# A time domain approach for system identification using hill climbing

Roland Büchi[1*]

[1]Zurich University of Applied Sciences, School of Engineering, Technikumstrasse 9, CH 8401 Winterthur, Switzerland; **roland.buechi@zhaw.ch*

**Abstract.** A large number of methods are known for system identification, which are used both in the time domain and in the frequency domain. In particular, genetic algorithms are increasingly being used today in order to determine the parameters of a model on the basis of measurements. In this article, the related method 'hill climbing' is used together with the least square criterion in order to correctly identify models of small order on the basis of measured step responses in the time domain. It is shown that the algorithm converges well for many starting values and that this method can be applied very well and efficiently for the topic of system identification.

## Introduction

System identification is a field of engineering that is particularly important for parameter identification and modeling. The identification in the time domain has been known in particular since the 20th century, for example in [1]. In the following years and in the course of the rapidly increasing computing power, identification methods with genetic algorithms or particle swarm optimization, PSO became known. In this paper, the 'hill climbing' method [2], [3] is used. This is also a stochastic method for the use in optimizing controllers, and it is related to PSO. The most important difference is that here, the particles are processed indepently to each other. This shows a good probability for finding smaller local minima. Since, in applications in mechanical engineering, calculations are often not made in the frequency domain but in the time domain [4], this approach is used for the parameter identification of such a problem.

Figure 1 shows a torsional oscillator. A spring is clamped on one side and connected to a rotatably mounted disc on the other side. The system is fraught with friction. First of all, it should be assumed that there is no model of it and that one only wants to identify it on the basis of a measurement in the time domain. The system is equipped with an encoder with 2000 steps per revolution. For the measurement, the disk is now turned one turn and then released. The resulting curve in the time domain is shown in figure 2.

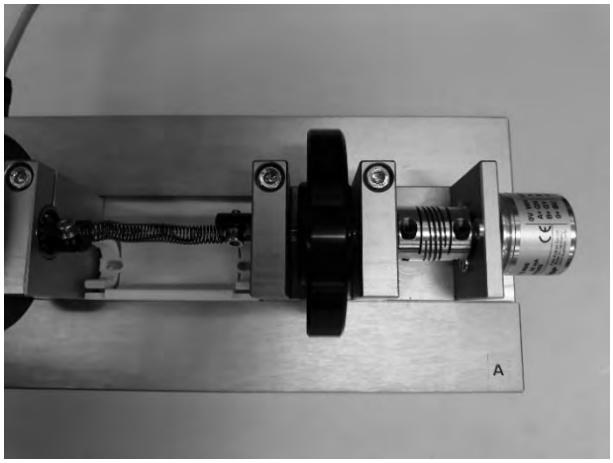


**Figure 1:** Torsional oscillator with spring, disk and encoder

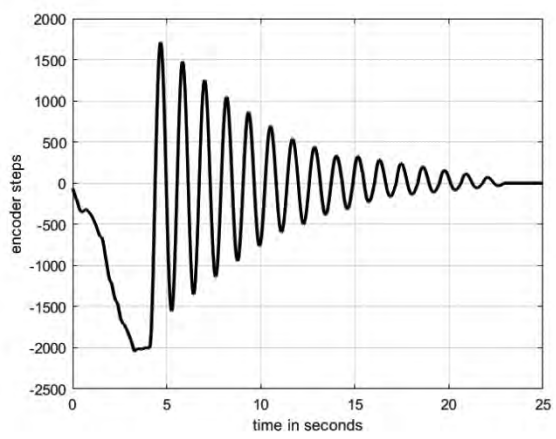


**Figure 2:** Torsional oscillator, measurement of step response

# 1 Identification

One now wants to determine a transfer function of the system $G(s) = \frac{\varphi(s)}{M(s)}$ using the criterion of the Least Square [5]. The output is the angle φ and the input is the torque M. The system has a torque -M0 as an initial condition due to the rotation, and when it is released the torque changes suddenly to 0. This affects the system in this way as if there were a permanent change in the torque from 0 to +M0 at time t = t0.

In a general case, the system can be represented as a transfer function with the Laplace operator 's' as shown in formula 1. First of all, it is assumed that the system model is not known and that all parameters $a_n$ and $b_n$ are different from zero. A 3rd order system is assumed here at the beginning:

$$G(s) = \frac{b_3 \cdot s^3 + b_2 \cdot s^2 + b_1 \cdot s + b_0}{a_3 \cdot s^3 + a_2 \cdot s^2 + a_1 \cdot s + a_0} \qquad (1)$$

This transfer function is the most compact form of system representation. Its step response can now be evaluated in the time domain using Matlab. The evaluation of the transfer function in the time domain with Matlab is also the basis of the following calculations and considerations. In the specific example it is of course clear that the transfer function to be identified in the time domain is of a smaller order, but it is hoped that the corresponding higher order coefficients in the following calculation are so small that the order can be reduced. In the following, it is here still assumed, that there is no physical model of the system. In principle, it would be possible to calculate the step responses of the transfer functions for all parameter combinations a0 to a3 and b0 to b3 and to compare them with the measurement using least square. However, if you consider that you have to calculate the parameters as example from 0 to 10 in steps of as example 1E-4, the result is a huge number of calculations. If the parameter a0 is normalized to 1, then the seven remaining parameters span an seven-dimensional space and with this number of steps there would be 1E5 * 1E7 = 1E12 calculations. The number theory results in a polynomial computational effort with nested loops, but this large number would be far too computationally intensive even with the fastest todays computers. Remember that for each calculation step, the least square criterion would have to be calculated and added up for each measurement and simulation value. The approach with the conventional method, with the calculation with all possible combinations, does not work here.

# 2 Hill Climbing algorithm with Least Square

The hill climbing algorithm is a method that has been known for a long time. Today it is assigned to the methods of artificial intelligence. In recent years, there were also similar methods used for parameter optimization in system identification and control theory [6] – [12]. The idea behind this hill climbing algorithm is simple: With each new calculation step, the smallest parameter change, multiplied by a random sign (+/- 1) or 0, is added to each of the parameters to be changed. Then the new least square is calculated and compared with the previous one. If the new least square is smaller, the new values are retained and assumed as new. If the new least square is larger, the old values are used again as the new. The hill climbing algorithm always works with a heuristic function that changes the parameters. The heuristic function for the shown problem is:

$$f_h = p + randomsign \cdot \Delta p \qquad (2)$$

Where p = [b3, b2, b1, b0, a3, a2] (there is a1 = 1)

The hill climbing method actually only finds local minima, in this case local minima of the least square value. It turns out, however, that with different starting values after many calculations, always the same or very similar parameters are found. This suggests that there are essentially only monotonically falling or rising least square values. This fact would have to be examined more closely in further steps.

The convergence of hill climbing with the search for the least square is also relatively good. For small parameter changes, the author only carries out up to a few million calculation steps, depending on the simulation. It would be possible to define as a termination criterion when the values of the least square no longer change significantly or no longer change at all. Here, too, further calculations are likely to be carried out. However, in practice the parameters found are quite good.

# 3 Results

In a first step, the parameters are calculated for a higher order than assumed; in this example, with the boundary conditions for a parameter change of 1E-4 per calculation step, the best result for the 3rd order system after 10 simulations with randomized starting values and 3'000'000 learning iterations each is:

$$G(s) = \frac{0.0018 \cdot s^3 + 0.0434 \cdot s^2 + 5.884 \cdot s + 6.32}{0.0322 \cdot s^3 + 0.0454 \cdot s^2 + 0.9457 \cdot s + 1} \quad (3)$$

In many representations of the transfer function, the parameter of the highest denominator order, here a3, is normalized to 1. When calculating the parameters with the hill climbing method, however, the author advises normalizing the parameter a0 to 1, since this exists in all systems with a stationary end value. In systems with a free integrator without a stationary end value, this parameter is zero. Otherwise, it would make sense to normalize the parameter a1 to 1.
The convergence of the parameters was also calculated over the iterations. It always behaves similarly over the 10 simulations carried out with different starting values. Figure 3 shows the course. Since all parameters are changed simultaneously using the hill climbing method, the method converges slowly.

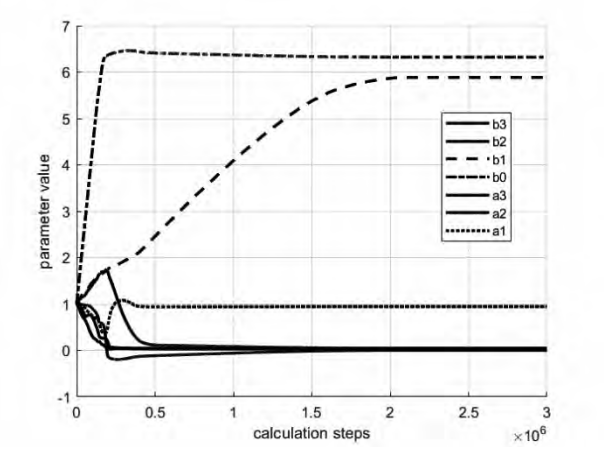


**Figure 3:** parameter convergence for the transfer function of formula 3

## 3.1 Reduction of system order

Now it turns out that the parameters for the higher orders a3 and a2 in the denominator and b3 and b2 in the numerator are much smaller than the other parameters. Another issue is also the reduction of order. It is also a field of science in itself. The literature on it is immense and there is only a very brief introduction given here. The transfer function from formula 3 can also be represented in the pole-zero notation [1]. With this notation according to the pole- zero notation, G(s) can also be represented as in formula 4.

$$G(s) = k \cdot \frac{(s - \beta_3) \cdot (s - \beta_2) \cdot (s - \beta_1)}{(s - \alpha_3) \cdot (s - \alpha_2) \cdot (s - \alpha_1)} \quad (4)$$

For the system order reduction you can now also search for dominant zeros and poles. These are those that are sufficiently close to 0. The non-dominant poles and zeros are those that are sufficiently far away from 0; these can be neglected if necessary. One also has to take into account the sampling time of the systems, because the resulting aliasing also affects the system behavior.
In this example, a zero and a pole are almost identical. One solution of the numerator polynomial is s = -1.0824 and one of the denominator polynomial is s = -1.0707. Such solutions result in identical local minimums of the least square. One can cancel the zeroes. In this case there is an order reduction of the numerator and the denominator by one order each. Another calculation is now carried out in such a way that the corresponding coefficients of the third order, b3 and a3, are set to zero, taking into account that order reduction. The best result after 10 simulations with randomized starting values and 1'500'000 learning iterations each is as follows.

$$G(s) = \frac{0.0033 \cdot s^2 + 0.0493 \cdot s + 6.317}{0.0346 \cdot s^2 + 0.0117 \cdot s + 1} \quad (5)$$

Here, too, it must be discussed whether a further reduction in order can be carried out. The two parameters b2 and b1 are small compared to b0. The modeling of the system that is available in this special case can also be used for discussion. This is shown below.

Assumed that all of this has been taken into account and one came to the conclusion that a3, b3, b2 and b1 can actually be neglected, one can do the calculation again. With the parameter change of 0.0001, the parameters of formula 6 result after 10 Simulations with 500'000 calculation steps. The number of calculation steps decreases, since there are less parameters to change. This results in a faster convergence.

$$G(s) = \frac{6.315}{0.0343 \cdot s^2 + 0.0119 \cdot s + 1} \tag{6}$$

Here, too, the parameters converge in all simulations. Since fewer different parameters are now changed via the hill climbing method, the convergence is much faster than with the formula 3 with more changed parameters.

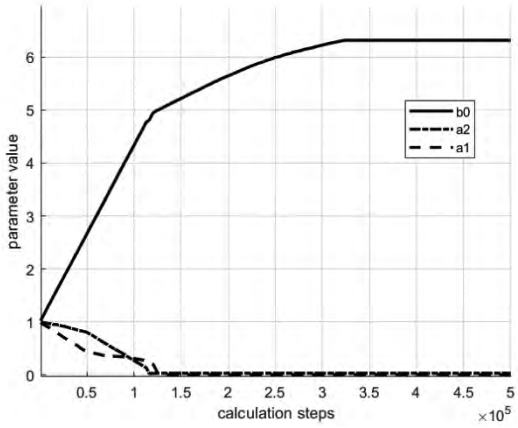


**Figure 4:** parameter convergence for the transfer function of formula 6

The comparison of simulation and measurement is shown in figure 5.

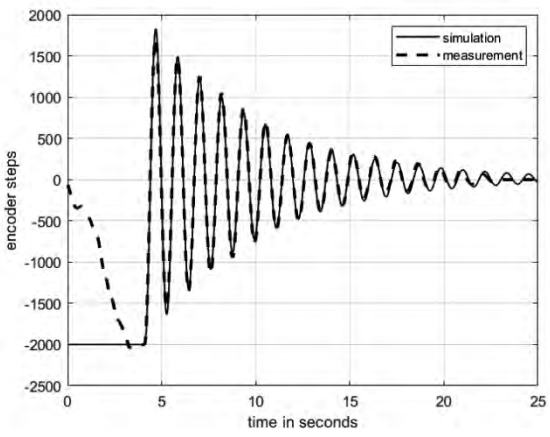


**Figure 5:** Comparison of simulation and measurement for the torsional oscillator

### 3.2 Reduction of system order using the physical model

The method presented here works in the general case without any knowledge of the model. This is how it was carried out up to here, up to the model reduction due to small parameter values or the shortening of poles and zeros. However, if a physical model can be derived, one can reduce the model to the relevant parameters right from the start. According to formula 7:

$$J \cdot \ddot{\varphi} = \sum M = M_0 - b \cdot \dot{\varphi} - c \cdot \varphi \tag{7}$$

Thus, the differential equation also results to:

$$J \cdot \ddot{\varphi} + b \cdot \dot{\varphi} + c \cdot \varphi = M_0 \tag{8}$$

And after the laplace transformation, the transfer function of formula 9 results:

$$\frac{\varphi(s)}{M(s)} = \frac{1}{J \cdot s^2 + b \cdot s + c} = \frac{1/c}{J/c \cdot s^2 + b/c \cdot s + 1} \tag{9}$$

This shows that the parameter or order reduction of formula 6 is correct.
If you take a closer look at the measured step response according to figure 2, it is also noticeable that the step response of the simulation no longer corresponds very well with the measurement, especially after a long time. This is due to the non-linear effect of static friction, which is responsible for the fact that the shaft simply stops after falling below a certain low speed. This small non-linearity is not taken into account in the model.

## 4 Discussion and outlook

It turns out that this is a very applicable and transparently explainable method to carry out the parameter identification in the time domain. The convergence is quite good in the discussed examples, although the orders are not very large. With many systems that occur in practice, however, measurements of sub-systems can also be carried out, which then often have an order that is small

enough that this method can be used. Often there are also several measurements with slightly different step responses. This can be caused by small measurement inaccuracies, for example. This can also be dealt with using this method. The least square method is then simply applied to the individual comparisons with the measurements, one after the other. In this case the least square results will be added up and compared with older added up least square results.
So that the reader can test out examples himself, a Matlab code of a simple example is also printed at the end. This can be copied into an editor. In order to simulate an existing measurement, a transfer function was set at the beginning and its step response was subsequently assumed as a measurement. The hill climbing method also converges well here and delivers the identified parameters as discussed above.

```
clear
ls = 1E20;
% didactic example with given Gmeas which
% replaces Measurement
Gmeas = tf([2 2],[3 2 1])
delta_t = 0.01;      %% sample time
t= 0.01:delta_t:50;  %% time vector
[xmeas,tmeas]= step(Gmeas,t);

b2 = 1; delta_b2 = 0.01;
%% hill climbing parameter
b1 = 1; delta_b1 = 0.01;
%% hill climbing parameter
b0 = 1; delta_b0 = 0.01;
%% hill climbing parameter
a2 = 1; delta_a2 = 0.01;
%% hill climbing parameter
a1 = 1; delta_a1 = 0.01;
%% hill climbing parameter
a0 = 1; delta_a0 = 0.0;
%% hill climbing parameter

for counter = 1:10000;
counter
sign_b2 = fix(5*rand(1))-2;
% [-2...2] heuristic function
b2 = b2 + sign_b2*delta_b2;
sign_b1 = fix(5*rand(1))-2; % [-2...2]
b1 = b1 + sign_b1*delta_b1;
sign_b0 = fix(5*rand(1))-2; % [-2...2]
b0 = b0 + sign_b0*delta_b0;
sign_a2 = fix(5*rand(1))-2; % [-2...2]
a2 = a2 + sign_a2*delta_a2;
sign_a1 = fix(5*rand(1))-2; % [-2...2]
a1 = a1 + sign_a1*delta_a1;
% a0 is kept as 1

G = tf([b2 b1 b0],[a2 a1 a0]);
[x,t1] = step(G,t);

lsnew = leastsquare(x,xmeas,delta_t);
  if (lsnew < ls)
% Backpropagation
      ls = lsnew
  else
      b2 = b2 - sign_b2*delta_b2;
      b1 = b1 - sign_b1*delta_b1;
      b0 = b0 - sign_b0*delta_b0;
      a2 = a2 - sign_a2*delta_a2;
      a1 = a1 - sign_a1*delta_a1;
  end
end %counter

plot(t,x,tmeas,xmeas)

ls = leastsquare(x,xmeas,delta_t);

function y = leastsquare(a,b,deltat)
y = 0;
  for k = 1:length(a);
    y = y + (a(k)-b(k))^2*deltat;
  end
end
```

## References


[1] Unbehauen, Heinz. Regelungstechnik. Braunschweig: Vieweg, 1992.

[2] Skalak, David B. "Prototype and feature selection by sampling and random mutation hill climbing algorithms." Machine Learning Proceedings 1994. Morgan Kaufmann, 1994. 293-301.

[3] Büchi, Roland. "Optimal ITAE Criterion PID Parameters for PTn Plants Found with a Machine Learning Approach." 2021 9th International Conference on Control, Mechatronics and Automation (ICCMA). IEEE, 2021

[4] Chatzis, Manolis N., Eleni N. Chatzi, and Andrew W. Smyth. "An experimental validation of time domain system identification methods with fusion of heterogeneous data." Earthquake Engineering & Structural Dynamics 44.4 (2015): 523-547.

[5] Verhaegen, Michel, and Vincent Verdult. Filtering and system identification: a least squares approach. Cambridge university press, 2007.

[6] Liu, Xiaoyong, Huajing Fang, and Zhaoxu Chen. "A novel cost function based on decomposing least-square support vector machine for Takagi-Sugeno fuzzy system identification." IET Control Theory & Applications 8.5 (2014): 338-347.

[7] Wang, Grace S. "Application of hybrid genetic algorithm to system identification." Structural Control and Health Monitoring: The Official Journal of the International Association for Structural Control and Monitoring and of the European Association for the Control of Structures 16.2 (2009): 125-153.

[8] Orszulik, Ryan R., and Jinjun Shan. "Active vibration control using genetic algorithm-based system identification and positive position feedback." Smart Materials and Structures 21.5 (2012): 055002.

[9] Alfi, Alireza, and Hamidreza Modares. "System identification and control using adaptive particle swarm optimization." Applied Mathematical Modelling 35.3 (2011): 1210-1221.

[10] Bian, Qi, et al. "System identification method for small unmanned helicopter based on improved particle swarm optimization." Journal of Bionic Engineering 13.3 (2016): 504-514

[11] Büchi, Roland. State space control, LQR and observer: step by step introduction with Matlab examples. Norderstedt Books on Demand, 2010.

[12] Razmjooy, Navid, and Mehdi Ramezani. "Training wavelet neural networks using hybrid particle swarm optimization and gravitational search algorithm for system identification." International Journal of Mechatronics, Electrical and Computer Technology 6.21 (2016): 2987-2997.