State variables in Feedback Control Systems

Stefano Borsarelli

March 5, 2022

Contents

1 Introduction

What is a "system" in control theory? It can be represented like an insulated-environment composed from many parts that interact themselves where people want study the behaviour. It's composed of "input variables", like external forces, and "output variables" namely some characteristic of the system that it's important to study for some reason. There are two kind of systems: *static* systems also called without memory where the output depends entirely from the input in that particular *instant* (like into a resistor current-voltage relation), and *dynamic* systems namely to *concentrated parameters* where the output does not depend only from the input (like a capacitor current-voltage). Mathematically spoken while the first ones are described by algebraic equations, the second ones are differential equations.

It is of particular interest to study the "dynamic systems" by a branch of the science called **Control Theory**, a very common concept with many characteristics. For example a system that involves a person controlling a vehicle, like a driver, is called *manual control*. Instead a system that involves machines only, as when a room temperature can be regulated by a thermostat, is called *automatic control*. Again, systems designed to hold an output steady versus an unknown disturbances are called *regulators*, while systems designed to track a reference signal are called *tracking* or *servos* systems. Control systems are also classified according to the information used to compute the controlling action. If the controller doesn't use the output measurement being controlled in computing the action to take, the system is called **open-loop control**. If the controlled output signal is measured and feed back for use in the control computation, the system is called closed-loop or **feed-back control**. There are many other properties of control systems in addition to these most basic characteristics. For example, in this document it will mainly be concerned with controlling processes that can be adequately described by **linear, time-invariant** equations, whereas all physical processes are non linear if the signals are large and their characteristics vary with time if observed for a long time. It will be clear that non linear processes will be studied around their particular points (like of equilibrium or expanding by Taylor series for small displacement around a fixed point) to apply the linear theory explained through the document.

The classic transform techniques used to design a compensator in feed back controls in order to obtain the desired characteristics of the systems are *root locus* and *frequency response* methods. There is a third major method of design feedback control systems: the state-space method. It will be introduced the state-variable method of describing differential equations of the systems studied.

A state variable is one of the set of variables that are used to describe the internal mathematical *state* of a dynamical system, which is determined by its history necessary to calculate the output. Intuitively, the state of a system describes enough about the system to determine its future behaviour in the absence of any external forces affecting the system. Models that consist of coupled first-order differential equations are said to be in state-variable form. There are some examples where state variables are used:

- In mechanical systems, the position coordinates and velocities of mechanical parts are typical state variables
- In thermodynamics, a state variable is an independent variable of a state function like internal energy, enthalpy, and entropy.
- In electronic/electrical circuits, the voltages of the nodes and the currents through components in the circuit are usually the state variables.
- In ecosystem models, population sizes (or concentrations) of plants, animals, and resources

(nutrients, organic material) are typical state variables.
- In pandemic models, restrictions(i.e. masks, lock-down, vaccines), percentage of cases, number of hospital beds, are used like state variables to create a control system.

In particular of last point feedbacks are part of Control Engineering (or control systems engineering), a discipline that deals with control systems, applying the above mentioned control theory to design equipment and systems with desired behaviors in control environments. The practice uses sensors and detectors to measure the output performance of the process being controlled; these measurements are used to provide corrective feedback helping to achieve the desired performance. Multi-disciplinary in nature, control systems engineering activities focus on implementation of control systems mainly derived by mathematical modeling of a diverse range of systems. There are two major divisions in control theory, namely, classical and modern, which have direct implications for the control engineering applications:

- **Classical SISO System Design**. The scope of classical control theory is limited to single-input and single-output system design, except when analyzing for disturbance rejection using a second input. The system analysis is carried out in the time domain using differential equations, in the complex-s domain with the Laplace transform, or in the frequency domain by transforming from the complex-s domain. Many systems may be assumed to have a second order and single variable system response in the time domain. The most common controllers designed using classical control theory are PID controllers. The ultimate end goal is to meet requirements typically provided in the time-domain called the step response, or at times in the frequency domain called the open-loop response. The step response characteristics applied in a specification are typically percent overshoot, settling time, etc.
- **Modern MIMO system Design**. Modern control theory is carried out in the state space, and can deal with multiple-input and multiple-output (MIMO) systems. This overcomes the limitations of classical control theory in more sophisticated design problems, such as fighter aircraft control, with the limitation that no frequency domain analysis is possible. In modern design, a system is represented to the greatest advantage as a set of decoupled first order differential equations defined using state variables. Nonlinear, multivariable, adaptive and robust control theories come under this division.

In state-space design, the control engineer designs a dynamic compensation by working directly with the state-variable description of the system. Because it is particularly well suited to the use of the computer techniques, state-space design is increasingly studied and used today by control engineers.

The main idea of state space comes of describing differential equations in a different method. By this way the dynamic is organized as a set of first order differential equations transformed in a vector-valued state of the system and the solution is visualized as a trajectory of this state vector space. State space control design is the technique in which the control engineer designs a dynamic compensation by working directly with the state variable description of the system.

There are several reason for studying equations in this form:

- To study more general models: The ODE do not have to be linear or stationary
- To introduce the ideas of geometry into differential equations: in physics the plane of position versus velocity of a particle is called the phase plane and the trajectory of the motion can be plotted as a curve in this plane. The main idea is to include inside the state more than two dimensions.

- To connect internal and external description: the state of dynamics system often directly describes the distribution of internal energy in the system. The internal energy can always be computed from the state variables. In contrast, the transfer function relates only the input to the output and does not show the internal behavior
- Advantage of state space design are especially apparent when the system to be controlled has more than one control input or more than one sensed output (MIMO systems).

The discussion of this paper starts from describing a dynamic system through an example passing to the state variable model, then the focus passes to the development of a state-variable equations and studying its block diagrams. Next steps are the way follow for a designer of feedback control:

1. Select closed-loop pole and develop the control law of the closed-loop system to satisfactory dynamics response.
2. Design an estimator
3. Combine the control law and the estimator
4. Introduce a reference input and track it robustly explaining the use of integral control in a state-space approach

In a way to describe the goals of feedbacks:

- obtain stability by the design
- reduce the uncertainty by the compensation
- resolve the disturbance also known as rejection
- attenuating the sensor noise

In the final chapter, to show the previous points from a practical point of view, it will be used Matlab to simulate the cart-pendulum behaviour. First of all it will be built a non linear model of the plant and studied the dynamic near its unstable equilibrium point. Then, matrices will be studied for a better comprehension of cart-pendulum poles and its controllability. A first simulation of feedback control will follow to show how poles can be set for a better trade-off between control effort and fast response; then, there will be introduced the linear quadratic regulator. The point of the observability will be touched introducing a simulation of disturbance-plant and noise measurement and how to recover the full state variables. This part is useful to put all together in order to simulate the linear quadratic gaussian in the loop-back of the control system. A classic topic is how to use reference; a simulation of input step will be explained using the equations displayed in the theoretic part. The last example of simulation involves the integral control from a state space point of view. The main strategy of state-space play around a changing of state-variables in a way to reproduce the *normal form* of state-vector equation. Last Matlab code snippet will show how to set the control robustly.

2 Preliminary fundamentals

The scope of this chapter is starting with an example of unstable system, the cart-inverted pendulum, deriving the motion equations and the relative state-space representation. It will be introduced the concept of integrator and the canonical forms related to transfer function. In LTI system exists a deep relation between time domain, frequency domain and state-space. Some properties of matrices will show how it is possible extract important information over the controllability of the system studied. The chapter ends with a quick overview of the dynamic response studied starting from the state equations.

2.1 System description with state-variables in the State Space

The motion of any finite dynamic system can be expressed as a set of first order ordinary differential equations. This is often referred to as the state-variable representation. Consider for example the motion equations of an inverted pendulum:

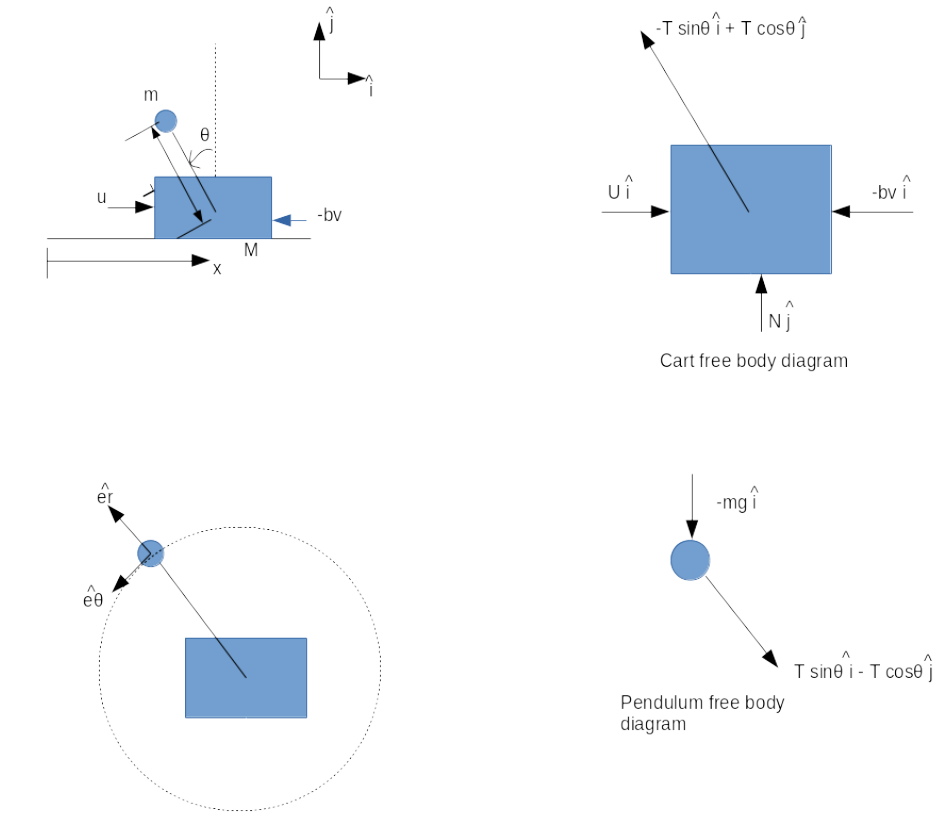

Fig. 1 - This figure shows the inverted pendulum affected by skin friction. Free body diagram of cart and pendulum and relative frame of pendulum

With the reference to to figure 1, M and m are the mass of cart and pendulum positioned to L distance from the cart mass center. The cart is affected from an external force $u\hat{i}$ and a skin friction speed proportional to the cart. The equation of the cart is one dimensional:

$$\hat{i} : u - Tsin\theta - b\dot{x} = M\ddot{x} \tag{1}$$

in order to write the pendulum equations is necessary to use a free body diagram (here "pr" stands for reference frame of the pendulum):

$$\hat{i} : Tsin\theta = ma_{px} \tag{2}$$

$$\hat{j} : -Tcos\theta - mg = ma_{py} \tag{3}$$

$$\vec{a}_p = \vec{a}_c + \vec{a}_{pr} = \ddot{x}\hat{i} + (L\ddot{\theta}\hat{e_\theta} - L\dot{\theta}^2\hat{e_r}) \tag{4}$$

Rewriting $\vec{a}_p$ over the reference frame:

$$\vec{a}_p = \ddot{x}\hat{i} - L\dot{\theta}^2(-sin\theta\hat{i} + cos\theta\hat{j}) + L\ddot{\theta}(-cos\theta\hat{i} - sin\theta\hat{j}) \tag{5}$$

It is possible put the last result in (2) and (3) to obtain:

$$\hat{i} : Tsin\theta = m\ddot{x} - mL\ddot{\theta}cos\theta + mL\dot{\theta}^2sin\theta \tag{6}$$

$$\hat{j} : -Tcos\theta - mg = -mL\ddot{\theta}sin\theta - mL\dot{\theta}^2cos\theta \tag{7}$$

then,rearranging Eqs. (1),(6),(7) to remove tension and it's obtained:

$$-mgsin\theta = m\ddot{x}cos\theta - mL\ddot{\theta} \tag{8}$$

$$u + mL\ddot{\theta}cos\theta - mL\dot{\theta}^2sin\theta = (m+M)\ddot{x} + b\dot{x} \tag{9}$$

The final step is to set $\dot{x} = v, \ddot{x} = \dot{v}, \dot{\theta} = \omega, \ddot{\theta} = \dot{\omega}$ and show 4 differential linear equations

$$\dot{x} = v \tag{10}$$

$$\dot{v} = \frac{-m^2L^2gcos(\theta)sin(\theta) + mL^2(mL\omega^2sin(\theta) - bv) + mL^2u}{mL^2(M + m(1 - cos(\theta)^2))} \tag{11}$$

$$\dot{\theta} = \omega \tag{12}$$

$$\dot{\omega} = \frac{(m+M)mgLsin(\theta) - mLcos(\theta)(mL\omega^2sin(\theta) - bv) + mLcos(\theta)u}{mL^2(M + m(1 - cos(\theta)^2))} \tag{13}$$

$$\tag{14}$$

where x is the cart position, v is the velocity, θ is the pendulum angle, ω is the angular velocity, m is the pendulum mass, M is the cart mass, L is the pendulum arm, g is the gravitational acceleration, b is a friction damping on the cart, and u is a control force applied to the cart. If we linearize the equations over a point of equilibrium $(\theta = \pi, 0)$ we can express them in a standard form:

$$\frac{d}{dt}\boldsymbol{x} = \boldsymbol{F}\boldsymbol{x} + \boldsymbol{G}u \tag{15}$$

where

$$F = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -b/M & mg/M & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{b}{ML} & \frac{(m+M)g}{ML} & 0 \end{bmatrix}, \boldsymbol{G} = \begin{bmatrix} 0 \\ \frac{1}{M} \\ 0 \\ \frac{1}{ML} \end{bmatrix}, \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} x \\ v \\ \theta \\ \omega \end{bmatrix} \tag{16}$$

The output of the system is, for example, the cart position $y = x_1 = x$ which is expressed in matrix form as

$$y = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \tag{17}$$

or,

$$y = \boldsymbol{Hx} \tag{18}$$

The equation (15) is called **state-variable form** where u is the input (the Force in this example), and the output is

$$y = \boldsymbol{Hx} + Ju \tag{19}$$

The column vector $\boldsymbol{x}$ is called **state of the system** and contain n elements for nth-order system. The quantity $\boldsymbol{F}$ is an $n\ x\ n$ **system matrix** called transition matrix, $\boldsymbol{G}$ is an $n\ x\ 1$ **input matrix**, $\boldsymbol{H}$ is a $1\ x\ n$ row matrix referred to as the **output matrix**, and J is a scalar called **direct transmission term** (please note that for the cart-pendulum example $J = 0$).

2.2 Building Block Diagrams with state-variables and State Space

The most effective way to understanding the state variable equations is via block-diagram representation. The fundamental structure of the representation is integrator, which is quite suitable for first-order, state variable representation of equations of motion for a system. Because an integrator is a device whose input is the derivative of its output, as shown in figure 2, if the output of the integrator is identified as the state,then it turns out that automatically appear the equations in state-variable form. Conversely, if a system is described by state variables, it's possible construct an analog-computer simulation of that system by taking one integrator for each state variable and connecting its input according to the given equation for that state variable as expressed in the state variable equations.

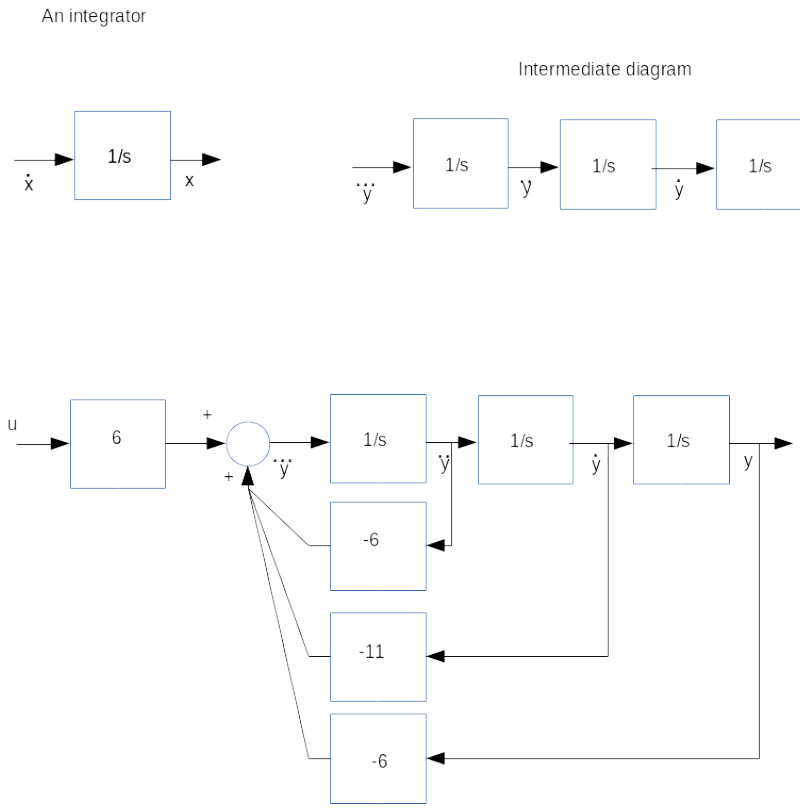

Fig. 2 - Block diagram of the system $\dddot{y} = -6\ddot{y} - 11\dot{y} - 6y + 6u$ using only integrators as dynamic elements.

In addiction of above mention it's known how to pass in Laplace domain to obtain a transfer function. In particular for the example of Fig. 2 :

$$s^3Y(s) + 6s^2Y(s) + 11sY(s) + 6Y(s) = 6U(s) \tag{20}$$

Rearranging the terms:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{6}{s^3 + 6s^2 + 11s + 6} \tag{21}$$

2.3 Block Diagrams and Canonical Forms

Transfer functions can be represented by block diagram and vice versa; we begin with a simple example:

$$G(s) = \frac{b(s)}{a(s)} = \frac{s+2}{s^2 + 7s + 12} = \frac{2}{s+4} + \frac{-1}{s+3} \tag{22}$$

The roots of the numerator $b(s)$ are the zeros of the transfer function, and the roots of the dominator $a(s)$ are the poles. Noting that $G(s)$ has been written in two form, it's possible construct a block diagram that corresponds to the transfer function using only isolated integrators as the dynamic elements. Here in figure is drawn a structure in **canonical form**

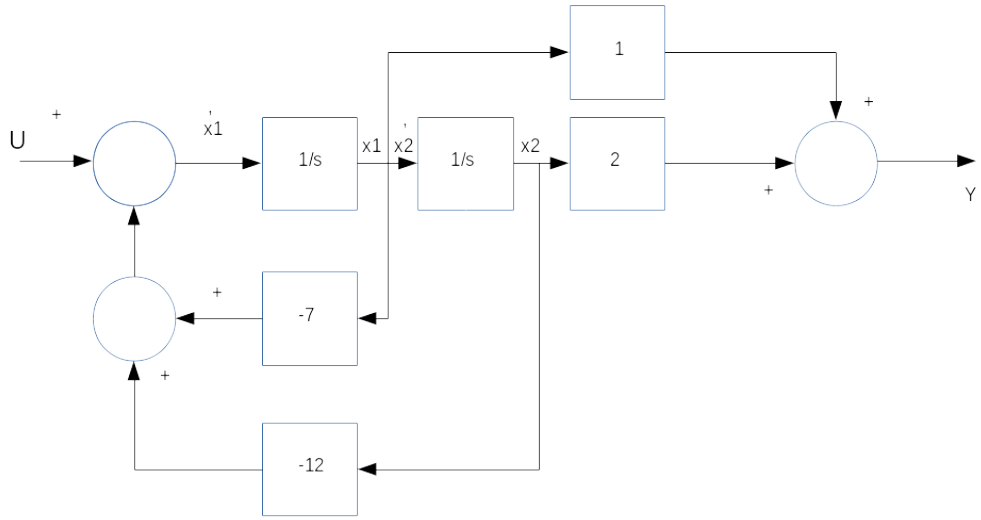

Fig. 3 - Control Block diagram of the system (22)

The transfer function (22) can be *transformed* in a couple of state-vector-space equations by using the technique explained just before to get:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}_c\boldsymbol{x} + \boldsymbol{B}_c u \\ y &= \boldsymbol{C}_c\boldsymbol{x} \end{aligned} \tag{23}$$

where

$$\begin{aligned} \boldsymbol{A}_c = \begin{bmatrix} -7 & -12 \\ 1 & 0 \end{bmatrix}, \boldsymbol{B}_c = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \\ \boldsymbol{C}_c = \begin{bmatrix} 1 & 2 \end{bmatrix}, \boldsymbol{D}_c = 0 \end{aligned} \tag{24}$$

There are 2 facts significant here: the first A_c row appear in $a(s)$ and the second one is that $b(s)$

is in C_c vector. More in general it turns out that every time it's known the transfer function it's possible by inspection write the matrices in canonical form

$$\boldsymbol{A}_c = \begin{bmatrix} -a_1 & -a_2 & .. & .. & -a_n \\ 1 & 0 & .. & .. & 0 \\ 0 & 1 & .. & .. & 0 \\ .. & .. & .. & .. & .. \\ .. & .. & .. & .. & .. \\ 0 & 0 & .. & 1 & 0 \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 1 \\ 0 \\ .. \\ .. \\ 0 \end{bmatrix}, \tag{25}$$
$$\boldsymbol{C}_c = \begin{bmatrix} b_1 & b_2 & .. & .. & b_n \end{bmatrix}, D_c = 0$$

Because (22) is written in 2 ways when it is in a partial-fraction expansion the relative block diagram form is:

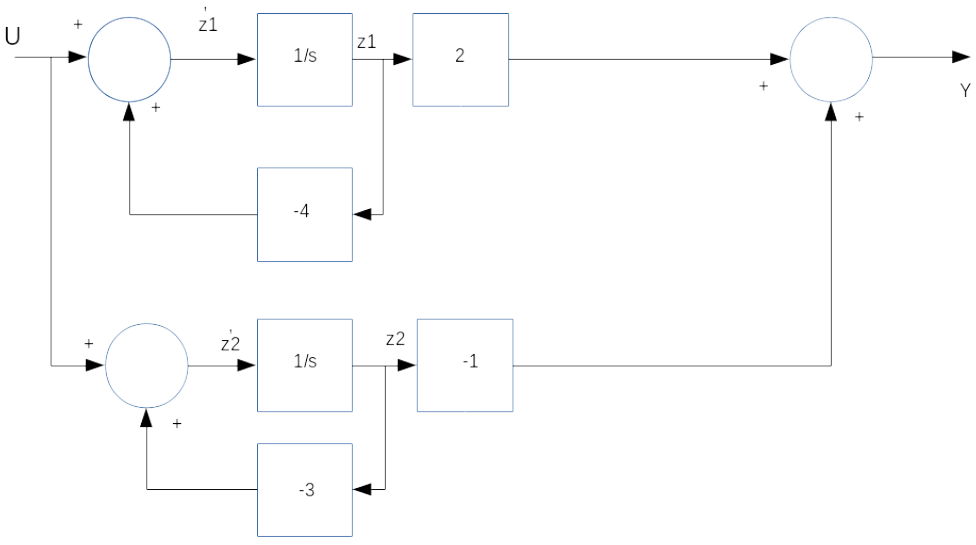

Fig. 4 - Modal Block diagram of the system

Using the same technique the matrix directly from the block diagram:

$$\begin{aligned} \dot{\boldsymbol{z}} &= \boldsymbol{A}_m \boldsymbol{z} + \boldsymbol{B}_m u \\ y &= \boldsymbol{C}_m \boldsymbol{z} + D_m u \end{aligned} \tag{26}$$

where

$$\begin{aligned} \boldsymbol{A}_m = \begin{bmatrix} -4 & 0 \\ 0 & -3 \end{bmatrix}, \boldsymbol{B}_m = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \\ \boldsymbol{C}_m = \begin{bmatrix} 2 & -1 \end{bmatrix}, \boldsymbol{D}_m = 0 \end{aligned} \tag{27}$$

and subscript m refers to **modal canonical form**. The important fact is that the system poles $(-4, -3)$ appear as the elements along the diagonal of the $\boldsymbol{A}_m$ matrix and the residues, the numerator terms in the partial fraction expansion $(2, -1)$, appear in the C_m matrix. It's important highlight that expressing a system in modal form can be harder because whenever the elements of the matrices will be complex, the poles of the system will be complex. In addiction the system matrix cannot be diagonal when the partial fraction expansion has repeated poles.

The control form is important because is connected with the controllability of a system: suppose it's known a set of state equations that describe some physical system. Is it possible to calculate the desired canonical form without obtaining the transfer functions first? It turns out this statement:

One can always transform a given state description to control canonical form if and only if the controllability matrix C is nonsingular

where the controllability matrix is:

$$\boldsymbol{\mathcal{C}} = \begin{bmatrix} \boldsymbol{G} & \boldsymbol{FG} & ... \boldsymbol{F}^{n-1}\boldsymbol{G} \end{bmatrix} \tag{28}$$

When $\mathcal{C}$ is non singular, the corresponding **F** and **G** matrices are said to be controllable. This is a technical property that usually holds for physical systems and will be important when it's considered the study of feedback. The controllability matrix $\boldsymbol{\mathcal{C}}$ and $\boldsymbol{F}$ are used in a recipe to get $\boldsymbol{T}$, the transformation matrix that links 2 set of matrices:

$$\begin{aligned} \boldsymbol{A} &= \boldsymbol{T^{-1}FT} \\ \boldsymbol{B} &= \boldsymbol{T^{-1}G} \\ \boldsymbol{C} &= \boldsymbol{HT} \\ D &= J \end{aligned} \tag{29}$$

Also state variables follows the same rules using $\boldsymbol{T}$ to connect a state vector with another:

$$\begin{aligned} \boldsymbol{x} &= \boldsymbol{Tz} \\ \dot{\boldsymbol{z}} &= \boldsymbol{Az} + \boldsymbol{B}u \end{aligned} \tag{30}$$

Another question is the effect of a state transformation. After the transformations (29) it's possible calculate the new controllability matrix $\boldsymbol{\mathcal{C}}_z$ to get at end:

$$\boldsymbol{\mathcal{C}_z} = \boldsymbol{T}^{-1}\boldsymbol{\mathcal{C}} \tag{31}$$

Read [2] for further insight.

$\boldsymbol{C_z}$ is non singular if and only if $\boldsymbol{C}$ is non singular so:

A change of state by a non singular linear transformation does not change controllability

There is a third canonical form called **observer**. Transfer function (22) for this form has matrices:

$$\begin{gathered} \boldsymbol{A}_o = \begin{bmatrix} -7 & 1 \\ -12 & -03 \end{bmatrix}, \boldsymbol{B}_o = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \\ \boldsymbol{C}_o = \begin{bmatrix} 1 & 0 \end{bmatrix}, \boldsymbol{D}_o = 0 \end{gathered} \tag{32}$$

It may be considered how change the controllability of this system as the zero at -2 is varied. For this purpose, we replace the second element 2 of $\boldsymbol{B}_o$ with the variable zero location $-z_0$ and form the controllability matrix:

$$\boldsymbol{C_x} = \begin{bmatrix} \boldsymbol{B}_0 & \boldsymbol{A}_0\boldsymbol{B}_0 \end{bmatrix} = \begin{bmatrix} 1 & -7 - z_0 \\ -z_0 & -12 \end{bmatrix} \tag{33}$$

The determinant is zero for $z_0 = -3, -4$, implying that the controllability **is lost** for these values. What does this mean? The transfer function may help:

$$G(s) = \frac{s - z_0}{(s+3)(s+4)} \tag{34}$$

If $z_0 = -3, -4$ there is a pole cancellation; when $z_0 = -3$ for example, the mode at -3 is decoupled from the input and the control is lost.

The take away of this observer canonical form are two:

- in the diagram block form, all the feedback is from the output to the state variable (see figure 5)
- the control form is always controllable for any value of the zero, while the observer form loses the controllability if the zero cancels either of the poles. These two form may represent the same transfer function, **but it may not be possible to transform the state of one to the state of the other**

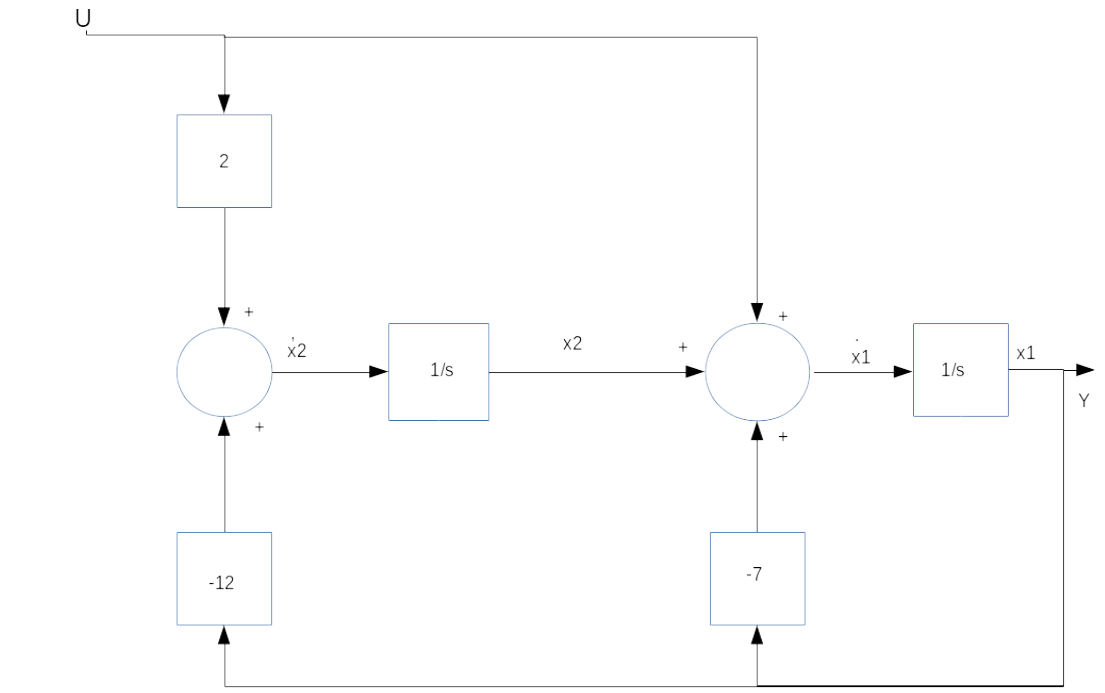

Fig. 5 - Observer canonical form of the system (22)

2.4 Dynamic response from the state equations

Having considered the structure of the state-variables equations, now it's the turn to finding the dynamic response from the state description and the relationship with poles and zeros of the transfer function. Considering (15) and (19) in a frequency domain it's possible to write by Laplace transform, the transfer function in terms of its state-space matrices:

$$G(s) = \frac{Y(s)}{U(s)} = \boldsymbol{H}(s\boldsymbol{I} - \boldsymbol{F})^{-1}\boldsymbol{G} + J \tag{35}$$

Because (35) expresses the transfer function in terms of the general state-space descriptor matrices $\boldsymbol{F}, \boldsymbol{G}, \boldsymbol{H}, J$, it's possible to express poles and zeros in term of these matrices. Manipulating the state equation like an eigenvalue problem it turns out that find poles is equal to solve :

$$det(p_i\boldsymbol{I} - \boldsymbol{F}) = 0 \tag{36}$$

These equations show again that the *poles* of the transfer function are the eigenvalues of the system matrix $\boldsymbol{F}$. The determinant (36) is known as the **characteristic equation**

We can also determine the zeros of a system from the state variable description matrices $\boldsymbol{F}, \boldsymbol{G}, \boldsymbol{H}, J$ using a system theory point of view. From this perspective, a zero is a value of a generalized frequency s such that the system can have a non-zero input and state yet have an output of zero. If the input at zero frequency z_i is given by

$$u(t) = u_0 e^{z_i t} \tag{37}$$

then the output is identically zero : $y(t) \equiv 0$

Again using (19) it's possible obtain:

$$y = \boldsymbol{H}\boldsymbol{x} + Ju = \boldsymbol{H}e^{z_i t}\boldsymbol{x}_0 + Ju_0 e^{z_i t} = 0 \tag{38}$$

Combining the above equations it's possible obtain a state vector equation similar for the poles problem:

$$\begin{bmatrix} z_i\boldsymbol{I} - \boldsymbol{F} & -\boldsymbol{G} \\ \boldsymbol{H} & J \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_0 \\ u_0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{0} \\ 0 \end{bmatrix} \tag{39}$$

In SISO systems the equation just mentioned has a square matrix and the solution is equivalent to :

$$det \begin{bmatrix} z_i\boldsymbol{I} - \boldsymbol{F} & -\boldsymbol{G} \\ \boldsymbol{H} & J \end{bmatrix} = 0 \tag{40}$$

Equations (36) and (40) can be conbined to express the transfer function in a compact form from state-description matrices:

$$G(s) = \frac{det \begin{bmatrix} s\boldsymbol{I} - \boldsymbol{F} & -\boldsymbol{G} \\ \boldsymbol{H} & J \end{bmatrix}}{det \left[s\boldsymbol{I} - \boldsymbol{F} \right]} \tag{41}$$

Having done a brief introduction about the connection between State Space and transfer function of a plant is worthwhile closes the discussion related three ways to represent a dynamic system when it's represented by a LTI system.
They are studied via impulse response, state space and transfer functions because have the same meaning in different domains. The figure below shows the relationship: there are three equivalent representations: 1) time-domain, in terms of the impulse response; 2) frequency domain, in terms of the transfer function; and 3) state-space, in terms of a system of differential equations.

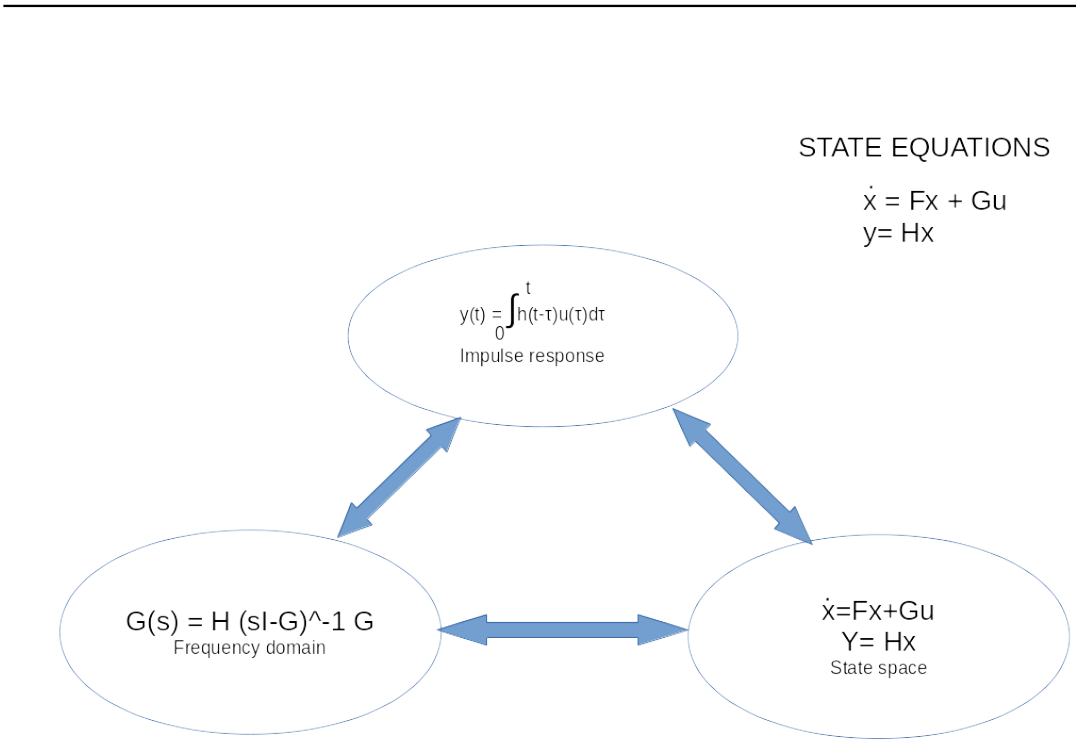

Fig. 6 - Three equivalent representations of Linear Invariant Systems

3 Control-law design for full-state feedback

The purpose of the control law is to allow the designer to assign as set of poles locations for the close-loop system that will correspond to satisfactory dynamic response in term of rise time and other measures of transient response. It's assumed for feedback purpose that all the elements of the state vector are at our disposal. In practice this assumption is ridiculous because it's known from design methods that rarely there are so many sensors. The assumption that all the state variables are available allows to proceed with this first step. When there will be introduced the reference in the system control, it will be used again in didactic purpose to shown the finality of tracking it, but the results will be almost identical by introducing noise inside the system to approach the reality.

3.1 Finding the control law

How does it change the state equation when the system is inside a control loop? Below it's show the figure of a design block scheme:

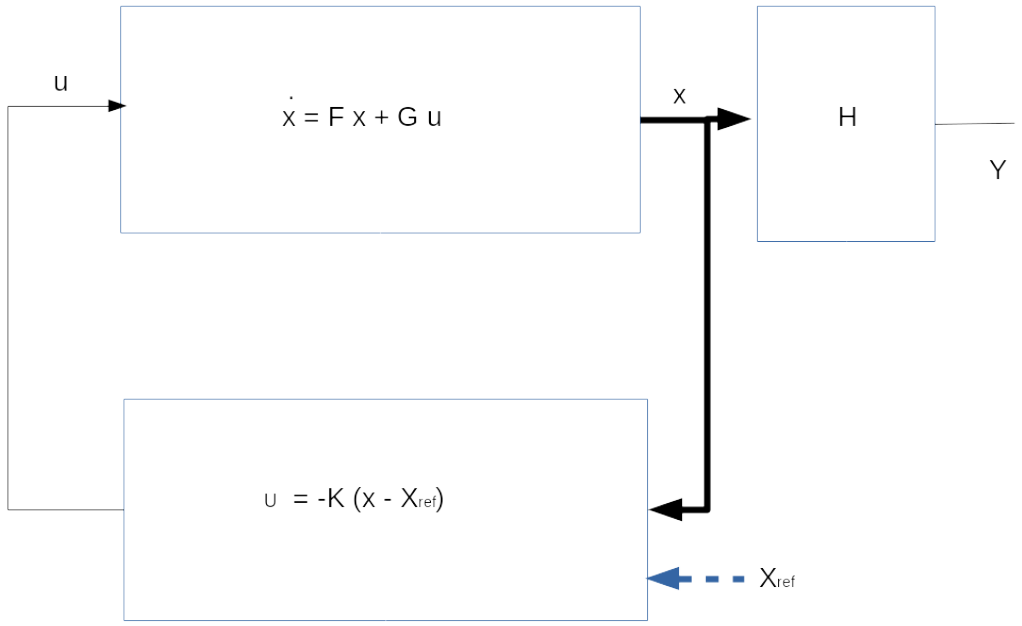

Fig. 7 - Assumed system for control law design

$$u = -\boldsymbol{K}\boldsymbol{x} = \begin{bmatrix} K_1 & K_2 & ...K_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_n \end{bmatrix} \tag{42}$$

Equation (42) tell us that the system has a constant matrix in the state-vector feedback path gains, $K1, K2, ...K_n$, and because there are n roots of the system, it's possible that there are enough degree of freedom to select arbitrarily and desired root location by choosing the proper value of K_i. Substituting the feedback law given in (42) into the system described by (15) yields:

$$\dot{\boldsymbol{x}} = \boldsymbol{F}\boldsymbol{x} - \boldsymbol{G}\boldsymbol{K}\boldsymbol{x} = (\boldsymbol{F} - \boldsymbol{G}\boldsymbol{K})\boldsymbol{x} \tag{43}$$

It turns out an eigenvalues problem that it can be solved if and only if

$$det[s\boldsymbol{I} - (\boldsymbol{F} - \boldsymbol{G}\boldsymbol{K})] = 0 \tag{44}$$

When evaluate, this yield an nth order polynomial in s containing the gains **K** so the roots of (44) are in the *desirable* locations. We assume that desired locations are known, say,

$$s = s_1, s_2, ..., s_n \tag{45}$$

Then the corresponding desired (control) characteristic equation is

$$\alpha_c(s) = (s - s_1)(s - s_2)....(s - s_n) = 0 \tag{46}$$

Hence the required elements of **K** are obtained by matching coefficients (44) and (46) ,forcing the systems's characteristic equation to be identical to the desired characteristic equation and the closed loop poles at the desired locations.

Calculating the gains by using this technique becomes rather tedious when the order of the system is higher than 3. However there is an alternative to this method called **Ackerman's formula** divided in three steps process of converting to $(\boldsymbol{F}_c\boldsymbol{G}_c)$ solving for the gains, and converting back again into the very compact form:

$$\boldsymbol{K} = [0, 0, ...0, 1]\mathcal{C}^{-1}\alpha_c(\boldsymbol{F}) \tag{47}$$

where $\mathcal{C}$ is the controllability matrix in (28),and :

$$\alpha_c(\boldsymbol{F}) = \boldsymbol{F}^n + \alpha_1\boldsymbol{F}^{n-1} + ... + \alpha_n\boldsymbol{I} \tag{48}$$

where the α_i are the coefficient corresponding to the desired poles locations, and n gives the order of the system and the dimension of the state variables vector.

3.2 How to select pole locations for a good design

As shown before, when selecting pole location it is always useful to keep in mind that control effort required is related to how far the open loop poles are moved by the feedback. Furthermore, when a zero is near a pole, the system may be nearly uncontrollable. The designer philosophy will take in account to fix only the undesirable aspects of the open loop response and avoids either large increases bandwidth or efforts to move poles that are near zeros will typically allow smaller gains, and thus smaller controls actuators. There are 2 main ways to overcome the problem. First deals dominant second order poles, second is Linear Quadratic Regulator; in this paper it will be show the second because more suited with state variables in feedback controls for higher dimension, for further reading [2] explains a technique with dominant second order.

3.2.1 Linear quadratic regulator

We have seen in the previous sections that if (F, G) is controllable, then it is possible to arbitrarily manipulate the eigenvalues of the closed-loop system $(\boldsymbol{F} - \boldsymbol{GK})$ through the choice of a full-state feedback control law $u = -\boldsymbol{Kx}$. This implicitly assumes that full-state measurements are available (i.e., $H = I$ and $J = 0$, so that $\boldsymbol{y} = \boldsymbol{x}$). Given a controllable system, and either measurements of the full-state or an observable system with a full-state estimate, there are many choices of stabilizing control laws $u = -\boldsymbol{Kx}$. It is possible to make the eigenvalues of the closed-loop system

($\boldsymbol{F} - \boldsymbol{GK}$) arbitrarily stable, placing them as far as desired in the left-half of the complex plane. However, overly stable eigenvalues may require exceedingly expensive control expenditure and might also result in actuation signals that exceed maximum allowable values. Choosing very stable eigenvalues may also cause the control system to over-react to noise and disturbances, like a new driver will over-react to vibrations in the steering wheel, causing the closed-loop system to jitter. Over stabilization can counter-intuitively degrade robustness and may lead to instability if there are small time delays or unmodeled dynamics. Choosing the best gain matrix $\boldsymbol{K}$ to stabilize the system without expending too much control effort is an important goal in optimal control. A balance must be struck between the stability of the closed-loop system and the aggressiveness of control. It is important to take control expenditure into account:

1. to prevent the controller from over-reacting to high-frequency noise and disturbances
2. that actuation does not exceed maximum allowed amplitudes
3. that control is not prohibitively expensive.

In particular, the cost function

$$J(t) = \int_0^t \boldsymbol{x}(\tau)^* \boldsymbol{Q} x(\tau) + \boldsymbol{u}(\tau)^* \boldsymbol{R} u(\tau) d\tau \tag{49}$$

balances the cost of effective regulation of the state with the cost of control. The matrices $\mathbf{Q}$ and $\mathbf{R}$ weight the cost of deviations of the state from zero and the cost of actuation, respectively. The matrix $\mathbf{Q}$ is positive semi-definite, and $\mathbf{R}$ is positive definite; these matrices are often diagonal, and the diagonal elements may be tuned to change the relative importance of the control objectives. Adding such a cost function makes choosing the control law a well-posed optimization problem. The linear-quadratic-regulator (LQR) control law $u = -\boldsymbol{K}_r \boldsymbol{x}$ is designed to minimize $J = \lim_{t \to \infty} J(t)$. LQR is so-named because it is a linear control law, designed for a linear system, minimizing a quadratic cost function, that regulates the state of the system to $J = \lim_{t \to \infty} \boldsymbol{x}(t) = \mathbf{0}$. Because the cost-function in (49) is quadratic, there is an analytical solution for the optimal controller gains K_r , given by

$$\boldsymbol{K}_r = \boldsymbol{R}^{-1} \boldsymbol{G}^* \boldsymbol{X} \tag{50}$$

where $\boldsymbol{X}$ is the solution to an algebraic Riccati equation:

$$\boldsymbol{F}^* \boldsymbol{X} + \boldsymbol{X} \boldsymbol{F} - \boldsymbol{X} \boldsymbol{G} \boldsymbol{R}^{-1} \boldsymbol{G}^* \boldsymbol{X} + \boldsymbol{Q} = \mathbf{0} \tag{51}$$

4 Estimator design with state variables

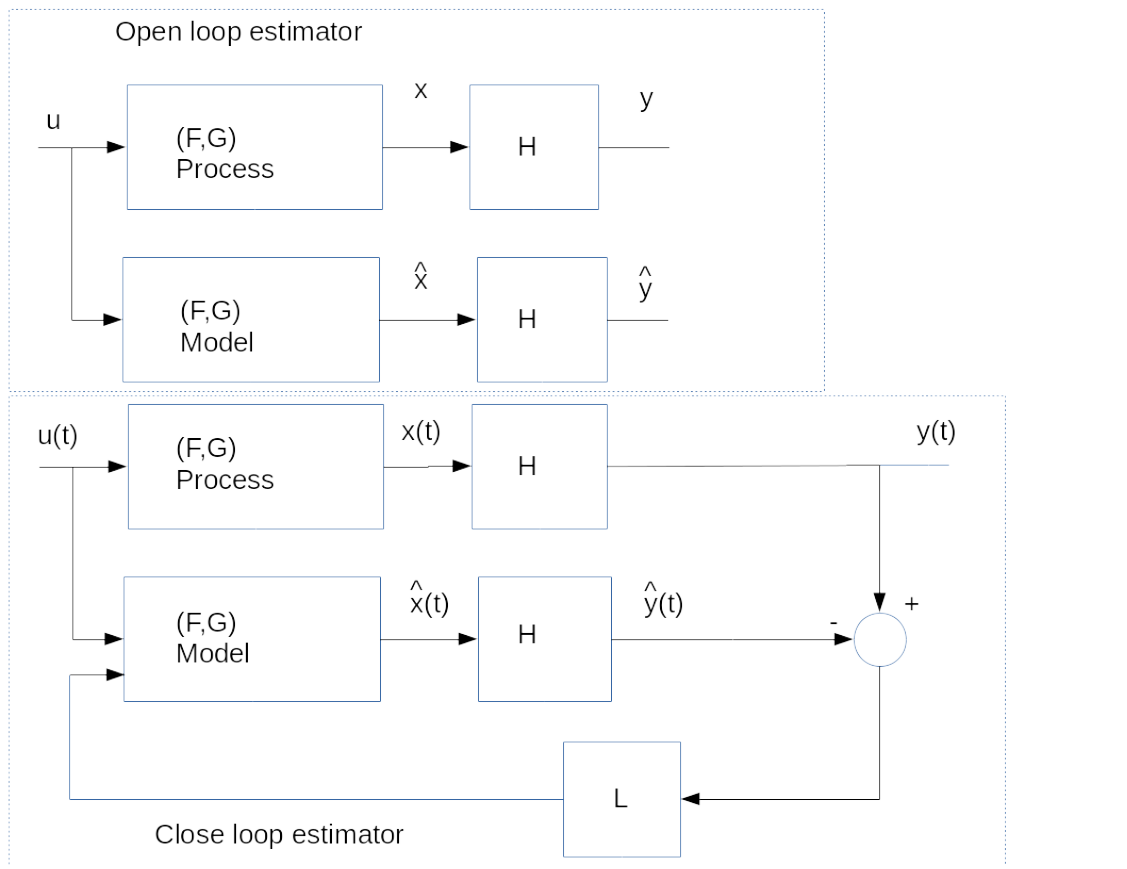

Fig 8 Open and close loop estimator block scheme

The control law assumes that all the state variables are available for feed back. In most cases, not all the state variables are measured. The cost of the required sensors may be prohibitive, or it may be physically impossible to measure all of the state variables. One method of estimating the state is to construct a full order model of the plant dynamics; by observing the above open-loop block scheme it's possible write:

$$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{F}\hat{\boldsymbol{x}} + \boldsymbol{G}u \tag{52}$$

where $\hat{\boldsymbol{x}}$ is the estimate of the actual state $\boldsymbol{x}$. To study the dynamics of this estimator, we define the error in the estimate to be:

$$\tilde{\boldsymbol{x}} = \boldsymbol{x} - \hat{\boldsymbol{x}} \tag{53}$$

Then the dynamics of this error system are given by:

$$\dot{\tilde{\boldsymbol{x}}} = \boldsymbol{F}\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}(0) = \boldsymbol{x}(0) - \hat{\boldsymbol{x}}(0) \tag{54}$$

The error convergences to zero for a stable system (**F** stable), but we have no ability to influence the rate at which the state estimate converges to the true state. Furthermore, the error is converging to zero at the same rate as the natural dynamics of **F**. If this convergence rate were satisfactory, no control or estimation would be required. The golden rule is: when in trouble use feedback. Looking the close loop observer block scheme, consider the feedback difference between the measured and estimate outputs and correcting the model continuously with this error signal. The equation for this scheme is:

$$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{F}\hat{\boldsymbol{x}} + \boldsymbol{G}u + \boldsymbol{L}(y - \boldsymbol{H}\hat{\boldsymbol{x}}) \tag{55}$$

Here **L** is a proportional gain defined as:

$$\boldsymbol{L} = [l_1, l_2, ..., l_n]^T \tag{56}$$

and is chosen to achieve satisfactory error characteristics. The dynamics of the error can be obtained by subtracting the estimate (55) from (15) to get the error equation:

$$\dot{\tilde{\boldsymbol{x}}} = (\boldsymbol{F} - \boldsymbol{L}H)\tilde{\boldsymbol{x}} \tag{57}$$

and the characteristic equation of the error is now given by:

$$det[s\boldsymbol{I} - (\boldsymbol{F} - \boldsymbol{LH})] = 0 \tag{58}$$

If we can choose $\boldsymbol{L}$ so that $\boldsymbol{F} - \boldsymbol{LH}$ has stable and reasonably fast eigenvalues, then $\tilde{\boldsymbol{x}}$ will decay to zero and remain there independent of the known forcing function $u(t)$ and its effect on the state $x(t)$ and irrespective on the initial condition $\tilde{\boldsymbol{x}}(0)$. This means that $\hat{\boldsymbol{x}}(t)$ will converge to $\boldsymbol{x}(t)$, regardless of the value of $\hat{\boldsymbol{x}}(0)$. However we can typically choose $\boldsymbol{L}$ so that the error system is still at least stable and the error remains acceptably small, even with modeling errors and disturbing inputs. The selection of $\boldsymbol{L}$ can be approached in exactly the same fashion as $\boldsymbol{K}$ is selected in the control law design. If we specify the desired location of the estimator error poles as:

$$s_i = \beta_1, \beta_2, ...\beta_n \tag{59}$$

then the desired estimator characteristic equation is:

$$\alpha_e(s) = (s - \beta_1)(s - \beta_2)....(s - \beta_n) \tag{60}$$

We can then solve for $\boldsymbol{L}$ by comparing coefficients in Eqs. (58) and (60).

4.1 The observability

As begun in the introduction it's time to spent some raw talking about the **observability** of state variables. In a development exactly parallel with the control-law case, we can find a transformation to take a given system to observer canonical form if and only if the system has structural properties that in this case we call as mention just like above. Roughly speaking, observability refers to our ability to deduce information about all the modes of the system by monitoring only

the sensed outputs. Not observability results when some mode or subsystem is disconnected physically from the output and therefore no longer appears in the measurements. The mathematical test for determining observability is that the **observability matrix**:

$$\mathcal{O} = \begin{bmatrix} \boldsymbol{H} \\ \boldsymbol{HF} \\ .. \\ .. \\ \boldsymbol{HF}^{n-1} \end{bmatrix} \tag{61}$$

must have independent columns. In general, we can find a transformation to observer canonical form if and only if the observability matrix is nonsingular (full rank).Note that is analogous to our earlier conclusion for transforming system matrices to control canonical form. As with control law design, we could find the transformation to observer form, compute the gains from the equivalent of $\boldsymbol{F}_o - \boldsymbol{L}\boldsymbol{H}_o$ and transform back. An alternative method of computing $\boldsymbol{L}$ is to use the Ackermann's formula in estimator form:

$$\boldsymbol{L} = \alpha_e(\boldsymbol{F})\mathcal{O}^{-1} \begin{bmatrix} 0 \\ 0 \\ .. \\ .. \\ 1 \end{bmatrix} \tag{62}$$

where $\mathcal{O}$ is the observability matrix given in (61). once again $\alpha_e(\boldsymbol{F})$ follows the rule just mentioned in the control law section.

4.2 Estimator poles selection. The optimal full-state estimation

In comparison with the selection of controllers poles, estimator poles selection require to be concerned with a much different relationship than with control effort. As in controller, there is a feedback term in the estimator that grows in magnitude as the requested speed of response increases. However this feedback is in the form of an electronic signal or a digital word in a computer, so its growth causes no special difficulty. The important consequence of increasing the speed of response of an estimator is that the bandwidth of the estimator becomes higher, thus causing more sensor noise to pass on to the control actuator. Like the controller the estimator design (and the poles selection) is a balance between good transient response and low-enough bandwidth that sensor noise does not significantly impact the actuator activity.
When deriving the optimal full-state estimator, it is necessary to re-introduce disturbances to the state, $\boldsymbol{w}_d$, and sensor noise, $\boldsymbol{w}_n$:

$$\dot{\boldsymbol{x}} = \boldsymbol{F}\boldsymbol{x} + \boldsymbol{G}\boldsymbol{u} + \boldsymbol{w}_d \tag{63}$$
$$\boldsymbol{y} = \boldsymbol{H}x + \boldsymbol{J}u + \boldsymbol{w}_n \tag{64}$$

By hypothesis is assumed that disturbance and noise are zero-mean Gaussian processes with known covariances:

$$E(\boldsymbol{w}_d(t)\boldsymbol{w}_d(\tau)^*) = Q\delta(t-\tau) \tag{65}$$
$$E(\boldsymbol{w}_n(t)\boldsymbol{w}_n(\tau)^*) = R\delta(t-\tau) \tag{66}$$

Here E is the expected value and $\delta(.)$ is the Dirac delta function. The matrices Q and R are positive semi-definite with entries containing the covariances of the disturbance and noise terms. It is possible to obtain an estimate $\hat{\boldsymbol{x}}$ of the full-state $\boldsymbol{x}$ from measurements of the input $\boldsymbol{u}$ and output $\boldsymbol{y}$, via the following estimator dynamical system:

$$\dot{\hat{\boldsymbol{x}}} = \boldsymbol{F}\hat{\boldsymbol{x}} + \boldsymbol{G}u + \boldsymbol{L}(\boldsymbol{y} - \hat{\boldsymbol{y}}) \tag{67}$$
$$\hat{\boldsymbol{y}} = \boldsymbol{H}\hat{\boldsymbol{x}} + \boldsymbol{J}u \tag{68}$$

The matrices $\boldsymbol{F}, \boldsymbol{G}, \boldsymbol{H}, \boldsymbol{J}$ are obtained from the system model and the filter gain $\boldsymbol{L}$ is determined via a similar procedure as in LQR. $\boldsymbol{L}$ is given by:

$$\boldsymbol{L} = \boldsymbol{Y}\boldsymbol{H}^*\boldsymbol{R} \tag{69}$$

where $\boldsymbol{Y}$ is the solution to another algebraic Riccati equation:

$$\boldsymbol{Y}F^* + \boldsymbol{F}Y - \boldsymbol{Y}\boldsymbol{H}^*\boldsymbol{R}^{-1}\boldsymbol{H}\boldsymbol{Y} + \boldsymbol{Q} = \boldsymbol{0} \tag{70}$$

Refer to [1] for insights.

This solution is commonly referred to as the Kalman filter, and it is the optimal full-state estimator with respect to the following cost function:

$$J = \lim_{t\to\infty} E[\tilde{x}(t)^*\tilde{x}(t)] \tag{71}$$

This cost function implicitly includes the effects of disturbance and noise, which are required to determine the optimal balance between aggressive estimation and noise attenuation. Thus, the Kalman filter is referred to as linear quadratic estimation (LQE), and has a dual formulation to the LQR optimization. The estimator dynamical system is expressed in terms of the estimate $\hat{\boldsymbol{x}}$ with inputs $\boldsymbol{y}$ and $\boldsymbol{u}$. If the system is observable it is possible to place the eigenvalues of $\boldsymbol{F} - \boldsymbol{L}\boldsymbol{H}$ arbitrarily with choice of $\boldsymbol{L}$. When the eigenvalues of the estimator are stable, then the state estimate $\hat{\boldsymbol{x}}$ converges to the full-state $\boldsymbol{x}$ asymptotically, as long as the model faithfully captures the true system dynamics. To see this convergence, consider the dynamics of the estimation error $\boldsymbol{\epsilon} = \boldsymbol{x} - \hat{\boldsymbol{x}}$:

$$\begin{aligned}\dot{\boldsymbol{\epsilon}} &= \dot{\boldsymbol{x}} - \dot{\hat{\boldsymbol{x}}} \\ &= [\boldsymbol{F}\boldsymbol{x} + \boldsymbol{G}\boldsymbol{u} + \boldsymbol{w}_d] - [(\boldsymbol{F} - \boldsymbol{L}\boldsymbol{H})\hat{\boldsymbol{x}} + \boldsymbol{L}\boldsymbol{y} + (\boldsymbol{G} - \boldsymbol{L}\boldsymbol{J})\boldsymbol{u}] \\ &= (\boldsymbol{F} - \boldsymbol{L}\boldsymbol{H})\epsilon + \boldsymbol{w}_d - \boldsymbol{L}\boldsymbol{w}_n\end{aligned} \tag{72}$$

where it has been used (64) in substitution of $\boldsymbol{y}$.

Therefore, the estimate $\hat{\boldsymbol{x}}$ will converge to the true full state when $\boldsymbol{F} - \boldsymbol{LH}$ has stable eigenvalues. As with LQR, there is a trade-off between over-stabilization of these eigenvalues and the amplification of sensor noise. This is similar to the behavior of an inexperienced driver who may hold the steering wheel too tightly and will overreact to every minor bump and disturbance on the road.

4.3 Optimal sensor-based control: Linear quadratic Gaussian (LQG)

The full-state estimate from the Kalman filter is generally used in conjunction with the full-state feedback control law from LQR, resulting in optimal sensor- based feedback. Remarkably, the LQR gain K_r and the Kalman filter gain K_f may be designed separately, and the resulting sensor-based feedback will re- main optimal and retain the closed-loop eigenvalues when combined. Combining the LQR full-state feedback with the Kalman filter full-state estimator results in the linear-quadratic Gaussian (LQG) controller. The LQG controller is a dynamical system with input y, output u, and internal state $\hat{\boldsymbol{x}}$:

$$\begin{aligned} \frac{d}{dt}\hat{\boldsymbol{x}} =& (\boldsymbol{F} - \boldsymbol{K}_f\boldsymbol{H} - \boldsymbol{G}\boldsymbol{K}_r)\hat{\boldsymbol{x}} + \boldsymbol{K}_f\boldsymbol{y} \\ \boldsymbol{u} =& -\boldsymbol{K}_r\hat{\boldsymbol{x}} \end{aligned} \tag{73}$$

The LQG controller is optimal with respect to the following ensemble-averaged version of the cost function from (49):

$$J(t) = \left\langle \int_0^t [\boldsymbol{x}(\tau)^*\boldsymbol{Q}\boldsymbol{x}(\tau) + \boldsymbol{u}(\tau)^*\boldsymbol{R}\boldsymbol{u}(\tau)d\tau] \right\rangle. \tag{74}$$

The controller $\boldsymbol{u} = \boldsymbol{K}_r\hat{\boldsymbol{x}}$ is in terms of the state estimate, and so this cost function must be averaged over many realizations of the disturbance and noise. Applying LQR to $\hat{\boldsymbol{x}}$ results in the following state dynamics:

$$\begin{aligned} \frac{d}{dt}\boldsymbol{x} =& \boldsymbol{F}\boldsymbol{x} - \boldsymbol{G}\boldsymbol{K}_r\hat{\boldsymbol{x}} + \boldsymbol{w}_d \\ =& \boldsymbol{F}\boldsymbol{x} - \boldsymbol{G}\boldsymbol{K}_r\boldsymbol{x} + \boldsymbol{G}\boldsymbol{K}_r(\boldsymbol{x} - \hat{\boldsymbol{x}}) + \boldsymbol{w}_d \\ =& \boldsymbol{F}\boldsymbol{x} - \boldsymbol{G}\boldsymbol{K}_r\boldsymbol{x} + \boldsymbol{G}\boldsymbol{K}_r\boldsymbol{\epsilon} + \boldsymbol{w}_d \end{aligned} \tag{75}$$

Again $\boldsymbol{\epsilon} = \boldsymbol{x} - \hat{\boldsymbol{x}}$ as before. Finally the closed-loop system may be written as

$$\frac{d}{dt}\begin{bmatrix}\boldsymbol{x} \\ \boldsymbol{\epsilon}\end{bmatrix} = \begin{bmatrix}\boldsymbol{F} - \boldsymbol{G}\boldsymbol{K}_r & \boldsymbol{G}\boldsymbol{K}_r \\ \boldsymbol{0} & \boldsymbol{F} - \boldsymbol{K}_f\boldsymbol{H}\end{bmatrix}\begin{bmatrix}\boldsymbol{x} \\ \boldsymbol{\epsilon}\end{bmatrix} + \begin{bmatrix}\boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{I} & -\boldsymbol{K}_f\end{bmatrix}\begin{bmatrix}\boldsymbol{w}_d \\ \boldsymbol{w}_n\end{bmatrix} \tag{76}$$

Thus, the closed-loop eigenvalues of the LQG regulated system are given by the **eigenvalues** of $\boldsymbol{F} - \boldsymbol{G}\boldsymbol{K}_r$ and $\boldsymbol{F} - \boldsymbol{K}_f\boldsymbol{H}$, which were optimally chosen by the LQR and Kalman filter gain matrices,

respectively. This in control theory is called **the separation principle** which essentially means it's possible design the LQR and LQE separately and when they are combined together the system retains the desirable properties of each of them.

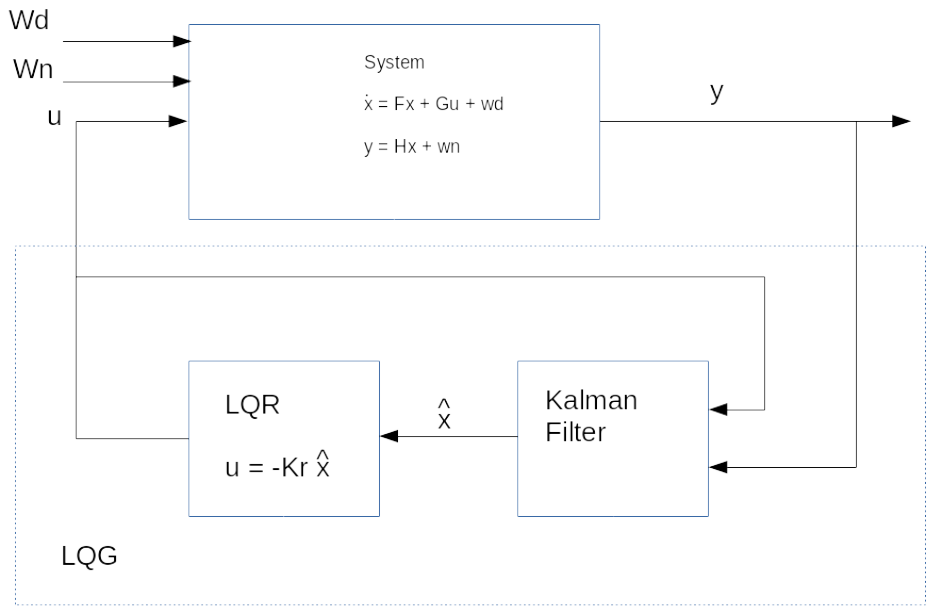

Fig 9 Schematic illustrating the linear quadratic Gaussian (LQG) controller for optimal closed-loop feedback based on noisy measurements y. The optimal LQR and Kalman filter gain matrices K_r and K_f may be designed independently, based on two different algebraic Riccati equations. When combined,the resulting sensor-based feedback remains optimal.

It's worthwhile to mention that LQG framework relies on an accurate model of the system and knowledge of the magnitudes of the disturbances and measurement noise, which are assumed to be Gaussian processes. In real-world systems, each of these assumptions may be invalid, and even small time delays and model uncertainty may destroy the robustness of LQG and result in instability. The lack of robustness of LQG regulators to model uncertainty motivates the introduction of *robust control*. For example, it is possible to robustify LQG regulators through a process known as *loop-transfer recovery*. However, despite robustness issues, LQG control is extremely effective for many systems, and is among the most common control paradigms.

5 Reference input and robust tracking control with state-variables

It's time to add a reference input in the feedback system. Once again, by state-variables method it will shown how the use of gain block is suited to get best performance when a unit step is in input. The choice of a reference gain will result in zero steady-state error to a step command, but it won't be robust because any change in the plant parameters will cause the error to be nonzero. To obtain robust control it's needed to use integral block. In the state-space design methods till now, no

mention has been made of integral control and no design examples have produces a compensation containing an integral term. After the reference topic it's going to introduce a general method for robust tracking that will present the internal model principle, which solve an entire class of tracking problems and disturbance rejection controls.

5.1 Introducing a reference input with full state feedback

In order to study the transient response of the pole placement designs to input commands, it's necessary to introduce the reference input into the system. An obvious way to do this is to change the control to $u = -\boldsymbol{K}\boldsymbol{x} + r$. However, the system will now almost surely have a non zero steady-state error to a step input. If the desired final values of the state and the control input are $\boldsymbol{x}_{ss}$ and u_{ss} respectively, then the new control formula should be

$$u = u_{ss} - \boldsymbol{K}(\boldsymbol{x} - \boldsymbol{x}_{ss}) \tag{77}$$

so that when $\boldsymbol{x} = \boldsymbol{x}_{ss}$ (no error), $u = u_{ss}$. To pick the correct final values, we must solve the equations so that the system will have zero steady-state error to any constant input. The system differential equations are defined by (15) and (19); in the constant steady state they are reduced to the pair:

$$\boldsymbol{0} = \boldsymbol{F}\boldsymbol{x}_{ss} + \boldsymbol{G}u_{ss} \tag{78}$$

$$y_{ss} = \boldsymbol{H}\boldsymbol{x}_{ss} + Ju_{ss} \tag{79}$$

We want to solve for values for which $y_{ss} = r_{ss}$. To do this, we make $\boldsymbol{x_{ss}} = \boldsymbol{N}_x r_{ss}$ and $u_{ss} = N_u r_{ss}$. With these substitution it turns out a matrix equation solvable for $\boldsymbol{N}_x$ and N_u. Putting the result in (77), it's obtained

$$u = N_u - \boldsymbol{K}(\boldsymbol{x} - \boldsymbol{N}_x r) = -\boldsymbol{K}\boldsymbol{x} + (N_u + \boldsymbol{K}\boldsymbol{N}_x)r \tag{80}$$

The coefficient of r in parentheses is a constant that can be computed beforehand. We give it the symbol $\bar{N}$ so

$$u = -\boldsymbol{K}\boldsymbol{x} + \bar{N}r \tag{81}$$

With the reference input in place, the close loop system has input r and output y. From the state description we know that the system poles are at the eigenvalues of the close loop system matrix $\boldsymbol{F} - \boldsymbol{G}\boldsymbol{K}$. In order to compute the close loop transient response, it's necessary to know where the close loop zeros of the transfer function from r to y are.They are to be found applying (40) to the close loop description, which it's assumed don't have direct path from input u to output y, so that $J = 0$. The zeros are values of s such that

$$det \begin{bmatrix} s\boldsymbol{I} - (\boldsymbol{F} - \boldsymbol{G}\boldsymbol{K}) & -\bar{N}\boldsymbol{G} \\ \boldsymbol{H} & 0 \end{bmatrix} = 0 \tag{82}$$

It's possible assume two elementary facts about determinants to simplify (82): in the first place if we divide the last column by $\bar{N}$, then the point where the determinant is zero remains unchanged. The second is that the determinant also not change if it's multiplied the last column by $\boldsymbol{K}$ and add it to the first block column; the result is that the $\boldsymbol{GK}$ term is canceled out. Thus the matrix equation for zeros reduces to

$$det \begin{bmatrix} s\boldsymbol{I} - \boldsymbol{F} & -\boldsymbol{G} \\ \boldsymbol{H} & 0 \end{bmatrix} = 0 \tag{83}$$

The above equation is the same as (40) for the zeros of the plant *before* the feedback was applied. The important conclusion is:

When full-state feedback is used as in (80) or (81), the zeros remain unchanged by the feedback

5.2 Robust Tracking Control: The error space Approach

The problem of tracking r and rejecting the disturbance can be seen as an exercise in design a control law to provide *regulation of the error*, which is to say that the error e tends to zero as time gets large. The control must also be **structurally stable** or **robust**, in the sense that regulation of e to zero in the steady-state occurs even in the presence of "small" perturbations of the original system parameters. Note that, in practice, it's impossible to have a perfect model of the plant, and the values of parameters are virtually always subject to some change, so robustness is always very important.

Suppose we have a system equations

$$\dot{\boldsymbol{x}} = \boldsymbol{F}\boldsymbol{x} + \boldsymbol{G}u + \boldsymbol{G}_1 w \tag{84}$$
$$y = \boldsymbol{H}\boldsymbol{x} \tag{85}$$

and a reference signal that is known to satisfy a specific differential equation. Plant disturbances of the same class may also be present. The goal is to design a controller for this system so that the close-loop system will have specified poles, and can also track input command signals, and reject disturbances of the type described without steady-state error. A solution is developed only for second order differential equation. Let's define the reference input to satisfy the relation

$$\ddot{r} + \alpha_1 \dot{r} + \alpha_2 r = 0 \tag{86}$$

and the disturbance to satisfy exactly the same equation:

$$\ddot{w} + \alpha_1 \dot{w} + \alpha_2 w = 0 \tag{87}$$

Defining the tracking error as $e = y - r$, enclosing the second order equations for $\boldsymbol{x}$ and $\boldsymbol{u}$ in two new variables ξ and μ, it's possible to write new standard state variable equation into the state-space error:

$$\dot{\boldsymbol{z}} = \boldsymbol{A}\boldsymbol{z} + \boldsymbol{B}\mu \tag{88}$$

where $\boldsymbol{z} = \begin{bmatrix} e & \dot{e} & \xi^T \end{bmatrix}^T$ and :

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 & \boldsymbol{0} \\ -\alpha_2 & -\alpha_1 & \boldsymbol{H} \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{F} \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{G} \end{bmatrix} \tag{89}$$

Having back the system into a canonical form it may be asked about the controllability. It turns out that the plant $(\boldsymbol{F}, \boldsymbol{G})$ is controllable and does not have a zero at any of the roots of the reference-signal characteristic equation

$$\alpha_r(s) = s^2 + \alpha_1 s + \alpha_2 \tag{90}$$

then the error system $(\boldsymbol{A}, \boldsymbol{B})$ is controllable. Assuming these conditions hold, there exist a control law of the form

$$\mu = -\begin{bmatrix} K_2 & K_1 & \boldsymbol{K}_0 \end{bmatrix} \begin{bmatrix} e \\ \dot{e} \\ \xi \end{bmatrix} = -\boldsymbol{K}\boldsymbol{z} \tag{91}$$

such that the error system has arbitrary dynamics by pole placement.

Combining the equations (91), with $\boldsymbol{x}$ and u used from the variables ξ and μ, in order to get the control law in term of u and $\boldsymbol{x}$ and noting that $\dot{r} = 0$ the equation becomes:

$$\dot{u} + \boldsymbol{K}_0 \dot{\boldsymbol{x}} = -K_1 e \tag{92}$$

Integrating for u revealing the control law and the action of integral control:

$$u = -K_1 \int^t e d\tau - \boldsymbol{K}_0 \boldsymbol{x} \tag{93}$$

A block diagram of the system is shown below; in the forward controller is possible to see the presence of a pure integrator.

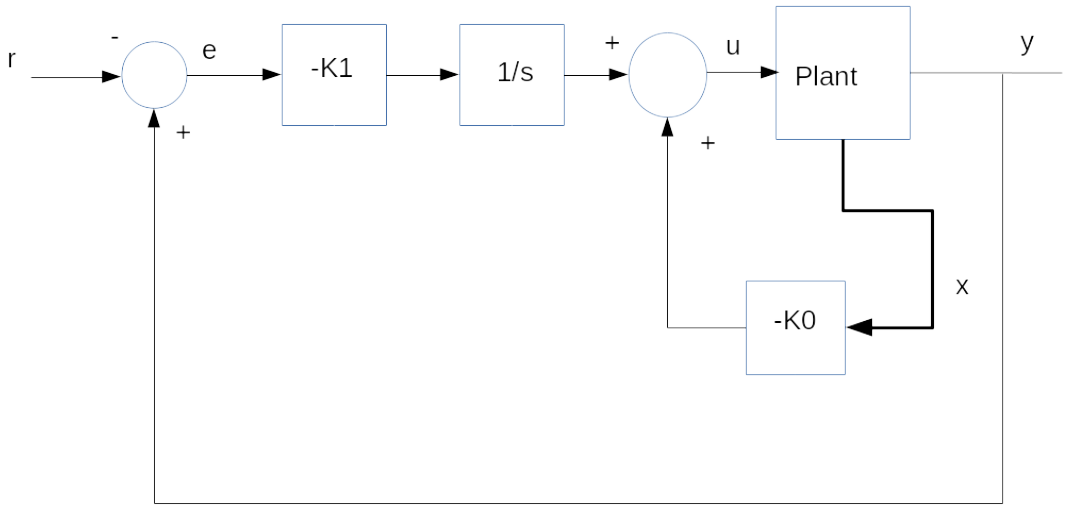

Fig. 10 Reference with integral control

From the nature of the pole placement problem, the state $\boldsymbol{z}$ in Eq.(88) will tend toward zero for all perturbations in the system parameters as long as $\boldsymbol{A} - \boldsymbol{BK}$ remains stable. It's clear that the signals rejected are those satisfy the equations with values α_i implemented in the model. Error values means a steady state error also.

6 Simulation of a cart pendulum

After the theory, it's time to simulate the card-pendulum model using state-variables in feedback control. First of all via Matlab it's explored the characteristic of the model like poles, F-matrix, controllability and observability. Then, a simulation of Linear Quadratic Regulator and Linea Quadratic Gaussian in particular with noise and disturbance are used. For the second one, it's inserted an input step reference to show the response of state-variables, in particular x and θ; it's also use Simulink, a Matlab tool (refer [3] for insight) for a better comprehension. Finally an input reference to track it is used in a feedback control system. A simple gain and an integral control are used to track an input step as last simulations.

6.1 The model

First of all it's worthwhile to design a non linear function of the cart-pendulum derived from (10). Then, it's used the m-code to check a situation when the pendulum is in the "up" position (unstable equilibrium) and the initial condition is $y[t = 0] = [x = 0; \dot{x} = 0; \theta = pi; \dot{\theta} = .5]$. It would be produced a periodic oscillation of θ because acceleration of pendulum put it out of its position of equilibrium (unstable). Think the cart pendulum over a railway and the pendulum-leg can rotate over its up down position freely without falling down a physical ground. Multi-oscillation are dumped for the cart-friction represented by a "d" parameter.

So the function cartpend define the unlinear differential equations:

```
function dy = cartpend(y,m,M,L,g,d,u)

Sy = sin(y(3));
Cy = cos(y(3));
D = m*L*L*(M+m*(1-Cy^2));

dy(1,1) = y(2);
dy(2,1) = (1/D)*(-m^2*L^2*g*Cy*Sy + m*L^2*(m*L*y(4)^2*Sy - d*y(2))) +
m*L*L*(1/D)*u;
dy(3,1) = y(4);
dy(4,1) = (1/D)*((m+M)*m*g*L*Sy - m*L*Cy*(m*L*y(4)^2*Sy - d*y(2))) -
m*L*Cy*(1/D)*u +.01*randn;
```

And the rest of the code to plot the variables evolution

```
m = 1;
M = 5;
L = 2;
g = -10;
d = 1;

tspan = 0:.1:30;
% pendulum in up positoin omega = .5
y0 = [0; 0; pi; .5];
% solve ode
[t,y] = ode45(@(t,y)cartpend(y,m,M,L,g,d,0),tspan,y0);
% plot theta(t)
plot(t,y(:,3))
```

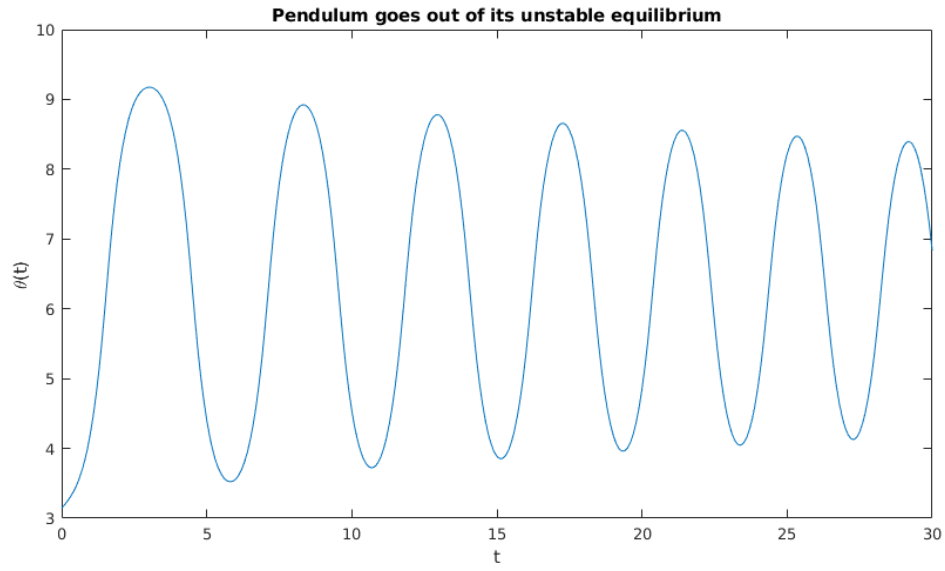

Fig. 11 - Imposing boundary condition to put the pendulum out of its unstable equilibrium and showing the evolution

6.2 The modal form and the meaning of its eigenvalues

Like shown in the chapter 2.2 the modal matrices form is really useful to shown poles over the its diagonal. Matlab gives the possibility to transform the initial set into the desired modal by the command *canon*. T is the transformation between these two representations that using different state variables describe the same physical system. The m-code:

```
s = 1; % pendulum up (s=1)
%% Base system Matrices
F = [0 1 0 0;
    0 -d/M -m*g/M 0;
    0 0 0 1;
    0 -s*d/(M*L) -s*(m+M)*g/(M*L) 0];

G = [0; 1/M; 0; s*1/(M*L)];
H = [1 1 1 1];
J = 0;
sys = ss(F,G,H,J);
[modal_sys,T] = canon(sys, 'modal')

modal_sys =

  A =
           x1       x2       x3       x4
   x1       0        0        0        0
   x2       0    2.434        0        0
```

```
  x3        0         0   -2.467         0
  x4        0         0         0  -0.1665

B =
          u1
 x1        1
 x2   0.2424
 x3  -0.2807
 x4   -1.052

C =
             x1          x2          x3          x4
 y1           1     0.02457     0.02786      0.9486
 y2           0     0.05981    -0.06874     -0.1579
 y3           0     0.07876     0.07793   -0.002645
 y4           0      0.1917     -0.1923   0.0004404

D =
     u1
 y1   0
 y2   0
 y3   0
 y4   0

Continuous-time state-space model.


T =

    1.0000    6.0000   -0.0000   -2.0000
         0   -0.0996    6.3858    2.6237
         0   -0.1138    6.3656   -2.5799
         0   -6.3195   -0.3524    2.1163
```

A note about $\boldsymbol{A}$ matrix; it's known that poles on the right half-plane means unstable systems. It's a confirmation of instability when the pendulum is in the top position (look at the figure 11 θ when the pendulum comes out of its unstable position i.e. either few cents of degree off its top position or like the simulation $\omega = 0.5$)

It turns out that the transformation matrix that will convert the state description matrices to modal form has its columns the eigenvectors of F. In order to get the eigenvalues of F is also possible to use the MATLAB function "eig()" like shown below:

```
>> eig(F)

ans =

         0
   -2.4674
```

```
   -0.1665
    2.4339
```

Backing to (28) it's time to check the controllability of the card pendulum system:

```
>> ctrb(A,B)

ans =

         0    0.2000   -0.0400    0.2080
    0.2000   -0.0400    0.2080   -0.0816
         0    0.1000   -0.0200    0.6040
    0.1000   -0.0200    0.6040   -0.1408

>> rank(ctrb(A,B))

ans =

     4
```

The rank of $\mathcal{C}$ is full and it's possible to control the **full state system** by looping **x**.

6.3 Hot to move poles to control the cart-pendulum

Matlab uses *place* command in order to move poles location of the plant. The first code snippet shows some set of pole location and plot the x-cart position. It can be observed than the more to the left position the poles are the more aggressive response the plant will have.

```
clear all, close all, clc

m = 1;
M = 5;
L = 2;
g = -10;
d = 1;

s = 1; % pendulum up (s=1)

A = [0 1 0 0;
    0 -d/M -m*g/M 0;
    0 0 0 1;
    0 -s*d/(M*L) -s*(m+M)*g/(M*L) 0];

B = [0; 1/M; 0; s*1/(M*L)];
eig(A)

rank(ctrb(A,B))  % is it controllable

%%  Pole placement
% For more complex case a more reliable formula is available with
```

```
% function place
% p is a vector of desired eigenvalues
p0 = [-.01; -.02; -.03; -.04]; % not enough
p1 = [-.3; -.4; -.5; -.6];  % just barely
p2 = [-1; -1.1; -1.2; -1.3]; % good
p3 = [-2; -2.1; -2.2; -2.3]; % aggressive
p4 = [-3; -3.1; -3.2; -3.3]; % aggressive
%p = [-3.5; -3.6; -3.7; -3.8]; % breaks
% create a list of vectors
list_p = {p1,p2,p3,p4};
% K = lqr(A,B,Q,R);

% Plot a list of non-minimum phase zero
for k = 1:numel(list_p)
    tspan = 0:.001:30;
    vec_p = list_p{k};
    % Used place instead acker function
    K = place(A,B,vec_p)
    y0 = [-3; 0; pi+.1; 0];
    [t,y] = ode45(@(t,y)cartpend(y,m,M,L,g,d,-K*(y-[1; 0; pi; 0])),tspan,y0);
    plot(t,y(:,1)); hold on;
end
```

Print out the K values:

```
K =

   -0.0360   -1.3420   71.9720   18.6840


K =

   -1.7160   -7.0260  142.5320   58.0520


K =

  -21.2520  -40.6460  379.6040  165.2920


K =

  -98.2080 -125.8660  851.5160  375.7320
```

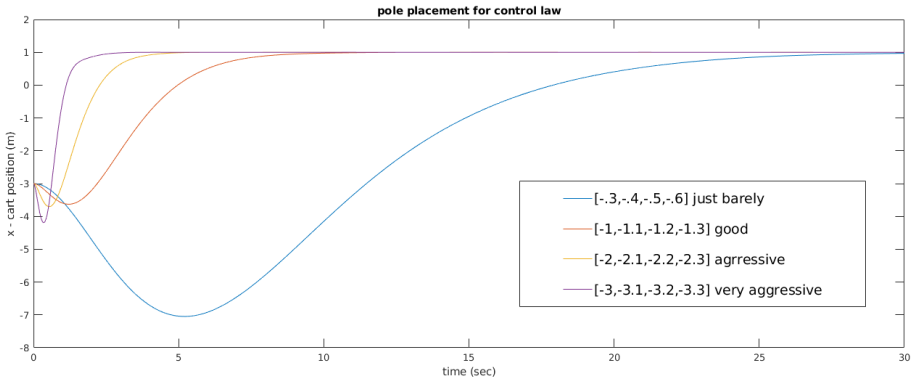

Fig. 12 - Some K values based on different pole placement. The start position is x = -3 and $\theta = \pi + 0.1$. Finale reference position is x = 1 $\theta = \pi$. It's worthwhile to note than more aggressively is equal to require more energy and less robustness

It should be to note the system has to work harder to move pole long way (large gains) and reduce the control effort. In addition it's necessary to mention that the more aggressive will be the control the less robust the system will be because non linearity comes out respect the approximation done.

6.3.1 The Linear Quadratic Regulator

A best trade-off from control effort and fast response is computed using the Linear Quadratic Regulator. In Matlab, the K_r choice is obtained via command:

```
>> Kr = lqr(F,G,Q,R);
```

Some plots are added to show *the spectrum* of state-variables. Another graph is added to plot the functionals represented the best trade-off versus different values of K

```
%% Compare with many examples of Pole Placement
x0 = [-1; 0; pi+.1; 0];  % initial condition
wr = [1; 0; pi; 0];      % reference position
K = lqr(F,G,Q,R);
u=@(x)-K*(x - wr);       % control law
[t,x] = ode45(@(t,x)pendcart(x,m,M,L,g,d,u(x)),tspan,x0);
xLQR = x;
for k=1:length(t)
    JLQR(k) = (x(k,:)-wr')*Q*(x(k,:)'-wr) + u(x(k,:)')^2*R;
end

CC = [0    0.4470    0.7410
    0.8500    0.3250    0.0980
    0.9290    0.6940    0.1250
    0.4940    0.1840    0.5560
    0.4660    0.6740    0.1880
    0.3010    0.7450    0.9330
    0.6350    0.0780    0.1840];
```

```
CCgray = [0.2    0.6470    0.9410
    0.9500    0.4250    0.1980
    1    0.7940    0.2250
    0.5940    0.2840    0.6560
    0.4660    0.6740    0.1880
    0.3010    0.7450    0.9330
    0.6350    0.0780    0.1840];

%%try 100 different poles choises, plot graph of state-variables
%%and cost-functions together LQR gain.

for count = 1:100
    p = [-.5-3*rand; -.5-3*rand; -.5-3*rand; -.5-3*rand];
    K = place(F,G,p);
    u=@(x)-K*(x - wr);       % control law
    [t,x] = ode45(@(t,x)pendcart(x,m,M,L,g,d,u(x)),tspan,x0);
    figure(1)
    for j=1:4
        plot(t(1:50:end),x(1:50:end,j),'Color',[.5 .5 .5] + .3*CC(j,:)), hold on;
    end
    for k=1:length(t)
        J(k) = (x(k,:)-wr')*Q*(x(k,:)'-wr) + u(x(k,:)')^2*R;
    end
    figure(2)
    Jz = cumtrapz(t,J);
    plot(t(1:50:end),Jz(1:50:end),'Color',[.5 .5 .5]), hold on;
end
figure(1)
    for j=1:4
        plot(t(1:10:end),xLQR(1:10:end,j),'Color',CC(j,:),'LineWidth',2)
        l1 = legend('x','v','\theta','\omega')
    end
figure(2)
plot(t,cumtrapz(t,JLQR),'k','LineWidth',2)
```

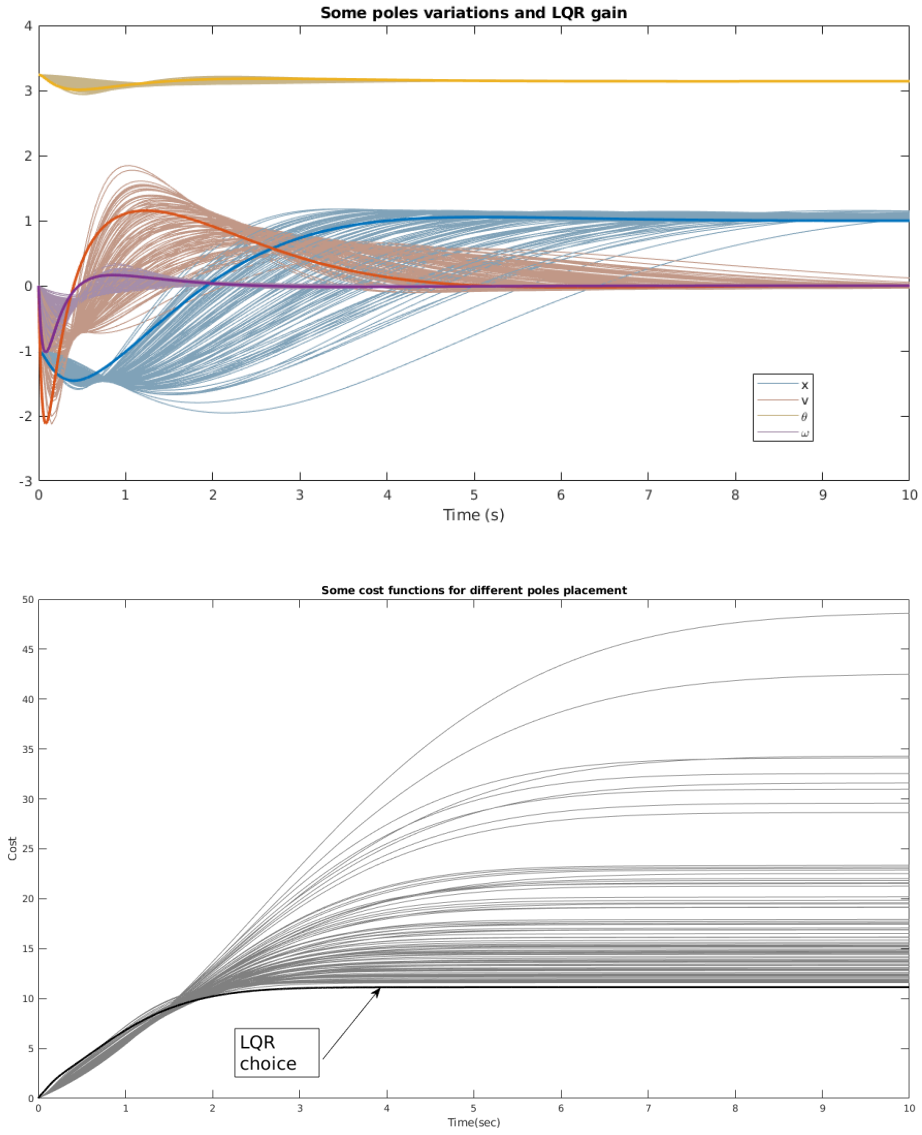

Fig. 13 - Solution for Linear Quadratic Regulator compared with different set of poles locations; then the cost function is plotted in a visual way to show its functional value

6.4 The observability of the cart-pendulum

This code snippet goes to calculate the observability matrix when the measurement matrix **H** (in the code is C) select a sensor related to x cart position and then reply the calculation for θ measurement. It can be highlighted that the determinant is nonzero (full rank) in the first trial and zero in the second one. In particular tuns out that in the second is impossible to reconstruct the x position because that coordinate does not affect the others; equations of cart pendulum are invariant for translations in x position. It's important reconstruct the cart position to have full estimate of state variable putting them in the LQR block for resulting compensator placed in the

feed back (as it will see later).

```
clear all, close all, clc

m = 1;
M = 5;
L = 2;
g = -10;
d = 1;

s = -1; % pendulum up (s=1)

A = [0 1 0 0;
    0 -d/M -m*g/M 0;
    0 0 0 1;
    0 -s*d/(M*L) -s*(m+M)*g/(M*L) 0];

B = [0; 1/M; 0; s*1/(M*L)];

C = [1 0 0 0];

obsv(A,C)
det(obsv(A,C))


C = [0 0 1 0]; %%  only observable if x measured... because x can't be reconstructed

obsv(A,C)
det(obsv(A,C))
```

The terminal result:

```
ans =

    1.0000         0         0         0
         0    1.0000         0         0
         0   -0.2000    2.0000         0
         0    0.0400   -0.4000    2.0000


ans =

     4


ans =

         0         0    1.0000         0
         0         0         0    1.0000
```

```
         0    0.1000   -6.0000         0
         0   -0.0200    0.2000   -6.0000

ans =

     0
```

6.4.1 Simulation of plant disturbance and noise measurement.

To give a demonstration of best choice of $L \equiv K_f$ it's considered a system with $J = 0$ in (19) and pendulum in bottom position. It's better create a practical solution around a stable equilibrium point because the algebra is the same. The system represented in the picture shows the linearized system with input u to be the force applied to the cart pendulum, w_d the gaussian zero-mean disturbance applied to the model and the cart position ($\boldsymbol{H} = [1000]$) noised as output of state variable observed. The goal is reconstruct the full state estimate. However the strategy is reconstruct the normal form of the first order state-equation. To do it is necessary to represent the system like 2 *augmented* blocks to feed in Matlab functions:

$$\boldsymbol{Q} = 0.1\boldsymbol{I} \qquad \boldsymbol{x} = \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} \tag{94}$$

$$R = 1 \tag{95}$$

$$\begin{aligned} \boldsymbol{G}_A &= \begin{bmatrix} \boldsymbol{G} & \boldsymbol{Q} & \underline{\boldsymbol{0}}x\boldsymbol{B} \end{bmatrix} \\ \boldsymbol{F}_A &= \boldsymbol{F} \\ \boldsymbol{H}_A &= \boldsymbol{H} \\ \boldsymbol{J}_A &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & R \end{bmatrix} \end{aligned} \tag{96}$$

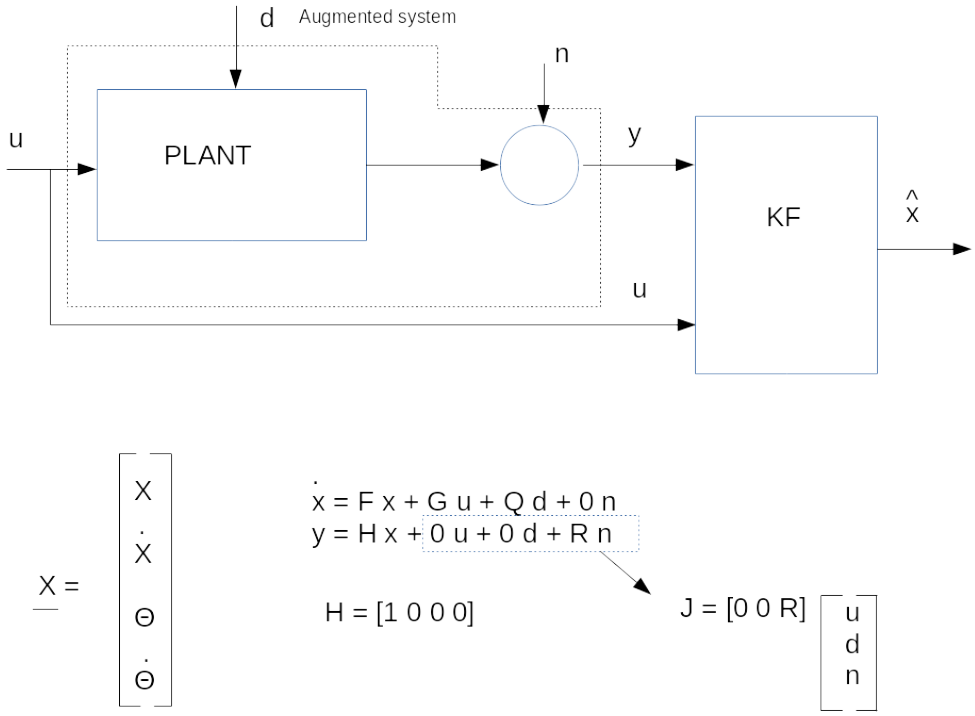

Fig 14 A scheme of Kalman Filter; d and n are disturbance of state system and noise measurement introduced in the simulation. Note y true is the output of the plant and y noise the estimator input

According to the equations (67), by inspecting (15) and (19) the *new* estimator-system matrices are:

$$\begin{aligned} \boldsymbol{F}_k &= \boldsymbol{F} - \boldsymbol{K}_f \boldsymbol{H} \\ \boldsymbol{G}_k &= \left[\boldsymbol{G}\boldsymbol{K}_f\right] \\ \boldsymbol{H}_k &= \boldsymbol{I} \\ \boldsymbol{J}_k &= \boldsymbol{0} \end{aligned} \tag{97}$$

The input u is augmented also to taking into account of disturbance plant and noise measurement:

$$\boldsymbol{u}_{aug} = \begin{bmatrix} u & Q^2 * \boldsymbol{u}_{dist} & u_{noise} \end{bmatrix} \tag{98}$$

The code:

```
clear all, close all, clc

m = 1;
```

```
M = 5;
L = 2;
g = -10;
d = 1;

s = -1; % pendulum up (s=1)

% y = [x; dx; theta; dtheta];
F = [0 1 0 0;
    0 -d/M -m*g/M 0;
    0 0 0 1;
    0 -s*d/(M*L) -s*(m+M)*g/(M*L) 0];

G = [0; 1/M; 0; s*1/(M*L)];

H = [1 0 0 0];

J = zeros(size(H,1),size(G,2));

%%  Augment system with disturbances and noise
Q = .1*eye(4);  % disturbance covariance
R = 1;       % noise covariance

GA = [G Q 0*G];  % augment inputs to include disturbance and noise

sysH = ss(F,GA,H,[0 0 0 0 0 R]);  % build big state space system... with single output

% system with full state output, disturbance, no noise
sysFullOutput = ss(F,GA,eye(4),zeros(4,size(GA,2)));

%%  Build Kalman filter
[L,P,E] = lqe(F,Q,H,Q,R);  % design Kalman filter
Kf = (lqr(F',H',Q,R))';   % alternatively, possible to design using "LQR" code

sysKF = ss(F-L*H,[G L],eye(4),0*[G L]);  % Kalman filter estimator

%%  Estimate linearized system in "down" position (Gantry crane)
dt = .01;
t = dt:dt:50;

uDIST = randn(4,size(t,2));
uNOISE = randn(size(t));
u = 0*t;
u(100:120) = 100;     % impulse
u(1500:1520) = -100;  % impulse

uAUG = [u; Q*Q*uDIST; uNOISE];
```

```
[y,t] = lsim(sysH,uAUG,t);% Here y is the noise measured signal

[xtrue,t] = lsim(sysFullOutput,uAUG,t); % The x cart-pendulum position of the plant

[x,t] = lsim(sysKF,[u; y'],t); % The estimate x of cart-pendulum


plot(t,xtrue,'-',t,x,'--','LineWidth',2) %% Plot together y,x true,x estimate
figure
plot(t,y)
hold on
plot(t,xtrue(:,1),'r')
plot(t,x(:,1),'k--')
```

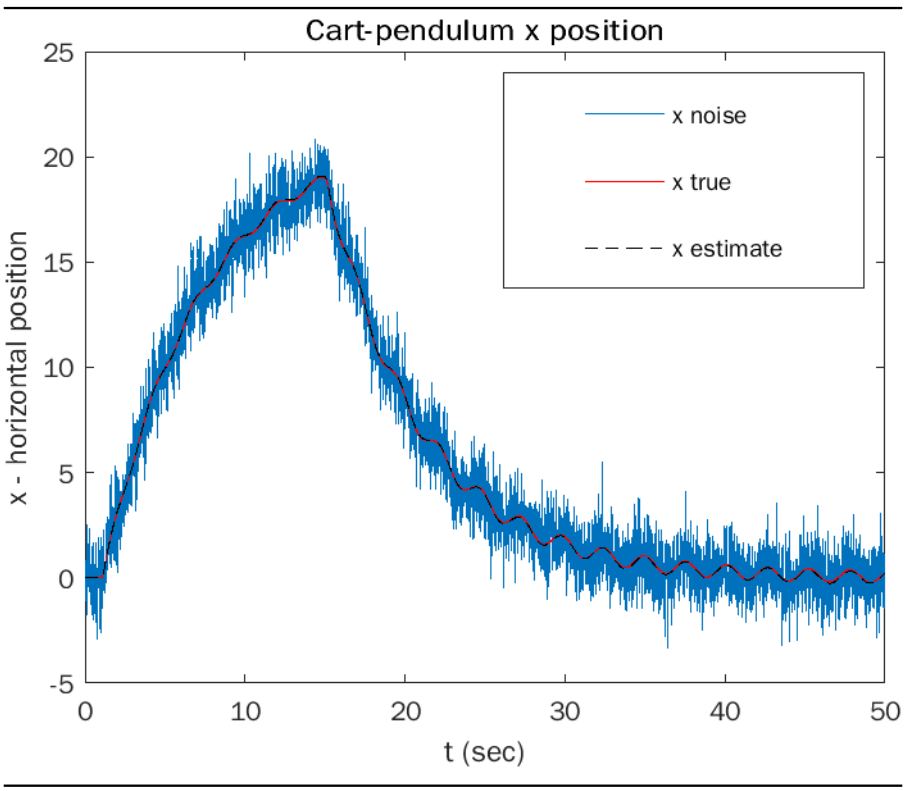

Fig. 15 Simulation of a noise position from an impulse system. Estimate position "cover" the true

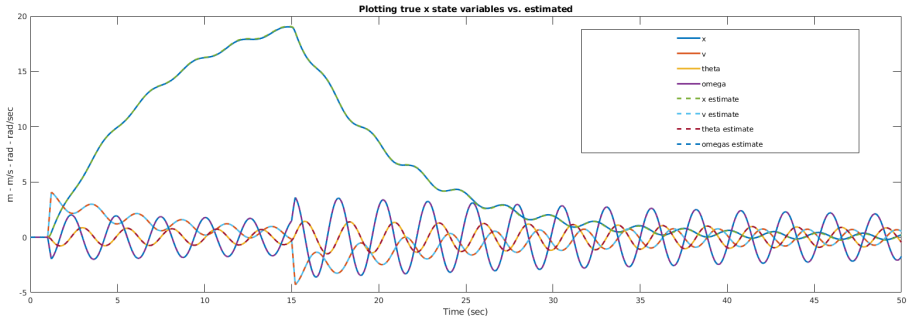

Fig. 16 Showing the state variable "true" and estimate

With the estimate state, it's the time to "*feedback*" it in a Linear Quadratic Regulator to combine control and estimation together.

6.4.2 Simulation of LQG in a feed back control. How to put all togheter.

To apply an LQG regulator to the inverted pendulum on a cart, we will simulate the full nonlinear system in Simulink, as shown in figure 17. The non-linear cart pendulum is connected by the force u in addiction to a gaussian random noise disturbance that affected the model. The output of the cart-pendulum is the state-variable vector connected with $\boldsymbol{H}$ matrix to go input in the estimator block only with the position-cart sensor noise measurement gaussian distributed also; the relation is $y = \boldsymbol{H}\boldsymbol{x} + w_n$. The Kalman filter reconstruct the full state variable for the compensator block.

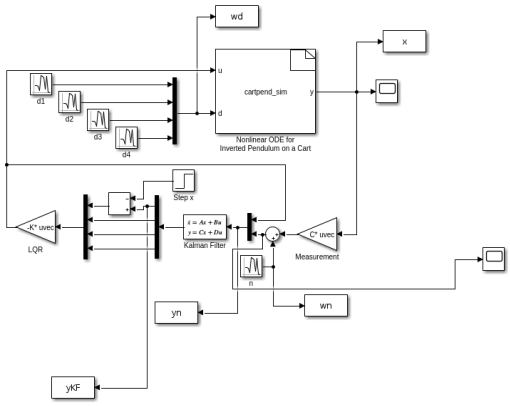

Fig. 17 Matlab Simulink model for sensor-based LQG feedback control.

The system starts near the vertical equilibrium, at $\boldsymbol{x}_0 = [0, 0, 3.14, 0]$, and **it's commanded a step** in the cart position from x = 0 to x = 1 at t = 10. The resulting response is shown in figure 18. Despite noisy measurements (figure 19) and disturbances (figure 20), the controller is able to effectively track the reference cart position while stabilizing the inverted pendulum.

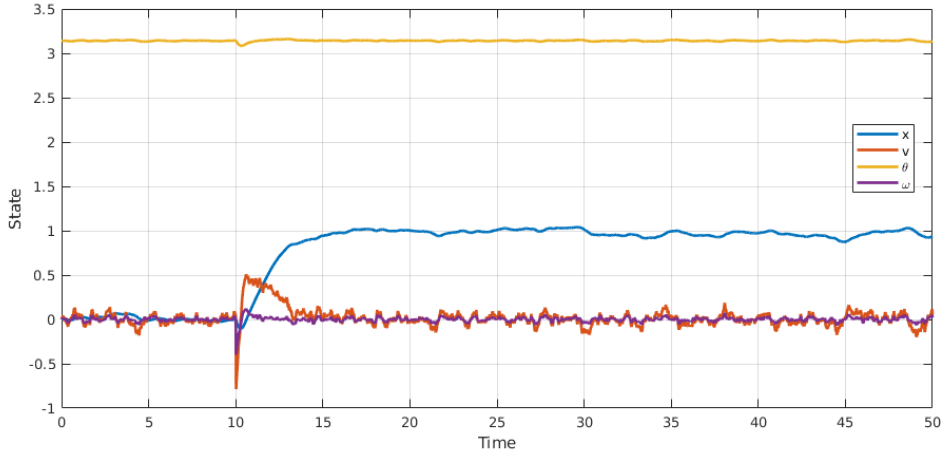

Fig. 18 Output response using LQG feedback control.

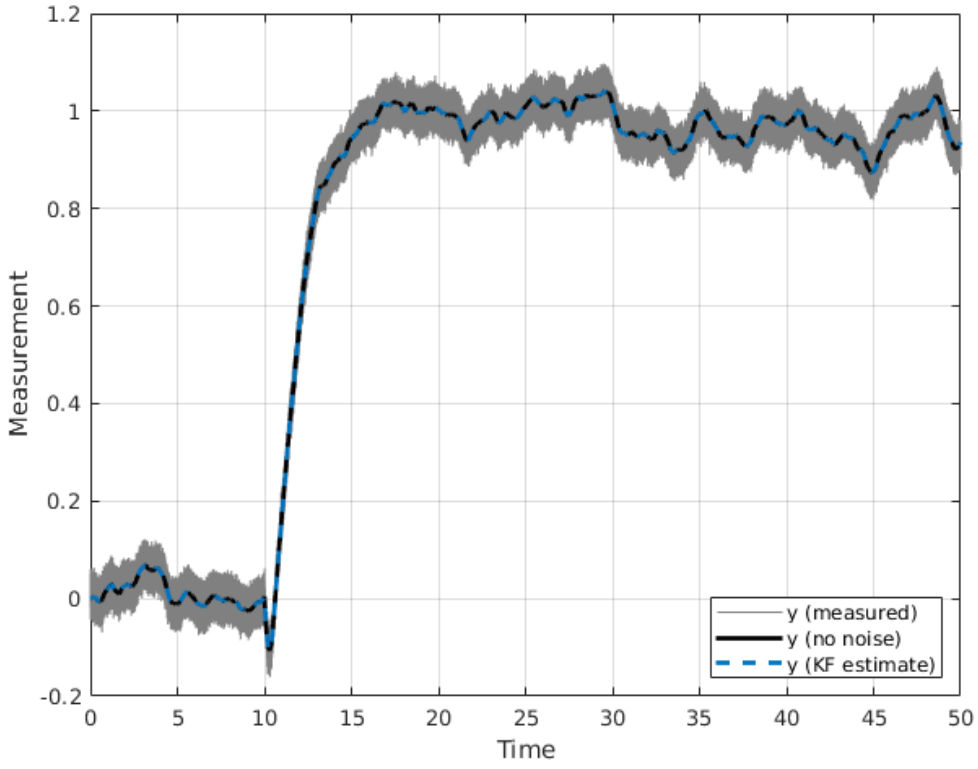

Fig. 19 Noisy measurement used for the Kalman filter, along with the underlying noiseless signal and the Kalman filter estimate.

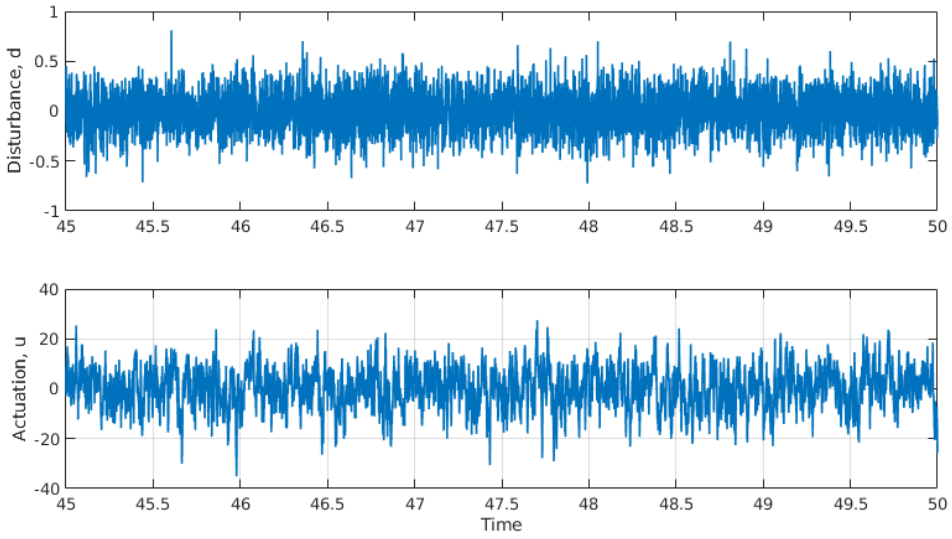

Fig. 20 Noisy measurement used for the Kalman filter, along with the underlying noiseless signal and the Kalman filter estimate

The m-code set some values needed for the simulation

```
clear all, close all, clc

%% Set cart-pendulum parameters
m = 1;
M = 5;
L = 2;
g = -10;
d = 1;

s = 1; % pendulum up (s=1)
%% Base system Matrices
F = [0 1 0 0;
    0 -d/M -m*g/M 0;
    0 0 0 1;
    0 -s*d/(M*L) -s*(m+M)*g/(M*L) 0];

G = [0; 1/M; 0; s*1/(M*L)];

% Set cart-position
H = [1 0 0 0];

J = zeros(size(H,1),size(G,2));

%% Linear Quadratic Regulator
Q = [1 0 0 0;
```

```
    0 1 0 0;
    0 0 1 0;
    0 0 0 1];
R = .000001;
K = lqr(F,G,Q,R);  % design controller u = -K*x


%% Augmented system
%  Augment system with disturbances and noise
Vdmag = .04;
Vd = Vdmag*eye(4);  % disturbance covariance
Vn = .0002;       % noise covariance

% GF sysC sysFulloutput not used in Simulink
GF = [G Vd 0*G];  % augment inputs to include disturbance and noise
sysH = ss(F,GF,H,[0 0 0 0 0 Vn]);  % build big state space system... with single output
 % system with full state output, disturbance, no noise
sysFullOutput = ss(F,GF,eye(4),zeros(4,size(GF,2)));

%  Build Kalman filter 2 alternatives
[L,P,E] = lqe(F,eye(4),H,Vd,Vn);  % design Kalman filter
Kf = (lqr(F',H',Vd,Vn))';   % alternatively, possible to design using "LQR" code

% The system matrices for LQE. Used in Simulink
sysKF = ss(F-Kf*H,[G Kf],eye(4),0*[G Kf]);  % Kalman filter estimator
```

Here the m-code to print the result

```
CC = [0    0.4470    0.7410
    0.8500    0.3250    0.0980
    0.9290    0.6940    0.1250
    0.4940    0.1840    0.5560
    0.4660    0.6740    0.1880
    0.3010    0.7450    0.9330
    0.6350    0.0780    0.1840];

figure(1)
plot(yn.Time,yn.Data(:,2),'Color',[.5 .5 .5])
hold on
plot(x.Time,x.Data(:,1),'Color',[0 0 0],'LineWidth',2)
plot(yKF.Time,yKF.Data(:,1),'--','Color',CC(1,:),'LineWidth',2)
set(gcf,'Position',[100 100 500 200])
xlabel('Time')
ylabel('Measurement')
l1 = legend('y (measured)','y (no noise)','y (KF estimate)');
set(l1,'Location','SouthEast')
grid on

%%
```

```
figure(2)
subplot(3,1,1)
plot(wd.Time(1:100:end),wd.Data(1:100:end,1),'LineWidth',1.2)
xlim([45 50])
ylabel('Disturbance, d')
subplot(3,1,2)
plot(wn.Time(1:100:end),wn.Data(1:100:end,1),'LineWidth',1.2)
xlim([45 50])
ylabel('Noise measurement, n')
subplot(3,1,3)
plot(yn.Time(1:100:end),yn.Data(1:100:end,1),'LineWidth',1.2)
xlim([45 50])
set(gcf,'Position',[100 100 500 200])
xlabel('Time')
ylabel('Actuation, u')
grid on

%%
figure(3)
plot(x.Time,x.Data,'LineWidth',2)
l1 = legend('x','v','\theta','\omega')
set(gcf,'Position',[100 100 500 200])
xlabel('Time')
ylabel('State')
grid on
```

6.5 A simulation for a cart pendulum into a feed-back control with a reference input

Having described in chapter 5 how to obtain precise result it's shown below the block scheme *without* a gain reference. It's expected to obtain an output error.

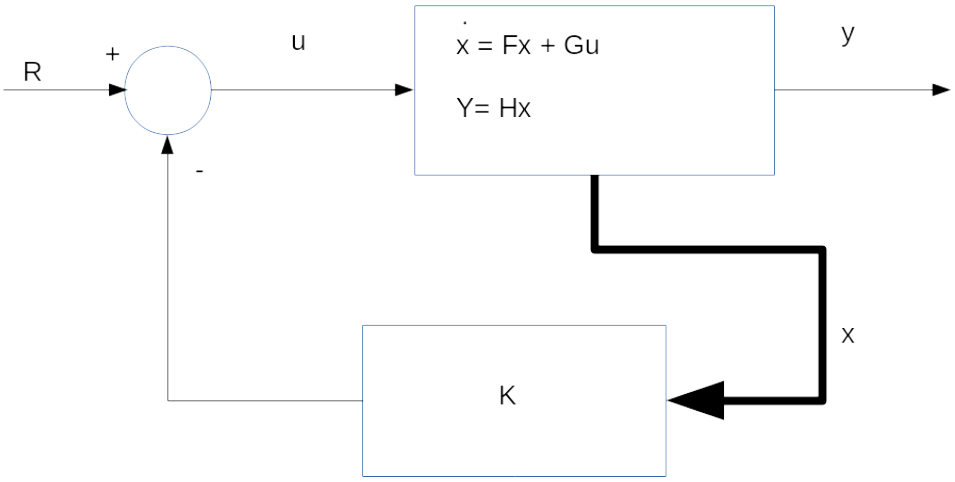

Fig. 21 Block scheme of reference system with Linear Quadratic Regulator

The design criteria for this system with the cart receiving a 0.2 step input are as follows:

- Settling time for x and theta of less than 5 seconds.
- Rise time for x of less than 1 second
- Overshoot of θ less than 20 degree (0.35 radians). Please note in this lab the up position is 0 degree and down position is π
- Steady-state error within 2%

For an inverted pendulum it is unrealistic to consider just the single output system. Using state-space/state variables methods it is relatively simple to work with a multi-output system, so in this hand-on lab it will design a controller with both the pendulum angle and the cart position in mind.

A piece of code

```
%% Set cart-pendulum parameters
m = 1;
M = 5;
L = 2;
g = -10;
d = 1;
```

```
s = 1; % pendulum up (s=1)
%% Base system Matrices
F = [0 1 0 0;
    0 -d/M -m*g/M 0;
    0 0 0 1;
    0 -s*d/(M*L) -s*(m+M)*g/(M*L) 0];

G = [0; 1/M; 0; s*1/(M*L)];

% Set cart-position
H = [1 0 0 0;
        0 0 1 0];

J = zeros(size(H,1),size(G,2))
```

The next step in the design process is to assume to have full-state feedback (i.e. that we can measure all four states), and find the vector *K* which determines the feedback control law. This can be done in a number of ways. If it's known the desired closed-loop poles, it's possible to use the *place* or *acker* command. Another option is to use the *lqr* function; this will give you the optimal controller (under certain assumptions. The lqr function allows to choose two parameters, R and Q, which will balance the relative importance of the input and state in the cost function to optimize the trade-off. The simplest case is to assume R=1, and Q=H'*H. It's worthwhile to note that it's possible using both outputs (the pendulum's angle and the cart's position). Essentially, the lqr method allows to control both outputs. In this case, it is pretty easy to do. The controller can be tuned by changing the nonzero elements in the Q matrix to get a desirable response. To find the structure of Q it's possible use the matlab command:

```
>> H'*H

ans =

     1     0     0     0
     0     0     0     0
     0     0     1     0
     0     0     0     0
```

The element in the 1,1 position will be used to weight the cart's position and the element in the 3,3 position will be used to weight the pendulum's angle. The input weighting R will remain at 1. Now that we know what the Q matrix should look like we can experiment to find the K matrix that will give a good controller. Then it's the time to find the K matrix and plot the response all in one step so that changes can be made in the control and be seen automatically in the response. Enter the following text into your m-file:

```
%% Linear Quadratic Regulator
%% try different value of Q
Q = [1 0 0 0;
    0 0 0 0;
    0 0 1 0;
    0 0 0 0];
```

```
R = 1;
K = lqr(F,G,Q,R);  % design controller u = -K*x
Ac = [(F-G*K)];
Bc = [G];
Cc = [H];
Dc = [J];

T=0:0.01:5;
U=0.2*ones(size(T));
[Y,X]=lsim(Ac,Bc,Cc,Dc,U,T);
plot(T,Y)
legend('Cart','Pendulum')
```

```
>> K

K =

   -1.0000   -5.4325  153.2343   63.8985
```

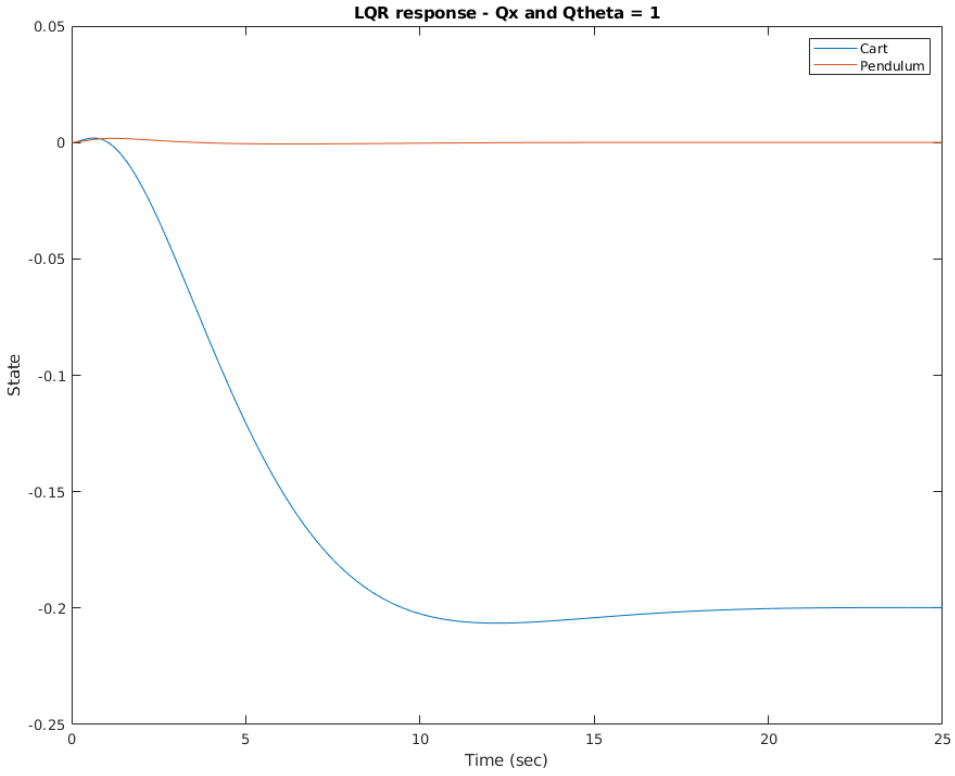

Fig. 22 The curve in red represents the pendulum's angle, in radians and the curve in blue represents the cart's position in meters. As you can see, this plot is not satisfactory

As you can see, this plot is not satisfactory. The pendulum and cart's overshoot appear fine, but

their settling times need improvement and the cart's rise time needs to go down. The cart is not near the desired location but has in fact moved in the other direction. This error will be dealt in the next section and right now the focus is on the settling and rise times. Increasing Q_x makes the settling and rise times go down, and lowers the angle the pendulum moves. Using $Q_x = 5000$ and $Q_\theta = 100$, the following value of K and step response were found:

```
%% Linear Quadratic Regulator
%% try different value of Q
Q = [5000 0 0 0;
    0 0 0 0;
    0 0 100 0;
    0 0 0 0];
R = 1;
K = lqr(F,G,Q,R);  % design controller u = -K*x
Ac = [(F-G*K)];
Bc = [G];
Cc = [H];
Dc = [J];

T=0:0.01:5;
U=0.2*ones(size(T));
[Y,X]=lsim(Ac,Bc,Cc,Dc,U,T);
plot(T,Y)
legend('Cart','Pendulum')

>> K

K =

  -70.7107  -91.3062  636.5903  280.1483
```

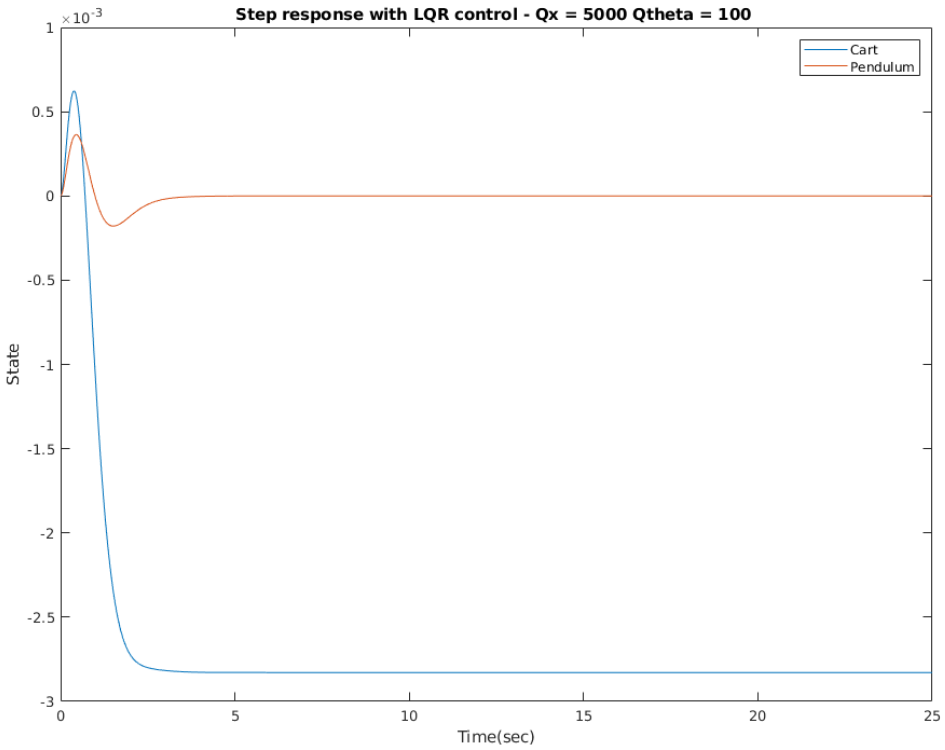

Fig. 23 Raising Q_x and Q_θ imply more control effort and smaller tracking error

If it's increased Q_x and Q_θ even higher, it's possible to improve the response even more. This plot was chosen because it satisfied the design requirements while keeping x and y as small as possible. In this problem, Q_x and Q_θ have been used to describe the relative weight of the tracking error in the cart's position and pendulum's angle versus the control effort. The higher these 2 parameters are, the more control effort is used, but the smaller the tracking error. The system response has a settling time under 3 seconds.

In contrast to the other design methods, where it's feedback the output and compare it to the reference input to compute an error, with a full-state feedback controller it's feeding back all the states. It necessary to compute what the steady-state value of the states should be, multiply that by the chosen gain K, and use a new value as our reference for computing the input. This can be done by adding a constant gain $\bar{N}$ after the reference. The schematic below shows this relationship:

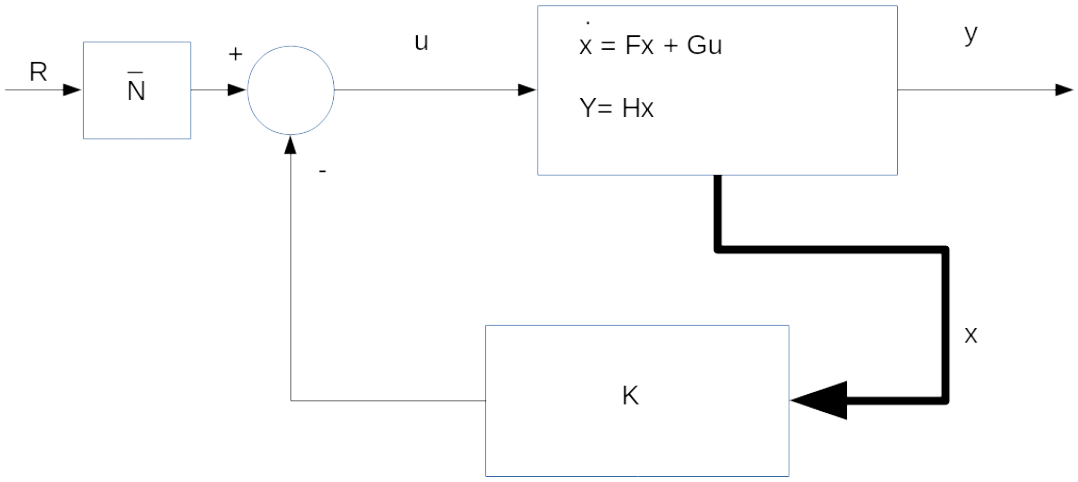

Fig. 22 Adding a gain in a reference input

Nbar can be found using the user-defined function rscale. A different H had to be used because the rscale function will not work for multiple outputs. However, the Nbar found is correct, as you can see from the output below:

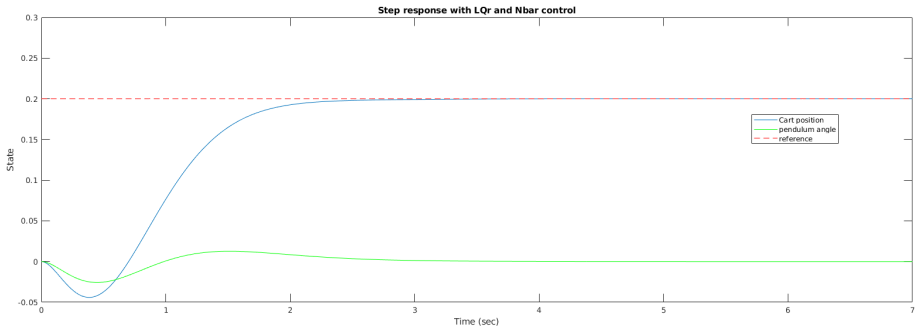

Fig. 23 Step response with LQR and $\bar{N}$ control

Here the matlab code:

```
%% Set augmented system
Ac = [(F - G*K)];
Bc = [G];
Cc = [H];
Dc = [J];

%% Plot system with reference
T=0:0.01:25;
U=0.2*ones(size(T));
[Y,X]=lsim(Ac,Bc,Cc,Dc,U,T);
%plot(T,Y)
%legend('Cart','Pendulum')


%% Set gain reference
Cn = [1 0 0 0];
Nbar = rscale(F,G,Cn,0,K);
Bcn = [Nbar*G];
[Y,X] = lsim(Ac,Bcn,Cc,Dc,U,T);
plot(T,Y)
legend('Cart position','pendulum angle')
```

```
function[Nbar]=rscale(A,B,C,D,K)

s = size(A,1);
Z = [zeros([1,s]) 1];
N = inv([A,B;C,D])*Z';
Nx = N(1:s);
Nu = N(1+s);
Nbar=Nu + K*Nx;
```

Note that rscale function implements the equation (81) and (80)

6.6 Add an integrator in a feedback control for inverted pendulum. The state-space approach

For the system:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{F}\boldsymbol{x} + \boldsymbol{G}u + \boldsymbol{G}_1 w \\ y &= \boldsymbol{H}\boldsymbol{x} \end{aligned} \tag{99}$$

it's possible feed back the integral control, $e = y - r$, together the state of the plant $\boldsymbol{x}$, by augmenting the plant state with the extra integral state x_I which respond to the differential equation:

$$\dot{x_I} = \boldsymbol{H}\boldsymbol{x} - r(=e), x_I = \int^t e d\tau \tag{100}$$

The augmented state equations become:

$$\begin{bmatrix} \dot{x_I} \\ \dot{\boldsymbol{x}} \end{bmatrix} = \begin{bmatrix} 0 & \boldsymbol{H} \\ \boldsymbol{0} & \boldsymbol{F} \end{bmatrix} \begin{bmatrix} x_I \\ \boldsymbol{x} \end{bmatrix} + \begin{bmatrix} 0 \\ \boldsymbol{G} \end{bmatrix} u - \begin{bmatrix} 1 \\ \boldsymbol{0} \end{bmatrix} r + \begin{bmatrix} 0 \\ \boldsymbol{G}_1 \end{bmatrix} w \tag{101}$$

and the feedback law is:

$$u = -\begin{bmatrix} K_I & \boldsymbol{K}_0 \end{bmatrix} \begin{bmatrix} x_I \\ \boldsymbol{x} \end{bmatrix} \tag{102}$$

The problem is traced back to a LQR system.

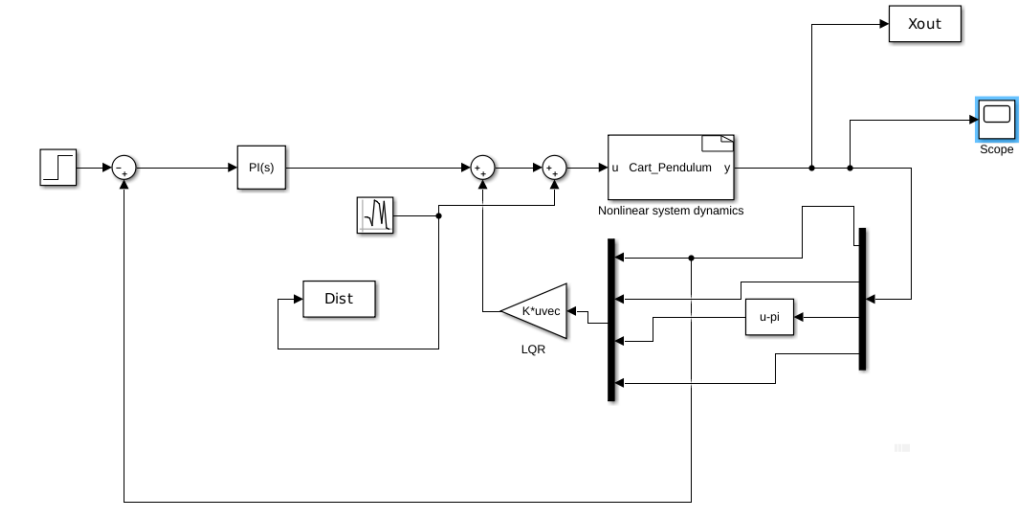

Fig. 25 Simulink for integral control with reference

Building a simulation via *simulink* means recover the LQR part plus the feedback with reference, an integral control plus a random number generator to create the disturbance over the u line. In particular the system is under a unit step reference that cart pendulum position "x" has to follow. The matlab code follow the LQR part for the augmented system:

```
clear all, close all, clc

%% Set cart-pendulum parameters
m = 1;
M = 5;
L = 2;
g = -10;
d = 1;

s = 1; % pendulum up (s=1)
%% Base system Matrices
F = [0 1 0 0;
    0 -d/M -m*g/M 0;
    0 0 0 1;
    0 -s*d/(M*L) -s*(m+M)*g/(M*L) 0];

G = [0; 1/M; 0; s*1/(M*L)];
```

```
% Set cart-position and pendulum angle
H = [1 0 0 0]


J = zeros(size(H,1),size(G,2));

%% Integral control augmented system
Fa = [0 H;0 F(1,:);0 F(2,:);0 F(3,:);0 F(4,:)];
Ga = [0; G];
R = 0.0001;
K = lqr(Fa,Ga,eye(5),R);
```

printing K via command line

```
>> K

K =

   1.0e+03 *

   -0.1000   -0.2696   -0.3145    2.0341    0.9096
```

The K values are inserted in the Simulink block.

Here the graphs of simulation:

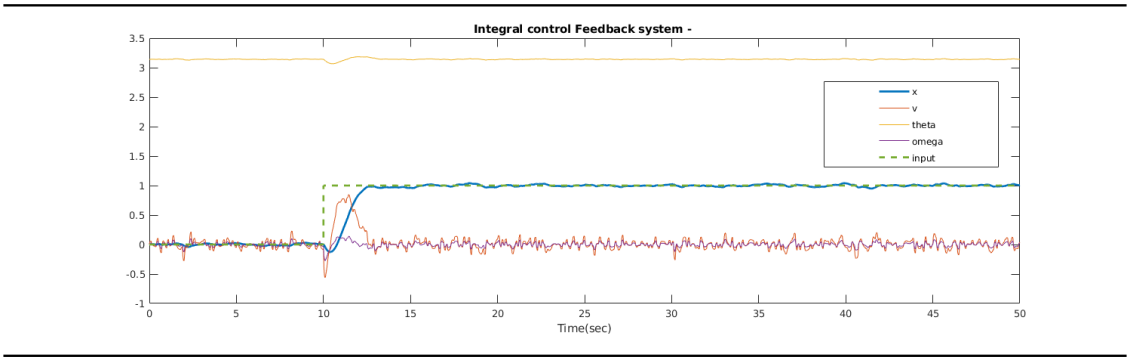

Fig. 26 As noted $\theta = \pi$ when pendulum is in "up" unstable position. The feed back reject the disturbance and the cart tracks the step reference starting at time $t = 10$ seconds

References

[1] Steven L Brunton and J Nathan Kutz. *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press, 2019.

[2] Gene F Franklin, J David Powell, and Abbas Emami-Naeini. *Feedback control of dynamic systems*. Pearson London, 2015.

[3] Mohinder S Grewal and Angus P Andrews. *Kalman filtering: Theory and Practice with MATLAB*. John Wiley & Sons, 2014.