Statistical summaries for streamed data from climate simulations: One-pass algorithms~~(v0.6.2)~~

Katherine Grayson[1], Stephan Thober[2], Aleksander Lacima-Nadolnik[1], Ivan Alsina-Ferrer[1], Llorenç Lledó[4], Ehsan Sharifi[2], and Francisco Doblas-Reyes[1,3]

[1]Earth Sciences Department, Barcelona Supercomputing Center, Barcelona, 08034, Spain
[2]Department of Computational Hydrosystems, Helmholtz Centre for Environmental Research, Leipzig, 04318, Germany
[3]Institució Catalana de Recerca i Estudis Avançats, Barcelona, 08010, Spain
[4]ECMWF, Bonn, 53177, Germany

Correspondence:
Katherine Grayson (katherine.grayson@bsc.es)

Abstract. ~~Projections from global~~

Global climate models (GCMs) are ~~a fundamental information source for climate adaptation policies and socio-economic decisions. As such, these models are being progressively~~ being increasingly run at finer ~~spatio-temporal resolutions to resolve smaller scale dynamics . Yet even with increased capacity from High Performance Computing~~ resolutions to better capture small-scale dynamics and reduce uncertainties associated to parameterizations. Despite advances in high-performance computing (HPC)~~the consequent size of the data output (≈ Terabytes to Petabytes), means that native resolution data cannot feasibly be stored for long periods. Lower resolution archives containing a reduced set of variables are often all that is kept, limiting data consumers from harnessing the full potential of these models. To overcome thisgrowing challenge, the climate modelling community is investigating data streaming; a novel way of processing GCM output without having to store variables on disk. In this paper we present a detailed analysis of the use of one-pass algorithms from the '~~, the resulting terabyte- to petabyte-scale data volumes now being produced from GCMs are overwhelming traditional long-term storage. To address this, some climate modelling projects are adopting a method known as data streaming, where model output is transmitted directly to downstream data consumers (any user of climate model data e.g. an impact model) during model run-time, eliminating the need to archive complete datasets. This paper introduces the One_Pass Python package (v0.8.0) that enables users to compute statistics on streamed GCM output via one-pass ~~' package, for streamed climate data. These intelligent data reduction techniques allow for the computation of statistics on-the-fly, enabling climate workflows to temporally aggregate the data output from GCMs into meaningful statistics for the end-user without having to store~~ algorithms – computational techniques that sequentially process data in a single pass, without requiring access to the full time series. Crucially, packaging these algorithms independently, rather than relying on standardised statistics from GCMs, provides flexibility for a diverse range of downstream data consumers and allows for integration into various HPC workflows. We present these algorithms for four different statistics: mean, standard deviation, percentiles and histograms. Each statistic is presented in the context of a use case, showing ~~the statistic applied~~ its application to a relevant variable. For statistics that can be represented by a single floating point value (i.e., mean, standard deviation, variance), the ~~accuracy is within the numerical precision of the machine and~~ results are identical to 'conventional'

approaches within numerical precision limits, while the memory savings scale linearly with the period of time covered by the statistic. For ~~the~~ statistics that require a distribution (percentiles and histograms), we ~~present~~ make use of the t-digest; an algorithm that ~~reduces the full time series~~ ingests streamed data and reduces it to a set of key clusters ~~that represent~~ representing the distribution. ~~We also examine the convergence of these statistics in the sense that they remain representative of a larger sample, ensuring that reduced representations maintain relevance across extended simulations.~~ Using this algorithm~~we find that the accuracy provided is well within the acceptable bounds for the climate variables examined while still providing memory savings that bypass the unfeasible storage requirements of high-resolution~~ , we achieve excellent accuracy for variables with near-normal distributions (e.g., wind speed), and acceptable accuracy for skewed distributions such as precipitation. We also provide guidance on the best compression factor (the memory vs. accuracy trade-off) to use for each variable. We conclude by exploring the concept of convergence in streamed statistics, an essential factor for downstream applications such as bias adjusting streamed data.

1 Introduction

~~Understanding how extreme events, both at regional and global scales, are going to impact society under climate change is of ever growing importance (Seneviratne et al., 2012; Rising et al., 2022; Pörtner et al., 2022)~~

Climate change impacts are often felt at the local level, where decision makers require timely and localised information to anticipate and develop necessary adaptation measures (Katopodis et al., 2021; Orr et al., 2021). Projections from regional and global climate models (RCMs and GCMs) are regularly used to create such information for climate adaptation policies and socio-economic decisions ~~. As demand for accuracy in these projections grows,~~ (Pörtner et al., 2022), but often lack the suitable spatial resolution and temporal frequency to be fully exploited. To support this need, RCMs and GCMs are being run at increasingly finer spatio-temporal ~~resolution to capture both small-scale processes, such as convective storms and oceanic eddies, as well as larger atmospheric dynamics through the improved representation of the interaction between small and large scale dynamical processes (Palmer, 2014; Bador et al., 2020; Iles et al., 2020; Stevens et al., 2020; Rackow et al., 2025). These increases in resolution are also beneficial for end-user analyses or applications, that typically require local information and frequent time samples to produce accurate estimates of climate impacts (Katopodis et al., 2021; Orr et al., 2021).~~

~~Due to the~~ resolutions, which can better resolve the smaller-scale dynamics of the Earth's climate system (Palmer, 2014; Bador et al., 202 . This ongoing movement in the climate community towards increasingly higher spatio-temporal resolutions ~~of climate data, questions are beginning to arise~~ raises the question as to how this data will be managed (Hu et al., 2018; Bauer et al., 2021). The growing size of the ~~data~~ model output makes the current state-of-the-art ~~archives~~ archival method (e.g., Coordinated Regional Climate Downscaling Experiment (CORDEX) and Coupled Model Intercomparison Project (CMIP)) ~~,~~ unfeasible. Moreover, the current archival method has left some data ~~users~~ consumers without their required data, as climate model protocols either limit the number of variables stored or reduce their resolution and frequency (e.g., by storing monthly means or interpolated grids) to cope with the size~~of the archives. As such~~. Here we refer to data consumers as external users who are not involved in the climate modelling process or diagnostics and traditionally access model output via static archives.

In this context, initiatives such as Destination Earth (DestinE) (Bauer et al., 2021; Hoffmann et al., 2023) are ~~investigating the novel method of data streaming, where output arrives to the data consumer~~ utilising a way to process the outputs of climate models as soon as they are produced, without having to save the full data sets to disk. This method, known as data streaming, transmits data in a continuous stream ~~(Kolajo et al., 2019; Marinescu, 2023). Data streaming allows access to the climate data at the highest frequency available~~ to data consumers, who can process it (e.g., ~~hourly) , at native spatial resolution in near-real model~~ via impact models) during run-time ~~. In the context of the climate impact community, this means that climate impact information can be created alongside the climate model, without having to wait for the full simulation to be completed. This provides an unprecedented time-scale reduction~~ (Kolajo et al., 2019; Marinescu, 2023). The data stream allows those data consumers currently limited by reduced storage archives, to access the ~~data and produce meaningful output compared with the current simulation paradigm~~ full model output at the highest frequency available (e.g., ~~CMIP) and the possibility of using variables and frequencies not previously available.~~

~~Yet~~ hourly) and native spatial resolution. However, the advent of data streaming in the climate community poses its own set of challenges. Often, downstream data ~~users~~ consumers require climate data that spans long ~~temporal~~ periods. For example, many hydrological impact models require daily, monthly or annual maximum precipitation values (Teutschbein and Seibert, 2012; Samaniego et al., 2019), while in the wind energy sector, accurate distribution functions of the wind speed are essential ~~(Pryor and Barthelmie, 2010; Lledo, 2019). Obtaining these statistics that span time scales that are potentially longer than the datastreaming window can no longer~~ information (Pryor and Barthelmie, 2010; Lledo, 2019). Moreover, as all models are subject to systematic biases, many users require bias-adjusted climate data. Normally, these computations would be done using ~~the traditional methods. As data in a stream can only be accessed once, this introduces~~ conventional methods, which require the entire dataset to be available when the computation is performed. In the streaming context, the one-pass problem is introduced; how to compute summaries, diagnostics or derived quantities without having access to the whole time series?

In this paper~~we present a detailed analysis of the use of~~ , we introduce the One_Pass package (v0.8.0); a flexible, bespoke Python package built to compute statistics and aid in the bias adjustment of streamed climate data using one-pass ~~algorithms and how they work as intelligent data reduction techniques for streamed climate simulation data . Normally, the computation of statistics is done using the conventional method, where two sequential passes are made through a dataset, first to gather relevant information, and then to perform calculations based on that information. This method requires having the entire dataset available when the computation is performed and we will refer to it as the 'conventional' method throughout the paper to signify the common way of calculating statistics~~algorithms. While one-pass algorithms have been adopted in other fields such as online trading (Loveless et al., 2013) and machine learning (Min et al., 2022), they have yet to find a foothold in climate science, mainly because their use was necessary until now. Unlike the conventional method, one-pass algorithms do not have access to ~~a~~ the whole time series, rather, they process data incrementally every time ~~that the~~ climate model outputs new time steps (Muthukrishnan, 2005). This is done by sequentially processing data chunks as they become available, with each chunk's value being incorporated into a rolling summary, which is then moved into an output buffer before processing the next chunk. ~~While these algorithms have been adopted in other fields such as online trading (Loveless et al., 2013) and machine learning (Min et al., 2022), they have yet to find a foothold in climate science, mainly because they have not been necessary until now.~~

~~Through this work, we show the foundations of the infrastructure required for the climate community to harness the capabilities of high spatio-temporal climate data through data streaming.~~

~~This paper is organised as follows~~Details of the package's design choices – including its requirements for flexibility – and its current use in high-performance computing (HPC) workflows are given in Sect. 2. The remainder of the paper focuses on the mathematical theory behind the one-pass algorithms and investigates their utility and accuracy for climate data analysis. We do not discuss the code implementation, however, examples of the packages can be found in the provided Jupyter notebooks. In Sect. 3 we present the mathematical notation used throughout this paper~~to describe the statistics~~. Sects. 4 to 6 then cover the algorithms used for the mean, standard deviation and ~~distributions respectively~~statistical distributions. These statistics have been chosen ~~as~~ because they represent the most commonly required statistics for climate data, ~~however~~ although many other statistics (i.e., minimum, maximum, threshold exceedance etc.) ~~can be implemented~~ have been implemented in the One_Pass package using the same approach. For each statistic, the one-pass algorithm is first presented, followed by ~~a use case example~~ an example use case which applies the algorithm to a relevant variable over a meaningful time span. With the aid of these use cases, the numerical accuracy is compared to the conventional approach (being able to read the dataset as a whole to compute the statistic)~~are given~~, along with the ~~memory savings provided. In~~ achieved memory savings. In this paper, the conventional approach is always calculated with Python's NumPy package (Harris et al., 2020).

In Sect. 7 we discuss ~~the concept of convergence and how the~~ challenges that arise when using streamed data from one-pass ~~statistics can~~ algorithms for certain applications. For example, climate model outputs are often bias-adjusted using probability distribution functions (PDFs) to perform a quantile-quantile mapping against a reference dataset (Lange, 2019). The One_Pass package can be used to create these PDFs from the streamed data. However, these estimates will initially fluctuate and only converge as more data points are added. It is then crucial to know after how many samples the estimate (i.e., the PDF) is representative of the entire period – in other words, when it has converged. Only if convergence is attained can the PDF estimate be used for ~~bias-adjustment of streamed climate data . Finally conclusions are drawn in 8. We further note that the full Python implementation of these algorithms, ready for use in a streaming workflow, can be found at Grayson (2025a) (v0.6.2)and request~~ downstream applications (e.g., bias adjustment). Overall, the aim of this work is to both showcase the utility and highlight the current limitations of one-pass algorithms for climate data analysis. Moreover, through the provision of the One_Pass Python package, we show the foundations of the infrastructure required for the climate community to harness the capabilities of km-scale climate data through data streaming.

2 Design and implementation in a workflow

The Climate Change Adaptation Digital Twin (ClimateDT) is one of the components of Destination Earth, a flagship initiative of the European Commission (Bauer et al., 2021; Hoffmann et al., 2023). The ClimateDT aims to combine km-scale global climate modeling with impact sector modeling in a single end-to-end workflow. One of the overarching aims is, therefore, to provide climate impact information tailored to different user needs by directly transforming climate model output. However, a gap often exists between the needs of specific data consumers and the general requirements of models. Moreover, the

ClimateDT runs on several different HPC platforms, with several different GCMs. While many modelling groups will agree on providing monthly means via online calculations for specific variables (e.g. FMS (2007)), it is not realistic, nor feasible, for GCMs to tailor and maintain the disparate needs of data consumers from the impact modeling side. For instance, a data consumer focused on renewable energy production might be interested in monthly wind speed percentiles at a specific height in a particular location. The implementation of this output in climate models is impractical, whose goal is to provide outputs that are as general as possible. The development of the ClimateDT as a flexible, scalable framework emerges as a challenge in addition to that posed by the increase in spatio-temporal resolution. The One_Pass package was conceived in the frame of the ClimateDT to fulfil the requirements of downstream data consumers, while maintaining the model outputs relatively agnostic.

The data flow in the ClimateDT is orchestrated through a structured workflow that organizes a sequence of interdependent computational jobs. This workflow is managed using Autosubmit (Manubens-Gil et al., 2016), which automates the submission and monitoring of jobs on the target HPC platform. The One_Pass package runs as one of these jobs in the ClimateDT workflow, serving as an interface between the data production and its consumption. In this job, the One_Pass package operates via a central class `Opa`, which initially takes a request defined by the data consumer. The request contains primarily the information relevant to the ~~reader to follow the documentation for details on implementation~~aggregation of a variable. This information includes, among other things, the variable in question, the statistic to compute and the frequency of aggregation. Once this request is filed, and an instance of this class is created, the resulting object is set up to perform one-pass aggregations, taking in data chunks of arbitrary time length, which belong to the active streaming window. After performing a series of sanity checks on the incoming time steps, the instance of `Opa` updates its internal state to include the requested aggregation up to the time corresponding to the last incoming data chunk. Once enough data have been taken in to complete the requested statistic, the instance will output the requested variable aggregated at the specified frequency and move on to the next calculation.

Because the One_Pass package runs in an HPC environment, it is designed in a persistent manner. This means that the job in which it runs is not required to sit idly and consume HPC hours while there is no model output ready to process. This saves computing resources, however it does require the current status of the statistic to be saved between jobs. This is done via checkpointing, where each instance of the `Opa` class is stored after every update as a binary file. Similarly, upon initialisation, it will load an existing state from a previous execution (if one exists) to use as a starting point. It will not store any internal state after the completion of the statistic. In the context of the ClimateDT, this is all handled by Autosubmit, the workflow manager, although the package can be used with other HPC workflow managers.

3 Mathematical notation

For a given dataset, the following mathematical notation is used to describe the one-pass algorithms:

- n is the current number of data samples (time steps) passed to the statistic.
- w is the length of the incoming data chunk (number of time steps).

- c is the number of time steps required to complete the statistic (i.e., if the model provides hourly output and we require a daily statistic, $c = 24$).
- x_n is one time step of the data at time $t = n$.
- $X_n = \{x_1, x_2, \ldots, x_n\}$ represents the full dataset up to $t = n$.
- $X_w = \{x_{n+1}, \ldots, x_{n+w}\}$, is the incoming data chunk of length w.
- S_n is the rolling summary of the statistic before the new chunk at time $t = n$. This summary varies for each statistic (i.e., if it is the mean statistic $S_n = \bar{X}_n$). This rolling summary will always be of length one in the time dimension.
- g is a one-pass function that updates the previous summary S_n with then new incoming data X_w.

We introduce the chunk length w as, in many cases, a data stream containing a few consecutive time steps will be outputted from a climate model in one go. In the case where the incoming data stream has only one time step, $(w = 1)$, X_w, reduces to x_{n+1}.

4 Mean

4.1 Algorithm description

The one-pass algorithm for the mean is given by

$$\bar{X}_{n+w} = g(S_n, X_w) = \bar{X}_n + w\left(\frac{\bar{X}_w - \bar{X}_n}{n+w}\right), \tag{1}$$

where $\bar{X}_{n+w}$ is the updated rolling mean of the dataset with the new data chunk X_w and the rolling summary S_n is given by the rolling mean $\bar{X}_n$. If $w > 1, \bar{X}_w$, is the temporal conventional mean over the incoming data chunk, however if the data is streamed at the same frequency of the model output with $w = 1$ then $\bar{X}_w = x_{n+1}$, where x_{n+1} is the incoming time stamp.

4.2 ~~Temperature~~2 m air temperature

We apply the mean one-pass algorithm given in eq. (1) ~~in the context of temperature. Understanding the average trends of temperature is crucial, especially considering the ongoing global and regional occurrences of temperature extremes (Mikkonen et al., 2015; . New~~ for the analysis of 2 m air temperature data. European projects such as ~~Segura et al. (2025)~~ nextGEMS (Segura et al., 2025) and DestinationEarth (2024) are aiming to provide global climate projections for a variety of variables at spatial resolutions in the km-scale, ranging from 0.025 to 0.1°. This will allow for granular analysis of projected temperatures to inform climate adaptation at ~~regional scale, yet~~ local to regional scales, although performing basic computations with such vast amounts of data can prove challenging. In this use case for the mean algorithm, we use data from ECMWF's Integrated Forecasting System (IFS) model coupled with the Finite Element / Volume Sea-ice Ocean Model (FESOM) (experiment tco2559-ng5-cycle3)

(Koldunov et al., 2023; Rackow et al., 2025) (run as part of the ~~Segura et al. (2025)~~ nextGEMS project), looking at ~~the~~ 2 m temperature over March 2020. We use ~~the~~ hourly data at native ~~spatial~~ model resolution ($\sim 0.04°$), resulting in a global map containing approximately 26.31 million spatial grid cells, 744 time steps and a full size of 145.82 GB with double precision (float64).

We calculated the March monthly mean of this data using both the conventional method and the one-pass algorithm given in eq. (1). Computing the temporal conventional mean of this data requires the full time series to be loaded into memory, summed across every cell, then divided by the length of the time dimension (in this case, 744). Due to the high memory requirements of this dataset, this was performed on a ~~high memory~~ high-memory node (256 GB) on the Levante ~~supercomputer~~HPC system. The data set was re-chunked into 10 spatial chunks using the Python library ~~dask-xarray~~ Dask-Xarray (Dask, 2024), where each chunk could fit into available memory. The conventional mean was then computed on each chunk. For the one-pass computation, we used a chunk length of $w = 1$ and called into memory each hour x_n of the dataset to simulate streaming, iteratively updating eq. (1) until $n = c$. These two methods highlight that with such large data sets, the conventional mean is not necessarily the simplest approach, as we still need specialised tools and high memory resources for computation. Rather than adding additional complexity, the one-pass approach allows for simpler handling and easier computation.

The results of the one-pass mean algorithm can be seen in Figure 1(a). We note that for plotting convenience, the native grid was interpolated to a $0.1°$ regular lat-lon. Figure 1(b) shows the absolute difference between the conventional (NumPy) and the one-pass mean shown in (a). The difference is represented by randomly distributed noise at the order of 10^{-12}, ~~consistent with the numerical precision of the data. This negligible absolute difference is therefore attributed to~~ an accuracy within an insurmountable precision limit set by the machine precision as opposed to algorithmic discrepancies~~and is well below the accuracy required for the variable~~.

With regards to ~~the~~ memory savings, the one-pass method requires only two data memory blocks, $\bar{X}_w$ and x_n, both approximately 200 MB with double precision (which is the size of one global time step), to be kept in memory at any ~~one~~ point in time. If this computation was performed in real streaming mode, this would require only ~400 MB of memory to compute the monthly mean, $2/744^{\text{th}}$ of the 145.82 GB memory requirements for the conventional method. We also note here that the memory cost of the one-pass method is independent of the length of the statistic time span. For the case of $w = 1$, the memory requirements will always be twice that of storing a spatial field, as opposed to the conventional method, where the memory requirements will increase linearly with the length of the time series required to compute the statistic.

Moreover, this computation demonstrates that the one-pass algorithms do not merely provide memory savings; they provide user-oriented diagnostics that allow for easy computation. Due to this vast reduction in memory requirements, different variables can be computed in parallel to each other, further ~~aiding~~ enhancing usability. The current paradigm of loading the entire dataset is not practical (and in some cases not possible) for data of this magnitude. Indeed, special tools such as Python's ~~dask-xarray~~ Dask-Xarray packages are often required.

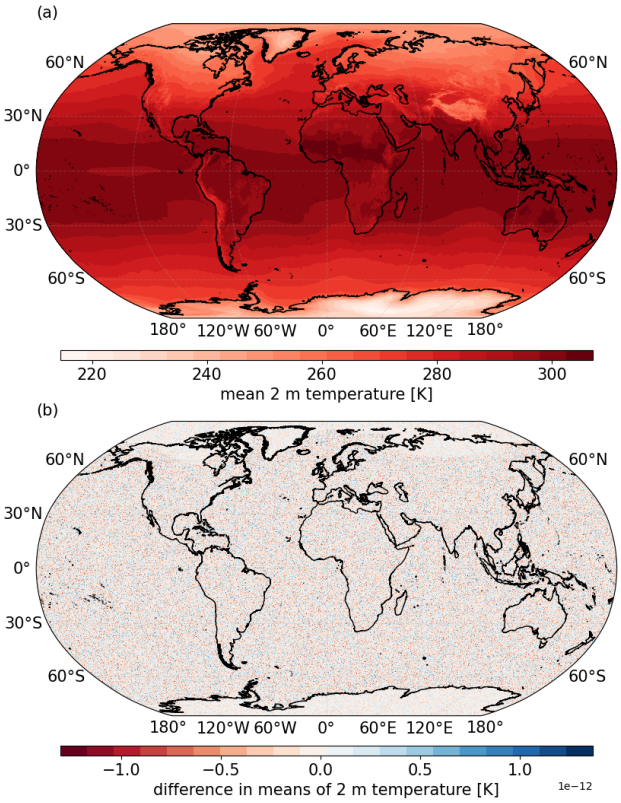

Figure 1. (a) Global monthly mean of ~~the~~ 2 m temperature over March 2020 using hourly data from the IFS model, computed using the one-pass algorithm ~~method~~ given in eq. (1). (b) The absolute difference between (a) and the mean calculated using the conventional method~~, calculated using the python package numpy (Harris et al., 2020)~~.

5 Standard deviation

5.1 Algorithm description

The one-pass algorithm for the standard deviation (and also variance) is calculated over the requested temporal frequency c, by updating two estimates iteratively; the one-pass mean and the sum of the squared differences. First, let the conventional summary for the sum of the squared differences, M_c, be defined as

$$M_c = \sum_{n=1}^{c} \left(x_n - \bar{X}_c\right)^2, \tag{2}$$

where $\bar{X}_c$ is the conventional mean of the whole dataset X_c required for the statistic. For the one-pass calculation of the standard deviation, the rolling summary S_n defines the sum of the squared differences, M_n. In the case where $w = 1$, the

rolling summary is updated by

$$M_{n+1} = g(S_n, x_{n+1})$$
$$= M_n + \left(x_{n+1} - \bar{X}_n\right)\left(x_{n+1} - \bar{X}_{n+1}\right), \tag{3}$$

where $\bar{X}_n$ and $\bar{X}_{n+1}$ are both given by the algorithm for the one-pass mean in eq. (1). Eq. (3) is known as Welford's algorithm. In the case where the incoming data has more than one time step ($w > 1$), M_n is updated by

$$M_{n+w} = g(S_n, X_w)$$
$$= M_n + M_w + \frac{wn\left(\bar{X}_n - \bar{X}_w\right)^2}{n+w}, \tag{4}$$

where M_w is the conventional sum of the squared differences over the incoming data block of length w (given by eq. (2) with $c = w$), $\bar{X}_n$ is the one-pass mean at $t = n$ calculated with eq. (1) and $\bar{X}_w$ is the conventional mean of the incoming data block. See Mastelini and de Carvalho (2021) for details.

Once enough data has been added to the rolling summary M_n so that $n = c$ we calculate the standard deviation σ using

$$\sigma = \sqrt{\frac{M_n}{n-1}}, \tag{5}$$

where M_n is divided by $n - 1$ to obtain the sample variance. Eq. (5) also applies to the conventional summary M_c.

5.2 Sea surface height variability

We apply the one-pass algorithm for standard deviation to the sea surface height (~~ssh). When evaluating the output of any climate model it is necessary to check scientific soundness and quantify uncertainty through quality assessment checks. For ocean data, one such method of soundness can be the standard deviation of the ssh~~ SSH). The standard deviation of SSH can be used to quantify uncertainty between different model ensembles compared to satellite altimetry data. The ~~ssh~~ SSH can be used to better understand ocean dynamics as its variability gives insights into the redistribution of mass, heat and salt within the water column (Close et al., 2020).

We calculate the annual standard deviation using data from the FESOM model (experiment tco2559-ng5-cycle3) (Rackow et al., 2025), again run as part of the ~~Segura et al. (2025)~~ nextGEMS project. We use daily data over 2021 at native model resolution (∼0.05°), making an annual time series - comprised of 7.4 million grid cells and 365 time slices - of 10.09 GB using single precision (float32). We first calculate the standard deviation using the conventional method defined in eq. (2), ~~implemented with Python's numpy package (Harris et al., 2020),~~ followed by the one-pass method in eq. (4), with $w = 2$. Like with the mean calculation, for the conventional calculation the data was spatially ~~rechunked~~ re-chunked into 10 chunks using the Python library ~~dask-xarray~~ Dask-Xarray and each chunk was computed separately, while for the one-pass, two daily time steps were iteratively called into memory to simulate data streaming.

Figure 2(a) shows the one-pass standard deviation calculated using eq. (4) and eq. (5) (using the One_Pass package, see notebooks). Figure 2(b) then shows the difference between the one-pass and the conventional calculation. Again, for plotting

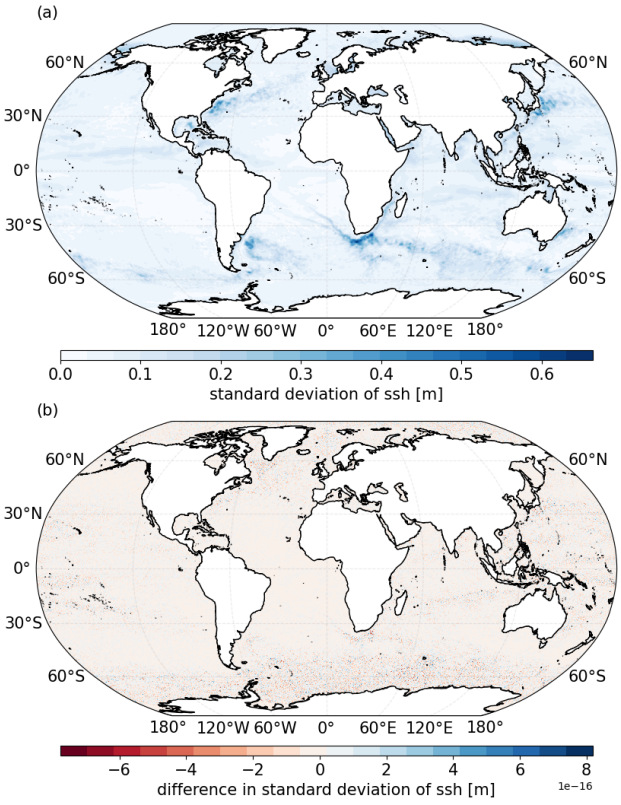

Figure 2. (a) Global annual standard deviation of the sea surface height over 2021 using daily data from the FESOM model, computed using the one-pass method given in eq. (4) and eq. (5), implemented with the One_Pass package. (b) The difference between the one-pass computation and the conventional computation.

convenience, the native grid was interpolated to a 0.25° regular lat-lon. Here, the order of magnitude ~~on~~ of the difference is 10^{-16}; even smaller compared with the mean difference in Fig. 1(b). Interestingly, we also see some structure emerging in the differences shown in Fig. 2(b), which correlates with areas of larger standard deviation~~, however~~ . However, due to the extremely ~~values~~ small values, it is considered negligible in comparison to the required accuracy of the statistic. Therefore, as with the mean statistic, this difference can also be attributed to machine precision limitations.

The memory savings for the standard deviation are slightly ~~less~~ inferior than the mean one-pass algorithm as here, in the case of $w > 1$, other than the current data memory block X_w, four additional data summaries are required to be kept in memory, M_n, M_w, $\bar{X}_n$, $\bar{X}_w$. ~~Yet~~However, as with the mean, these memory requirements are independent of the time-span (sample size) of the statistic and do not increase as the number of values required to complete the statistic (c) increases. In the example presented here, with $w = 2$, approximately 112 MB (single precision) is required in memory, as opposed to the full 10.09 GB of the full dataset. This is a reduction of two orders of magnitude ~~for the~~ in memory requirements.

6 Distributions: percentiles and histograms

6.1 Algorithm description

Unlike the one-pass algorithms for mean and standard deviation (and others such as minimum, maximum, threshold exceedance), where the rolling summary S_n can be described by one floating point value, estimates of a distribution cannot be condensed in such a way. The t-digest algorithm~~has been~~ , developed by Dunning and Ertl (2019) and Dunning (2021)~~to create~~ , creates reliable estimates of ~~probability distribution functions with one~~ PDFs with a single pass through the dataset. The t-digest algorithm is used here~~–~~ , to the best of our knowledge~~–~~ , for the first time ~~on climate data~~ in climate data analysis. Our One_Pass package (and all the results presented in Sect. 6) uses the Python package crick (Crist-Harif, 2023) for the implementation of the t-digest algorithm. We note that other packages may provide slightly different results and efficiency due to variations in the algorithm implementation, however, our preliminary analysis conducted when investigating different packages did not show these to be substantial.

The t-digest algorithm ~~is a clustering algorithm where the dataset ,~~ represents a dataset X_n ~~, is represented~~ by a series of clusters ~~. Each cluster is summarised by a~~ that are defined by their arithmetic mean value and ~~a corresponding weight , representing the average value in the cluster and~~ their corresponding weight (i.e., the number of samples that have contributed to the cluster~~respectively. The data~~/mean). For streamed data, each value (x_n) is added to ~~each cluster~~ its nearest cluster, based on the values proximity to the cluster's mean. As x_n is added, the weight and mean of that cluster are updated using eqn. (1). The clusters are organised in such a way that ~~clusters~~ those corresponding to the extremes of the distribution will contain ~~far~~ fewer samples than those around the median quantile, meaning that the error is relative to the quantile, as opposed to a constant absolute error seen in previous methods (Dunning and Ertl, 2019).

~~For all the results presented in Sect. 6 we use the Python package crick (Crist-Harif, 2023) for the implementation of the t-digest algorithm and note that other packages may provide different results. There are two different algorithmic methods within t-digest, one known as merging and one as clustering. Here we focus on the clustering algorithm, which adds each value in a streamed data chunk, X_w, to its nearest cluster. The unequal size~~ The unequal sizes (i.e., ~~number of samples) contributing~~ the weights) of these clusters are set by the monotonically increasing scale function. While there ~~are different scale functionsthat can be used, the implementation of the t-digest used in this paper uses the scale function~~ is a range of possible scale functions, here we use the most common

$$k(q) = \frac{\delta}{2\pi} \sin^{-1}(2q - 1), \tag{6}$$

where q is the quantile ~~, k is the scale function~~ and δ is the compression parameter. ~~Dunning and Ertl (2019) Figure 1~~ Fig. 3(a) provides a visual representation of the scale function ~~which shows~~ in eqn. (6) for four different δ values (also see Figure 1 in Dunning and Ertl (2019)). Regardless of the compression parameter, there is a steeper gradient of k near $q = 0$ and $q = 1$~~,~~ ~~reducing~~ . As the size of a given cluster is determined by the slope of k, this steeper gradient reduces the cluster weights ~~and increasing the accuracy at the tails . This~~ over the tails of the distribution. For mathematical details of how the cluster sizes are determined, see Dunning and Ertl (2019). The compression parameter does not affect the shape of the scale function, but

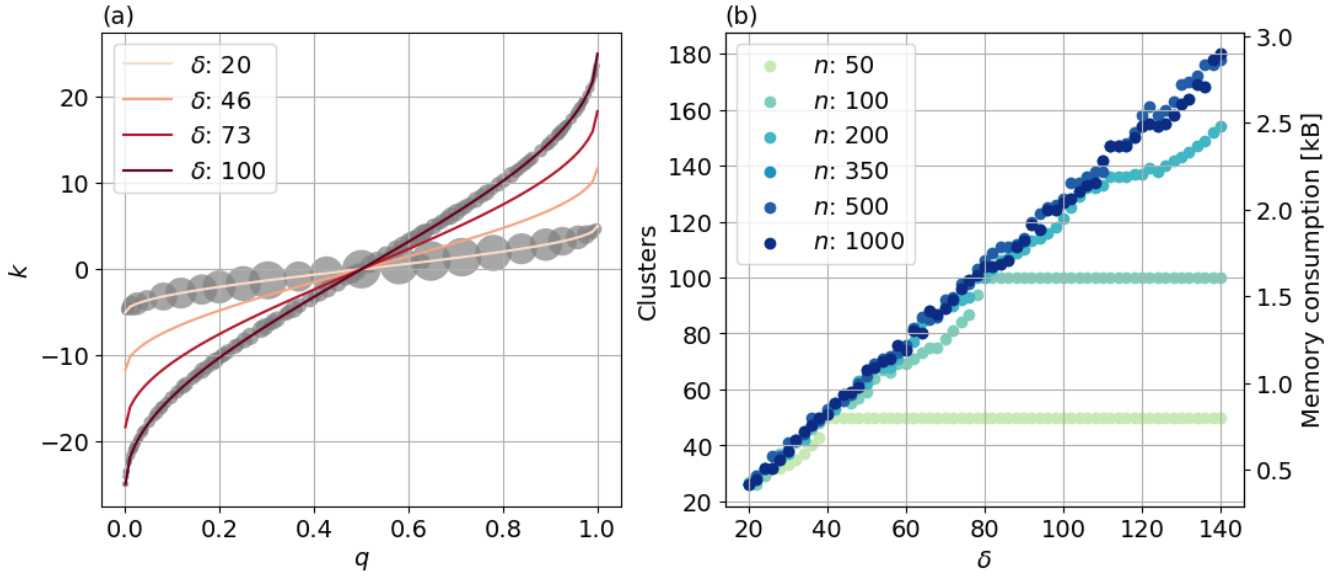

Figure 3. (a) Graphical representation of the scale function in (6) for four different δ. The grey dots for $\delta = 20$ and $\delta = 100$ show the clusters that would be used to represent a dataset of $n = 500$, with a mean value and associated quantile (q) and a weight represented by the size of the dot. (b) Number of clusters used to represent datasets of varying lengths as a function of the compression parameter δ. The corresponding memory consumption [kB] is given on the right-hand axis. Six random datasets (sampled from a uniform distribution) of lengths ranging from 50 to 1000 are used.

it does increase the range of k. The greater the range of k, the more clusters will be used to represent ~~the dataset, meaning a small value of δ will lead to less clusters, less accuracy and more memory saving and vice-versa for a higher δ~~ X_n, providing more accuracy but also increasing the memory required. This is demonstrated by the grey shaded dots in Fig. 3(a), located on the lines for $\delta = 20$ and $\delta = 100$. The number of dots shows the number of clusters used to represent a dataset X_{500}, while the size of the dot indicates the weight. Significantly less clusters are used for $\delta = 20$ compared to $\delta = 100$, reducing the accuracy along with the memory requirements. These clusters ~~can then be converted either to histograms~~ are ultimately converted to a percentile or a histogram (where bin densities may have non-integer values due to the underlying cluster representation)~~or percentile estimates of the distributions~~, based on the closest cluster mean to the required percentile.

The effect of the compression parameter δ on the memory requirements is shown in Fig. 3(b), while the effects on accuracy of the percentile estimates are given in Sects. 6.2 and 6.3~~). Here~~. In Fig. 3(b), six random datasets (from a uniform distribution) of different lengths (n) are used to show how many clusters are required to represent the data as a function of δ (within the range $20 \leq \delta \leq 140$). Beyond ~~a sample size of approximately 350~~ $n > 350$, shown by the darkest three samples, the number of clusters used to represent the data grows linearly with δ. This means that, for the range of δ tested, for all datasets with more than 350 values, the number of clusters is independent of the size of the dataset and is set only by δ i.e., a dataset of 500 values will be represented with the same number of clusters as a dataset of 5 million values.

~~Number of clusters used to represent datasets of varying lengths as a function of the compression parameter δ. The corresponding memory consumption kBis given on the right hand axis. Six random datasets (sampled from a uniform distribution) of lengths ranging from 50 to 1000 are used.~~

For smaller dataset sizes (anything below 350, as shown by the three lighter blue/green samples on Fig. 3~~) the limit on the number of clusters may be set by the number of samples in the dataset. For example, beyond $\delta = 40$, the maximum number of clusters used to represent the the shortest (light green) dataset is 50. The number of clusters cannot grow anymore as it is already the length of the dataset meaning the distribution is represented in its entirety, with each cluster containing one data sample and a weight of one. As the length of the dataset grows, the number of clusters also increases until it reaches the maximum limit defined by δ and shown by the larger datasets in Fig. 3. The key message to take away from this analysis is that, for datasets containing less than 350 samples,~~ (b)) there may be no memory savings (depending on δ) generated from using ~~this~~ the algorithm, as each cluster requires two values (mean and weight) for its representation. For example, when considering $\delta = 140$ and a sample size of 350, 180 clusters are used, requiring 360 values, which exceeds the length of the original dataset. Indeed, for very short datasets such as $n = 50$ (lightest green line on Fig. 3(b)), the number of clusters can not grow beyond 50 as the distribution is already represented in its entirety, with each cluster containing one data sample with a weight of one. For these shorter datasets, there will be no added benefit of increasing δ. However, for longer sample sizes and/or smaller δ, the memory savings generated are substantial.

In the following Sects. 6.2 and 6.3, we examine the use of the t-digest algorithm with two case studies; wind energy in Sect. 6.2 and extreme precipitation events in Sect. 6.3. These two examples have been chosen to examine how well the t-digest algorithm represents ~~both the middle of adistribution~~ (a) a more normally distributed variable around the median percentiles and ~~its ability to capture~~ (b) extreme events at the tails of a heavily skewed distribution. Comparisons are made against the conventional method, calculated with NumPy, which has access to the full dataset and does not rely on one-pass methods. For the t-digest method, we used $w = 1$ in both examples, meaning each x_n was added to its respective digest consecutively, to simulate data streaming.

6.2 Wind energy

We present ~~an~~ here the application of the t-digest in the context of wind energy. With the decarbonisation of the energy system turning into a global necessity, renewable energies such as solar and wind are becoming major contributors to the power network (Jansen et al., 2020). ~~According to the Agency (2024), global installed capacity of wind energy, including both offshore and onshore resources, is expected to reach 1.72 TW by the end of 2028.~~ However, unlike with fossil fuels, wind energy production is heavily affected by atmospheric conditions, subject to both short-term variability (i.e., weather) ~~,~~ ~~and longer term~~ and longer-term variations caused by seasonal and/or interannual variability (Grams et al., 2017; Staffell and Pfenninger, 2018). This volatility makes the integration of wind energy into the power network a challenging task (Jurasz et al., 2022).

Having access to histograms of wind speed ~~at high frequency~~ from high-frequency data (i.e., at hourly or sub-hourly scale) and at hub height (i.e., height of the turbine rotor) is among the requirements of wind farm operators and stakeholders to

estimate the available wind resources at a particular location. This information can be combined with the power curve of the turbines installed at each location to give reasonable estimates of energy production over a period of time. Obtaining an accurate representation of the wind distribution is therefore crucial, as they condense the information from climate simulations required by the wind energy industry and aid in both the understanding of future output from current farms and in the decision-making relating to the viability of a proposed wind farm location (Lledo, 2019). Currently, there are two main methods for describing wind speed distributions; through full histograms of time series data or through fitting probability distribution functions to the data ~~(Morgan et al., 2011; Shi et al., 2021)~~(Morgan et al., 2011; Shi et al., 2021). While the non-parametric approach (time series) generally outperforms the parametric (statistical) one in accurately ~~characterizing~~ characterising the distribution (Wang et al., 2016), it poses numerous challenges attributable to the large amounts of data required (Shi et al., 2021).

~~Here we~~ We investigate the use of the t-digest algorithm to estimate the wind speed distribution from streamed climate data. We use again data from ECMWF's IFS model (experiment tco2559-ng5-cycle3), this time looking at the 10 m wind speeds over December 2020. We again use the hourly data at native ~~spatial~~ model resolution ($\sim 0.04°$), resulting in a global map containing approximately 26.31 million spatial grid cells, 744 time steps and a full size of 145.82 GB with double precision (float64). However, for plotting convenience and storing in Zenodo (Grayson, 2024), the data has been spatially regridded to 1°. Wind speed is calculated from the root of the sum of the squares of the 10 m hourly-mean ~~northward and eastward horizontal~~ zonal and meridional components. We conduct a detailed analysis on two locations, the offshore Moray East wind farm, located at (58.25 °N, 2.75 °E) in the North Sea, and the onshore Roscoe wind farm, positioned at (32.35 °N, 100.45 °W) in Texas. Both are marked on the global map in Fig. 4(a) in red and pink, respectively.

~~Figure~~ Fig. 4 shows a detailed comparison between how the t-digest and the conventional method ~~(implemented using the numpy package in Python) describe a distribution. For the t-digest method, we used $w = 1$, meaning each x_n was added to its' respective digest consecutively, to simulate data streaming. We start by examining the quantile-quantile plots in~~ describe typical wind speed distributions. Fig. 4(b) and (~~e) . Here the numpy percentile estimate is compared to~~ c) show quantile-quantile plots for the NumPy percentile estimates against the t-digest ~~estimate~~ estimates (using $\delta = 60$) for all percentiles ranging from 1 to ~~100.~~ 100 for (b) the Moray East wind farm timeseries and (c) the Roscoe timeseries. The grey shaded areas indicate the range of wind speeds in which most commonly used turbine classes operate (Lledo, 2019). For the ~~off-shore location in the North Sea~~offshore Moray East, the extreme maximum percentiles lie outside the grey shaded region, but the lower tail is within it, whereas for the on-shore ~~location in Texas~~ Roscoe farm, the opposite is true~~for the shaded region~~. This shows that an accurate representation of the full wind speed distribution is required in order to cover the typical range of wind speeds relevant to wind farms. Both (b) and (~~e~~c) show an almost perfect linear relationship~~. The main~~ , evidencing that the t-digest method provides the same level of accuracy as NumPy for typical wind speed distributions. This is further shown in (e, f, h, i), where the histograms of the distributions for the two locations are given using the t-digest method (e,f) and NumPy (h,i). The difference between the ~~two lies~~ distributions provided by the two methods is minimal, with their main difference lying in the storage requirements~~, with~~ . For the NumPy estimate, 5.95 kB is needed to store the full 744 value time series of one grid cell~~used to generate the numpy distribution estimate~~, compared to the 1.28 kB storage for the t-digest estimate based on 80 clusters ($\delta = 60$).

To ensure that the minor differences shown in 4(b) and (c) are not specific to the two chosen locations, we calculated the absolute mean percentile difference (average across all percentile estimates from 1 to 100 i.e., over the quantile-quantile plots) for every global grid cell for $\delta = 60$, the results of which are shown in 4(a). In this global map, no difference exceeds 0.9% of the NumPy value. Converting to absolute terms, this translates to no mean difference exceeding 0.068 m s^{-1} across the globe. Taking the global spatial average of these mean differences gives 0.020 m s^{-1}.

Given such a high level of accuracy achieved with $\delta = 60$, we further investigate the effects of compression in ~~Figures 4(c~~Figs. 4(d) and (~~d~~g). For the same two locations, the difference in the estimate of the 50th and 80th percentiles between the two methods ~~are~~ is shown as a function of δ. The difference is represented as a percentage of the ~~numpy percentile estimate~~NumPy percentile estimate, calculated using the default linear interpolation method. The error bars represent the range of possible differences between the t-digest and the ~~numpy~~ NumPy estimates obtained from using ~~eight of the available interpolation schemes . These different~~ the eight different interpolation schemes available in NumPy. The difference between the interpolation schemes, outlined in Hyndman and Fan (1996), will ~~provide a larger range in a~~ be larger for a given percentile estimate if the data points are more sparsely distributed in the ~~region of interest~~original dataset. Therefore, when any type of interpolation is required to estimate a percentile, there will always be a range ~~a~~ of possible values depending on the interpolation method chosen. ~~These error bars show how the t-digest compares against different methods available in Python.~~ We see that the difference for the Roscoe wind farm in Texas, while incredibly small, is slightly higher for both the 50th and 80th percentiles. This is due to the shape of the two distributions, as evident in the histograms in (~~b~~f) and (~~e~~i). Although the Moray East dataset has a larger variance, it more closely resembles a normal distribution, whereas the dataset for Roscoe is more uniform, with the peak skewed to the left. The shape of the scale function given in eq. (6) will result in clusters with the lowest weight representing the distribution tails, while the clusters representing the middle of the distribution will have a larger weight. This is a clever characteristic of the algorithm, as due to the bulk of the data in a normal distribution being centered around the median, these middle clusters can afford to be larger and cover a broader range of values without impacting precision. As the data from the Roscoe site slightly deviates from this normal distribution, a small increase in the difference is observed.

However, despite this perceived higher difference for the Roscoe wind farm in Texas, the maximum differences for both the 50th and 80th percentiles are approximately 2% and 0.6% respectively ~~, decreasing significantly~~ for the lowest compression factor, further decreasing as we increase δ. ~~Converting these differences back into~~ In absolute terms, the maximum differences (~~at the lowest values of δ~~for $\delta = 20$) are 0.075 and 0.1 m s^{-1} respectively, errors which would be considered ~~extremely small~~ negligible for the end users. Moreover, while the 50th percentile estimate at the Roscoe wind farm has the largest differences across all compression factors (light pink data in (~~c))~~d)), it also has the largest error bars, indicating that the conventional method also contains greater uncertainty. This highlights that the different interpolation methods used by ~~numpy have a larger~~ NumPy have a greater impact on the given result due to the sparser data, also explaining why the discrepancy between the one-pass and conventional methods is higher for this percentile estimate.

~~To ensure that the low differences shown in 4(b)-(e) are not specific to the two chosen locations, we calculated the absolute mean percentile difference (average across all percentile estimates from 1 to 100 i.e., over the quantile-quantile plots) for every global grid cell for $\delta = 60$, the results of which are shown in 4(a). Again, the data has been regridded to 1° for plotting~~

convenience. In this global map, no difference exceeds 0.9% of the numpy value. Converting to absolute terms, this translates to no mean difference exceeding 0.068 m s^{-1} across the globe. Taking the global spatial average of these mean differences gives 0.020 m s^{-1}. Based on this analysis, and the observed We also observe an asymptote in the difference differences around $\delta = 60$ in 4(ed) and (d), we note g), showing that further increasing the compression parameter would not significantly enhance the accuracy of the results for the wind speed distributionthese wind speed distributions. Indeed, given the extremely small calculated difference, using a $\delta = 40$ would likely be sufficient to capture the distribution of global wind speed data required for users.

Overall, as wind speed is best described by a Bi-modal Weibull distribution (Morgan et al., 2011), for monthly wind speed data at hourly time steps, the t-digest with $\delta = 60$ is more than suitable to fully represent the overall distribution while reducing the overall size of the global monthly data from 145.82 GB to ∼33.2 GB. Looking at the monthly time series in one grid cell , this At the grid cell level, the monthly time series is compressed from 5.9 kB to 1.2 kB. For $\delta = 40$, this would reduce further further reduce to 0.85 kB.

This analysis has been conducted on monthly wind speed data, consisting of 744 samples (i.e. hourly values). With the compression of $\delta = 60$, we obtain an approximate five-fold reduction in memory requirements. If We further note that if we were interested in a longer time series , for exampleannually, time series longer than the month shown here – for example, annually – the hourly time series for one grid cell would require 8760 values (∼ 70 kB), while its representation with the t-digest would still only require 1.2 kB. On the contrary, if our interest were in weekly datasets with a time step of 1 hour, containing only 168 values (∼ 1.3 kB), using $\delta = 60$ would would still require 1.2 kB. Here Although no significant memory savings would be obtained, although the histograms could still be provided in real time to the users due to the data streaming.

6.3 Precipitation

In the following Sect. we focus on the t-digest algorithm in the context of extreme precipitation events. It is necessary to examine these extreme events, such as intense rainfall and potential flood risk, as they pose great social, economical economic and environmental threats. Both theory and evidence are showing that anthropogenic climate change is increasing the risk of such extreme events, especially in areas with high moisture availability and during tropical monsoon seasons (Gimeno et al., 2022; Thober et al., 2018; Donat et al., 2016; Asadieh and Krakauer, 2015). The need for climate adaptation measures in vulnerable communities exposed to these risks is pressing and, as with the other use cases in this paper, an accurate representation of the hazard is essential. Consequently, our focus here is on assessing how accurately the t-digest algorithm captures extreme events associated with the upper tail of precipitation distributions.

In this analysis, we use data from the ICOsahedral Non-hydrostatic (ICON) model (Jungclaus et al., 2022; Hohenegger et al., 2023) (experiment ngc2009), looking at precipitation over August 2021. We use half-hourly data using the Healpix spatial grid (Gorski et al., 2005) (∼ 0.04°) containing 20.9 million grid cells. The full monthly dataset for this variable, containing 1488 time steps, requires 116.25 GB of memory using single precision (float32). The precipitation flux, given in kg m^{-2} s^{-1}, has been converted to mm day^{-1} for greater clarity. In the same way as the other sections, for the one-pass method we process the data sequentially with $w = 1$ to mimic data streaming.

As with Sect. 6.2, for plotting convenience and storing in Zenodo (Grayson, 2024), the data has been spatially regridded to 1°.

Figure 5 illustrates the comparison between ~~numpy~~ NumPy and the t-digest in their estimates of the 99th percentile ~~, with a~~ of a precipitation distribution. We focus on four specific locations~~characterized by different precipitation distributions. The chosen four locations, shown by different shades of pink in Fig. 5(a), represent a range of different precipitation distributionsover this period and~~ , each characterised by different distributions, with the locations in Brazil and the North Pacific ~~have been~~ specifically chosen to highlight the areas of largest discrepancy between the one-pass and conventional methods. The histograms in Figure 5(d-k) show the full distributions for the four locations, with (d-e) created using the t-digest and (h-k) using NumPy. A common theme ~~amongst all of these~~ among precipitation distributions is that they are heavily skewed, with the majority of the data falling around zero when there is very little to no precipitation. The dark red ~~histogram in 5(c) represents a~~ histograms in (d,h) are for the location in Brazil ~~(13.50 °S, 60.00 °W) and shows~~ and show an extremely dry month with almost all of the values in the 0 and 1 mm day^{-1} bin (notice the logarithmic scale). ~~We note there is a non-integer number of samples in some of the histogram bins. These non-integer values are due to a weighted contribution from the clusters to the histogram. The histogram in 5(d) is for a location in Colombia (4.50 °N,78.00 °W) and depicts~~ On the contrary, the location in Columbia (e,i) reveals heavy precipitation over the month with maximum values above 400 mm day^{-1} and a more even spread across the distribution. ~~The distribution over the location in the North Pacific (25.50 °N, 142.00 °E), shown in 5(e), also represents a large range of precipitation over the month with high maximum values, however the distribution is significantly more skewed to the first bin compared with 5(d). The final location, for Pennsylvania in North America (40.50 °N, 75.00 °W), shown in 5(f), has a lower range of precipitation values and again has the vast majority of the samples located in the first bin~~Despite the ranges in maximum values, all the t-digest histograms show a non-integer number of samples in some of the histogram bins, whereas the NumPy counterparts show integer density values, representing the exact values in the distribution. These non-integer values are due to a weighted contribution from the clusters to the histogram.

Figure 5(~~b~~c) shows the absolute difference between the ~~numpy~~ NumPy and t-digest estimate of the 99th percentile for the total precipitation as a function of δ. The corresponding number of clusters for each δ is indicated in grey along the upper axis. ~~These differences are given on a log scale, as~~ Here the North Pacific location ~~has larger absolute differences compared to the other locations. This location was chosen explicitly to show where the one-pass and conventional estimates supposedly deviate. In this case (as with many locations with high absolute differences that were examined) , due to the majority of data sitting around zero, there are only a few values that comprise the remainder of the distribution~~shows the greatest absolute difference in the 99th percentile estimate. The reason for this is clear when examining the corresponding histograms for the t-digest (f) and NumPy (j). Focusing on the NumPy histogram (the actual distribution) at the upper end, the data is sparse, with only 4 values in the top quartile of the data range. Due to this sparseness, the 99th percentile estimate falls in-between two data points, so the interpolation method used by NumPy significantly impacts the estimate. This is reflected in the error bars for the North Pacific location in 5(b). While the absolute difference between the t-digest and the ~~numpy~~ NumPy estimate using linear interpolation is large ~~, other numpy~~ (dots), other NumPy interpolation schemes result in a negative difference (not shown due to the logarithmic scale), showing that the t-digest estimate lies in-between the different estimates obtained with the

available ~~numpy~~ NumPy interpolation schemes. These larger differences can therefore be better attributed to the low density of values at the upper tails as opposed to poor representation of these tails by the t-digest. ~~The other three locations show absolute differences around 1 mm day^{-1} (which is considered negligible for most applications) with a decrease in difference as~~ While the location in Colombia also has a high range of values, as there are more samples in the upper tail of the distribution, the t-digest and NumPy estimates are more similar. As seen in Sect. 6.2, there is also a decrease in differences as δ ~~increases. These locations also show large error bars, again due to the highly skewed distribution and impact of interpolation, as was seen in the North Pacific location~~increases.

Figure 5(~~g~~b) shows the same differences presented in 5(~~b~~c) but given as a percentage of the ~~numpy estimate. Again this is shown on a logarithmic scale due to an increase (in some cases by orders of magnitude) in the difference for some locations when given as a percentage, as shown by the Brazilian location in dark red. This dramatic increase is due to the~~ NumPy estimate. Here, the largest error is seen for the location in Brazil. The reason for this large error is similar to the results in (c), where the sparseness of the data in other bins greater than 0-1 mm day^{-1} is obvious in the NumPy histogram (h). However, as this area is so dry and the distribution so skewed, around 99$^{\text{th}}$ ~~percentile estimate being so close to zero for distributions with almost no precipitation, such as the one in Brazil. In the t-digest representation of these distributions, all of the streamed data points with values very close to zero are placed in a few clusters at the tail of the distribution where they then are averaged. As the granularity inside the cluster has then been lost, the percentile estimate will be inaccurate. In absolute value terms this is in inconsequential as the values are so close to zero. Indeed for the Brazilian location the absolute estimate of~~ % of the data lies in the first bin and the 99$^{\text{th}}$ percentile ~~is extremely close to that of numpy and lies around 1~~ estimate is 0.06 mm day^{-1} and 1.02 mm day^{-1} ~~When represented as a percentage however, the difference can be greater than 100.~~ for the NumPy and t-digest (δ = 80) respectively. This highlights the challenges of working with precipitation values below 1, as percentage errors can show unrealistically poor results. To account for these unrealistically poor estimates we calculated the percentage difference for both ~~5(g~~Fig. 5(a) and 5(~~a~~b) using error $= 100|(a-b)|/(b+\epsilon)$ where a is the t-digest estimate, b is the ~~numpy~~ NumPy estimate using the linear interpolation (other than the error bars in 5(g)) and $\epsilon = 1$. This small constant ϵ is introduced to stabilise the calculation when b is extremely small. The results are shown globally (using $\delta = 80$) in Fig. 5(a)~~, again on a logarithmic scale. The data has been regridded to 1° for plotting convenience~~. Most of the differences fall between 1% and 10% of the ~~numpy value, however~~ NumPy value. However, due to the reasons just described~~some differences are unrealistically large~~, drier regions show larger percentage errors. Indeed, larger values are seen around the Saharan desert and in regions of eastern Australia. The average of the global differences ~~are~~ is shown in Table 1 as a function of δ, given as both the percentile estimate and the absolute value. We have included this table to highlight that, due to the extremely low precipitation values for some of the estimates, a higher percentage difference does not necessarily translate to a higher difference in absolute terms. Even with a relatively small δ of 40, the overall relative difference of the 99$^{\text{th}}$ percentile (when averaged globally) is less than 4% (absolute difference of 1.27 mm day^{-1}). Both of these differences decrease to less than 2% and 0.60 mm day^{-1} as δ increases to 120.

Based on the results from Table 1 and Figs. 5(b) and (~~g~~c) we see a reduction in difference with increasing compression that asymptotes around $\delta \approx 80$. This is higher than the asymptotic value seen in Figs. 4(~~e~~d) and (~~d~~g) of approximately $\delta \approx$

Table 1. Global average of the 99$^{\text{th}}$ percentile difference ~~as a function of~~ for different compression values, given as both the absolute difference in (mm day^{-1}) and as a %.

δ	$\lvert(a-b)\rvert$ (mm day^{-1})	$100\lvert(a-b)\rvert/(b+\epsilon)$ (%)
40	1.27	3.77
60	0.91	2.63
80	0.75	2.14
100	0.65	1.86
120	0.60	1.67

60. In general, there are no significant accuracy improvements from using a δ more than approximately 80 (100 clusters) to represent the precipitation distributions, and we would not recommend exceeding this compression factor. Indeed, depending on the specific accuracy requirements, it may be beneficial to reduce this even further. Based on ~~a the~~ results from $\delta = 80$, the entire dataset of 116.25 GB could be represented with 36.57 GB, reducing the memory requirements by approximately a third.

One interesting point to raise is how the scale function for the t-digest algorithm~~–~~, given in eq. (6)~~–~~, impacts these results. While the differences obtained for the precipitation distributions are well within the acceptable limits for most use cases, comparing them with the results in 6.2, we see poorer accuracy. This is due to the wind distributions more closely resembling a normal distribution, which is the distribution that the symmetric scale function describes best. While outside the scope of this investigation, we note that to better represent these skewed precipitation distributions, a non-symmetric scale function that would create larger clusters at the lower tail may more accurately capture the underlying distribution. Another method to improve the representation of the dataset would be to simply impose a cut-off (such as 1 mm day^{-1}) for the data that is added to the digests. Removing this extremely ~~larger~~ large cluster close to zero would, in many cases, improve the representation of the data by the t-digest.

7 Convergence

~~So far we have considered the comparison between the output from the one-pass algorithms vs their conventional equivalents at the end of the full dataset ($n = c$), i.e. how well do they represent the final statistic. However, there is an additional aspect~~

While earlier sections focused on the accuracy and efficiency of one-pass algorithms ~~that requires consideration: the concept of convergence. These one-pass algorithms provide a rolling summary S_n of the statistic after every time chunk, offering potential value to certain applications. For example, when performing bias-adjustment on climate models, it is necessary to compare the model climatology (or probably distribution function) against a reference climatology of a certain area. In streaming mode, we use the t-digest method to build this model probability distribution function , which will evolve as we add more samples. Knowing how many samples need to be added to the~~ for computing statistics, the broader goal of the One_Pass package is to support flexible, real-time analysis of streamed climate data. Beyond statistics, it also integrates

with a companion Python package (not detailed here) designed for performing bias adjustment on streamed model output. Typically, bias adjustment uses quantile-quantile (Q-Q) mapping (Lange, 2019) to correct statistical biases. The t-digest ~~(or rolling~~ algorithm is used to construct a dynamic distribution that enables the bias adjustment on streamed data. However, Q–Q mapping requires a stable and well-sampled distribution (S_n) of the model variable to function effectively. How to determine this leads to the vital question of how long the data stream needs to run for the statistic summary S_n ~~) until it accurately represents the distribution is highly valuable to these bias-adjustment algorithms within~~ to stabilise. We determine this based on the statistical summary's convergence rate (Grau-Sánchez et al., 2010).

The convergence rate is used in numerical analysis to determine how long a sequence of computations needs to run before reaching asymptotic behaviour. In the context of streamed climate data~~. By design, the t-digests provide a summary of the sample given to it. Convergence is related to the fact that the summary is becoming more stable as the sample grows. We investigate at which threshold of n the summary is stable. As an example, the bias adjustment is only effective after enough samples have been added and adjusted, before this~~, this means that the statistic is not only representative of the data seen so far but also stable enough to reflect long-term behaviour. Taking the mean temperature as an example: at the beginning of a time series, the mean may change significantly with each new data point, but as more data accumulates, the changes diminish, signalling that convergence has been reached. Our intent is not to suggest that climate variables become stationary or stop evolving; rather, we aim to determine when the rolling estimate of a one-pass statistic summary S_n ~~should not~~ has stabilised sufficiently to be used confidently for subsequent statistical calculations such as bias adjustment. Moreover, this analysis does not contradict the earlier benefits of one-pass methods (e.g., reduced memory usage, early access to useful summaries). Rather, it offers guidance on how the rolling summaries can be used when $n \neq c$ and how soon those summaries can be used for ~~further analysis~~bias adjustment or similar post-processing steps.

This concept of convergence is explored by examining the rolling summary S_n for 2 m temperature, 10 m wind speed and precipitation ~~flux~~ from the IFS and ICON models. The temperature and wind speed datasets are the same IFS datasets used in Sects. 4 and 6.2 respectively, and the precipitation dataset is the same ICON data used in Sect. 6.3. For both temperature and wind speed the rolling summary $S_n = \bar{X}_n$, while for precipitation S_n is the rolling 50$^{\text{th}}$ percentile estimate. For all rolling summaries $w = 1$, meaning that the number of samples, n, contributing to S_n grows by one each time. Unlike in the previous Sects. where we were interested in the value of S_n at $n = c$, here S_n is stored at every time step, providing a time series of its development denoted as $\mathbf{S}_n = \{S_1, S_2, ..., S_n\}$. The rolling standard deviation (σ) is then taken of $\mathbf{S}_n$ using eq. (4) and eq. (5). This results in a time series of σ defined as $\boldsymbol{\sigma}_n = \{\sigma_1, \sigma_2, ..., \sigma_{n-1}, \sigma_n\}$, where, for example, σ_{n-1} represents the standard deviation of the series $\mathbf{S}_{n-1}$.

The outer axis in Fig. 6 shows the evolution of the running standard deviation, $\boldsymbol{\sigma}_n$, over time using the temperature data at four locations marked in the legend. As expected, Fig. 6 shows how the series $\boldsymbol{\sigma}_n$ ~~for the different datasets are represented - after an initial peak - by inverse exponential asymptotes~~ initially fluctuates before settling into a gradually stabilising trajectory, resembling inverse exponential decay when plotted over time. ~~σ_n decreases as the distribution represented by~~ This reflects how the summary statistic S_n ~~stabilizes due to the addition of more samples. The data shown here uses temperature data but results using precipitation or~~ (be it a mean or a distribution) becomes more stable as additional samples are incorporated. While the

figure shows results for temperature, we observe similar behaviour for precipitation and wind speed datafollow the same shape, just with different maximum , differing only in peak values.

The convergence of the series $\boldsymbol{\sigma}_n$ for the temperature dataset. The outer axis shows four $\boldsymbol{\sigma}_n$ series where the location of the points are given in the legend. The inner grid shows the result of the left hand side of eq. (8).

The inner axis of Fig. 6 then shows the time series of σ_{n+1}/σ_n of the four datasets. We plot this standard deviation ratio to define when the time series have converged, i.e., when sufficient samples have been added to S_n that the standard deviation of the series $\mathbf{S}_n$ has stabilised. This is formally defined using the order of convergence definition We then use the classical definition of convergence rate (Grau-Sánchez et al., 2010)

$$\lim_{n\to\infty} \frac{|\sigma_{n+1} - L|}{|\sigma_n - L|^q} = \mu, \tag{7}$$

where q is the order of convergence, L is the convergence limit, σ_n is the standard deviation of the $\mathbf{S}_n$ time series at time $t = n$ and μ is the rate of convergence (Grau-Sánchez et al., 2010). As both q and μ are unknown in eq. (7) the order of convergence was estimated using

$$q \approx \frac{\log\left|\frac{\sigma_{n+1}-\sigma_n}{\sigma_n-\sigma_{n-1}}\right|}{\log\left|\frac{\sigma_n-\sigma_{n-1}}{\sigma_{n-1}-\sigma_{n-2}}\right|}.$$

Interestingly, when eq. (A1) was calculated over $\boldsymbol{\sigma}_n$ for a random selection of grid points the approximation of q (for all variables) was centered around 1. This indicated a linear convergence ratefor all the standard deviation time series over the global grid cells. Using this approximation and taking convergence rate. Taking $q = 1$, $L = 0$ (see Appendix A for details on these values), eq. (7) could be reduced to

$$\frac{|\sigma_{n+1}|}{|\sigma_n|} = \mu + \epsilon, \tag{8}$$

where we have added the small parameter $\epsilon = 0.005$, which defines the boundaries of convergence, as shown by the black dashed lines on the . The inner axis of Fig. 6 . Once the $\boldsymbol{\sigma}_n$ series fell within this range (and did not leave again) then $\boldsymbol{\sigma}_n$ was considered to have converged. We emphasise that this analysis is not saying the standard deviation of the series $\mathbf{S}_n$ isno longer changing. These climate shows the time series of σ_{n+1}/σ_n for the same locations with ϵ marked by the black dashed lines. We use this standard deviation ratio to define when the time series has "converged" — that is, when a sufficient number of samples have accumulated such that S_n serves as a reliable statistical summary of the statistic up to time n. We reiterate that we do not claim the statistic has reached a final or static value. Climate variables will exhibit long time-scale temporal variability due to climate variability and change and we accept that their mean values and 50$^{\text{th}}$ percentile estimates will shift temporal variability and long-term trends, and we do not assume that their statistical properties remain fixed over time. What these results show is how many samples are required to contribute to the Rather, our focus is on when the variability in the estimate of σ_n itself has sufficiently diminished (i.e., when its fluctuations due to sample size limitations become negligible compared to inherent data variability). This distinction is critical when using one-pass summary algorithms; for example, we must determine when S_n until we can consider it an accurate is a reliable enough representation of the overall distribution at that point in timeunderlying distribution to apply Q-Q mapping for bias adjustment.

~~This analysis is shown globally in Fig. 7.~~ In Fig. 7(a), (c) and (e), we show the global standard deviation σ_n of $\mathbf{S}_n$ at ~~$t = c$ for~~ the end of the month for the air temperature, wind and precipitation data, respectively. Here $c = 744$ for the hourly temperature and wind speed data, while $c = 1488$ for the half-hourly precipitation data. As expected, the standard deviations for all the variables are larger in areas that experience more climate variability. For example, the final standard deviation of the rolling temperature mean shown in (a) shows much larger values away from the equator, where temperature averages will experience seasonal variation. Figure 7(b), (d) and (f) ~~then show the~~ show the number of samples required for the standard deviation σ_n of these rolling summaries to converge, as defined in eqn. (8) when $|\sigma_{n+1}|/|\sigma_n|$ falls within the range $\mu \pm \epsilon$.

~~Unexpectedly~~Encouragingly, the convergence time does not have a strong correlation with the final standard deviation σ_n shown ~~on~~ in the left column. This is ~~re-assuring~~reassuring, as it shows that convergence is partially independent of the actual spread of the data and allows conservative estimates of sample size to be used for global data. What is particularly noteworthy is that, across all these datasets, the number of samples required for convergence is extremely similar. Taking the mean value of Figs. 7(b), (d) and (f) results in 82, 77 and 81 samples, respectively. This is striking, especially given that Fig. 7(f) represents the convergence of the 50$^{\text{th}}$ percentile estimate, as opposed to the mean in Figs. 7(b) and (d). This indicates that, for hourly and half-hourly data, approximately 4 days are required to accurately represent the month.

~~It should be mentioned however that this method is used as a general test of convergence for specific datasets. This analysis was also carried on out on observational data sets (not presented), using both monthly and daily time steps, as opposed to the climate model hourly and half-hourly data shown here. For these data sets the average number of samples required for convergence was approximately 50 when averaged globally. As the values of this observational data were already given as daily or monthly summaries, extreme events (such as wind gusts) were already smoothed in the original time series. The better the representation of shorter extreme events in the original data, the longer we would except for the data to 'converge', as these values will not have been pre-averaged. This is highlighted in 7(f), which uses the half hourly precipitation data. Even though the average number of samples required is 81, there are some areas where 600 samples are needed before convergence, approximately double the maximum shown in 7(b) and (d). Due to the extremely short time step of this data, extreme events will be better represented.~~ Overall, in this Sect. we have presented how the one-pass algorithms provide added value in the context of ~~bias-adjustment~~ bias adjustment for streamed climate data. We present a criteria for stabilisation that we use to define how many samples are required to be added to a rolling one-pass summary S_n before that summary can be used as a representation of the whole distribution.

8 Conclusions

Within the climate modelling community, the generation of increasingly larger data sets from ~~higher resolution~~ km-scale GCMs is becoming almost inevitable. While there is a clear argument for the added value of these ~~high resolution~~ high-resolution models, new challenges of handling, storing and accessing the resultant data are emerging. One novel method being investigated ~~by~~ within the DestinE project (Bauer et al., 2021; Hoffmann et al., 2023) is data streaming, where climate variables at native model resolution are passed directly to ~~climate~~ downstream impact models in near model ~~run time~~run-time. This article has

presented ~~algorithms~~ the algorithms behind the One_Pass (v0.8.0) package, designed to handle ~~this~~ streamed climate data at the km-scale. The application of each algorithm has been demonstrated through relevant use cases in the context of climate change impact studies. We ~~categorized~~ categorised the statistics into two different ~~classes, ones~~ groups: a first group that can be represented by a single floating-point value, and ~~those which require a distribution . For those~~ a second group which requires a distribution estimate. For the first group that require only a single value (e.g., mean, standard deviation, minimum, maximum, threshold exceedance), we obtain accuracy at the order of ~~the~~ machine precision, well beyond the accuracy required or indeed provided by climate models themselves. While providing the same result as the conventional method, these algorithms allow the user to keep only a few rolling summaries in memory at any ~~one~~ moment in time. Unlike a conventional statistic, where the memory requirements for computation scale linearly as the time series grows, the one-pass algorithms for these statistics provide an easily implemented, user-oriented method that bypasses these potentially unfeasible memory requirements.

For the statistics that require a representation of the distribution (e.g. percentiles and histograms), we applied the t-digest method, framed within relevant use cases. In Sect. 6.2 we focused on wind, a variable which requires an accurate representation of the full distribution, useful in the context of renewable energy. ~~Using~~ We recommended using a compression factor of $\delta = 60$ (approximately 80 clusters), where the mean absolute percentile differences for global monthly wind speeds did not exceed 0.9% of the estimate given by the conventional method. For precipitation, given in Sect. 6.3, we focused on the extremes of ~~this skewed precipitation distributions~~skewed precipitation distributions. Due to the presence of ~~low frequency extreme events~~low-frequency extreme events, there was more discrepancy between the ~~numpy~~ NumPy and t-digest estimates. In the cases of high absolute difference between the two estimates, there were also large error bars from the different interpolation schemes of ~~numpy~~NumPy. Examining these distributions showed that these higher differences were due to the sparseness of data in the distribution as opposed to poor representation from the t-digest. In the case of high percentile difference, these were unrealistic differences that occurred due to division by extremely small numbers generated from the ~~numpy~~ NumPy estimate and also occurred when precipitation fell around 0 to 1 mm day^{-1}, negligible values in terms of the user interested in extreme rainfall events. Overall, when averaging the differences globally, we obtained (for $\delta = 60$) 2.63% or 0.91 mm day $^{-1}$ in absolute terms.

~~In both the wind speed and precipitationanalyses, increasing the compression factor and using more clusters to represent the distribution did not necessarily result in higher accuracy.Due to the larger memory requirements at higher compressions and with achieving such accurate representations for both wind speed and precipitation, distributions with $\delta = 60$ and~~ We noted three methods for improving the t-digest estimate for precipitation: (a) use a different scale function (not currently implemented in the One_Pass package), (b) set a threshold for the data (i.e. ignore all samples below 1 mm day^{-1}) (c) examine the t-digest histograms rather than a single percentile estimate. We also recommended a slightly higher compression factor of $\delta = 80$~~respectively, we would not recommend using larger compression factors~~.

In the final Sect. 7, we presented the concept of convergence for the one-pass statistical summaries S_n. While one-pass algorithms provide immediate access to evolving statistical summaries, their reliability for distribution-based applications like bias adjustment depends on the stability of the underlying estimates. Our convergence analysis offered a practical guideline for determining when these summaries can be considered representative of the full distribution. This does not imply that the

climate variables themselves have stabilised, but rather that the evolving summary has reached a usable approximation (i.e., a sufficiently representative estimate of the complete time series). By quantifying convergence in this way, we provide users with the tools to make informed decisions about when to trust and apply one-pass statistics in downstream statistical applications, such as quantile-quantile mapping.

Overall, we have demonstrated the effectiveness of one-pass algorithms on streamed climate data and ~~provide~~ provided their Python implementation in the One_Pass package, ready for use in data streaming ~~work-flows~~ workflows (Grayson, 2025a). These algorithms not only provide accuracy well within the required limits of climate model variables but also empower users to harness the full potential of high-resolution data, both in space and time. Indeed, while some of the methods contain small errors (specifically the t-digest), we note that not harnessing the added value of high-resolution data due to storage limitations will also incur a potentially more significant error. Due to the fact that only a few rolling summaries are required to be kept in memory, these statistics become time independent, allowing users dealing with high-resolution GCMs to select any variables at their native model resolution and process them in ~~next to near model runtime. This eliminates the constraints of~~ near real-time model run-time. This has the potential to eliminate the constraints experienced by some users when relying on pre-defined archives of set climate variables, typically provided months to years after the ~~models have been run~~simulations have been completed. With the continuing movement ~~to~~ towards higher resolution, ~~the streaming of climate data~~ climate data streaming will become a fundamental paradigm of data processing and analysis. This paper showcases the features of one-pass algorithms across a range of relevant statistics that can be harnessed to work in the new era of data streaming.

Data availability. Full data from the nextGEMS cycle 3 are openly accessible and can be found at NextGEMS, including output from the development Cycle 3 (Koldunov et al., 2023) and production runs (Wieners et al., 2024) for both ICON and IFS. All the nextGEMS netCDF data used to demonstrate the use of the one-pass algorithms described in this paper (e.g. looping through the time dimension to simulate data streaming) and to create all the figures in the study are available at the Zenodo repository 'nextGEMS cycle3 datasets: statistical summaries for streamed data from climate simulations v3' with DOI https://doi.org/10.5281/zenodo.12533197, hosted at https://zenodo.org/records/12533197, with Creative Commons Attribution 4.0 International license (Grayson, 2024).

Code availability. Version 0.5 of the repository that contains the Jupyter notebooks to reproduce all the figures, is preserved at DOI https://doi.org/10.5281/zenodo.15439803 and developed on Github (Grayson, 2025b) with url https://github.com/kat-grayson/one_pass_algorithms_paper/tree/main. The source code for the one-pass package v0.8.0 implementation, ready for integration into data streaming workflows, with DOI https://doi.org/10.5281/zenodo.15438184 can be found at https://github.com/DestinE-Climate-DT/one_pass and licensed under Apache License, version 2.0 (Grayson, 2025a).

Author contributions. Llorenç Lledó conceptualised the ideas behind the study, while Katherine Grayson and Stephan Thober developed the methodology. Katherine Grayson wrote the One_Pass package, conducted the formal analysis and wrote the original draft. Stephan Thober

and Francisco Doblas-Reyes both supervised. Aleksander Lacima-Nadolnik and Ehsan Sharifi both helped validate the study. Ivan Alsina-Ferrer helped with revisions, the continued maintenance of the package and reformatting of the notebooks. All authors contributed to the review and editing process.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. We would first like to acknowledge Bruno Kinoshita for his help and advice on the t-digest algorithm. We would also like to acknowledge Paolo Davini and Jost Von Hardenberg for developing tools that allowed for all the data retrieval. We would like to acknowledge the Generalitat de Catalunya (ClimCat, ARD209/22/000001) and the European Commission Destination Earth Program. Destination Earth is a European Union funded initiative and is implemented by ECMWF, ESA, and EUMETSAT. We would also like to acknowledge the European Commission Horizon 2020 Framework Program nextGEMS, under grant number GA 101003470.

Appendix A: Convergence order estimate

As both q and μ are unknown in eq. (7) the order of convergence was estimated using

$$q \approx \frac{\log\left|\frac{\sigma_{n+1}-\sigma_n}{\sigma_n-\sigma_{n-1}}\right|}{\log\left|\frac{\sigma_n-\sigma_{n-1}}{\sigma_{n-1}-\sigma_{n-2}}\right|}. \tag{A1}$$

When eq. (A1) was calculated over $\boldsymbol{\sigma}_n$ for a random selection of grid points, the approximation of q (for all variables) was centered around 1. This indicated a linear convergence rate for all the standard deviation time series over the global grid cells.

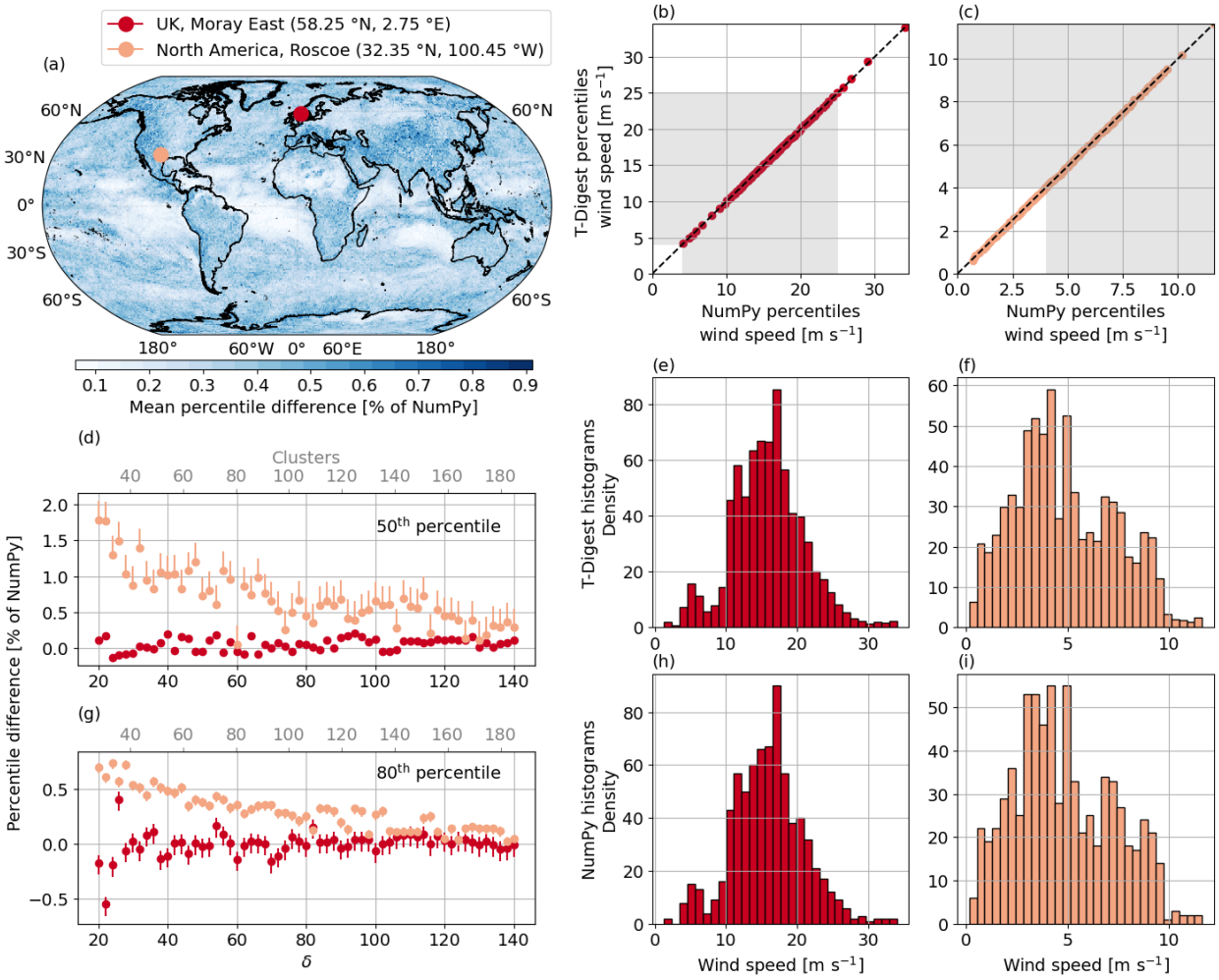

Figure 4. (a) Global map showing the absolute mean difference between the t-digest ($\delta = 60$) and ~~numpy~~ NumPy (using the linear interpolation method) estimate of all wind speed percentiles from 1 to 100 given as a percentage of the ~~numpy~~ NumPy value. Wind speed hourly data across December 2020 from the IFS model. The ~~map highlights two wind farm locations, one off-shore called~~ red dot marks the offshore Moray East wind farm in the North Sea (58.25 °N, 2.75 °E), ~~marked with a red dot~~ and ~~another on-shore called~~ the pink dot the onshore wind farm Roscoe, in Texas (32.35 °N, 100.45 °W)~~, marked with a pink dot~~. (b) ~~The quantile-quantile~~ Quantile-quantile plot for percentiles 1 to 100 ~~calculated using Python's numpy~~ comparing NumPy and the t-digest algorithm (with δ = 60) for ~~the off-shore location in the North Sea~~Moray East. The black dashed line represents the one-to-one fit, while the grey shaded area shows the range of wind speeds that most commonly used turbines operate in (Lledo, 2019). ~~The upper light blue histogram is made using~~ (c) Same as (b) but for the Roscoe wind farm. (e,h) Histograms of the Moray East time series, (e) calculated with the t-digest ~~algorithm~~ ($\delta = 60$)~~while the darker blue is made using~~ , (h) NumPy. (f,i) Same as (e,h) but for the ~~built-in Python function~~Roscoe time series. (~~e~~d) The difference between the t-digest and ~~numpy~~ NumPy calculation of the 50th percentile, given as a percentage of the ~~numpy~~ NumPy value, as a function of δ for both marked locations. The error bars show the possible differences when employing all available ~~numpy~~ NumPy interpolation schemes, rather than the default linear interpolation method. (~~d~~g) The same ~~information~~ as (~~e~~d) but ~~showing the difference~~ for the 80th percentile.~~(e) The same as (b) but for the on-shore location in Texas.~~

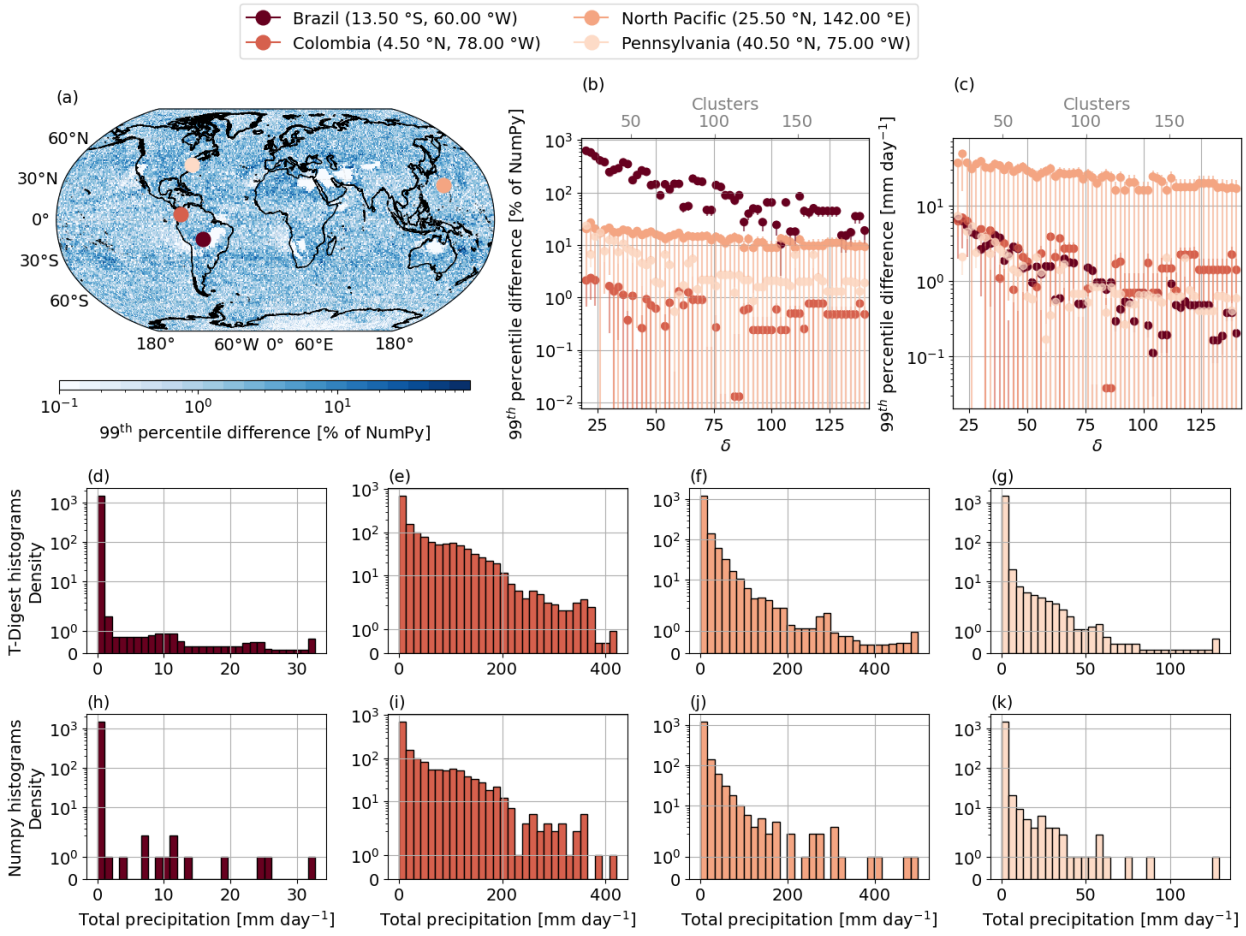

Figure 5. (a) The global map showing the absolute difference between the t-digest ($\delta = 80$) and ~~numpy~~ NumPy (using the linear interpolation method) estimate of the 99th precipitation percentile given as a percentage of the ~~numpy~~ NumPy value. Precipitation half-hourly data from the ICON model (experiment ngc2009) over August ~~2021.~~ 2021 is used in each panel. The ~~locations~~ location of the four solid pink circles ~~are~~ is given in ~~their respective histograms~~the legend, and the following panels use the same colours to indicate the locations. (b) The ~~absolute~~ difference between the ~~numpy~~ NumPy and t-digest 99th percentile estimate, given as a % of the NumPy estimate, shown as a function of compression for the four marked locations on (a). The upper grey axis shows the number of clusters used in each digest for each δ. The error bars represent the range of possible ~~absolute~~ differences based on all available ~~numpy~~ NumPy interpolation schemes, in contrast to the default linear interpolation method (see text for details). (c) ~~S~~Same as (~~f~~b) but given as the absolute difference in mm day^{-1}. (d-g) Histograms calculated using the t-digest with $\delta = 80$ showing the distribution of the total precipitation (mm day^{-1}) ~~at all four location~~ for ~~their respective August 2021 time series. Each histogram color corresponds to its~~ each location~~on the global map and the histograms are calculated using the t-digest with $\delta = 80$.~~ . (gh-k) Same as (~~b~~d-g) but ~~given as a percentage of the numpy value~~calculated using NumPy.

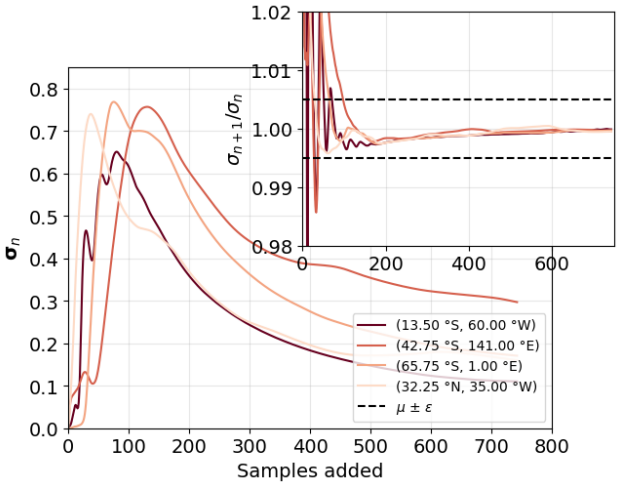

Figure 6. The convergence of the series $\boldsymbol{\sigma}_n$ for the temperature dataset. The outer axis shows four $\boldsymbol{\sigma}_n$ series where the location of the points is given in the legend. The inner axis shows the convergence rate given in eq. (8).

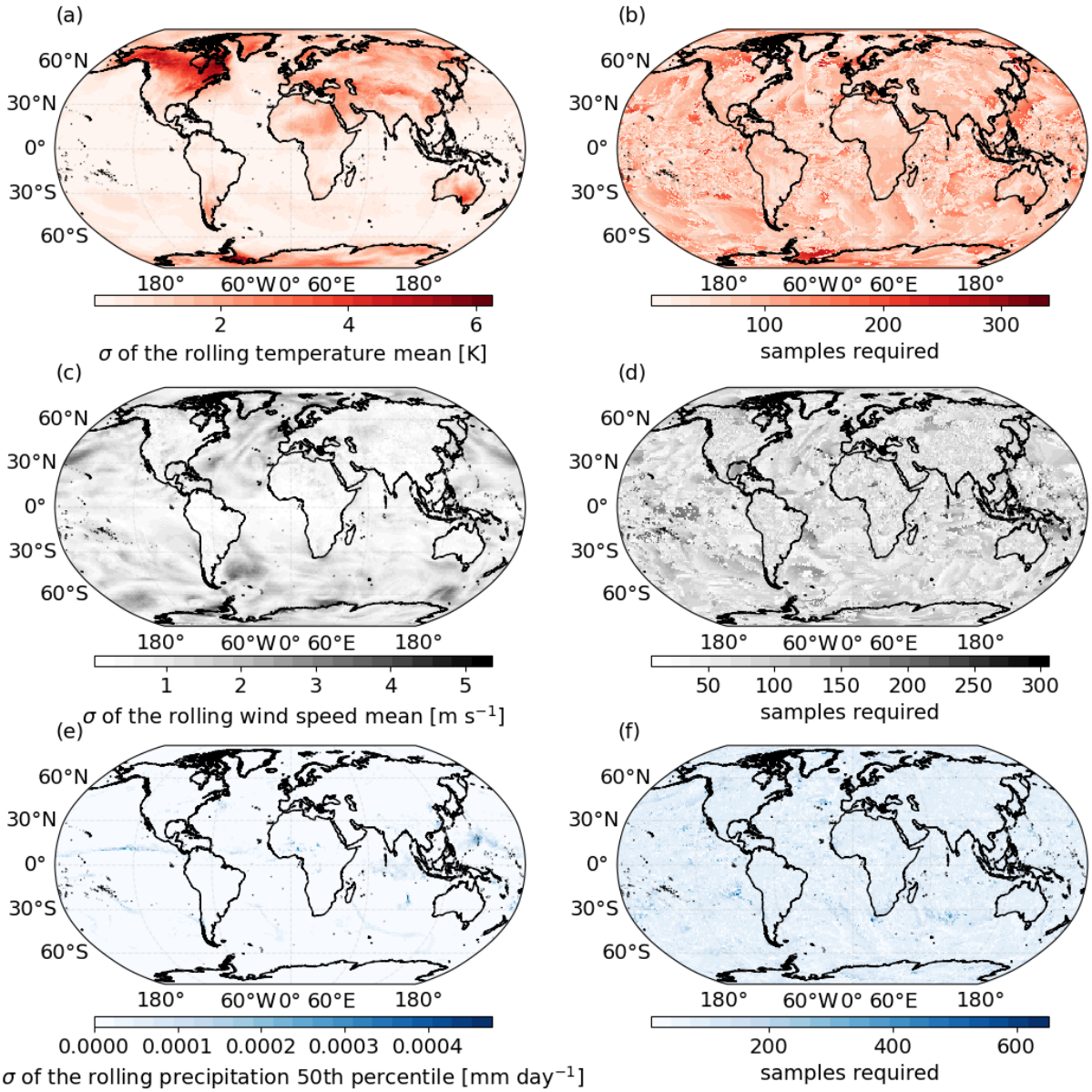

Figure 7. (a) The standard deviation of the full $\mathbf{S}_n$ time series at the end of the IFS monthly temperature ~~time-series~~ time series during March 2020. Here $S_n = \bar{X}_n$. (b) The number of samples required (i.e. number of time steps) that it takes for the rolling standard deviation of the $\mathbf{S}_n$ time series to converge. Convergence is defined in the text. (c) Same as (a) but for the IFS monthly wind speed time series over December 2020. (d) Same as (b) but for wind speed. (e) Same as (a) and (c) but here S_n is the estimate of 50$^{\text{th}}$ using the ICON precipitation time series over August 2021. (f) Same as (b) and (d) but for precipitation.

References

Agency, I. E.: Renewable electricity capacity additions by technology and segment, 2016 - 2018, https://www.iea.org/data-and-statistics/charts/renewable-electricity-capacity-additions-by-technology-and-segment-2016-2028, 2024.

Asadieh, B. and Krakauer, N. Y.: Global trends in extreme precipitation: Climate models versus observations, Hydrology and Earth System Sciences, 19, 877–891, https://doi.org/10.5194/hess-19-877-2015, 2015.

Bador, M., Boé, J., Terray, L., Alexander, L. V., Baker, A., Bellucci, A., Haarsma, R., Koenigk, T., Moine, M. P., Lohmann, K., Putrasahan, D. A., Roberts, C., Roberts, M., Scoccimarro, E., Schiemann, R., Seddon, J., Senan, R., Valcke, S., and Vanniere, B.: Impact of Higher Spatial Atmospheric Resolution on Precipitation Extremes Over Land in Global Climate Models, Journal of Geophysical Research: Atmospheres, 125, https://doi.org/10.1029/2019JD032184, 2020.

Bauer, P., Stevens, B., and Hazeleger, W.: A digital twin of Earth for the green transition, Nature Climate Change, 11, 80–83, https://doi.org/10.1038/s41558-021-00986-y, 2021.

Close, S., Penduff, T., Speich, S., and Molines, J. M.: A means of estimating the intrinsic and atmospherically-forced contributions to sea surface height variability applied to altimetric observations, Progress in Oceanography, 184, https://doi.org/10.1016/j.pocean.2020.102314, 2020.

Crist-Harif, J.: https://github.com/dask/crick, 2023.

Dask: https://github.com/dask/dask, 2024.

DestinationEarth: https://destination-earth.eu/, 2024.

Donat, M. G., Lowry, A. L., Alexander, L. V., O'Gorman, P. A., and Maher, N.: More extreme precipitation in the world's dry and wet regions, Nature Climate Change, 6, 508–513, https://doi.org/10.1038/nclimate2941, 2016.

Dunning, T.: The t-digest: Efficient estimates of distributions, Software Impacts, 7, 100 049, https://doi.org/10.1016/j.simpa.2020.100049, 2021.

Dunning, T. and Ertl, O.: Computing Extremely Accurate Quantiles Using t-Digests, arXiv[preprint],arXiv:1902.04023, 2019.

FMS: https://github.com/NOAA-GFDL/FMS/tree/main?tab=License-1-ov-file, 2007.

Gimeno, L., Sorí, R., Vázquez, M., Stojanovic, M., Algarra, I., Eiras-Barca, J., Gimeno-Sotelo, L., and Nieto, R.: Extreme precipitation events, Wiley Interdisciplinary Reviews: Water, 9, 1–21, https://doi.org/10.1002/wat2.1611, 2022.

Gorski, K. M., Hivon, E., Banday, A. J., Wandelt, B. D., Hansen, F. K., Reinecke, M., and Bartelmann, M.: HEALPix: A Framework for High-Resolution Discretization and Fast Analysis of Data Distributed on the Sphere, The Astrophysical Journal, 622, 759–771, https://doi.org/10.1086/427976, 2005.

Grams, C. M., Beerli, R., Pfenninger, S., Staffell, I., and Wernli, H.: Balancing Europe's wind-power output through spatial deployment informed by weather regimes, Nature Climate Change, 7, 557–562, https://doi.org/10.1038/NCLIMATE3338, 2017.

Grau-Sánchez, M., Noguera, M., and Gutiérrez, J. M.: On some computational orders of convergence, Applied Mathematics Letters, 23, 472–478, https://doi.org/10.1016/j.aml.2009.12.006, 2010.

Grayson, K.: https://doi.org/https://zenodo.org/doi/10.5281/zenodo.12533197, 2024.

Grayson, K.: https://doi.org/https://doi.org/10.5281/zenodo.15438184, 2025a.

Grayson, K.: https://doi.org/https://doi.org/10.5281/zenodo.15439803, 2025b.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant,

P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E.: Array programming with NumPy, Nature, 585, 357–362, https://doi.org/10.1038/s41586-020-2649-2, 2020.

Hoffmann, J., Bauer, P., Sandu, I., Wedi, N., Geenen, T., and Thiemert, D.: Destination Earth – A digital twin in support of climate services, Climate Services, 30, 100 394, https://doi.org/10.1016/j.cliser.2023.100394, 2023.

Hohenegger, C., Korn, P., Linardakis, L., Redler, R., Schnur, R., Adamidis, P., Bao, J., Bastin, S., Behravesh, M., Bergemann, M., Biercamp, J., Bockelmann, H., Brokopf, R., Brüggemann, N., Casaroli, L., Chegini, F., Datseris, G., Esch, M., George, G., Giorgetta, M., Gutjahr, O., Haak, H., Hanke, M., Ilyina, T., Jahns, T., Jungclaus, J., Kern, M., Klocke, D., Kluft, L., Kölling, T., Kornblueh, L., Kosukhin, S., Kroll, C., Lee, J., Mauritsen, T., Mehlmann, C., Mieslinger, T., Naumann, A. K., Paccini, L., Peinado, A., Praturi, D. S., Putrasahan, D., Rast, S., Riddick, T., Roeber, N., Schmidt, H., Schulzweida, U., Schütte, F., Segura, H., Shevchenko, R., Singh, V., Specht, M., Stephan, C. C., Von Storch, J. S., Vogel, R., Wengel, C., Winkler, M., Ziemen, F., Marotzke, J., and Stevens, B.: ICON-Sapphire: Simulating the components of the Earth system and their interactions at kilometer and subkilometer scales, Geoscientific Model Development, 16, 779–811, https://doi.org/10.5194/gmd-16-779-2023, 2023.

Hu, F., Yang, C., Schnase, J. L., Duffy, D. Q., Xu, M., Bowen, M. K., Lee, T., and Song, W.: ClimateSpark: An in-memory distributed computing framework for big climate data analytics, Computers & Geosciences, 115, 154–166, https://doi.org/https://doi.org/10.1016/j.cageo.2018.03.011, 2018.

Hyndman, R. J. and Fan, Y.: Sample Quantiles in Statistical Packages, American Statistician, 50, 361–365, https://doi.org/10.1080/00031305.1996.10473566, 1996.

Iles, C. E., Vautard, R., Strachan, J., Joussaume, S., Eggen, B. R., and Hewitt, C. D.: The benefits of increasing resolution in global and regional climate simulations for European climate extremes, Geoscientific Model Development, 13, 5583–5607, https://doi.org/10.5194/gmd-13-5583-2020, 2020.

Jansen, M., Staffell, I., Kitzing, L., Quoilin, S., Wiggelinkhuizen, E., Bulder, B., Riepin, I., and Müsgens, F.: Offshore wind competitiveness in mature markets without subsidy, Nature Energy, 5, 614–622, https://doi.org/10.1038/s41560-020-0661-2, 2020.

Jungclaus, J. H., Lorenz, S. J., Schmidt, H., Brovkin, V., Brüggemann, N., Chegini, F., Crüger, T., De-Vrese, P., Gayler, V., Giorgetta, M. A., Gutjahr, O., Haak, H., Hagemann, S., Hanke, M., Ilyina, T., Korn, P., Kröger, J., Linardakis, L., Mehlmann, C., Mikolajewicz, U., Müller, W. A., Nabel, J. E., Notz, D., Pohlmann, H., Putrasahan, D. A., Raddatz, T., Ramme, L., Redler, R., Reick, C. H., Riddick, T., Sam, T., Schneck, R., Schnur, R., Schupfner, M., von Storch, J. S., Wachsmann, F., Wieners, K. H., Ziemen, F., Stevens, B., Marotzke, J., and Claussen, M.: The ICON Earth System Model Version 1.0, Journal of Advances in Modeling Earth Systems, 14, https://doi.org/10.1029/2021MS002813, 2022.

Jurasz, J., Kies, A., and De Felice, M.: Complementary behavior of solar and wind energy based on the reported data on the European level—a country-level analysis, Complementarity of Variable Renewable Energy Sources, pp. 197–214, https://doi.org/10.1016/B978-0-323-85527-3.00023-6, 2022.

Katopodis, T., Markantonis, I., Vlachogiannis, D., Politi, N., and Sfetsos, A.: Assessing climate change impacts on wind characteristics in Greece through high resolution regional climate modelling, Renewable Energy, 179, 427–444, https://doi.org/https://doi.org/10.1016/j.renene.2021.07.061, 2021.

Kolajo, T., Daramola, O., and Adebiyi, A.: Big data stream analysis: a systematic literature review, Journal of Big Data, 6, https://doi.org/10.1186/s40537-019-0210-7, 2019.

Koldunov, N., Kölling, T., Pedruzo-Bagazgoitia, X., Rackow, T., Redler, R., Sidorenko, D., Wieners, K.-H., and Ziemen, F. A.: https://doi.org/doi:10.26050/WDCC/nextGEMS_cyc3, 2023.

Lange, S.: Trend-preserving bias adjustment and statistical downscaling with ISIMIP3BASD (v1.0), Geoscientific Model Development, 12, 3055–3070, https://doi.org/10.5194/gmd-12-3055-2019, 2019.

Lledo, L.: Seasonal forecasts of wind power generation, Renewable Energy, 143, 91–100, https://doi.org/10.1016/j.renene.2019.04.135, 2019.

Loveless, J., Stoikov, S., and Waeber, R.: Online Algorithms in High-frequency Trading, Queue, 11, 30–41, https://doi.org/10.1145/2523426.2534976, 2013.

Manubens-Gil, D., Vegas-Regidor, J., Prodhomme, C., Mula-Valls, O., and Doblas-Reyes, F. J.: Seamless management of ensemble climate prediction experiments on HPC platforms, in: 2016 International Conference on High Performance Computing Simulation (HPCS), pp. 895–900, https://doi.org/10.1109/HPCSim.2016.7568429, 2016.

Marinescu, D. C.: Chapter 12 - Big Data, data streaming, and the mobile cloud, in: Cloud Computing (Third Edition), edited by Marinescu, D. C., pp. 453–500, Morgan Kaufmann, third edition edn., https://doi.org/https://doi.org/10.1016/B978-0-32-385277-7.00019-1, 2023.

Mastelini, S. M. and de Carvalho, A. C. P. d. L. F.: Using dynamical quantization to perform split attempts in online tree regressors, Pattern Recognition Letters, 145, 37–42, https://doi.org/10.1016/j.patrec.2021.01.033, 2021.

Mikkonen, S., Laine, M., Mäkelä, H. M., Gregow, H., Tuomenvirta, H., Lahtinen, M., and Laaksonen, A.: Trends in the average temperature in Finland, 1847–2013, Stochastic Environmental Research and Risk Assessment, 29, 1521–1529, https://doi.org/10.1007/s00477-014-0992-2, 2015.

Min, Y., Ahn, K., and Azizan, N.: One-Pass Learning via Bridging Orthogonal Gradient Descent and Recursive Least-Squares, 2022 IEEE 61st Conference on Decision and Control (CDC), pp. 4720–4725, https://doi.org/10.1109/CDC51059.2022.9992939, 2022.

Morgan, E. C., Lackner, M., Vogel, R. M., and Baise, L. G.: Probability distributions for offshore wind speeds, Energy Conversion and Management, 52, 15–26, https://doi.org/10.1016/j.enconman.2010.06.015, 2011.

Muthukrishnan, S.: Data streams: Algorithms and applications, Foundations and Trends in Theoretical Computer Science, 1, 117–236, https://doi.org/10.1561/0400000002, 2005.

NextGEMS: https://nextgems-h2020.eu/data-sets/.

Orr, H. G., Ekström, M., Charlton, M. B., Peat, K. L., and Fowler, H. J.: Using high-resolution climate change information in water management: A decision-makers' perspective, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 379, https://doi.org/10.1098/rsta.2020.0219, 2021.

Palmer, T.: Climate forecasting: Build high-resolution global climate models, Nature, 515, 338–339, https://doi.org/10.1038/515338a, 2014.

Pryor, S. C. and Barthelmie, R. J.: Climate change impacts on wind energy: A review, Renewable and Sustainable Energy Reviews, 14, 430–437, https://doi.org/10.1016/j.rser.2009.07.028, 2010.

Pörtner, H.-O., Roberts, D., Tignor, M., Poloczanska, E., Mintenbeck, K., Alegría, A., Craig, M., Langsdorf, S., Löschke, S., Möller, V., Okem, A., and Rama, B. e.: IPCC, 2022: Climate Change 2022: Impacts, Adaptation and Vulnerability. Contribution of Working Group II to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change, Technical Summary, Cambridge University Press, Cambridge, UK and New York, USA, https://doi.org/10.1017/9781009325844, 2022.

Rackow, T., Pedruzo-Bagazgoitia, X., Becker, T., Milinski, S., Sandu, I., Aguridan, R., Bechtold, P., Beyer, S., Bidlot, J., Boussetta, S., Deconinck, W., Diamantakis, M., Dueben, P., Dutra, E., Forbes, R., Ghosh, R., Goessling, H. F., Hadade, I., Hegewald, J., Jung, T., Keeley, S., Kluft, L., Koldunov, N., Koldunov, A., Kölling, T., Kousal, J., Kühnlein, C., Maciel, P., Mogensen, K., Quintino, T., Polichtchouk, I., Reuter, B., Sármány, D., Scholz, P., Sidorenko, D., Streffing, J., Sützl, B., Takasuka, D., Tietsche, S., Valentini, M., Vannière, B., Wedi, N., Zampieri, L., and Ziemen, F.: Multi-year simulations at kilometre scale with the Integrated Forecasting System coupled to FESOM2.5 and NEMOv3.4, Geoscientific Model Development, 18, 33–69, https://doi.org/10.5194/gmd-18-33-2025, 2025.

Rising, J., Tedesco, M., Piontek, F., and Stainforth, D. A.: The missing risks of climate change, Nature, 610, 643–651, https://doi.org/10.1038/s41586-022-05243-6, 2022.

Russo, S., Sillmann, J., and Fischer, E. M.: Top ten European heatwaves since 1950 and their occurrence in the coming decades, Environmental Research Letters, 10, https://doi.org/10.1088/1748-9326/10/12/124003, 2015.

Samaniego, L., Thober, S., Wanders, N., Pan, M., Rakovec, O., Sheffield, J., Wood, E. F., Prudhomme, C., Rees, G., Houghton-Carr, H., Fry, M., Smith, K., Watts, G., Hisdal, H., Estrela, T., Buontempo, C., Marx, A., and Kumar, R.: Hydrological forecasts and projections for improved decision-making in the water sector in europe, Bulletin of the American Meteorological Society, 100, 2451–2471, https://doi.org/10.1175/BAMS-D-17-0274.1, 2019.

Segura, H., Pedruzo-Bagazgoitia, X., Weiss, P., Müller, S. K., Rackow, T., Lee, J., Dolores-Tesillos, E., Benedict, I., Aengenheyster, M., Aguridan, R., Arduini, G., Baker, A. J., Bao, J., Bastin, S., Baulenas, E., Becker, T., Beyer, S., Bockelmann, H., Brüggemann, N., Brunner, L., Cheedela, S. K., Das, S., Denissen, J., Dragaud, I., Dziekan, P., Ekblom, M., Engels, J. F., Esch, M., Forbes, R., Frauen, C., Freischem, L., Garc´-Maroto, D., Geier, P., Gierz, P., González-Cervera, Á., Grayson, K., Griffith, M., Gutjahr, O., Haak, H., Hadade, I., Haslehner, K., ul Hasson, S., Hegewald, J., Kluft, L., Koldunov, A., Koldunov, N., Kölling, T., Koseki, S., Kosukhin, S., Kousal, J., Kuma, P., Kumar, A. U., Li, R., Maury, N., Meindl, M., Milinski, S., Mogensen, K., Niraula, B., Nowak, J., Praturi, D. S., Proske, U., Putrasahan, D., Redler, R., Santuy, D., Sármány, D., Schnur, R., Scholz, P., Sidorenko, D., Spät, D., Sützl, B., Takasuka, D., Tompkins, A., Uribe, A., Valentini, M., Veerman, M., Voigt, A., Warnau, S., Wachsmann, F., Wacławczyk, M., Wedi, N., Wieners, K.-H., Wille, J., Winkler, M., Wu, Y., Ziemen, F., Zimmermann, J., Bender, F. A.-M., Bojovic, D., Bony, S., Bordoni, S., Brehmer, P., Dengler, M., Dutra, E., Faye, S., Fischer, E., van Heerwaarden, C., Hohenegger, C., Järvinen, H., Jochum, M., Jung, T., Jungclaus, J. H., Keenlyside, N. S., Klocke, D., Konow, H., Klose, M., Malinowski, S., Martius, O., Mauritsen, T., Mellado, J. P., Mieslinger, T., Mohino, E., Pawłowska, H., Peters-von Gehlen, K., Sarré, A., Sobhani, P., Stier, P., Tuppi, L., Vidale, P. L., Sandu, I., and Stevens, B.: nextGEMS: entering the era of kilometer-scale Earth system modeling, EGUsphere, 2025, 1–39, https://doi.org/10.5194/egusphere-2025-509, 2025.

Seneviratne, S., Nicholls, N., Easterling, D., Goodess, C., Kanae, S., Kossin, J., Luo, Y., Marengo, J., McInnes, K., Rahimi, M., Reichstein, M., Sorteberg, A., Vera, C., and Zhang, X.: Changes in climate extremes and their impacts on the natural physical environment, in: Managing the Risks of Extreme Events and Disasters to Advance Climate Change Adaptation [Field, C.B., V. Barros, T.F. Stocker, D. Qin, D.J. Dokken, K.L. Ebi, M.D. Mastrandrea, K.J. Mach, G.-K. Plattner, S.K. Allen, M. Tignor, and P.M. Midgley (eds.)]. A Special Report of Working Groups I and II of the Intergovernmental Panel on Climate Change (IPCC), pp. 109–230, Cambridge University Press, Cambridge, UK, and New York, NY, USA, https://doi.org/10.1017/CBO9781139177245.006, 2012.

Shi, H., Dong, Z., Xiao, N., and Huang, Q.: Wind Speed Distributions Used in Wind Energy Assessment: A Review, Frontiers in Energy Research, 9, 1–14, https://doi.org/10.3389/fenrg.2021.769920, 2021.

Staffell, I. and Pfenninger, S.: The increasing impact of weather on electricity supply and demand, Energy, 145, 65–78, https://doi.org/10.1016/j.energy.2017.12.051, 2018.

Stevens, B., Acquistapace, C., Hansen, A., Heinze, R., Klinger, C., Klocke, D., Rybka, H., Schubotz, W., Windmiller, J., Adamidis, P., Arka, I., Barlakas, V., Biercamp, J., Brueck, M., Brune, S., Buehler, S. A., Burkhardt, U., Cioni, G., Costa-Surós, M., Crewell, S., Crüger, T., Deneke, H., Friederichs, P., Henken, C. C., Hohenegger, C., Jacob, M., Jakub, F., Kalthoff, N., Köhler, M., van LAAR, T. W., Li, P., Löhnert, U., Macke, A., Madenach, N., Mayer, B., Nam, C., Naumann, A. K., Peters, K., Poll, S., Quaas, J., Röber, N., Rochetin, N., Scheck, L., Schemann, V., Schnitt, S., Seifert, A., Senf, F., Shapkalijevski, M., Simmer, C., Singh, S., Sourdeval, O., Spickermann, D., Strandgren, J., Tessiot, O., Vercauteren, N., Vial, J., Voigt, A., and Zängl, G.: The added value of large-eddy and storm-resolving models

for simulating clouds and precipitation, Journal of the Meteorological Society of Japan, 98, 395–435, https://doi.org/10.2151/jmsj.2020-021, 2020.

Teutschbein, C. and Seibert, J.: Bias correction of regional climate model simulations for hydrological climate-change impact studies: Review and evaluation of different methods, Journal of Hydrology, 456-457, 12–29, https://doi.org/10.1016/j.jhydrol.2012.05.052, 2012.

Thober, S., Kumar, R., Wanders, N., Marx, A., Pan, M., Rakovec, O., Samaniego, L., Sheffield, J., Wood, E. F., and Zink, M.: Multi-model ensemble projections of European river floods and high flows at 1.5, 2, and 3 degrees global warming, Environmental Research Letters, 13, https://doi.org/10.1088/1748-9326/aa9e35, 2018.

Wang, J., Hu, J., and Ma, K.: Wind speed probability distribution estimation and wind energy assessment, Renewable and Sustainable Energy Reviews, 60, 881–899, https://doi.org/10.1016/j.rser.2016.01.057, 2016.

Wieners, K.-H., Rackow, T., Aguridan, R., Becker, T., Beyer, S., Cheedela, S. K., Dreier, N.-A., Engels, J. F., Esch, M., Frauen, C., Klocke, D., Kölling, T., Pedruzo-Bagazgoitia, X., Putrasahan, D., Sidorenko, D., Schnur, R., Stevens, B., and Zimmermann, J.: https://doi.org/https://www.wdc-climate.de/ui/entry?acronym=nextGEMS_prod, 2024.