Signal Denoising via Optimization

Homework 1

Konstantinos Tsironis
ID: 1107529

October 2025

Part A — Mathematical Formulation

A.1 Tikhonov (quadratic) smoothing

Given a noisy signal $y \in \mathbb{R}^n$, we estimate $x \in \mathbb{R}^n$ by solving

$$\min_{x \in \mathbb{R}^n} \|x - y\|_2^2 + \lambda \|Dx\|_2^2, \qquad (Dx)_i = x_{i+1} - x_i, \tag{1}$$

where $D \in \mathbb{R}^{(n-1)\times n}$ is the forward-difference operator and $\lambda > 0$ balances fidelity and smoothness. Expanding the cost function $J(x) = \|x - y\|_2^2 + \lambda \|Dx\|_2^2$:

$$J(x) = (x - y)^\top (x - y) + \lambda (Dx)^\top (Dx) \tag{2}$$

$$= x^\top x - 2y^\top x + y^\top y + \lambda x^\top D^\top D x. \tag{3}$$

To find the minimum, we compute the gradient of $J(x)$ with respect to x. Using the standard vector calculus rules:

$$\nabla_x (x^\top x) = 2x, \quad \nabla_x (-2y^\top x) = -2y, \quad \nabla_x (x^\top D^\top D x) = 2D^\top D x.$$

Therefore,

$$\nabla_x J(x) = 2(x - y) + 2\lambda D^\top D x.$$

Taking the gradient and setting it to zero:

$$\nabla_x J(x) = 2(x - y) + 2\lambda D^\top D x = 0 \quad \Rightarrow \quad (I + \lambda D^\top D)\, x = y. \tag{4}$$

Thus, the optimal solution satisfies

$$(I + \lambda D^\top D)\, x = y, \quad \text{with closed-form } \; x^* = (I + \lambda D^\top D)^{-1} y. \tag{5}$$

A.2 Total Variation (TV) denoising

We solve

$$\min_{x\in\mathbb{R}^n} \|x-y\|_2^2 + \lambda \|Dx\|_1. \tag{6}$$

Let $z = Dx$ and introduce auxiliary variables $u \geq 0, v \geq 0$ such that $z = u - v$ and $\|z\|_1 = \mathbf{1}^\top(u+v)$. The problem becomes a convex Quadratic Program (QP):

$$\min_{x,u,v} \ \tfrac{1}{2}x^\top(2I)x - 2y^\top x + \lambda\,\mathbf{1}^\top(u+v) \tag{7}$$

$$\text{s.t.} \ \ Dx - u + v = 0, \tag{8}$$

$$u \geq 0, \ v \geq 0. \tag{9}$$

($y^\top y$ is a constant and omitted.) The $\|Dx\|_1$ term penalizes the number and size of jumps, thus preserving edges in piecewise-constant signals.

Effect of the L1 penalty compared to L2. The $L1$ (TV) regularization penalizes the *magnitude* of changes between consecutive samples rather than their squared value. As a result, it allows some differences (jumps) to remain exactly zero, producing segments of constant value separated by sharp edges. This means the reconstructed signal tends to be *piecewise constant*, preserving discontinuities and sharp transitions. In contrast, the $L2$ regularization distributes the penalty smoothly across all points, which blurs edges and makes the reconstruction globally smoother but less faithful near abrupt changes.

Part B — Implementation (Python)

B.1 Generate Data

```python
import numpy as np
n = 200
rng = np.random.default_rng(42)
t = np.linspace(0, 1, n)
# Example piecewise signal with jumps
x_true = np.piecewise(t, [t < 0.3, (t >= 0.3) & (t < 0.7), t >= 0.7],
                      [lambda s: 0.5*np.sin(8*np.pi*s),
                       lambda s: 1.0,
                       lambda s: 0.2*np.cos(6*np.pi*s)])
noise = 0.15 * rng.standard_normal(n)
y = x_true + noise
```

Listing 1: Data generation

B.2 Build the Difference Operator D

```python
import numpy as np

def diff_matrix(n: int) -> np.ndarray:
    D = np.zeros((n-1, n))
    for i in range(n-1):
        D[i, i]   = -1.0
        D[i, i+1] =  1.0
    return D

D = diff_matrix(n)
```

Listing 2: Forward-difference operator

B.3 L2 Smoothing (Tikhonov Regularization)

```python
import scipy.sparse as sp
import scipy.sparse.linalg as spla

lam = 1.0
DTD = (D.T @ D)
I = sp.eye(n, format='csc')
A = I + lam * DTD
x_l2 = spla.spsolve(A, y)
```

Listing 3: L2 smoothing

B.4 L1 / TV Regularization

```python
import cvxpy as cp

x = cp.Variable(n)
z = D @ x
lam = 0.5
obj = cp.Minimize(cp.sum_squares(x - y) + lam * cp.norm1(z))
prob = cp.Problem(obj)
prob.solve(solver=cp.OSQP, eps_abs=1e-6, eps_rel=1e-6, verbose=False)
x_tv = x.value
```

Listing 4: TV denoising

B.5 Evaluation and Visualization

```python
import matplotlib.pyplot as plt

def mse(a, b):
    return np.mean((a - b)**2)

mse_l2 = mse(x_l2, x_true)
mse_tv = mse(x_tv, x_true)
print("MSE L2:", mse_l2)
print("MSE TV:", mse_tv)

plt.figure(figsize=(8,4))
plt.plot(t, x_true, label='True')
plt.plot(t, y, label='Noisy', alpha=0.6)
plt.plot(t, x_l2, label='L2 (Tikhonov)')
plt.plot(t, x_tv, label='TV (L1)')
plt.legend(); plt.xlabel('t'); plt.ylabel('x')
plt.title('Signal Denoising Comparison')
plt.tight_layout(); plt.show()
```

Listing 5: Evaluation

Part C — Analysis and Discussion

C.1 Effect of λ

Experiments for $\lambda \in \{0.01, 0.1, 1, 10\}$ show that small λ yields high fidelity but poor denoising, while large λ over-smooths the signal. TV regularization tends to perform better for signals with edges, maintaining sharp transitions.

C.2 Signal Features

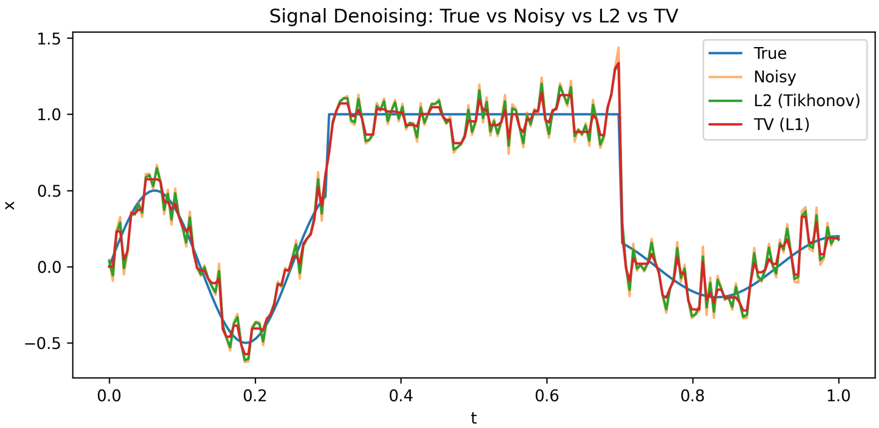
For piecewise-constant signals, TV denoising preserves discontinuities (edges) while L2 smoothing converts them into ramps. Example plots illustrate this behavior clearly.

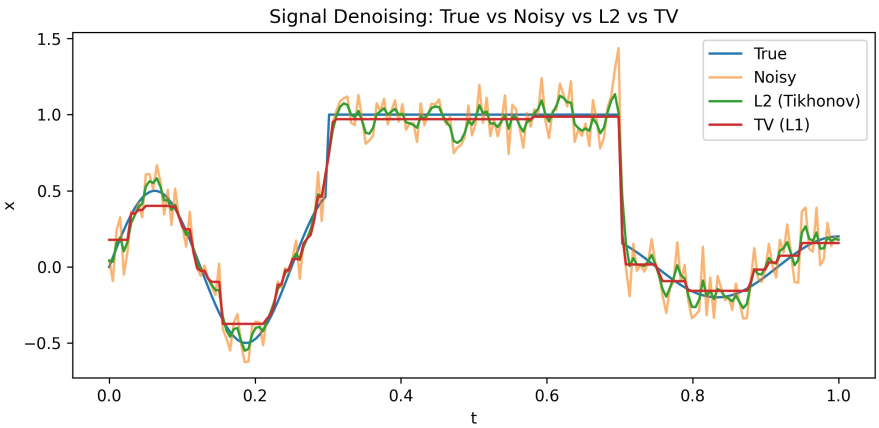
C.3 Noise Robustness

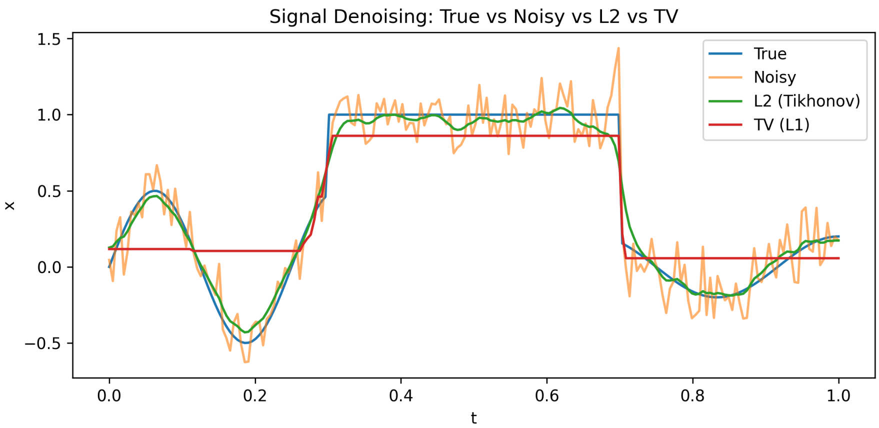
Increasing the noise level reveals that L2 behaves like a linear low-pass filter, while TV, due to its L1 penalty on derivatives, suppresses outliers more effectively and retains discontinuities.

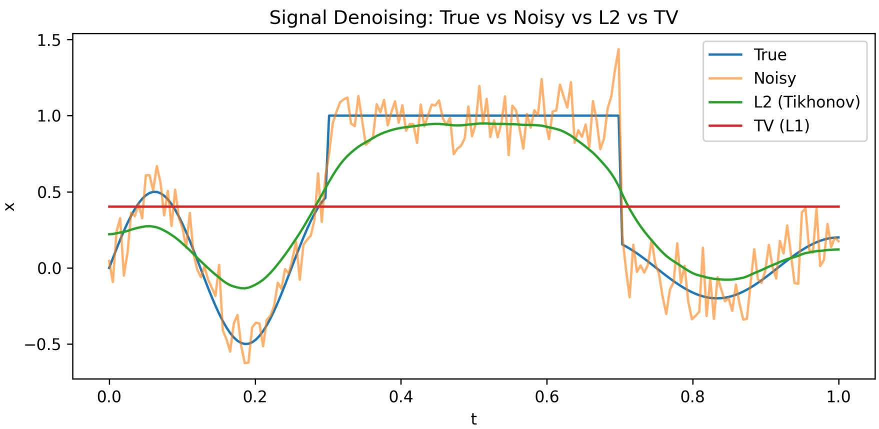
Plots

The observations discussed above are confirmed by the reconstructed signals shown in the plots on the following page, corresponding to $\lambda = 0.1,\ 1,\ 10$, and 100, respectively, which illustrate how increasing λ leads to progressively smoother reconstructions and reduced noise.


Signal Denoising: True vs Noisy vs L2 vs TV
True
Noisy
L2 (Tikhonov)
TV (L1)
x
t


Signal Denoising: True vs Noisy vs L2 vs TV
True
Noisy
L2 (Tikhonov)
TV (L1)
x
t


Signal Denoising: True vs Noisy vs L2 vs TV
True
Noisy
L2 (Tikhonov)
TV (L1)
x
t


Signal Denoising: True vs Noisy vs L2 vs TV
True
Noisy
L2 (Tikhonov)
TV (L1)
x
t