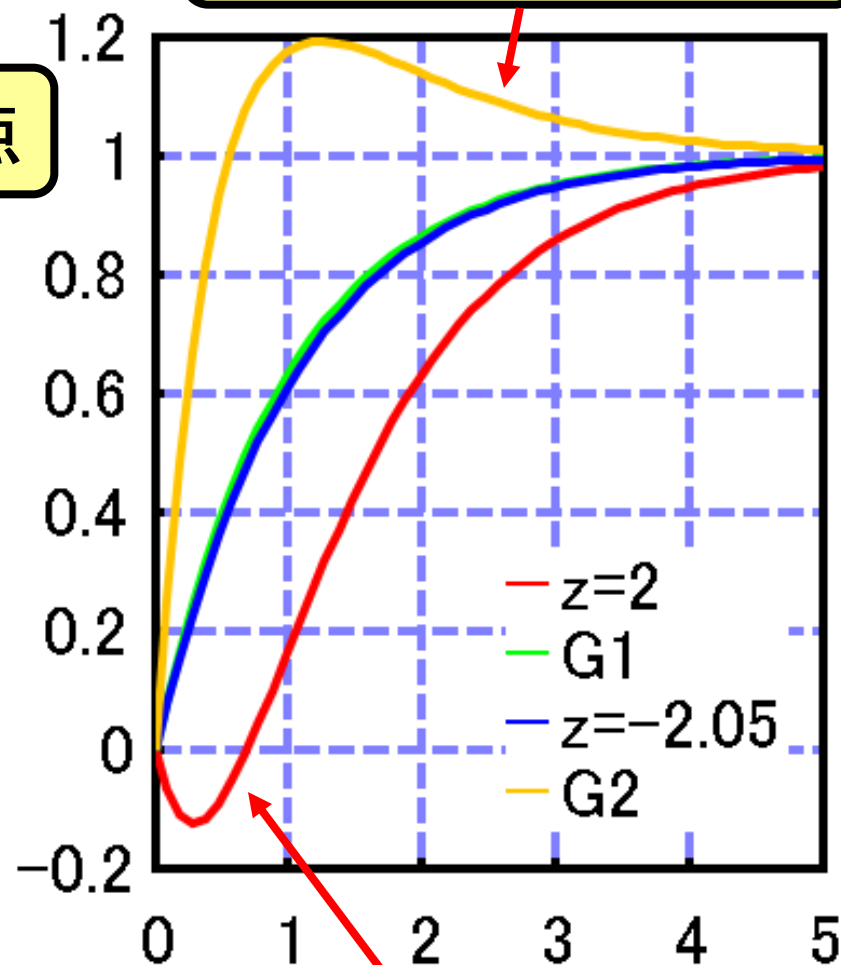


代表極 (虚軸に近い極) が出力に最も影響を与える

零点の存在するシステムの応答



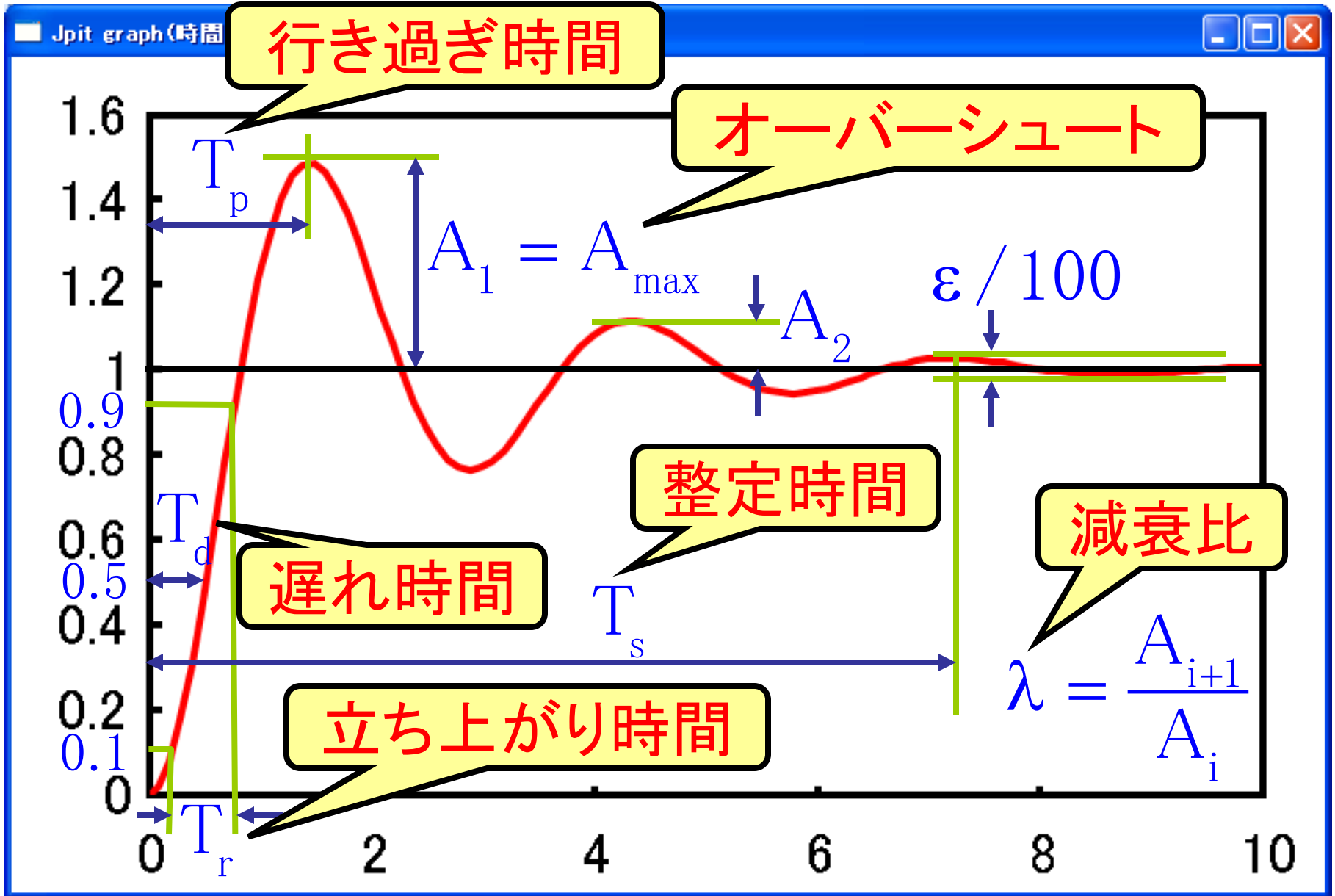
オーバーシュート



ダイポール: 接近した極と零点の組

逆ぶれ

過渡応答の特性値



過渡応答の特性値

- **立ち上がり時間**: T_r
定常値の10%~90%に要する時間
- **遅れ時間**: T_d
定常値の50%に要する時間
- **行き過ぎ時間**: T_p
最初の行き過ぎの時間
- **整定時間**: T_s
定常値の $\pm(1,2,5)\%$ への収束に要する時間
- **オーバーシュート**: A_{max}
最大ピーク値
- **減衰比**: λ
1番目と2番目の行き過ぎ量の比

二次系の過渡特性と係数の関係

- 減衰比

$$\lambda = \exp\left(-\frac{2\pi\zeta}{\sqrt{1-\zeta^2}}\right)$$

$$\zeta = \frac{|\log \lambda|}{\sqrt{1+4\pi^2}}$$

- 行き過ぎ時間

$$T_p = \frac{\pi}{\omega_n \sqrt{1-\zeta^2}}$$

$$\omega_n = \frac{\pi}{T_p \sqrt{1-\zeta^2}}$$

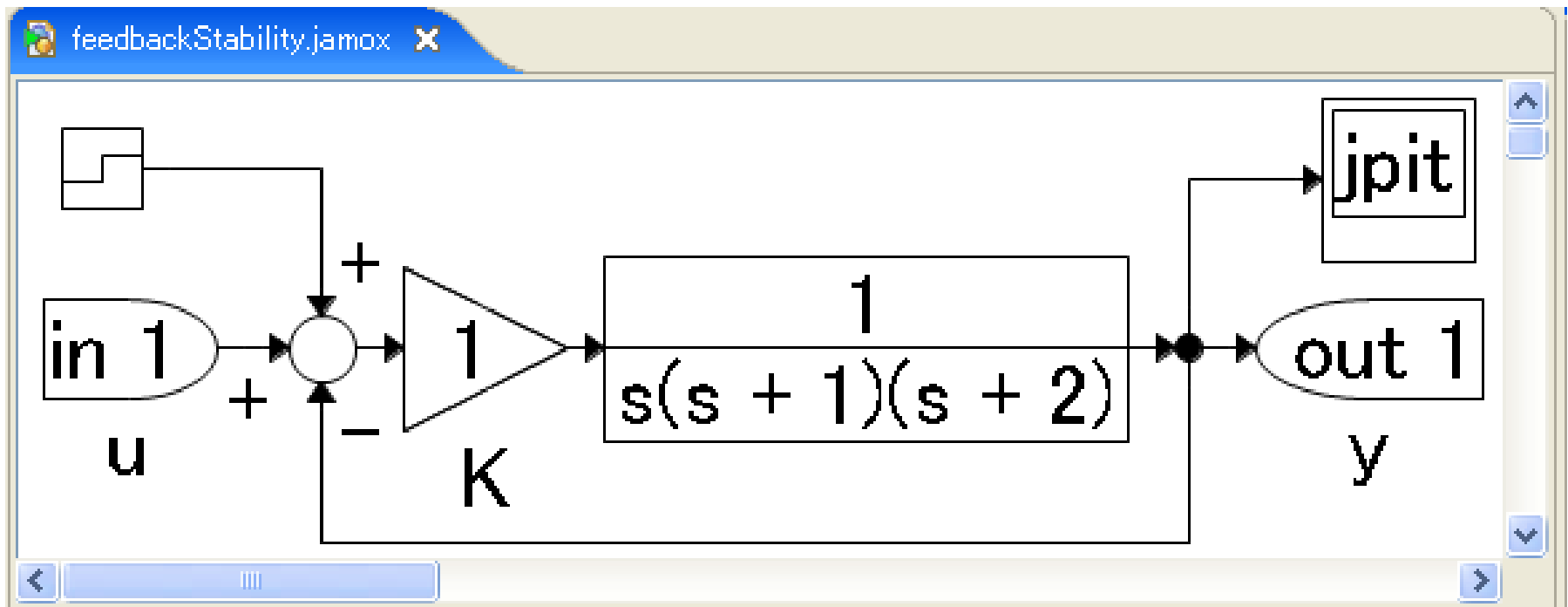
- オーバーシュート

$$A_{\max} = K \cdot \exp\left(-\frac{\pi\zeta}{\sqrt{1-\zeta^2}}\right)$$

$$K = A_{\max} \exp\left(\frac{\pi\zeta}{\sqrt{1-\zeta^2}}\right)$$

演習1: フィードバック系の安定性

- $K=1$ 、 $K=10$ のとき、極を求め安定性を判別せよ。また、そのときのステップ応答を求めよ。
- K の値を変更しながら極を求めることで、フィードバック系が安定である $K > 0$ の範囲を求めよ。



演習2: 応答が似ているシステム

- 伝達関数が以下で与えられるシステムについて、ステップ応答が似ている組を求めよ。また、応答が似ている理由を述べよ。

$$G_1(s) = \frac{1}{s+1}$$

$$G_2(s) = \frac{10}{s+10}$$

$$G_3(s) = \frac{10}{(s+1)(s+10)}$$

$$G_4(s) = \frac{(10/0.95)(s+0.95)}{(s+1)(s+10)}$$

演習3: 過渡特性を満たす二次系

減衰比 $\lambda = 0.2$ 、行き過ぎ時間 $T_p = 2.0$ 、
オーバーシュート $A_{\max} = 0.45$ である二次系を
求め、ステップ応答を得よ。

