2nd Workshop on ML4Sys and Sys4ML @ BTW25

Learned Compression of Nonlinear Time Series With Random Access

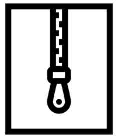
Andrea Guerra, **Giorgio Vinciguerra**, Antonio Boffa, Paolo Ferragina

Università di Pisa

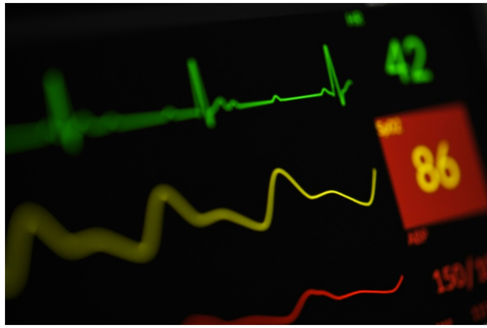
Time Series Data Management

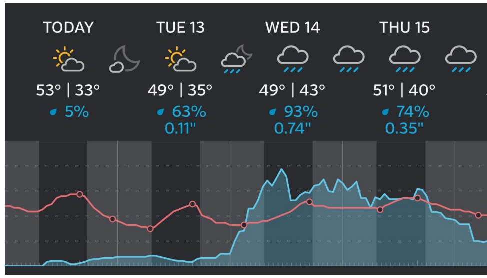
- Massive time series data in IoT, crypto and finance, healthcare, environmental and system monitoring
- Invaluable for analysis, forecasting, and decision-making
- Storage is increasingly challenging
  - Impacting transmission, and real-time analysis
  - Cloud storage costs increase with data volume

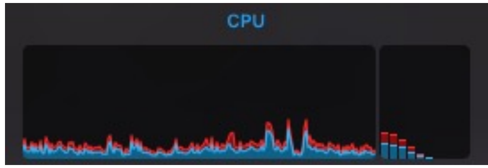
Set **data retention policies**
...but lose historical data



Data compression

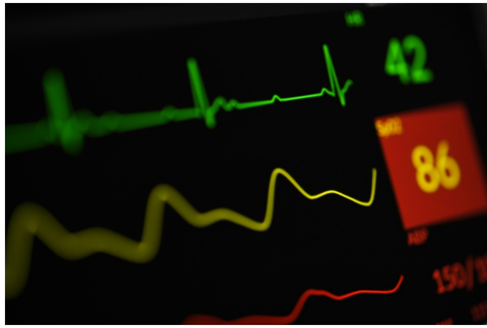


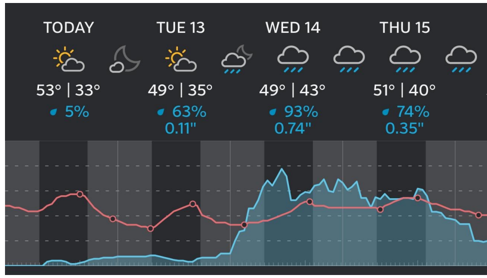


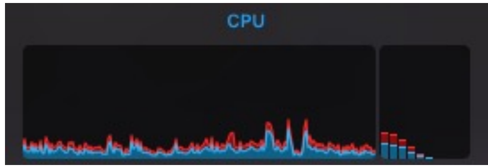


Time Series Compression

- **General-purpose compressors**
  - Zstd, Lz4, Brotli, …
  - \+ Good compression ratios
  - \- Significant de/compression overhead
- **Special-purpose compressors**
  - Gorilla [VLDB 15], Chimp [VLDB 22], ALP [SIGMOD 24], …
  - \+ Optimized for fast de/compression
  - \- Lower compression effectiveness
- **Limited focus on efficient random access**
- **TS data can often be approximated by nonlinear functions**







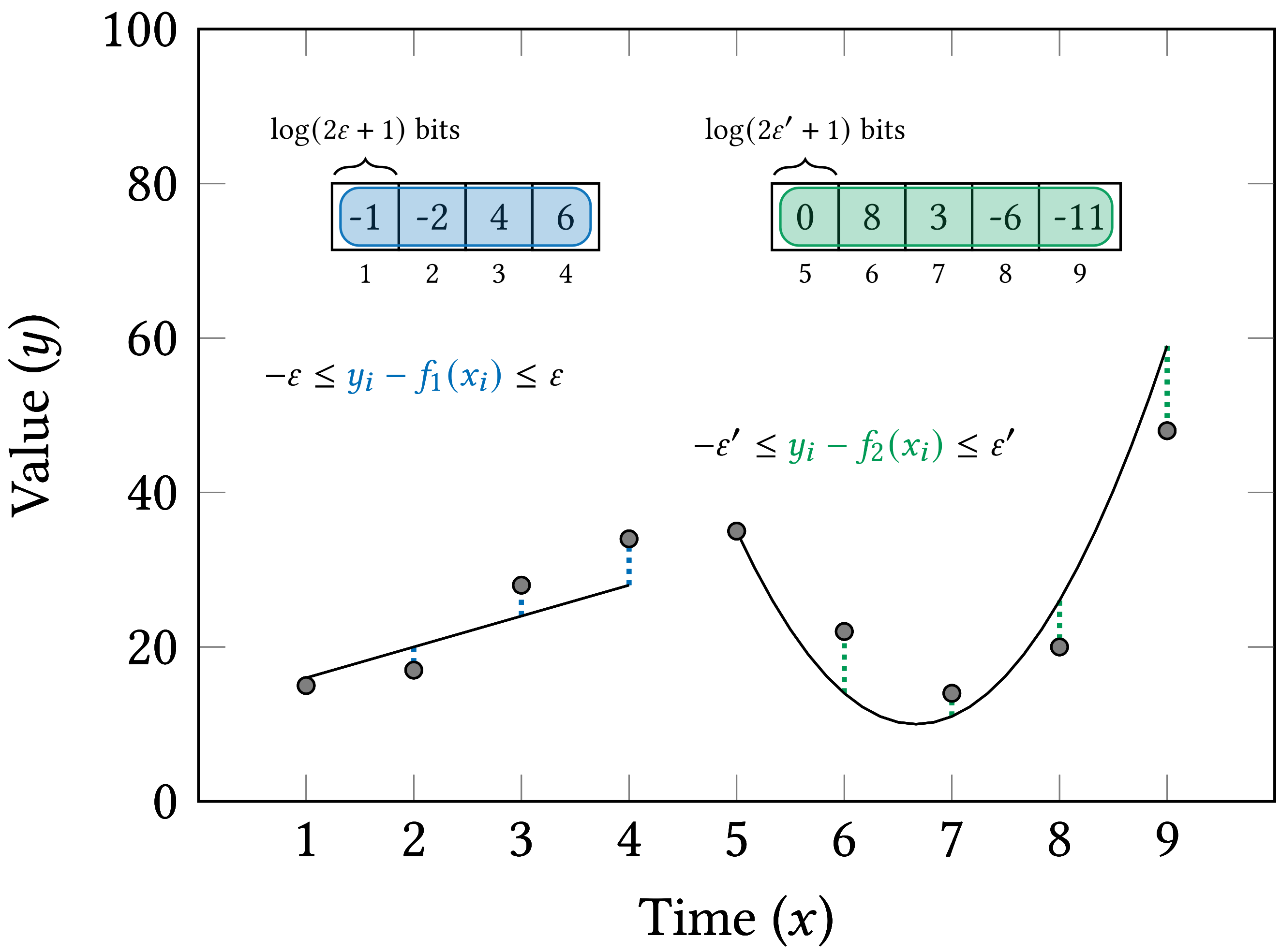
NeaTS: An overview

Lossy compression, with bounded errors

Lossless compression

Problem 1: How to compute nonlinear ε-approximations?

Problem 2: How to orchestrate different function types to minimize the space?

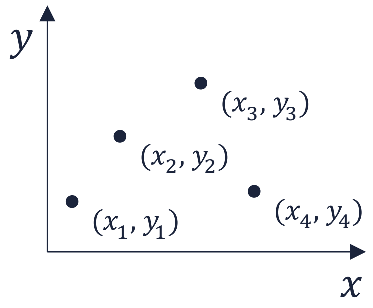


Problem 3: How to enable random access?

Problem 1: Computing nonlinear ε-approximations

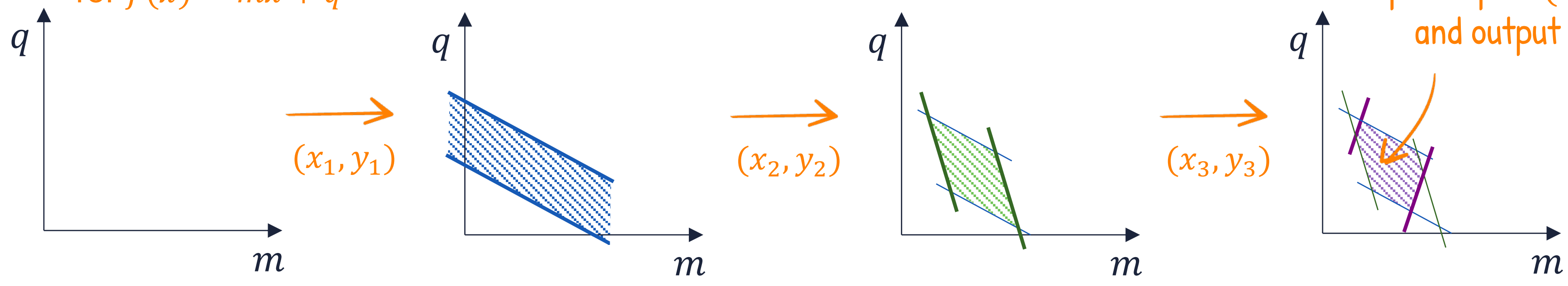
Start from the algorithm by O'Rourke [CACM 1981] for linear functions

Input points



Feasible coefficients for $f(x) = mx + q$

If the next constraints make this region empty, we pick a pair (m, q) here, and output a line



$|(mx_1 + q) - y_1| \le \varepsilon$
$q \le (-x_1)m + (y_1 + \varepsilon)$
$q \ge (-x_1)m + (y_1 - \varepsilon)$

$q \le (-x_2)m + (y_2 + \varepsilon)$
$q \ge (-x_2)m + (y_2 - \varepsilon)$

$q \le (-x_3)m + (y_3 + \varepsilon)$
$q \ge (-x_3)m + (y_3 - \varepsilon)$

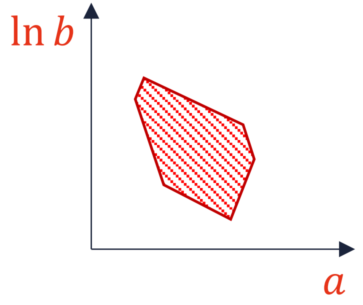
Key idea: constraints arising from some nonlinear functions can be linearized, and O'Rourke's algorithm still works

E.g.: Given $f(x) = be^{ax}$

Transform $|(be^{ax_k}) - y_k| \le \varepsilon$ into

$$\underbrace{\ln b}_{q} \le (-x_k)\, a + \ln(y_k + \varepsilon)$$
$$\underbrace{\ln b}_{q} \ge (-x_k)\, \underbrace{a}_{m} + \ln(y_k - \varepsilon)$$

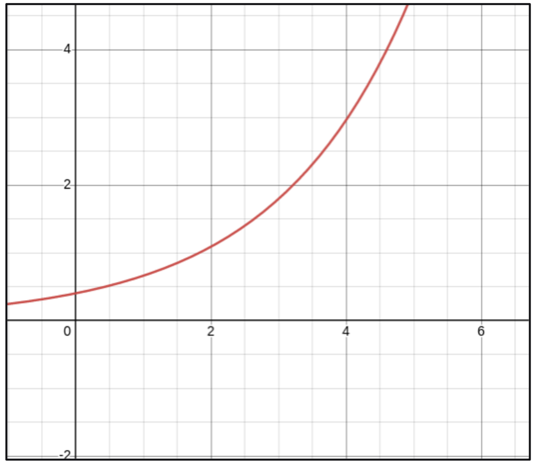
Feasible coefficients

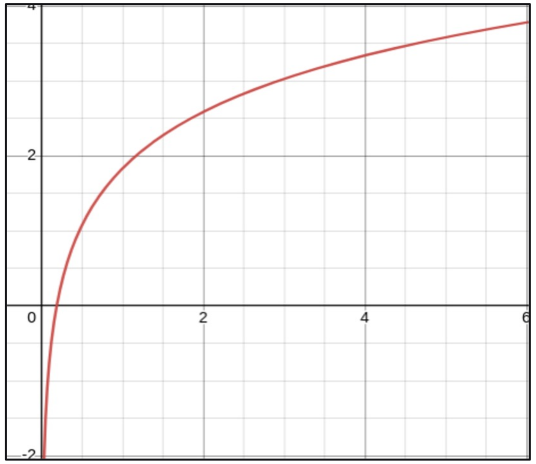


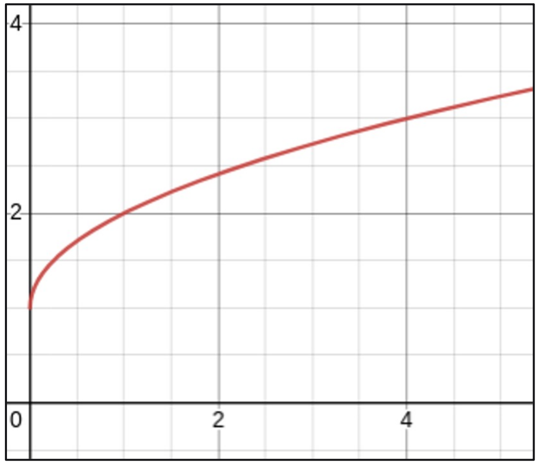
- Piecewise function with the minimal number of pieces
- Runs in $O(n)$ time

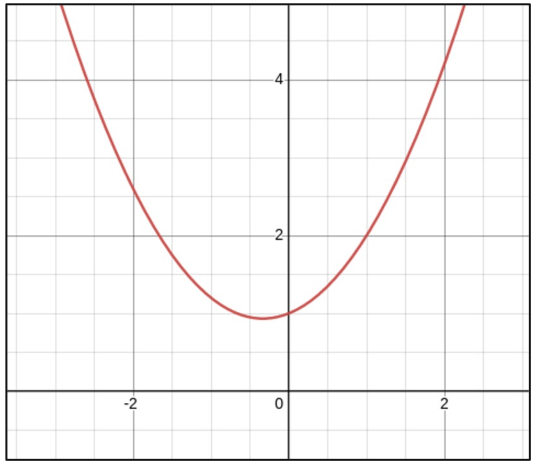
Some example of functions we can use

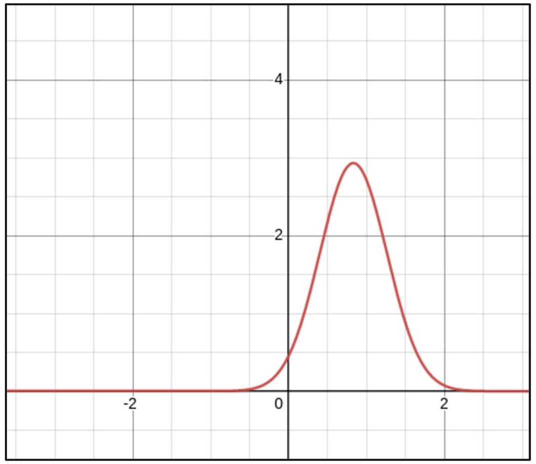
- **Exponential** $f(x) = be^{ax}$
- **Logarithmic** $f(x) = \log(ax^b)$
- **Radical** $f(x) = a\sqrt{x} + b$
- **Quadratic** $f(x) = ax^2 + bx + c$
- **Bell-shaped** $f(x) = e^{ax^2+bx+c}$



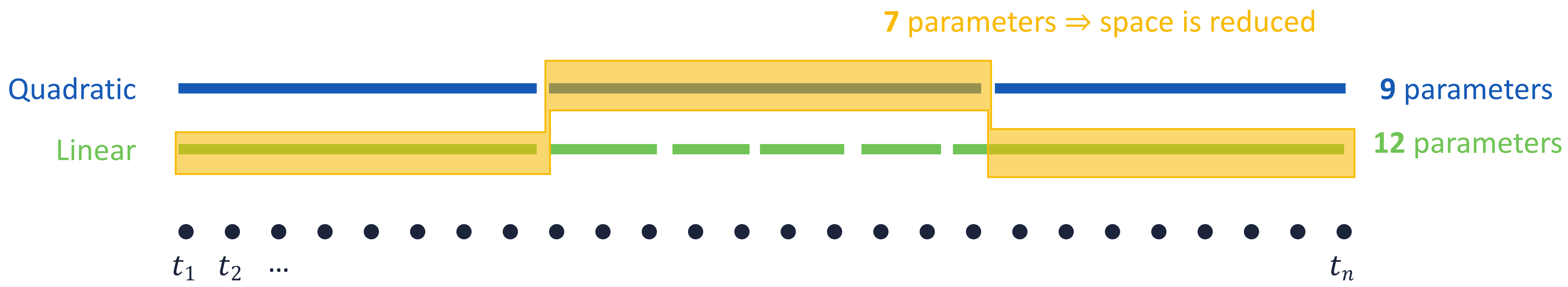




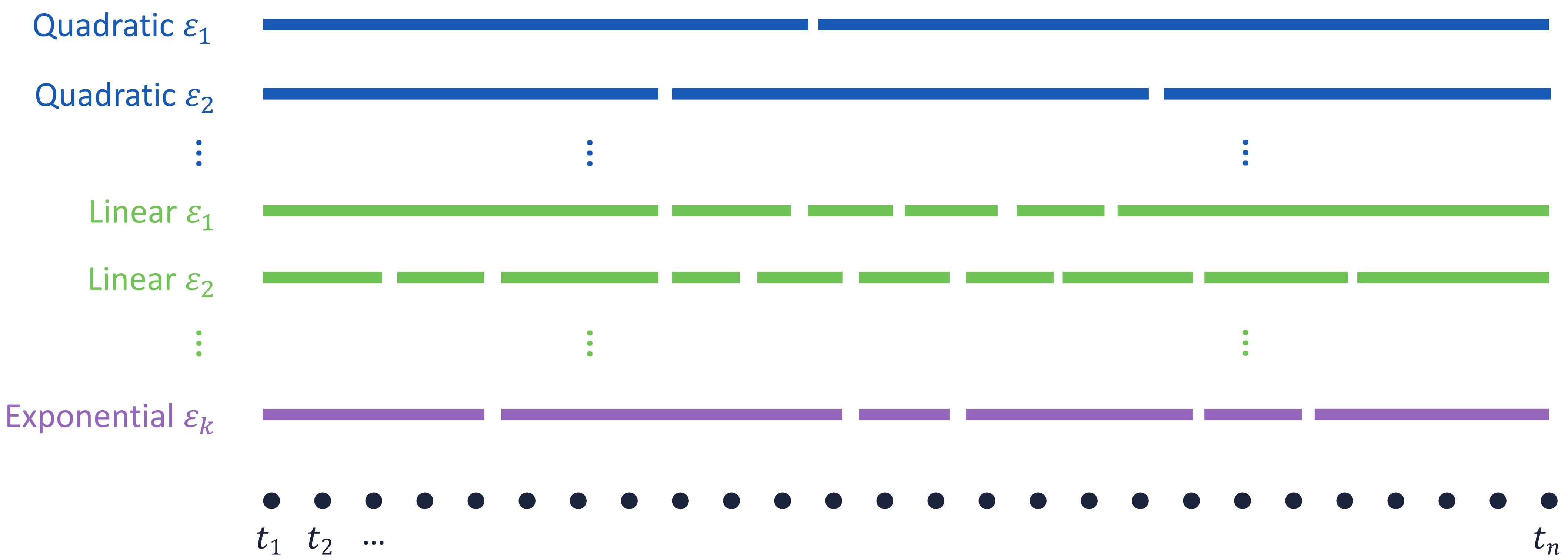




Problem 2: Orchestrate different function types



Problem 2: Orchestrate different function types



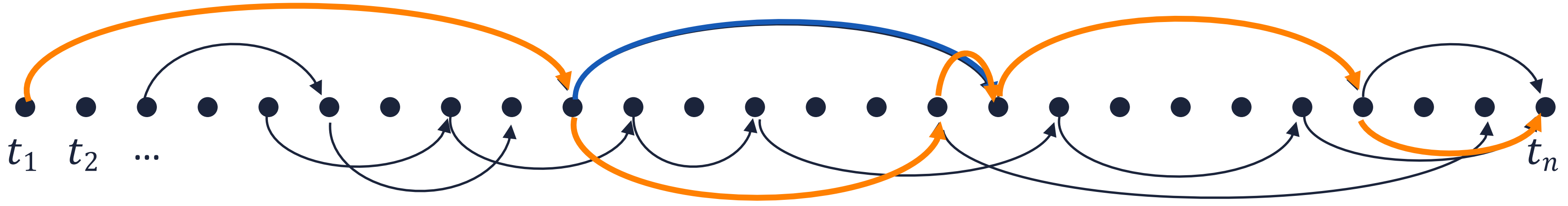
How to efficiently find the partition that minimises the space?

Problem 2: Orchestrate different function types

1. Transform each data point into a node in a graph

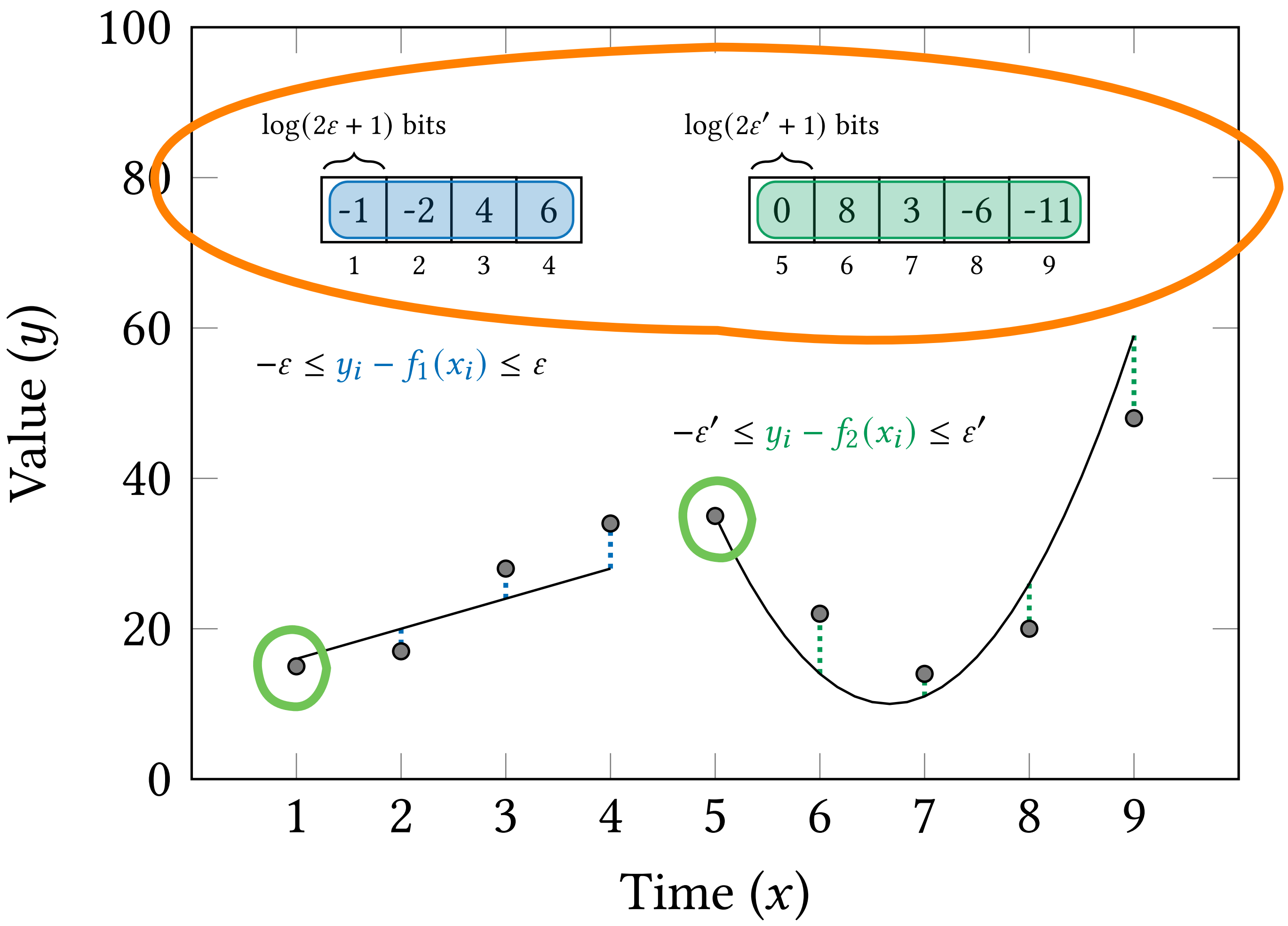
2. Transform each function covering $\mathrm{TS}[i, j]$ into an edge $i \rightarrow j$

3. Set $w(i, j)$ = Bit-size of $\mathrm{TS}[i, j]$ encoded with f_ε



4. Compute the shortest path in the graph

Problem 3: Enabling random access in NeaTS



1. Store the residuals and function parameters in packed arrays

2. Store the functions' starting timestamps into succinct indexes

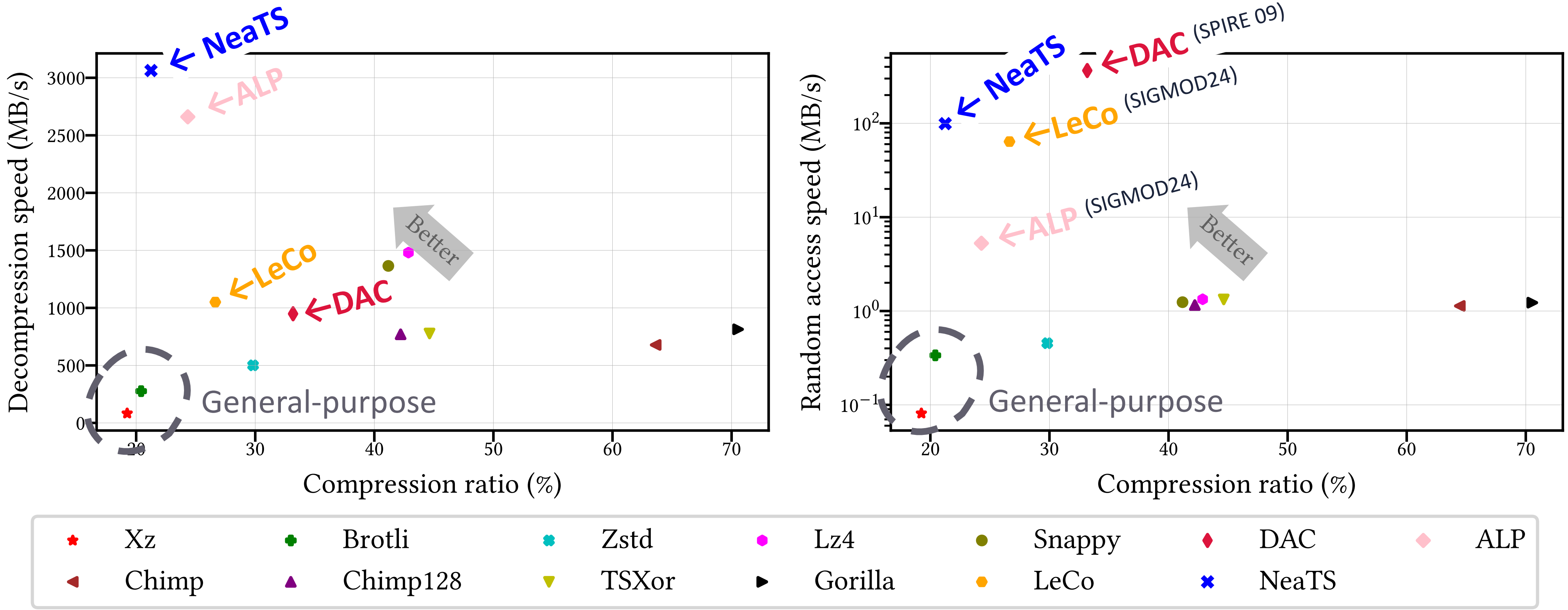
Experimental setting

- **16 datasets:** stock prices, ECG signals, sensors data, trajectories
  - 1.5 billion values
  - Plots represent average across all datasets
- **Metrics:** compression ratio, de/compression and random-access speed
- Compressors without random access are applied to blocks of 1K values
- NeaTS uses linear, exponential, quadratic, and radical functions
- SIMD instructions in NeaTS and ALP (SIGMOD 24)

Lossy version of NeaTS

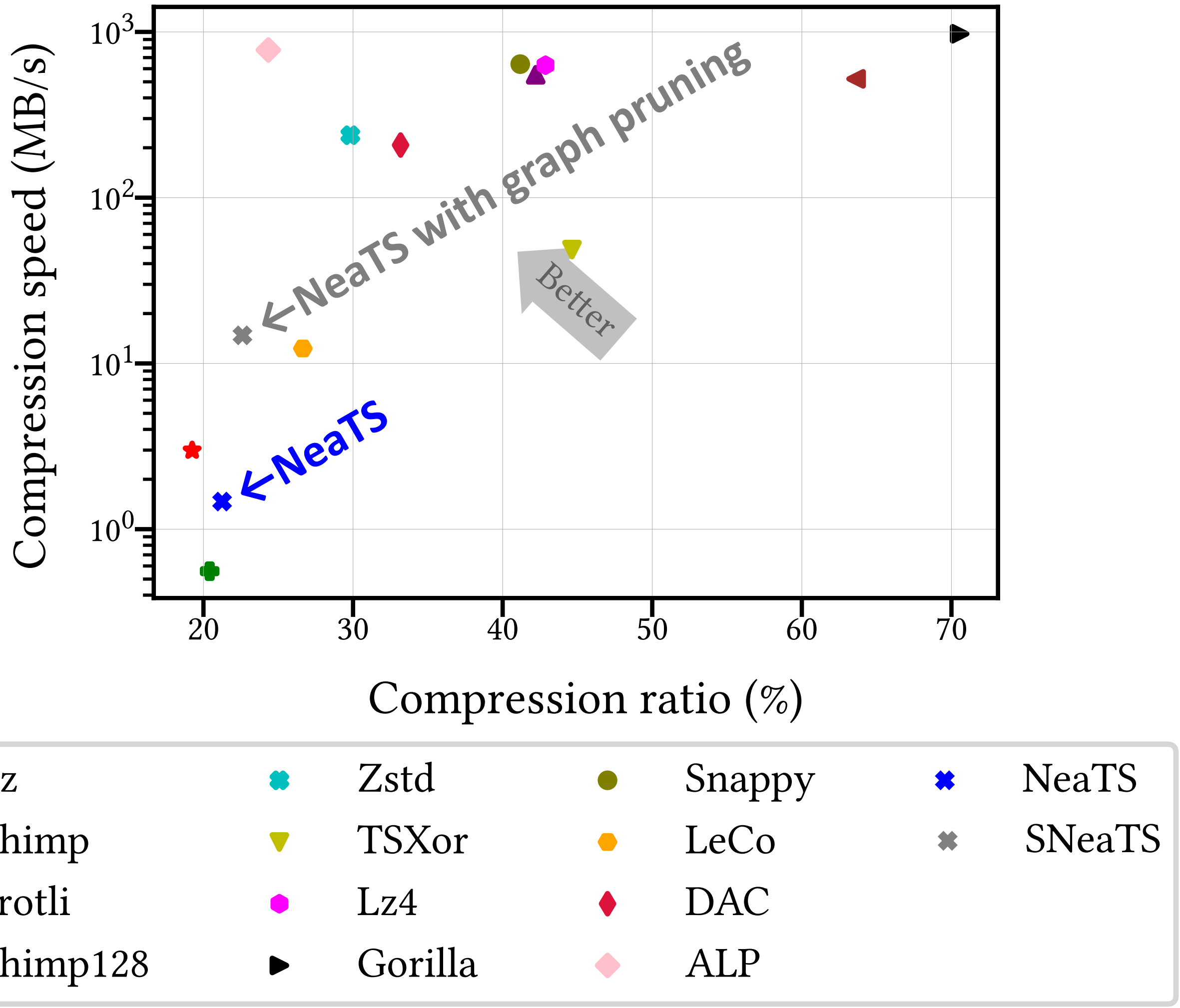
- Compared with algorithms that bound the infinity norm of residuals
- Improved compression ratio:
  - By **7%** wrt linear models only (CACM81)
  - By **12%** wrt Adaptive Approximation (EDBT12)

Lossless compressors

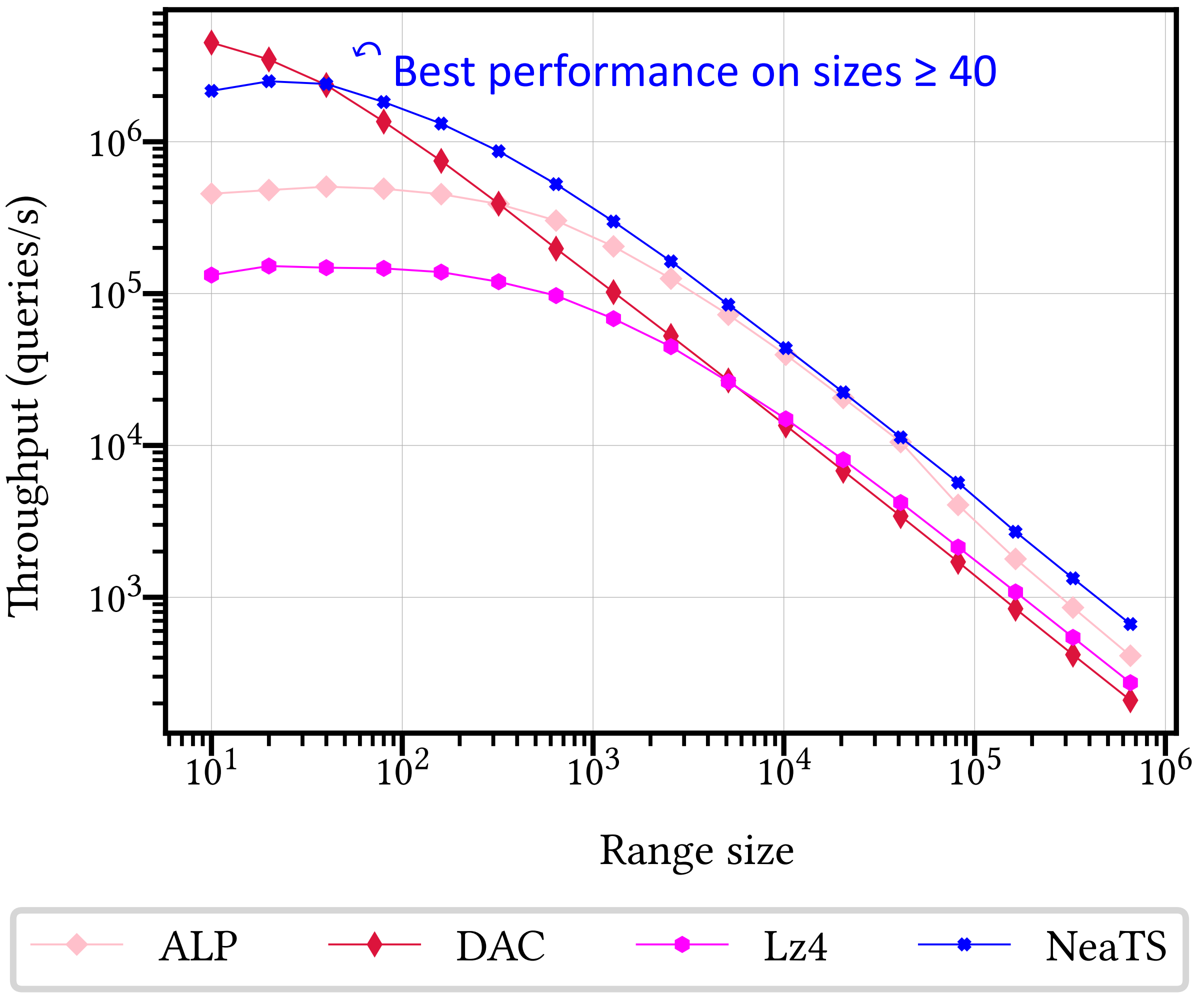


NeaTS obtains simultaneously compression ratios close to Xz/Brotli, up to 1 order of magnitude faster decompression, and 3 orders of magnitude faster random-access

Compression speed



Range scans performance



Conclusion

- **NeaTS** compresses the time series with error-bounded nonlinear functions of different shapes
- **Simultaneously:** compression ratios of general-purpose compressors, up to 1 order of magnitude faster decompression, up to 3 orders of magnitude faster random-access speed
- No other compressor could achieve a good trade-off across all these factors

Future work:

- Compress similar TS fragments together (exploit seasonality)
- Error-bounded piecewise *nonlinear* functions in learned indexing, hashing, sorting