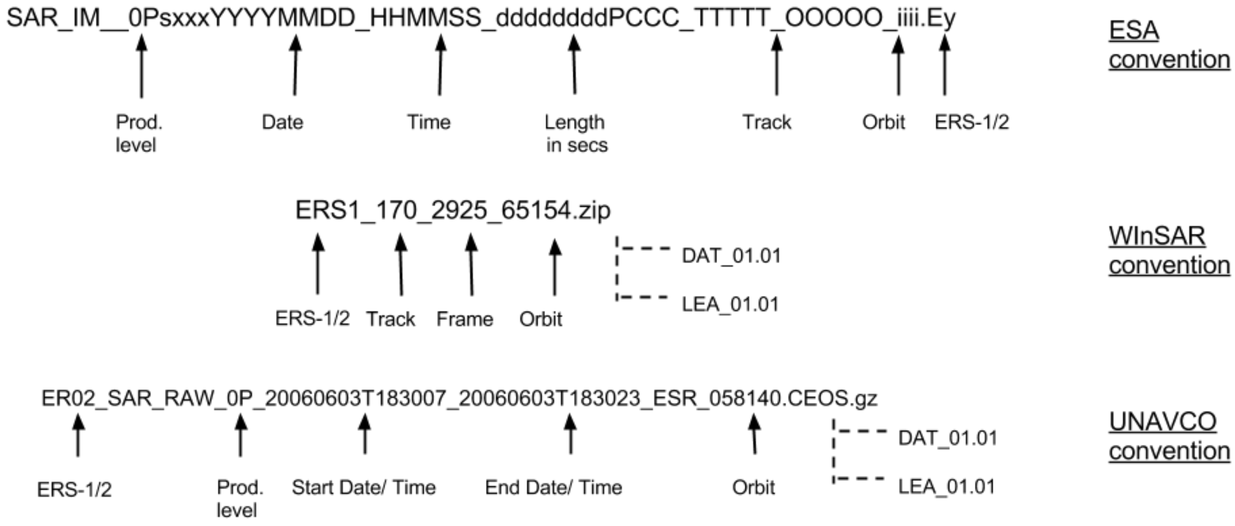
# 1. Understanding ERS Data Set Names

The European Space Agency (ESA) launched and operated two nearly identical satellites call the European Remote Sensing (ERS) satellites, ERS-1 (1992–2000) and ERS-2 (1995-2011, although data acquired after 2000 has problems) (see also Table 1 of Lab 3.0). In this lab, we will learn how to process data from these satellites. Because the two satellites had the same radar characteristics and the same orbits for most of their operation, most of the data from the two satellites can be used interchangeably. The two satellites were operated in what ESA called a Tandem mission especially during 1995–1996 and 1998–1999 where the ERS-1 and ERS-2 satellites acquired data with a 1-day time separation over most of the Earth's land area.



The graphic above shows three standard naming conventions for ERS data files from three different processing and distribution centers in the world: The ESA archives, the Alaska Satellite Facility, and UNAVCO. Some ERS data is also available from the GEO Geohazards Supersites. The ESA "Envisat-style" file format (top line) combines the data with the metadata into a single file, similar to the format for Envisat data (see Lab 5). ASF and UNAVCO supply compressed files that contain two files each, the data, and metadata (so-called "leader" file), with different names for the container file. The data files inside have very simple and generic names as show above.

# 2. How to insert ERS filenames into the ISCE xml input files

The basic ISCE input file is similar to that for ALOS PALSAR. If you have not gone through the tutorial for running ISCE with ALOS, you should at least read through Step 2 of that tutorial, if not also setting up the input file and processing the ALOS data used in Step 3 of that tutorial. In this tutorial we assume you have read Step 2 of the ALOS tutorial and will only talk about the differences for processing ERS-1 and ERS-2 data.

As in the ALOS input files the ERS input files contain the `MASTER` and `SLAVE` component tags that contain the property tags `IMAGEFILE`, `LEADERFILE`, and `OUTPUT`. Different from the ALOS input files is two new property tags (the orbit type and the path to the directory containing the orbit files) that contain information about the orbit data for the data take. The ALOS `LEADERFILE` contained sufficiently accurate orbit data for InSAR processing. The ERS-1 orbit data in the `LEADERFILE`, however, are generally not adequate for accurate processing of InSAR data. The Delft Institute for Earth-oriented Space Research (DEOS) provides precise orbits for the entire ERS-1 mission (and most of the ERS-2 mission). We have installed these orbits in the directory `/home/ubuntu/data/orbits/ERS/ODR/ERS1`.

If you use the `ls` command to list the contents of that directory you will see over 500 files named `ODR.nnn`. These files contain the orbital data records (time and position) for several day long “arcs” of the orbit. The number `nnn` in the name of the ODR file indicates the arc number. In that directory you will find the key to determining which arc is needed for your data take in the `arclist` file,

```
> ls /home/ubuntu/data/orbits/ERS/ODR/ERS1/arclist
```

The `arclist` file lists the start and end time for each of the ODR orbital data record arc files. You can use the `cat` command to list the more than 500 lines of the arclist file or you can use the `more` command to see just one page full of the contents of the arclist file, just to get a taste of the contents of that file (page through the file using the “spacebar” and terminate the output by typing “q”),

```
> more /home/ubuntu/data/orbits/ERS/ODR/ERS1/arclist
```

You will never need to read the `arclist` file to find the correct ODR arc file to use. The ISCE processor does that for you. You only need to tell ISCE the type of orbit to use in the processing and the name of the directory containing the `arclist` and ODR files. The way to tell ISCE this information is to insert the following `ORBIT_TYPE` and `ORBIT_DIRECTORY` tags in *both* the `Master` and the `Slave component`:

```
<property name="ORBIT_TYPE">
    <value>ODR</value>
</property>

<property name="ORBIT_DIRECTORY">
    <value>/home/ubuntu/data/orbits/ERS/ODR/ERS1</value>
</property>
```

# 3. Processing ERS data with ISCE

You should change your directory to the lab4 directory for ERS processing. You can position yourself in that directory from wherever you might be positioned currently as follows:

```
> cd
> cd data/lab4/ers
```

There are two directories there for two different data acquisition dates:

```
> ls
19950421  19971227
```

Within those directories you will see ‘tar.gz’ files, which are compressed containers of several files. The names of these ‘tar.gz’ files follow the UNAVCO name convention described at the start of this lab. To unpack these files use the tar command as follows,

```
> cd 19950421
> ls
ER01_SAR_IM__0P_19950421T183128_19950421T183145_DPA_19697_0000.CEOS.tar.gz
> tar -xzvf ER01_SAR_IM__0P_19950421T183128_19950421T183145_DPA_19697_0000.CEOS.tar.gz
> ls
DAT_01.001
ER01_SAR_IM__0P_19950421T183128_19950421T183145_DPA_19697_0000.CEOS.tar.gz
LEA_01.001
NUL_DAT.001
SAR_IM__0PXDLR19950421_183128_00000017G145_00170_19697_7041.E1.ps
VDF_DAT.001
```

The relevant files as inputs to ISCE are the file `LEA_01.001`, which is the leader file, and the file `DAT_01.001`, which is the image file.

Similarly for the `19971227` directory:

```
> cd ../19971227
> ls
ER02_SAR_IM__0P_19971227T183128_19971227T183145_DPA_14052_0000.CEOS.tar.gz
> tar -xzvf ER02_SAR_IM__0P_19971227T183128_19971227T183145_DPA_14052_0000.CEOS.tar.gz
> ls
DAT_01.001
ER02_SAR_IM__0P_19971227T183128_19971227T183145_DPA_14052_0000.CEOS.tar.gz
LEA_01.001
```

```
NUL_DAT.001
SAR_IM__0PXDLR19971227_183128_00000017A028_00170_14052_7031.E2.ps
VDF_DAT.001
```

Note that the `tar.gz` file in the 1995 directory starts with `ER01` indicating that it was from the ERS-1 instrument and the `tar.gz` file in the 1997 directory starts with `ER02` indicating that it was from the ERS-2 instrument. We will process these data sets from these two different compatible instruments using `insarApp.py`. In each directory there is an IMAGERY file named `DAT_01.001` and a LEADER file named `LEA_01.001` corresponding to two different frames with different acquisition times encoded in the associated `tar.gz` filenames. Because the actual file names for each date are the same, it is essential that the data for each date be in separate directories.

To prepare for processing the ERS data, first make a new processing directory in the `data/lab4/ers` directory (first making sure that you are in the proper directory using the `pwd` command or use the `cd` command once without an argument to first move to the top directory and then with the argument `data/lab4/ers` to move into the proper directory),

```
> cd
> cd data/lab4/ers
> pwd
/home/ubuntu/data/lab4/ers

> mkdir 19950421_19971227
> cd 19950421_19971227
> pwd
/home/ubuntu/data/lab4/ers/19950421_19971227
```

At this time you should be able to copy your input files (whether you used the all-in-one or the multiple input files styles) from the `lab3/alos/20070612_20090802` directory into the current `lab4/ers/19950421_19971227` directory,

```
> cp ../../../lab3/alos/20070612_20090802/insar_input.xml .
```

Notice the “`.`” at the end of that command line. It is shorthand for the “current directory”. This command copies the file `insar_input.xml` at the path

```
../../../lab3/alos/20070612_20090802/insar_input.xml
```

into the current directory giving it the same name `insar_input.xml`. Note that the name of this file is not significant as it is entered as the first (usually only) xml file on the command line. You may call it anything you like, such as `insar_19950421_19971227.xml` or `insarApp.xml`.

If you used the all-in-one style, then this is the only file you need to copy. If you used multiple files, then you will need to copy the "Master" and "Slave" xml files also. Those files can have any name you chose. Remember, all that matters is that the file names you use are also the names entered in the catalog tags in the Master and Slave component tags in the insar_input.xml file.

If you followed the names in the text of the ALOS Datasets lab, then you probably used Master.xml and Slave.xml, and you can use the cp command as above with the "." at the end of the command to copy those files from the alos directory to the current directory using the same names.

If you followed the "date" naming convention given in the ALOS examples link, however, then you may have called them 20070612.xml and 20090802.xml. When you copy those files to the current directory, then you should change the date on the filename. For the Master you would use the command,

```
> cp ../../../lab3/alos/20070612_20090802/20070612.xml 19950421.xml
```

The second argument on this version of the cp command says to copy the contents of the file 20070612.xml in directory ../../alos/20070612_20090802 into a file named 920424.xml in the current directory.

SImilarly for the Slave,

```
> cp ../../../lab3/alos/20070612_20090802/20090802.xml 19971227.xml
```

Now use your favorite editor to enter the paths and names of this lab's ERS files into the IMAGEFILE, LEADERFILE, and OUTPUT tags in your input files. Then you should be able to copy or type in the ORBIT_TYPE and ORBIT_DIRECTORY tags given in Step 2 above.

Once you have inserted these tags into *both* your Master and Slave component tags, then you should be ready to process the data. At this point you can either go ahead and try to run insarApp.py and let the computer tell you if you have prepared the input file(s) correctly (if the processing runs to the end in about 20 minutes) or incorrectly (if the processing terminates early with a computer generated "traceback" indicating the location of the error in the code),

```
> insarApp.py insar_input.xml
```

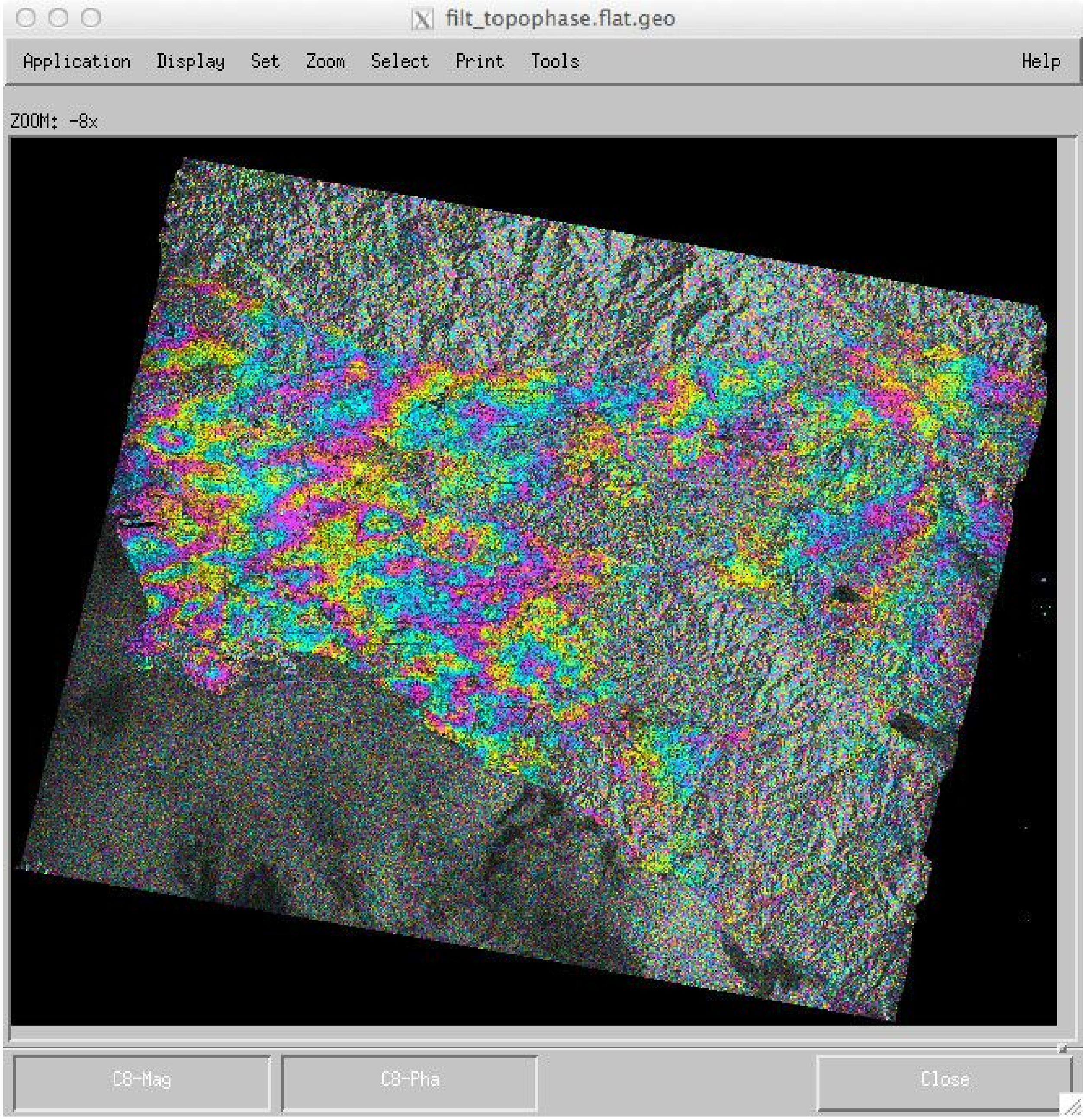
where insar_input.xml is the file you have prepared using either the all-in-one file containing the complete Master and Slave component tags or the multiple input files where the insar_input.py file contains catalog tags that point to two other xml files containing the information for the Master and Slave components.

If you choose to be cautious or if you attempted to process and it did not run to the end successfully and you cannot figure out what is wrong with your input files, then you can compare your input file(s) with these examples.

## 4. Your completed run

Now we can look at the final geocoded, wrapped interferogram with MDX:

```
> mdx.py filt_topophase.flat.geo &
```


filt_topophase.flat.geo
Application
Display
Set
Zoom
Select
Print
Tools
Help
C8-Mag
C8-Pha
Close

ZOOM: -8x