Check for updates

SOFTWARE TOOL ARTICLE

REVISED scikit-matter : A Suite of Generalisable Machine Learning Methods Born out of Chemistry and Materials Science

[version 3; peer review: 4 approved]

Alexander Goscinski[1], Christian A. Jorgensen[2], Victor Paul Principe[1], Guillaume Fraux[1], Sergei Kliavinek[1], Benjamin Aaron Helfrecht[1,3], Rhushil Vasavada[4], Philip Loche[1], Michele Ceriotti[1], Rose Kathleen Cersonsky[1,2]

[1]Laboratory of Computational Science and Modeling (COSMO), Institute of Materials, Ecole Polytechnique Federale de Lausanne, Lausanne, Vaud, 1015, Switzerland
[2]Department of Chemical and Biological Engineering, University of Wisconsin-Madison, Madison, Wisconsin, 53706, USA
[3]Pacific Northwest National Laboratory, Richland, WA, 99352, USA
[4]Department of Computer Sciences, University of Wisconsin, Madison, USA

v3 **First published:** 16 May 2023, **3**:81
https://doi.org/10.12688/openreseurope.15789.1
Second version: 18 Sep 2023, **3**:81
https://doi.org/10.12688/openreseurope.15789.2
Latest published: 26 Dec 2025, **3**:81
https://doi.org/10.12688/openreseurope.15789.3

Abstract

Easy-to-use libraries such as scikit-learn have accelerated the adoption and application of machine learning (ML) workflows and data-driven methods. While many of the algorithms implemented in these libraries originated in specific scientific fields, they have gained in popularity in part because of their generalisability across multiple domains. Over the past two decades, researchers in the chemical and materials science community have put forward general-purpose machine learning methods. The deployment of these methods into workflows of other domains, however, is often burdensome due to the entanglement with domain-specific functionalities. We present the python library scikit-matter that targets domain-agnostic implementations of methods developed in the computational chemical and materials science community, following the scikit-learn API and coding guidelines to promote usability and interoperability with existing workflows.

Open Peer Review

Approval Status ✓ ✓ ✓ ✓

	1	2	3	4
version 3 (revision) 26 Dec 2025		✓ view		
version 2 (revision) 18 Sep 2023	✓ view		✓ view	✓ view
version 1 16 May 2023	✓ view	? view		

1. **Andrew S. Rosen**, University of California, Berkeley, Berkeley, USA
2. **Ekaterina I. Izgorodina**, Monash University, Clayton, Australia
3. **Brian DeCost**, National Institute of Standards and Technology, Gaithersburg,

Keywords
Python, feature selection, sample selection, PCovR, KPCovR, feature reconstruction, directional convex hull

USA

4. **Robert Meissner**, Hamburg University of Technology, Hamburg, Germany

Any reports and responses or comments on the article can be found at the end of the article.

This article is included in the Horizon 2020 gateway.

This article is included in the Analytical Chemistry gateway.

This article is included in the European Research Council (ERC) gateway.

This article is included in the Machine Learning collection.

This article is included in the Computational Chemistry collection.

Corresponding author: Rose Kathleen Cersonsky (rose.cersonsky@wisc.edu)

Author roles: Goscinski A: Conceptualization, Data Curation, Formal Analysis, Software, Validation, Visualization, Writing – Original Draft Preparation; **Jorgensen CA**: Conceptualization, Data Curation, Formal Analysis, Methodology, Software, Visualization, Writing – Original Draft Preparation; **Principe VP**: Conceptualization, Data Curation, Formal Analysis, Software, Validation, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing; **Fraux G**: Software, Validation, Writing – Review & Editing; **Kliavinek S**: Software, Validation, Writing – Review & Editing; **Helfrecht BA**: Data Curation, Methodology, Software, Writing – Review & Editing; **Vasavada R**: Software, Visualization, Writing – Original Draft Preparation; **Loche P**: Software; **Ceriotti M**: Conceptualization, Funding Acquisition, Methodology, Software, Supervision, Writing – Review & Editing; **Cersonsky RK**: Conceptualization, Data Curation, Formal Analysis, Methodology, Project Administration, Software, Supervision, Visualization, Writing – Original Draft Preparation, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: V.P.P. and M.C. received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme Grant No. 101001890-FIAMMA. M.C., R.K.C., B.A.H. received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme grant No Grant No. 677013-HBMAP. A.G. and M.C. acknowledge support from the Swiss National Science Foundation (Project No. 200021-182057). G.F. acknowledges support from the Swiss Platform for Advanced Scientific Computing (PASC). R.K.C. acknowledges support from the University of Wisconsin - Madison and the Wisconsin Alumni Research Foundation (WARF)
The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Copyright: © 2025 Goscinski A *et al*. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Goscinski A, Jorgensen CA, Principe VP *et al.* **scikit-matter : A Suite of Generalisable Machine Learning Methods Born out of Chemistry and Materials Science [version 3; peer review: 4 approved]** Open Research Europe 2025, **3**:81 https://doi.org/10.12688/openreseurope.15789.3

First published: 16 May 2023, **3**:81 https://doi.org/10.12688/openreseurope.15789.1

REVISED **Amendments from Version 2**

In this revision, we introduce Principal Covariates Classification (PCovC) and its kernelized variant (KPCovC) to the scikit-matter library, extending our dimensionality reduction framework to classification tasks. These methods balance feature reconstruction and class separability through a tunable mixing parameter and support integration with scikit-learn classifiers. We demonstrate their utility on both chemical and socioeconomic datasets, showing improved interpretability and performance compared to PCA and kernel PCA, particularly in highly imbalanced and nonlinear scenarios. These improvements were led by Christian Jorgensen, Rhushil Vasavada, and Rose K. Cersonsky, with the two former being added as new authors to the revised edition.

Any further responses from the reviewers can be found at the end of the article

1 Introduction

While machine learning (ML) algorithms are applied in a wide variety of fields, the relative importance of different aspects of workflows can vary widely between disciplines. For those who apply ML to predict physical or chemical properties, there is an increased emphasis on engineering and understanding numerical features from the underlying physical entities—namely, atoms and their relative position in the structure or molecule—in a format compatible with ML pipelines[3–6]. Engineering these features is a rich sub-discipline of machine learning unto itself, and thus several best practices have been established for analysing the resulting representations (see Page 4 for an introduction), including a reduction of redundant information in features and samples[7–9], comparing different featurisations on datasets[10,11], and analysing their reduced manifolds[12]. These methods are, however, often inextricably linked to the libraries computing these representations[13–15] and are not available to a wider audience outside of these sub-communities. The objective of the open-source library `scikit-matter` is to make these ML methods accessible to a wider community by following the `scikit-learn` API and coding guidelines, and by treating the features as agnostic to their domain. It not only serves as a conduit between featurisation software for comparing alternative descriptors but allows a 'plug-and-play' of these methods into any workflows, irrespective of the representation or field of use.

In this text, we will review several major modules in `scikit-matter`: 1) Feature Reconstruction Measures (Page 5), used to assess the mutual information contained in two separate representations of the same dataset, 2) Linear and Kernelized Principal Covariates Regression (Page 6), used to construct a new set of features designed to correlate with one or more target properties, 3) Linear and Kernelized Principal Covariates Classification (Page 9), used to construct features that are linearly separable for categorical problems, 4) Farthest Point Sampling, CUR, and their PCovR-Adaptations (Page 13), methods used to sub-select features and/or samples for machine learning problems, and 5) the Directional Convex Hull Construction (Page 16), used to identify data points that sit on a bounding manifold of a demonstrative dataset.

We demonstrate the use of these methods in contexts both inside and outside the chemical domain. We first revisit a dataset originally published in Engel *et al.*[16] and available publicly via the Materials Cloud Archive[17,18]. This dataset contains theoretical and known ice structures, *i.e.* 15'869 "reasonable" crystal structures with hydrogen and oxygen in a 2:1 ratio, their lattice energies (in eV/molecule) and densities (in g/cm^3). We have augmented this dataset to include the atomic Mulliken charges, as computed with DFTB+[19], in units of elementary charges [e].

We also show the utility of these methods beyond the chemical sciences by employing the World Health Organization statistics on life expectancy[20], curated from the open repository from the World Bank. This dataset contains 2,020 data points, each representing a different country during a given year, reporting variables pertaining to the population size[21], gross domestic product (GDP)[22], health-based[23] and education-based[24] spending, prevalence of HIV/AIDS[25] and tuberculosis[26], immunisations[27,28], and undernourishment of the country's population[29]. We have also included a modification to this dataset containing statistics on the World Bank's income group classifications for 1,687 countries[30,31].

2 A Quick Guide to the Smooth Overlap of Atomic Positions (SOAP) Representation

Throughout the paper, we use the Smooth Overlap of Atomic Positions (SOAP) representation to numerically encode our chemical data[32]. We note, however, that other numerical encodings of chemical information are compatible with the methods presented here[33]. As the SOAP formalism is not well-known outside the atomistic machine learning field, here we give a quick and simplified overview of the concepts and terminology associated with this data representation.

Numerous electronic and chemical properties are determined by the spatial relationship between a "central" atom and its nearest neighbouring atoms[34]. For most properties, the importance of neighbouring atoms decays with their distance from the central atom. It is therefore worthwhile to only represent the neighbourhood around each atom up to a cutoff. In the SOAP representation, each atom in the neighbourhood is represented by a Gaussian density. A finite set of numerical descriptors of all densities in the neighbourhood is then used as features.

However, to make efficient use of our data, we need to incorporate the symmetries of our system, as many physical properties do not change under rotations and translations. We, therefore, use the neighbourhood centred on each atom i, known as

atomic density ρ_i, to remove the dependency on the origin of our system[8]. The neighbourhood is described by a combination of radial and spherical functions to form a basis for the expansion. One can choose one of many functional forms for radial bases[6,32,35], and any number of basis functions to do this expansion. The number of basis functions determines the number of features. Generally, a higher amount of features corresponds to a higher "resolution" of the neighbourhood.

The set of expansion coefficients of ρ_i in this basis results in something known as *density coefficients* for each central atom i. We can take the correlation of these coefficients to build descriptors corresponding to two-body correlations $\rho_i^{\otimes 1}$, three-body correlations $\rho_i^{\otimes 2}$, or higher. Here, the superscript corresponds to the νth-neighbour correlation order expansion, where a two-body correlation is the first-order expansion. We often write features symmetrised over rotations as $\overline{\rho_i^{\otimes \nu}}$ specifically calling $\overline{\rho_i^{\otimes 1}}$ the "Radial Spectrum" and $\overline{\rho_i^{\otimes 2}}$ the "Power Spectrum".

A full explanation of the SOAP formalism is contained in Bartók *et al.*[32] and Willatt *et al.*[8]. SOAP is not the only popular machine-learning representation in chemical sciences, and a full review of many of the different physicsinformed representations is available in Musil *et al.*[6].

In line with its goal of being disentangled from domainspecific components, `scikit-matter` itself does not compute these atomic descriptors and instead takes as input matrices corresponding to descriptors computed by prominent software such as `librascal`[6], `QUIP`[36–38], and `DScribe`[39]. For the purposes of testing and examples, `scikit-matter` does contain minimal datasets, each including a small set of molecules or materials, a suitable featurisation, and a set of properties[11,40].

3 Feature reconstruction measures

In order to compare two separate sets of features, one can employ regression errors to quantify the mutual relationships of different forms of featurisations, as demonstrated in Goscinski *et al.*[11]. We determine this error, or the feature reconstruction measure (FRM), by reconstructing one set of features from the other with a constrained transformation, where different constraints express different types of relationships between the two sets of features.

Say we have a dataset that is represented in full detail by a representation Θ, and we want to assess the amount of information lost by using an alternate representation $\mathscr{F}$. We can check the detail contained in $\mathscr{F}$ by computing FRM($\mathscr{F}$, Θ), where FRM = 0 corresponds to a perfect reconstruction with no loss, and FRM $\approx$ 1 denotes a complete information loss wrt. Θ. However, there rarely exists a ground-truth representation Θ and we are more commonly comparing two likely-imperfect representations $\mathscr{F}$ and $\mathscr{G}$. In this case, we compute FRM($\mathscr{F}$, $\mathscr{G}$) and FRM($\mathscr{G}$, $\mathscr{F}$). The feature set that results in the higher reconstructive measure is considered higher in information capacity (*e.g.* if FRM($\mathscr{F}$, $\mathscr{G}$) > FRM($\mathscr{G}$, $\mathscr{F}$), then $\mathscr{F}$ is the more information-rich feature set). The advantage of the FRM is that it can give quantitative results about the shared information content between two sets of features, even without the existence of a ground truth.

In Figure 1 we show a schematic of the different FRMs contained in `scikit-matter`. The simplest FRM is the global feature reconstruction error (GFRE), expressed as the linearly decodable information, as given by performing ridge regression between the two sets of features. We note that in the case of choosing the property as ground truth, the GFRE is equivalent to a regression task on the standardized property. The global feature reconstruction distortion (GFRD) constrains the transformation to be orthogonal to demonstrate the deformation incurred by transforming between the two feature spaces.

Extending the analysis to non-linear dependencies, the local feature reconstruction error (LFRE) applies ridge regression for each point locally on the k-nearest neighbours. These methods are of particular use in assessing the hyperparameters of ML descriptors[11] and have been employed to compare the efficiency of different basis sets in encoding geometrical information[6,41].

Implementation

The FRMs differ in two aspects: the locality of the reconstruction (global or local) and the constraints of the regression. In each FRM, the two feature sets are partitioned into training and testing sets. We standardise the features of $\mathscr{F}$ and $\mathscr{G}$ individually, then we regress the features of $\mathscr{F}$ onto $\mathscr{G}$ to compute the errors. In the global measures, we use the entire dataset for the reconstruction, whereas in the local measure, we perform a regression for each sample on the set of the k-nearest points within $\mathscr{F}$. The number k is given by the user with the parameter `n_local_points`. The reconstruction *error* is by default computed using a 2-fold cross-validation (CV) and ridge regression as estimator. As most interesting applications use large feature vectors, we implemented a custom `Ridge2FoldCV` in the `skmatter.linear_model` module to improve computational efficiency. In `scikit-learn`, an implementation of the leave-one-out CV with a similar purpose of speeding up the CV exists. A detailed comparison between the two approaches can be found in one of the examples which are included in the documentation of `scikit-matter`, https://scikit-matter.readthedocs.io/en/v0.2.0/examples/regression/Ridge2FoldCVRegularization.html. For the reconstruction *distortion* we use orthogonal regression as implemented in `OrthogonalRegression` in the `skmatter.linear_model` module.

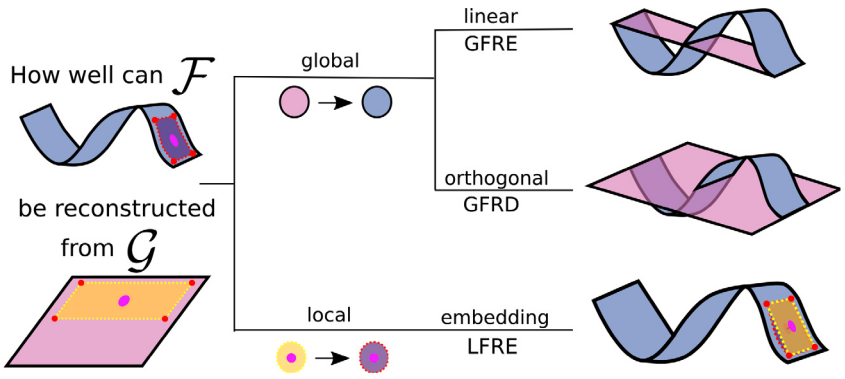

Figure 1. The different forms of feature reconstructions to assess two feature spaces (blue and pink) describing the same dataset. Here, we are reconstructing the curved manifold (blue) using the planar manifold (pink), as it is often the case to approximate a complex manifold with a simpler alternative. The area framed by the dotted line is an example of a local neighbourhood of one sample (the pink dot) that enables the reconstruction of nonlinearities. (Top) The linear transformation is used in the global feature reconstruction error (GFRE). (Middle) The orthogonal transformation is used in the global feature reconstruction distortion (GFRD). (Bottom) A local linear transformation of a neighbourhood is used in local feature reconstruction error (LFRE). On the right, the reconstructions of the manifold are drawn in pink together with the curved manifold in blue. The measures correspond to the root-mean-square difference between the reconstructed and curved manifold.

Workflow The FRMs can all be called similarly:

```
from skmatter.metrics import
    global_reconstruction_error as GFRE
gfre = GFRE(F, G)

from skmatter.metrics import
    local_reconstruction_error as LFRE
lfre = LFRE(F, G, n_local_points=10)
```

It is also possible to specify general feature scalers and estimators in any of the FRM classes, in which case the invocation looks like:

```
from skmatter.metrics import
    global_reconstruction_error as GFRE

gfre = GFRE(F, G, scaler=my_scaler,
            estimator=my_estimator)
```

Use case: Determining the number of radial basis functions necessary for representing neighbourhood shells

In common frameworks for machine learning potentials, each atomic neighbourhood is represented by expanding the many-body correlation information over a set of basis functions[6]. The resolution of this expansion varies greatly based on the type of dataset and choice of basis. In this use case, we analyse the number of radial basis functions required to resolve each neighbour shell from a central atom. Here, "shell" refers to a spherical shell that corresponds to a peak in the radial distribution function as sketched out in Figure 2a). For a given choice of basis functions, we say that the representation is *converged* when adding new bases in the series yields no new information. In other words, we can say that the number of basis functions n has converged when the GFRE $n \rightarrow n{+}1$ saturates. Here we show the convergence behaviour of a PCA-optimised[1] basis set[41] expanded on the "Radial Spectrum", showing the number of basis functions needed to resolve each neighbour shell up to numerical accuracy of the linear system solver within the ice dataset. The numerical accuracy of the solver is computed with GFRE for the identity relationship for 100 basis functions.

For this dataset, we determine two neighbour shells for the species pairs H-H and O-H (demonstrated in Figure 2(a)). With each additional shell, we need a greater number of basis functions, as shown in Figure 2(c). We see a near-linear relationship between the number of shells considered and the number of basis functions needed to obtain convergence. In fact, using a toy dataset with equidistant shells and a uniform distribution of atoms in each shell, we recover a perfect linear relationship. Thus, the number of basis functions needed to obtain similar resolutions scales linearly with the number of shells. The difference between these results and those of the toy dataset can be explained by the irregularly spaced shells and the non-uniform number of atoms across shells in the ice dataset.

4 Linear and non-linear principal covariates regression

Often, one wants to construct new ML features from their current representation in order to compress data or visualise trends in the dataset. In the archetypal method for this dimensionality reduction, principal components analysis (PCA), features are transformed into the latent space which best preserves the variance of the original data. *Principal Covariates Regression* (PCovR), as introduced by de Jong and Kiers[42], is

[1] We optimise our basis functions starting from 100 basis functions of the DVR basis[6].

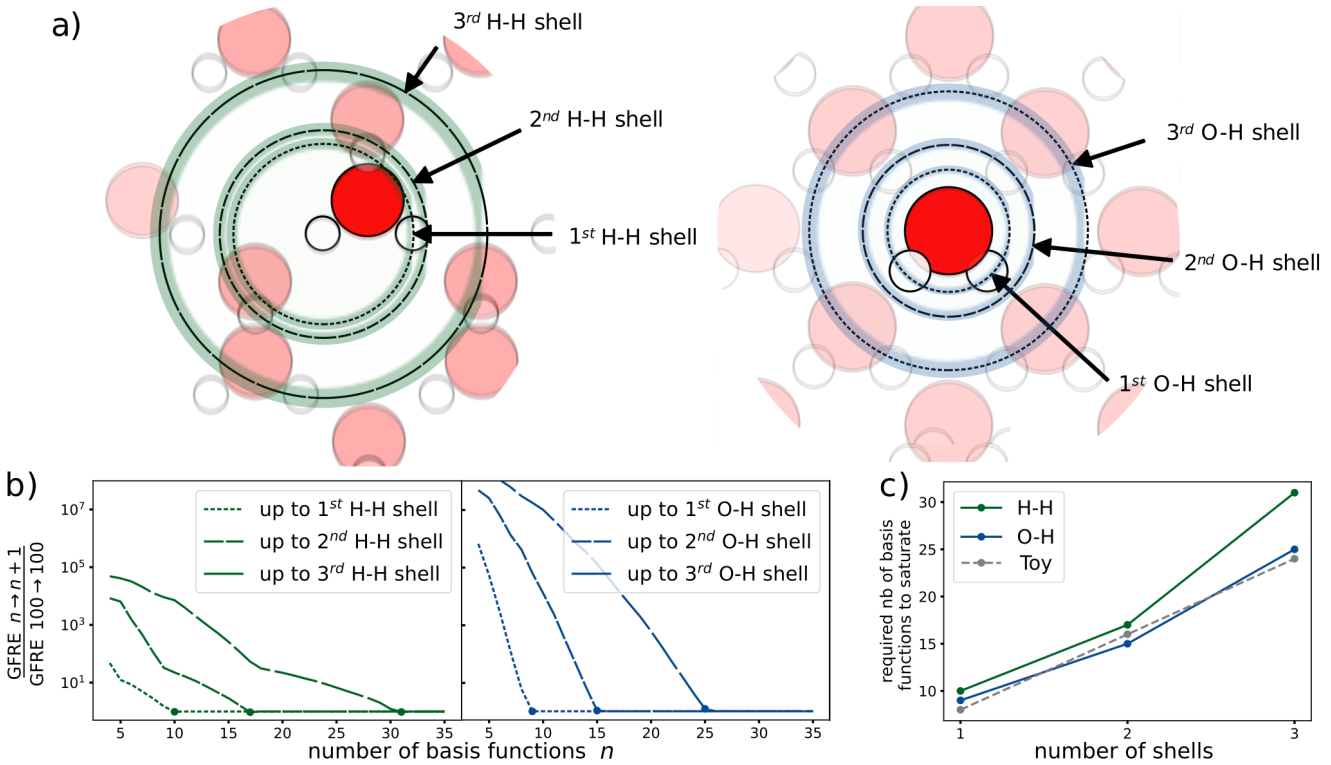

Figure 2. The relationship between the number of radial basis functions and the resolution of each neighbourhood shell in water. a) The different atomic shells present in the ice dataset. **b**) The convergence of the GFRE for an incremental change in the number of basis functions normalised by the numerical accuracy of the linear system solver. The numerical accuracy of the solver is determined by computing the GFRE for the identity relation for 100 basis functions. **c**) The number of basis functions required to resolve up to the nth neighbour shell.

a modification to PCA that incorporates target information, such that the resulting embedding could be tuned using a mixing parameter α to improve performance in regression tasks ($\alpha = 0$ corresponding to linear regression and $\alpha = 1$ corresponding to PCA). Helfrecht *et al.*[12] introduced the non-linear version, *Kernel Principal Covariates Regression* (KPCovR), where the mixing parameter α now interpolates between kernel ridge regression ($\alpha = 0$) and kernel principal components analysis (KPCA, $\alpha = 1$)[43].

The α parameter determines how much emphasis to give either the regression performance or feature reconstruction. As shown for a toy dataset in Figure 3, a KPCovR of $\alpha = 1.0$ will replicate the results of a Kernel PCA (weighted entirely on the reconstruction task), whereas $\alpha = 0.0$ contains the regression weights as features. Typically, a value of $\alpha \approx 0.5$ yields the most qualitatively insightful results, provided that the features and targets are properly standardised.

Implementation

In a PCA decomposition, we obtain a latent-space projection by taking the singular value decomposition of the feature matrix $\mathbf{X}$,

$$\mathbf{X} = \mathbf{U}_K \Sigma \mathbf{U}_C^T \tag{1}$$

where $\mathbf{U}_K$ and $\mathbf{U}_C$ are the eigenvectors of the Gram matrix ($\mathbf{X}\mathbf{X}^T$) and covariance matrix $\mathbf{X}^T\mathbf{X}$ and Σ are the singular values. In PCovR, we instead use the eigendecomposition of a modified Gram or covariance matrix, for example mixing $\mathbf{X}\mathbf{X}^T$ with the outer product of our predicted targets $\hat{\mathbf{Y}}\hat{\mathbf{Y}}^T$. As covered by de Jong and Kiers[42], it is useful to build this decomposition on the approximated target values. In `scikit-matter`, this is done by providing the predicted targets $\hat{\mathbf{Y}}$ or a scikit-learn-style estimator with which to regress $\mathbf{X}$ onto $\mathbf{Y}$.

Similarly, KPCovR constructs a modified kernel matrix, replacing the Gram matrix $\mathbf{X}\mathbf{X}^T$ with a user-selected kernel

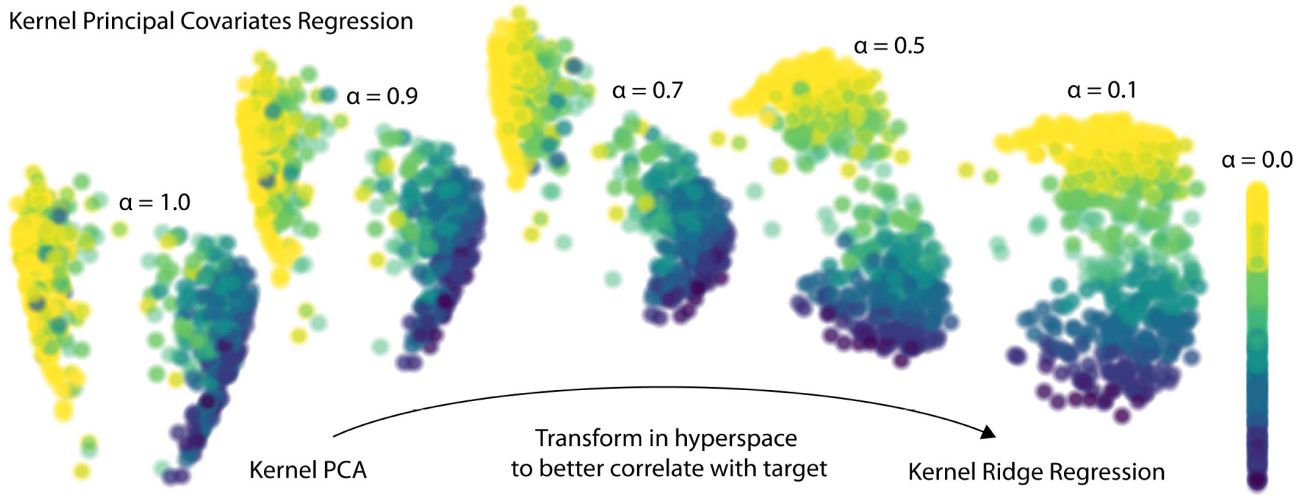

Figure 3. The evolution of latent-space projections and regressions as the mixing parameter *α* goes from 1 (Kernel PCA) to 0 (Kernel Ridge Regression) in Kernel PCovR. This procedure transforms the latent space projection to minimise combined KPCA and KRR loss. Typically, a value of $\alpha = 0.5$ yields a balanced projection and can be used to construct insightful feature-property maps.

(with kernel-building capabilities and an API consistent with `scikit-learn`) and requires a kernel regressor to obtain the approximated $\hat{\mathbf{Y}}$.

Workflow For example, a typical invocation of KPCovR is:

```
import skmatter.kernel_ridge.KernelRidge
import skmatter.decomposition.KernelPCovR

kernel_params = dict(kernel="rbf",
                     gamma=1)

regressor = KernelRidge(**kernel_params)

kpcovr = KernelPCovR(
                     mixing=0.5,
                     n_components=2,
                     regressor=regressor,
                     **kernel_params
                    )
kpcovr.fit(X, Y)

T = kpcovr.transform(X)
Yp = kpcovr.predict(X)
```

where $\mathbf{X}$ and $\mathbf{Y}$ are standardised matrices containing our features and target properties, respectively. For linear `PCovR`, `regressor` can be set to be any regression object (estimator) within `scikit-learn`.

Use Case: Mapping the charge of oxygen in ice

We build a linear PCovR map using as $\mathbf{X}$ the 3-body SOAP representations reported in Engel *et al.*[16] and as targets $\mathbf{Y}$, the Mulliken charges[2] (in units of e) for each of the water atoms in a subset of the ice structures.

By changing the mixing parameter α, we directly specify the attention that PCovR (or KPCovR) gives to the learning tasks of the target properties. In Figure 4, we show the performance of a PCovR mapping at both reconstructing the original SOAP vectors $\left(\ell_X \equiv \|\mathbf{X} - \hat{\mathbf{X}}\|^2\right)$ and regressing the charges $\left(\ell_Y \equiv \|\mathbf{Y} - \hat{\mathbf{Y}}\|^2\right)$. Here we use a fixed regularisation constant for comparability between different numbers of components and mixing parameters.

While all latent-space projections approach the same loss with increasing dimensionality when using fewer components, the effect on the regression loss ℓ_Y is much larger than on the reconstruction loss ℓ_X, as shown by the nearflat curves in ℓ_X across the mixing parameter. Conversely, the change in ℓ_Y across α can be quite large (as demonstrated by the poor regression saturation to $\alpha = 0.999$ in the top-right panel). You will also notice, particularly in the top-left panel, that the errors increase asymptotically in extreme cases ($\alpha = 0$, $\alpha = 1$). Thus in the majority of cases, an intermediate value will not only provide a better combined error but also better performance in regression tasks while imparting minimal impact on the reconstruction task.

We also often choose a mixing parameter of $\alpha = 0.5$ for visualisation, to weight equally the reconstruction and regression tasks. The resulting map, shown in the right side of

[2] computed using DFTB+[19] with the Third-Order Parameters for Organic and Biological Systems (3OB)

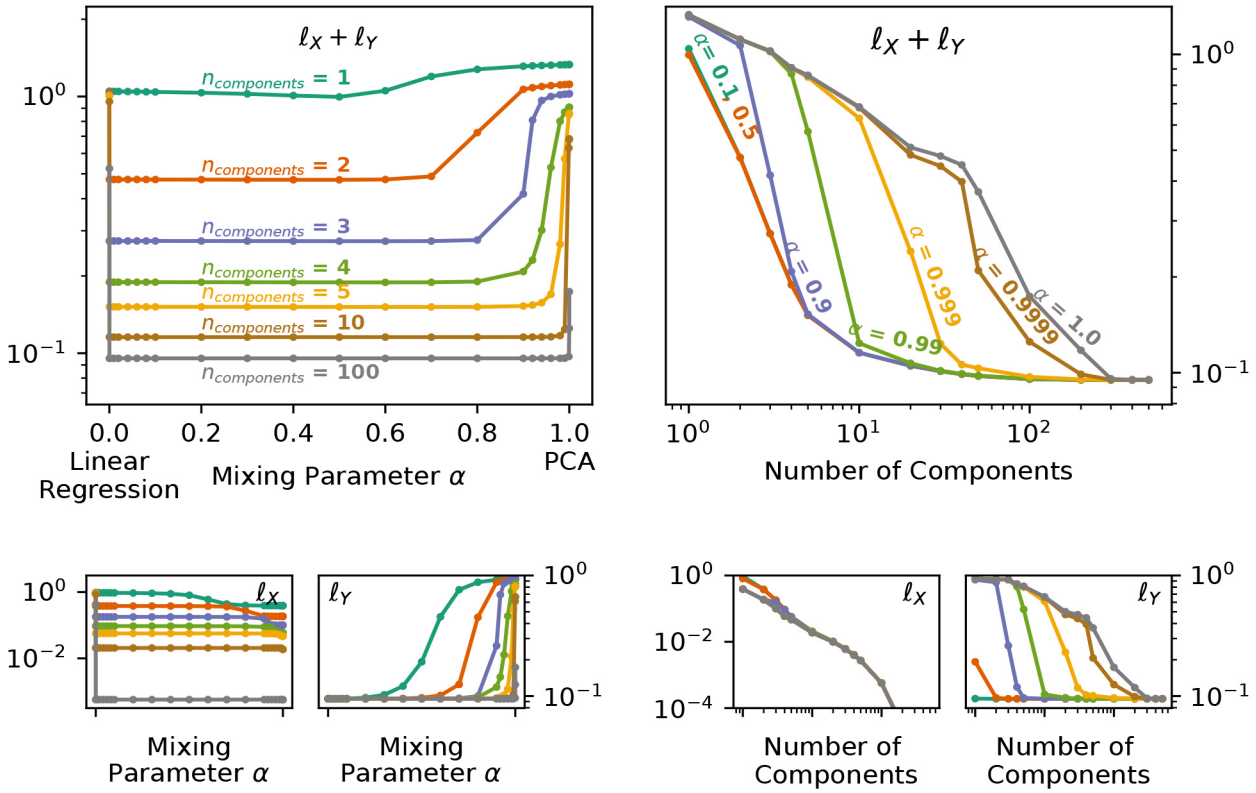

Figure 4. Reconstruction and Regression Errors for the PCovR Mappings. In each case, we report the unitless l^2 loss for multiple values of α and number of components for reconstruction $\ell_X \equiv \|\mathbf{X} - \hat{\mathbf{X}}\|^2$ and regression of the Mulliken charges $\ell_Y \equiv \|\mathbf{Y} - \hat{\mathbf{Y}}\|^2$, both the sum (top) and separately (bottom). (left) Plotted against the Mixing Parameter α. (right) Plotted against the Number of Components.

(Figure 5), delineates our oxygen environments based upon chemical similarity and our target charges, and from here we can see the separation of oxygens in hydroxide ($\star$, $\square$, ∇, $\diamond$), hydronium (last two in second row) ($\triangleleft$, $\triangleright$), and different arrangements of water, as shown in the insets below the maps. This is not true in the corresponding PCA (left), which is unable to distinguish these very different environments.

Use Case: Reducing the dimensions of the WHO dataset to predict life expectancy

The variables in the WHO dataset suffice to predict life expectancy with an R^2 score of 0.87 on an offset testing set with cross-validated ridge regression. When we use traditional PCA to reduce this set of variables to two components, our accuracy in prediction drops to $R^2 = 0.81$ (Figure 6(a)). This is marginally raised by turning to PCovR, where regressing on the first two covariates gives $R^2 = 0.83$ (Figure 6(b)).

The most dramatic results come from using non-linear methods. Employing kernel ridge regression with an optimised RBF kernel on the original features raises our maximum accuracy to $R^2 = 0.96$. Using the same kernel parameters to choose two features via kernel PCA reduces our accuracy to $R^2 = 0.57$ (Figure 6(c)); however, by doing the same with kernel PCovR we sacrifice little in accuracy ($R^2 = 0.96$, Figure 6(d)). In other words, we can obtain nearly the same regression performance using two covariates determined by KPCovR as we did with the full kernel. This example, from start-to-finish, is available in the `scikit-matter` documentation via https://scikit-matter.readthedocs.io/en/v0.2.0/examples/selection/FeatureSelection-WHODataset.html.

These kernel principal covariates can then be used to perform correlative analysis with the original demographic or statistical features, as was demonstrated by Cersonsky *et al.*[44] for medical diagnostics in the context of stillbirth outcomes.

5 Linear and non-linear principal covariates classification

While PCovR can be used to visualizing correlations with continuous targets, one may also need a dimensionality reduction method that can be used to visualize decision boundaries or trends with discrete targets. *Principal Covariates Classification* (PCovC), proposed in Jorgensen *et al.*[45], adapts PCovR for classification tasks by introducing a matrix of class likelihoods, $\mathbf{Z}$, as an approximation of target class labels. In the case of non-linear data, one can use *Kernel Principal Covariates*

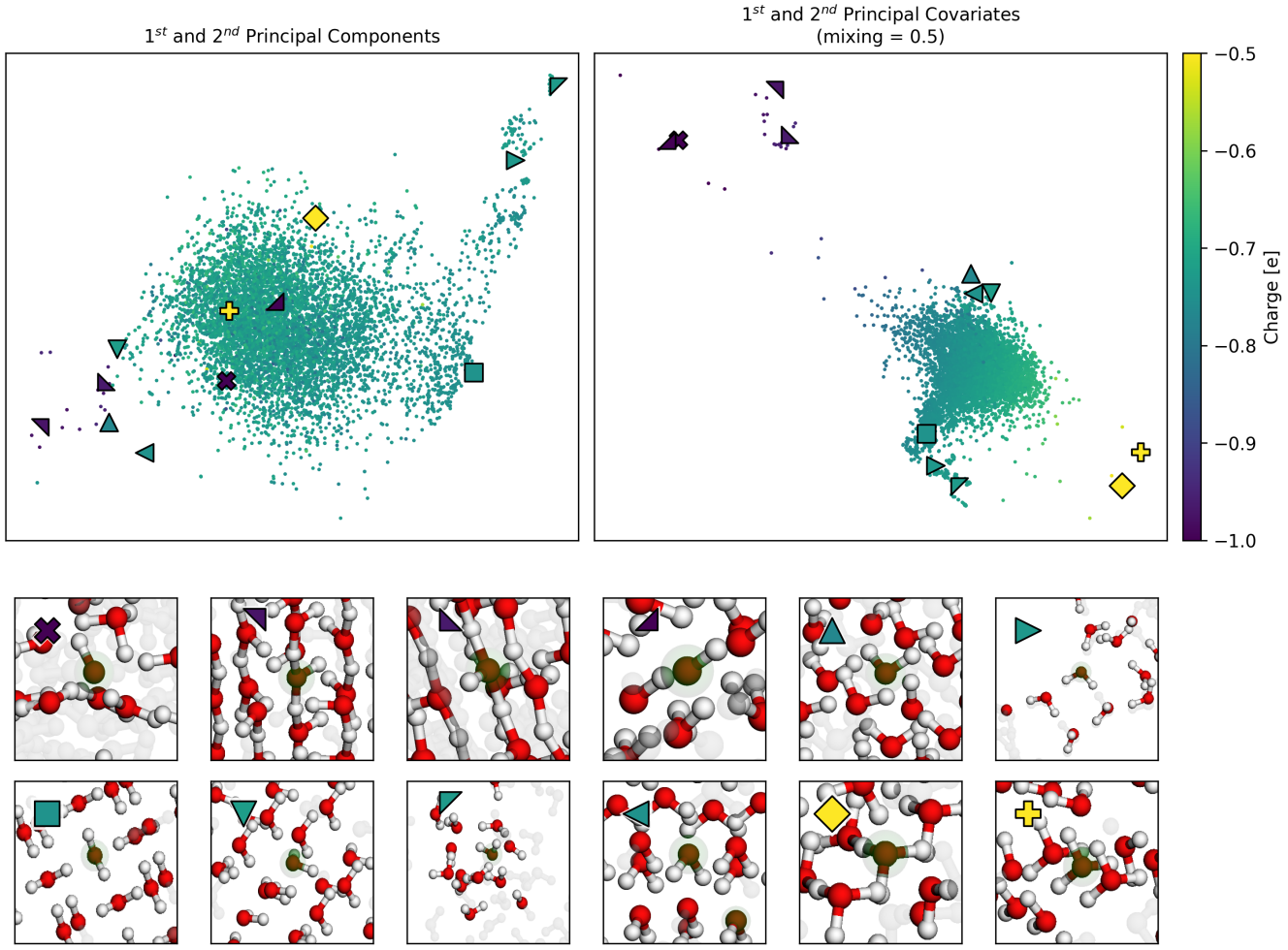

Figure 5. A Comparison of PCovR and PCA maps of Oxygens in Ice Structures. Both the upper maps show the first two components (left) or covariates (right) of the oxygen environments, coloured by the Mulliken charge. We have highlighted several extreme examples with special markers, corresponding to the markers also shown in the upper left corner of each inset. An interactive demo of these latent space maps can be loaded with chemiscope[46] using the link https://chemiscope.org/?load=https://github.com/lab-cosmo/skmatter-ore/raw/main/paper/chemiscope-oxygen-ice-structures-small.json.gz.

Classification (KPCovC), which similarly adopts PCovC for problems involving similarity kernels.

Similarly to PCovR, in PCovC, the α parameter determines the relative weight assigned to the classification and feature reconstruction tasks, with $\alpha = 1$ corresponding to a PCA decomposition (or KPCA, if using KPCovC) and $\alpha = 0$, while not analogous to a linear classification task, results in highly discriminative features. PCovC also requires a linear classifier with which to compute $\mathbf{Z}$, which allows for properties such as regularization to be retained in the resultant feature set. In Figure 7, we show how the choice of linear classifier used in PCovC can change the resultant map with a synthetic dataset; ultimately, while logistic regression and support vector classification tend to give the most useful results, the choice of classifier is truly of preference or performance.

Implementation

In a linear classification task, predictions are obtained through applying an activation function to a linear projection of the feature matrix $\mathbf{X}$,

$$\hat{\mathbf{Y}} = \phi(\mathbf{Z}) = \phi(\mathbf{X}\mathbf{P}_{\mathbf{XZ}}) \tag{2}$$

where ϕ is an activation function and $\mathbf{P}_{\mathbf{XZ}}$ is a set of learnable weights that map $\mathbf{X}$ to a set of class likelihoods, $\mathbf{Z}$. In PCovC, we similarly perform an eigendecomposition of a modified Gram or covariance matrix, in this instance using a product of class likelihoods, such as $\mathbf{Z}\mathbf{Z}^T$, as a proxy for the predicted targets. In `scikit-matter`, we obtain $\mathbf{Z}$ by fitting a scikit-learn-style linear classifier between $\mathbf{X}$ and $\mathbf{Y}$, or between a user-selected kernel and $\mathbf{Y}$ in the case of KPCovC.

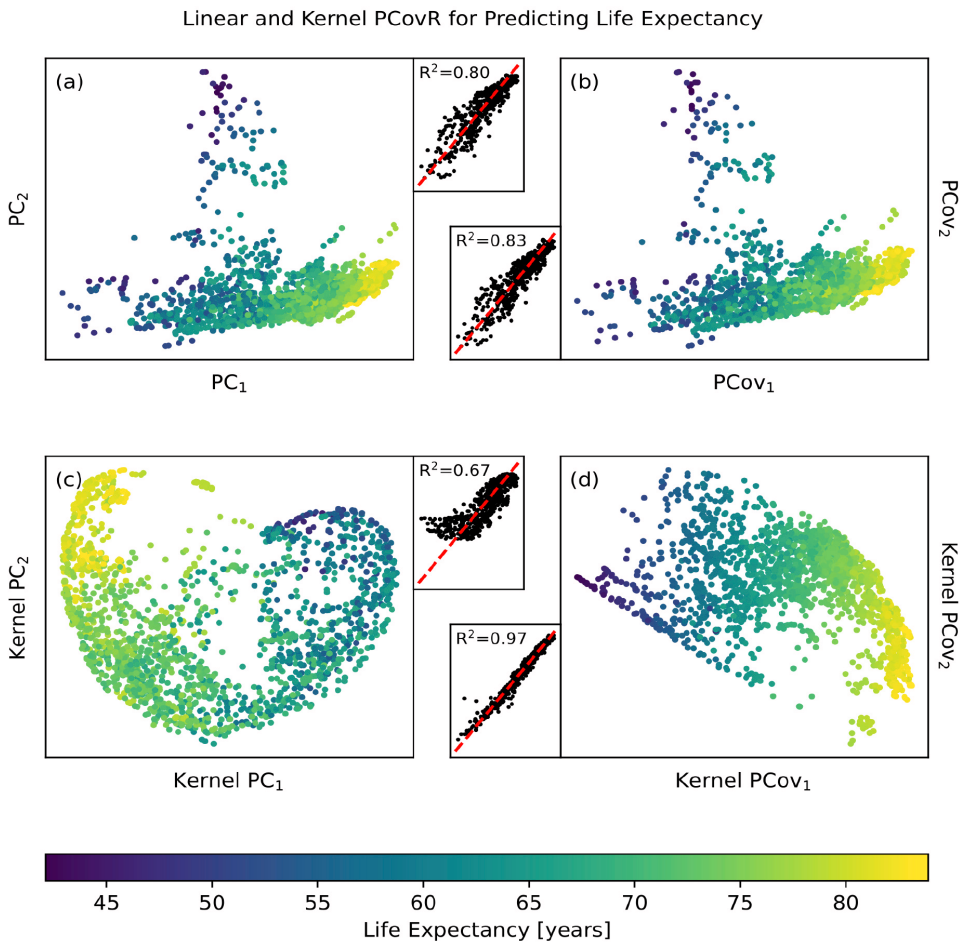

Figure 6. Linear and Non-linear Principal Components Analysis and Principal Covariates Regression Applied to the WHO Dataset. Here we show the resultant map and parity plot for (**a**) PCA, (**b**) PCovR at α = 0.5, (**c**) KPCA, and (**d**) KPCovR at α = 0.5 based on 13 statistical variables in order to predict life expectancy for a given country. In each case, we used the same standardised variable matrix to determine a two-dimensional latent space mapping (shown in the map) that was then fed into an appropriate regularised regression model, yielding the predictions in the parity plot. Regressions were all computed using the same 90/10 train/test split, with regression results reported for only the testing split.

Workflow As an example, a typical invocation of PCovC is:

```
from sklearn.linear_model import
LogisticRegression
from skmatter.decomposition import PCovC


classifier = LogisticRegression()

pcovc = PCovC(
              mixing=0.5,
              n_components=2,
              classifier=classifier
              )

pcovc.fit(X, Y)

T = pcovc.transform(X)
Z = pcovc.decision_function(X)
Yp = pcovc.predict(X)
```

where **X** is a standardized matrix containing our features and **Y** contains a set of discrete labels representing our target properties. For both linear and kernel PCovC, `classifier` can be set to any linear classifier object within `scikit-learn`.

Use Case: Classifying charges with discrete thresholds

In commonly used water models such as TIP3P, a value of -0.83 e[47–49] is often assigned to the charge of oxygen atoms. Here, we create a linear PCovC map using the 3-body SOAP representations reported in Engel *et al.*[16] as **X**, this time using whether the charge of each oxygen atom is below (electron-rich) or above (electron-poor) the threshold of -0.83 e as **Y.** As shown in the histogram in Figure 8, the resulting dataset is highly imbalanced, with 71,515 oxygen atoms considered as electron-poor and only 145 considered as electron-rich.

Linear Discriminant Analysis (LDA)[50] is a commonly used method for identifying highly discriminative latent spaces, and it can be used in classification problems with such a severe class imbalance. However, LDA is only able to reduce data

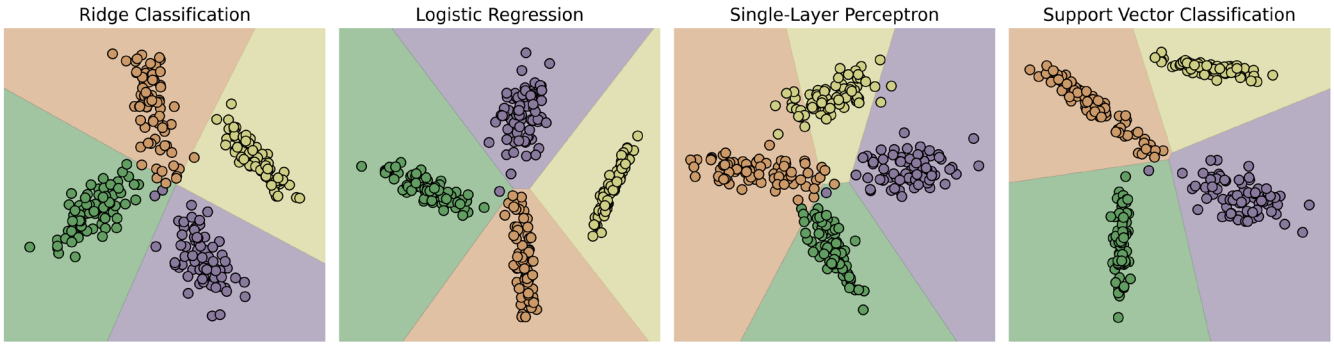

Figure 7. Influence of underlying PCovC classifier on latent space maps. We utilize ridge classification, logistic regression, a single-layer perceptron, and support vector classification to visualize the PCovC maps with α = 0.5. Point colors denote class; backgrounds show decision boundaries learned from fitting a separate instance of the classifier in the reduced feature set.

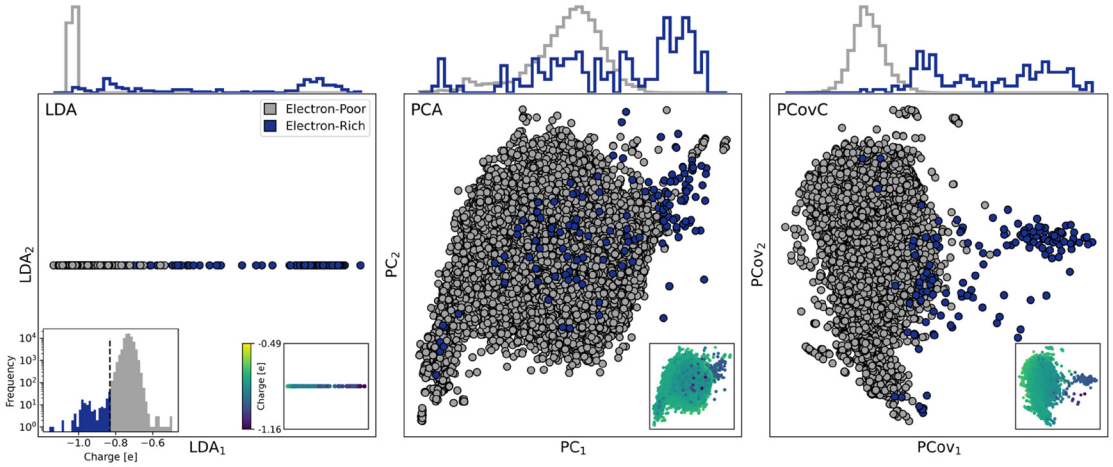

Figure 8. Linear Discriminant Analysis, Principal Component Analysis, and Principal Covariates Classification maps of Oxygens in Ice Structures. Environments deemed electron-poor (above the -0.83 *e* Mulliken charge threshold) are colored gray, and environments deemed oxygen-rich (below the -0.83 *e* threshold) are colored blue. The top corners of each plot show the Mulliken charges of each environment in the map, and the bottom left corner shows a histogram (with frequency in log scale) of the Mulliken charges, with a vertical line at -0.83 *e*.

to at most one dimension for a binary classification problem. Thus, while the LDA-reduced representation shown in Figure 8 is capable of distinguishing between electron-rich and electron-poor environments (yielding a classification F1 score of 76%), any dimensions of chemical diversity orthogonal to this representation are lost. Because of this, the interpretation of this map becomes difficult, as molecules that are similar in the LDA dimension, but are otherwise dissimilar, cannot be distinguished.

In contrast, PCA produces a map where electron-rich environments are interspersed with the electron-poor environments; in fact, building a logistic regression classifier on the PCA-reduced space results in an F1 score of 1.2%. PCovC (α = 0.5) using a linear SVC as its classifier, on the other hand, can identify a feature set that nearly matches the classification performance of LDA itself (F1 score of 71%) while also enabling further interpretation.

Use Case: Predicting World Bank Income Groups

In our modified WHO dataset, we use a subset of the original data to classify each country by their respective World Bank income group[30]. We also exclude GDP per capita from the original feature set, as the prediction of income group using GDP is trivial.

Using cross-validated logistic regression to predict income group yields a prediction accuracy of 73%. In Figure 9, reducing the dimensionality of the dataset to two features with PCA reduces the test accuracy of a logistic regression classifier to 61%. Using PCovC with a logistic regression classifier, on the other hand, raises the test accuracy to 72%, almost equal to the classification accuracy when using the full-dimensional space. We again observe that nonlinear methods outperform their linear counterparts. A kernel logistic regression model built with a RBF kernel results in a substantially higher test accuracy of 91%. While using Kernel PCA for dimensionality

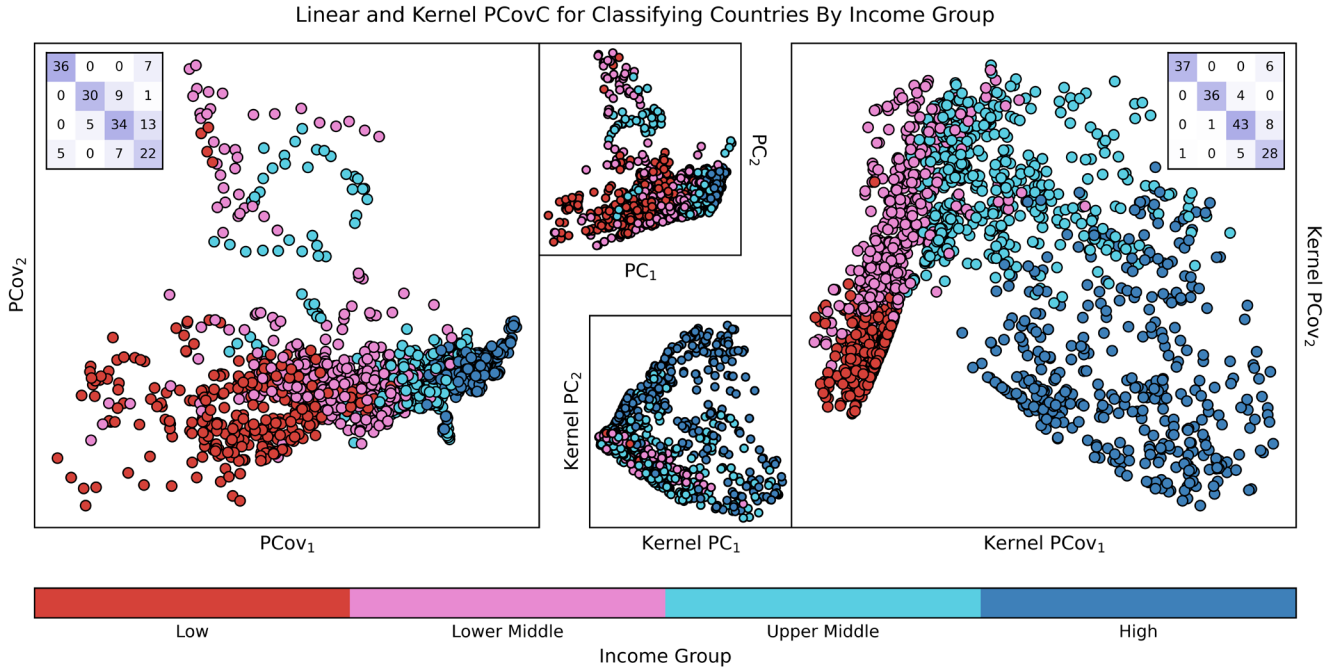

Figure 9. Linear and Non-linear Principal Components Analysis and Principal Covariates Classification Applied to the Modified WHO Dataset. Here, we show the maps for PCovC α = 0.5 (left), PCA (upper middle), KPCA (lower middle), and KPCovC α = 0.5 (right) on a dataset for nine statistical variables to predict income group for a given country. For each model, we used the same standardised matrix to determine a two-dimensional latent space mapping that was then used as input data for a logistic regression model. Confusion matrices for PCovC and KPCovC are displayed on the top left and top right corners of the respective plots; classification metrics were computed using a 90/10 train/test split, with results reported for only the testing split.

reduction results in a test accuracy of 45%, we again see that Kernel PCovC is able to outperform its linear counterpart in the subsequent classification task, resulting in a test accuracy of 85% with only two dimensions.

Consider the impact of this finding: using these kernel principal covariates, one can perform correlative analysis to determine the non-economic features that most influence income group categorization. This could not only reduce the need for computationally expensive GNI calculations, but it could also lead to insights into the demographic and other non-economic characteristics of a country that most impact their income group.

6 Feature and sample selection

In this section, we will detail two very different classes of selectors: 1) Farthest Point Sampling, CUR, and their PCovR adaptions, which are flexible score-based greedy selectors[3] that can be employed for either feature or sample selection; and 2) the directional convex hull, a non-greedy sample selection algorithm which is primarily used in chemical thermodynamic analyses.

[3] Greedy algorithms choose points iteratively based upon a scoring metric, whereas non-greedy algorithms simultaneously choose points that meet certain criteria.

The feature and sample selection methods contained in `scikit-matter` are both programmatically alike and contextually different. Therefore, we established a consistent API to allow users to apply these similar concepts in their different contexts, and we will start by reviewing this common implementation.

Implementation

The classes are exposed to the user via `skmatter.feature_selection` and `skmatter.sample_selection`. Each selector is passed an input of $\mathbf{X}$ (and $\mathbf{Y}$, when appropriate) to a `fit` function. This `fit` function will perform the sub-selection up until the desired number of selections (initialisation parameter `n_to_select`), the algorithm has terminated, or some score threshold has been reached (initialisation parameter `score_threshold`). Once the fitting is complete, the selector will contain the indices of the selections (relative to input matrices) in a member variable `selected_idx_`. The scores that evaluate all points in the context of these selections are available via the `score` function.

Farthest point sampling, CUR, and their PCovR-adaptations

Farthest point sampling (FPS) and CUR decomposition are available in unsupervised[51,52] or hybrid (PCovFPS and PCovCUR)[9] variants for both feature and sample selection. In the hybrid variant, similar to the dimensionality reduction

techniques discussed in Section "Linear and non-linear principal covariates regression" on Page 6, a parameter α is used to weight the supervised and unsupervised contributions in the covariance (for feature selection) or the Gram matrix (for sample selection) in the range from 0 to 1, with $\alpha = 1.0$ directly corresponding to the unsupervised version of each algorithm published in Imbalzano *et al.*[52]

FPS. In Farthest Point Sampling (FPS), we start with a random point, and at each iteration select the point that maximises the distance to the previously selected set. This min-max distance is traditionally called the Hausdorff distance, which serves as the FPS scoring metric. The unsupervised variant uses the feature-induced Euclidean distance as the default measure for the Hausdorff distance. The hybrid variant (PCovFPS) incorporates regression weights from an estimator into the distance calculation based on the relationship between Euclidean distance and the corresponding covariance or Gram matrices of the input matrix.

Computing the Hausdorff distance, in any variable space, constructs an implicit Voronoi tessellation[53] that can be exploited to reduce the amount of distance calculations throughout the selection procedure. At each iteration, only the selected points at the boundaries of the Voronoi polyhedra need to be considered for new distance calculations. We have implemented an FPS variant computation of Voronoi `skmatter.sample_selection.VoronoiFPS` which results in large speedups for sample selections in dense datasets, where the effects of this algorithm can be largest. Benchmark results comparing classical FPS and the Voronoi version can be found in the supporting information of Cersonsky *et al.*[9]

Workflow A typical workflow using FPS looks like this:

```
import skmatter.feature_selection.FPS
selector = FPS(
               n_to_select=4,
               progress_bar=True,
               score_threshold=1E-12,
               initialize=[0,1,2]
              )
selector.fit(X)
```

where the parameter `n_to_select` refers to the number of selections to be made, `progress_bar` signifies whether to show a `tqdm`-style progress bar[54], `score_threshold` is the score to terminate the selection at (potentially before `n_to_select`), and `initialise` signifies the index or indices with which to start the selection procedure.

CUR. selects features or samples based upon an "importance score" π, defined as the magnitude of a feature or sample's projection along the first k principal components (unsupervised) or covariates (hybrid PCovCUR) of the overall dataset[9,51,52]. π serves as the scoring metric for CUR. In the iterative implementation, at each iteration, the remaining features/samples are orthogonalised to the previous selection, as this ultimately minimises the mutual information in each subsequent selection[52].

Workflow A typical workflow using CUR looks like this:

```
import skmatter.feature_selection.CUR
selector = CUR(
               n_to_select=4,
               progress_bar=True,
               score_threshold=1E-12,
               k=1,
               recompute_every=1
              )
selector.fit(X)
```

where, similar to FPS, the parameter `n_to_select` refers to the number of selections to be made, `progress_bar` signifies whether to show a `tqdm`-style progress bar, and `score_threshold` is the score to terminate the selection at (potentially before `n_to_select`). `k` is the number of components or covariates to consider, and the parameter `recompute_every` indicates after how many iterations to orthogonalise the remaining features or samples. The latter can speed computation (at the cost of including multiple redundant features or samples), as orthogonalising and recomputing the π score are the most costly steps in the algorithm.

Use Case: Nearsightedness of lattice energies in ice structures

There are two interesting regimes for feature or sample selection – the practical reduction of data spaces for efficiently computing supervised models and the interpretability of a small handful of selected features or samples. As the former is thoroughly covered in Cersonsky *et al.*[9], we will focus on the latter here.

Here we employ the real space "Radial Spectrum" expanded on the O-H distances of the central oxygen environments for the stable ice crystals. The distribution of the $\rho_{OH}^{\otimes 1}(r)$ features is shown in the top panel of Figure 10.

We compare CUR with its hybrid variant PCovCUR to select 5 features. While the two selection methods choose a similar initial feature (corresponding to the hydrogen neighbours at $\approx$ 3.0 Å), the subsequent selected features diverge between the methods thereafter.

Within five selections, PCovCUR selects features that are capable of regressing the energies with fair accuracy, with an R^2 value of 0.96. PCovCUR selects features corresponding to the density values between the first and second (2nd and 4th feature), and the second and third neighbour shell (1st, 3rd and 5th feature). This makes sense from a physical perspective – the greatest proportion of the lattice energy in a molecular solid such as ice comes from the intermolecular interactions close to the central atom. Especially when working with highly redundant or non-efficient features, hybrid feature selection methods provide both improved performance and can reflect physical intuition.

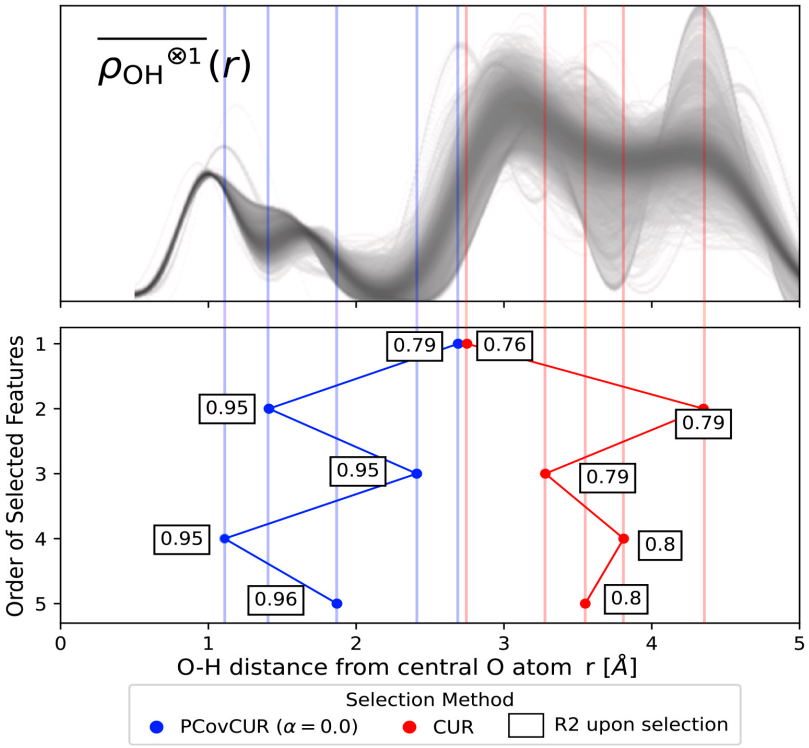

Figure 10. Selecting Features for Learning Lattice Energies. (Top) The real space "Radial Spectrum" $\overline{\rho_{OH}^{\otimes 1}}(r)$ for the hydrogen neighbours of central oxygen environments in the ice dataset. Each line corresponds to one oxygen environment. (Bottom) Selected features for both methods, noting the distance on the x-axis and selection method by colour (blue for PCovCUR and red for CUR). The number next to each point corresponds to the R^2 for the prediction of the lattice energies with a linear model upon selection of the feature.

Conversely, the unsupervised selection method CUR preferentially chooses the far-sighted features. This is explained by the fact that unsupervised methods will aim to maximise the resulting span of the feature space, and in this representation, far-sighted features show the greatest variance[4]. This also explains the poorer performance of regressions built on these features, which obtain a maximum R^2 value of 0.8.

Use Case: Choosing Features for Kernel Regression in the WHO Dataset

As shown in the previous use case, it is most effective to regress the life expectancies using a non-linear model. Here we show how feature selection mechanisms incorporating linear principal covariates yield comparable results to a recursive feature selection method based upon linear regression.

We use the previously detailed selection schemes and compare them to Recursive Feature Selection (RFS) based on linear regression on the training set. In the latter method, we iteratively select the feature that best improves regression performance, and this can serve as a putative "best selection" in datasets with these few features. For the FPS methods, we again choose the equivalent CUR-selected first feature to initialise the algorithm.

In each feature selection scheme, we select using the described methods and then compute the linear regression error for a test set. As shown in Figure 11a), the parameters with the greatest impact on the regression are a country's GDP and prevalence of HIV/AIDS. This is unsurprising, as it is well-recognised in the literature that the wealth of a country and the prevalence of HIV/AIDS in a country directly impacts the life expectancy[55–57]. This is shown in the jump in accuracy for PCov-FPS and PCov-CUR at the second selection as well as for FPS and CUR at the fifth and seventh selections, respectively. Otherwise, the most successful algorithms (RFS, PCov-CUR, and PCov-FPS) also incorporate different immunisations and health expenditures. The relationship between the selectors becomes more apparent when the similarity is plotted using the weighted Kendall's τ[58] as illustrated in Figure 11b). A strong similarity is observed between the selector and their PCovR adaption (e.g. CUR and PCovCUR) as well as among the supervised selectors (PCovCUR, PCovFPS and RFA). While the supervised selectors incorporate the same information of the targets to assess feature importance, the unsupervised selectors rely on different mathematical assumptions for assessment, resulting in lower similarity.

[4] To compensate for his effect, radial scaling is often applied to SOAP features, usually leading to improved regression of atomistic properties.

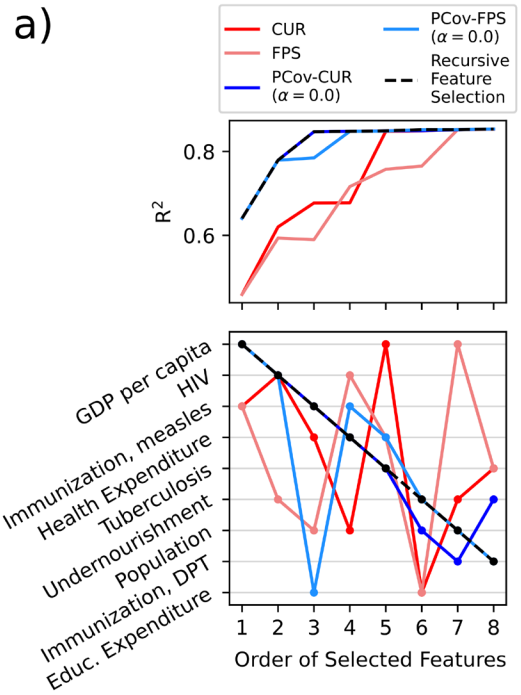

Figure 11. Feature Selection Methods Applied to the WHO Dataset. a) (top) Linear regression performance for the different selection methods with an increasing number of selected features. (bottom) Order of selected features for each selection method. We report the features in order of their selection by Recursive Feature Selection, which shows that the PCov-inspired methods most closely align with this "ground-truth". **b**) Similarity between the feature selectors using weighted Kendall's τ using the selection order as rank[58].

Directional convex hull

In the context of chemical sciences, a convex hull represents the set of structures that are thermodynamically stable from the perspective of a mixing model. In other words, if phases A, B, and C are of compatible stoichiometries, and A and B lie on the convex hull and phase C lies above, then C is unstable towards decomposition into a separated mixture of A and B (Figure 12).

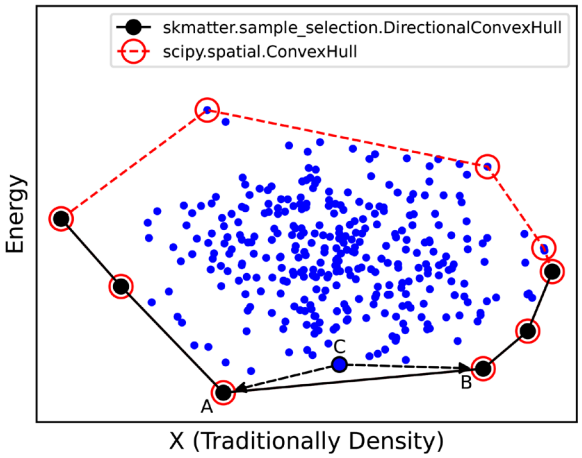

Figure 12. Schematic of a Convex Hull Construction. We first represent our data points across some variable (X, traditionally macroscopic density in thermodynamic convex hulls) and our energies. As shown by the black circles, our energetic convex hull lies at the bottom boundary of this plot. Programmatically, we use `scipy.spatial.ConvexHull` to determine the omnidirectional convex hull (red points), then select those simplices whose normal vector points negative to the energy. Any phase above the convex hull (e.g. **C**) will decompose into the two phases directly below it on the convex hull (e.g. **A** and **B**).

Usually, convex hulls are determined by a plot of the phases' densities and energies. The convex hull is the set of points sitting on the lower boundary of this plot. Anelli *et al.*[59] showed that the comparative dimension can be a generalised latent variable that spans the diversity of the given dataset, in their case, a `sketchmap` component[60].

Outside of chemical sciences, a convex hull is typically considered the set of points that define a bounding manifold, i.e., the vertices of the convex shape in which all other points are contained. This is implemented in Python in the package `scipy`[61] that employs `Qhull`[62]. We extend this functionality by allowing users to select the vertices subject to a directionality constraint (e.g., those points that define the surface below all other points with respect to a given observable). We have implemented this without explicit reference to chemical or energetic considerations, such that the `DirectionalConvexHull` can be used for similar tasks, including those shown in the mathematics[63] and economics[64] fields.

Implementation. In `scikit-matter`, `DirectionalConvexHull` is exposed through the `skmatter.sample_selection` submodule. In addition to the traditional input matrix and target vector, the `fit` function takes as input which columns of $\mathbf{X}$ to consider in the convex hull construction, where the default behaviour is to use the first column of $\mathbf{X}$.

During the `fit` function, we employ `scipy.spatial.ConvexHull` to determine the omnidirectional convex hull. We then determine which of these points lie on simplices

whose normal points downwards in the supplied property dimension. Like the selectors covered on Page 13, the indices of these points are saved into the class variable `selected_idx_`. Once the hull has been determined, the `score_samples` function determines the distance of given samples to the hull in the property dimension (i.e. a zero score implies that the point lies on the hull). One may also use the `score_feature_matrix` function to obtain the distance of samples to the hull in the dimensions *not* used to construct the directional convex hull.

Case Study: Determining the convex hull of ice structures. Engel *et al.*[16] determined a "generalised" convex hull that explicitly accounts for energetic and conformational uncertainty by using numerous trials of the algorithm detailed here. In each trial, they would perturb the atomic coordinates and energies of a selected number of structures within the uncertainty estimate of their calculations and compute the convex hull. They determined the convex hull as the points that are most often selected across many trials. Here, we will demonstrate one such trial by selecting the hull points using the REMatch kernel[65] as done in the original publication.

We build a two-component Kernel PCA on the normalised kernel matrix using the `scikit-learn` implementation. We construct our directional convex hull on these components, with the resulting hull shown in Figure 13.

In Figure 13, we see that the points that lie on the vertices of the directional convex hull are lower in energy relative to chemically similar points in their surroundings. From a thermodynamic standpoint, all structures that sit above the hull energetically will decompose into a mixture of the more stable phases, those on the vertices are more likely to be found in nature or be candidates for experimental synthesis. Although the KPCA features may appear abstract, they often correlate with characteristics of the structures, which are qualitatively analysable through chemiscope[46] or similar visualisation suites.

7 Conclusions

In this work, we have demonstrated the use of `scikit-matter`, a `scikit-learn` extension that focuses on functions of particular relevance in materials science and chemistry. As the examples on the WHO dataset show, the features considered need not be chemicallybased correlation functions but might be economic predictors, diagnostic test results, or the engineered features of different autoencoder infrastructures. This illustrates how the compatibility of `scikit-matter` with `scikit-learn` allows a frictionless embedding of the demonstrated implementations into data-driven workflows in any domain. This has the capability of not only introducing beneficial algorithms to users who may be unfamiliar with them but also reducing the time-consuming implementation of these methods for knowledgeable users, thereby accelerating research endeavours across a wide range of fields. The module also provides a basis for further methods developed in materials science and chemistry communities to be integrated into a stable library in the future.

Installing `scikit-matter`

`scikit-matter` is available via the Python Package Index (PyPi)[66], Anaconda[67] and through GitHub. Independent

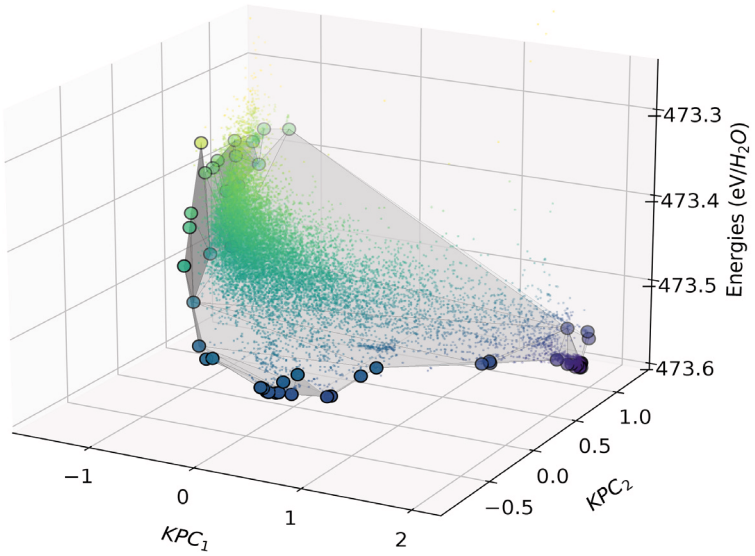

Figure 13. The directional convex hull of ice structures. Here we show a directional convex hull, constructed using the first two KPCA dimensions as features and the per-molecule energies of ice structures as the target property. The small points correspond to all the structures in the dataset, whereas the larger points correspond to those that lie on the convex hull (shown by the grey surface), as determined using the `fit` function of `scikit-matter`.

developers are encouraged to contribute and can find more information on how to do so on the package documentation at `scikit-matter.readthedocs.io`.

Data availability

Materials Cloud: Mapping uncharted territory in ice from zeolite networks to ice structures.

DOI: 10.24435/materialscloud:2018.0010/v1. Engel *et al.*[18]

The project employs the following underlying data:

- `dataset.xyz`: Structures in XYZ format for 15,869 geometry-optimised ice structures. Optimisation was carried out using PBE-DFT in the CASTEP code, with a plane-wave energy cut-off of 490 eV, maximum k-point spacing of $2\pi \times 0.07$ Å^{-1}, and on-the-fly generated ultrasoft pseudopotentials.
- `properties.dat`: Properties of all ice structures in `dataset.xyz`, including the CSD identifier, number of atoms per unit cell, density, configurational energy, and energy with respect to the energy-density convex hull.

We also employ publicly available datasets provided by The World Bank:

- Life expectancy at birth, total (years), World Bank[20].
- Population, total, World Bank[21].
- GDP per capita (current USD), World Bank[22].
- Current health expenditure (percentage of GDP), World Bank[23].
- Government expenditure on education, total (percentage of GDP)[24].
- Prevalence of HIV, total (percentage of population ages 15–49) World Bank[25].
- Incidence of tuberculosis (per 100,000 people) World Bank[26].
- Immunization, measles (percentage of children ages 12–23 months), World Bank[27].
- Immunization, DPT (percentage of children ages 12–23 months) World Bank[28].
- Prevalence of undernourishment (percentage of population) World Bank[29].

The aggregated WHO dataset can be retrieved through `skmatter.datasets.load_who_dataset`.

We also employ the following World Bank data for use as categorical targets exclusively in the Section "Linear and non-linear principal covariates classification" on Page 9:

- World Bank Country and Lending Groups, World Bank[30].

Data are available under the terms of the Creative Commons Attribution 4.0 International license (CC-BY 4.0).

The scripts together with the required software packages used to generate the figures are available in the repository https://github.com/lab-cosmo/skmatter-ore in the form of executed jupyter notebooks with output. The code is available under the 3-Clause BSD License.

Acknowledgements

We acknowledge contributions from the Laboratory of Computational Science and Modeling, particularly Sergey Podznyakov, Giulio Imbalzano, and Max Veit.

References

1. Pedregosa F, Varoquaux G, Gramfort A, *et al.*: **Scikit-learn: machine learning in Python.** *J Mach Learn Res.* 2011; **12**: 2825–2830.
Reference Source
2. Buitinck L, Louppe G, Blondel M, *et al.*: **API design for machine learning software: experiences from the scikit-learn project.** In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. 2013; 108–122.
Reference Source
3. Shapeev AV: **Moment tensor potentials: a class of systematically improvable interatomic potentials.** *Multiscale Model Simul.* 2016; **14**(3): 1153–1173.
Publisher Full Text
4. Drautz R: **Atomic cluster expansion for accurate and transferable interatomic potentials.** *Phys Rev B.* 2019; **99**(1): 014104.
Publisher Full Text
5. Deringer VL, Bartók AP, Bernstein N, *et al.*: **Gaussian process regression for materials and molecules.** *Chem Rev.* 2021; **121**(16): 10073–10141.
PubMed Abstract | Publisher Full Text | Free Full Text
6. Musil F, Grisafi A, Bartók AP, *et al.*: **Physics-inspired structural representations for molecules and materials.** *Chem Rev.* 2021; **121**(16): 9759–9815.
PubMed Abstract | Publisher Full Text
7. Bartók AP, De S, Poelking C, *et al.*: **Machine learning unifies the modeling of materials and molecules.** *Sci Adv.* 2017; **3**(12): e1701816.
PubMed Abstract | Publisher Full Text | Free Full Text
8. Willatt MJ, Musil F, Ceriotti M: **Feature optimization for atomistic machine learning yields a data-driven construction of the periodic table of the elements.** *Phys Chem Chem Phys.* 2018; **20**(47): 29661–29668.
PubMed Abstract | Publisher Full Text

9. Cersonsky RK, Helfrecht BA, Engel EA, *et al.*: **Improving sample and feature selection with principal covariates regression.** *Mach Learn Sci Technol.* 2021; **2**(3): 035038.
Publisher Full Text

10. Parsaeifard B, De DS, Christensen AS, *et al.*: **An assessment of the structural resolution of various fingerprints commonly used in machine learning.** *Mach Learn Sci Technol.* 2021; **2**(1): 015018.
Publisher Full Text

11. Goscinski A, Fraux G, Imbalzano G, *et al.*: **The role of feature space in atomistic learning.** *Mach Learn Sci Technol.* 2021; **2**(2): 025028.
Publisher Full Text

12. Helfrecht BA, Cersonsky RK, Fraux G, *et al.*: **Structure-property maps with kernel principal covariates regression.** *Mach Learn Sci Technol.* 2020; **1**(4): 045021.
Publisher Full Text

13. Behler J: **RuNNer**.
Reference Source

14. Bartók-Pártay A, Bartók-Pártay L, Bianchini F, *et al.*: **libAtoms+QUIP**. 2020.
Reference Source

15. Novikov IS, Gubaev K, Podryabinkin EV, *et al.*: **The MLIP package: moment tensor potentials with MPI and active learning.** *Mach Learn Sci Technol.* 2021; **2**(2): 025002.
Publisher Full Text

16. Engel EA, Anelli A, Ceriotti M, *et al.*: **Mapping uncharted territory in ice from zeolite networks to ice structures.** *Nat Commun.* 2018; **9**(1): 2173.
PubMed Abstract | Publisher Full Text | Free Full Text

17. Talirz L, Kumbhar S, Passaro E, *et al.*: **Materials cloud, a platform for open computational science.** *Sci Data.* 2020; **7**(1): 299.
PubMed Abstract | Publisher Full Text | Free Full Text

18. Engel EA, Anelli A, Ceriotti M, *et al.*: **Mapping uncharted territory in ice from zeolite networks to ice structures.** 2018.
Reference Source

19. Hourahine B, Aradi B, Blum V, *et al.*: **DFTB+, a software package for efficient approximate density functional theory based atomistic simulations.** *J Chem Phys.* 2020; **152**(12): 124101.
PubMed Abstract | Publisher Full Text

20. World Bank: **Life expectancy at birth, total (years).** Technical report, The World Bank, Washington, DC, 2023.
Reference Source

21. World Bank: **Population, total.** Technical report, The World Bank, Washington, DC, 2023.
Reference Source

22. World Bank: **Gdp per capita (current us$).** Technical report, The World Bank, Washington, DC, 2023.
Reference Source

23. World Bank: **Current health expenditure (% of gdp).** Technical report, The World Bank, Washington, DC, 2023.
Reference Source

24. World Bank: **Government expenditure on education, total (% of gdp).** Technical report, The World Bank, Washington, DC, 2023.
Reference Source

25. World Bank: **Prevalence of HIV, total (% of population 15-49).** Technical report, The World Bank, Washington, DC, 2023.
Reference Source

26. World Bank: **Incidence of tuberculosis (per 100,000 people).** Technical report, The World Bank, Washington, DC, 2023.
Reference Source

27. World Bank: **Immunization, measles (% of children ages 12-23 months).** Technical report, The World Bank, Washington, DC, 2023.
Reference Source

28. World Bank: **Immunization, dpt (% of children ages 12-23 months).** Technical report, The World Bank, Washington, DC, 2023.
Reference Source

29. World Bank: **Prevalence of undernourishment (% of population).** Technical report, The World Bank, Washington, DC, 2023.
Reference Source

30. World Bank: **Country and Lending Groups.** Technical report, The World Bank, Washington, DC, 2025.
Reference Source

31. World Bank: **How does the World Bank classify countries?** The World Bank, Washington, DC, 2025.
Reference Source

32. Bartók AP, Kondor R, Csányi G: **On representing chemical environments.** *Phys Rev B.* 2013; **87**(18): 184115.
Publisher Full Text

33. Capecchi A, Probst D, Reymond JL: **One molecular fingerprint to rule them all: drugs, biomolecules, and the metabolome.** *J Cheminform.* 2020; **12**(1): 43.
PubMed Abstract | Publisher Full Text | Free Full Text

34. Prodan E, Kohn W: **Nearsightedness of electronic matter.** *Proc Natl Acad Sci U S A.* 2005; **102**(33): 11635–8.
PubMed Abstract | Publisher Full Text | Free Full Text

35. Caro MA: **Optimizing many-body atomic descriptors for enhanced computational performance of machine learning based interatomic potentials.** *Phys Rev B.* 2019; **100**(2): 024112.
Publisher Full Text

36. Kermode JR: **QUIP.** 2008.
Reference Source

37. Csányi G, Winfield S, Kermode JR, *et al.*: **Expressive programming for computational physics in fortran 95+.** *IoP Comp Phys Newsletter.* 2007.

38. Kermode JR: **f90wrap: an automated tool for constructing deep python interfaces to modern fortran codes.** *J Phys Condens Matter.* 2020; **32**(30): 305901.
PubMed Abstract | Publisher Full Text

39. Himanen L, Jäger MOJ, Morooka EV, *et al.*: **DScribe: library of descriptors for machine learning in materials science.** *Comput Phys Commun.* 2020; **247**: 106949.
Publisher Full Text

40. Ceriotti M, Emsley M, Paruzzo F, *et al.*: **Chemical shifts in molecular solids by machine learning datasets.** *Materials Cloud Archive.* 2019.
Publisher Full Text

41. Goscinski A, Musil F, Pozdnyakov S, *et al.*: **Optimal radial basis for density-based atomic representations.** *J Chem Phys.* 2021; **155**(10): 104106.
PubMed Abstract | Publisher Full Text

42. de Jong S, Kiers HAL: **Principal covariates regression: part I. Theory.** *Chemometr Intell Lab Syst.* 1992; **14**(1–3): 155–164.
Publisher Full Text

43. Schölkopf B, Smola A, Müller KR: **Nonlinear component analysis as a kernel eigenvalue problem.** *Neural Comput.* 1998; **10**(5): 1299–1319.
Publisher Full Text

44. Cersonsky TEK, Cersonsky RK, Saade GR, *et al.*: **Placental lesions associated with stillbirth by gestational age, according to feature importance: results from the Stillbirth Collaborative Research Network.** *Placenta.* 2023; **137**: 59–64.
PubMed Abstract | Publisher Full Text | Free Full Text

45. Jorgensen C, Lin AY, Vasavada R, *et al.*: **Interpretable visualizations of data spaces for classification problems.** *arXiv.* 2025; 2503.05861.
Publisher Full Text

46. Fraux G, Cersonsky RK, Ceriotti M: **Chemiscope: interactive structure-property explorer for materials and molecules.** *J Open Source Softw.* 2020; **5**(51): 2117.
Publisher Full Text

47. Jorgensen WL, Chandrasekhar J, Madura JD, *et al.*: **Comparison of simple potential functions for simulating liquid water.** *J Chem Phys.* 1983; **79**(2): 926–935.
Publisher Full Text

48. MacKerell AD Jr, Bashford D, Bellott M, *et al.*: **All-atom empirical potential for molecular modeling and dynamics studies of proteins.** *J Phys Chem B.* 1998; **102**(18): 3586–616.
PubMed Abstract | Publisher Full Text

49. Price DJ, Brooks CL III: **A modified TIP3P water potential for simulation with Ewald summation.** *J Chem Phys.* 2004; **121**(20): 10096–103.
PubMed Abstract | Publisher Full Text

50. Hastie T, Tibshirani R, Friedman J: **The elements of statistical learning.** Springer Series in Statistics. Springer, New York, NY, 2009.
Publisher Full Text

51. Mahoney MW, Drineas P: **CUR matrix decompositions for improved data analysis.** *Proc Natl Acad Sci U S A.* 2009; **106**(3): 697–702.
PubMed Abstract | Publisher Full Text | Free Full Text

52. Imbalzano G, Anelli A, Giofré D, *et al.*: **Automatic selection of atomic fingerprints and reference configurations for machine-learning potentials.** *J Chem Phys.* 2018; **148**(24): 241730.
PubMed Abstract | Publisher Full Text

53. Du Q, Faber V, Gunzburger M: **Centroidal voronoi tessellations: applications and algorithms.** *SIAM review.* 1999; **41**(4): 637–676.
Publisher Full Text

54. da Costa-Luis C, Larroque SK, Altendorf K, *et al.*: **tqdm: a fast, extensible progress bar for Python and CLI.** *Zenodo.* 2022.
Publisher Full Text

55. Mathers CD, Sadana R, Salomon JA, *et al.*: **Healthy life expectancy in 191 countries, 1999.** *Lancet.* 2001; **357**(9269): 1685–1691.
PubMed Abstract | Publisher Full Text

56. Ashford LS: **How HIV and AIDS affect populations.** *World.* 2006; **1**: 38–600.
Reference Source

57. Hansen CW: **The relation between wealth and health: evidence from a world panel of countries.** *Econ Lett.* 2012; **115**(2): 175–176.
Publisher Full Text

58. Shieh GS: **A weighted Kendall's tau statistic.** *Stat Probab Lett.* 1998; **39**(1): 17–24.
Publisher Full Text

59. Anelli A, Engel EA, Pickard CJ, *et al.*: **Generalized convex hull construction for materials discovery.** *Phys Rev Materials.* 2018; **2**(10): 103804.
Publisher Full Text

60. Ceriotti M, Tribello GA, Parrinello M: **Simplifying the representation of complex free-energy landscapes using sketch-map.** *Proc Natl Acad Sci U S A.* 2011; **108**(32): 13023–13028.
PubMed Abstract | Publisher Full Text | Free Full Text

61. Virtanen P, Gommers R, Oliphant TE, *et al.*: **SciPy 1.0: fundamental algorithms for scientific computing in Python.** *Nat Methods.* 2020; **17**(3): 261–272.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**

62. Barber CB, Dobkin DP, Huhdanpaa H: **The quickhull algorithm for convex hulls.** *ACM Trans Math Softw (TOMS).* 1996; **22**(4): 469–483.
Publisher Full Text

63. Liu W, Yuen SY, Chung KW, *et al.*: **A general-purpose multi-dimensional convex landscape generator.** *Mathematics.* 2022; **10**(21): 3974.
Publisher Full Text

64. Anderson G, Crawford I, Leicester A: **Efficiency analysis and the lower convex hull approach**. *Quantitative Approaches to Multidimensional Poverty Measurement*. 2008; 176–191.
Publisher Full Text

65. De S, Bartók AP, Csányi G, *et al.*: **Comparing molecules and solids across structural and alchemical space.** *Phys Chem Chem Phys.* 2016; **18**(20): 13754–13769.
PubMed Abstract | **Publisher Full Text**

66. **Python package index - pypi.** 2003.
Reference Source

67. **Anaconda software distribution**. 2020.
Reference Source

Open Peer Review

Current Peer Review Status: ✔ ✔ ✔ ✔

Version 3

Reviewer Report 10 January 2026

https://doi.org/10.21956/openreseurope.23357.r66960

© **2026 Izgorodina E.** This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

✔ **Ekaterina I. Izgorodina**
Monash Computational Chemistry Group, School of Chemistry, Monash University, Clayton, Victoria, Australia

The authors fully addressed the raised comments.

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Theoretical and computational chemistry, development of ML/AI algorithms to study thermodynamics and kinetics of chemical reactions and predict molecular and thermodynamic properties of individual molecules, large-scale systems and chemical reactions

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Version 2

Reviewer Report 21 February 2024

https://doi.org/10.21956/openreseurope.17559.r36893

© **2024 Meissner R.** This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

✔ **Robert Meissner**
Hamburg University of Technology, Hamburg, Germany

Along with their manuscript, the authors provide a solid and robust software extension, fully in the

spirit of the well-known scikit-learn package, and dedicated to extending those functions "that are actively used in the field of computational science and modeling of materials and chemical systems". The work presented here not only focuses on methods that are particularly relevant to materials science and chemistry, but also extends its methods to relevant data outside of the field of materials science and chemistry. Installing this extension using PIP or Anaconda is easy even for novice Pythonistas. Apart from that, a cornerstone of such an implementation is well-done documentation and relevant but easy-to-understand examples. While the indeed excellent documentation is available on the package website, the latter mentioned examples are not only illustrated very well in the manuscript, but are also provided as easy-to-use Jupyter notebooks that can be downloaded from the website.

I have read the previous reports carefully and both have done a very thorough investigation and I do not have much to add to their review. I agree that especially the documentation is important and the authors have done a good job here, while also trying their best to make their code accessible by providing easy-to-use but relevant examples. Their brief explanation of the available methods in the manuscript is sufficient and summarizes the key principles of the underlying methods very well, while always providing additional literature for the more curious reader.

I followed the previous discussion about more general frameworks like SMILES strings and whether they could be discussed in addition to the density-based approach present in the manuscript. SMILES and related techniques, such as SMARTS and SMIRKS, for finding patterns in molecular graphs or describing reactions, are a very active and important field in computational chemistry. However, SMILES have the disadvantage that they are not necessarily unique. Several SMILES strings represent the same molecule. This is solved by using canonical SMILES. However, they still depend on the specific canonicalization algorithm used to generate them. This can cause problems when used with natural language processing (NLP) algorithms if this redundancy is not properly handled in the raw data or if different (SMILES) data sources are used. In my opinion, the concept of SMILES should not be included in this work, as useful as they are, as they are conceptually too different from what is presented here and usually require other approaches, e.g. NLP.

However, to perhaps answer Prof. Izgorodina's question: It is possible to use SMILES strings in a high-throughput manner, called STaGE (Small molecule Topology GEnerator), to generate reliable 3D molecular structures and calculate from those corresponding physical properties such as solvation free energies ref [1]. Perhaps this is a bit of a niche application, but as we have shown, SMILES can thus be used in combination with STaGE, SOAP and Kernel Ridge Regression methods ref [2] to predict corrosion properties based on the resulting molecular similarities. Thus, I think it is a bit unfair to conclude from the statement that SOAP is not well-known outside the field of atomistic machine learning field that this is "a very small community" and that "this feature may not be of interest to a much broader community working on the discovery of new materials".

Leaving the important area of string representations for now, scikit-matter has the potential to make density-based approaches, e.g. SOAP, and related ML approaches more accessible to a wider community. Moreover, the techniques presented here are made available to other fields as well, since they do not necessarily have to be used in the context of density-based representations, as impressively demonstrated with the WHO data on life expectancy. This efforts of the authors to make their computational approaches more accessible to the wider community are much appreciated.

I have a couple of minor things which might be considered:

- I wonder if the labeling of the y-axis in 2b) is correct. Is the notation GFRE 100->100 correct or is it related to the statement "The numerical accuracy of the solver is computed with GFRE for the identity relationship for 100 basis functions"? This may need further explanation.
- What is the underlying toy model of Figure 3? Perhaps it would be helpful to mention roughly what the colors in Figure 3 mean in the caption to help understand what the figure is supposed to demonstrate.
- PCovCUR, PCov-CUR, PCovFPS and PCov-FPS are not consistently used.

References

1. Lundborg M, Lindahl E: Automatic GROMACS topology generation and comparisons of force fields for solvation free energy calculations.*J Phys Chem B*. 2015; **119** (3): 810-23 PubMed Abstract | Publisher Full Text
2. Würger T, Mei D, Vaghefinazari B, Winkler D, et al.: Exploring structure-property relationships in magnesium dissolution modulators. *npj Materials Degradation*. 2021; **5** (1). Publisher Full Text

Is the rationale for developing the new software tool clearly explained?

Yes

Is the description of the software tool technically sound?

Yes

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?

Yes

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?

Yes

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Molecular Dynamics, Atomistic Machine Learning, Electrochemistry

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Reviewer Report 16 January 2024

https://doi.org/10.21956/openreseurope.17559.r36900

© 2024 DeCost B. This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Brian DeCost
National Institute of Standards and Technology, Gaithersburg, Maryland, USA

This manuscript presents an accessible and composable materials modeling interface that is compatible with a large sector of the general python-based data driven modeling ecosystem. The library currently focuses on materials-informatics-specific workflows for feature and sample selection and model evaluation, as well as high-level compatibility with the diverse set of quickly-evolving materials modeling methodologies. The manuscript is clearly presented, and the online documentation is complete, clear, and presented. The library will play an important role in helping the materials and chemical informatics community develop and apply data-driven research methods efficiently and correctly, which will have a large impact across this growing field.

Is the rationale for developing the new software tool clearly explained?
Yes

Is the description of the software tool technically sound?
Yes

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Yes

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Yes

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Computational materials science; applied machine learning

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Reviewer Report 16 October 2023

https://doi.org/10.21956/openreseurope.17559.r35168

© 2023 Rosen A. This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Andrew S. Rosen
Department of Materials Science and Engineering, University of California, Berkeley, Berkeley, CA, USA

I thank the authors for their detailed and valuable response. They have addressed all of my requested (minor) changes, and I am very happy to formally "approve" this article. Thank you for the excellent contribution, which I am confident will greatly benefit the ML community!

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Computational materials chemistry, quantum-chemical methods, machine learning for materials discovery.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Version 1

Reviewer Report 17 July 2023

https://doi.org/10.21956/openreseurope.17055.r31732

© 2023 Izgorodina E. This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

?

Ekaterina I. Izgorodina
Monash Computational Chemistry Group, School of Chemistry, Monash University, Clayton, Victoria, Australia

The manuscript by Alexander Goscinski et al. “scikit-matter: A Suite of Generalisable Machine Learning Methods Born out of Chemistry and Materials Science” describes the implementation of four new features within scikit-matter that are applicable to study materials science. These features include feature reconstruction measures, Linear and Kernelized Principal Covariates Regression, Farthest Point Sampling, CUR, and their PCovR-Adaptations and Directional Convex Hull Construction. I enjoyed reading the manuscript and I think it has successfully showcased how new features can further improve the discovery of new materials. There are still several minor issue that must be addressed prior to indexing.

All subheadings are formatted equally which is rather confusing. For each which any subsection included within the feature section could be represented differently. For example, a smaller sized and italicised font could be used instead.

In terms of the first feature discussed in the manuscript – feature reconstruction measures – gives an example of how the number of basis functions used to represent neighbouring shells of water

molecules could be much easier minimised by using this feature. The feature is based around the SOAP representation of a chemical system, in which each atom in the system is represented with a Gaussian density. I was wondering if a more general example could be given here. As the authors pointed out themselves, the SOAP representation is not well known outside of a very small community and therefore, this feature may not be of interest to a much broader community working on the discovery of new materials. Majority of scientists in the field use the smiles string representation to predict descriptors to be used in the discovery process. I was wondering if this feature could be used in combination with the smiles string to generate a density-based representation of the system.

Figure 5 is a little confusing. The authors argue that the combination of the PCovR kernel can distinguish among varying oxygen environments – hydroxide, hydronium cation and different arrangements of water. Can the three environments be elaborated and accompanied with corresponding figures that showcase difference in the oxygen environments. Equally important, it should be clearly stated how these environments were selected and why Mulliken charges were used for this purpose. I am a little hesitant about a wider applicability of Mulliken charges as this is one of the most approximate schemes to predict partial atomic charges and strongly relies on the basis set used. The last point should be comment on in the manuscript.

Figure 8 is difficult to understand. It took me a while to realise that essentially the order of the parameters on the y-axis are given with respect to the best performing model proposed in the manuscript – Recursive Feature Selection. The “jagged” trends for other models are slightly confusing as the clearly favoured a different order of the parameters. I was wondering gif there was a better and clear way to represent these data in a Figure.

Is the rationale for developing the new software tool clearly explained?

Yes

Is the description of the software tool technically sound?

Yes

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?

Yes

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?

Yes

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Theoretical Chemistry, Molecular Dynamics and Machine Learning

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 25 Aug 2023

Alexander Goscinski

The manuscript by Alexander Goscinski et al. "scikit-matter: A Suite of Generalisable Machine Learning Methods Born out of Chemistry and Materials Science" describes the implementation of four new features within scikit-matter that are applicable to study materials science. These features include feature reconstruction measures, Linear and Kernelized Principal Covariates Regression, Farthest Point Sampling, CUR, and their PCovR-Adaptations and Directional Convex Hull Construction. I enjoyed reading the manuscript and I think it has successfully showcased how new features can further improve the discovery of new materials. There are still several minor issue that must be addressed prior to indexing.

We thank Prof. Izgorodina for her kind words, and are happy that she finds the publication useful!

All subheadings are formatted equally which is rather confusing. For each which any subsection included within the feature section could be represented differently. For example, a smaller sized and italicised font could be used instead.

We changed the structure of the paper by adding a numeration to the sections as well as removing the method section to move all features one level higher. For example, the methods "FPS" and "CUR" are now subsubsections so that the paragraph "Workflow" can be clearly identified as part of the corresponding method.

In terms of the first feature discussed in the manuscript – feature reconstruction measures – gives an example of how the number of basis functions used to represent neighbouring shells of water molecules could be much easier minimised by using this feature. The feature is based around the SOAP representation of a chemical system, in which each atom in the system is represented with a Gaussian density. I was wondering if a more general example could be given here. As the authors pointed out themselves, the SOAP representation is not well known outside of a very small community and therefore, this feature may not be of interest to a much broader community working on the discovery of new materials. Majority of scientists in the field use the smiles string representation to predict descriptors to be used in the discovery process. I was wondering if this feature could be used in combination with the smiles string to generate a density-based representation of the system.

The main benefit of SMILES strings is to leverage advances in natural-language processing for chemical systems. While there are possibilities for scikit-matter to support NLP-like functionalities, currently scikit-learn is not the primary package used for NLP workflows, and therefore we wouldn't necessarily argue that scikit-matter should first-and-foremost cater to NLP models. That being said, there are many ways to embed SMILES strings into numeric representations, and, in this context, we agree with the reviewer that we could make an effort to include the greater community in these case studies. We have added some notes to sections on how SMILES-like workflows could be augmented with these methods:
Throughout the paper we use the Smooth Overlap of Atomic Positions (SOAP) representation to

numerically encode our chemical data\cite{soap}. We note however, that other numerical encodings of chemical information are compatible with the methods presented here\cite{capecchi_one_2020}. Unfortunately, SMILES strings often lack the configurational information to generate a density-based representation, and there is an active, non-trivial effort in the field to predict thermodynamically-sensible configurations from connectivity information (e.g. SMILES), see DOIs 10.1039/d0cp04670a and 10.48550/arXiv.2106.07802. We feel that in order to include a useful example based on SMILES-type descriptors we would have to focus on the methodology to translates them into continuous features, which is, in our view, a pre-processing step that is orthogonal to the methods that are currently included in the package.

Figure 5 is a little confusing. The authors argue that the combination of the PCovR kernel can distinguish among varying oxygen environments – hydroxide, hydronium cation and different arrangements of water. Can the three environments be elaborated and accompanied with corresponding figures that showcase difference in the oxygen environments

We thank Prof. Izgorodina for her comment – Figure 5 is meant to showcase the difference in chemical analyses between PCA and PCovR; PCovR can incorporate target information into structure-property maps, whereas PCA will be entirely chemically-determined. The importance here is not the environments themselves, but what we would consider “similar” environments, as visualization and mapping techniques often aim to represent similar data points nearby each other in latent space. When aiming to identify QSPR relationships, PCA is often invoked; however, PCovR is directly suited to this task, as sometimes, mappings which best represent chemical similarity (such as here) do not reflect similarity in properties as well. We invite the reviewer to explore herself the difference in the relationship using the interactive demo https://chemiscope.org/?load=https://github.com/lab-cosmo/skmatter-ore/raw/main/paper/chemiscope-oxygen-ice-structures-small.json.gz

Equally important, it should be clearly stated how these environments :were selected and why Mulliken charges were used for this purpose. I am a little hesitant about a wider applicability of Mulliken charges as this is one of the most approximate schemes to predict partial atomic charges and strongly relies on the basis set used. The last point should be comment on in the manuscript.

Here we use Mulliken charges not to make any larger scientific claim, but rather to provide a reasonable and easily-understandable target quantity for PCovR for a general audience. Please see our comments to Prof. Rosen on the exact computation and provenance of these quantities.

Figure 8 is difficult to understand. It took me a while to realise that essentially the order of the parameters on the y-axis are given with respect to the best performing model proposed in the manuscript – Recursive Feature Selection. The “jagged” trends for other models are slightly confusing as the clearly favoured a different order of the parameters. I was wondering gif there was a better and clear way to represent these data in a Figure.

During the creation of the Figure we tried different orders of the y-axis and came to the conclusion that the order corresponding to the best regression performance is the clearest as this gives an interpretable importance score of the features. As the red methods are purely based on unsupervised information, it is clear that the order does not need to be correlated with RFE, therefore resulting in the "jagged" trend. We explained this in the text more precisely *The relationship between the selectors becomes more apparent the similarity is plotted using the weighted Kendall's τ\cite{shieh_weighted_1998} as illustrated in \figref{fig:who_sel}b). A strong similarity is observed between the selector and their PCovR adaption (e.g. CUR and PCovCUR) as well as among the supervised selectors (PCovCUR, PCovFPS and RFA). While the supervised selectors incorporate the same information of the targets to assess feature importance, the unsupervised selectors rely on different mathematical assumptions for assessment, resulting in lower similarity.* On the other hand the feature selection schemes that are based solely on the target information (blue) are quite strongly correlating with the recursive feature selection. We updated the figure so this can be seen much easier. For that we changed the model to a linear model to remove the effect of nonlinearities in the feature selection process. In addition we added a heatmap of similarity between the selected features of the selector using the selected order of the corresponding feature selection method as rank. In this heatmap the correlation can be seen clearly.

Competing Interests: No competing interests were disclosed.

Reviewer Report 19 June 2023

https://doi.org/10.21956/openreseurope.17055.r31730

© **2023 Rosen A.** This is an open access peer review report distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Andrew S. Rosen
Department of Materials Science and Engineering, University of California, Berkeley, Berkeley, CA, USA

Overall Comments

In their submitted work, Goscinski et al. have described and released `scikit-matter`, a Python package built around the `scikit-learn` API that provides a suite of feature selection/analysis tools that are likely to be of particular interest to the computational materials science community. After going through the paper and corresponding code, I am confident that this work warrants acceptance in Open Research Europe. To the best of my knowledge, there is no comparable package published to date, and the code itself fills a niche between pure ML regression codes like `scikit-learn` and ML featurization codes like `QUIP` and `DScribe`.

In short, I recommend this work for acceptance in Open Research Europe with only minor revisions, mainly focused on making the documentation more accessible to a reader who may not

(yet) be familiar with the underlying techniques and perhaps has not (yet) read the corresponding paper. I also have a few very minor comments and questions about the paper, as described below.

Congratulations to the authors on this nice work, and I look forward to seeing `scikit-matter` continue to evolve in the future.

The Paper

Overall Comments

Overall, I found the paper to be excellent. I learned a lot from it, and I thought it was exceptionally clear. I found the examples to be very insightful, such as the convergence test for the number of basis functions and the convex hull analysis broken down by the principal components. The paper (and corresponding code) strike a unique and admittedly challenging balance between being domain-agnostic yet also catering to methods that are likely to be especially useful to computational material scientists. I do not have any major comments on the paper, which I think will be a very useful reference for machine learning practitioners and students alike.

Feature Reconstruction Measures

This section was very clear and effective! As someone who has not used feature reconstruction measures before, one of the questions I had when reading the text was the following: "for a given dataset of interest, wouldn't doing a standard evaluation of the model cross-validation error for two separate representations also tell me which is more information-rich?" With this in mind, perhaps it might be worth taking a sentence to explain the major benefit of the feature reconstruction method approach. Is it simply that it's a more direct measure of the ability of the representation to reproduce the true data?

The authors state that "As most interesting applications use large feature vectors, we implemented a custom RidgeRegression2FoldCV in the skmatter.linear_model module to improve computational efficiency." This immediately made me think about SOAP and how, particularly over a dataset spanning much of the periodic table in terms of unique elements, the feature vectors can be absolutely enormous. This often makes something like kernel ridge regression (and, therefore, the hyperparameter tuning process) very memory- and compute-intensive. Could a method like GFRE be used to narrow down the range of SOAP-based hyperparameters to search over by plotting a convergence curve like in Figure 2? Is that the primary use case?

Principal Covariates Regression

I thought this section was great, and I don't have any major comments. The demonstration of how PCovR can be very beneficial in KRR (particularly with non-linear kernel functions) was very insightful.

FPS, CUR, and Hulls

Again, I thought this final section was written very clearly. I have one extremely minor comment. On the bottom-left of Page 13, the text states "As all structures that sit above the hull energetically will decompose into a mixture of the more stable phases...". Technically, the convex hull is based

solely on thermodynamics and therefore can only describe thermodynamic (in)stability, if I understand right. Therefore, if the kinetic barrier is high for the transition, it's not necessarily guaranteed that the structure will decompose (i.e. the structure could be meta-stable).

Methods

- Could you please include the version of the DFTB+ code that was used to generate the Mulliken partial charges? Are there any other details that need to be included here for reproducibility purposes, such as chosen Slater-Koster files?

- What code was used to generate the features in the paper? In particular, I know that different codes have implemented SOAP differently, so for the sake of reproducibility, it would be beneficial to know any pertinent details (e.g. for reproducing Figure 2).

Citations

The citations are thorough and appropriate. You may also wish to consider the following:

- You may wish to cite `scikit-learn` since `scikit-matter` is so heavily built around it. See https://scikit-learn.org/stable/about.html#citing-scikit-learn for the citation information.

- You may wish to cite Bartel, _J. Mater. Sci._ (2022) https://link.springer.com/article/10.1007/s10853-022-06915-4 in the convex hull section since most people aren't very familiar with convex hull analyses and the Bartel paper is extremely accessible to a broad audience in case the reader wishes to learn more.

The Code

GitHub Repo

Everything looks great here! I'm very happy to see the significant test coverage. Kudos! The Python package has all the ingredients for a robust and well-maintained code.

Installation

Everything worked as expected!

That said, I do have one very minor suggestion. Although it's in the `pyproject.toml` file, less experienced Pythonistas may not be aware of what versions of Python are supported by `scikit-matter`. So, it might be worth either mentioning that in the docs or adding the associated [PyPI badge](https://shields.io/) to the `README.md`.

Documentation

Overall, I am very pleased by the documentation! My main suggestion has to do with the fact that most users will likely check out the code/docs before ever opening the paper, and right now the documentation isn't sufficient to explain to the user why this code might be for them. For instance, perhaps someone stumbled upon the code on GitHub after someone else star'd it. Without

reading the paper, they are going to see "PCov-FPS extends upon FPS much like PCov-CUR does to CUR" on the "What's in sckit-matter" page and likely be confused as to what this means and may doubt that this is a relevant package for them.

Thankfully, this is an easy fix in my opinion. The paper itself is extremely clear and has the key components for what I would also like to see in the docs: a brief overview of each technique and why it might be useful, along with a minimal working example for each. You can simply adapt snippets of text and/or figures from your paper straight into the docs and it'd likely be perfect. This would best be done in a new "Methods" or "Theory" section, perhaps mimicking the section for "Descriptors" in the [DScribe](https://singroup.github.io/dscribe/latest/tutorials/tutorials.html#descriptors) package.

There are admittedly some details in the API Reference section already, but it's somewhat technical. The API Reference section is also really meant for current users of the code who want to look up the details of a particular function/class. It's not really meant for new users who are trying to figure out the methodology, which is why I think the Python package would be best served by having an additional section in the documentation dedicated to this.

Overall, I would say this is probably the most important comment I have about the submitted work that I hope to see addressed in a future (minor) revision.

Examples

The examples in the documentation are great, and the ability to directly download the code as a Jupyter Notebook is very welcome. As a suggestion for further improvement, I think it would be especially helpful to provide a few sentences of context at the start of each example to orient the user about what the goal of the case-study is. Setting the stage a bit and defining the scope would go a long way, as it wasn't immediately clear to me what the goal was in each example until I completed it. Like above, I also think users who haven't read the paper or references therein likely won't know what PCovR means, for instance, and therefore won't immediately know which examples are relevant to their interests/needs. Providing some background to motivate things at a higher level would help close this gap.

Contributing/References

Everything is very clear here in terms of guidelines! Same for the references. I have no suggestions here.

Miscellaneous

In case the authors are curious, I made a fork of the repo and ran the codebase through several GitHub-based code analyzers, including [Deepsource](https://deepsource.io), [Codacy](https://app.codacy.com/), and [Sourcery](https://sourcery.ai/), to see if there were any glaring bugs or major issues. Good news --- I didn't see anything crucial! Most of the hits were stylistic in nature. This isn't too surprising given the high test coverage. Nice!

References

1. Bartel C: Review of computational approaches to predict the thermodynamic stability of

inorganic solids. *Journal of Materials Science*. 2022; **57** (23): 10475-10498 Publisher Full Text

Is the rationale for developing the new software tool clearly explained?
Yes

Is the description of the software tool technically sound?
Yes

Are sufficient details of the code, methods and analysis (if applicable) provided to allow replication of the software development and its use by others?
Yes

Is sufficient information provided to allow interpretation of the expected output datasets and any results generated using the tool?
Yes

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Computational materials chemistry, quantum-chemical methods, machine learning for materials discovery.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Author Response 25 Aug 2023

Alexander Goscinski

Answer to Andrew S. Rosen

 # Overall Comments

In their submitted work, Goscinski et al. have described and released `scikit-matter`, a Python package built around the `scikit-learn` API that provides a suite of feature selection/analysis tools that are likely to be of particular interest to the computational materials science community. After going through the paper and corresponding code, I am confident that this work warrants acceptance in Open Research Europe. To the best of my knowledge, there is no comparable package published to date, and the code itself fills a niche between pure ML regression codes like `scikit-learn` and ML featurization codes like `QUIP` and `DScribe`.

In short, I recommend this work for acceptance in Open Research Europe with only minor revisions, mainly focused on making the documentation more accessible to a reader who may not (yet) be familiar with the underlying techniques and perhaps has not (yet) read the corresponding paper. I also have a few very minor comments and questions about the paper, as described below.

Congratulations to the authors on this nice work, and I look forward to seeing `scikit-matter` continue to evolve in the future.

We thank Prof. Rosen for his kind words, and are happy that he finds the publication suitable for ORE.

 # The Paper
Overall Comments
Overall, I found the paper to be excellent. I learned a lot from it, and I thought it was exceptionally clear. I found the examples to be very insightful, such as the convergence test for the number of basis functions and the convex hull analysis broken down by the principal components. The paper (and corresponding code) strike a unique and admittedly challenging balance between being domain-agnostic yet also catering to methods that are likely to be especially useful to computational material scientists. I do not have any major comments on the paper, which I think will be a very useful reference for machine learning practitioners and students alike.

Again, we thank Prof. Rosen for his kind words.

Feature Reconstruction Measures This section was very clear and effective! As someone who has not used feature reconstruction measures before, one of the questions I had when reading the text was the following: "for a given dataset of interest, wouldn't doing a standard evaluation of the model cross-validation error for two separate representations also tell me which is more information-rich?" With this in mind, perhaps it might be worth taking a sentence to explain the major benefit of the feature reconstruction method approach. Is it simply that it's a more direct measure of the ability of the representation to reproduce the true data?

It is correct that applying the feature reconstruction error (FRE) on a target property as ground truth is equivalent to a regression. One extension of the FRE is to use an arbitrary set of features as ground truth as it is more suitable to compare features when doing a quantitative assessment of different featurizations of the same representations (e.g. different radial basis functions). We have added the following to the manuscript to clarify this point: The feature set that results in the higher reconstructive measure is considered higher in information capacity (e.g. if FRM($\mathscr{F}$, G) > FRM(G, $\mathscr{F}$), then $\mathscr{F}$ is the more information-rich feature set). The advantage of the FRM is that it can give quantitative results about the shared information content between two sets of features, even without the existence of a ground truth. The simplest FRM is the global feature reconstruction error (GFRE), expressed as the linearly decodable information, as given by performing ridge regression between the two sets of features. We note that in the case of choosing the property as ground truth, the GFRE is equivalent to a regression task on the standardized property.

The authors state that "As most interesting applications use large feature vectors, we implemented a custom RidgeRegression2FoldCV in the skmatter.linear_model module to improve computational efficiency." This immediately made me think about SOAP and how, particularly over a dataset spanning much of the periodic table in terms of

unique elements, the feature vectors can be absolutely enormous. This often makes something like kernel ridge regression (and, therefore, the hyperparameter tuning process) very memory- and compute-intensive. Could a method like GFRE be used to narrow down the range of SOAP-based hyperparameters to search over by plotting a convergence curve like in Figure 2? Is that the primary use case?

Prof. Rosen is correct that this is one beneficial use case for GFRE; optimizing the regularization parameter over a large grid but no other hyperparameters. We have added the following statements to the manuscript: In \sklearn an implementation of the leave-one-out CV with the similar purpose of speeding up the CV exists. A detailed comparison to this method can be found in one of the examples which are included in the documentation of \skmat \url{https://scikit-matter.readthedocs.io/en/v0.2.0/examples/regression/Ridge2FoldCVRegularization.html}. that includes a new sphinx gallery example in the documentation with more detailed analysis.

On the bottom-left of Page 13, the text states "As all structures that sit above the hull energetically will decompose into a mixture of the more stable phases...". Technically, the convex hull is based solely on thermodynamics and therefore can only describe thermodynamic (in)stability, if I understand right. Therefore, if the kinetic barrier is high for the transition, it's not necessarily guaranteed that the structure will decompose (i.e. the structure could be meta-stable).

It is true that kinetics can be an obstacle to spinodal decomposition – convex hulls are typically used to assess thermodynamic processes, as opposed to kinetic ones. We have added the phrase “From a thermodynamic standpoint,” at the beginning of this sentence to clarify the physics here.

Could you please include the version of the DFTB+ code that was used to generate the Mulliken partial charges? Are there any other details that need to be included here for reproducibility purposes, such as chosen Slater-Koster files?

To the best of our knowledge, these charges were computed using DFTB+ version 21.2/22.1 and ase version 3.22.1. We apologize for the uncertainty, as an untimely computer crash has left some ambiguity in the DFTB+ versioning. DFTB+ changes between these two versions would have no impact on the ultimate computations. We used the 3ob Slater-Koster files downloaded from the dftbplus repository and the following code to generate all charges for a given frame from the ice dataset: from ase.units import Hartree, Bohr from ase.calculators.dftb import Dftb calc = Dftb(atoms=frame, label=directory + "/dftb".format(frame.info["name"]), Hamiltonian_MaxAngularMomentum_="", Hamiltonian_MaxAngularMomentum_O="p", Hamiltonian_MaxAngularMomentum_H="s", kpts=(1, 1, 1), Hamiltonian_SCC="Yes", Hamiltonian_MaxSCCIterations=1000, Hamiltonian_Charge=0.0, Hamiltonian_DampXH="Yes", Hamiltonian_DampXHExponent=4.0, Hamiltonian_HubbardDerivs_="", Hamiltonian_HubbardDerivs_O=-0.1575, Hamiltonian_HubbardDerivs_H=-0.1857, Hamiltonian_ThirdOrderFull="Yes",) frame.calc = calc calc.calculate(frame)

frame.arrays["charges"] = frame.get_charges()

This code, as well as all codes (including those written below) used to generate figures in the paper, has been archived at https://github.com/lab-cosmo/skmatter-ore. This link has been added to the “Data Availability” section Data are available under the terms of the \href{https://creativecommons.org/licenses/by/4.0/}{Creative Commons Attribution 4.0 International license} (CC-BY 4.0). The scripts together with the required software packages used to generate the figures are available in the repository https://github.com/lab-cosmo/skmatter-ore in the form of executed jupyter notebooks with output.

What code was used to generate the features in the paper? In particular, I know that different codes have implemented SOAP differently, so for the sake of reproducibility, it would be beneficial to know any pertinent details (e.g. for reproducing Figure 2).

We used librascal (github.com/lab-cosmo/librascal, commit 4e8f9fb) to compute the SOAP features using the following code. We included in the manuscript a link to the repository that contains the notebooks used for the computation of all figures. The specific hypers for each figure can be found in the corresponding notebooks You can find the hypers of Figure 2 in the following notebook https://github.com/lab-cosmo/skmatter-ore/blob/main/paper/atomrings.ipynb

**You may wish to cite `scikit-learn` since `scikit-matter` is so heavily built around it. See https://scikit-learn.org/stable/about.html#citing-scikit-learn for the citation information. Thank you for the advice, a needed citation hidden-in-plain-sight!
You may wish to cite Bartel, J. Mater. Sci. (2022) https://link.springer.com/article/10.1007/s10853-022-06915-4 in the convex hull section since most people aren't very familiar with convex hull analyses and the Bartel paper is extremely accessible to a broad audience in case the reader wishes to learn more.**

Thanks for the suggestion! We have been sure to add in this appropriate citation.

That said, I do have one very minor suggestion. Although it's in the `pyproject.toml` file, less experienced Pythonistas may not be aware of what versions of Python are supported by `scikit-matter`. So, it might be worth either mentioning that in the docs or adding the associated [PyPI badge](https://shields.io/) to the `README.md`.

We updated the metadata of our package and it is now visible on pypi. We also specify what OS and Python version we perform the CI on (the suggestion resulted in the PR https://github.com/scikit-learn-contrib/scikit-matter/pull/200)

The paper itself is extremely clear and has the key components for what I would also like to see in the docs: a brief overview of each technique and why it might be useful, along with a minimal working example for each. You can simply adapt snippets of text and/or figures from your paper straight into the docs and it'd likely be perfect. This would best be done in a new "Methods" or "Theory" section, perhaps mimicking the section for "Descriptors" in the [DScribe](https://singroup.github.io/dscribe/latest/tutorials/tutorials.html#descriptors)

package.

We thank Prof. Rosen for the nice comment and have made appropriate changes to the documentation by adding a “Getting started” section, see https://scikit-matter.readthedocs.io/en/latest/getting-started.html

I think it would be especially helpful to provide a few sentences of context at the start of each example to orient the user about what the goal of the case-study is. Setting the stage a bit and defining the scope would go a long way, as it wasn't immediately clear to me what the goal was in each example until I completed it. Like above, I also think users who haven't read the paper or references therein likely won't know what PCovR means, for instance, and therefore won't immediately know which examples are relevant to their interests/needs. Providing some background to motivate things at a higher level would help close this gap.

Thanks for the comment, we hope with the documentation overhaul we meet the criteria of Prof. Rosen https://scikit-matter.readthedocs.io/en/latest/index.html We put a lot of effort into making it more friendly to less experienced users.

Competing Interests: No competing interests were disclosed.