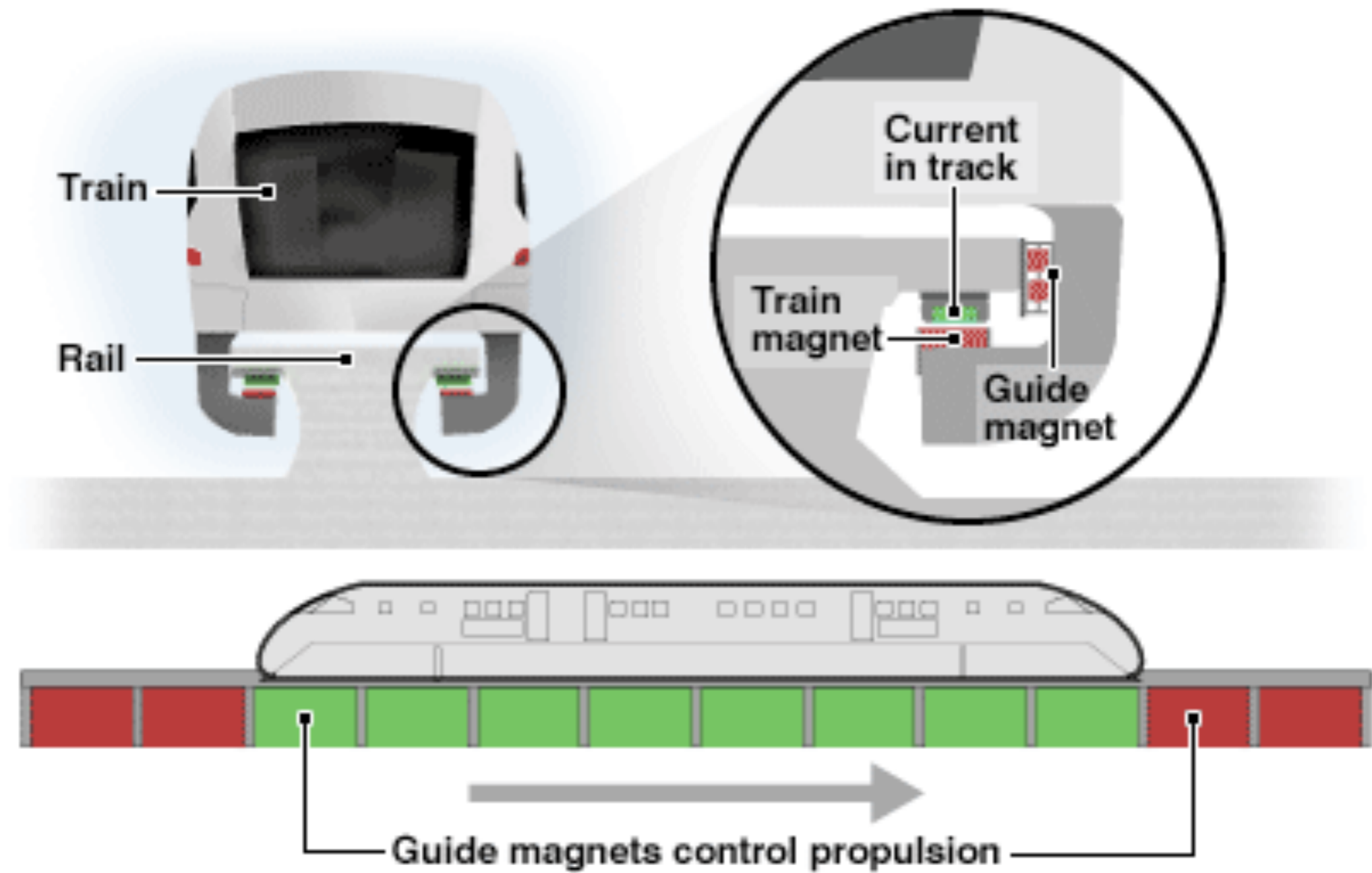


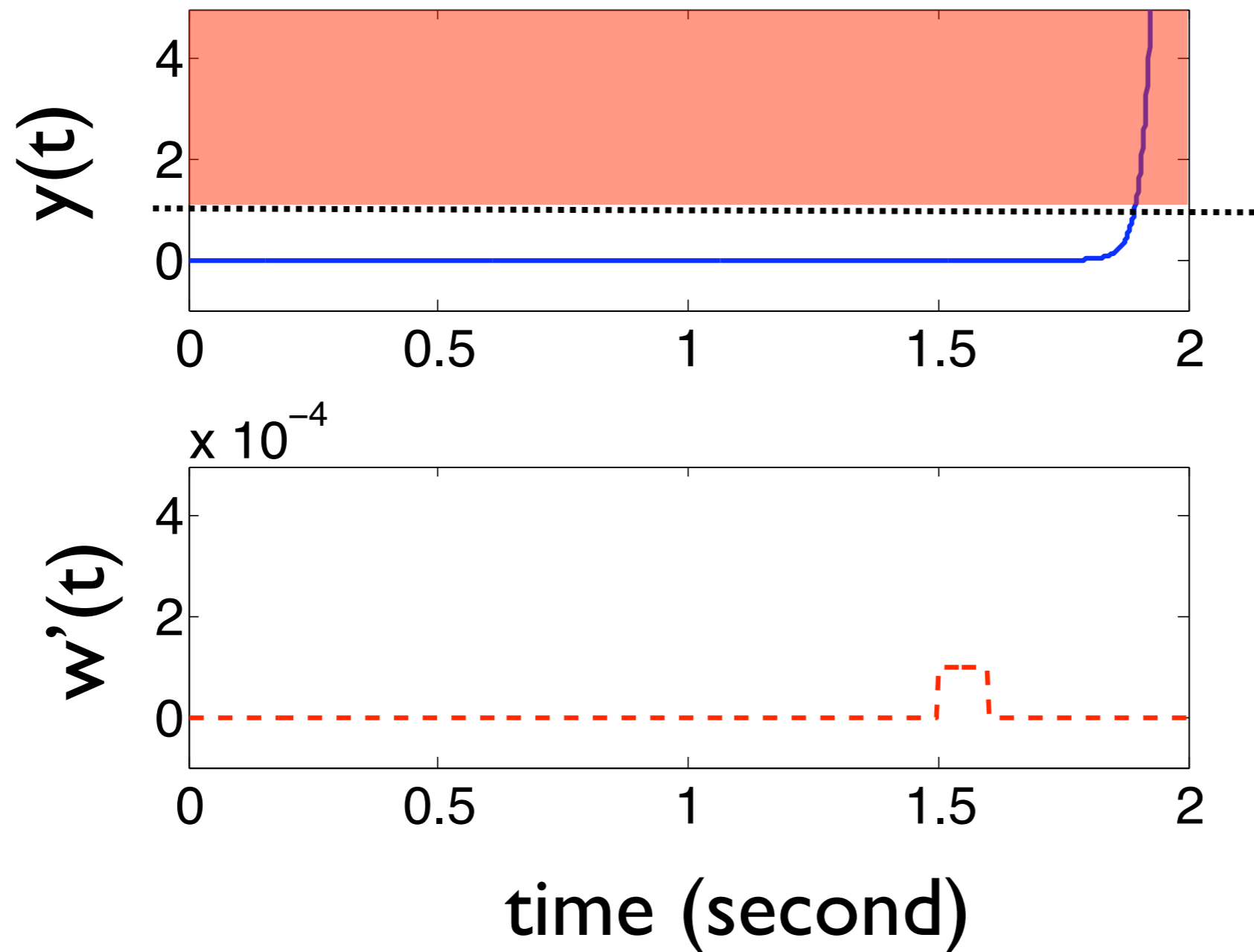
Maglev



Simulation (no control)

$$\ddot{y} = ay + bu + w'$$

$$u(t) \equiv 0$$



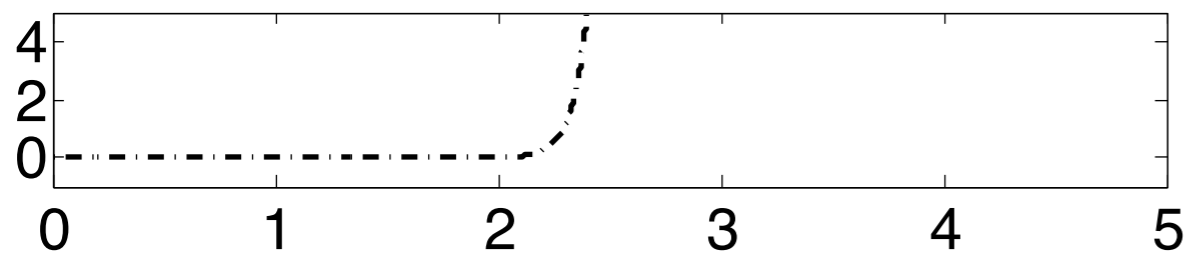
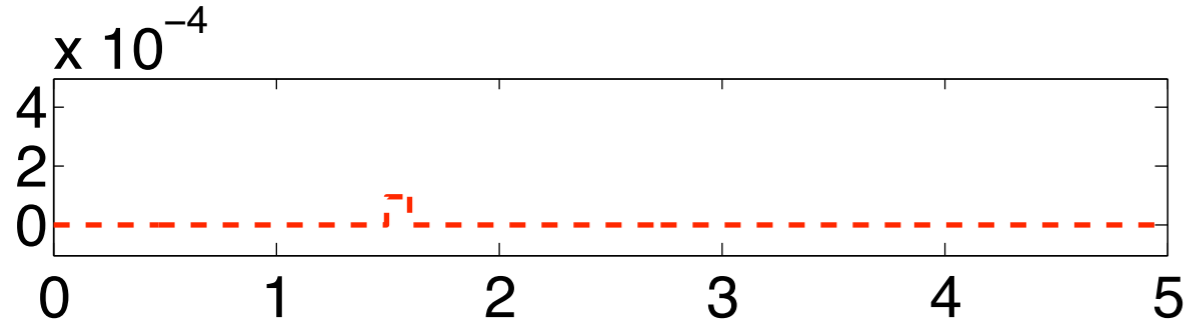
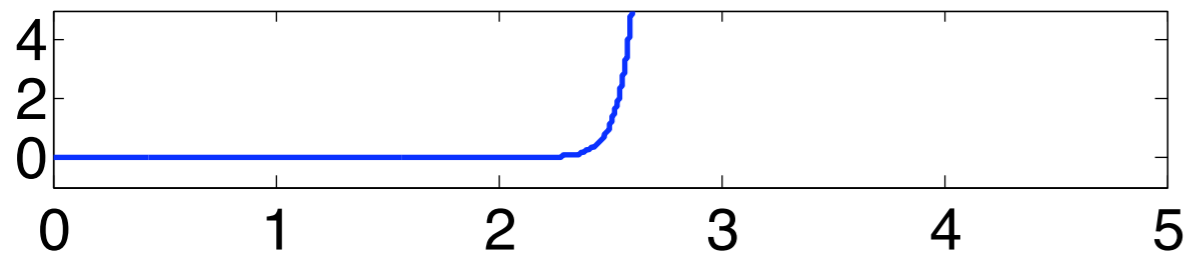
```
g = 9.81;  
h_0 = 0.008; %  
meter  
m = 750; %kg  
mu_0 = 4*pi*1e-7;  
A = 0.021;  
N = 324;  
.....  
  
alpha = 6.9256e-04  
I_0 = 26.0751
```

Let's try feedback control !

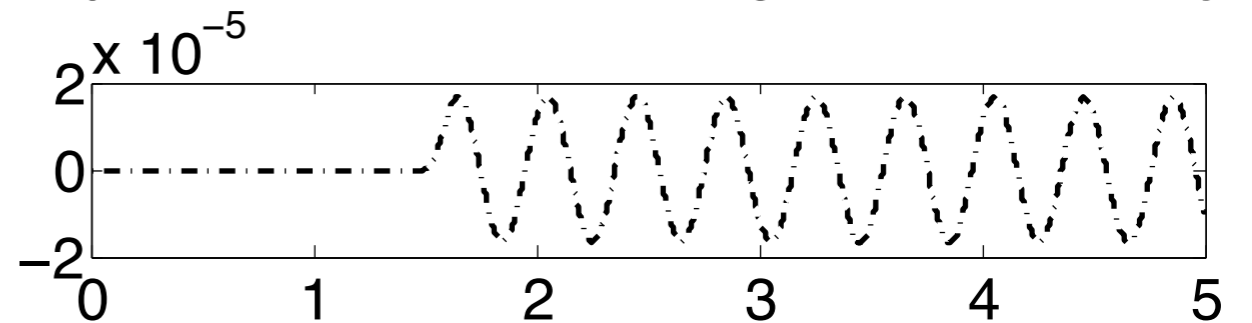
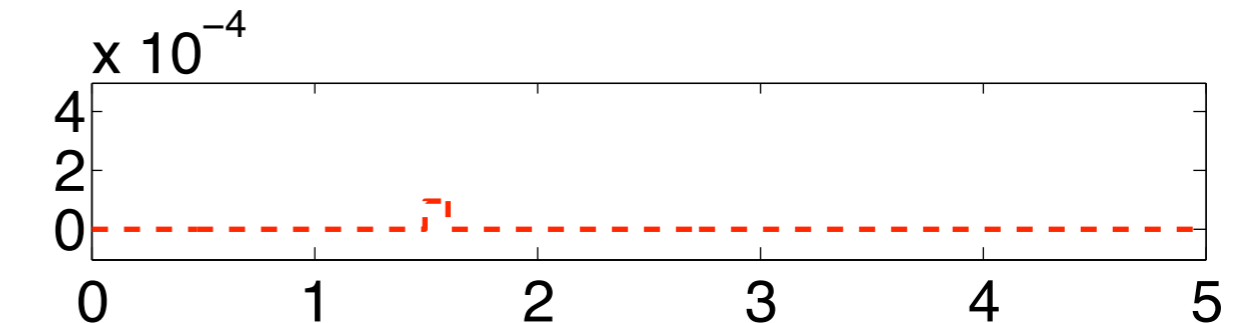
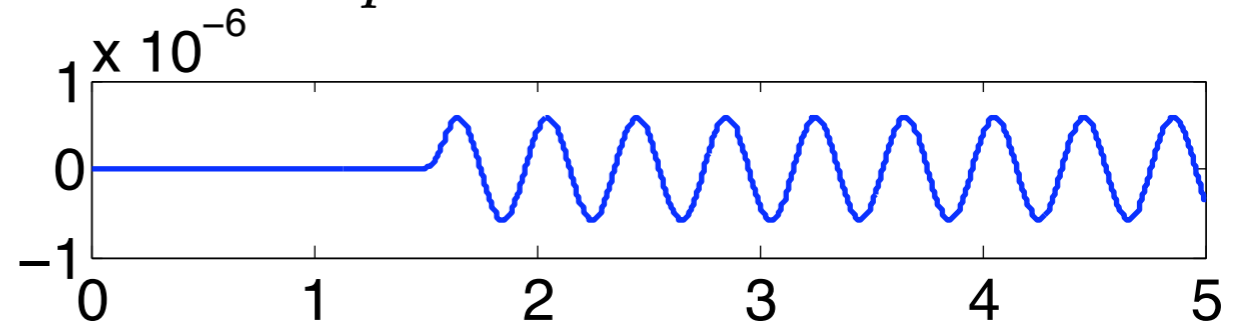
with Proportional control law

$$\ddot{y} + (bK_p - a)y = w'$$

$$bK_p - a = -245.25$$



$$bK_p - a = 245.25$$



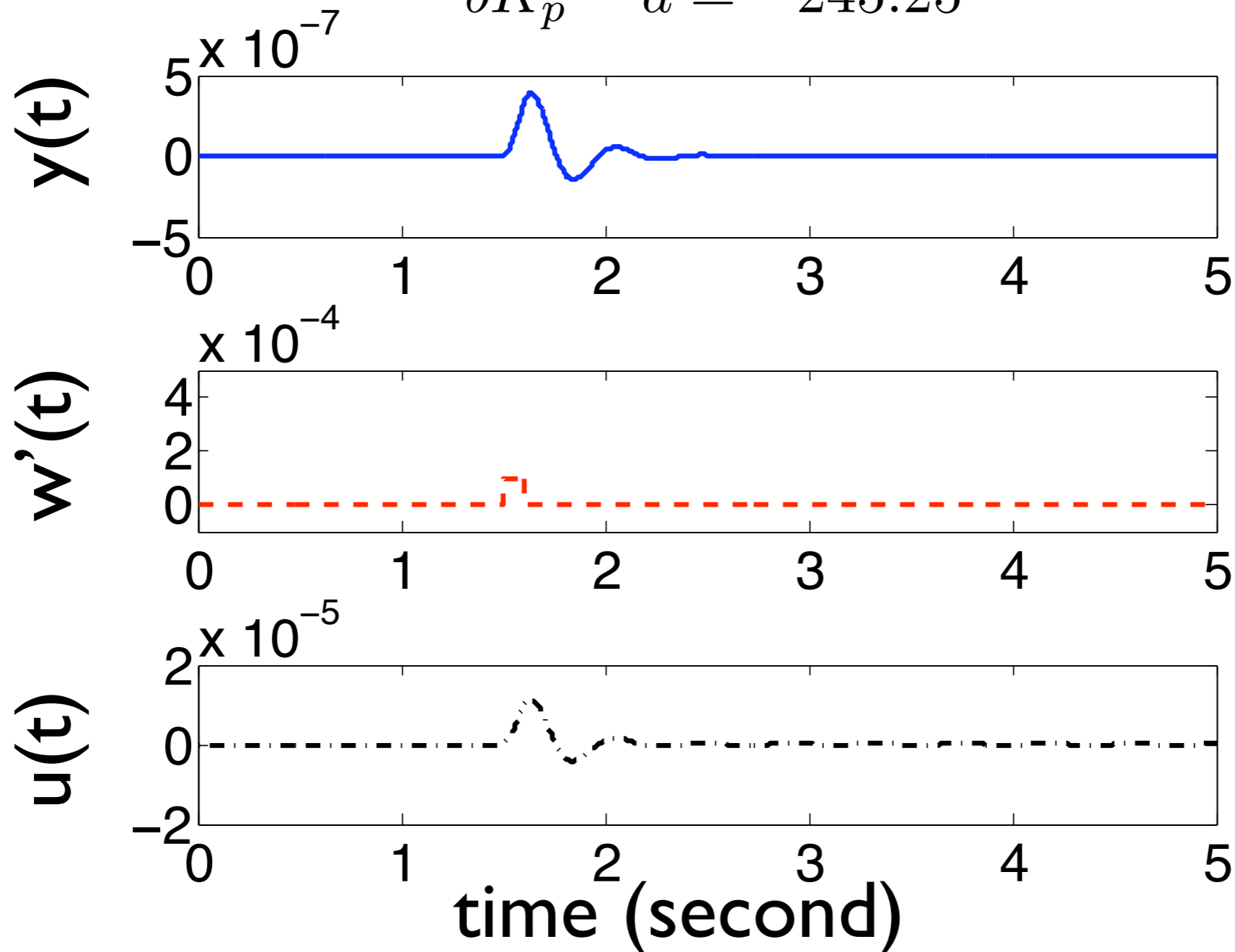
with Proportional Derivative control law

$$\ddot{y} + K_d \dot{y} + (bK_p - a)y = w'$$

$$K_d = 0.1$$

$$bK_p - a = 245.25$$

Good !



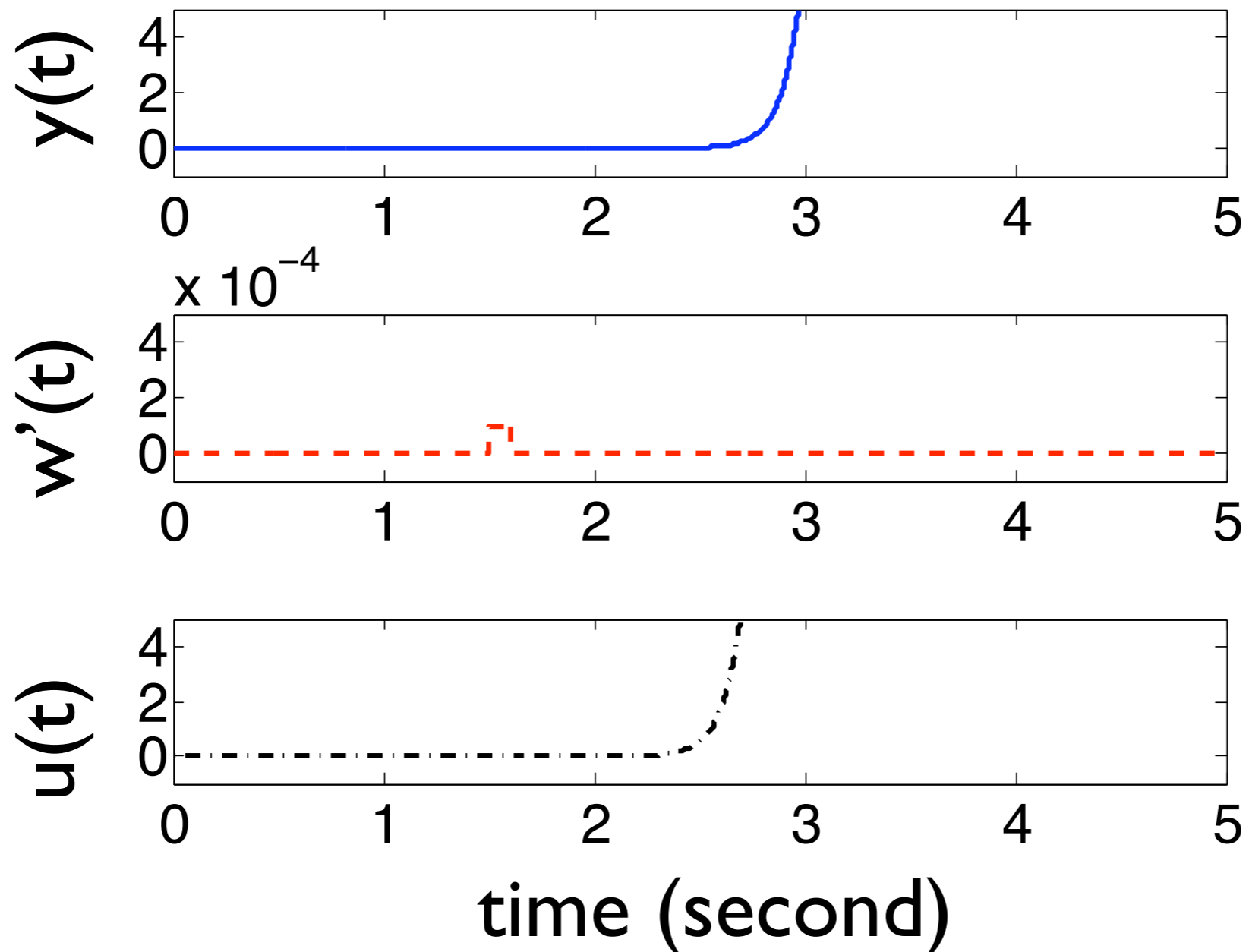
with Proportional Derivative control law

$$\ddot{y} + K_d \dot{y} + (bK_p - a)y = w'$$

$$K_d = 0.1$$

$$bK_p - a = -245.25$$

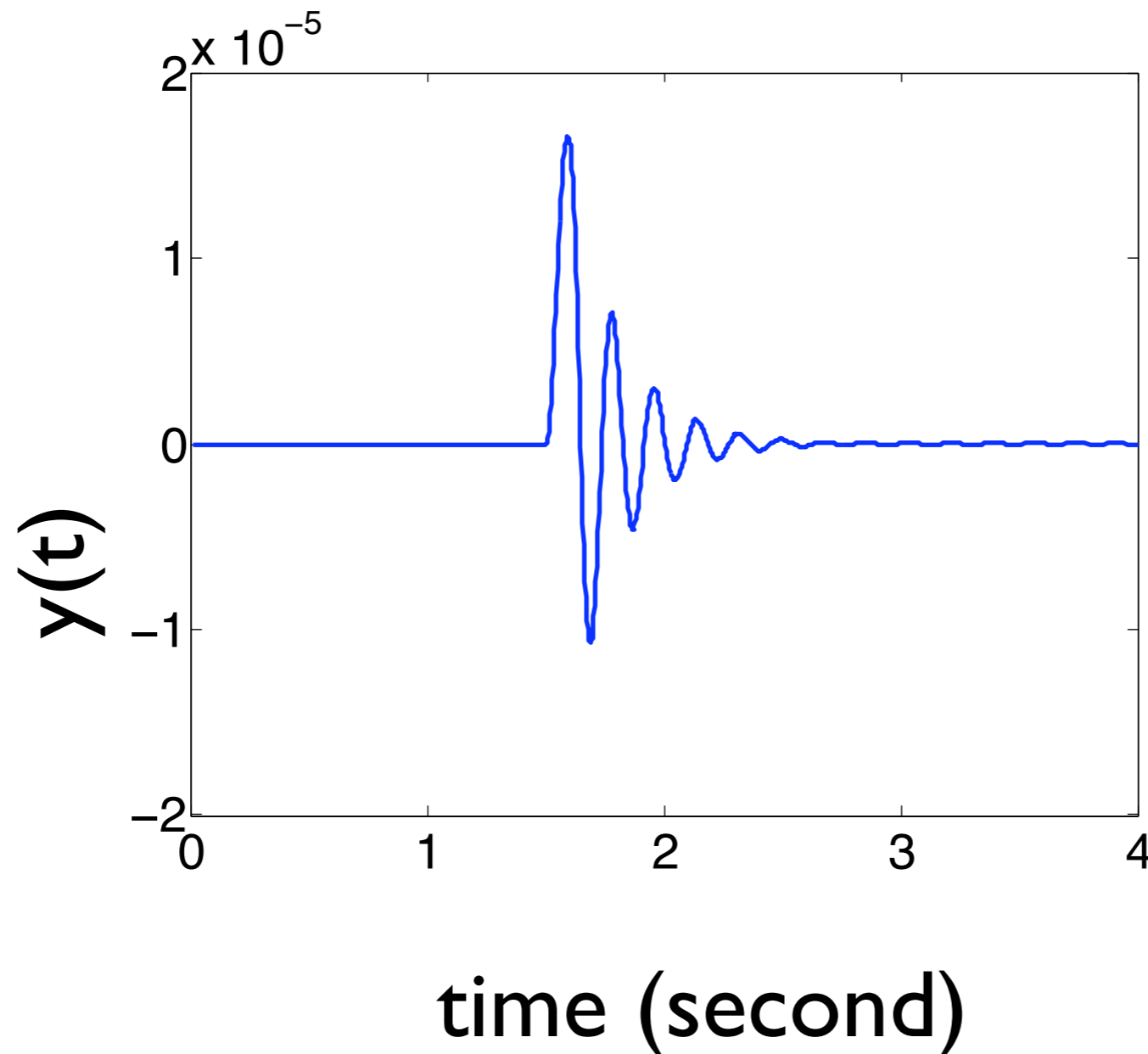
Still bad!



PD control law with the “real” system

$$m\ddot{y} = \frac{mg}{h_0} - \frac{mg}{h_0} \frac{\left(1 - \frac{u}{I_0}\right)^2}{(1+y)^2}$$

$$u = K_p(-y) + K_d(-\dot{y})$$



```

/*****/
/* timer0 interrupt service routine */
/*****/
void isr_t0()
{
.
.
/*****/
/***** READ IN DATA FROM DS2001: CH2 BAD *****/
start_ds2001(DS2001_1_BASE);
y1 = ds2001(DS2001_1_BASE,3); /*valve 1, kulite? */
.
/* ref signal selection 1, 2, 3 */
ref_sig = ext_ref*(ref_sig_select==1) + y1*(ref_sig_select==2) + y2*(ref_sig_select==3);
.
.
.
/* compute control command */
commanded_shaft_rpm=modulation_frequency*INV_NUM_OF_HOLES*60;
sw_rpm_command=commanded_shaft_rpm*INV_RPM_PER_VOLT*(STOP_MOTOR==0);
.
.
.
/*****send outptus - valve control*****/
.
.
cmd2valve1 = (cmd2valve1>0)*cmd2valve1;
phs_bus[motor1ch] = DACSCALE * cmd2valve1;
.
.
end_isr_t0();
/* end of interrupt service routine */
}

void main()
{
#include "spvInitVarsEKF_JN.c" /* initializes variables for EKF */
init(); /* initialize hardware system */
init_data_log(); /* initialize data logging with dRAM*/
label_dram(label_fn); /* label_fn is user-defined */

ds2003_init(DS2003_1_BASE, numofinCHANNELS, inCHANNELS, rng, res); /* intialize 2003 A/D board */
.
start_isr_t0(SAMPLING_TIME); /* call interrupt routine */

while (*error==NO_ERROR)
{
CPU_IDLE(); /* minimize ISR start jitter */
.
.
}
}

```