Chapter 6
Approximate Recurrence Quantification Analysis (aRQA) in Code of Best Practice

Stephan Spiegel, David Schultz and Norbert Marwan

Abstract Recurrence quantification analysis (RQA) is a well-known tool for studying nonlinear behavior of dynamical systems, e.g. for finding transitions in climate data or classifying reading abilities. But the construction of a recurrence plot and the subsequent quantification of its small and large scale structures is computational demanding, especially for long time series or data streams with high sample rate. One way to reduce the time and space complexity of RQA are approximations, which are sufficient for many data analysis tasks, although they do not guarantee exact solutions. In earlier work, we proposed how to approximate diagonal line based RQA measures and showed how these approximations perform in finding transitions for difference equations. The present work aims at extending these approximations to vertical line based RQA measures and investigating the runtime/accuracy of our approximate RQA measures on real-life climate data. Our empirical evaluation shows that the proposed approximate RQA measures achieve tremendous speedups without losing much of the accuracy.

Keywords Recurrence plots · Recurrence quantification analysis · Time series · (Dis)similarity measures · Approximate solutions

S. Spiegel (✉) · D. Schultz
DAI-Lab, Technische Universität Berlin, Ernst-Reuter-Platz 7,
10587 Berlin, Germany
e-mail: spiegel@dai-lab.de

D. Schultz
e-mail: schultz@dai-lab.de

N. Marwan
Potsdam Institute for Climate Impact Research, Pappelallee 20,
14412 Potsdam, Germany
e-mail: marwan@pik-potsdam.de

C.L. Webber, Jr. et al. (eds.), *Recurrence Plots and Their Quantifications: Expanding Horizons*, Springer Proceedings in Physics 180,
DOI 10.1007/978-3-319-29922-8_6

6.1 Introduction

In recent years, recurrence quantification analysis (RQA) has gained popularity in the time series community [1–3], where recurrence plot-based tools have been developed (i) to measure the pairwise (dis)similarity between temporal measurements based on co-occurring patterns [3, 4], (ii) for classification purposes in different scientific disciplines [5–9], (iii) to detect regime transitions [10–12], or even (iv) to study interrelationships and synchronization between different dynamical systems [13–15].

Since the quantification of recurring patterns is computational expensive, speedup techniques [16] and approximations [17] have been proposed. Speedup techniques commonly use distributed computing that ensures exact RQA results, e.g. by performing parallel processes on multiple Graphic Processing Units (GPUs), whereas approximation techniques estimate the RQA measures by means of less computational expensive algorithms. Given a time series with about one million data points, distributed computing with two GPUs has been shown to reduce the RQA calculation time by 1–2 orders of magnitude [16]. However, this work demonstrates that the proposed approximations [17] are able to reduce the RQA calculation time (for the same one million measurements) by 4 orders of magnitude. This tremendous speedup makes the approximation approach extremely valuable for many real-life data analysis tasks, although it does not yield exact results.

In this work we extend the approximation approach [17] to vertical line based measures, assess the runtime of our approximate RQA measures for relatively long time series (from climate impact research), and investigate the use our approximate RQA measures for transition detection.

6.2 Background and Notation

6.2.1 Recurrence Plots (RPs)

Recurrence plots (RPs) have been introduced to study the dynamics of complex systems that is represented in an m-dimensional phase space by its phase space trajectory $\mathbf{x}_i \in \mathbb{R}^m$ (assuming discrete sampling, $i = 1, \ldots, N$) [18]. A phase space trajectory can be reconstructed from a time series u_i ($t = i\Delta t$, where Δt is the sampling time) by different embedding schemes. The most frequently used scheme is the time delay embedding [19],

$$\mathbf{x}_i = \left(u_i, u_{i+1}, \ldots, u_{i+(m-1)\tau}\right), \tag{6.1}$$

with m the embedding dimension and τ the embedding delay. Both parameters can be estimated from the original data using false nearest neighbors and mutual information [20]. In the following we only consider the trajectory $\mathbf{x}$ and no longer the underlying time series u. That means the process of creating $\boldsymbol{x}$ from u by time delay

embedding is considered to be completed. Later on, we will apply time delay embedding to the trajectory $\boldsymbol{x}$ again. It is important to distinguish between both embedding procedures. The first is used for reconstruction purposes and the latter is used to express RQA-measures in a way that allows fast computation.

A RP is a 2-dimensional representation of those times when the phase space trajectory $\mathbf{x}_i$ recurs. As soon as a dynamical state at time j comes close to a previous (or future) state at time i, the recurrence matrix $\mathbf{R}$ at (i,j) has an entry one [20]:

$$R_{i,j} := \Theta(\varepsilon - \|\mathbf{x}_i - \mathbf{x}_j\|), \qquad i,j = 1,\ldots,N, \tag{6.2}$$

where $\|\cdot\|$ is a norm (representing the spatial distance between the states at times i and j), ε is a predefined recurrence threshold, and Θ is the Heaviside function (ensuring a binary $\mathbf{R}$). The RP has a square form and usually the identity $R_{i,i} \equiv 1$ is included in the graphical representation, although for calculations it might be useful to remove it [20]. The graphical representation of the RP allows to derive qualitative characterizations of the dynamical systems. For the quantitative description of the dynamics, the small-scale patterns in the RP can be used, such as diagonal and vertical lines. The histograms of the lengths of these lines are the base of the recurrence quantification analysis (RQA) developed by Webber and Zbilut and later by Marwan et al. [7, 21, 22].

6.2.2 Recurrence Rate (RR)

The simplest measure of RQA is the density of recurrence points in the RP, the recurrence rate:

$$RR := \frac{1}{N^2} \sum_{i,j=1}^{N} R_{i,j}, \tag{6.3}$$

that can be interpreted as the probability that any state of the system will recur.

6.2.3 Determinism (DET)

The fraction of recurrence points that form diagonal lines of minimal length μ is the determinism measure:

$$DET^{(\mu)} := \frac{\sum_{l=\mu}^{N} l \cdot D(l)}{\sum_{i,j=1}^{N} R_{i,j}} = \frac{\sum_{l=\mu}^{N} l \cdot D(l)}{\sum_{l=1}^{N} l \cdot D(l)} \tag{6.4}$$

where

$$D(l) := \sum_{i,j=1}^{N} \left\{ \left(1 - R_{i-1,j-1}\right) \cdot \left(1 - R_{i+l,j+l}\right) \cdot \prod_{k=0}^{l-1} R_{i+k,j+k} \right\}$$

is the histogram of the lengths of the diagonal lines. The understanding of 'determinism' in this sense is of heuristic nature.

6.2.4 Average Diagonal Line Length (L)

The average length of all diagonal lines (of at least length μ) in the RP is

$$L^{(\mu)} := \frac{\sum_{l=\mu}^{N} l \cdot D(l)}{\sum_{l=\mu}^{N} D(l)}, \tag{6.5}$$

and can be interpreted as the mean prediction time. As the diagonal lines in the RP are related to the divergence behavior of the phase space trajectory, its relationship with the Lyapunov exponents are obvious. Indeed, there is clear link between the distribution of the diagonal line lengths and the K_2 entropy of the system [23].

6.2.5 Laminarity (LAM)

Similar to the measure *DET*, the fraction of recurrence points that form vertical lines of a certain minimum length μ can be calculated. The corresponding measure is called laminarity:

$$LAM^{(\mu)} := \frac{\sum_{l=\mu}^{N} l \cdot V(l)}{\sum_{l=1}^{N} l \cdot V(l)}, \tag{6.6}$$

with

$$V(l) := \sum_{i,j=1}^{N} \left\{ \left(1 - R_{i,j-1}\right) \cdot \left(1 - R_{i,j+l}\right) \cdot \prod_{k=0}^{l-1} R_{i,j+k} \right\},$$

the histogram of the lengths of the vertical lines in the RP. Vertical (as well as horizontal) lines appear when states do not change or change only very slowly, as it is typical for intermittence and laminar regimes [7].

Further measures have been introduced that incorporate such line length distributions and also network properties [20, 24]. All these measures can be used to classify different dynamical regimes and to detect their transitions.

6.3 Approximate Recurrence Quantification Analysis

In this section we propose an alternative way of computing the RQA measures introduced previously. Our idea can be expressed as follows: (1) We propose two novel quantification techniques, namely pairwise proximities *PP* and stationary states *SS*, which account for diagonal and vertical lines by means of an embedded trajectory. (2) Based on *PP* and *SS* we introduce alternative formulations of the traditional RQA measures that are equivalent to the original formulations, provided that the phase space norm that measures the spatial distances (e.g. in (6.2)) is the maximum-norm, defined by $\|\mathbf{y}\|_\infty = \max_i |\mathbf{y}_i|$. (3) Using these new formulations, the RQA measures can be computed quickly if the similarity threshold is zero ($\varepsilon = 0$). (4) If the similarity threshold is greater than zero, we first discretize the data and then set the threshold to zero in order to make use of fast algorithms that are facilitated by our alternative formulations. In this case—due to discretization—we only get an approximation of the exact RQA measures.

In earlier work [17] we have proven the equivalence between our alternative and the original formulation of the RQA measures and, furthermore, analyzed the approximation error theoretically. Moreover, we have provided detailed information on the discretization and employed algorithms [17], which have complexity of $(N \log(N))$. Our implementation of the discretization and employed algorithms can be found in Sect. 6.4.

Important Note. In this section we assume that the similarity threshold is zero. That means the recurrent states we aim at quantifying are only states that are equal. This case is relevant if the trajectory $\mathbf{x}$ is discrete-valued or has been discretized beforehand in order to compute the approximate RQA-measures. To be more clear on the role of the threshold, we define *PP* and *SS* for general $\varepsilon \geq 0$, but the reader may imagine that in application of the fast (approximate) RQA algorithms we have $\varepsilon = 0$.

Given a phase space trajectory $\mathbf{x}$, the number of pairwise proximities *PP* can be defined as follows:

$$PP^{(\nu)} := \sum_{i,j=1}^{N-\nu+1} \Theta(\varepsilon - \|\mathbf{x}_i^{(\nu)} - \mathbf{x}_j^{(\nu)}\|), \tag{6.7}$$

where $\mathbf{x}^{(\nu)}$ is a time-delay embedded version of the trajectory $\mathbf{x}$ with embedding dimension $\nu \in \mathbb{N}$ and time-delay 1, i.e.,

$$\mathbf{x}_i^{(\nu)} = (\mathbf{x}_i, \ldots, \mathbf{x}_{i+\nu-1}), \qquad i = 1, \ldots, N - \nu + 1. \tag{6.8}$$

We want to emphasize that the key idea of our quantification techniques (*PP* and *SS*) is to embed the trajectory, since recurrent states of embedded trajectories indicate recurrent sequences in the original trajectory $\mathbf{x}$ if the phase space norm is $\|\cdot\|_\infty$. To see this, assume that, for instance, the recurrence plot of the embedded trajectory $\mathbf{x}^{(2)}$ indicates a recurrence point at position (i, j), that means

$$\|\mathbf{x}_i^{(2)} - \mathbf{x}_j^{(2)}\|_\infty \leq \varepsilon.$$

By definition of $\|\cdot\|_\infty$ and the trajectory embedding, this is equivalent to

$$\|\mathbf{x}_i - \mathbf{x}_j\|_\infty \leq \varepsilon \quad \text{and} \quad \|\mathbf{x}_{i+1} - \mathbf{x}_{j+1}\|_\infty \leq \varepsilon,$$

which exactly means that the recurrence plot of the original trajectory $\mathbf{x}$ contains a diagonal line of length 2 starting at position (i,j). Note that this equivalence is not true for arbitrary norms.

Our implementation of the general time delay embedding, (6.1), can be found in Sect. 6.4.1.

As shown in [17], if $\varepsilon = 0$, the measure of pairwise proximities $PP^{(\nu)}$ can also be interpreted as the sum over the squared frequencies of recurring states, which can be determined using the histogram $h(\mathbf{x}^{(\nu)})$ of the embedded trajectory:

$$PP^{(\nu)} = h(\mathbf{x}^{(\nu)}) \cdot h(\mathbf{x}^{(\nu)}). \tag{6.9}$$

In (6.9) the histograms are represented as vectors containing the frequencies of the elements in $\mathbf{x}^{(\nu)}$ and the dot denotes the inner product, defined by $u \cdot w = \sum_i u_i w_i$. This relation is the key for the fast computation of the RQA-measures since the histograms can be obtained in $\mathcal{O}(N \log(N))$, where N is the length of the trajectory $\mathbf{x}$. It is important to note that (6.9) does only hold for $\varepsilon = 0$. This is the reason why the data has to be discretized if $\varepsilon > 0$ is required.

Based on our definition of pairwise proximities PP we can introduce alternative formulations for the original diagonal line based RQA measures introduced in Sect. 6.2. In the following we discuss an alternative formulation for recurrence rate RR, determinism DET, average diagonal line length L, and laminarity LAM.

Before moving on to more advanced recurrence quantification measures, we want to provide an image representation of pairwise proximities PP. Figure 6.1 shows the recurrence plot $\mathbf{R}$ of a discrete-valued sample trajectory $\mathbf{x}$ and its corresponding histogram $h(\mathbf{x}^{(\nu)})$ for trajectory embedding dimension $\nu = 1$ (note that $\mathbf{x}^{(1)} = \mathbf{x}$). The pairwise proximities $PP^{(1)}$ are equal to the total number of recurrence points in $\mathbf{R}$ and are given by the sum over the squared frequencies:

$$PP^{(1)} = h(\mathbf{x}^{(1)}) \cdot h(\mathbf{x}^{(1)}) = 3^2 + 5^2 + 2^2 = 38$$

6.3.1 Reformulation of Recurrence Rate (RR)

The pairwise proximities $PP^{(\nu)}$ for trajectory embedding dimension $\nu = 1$ can be interpreted as the number of recurrence points, which are traditionally expressed by the sum over all recurrence plot entries $\sum_{i,j=1}^{N} R_{i,j}$ (see 6.3). To compute the

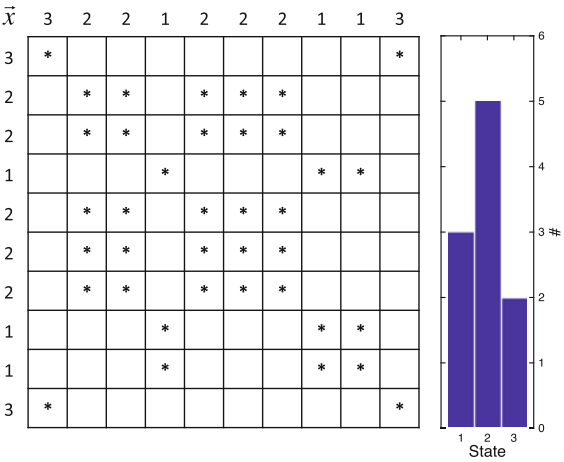

Fig. 6.1 Recurrence plot **R** of trajectory $\mathbf{x} = (3, 2, 2, 1, 2, 2, 2, 1, 1, 3)$ with similarity threshold $\varepsilon = 0$ (*left*) and its histogram $h(\mathbf{x}^v)$ for embedding dimension $v = 1$ (*right*), showing the frequencies of recurring states. The pairwise proximities $PP^{(1)}$ equal the total number of recurrence points in **R** and are given by the sum over the squared frequencies (see 6.9)

recurrence rate, the number of recurrence points is divided by the size of the recurrence plot, which is the squared length N^2 of the time series under study. Hence, the alternative way of computing the recurrence rate *RR* can be formalized as followed:

$$RR = PP^{(1)}/N^2. \tag{6.10}$$

For our sample trajectory **x** (shown in Fig. 6.1) with pairwise proximities $PP^{(1)} = 38$ and length $N = 10$ the recurrence rate is:

$$RR = 38/10^2 = 0.38$$

6.3.2 Reformulation of Determinism (DET)

The determinism *DET* can also be expressed in terms of pairwise proximities. Traditionally the determinism *DET* is described as the percentage of recurrence points which form diagonal lines (refer to 6.4). In the previous Sect. 6.3.1 we have already explained that the total number of recurrence points is equivalent to the pairwise proximities $PP^{(1)}$. Hence, the denominator of *DET* is known and it remains the question of how to compute the number of recurrence points that contribute to diagonal lines of minimum length μ. Our idea is to quantify the recurrence plot $\mathbf{R}^{(\mu)}$ of the embedded trajectory $\mathbf{x}^{(\mu)}$ in relation to the recurrence plot **R** of the original trajectory **x**. First note that each point in $\mathbf{R}^{(\mu)}$ indicates that there is a diagonal line of length $\geq \mu$ in **R**. Consequently, only lines we are interested in remain in $\mathbf{R}^{(\mu)}$. However, each diagonal line in $\mathbf{R}^{(\mu)}$ is $\mu - 1$ shorter than the corresponding line in **R**. Thus, we need to add the missing points. Evidently, the number of missing points is exactly given by "the number of diagonal lines of length $\geq \mu$" times $\mu - 1$, where "the

number of diagonal lines of length $\geq \mu$" in $\mathbf{R}$ is given by $(PP^{(\mu)} - PP^{(\mu+1)})$. To sum up, we have argued that

$$\sum_{l=\mu}^{N} l \cdot D(l) = PP^{(\mu)} + (PP^{(\mu)} - PP^{(\mu+1)}) \cdot (\mu - 1). \tag{6.11}$$

By simplifying (6.11) we achieve our alternative formulation for the determinism (proved in [17]), which is true for arbitrary similarity threshold $\varepsilon \geq 0$ and arbitrary minimum diagonal line length μ, provided that the phase space norm is the maximum-norm $\|\cdot\|_\infty$:

$$DET^{(\mu)} = \frac{\mu \cdot PP^{(\mu)} - (\mu - 1) \cdot PP^{(\mu+1)} \ldots - N}{PP^{(1)}}. \tag{6.12}$$

Depending on whether or not we want to include the recurrence points of the main diagonal in our calculation, we need to subtract N in the numerator (6.12).

Figure 6.2 illustrates how to employ the concept of pairwise proximities in order to compute the determinism for our sample time series $\mathbf{x}$ introduced in Fig. 6.1.

For example in Fig. 6.2, the determinism $DET^{(2)}$ for minimum diagonal line length $\mu = 2$ can be described as the number of recurrence points that rest on highlighted lines divided by the total number of recurrence points (i.e. 14/38). The total number of recurrence points is given by $PP^{(1)}$ (see Sect. 6.3.1) and the number of recurrence points that form diagonals of minimum length $\mu = 2$ can be expressed in terms of $PP^{(2)}$ and $PP^{(3)}$ (see 6.12). In Fig. 6.2, single circles illustrate the recurrences that are given by our formulation of pairwise proximities $PP^{(2)}$ for embedding dimension $\nu = 2$. By multiplying the length $\mu = 2$ and number $PP^{(2)}$ of the identified structures we quantify all recurrence points that rest on diagonal lines, including those of overlapping structures. To subtract recurrence points of overlapping struc-

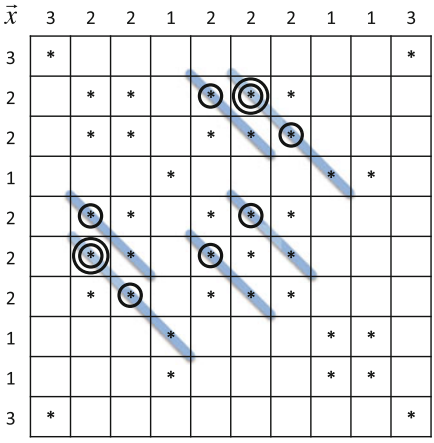

Fig. 6.2 Recurrence plot R of time series $\mathbf{x}$, where *highlighted lines* indicate diagonals that contribute the determinism DET, *single circles* illustrate recurrences that are given by our formulation of pairwise proximities $PP^{(2)}$ for embedding dimension $\nu = 2$, and *double circles* show recurrences that are quantified by $PP^{(3)}$ for $\nu = 3$ respectively

tures we compute the pairwise proximities $PP^{(3)}$ of higher embedding dimension $\nu = 2 + 1$, which are illustrated by double circles. Consequently, the determinism $DET^{(2)}$ for our sample time series $\mathbf{x}$ is computed in the following way:

$$DET^{(2)} = \frac{2 \cdot PP^{(2)} - 1 \cdot PP^{(3)}}{PP^{(1)}}$$
$$\left(= \frac{2 \cdot Single\ Circles - 1 \cdot Double\ Circles}{Total\ Number\ of\ Recurrences} \right)$$
$$= \frac{2 \cdot 8 - 1 \cdot 2}{38} = \frac{14}{38} \approx 0.37$$

6.3.3 Reformulation of Average Diagonal Line Length (L)

Given our new formulation for the determinism (see 6.12), the formalization of the average diagonal line length L in terms of pairwise proximities PP is straightforward. Informally speaking, L is defined as the number of recurrence points that form diagonals of minimum length μ divided by the number of diagonals of minimum length μ (see 6.5). We have already shown how to compute the first term or numerator in the previous Sect. 6.3.2. The second term or denominator can be computed by $PP^{(\mu)} - PP^{(\mu+1)}$, which is the number of diagonals with minimum length μ. Since we know that $PP^{(\mu)}$ accounts for all the diagonal line structures with minimum length μ including overlapping ones, we need to subtract the number of overlapping structures which are quantified by the term $PP^{(\mu+1)}$. Ultimately, under the same assumptions as for DET, our alternative formulation of L can be formalized as followed:

$$L^{(\mu)} = \frac{\mu \cdot PP^{(\mu)} - (\mu - 1) \cdot PP^{(\mu+1)} \cdots - N}{PP^{(\mu)} - PP^{(\mu+1)}}. \tag{6.13}$$

Same as for the determinism, we might not want to consider the main diagonal for our calculation and, thus, need to subtract N (the time series length) from the numerator.

For our sample time series $\mathbf{x}$ (shown in Fig. 6.2) we can compute the average diagonal line length for $\mu = 2$ as followed:

$$L^{(2)} = \frac{2 \cdot 8 - 1 \cdot 2}{8 - 2} = \frac{14}{6} \approx 2.33$$

6.3.4 Reformulation of Laminarity (LAM)

The laminarity is the percentage of recurrence points which form vertical lines (see 6.6) and **cannot** be computed by means of the PP measure, since it quantifies diagonal line structures. Therefore, we need to introduce a novel measure for stationary

states *SS*, which accounts for time intervals where the corresponding trajectory stays in the same phase space. Stationary states *SS* which stay stable for ν time points can be quantified in the following way.

$$SS^{(\nu)} := \sum_{i=1}^{N-\nu+1} \sum_{j=1}^{N} \Theta(\varepsilon - \|\mathbf{x}_i^{(\nu)} - \mathbf{1}^{(\nu)}\mathbf{x}_j\|), \tag{6.14}$$

where $\mathbf{1}^{(\nu)}\mathbf{x}_j = (\mathbf{x}_j, \dots, \mathbf{x}_j)$ is the concatenation of ν copies of $\mathbf{x}_j$. Hence *SS* accounts for states where all elements in $\mathbf{x}_i^{(\nu)} = (\mathbf{x}_i, \dots, \mathbf{x}_{i+\nu-1})$ are in a ε-neighborhood of $\mathbf{x}_j$, indicating that state $\mathbf{x}_i$ stays stationary for ν time points.

Analogously to (6.9) we can compute the stationary states efficiently in $\mathcal{O}(N \log(N))$ using histograms if $\varepsilon = 0$:

$$SS^{(\nu)} = \hbar(\mathbf{x}^{(\nu)}) \cdot \hbar(\mathbf{x}^{(1)}), \tag{6.15}$$

where $\hbar(\mathbf{x}^{(\nu)})$ is the *stationary state histogram* of the embedded trajectory, which—that is important—only accounts for stationary states of exact length ν (including overlapping structures in $\mathbf{x}$); and $\hbar(\mathbf{x}^{(1)})$ is the histogram of the original trajectory. Attention should be paid to the calculation of the inner product between stationary state histograms, since only frequencies of corresponding states are multiplied. For example the frequency of the stationary state $(1, 1)$ in $\mathbf{x}^{(2)}$ is multiplied with the frequency of state (1) in $\mathbf{x}$. Furthermore, it is important to mention that although **non**-stationary states may occur in an embedded trajectory (e.g. $(2, 1)$, in $\mathbf{x}^{(2)}$), their frequency in the corresponding stationary state histogram is always zero.

Given our new definition of stationary states *SS* and assuming that the phase space norm is the maximum-norm $\|\cdot\|_\infty$, we can compute the laminarity *LAM* for a given minimum vertical line length μ and an arbitrary threshold $\varepsilon \geq 0$ as follows:

$$LAM^{(\mu)} = \frac{\mu \cdot SS^{(\mu)} - (\mu - 1) \cdot SS^{(\mu+1)} \cdots - N}{SS^{(1)}}, \tag{6.16}$$

where the denominator $SS^{(1)}$ denotes the total number of recurrence points and the numerator $\mu \cdot SS^{(\mu)} - (\mu - 1) \cdot SS^{(\mu+1)}$ denotes the number of recurrence points that form vertical lines of minimum length μ. The thoughtful reader might have noticed that our new formulation of *LAM* and *DET* (6.16) and (6.12) resemble each other. The difference is that *DET* accounts for diagonal lines using *PP* and *LAM* quantifies vertical lines using *SS*. Therefore the proof of the *LAM* formula (6.16) is very similar to the proof of the *DET* formula presented in earlier work [17].

Figure 6.3 illustrates how to compute the laminarity $LAM^{(2)}$ for our sample trajectory $\mathbf{x}$. As defined in (6.16), the laminarity $LAM^{(2)}$ for minimum vertical (or rather horizontal) line length $\mu = 2$ can be computed in terms of $SS^{(1)}$, $SS^{(2)}$, and $SS^{(3)}$. For example in Fig. 6.3, the total number of recurrences points is described by the term $SS^{(1)}$, which can also be interpreted as the sum over the square frequencies of stationary states, given by the histogram $\hbar(\mathbf{x}^{(1)})$ (see Table 6.1).

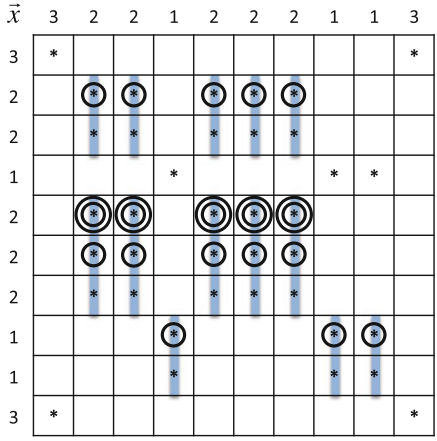

Fig. 6.3 Recurrence plot of trajectory $\mathbf{x}$ with similarity threshold $\varepsilon = 0$ and *highlighted vertical lines* of minimum length $\mu = 2$. The laminarity $LAM^{(2)} = 31/38$ is the percentage of recurrence points that form vertical lines of minimum length μ

Table 6.1 Stationary state histograms for our sample trajectory $\mathbf{x}$ in Fig. 6.3, showing the frequency of states that are stationary over μ time points

	State	#
$\hbar(\mathbf{x}^{(1)})$	[1]	3
	[2]	5
	[3]	2
$\hbar(\mathbf{x}^{(2)})$	[1,1]	1
	[2,2]	3
	[3,3]	0
$\hbar(\mathbf{x}^{(3)})$	[1,1,1]	0
	[2,2,2]	1
	[3,3,3]	0

Furthermore, the highlighted lines in Fig. 6.3 indicate all recurrence points that form vertical structures of minimum length $\mu = 2$, which can be quantified in terms of $SS^{(2)}$ and $SS^{(3)}$. For our sample trajectory $\mathbf{x}$ in Fig. 6.3, all stationary states $SS^{(2)}$ that are stable for 2 time points are illustrated by single circles. Moreover, double circles indicate stationary states $SS^{(3)}$ that are stable for 3 successive observations. The terms $SS^{(2)}$ and $SS^{(3)}$ can be computed by means of (6.14) and expressed in terms of the respective stationary state histograms $\hbar(\mathbf{x}^{(1)})$, $\hbar(\mathbf{x}^{(2)})$ and $\hbar(\mathbf{x}^{(3)})$:

$$
\begin{aligned}
SS^{(1)} &= \hbar(\mathbf{x}^{(1)}) \cdot \hbar(\mathbf{x}^{(1)}) = 38 \\
SS^{(2)} &= \hbar(\mathbf{x}^{(2)}) \cdot \hbar(\mathbf{x}^{(1)}) = 18 \\
SS^{(3)} &= \hbar(\mathbf{x}^{(3)}) \cdot \hbar(\mathbf{x}^{(1)}) = 5.
\end{aligned}
$$

The corresponding stationary state histograms for our plot in Fig. 6.3 are shown in Table 6.1. The multiplication of stationary state histograms is performed element-wise using the inner product, i.e. multiplying the frequencies of unique states with the corresponding stationary state counterparts and summing the results (see

Algorithm 4). For our sample trajectory $\mathbf{x}$, the inner product is $\hbar(\mathbf{x}^{(1)}) \cdot \hbar(\mathbf{x}^{(2)}) = 18$, since $3 \cdot 1 + 5 \cdot 3 + 2 \cdot 0 = 18$.

By multiplying the length $\mu = 2$ and the number $SS^{(2)}$ of identified stationary states we quantify all recurrence points that rest on highlighted lines, including those of overlapping vertical structures. To subtract the recurrence points of overlapping structures we compute the number of stationary states $SS^{(3)}$ with higher length $\mu = 2 + 1$, leading to the following formalization: $2 \cdot SS^{(2)} - 1 \cdot SS^{(3)}$. Eventually, we can compute the laminarity $LAM^{(2)}$ for the plot in Fig. 6.3 by dividing the amount of recurrence point that form vertical lines by the total number of recurrences $SS^{(1)}$:

$$\begin{aligned} LAM^2 &= \frac{2 \cdot SS^{(2)} - 1 \cdot SS^{(3)}}{SS^{(1)}} \\ &\left(= \frac{2 \cdot \textit{Single Circles} - 1 \cdot \textit{Double Circles}}{\textit{Total Number of Recurrences}} \right) \\ &= \frac{2 \cdot 18 - 1 \cdot 5}{38} = \frac{31}{38} \approx 0.82. \end{aligned}$$

Given our proposed definition of stationary states SS (6.14), we can also restate other vertical line based measures, such as trapping time TT or longest vertical line length V_{max}. However, this goes beyond the scope of this study.

6.4 Approximate Recurrence Quantification Analysis with MATLAB

Before we present our empirical results we want to discuss the implementation of our proposed approximate recurrence quantification analysis. The provided source code will help other researchers to reproduce our results and to continue with further ideas right where we left off. We decided to provide MATLAB code since it is very compact and often used in academia. However, our code snippets can also be executed in Octave, which is an open source alternative to MATLAB. Please note that our code is protected by copyright laws and is not provided for commercial use. If you plan to use our implementation for academic purpose (e.g. for reproduction of experimental results or further enhancements of the introduced concepts) we kindly remind you to cite this chapter.

In the following subsections we explain our implementation of: (i) time delay embedding (according to Takens' theorem), (ii) discretization (e.g. of a multivariate time series or an embedded phase space trajectory), (iii) pairwise proximities (for diagonal line based measures), (iv) stationary states (for vertical line based measures), and (v) our experimental protocol for the approximate recurrence quantification analysis of the Potsdam time series of hourly air temperature [16].

6.4.1 Time Delay Embedding

Our implementation of the time delay embedding is used for the following purposes: (i) given a sequence of temporal observations we aim at embedding the recorded time series into the phase space using predetermined parameters for the embedding dimension and time delay, (ii) given a (reconstructed) phase space trajectory we aim at embedding it (once again) in order to quantify recurring segments $PP^{(v)}$ and stationary intervals $SS^{(v)}$ with a certain number of time points v.

In general our time delay embedding function accepts an input time series of size $n \times d$, where n denotes the number of time points and d represents the dimensionality of the data. The function call furthermore requires us to specify the embedding dimension m and time delay τ. It is important to note that, in contrast to the time series embedding (i), the trajectory embedding (ii) always assumes unit time delay ($\tau = 1$) [17]. The output of our time delay embedding function is a time series of size $[n - (m - 1) * t] \times [d * m]$. What makes our *fTDE* implementation time efficient is the fact that the for loop in *Line* 11 does not run over the length n of the time series, but iterates over the embedding dimension m which is usually much smaller.

```
1  function X = fTDE(x,m,t)
2  %FTDE time delay embedding (C) Spiegel. et al.
3  %   x   ..  time series [n times d]
4  %   m   ..  embedding
5  %   t   ..  dalay
6  %   X   ..  time series [n-(m-1)*t times d*m]
7
8      [n,d] = size(x);
9
10     X = zeros(n-(m-1)*t,d*m);
11     for i = 1:m
12         a = i+(t-1)*(i-1);
13         b = a+n-1-(m-1)*t;
14         X(:,d*(i-1)+1:d*i) = x(a:b,:);
15     end
16 end
```

6.4.2 Discretization

In most real-life time series applications we aim at analyzing temporal data with continuous values. Since our concept of pairwise proximities *PP* and stationary states *SS* mainly relies on histograms, we need to apply some kind of binning to the continuous values beforehand. In our approach we first discretize the original time series and then create a histogram for the previously discretized data.

Our implemented discretization function requires an input time series of size $n \times d$ and the specification of a similarity threshold ε, which defines the size of the bins. In earlier work [17] we proposed to perform the discretization according to the size of the ε-neighborhood in the following manner: $\tilde{x} = \lfloor x/2\varepsilon \rfloor$ (see *Line* 23). The discretization is done element-wise and effects the approximation error of the subsequent recurrence quantification analysis [17].

```
17 function x = fDiscrete(x,eps)
18 %FDISCRETE Discretize Time Series (C) Spiegel et al.
19 %   x ..    time series [n-times-d]
20 %   eps ..  similarity thresholds [1-times-d]
21
22     if eps>0
23         x = floor(x*diag(1./(2*eps)));
24     end
25 end
```

6.4.3 Pairwise Proximity

Given a (reconstructed and subsequently embedded) phase space trajectory, we can use the concept of pairwise proximities $PP^{(\mu)}$ to quantify recurring segments of certain length μ that correspond to diagonal line structures in a recurrence plot. Having quantified the number of length of recurring segments, we can compute all diagonal line based RQA measures in a straightforward manner. For example in Sect. 6.3 we have explained how to use pairwise proximities *PP* to calculate the determinism *DET* and average diagonal line length *L*.

Our implementation of the pairwise proximities function takes an input time series of size $n \times d$ and returns the number recurring d-dimensional states, which (in our case) are the result of phase space reconstruction and subsequent trajectory embedding. The pairwise proximities function requires a time series with discrete values, since we aim at finding unique states (*Line* 31). Having identified unique states, we create a histogram that captures the frequency of the unique state in the next step (*Line* 32). Finally, we calculate the sum over the squared frequencies, which is equivalent to the inner product (dot-product) of the histogram with itself (*Line* 33).

```
26 function pp = fPProximities(x)
27 %FPPROXIMITIES pairwise proximities (C) Spiegel et al.
28 %   x ..    time series [nx-times-d]
29 %   pp ..   pairwise proximities
30
31     [~,~,ix] = unique(x,'rows');
32     hx = hist(ix,min(ix)-1:max(ix)+1);
33     pp = dot(hx,hx);
34 end
```

6.4.4 Stationary States

The idea of stationary states *SS* is an extension of earlier work [17] on approximate recurrence quantification analysis, but the concept is novel in that it enables us to quantify vertical line structures in an efficient way (without creating the recurrence plot). Although our definition of stationary states *SS* (6.14) resembles our definition of pairwise proximities *PP* (6.7), there is an important difference between the two concepts. In contrast to pairwise proximities *PP*, the computation of stationary states $SS^{(\nu)}$ with length ν is performed by comparing an embedded version of the reconstructed phase space trajectory $\mathbf{x}^{(\nu)}$ with the original trajectory $\mathbf{x}^{(1)}$ (see 6.14). This is due to the fact that we aim at identifying states that are stationary over a segment of ν time points (as explained at full length by our running example in Sect. 6.3.4).

Our implementation of the stationary state function requires as input the reconstructed phase space trajectory and its embedded version, regardless in which order. In *Line* 43–44 we identify stationary states by extracting those rows from the corresponding time series matrix, where all entries are the same. For this purpose we calculate the root mean squared value for each row vector, that is `sqrt(sum(x.^2,2)/d)`, and check for which row values the remainder after division by 1 equals 0, using the modulo operator ($\texttt{mod(value,1)}^{=} = 0$). In case that the remainder equals 0 we know that the corresponding row solely contains one and the same discrete entries.

Having identified the rows that contain only same entries, we create a stationary state histogram for the reconstructed phase trajectory as well as for its embedded version (see *Line* 46–47). By multiplying both histograms using the inner product (*Line* 48) we eventually get the number of stationary states that are steady for a certain time interval, whose length is given by the embedding dimension.

```
35 function ss = fSStates(x,y)
36 %FSSTATES stationary states (C) Spiegel et al.
37 %   x ..    time series [nx-times-d1]
38 %   y ..    time series [ny-times-d2]
39 %   ss ..   stationary states
40
41     [~,d1] = size(x);
42     [~,d2] = size(y);
43     x = x(mod(sqrt(sum(x.^2,2)/d1),1)==0,1);
44     y = y(mod(sqrt(sum(y.^2,2)/d2),1)==0,1);
45
46     hx = hist(x,min([x;y])-1:max([x;y])+1);
47     hy = hist(y,min([x;y])-1:max([x;y])+1);
48     ss = dot(hx,hy);
49 end
```

6.4.5 Approximate RQA

Having explained the implementation of time delay embedding, discretization, pairwise proximities, and stationary states, we are now in the position to introduce our experimental setup. First of all we load the Potsdam time series (*Line* 54), set the predefined parameters [16] (*Line* 57), and apply the time delay embedding (*Line* 59) to reconstruct the phase space trajectory. Afterwards we discretize the reconstructed trajectory (*Line* 62), which is a prerequisite for computing the approximate RQA measures. In the next step we embed the reconstructed and discretized trajectory (*Line* 63–64) in order to quantify the pairwise proximities (*Line* 66–68) and stationary states (*Line* 70–72). Given the number of pairwise proximities and stationary states for different embedding dimensions we are able to approximate the discussed diagonal and vertical line based RQA measures (*Line* 75–78). The results and runtimes of our experiments are presented in Sect. 6.5.

```
50 function fApproxRQA
51 %FAPPROXRQA approximate RQA (C) Spiegel et al.
52
53     % load Potsdam time series
54     x = load('../Data/temp_pdm_1893-2011.txt');
55
56     % set (predefined) parameters
57     eps = 1; m = 5; tau = 3; minL = 2;
58
59     x = fTDE(x,m,tau);          % time delay embedding
60     [n,~] = size(x);            % length of trajectory
61
62     x1 = fDiscrete(x,eps);      % discretized trajectory
63     x2 = fTDE(x1,minL,1);       % trajectory embedding
64     x3 = fTDE(x1,minL+1,1);     % trajectory embedding
65
66     pp1 = fPProximities(x1);    % pairwise proximities
67     pp2 = fPProximities(x2);    % pairwise proximities
68     pp3 = fPProximities(x3);    % pairwise proximities
69
70     ss1 = pp1;                  % stationary states
71     ss2 = fSStates(x1,x2);      % stationary states
72     ss3 = fSStates(x1,x3);      % stationary states
73
74     % compute approximate RQA measures
75     RR  = pp1/(n*n);
76     DET = (minL*pp2 - (minL-1)*pp3) / (pp1 + 10^-10);
77     L   = (minL*pp2 - (minL-1)*pp3) / (pp2 - pp3);
78     LAM = (minL*ss2 - (minL-1)*ss3) / (ss1 + 10^-10);
79 end
```

6.5 Empirical Results

The goal of our empirical evaluation is twofold: (i) we assess the runtime of original and approximate RQA measures for relatively long time series (with about a million data points); and (ii) investigate the correlation between original and approximate RQA measures for the purpose of finding transitions in time series streams using the sliding window technique. Both experiments are performed on the same real-life data set described in Sect. 6.5.1.

6.5.1 Data

For illustrating the approximation approach and to evaluate it we use the measured time series of hourly air temperature in Potsdam [16], which covers the period from 1893 until 2014 and contains 1,069,416 data points. This time series is one of the longest, non-interrupted, hourly climate records in the world. The Potsdam time series is divided into two intervals (1893–1974 and 1975–2014), because the warming trend of the annual mean temperature shows an abrupt change in 1975 [16]. However, recurrence quantification analysis has shown that, in contrast to longer time-scales, the short-term dynamics, and, thus, the short-term weather predictability, has not (yet) changed due to climate change [16]. Moreover, between 1975 and 1976 the measurement protocol has changed from manual to electronic recording. Such changes could be systematically influencing the measurements and should be visible by recurrence quantification. We, therefore, apply a windowing approach in order to investigate a potential shift in the recurrence properties after 1975. This approach can be further used to detect regime transitions in the local weather regime of Potsdam, but this is focus of a separate future study.

6.5.2 Experimental Protocol

We conducted experiments on (i) the runtime and (ii) the accuracy between original and approximate RQA measures.

Our experiments on (i) runtime were performed on different intervals of the Potsdam time series, namely 1893–1974, 1975–2014, and 1893–2014, as well as on yearly intervals (sliding window analysis). Before analysis the time series has been normalized to zero mean and standard deviation one. The original RQA measures have been calculated using three different implementations, basing on C++ for (i) single thread and (ii) multi thread CPU, and on Python (pyRQA) for (iii) GPU computing [16, 25]. The approximate RQA measures have been calculated by our own MATLAB implementation as described in Sect. 6.4.5, while the runtime has been determined for each measure individually (using one CPU). The runtime comparison

of both original and approximate RQA measures for all three time intervals can be found in Sect. 6.5.3.

Our evaluation on (ii) the accuracy between original and approximate RQA measures is presented in Sect. 6.5.4. For these experiments we considered the Potsdam time series in its entire length (1893–2014) and slide a '1-year' window with '1-year' step size from the beginning to the end. This approach is often referred to as sliding window technique and is commonly used to detect transitions in time series [20]. For each window we compute both original and approximate RQA measures (and using different recurrence thresholds to demonstrate the effect of the threshold), which gives us the temporal changes of the RQA measures under study. Given these temporal changes, we are able to compare the variations of the original and approximate RQA measures. For comparison we are using the Pearson correlation coefficient, the root mean square error, and the relative root mean square error. A high correlation, i.e., both measures vary in a similar way would confirm our hypothesis that the loss in accuracy is still a reasonable trade-off with the gain of speed and that the proposed approximations can be used to find transitions in time series streams.

6.5.3 Results on Runtime

Table 6.2 shows the runtimes of various RQA implementations. It is important to mention that the runtimes are merely an indicator for the performance of the examined implementations, since the experiments were performed under different conditions (using various hardware setups and programming languages).

The runtime experiments were conducted on a computer cluster (PIK HLRS2015—Lenovo/IBM NeXtScale nx360M5), consisting of compute nodes with Intel Xeon E5-2667v3 2×8 core CPUs at up to 3.2 GHz and 64 GB main memory. It furthermore includes NVIDIA Tesla K40 nodes that provide GPU processors running at up to 745 GHz, 2880 stream processors and each supplied with 12 GB of memory. The cluster runs on a 64-bit version of Suse Linux Enterprise Server 11 SP3 with version 7.0.28 of CUDA. The CUDA platform was utilized for the experiment that employ the OpenCL/Python implementation. One CPU node of the cluster was

Table 6.2 Runtime (in sec) for RQA calculation for Potsdam temperature time series

Data set	1893–1974	1975–2014	1893–2014
Data points	718,776	350,640	1,069,416
Single thread (CPU)	9,978.00	2,373.00	22,067.00
16 thread OpenMP (CPU)	782.00	188.00	1,743.00
OpenCL (1x GPU)	439.00	104.00	995.00
Approximation (CPU)	1.83	0.79	2.88

Table 6.3 RQA results for Potsdam time series for three different epochs calculated using maximum-norm, embedding dimension 5, embedding delay 3, threshold 0.75, and Theiler window 0

Data set	1893–1974	1975–2014	1893–2014
Data points	718,776	350,640	1,069,416
RR	0.15	0.15	0.15
RR approx.	0.12	0.12	0.12
DET	0.92	0.92	0.92
DET approx.	0.89	0.90	0,89
L	7.6	7.7	7.7
L approx.	6.9	7.2	7.0
LAM	0.96	0.95	0.96
LAM approx.	0.91	0.91	0.91

exploited for the single-thread and 16 core multi-thread (OpenMP) implementation, using C++ programming language. The approximation experiment was performed on the same hardware using MATLAB 2011b.

Although the runtime experiments were conducted with varying hardware setups and programming languages, the results give some indication of the speed-up factor achieved by our proposed approximation techniques. However, the approximation error is not to be neglected. Table 6.3 demonstrates the results for the original and approximate RQA measures, which also confirm the previous climatological findings and interpretations given in [16].

Although the results presented in Table 6.3 slightly differ for the original and approximate RQA measures, it is more important that the variation (tendency) of these measures is similar. This is considered in more detail in the next section.

6.5.4 Results on Correlation

In several applications, such as RQA based transition detection, the absolute values of the RQA measures are less important than the tendency of their variation with time. Here we compare the ability of the approximation approach to uncover variations that are similar to ones found by the exact measures. We apply both (exact and approximate) RQA measures to the Potsdam temperature series using a sliding windowing technique (non-overlapping windows with length of 1 year), which is the standard approach for detecting transitions or regime changes.

In Fig. 6.4 we find a similar variation between the RQA measures calculated using the exact as well as the approximation approaches. Figure 6.4a was derived with the same recurrence threshold $\varepsilon = 0.75$ that was taken for the analysis of the whole Potsdam time series (see Table 6.3). This threshold is considered as baseline here, meaning that the exact RQA measures show desired characteristics. Visually, the most

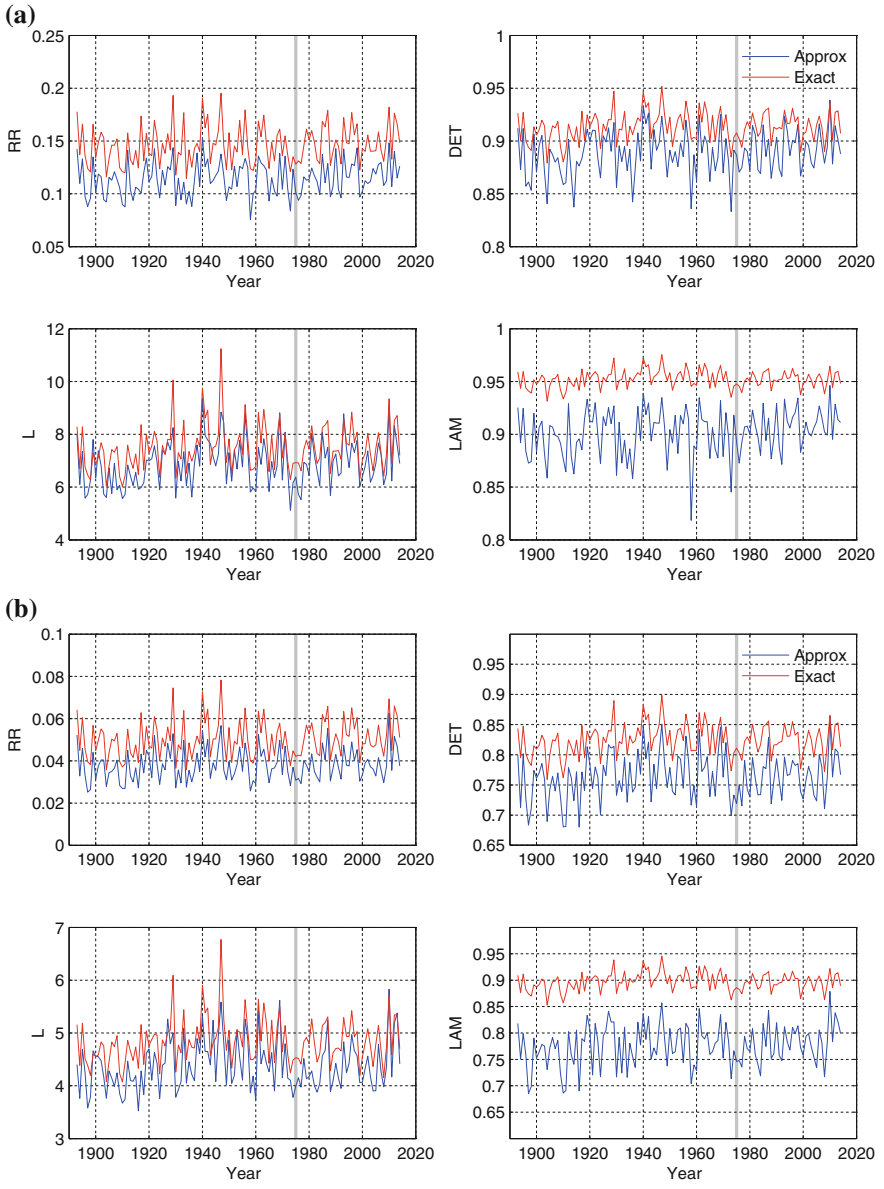

Fig. 6.4 Windowed RQA results for Potsdam time series calculated using a non-overlapping sliding window of length 1 year, maximum-norm, embedding dimension 5, embedding delay 3, Theiler window 0 and (**a**) threshold 0.75, (**b**) threshold 0.5. **a** Baseline similarity threshold: $\varepsilon = 0.75$, **b** Lower Similarity Threshold: $\varepsilon = 0.5$

Table 6.4 Correlation coefficient, root mean square and relative error between exact and approximate RQA measures for Potsdam temperature series as shown in Fig. 6.4

$\varepsilon = 0.75$	Correlation	RMSE	rel. RMSE (%)	$\varepsilon = 0.5$	Correlation	RMSE	rel. RMSE (%)
RR	0.79	0.034	17	RR	0.94	0.013	16
DET	0.81	0.028	3	DET	0.88	0.064	7
L	0.84	0.79	7	L	0.82	0.526	8
LAM	0.67	0.054	6	LAM	0.78	0.124	13

similar variation (and amplitude) is for the measure *L*, followed by *DET*, whereas the amplitude of the *LAM* measure has the largest deviation. This is quantitatively confirmed by the correlation coefficient and the error measures (Table 6.4). The highest correlations, thus the best coincidence of the variation, have the measures *L* and *DET*, whereas *LAM* has the lowest correlation. In contrast, the relative error is smallest for *DET* but largest for *RR*.

The values of the correlation and the errors can be controlled by changing the threshold ε (Fig. 6.5). For *L* and *LAM*, we find the best correlation for recurrence thresholds between 0.4 and 0.6 in units of the signal's standard deviation. The root mean square error decreases for *L* where it increases for *LAM* with decreasing threshold. The threshold that leads to the overall best correlation is $\varepsilon = 0.5$ (Fig. 6.4b), the corresponding windowed analysis is illustrated in Fig. 6.4b. Hence, if we choose this threshold for the sliding window analysis, then the mean correlation between the exact and the approximate RQA measures is greater than 0.85, which indicates a very strong linear relationship. But how does the choice of a lower threshold influence the sliding window analysis? Visually, the corresponding exact measures in Fig. 6.4 vary in a similar fashion, they only obey a different scale. This observation is confirmed by Table 6.5, all corresponding exact measures have a correlation of about 0.99 for the baseline threshold $\varepsilon = 0.75$ compared with the lower threshold $\varepsilon = 0.5$. In summary, the choice of a lower threshold can keep nearly the entire information on the variation of the exact RQA measures, and at the same time can significantly improve the approximate RQA in that the correlation between the exact and the approximate RQA measures increases.

From the climatological point of view, both approaches reveal some variation in the dynamics represented by the temperature time series. A general shift after the time point of the change in recording procedure 1975 is not visible. However, besides several short periods of decrease in the measures, around 1975 a clear drop can be identified and might be more an indication of a sudden change in the general climate regime by passing a tipping point [26] than of a change in the recording procedure. A future study will investigate these variations more systematically and should also consider significance tests [10].

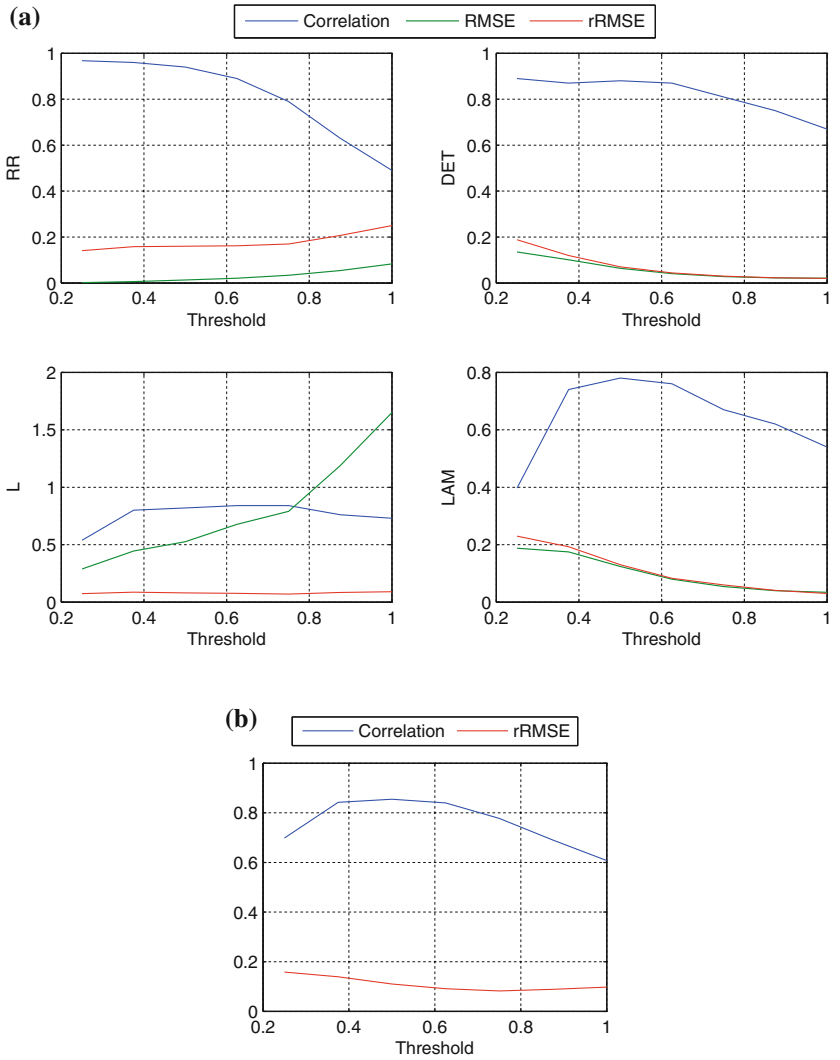

Fig. 6.5 Correlation coefficient, root mean square error (RMSE), and relative RMSE between exact and approximate RQA measures with varying thresholds ε, for our sliding window analysis of the Potsdam temperature series as shown in Fig. 6.4. **a** Individual RQA measures, **b** Mean over all RQA measures from (**a**)

6.6 Conclusion and Future Work

This chapter extends our theoretical work on approximate recurrence quantification analysis (aRQA) [17] and includes some practical considerations that occur when analyzing real-life data such as the Potsdam temperature time series [16]. We have

Table 6.5 Correlation between **exact** RQA measures for threshold $\varepsilon = 0.75$ and **exact** RQA measures for $\varepsilon = 0.5$ for Potsdam temperature series as shown in Fig. 6.4

	RR	DET	L	LAM
Correlation	0.9953	0.9915	0.9877	0.9942

not only discussed the formulation of diagonal line based measures by means of pairwise proximities (*PP*) [17], but also introduced our novel idea of stationary states (*SS*) that enables us to reformulate vertical line based RQA measures. In addition to our new formulation of the original RQA measures, we furthermore presented an efficient implementation that allows fast computation of the approximate RQA measures based on histograms.

Our experiments on relatively long time series (with about a million measurements) demonstrated that the proposed approximation is not only up to four orders of magnitude faster than single thread (exact) computations, but also gives results that are very close to the original measures. Furthermore, we were able to show that our approximate RQA measures strongly correlate with the corresponding exact RQA measures (when applying the sliding window technique) and, therefore, can be used for an efficient transition detection. The presented empirical results are also in agreement with our theoretical analysis [17] in that the error of the approximation is decided by the discretization or strictly speaking depends on the similarity threshold and the distribution of the data under study.

In future work we are going to investigate the discretization more deeply and develop time series representations that enable us to bound the approximation error. Moreover, we will transfer our idea of pairwise proximities (*PP*) and stationary states (*SS*) to cross recurrence plots (CRPs) and corresponding measures.

References

1. N. Marwan, A historical review of recurrence plots. Eur. Phys. J. Spec. Topics **164**(1), 3–12 (2008)
2. C.L. Webber Jr., N. Marwan, A. Facchini, A. Giuliani, Simpler methods do it better: success of recurrence quantification analysis as a general purpose data analysis tool. Phys. Lett. A **373**, 3753–3756 (2009)
3. S. Spiegel, J.B. Jain, S. Albayrak, A Recurrence Plot-Based Distance Measure, vol. 103 (Springer, Cham, 2014), pp. 1–15
4. S. Spiegel, D. Schultz, S. Albayrak, BestTime: Finding representatives in time series datasets. Lecture Notes in Computer Science: Artificial Intelligence, vol. 8726, pp. 477–480 (2014)
5. F. Hasselman, Classifying acoustic signals into phoneme categories: average and dyslexic readers make use of complex dynamical patterns and multifractal scaling properties of the speech signal. PeerJ **3**, e837 (2015)
6. M.A.F. Harrison, M.G. Frei, I. Osorio, Detection of seizure rhythmicity by recurrences. Chaos **18**(3), 033124 (2008)

7. N. Marwan, N. Wessel, U. Meyerfeldt, A. Schirdewan, J. Kurths, Recurrence plot based measures of complexity and its application to heart rate variability data. Phys. Rev. E **66**(2), 026702 (2002)
8. A. Giuliani, M. Tomasi, Recurrence quantification analysis reveals interaction partners in paramyxoviridae envelope glycoproteins. Proteins: Struct. Func. Genetics **46**(2), 171–176 (2002)
9. S. Spiegel, *Discovery of Driving Behavior Patterns* (Springer, Cham, 2015), pp. 315–343
10. N. Marwan, S. Schinkel, J. Kurths, Recurrence plots 25 years later: gaining confidence in dynamical transitions. Europhys. Lett. **101**, 20007 (2013)
11. J.F. Donges, R.V. Donner, M.H. Trauth, N. Marwan, H.J. Schellnhuber, J. Kurths, Nonlinear detection of paleoclimate-variability transitions possibly related to human evolution. Proc. Natl. Acad. Sci. **108**(51), 20422–20427 (2011)
12. G. Litak, A.K. Sen, A. Syta, Intermittent and chaotic vibrations in a regenerative cutting process. Chaos Solitons Fractals **41**(4), 2115–2122 (2009)
13. M.C. Romano, M. Thiel, J. Kurths, I.Z. Kiss, J. Hudson, Detection of synchronization for non-phase-coherent and non-stationary data. Europhys. Lett. **71**(3), 466–472 (2005)
14. Y. Hirata, K. Aihara, Identifying hidden common causes from bivariate time series: a method using recurrence plots. Phys. Rev. E **81**(1), 016203 (2010)
15. J.H. Feldhoff, R.V. Donner, J.F. Donges, N. Marwan, J. Kurths, Geometric signature of complex synchronisation scenarios. Europhys. Lett. **102**(3), 30007 (2013)
16. T. Rawald, M. Sips, N. Marwan, D. Dransch, *Fast Computation of Recurrences in Long Time Series*, vol. 103 (Springer, Cham, 2014), pp. 17–29
17. D. Schultz, S. Spiegel, N. Marwan, S. Albayrak, Approximation of diagonal line based measures in recurrence quantification analysis. Phys. Lett. A **379**(14–15), 997–1011 (2015)
18. J.-P. Eckmann, S. Oliffson, Kamphorst, D. Ruelle, Recurrence plots of dynamical systems. Europhys. Lett. **4**(9), 973–977 (1987)
19. N.H. Packard, J.P. Crutchfield, J.D. Farmer, R.S. Shaw, Geometry from a time series. Phys. Rev. Lett. **45**(9), 712–716 (1980)
20. N. Marwan, M.C. Romano, M. Thiel, J. Kurths, Recurrence plots for the analysis of complex systems. Phys. Rep. **438**(5–6), 237–329 (2007)
21. J.P. Zbilut, C.L. Webber Jr., Embeddings and delays as derived from quantification of recurrence plots. Phys. Lett. A **171**(3–4), 199–203 (1992)
22. C.L. Webber Jr., J.P. Zbilut, Dynamical assessment of physiological systems and states using recurrence plot strategies. J. Appl. Physiol. **76**(2), 965–973 (1994)
23. M. Thiel, M.C. Romano, P.L. Read, J. Kurths, Estimation of dynamical invariants without embedding by recurrence plots. Chaos **14**(2), 234–243 (2004)
24. N. Marwan, J.F. Donges, Y. Zou, R.V. Donner, J. Kurths, Complex network approach for recurrence analysis of time series. Phys. Lett. A **373**(46), 4246–4254 (2009)
25. T. Rawald, pyRQA (2015)
26. M. Scheffer, J. Bascompte, W.A. Brock, V. Brovkin, S.R. Carpenter, V. Dakos, H. Held, E.H. van Nes, M. Rietkerk, G. Sugihara, Early-warning signals for critical transitions. Nature **461**(7260), 53–59 (2009)