**READ2018.PDF** (Introduction to Recurrence Quantification Analysis, v 15.1)

Author: C.L. Webber, Jr.

Date: May 4, 2018

Recurrence Quantitative Analysis (RQA) analysis is a graphical, statistical and analytical tool for the study of nonlinear dynamical systems. Stimulated by the work of Poincaré (1896) and first developed as a method for representing multidimensional systems in two-dimensional plots (1987), research advances gave definition to five quantitative recurrence variables (1992, 1994) which were extracted as features from recurrence plots. Later, three additional variables were added to the set (2002), but possibly more variables remain to be defined. The power of recurrence quantification analysis resides in its elegant independence from constraining assumptions and limitations plaguing other analyses. Because recurrence structures (or repeat patches) are simply tallied within the signal (or between signals), there is no need to pre-condition the data by filtering, linear detrending, or some other transformation to conform to any particular statistical distribution. Recurrence analysis is not stymied by signal non-stationary, transients, outliers or noise, but can be used to detect state changes experienced by complex systems. RQA has an almost three-decade history of practical implementation in numerous real-world systems encompassing a growing multitude of disciplines (physiology, mathematics, physics, chemistry, biology, medicine, neurology, pathology, psychology, astronomy, geology, metallurgy, forestry, botany, genomics, proteomics, semiotics, linguistics, etc.).

Poincaré, H. (1896). Sue les solutions periodiques et al principe de moindre action. *Compter rendus de l'Academie des Sciences, Paris* 123: 915-918.

Eckmann, J.-P., Kamphorst, S.O., Ruelle, D. (1987). Recurrence plots of dynamical systems. *Europhys. Lett.* 4: 9773-977.

Zbilut, J.P. and Webber, C.L., Jr. (1992). Embeddings and delays as derived from quantification of recurrence plots. *Physics Letters A* 171: 199-203.

Webber, C.L., Jr. and Zbilut, J.P. (1994). Dynamical assessment of physiological systems and states using recurrence plot strategies. *Journal of Applied Physiology* 76: 965-973.

Marwan, N, Wessel, N., Meyerfeldt, U., Schirdewan, A., Kurths, J. (2002). Recurrence-plot-based measures of complexity and their application to heart rate variability data. *Physical Review E*, 66, 026702.1-026702.8.

Marwan, N., Romano, M.C., Thiel, M., Kurths, J. (2007). Recurrence plots for the analysis of complex systems. *Physics Report* 438: 237–329.

Marwan, N., Webber, C.L., Jr. (2015). Mathematical and Computational Foundations of Recurrence Quantifications. In: *Recurrence Quantification Analysis: Theory and Best Practices.* Charles Webber and Norbert Marwan, eds. Springer Series in Complexity, pp. 1-41.

In addition, three key books on RQA are available from Springer Verlag. The first book edited by Webber and Marwan represents a summary of RQA since its inception in 1992. The second book edited by Marwan, Riley, Giuliani and Webber represents proceedings from the 5th International Symposium on Recurrence Plots, Chicago, USA in 2013. And the third book edited by Webber, Ioana and Marwan, represents proceedings from the 6th International Symposium on Recurrence Plots, Grenoble, France in 2015.

Webber, C.L., Jr., Marwan, N., Editors (2015). Recurrence Quantification Analysis: Theory and Best Practices. Understanding Complex Systems. Springer International Publishing, Cham, Switzerland.
http://dx.doi.org/10.1007/978-3-319-07155-8

Marwan, N., Riley, M., Giuliani, A., Webber, C.L., Jr., Editors (2014). Translational Recurrences: From Mathematical Theory to Real-World Applications. Proceedings of the 5th International Symposium on Recurrence Plots, Chicago, USA, 14-16 Aug. 2013. Springer Proceedings in Mathematics & Statistics, Vol. 103. Springer International Publishing, Cham, Switzerland.
http://dx.doi.org/10.1007/978-3-319-09531-8

Webber, C.L., Jr., Ioana, C., Marwan, N., Editors (2016). Recurrence Plots and Their Quantifications: Expanding Horizons. Proceedings of the 6th International Symposium on Recurrence Plots, Grenoble, France, 17-19 June 2015. Springer Proceedings in Physics. Vol. 180. Springer International Publishing, Cham Switzerland.
http://dx.doi.org/10.1007/978-3-319-29922-8

The attendant software (compiled C code, version 15.1) and example data files are provided free of charge to serious users studying dynamical systems of any type. RQA programs are designed for research purposes only and are not intended for clinical diagnostics or patient care. Each of the executable RQA programs are copyrighted (CLW © 2015), and all usual disclaimers apply in accordance with United States Copyright Law. Users may freely distribute these programs to others provided they are accompanied by this READ2018.PDF file. All programs run under DOSBOX on the Windows 10 operating system (64-bit).

http://sourceforge.net/projects/dosbox/

To capture DOS screen graphics, the Snarf utility can also be installed.

http://pixelmetrics.com/Snarf/index.htm/

If the applications of these programs lead to significant research advances in the form of professional publications, we request that proper citation credit be made to our work. More complete and continuously updated listings of our published papers are provided at the following URLs as well as a master reference list maintained by Norbert Marwan.

http://cwebber.sites.luc.edu (Chuck Webber)

http://www.recurrence-plot.tk/bibliography.php (Norbert Marwan)

Additional utility programs (public domain software) and example data files are also provided for tutorial assistance. Before blindly applying RQA methodology to systems with unknown dynamics, new users should work through these specific examples. Although the operation of the programs is easy and straightforward, selection of RQA parameters and interpretation of RQA variables are decidedly difficult. Much patience is required to master the proper implementation of recurrence strategies in the diagnosis of dynamical systems.

## RECURRENCE QUANTIFICATION ANALYSIS

Before describing, in detail, the operation of each program, the following logical nomenclature scheme has been devised to assure consistency in naming the growing number of RQA programs. Forty-nine RQA program names and related terms are derived from the following 16-letter codes, each letter of which is unique and free from ambiguity with no double-meanings. Three types of recurrence analyses can be performed: single recurrences, kross recurrences and joint recurrences.

A = Analysis
C = Coordinates
D = Display
E = Epochs
F = Frequency
H = Hold
I = Intervals
J = Joint
K = Kross
L = Label
M = Matrix
N = Number
P = Parameters
Q = Quantification
R = Recurrence
S = Scale

3 programs display recurrence plots

RQD: Recurrence Quantification Display (p. 5)
KRQD: Kross Recurrence Quantification Display (p. 16)
JRQD: Joint Recurrence Quantification Display (p. 23)

3 programs report coordinates and distances of recurrent points

RQC: Recurrence Quantification Coordinates (p. 8)
KRQC: Kross Recurrence Quantification Coordinates (p. 17)
JRQC: Joint Recurrence Quantification Coordinates (p. 23)

3 programs explore the input data in parameter space

RQS: Recurrence Quantification Scale (p. 9)
KRQS: Kross Recurrence Quantification Scale (p. 17)
JRQS: Joint Recurrence Quantification Scale (p. 24)

6 programs explore the input data in moving windows

- RQE: Recurrence Quantification Epochs (p. 10)
- RQH: Recurrence Quantification Hold (p. 12)
- KRQE: Kross Recurrence Quantification Epochs (p. 18)
- KRQH: Kross Recurrence Quantification Hold (p. 20)
- JRQE: Joint Recurrence Quantification Epochs (p. 24)
- JRQH: Joint Recurrence Quantification Hold (p. 25)

6 programs compute the distribution of recurrence intervals or frequencies

- RQI: Recurrence Quantification Intervals (p. 13)
- RQF : Recurrence Quantification Frequency (p. 14)
- KRQI: Kross Recurrence Quantification Intervals (p. 21)
- KRQF : Kross Recurrence Quantification Frequency (p. 22)
- JRQI: Recurrence Quantification Intervals (p. 25)
- JRQF : Recurrence Quantification Frequency (p. 26)

2 programs extract the recurrence variables from recurrence matrices

- RQM: Recurrence Quantification Matrix (p. 15)
- KRQM: Kross Recurrence Quantification Matrix (p. 23)

15 programs generate parameter files to run programs in the batch mode

- RQSP: Recurrence Quantification Scale Parameters (p. 10)
- RQEP: Recurrence Quantification Epochs Parameters (p. 11)
- RQHP: Recurrence Quantification Hold Parameters (p. 13)
- RQIP: Recurrence Quantification Intervals Parameters (p. 14)
- RQFP : Recurrence Quantification Frequency Parameters (p. 15)
- KRQSP: Kross Recurrence Quantification Scale Parameters (p. 18)
- KRQEP: Kross Recurrence Quantification Epochs Parameters (p. 19)
- KRQHP: Kross Recurrence Quantification Hold Parameters (p. 20)
- KRQIP: Kross Recurrence Quantification Intervals Parameters (p.21)
- KRQFP : Kross Recurrence Quantification Frequency Parameters (p. 22)
- JRQSP: Joint Recurrence Quantification Scale Parameters (p. 24)
- JRQEP: Joint Recurrence Quantification Epochs Parameters (p. 24)
- JRQHP: Joint Recurrence Quantification Hold Parameters (p. 25)
- JRQIP: Joint Recurrence Quantification Intervals Parameters (p. 26)
- JRQFP : Joint Recurrence Quantification Frequency Parameters (p. 26)

11 programs display/print column labels for output variable identification

- RQSL: Recurrence Quantification Scale Label (p. 10)
- RQEL: Recurrence Quantification Epochs Label (p. 12)
- RQHL: Recurrence Quantification Hold Label (p. 13)
- RQML: Recurrence Quantification Matrix Label (p. 16)
- KRQSL: Kross Recurrence Quantification Scale Label (p. 18)
- KRQEL: Kross Recurrence Quantification Epochs Label (p. 20)
- KRQHL: Kross Recurrence Quantification Hold Label (p. 21)
- KRQML: Kross Recurrence Quantification Matrix Label (p. 23)
- JRQSL: Joint Recurrence Quantification Scale Label (p. 24)
- JRQEL: Joint Recurrence Quantification Epochs Label (p. 25)
- JRQHL: Joint Recurrence Quantification Hold Label (p. 25)

Finally, toward the end of this file (p. 34), numerical examples provide explicit step-by-step details on how input vectors (data) are delayed, embedded, normed, and rescaled to define elements (distances) in the distance matrix (DM). Also described is how the recurrence matrix (RM) is derived from the DM by selecting only those recurrent points with distances falling within a selected RADIUS. Understanding of these simple examples should remove the mathematical mystery behind the basis of recurrence quantification analysis which simply finds patterns among recurrent points according to objective rules. In short, this latter section will answer many of the computational details without using mathematical notation and is a must read/study.

As one works through these programs, important insights to recurrence theory and practice can be gleaned from the following key reference (see below). This must-read tutorial is bundled with the RQA151.ZIP software (RQA_NLDS.PDF).

Webber, C.L., Jr. and Zbilut, J.P. (2005). Recurrence quantification analysis of nonlinear dynamical systems. In: Tutorials in contemporary nonlinear methods for the behavioral sciences. Chapter 2, pp. 26-94, M.A. Riley, G. Van Orden, eds. Retrieved December 1, 2004 http://www.nsf.gov/sbe/bcs/pac/nmbs/nmbs.pdf

---

1

**RQD.EXE** (Recurrence Quantification Display, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: RQD FILE_IN

Program RQD will run on DOS machines using ASCII-file inputs of any length. This program computes and displays recurrence plots from a designated subset of the named FILE_IN. The size of the plot is automatically proportioned to the length of the input data subset, but beyond 400 input points the graph size is maximized (400 X 400), forcing displayed recurrent points to sometimes overlap (pixel resolution exceeded). Several quantitative measurements are performed and reported, none of which are affected by the overlap of plotted points (merely a display limitation). As detailed below, 11 parameters must first be defined before performing each recurrence plot analysis.

SKIP = Number of points to skip when defining sequential vector starts (0, 1 or 2)

DELAY = Spacing between selected input points in FILE_IN if the embedding dimension > 1

EMBEDDING DIMENSION = Estimated number of dominant operating variables (N) in dynamical system of interest (or ED = 2N + 1 if noise inflation is present)

TYPE OF NORMING = Method for norming vectors in higher dimensional space: maximum norm, minimum norm, Euclidean norm or order norm options

FIRST POINT = Position in FILE_IN to start recurrence analysis

LAST POINT = Position in FILE_IN to end recurrence analysis

RANDSEQ = Option to randomize selected data sequence (local randomization) before performing recurrence analysis (y/n)

RESCALE = Method for rescaling the recurrence matrix: absolute units (no rescaling), rescaled to meandist = 100.0, rescaled to maxdist = 100.0

RADIUS = Largest normed distance at and below which recurrent points are defined and displayed

COLORBND = Width of each color contour within the selected RADIUS limit

LINE = Minimum number of sequential recurrent points required to define diagonal and vertical lines

Besides the input file name, 16 quantitative measurements are computed and reported by program RQD as listed below. All computations are performed on the upper triangular area of the recurrence matrix, excluding the long central diagonal where distances are always zero (i-index = j-index). The plot is necessarily symmetrical across this diagonal (distance[i,j]= distance[j,i]). The selected value of RADIUS is always relative to the scaled MAXDIST (absolute or relative distances).

FILE = Name of ASCII input file being examined

MEAN = Mean of selected subset of input points (first point through last point)

STDEV = Standard deviation of selected subset of input points

COVAR = Coefficient of variation = STDEV / MEAN of selected subset of input points (reported as -1.000 if MEAN = 0.0)

MEANDIST = Mean of rescaled distances in upper triangular area

MAXDIST = Maximum of rescaled distances in upper triangular area

MATRIX = N by N size of recurrence matrix where N = size of selected subset of input points

#RECUR = Total number of recurrent points in upper triangular area

#LINE = Number of diagonal lines >= LINE parameter in upper triangular area

%RECUR = Percent recurrence = #RECUR / upper triangular area

%DETERM = Percent determinism = #recurrent points forming upward diagonal lines / #RECUR (reported as -1.000 if #RECUR = 0)

DMAX = Diagonal line maximum = length of longest diagonal line segment which Is inversely proportional to the largest positive Lyapunov exponent

(reported as -1 if #LINE diagonal = 0)

ENTROPY = Shannon information entropy of line distribution >= LINE parameter measured in binary (base 2) bits per bin (reported as -1.000 if #LINE diagonal = 0)

TREND = Slope of line-of-best-fit through %recurrence as function of displacement from main diagonal (excluding the last 10% of range), expressed in units of 1000 * %local recurrence / point (reported as -1.000 if #RECUR = 0)

%LAMINAR = Percent laminarity = #recurrent points forming vertical lines / #RECUR (reported as -1.000 if #RECURS = 0)

VMAX = Vertical line maximum = length of longest vertical line segment (reported as -1 if #LINE vertical = 0)

TRAPTIME = Trapping time = mean vertical line length (reported as -1.000 if #LINE vertical = 0)

Accompanying each recurrence plot are small, color-coded graphs showing: 1) the distribution of diagonal line lengths on double logarithmic scales (purple for diagonal lines < LINE; yellow for diagonal lines >= LINE); and 2) %recurrence of diagonal lines as a function of displacement from the main diagonal on double linear scales (bright red for trend calculations; dark red for 10% exclusions). The horizontal line (blue) below the super-positioned graphs represents the size of the data subset (less 1 because the central diagonal is ignored) which is thickened for diagonal line lengths >= LINE.

The input data is plotted directly beneath the recurrence plot in proper alignment. Pressing the ENTER key causes a "cross" cursor appears on the recurrence plot which is coupled with vertical cursors in the input file (red for x-cursor; green for y-cursor). Cursor locations are also identified as integer positions in the input file, shifted to higher values to account for the last embedded point. By moving the cursor, it is possible to associate specific patterns in the recurrence plot with their corresponding positions in the time (or spatial) series.

The cursor can be moved around the recurrence plot by pressing any one of the eight (or four) arrow keys. The cursor skip interval is defaulted to 10 points, but various skip intervals can be selected by pressing the proper function key.

F1 = Skip interval of 1 point (fine steps)
F2 = Skip interval of 10 points
F3 = Skip interval of 200 points
F4 = Skip interval of 500 points (course steps)

The default cursor color is yellow, but it may be necessary to change the color to increase contrast with recurrent points in the background. This is easily accomplished by pressing other function keys.

F9 = Red cursor
F10 = Yellow cursor
F11 = Green cursor
F12 = Black cursor

If working under a DOS emulator, it is possible to hardcopy-capture the main recurrence plot, attendant graphs, and reported computations by using the freeware, Snarf.
http://pixelmetrics.com/Snarf/index.htm

The RQD program cycles in the refresh display mode (ESC key), allowing the user to assign new parameter values to RADIUS, COLORBND, and LINE for the currently active subset of data. That is, new recurrence matrices can be derived from the present distance matrix. The proper way to exit the program is by entering a negative number (e.g. -1) for any of the three requested parameter values.

---

2
**RQC.EXE** (Recurrence Quantification Coordinates, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: RQC FILE_IN > FILE_OUT

Program RQC is used to generate a three-column matrix of recurrence point coordinates and scaled distance values (i-index, j-index, distance). This program is identical to program RQD except that all graphics are suppressed, and parameters COLORBND and LINE are precluded. The N*3 matrix is written to FILE_OUT, the size of which depends upon the number of recurrent points which in turn is a function of parameter RADIUS. For example, if RESCALE = 1, RADIUS = 10.0, and N = 9 cycles, FILE_OUT might appear as below. Symmetry can be noted (distance[i,j] = distance[j,i]), and coordinates with distances exceeding RADIUS = 10.0 are excluded.

| i-index | j-index | distance |
|---|---|---|
| 1 | 1 | 0.000 |
| 1 | 3 | 4.533 |
| 1 | 5 | 9.265 |
| 2 | 2 | 0.000 |
| 2 | 9 | 7.118 |
| 3 | 3 | 0.000 |
| 3 | 1 | 4.533 |
| – | – | – |
| 9 | 2 | 7.118 |
| 9 | 9 | 0.000 |

The matrix data in FILE_OUT can be easily imported by commercial software and plotted as desired. This is especially useful when working in an emulated DOS environment (for which there are no screen-dump utilities). Plotting points at coordinates i-index and j-index will generate recurrence plots identical to those

displayed on the screen by programs RQD and RQDV. Three-dimensional mesh plots can be constructed by including distance as the third variable and fitting it with a surface. If so desired, FILE_OUT can also be partitioned into three separate files by program READCOL (see below).

---

3

**RQS.EXE** (Recurrence Quantification Scale, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: RQS FILE_IN [FILE_PAR] > FILE_OUT

Program RQS scales recurrence quantifications for a single epoch of data by incrementing parameter values over specified ranges. The user must define 16 parameters as listed below. Note that the DELAY and EMBED parameters are integers that are automatically incremented by steps of one. Computation time depends upon the file lengths and specific parameter selections. The program can be called serially from a batch file if parameter values are previously defined in parameter files [FILE_PAR] (see program RQSP below).

SKIP = Number of points to skip when defining sequential vector starts (0, 1 or 2)
DELAY MIN = Minimum number of points to skip for embedding
DELAY MAX = Maximum number of points to skip for embedding
EMBED MIN = Minimum embedding dimension
EMBED MAX = Maximum embedding dimension
NORM = Maximum, minimum, Euclidean or order norm (1, 2, 3, 4)
FIRST = First point for recurrence analysis in first epoch
LAST = Last point for recurrence analysis in first epoch
SHIFT = Integer shift between data set epoch starts
EPOCH = Number of Consecutive data set epochs
RANDSEQ = Option to randomize selected data sequence (y/n)
RESCALE = Absolute, meandist, maxdist rescaling of matrix (1, 2, 3)
RADIUS MIN = Min inclusion threshold for scaled dist <= RADIUS
RADIUS MAX = Max inclusion threshold for scaled dist <= RADIUS
RADIUS STEP =Step increment in RADIUS values
LINE = Minimum number of diagonal points to define lines

Twelve output variables (columns) are printed to the screen and to FILE_OUT for each epoch (rows) as shown below. FILE_OUT data can be imported by commercial software to generate 3-dimensional plots (surfaces) by selecting any one recurrence variable (e.g. %DETERM) and plotting it as a function of any two incremented parameters (e.g. EMBED and RADIUS).

| | | | | | |
|---|---|---|---|---|---|
| START | DELAY | EMBED | RADIUS | %RECUR | %DETERM |
| DMAX | ENTROPY | TREND | %LAMINAR | VMAX | TRAPTIME |

---

4

**RQSP.EXE** (Recurrence Quantification Scale Parameters, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: RQSP FILE_IN > FILE_PAR

Program RQSP is used to generate parameter files that can be read by program RQS in batch processing mode for multiple data sets. There must be 15 parameters in each parameter file, in proper order, as listed under program RQS above. The program disallows illegal entries, but parameter files can be edited (and rendered illegal if not careful) with the DOS editor (e.g. EDIT FILE_PAR). If any illegal parameter is encountered at run time, the program is immediately halted, and the entire batch process is terminated.

Batch files like the one shown below can also be generated by the DOS editor. Note that the {brackets} only indicate batch-file contents. The batch file can be executed by simply entering at the DOS prompt: DOTHIS.

```
DOTHIS.BAT {
    RQS IN_1 PAR_1 > OUT_1
    RQS IN_2 PAR_2 > OUT_2
    RQS IN_3 PAR_3 > OUT_3
    RQS IN_4 PAR_4 > OUT_4
   }
```

---

5

**RQSL.EXE** (Recurrence Quantification Scale Labels, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: RQSL

For convenience, this program labels the column names of the 12 variables written by program RQS in the exact order of the file extensions assigned by program READCOL (see below).

---

6

**RQE.EXE** (Recurrence Quantification Epochs, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: RQE FILE_IN [FILE_PAR] > FILE_OUT

Program RQE computes recurrence quantifications on an epoch-by-epoch basis. The output is entirely numeric with no actual recurrence plots being visualized (not necessary). This program runs on DOS machines using ASCII file inputs of any length. Computation time depends upon the file lengths and specific parameter selections. Multiple files can be batched by assigning legal parameter values to FILE_PAR (see program RQEP below). Except for COLORBND, the remaining 9 parameters described for program RQD must be specified, along with 2 new parameters defining DATA SHIFT and NUMBER OF EPOCHS (12 total parameters).

FIRST POINT = Position in FILE_IN to start analysis for the first epoch

LAST POINT = Position in FILE_IN to end analysis for the first epoch

DATA SHIFT = Number of points to shift between epoch starts

NUMBER OF EPOCHS = Number of times to repeat the recurrence quantification analysis within the epoch window sliding down the input data set

Eleven output variables (columns) are printed to the screen and to FILE_OUT for each epoch (rows) as shown below. These include START plus the same 10 variables defined by program RQD.

| | | | | | |
|---|---|---|---|---|---|
| START | MEAN | STDEV | %RECUR | %DETERM | DMAX |
| ENTROPY | TREND | %LAMINAR | VMAX | TRAPTIME | |

The START variable identifies the very first point for each epoch calculation. For making proper time alignments between the output variables and the input data set, program RQE also reports the range of FILE_IN points to be plotted so that false anticipatory responses are precluded. These two computed limits coincide with the last embedded point of the first epoch and the last embedded point of the last epoch. These two values should always be carefully noted.

---

7

**RQEP.EXE** (Recurrence Quantification Epochs Parameters, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: RQEP FILE_IN > FILE_PAR

Program RQEP is used to generate parameter files that can be imported to program RQE for the automatic batch processing of multiple data sets. There must be 12 parameters defined in each file, in proper order, as listed below. The program disallows illegal entries, but parameter files can be edited (and rendered illegal if not careful) with the DOS editor (e.g. EDIT FILE_PAR). If any illegal parameter is encountered at run time, the program is immediately halted, and the entire batch processing is terminated.

SKIP = Number of points to skip when defining sequential vector starts (0, 1 or 2)
DELAY = Number of points to skip for embedding
EMBED = Embedding dimension
NORM = Maximum, minimum, Euclidean or order norm (1, 2, 3, 4)
FIRST = First point for recurrence analysis in first epoch
LAST = Last point for recurrence analysis in first epoch
SHIFT = Integer shift between data set epoch starts
N_EPOCH = Total number if consecutive data set epochs
RANDSEQ = Option to randomize selected data sequence (y/n)
RESCALE = Absolute, meandist, maxdist rescaling of matrix (1, 2, 3)
RADIUS = Exclusion threshold for vector distances > radius
LINE = Minimum number of diagonal points to define lines

The example batch file shown below can be generated by the DOS editor. Note that the {brackets} only indicate batch-file contents. The batch file can be executed by simply entering at the DOS prompt: DOTHAT.

```
DOTHAT.BAT {
        RQE IN_5 PAR_5 > OUT_5
        RQE IN_6 PAR_6 > OUT_6
        RQE IN_7 PAR_7 > OUT_7
        RQE IN_8 PAR_8 > OUT_8
        }
```

Multiple batch files can be serially processed by a single master batch file. Again, the brackets indicate batch-file contents which can be executed by entering: MASTER.

```
MASTER.BAT {
        CALL DOTHIS
        CALL DOTHAT
        }
```

---

8

**RQEL.EXE** (Recurrence Quantification Epochs Labels, v 15.1)

Programmer C.L. Webber Jr. © 2012

Usage: RQEL

For convenience, this program labels the column names of the 11 variables written by program RQE in the exact order of the file extensions assigned by program READCOL (see below).

---

9

**RQH.EXE** (Recurrence Quantification Hold, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: RQI FILE_IN [FILE_PAR] > FILE_OUT

Program RQEH is identical to program RQE except that %RECUR is held constant as a parameter and RADIUS is treated as a variable. The user selects the desired target %RECUR value and the program automatically adjusts the RADIUS up and down until the most accurate %RECUR is found. For small data sets and the presence of unavoidable computer round-off errors, exact targeted %RECUR values cannot always be found, even though computations are performed in double precision. And because of the iterative search, this program necessarily runs decidedly slower than it counterpart in which RADIUS is a parameter and %RECUR is a variable.

Eleven output variables (columns) are printed to the screen and to FILE_OUT for each epoch (rows) as shown below. Note that RADIUS substitutes for STDEV as reported by program RQE.

```
START     MEAN      RADIUS    %RECUR    %DETERM   DMAX
ENTROPY   TREND     %LAMINAR  VMAX      TRAPTIME
```

---

10

**RQHP.EXE** (Recurrence Quantification Hold Parameters, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: RQEHP FILE_IN > FILE_PAR

Program RQEHP is used to generate parameter files that can be imported to program RQEH for the automatic batch processing of multiple data sets. The same 12 parameters are required as for program RQEP, except that %RECUR is substituted for %RECUR.

---

11

**RQHL.EXE** (Recurrence Quantification Hold Labels, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: RQEHL

For convenience, this program labels the column names of the 11 variables written by program RQEH in the exact order of the file extensions assigned by program READCOL (see below).

---

12

**RQI.EXE** (Recurrence Quantification Intervals, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: RQI FILE_IN [FILE_PAR] > FILE_OUT

This program computes the distribution of recurrence intervals. Recurrence intervals are defined as the j-separation of recurrent points at constant i-index values. Intervals truncated by the boarders of the recurrence matrix are excluded. The entire recurrence matrix is scanned at each value of i-index, including the ubiquitous central diagonal. FILE_OUT contains the number of integer bin counts at each i-index value from 1 to N where N equals the number of cycles selected. Percent recurrence is reported, but this variable is slightly inflated over other RQA programs because recurrent points along the central diagonal are not excluded. The output file can be plotted in histogram format, but note that bin counts can be exceptionally high, especially for periodic systems. If FILE_PAR is included in the command line the program can be run in the batch mode (see program RQIP below).

Distributions of recurrence intervals can be transformed into nonlinear recurrence spectra (analogous to linear frequency spectra; e.g. Fast Fourier transforms). This is done by taking the reciprocal of the interval positions of all bins with non-zero counts. Typically, the first bin is rejected as noise. Knowing the digitization rate allows the horizontal frequency scale to be calibrated (Hz). Bin counts (vertical "power" axis) are usually log transformed for scaling purposes.

---

13

**RQIP.EXE** (Recurrence Quantification Intervals Parameters, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: RQIP FILE_IN > FILE_PAR

Program RQIP is used to generate parameter files that can be used in conjunction with program RQI for the automatic batch processing of multiple data sets. There must be 8 parameters defined in each file, in proper order, as listed below. The program disallows illegal entries, but parameter files can be edited (and rendered illegal if not careful) with the DOS editor (e.g. EDIT FILE_PAR). If any illegal parameter is encountered at run time, the program is immediately halted, and the entire batch-file processing is terminated.

DELAY = Number of points to skip for embedding
EMBED = Embedding dimension
NORM = Maximum, minimum, Euclidean or order norm (1, 2, 3, 4)
FIRST = First point for recurrence analysis in first epoch
LAST = Last point for recurrence analysis in first epoch
RANDSEQ = Permission to randomize selected data sequence (y/n)
RESCALE = Absolute, meandist, maxdist rescaling of matrix (1, 2, 3)
RADIUS = Exclusion threshold for vector distances > radius

---

14

**RQF.EXE** (Recurrence Quantification Frequency, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: RQF FILE_IN [FILE_PAR] > FILE_OUT

Program to compute the distribution of recurrence quantification frequencies derived from reciprocal recurrence intervals. The frequency scale is calibrated by specifying the digitization frequency (Hz) at which the input data were originally obtained. Intervals are defined as the as the j-separation of recurrent points for constant i-index values (as for program RQI). Intervals truncated by the boarders of the matrix are excluded. The entire recurrence matrix is scanned for all values of i, including the central diagonal. For N number of input points, the [N,2] output matrix contains the number of bin counts (recurrences) at each calibrated frequency (Hz). The frequency scale is non-linear and ranges from minimum frequency (digfreq/N) to maximum frequency (digfreq). Percent recurrence is reported, but this variable is slightly inflated above other RQA programs that exploit only the upper triangle of the recurrence matrix, excluding the central diagonal.

DELAY = Number of points to skip for embedding
EMBED = Embedding dimension
NORM = Maximum, minimum, Euclidean or order norm (1, 2, 3, 4)
FIRST = First point for recurrence analysis in first epoch
LAST = Last point for recurrence analysis in first epoch
RANDSEQ = Permission to randomize selected data sequence (y/n)
RESCALE = Absolute, meandist, maxdist rescaling of matrix (1, 2, 3)
RADIUS = Exclusion threshold for vector distances > radius
DIGFREQ = Digitization frequency (Hz)

---

15

**RQFP.EXE** (Recurrence Quantification Frequency Parameter, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: RQFP FILE_IN > FILE_PAR

Program to generate a parameter file with 9 predefined parameters to be used in conjunction with program RQF.EXE in the batch-file processing mode.

DELAY = Number of points to skip for embedding
EMBED = Embedding dimension
NORM = Maximum, minimum, Euclidean or order norm (1, 2, 3, 4)
FIRST = First point for recurrence analysis in first epoch
LAST = Last point for recurrence analysis in first epoch
RANDSEQ = Permission to randomize selected data sequence (y/n)
RESCALE = Absolute, meandist, maxdist rescaling of matrix (1, 2, 3)
RADIUS = Exclusion threshold for vector distances > radius
DIGFREQ = Digitization frequency (Hz)

---

16
**RQM.EXE** (Recurrence Quantification Matrix, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: RQM MTX_IN > FILE_OUT

This program extracts eleven recurrence variables from any pre-defined recurrence matrix. Matrix elements are identified as unique integers of I,J coordinates (columns). Successive paired coordinates consist of rows in the matrix. It is assumed that the matrix is sparse, square and possess a central line of identity (non-critical, however). For example, program RQM can be run on the output file from program RQC after the floating-point distance column has first been deleted. The output of program RQM will exactly match the output of program RQD provided the recurrence parameters are identically chosen.

Eleven output variables are printed to the screen and to FILE_OUT as a single horizontal row as shown below. Besides the standard recurrence variables, the number of recurrences, number of diagonal lines and number of vertical lines in the recurrence matrix are reported.

| | | | | | |
|---|---|---|---|---|---|
| #RECUR | #DIAG | #VERT | %RECUR | %DETERM | DMAX |
| ENTROPY | TREND | %LAMINAR | VMAX | TRAPTIME | |

---

17
**RQML.EXE** (Recurrence Quantification Matrix Label, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: RQM FILE_IN > FILE_OUT

For convenience, this program labels the column names of the 11 variables written by program RQK in the exact order of the file extensions assigned by program READCOL (see below).

---

18

**KRQD.EXE** (Kross Recurrence Quantification Display, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: KRQD FILE_1 FILE_2

This program displays cross-recurrence plots generated from two different files, but is similar to program RQD.EXE in most other respects. The program accepts the same input parameters, computes the identical output variables, plots both input files beneath the recurrence plot (FILE_1 above FILE_2), and positions movable cursors on the plots (yellow cross on recurrence plot, red line on file 1 plot, green line on file 2 plot). Since there is no longer any guarantee that the recurrence plot is symmetrical, computations are performed on the entire recurrence matrix, not just the upper triangle. Consequently, if an auto-recurrence plot is attempted with this program by comparing a single file with itself (e.g. KRQD FILE_1 FILE_1 > FILE_OUT), the resultant recurrence plot will be identical to that generated by program RQD. But the computed variables will not necessarily agree because the central diagonal is included in the calculations (e.g. LMAX will always equal N). Accompanying each recurrence plot are small, color-coded graphs like those displayed with program RQD.

One very important point to remember is that when performing kross-recurrence analyses, both input files must represent continuous data, not discontinuous data. The continuous data must also be simultaneously recorded and digitized at the same frequency. Thus, it would be appropriate to correlate two different ECG lead recordings, each sampled at the same time at 1000 Hz, but not interspike intervals from two simultaneously recorded neurons. However, it would be very appropriate to correlate two different DNA strands or two different amino-acid sequences to perform new types of homology comparisons. To handle the interval problem, it is possible to first resample the interval data and then normalize the outputs over the unit interval before subjecting the data to cross-recurrence analysis. The details for these procedures are described under programs RESAMPLE.EXE (see below).

When performing cross-recurrence on two different files, unless the two inputs are of very similar magnitudes, computed distances will be necessarily large, making it very difficult to select a proper RADIUS value. To overcome this problem, within program KRQD both input files are separately and automatically rescaled over the unit interval (and multiplied by 100).

---

19

**KRQC.EXE** (Kross Recurrence Quantification Coordinates, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: KRQC FILE_1 FILE_2 > FILE_OUT

Program KRQC is used to generate a 3-column matrix of recurrence point coordinates and scaled distance values. This program is identical to program KRQP except that all graphics are suppressed, and parameters COLORBND and LINE are not required. The matrix is written out to FILE_OUT, the size of which depends upon the number of recurrent points and RADIUS parameter. The data in the FILE_OUT matrix appear in the format as illustrated below. If FILE_1 and FILE_2 are both unique, distance[i,j] and distance[j,i] will be different (asymmetrical recurrence matrix), and there will be no central line of identity.

| i-index | j-index | distance |
|---|---|---|
| 1 | 1 | 1.020 |
| 1 | 3 | 4.533 |
| 1 | 5 | 9.265 |
| 2 | 2 | 5.553 |
| 2 | 9 | 7.118 |
| 3 | 3 | 2.701 |
| 3 | 1 | 5.956 |
| – | – | – |
| 9 | 2 | 2.333 |
| 9 | 9 | 8.987 |

The matrix data in FILE_OUT can be easily imported by commercial software and plotted as desired. Plotting points at coordinates i-index and j-index will generate recurrence plots identical to those displayed on the screen by programs KRQD and KRQDV. Three-dimensional mesh plots can also be constructed by including distance as the third variable and fitting it with a surface. If so desired, FILE_OUT can also be partitioned into three separate files by program READCOL (see below).

---

20

**KRQS.EXE** (Kross Recurrence Quantification Scale, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: KRQS FILE_1 FILE_2 [FILE_PAR] > FILE_OUT

Program KRQS scales recurrence quantifications for single epochs of paired data by incrementing parameter values over specified ranges. This program is identical to program RQS in all other aspects. This includes the rescaling of both input files over the unit interval (0%-100%) before any computations are made. The program can also be run in the batch-processing mode if FILE_PAR be properly defined (see program KRQSP below).

---

21

**KRQSP.EXE** (Kross Recurrence Quantification Scale Parameters, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: KRQSP FILE_1 FILE_2 > FILE_PAR

Program KRQSP is used to generate parameter files that can be imported to program KRQS for the automatic batch processing of multiple data sets. There must be 15 parameters defined in each file, in proper order, as listed below. The program disallows illegal entries, but parameter files can be edited (and rendered illegal if not careful) with the DOS editor (e.g. EDIT FILE_PAR). If any illegal parameter is encountered at run time, the program is immediately halted, and the entire batch-file processing is terminated.

DELAY MIN = Minimum number of points to skip for embedding
DELAY MAX = Maximum number of points to skip for embedding
EMBED MIN = Minimum embedding dimension
EMBED MAX = Maximum embedding dimension
NORM = Maximum, minimum, Euclidean or order norm (1, 2, 3, 4)
FIRST = First point for recurrence analysis in first epoch
LAST = Last point for recurrence analysis in first epoch
SHIFT = Integer shift between data set epoch starts
EPOCH = Number of Consecutive data set epochs
RANDSEQ = Option to randomize selected data sequence (y/n)
RESCALE = Absolute, meandist, maxdist rescaling of matrix (1, 2, 3)
RADIUS MIN = Min inclusion threshold for scaled dist <= RADIUS
RADIUS MAX = Max inclusion threshold for scaled dist <= RADIUS
RADIUS STEP =Step increment in RADIUS values
LINE = Minimum number of diagonal points to define lines

---

22
**KRQSL.EXE** (Kross Recurrence Quantification Scale Labels, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: KRQSL

For convenience, this program prints the title names of the 12 variables written by program KRQS in order of the file extensions assigned by program READCOL (see below).

---

23
**KRQE.EXE** (Kross Recurrence Quantification Epochs, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: KRQE FILE_1 File_2 [FILE_PAR] > FILE_OUT

Program KRQE computes cross-recurrence quantifications on an epoch-by-epoch basis, but is similar to program RQE in most other respects. Data within each and every episodic window are rescaled over the unit interval (0%-100%). The output is entirely numeric with no actual recurrence plots being visualized which is unnecessary. This program runs on DOS machines using ASCII file inputs of any length. Computation time depends upon the file lengths and specific parameter selections. The program can be called serially from a batch file if parameter values are previously

defined in parameter files [FILE_PAR] (see program KRQEP below). In sum, the same 11 parameters as previously defined for program RQE must be set for this program.

Eleven output variables (columns) are printed to the screen and to FILE_OUT for each epoch (rows) as shown below. These include the starting position, input data means, and the same 8 recurrence variables defined by program RQE.

| | | | | | |
|---|---|---|---|---|---|
| START | MEAN1 | MEAN2 | %RECUR | %DETERM | DMAX |
| ENTROPY | TREND | %LAMINAR | VMAX | TRAPTIME | |

Again, as cautioned for program KRQD above, when performing cross-recurrence analysis, both input files must represent simultaneously recorded, continuous data, not discontinuous data. As for program RQE, points are identified for the proper alignment of input data with computed recurrence variables.

---

24

**KRQEP.EXE** (Kross Recurrence Quantification Epochs Parameters, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: KRQEP FILE_1 FILE_2 > FILE_PAR

Program KRQEP is used to generate parameter files that can be imported to program KRQE for the automatic batch processing of multiple data sets. There must be 11 parameters defined in each file, in proper order, as listed below. The program disallows illegal entries, but parameter files can be edited (and rendered illegal if not careful) with the DOS editor (e.g. EDIT FILE_PAR). If any illegal parameter is encountered at run time, the program is immediately halted, and the entire batch-file processing is terminated.

DELAY = Number of points to skip for embedding
EMBED = Embedding dimension
NORM = Maximum, minimum, Euclidean or order norm (1, 2, 3, 4)
FIRST = First point for recurrence analysis in first epoch
LAST = Last point for recurrence analysis in first epoch
SHIFT = Integer shift between data set epoch starts
EPOCH = Total number if consecutive data set epochs
RANDSEQ = Permission to randomize selected data sequence (y/n)
RESCALE = Absolute, meandist, maxdist rescaling of matrix (1, 2, 3)
RADIUS = Exclusion threshold for vector distances > radius
LINE = Minimum number of diagonal points to define lines

The example batch file shown below can also be generated by the DOS editor. Note that the {brackets} only indicate batch-file contents. The batch can be executed by simply entering at the DOS prompt: DOALSO.

```
DOALSO.BAT {
        KRQE IN_9A IN_9B PAR_9 > OUT_9
```

```
KRQE IN_10A IN_10B PAR_10 > OUT_10
KRQE IN_11A IN_11B PAR_11 > OUT_11
KRQE IN_12A IN_12B PAR_12 > OUT_12
}
```

---

25

**KRQEL.EXE** (Kross Recurrence Quantification Epochs Labels, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: KRQEL

For convenience, this program prints the title names of the 10 variables written by program KRQE in order of the file extensions assigned by program READCOL (see below).

---

26

**KRQH.EXE** (Kross Recurrence Quantification Hold, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: KRQEH FILE_1 File_2 [FILE_PAR] > FILE_OUT

Program KRQH is identical to program KRQE except that %RECUR is held constant as a parameter and RADIUS is treated as a variable. The user selects the desired target %RECUR value and the program automatically adjusts the RADIUS up and down until the most accurate %RECUR is found. For small data sets and the presence of unavoidable computer round-off errors, exact targeted %RECUR values cannot always be found, even though computations are performed in double precision. And because of the iterative search, this program necessarily runs decidedly slower than it counterpart in which RADIUS is a parameter and %RECUR is a variable.

Twelve output variables (columns) are printed to the screen and to FILE_OUT for each epoch (rows) as shown below. Note that the RADIUS parameter, not found in KRQE data, is added to the output list.

| | | | | | |
|---|---|---|---|---|---|
| START | MEAN1 | MEAN2 | RADIUS | %RECUR | %DETERM |
| DMAX | ENTROPY | TREND | %LAMINAR | VMAX | TRAPTIME |

---

27

**KRQHP.EXE** (Kross Recurrence Quantification Hold Parameters, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: KRQEHP FILE_1 FILE_2 > FILE_PAR

Program KRQEHP is used to generate parameter files that can be imported to program KRQEH for the automatic batch processing of multiple data sets. The same 11 parameters are required as for program KRQEP, except that %RECUR is substituted for %RECUR.

---

28

**KRQHL.EXE** (Kross Recurrence Quantification Hold Labels, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: KRQEHL

For convenience, this program labels the column names of the 12 variables written by program KRQH in the exact order of the file extensions assigned by program READCOL (see below).

---

29

**KRQI.EXE** (Kross Recurrence Quantification Intervals, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: KRQI FILE_1 FILE_2 [FILE_PAR] > FILE_OUT

Program KRQI computes recurrence intervals for paired files and is identical to program RQI in all other respects. As for all cross-recurrence programs the input data files are rescaled over the unit interval (0%-100%) before any computations are made. The program can also be run in the batch-processing mode if FILE_PAR be properly defined (see program KRQIP below).

As described for program RQI above, distributions of recurrence intervals can be transformed into nonlinear recurrence spectra.

---

30

**KRQIP.EXE** (Kross Recurrence Quantification Intervals Parameters, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: KRQIP FILE_1 FILE_2 > FILE_PAR

Program KRQIP is used to generate parameter files that can be imported to program KRQI for the automatic batch processing of multiple data sets. There must be 8 parameters defined in each file, in proper order, as listed below. The program disallows illegal entries, but parameter files can be edited (and rendered illegal if not careful) with the DOS editor (e.g. EDIT FILE_PAR). If any illegal parameter is encountered at run time, the program is immediately halted, and the entire batch-file processing is terminated.

DELAY = Number of points to skip for embedding
EMBED = Embedding dimension
NORM = Maximum, minimum, Euclidean or order (1, 2, 3, 4)
FIRST = First point for recurrence analysis in first epoch
LAST = Last point for recurrence analysis in first epoch
RANDSEQ = Permission to randomize selected data sequence (y/n)
RESCALE = Absolute, meandist, maxdist rescaling of matrix (1, 2, 3)
RADIUS = Exclusion threshold for vector distances > radius

---

31

**KRQF.EXE** (Recurrence Quantification Frequency, v 15.1)

Programmer: C.L. Webber Jr. © 2018
Usage: KRQF FILE_1 FILE_2 [FILE_PAR] > FILE_OUT

Program to compute the distribution of recurrence quantification frequencies computed from reciprocal recurrence intervals. The frequency scale is calibrated by specifying the digitization frequency (Hz) at which the input data were originally obtained. Intervals are defined as the j-separation of recurrent points for constant i-index values (as for program RQI). Intervals truncated by the boarders of the matrix are excluded. The entire recurrence matrix is scanned for all values of i, including the central diagonal. For N number of input points, the [N,2] output matrix contains the number of bin counts (recurrences) at each calibrated frequency (Hz). The frequency scale is non-linear and ranges from minimum frequency (digfreq/N) to maximum frequency (digfreq). Percent recurrence is reported, but this variable is slightly inflated above other RQA programs that exploit only the upper triangle of the recurrence matrix, excluding the central diagonal.

DELAY = Number of points to skip for embedding
EMBED = Embedding dimension
NORM = Maximum, minimum, Euclidean or order norm (1, 2, 3, 4)
FIRST = First point for recurrence analysis in first epoch
LAST = Last point for recurrence analysis in first epoch
RANDSEQ = Permission to randomize selected data sequence (y/n)
RESCALE = Absolute, meandist, maxdist rescaling of matrix (1, 2, 3)
RADIUS = Exclusion threshold for vector distances > radius
DIGFREQ = Digitization frequency (Hz)

---

32
**KRQFP.EXE** (Recurrence Quantification Frequency Parameter, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: KRQFP FILE_1 FILE_2 [FILE_PAR] > FILE_PAR

Program to generate a parameter file with 9 predefined parameters to be used in conjunction with program KRQF.EXE in the batch-file processing mode.

DELAY = Number of points to skip for embedding
EMBED = Embedding dimension
NORM = Maximum, minimum, Euclidean or order norm (1, 2, 3, 4)
FIRST = First point for recurrence analysis in first epoch
LAST = Last point for recurrence analysis in first epoch
RANDSEQ = Permission to randomize selected data sequence (y/n)
RESCALE = Absolute, meandist, maxdist rescaling of matrix (1, 2, 3)
RADIUS = Exclusion threshold for vector distances > radius
DIGFREQ = Digitization frequency (Hz)

---

33
**KRQM.EXE** (Kross Recurrence Quantification Matrix, v 15.1)
Programmer: C.L. Webber Jr. © 2018

Usage: KRQM KMTX_IN > FILE_OUT

This program extracts eleven recurrence variables from any pre-defined cross recurrence matrix. Matrix elements are identified as unique integers of I,J coordinates (columns). Successive paired coordinates consist of rows in the matrix. It is assumed that the matrix is sparse, square, but does not possess a central line of identity. For example, program KRQM can be run on the output file from program KRQC after the floating-point distance column has first been deleted. The output of program KRQM will exactly match the output of program KRQD provided the recurrence parameters are identically chosen.

Eleven output variables are printed to the screen and to FILE_OUT as a single horizontal row as shown below. Besides the standard recurrence variables, the number of recurrences, number of diagonal lines and number of vertical lines in the recurrence matrix are reported.

| | | | | | |
|---|---|---|---|---|---|
| #RECUR | #DIAG | #VERT | %RECUR | %DETERM | DMAX |
| ENTROPY | TREND | %LAMINAR | VMAX | TRAPTIME | |

---

34

**KRQML.EXE** (Kross Recurrence Quantification Matrix Label, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: KRQM FILE_IN

For convenience, this program labels the column names of the 11 variables written by program KRQM in the exact order of the file extensions assigned by program READCOL (see below).

---

35

**JRQD.EXE** (Joint Recurrence Quantification Display, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: JRQD FILE_1 FILE_2

This program is identical to programs RQD and KRQD except that it computes the joint recurrence plot between two input files.

---

36

**JRQC.EXE** (Joint Recurrence Quantification Coordinates, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: JRQC FILE_1 FILE_2 > FILE_OUT

This program is identical to programs RQC and KRQC except that it computes the joint recurrence coordinates between two input files.

---

37

**JRQS.EXE** (Joint Recurrence Quantification Scale, v 15.1)
Programmer: C.L. Webber Jr. © 2018

Usage: JRQS FILE_1 FILE_2 [FILE_PAR] > FILE_OUT

This program is identical to programs RQS and KRQS except that it computes the joint recurrence scales between two input files.

---

38

**JRQSP.EXE** (Joint Recurrence Quantification Scale Parameters, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: JRQSP FILE_1 FILE_2 > FILE_PAR

This program is identical to programs RQSP and KRQSP except that it generates a parameter file for program JRQS to run in the batch-file processing mode.

---

39

**JRQSL.EXE** (Joint Recurrence Quantification Scale Labels, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: JRQSL

For convenience, this program labels the column names of the 12 variables written by program JRQS in the exact order of the file extensions assigned by program READCOL (see below).

---

40

**JRQE.EXE** (Joint Recurrence Quantification Epochs, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: JRQE FILE_1 FILE_2 [FILE_PAR] > FILE_OUT

This program is identical to programs RQE and KRQE except that it computes the joint recurrence epochs between two input files.

---

41

**JRQEP.EXE** (Joint Recurrence Quantification Epochs Parameters, v 15.1)
Programmer: C.L. Webber Jr. © 2018
Usage: JRQEP FILE_1 FILE_2 > FILE_PAR

This program is identical to programs RQEP and KRQEP except that it generates a parameter file for program JRQE to run in the batch-file processing mode.

---

42

**JRQEL.EXE** (Joint Recurrence Quantification Epochs Labels, v 15.1)
Programmer: C.L. Webber Jr. © 2018

Usage: JRQEL

For convenience, this program labels the column names of the 11 variables written by program JRQE in the exact order of the file extensions assigned by program READCOL (see below).

---

43

**JRQH.EXE** (Joint Recurrence Quantification Hold, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: JRQH FILE_1 FILE_2 [FILE_PAR] > FILE_OUT

This program is identical to programs RQH and KRQH except that it computes the joint recurrence variables while holding the %recurrence constant while adjusting the radius.

---

44

**JRQHP.EXE** (Joint Recurrence Quantification Hold Parameters, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: JRQHP FILE_1 FILE_2 > FILE_PAR

This program is identical to programs RQHP and KRQHP except that it generates a parameter file for program JRQH to run in the batch-file processing mode.

---

45

**JRQHL.EXE** (Joint Recurrence Quantification Hold Labels, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: JRQHL

For convenience, this program labels the column names of the 12 variables written by program JRQH in the exact order of the file extensions assigned by program READCOL (see below).

---

46

**JRQI.EXE** (Joint Recurrence Quantification Intervals, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: JRQI FILE_1 FILE_2 [FILE_PAR] > FILE_OUT

This program is identical to programs RQI and KRQI except that it computes the joint recurrence intervals between two input files.

---

47

**JRQIP.EXE** (Joint Recurrence Quantification Intervals Parameters, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: JRQIP FILE_1 FILE_2 > FILE_PAR

This program is identical to programs RQIP and KRQIP except that it generates a parameter file for program JRQI to run in the batch-file processing mode.

---

48

**JRQF.EXE** (Joint Recurrence Quantification Frequency, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: JRQF FILE_1 FILE_2 [FILE_PAR] > FILE_OUT

This program is identical to programs RQF and KRQF except that it computes the joint recurrence frequency between two input files.

---

49

**JRQFP.EXE** (Joint Recurrence Quantification Frequency Parameters, v 15.1)

Programmer: C.L. Webber Jr. © 2018

Usage: JQFP FILE_1 FILE_2 > FILE_PAR

This program is identical to programs RQFP and KRQFP except that it generates a parameter file for program JRQF to run in the batch-file processing mode.

---

50

**RESAMPLE.EXE** (Resample Function)

Programmer: C.L. Webber Jr., Public Domain

Usage: RESAMPLE F1 [F2 [F3 [F4 [F5]]]] > FILE_OUT

This program resamples time-interval data from up to 5 parallel files at any selected frequency. It is assumed that the first entry of each input file represents the delay interval from the start of data acquisition to the occurrence of the first event. For proper time calibration, the user must specify the original digitization rate at which the data were obtained (Hz), as well as the desired resampling rate (Hz). For N inputs files, the program generates N+1 output columns (sec), one column per input file, and one final column for clock-time (1/resampling rate). The algorithm implements latch logic such that at every click of the resampling clock, the current interval time is reported without interpolation. With each new event, latch times are updated independently, only for that channel in which the event occurred. Outputs are suppressed until the longest offset delay among the input files has expired. Outputs are terminated when the cumsum of the shortest file is exhausted. By this means, resampled data columns have identical counts, are perfectly synchronized in time, and are rendered suitable for subsequent paired analysis. If the input files are on different ranges, it is advisable to rescale the data over the unit interval or according to a standard score using programs UNITINT or ZSCORE (see below). Finally, the output data matrix can be extracted, column by column, by using program READCOL (see below).

---

51

**UNITINT.EXE** (Unit Interval Function)

Programmer: C.L. Webber Jr., Public Domain

Usage: UNITINT FILE_IN > FILE_OUT

This program rescales all N elements of the input file over the unit interval. The program first searches out the maximum and minimum values from the input file. An output file of N elements is then generated, the elements being rescaled over the range from 0.0 to 1.0 (e.g. minimum to maximum). Rescaling data over the unit interval is useful when trying to correlate data with widely disparate scales (e.g. seconds versus milliseconds; meters versus millimeters). Nevertheless, unit-interval rescaling is implemented in all cross-recurrence programs.

---

52

**ZSCORE.EXE** (Standard Score Function)

Programmer: C.L. Webber Jr., Public Domain

Usage: ZSCORE FILE_IN > FILE_OUT

This program rescales all N elements of the input file to a standard score. The program first computes the mean and standard deviation from the input file. An output file of N elements is then generated, the elements being redefined as standard scores with a mean of 0.0 and a standard deviation of 1.0. As for program UNITINT, rescaling data by the standard score is useful in bringing data from disparate scales into similar range. If the input file has a single outlier (e.g. element > 3 stdev from the mean), output files written by program ZSCORE will be less influenced (e.g. less compressed) than those written by program UNITINT.

Standard Score = (raw score - mean raw) / stdev raw

---

53

**READCOL.EXE** (Read Column)

Programmer: C.L. Webber Jr., Public Domain)

Usage : READCOL

This utility program is designed to extract columns of data from matrix files generated by programs RQS, KRQS, RQE, KRQE, RQH and KRQH. Upon prompting, the user specifies the name of the file to be parsed (FILE_OUT, which should have no file extension) and the number (N) of variable columns represented for above named programs (N = 11 or 12). Program READCOL then generates 11 or 12 new vector files identified by unique integer extensions appended to the FILE_OUT base name (e,g, FILE_OUT.1, FILE_OUT.2, FILE_OUT.3 ..., FILE_OUT.11, FILE_OUT.12). Labeling programs RQSL, KRQSL, RQEL, KRQEL, RQHL and KRQHL can be run as reminders, properly matching RQA variables with designated file extensions. Explicit file extensions are identified below for a DATA matrix.

| File Name | RQS Vars | KRQS Vars | RQE Vars | KRQE Vars | RQH Vars | KRQH Vars |
|---|---|---|---|---|---|---|
| DATA.1 | DELAY | DELAY | START | START | START | MEAN1 |

| | | | | | | |
|---|---|---|---|---|---|---|
| DATA.2 | EMBED | EMBED | MEAN | MEAN1 | MEAN | MEAN2 |
| DATA.3 | RADIUS | RADIUS | STDEV | MEAN2 | RADIUS | RADIUS |
| DATA.4 | %REC | %REC | %REC | %REC | %REC | %REC |
| DATA.5 | %DET | %DET | %DET | %DET | %DET | %DET |
| DATA.6 | DMAX | DMAX | DMAX | DMAX | DMAX | DMAX |
| DATA.7 | ENT | ENT | ENT | ENT | ENT | ENT |
| DATA.8 | TREND | TREND | TREND | TREND | TREND | TREND |
| DATA.9 | %LAM | %LAM | %LAM | %LAM | %LAM | %LAM |
| DATA.10 | VMAX | VMAX | VMAX | VMAX | VMAX | VMAX |
| DATA.11 | TTIME | TTIME | TTIME | TTIME | TTIME | TTIME |
| DATA.12 | – | – | – | – | – | START |

Outputs from program RESAMPLE are likewise written to a single matrix. This program can accept from one to five input files for resampling, generating a column for each as well as the clock time (TIME). For example, program READCOL can be used to extract individual matrix SAMP with the following extension identifications.

| # inputs | SAMP.1 | SAMP.2 | SAMP.3 | SAMP.4 | SAMP.5 | SAMP.6 |
|---|---|---|---|---|---|---|
| 1 | FILE_1 | TIME | | | | |
| 2 | FILE_1 | FILE_2 | TIME | | | |
| 3 | FILE_1 | FILE_2 | FILE_3 | TIME | | |
| 4 | FILE_1 | FILE_2 | FILE_3 | FILE_4 | TIME | |
| 5 | FILE_1 | FILE_2 | FILE_3 | FILE_4 | FILE_5 | TIME |

---

54

**PLOT.EXE** (Plot)

Programmer: C.L. Webber Jr., Public Domain
Usage: PLOT FILE_IN

This convenience program is useful for rapidly plotting individual variables generated by programs RQE or RQM and extracted by program READCOL. For example, recurrence quantification variables can be plotted as functions of epoch number (a time or space function). After the total number of points in the file are reported (e.g. #EPOCHS), the user selects the range of points to plot as well as the vertical-scaling range. To make accurate time (or space) alignments with the original time (or space) series, FILE_IN must be plotted over the range of points reported by program RQE or program RQK. The number of excluded points is related to the epoch size and embedding dimension, allowing for perfect alignments between input and outputs. Ignoring this offset will lead to erroneous and potentially serious mismatches between FILE_IN and FILE_OUT.N (e.g. implying anticipatory responses in recurrence variables incorrectly).

Hard copies of resultant graphic screens can be obtained with standard screen-dump utilities. Combined plots of multiple variables with descriptive labels and calibrations of axes can be obtained with more sophisticated plotting programs.

---

55

**ACOR.EXE** (Auto-correlation Function)

Programmer: C.L. Webber Jr., Public Domain
Usage: ACOR FILE_IN

This program computes and displays the auto-correlation function of the input data file and can be used to suggest a value for the DELAY parameter when performing RQA. For discrete systems (maps: cardiac R-R intervals, breathing periods, DNA codes, etc.) the DELAY is usually selected as one, but for continuous systems (flows: muscle EMG potentials, cardiac ECG signals, pulmonary pressures), the DELAY parameter is dependent upon the digitization frequency and usually exceeds one. In an attempt to minimize correlations between analyzed points, the DELAY is typically set equal to the first zero crossing of the auto-correlation function (or first minimum if there is no zero crossing).

Program ACOR reports the number of points in FILE_IN and the user is given the option of selecting the number of points to include in each correlation (N = power of 2). The starting point of the first correlation and total number of correlations desired can be opted, the limits depending upon the total length of the input data set. When the graphics screen appears, the first epoch time-series data points are plotted (yellow) within rectangle (red) calibrated from minimum to maximum on the vertical axis. After the user enters the value of X_MAX, the number of LAG cycles to be plotted along the horizontal axis (X_MAX < N/2), the auto-correlation function is plotted (purple) within a rectangle (blue) with displayed calibration values. A horizontal line (white) is constructed through the data to mark the absolute zero level of correlation. A short vertical cursor (green) appears in the center of the screen. By pressing the left or right arrow keys this cursor can be moved along the auto-correlation function while its lag position is updated and reported. The goal is to position the cursor at either the first zero crossing or first minimum in the auto-correlogram. Pressing the escape key freezes the cursor in position (light blue), after which the program can be exited, or the current run recycled with an updated X_MAX selection.

---

56
**RANDSEQ.EXE** (Sequence Randomization)
Programmer: C.L. Webber Jr., Public Domain
Usage: RANDSEQ FILE_IN > FILE_OUT

This program randomizes reorders the sequence of data in FILE_IN (global randomization) while leaving the mean and standard deviation unchanged. Such randomization completely destroys phasic information in the data, permitting RQA comparisons on native sequences versus randomized versions of the entire data set. Global randomization is limited to files containing no more than 16331 entries. Longer files can be partitioned by program READCOL (see below), randomized by parts, and then reassembled by concatenation.

---

57
**MARKTIME.EXE** (Mark Time Function)

Programmer: C.L. Webber Jr., Public Domain
Usage: MARKTIME FILE_IN > FILE_OUT

This program is exceptionally useful for manually identifying the times of occurrence of various events in very long time series. The digitization rate at which the data was acquired must be specified for the accurate accounting of time. (For spatial data, enter 1 for digitization rate). The ASCII data is windowed and only 640 points are plotted at any one time. The user can scale the data (amplify or attenuate) and scroll the data (move the window back and forth through the data). A maximum of 21 half-windows can be stored in memory at one time, so when this larger block is exceeded, the next new data over writes the present data. However, a warning beep sounds before any data in memory is over-written. To return to former data requires a restart of the program. The user can mark events in the time series and compute the duration to the next marked event. A sequence of time intervals accumulates in FILE_OUT which can be further processed after exiting from the program. Long jumps can also be made through the data, making it very convenient to locate the exact location of certain events or artifacts in exceptionally large data sets (e.g. N > 1,000,000 elements). Program controls are effected through the function keys, arrow keys, and escape key as detailed below.

F1 = Move display backward one full screen (640 points)
F2 = Move display backward one-half screen (320 points)
F3 = Move display forward one-half screen (320 points)
F4 = Move display forward one full screen (640 points)
F5 = Attenuate vertical gain of plot by 50 percent (halve)
F6 = Magnify vertical gain of plot by 200 percent (double)
F7 = Apply negative bias to plot (-50 points)
F8 = Apply positive bias to plot (+50 points)
F9 = Mark record at current cursor position
F10 = Reset scaling and bias parameters to initial values
F11 = Move display forward 10 full screens (6400 points)
F12 = Move display forward 100 full screens (64000 points)

The cursor can be moved through the window of displayed data in variable steps (S) by using the arrow keys. Parameter S is initially set to equal to 64.

Left arrow = Move cursor to the left S positions
Right arrow = Move cursor to the right S positions
Up arrow = Multiply step size by 8 (maximum of 512)
Down arrow = Divide step size by 8 (minimum of 1)

Data from the input file is displayed in blue against a gray background. Whenever a data position is marked (F9), the trace (from the left) up to and including the marker is changed to yellow, and the remainder of the trace (to the right) stays blue. The cursor (red horizontal/vertical cross hairs) cannot be moved backward into the yellow trace (negative time is illegal). Because of memory limitations, only 21 half-screens of data can be handled at one time (6720/(640/2)). Whenever a data request brings into view the last relative window (F3, F4) or next to last relative window (F4), a warning beep is sounded as mentioned previously. Proceeding beyond this point causes the trace

stored in memory to be updated with new data. When advancing through the data, it is not possible to retrieve former portions of the trace which have been over written, even if the data has not been marked. To pick up missed events after advancing too far requires a restart of the program. At the end of the data set, the remaining memory is filled with zeros. Each time a function key is depressed, the display is refreshed and the values of 11 variables are reported.

File name
Relative window (1-20)
File percentage
File length
Absolute cursor position in file
Absolute marker position in file
Time from last marker to cursor (sec)
Time interval between two last markers (sec)
Total elapsed time since first marker (sec)
Absolute vertical value (units)
Vertical gain factor

Marking the cursor position (F9) causes the display to report the elapsed time (sec) since the previous mark. Except for the very first interval which reports the accumulated time from the start of the record, all interval times are written to an output file by indirection. The time resolution, of course, depends upon the selected rate of digitization which must be specified at the outset (Hz units). By mistake, if an attempt is made to mark a previously marked cursor position, a warning tone sounds, and the selection is ignored. The program can be exited at any point by pressing the escape key (ESC).

---

58

**BINIT.EXE** (Discretize floating-point data into integer bins)

Programmer: C.L. Webber Jr., Public Domain
Usage: BINIT FILE_IN > FILE_OUT

Program BINIT can partition floating-point data into as many equally-sized bins as desired. Doing so constitutes a filtering function, but this is required if one desires to properly compare the distance between vectors by the Hamming norm method. Hamming distances can range from 0 to embedding dimension. For example, the following example shows 5 vectors at an embedding dimension of 4.

V1 = [1, 2, 3, 4]
V2 = [1, 2, 3, 5]
V3 = [1, 2, 5, 6]
V4 = [1, 5, 6, 7]
V5 = [5, 6, 7, 8]

Distances are computed for all combinations of vectors whereby matching elements add 0 to the sum, and dissimilar elements add 1 to the sum.

RM(Hamming norm) =
[1,5]=4; [2,5]=4; [3,5]=4; [4,5]=4; [5,5]=0

[1,4]=3; [2,4]=3; [3,4]=3; [4,4]=0
[1,3]=2; [2,3]=2; [3,3]=0
[1,2]=1; [2,2]=0
[1,1]=0

---

59

**LEMPZIV.EXE** (Lempel-Ziv Compression)

Programmer: C.L. Webber Jr., Public Domain
Usage: LEMPZIV FILE_IN > FILE_OUT

Program LEMPZIV performs a Lempel-Ziv transformation on the input data vector. This means that a numerical sequence is converted to a binary string by substituting a 1 for each element greater than the median, and a 0 for each element equal to or less than the median. Sequence positions are retained. For example, Lempel-Ziv compression of a sine wave, transforms it into a square wave. Lempel-Ziv transformed data (integers) are fully amenable to recurrence analysis using the Hamming norm.

---

60

**JOINT.EXE** (File-Joining)

Programmer: C.L. Webber Jr., Public Domain
Usage: JOINT FILE1_IN FILE2_IN > FILE_OUT

Program to compute the relative and absolute overlap shared between intervals in two coupled files. The number of intervals in each input file can be different, but the sum of the intervals should be near the same total time. Relative overlap (percent) and absolute overlaps (units) are defined separately from the perspective of file_1in and the perspective of file_2in. The output consists of an [N,3] matrix as follows.

Column 1 = percent overlap of file 1 with file 2
Column 2 = percent overlap of file 2 with file 1
Column 3 = absolute overlap of file 1 with file 2

Each of the columns of the output matrix can be split off into three vectors and submitted to recurrence analysis.

---

**DATA FILES** (representative mathematical, physiological and linguistic systems)

Twenty-one ASCII files are provided for careful analysis by the recurrence quantification software. Four classes of examples are represented.

1. Mathematical examples (n=10)

   WHITE: Uniform random, white noise process (1000 points)

   BROWN: Brownian motion, integrated white noise process (1000 points)

   SINE: Ten-cycle sine wave (1000 points)

   SINENOIS: Ten-cycle sine wave with added uniform white noise (1000 points)

   HENXP: Hénon X variable for 16-point periodic process (1000 points)

   HENYP: Hénon Y variable for 16-point periodic process (1000 points)

   HENXC: Hénon X variable for multi-point chaotic process (1000 points)

   HENYC: Hénon Y variable for multi-point chaotic process (1000 points)

   LOGISP: Logistic difference equation, period-4 region (1000 points)

   LOGISC: Logistic difference equation, chaotic region (1000 points)

2. Discrete physiological examples (n=2)

   RRINT: Time intervals (sec) between sequential R waves of the human electrocardiogram (2334 points)

   RESPPER: Time intervals (sec) between breaths of an awake rodent challenged with carbon dioxide (low, high, low) in its environment (5703 points)

3. Continuous physiological examples (n=4)

   EMGCONT: Control electromyogram (amplified volts) from a human biceps muscle digitized at 1000 Hz (8000 points)
   EMGFAT: Fatigue electromyogram (amplified volts) from a human biceps muscle digitized at 1000 Hz (8000 points)
   ECG: Standard human limb lead II electrocardiogram (amplified volts) digitized at 1000 Hz (9000 points)
   PTRACH: Tracheal pressure trace from a spontaneously breathing rodent (8100 points)

4. Linguistic examples (n=5)

HIVDNA: DNA code for Human Immunodeficiency Virus (HIV)
(1=A, 2=C, 3=G, 4=T) (9770 points)

HAMLET: English letter code of Dr. Suess' poem, "Green Eggs and Ham"
(1=A, 2=B, 3=C, ..., 25=Y, 26=Z) (2438 points)

HAMWRD: English word code of Dr. Suess' poem, "Green Eggs and Ham"
(1=first word, 2=second word, 3=third word, ..., 50=last word) (812 points)

SPECTA: Amino acid chain of protein alpha-spectrin encoded by integers 1-20
(2429 points)

SPECTB: Amino acid chain of protein beta-spectrin encoded by integers 1-20
(2137 points)

---

**COMPUTATIONAL DERIVATION OF THE RECURRENCE MATRIX**

RQA employs the method of time delays to embed experimental data into higher dimensions. Explicit examples of how distance matrices (DM) and recurrence matrices (RM) are constructed for Euclidean, maximum and minimum norms are detailed below for a contrived time-series vector (TS) with 29 elements.

TS = [3.7, 9.2, 2.1, -5.4, 0.0, -10.9, 9.2, 3.1, 1.7, 1.8,
-0.3, -4.9, 2.7, 3.5, 7.5, -9.9, -9.9, -4.7, 1.3, 2.7,
7.6, 3.9, 7.3, 8.0, 0.3, -1.9, 5.1, 8.8, 8.2]

For DELAY=8, EMBED=4, FIRST=1 and LAST=5, the following 5 time-delay vectors are constructed in specified units (V) or rank order (R). (Note, tied rank orders are legal, but not illustrated here.)

V1 = [+3.7, +1.7, -9.9, +0.3]
V2 = [+9.2, +1.8, -4.7, -1.9]
V3 = [+2.1, -0.3, +1.3, +5.1]
V4 = [-5.4, -4.9, +2.7, +8.8]
V5 = [+0.0, +2.7, +7.6, +8.2]

R1 = [4, 3, 1, 2]
R2 = [4, 3, 1, 2]
R3 = [3, 1, 2, 4]
R4 = [1, 2, 3, 4]
R5 = [1, 2, 3, 4]

Comparison of the 5 vectors constructs a single 5*5 recurrence matrix of distances for each of the 4 norm types. For example, the Euclidean distance in the distance matrix (DM) between vectors V4 and V5 is calculated as follows.

DM(Euclid) = SQRT (SQR(-5.4-0.0) + (SQR(-4.9-2.7) + SQR(2.7-7.6) +
SQR(8.8-8.2)) = 10.549

To compute the maximum and minimum distances between vectors V4 and V5, the vectors are compared element by element. By definition, maximum and minimum distances are simply the maximum and minimum differences, respectfully.

DM(reject) = ABS(-5.4-0.0)) = 5.4
DM(max) = ABS(-4.9-2.7)) = 7.6 (largest difference)
DM(reject) = ABS(2.7-7.6)) = 4.9
DM(min) = ABS(8.8-8.2)) = 0.6 (smallest difference)

To compute the distance between the ordered vectors R4 and R5, the vectors are compared order by element order.

DM(order) = ABS(1-1) + ABS(2-2) + ABS(3-3) + ABS(4-4) = 0

This procedure is repeated for each cell, giving the following results for DM(i,j). Only the distances in the upper triangle are shown since the lower half is perfectly symmetrical. Note that the central diagonal is designated by 0.000 distances (vector identity matches).

DM(Euclid norm) =
[1,5] = 19.579; [2,5] = 18.405; [3,5] = 7.919; [4,5] = 10.549; [5,5] = 0.000
[1,4] = 18.904; [2,4] = 20.671; [3,4] = 9.647; [4,4] = 0.000
[1,3] = 12.452; [2,3] = 11.825; [3,3] = 0.000
[1,2] = 7.883; [2,2] = 0.000
[1,1] = 0.000

DM(max norm) =
[1,5] = 17.500; [2,5] = 12.300; [3,5] = 6.300; [4,5] = 7.600; [5,5] = 0.000
[1,4] = 12.600; [2,4] = 14.600; [3,4] = 7.500; [4,4] = 0.00
[1,3] = 11.200; [2,3] = 7.100; [3,3] = 0.000
[1,2] = 5.500; [2,2] = 0.000
[1,1] = 0.000

DM(min norm) =
[1,5] = 1.000; [2,5] = 0.900; [3,5] = 2.100; [4,5] = 0.600; [5,5] = 0.000
[1,4] = 6.600; [2,4] = 6.700; [3,4] = 1.400; [4,4] = 0.000
[1,3] = 1.600; [2,3] = 2.100; [3,3] = 0.000
[1,2] = 0.100; [2,2] = 0.000
[1,1] = 0.000

DM(order norm) =
[1,5] = 8; [2,5] = 8; [3,5] = 4; [4,5] = 0; [5,5] = 0
[1,4] = 8; [2,4] = 8; [3,4] = 4; [4,4] = 0
[1,3] = 6; [2,3] = 6; [3,3] = 0
[1,2] = 0; [2,2] = 0
[1,1] = 0

These distances are all expressed in absolute units and can be retained as such by selecting rescaling option 1.

By selecting rescaling option 2, RM distances are normalized to the mean distance of the RM by dividing each cell by the absolute mean distance and multiplying by 100. All central diagonal values of zero are excluded from the calculation of mean distance.

DM mean distance (Euclid norm) = 13.783 = 100.0%
DM mean distance (max norm) = 10.220 = 100.0%
DM mean distance (min norm) = 2.310 = 100.0%
DM mean distance (order norm) = 8.000 = 100.0%

By selecting rescaling option 3, DM distances are normalized to the maximum distance of the DM by dividing each cell by the absolute maximum distance and multiplying by 100. Be careful not to confuse max distance and max norm which are different.

DM max distance (Euclid norm) = 20.671 = 100.0%
DM max distance (max norm) = 17.500 = 100.0%
DM max distance (min norm) = 6.700 = 100.0%
DM max distance (order norm) = 6.700 = 100.0%

Recurrence matrices are derived from distance matrices by setting a RADIUS threshold. As shown below (for absolute distances), the Heaviside function assigns values of 0 or 1 to array elements. The RADIUS parameter is always relative to the reported MAXDIST, whether it be expressed in absolute units or relative units. Only those distances in RM[I,J] equal to or less than the RADIUS are defined as recurrent points at coordinates I, J.

RM(Euclid norm with RADIUS of 8.0 ) =
[1,5] = 0; [2,5] = 0; [3,5] = 1; [4,5] = 0; [5,5] = 1
[1,4] = 0; [2,4] = 0; [3,4] = 0; [4,4] = 1
[1,3] = 0; [2,3] = 0; [3,3] = 1
[1,2] = 1; [2,2] = 1
[1,1] = 1

RM(max norm with RADIUS of 12.3) =
[1,5] = 0; [2,5] = 1; [3,5] = 1; [4,5] = 1; [5,5] = 1
[1,4] = 0; [2,4] = 0; [3,4] = 1; [4,4] = 1
[1,3] = 1; [2,3] = 1; [3,3] = 1
[1,2] = 1; [2,2] = 1
[1,1] = 1

RM(min norm with RADIUS of 1.2) =
[1,5] = 1; [2,5] = 1; [3,5] = 0; [4,5] = 1; [5,5] = 1
[1,4] = 0; [2,4] = 0; [3,4] = 0; [4,4] = 1
[1,3] = 0; [2,3] = 0; [3,3] = 1
[1,2] = 1; [2,2] = 1
[1,1] = 1

RM(order norm with RADIUS of 0) =

[1,5] = 0; [2,5] = 0; [3,5] = 0; [4,5] = 1; [5,5] = 1
[1,4] = 0; [2,4] = 0; [3,4] = 0; [4,4] = 1
[1,3] = 0; [2,3] = 0; [3,3] = 1
[1,2] = 1; [2,2] = 1
[1,1] = 1

RQA looks for patterns among these recurrent points, and this need not/must not be done manually (too subjective). Thus, pattern recognition algorithms (very objective) are written into the many recurrence programs to define the 8 fundamental RQA variables: %RECUR, %DETERM, LMAX, ENTROPY, TREND, VMAX, %LAMINAR and TRAPTIME.

---

**IMPORTANT CAUTIONS**

1. Setting the DELAY parameter can be difficult. Program ACOR uses standard linear, one-dimensional algorithms that assume data stationarity and normal data distributions. Insofar that input data may violate these assumptions, alternate methods are available for estimation of the best DELAY values (e.g. mutual information content). Also, caution should be exercised when computing auto-correlations over large data sets which increases the probability for nonstationarity).

2. The EMBEDDING DIMENSION can be selected too low. If this parameter is set too small, the dynamic system under study will be under-determined. In theory, one could estimate the EMBEDDING DIMENSION by computing correlation dimensions, but such calculations are fraught with problems for short, noisy, or non-stationary data sets. It is usually better to overestimate the EMBEDDING DIMENSION, but not too high or noise will amplify. For physiological data we typically start with embedding values of 10 to 20 and work downward from there. For typical noisy data, the relationship between the embedding dimension (M) and the number of operating variables (N) can be estimated as: M = 2*N + 1.

3. The size of FILE_IN and/or EPOCH can be too short. If the recurrence plot MATRIX size is too small, zero values will appear in the ENTROPY variable. In such cases, the statistical validity of most of the computed recurrence plot variables are subject to question.

4. The RANDSEQ option can be misunderstood. Data randomization allows the evaluation of the deterministic content of the normally-sequenced input data. For example, comparison of %DETERM and ENTROPY values from a signal before and after randomization reveals the level of deterministic structuring originally present in the data. The more deterministic the input signal, the more the %DETERM and ENTROPY values will be affected (decreased) by randomization. By contrast, stochastic (high dimensional) data remain unaffected by such randomization. When comparing randomized and native versions of any one data set, care must be taken to assure that the RQA

parameter values are identical for each run. Randomization procedures invoked within RQA programs are performed locally within the specified time window (restricted to size of epoch). An alternate approach to attempt is the randomization of the entire time series of the selected dynamic before submitting the file for RQA (see program RANDSEQ).

5. The RESCALE option can be confusing. Choosing the absolute option, distances in the recurrence matrix are expressed in absolute units (not rescaled). Normalizing the distances to MAXDIST rescales the recurrence matrix over the unit interval (0-100%), but the presence of a single outlier can lead to the serious compression of remaining distances, causing %RECUR and %DETERM to greatly increase. Consequently, episodic computations (e.g. RQE) can lead to high variability in recurrence variables. If necessary, this effect can be blunted by normalizing the distances to MEANDIST which minimizes the influence of large distances (but also assumes that the distributed distances as Gaussian). The RADIUS parameter, of course, is always relative to MAXDIST no matter how the distances are rescaled.

6. The RADIUS parameter can be set too high. In recurrence plot analysis, interest centers on defining recurrences within local neighborhoods. The larger the RADIUS the broader the neighborhood. In fact, when RADIUS equals or exceeds the MAXDIST, every point in the recurrence plot is recurrent (full saturation). In such cases, beautiful pictures are generated depending upon how parameter COLORBND is set, but they have no known meaning (global RQA?). If the RADIUS parameter is too high, %RECUR and % DETERM values will tend to saturate at 100%, and ENTROPY, DMAX and VMAX will maximize. The RADIUS parameter should be set low, but not so low such that %RECUR approaches 0.0%, meaning that the statistical sampling of recurrent points is inadequate. For practical purposes, program RQS or KRQS should be used to scale %recurrence with increasing values of RADIUS. The log (%RECUR) is then plotted as a function of log (RADIUS). The proper RADIUS is selected just above the noise floor where the curve swings up from flat portion of the curve (hockey stick effect) at low RADIUS values. If there is no flat noise floor (digitization effect), the RADIUS can be selected that gives a non-log %RECUR value between 0.1% and 2.0%. Larger RADIUS values up to 5% may be necessary to obtain values for %LAMINAR and TRAPTIME. Note also that systems with distinct singularities (plateau pauses in the time series may normally have %RECUR values way over 5%, but this is normal and acceptable.

7. RQA technology can be applied to experimental data too quickly. It is strongly recommended that new users study known mathematical examples with RQA before applying RQA procedures to unknown experimental data. For example, it is very instructive to analyze sine waves of different frequencies and noise characteristics, as well as "classic" equations defining chaotic attractors (Hénon, Lorenz, Rössler, etc.). Programs such as RQS and RQE allow the convenient adjustment of multiple RQA parameters to examine the effect of RQA variables. Also, programs RQD and KRQD are particularly useful in correlating recurrence plot patterns with specific positions in the time (or spatial) series.

8. Too much emphasis can be placed on the curious recurrence plots. Key information derived from input data is represented in the recurrence variables, not the plots themselves. It is very easy to be enthralled with beautiful graphs, forgetting that the true diagnosis resides within the objective, quantitative variables, not their subjective, qualitative counterparts.

9. The nonlinearity and multi-dimensionality of RQA may not be fully appreciated. RQA can characterize systems that consist of multiple variables possessing nonlinear interactions (feedbacks, feed-forwards). The input data is neither constrained to any type of statistical distribution, nor must the data be stationary over time. There is also no need to know the mathematical rules (if any) governing the system under study. Although embedding procedures allow single observables (surrogate variables) to capture the participation of multiple variables, it is not possible to exactly identify those (unmeasured) participants. It remains the full responsibility of the investigator to be aware of possible candidate variables in each specific system being analyzed. Users must also be aware that real-world noise in data under examination can also influence recurrence results, inflating the dimension.

10. Certain recurrence variables can be useless in some situations. Thus, for highly periodic systems which generate long diagonal lines of different lengths, ENTROPY values are greatly inflated when, in fact, they should be near zero. The reason for this is that the diagonal lines are artificially truncated at the boarder of the recurrence matrix (edge effect) giving deceiving broad distributions of line lengths and excessively high ENTROPY values. Likewise, LMAX has been shown to be inversely proportional to positive Lyapunov exponents, but only for chaotic dynamics. The lesson is that decreases in LMAX values do not necessarily imply increases in positive Lyapunov exponents.

11. Confusion can exist between continuous versus discrete representations of the same system. For example, the heart beat can be described as either continuous ECG waveforms or discrete R-R intervals. Patterns of recurrent points from these two different representations have different meanings. Thus, a string of diagonal recurrent points for the continuous ECG signal means that the trajectories of the potentials are similar. That is, the shapes of the subsequent ECG waveforms match. On the other hand, a string of diagonal recurrent points for the discontinuous R-R intervals means that there is a run of beats with very similar timing characteristics. So, depending upon the system's representation selected, one can study the shape of waveforms or the timing of rhythmical events. These are simply alternate perspectives on the same system.

12. There is very little experience with kross-recurrence analysis which is only recently implemented. Several associated kross programs (leading K in program name) are provided to encourage exploration of this new type of nonlinear cross-correlation analysis. In this context, however, much care must be taken to assure that the parallel files to be correlated represent continuous data digitized at the same rate. Discontinuous data must first be resampled and rescaled by proper implementation of attendant programs RESAMPLE.EXE and

UNITINT.EXE before attempting cross-recurrence analysis. Failure to take note of these cautions will result in spurious results.

13. RQA as a methodology is not yet standardized. RQA procedures are currently being implemented around the world, but the jury is still out regarding a consensus report. It may be possible that different fields of inquiry (e.g. physiology versus protein chemistry) may each require unique sets of standards. What is becoming clearer is that the high sensitivity of RQA may have utility in diagnosing state fluctuations in living systems. Such sensitivity using commonly measured variables (e.g. cardiac intervals, blood pressure, tidal volume, oxygen saturation, body temperature, etc.) may turn out to be significant in various clinical situations, but this will require patient, systematic and detailed investigations by qualified clinical researchers.

14. The proper selection of recurrence parameters and correct interpretation of recurrence variables are no easy tasks. With very little effort users will be able to generate beautiful recurrence plots, yet have no idea as to what they mean. The caution here is for new users to take their time in understanding the theory behind recurrence analysis. This will be of inestimable value when deciding how to properly set the many recurrence parameters. And skill comes only with first-hand experience.

15. There is scant information on the meaning of vertical line structures in recurrence plots that give rise to two new recurrence variables, %laminarity and trapping time. Whereas diagonal line structures denote parallel trajectories, there are no corresponding explanations for vertical bandings. These particular lines may be hallmark features of terminal dynamics (continuous systems that rest at singular points). Alternatively, these same features may denote situations of recurrence saturation which arise when the radius parameter is set too high. In such cases they would be artifacts of non-sparse recurrence matrices.

16. There is little experience using %RECUR as a recurrence parameter. This recent implementation, however, gives exciting functionality to the analysis of nonlinear dynamical systems. Now it is possible to study recurrence variables based on similar %RECUR values (e.g. same number of recurrent points for any specified window size), not wandering %RECUR values. By analogy, what the voltage-clamp technique did for excitable cells by elucidating membrane current contributions, so too the %RECUR- clamp technique may dissect out dynamical features of complex systems.

17. Order recurrence plots are relative new and there is not much experience using this type of norming. It should be appreciated that order recurrences perform a high-pass filter function on the input data and ignore baseline shifts.

18. The two recurrence matrix programs (RQM and KRQM) allow for computation of all eight RQA parameters from any defined square recurrence matrix. These matrices could represent real systems as diverse as grids of protein-protein binding iterations to geographical maps of power grids. It is anticipated that

users will contribute many other practical examples to this unique recurrence approach.

19. The interpretation of recurrence variables can be decidedly difficult. Except for recurrence intervals which can carry units of time (and from which nonlinear spectral plots can be derived, calibrated in units of frequency), it is not always possible to endow most other RQA variables with concrete, physical meanings. Thus, recurrence analysis is very conceptual, but nonetheless very useful in delineating states and state changes experienced by dynamical systems.

20. Recurrence quantification analysis (RQA) is definitely not intended as a be-all or end-all of analyses on dynamical systems. In fact, users are encouraged to couple the linear technique of principle component analysis (PCA) with RQA to evaluate how recurrence variables interact with each other (parallel versus orthogonal behaviors).

---

**PROBLEMS, QUESTIONS AND SUGGESTIONS**

Please contact Chuck Webber with any practical questions or problems you have regarding RQA procedures. Despite exhaustive software testing, helpful suggestions that would improve our programs and eliminate bugs are also encouraged. When attempting to analyze your own experimental data using RQA software, do proceed cautiously and at your own risk. Thank you for your interest in and support of RQA as a nonlinear, multidimensional approach to dynamical systems. The list of users from many different fields of study has grown dramatically since 1992. And the professional literature on RQA continues to expand into various academic disciplines. Lastly, do let us know of your successes, frustrations, and failures with RQA which is but one nonlinear approach to complex dynamical systems.

Charles L. Webber, Jr., Ph.D.
Professor Emeritus of Cell and Molecular Physiology
Stritch School of Medicine
Loyola University Chicago
Health Sciences Division
2160 South First Avenue
Maywood, IL 60153-3328
U.S.A.

(C) 708-638-7497
(F) 708-216-8355
(E) cwebber@luc.edu
(U) http://cwebber.sites.luc.edu

---