THE DIGITAL MODELING OF A SIMPLE RC LOW PASS FILTER

Ralph J. Coppola

r_j_coppola@hotmail.com

Introduction:

This presentation is a rewrite of an article that I submitted to TCS, a number of years ago. In it I will demonstrate a simple programming technique that simulates a RC Low Pass Filter that can be used on digital data.

Quite often our measurements or data are masked by short term clutter or noise. If we use an analogue sensor, this clutter can be reduced, fairly easily, by a simple **RC Low Pass Filter**. Unfortunately, this method will not work with digital data.

Smoothing digital data, must be accomplished by using software. The book **Physical Computing**, and this **Smoothing Tutorial** show several examples of algorithms that can be used to clean up measured data. While these examples get the job done, I feel that they take up more lines of code than I am willing to spend.

The Basic RC Low Pass Filter:

Figure 1 illustrates a simulation of the output signal from a noisy analogue sensor.

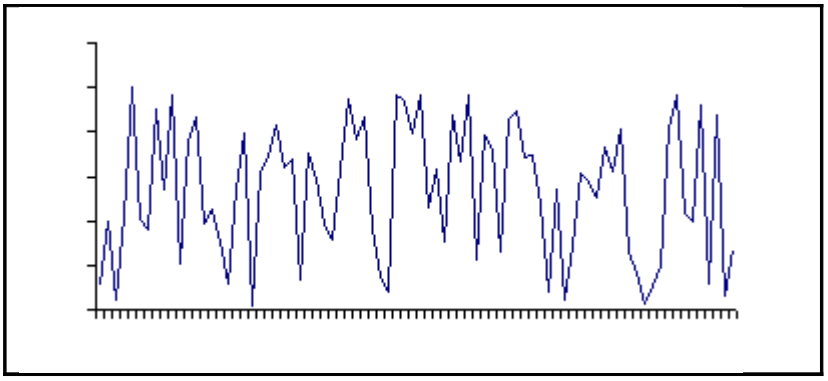


Figure 1
Exaggerated Raw Signal

In order to clearly see the long term or low frequency trends, the short-term, high frequency, noise must be removed. This is especially true if the measurement is

being used to control a process where the process must respond to the over all trend rather than short term spikes. Analogue signals are easy to filter by using a Resistor/Capacitor (RC) Low Pass Filter, such as the one shown in Figure 2.

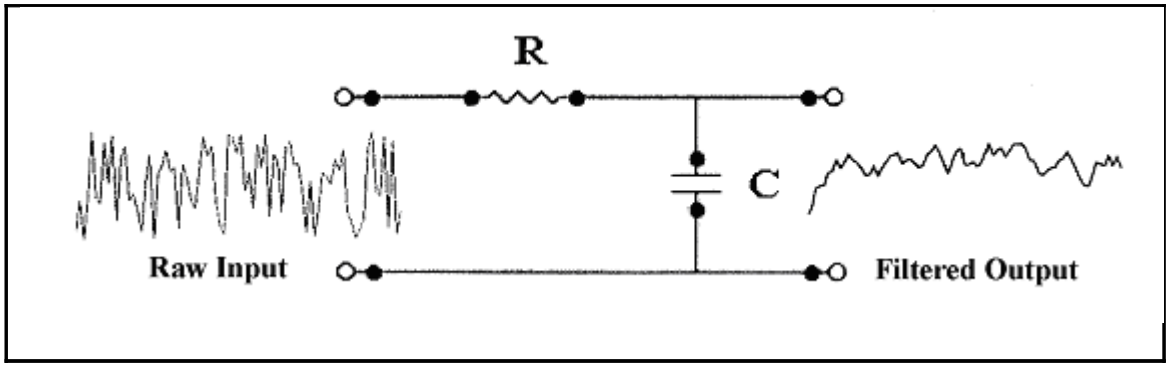


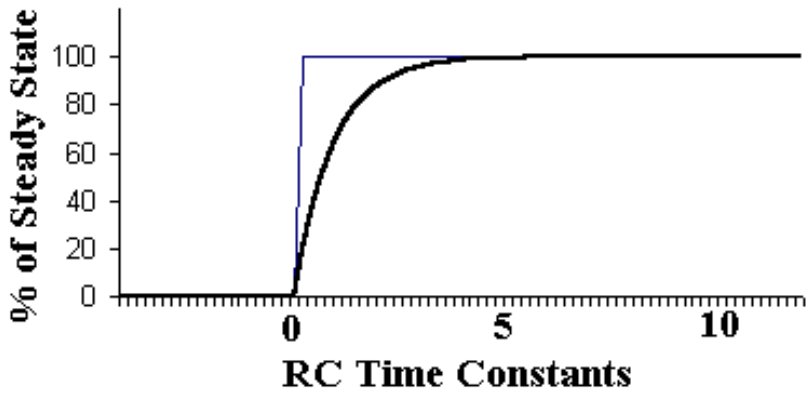
Figure 2
RC Low Pass Filter

It is not my intention to go too deeply into the theory of RC Filters. Suffice to say that the operation of this filter is based on the time that it takes for the capacitor to charge or discharge. Those who may not be familiar with the subject are encouraged to refer to the links on RC Filters and/or RC Time Constants provided at the end of this article.

The effect of the filter is governed by the value of the RC Time Constant (1 Time Constant (seconds) = R (ohms) X C (Farads)). In other words, the longer the Time Constant, the greater the filtering action will be.

As a rule of thumb, the output of the circuit is considered to have reached a steady state level after 5 RC Time Constants. This is true for both the charging and discharging cycles of the filter.

Figure 3 shows a RC Low Pass Filter's DC response to a positive step input.



Time	% of Steady State
0 TC	0
1 TC	63.2%
2 TC	86.5%
3 TC	95.0%
4 TC	98.2%
5 TC	99.3 %

Note --- 1 Time Constant (seconds) = R (ohms) X C (farads)

Figure 3
RC Low Pass Filter's Response to a Positive STEP Input

Filter Modelling:

If the data of interest is in a digital format, such as the output from an analogue/digital converter or a computer text file, some method other than a RC filter must be used.

Figure 4 illustrates a program, using only simple arithmetic, which closely simulates the characteristics of a RC Low Pass Filter. The smoothing is accomplished by taking a fraction of the new input (D) and adding it to a fraction of the previous filtered result (P).

$$S = (f * D) + ((1 – f) * P)$$

Where:

S = Smoothed or filtered value.
f = Filter factor.
0 = Maximum filtering (No output : S = 0)
1 = Minimum filtering (S = D)
D = Raw data input
P = Previous value of S

Figure 4
Simulation Of An RC Filter

Code:

The following code is written in a generic form and should be easily translated into your favourite language.

```
# ***********************************
# *  Example of Filter Program  *
#                                   *
# ***********************************

# S = (f*D) + ((1-f) * P)

S = 0      #Smoothed or filtered value
f = 0.2    # Filter factor (20%)
D = 0      # Raw data input
P = 0      # Previous smoothed value

Start    input D                    # Get the raw (D)ata input sample
         S = (f*D) + ((1-f) * P)    # Compute the new (S)moothed value
         output S                   # Output the (S)moothed value to the application
         P = S                      # Save the (S)moothed value as (P)revious
         goto Start                 # Loop for the next input sample
End
```

Figure 5
Generic Code

Filter Response:

In this example, a Filter Factor of 0.2 is used to simulate an actual RC Low Pass filter very closely.

RC Low Pass Filter		Simulation (f=0.2)	
Time	% of Max	Samples	% of Max
0 RC	0	0	0
1 RC	63.2%	4	59.0%
2 RC	86.5%	8	83.2%
3 RC	95.0%	12	93.1%
4 RC	98.2%	16	97.2%
5 RC	99.3 %	20	98.8%

Figure 6
Comparison Of An RC Low Pass Filter To The Simulation

It's worth noting that, unlike the RC filter, which is related to time, the simulation's output is instead related to the number of samples.

Figure 7 shows the simulation's response to a positive / negative step function, while Figure 8 illustrates its response to a random cyclic input.

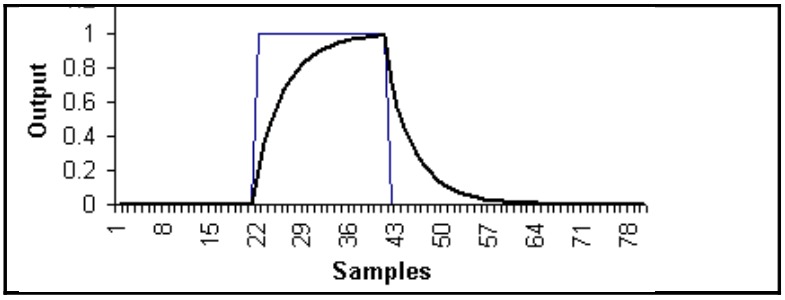


Figure 7
Response to a Positive / Negative Step Function

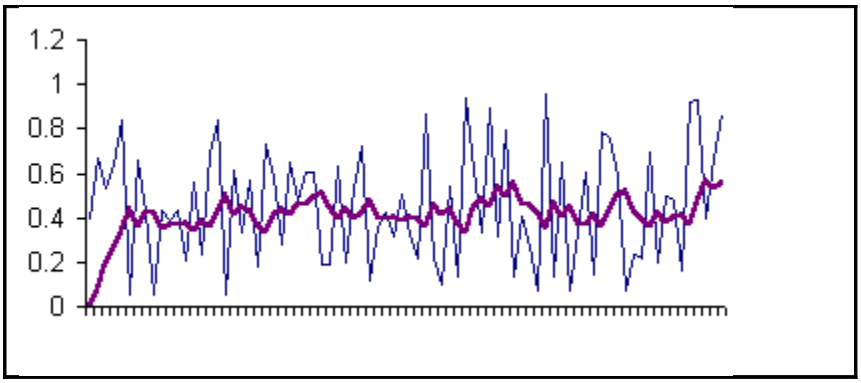


Figure 8
Response to a Random Function

Conclusion:

I have successfully used this filter in systems ranging from 256 byte microcontrollers to full blown PC's.

Besides its simplicity, the main advantage of this model is that it can be used in real time, while the data is being collected, or at a later time on stored data.

Similar techniques appear in many industrial process control applications. It is hoped that you find it of some value in your projects.

Further Reading:

- **Sensors**
- **The Data Acquisition Handbook**
- **An Introduction to Analog-to-Digital Data Collection**

References:

- **Charging a Capacitor**
- **RC Time Constant**
- **RC Time Constant JAVA Demo**
- **RC Low Pass Filter JAVA Demo**
- **RC Low Pass Filter (AC)**

= = = = = = (.) = = = = = =