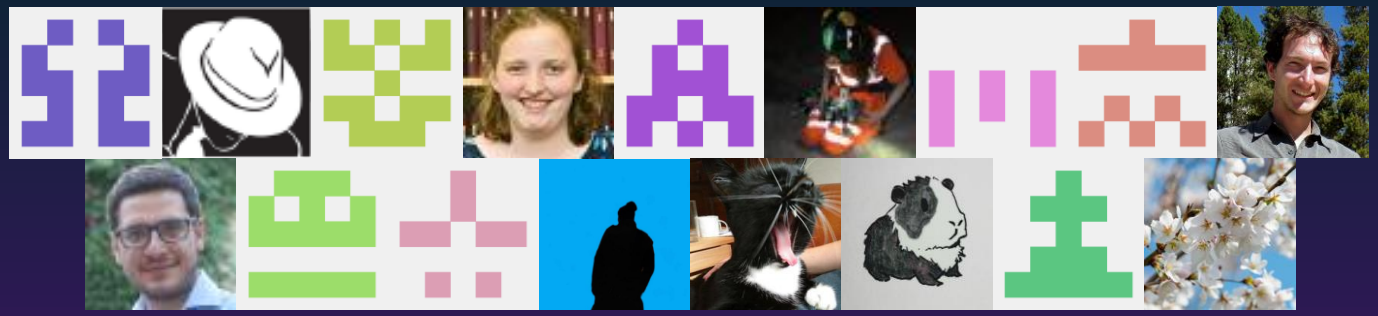
Science and Technology Facilities Council

Scientific Computing

CIL

CORE IMAGING LIBRARY

# CIL v25.0.0



# CIL v25.0.0 – Imminent

- Major release
- New features
  - Initial implementation of Cone3D_Flex geometry
- Enhancements :
  - Performance improvements
  - Useability updates
  - Build improvements
- Backward compatibility breaking changes

- Changelog
  - Full list of new features, enhancements, bug fixes, and breaking changes
- Documentation
  - Updated tutorials, API references, and examples
- Installation instructions:
  - Step-by-step guide for installing and building CIL



Warning sign icons created by SyafriStudio - Flaticon

# Installation

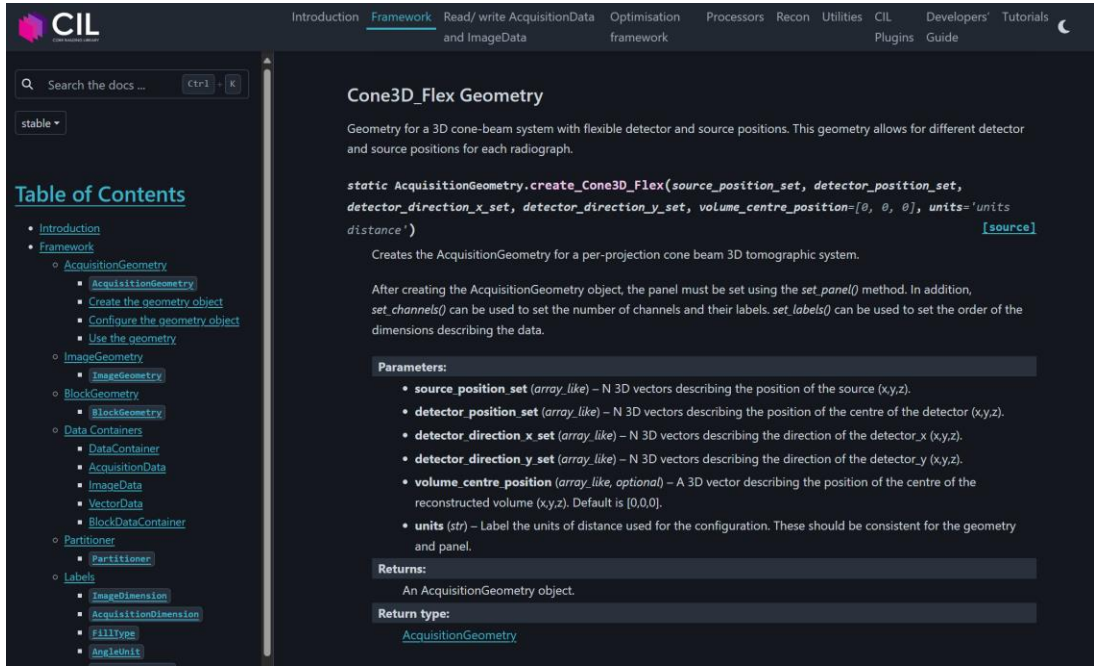
```
conda create --name cil -c https://software.repos.intel.com/python/conda -c conda-forge -c ccpi cil=25.0.0
```

```
conda env create -f https://tomographicimaging.github.io/scripts/env/cil_demos.yml
```

- Reduced mandatory run requirements in favour of documenting tested combinations
- Added environment files for common sets of packages
- Still working towards:
  - `conda install ccpi:cil`
  - `Pip install cil`

# Cone3D_Flex: CIL v25.0.0

- New Geometry Type: Enables reconstruction from non-circular trajectories.
- ASTRA Backend Support: Integration with `ProjectionOperator` and `FBP`.
- Visualisation Tools: `show_system_positions`
- Impact: Opens up new experimental setups and reconstruction possibilities i.e Tomosynthesis, Helical, multi-resolution CT



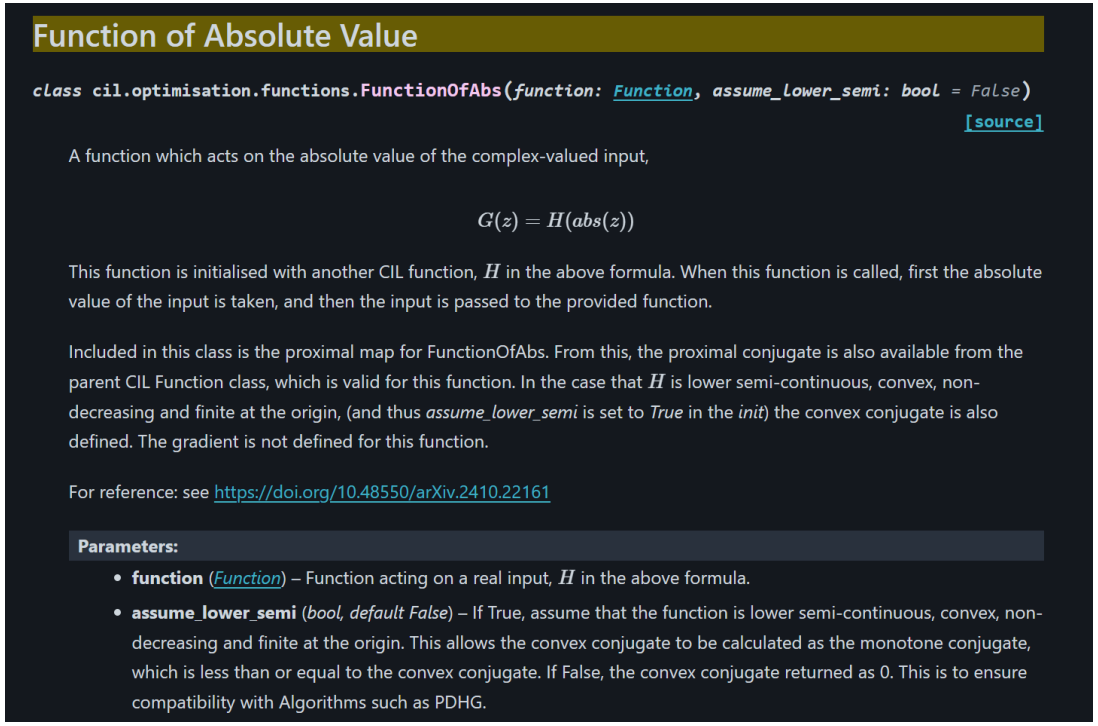
# Cone3D_Flex: WIP

- TIGRE support not included in this release (CIL work, not TIGRE)
- Many CIL processors on the to-do list will raise a `NotImplementedError`
- The `NexusDataWriter` and `NexusDataReader` currently won't work
- If you use `Cone3D_Flex` and things behave unexpectedly, please talk to us or raise an issue

- Let us know how you're using it and we will support you!

# Optimisation Framework

- Added `FunctionOfAbs` for complex data support
- Accelerated Proximal Gradient Descent (APGD)
  - Allows custom momentums
  - Current options: Nesterov momentum and constant momentum
- Improved consistency of `step_size` across GD, ISTA, FISTA, APGD
- PDHG now allows dual variable initialisation and updated convergence checks



## Function of Absolute Value

*class* **cil.optimisation.functions.FunctionOfAbs**(*function: Function, assume_lower_semi: bool = False*)

[source]

A function which acts on the absolute value of the complex-valued input,

$$G(z) = H(abs(z))$$

This function is initialised with another CIL function, $H$ in the above formula. When this function is called, first the absolute value of the input is taken, and then the input is passed to the provided function.

Included in this class is the proximal map for FunctionOfAbs. From this, the proximal conjugate is also available from the parent CIL Function class, which is valid for this function. In the case that $H$ is lower semi-continuous, convex, non-decreasing and finite at the origin, (and thus *assume_lower_semi* is set to *True* in the *init*) the convex conjugate is also defined. The gradient is not defined for this function.

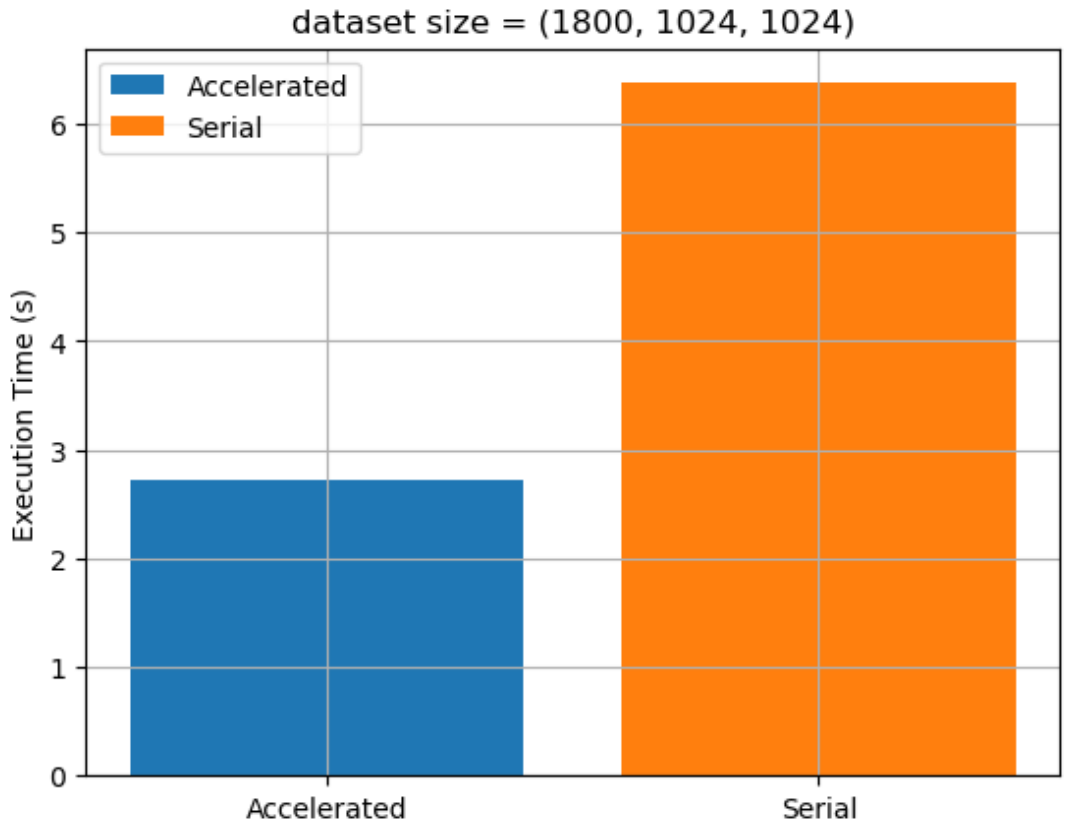
For reference: see https://doi.org/10.48550/arXiv.2410.22161

**Parameters:**

- **function** (*Function*) – Function acting on a real input, $H$ in the above formula.
- **assume_lower_semi** (*bool, default False*) – If True, assume that the function is lower semi-continuous, convex, non-decreasing and finite at the origin. This allows the convex conjugate to be calculated as the monotone conjugate, which is less than or equal to the convex conjugate. If False, the convex conjugate returned as 0. This is to ensure compatibility with Algorithms such as PDHG.

# Performance & Usability Enhancements

- Accelerated `TransmissionAbsorption` with numba
- Improved `Normaliser` performance and RAM use:
  - 512x2048x2048  11.5s -> 2.6s
  - Very low RAM use compared to inputs



# Developer Experience

- Build changes:
  - Move from CMake to  pip install for installing into environment with dependencies
  - Working towards ‘pip install cil’
- Windows Support:
  - Build and test environment files
  - Windows CI with GitHub actions
- Testing Enhancements: Full CI matrix via GitHub UI, new unit tests, and Windows test environment setup.

# Breaking Changes: Random methods

- The random methods to populate a data container can now be accessed via :
  - `geometry.allocate('RANDOM')`
  - `geometry.allocate('RANDOM_INT')`
  - `container.fill('RANDOM')`
  - `container.fill('RANDOM_INT')`

- To access the old methods where this behaviour is unchanged, call
  - `geometry.allocate('RANDOM_DEPRECATED')`
  - `geometry.allocate('RANDOM_INT_DEPRECATED')`

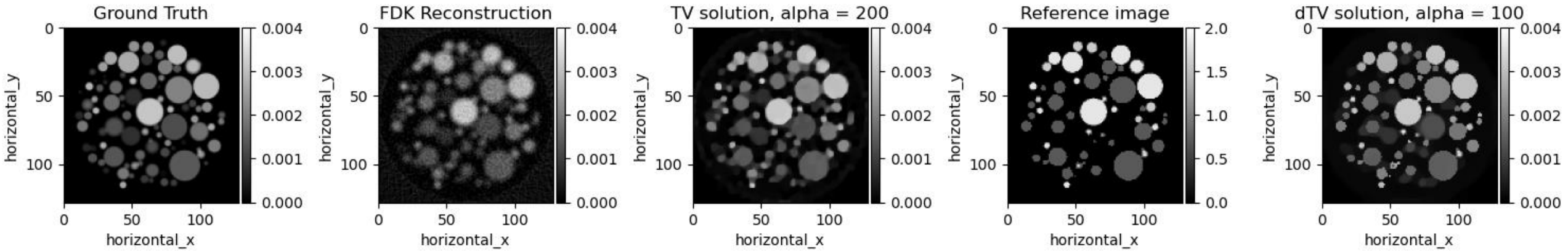
- The default memory footprint of these methods is now much lower because we use NumPy random number generators

# Breaking Changes: Seeding methods

- You can no longer set a random seed externally:
  - `numpy.random.seed() -> geometry.allocate('RANDOM', seed=5)`
- Your array values for a set seed will now change compared to previous versions

# Documentation: CIL-Demos

- Documentation improvements
  - Clearer examples and docstrings
- WIP: Automatic notebook testing
- New deep-dive demos:
  - 06_directional_TV.ipynb
  - 07_flexible_geometry.ipynb

# Documentation: Tutorials

CIL CORE IMAGING LIBRARY

Science and Technology Facilities Council

Scientific Computing

CIL | Introduction | Framework | Read/ write AcquisitionData and ImageData | Optimisation framework | Processor

Search the docs ... Ctrl + K

nightly

## Table of Contents

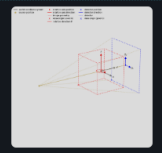


Edit on GitHub

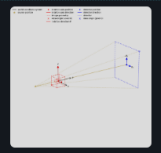
Tutorials

# Tutorials

## Load and visualise data

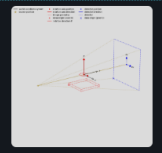


Load and visualise data with ZeissDataReader



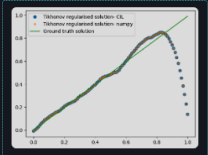
Load and visualise data with NikonDataReader

## Geometry



A detailed look at CIL geometry

## Advanced topics



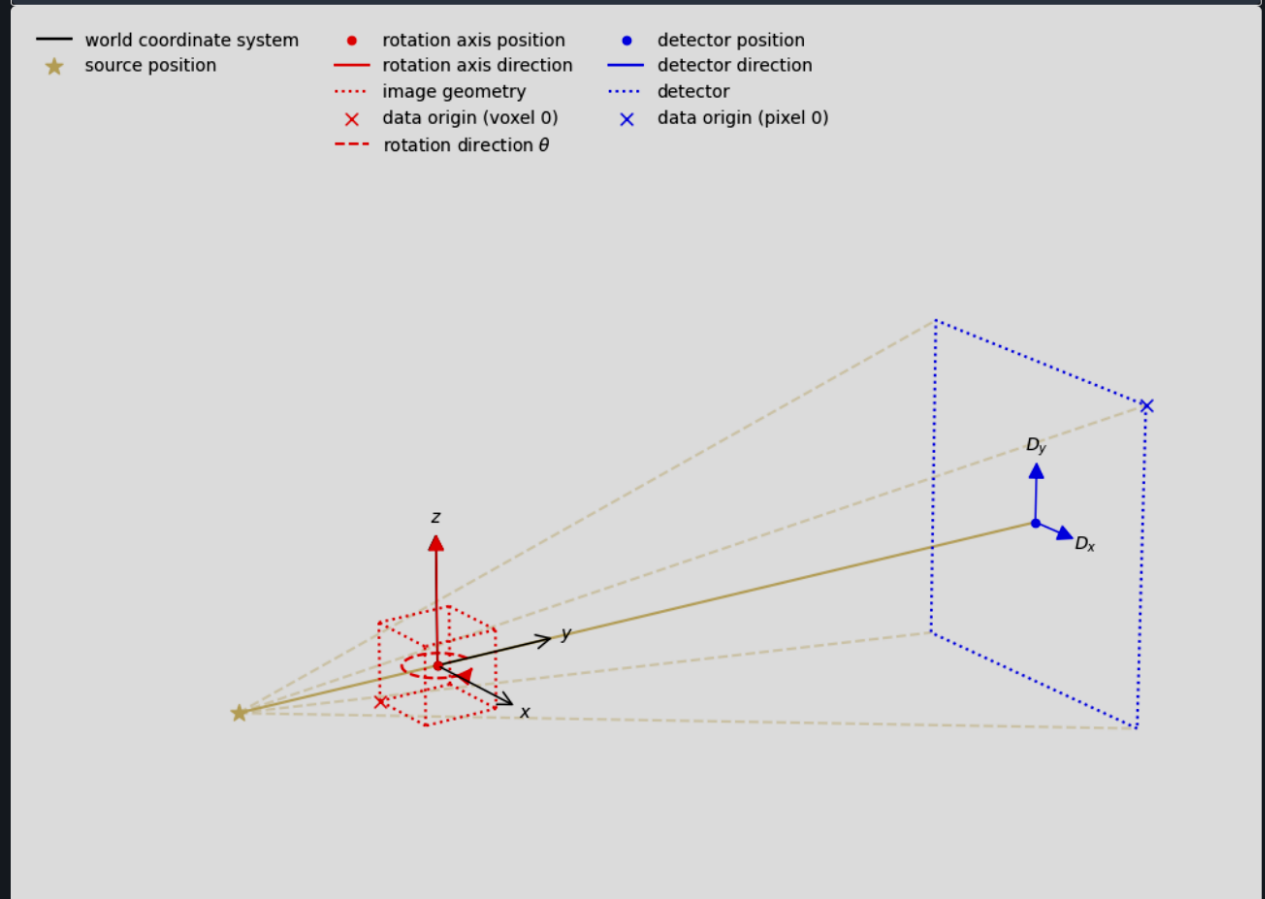
1D inverse problem demo using deriv2


CIL Callbacks How To

```
[3]: from cil.io import NikonDataReader
     file_name = '../data/korn/Korn i kasse/47209 testscan korn01_recon.xtekct'
     data_reader = NikonDataReader(file_name=file_name)
     data = data_reader.read()
```

Check the data has been loaded correctly by viewing the geometry with `show_geometry()` to display information about the source and detector setup.

```
[4]: from cil.utilities.display import show_geometry
     show_geometry(data.geometry)
```



# CIL v25.1.0 plan...

- NumPy 2.0 support
- TIGRE support for Cone3D_Flex
- Wrappers for TIGRE algorithms
- Improve ASTRA wrappers
- Continue simplification of dependencies
- MacOS CI

- Community contributions are very welcome
- We can work with you to get your code into CIL.
- Readers hackathon in November!

# 🎉 Thank you to everyone who 🎉 contributed to the v25.0.0 release! 🎉

CIL CORE IMAGING LIBRARY
UKRI Science and Technology Facilities Council
Scientific Computing

- Special thanks to our new contributors
  - **@effepivi** – key work on the new Cone3D_Flex geometry and ASTRA integration (#2039)
  - **@emmanuel-ferdman** – fixed deprecation warnings for rtol and atol in GD (#2056)
  - **@M-A-Demir** – made ProjectionOperator device input case-insensitive (#2065)
  - **@hsw43** – added dual variable initialisation for PDHG (#2169)
  - **@hussam-stfc** – added a flag to show_geometry to disable plt.show() (#2195)
  - **@fmwatson** – added AbsFunctions class (#1976)
  - **@purepani** – contributed to the build system overhaul, including PyPI packaging and environment setup (multiple PRs, e.g. #2097)
- Thanks also to:
  - **@epapoutsellis** – split FISTA and APGD to provide more momentum options (#2061)
  - **@samtygier-stfc** – fixed argument order for subtraction and division between different container types (#2133)