ELEC-E8116 Model-based control systems
/exercises 10 Solutions

Problem 1:

Consider the multivariable plant

$$G(s) = \frac{1}{(0.2s+1)(s+1)}\begin{bmatrix} 1 & 1 \\ 1+2s & 2 \end{bmatrix}$$

a. Use RGA analysis to evaluate how bad the interconnections between the channels are. Calculate RGA both at zero frequency and the gain crossover frequency. Choose the preferred *pairing* and design *decentralized* PID controllers to control the system.
Implement the controller in Matlab/Simulink and plot the responses of the outputs when

   a. a unit step enters the reference of channel 1,
   b. a unit step enters the reference of channel 2 and
   c. unit steps enter at both channels simultaneously.

You may use Matlab in implementing the controller and simulating the closed loop, or you can use Simulink if you wish.

b. Design a decoupling controller using the singular value decomposition at zero frequency and choosing the weight matrices W_1 and W_2 accordingly (see Chapter 6 in the lecture slides). Use PID-controllers in the decoupled system. Simulate as in part a.

Hints: Gain crossover of a MIMO system is calculated based on the largest singular value. In tuning the PID controllers you may use the tuning functions in Matlab, see e.g. *pidtune*.

Solution:

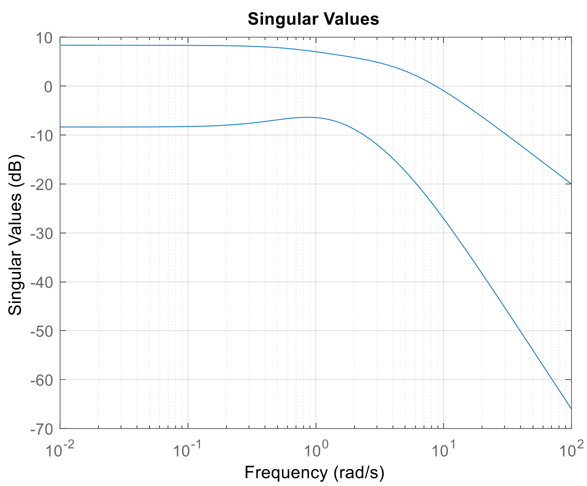
The Matlab code is at the end of the solution. (Note that many of the commands have been set as comments, %. The idea is that for different experiments only a part of the commands is made active by removing the %.)

a.

$$G(s=0) = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}$$

The RGA at zero frequency is

$$RGA(G(s=0)) = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \times \underbrace{\left(\begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}^{-1} \right)^T}_{\begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}} = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}.$$ By the command *sigma* we get the singular values of $G(s)$.



The gain crossover (larger singular value) is approximately 8.8 rad/s. The RGA as a function of angular frequency is

$$RGA = G(j\omega).*G^{-1}(j\omega)^T = \frac{1}{(1+j0.2\omega)(1+j\omega)} \begin{bmatrix} 1 & 1 \\ 1+j2\omega & 2 \end{bmatrix}$$

$$.*\frac{(1+j0.2\omega)(1+j\omega)}{1-j2\omega} \begin{bmatrix} 2 & -(1+j2\omega) \\ -1 & 1 \end{bmatrix} \overset{[1]}{=} \frac{1}{1-j2\omega} \begin{bmatrix} 2 & -(1+j2\omega) \\ -(1+j2\omega) & 2 \end{bmatrix}$$

We don't need to multiply the common terms with each element. Instead, we can do the following.

[1]

$$\frac{1}{(1+j0.2\omega)(1+j\omega)} \begin{bmatrix} 1 & 1 \\ 1+j2\omega & 2 \end{bmatrix} .*\frac{(1+j0.2\omega)(1+j\omega)}{1-j2\omega} \begin{bmatrix} 2 & -(1+j2\omega) \\ -1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \\ 1+j2\omega & 2 \end{bmatrix} .*\frac{1}{\cancel{(1+j0.2\omega)(1+j\omega)}} \frac{\cancel{(1+j0.2\omega)(1+j\omega)}}{1-j2\omega} \begin{bmatrix} 2 & -(1+j2\omega) \\ -1 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \\ 1+j2\omega & 2 \end{bmatrix} .*\frac{1}{1-j2\omega} \begin{bmatrix} 2 & -(1+j2\omega) \\ -1 & 1 \end{bmatrix} = \frac{1}{1-j2\omega} \begin{bmatrix} 1 & 1 \\ 1+j2\omega & 2 \end{bmatrix} .* \begin{bmatrix} 2 & -(1+j2\omega) \\ -1 & 1 \end{bmatrix}$$

which would give the result above.

Clearly, at the zero angular frequency we get the same result as above. But for ω = 8.8 the result becomes (after some algebra)

$$RGA \approx \begin{bmatrix} 0.0064 + j0.1133 & 0.9936 - j0.1133 \\ 0.9936 - j0.1133 & 0.0064 + j0.1133 \end{bmatrix}$$

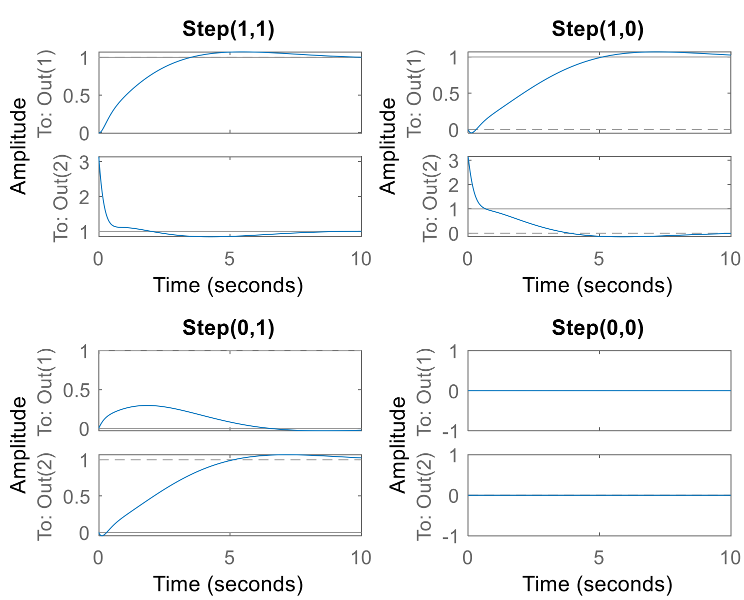
(As for the algebra, for example

$$\frac{2}{1 - j2\omega} = \frac{2(1 + j2\omega)}{(1 - j2\omega)(1 + j2\omega)} = \frac{2 + j4\omega}{1 + 4\omega^2} = \frac{2}{1 + 4\omega^2} + j\frac{4\omega}{1 + 4\omega^2} \;)$$

Now it is interesting to note that at the zero angular frequency the preferable pairing would seem to be $u_1 \leftrightarrow y_1, u_2 \leftrightarrow y_2$, although neither one is really good. But at the gain crossover frequency the pairing $u_1 \leftrightarrow y_2, u_2 \leftrightarrow y_1$ should be preferred.

Simulation:
PID ($u_1 \leftrightarrow y_1, u_2 \leftrightarrow y_2$) tuned by Matlab's *pidtune*. The solution is unstable. The tuning values for *Fy1* were *Kp* = 1.95, *Ki* = 2.65, *Kd* = 0.314, and for *Fy2 Kp* = 0.973, *Ki* = 1.32, *Kd* = 0.157.

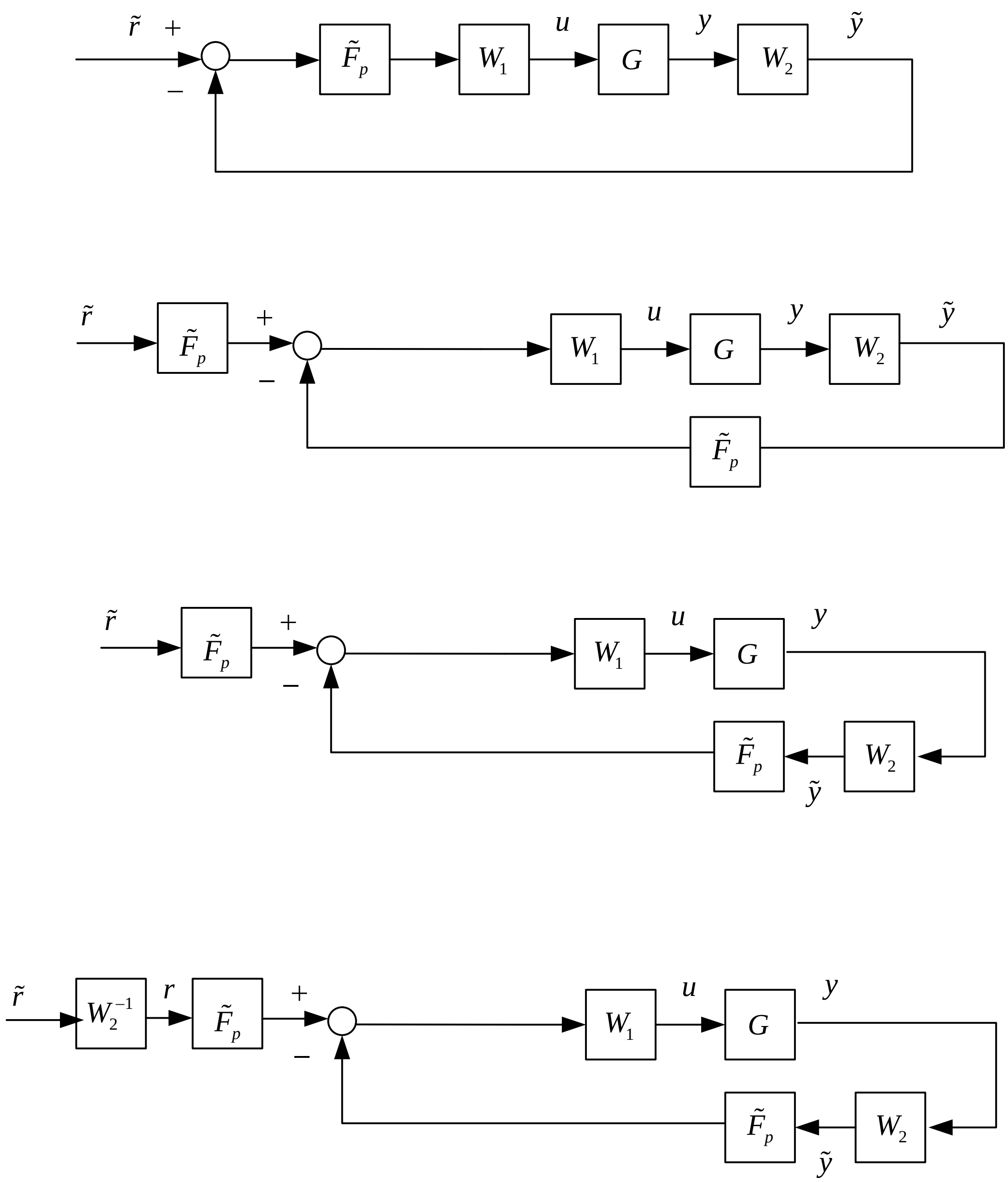
But when they are changed to *Fy1* *Kp* = 1, *Ki* = 1, *Kd* = 0.314, *Fy2 Kp* = 0.5, *Ki* = 0.5, *Kd* = 0.157 the following result is obtained



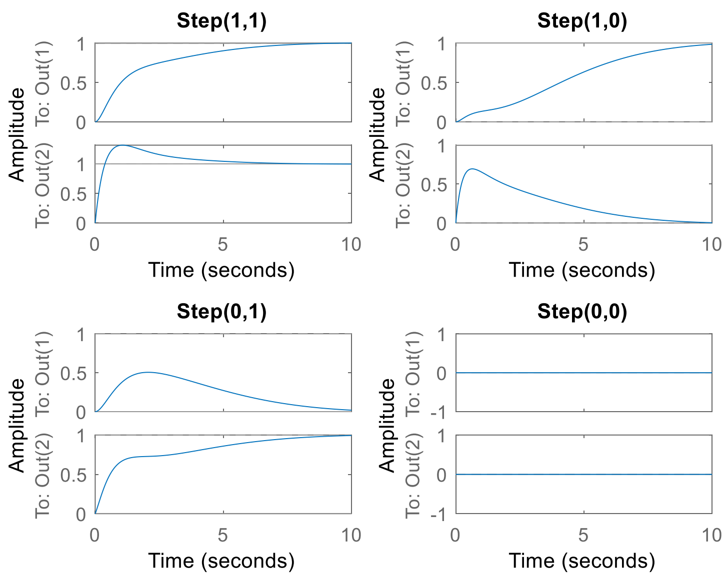
PID ($u_1 \leftrightarrow y_2, u_2 \leftrightarrow y_1$) TUNED BY Matlab's *pidtune*. The result is unstable. In this case it turned out to be difficult to re-tune the controller such that a good response would be obtained.

Conclusion: *pidtune* is for SISO tuning only. There is no guarantee that it would be good for MIMO systems. The RGA analysis showed that at zero frequency the coupling $u_1 \leftrightarrow y_1, u_2 \leftrightarrow y_2$ should be preferred. For step responses that seems to be so. At the gain crossover frequency the pairing $u_1 \leftrightarrow y_2, u_2 \leftrightarrow y_1$ seemed to be preferable, but that could not be observed by simulations with step reference inputs.

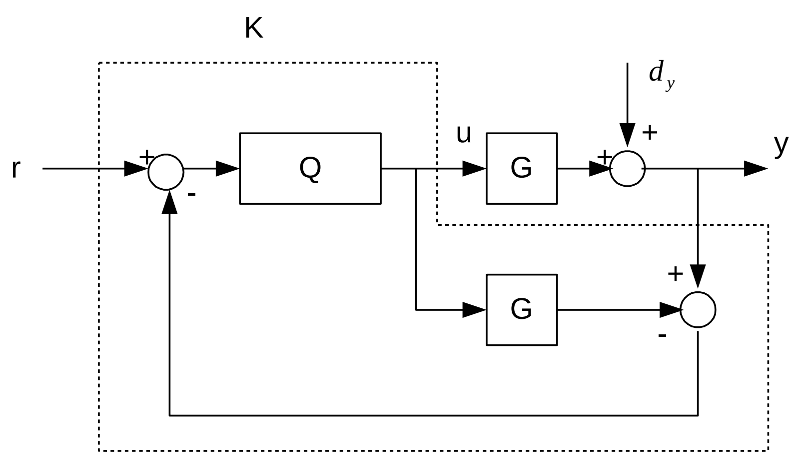
b. There are many ways to solve this. The solution below is based on the below figure, from which the program code becomes understandable. See also Chapter 6 in the lecture slides.



The last part of the figure can be explained by noting that since $y = W_2^{-1}\tilde{y}$ then also $r = W_2^{-1}\tilde{r}$. The reference signal has to be scaled to get the static gain from r to y to be 1. The following results are obtained by the below program code. The result is acceptable but a bit slow. Again, the PID tuning could be improved.



Problem 2: Consider the following IMC-control configuration, in which the process G is assumed stable.



a. Prove that to study the internal stability, the stability of the transfer functions

$$K(I+GK)^{-1} = Q$$
$$(I+GK)^{-1} = I - GQ$$
$$(I+KG)^{-1} = I - QG$$
$$G(I+KG)^{-1} = G(I-QG)$$

must be investigated. Prove that the system is internally unstable, if either Q or G is unstable.

b. Let a stable controller K be given. How can you characterize those processes, which can be stabilized with this controller? (Hint: Change the roles of the controller and process.)

Solution:

a. For the control it holds

$$y = Gu + d_y$$
$$y_m = y - Gu$$
$$e = r - y_m$$
$$u = Qe = Qr - Qy_m = Qr - Qy + Gu$$
$$u = Q[r - (y - Gu)] = Q(r - y) + QGu$$

from which it follows easily

$$u = (I - QG)^{-1} Q(r - y)$$

But this has the form

$$u = K(r - y)$$

where $K = (I - QG)^{-1} Q = Q(I - GQ)^{-1}$

and

$$Q = K(I + GK)^{-1}$$

Recall from exercise 5

$$u = (I + KG)^{-1} d_u - K(I + GK)^{-1} d_y$$
$$y = G(I + KG)^{-1} d_u - (I + GK)^{-1} d_y$$

By this controller the configuration is equivalent to the ”one-degree-of-freedom”-structure. Based on lectures (Chapter 3, Internal stability of closed-loop systems) it is known that the system is internally stable, if the transfer functions

$$K(I+GK)^{-1} = Q$$
$$(I+GK)^{-1} = I - GQ$$
$$(I+KG)^{-1} = I - QG$$
$$G(I+KG)^{-1} = G(I - QG)$$

are stable (the ”right sides” follow easily from the choice of Q).

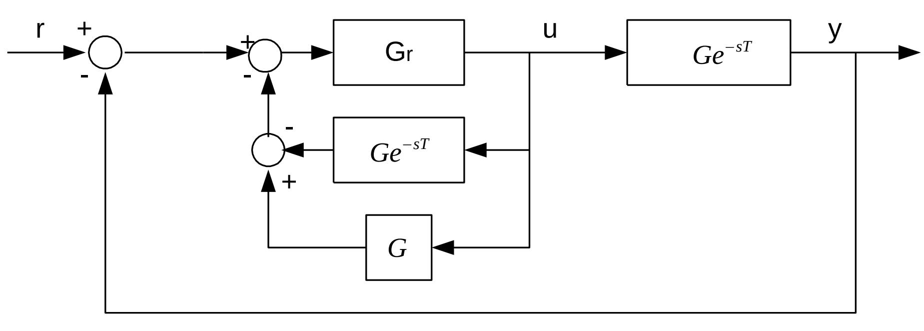
But the functions are clearly stable, if Q and G are stable. Correspondingly, if either one is unstable, the system is internally unstable.

b. These systems can be represented in the form (parameterization)

$$G = (I - QK)^{-1} Q = Q(I - KQ)^{-1}$$

where Q is any stable transfer function matrix.

Problem 3. Consider the control configuration shown in the figure (known as the *Smith-predictor*). Calculate the closed loop transfer function and verify the idea behind this controller. Compare to the *IMC*-controller and prove that the Smith predictor always leads to an internally unstable system, if the process is unstable.



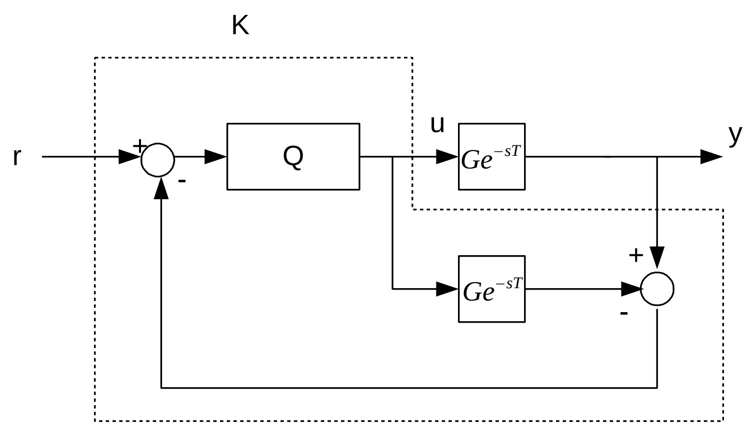
Solution: By using block diagram algebra the transfer function from r to y is easily calculated to be

$$Y(s) = \frac{G_r(s)G(s)}{1+G_r(s)G(s)} e^{-sT} R(s)$$

which reveals the idea behind this control configuration: the basic controller G_r can be designed to give a good closed loop response without paying any attention to the process delay. The real response is then the same but added with a pure delay *T*. The term e^{-sT} is not shown in the characteristic equation (which would happen, if G_r would directly control the process with delay). But note that in this ideal case the process is exactly known and the intermediate block in the controller generates the predicted value of the output. In reality an inaccurate process model has to be used for this purpose.

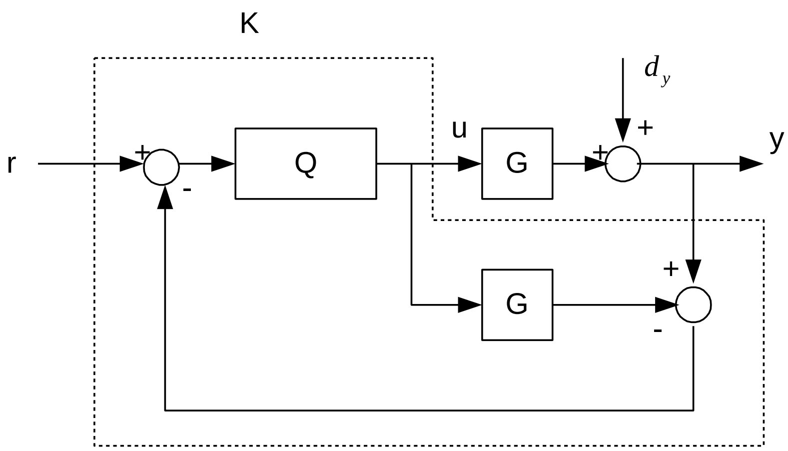
But: moving the block Ge^{-sT} in the figure a bit (without changing the control signal *u*, of course), the configuration below is obtained. That is directly the *IMC*-structure. There

$$Q = \frac{G_r}{1+G_r G}$$



But what if the process *G* is unstable? Look at the previous problem, in which it was shown that the closed loop system is internally unstable, if *Q* or *G* is unstable. Because the Smith predictor structure was above shown to be equivalent to the *IMC*-structure, the closed loop is inevitably (internally) unstable, if the process is unstable.

Problem 4. Consider the IMC control structure, which is used to control a stable and minimum phase SISO process *G*.



Note that in addition to the reference r a disturbance signal d_y is modelled to enter at the output of the process. By using the IMC design discussed in the lectures analyse the response to step inputs at r and d_y.

Solution:

The figure represents a two-degrees-of-freedom control configuration, where the inputs to the controller K are r and y. Again, it is easy to write

$$u = Q\left[r-(y-Gu)\right] = Q(r-y)+QGu \Rightarrow u = (I-QG)^{-1}Q(r-y)$$

But that can be interpreted as a one-degree-of-freedom configuration with the controller

$$u = K_1(r-y), \quad K_1 = (I-QG)^{-1}Q = \frac{Q}{1-QG} \qquad \text{(SISO!)}$$

Using the design (see lecture slides). G has more poles than zeros

$$Q = \frac{1}{\left(\lambda s+1\right)^n} G^{-1}$$ and writing equations from the topology in the figure

$$y = d_y + Gu = d_y + GK_1(r-y) \Rightarrow y = \frac{GK_1}{1+GK_1} r + \frac{1}{1+GK_1} d_y$$

Setting K_1 to this gives after simple calculations

$$y = \frac{\frac{GQ}{1-QG}}{1+\frac{GQ}{1-QG}} r + \frac{1}{1+\frac{GQ}{1-QG}} d_y = GQr + (1-QG) d_y = \frac{1}{(\lambda s+1)^n} r + \left[1 - \frac{1}{(\lambda s+1)^n}\right] d_y$$

Note that $GQ = QG$ for SISO systems. Also $y = GQr + (1-QG) d_y$ could have been obtained directly from the figure (careful!).

$$\begin{cases} y = Gu + d_y \\ u = (I-QG)^{-1}Qr - (I-QG)^{-1}Qy \end{cases}$$

$$\Leftrightarrow y = G(I-QG)^{-1}Qr + d_y$$

$$\Leftrightarrow y + G(I-QG)^{-1}Qy = G(I-QG)^{-1}Qr + d_y$$

$$\Leftrightarrow y + \frac{GQ}{(1-QG)} y = \frac{GQ}{(1-QG)} r + d_y$$

$$\Leftrightarrow y = \frac{1-QG}{1} \frac{GQ}{(1-QG)} r + \frac{1-QG}{1} d_y$$

$$\Leftrightarrow y = GQr + (1-GQ) d_y$$

Setting $s = 0$ we find that the static gain from r to y is 1 and from d_y to y it is0, so that the output follows the reference and mitigates the disturbance asymptotically. Note that internal stability was guaranteed by the fact that G was stable and minimum phase (G^{-1} stable) and Q stable.

Program code:

```
% Model-based control systems
%
%
s=tf('s');
G=1/((0.2*s+1)*(s+1))*[1 1;1+2*s 2];
G0=[1 1;1 2];
RGAG0=G0.*(inv(G0))';
sigma(G);
%
% Let us try a pairing u1-y1, u2-y2 first.  PID
control.
%[Fy1,Info1]=pidtune(G(1,1),'pid');
%[Fy2,Info2]=pidtune(G(2,2),'pid');
```

```
%1 DOF controllers
%Fy=[Fy1 0;0 Fy2];
%Fr=Fy;

% Let us then try a pairing u1-y2, u2-y1.  PID
control.
%[Fy1,Info1]=pidtune(G(1,2),'pid');
%[Fy2,Info2]=pidtune(G(2,1),'pid');

%1 DOF controllers
%Fy=[0 Fy1;Fy2 0];
%Fr=Fy;

%SVD and construction of pre- and post
compensators
[U,S,V]=svd(G0);
W1=V; W2=U';
%Diagonalazied plant; design of controller
Gworm=minreal(W2*G*W1);
[Fp1worm,Info1]=pidtune(Gworm(1,1),'pid');
[Fp2worm,Info2]=pidtune(Gworm(2,2),'pid');
%1 DOF controller in "worm" domain
Fpworm=[Fp1worm 0;0 Fp2worm];
% 2 DOF
Fy=Fpworm*W2;
Fr=Fpworm*inv(W2);
G=G*W1; % Modified plant

%Sensitivity functions
L=minreal(G*Fy);
S=minreal(inv(eye(2)+L));
Tcomp=minreal(eye(2)-S);
Gc=S*G*Fr;
Gc2=Gc;

figure(1)
sigma(L,S,Tcomp)
title('Functions L, S, T')
```

```
%
%
%Simulation
T=0:0.01:10;
figure(2)
Uinp=ones(length(T),2);
subplot(221)
lsim(Gc2,Uinp,T)
title('Step(1,1)')
%
Uinp=[ones(length(T),1) zeros(length(T),1)];
subplot(222)
lsim(Gc2,Uinp,T)
title('Step(1,0)')
%
Uinp=[zeros(length(T),1) ones(length(T),1)];
subplot(223)
lsim(Gc2,Uinp,T)
title('Step(0,1)')
%
Uinp=[zeros(length(T),1) zeros(length(T),1)];
subplot(224)
lsim(Gc2,Uinp,T)
title('Step(0,0)')
```