SoftwareX 9 (2019) 317–323

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

CCPi-Regularisation toolkit for computed tomographic image reconstruction with proximal splitting algorithms

Daniil Kazantsev [a,*,1], Edoardo Pasca [b], Martin J. Turner [b], Philip J. Withers [c,d]

[a] *Diamond Light Source Ltd., Diamond House, Harwell Science & Innovation Campus, Didcot, Oxfordshire, OX11 0DE, UK*
[b] *Scientific Computing Department, Science & Technology Facilities Council - STFC, Rutherford Appleton Laboratory, Didcot, Oxfordshire, OX11 0QX, UK*
[c] *The Henry Royce Institute, School of Materials, The University of Manchester, Manchester, M13 9PL, UK*
[d] *The Research Complex at Harwell, Didcot, Oxfordshire, OX11 0FA, UK*

A R T I C L E I N F O

Article history:
Received 6 September 2018
Received in revised form 21 March 2019
Accepted 2 April 2019

Keywords:
X-ray CT
Iterative methods
Model-based
Regularisation
Denoising
Primal–dual
Big-data

A B S T R A C T

Iterative reconstruction algorithms are often needed to help solve ill-posed inverse problems in computed tomography (CT), especially cases when tomographic projection data are corrupt, noisy or angularly undersampled. Model-based iterative methods can be adapted to fit the measurement characteristics of the data (e.g. noise statistics) and expectations regarding the reconstructed object (e.g. morphology). The prior information is usually introduced in the form of a regulariser, making the inversion task well-posed.

The CCPi-Regularisation toolkit provides a set of variational regularisers (denoisers) which can be embedded in a plug-and-play fashion into proximal splitting methods for image reconstruction. CCPi-RGL comes with algorithms that can satisfy various prior expectations of the reconstructed object, for example being piecewise-constant or piecewise-smooth in nature. The toolkit is written in C language and exploits parallelism with OpenMP directives and the CUDA API; and is wrapped for the Python and MATLAB environments. This paper introduces the toolkit and gives recommendations for selecting a suitable prior model.

Code metadata

Current code version	Version v.19.03
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX_2018_161
Legal Code License	Apache License v.2.0
Code versioning system used	git
Software code languages, tools, and services used	C-OpenMP, CUDA, Python, Matlab
Compilation requirements, operating environments	C compilers (GCC/MinGW), Cython, CMake; NVCC; Linux, Windows, Mac OS
If available Link to developer documentation/manual	For example: https://github.com/vais-ral/CCPi-Regularisation-Toolkit/tree/master/docs
Support email for questions	daniil.kazantsev@diamond.ac.uk

1. Motivation and significance

X-ray computed tomography (CT) [1] is a versatile, often non-invasive technique which uses penetrating radiation to reveal information about the inner structure of an object. In order to obtain a reconstructed image or a volume, a mathematical reconstruction algorithm must be applied to the projection data.

* Corresponding author.
E-mail address: daniil.kazantsev@diamond.ac.uk (D. Kazantsev).

1 The work was completed while the author was with The Manchester X-ray Imaging Facility, School of Materials, The University of Manchester, Manchester, M13 9PL, UK; The Research Complex at Harwell, Didcot, Oxfordshire, OX11 0FA, UK.

https://doi.org/10.1016/j.softx.2019.04.003

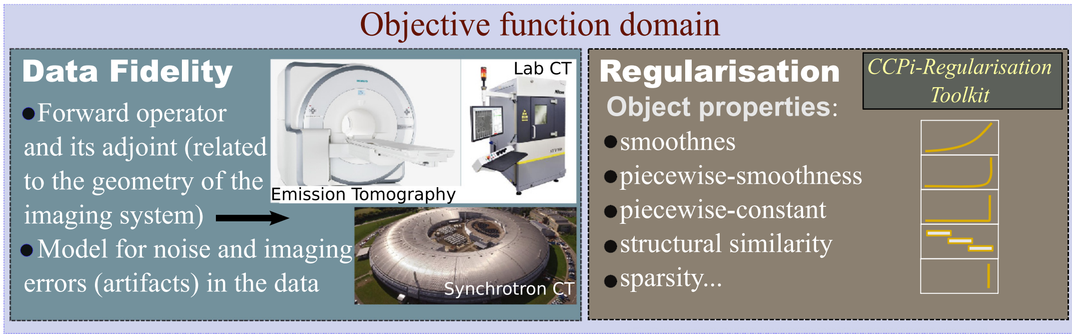

Fig. 1. A place of the CCPi-RGL toolkit within the general optimisation framework.

However, when the measured data are corrupt, noisy or angularly undersampled, direct reconstruction methods, such as the Filtered BackProjection (FBP) method, become ineffective and iterative techniques should be used instead. Iterative methods can help to solve ill-posed inverse problems by choosing a suitable noise model for the measurements [2], and by applying a *regulariser* which incorporates *a priori* knowledge of the solution.

One of the main disadvantages of using regularised iterative methods for tomographic reconstruction is their computational cost in optimising the *objective function* which consists of the *data fidelity* and the regularisation terms. When the terms of the objective function are differentiable, then the gradient or Hessian-based methods can be employed [3]. The smoothness constraint, however, might not always be a desirable feature and one needs to resort to non-smooth optimisation strategies.

Fortunately, the framework of *proximal splitting operators* [4–7] can be applied to indifferentiable cost functions which allows decoupling of its terms resulting in simpler, frequently parallelisable, optimisation steps (see Appendix). Using splitting methods, one can rely on a rigorous mathematical framework which allows a flexible selection of objectives with different properties. This *plug-and-play* approach accelerates prototyping and simplifies implementation of novel reconstruction algorithms which generally perform better for large-dimensional problems [8].

In this paper, we introduce the CCPi[2]-Regularisation Toolkit (CCPi-RGL) which delivers a selection of various regularisers for proximal splitting reconstruction methods. The CCPi-RGL toolkit features more than ten scalar and vectorial variational methods, implemented efficiently using multi-threaded OpenMP directives and the CUDA API with wrappers to Python and MATLAB. Although the CCPi-RGL toolkit can be applied to different image processing tasks (e.g. denoising, deblurring, inpainting), the main focus is tomographic image reconstruction. We demonstrate the applicability of the toolkit by using the primal–dual type of method for 3D image reconstruction of synthetic and real data.

2. Software description

The CCPi-RGL software contains various state-of-the-art variational regularisation techniques which include a second and fourth-order diffusion-based methods as well as local and non-local approaches. Fig. 1 shows a place of the CCPi-RGL toolkit within the general optimisation framework for image reconstruction. The methods of CCPi-RGL are independent of the data fidelity term hence the imaging modality.

In Table 1, we catalogue scalar single-channel regularisation methods of the CCPi-RGL toolkit[3] and below briefly discuss their advantages and disadvantages. More detailed information about each method is given in references [9–17].

Along with a short description we give formulae for the regularisation terms and briefly state the optimisation approaches used for a particular regulariser. For instance, for some regularisers (ROF-TV, NDF, DIFF4th, ROF-LLT), a classical gradient-based technique is used to minimise the objective function. In other non-smooth cases, primal–dual algorithms [5] were used.

In addition to the method description, input data dimensionality requirements, and architecture, we list the main parameters required for the algorithm and also the memory usage for the 3D case. This information can be helpful in practice if one is constrained by the computational resources. Memory requirement is estimated as a total number of volume elements (voxels). For instance, for the 3D ROF-TV method, one needs in total $5 \times N$ voxels allocated when the input volume is of $N = N_x N_y N_z$ in size.

Similarly to Table 1 for scalar (single-channel) images, in Table 2 we demonstrate available algorithms for vectorial (multi-channel) images. Note here that the dFGP-TV algorithm is proposed originally for the two-channel case [18]. It has been recently adapted to the multi-channel case in the multi-spectral image reconstruction problem [19]. The implementation for the TNV penalty has been adopted from the code by Duran et al. [20].

2.1. Software architecture

Core modules of the CCPi-RGL toolkit are developed in the C language with OpenMP directives and with the CUDA API, while the wrappers enable easy access to software from both MATLAB and Python environments (see Fig. 2). We use Cython for Python and the C-MEX interface for MATLAB in order to wrap the C and CUDA-C code. To compile the C code one also needs OS-specific compilers (e.g. GNU GCC, MinGW, Microsoft Visual Studio) and NVCC compiler for CUDA. The user can specify whether to build the CUDA routines with MATLAB and/or Python wrappers. With CMake, CCPi-RGL source code can be built on different operating systems and continuous integration delivers nightly builds of the package on an Anaconda channel.

2.2. Software functionalities

The main functionalities of the CCPi-RGL toolkit include promoting well-posed inversion for a general class of inverse problems. Specifically, the toolkit has been developed for big-data tomographic image reconstruction problems. The selection of regularisers provide a plug-and-play experience while prototyping or developing novel reconstruction methods.

3. Illustrative case studies

In order to demonstrate the functionalities of the CCPi-RGL software, we consider three case studies: volume denoising, 3D image reconstruction of synthetic and real data.

[2] CCPi: Collaborative Computational Project in Tomographic Imaging (https://www.ccpi.ac.uk/).

[3] CCPi-RGL toolkit version 19.03 was used in writing this paper.

Table 1
Single-channel methods of `CCPi-RGL`. Iteration number (T) is required for all methods, we provide the recommended range of iterations needed for each method.

Method Dimensionality Architecture	Description	Main parameters	Memory (3D case) $N = N_x N_y N_z$
ROF-TV 2D/3D CPU/GPU	*Rudin–Osher–Fatemi* total variation (TV) algorithm [9]; $g(\boldsymbol{x}) = \lambda\|\nabla_\epsilon \boldsymbol{x}\|$, $\epsilon = 1e{-}12$, PDE minimisation (explicit)	λ - regul. const. τ - time step $T = 800 - 1000$	5 ×N
FGP-TV 2D/3D CPU/GPU	*Fast Gradient Projection* TV algorithm [10,11]; $g(\boldsymbol{x}) = \lambda\|\nabla \boldsymbol{x}\|$, proximal point algorithm	λ - regul. const. $T = 200 - 400$	11 ×N
SB-TV 2D/3D CPU/GPU	*Split-Bregman* TV algorithm [12]; $g(\boldsymbol{x}) = \lambda\|\nabla \boldsymbol{x}\|$, proximal point algorithm	λ - regul. const. $T = 50 - 150$	8 ×N
NDF 2D/3D CPU/GPU	*Nonlinear Diffusion of the 2-nd order* [13]; $g(\boldsymbol{x}) = \lambda\|\phi(\|\nabla \boldsymbol{x}\|_2^2)\|$, PDE minimisation (explicit)	Linear, Huber, Perona or Tukey λ - regul. const. σ - edge pres. const. τ - time step $T = 600 - 800$	2 ×N
NLTV 2D CPU/GPU	*Nonlocal TV* method [14]; $g(\boldsymbol{x}) = \lambda\|\nabla_\epsilon(\omega)\boldsymbol{x}\|$, Fixed point iteration	λ - regul. const. σ - edge pres. const. N_ω - no. neighbours $T = 2 - 3$	(2D case) 2 ×$\boldsymbol{x}$ N_ω ×$\boldsymbol{x}$ N_ω ×uint8($\boldsymbol{x}$) N_ω ×uint8($\boldsymbol{x}$)
DIFF4th 2D/3D CPU/GPU	*Nonlinear Diffusion of the 4-th order* [15]; $g(\boldsymbol{x}) = \lambda\|\phi(\|\nabla^2 \boldsymbol{x}\|_2^2)\|$, PDE minimisation (explicit)	λ - regul. const. σ - edge pres. const. τ - time step $T = 200 - 400$	3 ×N
TGV 2D/3D CPU/GPU	*Total Generalised Variation* [16]; $g(\boldsymbol{x}) = \alpha_1\|\nabla \boldsymbol{x} - \boldsymbol{v}\| +$ $+ \alpha_0\|\mathcal{E}(\boldsymbol{v})\|$, proximal point algorithm	α_1 - regul. const. α_0 - regul. const. L - Lipschitz const. $T = 500 - 1000$	17 ×N
ROF-LLT 2D/3D CPU/GPU	*ROF* model [9] + *Lysaker-Lundervold-Tai* (LLT) [17]; $g(\boldsymbol{x}) = \lambda_1\|\nabla_\epsilon \boldsymbol{x}\| +$ $+ \lambda_2\|\nabla^2_{\epsilon_2} \boldsymbol{x}\|$, PDE minimisation (explicit)	λ_1 - regul. const. λ_2 - regul. const. τ - time step $T = 800 - 1000$	8 ×N

Table 2
Multi-channel methods of the `CCPi-RGL` toolkit. Iteration number (T) is required for all methods, we provide the recommended range of iterations needed for each method.

Method Dimensionality Architecture	Description	Main parameters	Memory (3D case) $N = N_x N_y N_z$
dFGP-TV 2D/3D+(1) CPU/GPU	*Directional FGP* TV algorithm [18]; $g(\boldsymbol{x}) = \lambda\|\mathcal{P}_\xi \nabla \boldsymbol{x}\|$, proximal point algorithm	λ - regul. const. η - smooth. const. $T = 200 - 400$	13 ×N
TNV 2D+(K) CPU	*Total Nuclear Variation* [20]; $g(\boldsymbol{x}) = \lambda\|\nabla \boldsymbol{x}\|_*$, proximal point algorithm	λ - regul. const. $T = 200 - 400$	(2D case) 22 ×N

3.1. Case study 1: Volume denoising

In order to assess the performance of `CCPi-RGL`, we provide a volume denoising benchmark for CPU and GPU implementations. Using the `TomoPhantom` software [21], we generate a 128^3 voxels volume (see Fig. 3) and apply randomly distributed Gaussian noise.

We aim to solve the volume denoising problem to the required precision. The chosen tolerance parameter to be set at $\delta = 1e{-}6$ and iterations terminated when $\|u^{k+1} - u^k\|/\|u^k\| \leq \delta$ for a three subsequent iterations. This rule suggests a stopping criteria to avoid stagnation or slow convergence of an algorithm. In Table 3, we highlight the values which represent a superior performance of an algorithm using optimal regularisation parameters. The chosen precision is proven to be quite low for the data and lead to many iterations especially for the explicit schemes. In practice, substantially fewer number of iterations is required to reach a satisfactory solution. In Table 2, we provide the recommended range of regularisation iterations to be used for reconstruction.

3.2. Case study 2: Tomographic reconstruction using synthetic data

For our numerical experiments, we use the `TomoPhantom` software [21] to generate a 3D volume size of 256^3 voxels and analytical projection data with Poisson noise and imaging errors.

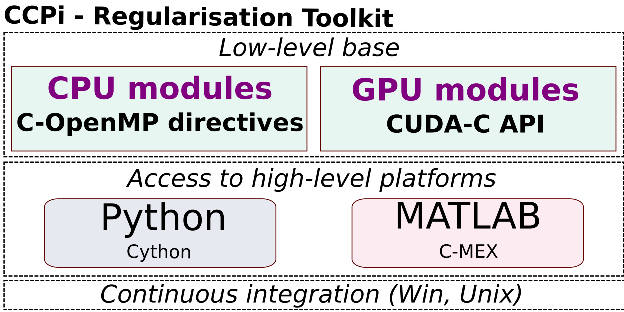

Fig. 2. A block diagram of the `CCPi-RGL` toolkit.

Table 3
Denoising benchmark for 128^3 voxels phantom, RMSE: 88, MSSIM: 0.50. Hardware: GPU Quadro P2000 and CPU Intel(R) Xeon(R) W-2123 CPU @ 3.60 GHz, 4 cores.

Method	Iterations	Time (CPU)	Time (GPU)	GPU speedup	RMSE $\times 100$	MSSIM
ROF-TV	8330	1153s	25s	46	**31.6**	0.79
FGP-TV	930	176s	2.47s	71.2	34.7	0.79
SB-TV	**225**	76.9s	0.89s	**86.4**	34.0	0.79
NDF	530	**4.43s**	**0.33s**	13.4	33.0	0.79
DIFF4th	2425	123.2s	2.70s	45.6	32.2	**0.82**
TGV	7845	4100s	85.5s	47.9	33.7	0.81
ROF-LLT	8500	1664s	31s	53.1	33.5	0.80

The chosen volumetric phantom consists of piecewise-smooth (Gaussians and paraboloids) and piecewise-constant (cuboids) objects (see upper row in Fig. 3). The choice of such a phantom is explained by the abundance of piecewise-smooth objects in material science [22] and medical imaging [23]. The realistic projection data (see bottom row in Fig. 3) were generated with `TomoPhantom` using a mode where textural, noisy 2D flat-fields were simulated and imaging errors (artifacts) were introduced through the normalisation process. Poisson noise is applied to the raw data assuming the flux intensity to be equal 6×10^4 (photon count). Additionally, the noiseless projection data is generated analytically which helps to avoid an 'inverse crime' reconstruction [24].

Table 4
RMSE's and MSSIM's for the reconstructed 3D model in Fig. 3.

Method	FBP	SB-TV	ROF-LLT	TGV
RMSE $\times 100$	10.6	**7.3**	7.5	7.6
MSSIM	0.28	0.670	0.671	**0.674**

In Fig. 4, we demonstrate the reconstructed phantom using various methods: FBP, iterative ADMM (see Appendix) with regularisers from the `CCPi-RGL` toolkit: SB-TV, ROF-LLT, and TGV. The image quality measures: RMSE and MSSIM [25] are presented in Table 4. Note that the achieved values are given for optimally selected regularisation parameters (see the first column of Fig. 4). The FBP reconstructed image (first row of Fig. 4) is noisy with high RMSE and low MSSIM as expected. The iterative reconstruction using the regularised ADMM method improves the signal-to-noise (SNR) characteristics substantially. Notably the regularisation is performed in 3D which further improves the quality compare to 2D case [22]. From the 1D profiles it is notable that the SB-TV method tends to generate piecewise-constant regions and flattening smooth objects whereas higher-order ROF-LLT and TGV methods are designed to reconstruct piecewise-smooth objects. Here, the TGV method reconstructs smooth objects better than ROF-LLT but it overfits flat regions. Also the variations in the background are more evident with the TGV reconstruction, which might negatively contribute to the total RMSE.

The used ADMM algorithm is from the `ToMoBAR`[4] package which employs forward–backward parallel beam projection operator from the `ASTRA-toolbox` [26].

3.3. Case study 3: Tomographic reconstruction using real data

We apply the same methods as in Section 3.2 to real data (see Fig. 5). The data have been collected using parallel pink beam

4 https://github.com/dkazanc/ToMoBAR.

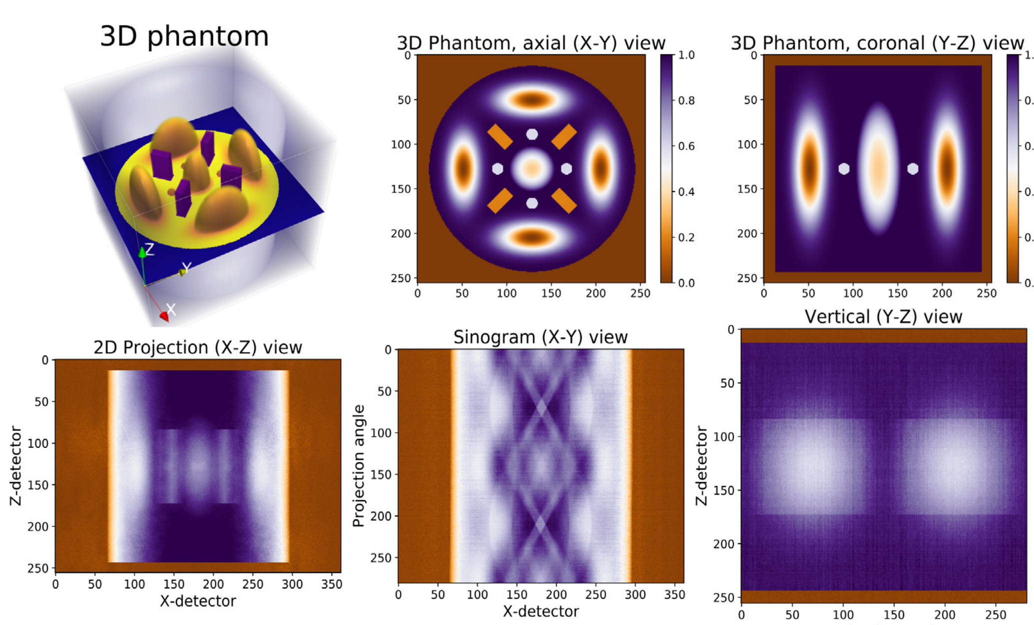

Fig. 3. Upper row: 3D 256^3 voxels phantom no. 16 from the `TomoPhantom` [21] library; bottom row: analytically generated projection data (detector sizes: $D_X = 362$, $D_Y = 256$, projection angles: $\Theta = 281$) with Poisson noise and simulated imaging artifacts.

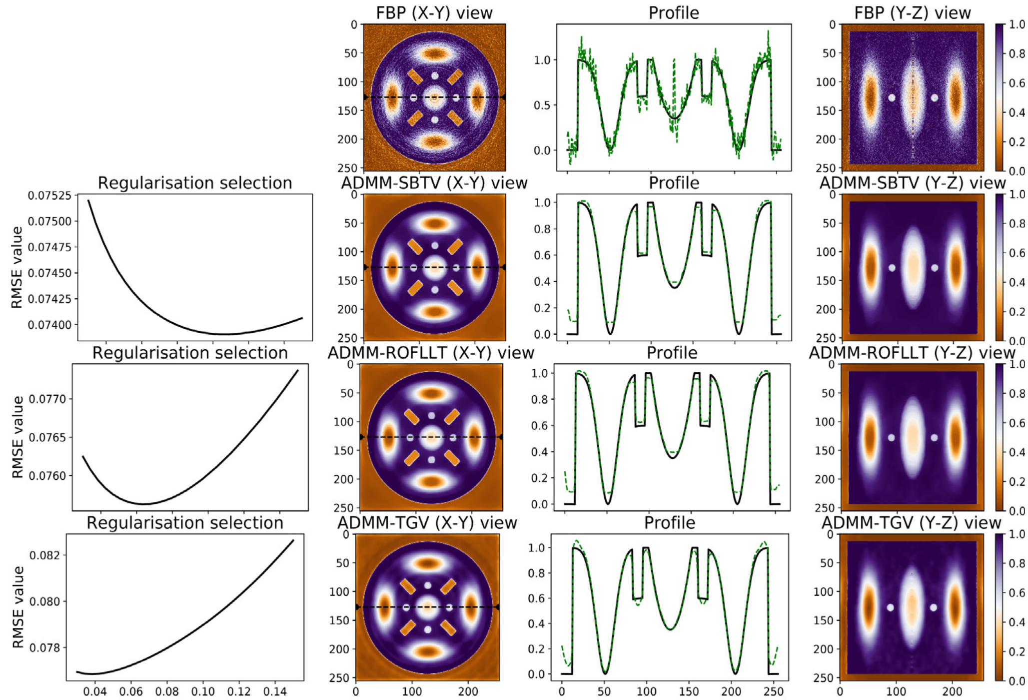

Fig. 4. First column: the result of optimisation procedure to find optimal regularisation parameters for ADMM reconstruction algorithm with SB-TV (second row), ROF-LLT (third row) and TGV (fourth row) regularisers. Reconstructions were obtained running 25 outer ADMM iterations with 50 inner iterations for SB-TV, 600 for ROF-LLT and 600 for TGV.

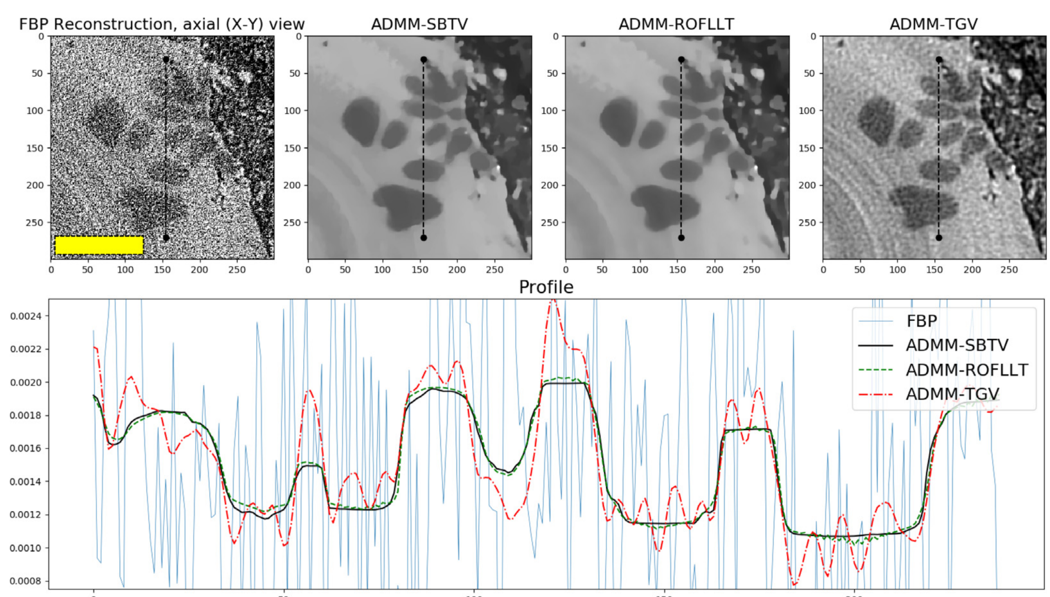

Fig. 5. A magnified region with a line profile of a reconstructed 1k^3 voxels volume from the 3D projection data (detector sizes: $D_X = 1280$, $D_Y = 1000$, projection angles: $\Theta = 360$). Scale bar corresponds to 200 µm.

(energy range 15–30 keV) at the Diamond-Manchester branchline (I13-2) at the Diamond Light Source. Here a metal alloy sample solidifies from its melt while being imaged by X-rays. During the solidification, the dendritic arms are continuously growing (more technical details are available in [22]).

Due to low flux and angularly undersampled conditions, the reconstruction quality using the FBP method is extremely poor (see Fig. 5). Using the regularised ADMM method we can increase the SNR of reconstructions significantly. Notably both SB-TV and ROF-LLT regularisers perform very well by removing noise while TGV struggles to do this. This can be due to suboptimal selection of $\alpha_{0,1}$ parameters for TGV (see Table 1). We noticed that it is difficult to control TGV when noise levels are high, as in this case. For this we recommend to use the ROF-LLT regulariser instead. For low-level noise conditions, TGV normally slightly outperform ROF-LLT.

4. Impact and conclusions

In this paper we present an open-source software `CCPi-RGL` which can be used primarily for tomographic image reconstruction in application to different imaging modalities across various disciplines. The current version of the CCPi-RGL toolkit consists of more than 10 modules and the number is expected to grow in the future. The plug-and-play selection of different regularisers provide a desirable flexibility to a user to establish the most suitable prior to the problem. The core of the toolkit is written in the C-OpenMP and CUDA languages, and wrappers for Python and MATLAB environments are provided.

We demonstrate that the toolkit can be used efficiently and effectively for rigorous testing and benchmarking of novel reconstruction algorithms in application to real big data problems.

Acknowledgements

This work has been funded by the EPSRC, UK grant EP/P02226X/1: A 'Reconstruction Toolkit for Multichannel CT' and the CCPi initiative (EP/M022498/1). The authors acknowledge facilities and the support provided by the Research Complex at Harwell and the Manchester-Diamond collaboration. This work made use of computational support by CoSeC, the Computational Science Centre for Research Communities, through CCPi.

Conflict of interest

The authors do not recognise any potential conflict of interest with the proposed software and research.

Appendix. Proximal methods for tomographic image reconstruction

Here we provide a list of reconstruction methods based on the *proximal operators* framework [4–6] in which the CCPi-RGL algorithms can be easily integrated.

The general optimisation form for tomographic image reconstruction can be formulated as:

$$\min_{\boldsymbol{x}\in\mathbb{R}^N} \mathcal{F}(\boldsymbol{x}) + g(\boldsymbol{x}) \equiv f(\boldsymbol{Ax}) + g(\boldsymbol{x}) \equiv \sum_{i=1}^{n} f_i(\boldsymbol{A}_i\boldsymbol{x}) + g(\boldsymbol{x}), \tag{A.1}$$

where $f_i : \mathbb{R}^{M_i} \to \mathbb{R}, f : \mathbb{R}^M \to \mathbb{R}$ is a continuously differentiable convex function with Lipschitz continuous gradient. Thus, $\mathcal{F}$ also has Lipschitz continuous gradient and we denote its constant by L. The functions f_i measure the *fidelity* of $\boldsymbol{Ax}$ to the normalised projection data $\boldsymbol{b} \in \mathbb{R}^M$ where $\boldsymbol{A} = (\boldsymbol{A}_1; \ldots; \boldsymbol{A}_n) \in \mathbb{R}^{M\times N}$ is the linear forward operator and $\boldsymbol{x} \in \mathbb{R}^N$ is the unknown solution. In accordance with Beers law, raw projection data $\boldsymbol{y}$ is normalised with a registered flat field $\boldsymbol{z}$ as $\boldsymbol{b} = -\ln(\boldsymbol{y}/\boldsymbol{z})$. The *regularisation* term $g : \mathbb{R}^N \to \mathbb{R}$ is a convex, possibly non-differentiable function expressing a prior knowledge of the unknown estimate $\boldsymbol{x}$. The CCPi-RGL toolkit provides different choices for $g(\boldsymbol{x})$.

The common choice for the data fidelity term is the Least-Squares (LS) model: $f(\boldsymbol{Ax}) = \|\boldsymbol{Ax}-\boldsymbol{b}\|_2^2$ or the Penalised Weighted Least Squares (PWLS): $f(\boldsymbol{Ax}) = \|\boldsymbol{Ax} - \boldsymbol{b}\|_{\boldsymbol{W}}^2$, where $\boldsymbol{W} \in \mathbb{R}^{M\times M}$ is a diagonal matrix such as $\{W_{ii} = 1/\sigma_i^2\}_{i=1}^M$ and $\sigma_i^2 \approx y_i^2$ is the variance of the measurements. It is not uncommon to use a more realistic non-linear Poisson model $f(\boldsymbol{Ax}) = \langle \boldsymbol{y}, \boldsymbol{Ax}\rangle + \langle \boldsymbol{z}\exp(-\boldsymbol{Ax}), \boldsymbol{1}\rangle$ [27], or other models [28].

In order to solve the problem (A.1) efficiently, we rely on the theory of the proximal methods [5,6,29] which split the problem into parts which are easier to solve. Before presenting various splitting approaches, we introduce the notion of the *proximal operator*:

$$\mathrm{prox}_{\tau g}(\boldsymbol{u}) = \min_{\boldsymbol{x}\in\mathbb{R}^N} g(\boldsymbol{x}) + \frac{1}{2\tau}\|\boldsymbol{x}-\boldsymbol{u}\|^2. \tag{A.2}$$

All regularisation algorithms of CCPi-RGL aim to solve (A.2) and therefore one needs to be concerned only with f-related sub-problem which is specific to the imaging modality.

Algorithm 1 forward–backward splitting (FBS) method (fixed step)

Require: $\boldsymbol{x}^0 \in \mathbb{R}^N, K$;
$\tau = 1/L$
for $k = 0$ to $K-1$ **do**
$\boldsymbol{x}^{k+1} = \mathrm{prox}_{\tau g}(\boldsymbol{x}^k - \tau\nabla F(\boldsymbol{x}^k))$
end for

The simplest reconstruction algorithm to use is the forward–backward splitting (FBS) method (see Alg. 1).

The slow $O(1/k)$ convergence of FBS can be improved to $O(1/k^2)$ using the optimal step strategy of FISTA [10] (see Alg. 2).

Algorithm 2 Fast Iterative Shrinkage-Thresholding Algorithm (FISTA)

Require: $\boldsymbol{x}^0 \in \mathbb{R}^N, K$;
$\tau = 1/L,\ t_0 = 1$
for $k = 0$ to $K-1$ **do**
1. $\boldsymbol{y}^k = \boldsymbol{x}^k + \left(\frac{t_k-1}{t_{k+1}}\right)(\boldsymbol{x}^k - \boldsymbol{x}^{k-1})$
2. $\boldsymbol{x}^{k+1} = \mathrm{prox}_{\tau g}\left(\boldsymbol{y}^k - \tau\nabla F(\boldsymbol{y}^k)\right)$
3. $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
end for

Both FBS and FISTA require f in (A.1) to be Lipschitz differentiable and include only one proximal step on each iteration. The group of *primal–dual* methods [6,7] relax the differentiability condition but normally rely on two proximal steps instead.

Algorithm 3 Primal–Dual Hybrid Gradient (PDHG) algorithm.

Require: $\boldsymbol{x}^0 \in \mathbb{R}^N, c > 0, K$;
$\sigma = c/\|\boldsymbol{A}\|,\ \tau = 1/(c\|\boldsymbol{A}\|),\ \boldsymbol{y}^0 = 0 \in \mathbb{R}^M$
for $k = 0$ to $K-1$ **do**
1. $\boldsymbol{x}^{k+1} = \mathrm{prox}_{\tau g}(\boldsymbol{x}^k - \tau\boldsymbol{A}^T\boldsymbol{y}^k)$
2. $\boldsymbol{y}^{k+1} = \mathrm{prox}_{\sigma f^*}\left(\boldsymbol{y}^k + \sigma\boldsymbol{A}(2\boldsymbol{x}^{k+1} - \boldsymbol{x}^k)\right)$
end for

Evaluation $\boldsymbol{A}$ or $\nabla F = \boldsymbol{A}^\top \circ \nabla f \circ \boldsymbol{A}$ in each iteration is cumbersome. An approach to overcome this hurdle is "randomisation" which can for instance be achieved within PDHG by selection only a few dual variables $\boldsymbol{y}_i$ in each iteration, resulting in the stochastic PDHG [30,31] method (see Alg. 4).

Algorithm 4 Stochastic PDHG algorithm.

Require: $\boldsymbol{x}^0 \in \mathbb{R}^N, c > 0, K$;
$\sigma_i = c/\|\boldsymbol{A}_i\|,\ \tau = 1/(cn\max_j\|\boldsymbol{A}_j\|),\ \boldsymbol{y}^0 = 0 \in \mathbb{R}^M,\ \boldsymbol{z}^0 = 0 \in \mathbb{R}^N$
for $k = 0$ to $K-1$ **do**
1. $\boldsymbol{x}^{k+1} = \mathrm{prox}_{\tau g}(\boldsymbol{x}^k - \tau\bar{\boldsymbol{z}}^k)$
2. Select $j \in \{1, \ldots, n\}$ uniform at random.
3. $\boldsymbol{y}_i^{k+1} = \begin{cases} \mathrm{prox}_{\sigma_i f_i^*}\left(\boldsymbol{y}_i^k + \sigma_i\boldsymbol{A}_i\boldsymbol{x}^{k+1}\right), & \text{if } i = j \\ \boldsymbol{y}_i^k, & \text{else} \end{cases}$
4. $\Delta\boldsymbol{z} = \boldsymbol{A}_j^\top(\boldsymbol{y}_j^{k+1} - \boldsymbol{y}_j^k)$
$\boldsymbol{z}^{k+1} = \boldsymbol{z}^k + \Delta\boldsymbol{z},\ \bar{\boldsymbol{z}}^{k+1} = \boldsymbol{z}^k + n\Delta\boldsymbol{z}$
end for

Under the linearisation conditions, the PDHG method becomes the well-known ADMM method [6,32] (see Alg. 5).

Step 1. of the Alg. 5 is a quadratic optimisation problem when data fidelity is chosen to be PWLS: $F(\boldsymbol{x}) = 1/2\|\boldsymbol{Ax} - \boldsymbol{b}\|_{\boldsymbol{W}}^2$.

Algorithm 5 Alternating Directions of Multipliers (ADMM)

Require: $\boldsymbol{x}^0 \in \mathbb{R}^N$, step $\tau > 0$, K;
$\boldsymbol{u}^0 = 0 \in \mathbb{R}^N$, $\boldsymbol{y}^0 = \boldsymbol{x}^0$
for $k = 0$ to $K - 1$ **do**
1. $\boldsymbol{x}^{k+1} = \mathtt{prox}_{\tau F}\left(\boldsymbol{v}^k - \boldsymbol{u}^k\right)$
2. $\boldsymbol{v}^{k+1} = \mathtt{prox}_{\tau g}\left(\boldsymbol{x}^{k+1} + \boldsymbol{u}^k\right)$
3. $\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \boldsymbol{x}^{k+1} - \boldsymbol{v}^{k+1}$
end for

Therefore one needs to solve: $\boldsymbol{x}^{k+1} = (\boldsymbol{I} + \tau \boldsymbol{A}^\top \boldsymbol{W} \boldsymbol{A})^{-1}(\tau \boldsymbol{A}^\top \boldsymbol{W} \boldsymbol{b} + \boldsymbol{v}^k - \boldsymbol{u}^k)$ for which Krylov-type methods or Newton solvers can be used [3].

References

[1] Buzug TM. Computed tomography: from photon statistics to modern cone-beam CT. Springer; 2008, http://dx.doi.org/10.1007/978-3-540-39408-2.
[2] Bertero M, Boccacci P. Introduction to inverse problems in imaging. CRC press; 1998.
[3] Demmel JW. Applied numerical linear algebra, vol. 56. Siam; 1997.
[4] Combettes PL, Wajs VR. Signal recovery by proximal forward-backward splitting. Multiscale Model Simul 2005;4(4):1168–200. http://dx.doi.org/10.1137/050626090.
[5] Rockafellar RT. Monotone operators and the proximal point algorithm. SIAM J Control Optim 1976;14(5):877–98. http://dx.doi.org/10.1137/0314056.
[6] Chambolle A, Pock T. An introduction to continuous optimization for imaging. Acta Numer 2016;25:161–319. http://dx.doi.org/10.1017/S096249291600009X.
[7] Chambolle A, Pock T. A first-order primal-dual algorithm for convex problems with applications to imaging. J Math Imaging Vision 2011;40(1):120–45. http://dx.doi.org/10.1007/s10851-010-0251-1.
[8] Venkatakrishnan SV, Bouman CA, Wohlberg B. A first-order primal-dual algorithm for convex problems with applications to imaging. J Math Imaging Vision 2011;40(1):120–45. http://dx.doi.org/10.1007/s10851-010-0251-1.
[9] Rudin LI, Osher S, Fatemi E. Nonlinear total variation based noise removal algorithms. Physica D 1992;60:259–68. http://dx.doi.org/10.1016/0167-2789(92)90242-F.
[10] Beck A, Teboulle M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM J Imaging Sci 2009;2(1):183–202. http://dx.doi.org/10.1137/080716542.
[11] Beck A, Teboulle M. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems. IEEE Trans Image Process 2009;18(11):2419–34. http://dx.doi.org/10.1109/TIP.2009.2028250.
[12] Goldstein T, Osher S. The split Bregman method for L1-regularized problems. SIAM J Imaging Sci 2009;2(2):323–43. http://dx.doi.org/10.1137/080725891.
[13] Perona P, Malik J. Scale-space and edge detection using anisotropic diffusion. IEEE Trans Pattern Anal Mach Intell 1990;12(7):629–39. http://dx.doi.org/10.1109/34.56205.
[14] Elmoataz A, Lezoray O, Bougleux S. Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing. IEEE Trans Image Process 2008;17(7):1047–60. http://dx.doi.org/10.1109/TIP.2008.924284.
[15] Hajiaboli MR. An anisotropic fourth-order diffusion filter for image noise removal. Int J Comput Vis 2011;92(2):177–91. http://dx.doi.org/10.1007/s11263-010-0330-1.
[16] Bredies K, Kunisch K, Pock T. Total generalized variation. SIAM J Imaging Sci 2010;3(3):492–526. http://dx.doi.org/10.1137/090769521.
[17] Lysaker M, Lundervold A, Tai XC. Noise removal using fourth-order partial differential equation with applications to medical magnetic resonance images in space and time. IEEE Trans Image Process 2003;12(12):1579–90. http://dx.doi.org/10.1109/TIP.2003.819229.
[18] Ehrhardt MJ, Betcke MM. Multi-contrast MRI reconstruction with structure-guided total variation. SIAM J Imaging Sci 2016;9(3):1084–106. http://dx.doi.org/10.1137/15M1047325.
[19] Kazantsev D, Jørgensen JS, Andersen MS, Lionheart WR, Lee PD, Withers PJ. Joint image reconstruction method with correlative multi-channel prior for x-ray spectral computed tomography. Inverse Problems 2018;34(6):064001. http://dx.doi.org/10.1088/1361-6420/aaba86.
[20] Duran J, Moeller M, Sbert C, Cremers D. Collaborative total variation: a general framework for vectorial TV models. SIAM J Imaging Sci 2016;9(1):116–51. http://dx.doi.org/10.1137/15M102873X.
[21] Kazantsev D, Pickalov V, Nagella S, Pasca E, Withers PJ. TomoPhantom, a software package to generate 2D-4D analytical phantoms for CT image reconstruction algorithm benchmarks. SoftwareX 2018;7:150–5. http://dx.doi.org/10.1016/j.softx.2018.05.003.
[22] Kazantsev D, Guo E, Phillion AB, Withers PJ, Lee PD. Model-based iterative reconstruction using higher-order regularization of dynamic synchrotron data. Meas Sci Technol 2017;28(9):094004. http://dx.doi.org/10.1088/1361-6501.
[23] Wernick MN, Aarsvold JN. Emission tomography: the fundamentals of PET and SPECT. Elsevier; 2004.
[24] Kaipio J, Somersalo E. Statistical inverse problems: discretization, model reduction and inverse crimes. J Comput Appl Math 2007;198(2):493–504. http://dx.doi.org/10.1016/j.cam.2005.09.027.
[25] Wang Z, Bovik AC, Sheikh HR, Simoncelli EP. Image quality assessment: From error visibility to structural similarity. IEEE Trans Image Process 2004;13(4):600–12. http://dx.doi.org/10.1109/TIP.2003.819861.
[26] van Aarle W, Palenstijn WJ, Cant J, Janssens E, Bleichrodt F, Dabravolski A, De Beenhouwer J, Batenburg KJ, Sijbers J. Fast and flexible X-ray tomography using the ASTRA toolbox. Opt Express 2016;24(22):25129–47. http://dx.doi.org/10.1364/OE.24.025129.
[27] Melot C, Boursier Y, Aujol JF, Anthoine S. Some proximal methods for Poisson intensity CBCT and PET. Inverse Probl Imaging 2012;6(4):565. http://dx.doi.org/10.3934/ipi.2012.6.565.
[28] Kazantsev D, Bleichrodt F, van Leeuwen T, Kaestner A, Withers PJ, Batenburg KJ, Lee PD. A novel tomographic reconstruction method based on the robust Student's t function for suppressing data outliers. IEEE Trans Comput Imaging 2017;3(4):682–93. http://dx.doi.org/10.1109/TCI.2017.2694607.
[29] Combettes PL, Pesquet JC. Proximal splitting methods in signal processing. In: Fixed-point algorithms for inverse problems in science and engineering. New York, NY: Springer; 2011, p. 185–212. http://dx.doi.org/10.1007/978-1-4419-9569-8_10.
[30] Chambolle A, Ehrhardt MJ, Richtárik P, Schönlieb CB. Stochastic primal-dual hybrid gradient algorithm with arbitrary sampling and imaging applications. SIAM J Optim 2018;28(4):2783–808. http://dx.doi.org/10.1007/s10851-010-0251-1.
[31] Ehrhardt MJ, Markiewicz P, Chambolle A, Richtárik P, Schott J, Schönlieb CB. Faster PET reconstruction with a stochastic primal–dual hybrid gradient method. In: Wavelets and sparsity XVII, vol. 10394. International Society for Optics and Photonics; 2017, p. 103941O.
[32] Boyd S, Parikh N, Chu E, Peleato B, Eckstein J. Distributed optimization and statistical learning via the alternating direction method of multipliers. Found Trends Mach Learn 2011;3(1):1–122. http://dx.doi.org/10.1561/2200000016.