Text Analysis Tools (TAT)

Andreas Fischlin

The following describes TAT v1.0fc9 going together with REtool v1.8fc9

Table of Contents

Why TAT?

The purpose of TAT is to extract from an IPCC draft, typically made available in form of a PDF (or perhaps also in form of a Word file), the meta data on the draft's content. Text processing is in general already quite complex, which is of course even more true for scientific texts such as IPCC reports. During review rounds it is necessary to reliably reference text passages, which is typically done by page and line numbers. The latter characterize a draft in contrast to the final product no longer having line numbers. The meta data TAT extracts are page and line numbers for every element contained in IPCC drafts, i.e. figures, tables, boxes, box figures, box tables, FAQs, footnotes, and paragraphs.

Unfortunately todays text processing tools, notably Word, do not provide the means to reliably extract page and line numbers as needed for referencing text portions or other elements of a draft. These numbers are, however, of course quite essential, notably during the critical review rounds by IPCC. Note, even if Word would provide such a feature or at

least allow to directly extract the wanted information through VBA or similar means (e.g. AppleScript), it might not work always reliably unless done on the very same system on which an IPCC draft PDF is produced, since page and line breaks are determined by Word at the moment of generating the PDF and may depend on the fonts used and perhaps even the precise font versions present during the rendering. Thus elaborate evaluation of many techniques showed that not the Word file, but the actual PDF generated provides the most reliable source of the wanted meta information.

The solution to all these technical obstacles is TAT. TAT can extract the needed information on page and line numbers for headings, figures, tables, boxes, footnotes, and many other parts of an IPCC draft in a reliable manner if the PDF has been properly generated[1]. It does that by extracting the wanted information from any PDF containing exactly page and line numbers as used by IPCC authors, reviewers, and IPCC Review Editors (RE). Extraction comes first in form of text files (Step 1, see below), which are then stepwise filtered, modified, and imported into a data base (TAT-drafttext) and there processed until a spreadsheet can be exported from the final data base (TAT-draftTOC). The result is a table, a large Table Of Content, listing all elements of the original draft text's elements, but in contrast to an ordinary table of contents, not only with the associated page numbers, but also with line numbers. E.g. you know where to find a figure or on which page at which line a section starts. In principle TAT can go so far to even extract page and line numbers of any draft element, even paragraphs or individual words, should that be of interest.

Notably page numbers and line numbers are of great interest to anyone involved in the IPCC review rounds. It is in particular also needed by REtool to support correct display of text pieces to which a comment refers and to conveniently jump to those locations. Note, however, not only Review Editors, but also authors or TSU can analyze and compare drafts easier using TAT.

The following help file contains rough instructions on how to use TAT. This is done with the view to support first of all REtool, a tool, which can greatly facilitate to work of a Review Editor (RE).

Note also, as of this writing all TAT software as described here has been tested only on Mac OS X systems. However, on any UNIX based platform where FileMaker is also available, TAT should run fully. This excludes Windows partly, since the first part of TAT makes heavy use of UNIX tools such as awk and shell scripts. They are needed for the first processing of the text files involved in the extraction of the meta data. Fortunately, TAT processing can also be done stepwise on several systems. E.g. the Unix dependent parts (Steps 2, 3, and 4, see below) can be done on any UNIX system, while all other steps as the second FileMaker dependent steps 5 till 9 (see below), which no longer require Unix features, could then be done on Windows and/or Macintosh systems.

The following consists of five parts:

(i) How to produce an initial text file from the original pdf (Note, there seems to exist no reliable method to work from Word files themselves due to Word's limited functionality).

---

[1] This means that no text has been converted into a bitmap graphic. Unfortunately unless the generation of the PDF is carefully done, Word may easily generate in this sense corrupted PDF's, notably under Windows.

(ii) How to process the thus obtained text file to prepare it for import into REtool data bases and how to process the meta data within REtool data bases to obtain the final result (in form of a metadata spreadsheet)
(iii) General rules to observe and use of draft specific parameters
(iv) Installation hints
(v) Examples of the use of TAT

Note, if you are not using REtool, TAT allows you to obtain as an end result of above processing an Excel spreadsheet that can be used otherwise (independent of REtool).

1 Methods of producing from the IPCC draft PDF a text file

To accomplish the first critical step, it is recommended to work first of all with the text extracting method described below (called b9) for historical reasons) (This is a method, which, as of this writing, is tested to work well and reliably on the Macintosh platform):

1) Open the pdf in Acrobat and make sure you have continuous view active (menu command “View -> Page Display -> Enable Scrolling”)

2) Select all (menu command “Edit -> Select All”)
3) Copy into the clipboard (menu command “Edit -> Copy”) — This process may take a while with large documents; wait until Acrobat is really finished.
4) Open in TextEdit (standard application under OS X) a new document (menu command “File -> New”) and make it text only (menu command “Format -> Make Plain Text”, allow for it and click “OK”)
5) Paste the large clipboard into the empty new TextEdit document (menu command “Edit -> Paste”)
6) Save the document with UTF-8 encoding (menu command “File -> Save” with “File Format”)


The following methods have also been tested, but generally with strongly varying results of usability:

b1) Using Preview, select all, and then copy paste everything into a text file
b2) Using Acrobat “File -> Save As Other… -> More Options -> Text (Accessible)”
b3) Using Acrobat “File -> Save As Other… -> More Options -> Text (Plain)”
b4) Using Acrobat “File -> Save As Other… -> More Options -> Rich Text Format”
b5) Using Acrobat “File -> Save As Other… -> HTML Web Page”
b6) Using Acrobat “File -> Save As Other… -> Microsoft Word -> Word Document”
b7) based on b6) Using Word “File -> Save As…” select “Plaint text (.txt)” with options ‘Other encoding: Unicode 6.1 UTF-8’ and check ‘Insert line breaks’ (CR only)

b8) Using Acrobat “File -> Save As Other… -> Image -> TIFF” and open all tiff files in Acrobat to save them again as a PDF, OCR, all then copy paste everything into a text file
b9) *is actually the method described above*
b10) Using Word (original file) “File -> Save As…” select “Plaint text (.txt)” with options ‘Other encoding: Unicode 6.1 UTF-8’ and check ‘Insert line breaks’ (CR only)
b11) Using Word from Office 2016 to open the pdf and continue with method b6) or b7)

Any of these methods is only recommended if you should encounter difficulties with the recommended method b9). Note, the recommended method does only work well if religiously followed. E.g. altering the sequence of the steps 4) and 5) slightly by first pasting and converting the file to text only after the paste does generally NOT work and may easily result in corrupted, i.e. truncated data!

2 Processing the data (core of TAT)

It is recommended to always perform following steps:

1) Use the distributed pdf with method b9) (see above, part (i)) and paste the possibly large clipboard into a new TextEdit file, save as text only UTF-8. IMPORTANT NOTE: Do first make the TextEdit file a text only file before you paste the large clipboard. Otherwise, i.e. if you paste into a rtf TextEdit file, you risk to get only a portion of the document. Such a portion regularly ends after a table. However, if the receiving TextEdit file is a plain text file before you paste, you should get the entire document in one go.

2) Prepare file ‘fixDraftPars.txt’ and make sure a copy resides in the same folder as where the result from step 1 resides

3) Run shell script ‘fixDraft.sh -f <filename>.txt’ (the options may depend on the case and method used in step 1), consult help of fixDraft.sh accordingly and explanations below) (*)

4) Possibly repeat step 3 until all looks ok, which may require some editing of the file resulting from step 1. Editing is typically required if an author has mistyped the label of a figure or another item. E.g. ‘Box 1, Figure 1’ is not correct, since the chapter is missing. Correct would be ‘Box 3.1, Figure 1’, since we talk about Box 1 within Chapter 3. You may do such editing either manually with any text editor such as TextEdit or then using an additional shell script. I greatly prefer the latter. This consists of an optional file named ‘fixTypos.sh’, which resides in the same folder where the result from step 1 resides. Repeating step 3 means then you edit ‘fixTypos.sh’ until step 3 results in a flawless text (for more details on this technique see Examples below).

5) Open ‘TAT-drafttext.fmp12’ and import the file resulting from steps 3) and 4) (FM script ‘Import draft text...’)

6) If you do it chapter by chapter, make sure the value global report page number (field 'First_page_of_draft_core') is correct before going to the next step

7) Run first a preparatory FM script and then the actual parsing FM script 'Parser - TOC' (first time Replace, later Append) in 'TAT-drafttext.fmp12'. The standard preparatory FM script is 'Prepare for TOC Parsing' (but there are others described below under Examples).

8) Repeat from steps 3) till 7) as necessary (Append) until the TOC in 'TAT-draftTOC.fmp12' looks good and jumping to the PDF works correctly

9) Run FM script 'Export to spreadsheet' in 'TAT-draftTOC.fmp12' to obtain the metadata, which you can then import into 'REtool-textlinks.fmp12' for use with 'REtool-main'

That's all folks!

(*) In step 2 I like to use following variant 'fixDraft.sh -f <filename>.txt ; open <filename>-OUT.txt' for immediate checking of the result while iterating between steps 3) and 4).

The following explanations can be easily skipped if you are mostly interested in the basics of TAT. Skip with your reading to the next part **(3)** «General rules to observe and use of draft specific parameters».

On Steps 1,2,3, and 4): TAT is able to process an entire report with front matter (FM), several chapters, and back matter (BM) (e.g. an appendix with the actual figures) in almost one go. The recommended method for step 1) is now reliable if done exactly as described in part **(1)** and the following considerations are here mentioned only in the unlikely case things do not work as expected.

First note, resorting to any other technique for step 1) (cf. part **(1)**) is still possible and a supported option if something should go badly wrong with the recommended method (in all these cases use 'fixDraft.sh -9). However, in my experience any other method will require considerably more fixing in step 4) until the result is satisfying, which may easily require a huge amount of work possibly defying the entire exercise to automate the extraction. The sequence of steps remains basically the same, only parameters of 'fixDraft.sh' need to be adjusted (e.g. do use option -9 if you have NOT used method b9) etc). If the pdf contains no text, I would suggest to use then first OCR, e.g. within Acrobat, and then proceed by the recommended method b9) as usual. In such cases, however, following variations of step 3) may then be helpful as an example for varying step 3):

```
cat <filename>.txt | fixDraft.sh -d | fixOCRissues.sh | repairLineNos.awk | insertTabDels.awk >
<filename>-OUT.txt ; open <filename>-OUT.txt
```

Note, in above commands the option -d does suppress the call to insertTabDels.awk by fixDraft.sh, which you then need to call later yourself (in general best called as the last step). You may freely experiment with many options for fixDraft.sh -dgnt to suppress further calls to awk scripts by fixDraft.sh (-g suppresses call to rmFigures.awk, -t that to rmTables.awk, -n that to markFootnotes.awk) allowing you to alter the sequence by which any of these scripts is called and possibly call even one of these scripts several times. Alternatively to above

command sequence you could also call fixDraft.sh with option -o (asks ‘fixDraft.sh’ to try to fix OCR issues) with following command sequence:

```
cat <filename>.txt | fixDraft.sh -o > <filename>-OUT.txt ; open <filename>-OUT.txt
```

Experimenting with the sequence and the number of times a fixing script is called may help in difficult cases. Remember just the following when using fixDraft.sh:

- -z This option should always be used if line numbers do not restart with 1 at every page
- -9 This option should always be used if you have NOT used method b9)
- -i This option should be used with caution and is known to give questionable results (resort to it only if any other method that can capture page footers and headers really should be impossible)
- -s This option is normally not needed and mostly helps to debug particular difficult cases in combination with option -z

Then you need to be aware also of the fact that the parameters as used by the awk scripts are not read from the file ‘fixDraftPars.txt’. This is done by ‘fixDraft.sh’ only, which then adjusts the calls to the awk scripts accordingly (one of the bigger advantages of ‘fixDraft.sh’). If you need to deviate from the ‘fixDraft.sh’ default sequence of calling awk scripts, the results may be suboptimal unless you really use consistently the proper parameters. E.g. insertTabDels.awk with option -z needs be called with parameter -v maxLineNo=32767 (or any large integer number larger than the highest line number in the report). The call to insertTabDels.awk must then be as this

```
cat <filename>.txt | fixDraft.sh -d | fixOCRissues.sh | repairLineNos.awk | insertTabDels.awk -v
maxLineNo=32767 > <filename>-OUT.txt ; open <filename>-OUT.txt
```

Note also, all default parameters used by the awk scripts are those compatible with a FOD, e.g. FOD of IPCC SR1.5. Otherwise additional parameters may need to be added. E.g. when calling insertTabDels.awk an additional parameters such as the headerToken (-v footerToken=“Second Order Draft Chapter”) is best added. In difficult cases you may need to open the awk scripts to see which parameters can be overwritten by the callee and adjust the call accordingly. Remember, while the sequence by which you call awk scripts is quite open to experimentation and may be well what you need in difficult cases, consistency among all parameters as used by the callee is most important for optimal results. Otherwise you can be easily mislead to misjudge the relevance of a change in the calling sequence of awk scripts or their capability to help you in general.

On Steps 5,6,7, and 8): TAT offers also the big advantage of exploiting redundancy. Any Table of Contents (TOC) information can be used to double-check. E.g. if the front matter of a report contains a TOC, this can be processed also, even using method b10) in the most straightforward manner and then proceed with steps 5) on to feed at least some of the TOC data into ‘TAT-draftTOC.fmp12”s table ‘Table_of_Contents’ (flag in field ‘Tlob_Pars::Text_is_only_TOC’ from ‘TAT-drafttext.fmp12’ set to TRUE). Of course line numbers are then always missing. But, iterative refinement of the TOC data is then possible while executing FM script ‘Parser - TOC’ from step 7) always in Append mode using different sources such as the actual chapter (flag in field ‘Tlob_Pars::Text_is_only_TOC’ from ‘TAT-drafttext.fmp12’ set to FALSE) in addition to a mere TOC input.

Such an approach allows also to detect inconsistencies, e.g. I detected that the TOC listed a subsection that was actually not present in the ZOD text of chapter 3 from IPCC SR1.5. E.g. the heading «3.2.4.1 Identifying hot spots» is in report wide TOC in the front matter (FM), but is missing in the chapter itself. Processing a mere TOC does not necessarily require to run the full script 'fixDraft.sh'. It may suffice to execute following commands (example of report wide TOC in the FM of the ZOD SR1.5 with a couple of minor edits manually done on the initial text file 'ZOD FM-b1.txt' as resulting from method b1):

```
cat "ZOD FM-b1-ES3 edited-FM deleted.txt" | massageDraft.sh | insertTabDels.awk > "ZOD FM-b1-ES3
edited-FM deleted-OUT.txt"
```

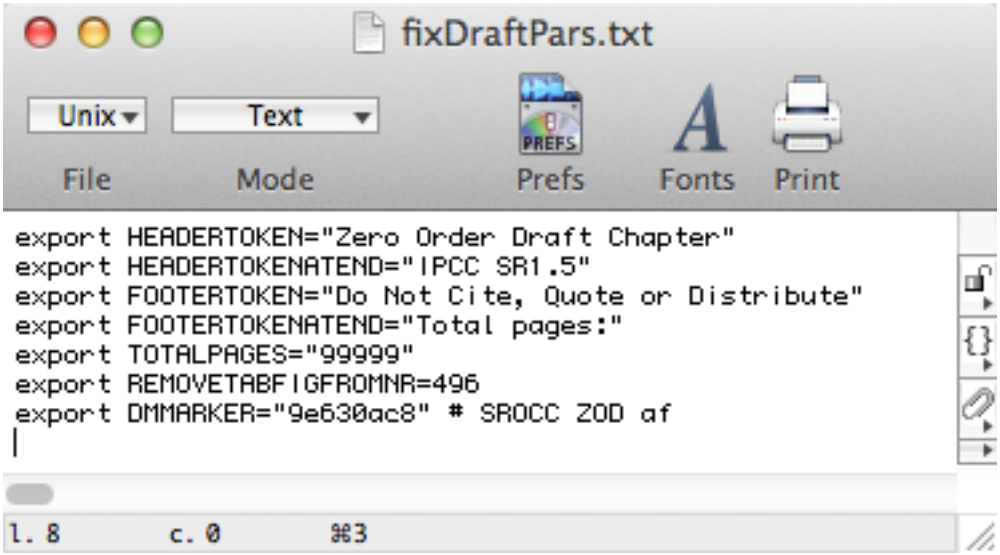
Finally note also that several FileMaker scripts in 'TAT-drafttext.fmp12' can be executed also in a modular fashion, e.g. 'Renumber pages' or 'Identify front (FM) and back (BM) matter'. They generally produce a report of their action in field 'Glob_Pars::LogReport'). Alerts are only issued if a warning is inevitable or otherwise clear inconsistencies threatening the quality of the final results have become obvious. A handy FileMaker script "Toggle layouts 'Text' and 'Parameters'" (Cmd^0) allows to quickly switch between the text view (layout 'Text') and the parameter settings (layout 'Parameters'). Since Step 8) can be repeated in mode "Append" as many times you wish, the quality of the result in TAT-draftTOC can be enhanced stepwise easily and conveniently, possibly even going back to previous steps such as Step 4) by editing 'fixTypos.sh'. In my experience it is actually often the case that you notice only in the final TOC as shown in TAT-draftTOC that some chapters were mistyped by the authors and require fixing before the heading can really be recognized by the parsing as done in Step 7). Without the fix an important chapter or section heading or figure may simply be missing. Similarly correctness of page or line numbers may need testing by trying to jump to the wanted PDF's locations. The latter may be particularly relevant if you needed to resort to OCR in Step 1), which may easily have confounded page or line numbers (e.g. '25' as '2S' resulting in a wrong or even malformed page number).

3 General rules to observe and use of draft specific parameters

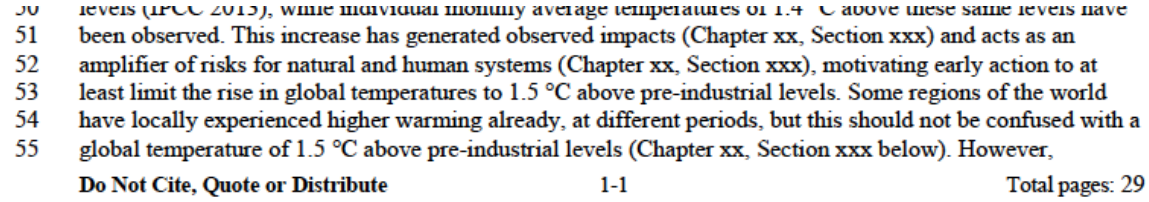
Above procedures can be modified as needs arise, yet following few rules are best observed always:

A) Favour any method for Step 1) that allows to preserve page footers and page headers. The critical pagination as done in Step 5) works otherwise not reliable. This means only methods b1) or b9) should be used for complex texts (excluding methods b2), b3), b4), b5) and b7), assuming b10 can never be used anyway (unless line numbers would not matter, but which would defy the entire purpose of TAT except for simple table of content analysis, see Example 3 below).

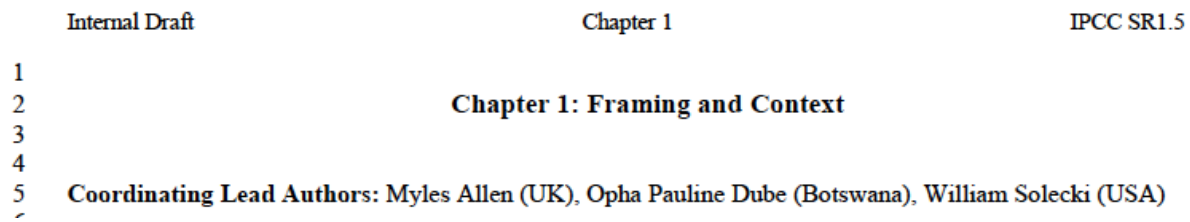
B) Every draft comes with slightly different formatting, notably different page headers and footers. TAT offers a convenient technique to provide the draft specific tokens that are essential and critical for TAT to perform its tasks. It is recommended to specify the draft specific parameters in a simple text file, which is always named 'fixDraftPars.txt'. The following example of this file is for the IPCC SR1.5 ZOD draft:



This file is automatically used by 'fixDraft.sh' (see Step 3) above, if present at all. Currently this file needs to be setup manually with a text editor. Note, spelling is critical and fundamental, to have an Oxford comma or not, makes all the difference, e.g. the sentence expected here to be written in each page footer, i.e. "Do Not Cite, Quote or Distribute" vs. "Do Not Cite, Quote, or Distribute". Depending on the draft, TAT fails most likely entirely or succeeds, depending whether the Oxford comma is actually present in the original PDF or not! The string value that is assigned to the variable FOOTERTOKEN (footer token) needs to precisely specify what is actually written in the original PDF, here i.e. "Do Not Cite, Quote or Distribute". Here a snapshot from the original PDF:


levels (IPCC 2013), while individual monthly average temperatures of 1.4 °C above these same levels have
been observed. This increase has generated observed impacts (Chapter xx, Section xxx) and acts as an
amplifier of risks for natural and human systems (Chapter xx, Section xxx), motivating early action to at
least limit the rise in global temperatures to 1.5 °C above pre-industrial levels. Some regions of the world
have locally experienced higher warming already, at different periods, but this should not be confused with a
global temperature of 1.5 °C above pre-industrial levels (Chapter xx, Section xxx below). However,
Do Not Cite, Quote or Distribute 1-1 Total pages: 29

A comment on the header matching above 'fixDraftPars.txt'. This is a header from the original PDF:


Internal Draft Chapter 1 IPCC SR1.5

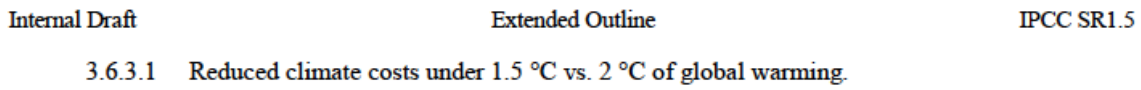
Chapter 1: Framing and Context

Coordinating Lead Authors: Myles Allen (UK), Opha Pauline Dube (Botswana), William Solecki (USA)

The original PDF uses in the header not "Zero Order Draft", but "Internal Draft" followed by the word "Chapter". Yet the value assigned to parameter HEADERTOKEN (header token) in above example is "Zero Order Draft Chapter". Why? First note, most text extracting methods from Step 1) (see also part **(2)**) insert a single blank between "Draft"

and “Chapter”. Secondly, for technical simplicity reasons TAT prefers to have always a three word phrase in headers such as “Zero Order Draft”, “First Order Draft”, and “Second Order Draft” etc. TAT (as well as all of the REtool software) also uses consistently the abbreviations ZOD, FOD, SOD, and FGD throughout. Therefore TAT translates any “Internal Draft Chapter” string into “Zero Order Draft Chapter” (actually done by shell script ‘massageDraft.sh’ as also always called by ‘fixDraft.sh’). Therefore the value assigned to parameter HEADERTOKEN in above example is “Zero Order Draft Chapter” and is needed to successfully process the IPCC SR1.5 ZOD PDF. Similar arguments apply for the parameter HEADERTOKENATEND (header token at end), which must exactly match what TAT sees, here exactly what can be found at the end of each page header of the original PDF. Similarly TAT may need to be informed what can be found at the end of each footer of every page (parameter FOOTERTOKENATEND, footer token at end). Note the latter does on purpose not use the chapter specific number “29” or TAT would only work for chapter 1 of SR1.5 ZOD and would fail for every other chapter or draft where the chapter 1 has another total number of pages. While in above example parameter TOTALPAGES (total pages) does allow to assign the precise value, this value matters only in cases where TAT is asked to regenerate footers lost by a text extracting technique such as method b2) etc. Otherwise TAT ignores the highly variable number “29” and recognizes or fixes a footer only by looking for the string tokens as assigned to the parameters FOOTERTOKEN and FOOTERTOKENATEND. Consequently, the parameter TOTALPAGES is optional and could be missing from above file, yet TAT would still successfully process the SR1.5 ZOD PDF.

Finally note also, the pages in the front matter of the SR1.5 ZOD are not meant here. On the contrary. Those are different and this fact allows TAT to differentiate between front matter and the actual core of the draft where the chapters are. Here the aforementioned page header illustrating this difference:



Note, the words “Chapter 1” vs. “Extended Outline” make all the difference, a difference that is critical for TAT to succeed and work well throughout.

Please watch out for the following caveats also:

(i) ‘fixDraftPars.txt’ must be a syntactically correct text file that can be successfully sourced (by Unix shell script ‘fixDraft.sh’). This requires in particular to not change any of the variable names used. Most importantly, it also requires to always enclose any text containing a blank within double quotes as shown above.

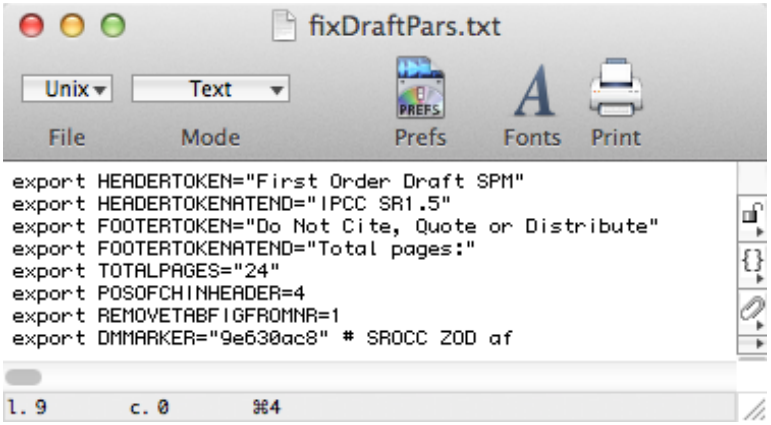
(ii) There exist currently no techniques to make sure the parameters as used in this text file are identical to those used in TAT-drafttext.fmp12 table ‘Glob_Pars’. These parameters need to be consistently defined and therefore need typically

to be checked and possibly adjusted when one switches from a draft to another, e.g. from ZOD to FOD etc.

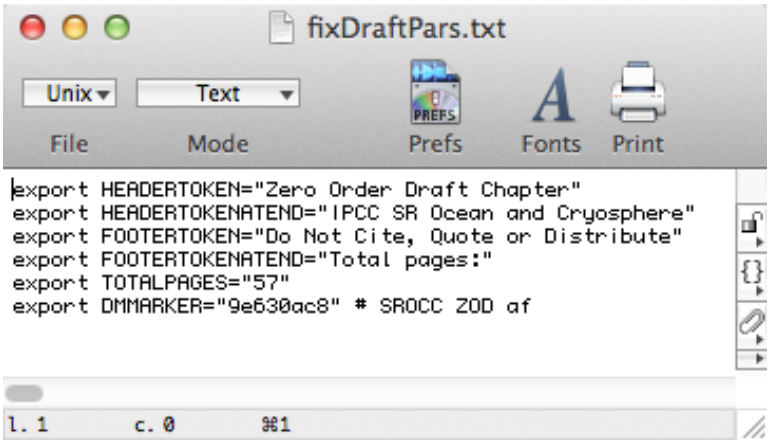
(iii) Note, if any of the parameters shown above are not contained in this file, default values are used by 'fixDraft.sh' and/or the called awk scripts (see e.g. TOTALPAGES as discussed above). Most defaults used by TAT are those needed for the SR1.5 FOD, which could be processed without any 'fixDraftPars.txt' file at all.

(iv) Parameters can also be defined in any sequence, since above files are simply sourced by 'fixDraft.sh' if available before any values are used.

Following examples show how to change file 'fixDraftPars.txt' for IPCC SR1.5 SPM FOD



or IPCC SROCC ZOD Chapter 1



C) If a report wide front matter (FM) is present, e.g. as with SR1.5 ZOD, then it is recommended to not use fixDraft.sh -c to allow 'REtool-textdraft.fmp12' to correctly determine the number of pages in the FM. This is particularly important in cases where no reliable headers and footers are present in the original input file, e.g. methods b1) or b7). Otherwise subsequent paginations in 'REtool-textdraft.fmp12' will not succeed to assign correct page numbers. On the other hand, since the main purpose of TAT is to reconstruct a detailed TOC (Table Of Content), the original TOC is not really needed, since mostly redundant (but see Example 4 below).

D) For fixDraft.sh -i to succeed always, line numbers need to restart within each page and every page should start with line number 1. If that is not the case, then a method needs to be used that preserves page headers and footers in a reliable manner (see rule A)

E) Page numbers are critical, notably if they are to refer to a single global report pdf (e.g. SR1.5 ZOD) for the jumping to locations to work always properly. If working piecewise, e.g. chapter by chapter, you need to understand the following:

   (i) The chapter page numbers are assigned automatically in 'REtool-textdraft.fmp12' (field 'Text::PageNo') right after the import (FM script 'Renumber pages'). This assumes by default each chapter to start from page 1.
   (ii) Yet, report wide global pages are assigned only in 'REtool-textTOC.fmp12' during the execution of FM script 'Parser - TOC' in 'REtool-textdraft.fmp12'. They use an offset from all previous material (field 'Glob_Pars::First_page_of_draft_core' in 'REtool-textdraft.fmp12'). In such a case make sure you either work one chapter after the other and determine the values of that parameter as obtained from processing the previous chapter or you look it up manually in the original pdf. Such hassle can be avoided by processing all chapters in one go, the recommended method.

4 Installation Hints

For TAT to work, you need only to copy a few files to your system (exception is FileMaker, which is needed as well for a fully functional TAT, see below).

4a) Unix dependent part

The following shell and awk scripts are needed as released within the REtool package 'Text Analysis Tools' (TAT). They are best copied or moved to a folder or directory [2] called

```
bin/TAT
```

in your home directory (if the folder does not yet exist, please create it first) (note this makes no sense under any non Unix system):

```
fixb9-a.awk
fixb9-b.awk
fixb9-c.awk
fixb9-d.awk
fixb9-e.awk
fixFooters.awk
fixHeaders.awk
fixMarkedFootnotes.awk
fixOCRissues.sh [3]
insertPageBreaks.awk
insertTabDels.awk
```

[2] Unix terminology

[3] helpful only if using method b8 or OCR in combination with method b9

```
markFootnotes.awk
massageDraft.sh
repairLineNos.awk [4]
rmDMHeaders.awk
rmFigures.awk
rmTables.awk
rmTOC.awk
splitLineNoRows.awk
```

and the key to all above scripts, the shell script:

```
fixDraft.sh
```

For TAT to be fully functional on your system It is critical that the so-called environmental variable $PATH contains the directory ‘bin/TAT’. The following steps need to be done only once, i.e. before very first use of TAT.

On the Macintosh platform test this by opening application Terminal [5], which you find within your system wide Applications folder. There type

```
echo $PATH
```

followed by RETURN (to issue the actual command). The result may be a string similar to the following:

```
.:./bin:/Users/<yourUserName>/bin:
```

This is a list of directories, separated from each other by a colon ‘:’ (without any blanks), in which Unix command line tools are searched. For TAT to work it is very critical that the ‘bin/TAT’ directory into which you have copied above shell and awk scripts is known to the system via this environment variable $PATH. Otherwise TAT will not work. First, make sure that the ‘bin/TAT’ directory is listed as shown in the example above, where the path

```
/Users/<yourUserName>/bin
```

with <yourUserName> replaced by your actual User Name points to the wanted ‘bin’ directory (Note, <yourUserName> is the name your home directory).[6] If above checking does show that the ‘bin’ directory is NOT yet contained in the $PATH environment variable – typically the case at the very first installation of TAT – you need to fix this. On the Macintosh platform this requires following two simple steps:

1) Open the so-called user profile, this is a normally invisible file named `.profile`. Open this from the Terminal with following command

```
open .profile
```

The file may contain already a definition for the $PATH variable. A safe technique is then to leave that definition as is and to merely add the definition ~/bin/TAT after the first definition of PATH:

---

[4] `needed only if using method b8 or OCR in combination with method b9`

[5] available in the Utilities folder of your system, i.e. directory /Applications/Utilities

[6] Unix allows to abbreviate your home directory by a simple tilde. Thus ‘/Users/<yourUserName>/bin/TAT’ is equivalent to ‘~/bin/TAT’.

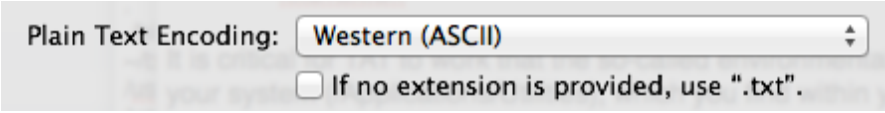
```
export PATH="$PATH:~/bin/TAT"
```

This will append to the existing $PATH the wanted custom path /Users/<yourUserName>/bin/TAT (here given as ~/bin/TAT) and will then export that new environment PATH variable to the Unix environment as used by the Terminal and TAT. [7]Once you have added above second line, do save and close the file.

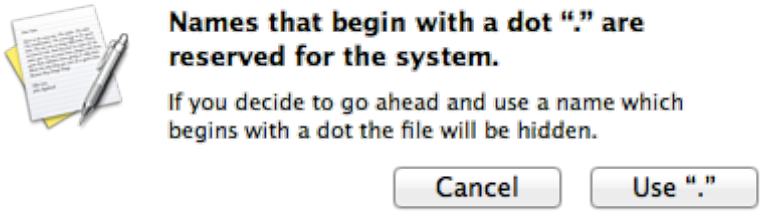
If you get the response that the file does not exist, you need first to create it. Launch TextEdit, either the usual way or why not e.g. with following command executed from the Terminal window:

```
open -a TextEdit
```

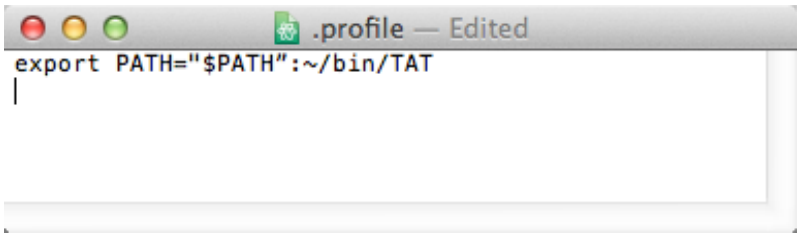
Once in TextEdit use menu command “File -> New” and save the file as text only to your home directory with ASCII encoding:



Then click button ‘Use “.”’ in the subsequent warning alert, because you want to provide a system reserved file:



Finally enter the wanted PATH definition needed by TAT into this file as shown below. The new file should then look similar to this:

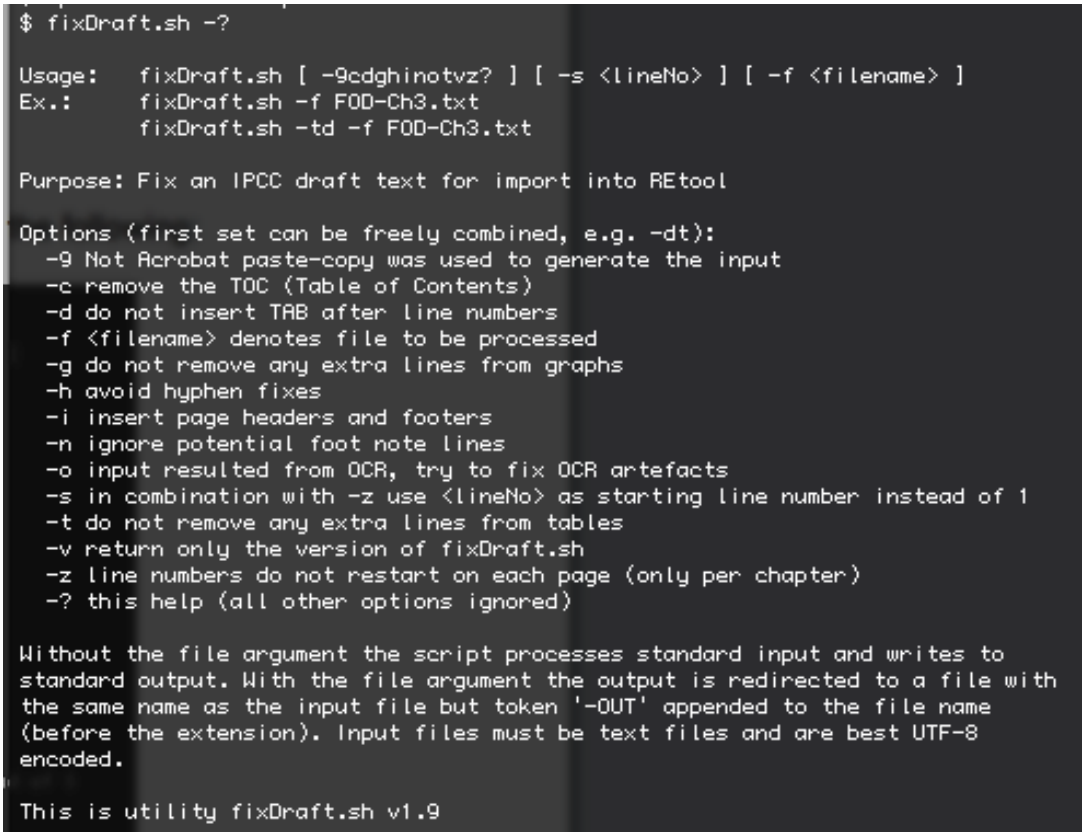


[7] If you use also REtool you should also make the path ~/bin known to your system. The line ‘`export PATH="$PATH:~/bin/TAT"`’ should then be written as ‘`export PATH="$PATH:~/bin:~/bin/TAT"` (without single quotes of course).

2) Relaunch application Terminal[8] and test again by executing the command echo $PATH. Repeat the two steps until you can see in $PATH the wanted path. Test whether all works by entering e.g. following command requesting fixDraft.sh to provide its help:

```
fixDraft.sh -?
```

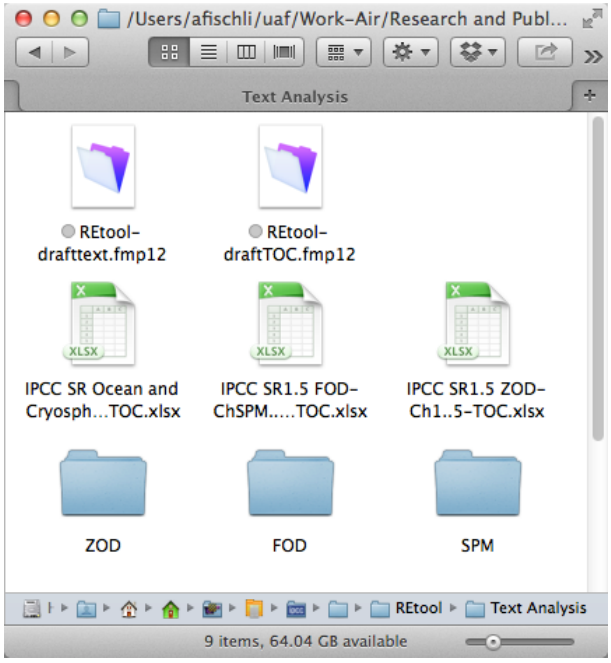
You should get the help from TAT utility fixDraft.sh similar to the following:



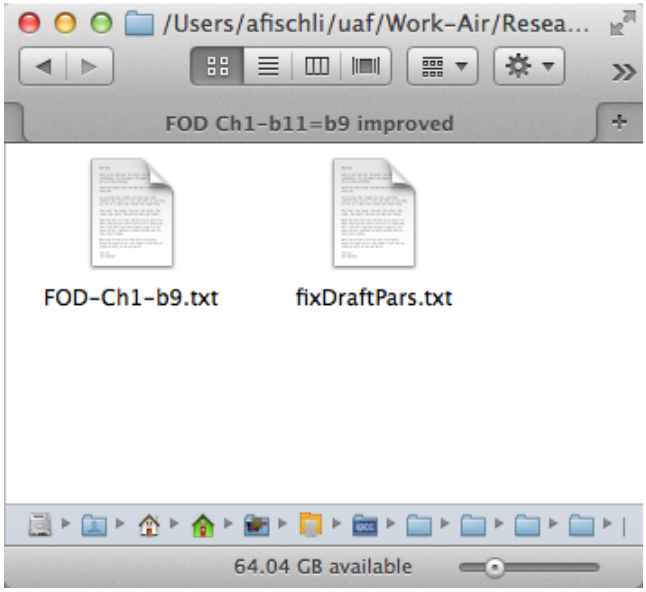
4b) FileMaker dependent part

For the processing relying on the data base files ‘TAT-drafttext.fmp12’ and ‘TAT-draftTOC.fmp12’ (Steps 5) .. 9)), simply copy those files into any working folder you like. However, I recommend to use another folder than the one where you prepare the input files for ‘TAT-drafttext’ as described under Steps 1) to 4). This is because you may need for every draft text you wish to process a separate copy of ‘fixDraftPars.txt’, ‘fixTypos.sh’. I use on my system a TAT home folder, here for the IPCC SR1.5:

[8] This is quite important to make sure that your terminal session does update the environment variable $PATH. A simple way to do that is to quit and relaunch the application Terminal.



Within the ZOD folder I then have working folders for individual chapters, e.g. for SR1.5 FOD Chapter 1:



Finally you need as of this writing also FileMaker Pro version 14 or later (Note, all testing described as of this writing was done with FileMaker Pro Advanced v14.0.6 under OS X 10.9.5 (Mavericks)). I will perhaps make versions available that will NOT depend on having a FileMaker Pro license available. Ask me if you are interested.

No further installation steps are needed in addition to what was described above.

5 Examples

The following examples represent tutorials, which start with a simple case, introducing with each example more advanced techniques.

All examples were done using Acrobat XI (11.0.23) and TextEdit 1.9 (310) on a Macintosh under OS X 10.9.5 (Mavericks). They should work with other Software versions on any other Macintosh system. Some parts of TAT should work under Linux, but nothing was tested. TAT does not work under Windows, since Windows is not Unix based. Yet, the REtool FileMaker data bases would work also under Windows, given the needed input files are prepared on a Unix system. Input files are available from folders nearby. The expected output files are also provided for comparison reasons.

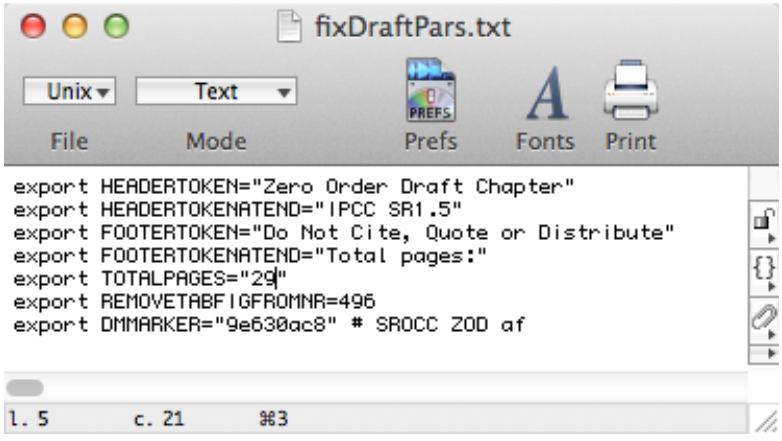
Example 1 - Chapter 1 of SR1.5 ZOD (simple example)

Example 1 is for IPCC SR1.5 ZOD Chapter 1 extracted from the original large ZOD.pdf (single file except for figures, which came extra), i.e. using Acrobat to delete all other pages from the large ZOD.pdf than those of Chapter 1. This step is of course not really needed, but was done for the following example for easier repetition (nearby provided). Alternatively select in step 1 only chapter 1.

Step 1): Method b9) extracts the wanted text in one go. Paste the clipboard obtained in Acrobat into a new TextEdit window converted to text before the paste and save all with UTF-8 encoding into file:

"ZOD Ch1-b9.txt" [9]

Step 2): Add file 'fixDraftPars.txt' into the same directory where above file resides. Ex.:



Most parameters should be self-explanatory. Note however, the parameter TOTALPAGES, being the total number of pages of the current chapter, does not matter when using method b9). It only matters for option -i for shell script fixDraft.sh for a method such as b7) where no headers and footers are generated and need to be reconstructed. On the other hand, the value is merely cosmetic and should not affect the workings of TAT in any way. Note also the

[9] The name is to be read without the quotes

parameter DMMARKER, which stands for Document Manager Marker. It does not matter for the ZOD of SR1.5, but is needed for other reports such as the ZOD of the SROCC. The meaning of parameter REMOVETABFIGFROMNR will be explained below under Example 4.

Step 3):

Process ‘ZOD Ch1-b9.txt’ with following command (**Step 3**)

```
cat "ZOD Ch1-b9.txt" | fixDraft.sh -z > ZOD Ch1-b9-OUT.tx ; open "ZOD Ch1-b9-OUT.txt"
```

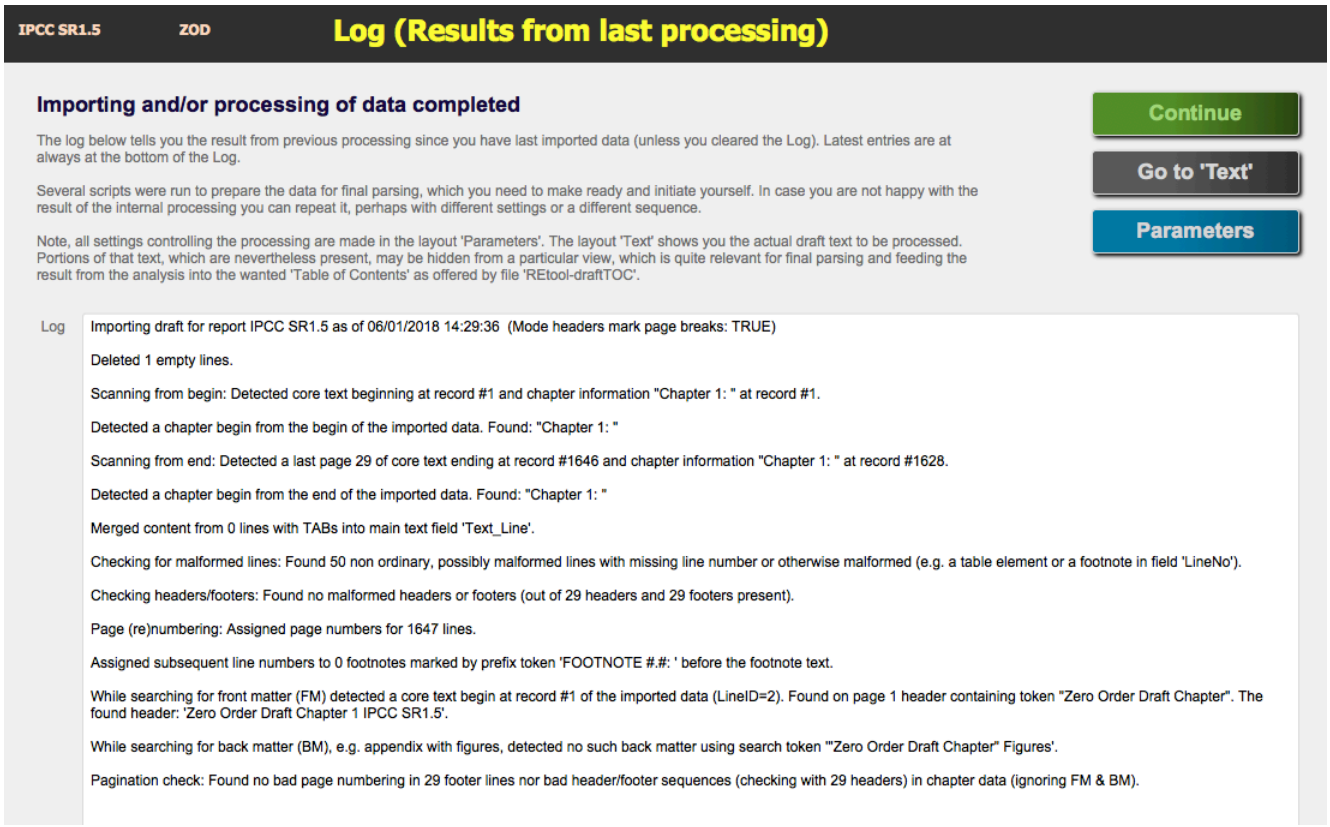
Step 4):

The inspection of the result file from previous step

```
open "ZOD Ch1-b9-OUT.txt"
```

should show that all looks fine (no editing of input file and repeating of step 3 necessary).

Step 5): Import the resulting file ‘ZOD Ch1-b9-OUT.txt’ into ‘TAT-drafttext.fmp12’ by executing FileMaker script (FM script) ‘Import draft text...’ (Cmd^1). When asked allow for replacing all data. You are first shown the result from the importing and subsequent processing, which should look similar to this:



Click button «Continue» (or «Go to ‘Text’») and in layout ‘Text’ you can see the entire draft text as imported:

IPCC SR1.5	ZOD	Text
1		Zero Order Draft Chapter 1 IPCC SR1.5
1		Begin of page 1-1 ---- Do Not Cite, Quote or Distribute 1-1 Total pages: 29
1	1	
1	2	Chapter 1: Framing and Context
1	3	
1	4	
1	5	Coordinating Lead Authors: Myles Allen (UK), Opha Pauline Dube (Botswana), William Solecki (USA)
1	6	
1	7	Lead Authors: Fernando Aragón-Durand (Mexico), Wolfgang Cramer (France), Mikiko Kainuma (Japan),
1	8	Jatin Kala (Australia), Natalie Mahowald (USA), Yacob Mulugetta (UK), Rosa Perez (Philippines), Graciela
1	9	Raga (Mexico), Morgan Wairiu (Solomon Islands), Kirsten Zickfeld (Canada)
1	10	
1	11	Contributing Authors:
1	12	
1	13	Review Editors: Ismail A. Elgizouli (Sudan), Andreas Fischlin (Switzerland), Xuejie Gao (China)
1	14	
1	15	Date of Draft: 09/04/17
1	16	
1	17	
1	18	Executive Summary
1	19	
1	20	
1	21	1.1 Human, ecological, and physical dimensions of 1.5 °C: Building a knowledge base for this report
1	22	
1	23	In December 2015, the Paris Agreement was negotiated by representatives of 195 countries at the 21st
1	24	Conference of the Parties of the United Nations Framework Convention on Climate Change (UNFCCC) to

Step 6): The parameters (layout ‘Parameters’ in ‘TAT-drafttext.fmp12’) currently in use should look as this:

Draft_token	IPCC SR1.5	
theRevStage	ZOD	
Text_is_only_TOC	0	
Draft_with_FM	0	
Headers_mark_page_breaks	1	
TOC_embedded_in_text	0	
Figures_embedded_in_text	0	
Tables_embedded_in_text	0	
Parse_from_RecNo	1	2
First_page_of_draft_core	1	
First_chapter_in_data_base	1	
Last_chapter_in_data_base	1	
MaxLineNo	32767	
MinConsecutiveLineNos	3	
Parse_till_RecNo	1647	1648

The importing should have updated some information such as ‘First_page_of_draft_core’, ‘First_chapter_in_data_base’, and ‘Last_chapter_in_data_base’ automatically for you. Should the information look differently, e.g. ‘First_chapter_in_data_base’ is not 1, some tokens may be not correctly defined. The following parameters matter for this and subsequent processing:

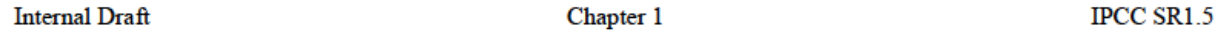
Header_token_in_FM	Zero Order Draft
Footer_token_not_FM	Total pages:
Header_token	Zero Order Draft Chapter
Word_pos_of_chapter_in_header	5
Footer_token	Do Not Cite, Quote or Distribute
Footer_token_is_at_footer_begin	0
Word_pos_of_pageInfo_in_footer	4

The following describes the purpose of these values and what to do about them if something does not work as expected. Skip this text (indented) if all seems fine and you wish to proceed quickly. Continue reading at the end of this step.

If ‘First_page_of_draft_core’ is not 1, then check the relevant tokens in parameters ‘Header_token_in_FM’, ‘Footer_token_not_FM’, and ‘Header_token’. They are used to determine the possible presence of the so-called front matter (FM), containing perhaps a table of content or some preamble text. Note, above values for these parameters

mean that page headers from chapters, i.e. where the core of the text starts, are expected all to contain also the word ‘Chapter’ (in the given sequence), while headers in the front matter are expected to contain only the phrase ‘Zero Order Draft’. Thanks to these parameter values the FM script ‘Clean import and renumber pages’ – which is called also by FM script ‘Import draft text...’ – should be able to determine front matter and determine that the first chapter starts at page 1 (‘First_page_of_draft_core’ shows value 1). Only if the values of these token parameters are correct will the execution of FM script ‘Identify front (FM) and back (BM) matter’ give you the correct value of parameter ‘First_page_of_draft_core’. If all is correct you should get the value 1 for ‘First_page_of_draft_core’ and by the way also the values for ‘Parse_from_RecNo’ exactly as shown.

If ‘First_page_of_draft_core’ is correct, but the chapter number is not correctly given in ‘First_chapter_in_data_base’, and ‘Last_chapter_in_data_base’, then some other parameters may be wrongly set and do not match the actual text. Let us look at a header from the original PDF:



Note, it starts with the phrase ‘Internal Draft’, yet field ‘Header_token’ uses the phrase ‘Zero Order Draft Chapter’. How come? First, it is the use of fixDraft.sh in **Step 3**) which has changed ‘Internal Draft’ to ‘Zero Order Draft’ (precisely it was script ‘massageDraft.sh’). The reason is a practical one related to parameter ‘Word_pos_of_chapter_in_header’ and explained just below. But note, also the entire terminology of REtool sticks consistently to following terms: ‘Zero Order Draft’ (ZOD), ‘First Order Draft’ (FOD), ‘Second Order Draft’ (SOD), and ‘Final Government Draft’ (FGD). Thus ‘Header_token_in_FM’ and ‘Header_token’ are correctly set to ‘Zero Order Draft’ and ‘Zero Order Draft Chapter’, respectively.

If those token values are all set as they should, there is still another possibility why parameters ‘First_chapter_in_data_base’ and ‘Last_chapter_in_data_base’ may contain wrong or even bad values. You need to make sure all values of the other parameters shown above to the right are exactly those as shown.

First note, the chapter number is typically retrieved from what is found in a header line (containing the token ‘Header_token’) at word position ‘Word_pos_of_chapter_in_header’. The value of 5 for the latter parameter means in a header line as this one

```
Zero Order Draft Chapter 1 IPCC SR1.5
```

the chapter number is to be found in the 5th word. Having either ‘Zero Order Draft Chapter’ or ‘First Order Draft Chapter’ or ‘Second Order Draft Chapter’ or ‘Final Government Draft Chapter’ does not affect that value, it is always 5. Convenient, isn’t it? If that parameter value is set to 0, this means there is no chapter information to be found in headers. You could try to use this value, since not all is lost as the following considerations show.

‘Word_pos_of_pageinfo_in_footer’ may also matter here and should be set correctly for all to work flawlessly. Its value 4 means that in a footer as this one

Begin of page 1-13 ---- Do Not Cite, Quote or Distribute 1-13 Total pages: 29

the page information, here 1-13, is to be found at the 4th word position. This value could also be 12, since the same information is redundantly contained twice in above footer. If you compare this to the original pdf, the footer line looks like this

Do Not Cite, Quote or Distribute 1-13 Total pages: 29

The line does not start with phrase 'Do Not Cite, Quote or Distribute', but with 'Begin of page 1-13 - - - -'. Why? The previous **Step 3**) has inserted that additional text. Since method b9) from **Step 1**) has been used, the footer line is actually placed right next after the header of every page, meaning that the footer is now actually at the begin of the page 13, not at its bottom. Other methods than method b9) used in **Step 1**) work differently and fixDraft.sh changes this accordingly, so that you are always well informed to which page and chapter the footer belongs and where in the text it sits.

Note, this page information contains not only the page number, here 13, but also the chapter information, here chapter 1, once more, redundantly to the header. That is the reason why you could have the parameter 'Word_pos_of_chapter_in_header' set to 0, and still be able to determine correctly the first and last chapter in your data. Only if you would set both these parameters to 0 would that no longer work. Note, then you would also set the parameter 'Headers_mark_page_breaks' to 0 (false), since then only line numbers could be used to determine page breaks.

Finally, this all means also that the parameter 'Footer_token_is_at_footer_begin' must be set to 0 (false). The phrase 'Do Not Cite, Quote or Distribute' as given in field 'Footer_token' is not at the begin of the footer line. Only if you would have used in **Step 3**) for script 'fixDraft.sh' the option -d, then that value would have to be adjusted accordingly and set to 1 (true). Only then the footer lines would look exactly as in the original PDF. Yet, know, doing that is not advisable. It would interfere with the import process (FM script 'Import draft text…'). The line numbers would be concatenated with all of the line texts, all then ending up in the wrong field, i.e. in the 2nd column (cf. layout 'Text'). Normally that field should hold only line numbers. Thus, in normal use of 'fixDraft.sh', the option -d should never be used.

Assuming the chapter number and first page was identified fine, let us continue. Note, you can conveniently toggle back and forth between the draft text (layout 'Text') and the parameters (layout 'Parameters'). Use FM script "Toggle layouts 'Text' and 'Parameters'" (Cmd^0) or the buttons provided. Before going to the next step, however, make sure all is prepared exactly as it should, notably the field 'First_page_of_draft_core' is now important. You have to set its value to 17. Why? This is because we intend to use with REtool the original ZOD.pdf as distributed by TSU. Perhaps confirm this value of 17 by looking at the original large ZOD.pdf (provided nearby with this release). The 17th page, labelled 1-1, is the one where chapter 1 really begins. At the end of this step, your parameters should then look like this

Draft_token	IPCC SR1.5	
theRevStage	ZOD	
Text_is_only_TOC	0	
Draft_with_FM	1	
Headers_mark_page_breaks	1	
TOC_embedded_in_text	0	
Figures_embedded_in_text	0	
Tables_embedded_in_text	0	
Parse_from_RecNo	1	2
First_page_of_draft_core	17	
First_chapter_in_data_base	1	
Last_chapter_in_data_base	1	
MaxLineNo	32767	
MinConsecutiveLineNos	3	
Parse_till_RecNo	1647	1648

Step 7): Select first the records to be parsed, which means basically excluding from the current set all front and back matter. To accomplish this simply execute FM script ‘Prepare for TOC Parsing’. Not much will happen, except that some lines are no longer shown, e.g. page headers and footers should not show. In layout ‘Text’ you should see something similar to this:

Layout: Text | View As: | Preview

IPCC SR1.5	ZOD	Text
1	2	Chapter 1: Framing and Context
1	17	
1	18	Executive Summary
1	19	
1	20	
1	21	1.1 Human, ecological, and physical dimensions of 1.5 °C: Building a knowledge base for this report
1	22	
1	23	In December 2015, the Paris Agreement was negotiated by representatives of 195 countries at the 21st
1	24	Conference of the Parties of the United Nations Framework Convention on Climate Change (UNFCCC) to
1	25	undertake ambitious efforts towards mitigation of greenhouse-gas emissions, adaptation and finance, to start

Then execute FM script ‘Parser - TOC’ to parse all the data in the current found set. When you are asked to append or replace the data, best answer by replace, since this is the first example. This extracts the wanted meta data and exports them to the data base file ‘TAT-draftTOC.fmp12’ where they can be looked at as a detailed Table of Content (TOC) (layout ‘Table_of_Contents’) not only containing page numbers, but also line numbers for every item listed.

The Log entries in ‘TAT-drafttext.fmp12’ (layout ‘Results from last processing’) from **Steps 5) to 7)** have created a log, which should look similar to this:

Log

Importing draft for report IPCC SR1.5 as of 16/12/2017 14:35:52 (Mode headers mark page breaks: TRUE)

Deleted 2 empty lines.

Detected a chapter begin from the begin of the imported data. Found: "Chapter 1: "

Detected a chapter begin from the end of the imported data. Found: "Chapter 1: "

Merged content from 0 lines with TABs into main text field 'Text_Line'.

Checking for malformed lines: Found 71 non ordinary, possibly malformed lines with missing line number or otherwise malformed (e.g. a table element or a footnote in field 'LineNo').

Checking headers/footers: Found no malformed headers or footers (out of 29 headers and 29 footers present).

Page (re)numbering: Assigned page numbers for 1668 lines.

While searching for front matter (FM) detected a chapter begin at record #1 of the imported data (LineID=3). Found on page 1 header containing token "Zero Order Draft Chapter". The found header: 'Zero Order Draft Chapter 1 IPCC SR1.5'.

While searching for back matter (BM), e.g. appendix with figures, detected no such back matter using search token "Zero Order Draft Chapter" Figures'.

Pagination check: Found no bad page numbering in 29 footer lines nor bad header/footer sequences (checking with 29 headers) in chapter data (ignoring FM & BM).

TOC Parsing preparation: Found the minimally expected 1 record with 'Executive Summary' (ID=21, expected 1, since 'FM_with_chapter_TOC' is FALSE).

TOC Parsing preparation: Readied 1582 records for TOC parsing.

Parsing the TOC: Parsed 1582 records and detected and exported 1 chapter, 1 FM, 1 ES, 51 headings, 8 figures, 0 tables, 4 boxes, 0 FAQs, 0 footnotes, and 1 refs section.

If some elements, say figures would be missing or no list of reference section could be detected, then this may depend on the values for other parameters (not explained above). For this example to work fully, the following parameter values are to be used:

Global parameters characterizing the draft text **Parameters** Text Analysis Version 1.0fc5

Draft_token	IPCC SR1.5	Header_token	Zero Order Draft Chapter
theRevStage	ZOD	Word_pos_of_chapter_in_header	5
Text_is_only_TOC	0	Footer_token	Do Not Cite, Quote or Distribute
Draft_with_FM	0	Footer_token_is_at_footer_begin	0
Headers_mark_page_breaks	1	Word_pos_of_pageInfo_in_footer	4
TOC_embedded_in_text	0	SPM_title_token_in_text	Summary for Policymakers
Figures_embedded_in_text	0	SPM_section_token_in_text	SPM.#
Tables_embedded_in_text	0	TS_title_token_in_text	Technical Summary
Parse_from_RecNo	1 2	Chapter_token_in_text	Chapter #:
First_page_of_draft_core	1	FM_token_in_text	Front Matter
First_chapter_in_data_base	1	Figure_token_in_text	Figure #.#:
Last_chapter_in_data_base	1	Table_token_in_text	Table #.#:
MaxLineNo	32767	Box_token_in_text	Box #.#:
MinConsecutiveLineNos	3	BoxFigure_token_in_text	Box #.#, Figure #:
Parse_till_RecNo	1647 1648	BoxTable_token_in_text	Box #.#, Table #:
NOTE: Some tokens may have leading or trailing blanks, which may matter critically for parsing		FAQ_token_in_text	FAQ #.#:
Header_token_in_FM	Zero Order Draft	Footnote_token_in_text	FOOTNOTE #.#:
Footer_token_not_FM	Total pages:	Refs_token_in_text	References
BM_token	Figures		

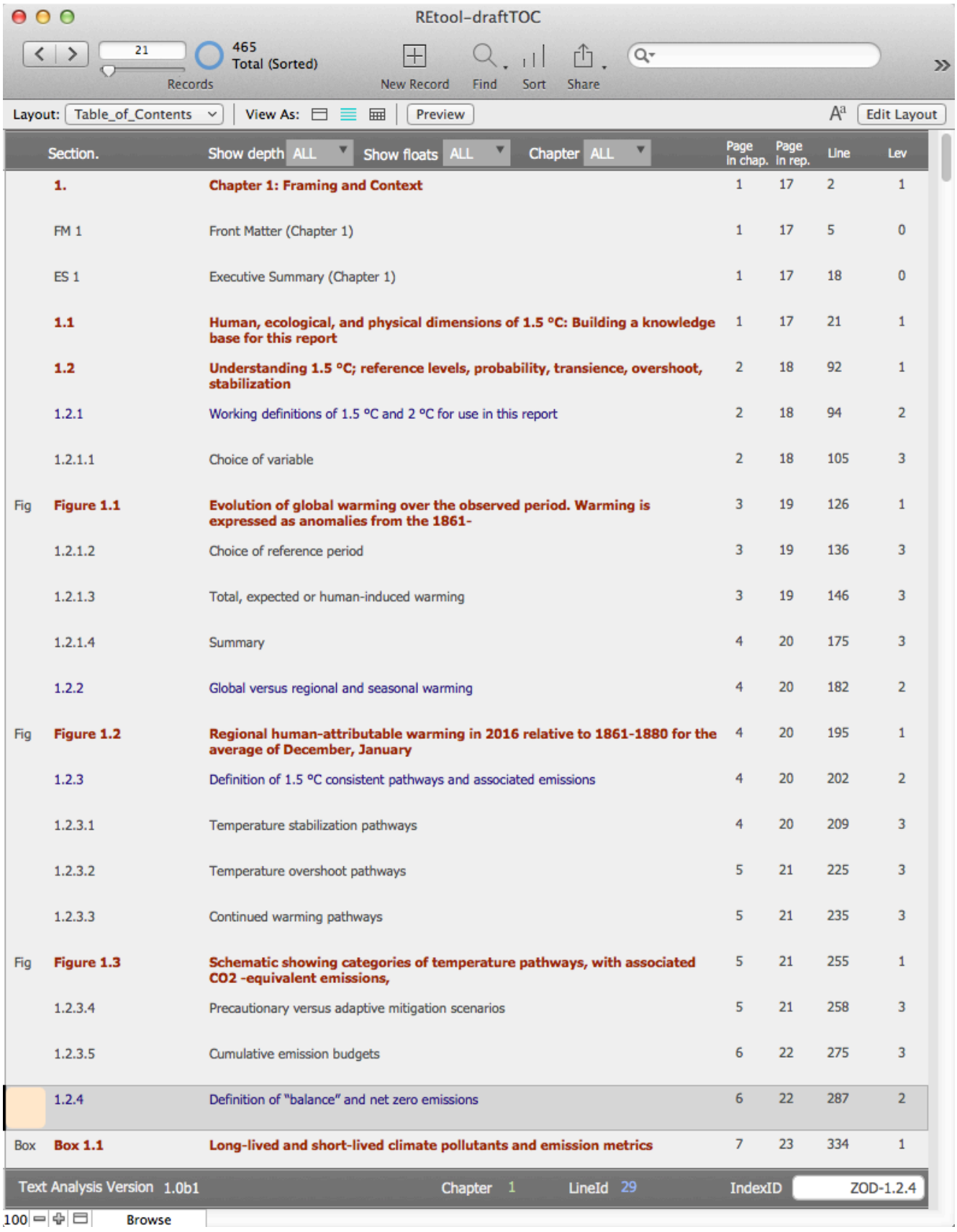
Note, the hash mark stands for a number. So a figure such as figure 3 in chapter 1 is expected to have a caption starting with the phrase ‘Figure 1.3: ’ (note also the blank at the end after the colon). If authors have mistyped this, the parsing would not have succeed for that figure and it would be missing from the generated table of contents. Note also, the colon is quite relevant. Why? The entire chapter text should contain at least one reference to the figure and such a reference may even end up at the begin of a line. Parsing would then fail, since it could no longer correctly distinguish between a mere reference and the actual figure caption. Yet the location of the figure caption is the only piece of information the parser wants to extract, i.e. the page and line number where the figure caption begins.

If the list of reference section is not found, perhaps there is a typo or another convention followed. E.g. if authors would have written ‘List of References’, that section heading would be missed, since parameter value ‘Refs_token_in_text’ set to ‘References’ means a different thing. It means that the heading of that section must start only with ‘References’, nothing else. Otherwise it will not be detected by the TOC parser. However, in this example the value shown above is what the authors wrote. If you wish, you can experiment with that. E.g. change the parameter value to ‘List of References’ and repeat **Step 7**) (Cmd^6 and Cmd^7, Append). The log should say 0 refs sections:

```
Parsing the TOC: Parsed 1560 records and detected and exported 0 SPMs, 0 TSs, 1 chapter, 0 FMs, 1 ES, 51 headings, 8 figures, 0 tables, 4 boxes, 0 FAQs, 0 footnotes, and 0 refs sections.
```

You can change the text of line 1047 also to ‘List of References’ and repeat **Step 7**) (Cmd^6 and Cmd^7, Append). Then the log should say again that 1 refs section was found. Note, ‘TAT-draftTOC.fmp12’ will obtain another entry for the reference section, which should read ‘List of References (Chapter1)’. Thus do it a last time to ensure all is back to what it was and repeat **Step 7**) (Cmd^6 and Cmd^7) to make sure the title used by ‘TAT-draftTOC.fmp12’ is the correct one ‘References (Chapter1)’.

Step 8) The final result from all this in ‘TAT-draftTOC.fmp12’ should look similar to this:



To test the correctness of all you can click on the beige button on the left of the above selected heading 1.2.4. This should get you to the 22nd page of the ZOD.pdf, i.e. page 1-6, where line 287 can be found, if you have that pdf open in Skim. Perhaps check this with

some other items listed, whichever you want. Note, the parsing for the Table Of Content (TOC) meta data as done during step 7, i.e. executing FM script ‘Parser -TOC’, creates additional entries, which you will not find in an ordinary TOC. Look for an entry ‘End Chapter 1’ (Index ‘End Ch 1’). With it you can jump to the end of the chapter 1. Similarly this step would also create entries for footnotes, e.g. ‘fn 4.1’, if present in the original chapter etc. (not the case for the chapter used in this example, but see examples below).

The inspection results in no obvious flaw in the meta data and all seems fine.

Step 9): Finally, export from ‘TAT-draftTOC.fmp12’ the wanted spreadsheet using the dedicated FM script ‘Export to spreadsheet’ (Cmd^1). The nearby folder ‘Example output files’ contains the exported spreadsheet.

Example 2 - Ch3 SR1.5 ZOD (fixing typos)

Example 2 is for IPCC SR1.5 ZOD Chapter 3 extracted from the original large ZOD.pdf (single file except for figures, which came extra), i.e. using to delete all other pages than those of Chapter 3. This is not really needed, but was done for this example (file nearby). This example is slightly more complex than the previous example, since some typos by the authors prevent a processing as straightforward as in the previous example.

Step 1): Method b9) does not extract the wanted text in one go as in Example 1 unless you make sure the receiving text file in TextEdit is already a text file before you paste the clipboard as you copied it from Acrobat. Otherwise the paste process from the clipboard aborts prematurely, here after having encountered Table 3.1 on page 3-11. Note, TextEdit as used in this example fails to display an alert or other error message that the clipboard was not fully copied. You may need to check manually that the entire chapter was copied by inspecting its end. If you follow exactly the instructions as given above, all should be fine and you can save all of chapter 3 with UTF-8 encoding to following text file:

```
"ZOD Ch3-b9.txt"
```

Step 2): You can use the same file as for Example 1. Note, as explained above under Example 1, the parameter TOTALPAGES does not matter. Thus, there is no need to adjust this value to the correct value of 88 pages.

Step 3): Process ‘ZOD Ch3-b9.txt’ with following command (first attempt at **Step 3**)

```
cat "ZOD Ch3-b9.txt" | fixDraft.sh -z > a.txt ; open a.txt
```

Step 4): The inspection shows file ‘ZOD Ch3-b9.txt” needs editing. See after lines 2343. It looks similar to this:

```
2340
2341    [INSERT Box 3, Table 1 HERE]
2342    Box 3, Table 1: Projected Annual Property Damage from Coastal Flooding for Se
2343    Century. Source: Yohe, et al., 2010.
         Panel A: Annual Damages Along a Sea Level Rise Scenario that Reaches 1.0m by
        Year
        10th Percentile
        Median
        90th Percentile
        Current
        $0
        $0
        $110
2030
        $0
        $50
        $220
2050
        $50
        $110
        $250
2070
        $160
        $230
        $350
2090
        $220
        $290
        $400
        Panel B: Annual Damages Along a Sea Level Rise Scenario that Reaches 0.6m by
        Year
        10th Percentile
```

These lines are not numbered and make little sense. They are so-called extra table lines, which should have been discarded. Normally fixDraft.sh does manage to remove such lines (unless you would have used option -t for not removing such extra lines for tables, and similarly option -g for graphs). Why did it fail? The reason is that the authors of chapter 3 have not properly labelled on line 2342 the ‘Box 3, Table 1’. The correct format would have been ‘Box 3.3, Table 1’ (note the chapter number is missing in ‘Box 3’, it is ‘Box 3.3’, the third box in chapter 3). As a consequence fixDraft.sh has failed to recognize the table and left there everything as is (except for line numbers where they could be found). To also remove above extra lines only disturbing later processing you need to correct for that typo of the authors by following edit:

‘Box 3, Table 1: ’ -> ‘Box 3.3, Table 1: ’

Note, the edit has to be done where the colon is, not in the line ‘[INSERT Box 3, Table 1 HERE]’. You can do the edit there as well, but it does not matter for TAT. This is because TAT detects a table, figure, box, FAQ caption only if it has a colon right after the number of the item. Actually, screening the chapter’s PDF systematically for such typos, there is a similar typo present in the chapter on line 2108, which also needs fixing:

‘Box 1, Figure 1: ’ -> ‘Box 3.1, Figure 1: ’

Store both edits in file

"ZOD Ch3-b9-edited.txt"

Then reprocess ‘ZOD Ch3-b9-edited.txt’ with following command (Repeat **Step 3**)

```
cat "ZOD Ch3-b9-edited.txt" | fixDraft.sh -z > "ZOD Ch3-b9-edited-OUT.txt"
```

The result, i.e. file

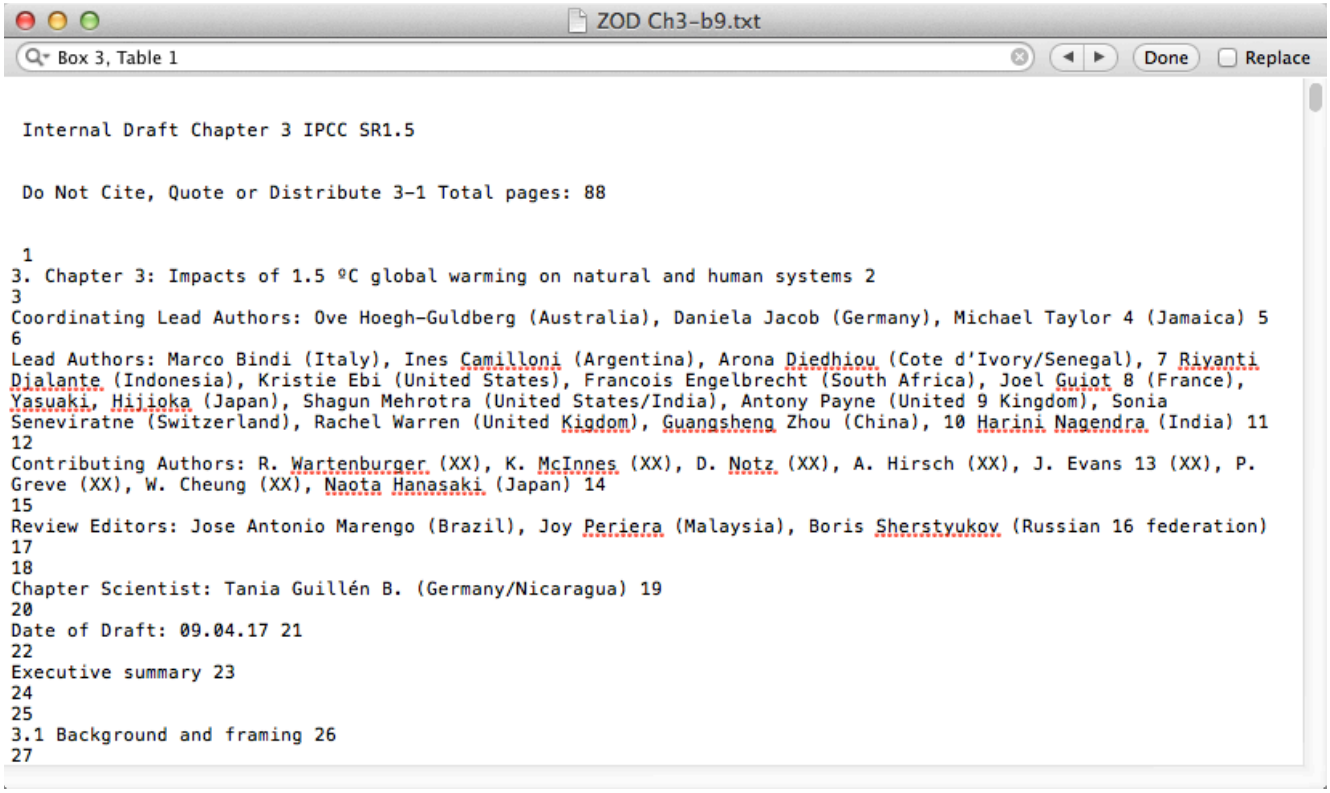
"ZOD Ch3-b9-edited-OUT.txt"

looks now very good and you can see that the extra table lines have been removed by fixDraft.sh:

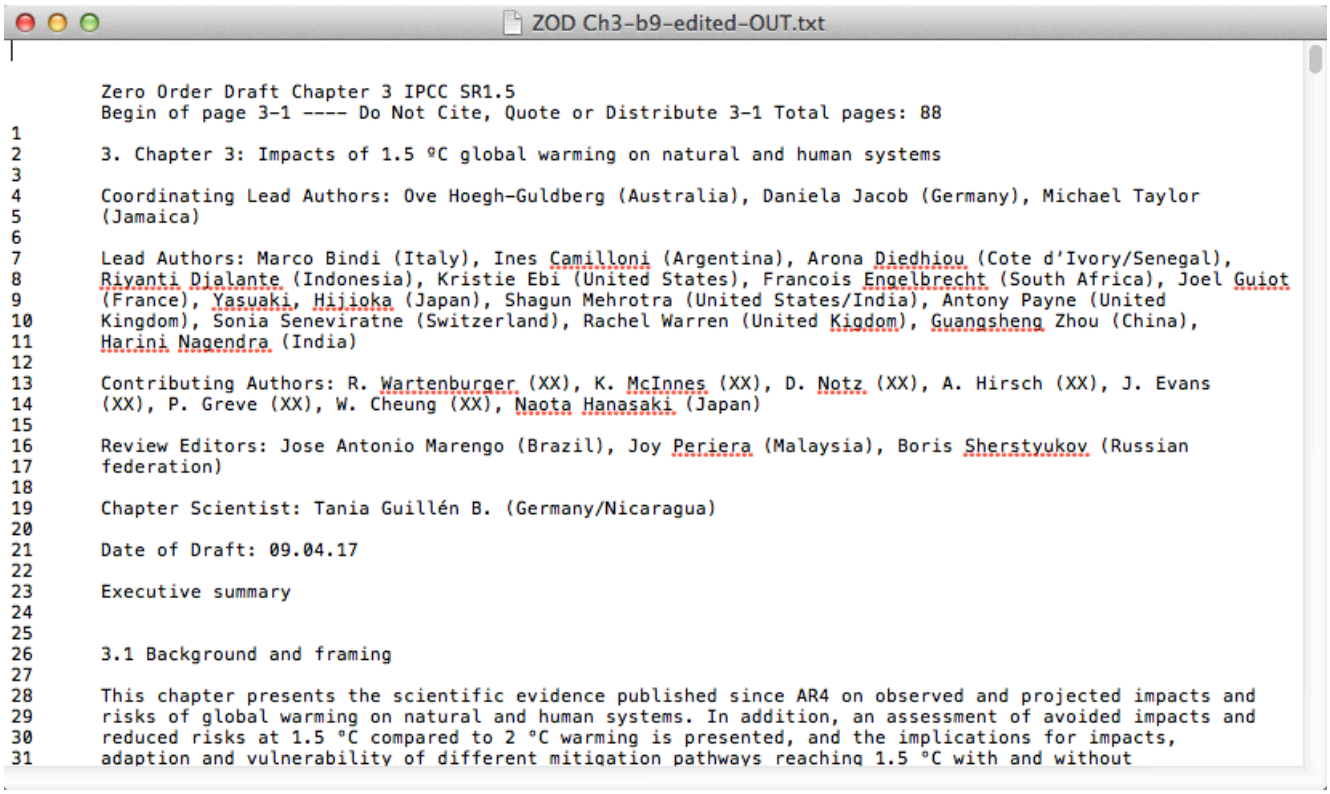
```
2339    0.6m alternative.
2340
2341    [INSERT Box 3, Table 1 HERE]
2342    Box 3.3, Table 1: Projected Annual Property Damage from Coastal Flooding for Sel
2343    Century. Source: Yohe, et al., 2010.

        REMOVED as of here ALL table lines
        STOPPED removing table lines (removed a total of 51 lines)

2344
2345
2346
2347    One $50 million adaptation project (a protecting barrier) was considered. Along
```

Compare how readable the file ‘ZOD Ch3-b9-edited-OUT.txt’ is in contrast to the original file ‘ZOD Ch3-b9-edited.txt’, which resembles only garbled up data at first glance. Here a comparison:

Before (input to `fixDraft.sh -z`):



After (output from `fixDraft.sh -z`):



Step 5): Import the resulting file 'ZOD Ch3-b9-edited-OUT.txt' into 'TAT-drafttext.fmp12' using the same parameters as you used in example 1. Use the FM script 'Import draft text…" (Cmd^1) and select the file. When asked allow for replacing all data. Note, this does

not delete any TOC data you might already have in the file 'TAT-draftTOC.fmp12'. This deletes only the draft text as contained in 'TAT-drafttext.fmp12' layout 'Text'. Since you normally do not edit the draft text and can easily reimport data from any files as have resulted from **Steps 1**) till **3**), there is no risk of losing valuable data. The text of chapter 1 of the IPCC SR1.5 ZOD should now be in the FM data base 'TAT-drafttext.fmp12', visible in default layout 'Text'. The parameter (layout 'Parameters' in 'TAT-drafttext.fmp12') should look as this:

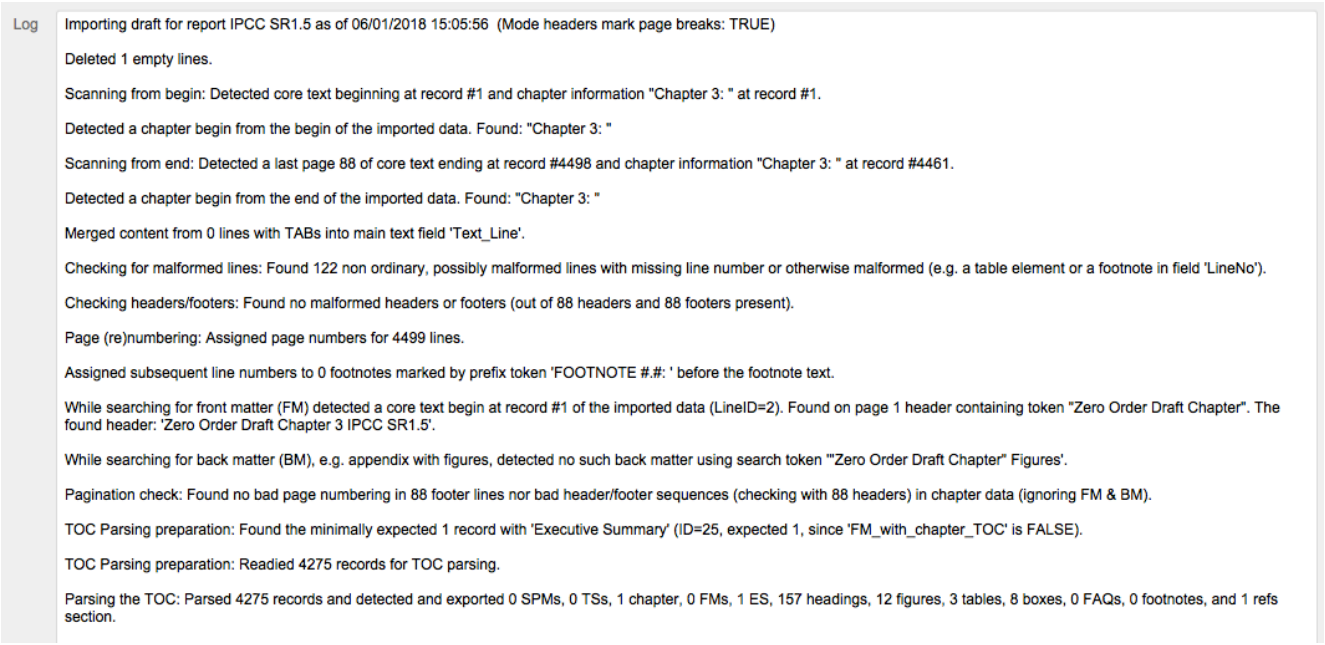
Draft_token	IPCC SR1.5	
theRevStage	ZOD	
Text_is_only_TOC	0	
Draft_with_FM	1	
Headers_mark_page_breaks	1	
TOC_embedded_in_text	0	
Figures_embedded_in_text	0	
Tables_embedded_in_text	0	
Parse_from_RecNo	1	2
First_page_of_draft_core	1	
First_chapter_in_data_base	3	
Last_chapter_in_data_base	3	
MaxLineNo	32767	
MinConsecutiveLineNos	3	
Parse_till_RecNo	4499	4500

Step 6) Set the field 'First_page_of_draft_core" to the correct page number in the original pdf, i.e. page 86, as chapter 3 begins on that page in the large, original ZOD.pdf.

Step 7): Then perform the actual parsing and export the TOC meta data from chapter 3 to 'TAT-draftTOC.fmp12'. Accomplish this by first executing as in the previous example FM script 'Prepare for TOC Parsing' (Cmd^6) and then 'Parser - TOC' (Cmd^7) by appending the data by clicking on button 'Append':

Append would be meaningful and necessary if you wish to keep the data added from previous Example 1, which has added the TOC meta data from chapter 1. As a result from appending you have the TOC meta data from both chapters 1 and 3. You see that TAT allows you to accumulate the TOC meta data and work iteratively.

The report (log) of all processing from **Step 5**) till this **Step 7**) should look similar to this:



Log

Importing draft for report IPCC SR1.5 as of 06/01/2018 15:05:56 (Mode headers mark page breaks: TRUE)

Deleted 1 empty lines.

Scanning from begin: Detected core text beginning at record #1 and chapter information "Chapter 3: " at record #1.

Detected a chapter begin from the begin of the imported data. Found: "Chapter 3: "

Scanning from end: Detected a last page 88 of core text ending at record #4498 and chapter information "Chapter 3: " at record #4461.

Detected a chapter begin from the end of the imported data. Found: "Chapter 3: "

Merged content from 0 lines with TABs into main text field 'Text_Line'.

Checking for malformed lines: Found 122 non ordinary, possibly malformed lines with missing line number or otherwise malformed (e.g. a table element or a footnote in field 'LineNo').

Checking headers/footers: Found no malformed headers or footers (out of 88 headers and 88 footers present).

Page (re)numbering: Assigned page numbers for 4499 lines.

Assigned subsequent line numbers to 0 footnotes marked by prefix token 'FOOTNOTE #.#: ' before the footnote text.

While searching for front matter (FM) detected a core text begin at record #1 of the imported data (LineID=2). Found on page 1 header containing token "Zero Order Draft Chapter". The found header: 'Zero Order Draft Chapter 3 IPCC SR1.5'.

While searching for back matter (BM), e.g. appendix with figures, detected no such back matter using search token '"Zero Order Draft Chapter" Figures'.

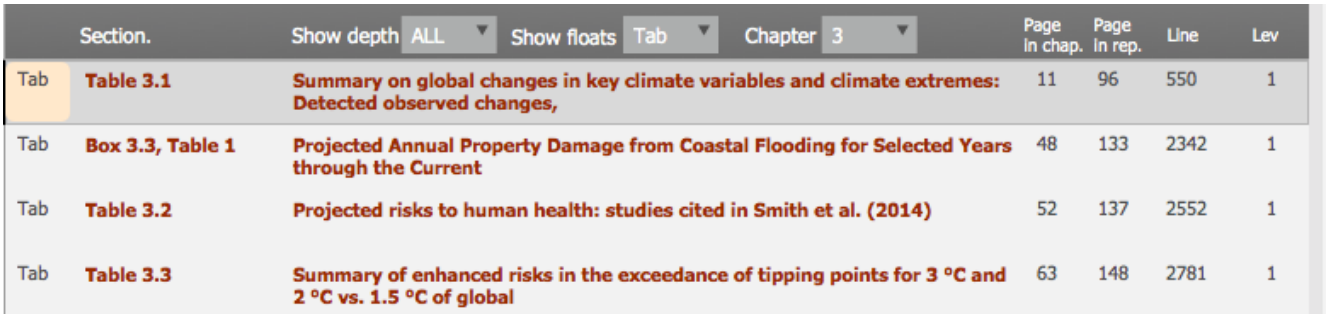
Pagination check: Found no bad page numbering in 88 footer lines nor bad header/footer sequences (checking with 88 headers) in chapter data (ignoring FM & BM).

TOC Parsing preparation: Found the minimally expected 1 record with 'Executive Summary' (ID=25, expected 1, since 'FM_with_chapter_TOC' is FALSE).

TOC Parsing preparation: Readied 4275 records for TOC parsing.

Parsing the TOC: Parsed 4275 records and detected and exported 0 SPMs, 0 TSs, 1 chapter, 0 FMs, 1 ES, 157 headings, 12 figures, 3 tables, 8 boxes, 0 FAQs, 0 footnotes, and 1 refs section.

Step 8): The results should be fine, including Box 3.3, Table 1. Check it out.



	Section.	Show depth ALL / Show floats Tab / Chapter 3	Page in chap.	Page in rep.	Line	Lev
Tab	**Table 3.1**	**Summary on global changes in key climate variables and climate extremes: Detected observed changes,**	11	96	550	1
Tab	**Box 3.3, Table 1**	**Projected Annual Property Damage from Coastal Flooding for Selected Years through the Current**	48	133	2342	1
Tab	**Table 3.2**	**Projected risks to human health: studies cited in Smith et al. (2014)**	52	137	2552	1
Tab	**Table 3.3**	**Summary of enhanced risks in the exceedance of tipping points for 3 °C and 2 °C vs. 1.5 °C of global**	63	148	2781	1

Step 9): Finally, you can export from 'TAT-draftTOC.fmp12' the wanted spreadsheet using the dedicated FM script 'Export to spreadsheet' (Cmd^1). That spreadsheet is then ready to be imported into the main REtool using in file 'REtool-textlinks.fmp12' the FM script 'Import TOC from spreadsheet'.

Example 3 - Processing Table of Content of Front Matter SR1.5 ZOD

Example 3 is for IPCC SR1.5 ZOD the report wide table of contents (TOC) contained in the so called front matter (FM). The procedure is basically the same as the one given for Example 1 and consists of following slight variations:

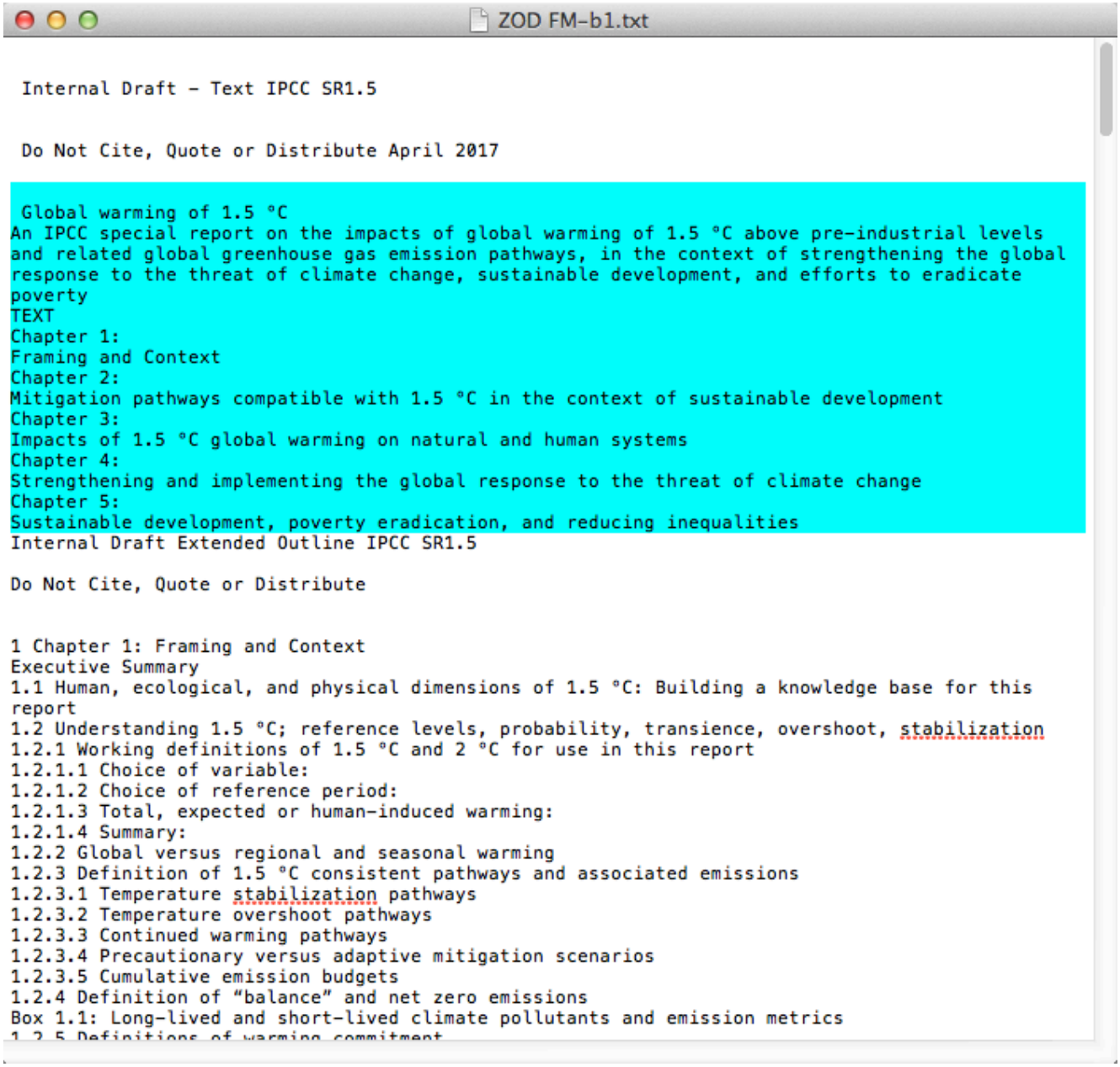
Step 1): Use e.g. method b1) to obtain the first text file, but if works also with method b10:

> "ZOD FM-b1.txt"

Step 2): As for Example 1.

Steps 3 and 4):

Delete the redundant content of the first page (selected text), which does only disturb the analysis of the TOC:



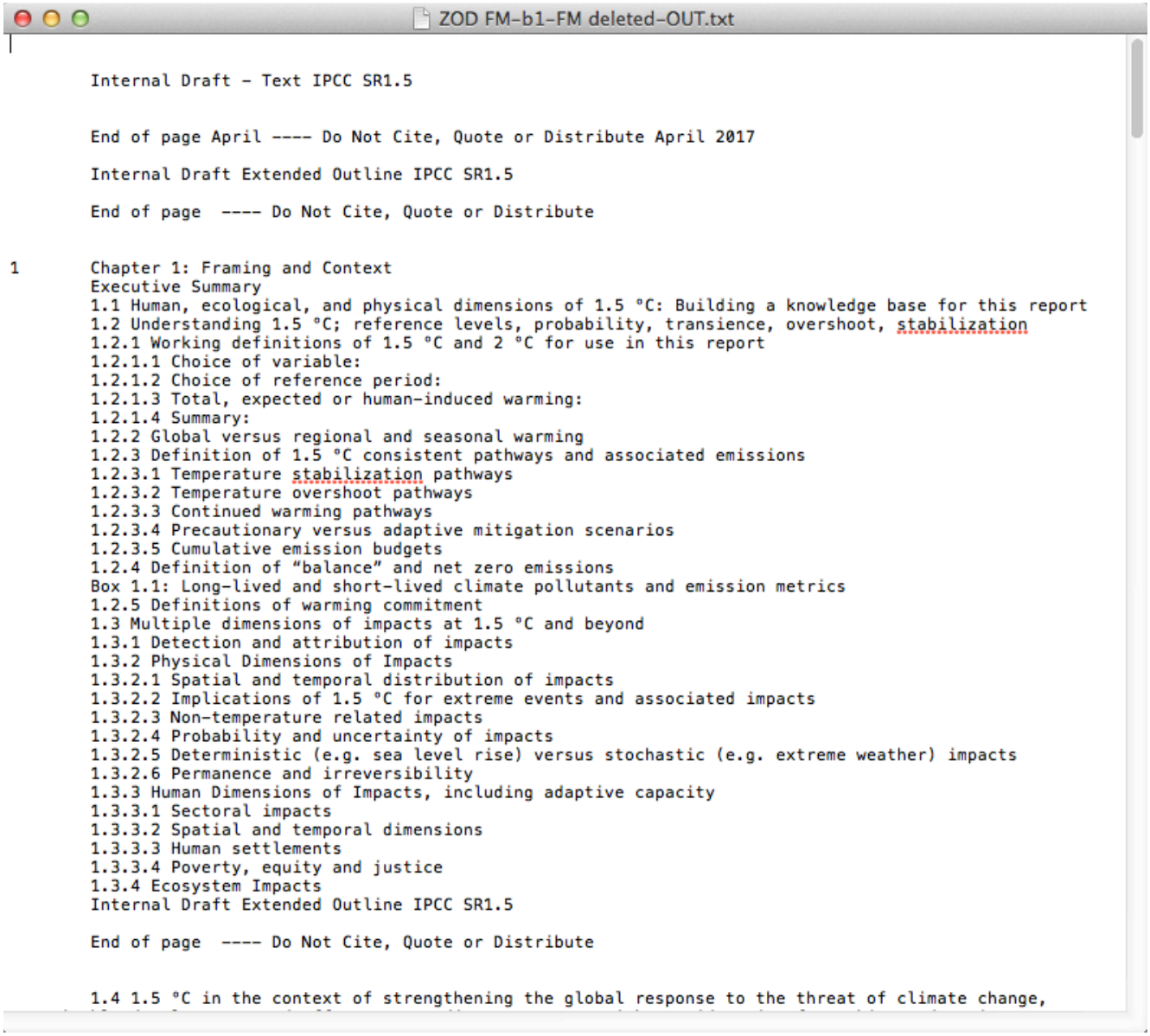
Then execute command:

```
cat "ZOD FM-b1-FM deleted.txt" | massageDraft.sh | insertTabDels.awk > "ZOD FM-b1-FM deleted-OUT.txt"
```

to obtain the wanted file

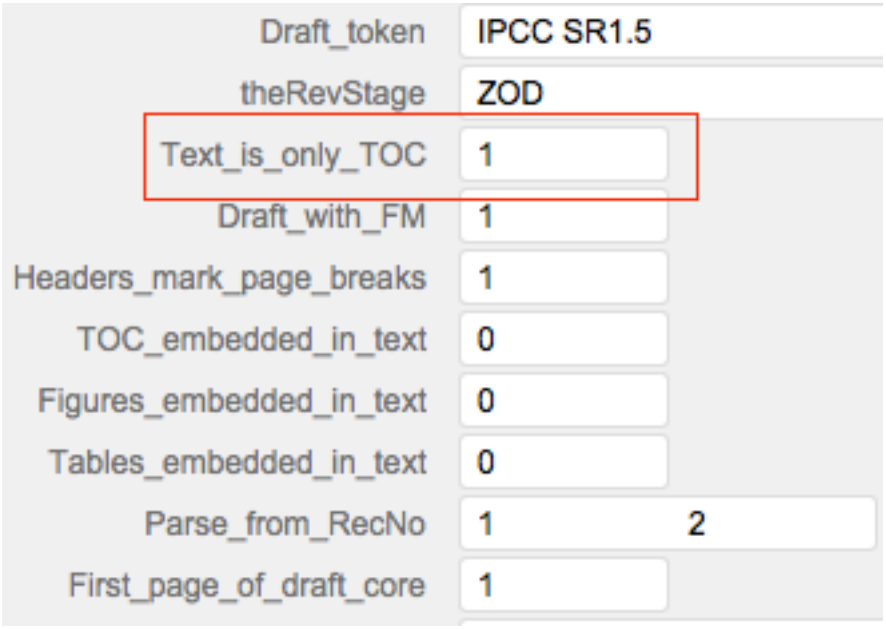
‘ZOD FM-b1-FM deleted-OUT.txt’

which should look similar to this:



Steps 5) to 9):

Process in 'TAT-drafttext.fmp12' the data from this file 'ZOD FM-b1-FM deleted-OUT.txt' with the same parameters as used before (import with FM script 'Import draft text...' (replace all data). But before any parsing make sure parameter 'Text_is_only_TOC' is set to 1 (true). In contrast to previous examples this is now the case, we have imported only a table of content from the ZOD's front matter. Also parameter 'First_page_of_draft_core' needs to be set to 1 and parameters are to look as this:



Then parse and export the TOC meta data (Cmd^6, Cmd^7). The result is a complete TOC in 'TAT-draftTOC.fmp12'. Yet, only with titles and page numbers and line numbers are missing. A subsequent step as described with Example 1 can add those meta data to the table 'Table_of_Contents' as it accumulates in 'TAT-draftTOC.fmp12' as pieces of the text are processed.

Here the Log from the import and the subsequent processing (Cm^1, Cmd^6, Cmd^7):

Log

Importing draft for report IPCC SR1.5 as of 06/01/2018 15:21:43 (Mode headers mark page breaks: TRUE)

Deleted 2 empty lines.

Scanning from begin: Detected a first page Do of core text beginning at record #4 and chapter information "April" at record #4.

Detected a chapter begin from the begin of the imported data. Found: "Chapter 1: "

Scanning from end: Detected a last page Not of core text ending at record #530 and chapter information "Do" at record #525.

Detected a chapter begin from the end of the imported data. Found: "Chapter 5: "

Merged content from 0 lines with TABs into main text field 'Text_Line'.

Checking for malformed lines: Found 510 non ordinary, possibly malformed lines with missing line number or otherwise malformed (e.g. a table element or a footnote in field 'LineNo').

Checking headers/footers: Found no malformed headers or footers (out of 0 headers and 16 footers present).

Page (re)numbering: Assigned page numbers for 531 lines.

Assigned subsequent line numbers to 0 footnotes marked by prefix token 'FOOTNOTE #.#: ' before the footnote text.

Sorry, to succeed FM script 'Identify front matter by token argument (internal)' requires the presence of page header records containing token 'Zero Order Draft Chapter'. Found no such record in the data base!

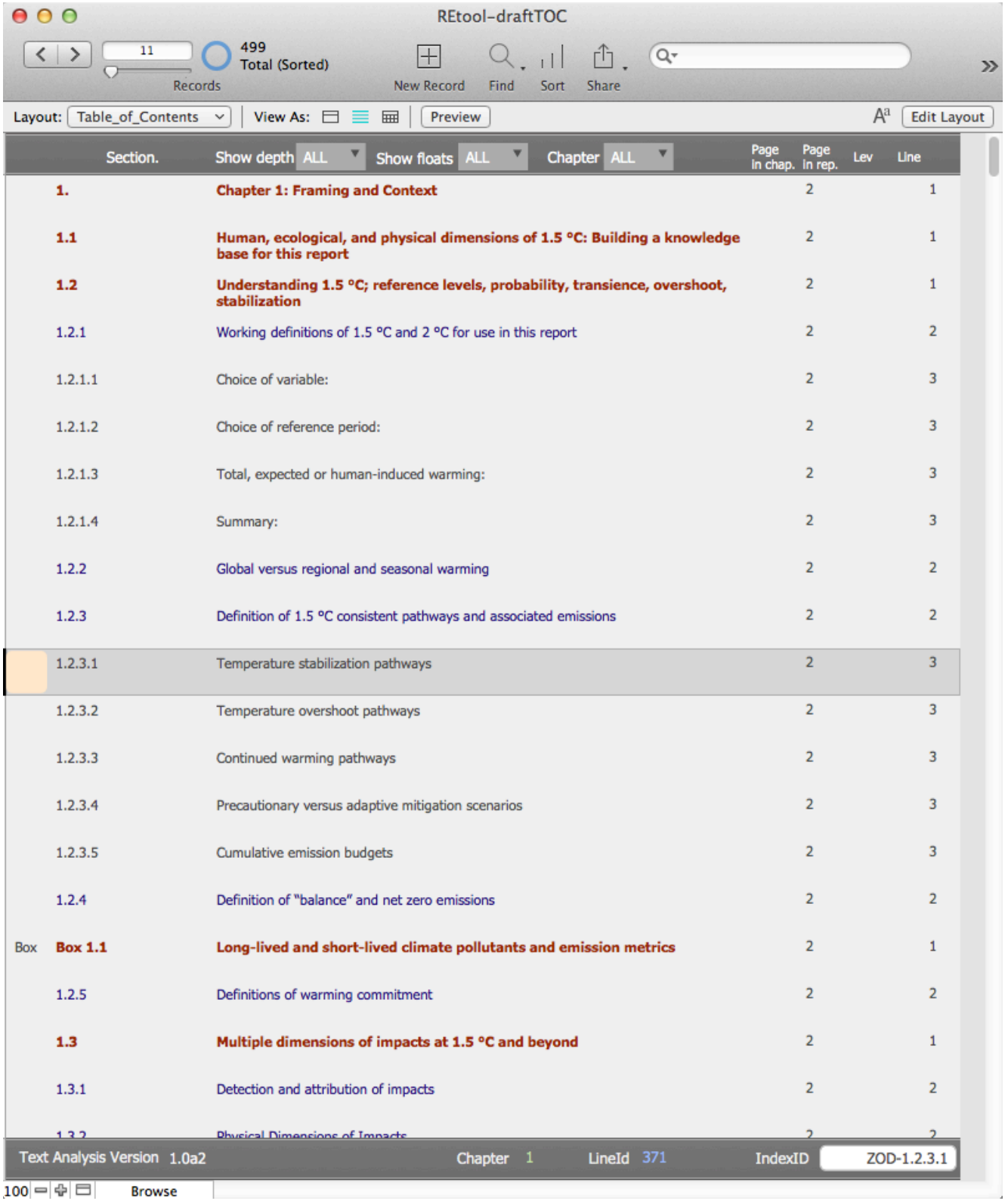
While searching for back matter (BM), e.g. appendix with figures, detected no such back matter using search token '"Zero Order Draft Chapter" Figures'.

Pagination check: Found bad page numbering for 16 footer lines (out of 16) and/or 16 bad header/footer sequences (checking with 0 headers) as of page 1.

TOC Parsing preparation: Readied 499 records for TOC parsing.

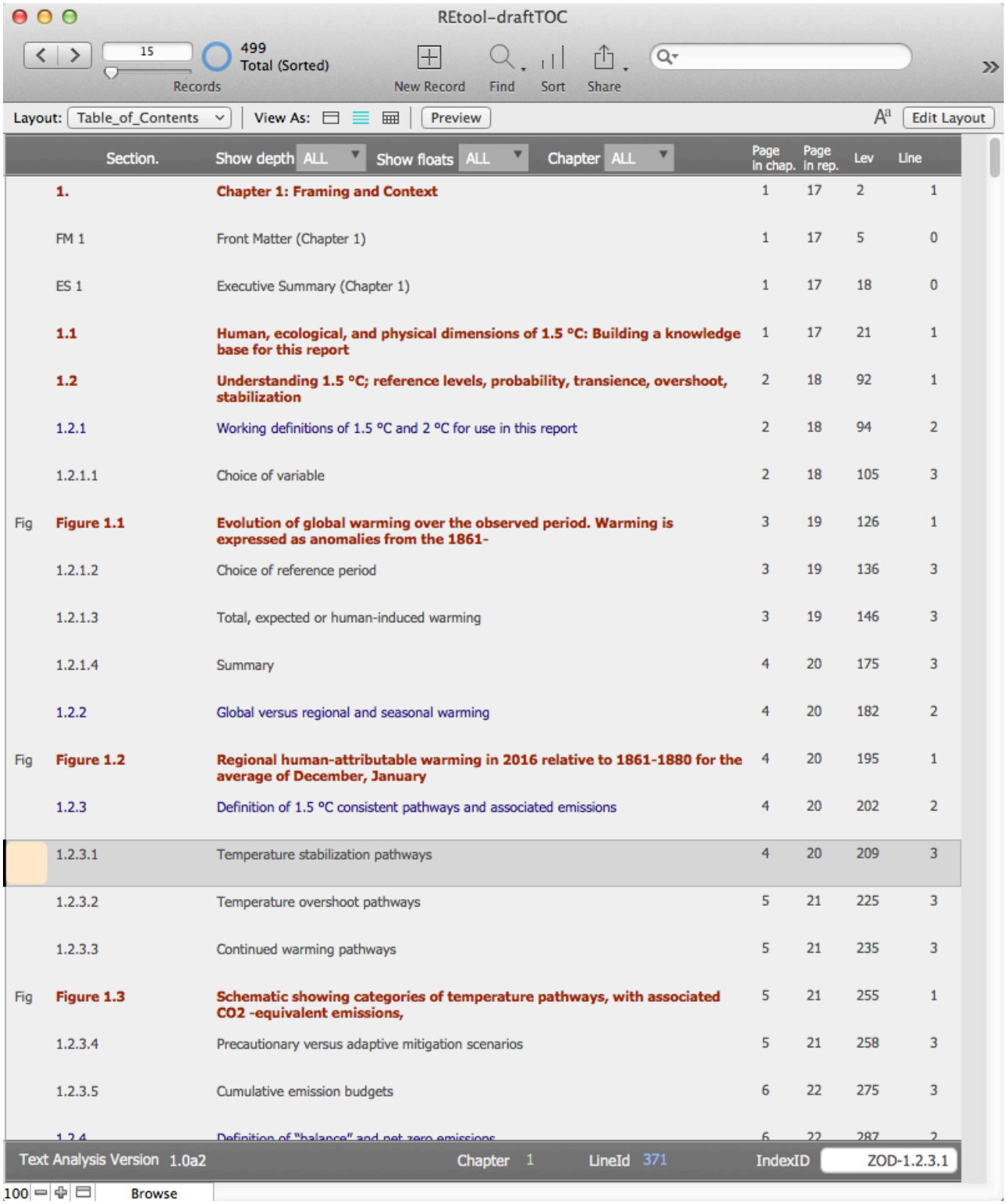
Parsing the TOC: Parsed 499 records and detected and exported 0 SPMs, 0 TSs, 5 chapters, 0 FMs, 5 ESs, 407 headings, 0 figures, 15 tables, 12 boxes, 0 FAQs, 0 footnotes, and 5 refs sections.

Note, the log shown is after having executed Cmd^6 (FM script 'Prepare for TOC Parsing') and Cmd^7 (FM script 'Parser - TOC', replace all data in 'TAT-draftTOC.fmp12'). Here how it looks in 'TAT-draftTOC.fmp12':



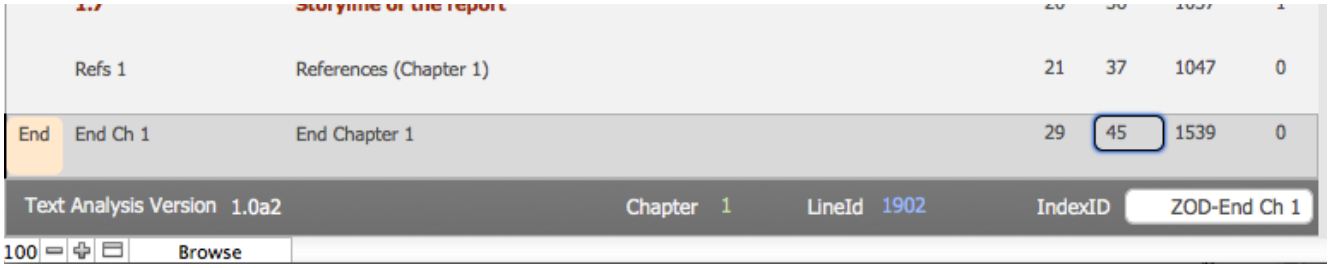
Note, many page and line numbers will be incorrect and may give only the page where the selected TOC entry was actually found. Thus, the test jumping (beige button at the very left) as supported from here does not work as intended. Only processing the actual chapter will fix this. Yet, all TOC elements such as titles, headings etc. should be fine and if a TOC would be processed that contains page numbers (not the case for SR1.5 ZOD), the jump near those chapters might already be possible.

Here the example of processing also chapter 1 similar to Example 1 (above) but using for ‘First_page_of_draft_core’ value 17 (where chapter 1 starts in the entire report’s pdf):



The beige jump button should now work and bring you immediately to the section «1.2.3.1 Temperature stabilization pathways” starting on line 209 of page 1-4 on page 20 of the

originally distributed ZOD.pdf (here only Skim is supported). For processing the next chapter correctly go to the chapter's end:



Page 46 (45+1) is the value to be used in field 'First_page_of_draft_core' (as used in Step 6) for the next chapter 2.

In the case of the IPCC SR1.5 ZOD, it might actually be in general a good idea to first process the data exactly as shown in the Example 3 by creating a clean data base (mode Replace of FM script 'Parser - TOC') and then add for each chapter (or the entire report at once, see next example) the additional meta data (mode Append of FM script 'Parser - TOC'). This use of redundancy gives an optimal check of consistency and makes it less likely to miss out on some data. For instance, I detected with this technique that the TOC in front of the report lists a section «3.2.4.1 Identifying hot spots», which does not exist in the chapter itself.

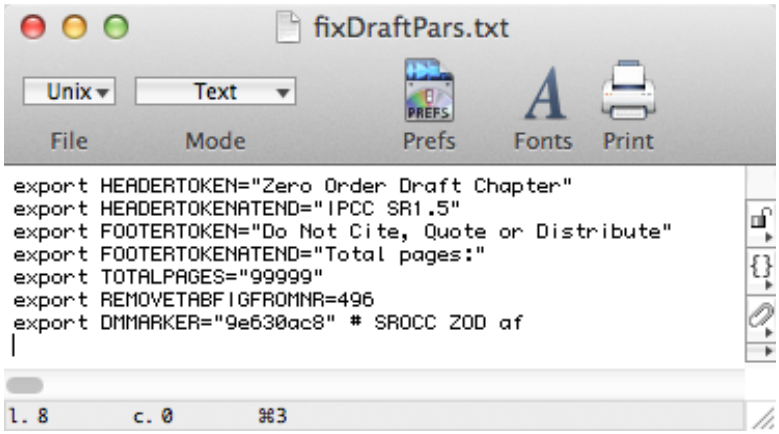
Example 4 - Process entire IPCC SR1.5 ZOD with front and back matter

Example 4 is for the entire IPCC SR1.5 ZOD report wide and quite elegant. The procedure is basically the same as the one given for above examples. Yet, the entire report can be imported in one go and then processed in three substeps to accomplish the same and more as if you would execute first Example 3, then Examples 1 and 2. Here the steps with their slight variations to what was previously described:

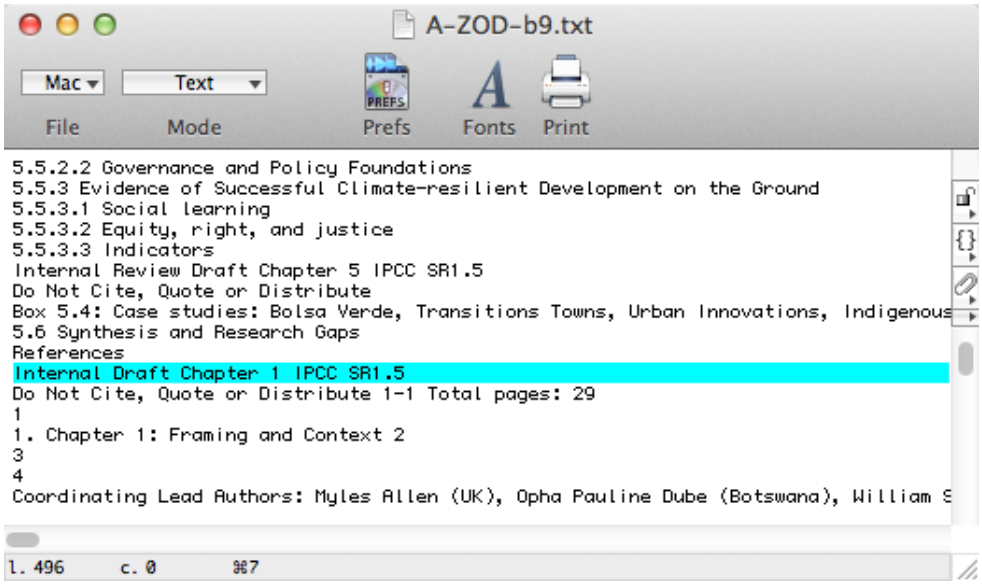
Step 1): Use method b9) to obtain the first text file using now the large ZOD.pdf as obtained from TSU (in this release nearby):

'A-ZOD-b9.txt'

Step 2): You can use the same file as used for Example 1.

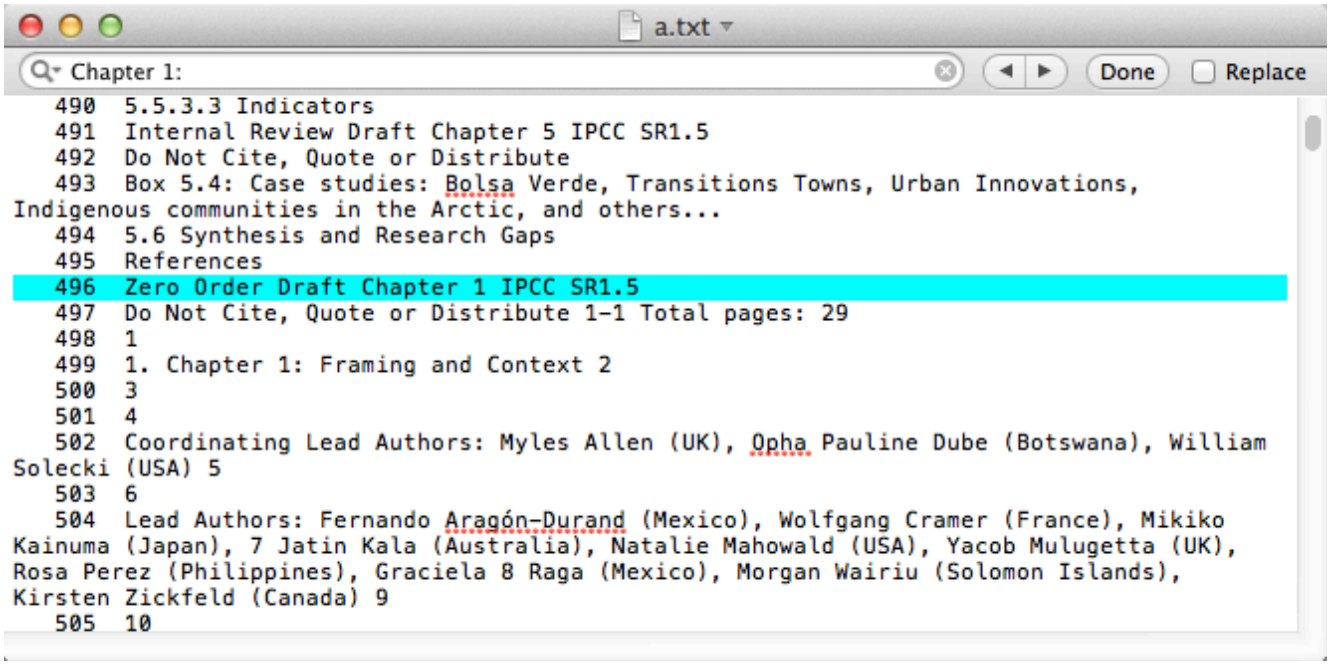


Note that the value for parameter REMOVETABFIGFROMNR is quite critical now. The value is the line where the core of the draft starts, i.e. the first record beyond the front matter. This line is highlighted in this snapshot (made with shareware editor AlphaX) and is the first header on page 1 of chapter 1:



The editor tells me the line number (left corner). In case you have no such tool available, the following Unix command allows you also to easily determine the wanted line number:

```
cat "A-ZOD-b9.txt" | massageDraft.sh | cat -n > a.txt ; open a.txt
```



Once you know the line, you can safely discard the file ‘a.txt’.

Why is it important to determine this value? First remember, under Example 3 you had to prepare the front matter manually. Now, we can avoid this. Secondly note, by default parameter ‘REMOVETABFIGFROMNR’ is set to the value 1 and therefore the entire file is

involved in the processing as done by rmFigures.awk and rmTables.awk, two important awk scripts that are normally called when using fixDraft.sh (**Step 3**). This would corrupt the table of contents (TOC) in the beginning of file ZOD.pdf, i.e. in the front matter (FM), whenever it lists a Figure or Table or Box Figure or Box Table matching the expected format. Calling fixDraft.sh with the options -g and -t would of course suppress the call to those two utilities and therefore avoid the corruption of the TOC in the FM. Yet, we need those utilities to do their processing later in the draft, or we will have hundreds of extra lines of text as resulting from figure legends or text contained in table cells (see e.g. **Step 4** under Example 2). Such lines are likely to interfere with the subsequent parsing. To prevent this corruption, yet be able to parse the entire file, the value of 496 for parameter 'REMOVETABFIGFROMNR' tells TAT to exclude all lines before line 496 from the full processing as done by fixDraft.sh, in particular to exclude all those lines before line 496 from any processing by rmFigures.awk and rmTables.awk. This helps later for **Substep A** (see below), which can then parse the TOC as contained in the FM. Note, if the FM would have line numbers like the rest of the ZOD, this would all be no issue. In general line numbers do prevent rmFigures.awk and rmTables.awk from doing anything. But this was not the case for the ZOD of the IPCC SR1.5 and lines without a line number are easily mistaken by rmFigures.awk and rmTables.awk to be table or figure lines that need to be removed.

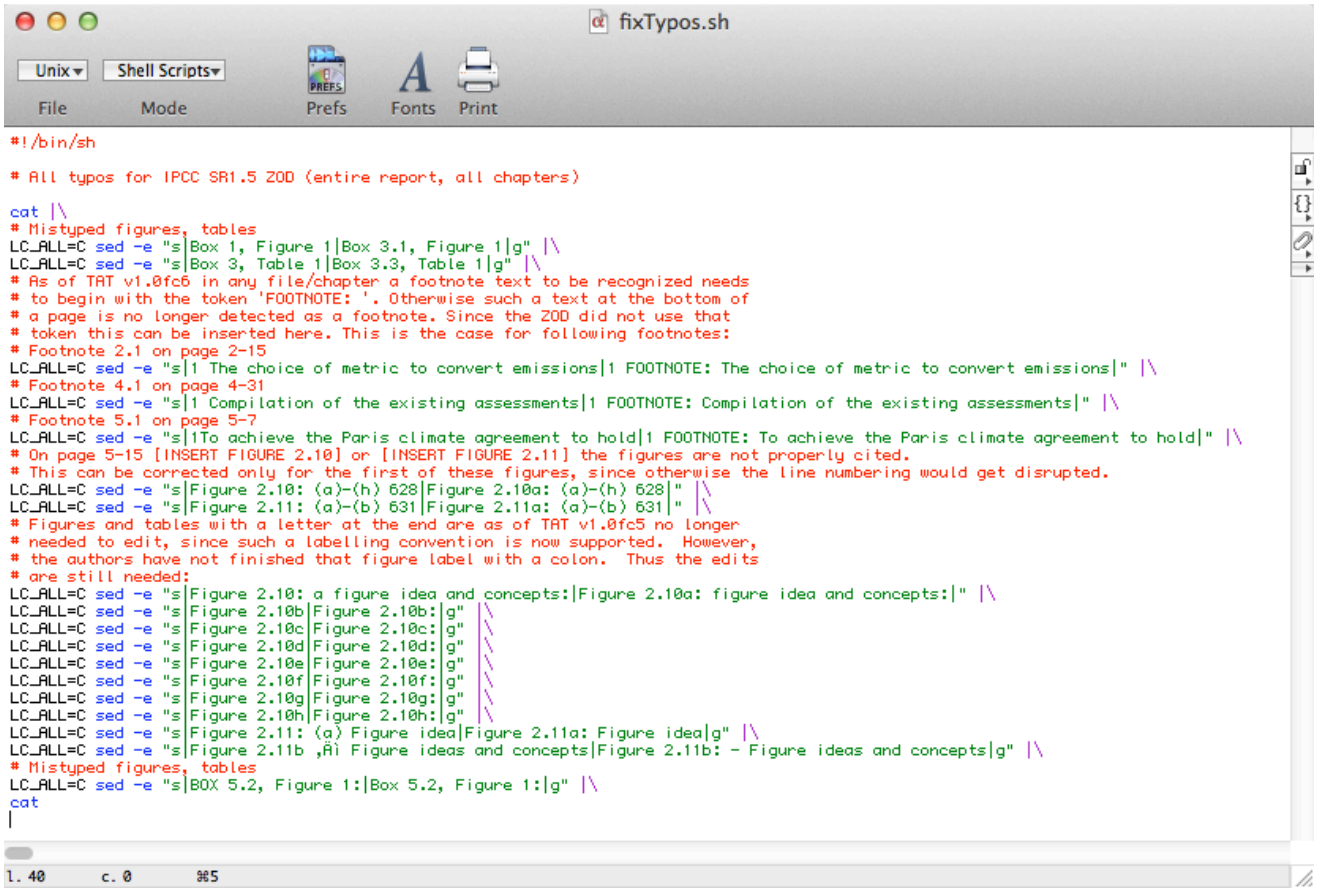
Step 3): The file needs some editing, mostly since authors have introduced several typos that impeded complete processing. My analysis of the ZOD showed the following issues are present in the ZOD that matter for TAT:

- In Ch3: 'Box 1, Figure 1:' -> 'Box 3.1, Figure 1:'
- In Ch3: 'Box 3, Table 1:' -> 'Box 3.3, Table 1:'
- In Ch2: Footnote on p. 2-15, after line 666 is swallowed by rmFigures.awk (fixDraft.sh -z, without option -g) unless protected by token 'FOOTNOTE '.
- In Ch5: Footnote 1, at the bottom page 5-7, between lines 361 and 362, is not recognized as footnote, i.e. there is no blank between the footnote number and the footnote text '1To achieve the Paris climate agreement'
- In back matter: 'Figure 2.10: a ' -> 'Figure 2.10a: '
- In back matter: 'Figure 2.10b' -> 'Figure 2.10b:'
- In back matter: 'Figure 2.10c' -> 'Figure 2.10c:'
- In back matter: 'Figure 2.10d' -> 'Figure 2.10d:'
- In back matter: 'Figure 2.10e' -> 'Figure 2.10e:'
- In back matter: 'Figure 2.10f' -> 'Figure 2.10f:'
- In back matter: 'Figure 2.10g' -> 'Figure 2.10g:'
- In back matter: 'Figure 2.10h' -> 'Figure 2.10h:'
- In back matter: 'Figure 2.11: (a) Figure idea' -> 'Figure 2.11a: Figure idea'
- In back matter: 'Figure 2.11b – Figure ideas and concepts' -> 'Figure 2.11b: – Figure ideas and concepts'
- In back matter: 'BOX 5.2, Figure 1:' -> 'Box 5.2, Figure 1:'

Note for the last edit, BOX' is not the same as 'Box' and not recognized by TAT until exactly written as specified. As of version 1.0b2 TAT offers you another technique to make the necessary edits. Before executing fixDraft.sh create a little text file to do the edits for you. This may be particularly handy if you suspect to stumble over further typos by the authors. Instead of making these edits by hand, simply create a little shell script named

'fixTypos.sh'

For this example it looks like this using simple sed commands:



Make it executable (e.g. chmod 777) and the shell script ‘fixDraft.sh’ will notice the presence of this script and execute it automatically. The result is that you should get the wanted input for ‘TAT-drafttext.fmp12’ in one simple go. Of course if you prefer to do it manually, you can also do it in the manner as described above for Example 2 and have the working directory without a file ‘fixTypos.sh’. However, in general, it is quite handy to have such a file present to overcome all remaining issues with a specific file. Moreoever, since some features of fixDraft.sh can conflict with each other, e.g. removing graph lines (default, suppressed with option -g) can occasionally remove a footnote or difficult to remove junk text lines are interpreted as footnotes (default, suppress recognition of footnotes with option -n) etc., this file can help to deal with all these problems elegantly and in a reversible manner while optimizing the outcome, since the original file ‘A-ZOD-b9.txt’ is always left untouched.

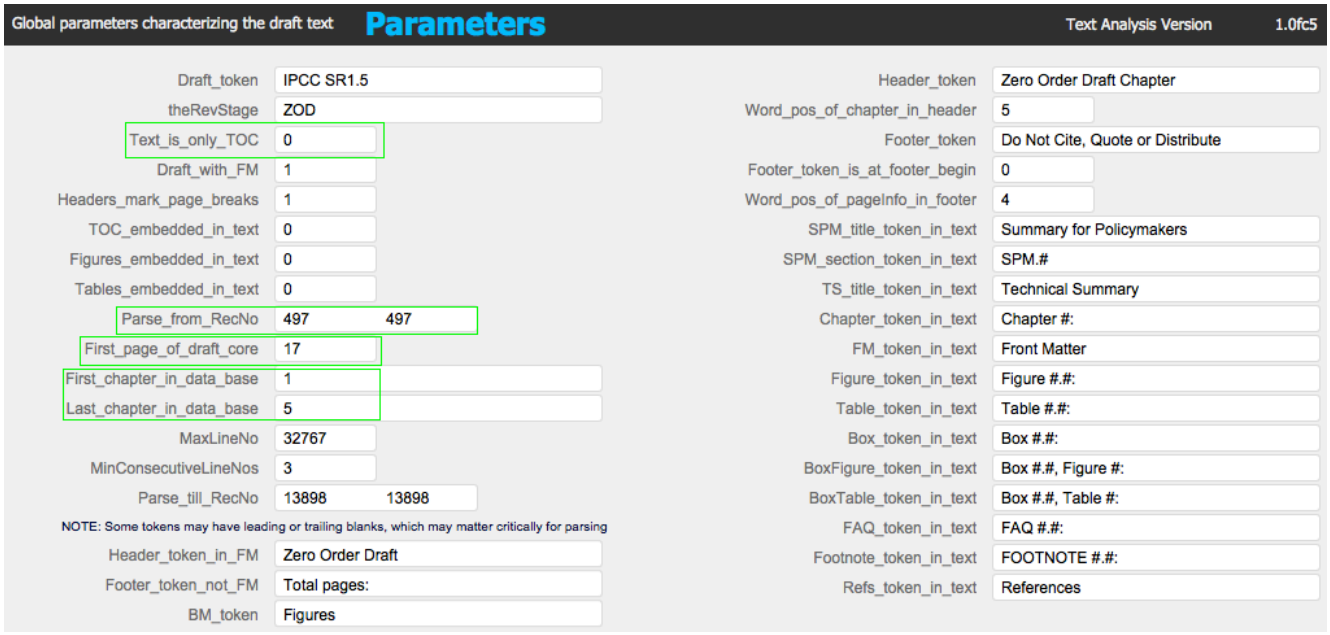
For this example using ‘fixTypos.sh’ as shown above the following command does all in one simple go:

```
fixDraft.sh -z -f "A-ZOD-b9.txt" ; open "A-ZOD-b9-OUT.txt"
```

Above commands should generate the wanted output, and allow you to inspect its quality thanks to the additional ‘open’ command.

Step 4): Iterations repeating Step 3 have identified the need for above fixTypos.sh script. However, not all typos show up immediately, i.e. from a mere inspection of the output of fixDraft.sh as done in previous examples under **Step 4**). Some of above typos show only up later and the reasons will be discussed below (under **Step 8**).

Step 5): As in previous Examples (replace all data) and import the entire ZOD from the file as generated during the previous step. Make sure the parameters are all correct. Set the parameter 'Text_is_only_TOC' back to 0 (false) before importing the data. Afterwards check that the parameter 'First_page_of_draft_core' shows the value 17, that the record to parse from is 497 ('Parse_from_RecNo') and that the first and last chapter are 1 and 5 ('First_chapter_in_data_base', 'Last_chapter_in_data_base'). If the import and subsequent processing worked fine, parameters should look as this (the aforementioned one with a green frame):

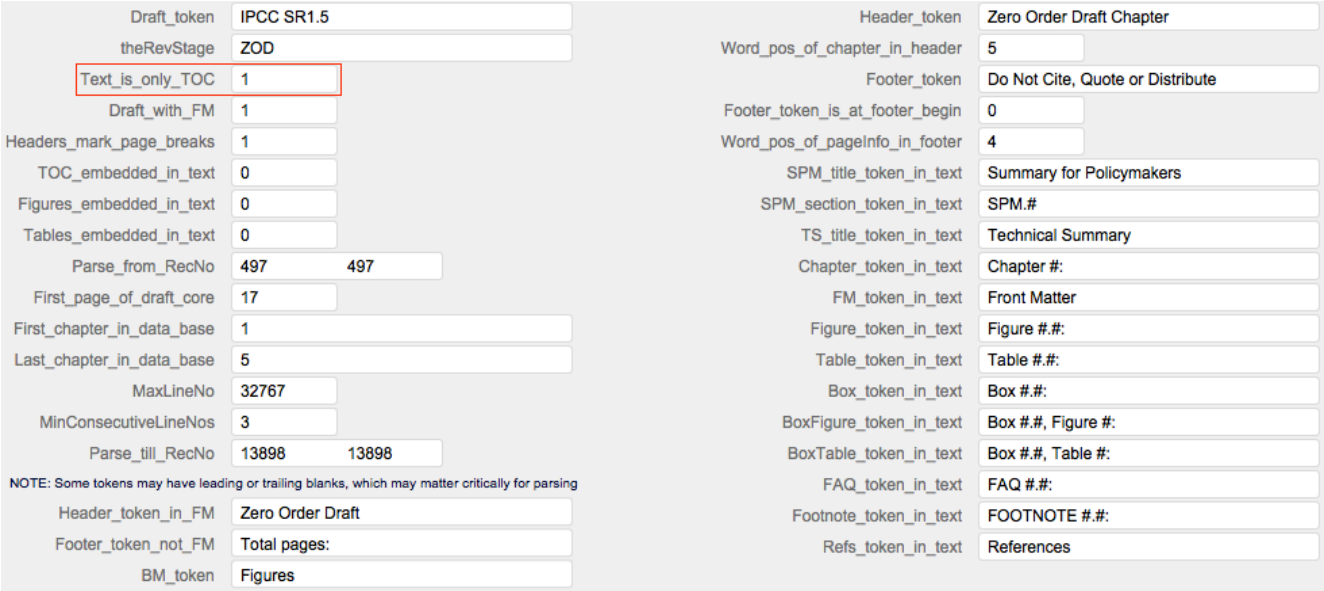


If 'First_page_of_draft_core' is not 17, then you need also to check the other parameters, notably those defining the relevant tokens in parameters 'Header_token_in_FM', 'Footer_token_not_FM', and 'Header_token'. They are used to determine the possible presence of the so-called front matter (FM), containing perhaps a table of content or some preamble text. Note, above values for these parameters mean that page headers from chapters, i.e. where the core of the text starts, are expected all to contain also the word 'Chapter' (in the given sequence), while headers in the front matter are expected to contain only the phrase 'Zero Order Draft'. Thanks to these parameter values the FM script 'Clean import and renumber pages' – which is called also by FM script 'Import draft text...' – should be able to determine front matter and determine that the first chapter starts at page 17 ('First_page_of_draft_core' shows value 17). You can change values of these token parameters and execute FM script 'Identify front (FM) and back (BM) matter' as many times you want. If you forgot to reset the value of parameter 'Text_is_onlh_TOC' to 0 (false) from previous example, you might also get a wrong 'First_page_of_draft_core'. If all is correct you should get the value 17 for 'First_page_of_draft_core' and the shown values for 'Parse_from_RecNo'. If you still get other values, consult the hints on how to set parameters as described under Example 1. It is worth ensuring all parameters are correct before proceeding.

Steps 6 and 7): Consists of 3 substeps processing different parts of the entire draft, first the front matter (FM), i.e. **Substep A**), then the chapters, i.e. **Substep B**), then the back matter

(BM), i.e. **Substep C**). This example does it all without having to reimport different data files in **Step 6**).

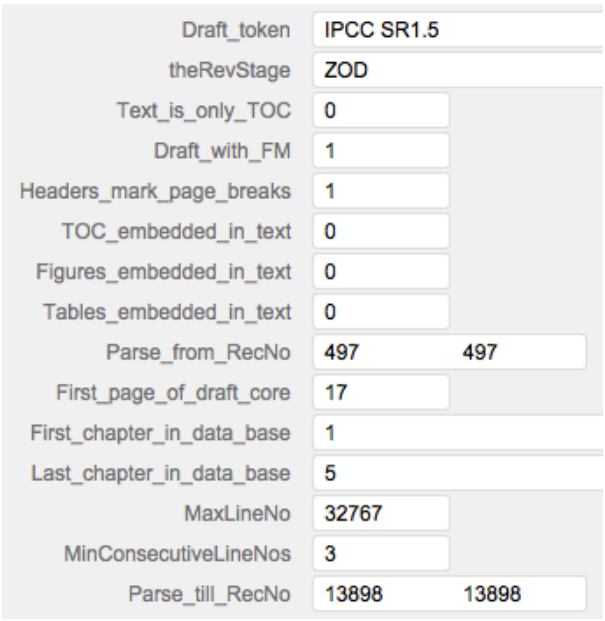
Substep A) Do first the same parsing as described under Example 3, i.e. parse the front matter (FM). To accomplish this you can skip **Step 6**), since the value has been correctly identified during **Step 5**). Simply set the parameter 'Text_is_only_TOC' to 1 (true). Your parameters should look then like this:



Draft_token	IPCC SR1.5		Header_token	Zero Order Draft Chapter
theRevStage	ZOD		Word_pos_of_chapter_in_header	5
Text_is_only_TOC	1		Footer_token	Do Not Cite, Quote or Distribute
Draft_with_FM	1		Footer_token_is_at_footer_begin	0
Headers_mark_page_breaks	1		Word_pos_of_pageInfo_in_footer	4
TOC_embedded_in_text	0		SPM_title_token_in_text	Summary for Policymakers
Figures_embedded_in_text	0		SPM_section_token_in_text	SPM.#
Tables_embedded_in_text	0		TS_title_token_in_text	Technical Summary
Parse_from_RecNo	497	497	Chapter_token_in_text	Chapter #:
First_page_of_draft_core	17		FM_token_in_text	Front Matter
First_chapter_in_data_base	1		Figure_token_in_text	Figure #.#:
Last_chapter_in_data_base	5		Table_token_in_text	Table #.#:
MaxLineNo	32767		Box_token_in_text	Box #.#:
MinConsecutiveLineNos	3		BoxFigure_token_in_text	Box #.#, Figure #:
Parse_till_RecNo	13898	13898	BoxTable_token_in_text	Box #.#, Table #:
NOTE: Some tokens may have leading or trailing blanks, which may matter critically for parsing			FAQ_token_in_text	FAQ #.#:
Header_token_in_FM	Zero Order Draft		Footnote_token_in_text	FOOTNOTE #.#:
Footer_token_not_FM	Total pages:		Refs_token_in_text	References
BM_token	Figures			

Then perform **Step 7**) by executing FM script 'Prepare front matter (FM) for TOC Parsing' (Cmd^8) instead of 'Prepare for TOC Parsing' (Cmd^6) followed by FM script 'Parser - TOC' (Cmd^7) while allowing for replacing all data. The result should be the same as what you would have obtained from Example 3.

Substep B): Again **Step 6**) can be skipped but to parse now all chapters you need to first set the parameter 'Text_is_only_TOC' back to 0 (false). Your parameters should like this:



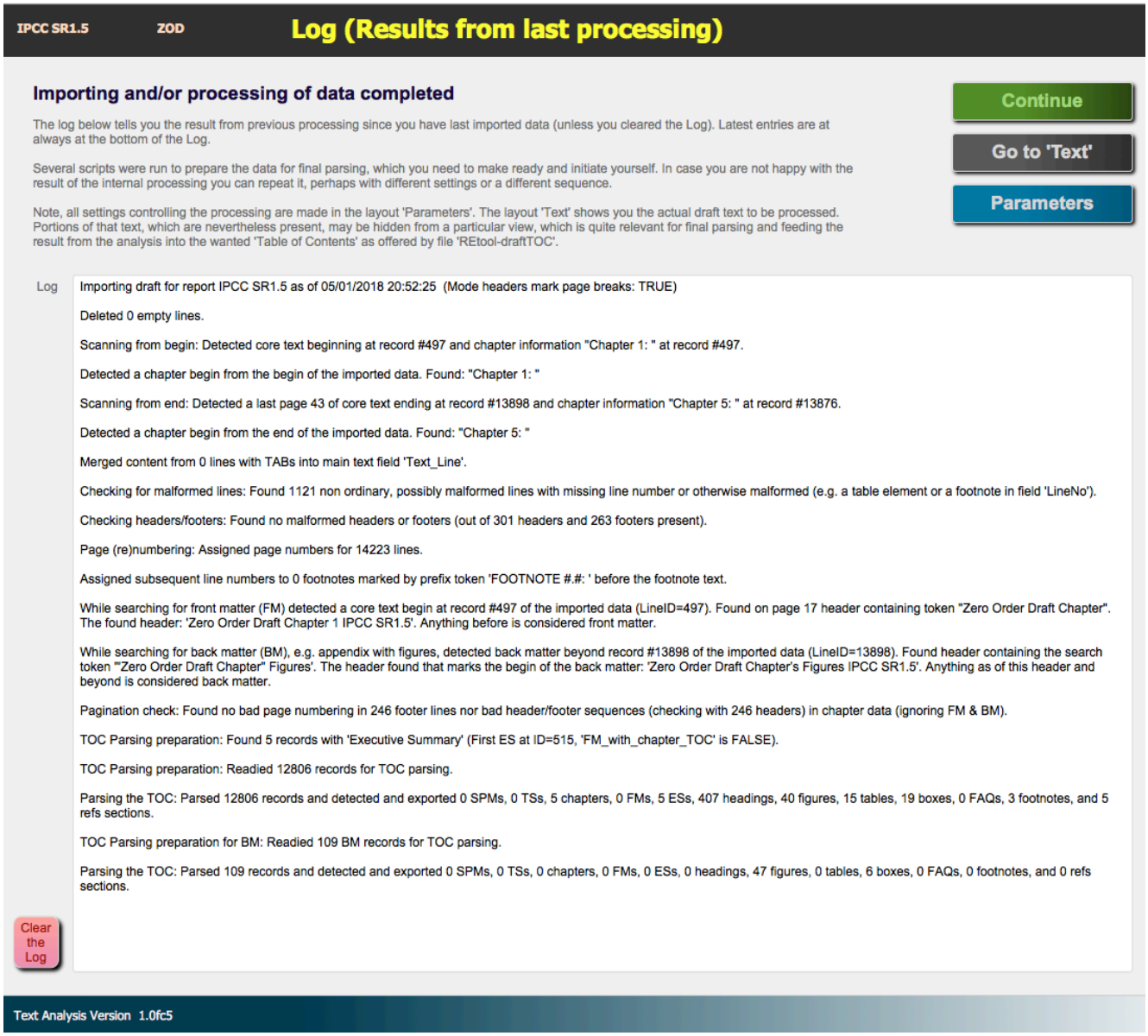
Draft_token	IPCC SR1.5	
theRevStage	ZOD	
Text_is_only_TOC	0	
Draft_with_FM	1	
Headers_mark_page_breaks	1	
TOC_embedded_in_text	0	
Figures_embedded_in_text	0	
Tables_embedded_in_text	0	
Parse_from_RecNo	497	497
First_page_of_draft_core	17	
First_chapter_in_data_base	1	
Last_chapter_in_data_base	5	
MaxLineNo	32767	
MinConsecutiveLineNos	3	
Parse_till_RecNo	13898	13898

Then perform **Step 7**) the usual way by executing FM scripts ‘Prepare for TOC Parsing’ (Cmd^6) followed by ‘Parser - TOC’ (Cmd^7), the latter of course by appending the data. That parsing may take quite a while, since it processes all five chapters.

Substep C): Process the back matter (BM).Accomplish this first by **Step 6**), i.e. by first setting the parameter ‘First_page_of_draft_core’ to 264. Your parameters should like this:

Draft_token	IPCC SR1.5	
theRevStage	ZOD	
Text_is_only_TOC	0	
Draft_with_FM	1	
Headers_mark_page_breaks	1	
TOC_embedded_in_text	0	
Figures_embedded_in_text	0	
Tables_embedded_in_text	0	
Parse_from_RecNo	497	497
First_page_of_draft_core	264	
First_chapter_in_data_base	1	
Last_chapter_in_data_base	5	
MaxLineNo	32767	
MinConsecutiveLineNos	3	
Parse_till_RecNo	13898	13898

Then perform **Step 7**) by executing the FM scripts ‘Prepare back matter (BM) for TOC Parsing’ (Cmd^9) – not ‘Prepare for TOC Parsing’ (Cmd^6) – followed by ‘Parser - TOC’ (Cmd^7), again of course by appending. The final log should look similar to this:



Step 8): First note, the effect of **Substep C**) is that the page numbers, where the actual figures can be found in the pdf, is entered in the field 'Page_MnNo' by overwriting the value that resulted from **Substep B**). However, the page number for the chapter is left as resulting from **Substep B**). This offers the advantage that you can jump to both locations, i.e. where a figure is to be inserted into the chapter text (click on page number in column 'Page in chap.', value of field 'Page_ChNo'), as well as where the actual figure can be found and looked at (click the usual button to the very left, or page number in column 'Page in rep.', i.e. the value of field 'Page_MnNo').

Secondly note, the typo 'Box 3, Figure 1' is unlikely to be detected in Example 2, albeit it is a similar typo as the one with 'Box 3, Table 1', where the authors should have written 'Box 3.3, Figure 1' (cf. Example 2). This is difficult to detect, since this figure is simply missing when processing only the chapter (**Substep B**). However, in this example where also the back matter was parsed (**Substep C**), it has become easy to detect this typo. The redundancy as contained in the back matter lets the parser enter an item for this figure, since in the back matter the figure has been properly labelled. Therefore it was added to the TOC in 'TAT-

draftTOC.fmp12', but of course without any information on where the figure should be inserted, neither the chapter page (field 'Page_ChNo', first number column) nor the line information (field 'Line_No', third number column). They could have retrieved only during **Substep B**). The TOC looks therefore like this (note the empty first and third number columns):

Only by adding the needed edit to 'fixTypos.sh' (see above, first edit) will this gap go away and after repeating the entire processing (**Steps 3**), **5**), i.e. fixDraft.sh, Cmd^1, **Substep B**), i.e. Cmd^6, Cmd^7, and **Substep C**), i.e. Cmd^9, Cmd^7) will the TOC now also look fine for Box 3.1, Figure 1 and all page numbers are as wanted:

Of course all additional parsing as described for this iteration is never done by replacing the data but merely by adding the meta data. You can safely iterate in this manner until all results are as desired. You need to watch out only for bad TOC entries, e.g. malformed chapter headings that conform to the standard, but are otherwise wrong, inexistent or malformed. Note, REtool uses a relationship where the IndexId in both tables have to match (it consists of the Index, 2nd column, and the draft, e.g. ZOD or FOD). If an orphaned Index is in your TOC data base in 'TAT-draftTOC.fmp12', e.g. because it is malformed, it will neither be removed nor updated unless you delete it or start to accumulate TOC data again from scratch as done in **Substep A**). Since parsing can be done quite efficiently, starting from scratch by replacing all meta data first to ensure no bad records from previous processing remain, is not such a big deal.

Step 9): Export from 'TAT-draftTOC.fmp12' all TOC meta data for the entire IPCC SR1.5 ZOD to a spreadsheet (is provided nearby for comparison reasons).

Note, the nearby distributed FileMaker (FM) files 'TAT-drafttext.fmp12' and 'TAT-draftTOC.fmp12' contain the records and are in a state of having done Example 4.