

EPJ manuscript No.
 (will be inserted by the editor)

Energy-efficient recurrence quantification analysis

Norbert Marwan^{1,2,3,a}

- ¹ Potsdam Institute for Climate Impact Research (PIK), Member of the Leibniz Association, Telegrafenberg A31, 14473 Potsdam, Germany
- ² University of Potsdam, Institute of Geoscience, Karl-Liebknecht-Straße 32, 14476 Potsdam, Germany
- ³ University of Potsdam, Institute of Physics and Astronomy, Karl-Liebknecht-Straße 32, 14476 Potsdam, Germany

Abstract. Recurrence quantification analysis (RQA) is a widely used tool for studying complex dynamical systems, but its standard implementation requires computationally expensive calculations of recurrence plots (RPs) and line length histograms. This study introduces strategies to compute RQA measures directly from time series or phase space vectors, avoiding the need to construct RPs. The calculations can be further accelerated and optimised by applying a random sampling procedure, in which only a subset of line structures is evaluated. These modifications result in shorter run times, less memory use and access, and lower overall energy consumption during analysis while maintaining accuracy. This makes them especially appealing for large-scale data analysis and machine learning applications. The ideas are not limited to diagonal line measures, but can likewise be applied to vertical line-based measures and to recurrence network measures. By lowering computational costs, the proposed strategies contribute to energy saving and sustainable data analysis, and broaden the applicability of recurrence-based methods in modern research contexts.

1 Introduction

In the light of sustainable energy production and climate change, energy efficiency in modelling and data analysis is an important and growing topic [37, 24]. As the demand for data processing grows, it is crucial to focus on reducing related energy consumption [23, 12, 2]. This problem is increasingly attracting attention across various fields, but it remains in its early stages. Among the many goals and actions that help address energy-efficient computations [2] and are especially feasible in daily work and short-term contexts, the key measures include educating about this topic and enhancing implementations [1].

In the context of sustainable data analysis, energy efficiency is becoming increasingly critical. This is particularly relevant for computationally intensive methods such

^a e-mail: marwan@pik-potsdam.de

as recurrence quantification analysis (RQA), a powerful tool in nonlinear data analysis [7, 18, 35, 36]. This framework has shown high potential in investigating diverse research questions across many disciplines [14] and its integration with machine learning (ML) has recently sparked considerable interest [15].

However, this potential comes at a cost: RQA stands out as a particularly energy-intensive method in the ML toolkit. Unlike linear techniques, RQA relies on pairwise comparisons in high-dimensional data, resulting in computational complexity that scales quadratically with data size. In ML workflows, this challenge is amplified: RQA is often embedded in iterative processes – such as feature extraction, hyperparameter tuning, or real-time inference – where its energy demands compound with those of the broader pipeline. For example, extracting RQA-based features from thousands of time series segments or combining RQA with neural networks in hybrid models can result in high energy consumption costs [30, 25]. Moreover, the nonlinear nature of RQA resists simple optimisations, making efficient implementations not just a performance concern, but a necessity for sustainable ML. Addressing this gap is critical to unlocking RQA’s potential in data-intensive fields like climate science, biomedicine, and IoT – where energy-efficient algorithms are crucial for scalable, real-world applications [30]. Beyond reducing energy consumption, optimised algorithms generally provide faster computations, enabling real-time applications and large-scale deployments [27, 37, 25].

The standard RQA algorithm begins with calculating the RP, which has a computational complexity of $\mathcal{O}(N^2)$ [18]. The lines formed by consecutive points in the RP are then detected, and their lengths are measured. As we consider diagonal and vertical lines, the computational complexity for each of these steps is also of order $\mathcal{O}(N^2)$ (also in the case of symmetric RPs, where we would consider only one triangle of the RP). This complexity requires many calculation steps and, therefore, also increases calculation times. In the last decade, various approaches have been proposed to speed up RQA calculations. Classical approaches utilise parallelisation and use multiple graphics processing unit (GPU) devices [27, 28]. The latter distributes the calculations of RPs and line lengths on different GPU devices using the divide and recombine paradigm [26]. Although the acceleration of the computations is remarkable, the complexity of the calculations has not changed. An approach that can indeed reduce the computational complexity extremely is a numerical and geometrical approximative ansatz, which can heavily accelerate the calculations by several magnitudes, but with the costs of increasing uncertainties [31, 33]. More recently, the introduction of microstate recurrence analysis [5] has offered another (random sampling-based) approximative but fast approach, with much less uncertainty in the RQA measures [6]. Here, the line length distributions, required for the RQA measures, are estimated from randomly sampled sub-matrices from the RP.

In the following, we will consider the original approach of RQA calculations, without parallelisation (although it would be possible) and without numerical/ geometrical approximation, but using the diagonal line structures of interest and developing some optimisation strategies. This optimisation follows the action numbers M2 (“Reduce and compress data having the anticipated scientific value of the retained information and the resource requirements in mind”) and M7 (“Design software for optimized energy consumption and provide tools to measure it”) [2]. For the sake of simplicity, only the diagonal lines are considered here; the same approach can be applied for calculating the measures based on the vertical lines or even more complex patterns, such as triangle motifs (required for recurrence network measures [19]). The first optimisation will be very simple and has surely already been applied in several implementations. The second one is inspired by the microstates approach by da Cruz et al. [6]. These approaches are compared to a simple, straightforward standard implementation without any optimisations, as someone lacking knowledge of algorithmic and language-specific

optimisations (such as those available in Julia) would perform it. The algorithms are implemented using the Julia language because it enables easy measurement of the performance of the implementations.

2 Recurrence Plots and Recurrence Quantification Analysis

A recurrence plot (RP) is a square matrix \mathbf{R} indicating the similarity of two states $\vec{x}(i)$ and $\vec{x}(j)$ at different points in time i and j :

$$R(i, j) = \Theta(\varepsilon - \|\vec{x}(i) - \vec{x}(j)\|), \quad (1)$$

with states $\vec{x} \in \mathbb{R}^d$ (d the dimension of the phase space), $i, j = 1, \dots, N$ the time indices, Θ the Heaviside function, and ε the recurrence threshold [18]. The RP exhibits typical large-scale appearances, depending on the system's dynamics [7, 18]. It further contains small-scale line structures that represent temporally close evolution of the phase space trajectory pairs (diagonal lines) or trapped states (vertical lines). The distributions of these line lengths provide insights into the dynamics of the system and are used to define measures of complexity within recurrence quantification analysis (RQA).

A typical RQA measure is the determinism measure (DET), which quantifies the ratio of recurrence points ($R(i, j) = 1$) which form diagonal lines and the total number of recurrence points, and is based on the histogram $P(\ell)$ of the occurrences of lines of length ℓ ,

$$DET = \frac{\sum_{\ell=2}^N \ell P(\ell)}{\sum_{\ell=1}^N \ell P(\ell)}. \quad (2)$$

This measure is related to predictability, with large values for highly predictable dynamics and low values for non-predictable dynamics. It can be used, e.g., to identify regime changes or classify different regimes or systems [4, 9, 20, 22].

For a more general overview of the method and its application potential, the reader is referred to [14, 16].

3 Optimisation Strategies

The following implementations have been tested on a single core of the high-performance cluster "Eunice Newton Foote" at the Potsdam Institute of Climate Impact Research (AMD EPYC 9554 processors with scalar frequencies of up to 3.75 GHz and 6 GByte DDR5 memory per core). The performance of an implementation was tested using the Julia macros `@time` and `@timed`. The calculations were performed 500 times to ensure reliable statistical estimates of the performance measurements.

As test data, the three components of the Rössler system (using standard parameters $a = 1.2$, $b = 0.2$, and $c = 5.7$) are used, with length 25,000 and sampling time $\Delta t = 0.2$ [29]. The Rössler system was integrated using a Tsitouras 5/4 Runge–Kutta–Solver as implemented by the DifferentialEquations.jl package in Julia.

Recurrence, Eq. (1), is calculated for all three components of the Rössler system, using a threshold ε of 10% of the range of the values of all three components, i.e., $\varepsilon = 0.1(\max(\vec{x}) - \min(\vec{x}))$.

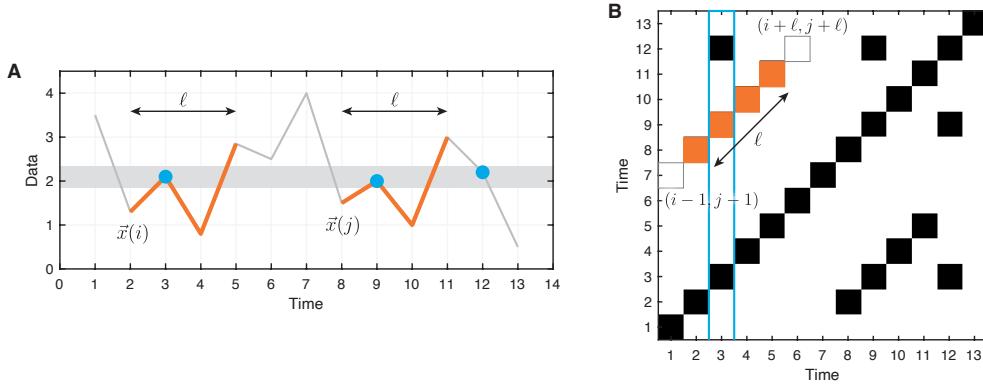


Fig. 1. (A) Time series sequence and (B) corresponding recurrence plot. The sequence at time points 2 to 5 (orange) repeats four times within a small error ε (here $\varepsilon = 0.25$) during the interval 8 to 11, forming a diagonal line with points $(i = 2, j = 8)$, $(i + 1 = 3, j + 1 = 9)$, $(i + 2 = 4, j + 2 = 10)$, and $(i + 3 = 5, j + 3 = 11)$ in the RP (orange points). The line is preceded by condition Eq. (4), $\|\vec{x}(i - 1) - \vec{x}(j - 1)\| > \varepsilon$, and followed by condition Eq. (5), $\|\vec{x}(i + \ell) - \vec{x}(j + \ell)\| > \varepsilon$, (grey boxes in panel (B)), ensuring the lines start and end points. The length of the diagonal line ($\ell = 4$) can be measured in the RP or directly from the time series. The state at time 3 recurs (within the ε uncertainty, grey horizontal bar in panel (A)) at time 9 and 12 (blue dots), indicated by the points in the column 3 (marked by blue box).

3.1 Histogram Estimation without RP (RQA_woRP)

A line structure in an RP is a sequence of pairs of time indices. A diagonal line in the RP of length ℓ with the coordinates $\{(i, j), (i + 1, j + 1), \dots, (i + \ell - 1, j + \ell - 1)\}$ represents that the sequences of states $\{\vec{x}(i), \vec{x}(i + 1), \dots, \vec{x}(i + \ell - 1)\}$ and $\{\vec{x}(j), \vec{x}(j + 1), \dots, \vec{x}(j + \ell - 1)\}$ are similar within the recurrence uncertainty defined by the threshold ε (Fig. 1).

It is clear that we can find line lengths in the RP without the RP. We can test the condition

$$\|\vec{x}(i) - \vec{x}(j)\| \leq \varepsilon \quad (3)$$

directly at the series of the phase space vectors (or the time series) for increasing indices i and j . Considering the condition

$$\|\vec{x}(i - 1) - \vec{x}(j - 1)\| > \varepsilon \quad (4)$$

for the start of a line and

$$\|\vec{x}(i + \ell) - \vec{x}(j + \ell)\| > \varepsilon \quad (5)$$

as the end of a line, we can get the distribution of line lengths $P(\ell)$. This is a very simple and straightforward calculation of $P(\ell)$ and allows us to get the RQA measures without previous calculating the RP beforehand. The resulting histogram of the diagonal lines will be exactly the same as the one obtained from the RP. It will not reduce the numerical complexity but the number of calculation steps, and should, therefore, result in more efficient and faster RQA calculations when the RP is not required. It also reduces the number of memory allocations, as we do not need to store and calculate the matrix \mathbf{R} .

This kind of optimising the histogram calculation without calculating the RP beforehand is surely not novel. It has been used by the author before and surely

might also have been implemented by others. For example, such implementation is available by [21] and was used in the run-time comparisons in the studies by Rawald et al. [26, 28] and Spiegel et al. [33].

The comparison with the standard implementation using RP-based line histogram calculation shows a speed-up by a factor of more than 2 (Tab. 1). The number of memory allocations dropped from 7 to 3. The just-in-time compiler (JIT) in Julia allows some internal optimisation steps, e.g., by applying the macro `@inbounds` to the for-loops [32]. Using it can further accelerate the implementation. Here, it finally speeds up the implementation by a factor of more than 4. In the following analysis, we will use the `@inbounds` optimised version.

Table 1. General performance values of the different implementations when calculating the line length histograms required for RQA and using the Rössler system with all three components and of length $N = 25,000$; number of samplings for the sampling version and microstates $M = 4N = 100,000$, except for (RQA_Samp²);

¹ the calculation directly from the time series (RQA_woRP) has been performed additionally with the Julia instruction `@inbounds` which can accelerate the code further

²calculation of RQA_Samp using a smaller number of samplings $M = 0.2N = 5,000$.

Implementation	Execution time	Speedup	Allocations	Energy
RQA from RP (RQA_RP)	2.06 s	1	7	28.6 mWh
RQA without RP (RQA_woRP)	0.89 s	2.3	3	12.4 mWh
RQA without RP ¹ (RQA_woRP ¹)	0.48 s	4.3	3	6.7 mWh
RQA from sampling (RQA_Samp)	0.26 s	7.9	3	3.6 mWh
RQA from sampling ² (RQA_Samp ²)	0.013 s	158	3	0.18 mWh
RQA from microstates	0.018 s	114	29	0.25 mWh

3.2 Sampling-based Histogram Estimation (RQA_Samp)

Based on the histogram estimation directly from the data, and without RP, we can further reduce the calculation costs by applying a random sampling scheme. Instead of sequentially testing every index j from 1 to N for every i , we can randomly draw indices i and j and test whether this pair fulfills the conditions given by Eqs. (3) and (4), i.e., correspond to the beginning of a diagonal line. If not, we randomly draw a new pair i and j . If yes, we can measure the line length as is done in RQA_woRP, but stop after the end of a line and randomly sample a new line. By this sampling schema, we get only a subset of lines in the histogram $P(\ell)$, depending on the number M of samplings. Using a smaller number of M , e.g., $M = 4N$, we can significantly reduce the number of calculations to estimate $P(\ell)$. The resulting histogram $P(\ell)$ is a direct approximation of the histogram as it would be obtained using the RP_woRP algorithm described in Subsect. 3.1. Importantly, this approximation concerns the same underlying distribution, and the standard RQA measures can be computed from $P(\ell)$ without any modification. Even for small values of M , this distributional approximation is already quite accurate.

In general, this sampling approach is, somehow, similar to estimating RQA measures using randomly sampled microstates as explained in [6], but with the difference that RQA measures (such as DET) can be directly estimated from the histogram $P(\ell)$, i.e., they are not reconstructed from a different (the microstated) distribution, and, thus, will result in higher accuracy when using the same sampling size M .

Comparing to the standard estimation algorithm (RQA_RP) and the direct one (RP_woRP), we find a further speed-up depending on the number M (Tab. 1). The memory allocations are the same as for the direct histogram estimation without RPs. A large number of samples $M = 4N$, the calculation time is halved; further reducing M to 20% of the data length speeds up the RQA estimation by almost two orders of magnitude.

Therefore, it might be desired to have a small M , but it obviously causes worse estimates of the line length measures. Considering the deviation $DET_{\text{sampled}} - DET_{\text{true}}$ shows that increasing M reduces the error of the DET estimation when obtained with this sampling schema (Fig. 2). For the considered data with $N = 25,000$, a sampling size of $M = 500$ would already result in excellent estimations of DET with deviations of $< 10^{-4}$, using $M = 5,000$ the deviations drop below 10^{-5} .

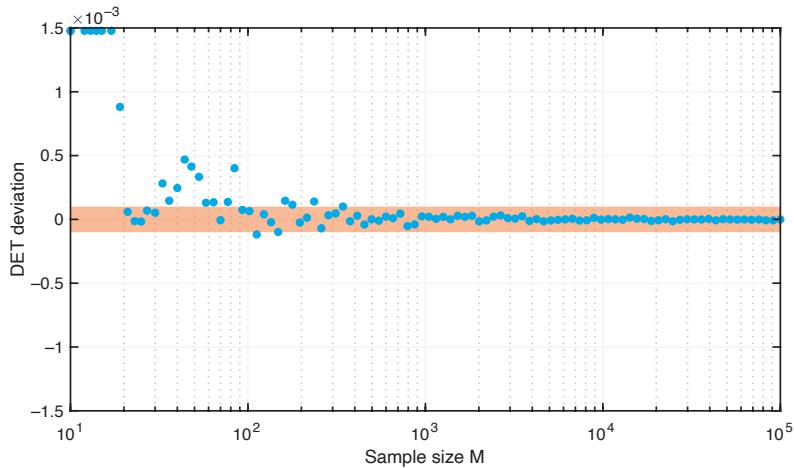


Fig. 2. Deviation values $DET_{\text{sampled}} - DET_{\text{true}}$ derived from sampled line length histogram for the Rössler system with 25,000 data points and for sampling size ranging from $M = 10$ to 100,000. The orange region indicates the deviation range of $\pm 10^{-4}$. A random sampling of only 500 line segments already results in minor deviations smaller than 10^{-4} .

3.3 Efficiency by computation time

One aspect of efficient implementations is the reduced number of calculation steps, saving energy and also computation time. The first approach of calculating the RQA directly from the data, without the RP (RP_woRP) shows already a significant reduction of computation time of more than 50%, because calculating the RP is not required (Fig. 3). Further optimisation using the sampling schema (RP_Samp) reduces the computation time remarkably. This reduction depends on the sampling size M . However, it is not necessary to scale the sampling size M with the squared size N of the data. We find high accuracy already when we scale M linearly with the size N , e.g., by $M = 4N$ or $M = 0.2N$, as it is used here in the comparison. This results in a lower scaling exponent of the computation time as data length increases (Fig. 3). This means, the longer the data, the more efficient this method becomes.

We also note that for small data ($N < 10,000$), the sampling schema is not necessarily faster than RQA_woRP. Additionally, uncertainties in the RQA results

increase for shorter time series when using sampling-based approaches, affecting both RQA_Samp and the microstates method (see appendix, Fig. 5), suggesting that RQA_woRP is preferable in such cases.

Using the same number M in the recurrence microstates approach, the computation time is, in general, smaller (although it might not be directly comparable, because the microstates calculation is based on compiled C code highly optimised for Julia [34]), but with the cost of less accurate results (see appendix, Figs. 4 and 5).

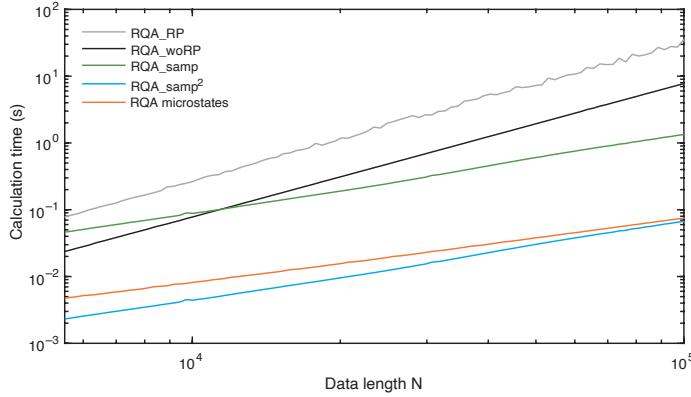


Fig. 3. Computation times for the RQA implementations using RP (RQA_RP), without RP (RQA_woRP), with a sampling scheme (RQA_samp), and using a recurrence microstates approximation for increasing data length N (Rössler system, same parameters as in Fig. 2). The sampling size M for RP_Samp and microstates is $M = 4N = 100,000$, and for RP_Samp² it is $M = 0.2N = 5,000$.

The reduced computation time directly translates into lower energy consumption. Estimating a specific power draw per CPU core for RQA calculations is not easy, as it depends on the workloads, architecture, computer system (e.g., HPC, laptop), underlying operating systems, and other factors [11, 12, 23]. However, assuming an average power draw of ≈ 50 W per CPU core as a realistic baseline, replacing RQA_RP (2.06 s for data with length $N = 25,000$) with RQA_Samp (0.26 s) reduces the energy usage from approximately 0.029 Wh to 0.0036 Wh per run, corresponding to an energy saving of about 0.025 Wh, or almost 90%. Using the even faster RQA_Samp² (0.013 s) further lowers the consumption to 0.00018 Wh, i.e., a saving of roughly 0.028 Wh per run. While these savings are small for a single computation, they accumulate substantially when processing large data sets, performing repeated analyses, or running parallel workflows on HPC systems. For example, 100,000 runs of a typical RQA of data with length $N = 25,000$ accumulate to an energy saving of 2.8 kWh, equivalent to ≈ 1.0 kg of CO₂ emissions (assuming 363 g CO₂ per kWh based on the German electricity mix in 2024 [10]).

4 Discussion and conclusion

Efficiency gains in data analysis can be achieved at multiple levels. Low-level optimisations, such as compiler hints (e.g., `@inbounds` in Julia's JIT compiler) or vectorisation directives, can reduce execution time without changing the underlying algorithm. However, more substantial improvements are typically obtained through algorithmic

modifications, such as avoiding intermediate data structures and calculation steps, or applying sampling schemes, like the RQA_woRP and RQA_Samp approaches.

These discussed approaches demonstrate that recurrence-based complexity measures can be obtained in a direct and computationally efficient way without the need for pre-computed RP. When this idea is combined with a sampling strategy, in which only a fraction of the possible structures are analysed, the computational costs can be reduced even further. This leads not only to faster execution, but also to lower memory requirements and energy consumption, which are relevant considerations in modern large-scale data analysis pipelines.

Such efficiency gains are particularly important in applications where RQA is employed in combination with machine learning techniques, as efficient computation allows more extensive training and validation procedures and facilitates the systematic exploration of the model architectures. They are equally relevant in the context of large-scale (big) data analysis, where reduced memory and runtime requirements make it possible to apply RQA to very long time series, large ensembles of signals, or even continuous data streams. By strengthening both domains, the proposed approach enhances the accessibility and practical utility of RQA in modern data-driven research. Furthermore, the reduced computational overhead may facilitate the integration of RQA methods into real-time or embedded systems, where resources are limited.

The choice of which approach to use depends on the research questions, data characteristics, and computational constraints. For small datasets ($N < 10,000$) and when RPs are not needed, the RQA_woRP approach is the best choice, ensuring high accuracy and reasonable computation time. RQA_Samp becomes advantageous for longer time series, due to its linear sampling complexity ($M \propto N$), which scales more favorably than the $\mathcal{O}(N^2)$ complexity of RQA_woRP, i.e., the computational savings grow with data length.

The results presented here demonstrate typical behaviour for a continuous chaotic system (Rössler attractor). Different dynamical regimes may affect the performance of these approaches differently. For instance, discrete maps like the logistic map generate RPs with shorter, more heterogeneous line structures. This leads to faster sampling (as valid line starts are found more frequently) but potentially larger uncertainties, particularly when M is insufficient to adequately represent the line length distribution (Fig. 6A, C). In contrast, periodic systems have more homogeneous RPs with very long, non-interrupted, but less frequent diagonal lines. This reduces the probability of finding lines through random sampling, increasing computation time (can be even worse than the standard RQA_RP approach). However, the limited diversity of line lengths ensures accurate results even for small M (Fig. 6B, D).

Sampling approaches are most efficient for homogeneously structured RPs characteristic of stationary dynamics. Non-stationary systems with heterogeneous RPs are expected to lead to larger uncertainties in RQA_Samp results, requiring either the use of RQA_woRP or increasing M . Systematic guidelines for selecting M in non-stationary cases warrant future investigation.

The presented approaches also come with potential trade-offs. The sampling of structures introduces an additional source of variability, and its influence on the robustness and reliability of different RQA measures still needs to be systematically evaluated. Future studies should investigate how sampling density, noise, and parameter settings affect the stability of the obtained measures, particularly for non-stationary dynamics where line structures may be heterogeneously distributed. Another important aspect concerns the integration of established correction schemes, such as those for removing sojourn points [8, 13, 18] or mitigating border effects [3, 13], into the sampling framework.

Beyond methodological refinements, several directions could further enhance the computational efficiency and applicability of these approaches. Combining the proposed sampling strategy with parallel computing architectures could improve scalability for very large datasets and real-time applications. Additionally, the current implementations could benefit from further refinements. Replacing the fixed sampling size M with an adaptive convergence scheme – which stops sampling once results stabilise within a predefined threshold – could optimise the trade-off between accuracy and computation time. Combined with low-level compiler optimisations such as JIT compiler hints and vectorisation [32], these enhancements could yield additional modest speedups.

The strategies discussed here are not limited to the diagonal line measures. They can be analogously applied to RQA measures based on vertical line structures [17] and to recurrence network measures, such as clustering coefficient and transitivity [19], broadening the scope of energy-efficient recurrence analysis.

In summary, the combination of direct line detection and sampling represents a promising way forward in the efficient computation of RQA and related measures. Beyond the methodological advantages, the reduction in computational cost also contributes to the principles of green computing by lowering energy consumption, which makes the approach more sustainable for large-scale or long-term data analysis.

Acknowledgements

The author thanks Felipe Eduardo Lopes da Cruz, Thiago de Lima Prado, and Sergio Roberto Lopes for inspiring discussions on sampling microstates and for motivating this research. The author gratefully acknowledges support from the German Academic Exchange Service (DAAD) and Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) via the project “Recurrence quantifiers as features for machine-learning-decision-making processes” (Grant No. 57705568), as well as from the Ministry of Research, Science and Culture (MWFK) of Land Brandenburg for supporting this project by providing resources on the high-performance computer system “Eunice Newton Foote” at the Potsdam Institute for Climate Impact Research (Grant No. 22-Z105-05/002/001).

Data and code availability

Code used for this analysis and to reproduce the results and figures of this study are available at Zenodo: <https://zenodo.org/10.5281/zenodo.17620044>.

Appendix

Energy consumption per run

Assuming a constant power draw P (e.g., $P = 50\text{ W}$) and a computation time t measured in seconds, the energy consumption per run is

$$E = P \frac{t}{3600} [\text{Wh}].$$

Accordingly, the energy saving obtained when replacing a reference method with runtime t_{ref} by a faster method with runtime t is

$$\Delta E = P \frac{t_{\text{ref}} - t}{3600} [\text{Wh}].$$

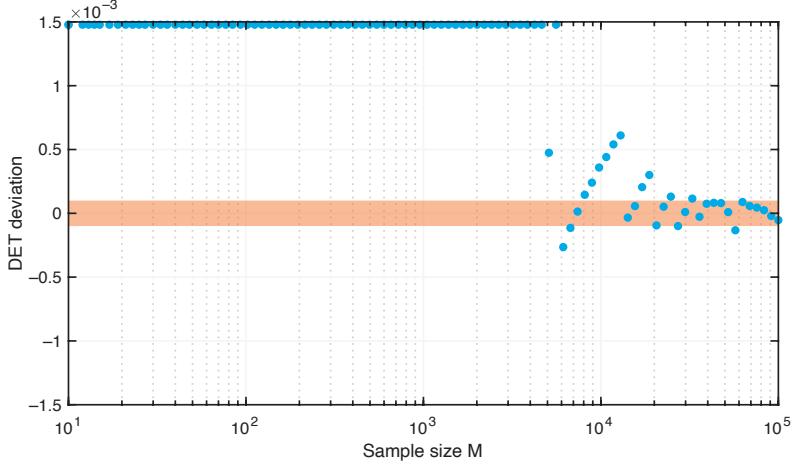


Fig. 4. Deviation of DET derived from recurrence microstates approximation for the Rössler system with 25,000 data points and sampling size ranging from $M = 10$ to 100,000. The orange region indicates the deviation range of $\pm 10^{-4}$. To achieve deviations below 10^{-4} , the sampling size must be at least 40,000. Up to $M = 5,000$, the estimated DET remains at 1.00, resulting in the constant deviation.

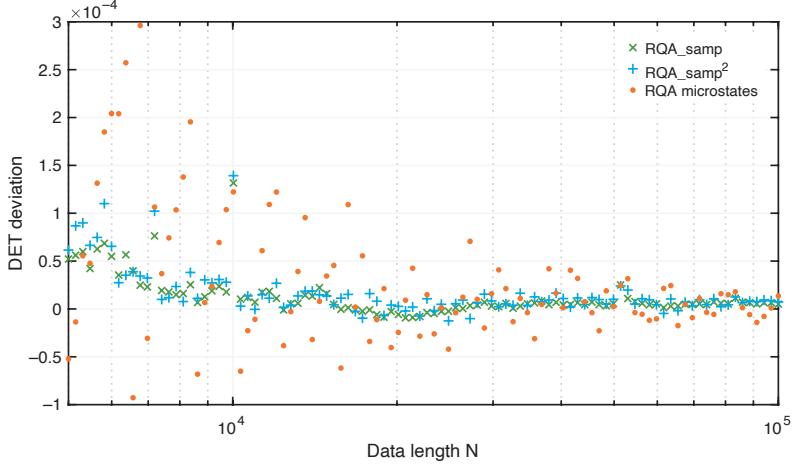


Fig. 5. Deviation $DET_{\text{sampled}} - DET_{\text{true}}$ for DET estimations using RQA_Samp, RQA_Samp², and recurrence microstates approximation for the Rössler system for increasing data length N . The sampling was $M = 100,000$ for RQA_Samp and microstates RQA, and $M = 5,000$ for RQA_Samp². Even for a very small sampling number, the RQA_Samp approach has small deviations $< 10^{-4}$.

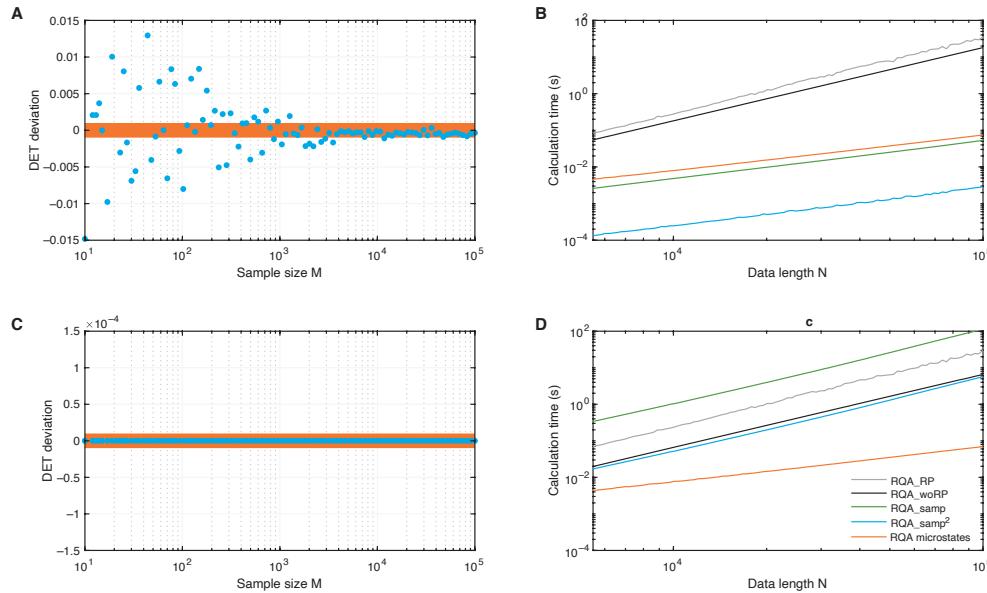


Fig. 6. (A, C) Deviation $DET_{\text{sampled}} - DET_{\text{true}}$ and (B, D) computation times as in Figs. 2 and 3 for the (A, B) the logistic map and (C, D) a periodic signal. In (A, C) results for $N = 25,000$ and using RQA_Samp² are shown.

References

1. Pierre Augier, Carl Friedrich Bolz-Tereick, Serge Guelton, and Ashwin Vishnu Mohanan. Reducing the ecological impact of computing through education and python compilers. *Nature Astronomy*, 5(4):334–335, 2021. doi: 10.1038/s41550-021-01342-y.
2. Ben Bruers, Marilyn Cruces, Markus Demleitner, Guenter Duckeck, Michael Düren, Niclas Eich, Torsten Enßlin, Johannes Erdmann, Martin Erdmann, Peter Fackeldey, Christian Felder, Benjamin Fischer, Stefan Fröse, Stefan Funk, Martin Gasthuber, Andrew Grimshaw, Daniela Hadasch, Moritz Hannemann, Alexander Kappes, Raphael Kleinemühl, Oleksiy M. Kozlov, Thomas Kuhr, Michael Lupberger, Simon Neuhaus, Pardis Niknejadi, Judith Reindl, Daniel Schindler, Astrid Schneidewind, Frank Schreiber, Markus Schumacher, Kilian Schwarz, Achim Streit, R. Florian von Cube, Rodney Walker, Cyrus Walther, Sebastian Wozniewski, and Kai Zhou. Resource-aware research on universe and matter: call-to-action in digital transformation. *The European Physical Journal – Special Topics*, pages 1–17, 2024. ISSN 1951-6355. doi: 10.1140/epjs/s11734-024-01436-4.
3. F. Censi, G. Calcagnini, and S. Cerutti. Proposed corrections for the quantification of coupling patterns by recurrence plots. *IEEE Transactions on Biomedical Engineering*, 51(5):856–859, 2004. doi: 10.1109/TBME.2004.826594.
4. M. Colafranceschi, M. Papi, A. Giuliani, G. Amiconi, and A. Colosimo. Simulated Point Mutations in the A α -Chain of Human Fibrinogen Support a Role of the A α C Domain in the Stabilization of Fibrin Gel. *Pathophysiology of Haemostasis and Thrombosis*, 35(6), 2006.
5. G. Corso, T. D. L. Prado, G. Z. dos Santos Lima, J. Kurths, and S. R. Lopes. Quantifying entropy using recurrence matrix microstates. *Chaos*, 28(8):083108, 2018. doi: 10.1063/1.5042026.
6. F. E. L. da Cruz, T. D. L. Prado, S. R. Lopes, N. Marwan, and J. Kurths. Density-based recurrence measures from microstates. *Physical Review E*, 111: 044212, 2025. doi: 10.1103/PhysRevE.111.044212.
7. J.-P. Eckmann, S. Oliffson Kamphorst, and D. Ruelle. Recurrence Plots of Dynamical Systems. *Europhysics Letters*, 4(9):973–977, 1987. doi: 10.1209/0295-5075/4/9/004.
8. J. B. Gao. Recurrence Time Statistics for Chaotic Systems and Their Applications. *Physical Review Letters*, 83(16):3178–3181, 1999. doi: 10.1103/PhysRevLett.83.3178.
9. E. García-Ochoa, J. González-Sánchez, N. Acuña, and J. Euan. Analysis of the dynamics of Intergranular corrosion process of sensitised 304 stainless steel using recurrence plots. *Journal of Applied Electrochemistry*, 39(5):637–645, 2009. doi: 10.1007/s10800-008-9702-4.
10. Petra Icha and Thomas Lauf. Entwicklung der spezifischen Treibhausgas-Emissionen des deutschen Strommix in den Jahren 1990 – 2024. Technical report, Umweltbundesamt, April 2025. URL <https://www.umweltbundesamt.de/publikationen/entwicklung-der-spezifischen-treibhausgas-11>.
11. Jóakim von Kistowski, Hansfried Block, John Beckett, Cloyce Spradling, Klaus-Dieter Lange, and Samuel Kounev. Variations in CPU power consumption. *Proceedings of the 7th ACM/SPEC on International Conference on Performance Engineering*, pages 147–158, 2016. doi: 10.1145/2851553.2851567.
12. Bartłomiej Kocot, Paweł Czarnul, and Jerzy Proficz. Energy-aware scheduling for high-performance computing systems: A survey. *Energies*, 16(2):890, 2023. doi: 10.3390/en16020890.
13. K. H. Kraemer and N. Marwan. Border effect corrections for diagonal line based recurrence quantification analysis measures. *Physics Letters A*, 383(34):125977,

2019. doi: 10.1016/j.physleta.2019.125977.
- 14. N. Marwan. A Historical Review of Recurrence Plots. *European Physical Journal – Special Topics*, 164(1):3–12, 2008. doi: 10.1140/epjst/e2008-00829-1.
 - 15. N. Marwan. A Bibliographic View on Recurrence Plots and Recurrence Quantification Analyses. In Y. Hirata, M. Shiro, M. Fukino, C. L. Webber Jr., K. Aihara, and N. Marwan, editors, *Recurrence Plots and Their Quantifications: Methodological Breakthroughs and Interdisciplinary Discoveries*, pages 1–27. Springer, Cham, 2025. ISBN 978-3-031-91061-6. doi: 10.1007/978-3-031-91062-3_1.
 - 16. N. Marwan and K. H. Kraemer. Trends in recurrence analysis of dynamical systems. *European Physical Journal – Special Topics*, 232:5–27, 2023. doi: 10.1140/epjs/s11734-022-00739-8.
 - 17. N. Marwan, N. Wessel, U. Meyerfeldt, A. Schirdewan, and J. Kurths. Recurrence Plot Based Measures of Complexity and its Application to Heart Rate Variability Data. *Physical Review E*, 66(2):026702, 2002. doi: 10.1103/PhysRevE.66.026702.
 - 18. N. Marwan, M. C. Romano, M. Thiel, and J. Kurths. Recurrence Plots for the Analysis of Complex Systems. *Physics Reports*, 438(5–6):237–329, 2007. doi: 10.1016/j.physrep.2006.11.001.
 - 19. N. Marwan, J. F. Donges, Y. Zou, R. V. Donner, and J. Kurths. Complex network approach for recurrence analysis of time series. *Physics Letters A*, 373(46):4246–4254, 2009. doi: 10.1016/j.physleta.2009.09.042.
 - 20. N. Marwan, S. Foerster, and J. Kurths. Analysing spatially extended high-dimensional dynamics by recurrence plots. *Physics Letters A*, 379(10–11):894–900, 2015. doi: 10.1016/j.physleta.2015.01.013.
 - 21. Norbert Marwan. RQA_openmp. *Zenodo*, 2025. doi: 10.5281/zenodo.16786351.
 - 22. M. Mosdorf. Method for detecting software anomalies based on recurrence plot analysis. *Journal of Theoretical and Applied Computer Science*, 6(1):3–12, 2012. URL <http://www.jtacs.org/archive/2012/1/1>.
 - 23. Jonathan Muraña, Sergio Nesmachnow, Fermín Armenta, and Andrei Tchernykh. Characterization, modeling and scheduling of power consumption of scientific computing applications in multicores. *Cluster Computing*, 22(3):839–859, 2019. ISSN 1386-7857. doi: 10.1007/s10586-018-2882-8.
 - 24. Showmick Guha Paul, Arpa Saha, Mohammad Shamsul Arefin, Touhid Bhuiyan, Al Amin Biswas, Ahmed Wasif Reza, Naif M. Alotaibi, Salem A. Alyami, and Mohammad Ali Moni. A comprehensive review of green computing: Past, present, and future research. *IEEE Access*, 11:87445–87494, 2023. ISSN 2169-3536. doi: 10.1109/access.2023.3304332.
 - 25. Lorena Poenaru-Olaru, June Sallou, Luis Cruz, Jan Rellermeyer, and Arie van Deursen. Sustainable machine learning retraining: Optimizing energy efficiency without compromising accuracy. *arXiv*, 2025. doi: 10.48550/arxiv.2506.13838.
 - 26. T. Rawald, M. Sips, N. Marwan, and D. Dransch. Fast Computation of Recurrences in Long Time Series. In N. Marwan, M. A. Riley, A. Giuliani, and C. L. Webber, Jr., editors, *Translational Recurrences – From Mathematical Theory to Real-World Applications*, volume 103, pages 17–29. Springer, Cham, 2014. doi: 10.1007/978-3-319-09531-8_2.
 - 27. T. Rawald, M. Sips, N. Marwan, and U. Leser. Massively Parallel Analysis of Similarity Matrices on Heterogeneous Hardware. In *Proceedings of the EDBT/ICDT Joint Conference 2015*, pages 56–62, 2015. URL <http://ceur-ws.org/Vol-1330/#paper-11>.
 - 28. T. Rawald, M. Sips, and N. Marwan. PyRQA – Conducting Recurrence Quantification Analysis on Very Long Time Series Efficiently. *Computers & Geosciences*, 104:101–108, 2017. doi: 10.1016/j.cageo.2016.11.016.
 - 29. O. E. Rössler. An equation for continuous chaos. *Physics Letters A*, 57(5):397–398, 1976. doi: 10.1016/0375-9601(76)90101-8.

30. Rafał Różycki, Dorota Agnieszka Solarska, and Grzegorz Waligóra. Energy-aware machine learning models—a review of recent techniques and perspectives. *Energies*, 18(11):2810, 2025. doi: 10.3390/en18112810.
31. D. Schultz, S. Spiegel, N. Marwan, and S. Albayrak. Approximation of diagonal line based measures in recurrence quantification analysis. *Physics Letters A*, 379 (14–15):997–1011, 2015. doi: 10.1016/j.physleta.2015.01.033.
32. Malcolm Sherrington. *Mastering Julia: Enhance your analytical and programming skills for data modeling and processing with Julia*. Packt Publishing Ltd, 2024.
33. S. Spiegel, D. Schultz, and N. Marwan. Approximate Recurrence Quantification Analysis (aRQA) in Code of Best Practice. In C. L. Webber, Jr., C. Ioana, and N. Marwan, editors, *Recurrence Plots and Their Quantifications: Expanding Horizons*, pages 113–136. Springer, Cham, 2016. doi: 10.1007/978-3-319-29922-8_6.
34. Dynamics UFPR. RecurrenceMicrostatesAnalysis.jl. *Github*, 2025. URL <https://github.com/DynamicsUFPR/RecurrenceMicrostatesAnalysis.jl>.
35. C. L. Webber, Jr. and N. Marwan. *Recurrence Quantification Analysis – Theory and Best Practices*. Springer, Cham, 2015. ISBN 978-3-319-07154-1. doi: 10.1007/978-3-319-07155-8.
36. J. P. Zbilut and C. L. Webber, Jr. Embeddings and delays as derived from quantification of recurrence plots. *Physics Letters A*, 171(3–4):199–203, 1992. doi: 10.1016/0375-9601(92)90426-M.
37. Shaoqing Zhang, Haohuan Fu, Lixin Wu, Yuxuan Li, Hong Wang, Yunhui Zeng, Xiaohui Duan, Wubing Wan, Li Wang, Yuan Zhuang, Hongsong Meng, Kai Xu, Ping Xu, Lin Gan, Zhao Liu, Sihai Wu, Yuhu Chen, Haining Yu, Shupeng Shi, Lanning Wang, Shiming Xu, Wei Xue, Weiguo Liu, Qiang Guo, Jie Zhang, Guanghui Zhu, Yang Tu, Jim Edwards, Allison Baker, Jianlin Yong, Man Yuan, Yangyang Yu, Qiuying Zhang, Zedong Liu, Mingkui Li, Dongning Jia, Guangwen Yang, Zhiqiang Wei, Jingshan Pan, Ping Chang, Gokhan Danabasoglu, Stephen Yeager, Nan Rosenbloom, and Ying Guo. Optimizing high-resolution community earth system model on a heterogeneous many-core supercomputing platform. *Geoscientific Model Development*, 13(10):4809–4829, 2020. doi: 10.5194/gmd-13-4809-2020.