Data-Centric Engineering (2023), 4: e11
doi:10.1017/dce.2023.4

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

Uniform-in-phase-space data selection with iterative normalizing flows

Malik Hassanaly[1], Bruce A. Perry[1], Michael E. Mueller[1,2] and Shashank Yellapantula[1]

[1]Computational Science Center, National Renewable Energy Laboratory, Golden, CO, USA
[2]Mechanical and Aerospace Engineering, Princeton University, Princeton, NJ, USA
Corresponding author: Malik Hassanaly; E-mail: malik.hassanaly@gmail.com

Received: 26 August 2022; **Revised:** 22 February 2023; **Accepted:** 27 February 2023

Keywords: Data reduction; instance selection; normalizing flows

Abstract

Improvements in computational and experimental capabilities are rapidly increasing the amount of scientific data that are routinely generated. In applications that are constrained by memory and computational intensity, excessively large datasets may hinder scientific discovery, making data reduction a critical component of data-driven methods. Datasets are growing in two directions: the number of data points and their dimensionality. Whereas dimension reduction typically aims at describing each data sample on lower-dimensional space, the focus here is on reducing the number of data points. A strategy is proposed to select data points such that they uniformly span the phase-space of the data. The algorithm proposed relies on estimating the probability map of the data and using it to construct an acceptance probability. An iterative method is used to accurately estimate the probability of the rare data points when only a small subset of the dataset is used to construct the probability map. Instead of binning the phase-space to estimate the probability map, its functional form is approximated with a normalizing flow. Therefore, the method naturally extends to high-dimensional datasets. The proposed framework is demonstrated as a viable pathway to enable data-efficient machine learning when abundant data are available.

Impact Statement

Uniform-in-phase-space sampling is achieved by selecting data points according to an acceptance probability constructed with the probability density function (PDF) of the data. Here, the functional form of the PDF (rather than a discretized form) is computed via normalizing flows, which can easily handle high-dimensional configurations. With large number of instances, the data PDF can be computed iteratively by first roughly estimating the PDF and then correcting it. This strategy efficiently learns the data PDF, even in presence of large number of instances. The method is beneficial for ML tasks on a memory and computational budget. The sampling strategy allows constructing more robust models in the sense, where the trained models perform well in a wide variety of situations.

1. Introduction

Advances in high-performance computing (HPC) and experimental diagnostics have led to the generation of ever-growing datasets. Scientific applications have since successfully embraced the era of Big Data

This research article was awarded Open Data and Open Materials badges for transparent practices. See the Data Availability Statement for details.

(Baker et al., 2019; Duraisamy et al., 2019; Brunton et al., 2020) but are now struggling to handle very large datasets. For example, numerical simulations of combustion or fusion can generate hundreds of petabytes per day, thereby heavily stressing storage and network resources (Klasky et al., 2021). Likewise, experimental facilities generate measurements with ever-higher spatial and temporal resolution (Barwey et al., 2019; Stöhr et al., 2019). In this context, data reduction is becoming a critical part of scientific machine learning (ML; Peterka et al., 2020).

A scientific dataset can be thought of as an element of $\mathbb{R}^{N\times D}$, where N is the number of data points and D is the dimension of a data point. Data reduction can first be achieved by recognizing that, although the data are D-dimensional, it may in effect lie on a low-dimensional manifold of size $d \ll D$. With an appropriate d-dimensional decomposition, the full dataset will be well approximated. This approach is commonly termed *dimension reduction* and has received extensive attention (Sirovich, 1987; Hinton and Salakhutdinov, 2006; Oseledets, 2011; Brunton et al., 2020). The dimension reduction strategy is also at the heart of projection-based reduced-order models (Gouasmi et al., 2017; Akram et al., 2022).

Another approach consists of discarding some data points from the dataset, that is, representing the dataset with $n \ll N$ data points, which is commonly termed *instance selection* (IS; Jankowski and Grochowski, 2004), *data-sampling* (Woodring et al., 2011; Biswas et al., 2018), or *data-pruning* (Saseendran et al., 2019). Other data compression methods such as tensor-decomposition do not distinguish between data dimensions and adopt a holistic approach (Ballard et al., 2020). Although there are well-established concepts that ensure that minimal information is lost during *dimension reduction*, data reconstruction may still be necessary when disseminating data or performing scientific tasks. For example, dimension reduction techniques alone do not ease the visualization of large datasets. In contrast, IS does not require reconstruction and naturally eases scientific tasks. Note that if data reconstruction is not needed, IS can be used in conjunction with dimension reduction for more aggressive data reduction. In this paper, a new method for IS is proposed for large N and large D.

IS can be an attractive method for improving the accuracy of a model that uses a given dataset. For example, in the case of classification tasks, if one class is over-represented compared to another (class imbalance), it may be advantageous to remove some data points from the majority class. Several techniques have been successful, including random removal of majority class samples (Leevy et al., 2018). More advanced techniques have proposed removing noisy data points (instance editing; Wilson, 1972; Tomek, 1976; Angelova et al., 2005), removing data points located away from the decision boundary (instance condensation; Hart, 1968), or using a hybrid of both approaches (Batista et al., 2004). Alternatively, the pruned dataset can be constructed iteratively by testing its classification accuracy (Skalak, 1994; García and Herrera, 2009; López et al., 2014). These approaches are particularly well suited for nearest-neighbor classification, which exhibits degrading performance with increasing training dataset size (Wilson and Martinez, 2000; Garcia et al., 2012). In natural language processing, IS has also been shown to be useful for improving the accuracy of predictive models (Mudrakarta et al., 2018). Despite the existence of a wide variety of approaches, the aforementioned pruning techniques often do not scale linearly with the number of data points in the original dataset and may not be appropriate to address Big Data in scientific applications (Jankowski and Grochowski, 2004; Triguero et al., 2016).

In some situations, IS may simply be unavoidable—for instance, when so much data are being generated that some needs be discarded due to storage constraints. In particular, although parallel visualization tools are available, they may be cumbersome and intensive for computing and network resources. Besides, for very large datasets, input/output time may become the bottleneck of numerical simulations (Woodring et al., 2011). To address this issue, researchers have proposed various methods for spatially downselecting data points while maintaining the relevant visual features. Woodring et al. (2011) proposed a stratified sampling method for astronomical particle data such that the down-selected dataset maintains the original spatial distribution of particles. A variant of adaptive mesh refinement for data was also proposed to guide downsampling, where the refinement criterion is

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

user-defined (Nouanesengsy et al., 2014). For the visualization of scatter plots, a perceptual loss was proposed to assess whether the scatter plot of downselected data was consistent with the original dataset (Park et al., 2016). Methods similar to the one proposed here have been proposed in the past (Biswas et al., 2018; Rapp et al., 2019), but either do not scale with the number of dimensions (D) or instances (N). Rapp et al. (2019) proposed a method to display a multidimensional scatter plot by transforming the distribution into an arbitrary one. However, this method may not scale well with the dimensionality of the data, as it requires a neighbor search for the density estimation. In addition, the authors show that the method can become unreasonably slow for very large datasets. In the same vein, Biswas et al. (2018) proposed performing spatial downsampling by achieving uniform sampling of scalar quantities. They first estimate the probability density function (PDF) of the scalar values and then use it to downselect data points. The method was later extended to include the scalar gradients to ensure visual smoothness (Biswas et al., 2020). This technique requires binning the phase-space to construct the PDF, which poses two main issues. First, the choice of the number of bins can affect the quality of the results (Biswas et al., 2020). Second, binning high-dimensional spaces can become intractable in terms of computational intensity and memory, which makes extension to uniform sampling in higher scalar dimensions nontrivial (Dutta et al., 2019).

Besides visualization, IS may also be needed for other scientific tasks in power and memory-constrained environments, which may become ubiquitous with the rise of the Internet of Things (Raman and Hassanaly, 2019). In a practical edge computing case, data may simply be randomly discarded to reduce memory and power requirements. This is especially true for streaming data, as newer data are typically retained, whereas old data are discarded to adapt to drift in the observations (Hulten et al., 2001; Krawczyk and Woźniak, 2015; Ramírez-Gallego et al., 2017). In such cases, the IS method adopted is critical for the performance of the device trained with the streaming data (Du et al., 2014). Finally, memory requirements that make IS unavoidable do not solely exist for edge devices, but can also occur in supercomputing environments. Consider the example of ML applied to turbulence modeling. Typically, high-fidelity data—such as direct numerical simulation (DNS)—is used to train ML-based closure models. For this application, each location in physical space is a new data point. Given that a single DNS snapshot nowadays comprises $O(10^9)$ data points, multiple snapshots can easily amount to $O(10^{11})$ training data points. With this number of data points, training time may become unreasonable, and data may need to be discarded. Data are often randomly discarded, but more elaborate methods have recently been proposed, such as clustering the data and randomly selecting data from each cluster (Lloyd, 1982; Nguyen et al., 2021; Yellapantula et al., 2021). This technique enables the efficient elimination of redundant data points from quiescent or otherwise unimportant regions when the phenomenon of interest is intermittent, effectively attempting an approximate phase-space sampling like the one proposed in the present work. However, IS via clustering is sensitive to the choice of the number of clusters, and there is no systematic way of deciding on an appropriate number of clusters (Barwey et al., 2019; Barwey et al., 2020; Hassanaly et al., 2021b). Furthermore, in practice, clustering tends to underrepresent rare data points (see Appendix A). In separate studies, researchers have proposed clustering the dataset and training a separate model for each cluster. This technique is beneficial not only for a priori (Barwey et al., 2021) but also a posteriori analyses (Nguyen et al., 2021), suggesting that a proper downsampling technique could help with the deployment of ML models.

In this work, a novel method for IS in large and high-dimensional datasets is proposed. Redundant samples are pruned based on their similarity in the original D-dimensional space. This is achieved by first estimating a probability map of the original dataset and then using it to uniformly sample phase-space. The proposed solution estimates the probability map with a normalizing flow, which has been shown to perform well in high-dimensional systems. In addition, an iterative method is proposed for the probability map estimation, thereby efficiently and accurately estimating the probability map without processing the full dataset. As such, the computational cost may be drastically reduced compared to methods that process the full dataset. The proposed algorithm is described at length in Section 2. In Section 3, the uniformity of the pruned dataset is assessed and the scaling of the

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

computational cost with respect to the number of data points and their dimension is evaluated. The effect of the uniform sampling on ML tasks is also compared to other sampling methods in Section 4. Section 5 provides conclusions and perspectives for extending the proposed approach.

2. Instance Selection Method

2.1. Problem statement and justification

Consider a dataset $\mathcal{X} \in \mathbb{R}^{N \times D}$, where N is the number of instances (or data points) and D is the number of dimensions (or features) of the dataset. The objective is to create a smaller dataset $\mathcal{Y} \in \mathbb{R}^{n \times D}$, where $n \ll N$, and where every instance of $\mathcal{Y}$ is included in $\mathcal{X}$. In other words, $\mathcal{Y}$ does not contain new data points. In addition, during the downselection process, the redundant samples should be pruned first. Redundancy is defined as proximity in the D-dimensional phase-space. The objective is that all regions of phase-space should have the same density of data points, that is, the distribution of $\mathcal{Y}$ should be as uniform as possible in phase-space.

Compared to a random sampling approach, uniform sampling of the phase-space emphasizes rare events in the reduced dataset. In contrast, random downsampling will almost always discard rare events from the dataset. The motivation behind preserving rare data points is twofold. First, scientific discovery often stems from rare events. For example, state transitions may occur rarely, but observing them is critical for prediction, causality inference, and control (Barwey et al., 2019; Hassanaly and Raman, 2021). If a dataset is reduced without appropriate treatment, scientific interpretation may be hindered. Second, to ensure that data-driven models are robust, they need to be exposed to a dataset that is as diverse as possible. Although the average training error may only be mildly affected, the model may still incur large errors in rare parts of phase-space. In some applications, accurate prediction of rare events is crucial when the model is deployed. For example, in non-premixed rotating detonation engines, detonations are heavily influenced by triple points and fuel mixing ahead of the detonation (Rankin et al., 2017; Barwey et al., 2021; Sato et al., 2021), which are very localized phenomena. For other combustion applications, the lack of data in some parts of phase-space has even required the addition of synthetic data (Wan et al., 2020). The proposed sampling objective has also been noted to maximize the information content of the subsampled dataset (Biswas et al., 2018).

2.2. Method overview

To perform the downsampling task, the first step is to estimate the PDF of the dataset in phase-space. Once this probability map has been constructed, it is used to construct an acceptance probability. The acceptance probability determines whether a given data point should be selected or not. In practice, the acceptance probability is used as follows. For each data point, a random number is drawn from the uniform distribution $\mathcal{U}(0,1)$. If this number falls below the acceptance probability, the point is integrated into the downsampled dataset. The acceptance probability varies for each data point and depends on the likelihood of observing such data. In regions of high data density, the acceptance probability is low, but in regions of low data density (rare regions), the acceptance probability is high. The link between data PDF and the appropriate acceptance probability is illustrated hereafter for a simple example.

Consider two events, A and B, observed, respectively, a and b times, where $b < a$. Suppose that one wants to select c samples such that $c \leq 2b < a + b$. In that case, the acceptance probability $s(A)$ of every event A (or $s(B)$ of every event B) should be such that the expected number of event A (or B) in the downsampled dataset is $c/2$. In other words, $s(A) \times a = c/2$ (or $s(B) \times b = c/2$). Therefore, the acceptance probability should be proportional to the inverse of the PDF, with the proportionality factor controlled by the desired size of the downsampled set. The specific data points selected depend on the random number drawn from $\mathcal{U}(0,1)$ for each of the data points. This is a source of randomness in the proposed algorithm that is studied in Section 4.

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

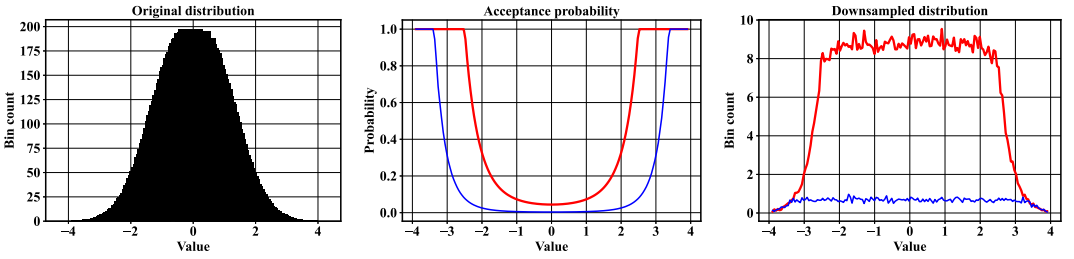

Figure 1. *Illustration of the proposed method for a canonical example. (Left) Histogram of the original distribution. (Middle) Acceptance probability plotted against the random variable value when downsampling to* $n = 100$ *data points (—) and* $n = 1{,}000$ *data points (—). (Right) Distribution of the downsampled dataset when downsampling to* $n = 100$ *data points (—) and* $n = 1{,}000$ *data points (—).*

In the event, where $2b < c < a + b$, all B events should be selected. Although the resulting downsampled dataset is imbalanced, it is as close as possible to a uniform distribution. To accommodate for this case, the acceptance probability $s(x)$ for the data point x can be simply constructed as

$$s(x) = min\left(\frac{\alpha}{p(x)}, 1\right), \tag{1}$$

where x is an observation, $p(x)$ is the PDF of the dataset evaluated in x, and α is a proportionality factor constant for the whole dataset constructed such that the expected number of selected data points equals c, that is,

$$\sum_{i=1}^{N} s(x_i) = c, \tag{2}$$

where x_i is the ith observation and c is the desired number of samples. Using a constant proportionality parameter α for the full dataset ensures that the relative importance of every event is maintained. The method extends to continuous distributions, as only the local PDF values are needed to construct the acceptance probability. Figure 1 shows an application of the method for selecting 100 and 1,000 data points from 10,000 samples distributed according to a standard normal distribution $\mathcal{N}(0,1)$. Here, the exact PDF is used to construct the acceptance probability. Then, the acceptance probability is adjusted to satisfy Eq. (2). The downsampled distributions are obtained by repeating the sampling algorithm 100 times. As can be seen in the figure, the distribution of samples approaches a uniform distribution in the high-probability region. For a large number of samples, the tail of the PDF for the downselected dataset is underrepresented due to the low data availability. In those regions, the acceptance probability is maximal and all the samples are selected. Note that to obtain exactly 1,000 samples (compared to 100 samples), a larger part of the dataset needs to be selected with maximal acceptance probability. Despite achieving uniform sampling where possible, the method cannot ensure a perfect balance between all events in the dataset. Because the acceptance probability is clipped at unity (i.e., the method does not create new data), rare events can be underrepresented in the reduced dataset if c is large. The larger c is the more imbalance can be expected in the dataset. Addressing the imbalance in the reduced dataset would require generating new data points and is out of the scope of this paper. In the case where uniform sampling needs to be achieved, reducing the number of data points in the final dataset will help extending the volume of phase-space over which data are uniformly distributed. This effect can be observed in Figure 1 (right) where using a lower n value extends the range over which uniform sampling is achieved.

2.3. Practical implementation

Practical implementation of the method requires overcoming several numerical challenges, which are outlined below. In particular, because the objective is to tackle the issues associated with very large

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

datasets, scalability with the number of instances and dimensions is necessary. A predictor procedure and an predictor–corrector procedure are proposed.

2.3.1. Probability map estimation

The main hurdle to overcome is the estimation of the PDF of the dataset in phase-space, which is ultimately used to compute the acceptance probability. Different solutions have been proposed, including kernel methods (Rapp et al., 2019) and binning (Biswas et al., 2018). Although the binning strategy is scalable with respect to the number of data points, the quality of the estimation may be dependent on the number of bins used (Biswas et al., 2020). Kernel methods offer a more continuous description of the PDF but tend to generate spurious modes in high-dimension Wang and Scott (2019). In addition, the PDF may be sensitive to the choice of the smoothing parameter Tabak and Turner (2013). Finally, as local density value requires knowledge of the relative locations of all data points, specific treatments are needed such as using compact kernels, but in this case, the neighborhood of every point needs to be identified, which can be problematic with many dimensions (Rapp et al., 2019).

To estimate the density of the dataset, it is proposed to use normalizing flows, which are primarily applied to generative modeling (Dinh et al., 2015, 2016; Kingma et al., 2016; Papamakarios et al., 2017; Durkan et al., 2019; Müller et al., 2019) and are increasingly being used for scientific applications (Bothmann et al., 2020; Verheyen and Stienen, 2020). Given a dataset, generative modeling attempts to create new data points that were previously unseen but are distributed like the original dataset. Some notable applications of generative modeling include semi-supervised learning (Salimans et al., 2016), sampling of high-dimensional PDFs (Hassanaly et al., 2022a, 2022b), and closure modeling (Bode et al., 2021). Normalizing flows are types of neural networks that generate new data by simultaneously learning the PDF of the dataset and generating data points with maximum likelihood (or minimum negative log-likelihood). The likelihood of the data is computed by using a transformation G to map from a known random variable z onto a random variable x that is distributed like the dataset. The likelihood of the generated data can be computed as

$$p(x) = p(z)\det\left(\frac{dG^{-1}}{dx}\right). \tag{3}$$

The peculiarity of normalizing flows lies in their attempt to directly evaluate the right-hand side of Eq. (3) allowing to evaluate $p(x)$. Normalizing flows build a mapping G such that $\det\left(\frac{dG^{-1}}{dx}\right)$ is tractable, even in high dimensions. In general, evaluating the determinant in high dimensions scales at least quadratically with the matrix dimension. Normalizing flows adopt structures that make the determinant of the Jacobian of the inverse of G easy to compute. For instance, if the Jacobian is triangular, the cost of the determinant calculation scales linearly with data dimension (Dinh et al., 2015, 2016). The attention given to scaling with data dimension has allowed normalizing flows to be used for image generation, which shows that they can learn PDFs in a $\sim 10^3$-dimensional phase-space. Recent types of normalizing flows typically attempt to find a balance between neural network expressiveness and the tractability of the right-hand side of Eq. (3) (Dinh et al., 2015, 2016; Durkan et al., 2019). Among them, neural spline flows are chosen here, as they have been shown to capture complex multimodal distributions with a small number of trainable parameters (Durkan et al., 2019). Importantly, the method proposed here may be easily adapted to future other density estimation methods that may tackle high-dimensions or handle large numbers of data points more efficiently.

2.3.2. Predictor algorithm

Instead of using a normalizing flow to generate new samples, it is proposed here to leverage its density estimation capabilities to construct an appropriate acceptance probability. Given N instances of the

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

dataset, training a normalizing flow requires processing all N instances at every epoch. This approach is computationally expensive and therefore unacceptable, especially when the ultimate goal is to downselect data to efficiently develop an ML-based model. It is proposed to train a normalizing flow on a random subset of the data, thereby greatly accelerating the PDF estimation process. The algorithm is summarized in Algorithm 1.

Algorithm 1. Predictor instance selection of n data points.

1: Shuffle dataset $\mathcal{X} \in \mathbb{R}^{N \times d}$.
2: Create a working dataset $\xi \in \mathbb{R}^{M \times d}$, where $M \ll N$, by randomly probing $\mathcal{X}$.

Compute acceptance probability s

3: Train a normalizing flow with working dataset to learn the probability map p. (Step 1)
4: **for** $x_i \in \mathcal{X}$ **do**
5: Evaluate $p(x_i)$ and acceptance probability $S(x_i) = 1/p(x_i)$. (Step 2a)
6: **end for**

Select n **data points (Step 2b)**

7: Find α such that $\sum_{i=1}^{N} min(max(\alpha s(x_i), 0), 1) = n$.
8: numberSelectedDataPoints $= 0$
9: **for** $i = 1, N$ **do**
10: Draw a random number r from the uniform distribution $\mathcal{U}(0,1)$.
11: **if** $r < min(max(\alpha s(x_i), 0), 1)$ and numberSelectedDataPoints $< n$ **then**
12: Append the data point to the selected instances.
13: numberSelectedDataPoints $+ = 1$.
14: **end if**
15: **end for**

The outcome of the base algorithm is illustrated in Figure 2. The dataset considered here is a bivariate normal distribution of mean $[1,1]$ and diagonal covariance matrix $diag([1,2])$ that contains 10^6 samples. For computational efficiency, the probability map is learned with $M = 10^5$ data points that are randomly selected (sensitivity to the choice of M is assessed in Section 3.5). A scatter plot of the randomly selected data is shown in Figure 2 for 10^3 and 10^4 selected data points. Compared to a random sampling method (left), Algorithm 1 provides better coverage of phase-space, no matter the value of n. However, a nonuniform distribution of the data points can be observed, especially in low-probability regions. This problem is addressed in the next section.

2.3.3. Accommodating for rare observations: the predictor–corrector algorithm

The downside of accelerating the normalizing flow calculation by using a random subsample of the dataset when learning its PDF is that the probability of rare events is not well approximated, simply because too few rare events are shown to the normalizing flow. However, one can recognize that the wrongly downsampled data shown in Figure 2 is an image of the error in the density estimation obtained from the normalizing flow. By computing the density of the downsampled data itself, one can apply a correction to the original density estimate such that, if Algorithm 1 is used again, the resulting data distribution is close to being uniform. The underlying idea that motivates the algorithm is that is it easier to learn a correction to the density than the full density itself. Similar methods have been developed in the context of multifidelity methods (De et al., 2023). The predictor–corrector is summarized in Algorithm 2. In the predictor–corrector algorithm, the normalizing flow is trained *nFlowIter* times, where $nFlowIter \in \mathbb{N}^* - \{1\}$. The procedure is used on the same dataset as in Section 2.3.2, and the results are shown in Figure 3.

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

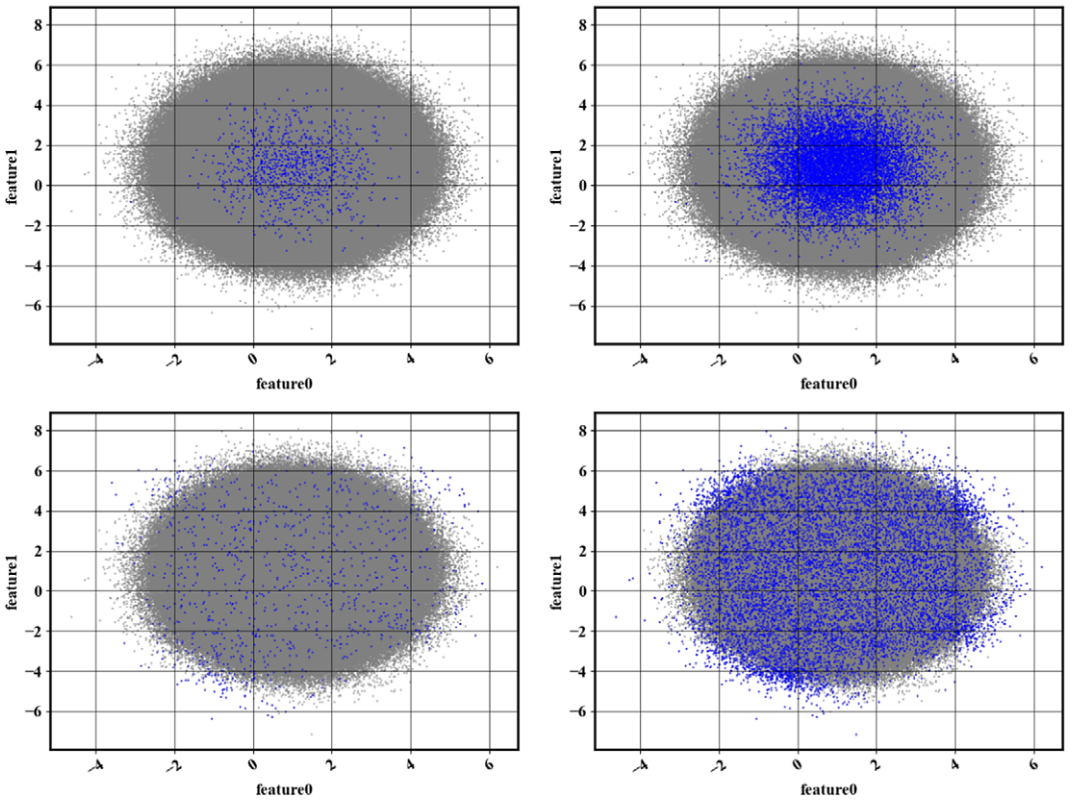

Figure 2. *Illustration of the results obtained with Algorithm 1. The full dataset (•) is overlaid with the reduced dataset (•). (Top left) Random selection of $n = 10^3$ data points. (Bottom left) Result of Algorithm 1 with $n = 10^3$. (Top right) Random selection of $n = 10^4$ data points. (Bottom right) Result of Algorithm 1 with $n = 10^4$.*

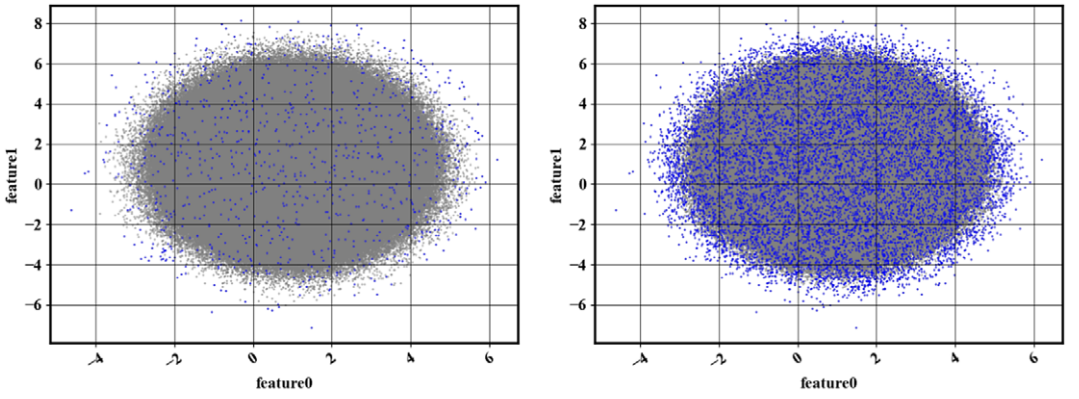

Figure 3. *(Left) Result of Algorithm 2 with $n = 10^3$. (Right) Result of Algorithm 2 with $n = 10^4$.*

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

Algorithm 2. Predictor–corrector instance selection of n data points.

1: Shuffle dataset $\mathcal{X} \in \mathbb{R}^{N \times d}$.
2: Create a working dataset $\xi \in \mathbb{R}^{M \times d}$, where $M \ll N$, by randomly probing $\mathcal{X}$.
3: **for** flowIter $= 1,$nFlowIter **do**
4: Compute acceptance probability $s_{flowIter}$ (lines 3–6 of Algorithm 1).
5: **if** flowIter==1 **then**
6: Select M data points (lines 7–15 of Algorithm 1).
7: Replace working dataset with downselected dataset
8: **else**
9: $s_{flowIter} = s_{flowIter} \times s_{flowIter-1}$.
10: **if** flowIter==nFlowIter **then**
11: Select n data points (lines 7–15 of Algorithm 1).
12: **else**
13: Select M data points (lines 7–15 of Algorithm 1).
14: Replace working dataset with downselected dataset
15: **end if**
16: **end if**
17: **end for**

For the specific case considered here, because the exact PDF is known, the true acceptance probability can be computed and compared to the sampling probability obtained at every iteration of the algorithm. Figure 4 shows the relative error in the acceptance probability after different numbers of iterations. It can be seen that, as expected, iterating decreases errors for rare points (high acceptance probabilities). In addition, although errors decrease after the first iteration, they remain stagnant. Error stagnation can be explained by the fact that iterating is not expected to decrease the modeling errors of the normalizing flow —only the statistical errors associated with estimating rare event probabilities.

Finally, some important properties for the predictor–corrector algorithm are described hereafter. First, after every iteration, the loss function (the negative log-likelihood) of the converged normalizing flow should increase. This can be understood by computing the exact negative log-likelihood of a normal distribution with increasing standard deviation (left-hand side panel of Figure 5). As the standard deviation increases, the negative log-likelihood increases as well. Compared to random sampling, the first iteration of the method should spread the data distribution. If that is the case, the second iteration of

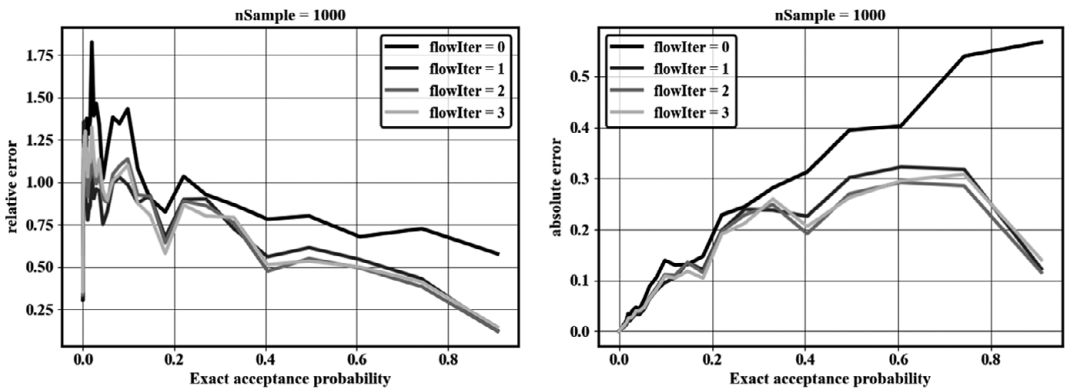

Figure 4. *Effect of iterations on accuracy of acceptance probability. (Left) Conditional mean of relative error in the acceptance probability plotted against the exact acceptance probability. (Right) Conditional mean of absolute error in the acceptance probability plotted against the exact acceptance probability.*

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

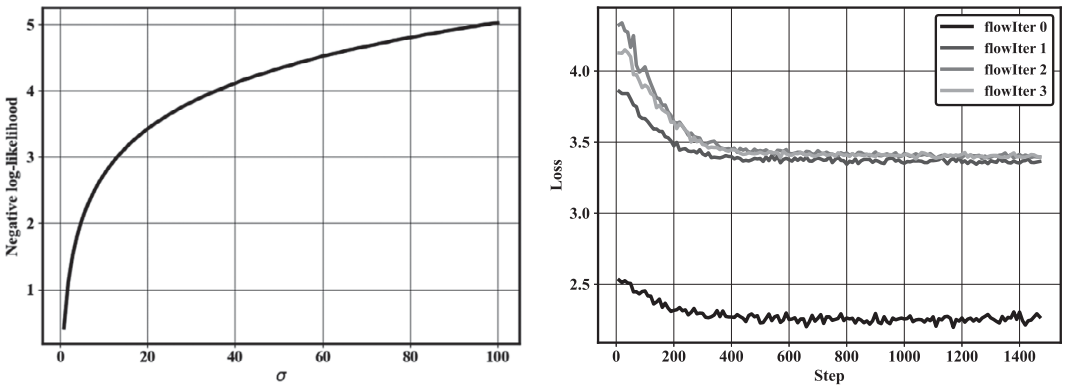

Figure 5. *(Left) Negative log-likelihood against standard deviation for a standard normal distribution. (Right) History of the negative log-likelihood loss of the normalizing flow trained at each flow iteration.*

the predictor–corrector algorithm should result in a loss (negative log-likelihood) larger than at the first iteration. This behavior can be observed for the bivariate normal case (right-hand side panel of Figure 5).

Second, the PDF corrected with the iterative procedure will not approach the true PDF, but rather the PDF needed to obtain a uniform sampling of the data. This principle can be understood with a simple bimodal case such as the one in Section 2.2. In the case where $c > 2b$, the downsampled distribution simply cannot be exactly uniform. While iterating, even with an exact estimation of the data density at the first iteration, the density correction at the next iteration will be nonzero, because a uniform distribution of data is not possible (see discussion at the end of Section 2.2). However, the rescaling factor α will ensure that the density estimate corrections do not affect the downsampled distribution, which is what matters here.

2.3.4. Sampling probability adjustment

When computing the proportionality factor α that satisfies Eq. (2), it is necessary to perform summations and probability clipping (Eq. (1)) over the entire dataset. When there are a large number of instances (N), the cost of this step may approach the cost of density estimation. However, α can be computed with only a subset of all the acceptance probabilities. To see this, recall that the normalizing constant α must satisfy $\sum_{i=1}^{N} \alpha s(x_i) = n$, where x_i is the ith data point. Since α does not depend on the data point considered, the problem simplifies to estimating $\frac{1}{N}\sum_{i=1}^{N} s(x_i)$. Since the data points are shuffled, the choice of the first N' data points where $N' \ll N$ can be considered as N' random draw of the random variable $s(x)$. Using the central limit theorem, $\frac{1}{N'}\sum_{i=1}^{N'} s(x_i)$ is a sample of a normal distribution centered on $\frac{1}{N}\sum_{i=1}^{N} s(x_i)$ and with a standard deviation $\sigma/\sqrt{N'}$, where σ is the standard deviation of the random variable $s(x)$. The probability of making a large error in the estimation of $\frac{1}{N}\sum_{i=1}^{N} s(x_i)$ depends on σ and the number of samples N'. The decision on the value of N' is therefore case dependent. Note, however, that since $\forall i, 0 < s(x_i) < 1$, $\sigma < 1/2$ following Popoviciu's inequality. Therefore, the number of required N' to achieve a given accuracy can be computed a priori and does not increase with N. Using $N' \ll N$ acceptance probability values, the constraint used to compute α becomes

$$\frac{N}{N'}\sum_{i=1}^{N'} s(x_i) = c. \tag{4}$$

With this approximation, and the ability to compute the probability map with a subset of the full dataset, the only task of Algorithm 2 that scales with the number of data points is the evaluation of the probability with the normalizing flow. In Section 3.6, a parallel strategy is described to further reduce the cost of the procedure.

During the selection process—when the acceptance probability is used to decide which data point to keep in the downsampled dataset—the order of the data points can matter. For instance, if the most rare data points

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

are seen first, then those data points will be downselected before a sufficient amount of high-probability points are observed. To avoid this issue, it is crucial to randomly shuffle the dataset (first step of Algorithms 1 and 2). Finally, since the selection process is probabilistic, the probability adjustment only guarantees that n data points is the expected number of data points to be selected. In practice, the number of data points selected may vary. If more than n data points are selected, the first n data points can be kept. If less than n data points are selected, then another pass through the data can be done to complete the dataset.

3. Numerical Experiments

In this section, the proposed method is tested on a real scientific dataset. The sensitivities of the proposed algorithm to the main hyperparameters (number of flow iterations and size of dataset chosen to train the normalizing flow) are assessed. The performance of the algorithm in high dimensions is evaluated. The computational cost of the method and the efficiency of parallelization strategies are studied.

3.1. Dataset

The dataset considered here is the one used by Yellapantula et al. (2021) in a turbulent combustion modeling application. In particular, the data was used to learn a model for the filtered dissipation rate of a reaction progress variable C in large eddy simulations of turbulent premixed combustion systems. Dissipation rate is an unclosed term in the transport equation of the progress variable variance $\widetilde{C''^2}$, which is useful to estimate the filtered progress variable source term. The dataset was constructed by aggregating the data from multiple DNS datasets under different operating conditions. For computational tractability, the DNS was reduced with a stratified sampling approach, but it still exhibits a multimodal aspect, as can be seen in Figure 6. The features of the dataset are $\left\{\widetilde{C}, \widetilde{C''^2}, \widetilde{\chi}_{res}, \alpha, \beta, \gamma\right\}$, where $\widetilde{C}$ is a filtered progress variable, $\widetilde{C''^2}$ is the subfilter variance of the progress variable, $\widetilde{\chi}_{res} = 2D_C\left|\nabla\widetilde{C}\right|^2$ is the resolved dissipation rate, D_C is the progress variable diffusivity, and α, β, and γ are the principal rates of strain. Additional details about the definitions of the dimensions are available in Yellapantula et al. (2021). For every dimension D considered, the phase-space contains the first D features mentioned above. The number of data points is $N = 8 \times 10^6$. Each physical feature is rescaled in the dataset. In the present work, the distribution of the data points is altered in the downsampled dataset. However, this alteration does not affect the progress variable statistics since they are assembled prior to the downsampling process. An illustration of the two-dimensional version of the dataset is shown in Figure 6.

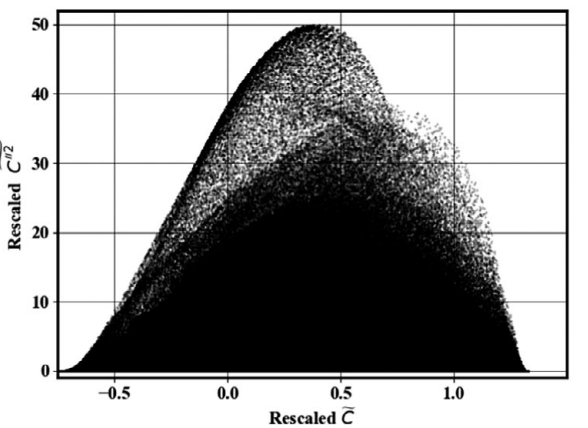

Figure 6. *Scatter plot of the full dataset with $D = 2$.*

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

3.2. Quantitative criterion

3.2.1. Definition

Evaluation of the performance of Algorithm 2 was done visually in Section 2, which is possible in at most two dimensions. To quantitatively evaluate the quality of the downsampled dataset, an intuitive approach is to ensure that the distance of every data point to its closest neighbor is large. This method emphasizes coverage of the phase-space and evaluates whether there exist clusters of data points. In practice, the average distance of each point to its closest neighbor is used, and the distance criterion is expressed as

$$\text{distance criterion} = \frac{1}{n}\sum_{i=1}^{n} \|x_i - x_{ni}\|_2, \tag{5}$$

where x_i is the ith data point of the reduced dataset and x_{ni} is the nearest neighbor of the ith data point. This metric is referred to as the *distance criterion* in the rest of the manuscript.

3.2.2. Suitability of the distance metric

Starting from an arbitrarily distributed dataset, the objective of the method described in Section 2 is to downselect data points to obtain a uniform distribution in phase-space. In this section, the focus is on assessing whether the samples are distributed as expected. For a two-dimensional phase-space, a visual assessment can be used as in Sections 2.3.2 and 2.3.3. In higher dimensions, such an assessment is not possible, and quantitative metrics become necessary. Here, it is proposed to compute the average distance of every data point to its closest neighbor. To make distances along each dimension equivalent, the metric is computed on a rescaled dataset such that each dimension spans the interval $[-4,4]$. The suitability of the metric is assessed with a canonical bivariate normal distribution $\mathcal{N}([0,0], diag([0.1,0.1]))$. The starting dataset comprises 10^4 samples and is reduced to 10^2 in the final dataset. Three strategies for selecting the reduced dataset are compared to evaluate whether the criterion chosen coincides with a uniform distribution of data. A random set of samples is shown in Figure 7 (left) and results in a distance criterion of 0.1894. The result of a brute force search in the set of 10^2 data points that maximizes the criterion is shown in Figure 7 (middle). The brute force search is conducted by repeating a random selection of 10^2

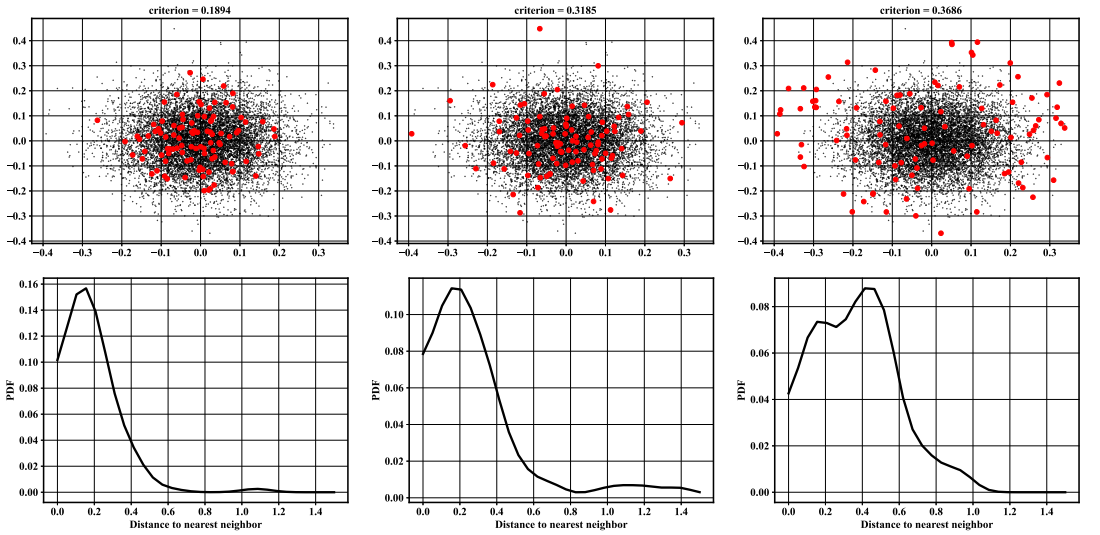

Figure 7. *(Top) Comparison of selection schemes for 100 samples (•) out of* 10^5 *data points distributed according to* $\mathcal{N}([0,0], diag([0.1,0.1]))$ *(•). (Bottom) PDF of the distance to nearest neighbor in the downselected dataset. (Left) Random selection scheme. (Middle) Brute force optimization of distance criterion. (Right) Algorithm 2.*

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

samples at each iteration (10^5 iterations are used) and selecting the set that maximizes the distance criterion. This procedure results in a distance criterion of 0.3185, and the set of downselected points obtained offers better coverage of phase-space than the random search. Samples obtained from Algorithm 2 (Figure 7, right) give a distance criterion of 0.3686. There again, the criterion improved, which qualitatively coincides with a uniform distribution of the downselected points. In this section, it is not claimed that Algorithm 2 converges to the dataset that optimizes the distance criterion value. However, it is claimed that improving the distance criterion coincides with uniformly distributing the data points and that the performance of Algorithm 2 can be assessed quantitatively using the proposed criterion.

3.3. Effect of the number of flow iterations

The proposed distance criterion is used to illustrate the effect of the number of iterations. The results are shown in Table 1. The mean and standard deviation of the distance criterion values obtained with five independent runs using $M = 10^5$ are shown. Sampling quality dramatically increases after the first iteration and then stagnates. This result is in line with Figure 4, where a conditional error on the sampling probability was shown to not improve after two iterations. A consequence of this result is that Algorithm 2 increases the computational cost only twofold. Another valuable result is that the standard deviation of the distance criterion decreases when using more than one iteration, which suggests that iterating improves the robustness of the method.

The results were compared with random IS and stratified sampling (Nguyen et al., 2021; Yellapantula et al., 2021), where each stratum is a cluster obtained from the k-means algorithm (Lloyd, 1982). Different numbers of clusters were tested in the range $[20, 160]$. The best results are shown in Table 1 and were obtained with 40 clusters. Overall, Algorithms 1 and 2 are superior to random and stratified sampling. Visual inspection of the results (see Appendix A) confirms that using more than two iterations does not help with the sampling quality and that the sampling quality of the proposed method exceeds that of the random and stratified sampling. Appendix B shows the application of Algorithm 2 to a different dataset. Two iterations appears to also suffice in this case, which suggests that the aforementioned findings might be applicable to a wide variety of datasets.

Table 2 shows the results when using a binning strategy for the computation of the PDF, as well as the method of Biswas et al. (2018), where the PDF is computed with all the available data with a binning

Table 1. *Dependence of the distance criterion value on the number of iterations and instances selected (n).*

Normalizing flow	Algorithm 1	Algorithm 2 (2 iterations)	Algorithm 2 (3 iterations)	Random	Stratified
$n = 1{,}000$	0.064 ± 0.014	$\mathbf{0.098 \pm 0.003}$	0.096 ± 0.003	0.046	0.067
$n = 10{,}000$	0.022 ± 0.0033	$\mathbf{0.030 \pm 0.0002}$	0.029 ± 0.0006	0.014	0.021

Note. Larger distance criterion value is better. Normalizing flow is used for the PDF estimation. Bold entries denote best performances.

Table 2. *Dependence of the distance criterion value on the number of iterations and instances selected (n).*

Binning	Algorithm 1	Algorithm 2 (2 iterations)	Algorithm 2 (3 iterations)	Biswas et al. (2018)
$n = 1{,}000$	0.069 ± 0.0005	0.091 ± 0.0099	$\mathbf{0.099 \pm 0.0023}$	0.097 ± 0.0011
$n = 10{,}000$	0.019 ± 0.0001	$\mathbf{0.030 \pm 0.0002}$	0.030 ± 0.0003	$\mathbf{0.031 \pm 0.0002}$

Note. Larger distance criterion value is better. Binning is used for the PDF estimation. Bold entries denote best performances.

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

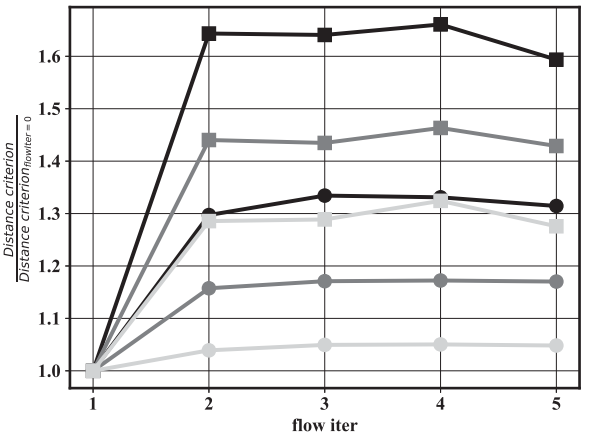

Figure 8. *Sensitivity of distance criterion to the number of flow iterations, normalized by the distance criterion after the first flow iteration. Symbols represent the mean distance criterion. Results are shown for $D = 2$ (-●-) and $D = 4$ (-■-), for $n = 10^3$ (—), $n = 10^4$ (—), and $n = 10^5$ (—).*

strategy. Since the full dataset is used compute the PDF, no iterative procedure is needed. The sampling is done by discretizing each dimension with 100 equidistant bins. The binning strategy with $M = 10^5$ does exhibit similar results as the one with normalizing flow. The iterative procedure does improve the computation of the PDF which demonstrate that the predictor–corrector strategy is applicable to other density estimation methods. Finally, the method of Biswas et al. (2018) does perform as well as the proposed method, albeit requiring to compute the density with the full dataset. Visualization of the effect of iteration with the binning strategy are shown in Appendix A.

In high dimensions, the same conclusions can be obtained. Figure 8 shows the average distance criterion obtained over five independent runs, for D varying in the set $\{2,4\}$ and n varying in the set $\{10^3, 10^4, 10^5\}$. The distance criterion is normalized by the value reached at the first iteration to quantify the benefit of iterating. For $D = 2$ and $D = 4$, the quality of the downsampled dataset dramatically improves after two iterations and then ceases to improve. It also appears that iterating mostly benefits aggressive data reduction (small n) and cases with more dimensions.

3.4. Sampling quality in high dimension

Here, the distance criterion is used to assess the sampling quality in higher dimensions. The distance criterion obtained with Algorithm 2 is shown for different phase-space dimensions using $M = 10^5$. Random sampling and stratified sampling are used to compare the obtained sampling quality. The stratified sampling procedure was performed with the optimal number of clusters found in Section 3.3. The sampling results are shown in Table 3.

As can be seen in Table 3, the proposed sampling technique also performs better than traditional methods in more than two dimensions. To further verify this conclusion, a two-dimensional projection of the data along every dimension pair is shown for the three methods for $D = 4$ and $n = 10{,}000$ in Figure 9. Visually, the proposed method better covers the phase-space. The data points selected by Algorithm 2 cover the phase-space differently than when only the first two features are considered (Figure 9, right). This observation can be explained by the other two-dimensional projections; variations in $\widetilde{\chi}_{res}$, α, and β in the dataset are mostly observed at low $\widetilde{C''^2}$ values.

Similar to the two dimension case, Table 4 shows that the binning strategy also benefits from the predictor–corrector algorithm, which further suggests that Algorithm 2 can benefit density estimation methods in general. For the case $D = 3$, the binning strategy with $M = 10^5$ outperforms normalizing flows. After further investigation, it appeared that using $M = 4 \times 10^5$ significantly improved the case

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

Table 3. *Distance criterion values for different numbers of instances selected (n) and data dimensions (D).*

Norm. flow	Algorithm 1	Algorithm 2 ($M = 10^5$, 2 iterations)	Random	Stratified
$n = 1{,}000$ $D = 3$	0.094 ± 0.0161	0.109 ± 0.0132 (**0.136 ± 0.0032**)	0.065 ± 0.0012	0.095 ± 0.0030
$n = 1{,}000$ $D = 4$	0.159 ± 0.0267	**0.223 ± 0.0129**	0.106 ± 0.0048	0.137 ± 0.0034
$n = 1{,}000$ $D = 5$	0.282 ± 0.0244	**0.330 ± 0.0036**	0.133 ± 0.0031	0.188 ± 0.0064
$n = 10{,}000$ $D = 3$	0.034 ± 0.0083	0.041 ± 0.0044 (**0.057 ± 0.0006**)	0.026 ± 0.0003	0.038 ± 0.0005
$n = 10{,}000$ $D = 4$	0.084 ± 0.0028	**0.098 ± 0.0064**	0.049 ± 0.0004	0.065 ± 0.0014
$n = 10{,}000$ $D = 5$	0.166 ± 0.0073	**0.173 ± 0.0061**	0.068 ± 0.0009	0.094 ± 0.0059

Note. Larger distance criterion value is better. Normalizing flow is used for the PDF estimation. Data in parenthesis are results with $M = 4 \times 10^5$. Bold entries denote best performances.

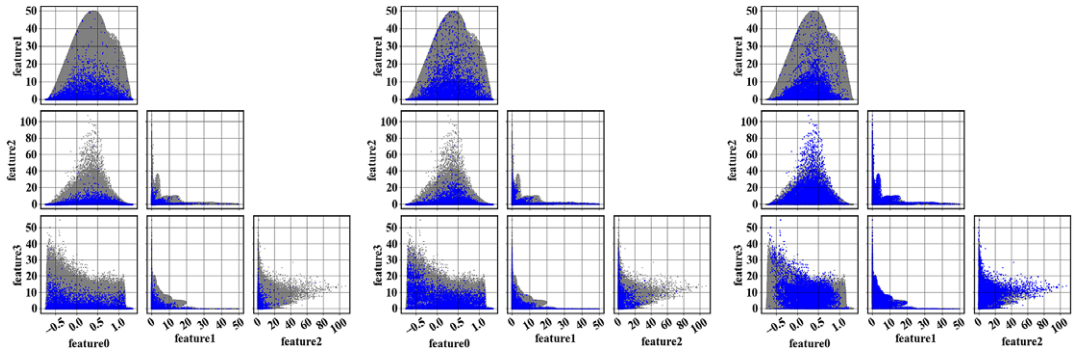

Figure 9. *Two-dimensional projection of $n = 10{,}000$ selected data points (•) out of an original four-dimensional dataset (·). (Left) Random sampling. (Middle) Stratified sampling. (Right) Algorithm 2.*

Table 4. *Distance criterion values for different numbers of instances selected (n) and data dimensions (D).*

Binning	Algorithm 1	Algorithm 2 ($M = 10^5$, 2 iterations)	Biswas et al. (2018)
$n = 1{,}000$ $D = 3$	0.124 ± 0.0038	0.133 ± 0.0021	**0.151 ± 0.0015**
$n = 1{,}000$ $D = 4$	0.183 ± 0.0023	0.206 ± 0.0004	**0.233 ± 0.0019**
$n = 1{,}000$ $D = 5$	Out of memory	Out of memory	Out of memory
$n = 10{,}000$ $D = 3$	0.047 ± 0.0001	0.046 ± 0.001	**0.063 ± 0.0003**
$n = 10{,}000$ $D = 4$	0.083 ± 0.0005	0.096 ± 0.0006	**0.112 ± 0.0004**
$n = 10{,}000$ $D = 5$	Out of memory	Out of memory	Out of memory

Note. Larger distance criterion value is better. Binning is used for the PDF estimation. Bold entries denote best performances.

$D = 3$ with normalizing flow (the distance criterion values are shown in parenthesis in Table 3). The sensitivity analysis with M shown in Figure 10 also suggests that the case $D = 3$ gives poor results with $M < 4 \times 10^5$. Note that for other higher dimensions, similar considerations also apply but have a lesser impact than for $D = 3$. The method of Biswas et al. (2018) outperforms the present method for $D < 5$, at

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

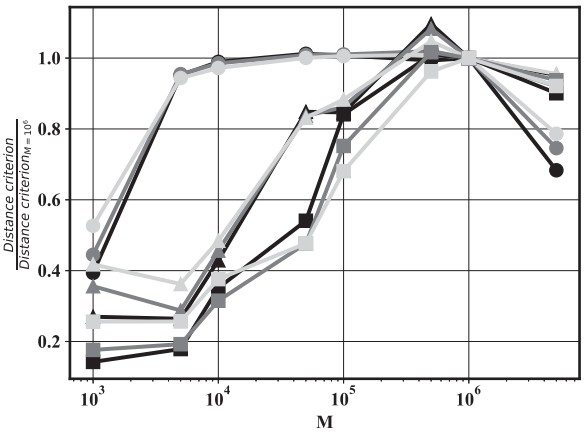

Figure 10. *Sensitivity of distance criterion to M normalized by the distance criterion at* $M = 10^6$. *Symbols represent the mean distance criterion. Results are shown for* $D = 2$ *(-●-),* $D = 3$ *(-▲-), and* $D = 4$ *(-■-), for* $n = 10^3$ *(—),* $n = 10^4$ *(—), and* $n = 10^5$ *(—).*

the expense of higher memory requirements and computing the PDF with the full dataset. For $D \geq 5$, all binning strategies fail due to too large memory requirements.

3.5. Data size limits

To construct the probability map efficiently, it is proposed in Section 2 training a normalizing flow on a subset of the full dataset. In addition, to account for rare realizations, it was proposed to estimate the probability map iteratively (Section 2.3.3). In this section, it is investigated how large the data subset needs to be to still obtain a uniform-in-phase-space reduced -dataset. In other words, the sensitivity of Algorithm 2 to M is evaluated. The dataset presented in Section 2.3.2 is considered. Algorithm 2 is used with two iterations with a feature space of dimension D in the set $\{2,3,4\}$. The original dataset ($N = 8 \times 10^6$) is downsampled to n data points in the set $\{10^3, 10^4, 10^5\}$. For each case, M is varied in the set $\{10^3, 5\times10^3, 10^4, 5\times10^4, 10^5, 5\times10^5, 10^6, 5\times10^6\}$. Each data reduction is performed five times, and the ensemble averages of the distance criterion are shown in Figure 10. The number of training steps is fixed at 12,000 (the number of epochs varies between 6 and 12,000) to ensure that larger M values do not result in worse convergence of the normalizing flow.

Figure 10 shows that Algorithm 2 does not favor small or high n values and that varying M equally affects cases with small or large n. The value of M only affects the accuracy of the probability map, which has equal implications for all sizes of downsampled data.

In the limit case where M is small, the distance criterion is, across the board, lower than the optimal value. This can be understood by considering an asymptotic scenario where $M = 1$. In that case, the best possible probability map will be a delta distribution centered on the data point used. At the next flow iteration, the probability map will be corrected with another delta distribution. A very small value of M inherently limits the expressiveness of the normalizing flow, that is, the complexity of the probability map that can be approximated, thereby adversely affecting the reduced dataset. M needs to be sufficiently high to formulate a first reasonable approximation of the probability map that can be accurately corrected at the next flow iteration.

As M increases, the distance criterion ceases to improve, which confirms that a subset of the full dataset can be used to train the normalizing flow. The higher the phase-space dimension D, the higher the smallest M value required to achieve uniform-in-phase-space sampling. As the dimension increases, the probability map becomes more difficult to approximate, requiring a larger dataset size.

For very large M (close to the size of the full dataset N), the performance of the algorithm degrades across the board. This effect can be explained by again considering the asymptotic limit ($M = N$). In the

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

ideal case, the probability map is already approximated at the first iteration. At the second iteration, a correction is applied if the true probability map is not uniform, which is the case for most datasets.

If the optimal M value depends on the complexity of the probability map to be approximated, it also depends on the particular dataset considered. Because the performances of the method vary smoothly with M, it would be possible, in practice, to try multiple M values until an optimum is reached.

3.6. Computational cost

Compared to random or stratified sampling, Algorithm 2 is more expensive, as a probability needs to be estimated and evaluated for every data point. The computational cost of the method and its scale-up with the number of data points and dimensions is described here.

The computational cost of the proposed method can be divided into three separate components. These components (referred to as steps) are the building blocks of Algorithm 1, which is itself the building block of Algorithm 2. First, the predictive model for the probability of the data points as a function of the location in phase-space is trained (Step 1). Second, the predictive model is evaluated for each data point in the dataset (Step 2a). Third, the acceptance probability for each data point is constructed, and the data points of the reduced dataset are selected (Step 2b). The computational cost study is done with the dataset D_1 (Lapointe et al., 2015; Savard and Blanquart, 2017) used in Yellapantula et al. (2021), which is different than the dataset considered in Section 3.1 in that it contains more data points ($N = 4.8 \times 10^7$). The larger number of data points allows to clearly distinguish regimes where the cost of Step 1 or Step 2a dominates. The scaling with respect to N is obtained with $D = 2$. The scaling with respect to D is obtained with $N = 4.8 \times 10^7$. The breakdown of the computational cost of Algorithm 2 is shown in Figure 11 as a function of N (Figure 11, left) and D (Figure 11, right).

The computational cost of Step 1 mainly depends on the number of data points used to train the model. As seen in Section 2, a subsample of the full dataset can be used to train the model. No matter how many data points are in the original dataset, the predictor–corrector algorithm (Algorithm 2) allows the model to be efficiently trained with a small number of data points. Therefore, as can be observed in Figure 11 (left), the computational cost of Step 1 does not depend on the total number of data points N. The complexity of Step 1 is difficult to rigorously define as it depends of the convergence of the optimization. Typically, it will scale linearly with M if the number of epochs is kept constant for different M values. The number of dimensions D influences the number of parameters in the network as the input space and the output space depend on D. In turn, the training time linearly varies with the number of trainable parameters. In the present case, the number of parameters increased as $D^{1/4}$. In our case, the time complexity of Step 1 can be estimated as $O\left(MD^{1/4}\right)$.

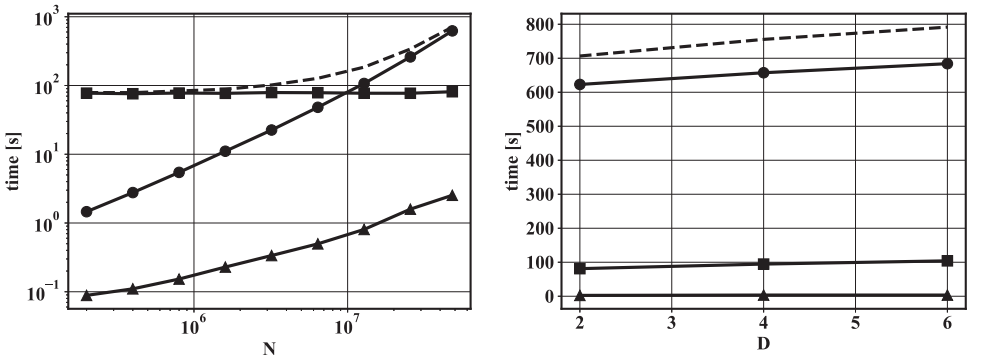

Figure 11. *Scaling of computational cost against the number of data points N in the full dataset (left) and the dimension D (right). The computational cost is divided into the probability map estimation (-■-), called Step 1; the construction of acceptance probability (-●-), called Step 2a; and the data selection (-▲-), called Step 2b. The total computational cost is also indicated for both scaling plots (- -).*

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

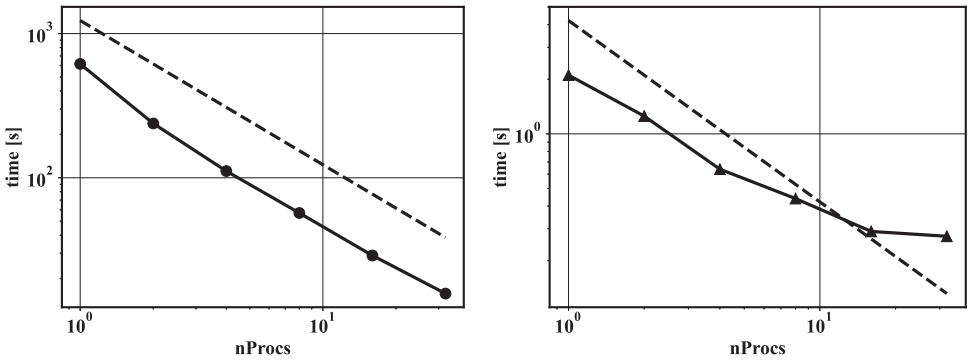

Figure 12. *(Left) Scaling of Step 2a (-●-) overlaid with linear scaling (- -). (Right) Weak scaling of Step 2b (-●-) overlaid with linear scaling (- -).*

The computational cost of Step 2a linearly scales with N as the probability model needs to be evaluated for every data point. The computational cost also linearly scales with the number of trainable parameters used in the normalizing flow as each probability evaluation requires a neural-network inference. The complexity of Step 2a can be estimated as $O\left(ND^{1/4}\right)$.

Finally, Step 2b scales as $O(N)$, but its contribution to the total computational cost is negligible compared with Step 2a.

In Figure 11, it can be observed that for large datasets, the total cost mainly depends on Step 2a. However, since Step 2a solely consists of probability density evaluations, it is embarrassingly parallelizable. With a simple MPI-parallelization, the computational time of Step 2a can be considerably decreased. The weak scaling plot of Step 2a is shown in Figure 12. Here, memory requirements are driven by the data points stored in memory, rather than by the probability density model (as would be the case if one stored it as a high-dimensional histogram). Therefore, the MPI-parallelization also decreases the memory requirements, which is beneficial for very large datasets. In addition, although Step 2b (adjustment of the acceptance probability to reach the desired data reduction) is not embarrassingly parallel, the global summation of acceptance probability can easily be parallelized by partitioning the data (see Figure 12, right). The parallelized implementation of Steps 2a and 2b is also made available in the companion repository.

In conclusion, the cost of the procedure linearly scales with the number of data points and can be significantly driven down if executed on a parallel architecture. Steps 2a and 2b can be accelerated via MPI-parallelization. Step 1 can be accelerated by a judicious choice of the number of data points used to train the normalizing flow. In addition, because Step 1 consists of training a neural net, it can benefit from GPU-acceleration for large batches or networks (which is also implemented in the companion repository).

4. Application: Data-Efficient ML

In this section, the effect of the uniform selection of data in phase-space is assessed for the training of data-driven models. This data selection method is compared to random sampling and stratified sampling on a realistic scientific dataset. The input–output mapping in this dataset is noisy because the input dimensions are insufficient to perfectly model the output. Thus, a synthetic dataset where the noise level can be controlled is considered to understand the resilience of the method to noise.

4.1. Example of closure modeling

Typically, combustion problems require solving partial differential equations for the transport of species mass fractions or some representative manifold variables, such as the progress variable (Pierce and Moin, 2004). In large eddy simulations of turbulent combustion problems, a closure model is required for the filtered reaction rate of such variables. One strategy consists of using data available via DNS to formulate

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

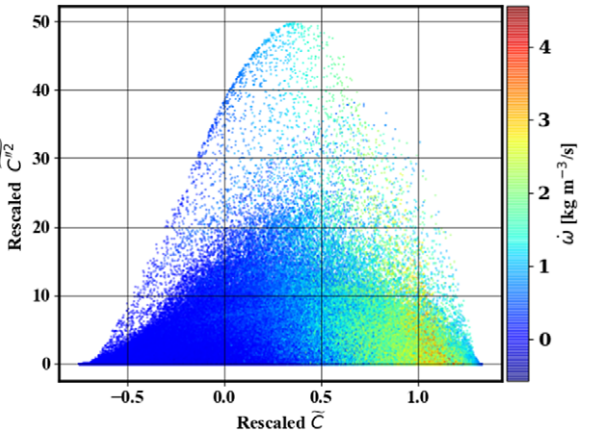

Figure 13. *Illustration of the combustion dataset. The dots are colored by progress variable source term $\dot{\omega}$.*

a data-driven model for the unclosed terms. For example, in premixed flames, it may be useful to transport a filtered progress variable $\widetilde{C}$ and its variance $\widetilde{C''^2}$, and to model the source term of the progress variable transport equation $\dot{\omega}$ as a function of $\widetilde{C}$ and $\widetilde{C''^2}$. The functional form of the model would be obtained from an existing dataset. In this section, the objective is to learn such a model. The model is trained on the reduced versions of the dataset used in Section 3.4. The full dataset is illustrated in Figure 13. From this figure, it is apparent that the output quantity is a noisy function of the inputs, indicating that any model formulated will be affected by aleatoric errors.

The IS algorithm proposed in Section 2 can, in principle, be applied for any data-driven model. Two approaches are used here: a Gaussian process (Pedregosa et al., 2011) and a feed-forward neural network (Abadi et al., 2016) with two hidden layers of 32 units each. The learning rate is initially set at 10^{-1} and exponentially decays to 10^{-3}. The batch size is fixed at 100 and the number of epochs is 400. The Gaussian process kernel consists of white noise superimposed with a radial basis function (RBF), where the length scale of the RBF and the standard deviation of the noise are optimized at every realization. The length scale of the RBF is optimized in the interval $[10^{-3}, 1]$ and the noise level is optimized in the interval $[10^{-9}, 10^{5}]$. The results shown can be reproduced with the code available in the companion repository. The original dataset contains about $N = 8 \times 10^6$ data points. The model training is done on reduced datasets that contain $n = 1{,}000$ or $n = 10{,}000$ data points. The data reduction is done with either Algorithm 2 ($M = 10^5$ and two flow iterations), random sampling, or stratified sampling. For computational tractability of the Gaussian process, only cases with $n = 1{,}000$ are shown. For every case, the sampling and training are done five times to collect the metric statistics. Two metrics are used here: the mean absolute error and the maximal error over the full dataset (N data points).

The training results for the Gaussian process and the feed-forward neural network are shown in Figure 14. Despite covering the phase-space better than the stratified and random sampling, the maximal error obtained by using Algorithm 2 is only slightly better than other methods (about 1% improvement compared to stratified sampling and 2% compared to random sampling). Furthermore, the average absolute error obtained via Algorithm 2 can be slightly larger (by about 6%) than when using plain random sampling.

To further investigate how the effect of phase-space coverage on the training results, the conditional error obtained with the different sampling procedures is plotted against the sampling probability computed for the $n = 10{,}000$ for five training realizations with all three sampling and training techniques (Figure 15). As a reminder, sampling probability is inversely proportional to data density; thus, low sampling probabilities denote high data density regions, and high sampling probabilities denote rare events. For all cases, Algorithm 2 induces the lowest errors in rare regions of phase-space (high sampling probability). Stratified sampling, which increases the occurrence of rare events compared with random

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

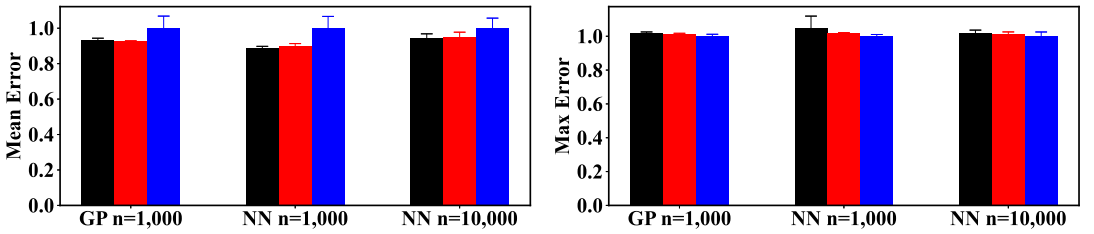

Figure 14. *Statistics of the mean (left) and maximal error (right) obtained over five repetitions of downselection and training of a neural network and Gaussian process on the combustion dataset for random sampling (—), stratified sampling (—), and Algorithm 2 (—). Bar height shows ensemble mean and error bar shows standard deviation. Bar and error bar size are rescaled by the ensemble mean obtained with Algorithm 2.*

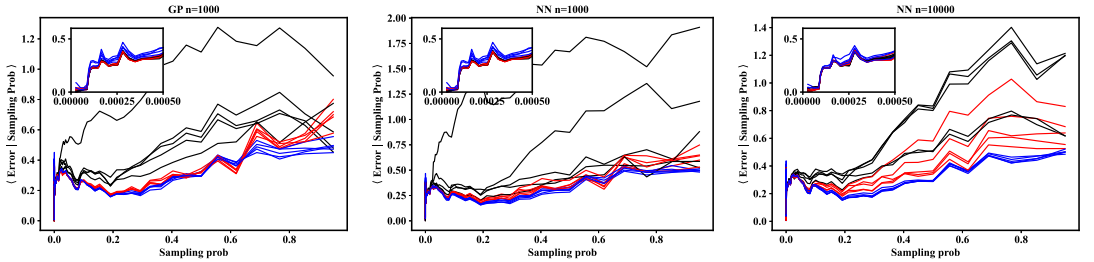

Figure 15. *Prediction errors over the full dataset conditioned on sampling probability, calculated with $n = 10^4$ for five independent datasets constructed with random sampling (—), stratified sampling (—), and Algorithm 2 (—). Inset zooms in on low sampling probability. (Left) Gaussian process with $n = 10^3$. (Middle) Feed-forward neural network with $n = 10^3$. (Right) Feed-forward neural network with $n = 10^4$.*

sampling, also reduces errors compared to random sampling, albeit to a smaller extent than Algorithm 2. However, in high-probability regions of phase-space (small sampling probability), Algorithm 2 induces the largest errors (see inset), especially for low values of n. Interestingly, Algorithm 2 also leads to the least amount of variation when repeating training runs, which indicates robustness to randomness in the construction of the dataset. For metrics such as the mean-squared error, errors in high-probability regions of phase-space dominate, which explains the poor performance of Algorithm 2 observed in Figure 14.

Another explanation for the marginal improvement shown in Figure 14 is the presence of noise. The core assumption of Algorithm 2 is that data points close in phase-space are redundant. However, in presence of noise, redundancy is necessary to correctly approximate the optimal estimator that is being sought. The lower performance of Algorithm 2 in high-probability regions compared to random and stratified sampling indicates that the high-probability region of phase-space is also noisy, and thereby benefits from methods that oversample data. The effect of noise is investigated in the next section.

4.2. Synthetic dataset

To better understand the effect of noise on the performance of Algorithm 2, a synthetic dataset is considered. The variables of the phase-space are kept the same, but the variable to learn is replaced by two-dimensional sinusoids shown in Figure 16. The advantage of the synthetic dataset is that one can control the amount of noise to evaluate its effect on the training results. Here, the two-dimensional sinusoids are superimposed with a randomly distributed noise of standard deviation ε in the set $\{0,1,2,3,4\}$. The functional form of the data is

$$10\cos\left(f_1\omega_{f_1}\right)\sin\left(f_2\omega_{f_2}\right)+\eta, \tag{6}$$

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

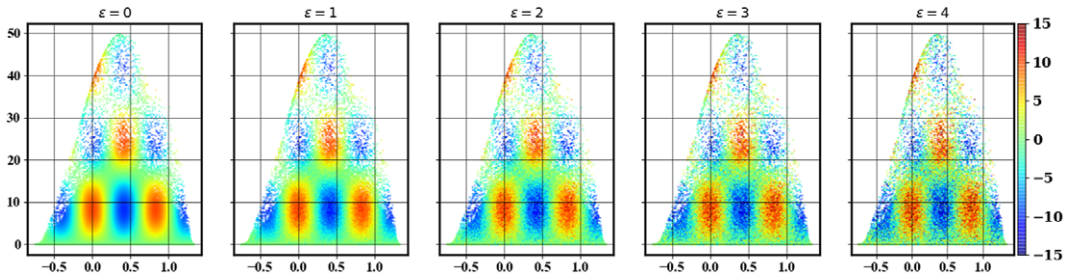

Figure 16. *Illustration of the synthetic datasets. Noise level increases from left to right.*

where f_1 denotes the first feature direction, f_2 denotes the second feature direction, ω_{f_1} is chosen such that 2.5 oscillation periods span the first feature direction, and ω_{f_2} is chosen such that 1.5 oscillation periods span the second feature direction, and η is the additive noise.

Figure 17 shows the results obtained with the Gaussian process. For different levels of noise, the optimal mean absolute error is $\frac{2}{\pi}\varepsilon$, and the optimal maximal error is obtained by computing the expected value of the maximum of a random normal variable out of N samples. In absence of noise, Algorithm 2 ($M = 10^5$, two iterations) offers the lowest maximal error and outperforms the other sampling methods by several orders of magnitude. In turn, the mean absolute error is also the lowest, albeit by a smaller margin than the maximal error. For the neural network case (Figure 18), the maximal error is lower by nearly an order of magnitude, but the average error is larger than the random and stratified sampling. This result also highlights how good coverage of the phase-space improves the robustness of the model. However, robustness is not well described by common metrics like mean absolute error, which is biased toward high-probability data points.

For the neural network and the Gaussian process, Algorithm 2 exhibits the lowest maximal absolute error (close to the optimal value) for low levels of noise ($0 < \varepsilon \leq 2$), but the mean absolute error is the largest out of all the methods. In particular, in the case of the Gaussian process, while the mean error was the lowest in absence of noise, it is the highest even at low noise levels. However, it should be noted that the mean error remains close to the optimal value with Algorithm 2, with the mean error penalty relative to the other approaches being small in comparison to the unavoidable error due to noise in the data. In contrast, the other sampling approaches can result in maximal absolute errors that are significantly larger than the optimal value. These results could be explained by the reduced ability of a uniform phase-space sampling to correctly estimate the conditional mean of a noisy variable. In other words, because uniform phase-space sampling eliminates redundancy by design, it cannot benefit from redundancy when estimating a conditional average when there is nonzero conditional variance. As the level of noise

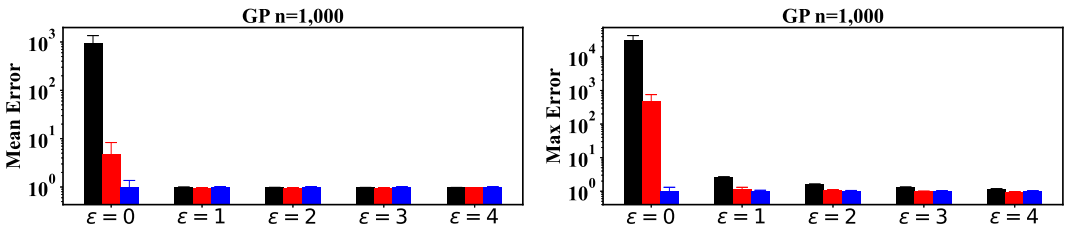

Figure 17. *Statistics of the mean (left) and maximal error (right) obtained over five repetitions of downselection and training of a Gaussian process on the synthetic dataset for random sampling (—), stratified sampling (—), and Algorithm 2 (—). Bar height shows ensemble mean and error bar shows standard deviation. Bar and error bar size are rescaled by the ensemble mean obtained with Algorithm 2.*

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

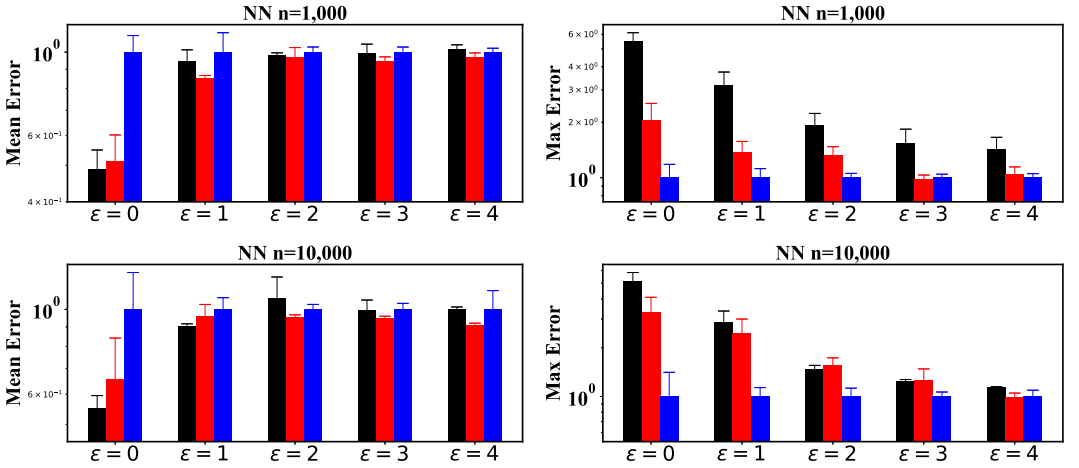

Figure 18. *Statistics of the mean (left) and maximal error (right) obtained over five repetitions of downselection and training of a neural network on the synthetic dataset for random sampling (—), stratified sampling (—), and Algorithm 2 (—). Bar height shows ensemble mean and error bar shows standard deviation. Bar and error bar size are rescaled by the ensemble mean obtained with Algorithm 2.*

increases, reducing statistical error in estimating the conditional average for high-probability regions becomes more beneficial than exploring new regions of phase-space to provide better predictions for rare points. This benefits random sampling over Algorithm 2, leading to the observed performance trends as noise levels increase (Figures 17 and 18).

This analysis confirms that the results shown in Figure 14 can also be explained by the presence of a low level of noise for $\dot{\omega}$. Although the maximal error is lowest with Algorithm 2, it is outperformed by other methods in terms of the mean absolute error because these methods emphasize accuracy for high-probability events and allow redundancy to counteract noise.

5. Conclusions

A novel data reduction scheme was introduced in the form of an IS method. The method enables uniformly covering the phase-space of a dataset with an arbitrarily low number of data points. Uniform sampling is implemented by constructing an acceptance probability for each data point that depends on its probability of appearing in the full dataset. To obtain the probability map of the dataset, a normalizing flow is trained on a subsample of the full dataset. Iterative estimation of the probability map allows efficient treatment of rare events in the dataset. The advantage of the present method lies in its efficient parameterization of the probability map, which holds even for high dimensions.

The IS method can serve several purposes, including efficient visualization, data dissemination, and data-efficient model training. For the latter, the performance of the IS method was compared to other common methods. In the absence of noise, uniform phase-space sampling offers better robustness. However, the lack of redundancy of data can induce error in approximating the optimal estimator of a variable if it is noisy.

Allowing targeted redundancy in the data could be implemented in the same framework by biasing the acceptance probability of the data points toward noisy regions of the phase-space. Likewise, a refined description of the phase-space could be obtained by allocating more points in regions of large gradients. These extensions are currently being pursued and will be investigated in future publications.

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

Acknowledgments. Useful discussions with Alex Gorodetsky (University of Michigan), Marc T. Henry de Frahan (NREL), Ryan N. King (NREL), Marc Day (NREL), Juliane Mueller (NREL), and Stephen Frank (NREL) are gratefully acknowledged. A preprint version is available at https://arxiv.org/pdf/2112.15446.pdf (Hassanaly et al., 2021a).

Author contribution. Conceptualization: M.H.; Data visualization: M.H., B.A.P.; Methodology: M.H., M.E.M.; Funding: S.Y.; Review and editing: all authors; Writing original draft: M.H. All authors approved the final submitted draft.

Competing interest. The authors declare no competing interests exist.

Data availability statement. An implementation of the method along with tutorials is available in a companion repository (https://github.com/NREL/Phase-space-sampling) under BSD-3 license. A subsample of the 2D combustion dataset (Section 3.1) is also available in the repository. Data were obtained from Buildings Data Hub funded by U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy's Building Technologies Office operated and maintained by Pacific Northwest National Laboratory at https://bbd.labworks.org.

Funding statement. This work was authored by the National Renewable Energy Laboratory (NREL), operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. This work was supported by the U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Vehicle Technologies Office (VTO). The research was performed using computational resources sponsored by the Department of Energy's Office of Energy Efficiency and Renewable Energy and located at the National Renewable Energy Laboratory. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

References

Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M., Kudlur M, Levenberg J, Monga R, Moore S, Murray DG, Steiner B, Tucker P, Vasudevan V, Warden P, Wicke M, Yu Y, Zheng X and Brain G (2016). Tensorflow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283.

Akram M, Hassanaly M and Raman V (2022) An approximate inertial manifold (AIM) based closure for turbulent flows. *AIP Advances 12*(7), 075118.

Angelova A, Abu-Mostafam Y and Perona P (2005) Pruning training sets for learning of object categories. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. *1*. New York, NY: IEEE, pp. 494–501.

Baker N, Alexander F, Bremer T, Hagberg A, Kevrekidis Y, Najm H, Parashar M, Patra A, Sethian J, Wild S, et al. (2019) Workshop report on basic research needs for scientific machine learning: Core technologies for artificial intelligence. Technical report, DOE Office of Science (SC).

Ballard G, Klinvex A and Kolda TG (2020) TuckerMPI: A parallel C++/MPI software package for large-scale data compression via the tucker tensor decomposition. *ACM Transactions on Mathematical Software (TOMS) 46*(2), 1–31.

Barwey S, Ganesh H, Hassanaly M, Raman V and Ceccio S (2020) Data-based analysis of multimodal partial cavity shedding dynamics. *Experiments in Fluids 61*(4), 1–21.

Barwey S, Hassanaly M, An Q, Raman V and Steinberg A (2019) Experimental data-based reduced-order model for analysis and prediction of flame transition in gas turbine combustors. *Combustion Theory and Modelling 23*(6), 994–1020.

Barwey S, Prakash S, Hassanaly M and Raman V (2021) Data-driven classification and modeling of combustion regimes in detonation waves. *Flow, Turbulence and Combustion 106*(4), 1065–1089.

Batista GE, Prati RC and Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter 6*(1), 20–29.

Biswas A, Dutta S, Lawrence E, Patchett J, Calhoun JC and Ahrens J (2020) Probabilistic data-driven sampling via multi-criteria importance analysis. *IEEE Transactions on Visualization and Computer Graphics 27*, 4439–4454.

Biswas A, Dutta S, Pulido J and Ahrens J (2018) In situ data-driven adaptive sampling for large-scale simulation data summarization. In *Proceedings of the Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization*, pp. 13–18.

Bode M, Gauding M, Lian Z, Denker D, Davidovic M, Kleinheinz K, Jitsev J and Pitsch H (2021) Using physics-informed enhanced super-resolution generative adversarial networks for subfilter modeling in turbulent reactive flows. *Proceedings of the Combustion Institute 38*, 2617–2625.

Bothmann E, Janßen T, Knobbe M, Schmale T and Schumann S (2020) Exploring phase space with neural importance sampling. *SciPost Physics 8*(4), 069.

Brunton SL, Noack BR and Koumoutsakos P (2020) Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics 52*, 477–508.

De S, **Reynolds M**, **Hassanaly M**, **King RN and Doostan A** (2023) Bi-fidelity modeling of uncertain and partially unknown systems using DeepONets. *Computational Mechanics*, 1–17.

Dinh L, **Krueger D and Bengio Y** (2015) NICE: Non-linear independent components estimation. In *International Conference on Learning Representations (ICLR)*.

Dinh L, **Sohl-Dickstein J and Bengio S** (2016). Density estimation using Real NVP. Preprint, arXiv:1605.08803.

Du L, **Song Q and Jia X** (2014) Detecting concept drift: An information entropy based method using an adaptive sliding window. *Intelligent Data Analysis 18*(3), 337–364.

Duraisamy K, **Iaccarino G and Xiao H** (2019) Turbulence modeling in the age of data. *Annual Review of Fluid Mechanics 51*, 357–377.

Durkan C, **Bekasov A**, **Murray I and Papamakarios G** (2019) Neural spline flows. *Advances in Neural Information Processing Systems 32*.

Dutta S, **Biswas A and Ahrens J** (2019) Multivariate pointwise information-driven data sampling and visualization. *Entropy 21*(7), 699.

Garcia S, **Derrac J**, **Cano J and Herrera F** (2012) Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Transactions on Pattern Analysis and Machine Intelligence 34*(3), 417–435.

García S and Herrera F (2009) Evolutionary undersampling for classification with imbalanced datasets: Proposals and taxonomy. *Evolutionary Computation 17*(3), 275–306.

Gouasmi A, **Parish EJ and Duraisamy K** (2017) A priori estimation of memory effects in reduced-order models of nonlinear systems using the mori–zwanzig formalism. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences 473*(2205), 20170385.

Hart P (1968) The condensed nearest neighbor rule (corresp.). *IEEE Transactions on Information Theory 14*(3), 515–516.

Hassanaly M, **Glaws A and King RN** (2022a) GANISP: A GAN-assisted importance splitting probability estimator. In *AAAI 2022 Workshop on AI for Design and Manufacturing (ADAM)*.

Hassanaly M, **Glaws A**, **Stengel K and King RN** (2022b) Adversarial sampling of unknown and high-dimensional conditional distributions. *Journal of Computational Physics 450*, 110853.

Hassanaly M, **Perry BA**, **Mueller ME and Yellapantula S** (2021a) Uniform-in-phase-space data selection with iterative normalizing flows. Preprint, arXiv:2112.15446.

Hassanaly M and Raman V (2021) Classification and computation of extreme events in turbulent combustion. *Progress in Energy and Combustion Science 87*, 100955.

Hassanaly M, **Tang Y**, **Barwey S and Raman V** (2021b) Data-driven analysis of relight variability of jet fuels induced by turbulence. *Combustion and Flame 225*, 453–467.

Hinton GE and Salakhutdinov RR (2006) Reducing the dimensionality of data with neural networks. *Science 313*(5786), 504–507.

Hulten G, **Spencer L and Domingos P** (2001) Mining time-changing data streams. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY: ACM, pp. 97–106.

Jankowski N and Grochowski M (2004) Comparison of instances selection algorithms I. Algorithms survey. In *International Conference on Artificial Intelligence and Soft Computing*. Berlin, Heidelberg: Springer, pp. 598–603.

Kingma DP, **Salimans T**, **Jozefowicz R**, **Che X**, **Sutskever I and Welling M** (2016) Improving variational inference with inverse autoregressive flow. *Advances in Neural Information Processing Systems 29*.

Klasky S, **Thayer J and Najm H** (2021) Data reduction for science: Brochure from the advanced scientific computing research workshop. Technical report, DOE Office of Science (SC).

Krawczyk B and Woźniak M (2015) One-class classifiers with incremental learning and forgetting for data streams with concept drift. *Soft Computing 19*(12), 3387–3400.

Lapointe S, **Savard B and Blanquart G** (2015) Differential diffusion effects, distributed burning, and local extinctions in high Karlovitz premixed flames. *Combustion and Flame 162*(9), 3341–3355.

Leevy JL, **Khoshgoftaar TM**, **Bauder RA and Seliya N** (2018) A survey on addressing high-class imbalance in big data. *Journal of Big Data 5*(1), 1–30.

Lloyd S (1982) Least squares quantization in PCM. *IEEE Transactions on Information Theory 28*(2), 129–137.

López V, **Triguero I**, **Carmona CJ**, **García S and Herrera F** (2014) Addressing imbalanced classification with instance generation techniques: IPADE-ID. *Neurocomputing 126*, 15–28.

Mudrakarta PK, **Taly A**, **Sundararajan M and Dhamdhere K** (2018) It was the training data pruning too! Preprint, arXiv: 1803.04579.

Müller T, **McWilliams B**, **Rousselle F**, **Gross M and Novák J** (2019) Neural importance sampling. *ACM Transactions on Graphics (TOG) 38*(5), 1–19.

Nguyen H-T, **Domingo P**, **Vervisch L and Nguyen P-D** (2021) Machine learning for integrating combustion chemistry in numerical simulations. *Energy and AI 5*, 100082.

Nouanesengsy B, **Woodring J**, **Patchett J**, **Myers K and Ahrens J** (2014) ADR visualization: A generalized framework for ranking large-scale scientific data using analysis-driven refinement. In *2014 IEEE 4th Symposium on Large Data Analysis and Visualization (LDAV)*. New York, NY: IEEE, pp. 43–50.

Oseledets IV (2011) Tensor-train decomposition. *SIAM Journal on Scientific Computing 33*(5), 2295–2317.

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

Papamakarios G, **Pavlakou T and Murray I** (2017) Masked autoregressive flow for density estimation. *Advances in Neural Information Processing Systems 30*.

Park Y, **Cafarella M and Mozafari B** (2016) Visualization-aware sampling for very large databases. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. New York, NY: IEEE, pp. 755–766.

Pedregosa F, **Varoquaux G**, **Gramfort A**, **Michel V**, **Thirion B**, **Grisel O**, **Blondel M**, **Prettenhofer P**, **Weiss R**, **Dubourg V**, **Vanderplas J**, **Passos A**, **Cournapeau D**, **Brucher M**, **Perrot M and Duchesnay E** (2011) Scikit-learn: Machine learning in python. *Journal of Machine Learning Research 12*, 2825–2830.

Peterka T, **Bard D**, **Bennett JC**, **Bethel EW**, **Oldfield RA**, **Pouchard L**, **Sweeney C and Wolf M** (2020) Priority research directions for in situ data management: Enabling scientific discovery from diverse data sources. *The International Journal of High Performance Computing Applications 34*(4), 409–427.

Pierce CD and Moin P (2004) Progress-variable approach for large-eddy simulation of non-premixed turbulent combustion. *Journal of Fluid Mechanics 504*, 73–97.

Raman V and Hassanaly M (2019) Emerging trends in numerical simulations of combustion systems. *Proceedings of the Combustion Institute 37*(2), 2073–2089.

Ramírez-Gallego S, **Krawczyk B**, **García S**, **Woźniak M and Herrera F** (2017) A survey on data preprocessing for data stream mining: Current status and future directions. *Neurocomputing 239*, 39–57.

Rankin BA, **Richardson DR**, **Caswell AW**, **Naples AG**, **Hoke JL and Schauer FR** (2017) Chemiluminescence imaging of an optically accessible non-premixed rotating detonation engine. *Combustion and Flame 176*, 12–22.

Rapp T, **Peters C and Dachsbacher C** (2019) Void-and-cluster sampling of large scattered data and trajectories. *IEEE Transactions on Visualization and Computer Graphics 26*(1), 780–789.

Salimans T, **Goodfellow I**, **Zaremba W**, **Cheung V**, **Radford A and Chen X** (2016) Improved techniques for training GANs. In *Advances in Neural Information Processing Systems*, pp. 2234–2242.

Saseendran AT, **Setia L**, **Chhabria V**, **Chakraborty D and Roy AB** (2019) Impact of data pruning on machine learning algorithm performance. Preprint, arXiv:1901.10539.

Sato T, **Chacon F**, **White L**, **Raman V and Gamba M** (2021) Mixing and detonation structure in a rotating detonation engine with an axial air inlet. *Proceedings of the Combustion Institute 38*(3), 3769–3776.

Savard B and Blanquart G (2017) Effects of dissipation rate and diffusion rate of the progress variable on local fuel burning rate in premixed turbulent flames. *Combustion and Flame 180*, 77–87.

Sirovich L (1987) Turbulence and the dynamics of coherent structures. I. Coherent structures. *Quarterly of Applied Mathematics 45* (3), 561–571.

Skalak DB (1994) Prototype and feature selection by sampling and random mutation hill climbing algorithms. In *Machine Learning Proceedings 1994*. San Fransisco, CA: Elsevier, pages 293–301.

Stöhr M, **Geigle KP**, **Hadef R**, **Boxx I**, **Carter CD**, **Grader M and Gerlinger P** (2019) Time-resolved study of transient soot formation in an aero-engine model combustor at elevated pressure. *Proceedings of the Combustion Institute 37*(4), 5421–5428.

Tabak EG and Turner CV (2013) A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics 66*(2), 145–164.

Tomek I (1976) Two modifications of CNN. *IEEE Transactions on Systems, Man, and Cybernetics SMC-6*, 769–772.

Triguero I, **Galar M**, **Merino D**, **Maillo J**, **Bustince H and Herrera F** (2016) Evolutionary undersampling for extremely imbalanced big data classification under apache spark. In *2016 IEEE Congress on Evolutionary Computation (CEC)*. New York, NY: IEEE, pp. 640–647.

Verheyen R and Stienen B (2020) Phase Space Sampling and Inference from Weighted Events with Autoregressive Flows. *SciPost Physics 10*(2), 038.

Wan K, **Barnaud C**, **Vervisch L and Domingo P** (2020) Chemistry reduction using machine learning trained from non-premixed micro-mixing modeling: Application to DNS of a syngas turbulent oxy-flame with side-wall effects. *Combustion and Flame 220*, 119–129.

Wang Z and Scott DW (2019) Nonparametric density estimation for high-dimensional data—Algorithms and applications. *Wiley Interdisciplinary Reviews: Computational Statistics 11*(4), e1461.

Wilson DL (1972) Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics* (*3*), 408–421.

Wilson DR and Martinez TR (2000) Reduction techniques for instance-based learning algorithms. *Machine Learning 38*(3), 257–286.

Woodring J, **Ahrens J**, **Figg J**, **Wendelberger J**, **Habib S and Heitmann K** (2011) In-situ sampling of a large-scale particle simulation for interactive visualization and analysis. In *Computer Graphics Forum*, vol. 30. Wiley Online Library, pp. 1151–1160.

Yellapantula S, **Perry BA and Grout RW** (2021) Deep learning-based model for progress variable dissipation rate in turbulent premixed flames. *Proceedings of the Combustion Institute 38*(2), 2929–2938.

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

Appendix A. Visual Evaluation of the Effect of Iterations

The effect of using more than two iterations is illustrated in this section. As can be seen in Figure A1, using more than two iterations does not significantly improve the uniformity of the distribution of samples. This observation holds for 1,000 (Figure A1, left) and 10,000 (Figure A1, right) samples. This is consistent with the results suggested by the distance criterion shown in Table 1.

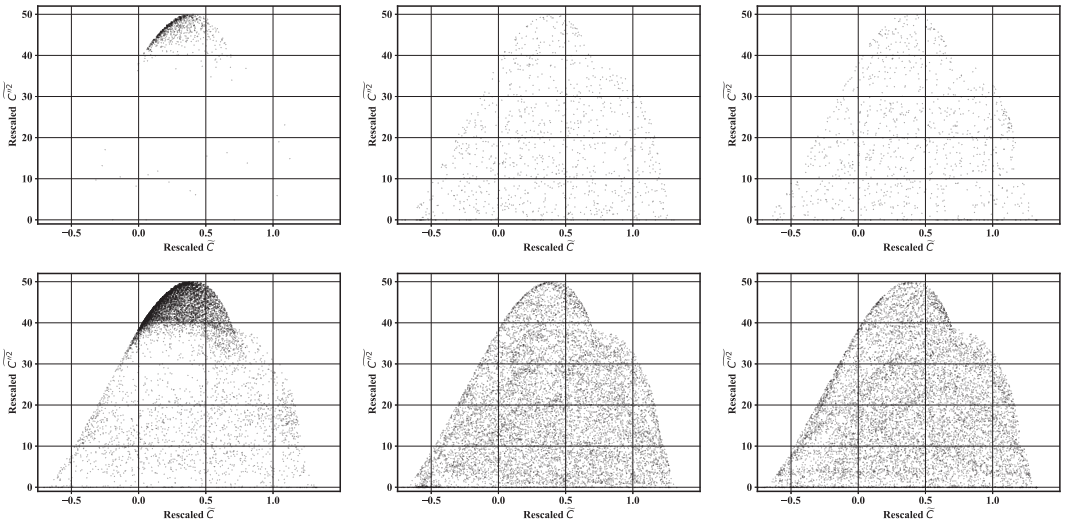

Figure A1. *Illustration of the effect of using more than two iterations to select* $n = 1{,}000$ *(top) and* $n = 10{,}000$ *(bottom) data points with one iteration (left), two iterations (middle), and three iterations (right) of Algorithm 2.*

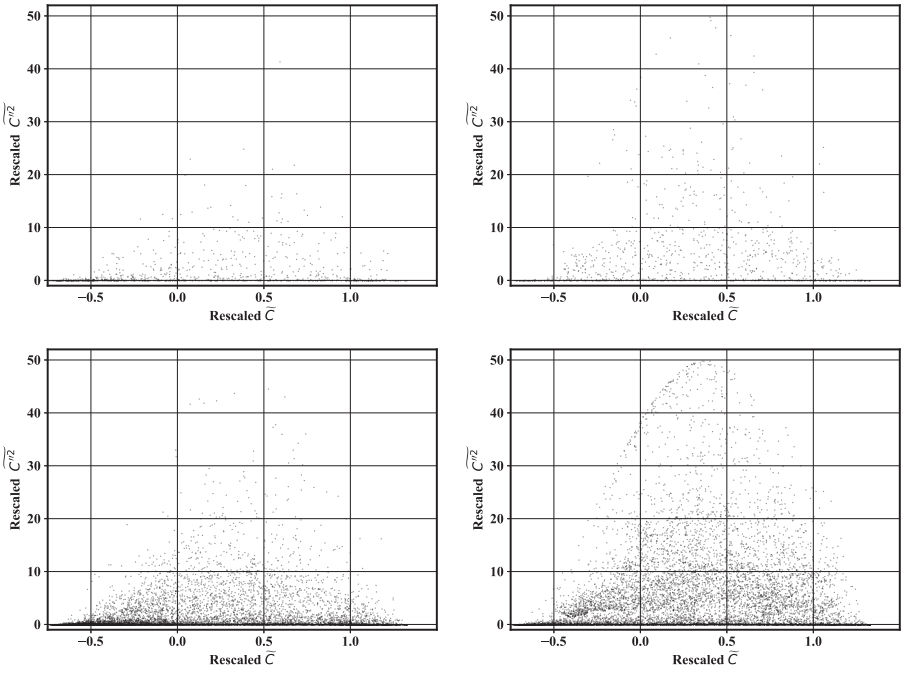

Figure A2. *Illustration of random (left) and stratified (right) selection schemes of* $n = 1{,}000$ *(top) and* $n = 10{,}000$ *(bottom) data points.*

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

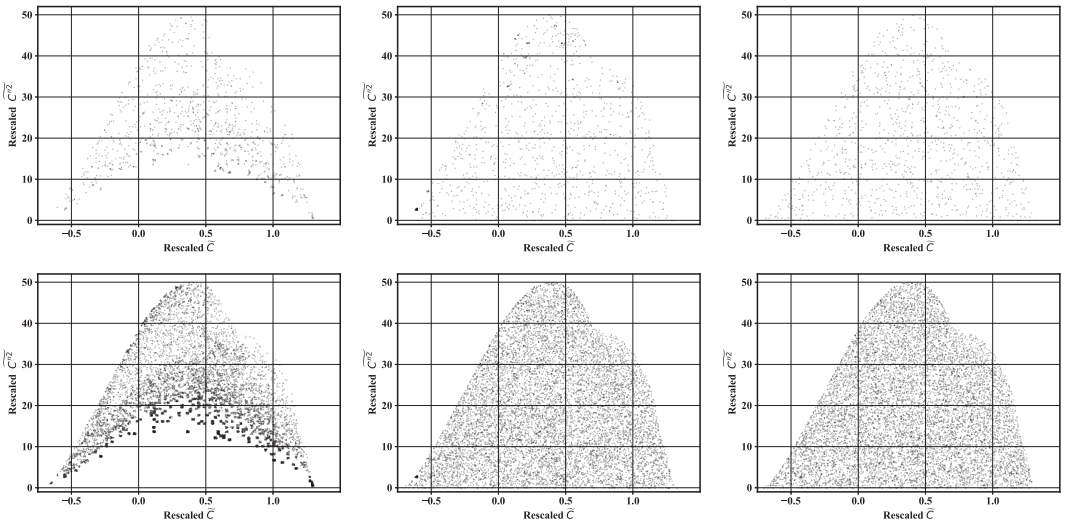

Figure A3. *Illustration of the effect of using more than two iterations to select $n = 1{,}000$ (top) and $n = 10{,}000$ (bottom) data points with one iteration (left), two iterations (middle), and three iterations (right) of Algorithm 2, using a binning method for the density estimation.*

The results should be compared to random and stratified sampling for reference, shown in Figure A2. It can be observed that the relative improvement obtained by using three iterations instead of two is minimal compared to random or stratified sampling.

Using a binning strategy for the density estimation leads to results similar to using normalizing flow as shown in Figure A3. While the first iteration oversamples rare data points, the next iterations lead to more uniform distribution of the downsampled data points.

Appendix B. Downsampling in a Different Dataset

The algorithm for uniform-in-phase-space data selection is generally suitable for large N, moderate D datasets and is not limited to the turbulent combustion and synthetic datasets described in this work. To demonstrate this, the algorithm is applied to sensor data relating to building power consumption on the campus of the National Renewable Energy Laboratory (NREL) in Golden, CO. NREL maintains instrumentation to track and record a wide array of data regarding power consumption in its buildings and weather conditions on campus to enable research into the integration of renewable energy and building efficiency technologies. Here, a dataset containing four streams of data corresponding to net power consumption in kW at NREL facilities taken at 1-min intervals for the period from January 1, 2017 to November 30, 2022 is considered. The particular facilities selected were chosen to

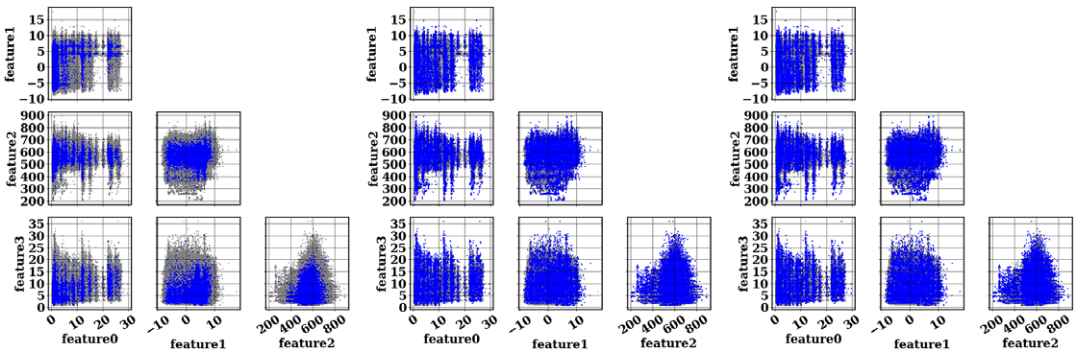

Figure B1. *Two-dimensional projection of $n = 10{,}000$ selected data points (•) out of an original four-dimensional building power consumption dataset ( ). (Left) Random sampling. (Middle) Algorithm 2 (two iterations). (Right) Algorithm 2 (three iterations).*

https://doi.org/10.1017/dce.2023.4 Published online by Cambridge University Press

Table B1. *Dependence of the distance criterion value on the number of iterations and instances selected (n) for the building power consumption dataset.*

Norm. flow	Algorithm 1	Algorithm 2 (2 iterations)	Algorithm 2 (3 iterations)	Random
$n = 1{,}000$	0.412 ± 0.024	$\mathbf{0.507 \pm 0.010}$	0.491 ± 0.008	0.280 ± 0.008
$n = 10{,}000$	0.225 ± 0.004	0.247 ± 0.001	$\mathbf{0.248 \pm 0.001}$	0.144 ± 0.002

Note. Larger distance criterion value is better. Bold entries denote best performances.

minimize the number of sampled time instances with missing data from any of the considered sensors; after discarding all time instances with missing data this results in a dataset with $N = 2.7 \times 10^6$ and $D = 4$. In the present use case and for energy systems in general, a data-driven model must be available to maintain a high reliability. Therefore, a reduced dataset must contain rarely observed conditions to ensure they are accounted for. This problem is prototypical of a broader class of problems where streaming data from a limited set of sensors may be available over a long period of time and robustness to rare events is important.

The sampled data points using Algorithm 2 and random sampling are compared in Figure B1. As was observed for the turbulent combustion dataset in Figure 9, Algorithm 2 leads to substantially more frequent sampling of points in the sparser regions of phase space. Notably, the character of the distribution of this dataset is very different than the previously considered dataset; in this case, the density of data points is strongly multimodal, with data concentrated in bands around particular power consumption values for each facility. Despite this multimodal character, the iterative procedure using normalizing flows to estimate the probability distribution works similarly as before. Algorithm 2 yields similar results with two and three iterations, indicating that the conclusion that two iterations are sufficient may generalize across different datasets. This is quantitatively confirmed by the data shown in Table B1. The distance criterion values are similar with two and three iterations, while both are substantially improved relative to the non-iterative Algorithm 1 and random sampling.

Cite this article: Hassanaly M, Perry BA, Mueller ME and Yellapantula S (2023). Uniform-in-phase-space data selection with iterative normalizing flows. *Data-Centric Engineering*, 4, e11. doi:10.1017/dce.2023.4