IRR

George Kinnear

28/11/2020

IRR data

Read in the `data_rr.csv` file which contains the ratings by all raters during the various calibration phases.

```
irr_data = read.csv("data_irr.csv", header = TRUE, stringsAsFactors = FALSE) %>%
  mutate(
    Rater1 = str_trim(Rater1),
    Rater2 = str_trim(Rater2),
    Rater3 = str_trim(Rater3),
    Rater4 = str_trim(Rater4)
  )

irr_data %>%
  head() %>%
  kable()
```

Phase	Course	Assessment	Question	Marks	Rater1	Rater2	Rater3	Rater4	Agreed
Calibration1	DiagTest	Diagnostic Test	1	5	A3	A3	A3	A3	A3
Calibration1	DiagTest	Diagnostic Test	2	5	B2	B2	B2	B2	B1
Calibration1	DiagTest	Diagnostic Test	3a	2	A3	A3	A3	A3	A3
Calibration1	DiagTest	Diagnostic Test	3b	1	A2	A2	A2	A2	A2
Calibration1	DiagTest	Diagnostic Test	3c	2	A3	A3	A3	A3	A2
Calibration1	DiagTest	Diagnostic Test	4	5	A3	A3	A3	A3	A3

```
irr_data %>%
  group_by(Phase) %>%
  tally() %>%
  kable()
```

Phase	n
Calibration1	32
Calibration2	40
Calibration3	43
Calibration4A	18
Calibration4B	27
Calibration5	21
Post	35

Computing Krippendorf's alpha

Here we use the `irr` package.

```r
do_kripp_alpha_MATH <- function(df) {
  # df should be a tibble with one column per rater and their ratings in each row
  kripp.alpha(df %>%
                # replace A1 etc with unique integers
                mutate_all(funs(str_replace(., "A", "1"))) %>%
                mutate_all(funs(str_replace(., "B", "2"))) %>%
                mutate_all(funs(str_replace(., "C", "3"))) %>%
                mutate_all(as.numeric) %>%
                # transpose and convert to a matrix
                t %>%
                data.matrix)
}

kripp_alpha_of_phase <- function(phase) {
  ka = do_kripp_alpha_MATH(
    irr_data %>%
      filter(Phase == phase) %>%
      select(Rater1:Rater4) # some phases have only Rater1:Rater2 but this
                            #does not affect the value of kripp.alpha
  )
  return(ka$value)
}

kripp_alpha_of_phase("Calibration4A")
```

```
## Warning: `funs()` is deprecated as of dplyr 0.8.0.
## Please use a list of either functions or lambdas:
##
##   # Simple named list:
##   list(mean = mean, median = median)
##
##   # Auto named with `tibble::lst()`:
##   tibble::lst(mean, median)
##
##   # Using lambdas
##   list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```
## [1] 0.9195402
```

```r
do_kripp_alpha_MATH(
  irr_data %>%
    filter(Phase == "Calibration1") %>%
    select(Rater1:Rater4)
)
```

```
##  Krippendorff's alpha
##
##  Subjects = 32
##    Raters = 4
##     alpha = 0.741
```

```r
do_kripp_alpha_MATH(
  irr_data %>%
    filter(Phase == "Calibration2") %>%
    select(Rater1:Rater4)
)
```

```
##  Krippendorff's alpha
##
##  Subjects = 40
##    Raters = 4
##     alpha = 0.731
```

```r
do_kripp_alpha_MATH(
  irr_data %>%
    filter(Phase == "Calibration3") %>%
    select(Rater1:Rater4)
)
```

```
##  Krippendorff's alpha
##
##  Subjects = 43
##    Raters = 4
##     alpha = 0.925
```

```r
do_kripp_alpha_MATH(
  irr_data %>%
    filter(Phase == "Calibration4A") %>%
    select(Rater1:Rater2)
)
```

```
##  Krippendorff's alpha
##
##  Subjects = 18
##    Raters = 2
##     alpha = 0.92
```

```r
do_kripp_alpha_MATH(
  irr_data %>%
    filter(Phase == "Calibration4B") %>%
    select(Rater1:Rater4)
)
```

```
##  Krippendorff's alpha
##
##  Subjects = 27
##    Raters = 4
##     alpha = 0.946
```

```r
do_kripp_alpha_MATH(
  irr_data %>%
    filter(Phase == "Calibration5") %>%
    select(Rater1:Rater4)
)
```

```
##  Krippendorff's alpha
##
##  Subjects = 21
##    Raters = 4
##     alpha = 0.939
```

Analysing “Agreed” code

This computes the percentage of items which were recoded to a completely new “Agreed” code at the final stage, i.e. not a code that was selected by one of the raters in the original calibration phase.

It also adds Krippendorff alpha scores for each phase.

```
IRR_Results = irr_data %>%
  mutate(
    consensus = paste(Rater1,Rater2,Rater3,Rater4),
    existing_code = if_else(str_detect(consensus,Agreed),"yes","no")
  ) %>%
  # Count the number of times a new code was selected
  select(Phase,existing_code) %>%
  group_by(Phase,existing_code) %>%
  count() %>%
  # Compute the agreement stats
  group_by(Phase) %>%
  mutate(
    num_in_phase = sum(n),
    pc_of_phase = paste0(format(n / num_in_phase * 100, digits = 0), "%"),
    kripp = kripp_alpha_of_phase(Phase)
  ) %>%
  filter(existing_code == "yes") %>%
  arrange(existing_code) %>%
  select(Phase,num_in_phase,kripp,pc_of_phase) %>%
  ungroup() %>%
  mutate(
    Phase = str_replace(Phase, "Calibration", "")
  )

IRR_Results %>%
  knitr::kable(#format = "latex",
               col.names = c("Phase", "Number of questions", "Krippendorf's alpha", "Agreed code among
               digits = 2,
               booktabs = T)
```

Phase	Number of questions	Krippendorf’s alpha	Agreed code among original codes (%)
1	32	0.74	75%
2	40	0.73	88%
3	43	0.93	58%
4A	18	0.92	89%
4B	27	0.95	81%
5	21	0.94	100%

Using `irrCAC`

An alternative method, using code from http://www.agreestat.com which is referenced in:

Quarfoot, D., & Levine, R. A. (2016). How Robust Are Multirater Interrater Reliability Indices to Changes in Frequency Distribution? The American Statistician, 70(4), 373–384. https://doi.org/10.1080/00031305.2016.1141708

```
## Warning: package 'irrCAC' was built under R version 3.6.2
```

Calibration1

coeff.name	pa	pe	coeff.val	coeff.se	conf.int	p.value	w.name
AC1	0.8333333	0.1281982	0.80883	0.05278	(0.701,0.916)	0	unweighted
Krippendorff's Alpha	0.8346354	0.3590088	0.74202	0.06430	(0.611,0.873)	0	unweighted
Fleiss' Kappa	0.8333333	0.3590088	0.73999	0.06430	(0.609,0.871)	0	unweighted

Calibration2

coeff.name	pa	pe	coeff.val	coeff.se	conf.int	p.value	w.name
AC1	0.7916667	0.1544063	0.75362	0.05552	(0.641,0.866)	0	unweighted
Krippendorff's Alpha	0.7929688	0.2279687	0.73184	0.06126	(0.608,0.856)	0	unweighted
Fleiss' Kappa	0.7916667	0.2279687	0.73015	0.06126	(0.606,0.854)	0	unweighted

Calibration3

coeff.name	pa	pe	coeff.val	coeff.se	conf.int	p.value	w.name
AC1	0.9534884	0.1229854	0.94697	0.03723	(0.872,1)	0	unweighted
Krippendorff's Alpha	0.9540292	0.3850730	0.92524	0.05155	(0.821,1)	0	unweighted
Fleiss' Kappa	0.9534884	0.3850730	0.92436	0.05155	(0.82,1)	0	unweighted

Calibration4A

coeff.name	pa	pe	coeff.val	coeff.se	conf.int	p.value	w.name
AC1	0.9444444	0.1678241	0.93324	0.06729	(0.791,1)	0	unweighted
Krippendorff's Alpha	0.9459877	0.3287037	0.91954	0.08021	(0.75,1)	0	unweighted
Fleiss' Kappa	0.9444444	0.3287037	0.91724	0.08021	(0.748,1)	0	unweighted

Calibration4B

coeff.name	pa	pe	coeff.val	coeff.se	conf.int	p.value	w.name
AC1	0.9629630	0.1668381	0.95555	0.04467	(0.864,1)	0	unweighted
Krippendorff's Alpha	0.9636488	0.3326475	0.94553	0.05455	(0.833,1)	0	unweighted
Fleiss' Kappa	0.9629630	0.3326475	0.94450	0.05455	(0.832,1)	0	unweighted

Calibration5

coeff.name	pa	pe	coeff.val	coeff.se	conf.int	p.value	w.name
AC1	0.9523810	0.1556689	0.94360	0.03884	(0.863,1)	0	unweighted
Krippendorff's Alpha	0.9529478	0.2216553	0.93955	0.04245	(0.851,1)	0	unweighted
Fleiss' Kappa	0.9523810	0.2216553	0.93882	0.04245	(0.85,1)	0	unweighted

Checking the calculations by hand

The following computations give the values that go into computing the agreement coefficients by hand.

```
cal3 = irr_data %>%
  filter(Phase == "Calibration3") %>%
  select(Rater1:Rater2) %>%
  mutate(
    pair1 = paste0(Rater1,Rater2),
    pair2 = paste0(Rater2,Rater1)
  )

cal3_pairs = bind_rows(cal3 %>% transmute(pair=pair1), cal3 %>% transmute(pair = pair2))

cal3_pairs %>%
  group_by(pair) %>%
  tally()
```

```
## # A tibble: 10 x 2
##    pair      n
##    <chr> <int>
##  1 A2A2     24
##  2 A3A3     46
##  3 A3B1      1
##  4 B1A3      1
##  5 B1B1      2
##  6 B2B2      4
##  7 B2C1      1
##  8 C1B2      1
##  9 C1C1      4
## 10 C2C2      2
```

Checking cases where there was disagreement

This table shows all questions where there were disagreements between the coders during the calibration phases, along with the agreed code. This helps to see where the most common disagreements arise between particular pairs of codes.

```
irr_disagreement_cases = irr_data %>%
  gather(Rater1:Rater4, key = "rater", value = "code") %>%
  mutate(
    qid = paste(Assessment, Question)
  ) %>%
  select(Agreed, qid, code) %>%
  filter(!code == "") %>%
  group_by(qid, Agreed, code) %>%
  tally() %>%
  group_by(qid, Agreed) %>%
  mutate(
    disagree = n()>1
  ) %>%
  filter(disagree) %>%
  arrange(qid, code) %>%
  summarise(
    num_chosen = n(),
    codes = paste0(code, collapse = ",")
  ) %>%
  filter(num_chosen > 1)
```

```
## `summarise()` regrouping output by 'qid' (override with `.groups` argument)
```

```
irr_disagreement_cases %>% kable(booktabs = T)
```

qid	Agreed	num_chosen	codes
2017 P1 10	A3	2	A2,A3
2017 P1 11	A3	2	A3,B1
2017 P1 15	B1	2	A3,B1
2017 P1 16	B1	2	A2,A3
2017 P1 4	A3	2	A3,B1
2017 P1 5	B1	2	A2,A3
2017 P1 7	B1	2	B1,B2
2017 P1 8	B1	3	A2,A3,B2
2017 P2 11	B1	2	B1,B2
2017 P2 12	B1	2	B1,C1
2017 P2 14	B1	2	A3,B1
2017 P2 15	A3	2	B1,B2
2017 P2 16	A2	3	A2,C1,C2
2017 P2 6	A3	2	B1,B2
Diagnostic Test 10	C2	2	A2,A3
Diagnostic Test 14b	B1	2	A2,A3
Diagnostic Test 19	B1	2	B1,B2
Diagnostic Test 20	B2	2	A3,B2
Diagnostic Test 9	B1	2	A3,B1
Diagnostic Test Removed 8	A3	2	A3,B1
Exam Dec 2018 5b	B1	2	A2,B1
Exam Dec 2018 9a	A3	2	A3,B2
Online4 4	A2	2	A3,B1
Reading9B 2	A2	2	A2,B1
Written1 35	B1	2	A3,B1
Written1 52a	B2	2	A2,B2
Written1 52b	B2	2	A2,B2
Written2 17	B2	2	B2,C2
Written4 46	B2	2	B2,C1
Written8 Dec 2014 A4 2)	C1	2	B2,C1