DBHDS
Virginia Department of Behavioral Health and Developmental Services

Support Coordination Quality Review

Process Documentation

Updated: January 15, 2025

Overview

The Support Coordination Quality Review (SCQR) process was established to assess and improve the quality of support coordination (also referred to as "case management") services provided by community services boards (CSBs) to individuals with developmental disabilities (DD) who are receiving one of the home- and community-based services waivers (HCBS Waivers). The results of the SCQR are intended to inform improvements to the system overall.

All CSBs receive feedback and technical assistance from the Office of Community Quality Improvement (OCQI) and the Developmental Services Community Resource Consultants (CRCs).

The Director, Provider Development is the business owner for this project. Progress and results are reported to the Case Management Steering Committee, and a subcommittee called the SCQR Work Group meets regularly to oversee the project.

The Quality Research Specialist[1] in the Office of Quality Assurance and Healthcare Compliance contributes to the SCQR in the following ways:

- Provide guidance on methodology and question wording
- Draw a sample of records for CSB review and a subsample of records for the Look Behind
- Monitor CSB review completion
- Report quantitative results for all indicators and supplemental questions
- Analyze agreement between OCQI and CSB reviewers, and calculate interrater reliability

[1] Previously, the Office of Data Quality and Visualization served in this role.

- Create CSB-level reports to facilitate technical assistance
- Write the annual report

The SCQR runs on an annual cycle, with the final report completed in October of each calendar year.

FY2025 Timeline Update

In previous years, CSBs reviewed the previous calendar year and submitted responses between January 1 and July 15. Look Behind reviews were completed during July, August, and September. Analysis, report writing, and question revision occurred in the final quarter of the calendar year.

The lengthy process has always come with drawbacks, most notably a long delay between the records being completed and CSBs receiving feedback. The SCQR Work Group determined that there would be additional drawbacks in FY2025 because there will be a major update to the ISP in November 2024.

ISP 4.0 will change the way Support Coordinators assess and respond to risks. Specifically, the RAT will be embedded in the ISP. If we asked CSBs to review records from CY2024, about 80% of the sampled records would have used ISP 3.4, and the feedback would not be relevant to Support Coordinators' future work.

Therefore, to ensure that the FY2025 is a meaningful and relevant process for quality improvement, the SCQR Work Group proposed the following timeline:

- We will sample 400 records with an **ISP Effective Date between November 1, 2024 and March 15, 2025.**
- CSBs will review records from March 15, 2024 to March 15, 2025 (for example, when looking back on the previous 4 visits) but all of the records sampled will be those with ISP dates after November 1, using ISP 4.0.
- The CSB reviews will begin on April 1, 2025. The rest of the timeline will proceed as usual.

Instead of receiving technical assistance at the midpoint, all CSBs will receive technical assistance in January or February 2025 that covers the new ISP and revisions to the SCQR technical guidance, in addition to ongoing challenges identified by the previous SCQR.

January – February 2025	CSBs receive pre-SCQR technical assistance
March 15, 2024 - March 15 2025	CSBs complete records
April 1 - June 2025	SC supervisors review records for the SCQR
July – September 2025	OCQI conducts the Look Behind
October 2025	Final report

This timeline is expected to improve reliability because everyone will use the same ISP version for all questions, reducing confusion. Most importantly, CSBs will receive more timely and relevant feedback.

Population and Sample

Requesting the SCQR Population

Two weeks before CSB reviews begin, the business owner should request the population from which the SCQR sample will be drawn. The Division of Developmental Services runs a query to pull the population from WaMS.

The FY2025 query should be run between March 16 2025 and March 30 2025, and should use the following criteria:

- Adults and children[2]
- Enrolled in one of the three DD waivers on March 15 2024 in active, hold, or pending appeal status
- With an authorization for least one DD waiver service on March 15 2024 AND that individual was still enrolled as of the date (less one day or D-1) of the data pull with an authorization for at least one DD waiver service on D-1.
- With an ISP effective date after November 1, 2024
- Population data will include at least person's full name, DOB, Medicaid Number
- Please also include the CSB where they receive case management

These criteria should be submitted in a formal data request to the WaMS Data and Administration Manager in DDS. The spreadsheet should be forwarded to the Quality Research Specialist in time for the sample to be drawn by approximately April 1, 2025.

[2] The SCQR population was restricted to individuals aged 18 and older from FY2020 – FY2022. Children were added to the population in FY2023 and continue to be included moving forward.

DBHDS
Virginia Department of Behavioral Health and Developmental Services

Structure of the SCQR sample

The sample includes a minimum of 400 records stratified by CSB. [3]

Each CSB should review a minimum of five records (40 x 5 = 200). The remaining 200 records should be distributed according to population (Individuals assigned to CSB / Total individuals * 200).

If rounding each number at the 0.5 threshold results in a total that is more or fewer than 400, the rounding threshold is adjusted to get exactly 400.

The sample is drawn using the "sampling" package in R. To do this, you need a spreadsheet with a unique identifier for each individual in the population, the individual's CSB, and the number of individuals that should be sampled for the CSB.

ClientId	PersonId	CSB	Number_tosamp
XXXXXXXXXXXX	XXXXXXXXX	ALEXANDRIA COMMUNITY SERV BD	6
XXXXXXXXXXXX	XXXXXXXXX	ALEXANDRIA COMMUNITY SERV BD	6
XXXXXXXXXXXX	XXXXXXXXX	ALEXANDRIA COMMUNITY SERV BD	6
XXXXXXXXXXXX	XXXXXXXXX	ALEXANDRIA COMMUNITY SERV BD	6
XXXXXXXXXXXX	XXXXXXXXX	ALLEGHANY HIGHLANDS CSB	7
XXXXXXXXXXXX	XXXXXXXXX	ALLEGHANY HIGHLANDS CSB	7

The CSBs must be in alphabetical order.

In R, load the sampling and tidyverse packages and import the spreadsheet.

```
library(tidyverse)
library(sampling)

pop <- read.csv("C:\\Users\\qnq23447\\Documents\\Population to sample for CM
```

[3] In FY2023, an additional 79 children were included in an oversample for a total of 479 individuals. The purpose of this one-time oversample was to detect any large differences between children and adults. The sample returned to the typical number, 400, in FY2024.

```
look behind.csv")

# Make sure the number to sample is numeric

pop$Number_tosamp <- as.numeric(pop$Number_tosamp)
```

Create a small table with each CSB name and the number of records to sample. Add an additional 5 records per CSB for replacement cases.

```
strat <- pop %>%
group_by(CSB) %>%
summarize(Num=mean(Number_tosamp), Plusreplace=(mean(Number_tosamp) + 5))

strat <- as.data.frame(strat)
```

Draw the sample. The result will be a data frame that lists the row number of each sampled record in a variable called ID_unit. Subset the population to those row numbers only to obtain the sample (plus the replacement cases).

```
drawsample <- sampling:::strata(data=pop, stratanames="CSB",
size=strat$Plusreplace)

samp <- pop[drawsample$ID_unit,]
```

Designate 5 cases per CSB as replacement cases.

```
fiveeach <- rep(5, 40)

replace <- sampling:::strata(data=samp, stratanames="CSB", size=fiveeach)

samp$type <- "sampled"
samp[replace$ID_unit,]$type <- "replacement"
```

Sample distribution to CSBs

The following script splits the sample in half and saves two separate files for each CSB (one for the first half and one for the second half).

```
fullsamp <- subset(samp, samp$type==”sampled”) # remove replacement cases
fullsamp$half <- ceiling(fullsamp$FYTotal/2) # Divide in half and round up

CSBnames <- unique(fullsamp$CSB)  # Store CSB names

for i in (1:40) {
```

```
temp <- subset(fullsamp, fullsamp$CSB==CSBnames[i])
keep <- temp$half[1]
firsthalf <- temp[1:keep,]  # rows 1 through (half rounded up)

firsthalf$FYTotal <- NULL # Delete temporary variable
firsthalf$half <- NULL # Delete temporary variable

# Write CSV file for each CSB
write.csv(firsthalf, paste("First half", CSBnames[i], "SCQR FY20XX
sample.csv"))

keep <- temp$half[1] + 1

secondhalf <- temp[keep:nrow(temp),]  # remaining rows

secondhalf$FYTotal <- NULL # Delete temporary variable
secondhalf$half <- NULL # Delete temporary variable

# Write CSV file for each CSB
write.csv(secondhalf, paste("Second half", CSBnames[i], "SCQR FY20XX
sample.csv"))

}
```

The files are saved in a zip file and provided to the Director, Provider Development using a secure method. The DPD and team distribute the files to the CSBs.

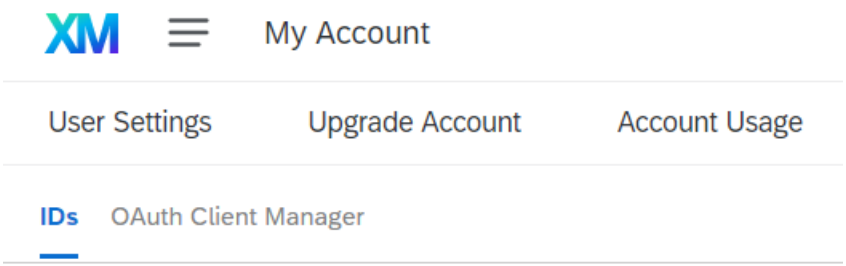
If the CSBs determine that a record cannot be reviewed, for example because the individual moved or passed away, CSBs are instructed to request a replacement case. The Quality Research Specialists maintains a master spreadsheet of cases in the sample, and add replacement cases as they are used.

Monitoring CSB Progress

Accessing responses in Qualtrics

Responses can be accessed directly in Qualtrics or by using the API to access the data through R. A unique API key can be obtained from the Qualtrics website by going to "Account" and then "Qualtrics IDs."

DBHDS
Virginia Department of Behavioral Health and Developmental Services



API instructions: Load the package "qualtRics" in R and store your credentials. You can use all_surveys() to see all the surveys on your account and find the ID for the SCQR. Then use the fetch_survey command with the appropriate ID.

```
library(qualtRics)
# API key and DBHDS base URL

qualtrics_api_credentials(api_key = "YOUR KEY HERE",
                          base_url = "virginiadbhds.az1.qualtrics.com",
                          install = TRUE,
                            overwrite=TRUE)

# See all surveys available in the directory
# Find the survey ID of the one you need

surveys <- all_surveys()

# Import the SCQR response data

sc <- fetch_survey(surveyID = "SURVEY ID HERE",
                   verbose = TRUE,
                    stringsAsFactors = FALSE,
                    force_request = TRUE,
                    convert = FALSE)


sc <- as.data.frame(sc)
```

Match completed responses to sample

Submitted responses need to be matched to the sample. However, there are often small errors such as spelling mistakes. Additionally, sometimes a CSB will complete a partial review and then start over, leading to duplicate responses. For these reasons, the Quality Research Specialist uses a script with fuzzy matching and logic to find matches and to choose between duplicates (which typically appear when a CSB entered a partial

response and then started over).

Matching instructions: In the code below, "sc" is the name of the data frame containing the responses and "samp" is the name of the data frame containing the sample.

```
# Make full name uppercase in responses and sample

sc$FIRST <- toupper(sc$Q4)
sc$LAST <- toupper(sc$Q5)

samp$FIRST <- toupper(samp$FirstName)
samp$LAST <- toupper(samp$LastName)

# Store first character of last name

sc$LAST1 <- substring(sc$LAST, 1, 1)
samp$LAST1 <- substring(samp$LAST, 1, 1)
```

Join all records by first character of last name. This usually catches them all, but on occasion you will have to manually correct a response where the first and last name are inverted (for example).

```
matchName<-left_join(samp, sc,
                     by=c("LAST1"="LAST1"),
                     match="all")
```

Calculate Levenshtein Distance for each identifier using the "RecordLinkage" package. Levenshtein distance calculates the difference between two text strings. It is the number of single-character edits required to change one string into another. A low score means the two strings are very similar, possibly indicating a typo.

```
library(RecordLinkage)

matchName <-
matchName %>%
  mutate(
       distanceFIRST = levenshteinDist(FIRST.x, FIRST.y),
       distanceLAST = levenshteinDist(LAST.x, LAST.y),
       distanceID = levenshteinDist(PersonId, Q3),
       distanceCSB = levenshteinDist(CSB_full, Q7)
       )
```

This code gives each potential match a score based on the number of full matches

(Levenshtein distance score = 0) and partial matches (Levenshtein distance = a low number) for each identifier.

```
matchName <-
matchName %>%
  mutate(
         firstfull=ifelse(distanceFIRST < 1, 2, 0),
         firstfuzzy=ifelse(distanceFIRST < 4, 1, 0),

         lastfull=ifelse(distanceLAST < 1, 2, 0),
          lastfuzzy=ifelse(distanceLAST < 2, 1, 0),

         IDfull=ifelse(distanceID < 1, 2, 0),
          IDfuzzy=ifelse(distanceID < 4, 1, 0),

         CSBfull=ifelse(distanceCSB < 1, 2, 0),
          CSBfuzzy=ifelse(distanceCSB < 4, 1, 0),

         matchscore = firstfull + lastfull + firstfuzzy + lastfuzzy + IDfull +
         IDfuzzy + CSBfull + CSBfuzzy
     )
```

Choose the match with the highest score for each response. Merge back into the "samp" data frame (containing the original sample).

```
matchName <- matchName %>%
  group_by(ResponseID) %>%
  slice(which.max(matchscore)) # Choose the one with the highest score

matchName <- as.data.frame(matchName)
matchName <- subset(matchName, !is.na(matchName$ResponseId))

# Merge status back into total sample

matchsmall <- data.frame(PersonId=matchName$PersonId, submitted="complete")
samp <- merge(samp, matchsmall, by="PersonId", all.x=TRUE, all.y=TRUE)
```

There may be duplicate responses if a CSB submitted a second review for the same person. Choose the response that is the most complete. In the code below, a response gets one point if the survey is 99% complete

```
key <- select(matchName, ClientId, ResponseId) # Small dataset

sc <- merge(sc, key, by="ResponseId", all.x=TRUE, all.y=FALSE)

sc$complete99 <- ifelse(sc$Q88=="Yes", 1, 0) # 99% complete
```

```
sc$complete100 <- ifelse(sc$Finished==1, 1, 0)  # Finished survey
sc$completescore <- sc$complete99 + sc$complete100

sc <- sc %>%
  group_by(ClientId) %>%
  slice(which.max(completescore)) # Choose the one with the highest score
```

If both responses are equally complete, choose the response that was completed at a later date.

```
sc <- sc %>%
  group_by(ClientId) %>%
  slice(which.max(StartDate)) # Choose the one with the highest score
```

Tabulating submitted responses

The first step for tabulating the responses is to match the submitted responses to the sample and eliminate duplicates. (See "Match completed responses to sample" above)

Then, compliance for each of the indicators must be determined. Some indicators come from just one question, but most involve multiple questions.
Instructions: Use the "mutate" function in the "tidyverse" package to create a variable for each indicator that will be 1 (Yes) or 0 (No) depending on the logic.

```
sc <- sc %>%
mutate(

# Indicator 1
## Compliance Criteria: Q24 must be "Yes"

  Ind01  = ifelse(Q24=="Yes", 1, 0),  # 1 if Q24 is Yes, otherwise 0

# Indicator 2
## Compliance Criteria: Both Q26_1 and Q26_2 must be "Yes"

  Ind02 = 1,  # Start by giving everyone a 1
  Ind02 = ifelse(is.na(Q26_1), 0,  # If Q26_1 is missing, change to 0
  ifelse(is.na(Q26_2), 0,          # If Q26_2 is missing, change to 0
  ifelse(Q26_1=="Yes" & Q26_2=="Yes", Ind02, 0))), # If both are not yes, 0
```

Code the logic for each indicator so that you have ten indicator variables, Ind01 – Ind10.

Determine how many indicators a record has in common:

```
sc <- sc %>%
  mutate(
       IndTotal = Ind01 + Ind02 + Ind03 + Ind04 + Ind05 + Ind06 + Ind07 + Ind08
+ Ind09 + Ind10
         )
```

The CMSC looks at how many records are in compliance on 86% of indicators (at least 9 out of 10). The following code calculates how many have 9+ indicators in compliance.

```
sc$Ind9plus <- ifelse(sc$IndTotal > 8, 1, 0) 9+ in compliance
```

Calculate the average indicator score per CSB:

```
CSBsum <- sc %>%
group_by(CSB=Q8) %>%
summarize(
       Responses = n(),
       Ind01 = mean(Ind01),
       Ind02 = mean(Ind02),
       Ind03 = mean(Ind03),
       Ind04 = mean(Ind04),
       Ind05 = mean(Ind05),
       Ind06 = mean(Ind06),
       Ind07 = mean(Ind07),
       Ind08 = mean(Ind08),
       Ind09 = mean(Ind09),
       Ind10 = mean(Ind10),
       Num_Records_in_Compliance = sum(Ind9plus),
       Num_Records_NOT_in_Compliance = Responses - sum(Ind9plus)
)
```

Calculate the average indicator score for the agency:

```
sumall <- sc %>%
summarize(
       Responses = n(),
       Ind01 = mean(Ind01),
       Ind02 = mean(Ind02),
       Ind03 = mean(Ind03),
       Ind04 = mean(Ind04),
       Ind05 = mean(Ind05),
       Ind06 = mean(Ind06),
       Ind07 = mean(Ind07),
       Ind08 = mean(Ind08),
       Ind09 = mean(Ind09),
       Ind10 = mean(Ind10)
)
```

Add a total row to the CSB-level report manually. For example:

```
CSBsum[41,] <- NA

CSBsum[41,]$CSB <- "TOTAL"
CSBsum[41,]$submitted <- sum(CSBsum$submitted[1:40])
CSBsum[41,]$not_submitted <- sum(CSBsum$not_submitted[1:40])

CSBsum[41,]$Ind01 <- sumall$Ind01
```

Then save the results to an excel spreadsheet to serve as the preliminary data report.

Additional Calculations

The SCQR calculates the number of *records* with either 9 or 10 indicators in compliance, both for the entire state and for each CSB. This calculation is done without adjusting based on agreement.

The SCQR also calculates the number of CSBs scoring less than 60% on two or more indicators, using only indicators with moderate or substantial agreement between CSBs and OCQI.

Look Behind (Retrospective Reviews)

Each year, QI specialists conduct independent reviews of 100 records that were sampled for the SCQR. The results are used to determine the validity and reliability of responses submitted by CSBs. Additionally, QI specialists review 50 interrater cases to ensure that their own scoring is consistent across reviewers.

Look Behind and Interrater Samples

The Look Behind sample is a sub-sample of the original sample that is reviewed by QI specialists. It is stratified by CSB. A minimum of 2 records (40 x 2 = 80) come from each CSB. The remaining 20 records should be distributed according to population.

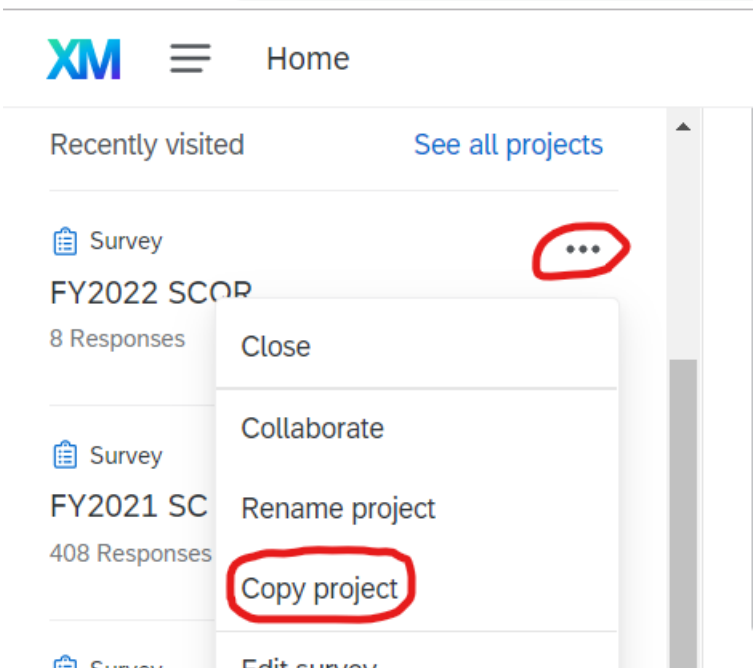
The threshold for rounding can be adjusted if the total is not exactly 100. Regions with more individuals will have a larger total Look Behind sample—this means specialist assigned to those regions will have more records to review.

Using the main sample of 400 records as the "population," draw the subsample using the same package in R that draws stratified samples.

For the interrater sample, each OCQI specialist should review 10 records from a different region. The specialists choose their own interrater cases, typically choosing CSBs that are conveniently located to minimize travel time. Each specialist will review exactly 10 cases for a total of 50 interrater cases.

Look Behind and Interrater Reviews in Qualtrics

A duplicate of the SCQR is created for the Look Behind and interrater reviews only, to keep them separate from the CSB reviews.



It should have a different name such as "FY20XX SCQR Look Behind and Interrater." There should also be text or a picture indicating that this is the link for OCQI to avoid confusion between the two links.

Matching Look Behind Reviews to CSB Reviews

Just as CSB reviews cannot be matched to the sample without accounting for typos and other discrepancies, Look Behind reviews typically cannot be matched to the

corresponding CSB reviews without some manual adjustments. Additionally, some of the responses entered into the same form in Qualtrics are interrater reviews which need to be excluded from the Look Behind and analyzed separately.

Instructions: OCQI keeps track of which records in the Look Behind sample are used for interrater reviews, and the name of the person completing the interrater reviews, in a spreadsheet. Compare the spreadsheet with the Excel export of the Look Behind and Interrater reviews and make a list of the Response IDs associated with the Look Behind reviews.

Use the Qualtrics API to import the Look Behind / IR results using the same process used to import the original SCQR responses. Match them to the sample using the same method used to match the original responses to the sample.

Then, subset the data frame to the response IDs associated with the Look Behind reviews.

```
x <- c([list of response IDs])

include <-  x
sc$check <- ifelse(sc$ResponseId %in% include, "Included", "Not included")
sc <- subset(sc, sc$check=="Included")
```

Run the script to match the SCQR responses to the sample and then subset the data frame to those included in the Look Behind subsample.

The Look Behind data frame and the subsetted SCQR data frame should both have exactly 100 responses. There should not be any duplicate responses, since the script matching them to the sample should have taken care of any records reviewed more than once.

The Look Behind and corresponding SCQR records can be merged based on the names of the individuals. Use the uppercase names created as part of the process that matches responses to the sample. This will eliminate differences caused by capitalization.

If any of the records from the CSBs and Look Behind fail to merge, there may be different spellings, nicknames, or errors causing them to not match exactly. Manually adjust the names until they match. Typically, there are about five responses that need to

be adjusted.

Follow the same process to match interrater reviews to the original reviews conducted by the QI specialist assigned to the region.

Calculating Percent Agreement and Maxwell's Random Error

For both the Look Behind and Interrater analyses, the percent agreement and Maxwell's Random error are calculated for each item on the SCQR, except for the individual's CSB and free-text responses.

Maxwell's random error coefficient (RE) adjusts for agreement expected by chance alone[4]. Since Maxwell's RE is for binary outcomes only, an extension proposed by Janes (1979) for questions with three or more possible outcomes is used.[5]

Cohen's kappa is a common interrater statistic, but it is not used for the SCQR because the kappa coefficient can be low when one of the outcomes is highly prevalent.[6] The goal is for the vast majority of records to be in compliance on all indicators, which would mean that the "Yes" response would be highly prevalent. In that case, Cohen's kappa would be low even in the event of strong agreement between raters.

Percent agreement: For each item, the percent agreement is simply the percentage of records in which both raters selected the same response. This can be calculated manually or with the "agree" function in the irr package in R.[7]

Maxwell's RE: If an item has only two possible responses, the "maxwell" function in the irr package may be used. However, many items have 3+ different possible outcomes, requiring the Janes (1979) extension.

Instructions: The Look Behind and Interrater scripts contain a function to calculate the

---

[4] Maxwell, A. E. (1977) Coefficients of agreement between observers and their interpretation. *British Journal of Psychiatry* **130**, 79-83.
[5] Janes, C. L. (1979) An extension of the random error coefficient of agreement to NxN tables. *British Journal of Psychiatry* **134**, 617-19.
[6] Feng, G. C. (2013) Factors affecting intercoder reliability: a Monte Carlo experiment. *Quality & Quantity* **47**, 2959–2982.

[7] Documentation for the IRR package: https://cran.r-project.org/web/packages/irr/irr.pdf

coefficient using the extension. This is the beginning of the function.

```
maxwellNxN <- function(x, y){
```

The function determines the number of response categories (N) and then constructs an NxN matrix with one cell for each possible combination of responses. The cells are filled with counts for the number of times that each combination appears in the data. The matrix is used to calculate the RE coefficient using the formula in Janes (1979).

The input for the function should be two columns of data, one containing the first rater's responses and the other containing the second rater's responses. After setting up the data for the look behind or interrater analyses, each variable will end with .x (rater 1) or .y (rater 2). To run the function for Q100, for example, you would input:

```
maxwellNxN(sc$Q100.x, sc$Q100.y)
```

The scripts contain a loop that calculates percent agreement and the RE coefficient for a list of variables. Check and modify the list so that it contains every quantitative item in the SCQR. The output will be a data frame containing 3 columns: the variable name, the percent agreement, and the RE coefficient.

When agreement is perfect (100% agreement) the irr package may return NaN. This can be changed to a value of RE = 1.

Interpreting Percent Agreement and Maxwell's RE

The percent agreement can be misleading because some agreement is expected due to chance alone. For example, for an item with 2 response options, agreement is expected 50% of the time by chance alone. This is the reason for reporting the RE coefficient alongside percent agreement.

The Maxwell RE coefficient ranges from -1 (perfect disagreement) to 0 (no agreement beyond what is expected by chance) to 1 (perfect agreement). Scores in between those values can be interpreted on a spectrum; cutoff scores are arbitrary, as there is no consequential difference between a value of 0.599 and 0.600, for example. However, for easier interpretation, scores were coded with the following color scheme:

No agreement	< 0

Weak agreement	0.00 to 0.39
Moderate agreement	0.40 to 0.59
Substantial agreement	0.60 to 1

Negative values will occur if agreement is worse than expected by chance alone. This can happen, for example, if there is an item where the CSBs say "Yes" on the majority of records and OCQI says "No" on the majority of records.
The probability of agreement expected by chance alone varies depending on the number of response categories. For an intuitive example, think about two raters flipping coins verses two raters each rolling a six-sided dice. Agreement is less likely to occur by chance when rolling dice; therefore, if raters get the same result over and over it's likely that something is going on besides chance.

For this reason, two different questions on the SCQR can have the same percent agreement but different RE coefficients, due to having different numbers of response options.

Generating PDF reports for technical assistance

PDF reports are generated in R Markdown with LaTeX to facilitate technical assistance with CSBs. There are scripts for two types of reports: One showing all CSB responses broken down by individual ("CSB report") and one showing OCQI responses alongside CSB responses for the cases sampled for the Look Behind ("CSB Look Behind Report").

Instructions: The scripts use a parameter for the CSB name, so that you can enter the CSB in one spot to make a custom report for that CSB. Change the CSB in this part of the header:

```
knit: (
  function(inputFile, encoding) {

    mycsb <- 'Western Tidewater Community Services Board'

    rmarkdown::render(
      input       = inputFile,
      encoding    = encoding,
      params      = list(csb = mycsb),
      output_file = paste0('SCQR FY2022 with names - ', mycsb, '.pdf'))})
```

The "output_file" gives the pdf file a name that includes the CSB name. The CSB name

also appears on the title of the document.

```
---
title: "SCQR Data Report"
params:
  csb:
    value: !r "Whatever"
author: "`r params$csb`"
```

To generate reports for all 40 CSBs, manually enter one CSB name at a time and then knit the report by clicking on the icon with a small ball of yarn.

Each chunk of code begins with ``` and a unique name. "include=TRUE" means that the output will be included in the document. "echo=FALSE" means that the R code will not appear in the document.

```
```{r Q12, include=TRUE, echo=FALSE}
```

Each chunk of code ends with ```

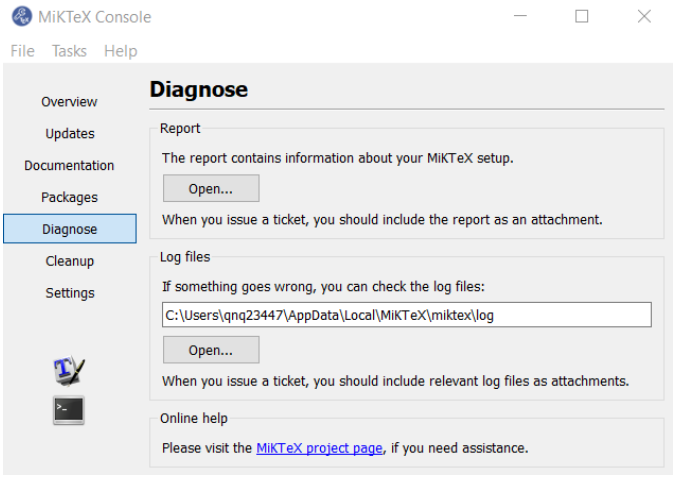
If there are any errors in the R code, the R Markdown script will stop running (this is different from a typical R script where it will print the error message and move to the next line). If you run the code and get an error, check the output to determine which line caused a problem.

```
Line 39 Error in library(qualtRicXX) : there is no package called 'qualtRicXX'
        > withVisible -> eval -> eval -> library In addition: Warning message:
        knit_params_handlers(evaluate = evaluate), : an error occurred when han
```

In this example, the error occurred in the block of code beginning on line 39. The code called for a package named qualtRicxx which does not exist.

MikTeX installation

The R Markdown scripts require MikTeX (https://miktex.org/), which uses code to render documents.
```



After installation, MikTeX will require additional packages to run the R Markdown script. Before attempting to run the script, go to the MikTeX console and click "Settings" and then check that packages should be installed automatically, on the fly. Then attempt to run the script. Most of the packages will install as needed, but often some of them will need to be installed manually.

To figure out which packages need to be manually installed, go to the MikTeX console and click "Diagnose" and then check out the file called "miktex-pdftex.log"

This will tell you which packages failed to install. Click "Packages" in the console, manually search for the missing packages, and install them. This process may need to be repeated multiple times before the R Markdown script works.

R Package kableExtra

The R Markdown scripts also use an R package called kableExtra. The documentation for this package is here:

https://cran.r-project.org/web/packages/kableExtra/vignettes/awesome_table_in_pdf.pdf

The code makes small data frames containing only the variables to be used in the table. Here, the data frame "dtemp" contains the names of the individuals and the responses to question 29. Then, it removes the row numbers so that they won't appear in the table.

```
dtemp <- (data.frame(sc$rname, question=sc$Q29))
rownames(dtemp) <- NULL
```

This command formats the table using the "booktabs" style. "Longtable" allows the table to continue on the next page if necessary. "col.names" assigns names to each of the columns, while the "align" argument changes the alignment of the text, in this case to left (for the first column) and centered (for the second column). The last line adds shading in a striped pattern to make the table easier to read.

```
kable(dtemp, "latex", booktabs = T, longtable=TRUE, col.names = c("Name",
"Response"), align=c("l", "c")) %>%
      kable_styling(latex_options =  c("striped"))
```

For the free text responses, it is necessary to specify column widths. In this example, the first column (containing individual names) is three centimeters, and the column with the text responses is 13 centimeters. The code for free text responses also deletes any rows with missing values to reduce clutter on the page.

```
# PLEASE EXPLAIN

dftemp <- na.omit(data.frame(sc$rname, sc$Q53)) # delete blank rows
rownames(dftemp) <- NULL

kable(dftemp, "latex", booktabs = T, longtable=TRUE, col.names = c("Name",
"Please explain")) %>%
 column_spec(1, width = "3cm")  %>%
   column_spec(2, width = "13cm")  %>%
    kable_styling(latex_options = "striped")
```

In addition to listing the response by individual, the script prints the percentage of records that fell into each response category.

```
formattable::percent(prop.table(table(sc$Q47, exclude=NULL)))
```

The "formattable::percent" command changes decimals such as 0.95 into percentages such as 95%.

Charts in ggplot2

The scripts display the results for the 10 indicators in charts generated using the ggplot2 package in R. These charts are not essential, but they add some color and make the output more visually pleasing. For each indicator, the code generates a factor with two options ("Met" or "Not met") to used in the charts.

```
sc$Ind1fac <- NA
sc[sc$Ind01==0,]$Ind1fac <- "Not met"
sc[sc$Ind01==1,]$Ind1fac <- "Met"
sc$Ind1fac <- factor(sc$Ind1fac,levels=c("Not met","Met"))
```

The dimensions are established in the code chunk options.

```
```{r Ind1, fig.width=7, fig.height=0.5, fig.align="center", include=TRUE,
echo=FALSE, comment=NA, results="asis"}
```

This is an example of the ggplot2 code. For documentation of each of these options, see: https://ggplot2.tidyverse.org/

```
sc %>%
  count(Ind1fac) %>%
  mutate(pct = n /sum(n))%>%
  ggplot(aes(x = "", y = pct, fill = Ind1fac)) +
  coord_flip() +
  geom_col(width=0.2) +
  scale_x_discrete(expand=c(0,0)) +
  scale_y_discrete(expand=c(0,0)) +
  scale_fill_manual(values=c("Not met" = "#c2c2a3", "Met" = "#C6E2EF")) +
  geom_text(aes(label = paste0(round(pct * 100), '%')),
            position = position_stack(vjust = 0.5)) +
  theme_void() +
  theme(legend.title = element_blank())    +
  guides(fill = guide_legend(reverse = TRUE))
```

The color can be changed on the scale_fill_manual line. Any hex code will work. (Find the hex codes for colors on many websites, for example: https://htmlcolorcodes.com/)

Look Behind PDF Reports

The script for Look Behind PDF reports places the CSB and OCQI responses side-by-side so that discrepancies can be easily spotted and discussed. The script runs the code to match the CSB responses to the Look Behind responses and merge them (see previous section). The table contains both responses:

```
dtemp <- (data.frame(Name=sc$rname, QI=sc$Q81_2.x, CSB=sc$Q81_2.y))  #Q

rownames(dtemp) <- NULL

dtemp %>%
```
```

```
kable(escape = F, "latex", booktabs = T, longtable=TRUE) %>%
     kable_styling(latex_options = "striped")
```

The output will display 3 columns: Name, QI, and CSB. QI = the QI specialist responses and CSB = the corresponding CSB responses.

Final Report

The Quality Research Specialist prepares the final report. It is due in the middle of October, or if the Look Behind takes longer than expected, one month after the Look Behind and interrater reviews are complete. The report includes, at minimum:

- Purpose of the SCQR and brief background
- Criteria for inclusion in the population
- Sampling method and number sampled for SCQQ and Look Behind
- Quantitative results for all indicators and supplemental questions
- Look Behind and interrater results (% agreement and Maxwell's RE)
- Qualitative explanations of findings
- Discussion of any observed bias or reliability issues
- Comparisons with previous years, when applicable

For each indicator, the results should be broken down by the individual items that comprise the indicator, and should include the percent agreement and RE value for both the Look Behind and interrater analyses.

Sometimes, the number of records meeting the indicator is lower than the number that meet the individual items, as in the above example. This can occur when marking either item as "No" will result in the indicator not being met.

On other indicators, the number of records scored as meeting an indicator can be higher than the number of records marked "Yes" on individual items. This happens when an indicator can be scored as "met" for different reasons. For example, one reviewer might indicate that no disagreement occurred during the ISP meeting, while another could indicate that a disagreement did occur but was resolved. In this case, they would both say the indicator was "met" but they would not agree on the items comprising the indicator.

DBHDS
Virginia Department of Behavioral Health and Developmental Services

Explaining Look Behind results

When the Look Behind analysis reveals low agreement on an indicator, it could mean that CSBs were more likely to score an indicator as met *or* that QI specialists were more likely to score an indicator as met. Sometimes, there are about the same number of disagreements in both directions.

A 2x2 table showing the breakdown can clarify what happened:

	CSB met	CSB not met
QI met	70	5
QI not met	**16**	9

In the above example, CSBs and QI specialists agreed on 70+9=79 records. On 5 records, the QI specialists scored the indicator as "met" while CSBs did not. On 16 records, the CSBs scored the indicator as "met" while QI specialists did not.

The most concerning disagreement is found in the bottom left cell, where CSBs said the indicator was met and QI specialists disagreed. A high number in this cell casts doubt on the validity of the self-reported CSB results, particularly if the self-reported results show 86% of records (or more) meeting the indicator.

When disagreement is high, the free text answers explaining "No" responses can be used to determine the source of the disagreement.

Technical Assistance

Collaboration

DBHDS
Virginia Department of Behavioral Health and Developmental Services

The Office of Provider Network Supports (OPNS) and the Office of Community Quality Improvement (OCQI), in collaboration with the Case Management Steering Committee (CMSC), annually assess any trends with scoring or results of the Support Coordination Quality Reviews (SCQR). Prior to the start of the next SCQR round, these offices work together to provide technical assistance related to the findings of the previous year and any revisions taking place.

Office of Provider Network Supports

Community Resource Consultants (CRCs) within the Office of Provider Network Supports meet with Community Service Boards (CSBs), virtually or in person. This technical assistance meeting is provided to all CSBs prior to the start of the SCQR round. It is designed to accomplish the following:

- Review technical guidance document focusing on any revisions made from the previous year
- Discuss SCQR indicator questions that were not scored in high agreement
- Emphasize the importance of following the technical guidance document to ensure inter-rater reliability
- Review the timeline and expectations of SCQR participation and completion
- Collect feedback regarding the SCQR survey, technical guidance, and process, as applicable

Office of Community Quality Improvement

Quality Improvement Specialists within the Office of Community Quality Improvement meet with CSBs once all submissions have occurred and the Look Behind process is completed, generally at an exit meeting. This technical assistance meeting is provided to all CSBs. It is designed to accomplish the following:

- Summarize findings from the retrospective reviews (RR)
- Discuss areas of disagreement between the CSB score and the OCQI QI Specialist's RR results
- Discuss trends/patterns (2 or more records) that were met/not met in the CSB SCQR results for the 10 elements and PMIs currently monitored by the CMSC
- Discuss CSB improvement efforts related to underperforming measures

- Provide feedback and additional reference materials as indicated to CSB case management supervisor(s) to increase the reliability of future reviews (such as clarifying SCQR questions, protocol or guidance)

The Office of Community Quality Improvement also conducts a statewide virtual meeting to include the Office of Provider Network Supports and all CSBs following the completion of the SCQR round and the final report. This call is designed to:

- Present and review statewide aggregate data in comparison to prior year results
- Discuss the use of the SCQR as an integral part of the CSB quality improvement processes
- Gather any additional feedback for improvement from all parties involved in SCQR

Revisions and Updates

After the SCQR administration, report, and technical assistance are complete, the SCQR Work Group meets to discuss revisions to the questions and/or the technical guidance. Changes may be made to clarify expectations with the goal of increasing reliability, and/or to reflect changes to DBHDS processes and requirements.