TIMESAT - a Program for Analyzing Time-Series of Satellite Sensor Data

Users Guide for TIMESAT 2.3

Per Jönsson
NMS
Teachers Education
Malmö University
SE-205 06 Malmö, Sweden

per.jonsson@ts.mah.se

Lars Eklundh
Dept. of Physical Geography
and Ecosystems Analysis
Lund University
SE-223 62 Lund, Sweden

lars.eklundh@nateko.lu.se

Malmö and Lund 2007

Contents

Part I
TIMESAT

1 Introduction

Time-series of vegetation index derived from satellite spectral measurements can be used to gain information on seasonal vegetation development. This information aids analyzes of the functional and structural characteristics of the global and regional land cover and adds to our current knowledge of global cycles of energy and matter. Long time-series of vegetation index data can also provide information on shifts in the spatial distribution of bio-climatic zones, indicating variations in large-scale circulation patterns or land-use changes.

Although the value of remotely sensed time-series data for monitoring vegetation seasons has been firmly established, only a limited number of methods for exploring and extracting seasonality parameters from such data series exists. For this reason the TIMESAT program package for extracting seasonal parameters has been developed. The program uses an adaptive Savitzky-Golay filtering method and newly developed methods based on upper envelope weighted fits to asymmetric Gaussian and double logistic model functions (Jönsson and Eklundh, 2002, 2003, 2004). From the fitted model functions a number of phenological parameters, e.g. beginning and end of the growing season, can be extracted. The TIMESAT program is tested with the 8 km by 8 km pixel resolution Pathfinder AVHRR Land (PAL) data set generated by NASA/NOAA and the 250 m by 250 m pixel resolution Terra/MODIS data. Being a completely general program TIMESAT can handle also other types of data such as meteorological index and fire data (Verbesselt *et al*, 2006).

2 Data

Vegetation index data are normally organized in images (two-dimensional spatial arrays). Each image gives the vegetation index values, y, in the array at a specified time t. By extracting vegetation index values at a pixel (j, k) in the array for consecutive times, a time-series $(t_i, y_i), \quad i = 1, 2, \ldots, N$ is obtained for this pixel (see figure 1).
The vegetation index data used in the test runs are 10- and 14-day maximum-value composites (MVC). For each 10- and 14-day period, respectively, the highest vegetation index is selected to represent the period. The method reduces negatively biased noise due to interference of clouds and atmospheric constituents. However, negatively biased residual atmospherically related noise, as well as some noise due to other factors such as surface anisotropy and sensor problems, will remain in the data.

Cloudiness products are routinely generated from different sensor data. The products range from detailed measurements of cloud properties to simple binary classifications indicating presence or absence of clouds. These products, as well as any other ancillary data, may be used to assign weights to the values in the time-series The weights can be used to improve and aid the fitting and processing procedures.

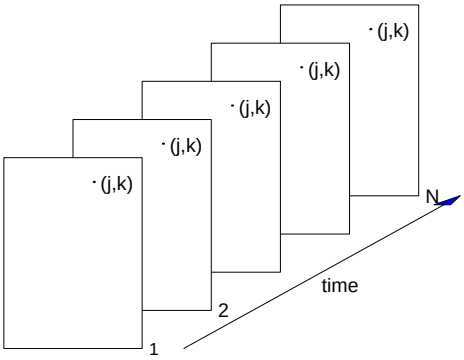

Figure 1: *Vegetation index data are organized in images. Each image gives index values for a time t. By extracting values at a pixel* (j, k) *for consecutive times, a time-series* (t_i, y_i), $i = 1, 2, \ldots, N$ *is obtained for this pixel.*

3 Methodology

TIMESAT implements three processing methods based on least-squares fits to the upper envelope of the vegetation index data. The first method uses local polynomial functions in the fitting, and the method can be classified as an adaptive Savitzky-Golay filter. The other two methods are ordinary least-squares methods, where data are fitted to model functions of different complexity. All three processing methods use a preliminary definition of the seasonality (uni-modal or bi-modal) along with approximate timings of the growing seasons.

We start by a general description of weighted least-squares fits. Pre-processing and removal of outliers is discussed and then we go on to describe an iterative method to adapt the fitted functions to the upper envelope of the data. This is followed by an account on how to determine the number of annual growing seasons and their approximate timing. The details of the three processing methods are given, and finally the extraction of seasonality information is described.

3.1 Least-squares fitting

Assume that we have a time-series (t_i, y_i), $i = 1, 2, \ldots, N$ and a model function $f(t)$ of the form

$$f(t) = c_1\varphi_1(t) + c_2\varphi_2(t) + \ldots + c_M\varphi_M(t), \tag{1}$$

where $\varphi_1(t), \varphi_2(t), \ldots, \varphi_M(t)$ are given basis functions. The best values, in the least-squares sense, of the parameters $c_1, c_2, \ldots, c_M$ are obtained by solving the system of normal equations

$$\mathbf{A}^T\mathbf{A}\mathbf{c} = \mathbf{A}^T\mathbf{b}, \tag{2}$$

where

$$A_{ij} = w_i\varphi_j(t_i), \qquad b_i = w_i y_i. \tag{3}$$

Here w_i is the weight of the ith data value, presumed to be known. Values with small weights will influence the fit less than values with large weights. If the weights are not known they may all be set to the constant value $w = 1$.

3.2 On the use of ancillary data for assigning weights

In TIMESAT cloud classifications and other ancillary data may be used to assign weights to the values in the time-series. For example, the PAL NDVI test data used in this study are accompanied by CLAVR that is a cloud indicator based on thresholds of the AVHRR reflectance and thermal channels. The original CLAVR data lie between 1 and 31, representing the three broad groups; cloudy (1–11), mixed (12–21), and clear (22–31). Values in the time-series associated with these three groups can be assigned different weights. In the test runs we will, somewhat *ad hoc*, use the weighs $w = 1$, 0.5 and 0 for values in the time-series associated with, respectively, clear, mixed and cloudy conditions. There are, of course, no general rules for converting ancillary data to weights associated with the values in the time-series and the user of the TIMESAT program is encouraged to take an experimental approach and test different settings. Figure 2(a) depicts a time-series for which the values have been assigned weights based on the CLAVR values. Large circles indicate clear conditions ($w = 1$), small circle indicate mixed conditions ($w = 0.5$), and no circle indicate clouds ($w = 0$). From the figure it is seen that several of the negatively biased outliers are associated with cloudy conditions. By assigning zero weight to these values they will not influence the subsequent fitting.

3.3 Pre-processing to remove spikes and outliers

As we have seen some spikes and outliers may be detected using ancillary data such as CLAVR. In many time-series there are, however, remaining positive and negative outliers that seriously impair the function fits. To remove these spikes data are pre-processed. Values in the time-series that are substantially different from both the left- and right-hand neighbors and from the median in a window are classified as outliers and are assigned weight 0 (see figure 2(b)). It is important to pay attention to the pre-processing since remaining spikes and outliers may seriously degrade the final function fits.

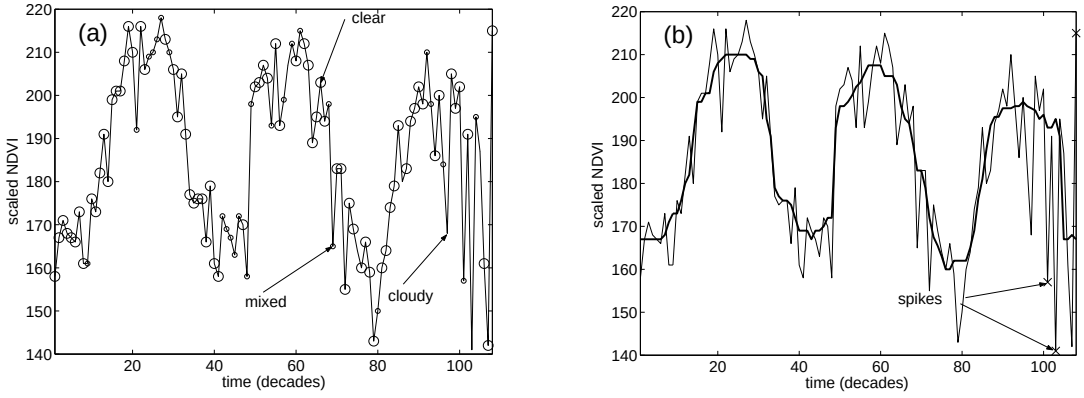

Figure 2: *(a) Time-series where the values have been assigned weights. Large circles indicate clear conditions ($w = 1$), small circles mixed conditions ($w = 0.5$), and no circle clouds ($w = 0$). (b) Time-series together with values from a median filtering. Values in the time-series that are sufficiently different from both the left- and right-hand neighbors and the median filtered value are classified as outliers and are assigned weight 0. Detected spikes and outliers are marked by crosses.*

3.4 Adaption to the upper envelope

To take into account that most noise in NDVI data, even for data classified as clear by the cloud mask, is negatively biased, the determination of the parameters $c_1, c_2, \ldots, c_N$ of the model function is done in two or more steps. In the first step the parameters are obtained by solving the system of normal equations with weight w_i obtained from the ancillary cloud data. Data points below the model function of the first fit are thought of as being less important, and in the second step the system is solved with the w_i of the low data points decreased by some factor. If deemed important, additional fits can be done until some sort of self-consistency is obtained. This multi-step procedure leads to a model function that is adapted to the upper envelope of the data (see figure 3).

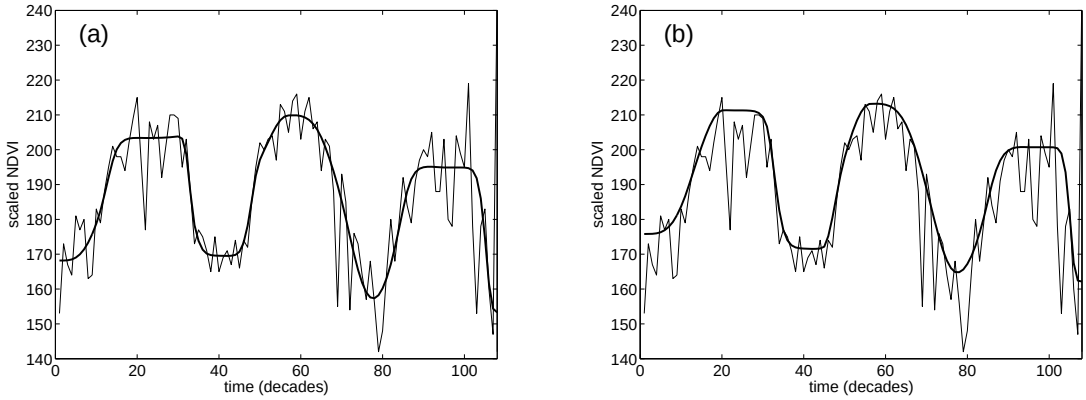

Figure 3: *Fitted functions from a multi-step procedure. The thin solid line represent the original NDVI data. (a) The thick line shows the fitted function from the first step. (b) The thick solid line displays the fit from the last step where the weights of the low data values have been decreased.*

3.5 Determination of the number of seasons

The high level of noise often makes it difficult to determine the number of annual seasons based on data for only one year. Including data from surrounding years reduces the risk for erroneous determinations. In TIMESAT, de-trended data values (t_i, y_i), $i = 1, 2, \ldots, N$ for all years in the time-series are fit to a model function

$$f(t) = c_1 + c_2 \sin(\omega t) + c_3 \cos(\omega t) + c_4 \sin(2\omega t) + c_4 \cos(2\omega t), \tag{4}$$

where $\omega = 6\pi/N$. The first basis function determines the base level whereas the pairs of sine and cosine functions correspond to, respectively, one and two annual vegetational seasons.

The fitting procedure always gives primary maxima. In addition, secondary maxima may be found. If the amplitude of the secondary maxima exceeds a certain fraction of the amplitude of the primary maxima we have two annual seasons. For cases where the amplitude of the secondary maxima is low, the number of annual seasons is set to one. The information from the fits to the sine and cosine functions is used to define intervals in which to perform the local fits to Gaussians and double logistic functions (see section 3.7).

3.6 Adaptive Savitzky-Golay filtering

One way of smoothing data and suppressing disturbances is to use a filter, and replace each data value y_i, $i = 1, \ldots, N$ by a linear combination of nearby values in a window

$$\sum_{j=-n}^{n} c_j y_{i+j}. \tag{5}$$

In the simplest case, referred to as a moving average, the weights are $c_j = 1/(2n+1)$, and the data value y_i is replaced by the average of the values in the window. The moving average method preserves the area and mean position of a seasonal peak, but alters both the width and height. The latter properties can be preserved by approximating the underlying data value, not by the average in the window, but with the value obtained from a least-squares fit to a polynomial. For each data value y_i, $i = 1, 2, \ldots, N$ we fit a quadratic polynomial $f(t) = c_1 + c_2 t + c_3 t^2$ to all $2n+1$ points in the moving window and replace the value y_i with the value of the polynomial at position t_i. The procedure above is commonly referred to as a Savitzky-Golay filter. To account for the negatively biased noise, the fitting is done in multiple steps as described in the previous section. The result is a smoothed curve adapted to the upper envelope of the values in the time-series.

The width, n, of the moving window determines the degree of smoothing, but it also affects the ability to follow a rapid change. In TIMESAT the width n can be set by the user. Even if the global setting of the moving window work fairly well, it is sometimes necessary to locally tighten the window. To capture the corresponding sudden rise in data values, only a small window can be used. In the program the Savitzky-Golay filtering is performed using the global value n of the window. The filtered data are then scanned. If there is a large increase or decrease in an interval around a data point y_i, this data point will be associated with a smaller window. The filtering is then redone with the new locally adapted size of the window. The adaptive procedure is illustrated in Fig. 4.

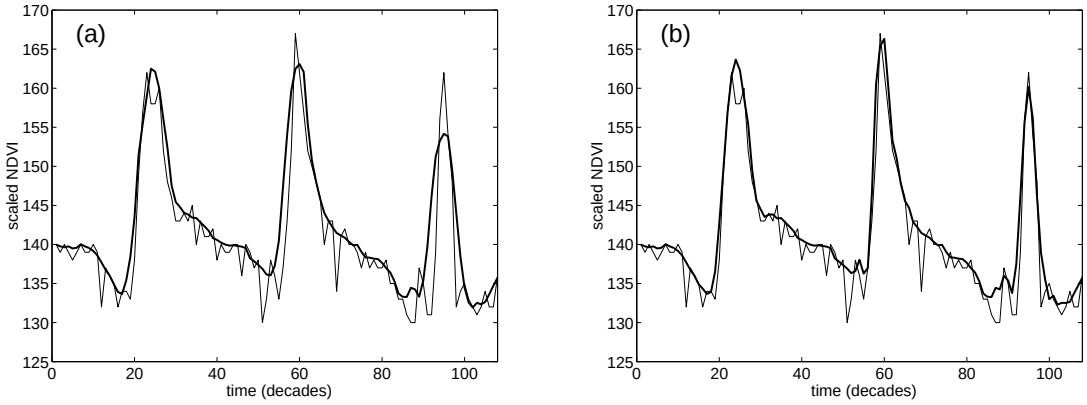

Figure 4: *In (a) the filtering is done with $n = 5$, which is too large for the filtered data to follow the sudden increase and decrease of the underlying data values. A scan of the filtered data identifies the data points for which there are large increases or decreases in surrounding intervals. Setting $n = 3$ for these points and redoing the filtering gives the curve in (b). Note the improved fit at the rising edges and at the narrow seasonal peaks.*

3.7 Non-linear least-squares fits to asymmetric Gaussians and double logistic functions

In these two methods local model functions are fit to data in intervals around maxima and minima in the time-series. The local model functions have the general form

$$f(t) \equiv f(t; \mathbf{c}, \mathbf{x}) = c_1 + c_2\, g(t; \mathbf{x}), \tag{6}$$

where the linear parameters $\mathbf{c} = (c_1, c_2)$ determine the base level and the amplitude. The non-linear parameters $\mathbf{x} = (x_1, x_2, \ldots, x_n)$. Determines the shape of the basis function $g(t; \mathbf{x})$.

Asymmetric Gaussians

In this case the basis function

$$g(t; x_1, x_2, \ldots, x_5) = \begin{cases} \exp\left[-\left(\dfrac{t - x_1}{x_2}\right)^{x_3}\right] & \text{if } \; t > x_1 \\ \exp\left[-\left(\dfrac{x_1 - t}{x_4}\right)^{x_5}\right] & \text{if } \; t < x_1 \end{cases} \tag{7}$$

is a Gaussian type of function. For this function x_1 determines the position of the maximum or minimum with respect to the independent time variable t, while x_2 and x_3 determine the width and flatness (kurtosis) of the right function half. Similarly, x_4 and x_5 determine the width and flatness of the left half. The effects of varying the parameters $x_2, \ldots, x_5$ are shown in figure. 5.

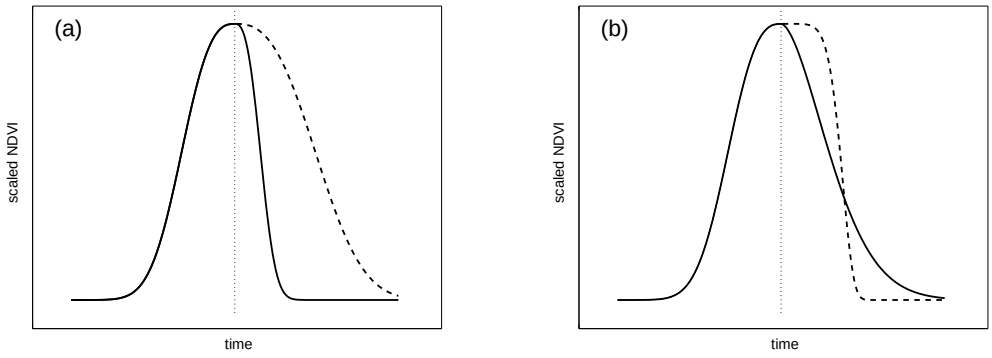

Figure 5: *Effect of parameter changes on the local functions. In (a) the parameter x_2, which determines the width of the right function half, has been decreased (solid line) and increased (dashed line) compared to the value of the left half. In (b) the parameter x_3, which determines the flatness of the right function half, has been decreased (solid line) and increased (dashed line) compared to the value of the left half.*

In order to ensure smooth shapes of the model functions, consistent with what is observed for data, the parameters $x_2, \ldots, x_5$ are restricted in range. For example, x_3 and x_5 are assumed to be larger than 2 in order to avoid a cusp at the matching point $t = x_1$ of the Gaussian function.

Double logistic functions

Here the basis function

$$g(t; x_1, ..., x_4) = \frac{1}{1 + \exp\left(\frac{x_1 - t}{x_2}\right)} - \frac{1}{1 + \exp\left(\frac{x_3 - t}{x_4}\right)} \tag{8}$$

is a double logistic function. x_1 determines the position of the left inflection point while x_2 gives the rate of change. Similarly x_3 determines the position of the right inflection point while x_4 gives the rate of change at this point. Also for this function the parameters are restricted in range to ensure a smooth shape.

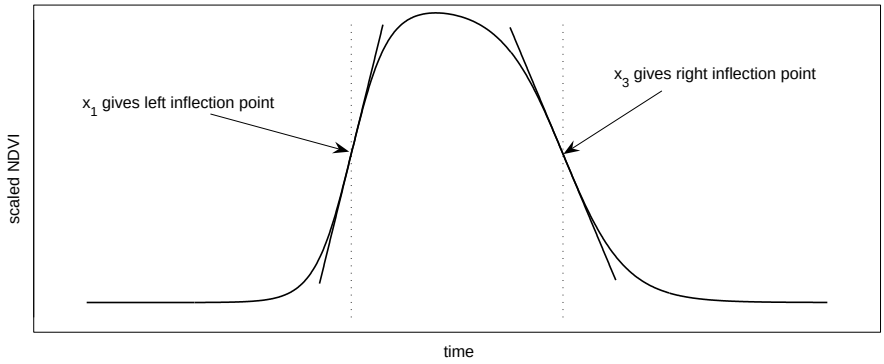

Figure 6: *In the double logistic function x_1 determines the position of the left inflection point while x_2 gives the rate of change. Similarly x_3 determines the position of the right inflection point while x_4 gives the rate of change at this point.*

The local model functions are well suited for describing the shape of the time-series in overlapping intervals around maxima and minima. Given a set of data points (t_i, y_i), $i = n_1, \ldots, n_2$ in an interval around a maximum or a minimum, the parameters $\mathbf{c}$ and $\mathbf{x}$ are obtained by minimizing the merit function

$$\chi^2 = \sum_{i=n_1}^{n_2} \left[w_i(f(t_i, \mathbf{c}, \mathbf{x}) - y_i)\right]^2 . \tag{9}$$

The function depends linearly on $\mathbf{c}$ and non-linearly on $\mathbf{x}$. In TIMESAT the minimization is done using a separable Levenberg-Marquardt method, where the box constraints on the non-linear parameters are enforced by projecting onto the feasible parameter interval. The separable Levenberg-Marquardt method is discussed in K. Madsen, H.B. Nielsen and O. Tingleff (2002) and in H.B. Nielsen (1999, 2000). Box constraints are analyzed in C. Kanzow, N. Yamashita and M. Fukushima (2002).

For the asymmetric Gaussians and double logistic functions the fitting is done in steps, as described in section 3.4, to account for the negatively biased noise.

3.8 Merging of local functions

The local model functions describe sensor data very well in broad intervals around maxima and minima. At the limbs, however, the fits are less good. Denoting the local functions describing the time-series in intervals around the left minima, the central maxima and the right minima by $f_L(t)$, $f_C(t)$, and $f_R(t)$ (see figures 7(a-c)), the global function $F(t)$, that correctly models the time-series in the full interval $[t_L, t_R]$, is

$$F(t) = \begin{cases} \alpha(t) f_L(t) + [1 - \alpha(t)] f_C(t), & t_L < t < t_C \\ \beta(t) f_C(t) + [1 - \beta(t)] f_R(t), & t_C < t < t_R. \end{cases} \tag{10}$$

Here $\alpha(t)$ and $\beta(t)$ are cut-off functions that in small intervals around $(t_L + t_C)/2$ and $(t_C + t_R)/2$, respectively, smoothly drop from 1 to 0. Loosely speaking, the global function $F(t)$, shown in figure 7(d), assumes the character of $f_L(t)$, $f_C(t)$ and $f_R(t)$ in, respectively, the left, central and right part of the interval $[t_L, t_R]$. The merging of local functions to a global function is a key feature of the method. It increases the flexibility and allows the fitted function to follow a complex behavior of the time-series (Jönsson and Eklundh, 2002).

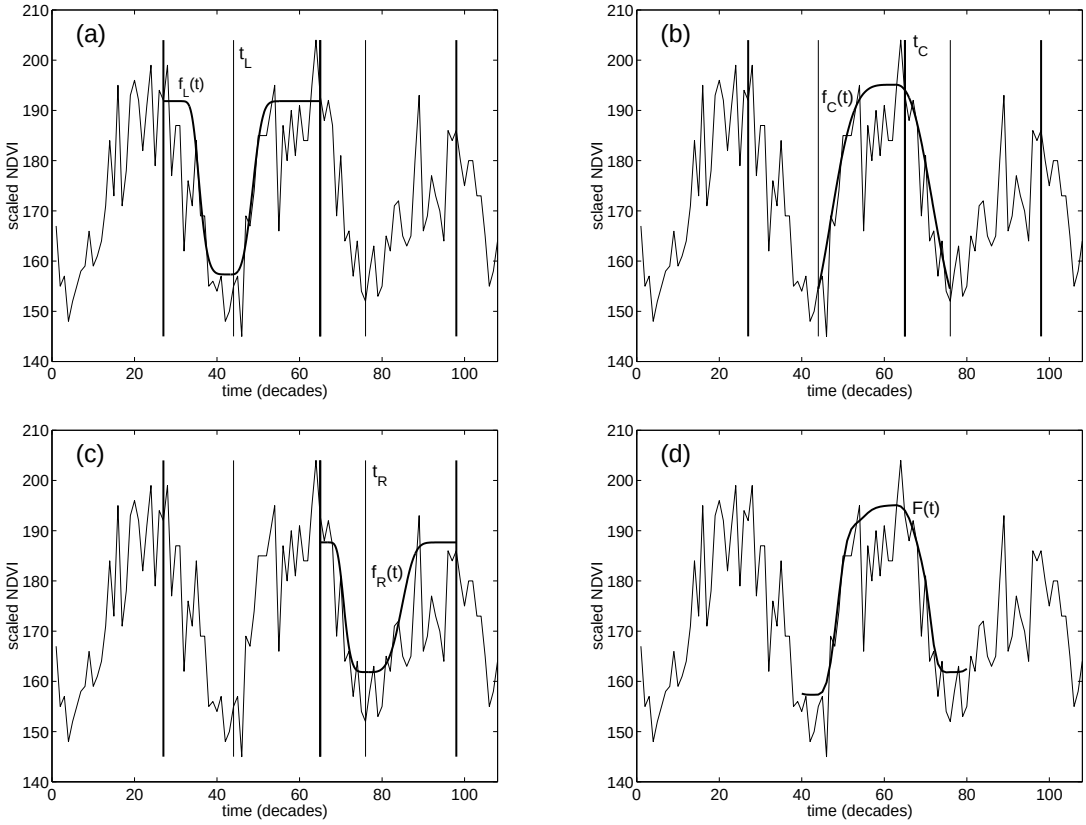

Figure 7: *(a-c) display local model functions fitted to, respectively, the left minimum, the central maximum, and the right minimum. (d) shows the global model function that is obtained by merging the three local functions.*

4 Phenological parameters

Consider a time-series with one growing season a year. During a period of n years there will, in the general case, be $n-1$ full seasons together with two fractions of a season in the beginning and end of the time-series. Using the fitted functions phenological parameters can be extracted for each of the $n-1$ full seasons (see figure 8).

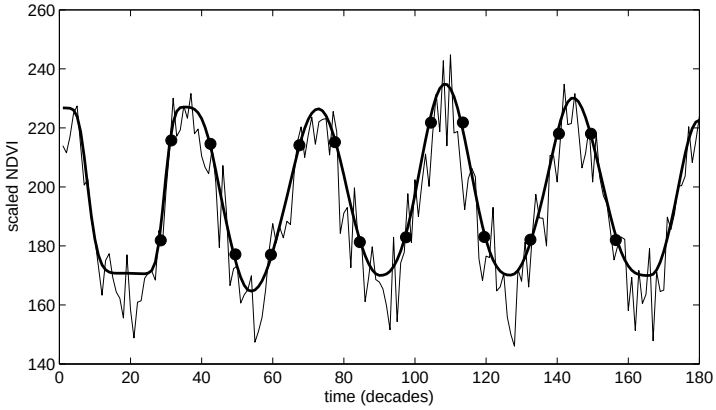

Figure 8: *Time-series covering a period of 5 years. Phenology parameters from the 4 full seasons are determined from fitted functions. The start and end of the seasons as well as the 80% levels are marked with filled circles.*

In the current version of TIMESAT the following phenological parameters are computed for each of the full seasons.

1. **time for the start of the season;** time for which the left edge has increased to a user defined level (often 20% of the seasonal amplitude) measured from the left minimum level.
2. **time for the end of the season;** time for which the right edge has decreased to a user defined level measured from the right minimum level.
3. **length of the season;** time from the start to the end of the season.
4. **base level;** given as the average of the left and right minimum values.
5. **time for the mid of the season;** computed as the mean value of the times for which, respectively, the left edge has increased to the 80 % level and the right edge has decreased to the 80 % level.
6. **largest data value for the fitted function during the season;**
7. **seasonal amplitude;** difference between the maximal value and the base level.
8. **rate of increase at the beginning of the season;** calculated as the ratio between the values evaluated at the season start and at the left 80 % level divided by the corresponding time difference.
9. **rate of decrease at the end of the season;** calculated as the ratio between the values evaluated at the season end and at the right 80 % level divided by the corresponding time difference.

10. **large seasonal integral;** integral of the function describing the season from the season start to the season end.

11. **small seasonal integral;** integral of the difference between the function describing the season and the base level from season start to season end.

5 The TIMESAT package

5.1 Download

The TIMESAT package consists of a set of programs written in Matlab and Fortran together with test data and documentation. The package can be downloaded from

`http://www.natgeo.lu.se/personal/Lars.Eklundh/TIMESAT/timesat.html`.

Unzipping the file **timesat2_3.zip** gives the following directory structure

```
TIMESAT2_3
  -- DATA
  -- DOCUMENTATION
  -- TIMESAT_MATLAB
  -- TIMESAT_FORTRAN
  -- TOOLS
  -- RUN
```

5.2 Overview

The data needed for the test runs reside in `DATA`. Documentation can be found in `DOCUMENTATION`. In `TIMESAT_MATLAB` the two Matlab programs **timesatseries** and **timesatimage** can be found. The first program reads formatted lists of time-series whereas the second program reads sensor data image files. Both programs have a graphical interface that allows the user to display data and fitted functions. The directory `TIMESAT_FORTRAN` holds the Fortran versions of the programs **timesatseries** and **timesatimage**. The Fortran programs work in exactly the same way as the Matlab programs, but there is no graphical interface. The Fortan programs run very fast and it possible to process large amount of data on an ordinary PC. In `TOOLS` a number of supporting Fortran programs have been collected. These programs read the output data structures from the **timesatseries** and **timesatimage** programs and produce binary images with phenological data that can be displayed with IDRISI, PCI or any equivalent image processing software. There are also a number of routines in Matlab that can be used to read the output data structures. The directory `TIMESAT_MATLAB` also contains the Matlab program **timesatGUI**. This program is based on a user friendly graphical user interface where all parameter settings can be changed interactively by pressing buttons. This program is useful for getting acquainted with the data and to test different settings of the parameters that governs the fits. We recommend the user to try it out! A separate manual for **timesatGUI** can be found in `DOCUMENTATION`.

5.3 Compiling the Fortran programs

In `TIMESAT_FORTRAN` the executables **timesatseries** and **timesatimage**, for both Windows and Linux, as well as the Fortran source code can be found. The source code consists of

the two main programs together with 14 subroutines. The source code can be compiled if modifications of the programs are needed or if the executables are requested for platforms other than Windows and Linux.

The program components have been tested with the HP F90 compiler for Windows and with the Intel compiler for Linux. With small modifications the code also runs under GNU Fortran F95. To compile the **timesatseries** program using the HP F90 compiler start a DOS window, change to the `TIMESAT_FORTRAN` directory and give the following command

```
>f90 /check:bounds /traceback timesatseries.f90 basis.f90 fitgauss.f90
   fitlogistic.f90 fungauss.f90 funlogistic.f90 gauss.f90 linlsq.f90
   marquardt.f90 median.f90 modweight.f90 phenology.f90 savgol.f90
   season.f90 spike.f90
```

The **timesatimage** program is compiled in the same way but with the main program `timesatseries.f90` changed to `timesatimage.f90`.

6 Input and output

In this section we describe the input data needed to run the **timesatseries** and **timesatimage** programs. We also describe the output data. The sections on the two programs can be read independently from each other and thus there will be some overlap and duplication in the presentation.

6.1 The timesatseries program

To run the **timesatseries** program the user needs to prepare a text file with time-series of sensor data values and, optionally, a file with the corresponding mask data values. The program reads the files and processes the time-series under the controll of a few input parameters. Fitted functions and extracted phenological data are written to file (see figure 9).

Figure 9: *The* **timesatseries** *program reads the sensor data file and, optionally, the mask data file and processes the time-series under the controll of a few of input parameters. Fitted functions and extracted phenological data are written to file.*

Input files

The sensor data file has the following structure

$$
\begin{array}{l}
nyear\ nptperyear\ nts \\
\left.\begin{array}{cccc} y_1 & y_2 & \cdots & y_N \\ y_1 & y_2 & \cdots & y_N \\ & & & \\ y_1 & y_2 & \cdots & y_N \end{array}\right\} nts
\end{array}
$$

where *nyears* gives the number of years spanned by the time-series, *nptperyear* is number of data values in one year and *nts* is the number of time-series in the file. $y_1, y_2, \ldots, y_N$ are the time-series of sensor data values. The mask data file has a similar structure

Input parameters

The processing of the time-series is governed by a few input parameters.

1. **Give name of the sensor data file;** the first line of the file should give; number of whole years, number of points per year, number of time-series. Then comes the time-series, one per row.

2. **Range for sensor data values;** sensor data values are assumed to be in the range $[y_{low}, y_{high}]$. Values outside this range are assigned weight zero. This setting can be used to exclude obvious outliers in data.

3. **Give name of mask data file (none if not available);** if a mask data file is available the first line of the file should give; number of whole years, number of points per year, number of time-series. Then comes the time-series, one per row.

4. **Conversion of mask data to weights;** if mask data are available the user should define three ranges $[a_1, b_1]$, $[a_2, b_2]$, $[a_3, b_3]$ and corresponding weights w_1, w_2, w_3. If a mask data value is in range $[a_i, b_i]$, $i = 1, 2, 3$, then the corresponding sensor data value is assigned weight w_i. If a mask data value is outside the three ranges the sensor data value is assigned weight zero.

5. **Cutoff for amplitude in sensor data;** only time-series with an amplitude, as obtained from the fit to the sine and cosine functions (see section 3.5), higher than a certain cutoff are processed. This setting can be used to make sure that uninteresting time-series with low variation, e.g. time-series over deserts or water, are not processed.

6. **Cutoff for spike;** spikes are detected by comparing each data value y_i, $i = 1, 2, \ldots, N$ with median filtered values and with closest neighbors. If the distance is greater than $S \times y_{std}$, where y_{std} is the standard deviation for the values in the time-series, we have a spike and the value y_i is assigned weight 0.

7. **Parameter for determining the number of annual seasons;** the number of annual seasons are determined from the fitted sine and cosine functions. The fitting procedure always gives primary maxima. In addition, secondary maxima may be found. If the amplitude of the secondary maxima exceeds a certain fraction of the amplitude of the primary maxima we have two annual seasons. Setting the fraction close to 0 causes the

program to interpret a small depression in the main curve as a second annual season. Setting the fraction close to 1 forces the program to process data as if there were only one annual season. The latter setting should be used for areas where we know that there is only one annual season.

8. **Number of fitting steps;** the user should specify the number of fitting steps (1,2 or 3). If more than one step the fitted functions are forced to the upper envelope of the sensor data (see section 3.4).

9. **Strength of envelope adaptation;** a value between 1 and 10, where 10 strongly forces the fit to follow the upper envelope of the data.

10. **Specify the processing methods;** indicate what processing methods should be used. Savitzky-Golay filter, asymmetric Gaussians, or double logistic functions can be used. Asymmetric Gaussian and double logistic often give similar results.

11. **Set window size**; define window size for the Savitzky-Golay filter (see section 3.6). A small window captures abrupt changes in the beginning and end of the growing seasons but yields comparatively little smoothing. A large window give smooth curves at the cost of less accurate time information. Separate window sizes are set for each of the fitting steps.

12. **Set the level that defines the season start and end;** the start and end of the season are defined as the times for which the fitted curves have, respectively, increased and decreased to a user defined level measured from the minimum level.

13. **Job name;** give a string that will be used to identify the output data.

Output files

The program outputs the time-series of sensor data to the binary file **sensordata**. Functions from Savitzky-Golay, asymmetric Gauss and double logistic fits are written to, respectively, the binary files **fitSG_job**, **fitAG_job** and **fitDL_job** where **job** is an identification string. The files are organized in the following way

$$
\begin{array}{l}
nyears \;\; nptperyear \;\; nd1 \;\; nd2 \;\; nd3 \;\; nd4 \\
row_1 \;\; col_1 \\
y_1 \;\; y_2 \;\; y_3 \;\; \cdots \;\; y_{N-1} \;\; y_N \\
row_2 \;\; col_2 \\
y_1 \;\; y_2 \;\; y_3 \;\; \cdots \;\; y_{N-1} \;\; y_N \\
\quad \vdots \\
row_M \;\; col_M \\
y_1 \;\; y_2 \;\; y_3 \;\; \cdots \;\; y_{N-1} \;\; y_N
\end{array}
$$

where *nyears* gives the number of years spanned by the time-series, *nptperyear* is number of data values in one year. *nd1, nd2, nd3, nd4* are dummy variables. $row_1, row_2, \ldots, row_M$ give the position of the time-series in the file **sensordata.txt**. $col_1, col_2, \ldots, col_M$ are, since we are reading individual time-series from a list, set to one.

Extracted phenological parameters from the fitted functions are written to the binary files **phenologySG_job**, **phenologyAG_job** and **phenologyDL_job** where **job** is an identification string. The files have the following structure

$$
\begin{array}{l}
nyears \;\; nptperyear \;\; nd1 \;\; nd2 \;\; nd3 \;\; nd4 \\
row_1 \;\; col_1 \;\; n_1 \\
\left.\begin{array}{llllllllllll}
p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11} \\
p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11} \\
 & & & & & \vdots & & & & & \\
p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11}
\end{array}\right\} n_1 \text{ lines with values} \\
row_2 \;\; col_2 \;\; n_2 \\
\left.\begin{array}{llllllllllll}
p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11} \\
p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11} \\
 & & & & & \vdots & & & & & \\
p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11}
\end{array}\right\} n_2 \text{ lines with values} \\
\quad \vdots \\
row_M \;\; col_M \;\; n_M \\
\left.\begin{array}{llllllllllll}
p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11} \\
p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11} \\
 & & & & & \vdots & & & & & \\
p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 & p_9 & p_{10} & p_{11}
\end{array}\right\} n_M \text{ lines with values}
\end{array}
$$

Here $n_1, n_2, \ldots, n_M$ gives the number of full seasons for which seasonality information has been determined. This information is followed by phenological parameters $p_1, p_2, \ldots, p_{11}$ for each of the the full seasons.

In the function and phenology files *nyears, nptperyear, nd1, nd2, nd3, nd4, row, col, n* are integers with the format `int32` whereas function values $y_1, y_2, \ldots, y_N$ and the phenology parameters $p_1, p_2, \ldots, p_{11}$ are written in single precision `real*4`.

The **timesatseries** program writes the input parameters to a log file **input_job.txt**, where **job** is an identification string. This file serves two purposes. Firstly, it allows the user to go back and look at the input parameters for a certain run that maybe was done several weeks ago. Secondly, the file can be used as input if the **timesatseries** program is to be run in batch mode.

6.2 The timesatimage program

To run the **timesatimage** program the user needs to prepare a sensor image list with name and path to the sensor data images that are needed to generate the time-series (see figure 1) and, optionally, a mask image list with the name and path to the corresponding mask data images. The program reads the image lists and opens and reads the sensor data and mask data images to extract time-series for a defined area. The extracted time-series are then processed under the controll of some input parameters. Fitted functions and phenological data are written to file (see figure 10).

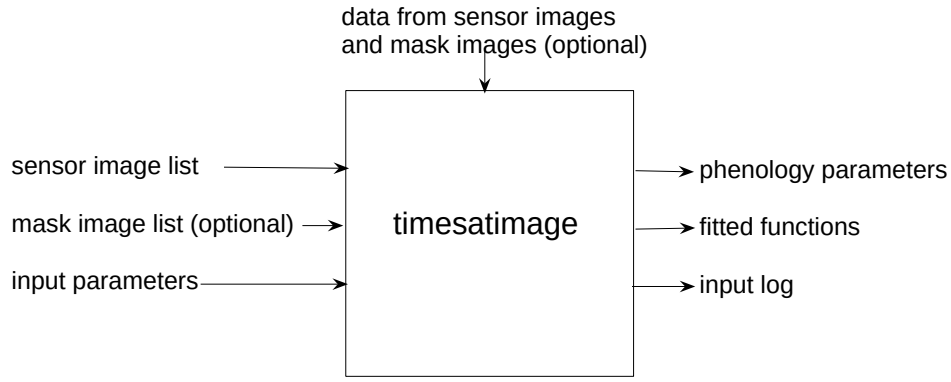

Figure 10: *The* **timesatimage** *program reads the sensor image list and, optionally, the mask image list. The program opens and reads the sensor data and mask data images to extract time-series for a defined area. The time-series are processes under the controll of a some of input parameters. Fitted functions and phenological data are written to file.*

Input files

The sensor image list is a text file with the following structure

$$\begin{array}{l} N \\ \left.\begin{array}{l} /path/imagename1 \\ /path/imagename2 \\ \quad\vdots \\ /path/imagenameN \end{array}\right\} N \text{ images} \end{array}$$

where N gives the number of images needed to generate the time-series. After this the path and names of the image files are given. The mask image list is a text file with a similar structure

Input parameters

The processing of the time-series is governed by a few input parameters. The first five are specific for the **timesatimage** program the remaining parameters are similar to those in the **timesatseries** program.

1. **Name of the sensor image list;**
2. **File type for image files;** specify the file type of the sensor and mask data images. The following type are currently supported (1) 8-bit unsigned integer (byte), (2) 16-bit signed integer (3) 32-bit real.
3. **Number of rows and columns in the image tile;**
4. **Define processing window: start row, stop row, start column, stop column;**
5. **Number of years and number points per year in the time-series;**
6. **Range for sensor data values;** The sensor data values are assumed to be in the range $[y_{low}, y_{high}]$. This setting can Values outside this range are assigned weight zero. This setting can be used to capture obvious spikes in data.

7. **Give name of mask image list (none if not available);**

8. **Conversion of mask data to weights;** if mask data are available the user should define three ranges $[a_1, b_1]$, $[a_2, b_2]$, $[a_3, b_3]$ and corresponding weights w_1, w_2, w_3. If a mask data value is in range $[a_i, b_i]$, $i = 1, 2, 3$, then the corresponding sensor data value is assigned weight w_i. If a mask data value is outside the three ranges the sensor data value is assigned weight zero.

9. **Cutoff for amplitude in sensor data;** only time-series with an amplitude, as obtained from the fit to the sine and cosine functions (see section 3.5), higher than a certain cutoff are processed. This setting can be used to make sure that uninteresting time-series with low variation, e.g. time-series over deserts or water, are not processed.

10. **Cutoff for spike;** spikes are detected by comparing each data value y_i, $i = 1, 2, \ldots, N$ with median filtered values and with closest neighbors. If the distance is greater than $S \times y_{std}$, where y_{std} is the standard deviation for the values in the time-series, we have a spike and the value y_i is assigned weight 0.

11. **Parameter for determining the number of annual seasons;** the number of annual seasons are determined from the fitted sine and cosine functions. The fitting procedure always gives primary maxima. In addition, secondary maxima may be found. If the amplitude of the secondary maxima exceeds a certain fraction of the amplitude of the primary maxima we have two annual seasons. Setting the fraction close to 0 causes the program to interpret a small depression in the main curve as a second annual season. Setting the fraction close to 1 forces the program to process data as if there were only one annual season. The latter setting should be used for areas where we know that there is only one annual season.

12. **Number of fitting steps;** the user should specify the number of fitting steps (1,2 or 3). If more than one step the fitted functions are forced to the upper envelope of the sensor data (see section 3.4).

13. **Strength of envelope adaptation;** a value between 1 and 10, where 10 strongly forces the fit to follow the upper envelope of the data.

14. **Specify the processing methods;** indicate what processing methods should be used. Savitzky-Golay filter, asymmetric Gaussians, or double logistic functions can be used. Asymmetric Gaussian and double logistic often give similar results.

15. **Set window size**; define window size for the Savitzky-Golay filter (see section 3.6). A small window captures abrupt changes in the beginning and end of the growing seasons but yields comparatively little smoothing. A large window give smooth curves at the cost of less accurate time information. Separate window sizes are set for each of the fitting steps.

16. **Set the level that defines the season start and end;** the start and end of the season are defined as the times for which the fitted curves have, respectively, increased and decreased to a user defined level measured from the minimum level.

17. **Job name;** give a string that will be used to identify the output data.

Output files

The program outputs the time-series of sensor data to the binary file **sensordata**. Functions from Savitzky-Golay, asymmetric Gauss and double logistic fits are written to, respectively, the binary files **fitSG_job**, **fitAG_job** and **fitDL_job** where **job** is an identification string. The files are organized in the following way

$$
\begin{array}{l}
nyears \;\; nptperyear \;\; rowstart \;\; rowstop \;\; colstart \;\; colstop \\
row_1 \;\; col_1 \\
y_1 \;\; y_2 \;\; y_3 \;\; \cdots \;\; y_{N-1} \;\; y_N \\
row_2 \;\; col_2 \\
y_1 \;\; y_2 \;\; y_3 \;\; \cdots \;\; y_{N-1} \;\; y_N \\
\quad \vdots \\
row_M \;\; col_M \\
y_1 \;\; y_2 \;\; y_3 \;\; \cdots \;\; y_{N-1} \;\; y_N
\end{array}
$$

where *nyears* gives the number of years spanned by the time-series, *nptperyear* is number of data values in one year. *rowstart, rowstop, colstart, colstop* give the area that has been processed. $row_1, col_1, row_1, col_1, \ldots, row_M, col_M$ give the position of the time-series in the image.

Extracted phenological parameters from the fitted functions are written to the binary files **phenologySG_job**, **phenologyAG_job** and **phenologyDL_job** where **job** is an identification string. The files have the following structure

$$
\begin{array}{l}
nyears \;\; nptperyear \;\; rowstart \;\; rowstop \;\; colstart \;\; colstop \\
row_1 \;\; col_1 \;\; n_1 \\
\left.\begin{array}{l}
p_1 \;\; p_2 \;\; p_3 \;\; p_4 \;\; p_5 \;\; p_6 \;\; p_7 \;\; p_8 \;\; p_9 \;\; p_{10} \;\; p_{11} \\
p_1 \;\; p_2 \;\; p_3 \;\; p_4 \;\; p_5 \;\; p_6 \;\; p_7 \;\; p_8 \;\; p_9 \;\; p_{10} \;\; p_{11} \\
\qquad\qquad\qquad\qquad\quad \vdots \\
p_1 \;\; p_2 \;\; p_3 \;\; p_4 \;\; p_5 \;\; p_6 \;\; p_7 \;\; p_8 \;\; p_9 \;\; p_{10} \;\; p_{11}
\end{array}\right\} n_1 \text{ lines with values} \\
row_2 \;\; col_2 \;\; n_2 \\
\left.\begin{array}{l}
p_1 \;\; p_2 \;\; p_3 \;\; p_4 \;\; p_5 \;\; p_6 \;\; p_7 \;\; p_8 \;\; p_9 \;\; p_{10} \;\; p_{11} \\
p_1 \;\; p_2 \;\; p_3 \;\; p_4 \;\; p_5 \;\; p_6 \;\; p_7 \;\; p_8 \;\; p_9 \;\; p_{10} \;\; p_{11} \\
\qquad\qquad\qquad\qquad\quad \vdots \\
p_1 \;\; p_2 \;\; p_3 \;\; p_4 \;\; p_5 \;\; p_6 \;\; p_7 \;\; p_8 \;\; p_9 \;\; p_{10} \;\; p_{11}
\end{array}\right\} n_2 \text{ lines with values} \\
\quad \vdots \\
row_M \;\; col_M \;\; n_M \\
\left.\begin{array}{l}
p_1 \;\; p_2 \;\; p_3 \;\; p_4 \;\; p_5 \;\; p_6 \;\; p_7 \;\; p_8 \;\; p_9 \;\; p_{10} \;\; p_{11} \\
p_1 \;\; p_2 \;\; p_3 \;\; p_4 \;\; p_5 \;\; p_6 \;\; p_7 \;\; p_8 \;\; p_9 \;\; p_{10} \;\; p_{11} \\
\qquad\qquad\qquad\qquad\quad \vdots \\
p_1 \;\; p_2 \;\; p_3 \;\; p_4 \;\; p_5 \;\; p_6 \;\; p_7 \;\; p_8 \;\; p_9 \;\; p_{10} \;\; p_{11}
\end{array}\right\} n_M \text{ lines with values}
\end{array}
$$

Here $n_1, n_2, \ldots, n_M$ gives the number of full seasons for which seasonality information has been determined. This information is followed by phenological parameters $p_1, p_2, \ldots, p_{11}$ for each of the the full seasons.

In the function and phenology files *nyears, nptperyear, rowstart, rowstop, colstart, colstop, row, col, n* are integers with the format `int32` whereas function values $y_1, y_2, \ldots, y_N$ and the phenology parameters $p_1, p_2, \ldots, p_{11}$ are written in single precision `real*4`.

The **timesatimage** program writes the input parameters to a log file **input_job.txt**, where **job** is an identification string. This file serves two purposes. Firstly, it allows the user to go back and look at the input parameters for a certain run that maybe was done several weeks ago. Secondly, the file can be used as input if the **timesatimage** program is to be run in batch mode.

6.3 TimesatGUI

The **timesatGUI** program can process both individual time-series and images. Parameters controlling the fits can be changed interactively using the graphical user interface. By playing around the user can find an optimal values of the parameters. These parameters can then be used as input for **timesatimage**. The extracted phenological parameters are displayed in a separate window. In figure 11 the graphical interface is displayed.

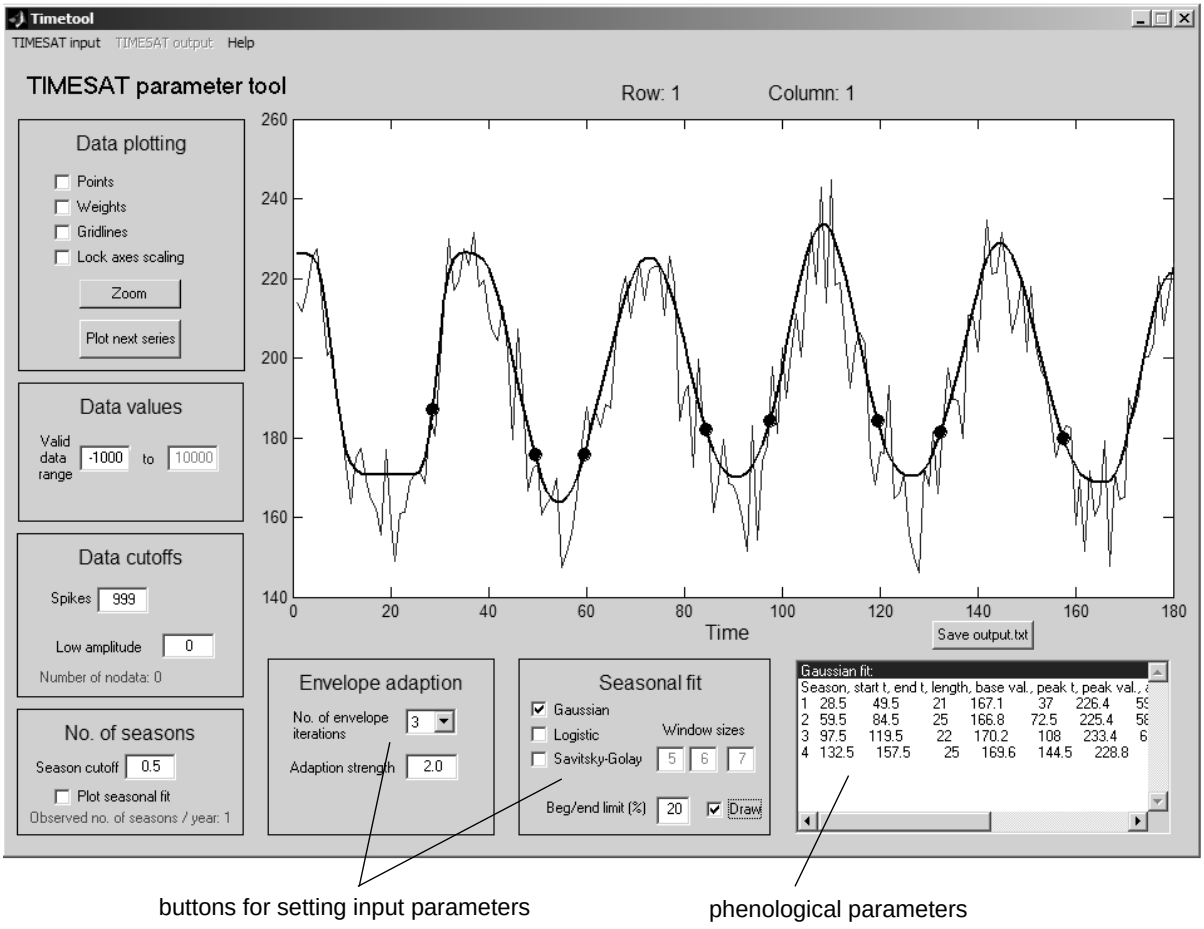

Figure 11: *Graphical interface to the* **timesatGUI** *program. Detailed information on how to use the program can be found in a separate manual.*

7 Test runs

7.1 Timesat single time-series (Matlab)

The test file **sensordata.txt** with one time-series covering five years and with 36 data values a year resides in the directory `RUN`. To perform a test run start Matlab and add the directory `TIMESAT_MATLAB` to the search path. Move to the directory `RUN` and complete the following dialog in the Matlab command window

```
>timesatseries

  TIMESAT series version 2.3 for Matlab
  Per Jönsson and Lars Eklundh
  per.jonsson@lut.mah.se, lars.eklundh@nateko.lu.se
  September 2006

  Give name of sensor data file
>sensordata.txt
  Sensor data values in the range [min,max] are accepted. Data outside
  this range are assigned weight 0. Give min and max
>[0 250]
  Give name of mask data file: write NONE if mask data is unavailable
>NONE
  Time-series with amplitudes less than A are not processed
  A = 0 forces all time-series to be processed. Give A
>0
  Single spikes are detected by a comparison with median filtered values
  and with closest neighbors. If the distance is greater than S*ystd,
  where ystd is the standard deviation for data values, we have a spike.
  S = 2 is the normal value. Give S
>2
  Give parameter that is used to determine the number of annual seasons.
  Parameter value should be in the range [0,1]. A value close to 0
  will force the program to interpret a small depression in the main
  curve as a second annual season. A value close to 1 will force the
  program to always use only one annual season. Give season parameter
>1
  Fitted curves adapts to the upper envelope of the sensor data values
  in an iterative procedure. Give the number of fitting steps: 1, 2 or 3
>3
  Give the strength of the adaptation. Value should be in the
  range [1,10] where 2 is the normal value.
>2
  Specify the processing methods
  Savitzky-Golay (0/1), Asymmetric-Gauss (0/1), Double-Logistic (0/1)
>[1 1 1]
  Savitzky-Golay window sizes for each of the fitting steps
>[4 5 6]
```

```
  The time for the season start (end) is defined as the time for which the
  sensor data value ,measured from the base level, has increased (decreased)
  to X % of the seasonal amplitude. X = 20 is the normal value. Give X
>20
  Plot and print to screen (0/1), Debug (0/1)
>[1 0]
  Give an identification tag for the job (text string of max 20 chars)
>test1

  Phenology parameters SG, row 1,  season 1
  27.2 49.7 22.5 162.9 35.8 233.4 70.4 9.2 4.4 4988.9 1078.3
  Phenology parameters SG, row 1,  season 2
  59.0 86.3 27.3 164.6 72.6 227.2 62.5 4.4 4.0 6044.1 1105.0
  Phenology parameters SG, row 1,  season 3
  96.4 121.7 25.3 164.6 108.7 240.0 75.4 4.8 4.8 5577.1 1131.9
  Phenology parameters SG, row 1,  season 4
  131.0 157.6 26.6 164.2 144.9 231.0 66.8 4.5 4.8 5923.6 1161.9
  Phenology parameters AG, row 1,  season 1
  28.1 49.4 21.3 167.0 36.8 226.8 59.7 11.0 5.4 4787.0 945.3
  Phenology parameters AG, row 1,  season 2
  59.6 84.8 25.2 166.7 72.6 225.9 59.2 4.8 4.8 5527.1 1026.5
  Phenology parameters AG, row 1,  season 3
  97.2 119.8 22.6 170.2 108.5 235.1 64.9 5.7 5.6 5066.2 981.4
  Phenology parameters AG, row 1,  season 4
  132.9 157.2 24.3 169.8 144.7 229.7 60.0 5.0 4.7 5571.6 988.2
  Phenology parameters DL, row 1,  season 1
  28.1 49.4 21.3 166.9 36.7 227.3 60.4 11.2 5.3 4785.7 946.3
  Phenology parameters DL, row 1,  season 2
  59.5 85.1 25.6 166.3 72.5 227.4 61.0 4.8 4.6 5720.5 1062.8
  Phenology parameters DL, row 1,  season 3
  97.2 119.7 22.5 169.4 108.5 237.8 68.4 5.5 5.6 5056.7 991.7
  Phenology parameters DL, row 1,  season 4
  132.6 157.3 24.7 169.0 144.8 230.2 61.2 4.8 4.8 5574.9 1011.8

  pixel 1 1, has been processed

  Processing finished

  Phenological parameters written to:
  phenologySG_test1
  phenologyAG_test1
  phenologyDL_test1
  Sensor data and fitted functions written to:
  sensordata_test1
  fitSG_test1
  fitAG_test1
  fitDL_test1
```

```
  Input data written to:
  input_test1.txt
```

Since plots were requested for this run a number of Matlab figures displaying different stages of the processing are generated. These figures are valuable when testing different input parameters. Below two of these figures are shown.

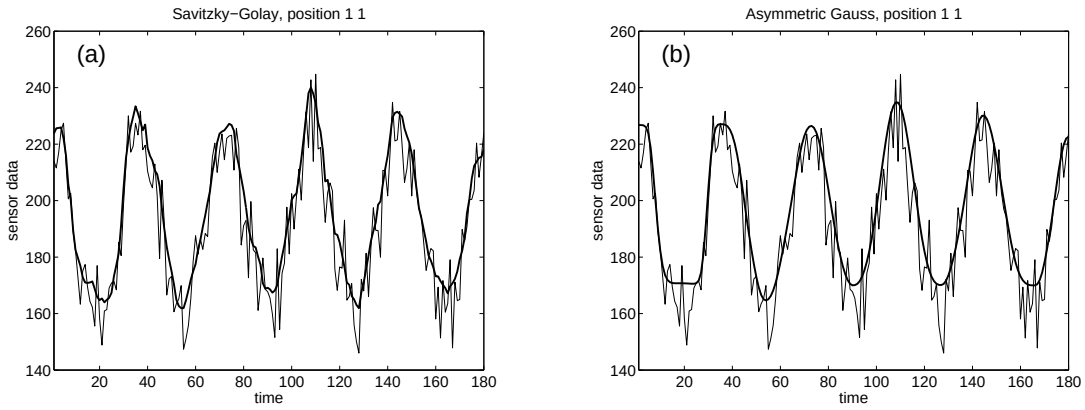

Figure 12: *Sensor data together with functions from Savitzky-Golay and from fits to asymmetric Gaussian functions.*

7.2 Timesat single time-series (Fortran under Windows)

The test file **sensordata.txt** with one time-series covering five years and with 36 data values a year resides in the directory `RUN`. To perform a test run open a DOS window and add the directory `TIMESAT_FORTRAN` to the search path. Move to the directory `RUN` and complete the following dialog

```
>timesatseries

  TIMESAT series version 2.3 for Fortran
  Per Jönsson and Lars Eklundh
  per.jonsson@lut.mah.se, lars.eklundh@nateko.lu.se
  September 2006

  Give name of sensor data file
>sensordata.txt
  Sensor data values in the range [min,max] are accepted. Data outside
  this range are assigned weight 0. Give min and max
>0,250
  Give name of mask data file: write NONE if mask data is unavailable
>NONE
  Time-series with amplitudes less than A are not processed
  A = 0 forces all time-series to be processed. Give A
>0
  Single spikes are detected by a comparison with median filtered values
  and with closest neighbors. If the distance is greater than S*ystd,
```

```
  where ystd is the standard deviation for data values, we have a spike.
  S = 2 is the normal value. Give S
>2
  Give parameter that is used to determine the number of annual seasons.
  Parameter value should be in the range [0,1]. A value close to 0
  will force the program to interpret a small depression in the main
  curve as a second annual season. A value close to 1 will force the
  program to always use only one annual season. Give season parameter
>1
  Fitted curves adapts to the upper envelope of the sensor data values
  in an iterative procedure. Give the number of fitting steps: 1, 2 or 3
>3
  Give the strength of the adaptation. Value should be in the
  range [1,10] where 2 is the normal value.
>2
  Specify the processing methods
  Savitzky-Golay (0/1), Asymmetric-Gauss (0/1), Double-Logistic (0/1)
>1,1,1
  Savitzky-Golay window sizes for each of the fitting steps
>4,5,6
  The time for the season start (end) is defined as the time for which the
  sensor data value ,measured from the base level, has increased (decreased)
  to X % of the seasonal amplitude. X = 20 is the normal value. Give X
>20
  Give an identification tag for the job (text string of max 20 chars)
>test2

  Processing finished

  Phenological parameters written to:
  phenologySG_test2
  phenologyAG_test2
  phenologyDL_test2
  Sensor data and fitted functions written to:
  sensordata_test2
  fitSG_test2
  fitAG_test2
  fitDL_test2
  Input data written to:
  input_test2.txt
```

Sensor data, fits and phenology parameters, that are written to the output files, can be displayed and manipulated using the programs in `TOOLS`. This will be further discussed in section 8.

7.3 Timesatimage West-Africa (Matlab)

The sensor image list **ndvilistwa.txt** giving name and path of 108 NDVI sensor images over West Africa spanning a period of three years resides in the directory `RUN`. The corresponding mask image list is named **clavrlistwa.txt**. The images, that are in byte format, contain 200×200 pixels. The mask data follows the classification described in section 3.2. To process the time-series for the pixel in row 100 and column 110, start Matlab and add the directory `TIMESAT_MATLAB` to the search path. Move to the directory `RUN` and complete the following dialog in the Matlab command window

```
>timesatimage

  TIMESAT image version 2.3 for MATLAB
  Per Jonsson and Lars Eklundh
  per.jonsson@lut.mah.se, lars.eklundh@nateko.lu.se
  September 2006

  Name of the sensor data file list
>ndvilistwa.txt
  Specify file type for image files
  1 = 8-bit unsigned integer (byte)
  2 = 16-bit signed integer
  3 = 32-bit real
  filetype
>1
  Number of rows and columns in the image tile
>[200 200]
  Define processing area: start row, stop row, start column, stop column
>[100 100 110 110]
  Number of years and number points per year in the time-series
>[3 36]
  Sensor data values in the range [min,max] are accepted. Data outside
  this range are assigned weight 0. Give min and max
>[140 250]
  Name of the mask data file list: write NONE if mask data is unavailable
>clavrlistwa.txt
  Sensor data values are assigned weights depending on the
  corresponding mask data values:
  if mask data in the range [a1,b1] then set weight to w1
  if mask data in the range [a2,b2] then set weight to w2
  if mask data in the range [a3,b3] then set weight to w3
  for mask data values outside these ranges weight is set to 0.
  Give a1,b1,w1,a2,b2,w2,a3,b3,w3
>[1 11 0 12 21 0.5 22 31 1]
  Time-series with amplitudes less than A are not processed
  A = 0 forces all time-series to be processed. Give A
>0
  Single spikes are detected by a comparison with median filtered values
```

```
  and with closest neighbors. If the distance is greater than S*ystd,
  where ystd is the standard deviation for data values, we have a spike.
  S = 2 is the normal value. Give S
>2
  Give parameter that is used to determine the number of annual seasons.
  Parameter value should be in the range [0,1]. A value close to 0
  will force the program to interpret a small depression in the main
  curve as a second annual season. A value close to 1 will force the
  program to always use only one annual season. Give season parameter
>1
  Fitted curves adapts to the upper envelope of the sensor data values
  in an iterative procedure. Give the number of fitting steps: 1, 2 or 3
>3
  Give the strength of the adaptation. Value should be in the
  range [1,10] where 2 is the normal value.
>2
  Specify the processing methods
  Savitzky-Golay (0/1), Asymmetric-Gauss (0/1), Double-Logistic (0/1)
>[1 1 1]
  Savitzky-Golay window sizes for each of the fitting steps
>[4 5 6]
  The time for the season start (end) is defined as the time for which the
  sensor data value ,measured from the base level, has increased (decreased)
  to X % of the seasonal amplitude. X = 20 is the normal value. Give X
>20
  Plot and print to screen (0/1), Debug (0/1)
>[1 0]
  Give an identification tag for the job (text string of max 20 chars)
>wa1

  Phenology parameters SG, row 100 col 110, season 1
  4.4 34.7 30.3 164.3 25.9 211.0 46.7 1.7 5.1 6148.1 891.3
  Phenology parameters SG, row 100 col 110, season 2
  44.9 77.1 32.2 158.3 58.0 210.2 51.9 3.6 2.8 6621.4 1080.9
  Phenology parameters AG, row 100 col 110, season 1
  11.2 34.0 22.8 172.2 24.5 208.4 36.1 2.8 5.5 4930.1 624.6
  Phenology parameters AG, row 100 col 110, season 2
  47.7 73.4 25.7 165.7 57.3 209.0 43.3 8.3 3.3 5470.0 831.4
  Phenology parameters DL, row 100 col 110, season 1
  10.8 33.9 23.0 172.2 24.9 208.8 36.6 2.5 5.9 4927.0 622.4
  Phenology parameters DL, row 100 col 110, season 2
  47.6 73.7 26.2 165.4 57.4 208.9 43.5 8.1 3.3 5474.9 844.2

  pixel 100 110, has been processed press return to continue

  Processing finished
```

```
Phenological parameters written to:
phenologySG_wa1
phenologyAG_wa1
phenologyDL_wa1
Sensor data and fitted functions written to:
sensordata_wa1
fitSG_wa1
fitAG_wa1
fitDL_wa1
Input data written to:
input_wa1.txt
```

Since plots were requested for this run a number of Matlab figures are generated. These figures are valuable when testing different input parameters. Some of these figures are shown below.

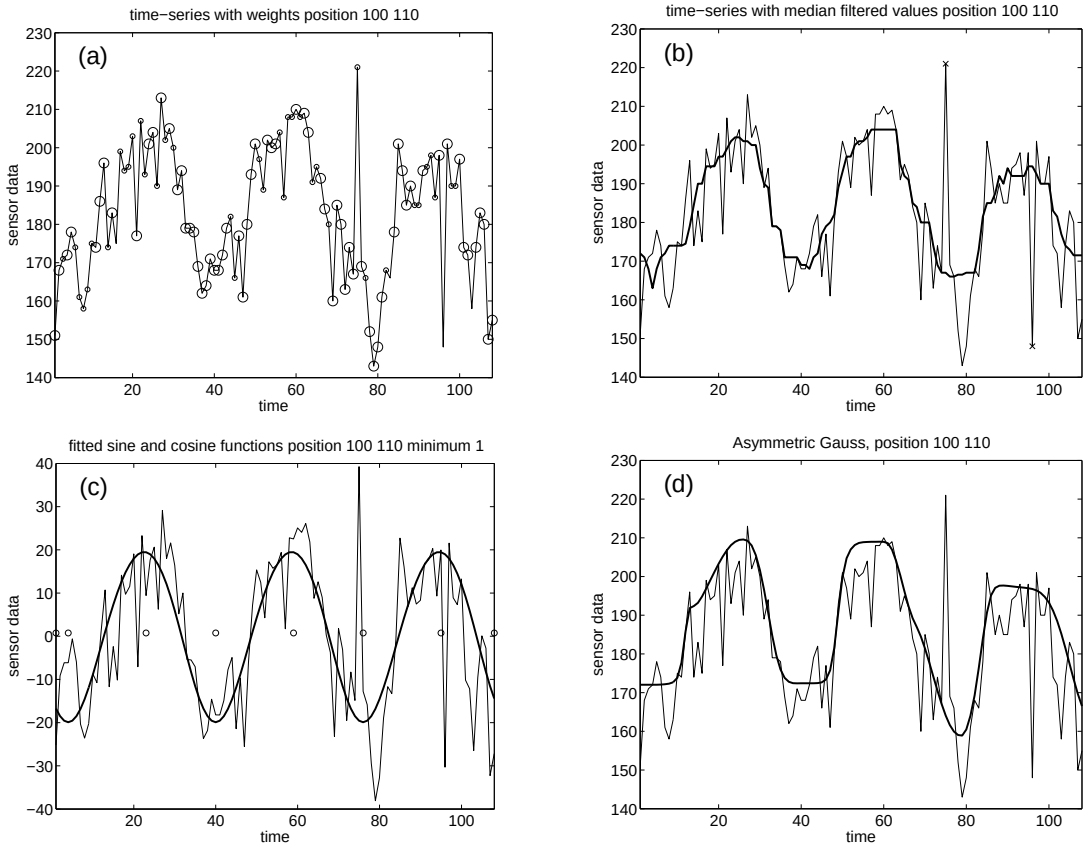

Figure 13: *(a) Time-series where the values have been assigned weights. Large circles indicate clear conditions ($w = 1$), small circles mixed conditions ($w = 0.5$), and no circle clouds ($w = 0$). (b) Time-series together with values from a median filtering. Values in the time-series that are sufficiently different from both the left- and right-hand neighbors and the median filtered value are classified as outliers and are assigned weight 0. Detected spikes and outliers are marked by crosses. (c) Fitted sine- and cosine functions. The circles define the intervals in which the local Gaussian and double logistic functions are fitted. (d) sensor data with function from fits to asymmetric Gaussians.*

7.4 Timesatimage West-Africa (Fortran under Windows)

The sensor image list **ndvilistwa.txt** giving name and path of 108 NDVI sensor images over West Africa spanning a period of three years resides in the directory `RUN`. The corresponding mask image list is named **clavrlistwa.txt**. The images, that are in byte format, contain 200×200 pixels. The mask data follows the classification described in section 3.2. To process the time-series for all pixels in the image open a DOS window and add the directory `TIMESAT_FORTRAN` to the search path. Move to the directory `RUN` and complete the following dialog

```
>timesatimage

  TIMESAT image version 2.3 for Fortran
  Per Jonsson and Lars Eklundh
  per.jonsson@lut.mah.se, lars.eklundh@nateko.lu.se
  September 2006

  Name of the sensor data file list
>ndvilistwa.txt
  Specify file type for image files
  1 = 8-bit unsigned integer (byte)
  2 = 16-bit signed integer
  3 = 32-bit real
  filetype
>1
  Numer of rows and columns in the image tile
>200,200
  Define processing window: start row, stop row, start column, stop column
>1,200,1,200
  Number of years and number points per year in the time-series
>3,36
  Sensor data values in the range [min,max] are accepted. Data outside
  this range are assigned weight 0. Give min and max
>100,255
  Name of the mask data file list: write NONE if mask data is unavailable
>clavrlistwa.txt
  Sensor data values are assigned weights depending on the
  corresponding mask data values:
  if mask data in the range [a1,b1] then set weight to w1
  if mask data in the range [a2,b2] then set weight to w2
  if mask data in the range [a3,b3] then set weight to w3
  for mask data values outside these ranges weight is set to 0.
  Give a1,b1,w1,a2,b2,w2,a3,b3,w3
>1,11,0,12,21,0.5,22,31,1
  Time-series with amplitudes less than A are not processed
  A = 0 forces all time-series to be processed. Give A
>0
  Single spikes are detected by a comparison with median filtered values
```

```
  and with closest neighbors. If the distance is greater than S*ystd,
  where ystd is the standard deviation for data values, we have a spike.
  S = 2 is the normal value. Give S
>2
  Give parameter that is used to determine the number of annual seasons.
  Parameter value should be in the range [0,1]. A value close to 0
  will force the program to interpret a small depression in the main
  curve as a second annual season. A value close to 1 will force the
  program to always use only one annual season. Give season parameter
>0.5
  Fitted curves adapts to the upper envelope of the sensor data
  values in an iterative procedure. Give the number of fitting steps: 1, 2 or 3
>3
  Give the strength of the adaptation. Value should be in the
  range [1,10] where 2 is the normal value.
>2
  Specify the processing methods
  Savitzky-Golay (0/1), Asymmetric-Gauss (0/1), Double-Logistic (0/1)
>1,1,1
  Savitzky-Golay window sizes for each of the fitting steps
>4,5,6
  The time for the season start is defined as the time for which the
  sensor data value ,measured from the base level, has risen to X %
  of the seasonal amplitude. X = 20 is the normal value. Give X
>20
  Give an identification tag for the job (text string of max 20 chars)
>wa2

  Row 1
  Row 2
   ...
  Row 200

  Processing finished

  Phenological parameters written to:
  phenologySG_wa2
  phenologyAG_wa2
  phenologyDL_wa2
  Sensor data and fitted functions written to:
  sensordata_wa2
  fitSG_wa2
  fitAG_wa2
  fitDL_wa2
  Input data written to:
  input_wa2.txt
```

Sensor data, fits and phenology parameters, written to the output files, can be extracted and displayed using the programs in TOOLS (see section 8).

7.5 Timesatimage Scania (Matlab)

The sensor image list **modisscania.txt** giving name and path of 69 MODIS sensor images over Scania in Southern Sweden spanning a period of three years resides in the directory RUN. The images, that are in 16-bit signed integer format, contain 600×600 pixels. To process the time-series for the pixel in row 100 and column 110 start Matlab and add the directory TIMESAT_MATLAB to the search path. Move to the directory RUN and complete the following dialog in the Matlab command window

```
>timesatimage

  TIMESAT image version 2.3 for Matlab
  Per Jonsson and Lars Eklundh
  per.jonsson@lut.mah.se, lars.eklundh@nateko.lu.se
  September 2006

  Name of the sensor data file list
>modisscania.txt
  Specify file type for image files
  1 = 8-bit unsigned integer (byte)
  2 = 16-bit signed integer
  3 = 32-bit real
  filetype
>2
  Numer of rows and columns in the image tile
>[600 600]
  Define processing window: start row, stop row, start column, stop column
>[100 100 100 100]
  Number of years and number points per year in the time-series
>[3 23]
  Sensor data values in the range [min,max] are accepted. Data outside
  this range are assigned weight 0. Give min and max
>[-10000 10000]
  Name of the mask data file list: write NONE if mask data is unavailable
none
  Time-series with amplitudes less than A are not processed
  A = 0 forces all time-series to be processed. Give A
>0
  Single spikes are detected by a comparison with median filtered values
  and with closest neighbors. If the distance is greater than S*ystd,
  where ystd is the standard deviation for data values, we have a spike.
  S = 2 is the normal value. Give S
>1.5
  Give parameter that is used to determine the number of annual seasons.
  Parameter value should be in the range [0,1]. A value close to 0
```

```
  will force the program to interpret a small depression in the main
  curve as a second annual season. A value close to 1 will force the
  program to always use only one annual season. Give season parameter
>1
  Fitted curves adapts to the upper envelope of the sensor data
  values in an iterative procedure. Give the number of fitting steps: 1, 2 or 3
>3
  Give the strength of the adaptation. Value should be in the
  range [1,10] where 2 is the normal value.
>2
  Specify the processing methods
  Savitzky-Golay (0/1), Asymmetric-Gauss (0/1), Double-Logistic (0/1)
>[1 1 1]
  Savitzky-Golay window sizes for each of the fitting steps
>[4 5 6]
  The time for the season start is defined as the time for which the
  sensor data value ,measured from the base level, has risen to X %
  of the seasonal amplitude. X = 20 is the normal value. Give X
>20
  Plot and print to screen (0/1), Debug (0/1)
>[1 0]
  Give an identification tag for the job (text string of max 20 chars)
>scania

  Phenology parameters SG, row 100 col 100, season 1
  5.2 23.0 17.8 4321.0 12.6 9380.1 5059.1 702.7 401.4 141896.8 59798.0
  Phenology parameters SG, row 100 col 100, season 2
  30.4 44.5 14.1 3868.3 37.8 9108.4 5240.1 623.7 886.6 114980.0 53086.6
  Phenology parameters AG, row 100 col 100, season 1
  7.0 22.1 15.1 4940.9 13.4 8824.4 3883.5 1462.0 614.7 136928.3 47992.0
  Phenology parameters AG, row 100 col 100, season 2
  31.1 44.1 12.9 4712.5 37.9 8967.3 4254.8 699.9 922.7 111821.5 41133.4
  Phenology parameters DL, row 100 col 100, season 1
  7.0 21.8 14.9 5030.5 13.4 8847.8 3817.3 1297.3 650.0 131732.8 46213.4
  Phenology parameters DL, row 100 col 100, season 2
  31.2 43.9 12.7 4650.8 38.1 9072.8 4422.0 684.1 1067.9 107195.2 42084.5

  pixel 100 100, has been processed press return to continue

  Processing finished

  Phenological parameters written to:
  phenologySG_scania
  phenologyAG_scania
  phenologyDL_scania
  Sensor data and fitted functions written to:
  sensordata_scania
```

```
fitSG_scania
fitAG_scania
fitDL_scania
Input data written to:
input_scania.txt
```

Since plots were requested for this run a number of Matlab figures displaying different stages of the processing are generated. Some of these figures are shown below.

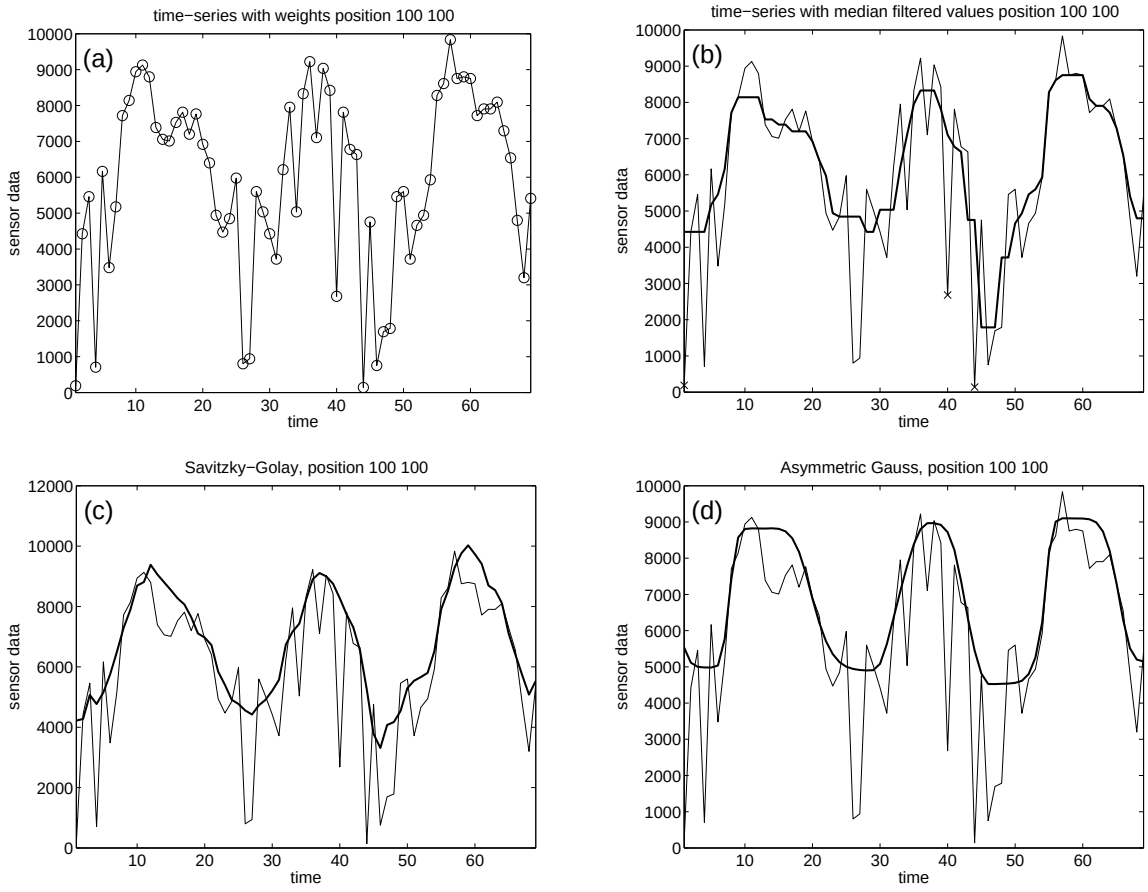

Figure 14: *(a) Time-series with weights. Since no ancillary data are available all weights are set to the constant value one. (b) Time-series together with values from a median filtering. Values in the time-series that are sufficiently different from both the left- and right-hand neighbors and the median filtered value are classified as outliers and are assigned weight 0. Detected spikes and outliers are marked by crosses. (c-d) sensor data with functions from Savitzky-Golay and fits to asymmetric Gaussians.*

8 Manipulating the output files

The output files are of main importance. These files can be manipulated in various ways. In the directory `TOOLS` there are Matlab and Fortran programs that read the output files and displays sensor data and fitted functions. There are also programs that reads the phenology

output files and produces images over various parameters. The different programs and their function are summarized in the table.

Name	Function	Interface
viewfits.m	view sensor data and fitted functions	Matlab
printphenology.m	read and print phenological parameters	Matlab
phen2img.exe	generate phenological images	Fortran
func2img.exe	generate images from fitted functions	Fortran
viewimage.m	view binary images	Matlab
func2time.exe	generate time-series from fitted functions	Fortran
readheader.exe	read data header	Fortran

8.1 View sensor data and fitted functions

Sensor data and the fitted functions stored in the output files can be displayed using the Matlab program **viewfits** that resides in the `TOOLS` directory. As a test we will read the output files from the run over West-Africa in section 7.4 and display the fitted double logistic functions in row 100, columns 100-101. Start Matlab and add the directory `TOOLS` to the search path and change the directory to `RUN`. Complete the following dialog in the Matlab command window

```
>viewfits

  This file reads the output functions from the TIMESAT program
  and displays sensor data together with fitted functions

  Read fitted function from SG (0/1), AG (0/1), or DL (0/1) ?
>[0 0 1]
  Give job name
>wa2
  View fits between row1, row2, col1 and col2
>[100 100 100 101]

  row 1
  row 2
  ...
  row 100
```

Sensor data and fitted double logistic functions are displayed in figure 15.

8.2 Read and print phenological parameters

Phenology data can be read and printed using the Matlab program **printphenology** in the `TOOLS` directory. To print the phenology data from the fitted double logistic functions in row 100, columns 100-101 complete the following dialog in the Matlab command window

```
>printphenology
```

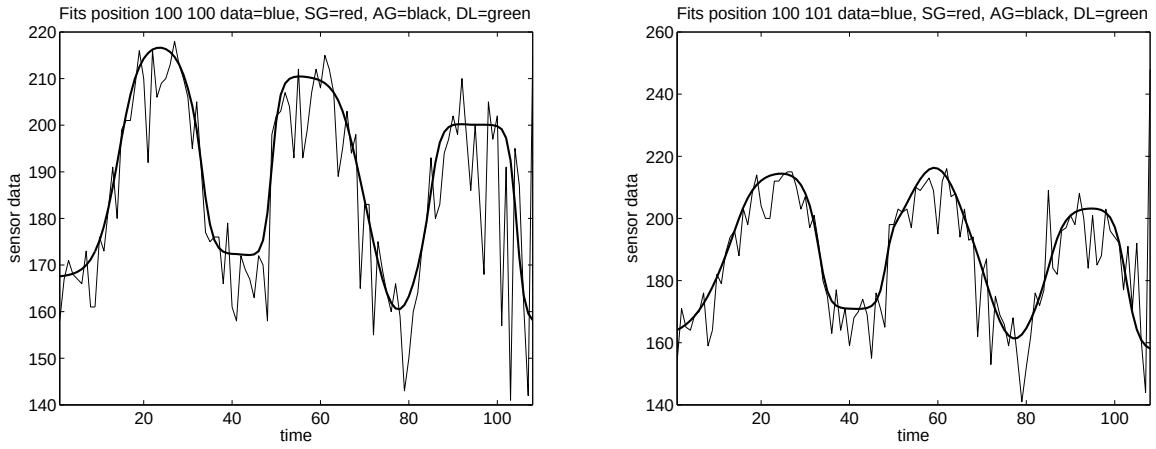

Figure 15: *Sensor data together with fits to double logistic functions.*

```
  This file reads the output functions from the TIMESAT program
  and prints phenology data

  Read phenology data from SG (1), AG (2), or DL (3) ?
>3
  Give job name
>wa2
  Ouput parameters for pixels between row1, row2, col1 and col2
>[100 100 100 101]

  row 1
  row 2
  ...
  row 100

  Pixel 100 100
  Phenology for season 1
  10.7 34.6 23.9 169.9 23.6 216.6 46.8 4.6 5.8 5258.7 842.4
  Phenology for season 2
  47.8 73.1 25.3 166.3 58.1 210.4 44.1 8.9 4.2 5530.7 873.3

  Pixel 100 101
  Phenology for season 1
  7.6 34.5 26.9 167.5 23.4 214.4 46.9 3.5 6.7 5790.9 934.1
  Phenology for season 2
  47.5 72.6 25.1 166.1 59.2 216.3 50.2 4.3 4.2 5385.8 900.3
```

8.3 Generate phenological images

Using the Fortran program **phen2img** phenology data can be converted to images. To generate an image over the season start in West-Africa during the time period from 30 to 80 from the fitted asymmetric Gaussian functions in the testrun in section 7.4; open a DOS window, add the directory `TOOLS` to the path, and give the following commands

```
>phen2img

  Program phen2img
  Program for generating images from TIMESAT phenology files

  Name of input phenology file
>phenologyAG_wa2

  Start of season ___________________1
  End of season _____________________2
  Length of season __________________3
  Base value ________________________4
  Position of middle of season _____5
  Maximum of fitted data ____________6
  Amplitude _________________________7
  Left derivative ___________________8
  Right derivative __________________9
  Large integral ___________________10
  Small integral ___________________11

  Seasonal parameter to output
>1
  Only seasons fully between given dates will be processed.
  Min and max dates for season to process (e.g. 1, 36).
>30,80
  Code for missing season within min and max dates.
>0
  Code for missing phenology data for pixel (could be the same)
>0
  Name of output files (no extension!)
>startwa
  Specify the file type for the output image files (2 or 3)
  2 = 16-bit signed integer (phenology values will be rounded to nearest integer)
  3 = 32-bit real
>3
  Opening startwa_s1
  Opening startwa_s2
  Opening startwa_nseas
  Opening startwa_warnings.txt

  Seasonality data written
```

```
  Number of seasons written
  File format 32-bit real
```

There can be one or more seasons in the specified time span. If no complete season is found within the time span the code for missing season will be written to the corresponding pixel in the image. If no phenology information at all is found for the pixel (e.g. over oceans or where there are too many missing values) the code for missing phenology data will be written to the pixel in the image. The start of the first season is written to the image file **startwa_s1** whereas the start of the second is written to **startwa_s2**. The image files can be displayed using IDRISI, PCI or any other imaging processing software capable of handling flat binary files. Here we will use the Matlab program **viewimage**. Go to the Matlab command window and complete the dialog

```
>viewimage
  Input image file name:
>startwa_s1
  Number of rows
>200
  Number of columns
>200
  File type (uint8, int16 or float32)
>float32
  View histogram to determine color scaling, hit a key to continue
>
  Image scaling interval [lower upper]
>[40 60]
```

The obtained image is displayed in Fig 16. The best color scaling interval can be inferred from an image histogram.

8.4 Generate images from fitted functions

Using the Fortran program func2img, fitted function data can be converted to images. To generate an image over the fitted function value in West-Africa from the fitted asymmetric Gaussian functions in the testrun in section 7.4 do as follows; open a DOS window, add the `tools` directory to the path, and give the following commands

```
>func2img

  Program func2img
  Program for generating images from TIMESAT fitted
  function files

  Name of input fitted function file
>fitAG_wa2
  Code for missing function value for pixel
>0
  Name of output files (no extension!)
>functionwa
```

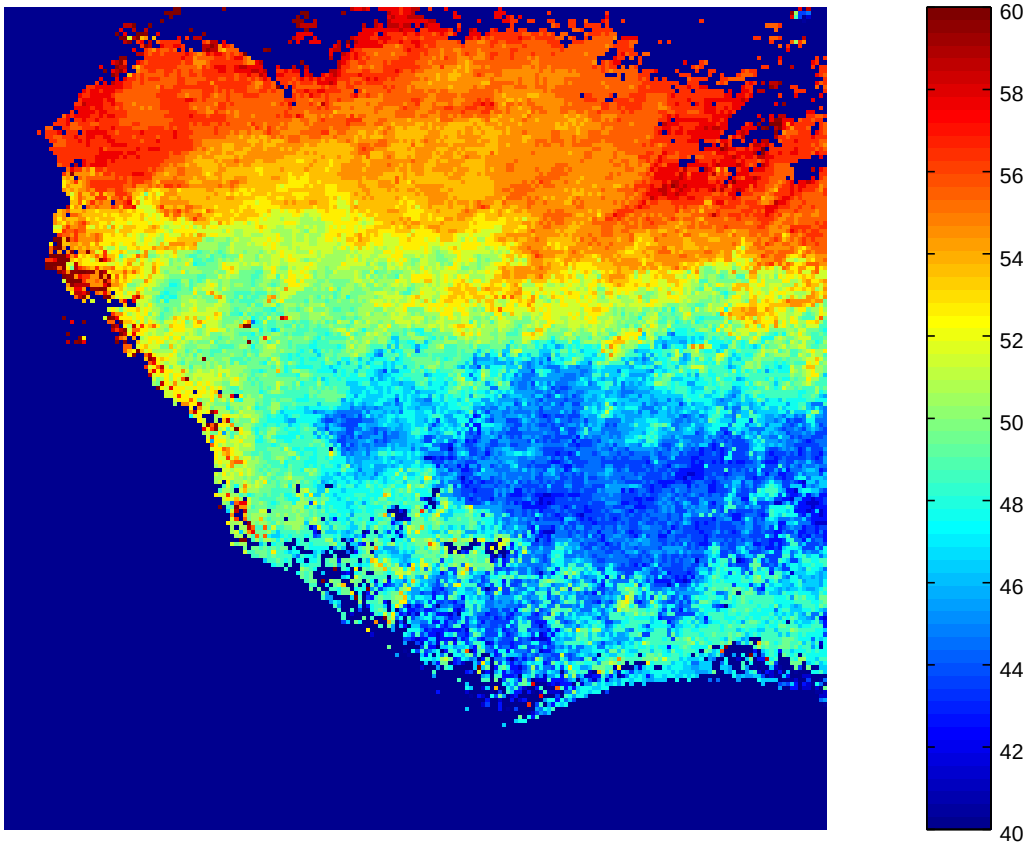

Figure 16: *Phenological image over the season start in West-Africa from fitted asymmetric Gaussian functions. The overall appearance is in accordance with climate conditions governed by the inter tropical convergence zone (ITCZ).*

```
  Specify the file type for the output image files (1,2 or 3)
  1 = 8-bit binary
  2 = 16-bit signed integer
  3 = 32-bit real
>3
  Number of years:    3,  Number of points per year:   36
  Image window (rows, cols):    1 -  200 ;    1 -  200

  Image number to map (-1 = all)
>12
  functionwa_0012 opened
  Seasonality data written
  File format 32-bit real
```

In the example, the fitted function values for date 12 are written to an image file. Missing data are given the value 0 and the file format is 32-bit real. If format 1 (8-bit binary)

is selected the user needs to make sure that the values fit within the range 0–255. If -1 is specified at `Image number to map`, function values for all dates will be written to files **functionwa_0001** to **functionwa_0108** where $108 = 3 \times 36$ is the last date in the time-series. Images can be displayed with IDRISI, PCI or any equivalent software, or with the Matlab program **viewimage** (see 8.3).

8.5 Generate time-series from fitted functions

ASCII files of time-series can be generated from fitted functions, or from sensordata, for a specific pixel location. The program **func2time** is run in a DOS window, and it is assumeed that the directory `tools` is added to the path:

```
>func2time
 ------------------------------------------------------------------------
  Program func2time
  Program for generating ASCII time-series from TIMESAT fitted function files

 ------------------------------------------------------------------------

  Name of input fitted function file:
>fitAG_wa2

  Row, column (give same number twice if referring
  to single time-series data):
>100,130
  Name of output file:
>fitAG_wa2_ASCII.txt

  No. of years:     3,  No. of points per year:    36

  Row   100 col   130 written.
```

8.6 Read data header

The program **readheader** is used for investigating header information of binary TIMESAT output files containing fitted functions, phenology data, or sensor data. The program is run in a DOS window, and the example assumes that the directory `tools` is added to the path:

```
>readheader
 ------------------------------------------------------------------
  Program readheader
  Program for reading the header of a TIMESAT binary output data file

 ------------------------------------------------------------------

  Name of input file:
>phenologyDL_wa2
```

```
No. of years:                  3
No. of points per year:       36
Total dates:                 108
No. of rows:                 200
No. of columns:              200
Image window (rows; cols):     1 -   200 ;     1 -   200
```

Acknowledgements

Support from the Crafoord and Magn. Bergwall foundations is gratefully acknowledged. Salary for Lars Eklundh was partly funded by the Swedish National Space Board.

References

P. Jönsson and L. Eklundh
Seasonality extraction and noise removal by function fitting to time-series of satellite sensor data
IEEE transactions of geoscience and remote sensing, vol 40, No 8, 1824 (2002).

P. Jönsson and L. Eklundh
Seasonality extraction from time-series of satellite sensor data
Frontiers of Remote Sensing Information Processing, World Scientific, pages 487-500, edited by C.H. Chen, (2003).

P. Jönsson and L. Eklundh
TIMESAT - a program for analyzing time-series of satellite sensor data
Computers and Geosciences **30**, 833 (2004).

J. Verbesselt, P. Jönsson, S. Lhermitte, J. van Aardt, and P. Coppin
Evaluating indices derived from satellite and climate data as fire risk indicators in savanna ecosystems
IEEE transactions of Geoscience and Remote Sensing Vol. 44, No. 6, 1622 (2006).

K. Madsen, H.B. Nielsen and O. Tingleff
Methods for non-linear least squares problems
Informatics and Mathematical Modeling (IMM), Technical University of Denmark (2004).

H.B. Nielsen
Damping parameter in Marquardt's method
Technical Report IMM-REP-1999-05
Informatics and Mathematical Modeling (IMM), Technical University of Denmark (1999).

H.B. Nielsen
Separable nonlinear least squares
Technical Report IMM-REP-2000-01
Informatics and Mathematical Modeling (IMM), Technical University of Denmark (2000).

C. Kanzow, N. Yamashita and M. Fukushima
Levenberg-Marquardt methods for constrained nonlinear equations with strong local convergence properties (2002).