Conversion of Linear Systems to the Frequency Domain:

Since:

$$y(t) = a_1 \cdot x_1(t) + a_2 \cdot x_2(t)$$

$$\circ\!-\!\bullet$$

$$Y(s) = a_1 \cdot X_1(s) + a_2 \cdot X_2(s)$$

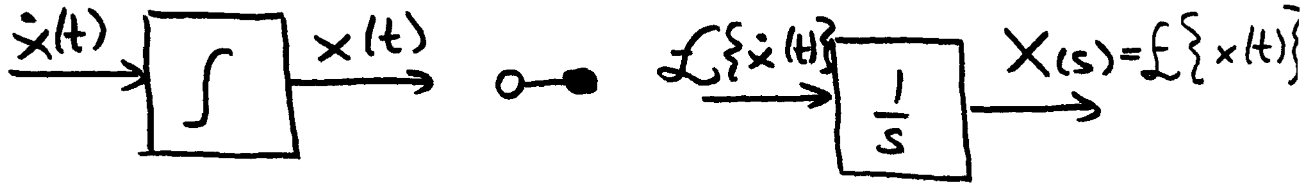
the structure of a block diagram is preserved in the conversion.

We only need to discuss the integration box separately.

Without initial conditions:

$$\mathcal{L}\{\dot{x}(t)\} = s \cdot X(s)$$

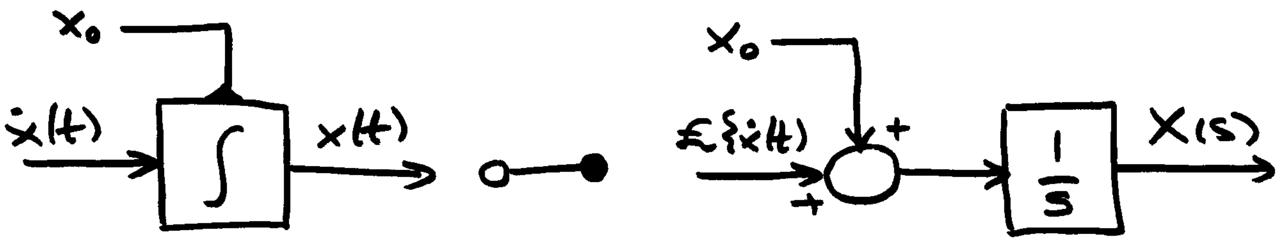
$$\Rightarrow X(s) = \frac{1}{s} \cdot \mathcal{L}\{\dot{x}(t)\}$$



With initial conditions:

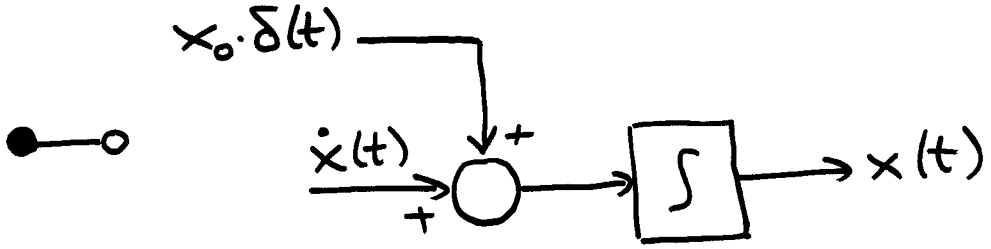
$$\mathcal{L}\{\dot{x}(t)\} = s \cdot X(s) - x_0$$

$$\Rightarrow X(s) = \frac{1}{s} \cdot \left[\mathcal{L}\{\dot{x}(t)\} + x_0\right]$$



Of course, we can also take the block diagram in the frequency domain and map it back.

Since $\mathcal{L}^{-1}\{x_0\} = x_0 \cdot \delta(t)$



We can replace the initial condition by a Dirac input.
(We shall talk about this later.)

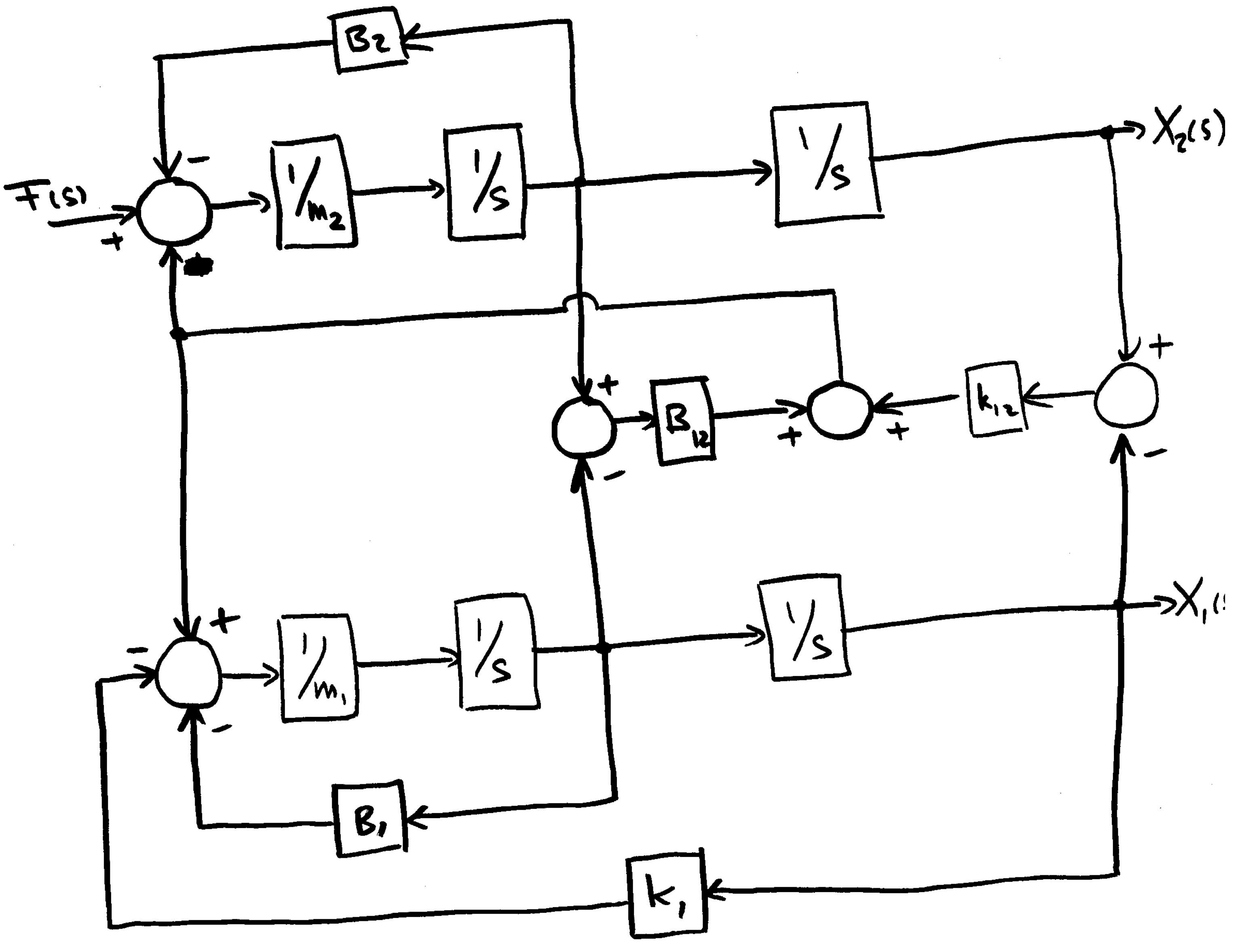
In the given example, we want to make the assumption that:

$$\dot{x}_1 = \dot{x}_2 = \emptyset \implies x_1 = x_2 = \emptyset$$

i.e., in steady-state ($\dot{x}_1 = \dot{x}_2 = \emptyset$), the positions are normalized to be zero as well.

If we furthermore assume that the simulation starts at steady-state, we don't need to worry about initial conditions.

Hence the block diagram can be translated to the frequency domain in a straightforward manner:



There after, all blockdiagram simplification techniques learned earlier will apply.

Given a state-space model:

$$\left| \begin{array}{l} \underline{\dot{x}} = A \cdot \underline{x} + \underline{b} \cdot u \\ y = \underline{c}' \cdot \underline{x} + d \cdot u \end{array} \right|$$

the conversion can be applied directly as well:

$$\underline{\dot{x}} \;\circ\!\!-\!\!\bullet\; sX(s) - x_0$$

$$\Rightarrow \left| \begin{array}{l} s\underline{X}(s) - \underline{x}_0 = A \cdot \underline{X}(s) + \underline{b} \cdot U(s) \\ Y(s) = \underline{c}' \cdot \underline{X}(s) + d \cdot U(s) \end{array} \right|$$

$$\Rightarrow (sI - A) \cdot \underline{X}(s) = \underline{b} \cdot U(s) + \underline{x}_0$$

$$\Rightarrow \underline{X}(s) = (sI-A)^{-1} \cdot \underline{b} \cdot U(s) + (sI-A)^{-1} \cdot \underline{x}_0$$

$$\Rightarrow Y(s) = \underline{c}'(sI-A)^{-1} \cdot \underline{b} \cdot U(s) + \underline{c}'(sI-A)^{-1} \cdot \underline{x}_0 + d \cdot U(s)$$

$$\Rightarrow Y(s) = [\underbrace{\underline{c}'(sI-A)^{-1} \cdot \underline{b} + d}_{G(s)}] \cdot U(s) + \underline{c}'(sI-A)^{-1} \cdot \underline{x}_0$$

—15—

5

Hierarchical Modular Modeling of Continuous Systems

Preview

To this point, we have dealt with very simple and small problems. In this chapter, we shall cover some of the techniques necessary for modeling larger systems. Very often, systems consist of subsystems that may be described in quite different ways. Besides state–space representations and topological descriptions (which we have met previously), subsystems may also be described in the frequency domain in terms of transfer functions or may simply be given as a static characteristic relating one output variable to one or several input variables. It is therefore important that models can be structured. Modular modeling enables us to encapsulate subsystem descriptions and treat them as unseparable entities that can be incorporated in a hierarchical fashion within ever–more–complex system descriptions.

5.1 Modeling Transfer Functions

Let us assume a system is described by the following transfer function:

$$G(s) = 200\frac{(s+1)}{(s+10)(s+20)} \tag{5.1}$$

In order to make this system amenable to simulation, we need to convert the specification back from the frequency domain into the time domain. The easiest way to do this is the following.

$$G(s) = \frac{200 + 200s}{200 + 30s + s^2} = \frac{P(s)}{Q(s)} = \frac{Y(s)}{U(s)} \tag{5.2}$$

where $G(s)$ denotes the transfer function, $P(s)$ denotes its numerator polynomial, $Q(s)$ denotes its denominator polynomial, $Y(s)$ denotes the output signal, and $U(s)$ denotes the input signal. We introduce an additional signal $X(s)$

$$G(s) = \frac{Y(s)}{U(s)} = \frac{Y(s)}{X(s)} \cdot \frac{X(s)}{U(s)} \tag{5.3}$$

such that

$$\frac{X(s)}{U(s)} = \frac{1}{Q(s)} \tag{5.4a}$$

$$\frac{Y(s)}{X(s)} = P(s) \tag{5.4b}$$

We shall look at Eq.(5.4*a*) first. We can rewrite this for our example as:

$$[200 + 30s + s^2]X(s) = U(s) \tag{5.5}$$

which can be transformed back into the time domain as:

$$200x(t) + 30\dot{x}(t) + \ddot{x}(t) = u(t) \tag{5.6}$$

assuming that all initial conditions are zero, which is standard practice when operating on transfer functions. We now solve Eq.(5.6) for its highest derivative:

$$\ddot{x}(t) = -200x(t) - 30\dot{x}(t) + u(t) \tag{5.7}$$

Finally, we introduce the following state variables:

$$\xi_1 = x \tag{5.8a}$$

$$\xi_2 = \dot{x} \tag{5.8b}$$

which leads us to the following state–space model:

$$\dot{\xi}_1 = \xi_2 \tag{5.9a}$$

$$\dot{\xi}_2 = -200\xi_1 - 30\xi_2 + u \tag{5.9b}$$

-17-

We now look at Eq.(5.4*b*), which can be written for our example as:

$$Y(s) = [200 + 200s]X(s) \tag{5.10}$$

or in the time domain:

$$y(t) = 200x(t) + 200\dot{x}(t) \tag{5.11}$$

and using our state variables:

$$y = 200\xi_1 + 200\xi_2 \tag{5.12}$$

We can rewrite Eqs.(5.9*a*–*b*) and Eq.(5.12) in a matrix form as:

$$\dot{\xi} = \begin{pmatrix} 0 & 1 \\ -200 & -30 \end{pmatrix} \xi + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u \tag{5.13a}$$

$$y = \begin{pmatrix} 200 & 200 \end{pmatrix} \xi \tag{5.13b}$$

In general, if a system is specified through the transfer function:

$$G(s) = \frac{b_0 + b_1 s + b_2 s^2 + \ldots + b_{n-1} s^{n-1}}{a_0 + a_1 s + a_2 s^2 + \ldots + a_{n-1} s^{n-1} + s^n} \tag{5.14}$$

we can immediately convert this to the following state–space description:

$$\dot{\mathbf{x}} = \begin{pmatrix} 0 & 1 & 0 & \ldots & 0 & 0 \\ 0 & 0 & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \ldots & 1 & 0 \\ 0 & 0 & 0 & \ldots & 0 & 1 \\ -a_0 & -a_1 & -a_2 & \ldots & -a_{n-2} & -a_{n-1} \end{pmatrix} \mathbf{x} + \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 1 \end{pmatrix} u \tag{5.15a}$$

$$y = \begin{pmatrix} b_0 & b_1 & b_2 & \ldots & b_{n-2} & b_{n-1} \end{pmatrix} \mathbf{x} \tag{5.15b}$$

In other words, the *state matrix* consists of all zero elements except for a superdiagonal of one elements and the last row in which the negative coefficients of the denominator polynomial are stored. The *input vector* consists of zero elements except for the last element, which is one, and the *output vector* contains the positive coefficients of the numerator polynomial.

This technique will work fine as long as the numerator polynomial is of lower degree than the denominator polynomial. If this is not the case, we need to divide the numerator by the denominator first and separate in this way the direct input/output coupling from the remainder of the system. This procedure will be illustrated by means of another simple example:

$$G(s) = \frac{6s^3 + 32s^2 + 10s + 2}{2s^2 + 8s + 4} \tag{5.16}$$

We always start by normalizing the highest–degree coefficient of the denominator polynomial to one, i.e.:

$$G(s) = \frac{3s^3 + 16s^2 + 5s + 1}{s^2 + 4s + 2} \tag{5.17}$$

The division of polynomials works the same way as the division of regular numbers:

$$\begin{array}{rrrrl}
 & (3s^3 + & 16s^2 + & 5s + & 1\ ) : (s^2 + 4s + 2) = 3s + 4 \\
- & 3s^3 + & 12s^2 + & 6s & \\
\hline
 & \backslash & 4s^2 - & s + & 1 \\
- & & 4s^2 + & 16s + & 8 \\
\hline
 & & \backslash & -17s - & 7
\end{array}$$

i.e., $G(s)$ can also be written as:

$$G(s) = (3s + 4) + \frac{-17s - 7}{s^2 + 4s + 2} \tag{5.18}$$

which can be interpreted as a parallel connection of two subsystems, as depicted in Fig.5.1.

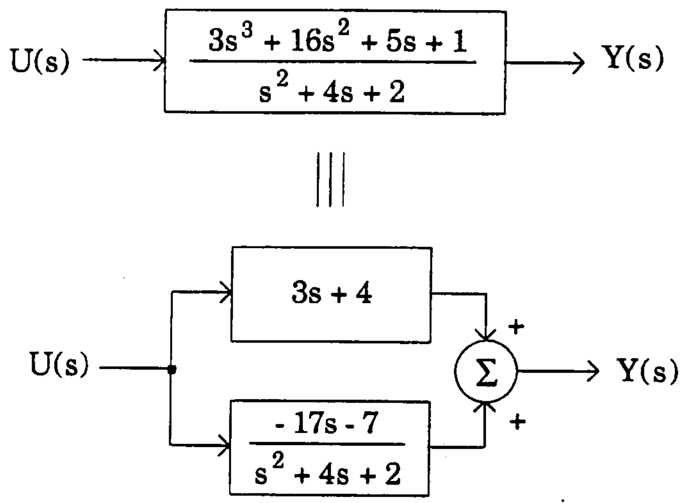

U(s)
$\frac{3s^3 + 16s^2 + 5s + 1}{s^2 + 4s + 2}$
Y(s)
|||
U(s)
3s + 4
$\frac{-17s - 7}{s^2 + 4s + 2}$
+
+
Σ
Y(s)

Figure 5.1. Separation of the direct input/output coupling.

-19-

The transfer function has been split into a polynomial that contains the *input/output coupling* of the system and a remainder transfer function, the numerator of which is now guaranteed to be of lower degree than the denominator polynomial. This contains the so-called *strictly proper* portion of the system.

In the case of our example, we end up with the following simulation model:

$$\dot{x}_1 = x_2 \tag{5.19a}$$
$$\dot{x}_2 = -2x_1 - 4x_2 + u \tag{5.19b}$$
$$y = -7x_1 - 17x_2 + 4u + 3\dot{u} \tag{5.19c}$$

As can be seen, a true differentiation of the input signal u was unavoidable in this case. This is always true when the numerator polynomial of a transfer function is of a higher degree than the denominator polynomial. As a small consolation, the necessary numerical differentiation is performed as part of the evaluation of output equations and has thereby been removed from the simulation loop. Numerical errors made in the process of numerical differentiation will not grow by being passed around the integration loop many times. We came upon this situation in Chapter 3, and at that time, I had mentioned (without a proof) that essential differentiators can, in linear systems, always be moved out of the simulation loop into the output equations. It has now become clear why this is the case and how this can be accomplished in practice.

5.2 Modeling Static Characteristics

Often, static but nonlinear functional relationships exist between input variables and output variables of a subsystem. Often, mathematical equations describing these relationships are not available. Instead, these relationships have been found through experimentation with a real system.

Let us demonstrate this concept by means of our lunar landing module, which now should be equipped to land on Earth instead. Of course, this wouldn't work with the rockets designed in the previous model, but let us be forbearing with these lesser details. However, it will be important to modify our mechanical equations to take the air