Observation Theory

Script V4PC – Non-linear least squares Solution and properties

Having seen examples of non-linear observation equations and knowing the principle of non-linear least squares, what remains to be shown is how to solve a system of m non-linear observation equations in n unknowns, and to present the properties of the non-linear least squares estimator.

So we are still looking at the problem where we the m observations have a non-linear relation with the unknown parameters in x.

In the previous lecture, the principle of non-linear estimation was introduced based on the scalar case

This means we have one observable and one unknown.

Let's review the principle, which is based on an iteration procedure.

In each iteration we use the estimate or approximation of x from the previous iteration.

Then we solve the forward model to see what the corresponding y-value would be with this approximation, and we calculate the difference with the actual observation giving us Delta-y.

Then Delta-x can be computed based on the linear approximation of the function a(), and finally a new approximation for x can be obtained.

This new approximation is used in the next iteration which continues until Delta-x becomes very small.

This iteration procedure is known as the Gauss-Newton iteration.

Now let's zoom in on the equation highlighted here.

In the scalar case it is straightforward to calculate Delta-x, but what if x is a vector?

In the previous video lecture it was already shown that then we would get the following result.

It is very similar, except that we need to use the gradient vector with the n partial derivatives of function a().

Let's now extend this to the general situation where we have m observations and n unknowns

So here we have m different non-linear functions of the n-vector x.

For each function we can find the corresponding gradient vector as on the previous slide.

For example for the first observation the expression for Delta-y becomes now the one shown here, and since we are looking only at the first observation, the partial derivatives of function a1 are to be considered .

We can do this for all observations, such that we can set up a system of m equations

This gives us the so-called linearized functional model.

The matrix on the right-hand side of the equation may look impressive.

But if you look closely you can see that for instance the first row contains the n partial derivatives of function a1 , the second row the partial derivatives of function a2 , etcetera.

The matrix is the Jacobian matrix, for which we use the notation J.

The system of equations in short-hand notation becomes this.

note that the Delta-y and the Jacobian will be different in each iteration, since they depend on the estimate of x from the previous iteration.

Now that we have a linear model again, we are able to estimate the vector Delta-x

To see this, compare the linearized model with our default linear model shown on the right.

And see the similarity: matrix A is replaced by the Jacobian at iteration i, and now we are estimating the Delta-x parameters from Delta-y instead x from y.

The dispersion is the same in both cases.

For the default linear model the best linear unbiased estimator was shown to be the optimal estimator of x.

Replacing the A-matrix with J, and x and y by the Delta-equivalents, the corresponding estimator of Delta-x thus becomes equal to the equation shown here.

Since we are now estimating the Delta-x using BLUE, we also use the notation with the hat.

Having estimated the Delta-x allows to calculate a new estimate of the unknown parameters x for this iteration.

Recall that in the scalar case the stop criterion for the iteration was simply when the Delta-x was very small.

Since x is now a vector, this should be changed to for example

The requirement that the length or norm of the Delta-x vector is very small.

However, since we may have different types of quantities in x, it may be even better to use different weights for the components in x and as such look at the weighted squared norm

With this, we have all the tools for non-linear least squares estimation.

But there are still a few remarks to be made.

Firstly, related to the initialization

Unfortunately, there is not a default way to get an initial guess for x at the start of the iteration procedure.

How to make a good guess depends on the problem at hand.

For example, if x is the position of a point, you may use a map or sketch to get an initial guess.

At the same time, a bad initial guess may have as a result that the solution does not converge.

In other words, the Delta-x does not become smaller and smaller while iterating.

Conversely: if you have a good initial guess, the solution may converge in only a few iterations.

In summary we can say that proper initialization is important and requires insight into the estimation problem at hand.

This brings me to the final part of this lecture on non-linear least squares, where we

Will look at some important properties of the non-linear least squares estimator.

Remember that the estimation is based on a linear approximation of the non-linear functions in our model.

This unfortunately implies that the non-linear least squares estimator is not a best linear unbiased estimator, even though we do apply BLUE to the linearized model.

This also means that the estimator does not have the normal distribution.

In practice though, these important properties can still be assumed to be valid

If the higher order terms in the Taylor approximation are close to 0.

In other words, when the linear approximation is a very good approximation in the vicinity of x.

This turns out to be true in many estimation problems.

With that let me conclude this lecture series on non-linear least squares estimation, from which you should have learned how to set up the non-linear functional model, how to apply the iteration procedure to estimate the unknown parameters using a Taylor series approximation, and finally to be aware of the properties of the non-linear least squares estimator.