arXiv:2603.21230v1 [math.OC] 22 Mar 2026

使用核心成像库解决逆问题的随机优化模块化方法

EVANGELOS PAPOUTSELLIS, Finden Ltd，卢瑟福·阿普尔顿实验室，英国MARGARET A. G. DUFF, 科学与技术设施委员会，卢瑟福·阿普尔顿实验室，英国JAKOB S. JØRGENSEN, 丹麦技术大学应用数学与计算机科学系，丹麦SAM PORTER, 伦敦大学学院核医学研究所，英国CLAIRE DELPLANCKE , EDF 巴黎-萨克雷实验室，法国GEMMA FARDELL, 科学与技术设施委员会，卢瑟福·阿普尔顿实验室，英国EDOARDO PASCA, 科学与技术设施委员会，卢瑟福·阿普尔顿实验室，英国KRIS THIELEMANS, 伦敦大学学院核医学研究所及UCLHawkesInstitute，英国

核心成像库（CIL）是一个开源且通用的 Python 框架，用于求解逆问题，尤其适用于计算机断层扫描（CT）等成像应用。它采用数据和算子的插件架构，并可与 ASTRA、TIGRE 和 SIRF 等工具箱对接。CIL 的关键组件之一是其优化模块，使用户能够灵活组合数学算子与泛函，构建光滑和非光滑优化问题，并使用一系列一阶算法求解。本文介绍了 CIL 的扩展：一种用于随机优化的新型模块化框架，使研究人员能够轻松使用多种现有随机优化算法，也能通过组合模块化构建模块来创建新算法。用户可以灵活配置算法组件，以适应不同的问题结构，并尝试各种采样和步长策略。我们的设计并非针对每种固定算法分别采用存在大量冗余的黑箱实现，而是采用模块化设计，使构建模块能够灵活组合，从而实现丰富多样的算法实例。该框架尤其适用于大规模应用，在这些应用中，随机方法相较于确定性方法具有显著的计算优势。为展示其通用性和实用价值，我们针对成像逆问题中的真实世界数据集开展了实验，例如X射线计算机断层扫描和正电子发射断层成像（PET）重建。

总而言之，此次软件扩展旨在为研究界提供一套稳健且可扩展的优化套件，用于开发、测试和基准测试逆问题的随机方法。

1 引言

成像逆问题是计算成像中的一项根本挑战，其目标是根据间接、不完整且往往含有噪声的测量结果估计未知量。这类问题出现在医学成像、材料科学、遥感和地球科学等多个领域。对于有限维空间 $\mathbb{X}$，$\mathbb{Y}$ 以及前向算子 $\boldsymbol{A}\colon \mathbb{X} \to \mathbb{Y}$，许多逆问题可用以下算子方程来表述

$$\boldsymbol{A}\boldsymbol{u} \approx \boldsymbol{b}, \tag{1}$$

作者联系信息：EVANGELOS PAPOUTSELLIS，epapoutsellis@gmail.com，Finden Ltd，卢瑟福·阿普尔顿实验室，哈威尔园区，牛津郡，英国；MARGARET A. G. DUFF，margaret.duff@stfc.ac.uk，科学与技术设施委员会，卢瑟福·阿普尔顿实验室，哈威尔园区，牛津郡，英国；Jakob S. Jørgensen，jakj@dtu.dk，丹麦技术大学应用数学与计算机科学系，哥本哈根，丹麦；SAM PORTER，sam.porter.18@ucl.ac.uk，核医学研究所，伦敦大学学院，伦敦，英国；CLAIRE DELPLANCKE，claire.delplancke@edf.fr，EDF 巴黎-萨克雷实验室，巴黎，法国；GEMMA FARDELL，gemma.fardell@stfc.ac.uk，科学与技术设施委员会，卢瑟福·阿普尔顿实验室，哈威尔园区，牛津郡，英国；EDOARDO PASCA，edoardo.pasca@stfc.ac.uk，科学与技术设施委员会，卢瑟福·阿普尔顿实验室，哈威尔园区，牛津郡，英国；KRIS THIELEMANS，k.thielemans@ucl.ac.uk，核医学研究所和伦敦大学学院霍克斯研究所，伦敦大学学院，伦敦，英国。

其中， $\boldsymbol{u} \in \mathbb{X}$ 是未知量， $\boldsymbol{b} \in \mathbb{Y}$ 是测量数据。这类问题往往是不适定的，*i.e.*，即解可能不存在、不唯一或不稳定。为了解决这一问题，一种常见方法是在变分（基于优化）方法中，以正则化的形式纳入解的先验信息。

带正则化的变分方法旨在通过求解以下优化问题，恢复 **(1)** 的近似解 $\boldsymbol{u} \in \mathbb{X}$

$$\min_{\boldsymbol{u}\in\mathbb{X}} D(\boldsymbol{Au}, \boldsymbol{b}) + \alpha R(\boldsymbol{u}). \tag{2}$$

这里， $D\colon \mathbb{Y} \times \mathbb{Y} \to \mathbb{R}_\infty := \mathbb{R} \cup \{\infty\}$ 表示保真项，用于衡量前向模型作用于未知量后所得结果与测量数据之间的差异。根据测量数据的噪声统计特性， $\boldsymbol{b}$，对于高斯噪声、脉冲噪声 [1], 和泊松噪声 [2], ，保真项通常分别采用最小二乘距离、绝对差或 Kullback-Leibler散度的形式。正则化项， $R(\boldsymbol{u})$，用于促使解 $\boldsymbol{u}$ 具有某些特性，例如平滑性、稀疏性或边缘保持；通常通过正则化参数 $\alpha > 0$ 加权，以平衡式 **(2)** 中的两个项。常见的正则化项包括 Tikhonov正则化 [3], 、全变分 **(TV)** [4] 及其高阶扩展，即全广义变分 **(TGV)** [5], 、全核变分 [6] ，以及更一般的基于张量结构的正则化 [7]。

文献中提出了许多旨在求解 **(2)** 的算法 [8–12] ，每种算法都针对特定的成像模态、数据保真度要求和先验假设进行了调整。部分作者在 [13], 中介绍了核心成像库（CIL），这是一个通用的开源软件框架，专为成像逆问题设计，尤其侧重于先进的断层成像任务。除了提供与计算机断层扫描（CT）及相关模态中成像数据的预处理、后处理和可视化有关的重要实用工具外，该框架还设计了灵活的优化框架，用于求解逆问题。灵活的 cil.optimisation模块基于三个主要构建模块，即函数、算子和算法，使用户能够以最少的工作量，为各种针对特定成像应用的优化问题构建解。CIL 已实现多种算法，用于求解形式为 **(2)** 的最小化问题，包括梯度下降（GD）、共轭梯度最小二乘（CGLS），以及基于近端分裂的算法，例如近端梯度下降（PGD）（别名为 ISTA）、快速迭代收缩阈值算法（FISTA） [14],原始-对偶混合梯度（PDHG） [15] 和线性化交替方向乘子法（LADMM） [16]。

这些算法在 CIL 中实现，已应用于各种场景，尤其侧重于大型且具有挑战性数据集上的断层重建问题，包括低计数、非标准几何、有限角度、多光谱或多模态数据；相关应用涵盖常规 X 射线计算机断层扫描、 [13], 动态 X 射线计算机断层扫描、 [17],有限角度计算机断层扫描、 [18], 高光谱成像和中子断层成像（ [19, 20] ），以及聚变等离子体成像（ [21]）。此外，CIL 优化为医学成像领域提供了独特的解决方案，因为它与协同图像重建软件（SIRF）完全兼容；SIRF 是用于医学成像应用的 [22], 开源软件，涵盖正电子发射断层成像（PET）、单光子发射计算机断层成像（SPECT）和磁共振成像（MRI）（ [23, 24]）。最后，CIL 库也已应用于断层成像之外的领域，尤其是合成孔径雷达（SAR）成像（ [25]）。

尽管近年来用于逆问题的迭代正则化算法取得了显著进展和成功，但它们的应用仍然有限，因为这些算法可能收敛缓慢或内存需求较高。传统的一阶求解器及其原始-对偶变体通常采用确定性模式设计，每次更新都处理整个数据集，也就是说，每次迭代都要遍历一次数据。这种方法的计算成本可能很高，尤其是随着断层扫描仪不断进步，生成的数据集越来越大、分辨率越来越高

。这一挑战推动了相关算法的发展，使其能够在每次数据遍历中更快收敛，并降低每次迭代的计算成本；其中许多算法采用了随机优化技术，详见 Ehrhardt 等人 [26] 的综述。与每次迭代都处理整个数据集的传统确定性方法不同，随机方法会随机选取并处理一部分采集数据，例如少量断层投影视角。与机器学习中广泛使用的随机梯度下降（SGD）方法类似，这种策略有望减少每次迭代的计算量，同时仍能收敛至解。凭借这些优势，随机优化成为应对现代逆问题中高分辨率、大规模数据集所带来挑战的一种颇具吸引力的方案。

1.1 贡献

本文通过引入一种独特且易于使用的设计，对确定性 CIL 优化框架进行了扩展，从而能够高效地实现和测试随机优化技术。优化研究的快速发展促使新算法不断发表，因此，要评估这些算法在特定成像问题中的实际表现颇具挑战。我们的软件为用户提供易于使用的环境，用于测试和验证这些新兴算法，使他们能够评估算法在包括 X 射线计算机断层扫描、正电子发射断层成像、SPECT 和磁共振重建在内的实际问题中的有效性。此外，CIL 24.3.0 版本提供了更多实用工具，以增强其功能和灵活性，例如数据拆分和采样方法、预条件、适用于非固定步长的线搜索方法，以及用于监控和分析算法进展的回调方法。这些功能使用户能够根据具体需求定制优化过程，因此在解决复杂逆问题时尤为有效。最后，所提出的模块化设计使用户能够轻松集成新的优化方法以扩展其功能，从而推动创新，并增强其应对各类棘手问题的适应能力。

本文的目的并非对 (3) 的算法进行全面的文献综述或比较，而是介绍随机优化，并展示其优越性，同时介绍旨在帮助本领域研究者应对各类优化问题的软件解决方案。我们着重提供一个优先考虑灵活性、通用性和可复现性的框架，使用户能够针对自身应用高效地对优化方法进行原型设计、改进和扩展。本文中的实验旨在展示这些算法应用于现实问题时的预期表现。本文介绍了 CIL 24.3.0 版本新增的功能。

1.2 相关工作

有许多开源库为成像领域中的逆问题提供优化算法，各自面向不同的应用和社区。PyProximal [27] 是一个用于凸优化的 Python 库，作为 PyLops 框架 [28], 的重要组成部分开发而成；PyLops 是一个用于无矩阵线性代数和大规模逆问题的库。SCICO（科学计算成像代码） [29] 是一个用于科学成像和逆问题的 Python 软件包，以 JAX 作为数值后端，并支持学习先验和深度学习重建方法。DeepInv [30] 是一个用于成像逆问题的开源 PyTorch 库，提供基于物理的前向模型（包括 MRI、断层成像和叠层成像的前向模型）、学习型重建网络以及即插即用去噪器。PyXu [31] 是一个用于计算成像流水线的开源 Python 框架，强调模块化算子以及在中央处理器和图形处理器上的可扩展执行。算子离散化库（ODL） [32] 是一个 Python 框架，用于使用抽象前向算子和迭代

重建方法。尽管这些库各有侧重——涵盖从变分正则化到深度学习，再到可扩展的算子代数——但它们主要提供 ISTA、FISTA、ADMM 和 PDHG 等确定性优化算法。据我们所知，我们提出的软件扩展首次为这一领域的随机优化提供了统一且可扩展的接口；上述库均未提供类似功能。

我们的工作为开源优化生态系统引入了一种新颖的软件设计。我们没有简单地将随机优化算法添加到现有模块中，而是提出了一个统一的优化接口，使随机方法（如数据拆分、基于采样的更新和方差缩减梯度方案）能够与近端算子或原始-对偶算法等经典方法无缝结合。正如后文将讨论的，确定性算法与随机基于梯度的算法之间的关键区别，在于目标函数的梯度如何计算。为此，我们将随机梯度估计封装在专用函数中。该设计使我们能够重用 CIL 现有的确定性基于梯度的算法进行随机优化，而无需创建单独的工作流。因此，同一基础算法既可以执行标准确定性方法，也可以执行其随机对应版本。这种方法为成像逆问题中的确定性和随机优化提供了统一、连贯的接口。

1.3 论文概述

在第 2 节中，我们介绍框架中的核心概念：将随机梯度估计器嵌入确定性的基于梯度的算法，从而构建随机算法。本节将从理论角度阐述这一做法，同时联系软件实现。在第 3 节中，我们将再次介绍该框架，并通过软件演示逐步讲解其在 CIL 中的实现，为读者提供构建随机算法的分步方法。随后，我们将通过两个案例研究展示 CIL 随机优化框架的各个方面：第 4 节中的大规模平行束X射线计算机断层成像数据集，以及第 5 节中的低计数正电子发射断层成像数据集。

2 理论与实现

随机框架的核心思想，是将**随机梯度估计器**嵌入**确定性梯度算法**中，从而快速、简便地组合算法，形成一系列不同的方法。本节将首先介绍确定性算法，然后介绍随机梯度估计器的概念，以及它如何涵盖文献中的许多随机算法。

在此过程中，我们将讨论确定性算法中其他可调整的方面：步长、预条件器和动量。我们还会提到随机算法中的**分区**和**采样**，但这些内容将在后续章节中详细介绍。

2.1 确定性优化框架

有关 CIL 中可用的确定性优化算法的详细讨论，请参阅参考文献 [13, 17]。在本节中，我们仅考虑随机框架及比较所需的算法。我们考虑一个通用优化框架，将 (2) 重写为三个项之和：

$$\boldsymbol{x}^{*} \in \underset{\boldsymbol{x}\in\mathbb{X}}{\arg\min}\, F(\boldsymbol{x}) := f(\boldsymbol{x}) + g(\boldsymbol{x}) + h(\boldsymbol{K}\boldsymbol{x}). \tag{3}$$

使用核心成像库求解逆问题的随机优化模块化方法

算法 1梯度下降

参数： 步长 $\gamma_k > 0$，预条件器 P_k
初始化： $\boldsymbol{x}_0 \in \mathbb{X}$
对于 $k = 0, \ldots, K-1$ **执行**
    $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \gamma_k P_k \boldsymbol{G_k}(\boldsymbol{x}_k)$
</style结束
循环

算法 2近端梯度下降

参数： 步长 $\gamma_k > 0$，预条件器 P_k
初始化： $\boldsymbol{x}_0 \in \mathbb{X}$
对于 $k = 0, \ldots, K-1$ **执行**
    $\boldsymbol{x}_{k+1}$ =近端算子 $_{\gamma_k g}(\boldsymbol{x}_k - \gamma_k P_k \boldsymbol{G_k}(\boldsymbol{x}_k))$
结束循环

算法3加速近端梯度下降

参数： 步长 $\gamma_k > 0$，动量参数 $\beta_k > 0$，预条件器 P_k
初始化： $\boldsymbol{x}_0 \in \mathbb{X}$
对于 $k = 0, \ldots, K-1$ **执行**
    $\boldsymbol{x}_{k+1}$ =近端算子 $_{\gamma_k g}(\boldsymbol{y}_k - \gamma_k P_k \boldsymbol{G_k}(\boldsymbol{y}_k))$
    $\boldsymbol{y}_{k+1} = \boldsymbol{x}_{k+1} + \beta_k(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k)$
结束循环

算法4PD3O

参数： 步长 $\sigma, \tau > 0$
初始化： $\boldsymbol{x}_0 \in \mathbb{X}, \boldsymbol{y}_0 \in \mathbb{Y}$
对于 $k = 0, \ldots, K-1$ **执行**
    $\boldsymbol{x}_{k+1}$ =近端算子 $_{\tau g}(\boldsymbol{x}_k - \tau \boldsymbol{G_k}(\boldsymbol{x}_k) - \tau \boldsymbol{K}^* \boldsymbol{y}_k)$
    $\bar{\boldsymbol{x}}_{k+1} = 2\boldsymbol{x}_{k+1} - \boldsymbol{x}_k + \tau(\boldsymbol{G_k}(\boldsymbol{x}_k) - \boldsymbol{G_k}(\boldsymbol{x}_{k+1}))$
    $\boldsymbol{y}_{k+1}$ = 近端算子$_{\sigma h}(\boldsymbol{y}_k + \sigma \boldsymbol{K} \bar{\boldsymbol{x}}_k)$
</style结束循环

表1. CIL 中与随机优化相关的确定性优化算法。通过选择 $\boldsymbol{G_k} = \nabla f$，即可得到现有的确定性变体。按照后续章节所述，为 $\boldsymbol{G_k}$ 选择梯度估计器，即可得到特定的随机优化算法。此处，P_k 表示预条件器，其用法将在第5节中说明。

其中，$\boldsymbol{K}$：$\mathbb{X} \to \mathbb{Y}$是一个连续线性算子，且

$$\|\boldsymbol{K}\| = \max\{\|\boldsymbol{K}\boldsymbol{x}\| : \boldsymbol{x} \in \mathbb{X}, \|\boldsymbol{x}\| \le 1\} < \infty,$$

且函数 f，g：$\mathbb{X} \to \mathbb{R} \cup \{\infty\}$ 和 h：$\mathbb{Y} \to \mathbb{R} \cup \{\infty\}$ 均为适当、凸且下半连续的。此外，f是 L-光滑的，即它是一个具有利普希茨梯度 $L > 0$的可微函数，

$$\|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\| \le L\|\boldsymbol{x} - \boldsymbol{y}\|, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{X}. \tag{4}$$

假设函数 g 及 h 的凸共轭均可近端计算。回顾一下，h的凸共轭定义为

$$h^*(\boldsymbol{y}^*) = \sup_{\boldsymbol{y} \in \mathbb{Y}} \{\langle \boldsymbol{y}, \boldsymbol{y}^* \rangle - h(\boldsymbol{y})\} \tag{5}$$

函数 g 的近端算子为

$$\operatorname{prox}_{\tau g}(\boldsymbol{z}) = \arg\min_{\boldsymbol{x} \in \mathbb{X}} \left\{ \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{z}\|^2 + \tau g(\boldsymbol{x}) \right\} \tag{6}$$

其中 $\tau > 0$。如果式 (6) 存在解析解，或可通过迭代过程在一定精度范围内高效求解，则称该函数为“可近端计算的”。CIL 最新版本中提供且本文讨论的算法以伪代码形式列于表1中。

我们考虑求解 (3) 的两大类算法：近端梯度算法和原始-对偶算法。

近端梯度算法。当 (3) 中缺少复合项，即 $h \circ \boldsymbol{K} = 0$ 时，我们得到一个更简单的两项优化问题，例如可用近端梯度下降（PGD）算法求解（见算法 2）。这种灵活的算法包含若干其他常见算法作为特例。例如，如果 g 是 ℓ_1 范数，即$g = \|\cdot\|_1$，我们便得到 ISTA 算法 [33]。请注意，在 CIL 中，ISTA 和 PGD 这两个名称可以互换使用；

这两种算法互为别名。此外，如果 g 是某个闭的非空凸集的示性函数，那么 g 的近端算子就是投影到该集合上，由此可得到投影梯度下降（GD）算法。此外，如果 $g = 0$ 也成立，我们便得到仅含一个可微项的简单情形，可用 GD 求解（见算法 1）。

有多种参数选择，经过恰当调优后，可以加速基于梯度的算法，包括随机变体：

- **步长**可以自适应调整，也可以在迭代过程中保持不变。默认情况下，CIL 中的 GD 使用 Armijo 规则（一种回溯线搜索算法）来选择步长，以确保迭代使优化目标充分下降（参见 [34] 中的算法 3.1）。CIL 还实现了 Barzilai-Borwein 步长法 [35] 。在 CIL 中，自定义递减步长规则也很容易实现。

- **基于动量的**加速已在加速近端梯度下降（APGD，见算法 3）中实现。用户可以自定义选择动量系数 β_k。默认情况下，动量参数按 $t_0 = 1$、$t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$、$\beta_k = \frac{t_k - 1}{t_{k+1}}$ 计算，从而得到快速迭代收缩阈值算法（FISTA） [36]。这就是 Nesterov 动量或 Nesterov 加速梯度法（NAG）；当 $g = h = 0$ [36–41] 时，无需额外内存需求即可实现 O $\left(\frac{1}{k^2}\right)$ 加速。有关更多详情，请参阅近期关于加速方法的综述 [42]。

- **预条件**梯度估计器会修正计算得到的梯度，以改善收敛性质，尤其是在病态问题中。预条件可以理解为对梯度应用一种变换，以反映问题的几何特性，实质上是对目标函数变化较慢的方向进行重新缩放。例如，在 CIL 中，可以选择灵敏度预条件：每次调用预条件器时，都会将梯度乘以 $(\boldsymbol{A}\mathbf{1})^{-1}$，其中$\mathbf{1}$是算子值域中一个元素，其值均为 1，参见 [43, 44]。CIL 框架支持用户自定义预条件器，预条件策略可以包括对角缩放、基于算子的预条件器，或针对特定逆问题量身定制的学习型预条件器， [45]。

原始-对偶算法。有许多原始-对偶算法旨在求解 (3)，各自具有不同的优缺点 [46–51]。

在没有光滑函数的情况下，即 (3) 中的 $f = 0$ ，我们得到适用于原始-对偶混合梯度（PDHG）算法的双项问题；该算法在文献中也称为 Chambolle-Pock，参见 [15, 52, 53]。CIL 中提供了该算法，算法详见 [15] 。

对于三项问题（3），PD3O算法已在 CIL 中实现（见算法4）。近期的一项研究 [54]在可证明最优的宽松步长条件下提供了收敛保证。PD3O具有宽松的步长要求、易于调参、较低的计算成本，并且能够自然扩展到随机优化。虽然我们并不声称它适用于所有使用场景，都是最优算法，但其基础实现可作为实现上述相关算法和扩展的有价值指南，例如 Bregman近端算子 [55]。

2.2 随机方法

在本节中，我们将确定性优化框架（3）扩展到其随机对应形式 t.

首先，我们假设 (3) 可以**划分**为有限和结构：

$$\boldsymbol{x}^* \in \arg\min_{\boldsymbol{x}\in\mathbb{X}} \sum_{i=0}^{N-1} f_i(\boldsymbol{x}) + \sum_{m=0}^{M-1} g_m(\boldsymbol{x}) + \sum_{t=0}^{T-1} h_t(\boldsymbol{K}_t\boldsymbol{x}). \tag{7}$$

使用核心成像库求解逆问题的随机优化模块化方法

特别地，有限维空间 $\mathbb{Y}$ 现在是 $(\mathbb{Y}_t)_{t=0}^{T-1}$ 的笛卡尔积，即 $\mathbb{Y} := \prod_{t=0}^{T-1} \mathbb{Y}t$，其元素为$\boldsymbol{y} = (\boldsymbol{y}_0, \text{fi } \boldsymbol{y}_{T-1})$。此外，我们定义线性有界算子 $\boldsymbol{K}_t \colon \mathbb{X} \to \mathbb{Y}_t$，其中 $\boldsymbol{K}_t x = (\boldsymbol{K}x)_t t$。在这些假设下，复合项 $h(\boldsymbol{x})$ 现在表示为复合项 $\sum_{t=0}^{T-1} h_t(\boldsymbol{K}_t \boldsymbol{x})$ 的和。函数 $f_i \colon \mathbb{X} \to \mathrm{R}$ 和 $g_m \colon \mathbb{X} \to \mathrm{R}$ 的和现在分别表示光滑函数和近端可计算函数的和。与之前一样， $f_i, i = 0, \ldots, N-1$ 和 $g_m, m = 0, \ldots, M-1$ 是适当、闭且凸的函数；此外，我们假设函数 f_i 是 L_i 光滑的。

式（7）的有限和设计是随机优化起源背后的一个关键概念，其更新步骤通过对上述函数进行**采样** 来计算。

举一个具体的例子，在断层成像重建中（更多详情请参见第 4 节），我们可以将保真项 f表示为有限和，方法是将采集数据 $\boldsymbol{b}$划分为一组测量数据（$\boldsymbol{b}_i$）$_{i=0}^{N-1}$，其中每个 $\boldsymbol{b}_i$通常称为一个子集。对于断层成像，一种自然的分区方式是将数据按投影角度划分为若干子集。我们可以定义相应的前向算子 $\boldsymbol{A}_i$。然后，对于每个 $i \in 0$、.....、 $N-1$，我们定义 $f_i(\boldsymbol{x}) = \|\boldsymbol{A}_i \boldsymbol{x} - \boldsymbol{b}_i\|^2$，然后

$$f(\boldsymbol{x}) = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2 = \left\| \begin{pmatrix} \boldsymbol{A}_0 \boldsymbol{x} - \boldsymbol{b}_0 \\ \vdots \\ \boldsymbol{A}_{N-1} \boldsymbol{x} - \boldsymbol{b}_{N-1} \end{pmatrix} \right\|^2 = \sum_{i=0}^{N-1} \|\boldsymbol{A}_i \boldsymbol{x} - \boldsymbol{b}_i\|^2 = \sum_{i=0}^{N-1} f_i(\boldsymbol{x}). \tag{8}$$

请注意，在我们专注于逆问题的框架中，我们考虑将 f分解为有限和结构 $f = \sum_{i=0}^{N-1} f_i$；而不是像机器学习应用中那样，通常将 f表示为损失函数 $f = \frac{1}{N} \sum_{i=0}^{N-1} f_i$ [56]的平均值。因此，与原始文献相比，CIL 中的随机算法有时需要进行细微修改。

2.2.1 随机梯度下降。考虑式（7）的最简单情形，其中 g_m， $h_t \equiv 0$，因此

$$x^* \in \arg\min_{\boldsymbol{x} \in \mathbb{X}} \sum_{i=0}^{N-1} f_i(\boldsymbol{x}). \tag{9}$$

这是可微的，因此可以使用梯度下降（算法 1）求解。对于“标准”梯度下降，每次迭代都会计算完整的梯度，即对所有函数 f_i以及全部数据计算梯度，因此在算法 1 中， $\boldsymbol{G_k}(x) = \sum_{i=0}^{N-1} \nabla f_i(x)$。随机梯度下降（SGD）算法的执行方式如下：

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \gamma \nabla f_{i_k}(\boldsymbol{x}_k), \tag{10}$$

其中 γ是步长， i_k从 0、1、.....、 $N-1$中采样。这与算法 1 等价，但 $\boldsymbol{G_k}(x) = N \nabla f_{i_k}(x)$ 是 $\nabla f(x)$ 的无偏近似。这是本文和软件中的**核心概念**：随机梯度算法可以通过加入**梯度的随机估计器**，以确定性算法的形式实现。

随机梯度下降算法不是一种下降法，即 f可能不会在每次迭代中都减小。不过，从期望来看，式（10）中的迭代点正沿着负全梯度的方向移动。随机算法的核心思想是：虽然达到同一个最小值可能需要更多次迭代，但如果随机梯度估计器的计算成本更低（理想情况下，仅为原计算成本的 N 分之一），那么随机算法总体上会降低优化的计算成本。

举一个具体的例子，考虑方程 8 中描述的 f_i。在确定性算法的每次迭代中，计算 $\boldsymbol{G_k}(x) = \sum_{i=0}^{N-1} \nabla f_i(x)$，都需要执行前向运算 $A_i x$（通常还需要计算其伴随）

随机梯度估计器	**CIL 近似梯度函数**	**梯度计算**	**初始化**
随机梯度下降（SGD）	SGFunction	$G_k(\mathbf{x}) = N\nabla f_{i_k}(\mathbf{x})$， 其中 i_k 从 $\{0, 1, \ldots, N-1\}$。	不适用
随机平均梯度（随机平均梯度）[57]	SAG函数	$G_k(\mathbf{x}) = \nabla f_{i_k}(\mathbf{x}) - \delta_{i_k}^k + \sum_{i=0}^{N-1}\delta_i^k$, $\delta_j^{k+1} = \begin{cases}\nabla f_j(\mathbf{x}), & \text{如果 } j = i_k,\\ \delta_j^k, & \text{否则,}\end{cases}$ 其中 i_k 从 $\{0, 1, \ldots\ldots, N-1\}$。	初始化 $\delta_j^0 = 0$ 或热启动——以 $\nabla f_j(\mathbf{x}_0)$ 作为 $j = 0, \ldots, N-1$ 的起始值。
SAGA（随机平均梯度-改进）[58]	SAGAFunction	$G_k(\mathbf{x}) = N(\nabla f_{i_k}(\mathbf{x}) - \delta_{i_k}^k) + \sum_{i=0}^{N-1}\delta_i^k$, $\delta_j^{k+1} = \begin{cases}\nabla f_j(\mathbf{x}), & \text{如果 } j = i_k,\\ \delta_j^k, & \text{否则,}\end{cases}$ 其中 i_k 从 $\{0、1、\ldots\ldots、N-1\}$ 中采样。	初始化 $\delta_j^0 = 0$ 或热启动使用 $\nabla f_j(\mathbf{x}_0)$
随机方差缩减梯度 (SVRG) [59]	SVRG函数	$G_k(\mathbf{x}) = \begin{cases}N\nabla f(\bar{\mathbf{x}}_k), & \text{如果 } k \text{ 模 } T = 0,\\ N(\nabla f_{i_k}(\mathbf{x}) - \nabla f_{i_k}(\bar{\mathbf{x}}_k)) + \nabla f(\bar{\mathbf{x}}_k), & \text{否则,}\end{cases}$ $\bar{\mathbf{x}}_k = \begin{cases}\mathbf{x}, & \text{如果 } k \bmod T = 0,\\ \bar{\mathbf{x}}_{k-1}, & \text{否则,}\end{cases}$ 其中 i_k 从 $\{0, 1, \ldots, N-1\}$ 中采样。	选择快照周期 T
无循环随机方差缩减梯度法（循环随机方差缩减梯度）[60–62]	LSVRG函数	$G_k = \begin{cases}N\nabla f(\bar{\mathbf{x}}_k), & \text{若 } c < p,\\ N(\nabla f_{i_k}(\mathbf{x}) - \nabla f_{i_k}(\bar{\mathbf{x}}_k)) + \nabla f(\bar{\mathbf{x}}_k), & \text{否则,}\end{cases}$ $\bar{\mathbf{x}}_k = \begin{cases}\mathbf{x}, & \text{如果 } c < p,\\ \bar{\mathbf{x}}_{k-1}, & \text{否则,}\end{cases}$ 其中 i_k 从 $\{0, 1, \ldots, N-1\}$ 中采样 以及 c 采样的 $U[0,1]$.	选择 $p \in (0,1]$ (快照概率)

表 2。CIL 中的随机梯度估计器。

for all $i \in \{0, \text{fi}, N-1\}$ performi计算 i在一个上的 ll the data i在每个h i_{terat} $\text{i}^{\text{on, w}}$ hile the SG $\text{stochasti}^{\text{c est}}$ i

量器，$G_k(x) = \nabla f_{i_k}$，仅使用一个采样的 A_{i_k}（以及对应的数据 b_{i_k}）对数据的一个子集进行计算。为确保确定性方法与随机方法之间的比较公平，通常以“数据遍历次数”而非迭代次数来绘制并讨论收敛情况；一次数据遍历对应于期望意义上查看数据的每个子集一次。对于 SGD，一次数据遍历中期望有 N 次迭代。这与机器学习社区常用的“轮次”相当。

*2.2.2 方差缩减梯度方法。*表 2 列出了目前 CIL 中的所有随机估计量。第一行介绍随机梯度下降估计量。如上所述，在算法 1 中使用这一随机估计器，可得到公式 10 中的迭代。

文献中有多个随机方法的示例，现在我们可以将其写成随机梯度估计器 G_k的形式，并将其插入确定性算法中，从而构造随机算法。通过结合平均化和全梯度计算，这些方法使用全梯度$G_k \approx \nabla f(x_k)$的随机估计，其误差小于 SGD，因此与 SGD 相比收敛速率更快，步长也更容易调整。这种改进的代价是，与 SGD 相比，计算成本有所增加，和/或需要额外的存储空间。下文将讨论 CIL 中目前提供的方法，这些方法列于表 2 中。

• **随机平均梯度：**第一个方差缩减方法是随机平均梯度（SAG）方法 [57], ，它是增量聚合梯度法的随机扩展， [63]。SAG 估计器（见表 2）需要存储一个包含 N 个随机梯度 δ_i^k的表，并将其初始化为零；例如，这些梯度会在每次迭代中使用。每一步中，只需计算一个随机抽取函数的梯度，即可更新表中的一个位置。这样，每次迭代的计算成本便与随机梯度下降算法相近。SAG 是第一个具有与全梯度方法相近收敛速率的算法。 [57] 中最初的收敛性分析仅考虑了光滑目标函数。后来，SAG 的分析扩展到了具有近端支持 [64]的非光滑目标函数。

• **SAGA：** 在 [58], 中，作者介绍了 SAG 的一个“改进”版本，称为 SAGA，其中额外的“A”代表法语 改良。按照上文讨论的约定，与 SAG 相比，表 2 中的 SAGA 梯度估计器在随机梯度与存储的梯度之差前乘以权重 N δ_i^k。SAG 使用的是 f全梯度的有偏估计器，而 SAGA 的缩放方式经过调整后，使其成为无偏随机估计量。

• **SVRG：** 为所有 $i \in \{0、1、\ldots\ldots、N-1\}$ 存储梯度表可能需要大量存储空间，这对于逆成像问题（例如断层重建应用） [65]尤为明显。在 [59], 中，作者介绍了随机方差缩减梯度（SVRG）算法，最初用于光滑目标函数，后来也用于具有近端支持的优化问题 [66]。SVRG 算法会存储一幅参考图像 $\tilde{x}_k$（也称为“快照”）以及在该参考图像处计算出的全梯度，并每隔 T 次迭代更新一次。调用 SVRG 的梯度估计器时，会发生以下两种情况之一：每隔 T 次迭代采集一次快照并计算其全梯度；在其他每次迭代中，则只计算一个子集的梯度，并利用先前计算出的全梯度信息来估算梯度。与 SAG 和 SAGA 相比，SVRG 降低了内存需求，但引入了一个额外参数 T，需要仔细调节。从计算成本来看，SVRG 除了每隔 T 次迭代计算一次全梯度外，每一步还要计算两个梯度。

• **LSVRG：** SVRG 的一种常见变体是无循环随机方差缩减梯度法（LSVRG） [60–62] ，它用一个概率 p 取代每隔 T次迭代获取快照的做法：以该概率获取快照并返回全梯度。每次更新时，以概率 p 更新参考点并计算全梯度；否则，参考点保持不变，同时无需计算全梯度。

文献中还有其他随机梯度估计器，例如，随机递归梯度算法（SARAH）是 SVRG 的一种有偏版本，后来也扩展为无循环 SARAH [67]，并且

在 [68] 中支持近端算子。后者这些算法尚未在当前版本的 CIL 中实现。不过，我们的库提供了必要的构建模块，可以轻松构建这些算法。

引入随机性的一种方法是对式（7）中的最后一项（即复合项之和）进行随机选择。为此，优化问题可写为

$$\boldsymbol{x}^* \in \arg\min_{\boldsymbol{x}\in\mathbb{X}} f(\boldsymbol{x}) + g(\boldsymbol{x}) + \sum_{t=0}^{T-1} h_t(\boldsymbol{K}_t\boldsymbol{x}). \tag{11}$$

随机原始-对偶混合梯度（SPDHG） [69]算法并非通过近似梯度引入随机性，而是将空间 $\mathbb{Y}$ 分解为乘积空间 $\prod_{t=0}^{T-1}\mathbb{Y}_i$，并在每次迭代中仅更新对偶变量的一个子集。CIL 中针对 $f=0$实现了 SPDHG，我们将其用作对比方法。

2.3 即插即用框架：将随机梯度估计器与确定性算法相结合

算法 / 函数	梯度下降（算法 1）	近端梯度下降（算法 2）	加速近端梯度下降（算法 3）	PD3O（算法 4）
SGFunction	随机梯度下降 [70]	近端随机梯度下降 [71]	加速近端随机梯度下降 [72]	PD3O-SGD [51]
SAG函数	随机平均梯度 [57]	近端随机平均梯度 [64]	加速近端随机平均梯度 [73]	PD3O随机平均梯度 [51]
SAGAFunction	SAGA [58]	ProxSAGA [58]	加速近端随机平均梯度算法 [73]	PD3O随机平均梯度算法 [51]
SVRG函数	SVRG [59]	ProxSVRG [66]	加速近端随机方差缩减梯度 [73]	PD3O-SVRG [51]
LSVRG函数	循环随机方差缩减梯度 [60]	ProxLSVRG [62]	加速近端循环随机方差缩减梯度 [73]	PD3O-LSVRG [51]

表3. 随机优化算法概览。每一行对应一个实现特定随机梯度近似的 CIL函数类，每一列则代表一种特定的确定性CIL算法。

在上一节中，我们讨论了如何将算法 1 中定义的 GD 与第 2 节中的随机估计器结合，得到 SGD、SAG、SAGA、SVRG 和 LSVRG 算法。

在表3中，我们展示了不同的随机估计器如何与表1中的确定性算法相结合，从而得到一系列随机算法。

此外，对于方程（7）中 $M=1$ 和 $h\equiv 0$ 的情形，我们可以将随机估计器与 PGD（算法2）和 APGD（算法3）相结合，得到随机近端方法和加速方法，相关参考文献列于表中。

最后，我们再次考虑存在复合项 $h(\boldsymbol{K}\boldsymbol{x})$时的方程（7）。为简单起见，我们考虑 $M=T=1$的情况。我们可以通过再次将梯度替换为以下内容来扩展PD3O算法（算法4）：

使用核心成像库求解逆问题的随机优化模块化方法

一种随机梯度估计器。在 [74], 中，针对 $K = \mathbb{I}$ 介绍了三算子分裂方法，并在 [75] 中将其扩展至用于随机梯度的 $K \neq \mathbb{I}$。最近，在 [51], 中，PD3O 与其他算法一起采用了 SAGA 和 SVRG 等随机方差缩减估计器。

3 软件解决方案：分步介绍

本节将逐步介绍我们的软件解决方案。接下来的两节将展示该软件在真实世界数据集上应用于多种成像任务的情况，以验证其通用性和实际价值。在补充材料部分，我们还提供了一个非成像任务的示例。

一种随机 算法，只要确定以下所有内容，就可以对上一节所述的算法进行完整定义 ：

(1) **数据分区器**：如何将数据集划分为子集；(2) **采样器**：随机梯度估计器的采样策略，用于选择要处理的子集；(3) **随机梯度估计器，** G_k；(4) **确定性算法**：基于梯度的确定性算法（GD、PGD 或 APGD），或 PD3O 算法；(5) **可选地，**定义预条件和步长策略，并为 APGD 选择加速技术，例如动量。

我们的软件解决方案支持定义上述全部 5 个要点，提供全新的 CIL 功能，以灵活地对计算机断层扫描数据进行分区并采用通用采样策略，同时提供灵活且可扩展的框架，将随机梯度估计器集成到确定性算法中，从而构建随机优化方法。

(1) **数据分区器：**在所介绍的随机框架中，我们着重将（3）中的可微项（第一项）拆分为 N 个可微项 $\{f_i\}$ 之和，参见（ 7 ）中的第一个求和项。为了使随机算法带来切实的加速，每个 f_i 在每次迭代中都必须仅处理数据的一个子集，因此，要将 f 分区为有限和，就必须对数据进行分区。具体的分区方式取决于特定的数据域以及 f 的形式。我们将在第 4.2.1 节介绍 CIL 中实现的 X射线计算机断层扫描数据分区，并在第 5.2.2 节简要介绍正电子发射断层成像数据分区。本节余下部分将定义数据分区器生成的函数列表：

分区函数

```
列表_fi = [f_1, ....., f_N]
```

(2) **采样器：**

表 2 中定义的所有随机梯度估计器在每次计算时都需要从 $\{0, 1, \ldots\ldots, N-1\}$ 中采样一个 i_k。在 CIL 中，我们定义了一个负责在每次迭代时生成索引 i_k 的采样器类。下面的代码片段展示了几种用户可用的采样策略。图1展示了各策略在多次迭代中采样所得索引的散点图和直方图。

采样策略

```
N = 30要采样的索引数量
采样器_顺序 = 采样器.顺序(N)
采样器_s06 = 采样器.交错(N, 步长=6)
采样器_s12 = 采样器.交错(N,          步长=12)
采样器_rwr = 采样器.随机_有_放回抽样(N, 随机种子=6)
采样器_rnu = 采样器.随机_有_放回抽样(N, 随机种子=6,
                                概率= [0.9/20]*20 + [0.1/10]*10)


采样器_rwo = 采样器.随机_无_放回抽样(N, 随机种子=6)
采样器_her = 采样器.herman_meyer(N)
def my_采样(iteration_number):
    return np.floor(N*np.sin((np.pi/(2*N))*迭代_number)**2)

采样器_cus = 采样器.from_function(num_索引=N,          function=my_采样)
```

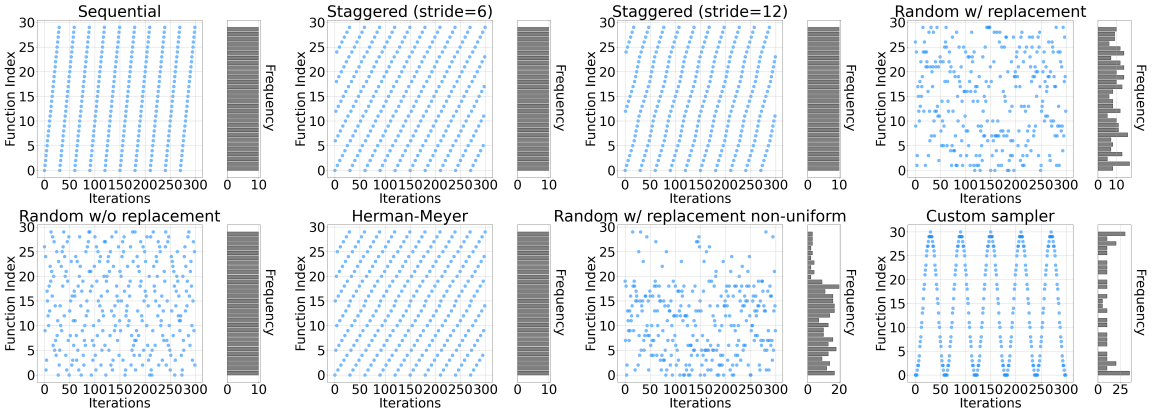


图 1. 展示 300 次调用采样器（10 次数据遍历）中对 30 个索引采用不同采样策略的图。散点图旁边绘制了每个索引在这 300 次调用中出现的频率。

这些示例包括有放回和无放回的随机采样，分别通过 random_with_replacement 和 random_without_replacement 实现。对于有放回采样，初始化采样器时可以指定非均匀概率，前提是这些概率均为正且总和为 1。对于顺序采样（sequential），索引按顺序选取。交错采样（staggered）是顺序采样的一种变体，使用预定义步长循环选取索引。Herman-Meyer 采样方法由 [76] 提出，并用于 [77],，主要应用于断层成像。该方法要求函数数量 f_i 不是素数，因为它基于素数分解。在大多数情况下，采样器会直接传递给随机梯度函数或算法。不过，用户也可以使用 next()[1] 方法直接从中采样。

[1]https://docs.python.org/3/library/functions.html#next

我们的采样器类与其他开源库中的采样器模块有相似之处。例如，在 PyTorch 中，采样器在数据加载器类中使用，以实现高效的数据批处理和随机打乱。类似地，JaxOpt [78]，是一个用于 JAX 的开源优化库，它提供了一个直接附加到相应算法的数据迭代器。相比之下，我们的方法将采样器与算法类及输入数据解耦。相反，采样器直接附加到随机梯度估计器上，从而提供更大的灵活性和模块化程度。

(3) 随机梯度估计器，G_k： 下一步是选择一个随机梯度估计器，用于近似函数 f 的梯度，即分区函数 $\{f_i\}$ 之和。我们没有为表 2 中列出的每种方法分别创建 CIL 算法类，而是引入了一个新的抽象基类：ApproximateGradientSumFunction，它是 CIL SumFunction 类的子类。该类是 CIL 中随机优化的核心组件，支持实现依赖不同梯度近似策略的各种算法。

该类使用函数列表 f_i 和一个随机采样器实例 sampler_xyz 进行初始化，该采样器可以是上文介绍的任意一种，也可以是自定义采样器。调用该函数时，在点 x 处，它会返回所有 $f_i(x)$ 的和。其关键特性是梯度类方法，该方法会调用一个抽象的近似_梯度方法。派生类的行为类似于 CIL 函数类，可以在近似_梯度方法中实现表 2 所定义的随机近似梯度。

对于用户而言，这些机制是隐藏的；他们只需将自己的 Python 函数列表 list_fi $\{f_0, f_1, \ldots\ldots, f_{N-1}\}$ 以及选定的随机采样器传递给所选的随机梯度估计器，例如用于 SG

初始化 SGFunction

```
f_SG =SGFunction(列表_fi，采样器=采样器_xyz)
```

(4) 确定性算法： 根据式 (7) 中定义的目标函数的性质，用户可以从以下选项中选择：

- 算法 1，GD，适用于光滑目标函数，
- 算法 2 和算法 3：PGD 和 APGD，适用于可拆分为光滑项和近端可计算项的目标函数，
- 算法4，PD3O，适用于可拆分为光滑项、近端可计算项以及具有近端可计算的凸共轭的附加复合项的目标函数。

在下面的两个代码片段中，我们从一个 CIL 函数 f 开始，它已被划分为由N 个 CIL 函数组成的列表 $\{f_i\}$。对于确定性算法，我们选择使用固定步长 gamma 的梯度下降，该步长取决于函数 f 的 Lipschitz常数。对于随机算法，我们首先设置近似梯度函数，将函数列表和上文展示的任一采样器作为输入。然后将其与固定步长 gamma 一同传入随机梯度下降算法；该步长取决于近似梯度函数 f_SG 的 Lipschitz常数，即 $\gamma = \frac{1}{\sum_{i=0}^{N-1} L_i}$。随后，两种方法都以相同的方式运行 K 次迭代。默认情况下，两种算法都以零图像初始化，但也可以使用自定义初始化。

梯度下降 (GD)

```
伽马  = 1/f.L
梯度下降  = 梯度下降(f=f, 步长_大小=伽马)

梯度下降.运行
(K)
```

随机梯度下降 :- 梯度下降 +SGFunction

```
f_SG =SGFunction(list_fi, 采样器=采样器_xyz)
伽马  = 1/f_SG。L
随机梯度下降  = 梯度下降(f=f_SG，步长_大小=伽马)

随机梯度下降。
运行(K)
```

只需稍作调整，例如更改随机梯度估计器或采样策略，就能以极少的工作量测试新的方法和算法。这体现了我们软件的即插即用特性：只需对代码进行少量修改，即可灵活开展实验。

4 平行束计算机断层扫描重建的随机优化

本节将展示如何将我们新颖的随机优化框架应用于一项实际成像任务：计算机断层扫描（CT）重建。具体而言，我们使用了一个通过平行束几何采集的真实三维 CT 数据集。虽然平行束 CT 数据通常按切片逐片重建，但我们选择一次性重建整个体积，以便利用跨切片的三维全变分（TV）正则化。这种方法提供了额外的正则化效果，而如果将各切片独立处理，则无法实现这一点。

真实世界的三维 CT 数据集因分辨率高、数据量大而面临独特挑战，因此需要采用高效的优化技术，以确保重建准确且可扩展。为此，我们引入了用于划分 CT 数据集的专用工具，助力用户便捷地开展大规模断层重建。

本节首先以 CT 应用为例说明如何对数据进行分区，然后介绍如何在 CIL 中创建所需的近似梯度和函数。在此应用中，我们旨在最小化一个包含不可微项的目标函数，此处该项为全变分正则化。因此，我们在近端方法或原始-对偶方法中使用这些近似梯度和函数，因为这些方法能够处理不可微部分。随后，我们展示这些随机算法的性能，并比较多种随机方法和确定性方法在真实三维数据重建中的表现。我们还比较了随机方法中采用不同子集数量时的效果。

4.1 数据集

我们使用了对一节 10440 NMC532 锂离子电池 [79] 进行的 X 射线 CT 测量数据，该测量在 Diamond Light Source 的 I12 光束线使用 100 keV 单色 X 射线束 [80]完成。该数据集已开源，可通过电池成像库（BIL）获取， [81]。射线照片由 X 射线成像相机（PCO.edge）记录，像素尺寸为 7.91 μm。每次扫描包含 1800 个投影（射线照片），覆盖 0°–180° 的角度范围，每帧采集时间为 8 ms。探测器尺寸为 1560×1560。在 CT 测量之前，还采集了平场图像（50 张）和暗电流图像（50 张），并用于在断层重建之前对采集的射线照片进行归一化。此外，我们应用了预处理程序， [82], 以去除

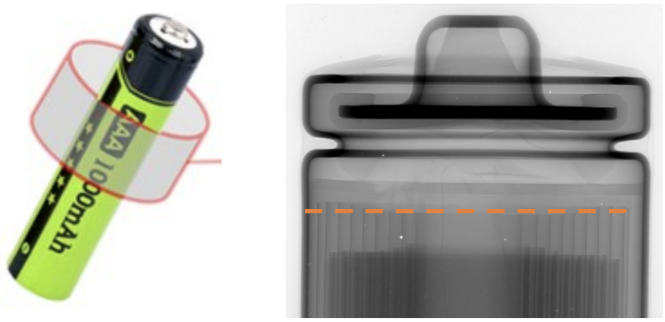


图 2. 以容量倍率 20 放电的锂离子 10440 NMC532 电池（左），以及电池顶部的射线照片（右）。

正弦图空间中的垂直条纹，这些条纹通常会在重建空间中形成环状伪影。对于下文所展示的数值实验，我们使用了一个裁剪后的数据集，其中包含 900 个投影角度和 300 个垂直切片。

4.2 方法

我们考虑用于 CT 重建的优化问题是带非负约束的全变分正则化。采集数据记为 $\boldsymbol{b}$，见图 3a； $\boldsymbol{A}$ 是通过 ASTRA 和 TIGRE 库之一并借助 GPU 加速获得的拉东变换离散版本，我们已将其封装在 CIL 的 ProjectionOperator 类中。

$$\arg\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\boldsymbol{b}\|_2^2 + \alpha \mathrm{TV}(\boldsymbol{x}) + \mathbb{I}_{\{\boldsymbol{x}>0\}}(\boldsymbol{x}). \tag{12}$$

将最小二乘函数拆分为和（如式 (8) 所示），可得到如下形式

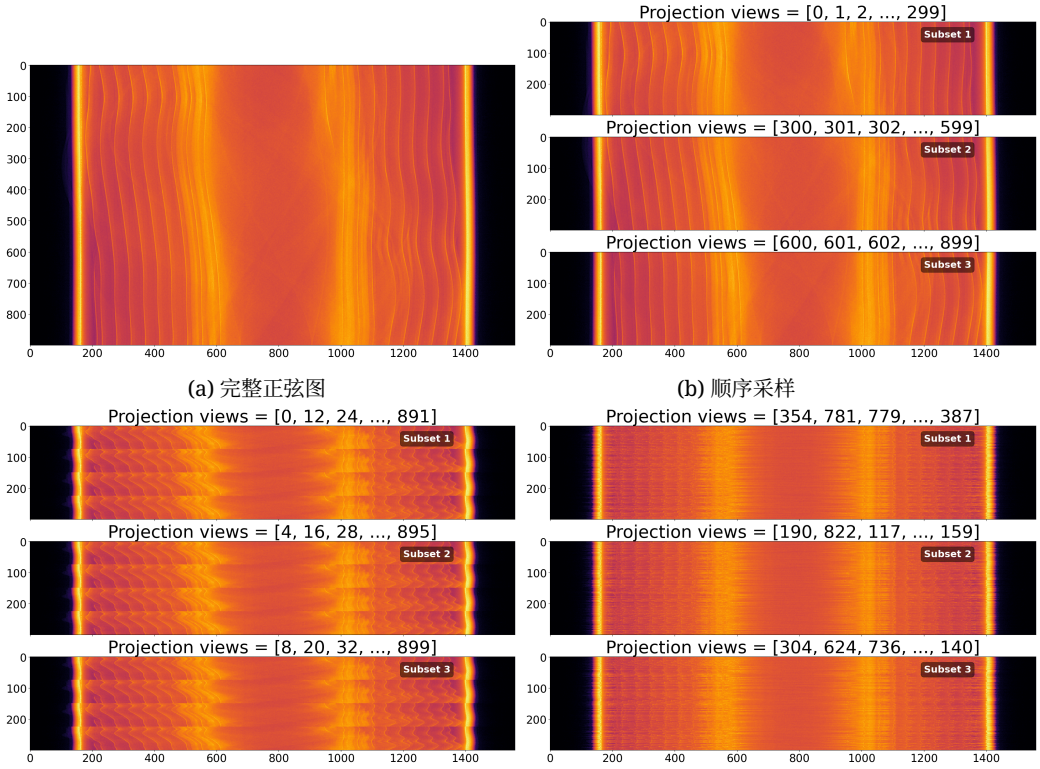
$$\arg\min_{\boldsymbol{x}} \frac{1}{2}\sum_{i=0}^{N-1}\|\boldsymbol{A}_i\boldsymbol{x}-\boldsymbol{b}_i\|_2^2 + \alpha \mathrm{TV}(\boldsymbol{x}) + \mathbb{I}_{\{\boldsymbol{x}>0\}}(\boldsymbol{x}). \tag{13}$$

所有实验均将正则化参数 α 设为 0.75。

4.2.1 断层摄影数据集的分区。本文提出的随机框架中，一个关键部分是针对断层成像数据的分区策略，该策略考虑了底层 CT 几何结构。在基础 CIL AcquisitionData 类中，我们引入了一种分区方法，将数据分成一个采集数据列表，并返回一个 BlockDataContainer。数据根据投影角度拆分为用户指定数量的分区，不同角度则根据用户定义模式分配到不同的组中。以下代码片段展示了用户如何将采集数据 b 拆分为长度为 N 的数据列表 list_bi，并以交错方式将角度分配到 N 个分区（来自投影角度 l，$l+N$，$l+2N$，$l+3N$，fi 的数据分配到第 l 个分区）。

数据分区方法

```
列表_bi  = b。partition(N,mode='交错')
```



(a) 完整正弦图　(b) 顺序采样

(c) 交错采样　(d) 随机采样

图 3. 锂离子电池正弦图的不同数据划分模式。在每个示例中，我们将数据分成 3 个子集，每个子集包含 300 个数据。我们选择将分区后的数据集显示为二维正弦图，但提醒读者这些是三维数据。

在图 3 中，我们展示了对断层成像数据进行分区的不同模式。我们以二维形式绘制每个分区中的投影视图，以展示交错、顺序和随机分区。请注意，由于样品具有对称性，交错和随机分区得到的各个分区正弦图彼此看起来都很相似；每个分区都包含广泛的信息，而对于顺序分区的数据，每个分区则大不相同。一般而言，建议采用这样的方式对数据进行分区，使每个分区都包含关于整个体模的信息，即采用交错或随机分区。

我们定义相应的投影算子 $(\boldsymbol{A}_i)_{i=0}^{N-1}$，并使用与每个 $\boldsymbol{b}_i$ 相关联的几何信息。CIL 投影算子可以接收一个 BlockDataContainer 采集几何，并返回一个 BlockOperator。此外，我们构造一个函数列表 $(f_i)_{i=0}^{N-1}$，用于表示式（13）中的有限和项。

使用核心成像库求解逆问题的随机优化模块化方法

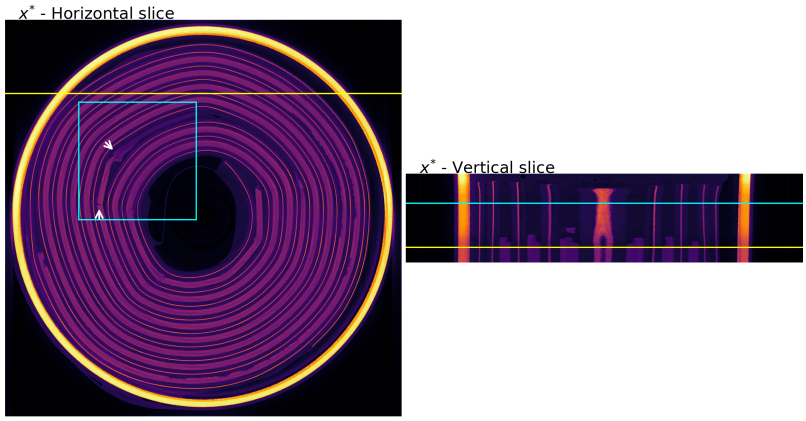


图4. CT电池数据集的最优三维重建，用于比较。水平切片上的蓝色方框和垂直切片上的蓝线用于图7a和图7b中的比较。黄色线标出了两幅图像之间的对应关系——也就是说，左侧图像中的黄色线对应右侧图像中的垂直切片。

*fi*系列

```
A_i =投影算子(图像_几何,                    列表_bi.几何)
列表_fi = []
对于i属于范围(N):
    列表_fi。追加(LeastSquares(A=Ai[i], b=列表_bi[i], c=0.5)
```

This 将 Ai 保留为 CIL块算子，将 list_fi 保留为函数列表，并将 list_bi 保留为 BlockDataContaine r.

4.2.2 实验设置与参数选择。 这个真实世界数据集没有真实图像，因此我们改为为确定性优化问题（12）构造一个最优解。

对于最优解，我们使用带有 Nesterov 动量的 APGD算法，即算法3，其中 TV近端算子使用应用于对偶 TV去噪问题的快速梯度投影（FGP）算法计算 [14]，该算法通过 CCPi正则化工具包中的 GPU 加速实现 [83] [84]。我们对 FGP 算法进行 300 次内迭代，以减少这种近端算子的隐式计算所造成的任何偏差。我们运行（外层）APGD算法 5000 次迭代。每次迭代都对应于对数据进行一次完整遍历（即一次“数据遍历”），因此总共进行 5000 次数据遍历，以确保收敛。我们选择步长为 $\frac{1}{L}$；最优重建的水平切片和垂直切片见图4。我们将这一参考优化算法与所提出框架中的一系列不同确定性和随机优化算法进行比较，分别用于最小化（12）或（13）。下文给出的参数选择旨在获得稳定且具有代表性的性能，并展示我们新软件设计的功能；若要进行完全优化的比较，则需要对分区大小、采样策略、步长、更新频率和概率进行微调，这超出了本文的范围。

对于随机优化比较，我们按照上述步骤考虑以下因素：

(1) **分区**：我们比较两种分区方式：交错和顺序。我们还比较不同数量的分区：10、30 和 50 个。(2) **采样器**：对于所有随机估计器，我们使用随机_有放回采样器，以确保比较公平，并与文献中的收敛性证明保持一致。(3) **随机梯度估计器**：我们选择以下随机梯度估计器、初始化方式和更新选项：

- ProxSAGA：此外，我们通过计算每个函数在 f_i 初始点处的梯度，对 SAGA近似进行“热启动”。
- ProxSVRG：我们每隔 $2N$进行一次全梯度更新并记录快照。
- ProxLSVRG：在每次迭代中，进行全梯度更新的概率为 $1/N$。

（4）**确定性算法**：对于我们的确定性算法，我们选择 PGD 算法（算法2）和 PD3O算法（算法4），因为二者都允许存在不可微项，例如 TV范数或非负约束。对于 PGD 算法，我们必须选择一个步长，并分别使用 $\frac{1}{3\tilde{L}}$、 $\frac{1}{\tilde{L}}$和 $\frac{1}{\tilde{L}}$作为˜SAGA、SVRG 和 LSVRG 随机梯度估计器，其中 L 是 f_0的 Lipschitz常数乘以函数数量。对于 PD3O，步长按照 CIL PD3O算法中的默认值进行选择。具体而言，采用 [85], 中的术语，原始步长为 $\gamma = \frac{0.99*2.0}{L}$，其中 L 是 f的 Lipschitz常数；对偶步长为 $\delta = \frac{L}{\|\boldsymbol{K}\|^2}$，其中 $\boldsymbol{K}$是与 PD3O 中的函数 h 复合的算子，在我们的例子中，该算子是乘以正则化参数的梯度算子。

综合以上内容，以下代码块展示了 PGD-SAGA 和 PD3O-SGD 的设置。在第一个代码块中，我们首先选择随机_有_放回采样器，这与我们对每种随机近似所做的一样。第二行和第三行定义了一个采用 SAGA 近似的随机近似梯度函数，并通过计算初始点处每个函数的梯度 f_i 来对 SAGA 近似进行“热启动”。每种算法，无论是确定性的还是随机的，都使用一个全零数组初始化。随后，代码块设置 TV 函数 G，并启用非负性约束，设置 100 次内迭代。SAGA 的步长为 $\frac{1}{3\tilde{L}}$。最后两行中，我们使用随机近似梯度函数设置 PGD，并运行指定次数的迭代，同时使用回调函数在每次数据遍历时计算均方误差。

ProxSAGA设置

```
采样器 =采样器.随机_有放回_(列表长度_fi), 随机种子=40) F_saga =
SAGAFunction(列表_fi, 采样器=采样器) F_saga.预热_启动_近似_梯度(初始值)
G =alpha * FGP_TV(最大_迭代次数=100, 设备='图形处理器', 非负性=True) 步
长_大小 = 1/(3*F_saga.L) 近端_saga = PGD(初始值=initial, f=F_saga , g=G,
步长_大小=step_size, 更新_目标函数_间隔=N) 近端_saga.run(迭代次数=K, 回调
=[度量_回调])
```

PD3O-SGD 设置也采用类似的方法，不过我们不使用正则化工具包中的 TV 函数，而是显式定义梯度算子和混合 L21 范数，并将它们传递给算法。

使用核心成像库求解逆问题的随机优化模块化方法

PD3O-SGD 设置

采样器 =Sampler.random_with_replacement(len(list_fi), seed=40) F_sgd = SGFunction(list_fi, sampler=sampler) G =IndicatorBox(lower=0) 算子 = alpha*GradientOperator(图像_几何) H =L1范数() pd3o_sgd = PD3O(initial=initial, f=F_sgd, g=G, h=H, 算子=operator, update_目标函数_ interval=N) pd3o_sgd.run(迭代=K, callbacks=[metric_callback])

我们还将随机算法与一些确定性算法进行比较:

- 加速近端梯度下降 - 虽然最优解是用加速近端梯度下降计算的，但为了更真实地模拟加速近端梯度下降的运行，我们只计算 50 次迭代，相当于 50 次数据遍历次数；TV近端算子仅通过 100 次内迭代计算。我们采用 $1/L$ 的步长，其中 L 是 f 的 Lipschitz常数。
- PD3O：我们使用 PD3O，设置方式与随机版本相同，但采用确定性的 f。同样，我们使用 CIL 中的默认步长。

为了比较结果，我们考虑两个指标：第一个是确定性方法和随机方法分别对应的优化目标，即式 (12) 或 (13)。第二个是重建体积 $\boldsymbol{x}$ 与最优体积 $\boldsymbol{x}^*$ 之间的归一化均方根误差,

$$NRMSE(x) = \frac{\|\boldsymbol{x} - \boldsymbol{x}^*\|_2}{\|\boldsymbol{x}^*\|_2}. \tag{14}$$

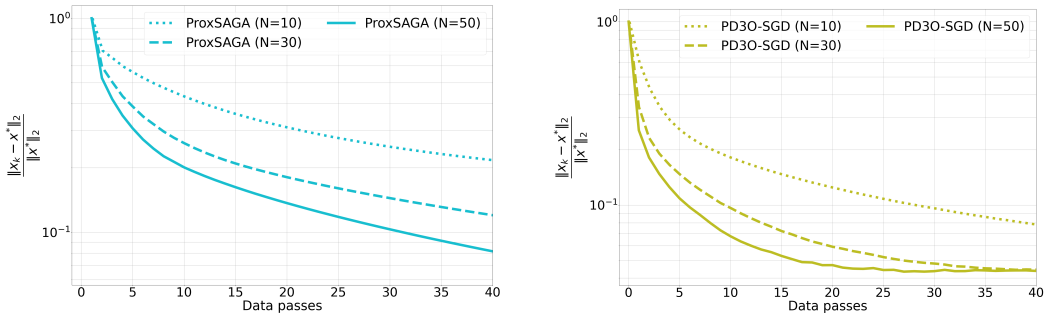
对于确定性方法，这些指标在每次迭代时计算并保存；对于随机方法，则在每次数据遍历时计算并保存。

4.3 结果

4.3.1 子集数量。 我们首先比较随机算法中子集数量的不同取值。图 5a 和图 5b 分别展示了 SAGA 和 PD3O-SGD 随机方法的归一化均方误差随数据遍历次数的变化。结果表明，增加子集数量能够提高这些方法的收敛速度（以数据遍历次数衡量）。其他方法也呈现出这一趋势，但为简洁起见，此处省略相关结果。本节其余部分将分析子集数量为 50 的情况。

4.3.2 原始-对偶算法。 图 6，尤其是子图 6a 和 6b，显示，确定性算法 PDHG 和 PD3O 在均方误差上的收敛速率相近，但就目标值而言，PDHG 似乎只需较少的数据遍历次数即可收敛。PDHG 与 PD3O 之间的差异表明，仅考察目标值的收敛情况或许不足以全面反映算法表现；在其他指标可用时，考察这些指标上的收敛情况也很有价值。我们看到，随机算法 SPDHG 和 PD3O-SGD 在这两项指标上的收敛速度都快于这两种确定性算法。特别是，PD3O-SGD 在 NRMSE 指标上的表现尤为出色。

在图 6 中，尤其是子图 6a 和 6b，我们观察到确定性算法 PDHG 和 PD3O 在均方误差方面表现出相似的收敛速率。然而，就目标值而言，PDHG 似乎只需更少的数据遍历次数即可收敛，这强调了这些指标衡量的是不同的特征。随机



(a) ProxSAGA 使用 10、30 和 50 个子集　　(b) PD3O-SGD 使用 10、30 和 50 个子集

图5. 比较 SAGA 和 PD3O-SGD 算法的归一化均方误差随数据遍历次数的变化。

算法 SPDHG 和 PD3O-SGD 在这两项指标上的收敛速度都快于两种确定性算法。值得注意的是，PD3O-SGD 在 NRMSE 指标方面表现尤为出色。

4.3.3 近端梯度下降算法。 再次，在图 6 中，尤其是在子图 6c 和 6d 中，我们将确定性 APGD 算法与 ProxSVRG、ProxSAGA 和 ProxLSVRG 进行比较，后者均使用 50 个子集。经过 20 次数据遍历后，ProxSVRG 和 ProxLSVRG 的结果均达到最优解的 10% 以内，而 APGD 算法在图中所示的 40 次数据遍历内未能达到这一水平。

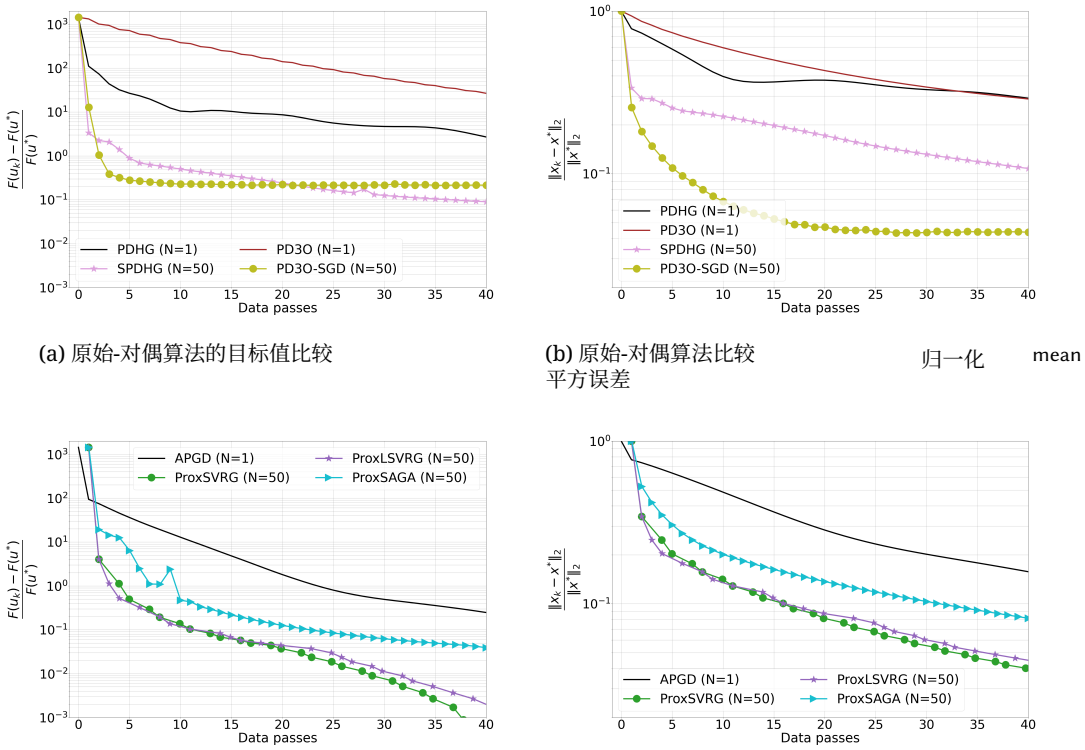
为了直观展示这一点，我们将 ProxSVRG、ProxLSVRG、ProxSAGA 和 PD3O-SGD 与确定性 APGD 算法在 10、20 和 30 次数据遍历后的结果进行比较。在图 7a 中，我们查看水平切片中感兴趣区域在 10 次和 30 次数据遍历后的图像。可以看到，经过 10 次数据遍历后，APGD 算法尚未重建出内部结构；例如，感兴趣区域左上方被截断的厚层并不可见。相比之下，所有随机算法的图像中都能看到这些结构，其中 PD3O-SGD 的重建结果最为清晰。经过 30 次数据遍历后，APGD 的重建结果中可以看到内部结构，但即便如此，其清晰度仍不如随机算法的重建结果。在图 7b 中，我们查看垂直切片横截面的折线图（参见图 4 中的线）。可以清楚地看到，与 PD3O-SGD 算法已经成功重建出最优解的所有峰值相比，APGD 的重建结果仍未能重建出中心峰值。

4.3.4 结论。 本案例研究展示了随机框架在计算机断层扫描应用中的表现，包括 CIL CT 数据分区器。我们发现，随机对应版本的表现优于确定性算法。我们还发现，子集数量的选择会产生显著影响；就数据遍历次数而言，较大的子集数量似乎能带来更好的收敛值。我们也展示了随机框架的灵活性：只需对代码进行少量修改，就能测试多种随机算法。

5 案例研究：正电子发射断层成像

在本节中，我们聚焦于正电子发射断层成像（PET）案例，这得益于 CIL 与协同图像重建框架（SIRF） [22] 的集成。与 SIRF 的集成已在 [13, 23],中介绍，其基于 Python 的鸭子类型，使我们能够在表现得类似于 CIL 的 Function、

使用核心成像库求解逆问题的随机优化模块化方法



(a) 原始-对偶算法的目标值比较

(b) 原始-对偶算法比较归一化 mean 平方误差

(c) 近端梯度下降算法的目标函数比较 (d) 近端梯度下降算法的归一化均方误差比较

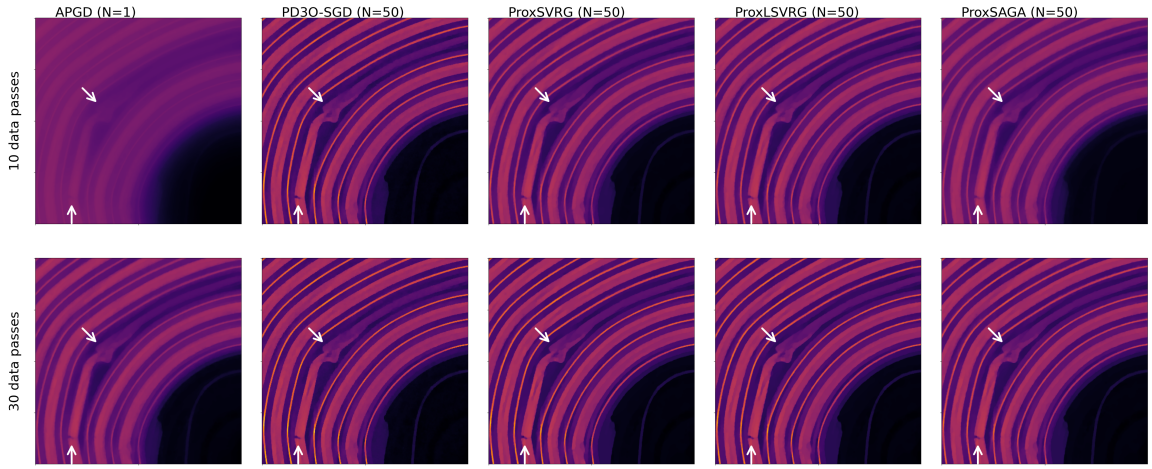
图 6. 比较近端梯度下降算法和原始-对偶算法相对于数据遍历次数的归一化均方误差和目标值。对于随机算法，我们仅考虑 50 个子集。

Operator 或数据容器的对象上使用 CIL 算法，从而扩展 CIL 原有的功能，并展示其与第三方库结合使用时的适应能力。

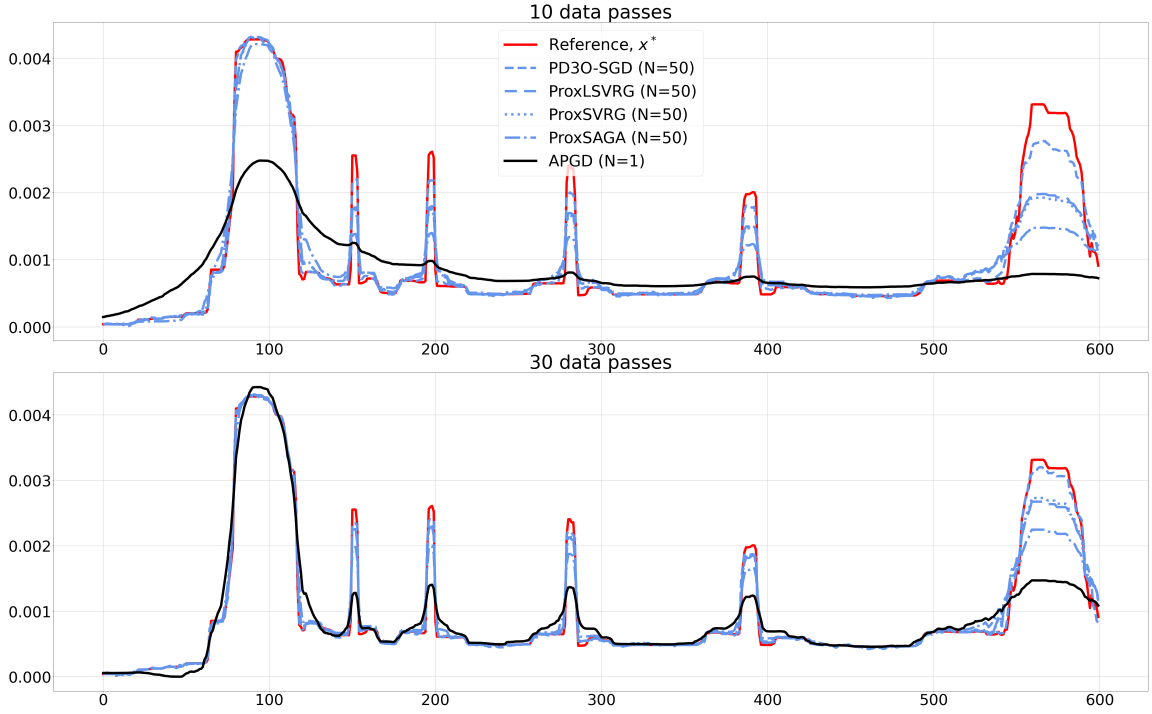
此外，我们将证明，设置随机算法来重建 PET 数据集不仅简单，而且对算法进行适当调优会对其收敛速度产生重大影响。我们将重点考察两种采用不同预条件和步长规则的 SAGA 算法。

5.1 数据集

为展示我们针对PET的新型随机优化框架，我们使用美国国家电气制造商协会图像质量（NEMA IQ）体模 [86], ，这是一种用于测试PET扫描仪和重建性能的标准体模，见图 8。该体模由一个有机玻璃容器和若干不同尺寸的球体嵌件组成，其中一些球体内装有放射性浓度高于背景的液体，另一些则装有冷水（非放射性水）。这使得我们能够评估分辨率和定量性能。我们使用了在伦敦大学学院医院（UCLH）核医学研究所通过西门子 Biograph mMR PET/MR扫描仪采集的NEMA IQ体模12分钟PET数据集 [87] 。本节中使用的数据、优化问题以及下文介绍的数据分区器，均为



(a) 水平切片中的感兴趣区域，放大区域见图4。



(b) 图4所示垂直切片剖面的折线图

图7. 比较不同数据遍历次数下各种算法重建体积的图表 .

PET快速图像重建挑战赛（PETRIC） [88],所开发。该挑战赛于2024年夏季举办，旨在推动针对PET重建的快速图像重建算法研究。

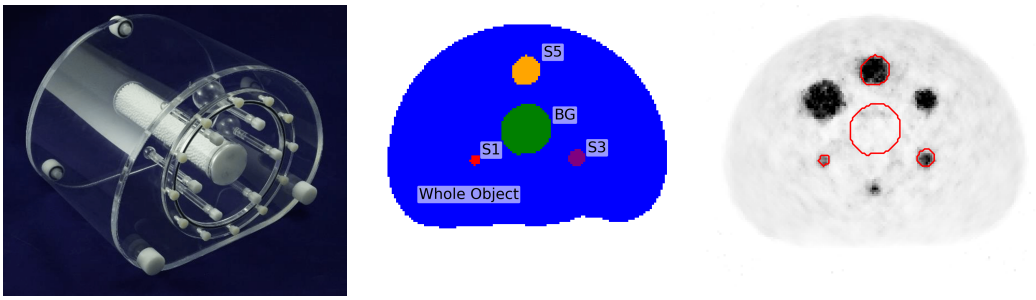


图 8。左：PET 对比中使用的 NEMA 体模照片（经 Data Spectrum 许可转载）。中：展示 PET 快速图像重建挑战赛 [88] 评估所用感兴趣体积（VOI）的切片：S1、S3、S5、BG 和 Whole Object 分别表示内部直径为 10 mm 的球体、内部直径为 17 mm 的球体、内部直径为 28 mm 的球体、背景和整个感兴趣对象。在这些区域中，我们使用了整个对象和 S1。右：PETRIC 组织方提供的参考重建切片，其中突出显示了 S1、S3、S4 和 BG。

5.2 方法

5.2.1 目标函数。 在本 PET 案例研究中，我们考虑一个最大后验（MAP）估计问题，其中包含平滑的相对差异惩罚项（RDP） [89] 和非负约束。目标是通过求解以下优化问题 [45]来恢复非负图像 $\boldsymbol{x}$：

$$\arg\min_{\boldsymbol{x}} \sum \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\log(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{\eta}) + \alpha \mathrm{RDP}(\boldsymbol{x}) + \mathbb{I}_{\{\boldsymbol{x}>0\}}(\boldsymbol{x}) \tag{15}$$

其中， $\boldsymbol{A}$ 是对系统进行建模的算子， $\boldsymbol{b}$ 是采集数据，而 $\boldsymbol{\eta}$ 是背景数据，例如随机事件和散射事件。

(15) 中的第一项是 Kullback-Leibler 数据保真项，它源自 MAP 公式中所用泊松分布的负对数似然。RDP 定义为

$$\mathrm{RDP}(\boldsymbol{x}) := \frac{1}{2}\sum_{i=1}^{N_v}\sum_{\ell\in N_i} w_{ij}\kappa_i\kappa_j \frac{(\boldsymbol{x}_i - \boldsymbol{x}_\ell)^2}{(\boldsymbol{x}_i + \boldsymbol{x}_\ell) + \omega\|\boldsymbol{x}_i - \boldsymbol{x}_\ell\| + \epsilon} \tag{16}$$

其中，N_i 是第 i 个图像体素周围的 3×3×3 邻域， $\omega = 2$ 是用于控制边缘保持的参数，w_{ij} 是权重因子（此处取“水平”体素尺寸除以 i号与 j号体素之间欧几里得距离）， ϵ 是用于使惩罚项可微的较小常数。参数 $\boldsymbol{\kappa}$ 是用于提供体素相关权重的图像。在我们的设置中， $\boldsymbol{\kappa}$ 通过标准有序子集期望最大化 (OSEM) 算法在初始重建处计算对数似然后得到的海森矩阵负行和的平方根预先计算得出（参见 [90]中的公式 25）。

5.2.2 数据分区。 对于随机优化算法，我们将式（15）表示为一个有限和优化问题，这与计算机断层扫描案例研究中的做法类似。我们首先对 PET 数据进行分区，采用前文所述的类似流程。给定子集数量 N，我们分别将采集数据 $\boldsymbol{b}$ 和含噪背景数据 $\boldsymbol{\eta}$ 分区为 $\boldsymbol{b}_i$ 和 $\boldsymbol{\eta}_i$，其中 $i = 0,\ 1,\ \cdots\ N-1$。对于每个 i，我们定义相应的算子 $\boldsymbol{A}_i$。利用上述

分区后的数据 $\boldsymbol{b}_i$、 $\boldsymbol{\eta}_i$ 和算子 $\boldsymbol{A}_i$，我们定义如下随机优化目标：

$$\arg\min_{\boldsymbol{x}} \sum_{i=0}^{N-1} f_i(\boldsymbol{x}) + g(\boldsymbol{x}) \tag{17}$$

$$f_i(\boldsymbol{x}) := \sum_i \boldsymbol{A}_i(\boldsymbol{x}) - \boldsymbol{b}_i \log(\boldsymbol{A}_i\boldsymbol{x} + \boldsymbol{\eta}_i) + \frac{\beta}{N}\mathrm{RDP}(\boldsymbol{x}) \tag{18}$$

$$g(\boldsymbol{x}) := \mathbb{I}_{\{\boldsymbol{x}>0\}}(\boldsymbol{x}) \tag{19}$$

PET 数据分区器（可在 SIRF-Contribs 存储库中找到） [91], 遵循与上述 CIL 数据分区器类似的接口。它的参数包括实验 PET 数据、背景校正（即散射和随机符合事件）、探测单元效率（包括衰减和探测效率），以及分区模式，可以是“顺序”、“交错”或“随机排列”。

与 CIL 不同，数据分区器不仅会对数据进行分区，还会为每个子集返回一个 SIRF 目标函数列表，我们仍将其称为 list_fi。这是因为 PET 中 A_i 算子的构造取决于子集所对应的探测单元效率和背景。这些 SIRF 目标函数的行为类似于 CIL 函数，可以传递给 CIL 算法。请注意，返回的 SIRF 目标函数对象用于最大化，因此若要将其用于执行最小化的 CIL 优化算法，需要将其乘以 -1。

与计算机断层扫描的情况一样，我们采用交错分区方案。我们将数据划分为 21、42 和 63 个子集，每个子集数量都能整除数据集中投影角度的总数，即 252。

5.2.3 实验设置与参数。 在本节中，我们展示不同的步长和预条件方案如何影响同一个 SAGA 算法的性能。该算法通过将 SAGAFunction 随机估计器接入 PGD 算法获得。

步长策略。 除了随机框架之外，我们还新增了支持一阶优化问题采用非固定步长的功能。具体而言，我们实现了 Armijo [92] 和 Barzilai-Borwein [93]等方法。此外，用户也可以灵活地定义自己的步长策略。用户可以继承基础抽象类 StepSizeRule，并重写 get_step_size 抽象方法。该方法会在算法类 GD、PGD 和 APGD 的更新步骤中调用，并可访问当前运行的算法实例的所有属性。这样，用户便可以自定义并实现各种步长规则。

第一个算法称为 SAGA-1，使用按 $\gamma_k = \gamma_0/(1 + \beta k)$ 定义的递减步长规则，其中 γ_0 表示初始步长， β 表示衰减参数， k 表示当前迭代次数；此处 $\beta = 0.02$ 且 $\gamma_0 = 1$。在 CIL 中，可以通过定义一个 DecreasingStepSize 类来实现，该类继承自 StepSizeRule 基抽象类，如下所示。

使用核心成像库求解逆问题的随机优化模块化方法

递减步长

```
类 DecreasingStepSize(StepSizeRule):

    定义 __init__(自身, gamma0,      贝塔=0.02):
        自身。gamma0 = gamma0
        自身。beta = beta


    定义 获取_步_长(自身, 算法):
        步_长 = 自身。gamma0/
             (1.0 + 自身。beta* 算法。迭代)
        返回 步_长

步_长_规则 =DecreasingStepSize(gamma0=初始_步_长)
```

预条件器。 我们通过预条件器类引入了一项新的预条件功能，可用于在每次迭代中调整梯度。该功能可以与算法 GD、PGD 和 APGD 结合使用。在许多情况下，预条件步骤会将梯度乘以一个预条件数组 P_k，例如算法2中的步骤4。

对于 SAGA-1，我们选择一个预条件数组 P_k，其取值为泊松对数似然 [94]的海森矩阵各行和的负值的倒数：

$$P_k = \left(\boldsymbol{A}_i^\top \left\{ \frac{\boldsymbol{b}_i}{(\boldsymbol{A}_i \boldsymbol{x}_k + \boldsymbol{\eta}_i)^2} \right\} \boldsymbol{A}_i \mathbf{1} \right)^{-1} = (\kappa^2 + \varepsilon)^{-1}. \tag{20}$$

下面的代码块演示了如何通过继承 Preconditioner class 并重写 apply 方法来定义上述预条件器。这样，用户便可针对特定的优化问题实现自定义的预条件策略。

SAGA-1：P_k

```
类 TsaiPreconditioner(预条件器):

    定义 __初始化__(自身, kappa,   epsilon=1e-6):
        自身。kappasq  =kappa*kappa  +epsilon


    定义 应用(自身, 算法, 梯度,                输出=无):
        返回 梯度。除以(自身。kappasq,          输出=输出)

预条件器 =TsaiPreconditioner(kappa)
```

采用递减步长和预条件的 SAGA-1

```
采样器 =采样器。随机_有_放回抽样(长度(列表_fi), 随机种子=40)
F_saga = -SAGAFunction(列表_fi, 采样器=采样器)
G = IndicatorBox(下限=0)
saga_1 = ISTA(初始 =OSEM_重建,      f=F_SAGA, g=G,
              更新_目标函数_间隔 =N,
              步长_大小 =步长_大小_规则 ,
              预条件器 = 预条件器)
SAGA_1。运行(迭代=K,      回调=[指标_回调])
```

请注意，在上述代码片段中，SAGAFunction 前面的负号是为了适应最小化问题而添加的，参见 5.2.2。

第二种算法 SAGA-SOS 是随机优化提交（SOS）团队提交至 PETRIC 挑战赛的 SAGA 方案 [88] 。该算法采用衰减步长规则和经过修改的 BSREM 型预条件器 [95],，定义如下：

$$P_k = \frac{\boldsymbol{x}^k}{\boldsymbol{A}^\top \mathbf{1}} \tag{21}$$

其中， $\boldsymbol{A}^\top \mathbf{1}$ 表示灵敏度图像，该图像通过对均匀数据进行反投影得到。步长通过 Armijo 规则确定：从用户定义的初始步长开始，每次迭代将其缩小为原来的一半，直至满足条件

$$f(\boldsymbol{x}_k - \gamma_k d_k) \leq f(\boldsymbol{x}_k) + \sigma\gamma_k \nabla f(\boldsymbol{x}_k)^\top d_k \tag{22}$$

其中， σ是用户定义的容差（SAGA-SOS 中设为 0.2）， $d_k = P_k \nabla f(\boldsymbol{x}_k)$表示预条件搜索方向 [92]。SAGA-SOS 算法首先通过 5 次近端梯度下降迭代估算合适的步长。请注意，在非负约束下，这相当于采用 Armijo 线搜索的投影梯度下降。最小步长定义为 $\gamma_0 = \min\{\alpha_1, \alpha_2, \ldots, \alpha_5\}$，其中 α_l是第 l 次迭代计算出的步长。随后将该 γ_0用作随机重建阶段的初始步长。在此阶段，步长按照 $\gamma_k = \gamma_0/(1 + \beta k)$ 逐步更新，其中 k 为迭代索引， $\beta = 0.01$ 为用户定义的衰减参数。

5.3 结果

我们比较了图 8 所示 NEMA 模体低计数采集数据的 SAGA-1 和 SAGA-SOS 重建结果，采用了 3 种子集数量逐渐增加的配置，分别为 21、42 和 63。我们还分别使用单个子集 $N = 1$ 运行了 SAGA-SOS 和 SAGA-1，从而得到各自算法的确定性版本。图 9 展示了在 30 次数据遍历后，SAGA-SOS 和 SAGA-1 算法所得重建结果相对于图 8 所示整个体积参考解的相对差异。图 10 展示了所选指标随数据遍历次数的变化情况，这些指标用于评估 PETRIC 挑战赛，包括整个对象的均方根误差（RMSE）和最小球体的均值绝对误差（AEM）。这些指标用于比较算法得到的解与 PETRIC 挑战赛组织方提供的参考解（该参考解通过求解相同的优化问题获得）。这些指标和参考算法的确切定义可在 PETRIC 网站上找到， [88], 不过在这两种情况下，数值越低，表示解越接近参考解。从所有这些图中可以看出，SAGA-SOS 的收敛速度更快

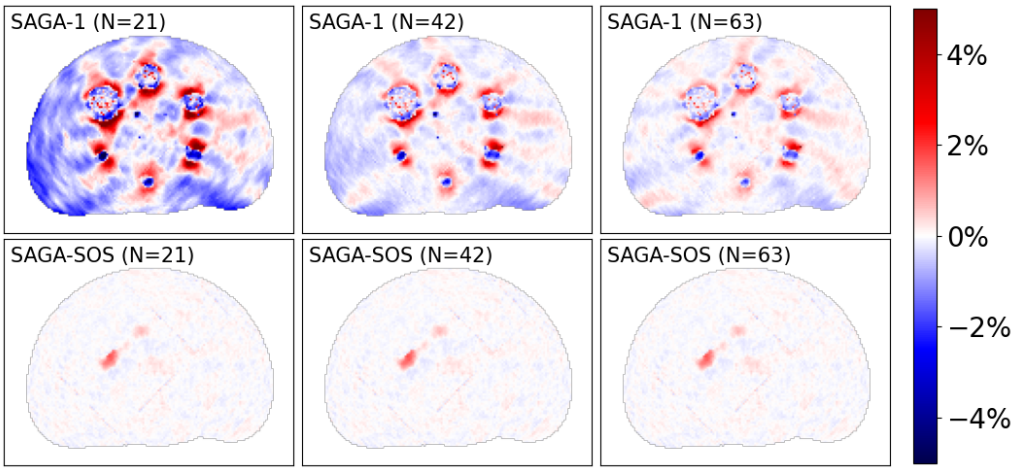


图 9. 根据图 8 中对整个体积的定义，SAGA-SOS 和 SAGA-1 算法在 30 次数据遍历时相对于参考解的重建结果相对差异。第一行对应 SAGA-1 算法，第二行对应 SAGA-SOS。各列对应子集数量：从左到右分别为 21、42、63。发散色图范围从 -5%（蓝色）到 +5%（红色），白色表示完全匹配。红色表示算法高估了该值，而蓝色表示低估了该值。可以看出，与 SAGA-1 相比，SAGA-SOS 算法更接近参考解地收敛，且不明显依赖于子集数量。

收敛至参考解的速度快于 SAGA-1。还可以观察到，SAGA-SOS 对子集数量的依赖性不强，而 SAGA-1 似乎会随着子集数量增加而略有改善。我们还可以看到，该算法的确定性版本始终不如其随机版本，见图 10。

5.3.1 结论。 该案例研究展示了 CIL 随机框架结合 SIRF 处理 PET 问题的灵活性；我们演示了如何使用 CIL 的随机框架定义自定义预条件器和步长规则；我们观察到，SAGA 算法的性能并非固有属性，而是深受全局配置的影响，例如步长和预条件的选择。在 PETRIC 挑战赛期间，我们也观察到了其他算法（如 SVRG）的类似结果，参见 [96]。

5.4 代码效率与加速

到目前为止，我们比较了不同算法在数据遍历次数方面的收敛速度，也就是计算复杂度。然而，其他参数也会影响这些算法的实际运行时间，例如随机存取存储器与中央处理器之间以及与图形处理器之间的内存访问和数据传输、底层代码的并行化等。下面我们将报告这些算法在墙钟时间方面的运行速度。

在图1中，我们绘制了调用 PET 算子的（墙钟）时间，包括正向计算和伴随运算、数据失配函数的梯度，以及 SAGA-1 算法每次数据遍历的墙钟时间，并将这些时间与问题中子集数量的倒数进行比较。PET 算子和数据失配函数的梯度都主要受数据计算影响；曲线显示，执行时间随子集数量倒数变化的预期线性关系。不过，SAGA-1 算法单次迭代的总运行时间还包括对梯度图像（数据量为图像数据的 3 倍）执行 3 次AX +BY 类型的代数运算、一次梯度内存复制，以及先验近端算子的计算。在本例中可以看出，这些额外任务占据了运行时间的主导地位

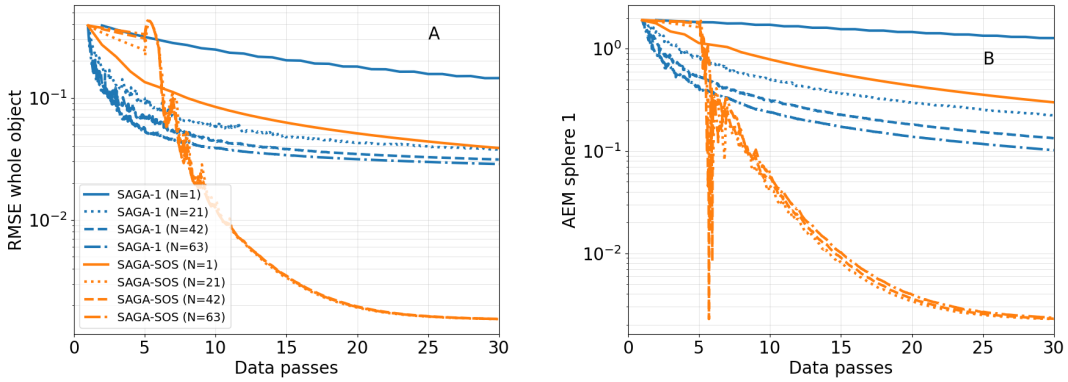


图 10. PETRIC 挑战赛中使用的指标随数据遍历次数的变化：SAGA-1 和 SAGA-SOS 提交方案在包含 1、21、42 和 63 个子集时的整体对象均方根误差（A）和球体 1 的均值绝对误差（B）。一个子集对应确定性算法，此处将其作为参考，以展示随机方法的潜力。这些指标用于比较算法得到的重建结果与参考图像；该参考图像由组织方在同一优化问题上运行另一种算法直至“收敛”后获得。指标越低，解就越接近参考解。可以看出，在这两项指标上，SAGA-SOS 均比 SAGA-1 更快收敛至参考解。

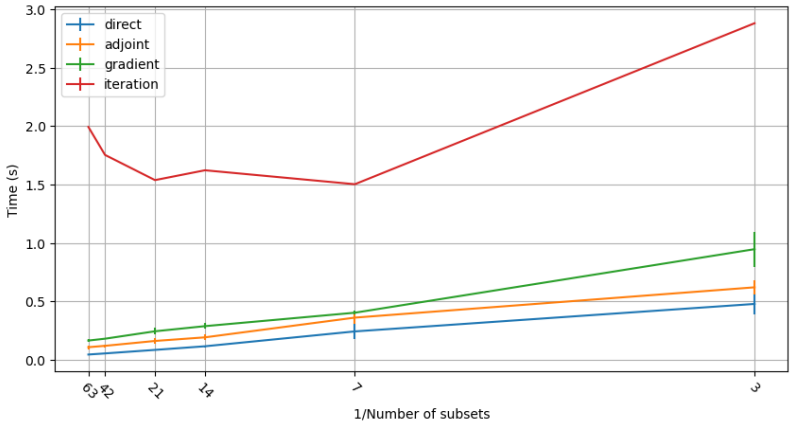


图 11. 计算 SAGA 算法（无先验）的梯度、正向计算、伴随计算所需时间，以及每次迭代所需时间。

（SIRF 3.8 中）的运行时间；由于先验仅取决于重建体积，增加子集数量并不会降低计算成本。我们观察到，最小运行时间出现在 7 到 21 个子集之间，子集数量过多或过少都会导致运行时间增加。具体而言，使用 21 个子集时，每次迭代的墙钟时间比使用 63 个子集时快 25%。

6 讨论

本文展示了 CIL 随机框架：进行分区和采样，并定义随机梯度估计器，再将其应用于确定性算法。我们还说明了这些算法

可以通过选择步长、预条件器和动量方法进一步定制。在计算机断层扫描案例中，与顺序采样相比，交错分区（即每个子集都包含来自各个投影的信息）效果更好。此外，以数据遍历次数衡量时，增加子集数量提高了收敛速度。与确定性对应算法相比，随机算法始终能以更少的数据遍历次数更快地收敛到“最优解”。在正电子发射断层成像案例中，我们强调了选择适当预条件器和步长策略的影响，并展示了这些选择如何显著提升算法性能。该案例还展示了 CIL 和随机框架与其他开源软件及应用集成的能力。

本文的主要目的是展示 CIL 中实现的随机框架，重点介绍其即插即用特性、灵活性和可扩展性。我们并未对算法进行全面比较，而是旨在展示如何将该框架应用于真实世界数据，让用户了解在自己的数据集上可能获得何种结果。因此，我们没有进行大规模参数扫描，也没有对算法性能进行精细调优。不过，该框架本身提供了必要的基础设施，可支持未来开展专门针对随机优化方法进行系统比较和基准测试的研究。

该框架的设计优先考虑易用性和快速原型开发，而非计算效率。虽然这使其能够灵活适配不同的应用和方法，但也意味着，针对特定任务定制并优化的算法，在速度方面可能胜过框架中实现的算法。目前的开发工作包括将 CIL 数据容器与数组 API（包括 PyTorch 张量）集成，这将加快小型数据上的原型开发。

尽管仅使用数据的一个子集可以降低每次迭代的计算负担，但从收敛时间角度获得的益处还需要考虑其他因素，例如子集间一致性、许多随机算法的步长限制，以及从随机存取存储器传输数据到中央处理器/图形处理器所需的数据传输时间。这些成本可能会削弱使用子集所带来的优势，因此需要针对具体情况进行评估。

另一个重要考虑因素是内存使用。随机梯度估计器，尤其是在大规模成像问题中，可能会占用大量内存，从而给真实数据应用带来挑战。未来的工作可以将内存效率更高的算法集成到 CIL 中，以帮助缓解这一问题。

7 结论

本研究提出了一个灵活的框架，通过将随机梯度估计器与确定性求解器相结合，并利用预条件、动量和自适应步长策略加以增强，从而实现各种随机算法。该方法在真实的 PET 和 CT 数据集上进行了演示，证明不仅适用于原型设计和算法开发，也适合部署于真实场景中。这个模块化框架简化了软件开发和代码维护，同时为用户自定义实验提供了灵活的接口。它使用户能够构建自己的随机估计器，并扩展 CIL 基础算法，以实现新的随机优化方法。鉴于这些算法正快速发展，这种适应性尤为重要。我们希望该框架能成为成像逆问题领域研究人员和从业者不可或缺的工具，并进一步推动随机方法在成像和逆问题领域的广泛应用，促进该领域的创新与实际影响。

数据可用性声明

本文所有结果的复现代码将在本稿件被接收后以开源形式发布。电池数据集——锂离子 10440 NMC532 电池——可通过电池成像库 [81] 获取，网址为 https://doi.org/10.5281/zenodo.17930456。PET 数据集可从 https://zenodo.org/records/1304454 获取。

致谢

EPap 感谢 Innovate UK 创新者分析（A4i）计划为项目 化学成像与断层成像数据去噪 提供资助，项目期间完成了该软件的初步设计与实现。CD 获得了正电子发射断层成像++、英国研究与创新署工程与自然科学研究理事会 EP/S026045/1 的支持。SP 部分获得了英国国家物理实验室通过科学、创新与技术部国家测量体系提供的支持（NPL 资助编号 PO496581），并获得了工程与自然科学研究理事会（EPSRC）通过工业 CASE 博士生奖学金（EPSRC 资助编号 EP/W522077/1）以及 Mediso 医学成像系统的支持。作者谨此感谢断层成像协同计算项目（通过 EPSRC 资助编号 EP/T026677/1 以及 STFC 通过英国研究与创新署数字研究基础设施计划提供支持）和生物医学成像协同重建协同计算项目 CCP SyneRBI（通过 EPSRC 资助编号 EP/T026693/1 以及 UKRI STFC 通过英国研究与创新署数字研究基础设施计划提供支持）的支持。

本研究得到了 CoSeC（研究社群计算科学中心）通过 CCP-SyneRBI 和 CCPi 提供的计算支持。作者谨此感谢 STFC 云平台为本研究提供计算平台。

作者还要感谢 Laura Murgatroyd、Matthias Ehrhardt、Tang Junqi、Imraj Singh、Robert Twyman、Antony Vamvakeros、Casper da Costa-Luis、Daniel Deidda、Ashley Gillman、Evgueni Ovtchinnikov、Georg Schramm、Hannah Robarts 和 Zeljko Kereta，感谢他们提出宝贵的意见并参与讨论，为本研究的完善提供了帮助。

参考文献

[1] M. Nikolova，“一种去除离群值和脉冲噪声的变分方法”，《数学成像与视觉杂志》，第20卷，第1期，第99–120页，2004年1月。 [在线]。网址：http://dx.doi.org/10.1023/B:JMIV.0000011326.88682.e5[2] T. Le、R. Chartrand 和 T. J. Asaki，“一种重建受泊松噪声污染图像的变分方法”，《数学成像与视觉杂志》，第27卷，第3期，第257âĂŞ263页，2007年3月。 [在线]。网址：http://dx.doi.org/10.1007/s10851-007-0652-y[3] A. N. Tikhonov 和 V. Y. Arsenin，《不适定问题的解》。华盛顿特区：John Wiley & Sons；纽约：V. H. Winston & Sons，1977年，俄文译本，译者序由 Fritz John 撰写，数学丛书 Scripta Series。[4] L. Rudin、S. Osher 和 E. Fatemi，“基于非线性全变分的噪声去除算法”，《Physica D：非线性现象》，第60卷，第259–268页，1992年。 [在线]。网址：http://dx.doi.org/10.1016/0167-2789(92)90242-F[5] K. Bredies、K. Kunisch 和 T. Pock，“全广义变分”，SIAM《成像科学杂志》，第3卷，第3期，第492–526页，2010年1月。 [在线]。网址：https://doi.org/10.1137/090769521[6] K. M. Holt，“向量场的全核变分和全变分的雅可比扩展”，IEEE《图像处理汇刊》，第23卷，第9期，第3975–3989页，2014年。 [在线]。网址：http://dx.doi.org/10.1109/TIP.2014.2332397[7] S. Lefkimmiatis、A. Roussos、M. Unser 和 P. Maragos，“基于结构张量的全变分凸推广及其在逆问题中的应用”，载于《计算机视觉中的尺度空间与变分方法》，2013年，第48–60页。 [在线]。网址：http://dx.doi.org/10.1007/978-3-642-38267-3_5[8] A. Chambolle 和 T. Pock，“成像连续优化导论”，《Acta Numer.》，第25卷，第161–319页，2016年。[9] M. Benning 和 M. Burger，“逆问题的现代正则化方法”，《Acta Numer.》，第27卷，第1âĂŞ111页，2018年。[10] K. Bredies 和 M. Holler，“高阶全变分方法及其推广”，《逆问题》，第36卷，第12期，第123001页，2020年12月。[在线]。网址：http://dx.doi.org/10.1088/1361-6420/ab8f80[11] E. K. Ryu 和 W. Yin，《大规模凸优化：基于单调算子的算法与分析》。剑桥大学出版社，2022年11月。[在线]。网址：http://dx.doi.org/10.1017/9781009160865[12] J. L. Mueller 和 S. Siltanen，《线性与非线性逆问题及其实用应用》。工业与应用数学学会，2012年10月。 [在线]。网址：http://dx.doi.org/10.1137/1.9781611972344

使用核心成像库求解逆问题的随机优化模块化方法

[13] J. S. Jørgensen、E. Ametova、G. Burca、G. Fardell、E. Papoutsellis、E. Pasca、K. Thielemans、M. Turner、R. Warr、W. R. B. Lionheart 和 P. J. Withers，“核心成像库——第一部分：用于断层成像的通用Python框架”，《英国皇家学会哲学汇刊 A 辑》20200192，2021年。 [在线]。网址：https://doi.org/10.1098/rsta.2020.0192[14] A. Beck 和 M. Teboulle，“用于约束全变分图像去噪和去模糊问题的快速基于梯度的算法”，IEEE《图像处理汇刊》，第18卷，第11期，第2419–2434页，2009年11月。 [在线]。网址：https://doi.org/10.1109/tip.2009.2028250[15] A. Chambolle 和 T. Pock，“凸问题的一阶原始-对偶算法及其在成像中的应用”，《数学成像与视觉杂志》，第40卷，第1期，第120–145页，2010年12月。 [在线]。网址：https://doi.org/10.1007/s10851-010-0251-1[16] D. OâĂŹConnor 和 L. Vandenberghe，“原始-对偶混合梯度法与 DouglasâĂŞRachford 分裂法的等价性”，《数学规划》，第179卷，第1âĂŞ2期，第85âĂŞ108页，2018年8月。 [在线]。网址：http://dx.doi.org/10.1007/s10107-018-1321-1[17] E. Papoutsellis、E. Ametova、C. Delplancke、G. Fardell、J. S. Jørgensen 等，“核心成像库——第二部分：动态与能谱断层成像的多通道重建”，《英国皇家学会哲学汇刊 A 辑》，第379卷，第2204期，第20200193页，2021年7月。 [在线]。网址：https://doi.org/10.1098/rsta.2020.0193[18] J. S. JÃ¿rgensen、E. Papoutsellis、L. Murgatroyd、G. Fardell 和 E. Pasca，“使用核心成像库（CIL）解决赫尔辛基有限角度断层成像挑战赛的方向正则化方法”，《现代挑战应用数学》，第1卷，第2期，第143âĂŞ169页，2023年。 [在线]。网址：http://dx.doi.org/10.3934/ammc.2023011[19] R. Warr、E. Ametova、R. J. Cernik、G. Fardell、S. Handschuh、J. S. JÃ¿rgensen、E. Papoutsellis、E. Pasca 和 P. J. Withers，“通过时空光谱重建增强用于生物成像的高光谱断层成像”，《科学报告》，第11卷，第1期，2021年10月。 [在线]。网址：http://dx.doi.org/10.1038/s41598-021-00146-4[20] E. Ametova、G. Burca、S. Chilingaryan、G. Fardell、J. S. Jørgensen、E. Papoutsellis、E. Pasca、R. Warr、M. Turner、W. R. B. Lionheart 和 P. J. Withers，“采用先进重建方法进行晶相鉴别的中子断层成像”，《物理学杂志D：应用物理》，2021年。 [在线]。网址：https://doi.org/10.1088/1361-6463/ac02f9[21] B. Simmendefeldt Schmidt、J. Sauer JÃ¿rgensen、J. Rueda-Rueda、J. Galdon-QuÃŋroga、M. GarcÃŋa-MuÃśoz 和 M. Salewski，“用于快离子损失探测器测量反演的各向异性正则化”，《核聚变》，第64卷，第10期，第106053页，2024年9月。 [在线]。网址：http://dx.doi.org/10.1088/1741-4326/ad75a5[22] E. Ovtchinnikov、R. Brown、C. Kolbitsch、E. Pasca、C. da Costa-Luis、A. G. Gillman、B. A. Thomas、N. Efthimiou、J. Mayer、P. Wadhwa、M. J. Ehrhardt、S. Ellis、J. S. Jørgensen、J. Matthews、C. Prieto、A. J. Reader、C. Tsoumpas、M. J. Turner、D. Atkinson 和 K. Thielemans，“SIRF：协同图像重建框架”，《计算物理通讯》，第249卷，第107087页，2020年4月。 [在线]。网址：https://doi.org/10.1016/j.cpc.2019.107087[23] R. Brown、C. Kolbitsch、C. Delplancke、E. Papoutsellis、J. Mayer、E. Ovtchinnikov、E. Pasca、R. Neji、C. da Costa-Luis、A. G. Gillman、M. J. Ehrhardt、J. McClelland、B. Eiben 和 K. Thielemans，“使用 SIRF 和 CIL 进行同步 PET/MR 的运动估计与校正”，《英国皇家学会哲学汇刊 A 辑》20200208，2021年。 [在线]。网址：https://doi.org/10.1098/rsta.2020.0208[24] L. Protopapa、M. A. G. Duff、J. Mayer、J. Schulz-Menger、K. Thielemans、C. Kolbitsch 和 E. Pasca，“通过随机优化实现高效运动校正三维心脏磁共振成像重建”，《医学与生物学中的物理学》，2025年。 [在线]。网址：http://iopscience.iop.org/article/10.1088/1361-6560/adf609[25] F. M. Watson、D. Andre 和 W. R. B. Lionheart，“解决多基地合成孔径雷达中的全波穿墙传输效应”，《逆问题》，第40卷，第8期，第085009页，2024年7月。 [在线]。网址：http://dx.doi.org/10.1088/1361-6420/ad5b83[26] M. J. Ehrhardt、Z. Kereta、J. Liang 和 J. Tang，“大规模逆问题随机优化指南”，《逆问题》，第41卷，第5期，第053001页，2025年5月。 [在线]。网址：https://dx.doi.org/10.1088/1361-6420/adc0b7[27] M. Ravasi、M. V. Örnhag、N. Luiken、O. Leblanc 和 E. UruÃśuela，“Pyproximal——Python中的可扩展凸优化”，《开放源代码软件杂志》，第9卷，第95期，第6326页，2024年3月。 [在线]。网址：http://dx.doi.org/10.21105/joss.06326[28] M. Ravasi 和 I. Vasconcelos，“Pylops——用于可扩展代数与优化的线性算子Python库”，SoftwareX，第11卷，第100361页，2020年1月。 [在线]。网址：http://dx.doi.org/10.1016/j.softx.2019.100361[29] T. Balke、F. Davis、C. Garcia-Cardona、S. Majee、M. McCann、L. Pfister 和 B. Wohlberg，“科学计算成像代码（SCICO）”，《开放源代码软件杂志》，第7卷，第78期，第4722页，2022年10月。 [在线]。网址：http://dx.doi.org/10.21105/joss.04722[30] J. Tachella、M. Terris、S. Hurault、A. Wang、D. Chen、M.-H. Nguyen、M. Song、T. Davies、L. Davy、J. Dong、P. Escande、J. Hertrich、Z. Hu、T. I. Liaudat、N. Laurent、B. Levac、M. Massias、T. Moreau、T. Modrzyk、B. Monroy、S. Neumayer、J. Scanvic、F. Sarron、V. Sechaud、G. Schramm、R. Vo 和 P. Weiss，“Deepinverse：用于通过深度学习求解成像逆问题的Python软件包”，2025年。 [在线]。网址：https://arxiv.org/abs/2505.20160[31] M. Simeoni、S. Kashani、J. RuÃĺ-Queralt 和 P. Developers，“pyxu-org/pyxu：pyxu。” [在线]。网址：https://doi.org/10.5281/zenodo.4486431[32] J. Adler、H. Kohr、A. Ringh、J. Moosmann、Sbanert、M. J. Ehrhardt、G. R. Lee、Niinimaki、Bgris、O. Verdier、J. Karlsson、Zickert、W. J. Palenstijn、O. Öktem、C. Chen、H. A. Loarca 和 M. Lohmann，“odlgroup/odl：ODL 0.7.0”，2018年。 [在线]。网址：https://zenodo.org/record/1442734[33] I. Daubechies、M. Defrise 和 C. De Mol，“一种用于带稀疏性约束线性逆问题的迭代阈值算法”，《纯粹与应用数学通讯》，第57卷，第11期，第1413âĂŞ1457页，2004年8月。 [在线]。网址：http://dx.doi.org/10.1002/cpa.20042[34] J. Nocedal 和 S. J. Wright，《数值优化》。施普林格，2006年。[35] J. Barzilai 和 J. M. Borwein，“两点步长梯度法”，IMA数值分析杂志，第8卷，第1期，第141–148页，1988年1月。 [在线]。网址：https://doi.org/10.1093/imanum/8.1.141

[36] A. Beck 和 M. Teboulle，“一种用于线性逆问题的快速迭代收缩阈值算法”，SIAM《成像科学杂志》，第 2 卷，第 1 期，第 183–202 页，2009年。 [在线]。可访问：https://doi.org/10.1137/080716542[37] Y. E. Nesterov，“一种求解凸规划问题的方法，其收敛速度为 $o(\frac{1}{k^2})$”，苏联科学院院刊，第 269 卷，第 3 期，第 543–547 页，1983年。[38] Y. Nesterov，“非光滑函数的光滑最小化”，《数学规划》，第 103 卷，第 1 期，第 127âĂŞ152 页，2004年12月。 [在线]。可访问：http://dx.doi.org/10.1007/s10107-004-0552-5[39] ——，《凸优化导论》。施普林格美国公司，2004年。 [在线]。可访问：http://dx.doi.org/10.1007/978-1-4419-8853-9[40] ——，“用于最小化复合函数的梯度方法”，《数学规划》，第 140 卷，第 1 期，第 125âĂŞ161 页，2012年12月。 [在线]。可访问：http://dx.doi.org/10.1007/s10107-012-0629-5[41] A. B. Taylor、J. M. Hendrickx 和 F. Glineur，“复合凸优化一阶方法的精确最坏情形性能”，SIAM 优化杂志，第 27 卷，第 3 期，第 1283âĂŞ1313 页，2017年1月。 [在线]。可访问：http://dx.doi.org/10.1137/16M108104X[42] A. d’Aspremont、D. Scieur 和 A. Taylor，“加速方法”，《优化基础与趋势》：第 5 卷：第 1–2 期，第 1–245 页（2021），2021年。[在线]。可访问：https://doi.org/10.1561/2400000036[43] L. Kaufman，“PET 的最大似然、最小二乘和惩罚最小二乘”，IEEE 医学成像汇刊，第 12 卷，第 2 期，第 200âĂŞ214 页，1993年6月。 [在线]。可访问：http://dx.doi.org/10.1109/42.232249[44] N. Clinthorne、T.-S. Pan、P.-C. Chiao、W. Rogers 和 J. Stamos，“用于提高迭代重建收敛速率的预条件方法”，IEEE 医学成像汇刊，第 12 卷，第 1 期，第 78âĂŞ83 页，1993年3月。 [在线]。可访问：http://dx.doi.org/10.1109/42.222670[45] J. Qi 和 R. M. Leahy，“发射计算机断层成像中的迭代重建技术”，《医学与生物学物理》，第 51 卷，第 15 期，第 R541âĂŞR578 页，2006年7月。 [在线]。可访问：http://dx.doi.org/10.1088/0031-9155/51/15/R01[46] L. Condat，“涉及 Lipschitz 连续、可近端计算及线性复合项的凸优化原始-对偶分裂方法”，《优化理论与应用杂志》，第 158 卷，第 2 期，第 460–479 页，2013年。 [在线]。可访问：https://doi.org/10.1007/s10957-012-0245-9[47] B. C. V˜u，“一种求解涉及余强制算子的对偶单调包含问题的分裂算法”，《计算数学进展》，第 38 卷，第 3 期，第 667âĂŞ681 页，2011年11月。 [在线]。可访问：http://dx.doi.org/10.1007/s10444-011-9254-8[48] P. Chen、J. Huang 和 X. Zhang，“一种用于最小化三个凸可分函数之和的原始-对偶不动点算法”，《不动点理论与应用》，第 2016 卷，第 1 期，2016年4月。 [在线]。可访问：http://dx.doi.org/10.1186/s13663-016-0543-2[49] M. Yan，“一种用于最小化带线性算子的三个函数之和的新型原始âĂŞ对偶算法”，《科学计算杂志》，第 76 卷，第 3 期，第 1698âĂŞ1717 页，2018年3月。 [在线]。可访问：http://dx.doi.org/10.1007/s10915-018-0680-3[50] P. Latafat 和 P. Patrinos，“用于求解涉及三个算子的单调包含问题的非对称前向âĂŞ后向âĂŞ伴随分裂方法”，《计算优化与应用》，第 68 卷，第 1 期，第 57âĂŞ93 页，2017年4月。 [在线]。可访问：http://dx.doi.org/10.1007/s10589-017-9909-6[51] A. Salim、L. Condat、K. Mishchenko 和 P. RichtÃąrik，“对偶化、分裂、随机化：迈向快速非光滑优化算法”，《优化理论与应用杂志》，第 195 卷，第 1 期，第 102âĂŞ130 页，2022年7月。 [在线]。可访问：http://dx.doi.org/10.1007/s10957-022-02061-8[52] E. Esser、X. Zhang 和 T. F. Chan，“成像科学中一类凸优化一阶原始-对偶算法的通用框架”，SIAM《成像科学杂志》，第 3 卷，第 4 期，第 1015âĂŞ1046 页，2010年1月。 [在线]。可访问：http://dx.doi.org/10.1137/09076934X[53] J. M. Fadili 和 G. PeyrÃĺ，“采用一阶方法的全变分投影”，IEEE《图像处理汇刊》，第 20 卷，第 3 期，第 657–669 页，2011 年。 [在线]。可访问：https://doi.org/10.1109/TIP.2010.2072512[54] M. Yan 和 Y. Li，“ 论若干原始 - 对偶算法的改进条件”，《科学计算杂志》，第 99 卷，第 3 期，2024年5月。 [在线]。可访问：http://dx.doi.org/10.1007/s10915-024-02537-x[55] X. Jiang 和 L. Vandenberghe，“布雷格曼三算子分裂方法”，《优化理论与应用杂志》，第 196 卷，第 3 期，第 936âĂŞ972 页，2022年11月。 [在线]。可访问：http://dx.doi.org/10.1007/s10957-022-02125-9[56] L. Bottou、F. E. Curtis 和 J. Nocedal，“大规模机器学习的优化方法”，SIAM《评论》，第 60 卷，第 2 期，第 223âĂŞ311 页，2018年1月。[在线]。可访问：http://dx.doi.org/10.1137/16M1080173[57] M. Schmidt、N. Le Roux 和 F. Bach，“使用随机平均梯度最小化有限和”，《数学规划》，第 162 卷，第 1 期，第 83–112 页，2017年3月。 [在线]。可访问：https://doi.org/10.1007/s10107-016-1030-6[58] A. Defazio、F. Bach 和 S. Lacoste-Julien，“SAGA：一种支持非强凸复合目标函数的快速增量梯度方法”，载于《神经信息处理系统进展》，第 2 卷，2014年，第 1646–1654 页。[59] R. Johnson 和 T. Zhang，“利用预测性方差缩减加速随机梯度下降”，载于《神经信息处理系统进展》，C. Burges、L. Bottou、M. Welling、Z. Ghahramani 和 K. Weinberger 编，第 26 卷。Curran Associates, Inc.，2013年。 [在线]。可访问：https://proceedings.neurips.cc/paper_files/paper/2013/file/ac1dd209cbcc5e5d1c6e28598e8cbbe8-Paper.pdf[60] D. Kovalev、S. Horváth 和 P. Richtárik，“别再绕圈子、去掉那些循环：没有外循环的 SVRG 和 Katyusha 效果更好”，《第31届国际算法学习理论会议论文集》，第 117 卷，第 451–467 页，2020年2月8日至11日。 [在线]。可访问：https://proceedings.mlr.press/v117/kovalev20a.html[61] T. Hofmann、A. Lucchi、S. Lacoste-Julien 和 B. McWilliams，“结合邻居的方差缩减随机梯度下降”，载于《神经信息处理系统进展》，C. Cortes、N. Lawrence、D. Lee、M. Sugiyama 和 R. Garnett 编，第 28 卷。Curran Associates, Inc.，2015年。 [在线]。可访问：https://proceedings.neurips.cc/paper_files/paper/2015/file/effc299a1addb07e7089f9b269c31f2f-Paper.pdf[62] X. Qian、Z. Qu 和 P. Richtárik，“采用任意抽样的 L-SVRG 和 L-Katyusha”，《机器学习研究杂志》，第 22 卷，第 112 期，第 1–47 页，2021年。 [在线]。可访问：http://jmlr.org/papers/v22/20-156.html

使用核心成像库求解逆问题的随机优化模块化方法

[63] D. Blatt、A. O. Hero 和 H. Gauchman，“一种采用固定步长的收敛增量梯度方法”，SIAM 优化杂志，第 18 卷，第 1 期，第 29â ĂŞ51 页，2007年1月。 [在线]。可访问：http://dx.doi.org/10.1137/040615961[64] D. Driggs、J. Liang 和 C.-B. Schonlieb，“关于有偏随机梯度估计”，《机器学习研究杂志》，第 23 卷，第 24 期，第 1–43 页，2022年。 [在线]。可访问：http://jmlr.org/papers/v23/20-316.html[65] D. Karimi 和 R. K. Ward，“用于锥束计算机断层扫描图像重建的混合随机-确定性梯度下降算法”，《生物医学物理与工程快报》，第 2 卷，第 1 期，文章编号 015008，2016年2月。 [在线]。可访问：http://dx.doi.org/10.1088/2057- 1976/2/1/015008[66] L. Xiao 和 T. Zhang，“采用渐进方差缩减的近端随机梯度法”，SIAM 优化杂志，第 24 卷，第 4 期，第 2057âĂŞ2075 页，2014年1月。 [在线]。可访问：http://dx.doi.org/10.1137/140961791[67] B. Li、M. Ma 和 G. B. Giannakis，“论 SARAH 及其拓展的收敛性”，载于《机器学习研究论文集》，2019年。 [在线]。可访问：https://arxiv.org/abs/1906.02351[68] N. H. Pham、L. M. Nguyen、D. T. Phan 和 Q. Tran-Dinh，“Proxsarah：一种用于随机复合非凸优化的高效算法框架”，载于《机器学习研究杂志》，2019年。 [在线]。可访问：https://arxiv.org/abs/1902.05679[69] A. Chambolle、M. J. Ehrhardt、P. Richtárik 和 C.-B. Schönlieb，“采用任意抽样并用于成像应用的随机原始-对偶混合梯度算法”，SIAM 优化杂志，第 28 卷，第 4 期，第 2783–2808 页，2018年1月。 [在线]。可访问：https://doi.org/10.1137/17m1134834[70] H. Robbins 和 S. Monro，“一种随机逼近法”，数学统计年刊，第 22 卷，第 3 期，第 400âĂŞ407 页，1951年9月。[在线]。可访问：http://dx.doi.org/10.1214/aoms/1177729586[71] J. Duchi 和 Y. Singer，“使用前向-后向分裂进行高效在线学习和批量学习”，《机器学习研究杂志》，第 10 卷，第 99 期，第 2899–2934 页，2009年。 [在线]。可访问：http://jmlr.org/papers/v10/duchi09a.html[72] K. Zhou、Y. Jin、Q. Ding 和 J. Cheng，“摊销 Nesterov 动量：一种稳健的动量及其在深度学习中的应用”，载于《第36届人工智能不确定性会议（UAI）论文集》，《机器学习研究论文集》系列，J. Peters 和 D. Sontag 编，第 124 卷。PMLR，2020年8月3日至6日，第 211–220 页。 [在线]。可访问：https://proceedings.mlr.press/v124/zhou20a.html[73] D. Driggs、M. J. Ehrhardt 和 C.-B. Schönlieb，“加速方差缩减随机梯度法”，《数学规划》，第 191 卷，第 2 期，第 671âĂŞ715 页，2020年9月。 [在线]。可访问：http://dx.doi.org/10.1007/s10107-020-01566-2[74] A. Yurtsever、B. C. Vu 和 V. Cevher，“随机三复合凸最小化”，载于《神经信息处理系统进展》，D. Lee、M. Sugiyama、U. Luxburg、I. Guyon 和 R. Garnett 编，第 29 卷。Curran Associates, Inc.，2016年。 [在线]。可访问：https://proceedings.neurips.cc/paper_files/paper/2016/file/5d6646aad9bcc0be55b2c82f69750387-Paper.pdf[75] R. Zhao、W. B. Haskell 和 V. Y. F. Tan，“一种用于随机三复合优化的最优算法”，载于《第二十二届人工智能与统计国际会议论文集》，《机器学习研究论文集》系列，K. Chaudhuri 和 M. Sugiyama 编，第 89 卷。PMLR，2019年4月16日至18日，第 428–437 页。 [在线]。可访问：https://proceedings.mlr.press/v89/zhao19a.html[76] G. Herman 和 L. Meyer，“代数重建技术可实现计算高效（正电子发射断层成像应用）”，IEEE 医学成像汇刊，第 12 卷，第 3 期，第 600âĂŞ609 页，1993年。 [在线]。可访问：http://dx.doi.org/10.1109/42.241889[77] I. R. Singh、R. Barbano、R. Twyman、Å. Kereta、B. Jin、S. Arridge 和 K. Thielemans，“基于 SIRF 目标函数的深度图像先验 PET 重建”，载于《2022年 IEEE 核科学研讨会暨医学成像会议（NSS/MIC）》，2022年，第 1–2 页。 [在线]。可访问：https://doi.org/10.1109/NSS/MIC44845.2022.10399292[78] M. Blondel、Q. Berthet、M. Cuturi、R. Frostig、S. Hoyer、F. Llinares-López、F. Pedregosa 和 J.-P. Vert，“高效且模块化的隐式微分”，arXiv 预印本 arXiv:2105.15183，2021年。[79] A. Vamvakeros、D. Matras、T. E. Ashton、A. A. Coelho、H. Dong、D. Bauer、Y. Odarchenko、S. W. T. Price、K. T. Butler、O. Gutowski、A. Dippel、M. v. Zimmerman、J. A. Darr、S. D. M. Jacques 和 A. M. Beale，“循环速率诱导的商用圆柱形锂离子电池中空间分辨异质性”，Small Methods，第 5 卷，第 9 期，2021年8月。 [在线]。可访问：http://dx.doi.org/10.1002/smtd.202100512[80] M. Drakopoulos、T. Connolley、C. Reinhard、R. Atwood、O. Magdysyuk、N. Vo、M. Hart、L. Connor、B. Humphreys、G. Howell、S. Davies、T. Hill、G. Wilkin、U. Pedersen、A. Foster、N. De Maio、M. Basham、F. Yuan 和 K. Wanelik，“I12：Diamond Light Source 的联合工程、环境与加工（JEEP）光束线”，同步辐射期刊，第 22 卷，第 3 期，第 828âĂŞ838 页，2015年4月。 [在线]。可访问：http://dx.doi.org/10.1107/S1600577515003513[81] R. Docherty、S. Riley、J. D. Morley、E. Papoutsellis、A. D. Bobitan、J. Donoghue、K. Rankin、F. Alvarez Borges、L. Salinas-Farran、S. Michalik、A. Liptak、G. Burca、P.-O. Autran、J. Wright、W. Kockelmann、O. Gutowski、A.-C. Dippel、M. von Zimmermann、D. Matras、B. Winiarski、J. Mangum、D. Finegan、J. A. Gott、D. Proprentner、H. Dong、C. George、A. M. Beale、S. D. Jacques、S. J. Cooper 和 A. Vamvakeros，“电池成像库：面向所有人的多长度尺度、多模态同步辐射与实验室电池成像数据”，2025年9月。 [在线]。可访问：https://chemrxiv.org/doi/full/10.26434/chemrxiv-2025-sbp73/v2[82] N. T. Vo、R. C. Atwood、M. Drakopoulos 和 T. Connolley，“平行束断层扫描中视野范围外样品的数据处理方法与数据采集”，Opt. Express，第 29 卷，第 12 期，第 17849–17 874 页，2021年6月。 [在线]。可访问：https://opg.optica.org/oe/abstract.cfm?URI=oe-29-12-17849[83] A. Chambolle，“一种全变分最小化算法及其应用”，J. Math. Imaging Vis.，第 20 卷，第 1 期，第 89–97 页，2004年。[84] D. Kazantsev、E. Pasca、M. J. Turner 和 P. J. Withers，“用于采用近端分裂算法进行计算机断层图像重建的 CCPi 正则化工具包”，SoftwareX，第 9 卷，第 317–323 页，2019年1月。 [在线]。可访问：https://doi.org/10.1016/j.softx.2019.04.003

[85] M. Yan, “一种用于最小化包含线性算子的三个函数之和的全新原始–对偶算法”，《科学计算杂志》，第 76 卷，页码 1698–1717，2018年。[86] N. S. P. N. 美国国家电气制造商协会，“正电子发射断层扫描仪的性能测量”，弗吉尼亚州罗斯林，2007年。[87] T. BA 和 S. T.，“西门子MMR扫描仪上的NEMA图像质量模体采集”，2018年。 [在线]。链接：https://doi.org/10.5281/zenodo. 1304454[88] C. da Costa-Luis、M. J. Ehrhardt、C. Kolbitsch、E. Ovtchinnikov、E. Pasca、K. Thielemans 和 C. Tsoumpas，“PET快速图像重建挑战赛（PETRIC）”，2025年。 [在线]。链接：https://arxiv.org/abs/2511.22566[89] N. J.、B. D.、D. P. 和 M. L.，“一种用于发射断层成像最大后验重建的相对差异惩罚凹先验”，*IEEE*核科学汇刊，第 49 卷，第 1 期，页码 56–60，2002年。 [在线]。链接：https://doi.org/10.1109/TNS.2002.998681[90] Y.-J. Tsai、G. Schramm、S. Ahn、A. Bousse、S. Arridge、J. Nuyts、B. F. Hutton、C. W. Stearns 和 K. Thielemans，“在 PET 重建中使用空间变化惩罚强度与解剖先验的益处”，*IEEE* 医学成像汇刊，第 39 卷，第 1 期，页码 11–22，2020年。 [在线]。链接：https://doi.org/10.1109/TMI.2019.2913889[91] https://github.com/SyneRBI/SIRF-Contribs/blob/master/src/Python/sirf/contrib/partitioner/partitioner.py.[92] L. Armijo，“具有利普希茨连续的一阶偏导数的函数的最小化”，太平洋数学杂志，第 16 卷，第 1 期，页码 1–3，1966年。 [在线]。链接：https://msp.org/pjm/1966/16-1/p01.xhtml[93] J. Barzilai 和 J. M. Borwein，“两点步长梯度法”，*IMA*数值分析杂志，第 8 卷，第 1 期，第 141âĂŞ148 页，1988年。 [在线]。链接：http://dx.doi.org/10.1093/imanum/8.1.141[94] Y.-J. Tsai、G. Schramm、S. Ahn、A. Bousse、S. Arridge、J. Nuyts、B. F. Hutton、C. W. Stearns 和 K. Thielemans，“在 PET 重建中使用空间变化惩罚强度与解剖先验的益处”，*IEEE* 医学成像汇刊，第 39 卷，第 1 期，页码 11–22，2020年。 [在线]。链接：https://doi.org/10.1109/TMI.2019.2913889[95] S. Ahn 和 J. A. Fessler，“使用松弛有序子集算法实现发射断层成像的全局收敛图像重建”，*IEEE* 医学成像汇刊，第 22 卷，第 5 期，页码 613–626，2003年。[96] M. J. Ehrhardt、Z. Kereta 和 G. Schramm，“通过方差缩减和先验感知预条件实现快速 PET 重建”，核医学前沿，第 5 卷，文章编号 1641215，2025年。 [在线]。链接：https://doi.org/10.3389/fnume.2025.1641215