

1 BM3DORNL: High-Performance BM3D Denoising for 2 Neutron Tomography

3 Chen Zhang  ^{1¶}, Jean-Christophe Bilheux  ², Dmitry Ganyushin¹, and Pete
4 Peterson  ¹

5 1 Computing and Computational Sciences Directorate, Oak Ridge National Laboratory, Oak Ridge, TN,
6 USA 2 Neutron Sciences Directorate, Oak Ridge National Laboratory, Oak Ridge, TN, USA ¶
7 Corresponding author

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Editor: [Open Journals](#) ↗

Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

License

Authors of papers retain copyright
and release the work under a
Creative Commons Attribution 4.0
International License ([CC BY 4.0](#))³

8 **Notice:** This manuscript has been authored by UT-Battelle, LLC, under contract DE-AC05-
9 00OR22725 with the US Department of Energy (DOE). The US government retains and the
10 publisher, by accepting the article for publication, acknowledges that the US government
11 retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the
12 published form of this manuscript, or allow others to do so, for US government purposes. DOE
13 will provide public access to these results of federally sponsored research in accordance with
14 the DOE Public Access Plan (<https://www.energy.gov/doe-public-access-plan>).

15 Summary

16 BM3DORNL is a high-performance Python library for denoising neutron and X-ray tomography
17 data using a modified Block-Matching and 3D Filtering (BM3D) algorithm. The library
18 provides two denoising modes: a generic mode for standard noise removal, and a specialized
19 streak mode optimized for removing vertical streak patterns in sinograms that manifest as ring
20 artifacts in reconstructed images. Built with a Rust backend and Python bindings via PyO3,
21 BM3DORNL achieves 160× faster processing than comparable BM3D implementations while
22 maintaining cross-platform compatibility including native Apple Silicon support.

23 Statement of Need

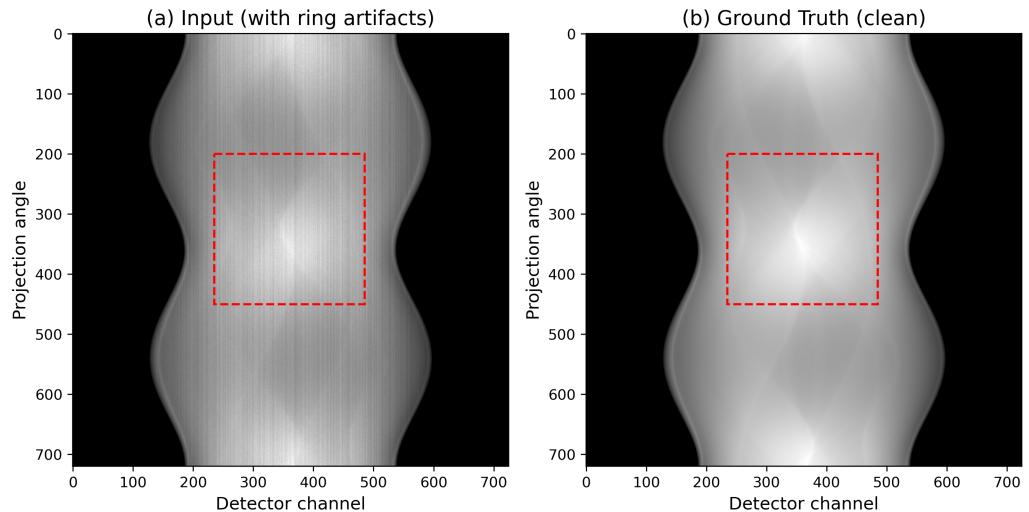
24 Ring artifacts are a persistent challenge in neutron and X-ray computed tomography, arising
25 from variations in detector pixel response, beam intensity fluctuations, and systematic errors
26 ([Münch et al., 2009](#)). These artifacts appear as concentric rings in reconstructed images,
27 obscuring sample structure and degrading quantitative analysis. Pre-reconstruction streak
28 removal addresses these artifacts directly in sinograms, avoiding the spatial spreading that
29 occurs during reconstruction. Existing approaches include wavelet-Fourier filtering ([Münch et
30 al., 2009](#)), polynomial fitting ([Vo et al., 2018](#)), and sorting-based methods ([Miqueles et al.,
31 2014](#)), with implementations available in TomoPy ([Gürsoy et al., 2014](#)).

32 Mäkinen et al. ([Mäkinen et al., 2021](#)) demonstrated that applying BM3D ([Dabov et al., 2007](#))
33 across multiple scales achieves superior artifact removal by exploiting the self-similarity of streak
34 patterns. However, their bm3d-streak-removal implementation is closed-source, restricted to
35 non-commercial use, and provides only x86_64 binaries incompatible with Apple Silicon and
36 modern Python versions. BM3DORNL provides the neutron imaging community with an open-
37 source, MIT-licensed, high-performance implementation that enables both high-throughput
38 batch processing and interactive parameter tuning.

39 State of the Field

40 Several software packages implement pre-processing streak removal for tomography: TomoPy
 41 ([Gürsoy et al., 2014](#)) provides wavelet-Fourier filtering ([Münch et al., 2009](#)) and Vo's sorting/fitting
 42 methods ([Vo et al., 2018](#)); ASTRA Toolbox ([Aarle et al., 2016](#)) offers GPU-accelerated
 43 preprocessing; and bm3d-streak-removal ([Mäkinen et al., 2021](#)) implements multiscale BM3D
 44 but remains closed-source and platform-limited.

45 [Figure 1](#) shows the benchmark input data: a synthetic sinogram (720×725 pixels) with simulated ring artifacts —
 46 removal, which provides only `x86_64` binaries. We compare eight methods :
 47 four BM3DORN L variants (`streak`, `generic`, `multiscale`, and `Fourier-SVD`), three TomoPy algorithms:
 48 `Fourier`, `sorting-fitting`, and `sorting-based`, and the original `bm3d-streak-removal`. The multiscale variant implements the pyramid approach from Mäkinen et al. (@mäkinen2021).
 49 `SVD` is a lightweight FFT-guided method that achieves comparable quality at 50× faster speeds.
 50



51

52 **Figure 1:** Benchmark input data. (a) Input sinogram with simulated ring artifacts. (b) Ground truth
 53 (clean sinogram). The dashed rectangle indicates the crop region shown in [Figure 2](#).

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

1000

70 reveal that all BM3DORNL variants produce clean vertical patterns, indicating successful
 71 artifact removal, while TomoPy methods leave faint residual artifacts.

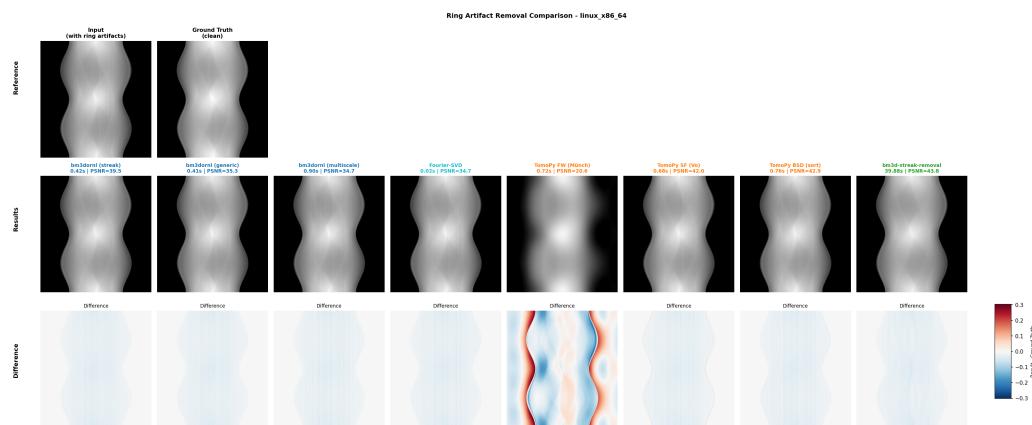


Figure 2: Method comparison. Top row: Cropped results from each method with processing times. Bottom row: Difference images (result minus ground truth) using a zero-centered RdBu colormap—blue indicates artifact removal, red indicates added artifacts. Data from Linux x86_64.

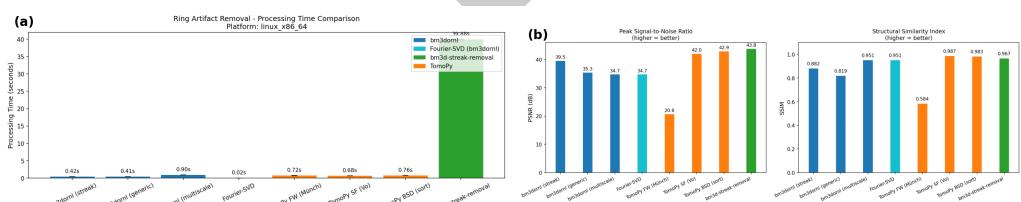


Figure 3: Performance metrics (Linux x86_64, n=100 runs). (a) Processing time comparison on linear scale, showing Fourier-SVD's 2300× speedup and standard BM3DORNL's 100× speedup over bm3d-streak-removal. (b) Quality metrics (PSNR vs SSIM) showing trade-offs between methods. BM3DORNL variants (blue/cyan), TomoPy (orange), bm3d-streak-removal (green).

72 **Platform support:** bm3d-streak-removal is unavailable on Apple Silicon (x86_64 binaries only),
 73 incompatible with Python 3.11+, and restricted to non-commercial use. BM3DORNL provides
 74 native performance on all platforms, supports Python 3.12+, and uses the MIT license for
 75 unrestricted commercial use.

76 Software Design

77 BM3DORNL employs a hybrid Python-Rust architecture: the core algorithm is implemented in
 78 Rust using the rayon crate for work-stealing parallelism, with Python bindings via PyO3 for
 79 seamless NumPy integration.

80 Key optimizations:

- 81 **Integral image pre-screening:** Before computing expensive patch distances, the block
 82 matching stage uses integral images to compute mean and norm bounds in O(1) time.
 83 Patches that fail these bounds are skipped, eliminating approximately 80% of distance
 84 calculations.
- 85 **Walsh-Hadamard transform:** For the default 8\$×\$8 patches, BM3DORNL uses Walsh-
 86 Hadamard transforms instead of FFT. This is multiplication-free (only additions and

87 subtractions), cache-friendly with fixed 64-element buffers, and computed in-place
 88 without memory allocation.

- 89 ▪ **Pre-computed FFT plans:** FFT plans are computed once and shared across threads via
 90 Arc, avoiding expensive plan initialization on each invocation (measured 45% speedup).
- 91 ▪ **Batched stack processing:** Sinograms are processed slice-by-slice to control memory
 92 usage, ensuring each slice fits in L2/L3 cache while minimizing memory allocation
 93 through buffer reuse.

94 The library provides a minimal Python API:

```
from bm3dornl import bm3d_ring_artifact_removal

cleaned = bm3d_ring_artifact_removal(
    sinogram,
    mode="streak",
    sigma=0.0, # default to automatic noise estimation
)
```

95 **GUI application:** BM3DORNL includes a native GUI built with the egui framework for Rust.
 96 The GUI enables interactive parameter tuning, side-by-side comparison with difference visual-
 97 ization, HDF5/TIFF file loading with dataset browsing, and real-time processing feedback. At
 98 0.2–0.3 seconds per sinogram, scientists can explore parameter space interactively—something
 99 impractical with bm3d-streak-removal's 41-second processing time.

100 Research Impact

101 BM3DORNL is being integrated into processing pipelines at the VENUS and MARS beamlines
 102 at Oak Ridge National Laboratory. The library provides multiple performance tiers: Fourier-
 103 SVD enables real-time parameter exploration at 0.017 seconds per sinogram (2300× faster
 104 than bm3d-streak-removal), standard BM3DORNL offers robust denoising at 0.4 seconds
 105 (100× speedup), and multiscale BM3DORNL provides maximum quality at 0.9 seconds (44×
 106 speedup). For batch processing, a 1000-sinogram dataset that would take 11 hours with bm3d-
 107 streak-removal completes in 17 seconds (Fourier-SVD), 7 minutes (standard BM3DORNL), or
 108 15 minutes (multiscale BM3DORNL).

109 The hybrid Rust-Python architecture demonstrates a modern approach to scientific software
 110 development: Rust provides memory safety and performance portability (single codebase com-
 111 piles natively to arm64 and x86_64), while Python ensures integration with the NumPy/SciPy
 112 ecosystem. This pattern is increasingly valuable as the scientific computing community
 113 diversifies hardware platforms.

114 BM3DORNL fills a critical gap: the neutron imaging community needed an open-source,
 115 actively maintained BM3D implementation that works on modern platforms. The MIT license
 116 enables unrestricted use at national facilities, and native Apple Silicon support ensures scientists
 117 can use contemporary hardware for analysis workstations.

118 AI Usage Disclosure

119 Generative AI tools (Claude) were used for code assistance and documentation drafting. All
 120 AI-generated content was reviewed, tested, and validated by the authors.

¹²¹ Acknowledgements

¹²² This research used resources at the Spallation Neutron Source, a DOE Office of Science User
¹²³ Facility operated by Oak Ridge National Laboratory. This work is supported by the U.S.
¹²⁴ Department of Energy under Contract No. DE-AC05-00OR22725.

¹²⁵ References

- ¹²⁶ Aarle, W. van, Palenstijn, W. J., Cant, J., Janssens, E., Bleichrodt, F., Dabravolski, A.,
¹²⁷ De Beenhouwer, J., Batenburg, K. J., & Sijbers, J. (2016). Fast and flexible x-ray
¹²⁸ tomography using the ASTRA toolbox. *Optics Express*, 24(22), 25129–25147. <https://doi.org/10.1364/OE.24.025129>
- ¹³⁰ Dabov, K., Foi, A., Katkovnik, V., & Egiazarian, K. (2007). Image denoising by sparse 3-d
¹³¹ transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, 16(8),
¹³² 2080–2095. <https://doi.org/10.1109/TIP.2007.901238>
- ¹³³ Gürsoy, D., De Carlo, F., Xiao, X., & Jacobsen, C. (2014). TomoPy: A framework for
¹³⁴ the analysis of synchrotron tomographic data. *Journal of Synchrotron Radiation*, 21(5),
¹³⁵ 1188–1193. <https://doi.org/10.1107/S1600577514013939>
- ¹³⁶ Mäkinen, Y., Marchesini, S., & Foi, A. (2021). Ring artifact reduction via multiscale nonlocal
¹³⁷ collaborative filtering of spatially correlated noise. *Journal of Synchrotron Radiation*, 28(3),
¹³⁸ 876–888. <https://doi.org/10.1107/S1600577521001910>
- ¹³⁹ Miqueles, E. X., Rinkel, J., O'Dowd, F., & Bermúdez, J. S. V. (2014). Generalized titarenko's
¹⁴⁰ algorithm for ring artefacts reduction. *Journal of Synchrotron Radiation*, 21(6), 1333–1346.
¹⁴¹ <https://doi.org/10.1107/S1600577514016919>
- ¹⁴² Münch, B., Trtik, P., Marone, F., & Stampanoni, M. (2009). Stripe and ring artifact
¹⁴³ removal with combined wavelet—fourier filtering. *Optics Express*, 17(10), 8567–8591.
¹⁴⁴ <https://doi.org/10.1364/OE.17.008567>
- ¹⁴⁵ Vo, N. T., Atwood, R. C., & Drakopoulos, M. (2018). Superior techniques for eliminating
¹⁴⁶ ring artifacts in x-ray micro-tomography. *Optics Express*, 26(22), 28396–28412. <https://doi.org/10.1364/OE.26.028396>