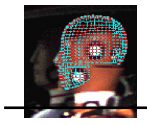




# Kalman Filter Applications

The Kalman filter (see Subject MI37) is a very powerful tool when it comes to controlling noisy systems. The basic idea of a Kalman filter is: Noisy data in $\Rightarrow$ hopefully less noisy data out.

The applications of a Kalman filter are numerous:

- Tracking objects (e.g., missiles, faces, heads, hands)
- Fitting Bezier patches to (noisy, moving, ...) point data
- Economics
- Navigation
- Many computer vision applications
  - Stabilizing depth measurements
  - Feature tracking
  - Cluster tracking
  - Fusing data from radar, laser scanner and stereo-cameras for depth and velocity measurements
  - Many more

This lecture will help you understand some direct applications of the Kalman filter using numerical examples.

The examples that will be outlined are:

1. Simple 1D example, tracking the level in a tank (this pdf)
2. Integrating disparity using known ego-motion (in MI64)

# Predict-Update Equations

First, recall the time discrete Kalman filter equations (see MI37) and the predict-update equations as below:

**Predict**:

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F}_t\hat{\mathbf{x}}_{t-1|t-1} + \mathbf{B}_t\mathbf{u}_t$$
$$\mathbf{P}_{t|t-1} = \mathbf{F}_t\mathbf{P}_{t-1|t-1}\mathbf{F}_t^T + \mathbf{Q}_t$$

**Update**:

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t\left(\mathbf{y}_t - \mathbf{H}_t\hat{\mathbf{x}}_{t|t-1}\right)$$
$$\mathbf{K}_t = \mathbf{P}_{t|t-1}\mathbf{H}_t^T\left(\mathbf{H}_t\mathbf{P}_{t|t-1}\mathbf{H}_t^T + \mathbf{R}_t\right)^{-1}$$
$$\mathbf{P}_{t|t} = (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\mathbf{P}_{t|t-1}$$

where

- $\hat{\mathbf{x}}$ : Estimated state.
- $\mathbf{F}$ : State transition matrix (i.e., transition between states).
- $\mathbf{u}$ : Control variables.
- $\mathbf{B}$ : Control matrix (i.e., mapping control to state variables).
- $\mathbf{P}$ : State variance matrix (i.e., error of estimation).
- $\mathbf{Q}$ : Process variance matrix (i.e., error due to process).
- $\mathbf{y}$ : Measurement variables.
- $\mathbf{H}$ : Measurement matrix (i.e., mapping measurements onto state).
- $\mathbf{K}$ : Kalman gain.
- $\mathbf{R}$ : Measurement variance matrix (i.e., error from measurements).

Subscripts are as follows: $t|t$ current time period, $t-1|t-1$ previous time period, and $t|t-1$ are intermediate steps.

# Model Definition Process

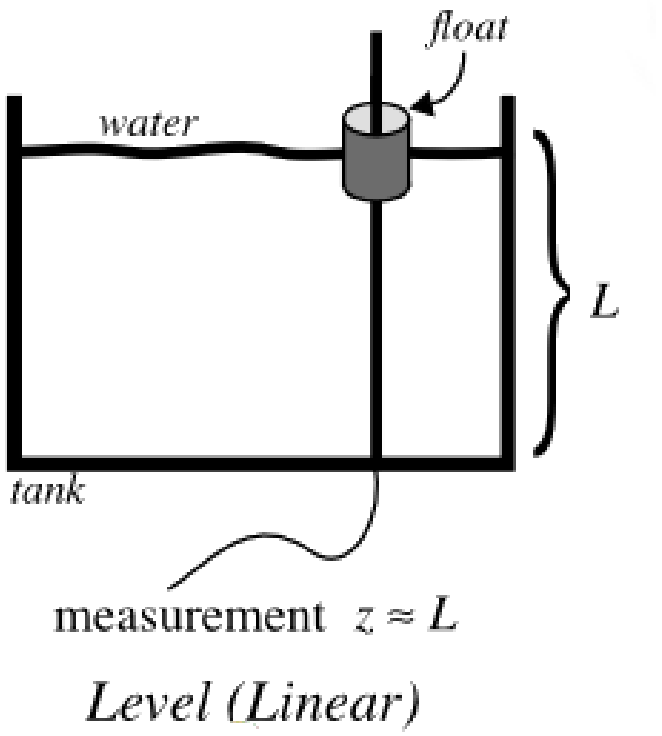
The Kalman filter removes noise by assuming a pre-defined model of a system. Therefore, the Kalman filter model must be meaningful. It should be defined as follows:

1. **Understand the situation:** Look at the problem. Break it down to the mathematical basics. If you don't do this, you may end up doing unneeded work.
2. **Model the state process:** Start with a basic model. It may not work effectively at first, but this can be refined later.
3. **Model the measurement process:** Analyze how you are going to measure the process. The measurement space may not be in the same space as the state (e.g., using an electrical diode to measure weight, an electrical reading does not easily translate to a weight).
4. **Model the noise:** This needs to be done for both the **state** and **measurement** process. The base Kalman filter assumes Gaussian (white) noise, so make the variance and covariance (error) meaningful (i.e., make sure that the error you model is suitable for the situation).
5. **Test the filter:** Often overlooked, use synthetic data if necessary (e.g., if the process is not safe to test on a live environment). See if the filter is behaving as it should.
6. **Refine filter:** Try to change the noise parameters (filter), as this is the easiest to change. If necessary go back further, you may need to rethink the situation.

# Example: Water level in tank

## 1. Understanding the situation

We consider a simple situation showing a way to measure the level of water in a tank. This is shown in the figure[a].




We are trying to estimate the level of water in the tank, which is unknown. The measurements obtained are from the level of the "float". This could be an electronic device, or a simple mechanical device. The water could be:

- Filling, emptying or static (i.e., the average level of the tank is increasing, decreasing or not changing).
- Sloshing or stagnant (i.e., the relative level of the float to the average level of the tank is changing over time, or is static).

[a] Taken from `http://www.cs.unc.edu/~welch/kalman/kftool/index.html`

# First Option: A Static Model

**2. Model the state process**

We will outline several ways to model this simple situation, showing the power of a good Kalman filter model.

The first is the most basic model, the tank is level (i.e., the true level is constant $L = c$).

Using the equations from Page 2, the state variable can be reduced to a scalar (i.e., $\hat{\mathbf{x}} = x$ where $x$ is the estimate of $L$).

We are assuming a constant model, therefore $x_{t+1} = x_t$, so $\mathbf{A} = 0$ and $\mathbf{F}_t = 1$, for any $t \geq 0$.

Control variables $\mathbf{B}$ and $\mathbf{u}$ are not used (i.e., both $= 0$).

**3. Model the measurement process**

In our model, we have the level of the float. This is represented by $\mathbf{y} = y$.

The value we are measuring could be a scaled measurement (e.g., a 1 cm measurement on a mechanical dial could actually be about 10 cm in the "true" level of the tank). Think of your petrol gauge on your car, 1 cm can represent 10 L of petrol!

For simplicity, we will assume that the measurement is the exact same scale as our state estimate $x$ (i.e., $\mathbf{H} = 1$).

**4. Model the noise**

For this model, we are going to assume that there is noise from the measurement (i.e., $\mathbf{R} = r$).

The process is a scalar, therefore $\mathbf{P} = p$. And as the process is not well defined, we will adjust the noise (i.e., $\mathbf{Q} = q$).

We will now demonstrate the effects of changing these noise parameters.

**5. Test the filter**

You can now see that you can simplify the equations from Page 2. They simplify as follows:

**Predict:**

$$x_{t|t-1} = x_{t-1|t-1}$$
$$p_{t|t-1} = p_{t-1|t-1} + q_t$$

**Update:**

$$x_{t|t} = x_{t|t-1} + K_t \left(y_t - x_{t|t-1}\right)$$
$$K_t = p_{t|t-1} \left(p_{t|t-1} + r\right)^{-1}$$
$$p_{t|t} = \left(1 - K_t\right) p_{t|t-1}$$

The filter is now completely defined. Let's put some numbers into this model. For the first test, we assume the true level of the tank is $L = 1$.

We initialize the state with an arbitrary number, with an extremely high variance as it is completely unknown: $x_0 = 0$ and $p_0 = 1000$. If you initialize with a more meaningful variable, you will get faster convergence. The system noise we will choose will be $q = 0{\cdot}0001$, as we think we have an accurate model. Let's start this process.

**Predict:**

$$x_{1|0} = 0$$
$$p_{1|0} = 1000 + 0{\cdot}0001$$

The hypothetical measurement we get is $y_1 = 0{\cdot}9$ (due to noise). We assume a measurement noise of $r = 0{\cdot}1$.

**Update:**

$$K_1 = 1000{\cdot}0001(1000{\cdot}0001 + 0{\cdot}1)^{-1} = 0{\cdot}9999$$
$$x_{1|1} = 0 + 0{\cdot}9999\,(0{\cdot}9 - 0) = 0{\cdot}8999$$
$$p_{1|1} = (1 - 0{\cdot}9999)\,1000{\cdot}0001 = 0{\cdot}1000$$

So you can see that Step 1, the initialization of 0, has been brought close to the true value of the system. Also, the variance (error) has been brought down to a reasonable value.

Let's do one more step:

**Predict:**

$$x_{2|1} = 0{\cdot}8999$$
$$p_{2|1} = 0{\cdot}1000 + 0{\cdot}0001 = 0{\cdot}1001$$

The hypothetical measurement we get this time is $y_2 = 0{\cdot}8$ (due to noise).

**Update:**

$$K_2 = 0{\cdot}1001\,(0{\cdot}1001 + 0{\cdot}1)^{-1} = 0{\cdot}5002$$
$$x_{2|2} = 0{\cdot}8999 + 0{\cdot}5002\,(0{\cdot}8 - 0{\cdot}8999) = 0{\cdot}8499$$
$$p_{2|2} = (1 - 0{\cdot}5002)\,0{\cdot}1001 = 0{\cdot}0500$$

You can notice that the variance is reducing each time. If we continue (with hypothetical $y_t$-values) this we get the following results:

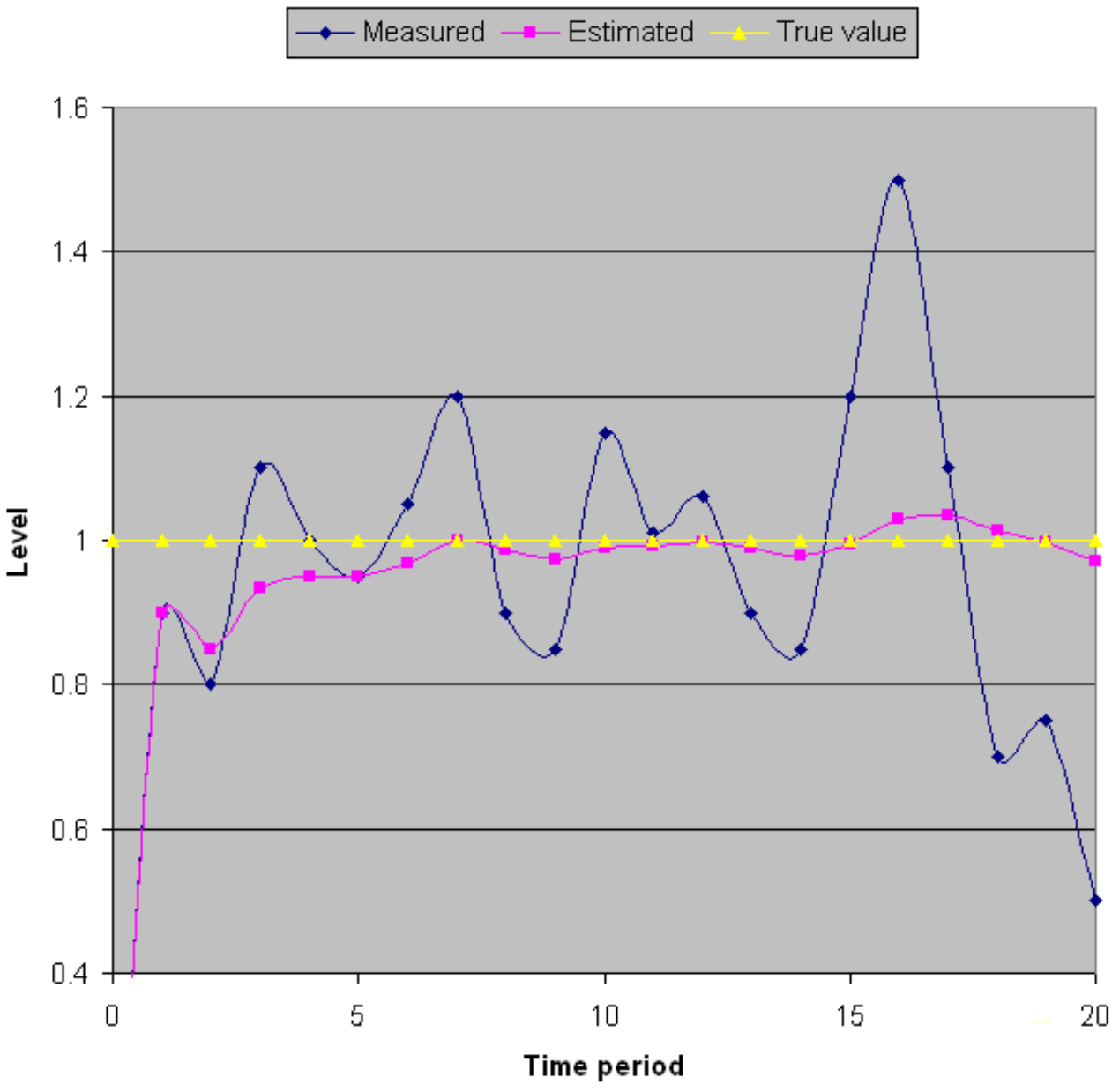
| | Predict | | Update | | | |
|---|---|---|---|---|---|---|
| $t$ | $x_{t\|t-1}$ | $p_{t\|t-1}$ | $y_t$ | $K_t$ | $x_{t\|t}$ | $p_{t\|t}$ |
| 3 | 0·8499 | 0·0501 | 1·1 | 0·3339 | 0·9334 | 0·0334 |
| 4 | 0·9334 | 0·0335 | 1 | 0·2509 | 0·9501 | 0·0251 |
| 5 | 0·9501 | 0·0252 | 0·95 | 0·2012 | 0·9501 | 0·0201 |
| 6 | 0·9501 | 0·0202 | 1·05 | 0·1682 | 0·9669 | 0·0168 |
| 7 | 0·9669 | 0·0169 | 1·2 | 0·1447 | 1·0006 | 0·0145 |
| 8 | 1·0006 | 0·0146 | 0·9 | 0·1272 | 0·9878 | 0·0127 |
| 9 | 0·9878 | 0·0128 | 0·85 | 0·1136 | 0·9722 | 0·0114 |
| 10 | 0·9722 | 0·0115 | 1·15 | 0·1028 | 0·9905 | 0·0103 |

Another reading sequence; try this:

| | Predict | | Update | | | |
|---|---|---|---|---|---|---|
| $t$ | $x_{t\|t-1}$ | $p_{t\|t-1}$ | $y_t$ | $K_t$ | $x_{t\|t}$ | $p_{t\|t}$ |
| 3 | | | 1·05 | | | |
| 4 | | | 0·95 | | | |
| 5 | | | 1·2 | | | |
| 6 | | | 1·1 | | | |
| 7 | | | 0·23 | | | |
| 8 | | | 0·85 | | | |
| 9 | | | 1·25 | | | |
| 10 | | | 0·76 | | | |

You can see (Page 8) that the model successfully works. After stabilization (about $t = 4$) the estimated state is within $0{\cdot}05$ of the "true" value, even though the measurements are between $0{\cdot}8$ and $1{\cdot}2$ (i.e., within $0{\cdot}2$ of the true value).

Over time we will get the following graph:



So what have we established so far?

If we create a model based on the true situation, our estimated state will be close to the true value, even when the measurements are very noisy (i.e., a 20% error, only produced a 5% inaccuracy).

This is the main purpose of the Kalman filter.

But what happens if the **true** situation is different?

We will keep the same model (i.e., a static model). This time, the **true** situation is that the tank is **filling** at a constant rate:

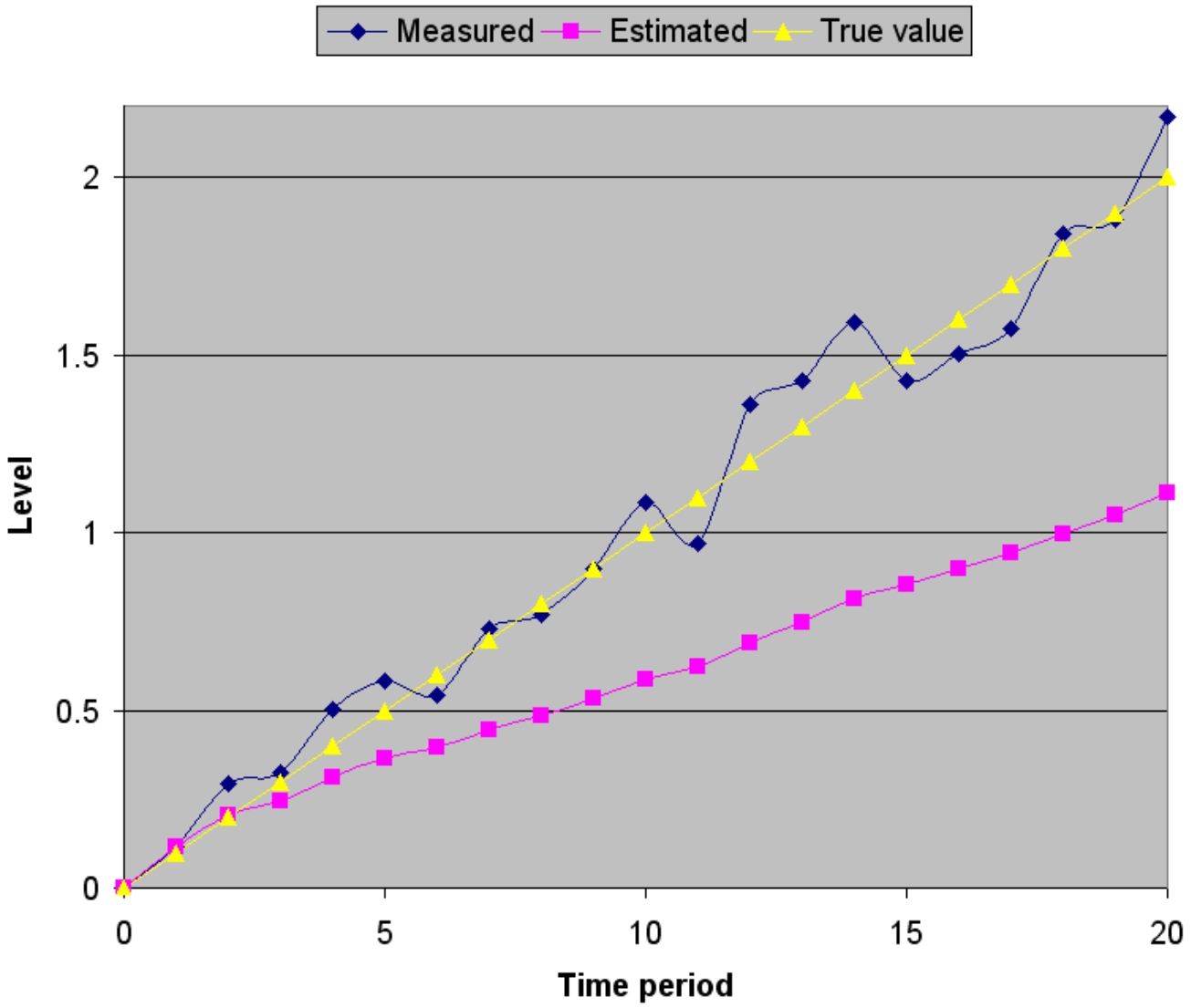
$L_t = L_{t-1} + f$

Let's assume that the tank is filling at a rate of $f = 0{\cdot}1$ per time frame, and we start with an initialization of $L_0 = 0$. We will assume that the measurement and process noise remain the same (i.e., $q = 0{\cdot}001$ and $r = 0{\cdot}1$).

Let's see what happens here:

| | Predict | | Measurement and Update | | | | Truth |
|---|---|---|---|---|---|---|---|
| $t$ | $x_{t\|t-1}$ | $p_{t\|t-1}$ | $y_t$ | $K_t$ | $x_{t\|t}$ | $p_{t\|t}$ | $L$ |
| 0 | – | – | – | – | 0 | 1000 | 0 |
| 1 | 0·0000 | 1000·0001 | 0·11 | 0·9999 | 0·1175 | 0·1000 | 0·1 |
| 2 | 0·1175 | 0·1001 | 0·29 | 0·5002 | 0·2048 | 0·0500 | 0·2 |
| 3 | 0·2048 | 0·0501 | 0·32 | 0·3339 | 0·2452 | 0·0334 | 0·3 |
| 4 | 0·2452 | 0·0335 | 0·50 | 0·2509 | 0·3096 | 0·0251 | 0·4 |
| 5 | 0·3096 | 0·0252 | 0·58 | 0·2012 | 0·3642 | 0·0201 | 0·5 |
| 6 | 0·3642 | 0·0202 | 0·54 | 0·1682 | 0·3945 | 0·0168 | 0·6 |

We can now see that over time the estimated state stabilizes (i.e., the variance gets very low). We can easily see this on the following graph:




Can you see the problem?

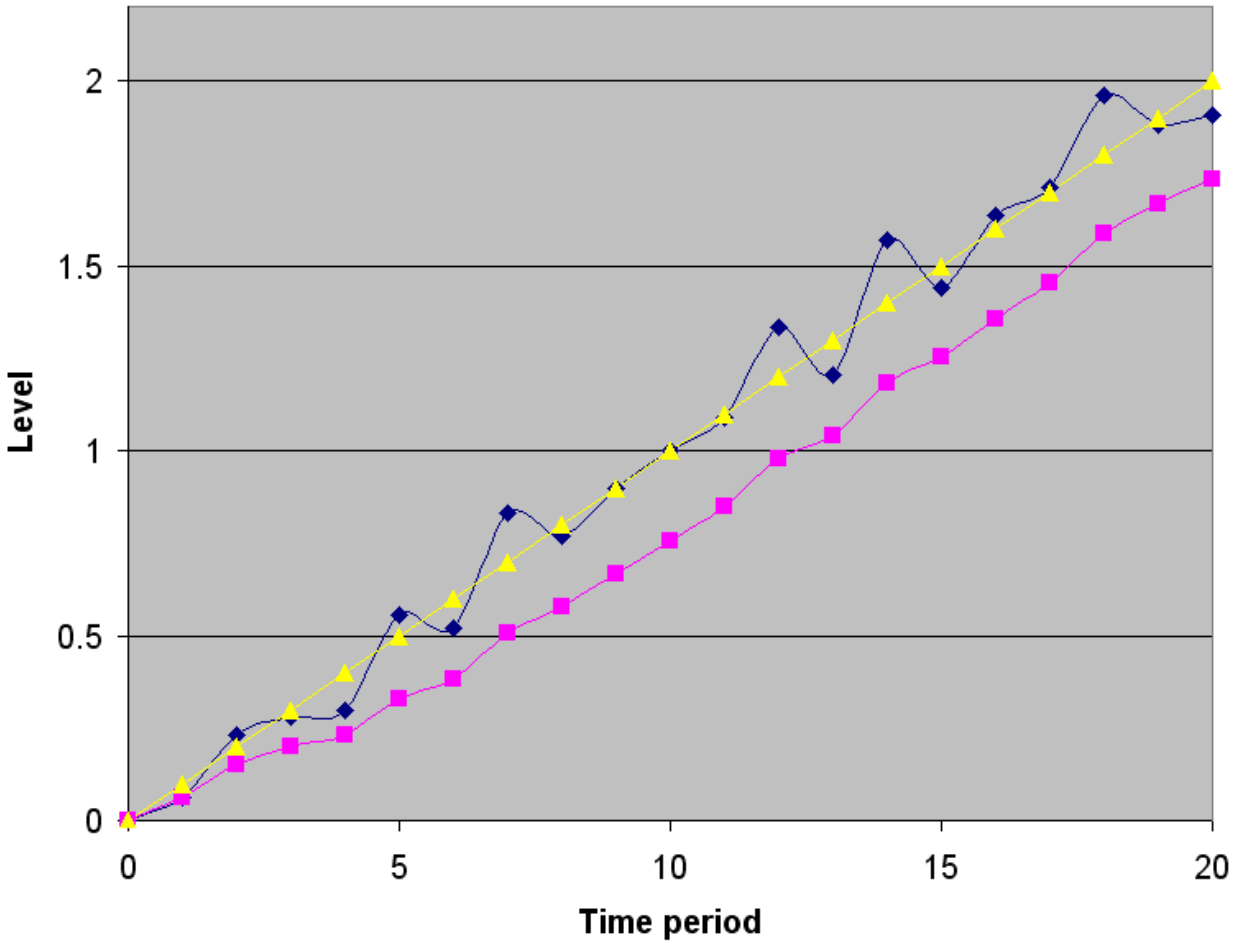
The estimated state is trailing behind the true level. This of course is not desired, as a filter is supposed to remove noise, not give an inaccurate reading. In this case the estimated state has a much greater error (compared to the truth) than the noise from the measurement process.

What is causing this? There are two contributions:

- The model we have chosen.
- The reliability of our process model (i.e., our chosen $q$ value).

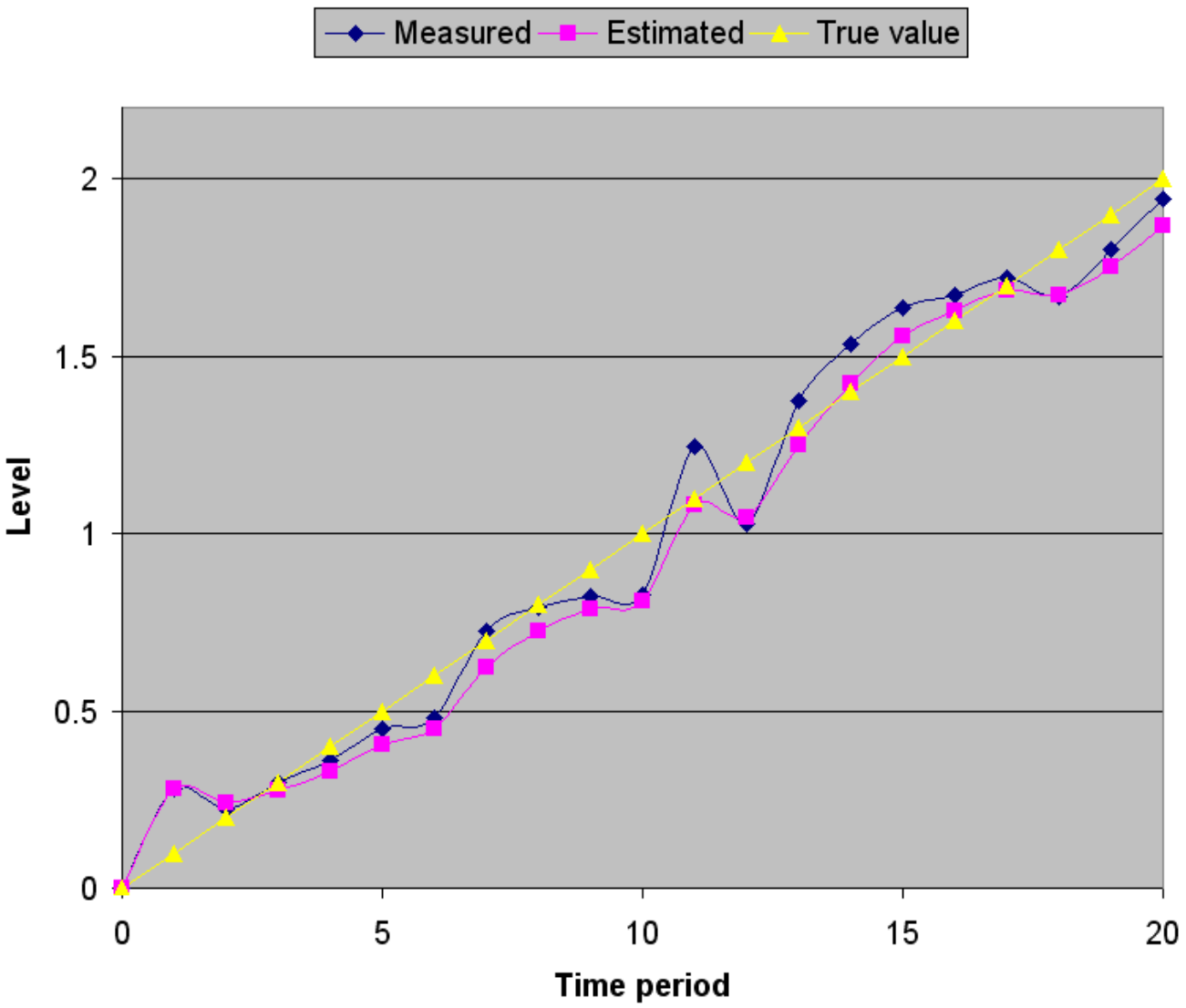
The easiest thing to change is our $q$ value. What was the reason we chose $q = 0{\cdot}0001$? It was because we thought our model was a good estimation of the true process. However, in this case our model was not a good estimator.

So why not relax this? Let's assume there is a greater error with our process model, and set $q = 0{\cdot}01$:



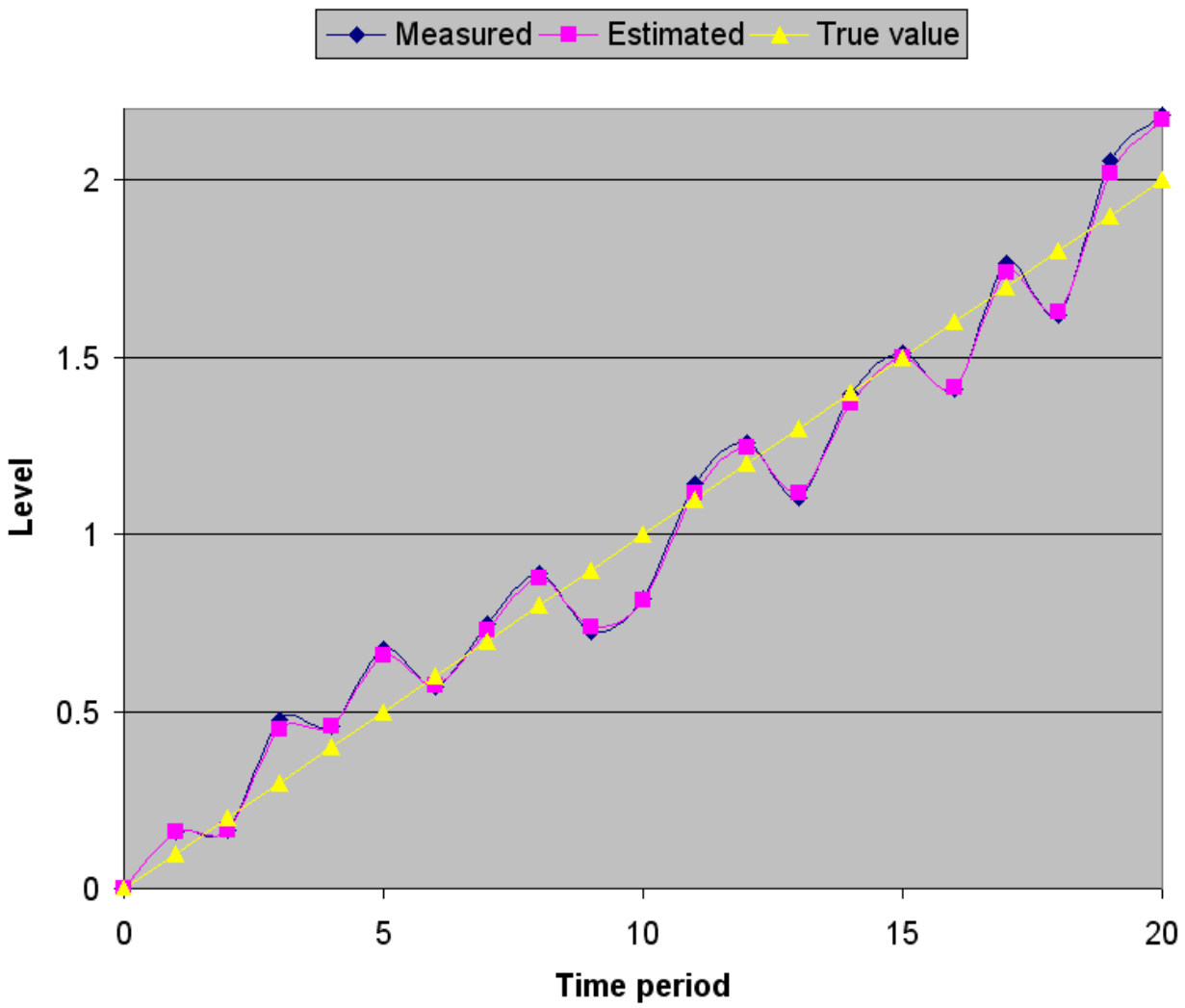
So the benefits of increasing the error were beneficial. However the estimated state still has more error than the actual noise.

Let's increase this value again! Let $q = 0{\cdot}1$:




This is getting close to the true value, and has less error than the measurement noise.

Let's try $q = 1$ to see what happens:




Now there is almost no difference to the measured value. There is minor noise removal, but not much. There's almost no point in having the filter.

So what has been learnt here?

If you have a badly defined model, you will not get a good estimate. But you can *relax* your model by increasing your estimated error. This will let the Kalman filter rely more on the measurement values, but still allow some noise removal.

In your own time, try playing with the measurement error $r$ and see what the effect is.

# Second Option: A Filling Model

**2. Model the state process**

To get better results, we need to change our Kalman model.

Let's redefine it then.

So, the actual model is: $L_t = L_{t-1} + f$

This translates to a *continuous* process transition of:

$$\mathbf{x} = (x_l, x_f)^\top$$

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$

where $x_l$ is the level $L$, and $x_f = \frac{dx_l}{dt}$ is the estimated filling rate, and $\mathbf{A}$ represents the tank continuously filling at rate $x_f$ (defined by $f$ and the used time scale).

However, we want a *discrete* process. So $\mathbf{A}$ needs to be made time-discrete. Just apply the infinite sum as given in MI37 (note that $\mathbf{A}^i$ is zero in all four components, for all $i > 1$):

$$\mathbf{F}_t = \begin{pmatrix} 1 & \Delta t \\ 0 & 1 \end{pmatrix}$$

for all $t \geq 0$. We will be ignoring $\mathbf{B}$ and $\mathbf{u}$ again.

**3. Model the measurement process**

In this case, we can not measure the filling rate directly, but we will still assume a noisy measurement of $L$. Therefore we have:

$$\mathbf{H} = (1\ ,\ 0)$$
$$\mathbf{y} = (y\ ,\ 0)^\top$$

Indicating that there is no measurement for filling rate and that $y$ is the estimated measurement of $x_l$.

**4. Model the noise**

The measurement process has not changed, so that means that the noise has not changed (i.e., $\mathbf{R} = r$).

The process has changed, so we need to redefine the noise.

Now if we assume the noise is only in the filling part of the process, this would give us the *continuous* noise model with

$$\mathbf{Q} = \begin{pmatrix} 0 & 0 \\ 0 & q_f \end{pmatrix}$$

where $q_f$ is the filling noise.

Now, this continuous process can be approximated by a time-discrete process using:

$$\mathbf{Q}(\Delta t) = \int_0^{\Delta t} e^{\mathbf{A}\tau}\mathbf{Q}e^{\mathbf{A}^\top\tau}d\tau$$

Using this continuous to discrete translation, we can take the *continuous* $\mathbf{A}$ and $\mathbf{Q}$ to approximate the *discrete* $\mathbf{Q}$. This gives us (after some calculations - to be skipped here; again, apply the infinite sum for the power of $e$):

$$\mathbf{Q} = \begin{pmatrix} \frac{\Delta t^3 q_f}{3} & \frac{\Delta t^2 q_f}{2} \\ \frac{\Delta t^2 q_f}{2} & \Delta t q_f \end{pmatrix}$$

For simplicity, we assume a continuous sampling rate of $\Delta t = 1$.

Therefore,

$$\mathbf{F}_t = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

$$\mathbf{Q} = \begin{pmatrix} q_f/3 & q_f/2 \\ q_f/2 & q_f \end{pmatrix}$$

Also, the process is no longer scalar, so the covariance matrix is

$$\mathbf{P} = \begin{pmatrix} p_l & p_{lf} \\ p_{lf} & p_f \end{pmatrix}$$

where the subscript denotes the relating variance and $p_{lf}$ is the covariance (which is symmetric, i.e., $p_{lf} = p_{fl}$).

**5. Test the filter**

The equations from Page 2 could be slightly simplified, but for our purposes, it is best just to put in the information from above into the equations. ($\mathbf{K}$ can be simplified quite nicely.)

**Predict**:

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F}_t \hat{\mathbf{x}}_{t-1|t-1}$$
$$\mathbf{P}_{t|t-1} = \mathbf{F}_t \mathbf{P}_{t-1|t-1} \mathbf{F}_t^T + \mathbf{Q}_t$$

**Update**:

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t \left( \mathbf{y}_t - \mathbf{H}_t \hat{\mathbf{x}}_{t|t-1} \right)$$
$$\mathbf{K}_t = \mathbf{P}_{t|t-1} \mathbf{H}_t^T \left( \mathbf{H}_t \mathbf{P}_{t|t-1} \mathbf{H}_t^T + \mathbf{R}_t \right)^{-1}$$
$$\mathbf{P}_{t|t} = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \mathbf{P}_{t|t-1}$$

This time we have an accurate model of a constant fill rate. We will assume that we have a noise of $r = 0{\cdot}1$, and an accurate process noise of $q_f = 0{\cdot}00001$.

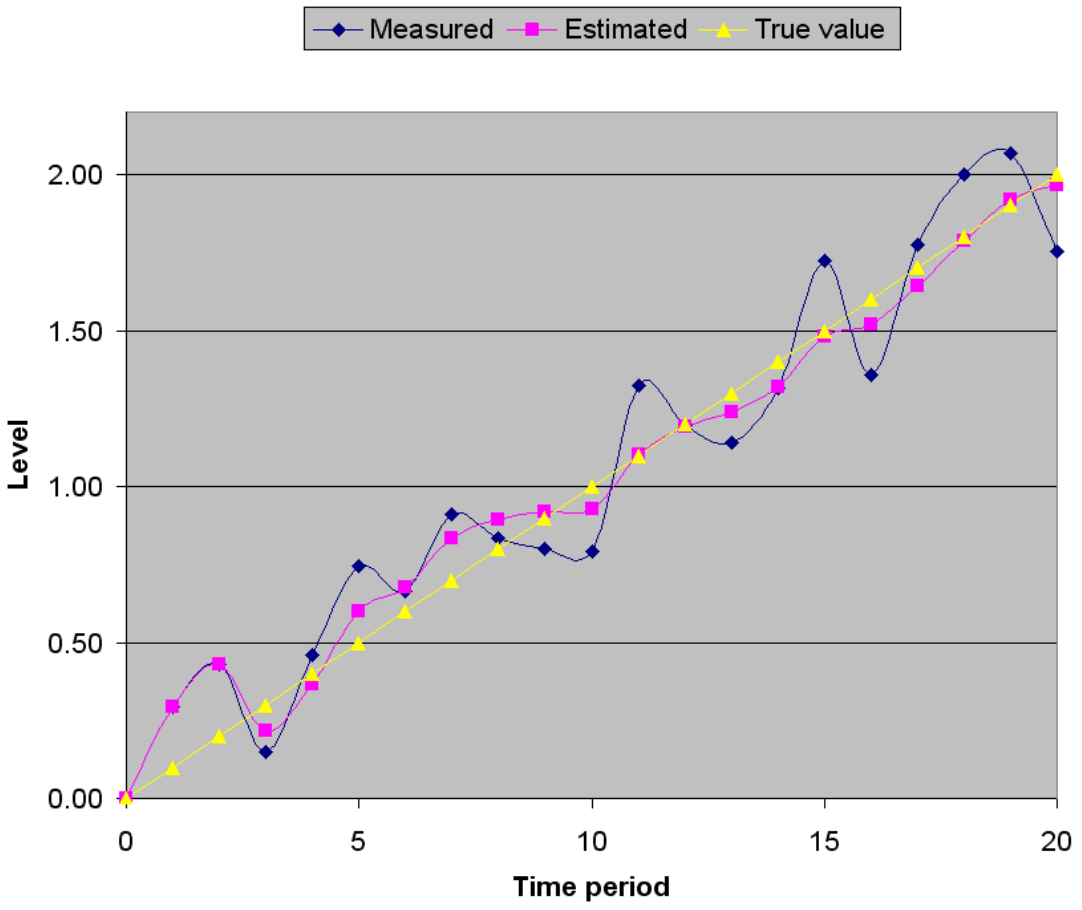
We have no idea of the initial state or filling rate, so we will make $\mathbf{x}_0 = (0\ ,\ 0)^\top$ with an initial variance of:

$$\mathbf{P}_{0|0} = \begin{pmatrix} 1000 & 0 \\ 0 & 1000 \end{pmatrix}$$

We are assuming we have no idea of either values, and we are assuming that there is little or no correlation between the values.

We will need to test these results under different conditions. The first test will be that the model is filling at a constant rate of $0{\cdot}1$ per time frame, with an actual measurement noise of $\pm 0{\cdot}3$.

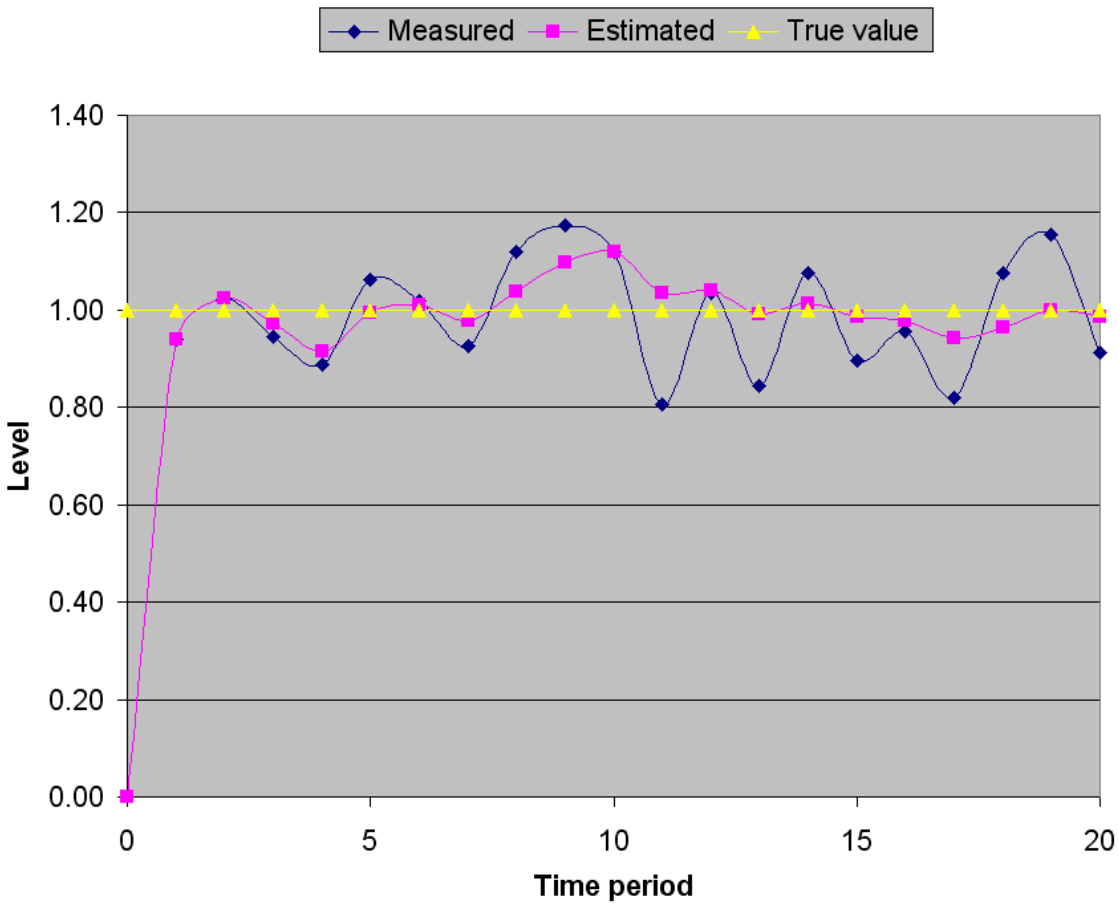
If we plot this on a graph it looks as follows:




Notice that the filter quickly adapts to the true value. Note that we did not tell the Kalman filter anything about the actual filling rate, and it figured it out all by itself. Even with a bad unsure initialization.

Actually, if you give a Kalman filter a bad initialization, it takes the first measurement as a "good" initialization.

Let's see what happens if you try to trick the Kalman filter by using a constant level (i.e., there is no actual filling):




The filter stabilizes in the exact same time frame as before. This is because the model stabilizes with a fill rate of 0.

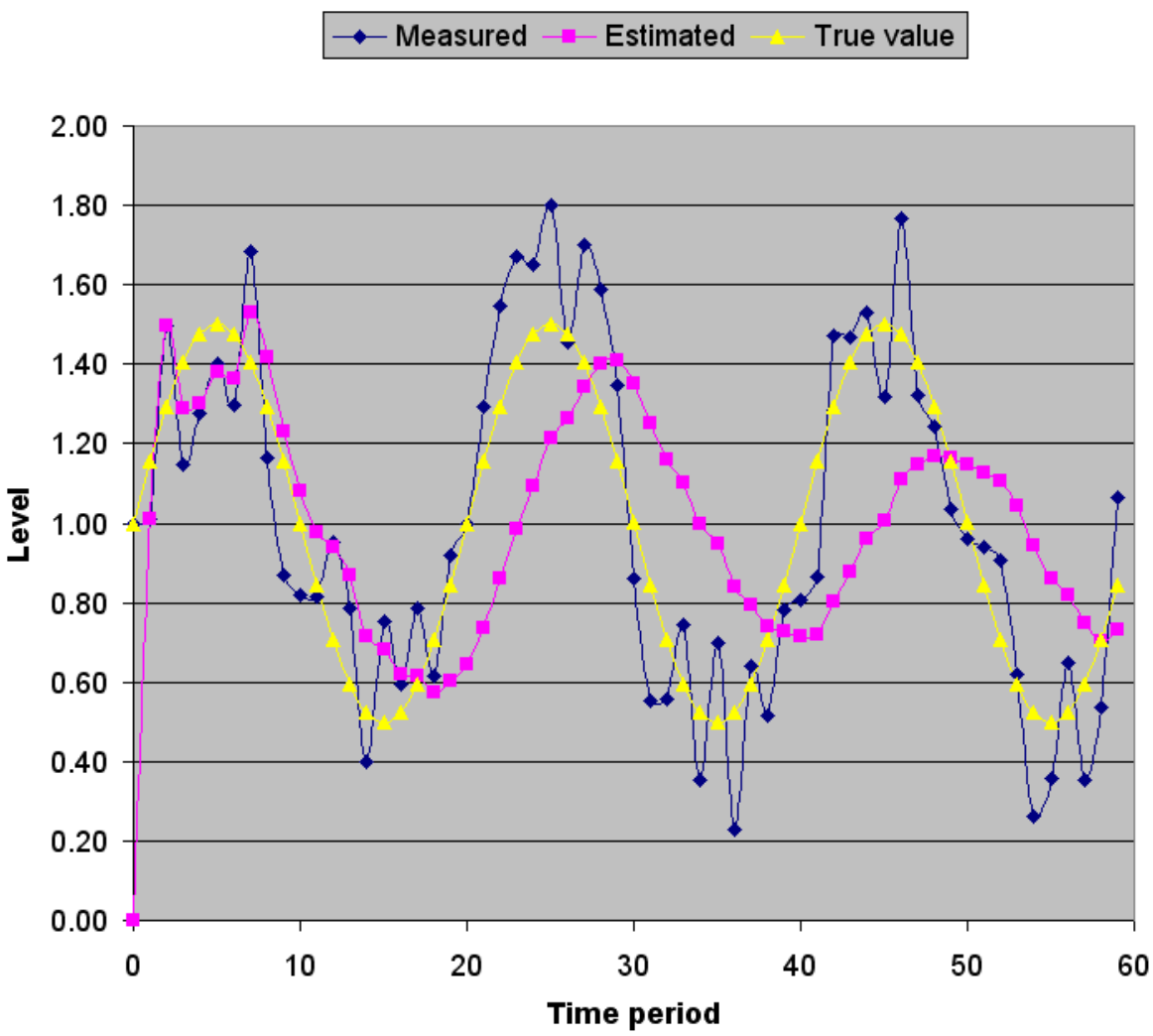
# Third Option: Constant, but Sloshing

Now let us test an extreme case. We will assume that the water is at a constant level, but it is sloshing in the tank. This sloshing can be modeled as a sine wave

$$L = c\ \sin(2\pi\ r\ \Delta t) + l$$

where $c$ is an amplitude scaling factor, $r$ is the cycle rate, and $l$ is the average level (as $\sin$ integrates to zero over time).

If we use $c = 0{\cdot}5$, $r = 0{\cdot}05$, and $l = 1$ we get the following result:



There are two things you should notice about the Kalman filter:

1. The model is smoother than the noisy measurement, but there is a lag behind the actual value. This is common when filtering a system that is not modelled correctly.

2. The amplitude of the filter is getting smaller and smaller. This is because the model is slowly converging to what it thinks is the truth... a constant level, which is accurate over time.

If you wanted to model the sloshing properly, you would need to use an extended Kalman Filter, which also takes into consideration the non-linearity of the system.

See

`www.cs.unc.edu/~welch/kalman/media/pdf/kftool_models.pdf`

for details.

# Summary of the Three Tank Examples

The main points that you should learn from these examples are:

1. The model for your filter will try to fit the given measurement data to the model you have provided. This may be undesirable. In the sloshing case, it may be desirable to filter out the sloshing as "noise", so you can get the average level of the tank. If you wanted to measure the sloshing, then you need to model it.
2. The initialization and noise components of your filter do effect the results of how the filter operates.
3. You need to think about the outcome of your filter; if you can approximate that the time steps are small enough, then a simple linear model will work, but you will get lag.

# Coursework

**63.1.** Download the *KalmanExample.xls* spreadsheet from the course's website.

Here there are two tabs, one for Options 1 and 2 (constant or with filling), and one for Option 3 (constant and sloshing). A tab says the name of the model, followed by the name of the actual situation.

For each model, try changing the different noise parameters $r$ and $q$ to see the effect. Keep $q$ static and change $r$, then vice versa.

Try changing the initialization values $x_{0|0}$ and $p_{0|0}$ to see the effect on the overall outcome.

**63.2.** Go to the UNC (University of North Carolina) website with a dedicated Kalman filter learning tool:

`www.cs.unc.edu/~welch/kalman/kftool/index.html`

Read the document describing the filter:

`www.cs.unc.edu/~welch/kalman/media/pdf/kftool_models.pdf`

Then play with the online Java tool. Once you have an understanding you can download the Matlab or Java code to see how it operates.