OSCILLATION REGULARIZATION

Steve Gu, Ying Zheng and Carlo Tomasi

Email: { steve, yuanqi, tomasi }@cs.duke.edu
Department of Computer Science
Duke University, Durham, NC USA 27708

ABSTRACT

We measure the degree of oscillation of a sampled function f by the number of its local extrema. The greater this number, the more oscillatory and complex f becomes. In signal denoising, we want a restored function g that is simple and fits the data f well. We propose to model this by a global optimization, coined oscillation regularization, that reduces both the data fitting error and the number of local extrema of g:

$$\tilde{g} = \arg\min_g \left\{ \underbrace{\text{err}(f,g)}_{\text{data fitting error}} + \lambda \cdot \underbrace{\#\text{ local extrema of } g}_{\text{oscillation}} \right\}$$

where $\text{err}(f,g)$ measures the discrepancy between f and g and λ is a regularization parameter. To the best of our knowledge, the number of local extrema of g is a topological prior that is rarely exploited in the literature of regularization.

We show that this global optimization can be done efficiently for one-dimensional signals. Specifically, we show that for sampled functions with values out of a discrete alphabet with V symbols defined on a chain of L nodes, the optimal solution can be found in $O(VL)$ time using dynamic programming. For functions with continuous ranges ($V \to +\infty$), we derive a polynomial time algorithm that depends only on L using linear programming.

Despite its simplicity in concept and algorithms, experiments show that oscillation regularization often yields state-of-art result. The availability of efficient algorithms also makes our method highly useful in other applications.

Index Terms— Oscillation, Regularization, Dynamic Programming, Global Optimization, Linear Programming

1. INTRODUCTION

Regularization constrains ill-posed problems which are commonly encountered in areas of signal processing, machine learning and computer vision. In this paper, we explore a new prior named *oscillation regularization* and apply it to the problem of one-dimensional signal denoising.

Throughout this paper, let $f : \Omega \to \mathcal{V}$ be a given sampled function where the domain $\Omega = \{1, 2, \cdots, L\}$ is a linear chain of L nodes and the range $\mathcal{V}$ can be either $\mathcal{V} = \{1, 2, \cdots, V\}$ or $\mathcal{V} = \mathbb{R}$. In both cases, the function f is equivalent to a vector in the space $\mathcal{V}^L$. Denoising restores a cleaner version $\tilde{g}$ from f, that attempts to remove the noise or measurement errors embedded in f.

1.1. Literature Review

The literature of denoising is rich and ranges from local methods [1, 2] to global methods [3, 4, 5, 6]. Local methods view denoising as a filtering problem and a local operator such as median, Gaussian or the bilateral filter[2] is applied to each position $1 \le l \le L$, with possible iterations. Compared to global methods, local methods are typically simple, fast, and effective, but may need parameter tuning and lack optimality guarantees in general.

Global methods utilize optimization tools which usually require intensive computation. For instances, many global methods can be cast as a regularization based optimization:

$$\tilde{g} = \arg\min_g \left\{ \underbrace{\text{err}(f,g)}_{\text{data fitting error}} + \lambda \cdot \underbrace{\text{Prior}(g)}_{\text{prior on } g} \right\} \tag{1}$$

where $\text{err}(f,g)$ measures the discrepancy between function f and g. For ease of discussion, throughout this paper we use:

$$\text{err}(f,g) \triangleq \|f - g\|_1 = \sum_{l=1}^{L} |f[l] - g[l]| \tag{2}$$

although other measures are possible without affecting the computational complexity of the algorithms described in this paper. For example, one useful variation is the truncation: $\text{err}(f,g) = \sum_{l=1}^{L} \min\{|f[l] - g[l]|, C\}$ where C is a constant that upper bounds the penalty induced by large deviations.

The term $\text{Prior}(g)$ encodes additional properties on g. It is often an art to chose which prior to use. For example, a smooth function that is sparse in its gradient [3] may have:

$$\text{Prior}(g) = \sum_{i=1}^{L-1} |g[i+1] - g[i]| \tag{3}$$

However, the original signal may not be smooth at all and the chosen prior obliviously blurs the possible sharp transi-

Fig. 1. The number of extrema of a sampled function can be drastically different from the number of zeros of its discrete gradient. From left to right: the signals have no extrema, 1 local maximum, 3 local maxima and 2 local minima, respectively. The definition of a local extremum is in Section 2.

tions. Can we do better with a prior, that is simple in concept and avoids transition erosion?

1.2. A Topological Prior

We propose to use the number of extrema of a function to form the prior term in Equation 1. The rationale is that the number of extrema of a function is a useful characterization of its complexity. Take polynomials for example, any polynomial of degree $d+1$:

$$g(x) = c_0 x^{d+1} + c_1 x^d + \cdots + c_d x + c_{d+1} \quad (c_0 \neq 0)$$

has at most d local extrema. In other words, the greater the number of extrema of g, the higher the degree of the polynomial is required to reconstruct it, the more oscillatory and complex the function is perceived to be.

The number of extrema of a function is a topological quantity independent of individual functional values. Sharp transitions incurs no penalty unless local extrema are induced. Conversely, two local extrema induce the same amount of penalty regardless of how much the values differ. In computational topology, this measure has been used to simplify a Morse function defined on a manifold, a technique also known as persistence based simplification [7].

Due to possible ties in values and the global patterns exhibited in the function, the number of extrema of g is in general different from the number of zero entries of the gradient of g, expressed by $L - \|\nabla g\|_0$ (See Figure 1).

1.3. Our Contributions

First, oscillation regularization is a global optimization: The role of the data term ensures that the restored function respects the actual values of the original input. The topological prior is conceptually simple but quite effective in practice. Second, we impose no assumptions on the input function and allow it to be an arbitrary L-dimensional vector with possibly many ties in values. Third, when the range is finite, our method enjoys a computational efficiency on par with local methods.

1.4. Organization

Section 2 presents relevant concepts and the framework of oscillation regularization. Section 3 introduces efficient algorithms for the global optimization using dynamic programming and linear programming. Section 4 demonstrates the advantage of oscillation regularization in effect and speed by comparisons to other popular methods. Section 5 concludes.

2. OSCILLATION REGULARIZATION

In the one-dimensional case, a local maximum or minimum is either a single point or an interval because of possible ties in the values. To formalize, we have:

Definition 1 (Local Maximum and Minimum)**.** *An interval $\mathcal{I} = [i, j] \subseteq (1, L)$ is a local maximum of g if $g[k]$ is a constant for $i \leq k \leq j$ and $g[i] > g[i-1]$ as well as $g[j] > g[j+1]$ where i, j, k all take integer values. $\mathcal{I}$ is a local minimum of g if it is a local maximum of $-g$.*

We use $\mathrm{osc}(g)$ to represent the number of local maxima and minima of g. The optimization in oscillation regularization, as we have described, is therefore to find:

$$\tilde{g} = \arg\min_{g \in \mathcal{V}^L} \left\{ \underbrace{\|f - g\|_1}_{\text{data fitting error}} + \lambda \underbrace{\mathrm{osc}(g)}_{\text{\# extrema of } g} \right\} \tag{4}$$

where λ is a regularization parameter. Note that λ has a physical meaning: it measures how much "work" needs to be done in order to remove the unstable local extremum (Figure 2), where the stability is measured by the incurred data cost.

For the ease of algorithm description, we first define ascending and descending intervals, similar to the notion of ascending and descending paths in critical net [8]:

Definition 2 (Ascending and Descending Intervals)**.** *An interval $[i, j], 1 \leq i < j \leq L$, is an ascending interval of g if $g[i] \leq g[i+1] \leq \cdots \leq g[j-1] \leq g[j]$. The interval $[i, j]$ is a descending interval of g if it is an ascending interval of $-g$.*

The alternation between ascending and descending intervals induces local extrema. An ascending (descending) interval is *maximal* if it is contained in no other ascending (descending) intervals. Let the number of maximal ascending and descending intervals of g be $\mathrm{asc}(g)$ and $\mathrm{des}(g)$, we have:

Theorem 1. *For any $g \in \mathcal{V}^L$ with $osc(g) \geq 1$,*

$$osc(g) = asc(g) + des(g) - 1 \tag{5}$$

The proof is omitted for brevity. The fact that local extrema are induced by an alternating sequence of maximal ascending and descending intervals form the basis for efficient optimization in oscillation regularization.

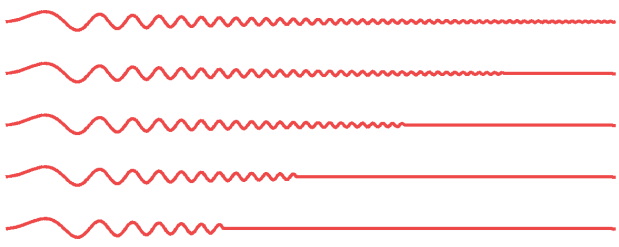

Fig. 2. Top row: input signal with range $V = 500$, in the form of a damped oscillation with increasing frequency from left to right. The bottom four rows: results of oscillation regularization with $\lambda = 100, 200, 500$ and 1000 respectively.

3. THE ALGORITHMS

We describe two algorithms. When the range space is finite (i.e. $\mathcal{V} = \{1, 2, \cdots, V\}$), we show that the global optimization in Equation 4 can be computed in $O(VL)$ time using dynamic programming. When $\mathcal{V} = \mathbb{R}$, we give a polynomial time algorithm in L alone using linear programming.

3.1. Finite Range: $\mathcal{V} = \{1, \cdots, V\}$

For ease of exposition, we first describe a simple algorithm of complexity $O(V^2L)$ using dynamic programming. We then improve the complexity to $O(VL)$ by a book keeping method.

3.1.1. The $O(V^2L)$ Algorithm

The dynamic programming procedure is intuitive: At each step, either an ascending interval continues from a previous ascending interval or it transits from a descending interval. In the latter case a penalty λ is incurred. The analysis works similarly for descending intervals as well.

Let $c^+(l, v)$ be the optimal cost when the optimal solution takes value v at l. Moreover, the superscript $+$ means that the value v is reached through an ascending interval. Similarly, let $c^-(l, v)$ be the optimal cost at position l and value v, but through a descending interval. We then can write the following state equations for dynamic programming:

$$\begin{cases} c^+(l,v) = e(l,v) + \min \begin{cases} \min_{v' \le v} c^+(l-1, v') \\ \lambda + \min_{v' < v} c^-(l-1, v') \end{cases} \\ c^-(l,v) = e(l,v) + \min \begin{cases} \min_{v' \ge v} c^-(l-1, v') \\ \lambda + \min_{v' > v} c^+(l-1, v') \end{cases} \end{cases}$$

where $e(l, v) = |f[l] - v|$ is the data fitting error.

The boundary conditions $c^+(1, v) = c^-(1, v) = |f[1]-v|$ hold for each $v \in \mathcal{V}$. Since dynamic programming takes L steps and at each step visits $O(V)$ items, each taking $O(V)$ time, the overall complexity is $O(V^2L)$.

3.1.2. The $O(VL)$ Algorithm

In the above dynamic programming procedure, we visit $O(V)$ items for each $v \in \mathcal{V}$ at each step. This is not necessary if we use a simple book keeping with intermediate variables:

$$C^+_{\le}(l, v) \triangleq \min_{v' \le v} c^+(l, v') \; ; \; C^-_{\le}(l, v) \triangleq \min_{v' \le v} c^-(l, v')$$
$$C^-_{\ge}(l, v) \triangleq \min_{v' \ge v} c^-(l, v') \; ; \; C^+_{\ge}(l, v) \triangleq \min_{v' \ge v} c^+(l, v')$$

We then obtain the following shortened state equations:

$$c^+(l, v) = e(l, v) + \min \left\{ C^+_{\le}(l-1, v), \lambda + C^-_{\le}(l-1, v-1) \right\}$$
$$c^-(l, v) = e(l, v) + \min \left\{ C^-_{\ge}(l-1, v), \lambda + C^+_{\ge}(l-1, v+1) \right\}$$

These variables can be updated in $O(1)$ time:

$$C^+_{\le}(l, v) = \min \left\{ C^+_{\le}(l, v-1), c^+(l, v) \right\}$$
$$C^+_{\ge}(l, v) = \min \left\{ C^+_{\ge}(l, v+1), c^+(l, v) \right\}$$
$$C^-_{\le}(l, v) = \min \left\{ C^-_{\le}(l, v-1), c^-(l, v) \right\}$$
$$C^-_{\ge}(l, v) = \min \left\{ C^-_{\ge}(l, v+1), c^-(l, v) \right\}$$

The overall complexity is therefore reduced to $O(VL)$, which is both asymptotically and practically fast.

3.2. Infinite Range: $\mathcal{V} = \mathbb{R}$

We give a polynomial time algorithm when the range is infinite. Notation is the same except that we write $c^+(l)$ and drop the second index on values. Let $T^+(i, j)$ be the minimal cost of modifying $[i, j]$ to an ascending interval. $T^-(i, j)$ is defined symmetrically. We have the following state equations:

$$c^+(l) = \min \left\{ \min_{2 \le i \le l} \left\{ c^-(i-1) + T^+(i, l) + \lambda \right\}, T^+(1, l) \right\}$$
$$c^-(l) = \min \left\{ \min_{2 \le i \le l} \left\{ c^+(i-1) + T^-(i, l) + \lambda \right\}, T^-(1, l) \right\}$$

Suppose that computing $T^+(i, l)$ takes time $h(l - i + 1)$. The overall time complexity is therefore $O(L^2 h(L))$. We show that $h(L)$ is a polynomial by reducing the computation of $T^+(1, L)$ to linear programming, which is polynomially solvable. Indeed, we are facing the constrained optimization:

Given: $f[1], f[2], \cdots, f[L]$;

Goal: $\min \sum_{l=1}^{L} |f[l] - g[l]|$;

Require: $g[1] \le g[2] \le \cdots \le g[L]$;

To solve, we introduce auxiliary variables $\varepsilon_i \ge 0$ and the constraints: $-\varepsilon_i \le g[i] - f[i] \le \varepsilon_i$ for $1 \le i \le L$. The additional constraints are: $g[1] \le g[2] \le \cdots \le g[L]$. The objective is therefore: $\min_{l=1}^{L} \varepsilon_l$. Clearly both the objective and the inequality constraints are linear, and hence the conclusion.

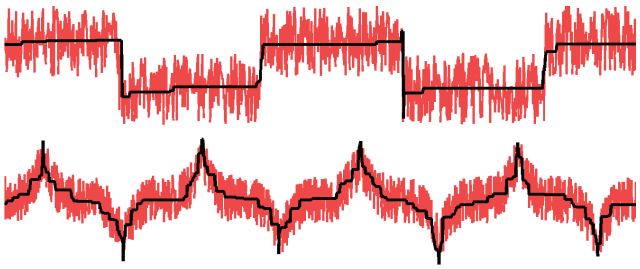

Fig. 3. Red: input functions with $V = 500$ and $L = 1000$. Black: results of oscillation regularization with $\lambda = 500$.

4. EXPERIMENTS

For ease of experiments, we use the $O(VL)$ algorithm for its efficiency. Three experiments are performed.

The first experiment applies oscillation regularization to noisy signals. Figure 3 shows two sample results of signal denoising. Importantly, our method preserves sharp transitions and copes well with the ground truth of the inputs.

In the second experiment (Table 1), we compare our method to state-of-the-art methods, notably median, Gaussian, bilateral and total variation (TV) with l_1 and l_2 norms. For fair comparisons, for each method, we search for the optimal parameter (e.g. the window size) that minimizes the error measure when compared against the ground truth. Let t be the ground truth of f. We use the error measure:

$$e = \frac{1}{L}\sum_{l=1}^{L} |g[l] - t[l]| \tag{6}$$

Experiments show that oscillation regularization performs best in our comparison, regardless of the amount of noise.

In the last experiment, we extend our method to the simplification of a geometric figure, parameterized by its boundary points $\{(\rho[l], \theta[l])\}_{l=1}^{L}$ in polar coordinate, and apply oscillation regularization to ρ (Figure 4).

It takes two milliseconds to compute the global optimization of oscillation regularization for a sampled function with $L = 1000$ and $V = 500$ on a quad-core laptop. The MATLAB/MEX code is available at `http://www.cs.duke.edu/~steve/oscillation.html`

5. CONCLUSIONS

Oscillation regularization reduces both the data fitting error and the number of local extrema in a global optimization. Interesting, open questions are how to determine the regularization parameter λ automatically and how to extend the computation to high dimensional spaces.

Acknowledgement: This work is supported by the Army Research Office under Grant No. W911NF-10-1-0387.

Table 1. Error measure. Bold: best. Underlined: second best. W1 and W2 are the input shown in Figure 3.W2-2 and W2-3 are the same as W2 but with increasing amounts of noise.

Input	Gaussian	Bilateral	Median	TV-2	TV-1	OSC
W1	6.54	6.53	9.53	10.06	<u>5.16</u>	**3.87**
W2	8.00	7.97	9.35	13.75	<u>7.63</u>	**6.80**
W2-2	8.12	8.10	9.12	13.50	<u>7.83</u>	**7.09**
W2-3	19.92	19.27	21.43	43.20	<u>17.71</u>	**15.67**

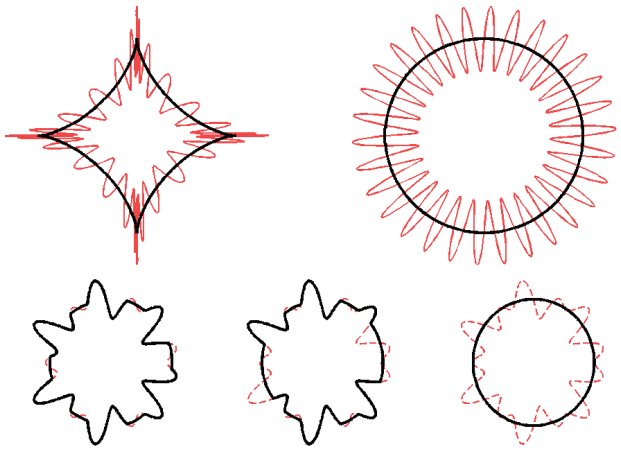

Fig. 4. Red: the input shapes with oscillatory perturbations. Black: the simplified shape via oscillation regularization. The bottom row shows the results when $\lambda = 1000, 2500, 5000$ respectively. As $\lambda \to \infty$, the figure reduces to a circle.

6. REFERENCES

[1] J. Lee, "Digital image enhancement and noise filtering by use of local statistics," *PAMI*, pp. 165–168, 1980.

[2] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *ICCV*, 1998, pp. 839–846.

[3] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physics*, pp. 259–268, 1992.

[4] S. Sardy, P. Tseng, and A. Bruce, "Robust wavelet denoising," *Transactions on Signal Processing*, pp. 1146–1152, 2001.

[5] E. Candes, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Comm. Pure Appl. Math.*, pp. 1207 – 1223, 2005.

[6] P. Felzenszwalb and R. Zabih, "Dynamic programming and graph algorithms in computer vision," *PAMI*, vol. 33, no. 4, pp. 721–740, 2011.

[7] H. Edelsbrunner, J. Harer, and A. Zomorodian, "Hierarchical morse complexes for piecewise linear 2-manifolds," in *Symp. on Computational Geometry*, 2001, pp. 70–79.

[8] S. Gu, Y. Zheng, and C. Tomasi, "Critical nets and beta-stable features for image matching," in *ECCV*, 2010, pp. 663–676.