Single Calculus Chain Documentation

Release 0.2

SCC team

March 30, 2012

CONTENTS

1 Introduction **3**

2 Tutorial **5**
- 2.1 Introduction . . . 5
- 2.2 Adding a station . . . 5
- 2.3 Adding a system . . . 6
- 2.4 Adding equipment . . . 6
- 2.5 Adding a channel . . . 7
- 2.6 Adding products . . . 8
- 2.7 Processing data . . . 8

3 Detailed documentation **9**
- 3.1 Introduction . . . 9
- 3.2 Adding stations . . . 9
- 3.3 Adding systems . . . 9
- 3.4 Adding channels . . . 9
- 3.5 Adding products . . . 9
- 3.6 Adding other equipment . . . 9
- 3.7 The Handbook of instruments . . . 9
- 3.8 Uploading measuremnents . . . 9
- 3.9 View processing results . . . 10

4 The SCC netCDF file format **11**
- 4.1 Rationale . . . 11
- 4.2 Example . . . 12
- 4.3 Example of file (CDL format) . . . 20

5 User management **27**
- 5.1 Acount types . . . 27
- 5.2 Requesting a new account . . . 27
- 5.3 User account security . . . 27

6 Indices and tables **29**

Contents:

CHAPTER

ONE

INTRODUCTION

- The Single calculus chain is made of different modules. These modules don't interact directly but only change value in a database.
- This interface will allow Earlinet memebers to interact with parts of the database.
- One part of the interface (the "Station admin" section) permits registering a new station, registerring new lidar systems and configuration, fill in details for the channels that constitute the system and finnaly define the products (extinction, backscatter e.t.c.) that need to be calculated by the SCC.
- The second part of the interface is dedicated to the uploading of new measurement files, the configuration of the measurement specific paramters and, finally, the retrieval of the calculated products.
- Different types of users, with different level of access permissions can have access in the interface. In this way, higher level of flexibility and security can be achived.

CHAPTER

TWO

TUTORIAL

2.1 Introduction

- The Single calculus chain is made of different modules. These modules don't interact directly but only change value in a database.
- This interface will allow Earlinet memebers to interact with parts of the database.
- One part of the interface (the "Station admin" section) permits registering a new station, registerring new lidar systems and configuration, fill in details for the channels that constitute the system and finnaly define the products (extinction, backscatter e.t.c.) that need to be calculated by the SCC.
- The second part of the interface is dedicated to the uploading of new measurement files, the configuration of the measurement specific paramters and, finally, the retrieval of the calculated products.
- Different types of users, with different level of access permissions can have access in the interface. In this way, higher level of flexibility and security can be achived.

2.2 Adding a station

You can change all your settings throught the *admin section* of the website. To reach it, click on **Station admin** link at the main menu of the site.

Note: You will need to have an account with admin access privilages to access this part of the site. See *User management* for details.

The first you have to do to start using the single calculus chain is to register your station. To do this, go the the admin section and click on the **HOI stations** in the *System settings* panel. This will take you to a page with a list of all stations that you account has access to. This list should be empty if this is the first time you add a station.

To add a new station to the database click on **Add HOI station** at the top right of the screen. This will take you to a new page were you can fill in the needed information. The fields in **bold** are mandatory and you will need to fill them before you can save you new station.

For now you will need to fill in the following fields:

Name The name of the station

Id The earlinet call sign with exactly 2 characters.

Institute name The name of the institute that own the system

Latitude In degrees north is the latitude of the station.

Longitude In degrees east is the longitude of the station.

Height asl The altitude of the station in meters above sea level.

PI The name of the Principal Investigator of the station.

You can leave all the other fields empty. When you are done, press the **save** button at the bottom right of the page. This will take you back to the list of your stations. If everything went OK your station you just added should appear in the list.

We don't have to make any more changes in this part, so you can click on **Home** on the top left of the page to return to the starting page of the *admin section*.

2.3 Adding a system

After adding a station to the database, we need to add a new system. To do this, click on the **HOI systems** in the *System settings* panel. This will take you to a page with a list of all availabe systems that are connected with your stations. This list should be empty if this is the first time you add a system.

Note: In the Single Calculus Chain, a *HOI System* represents a specific configuration of a lidar system. For example, if you are operating a lidar system and you use different channels during daytime and nightime, you will need to register *two different* systems in the database, one for each different configuration you use.

To add a new system to the database click on **Add HOI system** at the top right of the screen. This will take you to a new page were you can fill in the needed information. As before, the fields in **bold** are mandatory and you will need to fill them before you can save you new system.

Note: Not every field that is present in the database is used in the in the Single Calculs Chain. Many of them are part of the Handbook of Instruments.

For now you will need to fill in the following fields:

Name The name of your system.

Station (owner) From the drop-down list, select the station which this system belongs to.

Configuration The name of the specific configuration. For example you could specify "night time" if the system you are registering correspond to the night-time configuration of your system

Pi The principle investigator of this system

Height asl The altitude of the system above sea level (in meters).

You can leave all the other fields empty. When you are done, press the **save** button at the bottom right of the page. This will take you back to the list of your systems. If everything went OK your new system you just added should appear in the list.

We don't have to make any more changes in this part, so you can click on **Home** on the top left of the page to return to the starting page of the *admin section*.

2.4 Adding equipment

After adding a system to the database, we need to add at least one telescope and one laser before you add a channel.

2.4.1 Telescope

To add a new telescope, click on the **HOI telescopes** in the *System settings* panel. This will take you to a page with a list of all availabe telescopes that are connected with your station. This list should be empty if this is the first time you add a telescope.

To add a new telescope to the database click on **Add HOI telescope** at the top right of the screen. This will take you to a new page were you can fill in the needed information. Once again, the fields in **bold** are mandatory and you will need to fill them before you can save you new telescope.

The fields you need to add are:

Type The telescope type

Diameter The diameter of the primary mirror in mm

Focal length The equivalent focal length of the telescope in mm

Full overlap height The height where the full overlap is achieved.

When you are done, press the **save** button at the bottom right of the page. If no errors are present, you will return to the telescope list page. Your new telescope should appear in the list.

2.4.2 Laser

To add a new laser, click on the **HOI Laser** in the *System settings* panel. This will take you to a page with a list of all availabe lasers that are connected with your station. This list should be empty if this is the first time you add a laser.

To add a new telescope to the database click on **Add HOI laser** at the top right of the screen. This will take you to a new page were you can fill in the needed information. The fields in **bold** are mandatory and you will need to fill them before you can save you new telescope.

The fields you need to add are:

Manufacturer The manufacturer of the telescope

Model The model of the telescope

Repetition rate The repetition rate in Hz

Type The type of the laser (ex. Nd:YAG)

When you are done, press the **save** button at the bottom right of the page. If no errors are present, you will return to the laser list page. The new laser you added should be present there.

2.5 Adding a channel

After adding a system, a telescope and a laser to the database, you need to add a new channel. To do this, click on the **HOI channels** in the *System settings* panel. This will take you to a page with a list of all availabe channels that are connected with your lidar systems. This list should be empty if this is the first time you add a system.

To add a new channel to the database click on **Add HOI channel** at the top right of the screen. This will take you to a new page were you can fill in the needed information. As before, the fields in **bold** are mandatory and you will need to fill them before you can save you new system.

The fileds you have to fill here are more, as many of these are used during the processing of measurements.

Warning: There is a last step, different from the previous cases, when saving a new channel. You need to connect your channel with a lidar system before you save, or else all your entry will be lost. Read carefully through this document (or directly the *last section*) to avoid any problems.

2.5.1 Fill in the fields

To start using the single calculus chain you will need to fill the following fields:

Name The name of the channel ex. "355", "1064 analog" etc.

Telescope The telescope that is used for this channel

Laser The laser that is used for this channel

Interference filter center The center of the interference filter in nm

Interference filter FWHM The FWHM of the interference filter in nm

Emission wavelength The emission wavelength of the laser used for this channel

Field of view The field of view realated to this channel in mrad

Raw range resolution The raw range resolution of the measured data in m

Dead time The dead of the detector in ns. You should fill in this in case of a photon counting detector.

Trigger delay The trigger delay value for the channel in ns. Fill in 0 if not needed.

Scattering mechanism The scattering mechanism that is involved in this channel. Select the appropriate value from the drop-down list.

Dead time correction type The dead time correction type to be applied. Select *Not defined* if none needs to be defined.

Background mode The way to calculate the singal background. Select *Not defined* if none needs to be defined.

Signal type The type of the singal that is measured, ex. "elT" for total elastic, "vrRN2" for vibrational-rotational Raman signal from Nitrogen melecules etc. See *Signal types* for details.

Detection mode The detection mode of this channel.

2.5.2 Connecting to a system

Before you finish, you need to attach your channel to one of your systems. To do this, go at the bottom of the page and select your system from the drop-down list in the **System channels** area.

When you are done, press the **save** button at the bottom right of the page. This will take you back to the list of your channels. If everything went OK your new channel you just added should appear in the list.

2.6 Adding products

Walk-through of adding a product.

2.7 Processing data

Walk-through of how to upload a file and seeing the results.

CHAPTER

THREE

DETAILED DOCUMENTATION

Contents:

3.1 Introduction

3.2 Adding stations

You can add the definition of new systems that belong to the station by clicking on the Hoi System blue line that appears bollow the main station fields. For more details on the filed you need to fill in see the *Adding systems* section. You can add more stations by clicking on the "Add another Hoi System" option.

Note: You need to have *Javascript* enebled to add a new station from this page.

3.3 Adding systems

3.4 Adding channels

3.4.1 Signal types

(explain here all the signal type abreviations).

3.5 Adding products

3.6 Adding other equipment

3.7 The Handbook of instruments

3.8 Uploading measuremnents

123

3.9 View processing results

123

CHAPTER

FOUR

THE SCC NETCDF FILE FORMAT

4.1 Rationale

The Single Calculus Chain (SCC) is composed by two different modules:

- pre-processing module (scc_preprocessing)
- optical processing module (ELDA)

To perfom aerosol optical retrievals the SCC needs not only the raw lidar data but also a certain number of parameters to use in both pre-processing and optical processing stages. The SCC gets these parameters looking at two different locations:

- Single Calculus Chain relational database (SCC_DB)
- Input files

There are some paramenters that can be found only in the input files (those ones changing from measurement to measurement), others that can be found only in the SCC_DB and other ones that can be found in both these locations. In the last case, if a particular parameter is needed, the SCC will search first in the input files and then in SCC_DB. If the parameter is found in the input files the SCC will keep it without looking into SCC_DB.

The input files have to be submitted to the SCC in NetCDF format. At the present the SCC can handle four different types of input files:

1. Raw Lidar Data
2. Sounding Data
3. Overlap
4. Lidar Ratio

As already mentioned, the Raw Lidar Data file contains not only the raw lidar data but also other parameters to use to perform the pre-processing and optical processing. The Sounding Data file contains the data coming from a correlative radiosounding and it is used by the SCC for molecular density calculation. The Overlap file contains the measured overlap function. The Lidar Ratio file contains a lidar ratio profile to use in elastic backscatter retrievals. The Raw Lidar Data file is of course mandatory and the Sounding Data, Overlap and Lidar Ratio files are optional. If Sounding Data file is not submitted by the user, the molecular density will be calculated by the SCC using the "US Standard Atmosphere 1976". If the Overlap file is not submitted by the user, the SCC will get the full overlap height from SCC_DB and it will produce optical results starting from this height. If Lidar Ratio file is not submitted by the user, the SCC will consider a fixed value for lidar ratio got from SCC_DB.

The user can decide to submit all these files or any number of them (of course the file Raw Lidar Data is mandatory). For example the user can submit together with the Raw Lidar Data file only the Sounding Data file or only the Overlap file.

This document provides a detailed explanation about the structure of the NetCDF input files to use for SCC data submission. All Earlinet groups should read it carefully because they have to produce such kind of input files if they want to use the SCC for their standard lidar retrievals. Every comments or suggestions regarding this document can be sent to Giuseppe D'Amico by e-mail at `damico@imaa.cnr.it`

This document is available for downloading at `www.earlinetasos.org`

In table tab:rawdata is reported a list of dimensions, variables and global attributes that can be used in the NetCDF Raw Lidar Data input file. For each of them it is indicated:

- The name. For the multidimensional variables also the corresponding dimensions are reported
- A description explaining the meaning
- The type
- If it is mandatory or optional

As already mentioned, the SCC can get some parameters looking first in the Raw Lidar Data input file and then into SCC_DB. This means that to use the parameters stored in SCC_DB the optional variables or optional global attributes must not appear within Raw Lidar Data file. This is the suggested and recommended way to use the SCC. Please include optional parameters in the Raw Lidar Data only as an exception.

In table tab:sounding, tab:overlap and tab:lr are reported all the information about the structure of Sounding Data, Overlap and Lidar Ratio input files respectively.

4.2 Example

Let's now consider an example of Raw Lidar Data input file. Suppose we want to generate NetCDF input file corresponding to a measurement with the following properties:

Start Date	30^{th} January 2009
Start Time UT	00:00:01
Stop Time UT	00:05:01
Station Name	Dummy station
Earlinet call-sign	cc
Pointing angle	5 degrees with respect to the zenith

Moreover suppose that this measurement is composed by the following lidar channels:

1. 1064 lidar channel

Emission wavelength=1064nm	Detection wavelength=1064nm
Time resolution=30s	Number of laser shots=1500
Number of bins=3000	Detection mode=analog
Range resolution=7.5m	Polarization state=total

2. 532 cross lidar channel

Emission wavelength=532nm	Detection wavelength=532nm
Time resolution=60s	Number of laser shots=3000
Number of bins=5000	Detection mode=photoncounting
Range resolution=15m	Polarization state=cross

3. 532 parallel lidar channel

Emission wavelength=532nm	Detection wavelength=532nm
Time resolution=60s	Number of laser shots=3000
Number of bins=5000	Detection mode=photoncounting
Range resolution=15m	Polarization state=parallel

4. 607 N_2 vibrational Raman channel

Emission wavelength=532nm	Detection wavelength=607nm
Time resolution=60s	Number of laser shots=3000
Number of bins=5000	Detection mode=photoncounting
Range resolution=15m	

Finally let's assume we have also performed dark measurements before the lidar measurements from the 23:50:01 UT up to 23:53:01 UT of 29:math:^*mathrmth* January 2009.

4.2.1 Dimensions

Looking at table tab:rawdata we have to fix the following dimensions:

```
points
channels
time
nb_of_time_scales
scan_angles
time_bck
```

The dimension `time` is unlimited so we don't have to fix it.

We have 4 lidar channels so:

```
channels=4
```

Regarding the dimension `points` we have only one channel with a number of vertical bins equal to 3000 (the 1064nm) and all other channels with 5000 vertical bins. In cases like this the dimension `points` has to be fixed to the maximum number of vertical bins so:

```
points=5000
```

Moreover only one channel (1064nm) is acquired with a time resolution of 30 seconds, all the other channels have a time resolution of 60 seconds. This means that we have to define two different time scales. We have to set:

```
nb_of_time_scales=2
```

The measurement is performed only at one scan angle (5 degrees with respect to the zenith) so:

```
scan_angles=1
```

We have 3 minutes of dark measurements and two different time scales one with 60 seconds time resolution and the other one with 30 seconds time resolution. So we will have 3 different dark profiles for the channels acquired with the first time scale and 6 for the lidar channels acquired with the second time scale. We have to fix the dimension `time_bck` as the maximum between these values:

```
time_bck=6
```

4.2.2 Variables

In this section it will be explained how to fill all the possible variables either mandatory or optional of Raw Lidar Data input file.

Raw_Data_Start_Time(time, nb_of_time_scales) This 2 dimensional mandatory array has to contain the acquisition start time (in seconds from the time given by the global attribute `RawData_Start_Time_UT`) of each lidar profile. In this example we have two different time scales: one is characterized by steps of 30 seconds (the 1064nm is acquired with this time scale) the other by steps of 60 seconds (532cross, 532parallel and 607nm).

Moreover the measurement start time is 00:00:01 UT and the measurement stop time is 00:05:01 UT. In this case we have to define:

```
Raw_Data_Start_Time =
  0, 0,
  60, 30,
  120, 60,
  180, 90,
  240, 120,
  _, 150,
  _, 180,
  _, 210,
  _, 240,
  _, 270 ;
```

The order used to fill this array defines the correspondence between the different time scales and the time scale index. In this example we have a time scale index of 0 for the time scale with steps of 60 seconds and a time scale index of 1 for the other one.

Raw_Data_Stop_Time(time, nb_of_time_scales) The same as previous item but for the data acquisition stop time. Following a similar procedure we have to define:

```
Raw_Data_Stop_Time =
  60, 30,
  120, 60,
  180, 90,
  240, 120,
  300, 150,
  _, 180,
  _, 210,
  _, 240,
  _, 270,
  _, 300 ;
```

Raw_Lidar_Data(time, channels, points) This 3 dimensional mandatory array has to be filled with the time-series of raw lidar data. The photoncounting profiles have to submitted in counts (so as integers) while the analog ones in mV. The order the user chooses to fill this array defines the correspondence between channel index and lidar data.

For example if we fill this array in such way that:

Raw_Lidar_Data(time,0,points	→ is the time-series of 1064 nm
Raw_Lidar_Data(time,1,points	→ is the time-series of 532 cross
Raw_Lidar_Data(time,2,points	→ is the time-series of 532 parallel
Raw_Lidar_Data(time,3,points	→ is the time-series of 607 nm

from now on the channel index 0 is associated to the 1064 channel, 1 to the 532 cross, 2 to the 532 parallel and 3 to the 607nm.

Raw_Bck_Start_Time(time_bck, nb_of_time_scales) This 2 dimensional optional array has to contain the acquisition start time (in seconds from the time given by the global attribute `RawBck_Start_Time_UT`) of each dark measurements profile. Following the same procedure used for the variable `Raw_Data_Start_Time` we have to define:

```
Raw_Bck_Start_Time =
  0, 0,
  60, 30,
  120, 60,
  _, 90,
```

```
  _, 120,
  _, 150;
```

Raw_Bck_Stop_Time(time_bck, nb_of_time_scales) The same as previous item but for the dark acquisition stop time. Following a similar procedure we have to define:

```
Raw_Bck_Stop_Time =
  60, 30,
  120, 60,
  180, 90,
  _, 120,
  _, 150,
  _, 180 ;
```

Background_Profile(time_bck, channels, points) This 3 dimensional optional array has to be filled with the time-series of the dark measurements data. The photoncounting profiles have to submitted in counts (so as integers) while the analog ones in mV. The user has to fill this array following the same order used in filling the array `Raw_Lidar_Data`:

Background_Profile(time_bck,0,points	→ dark time-series at 1064 nm
Background_Profile(time_bck,1,points	→ dark time-series at 532 cross
Background_Profile(time_bck,2,points	→ dark time-series at 532 parallel
Background_Profile(time_bck,3,points	→ dark time-series at 607 nm

channel_ID(channels) This mandatory array provides the link between the channel index within the Raw Lidar Data input file and the channel ID in SCC_DB. To fill this variable the user has to know which channel IDs in SCC_DB correspond to his lidar channels. For this purpose the SCC, in its final version will provide to the user a special tool to get these channel IDs through a Web interface. At the moment this interface is not yet available and these channel IDs will be communicated directly to the user by the NA5 people.

Anyway to continue the example let's suppose that the four lidar channels taken into account are mapped into SCC_DB with the following channel IDs:

1064 nm	→ channel ID=7
532 cross	→ channel ID=5
532 parallel	→ channel ID=6
607 nm	→ channel ID=8

In this case we have to define:

```
channel_ID = 7, 5, 6, 8 ;
```

id_timescale(channels) This mandatory array is introduced to determine which time scale is used for the acquisition of each lidar channel. In particular this array defines the link between the channel index and the time scale index. In our example we have two different time scales. Filling the arrays `Raw_Data_Start_Time` and `Raw_Data_Stop_Time` we have defined a time scale index of 0 for the time scale with steps of 60 seconds and a time scale index of 1 for the other one with steps of 30 seconds. In this way this array has to be set as:

```
id_timescale = 1, 0, 0, 0 ;
```

Laser_Pointing_Angle(scan_angles This mandatory array contains all the scan angles used in the measurement. In our example we have only one scan angle of 5 degrees with respect to the zenith, so we have to define:

```
Laser_Pointing_Angle = 5 ;
```

Laser_Pointing_Angle_of_Profiles(time, nb_of_time_scales) This mandatory array is introduced to determine which scan angle is used for the acquisition of each lidar profile. In particular this array defines the link between the time and time scales indexes and the scan angle index. In our example we have a single scan angle that has to correspond to the scan angle index 0. So this array has to be defined as:

```
Laser_Pointing_Angle_of_Profiles =
  0, 0,
  0, 0,
  0, 0,
  0, 0,
  0, 0,
  _, 0,
  _, 0,
  _, 0,
  _, 0,
  _, 0 ;
```

Laser_Shots(time, channels) This mandatory array stores the laser shots accumulated at each time for each channel. In our example the number of laser shots accumulated is 1500 for the 1064nm channels and 3000 for all the other channels. Moreover the laser shots do not change with the time. So we have to define this array as:

```
Laser_Shots =
 1500, 3000, 3000, 3000,
 1500, 3000, 3000, 3000,
 1500, 3000, 3000, 3000,
 1500, 3000, 3000, 3000,
 1500, 3000, 3000, 3000,
 1500, _, _, _,
 1500, _, _, _,
 1500, _, _, _,
 1500, _, _, _,
 1500, _, _, _ ;
```

Emitted_Wavelength(channels) This optional array defines the link between the channel index and the emission wavelength for each lidar channel. The wavelength has to be expressed in nm. This information can be also taken from SCC_DB. In our example we have:

```
Emitted_Wavelength = 1064, 532, 532, 532 ;
```

Detected_Wavelength(channels) This optional array defines the link between the channel index and the detected wavelength for each lidar channel. Here detected wavelength means the value of center of interferential filter expressed in nm. This information can be also taken from SCC_DB. In our example we have:

```
Detected_Wavelength = 1064, 532, 532, 607 ;
```

Raw_Data_Range_Resolution(channels) This optional array defines the link between the channel index and the raw range resolution for each channel. If the scan angle is different from zero this quantity is different from the vertical resolution. More precisely if α is the scan angle used and Δz is the range resolution the vertical resolution is calculated as $\Delta z' = \Delta z \cos \alpha$. This array has to be filled with Δz and not with $\Delta z'$. The unit is meters. This information can be also taken from SCC_DB. In our example we have:

```
Raw_Data_Range_Resolution = 7.5, 15.0, 15.0, 15.0 ;
```

ID_Range(channels) This optional array defines if a particular channel is configured as high, low or ultranear range channel. In particular a value 0 indicates a low range channel, a value 1 a high range channel and a value of 2 an ultranear range channel. If for a particular channel you don't separate between high and low range channel, please set the corresponding value to 1. This information can be also taken from SCC_DB. In our case we have to set:

```
ID_Range = 1, 1, 1, 1 ;
```

Scattering_Mechanism(channels) This optional array defines the scattering mechanism involved in each lidar channel. In particular the following values are adopted:

0	→ Total elastic backscatter
1	→ N_2 vibrational Raman backscatter
2	→ Cross polarization elastic backscatter
3	→ Parallel polarization elastic backscatter
4	→ H_2O vibrational Raman backscatter
5	→ Rotational Raman Stokes line close to elastic line
6	→ Rotational Raman Stokes line far from elastic line
7	→ Rotational Raman anti-Stokes line close to elastic line
8	→ Rotational Raman anti-Stokes line far from elastic line
9	→ Rotational Raman Stokes and anti-Stokes lines close to elastic line
10	→ Rotational Raman Stokes and anti-Stokes lines far from elastic line

This information can be also taken from SCC_DB. In our example we have:

```
Scattering_Mechanism = 0, 2, 3, 1 ;
```

Acquisition_Mode(channels) This optional array defines the acquisition mode (analog or photoncounting) involved in each lidar channel. In particular a value of 0 means analog mode and 1 photoncounting mode. This information can be also taken from SCC_DB. In our example we have:

```
Acquisition_Mode = 0, 1, 1, 1 ;
```

Laser_Repetition_Rate(channels) This optional array defines the repetition rate in Hz used to acquire each lidar channel. This information can be also taken from SCC_DB. In our example we are supposing we have only one laser with a repetition rate of 50 Hz so we have to set:

```
Laser_Repetition_Rate = 50, 50, 50, 50 ;
```

Dead_Time(channels) This optional array defines the dead time in ns associated to each lidar channel. The SCC will use the values given by this array to correct the photoncounting signals for dead time. Of course for analog signals no dead time correction will be applied (for analog channels the corresponding dead time values have to be set to undefined value). This information can be also taken from SCC_DB. In our example the 1064 nm channel is acquired in analog mode so the corresponding dead time value has to be undefined. If we suppose a dead time of 10 ns for all other channels we have to set:

```
Dead_Time = _, 10, 10, 10 ;
```

Dead_Time_Corr_Type(channels This optional array defines which kind of dead time correction has to be applied on each photoncounting channel. The SCC will correct the data supposing a not-paralyzable channel if a value of 0 is found while a paralyzable channel is supposed if a value of 1 is found. Of course for analog signals no dead time correction will be applied and so the corresponding values have to be set to undefined value. This information can be also taken from SCC_DB. In our example the 1064 nm channel is acquired in analog mode so the corresponding has to be undefined. If we want to consider all the photoncounting signals as not-paralyzable ones: we have to set:

```
Dead_Time_Corr_Type = _, 0, 0, 0 ;
```

Trigger_Delay(channels) This optional array defines the delay (in ns) of the middle of the first rangebin with respect to the output laser pulse for each lidar channel. The SCC will use the values given by this array to correct for trigger delay. This information can be also taken from SCC_DB. Let's suppose that in our example all the photoncounting channels are not affected by this delay and only the analog channel at 1064nm is acquired with a delay of 50ns. In this case we have to set:

```
Trigger_Delay = 50, 0, 0, 0 ;
```

Background_Mode(channels This optional array defines how the atmospheric background has to be subtracted from the lidar channel. Two options are available for the calculation of atmospheric background:

1. Average in the far field of lidar channel. In this case the value of this variable has to be 1

2. Average within pre-trigger bins. In this case the value of this variable has to be 0

This information can be also taken from SCC_DB. Let's suppose in our example we use the pre-trigger for the 1064nm channel and the far field for all other channels. In this case we have to set:

```
Background_Mode = 0, 1, 1, 1 ;
```

Background_Low(channels) This mandatory array defines the minimum altitude (in meters) to consider in calculating the atmospheric background for each channel. In case pre-trigger mode is used the corresponding value has to be set to the rangebin to be used as lower limit (within pre-trigger region) for background calculation. In our example, if we want to calculate the background between 30000 and 50000 meters for all photoncounting channels and we want to use the first 500 pre-trigger bins for the background calculation for the 1064nm channel we have to set:

```
Background_Low= 0, 30000, 30000, 30000 ;
```

Background_High(channels) This mandatory array defines the maximum altitude (in meters) to consider in calculating the atmospheric background for each channel. In case pre-trigger mode is used the corresponding value has to be set to the rangebin to be used as upper limit (within pre-trigger region) for background calculation. In our example, if we want to calculate the background between 30000 and 50000 meters for all photoncounting channels and we want to use the first 500 pre-trigger bins for the background calculation for the 1064nm channel we have to set:

```
Background_High = 500, 50000, 50000, 50000 ;
```

Molecular_Calc This mandatory variable defines the way used by SCC to calculate the molecular density profile. At the moment two options are available:

1. US Standard Atmosphere 1976. In this case the value of this variable has to be 0
2. Radiosounding. In this case the value of this variable has to be 1

If we decide to use the option 1. we have to provide also the measured pressure and temperature at lidar station level. Indeed if we decide to use the option 2. a radiosounding file has to be submitted separately in NetCDF format (the structure of this file is summarized in table tab:sounding). Let's suppose we want to use the option 1. so:

```
Molecular_Calc = 0 ;
```

Pressure_at_Lidar_Station Because we have chosen the US Standard Atmosphere for calculation of the molecular density profile we have to give the pressure in hPa at lidar station level:

```
Pressure_at_Lidar_Station = 1010 ;
```

Temperature_at_Lidar_Station Because we have chosen the US Standard Atmosphere for calculation of the molecular density profile we have to give the temperature in C at lidar station level:

```
Temperature_at_Lidar_Station = 19.8 ;
```

Depolarization_Factor(channels) This array is required only for lidar systems that use the two depolarization channels for the backscatter retrieval. It represents the factor f to calculate the total backscatter signal S_t combining its cross S_c and parallel S_p components: $S_t = S_p + fS_c$. This factor is mandatory only for systems acquiring S_c and S_p and not S_t. For systems acquiring S_c, S_p and S_t this factor is optional and it will be used only for depolarizaton ratio calculation. Moreover only the values of the array corresponding to cross polarization channels will be considered; all other values will be not taken into account and should be set to undefined value. In our example for the wavelength 532nm we have only the cross and the parallel components and not the total one. So we have to give the value of this factor only in correspondence of the 532nm cross polarization channel that corresponds to the channel index 1. Suppose that this factor is 0.88. Moreover, because we don't have any other depolarization channels we have also to set all other values of the array to undefined value.

```
Depolarization_Factor = _,0.88,_,_ ;
```

LR_Input(channels) This array is required only for lidar channels for which elastic backscatter retrieval has to be performed. It defines the lidar ratio to be used within this retrieval. Two options are available:

1. The user can submit a lidar ratio profile. In this case the value of this variable has to be 0.
2. A fixed value of lidar ratio can be used. In this case the value of this variable has to be 1.

If we decide to use the option 1. a lidar ratio file has to be submitted separately in NetCDF format (the structure of this file is summarized in table tab:lr). If we decide to use the option 2. the fixed value of lidar ratio will be taken from SCC_DB. In our example we have to give a value of this array only for the 1064nm lidar channel because for the 532nm we will be able to retrieve a Raman backscatter coefficient. In case we want to use the fixed value stored in SCC_DB we have to set:

```
LR_Input = 1,_,_,_ ;
```

DAQ_Range(channels) This array is required only if one or more lidar signals are acquired in analog mode. It gives the analog scale in mV used to acquire the analog signals. In our example we have only the 1064nm channel acquired in analog mode. If we have used a 100mV analog scale to acquire this channel we have to set:

```
DAQ_Range = 100,_,_,_ ;
```

4.2.3 Global attributes

Measurement_ID This mandatory global attribute defines the measurement ID corresponding to the actual lidar measurement. It is a string composed by 12 characters. The first 8 characters give the start date of measurement in the format YYYYMMDD. The next 2 characters give the Earlinet call-sign of the station. The last 2 characters are used to distinguish between different time-series within the same date. In our example we have to set:

```
Measurement_ID= "20090130cc00" ;
```

RawData_Start_Date This mandatory global attribute defines the start date of lidar measurements in the format YYYYMMDD. In our case we have:

```
RawData_Start_Date = "20090130" ;
```

RawData_Start_Time_UT This mandatory global attribute defines the UT start time of lidar measurements in the format HHMMSS. In our case we have:

```
RawData_Start_Time_UT = "000001" ;
```

RawData_Stop_Time_UT‘‘ This mandatory global attribute defines the UT stop time of lidar measurements in the format HHMMSS. In our case we have:

```
RawData_Stop_Time_UT = "000501" ;
```

RawBck_Start_Date This optional global attribute defines the start date of dark measurements in the format YYYYMMDD. In our case we have:

```
RawBck_Start_Date = "20090129" ;
```

RawBck_Start_Time_UT This optional global attribute defines the UT start time of dark measurements in the format HHMMSS. In our case we have:

```
RawBck_Start_Time_UT = "235001" ;
```

RawBck_Stop_Time_UT This optional global attribute defines the UT stop time of dark measurements in the format HHMMSS. In our case we have:

```
    RawBck_Stop_Time_UT = "235301" ;
```

4.3 Example of file (CDL format)

To summarize we have the following NetCDF Raw Lidar Data file (in CDL format):

```
dimensions:
        points = 5000 ;
        channels = 4 ;
        time = UNLIMITED ; // (10 currently)
        nb_of_time_scales = 2 ;
        scan_angles = 1 ;
        time_bck = 6 ;
variables:
        int channel_ID(channels) ;
        int Laser_Repetition_Rate(channels) ;
        double Laser_Pointing_Angle(scan_angles) ;
        int ID_Range(channels) ;
        int Scattering_Mechanism(channels) ;
        double Emitted_Wavelength(channels) ;
        double Detected_Wavelength(channels) ;
        double Raw_Data_Range_Resolution(channels) ;
        int Background_Mode(channels) ;
        double Background_Low(channels) ;
        double Background_High(channels) ;
        int Molecular_Calc ;
        double Pressure_at_Lidar_Station ;
        double Temperature_at_Lidar_Station ;
        int id_timescale(channels) ;
        double Dead_Time(channels) ;
        int Dead_Time_Corr_Type(channels) ;
        int Acquisition_Mode(channels) ;
        double Trigger_Delay(channels) ;
        int LR_Input(channels) ;
        int Laser_Pointing_Angle_of_Profiles(time, nb_of_time_scales) ;
        int Raw_Data_Start_Time(time, nb_of_time_scales) ;
        int Raw_Data_Stop_Time(time, nb_of_time_scales) ;
        int Raw_Bck_Start_Time(time_bck, nb_of_time_scales) ;
        int Raw_Bck_Stop_Time(time_bck, nb_of_time_scales) ;
        int Laser_Shots(time, channels) ;
        double Raw_Lidar_Data(time, channels, points) ;
        double Background_Profile(time_bck, channels, points) ;
        double DAQ_Range(channels) ;

// global attributes:
                :Measurement_ID = "20090130cc00" ;
                :RawData_Start_Date = "20090130" ;
                :RawData_Start_Time_UT = "000001" ;
                :RawData_Stop_Time_UT = "000501" ;
                :RawBck_Start_Date = "20090129" ;
                :RawBck_Start_Time_UT = "235001" ;
                :RawBck_Stop_Time_UT = "235301" ;

data:

 channel_ID = 7, 5, 6, 8 ;
```

```
Laser_Repetition_Rate = 50, 50, 50, 50 ;

Laser_Pointing_Angle = 5 ;

ID_Range = 1, 1, 1, 1 ;

Scattering_Mechanism = 0, 2, 3, 1 ;

Emitted_Wavelength = 1064, 532, 532, 532 ;

Detected_Wavelength = 1064, 532, 532, 607 ;

Raw_Data_Range_Resolution = 7.5, 15, 15, 15 ;

Background_Mode = 0, 1, 1, 1 ;

Background_Low = 0, 30000, 30000, 30000 ;

Background_High = 500, 50000, 50000, 50000 ;

Molecular_Calc = 0 ;

Pressure_at_Lidar_Station = 1010 ;

Temperature_at_Lidar_Station = 19.8 ;

id_timescale = 1, 0, 0, 0 ;

Dead_Time = _, 10, 10, 10 ;

Dead_Time_Corr_Type = _, 0, 0, 0 ;

Acquisition_Mode = 0, 1, 1, 1 ;

Trigger_Delay = 50, 0, 0, 0 ;

LR_Input = 1,_,_,_ ;

DAQ_Range = 100,_,_,_ ;

Laser_Pointing_Angle_of_Profiles =
 0, 0,
 0, 0,
 0, 0,
 0, 0,
 0, 0,
 _, 0,
 _, 0,
 _, 0,
 _, 0,
 _, 0 ;


Raw_Data_Start_Time =
 0, 0,
 60, 30,
 120, 60,
 180, 90,
```

```
  240, 120,
  _, 150,
  _, 180,
  _, 210,
  _, 240,
  _, 270 ;

 Raw_Data_Stop_Time =
  60, 30,
  120, 60,
  180, 90,
  240, 120,
  300, 150,
  _, 180,
  _, 210,
  _, 240,
  _, 270,
  _, 300 ;


 Raw_Bck_Start_Time =
  0, 0,
  60, 30,
  120, 60,
  _, 90,
  _, 120,
  _, 150;


Raw_Bck_Stop_Time =
  60, 30,
  120, 60,
  180, 90,
  _, 120,
  _, 150,
  _, 180 ;


 Laser_Shots =
  1500, 3000, 3000, 3000,
  1500, 3000, 3000, 3000,
  1500, 3000, 3000, 3000,
  1500, 3000, 3000, 3000,
  1500, 3000, 3000, 3000,
  1500, _, _, _,
  1500, _, _, _,
  1500, _, _, _,
  1500, _, _, _,
  1500, _, _, _ ;


 Raw_Lidar_Data = ...

 Background_Profile = ...
```

Please keep in mind that in case you submit a file like the previous one all the parameters present in it will be used by the SCC even if you have different values for the same parameters within the SCC_DB. If you want to use the values already stored in SCC_DB (this should be the usual way to use SCC) the Raw Lidar Data input file has to be modified

as follows:

```
dimensions:
        points = 5000 ;
        channels = 4 ;
        time = UNLIMITED ; // (10 currently)
        nb_of_time_scales = 2 ;
        scan_angles = 1 ;
        time_bck = 6 ;
variables:
        int channel_ID(channels) ;
        double Laser_Pointing_Angle(scan_angles) ;
        double Background_Low(channels) ;
        double Background_High(channels) ;
        int Molecular_Calc ;
        double Pressure_at_Lidar_Station ;
        double Temperature_at_Lidar_Station ;
        int id_timescale(channels) ;
        int Laser_Pointing_Angle_of_Profiles(time, nb_of_time_scales) ;
        int Raw_Data_Start_Time(time, nb_of_time_scales) ;
        int Raw_Data_Stop_Time(time, nb_of_time_scales) ;
        int Raw_Bck_Start_Time(time_bck, nb_of_time_scales) ;
        int Raw_Bck_Stop_Time(time_bck, nb_of_time_scales) ;
        int LR_Input(channels) ;
        int Laser_Shots(time, channels) ;
        double Raw_Lidar_Data(time, channels, points) ;
        double Background_Profile(time_bck, channels, points) ;
        double DAQ_Range(channels) ;

// global attributes:
                :Measurement_ID = "20090130cc00" ;
                :RawData_Start_Date = "20090130" ;
                :RawData_Start_Time_UT = "000001" ;
                :RawData_Stop_Time_UT = "000501" ;
                :RawBck_Start_Date = "20090129" ;
                :RawBck_Start_Time_UT = "235001" ;
                :RawBck_Stop_Time_UT = "235301" ;

data:

 channel_ID = 7, 5, 6, 8 ;

 Laser_Pointing_Angle = 5 ;

 Background_Low = 0, 30000, 30000, 30000 ;

 Background_High = 500, 50000, 50000, 50000 ;

 Molecular_Calc = 0 ;

 Pressure_at_Lidar_Station = 1010 ;

 Temperature_at_Lidar_Station = 19.8 ;

 id_timescale = 1, 0, 0, 0 ;

 LR_Input = 1,_,_,_ ;

 DAQ_Range = 100,_,_,_ ;
```

```
Laser_Pointing_Angle_of_Profiles =
 0, 0,
 0, 0,
 0, 0,
 0, 0,
 0, 0,
 _, 0,
 _, 0,
 _, 0,
 _, 0,
 _, 0 ;


Raw_Data_Start_Time =
 0, 0,
 60, 30,
 120, 60,
 180, 90,
 240, 120,
 _, 150,
 _, 180,
 _, 210,
 _, 240,
 _, 270 ;

Raw_Data_Stop_Time =
 60, 30,
 120, 60,
 180, 90,
 240, 120,
 300, 150,
 _, 180,
 _, 210,
 _, 240,
 _, 270,
 _, 300 ;


Raw_Bck_Start_Time =
 0, 0,
 60, 30,
 120, 60,
 _, 90,
 _, 120,
 _, 150;


Raw_Bck_Stop_Time =
 60, 30,
 120, 60,
 180, 90,
 _, 120,
 _, 150,
 _, 180 ;


Laser_Shots =
 1500, 3000, 3000, 3000,
```

```
 1500, 3000, 3000, 3000,
 1500, 3000, 3000, 3000,
 1500, 3000, 3000, 3000,
 1500, 3000, 3000, 3000,
 1500, _, _, _,
 1500, _, _, _,
 1500, _, _, _,
 1500, _, _, _,
 1500, _, _, _ ;


Raw_Lidar_Data = ...

Background_Profile = ...
```

This example file contains the minimum collection of mandatory information that has to be found within the Raw Lidar Data input file. If it is really necessary, the user can decide to add to these mandatory parameters any number of additional parameters considered in the previous example.

Finally, suppose we want to make the following changes with respect to the previous example:

1. use a sounding file for molecular density calculation instead of "US Standar Atmosphere 1976"
2. supply a lidar ratio profile to use in elastic backscatter retrieval instead of a fixed value
3. provide a overlap function for overlap correction

In this case we have to generate the following NetCDF additional files:

rs_20090130cc00.nc The name of Sounding Data file has to be computed as follows: `"rs_"``+``Measurement_ID` The structure of this file is summarized in table tab:sounding.

ov_20090130cc00.nc The name of Overlap file has to be computed as follows: `"ov_"``+``Measurement_ID` The structure of this file is summarized in table tab:overlap.

lr_20090130cc00.nc The name of Lidar Ratio file has to be computed as follows: `"lr_"``+``Measurement_ID` The structure of this file is summarized in table tab:lr.

Moreover we need to apply the following changes to the Raw Lidar Data input file:

1. Change the value of the variable `Molecular_Calc` as follows:

   ```
   Molecular_Calc = 1 ;
   ```

   Of course the variables `Pressure_at_Lidar_Station` and `Temperature_at_Lidar_Station` are not necessary anymore.

2. Change the values of the array `LR_Input` as follows:

   ```
   LR_Input = 0,_,_,_ ;
   ```

3. Add the global attribute `Sounding_File_Name`

   ```
   Sounding_File_Name = "rs_20090130cc00.nc" ;
   ```

5. Add the global attribute `LR_File_Name`

   ```
   LR_File_Name = "lr_20090130cc00.nc" ;
   ```

6. Add the global attribute `Overlap_File_Name`

   ```
   Overlap_File_Name = "ov_20090130cc00.nc" ;
   ```

CHAPTER

FIVE

USER MANAGEMENT

5.1 Acount types

5.2 Requesting a new account

5.3 User account security

CHAPTER

SIX

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*