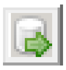
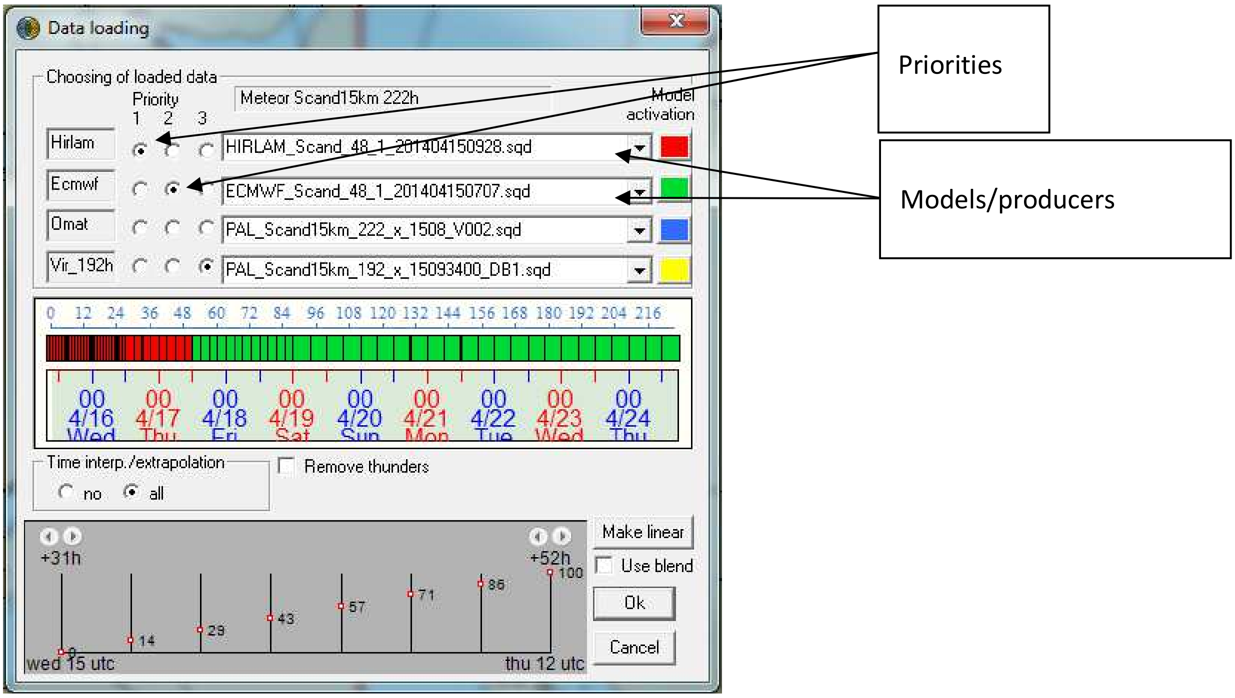
# SmartMet – quick guide to Editing

## 2015

# 1. **Loading data** 




## 1.1 Editable timesteps

The length of editable data varies from country to another depeding on the needs in each location. And so does the editable timesteps (i.e. the timesteps in the data that are edited). Quite common setup is that the nearest few to maybe **24 hours are edited with one hour interval** and **after that the data is edited with three hour interval** or even six. The easiest way of differentiating the editable and uneditable timesteps when doing the editing is that there isn't a couloured outer border in editable timesteps and there is an yellow outer border in uneditable timesteps in the main map view of SmartMet. SmartMet will automatically interpolate the data between editable timesteps when changes are made.

## 1.2 Official data and work data

The last two producers in the data loading dialog is wok data and official data. Official data is the latest data that is sent to the server. Work data is the latest work data that is saved to computers local hard drive. If SmartMet crashes, you are able to get the data that you were editing by selecting the right work data file (not always the latest in the menu).

## 1.3 Saving data and sending data to database 

Saving the data is made from the **save data button**. This means that the data you are editing is saved to the local hard drive on the computer but it isn't yet send to database. Sending to the server (database) is made from the **To database button**.

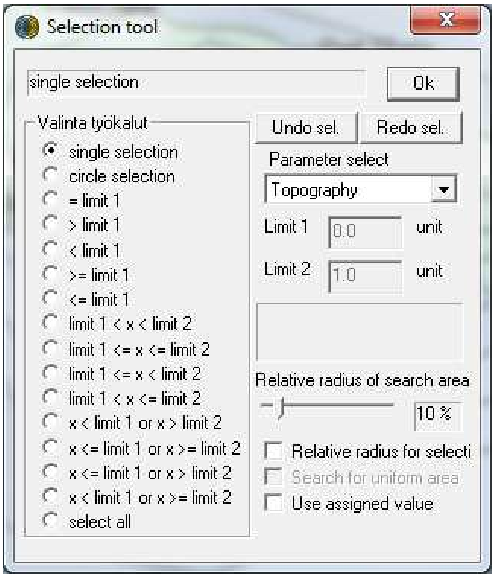
## 1.4 Undo/redo

With the help of the **Undo/redo** function you can discard unwanted changes made to the data (and redo them, if you undo them by mistake). Each tool that modifies data stores an image of each status and this image can be restored, if needed.

# 2. **Tools**

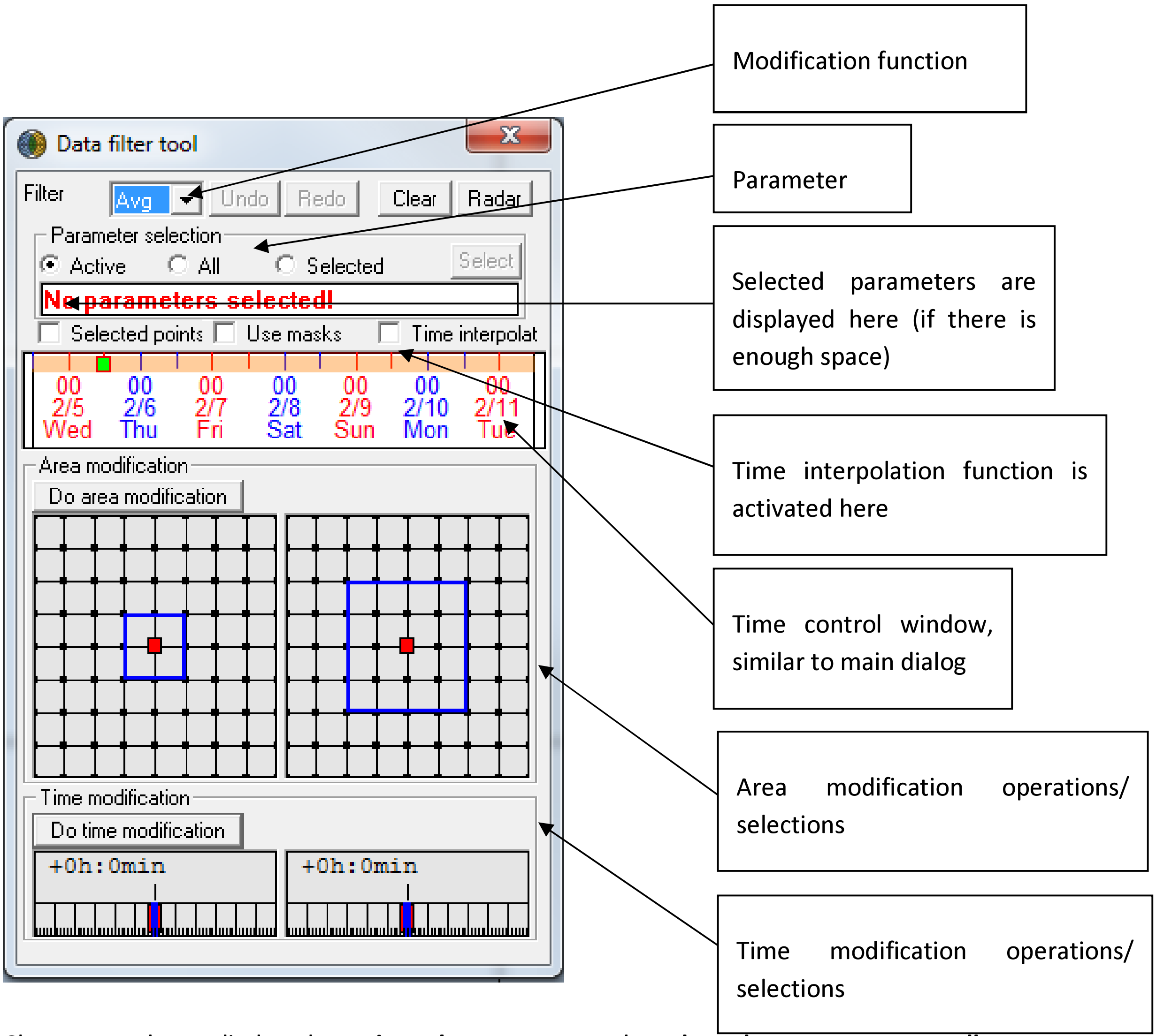
## 2.1 Selection tool 

You can choose grid points from the editable data. By **CTRL**-left click you can make new selections. By **SHIFT**-left click you can remove points from the selection. You can select grid points one at a time (default mode) or within a set circle. The radius of the circle is set at the **Relative radius of search area** slider.




## 2.2 Data filter tool 

Data filter tool can be used for evening data in relation to time or location, for moving data or for both moving and evening data between locations or times.



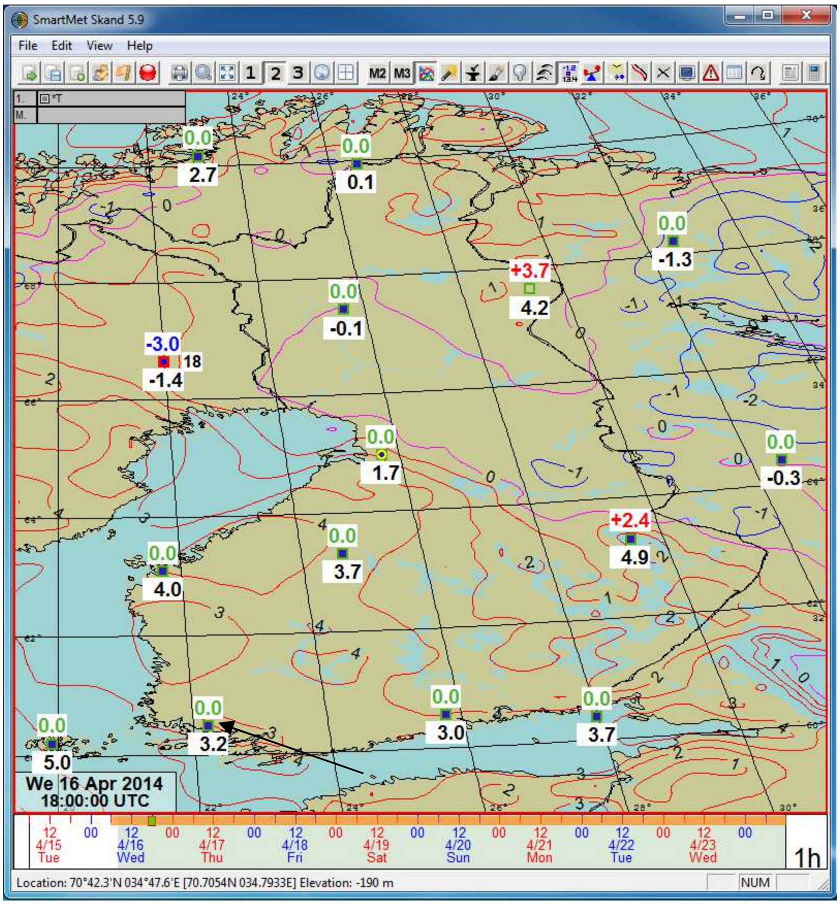

Changes can be applied to the **activated parameter**, to the **selected parameters** or to **all parameters**. If you choose to apply the change to the activated parameter, you should check in the parameter row (**Error! Reference source not found.**) which parameter is activated (which parameter is marked with * in the parameter display).

**NOTE! If you modify data and the result is not what you expected (if, for example, nothing seems to have changed), you should undo the change (with the undo button).**

**Selected points** (**Error! Reference source not found.**) means that the modification is done only to activated grid points. If you don't have a marker here, the modification is done automatically to all the grid points.

## 2.3 Control point tool 

Control point editing means that a few points are edited as a time series and the changes spread from these points to the surrounding locations. When you are in the CP (**CP = Control Point**) editing mode, CP's are displayed as boxes and numbers on the map.



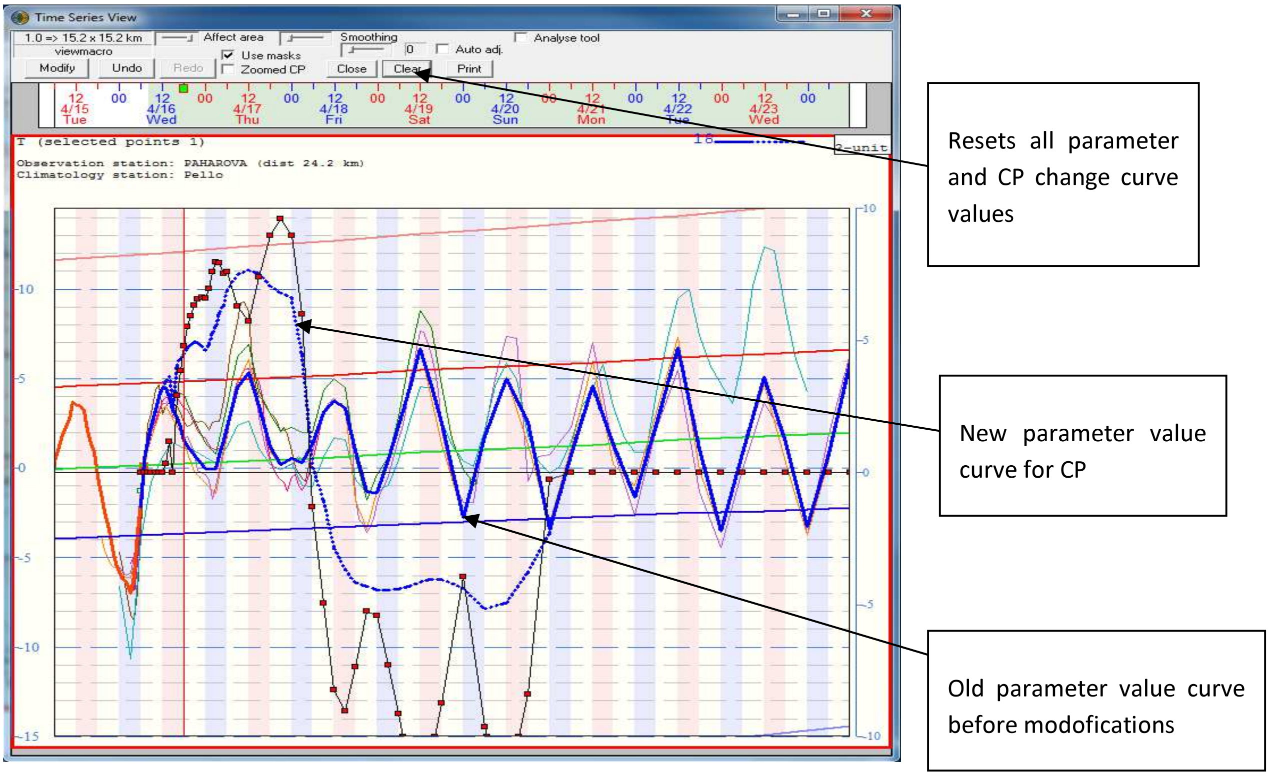
A CP can be moved with the mouse. Activate the CP by clicking it. Push the left mouse button, keep it pressed down and drag the CP where you want it.

You can add as many CPs as you want on the map. Keep **CTRL+SHIFT** pressed down and click the map with the left mouse button. You can delete a CP by activating it and pressing the **DEL** key.

### 2.3.1 Changing and editing values using a control points

Click the point you want to edit on the map. You can view the values of the grid point in the time series view. You can make parameter-specific change curves for each CP. For example, temperature and total cloudiness have their own curves. You are able to modify the existing curve in the time series view for the parameter by keeping left mouse button pressed down and drawing the curve. There is also an observation curve for the parameter from the nearest observation station (thick red line) and values of different models (thin coloured lines) in the time series view. Make the necessary changes for the parameter in all the CP's and finally press the **Modify** button. Note that the amount of the changes you made will only be reset when you press the **Clear** button. This means that once you have made the changes you wanted and pressed modify and you are satisfied with the results ***you need to***

***press the clear button***. Masks can be used in CP editing just like in ordinary time series editing, see more from the full manual.



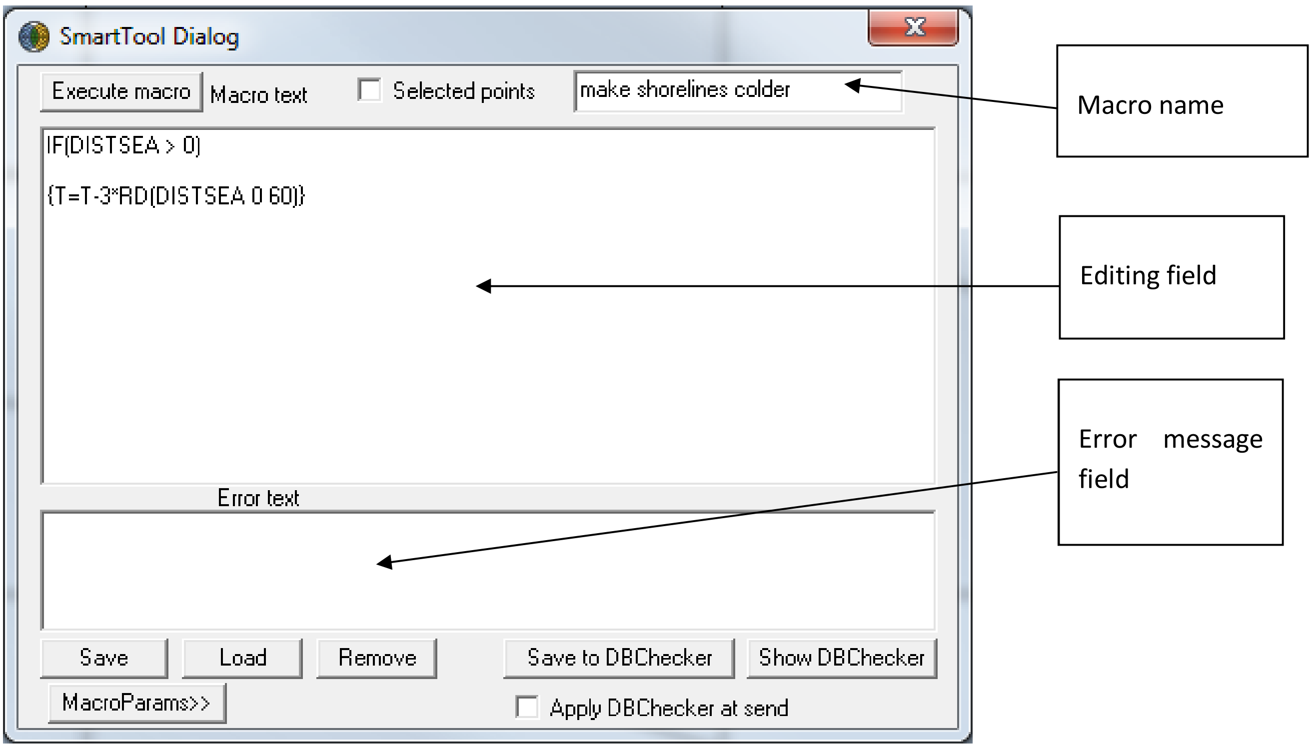
# 3. SmartTool macros

## 3.1 Principle

The purpose of the tool is to modify meteorological data with different formulas and rules. The editor uses a “programming” language designed for this purpose. The language is text-based, case insensitive, and it contains different statements and conditions. In addition to different modifications, the language can be used for sanity checks (which of course also modify the data). For example, the condition and possible modification “if there are no clouds, there can be no rain, so rain is set to zero” consists of the statements:

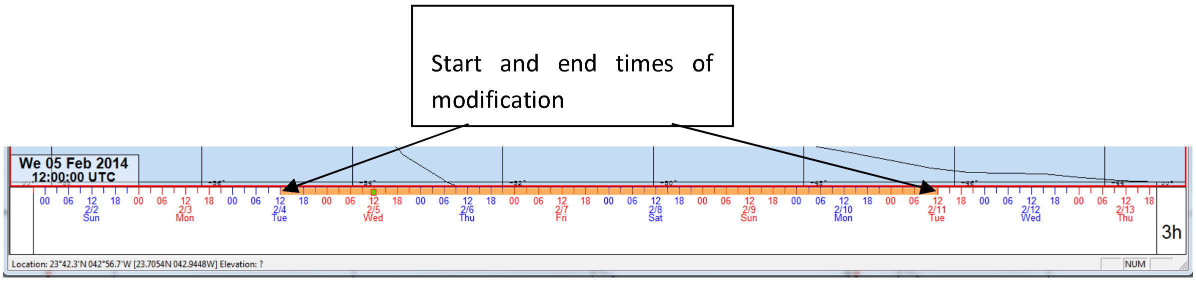
```
IF(N == 0)
{RR = 0}
```

## 3.2 SmartTool dialog



## 3.3 Time limiter (time line control)

Macros are executed for a time limited by the orange bar in the time line. There are two ways of adjusting the time limiter: First you can drag the start and the end time from the orange bar by pressing and holding down the left mouse button in the ends of the orange bar. Secondly you by keeping CTRL+SHIFT pressed down the left mouse click will adjust the start time and the right mouse click will adjust the end time.




## 3.4 Location limiter (selected points)

By default macro calculations are performed for all grid points.You can limit this with condition statements in your smart tool macro or by selecting the grid points with the selection tool and checking the  “Selected points” option in the SmartTool dialog.

## 3.5 SmartTool language (= macro language)

With condition expressions, you can set conditions to limit certain modifications. You can, for example, limit changes to land areas or rain areas etc. A condition expression block always begins with an IF expression. Assignment statements in the IF expression are calculated if the conditions are met. If necessary, you can continue an IF expression with ELSEIF or ELSE branches. In this way, you can have several different operations with different conditions side by side. A simple IF expression:

**IF(T – DP > 2) // If the difference between temperature and DP is over 2 degrees, do something…**
**T = T + 1**

You can add as many calculations as you want to an IF expression simply placing them them into a calculation block:

```
IF(T – DP > 2)
{
N=50
T = T + 1 // Only the calculations inside the block are performed, if the condition is true.
}
P = P + 1 // This calculation is always performed irrespective of any conditions.
```

| Condition expression | Description |
|---|---|
| **IF** | If you use condition expressions, you have to start with this. |
| **ELSEIF** | This can be used to add more conditions to calculations. It can only be used after IF statements. |
| **ELSE** | If other conditions are not met, this calculation is performed. It can follow an IF expression or an ELSEIF expression. |

### 3.5.1 Comparison operations

| Comparison operator | Description | Means the same as (you can use also) |
|---|---|---|
| > | Greater than | |
| >= | Greater than or equal to | |
| < | Lesser than | |
| <= | Lesser than or equal to | |
| == | Equal to | = |
| != | Not equal to | <> |

### 3.5.2 Combining conditions

You can combine conditions in condition expressions without limitations. All conditions together may need to be true or one of the combined conditions may need to be true. In the following example, two meterological masks have been used alone and combined in different ways:

```
IF(T > 14) // Mask nr. 1
IF(P < 1020) // Mask nr. 2
IF(T > 14 AND P < 1020) // Masks nr. 1 and 2 are both true at the same time (intersection)
IF(T > 14 OR P < 1020) // Either mask nr. 1 or 2 is true (union)
```

| Combining operators | Description | Means the same as (you can use either) |
|---|---|---|
| **AND** | Both conditions need to be true, intersection. | && |
| **OR** | One of the conditions needs to be true, union. | \|\| |

### 3.5.3 Variables, producers, levels and constants

Different variables, producers etc. need to have their own names in the SmartTool language. The names have been kept as short as possible. For example T stands for temperature, and so on.

| Name | Description |
|---|---|
| **T** | Temperature |
| **P** | Air pressure |
| **RH** | Relative humidity |
| **KIND** | K-index |
| **DP** | Dew point |
| **LRAD** | Long-wave radiation, earth radiation |
| **SRAD** | Short-wave radiation, solar radiation |
| **WD** | Wind direction |
| **WS** | Wind speed |
| **N** | Total cloudiness |
| **CL** | Amount of low clouds |
| **CM** | Amount of middle clouds |
| **CH** | Amount of high clouds |
| **RR** | Precipitation |
| **PREF** | Precipitation form (see values in **Error! Reference source not found.**) |
| **PRET** | Precipitation type (see values in **Error! Reference source not found.**) |
| **THUND** | Probability of thunder |
| **FOG** | Fog density (see values in **Error! Reference source not found.**) |

### 3.5.4 Producers

Normally, when you use a meteorological variable as such, it refers to the parameter of the edited data. So T means the temperature in the edited data. If you want to refer to different model datas, you need to specify this in the parameter. If you want to use the temperature from ECMWF in the data you are editing, you can do it like this:

**T = T_EC**

The producer of the most recent edited data is identified with MET.

```
IF(T > T_MET) // If T is higher than T in the last edited data,
    T = T_MET // the last edited T will be used.
```

| Name | Description |
|---|---|
| **HIR** | Hirlam model data |
| **EC** | ECMWF model data |
| **MET** | The most recently edited data |
| **GFS** | GFS Model data |
| **GEM** | Gem Model data |
| **SYNOP** | Obsevation data (†) |
| **METAR** | Metar-observation data (†) |
| **TEMP** | Sounding observation data (†) (example: T_temp_850) |

### 3.5.5 Levels

Calculations can also have parameters on different levels. This is indicated by adding a level identifier to the parameter. For example, pressure level temperature for 500hPa is expressed with T_500. If you want to indicate pressure level temperature from Gfs, you need to have both identifiers T_500_GFS. Examples:

```
T = T_850 * 0.93 // Pressure surface of the edited data (if available)
T = T_850_GFS * 0.93 // GFS's temperature for 850
T = T_GFS_850 * 0.93 // GFS's temperature for 850
```

The examples above show that you can have the pressure level and producer in which ever order you want.

| Level name | Description |
|---|---|
| **925** | Identifier of the pressure level |
| **850** | Identifier of the pressure level |
| **700** | " |
| **500** | " |
| **300** | " |

Instead of using pressure levels one may also use metric height in SmartMet. This is done by using notation Z(HEIGHT IN METERS). Example: Insert to surface temperature the temperature in the height of 100 meters from GFS- model.

```
T = T_GFS_z100
```

### 3.5.6 Named variables (var x = ?)

Script language has variables which are named. Variables are introduced with "var" as follows:

```
var x = 5 * T
```

After this, you can use the variable x like a meteorological variable, meaning that you can assign new values to it and use it in calculation and condition statements. After you have introduced the variable, you refer to it using only its name (without "var").

```
IF(x > 15)
T = x – Td
```

A variable is not a single value used in calculations. Its value can vary depending on time and location. It is calculated separately for each time and grid point (even if a constant value was assigned to it).

### 3.5.7 Identifiers of parameters, producers and layers

IF you know parameters ID number, you can use that in the name of the parameter variable. The ID for temperature is 4, so if you want, you can use the name of the variable, par4 instead of T:

```
IF(par4 < 0)
par4 = par4 * 1.1
```

The most important producers are known by the translator, but if necessary, you can use a number to identify the producer. For instance, prod131 is the identifier for ECMWF (T_EC = T_PROD131). Example:

```
IF(T > T_prod131) // If the forecast temperature is higher than the EC temperature,
T = T_prod131  // use T from EC.
```

The translator also understands the most important pressure levels. If you need other levels, you can make them available as follows: T_LEV500 (= T_500).

```
IF(T_LEV32 > 12) // for example temperature hybrid level 32
          T_LEV32  =  T_LEV32 * 1.1
```

You can also combine all the above, if needed. PAR4_PROD131_LEV500 (or alternatively with producer and level in reversed order PAR4_LEV500_PROD131) is EC temperature at 500mb pressure surface.

#### *Variable values for certain discontinuous parameters*

Some parameters are discontinuous and have codes for their values. In macro language, you need to use these code numbers directly to construct conditions, such as "if it is raining watery snow":

```
IF(PREF == 2)
```

**Precipitation form (PREF)**

| Values | Description |
|---|---|
| **0** | Drizzle |
| **1** | Water |
| **2** | Sleet |
| **3** | Snow |
| **4** | Freezing drizzle |
| **5** | Freezing rain |
| **6** | Hail |
| **7** | Value missing |

**Precipitation type (PRET)**

| Values | Description |
|---|---|
| **0** | No value |
| **1** | Large scale rain/Continuous rain |
| **2** | Convective rain/Showers |
| **3** | Value missing |

**Fog density (fog)**

| Values | Description |
|---|---|
| **0** | No fog |
| **1** | Light fog |
| **2** | Dense fog |

***Static variables***

| Name | Description | Unit |
|---|---|---|
| **TOPO** | Topographic altitude | m |
| **SLOPE** | Slope of ground surface (from horizontal plane, values between 0 and 90, at sea 0) | degree |
| **SLOPEDIR** | Direction, where the slope goes down steepest (between 1 and 360, at sea 0) | degree |
| **DISTSEA** | Shortest distance to sea (at sea 0) | km |
| **DIRSEA** | Direction to sea (between 1 and 360, at sea 0) | degree |
| **DISTLAND** | Shortest distance to land (on land 0) | km |

| | | |
|---|---|---|
| **DIRLAND** | Direction to land (between 1 and 360, on land 0) | degree |
| **LANDSEEMASK** | The propotion of land in the surface area (lakes included in calculations) | % |
| **RELTOPO** | Height in relation to surroundings (100 on top of a hill and 0 at the bottom of a hollow) | % |

***Calculated variables***

| Name | Definition |
|---|---|
| **LAT** | Latitude of the calculation point |
| **LON** | Longitude of the calculation point |
| **EANGLE** | Elevation angle. How high (between -90 and 90 degrees) the sun is at the time in the location. If the number is above zero, the sun at least peeks above the horizon. The greater the number, the higher the sun is. At night the values are negative. |

Example:

```
IF(EANGLE > 0)  // If the sun is above the horizon.
T = T + 1  // Increase temperature
```

### 3.5.8 Using previous model runs

The values from previous model run is obtained with notation *parameter_producer*[-1]. The values from the third newest model run is obtained with notation …**[-2]** respectively. Example:

```
T  = T_GFS[-1]  // insert to temperature the temperature from the previous model run of GFS.
```

Example 2: Calculate the average temperature of the last three model runs (GFS):

```
T  = (T_GFS + T_GFS[-1] + T_GFS[-2]) / 3
```

### 3.5.9 Assigning a missing value

Script language has to be very precice with missing values. Missing values easily occur in different situations and they cannot be placed freely in the edited data. You cannot assign a missing value for temperature using the normal method of assigning values. You can only assign a missing value for temperature with the following expression:

```
T = MISS
```

This expression must not contain anything else and MISS cannot be replaced with number 32700.

## 3.6 Functions

### 3.6.1 Integration functions

| Name | Description |
|---|---|
| **Avg** | Normal arithmetic average |
| **Min** | Function for finding the minimum value |
| **Max** | Function for finding the maximum value |
| **Sum** | Calculates the sum of the elements |

### 3.6.2 Areal integration

Areal integration can be employed for evening radical changes or for transferring data onto a new area. Here, area means a grid point "box" of a certain size with a relation to the calculated grid point. For example, the request of fuction

**T = MAX(T –1 –1 1 1)**

searches from the location of each calculated grid point a maximum value for temperature within the desired box. Parameters for areal integration (MAX) are:

1. T, the desired parameter (temperature)
2. –1, left edge of the box (grid point move from the calculation point)
3. –1, bottom edge of the box
4. 1, right edge of the box
5. 1, top edge of the box

### 3.6.3 Temporal integration

Temporal integration can be employed to even data over time (removing extreme values) or to move data in time. Time means the number of time steps you want to use. You can, for example, even temperature over three time steps:

**T = AVGT( –1, 1, T)**

In temporal integration, the sample parameters for functions (AVG) are:

- 1. T, the desired parameter (average of temperature)
- 2. –1, starting point for integration is one time step backwards (from the point of time in the calculated data)

- 3. 1, end point for integration is one time step forward (from the point of time in the calculated data)

| Name | Description |
|---|---|
| **Avgt** | Normal arithmetic average |
| **Mint** | Function for finding the minimum value |
| **Maxt** | Function for finding the maximum value |
| **Sumt** | Calculates the sum of the elements |

For example we make a MacroParameter (See manual part 1. chapter 4) to calculate the GFSmodel`s 24 hour precipitation sum (for the *past* 24 hours).

**RESULT = SUMT(-23, 0, RR_GFS)**

### 3.6.4 Meteorological functions

Smarttool language also includes the following basic functions that are needed especially in meteorology: Gradient, divergence, advection, laplace and rotor. Notation is as follows.

**Gradient:**
**result = grad(P_GFS)**
**Divergence:**
**result = div(ws_GFS)**
**Advection:**
**result = adv(T_GFS)**
**Laplace:**
**result = lap(T_GFS)**
**Rotor:**
**result = rot(wind_GFS)**

### 3.6.5 Mathematical functions

Mathematical functions included here have the following structure: The function is given a value as a parameter and returns a calculation value. The parameter given to the function can be, for example, a formula. Example: The square root of the difference between temperature and dew point is added to the temperature:

**T = T + SQRT(T - DP)**

The following table briefly introduces different functions.

| Function | Description | Formula: y = result and x = argument | Example |
|---|---|---|---|
| **EXP** | e to the power of x | $y = e^{x}$ | Exp(2) = 7.3891 |
| **SQRT** | Square root | $y = \sqrt{x}$ | Sqrt(9) = 3 |
| **LN** | natural logarithm | $y = \ln(x)$ | Ln(9) = 2.1972 |
| **LG** | base 10 logarithm | $y = \lg(x)$ | Lg(9) = 0.9542 |
| **SIN** | sin | $y = \sin(x)$ | Sin(120) = 0.8660 |
| **COS** | cos | $y = \cos(x)$ | Cos(120) = -0.5 |
| **TAN** | tan | $y = \tan(x)$ | Tan(120) = -1.7321 |
| **SINH** | hyperbolic sin | $y = \sinh(x)$ | Sinh(2) = 3.6269 |
| **COSH** | hyperbolic cos | $y = \cosh(x)$ | Cosh(2) = 3.7622 |
| **TANH** | hyperbolic tan | $y = \tanh(x)$ | Tanh(2) = 0.9640 |
| **ASIN** | arcus sin (-1 <= x <= 1) | $y = \arcsin(x)$ | Asin(0.5) = 30 |
| **ACOS** | arcus cos (-1 <= x <= 1) | $y = \arccos(x)$ | Acos(0.5) = 60 |
| **ATAN** | arcus tan | $y = \arctan(x)$ | Atan(0.5) = 26.6 |
| **CEIL** | rounding up | $y = ceil(x)$ | Ceil(1.1) = 2, Ceil(1.9) = 2 |
| **FLOOR** | rounding down | $y = floor(x)$ | Floor(1.1) = 1, Floor(1.9) = 1 |
| **ROUND** | rounding to the nearest | $y = round(x)$ | Round(1.1) = 1, Round(1.9) = 2 |
| **ABS** | absolute value | $y = abs(x)$ | Abs(-1.5) = 1.5, Abs(1.5) = 1.5 |
| **RAND** | random number between 0 and x | $y = rand(x)$ | Round(5) = 0 – 5 ? |

### 3.6.6 Ramp functions/masks (RU, RD and DD)

| Function | Description | Value below low limit | Change between limits | Value above high limit |
|---|---|---|---|---|
| **RD** | Ramp down | 1 | Decrease 1 -> 0 | 0 |
| **RU** | Ramp up | 0 | Increase 0 -> 1 | 1 |
| **DD** | Double ramp | -1 | Increase -1 -> 1 | 1 |
| **-DD** | Double ramp down (made using minus symbol) | 1 | Decrease 1 -> -1 | -1 |

### 3.6.7 Time functions (JDAY, LHOUR, FHOUR, MAXFHOUR)

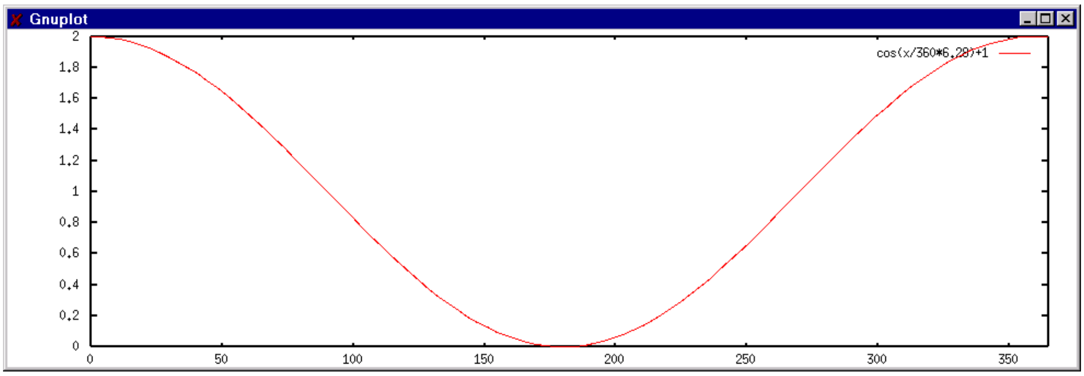
The following time functions look like variables because no parameters are assigned to them (and no empty brackets are put behind them), but they return different values depending on the season and the time of day.

### 3.6.8 JDAY or Julian day, Day of year

The **JDAY** function always returns the Julian day, which means the number of days that have elapsed since the beginning of the year (according to the Gregorian calendar) at the calculated point of time. It can be used for defining a factor that affects strongly in winter and weakly in summer. In winter, **JDAY** returns values close to 0 and 360 and in summer values close to 180. **Cosine** returns its maximum value in winter and minimum value in summer. Furthermore, value 1 needs to be added to the expression to avoid having a negative factor in summer. The expression of the example is

**T = T + 3 * (COS(JDAY) + 1)**

This expression increases temperature by 6 degrees in mid-winter and 0 degrees in mid-summer and by a value between these over the rest of the year.



If you want to have a reversed situation with a high factor in summer and a low one in winter, you can use a shift of 180 degrees:

**T = T + 3 * (COS(JDAY + 180) + 1)**

### 3.6.9 LHOUR or local hour

The LHOUR function returns the local time instead of the used forecast time (UTC). Forecasts are given in UTC time, but if you want to create changes that are strong at night and weaker during daytime, you can use the LHOUR function with sine or cosine functions. See the example in previous chapter.

The LHOUR function returns a value between 0 and 23. Local hour is calculated using the longitude and UTC time, meaning that national peculiarities and state borders are ignored. You need to multiply the figure you get from the LHOUR function by 15 if you want it to be useful with sine and cosine (360/24 = 15).

### 3.6.10 Forecast hour FHOUR ja MAXFHOUR

**FHOUR** (=Forecast HOUR) returns the forecast hour of the point of time. This is a value between 0 and **MAXFHOUR**. **MAXFHOUR** always returns the length of the forecast in hours.

#### *Visual testing aid for time-dependent functions*

You will probably often need a factor for a formula to modify for instance temperatures. For example:

**T = T + 1 * factor**

Here you also want the **factor** in the formula to be dependent on the time of the day. You can then test your formula by assigning its value directly to the parameter. For example:

**T = 5 * cos(LHOUR * 15 - 180) + 5**

You can then see the result in the time series dialog. When you think that the factor is behaving the way it should, assign it to the original formula. The formula in the figure requires a sine curve which uses factor 0 at midnight and factor 10 at noon.

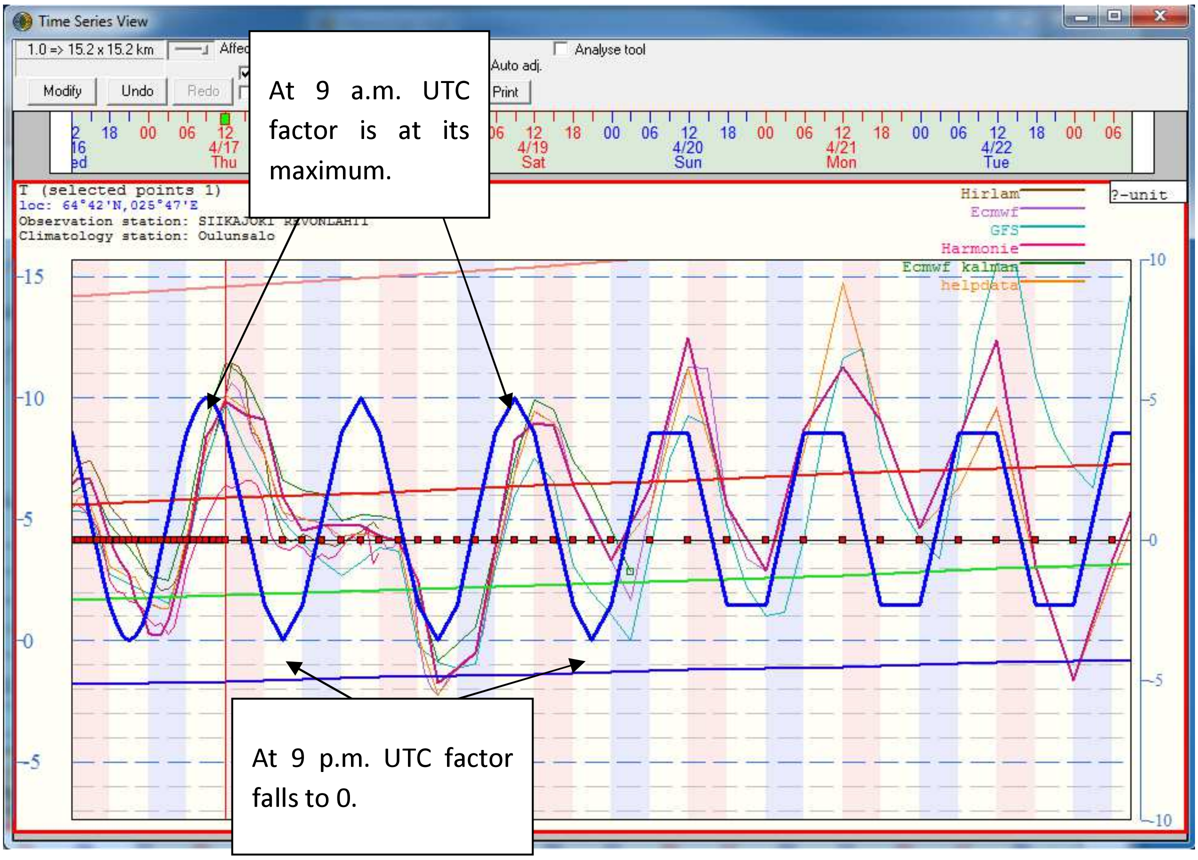
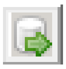
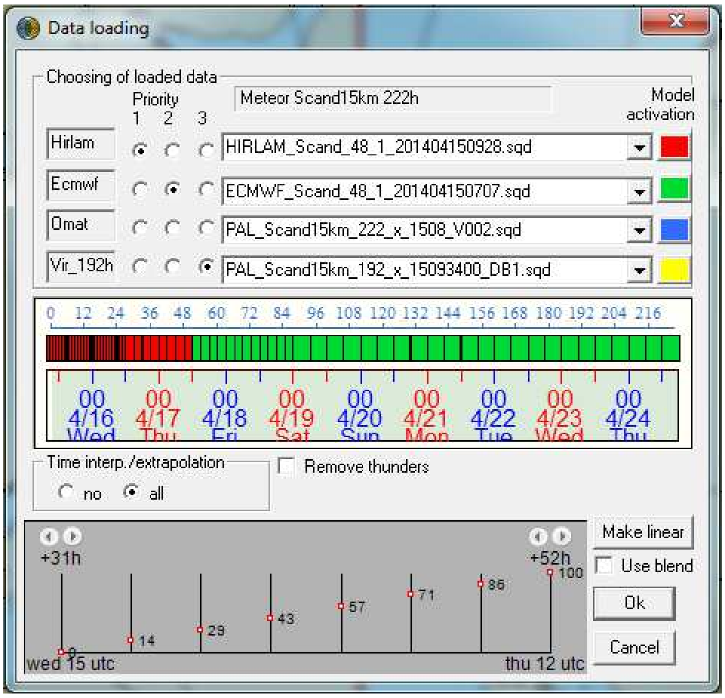



*Figure 1 Test different factor statements by assigning them directly to the parameter.*

# 4. **Editing procedure**

## 4.1 Load data

Press load data button  and select the desired model.




## 4.2 Edit Precipitation parameters and Thunder

Open up a view macro where you have editable parameters for precipitation (RR), probability of precipitation (POP), Precipitation type (PRET), precipitation form (PREF) and probability of thunder (THUND). Or you can create a new view macro where you add all the parameters and then you can modify the drawing properties to your liking if you want.

- Edit precipitation with SmartTool macros. Some example macros are in the Training folder; You can remove light precipitation, reduce precipitation randomly to make it look more like showers, increase precipitation during afternoon etc.

- If you want to give out more precise forecasts, you can also modify precipitation type, precipitation form and probability of precipitation (***PoP test***). Some example macros are given in the training folder.
- For thunder forecasts you can try out the SmartTool macros in the thunder folder. You can use the stability indeces to define thunder (***Thunder_GFS***) or for example Galvez-Davison Index based SmartTool macro (***Thunder from GDI***).

After you have done all the necessary modifications, click save , but don't send it to the database (server) yet. Also make sure that all the timesteps have acceptable values and nothing funny.

## 4.3 Edit Cloudiness and Fogs

Open up a view macro where you have editable cloudiness (N), and fog (FOG) if needed. Or you can create a new view macro of your own.

- You can edit cloudiness based on for example model RH values from different levels. See the different SmartTools in the Clouds folder.
- After executing a suitable macro, you can manually select areas and timesteps and run other macros for example ***N is 40-60 if N is below 40*** that set up the cloudiness to your liking.

After you have done all the necessary modifications, click save , but don't send it to the database (server) yet. Also make sure that all the timesteps have acceptable values and nothing funny.

## 4.4 Edit Temperature

Open up a view macro with editable temperature or create a new view macro if you don't have one.

- You can start temperature editing with SmartTool macros which modify the diurnal cycle and minimum and maximum values based on the climatological values (***temperature_October etc.***).
- Then you can run macros that modify temperature based on cloudiness (N), sun's elevation angle (EANGLE), precipitation (RR) etc.
- You can also use temperetature data from other models. For example T = T_GEM.
- As a last step you can edit temperature with the control point –tool and the time series display. See instructions in section 2.3.1. It's a easy and quick way to make sure you have the same forecasted temperature in your edited data, as in your text forecast.

After you have done all the necessary modifications, click save , but don't send it to the database (server) yet. Also make sure that all the timesteps have acceptable values and nothing funny.

## 4.5 Edit wind speed and direction

So again, start by opening up a view macro with editable wind speed (WS) and direction (WD) or wind vectors (WV). Or create a new one where you add those parameters.

- You can start by executing a SmartTool macro that increases the windspeed in sea areas a certain percentage.
- You can also use macros that create sea or land breeze.
- At least make sure that if you issue warnings or advisories, you have the same wind speed in your edited data.
- In small local changes you can also use the control point + time series method to do the modifications.

After you have done all the necessary modifications, click save , but don't send it to the database (server) yet. Also make sure that all the timesteps have acceptable values and nothing funny.

## 4.6 Save and send to the server

When you have done all of these steps, you should check that all the timesteps have reasonable data and there aren't any weird values in your forecast. After that you can send the data to the database (server) .

**! Note** though that you shouldn't send data to the server all the time, because always when the server gets new data, it will start the updating process of all the products. So, finish your editing first and at the end send it to server. Keep a minimum of 5 minutes between sending the data.