Staircase Transformations & Controllability/Observability

Input decoupling produced the following form:

$$\underline{\dot{x}} = \left[\begin{array}{c|c} A_c & A_{12} \\ \hline \phi & A_{\bar{c}} \end{array}\right] \underline{x} + \left[\begin{array}{c} B_c \\ \hline \phi \end{array}\right] \underline{u}$$

$$\underline{y} = \left[\begin{array}{c|c} C_c & C_{\bar{c}} \end{array}\right] \underline{x} + \left[D \right] \underline{u}$$

This could then be reduced to:

$$\underline{\dot{x}}_r = \left[A_c \right] \underline{x}_r + \left[B_c \right] \underline{u}$$

$$\underline{y} = \left[C_c \right] \underline{x}_r + \left[D \right] \underline{u}$$

as long as:

$$\max(\text{real}(\text{eig}(A_{\bar{c}}))) < \phi .$$

What was problematic with input decoupling was the use of Q_c and Q_c^{-1} to accomplish the similarity transformation.

Question: Does there exist a unitary transformation that will transform the system to a form similar to the desired one above?

The answer is yes!

Algorithm:

Given: $\dot{\underline{x}} = A\underline{x} + B\underline{u}$

We start with:

$$[U, \Sigma, V] = \text{svd}(B)$$

$$\begin{array}{cc} & m \\ n & \begin{bmatrix} B \end{bmatrix} \end{array} = \begin{bmatrix} U \end{bmatrix} \cdot \begin{bmatrix} \Sigma \end{bmatrix} \cdot \begin{bmatrix} V^* \end{bmatrix}$$

$$\Rightarrow T = U^*$$

$$\Rightarrow \hat{B} = T \cdot B = U^* \cdot B \equiv \Sigma \cdot V^* = \begin{bmatrix} \hat{B} \\ \hline \emptyset \end{bmatrix}$$

If we use:

$$T = U^*(n:-1:1,:) \qquad \text{(reverse sequence of rows)}$$

$$\Rightarrow \hat{B} = \begin{bmatrix} \emptyset \\ \hline \hat{B} \end{bmatrix}$$

$$\Rightarrow \dot{\underline{\xi}} = \bar{A} \cdot \underline{\xi} + \bar{B} \cdot \underline{u}$$

$$\bar{A} = \begin{array}{c} \\ n-m \\ m \end{array} \!\!\begin{array}{c} \begin{array}{cc} n-m & m \end{array} \\ \left[\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right] \end{array} \qquad \bar{B} = n \begin{array}{c} m \\ \left[\begin{array}{c} \emptyset \\ \hline \hat{B} \end{array}\right] \begin{array}{l} \}\, n-m \\ \}\, m \end{array} \end{array}$$

Let us now look at the different system:

$$\underline{\dot{\nu}} = A_{11} \cdot \underline{\nu} + A_{12} \cdot \underline{u}$$

Clearly, we can apply the same algorithm:

$$[U, \Sigma, V] = \mathrm{svd}(A_{12})$$

$$\tilde{T}_2 = U^*(n-m:-1:1, :)$$

$$\Rightarrow \underline{\dot{\mu}} = \bar{A}_{11} \underline{\mu} + \bar{A}_{12} \cdot \underline{\underline{u}}$$

where: $\bar{A}_{12} = \begin{array}{c} \begin{bmatrix} \phi \\ \hline \hat{A}_{12} \end{bmatrix} \end{array} \begin{array}{l} n-2m \\ m \end{array}$

Let us apply the following transformation:

$$T_2 = \left[\begin{array}{c|c} \tilde{T}_2 & \phi \\ \hline \phi & I \end{array}\right]$$

to the $\underline{\xi}$ - system.

$$\underline{y} = T_2 \cdot \underline{\xi}$$

$$\Rightarrow \quad \dot{\underline{y}} = \bar{\bar{A}} \cdot \underline{y} + \bar{\bar{B}} \cdot \underline{u}$$

$$\bar{\bar{A}} = T_2 \cdot \bar{A} \cdot T_2^{-1} = T_2 \cdot \bar{A} \cdot T_2^{*}$$

$$\bar{\bar{B}} = T_2 \cdot \bar{B}$$

$$\begin{bmatrix} \tilde{T}_2 & \emptyset \\ \emptyset & I \end{bmatrix} \cdot \begin{bmatrix} \emptyset \\ \hat{B} \end{bmatrix} \equiv \begin{bmatrix} \emptyset \\ \hat{B} \end{bmatrix}$$

$$\Rightarrow \quad \underline{\underline{\bar{\bar{B}} \equiv \bar{B}}}$$

$$T_2 \cdot \bar{A} = \begin{bmatrix} \tilde{T}_2 & \emptyset \\ \emptyset & I \end{bmatrix} \cdot \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} \tilde{T}_2 \cdot A_{11} & \tilde{T}_2 \cdot A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

However:

$$\tilde{T}_2 \cdot A_{12} = \begin{bmatrix} \emptyset \\ \hat{A}_{12} \end{bmatrix}$$

$$\Rightarrow T_2\cdot\bar{A} = \left[\begin{array}{c|c} \hat{A}_{11} & \emptyset \\ \cline{2-2} & \hat{A}_{12} \\ \hline A_{21} & A_{22} \end{array}\right]$$

$$\Rightarrow \bar{\bar{A}} = (T_2\bar{A})\cdot T_2^{*} = \left[\begin{array}{c|c} & \emptyset \\ \cline{2-2} & \\ \hline & \end{array}\right] \cdot \left[\begin{array}{c|c} \tilde{T}_2^{*} & \emptyset \\ \hline \emptyset & I \end{array}\right]$$

$\Rightarrow$ preserves the zeros in the right upper corner.

Now:

$$\left[\begin{array}{c:c|c} \bar{A}_{11} & \bar{A}_{12} & \emptyset \\ \hdashline & & \\ \hline & & \end{array}\right]$$

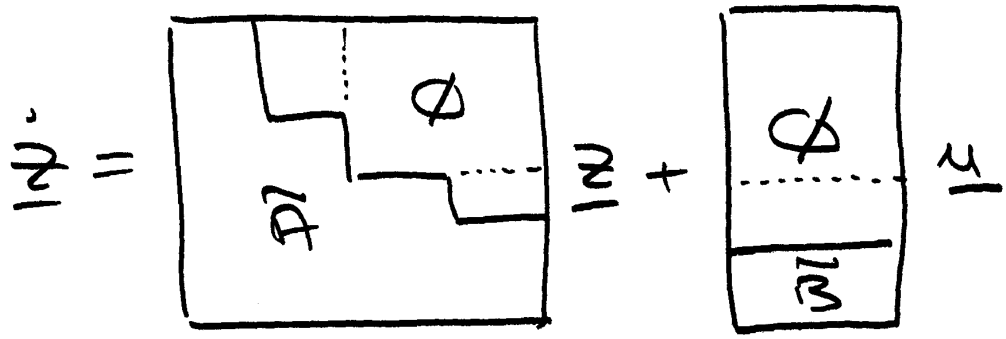
$$\dot{\underline{\gamma}} = \bar{A}_{11}\cdot\underline{\gamma} + \bar{A}_{12}\cdot\underline{u}$$

same algorithm:

$$\bar{\bar{\bar{A}}} = \left[\begin{array}{c|c|c} & \emptyset & \emptyset \\ \cline{2-2} & & \\ \hline & & \\ \hline & & \end{array}\right]$$

etc.

We end up with A in staircase form and B compressed to the bottom:



$$\Rightarrow \quad \underline{\dot{z}} = \left[\begin{array}{c|c} A_{\bar{c}} & \emptyset \\ \hline A_{21} & A_c \end{array}\right] \underline{z} + \left[\begin{array}{c} \emptyset \\ \hline B_c \end{array}\right] \underline{u}$$

has the desired form. It can be reduced to:

$$\underline{\dot{z}}_r = \left[A_c \right] \underline{z}_r + \left[B_c \right] \underline{u}$$

In Matlab:

$$[A_s, B_s, C_s, U, K] = \mathrm{ctrbf}(A, B, C)$$

such that:

$$\left.\begin{aligned} A_s &= U \cdot A \cdot U^* \\ B_s &= U \cdot B \\ C_s &= B \cdot U^* \end{aligned}\right\} \text{unitary transformation}$$

K is a row-vector that provides the breadth of the steps, i.e.

$$n_c = \mathrm{sum}(K)$$

$$\Rightarrow n_{\bar{c}} = n - n_c = n - \mathrm{sum}(K)$$

$$\Rightarrow \begin{aligned} A_{red} &= A_s(n_{\bar{c}}+1:n,\ n_{\bar{c}}+1:n) \\ B_{red} &= B_s(n_{\bar{c}}+1:n,\ :) \\ C_{red} &= C_s(:,\ n_{\bar{c}}+1:n) \end{aligned}$$

is the reduced-order system.

By duality:

$$[A_s, B_s, C_s, U, K] = \text{obsvf}(A, B, C)$$

is implemented as:

```
[Ah, Bh, Ch, U, K] = ctrbf(A', C', B');
As = Ah';
Bs = Ch';
Cs = Bh';
```

MINREAL: performs input decoupling followed by output decoupling using ctrbf and then obsvf.

⟹ All unitary transformations.

⟹ Works fine!