Modeling & Simulation of Dynamic Systems
On-line Parameter Estimation

Gitopoulos Giorgos, 9344

Electrical & Computer Engineering AUTh

1. Introduction

This project deals with **on-line parameter estimation** of dynamic systems. Some applications, require system modeling in real time, during its run-time (e.g. adaptive control, prediction and fault detection-identification). So, real-time unknown parameter estimation methods are required. Two of these methods are presented in this project, **Gradient Descent** and **Lyapunov Method**. The simulations were implemented in *MATLAB* and the relative code can be found **here**.

2. Gradient Descent Method

2.1. Estimator design

Firstly, we will use **Gradient Descent Method**. Consider the system that is described by the following differential equation:

$$\dot{y} = -ay + bu, \;\; y(0) = 0 \tag{1}$$

where the parameters a and b are constant but unknown, and the input u and output y are available for measurement. We assume that $a > 0$ and $u \in L_\infty$ so that $y \in L_\infty$. The objective is to generate an adaptive law for estimating a and b on-line by using the observed signals $u(t)$ and $y(t)$. We also assume that $u = 5sin(3t)$ and the real values of the unknown parameters are $a = 2$ and $b = 1$.
In order to use Gradient Descent, we need to transform the system to linear parametric model:

$$y = \theta^{*T}\phi \tag{2}$$

If we add and subtract in (1) the term $a_m y$, where a_m is an arbitrary design constant, we have

$$\dot{y} = a_m y - a_m y - ay + bu \Rightarrow$$

$$a_m y + \dot{y} = (a_m - a)y + bu \Rightarrow$$
$$sY + a_m Y = (a_m - a)Y = bU \Rightarrow$$

$$Y = \frac{1}{s + a_m}\left[(a_m - a)Y + bU\right] \tag{3}$$

If $\theta^* = [a_m - a \quad b]^T$ and $\Phi = \left[\frac{1}{s+a_m}Y \quad \frac{1}{s+a_m}U\right]^T$, (3) $\Rightarrow Y = \theta^{*T}\Phi \Rightarrow y = \theta^{*T}\phi$. The adaptive law for generating the estimates $\hat{a}$ and $\hat{b}$ of a and b, respectively, is to be driven by the estimation error

$$e = y - \hat{y} \tag{4}$$

where $\hat{y}$ is the estimated value of y formed by using the estimates $\hat{a}$ and $\hat{b}$. The estimation $\hat{y}$ is generated by the equation

$$\dot{\hat{y}} = -\hat{a}\hat{y} + \hat{b}u, \;\; \hat{y}(0) = 0 \tag{5}$$

In the same way as previously, we reach

$$\hat{y} = \hat{\theta}^T\phi \tag{6}$$

where $\hat{\theta} = \left[a_m - \hat{a} \quad \hat{b}\right]^T$.

Our goal is to minimize the estimation error e in order to approach the real system behavior. In this way, we will extract the estimates $\hat{a}$ and $\hat{b}$. We assume that we want to minimize the function $K(\hat{\theta}) = \frac{e^2}{2} = \frac{(y-\hat{\theta}^T\phi)^2}{2}$. We have

$$\nabla K(\hat{\theta}) = -(y - \hat{\theta}^T\phi)\phi = -e\phi \tag{7}$$

The Gradient Descent of function $K(\hat{\theta})$ is

$$\dot{\hat{\theta}} = -\gamma \nabla K(\hat{\theta}) \tag{8}$$

with $\gamma > 0$, small enough, so from (7) and (8) we take

$$\begin{cases} \dot{\hat{\theta}}_1 = \gamma e \phi_1, & \gamma > 0 \\ \dot{\hat{\theta}}_2 = \gamma e \phi_2, & \gamma > 0 \end{cases} \tag{9}$$

Also, applying the Inverse Laplace Transformation to Φ, we reach

$$\begin{cases} \dot{\hat{\phi}}_1 = -a_m \phi_1 + y, & \phi_1(0) = 0 \\ \dot{\hat{\phi}}_2 = -a_m \phi_2 + u, & \phi_2(0) = 0 \end{cases} \tag{10}$$

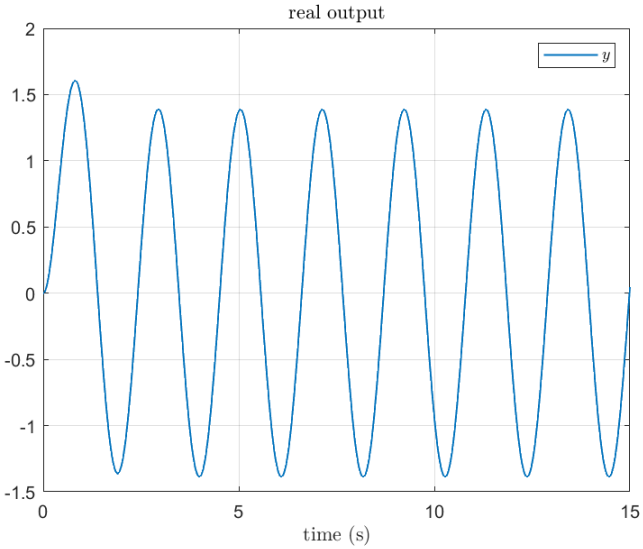
Moreover,

$$(5) \Rightarrow \dot{\hat{y}} = (\hat{\theta}_1 - a_m)\hat{y} + \hat{\theta}_2 u \tag{11}$$

Now, we have designed an **on-line parameter estimator** that consists of the differential equations (9), (10) and (11). The parameters a_m and γ have to be selected. By solving this system in every sampling moment, we export the real-time estimation $\hat{\theta} = \begin{bmatrix} \hat{\theta}_1 & \hat{\theta}_2 \end{bmatrix}^T$, which gives us $\hat{a}$ and $\hat{b}$.

2.2. Simulation

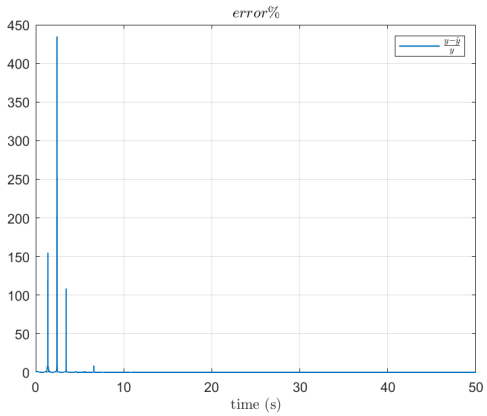
We simulated the described method using *MATLAB*. We used *ode45* function to solve the differential equations system in every sampling moment and produced the estimates $\hat{a}$ and $\hat{b}$. Firstly, we present the real system output during the first 15 seconds, with sampling step 0.01:

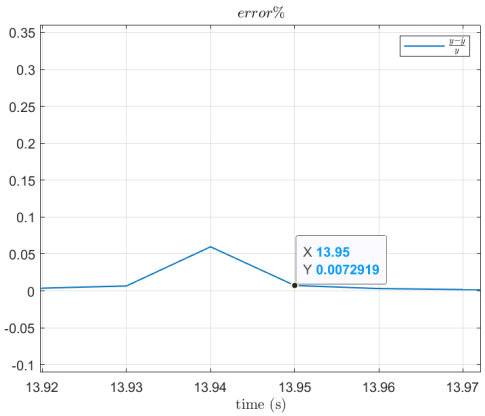


Now, we need to select the parameters a_m and γ. We will use two separated metrics to evaluate the results and select the optimal parameters in each case.

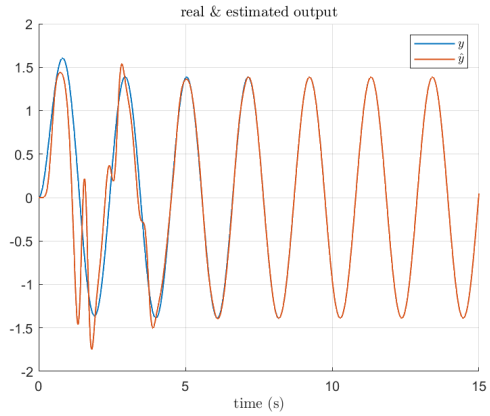
2.2.1. Settling time

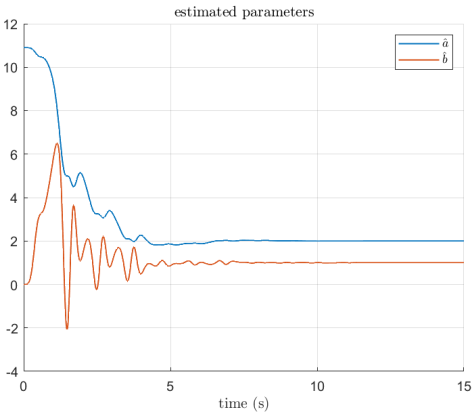
Settling time is the time required for an output to reach and remain within a given error band. We choose the 5% error band, so the settling time is the required time for the percentage error $e_{\%} = \left| \frac{y - \hat{y}}{y} \right|$ to reach and remain under the value 0.05. This method, should be preferred, when we need as fast as possible approach to the real system, while there is not any problem if the error is quite large before the settling time. After tests, we found the optimal combination $a_m = 10.3$ and $\gamma = 102$ that leads to settling time equal to 13.95 seconds. The percentage error for the selected parameters is presented below:

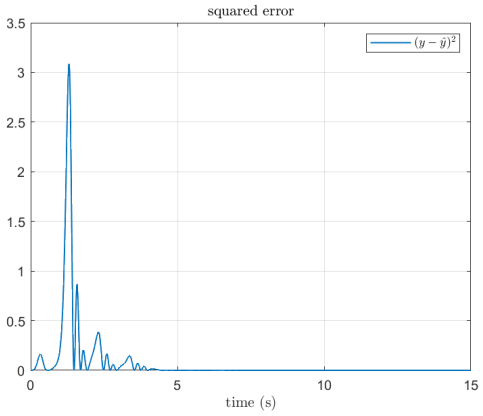




We can notice that before the settling time, the error takes large values, but the real output is approached very fast. For the same parameters, we present the estimated output, the estimated parameters and the squared output error in the next figures:



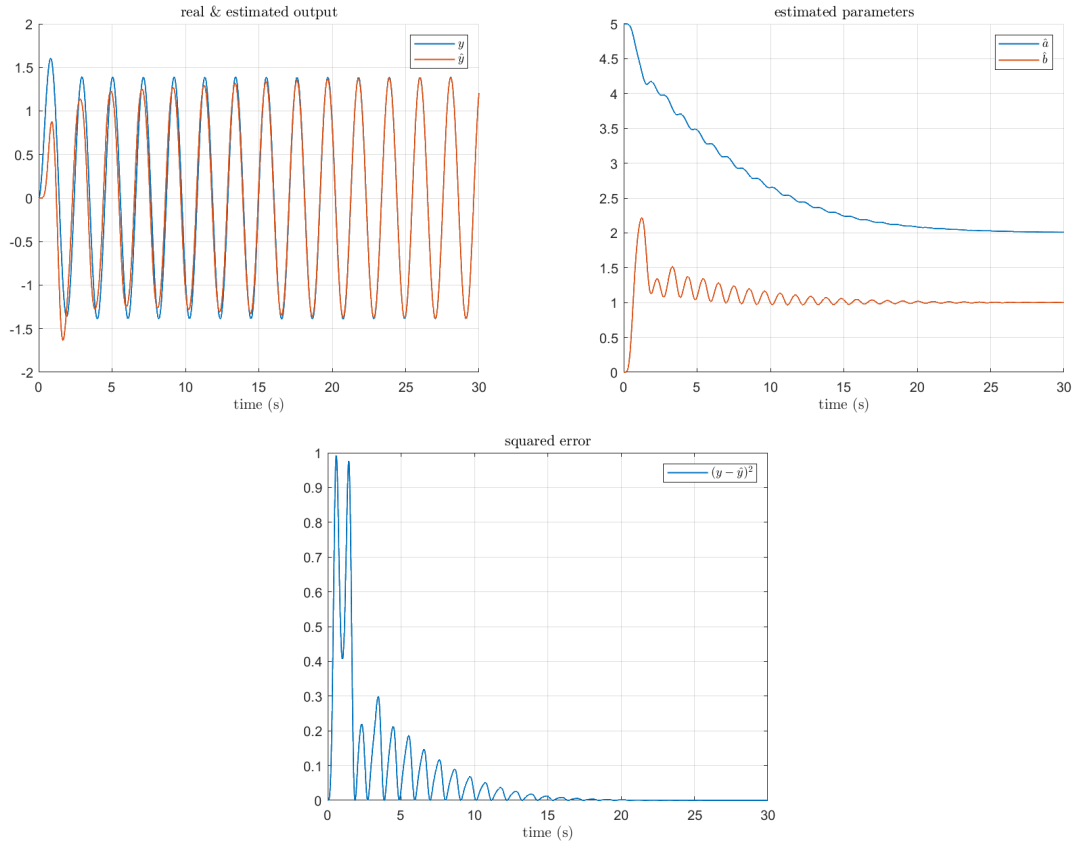




As we see, after some time, real and estimated output become optically identical, while $\hat{a}$ and $\hat{b}$ approach their real values $a = 2$ and $b = 1$.

2.2.2. *Max squared error*

While settling time provides very fast approach to the real system, its large error values until this happens can lead to serious problems in some systems. So, if we are more concerned about a stable estimation from the beginning of the process we can focus on minimizing the max squared error. After some tests, we selected $\gamma = 4.6$ and $a_m = 5$. The results are presented below:



Both the estimated output and the estimated parameters approach in a more stable but slower way to the real ones, compared to the previous results. The maximum squared error is upper-bounded by the value 1 (in the previous case it exceeded 3), while the settling time is equal to 29.68 seconds.

3. Lyapunov Method

In this section, we will design two real-time unknown parameter estimators for the system of the previous section, using **Lyapunov Theorem**. The first one, is based on **parallel model configuration (P)**, where the estimation model does not use the real output and the second one is based on **series-parallel model configuration (SP)**, where the estimation model uses the output of the real system.

3.1. Parallel model (P)

We assume that the real system is described by the equation

$$\dot{y} = -\theta_1^* y + \theta_2^* u, \quad y(0) = 0 \tag{12}$$

where $\theta_1^* = a$ and $\theta_2^* = b$. The estimation model is the following:

$$\dot{\hat{y}} = -\hat{\theta}_1 \hat{y} + \hat{\theta}_2 u, \quad \hat{y}(0) = 0 \tag{13}$$

with $\hat{\theta}_1 = \hat{a}$ and $\hat{\theta}_2 = \hat{b}$. We also have the error:

$$e = y - \hat{y} \Rightarrow$$
$$\dot{e} = \dot{y} - \dot{\hat{y}} \Rightarrow$$

$$\dot{e} = -\theta_1^* e + \bar{\theta}_1 \hat{y} - \bar{\theta}_2 u \tag{14}$$

where $\bar{\theta}_1 = \hat{\theta}_1 - \theta_1^*$ and $\bar{\theta}_2 = \hat{\theta}_2 - \theta_2^*$. Now, we assume the Lyapunov function

$$V = \frac{1}{2}e^2 + \frac{1}{2\gamma_1}\bar{\theta}_1^2 + \frac{1}{2\gamma_2}\bar{\theta}_2^2 \geq 0, \quad \gamma_1 > 0,\ \gamma_2 > 0 \tag{15}$$

While V approaches its equilibrium point $V = 0$, the estimation model approaches the real system. From (15) we take

$$\dot{V} = e\dot{e} + \frac{1}{\gamma_1}\bar{\theta}_1\dot{\hat{\theta}}_1 + \frac{1}{\gamma_2}\bar{\theta}_2\dot{\hat{\theta}}_2 \Rightarrow$$

$$\dot{V} = -e^2\theta_1^* + e\bar{\theta}_1\hat{y} - e\bar{\theta}_2 u + \frac{1}{\gamma_1}\bar{\theta}_1\dot{\hat{\theta}}_1 + \frac{1}{\gamma_2}\bar{\theta}_2\dot{\hat{\theta}}_2$$

We demand $\dot{V} \leq 0$, so we choose

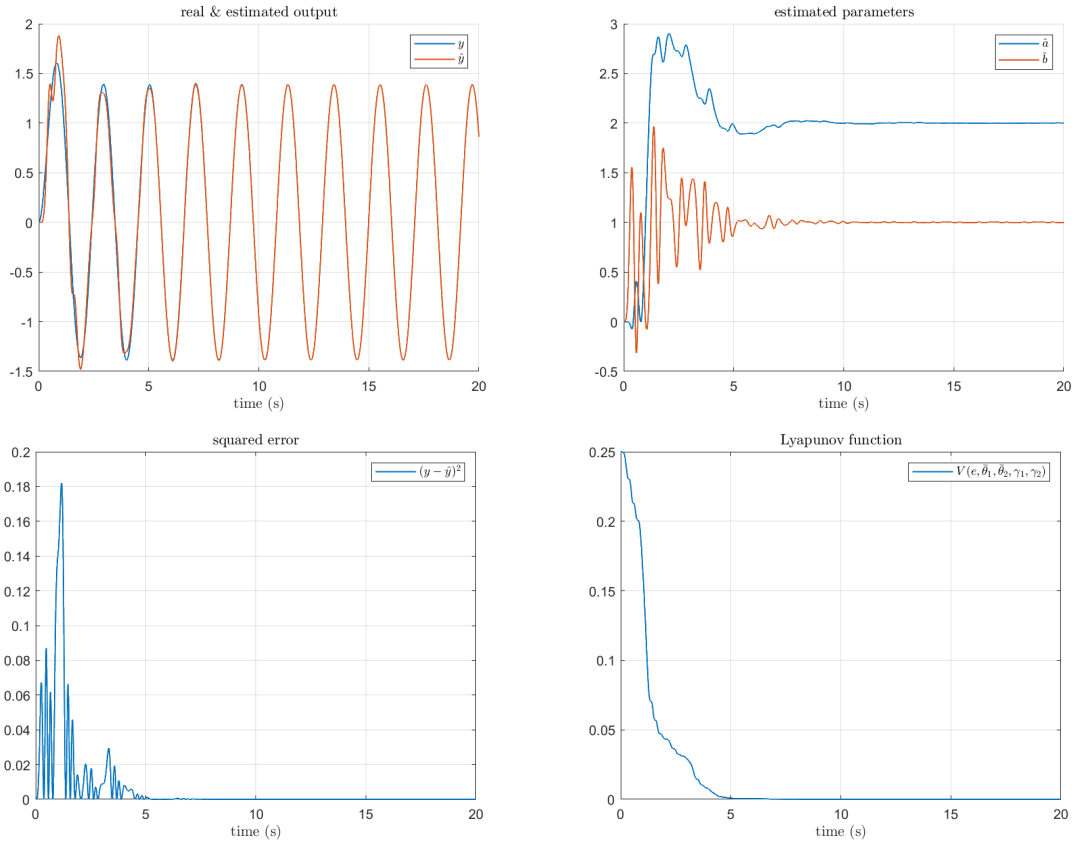
$$\begin{cases} \frac{1}{\gamma_1}\bar{\theta}_1\dot{\hat{\theta}}_1 = -e\bar{\theta}_1\hat{y} \\ \frac{1}{\gamma_2}\bar{\theta}_2\dot{\hat{\theta}}_2 = e\bar{\theta}_2 u \end{cases} \Rightarrow \tag{16}$$

$$\begin{cases} \dot{\hat{\theta}}_1 = -\gamma_1 e\hat{y} \\ \dot{\hat{\theta}}_2 = \gamma_2 eu \end{cases} \tag{17}$$

Now we have

$$\begin{cases} V \geq 0 \\ \dot{V} \leq 0 \end{cases} \tag{18}$$

so from Lyapunov Theorem we take that our estimation model approaches the real system. Our estimator consists of the differential equations (13) and (17). We just have to select parameters γ_1 and γ_2. After some tests simulating the method in *MATLAB*, we selected $\gamma_1 = \gamma_2 = 10$ and we present the results in the next figures:



The estimator approaches the real system with satisfying accuracy and speed. We can notice that Lyapunov function V decreases to 0, verifying our theoretical analysis.

3.2. Series-Parallel model (SP)

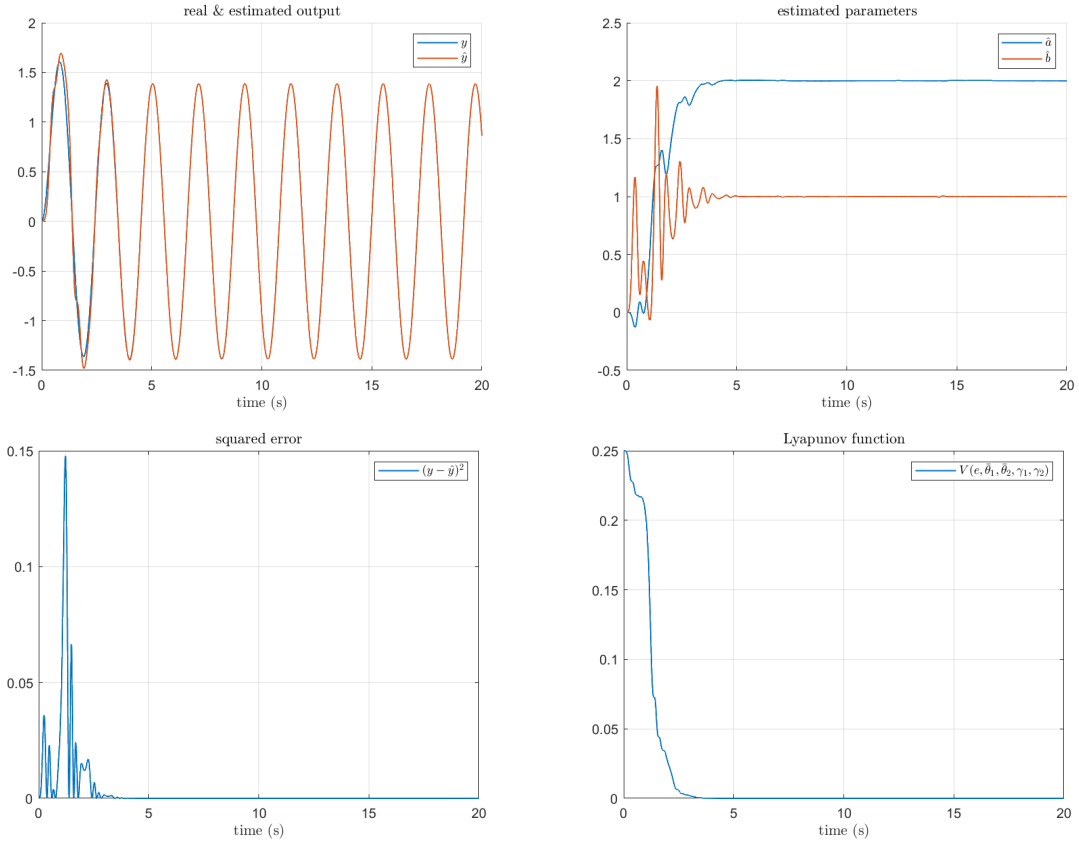
In this method, the estimator is described by

$$\dot{\hat{y}} = -\hat{\theta}_1 y + \hat{\theta}_2 u + \theta_m (y - \hat{y}), \;\; \hat{y}(0) = 0, \; \theta_m > 0 \tag{19}$$

with $\hat{\theta}_1 = \hat{a}$, $\hat{\theta}_2 = \hat{b}$ and θ_m a selected positive parameter. As we mentioned previously, series-parallel model uses the real output measurements y. Working in the exact same way as in the previous subsection and using the Lyapunov function of (15), we reach to the following equations:

$$\begin{cases} \dot{\hat{\theta}}_1 = -\gamma_1 e y \\ \dot{\hat{\theta}}_2 = \gamma_2 e u \end{cases} \tag{20}$$

So, the series-parallel estimator uses (19) and (20) to generate on-line estimations of $\hat{\theta}_1$ and $\hat{\theta}_2$, thus of $\hat{a}$ and $\hat{b}$. We simulated the estimator in *MATLAB* and selected $\gamma_1 = \gamma_2 = \theta_m = 10$. The results are presented below:



3.3. Noise effect

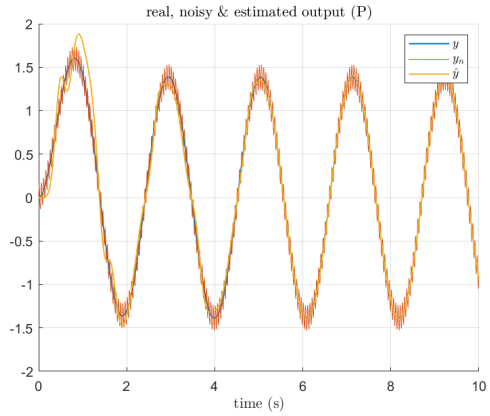
We suppose that noise n is applied in our output measurements:

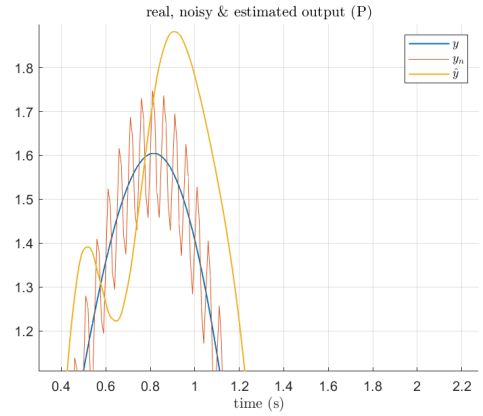
$$n(t) = n_0 sin(2\pi f t)$$

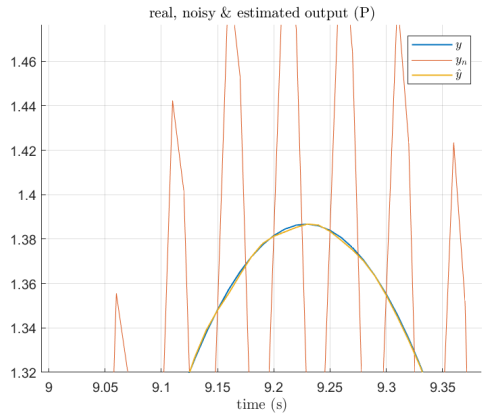
We will evaluate the two estimation models in these conditions, while n_0 increases and f changes. Firstly, we use $n_0 = 0.15$ and $f = 20$. The results are presented below:

3.3.1. $n_0 = 0.15, \quad f = 20$

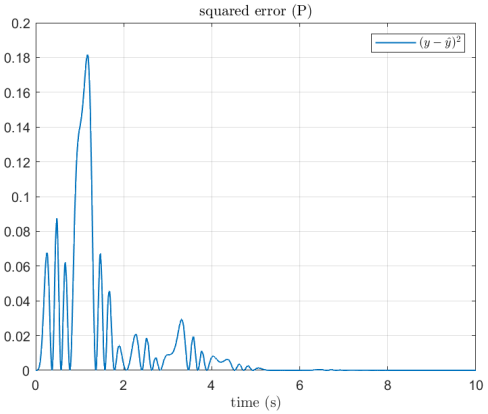
Parallel model

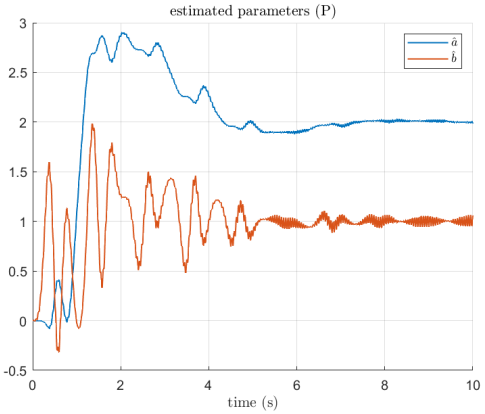




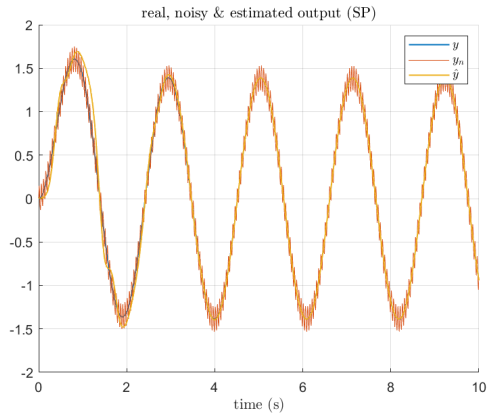


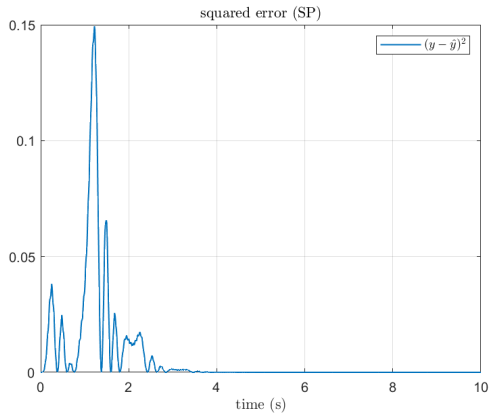
We can notice that at the beginning of the run-time of the **parallel model**, the estimated output presents some error compared to the real one, but the noise does not seem to affect the estimation remarkably. After some time, the estimator almost ignores noise and approaches the real output. The output error seems to decrease to 0, while the parameters estimations oscillate around their real values.

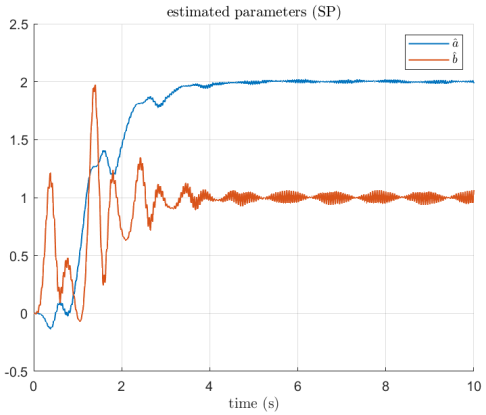




Series-Parallel model



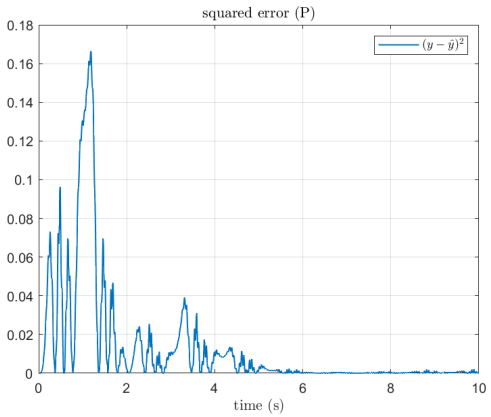


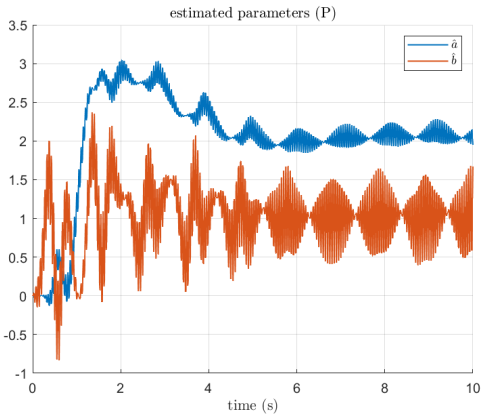


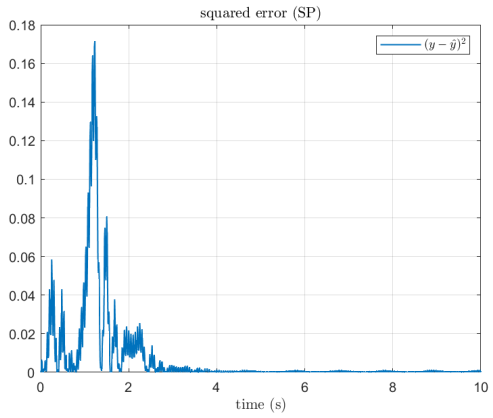
The **series-parallel model** behavior is similar to the **parallel model**. The two models seem to approach each other for weak noise.

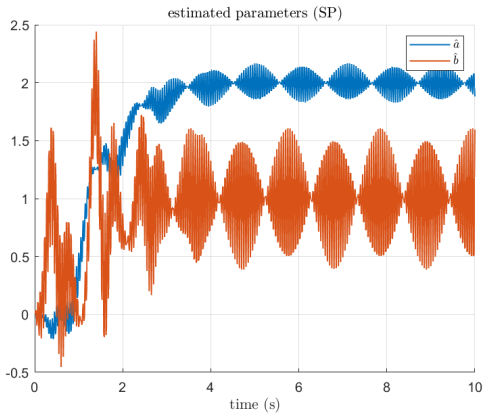
3.3.2. $n_0 = 1.5, \quad f = 20$

We increase amplitude to $n_0 = 1.5$:





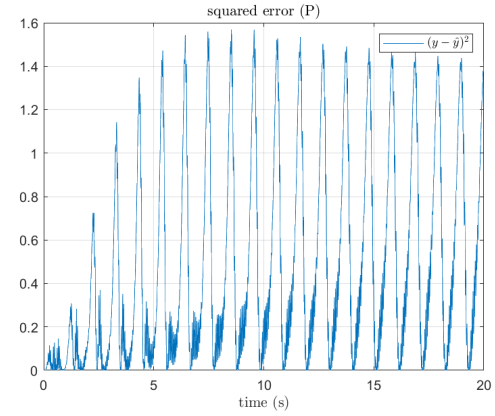


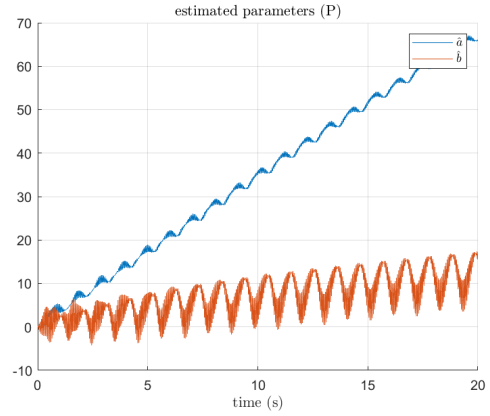


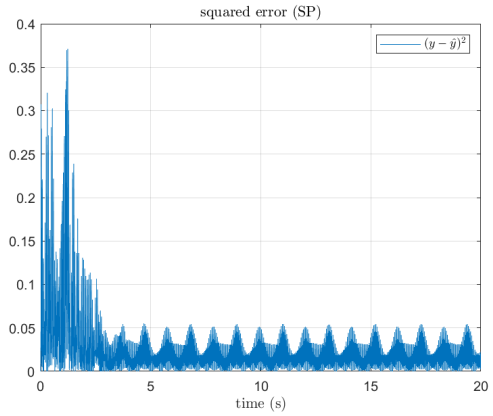
Both models approach the real system, but the oscillation of the parameters around their real values becomes stronger. In SP model the oscillation looks a bit stronger compared with P model.

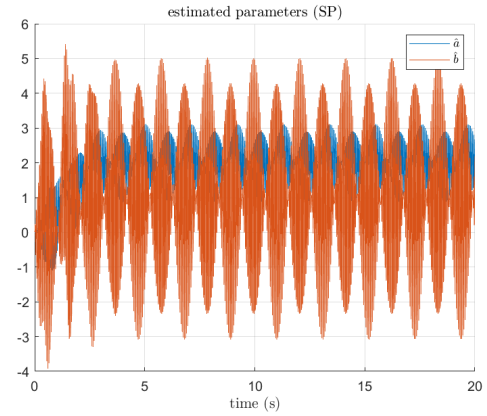
3.3.3. $n_0 = 10, \quad f = 20$

We increase noise even more and we set $n_0 = 10$:





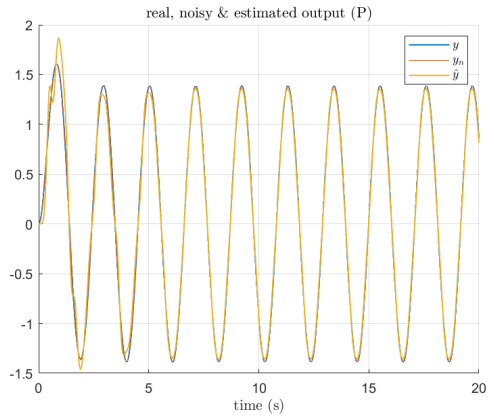


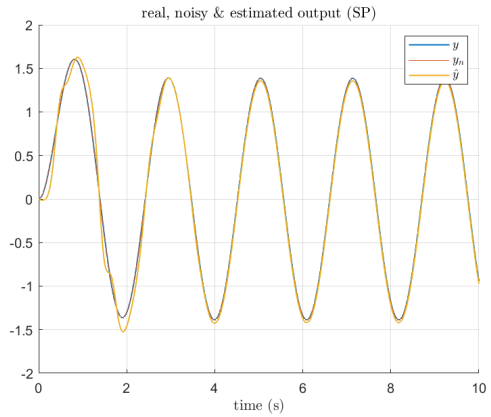


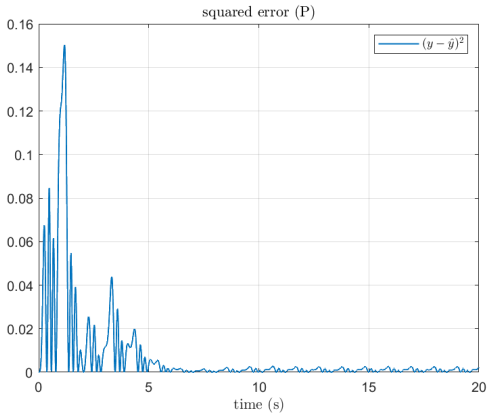
As we can see, **parallel model** is leaded to **instability**, as the error does not decrease and the parameters estimations does not approach the real values. In **series-parallel model**, the error is upper-bounded and the oscillation of the parameters estimation becomes even stronger, but remains around the real values. Overall, we conclude that **series-parallel model** provides **stronger noise endurance**.

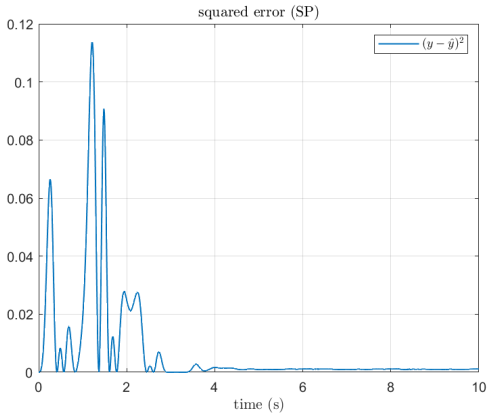
3.3.4. $n_0 = 10, \quad f = 100$

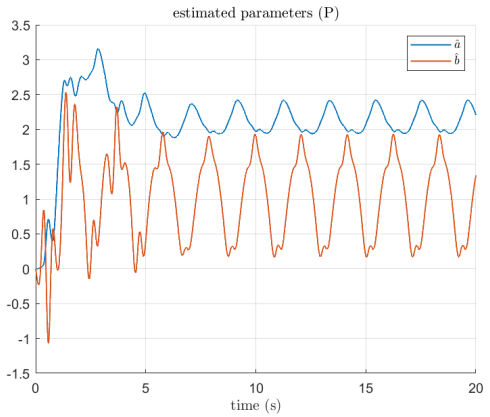
Now, using the value $n_0 = 10$ that causes instability to parallel model, we will increase noise frequency f to value 100:

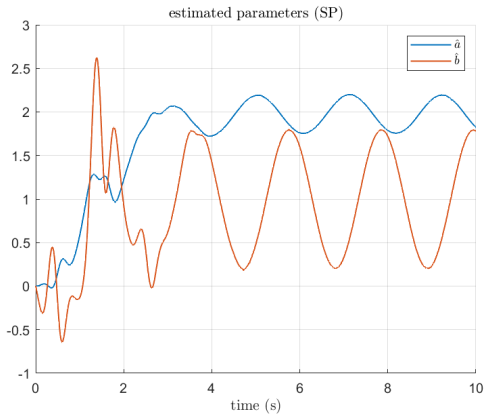












We notice that the instability in parallel model is avoided and replaced by oscillation of the estimated parameters near their real values. Simultaneously, the oscillation in series-parallel model is preserved, but the frequency is reduced. In both cases, the error is upper-bounded, so the models approach the real system despite the effect of the noise.

4. Higher dimension system

In this section, we will use a **parallel model** and applying **Lyapunov method** we will design and simulate an **on-line estimator** of the system

$$\dot{y} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} u, \quad y(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{21}$$

with $u = 10sin(2t) + 5sin(7.5t)$, $a_{11} = -0.25$, $a_{12} = 3$, $a_{21} = -5$, $a_{22} = -1$, $b_1 = 1$ and $b_2 = 2.2$.

If $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ and $B = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$, $(21) \Rightarrow \dot{y} = Ay + Bu$, where A and B are the matrices of the unknown parameters. The parallel estimation model is the following:

$$\dot{\hat{y}} = \hat{A}\hat{y} + \hat{B}u, \quad \hat{y}(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{22}$$

where $\hat{y}$ is the estimated output of the system and $\hat{A}$, $\hat{B}$ are the estimates of the real parameter matrices A and B. Let us consider the output error:

$$e = y - \hat{y} \Rightarrow$$
$$\dot{e} = Ay + Bu - \hat{A}\hat{y} - \hat{B}u \Rightarrow$$
$$\dot{e} = Ae - \bar{A}\hat{y} - \bar{B}u \tag{23}$$

where $\bar{A} = \hat{A} - A$, $\bar{B} = \hat{B} - B$. We select the **Lyapunov function**:

$$V = \frac{1}{2}e^T e + \frac{1}{2\gamma_1} tr\{\bar{A}^T \bar{A}\} + \frac{1}{2\gamma_2} tr\{\bar{B}^T \bar{B}\} \geq 0, \quad \gamma_1 > 0, \ \gamma_2 > 0 \tag{24}$$

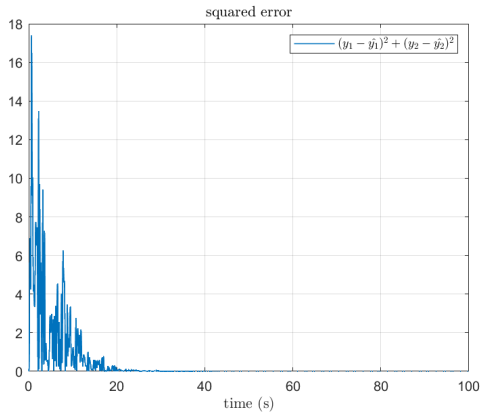
and we have

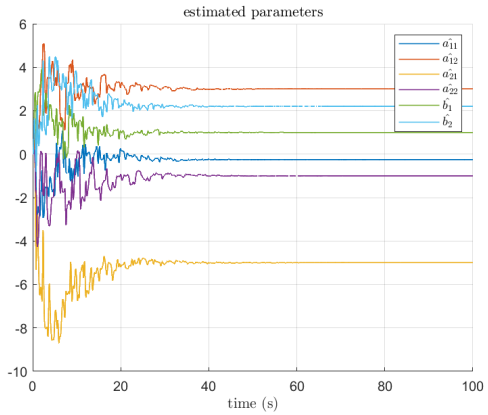
$$\dot{V} = e^T Ae + tr\left\{ - \bar{A}^T e\hat{x}^T - \hat{B}^T eu^T + \frac{1}{\gamma_1}\bar{A}^T\dot{\hat{A}} + \frac{1}{\gamma_2}\bar{B}^T\dot{\hat{B}} \right\}$$

Let us demand $\dot{V} \leq 0$, so we select

$$\begin{cases} \dot{\hat{A}} = \gamma_1 e\hat{x}^T \\ \dot{\hat{B}} = \gamma_2 eu \end{cases} \tag{25}$$

Our estimator consists of (22) and (25). So, the real-time estimates $\hat{A}$, $\hat{B}$ are generated in every sampling moment. We simulated the method in *MATLAB* and evaluated some γ_1 and γ_2 combinations with settling time as a metric. We selected $\gamma_1 = 0.9$ and $\gamma_2 = 0.5$ that provide settling time equal to 79.2 seconds. As settling time, now we declare the required time for $e_{\%} = \sqrt{\left(\frac{y_1 - \hat{y}_1}{y_1}\right)^2 + \left(\frac{y_2 - \hat{y}_2}{y_2}\right)^2}$ to be upper-bounded by 0.05. The results are presented in the next figures:





It is clear that squared error decreases to 0 and parameters estimates approach their real values.

5. Code functionality

All four simulations run in *demo* file with default parameters. Each simulation uses *ode45* function to solve the differential equations of the model, that are given by *msd* functions. Then, some necessary computations take place in each case and the plots are generated. In *demo* file, after each simulation, the plots are generated and the process pauses, until the user presses any key to continue.