quicR: An R Package for Real-Time Quaking Induced Conversion (RT-QuIC) Assays

true Peter Larsen

Real-time quaking induced conversion (RT-QuIC) has quickly become an emerging diagnostic tool for protein misfolding disorders such as Creutzfeldt-Jakob disease and Parkinson's disease. Given that the technology is still relatively new, academic and industry standards have yet to be established. 'quicR' was developed to fill this lack of standardization by providing functions for both data curation and analysis.

Introduction

Real-time quaking induced conversion is a diagnostic assay that converts recombinant protein substrate into a misfolded aggregate in the presence of a misfolded seed (Atarashi et al. 2011).

While standard metrics for determining a diagnosis have not been universally established, there are certain metrics that many research groups have found useful. These include time-to-threshold (TtT)(Orrú et al. 2015), rate of amyloid formation (RAF)(Gallups and Harms 2022), maxpoint ratio (MPR)(Rowden et al. 2023), and maximum slope (MS)(Henderson et al. 2015). All together, these metrics provide insight into the general kinetics of an RT-QuIC reaction, and can be used together to draw a more robust diagnostic decision.

Methods

This package requires the following dependencies: dplyr, ggplot2, janitor, openxlsx, readxl, reshape2, slider, stats, stringr, tidyr. Because the MARS software (BMG Labtech, Ortenberg, Germany) exports data as an Excel workbook, the packages, openxlsx and readxl, were fundamental to performing downstream handling. The tidyverse packages (dplyr, ggplot2, stringr, and tidyr), were vital for writing easy-to-read code and for data visualization. The janitor package has useful functions for performing quick data cleaning

Development

The quicR package was developed to address the need for efficient data conversion and analysis of RT-QuIC data. The functions were designed with usability and reproducibility in mind, ensuring compatibility between multiple labs. Currently, the package accepts data exported from the MARS software as an Excel workbook.

The functionality in this package revolves around data curation, metric calculations, and visualization.

Implementation

Input of Sample IDs into MARS

MARS allows input of a TXT file containing sample IDs, dilution factors, and their well locations. This file is uniquely formatted, and not easily reproduced manually. The function "BMG_format" allows for input of a CSV file containing the plate layout, and exports the formatted TXT file.

Example Plate Layout

	1	2	3	4	5	6	7	8	9	10	11	12
A	N	S01	S02	S03	S04	S05	S06	S07	S08	S09	S10	S11
B	N	S01	S02	S03	S04	S05	S06	S07	S08	S09	S10	S11
C	N	S01	S02	S03	S04	S05	S06	S07	S08	S09	S10	S11
D	N	S01	S02	S03	S04	S05	S06	S07	S08	S09	S10	S11
E	P	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22
F	P	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22
G	P	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22
H	P	S12	S13	S14	S15	S16	S17	S18	S19	S20	S21	S22

Formatted Plate Layout for MARS Input

The function, "BMG_format", includes the logical argument "write_file". If true, it will create a TXT file. The path can be given to the "save_path" argument, and the file name can be supplied to the "save_name" argument.

```
BMG_format(sample_file, write_file = FALSE)[1:18] %>%
  as.data.frame()
```

```
                .
1    A1  N      N
2    B1  N      N
3    C1  N      N
4    D1  N      N
5    E1  P      P
6    F1  P      P
7    G1  P      P
8    H1  P      P
9   A2  X1     S01
10  B2  X1     S01
11  C2  X1     S01
12  D2  X1     S01
13  E2  X2     S12
14  F2  X2     S12
15  G2  X2     S12
16  H2  X2     S12
17  A3  X3     S02
18  B3  X3     S02
```

Data Cleaning and Transformation

The MARS software (BMG Labtech, Ortenberg, Germany) exports real-time data as an Excel workbook. Typically, the first sheet in the workbook will include microplate views of data. The two tables that are most relevant on this sheet are the "Sample IDs" table and the "Dilutions" table (if included). For much of the downstream analysis, it is crucial the the "Sample IDs" table was exported. If there is no table, the user can simply add it manually.

Retrieving Metadata

The metadata is defined as either sample-dependent or -independent. Sample-dependent metadata includes information such as sample IDs and dilution factors, whereas sample-independent metadata

includes the date, time, reaction ID, etc.

Sample-Dependent Metadata

The dependent metadata can be retrieved using the "organize_tables" and "convert_tables" functions. The former returns a list of tables, and the ladder converts each table into a column in a single data frame.

```
tabs <- organize_tables(file)[-1]
tabs
```

```
$`Sample IDs`
# A tibble: 8 x 12
  ...1  ...2  ...3  ...4  ...5  ...6   ...7  ...8  ...9 ...10 ...11 ...12
  <chr> <chr> <chr> <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1 P     P1    P1    P1    P3    P3       NA    NA    NA    NA    NA    NA
2 P     P1    P1    P1    P3    P3       NA    NA    NA    NA    NA    NA
3 P     P1    P1    P1    P3    P3       NA    NA    NA    NA    NA    NA
4 P     P1    P1    P1    P3    P3       NA    NA    NA    NA    NA    NA
5 N     P1    P1    P1    P3    P3       NA    NA    NA    NA    NA    NA
6 N     P1    P1    P1    P3    P3       NA    NA    NA    NA    NA    NA
7 N     P1    P1    P1    P3    P3       NA    NA    NA    NA    NA    NA
8 N     P1    P1    P1    P3    P3       NA    NA    NA    NA    NA    NA

$Dilutions
# A tibble: 8 x 12
  ...1  ...2  ...3  ...4  ...5  ...6   ...7  ...8  ...9 ...10 ...11 ...12
  <chr> <chr> <chr> <chr> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1 1000  1000  1000  10000 1000  1000     NA    NA    NA    NA    NA    NA
2 1000  1000  1000  10000 1000  1000     NA    NA    NA    NA    NA    NA
3 1000  1000  1000  10000 1000  1000     NA    NA    NA    NA    NA    NA
4 1000  1000  1000  10000 1000  1000     NA    NA    NA    NA    NA    NA
5 1000  1000  1000  10000 1000  1000     NA    NA    NA    NA    NA    NA
6 1000  1000  1000  10000 1000  1000     NA    NA    NA    NA    NA    NA
7 1000  1000  1000  10000 1000  1000     NA    NA    NA    NA    NA    NA
8 1000  1000  1000  10000 1000  1000     NA    NA    NA    NA    NA    NA
```

```
tabs <- convert_tables(tabs)
tabs %>% head(15) %>% kable(row.names = FALSE)
```

Sample IDs	Dilutions
P	1000
P1	1000
P1	1000
P1	10000
P3	1000
P3	1000
NA	NA
NA	NA
NA	NA
NA	NA
NA	NA
NA	NA
P	1000
P1	1000
P1	1000

Sample-Independent Metadata

The independent metadata can be retrieved using the "get_meta" function. This data is included in the header of the excel workbook.

```
get_meta(file) %>% as.data.frame()
```

```
            Meta_ID                                          Meta_info
1              User                                               USER
2              Path  C:\\Program Files (x86)\\BMG\\Omega\\User\\Data
3           Test ID                                                236
4         Test Name                                RT-QuIC Plate Mode
5              Date                                          8/8/2024
6              Time                                         2:58:07 PM
7               ID1                      20240808_r5_GR_scrapie_RT
8 Fluorescence (FI)                                             <NA>
```

Retrieving and Manipulating Raw Data

The raw data is typically found on the second sheet of the Excel workbook. The raw real-time data can be retrieved using the "get_real" function. The logical argument, "ordered", indicates whether the user would prefer the columns to be ordered by well or by sample ID. By default, it is FALSE which will order the data by well. This should almost always be the case for easier integration with other downstream

functions. Additionally, since there can be more than one instance of real-time data (depending on if the user added some calculations in MARS), "get_real" returns a list of dataframes. Therefore, the output should be indexed to access the data frame of interest.

Retrieve Raw Data

```
df_ <- get_real(file, ordered = FALSE)[[1]] %>% as.data.frame()

df_[1:15, 1:6] %>% kable(row.names = FALSE)
```

Time	positive_control_p	sample_x1	sample_x1_2	sample_x2	sample_x3
0	611	599	564	588	597
0.75	524	513	475	482	542
1.5	576	545	515	509	565
2.25	609	559	533	535	591
3	615	570	540	547	602
3.75	623	576	552	556	595
4.5	623	574	559	561	604
5.25	617	588	562	579	623
6	637	590	563	566	609
6.75	619	585	573	569	616
7.5	625	591	562	565	619
8.25	631	590	572	565	623
9	632	601	577	563	613
9.75	624	594	569	560	628
10.5	597	593	562	555	631

Transpose Raw Data

This data is structured such that each sample is its own column as well as the time points. This format can be non-ideal, and should be transposed for downstream manipulation using the function, "transpose_real". After transposition, each time point has its own column, and there is a single column for every sample ID.

```
transpose_real(df_)[1:15, 1:13] %>% kable(row.names = FALSE)
```

Sample IDs	0	0.75	1.5	2.25	3	3.75	4.5	5.25	6	6.75	7.5	8.25
positive_control_p	611	524	576	609	615	623	623	617	637	619	625	631

Sample IDs	0	0.75	1.5	2.25	3	3.75	4.5	5.25	6	6.75	7.5	8.25
sample_x1	599	513	545	559	570	576	574	588	590	585	591	590
sample_x1_2	564	475	515	533	540	552	559	562	563	573	562	572
sample_x2	588	482	509	535	547	556	561	579	566	569	565	565
sample_x3	597	542	565	591	602	595	604	623	609	616	619	623
sample_x3_2	571	524	542	564	568	568	583	587	594	593	586	590
positive_control_p_2	594	540	582	622	628	643	642	628	640	652	647	642
sample_x1_3	594	510	536	557	565	568	562	565	552	573	566	565
sample_x1_4	571	493	520	529	549	556	554	563	567	570	570	566
sample_x2_2	609	509	513	529	550	544	553	567	566	557	573	573
sample_x3_3	686	642	678	691	708	709	721	741	745	752	745	750
sample_x3_4	588	509	554	568	585	592	592	598	604	619	615	620
positive_control_p_3	597	515	580	600	620	619	624	611	624	622	593	592
sample_x1_5	596	516	548	558	576	568	575	576	579	586	582	570
sample_x1_6	575	492	524	539	556	554	562	574	575	583	576	573

Normalize Raw Data

The function “normalize_RFU” will convert the raw data into a background normalized data set. The function includes two additional arguments, “bg_cycle” (the cycle which will be used as the background fluorescence value) and “transposed” (if FALSE, will make a call to the “transpose_real” function).

```
df_norm <- normalize_RFU(df_, bg_cycle = 4, transposed = FALSE)

df_norm[1:15, 1:12] %>%
  mutate_at(2:ncol(.), ~round(as.numeric(.), 2)) %>%
  kable(row.names = FALSE)
```

Sample IDs	0	0.75	1.5	2.25	3	3.75	4.5	5.25	6	6.75	7.5
positive_control_p	1.00	0.86	0.95	1	1.01	1.02	1.02	1.01	1.05	1.02	1.03
sample_x1	1.07	0.92	0.97	1	1.02	1.03	1.03	1.05	1.06	1.05	1.06
sample_x1_2	1.06	0.89	0.97	1	1.01	1.04	1.05	1.05	1.06	1.08	1.05
sample_x2	1.10	0.90	0.95	1	1.02	1.04	1.05	1.08	1.06	1.06	1.06
sample_x3	1.01	0.92	0.96	1	1.02	1.01	1.02	1.05	1.03	1.04	1.05
sample_x3_2	1.01	0.93	0.96	1	1.01	1.01	1.03	1.04	1.05	1.05	1.04
positive_control_p_2	0.95	0.87	0.94	1	1.01	1.03	1.03	1.01	1.03	1.05	1.04
sample_x1_3	1.07	0.92	0.96	1	1.01	1.02	1.01	1.01	0.99	1.03	1.02
sample_x1_4	1.08	0.93	0.98	1	1.04	1.05	1.05	1.06	1.07	1.08	1.08
sample_x2_2	1.15	0.96	0.97	1	1.04	1.03	1.05	1.07	1.07	1.05	1.08

Sample IDs	0	0.75	1.5	2.25	3	3.75	4.5	5.25	6	6.75	7.5
sample_x3_3	0.99	0.93	0.98	1	1.02	1.03	1.04	1.07	1.08	1.09	1.08
sample_x3_4	1.04	0.90	0.98	1	1.03	1.04	1.04	1.05	1.06	1.09	1.08
positive_control_p_3	1.00	0.86	0.97	1	1.03	1.03	1.04	1.02	1.04	1.04	0.99
sample_x1_5	1.07	0.92	0.98	1	1.03	1.02	1.03	1.03	1.04	1.05	1.04
sample_x1_6	1.07	0.91	0.97	1	1.03	1.03	1.04	1.06	1.07	1.08	1.07

The reader will notice that the fourth time column is all “1’s” since this was designated the background cycle.

Calculations

There are three analytical metrics with dedicated functions: time-to-threshold, maxpoint ratio, and maximum slope. The rate of amyloid formation does not have a designated function since it is simply the inverse of the time-to-threshold. Each function below accepts input from the “transpose_real” or the “normalize_RFU” functions.

Time-to-threshold

Time-to-threshold is calculated using the “calculate_TtT” function. The function must be supplied a threshold; default value is 2 (i.e. twice the background fluorescence if the data is normalized). A starting column should also be given as an integer; default value is 3. This is essentially asking how many columns of metadata are included before the fluorescence reads begin.

Time-to-threshold is calculated by iterating through each row and checking if a value is greater than the threshold. If the value is greater, the slope of the previous timepoint to the current timepoint is calculated, and the time intersection of the current read is returned.

Maxpoint Ratio

Maxpoint ratios (MPR) are calculated by the “calculate_MPR” function. Data must be normalized in order to derive this metric. In a normalized data set, the MPR is simply the maximum value achieved during the run.

Maximum Slope

The Maximum Slope (MS) is calculated by the "calculate_MS" function. The data can be either raw or normalized. The function iterates through each row using a rolling window which can be adjusted (default value is 3). Given the window size, the slope is calcualated based on the range of the window. The MS is simply the largest slope value recorded.

```
samples <- tabs$`Sample IDs` %>% na.omit()
dilutions <- tabs$Dilutions %>% na.omit() %>% as.numeric()

# Define the number of hours that the rxn ran for.
hours <- as.numeric(colnames(df_norm)[ncol(df_norm)])

df_analyzed <- data.frame("Sample IDs" = samples, check.names = FALSE) %>%
  mutate(
    Dilutions = -log10(dilutions),
    # Maxpoint Ratio
    MPR = calculate_MPR(df_norm, start_col = 3, data_is_norm = TRUE),
    # Max Slope
    MS = calculate_MS(df_norm, data_is_norm = TRUE),
    # Time to Threshold
    TtT = calculate_TtT(df_norm, threshold = 2, start_col = 3),
    # Rate of Amyloid Formation
    RAF = ifelse(TtT == hours, 0, 1 / TtT)
  )

df_analyzed%>%
  head(15) %>%
  kable(row.names = FALSE)
```

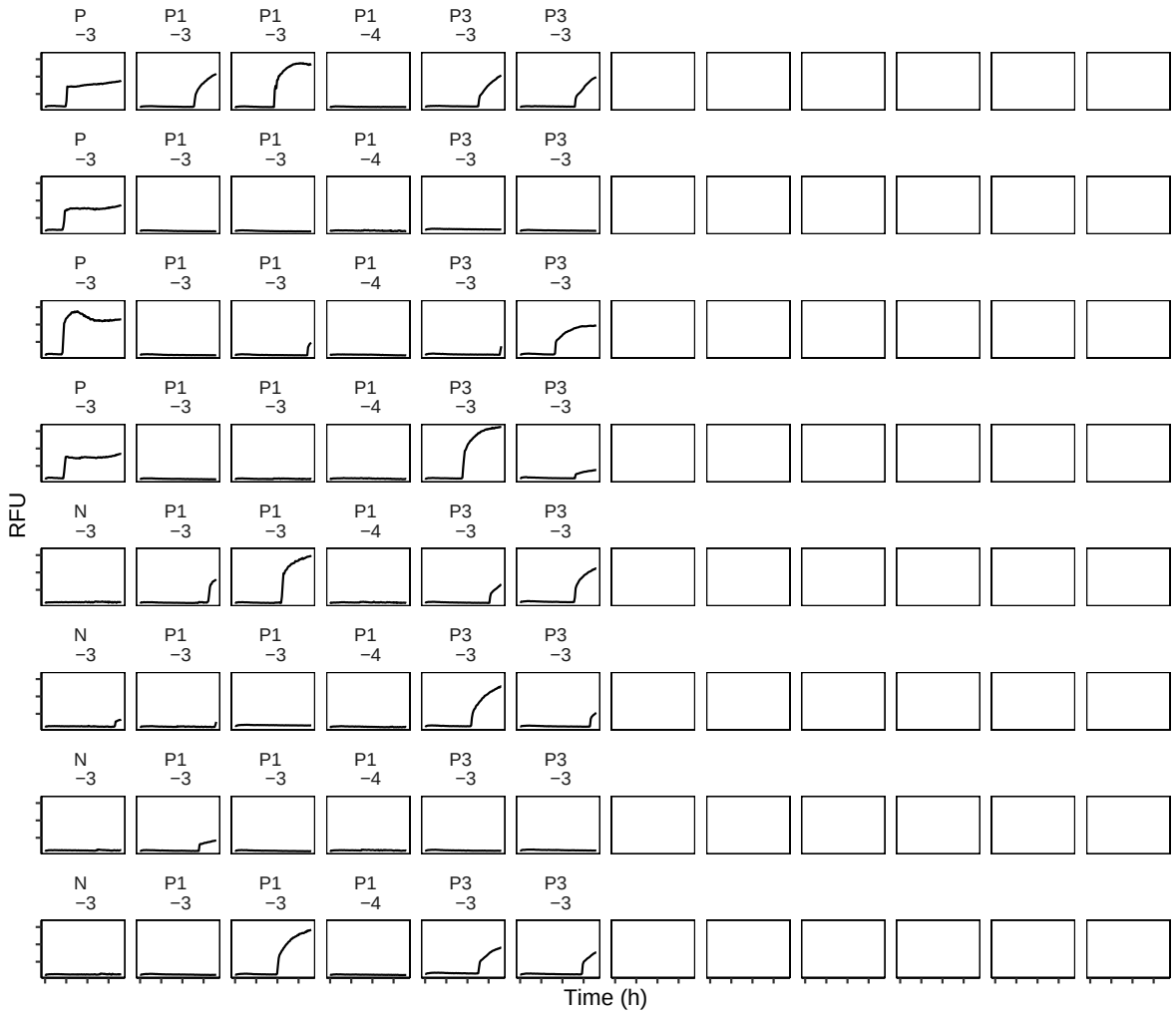
Sample IDs	Dilutions	MPR	MS	TtT	RAF
P	-3	5.806240	1.6785258	20.24882	0.0493856
P1	-3	7.740608	1.2442854	51.55198	0.0193979
P1	-3	10.405253	2.0412758	36.96472	0.0270528
P1	-4	1.082243	0.0880581	72.00000	0.0000000
P3	-3	6.993232	0.9753713	50.83224	0.0196726
P3	-3	6.985816	1.0315208	52.23304	0.0191450
P	-3	5.614148	1.5627009	17.55778	0.0569548
P1	-3	1.028725	0.0670257	72.00000	0.0000000
P1	-3	1.077505	0.0655324	72.00000	0.0000000
P1	-4	1.156900	0.0840160	72.00000	0.0000000

Sample IDs	Dilutions	MPR	MS	TtT	RAF
P3	-3	1.095514	0.0424505	72.00000	0.0000000
P3	-3	1.091549	0.0618153	72.00000	0.0000000
P	-3	9.213333	2.6466667	16.67595	0.0599666
P1	-3	1.057348	0.0637196	72.00000	0.0000000
P1	-3	3.512059	1.0463822	68.74862	0.0145457

Visualization

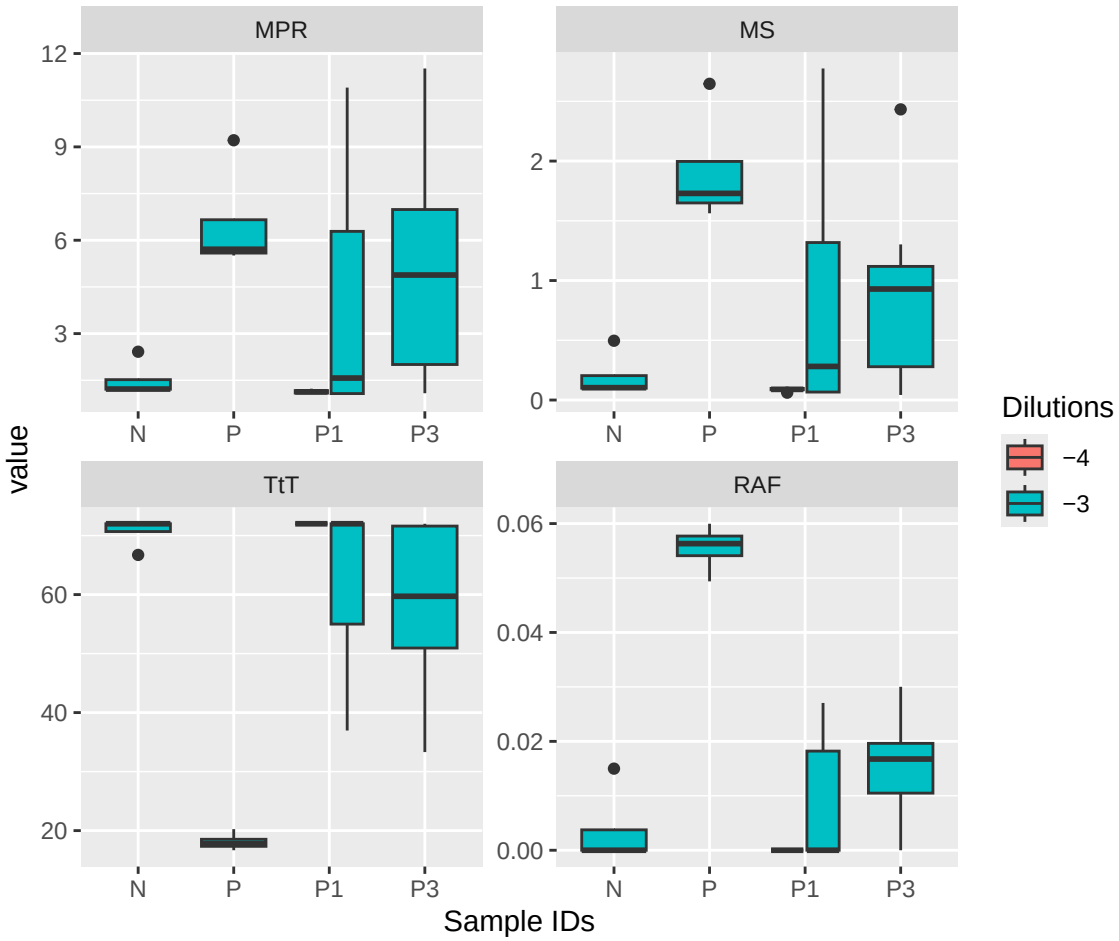
Plate View

```
sample_locations <- get_sample_locations(
  file,
  dilution_bool = TRUE,
  dilution_fun = function(x) -log10(x)
)

plate_view(df_, sample_locations)
```



Summary Plots

```
df_analyzed %>%
  melt(id.vars = c("Sample IDs", "Dilutions")) %>%
  mutate_at("Dilutions", as.factor) %>%

  ggplot(aes(`Sample IDs`, value, fill = Dilutions)) +
    geom_boxplot() +
    facet_wrap(~variable, scales = "free")
```



Usage

Validation & Performance

Discussion

Conclusion

Acknowledgments

References

Atarashi, Ryuichiro, Kazunori Sano, Katsuya Satoh, and Noriyuki Nishida. 2011. "Real-Time Quaking-Induced Conversion: A Highly Sensitive Assay for Prion Detection." *Prion*. Taylor & Francis. https://doi.org/10.4161/pri.5.3.16893.

Gallups, Nicole J., and Ashley S. Harms. 2022. "'Seeding' the Idea of Early Diagnostics in Synucleinopathies." *Brain* 145 (April): 418–19. https://doi.org/10.1093/BRAIN/AWAC062.

Henderson, Davin M., Kristen A. Davenport, Nicholas J. Haley, Nathaniel D. Denkers, Candace K. Mathiason, and Edward A. Hoover. 2015. "Quantitative Assessment of Prion Infectivity in Tissues and Body Fluids by Real-Time Quaking-Induced Conversion." *Journal of General Virology* 96 (January): 210–19. https://doi.org/10.1099/vir.0.069906-0.

Orrú, Christina D., Bradley R. Groveman, Andrew G. Hughson, Gianluigi Zanusso, Michael B. Coulthart, and Byron Caughey. 2015. "Rapid and Sensitive RT-QuIC Detection of Human Creutzfeldt-Jakob Disease Using Cerebrospinal Fluid." *mBio* 6 (January). https://doi.org/10.1128/mBio.02451-14.

Rowden, Gage R., Catalina Picasso-Risso, Manci Li, Marc D. Schwabenlander, Tiffany M. Wolf, and Peter A. Larsen. 2023. "Standardization of Data Analysis for RT-QuIC-Based Detection of Chronic Wasting Disease." *Pathogens* 12 (February): 309. https://doi.org/10.3390/PATHOGENS12020309/S1.