The Explainability of Time Series Downsampling

A Preprint

Morgan Frodsham
School of Computing
Newcastle University
Newcastle upon Tyne, UK
`M.C.M.Frodsham2@newcastle.ac.uk`

Matthew Forshaw
School of Computing
Newcastle University
Newcastle upon Tyne, UK
`matthew.forshaw@newcastle.ac.uk`

August 12, 2023

Abstract

Trusting data-driven decision-making goes beyond demonstrating comppliance with legal, regulatory and ethical obligations; decision-makers need to trust how the data is used. Handling, storing and visualising the volume of data being generated today requires data practitioners to make assumptions and processing choices that remain opaque to decision-makers. The research outlined by this paper explores decision-makers' trust in data, data pracitioners' experience of communicating data insights to decision-makers and a new visualisation methodology for explaining the impact of downsampling on high-volume time series data. It uniquely combines user research with R packages `imputeTS` [1] and `Rcatch22` [2] to identify and visualise time series features that are most sensitive to downsampling. It is hoped this will improve decision-makers' trust in data by helping data practitioners to create transparency in the data processing pipeline, communicate the impact of downsampling, and support conversations about which algorithms or parameters are most appropriate for particular decision-maker use cases.

***K*eywords** time series · downsampling · decision-making · visualisation

1 INTRODUCTION

HM Government is committed to making data-driven decisions that engender public trust [3]–[6]. Data-driven decisions are considered to be "more well-informed" [3], effective [6], consistent [5], and better "at scale" [4]. Despite this, there is a lack of trust in government use of data [7]. This suggests that public trust in data-driven decisions goes beyond how the "data complies with legal, regulatory and ethical obligations" [5]. The UK public need to have "confidence and trust in how data, including personal data, is used" [4], and this requires transparency [7].

To make data-driven decisions, government decision-makers also need to trust how the data used (cite user research here). This means trusting which data points are selected, how this data collected and stored, and the capability of data practitioners to understand the quality, insights and limitations of it. At every stage of the data processing pipeline, data practitioners have the opportunity to communicate the impact of the assumptions and choices they are making to support decision-makers in trusting the data informing their decisions.

Time series data is used across HM Government [8] to inform decision-makers across various domains [9]. It is also widely generated and used by industry and research [10]. The volume of time series data is continuously increasingly [11], posing significant challenges for handling and visualising this popular data type [10]. Data practitioners must utilise methods that reduce data volumes to align with limitations like processing time, computing costs, storage capabilities, and sustainability ambitions [10], [12], [13].

Downsampling is an established technique [14], [15] that involves selecting a representative subset of data to preserve its original shape while reducing the number of data points [11], [16]. This is a vital part of making voluminous time series understandable for human observation [12] and an essential step in many time series database solutions [11]. However, little attention has been devoted to how downsampling impacts decision-makers trust in the data.

Despite widespread use, how to communicate the impact of downsampling algorithms on time series data remains also understudied [11], [12]. Downsampling expands the boundaries of risk for decision-makers as data practitioners may not realise the significance of the data being discarded. Such choices throughout the data pipeline may have disproportionately larger consequences later as their ramifications for future decisions are not fully understood by all. It is important, therefore, that data practitioners are able to communicate the impact of choices made throughout the data pipeline.

To address these challenges, this paper shares initial insights from user research on the impact of downsampling on decision-makers' trust in data and suggests a visualisation methodology for communicating the impact of downsampling algorithms on time series. This methodology combines user research with R packages `imputeTS` [1] and `Rcatch22` [2] to identify and visualise time series features that are most sensitive to downsampling. It is hoped this will improve decision-makers' trust in data by helping data practitioners to create transparency in the data processing pipeline, communicate the impact of downsampling, and support conversations about which algorithms or parameters are most appropriate for particular decision-maker use cases.

2 RELATED WORK

This section provides an overview of previous related work to create a clear understanding of the most relevant fields of research and identify the gaps being addressed by the paper.

2.1 Data Transparency

Technology transparency, "including institutional data practices", is sociopolitical in nature [17]. There is a growing number of researchers reflecting on "societal needs in terms of what is made transparent, for whom, how, when and in what ways, and, crucially, who decides" [18].

The implicit assumption behind calls for transparency is that "seeing a phenomenon creates opportunities and obligations to make it accountable and thus to change it" [19]. However, without sufficient agency to explore the information being shared, seeing a phenomenon often results in "information overload" [20] that obfuscates or diverts [21]. Without agency, transparency is increasingly considered to be a fallacy [22].

Meaningful transparency is only realised when the information is provided with the tools to turn "access to agency" [19], [22]. This suggests that data practitioners communicating the assumptions and choices made throughout the data processing pipeline with decision-makers is not likely to create trust in how the data is used. Instead, data practitioners should be encouraged to find tools, such as interactive visualisations [11], that put agency into the hands of decision-makers.

2.2 Time series visualisation

Time series data is commonly visualised as a line graph [12], [23]. Line graphs help the human eye to observe only the most important data points [12] by convey the overall shape and complexity of the time series data [11], [14]. The most effective time series visualisations are, however, interactive [23], [24], turning access into agency [22] by allowing the user to access details on demand. Evaluation of time series visualisation is, therefore, a growing field of research [23].

However, this growing field of research does not extend to visualisations of choices and assumptions made during data processing pipline. Indeed, such visualisations are a side effect of the research. This dynamic is exemplified by the R package `imputeTS` [1] where the impact of imputation choices made by the user is only visualised to support the user through the complete process of replacing missing values in time series [25]. The research set out in this paper harnesses the capabilities of `imputeTS` and its 'process' visualisations to help data practitioners communicate the impact of downsampling choices made in the data processing pipeline.

2.3 Value Preserving Data Aggregation

Technological innovation has generated unprecedented amount of time series data and this data continues to grow [4], [10], [26], [27]. For example, tackling climate change is the UK Government's "number one international priority" [28], yet climate simulations that help inform decision-makers generate tens of terabytes per second [10], [29]. Downsampling (value preserving data aggregation) plays an important role in addressing how this voluminous data is processed, stored [10] and visualised [12], [30] by minimising computing resources needed [10], reducing network latency, and improving rendering time [11], [16].

An overview of commonly used downsampling (value preserving data aggregation) algorithms is provided in the table below:

Table 1: Downsampling Algorithms

Algorithm	Description
EveryNth	Selects every nth datapoint for downsampling.
Percentage Change	TBC
Mode-Median Bucket	Finds mode or median within equally sized data buckets for selection.
Min-Std-Error-Bucket	Uses standard error of the estimate (SEE) in linear regression for downsampling.
MinMax	Preserves minimum and maximum of data buckets.
OM^3	Maintains the minimum and maximum values at every time interval for rasterizing.
M4	Combines EveryNth and MinMax, selecting the first and last values of each data bucket as well as the minimum and maximum values.
Longest-Line Bucket	Calculates line length (Euclidean distance) instead of standard error between buckets.
Largest-Triangle One-Bucket	All the data points are ranked by calculating their effective areas. One point with the highest rank (largest effective area) is selected to represent each bucket.
Largest Triangle Three Buckets	The data point that forms the largest triangular surface between the previously selected data point and the average value of the next data bucket.
MinMaxLTTB	Preselects data using MinMax before applying Largest Triangle Three Buckets.

[11], [12], [16], [30]–[33]

Data practitioners have made recent advances in the performance and evaluation of downsampling approaches [11], [14]–[16], [24], [30]–[32]. These advances focus on the effectiveness of the algorithm in delivering downsampled data that represents the original data as accurately as possible. This is vital part of enabling and improving data-driven decision-making, but is focused on supporting data practitioners in their analysis of the data. Instead, the research set out in this paper aims to support data practitioners to communicate the impact of their downsampling choices for decision-makers.

2.4 Time series feature analysis

The increasing size of modern time series data sets has also generated new research into the dynamical characteristics of time series [34]. These characteristics are often used to identify features that enable efficient clustering and classification of time series data, especially for machine learning. A comprehensive library of such features is the *hctsa* (highly comparative time series analysis) toolbox. This shares the 4791 best performing features after computationally comparing thousands of features from across scientific time series analysis literature [35].

Utilising such a library, however, is computationally expensive [34]. C. H. Lubba et. al have attempted to address this by identified a subset of 22 features that are tailored to time series data mining tasks [34], [36]. Although further research is needed to evaluate the relative performance of different feature sets on different types of problems, `catch22` performs well against other feature libraries across 800 diverse real-world and model-generated time series [37].

Features used to classify time series data could provide a common framework by which to consistently compare different downsampling algorithms and parameters. The research set out in this paper utilises the `Rcatch22` subset of features to explore impact of downsampling and create a visual tool for explaining this impact.

3 MOTIVATION

The motivation for this research is grounded in the author's experience as a UK Civil Servant new to the domain of data science. Personal experience suggested that data practitioners found it difficult to communicate the impact of assumptions, choices and limitations of the data processing pipeline to decision-makers without an analytically background; in parallel, decision-makers appeared to desire greater clarity of how the data practitioner had generated the data insights in order to make data-driven decisions. To better understand this dynamic, and test the assumptions of the author, sixteen government officials were interviewed: nine decision-makers and seven data practitioners.

3.1 Decision-makers

- summary of what was done
- top themes
- example diagram
- summarise how this informed research design
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-

3.2 Data Practitioners

- summary of what was done
- top themes
- example diagram
- summarise how this informed research design
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-

•
•
•
•
•
•
•

4 METHODOLOGY

This section provides an overview of process used to develop the work outlined by this paper and highlight key decisions in the research design.

4.1 Data

The research outlined by this report starts with the eight demonstrative time series shared by the Alan Turing Institute `Annotated Change` GitHub repository [38]. The demonstrative time series are synthetic with known change points that provide examples of different types of time series. These synthetic time series (demo_100, demo_200, demo_300, demo_400, demo_500, demo_600, demo_700, demo_800) support change point annotators annotate real-world data sets for the `Turing Change Point Dataset` [38]. Nine synthetic time series are used for this research as the eighth `Annotated Change` demonstrative time series (demo_800) is multivate and split in two (800A and 800B) for this research to enable better comparison.

4.2 Downsampling

These nine synthetic time series are imported into `RStudio` from JSON scripts and two downsampling algorithms are applied to all with the`Jettison MVP Code` [39]. These simple algorithms are chosen to provide initial insights into the impacts of downsampling with minimally complex code; the algorithm *EveryNth* selects every other data point and *Percentage Change* selects every data point that has greater than one per cent difference between the last transmitted and newest values.

[SHOULD I ADD AN EQUATION HERE FOR THE ALGORITHMS?]

$$\xi_{ij}(t) = P(x_t = i, x_{t+1} = j | y, v, w; \theta) = \frac{\alpha_i(t) a_{ij}^{w_t} \beta_j(t+1) b_j^{v_{t+1}}(y_{t+1})}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i(t) a_{ij}^{w_t} \beta_j(t+1) b_j^{v_{t+1}}(y_{t+1})}$$

These algorithms are applied iteratively across each time series, creating parameters 1 to 50 for each downsampling algorithm and `Annotated Change` time series. There are now 900 time series for investigation; 450 for each downsampling algorithm.

The R package `imputeTS` [1] is applied to the time series to obtain the data volume, number of missing values (NAs) and number of gaps. This is important to understand the effect of each downsampling algorithm across the 900 time series. Data volume provides a common variable for comparison because the effect of each downsampling algorithm is unevenly distributed across the 50 parameters. For example, *EveryNth* discards data more quickly than *Percentage Change* so comparing the time series for each parameter is insufficient.

The missing values for each of the 900 time series are then replaced by the `imputeTS` linear interpolation function. This is a simple imputation algorithm that returns each time series to its original length, enabling a better comparison of the impact of downsampling. [DO I NEED TO EXPLAIN MORE THAN THIS?] For each `Annotated Change` synthetic time series there is now an imputed time series for parameters 1 to 50 and each downsampling method.

4.3 Features

The values for each catch22 time series feature [34] is calculated by the `Rcatch22` package [2]. This is applied to each of the nine original synthetic time series and 900 imputed time series. The difference between the original and imputed time series feature values is calculated and scaled to investigate the impact of downsampling on these catch22 features. The standard deviation across the time series for each catch22 feature is calculated to indicate whether some features are more sensitive to downsampling.

4.4 Visualisation

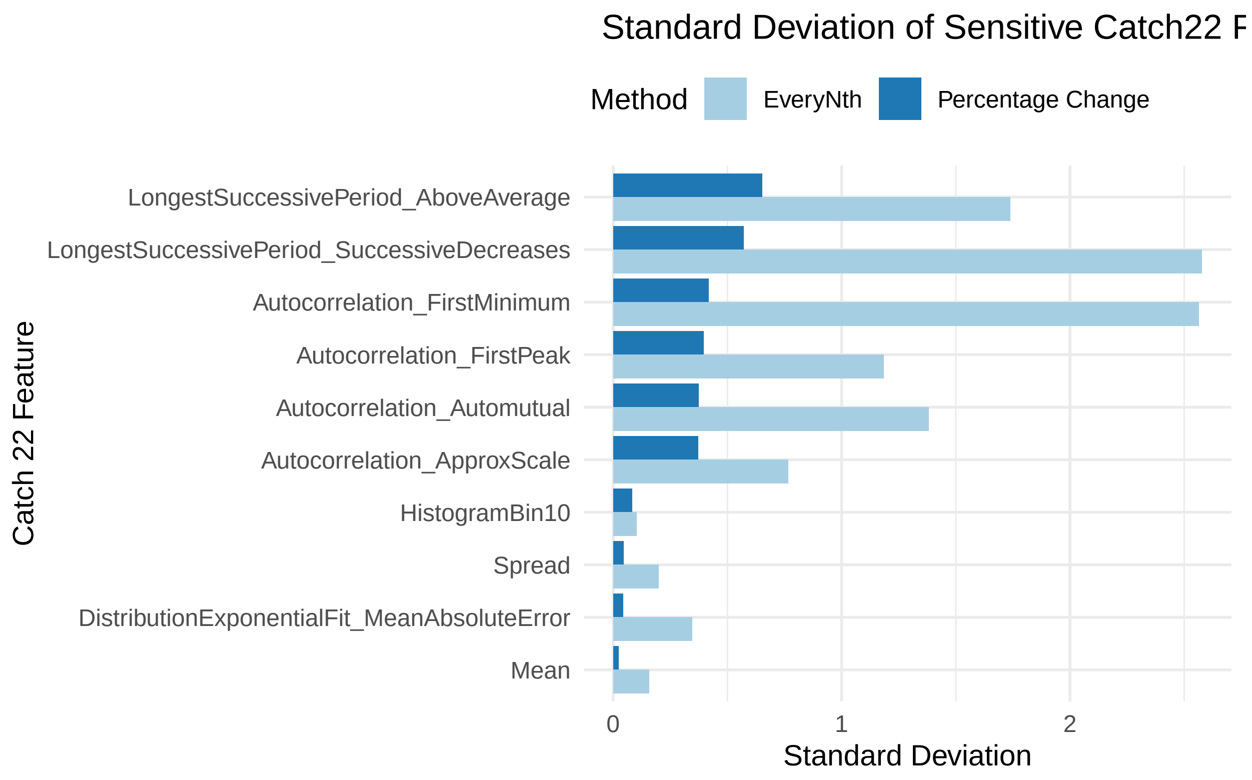
Combining `imputeTS` and `Rcatch22` in this way unstudied; initial visual analysis to observe the impact of downsampling on catch22 features is conducted utilising bar and line graphs alongside heatmaps. All the decision-makers who were interviewed as part of this research emphaised the importance of visualisation in data-driven decision-making and data transparency. In line with this, different approaches to visualising the impact of downsampling are tried to explore how best the impact of downsampling could be visually communicated by data practitioners for decision-makers.

5 RESULTS AND EVALUATION

This section provides an overview of results achieved from the research outlined in section IV; the limitations of these results and avenues for further investigation are highlighted to help contextualise these findings.

5.1 Downsampling Sensitivity

There appear to be seven catch22 features that deviate most across the 900 imputed time series. Due to their high deviation across the imputed time series, these seven features are considered to be more sensitive to the impacts of downsampling than the other 15. The standard deviation of all 22 features is shared in Annex A. To demonstrate the sensitivity of the seven features, the ten features with the highest standard deviation across all the imputed time series in the bar graph below.



This visualisation highlights the *EveryNth* algorithm appears to impact features more than the *Percentage Change* algorithm. Initial analysis suggests that this is likely to be because, in the 50 parameters created for this research, the *EveryNth* algorithm discards more data, more quickly. It would be beneficial to explore this dynamic further, creating parameters beyond 50 until the *Percentage Change* algorithm discards the same volume of data.

The standard deviation of the seven most sensitive catch22 features is presented in the table below. This table includes the standard deviation of these features for the *EveryNth* algorithm (‘everyNthSensitivity’), the *Percentage Change* algorithm (‘PercentageChangeSensitivity’), and the combined standard deviation (‘CombinedSensitivity’).

The combined sensitivity was used to identify the seven catch22 features that appear to be most sensitive. The table suggests that the first minimum of the autocorrelation function (‘Autocorr_FirstMinimum’ or ‘CO_FirstMin_ac’) is most sensitive across both of the algorithms; that the catch22 feature measuring the longest sequence of successive steps that decrease (‘LongestPe-

Table 2: Standard Deviation of the Seven Most Sensitive Catch22 Features

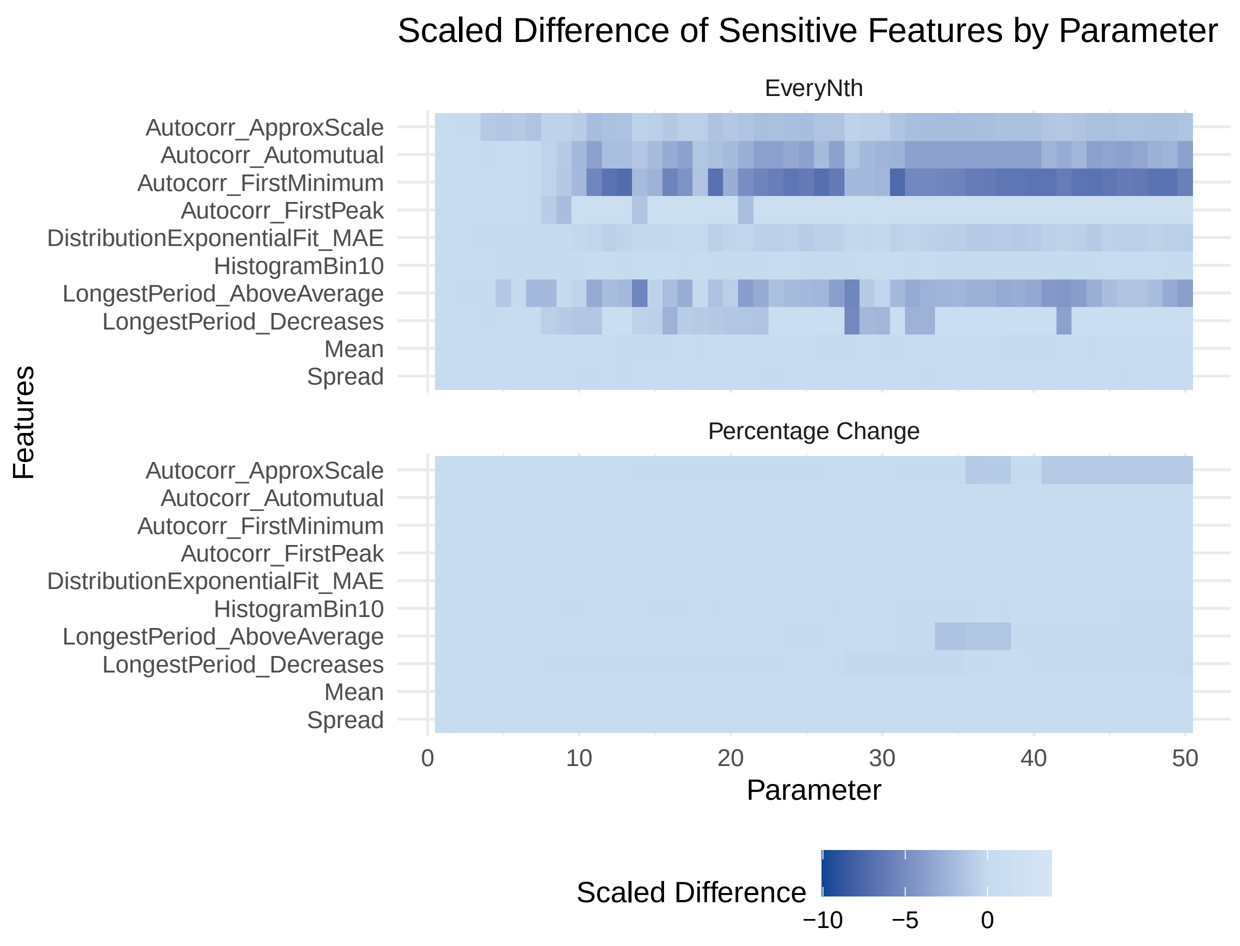
Feature	everyNthSensitivity	PercentageChangeSensitivity	CombinedSensitivity
Autocorr__FirstMinimum	2.5639294	0.4181318	2.5253400
LongestPeriod__Decreases	2.5778227	0.5711553	2.0901832
LongestPeriod__AboveAverage	1.7389139	0.6524714	1.7545755
Autocorr__Automutual	1.3808674	0.3736266	1.3890246
Autocorr__FirstPeak	1.1844657	0.3954080	0.9515096
Autocorr__ApproxScale	0.7661026	0.3713025	0.7172823
DistributionExponentialFit__MAE	0.3440829	0.0437543	0.3605538

riod__Decrease' or 'SB__BinaryStats__diff__longstretch0') is most sensitive to the *EveryNth* algorithm; and, the longest sequence of successive values greater than the mean ('LongestPeriod__AboveAverage' or 'SB__BinaryStats__mean__longstretch1') is most sensitive to the *Percentage Change* algorithm.

These results suggest that it may be possible to communicate the impact of downsampling through the algorithms imapct on catch22 features. However, standard deviation is a limited measure that suggests, rather than confirms sensitivity to the impacts of downsampling algorithms. To understand why these features are particularly sensitive, further investigation into the performance of the seven features on the 900 imputed time series is needed.

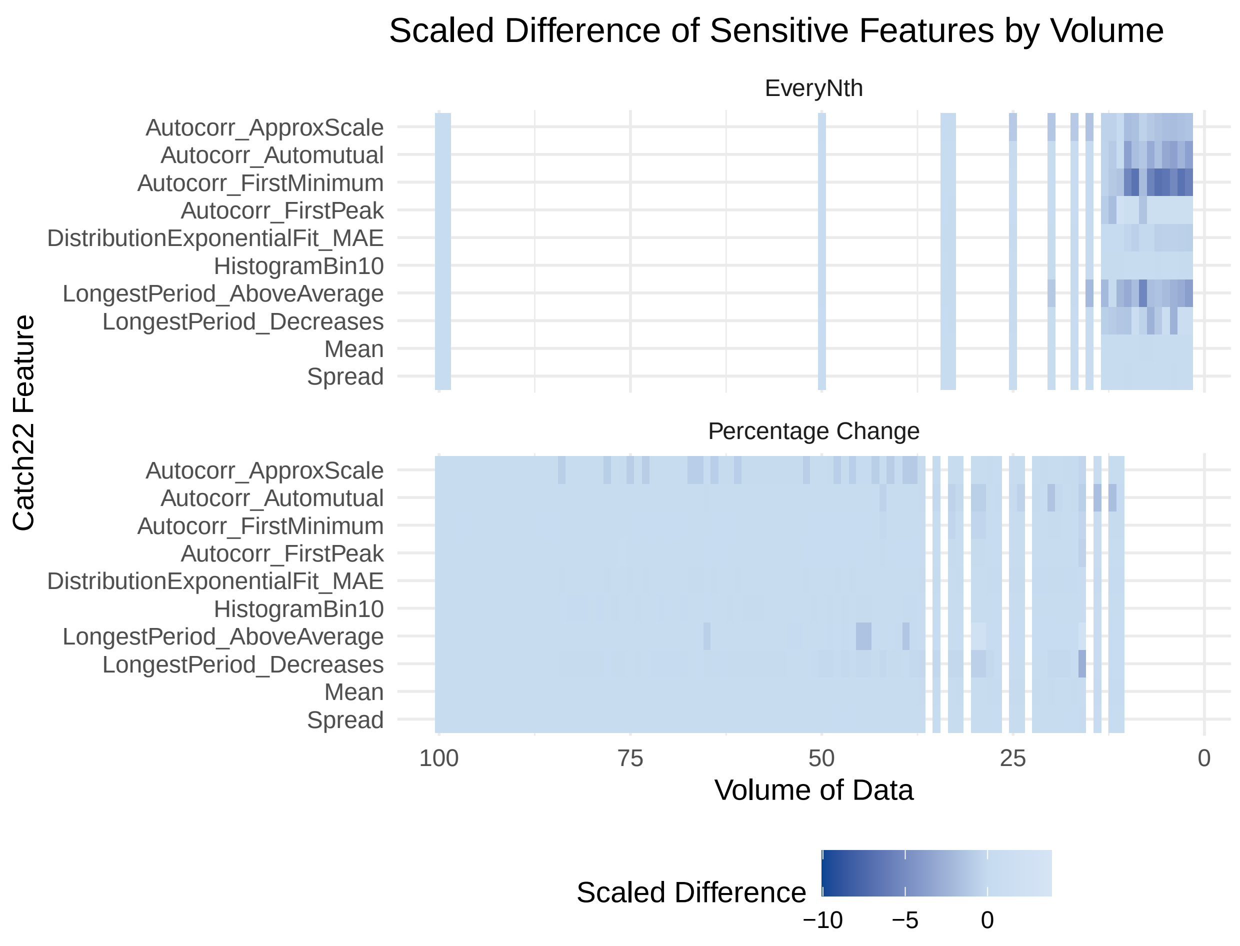
5.2 Downsampling Impact

Identifying that seven catch22 features apppear to be more sensitive to the impacts of downsampling will help communicate the impact of downsampling algorithms. The difference between the values of catch22 features for the nine original time series and 900 imputes time series is a measure of impact. The heapmap below visualises this difference, scaled, for the seven most senstive features across each of the 50 parameters.



This visualisation suggests that the approximate scale of autocorrelation ('Autocorr_ApproxScale' or 'CO_f1ecac') and the longest sequence of successive values greater than the mean ('LongestPeriod_AboveAverage' or 'SB_BinaryStats_mean_longstretch1') are the first catch22 features to be impacted by the downsampling algorithms. Interestingly, this visualisation also demonstrates that the catch22 features that are impacted by both downsampling algorithms tend to increase in comparison the original feature values. This dynamic warrants further investigation.

However, the parameters prevent a like-for-like comparison of the two downsampling algorithms as *EveryNth* discards data more quickly than *Percentage Change*. The heatmap below also visualises the scaled difference between the values of the original catch22 features and the imputed time series, but across the volume of data retained by the downsampling algorithms before `imputeTS` imputed the missing values.

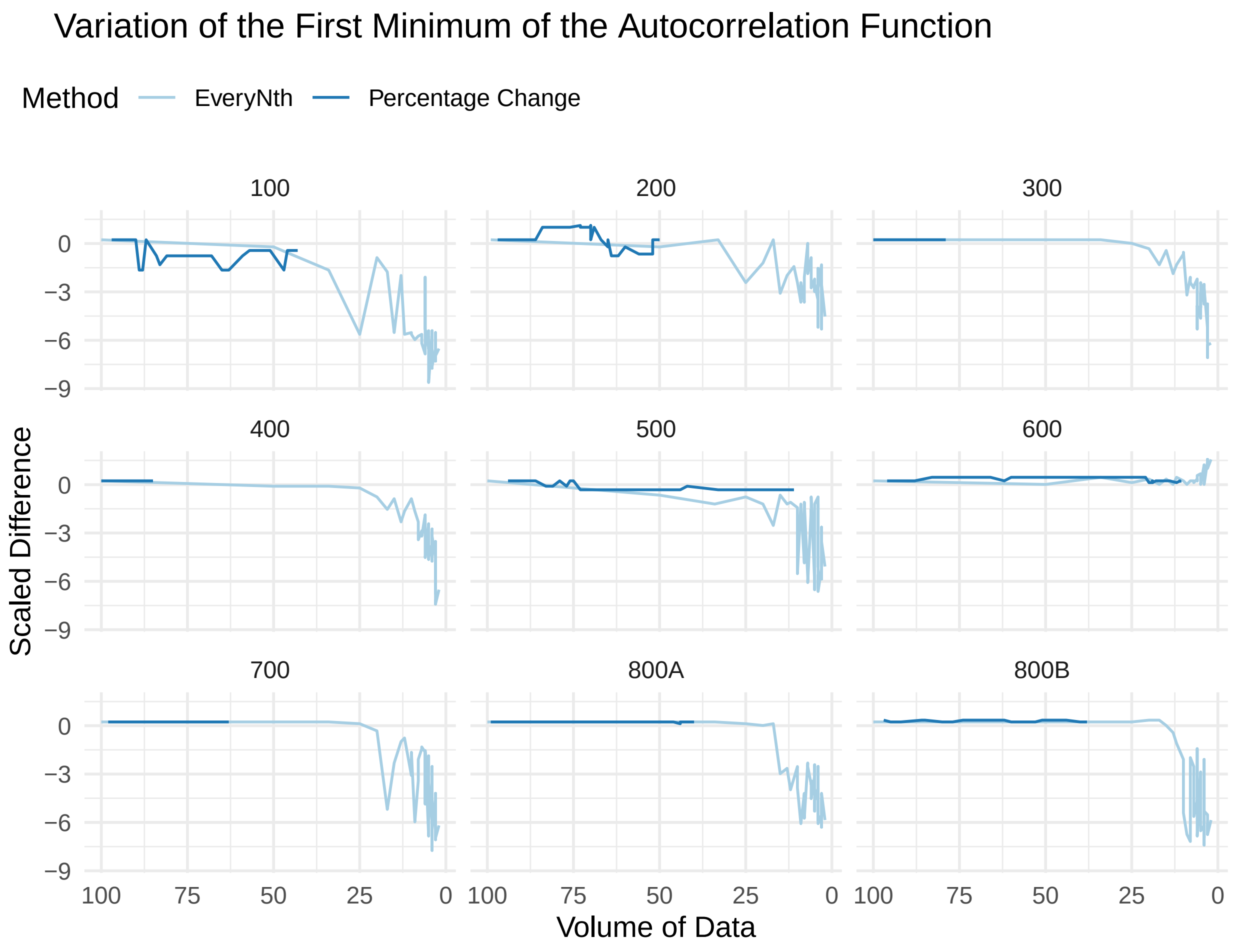


This visualisation demonstrates that data volume retained by the downsampling algorithms creates acute differences, especially when less than 20 data points remained after *EveryNth* is applied. Interestingly, the impact of *Percentage Change* appears to be inconsistent; there are some retained data volumes where `Autocorr_ApproxScale` and `LongestPeriod_AboveAverage` do not appear to be impacted by *Percentage Change* even though larger retained data volumes appear to be impacted. This dynamic warrants further investigation.

It is not possible, however, to understand the impact of both downsampling algorithms from this visualisation. The *Percentage Change* algorithm needs to be applied for more than 50 parameters so that it discards volumes of data comparable to the *EveryNth* algorithm. Despite this, the heatmap visualisation of downsampling impact holds potential; it suggests that, with the same volume of data, the downsampling algorithms impact the catch22 features differently.

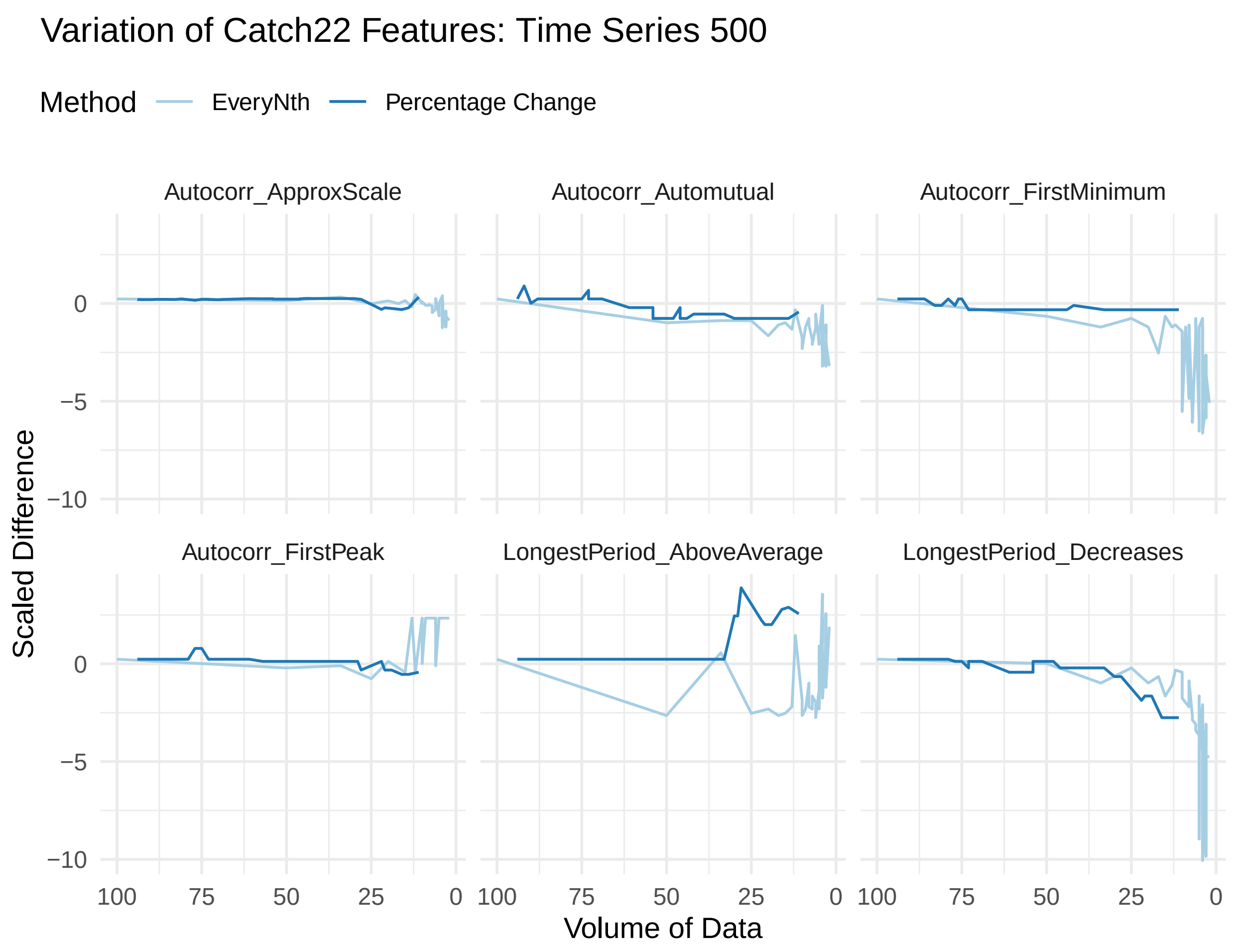
5.3 Feature Variation

The impact of both downsampling algorithms on the most sensitive features varies across different time series types. This is exemplified by the catch22 feature ‘Autocorr_FirstMinimum’ (‘CO_FirstMin_ac’): the first minimum of the autocorrelation function (‘Autocorr_FirstMinimum’ or ‘CO_FirstMin_ac’. This feature has the highest combined standard deviation across the 900 imputed time series represents “the number of steps into the future at which a value of the time series at the current point and that future point remain substantially (>1/e) correlated” **feature_book?**. The line graphs below highlight how this feature changes over the time series interpolated from different volumes of data after both downsampling algorithms are applied. The difference between the catch22 feature value of the original and imputed time series has been scaled for better comparisons.



The visualisations present the impact of discarding data for both downsampling algorithms; the volume of data retained by *Percentage Change* algorithm varies by the type of time series whereas the volume of data retained by the *EveryNth* algorithm trends towards zero.

This variation across time series types also suggests that different catch22 features may be more or less sensitive to downsampling for different time series types. This dynamic is invesitgated by the line graphs below, which plot the scaled difference of the six most sensitive catch22 features for time series '500'. Time series '500' was selected as more data is discarded by the *Percentage Change* algorithm, offering a better comparison.



This visualisation highlights that, although 'Autocorr_FirstMinimum' is identified as the most sensitive across both downsampling algorithms, 'Autocorr_Automutual' might better indicate the impact for time series '500'. 'Autocorr_Automutual' is the minimum of the automutual information function ('Autocorr_Automutual' or 'IN_AutoMutualInfoStats_40_gaussian_fmmi'); this catch22 feature outputs a measure of "autocorrelation in the time series, as the minimum of the automutual information function" **feature_book?**.

Variation of the most sensitive catch22 time series features is shared in Annex []. It would be beneficial to investigate this dynamic further as it may offer data practitioners method for identifying features that best indicate the preservation of a time series after downsampling is applied.

6 FUTURE WORK

The data pipeline developed by C. H. Lubba et. al could be used to generate other subsets of time series features for distinct tasks in any domain. This is likely to be important for highly specialised tasks or domains.

Highlight in text: - The values of catch22 features appear to increase as data is discarded. It would be beneficial to explore this dynamic further; - he *EveryNth* algorithm impacts more features than *Percentage Change*; this is likely to be because more data is discarded by the former algorithm than the latter in the 50 parameters explored by this research. It would be beneficial to explore this difference with further iterations of *Percentage Change* downsampling across the time series to understand its impact after more data is discarded.

7 CONCLUSION

8 REFERENCES

9 Annex A: Sensitivity of Catch22 Features

```r
# Reshape sensitivity subset data to a long format
subset_sensitivity_joined_long <- subset_sensitivity_joined %>%
  pivot_longer(cols = c(everyNthSensitivity, PercentageChangeSensitivity),
               names_to = "Method",
               values_to = "StandardDeviation")

# Change the Method into a factor and set the levels
subset_sensitivity_joined_long$Method <- factor(subset_sensitivity_joined_long$Method, levels = c("Perc

# Recode the names of the Methods
subset_sensitivity_joined_long$Method <- recode(subset_sensitivity_joined_long$Method,
                                                "PercentageChangeSensitivity" = "Percentage Change",
                                                "everyNthSensitivity" = "EveryNth")
# Recode the names of the Rcatch22 features
subset_sensitivity_joined_long$names <- recode(subset_sensitivity_joined_long$names,
                                               'CO_f1ecac' = "Autocorrelation_ApproxScale",
                                               'CO_FirstMin_ac' = "Autocorrelation_FirstMinimum",
                                               'SB_BinaryStats_mean_longstretch1' = "LongestSuccessivePe
                                               'PD_PeriodicityWang_th0_01' = "Autocorrelation_FirstPeak
                                               'DN_Mean' = 'Mean',
                                               'DN_HistogramMode_10' = "HistogramBin10",
                                               'DN_Spread_Std' = 'Spead',
                                               'CO_Embed2_Dist_tau_d_expfit_meandiff' = "DistributionExp
                                               'IN_AutoMutualInfoStats_40_gaussian_fmmi' = "Autocorrela
                                               'SB_BinaryStats_diff_longstretch0' = "LongestSuccessivePe

# Create a bar plot
sensitive_subset_plot <- ggplot(subset_sensitivity_joined_long, aes(x = StandardDeviation, y = names, fi
  geom_bar(stat = "identity", position = position_dodge()) +
  theme_minimal() +
  theme(text = element_text(size = 30), legend.position = "top") +
  labs(x = "Standard Deviation", y = "Catch22 Feature", fill = "Method", title = "Standard Deviation of
  scale_fill_brewer(palette = "Paired") +
  scale_x_continuous(limits = c(0, NA))
```

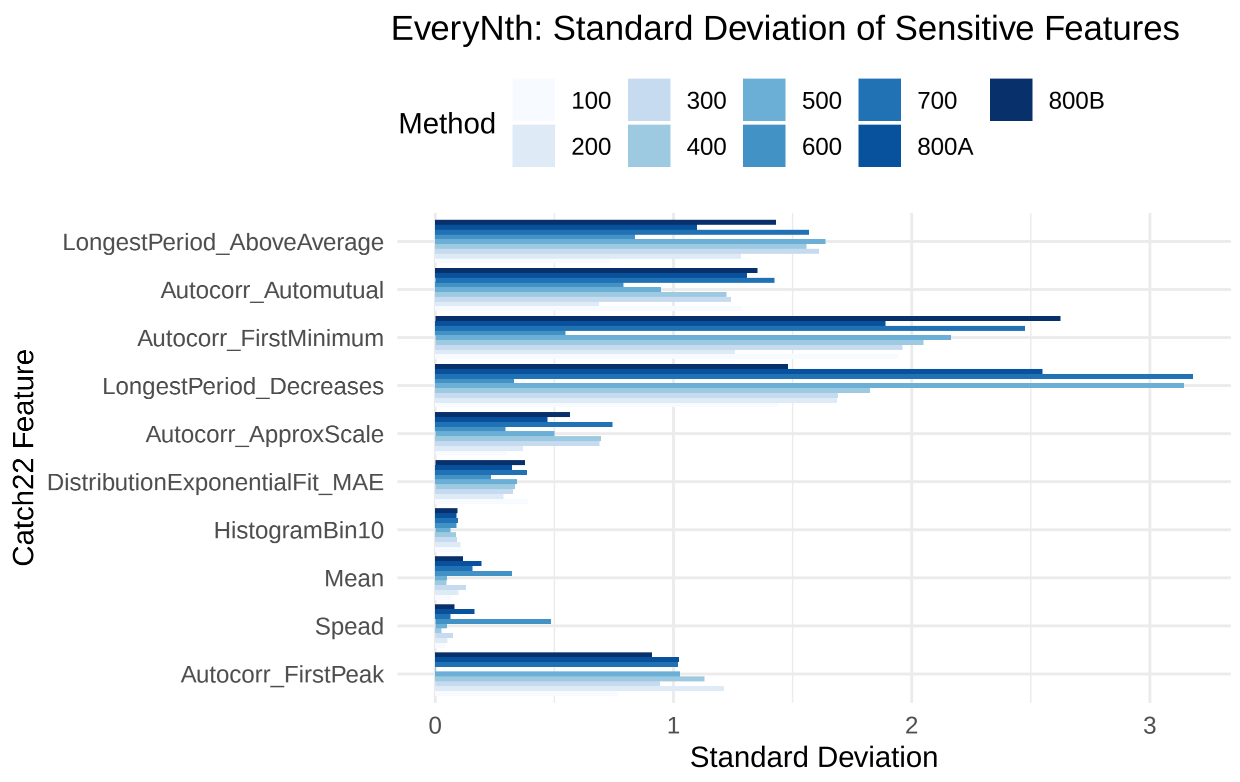
```r
knitr::kable((sensitivity_joined), caption = "Sensitivity of Catch22 Features to EveryNth and Percentag
```

10 Annex B: Top Ten Most Sensitive Catch22 Features

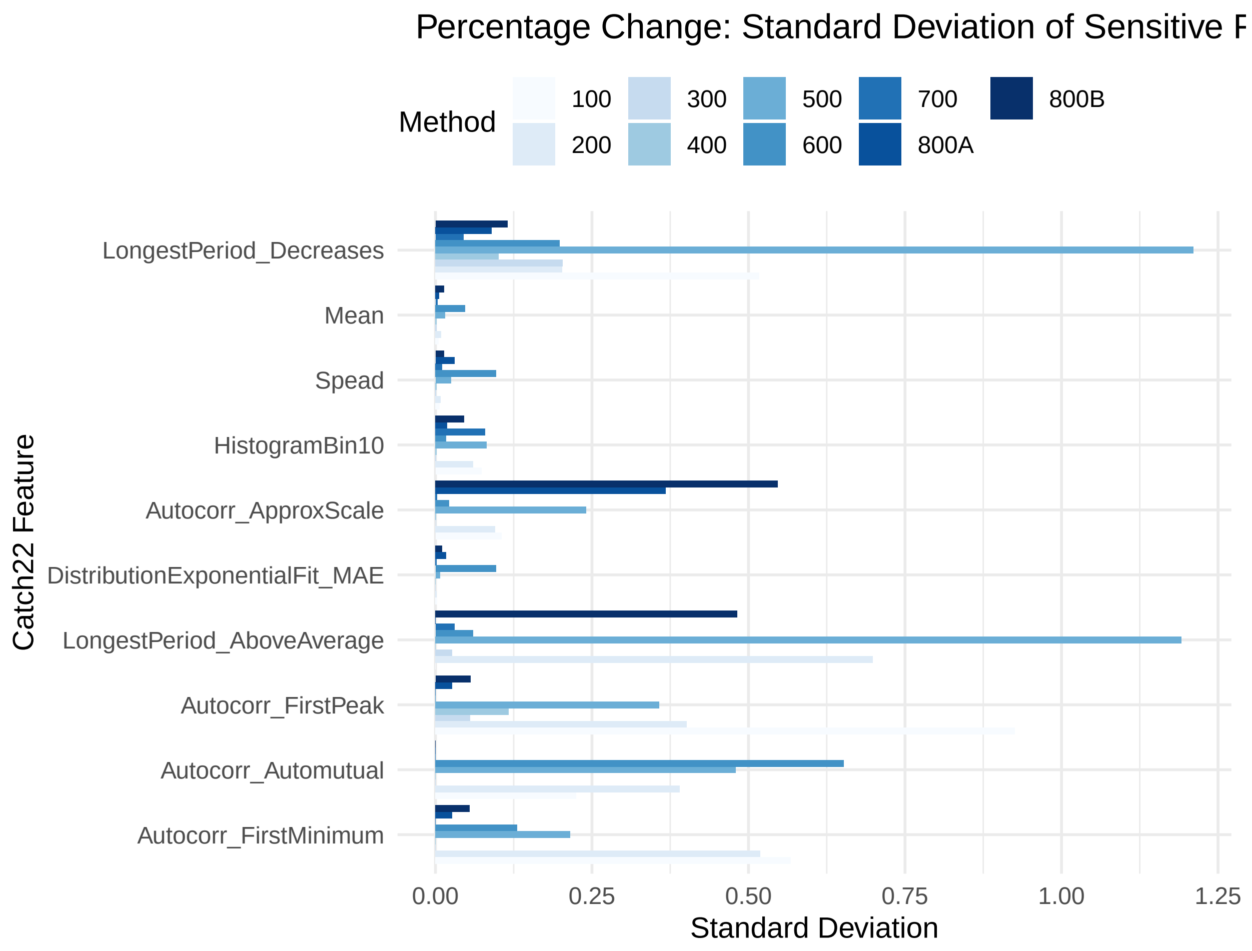
The ten features most impacted by both downsampling algorithms are selected for further investigation. The bar graph below visualises the standard deviation of the *EveryNth* algorithm for these features across all the parameters for each of the nine synthetic time series.

Table 3: Sensitivity of Catch22 Features to EveryNth and Percentage Change Downsampling

names	everyNthSensitivity	PercentageChangeSensitivity
SB_BinaryStats_diff_longstretch0	2.5778227	0.5711553
CO_FirstMin_ac	2.5639294	0.4181318
SB_BinaryStats_mean_longstretch1	1.7389139	0.6524714
IN_AutoMutualInfoStats_40_gaussian_fmmi	1.3808674	0.3736266
PD_PeriodicityWang_th0_01	1.1844657	0.3954080
CO_f1ecac	0.7661026	0.3713025
CO_Embed2_Dist_tau_d_expfit_meandiff	0.3440829	0.0437543
DN_Spread_Std	0.1985249	0.0455823
DN_Mean	0.1572628	0.0227554
DN_HistogramMode_10	0.1013119	0.0822124
DN_HistogramMode_5	0.0869504	0.0647597
SP_Summaries_welch_rect_centroid	0.0654187	0.0025698
FC_LocalSimple_mean1_tauresrat	0.0436216	0.0218936
DN_OutlierInclude_n_001_mdrmd	0.0434259	0.0050690
SC_FluctAnal_2_rsrangefit_50_1_logi_prop_r1	0.0432552	0.0277534
CO_trev_1_num	0.0425513	0.0130514
FC_LocalSimple_mean3_stderr	0.0384460	0.0049473
DN_OutlierInclude_p_001_mdrmd	0.0349796	0.0138568
SB_MotifThree_quantile_hh	0.0340103	0.0139764
SP_Summaries_welch_rect_area_5_1	0.0326959	0.0038222
CO_HistogramAMI_even_2_5	0.0315734	0.0134047
MD_hrv_classic_pnn40	0.0241202	0.0102366
SC_FluctAnal_2_dfa_50_1_2_logi_prop_r1	0.0224893	0.0087991
SB_TransitionMatrix_3ac_sumdiagcov	0.0053928	0.0037061



The visualisation above highlights that the impact of downsampling on each catch22 feature is different for each time series type. This holds true for the *Percentage Change* algorithm too. The bar graph below visualises the standard deviation for the same ten features across all the parameters for each of the nine synthetic time series.



Interestingly, the impact of *Percentage Change* on many of these features is acute despite the spread of standard deviation being smaller. For example... [ADD OBSERVATION]

11 Annex C: Data Volume vs. Downsampling Parameters

12 Annex D: Feature Variation

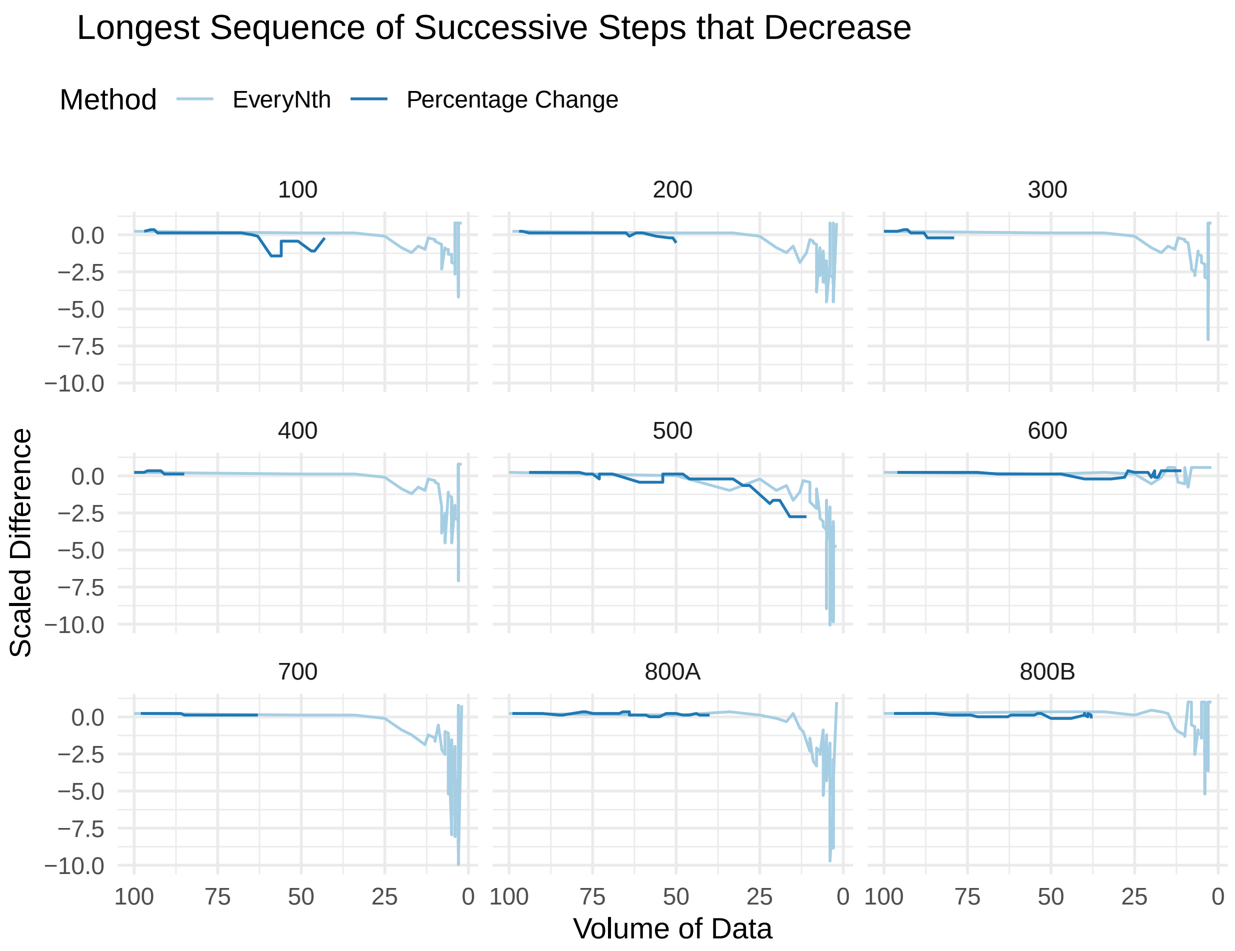
Line graphs of the original time series are provided in Annex D to support comparison. The original time series '100' and '200' are similarly shaped, with '100' trending upwards and '200' experiencing a noticeable drop towards the end of the time series. In the line graphs below, it is observable that '100' and '200' are similarly impacted by *EveryNth* and *Percentage Change*, respectively. The impact of *Percentage Change* is more acute and variable than for the other time series.

The original time series '300' and '400' are similarly shaped, with significant peaks and troughs that start earlier for '400'. This is likely to account for why *Percentage Change* retains more data, causing a negible impact on the catch22 feature for these time series. The time series '500' and '600' are particularly distinct with some obvious outliers in '500' and a gradual trend upwards for '600'. More data is discarded by the *Percentage Change* algorithm for these time series than any of the others. Interestingly, '600' appears to be the only time series for which the scaled difference between the original and imputed catch22 feature values trends positively.

The remaining original time series '700', '800A' and '800B' are distinct, but have similar slow-varying oscillation. This is mirrored by the impacts of *EveryNth* and *Percentage Change*. The scaled difference of the catch22 feature for the *EveryNth* algorithm in '700' has the widest spread of three and *Percentage Change* appears to retain more data than in '800A' and '800B'. Again, this is likley to be because the changes in the peaks and troughs are more acute in the original data.

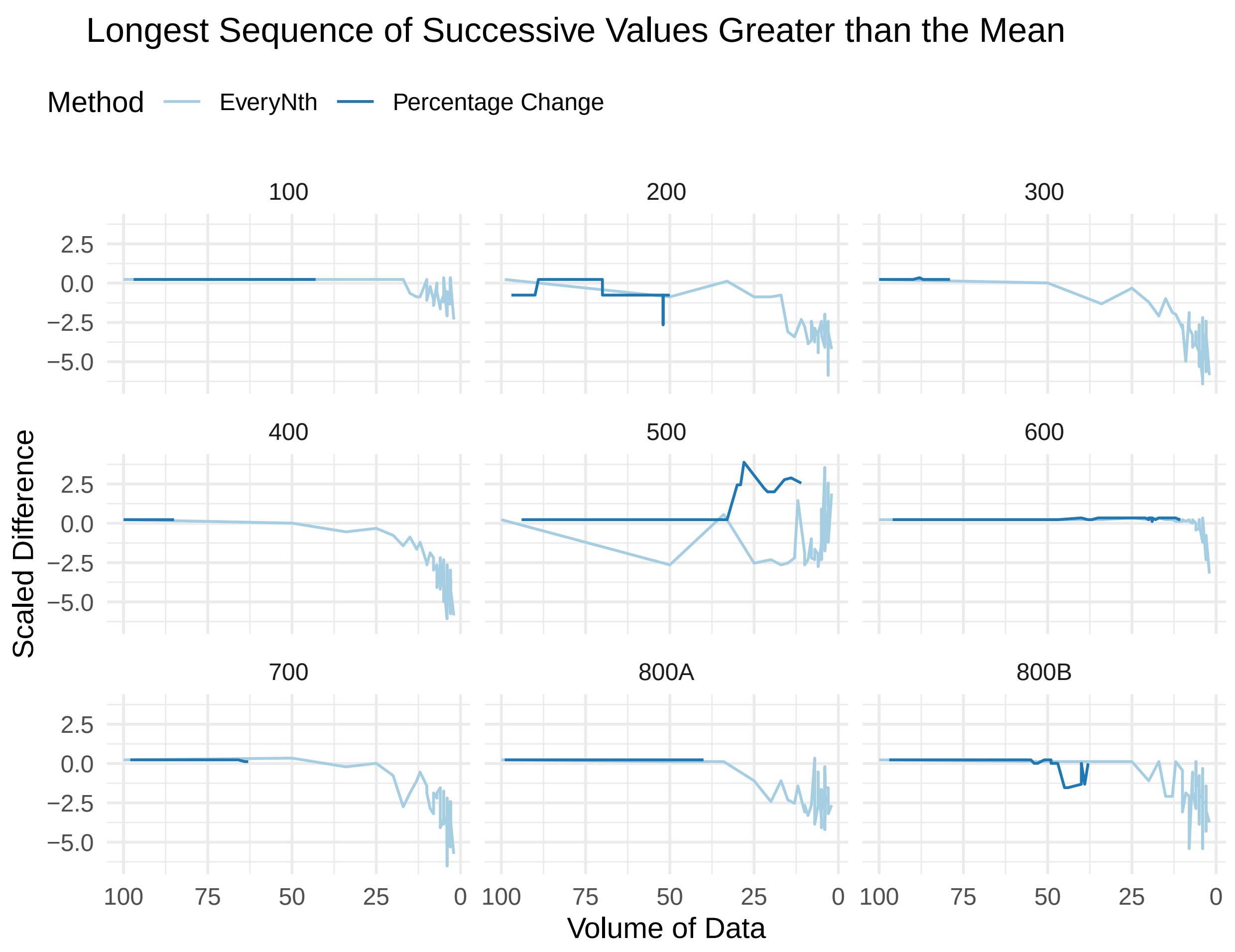
5.2b Longest Sequence of Successive Steps that Decrease

The feature with the second highest standard deviation across the 900 time series is the longest sequence of successive steps that decrease ('LongestPeriod_Decrease' or 'SB_BinaryStats_diff_longstretch0'). This catch22 feature "calculates the longest sequence of successive steps in the time series that decrease" **feature_book?**.



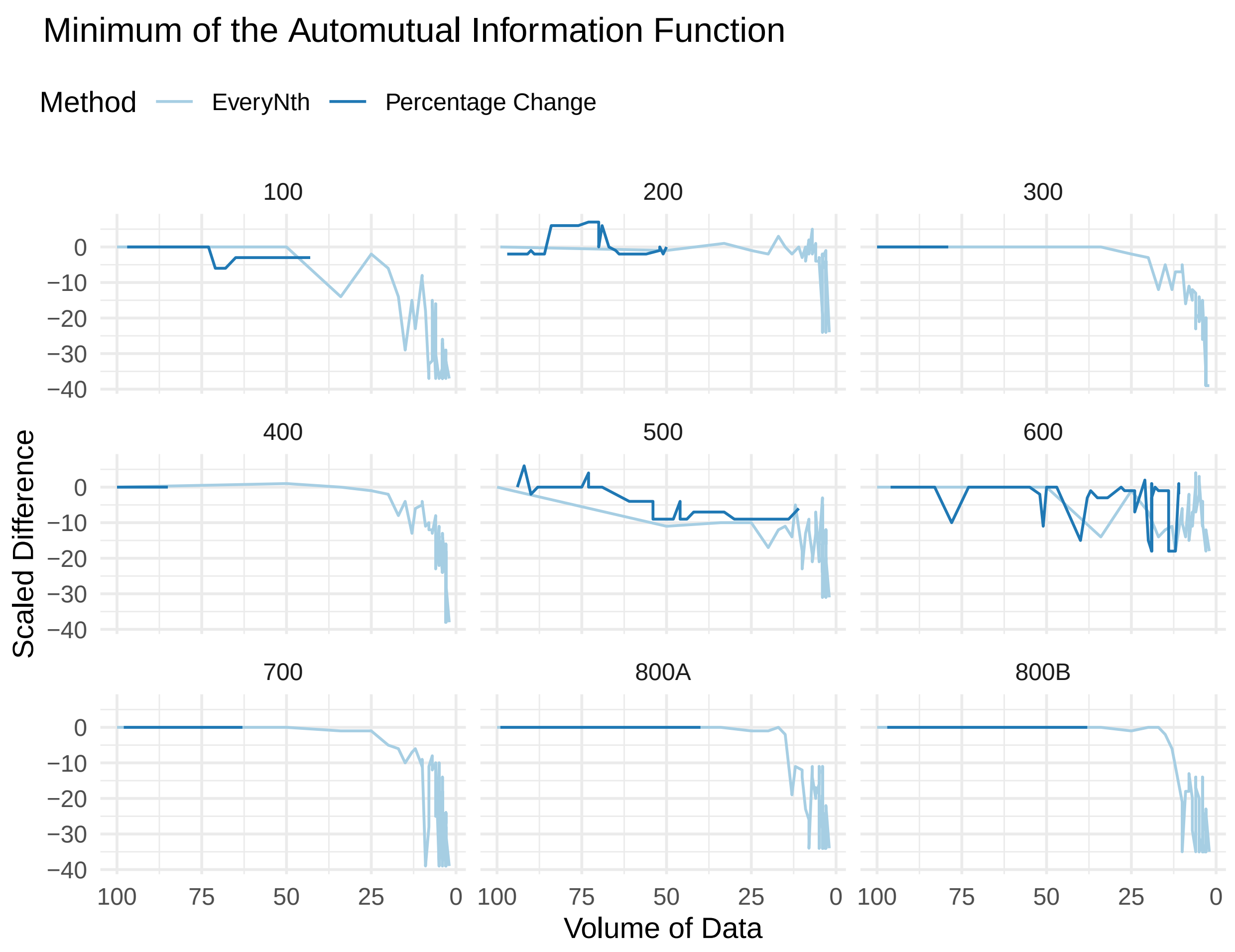
5.2c Longest Sequence of Successive Values Greater than the Mean

The feature with the third highest standard deviation across the 900 time series is the longest sequence of successive values greater than the mean ('LongestPeriod_AboveAverage' or 'SB_BinaryStats_mean_longstretch1'). This catch22 feature "calculates the longest successive period of above average values" **feature_book?**.



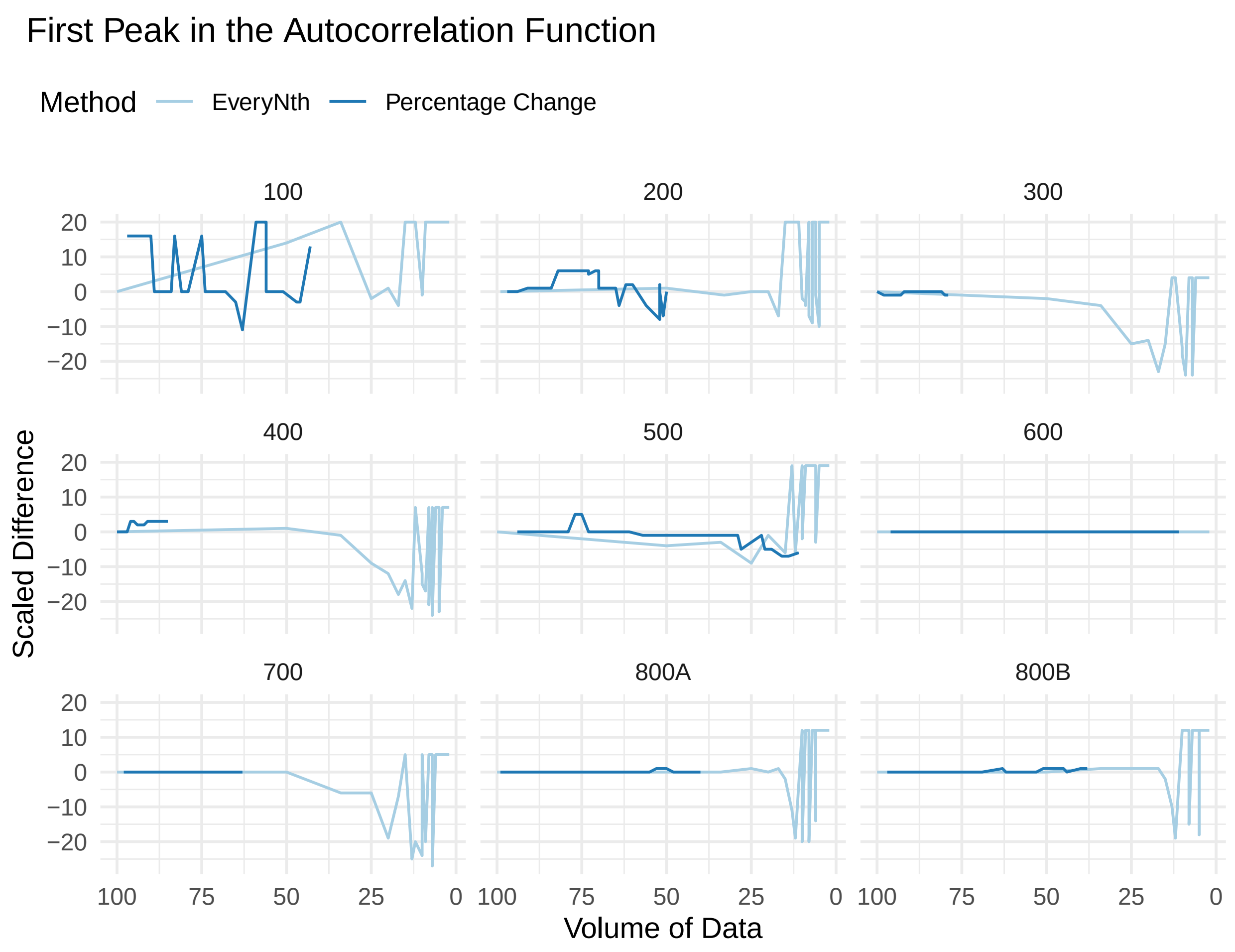
5.2d Minimum of the Automutual Information Function

The feature with the fourth highest standard deviation across the 900 time series is the minimum of the automutual information function ('Autocorr_Automutual' or 'IN_AutoMutualInfoStats_40_gaussian_fmmi'). This catch22 feature outputs a measure of "autocorrelation in the time series, as the minimum of the automutual information function" **feature_book?**.



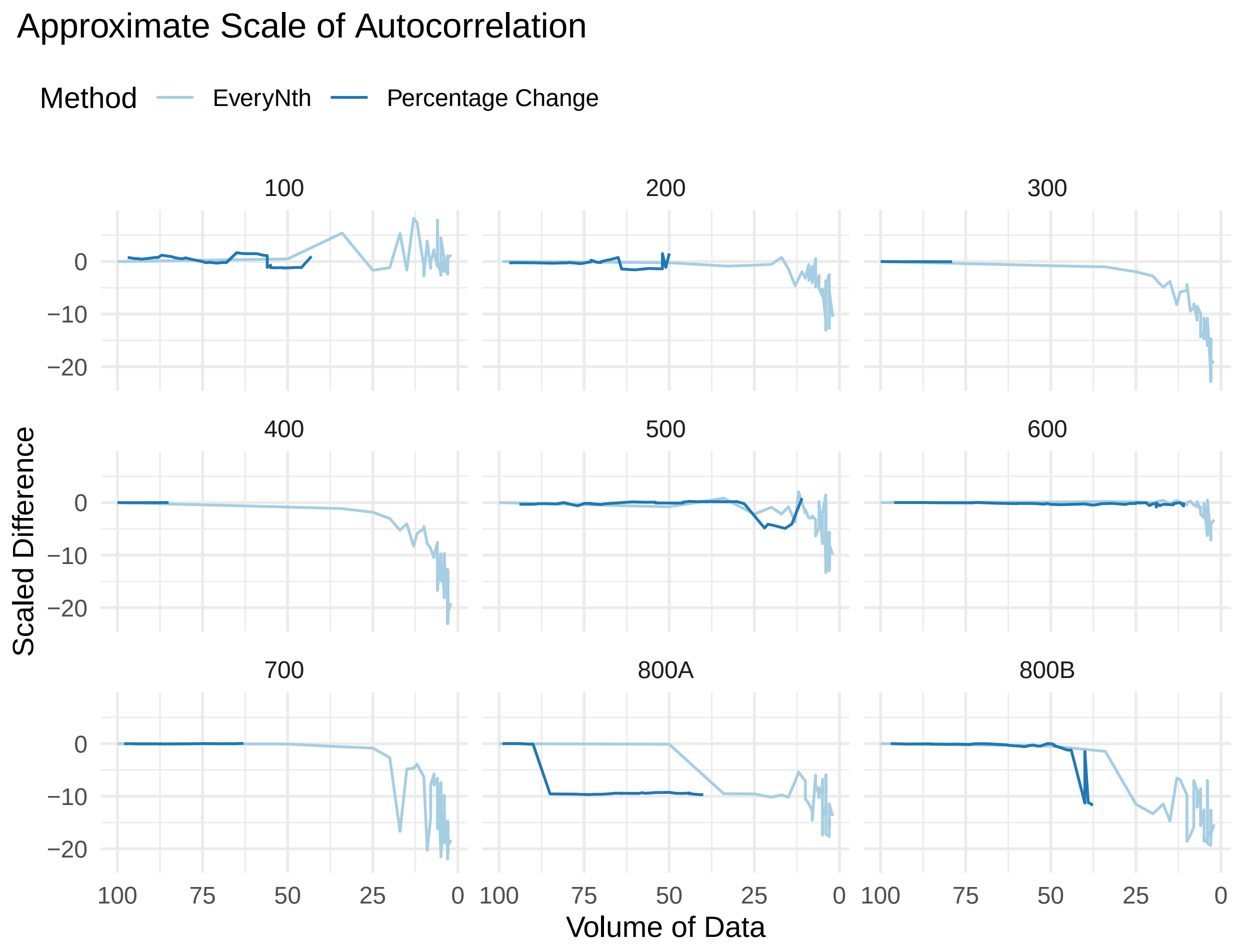
5.2e First Peak in the Autocorrelation Function

The feature with the fifth highest standard deviation across the 900 time series is the first peak in the autocorrelation function ('Autocorr_FirstPeak' or 'PD_PeriodicityWang_th0_01'). This catch22 feature "returns the first peak in the autocorrelation function satisfying a set of conditions (after detrending the time series using a single-knot cubic regression spline)" **feature_book?**. In general, the feature returns higher values for slower time series oscillation.



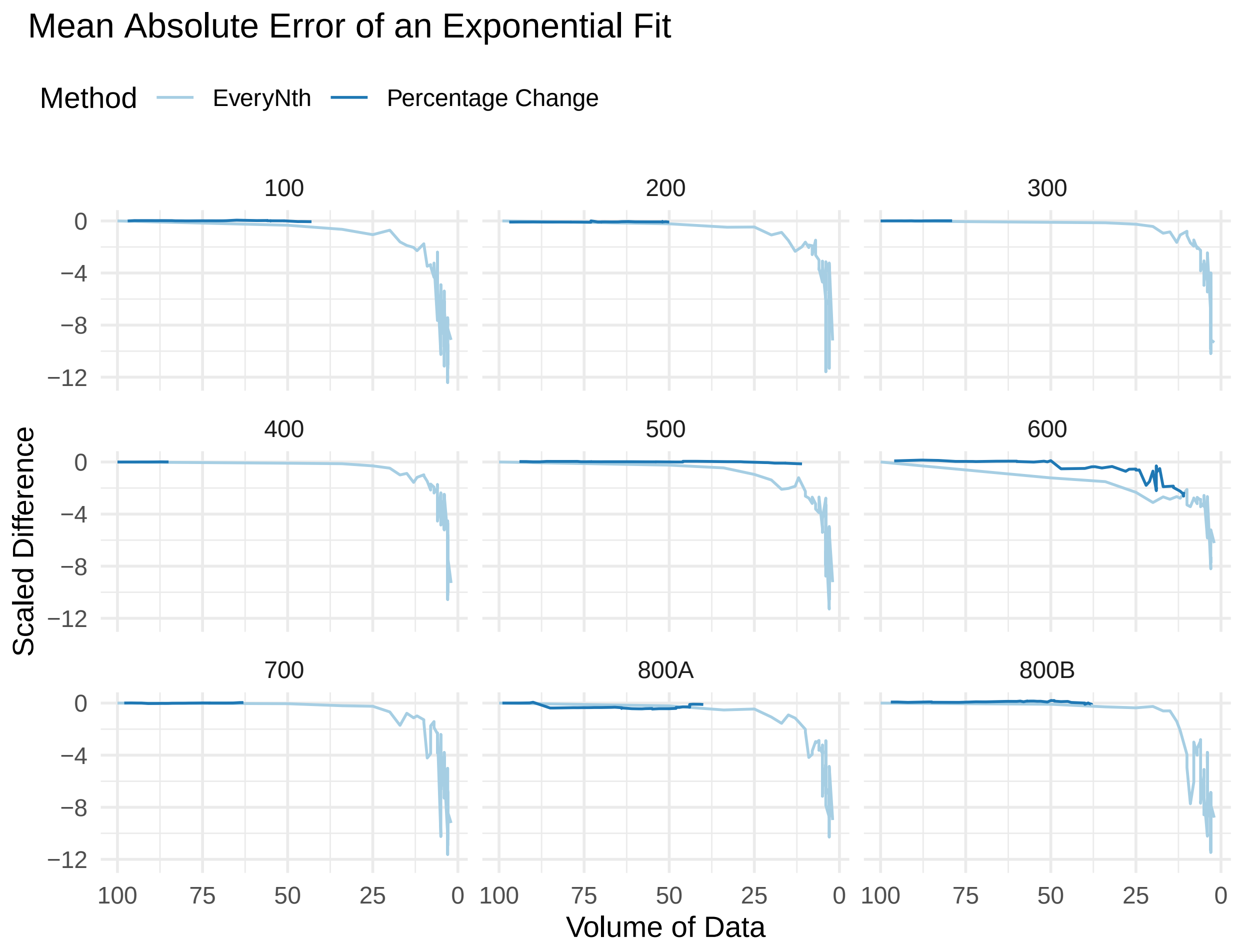
5.2f Approximate Scale of Autocorrelation

The feature with the sixth highest standard deviation across the 900 time series is the approximate scale of autocorrelation ('Autocorr_ApproxScale' or 'CO_f1ecac'). This catch22 feature "computes the first 1/e crossing of the autocorrelation function of the time series" **feature_book?**. It is similar to the first minimum of the autocorrelation function.



5.2g Mean Absolute Error of an Exponential Fit

The feature with the seventh highest standard deviation across the 900 time series is the approximate scale of autocorrelation (‘DistributionExponentialFit_MAE’ or ‘CO_Embed2_Dist_tau_d_expfit_meandiff’). This catch22 feature outputs the mean absolute error of an exponential fit to a distribution which is calculated by representing “the time series in a two-dimensional time-delay embedding space (using a time delay equal to the first zero-crossing of the autocorrelation function)… [then computing the] successive distances between points in this 2D embedding space and analyzes the probability distribution of these distances” **feature_book?**.



[1] S. Moritz and T. Bartiz-Beielstein, “imputeTS: Time series missing value imputation in r,” vol. 9.1. R Journal, 2017. doi: 10.32614/RJ-2017-009.

[2] C. H. Lubba, B. Fulcher, T. Henderspn, B. Harris, O. r. TL, and O. Cliff, “catch22: CAnonical time-series CHaracteristics.” R Journal, 2022. doi: 10.5281/zenodo.6673597.

[3] Cabinet Office and Government Digital Service, “Government transformation strategy: Better use of data.” HM Government; `https://www.gov.uk/government/publications/government-transformation-strategy-2017-to-2020/government-transformation-strategy-better-use-of-data`, 2017.

[4] Department for Digital, Culture, Media & Sport and Department for Science, Innovation & Technology, “National data strategy.” HM Government; `https://www.gov.uk/government/publications/uk-national-data-strategy/national-data-strategy`, 2020.

[5] M. of Defence, “Data strategy for defence,” *GOV.UK*. HM Government; `https://www.gov.uk/government/publications/data-strategy-for-defence/data-strategy-for-defence`, 2021.

[6] Central Digital & Data Office, “Transforming for a digital future: 2022 to 2025 roadmap for digital and data.” HM Government; `https://www.gov.uk/government/publications/roadmap-for-digital-and-data-2022-to-2025/transforming-for-a-digital-future-2022-to-2025-roadmap-for-digital-and-data`, 2022.

[7] Centre for Data Ethics & Innovation, “Addressing trust in public sector data use.” `https://www.gov.uk/government/publications/cdei-publishes-its-first-report-on-public-sector-data-sharing/addressing-trust-in-public-sector-data-use#introduction--context`.

[8] Government Analysis Function, "Types of data in government learning pathway." `https://analysisfunction.civilservice.gov.uk/learning-development/learning-pathways/types-of-data-in-government-learning-pathway/`, 2022.

[9] Office for National Statistics, "Time series explorer." `https://www.ons.gov.uk/timeseriestool?query=&topic=&updated=&fromDateDay=&fromDateMonth=&fromDateYear=&toDateDay=&toDateMonth=&toDateYear=&size=50`, Unknown.

[10] Y. An, Y. Su, Y. Zhu, and J. Wang, "TVStore: Automatically bounding time series storage via time-varying compression," in *Proceedings of the 20th USENIX conference on file and storage technologies*, in USENIX conference on file and STorage technologies. Santa Clara, CA, USA: USENIX Association, 2022, pp. 83–99.

[11] J. Donckt, J. Donckt, M. Rademaker, and S. Hoecke, "Data point selection for line chart visualization: Methodological assessment and evidence-based guidelines." 2023. doi: 10.48550/arXiv.2304.00900.

[12] S. Steinarsson, "Downsampling time series for visual representation." University of Iceland, Faculty of Industrial Engineering, Mechanical Engineering; Computer Science, School of Engineering; Natural Sciences, University of Iceland, Reykjavik, Iceland, 2013.

[13] The Shift Project, "Implementing digital sufficiency," 2020.

[14] W. Aigner, S. Miksch, W. Muller, H. Schumann, and C. Tominski, "Visual methods for analyzing time-oriented data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 1, pp. 47–60, 2008, doi: 10.1109/TVCG.2007.70415.

[15] B. C. Kwon, J. Verma, P. J. Haas, and C. Demiralp, "Sampling for scalable visual analytics," *IEEE Computer Graphics and Applications*, vol. 37, no. 1, pp. 100–108, 2017, doi: 10.1109/MCG.2017.6.

[16] J. Donckt, J. Donckt, M. Rademaker, and S. Hoecke, "MinMaxLTTB: Leveraging MinMax-preselection to scale LTTB." 2023. Available: `https://arxiv.org/abs/2305.00332`

[17] K. E. Levy and D. M. Johns, "When open data is a trojan horse: The weaponization of transparency in science and governance," *Big Data & Society*, vol. 3, no. 1, 2016, doi: 10.1177/2053951715621568.

[18] J. Bates, H. Kennedy, I. Medina Perea, S. Oman, and L. Pinney, "Socially meaningful transparency in data-based systems: Reflections and proposals from practice," *Journal of Documentation*, vol. ahead–of–print, 2023, doi: 10.1108/JD-01-2023-0006.

[19] M. Ananny and K. Crawford, "Seeing without knowing: Limitations of the transparency ideal and its application to algorithmic accountability," *New Media & Society*, vol. 20, no. 3, pp. 973–989, 2018, doi: 10.1177/1461444816676645.

[20] R. Matheus, M. Janssen, and T. Janowski, "Design principles for creating digital transparency in government," *Government Information Quarterly*, vol. 38, no. 1, 2021, doi: `https://doi.org/10.1016/j.giq.2020.101550`.

[21] N. A. Draper and J. Turow, "The corporate cultivation of digital resignation," *New Media & Society*, vol. 21, no. 8, pp. 1824–1839, 2019, doi: 10.1177/1461444819833331.

[22] J. A. Obar, "Sunlight alone is not a disinfectant: Consent and the futility of opening big data black boxes (without assistance)," *Big Data & Society*, vol. 7, no. 1, 2020, doi: 10.1177/2053951720935615.

[23] J. Walker, R. Borgo, and MW. Jones, "TimeNotes: A study on effective chart visualization and interaction techniques for time-series data," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, 2016, doi: 10.1109/TVCG.2015.2467751.

[24] J. Donckt, J. Donckt, E. Deprost, and S. Hoecke, "Plotly-resampler: Effective visual analytics for large time series," *IEEE Visualization and Visual Analytics*, 2022, doi: 10.1109/VIS54862.2022.00013.

[25] S. Moritz and T. Bartiz-Beielstein, "imputeTS: Time series missing value imputation in r." R Journal, 2017. Available: `https://cran.r-project.org/web/packages/imputeTS/vignettes/imputeTS-Time-Series-Missing-Value-Imputation-in-R.pdf`

[26] A. Visheratin *et al.*, "Peregreen – modular database for efficient storage of historical time series in cloud environments," in *2020 USENIX annual technical conference (USENIX ATC 20)*, USENIX Association, 2020, pp. 589–601. Available: `https://www.usenix.org/conference/atc20/presentation/visheratin`

[27] T. Schlossnagle, J. Sheehy, and C. McCubbin, "Always-on time-series database: Keeping up where there's no way to catch up," *Commun. ACM*, vol. 64, no. 7, pp. 50–56, 2021, Available: `https://doi.org/10.1145/3442518`

[28] HM Government, "Global britain in a competitive age: The integrated review of security, defence, development and foreign policy." GOV.UK, 2021. Available: `https://www.gov.uk/government/publications/global-britain-in-a-competitive-age-the-integrated-review-of-security-defence-development-and-fo` `global-britain-in-a-competitive-age-the-integrated-review-of-security-defence-development-and-fo`

[29] I. Foster *et al.*, "Computing just what you need: Online data analysis and reduction at extreme scales," in *Euro-par 2017*, F. F. Rivera, T. F. Pena, and J. C. Cabaleiro, Eds., in Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). Germany: Springer Verlag, 2017, pp. 3–19. doi: 10.1007/978-3-319-64203-1_1.

[30] A. Kohn, D. Moritz, and T. Neumann, "DashQL – complete analysis workflows with SQL." 2023. doi: 10.48550/arXiv.2306.03714.

[31] U. Jugel, Z. Jerzak, G. Hackenbroic, and V. Markl, "VDDA: Automatic visualization-driven data aggregation in relational databases," *The VLDB Journal*, vol. 25, 2016, doi: 10.1007/s00778-015-0396-z.

[32] W. Yunhai *et al.*, "OM3: An ordered multi-level min-max representation for interactive progressive visualization of time series," in *Proc. ACM manag. data*, ACM, 2023. Available: `https://doi.org/10.1145/3589290`

[33] U. Jugel, Z. Jerzak, G. Hackenbroich, and V. Markl, "M4: A visualization-oriented time series data aggregation. Proceedings of the VLDB endowment," vol. 7, 2014, Available: `https://www.vldb.org/2014/program/http://www.vldb.org/pvldb/vol7/p797-jugel.pdf`

[34] C. H. Lubba, S. S. Sarab, P. Knaute, S. R. Schultz, B. D. Fulcher, and N. S. Jones, "catch22: CAnonical time-series CHaracteristics," *Data Mining and Knowledge Discovery*, vol. 33, 2019, doi: 10.1007/s10618-019-00647-x.

[35] B. D. Fulcher and N. S. Jones, "Highly comparative feature-based time-series classification," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 12, pp. 3026–3037, 2014, doi: 10.1109/TKDE.2014.2316504.

[36] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh, "The great time series classification bake off: A review and experimental evaluation of recent algorithmic advances," *Data Mining and Knowledge Discovery*, vol. 31, 2017, doi: 10.1007/s10618-016-0483-9.

[37] T. Henderson and B. D. Fulcher, "An empirical evaluation of time-series feature sets," in *2021 international conference on data mining workshops (ICDMW)*, 2021, pp. 1032–1038. doi: 10.1109/ICDMW53433.2021.00134.

[38] G. J. J. van den Burg and C. K. I. Williams, "An evaluation of change point detection algorithms." 2022. doi: 10.48550/arXiv.2003.06222.

[39] M. F. et. al, "Jettison MVP code." 2023. Available: `https://github.com/MattForshaw/Jettison/tree/main/MVPCode`