Digital Discovery

ROYAL SOCIETY OF CHEMISTRY

PAPER

View Article Online
View Journal | View Issue

Cite this: *Digital Discovery*, 2025, **4**, 2579

nDTomo: a modular Python toolkit for X-ray chemical imaging and tomography

A. Vamvakeros, *[ab] E. Papoutsellis, [a] H. Dong, [a] R. Docherty, [b] A. M. Beale, [acd] S. J. Cooper [b] and S. D. M. Jacques [a]

Received 5th June 2025
Accepted 6th August 2025

DOI: 10.1039/d5dd00252d

rsc.li/digitaldiscovery

nDTomo is a Python-based software suite for the simulation, reconstruction and analysis of X-ray chemical imaging and computed tomography data. It provides a collection of Python function-based tools designed for accessibility and education as well as a graphical user interface. Prioritising transparency and ease of learning, *nDTomo* adopts a function-centric design that facilitates straightforward understanding and extension of core workflows, from phantom generation and pencil-beam tomography simulation to sinogram correction, tomographic reconstruction and peak fitting. While many scientific toolkits embrace object-oriented design for modularity and scalability, *nDTomo* instead emphasises pedagogical clarity, making it especially suitable for students and researchers entering the chemical imaging and tomography field. The suite also includes modern deep learning tools, such as a self-supervised neural network for peak analysis (PeakFitCNN) and a GPU-based direct least squares reconstruction (DLSR) approach for simultaneous tomographic reconstruction and parameter estimation. Rather than aiming to replace established tomography frameworks, *nDTomo* serves as an open, function-oriented environment for training, prototyping, and research in chemical imaging and tomography.

1 Introduction

X-ray chemical imaging and computed tomography techniques, such as X-ray powder diffraction computed tomography (XRD-CT), X-ray fluorescence computed tomography (XRF-CT), X-ray absorption near-edge structure spectroscopy computed tomography (XANES-CT) and X-ray pair distribution function computed tomography (PDF-CT), provide spatially-resolved structural and/or compositional information from complex, often heterogeneous samples[1–3] (see Fig. 1). These techniques are increasingly being applied in fields such as catalysis, battery research, and advanced materials development, where understanding the structure–function relationships in 2D or 3D under realistic operating conditions is crucial.

Although these imaging techniques are gaining increasing attention and adoption, the tools and workflows used to process the resulting data remain scattered and technically complex. Researchers often rely on custom scripts, beamline-specific software, or general-purpose image processing libraries to perform essential tasks such as sinogram correction, artefact removal and tomographic reconstruction.[4–7] This patchwork approach creates a high barrier to entry, particularly for students and early career scientists, and hinders reproducibility and broader adoption across disciplines.

nDTomo was developed to address these challenges through a pedagogically grounded Python-based framework for chemical imaging and tomography. Originally created during a post-doctoral project at the European Synchrotron Radiation Facility (ESRF), the code began as a set of GUI tools for handling XRD-CT datasets acquired at beamline ID15A.[8] Following the ESRF's extended shutdown period for the Extremely Brilliant Source (EBS) upgrade, which involved replacing the storage ring to create the world's first high-energy fourth-generation synchrotron, and the corresponding overhaul of their data acquisition and processing systems, the original codebase became obsolete. From 2019 the development of *nDTomo* resumed at Finden Ltd,[9] partially funded through European Union's Horizon Europe Research project STORMING,[10] with the goal of transforming the project into a more general-purpose open-source toolkit. Over time, particularly through focused development during 2024–2025, *nDTomo* evolved into its current form: a modular, function-based software suite with a GUI and a suite of educational notebooks.

While several open-source tools exist for hyperspectral data analysis, such as HyperSpy,[12] PyMca,[13] DAWN,[14] MANTiS,[15] and HyperGUI,[16] these primarily focus on spectral data exploration, visualisation and peak analysis. In contrast, *nDTomo* offers a more comprehensive environment that includes phantom

[a]*Finden Ltd, Building R71, Rutherford Appleton Laboratory, Harwell Science and Innovation Campus, Oxfordshire, OX11 0QX, UK. E-mail: antony@finden.co.uk*

[b]*Dyson School of Design Engineering, Imperial College London, London, SW7 2DB, UK. E-mail: a.vamvakeros@imperial.ac.uk*

[c]*Department of Chemistry, University College London, 20 Gordon Street, WC1H 0AJ, UK*

[d]*Research Complex at Harwell, Rutherford Appleton Laboratory, Harwell Science and Innovation Campus, OX11 0FA, UK*

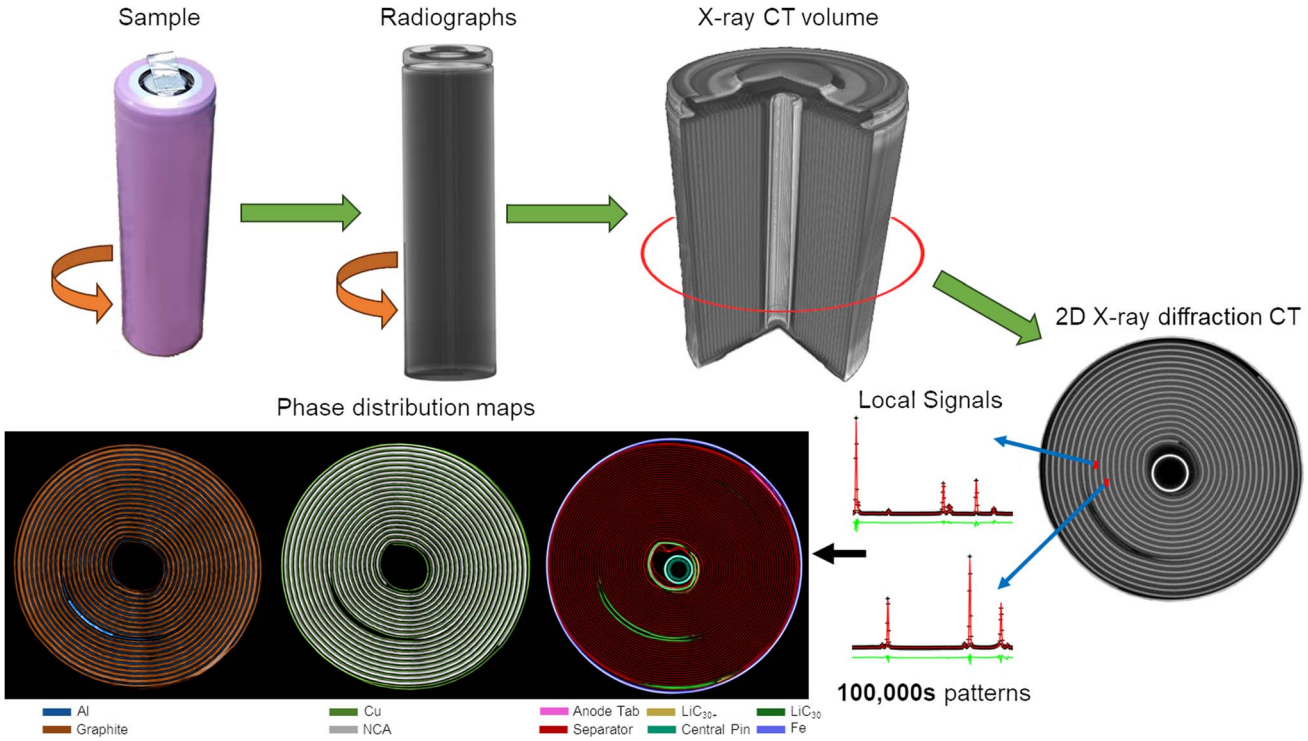

Fig. 1 Comparison between grayscale X-ray absorption-contrast CT and colour X-ray powder diffraction CT (XRD-CT) data acquired from a Li ion battery. For more details regarding this study see ref. 11.

simulation, sinogram preprocessing, tomographic reconstruction and spectral analysis. It includes functionality for multi-dimensional phantom generation, simulation of different pencil-beam computed tomography acquisition strategies, sinogram data correction methods, analytical and iterative tomographic imaging reconstruction methods, dimensionality reduction and peak fitting, as well as advanced deep learning approaches for peak fitting of chemical imaging and tomography data, such as the self-supervised PeakFitCNN and GPU-accelerated DLSR.[17]

nDTomo is not intended to replace specialised CT libraries such as ASTRA Toolbox,[18] CIL,[19,20] TomoPy,[21] Tomosipo,[22] Toupy[23] or TIGRE,[24] which are widely used for high-performance tomographic reconstruction. In fact, *nDTomo* includes a dedicated module that wraps selected ASTRA methods, simplifying their use within Python workflows. However, it should be noted that *nDTomo* also provides sinogram-based correction functions, including air signal subtraction, intensity normalisation due to beam decay, motor jitter artefact correction and centre-of-rotation estimation, which are essential for processing real experimental datasets, yet often missing or only partially addressed in CT libraries, with a notable exception being Algotom.[7]

What sets *nDTomo* apart is its emphasis on education and ease of use. Many researchers, particularly those in experimental domains, often use CT-based techniques in their work but lack the computational background to engage deeply with the highly abstracted APIs found in advanced CT reconstruction toolkits. *nDTomo* addresses this gap by offering transparent, well-documented code that prioritises clarity over compactness. Most modules are built around standalone, functional implementations that operate directly on NumPy arrays[25] or Torch tensors,[26] making them easy to inspect, modify, and reuse.

The software is modality-agnostic but is particularly well suited to diffraction and spectroscopic tomography. It is accompanied by example Jupyter notebooks that serve as both tutorials and reproducible analysis workflows. These examples cover the entire pipeline, from simulation of multi-dimensional phantoms and CT data acquisition strategies to reconstruction, unsupervised analysis and neural network-based peak fitting. Whether used as a teaching tool, a prototyping platform, or a reference implementation, *nDTomo* aims to make chemical imaging and tomography more transparent, reproducible and accessible.

2 Software overview

The software serves three complementary purposes:

- As a pedagogical platform for students and early career scientists.
- As a flexible research tool that can be used for prototyping and testing new ideas, such as simulating phantom objects and data acquisition strategies.
- Processing and analysis of experimental data.

nDTomo is structured as a lightweight, modular Python package that combines a set of standalone function-based modules with a graphical user interface (GUI). Its design emphasises straightforward usability and low entry barriers,

making it suitable both for researchers who need a practical analysis toolkit and for newcomers learning the fundamentals of chemical imaging and tomography.

2.1 Modular architecture

The codebase is organised into the following submodules (see Fig. 2):

- *nDTomo*.gui—source code for the PyQt-based graphical interface.
- *nDTomo*.sim—synthetic phantom generation and pencil-beam acquisition simulation.
- *nDTomo*.methods—general-purpose utilities varying from matrix size manipulation to *Cartesian* ↔ *Polar* coordinate transformations.
- *nDTomo*.tomo—sinogram correction, tomographic reconstruction methods including filtered backprojection (FBP) and iterative methods such as SIRT and CGLS.[27–29]
- *nDTomo*.analysis—peak shape models for data fitting.
- *nDTomo*.pytorch—machine learning code, including PeakFitCNN and DLSR, built on PyTorch.

Each module is intentionally built using functional programming principles rather than object-oriented inheritance. Most operations are carried out by calling a single function on NumPy arrays or Torch tensors, with clearly documented input and output arguments. This lowers the barrier for those unfamiliar with complex programming paradigms and facilitates inspection, debugging and modification of each step in the pipeline.

2.2 File formats and dependencies

All data is handled using standard open formats. Hyperspectral volumes are stored as .h5/.hdf5 files, with associated spectral axes saved under conventional group names such as/tth,/q, or/energy. These correspond to commonly used representations of the diffraction axis:/tth refers to 2θ, the scattering angle in degrees;/q represents the momentum transfer in reciprocal space (typically in $Å^{-1}$); and/energy denotes the incident or scattered X-ray energy (usually in keV). Exported results—such as image slices, fitted spectra, and peak parameter maps—can be saved in .h5, .asc, .xy, or .png formats.

Internally, *nDTomo* is built on a foundation of well-established scientific Python libraries. Numerical operations are handled using NumPy[25] and SciPy,[30] while image processing tasks make use of scikit-image.[31] For dimensionality reduction and other machine learning utilities, scikit-learn[32] is employed. Plotting and data visualization are performed with matplotlib,[33] and GPU-accelerated deep learning models are implemented using PyTorch.[26] These dependencies ensure both computational efficiency and ease of integration with the broader scientific Python ecosystem.

2.3 Installation and licensing

The software is compatible with Linux, macOS, and Windows. Although *nDTomo* is installed *via* pip, it depends on external libraries such as the ASTRA Toolbox and PyTorch, which may require additional configuration, particularly for GPU support. We recommend using a Conda environment to simplify the installation of core dependencies like ASTRA, while the main *nDTomo* package can be installed directly from PyPI or GitHub. The code is released under the GNU General Public License (GPLv3) and is freely available *via* GitHub, with packages hosted on PyPI and archived releases published on Zenodo.

3 Graphical user interface

To make chemical imaging analysis more accessible, *nDTomo* includes a standalone GUI built with PyQt5.[34] The GUI provides a structured, tabbed environment designed to support the entire chemical tomography analysis workflow: from the initial visualisation of raw hyperspectral datasets to tasks like extraction of region-of-interest (ROI) images or local signals, image

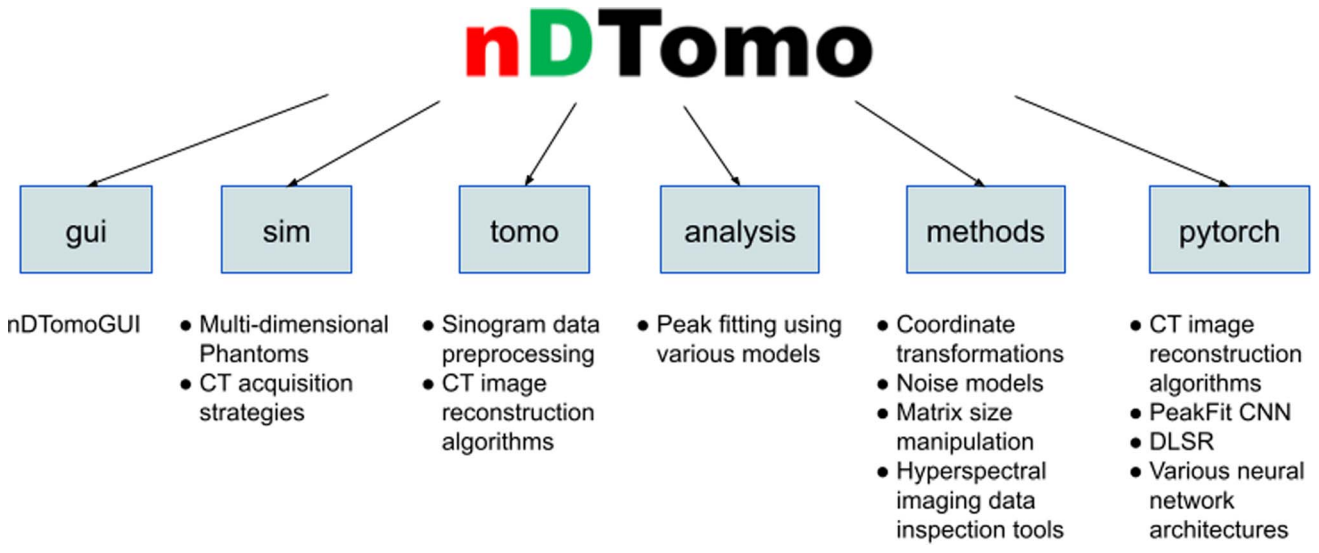

Fig. 2 Overview of nDTomo module structure and contents.

segmentation and peak fitting. It enables non-programmers and early-career scientists to interact with complex X-ray chemical imaging data without writing code, making it an ideal tool for education, training, as well as rapid exploration of experimental data, both during experiments to support real-time decision-making and post-experiment to aid in understanding the underlying chemistry and material properties of the sample. The GUI accepts .h5 or .hdf5 files containing 2D hyperspectral images of shape (X, Y, channels) and when present, spectral axes stored under/tth,/q,/d, or/energy. A dataset is loaded using a simple file dialog, and the interface automatically initializes spatial and spectral views.

3.1 Main tabs and workflow

The GUI is currently (v2025.05) organised into four core tabs, each representing a distinct stage in the hyperspectral imaging analysis pipeline:

1. Hyperexplorer: this entry tab allows users to interactively explore the spatial and spectral dimensions of chemical imaging data (see Fig. 3). The left panel displays a 2D spectral image while the right panel shows the local spectrum from a hovered pixel. The views are fully linked: hovering over the image updates the spectrum, and hovering over the spectrum shows the corresponding spectral slice as an image. Users can zoom, pan, and export both spectra and images. Additional controls allow colormap changes and export to .png, .h5, .asc, or .xy.

2. ROI image: in this tab, users define a spectral channel range and create a 2D image by summing across it. Two optional background subtraction strategies are available: mean subtraction and voxel-wise linear subtraction. The image is normalised before it is being transferred to the next tab. The resulting region-of-interest (ROI) image can be visualised, modified, and exported.

3. ROI pattern: this tab allows users to refine the ROI by thresholding the previously generated ROI image. The resulting binary mask can be applied to the dataset to extract a spatially averaged spectrum, which is then normalised. Furthermore, an automatic peak suggestion tool is provided (*via* Scipy[30]) with the detected peak positions being overlaid on the plot. The extracted spectrum and segmentation mask can be exported in multiple formats.

4. Peak fitting: users can batch-fit a single peak across the dataset using one of three models: Gaussian, Lorentzian, or Pseudo-Voigt. The user selects the fitting range and specifies initial guesses and parameter bounds for peak area, position, full width at half maximum (FWHM), and if using the Pseudo-Voigt profile the mixing ratio. A live progress bar displays fitting progress and fitted parameters (*e.g.*, intensity, position, width) can be visualised in real time. Results are saved as HDF5 datasets. Once the fitting process is completed, parameter maps (*e.g.*, intensity, position, FWHM) can be visualised or exported, and a diagnostic mode allows visual inspection of fits and residuals while hovering the mouse over the image on the left panel.

3.2 Additional features

Two key tools are available under the GUI's Advanced menu:

- Synthetic Phantom Generator: this feature creates a hyperspectral dataset on the fly, composed of five known

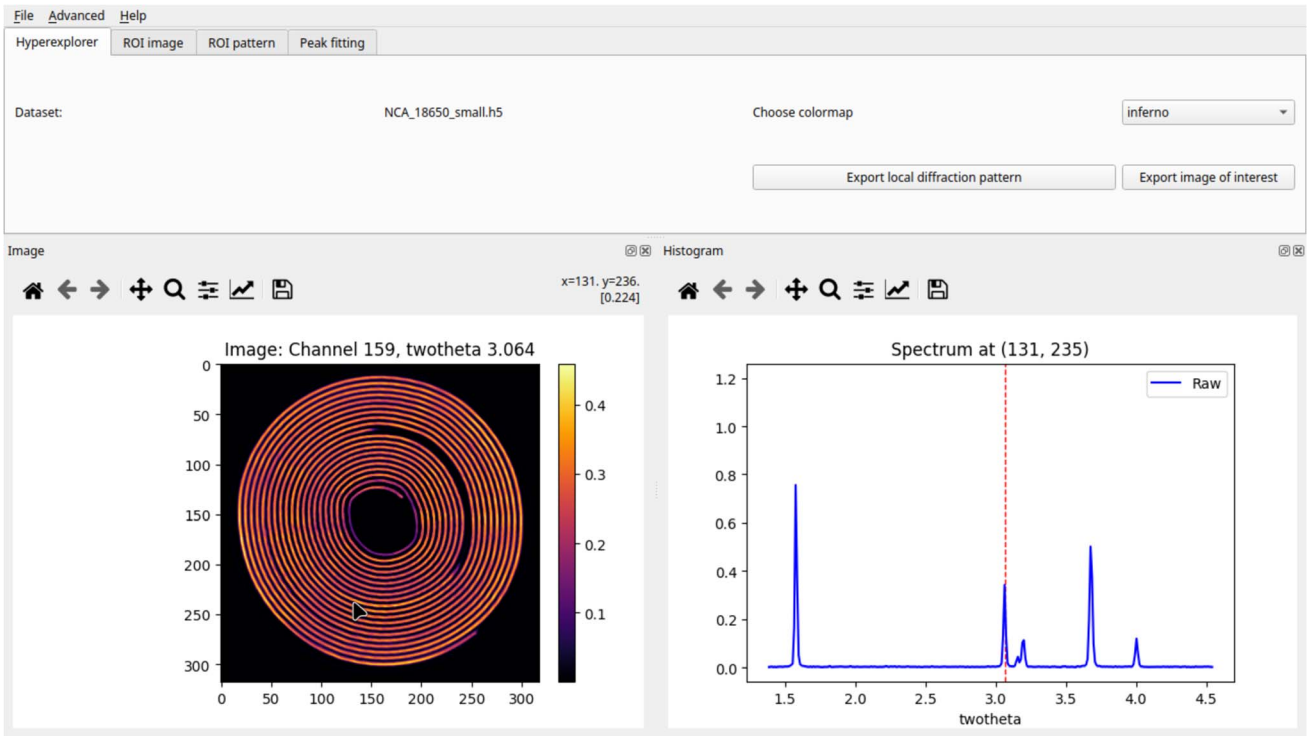

Fig. 3 Visualising an XRD-CT dataset of a Li-ion battery with nDTomoGUI. For more details see ref. 11.

diffraction patterns (Al, Cu, Fe, Pt, Zn) distributed across five corresponding predefined phantom images. The generated dataset is loaded automatically and can be used for testing, benchmarking, or training without requiring experimental data.

• Embedded IPython Console: for advanced users, an embedded IPython console provides direct access to all internal session variables (*e.g.* volume, spectrum, image, *x*-axis). This allows on-the-fly scripting, debugging, custom visualisation and manual data export within the live GUI session.

The *nDTomoGUI* is particularly valuable for experimentalists unfamiliar with Python programming, offering an intuitive route to analyse chemical imaging and tomography data. In academic and training environments, it serves as a hands-on tool to teach core concepts such as hyperspectral imaging datasets, simple threshold-based image segmentation and model-based peak analysis.

4 Core modules and workflows

Each functional component of *nDTomo* is organised into a self-contained module. In this section, we describe the purpose of each module, highlight representative functions and illustrate their use through a corresponding tutorial notebook. Together, these modules cover the entire pipeline from simulation to machine learning-based analysis.

4.1 *nDTomo*.sim – simulation of phantoms and CT acquisition strategies

The *nDTomo*.sim module provides tools for generating synthetic datasets and simulating X-ray pencil-beam computed tomography (CT) acquisition schemes. These functions are useful for benchmarking, training machine learning models, testing reconstruction algorithms and teaching basic principles of tomographic imaging.

Key capabilities. • Creation of 2D and 3D geometric phantoms using primitives such as circles, rectangles, cubes, spheres, cylinders and Voronoi diagrams (see Fig. 4).

• Construction of multi-dimensional phantoms (*i.e.* 3D–5D) by combining reference spectra with phase distribution maps.

• Simulation of various pencil-beam CT acquisition strategies, including zigzag[35] and continuous rotation-translation approaches.[36]

Accompanying notebooks demonstrate how to generate multi-dimensional phantoms and hyperspectral imaging data, as well as how to simulate pencil-beam CT data acquisition strategies including zigzag and zigzig scans using either the rotation or the translation as the fast axis,[35] as well as the continuous rotation–translation approach.[36]

4.2 *nDTomo*.methods – utilities and supporting tools

The *nDTomo*.methods module provides a collection of utility functions that include hypespectral imaging data visualisation,

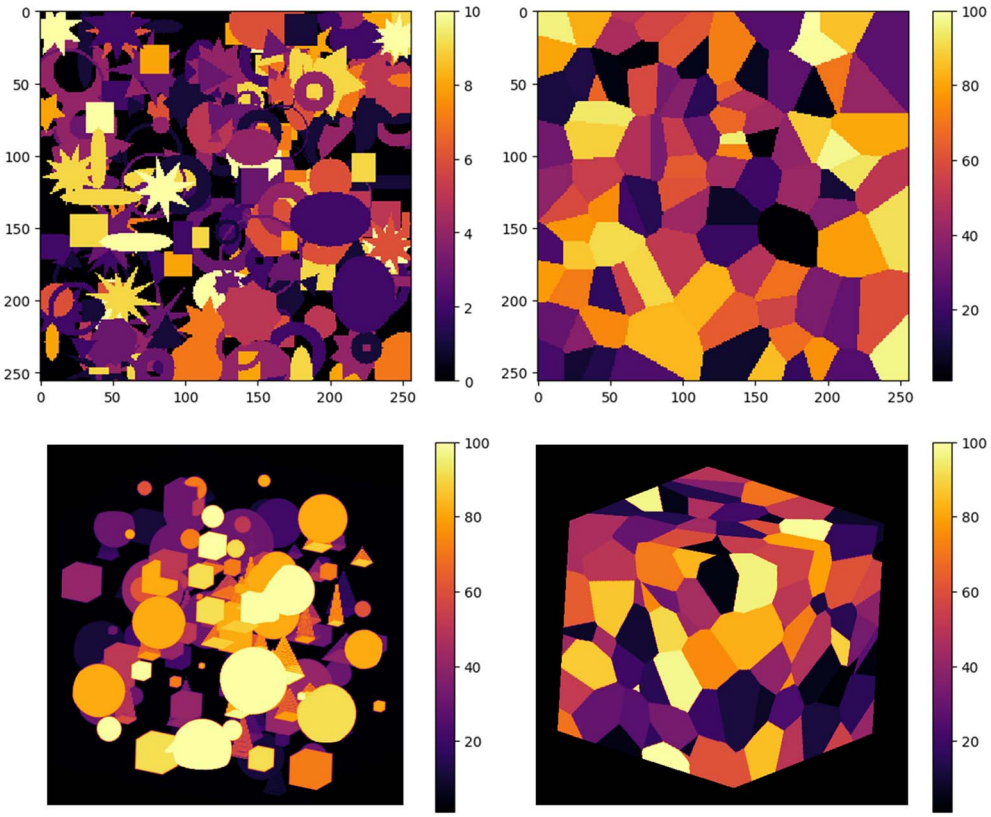

Fig. 4 Visualising 2D (256 × 256 pixels) and 3D geometric (256 × 256 × 256 voxels) phantoms generated with nDTomo (left column) as well as 2D and 3D Voronoi tessellations (right column).

coordinate transformations, noise simulation and various simpler general-purpose operations used throughout the *nDTomo* framework. These methods are not bound to a specific workflow but are frequently used across various modules and tutorial notebooks.

Key capabilities. • Conversion between physical quantities, such as 2θ, q, d-spacing, and photon energy.

• Coordinate transformations (*Cartesian* ↔ *Polar*).

• Interpolation tools for rebinning datasets.

• Simulation of synchrotron beam decay and Poisson noise for hyperspectral sinogram data.

• Matrix size manipulation functions *e.g.* even sizing, padding, cropping of tomographic data.

This module also includes lightweight tools for hyperspectral imaging data inspection, including hyperexplorer and related functions, which allow linked image-spectrum visualisation. These tools can be used either inside Jupyter notebooks or embedded within standalone scripts to enable interactive inspection of raw or fitted hyperspectral volumes. They provide a simplified alternative to the full GUI for users working within notebook environments, making them particularly suitable for early-stage data exploration, debugging, or teaching.

4.3 *nDTomo*.tomo – sinogram handling and tomographic reconstruction

The *nDTomo*.tomo module provides core functionality for sinogram construction, correction, and tomographic image reconstruction. It includes both analytical and iterative methods, including routines that simulate forward and backward projections. In addition to these pedagogical implementations, nDTomo also provides a wrapper module (nDTomo.tomo.astra_tomo) that simplifies access to ASTRA's algorithms. These functions preserve ASTRA's full performance while reducing boilerplate scripting and integrating seamlessly into the broader nDTomo workflow. This module is central to the processing of pencil-beam and parallel-beam tomography data and is designed to make each reconstruction step accessible and transparent for educational use.

Analytical methods such as FBP are computationally efficient and generally sufficient for well-sampled, low-noise datasets. Iterative methods such as SIRT and CGLS are useful when greater control over the forward and backprojection process is needed, for example when implementing non-standard geometries or exploring regularised extensions. However, without regularisation (*e.g.* Tikhonov or total variation), these iterative methods do not consistently yield better reconstructions than FBP and may amplify noise. In nDTomo, they are included primarily for pedagogical purposes and for prototyping new reconstruction strategies.

Key capabilities. • Construction and stacking of 2D sinograms from hyperspectral data.

• Sinogram preprocessing methods, including:

– airrem: background (air signal) removal using top/bottom row sampling.

– scalesinos: normalisation of projection intensity to account for beam decay.

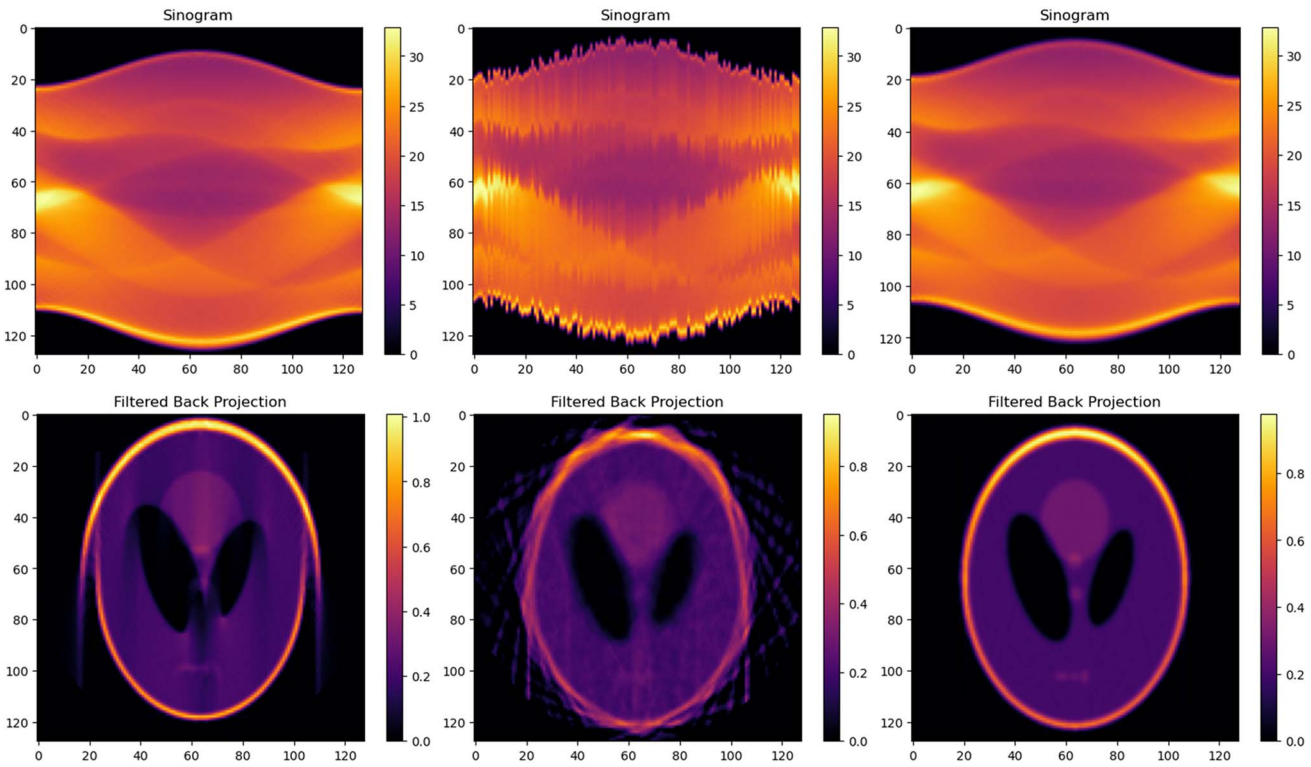

Fig. 5 Top left: Sinogram with centre of rotation offset, top middle: sinogram with motor jitter, top right: corrected sinogram for both centre of rotation offset and motor jitter, bottom row: corresponding FBP reconstructions.

– sinocomcor: sinogram centre-of-mass correction to correct mottor jitter.

– sinocentering: automatic detection and correction of centre-of-rotation offsets.

• Analytical reconstruction using the FBP algorithm implemented using numpy and scipy.

• Iterative reconstruction methods such as SIRT and CGLS, including both (sparse) matrix-based and matrix-free (functional) implementations in numpy and pytorch.

• Forward projection tools for simulating sinograms from 2D/3D volumes.

• Wrapper functions for ASTRA Toolbox enabling high-performance GPU-based reconstruction with minimal setup.

One tutorial notebook demonstrates the construction and correction of sinograms from a simulated Shepp–Logan phantom, highlighting the effects of background removal, normalisation and misalignment correction on reconstruction quality (see Fig. 5). Another notebook focuses on reconstruction methods, including FBP and iterative algorithms (SIRT, CGLS), with side-by-side comparisons of matrix-based and matrix-free implementations. These examples help users understand both the theory and practical implications of different reconstruction strategies.

This module is particularly valuable for users who wish to understand the fundamentals of sinogram formation and tomographic image reconstruction. The functional implementations are designed for pedagogical clarity and several routines are compatible with GPU acceleration when used with Pytorch.

4.4 *nDTomo*.analysis – peak models and spectral exploration

The *nDTomo*.analysis module contains a set of parametric peak models that serve as the foundation for peak fitting in *nDTomo*. These include simple implementations of Gaussian, Lorentzian and Pseudo-Voigt functions. Additional functions support multi-peak combinations with linear or quadratic background models, least-squares residual calculation, and HDF5-based import/export of fitting results. Although this module is minimal by design and not intended as a comprehensive fitting framework, it underpins several key workflows in *nDTomo* and is featured in tutorial notebooks that explore its practical applications.

One tutorial focuses on unsupervised learning for spectral unmixing (see Fig. 6). It compares the performance of *K*-means, Principal Component Analysis (PCA), and Non-negative Matrix Factorisation (NMF) on a synthetic XRD-CT dataset. Through direct visual and quantitative comparison, it highlights the limitations of commonly used methods like PCA and *K*-

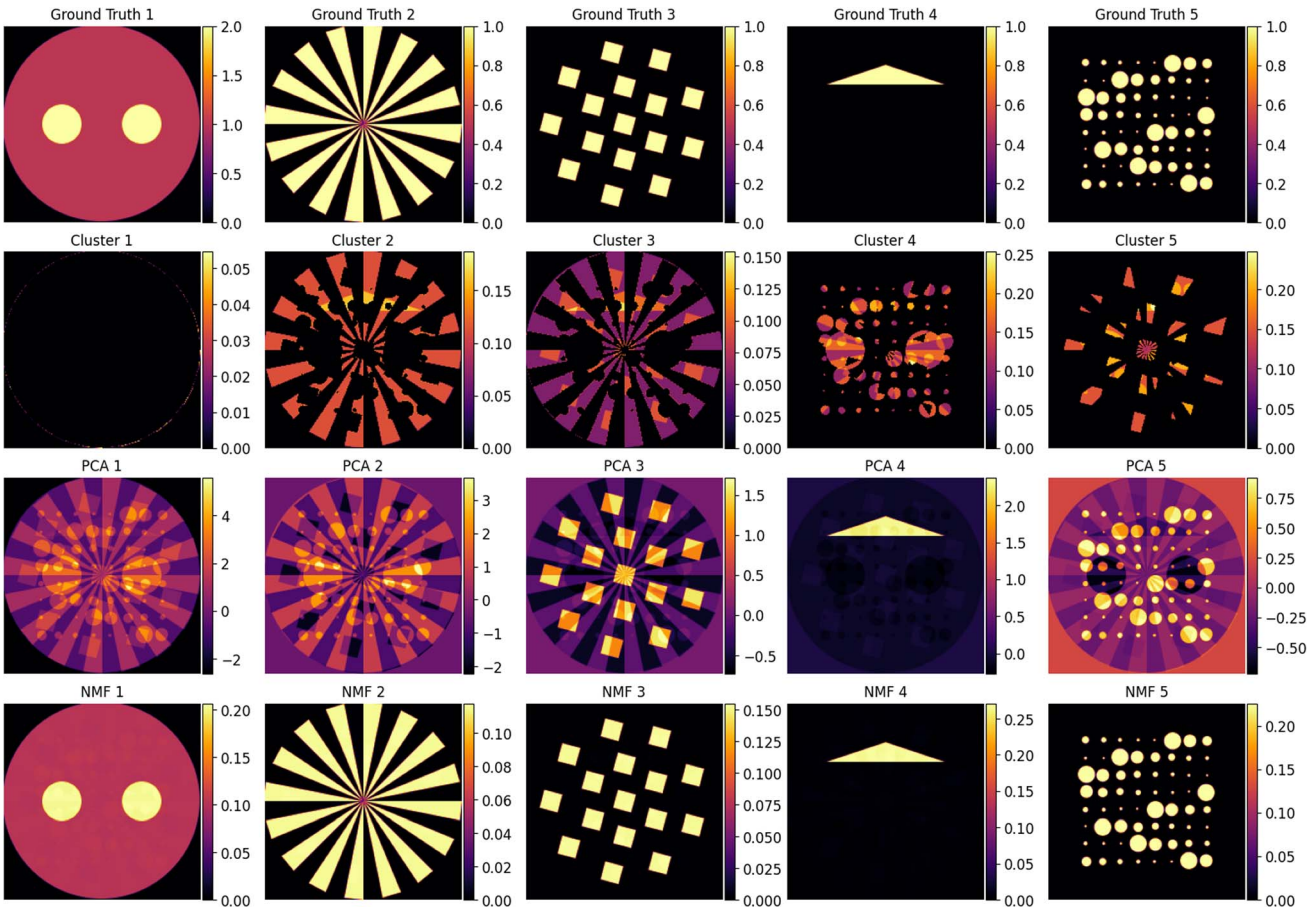

Fig. 6 First row: Ground truth maps used for the hyperspectral phantom, second row: *K*-means cluster maps with mean intensity per pixel, third row: PCA component maps, fourth row: NMF maps.

means—particularly their inability to resolve meaningful chemical components and demonstrates the superior interpretability of NMF when applied in the image domain. The notebook offers guidance on when and how to use these methods and serves as a cautionary note against relying solely on variance-based techniques for chemical analysis.

Another notebook walks through a complete single-peak fitting workflow, from ROI selection and background subtraction to pixel-wise fitting using Gaussian models. Despite using idealised synthetic data, it exposes practical challenges such as sensitivity to initial parameters, background modelling errors, and performance bottlenecks in CPU-based fitting. The tutorial concludes with a discussion on the need for GPU-accelerated approaches to handle the increasing size and complexity of hyperspectral imaging datasets, motivating the development of tools like PeakFitCNN.

4.5 *nDTomo*.pytorch – deep learning models and GPU acceleration

The *nDTomo*.pytorch module provides GPU-accelerated tools for deep learning workflows in chemical imaging and tomography. It includes convolutional neural network (CNN) architectures, differentiable tomographic operators, trainable peak models, and loss functions designed for inverse problems, spectral reconstruction, and peak parameter inference.

Key capabilities. • Differentiable 2D/3D forward and back-projectors implemented in PyTorch.

• Iterative reconstruction solvers: SIRT and CGLS using functional and sparse-matrix operators.

• CNN-based architectures for spectral and spatial tasks: CNN1D, CNN2D/3D, ResNet,[37] U-net[38] and hybrid parameterized models.

• Trainable peak function models (*e.g.*, Gaussian, Pseudo-Voigt) using normalized physical constraints.

• Total variation (TV) regularization and SSIM[39] loss for 2D and 3D imaging tasks.

• Sobol and Gaussian patch sampling for training efficiency and coverage control.[40]

One of the accompanying notebooks introduces PeakFitCNN, a lightweight self-supervised convolutional neural network designed for pixel-wise spectral peak fitting in chemical imaging data. Both the network architecture and training strategy are illustrated in Fig. 7.

The input to PeakFitCNN is a spatially downsampled version of the experimental hyperspectral image, reduced by a factor of 4 along both spatial dimensions (*i.e.* input shape: $(n_{\mathrm{pix}_x}/4, n_{\mathrm{pix}_y}/4, n_{\mathrm{ch}})$). This downsampled data cube is passed through a sequence of three main convolutional blocks, each comprising 2D convolutional layers, instance normalization, and ReLU activations. The final block includes an upsampling layer to restore the spatial resolution to its original scale *via* a 4× upscaling. A final sigmoid activation ensures that the output maps are bounded between 0 and 1, enabling them to represent normalised peak parameters.

Rather than producing hyperspectral data directly, the network outputs parameter maps corresponding to the selected spectral profile model. For example, if a single Gaussian peak is assumed, the network produces three output channels representing the area, peak position, and FWHM. These are stored as a tensor of shape $(n_{\mathrm{pix}_x}, n_{\mathrm{pix}_y}, n_{\mathrm{prm}})$, where n_{prm} is the number of parameters in the peak model. These maps are then denormalised using user-defined min/max ranges to recover physical parameter values.

To avoid memory bottlenecks and to speed up training, we do not reconstruct and compare the full hyperspectral image at each iteration. Instead, we use precomputed spatial indices to extract small 2D patches from both the experimental and model-predicted hyperspectral images. These predicted

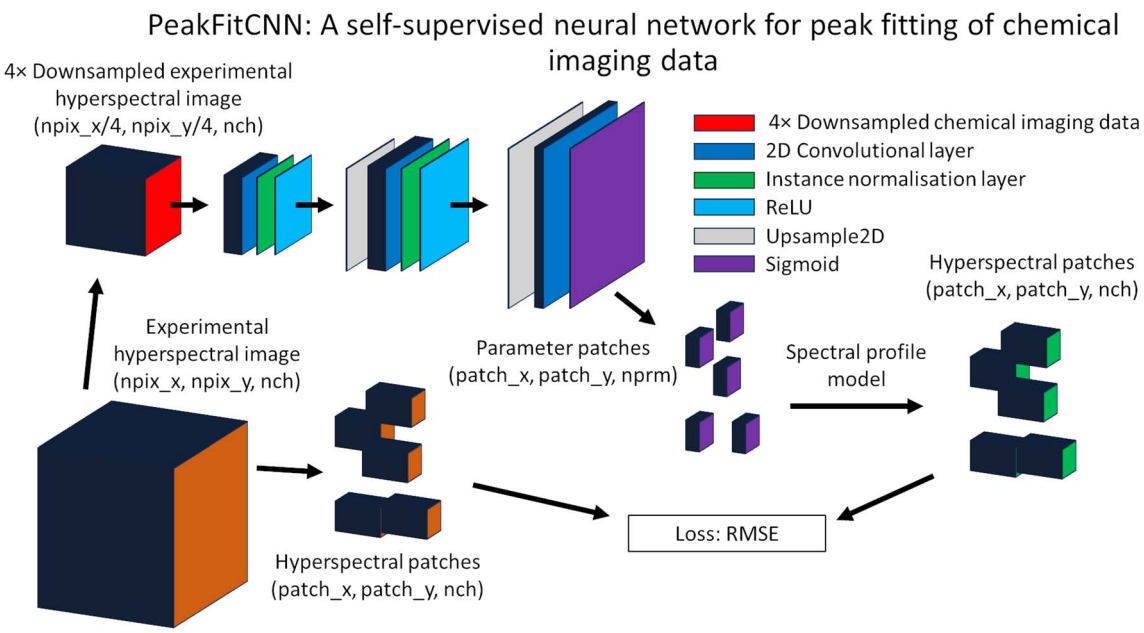

Fig. 7 Illustration of the PeakFitCNN architecture and the training loop which leads to the peak fitting of the chemical imaging data.

hyperspectral patches are generated by applying the spectral model to the parameter maps, and compared directly to the corresponding ground truth patches from the experimental data.

The network is trained by minimizing the root mean squared error (RMSE) between the predicted and actual hyperspectral patches. This patch-wise comparison allows for scalable training, even on large datasets, while maintaining the fidelity of the peak fitting operation. As training progresses, the network learns to generate spatially resolved maps of the peak parameters, achieving high accuracy and interpretability. At inference time, PeakFitCNN enables full-resolution estimation of chemically meaningful properties such as peak area, position, and width. Instead of relying on labelled parameter maps, the PeakFitCNN is trained in a self-supervised manner learning to produce relatively smooth with suppressed noise outputs directly from downsampled hyperspectral volumes. This self-supervised strategy avoids the need for synthetic or labelled datasets and instead learns directly from experimental spectra by optimizing the fit between predicted and observed profiles.

The key advantage of using PeakFitCNN emerges in scenarios involving noisy experimental chemical imaging data. As demonstrated in Fig. 8, the network yields noticeably improved results on a simulated phantom dataset corrupted with Poisson noise. Across all Gaussian peak parameter maps—area, position, and FWHM, the outputs from PeakFitCNN are consistently smoother while preserving sharp edges and they more closely resemble the ground truth parameter maps shown for comparison. While the improvements over conventional pixel-wise peak fitting are not drastic, they are clear and consistent, highlighting the model's robustness to noise and its suitability for practical applications in high-throughput settings. Quantitative results are presented in Section S1 of the ESI, demonstrating that PeakFitCNN produces parameter maps (peak area, position, FWHM, slope, and intercept) with lower mean absolute error (MAE), mean squared error (MSE), RMSE, as well as higher peak signal-to-noise ratio (PSNR) and structural similarity index (SSIM), compared to the prms-only approach. All metrics are calculated with respect to the ground truth parameter maps.

Another notebook presents the DLSR approach, which jointly fits peak models and reconstructs chemical tomography data.[17] By integrating a forward projection model into the optimisation loop, DLSR offers an alternative to the standard CT reconstruction followed by peak-fitting pipeline. This approach is computationally slow and requires significant resources in terms of RAM but it has some advantages when projections are sparse, reducing artefacts and stabilising parameter estimation. This tutorial provides a modular benchmarking framework to explore trade-offs between speed, interpretability, and reconstruction quality in low-data regimes.

The tutorial implements two versions of the DLSR workflow: a parameter-only approach and a PeakFitCNN-based approach. In both cases, the model reconstructs a set of peak parameter maps; for example, peak area, position, FWHM, and two background coefficients for a Gaussian peak with a linear background model. These maps are used to generate synthetic hyperspectral volumes (*via* differentiable peak functions),

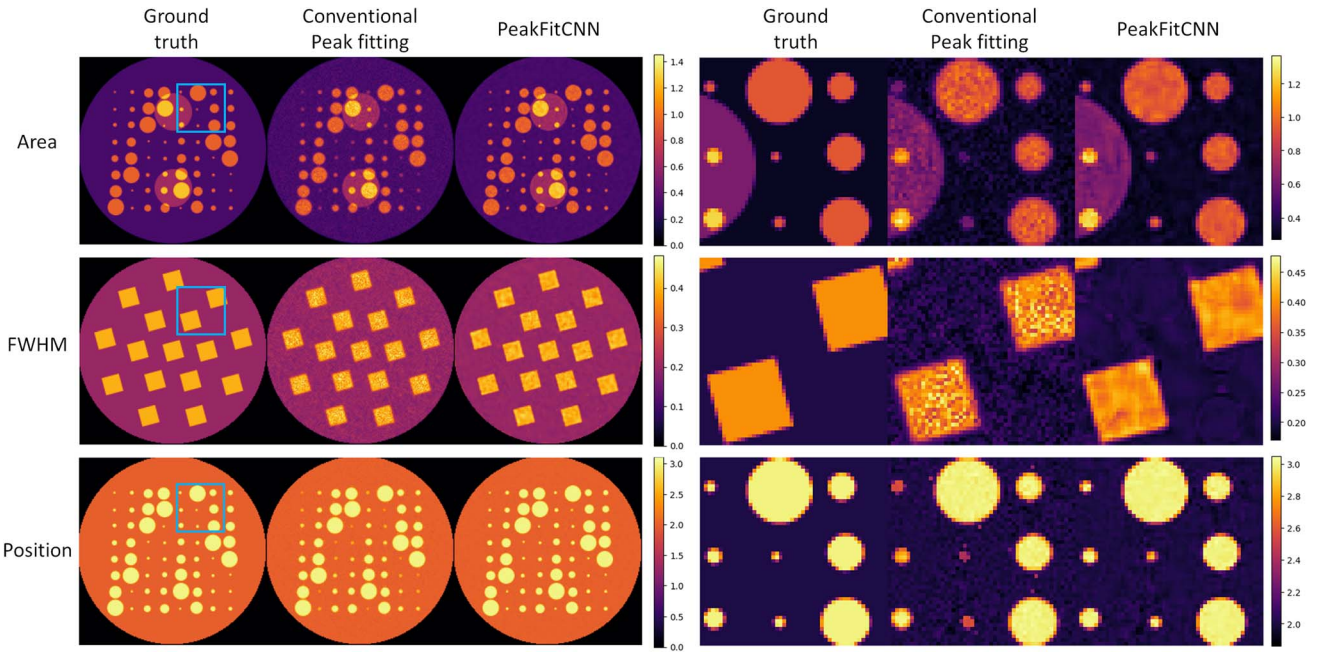

Fig. 8 Figures on the left hand side correspond to the full maps while figures on the right hand side correspond to the ROIs indicated by the cyan squares. The three maps from left to right correspond to the following: ground truth, conventional peak fitting of the chemical imaging data after adding Poisson noise and results obtained with the PeakFitCNN for the same dataset. Top row: Maps corresponding to the area of the Gaussian peak, middle row: maps corresponding to the FWHM of the Gaussian peak, bottom row: maps corresponding to the position of the Gaussian peak.

which are then forward projected and compared to the experimental sinogram data. This enables self-supervised parameter estimation (*i.e.* real space model parameter maps) using directly the sinogram data and without requiring labelled ground truth. The notebook demonstrates that DLSR combined with PeakFitCNN provides improved performance compared to the parameter-only DLSR and conventional FBP followed by real-space peak fitting, especially when the sinograms suffer from angular undersampling. Quantitative results are presented in Section S2 of the ESI, demonstrating that PeakFitCNN produces parameter maps (peak area, position, FWHM, slope, and intercept) with lower MAE, MSE, RMSE, as well as higher PSNR and SSIM, compared to both the DLSR-prms and the FBP-prms approaches. All metrics are calculated with respect to the ground truth parameter maps.

These notebooks showcase how GPU-based learning frameworks can complement traditional approaches, offering flexible strategies to tackle the growing data and complexity challenges in modern chemical tomography.

5 Applications and educational use

nDTomo has been developed and refined through its application in real-world research projects involving X-ray diffraction computed tomography (XRD-CT) of heterogeneous functional materials and devices such as Li ion batteries,[11,41–43] catalysts[44–51] and fuel cells.[52,53] These studies often examined chemical systems under *operando* or *in situ* conditions. During these projects, the core *nDTomo* codebase was used for diffraction pattern integration (*via* PyFAI[54]), sinogram-level corrections and tomographic reconstruction of chemical images. In most cases, tomographic reconstruction was performed using the nDTomo.tomo.astra_tomo wrapper, which simplifies access to ASTRA's GPU-accelerated CT reconstruction algorithms. The use of nDTomoGUI enabled threshold-based masking and export of local diffraction patterns of interest, which accelerated the phase identification process. All steps from raw sinogram construction to final chemical image reconstruction and pattern extraction could be performed within one Python environment, allowing researchers to handle hyperspectral tomographic datasets efficiently before completing structural refinement in external tools such as Rietveld analysis using TOPAS.[55] Variants of the DLSR framework were also tested in selected cases to investigate improved reconstruction strategies under angular undersampling, particularly for large samples such as Li-ion batteries and PEM fuel cells, although these early implementations were developed outside of the nDTomo framework.[17,41,56]

Beyond its use in research, components of *nDTomo* have been incorporated into two chemical imaging training courses hosted by Finden Ltd, introducing early career scientists, including PhD students and postdoctoral researchers, to key concepts such as X-ray computed tomography, hyperspectral imaging and the tools to handle and analyse such data. The combination of interactive tutorials and the GUI provided a hands-on entry point for exploring these topics in practice.

A major stable release of *nDTomo* (v2025.05) was recently released, featuring extended documentation and ten modular tutorial notebooks. These notebooks are designed to support both self-guided learning and classroom teaching, and cover a range of topics including phantom generation, sinogram simulation and correction, tomographic reconstruction, chemical imaging data dimensionality reduction and peak fitting. Each notebook is structured to provide intuitive step-by-step guidance with embedded code examples and visual outputs that make the underlying algorithms and workflows transparent and accessible.

Although originally focused on XRD-CT, the modular design of *nDTomo* makes it readily adaptable to other chemical imaging modalities such as XRF-CT, XANES-CT and extended X-ray absorption fine structure spectroscopy computed tomography (EXAFS-CT), as well as infrared (IR) and Raman imaging. IR and Raman imaging share key characteristics with XRF and XRD data, such as the presence of a background signal that can often be modelled with a high-degree polynomial, and characteristic spectral peaks that are typically fitted with Gaussian or other peak profiles. As long as the user converts the raw data into a 3D NumPy array (spatial $\times$ spatial $\times$ spectral), they can access the full range of tools available in *nDTomo*, from exploratory analysis with the GUI to advanced workflows such as PeakFitCNN for self-supervised peak fitting with GPU acceleration. *nDTomo*'s combination of graphical and scripting interfaces provides flexibility for both exploratory visualisation and reproducible, scriptable workflows. The modular design of *nDTomo* also facilitates extension to new or emerging modalities involving spectral and tomographic imaging. For example, *nDTomo* can support workflows based on next-generation hyperspectral detectors such as the STFC-developed HEXITEC[57,58] or DECTRIS[59,60] detectors, which are gaining traction in both scientific and medical imaging contexts.

6 Conclusion and future work

nDTomo is a Python-based software suite for the simulation, reconstruction and analysis of X-ray chemical imaging and computed tomography data. Developed through direct engagement with real-world imaging challenges, particularly in X-ray powder diffraction computed tomography (XRD-CT), the toolkit combines a modular Python backend with an intuitive graphical interface (*nDTomoGUI*). This dual structure supports both rapid visual exploration and reproducible scripting, helping researchers and students alike engage with high-dimensional spectroscopic or scattering/diffraction imaging data.

The design philosophy behind *nDTomo* prioritises simplicity and transparency. Core functions are implemented with minimal abstraction to promote clarity and extensibility, allowing users to easily prototype new workflows without needing to navigate deep class hierarchies or opaque APIs. While originally developed for internal use during active beamline experiments, *nDTomo* has matured into a general-purpose tool that supports both routine analysis and research-driven method development.

Key features include:

- A flat, function-oriented architecture that facilitates accessibility and modification.
- Integrated workflows for phantom generation, tomographic acquisition simulation, sinogram correction and tomographic image reconstruction.
- Interactive tools for exploring chemical images and performing pixel-wise spectral analysis or peak fitting.
- A PyQt-based GUI (*nDTomoGUI*) offering interactive exploration of hyperspectral imaging data, ROI image and histogram selection, simple image segmentation and batch peak fitting.
- A curated set of tutorial notebooks showcasing best practices in chemical image and tomography analysis and GPU-accelerated modelling for tomographic image reconstruction and peak fitting.

The toolkit has been used in a number of XRD-CT studies, including *operando* experiments on catalytic reactors, lithium-ion batteries and fuel cells as well as for algorithm development.[61,62] Although much of this usage occurred during active development and was driven by the first author, recent releases with expanded documentation and structured tutorials mark a shift toward broader community use.

6.1 Future directions

Several areas are under active development to enhance *nDTomo*'s performance, usability, and scientific reach:

- Expanded peak fitting capabilities: including multi-peak models, asymmetric line shapes and more flexible background functions.
- Full GPU acceleration: across all core workflows such as tomographic reconstruction, spectral fitting and deep learning to enable efficient processing of large-scale datasets.
- Self-supervised denoising for chemical tomography: developing methods that exploit the structure of hyperspectral and tomographic data to suppress noise without requiring ground truth, improving data quality in low-dose or time-constrained experiments.
- GPU-accelerated image and volume registration: implementing fast, PyTorch-based registration techniques for aligning 2D/3D datasets.
- Physics-informed direct segmentation from sinograms: exploring the feasibility of learning mappings from raw sinogram data to segmented images by incorporating basic physical constraints, bypassing intermediate reconstructions.

Alongside these technical developments, we continue to grow *nDTomo* as a resource for education and training. Its successful use in Finden-led workshops has demonstrated its suitability for early-career researchers, and ongoing improvements to the documentation and tutorials aim to support its integration into graduate-level materials science and imaging curricula.

Transparency and reproducibility remain core values of the project. By using standard data formats, clear function-based design and version-controlled tutorials, *nDTomo* enables researchers to track, share, and build upon each other's analyses, especially in collaborative, multi-institutional contexts such as synchrotron beamtime experiments involving many partners.

We are committed to growing the user community through open development, responsive feedback, and shared benchmarks. Contributions, extensions, and real-world use cases are welcome through the GitHub repository, where active development continues. As chemical imaging moves toward higher resolution, greater complexity, and faster acquisition speeds, tools like *nDTomo* tools like *nDTomo* will play a critical role in bridging the gap between raw data and scientific insight. By prioritising transparency, reproducibility, and accessibility, *nDTomo* aims to support the next generation of chemical imaging workflows.

Author contributions

A. V. led the conceptual design, development, and documentation of the *nDTomo* software suite. A. V. implemented the majority of the codebase, created the graphical user interface (GUI) and authored the tutorial notebooks. E. P. reviewed the notebooks creating also interactive and cloud-based execution environments. A. V. and H. D. co-developed the CT reconstruction approaches in Pytorch. A. V., R. D. and S. J. C. designed and implemented the PeakFitCNN as well as the Pytorch-based peak fitting approaches. A. V., E. P., H. D. and S. D. M. J. coordinated and delivered training workshops that shaped the educational design of the software. A. M. B. and S. D. M. J. secured funding for this work. The manuscript was written by A. V. with contributions and feedback given from all co-authors.

Conflicts of interest

The authors declare that they have no competing interests.

Data availability

The *nDTomo* software is open source and available under the GNU General Public License (GPLv3). The source code, example notebooks, and installation instructions can be found on GitHub:

- GitHub repository: **https://github.com/antonyvam/nDTomo**.
- Documentation (ReadTheDocs): **https://nDTomo.readthedocs.io**.
- Archived stable release (Zenodo): **https://doi.org/10.5281/zenodo.15483595**.

The GitHub repository includes a Conda environment file for reproducible setup, tutorial notebooks and extensive function-level documentation. Users are encouraged to submit issues, request features, or contribute enhancements through the GitHub platform.

The SI contains the results from the application of the PeakFitCNN in real space data and using the DLSR approach and it is compared with the conventional approaches. See DOI: **https://doi.org/10.1039/d5dd00252d**.

Acknowledgements

AV acknowledges funding through the Royal Society (IF\R2\222059) as a Royal Society Industry Fellow. The authors would like to thank participants of the chemical imaging training workshops hosted by Finden Ltd, whose feedback helped improve usability and tutorial coverage. The authors are grateful to the early users and collaborators whose feedback helped shape the development of *nDTomo* and in particular we would like to thank Dr Marco di Michiel (ESRF), Dr Gavin Vaughan (ESRF), Dr Dorota Matras (UKBIC), Dr Donal Finegan (NREL), Dr Tom Heenan (Gaussion Ltd), Dr Chun Tan (Gaussion Ltd), Dr Stephen Price (Finden Ltd) and Mr John Morley (Imperial College London). This project has received funding from the European Union's Horizon Europe Research and Innovation Programme, under grant agreement no. 101069690. The authors acknowledge that the open access fee for this publication was covered by the Imperial College London Open Access Fund.

References

1 N. E. Omori, *et al.*, Recent Developments in X-ray Diffraction/Scattering Computed Tomography for Materials Science, *Philos. Trans. R. Soc., A*, 2023, **381**(2259), DOI: **10.1098/rsta.2022.0350**.

2 D. Matras, *et al.*, Tomography in Catalyst Design, in *Heterogeneous Catalysts*, John Wiley & Sons, Ltd, 2021, ch. 15, pp. 263–278, DOI: **10.1002/9783527813599.ch15**, ISBN: 978-3-527-81359-9, **https://onlinelibrary.wiley.com/doi/pdf/10.1002/9783527813599.ch15**.

3 D. Matras, *et al.*, Case Studies: Mapping Using X-Ray Absorption Spectroscopy (XAS) and Scattering Methods, in *Springer Handbook of Advanced Catalyst Characterization*, ed. I. E. Wachs and M. A. Bañares, Springer International Publishing, Cham, 2023, pp. 671–688, DOI: **10.1007/978-3-031-07125-6_31**, ISBN: 978-3-031-07125-6.

4 A. Mirone, *et al.*, The PyHST2 hybrid distributed code for high speed tomographic reconstruction with iterative reconstruction and a priori knowledge capabilities, *Nucl. Instrum. Methods Phys. Res., Sect. B*, 2014, **324**, 41–48, DOI: **10.1016/j.nimb.2013.09.030**.

5 W. De Nolf, *et al.*, EWOKS: An Extensible WOrKflow System, *Synchrotron Radiation News*, 2024, **37**(6), 9–15, DOI: **10.1080/08940886.2024.2432305**.

6 Y. Liu, *et al.*, *TXM-Wizard*: a program for advanced data collection and evaluation in full-field transmission X-ray microscopy, *J. Synchrotron Radiat.*, 2012, **19**(2), 281–287, DOI: **10.1107/S0909049511049144**.

7 N. T. Vo, *et al.*, Data processing methods and data acquisition for samples larger than the field of view in parallel-beam tomography, *Opt. Express*, 2021, **29**(12), 17849–17874, DOI: **10.1364/OE.418448**.

8 G. B. M. Vaughan, *et al.*, ID15A at the ESRF – a beamline for high speed operando X-ray diffraction, diffraction tomography and total scattering, *J. Synchrotron Radiat.*, 2020, **27**(2), 515–528, DOI: **10.1107/S1600577519016813**.

9 Finden Ltd, **https://www.finden.co.uk/**, accessed: 2025-06-04.

10 STORMING Project: Structured Unconventional Reactors for CO-Free Methane Catalytic Cracking, **https://storming-project.eu/**, accessed: 2025-06-04.

11 D. Matras, *et al.*, Emerging Chemical Heterogeneities in a Commercial 18650 NCA Li-ion Battery during Early Cycling Revealed by Synchrotron X-ray Diffraction Tomography, *J. Power Sources*, 2022, **539**, 231589, DOI: **10.1016/j.jpowsour.2022.231589**.

12 F. de la Peña, *et al.*, *hyperspy/hyperspy: v2.3.0*, 2025, DOI: **10.5281/zenodo.14956374**.

13 V. A. Solé, *et al.*, A multiplatform code for the analysis of energy-dispersive X-ray fluorescence spectra, *Spectrochim. Acta, Part B*, 2007, **62**(1), 63–68, DOI: **10.1016/j.sab.2006.12.002**.

14 M. Basham, *et al.*, Data Analysis WorkbeNch (DAWN), *J. Synchrotron Radiat.*, 2015, **22**(3), 853–858, DOI: **10.1107/S1600577515002283**.

15 M. Lerotic, *et al.*, MANTiS: a program for the analysis of X-ray spectromicroscopy data, *J. Synchrotron Radiat.*, 2014, **21**(5), 1206–1212, DOI: **10.1107/S1600577514013964**.

16 S. Kabiri and S. M. O'Rourke, HyperGUI: A Web Application for Hyperspectral Image Analysis and Data Extraction, *Journal of Open Research Software*, 2024, **12**(1), 9, DOI: **10.5334/jors.509**.

17 A. Vamvakeros, *et al.*, DLSR: A Solution to the Parallax Artefact in X-ray Diffraction Computed Tomography Data, *J. Appl. Crystallogr.*, 2020, **53**(6), 1531–1541, DOI: **10.1107/S1600576720013576**.

18 W. van Aarle, *et al.*, The ASTRA Toolbox: a platform for advanced algorithm development in electron tomography, *Ultramicroscopy*, 2015, **157**, 35–47, DOI: **10.1016/j.ultramic.2015.05.002**.

19 J. S. Jørgensen, *et al.*, Core Imaging Library - Part I: a versatile Python framework for tomographic imaging, *Philos. Trans. R. Soc., A*, 2021, **379**(2199), 20200192, DOI: **10.1098/rsta.2020.0192**.

20 E. Papoutsellis, *et al.*, Core Imaging Library - Part II: multichannel reconstruction for dynamic and spectral tomography, *Philos. Trans. R. Soc., A*, 2021, **379**(2204), 20200193, DOI: **10.1098/rsta.2020.0193**.

21 D. Gürsoy, *et al.*, TomoPy: a framework for the analysis of synchrotron tomographic data, *J. Synchrotron Radiat.*, 2014, **21**(5), 1188–1193, DOI: **10.1107/S1600577514013939**.

22 A. Hendriksen, *et al.*, Tomosipo: Fast, Flexible, and Convenient 3D Tomography for Complex Scanning Geometries in Python, *Opt. Express*, 2021, **29**(24), 40494, DOI: **10.1364/oe.439909**.

23 J. C. da Silva, *et al.*, High energy near- and far-field ptychographic tomography at the ESRF, in *Developments in X-Ray Tomography XI*, ed. B. Müller and G. Wang, International Society for Optics and Photonics, SPIE, 2017, vol. 10391, p. 1039106, DOI: **10.1117/12.2272971**.

24 A. Biguri, *et al.*, TIGRE: a MATLAB-GPU toolbox for CBCT image reconstruction, *Biomedical Physics and Engineering Express*, 2016, **2**(5), 055010, DOI: **10.1088/2057-1976/2/5/055010**.

25 C. R. Harris, *et al.*, Array programming with NumPy, *Nature*, 2020, **585**(7825), 357–362, DOI: **10.1038/s41586-020-2649-2**.
26 A. Paszke, *et al.*, PyTorch: an imperative style, high-performance deep learning library, in *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Curran Associates Inc., Red Hook, NY, USA, 2019.
27 J. Wloka, Tomografia, in *Protokolle des Seminars "Anwendungen der Mathematik" (Lukaszewicz, Perkal, Steinhaus)*, Universität Breslau, Breslau, Poland, 1953.
28 P. Gilbert, Iterative methods for the three-dimensional reconstruction of an object from projections, *J. Theor. Biol.*, 1972, **36**(1), 105–117, DOI: **10.1016/0022-5193(72)90180-4**, ISSN: 0022-5193, **https://www.sciencedirect.com/science/article/pii/0022519372901804**.
29 M. R. Hestenes and E. Stiefel, Methods of conjugate gradients for solving linear systems, *J. Res. Natl. Bur. Stand.*, 1952, **49**, 409–435.
30 P. Virtanen, *et al.*, SciPy 1.0: fundamental algorithms for scientific computing in Python, *Nat. Methods*, 2020, **17**(3), 261–272, DOI: **10.1038/s41592-019-0686-2**.
31 S. van der Walt, *et al.*, scikit-image: image processing in Python, *PeerJ*, 2014, **2**, e453, DOI: **10.7717/peerj.453**.
32 F. Pedregosa, *et al.*, Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research*, 2011, **12**, 2825–2830.
33 J. D. Hunter, Matplotlib: a 2D graphics environment, *Comput. Sci. Eng.*, 2007, **9**(3), 90–95, DOI: **10.1109/MCSE.2007.55**.
34 Riverbank Computing Limited, *PyQt: Python bindings for the Qt application framework*, 2025, **https://www.riverbankcomputing.com/software/pyqt/**, accessed: 2025-05-27.
35 A. Vamvakeros, *et al.*, Interlaced X-ray Diffraction Computed Tomography, *J. Appl. Crystallogr.*, 2016, **49**(2), 485–496, DOI: **10.1107/S160057671600131X**.
36 A. Vamvakeros, *et al.*, 5D Operando Tomographic Diffraction Imaging of a Catalyst Bed, *Nat. Commun.*, 2018, **9**(1), 4751, DOI: **10.1038/s41467-018-07046-8**.
37 K. He, *et al.*, Deep Residual Learning for Image Recognition, *arXiv*, 2015, preprint, arXiv: 1512.03385, DOI: **10.48550/arXiv.1512.03385**.
38 O. Ronneberger, Convolutional Networks for Biomedical Image Segmentation, *arXiv*, 2015, preprint, arXiv: 1505.04597, DOI: **10.48550/arXiv.1505.04597**.
39 Z. Wang, *et al.*, Image quality assessment: from error visibility to structural similarity, *IEEE Transactions on Image Processing*, 2004, **13**(4), 600–612, DOI: **10.1109/TIP.2003.819861**.
40 I. M. Sobol', On the distribution of points in a cube and the approximate evaluation of integrals, *USSR Computational Mathematics and Mathematical Physics*, 1967, **7**(4), 86–112, DOI: **10.1016/0041-5553(67)90144-9**, ISSN: 0041-5553, **https://www.sciencedirect.com/science/article/pii/0041555367901449**.
41 A. Vamvakeros, *et al.*, Cycling Rate-Induced Spatially-Resolved Heterogeneities in Commercial Cylindrical Li-Ion Batteries, *Small Methods*, 2021, **5**(9), 2100512, DOI: **10.1002/smtd.202100512**.
42 S. R. Daemi, *et al.*, Exploring Cycling Induced Crystallographic Change in NMC with X-ray Diffraction Computed Tomography, *Phys. Chem. Chem. Phys.*, 2020, **22**(32), 17814–17823, DOI: **10.1039/D0CP01851A**.
43 D. P. Finegan, *et al.*, Spatial quantification of dynamic inter and intra particle crystallographic heterogeneities within lithium ion electrodes, *Nat. Commun.*, 2020, **11**(1), 631, DOI: **10.1038/s41467-020-14467-x**.
44 T. Karsten, *et al.*, Multi-Scale Studies of 3D Printed Mn–Na–W/SiO_2 Catalyst for Oxidative Coupling of Methane, *Catalysts*, 2021, **11**(3), DOI: **10.3390/catal11030290**.
45 D. M. Farmer, *et al.*, Following Cu Microstructure Evolution in CuZnO/Al_2O_3(-Cs) Catalysts during Activation in H_2 using in situ XRD and XRD-CT, *Chem.: Methods*, 2023, **3**(1), e202200015, DOI: **10.1002/cmtd.202200015**.
46 C. Jacquot, *et al.*, A Multi-Scale Study of 3D Printed Co-Al_2O_3 Catalyst Monoliths versus Spheres, *Chem. Eng. J. Adv.*, 2023, **16**, 100538, DOI: **10.1016/j.ceja.2023.100538**.
47 D. Matras, *et al.*, Multi-Length Scale 5D Diffraction Imaging of Ni–Pd/CeO_2–ZrO_2/Al_2O_3 Catalyst during Partial Oxidation of Methane, *J. Mater. Chem. A*, 2021, **9**(18), 11331–11346, DOI: **10.1039/D1TA01464A**.
48 A. Iborra-Torres, *et al.*, 3D Printed $SrNbO_2N$ Photocatalyst for Degradation of Organic Pollutants in Water, *Mater. Adv.*, 2023, **4**(16), 3461–3472, DOI: **10.1039/D2MA01076C**.
49 V. Middelkoop, *et al.*, 3D Printed Ni/Al_2O_3 Based Catalysts for CO_2 Methanation - A Comparative and Operando XRD-CT Study, *J. CO2 Util.*, 2019, **33**, 478–487, DOI: **10.1016/j.jcou.2019.07.013**.
50 C. A. Grande, *et al.*, Multiscale Investigation of Adsorption Properties of Novel 3D Printed UTSA-16 Structures, *Chem. Eng. J.*, 2020, **402**, 126166, DOI: **10.1016/j.cej.2020.126166**.
51 D. Matras, *et al.*, In situ X-ray diffraction computed tomography studies examining the thermal and chemical stabilities of working $Ba_{0.5}Sr_{0.5}Co_{0.8}Fe_{0.2}O_3$-membranes during oxidative coupling of methane, *Phys. Chem. Chem. Phys.*, 2020, **22**(34), 18964–18975, DOI: **10.1039/D0CP02144J**.
52 T. Li, *et al.*, Design of next-generation ceramic fuel cells and real-time characterization with synchrotron X-ray diffraction computed tomography, *Nat. Commun.*, 2019, **10**(1), 1497, DOI: **10.1038/s41467-019-09427-z**.
53 T. M. M. Heenan, *et al.*, The Detection of Monoclinic Zirconia and Non-Uniform 3D Crystallographic Strain in a Re-Oxidized Ni-YSZ Solid Oxide Fuel Cell Anode, *Crystals*, 2020, **10**(10), DOI: **10.3390/cryst10100941**.
54 G. Ashiotis, *et al.*, The fast azimuthal integration Python library: pyFAI, *J. Appl. Crystallogr.*, 2015, **48**(2), 510–519, DOI: **10.1107/S1600576715004306**.
55 A. A. Coelho, TOPAS and TOPAS-Academic: An Optimization Program Integrating Computer Algebra and Crystallographic Objects Written in C++, *J. Appl. Crystallogr.*, 2018, **51**(1), 210–218, DOI: **10.1107/S1600576718000183**.
56 I. Martens, *et al.*, Imaging Heterogeneous Electrocatalyst Stability and Decoupling Degradation Mechanisms in

Operating Hydrogen Fuel Cells, *ACS Energy Lett.*, 2021, **6**(8), 2742–2749, DOI: **10.1021/acsenergylett.1c00718**.

57 L. Jones, P. Seller, *et al.*, HEXITEC ASIC—a pixellated readout chip for CZT detectors, *Nucl. Instrum. Methods Phys. Res., Sect. A*, 2009, **604**(1), 34–37, DOI: **10.1016/j.nima.2009.01.046**.

58 M. C. Veale, *et al.*, Preliminary characterisation of the HEXITECMHz spectroscopic X-ray imaging detector, *J. Instrum.*, 2023, **18**(07), P07048, DOI: **10.1088/1748-0221/18/07/P07048**.

59 D. P. Clark, *et al.*, Photon-counting cine-cardiac CT in the mouse, *PLoS One*, 2019, **14**, 1–17, DOI: **10.1371/journal.pone.0218417**.

60 F. A. Huber, *et al.*, Detection and Characterization of Monosodium Urate and Calcium Hydroxyapatite Crystals Using Spectral Photon-Counting Radiography: A Proof-of-Concept Study, *Eur. J. Radiol.*, 2020, **129**, 109080, DOI: **10.1016/j.ejrad.2020.109080**.

61 H. Dong, *et al.*, A Scalable Neural Network Architecture for Self-Supervised Tomographic Image Reconstruction, *Digital Discovery*, 2023, **2**(4), 967–980, DOI: **10.1039/D2DD00105E**.

62 H. Dong, *et al.*, Obtaining Parallax-Free X-ray Powder Diffraction Computed Tomography Data with a Self-Supervised Neural Network, *npj Comput. Mater.*, 2024, **10**(1), 201, DOI: **10.1038/s41524-024-01389-1**.