TopoLines: Topological Smoothing for Line Charts

P. Rosen[1] and A. Suh[1,2] and C. Salgado[1] and M. Hajij[3]

[1]University of South Florida, Tampa FL, USA
[2]Tufts University, Medford, MA, USA
[3]KLA Tencor, Ann Arbor, MI, USA

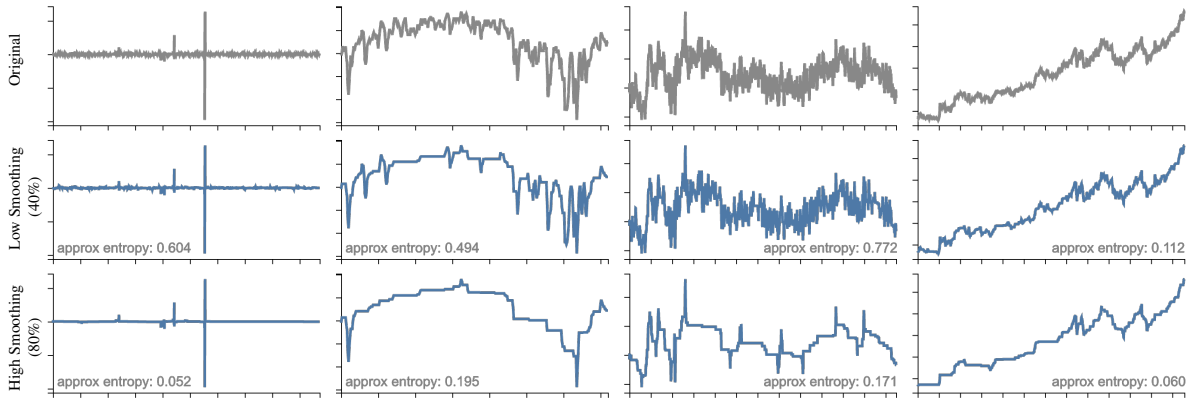


Figure 1: *Examples of the (top) 4 input datasets (see section 4 for a description of the data). TopoLines results are shown for (middle) low and (bottom) high levels of smoothing, defined by the percent of local extrema removed. TopoLines works by preserving high amplitude extrema and flattening low amplitude ones while maintaining low residual error. See the supplement for the measures described in section 3.*

Abstract

Line charts are commonly used to visualize a series of data values. When the data are noisy, smoothing is applied to make the signal more apparent. Conventional methods used to smooth line charts, e.g., using subsampling or filters, such as median, Gaussian, or low-pass, each optimize for different properties of the data. The properties generally do not include retaining peaks (i.e., local minima and maxima) in the data, which is an important feature for certain visual analytics tasks. We present TopoLines, a method for smoothing line charts using techniques from Topological Data Analysis. The design goal of TopoLines is to maintain prominent peaks in the data while minimizing any residual error. We evaluate TopoLines for 2 visual analytics tasks by comparing to 5 popular line smoothing methods with data from 4 application domains.

CCS Concepts

• ***Human-centered computing*** → ***Information visualization; Visualization design and evaluation methods;***

1. Introduction

Line charts are used to analyze data in a variety of applications, including identifying stock trends, tracking weather changes, understanding brain activity, etc. While significant increases in data availability allow users to create plots with many data points, relieving visual clutter requires performing additional data processing, such as smoothing. However, the way smoothing modifies the data can have an impact on the performance of visual analytics tasks. We consider smoothing in the context of 2 low-level tasks [AES05], finding extrema (i.e., local minima and maxima) and retrieving a value. These tasks, in essence, require that any smoothing method both retain extrema and minimize any residual error they introduce (i.e., the difference between the input and output data).

A variety of smoothing techniques are available. Uniform subsampling, for example, skips data on a regular interval, and while trivial to implement, the output optimizes upon no particular qual-

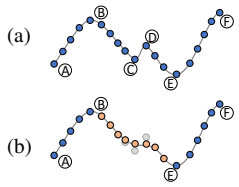


Figure 2: *(a) Input line chart (b) after removing an extrema pair.*

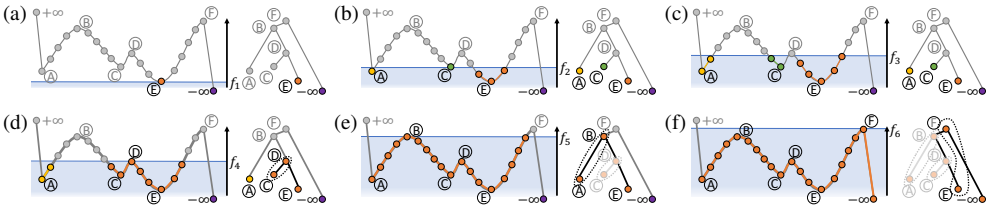


Figure 3: *Six steps of a lower-star filtration (left) and merge tree construction (right) on the line chart.*

ity of the input. Other common methods, such as median, Gaussian, and low-pass cutoff filters, retain low-frequency aspects of the data but potentially lose extrema in the data. Irregular sampling, such as Douglas-Peucker [Ram72, DP73], does a better job preserving extrema, but it retains little detail in the smoothing process.

We address the weaknesses of prior approaches by applying Topological Data Analysis (TDA) to line chart smoothing. We do this by using TDA to capture a hierarchical relationship between extrema that allows removing those of "low importance". At the same time, TopoLines minimizes the residual error between extrema, retaining much of the detail from the input data.

Previously, Kozlov and Weinkauf released *Persistence1D*, a TDA-based class for filtering 1D data using their persistence [KW14]. There has also been work done regarding topological smoothing of 2D and 3D functions [CSvdP10, EMP06, RSM*19, TFL*17]. However, we could find no prior studies that compared topological smoothing to conventional techniques in line charts. Therefore, our contributions are: 1) a description of 1D topological smoothing; 2) optimizations of topological smoothing for the visual analytics tasks of retrieving a value and finding extrema; and 3) an analytical evaluation of the effectiveness of TopoLines and 5 conventional smoothing methods on 4 dataset types.

Our results show that TopoLines is the most effective approach for many, but not all, combinations of data type and task. Almost as important, our results demonstrate the general ineffectiveness of several conventional methods, including median filters, cutoff filters, and uniform subsampling in the tasks and data evaluated.

2. TopoLines: Topologically Smoothed Line Charts

TopoLines smoothing requires 2 steps: 1) extraction of the topology of the data using persistent homology, and 2) smoothing the output by removing extrema based upon a user-selectable threshold.

2.1. Persistent Homology of a Line Chart

We provide a practical description of persistent homology using the line chart in Figure 2 and 3 as an example while leaving further details and theoretical justifications to [EH08].

We use the lower-star filtration of the simplicial complex, $\mathcal{F}$, i.e., the points and edges, on the function $f : \mathcal{F} \to \mathbb{R}$. The lower-star filtration of the data tracks the creation and merging of connected components of the sublevelset $|\mathcal{F}|_i = f^{-1}(-\infty, f_i]$, as f_i is swept from $-\infty \to \infty$, represented by the blue region in Figure 3. The filtration is calculated by first sorting the points of f in increasing order. Then, points are inserted into $|\mathcal{F}|$ one at a time. An edge is added between any neighboring points already in $|\mathcal{F}|_i$.

The relationship between connected components is tracked using a merge tree parameterized by f. When a component first appears at f_i, caused by a local minimum, a leaf node is added to the merge tree at f_i. For example in Figure 3(a), the orange connected component is formed at Ⓔ, and an equivalent leaf node is created in the merge tree. As the plane is swept higher, as in Figure 3(b), new connected components—Ⓐ in yellow and Ⓒ in green—are created.

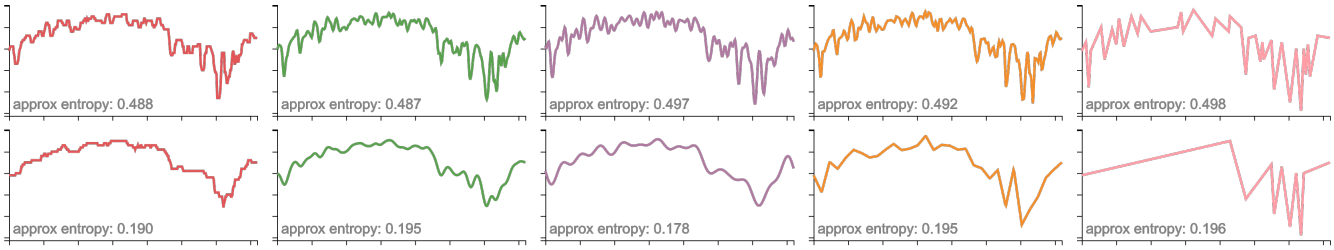
When 2 components merge, representing a local maximum, a merge node is created in the merge tree at f_i and connected to the merged components. In Figure 3(c)/3(d), the green and orange connected components merge at Ⓓ, a local maximum. The connected components are combined, in orange, and a merge node is added to the merge tree. When a merge node is created, it is also paired with a leaf node (i.e., a local maximum is paired with a local minimum). In particular, it is paired with the minimum from the two merging components with the *larger* value. Referring to Figure 3(c)/3(d), the point Ⓓ is paired with the minimum from the green and orange components with the larger value, in this case point Ⓒ. In other words, $f(Ⓒ) > f(Ⓔ)$, therefore, $[Ⓒ, Ⓓ)$ form an extrema pair. The new merged component in orange continues with minimum Ⓔ. Similarly, in Figure 3(d)/3(e), at Ⓑ, the value of the minimum of yellow $f(Ⓐ)$ and orange $f(Ⓔ)$ are compared, and $[Ⓐ, Ⓑ)$ are paired. The output of the operation is the set of all extrema pairs, $C = \{[b_0, d_0), [b_1, d_1), ..., [b_m, d_m)\}$, where b_i and d_i are the local minimum and maximum, respectively, and m is the number of pairs.

Boundaries require special handling, as notable in Figure 3. If a boundary point is a local minimum, e.g., Ⓐ, it is connected to a point at $+\infty$. Similarly, a local maximum boundary point is connected to $-\infty$, e.g., Ⓕ. The additional points ensure all extrema are paired. The algorithm has $\mathcal{O}(n \log n)$ complexity by using the disjoint-set data structure to track connected components. The complexity improves to $\mathcal{O}(n + m \log m)$ by removing all non-extrema from the input before merge tree construction.

2.2. Topological Simplification

The set of extrema pairs, C, is used to guide smoothing, as follows. For each pair, a measure known as *persistence* is calculated, which is simply the difference in function value between the local minimum and local maximum of the pair, i.e., $p_i = |f(d_i) - f(b_i)|$. In effect, this measures the *peak-to-peak amplitude*.

The simplification is controlled by removing extrema pairs from the output through either a user-specified persistence threshold, t, to



(a) Median (b) Gaussian (c) Cutoff (d) Uniform Subsampling (e) Douglas-Peucker

Figure 4: *Conventional smoothing on Climate 17/18 with approximate entropy similar to Figure 1(b)(middle and bottom).*

remove pairs, $\{C_i | p_i < t\}$, or by removing a percentage, q, of pairs by ranking/sorting them, $\{C_i | rank(C_i) < q \cdot m\}$. To reconstruct the line, the extrema that are not removed, in addition to the boundary points, are first placed into the output. For Figure 2(b), this includes Ⓐ, Ⓑ, Ⓔ, and Ⓕ. Next, the intermediate data is calculated.

As pointed out by prior work on 2D manifolds [EMP06] and contour trees [CSvdP10], removing a pair of critical points from the function is as simple as "flattening" the function. For a 1D function, this equates to making the function monotonic between neighboring extrema. For example, in Figure 2(b), removing the [Ⓒ,Ⓓ) critical point pair requires modifying the function such that it is monotonically decreasing between critical points Ⓑ and Ⓔ.

The design space of possible modifications is quite broad—*any* monotonic function satisfies the topological constraint. We apply the additional constraint that the remainder of the function is modified as little as possible. To accomplish this, we use isotonic regression [Bar72], which is a monotonic regression technique that minimizes the least square error. The time complexity of isotonic regression and our reconstruction is $\mathcal{O}(n)$.

3. Evaluation

We compare TopoLines to 5 other smoothing methods:

- A MEDIAN FILTER (see Figure 4(a)) is a nonlinear rank filter, which is particularly good at removing salt-and-pepper noise [Arc05]. For each input datum, the filter extracts a surrounding neighborhood window and outputs its median value. Smoothing is increased by enlarging the window size.
- The GAUSSIAN FILTER (see Figure 4(b)) a commonly used convolutional filter in signal and image processing [KS11]. The approach applies a stencil, whose weights come from a normal distribution, to an input neighborhood. The smoothing level is changed by adjusting the standard deviation of the distribution.
- A low-pass CUTOFF FILTER (see Figure 4(c)) converts the scalar data into the frequency domain via Discrete Fourier Transform (DFT) [CT65], zeros frequencies above a cutoff threshold, and computes the new scalar values with an inverse DFT. The level of smoothing is adjusted by modifying the cutoff frequency.
- UNIFORM SUBSAMPLING (see Figure 4(d)) selects points at regular intervals. Between selected points, linear interpolation is used. The smoothing level is increased by sampling fewer points.
- DOUGLAS-PEUCKER [Ram72, DP73] (see Figure 4(e)) is a non-uniform subsampling approach that optimizes the l^∞-norm of the residual error. The algorithm starts by selecting the boundary points of the input and connects them with linear interpolation. Points are then iteratively added by inserting the input point with the largest distance to the output. The process repeats until a user-specified threshold distance is reached.

3.1. Task Analysis

We considered a variety of low-level tasks based upon the taxonomy of Amar et al. [AES05] and settled upon 2 tasks that we hypothesized TopoLines would perform well. For each, we only consider the resulting impact on the modification of the data, not the perceptual impact of the smoothing (see future work in section 5). For each task, we provide a brief description along with average and worst case analytical measures of performance.

Retrieve Value is a task focused on finding a specific function value on a given chart. An example query would be, "What was the GOOG stock (Figure 1(d)) price on April 15, 2018?" The accuracy of retrieving a value is dependent upon how closely the values of the smoothed data reflect the values in the input data. We measure this by considering the residual error between the original and smoothed data using vector norms.

For the **average case** performance, we consider the l^1-norm: $\|\mathbf{l}\|_1 = \sum_{i=1}^{n} |x_i - x_i'|$, which measures the sum of the absolute value of errors. Since the data length is fixed, comparing the sum of errors is equivalent to comparing the average error. For the **worst case** performance, we consider the l^∞-norm: $\|\mathbf{l}\|_\infty = \max_i |x_i - x_i'|$, which measures only the point of the largest difference between the input and output data.

Find Extrema task is concerned with identifying minima and maxima in the data. An example query would be, "What are the dates of the top 3 peaks of GOOG (Figure 1(d))?" The performance of this task requires that in smoothing, extrema remain in the data. To measure the performance, we calculated the topological difference between the input and smoothed data using methods from TDA [EH10]. First, the persistent homology of the original and smoothed data are calculated, as described in subsection 2.1, to create 2 sets of extrema pairs C and C', respectively. For technical reasons, all pairs with infinite persistence are removed, and all pairs of 0-persistence $[c,c)$ are added to make the cardinality infinite [KMN17]. Let η be a bijection between the 2 sets.

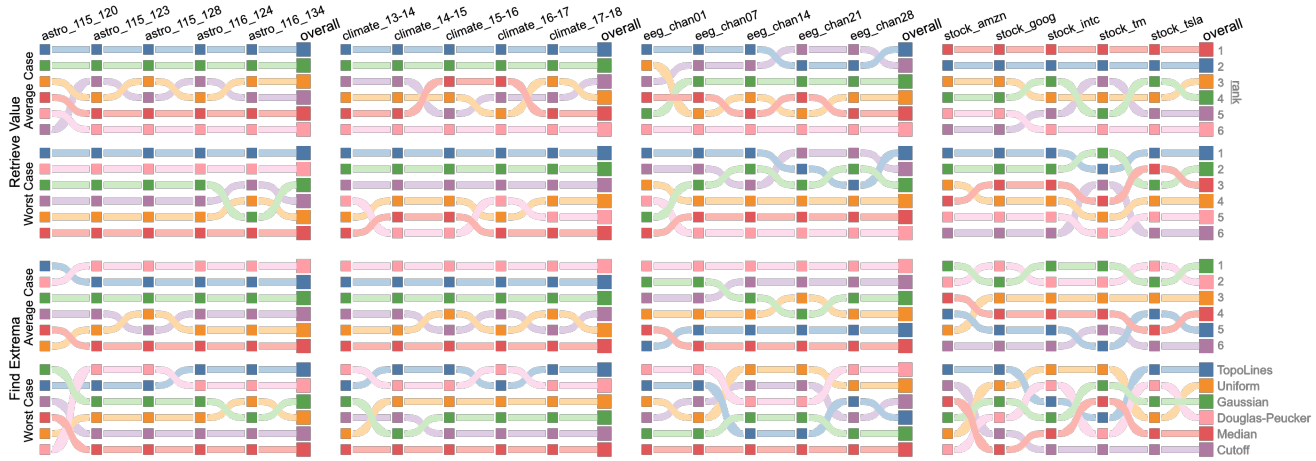


Figure 5: *Ranking of all methods for all data and both tasks. Overall rank is determined by the average rank for the connected datasets/tasks.*

The **average case** is measured using the 1-Wasserstein distance, $W_1(C,C') = \inf_{\eta:C\to C'} \Sigma_{c\in C} \|c-\eta(c)\|_1$, between the input and output extrema pairs, which identifies the average perturbation of extrema. The **worst case** is measured using Bottleneck distance, $W_\infty(C,C') = \inf_{\eta:C\to C'} \sup_{c\in C} \|c-\eta(c)\|_\infty$, which only returns the difference in the extrema with the largest distortion.

Baseline. Each smoothing method offers an adjustable simplification parameter, whose interpretation and output are approach dependent. This variation prevents us from directly using the threshold for comparing methods. Instead, we use approximate entropy as a calibration measure since it has been shown to be a good proxy for line chart complexity [RMCW19] (see Figure 4).

Comparison. To compare methods, we evaluated each technique using the 4 metrics, described above, across the full range of approximate entropy values. Each technique/metric then had the best fit line calculated, and the approaches were ranked by their area under the curve from smallest to largest. In other words, for a given measure, the methods are ranked by which produces the lowest error across the range of entropy values. See the supplemental materials for all measures and best fit lines.

4. Results and Discussion

We test our method using 4 application domains (see Figure 1) of 5 datasets each. Radio *astro*nomy data are 5 spectral "lines" that measure the frequency and amplitude of radio waves emitted by extraterrestrial matter (i.e., gas and dust) and was downloaded from [alm]. *Climate* is a measure of daily high temperature recorded from July to July over 5 periods (20-13/14 through 20-17/18) at a large metropolitan airport downloaded from [YKI*18]. The *EEG* data each contain a window from 5 (of 32 total) channels of a single subject undergoing a visual attention task and was acquired from [Del]. *Stock* trends contain daily closing values for 5 companies (Amazon, Google, Intel, Toyota, and Tesla) over 5 years, starting in February 2015, collected from Yahoo Finance. All results are available in our supplementary materials.

The results for all data, measures, and smoothing approaches are summarized in Figure 5. For all datasets, TopoLines performed best in both average and worst case for the retrieve value task, with the only exception being a second-best finish for average case with the *stock* data. For the find extrema task, TopoLines performed best or second-best in the average and worst cases for *astro* and *climate* data. For the *EEG* and *stock* data, TopoLines performed mostly unremarkably. Our best guess as to this result is that the high frequency of the noise makes many of the local extrema that TopoLines is trying to preserve unimportant for these data.

Among the conventional smoothing methods, it is relevant to note that for the retrieve value task, Gaussian smoothing performed reasonably well overall, and for finding extrema, Douglas-Peucker performed well. Among the other methods, uniform subsampling, cutoff filter, and median filter, none performed consistently well at either task on multiple data types. We, therefore, recommend care in choosing to use them, at least for the tasks we evaluated.

5. Conclusions

In conclusion, we presented a topology-based line chart smoothing method called TopoLines. In the process, we showed that TopoLines has the potential to perform well for certain visual analysis tasks. However, all of these methods, including TopoLines, would benefit from an evaluation framework that considers a broader set of tasks and perceptual differences resulting from their use. In the future, we would like to build upon our current tasks list and run user studies to evaluate how the effect of smoothing on line charts are perceived. We hope to formulate a set of guidelines, based on these studies, that would be helpful for deciding which smoothing methods are best to use in practice.

Acknowledgments

We would like to thank Bei Wang for providing valuable feedback on this project. This work was supported in part by a grant from National Science Foundation (IIS-1513616 and IIS-1845204).

References

[AES05] AMAR R., EAGAN J., STASKO J.: Low-level components of analytic activity in information visualization. In *IEEE Symposium on Information Visualization (InfoVis)* (2005), IEEE, pp. 111–117. doi:10.1109/infovis.2005.24. 1, 3

[alm] ALMA science archive. http://almascience.nrao.edu/aq/. Accessed: 2020-02-01. 4

[Arc05] ARCE G. R.: *Nonlinear signal processing: a statistical approach*. John Wiley & Sons, 2005. doi:10.1002/0471691852. 3

[Bar72] BARLOW R. E.: *Statistical inference under order restrictions; the theory and application of isotonic regression*. Tech. rep., Wiley New York, 1972. doi:10.1111/j.1467-9574.1973.tb00228.x. 3

[CSvdP10] CARR H., SNOEYINK J., VAN DE PANNE M.: Flexible isosurfaces: Simplifying and displaying scalar topology using the contour tree. *Computational Geometry: Theory and Applications 43*, 1 (2010), 42–58. doi:10.1016/j.comgeo.2006.05.009. 2, 3

[CT65] COOLEY J. W., TUKEY J. W.: An algorithm for the machine calculation of complex fourier series. *Mathematics of computation 19*, 90 (1965), 297–301. doi:10.1090/s0025-5718-1965-0178586-1. 3

[Del] DELORME A.: EEG / ERP data available for free public download. https://sccn.ucsd.edu/~arno/fam2data/publicly_available_EEG_data.html. Accessed: 2020-02-01. 4

[DP73] DOUGLAS D. H., PEUCKER T. K.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: the international journal for geographic information and geovisualization 10*, 2 (1973), 112–122. doi:10.3138/fm57-6770-u75u-7727. 2, 3

[EH08] EDELSBRUNNER H., HARER J.: Persistent homology - a survey. *Contemporary Mathematics 453* (2008), 257–282. doi:10.1090/conm/453/08802. 2

[EH10] EDELSBRUNNER H., HARER J.: *Computational Topology: An Introduction*. American Mathematical Society, Providence, RI, USA, 2010. doi:10.1007/978-3-540-33259-6_7. 3

[EMP06] EDELSBRUNNER H., MOROZOV D., PASCUCCI V.: Persistence-sensitive simplification of functions on 2-manifolds. *Proceedings of the 22nd Annual ACM Symposium on Computational Geometry* (2006), 127–134. doi:10.1145/1137856.1137878. 2, 3

[KMN17] KERBER M., MOROZOV D., NIGMETOV A.: Geometry helps to compare persistence diagrams. *Journal of Experimental Algorithmics (JEA) 22* (2017), 1–4. doi:10.1145/3064175. 3

[KS11] KOPPARAPU S. K., SATISH M.: Identifying optimal gaussian filter for gaussian noise removal. In *Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics* (Dec 2011), pp. 126–129. doi:10.1109/NCVPRIPG.2011.34. 3

[KW14] KOZLOV Y., WEINKAUF T.: Persistence1d. https://github.com/yeara/Persistence1D, 2014. 2

[Ram72] RAMER U.: An iterative procedure for the polygonal approximation of plane curves. *Computer graphics and image processing 1*, 3 (1972), 244–256. doi:10.1016/s0146-664x(72)80017-0. 2, 3

[RMCW19] RYAN G., MOSCA A., CHANG R., WU E.: At a glance: Pixel approximate entropy as a measure of line chart complexity. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2019), 872–881. doi:10.1109/tvcg.2018.2865264. 4

[RSM*19] ROSEN P., SETH A., MILLS B., GINSBURG A., KAMENETZKY J., KERN J., JOHNSON C. R., WANG B.: Using contour trees in the analysis and visualization of radio astronomy data cubes. In *Topological Methods in Data Analysis and Visualization (TopoInVis)* (2019). 2

[TFL*17] TIERNY J., FAVELIER G., LEVINE J. A., GUEUNET C., MICHAUX M.: The Topology ToolKit. *IEEE Transactions on Visualization and Computer Graphics* (2017). https://topology-tool-kit.github.io/. doi:10.1109/tvcg.2017.2743938. 2

[YKI*18] YOUNG A. H., KNAPP K. R., INAMDAR A., HANKINS W., ROSSOW W. B.: The international satellite cloud climatology project h-series climate data record product. *Earth System Science Data 10*, 1 (2018), 583–593. doi:10.5194/essd-10-583-2018. 4