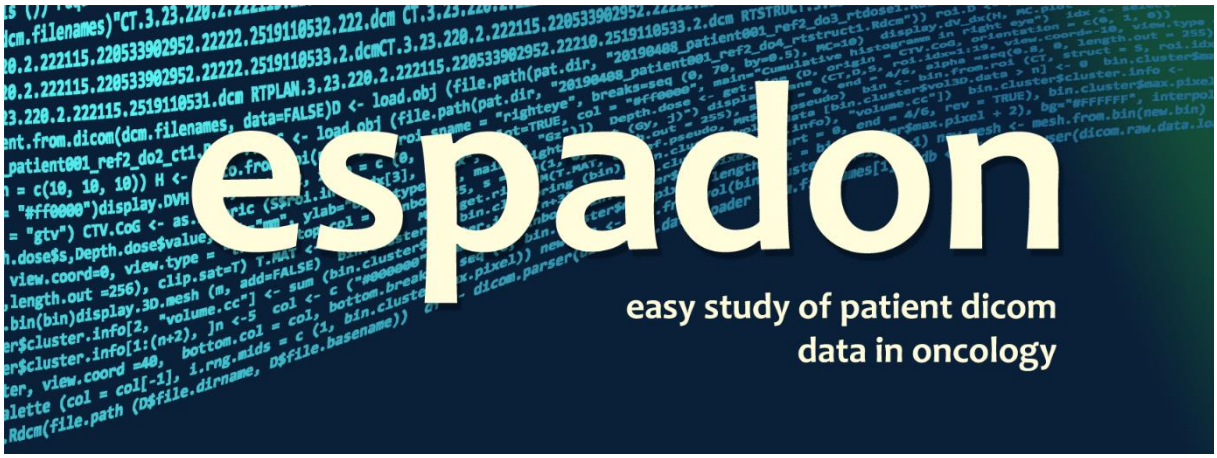


Data loading

C. Fontbonne and J.-M. Fontbonne | LPC Caen
Normandie Univ, ENSICAEN, UNICAEN, CNRS/IN2P3, LPC Caen, 14000 Caen, France
Contact.espadon@lpccaen.in2p3.fr
https://espadon.cnrs.fr/

Table of content

1 Introduction 3

2 Loading DICOM files into espadon 3

2.1 Loading data for simple studies 3

2.2 Exploring the Patient Oriented View (POV) 4

3 DICOM & complex studies 5

3.1 DICOM exams inspection 5

3.2 Loading specific exams 7

4 Rdcm format for fast and light DICOM handling 7

4.1 DICOM to Rdcm conversion 8

4.2 Rdcm handling 8

5 What we learned 8

1 Introduction

Espadon was built to facilitate access to the content of files in DICOM format for more elaborate clinical or research use. Before discussing its possibilities in detail, the first step is to load these files and explore their connections. This is precisely the purpose of this document.

We will first look at how DICOM files are loaded into espadon, how they are interconnected and introduce the patient-oriented view, which is a simple entry point to more advanced exploitation. For this purpose, it will be useful to have a set of files containing, for example, a treatment CT, the rt-dose file produced by the TPS, and the associated rt-struct file.

Clinical cases are often much more complex and integrate different imaging and/or dosimetry modalities. In a second step, we will see how to work on complex cases by using the patient-oriented view.

Finally, we will learn how to use espadon's internal storage format, Rdcm, which facilitates access to DICOM data by making it faster and less storage-intensive.

2 Loading DICOM files into espadon

Despite the fact that espadon reads the DICOM format natively, everything is done so that the user does not have to worry about it. In practice, espadon will convert the images (CT, MR, etc.) into 3D matrices, and will connect these matrices cleanly with the associated rt-dose and rt-struct files, managing the change of reference frames from the reg files. For special cases outside this framework, espadon offers several low-level functions for accessing the DICOM content itself (see the appropriate document).

For now, let's just load the DICOM content associated with a patient. For this, we assume that the data is stored in a unique directory associated with that patient. The basic rule in espadon is that **a directory can contain any type of data, but that data will be associated with a single patient**. So, if you have CTs, MRs, rt-plan, rt-dose, rt-struct, etc., corresponding to a patient, put them in a clearly identified directory. This is the only prerequisite.

2.1 Loading data for simple studies

In order to load the DICOM files, you can execute the following instructions:

```
require (espadon)

pat.dir <- choose.dir ()
pat <- load.patient.from.dicom (pat.dir, data = TRUE)
```

First line just loads espadon into your workspace. Second line opens a folder selection window, and last line loads data and store it into `pat` variable. By executing the last line, you will see a progress bar, which, when completed, will mean that all the data has been loaded and collected in the `pat` list.

Note the `data = TRUE` option when loading. It means that all the information is requested to be loaded. We will see later that this is not necessary at this stage and that espadon offers several possibilities to load only the useful information, which is quite practical for patients with multiple examinations, so as to limit the memory space required.

That's usually all there is to it, at least for simple studies!

2.2 Exploring the Patient Oriented View (POV)

If you want to see what you have loaded and you are using Rstudio, click on `pat`, otherwise run `str (pat, max.level = 1)`. This gives access to the patient-oriented view (POV). In our example, it looks like this:

```
> str (pat, max.level = 1)
List of 8
 $ patient           :'data.frame':     1 obs. of  3 variables:
 $ pat.pseudo        : chr "a07203ca"
 $ description       :'data.frame':     3 obs. of  9 variables:
 $ description.by.reg:List of 1
 $ T.MAT             :List of 4
  ..- attr(*, "class")= chr "t.mat"
 $ ct                :List of 1
 $ rtdose            :List of 1
 $ rtstruct          :List of 1
```

The loaded objects are located after the `$T.MAT` field. They are sorted by modalities. Here we captured only one CT, one rt-dose and one rt-struct (the last three lines).

`$T.MAT` is a list containing all the information needed to change reference frames. It is essential to espadon when objects, such as MRs and CTs, are to be represented and/or used simultaneously for calculations.

`$patient` and `$pat.pseudo` correspond to the patient's personal information (unique identifier, date of birth and gender). Normally, these are unique and can be repeated if multiple instances were discovered during loading. This would probably mean that the directory actually contains several patients mixed together.

`$description` and `$description.by.reg` contain information about the discovered objects and their connections. If all the objects are in the same geometric reference frame, we will look at `$description`, otherwise, we will have to examine `$description.by.reg` which will contain as many lists as there are common repositories. Here, `$description` contains:

	PIN	modality	obj	ref.pseudo	nb.of.subobject	description	nb	max	object.alias
1	a07203ca	ct	20160709_a07203ca_ref1_do1_ct	ref1	1	|	107	3071	20160709_a07203ca_ref1_do1_ct1
2	a07203ca	rtdose	20200601_a07203ca_ref1_do3_rtdose	ref1	1	|1	72	53.93	20200601_a07203ca_ref1_do3_rtdo
3	a07203ca	rtstruct	20200422_a07203ca_ref1_do2_rtstruct	ref1	1	|RS: Unapproved Structure Set	56	NA	20200422_a07203ca_ref1_do2_rtstr

`$modality` is the modality of the objects found. `$obj` is the espadon internal name of the object. `$ref.pseudo` is the name espadon gave to the reference frame at download. Only objects with the same reference frame or reference frames linked by a reg file can be used simultaneously. Here, we have no problem as `$descrition.by.reg` contains only one list, meaning all objects share the same reference frame. `$nb` and `$max` are quick information about the object depending on the context. For example, for an rt-struct, it is the number of regions of interest, in a dosimetry or imaging, the number of planes...

3 DICOM & complex studies

In the case of complex studies involving multiple modalities, it is usually desirable to sort the data before considering their use. Firstly, it is not necessarily clear what is available, and also whether these data are connected and share the same reference frame. It is then useful to proceed in several steps.

1) load the patient-oriented views, without the data.
2) manually or automatically select relevant exams.
3) Load the relevant exams.

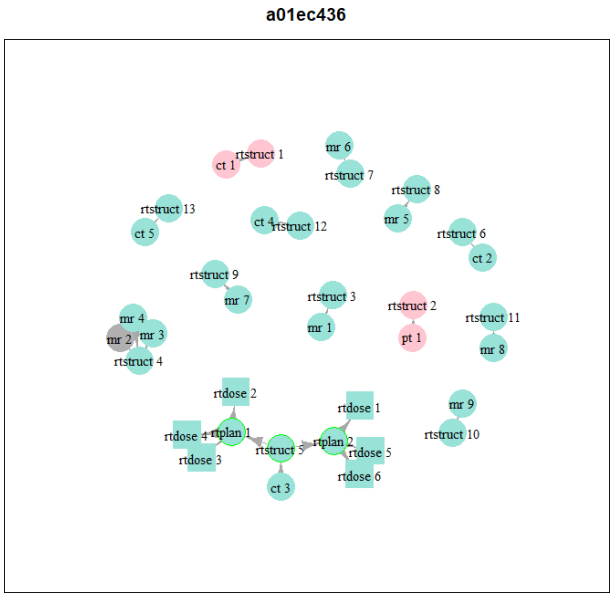
3.1 DICOM exams inspection

To inspect the examinations, it is sufficient to load the patient-oriented view by specifying `data = FALSE` (in fact, this is the default option in espadon):

```
pat.dir <- choose.dir ()
pat <- load.patient.from.dicom (pat.dir)
```

From here, one can of course visualise, as before, the patient-oriented view, but espadon also incorporates a tool for studying connections between objects. This is the `display.obj.links` function:

```
display.obj.links (pat)
```



Each object is referenced by its modality, and its number in the list of the patient-oriented view.

Directly connected objects are linked by an arrow. This is the case, for example, for the set of files rt-plan; rt-struct; CT and rt-dose at the bottom. The direction of the arrows depends on the TPS that produced the calculations and is not important as such.

The objects with the same colours were acquired in different reference frames; however, espadon has found a reg file that allows them to be transported from one to another without further difficulty. This is the case for the green and pink sets. Thus, it will be possible to work simultaneously on the dosimetry group and the MR group on the left of the graph, for example. However, it will not be possible to use the PT, CT group in pink with the dosimetry in green.

When espadon detects a reading error, for example a missing plane in an imaging sequence, the object is shown in grey, as in the case of MR number 2 on the left. You will need to pay attention to this if you want to explicitly use this object and identify whether the error is an insurmountable problem or not.

By executing:

```
my.selection <- display.obj.links (pat, interactive = TRUE)
```

you have an interactive mode allowing a summary view of the contents of the objects, and if necessary, their (manual) selection which will be returned in the `my.selection` list.

By selecting “get info”, you get basic information about objects by clicking on them, for example a MR and and a rt-struct:

```
***** Get infos *****

        ***** mr 2 *****
object      : 20191003_a01ec436_ref3_do2_mr1
study date  : 20191003  ||      acq date : 20191003     ||      creation date :
description : IRM cérébrale |AX T1 SE FS GADO
FoR         : ref3
n.ijk       : 512 512 13
============================================================

        ***** rtstruct 5 *****
object      : 20200109_a01ec436_ref8_do15_rtstruct1
study date  : 20200109  ||      acq date :      ||      creation date : 20200123
description : PROTON|RS: Approved Structure Set
status      : APPROVED
FoR         : ref8
roi number  : 61
ATMD | ATMG | BandeletteOptD | BandeletteOptG | Bones  | CanauxsemicirculairesD | CanauxsemicirculairesG |
CerveletAnt  | CerveletPost | Chiasma-NOD  | Chiasma  | CochleeD | CochleeG | ConduitIntD  | ConduitIntG  |
CorneeD  | CorneeG  | CristalD | CristalG | CTV-VO(-NOD) | CTV  | CTVboost | CTVboost-VO+3mm  | CTVuVO | ENC-
ptv  | Encephale  | EXTERNAL | GTV préCT  | GTVréliquat  | gtvTEPprechim  | HippocampeD  | HippocampeG  |
Hypophyse  | Hypothalamus | LacrymalD  | LacrymalG  | Moelle | NC_V_D | NC_V_G | NC_VII_D | NC_VII_G |
NC_VIII_D  | NC_VIII_G  | NOD  | NOG-CTV1mm | NOG  | OeilD  | OeilG  | ParotideD  | ParotideG  | Patient-PTV
| patient  | Peau 3mm | PRVVOGetchiasma  | PTV  | RetineD  | RetineG  | Tronc  | TroncExt | TroncInt | Voies
Optiques - NOD

============================================================

***** quit *****
```

If you choose “select exams” and click on several exams, the function will return a data frame containing the selected exams. We will see that it is very useful to easily load these reviews, and only these. When selected, exams appear in bold.

3.2 Loading specific exams

For the time being, we have loaded the patient-oriented view, and we have manually selected the records we are interested in. We can now load these documents to work on them.

If we have decided that we are interested in the following: a CT, an rt-struct, an rt-plan and an rt-dose, it may be useful to save this information for later use (this is the quality control on the data that starts with the question "what data, ultimately?).

```
> my.selection
  PIN.patient                              PIN.exam     type idx
1    a01ec436        20200109_a01ec436_ref8_do2_ct1       ct   3
2    a01ec436 20200109_a01ec436_ref8_do15_rtstruct1 rtstruct   5
3    a01ec436   20200109_a01ec436_ref8_do13_rtplan1   rtplan   1
4    a01ec436   20200109_a01ec436_ref8_do11_rtdose1   rtdose   2
```

Once this is done, it is easy to load these files when needed:

```
CT <- load.obj.data (pat$ct[[my.selection$idx[my.selection$type=="ct"]]])
S <- load.obj.data (pat$rtstruct[[my.selection$idx[my.selection$type=="rtstruct"]]])
D <- load.obj.data (pat$rtdose[[my.selection$idx[my.selection$type=="rtdose"]]])
display.plane (CT)
```

Note that here we have assumed that there is only one exam per modality. If this is the case, this way of loading exams simply assumes their existence as we retrieve the index from the previously produced data frame.

To summarise, loading the exams is very simple with the following path: i) loading the patient-oriented view using `load.patient.from.dicom`; ii) selecting useful exams using `display.obj.links` or any other automatic method; iii) loading useful exams using `load.obj.data`. By this means, the data is correctly linked, in particular concerning the reference frames.

espadon also has a statement that allows documents to be loaded directly, without going through the patient-oriented view. This is `load.obj.from.dicom` (see the documentation on it), which operates in a similar way to `load.obj.data`. Note, however, that bypassing the patient-oriented view requires a reference frame name to be manually assigned when using `load.obj.from.dicom`.

4 Rdcm format for fast and light DICOM handling

The DICOM format is particularly demanding in terms of data volume and number of files. For simple (one-shot) studies, this is not fundamentally a problem. However, in complex studies, where the same files are returned to multiple times, it may be worthwhile to have more efficient methods than systematically re-reading the whole data set to get a synthetic view necessary for their exploitation. For this purpose, espadon has a specially developed file format which we have called Rcdm. The Rdcm format will bring together all the files corresponding to an examination into a single object, thus reducing the volume of files handled. It also incorporates easy access to important connectivity data to speed up loading. Finally, the data is highly compressed to save disk space.

In summary, if you are doing one-shot studies, you can probably skip this chapter. If you have long and complex studies, you should probably read it carefully so as to save your time in accessing information.

4.1 DICOM to Rdcm conversion

To convert DICOM data to Rdcm, you must first build a directory that will hold the Rdcm. Then execute the following lines:

```
pat.src.dir <- choose.dir ()
dcm.filenames = list.files (pat.src.dir, recursive = TRUE, full.names = TRUE)
pat.dest.dir <- choose.dir ()

start.time <- Sys.time()
dicom.to.Rdcm.converter (dcm.filenames, pat.dest.dir, update = TRUE)
Sys.time() - start.time
```

The conversion work is done by the `dicom.to.Rdcm.converter` instruction. For the case illustrated above (multiple reviews), it converted the original 3712 files, occupying a volume of 1.58 Gb, into 42 files for a total volume of 577 Mb, i.e. a gain of 3 in storage volume. Note that the compression is done without loss of information. The conversion took about 3 minutes.

The `update = TRUE` option allows you to keep the reference frame names that may have been assigned during a previous conversion. If you have a new DICOM exam in the source directory, it will be converted to Rdcm, as it should be, but the exams already known in the destination directory will keep their original reference frame names. So if you have already performed quality control operations on the Rdcm data, it will remain valid, despite the fact that you have added a new exam.

4.2 Rdcm handling

Once converted, the Rdcm format is used exactly like the DICOM format. The patient-oriented view is available by means of the instruction `load.patient.from.Rdcm`. Note that the order of exams is not necessarily the same when using `load.patient.from.Rdcm` or `load.patient.from.dicom`.

You can load exams data using `load.obj.data` directly from the patient-oriented view, or using `load.obj.from.Rdcm` and specifying the Rdcm file you need.

In terms of execution time, loading a patient-oriented view takes 1.9 minutes in DICOM, and 0.1 seconds in Rdcm. the complete loading of a CT (512 x 512; 557 slices) takes 17 seconds in DICOM and 6 seconds in Rdcm.

5 What we learned

Espadon provides several ways to load DICOM depending of the complexity of your problem. In any case, **a good practice consists in creating a folder for the study, and inside this folder, having as many folder as patients in this study**. Each patient folder can be organized as you want (all DICOM at the same level, or stored into sub-folders by modalities, for instance).

- For “simple” studies, where DICOM files are well defined (each patient has the same organization, for instance MR + Reg + CT + rt-Struct + rt-Dose), the simplest way is to use `load.patient.from.dicom` instruction. It loads all the DICOMs into memory and provides the patient oriented view.
- For “complex” studies, where patient data can be numerous and not well organized, we recommend to convert DICOM files into Rdcm format using `dicom.to.Rdcm.converter`. Then,

`load.patient.from.Rdcm` provides you the POV where you can select the appropriate files to download using `load.obj.data` instruction.

At this point, you will be able to use espadon functionalities for imaging, contouring and dosimetry.

If espadon does not manage a DICOM modality, or if you need specific information stored into you DICOM file, you can access it using `dicom.raw.data.loader` followed by `dicom.parser` for explicit conversion into a simple data frame. See the appropriate document for more information about this functionality.

Last but not least, don't forget to have a look at espadon manual to discover instruction arguments that could help you.

You can see also:

dicom.browser	**DICOM raw data browser**
dicom.parser	**Conversion of DICOM raw data into a dataframe or a list of DICOM TAG information**
dicom.raw.data.anonymizer	**DICOM anonymizer**
dicom.raw.data.loader	**DICOM file loader in raw data**
dicom.set.tag.value	**Change TAG value in DICOM raw data**
dicom.tag.dictionary	**DICOM TAG dictionary**
dicom.tag.parserDICOM	**TAG parser**
dicom.to.Rdcm.converter	**Conversion of DICOM object into files that can be interpreted by the espadon package**
load.obj.data	**Load data of an espadon class object**
load.obj.from.dicom	**Loading an espadon object from DICOM file**
load.obj.from.Rdcm	**Loading an espadon object from *.Rdcm file**
load.patient.from.dicom	**Loading patient data from DICOM files**
load.patient.from.Rdcm	**Loading patient data from *.Rdcm files**
load.Rdcm.raw.data	**Loading a *.Rdcm file**
load.T.MAT	**Loading of information about transfer matrices between frames of reference of a patient directory**
save.T.MAT	**Save a T.MAT class object**
save.to.Rdcm	**Save a espadon object in a pre-formatted *.Rdcm file**
xlsx.from.dcm	**Converting DICOM files to .xlsx files**
xlsx.from.Rdcm	**Converting .Rdcm files to .xlsx files**