additional_exercises_data_resources_tmp

August 19, 2025

[1]:
```python
import warnings
warnings.simplefilter('error', RuntimeWarning)
```

[2]:
```python
# -*- coding: utf-8 -*-
#  Copyright 2019 - 2024 United Kingdom Research and Innovation
#  Copyright 2019 - 2024 Technical University of Denmark
#  Copyright 2019 - 2022 The University of Manchester
#
#  Licensed under the Apache License, Version 2.0 (the "License");
#  you may not use this file except in compliance with the License.
#  You may obtain a copy of the License at
#
#      http://www.apache.org/licenses/LICENSE-2.0
#
#  Unless required by applicable law or agreed to in writing, software
#  distributed under the License is distributed on an "AS IS" BASIS,
#  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#  See the License for the specific language governing permissions and
#  limitations under the License.
#
#   Authored by:    Jakob S. Jørgensen (DTU)
#                   Laura Murgatroyd (UKRI-STFC)
#                   Margaret Duff (UKRI-STFC)
```

1 Additional open-ended exercises, datasets and resources

To further experiment with tomographic reconstruction and data processing using CIL we provide a few additional data sets, suggestions for experiments and links to CIL resources including publications, documentation and demos.

1.1 Data sets

The following data sets have been downloaded and are available on the STFC Cloud shared drive (read only).

Most of the data sets provided are 3D. For faster execution times in the experiments we suggest to extract a single slice and work only with such a 2D data set. From the loaded 3D `AcquisitionData` object, `data3D`, a 2D slice `AcquisitionData` can be extracted using the `get_slice` method, for example

```
data2D = data3D.get_slice(vertical=27)
```

will extract vertical slice number 27 as a new 2D `AcquisitionData` object, `data2D`.

For Cone3D cone-beam datasets only the central slice can be extracted as a Cone2D (fan-beam) dataset, this is done using

```
data2D = data3D.get_slice(vertical='centre')
```

If an even number of slices are available in `data3D`, then the above command will produce a slice that is interpolated from the two central slices.

Nikon datasets can be loaded using the CIL `NikonDataReader` in the `cil.io` module.

Zeiss datasets can be loaded using the CIL `ZEISSDataReader` in the `cil.io` module.

Crystals in clay:

Cone-beam Nikon data set of crystals in clay from, described at

https://zenodo.org/record/4912635

SophiaBeads:

Cone-beam Nikon data set of glass beads. Data sets with different trade-offs between numbers of projections and exposure time are available, the 256-projection data set is available on the STFC Cloud. Description at

https://zenodo.org/record/16474

LEGO laminography dataset:

A rotary laminography tomography dataset of a sample of LEGO brick acquired on a Nikon micro-CT instrument, described at

https://zenodo.org/record/2540509

HDTomo datasets:

Six Zeiss cone-beam data sets including the walnut dataset, as well as Kinder Surprise chocolate eggs, a USB stick and more, with descriptions at

https://zenodo.org/record/4822516

Sandstone:

The parallel-beam sandstone synchrotron data set, both a small extracted data set and the full projections are available from

https://zenodo.org/record/4912435

Additional tomography data resources NOT on the STFC Cloud FIPS data: A collection of X-ray CT data sets is available from the Finnish Inverse Problems Society:

https://www.fips.fi/dataset.php

1.2 Suggestions for experiments

Try reconstructing different real data sets:

Choose one or more of the datasets listed above. Load the dataset, determine and carry out any preprocessing required, and compute an FBP or FDK reconstruction. Try also to reconstruct using your favourite iterative/regularized reconstruction method. Apply CIL processors to preprocess data as necessary, possibly after having added noise, artifacts or reduced the datasets yourself.

Relevant notebooks: 01_optimisation_gd_fista.ipynb, Week1/01_intro_walnut_conebeam.ipynb, Week1/02_intro_sandstone_parallel_roi.ipynb

Synthetic data:

Try out loading and generating simulated data from phantoms provided by CIL (cil.utilities.dataexample) or the TomoPhantom CIL plugin (cil.plugins.TomoPhantom). Choose either full data or incomplete data of your choice, add noise, and reconstruct first using FBP and then using regularised reconstruction methods.

Relevant notebooks: 01_optimisation_gd_fista.ipynb, 02_tikhonov_block_framework.ipynb, Week1/03_preprocessing.ipynb

Reduced data reconstruction:

Use CIL processors (e.g. Slicer or Binner or Masker/MaskGenerator) to remove parts of or down-sample data sets, for example to obtain a reduced number of projections, a limited angle problem, truncated projections (region of interest data), exterior problem, etc. Compare for example different regularised reconstruction methods at increasingly few projections.

Relevant notebooks: 01_optimisation_gd_fista.ipynb, 03_preprocessing.ipynb

Denoising, deblurring, inpainting:

CIL is developed for tomography but can handle general (at present, linear) inverse problems. Using IdentityOperator, BlurringOperator and MaskOperator provided by CIL it is possible to set up denoising, deblurring and inpainting problems. Choose one or more of these problems and a test image, simulate some data, choose a regularised recontruction problem and compute reconstructions.

Relevant notebooks: Week3/01_Color_Processing.ipynb

Effect of regularisation parameter:

Run Tikhonov and TV-regularized reconstruction with a wide range of different values for the regularisation parameter to see the effect on the reconstruction, ranging from under- to over-regularised.

Relevant notebooks: 01_optimisation_gd_fista.ipynb

Anisotropic regularisation:

Normally we use the same regularisation in all spatial dimensions. Sometimes we may have an image with different behaviour in different dimensions, for example smooth in the y-dimension but edges in the x-direction. Using CIL BlockOperators it is possible to use different regularisers in different dimensions, for example a FiniteDifferenceOperator in y and an IndentityOperator (or no regularisation) in x, or FiniteDifferenceOperators in both x and y but having different regularisation parameters. Implement such anisotropic regularisation in a Tikhonov formulation and demonstrate the effect on a synthetic data reconstruction problem of your choice.

Relevant notebooks: 01_optimisation_gd_fista.ipynb

Verify algorithms against each other:

Compare FISTA and PDHG for solving the same problem, such as TV-regularised or L1-norm regularised least squares. As the same optimisation problem is specified, the different algorithms should produce the same solution, when converged. Try to confirm whether they produce the same solution. You may need to run a large number of iterations. You can also compare with the smoothed TV regulariser, in which case the optimisation problem is smooth and so can be solved using the gradient descent algorithm.

Relevant notebooks: 01_optimisation_gd_fista.ipynb

Compare convergence speed of PDHG using different step sizes:

The sigma and tau step sizes in PDHG can have a dramatic influence on the convergence speed. Experiment with different choices (that must satisfy the constraint specified) and compare the convergence speed. Try on different test problems, including synthetic and real data, and see if there is a trend for the best choice of step sizes across data sets, or it is data set dependent.

Relevant notebooks: 03_PDHG.ipynb

SPDHG subsets and probabilities:

In SPDHG we need to specify the number of subsets to use and the probabilities with which to choose each subset and the regulariser. Experiment with different numbers of subsets and probabilities and compare the effect on reconstruction quality and speed.

Relevant notebooks: 04_SPDHG.ipynb

Other algorithms, operators and functions:

Explore other tools offered by CIL such as the LADMM algorithm, TGV regularisation and weighted least squares and Kullback-Leibler divergence data fidelities, set up test problems and try out new algorithms and optimisation problems and compare the results with problems previously solved.

Relevant notebooks: Week3/01_Color_Processing.ipynb

1.3 Resources

CIL documentation

https://tomographicimaging.github.io/CIL/

Main CIL GitHub repository

https://github.com/TomographicImaging/CIL

CIL demos and training material repository

https://github.com/TomographicImaging/CIL-Demos

Core Imaging Library – Part I: a versatile Python framework for tomographic imaging

by Jakob S. Jørgensen, Evelina Ametova, Genoveva Burca, Gemma Fardell, Evangelos Papoutsellis, Edoardo Pasca, Kris Thielemans, Martin Turner, Ryan Warr, William R. B. Lionheart, Philip J. Withers

https://arxiv.org/abs/2102.04560

Core Imaging Library – Part II: Multichannel reconstruction for dynamic and spectral tomography

by Evangelos Papoutsellis, Evelina Ametova, Claire Delplancke, Gemma Fardell, Jakob S. Jørgensen, Edoardo Pasca, Martin Turner, Ryan Warr, William R. B. Lionheart, and Philip J. Withers

https://arxiv.org/abs/2102.06126

Enhanced hyperspectral tomography for bioimaging by spatiospectral reconstruction

by Ryan Warr, Evelina Ametova, Robert J. Cernik, Gemma Fardell, Stephan Handschuh, Jakob S. Jørgensen, Evangelos Papoutsellis, Edoardo Pasca, and Philip J. Withers

https://arxiv.org/abs/2103.04796

Crystalline phase discriminating neutron tomography using advanced reconstruction methods

by Evelina Ametova, Genoveva Burca, Suren Chilingaryan, Gemma Fardell, Jakob S. Jørgensen, Evangelos Papoutsellis, Edoardo Pasca, Ryan Warr, Martin Turner, William R B Lionheart, and Philip J Withers

https://doi.org/10.1088/1361-6463/ac02f9

```
[ ]:
```