Exact Total Variation Minimizers as Non-Oscillatory Limiters in High-Order Methods for Conservation Laws

Gabriel P. Langlois[a], Jérôme Darbon[b], Rongjie Lai[c], Chi-Wang Shu[b], Xiangxiong Zhang[c,*]

[a] *Department of Mathematics, University of Illinois Urbana-Champaign, Urbana, IL 61801, USA*
[b] *Division of Applied Mathematics, Brown University, Providence, RI 02912, USA*
[c] *Department of Mathematics, Purdue University, West Lafayette, IN 47907, USA*

Abstract

High-order numerical methods for conservation laws can generate spurious oscillations near discontinuities. We propose to suppress these oscillations by post-processing the numerical solution with a total variation (TV) denoising step that reduces the total variation of the nodal values. For a general analysis operator, the resulting discrete minimization problem need not satisfy the submodularity property that existing exact max-flow algorithms require, so we instead compute the TV minimizer exactly using a differential inclusion algorithm. In one dimension, the differential inclusion algorithm computes the TV minimizer in finitely many steps, requires no tuning of algorithmic parameters, and preserves the total mass. In two dimensions, we apply the 1D algorithm dimension by dimension. We test the method as a post-processing limiter for Fourier pseudospectral methods and a fifth-order finite difference scheme applied to scalar conservation laws, compressible Euler equations, and the two-dimensional incompressible Euler equations.

Keywords: total variation denoising, non-oscillatory limiter, LASSO, differential inclusion, spectral methods, finite difference
2020 MSC: 65M70, 65M06, 65M12, 65K10, 90C25

*Corresponding author
Email addresses: gp42@illinois.edu (Gabriel P. Langlois), jerome_darbon@brown.edu (Jérôme Darbon), lairj@purdue.edu (Rongjie Lai), chi-wang_shu@brown.edu (Chi-Wang Shu), zhan1966@purdue.edu (Xiangxiong Zhang)

1. Introduction

1.1. Oscillations in high-order methods for conservation laws

Consider the scalar conservation law

$$u_t + f(u)_x = 0 \tag{1}$$

with suitable initial data. It is well known that solutions of (1) develop discontinuities in finite time even from smooth initial data, and that linear high-order methods for smooth solutions generate spurious oscillations near such discontinuities. For spectral methods [1, 2], the global Fourier or polynomial basis produces Gibbs oscillations that are $O(1)$ in amplitude and that pollute the entire computational domain. See [3, 4] for non-oscillatory high-order finite difference and finite volume methods such as ENO and WENO schemes. For discontinuous Galerkin (DG) methods [5, 6], limiters are needed to treat troubled cells.

Existing remedies are usually designed within a specific discretization framework. For spectral methods, common approaches include exponential filters [7], spectral viscosity [8], and Gegenbauer reconstruction [9]. For finite volume and DG methods, common approaches include TVD limiters [10], WENO reconstruction [4, 11], invariant-domain-preserving (IDP) limiters [12, 13, 14], flux-corrected transport (FCT) [15, 16], and optimization-based limiters [17, 18]. We refer to [19] for a comprehensive survey of IDP schemes. The bound-preserving sweeping technique [20] enforces pointwise bounds while conserving a weighted average, but it does not directly reduce the total variation.

1.2. TV regularization

Given a numerical solution $\mathbf{b} \in \mathbb{R}^n$ contaminated by oscillations, the TV denoising problem seeks

$$\mathbf{x}^* = \underset{\mathbf{x}\in\mathbb{R}^n}{\arg\min}\ \|D\mathbf{x}\|_1 + \frac{1}{2\lambda}\|\mathbf{x}-\mathbf{b}\|_2^2, \tag{2}$$

where $D \in \mathbb{R}^{(n-1)\times n}$ is the forward difference matrix, i.e., $(D\mathbf{x})_i = x_{i+1} - x_i$, and $\lambda > 0$ is a chosen parameter to control the regularization strength. The minimizer of (2) preserves the total mass of the input: $\sum_i x_i^* = \sum_i b_i$. This follows from $\ker(D) = \text{span}\{\mathbf{1}\}$ (see Section 2.2) and ensures that the TV limiter defined by (2) conserves total mass. TV regularization was introduced for image denoising [21, 22]. In this paper, we use (2) as a limiter to reduce the total variation of the nodal values.

1.3. Computational cost of TV solvers

In a PDE context where the TV filter may be invoked at every time step, iterative solver cost can dominate the computation. The key focus of this paper is to introduce an efficient exact solver for (2).

Table 1: Solver comparison for selected test problems. Steps and DI time are for the differential inclusion algorithm [23] (C++ implementation). CD (C++ implementation) time is for the Chambolle–Darbon parametric max-flow solver [24, 25]. All times are online solve times on a MacBook Pro with Apple M1 chip (offline precomputation of $(D^\top D)^\dagger D^\top$ excluded). Times are averages over R runs, where R is chosen so that the total elapsed time exceeds one second.

Test problem	n	λ	DI Steps	DI (s)	CD (s)	Sec.
Gaussian noise, pw. const.	200	1.0	13	4.5e-5	3.6e-5	§3.3.1
	1000	1.0	59	2.5e-3	3.1e-4	§3.3.1
Spectral Gibbs, pw. const.	200	0.3	3	1.8e-5	4.6e-5	§3.3.2
	2000	0.3	3	1.5e-3	3.2e-3	§3.3.2
Spectral Gibbs, pw. smooth	200	0.08	110	3.4e-3	3.5e-5	§3.3.3
	1000	0.08	562	4.2e-1	2.4e-4	§3.3.3
FD5 Burgers post-proc.	256	0.02	244	4.1e-2	4.2e-5	§4.1

For the one-dimensional problem (2), there are several specialized exact or fast solvers, including the taut string algorithm [26], the direct algorithm of Condat [27], and LARS/homotopy path-following for the generalized LASSO [28, 29]. The 1D TV problem reduces to a LASSO problem through the change of variables $\mathbf{z} = D\tilde{\mathbf{x}}$, where $\tilde{\mathbf{x}} = \mathbf{x} - \mathrm{mean}(\mathbf{x})\,\mathbf{1}$ is the mean-subtracted part of $\mathbf{x}$; see Section 2.2. The taut string algorithm is restricted to the 1D TV problem, and the Condat algorithm extends to the 1D fused LASSO (TV plus ℓ^1 sparsity), with $O(n)$ complexity in practice [27].

An earlier exact solver for the TV denoising problem (2), applicable when the problem satisfies a certain submodularity property, is the parametric max-flow algorithm [25, 30, 31, 32, 33, 24]. This algorithm reformulates the problem as a sequence of binary min-cut problems on the same graph, solved simultaneously by a single parametric max-flow computation. For the 1D TV problem, this reduces to a path graph.

The recent differential inclusion (DI) algorithm for the LASSO problem by Langlois and Darbon [23] exploits the polyhedral structure of the dual problem by tracing the piecewise continuous trajectory of its associated differential inclusions exactly, terminating in finitely many steps. Their algorithm is a novel extension of the differential inclusions algorithm proposed

in [34] for solving the equality-constrained basis pursuit problem efficiently and exactly, numerically up to machine precision.

Table 1 compares the two solvers: for test cases with few breakpoints, an optimized C++ implementation of the DI algorithm is *comparable to or faster* than the parametric max-flow algorithm (e.g., for the 3-step piecewise constant cases the DI solver is roughly 2–3× faster at both $n = 200$ and $n = 2000$), while for cases with many breakpoints (tens of steps and up) the max-flow algorithm is substantially faster. The max-flow algorithm requires a certain submodularity property, which does not hold for other analysis operators D such as those in Table 2 or Appendix A. The DI solver solves the LASSO for any measurement matrix, so it applies to any surjective analysis operator D, e.g., see Table 2 and examples (db1 and db4 wavelets) in Appendix A.

Table 2: CPU time comparison of MATLAB and C++ implementations of the DI algorithm with D replaced by a *fourth-order accurate* finite difference approximation to du/dx (interior rows: $[1, -8, 0, 8, -1]/12$, boundary rows: $[-1, 1]$). All times are online solve times on a MacBook Pro with Apple M1 chip.

Test problem	n	λ	DI Steps	DI MATLAB (s)	DI C++ (s)
Gaussian noise, pw. const.	200	1.0	32	1.0e-2	3.0e-4
	1000	1.0	138	2.8e-1	2.7e-2
Spectral Gibbs, pw. const.	200	0.5	24	2.0e-3	2.0e-4
	2000	0.5	282	3.1e+0	3.8e-1
Spectral Gibbs, pw. smooth	200	0.5	175	5.4e-2	3.3e-3
	1000	0.5	1441	2.4e+1	1.3e+0

For a general operator D, popular iterative methods for (2) include standard iterative solvers such as the Chambolle–Pock primal-dual hybrid gradient (PDHG) method [35], and the alternating direction method of multipliers (ADMM), which can be interpreted through Douglas–Rachford splitting [36, 37, 38]. Their performance depends on algorithmic parameters that require problem-dependent tuning.

For the simpler scalar bound-preserving optimization limiter with a conservation constraint, Bradley et al. [39] showed that the ℓ^2 minimizer is also an ℓ^1 minimizer and that an ℓ^1 minimizer can be constructed explicitly by clipping to bounds and redistributing the mass deficit, though ℓ^1 minimizers are generally non-unique [40]. Such a construction does not extend to the ℓ^1-penalized LASSO arising from TV denoising.

1.4. Main results

We apply the DI algorithm [23] developed by Langlois and Darbon to the TV denoising problem (2) and use it as a non-oscillatory limiter for numerical conservation law solvers. The algorithm computes the minimizer of (2) in finitely many steps, with the step count depending on signal complexity rather than grid size n for clean piecewise-constant data, such as the truncated Fourier sums of step functions in Section 3.3.2; for noisy piecewise-constant data (Section 3.3.1) and for piecewise-smooth signals the step count grows with n (Section 3.3.3). The matrix $(D^\top D)^\dagger D^\top$ (where $\dagger$ denotes the Moore–Penrose pseudoinverse) is formed once as an offline precomputation in $O(n^2 \log n)$ work using the known eigendecomposition of $D^\top D$ (the discrete Neumann Laplacian); for the general fourth-order operator of Table 2, $M = D^\dagger$ is instead formed by a dense pseudoinverse offline. The times reported below include only the online solve. Table 1 summarizes the solver performance for selected test problems.

Table 2 demonstrates the generality of the DI algorithm when we replace the first-difference operator D with a fourth-order accurate finite difference approximation to du/dx (5-point interior stencil). The max-flow algorithm cannot handle this operator because its graph-cut formulation requires each row of D to couple at most two variables with the sign pattern that makes the pairwise terms submodular. The DI algorithm applies without modification: only the offline pseudoinverse $(D^\top D)^\dagger D^\top$ changes. In Table 2, we also list the comparison of MATLAB and C++ implementations of the DI method.

As a limiter, the TV denoising step (2) is applied to the nodal values of the numerical solution at each time step or as a post-processing filter. We test the limiter on scalar convection equations using Fourier pseudospectral discretizations, and on the compressible Euler equations using a fifth-order finite difference scheme. In two dimensions, we apply the 1D TV limiter dimension-by-dimension.

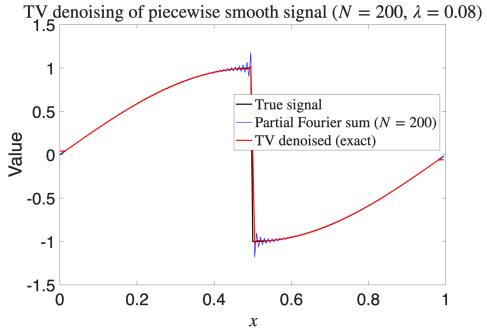
1.5. Organization

The rest of the paper is organized as follows. Section 2 formulates TV denoising as a non-oscillatory limiter, derives the reduction to LASSO, describes the dimension-by-dimension extension to 2D, and reviews PDHG and ADMM. Section 3 presents the DI algorithm and tests it on a few examples as a demonstration of its performance. Section 4 applies the TV limiter to Fourier pseudospectral and fifth-order finite difference computations of conservation laws. The concluding remarks are given in Section 5.

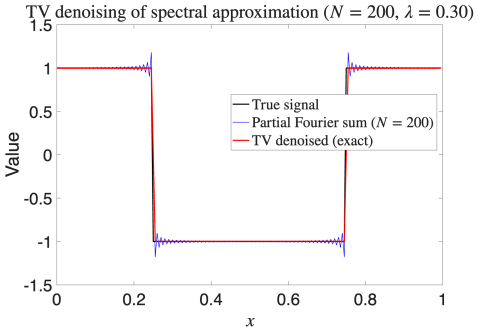
2. TV Limiter Formulation

2.1. TV denoising as a non-oscillatory limiter

We post-process the numerical solution by solving the TV denoising problem (2), where D is a first-order finite difference operator and $\lambda > 0$ controls the strength of regularization. We refer to this approach as a *TV limiter*. Figure 1 shows TV denoising applied to two spectral approximations with Gibbs oscillations.



(a) Piecewise smooth signal ($N = 200$ Fourier modes, $\lambda = 0.08$). Blue: partial Fourier sum. Black: true piecewise sinusoidal signal. Red: TV denoised.



(b) Piecewise constant signal ($N = 200$ Fourier modes, $\lambda = 0.3$). Blue: partial Fourier sum. Black: true step function. Red: TV denoised.

Figure 1: TV denoising as a non-oscillatory filter or limiter for removing Gibbs oscillations from truncated Fourier approximations. Left: piecewise smooth signal with staircasing artifact visible but mild. Right: piecewise constant signal recovered to plotting accuracy.

The forward difference matrix $D \in \mathbb{R}^{(n-1)\times n}$ is

$$D = \begin{pmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{pmatrix}, \quad (D\mathbf{x})_i = x_{i+1} - x_i, \quad i = 1, \ldots, n-1,$$

with $\ker(D) = \operatorname{span}\{\mathbf{1}\}$. The ℓ_1 penalty promotes piecewise constant structure, suppressing oscillations while preserving sharp jumps. Since $\ker(D) = \operatorname{span}\{\mathbf{1}\}$, the TV penalty is blind to the mean of $\mathbf{x}$, so the quadratic fidelity term forces $\sum_i x_i^* = \sum_i b_i$, preserving the total mass exactly.

2.2. Reduction to LASSO

For any vector $\mathbf{v} \in \mathbb{R}^n$, write its mean-subtracted part as $\tilde{\mathbf{v}} = \mathbf{v} - \bar{v}\mathbf{1}$ with $\bar{v} = \mathrm{mean}(\mathbf{v})$, so that $\tilde{\mathbf{v}} \perp \mathbf{1}$. Since $D\mathbf{1} = \mathbf{0}$, we decompose $\mathbf{x} = \bar{x}\mathbf{1} + \tilde{\mathbf{x}}$, so that the TV term depends only on $\tilde{\mathbf{x}}$. The full objective then splits into two independent pieces:

$$\|D\mathbf{x}\|_1 + \frac{1}{2\lambda}\|\mathbf{x} - \mathbf{b}\|_2^2 = \underbrace{\frac{n}{2\lambda}(\bar{x} - \bar{b})^2}_{\text{minimized by } \bar{x}^* = \bar{b}} + \underbrace{\|D\tilde{\mathbf{x}}\|_1 + \frac{1}{2\lambda}\|\tilde{\mathbf{x}} - \tilde{\mathbf{b}}\|_2^2}_{\text{TV in } \tilde{\mathbf{x}}}.$$

Setting $\mathbf{z} = D\tilde{\mathbf{x}}$ and recovering $\tilde{\mathbf{x}} = (D^\top D)^\dagger D^\top \mathbf{z}$ via a Poisson solve (see Section 3.2), the remaining minimization becomes the LASSO problem

$$\min_{\mathbf{z} \in \mathbb{R}^{n-1}} \ \|\mathbf{z}\|_1 + \frac{1}{2\lambda}\|M\mathbf{z} - \tilde{\mathbf{b}}\|_2^2, \qquad M = (D^\top D)^\dagger D^\top. \tag{3}$$

After solving for $\mathbf{z}^*$, the full solution is recovered as $\mathbf{x}^* = M\mathbf{z}^* + \bar{b}\mathbf{1}$. The difference operator $D \in \mathbb{R}^{(n-1)\times n}$ has rank $n-1$ and is surjective onto $\mathbb{R}^{n-1}$, so the change of variables $\mathbf{z} = D\tilde{\mathbf{x}}$ is a bijection between $\mathbf{1}^\perp$ and $\mathbb{R}^{n-1}$ and the LASSO reduction is exact.

2.3. Extension to two dimensions

In 2D, the stacked difference operator $D = \left[\begin{smallmatrix} D_x \\ D_y \end{smallmatrix}\right]$ is not surjective, and the LASSO reduction introduces a relaxation gap. We instead apply the 1D TV denoiser dimension-by-dimension: given $U \in \mathbb{R}^{N_x \times N_y}$, first solve the 1D TV problem for each column, then for each row. Each sweep preserves the column (resp. row) total mass, so the overall total mass of U is preserved. The cost is N_y 1D solves of size N_x plus N_x solves of size N_y, each terminating in finitely many breakpoint steps. A single pass suffices for limiting the 2D solution.

2.4. Classical iterative solvers

We briefly review two widely used iterative solvers for TV minimization. Both methods apply to any analysis operator D (not just the first-difference operator).

Chambolle–Pock accelerated PDHG [35]. The PDHG algorithm initializes two step sizes $\tau_0 = \sigma_0 = 0.5$ satisfying $\tau_0\sigma_0\|D\|_2^2 \le 1$ (since $\|D\|_2 < 2$) and iterates repeatedly:

1. *Dual step*: $\mathbf{y}^{k+1} = \mathrm{clip}\big(\mathbf{y}^k + \sigma_k D\hat{\mathbf{x}}^k,\, -1,\, 1\big)$
2. *Primal step*: $\mathbf{x}^{k+1} = \dfrac{\mathbf{x}^k - \tau_k D^\top \mathbf{y}^{k+1} + \frac{\tau_k}{\lambda}\mathbf{b}}{1 + \tau_k/\lambda}$
3. *Acceleration* ($\mu = 1/\lambda$): $\theta_{k+1} = \frac{1}{\sqrt{1+2\tau_k/\lambda}}$, $\tau_{k+1} = \theta_{k+1}\tau_k$, $\sigma_{k+1} = \sigma_k/\theta_{k+1}$
4. *Extrapolation*: $\hat{\mathbf{x}}^{k+1} = \mathbf{x}^{k+1} + \theta_{k+1}(\mathbf{x}^{k+1} - \mathbf{x}^k)$

ADMM (Douglas–Rachford). The ADMM algorithm splits $\mathbf{z} = D\mathbf{x}$ with penalty $\eta = 10/\lambda$ and iterates repeatedly:

1. $\mathbf{x}$-*update* (tridiagonal solve): $\big(\frac{1}{\lambda}I + \eta D^\top D\big)\mathbf{x} = \frac{1}{\lambda}\mathbf{b} + \eta D^\top(\mathbf{z} - \mathbf{v})$
2. $\mathbf{z}$-*update* (soft-thresholding): $\mathbf{z} = S_{1/\eta}(D\mathbf{x} + \mathbf{v})$, where

$$S_\alpha(w)_j = \mathrm{sign}(w_j)\max(|w_j| - \alpha,\, 0)$$

3. $\mathbf{v}$-*update*: $\mathbf{v} \leftarrow \mathbf{v} + D\mathbf{x} - \mathbf{z}$

Both methods are convergent for the composite minimization of lower semi-continuous proper convex functions. Accelerated PDHG attains the $O(1/k^2)$ rate, and ADMM often converges faster in practice but at a higher cost per iteration (one tridiagonal solve instead of two sparse matrix–vector products). Tuning the ADMM penalty η is crucial, and a poor choice of η can slow convergence by orders of magnitude (see Figure 3 in Section 3.3.2). ADMM is known as an efficient solver for TV minimization [38], and a detailed treatment can be found in [41, Section 4.2].

3. The Langlois–Darbon Differential Inclusion Algorithm

3.1. The dual problem and its polyhedral structure

The dual problem of (3), derived by introducing the dual variable $\mathbf{p} = (M\mathbf{z} - \tilde{\mathbf{b}})/\lambda$ and dualizing the ℓ_1 term, is

$$\min_{\mathbf{p}\in\mathbb{R}^n} \frac{\lambda}{2}\|\mathbf{p}\|_2^2 + \langle \mathbf{p}, \tilde{\mathbf{b}}\rangle \quad \text{subject to} \quad \|M^\top \mathbf{p}\|_\infty \le 1. \tag{4}$$

The constraint $\|M^\top\mathbf{p}\|_\infty \le 1$ (with $M^\top\mathbf{p} \in \mathbb{R}^{n-1}$) defines a closed convex polyhedron, the same polyhedral structure as in the standard LASSO.

3.2. Implementation

The DI algorithm solves the dual problem (4) by computing the *breakpoints* of the piecewise continuous trajectory of its associated differential inclusions. At a feasible point $\mathbf{p}$, the *equicorrelation set* $\mathcal{E}(\mathbf{p}) = \{j : |(M^\top \mathbf{p})_j| = 1\}$ identifies the active constraints along which the trajectory evolves. The algorithm proceeds as follows:

Step 1. Offline precomputation. Form the dense matrix $M = D^\dagger = (D^\top D)^\dagger D^\top \in \mathbb{R}^{n\times(n-1)}$, which is independent of the data $\mathbf{b}$.

Step 2. Initialization. Set $\tilde{\mathbf{b}} = \mathbf{b} - \mathrm{mean}(\mathbf{b}) \cdot \mathbf{1}$. If $\tilde{\mathbf{b}} = \mathbf{0}$ (constant input), the solution is $\mathbf{x}^* = \bar{b}\,\mathbf{1}$ and the algorithm terminates. Otherwise, set $\mathbf{p}_0 = -\tilde{\mathbf{b}}/\|M^\top \tilde{\mathbf{b}}\|_\infty$ (the unique scaling that places $\mathbf{p}_0$ on the boundary of the polyhedron) and compute $\mathcal{E}_0$.

Step 3. NNLS subproblem. At breakpoint $\mathbf{p}_k$ with active set $\mathcal{E}_k$ and signs $s_j = \mathrm{sign}(-(M^\top \mathbf{p}_k)_j)$, form the columns $\tilde{M} = [s_j\, M(:,j)]_{j\in\mathcal{E}_k}$ and solve the NNLS problem $\min_{\mathbf{u}\geq 0} \|\tilde{M}\mathbf{u} - (\lambda \mathbf{p}_k + \tilde{\mathbf{b}})\|_2^2$. The descent direction is $\mathbf{d}_k = \tilde{M}\mathbf{u} - (\lambda \mathbf{p}_k + \tilde{\mathbf{b}})$.

Step 4. Ratio test. Compute the step Δ_* to the next breakpoint: the smallest $\Delta > 0$ at which some inactive constraint $|(M^\top(\mathbf{p}_k + \Delta \mathbf{d}_k))_j| = 1$ becomes active. If $\lambda\Delta_* \geq 1$, the algorithm has converged. Otherwise, update $\mathbf{p}_{k+1} = \mathbf{p}_k + \Delta_*\, \mathbf{d}_k$.

Step 5. Recovery. Set $z_j^* = s_j u_j$ for $j \in \mathcal{E}$ (zero elsewhere) and recover $\mathbf{x}^* = M\mathbf{z}^* + \mathrm{mean}(\mathbf{b}) \cdot \mathbf{1}$.

Finite termination is guaranteed: $\|\mathbf{d}_k\|_2$ strictly decreases along each face of the polyhedron defined by $\|M^\top \mathbf{p}\|_\infty \leq 1$, so the algorithm terminates [23].

Forming M via the eigendecomposition of $D^\top D$. The matrix $D^\top D$ is the 1D Neumann Laplacian, a tridiagonal $n \times n$ matrix whose eigenvalues and orthonormal eigenvectors are known in closed form [41]:

$$\mu_k = 2 - 2\cos\big(\pi k/n\big), \qquad v_k(j) = c_k \cos\Big(\frac{k\pi(j-\frac{1}{2})}{n}\Big),$$

for $j = 1, \ldots, n$ and $k = 0, \ldots, n-1$, where $c_0 = 1/\sqrt{n}$ and $c_k = \sqrt{2/n}$ for $k \geq 1$. The eigenvectors correspond to the DCT-II basis on the one-half

grid $x_j = (j - \frac{1}{2})/n$. Setting $1/\mu_0 = 0$ (pseudoinverse convention for the zero eigenvalue), we compute

$$M = (D^\top D)^\dagger D^\top = C^\top \operatorname{diag}(\mu_0^{-1}, \ldots, \mu_{n-1}^{-1})\, C\, D^\top,$$

where C is the orthogonal matrix with entries $C_{kj} = c_k \cos\big(k\pi(j - \frac{1}{2})/n\big)$. This offline computation costs $O(n^2 \log n)$ via the DCT (or $O(n^3)$ with dense matrix multiplications) and produces a dense $n \times (n-1)$ matrix (less than 8 MB for $n = 1000$). Once M is stored, column extraction $M(:, \mathcal{E})$ is a trivial indexing operation, and the matrix–vector products $M\mathbf{z}$ and $M^\top \mathbf{p}$ needed at each breakpoint are standard dense matvecs.

Warm-started NNLS via Meyer's algorithm. The NNLS subproblem computed at each breakpoint is the computational bottleneck. We solve it using Meyer's algorithm [42], a generalization of the classical Lawson–Hanson active-set method that accepts an arbitrary starting active set. Because the equicorrelation set $\mathcal{E}$ typically grows by only one index per breakpoint, the passive set from the previous solve provides a near-optimal warm start: the NNLS needs only $O(1)$ active-set pivots instead of $O(|\mathcal{E}|)$ from a cold start. In addition, columns of M that have been extracted in previous breakpoints are cached so that only newly activated indices require a column lookup. The combination yields a roughly $5\times$ speedup at $n = 1000$.

Complexity. Each breakpoint step involves one NNLS solve and one ratio test ($O(n^2)$ for computing $M^\top \mathbf{d}$ and checking constraints). A cold-start NNLS solve costs on the order of $|\mathcal{E}|^2 n$ flops with active-set pivots, but with the warm start described above the NNLS typically needs only $O(1)$ pivots of $O(|\mathcal{E}|\, n)$ flops each, so the observed cost per breakpoint step is $O(n^2)$. Over K breakpoint steps the total empirical cost is therefore $O(Kn^2)$, and since $K = O(n)$ in all our tests ($K \le 1.5n$; see Tables 1 and 2), the empirical worst-case complexity is $O(n^3)$. Although the worst-case complexity of the NNLS problem with active-set methods remains open, in practice, for clean piecewise-constant data such as the truncated Fourier sums of step functions in Section 3.3.2, the number of breakpoint steps is much smaller than n and depends on signal complexity rather than grid size (see Table 1).

3.3. Numerical comparison

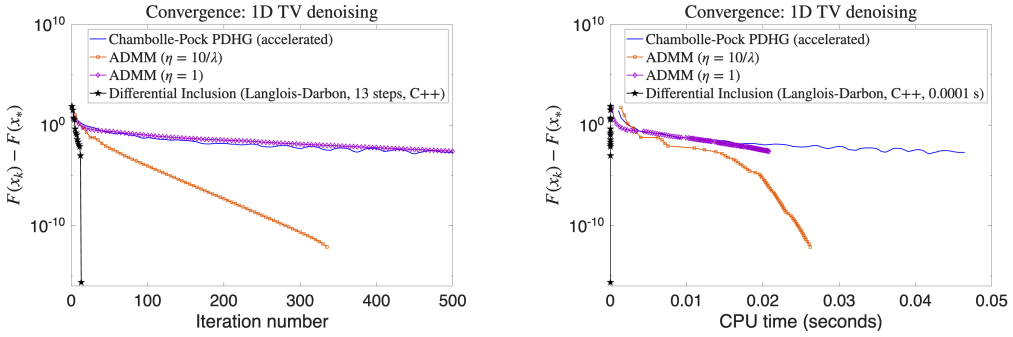
We compare the DI algorithm with the iterative methods of Section 2.4 on TV denoising problems. All three methods minimize the same 1D TV

objective $F(\mathbf{x}) = \|D\mathbf{x}\|_1 + \frac{1}{2\lambda}\|\mathbf{x} - \mathbf{b}\|_2^2$. The DI algorithm is implemented through the equivalent LASSO problem $\min_{\mathbf{z}\in\mathbb{R}^{n-1}} \|\mathbf{z}\|_1 + \frac{1}{2\lambda}\|M\mathbf{z} - \tilde{\mathbf{b}}\|_2^2$ with $M = D^\dagger$ and $\tilde{\mathbf{b}} = \mathbf{b} - \bar{b}\,\mathbf{1}$. Since D is surjective, this reformulation is exact.

3.3.1. Gaussian noise denoising

We first consider a piecewise constant signal corrupted by additive Gaussian noise. The true signal on $n = 200$ grid points has five constant segments with values $\{1, -0.5, 2, 0, -1\}$ and four jumps. The observation is $\mathbf{b} = \mathbf{x}_{\text{true}} + \boldsymbol{\varepsilon}$, where $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, 0.3^2 I)$, and we use $\lambda = 1$.

The DI algorithm converges in 13 breakpoint steps. As shown in Figure 2, ADMM with tuned $\eta = 10/\lambda$ reaches $\sim 10^{-13}$ within 500 iterations, while PDHG and untuned ADMM stall near 10^{-3}. At $n = 1000$ with the same noise level and proportionally scaled jump locations, the solver converges in 59 steps.



(a) Objective error vs. iteration number.

(b) Objective error vs. CPU time.

Figure 2: Convergence comparison for TV denoising of a piecewise constant signal with Gaussian noise ($n = 200$, $\sigma = 0.3$, $\lambda = 1$). The DI algorithm converges in 13 steps.

3.3.2. Spectral Gibbs denoising: piecewise constant signal

Consider removing Gibbs oscillations from a truncated Fourier series. The true signal is a piecewise constant function on $[0, 1)$ with two jumps:

$$f(x) = \begin{cases} +1 & x \in [0, 0.25) \cup [0.75, 1), \\ -1 & x \in [0.25, 0.75). \end{cases}$$

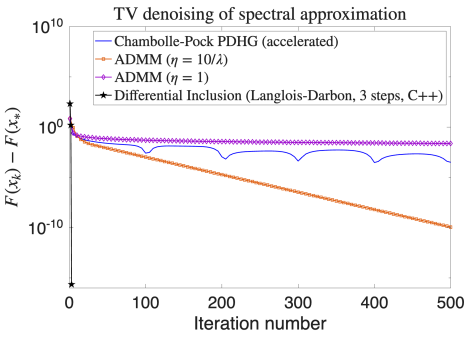
The "noisy" signal $\mathbf{b}$ is the N-term partial Fourier sum of f, evaluated at $n = N$ uniform grid points $x_j = j/N$:

$$b_j = \sum_{m=-N/2+1}^{N/2} \hat{f}_m \, e^{2\pi i m x_j}, \qquad \hat{f}_m = \int_0^1 f(x)\, e^{-2\pi i m x}\, dx.$$

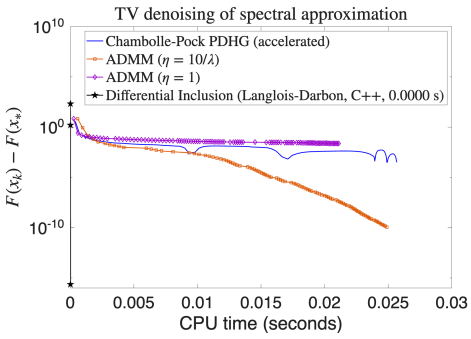
(The real part of the partial sum is taken, so that the unpaired mode $m = N/2$ does not introduce a spurious imaginary component.) This is the output of a Fourier spectral method on an N-point grid, with Gibbs oscillations near each discontinuity ($\|\mathbf{b} - \mathbf{x}_{\text{true}}\|_2 \approx 1.49$).

We apply the TV denoising model $F(\mathbf{x}) = \|D\mathbf{x}\|_1 + \frac{1}{2\lambda}\|\mathbf{x} - \mathbf{b}\|_2^2$ with $\lambda = 0.3$. This is smaller than the choice $\lambda = 1$ used in Section 3.3.1, reflecting the lower effective noise level. Figure 1b (Section 2) shows that TV denoising removes the Gibbs oscillations and recovers the piecewise constant structure.

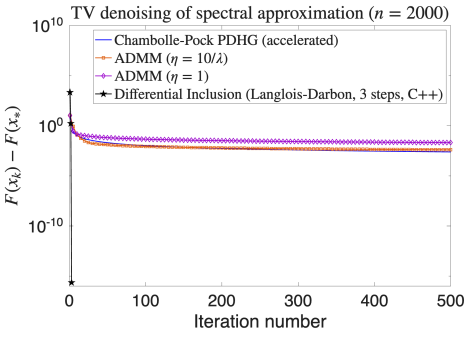
At $N = 200$, the DI algorithm converges in 3 breakpoint steps. At $N = 2000$, it again converges in 3 breakpoint steps. The step count does not increase with N because the underlying piecewise constant structure is unchanged. The iterative methods stall near 10^{-2}–10^{-3} within 500 iterations (see Figure 3).



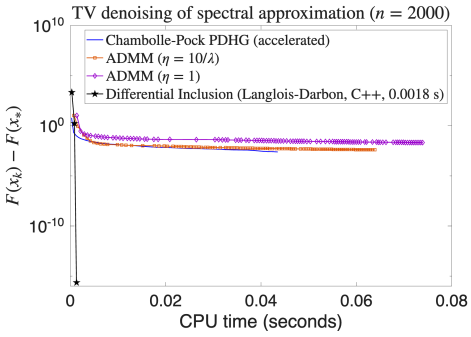
(a) $N = 200$: objective error vs. iteration.



(b) $N = 200$: objective error vs. CPU time.



(c) $N = 2000$: objective error vs. iteration.



(d) $N = 2000$: objective error vs. CPU time.

Figure 3: Convergence comparison for TV denoising of the partial Fourier sum of a piecewise constant signal ($\lambda = 0.3$). The DI algorithm converges in 3 steps at both $N = 200$ and $N = 2000$.

3.3.3. Spectral Gibbs denoising: piecewise smooth signal

The previous test used a piecewise *constant* signal, for which the DI algorithm required only a few breakpoint steps. Consider a piecewise *smooth* signal:

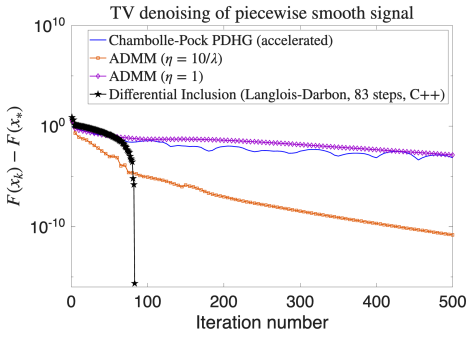
$$f(x) = \begin{cases} \sin(\pi x) & x \in [0, 0.5), \\ -\sin(\pi x) & x \in [0.5, 1), \end{cases}$$

which has a jump discontinuity of size 2 at $x = 0.5$ (from $\sin(\pi/2) = 1$ to $-\sin(\pi/2) = -1$), while the smooth portions are sinusoidal arcs. As before, the "noisy" signal $\mathbf{b}$ is the N-term partial Fourier sum of f evaluated at $n = N$ uniform grid points, producing Gibbs oscillations near $x = 0.5$. We apply TV denoising with $\lambda = 1$.

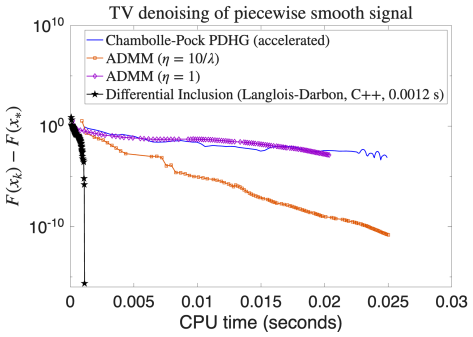
As shown in Figure 1a (Section 2), the Gibbs oscillations are eliminated while the sinusoidal profile is retained in the smooth regions. That figure uses a milder regularization ($\lambda = 0.08$) chosen for visual clarity; the $\lambda = 0.08$ setting is the one timed in Table 1, while the convergence study here uses $\lambda = 1$ (Figure 4). At $N = 200$, the DI algorithm requires 83 breakpoint steps, compared with 3 steps for the piecewise constant signal. The TV-optimal reconstruction of a smooth signal has many nonzero differences $D\mathbf{x}^*$, so the equicorrelation set grows to a large fraction of the $n - 1 = 199$ possible indices. For a piecewise constant signal with k jumps, the equicorrelation set typically has size $\sim k$. PDHG and untuned ADMM stall near 10^{-3} after 500 iterations.

At $N = 1000$ with $\lambda = 1$, the DI algorithm requires 496 breakpoint steps. The increase reflects the finer grid, which produces more nonzero differences in the TV-optimal solution and enlarges the equicorrelation set. ADMM with tuned $\eta = 10/\lambda$ reaches $\sim 10^{-6}$, while PDHG and untuned ADMM stall near 10^{-2}. Figure 4 shows the convergence for both $N = 200$ and $N = 1000$.

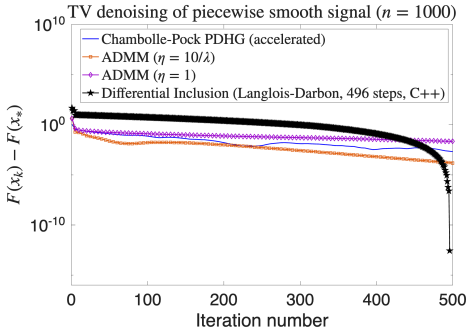
At first glance, Figures 4b and 4d may suggest that ADMM and PDHG are competitive in CPU time. However, Figure 4c shows that at $N = 1000$ both iterative methods remain far from convergence after 500 iterations. Each step of the DI algorithm is more expensive than one ADMM or PDHG iteration, since it involves an NNLS solve. Even so, ADMM and PDHG need substantially more iterations to reach the same accuracy.



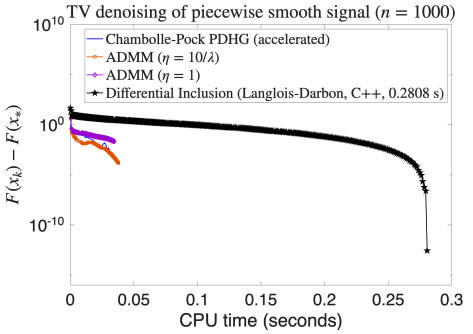
(a) $N = 200$: objective error vs. iteration.



(b) $N = 200$: objective error vs. CPU time.



(c) $N = 1000$: objective error vs. iteration.



(d) $N = 1000$: objective error vs. CPU time.

Figure 4: Convergence comparison for TV denoising of the partial Fourier sum of a piecewise smooth signal ($\lambda = 1$).

4. TV Limiter for Conservation Laws

We use TV denoising as a *limiter* in time-dependent PDE computations. We consider two classes of base schemes: a fifth-order upwind finite difference scheme (FD5), used for scalar Burgers' equation (Section 4.1), the 1D compressible Euler equations (Section 4.2), and a 2D Riemann problem for compressible Euler (Section 4.6); and Fourier pseudospectral methods for two-dimensional scalar convection and incompressible Euler (Sections 4.3–4.5). When applied as a per-step limiter, the TV denoising step is invoked every K time steps, but only when the current total variation $\mathrm{TV}(u^n)$ exceeds a threshold $\mathrm{TV}_{\mathrm{thresh}} = 1.2\,\mathrm{TV}(u_0)$, where $\mathrm{TV}(u_0)$ is the total variation of the initial condition.

4.1. FD5 for Burgers' equation: shock capturing

Consider Burgers' equation with periodic boundary conditions:

$$u_t + \Big(\frac{u^2}{2}\Big)_x = 0, \qquad x \in [0, 2\pi), \quad u(x,0) = 1 + \sin(x).$$

The shock forms at $t_s = -1/\min_x u_0'(x) = 1$ and propagates at speed 1. We discretize with $N = 256$ uniform grid points $x_i = i\,\Delta x$, $\Delta x = 2\pi/N$, using the fifth-order linear upwind conservative finite difference scheme (FD5)

$$\frac{d}{dt}u_i = -\frac{1}{\Delta x}\big(\hat{f}_{i+1/2} - \hat{f}_{i-1/2}\big), \tag{5}$$

with Lax–Friedrichs flux splitting [3, 11]. The flux is split as $f^{\pm}(u) = \frac{1}{2}(f(u) \pm \alpha u)$ with $\alpha = \max|f'(u)|$, and the numerical flux $\hat{f}_{i+1/2} = \hat{f}^{+}_{i+1/2} + \hat{f}^{-}_{i+1/2}$ is reconstructed as

$$\hat{f}^{+}_{i+1/2} = \frac{1}{60}\big(2f^{+}_{i-2} - 13f^{+}_{i-1} + 47f^{+}_{i} + 27f^{+}_{i+1} - 3f^{+}_{i+2}\big),$$

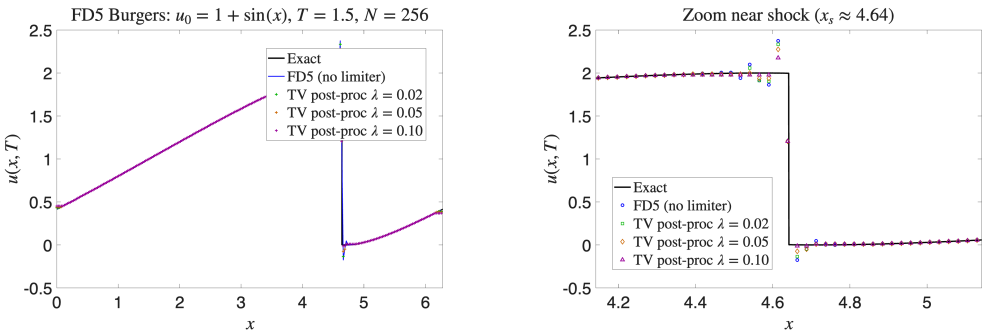
$$\hat{f}^{-}_{i+1/2} = \frac{1}{60}\big(-3f^{-}_{i-1} + 27f^{-}_{i} + 47f^{-}_{i+1} - 13f^{-}_{i+2} + 2f^{-}_{i+3}\big).$$

This is the linear scheme to which WENO5 [11] reduces when the optimal linear weights are used in place of the nonlinear WENO weights. We integrate (5) in time with RK4 (time step $\Delta t = 0.5\,\Delta x$, so the CFL number $\Delta t \max|u|/\Delta x \approx 1$ since $\max|u| \approx 2$; 123 steps to $T = 1.5$).

For this test case, the FD5 scheme is stable without any limiter: the no-limiter solution has L^2 error 5.75×10^{-2} and TV $= 5.83$ (versus exact TV $= 3.99$), with oscillations confined near the shock. TV post-processing with different regularization strengths reduces the error:

Method	L^2 error	TV
FD5, no limiter	5.75×10^{-2}	5.83
TV post-proc λ=0.02	5.53×10^{-2}	5.23
TV post-proc λ=0.05	5.31×10^{-2}	4.76
TV post-proc λ=0.10	5.11×10^{-2}	4.29

At $\lambda = 0.10$, TV post-processing reduces the L^2 error by 11% and brings the total variation closer to the exact value. The total mass is preserved to machine precision ($|\bar{u}_h - \bar{u}_0| < 10^{-15}$).

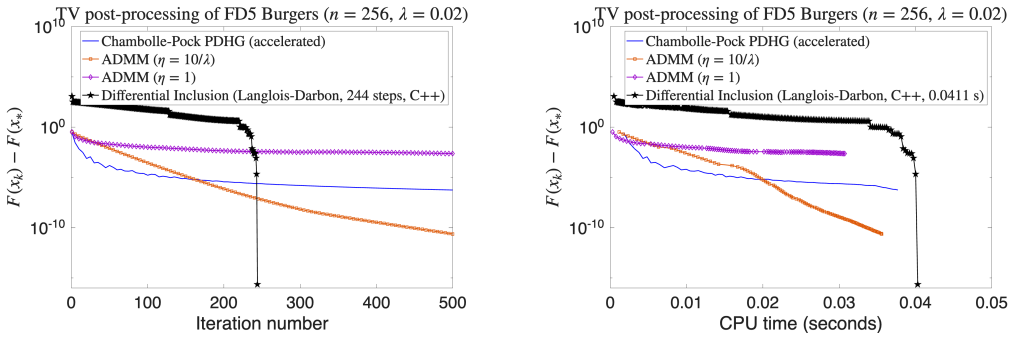


(a) Solution at $T = 1.5$.

(b) Zoom near the shock.

Figure 5: FD5 Burgers' equation with $u_0 = 1 + \sin(x)$ at $T = 1.5$ ($N = 256$). Blue: FD5 without post-processing. Colored markers: TV post-processing with $\lambda = 0.02, 0.05, 0.10$. The FD5 scheme is stable for this test case. TV post-processing removes the remaining oscillations near the shock.

Figure 6 compares the convergence of the three TV solvers on the post-processing task with $\lambda = 0.02$. The DI algorithm converges in 244 breakpoint steps. The step count is comparable to the hardest spectral denoising test (496 steps at $N = 1000$) and reflects the richer structure of the FD5 Burgers solution: the oscillatory signal near the shock produces many nonzero differences, so the equicorrelation set grows to a large fraction of the $n - 1 = 255$ possible indices. ADMM with tuned $\eta = 10/\lambda$ converges to $\sim 10^{-11}$ within 500 iterations. PDHG reaches $\sim 10^{-7}$. Untuned ADMM stalls near 10^{-3}.



(a) Objective error vs. iteration number.

(b) Objective error vs. CPU time.

Figure 6: Convergence comparison for TV post-processing of the FD5 Burgers solution ($n = 256$, $\lambda = 0.02$). The DI algorithm converges in 244 steps, comparable to the hardest spectral case (496 steps at $N = 1000$).

4.2. Tests on compressible Euler equations

Consider the 1D compressible Euler equations:

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2 + p\\ (E+p)\,u\end{pmatrix} = 0, \tag{6}$$

where ρ is density, u is velocity, p is pressure, and $E = p/(\gamma - 1) + \frac{1}{2}\rho u^2$ is the total energy with $\gamma = 1.4$.

We discretize (6) with the same fifth-order linear upwind FD5 scheme from Section 4.1, applied componentwise with Lax–Friedrichs flux splitting, integrated in time with the standard third-order SSP Runge–Kutta method. The resulting scheme is locally conservative and fifth-order accurate in smooth regions, but oscillatory near discontinuities. To ensure that density and pressure remain positive, one can augment the scheme with the Zhang–Shu positivity-preserving (PP) limiter for finite difference schemes [13], which ensures that the SSP Runge–Kutta update preserves the admissible set $G = \{(\rho, \rho u, E)^T : \rho > 0,\ p > 0\}$. The PP limiter guarantees positivity but does not suppress oscillations. We use an SSP scheme here, rather than the RK4 of Section 4.1, because the Zhang–Shu positivity-preserving limiter requires the time stepping to be a convex combination of forward-Euler steps. To suppress the oscillations, the TV solver can be used as a *post-processing* step to the primitive variables (ρ, u, p) after time integration is complete. Because the limiter is applied to the primitive variables, it preserves the discrete mean of each primitive but not the conserved momentum ρu or total

energy E; conservation in the sense of Section 2 holds only for the quantity actually denoised. For the post-processing tests here this is acceptable, but a conservative variant would denoise the conserved variables directly.

Lax shock tube. Figure 7 shows results for the Lax shock tube [43] ($N = 512$, $T = 0.13$). The unprocessed FD5+PP solution (blue circles) has $L^2(\rho) = 1.195 \times 10^{-1}$ with visible oscillations near the contact discontinuity and shock. Because TV denoising acts on each variable independently, different regularization strengths can target different wave features: the density profile contains the contact discontinuity in addition to the shock, so moderate smoothing is used to avoid flattening the contact, whereas velocity and pressure are continuous across the contact and tolerate stronger regularization to remove their oscillations. With per-variable parameters $\lambda_\rho = 0.2$, $\lambda_u = 1.0$, $\lambda_p = 1.0$ (red plus signs), TV post-processing removes oscillations and reduces the density L^2 error by 4.4% (1.195×10^{-1} to 1.143×10^{-1}).

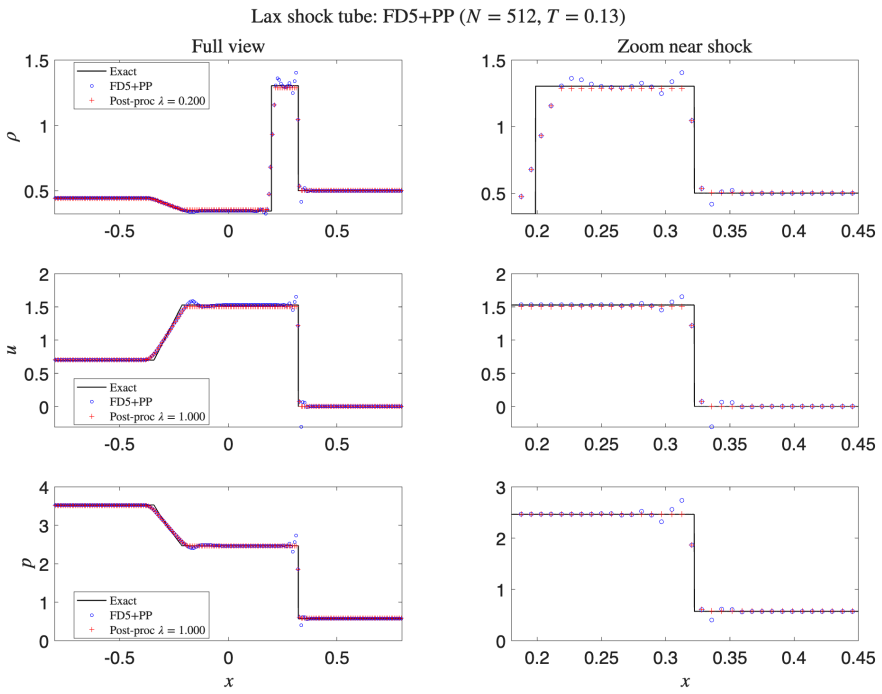


Figure 7: Lax shock tube with FD5+PP base scheme ($N = 512$, $T = 0.13$). Left: full view. Right: zoom near shock. Blue circles: FD5+PP numerical solution. Red plus signs: TV post-processed primitives ($\lambda_\rho = 0.2$, $\lambda_u = 1.0$, $\lambda_p = 1.0$). Black line: exact Riemann solution.

4.3. Rigid body rotation: LeVeque three-body test

We next solve the rigid body rotation problem

$$u_t - y\,u_x + x\,u_y = 0, \qquad (x, y) \in [-\pi, \pi)^2, \quad T = 2\pi,$$

following Example 4.7 of Liu, Cheng, and Shu [20]. The velocity field $\mathbf{v} = (-y, x)$ is divergence-free and produces solid-body rotation about the origin with period 2π. The initial condition is the LeVeque three-body test [44]: a cosine bell centered at $(-\pi/2, 0)$, a cone centered at $(0, -\pi/2)$, and a slotted disk centered at $(0, \pi/2)$, all with radius $r_0 = 0.3\pi$. The exact solution at $T = 2\pi$ equals the initial condition.

We discretize on a 128×128 grid using the Fourier pseudospectral method with Orszag's $\frac{2}{3}$ dealiasing rule [45] and RK4 time-stepping ($\Delta t = 2/(\pi N) \approx 4.97 \times 10^{-3}$). Nonlinear products transfer energy to higher wavenumbers than are present in the inputs; any content above the grid's Nyquist cutoff $|k| = N/2$ wraps around ("aliases") and pollutes the low-wavenumber spectrum. The $\frac{2}{3}$ rule restricts the solution to modes with $|k| < N/3$ (cutoff at $\frac{2}{3}$ of the Nyquist wavenumber $N/2$), zeroing the remaining modes before and after each nonlinear step. The product of two such inputs has spectrum in $|k| < 2N/3$, and any aliased content folds into $|k| \geq N/3$, which the filter discards. We compare the methods in Table 3. The PDE solver is implemented in MATLAB, and the reported TV time is from a MATLAB wrapper calling the C++ DI solver.

Method	λ	L^1 error	min	max	DI steps	TV calls	TV (s)	PDE (s)
No limiter	—	1.72e-2	−0.202	1.257	—	—	—	3.5
TV limiter (K=10)	0.02	1.94e-2	−0.025	0.983	76753	12	0.35	3.5
TV post-proc	0.05	6.81e-3	−0.012	1.057	5414	1	0.02	3.5
TV post-proc	0.10	9.14e-3	0.001	1.000	4457	1	0.01	3.5

Table 3: Rigid body rotation (128×128, $T = 2\pi$). Exact solution satisfies $u \in [0, 1]$. DI steps: total breakpoint iterations across all 1D solves. TV/PDE: wall-clock time (seconds) for TV operations and PDE solver. Post-processing uses the no-limiter PDE solution.

Without any limiter, the spectral solution overshoots by 25.7% and undershoots by 20.2%. The per-step TV limiter reduces bound violations to $[-0.025, 0.983]$ with only 12 TV calls over 1264 time steps. TV post-processing at $\lambda = 0.10$ achieves near-perfect bounds $[0.001, 1.000]$ with the smallest undershooting. The best L^1 accuracy (6.81×10^{-3}) is obtained by post-processing with $\lambda = 0.05$, which balances oscillation removal against

over-smoothing. We note that the per-step limiter slightly increases the L^1 error relative to the no-limiter solution (from 1.72×10^{-2} to 1.94×10^{-2}): repeated denoising accumulates smoothing error, which is the price paid for controlling the oscillations during the computation. When only the final solution is needed, post-processing avoids this accumulation. The full solutions and the corresponding cross-sectional cuts are shown in Figures 8 and 9.

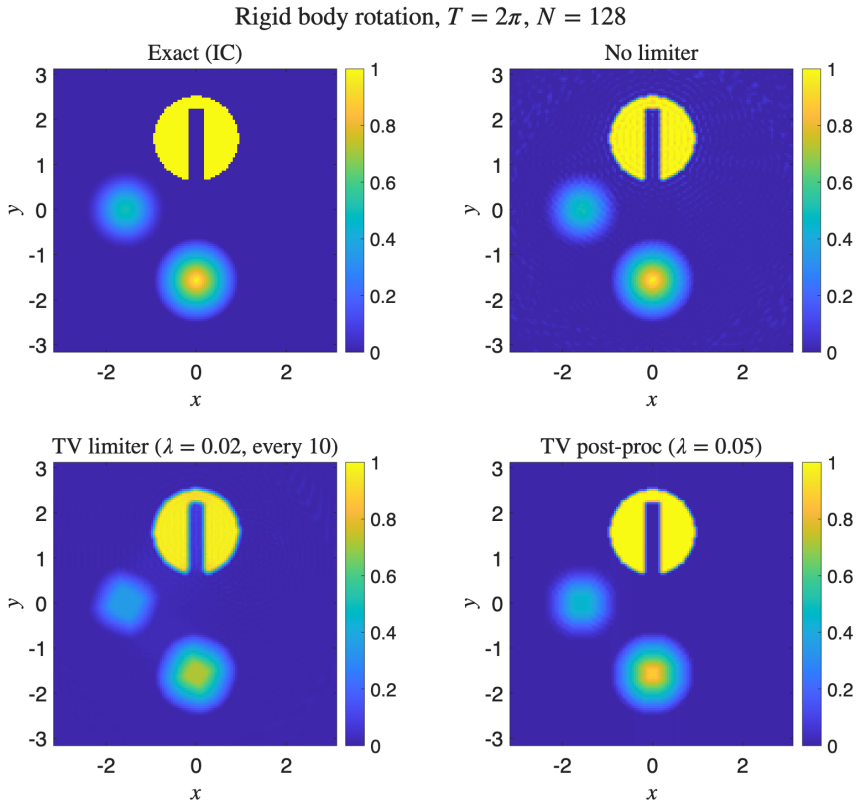


Figure 8: Rigid body rotation at $T = 2\pi$ (128×128). Top left: exact (= IC). Top right: spectral (no limiter). Bottom left: TV limiter (K=10, λ=0.02). Bottom right: TV post-processing (λ=0.05).

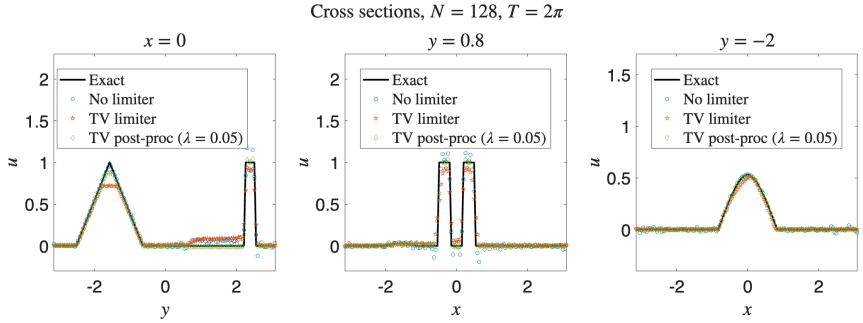


Figure 9: Cross-sectional cuts for the rotation test. Left: $x = 0$ (through cone and slotted disk). Center: $y = 0.8$ (through slotted disk). Right: $y = -2$ (near the cosine bell). Solid black line: exact solution. Circles (blue): no limiter. Stars (orange): TV limiter (K=10). Diamonds (green): TV post-processing (λ=0.05). Both TV approaches significantly reduce the Gibbs oscillations visible in the no-limiter solution, with post-processing providing the best L^1 accuracy.

4.4. Swirling deformation flow

The swirling deformation problem has a time-dependent, spatially varying velocity field (Example 4.8 of [20]):

$$u_t - \left(\cos^2\!\left(\tfrac{x}{2}\right)\sin(y)\,g(t)\,u\right)_x + \left(\sin(x)\cos^2\!\left(\tfrac{y}{2}\right)g(t)\,u\right)_y = 0,$$

where $g(t) = \pi\cos(\pi t/T)$ and $T = 1.5$. The velocity field is divergence-free. It deforms the solution into thin filaments during $0 < t < T/2$, then reverses, so the exact solution at $T = 1.5$ equals the initial condition. The IC is the same LeVeque three-body test as in Section 4.3.

On a 128×128 grid, the TV limiter reduces bound violations from $[-0.19, 1.25]$ to $[-0.035, 1.008]$ with 13 TV calls over 302 time steps. Post-processing at $\lambda = 0.05$ gives the best L^1 accuracy (6.20×10^{-3}), while $\lambda = 0.10$ achieves near-perfect bounds $[0.001, 0.994]$. As in Section 4.3, the per-step limiter trades some L^1 accuracy for improved bounds. The corresponding cross-sectional cuts through the three bodies are shown in Figure 10.

Method	L^1 error	min	max	TV calls
No limiter	1.03×10^{-2}	-0.188	1.249	—
TV limiter (K=10, λ=0.02)	2.21×10^{-2}	-0.035	1.008	13
TV post-proc (λ=0.05)	6.20×10^{-3}	-0.010	1.049	1
TV post-proc (λ=0.10)	8.65×10^{-3}	0.001	0.994	1

Table 4: Swirling deformation flow (128×128, $T = 1.5$). Exact solution satisfies $u \in [0, 1]$.

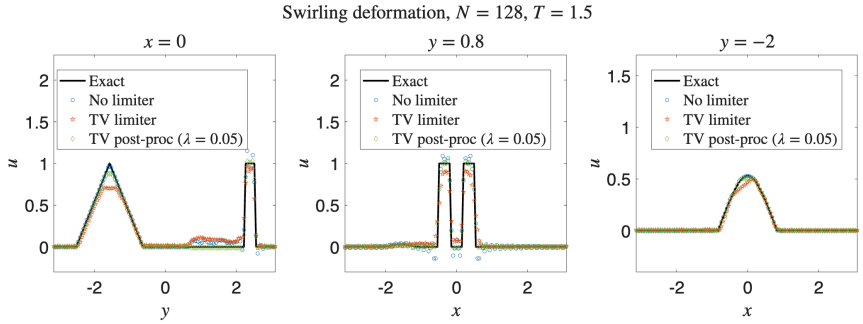


Figure 10: Cross-sectional cuts for the swirling deformation test. Same cuts and marker conventions as Figure 9. The TV limiter and post-processing effectively suppress oscillations despite the time-dependent, spatially varying velocity field.

4.5. Incompressible Euler: vortex patches

We finally consider the incompressible 2D Euler equations in vorticity form (Example 4.10 of [20]):

$$\omega_t + u\,\omega_x + v\,\omega_y = 0, \qquad \Delta\psi = \omega, \qquad (u, v) = (-\psi_y, \psi_x),$$

on $[0, 2\pi)^2$ with periodic boundary conditions. The initial vorticity consists of two rectangular patches: $\omega_0 = -1$ on $[\pi/2, 3\pi/2] \times [\pi/4, 3\pi/4]$, $\omega_0 = 1$ on $[\pi/2, 3\pi/2] \times [5\pi/4, 7\pi/4]$, and $\omega_0 = 0$ elsewhere. The maximum principle guarantees $\omega \in [-1, 1]$ at all times.

We discretize on a 128×128 grid using the Fourier pseudospectral method ($\frac{2}{3}$ dealiasing, RK4) and integrate to $T = 5$. The velocity is recovered at each stage by solving the Poisson equation $\Delta\psi = \omega$ in Fourier space. Table 5 summarizes the results. The PDE solver is implemented in MATLAB, and the reported TV time is from a MATLAB wrapper calling the C++ DI solver.

Without any limiter, the spectral solution violates the maximum principle by 30%. The per-step TV limiter reduces the range to $[-0.945, 0.945]$, well

Method	λ	min	max	DI steps	TV calls	TV (s)	PDE (s)
No limiter	—	-1.299	1.299	—	—	—	1.9
TV limiter (K=10)	0.02	-0.945	0.945	265119	41	0.95	1.7
TV post-proc	0.10	-1.053	1.053	3631	1	0.01	1.9

Table 5: Incompressible Euler (128×128, $T = 5$). Maximum principle: $\omega \in [-1, 1]$. DI steps: total breakpoint iterations across all 1D solves. TV/PDE: wall-clock time (seconds) for TV operations and PDE solver. Post-processing uses the no-limiter PDE solution.

within the physical bounds, with 41 TV calls over 435 time steps. TV post-processing at $\lambda = 0.10$ reduces the violation to about 5%, but cannot fully recover the bounds from the accumulated nonlinear oscillations. The vorticity fields and the corresponding cross-sectional cuts are shown in Figures 11 and 12.

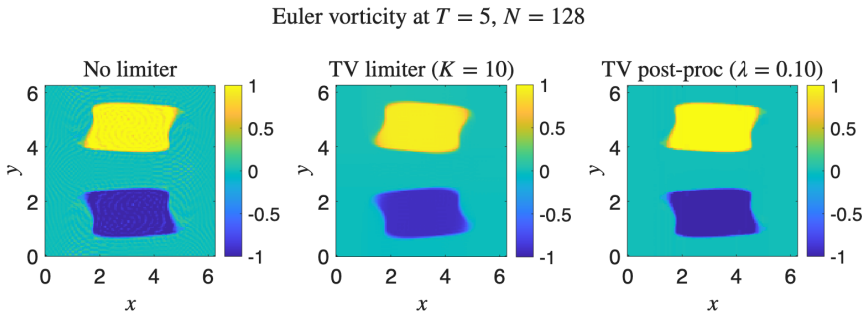


Figure 11: Vorticity fields at $T = 5$ (128×128). Left: no limiter. Center: TV limiter (K=10, λ=0.02). Right: TV post-processing (λ=0.10).

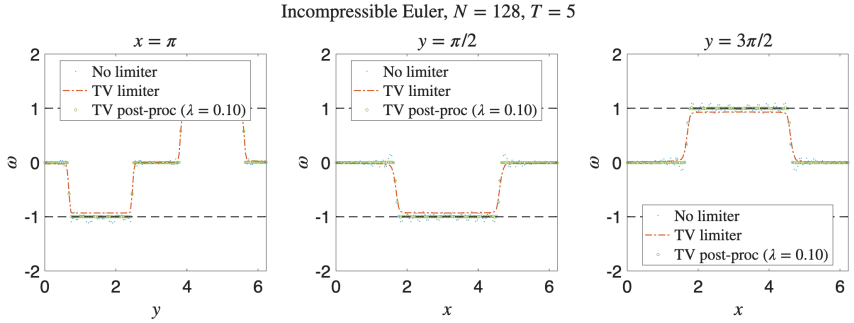


Figure 12: Cross-sectional cuts for incompressible Euler at $T = 5$. Left: $x = \pi$. Center: $y = \pi/2$. Right: $y = 3\pi/2$. Dots (blue): no limiter. Dash-dot (orange): TV limiter (K=10). Circles (green): TV post-processing (λ=0.10). Dashed lines mark the maximum principle bounds $\omega = \pm 1$.

4.6. 2D Riemann problem with FD5 scheme

We also solve a genuine two-dimensional Riemann problem for the compressible Euler equations using the fifth-order linear finite difference scheme (FD5) from Section 4.1, extended to two dimensions with Lax–Friedrichs splitting applied dimension by dimension. We use Configuration 6 of Lax and Liu [46], in which the initial data consist of four constant states meeting at the center of the domain $[0, 1]^2$:

$$(\rho, u, v, p) = \begin{cases} (2, 0.75, 0.5, 1) & x < 0.5,\ y \geq 0.5, \\ (1, 0.75, -0.5, 1) & x \geq 0.5,\ y \geq 0.5, \\ (1, -0.75, 0.5, 1) & x < 0.5,\ y < 0.5, \\ (3, -0.75, -0.5, 1) & x \geq 0.5,\ y < 0.5. \end{cases}$$

We compute a reference solution with the WENO5 scheme [11] on an 800×800 grid with SSP-RK3 time stepping.

Both the FD5 and WENO5 schemes use zero-order extrapolation (outflow) boundary conditions, following [46]. We discretize on a 200×200 grid with RK4 time stepping (CFL $= 0.5$) and integrate to $T = 0.2$. Figure 13 compares the density fields. The WENO5 reference (left) resolves sharp shock and contact structures, while the FD5 linear scheme (center) captures the wave structure but produces oscillations near all discontinuities. For TV post-processing (right), we use a per-slice regularization parameter. Since the 2D TV solver operates dimension by dimension (columns then rows), each 1D slice receives its own λ. Slices passing through the center of the

domain ($x \in [0.25, 0.75]$ for columns, $y \in [0.25, 0.75]$ for rows) use $\lambda = 0.02$, preserving the complex but not strongly oscillatory wave interactions. Slices in the corner regions use $\lambda = 0.3$, providing stronger smoothing where the solution is nearly constant. TV post-processing reduces the oscillations near these discontinuities in the cross-sectional cuts through $y = 0.10$, $y = 0.5$, and $x = 0.5$. See Figure 14.

The PDE solver requires 358 time steps (12.3 s). TV post-processing of the density field takes 19107 DI steps (0.46 s), at a fraction of the PDE solver cost.

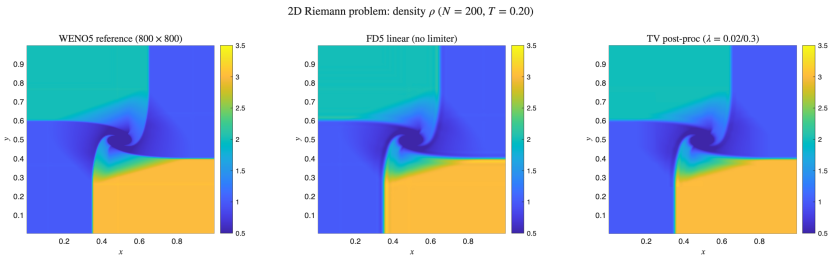


Figure 13: Density ρ for the 2D Riemann problem (Config. 6, 200×200, $T = 0.2$). Left: WENO5 reference (800×800). Center: FD5 linear (no limiter). Right: TV post-processing ($\lambda = 0.02$ center, 0.3 corners).

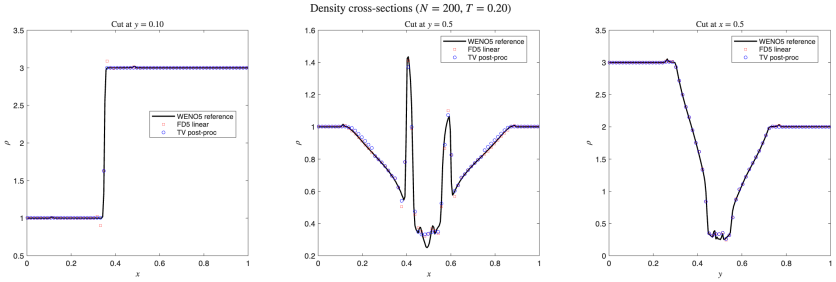


Figure 14: Density cross-sections for the 2D Riemann problem. Left: cut at $y = 0.10$. Center: cut at $y = 0.5$. Right: cut at $x = 0.5$. Black solid: WENO5 reference (800×800). Red squares: FD5 linear. Blue circles: TV post-processing ($\lambda = 0.02/0.3$).

5. Concluding remarks

In this work, we applied the DI algorithm to the one-dimensional TV denoising problem and used it as a non-oscillatory limiter for high-order

methods for conservation laws. The key idea is that the difference operator $D \in \mathbb{R}^{(n-1)\times n}$ is surjective, so the change of variables $\mathbf{z} = D\tilde{\mathbf{x}}$ reduces the TV problem to an equivalent LASSO problem, which the DI algorithm can solve exactly in finitely many steps. As a result, we obtain a 1D solver that computes the exact TV minimizer and preserves the total mass of the input. The only free parameter is λ, which directly controls the regularization strength. In two dimensions, we use a dimension-by-dimension splitting. We have tested the TV limiter on Fourier pseudospectral methods and a fifth-order finite difference scheme solving 1D and 2D problems.

Appendix A. Comparison of analysis operators

The denoising framework generalizes to any analysis operator D:

$$\min_{\mathbf{x}} \ \|D\mathbf{x}\|_1 + \frac{1}{2\lambda}\|\mathbf{x} - \mathbf{b}\|_2^2. \tag{A.1}$$

The TV problem studied in this paper corresponds to choosing D as the forward difference operator. As observed by Cai et al. [47], TV denoising is in fact the simplest member of a family of B-spline framelet models. The Haar framelet (Example 2.1 of [47]), constructed from the piecewise constant B-spline B_1, has a single detail mask $\mathbf{a}_1 = \frac{1}{2}[1, -1]$, which is the forward difference operator up to a factor of $1/2$. Consequently, the Haar framelet analysis-based approach and the Rudin–Osher–Fatemi (ROF) model [21] coincide under forward difference discretization [47, eq. (2.3)]. Higher-order B-spline framelets add detail bands that approximate higher-order difference operators, yielding natural generalizations of TV.

The piecewise linear framelet (Example 2.2 of [47]), constructed from the B-spline B_2, has two detail bands: $\mathbf{a}_1$ approximates a first-order difference and $\mathbf{a}_2$ a second-order difference ($2n \times n$ tight frame). The piecewise cubic framelet (Example 2.3 of [47]), constructed from B_4, has four detail bands approximating first- through fourth-order differences ($4n \times n$ tight frame). In all cases, the analysis operator D is formed by stacking the detail-band circulant matrices (excluding the low-pass band), and its null space is one-dimensional (the low-pass mode).

In addition, we compare against multi-level orthogonal discrete wavelet transforms (DWT). For an $n \times n$ orthogonal matrix W (e.g. Daubechies, Coiflet, or Symlet), the substitution $\mathbf{z} = W\mathbf{x}$ decouples the problem into componentwise soft thresholding: $\mathbf{z}^* = S_\lambda(W\mathbf{b})$, $\mathbf{x}^* = W^T\mathbf{z}^*$. Equivalently,

the DI algorithm applied to the LASSO (3) with measurement matrix $M = I$ and data $W\mathbf{b}$ in place of $\tilde{\mathbf{b}}$ recovers the same answer in a small number of DI steps (2–30).

The LASSO reduction of §2.2 is exact only when D is surjective (the forward-difference operator) or square and invertible (the orthogonal DWTs). For the redundant B-spline framelets (D tall, $m > n$), setting $M = D^\dagger$ and minimizing $\|\mathbf{z}\|_1 + \frac{1}{2\lambda}\|M\mathbf{z} - \tilde{\mathbf{b}}\|_2^2$ over $\mathbf{z} \in \mathbb{R}^m$ solves the *synthesis* formulation; the recovered $\mathbf{x}^* = M\mathbf{z}^*$ minimizes a relaxation of the analysis problem (A.1), not (A.1) itself, since the minimizer need not satisfy $\mathbf{z}^* = D\mathbf{x}^*$. We report the synthesis solutions below; they coincide with the analysis minimizer only in the surjective and orthogonal cases. We test the following operators on $n = 256$ grid points using the same two Gibbs oscillation test signals from §3.3.2 and §3.3.3, with $\lambda = 0.5$ for the piecewise constant signal and $\lambda = 0.1$ for the piecewise smooth signal:

1. *Forward difference* (TV / Haar framelet): $(n-1) \times n$, single detail band (first-order difference).
2. *Multi-level orthogonal DWT*: $n \times n$, db1 (Haar) and db4 (Daubechies-4).
3. *Piecewise linear framelet*: $2n \times n$, two detail bands (first- and second-order differences).
4. *Piecewise cubic framelet*: $4n \times n$, four detail bands (first- through fourth-order differences).

In every case the DI algorithm solves its LASSO (3) exactly after building $M = D^\dagger$ offline (or $M = I$ for the orthogonal DWT case); for the redundant framelets this LASSO is the synthesis relaxation rather than the analysis problem (A.1), as noted above.

Table A.6 reports the number of DI steps and L^2 error for each operator. Figures A.15 and A.16 show the denoised signals for all operators.

For the piecewise constant signal ($\lambda = 0.5$), the multi-level Haar DWT (db1) achieves the lowest L^2 error (0.058), ahead of TV (0.089) and all other operators. However, the smaller L^2 error does not fully reflect the solution quality. The db1 solution develops staircase artifacts near the jumps (see Figure A.15) whereas TV preserves sharper transitions. The L^2 metric is sensitive to the single grid point where TV passes through zero, but it does not capture this difference well. The longer-filter db4 wavelet performs worse ($L^2 = 0.17$) because its wider support smears the jumps.

For the piecewise smooth signal ($\lambda = 0.1$), all operators give comparable L^2 errors (0.063–0.068), with TV slightly smaller. Figure A.16 shows that

Table A.6: Comparison of analysis operators ($n = 256$). The DI algorithm solves the associated LASSO exactly (the synthesis form for the redundant framelets). Entries are piecewise constant ($\lambda = 0.5$) / piecewise smooth ($\lambda = 0.1$).

Operator	Type	DI steps	$\|\mathbf{x}^* - \mathbf{x}_{\text{true}}\|_2$
Forward diff. (TV)	Haar framelet	3 / 139	0.089 / 0.063
db1 (multi-level Haar)	$n \times n$ orth. DWT	2 / 30	0.058 / 0.066
db4	$n \times n$ orth. DWT	11 / 24	0.167 / 0.068
PL framelet	$2n \times n$ tight frame	3 / 148	0.091 / 0.064
PC framelet	$4n \times n$ tight frame	1 / 773	0.089 / 0.063

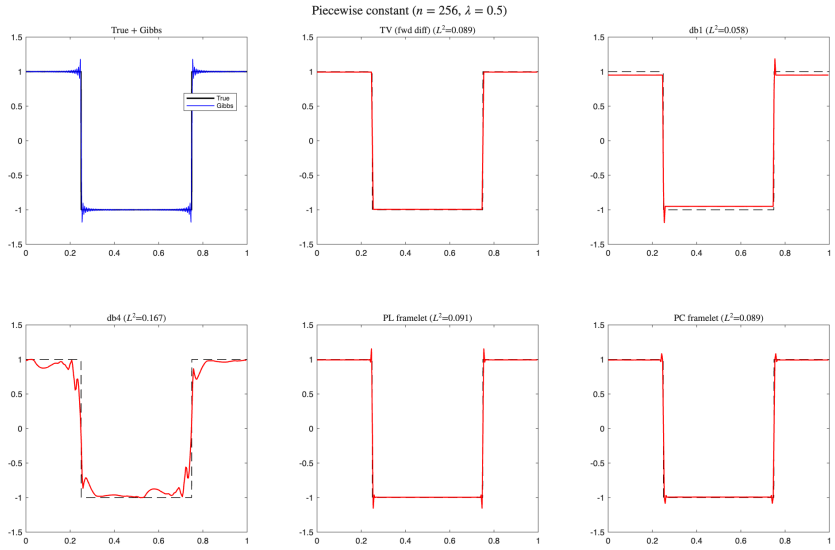


Figure A.15: Denoised piecewise constant signal ($n = 256$, $\lambda = 0.5$). TV and non-TV operators are shown.

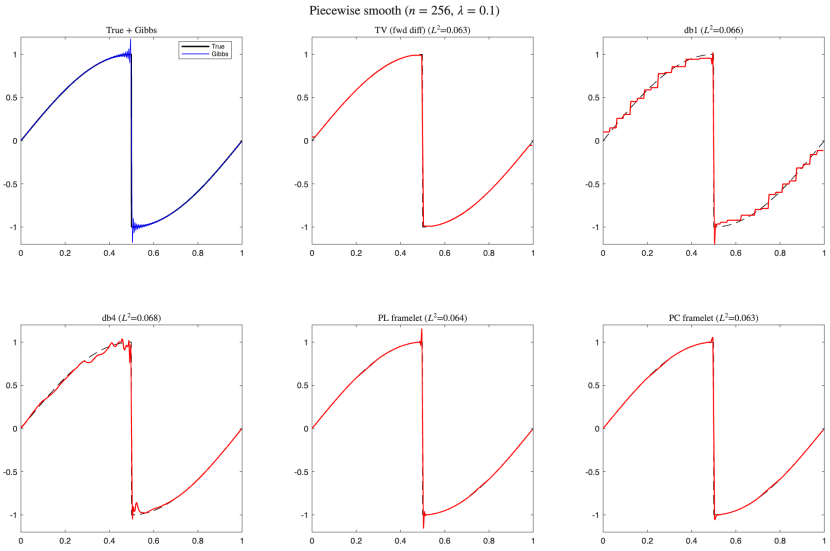


Figure A.16: Denoised piecewise smooth signal ($n = 256$, $\lambda = 0.1$). TV and non-TV operators are shown.

TV and the B-spline framelets preserve the smooth arcs and the jump more accurately, while the multi-level wavelets introduce oscillations or excessive rounding near the discontinuity.

These numerical findings are consistent with the theory of the Gibbs phenomenon in wavelet expansions. Kelly [48] proved that the Haar wavelet does *not* exhibit the Gibbs phenomenon, via an if-and-only-if criterion on its reproducing kernel $K_0(x, y) = \sum_k \varphi(x - k)\varphi(y - k)$: a Gibbs effect occurs near a jump if and only if $\int_0^\infty K_0(a, u)\, du > 1$ for some $a > 0$. For the Haar system this integral equals 1, and the kernel is nonnegative—analogous to the Fejér kernel—so the expansion converges without overshoot at jump discontinuities. Shim and Volkmer [49] showed, conversely, that if the scaling function φ is continuous, is differentiable at some dyadic rational d with $\varphi'(d) \neq 0$, and decays as $|\varphi(x)| \leq C(1 + |x|)^{-\beta}$ with $\beta > 3$, then the wavelet expansion exhibits a Gibbs phenomenon on at least one side of a jump. This covers the C^1 compactly supported Daubechies wavelets (db$N$, $N \geq 3$) and the C^1 Symlets and Coiflets (see [50] for a detailed exposition). Since the forward difference operator in TV denoising is the detail mask of the Haar framelet (up to a factor of $1/2$), TV is expected to inherit the Gibbs-free

behavior of the Haar system; this is consistent with its superior performance for post-processing discontinuous solutions. We note, however, that this connection is heuristic.

The DI algorithm applies unchanged to any analysis operator D after forming $M = D^\dagger$ offline. For the $n \times n$ orthogonal wavelets, the problem reduces to componentwise soft thresholding, so there is no coupling between coefficients. This limits their effectiveness compared with the rectangular operators ($m \neq n$), where the DI algorithm exploits dependencies in the analysis domain. Among the operators tested here, TV gives the most accurate overall reconstruction of the Gibbs-oscillatory signals. It preserves sharp jumps and retains the smooth parts of the signal well.

Funding

JD was supported in part by the Center for Information Geometric Mechanics and Optimization (CIGMO), a PSAAP-IV Focused Investigatory Center funded by the U.S. Department of Energy, National Nuclear Security Administration under Award Number DE-NA0004261. R. Lai is partially supported by NSF grant DMS-2401297. C.-W. Shu is partially supported by NSF grant DMS-2309249. X. Zhang is partially supported by NSF grant DMS-2208518.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data and code are available from the authors upon request.

CRediT authorship contribution statement

Gabriel P. Langlois: Methodology, Software, Investigation, Writing – review & editing. **Jérôme Darbon:** Conceptualization, Methodology, Software, Writing – review & editing, Funding acquisition. **Rongjie Lai:** Conceptualization, Methodology, Funding acquisition. **Chi-Wang Shu:**

Conceptualization, Methodology, Funding acquisition. **Xiangxiong Zhang:** Conceptualization, Methodology, Software, Investigation, Writing – original draft, Writing – review & editing, Supervision, Funding acquisition.

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process

During the preparation of this work, the authors used Claude Code (Anthropic) in order to assist with mathematical discussions, numerical implementation, and drafting of the manuscript. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

References

[1] D. Gottlieb, S. A. Orszag, Numerical Analysis of Spectral Methods: Theory and Applications, SIAM, Philadelphia, 1977.

[2] J. S. Hesthaven, S. Gottlieb, D. Gottlieb, Spectral methods for time-dependent problems, Cambridge University Press, Cambridge, 2007.

[3] C.-W. Shu, Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws, in: A. Quarteroni (Ed.), Advanced Numerical Approximation of Nonlinear Hyperbolic Equations, Vol. 1697 of Lecture Notes in Mathematics, Springer, 1998, pp. 325–432.

[4] C.-W. Shu, High order weighted essentially nonoscillatory schemes for convection dominated problems, SIAM Rev. 51 (1) (2009) 82–126.

[5] B. Cockburn, G. E. Karniadakis, C.-W. Shu (Eds.), Discontinuous Galerkin Methods: Theory, Computation and Applications, Vol. 11 of Lecture Notes in Computational Science and Engineering, Springer, Berlin, Heidelberg, 2000.

[6] J. S. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications, Springer, New York, 2008.

[7] H. Vandeven, Family of spectral filters for discontinuous problems, J. Sci. Comput. 6 (2) (1991) 159–192.

[8] E. Tadmor, Convergence of spectral methods for nonlinear conservation laws, SIAM J. Numer. Anal. 26 (1) (1989) 30–44.

[9] D. Gottlieb, C.-W. Shu, On the Gibbs phenomenon and its resolution, SIAM Rev. 39 (4) (1997) 644–668.

[10] A. Harten, High resolution schemes for hyperbolic conservation laws, J. Comput. Phys. 49 (3) (1983) 357–393.

[11] G.-S. Jiang, C.-W. Shu, Efficient implementation of weighted ENO schemes, J. Comput. Phys. 126 (1) (1996) 202–228.

[12] X. Zhang, C.-W. Shu, On positivity-preserving high order discontinuous Galerkin schemes for compressible Euler equations on rectangular meshes, J. Comput. Phys. 229 (23) (2010) 8918–8934.

[13] X. Zhang, C.-W. Shu, Positivity-preserving high order finite difference WENO schemes for compressible Euler equations, J. Comput. Phys. 231 (5) (2012) 2245–2258.

[14] J.-L. Guermond, M. Nazarov, B. Popov, I. Tomas, Second-order invariant domain preserving approximation of the Euler equations using convex limiting, SIAM J. Sci. Comput. 40 (5) (2018) A3211–A3239.

[15] J. P. Boris, D. L. Book, Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works, J. Comput. Phys. 11 (1) (1973) 38–69.

[16] S. T. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, J. Comput. Phys. 31 (3) (1979) 335–362.

[17] O. Guba, M. Taylor, A. St-Cyr, Optimization-based limiters for the spectral element method, J. Comput. Phys. 267 (2014) 176–195.

[18] P. Bochev, D. Ridzal, G. Scovazzi, M. Shashkov, Constrained-optimization based data transfer, in: D. Kuzmin, R. Löhner, S. Turek (Eds.), Flux-Corrected Transport: Principles, Algorithms, and Applications, Springer Netherlands, Dordrecht, 2012, pp. 345–398.

[19] K. Wu, X. Zhang, C.-W. Shu, High order numerical methods preserving invariant domain for hyperbolic and related systems, to appear in SIAM Review. arXiv:2512.09116.

[20] Y. Liu, Y. Cheng, C.-W. Shu, A simple bound-preserving sweeping technique for conservative numerical approximations, J. Sci. Comput. 73 (2–3) (2017) 1028–1071.

[21] L. I. Rudin, S. Osher, E. Fatemi, Nonlinear total variation based noise removal algorithms, Physica D 60 (1–4) (1992) 259–268.

[22] A. Chambolle, An algorithm for total variation minimization and applications, J. Math. Imaging Vis. 20 (1–2) (2004) 89–97.

[23] G. P. Langlois, J. Darbon, A fast algorithm for solving the lasso problem exactly without homotopy using differential inclusions, arXiv preprint arXiv:2507.05562 (2025).

[24] A. Chambolle, J. Darbon, On total variation minimization and surface evolution using parametric maximum flows, Int. J. Comput. Vis. 84 (3) (2009) 288–307.

[25] D. S. Hochbaum, An efficient algorithm for image segmentation, Markov random fields and related problems, J. ACM 48 (4) (2001) 686–701.

[26] P. L. Davies, A. Kovac, Local extremes, runs, strings and multiresolution, Ann. Statist. 29 (1) (2001) 1–65.

[27] L. Condat, A direct algorithm for 1-D total variation denoising, IEEE Signal Process. Lett. 20 (11) (2013) 1054–1057.

[28] R. J. Tibshirani, J. Taylor, The solution path of the generalized lasso, Ann. Statist. 39 (3) (2011) 1335–1371.

[29] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani, Least angle regression, Ann. Statist. 32 (2) (2004) 407–499.

[30] J. Darbon, M. Sigelle, A fast and exact algorithm for total variation minimization, in: Iberian Conference on Pattern Recognition and Image Analysis, Springer, 2005, pp. 351–359.

[31] J. Darbon, Composants logiciels et algorithmes de minimisation exacte d'énergies dédiés au traitement des images, Ph.D. thesis, Télécom ParisTech (2005).

[32] J. Darbon, M. Sigelle, Image restoration with discrete constrained total variation Part I: Fast and exact optimization, Journal of Mathematical Imaging and Vision 26 (3) (2006) 261–276.

[33] J. Darbon, M. Sigelle, Image restoration with discrete constrained total variation Part II: Levelable functions, convex priors and non-convex cases, Journal of Mathematical Imaging and Vision 26 (3) (2006) 277–291.

[34] Y. Tendero, I. Ciril, J. Darbon, S. Serna, An algorithm solving compressive sensing problem based on maximal monotone operators, SIAM Journal on Scientific Computing 43 (6) (2021) A4067–A4094.

[35] A. Chambolle, T. Pock, A first-order primal-dual algorithm for convex problems with applications to imaging, J. Math. Imaging Vis. 40 (1) (2011) 120–145.

[36] P.-L. Lions, B. Mercier, Splitting algorithms for the sum of two nonlinear operators, SIAM J. Numer. Anal. 16 (6) (1979) 964–979.

[37] L. Demanet, X. Zhang, Eventual linear convergence of the Douglas–Rachford iteration for basis pursuit, Math. Comp. 85 (297) (2016) 209–238.

[38] E. Gil Torres, M. Jacobs, X. Zhang, Asymptotic linear convergence of ADMM for isotropic TV norm compressed sensing, arXiv preprint arXiv:2505.01240 (2025).

[39] A. M. Bradley, P. A. Bosler, O. Guba, M. A. Taylor, G. A. Barnett, Communication-efficient property preservation in tracer transport, SIAM J. Sci. Comput. 41 (3) (2019) C161–C193.

[40] C. Liu, D. Milesis, C.-W. Shu, X. Zhang, Efficient optimization-based invariant-domain-preserving limiters in solving gas dynamics equations, J. Comput. Phys. 558 (2026) 114839.

[41] R. Lai, X. Zhang, Ubiquitous Laplacian: An Introduction to Numerical PDEs with Applications in Data Science, Vol. 2 of Progress in Data Science, World Scientific, 2026. doi:10.1142/14349.

[42] M. C. Meyer, A simple new algorithm for quadratic programming with applications in statistics, Comm. Statist. Simulation Comput. 42 (5) (2013) 1126–1139.

[43] P. D. Lax, Weak solutions of nonlinear hyperbolic equations and their numerical computation, Comm. Pure Appl. Math. 7 (1) (1954) 159–193.

[44] R. J. LeVeque, High-resolution conservative algorithms for advection in incompressible flow, SIAM J. Numer. Anal. 33 (2) (1996) 627–665.

[45] S. A. Orszag, On the elimination of aliasing in finite-difference schemes by filtering high-wavenumber components, Journal of the Atmospheric Sciences 28 (6) (1971) 1074.

[46] P. D. Lax, X.-D. Liu, Solution of two-dimensional Riemann problems of gas dynamics by positive schemes, SIAM J. Sci. Comput. 19 (2) (1998) 319–340.

[47] J.-F. Cai, B. Dong, S. Osher, Z. Shen, Image restoration: total variation, wavelet frames, and beyond, J. Amer. Math. Soc. 25 (4) (2012) 1033–1089.

[48] S. E. Kelly, Gibbs phenomenon for wavelets, Appl. Comput. Harmon. Anal. 3 (1) (1996) 72–81.

[49] H.-T. Shim, H. Volkmer, On the Gibbs phenomenon for wavelet expansions, J. Approx. Theory 84 (1) (1996) 74–95.

[50] K. Raeen, A study of the Gibbs phenomenon in Fourier series and wavelets, Master's thesis, University of New Mexico (2008).