diffpy.morph: Python tools for model-independent comparisons between sets of one-dimensional functions

Andrew Yang,[1,*] Christopher L. Farrow,[1] Pavol Juhás,[2] Luis Kitsu Iglesias,[3,4] Chia-Hao Liu,[1] Samuel D. Marks,[5,6] Vivian R. K. Wall,[7] Joshua Safin,[8] Sean M. Drewry,[8] Caden Myers,[1] Dillon F. Hanlon,[9] Nicholas Leonard,[9] Cedomir Petrovic,[10,†] Ahhyun Jeong,[11] Dmitri V. Talapin,[11] Linda F. Nazar,[12] Haidong Zhou,[13] Samuel W. Teitelbaum,[9] Tim B. van Driel,[14] Soham Banerjee,[1] Emil S. Bozin,[15] Michael F. Toney,[4,6,16] Katharine Page,[8,17] Naomi S. Ginsberg,[7,18,19,20,21] and Simon J. L. Billinge[1,‡,§]

[1]*Department of Applied Physics and Applied Mathematics, Columbia University, New York, New York 10027, USA*
[2]*Google Research, Google, Mountain View, California 94043, USA*
[3]*Electrochemical Energy Systems Laboratory, Department of Mechanical and Process Engineering, ETH Zurich, 8092 Zurich, Switzerland*
[4]*Department of Chemical and Biological Engineering, University of Colorado Boulder, Boulder, Colorado 80309, USA*
[5]*Advanced Photon Source, Argonne National Laboratory, Lemont, Illinois 60439, USA*
[6]*Materials Science and Engineering Program, University of Colorado Boulder, Boulder, Colorado 80309, USA*
[7]*Department of Chemistry, University of California, Berkeley, California 94720, USA*
[8]*Department of Materials Science and Engineering, The University of Tennessee, Knoxville, Tennessee 37996, USA*
[9]*Department of Physics, Arizona State University, Tempe, Arizona 85287, USA*
[10]*Brookhaven National Laboratory, Upton, New York 11973, USA*
[11]*Department of Chemistry and James Franck Institute, University of Chicago, Chicago, Illinois 60637, USA*
[12]*Department of Chemistry, University of Waterloo, Waterloo, Canada ON N2L 3G1*
[13]*Department of Physics and Astronomy, The University of Tennessee, Knoxville, Tennessee 37996, USA*
[14]*SLAC National Accelerator Laboratory, Stanford University, Menlo Park, California 94025, USA*
[15]*Center for Solid State Physics and New Materials, Institute of Physics Belgrade, 11080 Belgrade, Serbia*
[16]*Renewable and Sustainable Energy Institute, University of Colorado Boulder, Boulder, Colorado 80309, USA*
[17]*Neutron Scattering Division, Oak Ridge National Laboratory, Oak Ridge, Tennessee 37831, USA*
[18]*Department of Physics, University of California, Berkeley, California 94720, USA*
[19]*Molecular Biophysics and Integrated Bioimaging Division and Materials Sciences and Chemical Sciences Divisions, Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA*
[20]*Kavli Energy NanoSciences Institute, University of California, Berkeley, California 94720, USA*
[21]*STROBE, NSF Science & Technology Center, University of California, Berkeley, California 94720, USA*

(Received 1 February 2026; accepted 22 July 2026; published 5 November 2026)

DIFFPY.MORPH addresses a need to gain scientific insights from one-dimensional (1D) scientific spectra in model-independent ways. A powerful approach for this is to take differences between pairs of spectra and look for meaningful changes that might indicate underlying chemical, structural, or other modifications. The challenge is that the difference curve may contain uninteresting differences such as experimental inconsistencies and benign physical changes such as the effects of thermal expansion. DIFFPY.MORPH allows researchers to apply simple transformations, or "morphs," to one of the datasets to remove the unwanted differences, revealing, when they are present, nontrivial differences. DIFFPY.MORPH is an open-source Python package available on the Python Package Index and conda-forge. Here, we describe its functionality and apply it to solve a range of experimental challenges on diffraction and pair distribution function data from x-rays and neutrons, though we note that it may be applied to any 1D function in principle.

DOI: 10.1103/t39r-v3lx

[*]Present address: Department of Applied Physics and Materials Science, California Institute of Technology, Pasadena, California 91125, USA.
[†]Present address: Shanghai Key Laboratory of Material Frontiers Research in Extreme Environments (MFree), Shanghai Advanced Research in Physical Sciences (SHARPS), Shanghai 201203, People's Republic of China.
[‡]Contact author: sbillinge@ucsb.edu
[§]Present address: Materials Department, University of California, Santa Barbara, California 93106, USA.

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

I. INTRODUCTION

Neutron and x-ray powder diffraction (NXPD) and atomic pair distribution function (PDF) analysis are powerful tools for probing the local and average structure of materials [1,2]. They can provide many insights; for example, one can compare NXPD patterns and PDFs of samples measured at different temperatures to search for structural changes [3]. This may be done in a model-independent way by comparing two patterns and looking for meaningful changes, for example, by plotting the difference curve between the patterns or computing similarity metrics such as the weighted agreement factor (R_w [1,2]) or the Pearson correlation coefficient (PCC [4], both defined in Sec. A of the Supplemental Material [5]). This task is made more difficult in the presence of less interesting effects such as expansion/contraction due to temperature changes, thermal smearing due to increased thermal motion, and certain experimental artifacts. These are effects that may not represent "interesting" structural changes, but can produce large contributions to the residual curves and R_w's, hiding more scientifically interesting structural changes. A challenge is to remove the uninteresting changes to reveal scientifically interesting ones when they are present.

Here we present the Python program DIFFPY.MORPH that provides a solution to this problem by first applying simple transformations, or "morphs," to spectra to partially or fully account for these noninteresting effects, allowing any interesting changes that cannot be removed by the morph to become apparent. Data transformations are not new, and a variety of methods are used to match peaks in chromatograms and spectral data (e.g., correlation optimized warping, dynamic time warping, parametric time warping, etc.) [12–14]. Many diffraction processing software packages also account for instrument broadening effects [15–17]. DIFFPY.MORPH provides additional physics-inspired morphs to capture physical and other instrumental or environmental effects. Unlike the warping methods, peaks do not need to correspond between the two functions compared, meaning large residuals postmorph can be used to discern scientifically interesting differences. This approach is fast, model-independent, and a powerful tool for interrogating data in real time as they are being collected. We note that this approach is completely general and may be applied beyond diffraction and atomic pair distribution function [2] data to any kinds of one-dimensional (1D) spectra.

Here we describe the approach and test it on a number of different real-world examples from our own studies. We show how it can be used to reveal when a structural phase transition occurs, and when it does not, in temperature series PDF data. We then present an example in which a thermodynamic parameter β, the (linear) thermal expansion coefficient, is extracted directly from temperature-dependent data without modeling, and in contrast, where the temperature of a sample can be extracted given a known β, again in a model-independent way. We present an example in which experimental inconsistencies could be removed between datasets allowing them to be compared despite the inconsistencies. Finally, we also present some more advanced use cases, where we use DIFFPY.MORPH to find optimal parameters in a data reduction workflow by morphing low $Q_{\max}$ high-throughput data to a reference high $Q_{\max}$ target dataset.

DIFFPY.MORPH is easy to use and flexible. It is open source (BSD license) [18] and available on PyPi [19] and conda-forge [20]. It has a command-line interface and API allowing it to be integrated into users' Python scripts. The program can read most column-style data files (e.g., '.xy,' '.dat,' '.chi,' '.gr') with and without header information. It also automatically handles comparison of two functions plotted on different grids by using linear interpolation. Additionally, DIFFPY.MORPH can estimate the one-standard-deviation errors of all fitted parameters.

Though this software was originally designed in the context of PDFs, DIFFPY.MORPH and its morphs can be easily applied to any general function. For example, a neutron scattering experimentalist could compare two densities of states (DOS) measured at different temperatures by morphing one onto the other, or a computer vision scientist comparing two pictures taken at different brightnesses can give the (flattened) 2D images to DIFFPY.MORPH to correct for intensity differences. One should be careful when using morphs to avoid overfitting and ensure all their morphs are physically sensible. For example, Gaussian smearing morphs should not be applied to functions with Lorentzian peaks, and quantities extracted from morphs that assume isotropic effects should be interpreted with caution for highly anisotropic materials.

II. MOTIVATING EXAMPLES

Here, we introduce some examples that motivated our development of DIFFPY.MORPH but which also illustrate how it may be applied in different situations.

A. Identifying a structural phase transition

1. Identifying whether or not a phase transition has occurred by using morphs

Figures 1(a)–1(c) show pairs of PDFs that were collected at 10 K (blue) and 300 K (red) with the difference curve plotted below.We would like to answer the following questions: (i) "Do any of these show a structural phase transition in this range?" (ii) "If so, which?" We might be able to do this by comparing the two PDFs and considering various difference measures. A large difference measure should indicate that there may be a phase transition, and a smaller measure should indicate that there is not. We see the largest difference measures for (b) ($R_w = 0.6463$) and then (a) ($R_w = 0.5661$), with the smallest for (c) ($R_w = 0.4764$), which might suggest that it is more likely to find a structural phase transition in (a) and (b). However, this is a wide temperature range, and there are significant thermal effects. Figures 1(d)–1(f) show the corresponding pairs of PDFs after applying "stretching," "scaling," and "smearing" morphs, described further below, to the lower-temperature data. In this case, the difference measures become very small for (d) and (e) but remain somewhat large with a significant unresolved signal in the green difference curve for (f).

The sample in (c),(f) is an $IrTe_2$ compound, and in (a),(d) and (b),(e), the iridium is substituted with rhodium and platinum, respectively. Single crystals of $IrTe_2$ were grown with a self-flux method [21], while rhodium-substituted [22,23] and

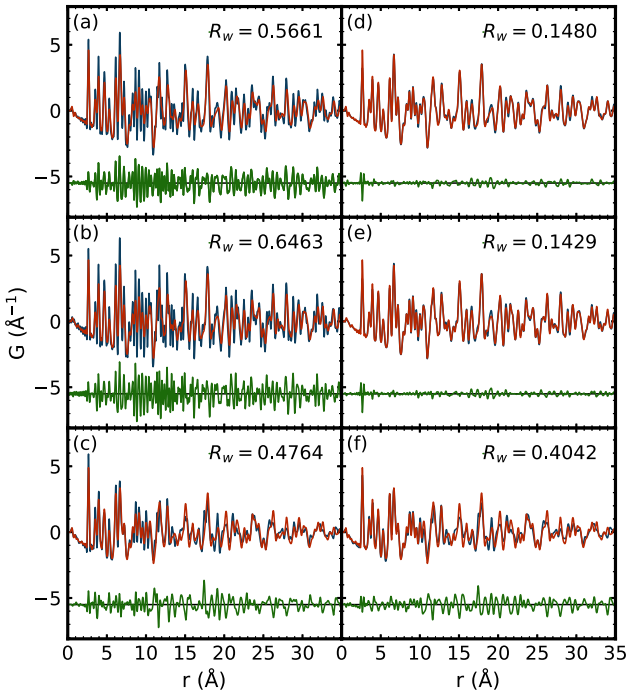


FIG. 1. PDFs of $IrTe_2$ samples with various forms of substitution measured at 10 K (blue) and 300 K (red). The difference curves (green) are plotted below. All R_w values are computed on the grid $r \in [0\ \text{Å}, 35\ \text{Å}]$ with spacing $dr = 0.01$. (a) Samples are rhodium-substituted $Ir_{0.8}Rh_{0.2}Te_2$. (b) Platinum-substituted $Ir_{0.95}Pt_{0.05}Te_2$. (c) Pure $IrTe_2$. Plots (d), (e), and (f) have the same high-temperature PDF as (a), (b), and (c), respectively, but a "stretching," "scaling," and "smearing" morph has been applied to the low-temperature PDF to best fit the high-temperature target PDF.

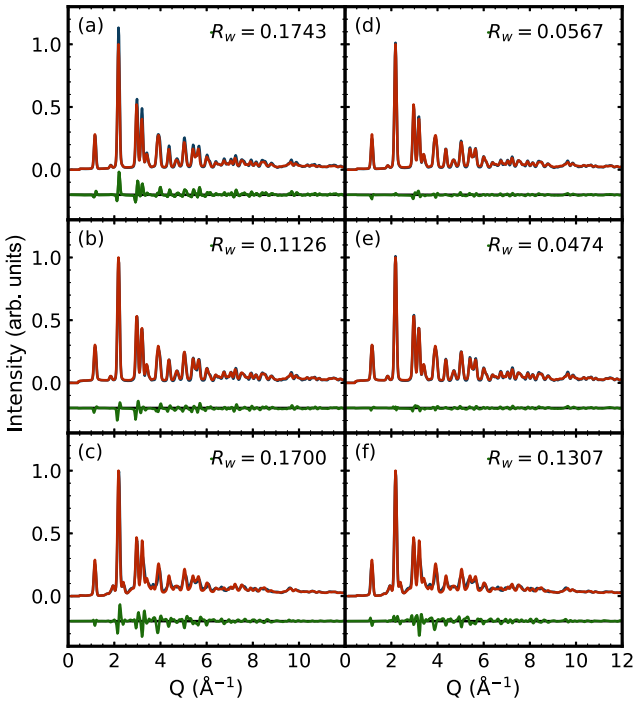


FIG. 2. $I(Q)$'s of $IrTe_2$ samples with various forms of substitution measured at 10 K (blue) and 300 K (red). The difference curves (green) are plotted below. All R_w values are computed on the grid $Q \in [0\ \text{Å}^{-1}, 12\ \text{Å}^{-1}]$ with spacing $dQ = 0.02$. (a) Rhodium-substituted $Ir_{0.8}Rh_{0.2}Te_2$. (b) Platinum-substituted $Ir_{0.95}Pt_{0.05}Te_2$. (c) Pure $IrTe_2$. The $I(Q)$'s plotted in (a)–(c) were used to generate the PDFs in Figs. 1(a)–1(c). Plots (d), (e), and (f) have the same high-temperature $I(Q)$ as (a), (b), and (c), respectively, but a `stretch` and `scale` morph has been applied to the low-temperature $I(Q)$ to best fit the high-temperature target $I(Q)$.

platinum-substituted [24] polycrystals were synthesized using a solid-state reaction method.

After morphing, it is clear that pure $IrTe_2$ has a phase transition in this temperature range, but substituting Pt, even at the five percent level, suppresses the phase transition. This was not apparent in the unmorphed data, but is confirmed by structural modeling in the original study [25].

Similar problems arise when comparing 1D diffraction intensities, $I(Q)$, prior to computing a PDF. Figures 2(a)–2(c) show the same analysis as carried out in Fig. 1 but this time on the $I(Q)$ data instead of the PDF.

As before, prior to morphing, it is much less clear from the difference curve which of these cases have structural phase transitions and which do not, but this is very apparent after morphing.

We now investigate in more detail how these morphing transformations actually affected the data. In Fig. 3(a) we compare PDFs from $Ir_{0.8}Rh_{0.2}Te_2$ at 250 K (blue) and 300 K (red).The agreement factor $R_w = 0.1042$, and there are some ripples in the green difference curve plotted below. By close inspection, we can observe that the peaks in the warmer sample are broader and shifted to higher-r. This is most evident in the region $30 \leqslant r \leqslant 35$ Å, plotted on an expanded scale in Fig. 3(d). More evidence of peak shifting across the entire r-grid is provided in the Supplemental Material [5], Fig. S1.

The peak shifts could come from thermal expansion and the peak broadening from increased thermal motion. DIFFPY.MORPH allows us to investigate whether a uniform "stretch" of the lattice of the lower-temperature sample can account for the peak shifts across the whole range of the PDF without having to know and model the structure of the material.

To accomplish this, DIFFPY.MORPH first interpolates the lower-temperature PDF onto a slightly stretched r-grid $r \mapsto r/(1+\eta)$. We call this the `stretch` morph with a stretch parameter η. Then, DIFFPY.MORPH multiplies the PDF by a constant scale factor s. This is the "`scale`" morph with scale factor s. The result of applying these two transformations is shown in Fig. 3(b). The R_w is significantly reduced, as are ripples in the difference curve.

From a practical perspective, it is tedious to find the exact `stretch` and `scale` morphs to best explain the difference, so we wrap the transformation in a least-squares regression method to find the η and s values that minimize the R_w between the PDF of the low-temperature signal after morphing [which we denote the "morphed" signal $G_m(r)$] and the untouched high-temperature PDF [the "target" signal $G_t(r)$].

Although R_w is reduced by almost a factor of 2 by the `stretch` and `scale` morphs, differences remain. We would like to see the extent to which they can be removed by a

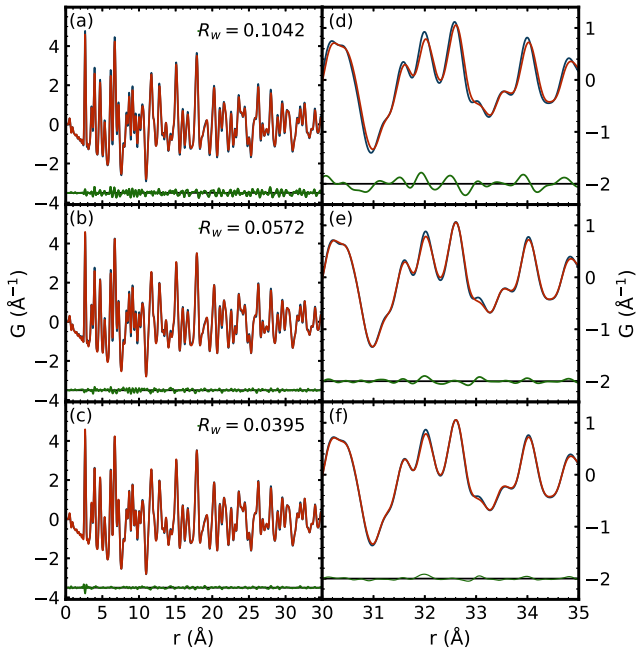


FIG. 3. Morphed PDFs of $Ir_{0.8}Rh_{0.2}Te_2$ samples measured at 250 K (blue) plotted against unmorphed PDFs measured at 300 K (red). The difference curves (green) are plotted below. All R_w values are computed on the grid $r \in [0\ \text{Å}, 35\ \text{Å}]$ with spacing $dr = 0.01$. (a) No morph is applied. (b) A `stretch` and `scale` morph is applied to the lower-temperature PDF. (c) A `stretch`, `scale`, and `smear` morph is applied to the lower-temperature PDF. Plots (d), (e), and (f) are the same as (a), (b), and (c), respectively, zoomed into the region $30 \leqslant r \leqslant 35$ Å.

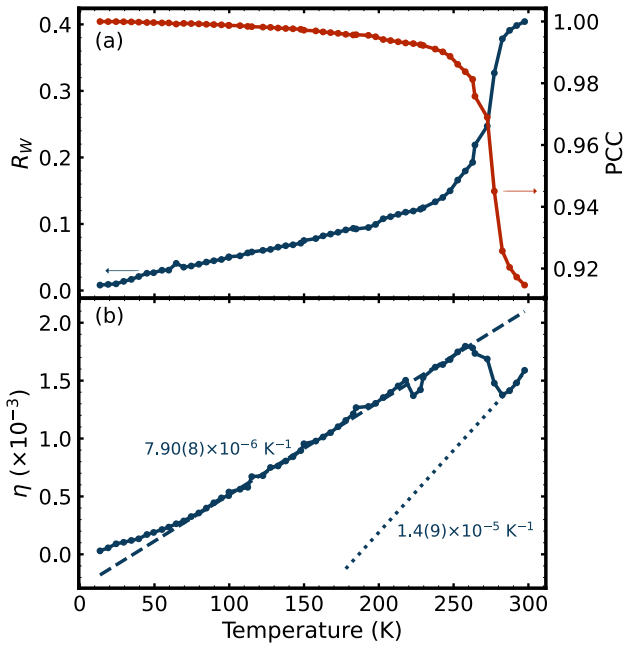


FIG. 4. (a) R_w (blue) and Pearson correlation coefficient (red) computed between a morphed and target PDF from $IrTe_2$ PDFs measured at temperatures ranging from 9 to 297 K. (b) The `stretch` parameter η found in each morph indicating thermal expansion. This parameter sharply drops above 263 K. The drop indicates contraction at the phase transition. Dashed lines are linear fits to the low- and high-temperature regions.

uniform broadening, or smearing, of the radial distribution function (RDF) of the lower-temperature sample. We accomplish this by computing the RDF from the morphed PDF, convolving the RDF with a normalized Gaussian function with variance σ^2, and we convert the smeared RDF back to a PDF. We then vary the σ^2 of that convolution function until we get the best agreement between the morphed PDF and the target PDF. This is the "`smear`" morph with smearing parameter σ. The result of turning on the `smear` morph transformation on top of the `stretch` and `scale` morphs is shown in Fig. 3(c). This results in a further reduction in R_w of 31% and an almost flat difference curve. For more details on how the morphs are implemented, see Sec. III.

In the absence of a structural phase transition, the one place in the difference curve where significant fluctuations remain after morphing is in the vicinity of the first peak at $r = 2.7$ Å. This is evident in the difference curve in Figs. 1(d) and 1(e) and Fig. 3(c). This phenomenon is well-known and is due to changes in the correlated atomic motion [26]. This could be removed with a smear-morph applied over just the range of the PDF incorporating this first peak. It would allow the temperature dependence of the broadening of the first peak to be extracted separately from the higher-r peaks and extract information about the degree of motional correlations, though we do not do that here. It also shows how careful morphing can remove some features and not others, revealing key insights.

2. *Finding the temperature of a phase transition by analyzing multiple morphs*

We have seen how DIFFPY.MORPH is helpful in identifying whether or not a phase transition has occurred, but it can also help to identify the temperature range in which the transition occurs.

On a dataset of $IrTe_2$ PDFs measured from 10 to 300 K, we have used DIFFPY.MORPH's capability to create multiple morphs (`stretch`, `scale`, `smear`) of the lowest-temperature PDF with each higher-temperature PDF in the set as a target. In Fig. 4(a), we plot the resulting (post-morph) R_w's.The R_w's vary slowly over the entire temperature range until 263 K, at which point there is a rapid change in slope until 283 K, where it again slows down. The same can be seen in the PCC curve shown in red. A phase transition (from low-temperature triclinic to high-temperature trigonal) has been observed near 280 K [27]. The DIFFPY.MORPH analysis shows the onset of this phase transition on cooling beginning at around 283 K with the distortion growing smoothly on cooling until around 263 K but using a completely model-independent analysis and a simple command in a terminal.

3. *Comparison of DIFFPY.MORPH with model refinement techniques*

This method of identifying phase transition temperatures using DIFFPY.MORPH produces results that agree with structural model fitting techniques such as PDFGUI [28]. There is a known structural change in $SrFe_2As_2$ between 192 and 198 K [3]. PDFs of $SrFe_2As_2$ measured from 150 to 246 K were

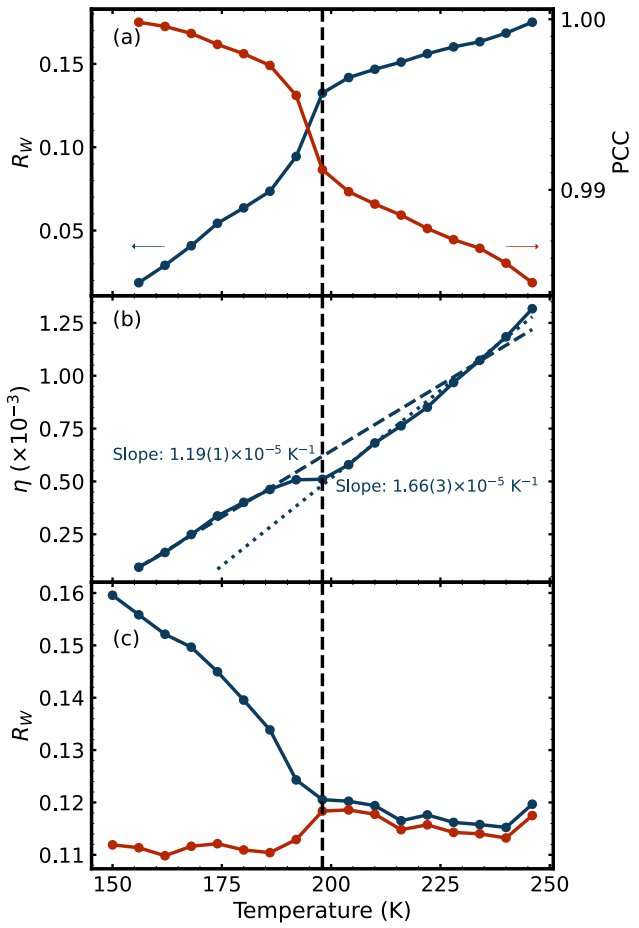


FIG. 5. (a) R_w (blue) and PCC (red) after morphing the PDF from $SrFe_2As_2$ measured at $T = 150$ K onto the PDF measured at target at temperatures up to 246 K. (b) The `stretch` parameter η obtained from each morph. The dashed and dotted lines show best-fit linear curves in the temperature ranges 156–192 K and 198–246 K, respectively. The slope gives the thermal expansion coefficient in the respective temperature ranges. (c) The R_w's obtained by fitting a tetragonal (blue) and orthorhombic (red) structure to each PDF on the r-range [0.5 Å, 100 Å]. The black vertical dashed line is a guide to the eye at the first measured point above the structural phase transition.

sourced from data published alongside the textbook "Atomic Pair Distribution Function Analysis: A Primer" [3,29]. In Fig. 5(a), we have plotted the R_w and PCC values resulting from morphing the PDF of $SrFe_2As_2$ measured at 150 K to PDFs of the same sample measured at the various higher temperatures.As the temperature increases, we observe a steep ascent in the R_w and descent in the PCC within the temperature range of the phase transition. We have also used DIFFPY.MORPH to plot the `stretch` parameter used in the morphs across temperature in Fig. 5(b). While this parameter increases linearly from 156 to 192 K and from 198 to 245 K, indicating some isotropic expansion in those ranges, the `stretch` parameter does not change from 192 to 198 K, further confirming that some more interesting structural change is occuring at those temperatures. Structural modeling refinement done by PDFGUI [28] shows that the tetragonal model only fits well past 198 K, indicating some structural phase change occurs past this temperature. This result was obtained in a model-independent DIFFPY.MORPH analysis.

B. Extracting material properties using morph comparisons

1. Thermal expansion and Debye temperature using stretch morph on temperature dependent x-ray data

Using a series of PDFs measured at different temperatures, one can estimate the linear thermal expansion coefficient of a material using the `stretch` morph. For nearly harmonic materials, one can further compute the Debye temperature by looking at the derivative of the stretch parameter with respect to temperature.

In Fig. 4(b), the value of the `stretch` morph parameter $\eta(T)$ computed on the grid $r \in [0\text{ Å}, 35\text{ Å}]$ is plotted against temperature for the $IrTe_2$ sample whose x-ray PDF was measured versus temperature. We see this grow smoothly, albeit noisily. By finding the gradient of $\eta(T)$, we can extract the thermal expansion coefficient as we discuss below.

The `stretch` parameter obtained from morphing between a temperature T_a and T_b is an average isotropic strain $\epsilon(T_a, T_b)$ of the material induced by the change in temperature (shown in the Supplemental Material [5], Sec. C). We can compute a linear thermal expansion coefficient $\beta(T_a)$ using

$$\beta(T_a) = \frac{\eta(T_b) - \eta(T_a)}{(T_b - T_a)[1 + \eta(T_a)]} \tag{1}$$

$$\approx \frac{\eta(T_b) - \eta(T_a)}{T_b - T_a}, \tag{2}$$

where the approximation holds for $\eta(T_a) \ll 1$. Each $\beta(T_a)$ was computed from discrete temperature data using finite differences of $\eta(T)$. From Fig. 4(b), we compute an average thermal expansion coefficient of $7.90 \pm 0.08 \times 10^{-6}$ K^{-1} for the triclinic phase of $IrTe_2$ (in the range $T \in [107\text{ K}, 263\text{ K}]$) and $1.4 \pm 0.9 \times 10^{-5}$ K^{-1} for the trigonal phase (in the range $T \in [283\text{ K}, 297\text{ K}]$). The uncertainties in expansion coefficients are the uncertainties in the linear fit propagated from one-standard-deviation errors estimated by DIFFPY.MORPH for each of the stretch parameters.

Note that the `stretch` yields an average linear expansion coefficient that more accurately reflects the actual expansion the more isotropic the material is, and if the material expansion is highly anisotropic (e.g., near a structural phase transition), the stretch morph will not remove the effects of the thermal expansion. An average linear thermal expansion coefficient can also be extracted for a specific subset of peaks (or a single peak) by applying the `stretch` morph to best fit a user-specified region in the PDF.

We can also get a sense of the anharmonicity of the material by examining the thermal expansion coefficient as a function of temperature. This $\beta(T)$ can be modeled by the temperature derivative of the stretch parameter $\eta(T)$,

$$\beta(T) \approx \frac{\partial \eta(T)}{\partial T}, \tag{3}$$

per Eq. (2). In Fig. 6, we plot the thermal expansion coefficient, extracted from the morphed data [using central differences to compute the derivative of $\eta(T)$], over a large temperature range for (a) Pt-substituted and (b) pure $IrTe_2$. We

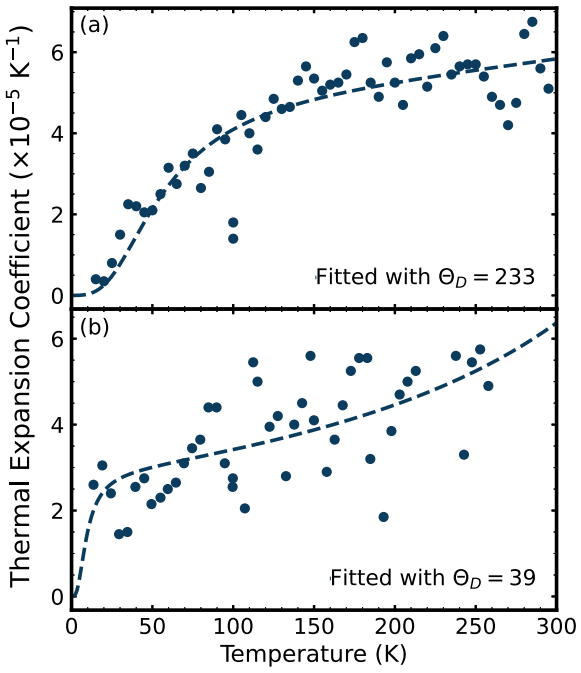


FIG. 6. Plots of thermal expansion coefficients [extracted using the approximation in Eq. (3)] at various temperatures. The samples involved are (a) $Ir_{0.95}Pt_{0.05}Te_2$ and (b) $IrTe_2$. Data are fitted to Eq. (4) with parameters A, B, and Θ_D (see the text). The fits have R^2 values of (a) 0.7937 and (b) 0.4143.

then fit the data to the curve for the (linear) thermal expansion $\beta_{G,A}$, using a quasiharmonic model given by

$$\beta_{G,A}(T;A,B,\Theta_D) \approx A\frac{\left(\frac{T}{\Theta_D}\right)^3 \int_0^{\Theta_D/T} \frac{x^4 e^x}{(e^x-1)^2}dx}{1-BTe^{-\Theta_D/(2T)}}, \qquad (4)$$

with fitting parameters A, B, and θ_D (Debye temperature). The expression is obtained from the Grüneisen equation of state [30,31], where the forms of the isochoric heat capacity (numerator) and adiabatic bulk modulus (denominator) come from the Debye model [32] and the Anderson approximation [31,33], respectively. For this model to work, we must assume a constant heat-capacity ratio C_V/C_P [31,32] to ensure that the isothermal and adiabatic bulk moduli are proportional, assume the change in volume is small, and assume a constant Grüneisen parameter [30,31,33]. Anharmonic (nonharmonic and non-quasi-harmonic) materials are likely to violate these conditions, while the platinum-substituted sample has a nice agreement and a sensible Debye temperature of 233 ± 29 K [22,24]. (The uncertainty reported is the estimated one-standard-deviation error of the fitting parameter Θ_D.) The opposite is true for the pure sample with an anomalous upturn in β at high-T [Fig. 6(b)] and a nonsensically low Debye temperature of 39 ± 24 K. These physical abnormalities and low R^2 value of 0.4143 suggest that the thermal expansion of the pure $Irte_2$ sample is not captured by the quasiharmonic model, and the extracted Debye temperature is not to be trusted. The anharmonicity of pure $IrTe_2$ has been explored previously using Raman spectrometry [21].

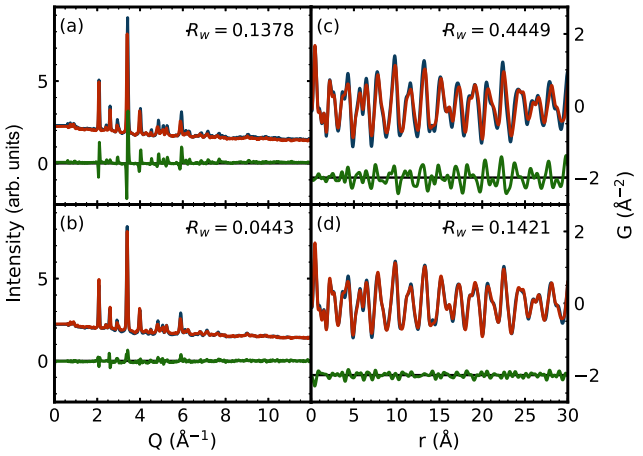


FIG. 7. $I(Q)$'s (a) and (b) and PDFs (c) and (d) of YSZ samples heated with a 30 W (plotted in blue) and 50 W (plotted in red) laser. Difference curves are plotted offset below in green. No morphs are applied in (a) and (c). `stretch` and `scale` morphs are applied to the lower-power (blue) signal in (a) to obtain the blue $I(Q)$ in (b). `stretch`, `scale`, and `smear` morphs are applied to the lower-power signal in (c) to obtain the blue PDF in (d). The R_w is computed from the entire Q, or r, range plotted.

2. *Using DIFFPY.MORPH for in situ sample thermometry*

The converse of finding the thermal expansion coefficient from temperature series data is using a known thermal expansion coefficient to estimate sample temperature in a model-independent way.

There are few available techniques for precisely measuring the temperature of diffraction experiment samples heated to high temperatures. One such technique is pyrometry, but thermal gradients caused by uneven heating of the material and ablation in the scattering region can greatly limit the accuracy of the measurement [34,35]. Further, pyrometry measures only the surface temperature, which makes it unreliable for transmission geometry scattering experiments.

This motivates the development and use of alternative methods for obtaining the bulk average temperature of a sample under *in situ* conditions. In Sec. II A 2, we demonstrated how DIFFPY.MORPH can be used to identify whether a phase transition has occurred between two PDFs. This can be matched with a known phase transition temperature to identify the temperatures of these PDFs [this works also for $I(Q)$'s]. On the other hand, if a sample does not undergo a phase transition, the following example shows how one can use the known thermal expansion coefficient of the sample material to obtain the average temperatures of the sample at the moment the $I(Q)$'s and PDFs were acquired [36].

A commercially obtained sample of yttria-stabilized zirconia (YSZ) was laser-heated from room temperature with laser powers ranging from 0 to 50 W. At each power level, a neutron powder diffraction $I(Q)$ was measured, and the corresponding PDF, $G(r)$, computed. In Fig. 7(a), we show the $I(Q)$ functions computed from laser-heated YSZ samples at powers 30 and 50 W. The difference curve shows a significant signal, due to thermal expansion from the additional 20 W of power. Similarly, the difference curve between the

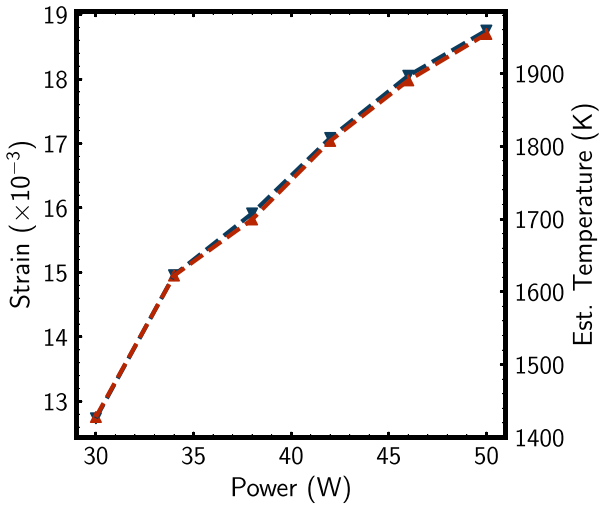


FIG. 8. Strains computed by morphing 0 W YSZ PDF data (blue) and $I(Q)$ data (red) to higher-power YSZ data. The corresponding temperature is given by the right axis.

corresponding PDFs [shown in Fig. 7(c)] also shows a significant signal. However, in (b) and (d), after applying a `stretch` and `scale` (as well as a `smear` morph for the PDF data) on the 30 W data with the 50 W data as a target, the majority of the differences are removed.

The thermal expansion coefficient of the YSZ was calculated to be $\beta = 11.29 \times 10^{-6}$ 1/K from optical dilatometry measurements. This thermal expansion coefficient is in agreement with existing literature [34,37,38]. With this thermal expansion, we can use the `stretch` morph to estimate the temperature of the laser-heated YSZ samples directly.

For small $\Delta T = T_b - T_a$, we assume $\eta(T_a) \approx \eta(T_b)$. Rearranging Eq. (2), we get

$$\Delta T \approx \frac{\eta_{gr}}{\beta}, \tag{5}$$

where η_{gr} is the PDF `stretch` parameter. Using the `stretch` parameter and β, we can obtain ΔT between the room-temperature (298 K) sample and each laser-powered sample. The actual temperature is then

$$T = 298 \text{ K} + \Delta T. \tag{6}$$

The strain, equivalent to η_{gr} (see the Supplemental Material [5], Sec. C), and the corresponding estimated temperatures (right axis label) are plotted in blue in Fig. 8.

We also carried out the same analysis directly on the $I(Q)$ data rather than the PDF. [A similar approach has been used to determine sample temperatures on x-ray $I(Q)$ data [36].] In this case, the `stretch` parameters, η_{iq}, correspond to a strain ϵ of

$$\epsilon = (1 + \eta_{iq})^{-1} - 1 \tag{7}$$

(see the Supplemental Material [5], Sec. D) and thus a temperature change of

$$\Delta T \approx \frac{(1 + \eta_{iq})^{-1} - 1}{\beta}. \tag{8}$$

The strains and corresponding temperatures extracted from the $I(Q)$ morphs correspond to the red curve in Fig. 8. These are in excellent agreement with each other. The largest discrepancy between the two curves is a difference of <0.3% at 38 W.

This example not only shows how temperatures can be estimated using DIFFPY.MORPH, but its applicability to not just x-ray diffraction signals but also neutron data.

3. Nanoparticle shape information from shape morphs

The shape and size of a nanoparticle can greatly affect its (electronic, optical, etc.) properties [39,40]. If one has PDF measurements of a bulk and a nanoparticle of the same material, it is possible to use a nanoparticle "`shape`" morph to determine the morphological parameters without a structural model. By way of example, we show how it is possible to determine the radius of a spherical nanoparticle using a `shape` morph.

This morph can be very useful if you have a PDF calculated from a bulk model but your data are from a nanocrystalline sample. DIFFPY.MORPH will then multiply the bulk PDF by the chosen nanoparticle characteristic function [2,41], $\gamma_0(r)$, automatically adjusting the parameters of the model to find the best agreement to the nanoparticle PDF.

If a good fit is obtained, the model choice and refined shape parameters can be considered useful, though other models should be tested to search for alternative possible candidates [41–44]. Significant difference curve signals or R_w's indicate large deviations from the desired shape, and new shape models may be tried.

This approach has previously been used to estimate diameters of spherical CdSe nanoparticles consistent with those obtained from transmission electron microscopy, ultraviolet-visible spectroscopy, and photoluminescence measurements [45].

Here, we have used this approach to estimate the size (radius) of PbS nanocrystals with thiostannate ($Sn_2S_6^{4-}$) ligands. These nanocrystals are known to be faceted [46], but for small diameters, they are approximately round. (Despite the mismatch in shape, the spherical `shape` morph can still be applied, and the radius extracted from the spherical `shape` morph will approximate the radius of the sphere with the same volume as our faceted shape. This is discussed in greater depth in the Supplemental Material [5], Sec. F2.)

In Fig. 9(a), we show the PDF measured from the PbS nanocrystal (blue) against a PDF computed from the bulk structure of PbS [47] (red). We see a clear reduction in the signal of the measured data (blue curve) as we go to higher r, indicating that our measured material is definitely nanocrystalline. We then apply the nanoparticle `shape` morph onto the computed bulk PDF. The resulting morphed PDF is shown in red in Fig. 9(b). The radius obtained from the morph was 24.5 ± 0.1 Å. For comparison, we used PDFGUI [28] to compute a spherical PbS nanocrystal PDF from a structure [47] to best fit our measured PDF. The resulting refinement yielded a nanocrystal diameter of 47.7 ± 1.2 Å (radius of 23.9 ± 0.6 Å) [red curve in Fig. 9(c)]. The refinement done by DIFFPY.MORPH (order of seconds) is much faster than the lengthy and human intensive structural refinement done by

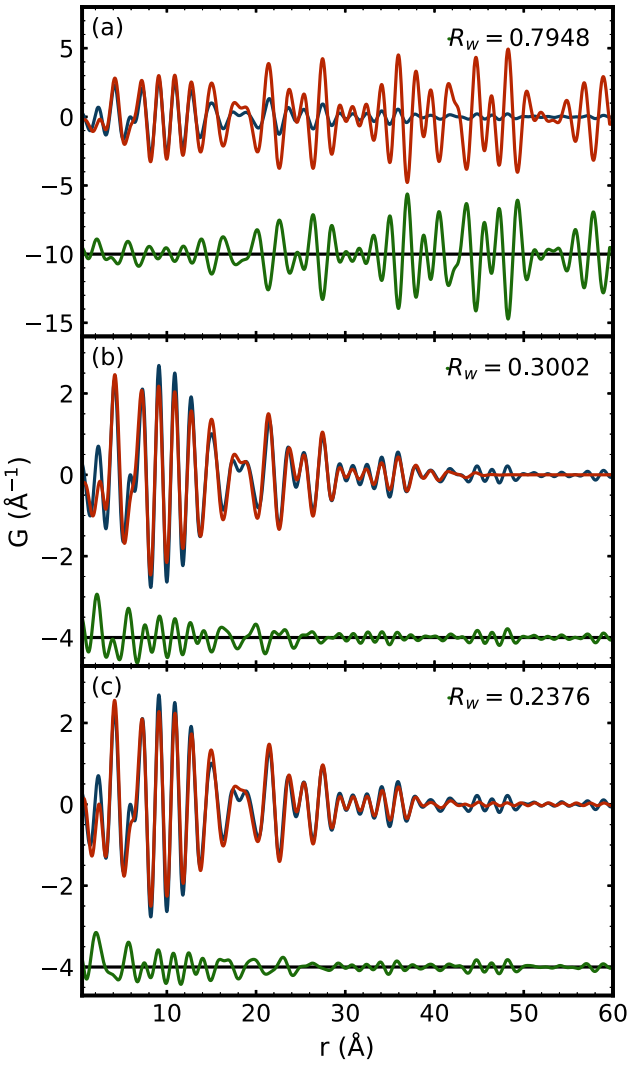


FIG. 9. (a) Measured PbS nanocrystal PDF (blue) plotted against a computed PDF of the bulk PbS (red). (b) The red curve is now the bulk PbS PDF after applying the spherical nanoparticle `shape` morph. (c) The red curve is a PDF of a spherical PbS nanocrystal computed from the structural model using PDFGUI. A difference curve (green) is vertically offset below each plot.

PDFGUI, with the two calculated radii agreeing within their uncertainty estimates.

The DIFFPY.MORPH-computed and PDFGUI-computed PDFs [red curves in (b) and (c), respectively] also show good agreement up to $r = 49$ Å. Above this, the DIFFPY.MORPH-computed PDF flat-lines, while the PDFGUI-computed PDF has small sinusoidal oscillations that account for termination ripples [2]. This example showcases how DIFFPY.MORPH can quickly compute nanoparticle and nanocrystal PDFs from bulk PDFs and obtain morphological parameters.

At the time of writing, DIFFPY.MORPH only includes morphing transformations to account for the effects of a spherical and spheroidal nanoparticle shape on its measured PDF. These should only be used for particles that are approximately of this shape. If the desired `shape` morph is not implemented, please consider requesting it with an issue at the DIFFPY.MORPH GitHub website [48] or by posting a request at the diffpy-users Google group [49]. If you are able to implement the morph yourself, please consider posting a pull request at the DIFFPY.MORPH repository on GitHub.

C. High-throughput experiment use cases

DIFFPY.MORPH has been utilized in high throughput experimental settings, such as synchrotron beam-times, for a variety of use cases. These include exposing sources of experimental error (e.g., incorrect sample-to-detector distance or misaligned beam center) to identify and remove unwanted signals such as laser-induced heating, and to identify optimal processing parameters to compute reliable PDFs from measured $I(Q)$ data.

A major advantage of using DIFFPY.MORPH in a high-throughput data setting is its speed and ease of use. Rather than performing a lengthy structural refinement, one can obtain morphing parameters in seconds.

1. Addressing experimental setup challenges

When integrating a 2D diffraction image to obtain a 1D scattering function $I(Q)$, we require various experimental geometric parameters such as the sample-to-detector distance, location on the detector of the beam-center, tilt angles of the detector away from orthogonality, and wavelength of the incident beam. These are normally handled by carefully performing calibration measurements. In practice, small deviations between calibration and sample datasets can arise due to a multitude of factors such as minor sample or detector repositioning, motor drift, thermal expansion or contraction of experimental components, and other instrumental or environmental effects. In such cases, DIFFPY.MORPH can be used to apply small corrective adjustments to the calibration parameters.

We prepared a sample of PbS nanocrystals with thiostannate ($Sn_2S_6^{4-}$) ligands dissolved in an amide solution. The synthesis of the nanocrystals and ligands follows previous published procedures [50]. A 2D diffraction pattern was obtained from this sample at an XFEL beamline (measured to $Q_{\max} = 4.05$ Å^{-1}). In Fig. 10(a), we showcase two $I(Q)$'s computed from this 2D diffraction pattern, but where the integration was carried out using different values for the sample-to-detector distances and beam centers.The numbers used were rough initial guesses from the geometry as an initial calibration was not available. This is compared to the final calibrated data [red curve in Fig. 10(a)]. Specifically, the uncalibrated sample-detector distance value used was $D_{\text{unc}} = 64.7$ mm with the calibrated distance found to be $D_{\text{cal}} = 55.7$ mm. Additionally, the beam center used for the uncalibrated curve is offset -2.26 mm both horizontally and vertically from that found from the calibrated curve. We can see a significant mismatch between the two curves, but by using the `stretch` and `scale` morphs along with a "`shift`" morph, in (b), we can obtain very good agreement Fig. 10(b). Here, the `shift` morph shifts the function to the right by adding a shift constant Δ_h to the x-array to account for the zero-offset. This helps to account for the misaligned detector center.

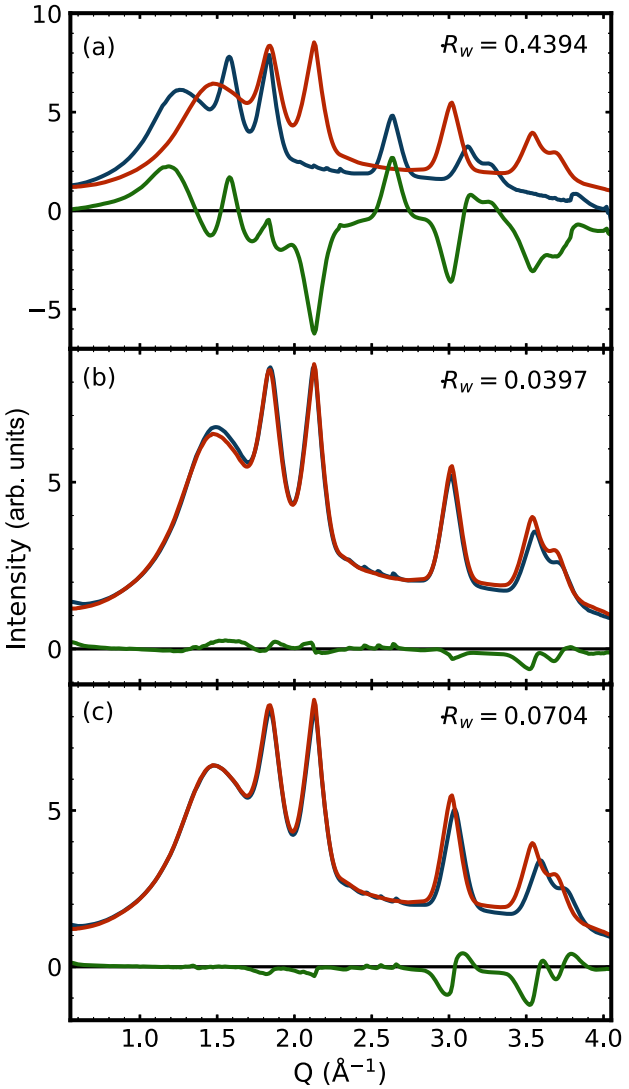


FIG. 10. $I(Q)$'s computed from the same diffraction image using the uncalibrated (blue) and calibrated (red) parameters are plotted in (a). In (b) `stretch`, `scale`, and `shift` morphs are applied to the uncalibrated data to maximize agreement with the calibrated data on $Q \in [0.55\ \text{Å},\ 4.05\ \text{Å}]$. For (c), the same morphs are used, but instead maximizing agreement on the Q region encompassing the first three peaks ($Q \in [0.55\ \text{Å},\ 2.35\ \text{Å}]$). We include a difference curve (green) below each plot.

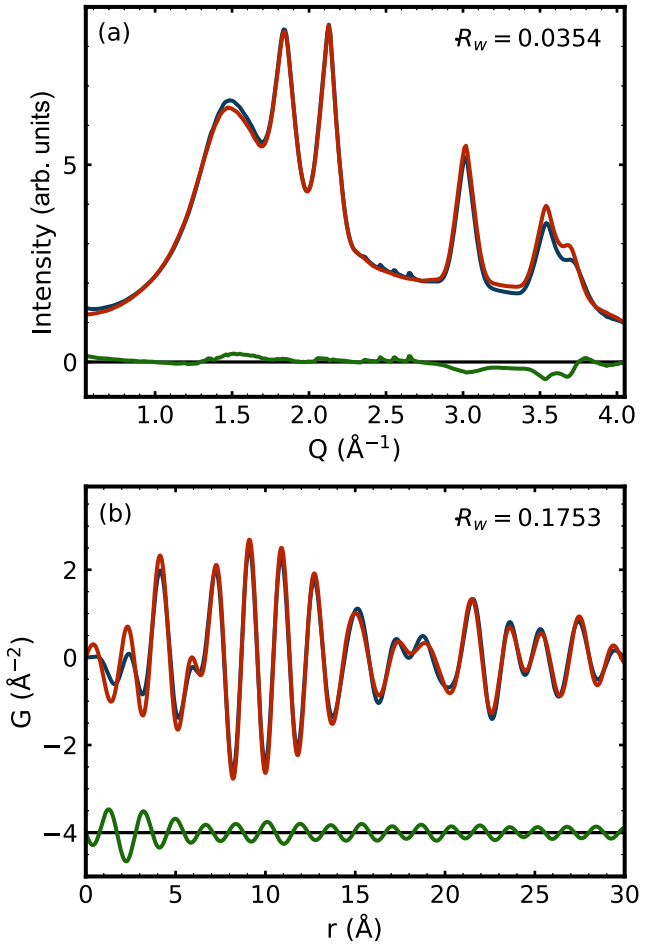


FIG. 11. (a) $I(Q)$'s computed from the uncalibrated but morphed data (blue) and calibrated (red) alignment parameters. (b) The PDFs computed from the two $I(Q)$'s in (a). Difference curves (green) are vertically offset below.

Such a morph allows us to make on-the-fly corrections for variability in sample calibration from dataset to dataset. However, it is also possible to invert the morph parameters to find spectrum-dependent corrections to the calibration values.

For the correction to the detector distance, we take advantage of the approximation that at small $Q \ll \lambda/4\pi$,

$$\epsilon_D = \frac{D_{\text{cal}} - D_{\text{unc}}}{D_{\text{unc}}} \approx -\eta, \tag{9}$$

where η is the `stretch` parameter as usual (proof in the Supplemental Material [5], Sec. G). To see how well this approximation worked, we applied the morphs to maximize agreement in only the lower-Q region (the first three peaks) in Fig. 10(c). The `stretch` parameter obtained in (c) was $\eta = 0.134$, which gives a correction to the sample-detector distance that is <4% away from the actual value of $\epsilon_D = -0.139$.

DIFFPY.MORPH includes a "`squeeze`" morphing transformation as well as the `stretch` and `shift` that we already introduced. The `squeeze` morph applies a higher-order polynomial to the x-array (see Sec. III H). In Fig. 11(a), we applied a `squeeze` using a fifth-order polynomial on the uncalibrated $I(Q)$ (blue) to attempt to match the properly calibrated $I(Q)$ (red). The use of a fifth-order polynomial in our `squeeze` morph is motivated in the Supplemental Material [5], Section G. We can see by comparing Figs. 10(b) and 10(c) and Fig. 11(a) that this quintic polynomial correction can, at the same time, account for the low-Q (near the 1.85 and 2.13 Å Bragg peaks) and high-$Q > 3.8$ Å^{-1} regions. The resulting PDFs are brought well into alignment [Fig. 11(b)]. As such, while the researcher's team is working to identify the new alignment parameters to recompute the $I(Q)$ correctly, the researcher can still analyze this new data, draw conclusions, and inform the next steps of the experiment using the DIFFPY.MORPH analysis.

As we discuss in the next section, in principle, for those using a Python program for integration of 2D detector images,

such as PYFAI [51], DIFFPY.MORPH can also help to obtain the optimal alignment parameters directly through the "`funcxy`" morph.

2. Obtaining data reduction parameters for an optimal PDF

So far we have used DIFFPY.MORPH to carry out rather simple morphing transformations such as `stretch`, `shift` and `scale` morphs. Complexity increases with `squeeze` that allows for nonlinear stretches of the data. However, in principle the morphing approach can be used for arbitrary transformations to the x- and y-arrays.

Here we describe an example in which we want to estimate the parameters used in processing raw 1D diffraction dataset all the way to a PDF. This processing is done using the PDFGETX3 software package [52], which takes a number of parameters such as how much to scale a background signal, the range of data to use in the Fourier transform, and *rpoly,* which controls the flexibility of a polynomial correction done in the process.

In this example, we have an $I(Q)$ measured from PbS nanoparticles and a thiostannate salt dissolved in an amide solution (the same sample as in Sec. II C 1). Our goal is to get a good PDF of the PbS after subtracting the correctly scaled signal from the solvent. The issue is illustrated in Fig. 12(a), where the raw signal from the sample (blue curve), scaled background (red curve), and background subtracted signal (green curve) are shown. After running this through PDFGETX3, we obtain the $F(Q)$ shown in Fig. 12(b). This has the wrong shape because the background has been undersubtracted and *rpoly* has not been optimized.

A more ideal background subtraction and $F(Q)$ is shown in Figs. 12(d) and 12(e), respectively. This was done using the `funcxy` morph in DIFFPY.MORPH and using a PDF calculated from a structural model of PbS as the target to morph onto [red curves in Figs. 12(c) and 12(f)].

PDFGETX3 does the PDF conversion by calling the `PDFGetter()` functor (which behaves like a function). This takes in the input data and a set of parameters such as background scales, *rpoly,* and so on and returns out the calculated $F(Q)$ and $G(r)$. Here we used `PDFGetter()` as the user-supplied morphing function in the `funcxy` morph in DIFFPY.MORPH, allowing it to vary the background scale and *rpoly* parameters with a target function that was a PDF calculated from a structural model for PbS.

This shows how an arbitrarily complicated function can be used for the morphing transformation (here the input and outputs were actually functions in different spaces, reciprocal and real, respectively, and the function involves a polynomial fit and subtraction). Although the morphing function being applied was complicated, the morph itself was not, in the spirit of the morphing philosophy, with only two well-controlled parameters being refined in such a way that real structural signal could not be lost.

The approach was very fast and allowed thousands of datasets to be rapidly processed each with its own optimized PDF transformation parameters.

The refinement process to obtain optimal PDF calculation parameters is particularly helpful for low $Q_{\max}$ data, especially for parameters like *rpoly* [52]. A scientist familiar with the PDF method may be able to estimate a good *rpoly* by seeing how it affects the overall shape of the $F(Q)$ function [52]. However, it is not always possible to measure over a wide enough Q-range to make good estimations [53,54] as we encountered recently when probing the ultrafast dynamics of a system at a recent x-ray free-electron laser (XFEL) experiment. This approach may also be valuable for laboratory-based sources that may also have low accessible $Q_{\max}$.

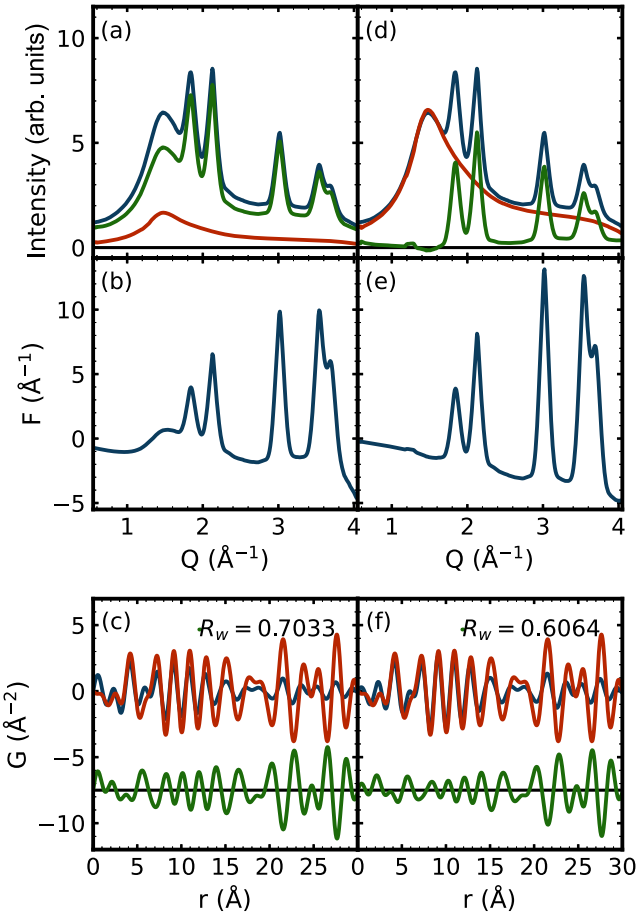


FIG. 12. (a),(d) The 1D scattering intensities of PbS nanoparticles dissolved in amide solution (blue), the raw signal from the empty solvent (red), and the sample signal with the background subtracted (green). Panels (b) and (e) show the resulting $F(Q)$ functions obtained by running the green dataset from the panel above through PDFGETX3 and the blue curves in (c) and (f) show the resulting PDFs. Parameters $rpoly = 0.9$ and $bgscale = 1$ were used to obtain the curves in (b) and (c), and $bgscale = 3.95$ and $rpoly = 1.78$ were used to obtain the curves in (e) and (f). The latter numbers were obtained by DIFFPY.MORPH. In (c) and (f) the red curve shows the PDF calculated from a bulk PbS structure model [47]. The green curve is the difference plotted offset below.

$Na_{2.9}Sb_{0.9}W_{0.1}S_4$ (NSWS) samples were synthesized following previously published procedures [55]. In Fig. 13(a), we plot $I(Q)$'s of the NSWS samples measured at EuXFEL [56] using the LPD detector at the FXE beamline [57] (plotted in blue) and measured at the National Synchrotron Light Source II on beamline 28-ID-2 (plotted in red).The XFEL measurements can be done at much higher time resolution

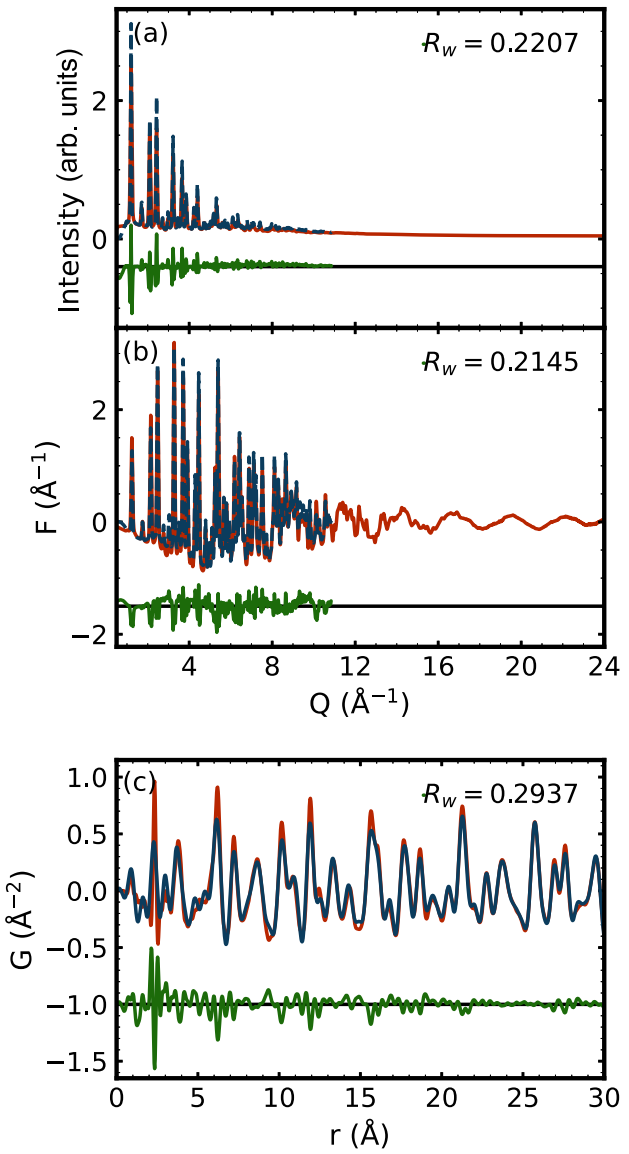


FIG. 13. (a) The 1D scattering intensities of NSWS measured at the XFEL over a narrow Q range (blue) and at the synchrotron over a wide Q range (red). (b) The $F(Q)$ obtained from the XFEL $I(Q)$ using PDFGETX3 and morphing (blue), and the target synchrotron $F(Q)$ (red). (c) The PDFs obtained by the morphed XFEL $F(Q)$ (blue) and target synchrotron $F(Q)$ (red).

compared to the synchrotron, at the cost of a lower $Q_{\max}$. In (b), we have obtained $F(Q)$ for the synchrotron data (red) by manually tuning the *rpoly* parameter until the large-Q behavior of $F(Q)$ becomes oscillations about zero. The largest Q ($Q_{\max} = 10.85$Å) for the XFEL data is too small to carry out the same analysis, making it difficult to tune *rpoly* that way. Instead, to obtain the blue curve in Fig. 13(b), we again wanted to use `PDFGetter()` from PDFGETX3 in the `funcxy` morph, but this time give the $F(Q)$ from the synchrotron experiment as a target function. In this case, in addition to applying the `PDFGetter()` corrections, it was necessary to apply a polynomial `squeeze` morph due to calibration challenges (see Sec. II C 1); these were applied together by defining a new Python function that wrapped `PDFGetter()`. Examples of how to do this can be found in the DIFFPY.MORPH documentation. The resulting PDF, plotted in (c), obtained from the morph-obtained XFEL $F(Q)$, is consistent with that obtained from the synchrotron $F(Q)$, albeit with slightly broader peaks due to the lower $Q_{\max}$.

These examples show how morphing provides a way for researchers to rapidly obtain PDFs that are as optimal as possible and process their data in a consistent manner.

III. MORPHING TRANSFORMATIONS

In this section, we define the current set of morphs available in DIFFPY.MORPH and develop some theory behind some of the most widely used morph transformations.

In the general setup of morphing, there is a "target" signal and a "morphed" signal. The target signal is unchanged, while the morphs are applied to the morphed signal.

The objective of the morphing process is to minimize the residual between the two signals by applying a series of simple "morphing" transformations. This is like fitting, but it is model-independent and it is designed simply to mimic physical effects, such as uniform lattice expansion or changes in thermal vibration, though it could be anything.

The signals could be any 1D functions or spectral signals, such as a diffraction patterns or atomic pair distribution functions, but they could also be Raman, NMR, or something else.

For simplicity, we develop the theory with the spectra being PDFs with the understanding that the morphs can be applied to other signals. Throughout this section, the unmorphed object PDF will be denoted $G(r)$, and the same PDF after morphing will be denoted as $G_m(r;\mathbf{p})$, where $\mathbf{p}$ denotes the set of morphing parameters.

A. Scaling morph

Scaling morphs correct the intensity of the morphed PDF by a constant s. Scaling simply captures the fact that a measured signal might be on a different scale due to differences in the incident flux during measurements, instruments used, uncertainties in the data reduction process, or any factor that serves to change the scale of a signal. The `scale` morph can be expressed simply as

$$G_m(r;s) = s \cdot G(r), \tag{10}$$

where s is the `scale` factor.

B. Stretching morph

Stretching morphs stretch the PDF linearly along the independent variable axis. Stretching simulates the effect of a uniform lattice expansion of the structure when η is positive, while $-1 < \eta < 0$ simulates a uniform lattice contraction.

The morph is done by linearly interpolating the r grid onto a new stretched r-grid, r'. We chose linear interpolation as most PDF data are stored on dense grids, so errors introduced are small and computational speed is a premium.

In detail, given a `stretch` factor η, the stretched distance, r', is given by

$$r' = \frac{r}{1+\eta}. \tag{11}$$

The resulting `stretch` morph is therefore given by

$$G_m(r;\eta) = G\left(\frac{r}{1+\eta}\right). \tag{12}$$

A real uniform lattice expansion with thermal strain ϵ will also rescale the PDF on the y-axis by a factor $(1+\epsilon)^{-2}$ (see the Supplemental Material [5], Sec. C). Therefore, we recommend using a `scale` morph any time a `stretch` morph is used to account for intensity changes due to isotropic expansion in the physical system.

C. Smearing morph (general)

Smearing morphs apply a Gaussian broadening factor to broaden a signal. For example, this will mimic different levels of thermal motion in PDF data (Sec. III D). To accomplish this, we carry out a convolution of the signal with a Gaussian function whose width can be varied.

The Gaussian broadening factor $T_\sigma(r)$ has the form

$$T_\sigma(r) = \exp\left\{-\frac{1}{2}\left(\frac{r-r_c}{\sigma}\right)^2\right\}, \tag{13}$$

where σ is the `smear` factor, and $r_c = \frac{1}{2}(r_{\max} + r_{\min})$ is the center of the r-grid. The choice of r_c is arbitrary, but centering the convolution Gaussian in the middle of the r-grid minimizes edge effects in the numerics of the resulting convolution.

The `smear` morph then convolves the morphed signal with the broadening function $T_\sigma(r)$ according to

$$R_m(r;\sigma) = \frac{(R \circledast T_\sigma)(r+r_c)}{\sqrt{2\pi\sigma^2}}. \tag{14}$$

This serves to broaden all peaks in $R(r)$; specifically, the variance of each Gaussian peak is increased by σ^2 after convolution. The $r + r_c$ shifts the smeared curve to ensure its centroid matches that of $R(r)$, and the normalization factor $1/\sqrt{2\pi\sigma^2}$ ensures the two curves have the same integrated magnitudes. Proof that this operation indeed broadens the peaks of $R(r)$ by σ^2 while maintaining the centers and integrated magnitudes of each peak can be found in the Supplemental Material [5], Sec. E.

Note that $T_\sigma(r) = T_{-\sigma}(r)$, so a `smear` parameter of $+\sigma$ is equivalent to a `smear` parameter of $-\sigma$.

D. Smearing morph (PDF-specific)

The `smear` morph described above can be used to smear any signal. When that signal is a measured $G(r)$ function, it is preferred to convert $G(r)$ to the radial distribution function (RDF [2]) before applying the `smear`. As such, DIFFPY.MORPH contains a modified version of the `smear` morph, called `smear-pdf`. This is because atomic motion acts to broaden peaks in the RDF, $R(r)$, and not the PDF $G(r)$.

The differences are small, and an approximately correct result can be obtained by smearing $G(r)$ directly. However, for more accurate results when dealing with PDFs, a user may use the `smear-pdf` morph instead of the `smear` morph. At the command line, this is done by using the `--smear-pdf` option instead of the `--smear` option. When using DIFFPY.MORPH in a Python script, you would use the `smear_pdf` parameter instead of `smear`. If both `smear-pdf` and `smear` options are used, the program ignores the `smear` input and applies the `smear-pdf` morph only.

For bulk materials, the RDF is related to the PDF by [41]

$$G(r) = \frac{R(r)}{r} - 4\pi r\rho_0, \tag{15}$$

where ρ_0 is the atomic density of the material, and $-4\pi\rho_0$ may be referred to as the "baseline slope." When `smear-pdf` is used, the input functions are assumed to be PDFs. The program will first convert it to an RDF, $R(r)$, using either a user-given baseline slope or a best-fit slope if the user is unable to provide one. (This best-fit slope is obtained by adding a "baseline slope" parameter to the set of refined parameters. In our use cases, we have found that the optimized σ is relatively insensitive to the choice of baseline slope.) The smear morph then applies a Gaussian broadening factor $T_\sigma(r)$ to $R(r)$. The morphed $R_m(r)$ is then converted back to a PDF, which is then returned as the morphed PDF.

E. Shifting morph

Shifting morphs shift the signal horizontally or vertically. They may be needed to account for constant backgrounds or calibration errors such as a shifted zero point in a diffraction pattern. Shifting is also useful for data visualization techniques (e.g. waterfall plotting).

Given a user-specified horizontal `shift` parameter Δ_h and vertical `shift` parameter Δ_v, the `shift` morph is expressed simply as

$$G_m(r) = G(r - \Delta_h) + \Delta_v. \tag{16}$$

F. Nanoparticle shape morph in the PDF

PDFs measured from a nanoparticle and bulk sample of the same material can look quite different as the particle's shape plays a role in shaping the pair distribution function. One can approximate the PDF of a nanoparticle $G_{\text{nano}}(r)$ from the PDF of a bulk $G_{\text{bulk}}(r)$ of the same material by the simple relation [41]

$$G_{\text{nano}}(r) \approx \gamma_0(r) G_{\text{bulk}}(r). \tag{17}$$

The `shape` and inverse `shape` morphs in DIFFPY.MORPH were designed to compare bulk and nanoparticle PDFs using the nanoparticle characteristic function, $\gamma_0(r;\mathbf{p})$. This characteristic function depends on r and on some parameters $\mathbf{p}$.

The `shape` morphs take input $G(r) = G_{\text{bulk}}(r)$ and simply multiply them by the characteristic function, $\gamma_0(r;\mathbf{p})$,

$$G_m(r;\mathbf{p}) = \gamma_0(r;\mathbf{p}) \cdot G(r). \tag{18}$$

The inverse `shape` morphs take input $G(r) = G_{\text{nano}}(r)$ and divide the characteristic function,

$$G_m(r) = \begin{cases} \frac{G(r)}{\gamma_0(r;\mathbf{p})}, & \gamma_0(r;\mathbf{p}) \neq 0, \\ 0, & \gamma_0(r,\mathbf{p}) = 0. \end{cases} \tag{19}$$

Currently, DIFFPY.MORPH has nanoparticle characteristic functions for spheres [58] and spheroids [59]. The spherical nanoparticle characteristic function, $\gamma_{0,\text{sph}}(r;s)$, refines the radius, s, of the nanoparticle. The spheroidal nanoparticle

characteristic function, $\gamma_{0,\mathrm{srd}}(r; s_e, s_p)$, refines the equatorial and polar radii, s_e and s_p, respectively.

G. General ordinate morph

In addition to the morphs introduced above, which are designed to account for specific physical effects, `funcy` provides a flexible interface for applying user-defined transformations to the y-axis data, enabling comparison of signals influenced by experiment- or system-specific distortions. The definition of the `funcy` morph is

$$G_m(r; f, \mathbf{p}) = f(r, G(r); \mathbf{p}), \tag{20}$$

where f is a user-supplied function in the form of a Python function, and $\mathbf{p}$ represents its parameter dictionary. The user simply defines a Python function that takes the signal r and $G(r)$ arrays (or equivalently, any x and y arrays) and a dictionary of parameters then returns the morphed x and y arrays. This function is passed to the `funcy` interface along with the morphed and target functions, and the parameters are refined to get a good agreement.

H. General grid (abscissa) morphs

Similarly it is possible to apply user-defined morphs that transform the x-axis of the function, generalizing the `stretch` morph. Below, we describe two morphs: `squeeze` and `funcx`. For reasons discussed at the end of this (sub)section, `squeeze` should be used when the expected change in the x-axis grid is small, while `funcx` could be used otherwise.

`squeeze` applies a polynomial transformation to the x-axis, where the user can control the order of the polynomial. This offers a flexible approach for correcting nonlinear distortions in signals. A zeroth-order polynomial is actually the `shift` morph, and a first-order polynomial recovers the behavior of `shift` and `stretch` morphs applied together. But the `squeeze` interface generalizes this to higher orders.

These morphs have been found useful to account for geometric artifacts introduced by 2D detectors, such as module tilts, curved detection planes, or angle-dependent offsets, as well as structural effects intrinsic to the sample. Its use was introduced in Sec. II C 1.

The transformation is defined as

$$\Delta r(r; a_0, \ldots, a_n) = a_0 + a_1 r + a_2 r^2 + \cdots + a_n r^n, \tag{21}$$

where $\{a_0, a_1, \ldots, a_n\}$ are the polynomial coefficients. The desired order, n, of the polynomial to use is a user-defined input. As with all polynomial fitting methods, one should be careful in their choice of n and avoid overfitting. This is applied as

$$r'(r; a_0, \ldots, a_n) = r + \Delta r(r; a_0, \ldots, a_n), \tag{22}$$

resulting in a modified coordinate grid with the morphed signal evaluated as

$$G_m(r') = G(r). \tag{23}$$

Given a morph and target function, DIFFPY.MORPH will find the best set of $a_0, \ldots, a_n$ to apply in order to match the target.

To compare to previously described morphs, if all $a_i = 0$ except $a_0 \neq 0$, `squeeze` is equivalent to the horizontal `shift` morph with $\Delta_h = a_0$ (Sec. III E), and if instead all $a_i = 0$ except $a_1 \neq 0$, `squeeze` is equivalent to a `stretch` morph with $\eta = a_1$ (Sec. III B).

The current implementation of the `squeeze` morph takes the original function as a two-column array $[[r], [G(r)]]$, where the notation $[\cdot]$ indicates that the quantity is an array. Then, `squeeze` computes $[r']$ using Eq. (23) to get $[[r'], [G_m(r')]]$. Finally, `squeeze` reinterpolates $G_m(r')$ back onto the original grid $[r]$ and returns the squeezed function interpolated onto the original grid, i.e., $[[r], [G_m(r)]]$. The reinterpolation is done for consistency and further processing. This is computed using a cubic spline interpolation. Extrapolation is allowed outside the original domain where needed (and the user will be notified if and where the extrapolation occurs). Care should be taken by the user if the morph results in a large extrapolation.

The `squeeze` morph will warn the user if the polynomial squeeze results in a modified grid, $[r'(r)]$, that is not monotonically increasing. Nonmonotonicity of r' poses a problem in our procedure above, as $G_m(r')$ may no longer be a single-valued function. This means there is no longer a unique way to interpolate $[[r'], [G_m(r')]]$ onto the grid $[r]$. We resolve this by modifying $G_m(r')$ to be a single-valued function by first identifying all duplicated grid points r'_{dup} in the array $[r'(r)]$; i.e., $r'_1 = r'_2 = \cdots r'_s = r'_{\mathrm{dup}}$ for $s \geqslant 2$. We then set

$$G_m(r'_{\mathrm{dup}}) = \sum_{i=1}^{s} G_m(r'_i), \tag{24}$$

giving $G_m(r')$ a single unique value for each grid point r'_{dup} where it may be multivalued. This procedure can result in an $[[r], [G_m(r)]$ that looks vastly different from $[[r], [G(r)]]$ if the function $G_m(r')$ is not smooth or the length of the nonmonotonic intervals is large, but seems to work well for small extrapolations that are the most common.

In general, we advise users to avoid a nonmonotonic `squeeze`. We have found success by (i) reducing the order of the polynomial and (ii) first performing safe morphs like `shift` and `stretch` to get estimates of a_0 and a_1, and giving those to `squeeze` as the initial parameter values.

While `squeeze` can only apply polynomials to the grid, the `funcx`, in analogy with `funcy`, morph allows arbitrary functions. The morphed signal is similarly given by Eq. (23), though the new grid r' is now given by

$$r'(r; f, \mathbf{p}) = f(r, G(r); \mathbf{p}). \tag{25}$$

Here, f is a user-provided Python function, and $\mathbf{p}$ is a dictionary of function parameters to refine over. The function f should take in both r and $G(r)$ (equivalently, x and y) and return the morphed grid $r'(r; \mathbf{p})$ based on the parameters $\mathbf{p}$.

The implementation of the `funcx` morph is slightly different from that of `squeeze`. After taking in $[[r], [G(r)]$, we simply return $[[f(r, G(r); \mathbf{p}])], [G(r)]]$. By design, no interpolation onto the original grid is done in this case. This choice allows arbitrary changes in the independent array, permitting, for example, functions that transform into different spaces, such as the `PDFGetter()` transform described in Sec. II C 2.

The advantage to `funcx` is apparent when the morph and target function grids are extremely different as `squeeze` could result in strange and uncontrolled behavior. However, this

means the grid of the morphed function is constantly changing. This makes computing the residual between the morph and target functions difficult, and the calculation of the difference curve can involve varying amounts of interpolation error for each iteration of the refinement. As such, we still recommend the `squeeze` morph for small changes to the grid as it operates on a single unchanging grid.

I. General function morph

While a user can use the `funcx` and `funcy` morphs together to change both the grid and data simultaneously, we have defined a `funcxy` morph for convenience. This morph is designed to be compatible with PDF calculators such as the `PDFGetter()` functor in PDFGETX3 [52], so one can refine the PDF calculation from a measured intensity function. This was described in Sec. II C 2.

For this morph, the user supplies a single function **f**,

$$\mathbf{f}(r, G(r); \mathbf{p}) = (f_0(r, G(r); \mathbf{p}), f_1(r, G(r); \mathbf{p})), \tag{26}$$

that takes in the arrays r and $G(r)$ as well as a dictionary of parameters **p**. The function must return two quantities: (a) the desired morph on the grid f_0 and (b) the desired morph on the function values f_1. In Python, this can be done as a tuple (f_0, f_1) or as two comma-separated quantities f_0, f_1.

The `funcxy` morph is then described by

$$G_m(r'; f_1, \mathbf{p}) = f_1(r, G(r); \mathbf{p}), \tag{27}$$

where

$$r'(r; f_0, \mathbf{p}) = f_0(r, G(r); \mathbf{p}). \tag{28}$$

One can see the obvious parallels to Eqs. (20) and (25).

J. Chaining together multiple morphs

All morphs (`scale`, `stretch`, `smear`, `smear-pdf`, `shape`, `squeeze`, `funcx`, `funcy`, `funcxy`, etc.) can be chained together with other morphs to incorporate multiple corrections. To do so, a user simply has to enable the morphs they want to be chained.

There are some limits to this, and the user should use good sense. In some cases, the code will prevent clashes. For example, only one of the `smear` and `smear-pdf` morphs can be used. If the user accidentally enables both, only the more specific `smear-pdf` morph will be done. Also, since a polynomial `squeeze` morph of order zero or higher includes the horizontal `shift` morph, the horizontal `shift` morph will be disabled when the `squeeze` morph is enabled (the vertical `shift` morph is not affected). For the same reason, if the polynomial `squeeze` is of order 1 or higher, the `stretch` is disabled.

Users enable morphs by specifying them and initializing the parameter values. Additionally, there are convenience commands to fix and free previously specified morphs. Detailed usage can be found in the documentation of the software.

When multiple morphs are specified, a predefined order is applied. Since we found the optimizer to perform better by applying more complex morphs first, the default order in which enabled morphs are applied is (at the time or writing) as follows: `funcxy`, `funcx`, `funcy`, `squeeze`, `scale`, `stretch`, `smear` or `smear-pdf`, `shift` (horizontal and vertical are refined simultaneously), then `shape`. This order may change in the future. One can also choose their own order of refinement by applying the morphs in stages as displayed in our quickstart tutorial on the DIFFPY.MORPH software documentation page (see Sec. IV). We recommend first refining the most sensitive parameters, which are generally associated with the more complex morphs. We have observed that when morph parameters are near optimal results, there is little difference in chaining the morphs together versus the step-by-step process.

K. Regression targets

For all morphs, the morphing process uses least-squares regression to minimize the difference between the morphed and target signals by varying morphing parameters, where R_w is used as the objective function.

If the morph and target functions are not on the same grid, the two functions are first interpolated (using linear interpolation) onto a common grid before computing R_w. The interval of this common grid is the intersection of the two original grid intervals (i.e., if the morphed grid is on the interval $[a, b]$ and the target grid is on the interval $[c, d]$, then the new grid interval is $[\max\{a, c\}, \min\{b, d\}]$). If the spacing (the distance between consecutive grid points) between the two grids is different, then the common grid uses the larger of the two spacings. If a grid has uneven spacing (i.e., the distance between consecutive grid points varies), then its spacing is defined as the average distance between consecutive grid points.

IV. DIFFPY.MORPH SOFTWARE

DIFFPY.MORPH is an open-source Python software package that runs on Windows, MacOS, and Linux operating systems as part of the DIFFPY project [60]. It can be installed from conda-forge [20] using conda (recommended) or from PyPI [19] using pip. The source code is available on GitHub [48].

The DIFFPY.MORPH program can be used both using a command line interface (e.g., a bash terminal) or in Python scripts directly.

The program takes in two input files: the file containing the function to be morphed and the file containing the target function. Readable files contain the data as a multicolumn table, where one column contains the x-array and another column has the function values. This uses the `loadData()` method in DIFFPY.UTILS [61]. By default, when `loadData()` encounters two columns, it assumes the first column is the x- and the second column is the y-array, but this behavior can be overridden. `loadData()` will skip over header information or it can be used to read and return header information.

The user also provides a set of morphs to apply (e.g., `scale`, `stretch`, and `smear`). The program then carries out the morphs described above, returning the optimized morphing parameters. It contains capabilities to plot the morphed function and the reference function with a difference function below, similar to figures in this paper. If requested, the program also returns the morphed function interpolated to the grid of the reference function.

The package also has an API allowing morphs to be called from within Python scripts.

Tutorials for the software and additional code documentation are available in the DIFFPY.MORPH documentation [62].

V. CONCLUSION

We demonstrate morphing as a model-independent tool for comparing 1D spectral data with worked examples that show how useful information may be extracted from diffraction and PDF data. As DIFFPY.MORPH uses no model/structural refinement, it is fast, making it ideal for on-the-fly analysis in high-throughput data settings such as a synchrotron experiment.

While initially developed for x-ray PDFs, our examples show that the technique also works for 1D diffraction patterns and neutron diffraction data. Since DIFFPY.MORPH is model-independent, it can be used on other 1D functions (e.g., density of states, Raman spectra, flattened image data), though it has not been extensively tested beyond powder diffraction patterns and PDFs.

ACKNOWLEDGMENTS

Algorithm design as well as Billinge-group involvement in the experiments and analysis, was supported by U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences (DOE-BES) under Contract No. DE-SC0024141. Software development and paper writing was supported by the W. M. Keck Foundation under Award No. 1567502. A.Y. also acknowledges support from DOE Office of Science, BES, Contract No. DE-FG02-03ER46055 during manuscript preparation. Initial development (supporting P.J. and C.L.F.) of DIFFPY.MORPH was done under the Distributed Analysis of Neutron Scattering Data (DANSE) project funded by the U.S. National Science Foundation (NSF) under DMR-0520547. Further developments by C.-H.L. were supported by the Next Generation Synthesis Center (GENESIS), an Energy Frontier Research Center funded by the U.S. Department of Energy, Office of Science, Basic Energy Sciences under Award No. DE-SC0019212. X-ray PDF measurements of the iridium telluride samples were conducted on beamline 28-ID-2 of the National Synchrotron Light Source II, a U.S. Department of Energy (DOE) Office of Science User Facility operated for the DOE Office of Science by Brookhaven National Laboratory under Contract No. DE-SC0012704. This contract also supported C.P.'s synthesis of the $IrTe_2$ and $Ir_{0.8}Rh_{0.2}Te_2$ samples. Crystal growth of the $Ir_{0.95}Pt_{0.05}Te_2$ samples by H.Z. was supported by the National Science Foundation Grant No. DMR-2003117. E.S.B. acknowledges support from the European Union's Horizon Europe research and innovation programme under Grant Agreement No. 101185375. X-ray PDF measurements of the PbS nanocrystals were taken at the SACLA beamline 2 at SPring-8 faclility in Hyogo, Japan. N.S.G. acknowledges support for the data acquisition by U.S.-DOE-BES Contract No. DE-SC0019375 and a David and Lucile Packard Foundation Fellowship for Science and Engineering. V.R.K.W.'s involvement in the experiment was supported by NSF Graduate Research Fellowship No. DGE1106400. A.J. and D.V.T. acknowledge support from the U.S. Department of Energy, Office of Science, Basic Energy Sciences, Materials Sciences and Engineering Division, under Grant No. DE-SC0025256. A.J. was also partially supported by the Kwanjeong Educational Foundation. We would like to thank Yuya Kubota and Jungmin Kang, Tadashi Togashi, Taito Osaka, and Kensuke Tono for help with the data acquisition. The NSWS experiments were carried out at the European XFEL in Schenefeld, Gemany at beamline FXE under Proposal No. 8015. The authors would like to thank Paul Evans, Peter Zalden, and Christopher Milne for their help with these experiments, and Sanjit Ghose for help with collecting synchrotron data on these samples at beamline 28-ID-2 at NSLS-II (funding noted above). L.F.N. would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) for funding through a platform Discovery Grant and a Canada Research Chair. The Neutron PDF measurements of YSZ used resources at beamline BL-1B (NOMAD) at the Spallation Neutron Source, a DOE Office of Science User Facility operated by the Oak Ridge National Laboratory. Special thanks is given to the SNS staff, Joerg Neuefeind, Jue Liu, Emily Van Aucken, and Dante Quirinale for aid in data collection and reduction as well as experiment design and planning. Thanks is also given to Margarita Petrova from the Page group for aiding in data collection, sample preparation, and data reduction.

DATA AVAILABILITY

The open source software is available at GitHub [48], on PyPI [19], and on conda-forge [20] with the name DIFFPY.MORPH in the DIFFPY organization on GitHub. Data used in the figures are openly available on Zenodo [63] or by request to the authors.

[1] R. E. Dinnebier and S. J. L. Billinge, in *Powder Diffraction*, edited by R. E. Dinnebier and S. J. L. Billinge (Royal Society of Chemistry, Cambridge, 2008), pp. 1–19.

[2] T. Egami and S. J. L. Billinge, *Underneath the Bragg Peaks: Structural Analysis of Complex Materials*, 2nd ed., Pergamon Materials Series Vol. 16 (Elsevier, Amsterdam, 2012).

[3] S. J. L. Billinge and K. M. Ø. Jensen, *Atomic Pair Distribution Function Analysis—A Primer*, International Union of Crystallography Texts on Crystallography (Oxford University Press, Oxford, England, 2024).

[4] K. Pearson and F. Galton, VII. Note on regression and inheritance in the case of two parents, Proc. R. Soc. London **58**, 240 (1895).

[5] See Supplemental Material at http://link.aps.org/supplemental/10.1103/t39r-v3lx for more physical and mathematical justifications for when one can apply the various morphs, which also contains Refs. [6–11].

[6] J. M. Léger, A. Pereira, J. Haines, S. Jobic, and R. Brec, Phase transformations of polymeric CdI_2-type $IrTe_2$ under high pressure, J. Phys. Chem. Solids **61**, 27 (2000).

[7] P. J. Brown, A. G. Fox, E. N. Maslen, M. A. O'Keefe, and B. T. M. Willis, Intensity of diffracted intensities, in *International Tables for Crystallography Volume C: Mathematical, Physical and Chemical Tables*, edited by E. Prince (Springer Netherlands, Dordrecht, 2006), pp. 554–595.

[8] M. L. Boas, *Mathematical Methods in the Physical Sciences* (Wiley, Hoboken, NJ, 2006).

[9] R. E. Dinnebier and S. J. L. Billinge, Overview and principles of powder diffraction, in *International Tables for Crystallography* (International Union of Crystallography, 2019), Vol. H, Chap. 1.1, pp. 2–23.

[10] L. A. Feigin, D. I. Svergun, and G. W. Taylor, General principles of small-angle diffraction, in *Structure Analysis by Small-Angle X-Ray and Neutron Scattering*, edited by G. W. Taylor (Springer US, Boston, 1987), pp. 25–55.

[11] L. C. Evans and R. F. Gariepy, *Measure Theory and Fine Properties of Functions*, 1st ed., Textbooks in Mathematics (Chapman and Hall/CRC, Boca Raton, FL, 2015).

[12] G. Tomasi, F. van den Berg, and C. Andersson, Correlation optimized warping and dynamic time warping as preprocessing methods for chromatographic data, J. Chemom. **18**, 231 (2004).

[13] P. H. C. Eilers, Parametric time warping, Anal. Chem. **76**, 404 (2004).

[14] Q. P. He, J. Wang, J. A. Mobley, J. Richman, and W. E. Grizzle, Self-calibrated warping for mass spectra alignment, Cancer Inform. **10**, 65 (2011).

[15] J. Kieffer and D. Karkoulis, PyFAI, a versatile library for azimuthal regrouping, J. Phys.: Conf. Ser. **425**, 202012 (2013).

[16] B. H. Toby and R. B. Von Dreele, *GSAS-II*: The genesis of a modern open-source all purpose crystallography software package, J. Appl. Crystallogr. **46**, 544 (2013).

[17] O. Arnold, J. Bilheux, J. Borreguero, A. Buts, S. Campbell, L. Chapon, M. Doucet, N. Draper, R. Ferraz Leal, M. Gigg, V. Lynch, A. Markvardsen, D. Mikkelson, R. Mikkelson, R. Miller, K. Palmen, P. Parker, G. Passos, T. Perring, P. Peterson, *et al.*, Mantid—Data analysis and visualization package for neutron scattering and μ SR experiments, Nucl. Instrum. Methods Phys. Res. Sect. A **764**, 156 (2014).

[18] Regents of the University of California, The 3-clause BSD license (1989), https://spdx.org/licenses/BSD-3-Clause.html.

[19] A. Yang, C. L. Farrow, C.-H. Liu, L. Kitsu, and S. J. L. Billinge, diffpy.morph (PyPI package), 2025, https://pypi.org/project/diffpy.morph.

[20] conda-forge community, diffpy.morph (Conda-forge package), 2025, https://anaconda.org/channels/conda-forge/packages/diffpy.morph/overview.

[21] N. Lazarević, E. S. Bozin, M. Šćepanović, M. Opačić, H. Lei, C. Petrovic, and Z. V. Popović, Probing $IrTe_2$ crystal symmetry by polarized Raman scattering, Phys. Rev. B **89**, 224301 (2014).

[22] Y. Liu, H. Lei, K. Wang, M. Abeykoon, J. B. Warren, E. Bozin, and C. Petrovic, Thermoelectric studies of $Ir_{1-x}Rh_xTe_2$ ($0 \leqslant x \leqslant 0.3$), Phys. Rev. B **98**, 094519 (2018).

[23] J. J. Yang, Y. J. Choi, Y. S. Oh, A. Hogan, Y. Horibe, K. Kim, B. I. Min, and S.-W. Cheong, Charge-orbital density wave and superconductivity in the strong spin-orbit coupled $IrTe_2$: Pd, Phys. Rev. Lett. **108**, 116402 (2012).

[24] S. Pyon, K. Kudo, and M. Nohara, Superconductivity induced by bond breaking in the triangular lattice of $IrTe_2$, J. Phys. Soc. Jpn. **81**, 053701 (2012).

[25] R. Yu, S. Banerjee, H. C. Lei, R. Sinclair, M. Abeykoon, H. D. Zhou, C. Petrovic, Z. Guguchia, and E. S. Bozin, Absence of local fluctuating dimers in superconducting $Ir_{1-x}(Pt, Rh)xTe_2$, Phys. Rev. B **97**, 174515 (2018).

[26] I.-K. Jeong, R. H. Heffner, M. J. Graf, and S. J. L. Billinge, Lattice dynamics and correlated atomic motion from the atomic pair distribution function, Phys. Rev. B **67**, 104301 (2003).

[27] R. Yu, S. Banerjee, H. Lei, M. Abeykoon, C. Petrovic, Z. Guguchia, and E. S. Bozin, Phase separation at the dimer-superconductor transition in $Ir_{1-x}Rh_xTe_2$, Phys. Rev. B **98**, 134506 (2018).

[28] C. L. Farrow, P. Juhás, J. W. Liu, D. Bryndin, E. S. Božin, J. Bloch, T. Proffen, and S. J. L. Billinge, PDFfit2 and PDFgui: Computer programs for studying nanostructure in crystals, J. Phys.: Condens. Matter **19**, 335219 (2007).

[29] B. Frandsen, L. Yang, and S. J. L. Billinge, pdfttp_data, 2023, https://github.com/Billingegroup/pdfttp_data.

[30] E. Grüneisen, Theorie des festen Zustandes einatomiger Elemente, Ann. Phys. **344**, 257 (1912).

[31] Y. Chang, On the temperature dependence of the bulk modulus and the Anderson-Grüneisen parameter δ of oxide compounds, J. Phys. Chem. Solids **28**, 697 (1967).

[32] D. Schroeder, *An Introduction to Thermal Physics* (Oxford University Press, 2021).

[33] O. L. Anderson, Derivation of Wachtman's equation for the temperature dependence of elastic moduli of oxide compounds, Phys. Rev. **144**, 553 (1966).

[34] S. V. Ushakov, A. Navrotsky, R. J. K. Weber, and J. C. Neuefeind, Structure and thermal expansion of YSZ and $La_2Zr_2O_7$ above 1500°C from neutron diffraction on levitated samples, J. Am. Ceram. Soc. **98**, 3381 (2015).

[35] P. Aldebert, Neutron and X-ray experiments at high temperature, Rev. Phys. Appl. (Paris) **19**, 649 (1984).

[36] M. I. Ahmad, D. G. Van Campen, J. D. Fields, J. Yu, V. L. Pool, P. A. Parilla, D. S. Ginley, M. F. A. M. Van Hest, and M. F. Toney, Rapid thermal processing chamber for *in-situ* X-ray diffraction, Rev. Sci. Instrum. **86**, 013902 (2015).

[37] H. Hayashi, T. Saitou, N. Maruyama, H. Inaba, K. Kawamura, and M. Mori, Thermal expansion coefficient of yttria stabilized zirconia for various yttria contents, Solid State Ion. **176**, 613 (2005).

[38] J. A. Krogstad, Y. Gao, J. Bai, J. Wang, D. M. Lipkin, and C. G. Levi, *In situ* diffraction study of the high-temperature decomposition of *t'*-zirconia, J. Am. Ceram. Soc. **98**, 247 (2015).

[39] M. Goyal and M. Singh, Size and shape dependence of optical properties of nanostructures, Appl. Phys. A **126**, 176 (2020).

[40] S. H. Tolbert and A. P. Alivisatos, Size dependence of a first order solid-solid phase transition: The Wurtzite to rock salt transformation in CdSe nanocrystals, Science **265**, 373 (1994).

[41] C. L. Farrow and S. J. L. Billinge, Relationship between the atomic pair distribution function and small-angle scattering: Implications for modeling of nanoparticles, Acta Crystallogr. A **65**, 232 (2009).

[42] S. Banerjee, A. Zangiabadi, A. Mahdavi-Shakib, S. Husremovic, B. G. Frederick, K. Barmak, R. N. Austin, and S. J. L. Billinge, Quantitative structural characterization of catalytically active TiO_2 nanoparticles, ACS Appl. Nano Mater. **2**, 6268 (2019).

[43] S. Banerjee, C.-H. Liu, K. M. Ø. Jensen, P. Juhás, J. D. Lee, M. Tofanelli, C. J. Ackerson, C. B. Murray, and S. J. L. Billinge,

Cluster-mining: An approach for determining core structures of metallic nanoparticles from atomic pair distribution function data, Acta Crystallogr A **76**, 24 (2020).

[44] S. Tao, J. Billet, J. De Roo, and S. J. L. Billinge, Rapid modeling of the local structure of metal oxide nanoparticles from PDF data: A case study using TiO_2 nanoparticles, Chem. Mater. **36**, 10912 (2024).

[45] A. S. Masadeh, E. S. Božin, C. L. Farrow, G. Paglia, P. Juhás, S. J. L. Billinge, A. Karkamkar, and M. G. Kanatzidis, Quantitative size-dependent structure and strain determination of CdSe nanoparticles using atomic pair distribution function analysis, Phys. Rev. B **76**, 115413 (2007).

[46] B. Guzelturk, J. Portner, J. C. Ondry, S. Ghanbarzadeh, M. Tarantola, A. Jeong, T. Field, A. M. Chandler, E. Wieman, T. R. Hopper, N. E. Watkins, J. Yu, X. Cheng, M.-F. Lin, D. Luo, P. L. Kramer, X. Shen, A. H. Reid, O. Borkiewicz, U. Ruett, *et al.*, Ultrafast symmetry control in photoexcited quantum dots, Adv. Mater. **37**, 2414196 (2025).

[47] Y. Noda, K. Masumoto, S. Ohba, Y. Saito, K. Toriumi, Y. Iwata, and I. Shibuya, Temperature dependence of atomic thermal parameters of lead chalcogenides, PbS, PbSe and PbTe, Acta Crystallogr. Sect. C **43**, 1443 (1987).

[48] A. Yang, C. L. Farrow, C.-H. Liu, L. Kitsu, and S. J. L. Billinge, diffpy.morph, 2025, https://github.com/diffpy/diffpy.morph.

[49] https://groups.google.com/g/diffpy-users.

[50] A. Jeong, J. Portner, C. P. N. Tanner, J. C. Ondry, C. Zhou, Z. Mi, Y. A. Tazoui, B. Lee, V. R. K. Wall, N. S. Ginsberg, and D. V. Talapin, Colloidal dispersions of sterically and electrostatically stabilized PbS quantum dots: Structure factors, second virial coefficients, and film-forming properties, ACS Nano **18**, 33864 (2024).

[51] J. Kieffer, J. Orlans, N. Coquelle, S. Debionne, S. Basu, A. Homs, G. Santoni, and D. De Sanctis, Application of signal separation to diffraction image compression and serial crystallography, J. Appl. Crystallogr. **58**, 138 (2025).

[52] P. Juhas, T. Davis, C. L. Farrow, and S. J. L. Billinge, *PDFgetX3*: A rapid and highly automatable program for processing powder diffraction data into total scattering pair distribution functions, J. Appl. Crystallogr. **46**, 560 (2013).

[53] F. Bertolotti and I. Robinson, Scientific opportunities with very hard XFEL radiation, XFEL.EU WR-2024-002 (2024).

[54] A. F. Sapnik, P. A. Chater, D. S. Keeble, J. S. O. Evans, F. Bertolotti, A. Guagliardi, L. J. Støckler, E. A. Harbourne, A. B. Borup, R. S. Silberg, A. Descamps, C. Prescher, B. D. Klee, A. Phelipeau, I. Ullah, K. G. Medina, T. A. Bird, V. Kaznelson, W. Lynn, A. L. Goodwin, *et al.*, High-quality ultra-fast total scattering and pair distribution function data using an X-ray free-electron laser, IUCrJ **12**, 531 (2025).

[55] T. Fuchs, S. P. Culver, P. Till, and W. G. Zeier, Defect-mediated conductivity enhancements in $Na_{3-x}Pn_{1-x}W_xS_4$ (Pn = P, Sb) using aliovalent substitutions, ACS Energy Lett. **5**, 146 (2020).

[56] W. Decking, in *Brilliant Light in Life and Material Sciences*, edited by V. Tsakanov and H. Wiedemann (Springer Netherlands, Dordrecht, 2007), pp. 21–30.

[57] A. Galler, W. Gawelda, M. Biednov, C. Bomer, A. Britz, S. Brockhauser, T.-K. Choi, M. Diez, P. Frankenberger, M. French, D. Görries, M. Hart, S. Hauf, D. Khakhulin, M. Knoll, T. Korsch, K. Kubicek, M. Kuster, P. Lang, F. Alves Lima, *et al.*, Scientific instrument femtosecond X-ray experiments (FXE): Instrumentation and baseline experimental capabilities, J. Synch. Radiat. **26**, 1432 (2019).

[58] K. Kodama, S. Iikubo, T. Taguchi, and S.-i. Shamoto, Finite size effects of nanoparticles on the atomic pair distribution functions, Acta Crystallogr. A **62**, 444 (2006).

[59] M. Lei, A. M. R. de Graff, M. F. Thorpe, S. A. Wells, and A. Sartbaeva, Uncovering the intrinsic geometry from the atomic pair distribution function of nanomaterials, Phys. Rev. B **80**, 024118 (2009).

[60] DiffPy team, Diffpy - atomic structure analysis in Python (2025), https://www.diffpy.org/.

[61] P. Juhás, C. L. Farrow, X. Yang, K. R. Knox, A. Yang, and S. J. L. Billinge, diffpy.utils, 2025, https://github.com/diffpy/diffpy.utils.

[62] https://diffpy.org/diffpy.morph/.

[63] diffpy.morph authors, Data used for figures in diffpy.morph manuscript [Data set], Zenodo, 2026, https://doi.org/10.5281/zenodo.20680507.