Application to Linear Systems:

Given the system:

$$\left| \begin{array}{l} \underline{\dot{x}} = A\underline{x} + B\underline{u} \\ \underline{y} = C\underline{x} + D\underline{u} \end{array} \right| \qquad \text{(MIMO)}$$

with state- and inputweighing:

$$PI = \underline{x}'(t_f)\, S\, \underline{x}(t_f) + \int_0^{t_f} \{ \underline{x}'(t) \cdot Q\, \underline{x}(t) + \underline{u}'(t)\, R\, \underline{u}(t) \}\, dt \overset{!}{=} \underset{\underline{u}(t)}{\mathrm{Min}}$$

where: $S \geq \emptyset$; $Q \geq \emptyset$; $R > \emptyset$

Assume: t_f = fixed ; $\underline{x}(t_f)$ = variable ;
$u(t)$ = unlimited

Q :: positive semidefinite (symmetric) state-weighting matrix

R :: positive definite (symmetric) input-weighting matrix

S :: positive semidefinite final-value-weighting matrix

We can build the Hamiltonian:

$$H(\underline{x}, \underline{u}, \underline{\psi}, t) = \underline{x}'Q\underline{x} + \underline{u}'R\underline{u} + \underline{\psi}'(A\underline{x} + B\underline{u})$$

$$= \underline{x}'Q\underline{x} + \underline{\psi}'A\underline{x} + \underline{\psi}'B\underline{u} + \underline{u}'R\underline{u}$$

$$\Rightarrow \dot{\underline{\psi}} = -\frac{\partial H}{\partial \underline{x}} = -2Q\underline{x} - A'\underline{\psi}$$

where: $\underline{\psi}(t_f) = 2S\underline{x}(t_f)$

$$\frac{\partial H}{\partial \underline{u}} = \phi = 2R\underline{u}_{opt} + B'\underline{\psi}$$

$$\Rightarrow \boxed{\underline{u}_{opt}(t) = -\tfrac{1}{2}R^{-1}B'\underline{\psi}}$$

$$\dot{\underline{x}} = A\underline{x} + B\underline{u}_{opt} = A\underline{x} - \tfrac{1}{2}BR^{-1}B'\underline{\psi}$$

$\Rightarrow$ We must solve the following boundary value problem:

$$\begin{bmatrix} \dot{\underline{x}} \\ \dot{\underline{\psi}} \end{bmatrix} = \begin{bmatrix} A & -\frac{1}{2}BR^{-1}B' \\ -2Q & -A' \end{bmatrix} \cdot \begin{bmatrix} \underline{x} \\ \underline{\psi} \end{bmatrix}$$

$$\left| \begin{array}{l} \underline{x}(\phi) = \underline{x}_0 \\ \underline{\psi}(t_f) = 2S\underline{x}(t_f) \end{array} \right|$$

We can simplify this a little bit by making:

$$\tilde{\underline{\psi}}(t) = \frac{1}{2}\cdot\underline{\psi}(t)$$

$$\Longleftrightarrow \quad \underline{\psi}(t) = 2\cdot\tilde{\underline{\psi}}(t)$$

$$\Longrightarrow \quad \dot{\underline{\psi}}(t) = 2\cdot\dot{\tilde{\underline{\psi}}}(t)$$

$$\Rightarrow \begin{bmatrix} \dot{\underline{x}} \\ \dot{\tilde{\underline{\psi}}} \end{bmatrix} = \begin{bmatrix} A & -BR^{-1}B' \\ -Q & -A' \end{bmatrix}\cdot\begin{bmatrix} \underline{x} \\ \tilde{\underline{\psi}} \end{bmatrix}$$

$$\left| \begin{array}{l} \underline{x}(0) = \underline{x}_0 \\ \tilde{\underline{\psi}}(t_f) = S\,\underline{x}(t_f) \end{array} \right|$$

This $(2n\times 2n)$ "system" matrix is called the <u>Hamiltonian matrix</u>.

Solution:

We write for $\tilde{\underline{\psi}}(t)$:

$$\boxed{\tilde{\underline{\psi}}(t) = P(t)\cdot\underline{x}(t)}$$

As nothing was said about

$P(t)$, this is obviously possible.

$$\Rightarrow \tilde{\underline{\psi}}(t_f) = P(t_f)\cdot \underline{x}(t_f) \equiv S\,\underline{x}(t_f)$$

$$\Rightarrow \boxed{P(t_f) \equiv S}$$

$$\dot{\underline{x}} = A\underline{x} - BR^{-1}B'\tilde{\underline{\psi}}$$

$$= A\underline{x} - BR^{-1}B'P\underline{x}$$

$$\Rightarrow \dot{\underline{x}} = \left[A - BR^{-1}B'P\right]\underline{x}$$

$$\dot{\tilde{\underline{\psi}}} = -Q\underline{x} - A'\tilde{\underline{\psi}}$$

$$= -Q\underline{x} - A'P\underline{x}$$

$$\Rightarrow \dot{\tilde{\underline{\psi}}} = -\left[Q + A'P\right]\underline{x}$$

but: $\tilde{\underline{\psi}} = P\underline{x}$

$$\Rightarrow \dot{\tilde{\underline{\psi}}} = (P\underline{x})^{\cdot} = \dot{P}\underline{x} + P\dot{\underline{x}}$$

$$-\left[Q + A'P\right]\underline{x} = \dot{P}\underline{x} + P\left[A - BR^{-1}B'P\right]\underline{x}$$

$$\Rightarrow \boxed{-\dot{P} = PA + A'P + Q - PBR^{-1}B'P}$$

This is called the <u>Riccati Differential Equation</u>.

$\Rightarrow$ We have decomposed the (2n) - boundary value problem into two initial value problems, one of size (n), the other of size (n×n).

<u>Recipe</u>: Integrate the Riccati equation <u>backward</u> in time from $t_f \longrightarrow \emptyset$:

$$\boxed{\dot{P} = PBR^{-1}B'P - PA - A'P - Q}$$

$$P(t_f) = S$$

$\Rightarrow P(t)$

Then plug the previously found $P(t)$ into the system equations and integrate them <u>forward</u> in time from $\emptyset \longrightarrow t_f$:

$$\left| \; \dot{\underline{x}} = \left(A - BR^{-1}B'P(t)\right)\underline{x} \; \right|$$

$$\underline{x}(\phi) = \underline{x}_0$$

$\Longrightarrow \underline{x}(t)$

Notice:

$$\underline{u}_{opt}(t) = -\tfrac{1}{2}R^{-1}B'\underline{\psi}(t)$$

$$= -R^{-1}B'\tilde{\underline{\psi}}(t)$$

$$\Rightarrow \quad \underline{u}_{opt}(t) = -\underbrace{R^{-1}B'P(t)}_{K(t)} \cdot \underline{x}(t)$$

$$\Rightarrow \quad \boxed{\underline{u}_{opt}(t) = \underline{r}(t) - K(t) \cdot \underline{x}(t)}$$

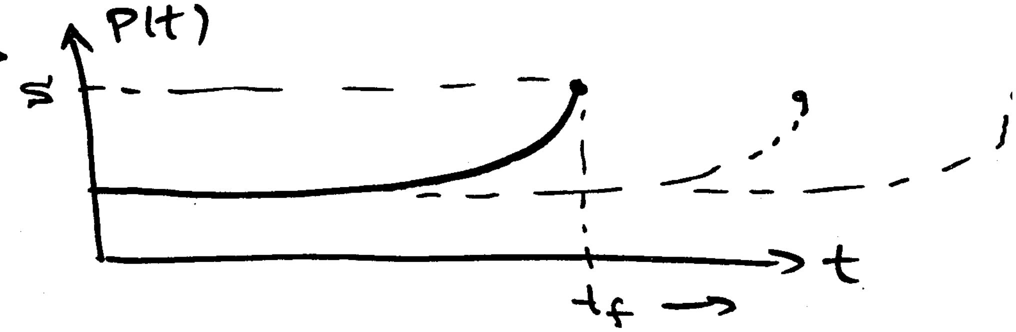
where: $\boxed{K(t) = R^{-1}B'P(t)}$

The optimal solution can be realized as a time-variant state-feedback.

Simplification:

$$\boxed{t_f \longrightarrow \infty}$$

$\Longrightarrow$



$\Rightarrow$ P(t) becomes constant.

$\Rightarrow \dot{P} = \emptyset$

$$\Rightarrow \boxed{PA + A'P + Q - PBR^{-1}B'P \equiv \emptyset}$$

$\Rightarrow$ Algebraic Matrix-Riccati Equation.

$$\Longrightarrow \boxed{K = R^{-1}B'P}$$

$$\rightarrow \boxed{\underline{u}_{opt}(t) = \underline{r}(t) - K \cdot \underline{x}(t)}$$

is a state-feedback.

- The "algebraic Matrix-Riccati Equation" is actually a set of n^2 nonlinear equations.

- This set of equations has <u>many</u> solutions, but only <u>one</u> leads to a stable feedback system.

Algorithm: (without proof)

(1) Check controllability. If not completely controllable, input-decouple the uncontrollable modes first.

(2) Compute the Hamiltonian matrix:

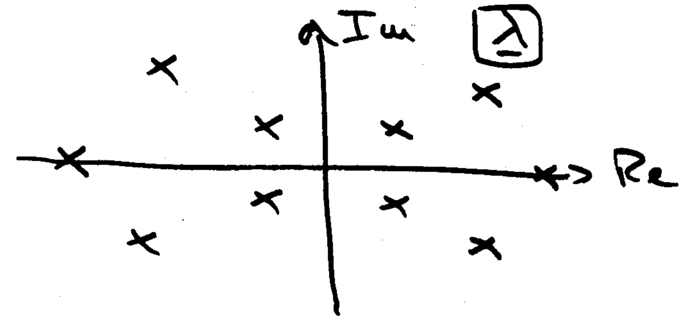
$$H = \begin{bmatrix} A & -BR^{-1}B' \\ -Q & -A' \end{bmatrix}$$

(3) Compute the spectral decomposition of H:

$$[V, \Lambda] = \mathrm{Eig}(H)$$

The eigenvalues of the Hamiltonian are not only symmetric to the real axis, but also to the imaginary axis.

If the system is controllable $\Rightarrow$ H has no eigenvalues on the imaginary axis. E.g.:

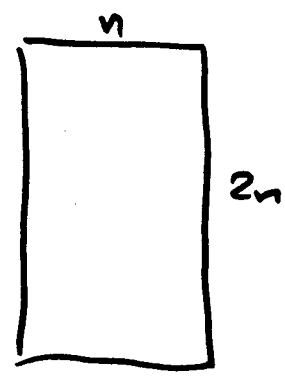


might be a possible set of eigenvalues of the Hamiltonian.

$\Rightarrow$ There are exactly n eigenvalues with negative real part.

(4) Take the subset of the right modal matrix that relates to eigenvalues with negative real part:

$\hat{V} =$



(5) Cut $\hat{V}$ into an upper and a lower portion:

$$\hat{V} = \begin{bmatrix} V_1 \\ V_2 \end{bmatrix} = \begin{array}{c c} n & \\ \left[\begin{array}{c} V_1 \\ \hline V_2 \end{array}\right] & \begin{array}{c} n \\ n \end{array} \end{array}$$

⑥ Compute the solution to the algebraic Riccati equation:

$$P = V_2 \cdot V_1^{-1}$$

(Notice: Any other n eigenvectors would have given us another solution, but this is the one and only solution that leads to a stable closed-loop system.)

⑦ $K = R^{-1} B' P$

is the desired state feedback.

This algorithm is numerically very benign (much better than pole placement).

In Matlab:

```
H = [A, -(B/R)*B'; -Q, -A'];
[V,L] = eig(H);
k=0;
for i=1:2*n,
    if real(L(i,i)) < 0 then
        k=k+1;
        V(:,k) = V(:,i);
    end
end
V1 = V(1:n, 1:n);
V2 = V(n+1:2*n, 1:n);
P = V2/V1;
K = (R\(B'))*P;
```

This algorithm also exists as a preprogrammed Matlab function:

K = lqr(A,B,Q,R)

↑ linear quadratic (Gaussian) regulator

Disadvantages: LQG has a tendency of placing poles too close to each other ⟹ large sensitivity to parameter changes.

Solution: There meanwhile exist techniques to individually influence pole locations in the Riccati design ⟹ mixture between Riccati & pole placement ⟹ probably better than any of the two alone

⟹ Hal Tharp

- LQG and PLACE can both be used for state-feedback design. ⟹ We can obtain output feedback e.g. by solving two LQG-problems one for the controller, and one for the observer:

```
[> k = LQR(A, B, Qc, Rc);
[> h = LQR(A', C', Qo, Ro);
[> h = h';
```

<u>Output Weighting:</u>

Sometimes, it is desirable to weight the outputs instead of the states:

$$\left| \begin{aligned} \dot{\underline{x}} &= A\underline{x} + B\underline{u} \\ \underline{y} &= C\underline{x} \end{aligned} \right| \qquad (\underline{\underline{D = \emptyset}})$$

$$PI = \int_0^\infty \{ \underline{y}' Q \underline{y} + \underline{u}' R \underline{u} \} \, dt \stackrel{!}{=} \underset{\underline{u}(t)}{\text{Min}}$$

$$Q \geq \emptyset \quad ; \quad R > \emptyset$$

$$\Rightarrow \underline{y}' Q \underline{y} = (C\underline{x})' Q (C\underline{x})$$

$$= \underline{x}' \underbrace{C' Q C}_{Q_n} \underline{x} \quad ; \quad Q_n \geq \emptyset$$

$$\Rightarrow PI = \int_0^\infty \{ \underline{x}' Q_n \underline{x} + \underline{u}' R \underline{u} \} \, dt \stackrel{!}{=} \underset{\underline{u}(t)}{\text{Min}}$$

is an equivalent problem with state weighting.

- The case $\underline{D \neq \emptyset}$ works also, but is a little more tricky.

$$\underline{y}'Q\underline{y} = (C\underline{x}+D\underline{u})'Q(C\underline{x}+D\underline{u})$$

$$= (\underline{x}'C'+\underline{u}'D')Q(C\underline{x}+D\underline{u})$$

$$= \underline{x}'C'QC\underline{x} + \underline{x}'C'QD\underline{u} + \underline{u}'D'QC\underline{x} + \underline{u}'D'QD\underline{u}$$

Let: $\left| \begin{array}{l} \hat{Q} = C'QC \\ \hat{R} = R + D'QD \\ \hat{N} = C'QD \end{array} \right|$

$$\Rightarrow PI = \int_0^\infty \{ \underline{x}'\hat{Q}\underline{x} + \underline{x}'\hat{N}\underline{u} + \underline{u}'\hat{N}'\underline{x} + \underline{u}'\hat{R}\underline{u} \}$$

or: $$PI = \int_0^\infty [\underline{x}' \;\; \underline{u}'] \cdot \begin{bmatrix} \hat{Q} & \hat{N} \\ \hat{N}' & \hat{R} \end{bmatrix} \cdot \begin{bmatrix} \underline{x} \\ \underline{u} \end{bmatrix} dt \stackrel{!}{=} \underset{\underline{u}(t)}{\mathrm{Min}}$$

Another variable transformation can remove the mixed terms:

$$PI = \int_0^\infty \left\{ \underline{x}'(\hat{Q} - \hat{N}\hat{R}^{-1}\hat{N}')\underline{x} + (\underline{u} + \hat{R}^{-1}\hat{N}'\underline{x})' \cdot \right.$$
$$\left. \cdot \hat{R}(\underline{u} + \hat{R}^{-1}\hat{N}'\underline{x}) \right\} dt \stackrel{!}{=} \underset{\underline{u}}{\mathrm{Min}}$$

(Can be verified easily by multiplying out.)

Let:
$$\left| \begin{array}{l} Q_n = \hat{Q} - \hat{N}\hat{R}^{-1}\hat{N}' \\ \underline{u}_n = \underline{u} + \hat{R}^{-1}\hat{N}'\underline{x} \end{array} \right|$$

$$\Rightarrow PI = \int_0^\infty \{ \underline{x}' Q_n \underline{x} + \underline{u}_n' \hat{R} \underline{u}_n \} dt \stackrel{!}{=} \underset{\underline{u}_n}{\mathrm{Min}}$$

$\Rightarrow$ is a performance index with state weighting and without mixed terms for a modified problem:

$$\left| \begin{array}{l} \dot{\underline{x}} = A_n \underline{x} + B_n \underline{u}_n \\ \underline{y} = C_n \underline{x} + D_n \underline{u}_n \end{array} \right|$$

$$\underline{y} = C\underline{x} + D\underline{u}$$

$$= C\underline{x} + D\underline{u}_n - D\hat{R}^{-1}\hat{N}'\underline{x}$$

$$\Rightarrow \underline{y} = [\underbrace{C - D\hat{R}^{-1}\hat{N}'}_{C_n}]\underline{x} + \underbrace{D}_{D_n}\underline{u}_n$$

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

$$= A\underline{x} + B\underline{u}_n - B\hat{R}^{-1}\hat{N}'\underline{x}$$

$$\Rightarrow \dot{\underline{x}} = [\underbrace{A - B\hat{R}^{-1}\hat{N}'}_{A_n}]\underline{x} + \underbrace{B}_{B_n}\underline{u}_n$$

$$\Rightarrow \left| \begin{array}{l} \dot{\underline{x}} = A_n\underline{x} + B\underline{u}_n \\ \underline{y} = C_n\underline{x} + D\underline{u}_n \end{array} \right|$$

where: $\left| \begin{array}{l} A_n = A - B\hat{R}^{-1}\hat{N}' \\ C_n = C - D\hat{R}^{-1}\hat{N}' \end{array} \right|$

$$PI = \int_0^\infty \{\underline{x}'Q_n\underline{x} + \underline{u}_n'\hat{R}\underline{u}_n\}dt \overset{!}{=} \min_{\underline{u}_n}$$

with:

$$\left|\begin{array}{l} \hat{Q} = C'QC \\ \hat{N} = C'QD \\ \hat{R} = R + D'QD \\ Q_n = \hat{Q} - \hat{N}\hat{R}^{-1}\hat{N}' \end{array}\right|$$

$$\Rightarrow \hat{H} = \begin{bmatrix} A_n & -B\hat{R}^{-1}B' \\ -Q_n & -A_n' \end{bmatrix}$$

$$\Rightarrow \ldots \Rightarrow \hat{K} = B'\hat{R}^{-1}\hat{P}$$

for new formulation.

$$\Rightarrow \underline{u}_n = \underline{r} - \hat{K}\underline{x} \equiv \underline{u} + \hat{R}^{-1}\hat{N}'\underline{x}$$

$$\Rightarrow \underline{u} = \underline{r} - \underbrace{\left[\hat{K} + \hat{R}^{-1}\hat{N}'\right]}_{K}\underline{x}$$

$$\boxed{\underline{u} = \underline{r} - K\underline{x}}$$ is again a state feedback

where: $$\boxed{K = \hat{K} + \hat{R}^{-1}\hat{N}'}$$

for old formulation

In Matlab:

Let us solve the problem:

$$\left| \begin{array}{l} \dot{\underline{x}} = A\underline{x} + B\underline{u} \\ \underline{y} = C\underline{x} + D\underline{u} \end{array} \right|$$

$$PI = \int_0^\infty [\underline{x}' \ \underline{u}'] \cdot \begin{bmatrix} \hat{Q} & \hat{N} \\ \hat{N}' & \hat{R} \end{bmatrix} \cdot \begin{bmatrix} \underline{x} \\ \underline{u} \end{bmatrix} dt \overset{!}{=} \min_{\underline{u}(t)}$$

Algorithm:

$$\left| \begin{array}{l} An = A - (B/\hat{R}) * \hat{N}'; \\ Qn = \hat{Q} - (\hat{N}/\hat{R}) * \hat{N}'; \\ \hat{K} = lqr(An, B, Qn, \hat{R}); \\ K = \hat{K} + (\hat{R} \backslash \hat{N}'); \end{array} \right.$$

This also exists as a pre-programmed Matlab function:

$$K = lqr(A, B, \hat{Q}, \hat{R}, \hat{N});$$

Let us now solve the complete output weighting problem:

$$\left| \begin{array}{l} \dot{\underline{x}} = A \cdot \underline{x} + B \cdot \underline{u} \\ \underline{y} = C \cdot \underline{x} + D \cdot \underline{u} \end{array} \right|$$

$$PI = \int_0^\infty [\underline{y}' \ \underline{u}'] \cdot \begin{bmatrix} Q & \emptyset \\ \emptyset & R \end{bmatrix} \cdot \begin{bmatrix} \underline{y} \\ \underline{u} \end{bmatrix} dt \overset{!}{=} \min_{\underline{u}(t)}$$

Algorithm:

```
Q̂ = C'*Q*C;
R̂ = R + D'*Q*D;
N̂ = C'*Q*D;
K = lqr(A, B, Q̂, R̂, N̂);
```

This can also be obtained by:

```
                # of outputs
                ↓   # of inputs
                    ↓
N = zeros(p, m);
K = lqry(A,B,C,D,Q,R,N);
```

Exponential Stability:

In order to guarantee a maximum settling time, we want all eigenvalues by at least a factor of σ left from the imaginary axis, where:

$$\boxed{\sigma \approx \frac{4}{T_s}}$$

With optimal control, this problem can also be solved very easily.

$$\left| \begin{array}{l} \dot{\underline{x}} = A\underline{x} + B\underline{u} \\ \underline{y} = C\underline{x} + D\underline{u} \end{array} \right|$$

be a system to be controlled.

- We can look at a different system:

$$\left| \begin{array}{l} \dot{\underline{x}}_n = \tilde{A}\underline{x}_n + B\underline{u} \\ \underline{y}_n = C\underline{x}_n + D\underline{u} \end{array} \right|$$

where: $\quad \tilde{A} = A + \alpha I$

$$\Rightarrow \{\mathrm{Eig}(\tilde{A})\} = \{\mathrm{Eig}(A) + \alpha\}$$

We now design the state feedback for the new system: K

$$\Rightarrow \tilde{A}_{CL} = \tilde{A} - BK$$

is stable.

- We now apply this feedback K to our original problem:

$$\Rightarrow A_{CL} = A - BK$$

$$\Rightarrow A_{CL} = \tilde{A} - \alpha I - BK$$
$$= \tilde{A} - BK - \alpha I$$
$$= \tilde{A}_{CL} - \alpha I$$

$$\Rightarrow \{\underbrace{\mathrm{Eig}}_{\Uparrow}(A_{CL})\} = \{\underbrace{\mathrm{Eig}}_{\Uparrow}(\tilde{A}_{CL}) - \alpha\}$$

at least α left from imaginary axis.

stable

Example:

Given the system:

$$\dot{\underline{x}} = \begin{bmatrix} -150 & 192 & 12 & 165 \\ 143 & -181 & -15 & -154 \\ -142 & 179 & 15 & 153 \\ -291 & 370 & 28 & 316 \end{bmatrix} \underline{x} + \begin{bmatrix} 1 \\ -1 \\ 1 \\ 2 \end{bmatrix} u$$

$$y = \begin{bmatrix} -27 & 51 & -18 & 48 \end{bmatrix} \underline{x}$$

(cf. page 232)

- We want to design an output feedback using optimal control where all controller poles are at least 4 from the imaginary axis, all observer poles are at least 8 from the imaginary axis. We use output weighting with:

$$Q = [1] \quad \text{and} \quad R = [1].$$

- We already have found a model for the system:

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -24 & 2 & 5 \end{bmatrix} \underline{x} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$$

$$y = \begin{bmatrix} 3 & 3 & 0 \end{bmatrix} \underline{x}$$

(cf. page 235)

$\Rightarrow$

```
[> A=[0,1,0; 0,0,1; -24,2,5];
[> B=[0;0;1];
[> C=[3,3,0];
[> D=0;
[> AT= A + 4*EYE(A);
[> Q = 1;
[> R = 1;
[> N = 0;
[> K=LQRY(AT,B,C,D,Q,R,N)
```

$\Rightarrow$ K = [769.0225   272.1969   34.0079]

```
[> EIG(A - B*K)
```

$$\Rightarrow \quad ANS = \begin{bmatrix} -6.0042 \\ -11.0395 \\ -11.9642 \end{bmatrix}$$

<u>all real !!!</u>

Observer design:

```
[> AT = A + 8*EYE(A);
[> H = LQRY(AT', C', B', D', Q, R, N);
[> H = H'
```

$$\Rightarrow \quad H = \begin{bmatrix} -73.7788 \\ 93.1127 \\ 312.9170 \end{bmatrix}$$

```
[> EIG(A - H*C)
```

$$\Rightarrow \quad ANS = \begin{bmatrix} -14.0011 \\ -19.0153 \\ -19.9854 \end{bmatrix}$$

$\Rightarrow$ $\underline{k}'$ and $\underline{h}$ are too large.

Balancing:

```
[> T = SQRT(ABS(K'./H));
[> T = DIAG(T);
[> AN = T*A/T;
[> BN = T*B;
[> CN = C/T;
```

$$\Rightarrow \left| \begin{array}{l} \dot{\underline{v}} = \begin{bmatrix} 0 & 1.8883 & 0 \\ 0 & 0 & 5.1864 \\ -2.4507 & 0.3856 & 5 \end{bmatrix} \underline{v} + \begin{bmatrix} 0 \\ 0 \\ 0.3297 \end{bmatrix} u \\ \\ y = \begin{bmatrix} 0.9292 & 1.7546 & 0 \end{bmatrix} \underline{v} \end{array} \right.$$

is a new model.

$\Rightarrow$ [> KN = K/T

$\Rightarrow$ $KN = [238.1965 \quad 159.2011 \quad 103.1583]$

[> HN = T * H

$$\Rightarrow HN = \begin{bmatrix} -238.1965 \\ 159.2011 \\ 103.1583 \end{bmatrix}$$

$\Rightarrow$ We have requested too much. Looking at the closed loop eigenvalues, we realize that LQR has placed them too far to the left.

- We repeat the design by lowering our demands:

  controller poles left from -2.5
  observer poles left from -5

(in the hope that LQR still places them where we want them to be).

⇒ [> AT= A + 2.5*EYE(A);
[> K= LQRY(AT, B, C, D, Q, R, N)

⇒ K= [193.4Ø91 125.3972 25.Ø227]
[> EIG (A - B*K)

⇒ $$ANS = \begin{bmatrix} -3.014 \\ -8.0676 \\ -8.9411 \end{bmatrix}$$ NO!

⇒ iterate a little ...

[> AT= A + 3*EYE(A);
[> K= LQRY(AT, B, C, D, Q, R, N)
⇒ K= [337.1588 168.2872 28.Ø144]
[> EIG (A - B*K)

⇒ $$ANS = \begin{bmatrix} -4.008 \\ -9.0554 \\ -9.9509 \end{bmatrix}$$ ← YES!

[> AT= A + 6*EYE(A);
[> H= LQRY (AT', C', B', D', Q, R, N);
[> H= H'

⇒ $$H = \begin{bmatrix} -31.1798 \\ 46.5142 \\ 214.1975 \end{bmatrix}$$

[> EIG(A - H*C)

⇒ $$ANS = \begin{bmatrix} -10.0019 \\ -15.0232 \\ -15.9784 \end{bmatrix}$$ ← iterate a little...

```
[> AT = A + 5 * EYE(A);
[> H = LQRY(AT', C', B', D', Q, R, N);
[> H = H'
```

$$\Rightarrow \quad H = \begin{bmatrix} -17.8802 \\ 31.2152 \\ 168.8428 \end{bmatrix}$$

```
[> EIG (A - H*C)
```

$$\Rightarrow \quad ANS = \begin{bmatrix} -8.0026 \\ -13.0297 \\ -13.9726 \end{bmatrix} \leftarrow \quad \text{very good!}$$

Balancing:

```
[> T = SQRT (ABS(K'./H));
[> T = DIAG(T);
[> KN = K / T,   HN = T*H
```

$$\Rightarrow KN = \begin{bmatrix} 77.6431 & 72.4784 & 68.7752 \end{bmatrix}$$

$$HN = \begin{bmatrix} -77.6431 \\ 72.4784 \\ 68.7752 \end{bmatrix}$$

This design looks okay.

The new model is:

$$\left| \begin{array}{l} \dot{\underline{\mu}} = \begin{bmatrix} 0 & 1.8702 & 0 \\ 0 & 0 & 5.7002 \\ -2.2513 & 0.3509 & 5 \end{bmatrix} \underline{\mu} + \begin{bmatrix} 0 \\ 0 \\ 0.4073 \end{bmatrix} u \\ y = \begin{bmatrix} 0.6909 & 1.292 & 0 \end{bmatrix} \underline{\mu} \end{array} \right.$$

Verification:

```
[> AN = T * A / T;
[> BN = T * B;
[> CN = C / T;
[> AT = AN + 3 * EYE (AN);
[> K = LQRY(AT, BN, CN, D, Q, R, N)
```

$$\Rightarrow \quad K = \begin{bmatrix} 77.6431 & 72.4784 & 68.7752 \end{bmatrix}$$

✓

```
[> AT = AN + 5 * EYE (AN);
[> H = LQRY(AT', CN', BN', D', Q, R, N);
[> H = H'
```

$$\Rightarrow \quad H = \begin{bmatrix} -77.6431 \\ 72.4784 \\ 68.7752 \end{bmatrix}$$

✓

From this, we can learn that obviously optimality and displacement of eigen= values (exponential stability) are invariant to this kind of similarity transformation.