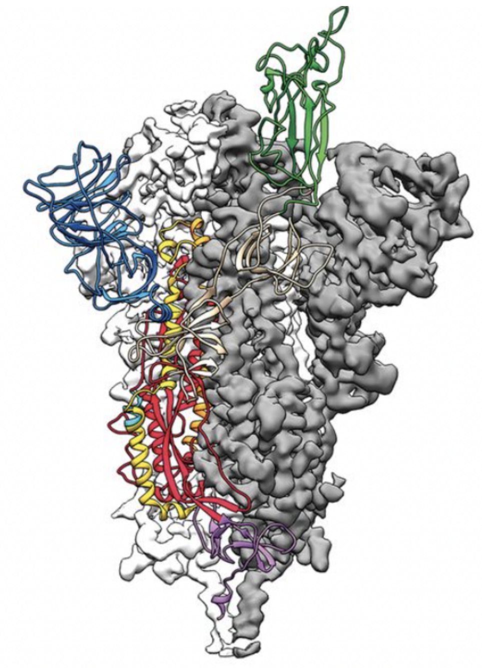
Molecular Reconstruction: Simulation and Algorithms for cryo-EM



Name Tyler Heim
Callum Hepworth
Arjun Swani
Jed Yeo

Sponsor Geoffrey Woollard
PhD Student
UBC Computer Science

Project 2203
ENPH459
Engineering Physics Project Laboratory
The University of British Columbia

April 11, 2022

Executive Summary

The structure of biomolecules is of great interest to researchers around the globe, as their function is intricately related to their highly complex shapes. Traditional methods such as X-ray crystallography are used to examine their structure closely. However, this requires a macroscopic crystal sample, i.e., an enormous number of protein samples all in the same regular orientation [3].

Cryogenic electron microscopy (cryo-EM) aims to overcome these limitations to achieve this, as we only require individual macromolecules. Transmission electron microscopy is performed on cryogenically cooled samples to cast shadows of proteins in various orientations. We can then use these "shadows" to algorithmically reconstruct the shape of the proteins that cast them.

Several problems currently beset the cryo-EM research community. Firstly, as a relatively new, emerging field, there is a lack of centralized projects and resources, meaning that research efforts are disparate and sparse. Further, a lack of unified metrics means no standardized process to evaluate the efficacy of reconstruction models. Finally, microscopy of real-life bio-samples remains challenging to produce, thus artificial data is required to efficiently develop reconstruction algorithms.

The goal of this project is to extend the compSPI toolkit to use Bayesian expectation-maximization iterative refinement for the 3D reconstruction of proteins from simulated cryo-EM images. This report covers our implementation of this toolkit.

Contents

List of Figures

1. Introduction

From enzymes catalyzing chemical reactions to antibodies for bolstering immune systems, proteins are vital actors in living organisms. Much of their function is associated with their structure. Research into their structure has resulted in better understanding causes of disease, developing new drugs and treatments and more recently enabling us to rapidly develop vaccines for global epidemics (Figure 1.1).

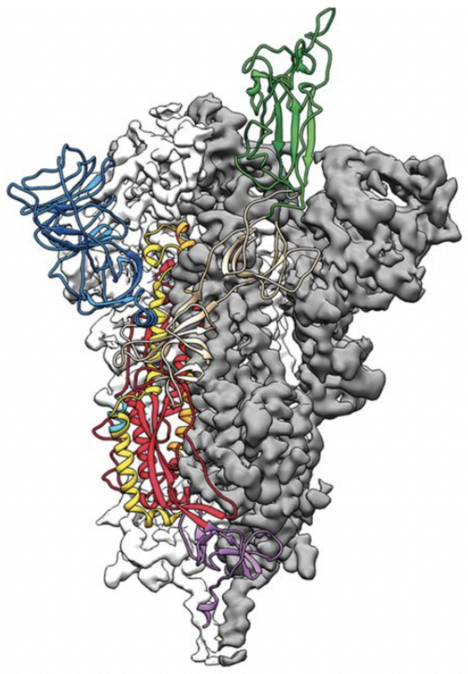


Figure 1.1: 2019-nCoV S primary structure

The primary structure of the spike protein of the Covid-19 virus is shown. The spike protein is the key point of viral engagement, meaning that disabling it disrupts the virus's ability to infect host cells [19]. Imaging viral proteins allows analyzing binding sites for vaccine development trials: understanding the structure of this spike protein was a key step in developing the Covid-19 vaccine.

Within just a few weeks of the publication of the virus genome, researchers had resolved the main features of the molecule using cryogenic electron microscopy (colloquially, cryo-EM). Current research regarding cryo-EM seeks to resolve the continuous heterogeneity of molecules at finer resolutions using novel reconstruction models.

1.1 Our Sponsor

Our sponsor is Geoffrey Woollard, a PhD-track master's degree student in the Department of Computer Science at the University of British Columbia. Mr. Woollard worked at Structural Biotechnology developing source code of cryoSPARC (software platform for cryo-EM) [9] in 2018 and has years of experience performing cryo-EM experiments at the University of Toronto. His main research interests lie in structural biology, inverse problems in scientific imaging, and how cryo-EM plays a role in today's particle realization problems.

As a means of furthering research into cryo-EM, Mr. Woollard is an early contributor to the open-source project "compSPI" (computational single particle imaging): a software toolkit with the aim of benchmarking the performance of the 3D reconstruction of proteins from simulated cryo-EM images. This toolkit is in its infancy but has a dedicated team of maintainers comprised of Nina Miolane, an Assistant Professor of Electrical & Computer Engineering at the University of California Santa Barbara, and Frédéric Poitevin, a Staff Scientist at the Stanford Linear Accelerator Center. Contribution to compSPI will form the foundation of this project.

1.2 Project Background

Developed in the early 20th century, X-ray crystallography was generally accepted as the superior imaging method capable of atomic resolution [4]. However, this requires a macroscopic crystal sample of the protein of interest. One must then acquire many protein samples, all in the same regular orientation [3]. Such a process can take months or even years. Even if it succeeds, the crystalline sample of the protein is far removed from its natural state (typically an aqueous solution), meaning that structures might be compromised or damaged. Moreover, samples may be difficult to purify [3].

This a method of imagery that required a single particle, leveraging electron beams. Electrons do not propagate in air - so imagery must take place in a vacuum environment [3]. Further, molecules must remain hydrated, and any hydrating solution will subsequently boil off in a vacuum. Ergo, samples are flash frozen [3]. Thus, molecules remain held in place and structure is preserved.

Electrons are sent through the frozen molecule and produce "voltage shadows" on photographic film [3]. These shadows are noisy, 2-dimensional, and randomly oriented. This imaging process is referred to as cryogenic electron microscopy, or more commonly cryo-EM. This process is well documented, and so the present region of interest in cryo-EM research lies in working with its output: reassembling 2D particle potential shadows into a reconstructed 3D particle.

Although there exist some single particle imaging suites in the cryo-EM field (such as cryoSPARC or RELION [16]), numerous pitfalls surround this research community. As of 2020, 150,000 biological samples have been resolved using X-ray crystallography. In contrast, cryo-EM has only 6,000 samples imaged [17], in part due to its nature as an emerging field. Our sponsors have identified three main issues affecting the field.

1. Lack of reference code-bases and centralized projects in cryo-EM. This causes research efforts to be disparate and sparse.
2. Lack of a standardized, unified set of reconstruction model metrics means that researchers have no means of comparison for the efficacy of existing and developed reconstruction models.
3. Real-life bio-molecules suitable for imagery remain difficult to produce, thereby simulated data is required. The ground truth is elusive in an experimental setting, so researchers require robust simulated data for algorithmic development.

1.3 Project Objectives

The proposed solution to these issues is compSPI, the main objectives of which can be summarized into three distinct areas.

1. **Open source dataset generation & storage**

   compSPI will act as a centralized resource for anyone interested in simulating and storing molecular datasets.

2. **Modularization of existing reconstruction workflows**

   While reconstruction models exist, centralized, modular implementations would improve extensibility and keep pace with the accelerating rate of change in cryo-EM research. Our reconstruction suite will modularize these existing implementations.

3. **Facilitating performance comparisons between reconstruction methods**

   compSPI will serve as a benchmarking suite for other reconstruction models. Facilitating the comparison of models against one another will aid in the development process for novel methods.

To achieve this, the compSPI toolkit is divided into three main repositories, which will serve as the key development areas for this project.

1. `simSPI` serves to generate and simulate artificial target cryo-EM datasets of choice.
2. `ioSPI` acts as an interface between a cloud storage solution for easy dataset access and storage.
3. `reconstructSPI` implements a reconstruction method to serve as the standard comparison model.

1.4 Scope & Limitations

The scope of this capstone project is developing and engineering compSPI to be the gold standard toolkit in molecular reconstruction algorithmic development - not to pioneer novel reconstruction methods nor resolve particles to competitive resolutions.

Usage of a deep generative model was suggested as an interesting method to compare reconstruction efficacy; however, we opted to reduce scope and instead focus on an iterative refinement approach. The scope was further narrowed after consultations with our sponsor to only include simulation of artificial molecular data, handling of said data, and a Bayesian-based iterative refinement method for reconstruction (an existing well-founded method [13]). These wholly represent the intended design of compSPI - the reference codebase for future algorithmic development in cryo-EM single particle biological imaging.

A requirement of the data pipeline was that dataset generation and storage functionality be developed before implementing the reconstruction model, which relied on these processes. The amount of requisite background knowledge for cryo-EM development combined with stringent coding conventions led to a backlog during development of the TEM wrapper and OSF.io modules, delaying the development of reconstruction. As a result, we opted to implement a proof-of-concept reconstruction suite instead of a holistic implementation. This final narrowing yielded the scope of our final deliverable.

2. Discussion

2.1 Theory & Fundamentals of cryo-EM Imaging & Molecular Reconstruction

2.1.1 Overview

Mathematical Abstraction of cryo-EM Imaging

Suppose there are 2D slices encompassing all possible 2D samples generated by a physical simulator with known parameters. Taking random samples from this space, called the image formation model, is akin to observing particle slices taken using a physical imaging apparatus. The model, with samples $X_i,\ i \in \{1, \ldots, n\}$ is defined as

$$X_i = PSF_i \star (t_i \circ \Pi_{2D} \circ R_i)(V^{(i)}) + \epsilon_i$$

[5]. This formulation is comprised of factors applied to a 3D particle volume $V^{(i)}$, representing the i^{th} idealized particle volume. These factors form matrix transformations applied to $V^{(i)}$, and represent physical perturbations of a particle during the imaging process.

- t_i : translation of the particle relative to the volume center
- Π_{2D} : the 2D potential shadow of the particle volume
- R_i : the 3D rotation of the particle
- PSF_i : the point spread function (PSF), representing the distorting effects and spherical aberration of the microscope lens
- ϵ_i : latent noise in the particle slice

The application of this model to an example image can be seen in Figure 2.1.

Leveraging the convolution theorem this model is converted to Fourier space.

$$\widetilde{X_i} = \text{CTF}_i \odot (P_i \circ \widetilde{V}^{(i)}) + N_i$$

Each of the image space quantities has an analogue in Fourier space, so the underlying structure of the model remains intact.

- t_i, Π_{2D}, and R_i combine to form P_i, the Fourier space projection of the particle volume

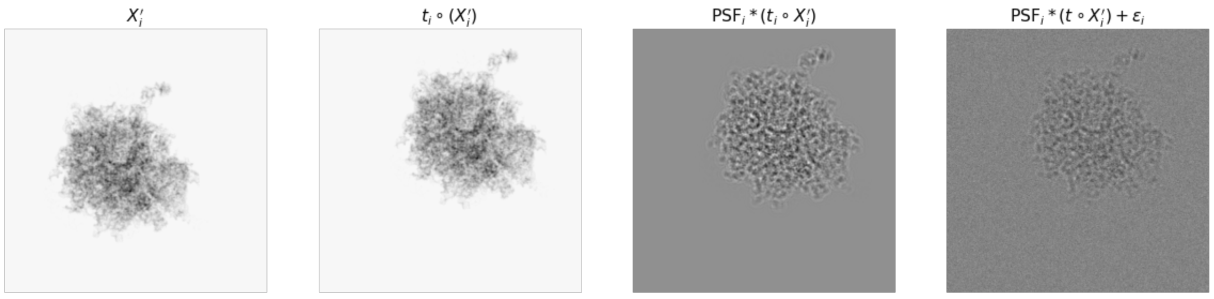

X_i'
$t_i \circ (X_i')$
$\mathrm{PSF}_i * (t_i \circ X_i')$
$\mathrm{PSF}_i * (t \circ X_i') + \varepsilon_i$

Figure 2.1: Factor-by-factor application of the image formation model to an example particle volume [10]. All factors are as defined, save $X_i' = (\Pi_{2D} \circ R_i)(V^{(i)})$.

- PSF_i : the PSF is transformed to become the contrast transfer function (CTF)
- ϵ_i is transformed to become N_i

For its advantages in computational efficiency, as well as other simplifying factors explained in future sections, the Fourier space representation of the image formation model is used most often in the generation of simulated particle images.

Iterative Refinement Summary

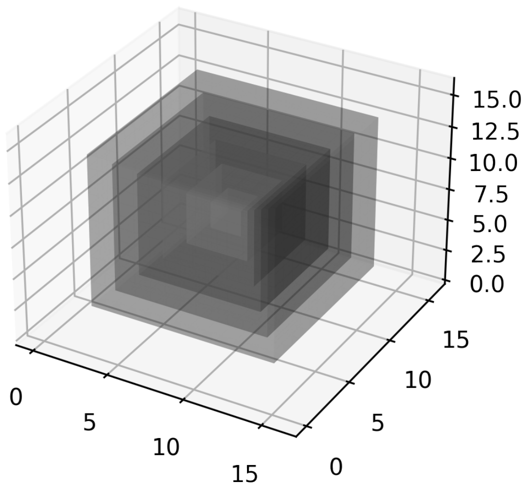
At the request of our sponsor, we implemented a Bayesian Expectation Maximization algorithm for molecular reconstruction that closely followed [13]. The goal is to raise the resolution and accuracy of an existing 3D model of a particle. The inputs are 2D projections of the particle produced through cryo-EM (or the TEM simulator), and a rough 3D model of the particle. The rough 3D model must be non-uniform under rotation. The large challenges of reconstruction are determining the orientation at which particles were imaged, and addressing the poor signal-to-noise ratios of the data (Figure 2.2).

The algorithm proceeds in Fourier space, pulling "slices" out of the model at uniformly distributed rotations (Figure 2.3). Each projection is compared with each slice, with pairs having assigned weights correlating with their similarity. Using a weighted combination of the slices in 3D, each 2D projection is approximated in 3D. Combining these approximations forms a new 3D model. The process iterates by using this new model in place of the old, ≈ 7 iterations is typical.

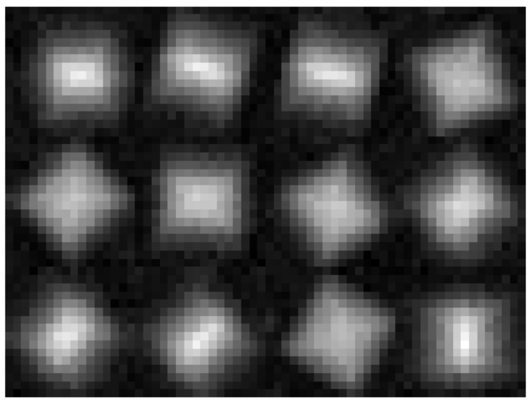
2.1.2 Concepts

Fourier Slice Theorem

In order to image particles, cryo-EM uses electrons, which lose energy due to being impeded by particles along their trajectories, so when the electrons contact the measurement surface a "shadow" is produced (Figure 2.2). In effect, electrons are performing a line integral of particle density over their trajectories. The 2D projection is an array of these line integrals.



(a) The particle (Units: px).



(b) Simulated cryo-EM-like projections.

Figure 2.2: Simple cubic "particle" used to demonstrate the reconstruction algorithm.

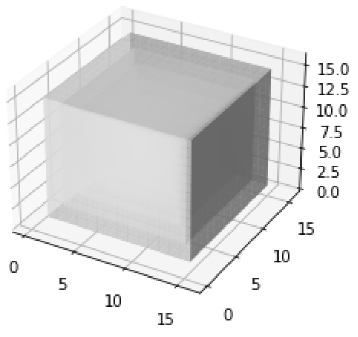
The particle is a cube of side length 12px consisting of concentric shells of decreasing magnitude, i.e., the "core" of side length 2px has magnitude 6 and the "skin" of side length 12px has magnitude 1. This sample was chosen as it is identifiable to the human eye in noisy situations, it is non-uniform under rotation, and it is well visualized using `matplotlib`'s `Axis.voxels` method.

Repeatedly performing line integrals is expensive. In Fourier space, the images of particles are represented by "slices" through the center of the 3-dimensional particle, where these slices lie parallel to their corresponding particles' projections. See Appendix A for mathematical details.

Bayesian Weighting

In order to assess the rotations of particle projections, we first need to develop materials against which we can compare them. We accomplish this by starting with an approximate 3D model of the particle. This model will have slices extracted (in Fourier space) at known rotations. Rotations will be uniformly distributed across the 3-dimensional rotational group SO(3) to develop a discrete sampling of all possible slices. As a result of the Fourier Slice Theorem, these slices can be directly compared against the Fourier transforms of particle projections. The goal is to assess how similar each projection is to each slice of known rotation. Using this data, we can approximate each projection as a weighted combination of slices, and since we know their rotations, this combination can be put into the 3D map.

The weighting of the slices is a Bayesian Expectation Maximization algorithm that closely follows [3]. The mathematics behind this are listed in Appendix C. In effect, weight$_i$ correlates with slice$_i$ resembling the given projection when imaged. By weighing each slice against a given projection, we devise a representation of the projection in 3D Fourier space. weighing all combinations of slices and projections, a new generation model is assembled.


(a) Source model.

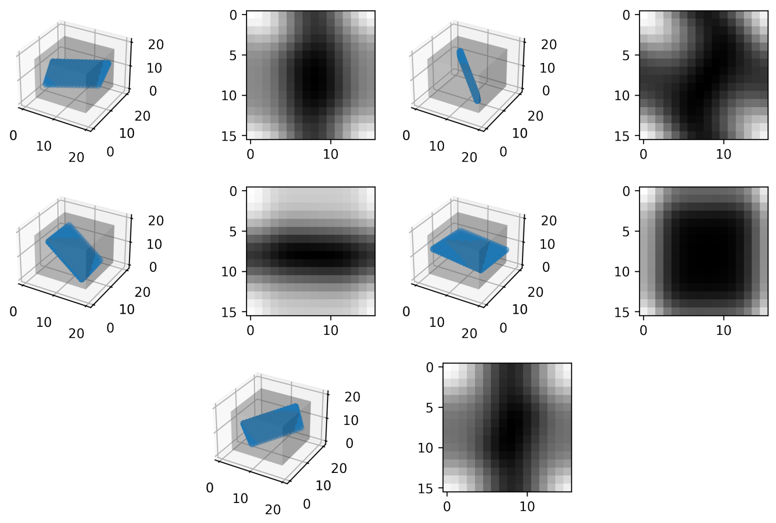

(b) Orientation planes (left) and slices (right) pulled from the 3D model of the particle.

Figure 2.3: Slices pulled from a partially reconstructed particle half-map.

Slices were pulled from randomly oriented planes intersecting the central point of the source model. Slices from a perfect reconstruction would resemble cube cross-sections as in 2.2b, however these demonstrate an imperfect reconstruction. These slices are compared to the projections in 2.2b in the Bayesian weighting algorithm. Data shown are in real space but note slices were pulled from the model in Fourier space.

Noise, CTFs, and Wiener Filters

The presence of non-Gaussian noise in images severely degrades the Bayesian model's ability to compare images. One such form in cryo-EM is artefacts due lens effects. These are abstracted into the contrast transfer function (CTF) - a mapping of artefacts' effects on an image in Fourier space.

The CTF is applied to slices prior to comparison with projections. Comparing noisy images is preferable to comparing "cleaned" images as the comparison is more robust to errors in the noise model.

The CTF is also used to filter noise from projections to increase comparability to slices via the Wiener Filter function, which for our purposes reads:

$$Y(\vec{r}) = \frac{\text{CTF}(\vec{r}) \cdot X(\vec{r})}{|\text{CTF}(\vec{r})|^2 + \alpha}.$$

Where α is some small number used to tune the filter and prevent singularities - large α will reduce noise and feature definition, whereas small α will amplify the same. Tuning α is covered in Appendix F.

Half-maps were also used to reduce noise artefacts, along with the Fourier Shell Correlation (FSC). These are detailed in Appendix B.

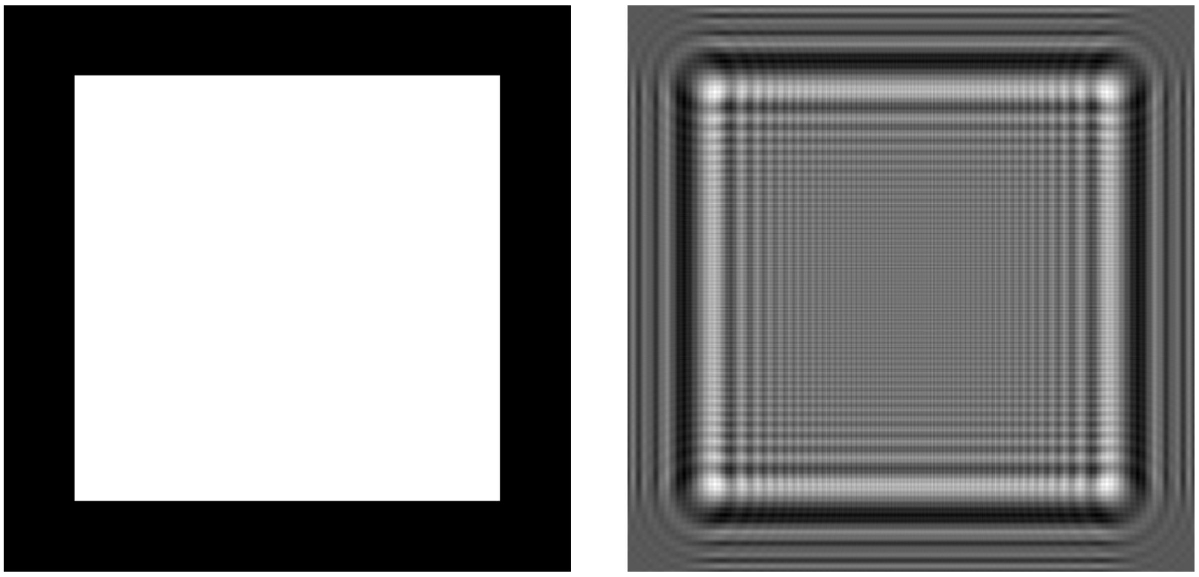

(a) Image pre-CTF. (b) Image post-CTF.

Figure 2.4: Contrast Transfer Function on a white square on a black background.

Note the periodic artefacts that have been introduced around the borders of the square - these are typical effects of the microscopy effects in cryo-EM. The CTF is applied in Fourier space, the images shown are in real space.

2.2 Repository Design

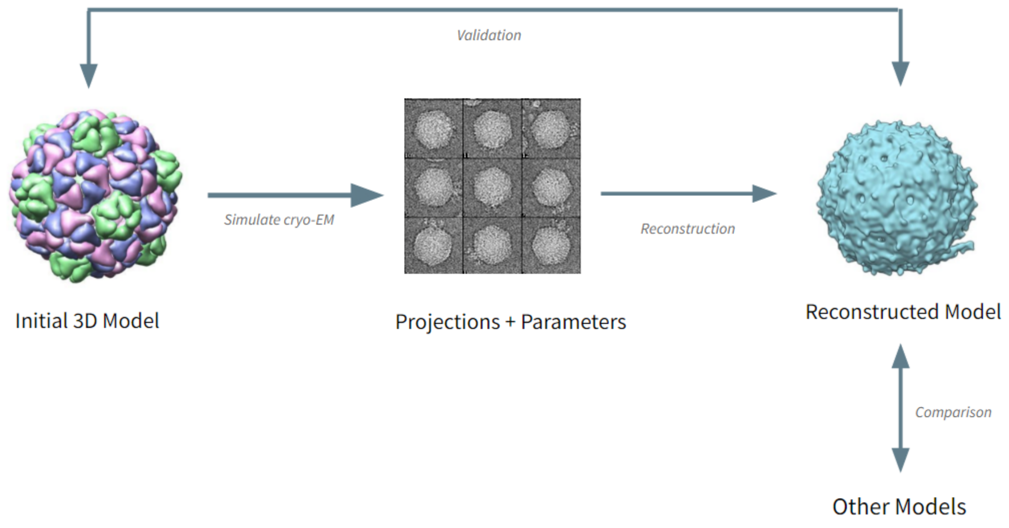


Figure 2.5: Idealized simulation and reconstruction workflow.

The three requisite parts of compSPI should come together to form an idealized, simulated cryo-EM workflow for further research and development of reconstruction methods. The researcher would start with an artificial 3D biomolecule of interest and simulate the microscopy process, the output of which is a generated stack of particle slices and generated experiment parameters.

2.2.1 TEM Engine

Our initial task was to generate example molecular data sets. Based on our sponsor's requirements, we used the TEM simulator - an open-source program intended for simulation of transmission electron microscope images and tilt series of biological samples. The program is developed at the Department of Mathematics, Stockholm University, and the Department of Cell and Molecular Biology, Karolinska Institute [11]. The simulator is based off an electron-specimen interaction model, a contrast transfer function for optics, and a noise model [12]. To generate the slices of interest these parameters are taken in combination with a .pdb file, a file format that allows for the representation of complex molecules like proteins.

The image formation model is represented through user defined parameters, which vary between samples and experiments. There also exist default parameters that are unlikely to change between simulations. These are not user-editable and include the physical elements of an example TEM setup. A full description of these parameters and common values may be found at [8].

Together these files serve as the inputs to a program that generates the particle slices of interest. TEM simulator outputs include these slices as well as metadata (E) associated with this stack guaranteeing simulation repeatability.

2.2.2 `simSPI`

While powerful, the TEM simulator does not have a user-friendly interface. It requires a cumbersome operating system dependant setup, and its input files adhere to a stringent format.

The solution is to "wrap" the TEM engine in a Python class (Figure 2.6). Users interface and generate data through this class, as opposed to running the simulator through a typical CLI approach. The simulator was containerized using Docker to simplify installation across user environments, ensuring users can focus on development instead of resolving compatibility issues.

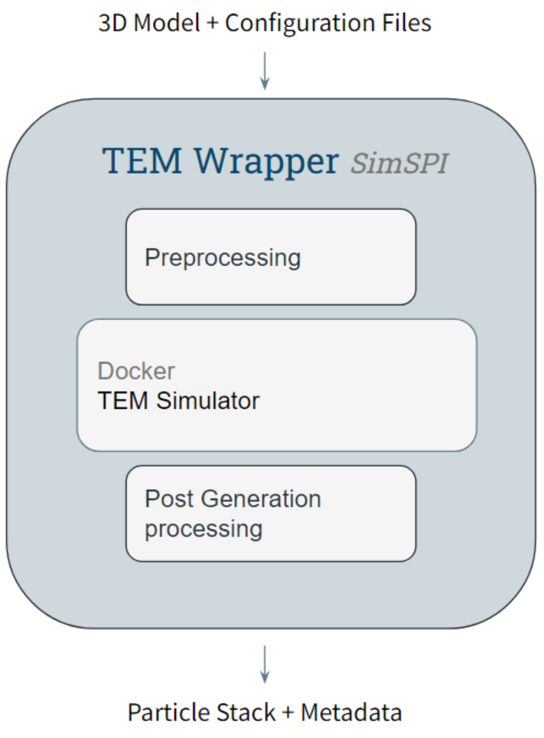

3D Model + Configuration Files
TEM Wrapper SimSPI
Preprocessing
Docker
TEM Simulator
Post Generation processing
Particle Stack + Metadata

Figure 2.6: Wrapper class structure with defined input and output.

The wrapper class is housed in the `simSPI` repository to serve as a black-box program used to generate artificial molecular data. The TEM wrapper requires a 3D particle map (.pdb), a simulation configuration file (.yaml) and a path configuration file (.yaml). The simulation configuration file contains both parameters used by TEM simulator and parameters used by the wrapper to generate simulator inputs. In the pre-processing stage, the wrapper maps the configuration files and 3D particle map into TEM simulator compatible inputs. The files generated during this process are found in Appendix D.

2.2.3 `ioSPI`

As outlined in section 1.3 one of compSPI's primary objectives is to act as a centralized repository for molecular datasets. The datasets require significant storage and compute power, making it a waste to regenerate them. Moreover, having standardized datasets would allow researchers to easily benchmark their reconstruction approaches.

OSF.io, a platform for researchers to share and organise research data, was chosen to host datasets due to:

- **Cost**
  - OSF.io is maintained by the Center of Open Science and is free to use.
- **Extensible Storage**
  - Compatible with a range of third party storage solutions.
- **Contributor Support**
  - Each project on the platform allows multiple users to contribute.

To catalogue datasets for easy retrieval and to prevent duplicates, we developed organisation guidelines to enforce a file structure (Figure 2.7). Explanation of this structure as well as the detailed formatting and accessing guidelines of `ioSPI` can be found in Appendix G.

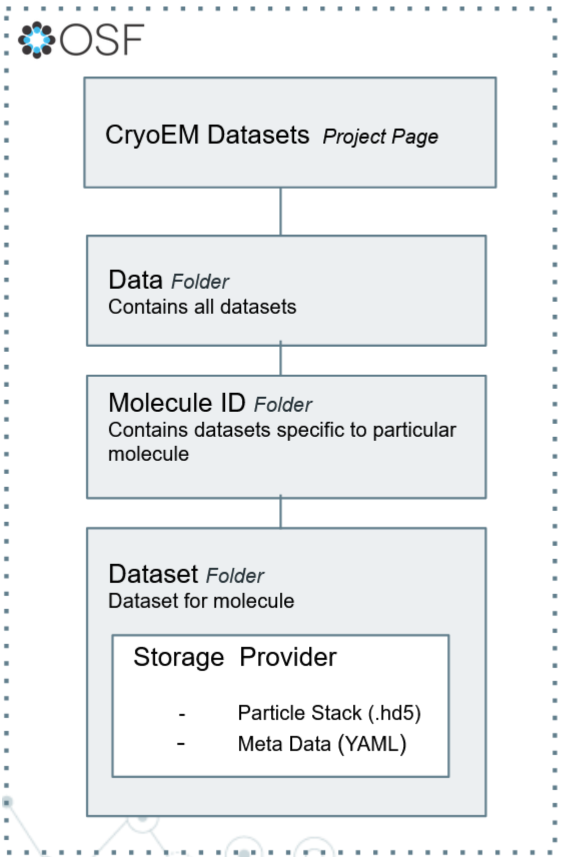


Figure 2.7: File structure for `ioSPI` datasets on OSF.io

In addition to the `OSFupload`, `ioSPI` contains additional methods for reading, writing and modifying different files such as metadata and atomic structures.

A full outline of the class including docstrings can be found at [15].

2.2.4 `reconstructSPI`

The goal of `reconstructSPI` is to provide modular reconstruction algorithms within the cryo-EM field. For the purposes of our project, the scope of this repository was limited to a basic implementation of Bayesian Expectation Maximization like [13]. This method was chosen due to its well-documented and strong performance statistics. For its robust documentation `numpy` was used in development. Future use of the `torch` library will aid in accelerating compute.

Our primary addition to the repository is the `IterativeRefinement` class, which exposes a Bayesian Expectation Maximization algorithm in the form of the `iterative_refinement` method. The class includes a suite of helper methods, which serve to modularize the stages of the iterative refinement process. This modularization ensures individual algorithmic elements are well-exposed and can be modified, tested, benchmarked, and visualized by the user.

A notable aspect of this implementation was the generation and insertion of slices. In this process a 2D $(x, y, 0)$ plane is generated and rotated about the origin. At each point on the plane a value is interpolated from the current best 3D model of the particle using `scipy`'s `map_coordinates` method. The set of these interpolated points is the "slice" generated (Figure 2.8). To insert the slice, the process is reversed: the slice is placed in a 3D space with its original rotation, and a 3D grid of points are interpolated from this slice. Points near the slice will have values similar to the slices' values, whereas points far from the slice will have values of 0. This interpolation is handled by `scipy`'s `griddata` method. Challenges with this implementation can be found in Appendix H.

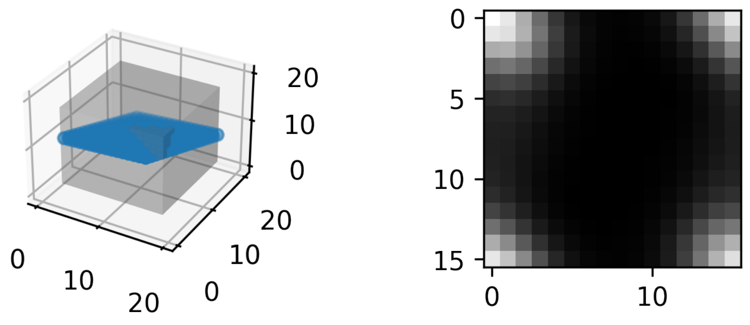


Figure 2.8: Interpolation between a 3D model and a 2D slice

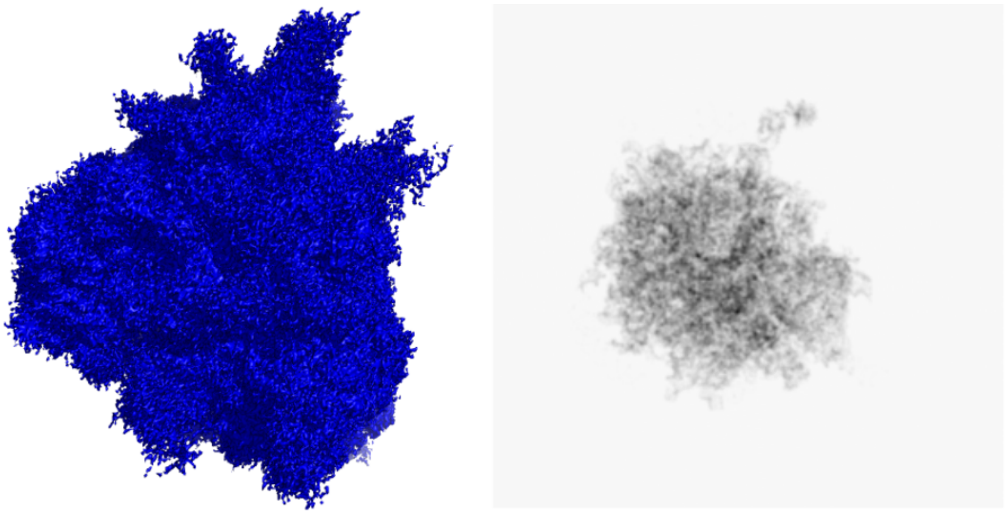
2.3 Tests & Results

We were successful in producing tools that delivered the requisite workflow (Figure 2.5). Each of the methods implemented adhere to compSPI's contribution guidelines and are documented, linted and unit-tested accordingly. Unit tests ensure that methods behave as specified in their docstrings. Ultimately compSPI's minimum 90% code-coverage requirement was exceeded with 98% and 97% coverage for `simSPI` and `ioSPI` respectively.

2.3.1 Simulated Microscopic Images

The implemented TEM simulator wrapper can successfully generate microscopic images and corresponding metadata containing rotation and parameter information for 3D structures. The wrapper is platform agnostic and currently only accepts 3D structures in .pdb format.

Since the underlying TEM simulator has been validated by its developers [12], our validation consisted of checking for errors thrown by the TEM simulator, verifying the integrity of input files produced by the wrapper and checking parameters in the log file.


(a) Human 80S ribosome [1].

(b) Simulated particle projection from ribosome.

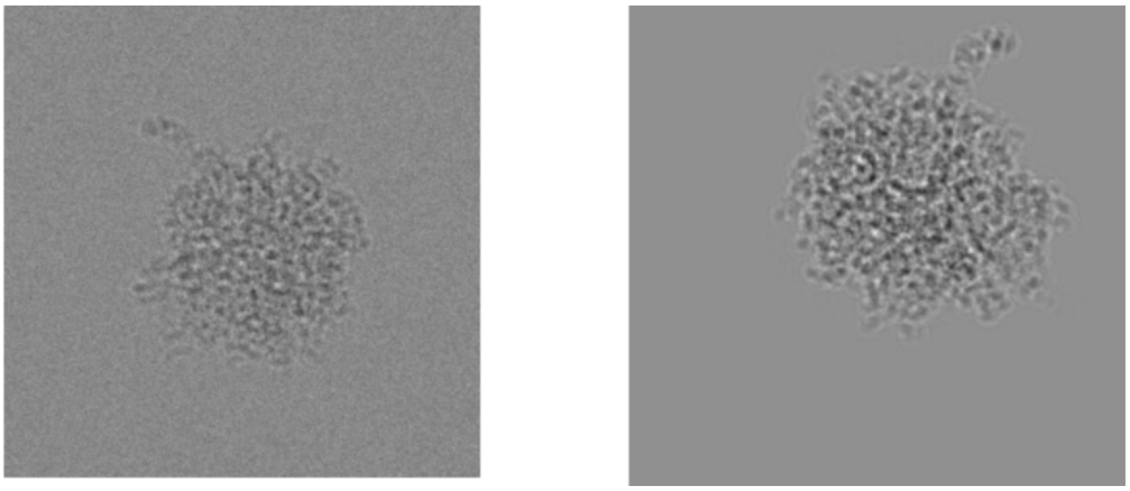
Figure 2.9: The noise-less output produced by the TEM wrapper for the Human 80S ribosome.

A successful simulation (Figure 2.9) run indicates that all necessary files (i.e.particle coordinates, defocus distributions and parameter input) were generated by the wrapper and adhered to the simulator's specifications. The veracity of added noise and lens defocus were confirmed visually (Figure 2.10). Unit tests were used to verify the input files' locations and accuracy.

To validate whether the parameters passed to and generated by the wrapper were correctly processed by the simulator, the generated log files were cross-checked against the input files.

2.3.2 Reconstruction of Molecules

Due to time and computational restrictions, reconstruction validation was sparse. A small proof-of-concept reconstruction was performed (Appendix I) that demonstrated basic functionality of our work in `reconstructSPI`.



(a) Simulated particle with Gaussian noise, $\sigma = 0.01$ [1].

(b) Simulated particle slice with defocus generation enabled.

Figure 2.10: The noise-less output produced by the TEM wrapper for the Human 80S ribosome.

It is worth noting that TEM wrapper added additional Gaussian noise and defocus distribution functionalities that are not part of the underlying simulator. However, since we used standard `torch` methods to generate required distributions, only simple unit tests were required to assess the size and range of the outputs.

3. Conclusion

At the outset of this project, we were tasked with developing the compSPI repository as a solution to the three most pressing issues facing the research of reconstruction algorithms in cryo-EM. From these issues a set of objectives for the project were derived:

I) The open source storage and generation of simulated datasets.

II) The implementation of modular versions of existing reconstruction workflows.

III) The establishment of a benchmarking and comparison suite for novel and existing reconstruction methods.

These objectives translated into deliverables, the scope of which were narrowed as we progressed through our sponsor consultations and research. This narrowed scope focused on development of a robust pipeline for the generation and storage of simulated particles with a minimal viable reconstruction suite, specifically implementing Bayesian Expectation Maximization.

Within this scope, the result of this project is a working success, and a proof of concept for the compSPI toolkit.

- Through `simSPI`, the user has the ability to generate stacks of simulated particle slices from a provided simulation configuration file and existing particle map, allowing for the simulated generation of datasets.
- `ioSPI` provides a programmatic means of classifying and uploading these generated particle stacks, storing them publicly to facilitate their use in other research projects.
- The Bayesian Expectation Maximization algorithm has been implemented in reconstructSPI, showcasing a minimum viable reconstruction of a trivial particle. This repository also houses functionality for determining the accuracy of a given reconstruction, providing the beginnings of a holistic benchmarking suite.

4. Recommendations

compSPI implements the functionality required in the goals and narrowed scope. However, there is room for more nuanced models in `simSPI` to improve the quality of simulations and workflow improvements in both `simSPI` and `ioSPI` to expand utility.

While the TEM wrapper is much more accessible than the original TEM simulator, it does not leverage the full functionality of the simulator. We believe improvements in the following would lead to more accurate simulations.

1. Specimen Model - The TEM wrapper uniformly rotates particles placed in the specimen. This should be extended to allow support for different rotation distributions which mimic different sample preparation techniques. Additionally, the current mathematical abstraction ignores different conformations that particles in the specimen may take. Such particles can be incredibly useful to test and develop more complex reconstruction algorithms.

2. Noise and CTF - The shot noise parameter is not supported in the wrapper. Enabling shot noise would allow for more realistic simulations of interactions between electrons and detectors. The TEM wrapper only supports a single defocus parameter generated uniformly or normally. Additional defocus parameters and distributions would allow for a more robust CTF model for simulated data.

3. File format - The TEM wrapper only supports 3D particle data in the form of .pdb files. Support should be extended to accommodate other file formats [2].

Adding new contributors to compSPI's datasets page on OSF.io can be a cumbersome process. New contributors must request access on both the main page as well as each folder they want to contribute to. The OSF.io API can be further leveraged to automate this process while maintaining control of write permissions to the page.

Exposing expectation maximization statistics implies a longer-term goal for molecular reconstruction - abstraction. Bringing molecular reconstruction closer to existing neural network frameworks like `keras` could allow research to focus more on granular optimization of reconstruction algorithms. Bayesian Expectation Maximization has parameters that can be tweaked and optimized, but the current focus on large mathematical paradigm shifts seems to imply that any goal of abstraction lies far in the future.

The most substantial risk for `reconstructSPI` is a lack of differentiation from existing libraries. Organizations such as RELION and CryoSPARC have robust and fast computational codebases. compSPI aims to integrate with these efforts, not compete against them. The `reconstructSPI` paradigm has an opportunity to build upon this in important ways.

5. Deliverables

The deliverables for this project comprise of contributions to the compSPI repository and the compSPI datasets page on OSF.io, namely:

- `simSPI`
  - `tem.py` The core TEM Simulator wrapper module.
  - `crd.py` Helper methods to generate and write particle coordinate information in .txt file.
  - `fov.py` Helper methods for displaying particles.
  - `tem_distributions.py` Utility file for generating and sampling distributions used in `tem.py`.
  - `tem_inputs.py` Methods to format TEM simulator inputs.
  - `tests` Unit tests for implemented modules.
  - `tem_tutoral.ipynb` A notebook demonstrating the use of `tem.py` to generate sample datasets.
  - `sim_config.yaml & path_config.yaml` Sample input configuration files for `tem.py` containing descriptions of input parameters.
- `ioSPI`
  - `datasets.py` Methods to upload and download from OSF.io.
  - `micrographs.py` Methods to read and write micrographs.
  - `tests` Unit tests for implemented modules.
  - `datasets.ipynb` Notebook demonstrating use of `datasets.py`.
- `compSPI datasets` compSPI project page on OSF.io.
- `reconstructSPI`
  - `expectation_maximization.py` Iterative refinement module with Bayesian expectation maximization.
  - `tests` Unit tests for implemented method.

Bibliography

[1] RCSB Protein Data Bank. *4V6X: Structure of the human 80s ribosome*. URL: https://www.rcsb.org/structure/4v6x.

[2] RCSB Protein Data Bank. *File download services*. URL: https://www.rcsb.org/docs/programmatic-access/file-download-services.

[3] Nelson P. C. *Physical Models of Living Systems new chapter: Single Particle Reconstruction in cryo-electron Microscopy.* 2021. URL: https://repository.upenn.edu/physics_papers/656.

[4] E. Callaway. *Revolutionary cryo-EM is taking over structural biology*. Accessed: 2022-04-07. 2020. URL: https://www.nature.com/articles/d41586-020-00341-9.

[5] Claire Donnat et al. *Deep Generative Modeling for Volume Reconstruction in Cryo-Electron Microscopy*. 2022. DOI: 10.48550/ARXIV.2201.02867. URL: https://arxiv.org/abs/2201.02867.

[6] *Fourier Shell Correlation Overview*. URL: https://blake.bcm.edu/emanwiki/EMAN2/FAQ/FSC.

[7] Tyler Heim. *Insert Slice Tolerance Demo*. 2022. URL: https://github.com/compSPI/reconstructSPI/blob/dev/notebooks/insert_slice_tolerance_demo.ipynb.

[8] Callum Hepworth. *Simulation Configuartion Parameters*. 2022. URL: https://github.com/compSPI/simSPI/blob/master/sim_config.yaml.

[9] Structura Biotechnology Inc. *cryoSPARC*. Last accessed 2022-04-10. 2022. URL: https://cryosparc.com/.

[10] Luís Henrique Simplício Ribeiro. *simSPI Tutorial*. 2022. URL: https://github.com/compSPI/simSPI/blob/master/notebooks/sim_tutorial.ipynb.

[11] H. Rullgård. *TEM-simulator*. Accessed: 2022-04-07. 2011. URL: http://tem-simulator.sourceforge.net/.

[12] H. Rullgård et al. "Simulation of transmission electron microscope images of biological specimens". In: *Journal of Microscopy* 243(3) (2011), pp. 234–256. DOI: https://doi.org/10.1111/j.1365-2818.2011.03497.

[13] S. H. W. Scheres. "RELION: Implementation of a Bayesian approach to cryo-EM structure determination". In: *Journal of Structural Biology* 180(3) (2012), pp. 519–530. DOI: 10.1016/j.jsb.2012.09.006.

[14] C. Sindelar and N. Grigorieff. "An adaptation of the Wiener filter suitable for analyzing images of isolated single particles". In: *Journal of Structural Biology* 176(1) (2011), pp. 60–74. DOI: 10.1016/j.jsb.2011.06.010.

[15] Arjun Swani. *OSF Upload*. 2022. URL: https://github.com/compSPI/ioSPI/blob/master/ioSPI/datasets.py.

[16] RELION development team. *RELION*. Last accessed 2022-04-10. 2022. URL: https://relion.readthedocs.io/en/release-3.1/.

[17] Wikipedia. *Cryogenic electron microscopy*. Accessed: 2022-04-07. 2020. URL: https://en.wikipedia.org/wiki/Cryogenic_electron_microscopy#Potential_rival_to_X-ray_crystallography.

[18] Wikipedia. *Fourier Shell Correlation*. Accessed: 2022-04-07. 2020. URL: https://en.wikipedia.org/wiki/Fourier_Shell_Correlation.

[19] D. Wrapp. "Cryo-EM structure of the 2019-nCoV spike in the prefusion conformation". In: *Science* 367(6483) (2020). DOI: 10.1126/science.abb2507.

A. Fourier Slice Theorem

A fundamental idea of how computations are simplified is found by examining the Fourier Slice Theorem:

Theorem 1. Fourier Slice Theorem. *Given some real-valued function $g : \mathbb{R}^3 \to \mathbb{R}^2$, let $G(u,v,w) = \mathscr{F}[g(x,y,z)]$ be its 3-dimensional Fourier transform. Then,*

$$G(u,v,0) = \mathscr{F}\left[\int g(x,y,z)dz\right].$$

That is, the values of G at any points on the plane $(u,v,w=0)$ are identical to the Fourier transform of the line integral in z of g at that point.

Proof. (Theorem 1) Let $g(x,y,z)$ be some real function. Then, its Fourier transform $G(u,v,w) = \mathscr{F}[g(x,y,z)]$ in its central plane is given by:

$$\begin{aligned}
G(u,v,0) &= \iiint g(x,y,z)e^{-2\pi i(xu+yv+0z)}dxdydz \\
&= \iiint g(x,y,z)e^{-2\pi i(xu+yv)}dxdydz \\
&= \iint e^{-2\pi i(xu+yv)}\left[\int g(x,y,z)dz\right]dxdy \\
G(u,v,0) &= \mathscr{F}\left[\int g(x,y,z)dz\right]
\end{aligned}$$

□

Augmenting this theorem is the following lemma:

Lemma 1. *The Fourier Slice Theorem is preserved under rotation.*

Proof. (Lemma 1) Let $g(\vec{x})$ be some real function with $\vec{x} \in \mathbb{R}^3$, and let $G(\vec{X}) = \mathscr{F}[g(\vec{x})]$. Then:

$$G(\vec{X}) = \int g(\vec{x})e^{-i\vec{x}\cdot\vec{X}}d\vec{x}$$

Now, let R be an orthonormal rotation matrix in $\mathbb{R}^3$ such that $\vec{u} = R\vec{x}$, and $\vec{x} = R^{-1}\vec{u} = R^T\vec{u}$. Then, consider the Fourier transform of $g(\vec{u})$:

$$\mathscr{F}[g(\vec{u})] = \int g(\vec{u})e^{-i\vec{x}\cdot\vec{X}}\text{det}(\mathbb{J}_R)d\vec{u}$$

Where $\mathbb{J}$ is the Jacobian for the coordinate transform of $\vec{x} \to \vec{u}$ in $\mathbb{R}^3$. Note that:

$$\text{det}(\mathbb{J}) = \text{det}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = 1$$

So that:

$$\begin{aligned} \mathscr{F}[g(\vec{u})] &= \int g(\vec{u}) e^{-i\vec{x}\cdot\vec{X}} d\vec{u} \\ &= \int g(\vec{u}) e^{-i(R^T\vec{u})\vec{X}} d\vec{u} \\ &= \int g(\vec{u}) e^{-i\vec{u}\cdot(R\vec{X})} d\vec{u} \\ \mathscr{F}[g(R\vec{x})] &= G(R\vec{X}) \end{aligned}$$

Since the Fourier transform is preserved under rotation, it follows immediately that the Fourier Slice Theorem is also. $\square$

B. Half-maps and Fourier Shell Correlation

Common to many cryo-EM reconstruction algorithms are half-maps, devised to combat noise and overfitting. The dataset is split in two "halves" prior to refinement, and each dataset is reconstructed in parallel. The result is two similar 3D particle maps ("half-maps").

Half-maps are combined to remove artefacts that arise due to pervasive noise. The process we implemented to combine half-maps is Fourier Shell Correlation (FSC) - FSCs originated in cryo-EM and are standard within the field ([18]). The FSC is defined as:

$$\text{FSC}(r) = \frac{\sum_{r_i \in r} F_1(r_i) \cdot F_2(r_i)^*}{\sqrt{\sum_{r_i \in r} |F_1(r_1)|^2 \cdot \sum_{r_i \in r} |F_2(r_i)|^2}}$$

with half-map structures F_1 and F_2 summed over voxels r_i at a given spectral radius r. This has applications in validation, and for our purposes it is used to weight the addition of shells when combining half maps, i.e.,

$$\text{3D Map}(r) = \frac{1}{2}\left(F_1(r) + F_2(r)\right) \cdot \text{FSC}(r).$$

Shells with higher FSCs contribute more to the final 3D map.

C. Bayesian Weighting Mathematics

Note that due to time constraints the Bayesian Weighting portion of the algorithm was understood but not implemented by our capstone group, rather by the sponsor Geoffrey Woollard.

The Bayesian Weighting algorithm used in reconstructSPI closely follows [3], as will this explanation. We have N particle projections X_i which have each been rotated and corrupted by noise relative to the respective "true" image of the particle A. So, we model the projections from the dataset as:

$$X_i = \mathcal{R}(\phi_i)A + \sigma\Xi_i$$

where $\mathcal{R}(\phi_i)$ is the rotation of the particle in image X_i and $\sigma\Xi_i$ is Gaussian white noise which is approximately that in the image X_i, which is taken as a known parameter.

The model for our data consists then of unknown parameters A and $\{\phi_i\}$, and the problem of finding the most probable true image A reduces to optimizing:

$$\mathscr{P}(A \mid \text{data}) = \int \mathscr{P}(A, \{\phi_i\} \mid \text{data}) d^N\phi$$

that is, the probability of true image A given the data $\{X_i\}$ is equal to the (continuous) sum of the probabilities of A and each rotation ϕ_i given the data $\{X_i\}$. Rewriting this using the Bayes formula:

$$= \int \mathscr{P}(\text{data} \mid A, \{\phi_i\}) \frac{\mathscr{P}(A, \{\phi_i\})}{\mathscr{P}(\text{data})} d^N\phi$$

This is a maximization problem so we can safely ignore the constant $\mathscr{P}(\text{data})$. The images are statistically independent, so their PDF factorizes. We assume a uniform distribution for A since we have no prior information, and since rotations are uniformly distributed each $\mathscr{P}(A, \{\phi_i\})$ is constant and can be ignored. Thus,

$$= \int \prod_{i=1}^{N} \mathscr{P}(X_i \mid A, \phi_i) d^N\phi$$

Our model for the image informs us as to the value of $\mathscr{P}(X_i \mid A, \phi_i)$: given the rotation of the true image, the only difference between it and the experimental image is the Gaussian noise Ξ.

Using this, and taking the logarithm (as maximizing f is equivalent to maximizing $\ln f$) and dropping additive constants we reduce the optimization problem to

$$L(A) = \sum_i \ln \int \gamma_i(\phi, A) d\phi$$

where the *latent probability* $\gamma_i(\phi, A)$ for image i to be rotated by ϕ is given by

$$\gamma_i(\phi, A) = e^{-(\mathcal{R}(\phi)A - X_i)^2/\left(2\sigma^2\right)}.$$

These γ_i are precisely our Bayesian weights. In practical terms, they are computed for each slice as

$$\text{weight}_i = \ln\left(\frac{-1}{2\sigma^2}\left[\| \text{ slice}_i \|^2 - 2\text{Re}\left(\text{particle} \cdot \text{slice}_i^*\right)\right]\right).$$

Do note that we have subtly added a dimension here - rather than counting i projections we count i slices and assess each projection as a 2D image which we then recombine afterward into a larger 3D map.

D. Files Generated by TEM Simulator

1. Coordinate file (.txt) : The coordinate file contains the coordinates of where the 3D structures are placed in the "specimen". The number of particles placed depend on the size of the structure and the field of view defined by the user. Since the goal of the wrapper is to generate data for reconstruction techniques and not particle picking techniques, the wrapper arranges the (rotated) particles in a symmetric grid that can be easily divided into individual particle slices.

2. Defocus file (.txt) : The defocus file contains distributions of defocus parameters used by the TEM Simulator's contrast transfer function. There is a defocus sample for every particle placed on the specimen.

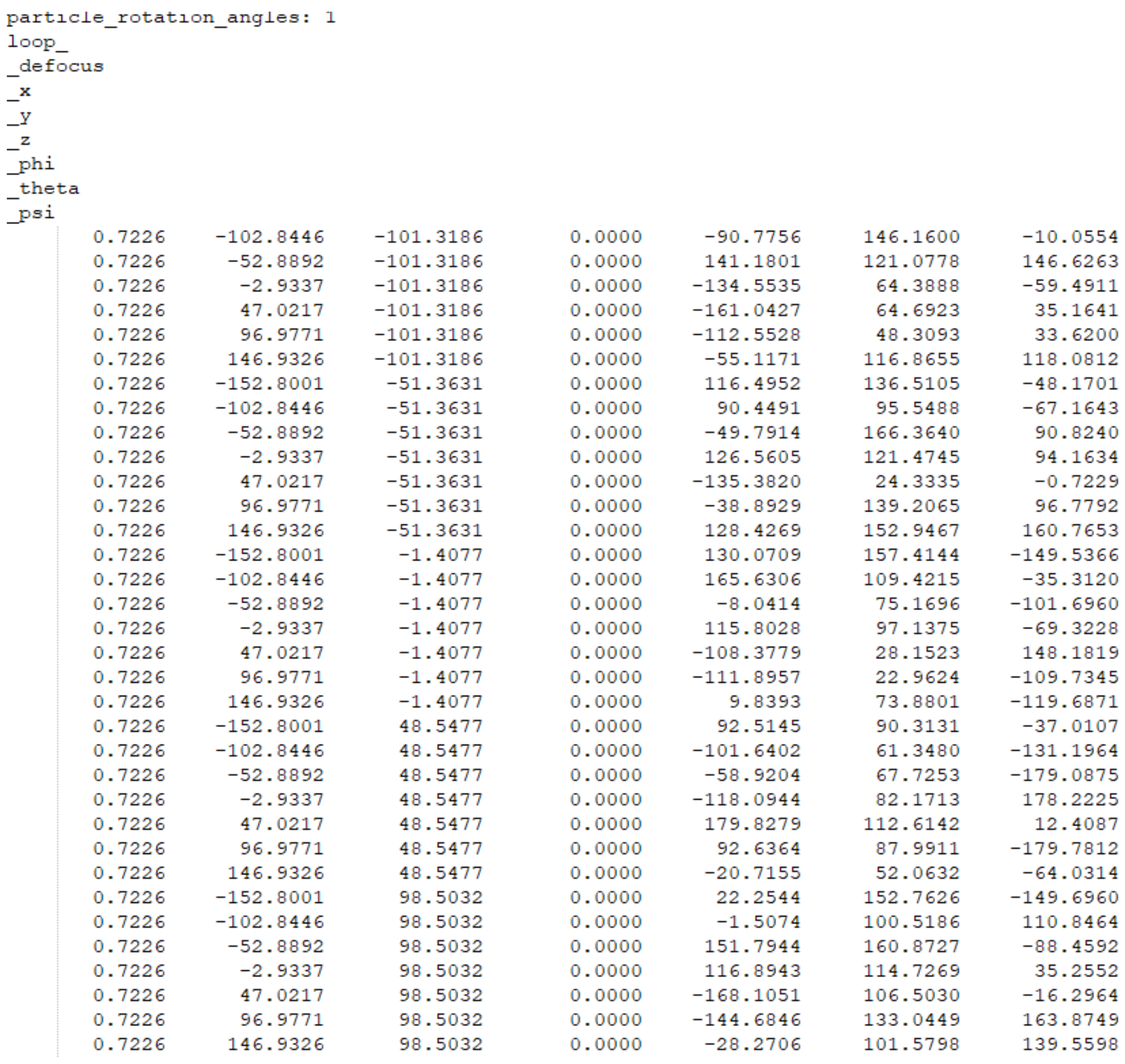
3. Input file (.inp) : The inputted and generated parameters along with file paths to relevant input files are formatted into a tabular input file that is parseable by the TEM Simulator.

After the post processing, The TEM-simulator is run. This generates a particle map (.mrc) and a log file.

The (.mrc) is post processed to extract individual particle slices. These slices are stored in (.h5) format due to its flexibility in storing a diverse range of labelled data. Further post processing includes adding white Gaussian noise to individual slices. If specified, an additional metadata file is generated that contains simulation parameters, rotation information and post-processing parameters for each individual slice.

E. Metadata Parameters

```
_seed                            9890708312
_log_file                   simSPI/temp_workspace/output/4v6x_randomrot.log

data_noise

data_sample
_hole_diameter_nm                      1200
_hole_thickness_center_nm               100
_hole_thickness_edge_nm                 100

data_particle
_particle_name                         toto
_voxel_size_nm                          0.1
_pdb_file                   simSPI/data/pdb_files/4v6x.pdb
_map_file_re_out              None_real.mrc
_map_file_im_out              None_imag.mrc

data_particleset
_particle_name                         toto
_crd_file                   simSPI/temp_workspace/output/4v6x_randomrot.txt

data_beam
_voltage_kv                             300
_energy_spread_v                        1.3
_electron_dose_e_per_nm2                100
_electron_dose_std_e_per_nm2                 0

data_optics
_magnification                        81000
_spherical_aberration_mm                2.7
_chromatic_aberration_mm                2.7
_aperture_diameter_um                    50
_focal_length_mm                        3.5
_aperture_angle_mrad                    0.1
_optics_defocusout                     None
_defocus_um                             1.0
_defocus_syst_error_um                  0.0
_defocus_nonsyst_error_um               0.0
_gen_defocus                             no
_defocus_file_in            simSPI/temp_workspace/output/4v6x_randomrot_defocus.txt

data_detector
_detector_nx_px                        5760
_detector_ny_px                        4092
_detector_pixel_size_um                   5
_average_gain_count_per_electron                 2
_noise                                   no
_detector_q_efficiency                  0.5
_image_file_out             simSPI/temp_workspace/output/4v6x_randomrot.mrc
```



```
particle_rotation_angles: 1
loop_
_defocus
_x
_y
_z
_phi
_theta
_psi
      0.7226   -102.8446   -101.3186      0.0000    -90.7756    146.1600    -10.0554
      0.7226    -52.8892   -101.3186      0.0000    141.1801    121.0778    146.6263
      0.7226     -2.9337   -101.3186      0.0000   -134.5535     64.3888    -59.4911
      0.7226     47.0217   -101.3186      0.0000   -161.0427     64.6923     35.1641
      0.7226     96.9771   -101.3186      0.0000   -112.5528     48.3093     33.6200
      0.7226    146.9326   -101.3186      0.0000    -55.1171    116.8655    118.0812
      0.7226   -152.8001    -51.3631      0.0000    116.4952    136.5105    -48.1701
      0.7226   -102.8446    -51.3631      0.0000     90.4491     95.5488    -67.1643
      0.7226    -52.8892    -51.3631      0.0000    -49.7914    166.3640     90.8240
      0.7226     -2.9337    -51.3631      0.0000    126.5605    121.4745     94.1634
      0.7226     47.0217    -51.3631      0.0000   -135.3820     24.3335     -0.7229
      0.7226     96.9771    -51.3631      0.0000    -38.8929    139.2065     96.7792
      0.7226    146.9326    -51.3631      0.0000    128.4269    152.9467    160.7653
      0.7226   -152.8001     -1.4077      0.0000    130.0709    157.4144   -149.5366
      0.7226   -102.8446     -1.4077      0.0000    165.6306    109.4215    -35.3120
      0.7226    -52.8892     -1.4077      0.0000     -8.0414     75.1696   -101.6960
      0.7226     -2.9337     -1.4077      0.0000    115.8028     97.1375    -69.3228
      0.7226     47.0217     -1.4077      0.0000   -108.3779     28.1523    148.1819
      0.7226     96.9771     -1.4077      0.0000   -111.8957     22.9624   -109.7345
      0.7226    146.9326     -1.4077      0.0000      9.8393     73.8801   -119.6871
      0.7226   -152.8001     48.5477      0.0000     92.5145     90.3131    -37.0107
      0.7226   -102.8446     48.5477      0.0000   -101.6402     61.3480   -131.1964
      0.7226    -52.8892     48.5477      0.0000    -58.9204     67.7253   -179.0875
      0.7226     -2.9337     48.5477      0.0000   -118.0944     82.1713    178.2225
      0.7226     47.0217     48.5477      0.0000    179.8279    112.6142     12.4087
      0.7226     96.9771     48.5477      0.0000     92.6364     87.9911   -179.7812
      0.7226    146.9326     48.5477      0.0000    -20.7155     52.0632    -64.0314
      0.7226   -152.8001     98.5032      0.0000     22.2544    152.7626   -149.6960
      0.7226   -102.8446     98.5032      0.0000     -1.5074    100.5186    110.8464
      0.7226    -52.8892     98.5032      0.0000    151.7944    160.8727    -88.4592
      0.7226     -2.9337     98.5032      0.0000    116.8943    114.7269     35.2552
      0.7226     47.0217     98.5032      0.0000   -168.1051    106.5030    -16.2964
      0.7226     96.9771     98.5032      0.0000   -144.6846    133.0449    163.8749
      0.7226    146.9326     98.5032      0.0000    -28.2706    101.5798    139.5598
```

Figure E.1: Metadata parameters generated after a simulation run.

F. Wiener Filter Tuning

The process we used for tuning the Wiener Filter closely followed section 2.6 of [14]. The goal, as in the paper, was to compute the spectral signal-to-noise ratio of Fourier shells and use the inverse of this as the small number for the CTF Wiener Filter.

The process used the following algorithm, with "true" images in Fourier space $X^{(i)}$ with corresponding CTFS CTF$^{(i)}$ evaluated at voxels $\vec{s}$ within shells of radius $R = |\vec{s}|$:

1. Compute $\hat{M}(\vec{s})$ a least squares estimate of the signal value for each data point:

$$\hat{M}(\vec{s}) = \frac{\sum_{i=1}^{N}\left(\text{CTF}^{(i)} X^{(i)}\right)}{\sum_{i=1}^{N}\left(\text{CTF}^{(i)}\right)^2}$$

2. Perform a weighted estimation of signal value variance over the Fourier shells of radius R:

$$\hat{\sigma}_{Rs}^2 = \frac{1}{\sum_R \sum_{i=1}^{N} \text{CTF}^{(i)}(\vec{s})^2} \sum_R \sum_{i=1}^{N} \text{CTF}^{(i)}(\vec{s})^2 |X^{(i)}(\vec{s})|^2$$

3. Using these signal estimates compute the unbiased noise estimator:

$$\hat{\sigma}_{Rn}^2(R) = \frac{\sum_R \sum_{i=1}^{N} \left| X^{(i)}(\vec{s}) - \text{CTF}^{(i)}(\vec{s}) \hat{M}^{(i)}(\vec{s}) \right|^2}{n_R(N-1)}$$

4. Compute the unbiased SSNR of the raw images:

$$\text{SSNR}_{\text{no CTF}}(R) \approx \frac{\hat{\sigma}_{Rs}^2(R)}{\hat{\sigma}_{Rn}^2(R)} - \frac{n_R}{\sum_R \sum_{i=1}^{N} (\text{CTF}_i(\vec{s}))^2}$$

Then, expanding the radial values of SSNR$_{\text{no CTF}}(R)$ into 2-dimensional Fourier shells allowed us to apply more accurate wiener filtration to the entirety of our image space.

The algorithm above inherently leaves zeros in several places. This is not ideal both because it would produce a singularity and because the original intent of the small number added to the Wiener Filter was to prevent singularities. So, to combat this these singularities were replaced by 0.01, a "typical" small number used in naive cryo-EM Wiener filters that should prevent any highly erroneous effects from emerging.

G. Structure of File Storage in OSF.io

The samples for a biological structures are stored in a folder with a unique *molecule ID*. This folder is divided into sub-directories, each corresponding to unique datasets.

Any compSPI user can download hosted datasets. The datasets can be accessed through the OSF.io webpage itself which provides a graphical file tree as well as search functionalities. Each dataset can be searched by its molecule ID and other tags (specified while uploading). The existing OSF.io API also provides programmatic dataset search and download utilities.

Only verified contributors have write permissions. Datasets are uploaded via the `OSFUpload` module in the `ioSPI` repository. The module abstracts the OSF.io API while adding additional constraints to enforce the dataset page's file structure. Further, it provides the functionality to tag uploaded data (used for searching and navigation). The module requires an API token to use which is validated by OSF.io for appropriate permissions.

H. Computational Challenges With Slice Interpolation

The first problem that emerged under the paradigm outlined in 2.2.4 was that of rotational centering. Due to how particle data is represented at the request of the sponsor, it is impossible to have slice coordinates centered such that there is an equal number on either side. The consequence of this is that when rotated, the boundaries of the coordinate space change. In effect, this means that when generating a slice there is an invariable "dead-zone" along the edges of the coordinate space where the slice and map values will always be zero. The alternative to this was to shift the slice coordinates such that they were rotated about a central axis and shift back after rotation. While mathematically sound, this solution introduced errors in how Fourier shells were generated (with an odd number of pixels diameter) and unreliability in how a particular map might have its coordinates represented. Because particles in Fourier space typically have near-zero values at the edges of their coordinates, the "un-shifted" solution was preferred and implemented.

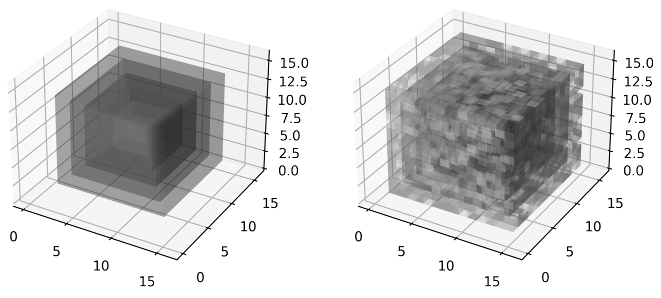
The second problem that emerged was one of 3-dimensional interpolation from a 2-dimensional object. This discussion is outlined in a Jupyter Notebook [7] that may be helpful. Interpolating 2D to 3D is an ill-defined request: 3D coordinates infinitely "near" a 2D plane would still not lie upon that plane, and so might be interpolated as zero. Computers store decimal numbers as "floats", which have an inherent imprecision that directly implies that even though a 3D point may "mathematically" lie on a 2D plane, it will not be taken to lie upon that plane. The solution to this problem is a region of tolerance about the plane: points within this region would still be taken as "on" the plane despite that not technically being the case. Our implementation of this functioned by shifting copies of the 2D plane to some small $\pm z_0$ values before rotating. Now, the plane has depth, and interpolation is well-defined. In particular, `scipy`'s `griddata` method will not accept the depth-less 2D plane as an interpolation base, and this added z-depth resolves a thrown error.

I. Reconstruction of a Basic Cubic Particle

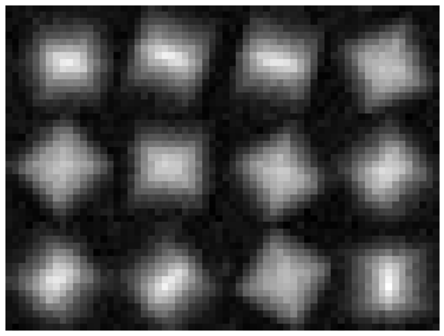
As a proof-of-function for the library, a simple cubic particle was reconstructed in a local Jupyter Notebook. The results of this can be seen in Figure I.1.

To gauge the similarity and repeatability of the reconstruction results, we generated FSC (Fourier shell correlation) curves between the two half maps produced after the final iteration and the completer map and the target particle (Figure I.2). The equation used for the FSC process is in Section B.

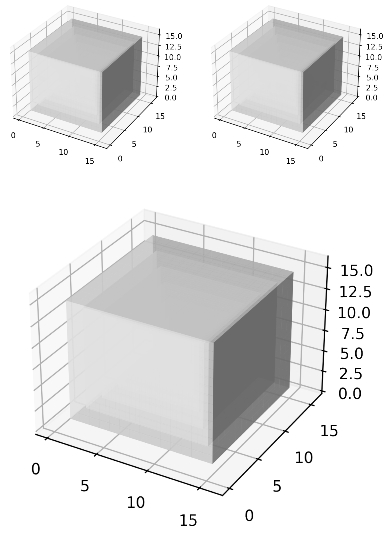
The sample size for the curve was unrealistically small, with only 6 samples per half-map due to limited computing power. This led to an unreliable final result as the reconstruction algorithm lacked ground-truth data for many sections of the 3D map. The small sample size also affects the quality of the reconstruction process which is apparent in the FSC between the final 3D map and the target particle. Slice data is concentrated at the center of Fourier space; reconstructions are most reliable at lower spectral radii. Since the FSC curve does not indicate complete accuracy at the minimum spectral radius, the dataset was not large enough to produce reliable results. However, since the FSC agrees up to a certain resolution for independently reconstructed half maps, it does indicate a level of robustness for the reconstruction algorithm.



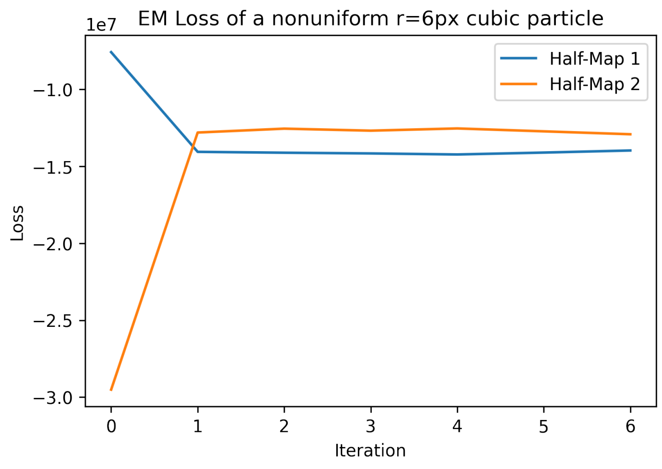
(a) Target "particle" (left) and noisy "guess" (right).



(b) Simulated cryo-EM-like projections.



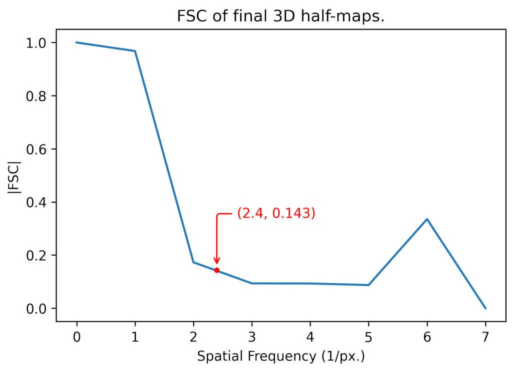
(c) Final half-maps (top), combined map (bottom).



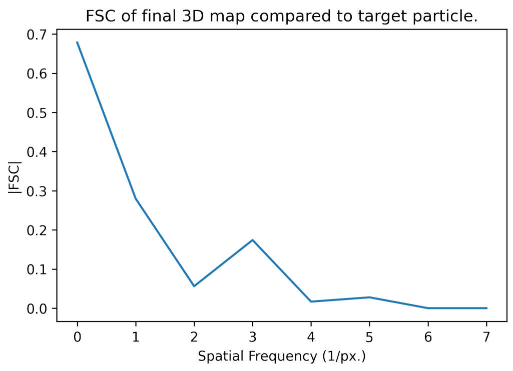
(d) Loss graph for reconstruction of cubic particle.

Figure I.1: Demonstration reconstruction of a simple cubic particle.

Severe ($\sigma^2 = 20$) Gaussian noise was applied to the target particle to generate the "guess" particle. Simulated particle projections had added Gaussian noise and CTF-induced artefacts. Reconstruction was performed using a ground-truth sample of 12 simulated particles (b).



(a) FSC graph of reconstructed half maps.



(b) FSC graph of combined map and target particle.

Figure I.2: Fourier Shell Correlation curves of reconstructed simple cubic particles.

The resolution sampled by the 3D half maps is 0.41 px using the 0.143 spatial frequency cut-off as in [6]. The odd shape of the correlation curves is a consequence of the small dataset used for reconstruction.