

# Excel ファイル

Andrew Ba Tran

## 目次

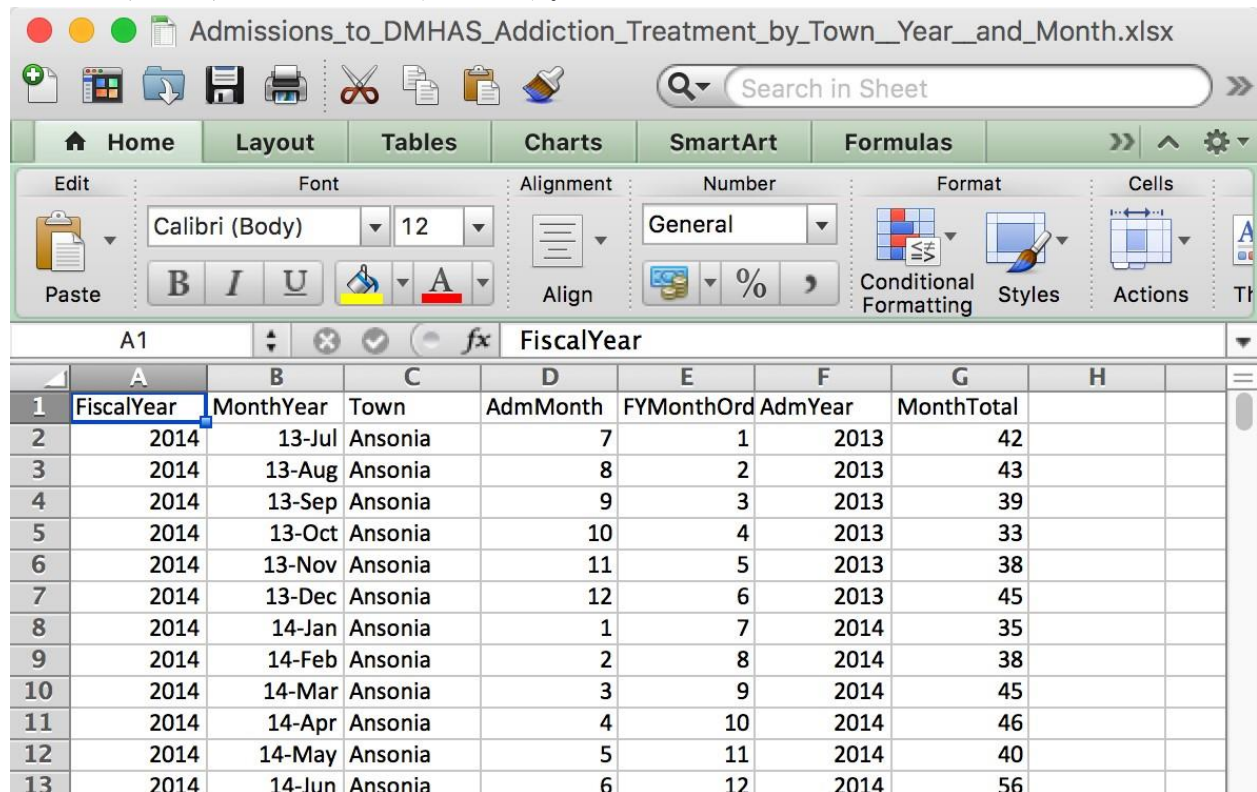
Excel ファイルとは .....	1
ファイルの中身はどうなっているのか .....	2
Excel ファイルのインポート .....	2
read_excel() .....	3
read_excel() 再挑戦 .....	3
クリーニング (part 1) .....	4
列の名前を変更する .....	5
df_xl シートは十分クリーンですか? .....	6
NA の排除 .....	6
Excel へのエクスポート .....	7

This is from the second chapter of [learn.r-journalism.com](http://learn.r-journalism.com).

Excel スプレッドシートには、(Excel のファイルである) ワークブックとして複数のスプレッドシートを含めることができるという固有の特徴があります。

## Excel ファイルとは

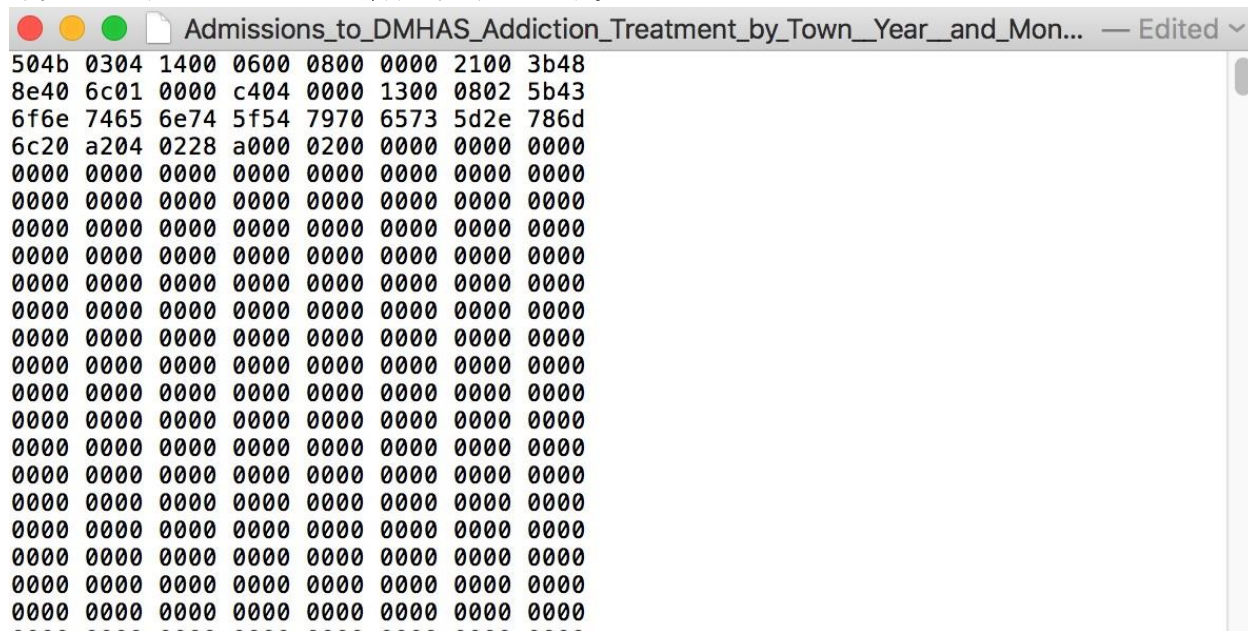
Excel のファイル名は.xls か .xlsx で終わります。



	A	B	C	D	E	F	G	H
1	FiscalYear	MonthYear	Town	AdmMonth	FYMonthOrd	AdmYear	MonthTotal	
2	2014	13-Jul	Ansonia	7	1	2013	42	
3	2014	13-Aug	Ansonia	8	2	2013	43	
4	2014	13-Sep	Ansonia	9	3	2013	39	
5	2014	13-Oct	Ansonia	10	4	2013	33	
6	2014	13-Nov	Ansonia	11	5	2013	38	
7	2014	13-Dec	Ansonia	12	6	2013	45	
8	2014	14-Jan	Ansonia	1	7	2014	35	
9	2014	14-Feb	Ansonia	2	8	2014	38	
10	2014	14-Mar	Ansonia	3	9	2014	45	
11	2014	14-Apr	Ansonia	4	10	2014	46	
12	2014	14-May	Ansonia	5	11	2014	40	
13	2014	14-Jun	Ansonia	6	12	2014	56	

## ファイルの中身はどうなっているのか

奇妙でしょう？このままでは解析は絶対無理です。



## Excel ファイルのインポート

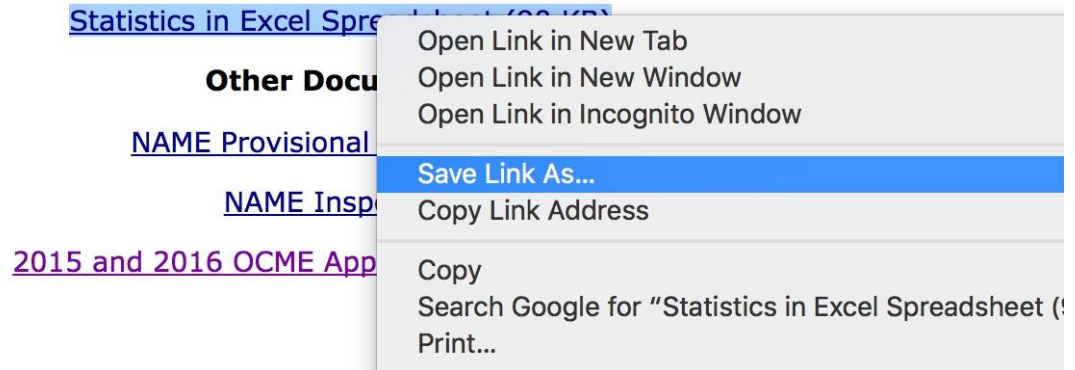
- Excel のインポートは複雑で、**readxl package** が必要です。
- Excel ファイルを処理して追加のシートを作成できるパッケージは他にもありますが、このインスタンスでは必要ありません。

まず `readxl` パッケージをインストールします。

パッケージの一部として `readxl` があります

```
## readxl をインストールしていなければ、以下の行のコメントを外して実行してください。  
#install.packages("readxl") library(readxl)
```

csv とは異なり、Excel シートの URL をコピーして貼り付けることはできません。最初にファイルをダウンロードする必要があります。[Excel データリンク](<http://www.ct.gov/ocme/cwp/view.asp?a=2165&Q=295128&ocmeNav=|>) を右クリックしファイルに名前を付けて保存をクリックします。



## read\_excel()

Excel スプレッドシートには複数のシートがあり、`read\_excel()` ではインポート時に特定のシートを参照する必要があるため、Excel がどのようなになっているかあらかじめ調べておくのが最善です。

一番はじめのシートでやってみましょう。

```
df_xl <- read_excel("data/StatisticsSummary.xls", sheet=1)
```

見てみましょう。

**View(df\_xl)**

						Office of The Chief Medical Examiner			
1	NA	NA	NA	NA	NA	Summary of Cases by Fiscal Year	NA	NA	NA
2	Fiscal Year 7/1-6/30	Accessions	Autopsies	Exam-inations	Other Cases	TOTAL	Cremations	% incl crem	Homicides
3	1990.000000	12211.000000	1384.000000	109.000000	39.000000	1532.000000	4367.000000	0.360000	NA
4	1991.000000	11899.000000	1269.000000	120.000000	17.000000	1406.000000	4999.000000	0.420000	195.000000
5	1992.000000	12333.000000	1270.000000	127.000000	40.000000	1437.000000	5574.000000	0.450000	194.000000
6	1993.000000	13035.000000	1291.000000	123.000000	40.000000	1454.000000	6352.000000	0.490000	200.000000
7	1994.000000	13174.000000	1307.000000	189.000000	40.000000	1536.000000	6622.000000	0.500000	219.000000
8	1995.000000	13364.000000	1277.000000	193.000000	37.000000	1507.000000	6910.000000	0.520000	194.000000
9	1996.000000	13380.000000	1113.000000	186.000000	23.000000	1322.000000	7078.000000	0.530000	150.000000

さっぱりだめです。

Excel ファイルは、Excel では見栄えがよくても R では意味がないフォーマットを好むという問題があります。

## read\_excel() 再挑戦

今回は `skip=2` を追加したのでデータを取り込むときに最初の行をスキップします。

```
df_xl <- read_excel("data/StatisticsSummary.xls", sheet=1, skip=2)
```

ずっとよくなりました。

**View(df\_xl)**

	Fiscal Year 7/1-6/30	Accessions	Autopsies	Exam-inations	Other Cases	TOTAL	Cremations	% incl crem	Homicides	Suicide	Accidents	Undetermined	ALL	U 20	U 17	SIDS	C
1	1990	12211	1384	109	39	1532	4367	0.3600000	NA	NA	NA	NA	NA	NA	NA	NA	
2	1991	11899	1269	120	17	1406	4999	0.4200000	195	314	724	100	229	40	20	42	
3	1992	12333	1270	127	40	1437	5574	0.4500000	194	378	725	78	285	45	21	44	
4	1993	13035	1291	123	40	1454	6352	0.4900000	200	316	772	82	267	39	15	27	
5	1994	13174	1307	189	40	1536	6622	0.5000000	219	332	848	58	298	45	20	36	
6	1995	13364	1277	193	37	1507	6910	0.5200000	194	316	804	44	258	39	15	28	
7	1996	13380	1113	186	23	1322	7078	0.5300000	150	323	786	57	220	26	12	19	
8	1997	13982	1176	192	27	1395	7740	0.5500000	169	308	795	56	221	34	12	22	
9	1998	13928	1229	215	27	1471	7674	0.5500000	139	265	833	57	173	22	6	22	
10	1999	14661	1220	213	51	1484	8357	0.5700000	151	285	890	79	189	23	4	27	
11	2000	14689	1186	290	46	1522	8752	0.6000000	104	304	875	70	202	14	4	24	

**Warning:** 列名がスペースと記号で保存されていることに注意してください。

# the `colnames()` 関数はデータフレーム `colnames(df_xl)` の列名をリストします。

```
## [1] "Fiscal Year"          "7/1-6/30" "Accessions"
## [3] "Autopsies"           "Exam-inations"
## [5] "Other Cases"         "TOTAL"
## [7] "Cremations"          "% incl crem"
## [9] "Homicides"           "Suicide"
## [11] "Accidents"           "Undetermined"
## [13] "ALL"                  "U 20"
## [15] "U 17"                 "SIDS"
## [17] "Clinicals"
```

それでは、スペースを含む列のデータをどのように参照すればよいのでしょうか。列を抽出するためにいつも通り `\$` を使ってみましょう。

```
head(df_xl$Other Cases)
```

```
## Error: <text>:1:18: unexpected symbol
## 1: head(df_xl$Other Cases
##
```

このように、スペースがあるとうまくいきません。スペースのある列を処理するには、バッククォートを追加します。

```
head(df_xl$"Other Cases")
```

```
## [1] 39 17 40 40 40 37
```

限られた基礎的な作業ならば、いくつか余分な操作をするだけでよいので大丈夫でしょう。

ただし、これから実行する作業を見越して、文字やスペースがないよう列名を簡略化する必要があります。これが

## クリーニング (part 1)

列名に `make.names()` 関数を使用します。この関数は、文字ベクトルから構文的に有効な名前を作成します（スペースを取り除きピリオドで置き換えます）。

```
colnames(df_xl) <- make.names(colnames(df_xl))
```

見てみましょう。

View(df\_xl)

	Fiscal.Year.....7.1.6.30	Accessions	Autopsies	Exam.inations	Other.Cases	TOTAL	Cremations	X..incl.crem	Homicides	Suicide	Accidents	Undetermined
1	1990	12211	1384	109	39	1532	4367	0.3600000	NA	NA	NA	
2	1991	11899	1269	120	17	1406	4999	0.4200000	195	314	724	
3	1992	12333	1270	127	40	1437	5574	0.4500000	194	378	725	
4	1993	13035	1291	123	40	1454	6352	0.4900000	200	316	772	
5	1994	13174	1307	189	40	1536	6622	0.5000000	219	332	848	
6	1995	13364	1277	193	37	1507	6910	0.5200000	194	316	804	
7	1996	13380	1113	186	23	1322	7078	0.5300000	150	323	786	
8	1997	13982	1176	192	27	1395	7740	0.5500000	169	308	795	
9	1998	13928	1229	215	27	1471	7674	0.5500000	139	265	833	

```
colnames(df_xl)
```

```
## [1] "Fiscal.Year.....7.1.6.30" "Accessions"
## [3] "Autopsies"                  "Exam.inations"
## [5] "Other.Cases"                "TOTAL"
## [7] "Cremations"                 "X..incl.crem"
## [9] "Homicides"                  "Suicide"
## [11] "Accidents"                  "Undetermined"
## [13] "ALL"                        "U.20" ## [15] "U.17"
## [17] "Clinicals"
## [19] "SIDS"
```

まあまあです。

名前にまだ少し奇妙なところがありますが、それは空白がピリオドに置き換えられたためです。

最初の列をチェックしてください: `Fiscal.Year.....7.1.6.30`

後で入力しやすいように変更しましょう。.

## 列の名前を変更する

ベース R で行う方法と dplyr パッケージを使用する方法を紹介します。

```
Fiscal.Year.....7.1.6.30 を colnames(dataframe_name)[colnames(dataframe_name)
== 'ColumnNameToBeChanged'] <- 'NewColumnName'にコピーします。
```

*# プロセスを示すためなので、コードを実行しないでください。*

```
colnames(df_xl)[colnames(df_xl) == 'Fiscal.Year.....7.1.6.30'] <- 'Year'
```

dplyr で行うときは `rename()` 関数を使います。

## dplyr をインストールしていなければ、以下の行のコメントを外して実行してください。

```
# install.packages("dplyr") library(dplyr)
```

```
df_xl <- rename(df_xl, Year=Fiscal.Year.....7.1.6.30)
```

それはわずかな違いです。丸括弧、角括弧と等号が少なくなります。

引用符を追加する必要はありません。



見てみましょう。

```
colnames(df_xl)
```

```
## [1] "Year"                "Accessions"         "Autopsies"          "Examinations"
## [5] "Other.Cases" "TOTAL" "Cremations" "X.incl.crem" ## [9] "Homicides" "Suicide"
"Accidents" "Undetermined"
## [13] "ALL"                "U.20"                "U.17"                "SIDS"
## [17] "Clinicals"
```

必要に応じて他の列の名前も修正してください。私はとりあえずこのままにしておきます。

## df\_xl シートは十分クリーンですか？

一番下までスクロールダウンしてみます。

23	2012	18133	1333	311	21	1893	13341	0.7700000	120	334	1041	
24	2013	18844	1420	540	12	1972	14562	0.7700000	135	344	1024	
25	2014	19336	1488	496	4	1988	15389	0.8000000	101	347	1330	
26	2015	20283	1993	401	3	2397	16316	0.8044175	110	398	1515	
27	NA	NA	NA	NA	NA	NA	NA	0.8000000	NA	NA	NA	
28	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	
29	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	
30	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	
31	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	
32	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	

まだクリーンではありません。たくさんの`NA`があります。

後で問題が発生するかもしれないので、今のうちに対処しましょう。

## NA の排除

NA を取り除く手っ取り早い方法は、1 列ごとに `NA` をサブセット化または除外することです。

`Year` 列を使用しましょう。

`subset()` と `filter()` の二つの方法があります。

### 1. ベース R

```
df_xl <- subset(df_xl, !is.na(Year))
```

### 2. dplyr

## dplyr をインストールしていなければ、以下の行のコメントを外して実行してください

```
# install.packages("dplyr") library(dplyr)
```

```
df_xl <- filter(df_xl, !is.na(Year))
```

違いは何でしょう？ dplyr は tidyverse パッケージの一部で、このコースの後半で取り上げます。使ってみましょう。

見てみましょう。

19	2008	16617	1426	363	180	1969	11365	0.6800000	127	282	1134	69	163	1
20	2009	16965	1360	397	94	1851	12350	0.7300000	130	320	1124	69	203	1
21	2010	17265	1401	400	80	1881	12541	0.7300000	141	318	1033	79	186	1
22	2011	17968	1358	415	8	1781	13421	0.7500000	138	366	1039	65	215	1
23	2012	18133	1333	511	21	1865	13941	0.7700000	128	354	1041	47	188	1
24	2013	18844	1420	540	12	1972	14562	0.7700000	135	344	1024	52	219	3
25	2014	19336	1488	496	4	1988	15389	0.8000000	101	347	1330	46	175	
26	2015	20283	1993	401	3	2397	16316	0.8044175	110	398	1515	60	178	1

`NAs`は全くありません。

数行のコードで、データは分析または視覚化するのに十分なほどクリーンアップされています。

## Excel へのエクスポート

オープンで他の人が開く際に有料のプログラムを要求しないために、データフレームは **CSV** として保存することをお勧めします。

しかし、そうしなければならない場合は、いくつかのチュートリアルがあります。

\* [xlsx パッケージを使う](<http://www.sthda.com/english/wiki/writing-data-from-r-to-excel-files-xls-xlsx>)

\* [Excel files を R で読み込んでインポートする](<https://www.datacamp.com/community/tutorials/r-tutorial-read-excel-into-r>)