The RJafroc Quick Start Book

Dev P. Chakraborty, PhD

2023-02-08

Contents

Chapter 1

Preface

- This online book is for those already somewhat familiar running Windows JAFROC to analyze data. It is also intended for those unfamiliar with Windows JAFROC but have read the other books and wish to apply the methods.
- The Windows program has been replaced by `RJafroc`.
- This book dives into how to use `RJafroc` to analyze ROC/FROC data.
- It starts with explanation of the dataset structures for ROC and FROC studies.
- This is followed by an explanation of DBM and OR analyses.
- TBA.

1.1 Rationale and Organization

- Intended as an online update to my print book (Chakraborty, 2017).
- All references in this book to `RJafroc` refer to the R package with that name (case sensitive) (Chakraborty and Zhai, 2022).
- Since its publication in 2017 `RJafroc`, on which the `R` code examples in the print book depend, has evolved considerably causing many of the examples to "break" if one uses the most current version of `RJafroc`. The code will still run if one uses `RJafroc` 0.0.1 but this is inconvenient and misses out on many of the software improvements made since the print book appeared.
- This gives me the opportunity to update the print book.
- The online book has been divided into 3 books.
    - **This book:** RJafrocQuickStartBook.
    - The RJafrocRocBook book.
    - The RJafrocFrocBook.

1.2 Getting help on the software

- If you have installed `RJafroc` from `GitHub`:
    - ?`RJafroc-package` (RStudio will auto complete …) followed by Enter.
    - Scroll down all the way and click on `Index`
- Regardless of where you installed from you can use the `RJafroc` website:
    - RJafroc help site
    - Look under the `Reference` tab.
    - For example, for help on the function `PlotEmpiricalOperatingCharacteristics`:
    - PlotEmpiricalOperatingCharacteristics

1.3 TBA Acknowledgements

1.3.1 Persons who have stimulated my thinking:

Harold Kundel, MD

Claudia Mello-Thoms, PhD

Dr. Xuetong Zhai (contributed significantly to the significance testing sections and other chapters of my book).

1.3.2 Contributors to the software

Dr. Xuetong Zhai (he developed the first version of `RJafroc`)

Dr. Peter Phillips

Online Latex Editor at this website. I found this very useful in learning and using Latex to write math equations.

1.3.3 Dataset contributors

TBA

1.4 Accessing files and code

To access files/code one needs to `fork` the `GitHub` repository. This will create, on your computer, a copy of all files used to create this document. To compile the files try `Build Book` and select `gitbook`. You will probably get errors corresponding to missing packages that are not loaded on your machine. All required packages are listed in the DESCRIPTION file. Install those packages and try again …

Quick Start

Chapter 2

JAFROC ROC data

2.1 How much finished

50% (remove duplication)

2.2 Introduction

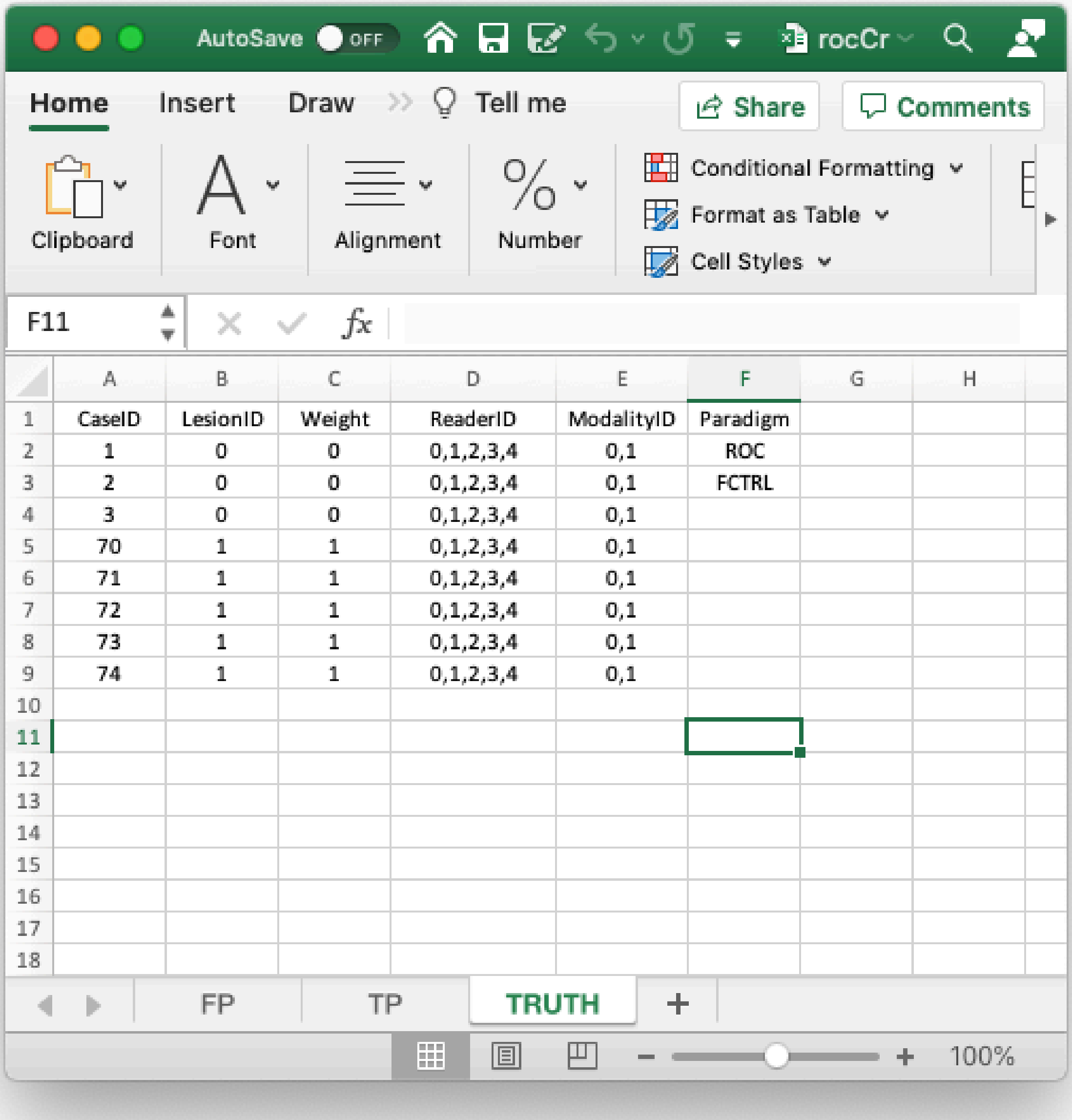
- The JAFROC Excel data format was adopted circa. 2006. The purpose of this chapter is to explain the format of this file.

2.3 Note to existing users

- The Excel file format has recently undergone changes involving three additional columns in the `Truth` worksheet. The changes are needed for easier generalization to other data collection paradigms (e.g., split plot designs) and for better data entry error control.
- `RJafroc` will work with original format Excel files provided the `NewExcelFileFormat` flag is set to `FALSE`, the default.
- Going forward, one should use the new format, described below, and use `NewExcelFileFormat = TRUE` to read the file.

2.4 Excel ROC file format

- The illustrations in this chapter correspond to Excel file `R/quick-start/rocCr.xlsx` in the project directory. See Section @ref(#quick-start-index-how-to-access-files) for how to get this file, and all other files and code in this `bookdown` book, to your computer.
- This is a *toy file*, i.e., an artificial small dataset used to illustrate essential features of the data format.
- The Excel file has three worksheets: `Truth`, `NL` (or `FP`) and `LL` (or `TP`). The worksheet names are case insensitive.



CaseID	LesionID	Weight	ReaderID	ModalityID	Paradigm
1	0	0	0,1,2,3,4	0,1	ROC
2	0	0	0,1,2,3,4	0,1	FCTRL
3	0	0	0,1,2,3,4	0,1	
70	1	1	0,1,2,3,4	0,1	
71	1	1	0,1,2,3,4	0,1	
72	1	1	0,1,2,3,4	0,1	
73	1	1	0,1,2,3,4	0,1	
74	1	1	0,1,2,3,4	0,1	

2.4.1 The `Truth` worksheet

- The `Truth` worksheet contains 6 columns: `CaseID`, `LesionID`, `Weight`, `ReaderID`, `ModalityID` and `Paradigm`.
- `CaseID`: **unique integers**, one per case, representing the cases in the dataset. In the current dataset, the non-diseased cases are labeled `1`, `2` and `3`, while the diseased cases are labeled `70`, `71`, `72`, `73` and `74`. The values do not have to be consecutive integers; they need not be ordered; the only requirement is that they be **unique integers**.
- `LesionID`: integers 0 or 1, with each 0 representing a non-diseased case and each 1 representing a diseased case.
- `Weight`: this field is not used for ROC data.
- `ReaderID`: a **comma-separated** string containing the reader labels, each represented by a **unique integer**, that have interpreted the case. In the example shown below each cell has the value `0, 1, 2, 3, 4` meaning

that each of these readers has interpreted all cases.

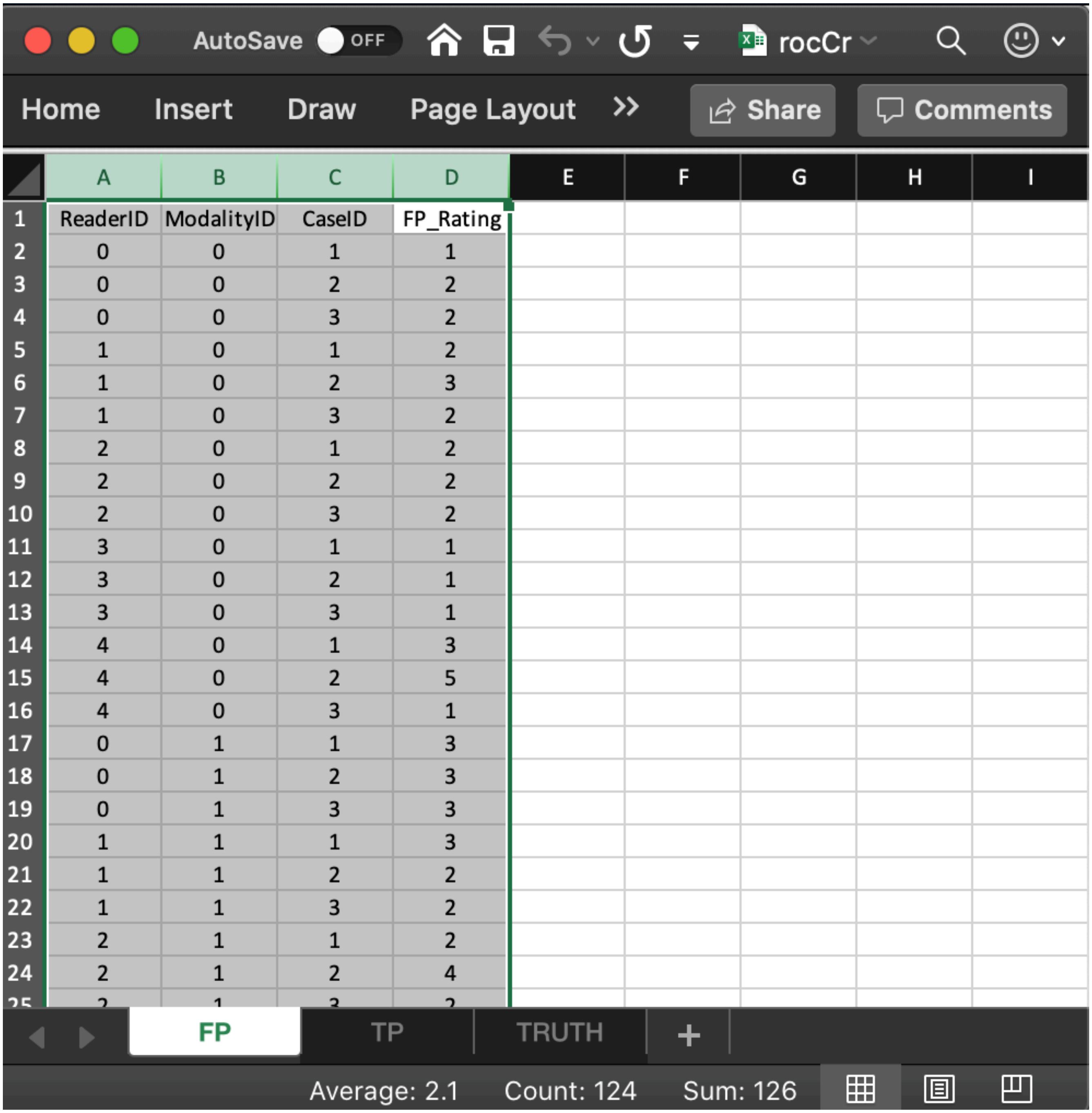
  - **With multiple readers each cell in this column has to be text formatted as otherwise Excel will not accept it.**
  - Select the worksheet, then `Format` - `Cells` - `Number` - `Text` - `OK`.

- `ModalityID`: a comma-separated string containing the modality labels, each represented by a **unique integer**. In the example each cell has the value `0, 1`.

  - **With multiple modalities each cell has to be text formatted as otherwise Excel will not accept it.**
  - Format the cells as described above.

- `Paradigm`: this column contains two cells, `ROC` and `factorial`. It informs the software that this is an ROC dataset, and the design is factorial, meaning each reader has interpreted each case in each modality.
- There are 5 diseased cases in the dataset (the number of 1's in the `LesionID` column of the `Truth` worksheet).
- There are 3 non-diseased cases in the dataset (the number of 0's in the `LesionID` column).
- There are 5 readers in the dataset (each cell in the `ReaderID` column contains the string `0, 1, 2, 3, 4`).
- There are 2 modalities in the dataset (each cell in the `ModalityID` column contains the string `0, 1`).

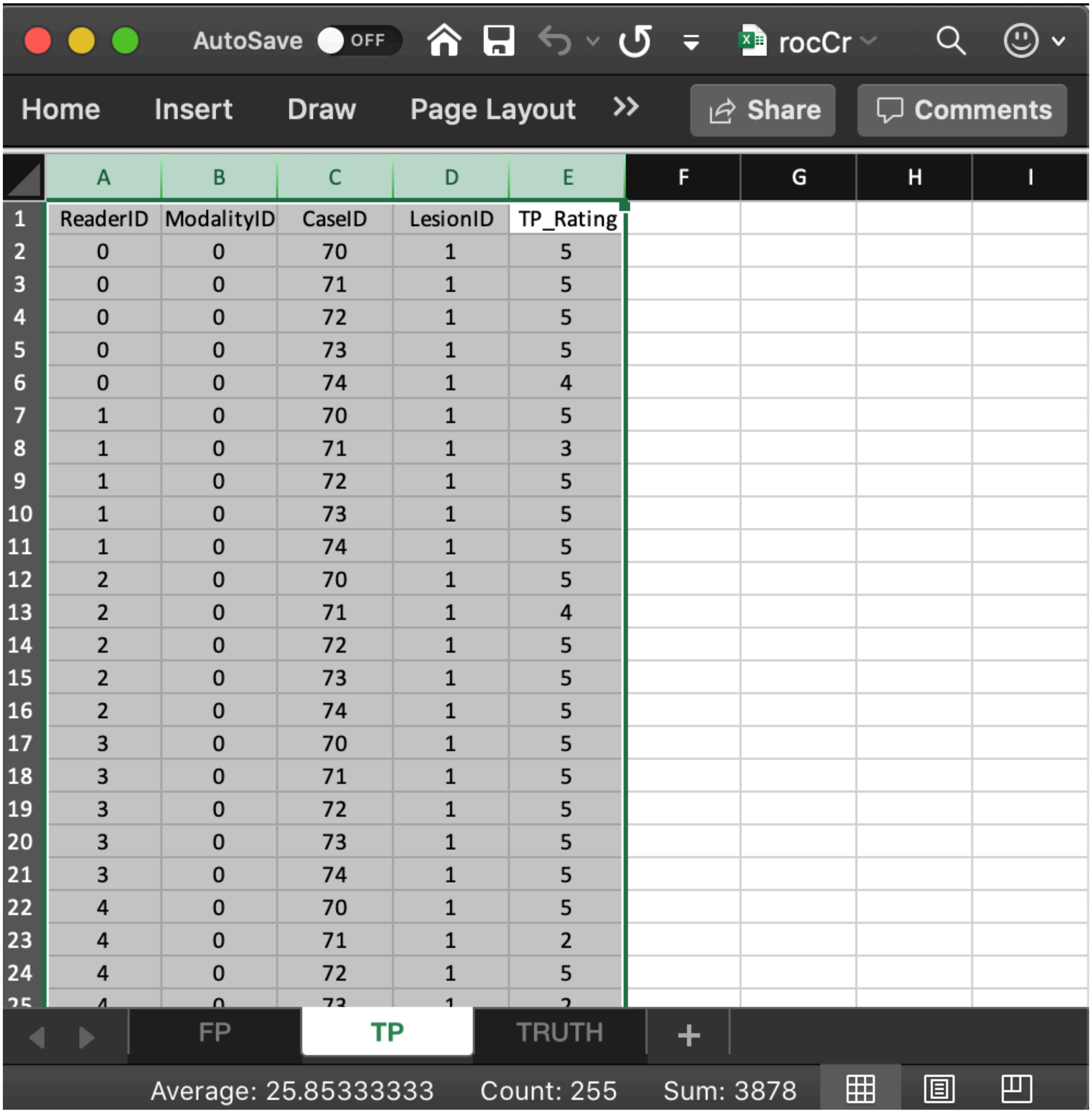
2.4.2 The false positive (FP/NL) worksheet



	A	B	C	D
1	ReaderID	ModalityID	CaseID	FP_Rating
2	0	0	1	1
3	0	0	2	2
4	0	0	3	2
5	1	0	1	2
6	1	0	2	3
7	1	0	3	2
8	2	0	1	2
9	2	0	2	2
10	2	0	3	2
11	3	0	1	1
12	3	0	2	1
13	3	0	3	1
14	4	0	1	3
15	4	0	2	5
16	4	0	3	1
17	0	1	1	3
18	0	1	2	3
19	0	1	3	3
20	1	1	1	3
21	1	1	2	2
22	1	1	3	2
23	2	1	1	2
24	2	1	2	4
25	2	1	3	2

- It consists of 4 columns, each of length 30 (# of modalities x number of readers x number of non-diseased cases).
- `ReaderID`: the reader labels: `0`, `1`, `2`, `3` and `4`. Each reader label occurs 6 times (# of modalities x number of non-diseased cases).
- `ModalityID`: the modality or treatment labels: `0` and `1`. Each label occurs 15 times (# of readers x number of non-diseased cases).
- `CaseID`: the case labels for non-diseased cases: `1`, `2` and `3`. Each label occurs 10 times (# of modalities x # of readers).
- The label of a diseased case cannot occur in the FP worksheet. If it does the software generates an error.
- `FP_Rating`: the floating point ratings of non-diseased cases. Each row of this worksheet contains a rating corresponding to the values of `ReaderID`, `ModalityID` and `CaseID` for that row.

2.4.3 The true positive (TP/LL) worksheet



	A	B	C	D	E
1	ReaderID	ModalityID	CaseID	LesionID	TP_Rating
2	0	0	70	1	5
3	0	0	71	1	5
4	0	0	72	1	5
5	0	0	73	1	5
6	0	0	74	1	4
7	1	0	70	1	5
8	1	0	71	1	3
9	1	0	72	1	5
10	1	0	73	1	5
11	1	0	74	1	5
12	2	0	70	1	5
13	2	0	71	1	4
14	2	0	72	1	5
15	2	0	73	1	5
16	2	0	74	1	5
17	3	0	70	1	5
18	3	0	71	1	5
19	3	0	72	1	5
20	3	0	73	1	5
21	3	0	74	1	5
22	4	0	70	1	5
23	4	0	71	1	2
24	4	0	72	1	5
25	4	0	73	1	2

Average: 25.85333333 Count: 255 Sum: 3878

- It consists of 5 columns, each of length 50 (# of modalities x number of readers x number of diseased cases).
- `ReaderID`: the reader labels: `0`, `1`, `2`, `3` and `4`. Each reader label occurs 10 times (# of modalities x number of diseased cases).
- `ModalityID`: the modality or treatment labels: `0` and `1`. Each label occurs 25 times (# of readers x number of diseased cases).
- `LesionID`: For an ROC dataset this column contains fifty 1's (each diseased case has one lesion).
- `CaseID`: the case labels for non-diseased cases: `70`, `71`, `72`, `73` and `74`. Each label occurs 10 times (# of modalities x # of readers). For an ROC dataset the label of a non-diseased case cannot occur in the TP worksheet. If it does the software generates an error.
- `TP_Rating`: the floating point ratings of diseased cases. Each row of this worksheet contains a rating corresponding to the values of `ReaderID`, `ModalityID`, `LesionID` and `CaseID` for that row.

2.5 Reading the Excel file

The following code reads the Excel file and saves it to object `x`.

```
x <- DfReadDataFile("R/quick-start/rocCr.xlsx", newExcelFileFormat = TRUE)
```

- `newExcelFileFormat` is set to `TRUE` as otherwise columns D - F in the `Truth` worksheet are ignored and the dataset is assumed to be factorial, with `dataType` "automatically" determined from the contents of the FP and TP worksheets. [1]
- Flag `newExcelFileFormat = FALSE`, the default, is for compatibility with the original JAFROC format Excel format, which did not have columns D - F in the `Truth` worksheet. Its usage is deprecated.

2.6 Structure of dataset object

Most users will not need to be concerned with the internal structure of the dataset object `x`. For those interested in it, for my reference, and for ease of future maintenance of the software, this is deferred to Section 16.2.1.

[1] The assumptions underlying the "automatic" determination could be defeated by data entry errors.

Chapter 3

JAFROC FROC data

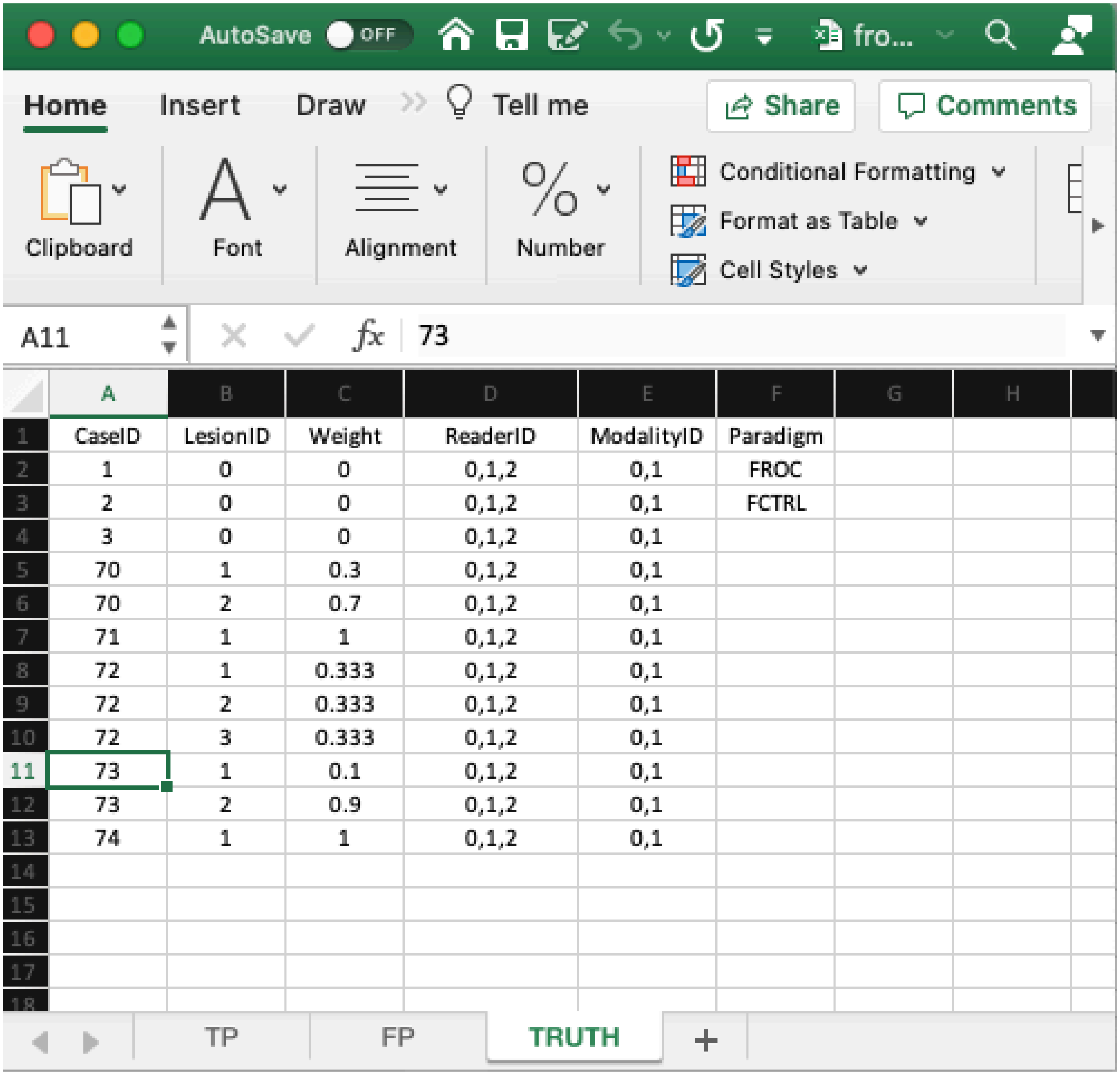
3.1 TBA How much finished

90%

3.2 Introduction

The chapter is illustrated with a toy data file, `R/quick-start/frocCr.xlsx` in which readers '0', '1' and '2' interpret 8 cases in two modalities, '0' and '1'. The design is 'factorial', abbreviated to `FCTRL` in the software; this is also termed a 'fully-crossed' design. The Excel file has three worksheets named `Truth`, `NL` (or `FP`) and `LL` (or `TP`). The names are case-insensitive.

3.3 The Truth worksheet



CaseID	LesionID	Weight	ReaderID	ModalityID	Paradigm
1	0	0	0,1,2	0,1	FROC
2	0	0	0,1,2	0,1	FCTRL
3	0	0	0,1,2	0,1	
70	1	0.3	0,1,2	0,1	
70	2	0.7	0,1,2	0,1	
71	1	1	0,1,2	0,1	
72	1	0.333	0,1,2	0,1	
72	2	0.333	0,1,2	0,1	
72	3	0.333	0,1,2	0,1	
73	1	0.1	0,1,2	0,1	
73	2	0.9	0,1,2	0,1	
74	1	1	0,1,2	0,1	

- The `Truth` worksheet contains 6 columns: `CaseID`, `LesionID`, `Weight`, `ReaderID`, `ModalityID` and `Paradigm`.
- Since a diseased case may have more than one lesion, the first five columns contain **at least** as many rows as there are cases (images) in the dataset. There are 8 cases in the dataset and 12 rows of data, because some of the diseased cases contain more than one lesion.
- `CaseID`: unique **integers** representing the cases in the dataset: '1', '2', '3', the 3 non-diseased cases, and '70', '71', '72', '73', '74', the 5 diseased cases. The ordering of the numbers is inconsequential. [1]
- `LesionID`: integers 0, 1, 2, etc.,
  - Each 0 represents a non-diseased case,
  - Each 1 represents the *first* lesion on a diseased case, 2 the *second* lesion, if present, and so on.
  - This field is zero for non-diseased cases '1', '2', '3'.

[1] `CaseID` should not be so large that it cannot be represented in Excel by an integer; to be safe use unsigned short 8-bit integers. For example, 108057200 or 9971103254 are too large to be a valid `caseID` and may cause errors.

  - For the first diseased case, i.e., ‘70’, it is 1 for the first lesion and 2 for the second lesion.
  - For the second diseased case i.e., ‘71’, it is 1, as this case has only one lesion.
  - For the third diseased case, i.e., ‘72’, it is 1 for the first lesion, 2 for the second lesion and 3 for the third lesion.
  - For the fourth diseased case, i.e., ‘73’, it is 1 for the first lesion and 2 for the second lesion.
  - For the fifth diseased case i.e., ‘74’, it is 1, as this case has only one lesion.

- There are 3 non-diseased cases in the dataset (the number of 0’s in the `LesionID` column).
- There are 5 diseased cases in the dataset (the number of 1’s in the `LesionID` column).
- `Weight` or clinical importance - e.g., mortality associated with lesion:
  - non-negative floating point values
  - 0 for each non-diseased case
  - For each diseased case values that sum to unity.
  - A shortcut to assigning equal weights to all lesions in a case is to fill the `Weight` column with zeroes.
- `LesionID`
  - Diseased case `70` has two lesions, with `LesionID`s ‘1’ and ‘2’, and weights 0.3 and 0.7.
  - Diseased case `71` has one lesion, with `LesionID` = 1, and `Weight` = 1.
  - Diseased case `72` has three lesions, with `LesionID`s 1, 2 and 3 and weights 1/3 each.
  - Diseased case `73` has two lesions, with `LesionID`s 1, and 2 and weights 0.1 and 0.9.
  - Diseased case `74` has one lesion, with `LesionID` = 1 and `Weight` = 1.
- `ReaderID`: a comma-separated listing of readers, each represented by a unique **text label**, that have interpreted the case. In the example shown below each cell has the value ‘0, 1, 2’.
- There are 3 readers in the dataset, as each cell in the `ReaderID` column contains ‘0, 1, 2’.
- `ModalityID`: a comma-separated listing of modalities (or treatments), each represented by a unique **integer**, that apply to each case. In the example each cell has the value `0, 1`. **Each cell has to be text formatted.**
- There are 2 modalities in the dataset, as each cell in the `ModalityID` column contains ‘0, 1’.
- `Paradigm`: The contents are `FROC` and `FCTRL`: this is an `FROC` dataset and the design is “factorial”.

3.4 Reading the FROC dataset

The example shown above corresponds to file `R/quick-start/frocCr.xlsx` in the project directory. The next code reads this file into an `R` object `x`.

```
frocCr <- "R/quick-start/frocCr.xlsx"
x <- DfReadDataFile(frocCr, newExcelFileFormat = TRUE)
str(x)
#> List of 3
#>  $ ratings     :List of 3
#>   ..$ NL   : num [1:2, 1:3, 1:8, 1:2] 1.02 2.89 2.21 3.01 2.14 ...
#>   ..$ LL   : num [1:2, 1:3, 1:5, 1:3] 5.28 5.2 5.14 4.77 4.66 4.87 3.01 3.27 3.31 3.19 ...
#>   ..$ LL_IL: logi NA
#>  $ lesions     :List of 3
#>   ..$ perCase: int [1:5] 2 1 3 2 1
#>   ..$ IDs    : num [1:5, 1:3] 1 1 1 1 1 ...
#>   ..$ weights: num [1:5, 1:3] 0.3 1 0.333 0.1 1 ...
#>  $ descriptions:List of 7
#>   ..$ fileName     : chr "frocCr"
#>   ..$ type         : chr "FROC"
#>   ..$ name         : logi NA
#>   ..$ truthTableStr: num [1:2, 1:3, 1:8, 1:4] 1 1 1 1 1 1 1 1 1 1 ...
#>   ..$ design       : chr "FCTRL"
#>   ..$ modalityID   : Named chr [1:2] "0" "1"
#>   .. ..- attr(*, "names")= chr [1:2] "0" "1"
#>   ..$ readerID     : Named chr [1:3] "0" "1" "2"
#>   .. ..- attr(*, "names")= chr [1:3] "0" "1" "2"
```

This follows the general description in Chapter 2. The differences are described below.

- The `x$descriptions$type` member indicates that this is an `FROC` dataset.
- The `x$lesions$perCase` member is a vector whose contents reflect the number of lesions in each diseased case, i.e., 2, 1, 3, 2, 1 in the current example.
- The `x$lesions$IDs` member indicates the labeling of the lesions in each diseased case.

```
x$lesions$IDs
#>      [,1] [,2] [,3]
#> [1,]    1    2 -Inf
#> [2,]    1 -Inf -Inf
#> [3,]    1    2    3
#> [4,]    1    2 -Inf
#> [5,]    1 -Inf -Inf
```

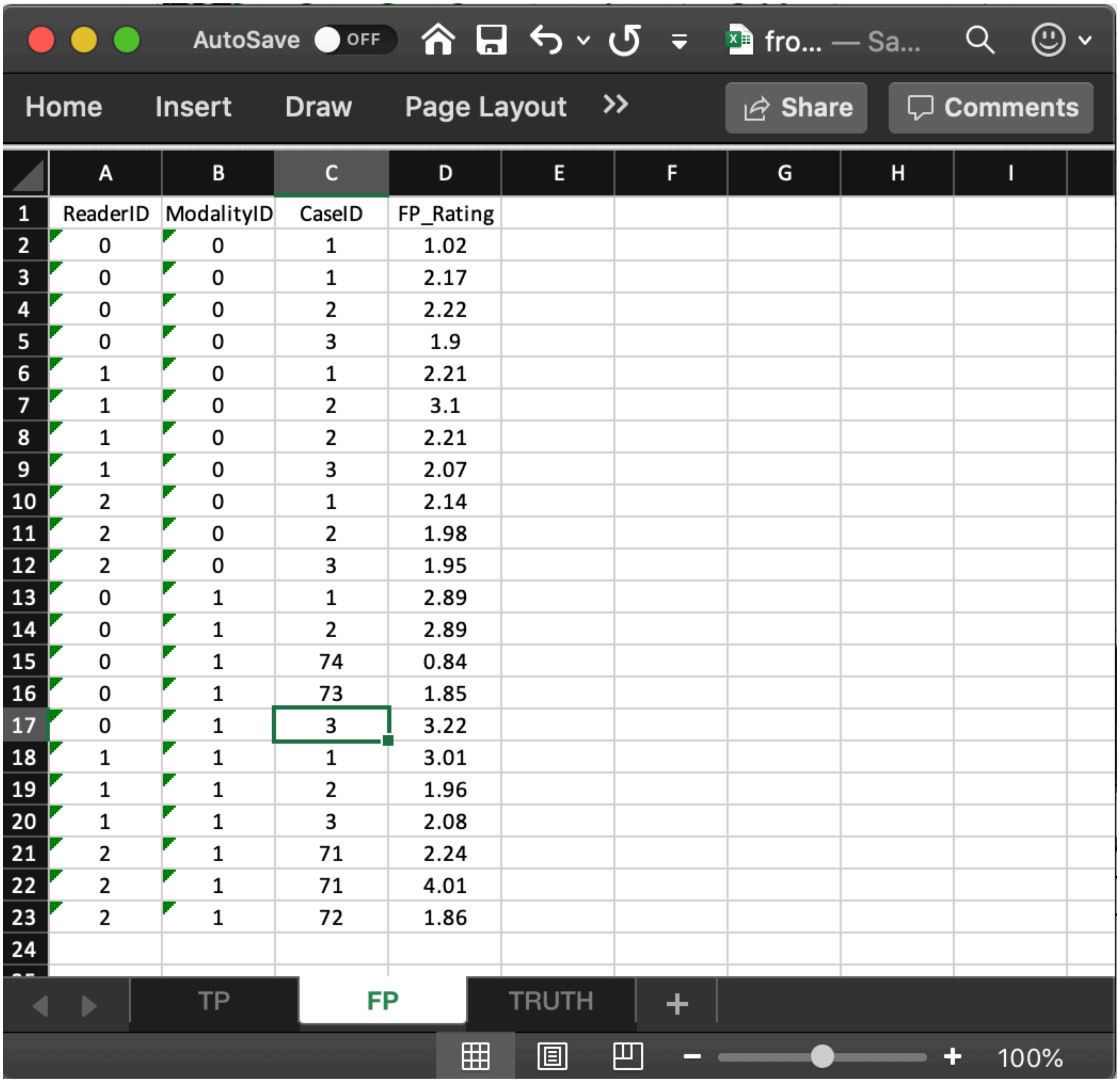
- This shows that the lesions on the first diseased case are labeled '1' and '2'. The `-Inf` is a filler used to denote a missing value. The second diseased case has one lesion labeled '1'. The third diseased case has three lesions labeled '1', '2' and '3', etc.
- The `lesionWeight` member is the clinical importance of each lesion. Lacking specific clinical reasons, the lesions should be equally weighted; this is *not* true for this toy dataset.

```
x$lesions$weights
#>           [,1]      [,2]      [,3]
#> [1,] 0.3000000 0.7000000      -Inf
#> [2,] 1.0000000      -Inf      -Inf
#> [3,] 0.3333333 0.3333333 0.3333333
#> [4,] 0.1000000 0.9000000      -Inf
#> [5,] 1.0000000      -Inf      -Inf
```

- The first diseased case has two lesions, the first has weight 0.3 and the second has weight 0.7.
- The second diseased case has one lesion with weight 1.
- The third diseased case has three equally weighted lesions, each with weight 1/3. Etc.

3.5 The false positive (FP) ratings

These are found in the `FP` or `NL` worksheet.



	A	B	C	D
1	ReaderID	ModalityID	CaseID	FP_Rating
2	0	0	1	1.02
3	0	0	1	2.17
4	0	0	2	2.22
5	0	0	3	1.9
6	1	0	1	2.21
7	1	0	2	3.1
8	1	0	2	2.21
9	1	0	3	2.07
10	2	0	1	2.14
11	2	0	2	1.98
12	2	0	3	1.95
13	0	1	1	2.89
14	0	1	2	2.89
15	0	1	74	0.84
16	0	1	73	1.85
17	0	1	3	3.22
18	1	1	1	3.01
19	1	1	2	1.96
20	1	1	3	2.08
21	2	1	71	2.24
22	2	1	71	4.01
23	2	1	72	1.86
24				

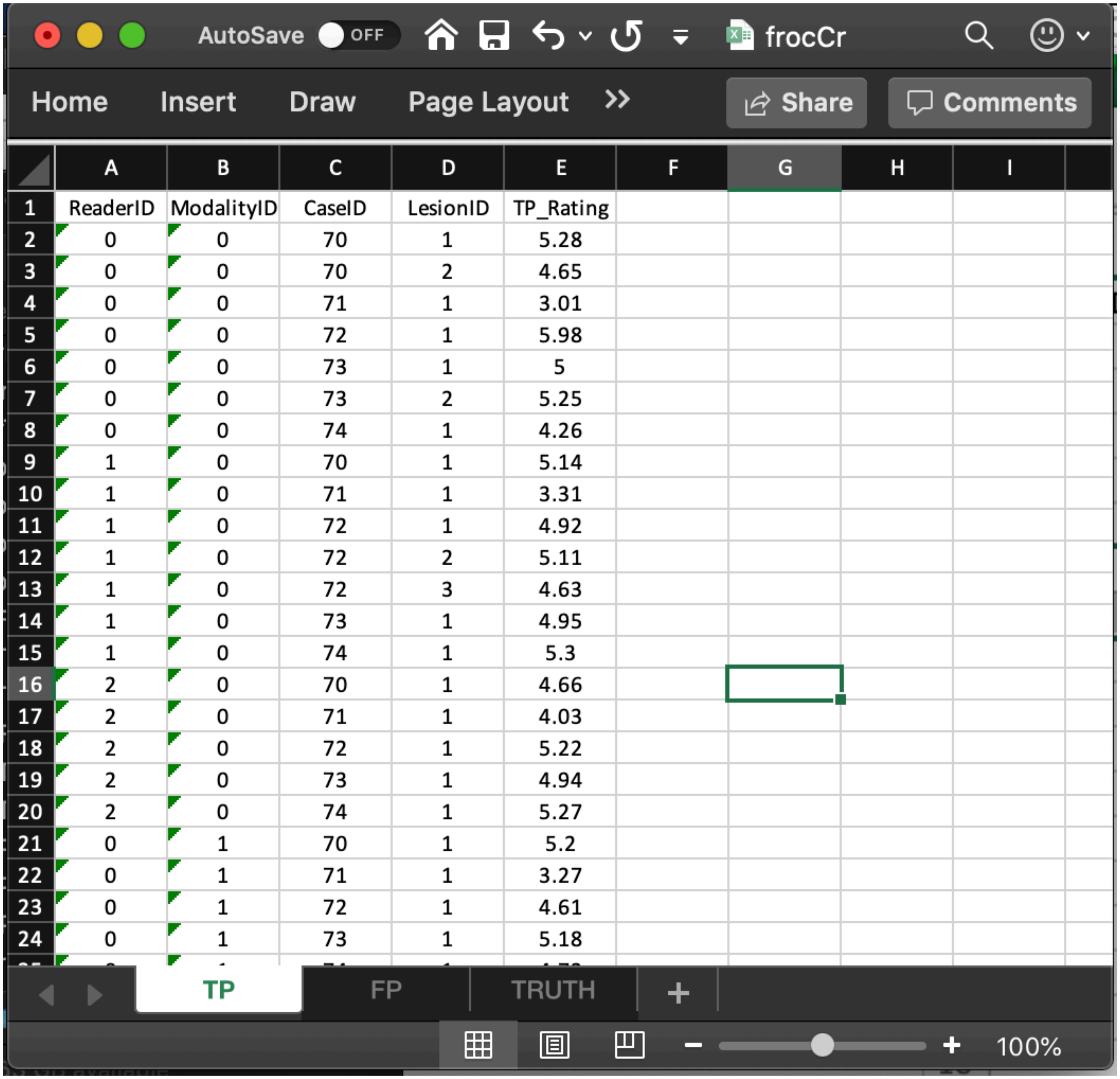
- It consists of 4 columns, of equal length. The common length is an integer random variable greater than or equal to zero. It could be zero if the dataset has no NL marks (a possibility if the lesions are very easy to find and the observer has perfect performance).

- In the example dataset, the common length is 22.
- `ReaderID`: the reader labels: these must be `0`, `1`, or `2`, as declared in the `Truth` worksheet.
- `ModalityID`: the modality labels: must be `0` or `1`, as declared in the `Truth` worksheet.
- `CaseID`: the labels of cases with `NL` marks. In the FROC paradigm `NL` events can occur on non-diseased **and** diseased cases.
- `FP_Rating`: the floating point ratings of `NL` marks. Each row of this worksheet yields a rating corresponding to the values of `ReaderID`, `ModalityID` and `CaseID` for that row.
- For `ModalityID` 0, `ReaderID` 0 and `CaseID` 1 (the first non-diseased case declared in the `Truth` worksheet), there is a single `NL` mark that was rated 1.02, corresponding to row 2 of the `FP` worksheet.
- Diseased cases with `NL` marks are also recorded in the `FP` worksheet. Some examples are seen at rows 15, 16 and 21, 22, 23.

- Rows 21 and 22 show that `caseID` = 71 got two `NL` marks, rated 2.24, 4.01.
- Since this is the *only* case with two NL marks, it determines the length of the fourth dimension of the `x$ratings$NL` list member, 2. Absent this case, the length would have been one.
- The case with the most `NL` marks determines the length of the fourth dimension of the `x$ratings$NL` list member.
- The reader should confirm that the ratings in `x$ratings$NL` reflect the contents of the `FP` worksheet.

3.6 The true positive (TP) ratings

These are found in the `TP` or `LL` worksheet, see below.



	A	B	C	D	E	F	G	H	I
1	ReaderID	ModalityID	CaseID	LesionID	TP_Rating				
2	0	0	70	1	5.28				
3	0	0	70	2	4.65				
4	0	0	71	1	3.01				
5	0	0	72	1	5.98				
6	0	0	73	1	5				
7	0	0	73	2	5.25				
8	0	0	74	1	4.26				
9	1	0	70	1	5.14				
10	1	0	71	1	3.31				
11	1	0	72	1	4.92				
12	1	0	72	2	5.11				
13	1	0	72	3	4.63				
14	1	0	73	1	4.95				
15	1	0	74	1	5.3				
16	2	0	70	1	4.66				
17	2	0	71	1	4.03				
18	2	0	72	1	5.22				
19	2	0	73	1	4.94				
20	2	0	74	1	5.27				
21	0	1	70	1	5.2				
22	0	1	71	1	3.27				
23	0	1	72	1	4.61				
24	0	1	73	1	5.18				

- This worksheet can only have diseased cases. The presence of a non-diseased case in this worksheet will generate an error.

- The common vertical length, 31 in this example, is a-priori unpredictable. The maximum possible length, assuming every lesion is marked for each modality, reader and diseased case, is 9 X 2 X 3 = 54. The 9 comes from the total number of non-zero entries in the `LesionID` column of the `Truth` worksheet, the 2 from the number of modalities and 3 from the number of readers.
- The fact that the actual length (31) is smaller than the maximum length (54) means that there are combinations of modality, reader and diseased cases on which some lesions were not marked.
- As examples, line 2 in the worksheet, the first lesion in `CaseID` equal to `70` was marked (and rated 5.28) in `ModalityID 0` and `ReaderID 0`. Line 3 in the worksheet, the second lesion in `CaseID` equal to `70` was also marked (and rated 4.65) in `ModalityID 0` and `ReaderID 0`. However, lesions 2 and 3 in `CaseID` = 72 were not marked (line 5 in the worksheet indicates that for this modality-reader-case combination only the first lesion was marked).
- The length of the fourth dimension of the `x$ratings$LL` list member, 3 in the present example, is determined by the diseased case (72) with the most lesions in the `Truth` worksheet.
- The reader should confirm that the ratings in `x$ratings$LL` reflect the contents of the `TP` worksheet.

3.7 On the distribution of numbers of lesions in diseased cases

- Consider a much larger dataset, `dataset11`, with structure as shown below (for descriptions of all embedded datasets the `RJafroc` documentation):

```
x <- dataset11
str(x)
#> List of 3
#>  $ ratings     :List of 3
#>   ..$ NL   : num [1:4, 1:5, 1:158, 1:4] -Inf -Inf -Inf -Inf -Inf ...
#>   ..$ LL   : num [1:4, 1:5, 1:115, 1:20] -Inf -Inf -Inf -Inf -Inf ...
#>   ..$ LL_IL: logi NA
#>  $ lesions     :List of 3
#>   ..$ perCase: int [1:115] 6 4 7 1 3 3 3 8 11 2 ...
#>   ..$ IDs    : num [1:115, 1:20] 1 1 1 1 1 1 1 1 1 1 ...
#>   ..$ weights: num [1:115, 1:20] 0.167 0.25 0.143 1 0.333 ...
#>  $ descriptions:List of 7
#>   ..$ fileName     : chr "dataset11"
#>   ..$ type         : chr "FROC"
#>   ..$ name         : chr "DOBBINS-1"
#>   ..$ truthTableStr: num [1:4, 1:5, 1:158, 1:21] 1 1 1 1 1 1 1 1 1 1 ...
#>   ..$ design       : chr "FCTRL"
#>   ..$ modalityID   : Named chr [1:4] "1" "2" "3" "4"
#>   .. ..- attr(*, "names")= chr [1:4] "1" "2" "3" "4"
#>   ..$ readerID     : Named chr [1:5] "1" "2" "3" "4" ...
#>   .. ..- attr(*, "names")= chr [1:5] "1" "2" "3" "4" ...
```

- Focus for now in the 115 diseased cases.
- The numbers of lesions in these cases is contained in `x$lesions$perCase`.

```
x$lesions$perCase
#>   [1]  6  4  7  1  3  3  3  8 11  2  4  6  2 16  5  2  8  3  4  7 11  1  4  3  4
#>  [26]  4  7  3  2  5  2  2  7  6  6  4 10 20 12  6  4  7 12  5  1  1  5  1  2  8
#>  [51]  3  1  2  2  3  2  8 16 10  1  2  2  6  3  2  2  4  6 10 11  1  2  6  2  4
#>  [76]  5  2  9  6  6  8  3  8  7  1  1  6  3  2  1  9  8  8  2  2 12  1  1  1  1
#> [101]  1  3  1  2  2  1  1  1  1  3  1  1  1  2  1
```

- For example, the first diseased case contains 6 lesions, the second contains 4 lesions, the third contains 7 lesions, etc. and the last diseased case contains 1 lesion.

- To get an idea of the distribution of the numbers of lesions per diseased cases, one could interrogate this vector as shown below using the `which()` function:

```
for (el in 1:max(x$lesions$perCase)) cat(
  "number of diseased cases with", el, "lesions = ",
  length(which(x$lesions$perCase == el)), "\n")
#> number of diseased cases with 1 lesions =  25
#> number of diseased cases with 2 lesions =  23
#> number of diseased cases with 3 lesions =  13
#> number of diseased cases with 4 lesions =  10
#> number of diseased cases with 5 lesions =  5
#> number of diseased cases with 6 lesions =  11
#> number of diseased cases with 7 lesions =  6
#> number of diseased cases with 8 lesions =  8
#> number of diseased cases with 9 lesions =  2
#> number of diseased cases with 10 lesions =  3
#> number of diseased cases with 11 lesions =  3
#> number of diseased cases with 12 lesions =  3
#> number of diseased cases with 13 lesions =  0
#> number of diseased cases with 14 lesions =  0
#> number of diseased cases with 15 lesions =  0
#> number of diseased cases with 16 lesions =  2
#> number of diseased cases with 17 lesions =  0
#> number of diseased cases with 18 lesions =  0
#> number of diseased cases with 19 lesions =  0
#> number of diseased cases with 20 lesions =  1
```

- This tells us that 25 cases contain 1 lesion
- Likewise, 23 cases contain 2 lesions
- Etc.

3.7.1 Definition of `lesDistr` array

- What is the fraction of (diseased) cases with 1 lesion, 2 lesions etc.

```
for (el in 1:max(x$lesions$perCase)) cat("fraction of diseased cases with", el, "lesions = ",
                                              length(which(x$lesions$perCase == el))/length(x$ratings$LL[1
#> fraction of diseased cases with 1 lesions =  0.2173913
#> fraction of diseased cases with 2 lesions =  0.2
#> fraction of diseased cases with 3 lesions =  0.1130435
#> fraction of diseased cases with 4 lesions =  0.08695652
#> fraction of diseased cases with 5 lesions =  0.04347826
#> fraction of diseased cases with 6 lesions =  0.09565217
#> fraction of diseased cases with 7 lesions =  0.05217391
#> fraction of diseased cases with 8 lesions =  0.06956522
#> fraction of diseased cases with 9 lesions =  0.0173913
#> fraction of diseased cases with 10 lesions =  0.02608696
#> fraction of diseased cases with 11 lesions =  0.02608696
#> fraction of diseased cases with 12 lesions =  0.02608696
#> fraction of diseased cases with 13 lesions =  0
#> fraction of diseased cases with 14 lesions =  0
#> fraction of diseased cases with 15 lesions =  0
#> fraction of diseased cases with 16 lesions =  0.0173913
#> fraction of diseased cases with 17 lesions =  0
#> fraction of diseased cases with 18 lesions =  0
#> fraction of diseased cases with 19 lesions =  0
#> fraction of diseased cases with 20 lesions =  0.008695652
```

- This tells us that fraction 0.217 of (diseased) cases contain 1 lesion
- And fraction 0.2 of (diseased) cases contain 2 lesions
- Etc.
- This information is obtained using the function `UtilLesionDistrVector()`

```
lesDistr <- UtilLesionDistrVector(x)
lesDistr
#>  [1] 0.217391304 0.200000000 0.113043478 0.086956522 0.043478261 0.095652174
#>  [7] 0.052173913 0.069565217 0.017391304 0.026086957 0.026086957 0.026086957
#> [13] 0.017391304 0.008695652
```

- TBA The `UtilLesionDistrVector()` function returns an array with two columns and number of rows equal to the number of *distinct non-zero* values of lesions per case.
- The first column contains the number of distinct non-zero values of lesions per case, 14 in the current example.
- The second column contains the fraction of diseased cases with the number of lesions indicated in the first column.
- The second column must sum to unity

```
sum(UtilLesionDistrVector(x))
#> [1] 1
```

- The lesion distribution array will come in handy when it comes to predicting the operating characteristics from using the Radiological Search Model (RSM), as detailed in TBA Chapter 17.

3.8 TBA Definition of `lesWghtDistr` array

- This is returned by `UtilLesionWeightsDistr()`.
- This contains the same number of rows as `lesDistr`.
- The number of columns is one plus the number of rows as `lesDistr`.
- The first column contains the number of distinct non-zero values of lesions per case, 14 in the current example.
- The second through the last columns contain the weights of cases with number of lesions per case corresponding to row 1.
- Missing values are filled with `-Inf`.

```
lesWghtDistr <- UtilLesionWeightsMatrixDataset(x, relWeights = 0)
cat("dim(lesDistr) =", dim(lesDistr),"\n")
#> dim(lesDistr) =
cat("dim(lesWghtDistr) =", dim(lesWghtDistr),"\n")
#> dim(lesWghtDistr) = 14 15
cat("lesWghtDistr = \n\n")
#> lesWghtDistr =
lesWghtDistr
#>       [,1]       [,2]       [,3]       [,4]       [,5]       [,6]       [,7]
#>  [1,]    1 1.00000000       -Inf       -Inf       -Inf       -Inf       -Inf
#>  [2,]    2 0.50000000 0.50000000       -Inf       -Inf       -Inf       -Inf
#>  [3,]    3 0.33333333 0.33333333 0.33333333       -Inf       -Inf       -Inf
#>  [4,]    4 0.25000000 0.25000000 0.25000000 0.25000000       -Inf       -Inf
#>  [5,]    5 0.20000000 0.20000000 0.20000000 0.20000000 0.20000000       -Inf
#>  [6,]    6 0.16666667 0.16666667 0.16666667 0.16666667 0.16666667 0.16666667
#>  [7,]    7 0.14285714 0.14285714 0.14285714 0.14285714 0.14285714 0.14285714
#>  [8,]    8 0.12500000 0.12500000 0.12500000 0.12500000 0.12500000 0.12500000
#>  [9,]    9 0.11111111 0.11111111 0.11111111 0.11111111 0.11111111 0.11111111
#> [10,]   10 0.10000000 0.10000000 0.10000000 0.10000000 0.10000000 0.10000000
#> [11,]   11 0.09090909 0.09090909 0.09090909 0.09090909 0.09090909 0.09090909
#> [12,]   12 0.08333333 0.08333333 0.08333333 0.08333333 0.08333333 0.08333333
#> [13,]   13 0.07692308 0.07692308 0.07692308 0.07692308 0.07692308 0.07692308
#> [14,]   14 0.07142857 0.07142857 0.07142857 0.07142857 0.07142857 0.07142857
#>               [,8]       [,9]      [,10]      [,11]      [,12]      [,13]
#>  [1,]       -Inf       -Inf       -Inf       -Inf       -Inf       -Inf
#>  [2,]       -Inf       -Inf       -Inf       -Inf       -Inf       -Inf
#>  [3,]       -Inf       -Inf       -Inf       -Inf       -Inf       -Inf
#>  [4,]       -Inf       -Inf       -Inf       -Inf       -Inf       -Inf
#>  [5,]       -Inf       -Inf       -Inf       -Inf       -Inf       -Inf
#>  [6,]       -Inf       -Inf       -Inf       -Inf       -Inf       -Inf
#>  [7,] 0.14285714       -Inf       -Inf       -Inf       -Inf       -Inf
#>  [8,] 0.12500000 0.12500000       -Inf       -Inf       -Inf       -Inf
#>  [9,] 0.11111111 0.11111111 0.11111111       -Inf       -Inf       -Inf
#> [10,] 0.10000000 0.10000000 0.10000000 0.10000000       -Inf       -Inf
#> [11,] 0.09090909 0.09090909 0.09090909 0.09090909 0.09090909       -Inf
#> [12,] 0.08333333 0.08333333 0.08333333 0.08333333 0.08333333 0.08333333
#> [13,] 0.07692308 0.07692308 0.07692308 0.07692308 0.07692308 0.07692308
#> [14,] 0.07142857 0.07142857 0.07142857 0.07142857 0.07142857 0.07142857
#>              [,14]      [,15]
#>  [1,]       -Inf       -Inf
#>  [2,]       -Inf       -Inf
#>  [3,]       -Inf       -Inf
#>  [4,]       -Inf       -Inf
#>  [5,]       -Inf       -Inf
#>  [6,]       -Inf       -Inf
#>  [7,]       -Inf       -Inf
#>  [8,]       -Inf       -Inf
#>  [9,]       -Inf       -Inf
#> [10,]       -Inf       -Inf
#> [11,]       -Inf       -Inf
#> [12,]       -Inf       -Inf
#> [13,] 0.07692308       -Inf
#> [14,] 0.07142857 0.07142857
```

- Row 3 corresponds to 3 lesions per case and the weights are 1/3, 1/3 and 1/3.
- Row 13 corresponds to 16 lesions per case and the weights are 0.06250000, 0.06250000, …, repeated 13 times.
- Note that the number of rows is less than the maximum number of lesions per case (20).
- This is because some configurations of lesions per case (e.g., cases with 13 lesions per case) do not occur in this dataset.

Chapter 4

DBM analysis text output

4.1 TBA How much finished

50%

4.2 Introduction

This chapter illustrates significance testing using the DBM method.

4.3 Analyzing the ROC dataset

This illustrates the `StSignificanceTesting()` function. The significance testing method is specified as `"DBM"` and the figure of merit `FOM` is specified as “Wilcoxon”. The embedded dataset `dataset03` is used.

```
ret <- StSignificanceTesting(dataset03, FOM = "Wilcoxon", method = "DBM")
```

4.4 Explanation of the output

The function returns a list with 5 members:

- `FOMs`: figures of merit.
- `ANOVA`: ANOVA tables.
- `RRRC`: random-reader random-case analyses results.
- `FRRC`: fixed-reader random-case analyses results.
- `RRFC`” random-reader fixed-case analyses results.

Let us consider them individually.

```
str(ret$FOMs)
#> List of 3
#>  $ foms        :'data.frame':    2 obs. of  4 variables:
#>   ..$ rdrREADER_1: num [1:2] 0.853 0.85
#>   ..$ rdrREADER_2: num [1:2] 0.865 0.844
#>   ..$ rdrREADER_3: num [1:2] 0.857 0.84
#>   ..$ rdrREADER_4: num [1:2] 0.815 0.814
#>  $ trtMeans    :'data.frame':    2 obs. of  1 variable:
#>   ..$ Estimate: num [1:2] 0.848 0.837
#>  $ trtMeanDiffs:'data.frame':    1 obs. of  1 variable:
#>   ..$ Estimate: num 0.0109
```

- FOMs is a list of 3
  - foms is a [2x4] dataframe: the figure of merit for each of of the four observers in the two treatments.
  - trtMeans is a [2x1] dataframe: the average figure of merit over all readers for each treatment.
  - trtMeanDiffs a [1x1] dataframe: the difference(s) of the reader-averaged figures of merit for all different-treatment pairings. In this example, with only two treatments, there is only one different-treatment pairing.

```
ret$FOMs$foms
#>          rdrREADER_1 rdrREADER_2 rdrREADER_3 rdrREADER_4
#> trtTREAT1  0.85345997  0.86499322  0.85730439  0.81524197
#> trtTREAT2  0.84961556  0.84350972  0.84011759  0.81433740
ret$FOMs$trtMeans
#>             Estimate
#> trtTREAT1 0.84774989
#> trtTREAT2 0.83689507
ret$FOMs$trtMeanDiffs
#>                        Estimate
#> trtTREAT1-trtTREAT2 0.010854817
```

```
str(ret$ANOVA)
#> List of 4
#>  $ TRCanova     :'data.frame':   8 obs. of  3 variables:
#>   ..$ SS: num [1:8] 0.0236 0.2052 52.5284 0.0151 6.41 ...
#>   ..$ DF: num [1:8] 1 3 99 3 99 297 297 799
#>   ..$ MS: num [1:8] 0.02357 0.06841 0.53059 0.00502 0.06475 ...
#>  $ VarCom       :'data.frame':   6 obs. of  1 variable:
#>   ..$ Estimates: num [1:6] 3.78e-05 5.13e-02 -7.13e-04 -2.89e-03 2.79e-02 ...
#>  $ IndividualTrt:'data.frame':   3 obs. of  3 variables:
#>   ..$ DF       : num [1:3] 3 99 297
#>   ..$ TrtTREAT1: num [1:3] 0.0493 0.294 0.105
#>   ..$ TrtTREAT2: num [1:3] 0.0242 0.3014 0.1034
#>  $ IndividualRdr:'data.frame':   3 obs. of  5 variables:
#>   ..$ DF         : num [1:3] 1 99 99
#>   ..$ rdrREADER_1: num [1:3] 0.000739 0.203875 0.091559
#>   ..$ rdrREADER_2: num [1:3] 0.0231 0.2234 0.0803
#>   ..$ rdrREADER_3: num [1:3] 0.0148 0.2142 0.0612
#>   ..$ rdrREADER_4: num [1:3] 4.09e-05 2.85e-01 6.06e-02
```

- ANOVA is a list of 4
  - TRCanova is a [8x3] dataframe: the treatment-reader-case ANOVA table, see below, where SS is the sum of squares, DF is the denominator degrees of freedom and MS is the mean squares, and T = treatment, R = reader, C = case, TR = treatment-reader, TC = treatment-case, RC = reader-case, TRC = treatment-reader-case.

  - VarCom is a [6x1] dataframe: the variance components, see below, where varR is the reader variance, varC is the case variance, varTR is the treatment-reader variance, varTC is the treatment-case variance, varRC is the reader-case variance, and varTRC is the treatment-reader-case variance.
  - IndividualTrt is a [3x3] dataframe: the individual treatment variance components averaged over all readers, see below, where msR is the mean square reader, msC is the mean square case and msRC is the mean square reader-case.

  - `IndividualRdr` is a [3x5] dataframe: the individual reader variance components averaged over treatments, see below, where `msT` is the mean square treatment, `msC` is the mean square case and `msTC` is the mean square treatment-case.

```
ret$ANOVA$TRCanova
#>                  SS  DF           MS
#> T       0.023565410   1 0.0235654097
#> R       0.205217999   3 0.0684059998
#> C      52.528398680  99 0.5305898857
#> TR      0.015060792   3 0.0050202641
#> TC      6.410048814  99 0.0647479678
#> RC     39.242953812 297 0.1321311576
#> TRC    22.660077641 297 0.0762965577
#> Total 121.085323149 799           NA
ret$ANOVA$VarCom
#>              Estimates
#> VarR    3.7755679e-05
#> VarC    5.1250915e-02
#> VarTR  -7.1276294e-04
#> VarTC  -2.8871475e-03
#> VarRC   2.7917300e-02
#> VarErr  7.6296558e-02
ret$ANOVA$IndividualTrt
#>       DF   TrtTREAT1   TrtTREAT2
#> msR    3 0.049266349 0.024159915
#> msC   99 0.293967531 0.301370323
#> msRC 297 0.105047872 0.103379843
ret$ANOVA$IndividualRdr
#>      DF   rdrREADER_1 rdrREADER_2 rdrREADER_3   rdrREADER_4
#> msT   1 0.00073897606 0.023077021 0.014769293 0.00004091217
#> msC  99 0.20387477465 0.223441908 0.214246773 0.28541990211
#> msTC 99 0.09155873437 0.080279256 0.061228980 0.06057067104
```

```
str(ret$RRRC)
#> List of 3
#>  $ FTests         :'data.frame': 2 obs. of  4 variables:
#>   ..$ DF   : num [1:2] 1 3
#>   ..$ MS   : num [1:2] 0.02357 0.00502
#>   ..$ FStat: num [1:2] 4.69 NA
#>   ..$ p    : num [1:2] 0.119 NA
#>  $ ciDiffTrt      :'data.frame': 1 obs. of  7 variables:
#>   ..$ Estimate: num 0.0109
#>   ..$ StdErr  : num 0.00501
#>   ..$ DF      : num 3
#>   ..$ t       : num 2.17
#>   ..$ PrGTt   : num 0.119
#>   ..$ CILower : num -0.00509
#>   ..$ CIUpper : num 0.0268
#>  $ ciAvgRdrEachTrt:'data.frame': 2 obs. of  5 variables:
#>   ..$ Estimate: num [1:2] 0.848 0.837
#>   ..$ StdErr  : num [1:2] 0.0244 0.0236
#>   ..$ DF      : num [1:2] 70.1 253.6
#>   ..$ CILower : num [1:2] 0.799 0.79
#>   ..$ CIUpper : num [1:2] 0.896 0.883
```

- RRRC, a list of 3 containing results of random-reader random-case analyses

  - `FTtests`: is a [2x4] dataframe: results of the F-tests, see below, where `FStat` is the F-statistic and `p` is the p-value. The first row is the treatment effect and the second is the error term.
  - `ciDiffTrt`: is a [1x7] dataframe: the confidence intervals between different-treatments, see below, where `StdErr` is the standard error of the estimate, `t` is the t-statistic and `PrGTt` is the p-value.
  - `ciAvgRdrEachTrt`: is a [2x5] dataframe: the confidence intervals for each treatment, averaged over all readers in the treatment, see below, where `CILower` is the lower 95% confidence interval and `CIUpper` is the upper 95% confidence interval.

```
ret$RRRC$FTests
#>           DF           MS     FStat          p
#> Treatment  1 0.0235654097 4.6940577 0.11883786
#> Error      3 0.0050202641        NA         NA
ret$RRRC$ciDiffTrt
#>                        Estimate       StdErr DF         t      PrGTt
#> trtTREAT1-trtTREAT2 0.010854817 0.0050101218  3 2.1665774 0.11883786
#>                           CILower     CIUpper
#> trtTREAT1-trtTREAT2 -0.0050896269 0.026799261
ret$RRRC$ciAvgRdrEachTrt
#>             Estimate      StdErr         DF    CILower    CIUpper
#> trtTREAT1 0.84774989 0.024402152  70.121788 0.79908282 0.89641696
#> trtTREAT2 0.83689507 0.023566416 253.644028 0.79048429 0.88330585
```

```
str(ret$FRRC)
#> List of 4
#>  $ FTests          :'data.frame':    2 obs. of  4 variables:
#>   ..$ DF   : num [1:2] 1 99
#>   ..$ MS   : num [1:2] 0.0236 0.0647
#>   ..$ FStat: num [1:2] 0.364 NA
#>   ..$ p    : num [1:2] 0.548 NA
#>  $ ciDiffTrt       :'data.frame':    1 obs. of  7 variables:
#>   ..$ Estimate: num 0.0109
#>   ..$ StdErr  : num 0.018
#>   ..$ DF      : num 99
#>   ..$ t       : num 0.603
#>   ..$ PrGTt   : num 0.548
#>   ..$ CILower : num -0.0248
#>   ..$ CIUpper : num 0.0466
#>  $ ciAvgRdrEachTrt :'data.frame':    2 obs. of  5 variables:
#>   ..$ Estimate: num [1:2] 0.848 0.837
#>   ..$ StdErr  : num [1:2] 0.0271 0.0274
#>   ..$ DF      : num [1:2] 99 99
#>   ..$ CILower : num [1:2] 0.794 0.782
#>   ..$ CIUpper : num [1:2] 0.902 0.891
#>  $ ciDiffTrtEachRdr:'data.frame':    4 obs. of  7 variables:
#>   ..$ Estimate: num [1:4] 0.003844 0.021483 0.017187 0.000905
#>   ..$ StdErr  : num [1:4] 0.0428 0.0401 0.035 0.0348
#>   ..$ DF      : num [1:4] 99 99 99 99
#>   ..$ t       : num [1:4] 0.0898 0.5362 0.4911 0.026
#>   ..$ PrGTt   : num [1:4] 0.929 0.593 0.624 0.979
#>   ..$ CILower : num [1:4] -0.0811 -0.058 -0.0522 -0.0682
#>   ..$ CIUpper : num [1:4] 0.0888 0.101 0.0866 0.07
```

- FRRC, a list of 4 containing results of fixed-reader random-case analyses
  - `FTtests`: is a [2x4] dataframe: results of the F-tests, see below.
  - `ciDiffTrt`: is a [1x7] dataframe: the confidence intervals between different-treatments, see below.

  - `ciAvgRdrEachTrt`: is a [2x5] dataframe: the confidence intervals for the average reader over each treatment
  - `ciDiffTrtEachRdr`: is a [4x7] dataframe: the confidence intervals for each different-treatment pairing for each reader.

```
ret$FRRC$FTests
#>           DF          MS      FStat          p
#> Treatment  1 0.023565410 0.36395597 0.54769704
#> Error     99 0.064747968         NA         NA
ret$FRRC$ciDiffTrt
#>                        Estimate      StdErr DF         t      PrGTt
#> trtTREAT1-trtTREAT2 0.010854817 0.017992772 99 0.60328764 0.54769704
#>                          CILower    CIUpper
#> trtTREAT1-trtTREAT2 -0.024846746 0.04655638
ret$FRRC$ciAvgRdrEachTrt
#>              Estimate      StdErr DF    CILower    CIUpper
#> trtTREAT1 0.84774989 0.027109386 99 0.79395898 0.90154079
#> trtTREAT2 0.83689507 0.027448603 99 0.78243109 0.89135905
ret$FRRC$ciDiffTrtEachRdr
#>                                       Estimate      StdErr DF           t
#> rdrREADER_1::trtTREAT1-trtTREAT2 0.00384441429 0.042792227 99 0.089839080
#> rdrREADER_2::trtTREAT1-trtTREAT2 0.02148349163 0.040069753 99 0.536152334
#> rdrREADER_3::trtTREAT1-trtTREAT2 0.01718679331 0.034993994 99 0.491135520
#> rdrREADER_4::trtTREAT1-trtTREAT2 0.00090456807 0.034805365 99 0.025989329
#>                                       PrGTt      CILower     CIUpper
#> rdrREADER_1::trtTREAT1-trtTREAT2 0.92859660 -0.081064648 0.088753476
#> rdrREADER_2::trtTREAT1-trtTREAT2 0.59305592 -0.058023592 0.100990575
#> rdrREADER_3::trtTREAT1-trtTREAT2 0.62441761 -0.052248882 0.086622469
#> rdrREADER_4::trtTREAT1-trtTREAT2 0.97931817 -0.068156827 0.069965963
```

```
str(ret$RRFC)
#> List of 3
#>  $ FTests         :'data.frame': 2 obs. of  4 variables:
#>   ..$ DF   : num [1:2] 1 3
#>   ..$ MS   : num [1:2] 0.02357 0.00502
#>   ..$ FStat: num [1:2] 4.69 NA
#>   ..$ p    : num [1:2] 0.119 NA
#>  $ ciDiffTrt      :'data.frame': 1 obs. of  7 variables:
#>   ..$ Estimate: num 0.0109
#>   ..$ StdErr  : num 0.00501
#>   ..$ DF      : num 3
#>   ..$ t       : num 2.17
#>   ..$ PrGTt   : num 0.119
#>   ..$ CILower : num -0.00509
#>   ..$ CIUpper : num 0.0268
#>  $ ciAvgRdrEachTrt:'data.frame': 2 obs. of  5 variables:
#>   ..$ Estimate: num [1:2] 0.848 0.837
#>   ..$ StdErr  : num [1:2] 0.0111 0.00777
#>   ..$ DF      : num [1:2] 3 3
#>   ..$ CILower : num [1:2] 0.812 0.812
#>   ..$ CIUpper : num [1:2] 0.883 0.862
```

- RRFC, a list of 3 containing results of random-reader fixed-case analyses
  - `FTtests`: is a [2x4] dataframe: results of the F-tests, see below.
  - `ciDiffTrt`: is a [1x7] dataframe: the confidence intervals between different-treatments, see below.

- `ciAvgRdrEachTrt`: is a [2x5] dataframe: the confidence intervals for the average reader over each over each treatment.

```
ret$RRFC$FTests
#>           DF           MS     FStat          p
#> Treatment  1 0.0235654097 4.6940577 0.11883786
#> Error      3 0.0050202641        NA         NA
ret$RRFC$ciDiffTrt
#>                         Estimate       StdErr DF         t      PrGTt
#> trtTREAT1-trtTREAT2 0.010854817 0.0050101218  3 2.1665774 0.11883786
#>                           CILower     CIUpper
#> trtTREAT1-trtTREAT2 -0.0050896269 0.026799261
ret$RRFC$ciAvgRdrEachTrt
#>             Estimate      StdErr DF    CILower    CIUpper
#> trtTREAT1 0.84774989 0.011098012  3 0.81243106 0.88306871
#> trtTREAT2 0.83689507 0.007771730  3 0.81216196 0.86162818
```

Chapter 5

OR analysis text output

5.1 TBA How much finished

90%

5.2 Introduction

This chapter illustrates significance testing using the DBM and OR methods.

5.3 Analyzing the ROC dataset

The only change is to specify `method = "OR"` in the significance testing function. The same dataset is used as was used in the previous chapter.

```r
ret <- StSignificanceTesting(dataset03, FOM = "Wilcoxon", method = "OR")
```

5.4 Explanation of the output

The function returns a list with 5 members.

- `FOMs`: figures of merit, identical to that in the DBM method.
- `ANOVA`: ANOVA tables.
- `RRRC`: random-reader random-case analyses results.
- `FRRC`: fixed-reader random-case analyses results.
- `RRFC`" random-reader fixed-case analyses results.

Let us consider the ones that are different from the DBM method.

- ANOVA is a list of 4
  - `TRanova` is a [3x3] dataframe: the treatment-reader ANOVA table, see below, where SS is the sum of squares, DF is the denominator degrees of freedom and MS is the mean squares, and T = treatment, R = reader, TR = treatment-reader.

  - `VarCom` is a [6x2] dataframe: the variance components, see below, where `varR` is the reader variance, `varTR` is the treatment-reader variance, `Cov1`, `Cov2`,`Cov3` and `Var` are as defined in the OR model. The second column lists the correlations defined in the OR model.
  - `IndividualTrt` is a [2x4] dataframe: the individual treatment mean-squares, variances and Cov_2, averaged over all readers, see below, where `msREachTrt` is the mean square reader, `varEachTrt` is the variance and `cov2EachTrt` is `Cov2EachTrt` in each treatment.
  - `IndividualRdr` is a [2x4] dataframe: the individual reader variance components averaged over treatments, see below, where `msTEachRdr` is the mean square treatment, `varEachRdr` is the variance and `cov1EachRdr` is Cov_1 for each reader.

```
ret$ANOVA$TRanova
#>             SS DF           MS
#> T  0.00023565410  1 2.3565410e-04
#> R  0.00205217999  3 6.8406000e-04
#> TR 0.00015060792  3 5.0202641e-05
ret$ANOVA$VarCom
#>            Estimates       Rhos
#> VarR   2.3319942e-05         NA
#> VarTR -6.8389146e-04         NA
#> Cov1   7.9168215e-04 0.51887172
#> Cov2   4.8363767e-04 0.31697811
#> Cov3   5.1250915e-04 0.33590059
#> Var    1.5257762e-03         NA
ret$ANOVA$IndividualTrt
#>           DF    msREachTrt   varEachTrt   cov2EachTrt
#> trtTREAT1  3 0.00049266349 0.0015227779 0.00047229915
#> trtTREAT2  3 0.00024159915 0.0015287746 0.00049497620
ret$ANOVA$IndividualRdr
#>             DF    msTEachRdr   varEachRdr   cov1EachRdr
#> rdrREADER_1  1 7.3897606e-06 0.0014771675 0.00056158020
#> rdrREADER_2  1 2.3077021e-04 0.0015186058 0.00071581326
#> rdrREADER_3  1 1.4769293e-04 0.0013773788 0.00076508897
#> rdrREADER_4  1 4.0912170e-07 0.0017299529 0.00112424616
```

- RRRC, a list of 3 containing results of random-reader random-case analyses
  - `FTtests`: is a [2x4] dataframe: results of the F-tests, see below, where `FStat` is the F-statistic and `p` is the p-value. The first row is the treatment effect and the second is the error term.
  - `ciDiffTrt`: is a [1x7] dataframe: the confidence intervals between different treatments, see below, where `StdErr` is the standard error of the estimate, `t` is the t-statistic and `PrGTt` is the p-value.
  - `ciAvgRdrEachTrt`: is a [2x5] dataframe: the confidence intervals for the average reader over each treatment, see below, where `CILower` is the lower 95% confidence interval and `CIUpper` is the upper 95% confidence interval.

```
ret$RRRC$FTests
#>           DF            MS     FStat          p
#> Treatment  1 2.3565410e-04 4.6940577 0.11883786
#> Error      3 5.0202641e-05        NA         NA
ret$RRRC$ciDiffTrt
#>                        Estimate       StdErr DF         t      PrGTt
#> trtTREAT1-trtTREAT2 0.010854817 0.0050101218  3 2.1665774 0.11883786
#>                           CILower     CIUpper
#> trtTREAT1-trtTREAT2 -0.0050896269 0.026799261
ret$RRRC$ciAvgRdrEachTrt
#>             Estimate      StdErr         DF    CILower    CIUpper          Cov2
#> trtTREAT1 0.84774989 0.024402152  70.121788 0.79908282 0.89641696 0.00047229915
#> trtTREAT2 0.83689507 0.023566416 253.644028 0.79048429 0.88330585 0.00049497620
```

- FRRC, a list of 5 containing results of fixed-reader random-case analyses
  - `FTtests`: is a [2x4] dataframe: results of the chisquare-tests, see below. Here is a difference from DBM: in the OR method for FRRC the denominator degrees of freedom of the F-statistic is infinite, and the test becomes equivalent to a chisquare test with the degrees of freedom equal to $I - 1$, where I is the number of treatments.
  - `ciDiffTrt`: is a [1x6] dataframe: the confidence intervals between different treatments, see below. An additional column lists
  - `ciAvgRdrEachTrt`: is a [2x5] dataframe: the confidence intervals for the average reader over each treatment
  - `ciDiffTrtEachRdr`: is a [4x6] dataframe: the confidence intervals for each different-treatment pairing for each reader.
  - `IndividualRdrVarCov1`: is a [4x2] dataframe: Var and Cov_1 for individual readers.

```
ret$FRRC$FTests
#>                   MS      Chisq DF          p
#> Treatment 0.0002356541 0.32101347  1 0.57099922
#> Error     0.0007340941         NA NA         NA
ret$FRRC$ciDiffTrt
#>                         Estimate      StdErr          z      PrGTz      CILower
#> trtTREAT1-trtTREAT2 0.010854817 0.019158472 0.56658051 0.57099922 -0.026695098
#>                          CIUpper
#> trtTREAT1-trtTREAT2 0.048404732
ret$FRRC$ciAvgRdrEachTrt
#>              Estimate      StdErr DF    CILower    CIUpper
#> trtTREAT1 0.84774989 0.027109386 99 0.79461647 0.90088331
#> trtTREAT2 0.83689507 0.027448603 99 0.78309680 0.89069334
ret$FRRC$ciDiffTrtEachRdr
#>                                       Estimate      StdErr           z
#> rdrREADER_1::trtTREAT1-trtTREAT2 0.00384441429 0.042792227 0.089839080
#> rdrREADER_2::trtTREAT1-trtTREAT2 0.02148349163 0.040069753 0.536152334
#> rdrREADER_3::trtTREAT1-trtTREAT2 0.01718679331 0.034993994 0.491135520
#> rdrREADER_4::trtTREAT1-trtTREAT2 0.00090456807 0.034805365 0.025989329
#>                                       PrGTz      CILower     CIUpper
#> rdrREADER_1::trtTREAT1-trtTREAT2 0.92841509 -0.080026809 0.087715638
#> rdrREADER_2::trtTREAT1-trtTREAT2 0.59185327 -0.057051781 0.100018765
#> rdrREADER_3::trtTREAT1-trtTREAT2 0.62333060 -0.051400174 0.085773761
#> rdrREADER_4::trtTREAT1-trtTREAT2 0.97926585 -0.067312693 0.069121830
ret$FRRC$IndividualRdrVarCov1
#>               varEachRdr   cov1EachRdr
#> rdrREADER_1 0.0014771675 0.00056158020
#> rdrREADER_2 0.0015186058 0.00071581326
#> rdrREADER_3 0.0013773788 0.00076508897
#> rdrREADER_4 0.0017299529 0.00112424616
```

- RRFC, a list of 3 containing results of random-reader fixed-case analyses
  - `FTtests`: is a [2x4] dataframe: results of the F-tests, see below.
  - `ciDiffTrt`: is a [1x7] dataframe: the confidence intervals between different treatments, see below.
  - `ciAvgRdrEachTrt`: is a [2x5] dataframe: the confidence intervals for the average reader over each over each treatment.

```
ret$RRFC$FTests
#>    DF           MS         F          p
#> T   1 2.3565410e-04 4.6940577 0.11883786
#> TR  3 5.0202641e-05        NA         NA
ret$RRFC$ciDiffTrt
```

```
#>                        Estimate       StdErr DF         t      PrGTt
#> trtTREAT1-trtTREAT2 0.010854817 0.0050101218  3 2.1665774 0.11883786
#>                           CILower     CIUpper
#> trtTREAT1-trtTREAT2 -0.0050896269 0.026799261
ret$RRFC$ciAvgRdrEachTrt
#>             Estimate      StdErr DF    CILower    CIUpper
#> TrtTREAT1 0.84774989 0.011098012  3 0.81243106 0.88306871
#> TrtTREAT2 0.83689507 0.007771730  3 0.81216196 0.86162818
```

Chapter 6

OR analysis Excel output

6.1 TBA How much finished

90%

6.2 Introduction

This chapter illustrates significance testing using the OR method. But, instead of the perhaps unwieldy output in Chapter 5, it generates an Excel output file containing the following worksheets:

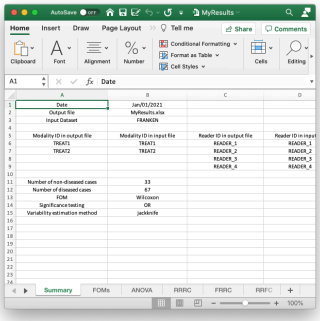
- `Summary`
- `FOMs`
- `ANOVA`
- `RRRC`
- `FRRC`
- `RRFC`

6.3 Generating the Excel output file

This illustrates the `UtilOutputReport()` function. The arguments are the embedded dataset, `dataset03`, the same dataset as in the previous two chapters, the report file base name `ReportFileBaseName` is set to `R/quick-start/MyResults`, the report file extension `ReportFileExt` is set to `xlsx`, the `FOM` is set to “Wilcoxon”, the `method` of analysis is set to “OR”, and the flag `overWrite = TRUE` overwrites any existing file with the same name, as otherwise the program will pause for user input.

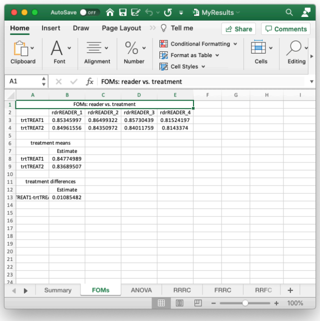
```
ret <- UtilOutputReport(get("dataset03"),
                        ReportFileBaseName = "R/quick-start/MyResults",
                        ReportFileExt = "xlsx",
                        FOM = "Wilcoxon",
                        method = "OR",
                        overWrite = TRUE)
```

The following screen shots display the contents of the created file `"R/quick-start/MyResults.xlsx"`.



A1 fx Date

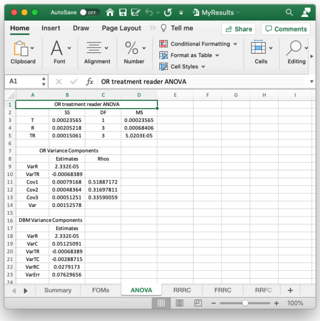
	A	B	C	D
1	Date	Jan/01/2021		
2	Output file	MyResults.xlsx		
3	Input Dataset	FRANKEN		
4				
5	Modality ID in output file	Modality ID in input file	Reader ID in output file	Reader ID in input
6	TREAT1	TREAT1	READER_1	READER_1
7	TREAT2	TREAT2	READER_2	READER_2
8			READER_3	READER_3
9			READER_4	READER_4
10				
11	Number of non-diseased cases	33		
12	Number of diseased cases	67		
13	FOM	Wilcoxon		
14	Significance testing	OR		
15	Variability estimation method	jackknife		



A1 fx FOMs: reader vs. treatment

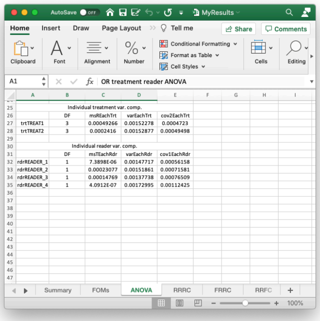
	A	B	C	D	E
1	FOMs: reader vs. treatment				
2		rdrREADER_1	rdrREADER_2	rdrREADER_3	rdrREADER_4
3	trtTREAT1	0.85345997	0.86499322	0.85730439	0.81524197
4	trtTREAT2	0.84961556	0.84350972	0.84011759	0.8143374
5					
6	treatment means				
7		Estimate			
8	trtTREAT1	0.84774989			
9	trtTREAT2	0.83689507			
10					
11	treatment differences				
12		Estimate			
13	TREAT1-trtTRE	0.01085482			

Figure 6.1: ‘Summary‘ and ‘FOMs‘ worksheets of Excel file ‘R/quick-start/MyResults.xlsx‘



A1 fx OR treatment reader ANOVA

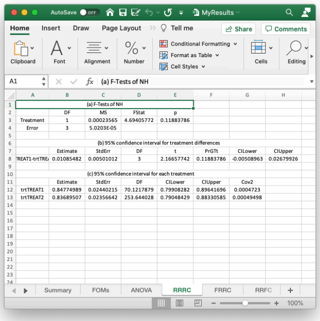
	A	B	C	D
1	OR treatment reader ANOVA			
2		SS	DF	MS
3	T	0.00023565	1	0.00023565
4	R	0.00205218	3	0.00068406
5	TR	0.00015061	3	5.0203E-05
6				
7	OR Variance Components			
8		Estimates	Rhos	
9	VarR	2.332E-05		
10	VarTR	-0.00068389		
11	Cov1	0.00079168	0.51887172	
12	Cov2	0.00048364	0.31697811	
13	Cov3	0.00051251	0.33590059	
14	Var	0.00152578		
15				
16	DBM Variance Components			
17		Estimates		
18	VarR	2.332E-05		
19	VarC	0.05125091		
20	VarTR	-0.00068389		
21	VarTC	-0.00288715		
22	VarRC	0.0279173		
23	VarErr	0.07629656		



A1 fx OR treatment reader ANOVA

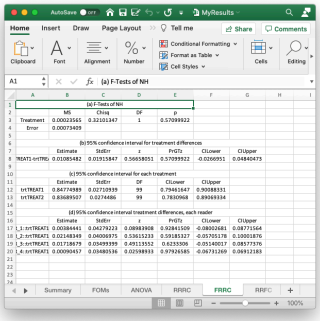
	A	B	C	D	E
25	Individual treatment var. comp.				
26		DF	msREachTrt	varEachTrt	cov2EachTrt
27	trtTREAT1	3	0.00049266	0.00152278	0.0004723
28	trtTREAT2	3	0.0002416	0.00152877	0.00049498
29					
30	Individual reader var. comp.				
31		DF	msTEachRdr	varEachRdr	cov1EachRdr
32	rdrREADER_1	1	7.3898E-06	0.00147717	0.00056158
33	rdrREADER_2	1	0.00023077	0.00151861	0.00071581
34	rdrREADER_3	1	0.00014769	0.00137738	0.00076509
35	rdrREADER_4	1	4.0912E-07	0.00172995	0.00112425

Figure 6.2: ‘ANOVA‘ worksheet of Excel file ‘R/quick-start/MyResults.xlsx‘



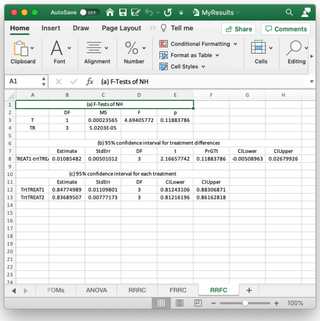
A1 fx (a) F-Tests of NH

	A	B	C	D	E	F	G	H
1	(a) F-Tests of NH							
2		DF	MS	FStat	p			
3	Treatment	1	0.00023565	4.69405772	0.11883786			
4	Error	3	5.0203E-05					
5								
6	(b) 95% confidence interval for treatment differences							
7		Estimate	StdErr	DF	t	PrGTt	CILower	CIUpper
8	TREAT1-trtTRE	0.01085482	0.00501012	3	2.16657742	0.11883786	-0.00508963	0.02679926
9								
10	(c) 95% confidence interval for each treatment							
11		Estimate	StdErr	DF	CILower	CIUpper	Cov2	
12	trtTREAT1	0.84774989	0.02440215	70.1217879	0.79908282	0.89641696	0.0004723	
13	trtTREAT2	0.83689507	0.02356642	253.644028	0.79048429	0.88330585	0.00049498	



A1 fx (a) F-Tests of NH

	A	B	C	D	E	F	G
1	(a) F-Tests of NH						
2		MS	Chisq	DF	p		
3	Treatment	0.00023565	0.32101347	1	0.57099922		
4	Error	0.00073409					
5							
6	(b) 95% confidence interval for treatment differences						
7		Estimate	StdErr	z	PrGTz	CILower	CIUpper
8	TREAT1-trtTRE	0.01085482	0.01915847	0.56658051	0.57099922	-0.0266951	0.04840473
9							
10	(c) 95% confidence interval for each treatment						
11		Estimate	StdErr	DF	CILower	CIUpper	
12	trtTREAT1	0.84774989	0.02710939	99	0.79461647	0.90088331	
13	trtTREAT2	0.83689507	0.0274486	99	0.7830968	0.89069334	
14							
15	(d) 95% confidence interval treatment differences, each reader						
16		Estimate	StdErr	z	PrGTz	CILower	CIUpper
17	_1::trtTREAT1	0.00384441	0.04279223	0.08983908	0.92841509	-0.08002681	0.08771564
18	_2::trtTREAT1	0.02148349	0.04006975	0.53615233	0.59185327	-0.05705178	0.10001876
19	_3::trtTREAT1	0.01718679	0.03499399	0.49113552	0.6233306	-0.05140017	0.08577376
20	_4::trtTREAT1	0.00090457	0.03480536	0.02598933	0.97926585	-0.06731269	0.06912183



A1 fx (a) F-Tests of NH

	A	B	C	D	E	F	G	H
1	(a) F-Tests of NH							
2		DF	MS	F	p			
3	T	1	0.00023565	4.69405772	0.11883786			
4	TR	3	5.0203E-05					
5								
6	(b) 95% confidence interval for treatment differences							
7		Estimate	StdErr	DF	t	PrGTt	CILower	CIUpper
8	TREAT1-trtTRE	0.01085482	0.00501012	3	2.16657742	0.11883786	-0.00508963	0.02679926
9								
10	(c) 95% confidence interval for each treatment							
11		Estimate	StdErr	DF	CILower	CIUpper		
12	TrtTREAT1	0.84774989	0.01109801	3	0.81243106	0.88306871		
13	TrtTREAT2	0.83689507	0.00777173	3	0.81216196	0.86162818		

Figure 6.3: ‘RRRC‘, ‘FRRC‘ and ‘RRFC‘ worksheets of Excel file ‘R/quick-start/MyResults.xlsx‘

ROC sample size

Chapter 7

ROC-DBM sample size

7.1 TBA How much finished

80%

7.2 Introduction

The question addressed here is "how many readers and cases", usually abbreviated to *sample-size*, should one employ to conduct a "well-planned" ROC study. The reasons for the quotes will shortly become clear. If cost were no concern, the reply would be: "as many readers and cases as one can get". While there are other considerations affecting sample-size, e.g., the data collection paradigm and the the analysis method, this chapter is restricted to the MRMC ROC data collection paradigm with data analyzed by the DBM method described in a previous chapter.

It turns out that provided one can specify conceptually valid effect-sizes between different paradigms (i.e., in the same "units"), the methods described in this chapter are extensible to other paradigms; see TBA Chapter 19 for sample size estimation for FROC studies. *For this reason it is important to understand the concepts of sample-size estimation in the simpler ROC context.*

For simplicity and practicality, this chapter, and the next, is restricted to analysis of two-treatment data ($I = 2$). The purpose of most imaging system assessment studies is to determine, for a given diagnostic task, whether radiologists perform better using a new treatment over the conventional treatment, and whether the difference is statistically significant. Therefore, the two-treatment case is the most common one encountered. While it is possible to extend the methods to more than two treatments, the extensions are not, in my opinion, clinically interesting.

Assume the figure of merit (FOM) θ is chosen to be the area AUC under the ROC curve (empirical or fitted is immaterial as far as the formulae are concerned; however, the choice will affect statistical power). The statistical analysis determines the significance level of the study, i.e., the probability or p-value for incorrectly rejecting the null hypothesis (NH) that the two θs are equal: $NH : \theta_1 = \theta_2$, where the subscripts refer to the two treatments and the bullet represents the average over the reader index. If the p-value is smaller than a pre-specified α, typically set at 5%, one rejects the NH and declares the treatments different at the α significance level. Statistical power is the probability of correctly rejecting the null hypothesis when the alternative hypothesis $AH : \theta_1 \neq \theta_2$ is true, (TBA Chapter 08).

The value of the *true* difference between the treatments, known as the *true effect-size* is, of course, unknown. If it were known, there would be no need to conduct the ROC study. One would simply adopt the treatment with the higher θ. Sample-size estimation involves making an educated guess regarding the true effect-size , called the *anticipated effect size*, and denoted by d. To quote Harold Kundel (ICRU, 1996): "any calculation of power amounts to specification of the anticipated effect-size". Increasing the anticipated effect size will increase statistical power but may represent an unrealistic expectation of the true difference between the treatments, in the sense that it overestimates the ability of technology to achieve this much improvement. Conversely, an unduly small d might be clinically insignificant, besides requiring a very large sample-size to achieve sufficient statistical power.

Statistical power depends on the magnitude of d divided by the standard deviation $\sigma(d)$ of d, i.e. $D = \frac{|d|}{\sigma(d)}$. The sign is relevant as it determines whether the project is worth pursuing at all (see TBA §11.8.4). The ratio is termed (Cohen, 1988) Cohen's D. When this signal-to-noise-ratio-like quantity is large, statistical power approaches 100%. Reader and case variability and data correlations determine $\sigma(d)$. No matter how small the anticipated d, as long as it is finite, then, using sufficiently large numbers of readers and cases $\sigma(d)$ can be made sufficiently small to achieve near 100% statistical power. Of course, a very small effect-size may not be clinically significant. There is a key difference between *statistical significance* and *clinical significance*. An effect-size in AUC units could be so small, e.g., 0.001, as to be clinically insignificant, but by employing a sufficiently large sample size one could design a study to detect this small - and clinically meaningless - difference with near unit probability, i.e., high statistical power.

What determines clinical significance? A small effect-size, e.g., 0.01 AUC units, could be clinically significant if it applies to a large population, where the small benefit in detection rate is amplified by the number of patients benefiting from the new treatment. In contrast, for an "orphan" disease, i.e., one with very low prevalence, an effect-size of 0.05 might not be enough to justify the additional cost of the new treatment. The improvement might have to be 0.1 before it is worth it for a new treatment to be brought to market. One hates to monetize life and death issues, but there is no getting away from it, as cost/benefit issues determine clinical significance. The arbiters of clinical significance are engineers, imaging scientists, clinicians, epidemiologists, insurance companies and those who set government health care policies. The engineers and imaging scientists determine whether the effect-size the clinicians would like is feasible from technical and scientific viewpoints. The clinician determines, based on incidence of disease and other considerations, e.g., altruistic, malpractice, cost of the new device and insurance reimbursement, what effect-size is justifiable. Cohen has suggested that d values of 0.2, 0.5, and 0.8 be considered small, medium, and large, respectively, but he has also argued against their indiscriminate usage. However, after a study is completed, clinicians often find that an effect-size that biostatisticians label as small may, in certain circumstances, be clinically significant and an effect-size that they label as large may in other circumstances be clinically insignificant. Clearly, this is a complex issue. Some suggestions on choosing a clinically significant effect size are made in (TBA §11.12).

Having developed a new imaging modality the R&D team wishes to compare it to the existing standard with the short-term goal of making a submission to the FDA to allow them to perform pre-market testing of the device. The long-term goal is to commercialize the device. Assume the R&D team has optimized the device based on physical measurements, (TBA Chapter 01), perhaps supplemented with anecdotal feedback from clinicians based on a few images. Needed at this point is a pilot study. A pilot study, conducted with a relatively small and practical sample size, is intended to provide estimates of different sources of variability and correlations. It also provides an initial estimate of the effect-size, termed the *observed effect-size*, d. Based on results from the pilot the sample-size tools described in this chapter permit estimation of the numbers of readers and cases that will reduce $\sigma(d)$ sufficiently to achieve the desired power for the larger "pivotal" study. [A distinction could be made in the notation between observed and anticipated effect sizes, but it will be clear from the context. Later, it will be shown how one can make an educated guess about the anticipated effect size from an observed effect size.]

This chapter is concerned with multiple-reader MRMC studies that follow the fully crossed factorial design meaning that each reader interprets a common case-set in all treatments. Since the resulting pairings (i.e., correlations) tend to decrease $\sigma(d)$ (since the variations occur in tandem, they tend to cancel out in the difference, see (TBA Chapter 09, Introduction), for Dr. Robert Wagner's sailboat analogy) it yields more statistical power compared to an unpaired design, and consequently this design is frequently used. Two sample-size estimation procedures for MRMC are the Hillis-Berbaum method (Hillis and Berbaum, 2004) and the Obuchowski-Rockette (Obuchowski, 1998) method. With recent work by Hillis, the two methods have been shown to be substantially equivalent.

This chapter will focus on the DBM approach. Since it is based on a standard ANOVA model, it is easier to extend the NH testing procedure described in Chapter 09 to the alternative hypothesis, which is relevant for sample size estimation. [TBA Online Appendix 11.A shows how to translate the DBM formulae to the OR method (Hillis et al., 2011).]

Given an effect-size, and choosing this wisely is the most difficult part of the process, the method described in this chapter uses pseudovalue variance components estimated by the DBM method to predict sample-sizes (i.e., different combinations of numbers of readers and cases) necessary to achieve a desired power.

7.3 Statistical Power

The concept of statistical power was introduced in [TBA Chapter 08] but is worth repeating. There are two possible decisions following a test of a null hypothesis (NH): reject or fail to reject the NH. Each decision is associated with a probability on an erroneous conclusion. If the NH is true and one rejects it, the probability of the ensuing Type-I error is denoted α. If the NH is false and one fails to reject it, the probability of the ensuing Type II- error is denoted β. Statistical power is the complement of β, i.e.,

$$Power = 1 - \beta \tag{7.1}$$

Typically, one aims for $\beta = 0.2$ or less, i.e., a statistical power of 80% or more. Like $\alpha = 0.05$, this is a *convention* and more nuanced cost-benefit considerations may cause the researcher to adopt a different value.

7.3.1 Observed vs. anticipated effect-size

Assuming no other similar studies have already been conducted with the treatments in question, the observed effect-size, although "merely an estimate", is the best information available at the end of the pilot study regarding the value of the true effect-size. From the two previous chapters one knows that the significance testing software will report not only the observed effect-size, but also a 95% confidence interval associate with it. It will be shown later how one can use this information to make an educated guess regarding the value of the anticipated effect-size.

7.3.2 Dependence of statistical power on estimates of model parameters

Examination of the expression for , Eqn. (11.5), shows that statistical power increases if:

- The numerator is large. This occurs if: (a) the anticipated effect-size d is large. Since effect-size enters as the *square*, TBA Eqn. (11.8), it is has a particularly strong effect; (b) If $J \times K$ is large. Both of these results should be obvious, as a large effect size and a large sample size should result in increased probability of rejecting the NH.
- The denominator is small. The first term in the denominator is $(\sigma_{\epsilon}^2 + \sigma_{\tau RC}^2)$. These two terms cannot be separated. This is the residual variability of the jackknife pseudovalues. It should make sense that the smaller the variability, the larger is the non-centrality parameter and the statistical power.
- The next term in the denominator is $K\sigma_{\tau R}^2$, the treatment-reader variance component multiplied by the total number of cases. The reader variance σ_R^2 has no effect on statistical power, because it has an equal effect on both treatments and cancels out in the difference. Instead, it is the treatment-reader variance σ_R^2 that contributes "noise" tending to confound the estimate of the effect-size.
- The variance components estimated by the ANOVA procedure are realizations of random variables and as such subject to noise (there actually exists a beast such as variance of a variance). The presence of the K term, usually large, can amplify the effect of noise in the estimate of σ_R^2, making the sample size estimation procedure less accurate.
- The final term in the denominator is $J\sigma_{\tau C}^2$. The variance σ_C^2 has no impact on statistical power, as it cancels out in the difference. The treatment-case variance component introduces "noise" into the estimate of the effect size, thereby decreasing power. Since it is multiplied by J, the number of readers, and typically $J << K$, the error amplification effect on accuracy of the sample size estimate is not as bad as with the treatment-reader variance component.
- Accuracy of sample size estimation, essentially estimating confidence intervals for statistical power, is addressed in (Chakraborty, 2010).

7.3.3 Formulae for random-reader random-case (RRRC) sample size estimation

7.3.4 Significance testing

7.3.5 p-value and confidence interval

7.3.6 Comparing DBM to Obuchowski and Rockette for single-reader multiple-treatments

Having performed a pilot study and planning to perform a pivotal study, sample size estimation follows the following procedure, which assumes that both reader and case are treated as random factors. Different formulae, described later, apply when either reader or case is treated as a fixed factor.

- Perform DBM analysis on the pilot data. This yields the observed effect size as well as estimates of all relevant variance components and mean squares appearing in TBA Eqn. (11.5) and Eqn. (11.7).
- This is the difficult but critical part: make an educated guess regarding the effect-size, d, that one is interested in "detecting" (i.e., hoping to reject the NH with probability $1-\beta$). The author prefers the term "anticipated" effect-size to "true" effect-size (the latter implies knowledge of the true difference between the modalities which, as noted earlier, would obviate the need for a pivotal study).
- Two scenarios are considered below. In the first scenario, the effect-size is assumed equal to that observed in the pilot study, i.e., $d = d_{obs}$.
- In the second, so-called "best-case" scenario, one assumes that the anticipate value of d is the observed value plus two-sigma of the confidence interval, in the correct direction, of course, i.e., $d = |d_{obs}| + 2\sigma$. Here σ is one-fourth the width of the 95% confidence interval for d_{obs}. Anticipating more than 2σ greater than the observed effect-size would be overly optimistic. The width of the CI implies that chances are less than 2.5% that the anticipated value is at or beyond the overly optimistic value. These points will become clearer when example datasets are analyzed below.
- Calculate statistical power using the distribution implied by Eqn. (11.4), to calculate the probability that a random value of the relevant F-statistic will exceed the critical value, as in §11.3.2.
- If power is below the desired or "target" power, one tries successively larger value of J and / or K until the target power is reached.

7.4 Formulae for fixed-reader random-case (FRRC) sample size estimation

It was shown in TBA §9.8.2 that for fixed-reader analysis the non-centrality parameter is defined by:

$$\Delta = \frac{JK\sigma_{\tau}^2}{\sigma_{\epsilon}^2 + \sigma_{\tau RC}^2 + J\sigma_{\tau C}^2} \tag{7.2}$$

The sampling distribution of the F-statistic under the AH is:

$$F_{AH|R} \equiv \frac{MST}{MSTC} \sim F_{I-1,(I-1)(K-1),\Delta} \tag{7.3}$$

7.4.1 Formulae for random-reader fixed-case (RRFC) sample size estimation

It is shown in TBA §9.9 that for fixed-case analysis the non-centrality parameter is defined by:

$$\Delta = \frac{JK\sigma_{\tau}^2}{\sigma_{\epsilon}^2 + \sigma_{\tau RC}^2 + K\sigma_{\tau R}^2} \tag{7.4}$$

Under the AH, the test statistic is distributed as a non-central F-distribution as follows:

$$F_{AH|C} \equiv \frac{MST}{MSTR} \sim F_{I-1,(I-1)(J-1),\Delta} \tag{7.5}$$

7.4.2 Fixed-reader random-case (FRRC) analysis TBA

It is a realization of a random variable, so one has some leeway in the choice of anticipated effect size - more on this later. Here J^* and K^* refer to the number of readers and cases in the *pilot* study.

7.4.3 Random-reader fixed-case (RRFC) analysis

7.4.4 Single-treatment multiple-reader analysis

7.5 Discussion/Summary/2

FROC analysis

Chapter 8

Analyzing FROC data

8.1 TBA How much finished

10%

8.2 Introduction

Analyzing FROC data is, apart from a single difference, very similar to analyzing ROC data. *The crucial difference is the selection of an appropriate location-sensitive figure of merit.* The reason is that the DBMH and ORH methods are applicable to any scalar figure of merit. Any appropriate FROC figure of merit reduces the mark rating data for a single dataset (i.e., a single treatment, a single reader and a number of cases) to a single scalar figure of merit.

The author recommends usage of the weighted AFROC figure of merit, where the lesions should be equally weighted, the default, unless there are strong clinical reasons for assigning unequal weights.

The chapter starts with analysis of a sample FROC dataset, #4 in Online Chapter 24. Any analysis should start with visualization of the relevant operating characteristic. Extensive examples are given using `RJafroc` implemented functions. Suggestions are made on how to report the results of a study (the suggestions apply equally to ROC studies). A method called *crossed-treatment analysis*, applicable when one has two treatment factors and their levels are crossed and one wishes to draw conclusions regarding the effect of treatments after averaging over all levels of the treatments.

8.3 Example 1

The following is a listing of file “mainAnalyzewAFROC.R”. It performs both wAFROC and inferred ROC analyses of the same dataset and the results are saved to tables similar in structure to the Excel output tables shown for DBMH analysis of ROC data in §9.10.2.

Empirical wAFROC-AUC and ROC-AUC for all combinations of treatments and readers, and reader-averaged AUCs for each treatment (Rdr. Avg.). The weighted AFROC results were obtained from worksheet FOMs in file FedwAfroc.xlsx. The highest rating AUC results were obtained from worksheet FOMs in file FedHrAuc.xlsx. The wAFROC-AUCs are smaller than the corresponding ROC-AUCs.

The datasets that come with this book are described in Online Chapter 24. Four of these are ROC datasets, one an LROC dataset and the rest (nine) are FROC datasets. For non-ROC datasets, the highest rating method was used to infer the corresponding ROC data. The datasets are identified in the code by strings contained in the string-array variable fileNames (line 7 - 8). Line 9 selects the dataset to be analyzed. In the example shown the “FED” dataset has been selected. It is a 5 treatment 4 radiologist FROC dataset1 acquired by Dr. Federica Zanca. Line 13 loads the dataset; this is done internal to the function loadDataFile(). Line 11 constructs the name of the wAFROC file and line 12 does the same for the ROC datafile. Line 15 which “spills over” to line 16 without the need for a special continuation character, generates an output file by performing DBMH significance testing (method = “DBMH”) using fom = “wAFROC”, i.e., the wAFROC figure of merit − this is the critical change. If one changes this to fom = “HrAuc”, lines 19 − 20, then inferred ROC analysis occurs. In either case the default analysis, i.e., option = “ALL” is used, i.e., random-reader random-case (RRRC), fixed-reader random-case (FRRC) and random-reader fixed-case (RRFC). Results are shown below for random-reader random-case only.

The results of wAFROC analysis are saved to FedwAfroc.xlsx and that of inferred ROC analysis are saved to FedHrAuc.xlsx. The output file names need to be explicitly stated as otherwise they would overwrite each other (as a time-saver, checks are made at lines 14 and 18 to determine if the analysis has already been performed, in which case it is skipped).

In the Excel data file the readers are named 1, 3, 4 and 5 − the software treats the reader names as labels. The author’s guess is that for some reason complete data for reader 2 could not be obtained. The renumber = TRUE option has the effect of renumbering the readers 1 through 4. Without renumbering, the output would be aesthetically displeasing, but have no effect on the conclusions.

Figures of merit, empirical wAFROC-AUC and empirical ROC-AUC, and the corresponding reader averages for both analyses are summarized in Table 19.1. The weighted AFROC results were obtained by copy and paste operations from worksheet FOMs in file FedwAfroc.xlsx. The highest rating AUC results were obtained by similar operations from worksheet FOMs in Excel file FedHrAuc.xlsx. As expected, each wAFROC-AUC is smaller than the corresponding ROC-AUC.

Table 19.1: Empirical wAFROC-AUC and ROC-AUC for all combinations of treatments and readers, and reader-averaged AUCs for each treatment (Rdr. Avg.). The weighted AFROC results were obtained from worksheet FOMs in file FedwAfroc.xlsx. The highest rating AUC results were obtained from worksheet FOMs in file FedHrAuc.xlsx. The wAFROC-AUCs are smaller than the corresponding ROC-AUCs.

Table 19.2 shows results for RRRC analysis using the wAFROC-AUC FOM. The overall F-test of the null hypothesis that all treatments have the same reader-averaged FOM, rejected the NH: F(4, 36.8) = 7.8, p = 0.00012 . The numerator degree of freedom ndf is I - 1 = 4. Since the null hypothesis is that all treatments have the same FOM, this implies that at least one pairing of treatments yielded a significant FOM difference. The control for multiple testing is in the formulation of the null hypothesis and no further Bonferroni-like2 correction is needed. To determine which specific pairings are significantly different one examines the p-values (listed under Pr>t) in the “95% CI’s FOMs, treatment difference” portion of the table. It shows that the following differences are significant at alpha = 0.05, namely “1 – 3”, “1 – 5”, “2 – 3”, “2 – 5”, “3 – 4” and “4 – 5”; these are indicated by asterisks. The values listed under the “95% CI’s FOMs, each treatment” portion of the table show that treatment 4 yielded the highest FOM (0.769) followed closely by treatments 2 and 1, while treatment 5 had the least FOM (0.714), slightly worse than treatment 3. This explains why the p-value for the difference 4 - 5 is the smallest (0.00007) of all the listed p-values in the “95% CI’s FOMs, each treatment” portion of the table. Each instance where the p-value for the individual treatment comparisons yields a significant p-value is accompanied by a 95% confidence interval that does not include zero. The two statements of significance, one in terms of a p-value and one in terms of a CI, are equivalent. When it comes to presenting results for treatment FOM differences, I prefer the 95% CI but some journals insist on a p-value, even when it is not significant. Note that two sequential tests are involved, an overall F-test of the NH that all treatments have the same performance and only if this yields a significant results is one justified in looking at the p-values of individual treatment pairings.

Table 19.2: wAFROC-AUC analysis: results of random-reader random-case (RRRC) analysis, in worksheet “RRRC”“. [ddf = denominator degrees of freedom of F-distribution. df = degrees of freedom of t-distribution. Stderr = standard error. CI = confidence interval. * = Significantly different at alpha = 0.05.]

Table 19.3 shows corresponding results for the inferred ROC-AUC FOM. Again the null hypothesis was rejected: F(4, 16.8) = 3.46, p = 0.032. This means at least two treatments have significantly different FOMs. Looking down the table, one sees that the same 6 pairs (as compared to wAFROC analysis) are significantly different, 1 – 3, 1- 5, etc., as indicated by the asterisks. The last five rows of the table show that treatment 4 had the highest performance while treatment 5 had the lowest performance. At the 5% significance level, both methods yielded the same significant differences, but this is not always true. While it is incorrect to conclude from a single dataset that a smaller p-value is indicative of higher statistical power, simulation testing under controlled conditions has consistently shown higher statistical power for the wAFROC-AUC FOM3,4 as compared to the inferred ROC-AUC FOM.

Table 19.3: Inferred ROC-AUC analysis: results of random-reader random-case (RRRC) analysis, in worksheet “RRRC”“. ddf = denominator degrees of freedom of F-distribution. df = degrees of freedom of t-distribution. Stderr = standard error. CI = confidence interval; * = Significantly different at alpha = 0.05.].

8.4 TBA Plotting wAFROC and ROC curves

It is important to display empirical wAFROC/ROC curves, not just for publication purposes, but to get a better feel for the data. Since treatments 4 and 5 showed the largest difference, the corresponding /ROC plots for them are displayed. The code is in file mainwAfrocRocPlots.R.

Sourcing this code yields Fig. 19.1. Plot (A), originating from lines 16 – 19, shows individual reader wAFROC plots for treatment 4 (solid lines) and treatment 5 (dashed lines). Running the software on one’s computer best shows the color-coding. While difficult to see, examination of this plot shows that all readers performed better in treatment 4 than in treatment 5 (i.e., for each color the solid line is above the dashed line). Plot (B), originating from lines 21 – 25, shows reader-averaged wAFROC plots for treatments 4 (red line, upper curve) and 5 (blue line, lower curve). If one changes, for example, line 19 from print(plot1$wAFROCPlot)toprint(plot$1wAFROCPoints) the code will output the coordinates of the points describing the curve, which gives the user the option to copy and paste the operating points into alternative plotting software.

Lines 16 – 19 create plots for all specified treatment-reader combinations. The “trick” to creating reader-averaged curves, such as in (B) is defining two list variables, plotT and plotR, at lines 21 – 22, the first containing the treatments to be plotted, list(4,5), and the second, a list of equal length, containing the arrays of readers to be averaged over, list(c(1:4), c(1:4)). More examples can be found in the help page for PlotEmpiricaOperatingCharacteristics().

Meaningful operating points on the reader average curves cannot be defined. This is because ratings are treatment and reader specific labels, so one cannot for example, average bin counts over all readers to construct a table like ROC Table 4.1 or its AFROC counterpart, Table 13.3.

Instead, the following procedure is used internal to PlotEmpiricaOperatingCharacteristics(). The reader-averaged plot for a specified treatment is obtained by dividing the FPF range from 0 to 1 into finely spaced steps of 0.005. For each FPF value the wLLF values for that treatment are averaged over all readers, yielding the reader-averaged ordinate. Calculating confidence intervals on the reader-averaged curve is possible but cumbersome and unnecessary in my opinion. The relevant information, namely the 95% confidence interval on the difference in reader-averaged AUCs, is already contained in the program output, see Table 19.2, row labeled “4 – 5*“. The difference is 0.05488 with a 95% confidence interval (0.03018, 0.07957).

Fig. 19.1: FED dataset; (A): individual reader wAFROC plots for treatments 4 and 5. While difficult to see, all readers performed better in treatment 4 as indicated by each colored solid line being above the corresponding dashed lines. (B): reader-averaged wAFROC plots for treatments 4 and 5. The performance superiority of treatment 4 is fairly obvious in this curve. The difference is significant, p = 0.00012.

Inferred ROC plots corresponding to Fig. 19.1 were generated by lines 20-24, i.e., by changing opChType = “wAFROC” to opChType = “ROC”, and print(plot2$wAFROCPlot)toprint(plot$2ROCPlot), resulting in Fig. 19.2. From Table 19.3 it is seen that the difference in reader-averaged AUCs is 0.04219 with a 95% confidence interval (0.00727, 0.07711). The observed wAFROC effect-size, 0.05488, is larger than the corresponding inferred ROC effect-size, 0.04219. This is a common observation, but sampling variability compounded with small differences, could give different results.

Fig. 19.2: FED dataset; (A): individual reader ROC plots for treatments 4 and 5. While difficult to see, all readers performed better in treatment 4. (B): reader-averaged ROC plots for treatments 4 and 5. The performance superiority of treatment 4 is fairly obvious in this curve. The difference is significant, p = 0.03054.

8.5 Reporting an FROC study

The methods section should make it clear exactly how the study was conducted. The information should be enough to allow some one else to replicate the study. How many readers, how many cases, how many treatments were used. How was ground truth determined and if the FROC paradigm was used, how were true lesion locations determined?

The instructions to the readers should be clearly stated in writing. Precautions to minimize reading order effects should be stated – usually this is accomplished by interleaving cases from different treatments so that the chances that cases from a particular treatment is always seen first by every reader are minimized. Additionally, images from the same case, but in different treatments, should not be viewed in the same reading session. Reading sessions are usually an hour, and the different sessions should ideally be separated by at least one day. Users generally pay minimal attention to training sessions. It is recommended that at least 25% of the total number of interpretations be training cases and cases used for training should not be used in the main study. Feedback should be provided during training session to allow the reader to become familiar with the range of difficulty levels regarding diseased and non-diseased cases in the dataset. Deception, e.g., stating a higher prevalence than is actually used, is usually not a good idea. The user-interface should be explained carefully. The best user interface is intuitive, minimizes keystrokes and requires the least explanation.

In publications, the paradigm used to collect the data (ROC, FROC, etc.) and the figure of merit used for analysis should be stated. If FROC, the proximity criterion should be stated. The analysis should state the NH and the alpha of the test, and the desired generalization. The software used and appropriate references should be cited. The results of the overall F-test, the p-value, the observed F-statistic and its degrees of freedom should be stated. If the NH is not rejected, one should cite the observed inter-treatment FOM differences, confidence intervals and p-values and ideally provide preliminary sample size estimates. This information could be useful to other researchers attempting to conduct a larger study. If the NH is rejected, a table of inter-treatment FOM differences such as Table 19.3 should be summarized. Reader averaged plots of the relevant operating characteristics for each treatment should be provided. In FROC studies it is recommended to vary the proximity criterion, perhaps increasing it by a factor of 2, to test if the final conclusions (is NH rejected and if so which treatment is highest) are unaffected.

Assuming the study has been done properly and with sufficiently large number of cases, the results should be published in some form, even if the NH is not rejected. The dearth of datasets to allow reasonable sample size calculations is a real problem in this field. The dataset set should be made available, perhaps on Research Gate, or if communicated to me, they will be included in the Online Appendix material. Datasets acquired via NIH or other government funding must be made available upon request, in an easily decipherable format. Subsequent users of these datasets must cite the original source of the data. Given the high cost of publishing excess pages in some journals, an expanded version, if appropriate for clarity, should be made available using online posting avenues.

8.6 Crossed-treatment analysis

This analysis was developed for a particular application6 in which nodule detection in an anthropomorphic chest phantom in computed tomography (CT) was evaluated as a function of tube charge and reconstruction method. The phantom was scanned at 4 values of mAs and images were reconstructed with adaptive iterative dose reduction 3D (AIDR3D) and filtered back projection (FBP). Thus there are two treatment factors and the factors are crossed since for each value of the mAs factor there were two values of the reconstruction algorithm factor. Interest was in determining if whether performance depends on mAs and/or reconstruction method.

In a typical analysis of MRMC ROC or FROC study, treatment is considered as a single factor with I levels, where I is usually small. The figure of merit for treatment i (i =1,2,...,I) and reader j (j =1,2,...,J) is denoted ; the case set index is suppressed. MRMC analysis compares the observed magnitude of the difference in reader-averaged figures of merit between treatments i and i', , to the estimated standard deviation of the difference. For example, the reader-averaged difference in figures of merit is , where the dot symbol represents the average over the corresponding (reader) index. The standard deviation of the difference is estimated using the DBMH or the ORH method, using for example jackknifing to determine the variance components and/or covariances. With I levels, the number of distinct i vs. i' comparisons is I (I −1)/2. If the current study were analyzed in this manner, where I =8 (4 levels of mAs and two image reconstruction methods), then this would imply 28 comparisons. The large number of comparisons leads to loss of statistical power in detecting the effect of a specific pair of treatments, and, more importantly, does not inform one of the main points of interest: whether performance depends on mAs and/or reconstruction method. For example, in standard analysis the two reconstruction algorithms might be compared at different mAs levels, and one is in the dark as to which factor (algorithm or mAs) caused the observed significant difference.

Unlike conventional ROC type studies, the images in this study are defined by two factors. The first factor, tube charge, had four levels: 20, 40, 60 and 80 mAs. The second factor, reconstruction method, had two levels: FBP and

AIDR3D. The figure of merit is represented by , where represents the levels of the first factor (mAs), and represents the levels of the second factor (reconstruction method), . Two sequential analyses were performed: (i) mAs analysis, where the figure of merit was averaged over (the reconstruction index); and (ii) reconstruction analysis, where the figure of merit was averaged over (the mAs index). For example, the mAs analysis figure of merit is , where the dot represents the average over the reconstruction index, and the corresponding reconstruction analysis figure of merit is , where the dot represents the average over the mAs index. Thus in either analysis, the figure of merit is dependent on a single treatment factor, and therefore standard DBMH or ORH methods apply.

The mAs analysis determines whether tube charge is a significant factor and in this analysis the number of possible comparisons is only six. The reconstruction analysis determines whether AIDR3D offers any advantage over FBP and in this analysis the number of possible comparisons is only one. Multiple testing on the same dataset increases the probability of Type I error, therefore a Bonferroni correction is applied by setting the threshold for declaring significance at 0.025; this is expected to conservatively maintain the overall probability of a Type I error at $=$ 0.05. Crossed-treatment analysis is used to describe this type of analysis of ROC/FROC data, which yields clearer answers on which of the two factors effects performance. The averaging over the other treatment has the effect of increasing the power of the study in detecting differences in each of the two factors.

Since the phantom is unique, and conclusions are only possible that are specific to this one phantom, the case (or image) factor was regarded as fixed. For this reason only results of random-reader fixed-case analyses are reported.

8.7 Discussion / Summary

An IDL (Interactive Data Language, currently marketed by Exelis Visual Information Solutions, www.exelisvis.com) version of JAFROC was first posted to a now obsolete website on 4/16/2004. This software required a license for IDL, which most users did not have. Subsequently, (9/27/2005) a version was posted which allowed analysis using the freely downloadable IDL Virtual Machine software (a method for freely distributing compiled IDL code). On 1/11/2011 the standalone Windows-compatible version was posted (4.0) and the current version is 4.2. JAFROC is windows compatible (XP, Vista and Windows 7, 8 and10).

To our knowledge JAFROC is the only easily accessible software currently available that can analyze FROC data. Workstation software for acquiring ROC and FROC data is available from several sources7-9. The Windows version is no longer actively supported (bugs, if pointed out, will be corrected). Current effort to conduct research and distribute software uses the R platform10. There are several advantages to this. R is an open-source platform - we have already benefited from a bug pointed out by a user . R runs on practically any platform (Windows, OSX, Linux, etc.). Also, developing an R package benefits from other contributed R-packages, which allow easy computation of probability integrals, random number generation, and parallel computing to speed up computations, to name just a few. The drawback with R, and this has to with its open source philosophy, is that one cannot readily integrate existing ROC code, developed on other platforms and other programming languages (specifically, DLLs are not allowed in R). So useful programs like CORROC2 and CBM were coded in C++, since R allows C++ programs to be compiled and included in a package.

Due to the random number of marks per image, data entry in the FROC paradigm is inherently more complicated and error-prone than in ROC analysis, and consequently, and in response to feedback from users, much effort has gone into error checking. The users have especially liked the feature where the program indicates the Excel sheet name and line-number where an error is detected. User-feedback has also been very important in detecting program bugs and inconsistencies in the documentation and developing additional features (e.g., ROI analysis).

Interest in the FROC paradigm is evidenced by the fact that Ref. 3 describing the JAFROC method has been cited over 273 times. Over 25,000 unique visitors have viewed my website, at least 73 have downloaded the software and over 107 publications using JAFROC have appeared. The list is available on my website. JAFROC has been applied to magnetic resonance imaging, virtual computerized tomography colonoscopy, digital tomosynthesis (chest and breast), mammography dose and image processing optimization, computer aided detection (CAD), computerized tomography, and other applications.

Since confusion still appears to exist, especially among statisticians, regarding perceived neglect of intra-image correlations of ratings and how true negatives are handled in FROC analysis11, we close with a quote from respected sources 12 “(Chakraborty and Berbaum) have presented a solution to the FROC problem using a jackknife resampling approach that respects the correlation structure in the images … their paradigm successfully passes a rigorous statistical validation test". Since 2005 the National Institutes for Health (NIH) has been generous with supporting the research and users of JAFROC have been equally generous with providing their datasets, which have resulted in several collaborations.

FROC sample size

Chapter 9

Sample size estimation for FROC studies

9.1 TBA How much finished

80%

9.2 Overview

This chapter is split into two parts.

- The first part goes into the details of FROC paradigm sample size estimation.
- The second part encapsulates most of the details in a new function `SsFrocNhRsmModel()`, which encapsulates some of the code in the first part (that relating to building the NH model), thereby making it easier for the user to perform FROC sample size estimation.

These parts are independently included in two `RJafroc` vignettes (on the GitHub `master` branch, not the CRAN uploaded version). These are located at:

https://dpc10ster.github.io/RJafroc/articles/Ch19Vig1FrocSampleSize.html

https://dpc10ster.github.io/RJafroc/articles/Ch19Vig2FrocSampleSize.html

9.3 Part 1

9.3.1 Introduction

FROC sample size estimation is not fundamentally different from the procedure outlined in TBA Chapter 11 for the ROC paradigm. To recapitulate, based on analysis of a pilot ROC dataset and using a specified FOM, e.g., `FOM = Wilcoxon`, and either `method = "DBM"` or `method = "OR"` for significance testing, one estimates the intrinsic variability of the data expressed in terms of variance components. For DBM analysis, these are the pseudovalue variance components, while for OR analysis these are the FOM treatment-reader variance component and the FOM covariances. The second step is to specify a clinically realistic effect-size, e.g., the anticipated AUC difference between the two modalities. Given these values, the sample size functions implemented in `RJafroc` (beginning with `Ss`) allow one to estimate the number of readers and cases necessary to detect (i.e., reject the null hypothesis) the modality AUC difference at specified Type II error rate β, typically chosen to be 20% - corresponding to 80% statistical power - and specified Type I error rate α, typically chosen to be 5%.

In FROC analysis the only difference, indeed the critical difference, is the choice of FOM; e.g., `FOM = "wAFROC"` instead of the inferred ROC-AUC, `FOM = "HrAuc"`. The FROC dataset is analyzed using either the DBM or the OR method. This yields the necessary variance components or the covariance matrix corresponding to the wAFROC-AUC. The next step is to specify the effect-size **in wAFROC-AUC units**. The ROC-AUC has a historically well-known interpretation: the classification ability at separating diseased patients from non-diseased patients while the wAFROC-AUC does not. Needed is a way of relating the effect-size in ROC-AUC units to one in wAFROC-AUC units: as should be obvious this requires a physical model, e.g., the RSM, that predicts both ROC and wAFROC curves and the respective AUCs.

1. One chooses an ROC-AUC effect-size that is realistic, one that clinicians understand and can therefore participate in, in the effect-size postulation process. Lacking such information I recommend, based on past ROC studies, 0.03 as typical of a small effect size and 0.05 as typical of a moderate effect size.
2. One converts the ROC effect-size to a wAFROC-AUC effect-size. The method for this is described in the next section.
3. One uses the sample size tools in in `RJafroc` to determine sample size or power.

It is important to recognize is that all quantities have to be in the same units. When performing ROC analysis, everything (variance components and effect-size) has to be in units of the selected FOM, e.g., `FOM = "Wilcoxon"` which is identical to the empirical ROC-AUC. When doing wAFROC analysis, everything has to be in units of the wAFROC-AUC, i.e., `FOM = "wAFROC"`. The variance components and effect-size in wAFROC-AUC units will be different from their corresponding ROC counterparts. In particular, as shown next, an ROC-AUC effect-size of 0.05 generally corresponds to a larger effect-size in wAFROC-AUC units. The reason for this is that the range over which wAFROC-AUC can vary, namely 0 to 1, is twice the corresponding ROC-AUC range.

The next section explains the steps used to implement #2 above.

9.3.2 Relating an ROC effect-size to a wAFROC effect-size

- If there are more than two treatments in the pilot dataset extract those treatments that represent null hypothesis data: for example `DfExtractDataset(dataset, trts = c(1,2,4))` extracts treatments 1, 2 and 4. Save the extracted dataset to `dataset`. More than two treatments can be used if they have similar FOMs as the averaging/ian procedure described below benefits from more data. However, the final sample size predictions are restricted to two treatments.

- If the original data is FROC, convert it to ROC using `DfFroc2Roc(dataset)`: this is because **the RSM fits binned ROC data**. The ROC dataset is `rocDataset`.

- If the data uses continuous scale ratings, bin the data using `DfBinDataset(rocDataset, opChType = "ROC")`. The default number of bins should be used. Unlike binning using arbitrarily set thresholds, the thresholds found by `DfBinDataset` are unique in that they maximize ROC-AUC. The binned dataset is `rocDatasetB`.

- For each treatment and reader the inferred ROC data is fitted by `FitRsmRoc()`, see example below, yielding estimates of the RSM *physical* (or pri) parameters (not the *intrinsic* values).

- The following example uses the *first two* treatments of the "FED" dataset, `dataset04`, which is a 5 treatment 4 radiologist FROC dataset acquired by Dr. Federica Zanca et. al. (Zanca et al., 2009). The dataset has 5 treatments and 4 readers and 200 cases and was acquired on a 5-point integer scale, i.e., it is already binned. If not one needs to bin the dataset using `DfBinDataset()`. I need to emphasize this point: **if the dataset represents continuous ratings, as with a CAD algorithm, one must bin the dataset**.

- The reason for using RSM parameter values only for the first two treatments is that these were found (Zanca et al., 2009) to be almost equivalent (more precisely, the NH could not be rejected for the first two treatments, so it makes sense to regard them as "almost" NH treatments.

- The following code block defines the pilot FROC data `frocData` (corresponding to `dataset04`, which is the "FED" dataset, but with only treatments 1 and 2 extracted, using `DfExtractDataset()`) and `rocData`, i.e., the highest-rating ROC dataset inferred from the FROC dataset using `DfFroc2Roc()`.

```
frocData <- DfExtractDataset(dataset04, trts = c(1,2))
rocData <- DfFroc2Roc(frocData)
rocDataB <- DfBinDataset(rocData, opChType = "ROC") # unnecessary as data is already binned
# but cant hurt
```

The next code block determines `lesDistr`, the lesion distribution array, which has `Lmax` (maximum number of lesions per diseased case over the dataset) rows and two columns. The first column contains the integers 1, 2, ..., `Lmax` and the second column contains the fraction of diseased cases with the number of lesions per case specified in the first column. The second column will sum to unity. The RSM fitting algorithm needs to know how lesion-rich the dataset is, as the RSM predicted ROC-AUC depends on the lesion-richness of the dataset. For reasons that will become clear below, one also needs the distribution of the lesion weights. Note that `lesDist` is determined from the FROC dataset as the lesion-richness information is lost upon conversion to an ROC dataset. The `PlotRsmOperatingCharacteristics` function used below sets `relWeights = 0`, which ensures equally weighted lesions: on cases with one lesion the weight of the lesion is unity, on cases with two lesions the weights of each lesion is 1/2 and on cases with three lesions the weight of each lesion is 1/3, etc.. TBA due to changes in lesDist etc.

```
lesDistr <- UtilLesionDistrVector(frocData)
print(lesDistr)
```

```
## [1] 0.69 0.20 0.11
```

For this dataset `Lmax` is 3, and fraction 0.69 of diseased cases have one lesion, fraction 0.2 of diseased cases have two lesions and fraction 0.11 of diseased cases have three lesions.

The next code block determines the number of treatments and readers (`I` and `J`) from the dimensions of the `frocData$ratings$NL` array. It creates an array `RsmParms` to hold the RSM fitted parameter values. For each treatment and reader it applies the fitting algorithm `FitRsmRoc()`. The first three returned values are `mu`, `lambda` and `nu`, corresponding to RSM parameters μ, λ' and ν'.

```
I <- dim(frocData$ratings$NL)[1]
J <- dim(frocData$ratings$NL)[2]
RsmParms <- array(dim = c(I,J,3))
for (i in 1:I) {
  for (j in 1:J)  {
    x1 <- FitRsmRoc(rocDataB, trt = i, rdr = j, lesDistr)
    RsmParms[i,j,1] <- x1[[1]] # mu
    RsmParms[i,j,2] <- x1[[2]] # lambda
    RsmParms[i,j,3] <- x1[[3]] # nu
  }
}
```

I recommend taking the ian of each of the parameters, over all treatment-reader indices, as representing the average NH dataset. The ian is less sensitive to outliers than the mean.

```
mu <- median(RsmParms[,,1])
lambda <- median(RsmParms[,,2])
nu <- median(RsmParms[,,3])
```

The defining values of the fitting model are `mu` = 3.3121519, `lambda` = 1.714368 and `nu` = 0.7036564. Note that these obey the constraints `lambda > 0` and `0 < nu < 1`. We are now ready to calculate the expected NH FOMs using the ROC -AUC and the wAFROC FOM.

```
aucRocNH <- PlotRsmOperatingCharacteristics(mu, lambda, nu,
                                            lesDistr = lesDistr, OpChType = "ROC")$aucROC
aucwAfrocNH <- PlotRsmOperatingCharacteristics(mu, lambda, nu,
                                               lesDistr = lesDistr, OpChType = "wAFROC")$aucwAFROC
```

- The plotting function `PlotRsmOperatingCharacteristics()` returns a number of other objects, most importantly the plot, but here we use only the AUC, which is obtained by numerical integration of the predicted operating characteristics.

- One has `aucRocNH = 0.8791542` and `aucwAfrocNH = 0.7199233`. Note that the wAFROC-FOM is smaller than the ROC-FOM as it includes the localization constraint.

- To induce the alternative hypothesis condition, one increments μ_{NH} by Δ_{μ}. The resulting ROC-AUC and wAFROC-AUC are calculated, again by numerical integration of the RSM predicted ROC and wAFROC curves, leading to the corresponding effect-sizes (note that in each equation below one takes the difference between the AH value minus the NH value):

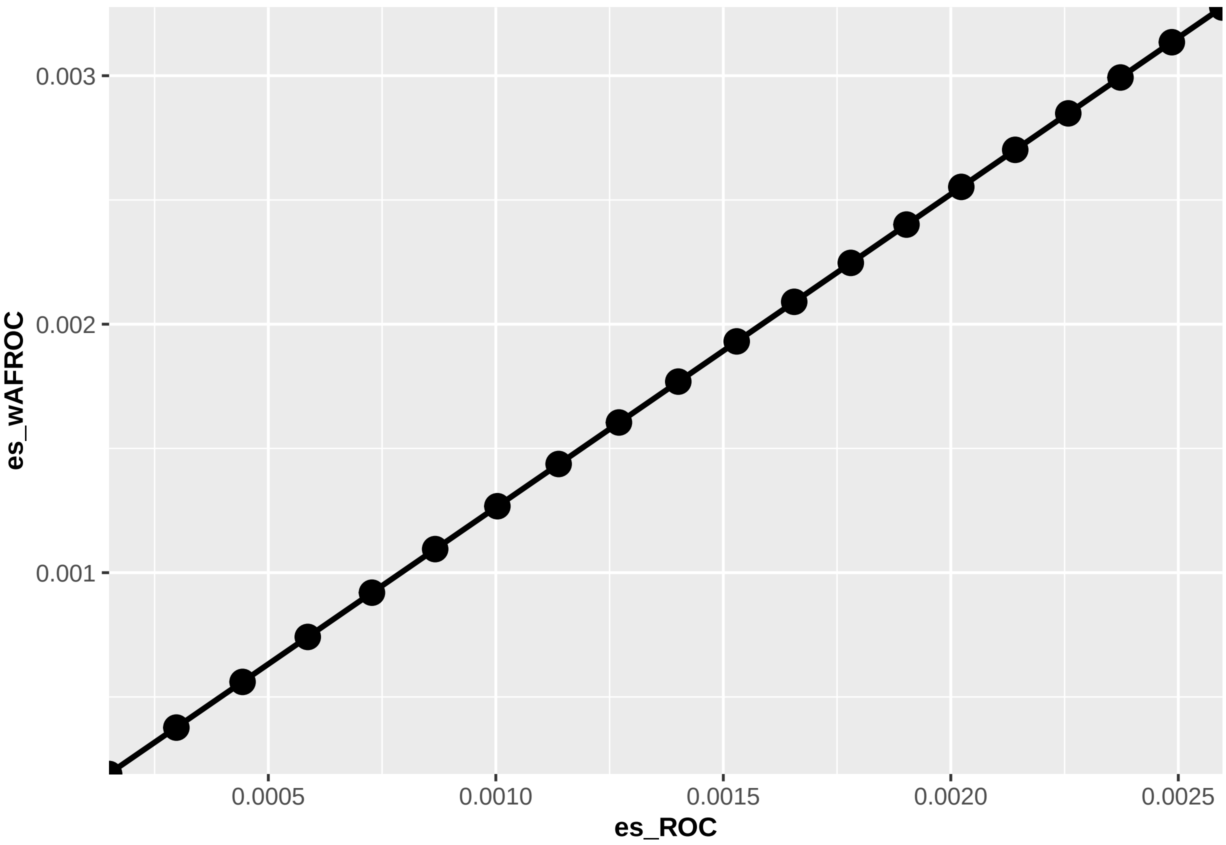
- The next step is to calculate the effect size (new value minus the NH value) using ROC and wAFROC FOMs for a series of specified `deltaMu` values. This generates values that can be used to interpolate a wAFROC effect size for a specified ROC effect size.

```r
deltaMu <- seq(0.01, 0.2, 0.01) # values of deltaMu to scan below
esRoc <- array(dim = length(deltaMu));eswAfroc <- array(dim = length(deltaMu))
for (i in 1:length(deltaMu)) {
  esRoc[i] <- PlotRsmOperatingCharacteristics(
    mu + deltaMu[i], lambda, nu, lesDistr = lesDistr, OpChType = "ROC")$aucROC - aucRocNH
  eswAfroc[i] <- PlotRsmOperatingCharacteristics(
    mu+ deltaMu[i], lambda, nu, lesDistr = lesDistr, OpChType = "wAFROC")$aucwAFROC - aucwAfrocNH
  cat("ES_ROC = ", esRoc[i], ", ES_wAFROC = ", eswAfroc[i],"\n")
}
```

```
## ES_ROC =  0.0001500649 , ES_wAFROC =  0.000189712
## ES_ROC =  0.0002978454 , ES_wAFROC =  0.0003764812
## ES_ROC =  0.0004433681 , ES_wAFROC =  0.0005603432
## ES_ROC =  0.0005866597 , ES_wAFROC =  0.0007413331
## ES_ROC =  0.0007277463 , ES_wAFROC =  0.0009194859
## ES_ROC =  0.0008666543 , ES_wAFROC =  0.001094837
## ES_ROC =  0.001003409 , ES_wAFROC =  0.00126742
## ES_ROC =  0.001138038 , ES_wAFROC =  0.001437269
## ES_ROC =  0.001270565 , ES_wAFROC =  0.00160442
## ES_ROC =  0.001401017 , ES_wAFROC =  0.001768904
## ES_ROC =  0.001529418 , ES_wAFROC =  0.001930757
## ES_ROC =  0.001655794 , ES_wAFROC =  0.002090012
## ES_ROC =  0.00178017 , ES_wAFROC =  0.002246701
## ES_ROC =  0.00190257 , ES_wAFROC =  0.002400857
## ES_ROC =  0.002023021 , ES_wAFROC =  0.002552513
## ES_ROC =  0.002141544 , ES_wAFROC =  0.002701702
## ES_ROC =  0.002258166 , ES_wAFROC =  0.002848455
## ES_ROC =  0.002372911 , ES_wAFROC =  0.002992804
## ES_ROC =  0.002485801 , ES_wAFROC =  0.00313478
## ES_ROC =  0.002596862 , ES_wAFROC =  0.003274416
```

Here is a plot of wAFROC effect size (y-axis) vs. ROC effect size.

```r
df <- data.frame(es_ROC = esRoc, es_wAFROC = eswAfroc)
p <- ggplot(data = df, aes(x = es_ROC, y = es_wAFROC)) +
  geom_smooth(method = "lm", se = FALSE, color = "black", formula = y ~ x) +
  geom_point(size = 4) +
  scale_color_manual(values = "black") +
  theme(axis.title.y = element_text(size = 10,face="bold"),
        axis.title.x = element_text(size = 10,face="bold")) +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0))
print(p)
```



The plot is very close to linear. This makes it easy to design an interpolation function. In the following code block the first line fits `eswAfroc` vs. `esRoc` using the linear model `lm()` function constrained to pass through the origin (the minus one): `scaleFactor <- lm(eswAfroc ~ -1 + esRoc)`. One expects this constraint since for `deltaMu = 0` the effect size must be zero no matter how it is measured.

```
scaleFactor<-lm(eswAfroc~-1+esRoc) # fit values to straight line thru origin
effectSizeROC <- seq(0.01, 0.1, 0.01)
effectSizewAFROC <- effectSizeROC*scaleFactor$coefficients[1] # r2 = summary(scaleFactor)$r.squared
```

The slope of the zero-intercept constrained straight line fit is `scaleFactor` = 1.2617239 and the squared correlation coefficient is `R2` = 0.9999997. Therefore, the conversion from ROC to wAFROC effect size is: `effectSizewAFROC = scaleFactor * effectSizeROC`. **The wAFROC effect size is twice the ROC effect size.** All that remains is to calculate the variance components using the two FOMs.

9.3.3 Computing the respective variance components

The code block applies `StSignificanceTesting()` to `rocData` and `frocData`, using the appropriate FOM, and extracts the variance components.

```
temp1 <- StSignificanceTesting(rocData, FOM = "Wilcoxon", method = "DBM", analysisOption = "RRRC")
temp2 <- StSignificanceTesting(frocData, FOM = "wAFROC", method = "DBM", analysisOption = "RRRC")
varCompROC <- temp1$ANOVA$VarCom
varCompwAFROC <- temp2$ANOVA$VarCom
```

The observed wAFROC effect-size is -0.00685625. This is a very small effect size; the corresponding ROC effect-size is -0.0051; the sign does not affect the calculations, which is too small to reach 80% power. It is not surprising that the study (Zanca et al., 2009) did not find a significant difference between these two treatments

The respective variance components are:

```
print(varCompROC)
```

```
##            Estimates
## VarR   0.00082773798
## VarC   0.03812334734
## VarTR  0.00015265067
## VarTC  0.00964432675
## VarRC  0.00354419640
## VarErr 0.09484636574
```

```
print(varCompwAFROC)
```

```
##             Estimates
## VarR    0.00185422886
## VarC    0.06117804981
## VarTR  -0.00044392794
## VarTC   0.01016518621
## VarRC   0.01355883396
## VarErr  0.09672559908
```

Only terms involving treatment are relevant to sample size. The wAFROC `varTC` and `varErr` values are slightly larger than the ROC ones - as expected - because the range of the wAFROC FOM is twice that of the ROC FOM.

9.3.4 Comparing ROC power to wAFROC power for equivalent effect-sizes

We are now ready to compare ROC and wAFROC powers for equivalent effect sizes. The following example is for 5 readers (`JPivot`) and 100 cases (`KPivot`) in the **pivotal study**.

```
powerROC <- array(dim = length(effectSizeROC));powerwAFROC <- array(dim = length(effectSizeROC))

JPivot <- 5;KPivot <- 100
for (i in 1:length(effectSizeROC)) {
  varYTR <- varCompROC["VarTR","Estimates"] # these are pseudovalue based variance components assuming FOM
  varYTC <- varCompROC["VarTC","Estimates"]
  varYEps <- varCompROC["VarErr","Estimates"]
  ret <- SsPowerGivenJK(dataset = NULL, FOM = "Wilcoxon", J = JPivot, K = KPivot, analysisOption = "RRRC",
        list(VarTR = varYTR, VarTC = varYTC, VarErr = varYEps))
  powerROC[i] <- ret$powerRRRC

  varYTR <- varCompwAFROC["VarTR","Estimates"] # these are pseudovalue based variance components assuming
  varYTC <- varCompwAFROC["VarTC","Estimates"]
  varYEps <- varCompwAFROC["VarErr","Estimates"]
  ret <- SsPowerGivenJK(dataset = NULL, FOM = "Wilcoxon", J = JPivot, K = KPivot, analysisOption = "RRRC",
        list(VarTR = varYTR, VarTC = varYTC, VarErr = varYEps))
  powerwAFROC[i] <- ret$powerRRRC

  cat("ROC-ES = ", effectSizeROC[i], ", wAFROC-ES = ", effectSizewAFROC[i],
      ", Power-ROC = ", powerROC[i], ", Power-wAFROC = ", powerwAFROC[i], "\n")
}
```

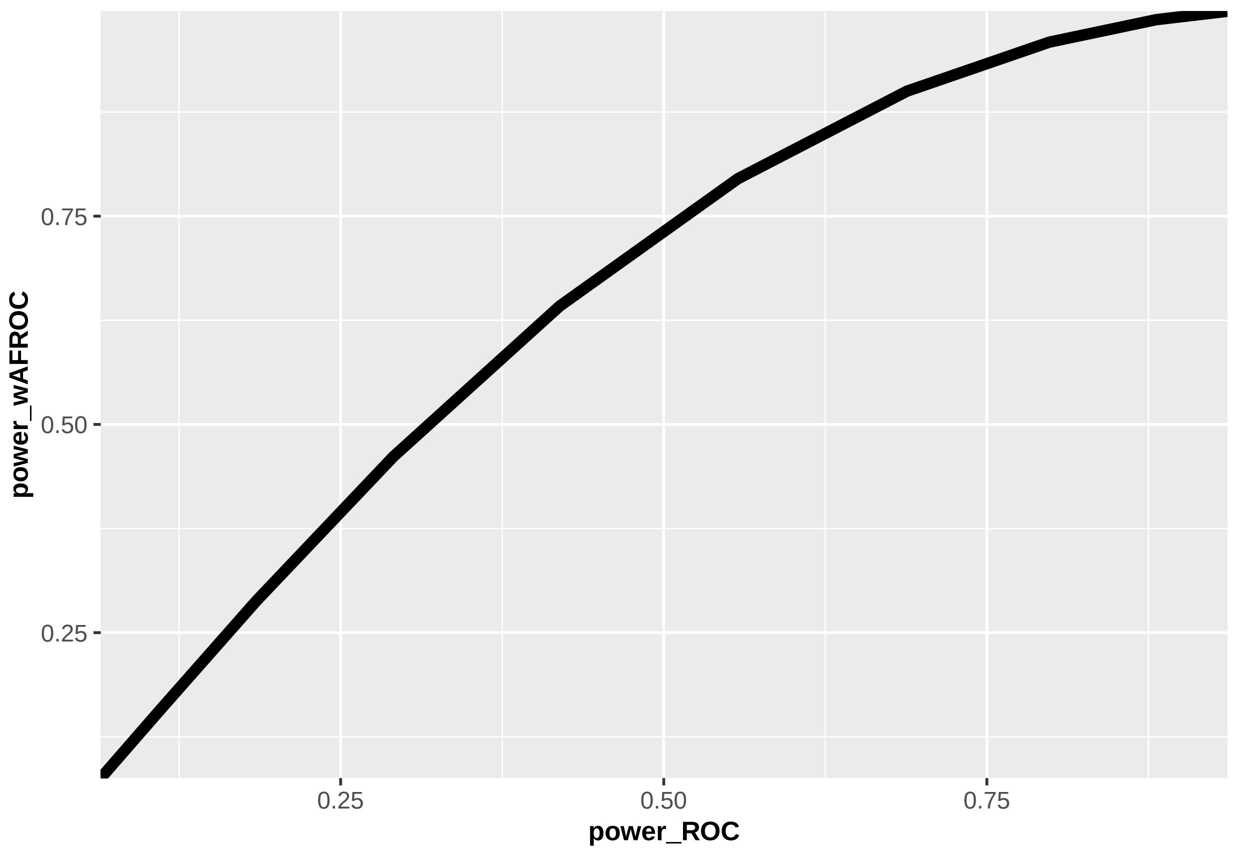
```
## ROC-ES =  0.01 , wAFROC-ES =  0.012617239 , Power-ROC =  0.064430457 , Power-wAFROC =  0.075439644
## ROC-ES =  0.02 , wAFROC-ES =  0.025234479 , Power-ROC =  0.10878897 , Power-wAFROC =  0.15449773
## ROC-ES =  0.03 , wAFROC-ES =  0.037851718 , Power-ROC =  0.18471152 , Power-wAFROC =  0.28797922
## ROC-ES =  0.04 , wAFROC-ES =  0.050468957 , Power-ROC =  0.29079274 , Power-wAFROC =  0.4612966
## ROC-ES =  0.05 , wAFROC-ES =  0.063086196 , Power-ROC =  0.41954431 , Power-wAFROC =  0.6420946
## ROC-ES =  0.06 , wAFROC-ES =  0.075703436 , Power-ROC =  0.55738123 , Power-wAFROC =  0.79495349
## ROC-ES =  0.07 , wAFROC-ES =  0.088320675 , Power-ROC =  0.68816012 , Power-wAFROC =  0.90003872
## ROC-ES =  0.08 , wAFROC-ES =  0.10093791 , Power-ROC =  0.79836108 , Power-wAFROC =  0.95891383
## ROC-ES =  0.09 , wAFROC-ES =  0.11355515 , Power-ROC =  0.88095077 , Power-wAFROC =  0.98585038
## ROC-ES =  0.1 , wAFROC-ES =  0.12617239 , Power-ROC =  0.93606799 , Power-wAFROC =  0.9959336
```

Since the wAFROC effect size is about a factor of two larger than the ROC effect size, wAFROC power is larger than that for ROC. The effect is magnified as the effect size enters as the square in the formula for the power (this overwhelms the slight increase in variability of wAFROC-FOM relative to ROC-FOM noted previously). The following is a plot of the respective powers.

```r
df <- data.frame(power_ROC = powerROC, power_wAFROC = powerwAFROC)
p <- ggplot(mapping = aes(x = power_ROC, y = power_wAFROC)) +
  geom_line(data = df, size = 2)+
  scale_color_manual(values = "black") +
  theme(axis.title.y = element_text(size = 10,face="bold"),
        axis.title.x = element_text(size = 10,face="bold"))  +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0))
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
```

```r
print(p)
```



9.4 Part 2

9.4.1 Introduction

This example uses the FED dataset as a pilot FROC study and function `SsFrocNhRsmModel()` to construct the NH model (encapsulating some of the code in the first part).

9.4.2 Constructing the NH model for the dataset

One starts by extracting the first two treatments from `dataset04`, which represent the NH dataset, see previous part. Next one constructs the NH model - note that the lesion distribution `lesDistr` can be specified here independently of that in the pilot dataset. This allows some control over selection of the diseased cases in the pivotal study.

```
lesDistr <- c(0.7, 0.2, 0.1)
frocNhData <- DfExtractDataset(dataset04, trts = c(1,2))
ret <- SsFrocNhRsmModel(frocNhData, lesDistr = lesDistr)
mu <- ret$mu
lambda <- ret$lambda
nu <- ret$nu
scaleFactor <- ret$scaleFactor
```

The fitting model is defined by `mu` = 3.31491361, `lambda` = 1.6930673 and `nu` = 0.70649936 and `lesDistr`. The effect size scale factor is 1.25445833.

```
aucRocNH <- PlotRsmOperatingCharacteristics(mu, lambda, nu,
                                            lesDistr = lesDistr, OpChType = "ROC")$aucROC
aucwAfrocNH <- PlotRsmOperatingCharacteristics(mu, lambda, nu,
                                               lesDistr = lesDistr, OpChType = "wAFROC")$aucwAFROC
```

The null hypothesis ROC AUC is 0.8790548 and the corresponding NH wAFROC AUC is 0.72320816.

9.4.3 Extracting the wAFROC variance components

The next code block applies `StSignificanceTesting()` to `frocNhData`, using `FOM = "wAFROC"` and extracts the variance components.

```
varCompwAFROC  <- StSignificanceTesting(frocNhData, FOM = "wAFROC", method = "DBM", analysisOption = "RRRC
```

9.4.4 wAFROC power for specified ROC effect size, number of readers J and number of cases K

The following example is for ROC effect size = 0.05, 5 readers (`J = 5`) and 100 cases (`K = 100`) in the **pivotal study**.

```
ROC_ES <- 0.05
effectSizewAFROC <- scaleFactor * ROC_ES
J <- 5;K <- 100

varYTR <- varCompwAFROC["VarTR","Estimates"]
varYTC <- varCompwAFROC["VarTC","Estimates"]
varYEps <- varCompwAFROC["VarErr","Estimates"]
ret <- SsPowerGivenJK(dataset = NULL, FOM = "Wilcoxon", J = J, K = K, analysisOption = "RRRC",
                      effectSize = effectSizewAFROC, method = "DBM", LegacyCode = TRUE,
                      list(VarTR = varYTR,
                           VarTC = varYTC,
                           VarErr = varYEps))
powerwAFROC <- ret$powerRRRC

cat("ROC-ES = ", ROC_ES, ", wAFROC-ES = ", ROC_ES * scaleFactor, ", Power-wAFROC = ", powerwAFROC, "\n")
```

```
## ROC-ES =  0.05 , wAFROC-ES =  0.062722916 , Power-wAFROC =  0.63713244
```

9.4.5 wAFROC number of cases for 80% power for a given number of readers J

```
VarTR <- varCompwAFROC["VarTR","Estimates"]
VarTC <- varCompwAFROC["VarTC","Estimates"]
VarErr <- varCompwAFROC["VarErr","Estimates"]
ret2 <- SsSampleSizeKGivenJ(dataset = NULL, J = 6, effectSize = effectSizewAFROC, method = "DBM", LegacyCo
                            list(VarTR = VarTR, VarTC = VarTC, VarErr = VarErr))

cat("ROC-ES = ", ROC_ES, ", wAFROC-ES = ", ROC_ES * scaleFactor,
    ", K80RRRC = ", ret2$KRRRC, ", Power-wAFROC = ", ret2$powerRRRC, "\n")
```

```
## ROC-ES =  0.05 , wAFROC-ES =  0.062722916 , K80RRRC =  123 , Power-wAFROC =  0.80210887
```

9.4.6 wAFROC-FOM power for a given number of readers J and cases K

```
ret3 <- SsPowerGivenJK(dataset = NULL, J = 6, K = ret2$KRRRC, effectSize = effectSizewAFROC, method = "DBM
                       list(VarTR = VarTR, VarTC = VarTC, VarErr = VarErr))

cat("ROC-ES = ", ROC_ES, ", wAFROC-ES = ", ROC_ES * scaleFactor,
    ", powerRRRC = ", ret3$powerRRRC, "\n")
```

```
## ROC-ES =  0.05 , wAFROC-ES =  0.062722916 , powerRRRC =  0.80210887
```

The estimated power is close to 80% as the number of cases (`ret2$KRRRC = 123`) was chosen deliberately from the previous code block.

RJafroc Vignettes

Chapter 10

F-distribution

10.1 TBA How much finished

10%

10.2 Background

A number of vignettes used to be part of the `RJafroc` package. These have been moved here.

10.3 Introduction

Since it plays an important role in sample size estimation, it is helpful to examine the behavior of the F-distribution. In the following `ndf` = numerator degrees of freedom, `ddf` = denominator degrees of freedom and `ncp` = non-centrality parameter (i.e., the Δ appearing in Eqn. (11.6) of (Chakraborty, 2017)).

The use of three `R` functions is demonstrated.

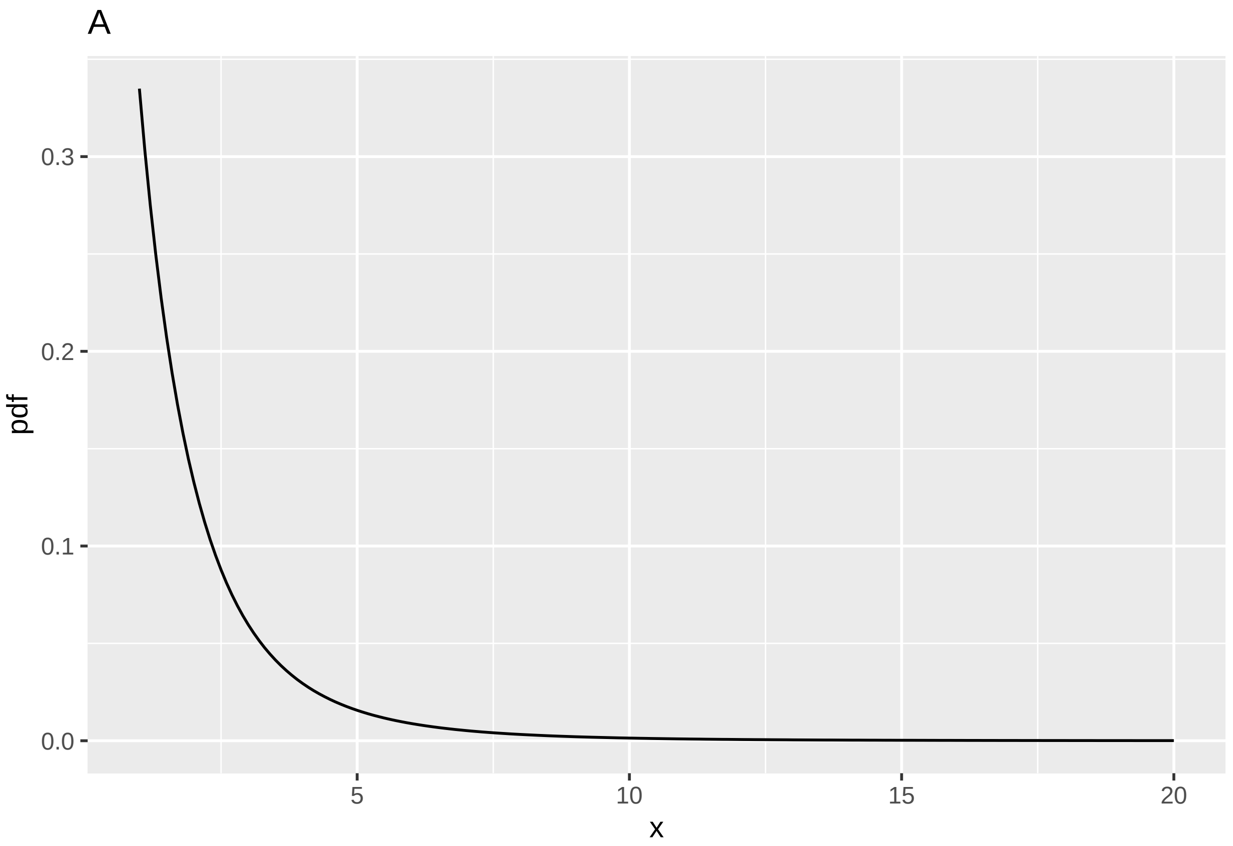
- `qf(p,ndf,ddf)` is the *quantile* function of the F-distribution for specified values of `p`, `ndf` and `ddf`, i.e., the value `x` such that fraction `p` of the area under the F-distribution lies to the right of `x`. Since `ncp` is not included as a parameter, the default value, i.e., zero, is used. This is called the *central* F-distribution.
- `df(x,ndf,ddf,ncp)` is the probability density function (*pdf*) of the F-distribution, as a function of `x`, for specified values of `ndf`, `ddf` and `ncp`.
- `pf(x,ndf,ddf,ncp)` is the probability (or cumulative) distribution function of the F-distribution for specified values of `ndf`, `ddf` and `ncp`.

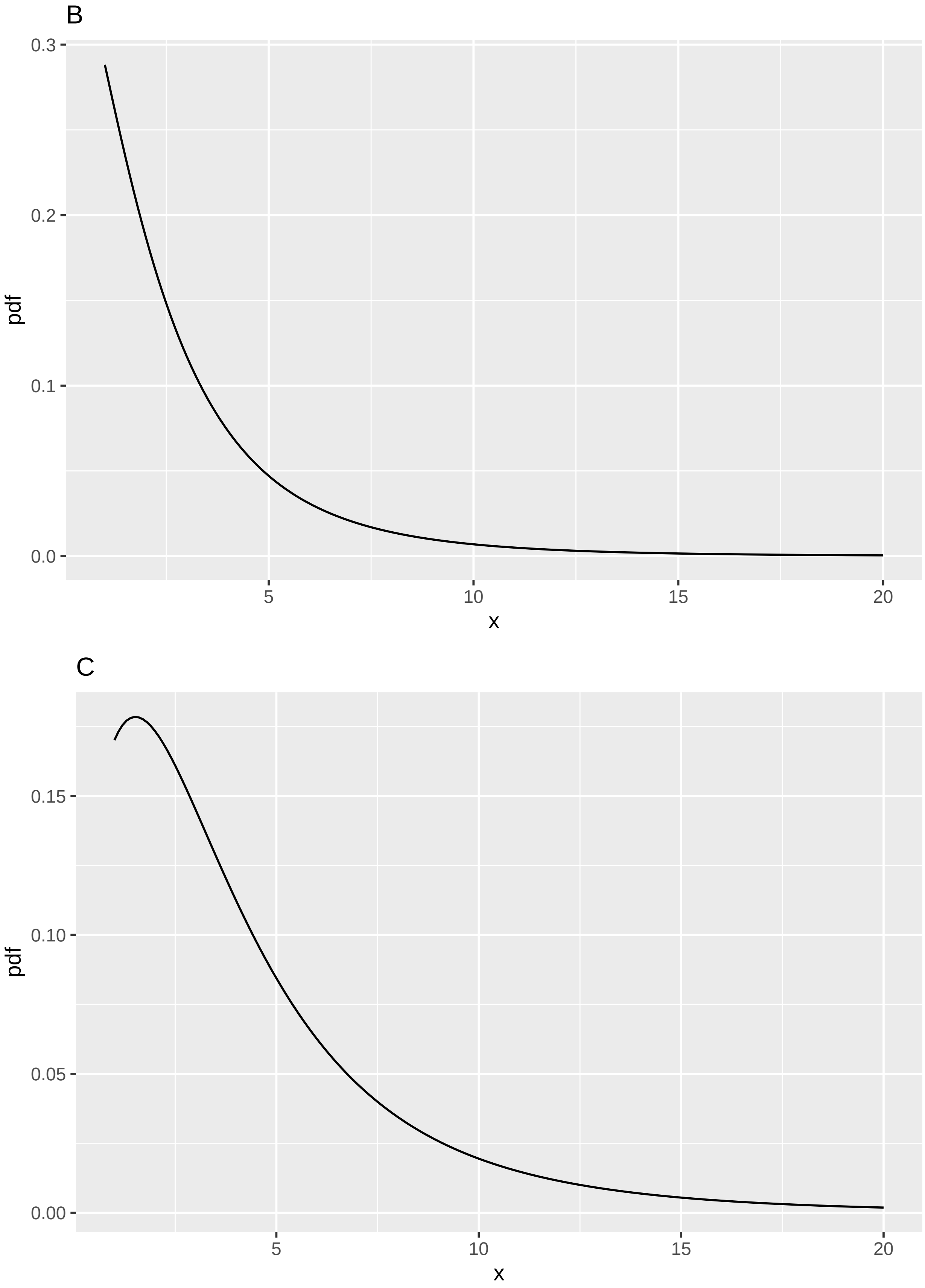
10.4 Effect of `ncp` for `ndf` = 2 and `ddf` = 10

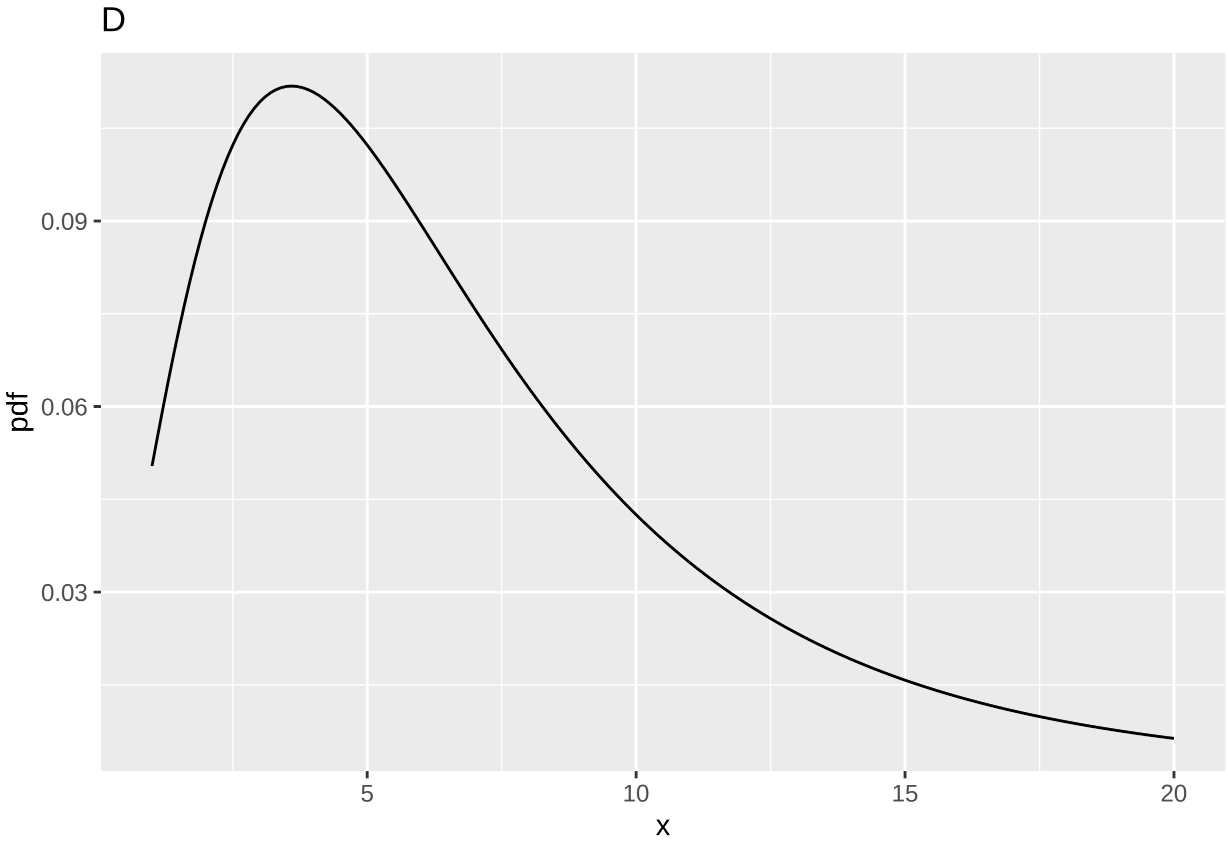
- Four values of `ncp` are considered (0, 2, 5, 10) for `ddf` = 10.
- `fCrit` is the critical value of the F distribution, i.e., that value such that fraction α of the area is to the right of the critical value, i.e., `fCrit` is identical in statistical notation to $F_{1-\alpha,ndf,ddf}$.

```
ndf <- 2;ddf <- 10;ncp <- c(0,2,5,10)
alpha <- 0.05
fCrit <- qf(1-alpha, ndf,ddf)
```

```r
x <- seq(1, 20, 0.1)
myLabel <- c("A", "B", "C", "D")
myLabelIndx <- 1
pFgtFCrit <- NULL
for (i in 1:length(ncp))
{
  y <- df(x,ndf,ddf,ncp=ncp[i])
  pFgtFCrit <- c(pFgtFCrit, 1-pf(fCrit, ndf, ddf, ncp = ncp[i]))
}
for (i in 1:length(ncp))
{
  y <- df(x,ndf,ddf,ncp=ncp[i])
  curveData <- data.frame(x = x, pdf = y)
  curvePlot <- ggplot(data = curveData, mapping = aes(x = x, y = pdf)) +
    geom_line() +
    ggtitle(myLabel[myLabelIndx]);myLabelIndx <- myLabelIndx + 1
  print(curvePlot)
}
fCrit_2_10 <- fCrit # convention fCrit_ndf_ddf
```




B
0.3
0.2
0.1
0.0
pdf
5
10
15
20
x
C
0.15
0.10
0.05
0.00
pdf
5
10
15
20
x



	ndf	ddf	fCrit	ncp	pFgtFCrit
A	2	10	4.102821	0	0.0500000
B	2	10	4.102821	2	0.1775840
C	2	10	4.102821	5	0.3876841
D	2	10	4.102821	10	0.6769776

10.5 Comments

10.5.1 Fig. A

- This corresponds to `ncp = 0`, i.e., the *central* F-distribution.
- The integral under this distribution is unity (this is also true for all plots in this vignette).
- The critical value, `fCrit` in the above code block, is the value of `x` such that the probability of exceeding `x` is α. The corresponding parameter `alpha` is defined above as 0.05.
- In the current example `fCrit` = 4.102821. Notice the use of the quantile function `qf()` to determine this value, and the default value of `ncp`, namely zero, is used; specifically, one does not pass a 4th argument to `qf()`.
- **The decision rule for rejecting the NH uses the NH distribution of the F-statistic**, i.e., reject the NH if F >= `fCrit`. As expected, `prob > fCrit` = 0.05 because this is how `fCrit` was defined.

10.5.2 Fig. B

- This corresponds to `ncp = 2`, `ndf` = 2 and `ddf` = 10.
- The distribution is slightly shifted to the right as compared to Fig. A, thereby making it more likely that the observed value of the F-statistic will exceed the critical value determined for the NH distribution.
- In fact, `prob > fCrit` = 0.177584, i.e., the *statistical power* (compare this to Fig. A where `prob > fCrit` was 0.05).

10.5.3 Fig. C

- This corresponds to `ncp = 5`, `ndf` = 2 and `ddf` = 10.
- Now `prob > fCrit` = 0.3876841.
- Power has increased compared to Fig. B.

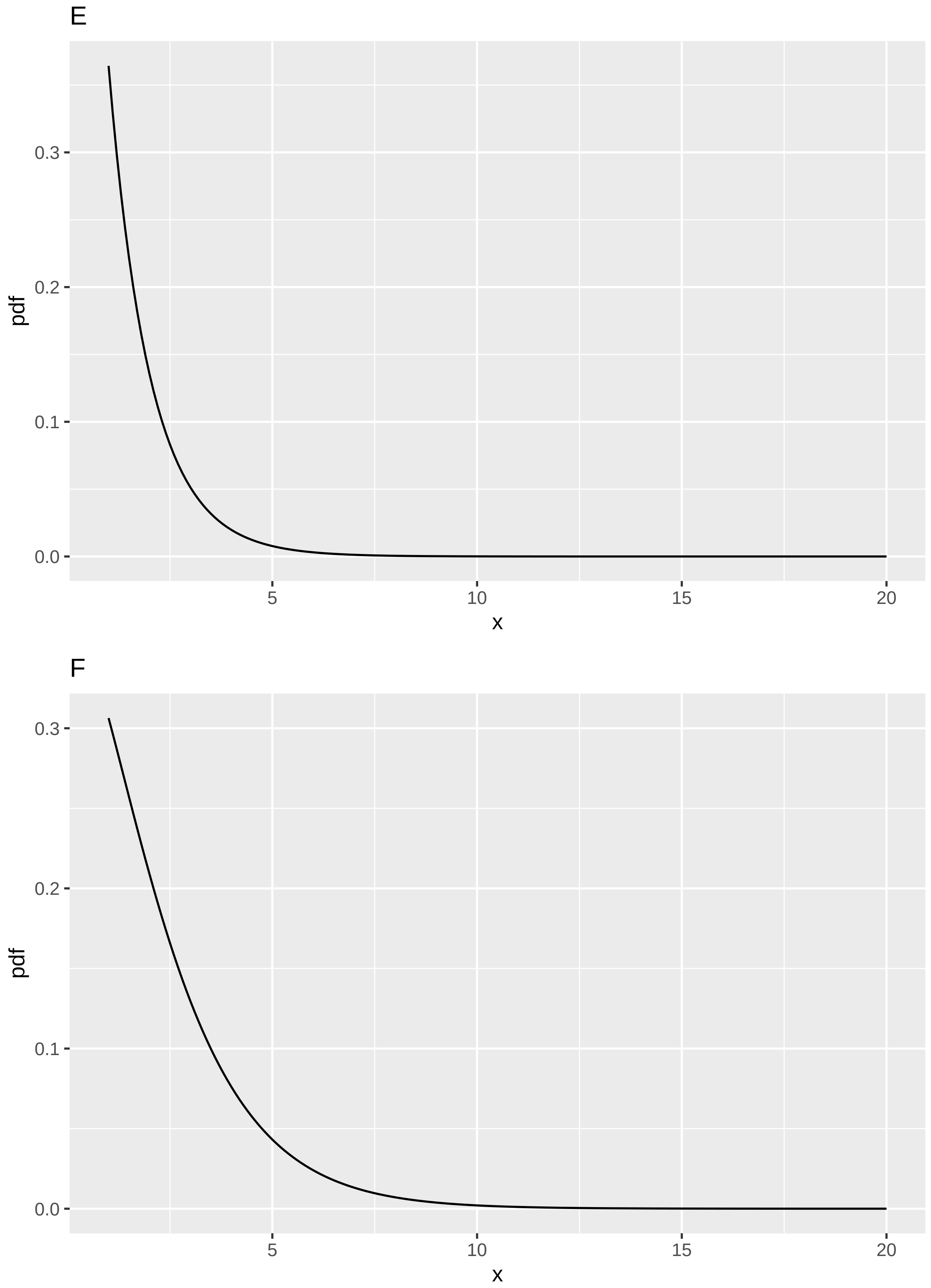
10.5.4 Fig. D

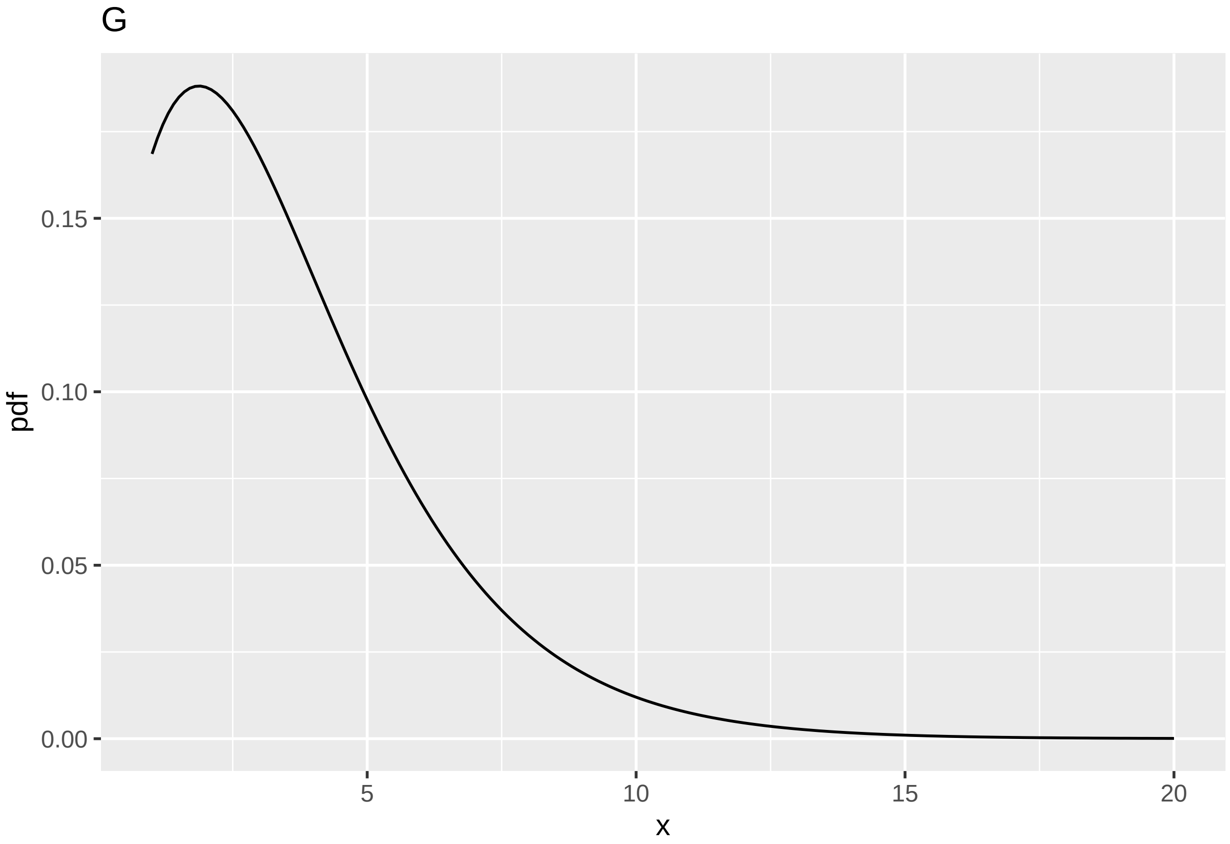
- This corresponds to `ncp = 10`, `ndf` = 2 and `ddf` = 10.
- Now `prob > fCrit` is 0.6769776.
- Power has increased compared to Fig. C.
- The effect of the shift is most obvious in Fig. C and Fig. D.
- Considering a vertical line at `x` = 4.102821, fraction 0.6769776 of the probability distribution in Fig. D lies to the right of this line
- Therefore the NH is likely to be rejected with probability 0.6769776.

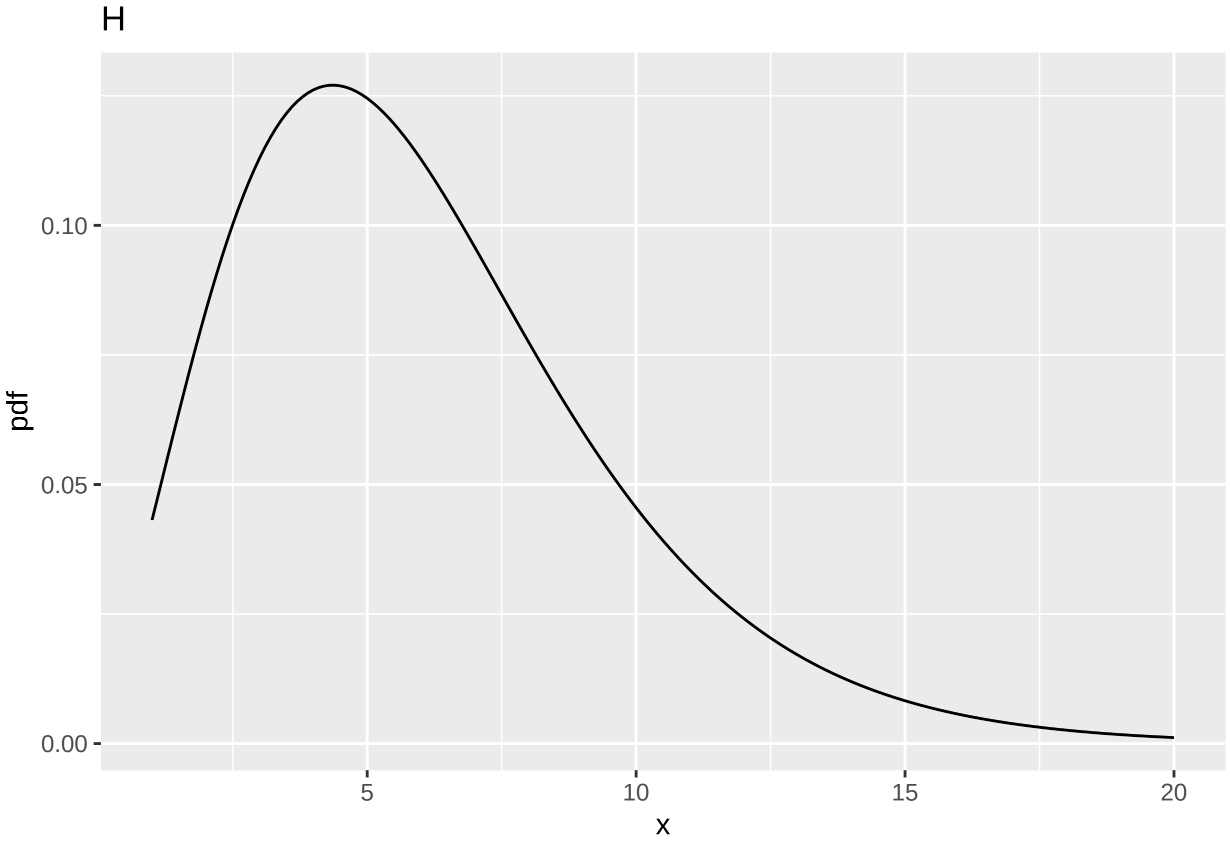
10.5.5 Summary

The larger that non-centrality parameter, the greater the shift to the right of the F-distribution, and the greater the statistical power.

10.6 Effect of `ncp` for `ndf` = 2 and `ddf` = 100




G
pdf
0.15
0.10
0.05
0.00
5
10
15
20
x


H
pdf
0.10
0.05
0.00
5
10
15
20
x

	ndf	ddf	fCrit	ncp	pFgtFCrit
A	2	10	4.102821	0	0.0500000
B	2	10	4.102821	2	0.1775840
C	2	10	4.102821	5	0.3876841
D	2	10	4.102821	10	0.6769776
E	2	100	3.087296	0	0.0500000
F	2	100	3.087296	2	0.2199264
G	2	100	3.087296	5	0.4910802
H	2	100	3.087296	10	0.8029764

10.7 Comments

- All comparisons in this sections are at the same values of `ncp` defined above.
- And between `ddf` = 100 and `ddf` = 10.

10.7.1 Fig. E

- This corresponds to `ncp` = 0, `ndf` = 2 and `ddf` = 100.
- The critical value is `fCrit_2_100` = 3.0872959. Notice the decrease compared to the previous value for `ncp` = 0, i.e., 4.102821, for `ddf` = 10.
- One expects that increasing `ddf` will make it more likely that the NH will be rejected, and this is confirmed below.
- All else equal, statistical power increases with increasing `ddf`.

10.7.2 Fig. F

- This corresponds to `ncp` = 2, `ndf` = 2 and `ddf` = 100.
- The probability of exceeding the critical value is `prob > fCrit_2_100` = 0.2199264, greater than the previous value, i.e., 0.177584 for `ddf` = 10.

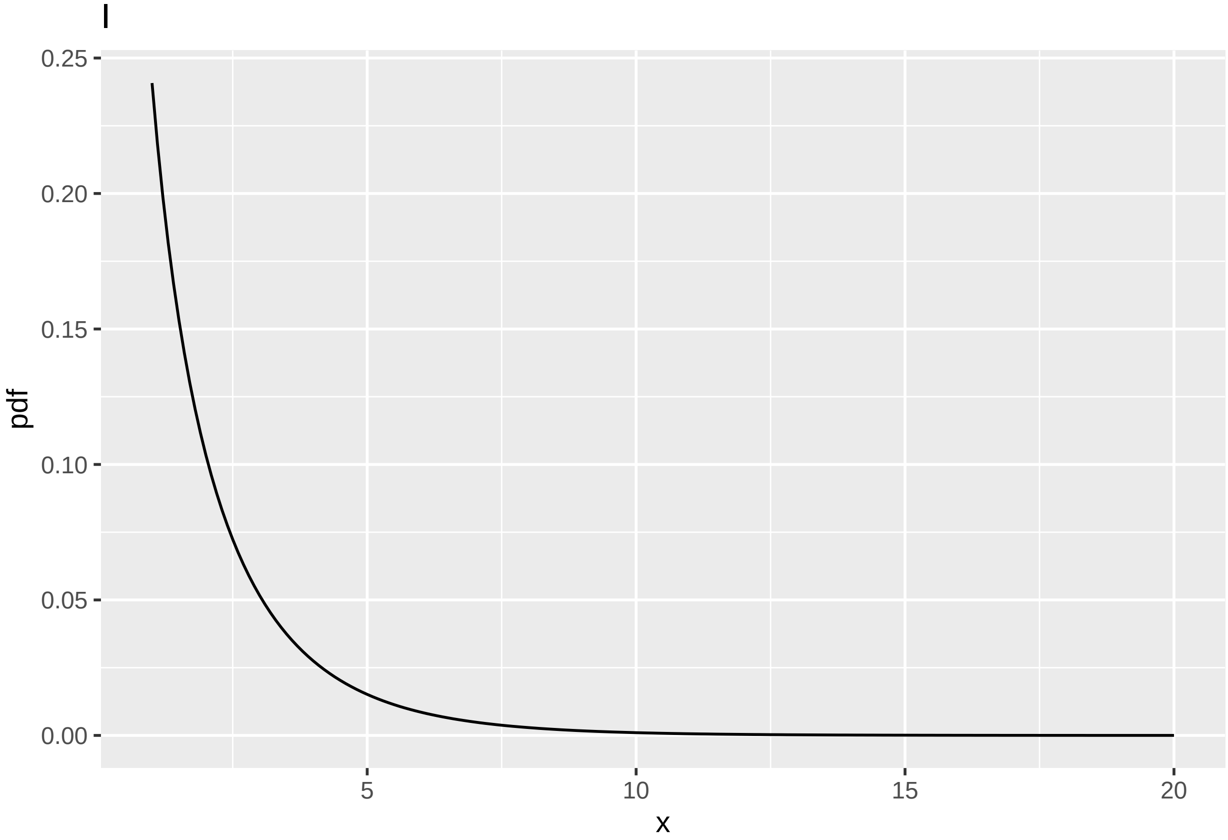
10.7.3 Fig. G

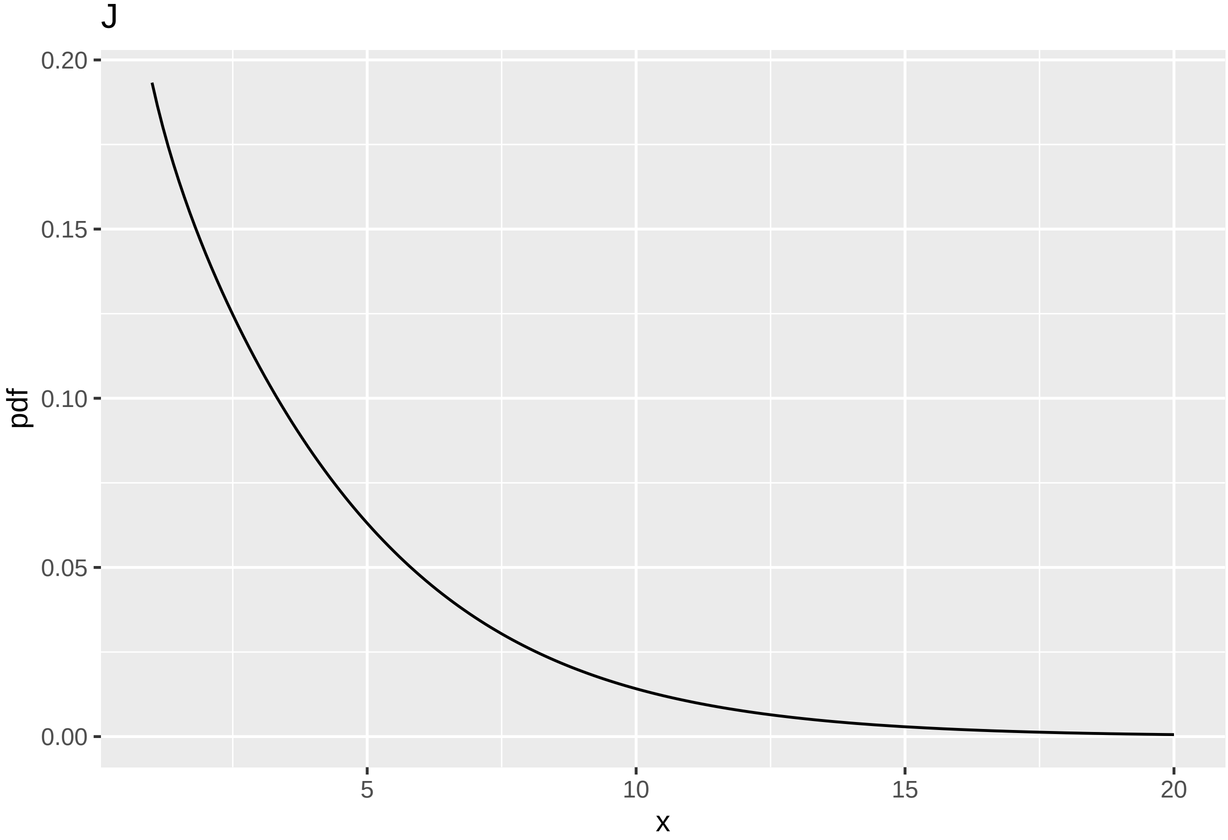
- This corresponds to `ncp = 5`, `ndf` = 2 and `ddf` = 100.
- The probability of exceeding the critical value is `prob > fCrit_2_100` = 0.4910802.
- This is greater than the previous value, i.e., 0.3876841 for `ddf` = 10.

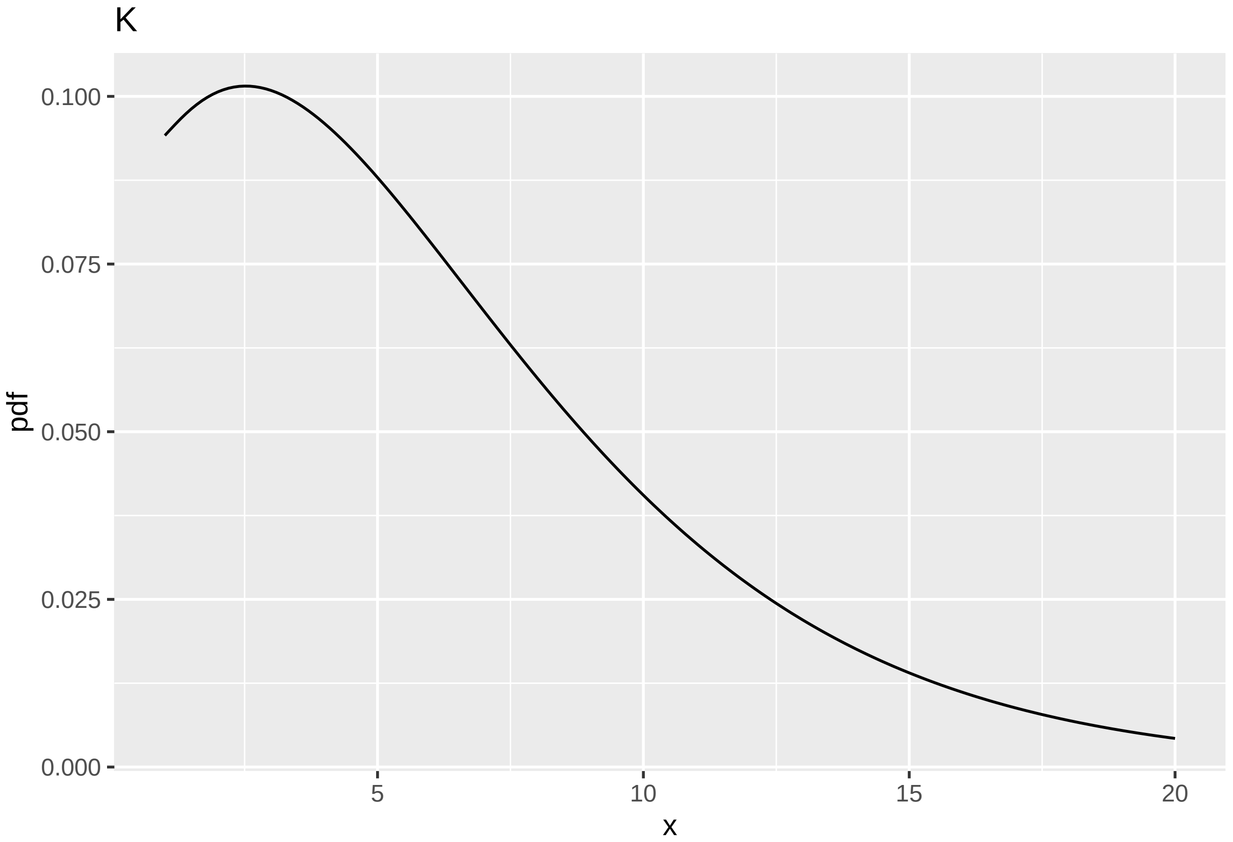
10.7.4 Fig. H

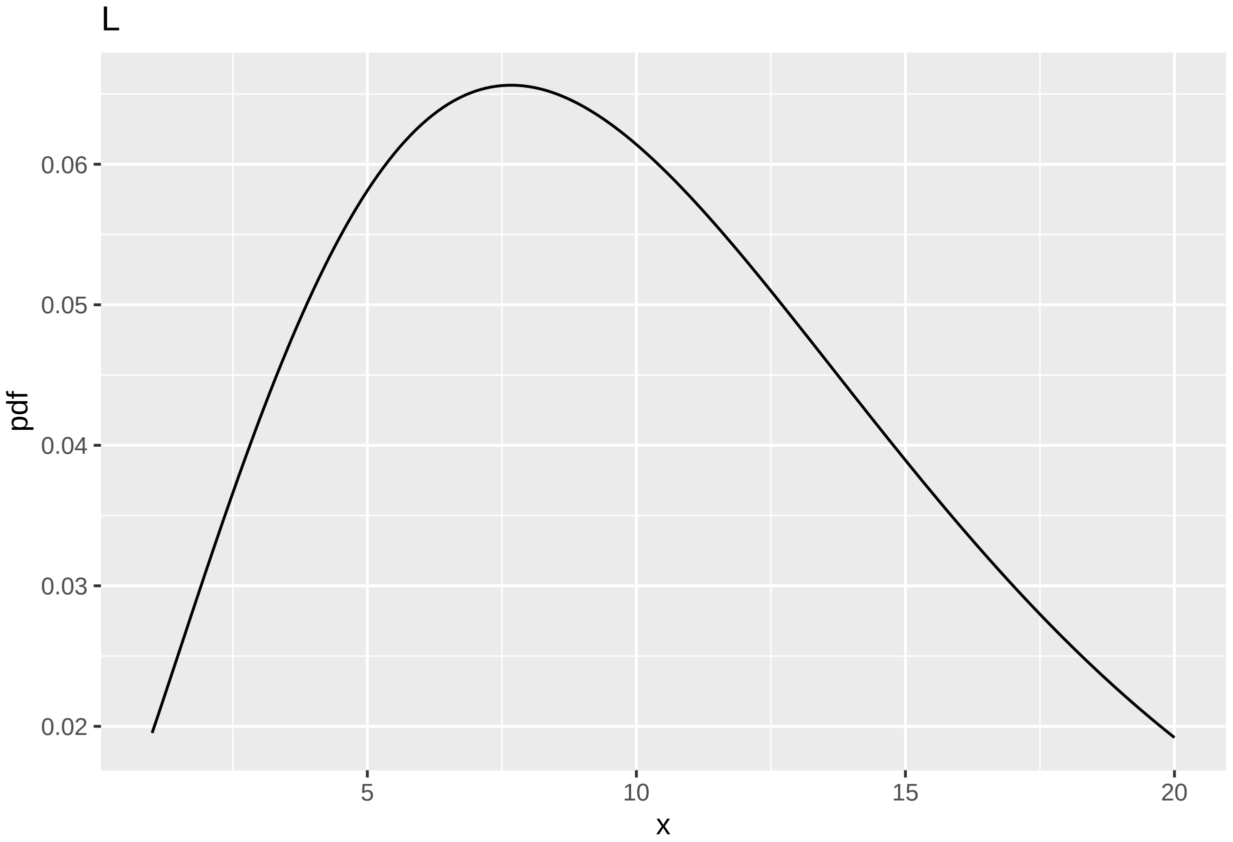
- This corresponds to `ncp = 10`, `ndf` = 2 and `ddf` = 100.
- The probability of exceeding the critical value is `prob > fCrit_2_100` is 0.8029764.
- This is greater than the previous value, i.e., 0.6769776 for `ddf` = 10.

10.8 Effect of `ncp` for `ndf` = 1, `ddf` = 100






K
0.100
0.075
0.050
0.025
0.000
pdf
5
10
15
20
x


L
0.06
0.05
0.04
0.03
0.02
pdf
5
10
15
20
x

	ndf	ddf	fCrit	ncp	pFgtFCrit
A	2	10	4.102821	0	0.0500000
B	2	10	4.102821	2	0.1775840
C	2	10	4.102821	5	0.3876841
D	2	10	4.102821	10	0.6769776
E	2	100	3.087296	0	0.0500000
F	2	100	3.087296	2	0.2199264
G	2	100	3.087296	5	0.4910802
H	2	100	3.087296	10	0.8029764
I	1	100	3.936143	0	0.0500000
J	1	100	3.936143	2	0.2883607
K	1	100	3.936143	5	0.6004962
L	1	100	3.936143	10	0.8793619

10.9 Comments

- All comparisons in this sections are at the same values of `ncp` defined above and at `ddf` = 100.
- And between `ndf` = 1 and `ndf` = 2.

10.9.1 Fig. I

- This corresponds to `ncp` = 0, `ndf` = 1 and `ddf` = 100.
- The critical value is `fCrit_1_100` = 3.936143.
- Notice the increase in the critical value as compared to the corresponding value for `ndf = 2`, i.e., 3.0872959.
- One expects power to decrease: the following code demonstrates that as `ndf` increases, the critical value `fCrit` decreases.
- In significance testing generally `ndf` = I -1.
- It more likely that the NH will be rejected with increasing numbers of treatments.

ndf	ddf	fCrit
1	100	3.936143
2	100	3.087296
5	100	2.305318
10	100	1.926692
12	100	1.850255
15	100	1.767530
20	100	1.676434

10.9.2 Fig. J

- This corresponds to `ncp` = 2, `ndf` = 1 and `ddf` = 100.
- Now `prob > fCrit_1_100` = 0.2883607, 0.1351602, 0.0168844, 8.9992114×10^{-4}, 3.2584757×10^{-4}, 8.1619807×10^{-5}, 1.1084132×10^{-5}, larger than the previous value 0.2199264.
- The power has actually increased.

10.9.3 Fig. K

- This corresponds to `ncp` = 5, `ndf` = 1 and `ddf` = 100‘’,
- Now `prob > fCrit_1_100` = 0.6004962, 0.3632847, 0.0699798, 0.0048836, 0.0018367, 4.6889533×10^{-4}, 6.2058692×10^{-5}, larger than the previous value 0.4910802.
- Again, the power has actually increased.

10.9.4 Fig. L

- This corresponds to `ncp` $= 10$, `ndf` $= 1$ and `ddf` $= 100$
- Now `prob > fCrit_1_100` is 0.8793619, 0.7000168, 0.2459501, 0.0290856, 0.0123033, 0.0035298, 5.1213398×10^{-4}, larger than the previous value 0.8029764.
- The power has actually increased.

10.10 Summary

- Power increases with increasing `ddf` and `ncp`.
- The effect of increasing `ncp` is quite dramatic. This is because power depends on the square of `ncp`.
- As `ndf` increases, `fCrit` decreases, which makes it more likely that the NH will be rejected.
- With increasing numbers of treatments the probability is greater that the F-statistic will be large enough to exceed the critical value.

Chapter 11

RSM operating characteristics

11.1 TBA How much finished

10%

11.2 Introduction

- The purpose of this vignette is to explain the operating characteristics predicted by the RSM. It relates to Chapter 17 in my book (Chakraborty, 2017).
- This vignette is under development …
- Also to explain the difference between `dataset` members (`lesionID`, `lesionWeight`) and RSM parameters (`lesDistr`, `lesWghtDistr`).

11.3 The distinction between predicted curves and empirical curves

- Operating characteristics predicted by a model have zero sampling variability.

- Empirical operating characteristics, which apply to datasets, have non-zero sampling variability.
- If the model is correct, as the numbers of cases in the dataset increases, the empirical operating characteristic asymptotically approaches the predicted curve.

11.4 The RSM model

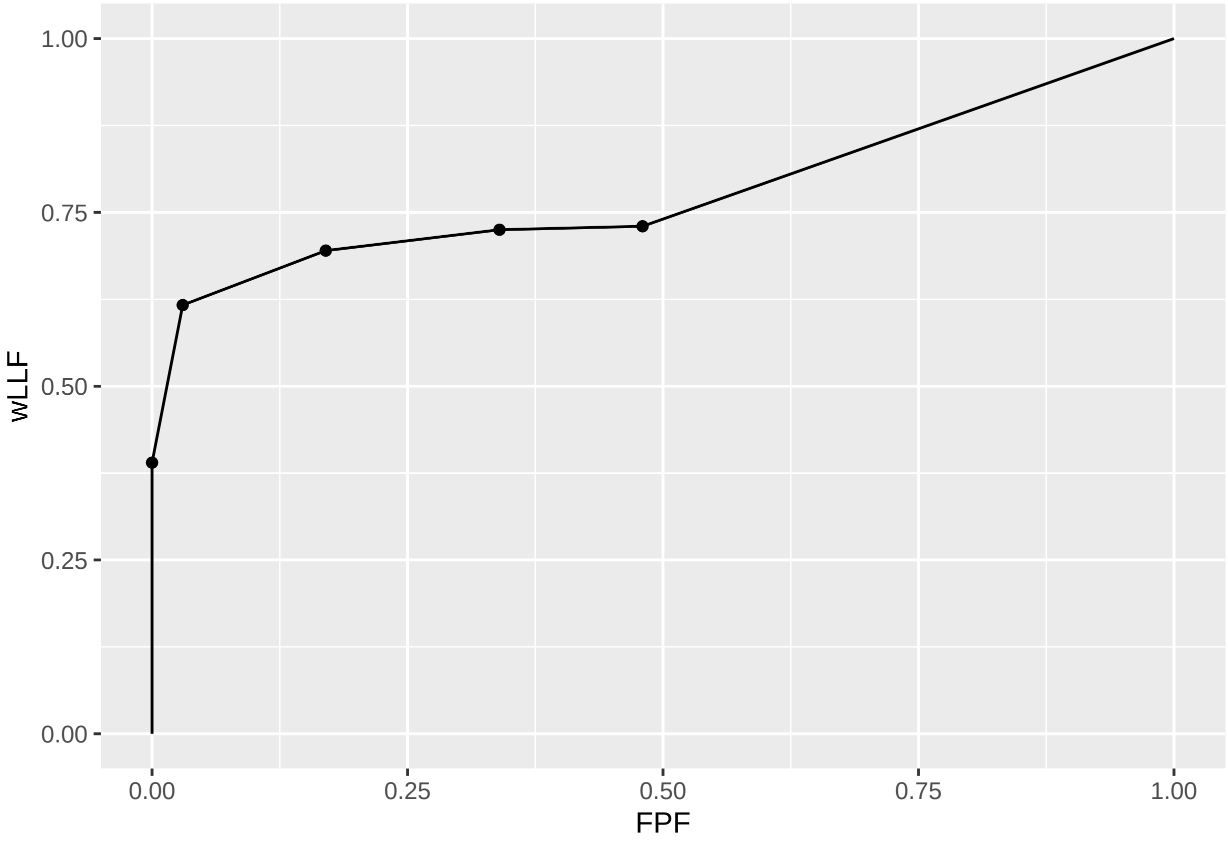
- The 3 RSM parameters and two additional parameters characterizing the dataset determine the wAFROC curve.
- The 3 RSM parameters are μ, λ and ν.
- The two dataset parameters are:
  - The distribution of number of lesions per diseased case, `lesDistr`.
  - The distribution of lesion weights, `lesWghtDistr`.
- These parameters do not apply to individual cases; rather they refer to a large population (asymptotically infinite in size) of cases.

```
str(dataset04$lesions$IDs)
#>  num [1:100, 1:3] 1 1 1 1 1 1 1 1 1 1 ...
str(dataset04$lesions$weights)
#>  num [1:100, 1:3] 1 1 1 1 1 1 1 1 1 1 ...
```

- Note that the first index of both arrays is the case index for the 100 abnormal cases in this dataset.
- With finite number of cases the empirical operating characteristic (or for that matter any fitted operating characteristic) will have sampling variability as in the following example.

11.5 The empirical wAFROC

```
p <- PlotEmpiricalOperatingCharacteristics(dataset04, opChType = "wAFROC")
p$Plot
```

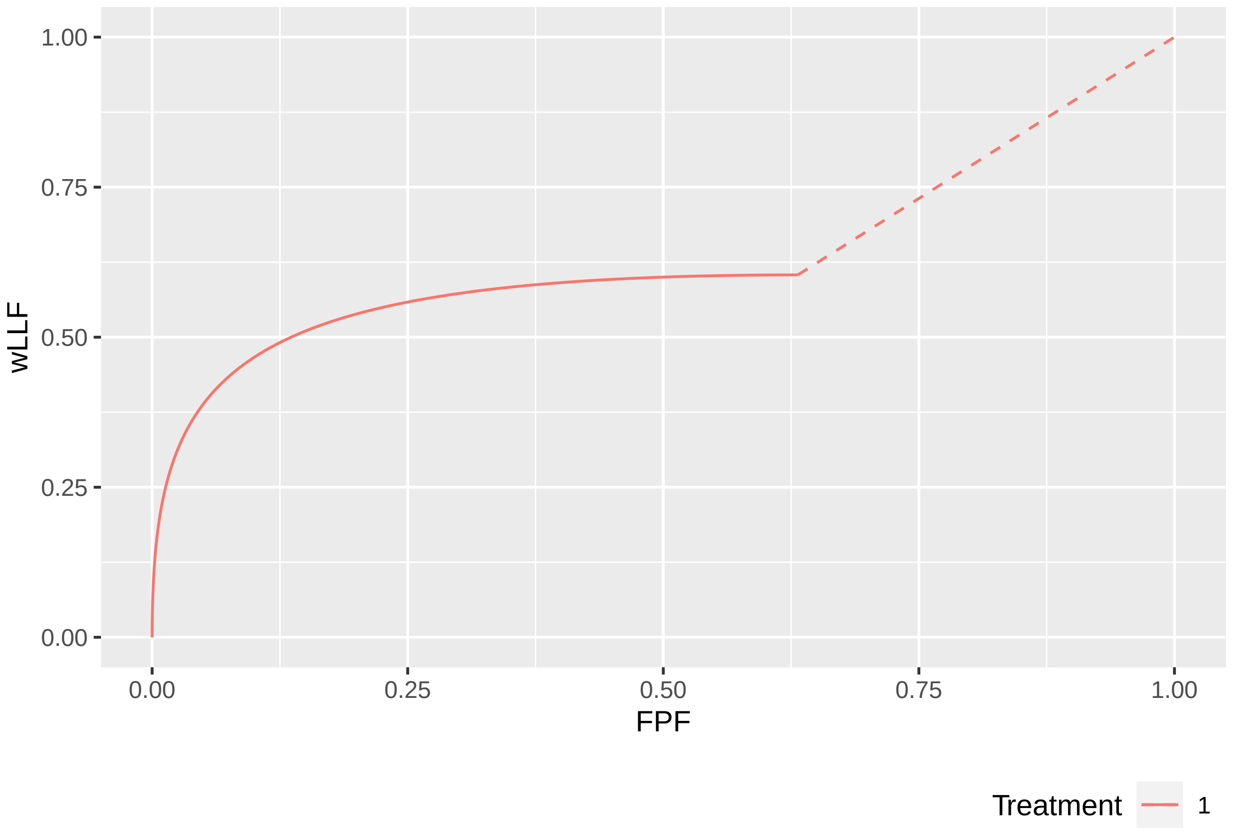


- The piecewise linear nature of the plot, with sharp breaks, indicates that this is due to a finite dataset.
- In contrast the following code shows a smooth plot, because it is a model *predicted* plot.

11.6 The predicted wAFROC

```
## Following example is for mu = 2, lambda = 1, nu = 0.6. 20% of the diseased
## cases have a single lesion, 40% have two lesions, 10% have 3 lesions,
## and 30% have 4 lesions.
lesDistr <- c(0.2, 0.4, 0.1, 0.3)

## On cases with one lesion the weights are 1, on cases with 2 lesions the weights
## are 0.4 and 0.6, on cases with three lesions the weights are 0.2, 0.3 and 0.5, and
## on cases with 4 lesions the weights are 0.3, 0.4, 0.2 and 0.1:
```

```
relWeights <- c(0.3,  0.4, 0.2,  0.1)
p <- PlotRsmOperatingCharacteristics(
  mu = 2,
  lambda = 1,
  nu = 0.6,
  OpChType = "wAFROC",
  lesDistr = lesDistr,
  relWeights = relWeights,
  legendPosition = "bottom", nlfRange = c(0, 1), llfRange = c(0, 1))
p$wAFROCPlot
```



11.7 The distribution of number of lesions and weights

```
lesDistr
#> [1] 0.2 0.4 0.1 0.3
relWeights
#> [1] 0.3 0.4 0.2 0.1
```

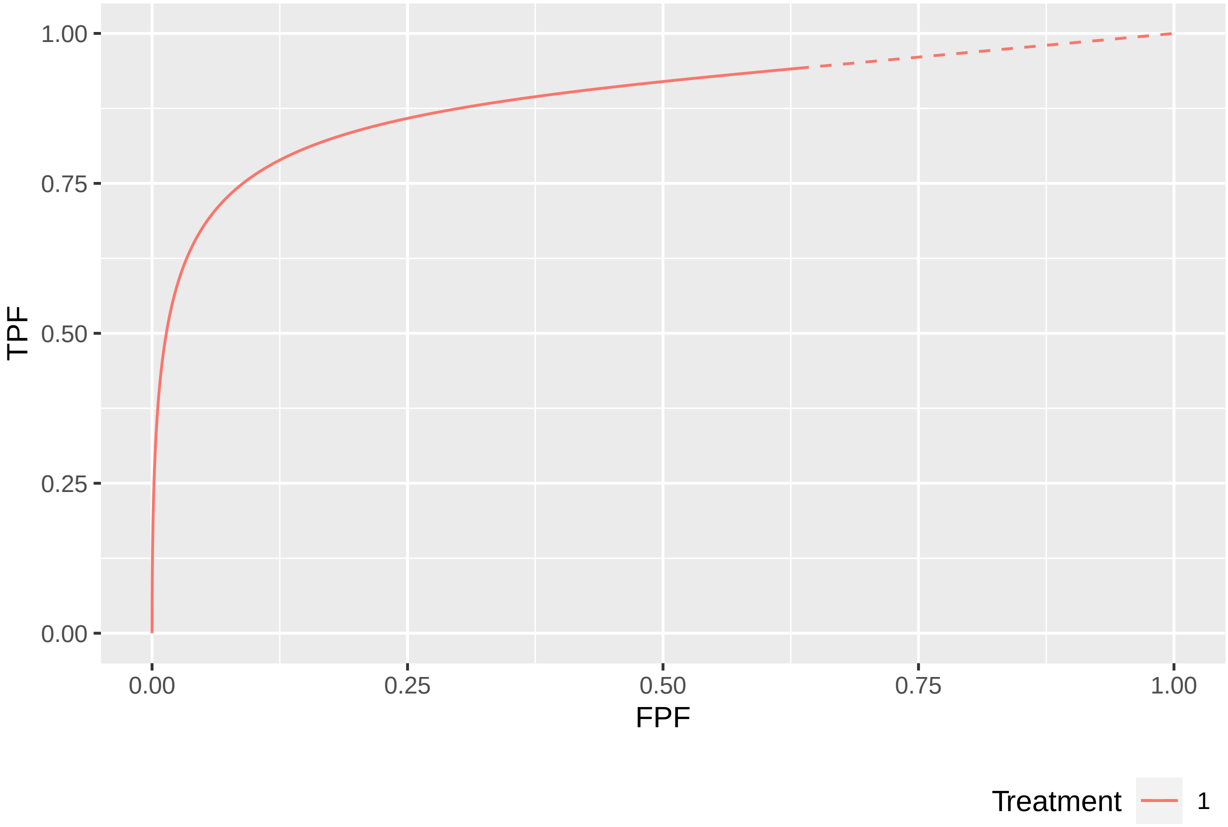
- The `lesDistr` array 0.2, 0.4, 0.1, 0.3 specifies the fraction of diseased cases with the number of lesions corresponding to the column index. To specify a dataset with exactly 3 lesions per diseased case use `lesDist = c(0, 0, 1, 0)`.
- The `relWeights` array 0.3, 0.4, 0.2, 0.1 specifies the relative weights.
- For cases with 1 lesion, the weight is 1.
- For cases with 2 lesions, the first lesion has weight 0.4285714 and the second lesion has weight 0.5714286, which are in the ratio 0.3 : 0.4 and sum to unity.

- For cases with 3 lesions, the first lesion has weight 0.3333333, the second lesion has weight 0.4444444 and the third lesion has weight 0.2222222, which are in the ratio 0.3 : 0.4 : 0.2, and sum to unity.
- For cases with 4 lesions, the weights are 0.3, 0.4, 0.2 and 0.1, which are in the ratio 0.3 : 0.4 : 0.2 : 0.1 and sum to unity.

11.8 Other operating characteristics

- By changing `OpChType` one can generate other operating characteristics.
- Note that lesiion weights argument is not needed for ROC curves. It is only needed for `wAFROC` and `wAFROC1` curves.

```
lesDistr <- c(0.2, 0.4, 0.1, 0.3)
p <- PlotRsmOperatingCharacteristics(
  mu = 2,
  lambda = 1,
  nu = 0.6,
  OpChType = "ROC",
  lesDistr = lesDistr,
  legendPosition = "bottom")
p$ROCPlot
```



11.9 Summary

Chapter 12

Improper ROC curves

12.1 TBA How much finished

10%

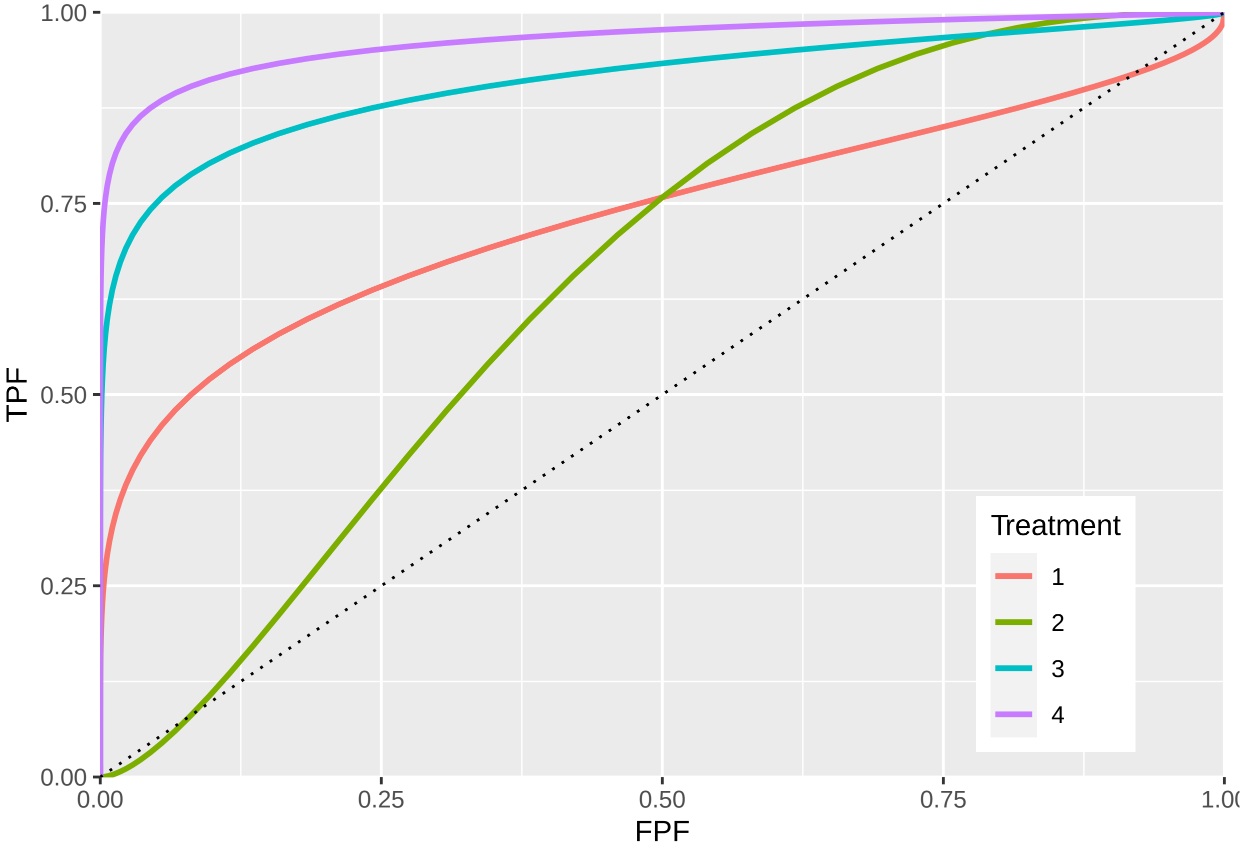
12.2 The binormal model

The binormal model has two parameters, `a` and `b`. The signal (or diseased cases) distribution has unit standard deviation. The noise (or non-diseased cases) distribution has standard deviation `b`. The `a` parameter is the separation of the two distributions.

12.3 Improper ROCs

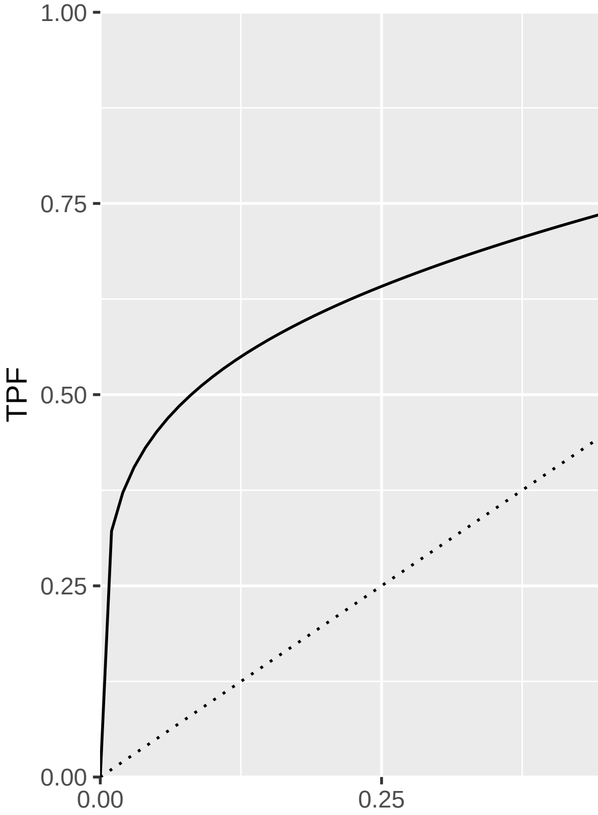
Binormal model fits invariably lead to ROC curves that inappropriately cross the chance diagonal, leading to a prediction of a region of the ROC curve where performance is worse than chance, even for expert observers. By convention, such curves are termed *improper*. This vignette illustrates improper ROCs predicted by the binormal model.

```
  aArray <- c(0.7, 0.7, 1.5, 2)
  bArray <- c(0.5, 1.5, 0.5, 0.5)
  chance_diag <- data.frame(x = c(0,1), y = c(0,1))
  p <- PlotBinormalFit(aArray, bArray) +
    scale_x_continuous(expand = c(0, 0)) +
    scale_y_continuous(expand = c(0, 0)) +
    theme(legend.position = c(0.85, 0.2))
p <- p + geom_line(data = chance_diag, aes(x = x, y = y), linetype="dotted")
print(p)
```

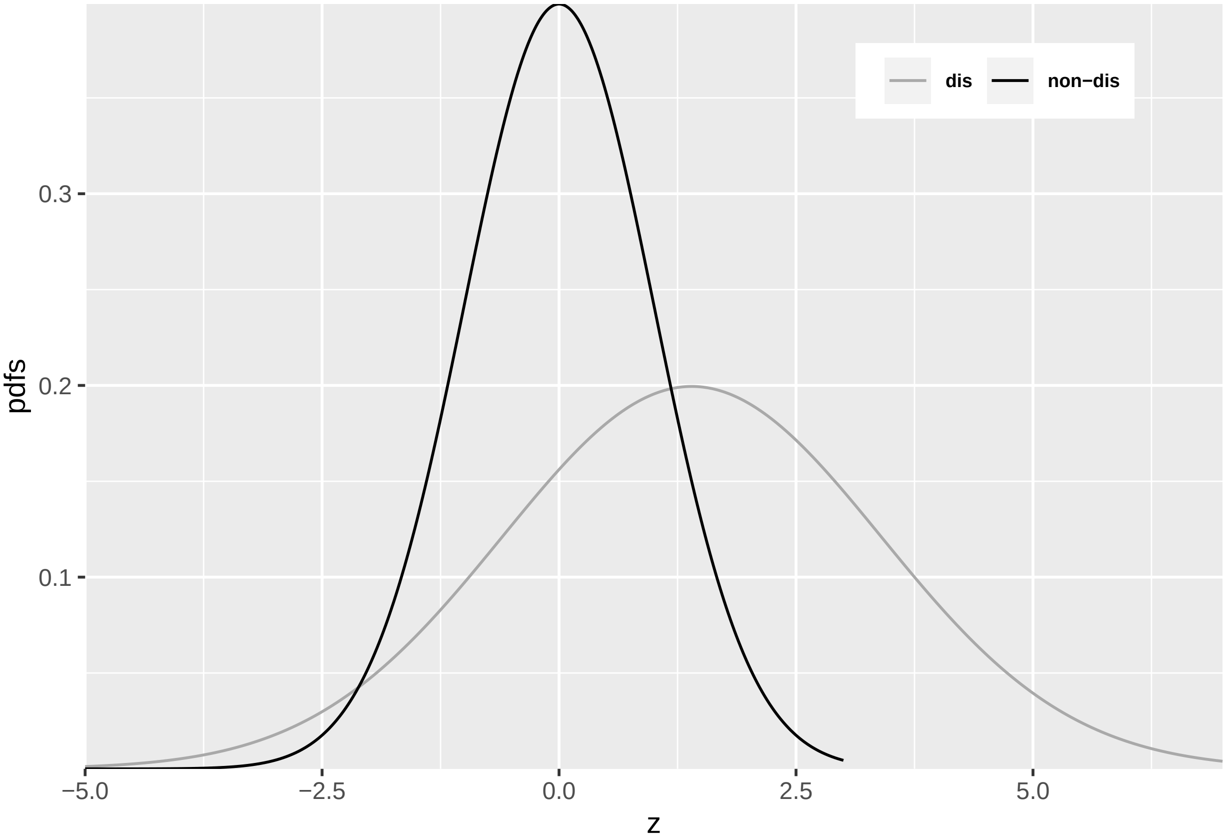


The red plot is the clearest example of an improper ROC. This type of curve occurs whenever `b < 1`. The chance line crossing near the upper right corner, around (0.919,0.919), and the fact that the ROC curve must eventually reach (1, 1) implies the curve must turn upwards as one approaches (1, 1), thereby displaying a "hook". Whenever `b != 1` the hook is there, regardless of whether it is easily visible or not. If `b < 1` the hook is near the upper right corner. If `b > 1` the hook is near the origin (see green line, corresponding to `b = 1.5`). With increasing `a` the hook is less prominent (blue line corresponding to `a = 1.5, b = 0.5` and purple line corresponding to `a = 2, b = 0.5`). But it is there.

12.4 Reason for improper ROCs



The reason for the "hook"" becomes apparent upon examination of the pdfs.



```
#> a =  0.7 , b =  0.5
```

Since `b < 1`, the diseased *pdf* is broader and has a lower peak (since the integral under each distribution is unity) than the non-diseased pdf. Sliding an imaginary threshold to the left, starting from the extreme right, one sees that initially, just below `z = 7`, the diseased distribution starts being *"picked up"* while the non-diseased distribution is not *"picked up"*, causing the ROC to start with infinite slope near the origin (because TPF is increasing while FPF is not). Around `z = 2.5` the non-diseased distribution starts being *"picked up"*, causing the ROC slope to decrease. Around `z = -3`, almost all of the non-diseased distribution has been *"picked up"* which means FPF is near unity, but since not all of the broader diseased distribution has been *"picked up"*, TPF is less than unity. Here is a region where `TPF < FPF`, meaning *the operating point is below the chance diagonal.* As the threshold is lowered further, TPF continues to increase, as the rest of the diseased distribution is *"picked up"* while FPF stays almost constant at unity. In this region, the ROC curve is approaching the upper right corner with almost infinite slope (because TPF is increasing but FPF is not).

Chapter 13

Degenerate ROC curves

13.1 Two helper functions

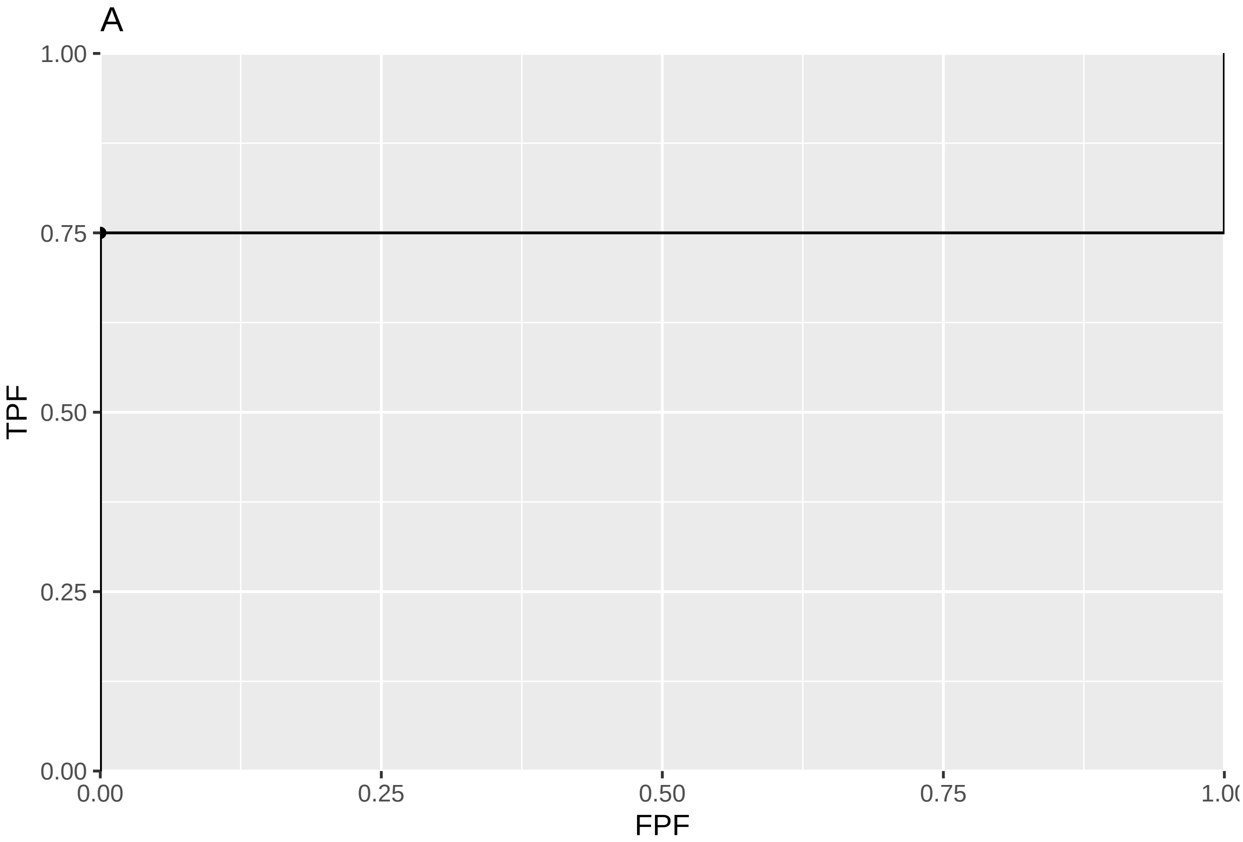
13.2 Degenerate datasets

Metz defined binormal degenerate data sets as those that result in exact-fit binormal ROC curves of inappropriate shape consisting of a series of horizontal and/or vertical line segments in which the ROC "curve" crosses the chance line. The crossing of the chance line occurs because the degenerate data sets can be fitted exactly by infinite or zero values for the model slope parameter `b`, and infinite values for the decision thresholds, or both.

13.3 Understanding degenerate datasets

To understand this, consider that the non-diseased distribution is a Dirac delta function centered at zero (by definition such a function integrates to unity) and the unit variance diseased distribution is centered at 0.6744898. In other words this binormal model is characterized by `a = 0.6744898` and `b = 0`. What is the expected ROC curve? As the threshold ζ is moved from the far right, gradually to the left, TPF will increase but FPF is stuck at zero until the threshold reaches zero. Just before reaching this point, the coordinates of the ROC operating point are (0, 0.75). The 0.75 is due to the fact that `z = 0` is -0.6744898 units relative to the center of the diseased distribution, so the area under the diseased distribution below `z = 0` is 0.25. Since `pnorm` is the probability *below* the threshold, TPF must be its complement, namely 0.75. This explains the operating point (0,0.75), which lies on the y-axis. As the threshold crosses the zero-width delta function, FPF shoots up from 0 to 1, but TPF stays constant. Therefore, the operating point has jumped from (0, 0.75) to (1, 0.75). When the threshold is reduced further, the operating point moves up vertically, along the right side of the ROC plot, until the threshold is so small that virtually all of diseased distribution exceeds it and the operating point reaches (1, 1). The ROC curve is illustrated in plot A.

```
plotOP <- data.frame(FPF = 0, TPF = 0.75)
a <- 0.6744898; b <- 0
plotCurve <- BMPoints(a, b)
figA <- ggplot(mapping = aes(x = FPF, y = TPF)) +
  geom_line(data = plotCurve) +
  geom_point(data = plotOP)  +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0)) +
  ggtitle("A")
print(figA)
```

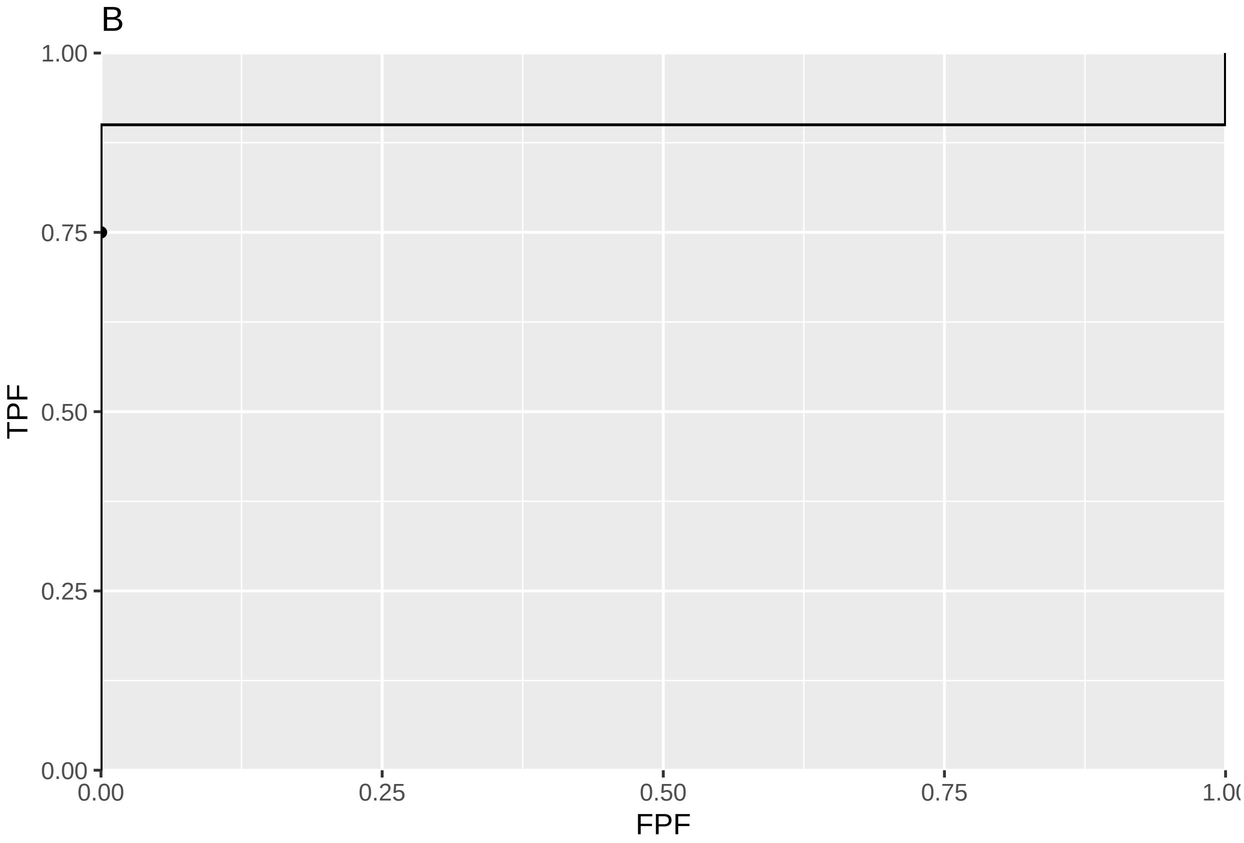


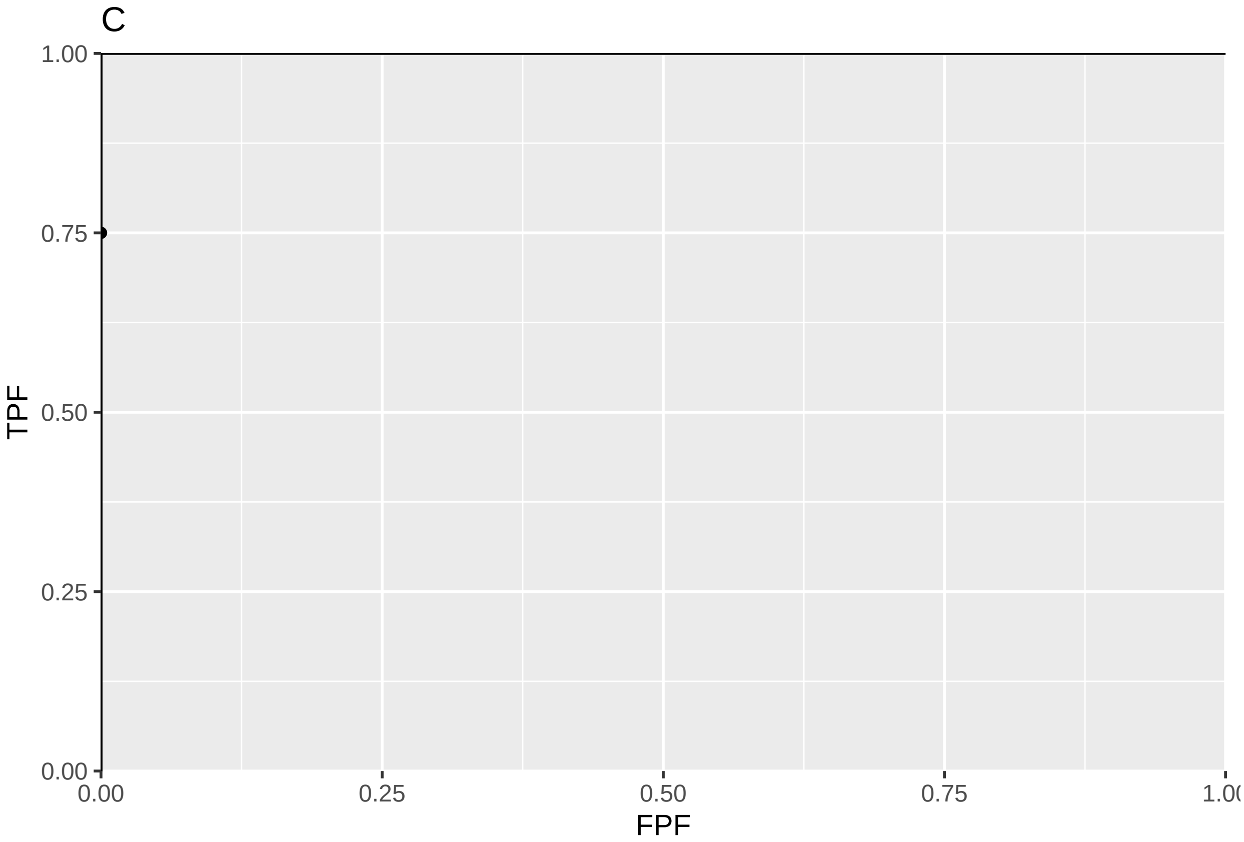
This is an extreme example of an ROC curve with a “hook”. If the data is such that the only operating point provided by the observer is (0,0.75) then this curve will be an exact fit to the operating point.

13.4 The exact fit is not unique

Actually, given one operating point (0, 0.75) the preceding fit is not even unique. If the diseased distribution is shifted appropriately to the right of its previous position, and one can determine the necessary value of a, then the ROC curve will shoot upwards through the operating point (0, 0.75) to (0, 0.9), as in plot B, before proceeding horizontally to (1, 0.9) and then completing the curve to (1, 1). If the diseased distribution is shifted well to the right, i.e., a is very large, then the ROC curve will shoot upwards past the operating point, as in plot C, all the way to (0,1) before proceeding horizontally to (1, 1).

```
a <- 1.281552; b <- 0
plotCurve <- BMPoints(a, b)
figB <- ggplot(mapping = aes(x = FPF, y = TPF)) +
  geom_line(data = plotCurve) +
  geom_point(data = plotOP)  +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0)) +
  ggtitle("B")

a <- Inf; b <- 0
plotCurve <- BMPoints(a, b)
figC <- ggplot(mapping = aes(x = FPF, y = TPF)) +
  geom_line(data = plotCurve) +
  geom_point(data = plotOP)  +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0)) +
  ggtitle("C")
print(figB);print(figC)
```





All of these represent exact fits to the observed operating point, with `b = 0` and different values of `a`. Not one of them is reasonable.

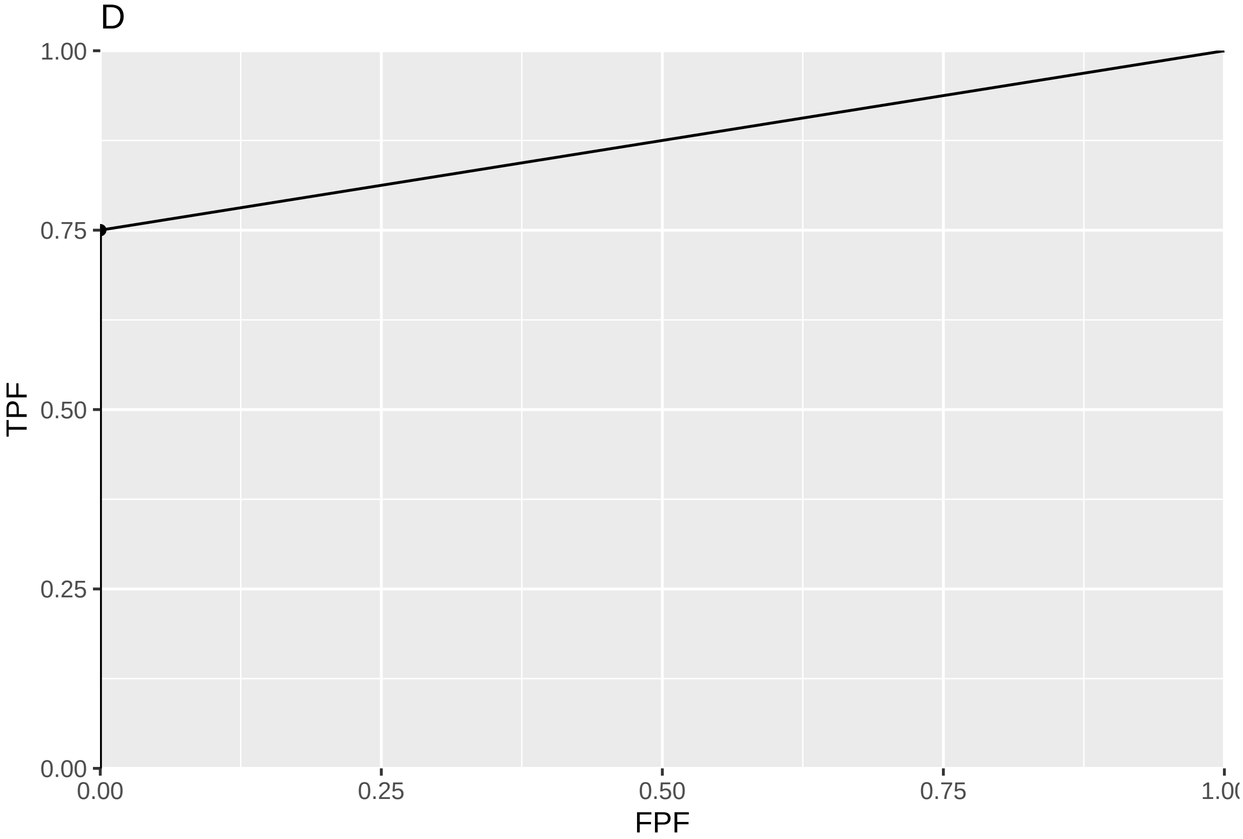
13.5 Comments on degeneracy

Degeneracy occurs if the observer does not provide any interior operating points. So why worry about it? Perhaps one has a non-cooperating observer, who is not heeding the instructions to *spread the ratings, use all the bins*. A simple example shows that the observer could if fact be cooperating fully and is still unable to provide any interior data points. Consider 100 diseased cases consisting of 75 easy cases and 25 difficult ones and 100 easy non-diseased cases. The observer is expected to rate the 75 easy diseased cases as *fives*, the difficult ones as *ones* and the 100 non-diseased cases are rated *ones*. No amount of coaxing *please, please spread your ratings* is going to convince this observer to rate with twos, threes and fours any of the 75 easy diseased cases. If the cases are obviously diseased, and that is what is meant by *easy cases*, they are supposed to be rated fives: *definitely diseased*. Forcing them to rate some of them as *probably diseased* or *possibly diseased* would be irrational and guilty of bending the reading paradigm to fit the convenience of the researcher (early in his research career, the author used to believe in the existence of non-cooperating observers, so Metz's advice to *spread the ratings* did not seem unreasonable at that time).

13.6 A reasonable fit to the degenerate dataset

If the dataset yields a single operating point (0, 0.75), what is a reasonable ROC plot? There is a theorem that given an observed operating point, the line connecting that point to (1, 1) represents a lower bound on achievable performance by the observer. The observer using a guessing mechanism to classify the remaining cases achieves the lower bound. Here is an explanation of this theorem. Having rated the 75 easy diseased cases as fives, the observer is left with 25 diseased cases and 100 non-diseased cases, all of which appear definitely non-diseased to the observer. Suppose the observer randomly rates 20% of the remaining cases as fours. This would pick up five of the actually diseased cases and 20 non-diseased ones. Therefore, the total number of diseased cases rated four or higher is 80, and the corresponding number of non-diseased cases is 20. The new operating point of the observer is (0.20, 0.80). Now, one has two operating points, the original one on the y-axis at (0, 0.75) and an interior point (0.20, 0.80). Next, instead of randomly rating 20% of the remaining cases as fours, the observer rates 40% of them as fours, then the interior point would have been (0.40, 0.85). The reader can appreciate that simply by increasing the fraction of remaining cases that are randomly rated fours, the observer can move the operating point along the straight line connecting (0, 0.75) and (1, 1), as in plot D. Since a guessing mechanism is being used, this must represent a lower bound on performance. The resulting ROC curve is proper and the net AUC = 0.875.

```
mu <- Inf; alpha <- 0.75
plotCurve <- CBMPoints(mu, alpha)
figD <- ggplot(mapping = aes(x = FPF, y = TPF)) +
  geom_line(data = plotCurve) +
  geom_point(data = plotOP)  +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0)) +
  ggtitle("D")
print(figD)
```



For this dataset this is in fact the fit yielded by the contaminated binormal model (CBM) and the radiological search model (RSM). Why should one select the lowest possible performance consistent with the data? Because it yields a *unique* value for performance: any higher performance would not be unique.

Chapter 14

PROPROC ROC curves

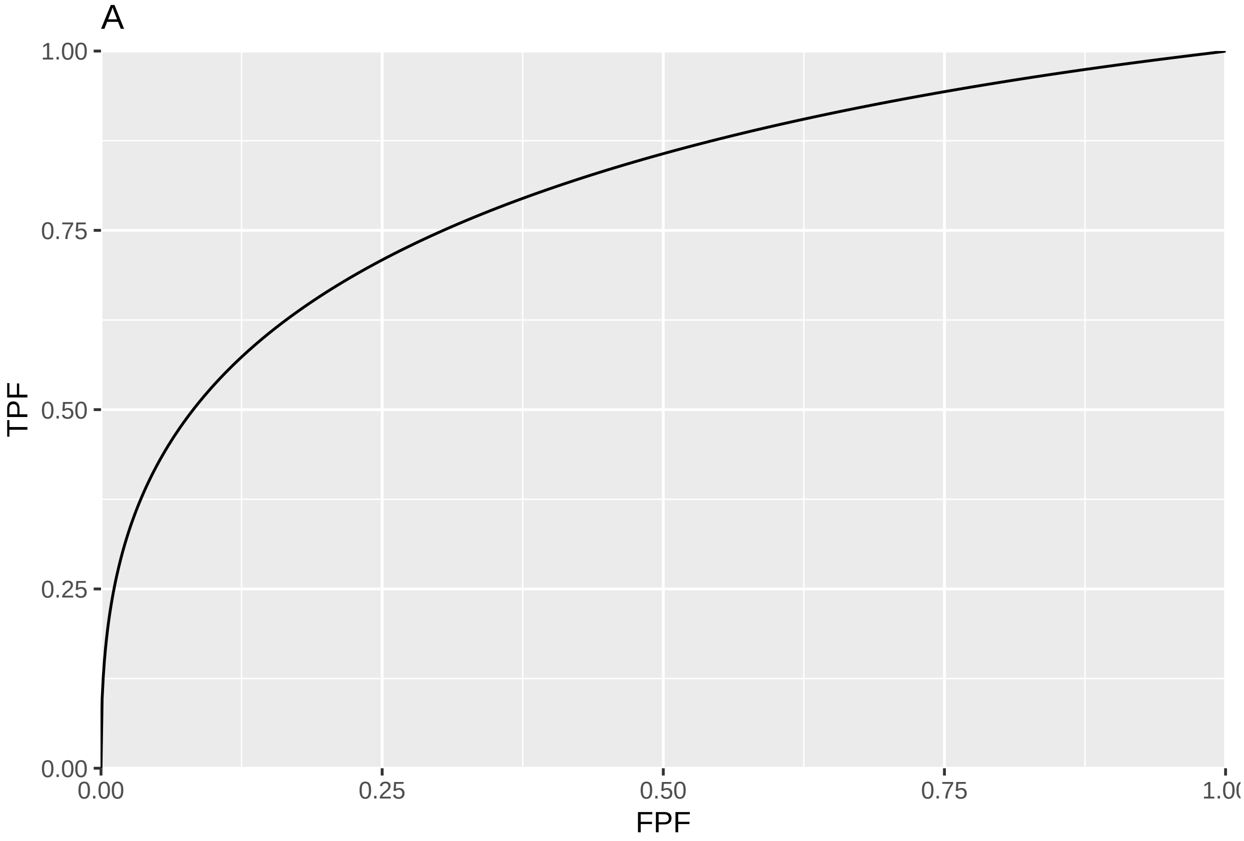
14.1 Helper functions

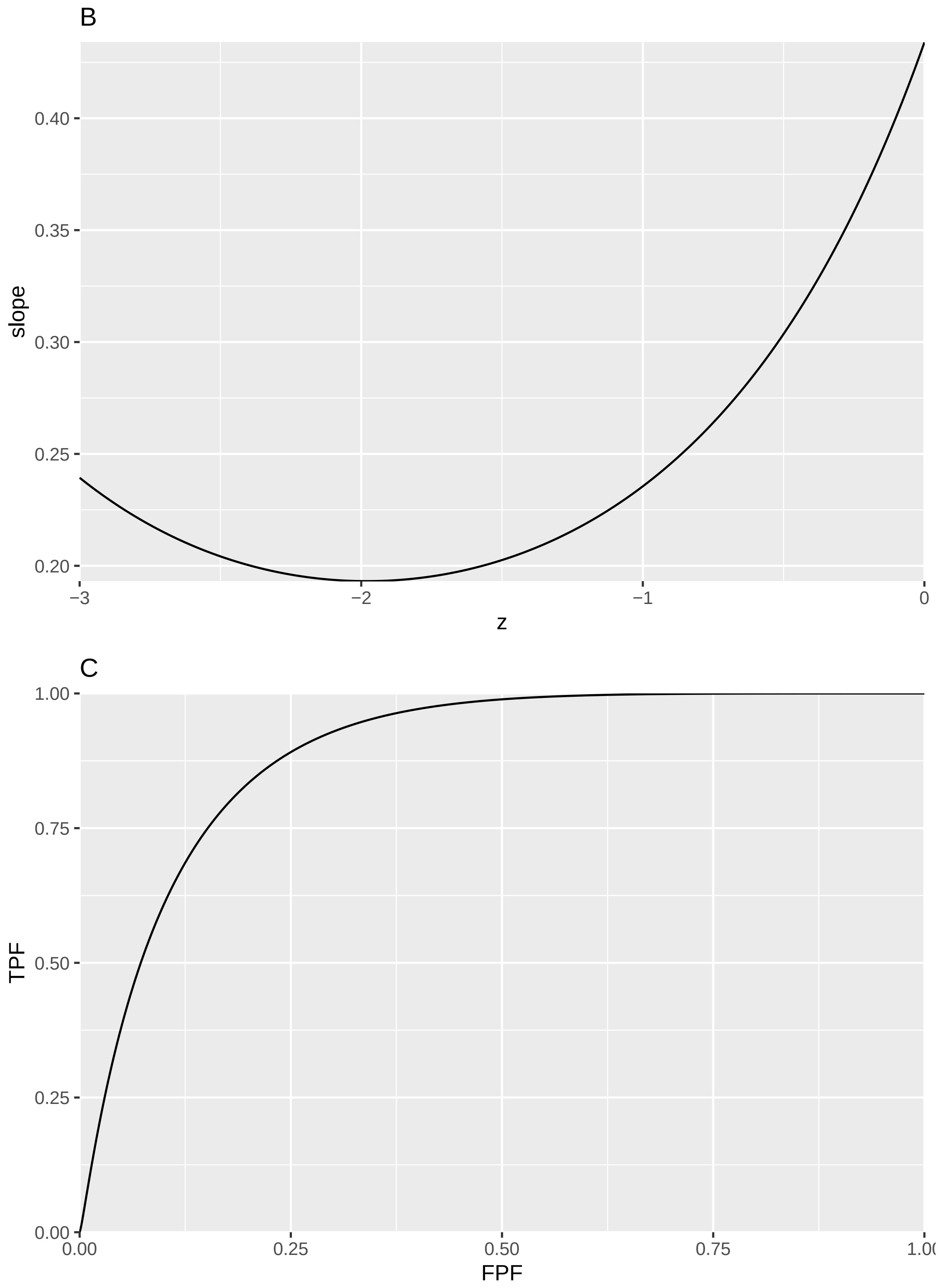
14.2 Definitions of PROPROC parameters in terms of binormal model parameters

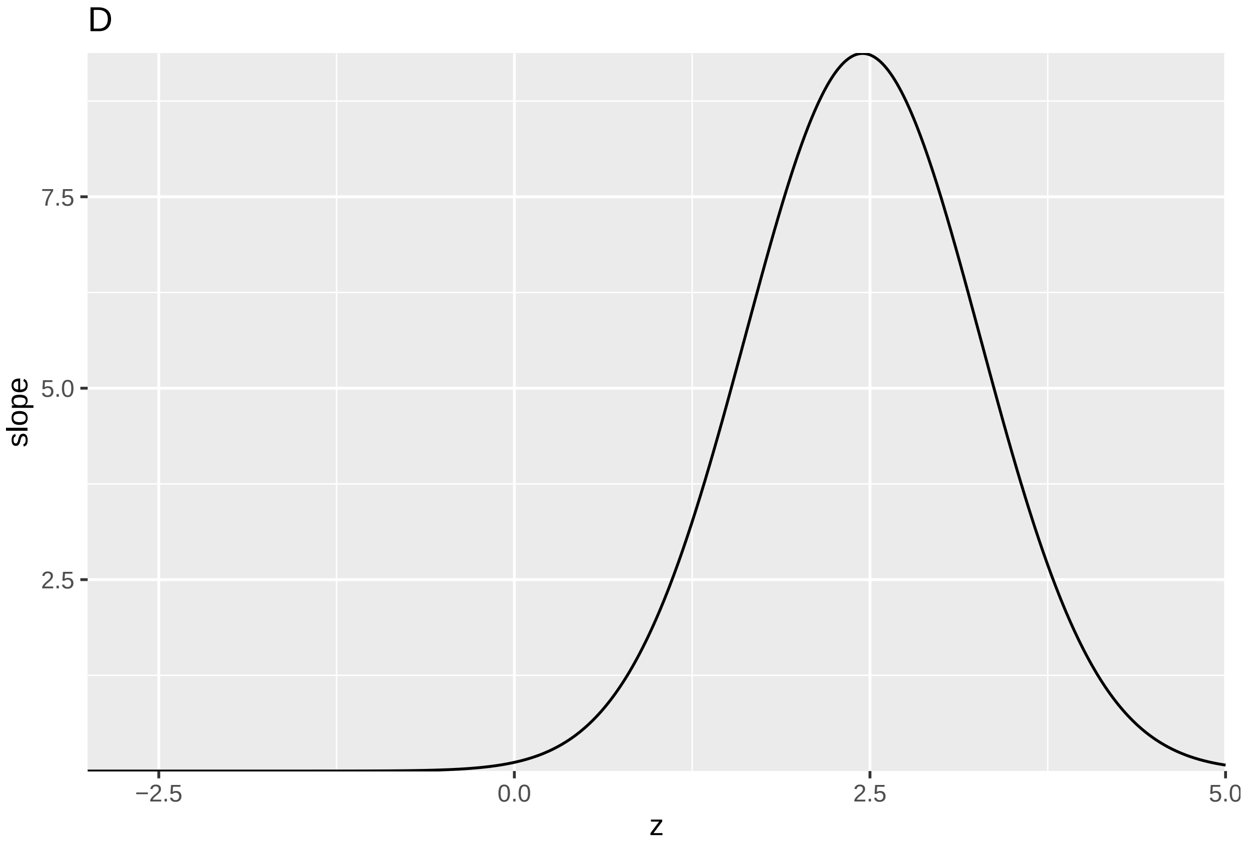
$$c = \frac{b-1}{b+1}; d_a = \frac{\sqrt{2}a}{\sqrt{1+b^2}}$$

14.3 Main code and output

```r
c1Arr <-   c(-0.1322804, 0.2225588); daArr  <-  c(1.197239, 1.740157)
myLabel <- c("A", "B", "C", "D")
myLabelIndx <- 1
for (i in 1:2)
{
  c1 <- c1Arr[i]
  da <- daArr[i]
  ret <- Transform2ab(da, c1)
  a <- ret$a;b <- ret$b
  if (i == 1) z <- seq(-3, 0, by = 0.01) # may need to adjust limits to view detail of slope plot
  if (i == 2) z <- seq(-3, 5, by = 0.01) # may need to adjust limits to view detail of slope plot

  FPF <- seq(0.0, 1, 0.001)
  TPF <- rocY(FPF, a, b)

  rocPlot <- data.frame(FPF = FPF, TPF = TPF)
  plotRoc <- ggplot(rocPlot, aes(x = FPF, y = TPF)) +
    geom_line()  +
   scale_x_continuous(expand = c(0, 0)) +
    scale_y_continuous(expand = c(0, 0))   +
    ggtitle(myLabel[myLabelIndx]);myLabelIndx <- myLabelIndx + 1

  slope <-b*dnorm(a-b*z)/dnorm(-z) # same as likelihood ratio

  slopePlot <- data.frame(z = z, slope = slope)
  p <- ggplot(slopePlot, aes(x = z, y = slope)) +
    geom_line()  +
    scale_x_continuous(expand = c(0, 0)) +
    scale_y_continuous(expand = c(0, 0))  +
    ggtitle(myLabel[myLabelIndx]);myLabelIndx <- myLabelIndx + 1
  print(plotRoc);print(p)
}
```




B
slope
0.40
0.35
0.30
0.25
0.20
−3
−2
−1
0
z
C
TPF
1.00
0.75
0.50
0.25
0.00
0.00
0.25
0.50
0.75
1.0
FPF



14.4 Discussion

Plot A is for `c1` = -0.1322804, `da` = 1.197239 while plot C is for `c1` = 0.2225588, `da` = 1.740157. Plots B and D are the corresponding slope plots as functions of the binormal model z-sample. In plot A, the slope is infinite near the origin and the curve approaches the upper-right corner with finite slope. The situation is reversed in plot C where the slope is finite near the origin and the curve approaches the upper-right corner with zero slope.

These two readers are from a clinical dataset, `dataset01`. Highest rating inferred ROC data from original FROC data, were analyzed by PROPROC and the resulting parameter values are coded here. They were chosen as they demonstrate key differences in the shapes of proper ROC plots. Plot A corresponds to a negative value of `c1`, which implies `b < 1`. The slope of the proper ROC is infinite near the origin and approaches a positive constant near the upper right corner of the ROC. Plot C is for a positive value of `c1`, i.e., for `b > 1`. Now the slope of the proper ROC is finite near the origin and approaches zero near the upper right corner.

Considering plot D, as one "cuts" the slope axis horizontally with a sliding threshold, starting with very high values and moving downwards, the slope of the ROC curve starts at the origin with a large but finite value. This corresponds to the peak in plot D. Above the peak, there are no solutions for `z`. The slope decreases monotonically to zero, corresponding to the flattening out of the slope at zero for `z ~  -2`.

The two values of `z` corresponding to each cut implies, of course, that the binormal model based proper algorithm has to do a lot of bookkeeping, since each horizontal cut splits the decision axis into 3 regions. One can think of shrinking each of plots B & D horizontally to zero width, and all that remains is the slope axis with a thick vertical line superimposed on it, corresponding to the horizontally collapsed curves. In plot B the vertical line extends from positive infinity down to about 0.1, and represents the range of decision variable samples encountered by the observer on the likelihood ratio scale. In plot D the vertical line extends from a finite value (~9.4) to zero. For the stated binormal model parameters values outside of these ranges are not possible.

Chapter 15

Metz Eqn 36 numerical check

15.1 Helper functions

15.2 Main code and output

```
npts <-  10000
for (i in 1:2) {
  for (j in 1:5) {
    C  <-  c1[i,j]
    da  <-  d_a1[i,j]
    ret <- GetLimits(da,C)
    LL <- ret$LL;UL <- ret$UL
    vc  <-  seq (LL, UL, length.out = npts)
    TPF  <-  TruePositiveFraction (vc, da, C)
    FPF <- FalsePositiveFraction (vc, da, C)
    FPF <- rev(FPF);TPF <- rev(TPF)
    df2 <- data.frame(FPF = FPF, TPF = TPF)
    # do integral numerically
    numAuc <- trapz(FPF, TPF)
    # Implement Eqn. 36 from Metz-Pan paper
    rho <- -(1-C^2)/(1+C^2);sigma <- rbind(c(1, rho), c(rho, 1))
    lower <- rep(-Inf,2);upper <- c(-da/sqrt(2),0)
    aucProproc <- pnorm(da/sqrt(2)) + 2 * pmvnorm(lower, upper, sigma = sigma)
    aucProproc <-  as.numeric(aucProproc)
    cat("i = ", i,"j = ", j,"C = ", C, ", da = ", da, "aucProproc =", aucProproc, "Norm. Diff. = ", (aucPro
  }
}
#> i =  1 j =  1 C =  -0.1322804 , da =  1.197239 aucProproc = 0.8014164 Norm. Diff. =  3.520017e-08
#> i =  1 j =  2 C =  -0.08696513 , da =  1.771176 aucProproc = 0.8947898 Norm. Diff. =  4.741875e-08
#> i =  1 j =  3 C =  -0.1444419 , da =  1.481935 aucProproc = 0.8526605 Norm. Diff. =  3.515431e-08
#> i =  1 j =  4 C =  0.08046016 , da =  1.513757 aucProproc = 0.8577776 Norm. Diff. =  4.971428e-08
#> i =  1 j =  5 C =  0.2225588 , da =  1.740157 aucProproc = 0.8909392 Norm. Diff. =  2.699855e-08
#> i =  2 j =  1 C =  -0.08174248 , da =  0.6281251 aucProproc = 0.6716574 Norm. Diff. =  2.801793e-08
#> i =  2 j =  2 C =  0.04976448 , da =  0.9738786 aucProproc = 0.7544739 Norm. Diff. =  5.275242e-08
#> i =  2 j =  3 C =  -0.1326126 , da =  1.155871 aucProproc = 0.7931787 Norm. Diff. =  3.472577e-08
#> i =  2 j =  4 C =  0.1182226 , da =  1.620176 aucProproc = 0.8740274 Norm. Diff. =  3.922161e-08
#> i =  2 j =  5 C =  0.0781033 , da =  0.8928816 aucProproc = 0.7360989 Norm. Diff. =  3.798459e-08
```

15.3 Discussion

Note the close correspondence between the formula, Eqn. 36 in the Metz-Pan paper and the numerical estimate. As a historical note, Eqn. 31 and Eqn. 36 (they differ only in parameterizations) in the referenced publication are provided without proof – it was probably obvious to Prof Metz or he wanted to leave it to us “mere mortals” to figure it out, as a final parting gesture of his legacy. The author once put a significant effort into proving it and even had a bright graduate student from the biostatistics department work on it to no avail. The author has observed that these equations always yield very close to the numerical estimates, to within numerical precisions, so the theorem is correct empirically, but he has been unable to prove it analytically. It is left as an exercise for a gifted reader to prove/disprove these equations.

Software details

Chapter 16

Excel file and dataset details

16.1 Introduction

This chapter is included to document recent Excel file format changes and the new dataset structure.

16.2 ROC dataset

```
x <- DfReadDataFile("R/quick-start/rocCr.xlsx", newExcelFileFormat = TRUE)
```

16.2.1 The structure of a factorial ROC dataset object

`x` is a `list` with 3 members: `ratings`, `lesions` and `descriptions`.

```
str(x, max.level = 1)
#> List of 3
#>  $ ratings     :List of 3
#>  $ lesions     :List of 3
#>  $ descriptions:List of 7
```

The `x$ratings` member contains 3 sub-lists.

```
str(x$ratings)
#> List of 3
#>  $ NL   : num [1:2, 1:5, 1:8, 1] 1 3 2 3 2 2 1 2 3 2 ...
#>  $ LL   : num [1:2, 1:5, 1:5, 1] 5 5 5 5 5 5 5 5 5 5 ...
#>  $ LL_IL: logi NA
```

- `x$ratings$NL`, with dimension [2, 5, 8, 1], contains the ratings of normal cases. The first dimension (2) is the number of treatments, the second (5) is the number of readers and the third (8) is the total number of cases. For ROC datasets the fourth dimension is always unity. The five extra values [1] in the third dimension, of `x$ratings$NL` which are filled with `NA`s, are needed for compatibility with FROC datasets.
- `x$ratings$LL`, with dimension [2, 5, 5, 1], contains the ratings of abnormal cases. The third dimension (5) corresponds to the 5 diseased cases.

[1] With only 3 non-diseased cases why does one need 8 values?

- `x$ratings$LL_IL`, equal to NA', is there for compatibility with LROC data, `IL` denotes incorrect-localizations.

The `x$lesions` member contains 3 sub-lists.

```
str(x$lesions)
#> List of 3
#>  $ perCase: int [1:5] 1 1 1 1 1
#>  $ IDs    : num [1:5, 1] 1 1 1 1 1
#>  $ weights: num [1:5, 1] 1 1 1 1 1
```

- The `x$lesions$perCase` member is a vector with 5 ones representing the 5 diseased cases in the dataset.

- The `x$lesions$IDs` member is an array with 5 ones.

```
x$lesions$weights
#>      [,1]
#> [1,]    1
#> [2,]    1
#> [3,]    1
#> [4,]    1
#> [5,]    1
```

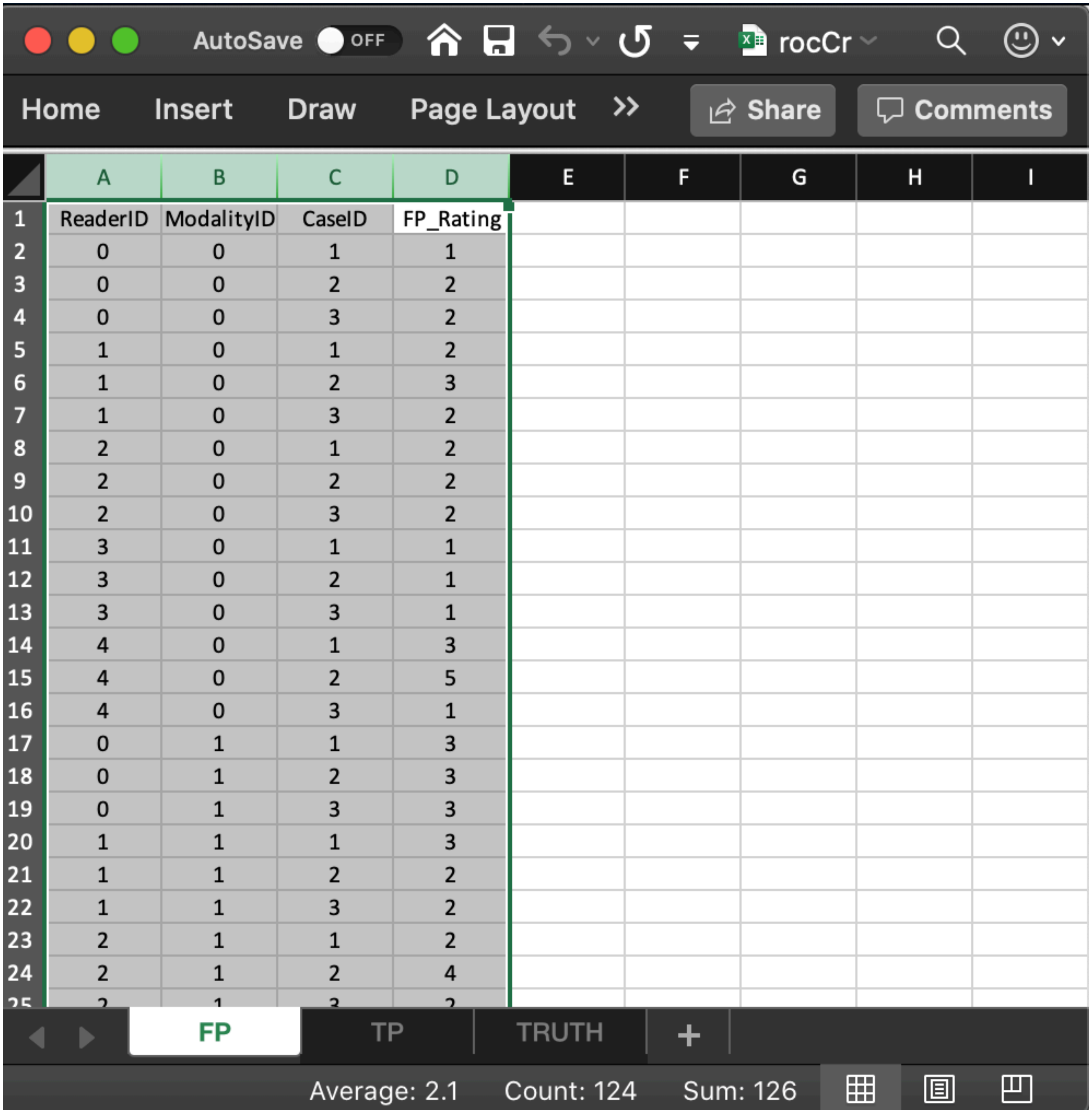
`x$lesions$weights` member is an array with 5 ones. These are irrelevant for ROC datasets. They are there for compatibility with FROC datasets.

`x$descriptions` contains 7 sub-lists.

```
str(x$descriptions)
#> List of 7
#>  $ fileName     : chr "rocCr"
#>  $ type         : chr "ROC"
#>  $ name         : logi NA
#>  $ truthTableStr: num [1:2, 1:5, 1:8, 1:2] 1 1 1 1 1 1 1 1 1 1 ...
#>  $ design       : chr "FCTRL"
#>  $ modalityID   : Named chr [1:2] "0" "1"
#>   ..- attr(*, "names")= chr [1:2] "0" "1"
#>  $ readerID     : Named chr [1:5] "0" "1" "2" "3" ...
#>   ..- attr(*, "names")= chr [1:5] "0" "1" "2" "3" ...
```

- `x$descriptions$fileName` is intended for internal use.
- `x$descriptions$type` indicates that this is an `ROC` dataset.
- `x$descriptions$name` is intended for internal use.
- `x$descriptions$truthTableStr` is intended for internal use, see Section 16.3.2.
- `x$descriptions$design` specifies the dataset design, which is "FCTRL" in the present example ("FCTRL" = a factorial dataset).
- `x$descriptions$modalityID` is a vector with two elements `"0"` and `"1"`, the names of the two modalities.
- `x$readerID` is a vector with five elements `"0"`, `"1"`, `"2"`, `"3"` and `"4"`, the names of the five readers.

16.2.2 The FP worksheet

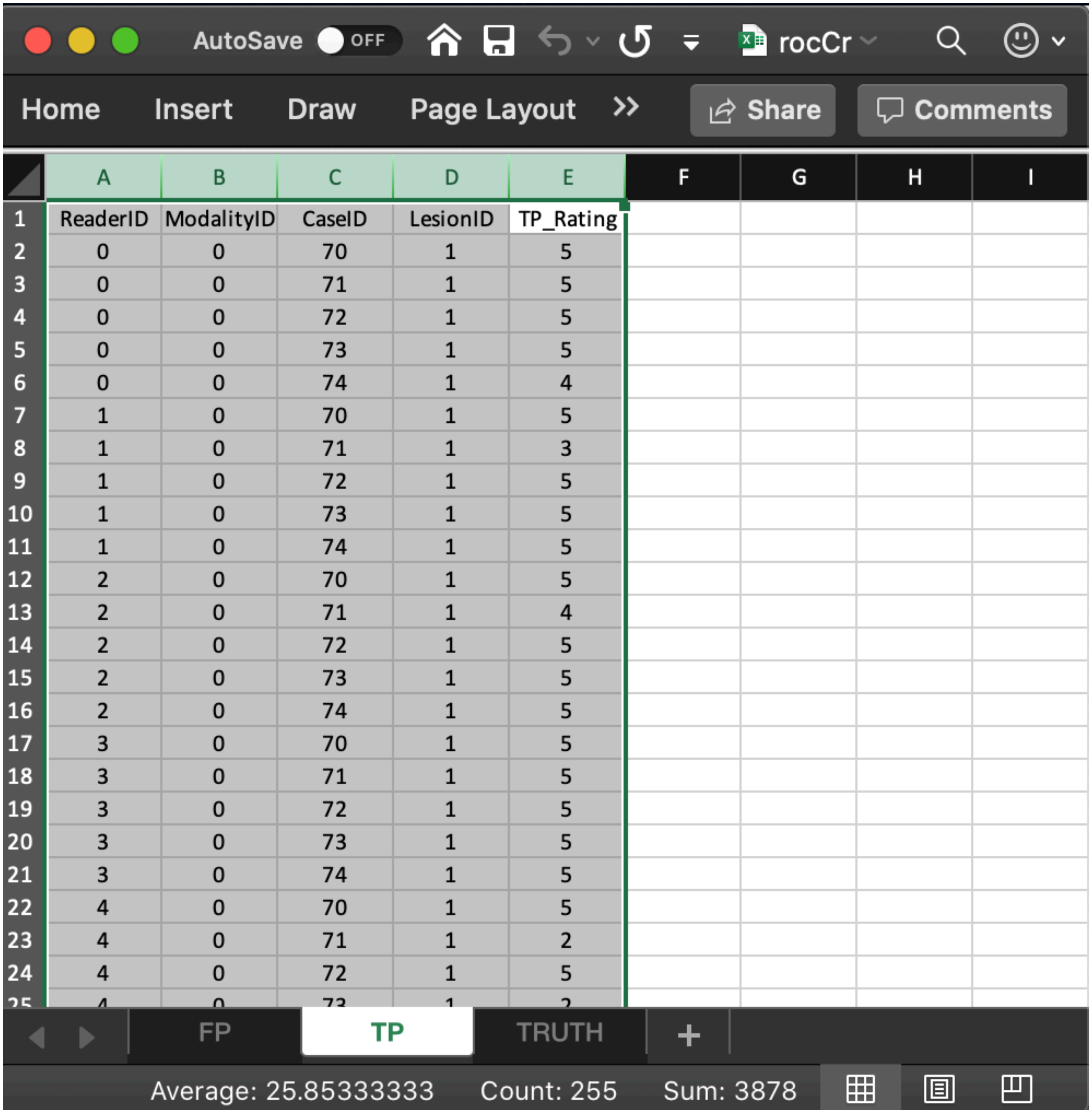


ReaderID	ModalityID	CaseID	FP_Rating
0	0	1	1
0	0	2	2
0	0	3	2
1	0	1	2
1	0	2	3
1	0	3	2
2	0	1	2
2	0	2	2
2	0	3	2
3	0	1	1
3	0	2	1
3	0	3	1
4	0	1	3
4	0	2	5
4	0	3	1
0	1	1	3
0	1	2	3
0	1	3	3
1	1	1	3
1	1	2	2
1	1	3	2
2	1	1	2
2	1	2	4
2	1	3	2

- The list member `x$ratings$NL` is an array with `dim = c(2,5,8,1)`.
    - The first dimension (2) comes from the number of modalities.
    - The second dimension (5) comes from the number of readers.
    - The third dimension (8) comes from the **total** number of cases.
    - The fourth dimension is always 1 for an ROC dataset.
- The value of `x$ratings$NL[1,5,2,1]`, i.e., 5, corresponds to row 15 of the FP table, i.e., to `ModalityID` = 0, `ReaderID` = 4 and `CaseID` = 2.
- The value of `x$ratings$NL[2,3,2,1]`, i.e., 4, corresponds to row 24 of the FP table, i.e., to `ModalityID` 1, `ReaderID` 2 and `CaseID` 2.

- All values for case index > 3 and case index <= 8 are `-Inf`. For example the value of `x$ratings$NL[2,3,4,1]` is `-Inf`. This is because there are only 3 non-diseased cases. The extra length is needed for compatibility with FROC datasets.

16.2.3 The `TP` worksheet



	A	B	C	D	E
1	ReaderID	ModalityID	CaseID	LesionID	TP_Rating
2	0	0	70	1	5
3	0	0	71	1	5
4	0	0	72	1	5
5	0	0	73	1	5
6	0	0	74	1	4
7	1	0	70	1	5
8	1	0	71	1	3
9	1	0	72	1	5
10	1	0	73	1	5
11	1	0	74	1	5
12	2	0	70	1	5
13	2	0	71	1	4
14	2	0	72	1	5
15	2	0	73	1	5
16	2	0	74	1	5
17	3	0	70	1	5
18	3	0	71	1	5
19	3	0	72	1	5
20	3	0	73	1	5
21	3	0	74	1	5
22	4	0	70	1	5
23	4	0	71	1	2
24	4	0	72	1	5
25	4	0	73	1	2

Average: 25.85333333 Count: 255 Sum: 3878

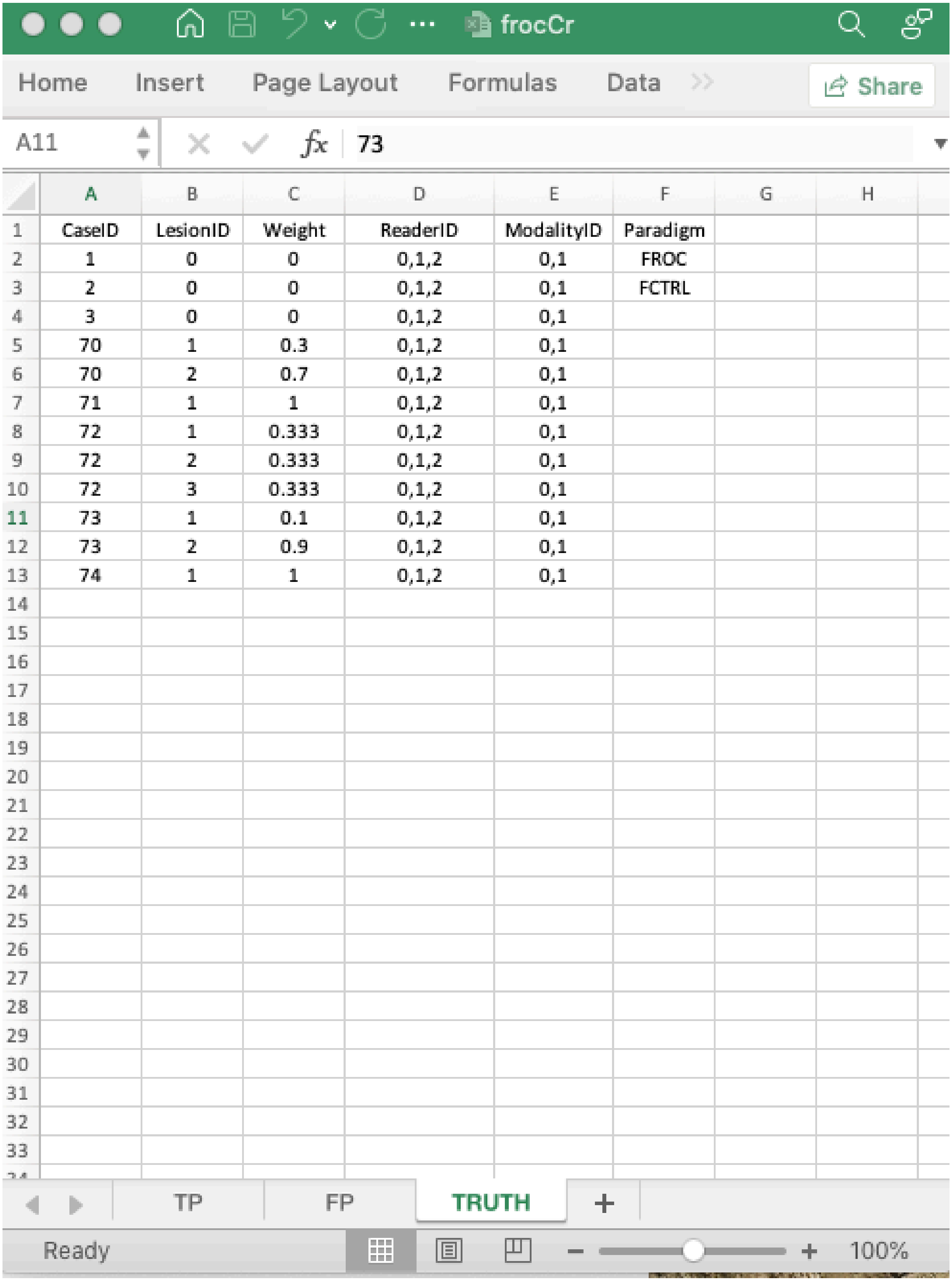
- The list member `x$ratings$LL` is an array with `dim = c(2,5,5,1)`.
  - The first dimension (2) comes from the number of modalities.
  - The second dimension (5) comes from the number of readers.
  - The third dimension (5) comes from the number of diseased cases.
  - The fourth dimension is always 1 for an ROC dataset.

- The value of `x$ratings$LL[1,1,5,1]`, i.e., 4, corresponds to row 6 of the TP table, i.e., to `ModalityID` = 0, `ReaderID` = 0 and `CaseID` = 74.
- The value of `x$ratings$LL[1,2,2,1]`, i.e., 3, corresponds to row 8 of the TP table, i.e., to `ModalityID` = 0, `ReaderID` = 1 and `CaseID` = 71.
- The value of `x$ratings$LL[1,4,4,1]`, i.e., 5, corresponds to row 21 of the TP table, i.e., to `ModalityID` = 0, `ReaderID` = 3 and `CaseID` = 74.
- The value of `x$ratings$LL[1,5,2,1]`, i.e., 2, corresponds to row 23 of the TP table, i.e., to `ModalityID` = 0, `ReaderID` = 4 and `CaseID` = 71.
- There are no `-Inf` values in `x$ratings$LL`: `any(x$ratings$LL == -Inf)` = FALSE. This is true for any ROC dataset.

16.2.4 caseIndex vs. caseID

- The `caseIndex` is the array index used to access elements in the NL and LL arrays. The case-index is always an integer in the range 1, 2, ..., up to the array length. Remember that unlike C++, R indexing starts from 1.
- The `caseID` is any integer value, including zero, used to uniquely label the cases.
- Regardless of what order they occur in the worksheet, the non-diseased cases are always ordered first. In the current example the case indices are 1, 2 and 3, corresponding to the three non-diseased cases with `caseIDs` equal to 1, 2 and 3.
- Regardless of what order they occur in the worksheet, in the NL array the diseased cases are always ordered *after* the last non-diseased case. In the current example the case indices in the `NL` array are 4, 5, 6, 7 and 8, corresponding to the five diseased cases with `caseIDs` equal to 70, 71, 72, 73, and 74. In the `LL` array they are indexed 1, 2, 3, 4 and 5. Some examples follow:
- `x$ratings$NL[1,3,2,1]`, a FP rating, refers to `ModalityID` 0, `ReaderID` 2 and `CaseID` 2 (since the modality and reader IDs start with 0).
- `x$ratings$NL[2,5,4,1]`, a FP rating, refers to `ModalityID` 1, `ReaderID` 4 and `CaseID` 70, the first diseased case; this is `-Inf`.
- `x$ratings$NL[1,4,8,1]`, a FP rating, refers to `ModalityID` 0, `ReaderID` 3 and `CaseID` 74, the last diseased case; this is `-Inf`.
- `x$ratings$NL[1,3,9,1]`, a FP rating, is an illegal value, as the third index cannot exceed 8.
- `x$ratings$NL[1,3,8,2]`, a FP rating, is an illegal value, as the fourth index cannot exceed 1 for an ROC dataset.
- `x$ratings$LL[1,3,1,1]`, a TP rating, refers to `ModalityID` 0, `ReaderID` 2 and `CaseID` 70, the first diseased case.
- `x$ratings$LL[2,5,4,1]`, a TP rating, refers to `ModalityID` 1, `ReaderID` 4 and `CaseID` 73, the fourth diseased case.

16.3 FROC dataset



	A	B	C	D	E	F
1	CaseID	LesionID	Weight	ReaderID	ModalityID	Paradigm
2	1	0	0	0,1,2	0,1	FROC
3	2	0	0	0,1,2	0,1	FCTRL
4	3	0	0	0,1,2	0,1	
5	70	1	0.3	0,1,2	0,1	
6	70	2	0.7	0,1,2	0,1	
7	71	1	1	0,1,2	0,1	
8	72	1	0.333	0,1,2	0,1	
9	72	2	0.333	0,1,2	0,1	
10	72	3	0.333	0,1,2	0,1	
11	73	1	0.1	0,1,2	0,1	
12	73	2	0.9	0,1,2	0,1	
13	74	1	1	0,1,2	0,1	

16.3.1 The structure of a factorial FROC dataset

```
x <- DfReadDataFile("images/software-details/frocCr.xlsx", newExcelFileFormat = TRUE)
```

The dataset `x` is a `list` variable with 3 members: `x$ratings`, `x$lesions` and `x$descriptions`.

```
str(x, max.level = 1)
#> List of 3
#>  $ ratings     :List of 3
#>  $ lesions     :List of 3
#>  $ descriptions:List of 7
```

The `x$ratings` member contains 3 sub-lists.

```
str(x$ratings)
#> List of 3
#>  $ NL   : num [1:2, 1:3, 1:8, 1:2] 1.02 2.89 2.21 3.01 2.14 ...
#>  $ LL   : num [1:2, 1:3, 1:5, 1:3] 5.28 5.2 5.14 4.77 4.66 4.87 3.01 3.27 3.31 3.19 ...
#>  $ LL_IL: logi NA
```

- There are `K2 = 5` diseased cases (the length of the third dimension of `x$ratings$LL`) and `K1 = 3` non-diseased cases (the length of the third dimension of `x$ratings$NL` minus `K2`).
- `x$ratings$NL`, a [2, 3, 8, 2] array, contains the NL ratings on non-diseased and diseased cases.
- `x$ratings$LL`, a [2, 3, 5, 3] array, contains the ratings of LLs on diseased cases.
- `x$ratings$LL_IL` is `NA`, this field applies to an LROC dataset (contains incorrect localizations on diseased cases).

The `x$lesions` member contains 3 sub-lists.

```
str(x$lesions)
#> List of 3
#>  $ perCase: int [1:5] 2 1 3 2 1
#>  $ IDs    : num [1:5, 1:3] 1 1 1 1 1 ...
#>  $ weights: num [1:5, 1:3] 0.3 1 0.333 0.1 1 ...
```

- `x$lesions$perCase` is the number of lesions per diseased case vector, i.e., 2, 1, 3, 2, 1.
- `max(x$lesions$perCase)` is the maximum number of lesions per case, i.e., rmax(x$lesions$perCase)‘.
- `x$lesions$weights` is the weights of lesions.

```
x$lesions$weights
#>           [,1]      [,2]      [,3]
#> [1,] 0.3000000 0.7000000      -Inf
#> [2,] 1.0000000      -Inf      -Inf
#> [3,] 0.3333333 0.3333333 0.3333333
#> [4,] 0.1000000 0.9000000      -Inf
#> [5,] 1.0000000      -Inf      -Inf
```

The weights for the first diseased case are 0.3 and 0.7. The weight for the second diseased case is 1. For the third diseased case the three weights are 1/3 each, etc. For each diseased case the finite weights sum to unity.

`x$descriptions` contains 7 sub-lists.

```
str(x$descriptions)
#> List of 7
#>  $ fileName     : chr "frocCr"
#>  $ type         : chr "FROC"
#>  $ name         : logi NA
#>  $ truthTableStr: num [1:2, 1:3, 1:8, 1:4] 1 1 1 1 1 1 1 1 1 1 ...
#>  $ design       : chr "FCTRL"
#>  $ modalityID   : Named chr [1:2] "0" "1"
#>   ..- attr(*, "names")= chr [1:2] "0" "1"
#>  $ readerID     : Named chr [1:3] "0" "1" "2"
#>   ..- attr(*, "names")= chr [1:3] "0" "1" "2"
```

- `x$descriptions$filename` is for internal use.
- `x$descriptions$type` is FROC, which specifies the data collection method.
- `x$descriptions$name` is for internal use.
- `x$descriptions$truthTableStr` is for internal use; it quantifies the structure of the dataset; it is explained in the next section.
- `x$descriptions$design` is FCTRL; it specifies the study design.
- `x$descriptions$modalityID` is a vector with two elements 0, 1 naming the two modalities.
- `x$readerID` is a vector with three elements 0, 1, 2 naming the three readers.

16.3.2 `truthTableStr`

- For this dataset `I` = 2, `J` = 3 and `K` = 8.
- `truthTableStr` is a `2 x 3 x 8 x 4` array, i.e., `I` x `J` x `K` x (maximum number of lesions per case plus 1 - the `plus 1` is needed to accommodate non-diseased cases).
- Each entry in this array is either `1`, meaning the corresponding interpretation happened, or `NA`, meaning the corresponding interpretation did not happen.

16.3.2.1 Explanation for non-diseased cases

Since the fourth index is set to 1, in the following code only non-diseased cases yield ones and all diseased cases yield `NA`.

```
all(x$descriptions$truthTableStr[,,1:3,1] ==1)
#> [1] TRUE
all(is.na(x$descriptions$truthTableStr[,,4:8,1]))
#> [1] TRUE
```

16.3.2.2 Explanation for diseased cases with one lesion

Since the fourth index is set to 2, in the following code all non-diseased cases yield `NA` and all diseased cases yield 1 as all diseased cases have at least one lesion.

```
all(is.na(x$descriptions$truthTableStr[,,1:3,2]))
#> [1] TRUE
all(x$descriptions$truthTableStr[,,4:8,2] == 1)
#> [1] TRUE
```

16.3.2.3 Explanation for diseased cases with two lesions

Since the fourth index is set to 3, in the following code all non-diseased cases yield `NA`; the first diseased case `70` yields 1 (this case contains two lesions); the second disease case `71` yields `NA` (this case contains only one lesion);

the third disease case 72 yields NA (this case contains only two lesions); the fourth disease case 73 yields 1 (this case contains two lesions); the fifth disease case 74 yields NA (this case contains one lesion).

```
# all non diseased cases
all(is.na(x$descriptions$truthTableStr[,,1:3,3]))
#> [1] TRUE
# first diseased case
all(x$descriptions$truthTableStr[,,4,3] == 1)
#> [1] TRUE
# second diseased case
all(is.na(x$descriptions$truthTableStr[,,5,3]))
#> [1] TRUE
# third diseased case
all(x$descriptions$truthTableStr[,,6,3] == 1)
#> [1] TRUE
# fourth diseased case
all(x$descriptions$truthTableStr[,,7,3] == 1)
#> [1] TRUE
# fifth diseased case
all(is.na(x$descriptions$truthTableStr[,,8,3]))
#> [1] TRUE
```

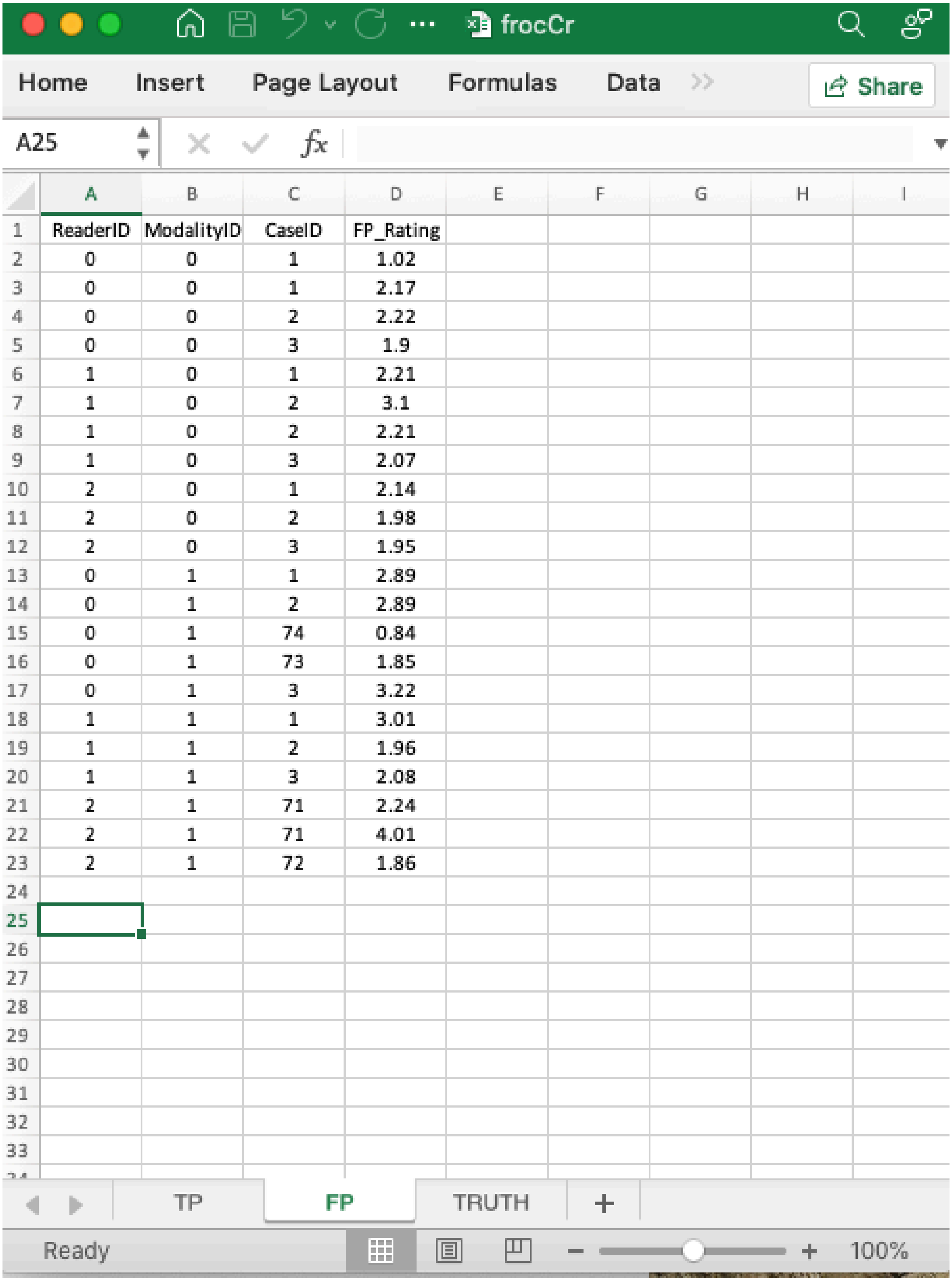
16.3.2.4 Explanation for diseased cases with three lesions

Since the fourth index is set to 4, in the following code all non-diseased cases yield NA; the first diseased case 70 yields NA (this case contains two lesions); the second disease case 71 yields NA (this case contains one lesion); the third disease case 72 yields NA (this case contains two lesions); the fourth disease case 73 yields 1 (this case contains three lesions); the fifth disease case 74 yields NA (this case contains one lesion).

```
# all non diseased cases
all(is.na(x$descriptions$truthTableStr[,,1:3,4]))
#> [1] TRUE
# first diseased case
all(is.na(x$descriptions$truthTableStr[,,4,4]))
#> [1] TRUE
# second diseased case
all(is.na(x$descriptions$truthTableStr[,,5,4]))
#> [1] TRUE
# third diseased case
all(x$descriptions$truthTableStr[,,6,4] == 1)
#> [1] TRUE
# fourth diseased case
all(is.na(x$descriptions$truthTableStr[,,7,4]))
#> [1] TRUE
# fifth diseased case
all(is.na(x$descriptions$truthTableStr[,,8,4]))
#> [1] TRUE
```

16.3.3 The FP worksheet

These are found in the FP or NL worksheet:



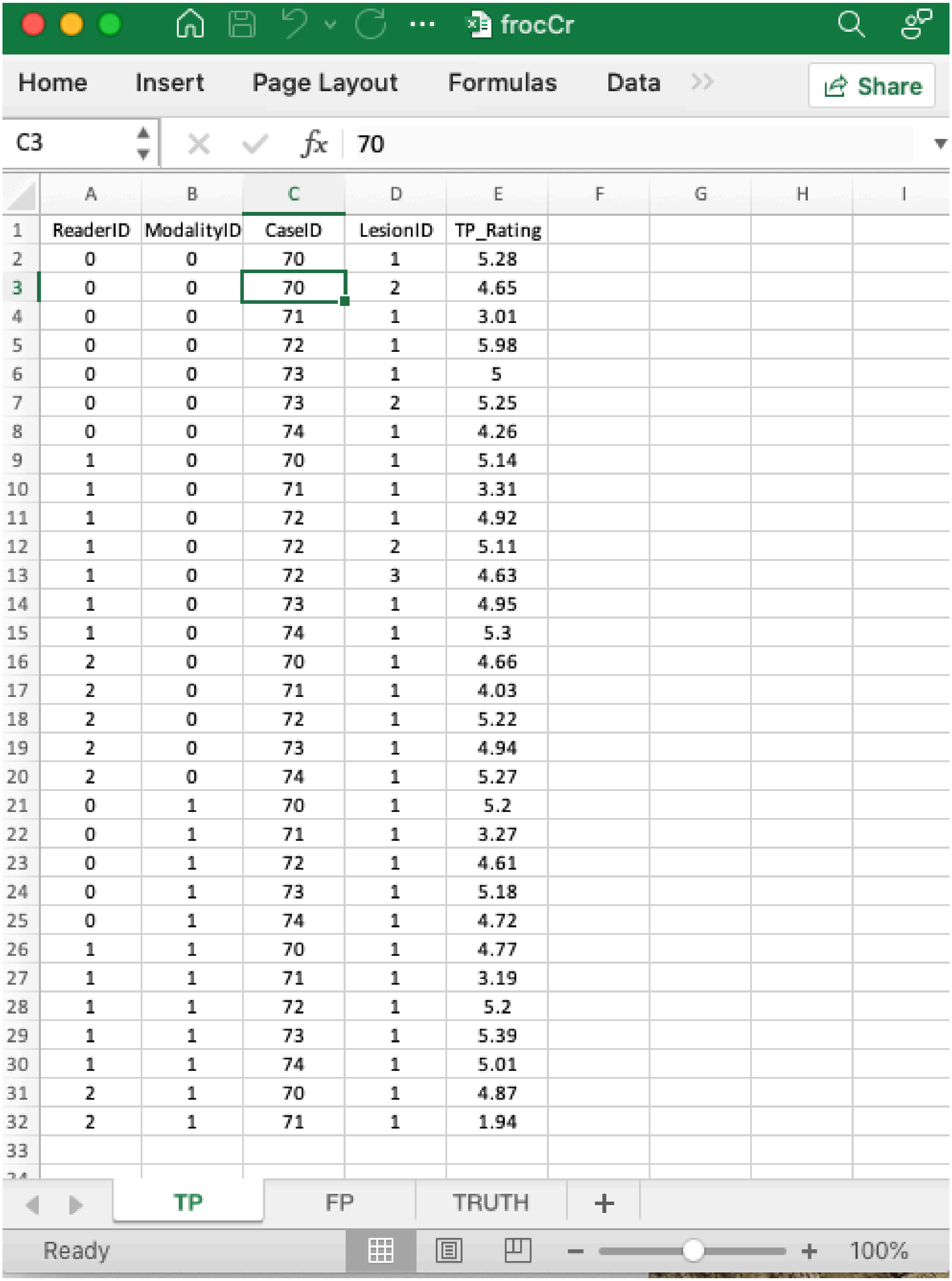
ReaderID	ModalityID	CaseID	FP_Rating
0	0	1	1.02
0	0	1	2.17
0	0	2	2.22
0	0	3	1.9
1	0	1	2.21
1	0	2	3.1
1	0	2	2.21
1	0	3	2.07
2	0	1	2.14
2	0	2	1.98
2	0	3	1.95
0	1	1	2.89
0	1	2	2.89
0	1	74	0.84
0	1	73	1.85
0	1	3	3.22
1	1	1	3.01
1	1	2	1.96
1	1	3	2.08
2	1	71	2.24
2	1	71	4.01
2	1	72	1.86

- The common vertical length is 22 in this example.
- `ReaderID`: the reader labels: `0`, `1`,`2, as declared in the`Truth‘ worksheet.
- `ModalityID`: the modality labels: `0` or `1`, as declared in the `Truth` worksheet.
- `CaseID`: `1`, `2`, `3`, `71`, `72`, `73`, `74`, as declared in the `Truth` worksheet; note that not all cases have NL marks on them.

- `NL_Rating`: the ratings of non-diseased cases.

16.3.4 The TP worksheet

These are found in the `TP` or `LL` worksheet, see below.



	A	B	C	D	E
1	ReaderID	ModalityID	CaseID	LesionID	TP_Rating
2	0	0	70	1	5.28
3	0	0	70	2	4.65
4	0	0	71	1	3.01
5	0	0	72	1	5.98
6	0	0	73	1	5
7	0	0	73	2	5.25
8	0	0	74	1	4.26
9	1	0	70	1	5.14
10	1	0	71	1	3.31
11	1	0	72	1	4.92
12	1	0	72	2	5.11
13	1	0	72	3	4.63
14	1	0	73	1	4.95
15	1	0	74	1	5.3
16	2	0	70	1	4.66
17	2	0	71	1	4.03
18	2	0	72	1	5.22
19	2	0	73	1	4.94
20	2	0	74	1	5.27
21	0	1	70	1	5.2
22	0	1	71	1	3.27
23	0	1	72	1	4.61
24	0	1	73	1	5.18
25	0	1	74	1	4.72
26	1	1	70	1	4.77
27	1	1	71	1	3.19
28	1	1	72	1	5.2
29	1	1	73	1	5.39
30	1	1	74	1	5.01
31	2	1	70	1	4.87
32	2	1	71	1	1.94
33					

- This worksheet has the ratings of diseased cases.
- `ReaderID`: the reader labels: these must be from `0`, `1`, `2`, as declared in the `Truth` worksheet.
- `ModalityID`: `0` or `1`, as declared in the `Truth` worksheet.
- `CaseID`: these must be from `70`, `71`, `72`, `73`, `74`, as declared in the `Truth` worksheet; not all diseased cases have LL marks.

- `LL_Rating`: the ratings of diseased cases.

Bibliography

Chakraborty, D. and Zhai, X. (2022). *RJafroc: Artificial Intelligence Systems and Observer Performance*. R package version 2.1.1.9000.

Chakraborty, D. P. (2010). Prediction accuracy of a sample-size estimation method for ROC studies. *Academic radiology*, 17:628–638.

Chakraborty, D. P. (2017). *Observer Performance Methods for Diagnostic Imaging: Foundations, Modeling, and Applications with R-Based Examples*. CRC Press, Boca Raton, FL.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences*. Lawrence Erlbaum Associates, 2 edition.

Hillis, S. L. and Berbaum, K. S. (2004). Power estimation for the dorfman-berbaum-metz method. *Acad. Radiol.*, 11(11):1260–1273.

Hillis, S. L., Obuchowski, N. A., and Berbaum, K. S. (2011). Power estimation for multireader ROC methods: An updated and unified approach. *Academic Radiology*, 18(2):129–142.

ICRU (1996). Medical imaging: the assessment of image quality. *JOURNAL OF THE ICRU*, 54(1):37–40.

Obuchowski, N. A. (1998). Sample size calculations in studies of test accuracy. *Statistical Methods in Medical Research*, 7(4):371–392.

Zanca, F., Jacobs, J., Van Ongeval, C., Claus, F., Celis, V., Geniets, C., Provost, V., Pauwels, H., Marchal, G., and Bosmans, H. (2009). Evaluation of clinical image processing algorithms used in digital mammography. *Medical Physics*, 36(3):765–775.