Intuitive understanding of superposition and time invariance of **linear** dynamic systems: sequence of steps for a given target behaviour, does NOT work with a lot of "inertia" (higher order system)

Video presentation and materials: http://personales.upv.es/asala/YT/V/linregla3ord2EN.html

This code executed in Matlab **R2022b** (Linux)

Objectives: intuitively understand the concept of a linear time-invariant (dynamic) system, and how to use "experimental" time response to compute input profiles that achieve certain objectives. Understand that in processes of order largen than 1 (with "inertia"), this procedure may not work as well as in the 1st-order case (companion video linregla3EN.html).

Table of Contents

Step response test to gather data

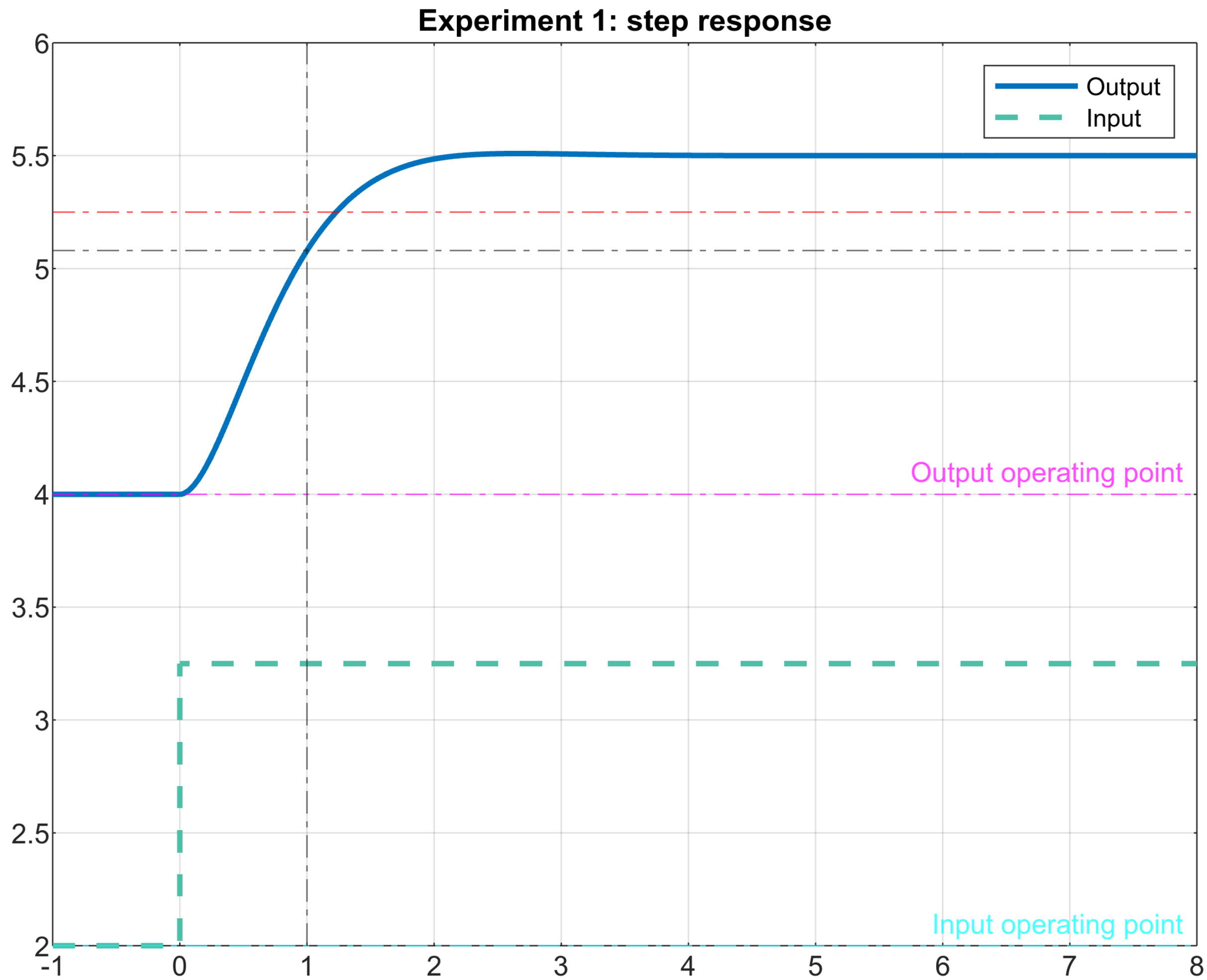
Let us consider a certain unknown system, of which we only know that it is linear (or approximately) around an operating point u=2, y=4;

Let's simulate its step response of 0.5 (incremental)

```
u_op=2; y_op=4;
inc_u=1.25;
u=@(t) u_op+inc_u*(t>=0);
Y=simulsystem(u); %code at the end... we will intentionally NOT see it.
yline(y_op,'-.m',Label="Output operating point")
yline(u_op,'-.c',Label="Input operating point")
```

We will highlight on the graph certain lines that we will need in later developments.

```
t_set_desired=1.0;
xline(t_set_desired,'-.')
yline(5.25,'-.r')
yline(5.08,'-.')
legend("Output","Input"), title("Experiment 1: step response")
```




```
Final_value=Y(end) %final equilibrium value
```

```
Final_value = 5.5000
```

Problems involving computation of input step amplitude

With only the above information from the step test, we can answer several questions.

Prefixed final value

1.) What input will be needed to raise the output to **5.25** units?

From linearity (i.e., "proportionality"), if with an input increase of 0.5 it goes up 0.75 units, to go up to 5.25 we need:

```
inc_output=Final_value-y_op %experimental measurement
```

```
inc_output = 1.5000
```

```
static_gain=inc_output/inc_u %increment per unit input
```

```
static_gain = 1.2000
```

```
desired_inc_output=5.25-y_op
```

```
desired_inc_output = 1.2500
```

```
computed_inc_input=desired_inc_output/static_gain
```

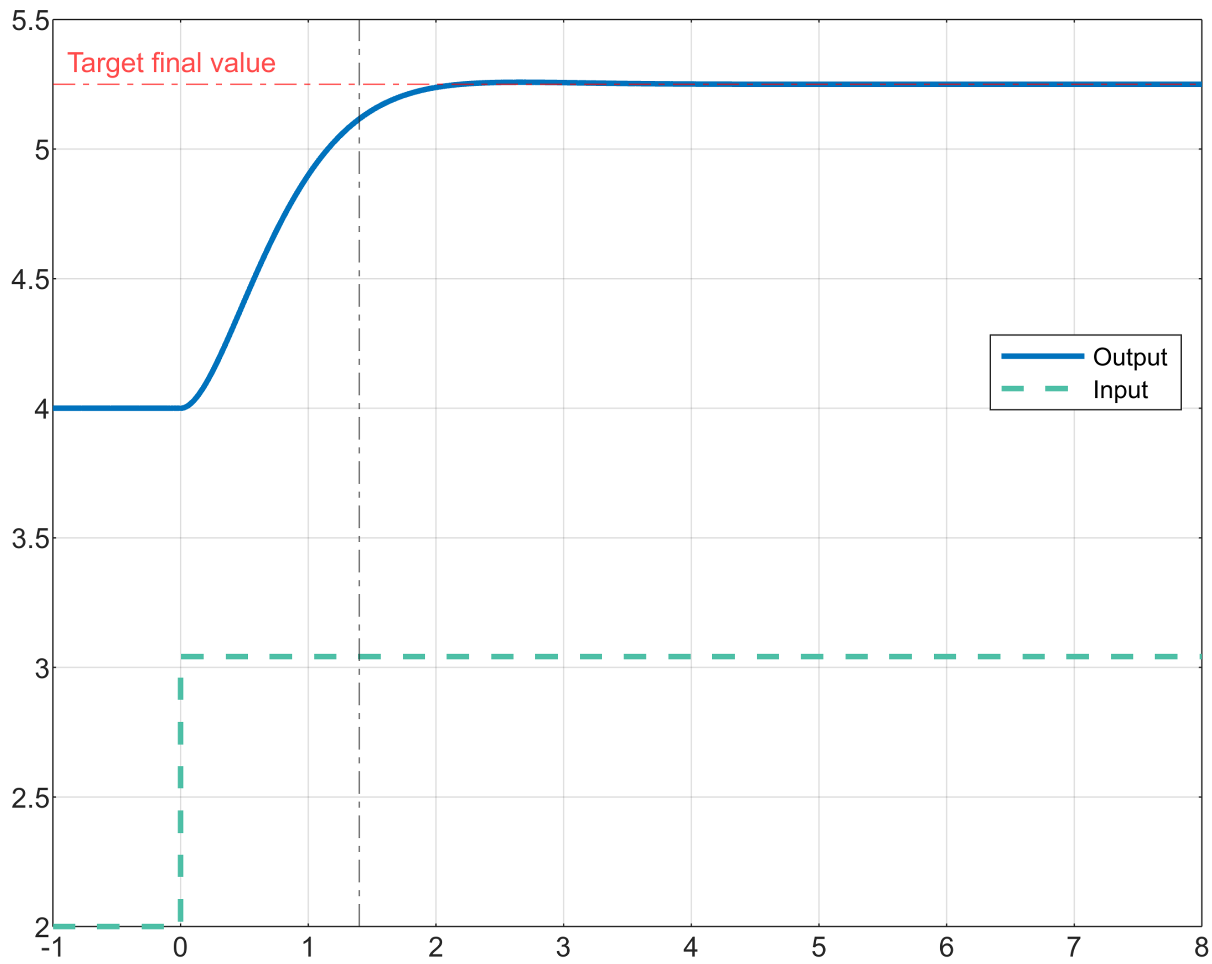
```
computed_inc_input = 1.0417
```

Therefore, the "absolute" (i.e., non-incremental) input value that in equilibrium will achieve the desired output will be:

```
u_computed=u_op+computed_inc_input
```

```
u_computed = 3.0417
```

```
u=@(t) u_op+computed_inc_input*(t>=0);
simulsystem(u);
xline(1.4,'-.')
yline(5.25,'-.r',Label="Target final value",LabelHorizontalAlignment="left")
legend("Output","Input",Location="best")
```



Preset final value and settling time

1.) What input will be needed to raise the output to **5.25** units in **1 seconds**?

```
inc_output=5.08-y_op %experimental measurement (supposedly)
```

```
inc_output = 1.0800
```

```
gain_in_1dot4seconds=inc_output/inc_u %increment per unit input in given time
```

```
gain_in_1dot4seconds = 0.8640
```

```
desired_inc_output=5.25-y_op
```

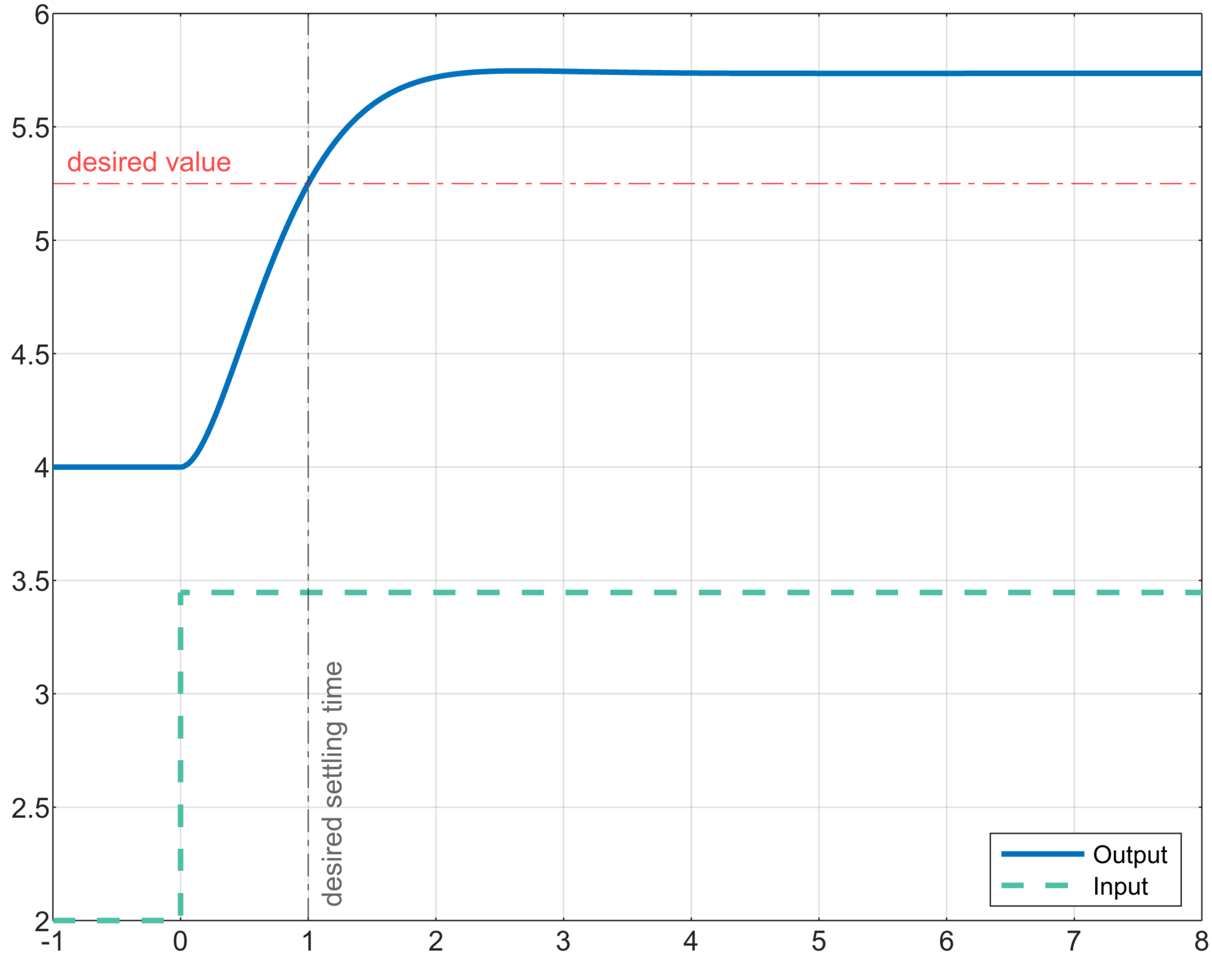
```
desired_inc_output = 1.2500
```

```
inc_input_computed2=desired_inc_output/gain_in_1dot4seconds
```

```
inc_input_computed2 = 1.4468
```

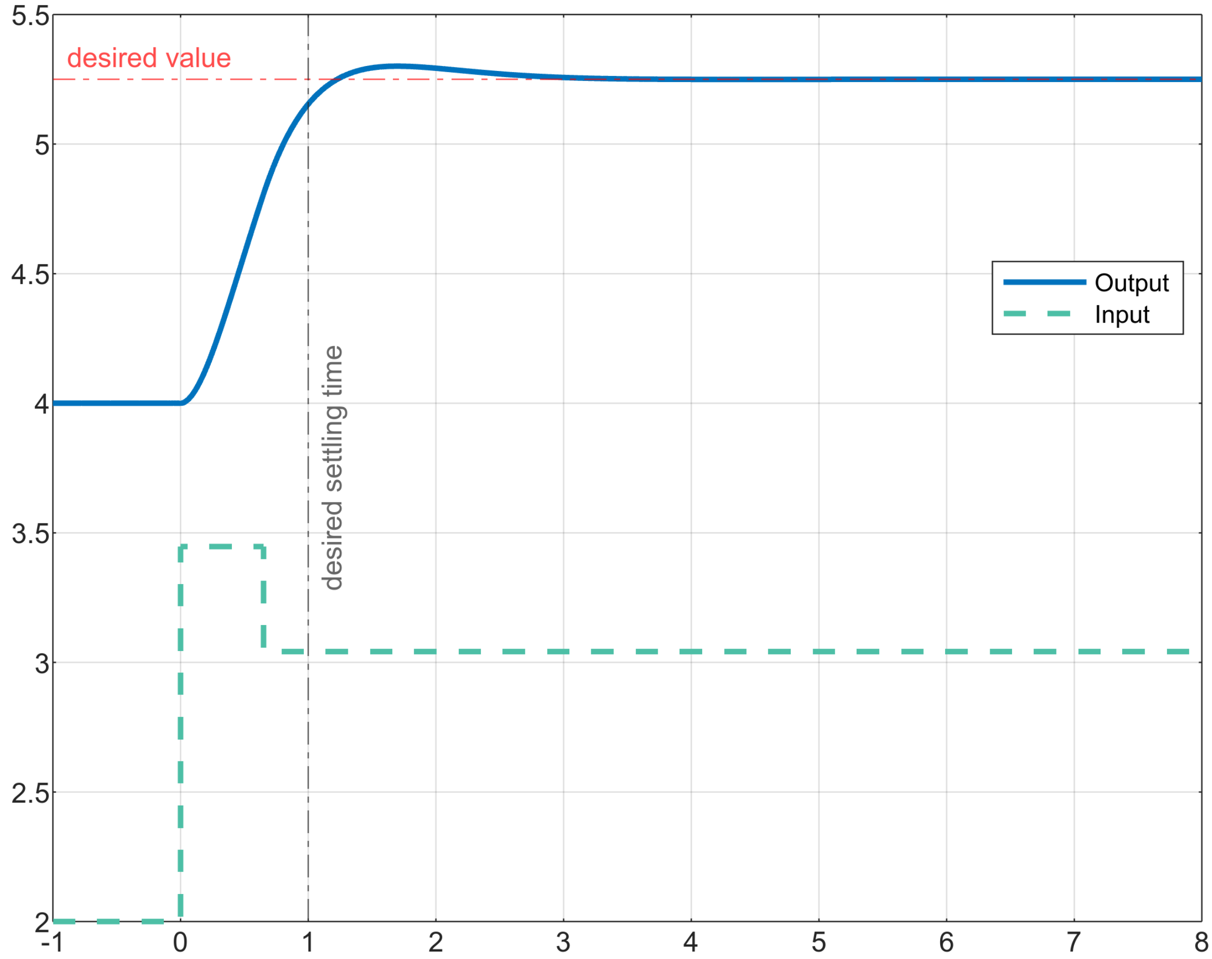
```
u=@(t) u_op+inc_input_computed2*(t>=0);
simulsystem(u);
```

```
yline(5.25,'-.r',Label="desired value",LabelHorizontalAlignment="left")
xline(t_set_desired,'-.',Label="desired settling time",LabelVerticalAlignment="bottom")
legend("Output","Input",Location="best")
```



In order to avoid exceeding the desired final value, it will be necessary to switch to the input value that maintains the desired value in equilibrium once it is reached:

```
switch_factor=0.65; %we'll test some early-switching heuristics later on.
t_switch=t_set_desired*switch_factor;
u=@(t) u_op+inc_input_computed2*(t>=0).*(t<=t_switch)+computed_inc_input*(t>t_switch);
simulsystem(u);
yline(5.25,'-.r',Label="desired value",LabelHorizontalAlignment="left")
xline(t_set_desired,'-.',Label="desired settling time",LabelVerticalAlignment="middle")
legend("Output","Input",Location="best")
```



That is, we have designed a two-step input profile: an initial "boost" to climb up faster and then a "final" steady-state value to stay at the desired point. This is a "precomputed" input profile (open-loop control): no measurement is taken while the step sequence is being applied in order to decide when to switch.

NOTE: the computations we made are only "accurate" in **first-order linear systems**; in higher order linear systems there is a certain "inertia" that will mean that even if the input is lowered to the calculated equilibrium point, there will be a certain "transient overshoot".

The generalisation of these ideas gives rise to "bang-bang or bang-off-bang optimal control", "deadbeat" control, etc., outside of the scope of this introductory material.

Appendix: auxiliary functions

This code is, supposedly, "*secret*": it's not needed to examine it in order to carry out the computations intended to be the goal of this material. This is a sort of abstraction of doing an "experiment":

```
function dxdt=model1(x,u)
 A=[0 1;-5 -3.8];B=[0;6];
 dxdt=A*x+B*u+[0;8];
end

function Y=simulsystem(u)
opts=odeset(Reltol=1e-5,AbsTol=1e-5);
[T,X]=ode45(@(t,x) model1(x,u(t)),[-1 8],[4;0],opts);
Y=X(:,1);
plot(T,Y,LineWidth=2), grid on
hold on
plot(T,u(T'),'--',LineWidth=2,Color=[.3 .75 .65])
hold off
end
```