A STUDY OF p-VARIATION AND THE p-LAPLACIAN FOR $0 < p \leq 1$ AND FINITE HYPERPLANE TRAVERSAL ALGORITHMS FOR SIGNAL PROCESSING

By

HEATHER A. VAN DYKE

A dissertation submitted in partial fulfillment of
the requirements for the degree of

DOCTOR OF PHILOSOPHY

WASHINGTON STATE UNIVERSITY
Department of Mathematics

May 2013

To the Faculty of Washington State University:

The members of the Committee appointed to examine the dissertation of HEATHER A. VAN DYKE find it satisfactory and recommend that it be accepted.

Thomas J. Asaki, Ph.D., Chair

Kevin R. Vixie, Ph.D.

David J. Wollkind, Ph.D.

Matthew B. Rudd, Ph.D.

ACKNOWLEDGEMENT

First, I would like to thank God for His loving kindness and grace in supplying all the right people at the right time for me to accomplish this goal. Without His strength, I would not be where I am today.

Second, I would like to express my deepest thanks to my advisor and friend, Tom Asaki. Your support and friendship is invaluable to me. Over these last three years you have helped me grow as a mathematician and a person. There are not enough words to express how grateful I am that God put you in my life. You've shown me how to laugh at myself when appropriate (like when I proved something was $\leq \infty$). You encouraged me when all seemed lost (every time). You supported me when I felt the need to fight for my principles. I look forward to many more years (at least 62) of friendship and collaboration.

I would also like to thank my committee for their support and collaborations. Kevin Vixie, you set me on my path to learning about myself and without that first step, I would not be where I am today. I also want to thank you for taking the time to stop in my office on October 17, 2008, to meet me and invite me to work with you. I really appreciate the opportunity that you laid in front of me.

Matthew Rudd, thank you for all of your time and all the interesting discussions during which I learned a great deal about PDEs. Thank you also for letting me work on the Medians paper with you, that was fun. I look forward to many more collaborations with you.

David Wollkind, I have enjoyed our morning discussions. I hope to work on a project together someday soon. Thank you also for agreeing to be on my committee and for taking the time to be interested in my work.

I would also like to thank Rick Chartrand for talking $p < 1$ stuff with me. You are the person who inspired me to pursue this topic and I really appreciate all of your help. I look forward to many more discussions and collaborations with you.

I am grateful to the mathematics department at Washington State University for giving me a place at which I could pursue my goals and dreams. Thank you for providing an environment that made this work possible. Thank you to Diane, Kris, and Jessanne for lots of fun conversations, hugs, and support. I will miss you all. Thank you Linda for putting up with my jokes.

Thank you Matt Hudelson for agreeing to substitute on my committee. Thank you for sitting in on my defense even though it turned out to be unnecessary. I really appreciate you willingness to help me.

I would like to thank my classmate and husband, Benjamin, for being patient with me while I pursued my PhD. Thank you for enduring my long hours. Thank you for never giving up on me, I love you.

Thank you to my classmates, Sharif Ibrahim, Amy Streifel, Eric Larson, and Abigail Higgins for making this fun (at times, ahem,... Eric). Let's always keep in touch. I hope for the best for all of your futures. I also hope that they intersect mine many more times.

I want to thank my friends, Melanie, Renee, and Beata, who offered me encouraging words and thoughtful conversations. These three years have been a bit of a roller coaster and I'm so thankful to God for putting such wonderful people like you in my life and at the perfect times. I look forward to many years of friendship with each of you.

Finally, I want to thank my dad for believing in me. You have always been my biggest fan even when I didn't feel like I deserved it. Thank you, dad, I love you.

This poem was an encouragement in one of those times when I needed it most. At times, it seemed like there was no hope, but there was always a thrush singing.

The Darkling Thrush

I leant upon a coppice gate
When Frost was spectre-grey,
And Winter's dregs made desolate
The weakening eye of day.

The tangled bine-stems scored the sky
Like strings of broken lyres,
And all mankind that haunted nigh
Had sought their household fires.

The land's sharp features seemed to be
The Century's corpse outleant,
His crypt the cloudy canopy,
The wind his death-lament.
The ancient pulse of germ and birth
Was shrunken hard and dry,
And every spirit upon earth
Seemed fervourless as I.

At once a voice arose among
The bleak twigs overhead
In a full-hearted evensong
Of joy illimited;
An aged thrush, frail, gaunt, and small,
In blast-beruffled plume,
Had chosen thus to fling his soul
Upon the growing gloom.

So little cause for carolings
Of such ecstatic sound
Was written on terrestrial things
Afar or nigh around,
That I could think there trembled through
His happy good-night air
Some blessed Hope, whereof he knew
And I was unaware.
–Thomas Hardy
December 29, 1900

One last note for the reader: a tip for the proof of Lemma 5.13 is to remember in the definitions of q_g, q_e, and q_ℓ, that it's all about u.

A STUDY OF p-VARIATION AND THE p-LAPLACIAN FOR $0 < p \leq 1$ AND FINITE HYPERPLANE TRAVERSAL ALGORITHMS FOR SIGNAL PROCESSING

Abstract

by Heather A. Van Dyke, Ph.D.
Washington State University
May 2013

Chair: Thomas J. Asaki

In this work (freely available by contacting the author), we study and find minimizers of the problem

$$\min \int_\Omega |\nabla u|^p + \lambda |f - u| \, dx, \quad \text{for } 0 < p \leq 1,$$

where $\Omega \subset \mathbb{R}^n$ is an open bounded convex set. We find fast algorithms that find minimizers for the $p = 1$ problem and local minimizers for the $p < 1$ problem. Our algorithms solve the minimization problem for $p = 1$ for all λ at the computational cost of solving only the $\lambda = 0$ problem. We also find and characterize the set of minimizers of the $\lambda = 0$ problem.

We compare minimizers to stationary solutions to the p-Laplacian evolution equation

$$u_t = \Delta_p u, \quad \text{for } 0 < p < 1.$$

We also consider a curvature approach to understanding the nature of the evolution in this equation. We use the curvature understanding to find families of classical stationary solutions.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Dedication

To my love

CHAPTER ONE

INTRODUCTION

The goal of this work is to study and find minimizers of the minimization problem

$$\min_{u \in L^2(\Omega)} \int_\Omega |\nabla u|^p + \lambda |f - u| dx, \quad \text{with } 0 < p \leq 1, \tag{1.1}$$

where $\Omega \subset \mathbb{R}^n$ is an open bounded convex set, for image and data analysis tasks. In this work, we will call this problem $L^1 pTV$. Of particular interest, are finding fast algorithms for the $p \leq 1$ cases and seeking minimizers of the $\lambda = 0$ case

$$\int_\Omega |\nabla u|^p, \quad 0 < p < 1. \tag{1.2}$$

I also consider other stationary points for (1.2) by considering the stationary solutions of the corresponding Euler-Lagrange equation, which is the p-Laplacian evolution equation,

$$\begin{cases} u_t = \Delta_p u \equiv \nabla \cdot \left(|\nabla u|^{p-2} \nabla u \right) & \text{in } \Omega \times [0, \infty) \\ u = u_0 & \text{in } \Omega \times \{0\} \end{cases}, \tag{1.3}$$

for $0 < p < 1$ and $\Omega \subset \mathbb{R}^n$. Here we use ∇ to mean derivatives with respect to the spatial variables.

1.1 Motivation and Past Work

The motivations behind these goals are found in signal and image analysis tasks such as denoising and finding scales in data. The idea for image denoising is that we are given noisy data, $f : \Omega \subset$

$\mathbb{R}^n \to \mathbb{R}$, from which we want to obtain an approximation, u^*, to the true image. The goal in denoising is to find u^* satisfying the expectations that it should approximate, in some sense, f and have little variation, since a noisy image will have high variation.

Variational and PDE based methods of denoising have been used for more than two decades ([29],[26],[6],and [8]). The most notable variational techniques solve the problem

$$u^* = \arg\min_{u \in L^2(\Omega)} \int_\Omega |\nabla u|^p + \lambda |f - u|^q dx, \tag{1.4}$$

where $p, q > 0$ and the fidelity and variation terms are weighted by $\lambda > 0$. We can see that the second term is small when u is close to f and so this term addresses the data fidelity requirement. We also see that if $|\nabla u|$ is small, that is, if u has little variation, then the first term is small as well. Notice that for large λ, the minimizer will be more like f and for small λ, the minimizer should have very little variation. In the case of denoising, with f a noisy signal or image, the minimizer when λ is very large is noisy, but the minimizer when λ is very small is rather flat.

Before discussing particular results, we discuss the calculus of variations used to find the desired minimizers. Given the minimization problem

$$u^* = \arg\min_u \int_\Omega \mathcal{L}(x, u, \nabla u)\, dx, \tag{1.5}$$

we can seek minimizers by considering the Euler-Lagrange equation (or the weak form of the Euler-Lagrange equation). More specifically, we know that if $\xi \mapsto \mathcal{L}(x, z, \xi)$ is convex, then solutions to the Euler-Lagrange equation

$$\nabla \cdot \mathcal{L}_\xi(x, u, \nabla u) = \mathcal{L}_z(x, u, \nabla u) \tag{1.6}$$

or solutions to the weak form of the Euler-Lagrange Equation

$$\int \mathcal{L}_\xi(x, u, \nabla u)\varphi + \mathcal{L}_z(x, u, \nabla u)\varphi dx = 0 \quad \forall \text{ test functions } \varphi \tag{1.7}$$

are minimizers (and stationary points) of (1.5). If $\xi \mapsto \mathcal{L}(x, z, \xi)$ is strictly convex, the only stationary point of problem (1.5) is the minimizer. And, thus, the solution to the Euler-Lagrange Equation is unique and is the minimizer of (1.5). However, if $\xi \mapsto \mathcal{L}(x, z, \xi)$ is nonconvex, then stationary solutions to (1.5) can be local or global minimizers, saddle points, or local or global maximizers. Thus, solutions to the Euler-Lagrange equations are merely these stationary points for the functional (1.5) (see [11], [12], [14] for more details).

Computationally, a common technique is to seek the minimizers of (1.5) using the method of gradient descent. That is, we start with an initial guess and we use the 'gradient' of the functional given in (1.5) to determine the direction of steepest descent, in the space of functions, to take us to a minimizer. That is, we introduce a new variable, t and seek $u : \mathbb{R}^n \times [0, \infty) \to \mathbb{R}$ so that $u(x, t)$ satisfies the evolution equation

$$u_t(x, t) = \mathcal{L}_z(x, u(x, t), \nabla u(x, t)) - \nabla \cdot \mathcal{L}_\xi(x, u(x, t), \nabla u(x, t)) \tag{1.8}$$

given initial data $u(x, 0) = u_0(x)$. Since the right-hand side is the negative gradient of the functional, we expect that u changes over time so that $\int_\Omega \mathcal{L}(u, \nabla u)\, dx$ goes down to a minimizer. So, we look to the limit as $t \to \infty$ to seek a stationary solution to (1.8). Notice that when the left-hand side of (1.8) is zero (that is, at a stationary solution), we have found a solution to (1.6).

Let us define the functional, $\mathcal{F}_{p,q} : L^2(\Omega) \to \mathbb{R}$ by

$$\mathcal{F}_{p,q}(u, \nabla u) = \int_\Omega |\nabla u|^p + \lambda |f - u|^q dx. \tag{1.9}$$

We now briefly discuss a few of the results for particular values of p and q.

$\mathcal{F}_{2,2}(u, \nabla u)$ was first introduced by Tikhonov [29]. This functional is strictly convex. Using results from the calculus of variations, we can say that there exists a unique minimizer. We can also find the minimizer by solving the corresponding Euler-Lagrange equation,

$$\Delta u = \lambda(f - u).$$

And, solutions to this equation are stationary solutions to the diffusion equation given by

$$u_t = \Delta u - \lambda(f - u).$$

In image denoising, we find the minimizer of $\mathcal{F}_{2,2}(u, \nabla u)$, has smoothed edges around objects ($\mathcal{F}_{2,2}(u, \nabla u)$ is larger for functions with jump discontinuities than for those that will increase steadily and so edge location is lost). Pixel intensity is also lost. This problem then is not sufficient for images with regions of high contrast or well defined object edges.

Rudin, Osher, and Fatemi proposed, in [26], minimizing $\mathcal{F}_{1,2}(u, \nabla u)$, also called the ROF functional, to allow jump discontinuities in u^* which makes sense in many real images. In this case, the regularization term is the total variation, $\int |\nabla u|$, which does not penalize jump discontinuities. $\mathcal{F}_{1,2}(u, \nabla u)$ is also strictly convex and the corresponding Euler-Lagrange equation has a unique solution and it is

$$\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) = \lambda(f - u)$$

the unique minimizer. In image denoising, we see that minimizers preserve the location of object edges, but still lose contrast (even when f is a noiseless image) and features with high curvature are lost [28]. That is, corners get rounded.

In [6], Chan and Esedoḡlu show that minimizing the L^1TV functional, $\mathcal{F}_{1,1}$, for imaging tasks will preserve pixel intensity. However, features of high curvatures are still lost. This functional is again convex, but this time it is not strictly convex. Thus minimizers can be found by solving the corresponding Euler-Lagrange equation,

$$\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) = \lambda \frac{f - u}{|f - u|}.$$

But, it should be noted that we cannot guarantee a unique minimizer for L^1TV. For a discussion about the discretized L^1TV see also [4] and [24].

In 2007, Chartrand [8] proposed to minimize $\mathcal{F}_{p,2}$ for $0 < p < 1$. It is worth noting that the

functionals, $\mathcal{F}_{p,2}$, are not convex and therefore standard methods do not guarantee that we find a global minimizer. Despite this lack of guarantee, Chartrand found success in obtaining what appear to be local minimizers by solving the corresponding Euler-Lagrange equation iteratively by substituting u_n into $|\nabla u|$ and then solving for u_{n+1} in the equation

$$\left(-\nabla \cdot \left(|\nabla u_n|_\beta^{p-2}\nabla\right) + \lambda I\right) u_{n+1} = \lambda f, \tag{1.10}$$

where $|\nabla u|_\beta = \sqrt{|\nabla u|^2 + \beta^2}$ is used instead of $|\nabla u|$ to avoid division by zero. For a cartoon image or an image with piecewise constant intensities, these solutions preserve object edges, pixel intensity, and areas of high curvature where sharp corners occur.

This led us to ask what difference the exponent on the regularization term makes. In order to understand how better results on noise reduction and edge preservation are obtained by only changing the exponent in the first term, for $0 < p < 1$, I consider the problem

$$\min_u \int_\Omega |\nabla u|^p. \tag{1.11}$$

The Euler-Lagrange equation is then the p-Laplacian equation

$$-\Delta_p u \equiv \nabla \cdot \left(|\nabla u|^{p-1}\frac{\nabla u}{|\nabla u|}\right) = 0. \tag{1.12}$$

We assume that the boundary data is fixed, that is, we suppose $u|_{\partial\Omega} = f|_{\partial\Omega}$. A typical way to find solutions to equation (1.12) is to iteratively solve the evolution equation (1.3). That is, we let an initial guess $u(0, x) = f(x)$ evolve over time, keeping the boundary fixed, as above, so that at any time t, $u(x, t)$ satisfies (1.3). The stationary solutions to (1.3) are solutions to (1.12).

Notice that when $p = 2$, (1.12) is Laplace's equation and (1.3) is the heat equation. Both equations are well studied and existence and uniqueness of solutions is well known, see [14].

Existence and uniqueness of local weak solutions for the degenerate ($p > 2$) case can be found

in [13] for the parabolic equation, (1.3). In fact, a Barenblatt solution is given for the problem

$$u \in C_{loc}(0,T;L^2_{loc}(\mathbb{R}^N)) \cap L^p_{loc}(0,T;W^{1,p}_{loc}(\mathbb{R}^N)), p > 2$$
$$u_t = \nabla \cdot (|\nabla u|^{p-2}\nabla u) \text{ in } \mathbb{R}^N \times (0,T) \tag{1.13}$$

by

$$\mathcal{B}(x,t) := t^{-N/\lambda}\left[1 - \gamma_p \left(\frac{|x|}{t^{1/\lambda}}\right)^{\frac{p}{p-1}}\right]_+^{\frac{p-1}{p-2}}, t > 0,$$
$$\gamma_p := \lambda^{-\frac{1}{p-1}}\frac{p-2}{p}, \lambda = N(p-2)+p. \tag{1.14}$$

Further, uniqueness is also established for any nonnegative solution defined on $\mathbb{R}^N \times (0,T)$ in the following sense.

Theorem 1.1. *(See [13] Thm 6.1) Given two nonnegative weak solutions, u, v of (1.13), if*

$$\lim_{t \searrow 0}(u(\cdot,t) - v(\cdot,t)) = 0, \text{ in the sense of } L^1_{loc}(\mathbb{R}^N),$$

then $u \equiv v$ in $\mathbb{R}^N \times (0,T)$.

In [18], an equivalence connecting radially symmetric solutions of the porous medium equation,

$$u_t = \Delta(u^m/m) \tag{1.15}$$

and those of the p-Laplacian evolution equation is given. More clearly, Iagar, Sánchez, and Vázquez give the following theorem.

Theorem 1.2. *(See [18]) Let $0 < m < 1$. Given $u : \mathbb{R}^n \to \mathbb{R}, \bar{u} : \mathbb{R}^{\bar{n}} \to \mathbb{R}$. Then u is a radially symmetric solution to (1.15) if and only if $\bar{u}$ is a solution to (1.3) where $\bar{u}$ is given by*

i.) for $0 < n < 2$,

$$\bar{u}_{\bar{r}}(\bar{r},t) = Kr^{\frac{2n-2}{m+1}}u(r,t), K := \left(\frac{(mn-n+2)^2}{m(m+1)^2}\right)^{\frac{1}{m-1}}, \tag{1.16}$$

where

$$p = m + 1, \bar{n} = \frac{(n-2)(m+1)}{n - mn - 2}$$

and $\bar{r} = r^{(mn-m+2)/(m+1)}$ *or*

ii.) for $n > 2$*,*

$$\bar{u}_{\bar{r}}(\bar{r}, t) = Kr^{\frac{2}{m+1}} u(r, t), K := \left(\frac{(2m)^2}{m(m+1)^2}\right)^{\frac{1}{m-1}}, \tag{1.17}$$

where

$$p = m + 1, \bar{n} = \frac{(n-2)(m+1)}{2m}$$

and $\bar{r} = r^{2m/(m+1)}$.

Using this result, we can find a Barenblatt solution to the porous medium equation (1.15) and then use it to find a solution to the p-Laplacian evolution equation (1.3) for $1 < p < 2$. Note that the dimension of the domain space of the solution to (1.15) is different than that of the solution to (1.3).

Iagar and Sánchez give a similar result for $0 < p < 1$ ($1 < m < 0$) in [17], but for the related equation

$$u_t = \frac{1}{p-1} \nabla \cdot \left(|\nabla u|^{p-2} \nabla u \right), \tag{1.18}$$

which they called the very fast p-Laplacian evolution equation. Their result is

Theorem 1.3. *(See [17]) For* $0 < p < 1$ *and* $m = p - 1$*, the radially symmetric solutions* $u : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}_{\geq 0}$ *and* $\bar{u} : \mathbb{R}^{\bar{n}} \times \mathbb{R} \to \mathbb{R}$ *of (1.15) and (1.18) respectively are related through the transformation*

$$\bar{u}_{\bar{r}}(\bar{r}, t) = Kr^{\frac{2n-2}{p}} u(r, t),$$

where

$$K = \left(\frac{(pn - 2n + 2)^2}{p^2}\right)^{\frac{1}{p-2}}, \bar{n} = \frac{p(n-2)}{2n - pn - 2}, \bar{r} = r^{\frac{pn-2n+2}{p}},$$

and $n \in (0, 2)$.

We now find a self-similar radially symmetric solution to (1.15) with $-1 < m < 0$ and use the theorem to find a self-similar radially symmetric solution to (1.18) with $0 < p < 1$.

That is, we seek u in the form,

$$u(x,t) = Mu(Lx, Tt). \tag{1.19}$$

In particular, we want u of the form

$$u(x,t) = t^{n\beta} v(xt^{\beta}). \tag{1.20}$$

We find the constant β that makes all this work by first plugging this into (1.15). This gives

$$n\beta t^{n\beta-1} v(xt^{\beta}) + \beta t^{n\beta-1} xt^{\beta} \cdot \nabla v(xt^{\beta}) = \frac{1}{m} t^{mn\beta+2\beta} \Delta(v^m)(xt^{\beta}). \tag{1.21}$$

Then β must satisfy $n\beta - 1 = mn\beta + 2\beta$. Letting $y = xt^{\beta}$, we can rewrite this as

$$n\beta v(y) + \beta \nabla v(y) \cdot y = \frac{1}{m} \Delta(v^m)(y). \tag{1.22}$$

Now, since we are seeking a radial solution, we can assume $v(y) = w(|y|)$ for some function $w : \mathbb{R} \to \mathbb{R}$ and we get (letting $r = |y|$)

$$n\beta w + \beta r w' = \frac{1}{m} \left((w^m)'' + \frac{n-1}{r} (w^m)' \right). \tag{1.23}$$

After multiplying through by r^{n-1}, we have

$$\begin{aligned} nr^{n-1}\beta w + \beta r^n w' &= \frac{1}{m} (r^{n-1}(w^m)'' + (n-1) r^{n-2} (w^m)') \\ \Rightarrow \quad (\beta r^n w)' &= \frac{1}{m} \left((w^m)' r^{n-1} \right)'. \end{aligned}$$

Integrating, gives

$$(\beta r^n w) = \frac{1}{m}(w^m)' r^{n-1} + a, \tag{1.24}$$

for some constant, a. Since we are seeking any solution that is radially symmetric, we find the solution corresponding to $a = 0$. And now we solve

$$\beta r^n w = \frac{1}{m}(w^m)' r^{n-1}. \tag{1.25}$$

That is,

$$(w^m)' = m\beta r w.$$

We then compute $(w^{m-1})'$ by

$$(w^{m-1})' = (m-1)w^{m-2}w' = \frac{m-1}{mw}(mw^{m-1}w') = \frac{m-1}{mw}(w^m)'. \tag{1.26}$$

Plugging $m\beta rw$ in for $(w^m)'$ gives

$$(w^{m-1})' = \frac{m-1}{mw}(w^m)' = (m-1)\beta r. \tag{1.27}$$

Again, we integrate to get

$$w^{m-1} = (m-1)\frac{\beta}{2}r^2 + b, \tag{1.28}$$

for some constant b. Recall that $u(x,t) = t^{n\beta}w(|xt^\beta|)$. So undoing all the substitutions, gives our solution:

$$u(x,t) = t^{n\beta}\left(\frac{\beta(m-1)}{2}|x|^2 t^{2\beta} + b\right)^{1/(m-1)}, \tag{1.29}$$

where

$$\beta = \frac{1}{n - mn - 2}. \tag{1.30}$$

Using the theorem and the solution above, we integrate with respect to $\bar{r}$, but first, we need to

convert from r to $\bar{r}$ and n to $\bar{n}$. We have

$$u(r,t) = t^{n\beta}\left(\frac{\beta(p-2)}{2}r^2t^{2\beta} + b\right)^{1/(p-2)}, \tag{1.31}$$

So,

$$u(\bar{r},t) = t^{\frac{\bar{n}-p}{p(p-1)}}\left(\frac{(2-p)(2\bar{n}-p\bar{n}-p)}{4p(p-1)}\bar{r}^{\frac{2\bar{n}-p\bar{n}-p}{1-p}}t^{\frac{2\bar{n}-p\bar{n}-p}{p(p-1)}} + b\right)^{1/(p-2)}. \tag{1.32}$$

Using the theorem, we have

$$\bar{u}_{\bar{r}}(\bar{r},t) = Kr^{\frac{2n-2}{p}}u(r,t). \tag{1.33}$$

That is,

$$\bar{u}_{\bar{r}}(\bar{r},t) = K\bar{r}^{\frac{\bar{n}-1}{1-p}}t^{\frac{\bar{n}-p}{p(p-1)}}\left(\frac{(2-p)(2\bar{n}-p\bar{n}-p)}{4p(p-1)}\bar{r}^{\frac{2\bar{n}-p\bar{n}-p}{1-p}}t^{\frac{2\bar{n}-p\bar{n}-p}{p(p-1)}} + b\right)^{1/(p-2)}, \tag{1.34}$$

$$\text{where } K = \left(\frac{4(1-p)^2}{(2\bar{n}-p\bar{n}-p)^2}\right)^{\frac{1}{p-2}}. \tag{1.35}$$

Finally, to get our solutions, we integrate

$$\bar{u}(\bar{r},t) = Kt^{\frac{\bar{n}-p}{p(p-1)}}\int_a^{\bar{r}} x^{\frac{\bar{n}-1}{1-p}}\left(\frac{(2-p)(2\bar{n}-p\bar{n}-p)}{4p(p-1)}x^{\frac{2\bar{n}-p\bar{n}-p}{1-p}}t^{\frac{2\bar{n}-p\bar{n}-p}{p(p-1)}} + b\right)^{1/(p-2)}dx. \tag{1.36}$$

Thus, we have a solution to the very fast p-Laplacian (1.18) for $0 < p < 1$.

The case for $p = 1$ is addressed in [16] and [25]. In [25], we show that there is a connection between the local median value property for a function $u : \mathbb{R}^2 \to \mathbb{R}$ and viscosity solutions to the 1-Laplacian equation.

Definition 1.1. We say that a function u satisfies the local median value property if

$$u(x) = \operatorname*{median}_{\partial B_r(x)}\{\, u(s)\,\} \quad \text{for } x \in \Omega \text{ and } r \le R(x), \text{ where } R(x) > 0\,. \tag{1.37}$$

We define viscosity solutions of the equation

$$-\Delta_1 u \equiv |Du|\nabla \cdot \left(\frac{Du}{|Du|}\right) = 0 \tag{1.38}$$

using the definition Juutinen, et. al. give in [19].

Definition 1.2. The continuous function $\bar{u} : \Omega \to \mathbb{R}$ is a viscosity supersolution of (1.38) if and only if

1. $\bar{u} \not\equiv \infty$, and

2. whenever $x_0 \in \Omega$ and $\varphi \in C^2(\Omega)$ satisfies

$$\begin{cases} \bar{u}(x_0) = \varphi(x_0), \\ \bar{u}(x) > \varphi(x) \quad \text{for } x \neq x_0, \\ \nabla\varphi(x_0) \neq 0 \end{cases} \tag{1.39}$$

   φ also satisfies $-\Delta_1\varphi(x_0) \geq 0$.

The continuous function $\underline{u}$ is a viscosity subsolution of (1.38) if and only if $-\bar{u}$ is a viscosity supersolution of (1.38). And, u is 1-harmonic in the viscosity sense if u is both a viscosity subsolution and a viscosity supersolution of (1.38).

Note: Our definition of a 1-harmonic function differs from the usual definition, in that, we include the coefficient, $|Du|$. Since we only consider points $x_0 \in \Omega$ so that $|D\varphi(x_0)| \neq 0$, we see that this definition is equivalent to the usual definition. Notice, if u is a viscosity supersolution according to our definition, then we have

$$-|D\varphi(x_0)|\text{div}\left(\frac{D\varphi(x_0)}{|D\varphi(x_0)|}\right) \geq 0.$$

Since $|D\varphi(x_0)| \neq 0$, we can divide it out to get

$$-\text{div}\left(\frac{D\varphi(x_0)}{|D\varphi(x_0)|}\right) \geq 0.$$

which gives us that u is a viscosity supersolution in the usual sense. Conversely, if u is a viscosity supersolution in the usual sense, then multiplying by $|D\varphi|$ in the inequality above does not change the direction of the inequality. Thus, u is a viscosity supersolution in the sense of our definition. Similarly, u is a viscosity subsolution according to our definition if and only if u is a viscosity subsolution in the usual sense. Thus, a function is 1-Harmonic according to our definition if and only if u satisfies the usual definition of 1-Harmonic, in the viscosity sense.

Now, I can state our main theorem in [25].

Theorem 1.4. *A function u satisfying the local medium value property, (1.37), is 1-harmonic in the viscosity sense.*

1.2 Outline of This Work

In this work, we explore the relationship between Equations (1.2) and (1.3). In Chapter 2, we find and characterize the set of minimizers for (1.2) and a β-regularized version of (1.2). We show that minimizers are the set of step functions. We also show that this set is convex and neither open nor closed.

To discuss (1.3), we recognize that the singularity poses a difficulty in solving (1.3). We know that in the case of functions defined on 1-dimensional domains, if we want $u'(x) \neq 0$, we are looking for functions that are monotone. Though monotonicity for functions from $\mathbb{R}$ to $\mathbb{R}$ is familiar in even the most elementary courses in mathematics, there are a variety of definitions in the case of functions from $\mathbb{R}^n$ to $\mathbb{R}$. In Chapter 3 we review the definitions found in the literature and suggest a new definition (with its variants) some of which are useful in discussing classical solutions to (1.3). We give examples and discuss the relationship between these definitions and characterize functions that are monotone according to our new definition.

In Chapter 4, we discuss standard techniques for solving problems similar to (1.3). We discuss the difficulties in using these techniques. We discuss the differences between stationary solutions of (1.3) and minimizers of (1.2), in particular, we recognize that a function whose graph is a line is a stationary solution to the 1-dimensional (1.3), but it is not a minimizer of (1.2). We compute the divergence in (1.3) to write the equation as the sum of well-known curvature expressions. We then use this geometric understanding to find classical stationary solutions that satisfy certain monotonicity properties. We then use these solutions to discuss some differences between stationary solutions to (1.3) and minimizers of (1.2).

In Chapter 5, we write a discrete formulation for (1.9). We introduce finite hyperplane traversal (ht) algorithms that find (local) minimizers for L^1pTV for all $\lambda \geq 0$ and for all $0 < p \leq 1$. For the algorithms solving the problems when $\lambda > 0$, we also show convergence of these algorithms to minimizers for $p = 1$ and local minimizers for $0 < p < 1$, in only finitely many iterations. We discuss time trials that suggest that the computational complexity for the L^1TV algorithm is $o(aN + M)$ a signal of length N with M initial flat places in our data. Finally, we show that this is indeed true for binary signals.

In Chapter 6, we show examples denoising synthetic signals using our L^1TV and L^1pTV (for $0 < p < 1$) ht algorithms. We verify that the results in denoising are indeed the results that we expect to get when denoising using L^1TV and L^1pTV for $0 < p < 1$. We also show that using our L^1TV hyperplane traversal algorithm, we can easily detect scales in signals without the need to choose the particular λ for which we would solve L^1TV.

Finally, in Chapter 7, we summarize the findings of this work. We also discuss open ended questions related to this work.

CHAPTER TWO

CHARACTERIZING $\lambda = 0$ SOLUTIONS

In this chapter, we discuss the more general problem

$$\min_{u \in \mathcal{B}_n(\Omega)} \int_\Omega |\nabla u|_\beta^p \, dx, \quad \text{for } 0 < p < 1, \tag{2.1}$$

where

$$\mathcal{B}_n(\Omega) = \{u \in BV(\Omega) : \mathcal{H}^{n-1}(\mathcal{J}_\Omega(u)) < \infty\} \tag{2.2}$$

and

$$\mathcal{J}_\Omega(u) = \{x \in \Omega | u \text{ is discontinuous at } x\} \tag{2.3}$$

is the set of discontinuities of u. We define $|\nabla u|_\beta$ as

$$|\nabla u|_\beta = ((\nabla u)^2 + \beta^2)^{1/2}, \quad \text{for } \beta > 0. \tag{2.4}$$

Notice that the $\beta = 0$ case is the one in which we are particularly interested. We begin by recognizing that this functional is nonconvex. This tells us that we cannot expect to find a unique minimizer. In fact, we may have local as well as global minimizers. Indeed, our first two results give us that the set of minimizers is the set of step functions, verifying that we do not have a unique global minimizer [7].

2.1 Minimizers

Lemma 2.1. *Given $\Omega \subset \mathbb{R}, u : \Omega \to \mathbb{R}, 0 < p < 1, u \in \mathcal{B}_1(\Omega)$, that is, u has only finitely many jumps, then u is equal to a step function a.e. if and only if u is a minimizer of*

$$\min_{u \in \mathcal{B}_1(\Omega)} \int_\Omega |u'|_\beta^p \, dx \tag{2.5}$$

Proof. Since $|u'| \geq 0$, we have that $\int_\Omega |u'|_\beta^p \, dx = \int_\Omega \left(|u'|^2 + \beta^2\right)^{p/2} \, dx \geq |\Omega|\beta^p$. Let $m \in \mathbb{Z}^+$ and define

$$\tilde{u} = \sum_{i=1}^{m} \alpha_i \chi_{\Omega_i} \quad a.e. \text{ in } \Omega,$$

a step function, where $\alpha_i \in \mathbb{R}$ and Ω_i are intervals (a_i, a_{i+1}) such that $\overline{\bigcup_{i=1}^m \Omega_i} = \Omega$. We claim that $\min_u \int_\Omega |u'|_\beta^p \, dx = |\Omega|\beta^p$. It suffices to show that $\int_\Omega |\tilde{u}'|_\beta^p \, dx = |\Omega|\beta^p$. Indeed, let u_h be defined by

$$u_h(x) = \begin{cases} \frac{1}{h}\left(\tilde{u}(a_i) - \tilde{u}(a_i - h)\right)(x - a_i) + \tilde{u}(a_i) & \text{if } x \in (a_i - h, a_i) \\ & , \quad \forall i = 1, \ldots, m. \\ \tilde{u}(x) & \text{otherwise} \end{cases}$$

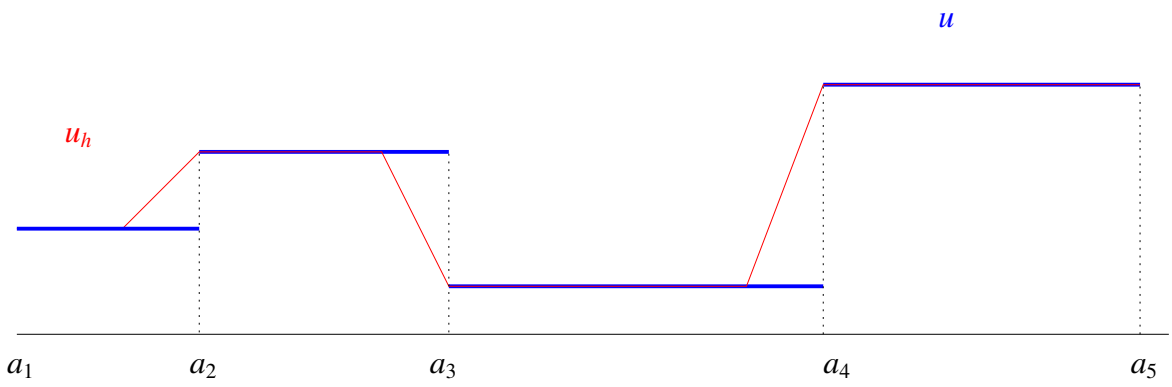


Figure 2.1: u_h, piecewise linear approximating a step function.

Then, $u_h \to \tilde{u}$ as $h \to 0$. Using Lebesgue Dominated Convergence, we have

$$\begin{aligned}
\int_\Omega |\tilde{u}'|_\beta^p \, dx &= \lim_{h\to 0} \int_\Omega \left|u_h'\right|_\beta^p \, dx = \lim_{h\to 0} \sum_{i=1}^{m} \int_{\Omega_i} \left|u_h'\right|_\beta^p \, dx \\
&= \lim_{h\to 0} \sum_{i=1}^{m-1} \left[\int_{a_i-h}^{a_i} \left(\frac{1}{h^2} \left(\tilde{u}(a_i) - \tilde{u}(a_i - h)\right)^2 + \beta^2 \right)^{p/2} \, dx + \int_{a_i}^{a_{i+1}-h} \beta^p \, dx \right] + \int_{a_{m-1}}^{a_m} \beta^p \, dx \\
&= \lim_{h\to 0} \sum_{i=1}^{m-1} \left[h^{1-p} \left(\left(\tilde{u}(a_i) - \tilde{u}(a_i - h)\right)^2 + h^2\beta^2 \right)^{p/2} + (a_{i+1} - h - a_i)\beta^p \right] + (a_m - a_{m-1})\beta^p \\
&= \sum_{i=1}^{m-1} (a_{i+1} - a_i)\beta^p = |\Omega|\beta^p.
\end{aligned}$$

Thus, $\tilde{u}$ is a minimizer of (2.5).

Conversely, suppose u is a minimizer of (2.5). Then, we have

$$\int_\Omega |u'|_\beta^p \, dx = |\Omega|\beta^p.$$

Then,

$$\int_\Omega \left((u')^2 + \beta^2 \right)^{p/2} \, dx = |\Omega|\beta^p.$$

Since $(u')^2 + \beta^2 \geq \beta^2$ and equality holds only when $(u')^2 = 0$ we get

$$|u'| = 0 \text{ a.e. in } \Omega \quad \Rightarrow \quad u' = 0 \text{ a.e in } \Omega.$$

Now, since u has only finitely many, say N, jumps (or points of discontinuity), we know that u is continuous except on a finite set. Let u be continuous on $\tilde{\Omega} = \Omega \setminus \mathcal{K}$, where $|\mathcal{K}| < \infty$. We now break $\tilde{\Omega}$ into the intervals separated by the jumps in u: $\tilde{\Omega} = \bigcup_{i=1}^{N} \tilde{\Omega}_i$. Then

$$u' = 0 \text{ on } \tilde{\Omega}_i \quad \Rightarrow \quad u = c_i \text{ on } \tilde{\Omega}_i \quad \Rightarrow \quad u = \sum_{i=1}^{N} c_i \chi_{\tilde{\Omega}_i} \text{ on } \tilde{\Omega}.$$

Thus, u is equal to a step function a.e. in Ω. □

We now extend the previous result to a higher dimensional result. That is, that minimizers of $\int_\Omega |\nabla u|^p \, dx$ are functions of the form $\sum_{i=1}^m \alpha_i \chi_{\Omega_i}$ where $\Omega_i \subset \Omega \subset \mathbb{R}^n$.

Theorem 2.1. *Given $\Omega \subset \mathbb{R}^n, u : \Omega \to \mathbb{R}, 0 < p < 1, u \in \mathcal{B}_n(\Omega)$ then there are $\alpha_i \in \mathbb{R}$, $\overline{\bigcup_{i=1}^m \Omega_i} = \Omega$ with $\partial\Omega_i$ lipschitz $\mathcal{H}^{n-1}(\partial\Omega_i) < \infty$ so that*

$$u = \sum_{i=1}^m \alpha_i \chi_{\Omega_i} \text{ a.e. in } \Omega \tag{2.6}$$

if and only if u is a minimizer of

$$\min_{u \in \mathcal{B}_n(\Omega)} \int_\Omega |\nabla u|_\beta^p \, dx \tag{2.7}$$

Proof. First, notice that since $\int_{\Omega_i} \left(|\nabla u|^2 + \beta^2\right)^{p/2} \, dx = \int_{\Omega_i} |\nabla u|_\beta^p \, dx$ we have

$$\begin{aligned} \beta^p |\Omega_i| &\leq \int_{\Omega_i} |\nabla u|_\beta^p \, dx \leq \sum_{k \neq i} \lim_{h \to 0} \int_{\partial\Omega_i \cap \partial\Omega_k} \int_0^h \left| \frac{\alpha_i - \alpha_k}{h} \right|^p \, dt d\sigma + \int_{\Omega_i} \beta^p \, dx \\ &= \sum_{k \neq i} \lim_{h \to 0} \mathcal{H}^{n-1}(\partial\Omega_i \cap \partial\Omega_k) h^{1-p} |\alpha_i - \alpha_k|^p + \beta^p |\Omega_i| = \beta^p |\Omega_i|. \end{aligned}$$

Thus,

$$\int_{\Omega_i} |\nabla u|_\beta^p \, dx = \beta^p |\Omega_i|.$$

And, so

$$\int_\Omega |\nabla u|_\beta^p \, dx = \sum_{i=1}^m \int_{\Omega_i} |\nabla u|_\beta^p \, dx = \beta^p \sum_{i=1}^m |\Omega_i| = \beta^p |\Omega|.$$

Now we prove the converse. We know that $\min \int_\Omega |\nabla u|_\beta^p \, dx = \beta^p |\Omega|$. Thus $|\nabla u| = 0$ a.e. which gives us that u is flat a.e.. Thus, we can find $\Omega_i \subseteq \Omega$ and $\alpha_i \in \mathbb{R}$ so that $u = \sum_{i=1}^m \alpha_i \chi_{\Omega_i}$. □

Theorem 2.1 gives us that the set of minimizers is

$$\mathcal{M} = \left\{ u \in \mathcal{B}_n(\Omega) : \exists \alpha_i \in \mathbb{R}, \Omega_i \subseteq \Omega, \partial\Omega_i \text{ Lipschitz }, \mathcal{H}^1(\partial\Omega_i) < \infty \text{ so that } u = \sum_{i=1}^m \alpha_i \chi_{\Omega_i} \right\}. \tag{2.8}$$

Using what we just learned about $\mathcal{M}$, we can now show that the set is convex.

2.2 Characteristics of $\mathcal{M}$

Lemma 2.2. *The set of minimizers of (2.7) is convex.*

Proof. Let $u, v \in \mathcal{M}$, that is, $\exists \alpha_i, \beta_i, \Omega_i, \Gamma_i$ so that

$$u = \sum_{i=1}^{m_1} \alpha_i \chi_{\Omega_i}, \qquad v = \sum_{i=1}^{m_2} \beta_i \chi_{\Gamma_i}.$$

Then

$$au = \sum_{i=1}^{m_1} (a\alpha_i)\chi_{\Omega_i}, \qquad bv = \sum_{i=1}^{m_1} (b\beta_i)\chi_{\Gamma_i} \in \mathcal{M}.$$

Thus

$$au + bv = \sum_{i=1}^{m_1+m_2} \gamma_i \chi_{\Lambda_i},$$

where $\gamma_1, \ldots, \gamma_{m_1} = a\alpha_1, \ldots, a\alpha_{m_1}$ and $\gamma_{m_1+1}, \ldots, \gamma_{m_1+m_2} = b\beta_1, \ldots, b\beta_{m_2}$ and where $\Lambda_1, \ldots, \Lambda_{m_1} = \Omega_1, \ldots, \Omega_{m_1}$ and $\Lambda_{m_1+1}, \ldots, \Lambda_{m_1+m_2} = \Gamma_1, \ldots, \Gamma_{m_1+m_2}$.

So we have that $au + bv \in \mathcal{M}$. □

Now, we know that because we can approximate any continuous function by a step functions, $\mathcal{M}$ is not a closed set. We show this more rigorously in the next two theorems. We also show that this set is not open.

Theorem 2.2. *(1-D) The set of minimizers, $\mathcal{M}$, of (2.5) is neither open nor closed in $\mathcal{B}_1(\Omega)$.*

Proof. First, we show that $\mathcal{M}$ is not closed. We do this by considering the sequence, $\{u_n\} \in \mathcal{M}$ given by the following pattern (see Figure 2.2)

$$u_0(x) = \begin{cases} 0 & x \in [0, \frac{1}{2}) \\ \\ 1 & x \in [\frac{1}{2}, 1] \end{cases}$$

$$u_n(x) = \begin{cases} 0 & x \in [0, \frac{1}{2^{n+1}}) \\ \\ \frac{j}{2^n} & x \in [\frac{2j-1}{2^{n+1}}, \frac{2j+1}{2^{n+1}}), \quad j = 1, \ldots, 2^n - 1 \quad , \quad n > 0 \\ \\ 1 & x \in [\frac{2^{n+1}-1}{2^{n+1}}, 1] \end{cases}$$

It is easily seen that $u_n \to x$ as $n \to \infty$, but $u(x) = x$ is not a minimizer of (2.5) since

$$\int_0^1 |u'(x)|_\beta^p \, dx = \int_0^1 \left((u'(x))^2 + \beta^2\right)^{p/2} \, dx = \int_0^1 \left(1 + \beta^2\right)^{p/2} \, dx = (1+\beta^2)^{p/2} > \beta^p = \min_{u \in \mathcal{B}_1([0,1])} \int_0^1 |u'|_\beta^p \, dx.$$

Thus, $\mathcal{M}$ is not closed.

Now, we need to show that $\mathcal{M}$ is not open. Suppose, to the contrary, that $\mathcal{M}$ is open. Let $u \in \mathcal{M}$ Then for every ball, B_u, centered at u, $B \subset \mathcal{M}$. But, by the argument given in Lemma 2.1, above, we can see that for h small, there is a $u_h \in B \setminus \mathcal{M}$, a contradiction. Thus, $\mathcal{M}$ is not open. □

Theorem 2.3. *(2-D) The set of minimizers of (2.7) is neither open nor closed in $\mathcal{B}_2(\Omega)$.*

Proof. We first show that the set of minimizers is not open. Using the argument for 2.1, we see that we can approximate a minimizer, u^*, by continuous functions that have the same value as u^* on $\Omega_i \setminus \mathcal{N}_\epsilon(\partial\Omega_i)$, where $\mathcal{N}_\epsilon(\partial\Omega_i)$ is an ϵ neighborhood of the boundary of the level set Ω_i. On this ϵ neighborhood, we use a linear approximation to approximate the jump between two level sets in the direction that is normal to $\partial\Omega_i$.

To show that $\mathcal{M}$ is not closed, we consider the sequence of functions which are flat on concentric rings, defined by (see Figure 2.3)

$$u_0(x) = \begin{cases} 0 & 0 \le |x| < \frac{1}{2}) \\ \\ 1 & \frac{1}{2} \le |x| \le 1 \end{cases}$$

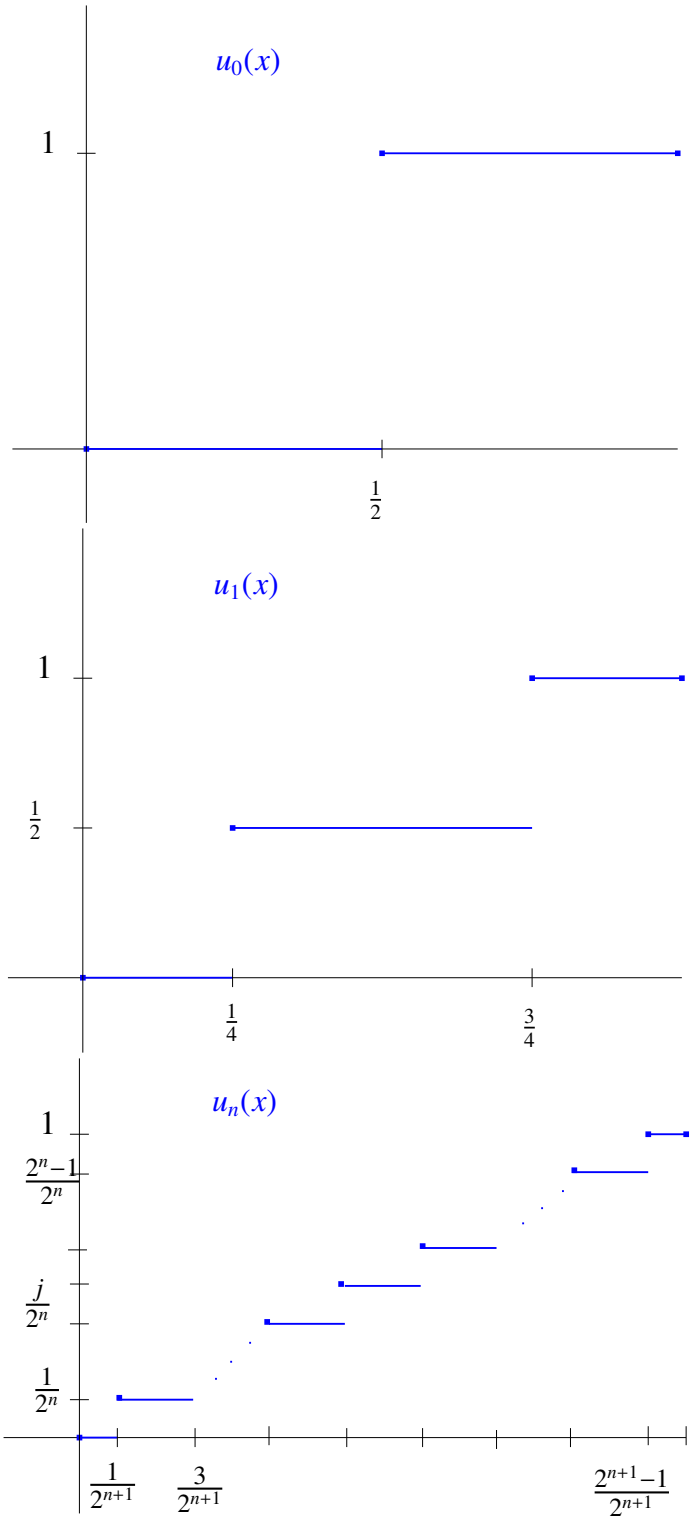


Figure 2.2: Sequence of step functions approaching a line.

$$u_n(x) = \begin{cases} 0 & 0 \leq |x| < \frac{1}{2^{n+1}} \\ \left(\frac{j}{2^n}\right)^2 & \frac{2j-1}{2^{n+1}} \leq |x| < \frac{2j+1}{2^{n+1}}, \quad j = 1, \ldots, 2^n - 1 \quad , \quad n > 0. \\ 1 & \frac{2^{n+1}-1}{2^{n+1}} \leq |x| \leq 1] \end{cases}$$

Clearly, this sequence of functions, $u_n : \mathbb{R}^2 \to \mathbb{R}$ tends to $u(x) = |x|^2$ which is not a minimizer. Thus, $\mathcal{M}$ is not closed. □

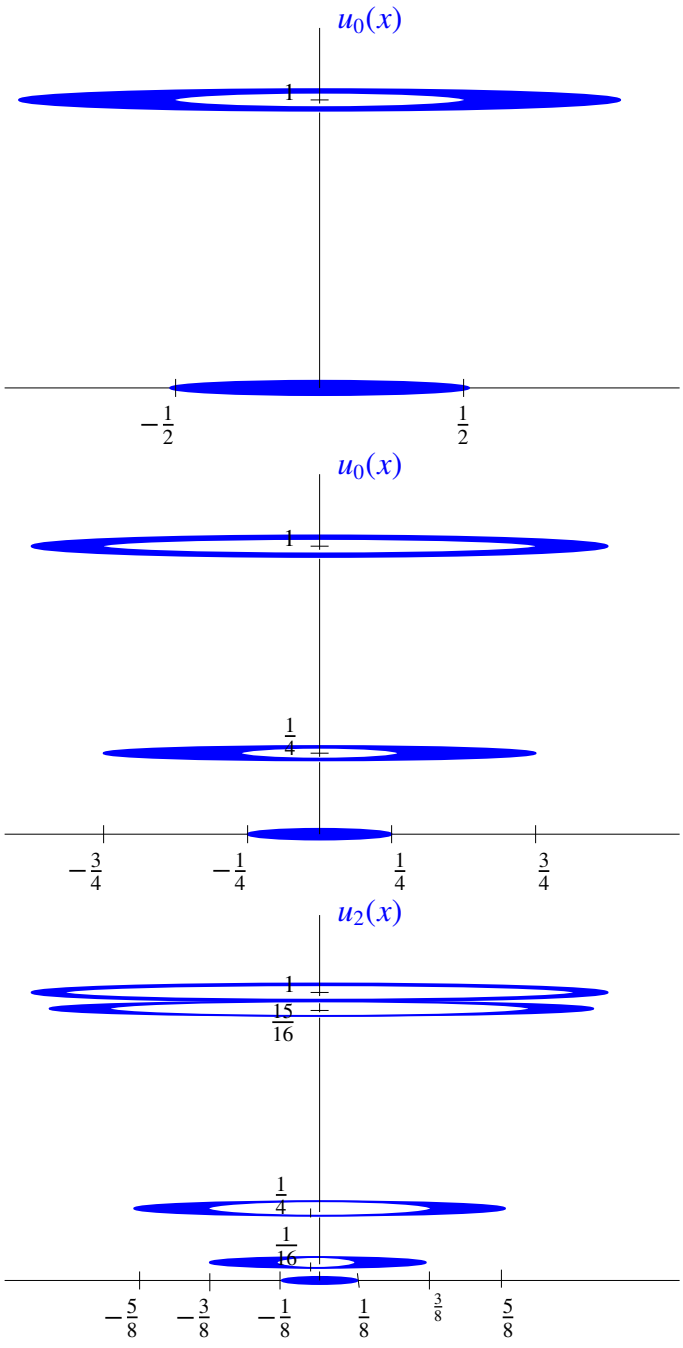


Figure 2.3: Sequence of functions that are flat on concentric rings and tend to $u(x) = |x|^2$

2.3 Minimizer Summary

In this chapter, we recognize that the functional in

$$\min_{u \in \mathcal{B}_n(\Omega)} \int_\Omega |\nabla u|^p \, dx \quad \text{for } 0 < p < 1 \tag{2.9}$$

is nonconvex. So, we cannot expect a unique minimizers. With that, we found that solutions of the minimization problem are step functions. We showed that this set is convex and neither open nor closed. Our plan is to compare these solutions to solutions of the p-Laplacian equation (1.12).

CHAPTER THREE

MONOTONE FUNCTIONS IN HIGHER DIMENSIONS

In exploring (1.3), we look for stationary solutions. In the case when the domain is 1-dimensional, we see that, for u to be a stationary solution, we must have

$$|u'|\left(\frac{u'}{|u'|}\right)' = (1-p)u''. \tag{3.1}$$

Clearly, any linear function u satisfies this condition. We know that the only strictly monotonic stationary solutions to the 1-dimensional p-Laplacian equation are functions whose graphs are lines. Indeed, if u is a monotone stationary solution to the 1-dimensional (1.3), then we get

$$0 = \left((u')^{p-2}u'\right)' = \left((u')^{p-1}\right)' = (p-1)(u')^{p-2}u''.$$

Since $u' \neq 0$ we get $u'' = 0$ thus, $u(x) = ax + b$ for some a and b.

Notice, also. that if $u : \Omega \subset \mathbb{R} \to \mathbb{R}$ is a stationary solution to (1.3), we get

$$|u'|^{p-2}u' = c \quad \Rightarrow \quad u \text{ is strictly monotone} \quad \Rightarrow \quad (u')^{p-1} = c \quad \Rightarrow \quad u' = c \quad \Rightarrow \quad u \text{ is linear.}$$

Above, we relax notation and allow the c to change throughout. This tells us that if u is a stationary solution to (1.3) then it is the affine function $u(x) = ax+b$. The next question is whether we can see similar results for functions with higher dimensional domains. But, we need a notion of monotone in higher dimensions. In this chapter, we take a detour to explore different definitions for monotone in higher dimensions and introduce our own.

3.1 Definitions and Examples

In [21], Lebesgue notes that on any interval in $\mathbb{R}$ a monotonic function f attains its maximum and minimum at the endpoints of this interval. This is the motivation he uses to define monotonic functions on an open bounded domain, $\Omega \subset \mathbb{R}^2$. His definition requires that these functions must attain their maximum and minimum values on the boundaries of the closed subsets of Ω. We state the definition found in [21] below.

Definition 3.1 (Lebesgue)**.** Let Ω be an open bounded domain. A continuous function $f : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ is said to be Lebesgue monotone if for every closed domain, $\Omega' \subseteq \Omega$, f attains its maximum, $\max_\Omega f$, and minimum, $\min_\Omega f$, values on $\partial\Omega'$.

Remark 3.1. This definition tells us that a function f is Lebesgue monotone if and only if every level set of f extends to the boundary.

Remark 3.2. Notice also that we can extend this definition to a function $f : \Omega \subset \mathbb{R}^n \to \mathbb{R}$.

We now give a couple of examples of functions that are Lebesgue monotonic.

Example 3.1. Since an n dimensional plane, $f(x) = c^T x + x_0$, can only take on extreme values on the boundary of any closed set in its domain, we know that it is Lebesgue Monotone.

Example 3.2. Let $\Omega = R(x, L)$ be the square of side length L, centered at a point $x \in \mathbb{R}^n$, for some $L > 0$. Any nD function whose level sets are lines is Lebesgue monotone. That is, any function of the form $f(x, y) = f(y)$ is Lebesgue monotone. Even a function that is not monotone when projected to its one dimensional domain is Lebesgue monotone, for example $f(x, y) = x^3 - x$ (see Figure 3.1). Because the function is constant in the y direction, we see that on the boundary of any closed subset of Ω, f must take on all the same values as it takes in the interior. Of course, the choice of Ω is somewhat arbitrary here (it need only be bounded).

We now move on to another definition given in [23]. Here Mostow, gives the following definition for monotone functions.

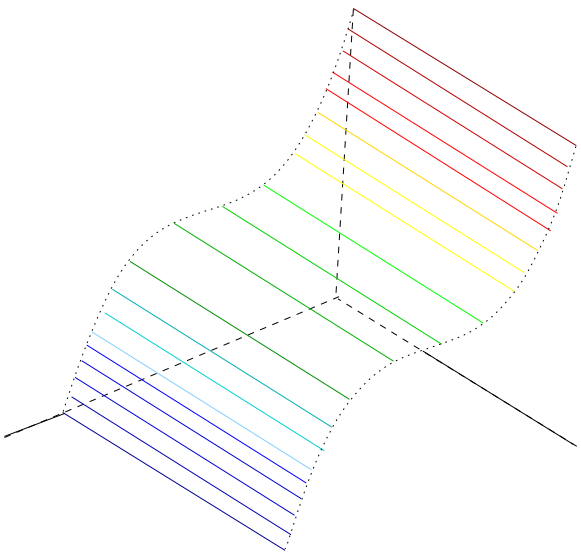


Figure 3.1: An example of a function that is Lebesgue monotone even though projecting to its 1D domain is not monotone.

Definition 3.2 (Mostow)**.** Let Ω be an open set in a locally connected topological space and let f be a continuous function on $\overline{\Omega}$. The function f is called Mostow monotone on Ω if for every connected open subset $U \subset \Omega$ with $U \neq \overline{U}$,

$$\sup_{x \in U} f(x) \leq \sup_{y \in \partial U} f(y) \quad \text{and} \quad \inf_{x \in U} f(x) \geq \inf_{y \in \partial U} f(y).$$

We see that if $\Omega = \mathbb{R}^2$ then we can choose a closed disk, $D_r = D(0, r)$ centered at the origin with radius r so that $U = \mathbb{R}^2 \setminus D_r$. On $\partial U = \partial D_r$ a function, f, that is Mostow monotone must obtain both its maximum and its minimum. But, we can let $r \searrow 0$. In doing this, we see that the maximum and minimum of f can be arbitrarily close. This tells us that if f is Mostow Monotone, then it must be a constant function. In [23], Mostow states that one can adjust this definition by requiring the function to take on its maximum or minimum on ∂U only for relatively compact open sets.

Example 3.3. It is not true that Lebesgue monotone functions are Mostow monotone (even if we follow the suggestion in [23] to adjust the definition of Mostow monotone). To see this, we consider a function $f : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ that is affine and has its gradient oriented along the domain

as in Figure 3.2. Here f will have supremum and infimum that are not attained on the boundary of the open set U.

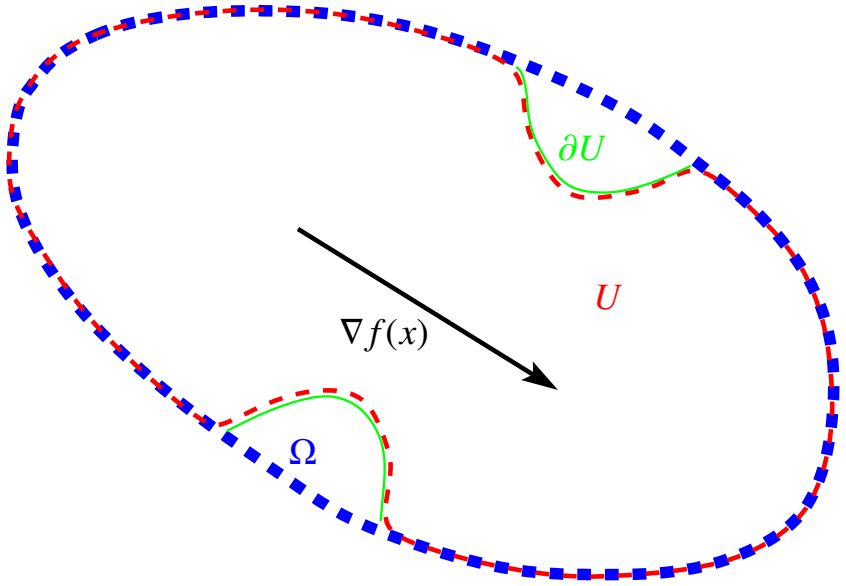


Figure 3.2: An example of a function that is Lebesgue monotone, but not Mostow monotone

Remark 3.3. Notice, if $\Omega \subset \mathbb{R}^2$ is a bounded domain then any continuous, Mostow monotone function is also Lebesgue monotone. This is true whether or not we are adjusting the definition as suggested in [23].

Before giving the next definition, we give some notation for clarity. Let $\Omega \subseteq \mathbb{R}^2$ be an open domain, $B(x, r)$ be the closed ball of radius r around the point $x \in \Omega$, and $S(x, r)$ be the boundary of the ball, $B(x, r)$. We say a function is $L^1_{loc}(\Omega)$ if $\int_U |u|\, dx < \infty$ for every bounded set $U \subset \Omega$. For comparison, we write the following definition for a less general function than what can be found in [31].

Definition 3.3 (Vodopyanov, Goldstein)**.** We say an L^1_{loc} function, $f : \Omega \to \mathbb{R}$ is Vodopyanov Goldstein Monotone at a point $x \in \Omega$ if there exists $0 < r(x) \leq \operatorname{dist}(x, \partial\Omega)$ so that for almost all $r \in [0, r(x)]$, the set

$$f^{-1}\left(f(B(x, r)) \cap [\mathbb{R} \setminus f(S(x, r))]\right) \cap B(x, r)$$

has measure zero. A function is then said to be Vodopyanov-Goldstein monotone on a domain, Ω if it is Vodopyanov Goldstein monotone at each point $x \in \Omega$.

Example 3.4. If we remove the continuity requirement for both Lebesgue and Mostow monotone functions we can create a function that is Mostow monotone but not Vodopyanov-Goldstein monotone. For the function in Figure 3.3, we see that any closed and bounded set must attain both the maximum and minimum of f on its boundary, but if we take a ball, B that contains the set $\{f = 0\}$, we see that $f(S) = \{-1, 1\}$. So, $f^{-1}(f(B \cap \mathbb{R} \setminus f(S))) \cap B$ does not have measure zero. That is, f is not Vodopyanov-Goldstein monotone.

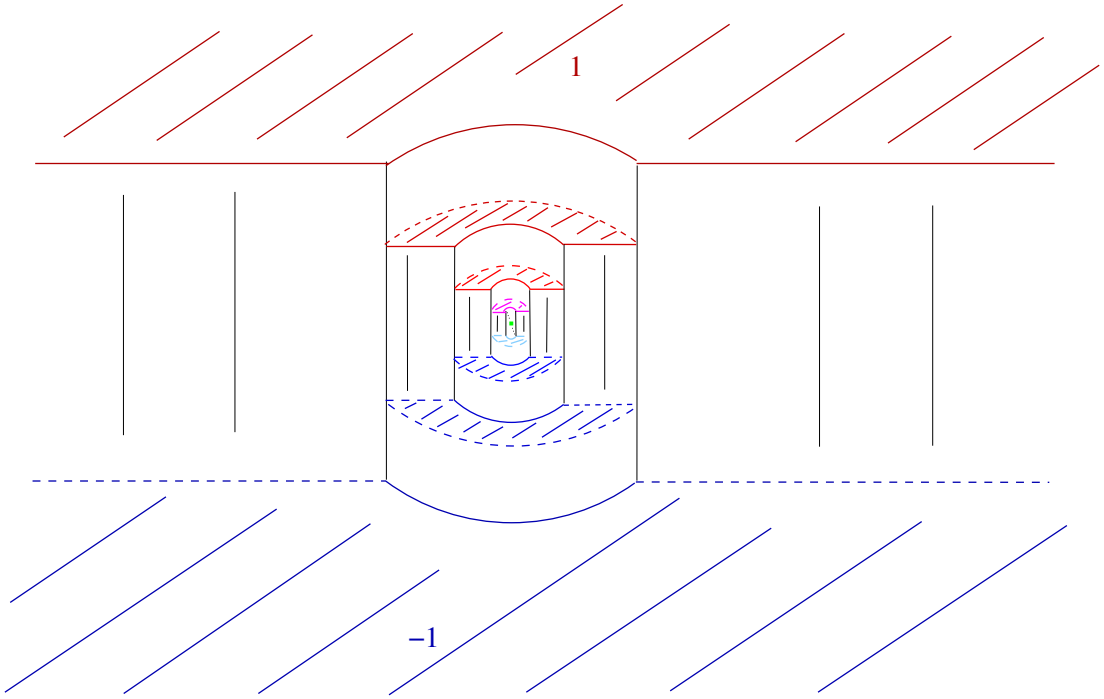


Figure 3.3: An example of a function satisfying all but continuity criteria for Mostow Monotone and is not Vodopyanov-Goldstein monotone.

Example 3.5. Now, a function can be Vodopyhanov-Goldstein monotone, but not Lebesgue monotone. An example of such a function is one in which f attains a minimum along a set, $\mathcal{M}$, that is long and narrow relative to the set Ω (see Figure 3.4). In this case, the boundary of any ball, $B(x, r) \subset \Omega$, that is centered along this set must intersect the set, $\mathcal{M}$ thus attaining both its maximum and minimum on the boundary of the ball, but the function will not reach its minimum on

the boundary of a closed set Ω' such as the one in Figure 3.4.

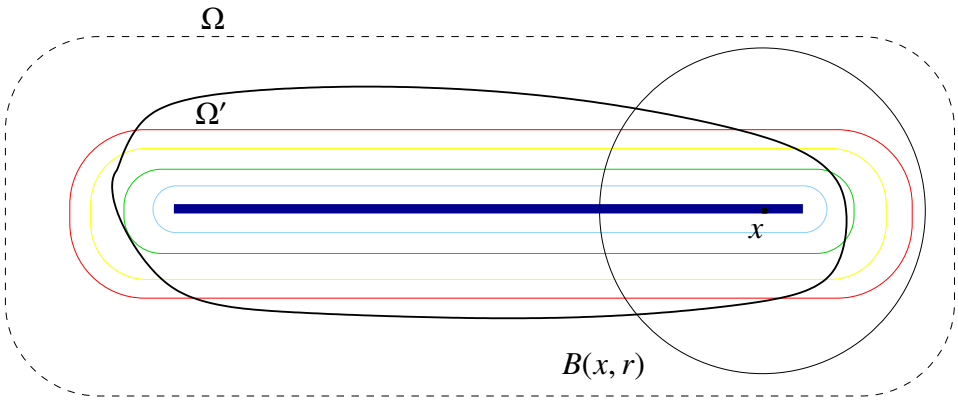


Figure 3.4: The level sets of an example function that is Vodopyhanov-Goldstein monotone but not Lebesgue monotone.

The next theorem shows that, for continuous functions, Lebesgue monotone functions are Vodopyanov-Goldstein monotone.

Theorem 3.1. *Let $\Omega \subset \mathbb{R}^2$ be a bounded domain and let $f : \Omega \to \mathbb{R}$ be continuous. Then f is Vodopyanov-Goldstein monotone function if f is Lebesgue monotone.*

Proof. Suppose f is Lebesgue monotone, then we know that for all closed sets $\Omega' \subset \Omega$, f attains its local extrema on $\partial\Omega'$. In particular, if we let $x \in \Omega$, we have that f attains its local extrema on the boundary of $B(x, r)$ for any $r > 0$. Let M and m be such that

$$M \equiv \sup_{y \in B(x,r)} f(y) \quad \text{and} \quad m \equiv \inf_{y \in B(x,r)} f(y).$$

Then we know that $f(B(x, r)) = (m, M)$ and $f(S(x, r)) = [m, M]$. So

$$\begin{aligned} & \mathbb{R} \setminus f(S(x, r)) = (-\infty, m) \cup (M, \infty) \\ \Rightarrow \quad & f(B(x, r)) \cap [(-\infty, m) \cup (M, \infty)] = \emptyset. \end{aligned}$$

Thus,

$$f^{-1}\left(f(B(x, r)) \cap [(-\infty, m) \cup (M, \infty)]\right) = \emptyset.$$

So, the measure of the set

$$B(x,r) \cap f^{-1}\left(f(B(x,r)) \cap [(-\infty, m) \cup (M, \infty)]\right)$$

is zero. Thus, f is Vodopyanov Goldstein monotone at x. Since x was chosen arbitrarily, f is Vodopyanov Goldstein monotone. □

In [22], Manfredi gives a definition for weakly monotone functions.

Definition 3.4 (Manfredi)**.** Let Ω be an open set in $\mathbb{R}^n$ and $f : \Omega \to \mathbb{R}$ be a function in $W^{1,p}_{loc}(\Omega)$. We say that u is weakly monotone if for every relatively compact subdomain $\Omega' \subset \Omega$ and for every pair of constants $m \leq M$ such that

$$(m - f)^+ \in W^{1,p}_0(\Omega') \quad \text{and} \quad (f - M)^+ \in W^{1,p}_0(\Omega'),$$

we have that

$$m \leq f(x) \leq M \quad \text{for a.e. } x \in \Omega'.$$

Manfredi also gives the following example of a function that is weakly monotone, but not continuous (in this case at the origin).

Example 3.6 (Manfredi)**.** Write $z = re^{i\theta}$ for $z \in \mathbb{R}^2$. Define u by

$$f(z) = \begin{cases} \theta & \text{for } 0 \leq \theta \leq \pi/2, \\ \pi/2 & \text{for } \pi/2 \leq \theta \leq \pi, \\ 3\pi/2 - \theta & \text{for } \pi \leq \theta \leq 3\pi/2, \\ 0 & \text{for } 3\pi/2 \leq \theta \leq 2\pi. \end{cases}$$

Because this function is not continuous, it does not satisfy the definition of Lebesgue or Mostow montone. We expect that all the above types of monotone functions should be weakly monotone.

Theorem 3.2. *Let $\Omega \subset \mathbb{R}^2$ be a bounded domain and $u : \Omega \to \mathbb{R}$, if u is Lebesgue monotone, then u is weakly monotone.*

Proof. Let $\Omega' \subset \Omega$, then by Definition 3.1, u is continuous and u attains its maximum and minimum on $\partial\Omega'$. Let m, M be a pair so that

$$(m-u)^+, (u-M)^+ \in W_0^{1,p}(\Omega'). \tag{3.2}$$

Since u is continuous so are $(m-u)^+$ and $(u-M)^+$. Thus, (3.2) gives us that

$$m \leq u \leq M \quad \text{ on } \partial\Omega'.$$

Thus, $m \leq \min_{x\in\Omega'} u(x) \leq u \leq \max_{x\in\Omega'} u(x) \leq M$. Thus, u is weakly monotone. □

Remark 3.4. Using Theorem 3.2 and Remark 3.3, we see that a function that is Mostow Monotone is also Weakly Monotone.

3.2 Normal Monotone, Cone Monotone, and K Monotone

In this section, we introduce a new definition of monotonicity which we call *Cone monotone*. We will discuss some variants of this new definition that we call *Normal monotone* and *K monotone*. We also characterize K monotone functions.

3.2.1 Cone Monotone

Motivated by the notion of monotone operators, we give a more general definition of monotonicity for functions in 2 dimensions. But first, we define the partial ordering, $\leq_K$ on $\mathbb{R}^2$.

Definition 3.5. Given a convex cone, $K \subset \mathbb{R}^2$ and two points $x, y \in \mathbb{R}^2$, we say that

$$x \leq_K y \quad \text{if} \quad y - x \in K. \tag{3.3}$$

Definition 3.6. We say a function $f : \Omega \subseteq \mathbb{R}^2 \to \mathbb{R}$ is cone monotone if at each $x \in \Omega$ there exists a cone, $K(x)$, so that

$$f(x) \leq f(y) \quad \text{whenever} \quad x \leq_{K(x)} y. \tag{3.4}$$

We say a function is K monotone if the the function is cone monotone with a fixed cone K.

Characterization of Cone Monotone

Here we first notice that a function that is K monotone cannot have any local extrema. This is stated more precisely in the following

Theorem 3.3. *Assume K is a convex cone with non-empty interior. If f is K monotone then there is no compact connected set M so that $f(M)$ is a local extremum.*

Proof. Suppose to the contrary. That is, suppose that $f(M)$ is a local minimum and suppose f is K monotone. Then we have for every point $x \in \partial M$ and every $y \in B_\varepsilon(x) \setminus M$, that $f(x) < f(y)$ (see Figure 3.5).

Pick $x \in \partial M$ so that the set $\{y \in M \cup B_\varepsilon(x) | y \leq_K x\} = \emptyset$, that is, we pick a point on the boundary so that the negative cone, $-K \equiv \{-x | x \in K\}$ does not intersect M near x. We know that if $\tilde{y} \in B_\varepsilon(x) \setminus M$ and $\tilde{y} - x \in -K$ then $x - \tilde{y} \in K$ so $f(x) \geq f(\tilde{y})$. Thus, we have a contradiction. □

Remark 3.5. Theorem 3.3 and Remark 3.1 give us that a continuous K monotone function is also Lebesgue monotone.

For the following discussion, we work in the graph space, $\mathbb{R}^{n+1}$ of a K monotone function $f : \mathbb{R}^n \to \mathbb{R}$. Assume a fixed closed, convex cone, K with non-empty interior. Set

$$\begin{aligned} \overline{K} &= K \times (-\infty, 0] \subset \mathbb{R}^{n+1} \\ \underline{K} &= -K \times [0, \infty) \subset \mathbb{R}^{n+1}. \end{aligned}$$

Let x denote the vector $(x_1, x_2, ..., x_n)$. We can translate these sections up to the graph of f so that

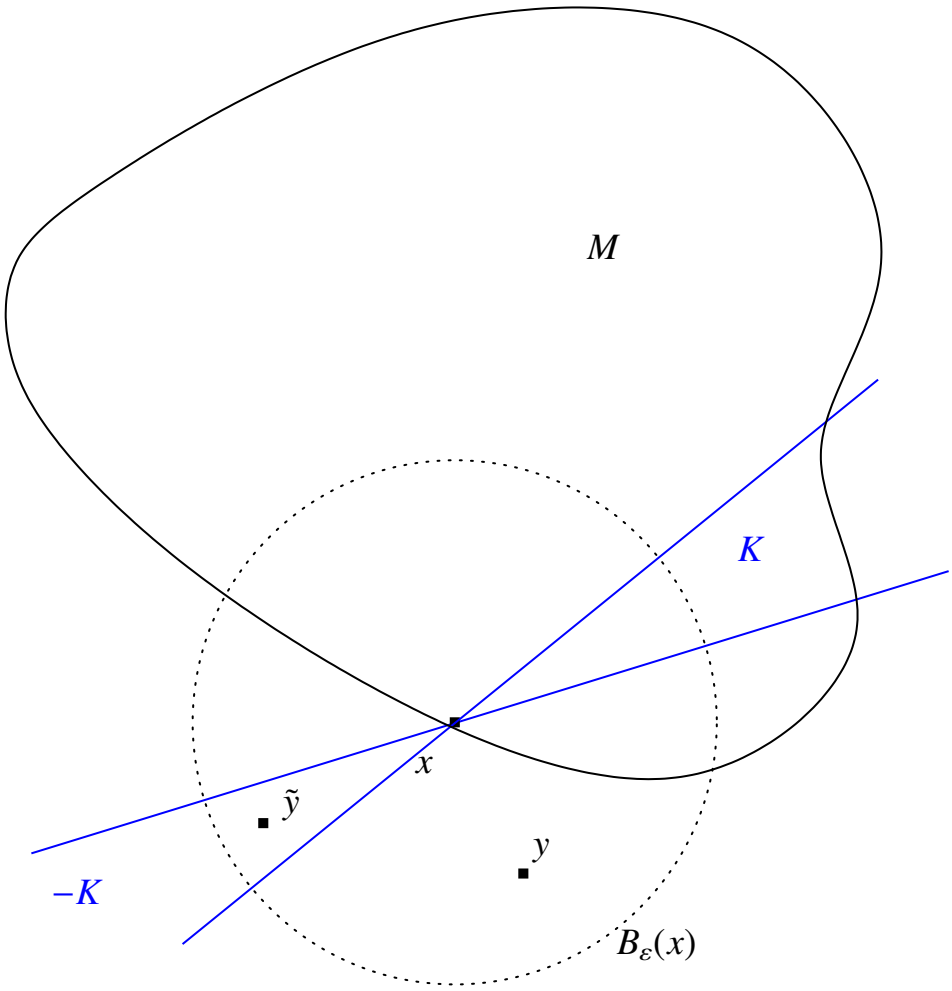


Figure 3.5: Cone monotone functions have no local extrema.

it touches at the point $(x, f(x))$. In doing this we see that we have (see Figure 3.6)

$$\begin{aligned} \overline{K} + (x, f(x)) &\subset \{(x, x_{n+1}) | x_{n+1} \leq f(x)\} \\ \underline{K} + (x, f(x)) &\subset \{(x, x_{n+1}) | x_{n+1} \geq f(x)\}. \end{aligned} \tag{3.5}$$

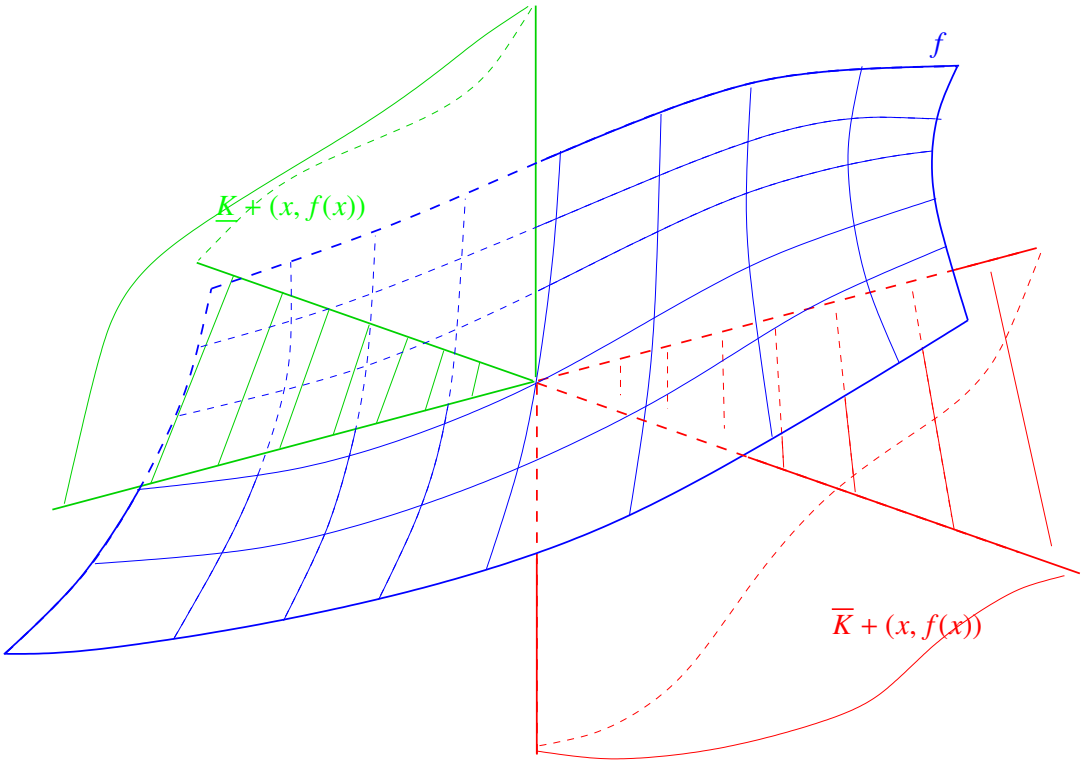


Figure 3.6: Example of $\overline{K} + (x, f(x))$ and $\underline{K}+(x, f(x))$.

We can do this for each point $(x, f(x))$ on the graph of f. Thus, the boundary of the epigraph and the boundary of the epograph are the same where we touch $\partial \text{epi} f$ with a translated $\overline{K}$ and $\underline{K}$. So, we can take the union of all such points to get

$$\begin{aligned} cl(\text{epi} f) &= \overline{\bigcup_{x \in \mathbb{R}^n} \underline{K} + (x, f(x))} \\ cl(\text{epo} f) &= \overline{\bigcup_{x \in \mathbb{R}^n} \overline{K} + (x, f(x))}. \end{aligned} \tag{3.6}$$

Care needs to be taken in the case when f has a jump discontinuity at x. Since for example, for an upper semicontinuous function epi$f$ does not contain points along the vertical section, $\{(x,r)|r \leq f(x)\}$, below the point $(x, f(x))$. Let

$$\mathcal{E} = \bigcup_{x\in\mathbb{R}^n} \underline{K} + (x, f(x)). \tag{3.7}$$

Using a limiting argument we notice that indeed this vertical section is contained in $\mathcal{E}$. If $(x,r) \in \{(x,r)|r \leq f(x)\}$, then we can find a sequence of points, $\{x_k\} \subset \mathbb{R}^n$ so that $x_k \to x$. Thus, for k large enough, $|(x_k, r) - (x, r)|$ is small. Thus, $cl(\mathcal{E}) = cl(\text{epi}f)$. A similar argument can be used to give the second equation in (3.6) for f lower semicontinuous. Using these two results, we get that (3.6) holds for any function f.

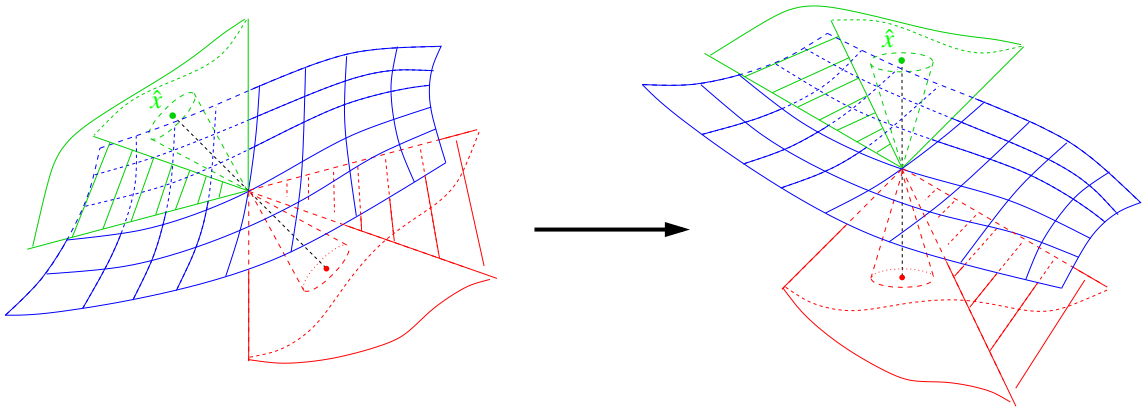


Figure 3.7: Rotating the graph of f so that the line segment from y to $\hat{x}$ becomes vertical

Picking $\hat{x} \in \underline{K}$ so that $B_\delta(\hat{x}) \subset \underline{K}$ and rotating so that $\hat{x}$ becomes vertical (see Figure 3.7), the piece of ∂epi$f$ in any $B_\delta(y)$, $y \in$ epi$f$ will be a Lipschitz graph with Lipschitz constant no more than $\left(\sqrt{||\hat{x}||^2 + \delta^2}\right)/\delta$. This implies that $\mu(\partial\text{epi f}) < \infty$ in any ball, that is, for $y \in \partial$epi, $\mu(\partial\text{epi}f \cap B(y,R)) < \infty$ for any R.

Theorem 3.4. *If f is K monotone and bounded, and K has non-empty interior then $f \in BV$.*

Proof. First, the slicing theorem from [20] gives us that

$$\int_{-\infty}^{\infty} (\partial\text{epo}f)_t \, dt < \mu(\partial\text{epo}f), \tag{3.8}$$

where $(\partial \text{epo} f)_t = \partial\{x | f(x) \geq t\}$. So we have that

$$\int_{-\infty}^{\infty} (\partial \text{epo} f)_t \, dt < \infty. \tag{3.9}$$

Using the coarea formula for BV functions from [15], we get that (3.9) implies that $f \in BV$. □

Examples of Cone Monotone Functions

We now consider some examples of K monotone functions.

Suppose K is a ray so that K has empty interior. Then for f to be K monotone all we need is for f monotone on all lines parallel to K, that is monotone in the positive direction of K. Therefore, f need not even be measurable.

Example 3.7. Let $f(\cdot, y) = rand(y)$, where $rand(y)$ assigns a particular random number to each value y. This function need not be measurable, but is K monotone with $K = \left\{ \alpha \begin{bmatrix} 0 \\ 1 \end{bmatrix} | \alpha > 0 \right\}$.

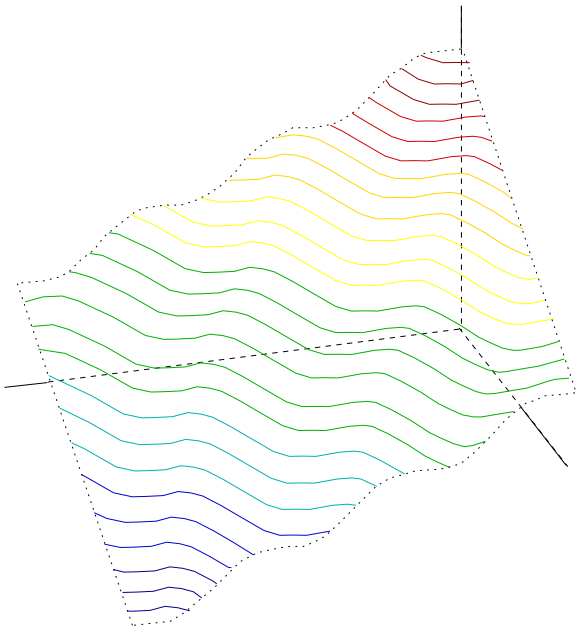


Figure 3.8: An example of a K monotone function with K having nonempty interior.

Example 3.8. An example of a K monotone function with the cone, K having nonempty interior is a function that oscillates, but is sloped upward (see Figure 3.8). More specifically, the function $f(x, y) = \sin(x) + x + y$ is K monotone. We can see this by noticing that f is increasing in the cone $K = \{(v_1, v_2) | v_1 > 0, v_2 > 0\}$.

Remark 3.6. Notice in this example that f has oscillatory behavior. Yet, f is still cone monotone. Notice also that if an oscillating function is tipped enough the result is K monotone. The more tipped the more oscillations possible and still be able to maintain K monotonicity.

Example 3.9. Some cone monotone functions are monotone in no other sense. An example of a function, $f : \mathbb{R}^2 \to \mathbb{R}$, that is Cone monotone, but not Vodopyanov Goldstein monotone is a paraboloid. At each point x, that is not the vertex of the paraboloid, we find the normal to the level set $\{f = f(x)\}$. We see the half space determined by this normal is a cone in which f increases from $f(x)$. At the vertex of the paraboloid, we see that all of $\mathbb{R}^2$ is the cone in which f increases.

Example 3.10. Not all Vodopyanov-Goldstein Monotone functions are cone monotone. An example of a function that is Vodopyanov-Goldstein Monotone, but is not cone monotone can be constructed with inspiration from Example 3.5. Level sets of this function are drawn in Figure 3.9. Here we see that the darkest blue level (minimum) set turns too much to be Cone monotone. We see this at the point y. At this point, there is no cone so that all points inside the cone have function value larger than $f(y)$ since any cone will cross the dark blue level set at another point.

Example 3.11. We can create a function, f that is Lebesgue Monotone, but is not Cone monotone. In this case, we need a level set that turns too much, but the level sets extend to the boundary of Ω. We see such a function in Figure 3.10. Let dark blue represent a minimum. Then at the point y, there is no cone that so that every point in the cone has function value larger than $f(y)$ since every cone will cross the dark blue level set.

Now if the domain has dimension higher than 2 and K is convex and has empty interior, but is not just a ray, then we can look at slices of the domain (see Figure 3.11). We can see that on

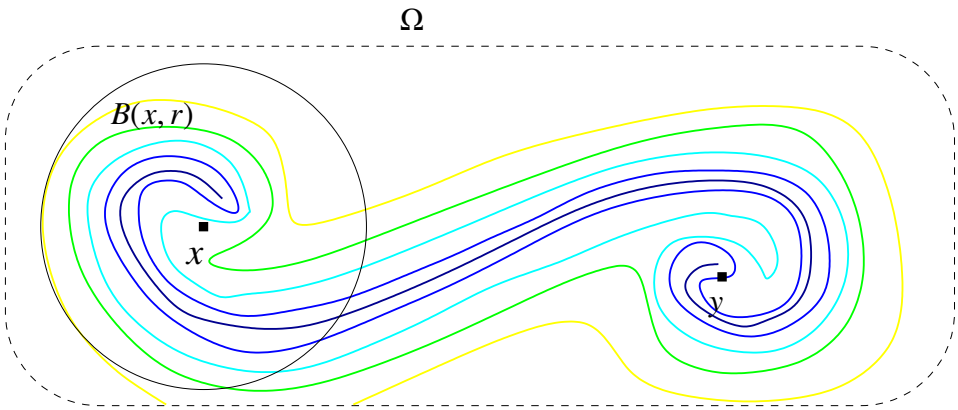


Figure 3.9: An example of a function that is Vodopyanov-Goldstein monotone, but is not Cone monotone.

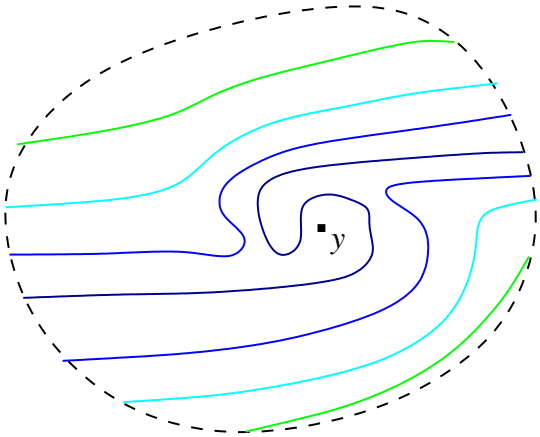


Figure 3.10: An example of a function that is Lebesgue monotone, but is not Cone monotone.

each slice of the domain, the function still satisfies Theorems 3.3 and 3.4. But, we also see that the behavior of the function from slice to slice is independent. This is the same behavior as we see when the function is defined on a 2-dimensional domain and K is a ray. That is, from line to line, the function behavior is independent (see Example 3.7). We can also see an example of the extended cones for a K monotone function where K is a ray, in Figure 3.12.

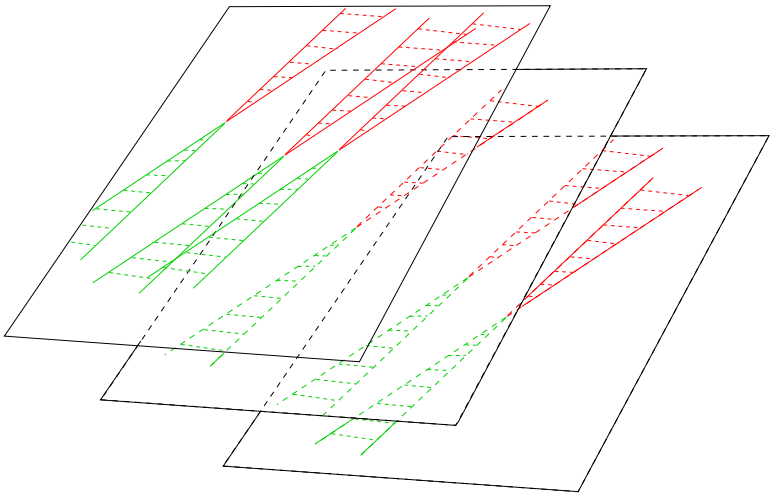


Figure 3.11: Cones with empty interior in a 3D domain that are not just rays.

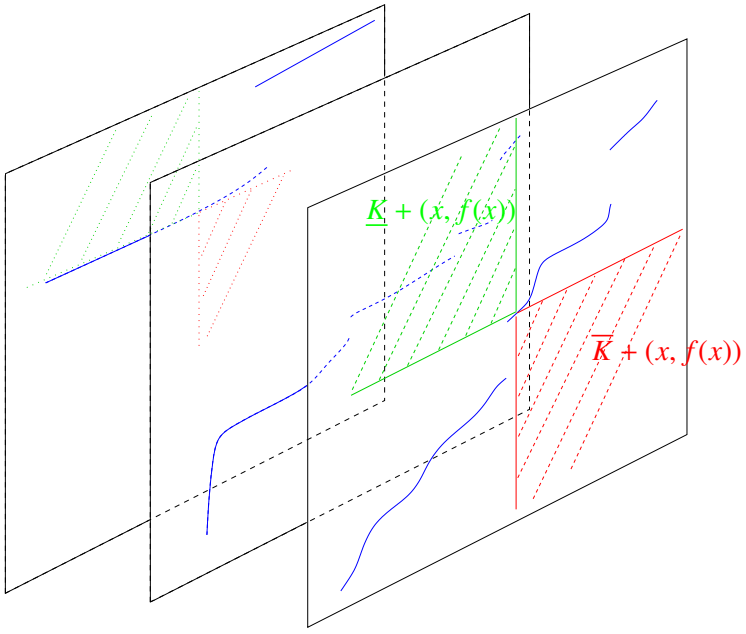


Figure 3.12: The extended cones are shown pinching the graphs of the functions, shown in blue. The key point is that the blue curve in each leaf of the foliation is independent of every other graph.

If K is a closed half space then f has level sets that are hyperplanes parallel to ∂K and f is one dimensional monotone.

Construction of K monotone functions

Recall from (3.6) that if f is K monotone, we have

$$\begin{aligned} cl(\text{epi}f) &= \overline{\bigcup_{x\in\mathbb{R}^n}(x, f(x)) + \underline{K}} \\ cl(\text{epo}f) &= \overline{\bigcup_{x\in\mathbb{R}^n}(x, f(x)) + \overline{K}} \end{aligned} \tag{3.10}$$

We can also construct a K monotone function by taking arbitrary unions of the sets $(x, x_{n+1})+\underline{K}$. By construction the boundary of this set is then the graph of the epigraph (and of the epograph) of a K monotone function.

Bounds on TV Norm

In this section, we find a bound on the total variation of K monotone functions. To do this we use the idea of a tipped graph introduced in Subsection 3.2.1.

Suppose $f < C$ on $\mathbb{R}^n$. Then $f\Big|_{B(0,R)\subset\mathbb{R}^n}$ has a graph that is contained in $B\left(0,\ \sqrt{R^2+C^2}\right) \subset \mathbb{R}^{n+1}$. Assuming that the tipped Lipschitz constant is $L\left(\leq \frac{\sqrt{\|x\|^2+\delta^2}}{\delta}\right)$, we get that the amount of $\partial\text{epi}f\Big|_{f|_{B(0,R)}}$ in $B\left(0,\ \sqrt{R^2+C^2}\right)$ is bounded above by $\alpha(n)\left(\sqrt{R^2+C^2}\right)^n \sqrt{1+L^2}$, where $\alpha(n)$ is the volume of the n dimensional unit ball.

Using the coarea formula discussed above, we get an upper bound on the total variation of a function that is K monotone as follows.

$$TV_{B(0,R)}(f) = \int_{B(0,R)} |\nabla u|\, dx \leq \mu(\partial\text{epi}(f(B(,R))) \leq \alpha(n)\left(\sqrt{R^2+C^2}\right)^n \sqrt{1+L^2}. \tag{3.11}$$

3.2.2 Normal Monotone

Motivated by the nondecreasing (or nonincreasing) behavior of monotone functions with domain in $\mathbb{R}$, we introduce a specific case of cone monotone. We consider a notion of monotonicity for functions whose domain is in $\Omega \subset \mathbb{R}^2$ by requiring that a monotone function be nondecreasing (or nonincreasing) in a direction normal to the level sets of the function.

First, we introduce a few definitions.

Definition 3.7. We say that a vector v is tangent to a set X at a point $x \in X$ if there is a sequence $\{x_k\} \subset X$ with $x_k \to x$ and a sequence $\{t_k\} \subset \mathbb{R}$ with $t_k \searrow 0$ so that

$$\lim_{k\to\infty} \frac{x_k - x}{t_k} = v. \tag{3.12}$$

The set of all tangents to the set X at the point $x \in X$ is the tangent cone and denote it by $T_X(x)$.

Definition 3.8. We say that a vector $n(x)$ is normal to a set X at a point $x \in X$ if for every vector $v \in T_X(x)$ we have that $n(x) \cdot v \leq 0$.

Definition 3.9. We say that a function $f : \mathbb{R}^2 \to \mathbb{R}$ is (strictly) Normal monotone if for every $c \in \mathbb{R}$ and every x on the boundary of the level set $\{f = c\}$ the 1-dimensional functions $\gamma \mapsto f(x + \gamma n(x))$ are (strictly) monotone for every vector, $n(x)$, normal to the level set $\{f = c\}$ at x.

Remark 3.7. The definition for normal monotone requires that the function be monotone along the entire intersection of a one dimensional line and the the domain of f. In the case of cone monotone, we require only monotonicity in the direction of the positive cone while in the case of K monotone, the fact that we can look forwards and backwards to get non-decreasing and non-increasing behavior follows from the invariance of K, not the definition of cone monotone.

Remark 3.8. A smooth function that is normal monotone is cone monotone for any cone valued function $K(x) \subset N(x)\ \forall x$.

We now explore this definition with a few examples.

Example 3.12. One can easily verify that a function whose graph is a non-horizontal plane is strictly normal monotone. This is desirable since a 1D function whose graph is a line is strictly monotone (assuming it is not constant).

Example 3.13. A parabola is not monotone in 1D so neither should a paraboloid be normal monotone. One can easily verify this to be the case.

Example 3.14. If we extend a nonmonotone 1D function to 2D, we should get a function that is not normal monotone. An example of such a function is the function $f(x, y) = x^3 - x$. Notice, this function is Lebesgue monotone, but neither K nor Normal monotone.

Example 3.15. In Figure 3.13, we show a function whose level sets are very oscillatory so that it is not normal monotone, while still being K monotone.

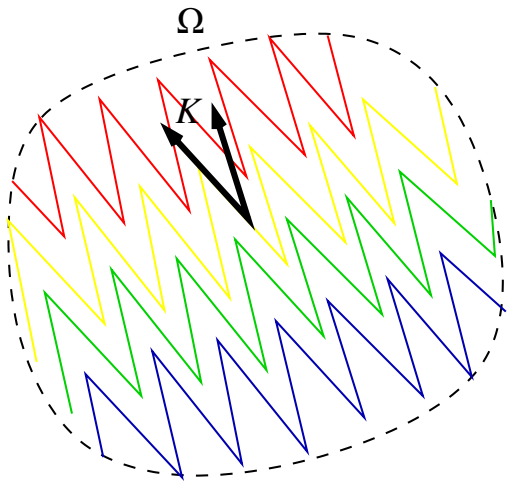


Figure 3.13: An example of a function that is K monotone, but not Normal monotone.

Example 3.16. In Figure 3.14, we see that if Ω is not convex, then we can construct a function that is not K monotone, but is Normal monotone. In this example, the function increases in a counter clockwise direction. This function is Normal monotone. We can see that it is not K monotone since at the point x any direction pointing to the north and west of the level line is a direction of ascent. But, at the point y, these directions are exactly the directions of descent. So the only cone of ascent

at both x and y must be along the line parallel to their level curves. But, we see that at other points, such as at z and w, we find the directions of ascent to be along the line containing these two points which does not coincide with the directions of ascent for x and y. Thus, this function cannot be K monotone.

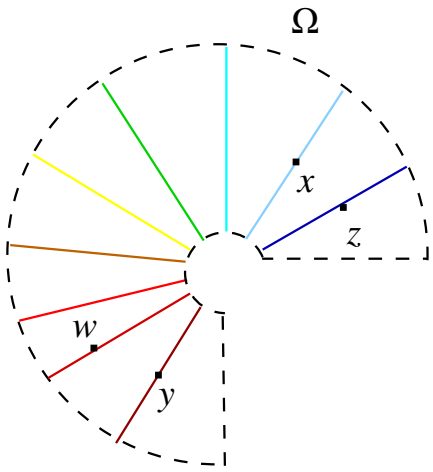


Figure 3.14: An example of a function that is not K monotone, but is Normal monotone.

The next theorem tells us that a normal monotone function is also Lebesgue monotone.

Theorem 3.5. *Let $\Omega \subset \mathbb{R}^2$ be a bounded domain and let $f : \Omega \to \mathbb{R}$ be a continuous, normal monotone function then f is also Lebesgue monotone.*

Proof. We prove the contrapositive. Suppose f is not Lebesgue Monotone. Then there exists a set Ω' so that

$$\inf_{x\in\Omega'} f(x) < \inf_{x\in\partial\Omega'} f(x).$$

We want to show that f is not normal monotone. Let us then define the nonempty set $M \subset (\Omega')^\circ$ to be the set where f attains a local minimum, at every point in M. That is,

$$M = \left\{ x \in \Omega' | f(x) = \inf_{x\in\Omega'} f(x) \right\}.$$

Let $(x_0, y_0) \in \partial M$ and let $n(x_0, y_0)$ be a normal at (x_0, y_0) to M. We know then that $\gamma \mapsto f((x_0, y_0) + \gamma n(x_0, y_0))$ is not monotone since f has a local minimum on M. Thus f is not normal monotone. □

Remark 3.9. This theorem gives us that a function that is normal monotone is also weakly monotone.

3.3 Monotone Summary

In this chapter we explored current and new definitions of monotone. For continuous functions, we compared several definitions of monotonicity, in higher dimensions. How these sets of functions are related is represented in the following Venn diagram.

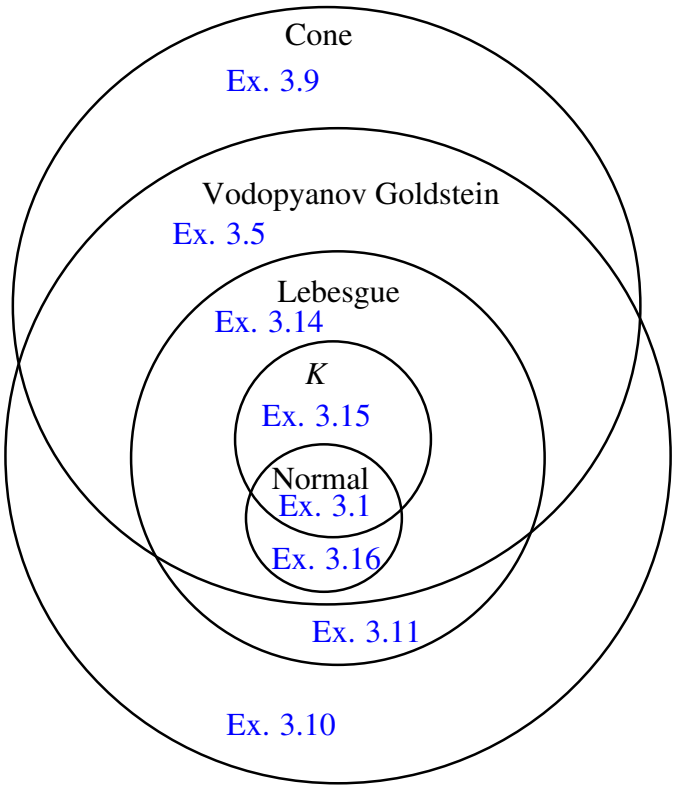


Figure 3.15: Types of monotonicity in higher dimensions and how they compare for a continuous function.

We also showed how to construct K monotone functions. We show that bounded K monotone functions are BV and we find a bound on the total variation of these functions.

CHAPTER FOUR

STATIONARY SOLUTIONS OF THE p-LAPLACIAN

4.1 The p-Laplacian and Its Difficulties When $0 < p < 1$

A common approach to finding minimizers of (2.1), with $\beta = 0$,

$$\min_{u \in \mathcal{B}_n(\Omega)} \int_\Omega |\nabla u|^p \, dx \quad \text{for } 0 < p < 1, \tag{4.1}$$

is to find solutions to the Euler-Lagrange equation. The idea is that we find stationary solutions when its variation is zero. The solutions to the Euler-Lagrange equation are precisely those points where the variation of the functional is zero. Here, we consider this approach and we compute the Euler-Lagrange equation of

$$\int_\Omega \mathcal{L}(\nabla u) \, dx \equiv \int_\Omega |\nabla u|^p \, dx. \tag{4.2}$$

So, for $\mathcal{L}(z) = |z|^p$, we get that the Euler-Lagrange equation is

$$0 = -\nabla \cdot \mathcal{L}_z(z). \tag{4.3}$$

That is,

$$0 = -\nabla \cdot \left(|\nabla u|^{p-2} \nabla u \right) \equiv \Delta_p u, \tag{4.4}$$

the p-Laplacian. The difficulty in finding minimizers of (4.1) with this method is that (4.1) is nonconvex and so the results from the Calculus of Variations do not guarantee that solutions to (4.4) are minimizers of (2.1) (see [11, 12, 14]). With that in mind, we still proceed to find solutions

to (4.4) as they are still stationary solutions to (4.1).

To solve this equation, we can seek stationary solutions to the p-Laplacian evolution equation

$$u_t = \nabla \cdot \left(|\nabla u|^{p-2} \nabla u \right). \tag{4.5}$$

Remark 4.1. We call (4.5) the p-Laplacian evolution equation, in the case $0 < p < 1$, without the factor $\frac{1}{p-1}$ as in [17].

We consider three types of solutions: viscosity, weak, and classical solutions. First, we define classical solutions.

Definition 4.1. We say u is a classical solution to the p-Laplacian equation (4.4) if $u \in C^2(\Omega)$ and u satisfies (4.4).

In Section 4.3, we find families of classical solutions so we will save the discussion until then.

Viscosity solutions.

Definition 4.2. [10] We say that $u : \Omega \to (-\infty, \infty]$ is a viscosity supersolution to the p-Laplacian equation (4.4) if

i u is lower semicontinuous,

ii $u \not\equiv \infty$, and

iii whenever $x_0 \in \Omega, \varphi \in C^2(\Omega)$ are such that $u(x_0) = \varphi(x_0), u(x) > \varphi(x)$ for $x \neq x_0$ and $\nabla\varphi(x_0) \neq 0$ we have $\Delta_p\varphi(x_0) \geq 0$.

If instead of i. and iii., we have

i′ u is upper semicontinuous

iii′ whenever $x_0 \in \Omega, \varphi \in C^2(\Omega)$ are such that $u(x_0) = \varphi(x_0), u(x) < \varphi(x)$ for $x \neq x_0$ and $\nabla\varphi(x_0) \neq 0$ we have $\Delta_p\varphi(x_0) \leq 0.$,

we call u a viscosity subsolution. And if both iii and iii′ hold, then we call u a viscosity solution.

Here we consider functions $\varphi \in C^2(\Omega)$ that touch u at a point x_0 and are everywhere else less than (or greater than) u. These functions also have a nonzero gradient at x_0. If all of these functions are then sub-(super-)solutions to (4.4), we say that u is a viscosity sub-(super-)solution of (4.4).

Notice that if we consider $u : \mathbb{R} \to \mathbb{R}$ to be an upper semicontinuous step function, then there are no points, x_0 and no functions φ so that $\varphi(x_0) = u(x_0), |\nabla\varphi(x_0)| \neq 0$ and φ stays below u everywhere else. In Figure 4.1, we see that if φ touches u at a point and is below u in a neighborhood of this point, $\varphi' = 0$ at that point. So any upper semicontinuous step function u is a viscosity

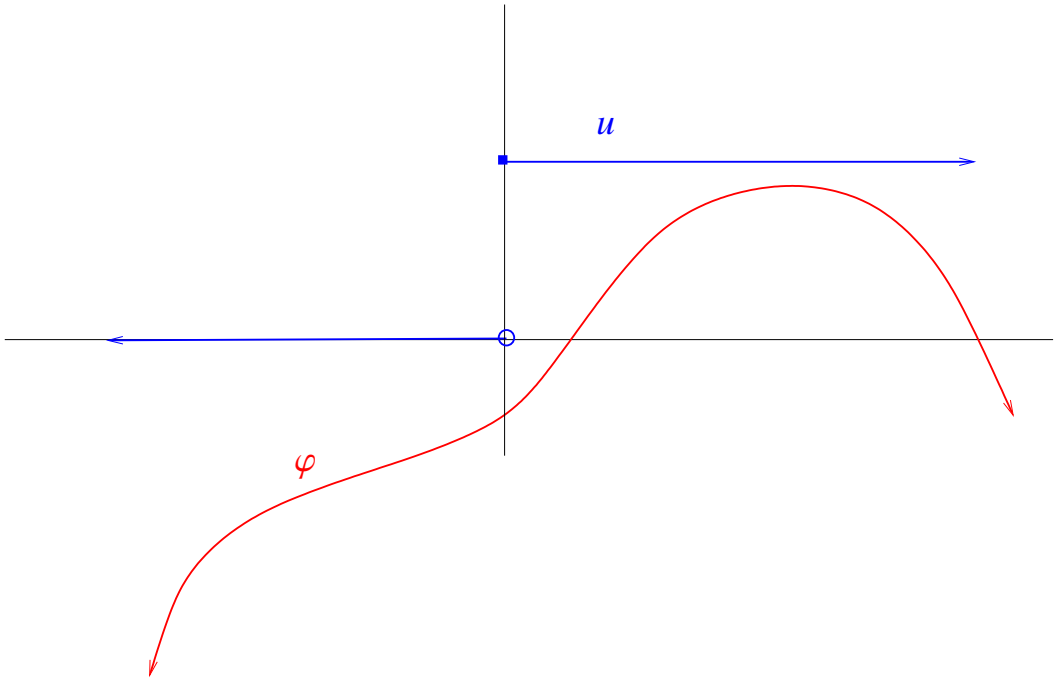


Figure 4.1: Upper semicontinuous step functions are viscosity solutions to (4.4)

subsolution to any PDE. A similar argument says that any lower semicontinuous step function is a viscosity supersolution to any PDE.

Notice that if we change the function $u : \mathbb{R} \to \mathbb{R}$ to be upper semicontinuous and piecewise linear as in Figure 4.2, u is still a viscosity subsolution since at any point x_0, if

$$\varphi \in C^2(\mathbb{R}), \varphi(x_0) = u(x_0), \text{ and } \varphi'(x_0) \neq 0. \tag{4.6}$$

then (say $u'(x_0) > 0$)

$$\Delta_p \varphi(x_0) = \left(|\varphi'(x_0)|^{p-1}\varphi'(x_0)\right)' = \left((\varphi'(x_0))^p\right)' = p(\varphi'(x_0))^{p-1}\varphi''(x_0) \leq 0 \tag{4.7}$$

Again, we can use a similar argument for lower semicontinuous u that are piecewise linear to say

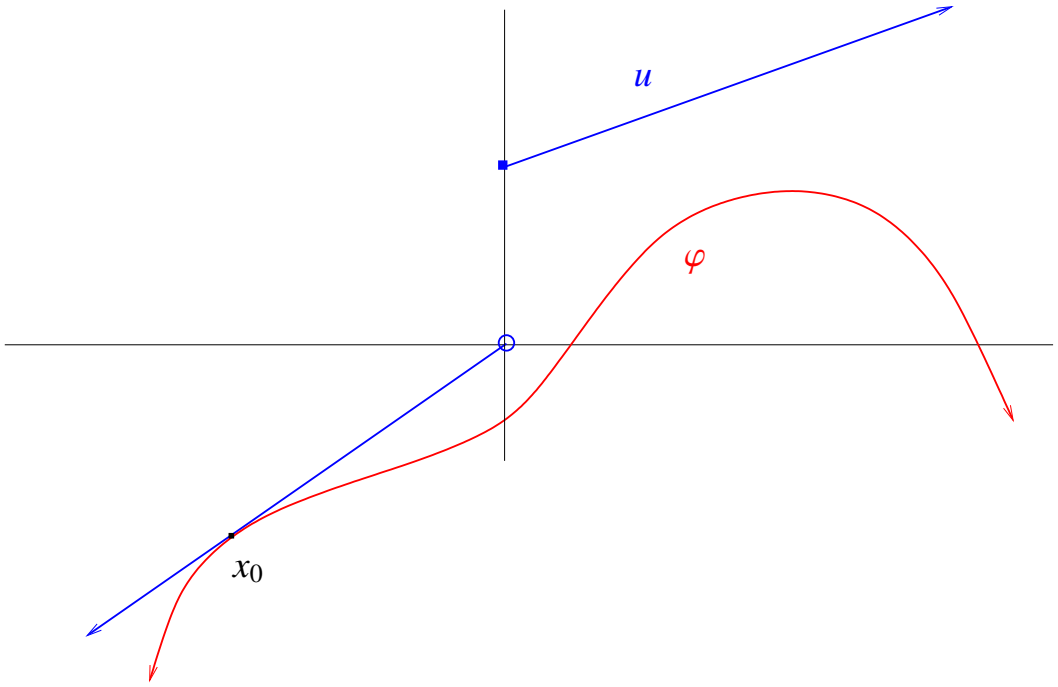


Figure 4.2: φ is a function satisfying $\varphi(x_0) = u(x_0), \varphi(x) < u(x)$ if $x \neq x_0$ and $\Delta_p \varphi(x_0) < 0$

they are super solutions of (4.4).

Now, if we make u continuous, we can find a function that is a subsolution, but is not a supersolution. Consider $u = -|x|$. We can find a function $\varphi \in C^2(\mathbb{R})$ so that $\varphi(0) = u(0)$, $\varphi(x) > u(x)$ for $x \neq 0$ and $\varphi'(0) \neq 0$, but $\Delta_p \varphi(0) < 0$ (see Figure 4.3). Notice that in a neighborhood of x_0, u is concave and therefore $u''(x_0) < 0$.

Notice if u is continuous but has any point, x_0, of nondifferentiability then u is not a viscosity solution because we can touch u at x_0 with a C^2 function, φ that curves toward u so that φ'' has the wrong sign as we did in Figure 4.3).

Now, we can try to create a pathological function u that is continuous and differentiable ev-

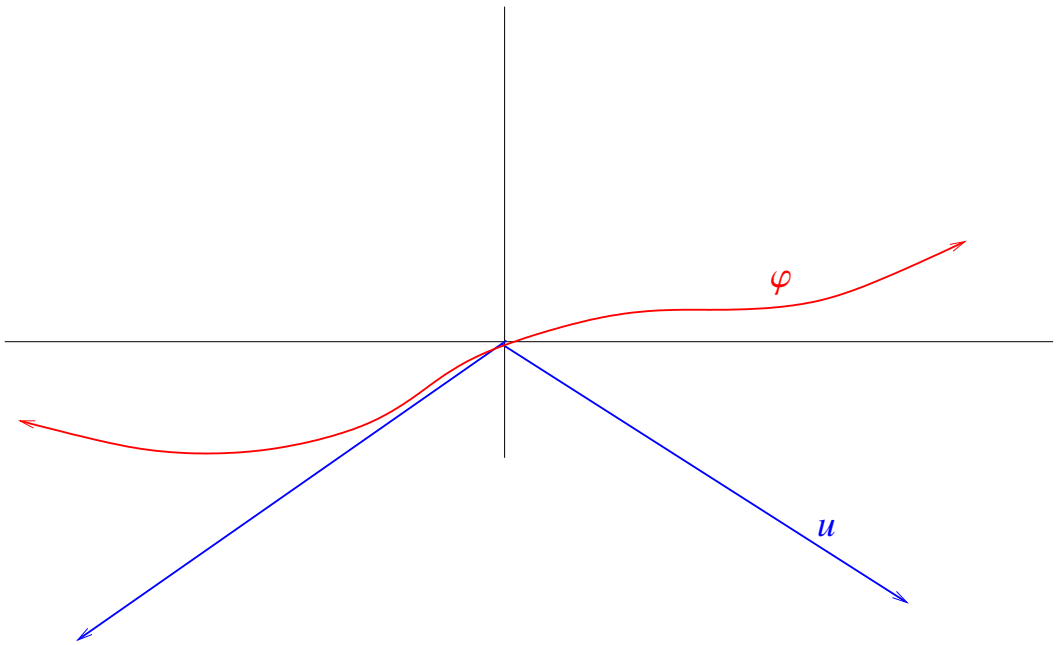


Figure 4.3: $u = -|x|$ is not a viscosity solution of (4.4)

erywhere, but not twice differentiable anywhere. This function can then possibly be a viscosity solution to (4.4). But, ignoring any pathological examples, if we consider a continuous function u that has any point x_0 where $u'(x_0) \neq 0$ and $u''(x_0) > 0$, we can then find a $\varphi \in C^2$ with $\varphi(x_0) = u(x_0)$ and $\varphi < u(x)$ for $x \neq x_0$ but has $\varphi'' > 0$ (see Figure 4.4). This tells us that for u to be a viscosity

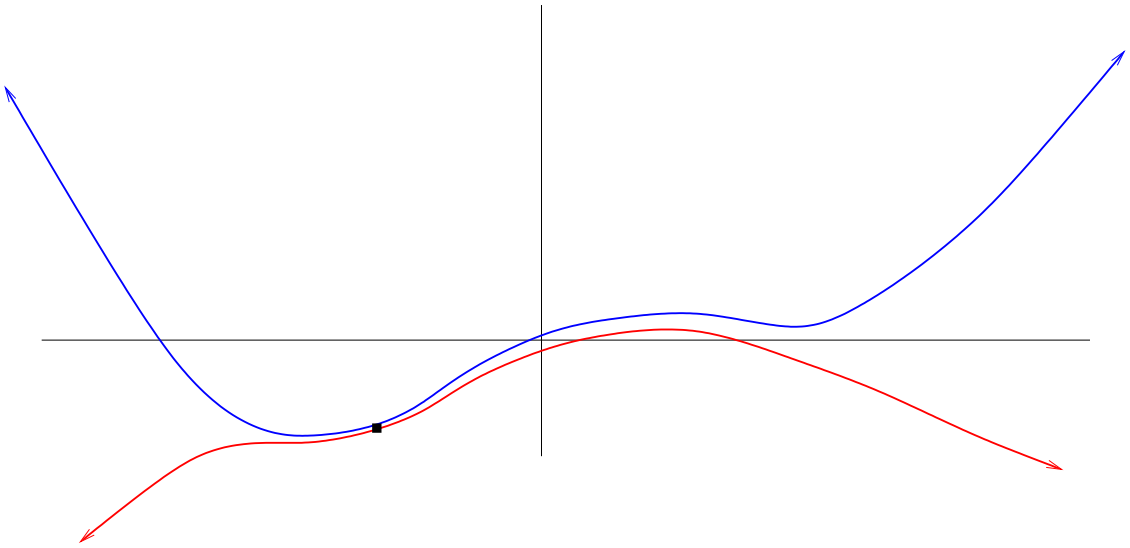


Figure 4.4: u is not a viscosity solution of (4.4) if at any point u'' exists and is not zero.

solution, wherever u'' exists, it must be zero. That is, for u to be a viscosity solution, it must be linear wherever u'' exists. So any viscosity solutions $u : \mathbb{R} \to \mathbb{R}$ of (4.4) that are not pathological, must be linear.

For functions $u : \mathbb{R}^2 \to \mathbb{R}$ we recognize that because we do not have a maximum principle, it does not make sense to talk about viscosity solutions. More exploration into obtaining a maximum principle is necessary to proceed in this discussion, but that is not the goal of this work.

Weak solutions.

In the literature, we see energy methods used to discuss uniqueness of weak solutions to initial value problems (see [14, 13]). Below we discuss the difficulties of such methods with this equation, but first, we define weak solutions.

Definition 4.3. We say u is a weak solution to the p-Laplacian equation (4.4) if $u \in W^{1,p}(\Omega)$ and u satisfies

$$\int_\Omega |\nabla u|^{p-2}\langle \nabla u, \nabla \varphi\rangle \, dx = 0, \quad \forall \varphi \in C_0^\infty(\Omega), \tag{4.8}$$

where we write ∇u and mean distributional derivative $\nabla u = (\frac{\partial u}{\partial x_1}, ..., \frac{\partial u}{\partial x_n})$ with all components p-summable.

We now attempt an energy method to explore the uniqueness of solutions to the initial value problem

$$\begin{cases} u_t = \Delta_p u(x), & (x,t) \in \Omega \times (0,\infty) \\ u(x,0) = u_0(x), & x \in \Omega. \end{cases} \tag{4.9}$$

Let $u, \tilde{u}$ both be solutions to (1.3) and let $w = u - \tilde{u}$. And, set

$$e(t) \equiv \int_\Omega w^2 \, dx. \tag{4.10}$$

Then we can differentiate with respect to t to get

$$e'(t) = \int_\Omega 2w_t w \, dx. \tag{4.11}$$

Now, if w is also a solution to $\Delta_p u = 0$ with initial value $w(x, 0) = 0$, then we can write

$$e'(t) = 2\int_\Omega w\nabla \cdot \left(|\nabla w|^{p-2}\nabla w\right) \, dx = -2\int_\Omega |\nabla w|^p \, dx \leq 0. \tag{4.12}$$

And we would be able to conclude that $e(t) \leq e(0) = 0$ so $w = 0$, but because the p-Laplacian has the extra coefficient $|\nabla u|^{p-2}$, we cannot follow the steps in (4.12). So we are not able to use energy methods.

Because $p < 1$, we are unable to get regularity results using the standard techniques (again, see [14, 13]). We can see this because as $|\nabla u|$ gets closer to 0, the coefficient $|\nabla u|^{p-2}$ tends toward infinity. To deal with such singularities, it is typical to consider finding bounds on the weak form of the equation. We consider

$$\left|\int_\Omega |\nabla u|^{p-2}\langle \nabla u, \nabla \varphi\rangle \, dx\right| \leq \int_\Omega |\nabla u|^{p-1}|\nabla \varphi| \, dx \leq C\int_\Omega |\nabla u|^{p-1} \, dx, \tag{4.13}$$

where $C = \|\varphi\|_\infty$. Notice, we cannot find an upper bound on this integral since the exponent $p - 1$ is negative. So, the integral may tend upward to infinity.

Because solutions will blow up whenever the domain contains a point where $|\nabla u| = 0$, we are unable to get a maximum principle and therefore we are unable to get a Harnack inequality that tells us that the values of u on a domain are comparable (see Chapter VII of [13]). So, in this work, we take a more geometric approach to find classical solutions.

4.2 Decomposition and Geometric Interpretation

In an attempt to find solutions to (4.4) we sought understanding of the geometry of $\Delta_p u$. In this section, we describe the connection between $\Delta_p u$ and curvature. We also explore the effects of some example function in the evolution given by (4.5).

4.2.1 Curvature

We can see the relationship of the p-Laplacian to curvature by breaking it into two terms. We now rewrite (4.4) as

$$\begin{aligned} 0 &= \Delta_p u \equiv \nabla \cdot \left(|\nabla u|^{p-1} \frac{\nabla u}{|\nabla u|} \right) = |\nabla u|^{p-1} \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) + \frac{\nabla u}{|\nabla u|} \nabla |\nabla u|^{p-1} \\ &= |\nabla u|^{p-1} \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) + (p-1) |\nabla u|^{p-2} \nabla (|\nabla u|) \cdot \nabla u. \end{aligned}$$

Factoring out $|\nabla u|^{p-1}$ and using the computation in Equation (A.39), we get

$$0 = |\nabla u|^{p-1} \left[\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) + \frac{p-1}{|\nabla u|} \frac{\nabla u}{|\nabla u|}^T D^2 u \frac{\nabla u}{|\nabla u|} \right]. \tag{4.14}$$

Curvature of level sets.

The first term is the curvature of the level sets, call this curvature κ_{ls}. Indeed, we know that the unit normal to a level set is given by

$$N = \frac{\nabla u}{|\nabla u|}.$$

We find the curvature of the level set by finding how these unit normals diverge as we trace along the level set. That is, have

$$\kappa_{ls} = \nabla \cdot N = \nabla \cdot \frac{\nabla u}{|\nabla u|}.$$

Thus, we conclude that the first term is κ_{ls}.

Curvature in the direction of the gradient.

The second term is related to the curvature of u in the direction of ∇u, call this curvature κ_g. To verify, we compute this curvature. We know that the curvature along some curve measures how the unit tangent T changes as we trace along that curve. Let $\alpha_0 \in \mathbb{R}^n$ and let us define, as in figure 4.5, the line, $\alpha(t)$ in $\mathbb{R}^n$ passing through α_0 parallel to the gradient of u, the curve, $\beta(t)$, in the graph space of u, and the unit tangent, $T(t)$ along this curve,

$$\alpha(t) = \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|} t + \alpha_0, \tag{4.15}$$

$$\beta(t) = (\alpha(t), u(\alpha(t))), \tag{4.16}$$

$$T(t) = \frac{\beta'(t)}{|\beta'(t)|}. \tag{4.17}$$

To find the curvature of $\beta(t)$, we differentiate (4.17):

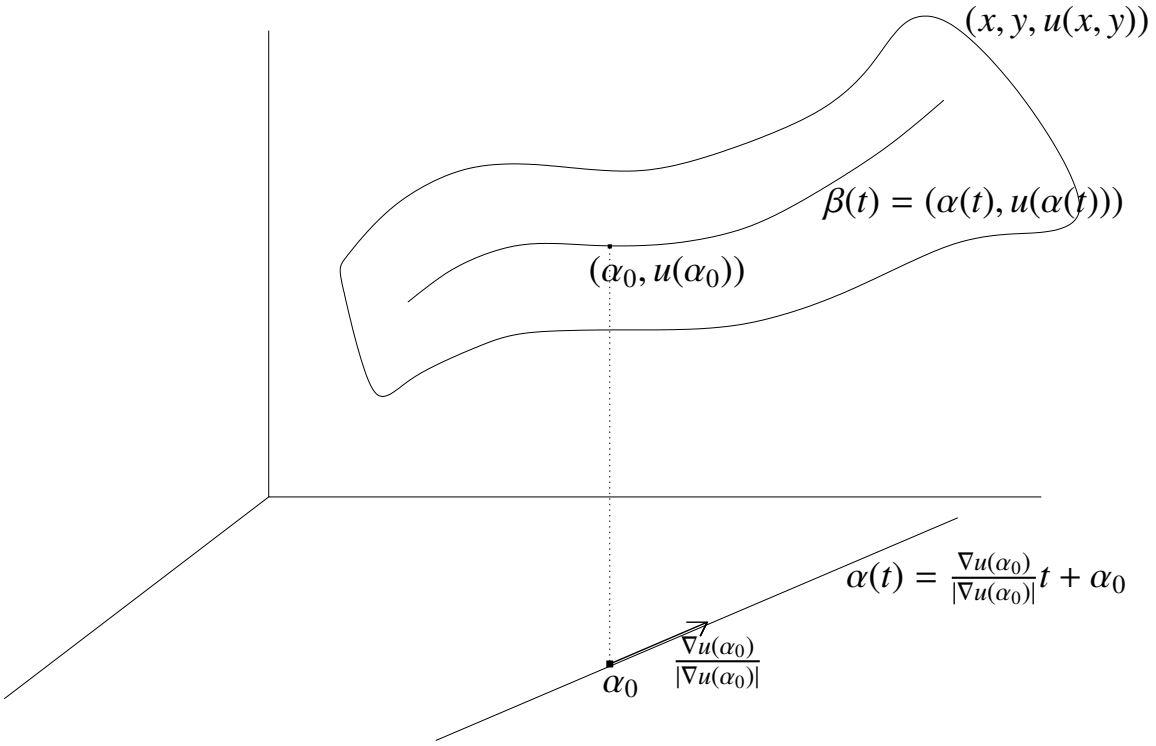


Figure 4.5: Curvature in the direction of the gradient.

$$T'(t) = \frac{\beta''(t)|\beta'(t)| - \beta'(t)(|\beta'(t)|)'}{|\beta'(t)|^2}. \tag{4.18}$$

We want to find the rate at which the tangent vector changes, at the point $(\alpha_0, u(\alpha_0))$, so we seek $\frac{T'(0)}{|\beta'(0)|}$ (we divide by $|\beta'(t)|$ to remove the change due to the speed of the parameterization). To compute this, we differentiate equations (4.15) and (4.16),

$$\begin{aligned}
\alpha'(t) = & \quad \tfrac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}, \\
\beta'(t) = & \quad (\alpha'(t), \nabla u(\alpha(t)) \cdot \alpha'(t)) \\
= & \quad \left(\tfrac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}, \nabla u(\alpha(t)) \cdot \tfrac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}\right),
\end{aligned}$$

Thus,

$$|\beta'(t)| = \frac{1}{|\nabla u(\alpha_0)|}\left(|\nabla u(\alpha_0)|^2 + (\nabla u(\alpha(t)) \cdot \nabla u(\alpha_0))^2\right)^{1/2} \tag{4.19}$$

Differentiating (4.19) gives

$$(|\beta'(t)|)' = \frac{1}{|\nabla u(\alpha_0)|} \frac{\nabla u(\alpha(t)) \cdot \nabla u(\alpha_0) D^2 u(\alpha(t)) \nabla u(\alpha_0)}{(|\nabla u(\alpha_0)|^2 + (\nabla u(\alpha(t)) \cdot \nabla u(\alpha_0))^2)^{1/2}}$$

Finally, we also compute the second derivative of β to get

$$\beta''(t) = \left(0, \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}^T D^2 u(\alpha(t)) \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}\right).$$

Evaluating each of the above at $t = 0$ gives

$$\begin{aligned}
\beta'(0) = & \quad \left(\tfrac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}, |\nabla u(\alpha_0)|\right), \\
|\beta'(0)| = & \quad \left(1 + |\nabla u(\alpha_0)|^2\right)^{1/2} \\
(|\beta'(0)|)' = & \quad \tfrac{D^2 u(\alpha_0) \nabla u(\alpha_0)}{(1+|\nabla u(\alpha_0)|^2)^{1/2}} \beta''(t) = \left(0, \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}^T D^2 u(\alpha_0) \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}\right).
\end{aligned}$$

Putting these into (4.18), we get

$$T'(0) = \frac{1}{(1+|\nabla u(\alpha_0)|^2)^{3/2}} \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}^T D^2 u(\alpha_0) \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|} (-\nabla u(\alpha_0), 1)\,.$$

The norm of this vector is

$$|T'(0)| = \frac{1}{(1+|\nabla u(\alpha_0)|^2)} \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}^T D^2 u(\alpha_0) \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}.$$

So, we have the curvature:

$$\kappa_g = \frac{|T'(0)|}{|\beta'(0)|} = \frac{1}{(1+|\nabla u(\alpha_0)|^2)^{3/2}} \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}^T D^2 u(\alpha_0) \frac{\nabla u(\alpha_0)}{|\nabla u(\alpha_0)|}.$$

Thus, we see that the second term is a multiple of κ_g. It should be noted that the coefficients of both terms depend on ∇u. We can now say that equation (4.14) describes motion by curvature.

Rewriting (4.14) in terms of κ_g and $\kappa_{\ell s}$ gives

$$u_t = |\nabla u|^{p-1}\kappa_{\ell s} + (p-1)|\nabla u|^{p-2}\left(1+|\nabla u|^2\right)^{3/2}\kappa_g. \tag{4.20}$$

Notice that if u^* is a stationary solution to (4.20), then we can write

$$\frac{\kappa_{\ell s}}{\kappa_g} = \frac{(1-p)|\nabla u^*|^{p-2}(1+|\nabla u^*|^2)^{3/2}}{|\nabla u^*|^{p-1}} = \frac{(1+p)(1+|\nabla u^*|^2)^{3/2}}{|\nabla u^*|}. \tag{4.21}$$

This tells us that $\kappa_{\ell s}$ is much larger than κ_g. Indeed we have

$$\frac{\kappa_{\ell s}}{\kappa_g} \sim \begin{cases} (1+p)|\nabla u^*|^2 & \text{as } |\nabla u^*| \to \infty \\ \frac{1+p}{|\nabla u^*|} & \text{as } |\nabla u^*| \to 0 \end{cases} \tag{4.22}$$

4.2.2 Curvature Computation Examples

We can see that, for certain functions, the terms of (4.14) have opposite signs while for other functions, these terms will have the same sign. I now explore examples to understand when there is competition between these terms. In the second term, there is a coefficient that also competes with the rest of the term. That is, the coefficient, $\frac{1}{|\nabla u|^2}$ could be large when $\nabla u^T D^2 u \nabla u$ is small.

Example 4.1. As a first example, we consider the radial function $u_0(x, y) = e^{-x^2/\sigma^2 - y^2/\sigma^2} = e^{-r^2/\sigma^2}$. We find that

$$\Delta_p u_0 = \left(\frac{2r}{\sigma^2} e^{-r^2/\sigma^2}\right)^{p-2} \left((p-1)\left(\frac{2r^2}{\sigma^2} - 1\right) - 1\right) = -\frac{1}{\sigma^2}\left(\frac{2r}{\sigma^2} e^{-r^2/\sigma^2}\right)^{p-2} \left[2(1-p)r^2 + \sigma^2 p\right].$$

We see that this is negative everywhere away from the origin. This function is not a stationary solution and will begin to flatten in the evolution. It flattens everywhere because $u_t < 0$, that means that at the first iteration, $u(x)$ decreases.

Now, we consider an example where the level sets are straight lines, thus curvature of level sets $\kappa_{ls} = 0$. We are interested in how the second term involving κ_g contributes to the evolution. We see that steepening happens where the function is already steepest.

Example 4.2. Next, we consider the function

$$u(x, y) = \arctan(x).$$

With this function, we never have $|\nabla u| = 0$. Indeed, we see that

$$\nabla u = \begin{pmatrix} \frac{1}{1+x^2} \\ 0 \end{pmatrix}, \quad |\nabla u| = \frac{1}{1+x^2}.$$

Computing the hessian, we find that

$$D^2u = \begin{pmatrix} -\frac{2x}{(1+x^2)^2} & 0 \\ 0 & 0 \end{pmatrix}.$$

We then put these into the left hand side of (4.14) to get

$$\begin{aligned}\Delta_p u &= (1+x^2)^{1-p}(1+x^2)^{-1}\nabla \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} + (p-1)(1+x^2)^{2-p}\begin{pmatrix} 1 \\ 0 \end{pmatrix}^T \begin{pmatrix} \frac{-2x}{(1+x^2)^2} & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= (p-1)(1+x^2)^{2-p}\frac{-2x}{(1+x^2)^2} = 2(1-p)x(1+x^2)^{-p}.\end{aligned}$$

We see that this tells us that for $x > 0$, $u_t = \Delta_p u > 0$ and for $x < 0$, $u_t < 0$. This means that u steepens at the origin and flattens everywhere else. A small step produces the function (see Figure 4.6)

$$u(x,y) = u_0(x,y) + \delta\Delta_p u(x,y) = \arctan(x) + 2\delta(1-p)x(1+x^2)^{-p},$$

where $\delta > 0$ is a small number.

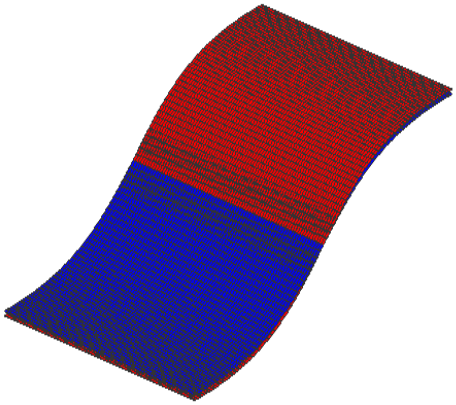
In the curvature computations above, we see that we can get competition between terms when both curvatures are positive or both are negative. As a simple example, we consider one where the curvature, $\kappa_{\ell s} \neq 0$ is constant on each level set. that is, we consider a function whose level sets are circles.

Example 4.3. We consider the paraboloid

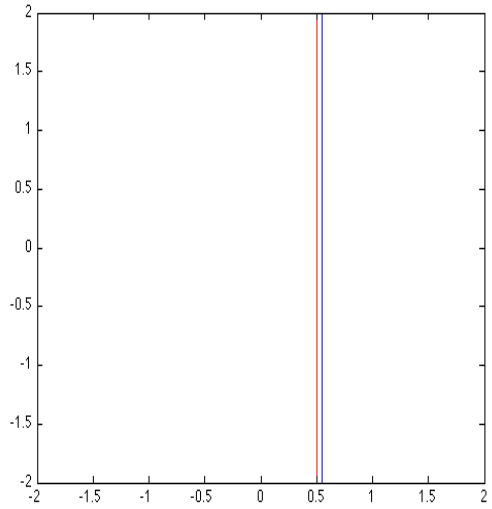
$$u_0(x,y) = x^2 + y^2$$

For now, let's assume $(x,y) \neq (0,0)$. We compute the first term, $\kappa_{\ell s}$:

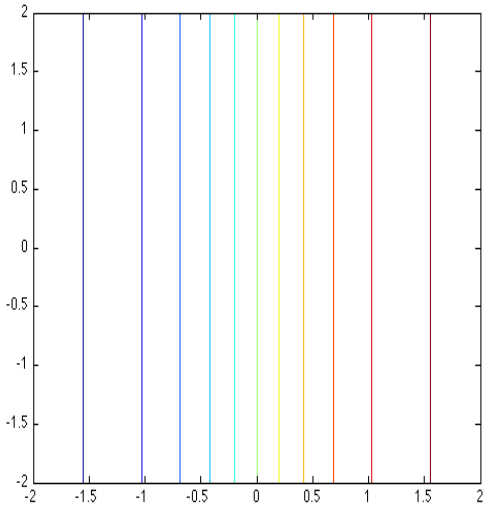
$$|\nabla u|^{p-1}\nabla \cdot \left(\frac{\nabla u}{|\nabla u|}\right) = 2^{p-1}(x^2+y^2)^{(p-2)/2}.$$



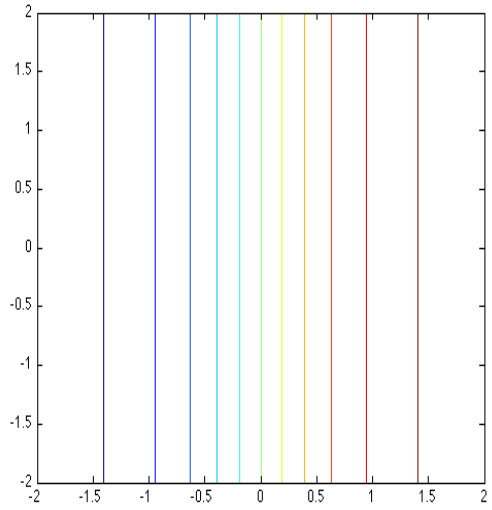
(a.)



(b.)



(c.)



(d.)

Figure 4.6: (a.) $u_0 = \arctan(x)$ (blue) and $u = u_0 + \delta\Delta_p u(x, y)$ (red), (b.) A level curve, at level 1, for each of u_0 (blue) and $u_0 + \delta\Delta_p u_0$ (red), (c.) level curves of u_0, and (d.) level curves of $u_0 + \delta\Delta_p u_0$

Now, the second term from above is

$$(p-1)|\nabla u|^{p-2}\frac{\nabla u}{|\nabla u|}^T D^2 u \frac{\nabla u}{|\nabla u|} = 2^{p-1}(p-1)(x^2+y^2)^{(p-2)/2}$$

Putting these together, we rewrite (4.14) as

$$u_t = 2^{p-1}p(x^2+y^2)^{(p-2)/2}$$

Everywhere away from the origin, this is positive, but it blows up at the origin. To try to understand how the function u changes if we allow a small step according to u_t, we compute

$$u(x,y) = u_0(x,y) + \delta\Delta_p u(x,y) = x^2+y^2+2^{p-1}\delta p(x^2+y^2)^{(p-2)/2},$$

where $\delta > 0$ is small. Figure 4.7 shows us the blow up at the origin, but it also shows us that at $r = 1$, the function does not change while everywhere else $u_0 + \Delta_p u_0 > u_0$.

Now we consider the case when $|\nabla u| = 0$. We try a β regularization. As in Chapter 2, we define

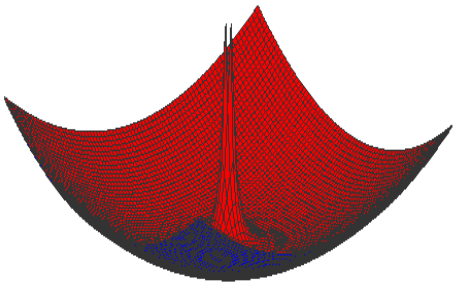
$$|\nabla u|_\beta \equiv \sqrt{\nabla u \cdot \nabla u + \beta^2}, \quad \text{for} \quad \beta \text{ small.}$$

Using $|\nabla u|_\beta$, we define

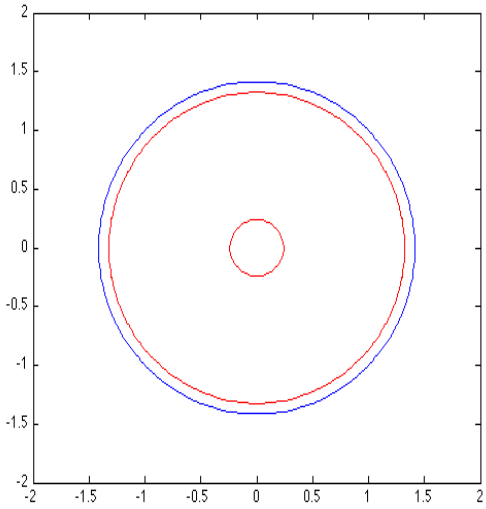
$$\Delta_{p,\beta} u \equiv \nabla \cdot \left(|\nabla u|_\beta^{p-2}\nabla u\right) = (p-2)|\nabla u|_\beta^{p-3}\nabla u^T D^2 u \frac{\nabla u}{|\nabla u|_\beta} + |\nabla u|_\beta^{p-2}\Delta u.$$

Replacing Δ_p with $\Delta_{p,\beta}$ in Example 4.3 we compute $\Delta_{p,\beta}u$. For $u_0 = x^2+y^2$, we have

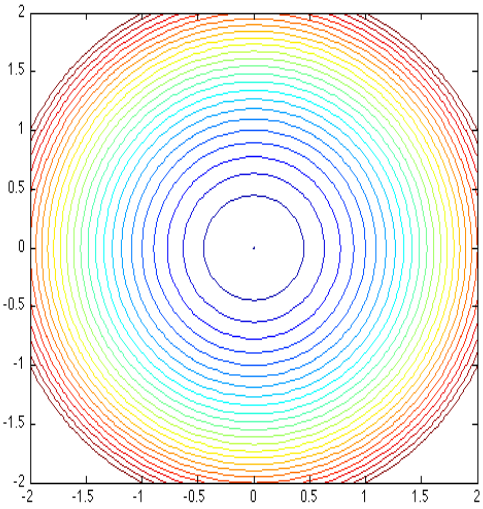
$$|\nabla u|_\beta = \left(4x^2+4y^2+\beta^2\right)^{1/2}.$$



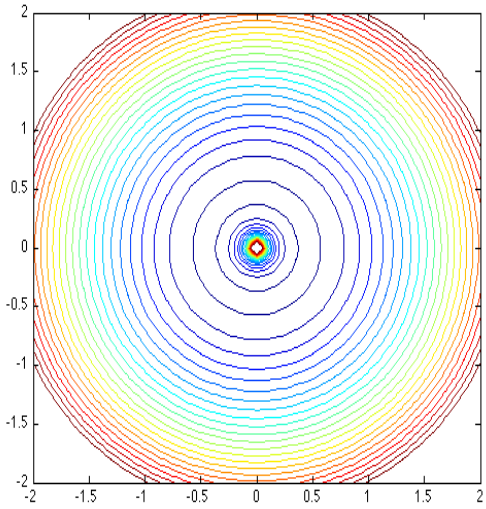
(a.)



(b.)



(c.)



(d.)

Figure 4.7: (a.) $u_0 = x^2 + y^2$ (blue) and $u = u_0 + \delta\Delta_p u(x, y)$ (red), (b.) A level curve, at level 2, for each of u_0 (blue) and $u_0 + \delta\Delta_p u_0$ (red), (c.) level curves of u_0, and (d.) level curves of $u_0 + \delta\Delta_p u_0$

And, we have

$$\begin{aligned}\Delta_{p,\beta}u &= (p-2)\left(4x^2+4y^2+\beta^2\right)^{(p-3)/2}\left(\tfrac{8x^2+8y^2}{\left(4x^2+4y^2+\beta^2\right)^{1/2}}\right)+4\left(4x^2+4y^2+\beta^2\right)^{(p-2)/2}\\ &= (p-2)\left(4x^2+4y^2+\beta^2\right)^{(p-4)/2}\left(8x^2+8y^2\right)+4\left(4x^2+4y^2+\beta^2\right)^{(p-2)/2}\\ &= \left(4x^2+4y^2+\beta^2\right)^{(p-4)/2}\left(8px^2+8py^2+4\beta^2\right).\end{aligned}$$

At the point $(0,0)$, we have $\Delta_{p,\beta}u(0,0,) = 4\beta^{p-2} \to \infty$ as $\beta \searrow 0$. So, we still have blowup at $(0,0)$.

We now consider a more interesting function where both curvature terms are nonzero.

Example 4.4. We consider the function $u(x,y) = \arctan(x) + y^2$. We begin computing as before. Notice that $|\nabla u| \neq 0$.

$$\nabla u = \begin{pmatrix} \frac{1}{1+x^2} \\ 2y \end{pmatrix}, \quad |\nabla u| = \left(\frac{1}{(1+x^2)^2}+4y^2\right)^{1/2}, \quad D^2u = \begin{pmatrix} \frac{-2x}{(1+x^2)^2} & 0 \\ 0 & 2 \end{pmatrix}.$$

We now put these into Equation (4.14). The first term is

$$|\nabla u|\nabla\cdot\left(\frac{\nabla u}{|\nabla u|}\right) = ((1+x^2)^{-2}+4y^2)^{(p-4)/2}\left(\tfrac{-2x((1+x^2)^{-2}+4y^2)}{(1+x^2)^2}+\tfrac{2x}{(1+x^2)^4}+2((1+x^2)^{-2}+4y^2)-8y^2\right)$$

The second term is

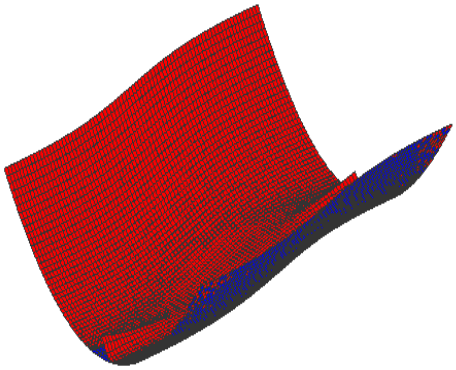
$$(p-1)\left((1+x^2)^{-2}+4y^2\right)^{(p-4)/2}\left(\frac{-2x}{(1+x^2)^4}+8y^2\right).$$

Putting these together gives

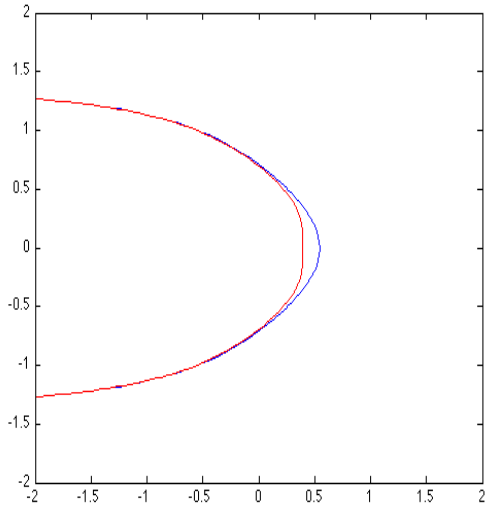
$$\Delta_p u = ((1+x^2)^{-2}+4y^2)^{(p-4)/2}\left(\frac{-2x((1+x^2)^{-2}+4y^2)}{(1+x^2)^2}+\frac{2x(2-p)}{(1+x^2)^4}+2((1+x^2)^{-2}+4y^2)-8(p-2)y^2\right)$$

We, again, want to see the effects on u. We look at $u+\delta\Delta_p u$. Notice, in Figure 4.8, that after this first step, the evolution introduces a point where $|\nabla u| = 0$. Notice also that , like the paraboloid, we see $u_0 + \delta\Delta_p u_0 > u_0$.

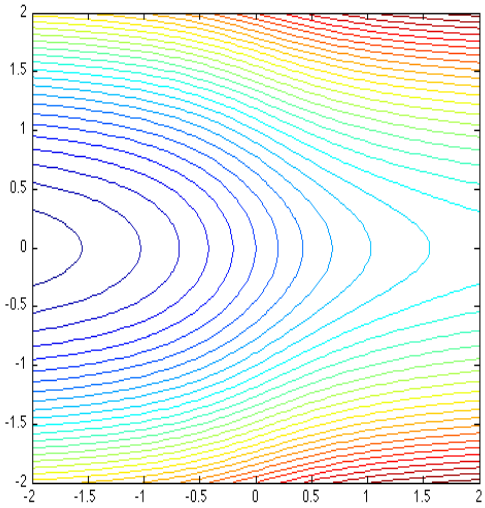
Here we see that the first term involving $\kappa_{\ell s}$ seems to win when the terms compete.



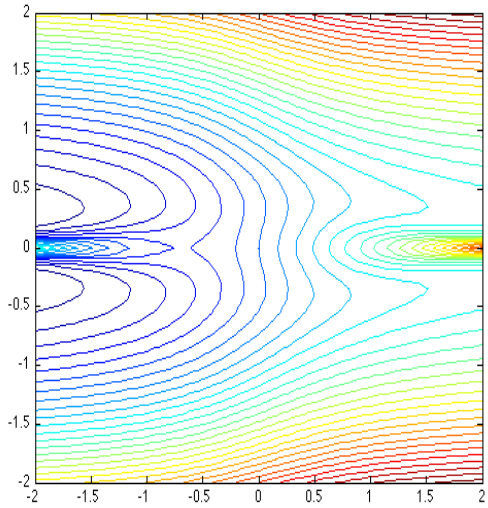
(a.)



(b.)



(c.)



(d.)

Figure 4.8: (a.) $u_0 = \arctan(x) + y^2$ (blue) and $u = u_0 + \delta\Delta_p u(x, y)$ (red), (b.) A level curve, at level 1, for each of u_0 (blue) and $u_0 + \delta\Delta_p u_0$ (red), (c.) level curves of u_0, and (d.) level curves of $u_0 + \delta\Delta_p u_0$

It seems that where u is steep, Equation 4.14 will steepen u. This leads us to conjecture that if a function is monotone, it will remain monotone in the evolution. We see this is the case for functions defined on 1-dimensional domains.

4.3 Classical Solutions

In this section, we find families of classical stationary solutions to (4.5). Here it is worth noting that if u is a minimizer for (2.1) then u is a step function, thus $u \notin C^2(\Omega)$ and is therefore not a classical solution of (4.4).

For a function to be a stationary solution we need that $u \in C^2(\Omega)$ and $\Delta_p u = 0$. That is, we need

$$|\nabla u| \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) = \frac{1-p}{|\nabla u|^2} \nabla u^T D^2 u \nabla u$$

4.3.1 1-dimensional classical solutions

In the 1-D case, this is the same as

$$|u'| \left(\frac{u'}{|u'|} \right)' = (1-p) u'' \tag{4.23}$$

If $|u'| \neq 0$, then u must be monotone and so $\frac{u'}{|u'|} = c$. So, u'' must be zero, telling us that as long as u is monotone, it cannot have any curvature. So, any linear function u satisfies this condition. Thus, no polynomial function with degree higher than 1 will satisfy (4.23). Indeed, if u is a solution to (4.23) with boundary conditions $u(a) = A, u(b) = B$, we get

$$\begin{aligned} |u'|^{p-2} u' = c \quad &\Rightarrow \quad u \text{ is strictly monotone} \quad \Rightarrow \quad (u')^{p-1} = c \quad \Rightarrow \quad u' = c \\ &\Rightarrow \quad u(x) = \frac{B-A}{b-a}(x-a) + A. \end{aligned} \tag{4.24}$$

Above, we relax notation and allow the c to change throughout. Further, Lemma 4.1 tells us that these functions are not stable stationary solutions to (4.5). Notice that, even when $\Omega \subset \mathbb{R}$, we do

not get that a linear function is a minimizer of (2.1).

Lemma 4.1. *u given in Equation (4.24) is an unstable stationary solution to the problem*

$$\begin{cases} u_t = \left(|u'|^{p-2}u'\right)' \\ u(a) = A \quad u(b) = B. \end{cases} \tag{4.25}$$

In particular, $\tilde{u}$ is a maximizer of $\int_a^b |u'(x)|^p \, dx$ over all strictly monotonic C^2 functions.

Proof. Without loss of generality, let us assume that $\frac{B-A}{b-a} > 0$. Let $u(x) = \tilde{u}(x) + v(x)$ for $v \in C^2((a,b))$ chosen so that $u' > 0$, $v(x) \neq 0$ for some $x \in (a,b)$, and $v(a) = v(b) = 0$. Our goal is to show that, in the evolution, u moves away from $\tilde{u}$. That is, if $u(x) > \tilde{u}(x)$, then $u_t(x) > 0$ and if $u(x) < \tilde{u}(x)$, then $u_t(x) < 0$. First, we compute u_t

$$u_t = \left(|u'|^{p-2}u'\right)' = \left((u')^{p-1}\right)' = \left(\left(\frac{B-A}{b-a} + v'\right)^{p-1}\right)' = (p-1)\left(\frac{B-A}{b-a} + v'\right)^{p-2} v''.$$

We now proceed by contradiction. That is, we assume $\tilde{u}$ is a stable stationary solution and we seek a contradiction. Since $\tilde{u}$ is stable, we know that if $u < \tilde{u}$ then $u_t > 0$ and if $u > \tilde{u}$ then $u_t < 0$. That is,

$$\begin{aligned} v(x) < 0 \text{ (i.e. } u(x) < \tilde{u}(x)) \quad &\Rightarrow \quad v''(x) < 0 \text{ (i.e. } u_t > 0) \\ &\text{and} \\ v(x) > 0 \text{ (i.e. } u(x) > \tilde{u}(x)) \quad &\Rightarrow \quad v''(x) > 0 \text{ (i.e. } u_t < 0). \end{aligned} \tag{4.26}$$

Now, (4.26) is equivalent to

$$\begin{aligned} u(x) < \tilde{u}(x) \quad &\Rightarrow \quad u''(x) < 0 \\ &\text{and} \\ u(x) > \tilde{u}(x) \quad &\Rightarrow \quad u''(x) > 0. \end{aligned} \tag{4.27}$$

Let,

$$x_1 = \max\{x : u(y) = \tilde{u}(y) \quad \forall y \in [a, x]\} \tag{4.28}$$

and let

$$x_2 \in (x_1, b] \quad \text{be such that} \quad u(x_2) = \tilde{u}(x_2) \quad \text{and} \quad u(x) < \tilde{u}(x) \quad \text{for all } x \in (x_1, x_2). \tag{4.29}$$

If x_2 exists, then we use that $\tilde{u}$ is a line and the Mean Value Theorem from Calculus to say that

$$\exists c \in (x_1, x_2) \quad \text{such that} \quad u'(c) = \frac{\tilde{u}(x_2) - \tilde{u}(x_1)}{x_2 - x_1} = \frac{B - A}{b - a} = \tilde{u}'(c).$$

But, since $u''(x) < 0$ for all $x \in (x_1, x_2)$, $u'(x) < \tilde{u}'(x)$ for all $x \in (x_1, x_2)$. But, $c \in (x_1, x_2)$ gives us that

$$\tilde{u}'(c) = u'(c) < u'(x_1) \leq \tilde{u}'(x_1).$$

But this cannot be. So, x_2 cannot exist.

Let

$$x_3 \in (x_1, b] \quad \text{be such that} \quad u(x_3) = \tilde{u}(x_3) \quad \text{and} \quad u(x) > \tilde{u}(x) \quad \text{for all } x \in (x_1, x_3). \tag{4.30}$$

If x_3 exists, we again, can use the fact that $\tilde{u}$ is a line and the Mean Value Theorem, to say that

$$\exists c \in (a, b) \quad \text{such that} \quad u'(c) = \frac{\tilde{u}(x_3) - \tilde{u}(x_1)}{x_3 - x_1} = \frac{B - A}{b - a} = \tilde{u}'(c).$$

But, since $u''(x) > 0$ for all $x \in (x_1, x_3)$, $u'(x) > \tilde{u}'(x)$ for all $x \in (x_1, x_3)$, but $c \in (x_1, x_2)$. So, we have that

$$\tilde{u}'(c) = u'(c) > u'(x_1) \geq \tilde{u}'(x_1).$$

This cannot be. So, x_3 cannot exist. But, this means that $v \equiv 0$ on $[a, b]$ which is a contradiction. Thus, $\tilde{u}$ cannot be stable. □

In the above, we don't allow $|u'| = 0$. But, we can consider a similar equation using a β regularization,

$$\begin{cases} u_t = (|u'|_\beta^{p-2}u')' & (x,t) \in [0,1] \times [0,\infty) \\ \\ u(a) = A, u(b) = B \end{cases} \tag{4.31}$$

where $|u'|_\beta = (|u'|^2 + \beta^2)^{1/2}$ for some $\beta > 0$. If u is a stationary solution to (4.31), then

$$0 = (|u'|_\beta^{p-2}u')' \quad \Rightarrow \quad |u'|_\beta^{p-2}u' = c \quad \Rightarrow \quad u' = c|u'|_\beta^{2-p} \quad \Rightarrow \quad u' = c \quad \Rightarrow \quad u \text{ is linear.}$$

Again, in the above, we let c change from step to step. The only difference we find in this is that u' can be zero in the β regularized equation. This means we can allow the boundary conditions with $A = B$ whereas in the previous case ($\beta = 0$), we cannot.

4.3.2 Higher dimensional classical solutions

To find stationary solutions with higher dimensional domain, I begin by considering what happens when one of the two curvatures discussed in Section 4.2 is zero. We consider strictly Normal monotone functions, $u \in C^2(\Omega)$, so that $|\nabla u(x)| \neq 0$ for x in some open bounded set Ω.

Notice that if $\nabla \cdot \left(\frac{\nabla u}{|\nabla u|}\right) = 0$, the level sets of u are minimal surfaces (see [14, 27]). This means that for $u : \mathbb{R}^2 \to \mathbb{R}$, the level sets of u are lines. We saw above that $u : \mathbb{R} \to \mathbb{R}$ is a monotone stationary solution to the 1-dimensional p-Laplacian if and only if the graph of u is a line. This result doesn't extend to higher dimensions because, even though it is easy to verify that an affine function is a stationary solution to (1.3), there are other functions whose level sets are lines.

We saw above in Example 4.3 that not a paraboloid is not a stationary solution, so we check to see if there exists a radial stationary solution. We begin by considering a function, $u = u(r)$, with domain $\Omega \subset \mathbb{R}^2$ so that u is strictly Normal monotone on Ω and so that u' is nonzero in Ω and $u'(r) > 0$. Converting (1.3) to polar coordinates (see Appendix A), we get that u must satisfy the

ordinary differential equation

$$\frac{1}{r} + \frac{p-1}{u'(r)}u''(r) = 0. \tag{4.32}$$

Rearranging this equation, we get that u must satisfy

$$\frac{u'(r)}{r} + (p-1)u''(r) = 0. \tag{4.33}$$

Thus, $u(r) = ar^m + b$, for some constants $a, b \in \mathbb{R}$ and for $m = \frac{2-p}{1-p}$ (See Figure 4.9). That is,

$$u(x,y) = a(x^2+y^2)^{(2-p)/(1-p)} + b. \tag{4.34}$$

Notice that if $u'(r) < 0$, we get a similar result. Notice that the gradient of u points away from the

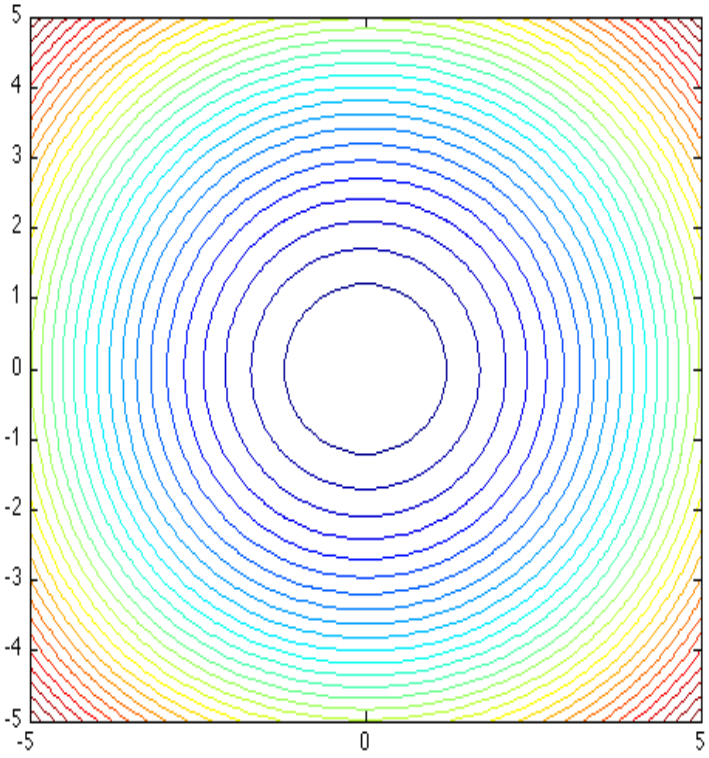


Figure 4.9: Radial solution, $u(r) = r^m$ for the p-Laplacian where $p = .4$.

origin. We also see that as long as we choose Ω so that the convex hull of Ω does not include the

origin, these functions are strictly Normal monotone.

Inspired by finding a radial solution, we consider looking for a polar solution. That is, we find a family of functions of the form $u = u(\theta)$ $0 < \theta < \pi$. In this case, we find, using Equation (A.23)

$$\nabla_{r,\theta} u = \begin{pmatrix} u_r \\ \frac{1}{r} u_\theta \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{1}{r} u'(\theta) \end{pmatrix} \tag{4.35}$$

$$\Rightarrow \quad \frac{\nabla_{r,\theta} u}{|\nabla_{r,\theta} u|} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \Rightarrow \quad \nabla_{r,\theta} \cdot \left(\frac{\nabla_{r,\theta} u}{|\nabla_{r,\theta} u|} \right) = 0. \tag{4.36}$$

Thus, u must satisfy

$$\nabla_{r,\theta} u^T D^2_{r,\theta} u \nabla_{r,\theta} u = 0 \quad \Rightarrow \quad u''(\theta) = 0. \tag{4.37}$$

That is, $u(\theta) = a\theta + b$ for some constants $a, b \in \mathbb{R}$.

Now, we look to extend this to higher dimensions. We consider a function $u : \mathbb{R}^n \to \mathbb{R}$ in n-dimensional spherical coordinates $(r, \theta_1, \theta_2, ...\theta_{n-1})$ so that $u = u(r)$. Then we compute $\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right)$ for $u'(r) > 0$.

$$\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) = \frac{1}{h_1 h_2 \cdots h_n} \frac{\partial}{\partial r} (h_2 h_3 \cdots h_n) = \frac{n-1}{r}, \tag{4.38}$$

where h_i are the scale factors computed in Equation (A.25). Thus,

$$\frac{n-1}{r} + \frac{p-1}{u'(r)} u''(r) = 0 \quad \Rightarrow \quad u(r) = c r^{m_1}, \tag{4.39}$$

where $m_1 = \frac{n-p}{1-p}$.

First, we begin by finding the azimuthal solution, $u = u(\theta_{n-1})$. We compute $\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right)$ for $u'(\theta_{n-1}) > 0$ to get

$$\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) = \frac{1}{h_1 h_2 \cdots h_n} \frac{\partial}{\partial \theta_{n-1}} (h_1 h_2 h_3 \cdots h_{n-1}) = 0. \tag{4.40}$$

Thus,

$$u''(\theta_{n-1}) = 0 \quad \Rightarrow \quad u(\theta_{n-1}) = a\theta_{n-1} + b. \tag{4.41}$$

We have already seen an example of such a function in Chapter 3. We show this example again in Figure 4.10. As we said in Example 3.16, the function in Figure 4.10 is Normal monotone.

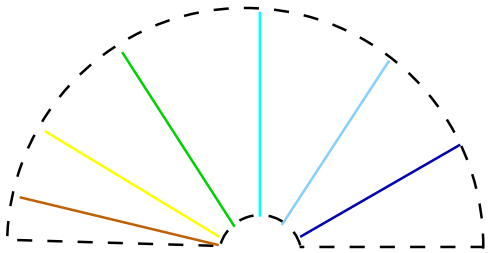


Figure 4.10: An azimuthal stationary solution to the p-Laplacian evolution equation.

If $u = u(\theta_j)$, $1 \geq j < n-1$, we have

$$\begin{aligned}
\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) &= \frac{1}{h_1 \cdots h_n} \frac{\partial}{\partial \theta_j} \left(h_1 \cdots h_{j-1} h_{j+1} \cdots h_n \right) \\
&= \frac{1}{r^{n-1} \sin^{n-2} \theta_1 \cdots \sin \theta_{n-2}} \frac{\partial}{\partial \theta_j} \left(r^{n-2} \sin^{n-3} \theta_1 \cdots \sin^{n-j-1} \theta_{j-1} \sin^{n-j-1} \theta_j \cdots \sin \theta_{n-2} \right) \\
&= \frac{1}{r \sin \theta_1 \cdots \sin \theta_{j-1} \sin^{n-j-1} \theta_j} ((n-j-1) \sin^{n-j-2} \theta_j \cos \theta_j) = \frac{(n-j-1) \cot \theta_j}{r \sin \theta_1 \cdots \sin \theta_{j-1}} \\
&= \frac{n-j-1}{h_j} \cot \theta_j. \qquad (4.42)
\end{aligned}$$

The second term of (4.14) for this u is

$$\frac{p-1}{|\nabla_{r\theta_1 \ldots \theta_{n-1}} u|^3} \left(\nabla_{r\theta_1 \ldots \theta_{n-1}} u \right)^T H_{r\theta_1 \ldots \theta_{n-1}} u \nabla_{r\theta_1 \ldots \theta_{n-1}} u = \frac{p-1}{h_j u'(\theta_j)} u''(\theta_j). \qquad (4.43)$$

Thus, u satisfies

$$\frac{n-j-1}{p-1} \cot \theta_j u'(\theta_j) + u''(\theta_j) = 0. \qquad (4.44)$$

We solve this equation to get

$$u(\theta_j) = c \int_{c_0}^{\theta_j} \sin^{m_j} \theta \, d\theta, \qquad (4.45)$$

where $m_j = \frac{n-j-1}{1-p}$ and c, c_0 are arbitrary constants. Notice that these solutions are also strictly

Normal monotone as long as the domain Ω does not include the polar axis associated to θ_j.

We now consider an even more general setting. We consider the orthogonal coordinate system $(q^1, q^2, \ldots, q^n)$. We assume $u = u(q^i)$ for some $1 \leq i \leq n$. In this case, we write the terms of (4.14) after converting to this orthogonal coordinate system. The first term is

$$\nabla_{q^1 \ldots q^n} \cdot \left(\frac{\nabla_{q^1 \ldots q^n} u}{|\nabla_{q^1 \ldots q^n}|} \right) = \frac{1}{\Pi_{j=1}^n h_j} \frac{\partial}{\partial q^i} \left(\Pi_{j \neq i} h_j \right). \tag{4.46}$$

The second term is

$$\frac{p-1}{|\nabla_{q^1 \ldots q^n} u|^3} \left(\nabla_{q^1 \ldots q^n} u \right)^T H_{q^1 \ldots q^n}(u) \nabla_{q^1 \ldots q^n} u = \frac{p-1}{h_i u'(q^i)} u''(q^i). \tag{4.47}$$

Putting these together, we see that for there to be a solution of the form $u = u(q^i)$, u must satisfy

$$\frac{1}{\Pi_{j=1}^n h_j} \frac{\partial}{\partial q^i} \left(\Pi_{j \neq i} h_j \right) + \frac{p-1}{h_i u'(q^i)} u''(q^i) = 0. \tag{4.48}$$

But this is not possible if we cannot remove the dependence on $q^k, k \neq i$ in $\Pi_{j=1}^n h_j$. If this dependence does not exist or if we can eliminate it then we can write the solution as

$$u(q^i) = c \int_{c_0}^{q^i} \exp\left(\frac{1}{1-p} A(q) \right) dq, \tag{4.49}$$

where $A(q^i)$ is any antiderivative of $\frac{1}{H} \frac{\partial H}{\partial q^i} dq^i$ for $H = \Pi_{j \neq i} h_j$ and c, c_0 are arbitrary constants.

Remark 4.2. It's worth noting that other than the spherical coordinate system, the other common orthogonal coordinate systems (i.e., parabolic, paraboloidal, elliptic,...) do not give us any new solutions of the form $u = u(u^i)$.

Notice that (4.4) is nonlinear so it is unlikely that if v and w are two solutions to (4.4) that $u = v + w$ would be also. We verify that this is not the case for two solutions from above. Let $w(r) = cr^{\frac{2-p}{1-p}}$ and $v(\theta) = a\theta + b$. We will show that $u = w + v$ is not a solution ot (4.4). First we

compute the gradient of u

$$\nabla u = \begin{pmatrix} c\frac{2-p}{1-p} r^{1/(1-p)} \\ \\ \frac{a}{r} \end{pmatrix} \quad \Rightarrow \quad |\nabla u| = \left(c^2 \left(\frac{2-p}{1-p} \right)^2 r^{2/(1-p)} + \frac{a^2}{r^2} \right)^{1/2}. \tag{4.50}$$

So,

$$\begin{aligned} \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) &= \frac{1}{r} \left(\frac{\partial}{\partial r} \left(c \frac{2-p}{1-p} r^{\frac{2-p}{1-p}} \left(c^2 \left(\frac{2-p}{1-p} \right)^2 r^{2/(1-p)} + \frac{a^2}{r^2} \right)^{-1/2} \right) \right) \\ &+ \frac{\partial}{\partial \theta} \left(\frac{a}{r} \left(c^2 \left(\frac{2-p}{1-p} \right)^2 r^{2/(1-p)} + \frac{a^2}{r^2} \right)^{-1/2} \right) \\ &= c \left(\frac{2-p}{1-p} \right)^2 r^{1/(1-p)} \frac{c^2 \left(\frac{2-p}{1-p} \right)^2 r^{2/(1-p)} + \frac{a^2}{1-p} r^{\frac{2p-1}{1-p}}}{\left(c^2 \left(\frac{2-p}{1-p} \right)^2 r^{2/(1-p)} + \frac{a^2}{r^2} \right)^{3/2}}. \end{aligned} \tag{4.51}$$

The hessian is

$$D^2 u = \begin{pmatrix} c \left(\frac{2-p}{(1-p)^2} \right) r^{2/(1-p)} & 0 \\ 0 & 0 \end{pmatrix}. \tag{4.52}$$

So,

$$\nabla u^T D^2 u \nabla u = c^3 \left(\frac{2-p}{1-p} \right)^3 \frac{1}{1-p} r^{\frac{2+p}{1-p}}. \tag{4.53}$$

Putting these together gives

$$\Delta_p u = \frac{a^2 c \left(\frac{2-p}{1-p} \right)^2 \frac{1}{1-p} r^{\frac{3p-1}{1-p}}}{\left(c^2 \left(\frac{2-p}{1-p} \right)^2 r^{2/(1-p)} + \frac{a^2}{r^2} \right)^{3/2}} \tag{4.54}$$

which is nonzero (except, of course, when $a = 0$). Thus, u is not a solution. This tells us that the set of stationary solutions is nonconvex.

4.4 Stationary Solution Summary

In this chapter, we looked for stationary solutions to the p-Laplacian evolution equations which is the Euler-Lagrange equation for $\int_\Omega |\nabla u|^p$. Even though the functional is nonconvex we still

search for stationary solutions of its Euler-Lagrange equation to compare with the minimizers of p-variation. We know that solutions to the p-Laplacian cannot be step functions, but seek out other stationary solutions.

We addressed the issues that occur because the p-Laplacian, $\Delta_p u = 0$ for $0 < p < 1$, is singular. We also recognize that since classical solutions require a function to be C^2, none of our classical solutions are step functions, that is, none are minimizers to the p-variation problem. To find classical solutions, we break this equation into the sum of two curvature expressions, $\kappa_{\ell s}$ and κ_g:

$$0 = |\nabla u|^{p-1}\kappa_{\ell s} + (p-1)|\nabla u|^{p-2}\left(1 + |\nabla u|^2\right)^{3/2}\kappa_g. \tag{4.55}$$

Using the curvature interpretation of this equation, we seek out solutions that are classical solutions to the p-Laplacian for $0 < p < 1$. We found families of classical solutions. In particular, we have the following families of stationary solutions

- Affine family: For $x \in \mathbb{R}^n$, $u(x) = a \cdot x + b$, where $a \in \mathbb{R}^n, b \in \mathbb{R}$ are constants.

And in n dimensional spherical coordinates $(r, \theta_1, \ldots, \theta_{n-1})$

- Radial family: $u(r) = ar^m + b$, where $m = \frac{n-p}{1-p}$,
- Polar family: $u(\theta_j) = a\int_b^{\theta_j} \csc^{m_j}\theta \, d\theta$, where $m_j = \frac{n-j-1}{p-1}$, where $1 \le j \le n-2$ and
- Azimuthal family: $u(\theta_{n-1}) = a\theta_{n-1} + b$.

We also found that similar techniques, to those we employed here, to find classical stationary solutions cannot work in the most common orthogonal coordinate systems.

We would still like to find or show no maximum principle can be obtained for the p-Laplacian and the corresponding evolution equation. With this, we would be able to give a more complete discussion of viscosity solutions as they relate to these equations.

In 1-dimensional examples, we have seen that monotone functions stay monotone in the evolution given by the p-Laplacian evolution equation. We conjecture that a similar result is true

about Normal monotone functions in higher dimensions. We would like to determine the notions of monotonicity in higher dimensions for which this is true. That is, we would like to determine for which notion(s) of monotonicity will a function remain monotone in the evolution.

CHAPTER FIVE

ALGORITHMS FOR 1-DIMENSIONAL L^1pTV MINIMIZATION

In this chapter, we introduce a discrete formulation for L^1pTV, that is, the $q = 1$ case of (1.4):

$$\min_{u\in\mathcal{B}_n(\Omega)} \int_\Omega |\nabla u|^p \, dx + \lambda \int_\Omega |f - u| \, dx, \quad \text{for } 0 < p \leq 1. \tag{5.1}$$

We introduce algorithms that find minimizers for (5.1) for the cases $p = 1$, $p < 1$, and $\lambda = 0$. We give a more efficient version of the L^1TV algorithm (that is, for the $p = 1$) that gives solutions for all λ thereby giving us a fast algorithm to compute scale signatures for 1-dimensional signals. This version of our algorithm not only finds all solutions, but it finds them all with only the computational cost of solving the problem for one value of λ because it picks up the solutions for each λ at each of the iterations. We generalize the more efficient algorithm to find local minimizers for L^1pTV, $p \leq 1$.

5.1 Discrete L^1pTV

In this section, I give a discrete formulation for (5.1). I then discuss the motivations behind the cases $p = 1$, $p < 1$, and $\lambda = 0$.

5.1.1 Discretization

To write a discrete formulation of (5.1) we begin with the idea of computing (5.1) for functions $u : \Omega \subset \mathbb{R} \to \mathbb{R}$ that are piecewise linear. We can then define the minimum (with $\Omega = (a, b)$) to be

$$\min_{u\in\mathcal{F}} \int_a^b |u'|^p + \lambda|f - u|\, dx \equiv \min_{u\in\mathcal{F}} \left(\sum_{i=1}^{m-1} \left| \frac{u_{i+1} - u_i}{x_{i+1} - x_i} \right|^p (x_{i+1} - x_i) + \frac{\lambda}{2} \sum_{i=1}^{m} |f_i - u_i|(x_{i+1} - x_{i-1}) \right)$$
$$= \min_{u\in\mathcal{F}} \left(\sum_{i=1}^{m-1} |u_{i+1} - u_i|^p \, (x_{i+1} - x_i)^{1-p} + \frac{\lambda}{2} \sum_{i=1}^{m} |f_i - u_i|(x_{i+1} - x_i) \right), \tag{5.2}$$

where $\lambda > 0$, $a = x_1 < x_2 < \ldots < x_m = b$ is a fixed partition of Ω and

$$\mathcal{F} = \left\{ u : \Omega \to \mathbb{R} : u \in C(\Omega), u = \sum_{i=0}^{m} L_i \chi_{[x_i, x_{i+1}]}, \text{ where } L_i \text{ are linear}, u(x_i) = \beta_i, u(x_{i+1}) = \beta_{i+1} \right\}.$$

Here, we use that on each piece, u is linear so we replace in the interval (x_i, x_{i+1}), u' with the slope of u on this interval:

$$u' = \frac{u_{i+1} - u_i}{x_{i+1} - x_i}. \tag{5.3}$$

The x dependence in the fidelity term depends on the length of the partition intervals. Instead of using $x_{i+1} - x_i$, we use the distance from the midpoints of (x_{i-1}, x_i) and (x_i, x_{i+1}). We choose this because the partition is not necessarily regular.

5.2 L^1TV in 1-Dimension

In this section, we discuss the discrete version of L^1TV:

$$\min \int_\Omega |\nabla u| + \lambda|f - u|\, dx. \tag{5.4}$$

We introduce algorithms that find minimizers of the discrete problem. In [30], the authors show that L^1TV is useful in picking out scale information from data. So, we use these algorithms to find scale information for 1-dimensional signals. We also prove that our algorithm will indeed find the

minimizer of the discretized problem. Inspired by a part of this proof, we create a new version of this algorithm which is more efficient and solves the discretized problem for all $\lambda > 0$ with the computational cost of solving only the $\lambda = 0$ problem.

5.2.1 Discrete Formulation of L^1TV

Using the formulation in Equation (5.2) on a fixed regular partition, we define the discrete L^1TV formulation as follows. Since $p = 1$, we no longer have an x dependence in the first sum. It makes sense to remove the x dependence in the second sum because the partition is regular and fixed. Thus, we get

$$G(u_1, u_2, \ldots, u_m) = \sum_{i=1}^{m-1} |u_{i+1} - u_i| + \lambda \sum_{i=1}^{m} |f_i - u_i|, \tag{5.5}$$

where $u_1 = \alpha, u_m = \beta$ are the boundary values and $f = (f_1, \ldots, f_m)$ is the given data.

Notice that F is nonsmooth at any point in in the following two sets

$$\mathcal{S}_u := \{u = (u_1, \ldots, u_m) : u_i = u_{i+1}, \text{ for some } i = 1...m\} \tag{5.6}$$

and

$$\mathcal{S}_f := \{u = (u_1, u_2, \ldots, u_m) : u_i = f_i, \text{ for some } i = 1, ..., m\}. \tag{5.7}$$

5.2.2 Properties of the Discrete of L^1TV Function

Here, we show some properties of G to show that it indeed has minimizers. Notice that, to show G has a minimizer, it suffices to show that G is convex, bounded below, and coercive. We begin with this result in the following lemma.

Lemma 5.1. *G is bounded below, convex, and coercive in that as $|u| \to \infty$ $G \to \infty$.*

Proof. G is bounded below by zero. To show G is convex, we let $0 \leq \sigma \leq 1$ and we show that

$G(\sigma u + (1-\sigma)\tilde{u}) \leq \sigma G(u) + (1-\sigma)G(\tilde{u})$. Indeed

$$\begin{aligned}
G(\sigma u + (1-\sigma)\tilde{u}) = & \quad \textstyle\sum_{(i,j)\in E} |\sigma u_i + (1-\sigma)\tilde{u}_i - (\sigma u_j + (1-\sigma)\tilde{u}_j)| + \lambda \sum_i |f_i - (\sigma u_i + (1-\sigma)\tilde{u}_i)| \\
= & \quad \textstyle\sum_{(i,j)\in E} |\sigma(u_i - u_j) + (1-\sigma)(\tilde{u}_i - \tilde{u}_j)| + \lambda \sum_i |\sigma(f_i - u_i) + (1-\sigma)(f_i\tilde{u}_i)| \\
\leq & \quad \sigma\left(\textstyle\sum_{(i,j)\in E} |u_i - u_j| + \lambda \sum_i |f_i - u_i|\right) + (1-\sigma)\left(\sum_{(i,j)\in E} |\tilde{u}_i - \tilde{u}_j| + \lambda \sum_i |(f_i - \tilde{u}_i)|\right) \\
= & \quad \sigma G(u) + (1-\sigma)G(\tilde{u})
\end{aligned}$$

Finally, we notice that $G(u) \geq \lambda \sum_i (|u_i - f_i|) \geq \lambda \sum_i (|u_i| - |f_i|) \to \infty \quad \text{as} \quad |u| \to \infty$. Thus, G is coercive. □

Now that we know G has a minimizer, we give the set of minimizers a name.

Definition 5.1. We define the set of global minimizers, $\mathcal{M}_\lambda$ for G, noting that this set depends on the value $\lambda > 0$,

$$\mathcal{M}_\lambda := \arg\min_u G.$$

We now follow up with some properties of $\mathcal{M}_\lambda$. The following lemma follows from Lemma 5.1.

Lemma 5.2. *$\mathcal{M}_\lambda$ is bounded.*

Proof. This result follows from the coercivity of G. For otherwise, if $\mathcal{M}_\lambda$ was unbounded, then for any α large, we could find a direction d and a point u so that $u + \alpha d \in \mathcal{M}_\lambda$ which contradicts the coercivity condition. □

Lemma 5.3. *$\mathcal{M}_\lambda$ is convex.*

Proof. Let $u^*, v^* \in \mathcal{M}_\lambda$ then $G(u^*) \leq G(u)$ for all $u \in \mathbb{R}^m$ and $G(v^*) \leq G(u)$ for all $u \in \mathbb{R}^m$. Using convexity of G, we have, for $0 \leq \eta \leq 1$,

$$G(\eta u^* + (1-\eta)v^*) \leq \eta G(u^*) + (1-\eta)v^* \leq \eta G(u) + (1-\eta)G(u) = G(u), \text{ for all } u \in \mathbb{R}^m.$$

Thus, $\eta u^* + (1-\eta)v^* \in \mathcal{M}_\lambda$. □

We also know that, using Lemma 5.1, that no algorithm will get stuck at a local minimizer because G has no local minimizers..

Lemma 5.4. *If u is a local minimizer of G, then $u \in \mathcal{M}_\lambda$. That is, if u is a local minimizer, it is also a global minimizer.*

We now define the finite set in which we will find our minimizer.

Definition 5.2. Let $G : \mathbb{R}^m \to \mathbb{R}$ be defined as in (5.5). Let Y be the set of points of intersections of at least m hyperplanes of the form $\{u_i = f_i\}$ and/or $\{u_i = u_{i+1}\}$.

We now proceed to show Y is finite and a minimizer of G is indeed in Y.

Lemma 5.5. *Let $E = \{(i,j) | u_i, u_j \text{ are neighbors}\}$ $|Y| \leq \frac{(|E|+m)!}{(|E|)!(m)!} < \infty$.*

Proof. First note that the number of hyperplanes of the form $\{u_i = u_j : (i,j) \in E\}$ is $|E|$. The number of hyperplanes of the form $\{u_i = f_i\}$ is m. Using the definition of Y, we can count all the possible intersections of m of these hyperplanes is $\begin{pmatrix} |E|+m \\ m \end{pmatrix}$. That is $|Y| \leq \begin{pmatrix} |E|+m \\ m \end{pmatrix} = \frac{(|E|+m)!}{(|E|)!(m)!}$. (Here, the first is an inequality because we may have hyperplanes that are everywhere the same.) □

Remark 5.1. In the 1-dimensional case $|E| = m-1$ so this is $|Y| \leq \frac{(2m-1)!}{(m-1)!m!}$.

Lemma 5.6. *There is a minimizer of G, call it x^*, in Y.*

Proof. Let $\hat{u}$ be a minimizer of G and $\hat{u} \notin Y$. Suppose first that $\nabla G(\hat{u})$ exists. Then $\nabla G(\hat{u}) = 0$. But then because G is affine at points where it is differentiable, G is constant in the whole region containing $\hat{u}$ up to and including the bounding hyperplanes where G is nonsmooth. By the coercivity condition, this region must be bounded. So, there is a point, x^*, on the boundary of this region that is in Y such that $G(x^*) = G(\hat{u})$ where G is given in (5.5).

Second, suppose that $\hat{u} \in \mathcal{H}$ where $\mathcal{H}$ is the intersection of $\ell < m$ hyperplanes where G is nonsmooth, then we consider the function $\tilde{G}$ which is G restricted to $\mathcal{H}$. Then $\nabla \tilde{G}(\hat{u})$ exists and is

zero. We can then use the same argument above to get that the set of minimizers includes a point in Y. □

5.2.3 Hyperplane Traversal Algorithm for L^1TV

We now propose an algorithm that minimizes G (see (5.5)). Figure 5.1 shows the level lines of a

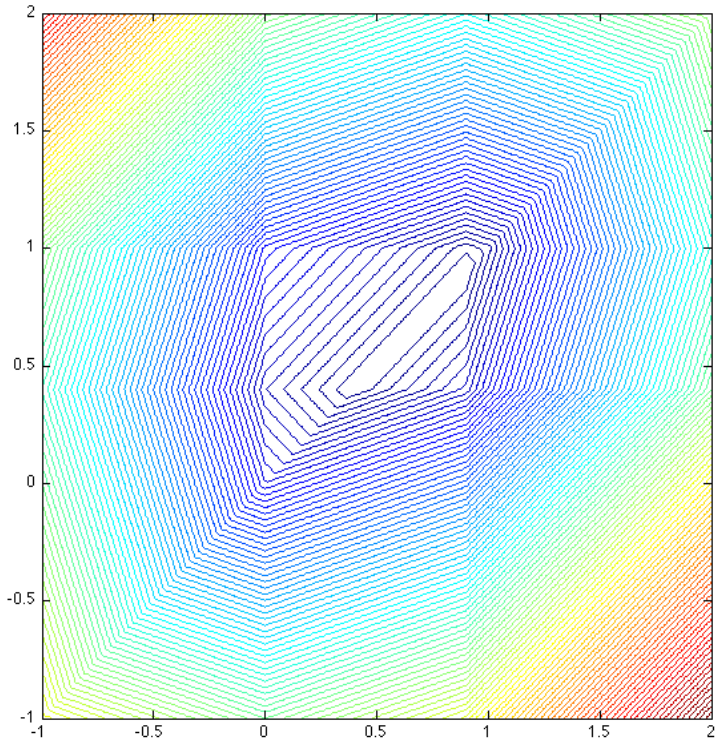


Figure 5.1: Level lines for the function $G(u_1, u_2) = |u_2 - u_1| + |u_1| + |1 - u_2|$, showing the affine nature of the discrete formulation for L^1TV

simple example of G with $\lambda = 1, f = (0, 0, 1, 1)$ and u_0, u_3 fixed. We see the hyperplanes (lines) where G is nonsmooth and the affine nature of G off these hyperplanes. We make use of this affine structure and that the hyperplanes contain the minimizer of G to formulate the hyperplane traversal (ht) algorithm. In the ht algorithm, we do a finite line search in an α-descent direction (Definition 5.3) $d^{(k)}$ by stepping to the closest point in $\mathcal{S}_u \cup \mathcal{S}_f$. We then recompute the α-descent direction in the lower dimensional space to which we just stepped, and continue. Notice that at each iteration

we step to a lower dimensional space which makes the computations easier the further we progress. First, before giving the formal algorithm, we introduce some notation.

Let the coordinate directions, e_i be defined as usual, that is

$$e_i = (0, \ldots, 0, 1, 0, \ldots, 0), \text{ where the 1 is located in the } i^{\text{th}} \text{ position.}$$

Definition 5.3. We define an α direction to be a vector v so that

$$v = \sum_i \alpha_i e_i,$$

where e_i is the ith coordinate direction and $\alpha_i \in \{-1, 0, 1\}$. We say that v is an α descent direction for G at the point u if v is an α direction and $G(u + \tilde{\gamma}v) < G(u)$ for all $0 < \tilde{\gamma} < \gamma$, for some γ.

Algorithm 5.1. *(L^1TV)*
Given $f = (f_1, \ldots, f_m)$*;*
Set $u^{(0)} = (u_1^{(0)}, \ldots, u_m^{(0)}) = (f_1, \ldots f_m)$*;*
Evaluate $G_0 = G(u_1^{(0)}, \ldots, u_m^{(0)})$*;*
Set $k \leftarrow 0$*;*
do
    Compute $d^{(k)}$ *using Algorithm 5.2*
    $\alpha_k \leftarrow \arg\min_\alpha \left\{G(u^{(k)} + \alpha d^{(k)}) : u^{(k)} + \alpha d^{(k)} \in \mathcal{S}_u \cup \mathcal{S}_f\right\}$*;*
    $u^{(k+1)} \leftarrow u^{(k)} + \alpha_k d^{(k)}$*;*
    $k \leftarrow k + 1$*;*
until $d^{(k)} \neq 0$

Table 5.1: L^1TV algorithm

As I stated above, motivating this algorithm is the structure of the function G. There are a finite number of nonparallel hyperplanes on which G is nonsmooth and everywhere else G is affine. We exploit the fact that a minimizer will be on the intersection of at least m of these hyperplanes. The algorithm uses iterative line searches to step between intersections of these hyperplanes eventually reaching the minimizer.

In words, the algorithm works as follows: Start at the point $u = f$. Check coordinate directions

Algorithm 5.2. *(Descent)*
Given $u_1, \ldots, u_m$*;*
$i = 1$*;*
Evaluate $G_0 = G(u^{(0)})$*;*
Set $k \leftarrow 0$*;*
while $i \leq m$
$l_{\max} = \arg\max \left\{ l : u_h^{(k)} = u_i^{(k)},\ \forall i \leq h \leq i + l \right\}$*;*
$v = \sum_{l=i}^{i+l_{\max}} e_l$*;*
$G1 = G\left(u^{(k)} + (u_{i-1}^{(k)} - u_i^{(k)})v\right)$*;*
$G2 = G\left(u^{(k)} + (u_{i+l_{\max}+1}^{(k)} - u_{i+l_{\max}}^{(k)})v\right)$*;*
if $G1 < G$
$d^{(k)} \leftarrow d^{(k)} + sign(u_{i-1}^{(k)} - u_i^{(k)})v$*;*
elseif $G2 < G$
$d^{(k)} \leftarrow d^{(k)} + sign(u_{i+l_{\max}+1}^{(k)} - u_{i+l_{\max}}^{(k)})v$*;*
else
$d^{(k)} \leftarrow 0$*;*
end
$i \leftarrow i + l_{\max} + 1$*;*
end

Table 5.2: α Descent algorithm

and their opposites for descent. Sum all coordinate descent directions to get an α descent direction. Do a finite line search in the α descent direction to step to a hyperplane in $\mathcal{S}_u$ or $\mathcal{S}_f$. At step k, if the algorithm steps to a point in $\mathcal{S}_f$, repeat the above, if the algorithm steps to a point in $\mathcal{S}_u$, project the algorithm to $\mathbb{R}^{\ell_k}$, the space that is isomorphic to the intersections of the hyperplanes of the form $\{u_i^{(k)} = u_j^{(k)}\}$ and repeat.

5.2.4 ht Algorithm Minimizes L^1TV, Proof

Using Lemmas 5.1, 5.5, and 5.6, we know that G has a minimizer in the finite set Y. Below we show that this algorithm finds a minimizer after finitely many iterations. To prove this, we show that if there is a descent direction at a point u, then there exists also an α-descent direction. We also show that whenever there is an α-descent direction, the particular α-descent direction of Algorithm 5.1 exists also and at each of the iterations the algorithm gives strict descent. Next, we show that

the algorithm takes only finitely many steps to get from one point in Y to the next, keeping the algorithm finite. Finally, we show that since Algorithm 5.1 starts at $u^{(0)} = f$, the α directions chosen by the algorithm are indeed α-descent directions.

Wherever descent exists, α-descent exists also.

In [9], the authors define the generalized gradient and generalized derivative which we use to discuss descent directions for G.

Definition 5.4. We define the generalized gradient of a locally Lipschitz function g at a point u to be

$$\partial g(x) = \text{co}\left\{\lim_{i\to\infty} \nabla g(x_i) : x_i \to x, \nabla g(x_i) \text{ exists}\right\}. \tag{5.8}$$

We define the generalized derivative, $g^\circ(u; v)$, of a function g at a point u in the direction v to be

$$g^\circ(u; v) \equiv \limsup_{y\to u, t\searrow 0} \frac{g(y + tv) - g(y)}{t}, \tag{5.9}$$

where $y \in \mathbb{R}^m$ and $t > 0$.

We also take from [9] the following proposition

Proposition 5.1. *Let $g : X \to \mathbb{R}$ be Lipschitz near u. Then for every $v \in X$, we have*

$$f^\circ(x; v) = \max\{\langle\zeta, v\rangle : \zeta \in \partial f(x)\}. \tag{5.10}$$

Because in a neighborhood of each point $u \in \mathbb{R}^m$, there are only finitely many $\nabla g(y)$, the above reduce to

$$\partial g(u) = \text{co}\,\{\nabla g(u_1), \ldots, \nabla g(u_\ell)\} = \left\{\alpha_1 \nabla g(u_1) + \ldots + \alpha_\ell \nabla g(u_\ell) | \alpha_i \geq 0, i = 1...\ell, \sum_i^\ell \alpha_i = 1\right\}. \tag{5.11}$$

and

$$g^{\circ}(u;v) = \max_{(\alpha_1,\ldots,\alpha_\ell)} \left\{ \alpha_1 \nabla g(u_1)\cdot v + \ldots + \alpha_\ell \nabla g(u_\ell)\cdot v | \alpha_i \geq 0, i = 1...\ell, \sum_i^\ell \alpha_i = 1 \right\}.$$

Let $K(u)$ be the cone of descent directions for G at u. We now prove a more general statement about functions that are continuous and piecewise affine.

Lemma 5.7. *Let $g : \mathbb{R}^n \to \mathbb{R}$ be a continuous, piecewise affine (with finitely many pieces) function that is smooth on convex domains. If $g^{\circ}(u;v) < 0$ then v is a descent direction.*

Proof. Let u be a point so that $\nabla g(u)$ does not exist. This means that u is on a section of the boundary of ℓ domains where g is smooth. Because the domains where g is smooth are convex, we can choose points, $u_1, \ldots, u_\ell$, one in each of these domains, so that g is linear along the line segments connecting u and u_i and the using Definition 5.4 we have that if $g^{\circ}(u;v) < 0$ then $\nabla g(u_i)\cdot v < 0$ for $i = 1, ..., \ell$. For otherwise if for some j, $\nabla g(u_j)\cdot v \geq 0$ then we could choose $\alpha_i = 0$ for $i \neq j$ and $\alpha_j = 1$ and $g^{\circ}(u;v) \geq 0$. Let $t_0 > 0$ be small enough so that g is linear along the line $u + tv$ for $0 < t \leq t_0$. Then, we know that

$$g(u_i) - g(u) = g(u_i + tv) - g(u + tv).$$

and so

$$g(u + tv) - g(u) = g(u_i + tv) - g(u_i) = t\nabla g(u_i)\cdot v < 0.$$

Thus, v is a descent direction. □

We recall that G divides $\mathbb{R}^m$ into domains where G is linear in the interior of these domains and G is nonsmooth on the boundary of these domains. Notice if R is one such domain, ∂R is contained in the union of hyperplanes of the form $\{u_i = u_j\}$ and/or $\{u_i = f_i\}$ for some i, j.

Using Lemma 5.7, we prove in the next few lemmas that if at a point u on the boundary of one of these regions there is a descent direction for G, then there is also an α-descent direction that points in the lower dimensional space to which we have stepped.

Lemma 5.8. *As above, let $K(u)$ be the cone of descent directions for G at u.*

a. $v \in K(u)$ *if and only if* $G^\circ(u; v) \leq 0$.

b. If $v \in \partial K(u)$ *then* $G(u + tv) = G(u)$ *for all* $t > 0$ *small enough.*

Proof.

a. First note that using Lemma 5.7 and since G is piecewise affine with finitely many pieces, we get

$$G^\circ(u; v) < 0 \Rightarrow v \in K(u).$$

So we need only show that

$$v \in K(u) \Rightarrow G^\circ(u; v) < 0.$$

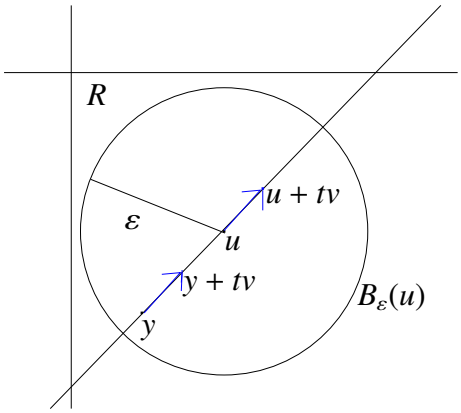
Let $v \in K(u)$. Then $G(u + tv) - G(u) < 0$ for all $t > 0$ small enough. We also know that since G is convex we have that $G(u) - G(u - tv) < 0$ for any $t > 0$. Suppose that at u, $u_i \neq u_j$ for some i, j and $u_k \neq f_k$ for some k, then we choose $\varepsilon > 0$ small enough so that for all $\tilde{u} \in B_\varepsilon(u)$, $\tilde{u}_i \neq \tilde{u}_j$ and $\tilde{u}_k \neq f_k$. Let $y \in B_\varepsilon(u)$. Let R be a region as described above We now break this argument into cases:

Case 1: Suppose $u, u + tv, y, y + tv \in \partial R$. We know then that G is continuous and affine in $B_\varepsilon(u) \cap \partial R$. So we know that $G(y + tv) - G(y) < 0$.

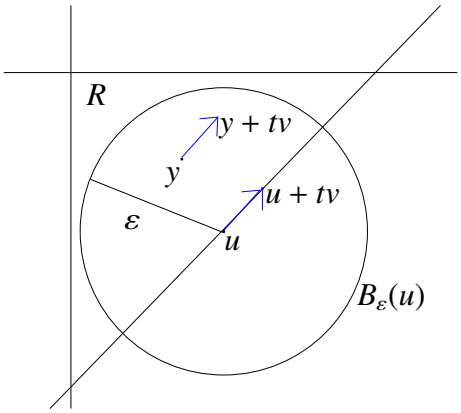
Case 2: Suppose $u, u+tv \in \partial R$, but that $y \notin \partial R$. Since G is affine in $B_\varepsilon(u) \cap \overline{R}$, $G(y+tv)-G(y) < 0$.

Case 3: Suppose $u, y \in \partial R$, but $u + tv, y + tv \notin \partial R$, then using that G is affine in R, we see that $G(y + tv) - G(y) < 0$.

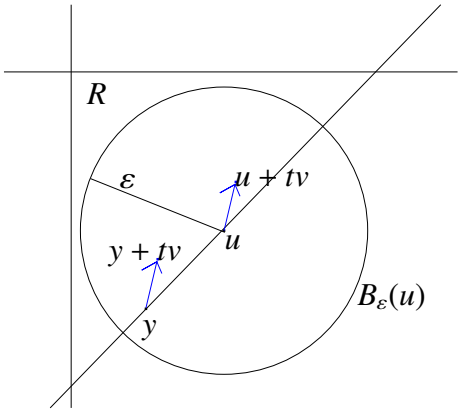
In each of the above cases, we see then that $G^\circ(u; v) < 0$.



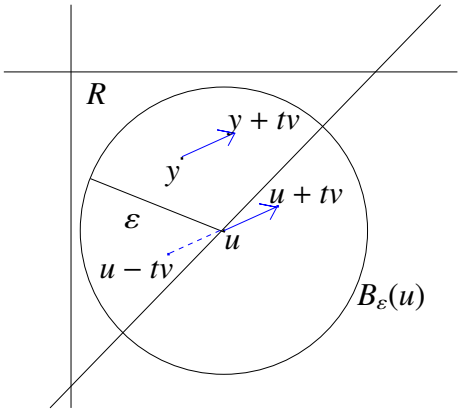
Case 1: $u, u + tv, y, y + tv \in \partial R$



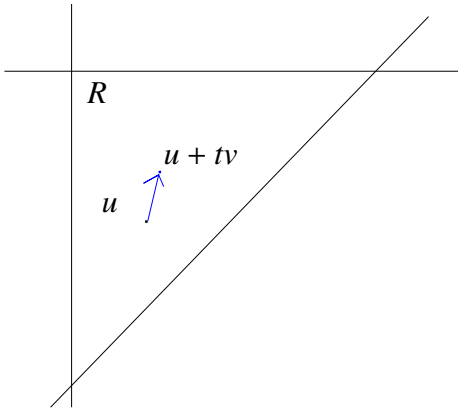
Case 2: $u, u + tv \in \partial R$ but $y \notin \partial R$.



Case 3: $u, y \in \partial R$ but $u + tv, y + tv \notin \partial R$



Case 4: $u \in \partial R$ but $y, u + tv, y + tv \notin \partial R$



Case 5: $u \in R \setminus \partial R$

Figure 5.2: 1D examples for the cases of Lemma 5.8

Case 4: Suppose $u \in \partial R$, but $u + tv, y \notin \partial R$. Then by convexity of G and that G is affine on $B_\varepsilon(u) \cap \overline{R}$, using case 3, we know that

$$G(y + tv) - G(y) = G(u) - G(u - tv) < 0. \tag{5.12}$$

Case 5: Finally, suppose $u \in R \setminus \partial R$, then G is smooth at u so $G^\circ(u; v) = \nabla G(u) \cdot v < 0$.

b. Let $v \in \partial K(u)$ then we can construct a sequence $\{v_k\} \subseteq K(u)$ so that for $v_k \to v$. Then there is a k_0 large enough so that for all $k \geq k_0$, $\|v - v_k\| < \epsilon$ for some $\epsilon > 0$. But $G(u + tv_k) - G(u) < 0$ for all k since $v_k \in K(u)$. G continuous gives us that $G(u + tv) - G(u) < \epsilon$ for all $\epsilon > 0$. Thus $G(u + tv) - G(u) \leq 0$. But if $G(u + tv) - G(u) < 0$ then, by continuity, $v \in K(u)$. So, we have $G(u + tv) - G(u) = 0$

□

Whenever α-descent exists, the α-direction of Algorithm 5.1 exists also.

Definition 5.5. Let $\mathcal{P} \subset \mathbb{R}^m$. We define

$$\pi : \mathcal{P} \to \mathbb{R}^{\tilde{m}} \quad \text{by} \quad \pi(u) = \tilde{u},$$

to be the projection map that removes redundancy in u, where $\tilde{m} \leq m$. That is, if $\{u_i = u_j\}$ is active at u, then the ith or jth (whichever is larger) component is removed from u to get $\tilde{u}$ and if $\{u_i = f_i\}$ is active at u, then the ith component of u is removed to get $\tilde{u}$.

Note, this projection is invertible.

Example 5.1. For example, let $\mathcal{P} \subset \mathbb{R}^4$ be the 2-dimensional subset given by

$$\mathcal{P} = \{(u_1, u_2, u_3, u_4) | u_2 = u_1 \text{ and } u_4 = f_4\}. \tag{5.13}$$

We define $\pi : \mathcal{P} \to \mathbb{R}^2$ by

$$\pi(u_1, u_1, u_3, f_4) = (u_1, u_3). \tag{5.14}$$

If we pick any point in $\mathbb{R}^2$, we can find its inverse projection in $\mathcal{P}$ by

$$\pi^{-1}(u_1, u_2) = (u_1, u_1, u_2, f_4). \tag{5.15}$$

Lemma 5.9. *Let $u \in \partial R$ where R is one of the regions described above. Let $K(u) \neq \emptyset$. Then $\mathcal{N}(u) \cap K(u) \cap \partial R$ has an α descent direction, where $\mathcal{N}(u)$ is some neighborhood of u.*

Proof. We know that ∂R is contained in the union and intersection of some hyperplanes of the form $\{u_i = u_j\}$ and/or $\{u_i = f_i\}$ for some i, j. By looking in $\mathcal{N}(u) \cap K(u) \cap \partial R$ we can restrict G to points in the lower dimensional space, $\mathcal{P}$, defined by the active hyperplanes at u. Let

$$\tilde{G} : \mathbb{R}^{\tilde{m}} \to \mathbb{R}$$

be defined by $\tilde{G}(\tilde{u}) = G(u)$, where $\pi^{-1}(\tilde{u}) = u$.

Then we see that $\nabla\tilde{G}(\tilde{u})$ exists. Since G is affine in $\overline{R}$ we have that $K(\tilde{u}) = \{v : \langle v, \nabla G(u)\rangle < 0\}$ which is a half space and therefore contains an α direction, $\tilde{v}$. We then have that $v = \pi^{-1}(\tilde{v})$ is an α descent direction in $\mathcal{P}$. □

Using the proof above, the following result holds immediately.

Scholium 5.1. *v is an α descent direction at $u \Leftrightarrow \pi(v)$ is an α descent direction at $\pi(u)$.*

Remark 5.2. The above lemma gives us that if at $u^{(k)}$ there is a descent direction for G then there is also an α descent direction for G at $u^{(k)}$.

Lemma 5.10. *If at $\tilde{u} \in \mathbb{R}^{\ell}$ an α descent direction, $\tilde{v}$, exists then there exists an α descent direction of the form*

$$\hat{\tilde{v}} = \sum \alpha_i \tilde{e}_i, \tag{5.16}$$

where $\alpha_i \in \{-1, 0, 1\}$, $\tilde{e}_i$ are coordinate directions, and $\alpha_i \tilde{e}_i$ are descent directions whenever $\alpha_i \neq 0$.

Proof. We can let $\tilde{v} = \sum_{i=1}^{\ell} \alpha_i \tilde{e}_i$ where $\alpha_i \in \{-1, 0, 1\}$ and $\tilde{e}_i$ are coordinate directions. Since $\tilde{G}$ is linear on $\mathbb{R}^\ell$

$$\begin{aligned} G(\tilde{u} + t\tilde{v}) &= G\left(\tilde{u} + t\sum_{i=1}^{\ell} \alpha_i \tilde{e}_i\right) = G\left(\sum_{i=1}^{\ell} (\eta_i \tilde{u} + t\alpha_i \tilde{e}_i)\right) \\ &= G\left(\sum_{i=1}^{\ell} \eta_i \left(\tilde{u} + \frac{t}{\eta_i}\alpha_i \tilde{e}_i\right)\right) = \sum_{i=1}^{\ell} \eta_i G\left(\tilde{u} + \frac{t}{\eta_i}\alpha_i \tilde{e}_i\right), \end{aligned} \tag{5.17}$$

where $\sum_{i=1}^{\ell} \eta_i = 1$. Now, since $\tilde{v}$ is a descent direction, some of the terms, $G\left(\tilde{u} + \frac{t}{\eta_i}\alpha_i \tilde{e}_i\right) < G(\tilde{u})$. For otherwise, if $G\left(\tilde{u} + \frac{t}{\eta_i}\alpha_i \tilde{e}_i\right) \geq G(\tilde{u})$, we would have

$$G(\tilde{u} + t\tilde{v}) = \sum_{i=1}^{\ell} \eta_i G\left(\tilde{u} + \frac{t}{\eta_i}\alpha_i \tilde{e}_i\right) \geq \sum_{i=1}^{\ell} \eta_i G(\tilde{u}) = G(\tilde{u}). \tag{5.18}$$

Let $\mathcal{I} = \{i : G\left(\tilde{u} + \frac{t}{\eta_i}\alpha_i \tilde{e}_i\right) < G(\tilde{u})\}$. Now, we know that $G(\tilde{u} + t\tilde{v}) < G(\tilde{u})$ for all $t > 0$ small enough. Thus, $G(\tilde{u} + \tilde{t}\alpha_i \tilde{e}_i) < G(\tilde{u})$ for $\tilde{t} > 0$ small enough and $i \in \mathcal{I}$. So, $\alpha_i \tilde{e}_i$ is a descent direction whenever $i \in \mathcal{I}$. So, we can create $\hat{\tilde{v}}$, by choosing $\hat{\alpha}$ in the following way

$$\hat{\alpha}_i = \begin{cases} \alpha_i & \text{whenever } i \in \mathcal{I} \\ 0 & \text{otherwise} \end{cases}. \tag{5.19}$$

Then

$$\hat{\tilde{v}} = \sum_{j=1}^{\ell} \hat{\alpha}_i \tilde{e}_i. \tag{5.20}$$

Then

$$G(\tilde{u}) > G(\tilde{u} + t\hat{\tilde{v}}) = G(\tilde{u}) + t\sum_{j=1}^{s} G(\alpha_{i_j} \tilde{e}_i) \tag{5.21}$$

□

Algorithm 5.1 takes only finitely many steps to get back to Y

Lemma 5.11. *Only finitely many steps of the algorithm are needed to get from one point in Y to another.*

Proof. Suppose $u^{(k)} \notin Y$, then $k > 0$ since $u^{(0)} = f \in Y$. Therefore $\nabla G(u^{(k)})$ does not exist because the algorithm always steps to a point where G is nonsmooth. Therefore $u^{(k)} \in \mathcal{H}$, where $\mathcal{H}$ is an $\ell < m$ dimensional hyperplane formed from intersections of some of the $\ell - 1$ hyperplanes of the form $\{u_i = u_j : (i, j) \in E\}$ and or $\{u_i = f_i\}$. Note that $\ell > 1$ since the algorithm stops when $\ell = 1$. If $u^{(k)}$ is not a minimizer there exists a descent direction for G at $u^{(k)}$. Then there exists an α descent direction in $\mathcal{H}$. The algorithm takes the step in this direction to get $u^{(k+1)}$ which lies on a hyperplane whose dimension is smaller than ℓ. We can continue this process at most ℓ times to land at a point $u^{(k')} \in Y$. □

Algorithm 5.1 Convergence Theorem

Combining the above lemmas, we have that G has a minimizer in the finite set Y, that Algorithm 5.1 takes only finitely many steps to get from a point in Y back to another point in Y, if at $u^{(k)}$ G has a descent direction, then the α-descent direction of the algorithm exists also. Finally, we showed that the α-directions found by Algorithm (5.2) are indeed α-descent directions. One key piece to this next theorem is to show that our idea of stepping to lower dimensional spaces does give strict descent.

Theorem 5.1. *Algorithm 5.1 converges to a minimum and is finite.*

α-directions found in Algorithm 5.1 give strict descent.

Recall that in Algorithm 5.1, we step to hyperplanes where G is nonsmooth and then work within this lower dimensional space defined by the hyperplanes to which we have previously stepped. We use clusters to algorithmically define our lower dimensional space. Here we define the notion of a cluster.

Definition 5.6. Let $\mathcal{C}^{(k)} = \left\{c_1 = 1 < c_2 < \ldots < c_{q_k} \leq m : u^{(k)}_{c_i - 1} \neq u^{(k)}_{c_i}\right\}$ be the set of indices so that $u^{(k)}_j = u^{(k)}_{c_i}$ for all $c_i \leq j \leq c_{i+1} - 1$. Then we define a cluster, $C^{(k)}_i$, to be the set of indices j so that $u^{(k)}_j$ has the same value as $u^{(k)}_{c_i}$, that is,

$$C^{(k)}_i := \{j : \text{ for all } c_i \leq j \leq c_{i+1} - 1\}.$$

Notice, that a cluster will have size $|C^{(k)}_i| = c_{i+1} - c_i$ (the last cluster will have size $m - c_{q_k} + 1$) and $\sum_{i=1}^{q} |C^{(k)}_i| = m$. We will say that $u^{(k)}_j$ is in a cluster C_i and mean $j \in C_i$.

Definition 5.7. If $u^{(k)} = \left(u^{(k)}_1, \ldots u^{(k)}_m\right) \in \mathbb{R}^m$ is obtained by k iterations of Algorithm 5.1, we let

$$\alpha^{(k)}_i = \begin{cases} -1 & \text{if } -\sum_{j=c_i}^{c_{j+1}-1} e_j \text{ is a descent direction for } G \text{ at } u^{(k)} \\ 1 & \text{if } \sum_{j=c_i}^{c_{j+1}-1} e_j \text{ is a descent direction for } G \text{ at } u^{(k)} \\ 0 & \text{otherwise} \end{cases}$$

for $1 \leq i \leq m$. For fixed boundary conditions, we set $\alpha^{(k)}_1, \alpha^{(k)}_m = 0$.

The next two lemmas show that our clusters only get larger in Algorithm 5.1 and that we find descent when no point, $u^{(k)}_j$ in cluster C_i will move independently of the cluster. This means that we never go back to the higher dimensional space, that is, the algorithm continues to step to lower and lower dimensional spaces. Actually, these two lemmas are for a more general algorithm, that is, for an algorithm that finds minimizers of L^1pTV for $0 < p \leq 1$. The first of these two lemmas gives us this result for iteration 1 of Algorithm 5.1 and the second lemma gives the result for all other iterations. Both algorithms are proved by looking at the various neighborhood cases that are possible at each point u_i to show that if u_i is in a cluster, C_i, it won't break away from the cluster in next steps. And then we determine which λ values give us descent when moving a cluster C_i.

Clusters need not break up for descent (iteration 1).

Lemma 5.12. *For $0 < p \leq 1$, let us define*

$$G_p(u) \equiv \sum_{i=0}^{m} |u_{i+1} - u_i|^p + \lambda \sum_{i=0}^{m+1} |f_i - u_i|. \tag{5.22}$$

Let $\eta_\ell \equiv |u_{c_i-1} - u_{c_i}|$ and $\eta_r \equiv |u_{c_{i+1}} - u_{c_{i+1}-1}|$. Then the following statements hold.

1. *If there exists a cluster C_i with $\{u_{c_i-1} > u_{c_i}$ and $u_{c_{i+1}-1} > u_{c_{i+1}}\}$ and*

$$0 < \lambda < \frac{\eta_\ell^p - (\eta_\ell - \alpha\eta)^p + \eta_r^p - (\eta_r + \alpha\eta)^p}{\eta|C_i|} \tag{5.23}$$

then a descent direction for G_p, at the point $u = f$, is

$$\sum_{j=c_i}^{c_{i+1}-1} e_j \text{ when } \eta_r < \eta_\ell \quad \text{and} \quad -\sum_{j=c_i}^{c_{i+1}-1} e_j \text{ when } \eta_r > \eta_\ell. \tag{5.24}$$

2. *If there exists a cluster C_i with $\{u_{c_i-1} < u_{c_i}$ and $u_{c_{i+1}-1} < u_{c_{i+1}}\}$ and*

$$0 < \lambda < \frac{\eta_\ell^p - (\eta_\ell + \alpha\eta)^p + \eta_r^p - (\eta_r - \alpha\eta)^p}{\eta|C_i|} \tag{5.25}$$

then a descent direction for G_p, at the point $u = f$, is

$$\sum_{j=c_i}^{c_{i+1}-1} e_j \text{ when } \eta_r > \eta_\ell \quad \text{and} \quad -\sum_{j=c_i}^{c_{i+1}-1} e_j \text{ when } \eta_r < \eta_\ell. \tag{5.26}$$

3. *If there exists a cluster C_i with $\{u_{c_i-1} < u_{c_i}$ and $u_{c_{i+1}-1} < u_{c_{i+1}}\}$ and*

$$0 < \lambda < \frac{\eta_\ell^p - (\eta_\ell - \alpha\eta)^p + \eta_r^p - (\eta_r - \alpha\eta)^p}{\eta|C_i|} \tag{5.27}$$

then a descent direction for G_p, at the point $u = f$, is

$$\sum_{j=c_i}^{c_{i+1}-1} e_j. \tag{5.28}$$

4. *If there exists a cluster C_i with $\{u_{c_i-1} < u_{c_i}$ and $u_{c_{i+1}-1} < u_{c_{i+1}}\}$ and*

$$0 < \lambda < \frac{\eta_\ell^p - (\eta_\ell + \alpha\eta)^p + \eta_r^p - (\eta_r + \alpha\eta)^p}{\eta|C_i|} \tag{5.29}$$

then a descent direction for G_p, at the point $u = f$, is

$$-\sum_{j=c_i}^{c_{i+1}-1} e_j. \tag{5.30}$$

Notice, for $p = 1$, in cases 1 and 2, the condition for λ is $0 < \lambda < 0$. Since there is no such λ, we see that our L^1TV algorithm will not find descent in these cases. The condition for λ for $p = 1$ in cases 3 and 4 is

$$0 < \lambda < \frac{2}{|C_i|}. \tag{5.31}$$

Proof. (of Lemma 5.12) We begin this proof by showing that if we start with $u = f$, to get descent, we need not break up clusters. We break this into 2 cases. We assume for these cases that $1 < c_i < m$, that is u_{c_i} is not a point on the boundary of Ω. We prove this by considering whether or not

$$\begin{aligned} G(u + \eta\alpha e_i) - G(u) = |u_i + \eta\alpha - u_{i-1}|^p + |u_i + \eta\alpha - u_{i+1}|^p + \lambda|u_i + \eta\alpha - f_i| \\ -|u_i - u_{i-1}|^p + |u_i - u_{i+1}|^p + \lambda|u_i - f_i| < 0. \end{aligned} \tag{5.32}$$

Case 1: Suppose $u_{c_i-1} = u_{c_i} = u_{c_i+1}$. We assume that $\eta > 0$ is small and compute

$$G(u + \eta\alpha e_i) - G(u) = 2\eta^p + \lambda\eta > 0, \ \forall\eta > 0. \tag{5.33}$$

Thus, in this case, no descent exists. That is, a data point in the middle of a cluster will not move in the first iteration. We can also see that for each point we move from the inside of a cluster causes

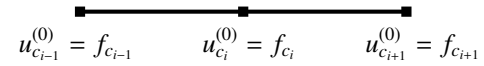


Figure 5.3: Case1: $u^{(0)}_{c_i}$ is a point in the middle of a cluster.

an increase in the fidelity term and we also see that the variation term will not decrease. Thus, we see that no descent is found by moving any points from inside the cluster in a direction different than the rest of cluster.

Case 2: Suppose $u_{c_i-1} = u_{c_i} < u_{c_i+1}$ (See Figure 5.4) We assume that $0 < \eta$ is at most $\eta_r \equiv |u_{c_i+1} - u_{c_i}|$ and compute

$$\begin{aligned} G(u + \eta\alpha e_i) - G(u) = &\quad \eta^p + (\eta_r - \alpha\eta)^p - \eta_r^p + \lambda\eta \\ = &\quad \eta^p(1 - a^p + (a - \alpha)^p) + \lambda\eta \quad \text{where } a = \frac{\eta_r}{\eta} \geq 1 \end{aligned} \tag{5.34}$$

Notice that if $\alpha = -1$, we have $G(u + \eta\alpha e_i) - G(u) = 1 - a^p + (a + 1)^p + \lambda\eta > 0$. Now if $\alpha = 1$, we have $G(u + \eta\alpha e_i) - G(u) = 1 - a^p + (a - 1)^p + \lambda\eta$. This is also positive since $a^p - 1 = (a - 1)^p$ when $a = 1$ and the left-hand side is increasing faster than the right-hand side for $a > 1$. Thus, in

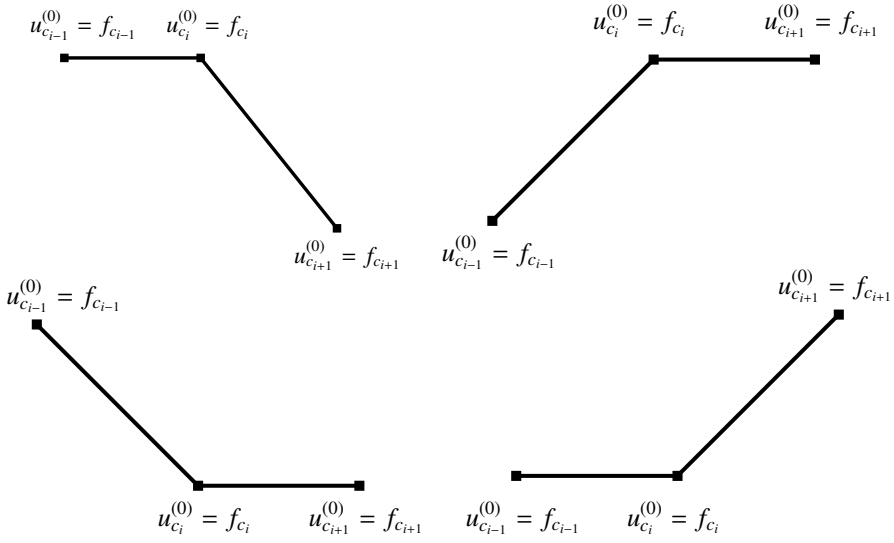


Figure 5.4: Case2: $u^{(0)}_{c_i}$ is a point on the end of a cluster (four cases).

this case, no descent exists. That is, a data point on the end of a cluster will not move in the first iteration. Notice the other cases for $u_{c_i-1} = u_{c_i} \neq u_{c_i+1}$ or $u_{c_i+1} = u_{c_i} \neq u_{c_i-1}$ have the same result and are proved similarly.

Notice that if we moved several points at the end together away from the rest of the cluster, we will see the same result in the variation term, but will multiply the fidelity term by the number of points we move. So, clusters will not break apart in the first iteration of our algorithm. Next we show that when clusters move together in the first iteration we find descent when λ satisfies the conditions stated in the lemma. We show this by considering whether or not

$$\begin{aligned} G\left(u + \eta\alpha \sum_{j=c_i}^{c_{i+1}-1} e_j\right) - G(u) &= |u_{c_i} + \eta\alpha - u_{c_i-1}|^p + |u_{c_{i+1}-1} + \eta\alpha - u_{c_{i+1}}|^p + \lambda \sum_{j=c_i}^{c_{i+1}-1} |u_j + \eta\alpha - f_j| \\ &- |u_{c_i} - u_{c_i-1}|^p + |u_{c_{i+1}-1} - u_{c_{i+1}}|^p + \lambda \sum_{j=c_i}^{c_{i+1}-1} |u_j - f_j| < 0. \end{aligned} \tag{5.35}$$

We break this step into four cases. Let $\eta_r \equiv |u_{c_{i+1}} - u_{c_{i+1}-1}|$ and $\eta_\ell \equiv |u_{c_i} - u_{c_i-1}|$. We also assume $\eta \leq \min\{\eta_r, \eta_\ell\}$.

Case 1 Suppose $u_{c_i-1} > u_{c_i} = \ldots = u_{c_{i+1}-1} > u_{c_{i+1}}$. We compute

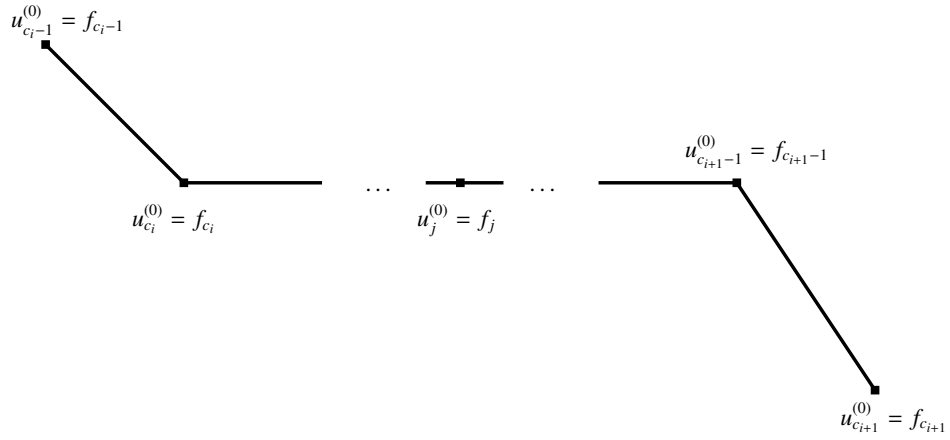


Figure 5.5: Case 1: $C_i(u_i^{(0)})$ has left neighbor above and right neighbor below.

$$G\left(u + \alpha\eta \sum_{j=c_i}^{c_{i+1}-1} e_j\right) - G(u) = (\eta_\ell - \alpha\eta)^p - \eta_\ell^p + (\eta_r + \alpha\eta)^p - \eta_r^p + \lambda|C_i|\eta. \tag{5.36}$$

We see that for this case, we find descent when

$$0 < \lambda < \frac{\eta_\ell^p - (\eta_\ell - \alpha\eta)^p + \eta_r^p - (\eta_r + \alpha\eta)^p}{\eta|C_i|}. \tag{5.37}$$

That is, descent is found, with this λ, by moving the cluster up when $\eta_r < \eta_\ell$ and down when $\eta_r > \eta_\ell$. (For $p = 1$, this condition is $0 < \lambda < 0$, so descent does not exist.)

Case 2: Suppose $u_{c_i-1} < u_{c_i} = \ldots = u_{c_{i+1}-1} < u_{c_{i+1}}$. If we use a similar argument to that of Case 1, we see that for this case, we find descent when

$$\lambda < \frac{\eta_\ell^p - (\eta_\ell + \alpha\eta)^p + \eta_r^p - (\eta_r - \alpha\eta)^p}{\eta|C_i|}. \tag{5.38}$$

That is, descent is found, with this λ, by moving the cluster up when $\eta_r > \eta_\ell$ and down when $\eta_r < \eta_\ell$. (For $p = 1$, this condition is $0 < \lambda < 0$, so descent does not exist.)

Case 3: Suppose $u_{c_i} = \ldots = u_{c_{i+1}-1} < u_{c_i-1}, u_{c_{i+1}}$. We compute

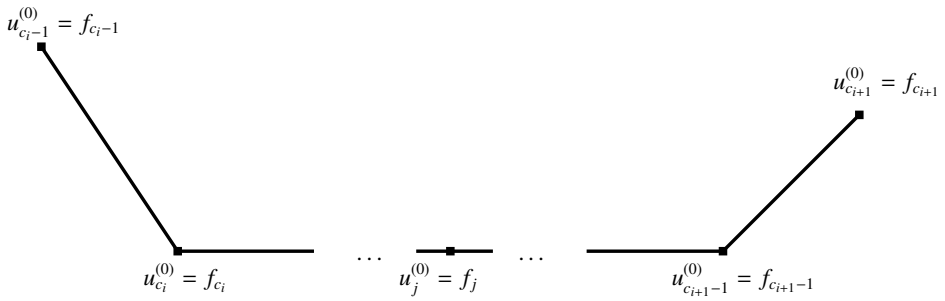


Figure 5.6: Case 3: $C_i(u_i^{(0)})$ has both neighbors above.

$$G\left(u + \alpha\eta \sum_{j=c_i}^{c_{i+1}-1} e_j\right) - G(u) = \quad (\eta_\ell - \alpha\eta)^p - \eta_\ell^p + (\eta_r - \alpha\eta)^p - \eta_r^p + \lambda|C_i|\eta. \tag{5.39}$$

We see that for this case, we find descent by moving the cluster up when

$$\lambda < \frac{\eta_\ell^p - (\eta_\ell - \eta)^p + \eta_r^p - (\eta_r - \eta)^p}{\eta|C_i|}. \tag{5.40}$$

(For $p = 1$, this condition is $0 < \lambda < 2/|C_i|$.)

Case 4: Suppose $u_{c_i} = \ldots = u_{c_{i+1}-1} > u_{c_i-1}, u_{c_{i+1}}$. Again, this case is similar to Case 3. So a similar argument gives us that moving the cluster down gives descent when

$$\lambda < \frac{\eta_\ell^p - (\eta_\ell - \eta)^p + \eta_r^p - (\eta_r - \eta)^p}{\eta |C_i|}. \tag{5.41}$$

(For $p = 1$, this condition is $0 < \lambda < 2/|C_i|$.)

Finally, in the case that we choose the free boundary option (letting boundaries move), we show that if u_{c_i} is on the boundary, we find descent when λ satisfies the conditions stated in the lemma. We prove this for the left endpoint of the data since the argument for the right endpoint is similar. We break this into two cases.

Case 1: $u_1 = u_2$. In this case, we assume $\eta > 0$ is small and we compute

$$G(u + \eta\alpha e_i) - G(u) = \eta^p + \lambda\eta > 0. \tag{5.42}$$

Thus we will not find descent by moving this endpoint without its neighbors.

Case 2: $u_1 = \ldots = u_{c_2-1} < u_{c_2}$. In this case, we assume $0 < \eta \leq \eta_r = (u_{c_2} - u_{c_2-1})$ and we compute

$$G(u + \eta\alpha e_i) - G(u) = |C_1|\lambda\eta + (\eta_r - \alpha\eta)^p - \eta_r^p. \tag{5.43}$$

Thus, we find descent by moving the endpoint up whenever $\lambda < \frac{\eta_r^p - (\eta_r - \eta)^p}{\eta |C_1|} = \frac{\eta_r^p}{\eta |C_1|}$ (For $p = 1$, this condition is $0 < \lambda < 1/|C_1|$.)

Clusters need not break up for descent (iteration k).

For this lemma, we need to define some notation.

Definition 5.8. We define q_g, q_l, q_e to be the number of elements $u_j^{(k)}$, in the cluster that are greater

than, less than, and equal to (respectively) the corresponding f_j:

$$q_g = \left|\left\{u_j^{(k)} \in C\left(u_i^{(k)}\right) : u_j^{(k)} > f_j\right\}\right|,$$
$$q_l = \left|\left\{u_j^{(k)} \in C\left(u_i^{(k)}\right) : u_j^{(k)} < f_j\right\}\right|,$$

and

$$q_e = \left|\left\{u_j^{(k)} \in C\left(u_i^{(k)}\right) : u_j^{(k)} = f_j\right\}\right|.$$

Lemma 5.13. *For $0 < p \leq 1$, let G be as in (5.22) Let q_g, q_e, and q_ℓ be defined as above. Let $\eta_\ell \equiv |u_{c_i-1} - u_{c_i}|$ and $\eta_r \equiv |u_{c_{i+1}} - u_{c_{i+1}-1}|$. And if $u^{(k)}$ is a point obtained using a ht algorithm. Then the following statements hold.*

1. *If there exists a cluster C_i with $\{u_{c_i-1} > u_{c_i} = u_{c_{i+1}-1} > u_{c_{i+1}}\}$ and*

$$0 < \lambda < \frac{\eta_\ell^p - (\eta_\ell - \alpha\eta)^p + \eta_r^p - (\eta_r + \alpha\eta)^p}{\eta((q_g - q_\ell)\alpha + q_e)} \tag{5.44}$$

then a descent direction for G_p, at the point $u^{(k)}$, is

$$\sum_{j=c_i}^{c_{i+1}-1} e_j \text{ when } \eta_r < \eta_\ell \quad \text{and} \quad -\sum_{j=c_i}^{c_{i+1}-1} e_j \text{ when } \eta_r > \eta_\ell. \tag{5.45}$$

2. *If there exists a cluster C_i with $\{u_{c_i-1} < u_{c_i} = u_{c_{i+1}-1} < u_{c_{i+1}}\}$ and*

$$0 < \lambda < \frac{\eta_\ell^p - (\eta_\ell + \alpha\eta)^p + \eta_r^p - (\eta_r - \alpha\eta)^p}{\eta((q_g - q_\ell)\alpha + q_e)} \tag{5.46}$$

then a descent direction for G_p, at the point $u^{(k)}$, is

$$\sum_{j=c_i}^{c_{i+1}-1} e_j \text{ when } \eta_r > \eta_\ell \quad \text{and} \quad -\sum_{j=c_i}^{c_{i+1}-1} e_j \text{ when } \eta_r < \eta_\ell. \tag{5.47}$$

3. *If there exists a cluster C_i with $\{u_{c_i-1} > u_{c_i}$ and $u_{c_i} = u_{c_{i+1}-1} < u_{c_{i+1}}\}$ and*

$$0 < \lambda < \frac{\eta_\ell^p - (\eta_\ell - \eta)^p + \eta_r^p - (\eta_r - \eta)^p}{\eta(q_g - q_\ell + q_e)} \tag{5.48}$$

then a descent direction for G_p, at the point $u^{(k)}$, is

$$\sum_{j=c_i}^{c_{i+1}-1} e_j. \tag{5.49}$$

4. *If there exists a cluster C_i with $\{u_{c_i-1} < u_{c_i}$ and $u_{c_i} = u_{c_{i+1}-1} > u_{c_{i+1}}\}$ and*

$$0 < \lambda < \frac{\eta_\ell^p - (\eta_\ell - \eta)^p + \eta_r^p - (\eta_r - \eta)^p}{\eta(-q_g + q_\ell + q_e)} \tag{5.50}$$

then a descent direction for G_p, at the point $u^{(k)}$, is

$$-\sum_{j=c_i}^{c_{i+1}-1} e_j. \tag{5.51}$$

Again, for $p = 1$, in cases 1 and 2, the condition for λ is $0 < \lambda < 0$. So our L^1TV algorithm will not find descent in these cases. The condition for λ for $p = 1$ in case 3 is

$$0 < \lambda < \frac{2}{q_g - q_\ell + q_e}. \tag{5.52}$$

And for case 4, with $p = 1$, the condition for λ is

$$0 < \lambda < \frac{2}{-q_g + q_\ell + q_e}. \tag{5.53}$$

Proof. (of Lemma 5.13) We begin, again, by showing, to get descent, we need not break up clusters. Again for ease of notation, we write u instead of $u^{(k)}$. We break this into two cases.

Case 1: $u_{c_i-1} = u_{c_i} = u_{c_i+1}$. We assume $\eta > 0$ is small. We compute

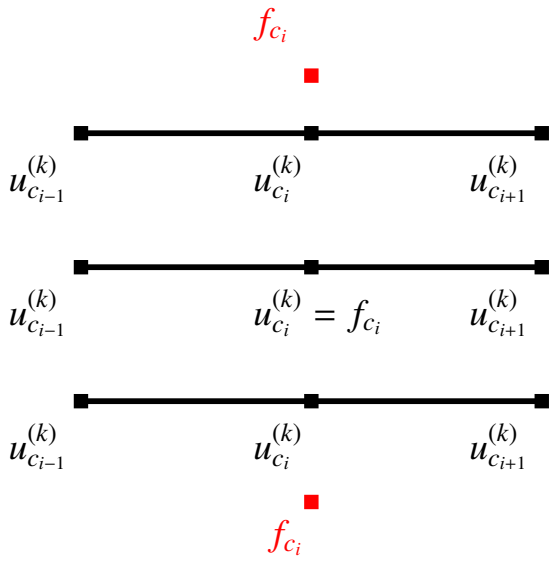


Figure 5.7: Case 1: $u^{(k)}_{c_i}$ is a point in the middle of a cluster (3 possible cases).

$$G(u + \eta\alpha e_i) - G(u) = 2\eta^p + \lambda\eta((q_g - q_\ell)\alpha + q_e). \tag{5.54}$$

Here we treat u_{c_i} as a cluster of size one by moving it alone. So only one of q_g, q_e, q_ℓ is one, while the others are zero. Notice that if $q_e = 1$, there is no descent. Notice also that $\lambda > 2\eta^{p-1}$ gives descent by moving u_{c_i} toward f_{c_i}, but this would be undoing what we did in a previous step. That is, in the previous step, we could have moved u_{c_i} to this cluster by itself in which case moving it back by itself is undoing a step that gave us descent and so it would give us ascent. The other possible case would have been if we moved u_{c_i} with a cluster to this position. In this case, we know from Lemma 5.12 that to move it by itself away would be a step that gives ascent. So, we will not find descent breaking up this cluster.

Case 2: $u_{c_i-1} = u_{c_i} \neq u_{c_i+1}$ or $u_{c_i+1} = u_{c_i} \neq u_{c_i-1}$. We will prove one of these cases, namely $u_{c_i-1} = u_{c_i} < u_{c_i+1}$, because the four cases are similar in argument. We assume that $\eta \leq \min\{\eta_r, |f_{c_i} - u_{c_i}|\}$ and we compute

$$G(u + \eta\alpha e_i) - G(u) = (\eta_r - \eta\alpha)^p - \eta_r^p + \eta^p + \lambda\eta((q_g - q_\ell)\alpha + q_e). \tag{5.55}$$

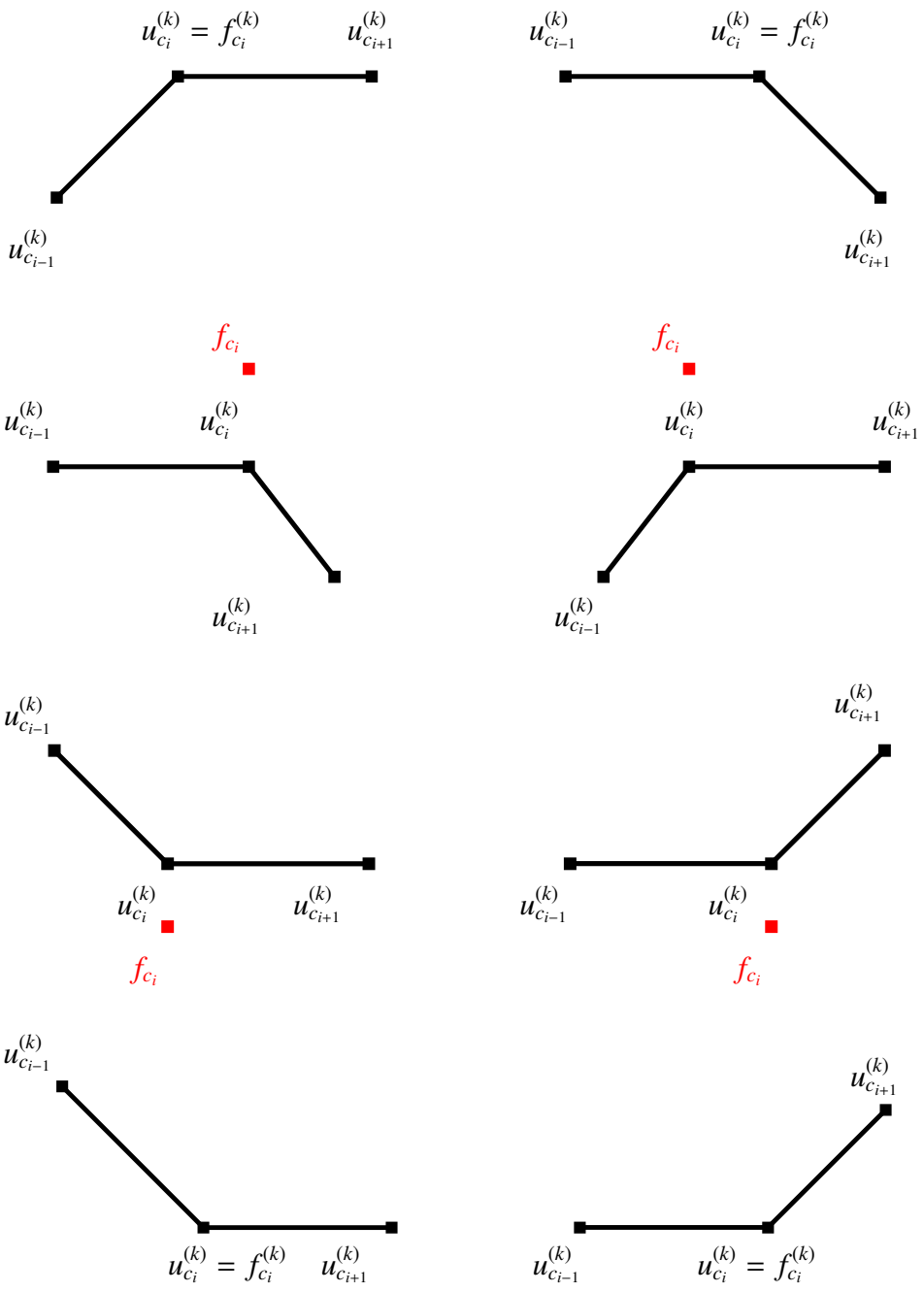


Figure 5.8: Case 2: $u_{c_i}^{(k)}$ is a point on the end of a cluster (8 possible cases).

Notice, we get descent if

$$\lambda < \frac{\eta_r^p - \eta^p - (\eta_r - \eta\alpha)^p}{\eta((q_g - q_\ell)\alpha + q_e)} = \eta^{p-1}\frac{a^p - 1 - (a-\alpha)^p}{(q_g - q_\ell)\alpha + q_e} < 0 \tag{5.56}$$

or

$$\lambda > \frac{-\eta_r^p + \eta^p + (\eta_r - \eta\alpha)^p}{\eta((q_g - q_\ell)\alpha + q_e)} = \eta^{p-1}\frac{a^p - 1 - (a-\alpha)^p}{(q_g - q_\ell)\alpha + q_e} \tag{5.57}$$

But, notice that this second inequality is taking us back toward f_i which is again, undoing a previous step. The first inequality says $\lambda < 0$. Thus, we do not find descent in this case either. So, we know that the algorithm will not break up clusters.

Now we consider moving the full cluster together. We will show that we find descent when λ satisfies the conditions stated in the lemma. We break this step into four cases. Let $\eta_r \equiv |u_{c_{i+1}} - u_{c_{i+1}-1}|$ and $\eta_\ell \equiv |u_{c_i} - u_{c_i-1}|$. We also assume $\eta \leq \min\{\eta_r, \eta_\ell\}$.

Case 1: Suppose $u_{c_i-1} > u_{c_i} = \ldots = u_{c_{i+1}-1} > u_{c_{i+1}}$. We compute

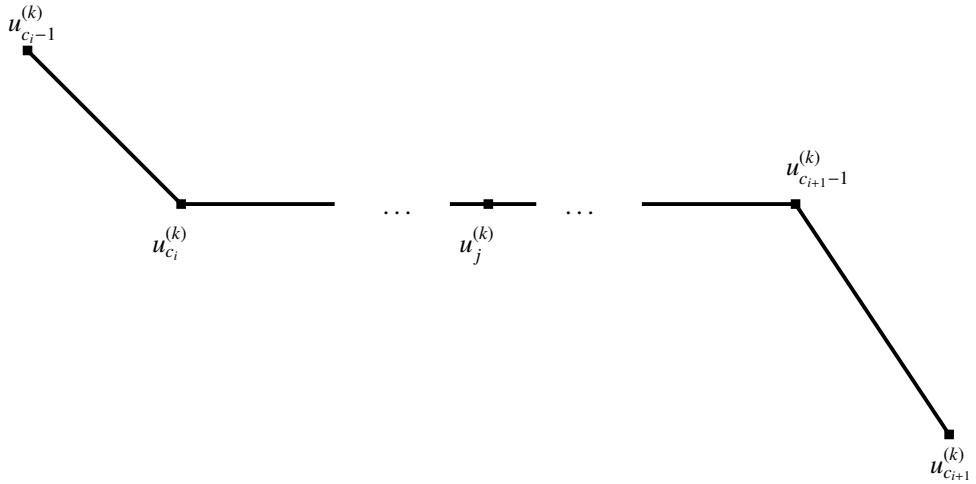


Figure 5.9: Case 1: $C_i(u_i^{(k)})$ has left neighbor above and right neighbor below.

$$G\left(u + \alpha\eta \sum_{j=c_i}^{c_{i+1}-1} e_j\right) - G(u) = \quad (\eta_\ell - \alpha\eta)^p - \eta_\ell^p + (\eta_r + \alpha\eta)^p - \eta_r^p + \lambda((q_g - q_\ell)\alpha + q_e)\eta. \tag{5.58}$$

We see that for this case, we find descent when

$$\lambda < \frac{\eta_\ell^p - (\eta_\ell - \alpha\eta)^p + \eta_r^p - (\eta_r + \alpha\eta)^p}{\eta((q_g - q_\ell)\alpha + q_e)}. \tag{5.59}$$

As in the last lemma, we find descent, with this λ, by moving the cluster up when $\eta_r > \eta_\ell$ and down when $\eta_r < \eta_\ell$. (As we saw in the last lemma, for $p = 1$, this condition is $0 < \lambda < 0$, so descent does not exist.)

Case 2: Suppose $u_{c_i-1} < u_{c_i} = \ldots = u_{c_{i+1}-1} < u_{c_{i+1}}$. If we use a similar argument to that of Case 1, we see that for this case, we find descent when

$$\lambda < \frac{\eta_\ell^p - (\eta_\ell + \alpha\eta)^p + \eta_r^p - (\eta_r - \alpha\eta)^p}{\eta((q_g - q_\ell)\alpha + q_e)}. \tag{5.60}$$

That is, descent is found, with this λ, by moving the cluster up when $\eta_r < \eta_\ell$ and down when $\eta_r > \eta_\ell$. (For $p = 1$ descent does not exist.)

Case 3: Suppose $u_{c_i} = \ldots = u_{c_{i+1}-1} < u_{c_i-1}, u_{c_{i+1}}$. We compute

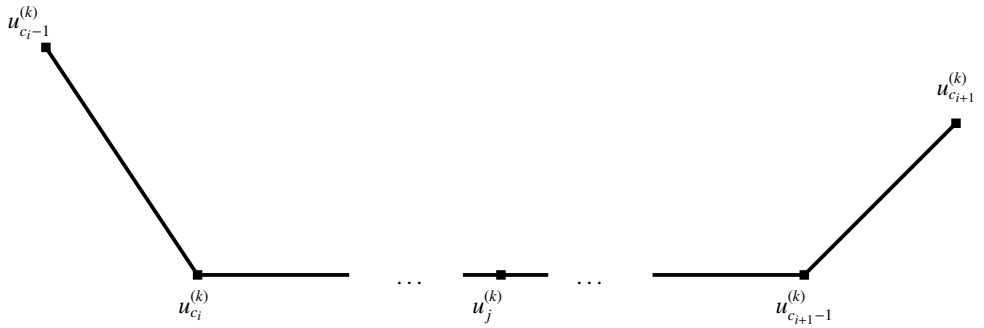


Figure 5.10: Case 3: $C_i(u_i^{(k)})$ has both neighbors above.

$$G\left(u + \alpha\eta \sum_{j=c_i}^{c_{i+1}-1} e_j\right) - G(u) = \quad (\eta_\ell - \alpha\eta)^p - \eta_\ell^p + (\eta_r - \alpha\eta)^p - \eta_r^p + \lambda((q_g - q_\ell)\alpha + q_e)\eta. \tag{5.61}$$

We see that for this case, we find descent by moving the cluster up when

$$\lambda < \frac{\eta_\ell^p - (\eta_\ell - \eta)^p + \eta_r^p - (\eta_r - \eta)^p}{\eta(q_g - q_\ell + q_e)}. \tag{5.62}$$

(For $p = 1$, this condition is $0 < \lambda < 2/(q_g - q_\ell + q_e)$.)

Case 4: Suppose $u_{c_i} = \ldots = u_{c_{i+1}-1} > u_{c_i-1}, u_{c_{i+1}}$. Again, this case is similar to Case 3. So a similar argument gives us that moving the cluster down gives descent when

$$\lambda < \frac{\eta_\ell^p - (\eta_\ell - \eta)^p + \eta_r^p - (\eta_r - \eta)^p}{\eta(-q_g + q_\ell + q_e)}. \tag{5.63}$$

(For $p = 1$, this condition is $0 < \lambda < 2/(-q_g + q_\ell + q_e)$.) □

Using the proof of Lemmas 5.12 and 5.13, we see that for every cluster $C_i^{(k)}$, we get $C_i^{(k)} \subseteq C_i^{(k+1)}$. That is, no point will leave a cluster and at each iteration the algorithm will reduce the problem to minimizing a lower dimensional problem. □

5.2.5 More Efficient L^1TV Algorithm

In this section we will introduce a more efficient algorithm for the case when $p = 1$. In the proof of Lemmas 5.12 and 5.13, we found conditions on λ for which descent occurs. Here we use those conditions to formulate a new algorithm that does not need to compute G values. Recall, we found that descent occurs, in the $p = 1$ case, only when clusters are lower than both neighbors or higher than both neighbors. We restate the conditions here. In the case when C_i is lower than its neighbors, we find descent in moving the cluster up when

$$0 < \lambda < \frac{2}{q_g - q_\ell + q_e}. \tag{5.64}$$

For such clusters, we call $Q = q_g - q_\ell + q_e$ the effective cluster size. In the case when C_i is higher than its neighbors, we find descent in moving the cluster down when

$$0 < \lambda < \frac{2}{-q_g + q_\ell + q_e}. \tag{5.65}$$

For these clusters, we call $Q = -q_g + q_\ell + q_e$ the effective cluster size.

Remark 5.3. For C_i on the boundary of our data, we use the results from Lemmas 5.12 and 5.13 to say the effective cluster size is twice the effective cluster size of an interior cluster. We see that this makes sense since moving the cluster only affects the variation based on one neighbor instead of two and so it takes a smaller λ to move it.

We also recognize that the value of G does not change when a cluster moves between its highest neighbor and its lowest neighbor. So, if instead of moving clusters up and down in parallel, we move up (down) all clusters with the appropriate effective cluster size for the given λ first. These clusters will move up (down) to meet another cluster, stopping at $\mathcal{S}_u$ or to meet an f value, stopping at $\mathcal{S}_f$. Since some clusters will join with others, we recompute effective cluster sizes for all clusters that changed and then move down (up) all clusters with the appropriate effective cluster size for the given λ.

For this version of the algorithm, we are still stepping in an α-descent direction to points in $\mathcal{S}_u$ and/or $\mathcal{S}_f$. We are not breaking up clusters as before, but we are now stepping through effective cluster sizes to make the algorithm more efficient. As long as the effective cluster size does not decrease, we know that the convergence given in Subsection 5.2.3 still holds. In fact, we can easily show that effective cluster size does not decrease.

Lemma 5.14. *The effective cluster size (ECS) for any cluster at any iteration given by*

$$Q_{up} = q_g - q_\ell + q_e \quad \text{or} \quad Q_{dwn} = -q_g + q_\ell + q_e, \tag{5.66}$$

will never decrease, here Q_{up} is the ECS for a cluster intended to move up and Q_{dwn} is the ECS for a cluster intended to move down.

Proof. We will prove this lemma is true for an up cluster C_i. The argument for a down cluster is similar. We recall Definition 5.8.

For this proof, we will say that u_j in C_i contributes to q_g if $u_j > f_j$, to q_ℓ if $u_j < f_j$, and to q_e if $u_j = f_j$. Notice that

- if u_j in C_i contributes to q_g and C_i moves up to form the new cluster C_i', then u_j' in C_i' contributes to q_g' since $u_j' > u_j > f_j$,

- if u_j in C_i contributes to q_e and C_i moves up to form the new cluster C_i' then u_j' in C_i' contributes to q_g' since $u_j' > u_j = f_j$, and

- if u_j in C_i contributes to q_ℓ and C_i moves up to form the new cluster C_i', then u_j' in C_i' contributes to either q_e' or q_ℓ' since a cluster will stop at the closest of its neighbors or corresponding f values so $f_j \geq u_j' > u_j$.

This all tells us that when C_i moves up, q_ℓ can only change by decreasing, q_g can only change by increasing, and q_e can change by either increasing or decreasing.

Now, we consider the effective cluster size for three cases for the newly formed cluster C_i'.

A $|C_i'| = |C_i|$, the actual cluster size does not change. This happens when C_i moves up to meet an f value,

B C_i' is lower than both of its neighbors, and

C C_i' is a cluster that is higher than both of its neighbors.

We don't consider the case when one of the neighbors of C_i' is below and the other above the cluster, since moving this cluster will not give descent in G.

In case A, since both neighboring clusters, C_{i-1} and C_{i+1} are still above C_i, the effective cluster size is given by Q_{up} in (5.66). Using the argument above we see that the new effective cluster size increases since at least one u_j in C_i that contributes to q_ℓ will move up to u_j' that contributes to q_e thus Q_{up} will increase.

In case B, C_i will move up to join with at least one of its neighboring clusters C_{i-1} and C_{i+1}. Now, let Q_{i-1}, Q_{i+1} denote the effective cluster sizes of C_{i-1}, C_{i+1}, respectively and Q_i' be the contribution from C_i after its move. From the above argument, we know that $Q_i' = q_g' - q_\ell' + q_e' \geq q_g - q_\ell + q_e = Q_i$.

If C_i moves up to join with C_{i-1}, the new effective cluster size is just the sum of the contributions from both clusters, that is, $Q = Q_{i-1} + Q'_i$. If C_i moves up to join with C_{i+1}, the new effective cluster size is $Q = Q'_i + Q_{i+1}$. And if C_i moves up to join with both C_{i-1} and C_{i+1}, the new effective cluster size is $Q = Q_{i-1} + Q'_i + Q_{i+1}$. In these three cases, if $Q_{i-1}, Q_{i+1} > 0$ then the effective cluster size does not decrease.

Notice that case C can only happen if C_i moves up to meet both of its neighbors (for otherwise, at least one will still be above C'_i). So, the new effective cluster size is $Q = Q_{i-1} + Q'_i + Q_{i+1}$. Also, for this case, the new cluster that is formed is a down cluster, that is Q is computed using Q_{dwn} in (5.66). Since C_i was below clusters C_{i-1} and C_{i+1} before the move, we know that C_{i-1} and C_{i+1} were down clusters before C_i moved up. So, since we are incrementing on the ECS, we know that $Q_{i-1}, Q_{i+1} \geq Q_i$. Since the newly formed cluster, C'_i is a down cluster, the amount that C_i contributes to Q is $Q'_i = -q_g + q_\ell + q_e$. Notice that Q'_i is not an effective cluster size, rather it only contributes to the new effective cluster size so it may be negative. If Q'_i is negative, then we will get the smallest value for Q. But the smallest this can be happens when $u_j = f_j$ for all u_j in C_i so that after the move they contributed to q_g, but then $Q_i = |C_j|$ and we get $Q = Q_{i-1} + Q'_i + Q_{i+1} = Q_{i-1} - |C_i| + Q_{i+1} \geq Q_i$. So, we know that, in this case also, the effective cluster size never decreases.

Notice that since the algorithm starts with $u = f$, $Q_i = |C_i| > 0$ for every cluster C_i thus using the above arguments, the minimum effective cluster sizes never decrease. □

Now we give the formal algorithm. Let $C_1^{(k)}, \ldots, C_{q_k}^{(k)}$ be the unique clusters at iteration k. Let $g_i^{(k)}, e_i^{(k)}, \ell_i^{(k)}$ be q_g, q_e, and q_ℓ for cluster i at iteration k. In this algorithm we start at $u^{(0)} = f$ and find the clusters. We, then, determine which clusters might move up, call them up clusters, and which might move down, call them down clusters (ignoring those that have both a neighbor below and a neighbor above the cluster). In the case of Algorithm 5.3, we see it is written with a preference to move clusters up first and then down. We find the minimum effective cluster size (ECS) and move up any up cluster, with this ECS, to its nearest neighboring cluster or f value. If no up cluster has this ECS, I move down any down clusters, that have the same ECS, to its nearest neighboring cluster or f value. We repeat this until the stopping condition is reached. If at any iteration the

Algorithm 5.3. *(L^1TV)*
Given $f = (f_1, \ldots, f_m)$*;*
Set $u^{(0)} = (u_1^{(0)}, \ldots, u_m^{(0)}) = (f_1, \ldots f_m)$*;*
Find $C_1^{(0)}, C_2^{(0)}, \ldots, C_{q_0}^{(0)}$*;*
Set $k \leftarrow 1$*;*
do
  Compute $g_i^{(k)}, e_i^{(k)}, \ell_i^{(k)}$ *for each* $i = 1...q_0$*;*
  $U \leftarrow \{j :$ *all nbrs of* C_j *are above* $C_j\}$
  $D \leftarrow \{j :$ *all nbrs of* C_j *are below* $C_j\}$
  $mincs_k \leftarrow \min_{1\le i\le q_k}\{\min_{i\in U}\{g_i^{(k)} - \ell_i^{(k)} + e_i^{(k)}\}, \min_{i\in D}\{-g_i^{(k)} + \ell_i^{(k)} + e_i^{(k)}\}\}$*;*
  $mvcl \leftarrow \{i \in U : g_i^{(k)} - \ell_i^{(k)} + e_i^{(k)} = mincs\}$*;*
  if $mvcl \neq \emptyset$
    for $idx = 1 : |mvcl|$
      Move up, $C_{mvcl(idx)}$ *to closest of* f*,* $C_{I(idx)-1}$*, and* $C_{I(idx)+1}$
    end
  else
    $mvcl \leftarrow \{i \in D : -g_i^{(k)} + \ell_i^{(k)} + e_i^{(k)} = mincs\}$*;*
    for $idx = 1 : |mvcl|$
      Move down, $C_{mvcl(idx)}$ *to closest of* f*,* $C_{I(idx)-1}$*, and* $C_{I(idx)+1}$
    end
  end
  $k \leftarrow k + 1$
  Update list of clusters, $[C_1^{(k)}, \ldots, C_{q_k}^{(k)}]$*;*
  if $mincs_k \neq mincs_{k-1}$
    Append list of solutions with $[C_1, \ldots, C_{q_k}]$*;*
    Append list of λ *with* $\frac{2}{mincs+1}$*;*
  end
until *no descent exists.*

Table 5.3: Efficient L^1TV algorithm (written with an up preference)

minimum effective cluster size changed from the previous iteration, we update the list of solutions and the λ value.

The stopping condition for this algorithm depends on the type of data and the boundary conditions. In the general case for data (that is, not necessarily binary data), we stop the algorithm when monotonicity is reached for fixed boundary data or, for free boundary data, when there is only one cluster left (the solution is flat).

For binary data, this algorithm is greatly simplified. There is never a need to check nbrs of a

cluster. If the data is binary, the neighbors have the opposite value of that of the cluster. That is, if the cluster is at a height of 1, its neighbors are at a height of 0 and it is then a down cluster. For down clusters, C_i, the effective cluster size is dependent on ℓ_i and e_i whereas for an up cluster, the effective cluster size is dependent on g_i and e_i. Another simplification for this algorithm when the data is binary is that we never need to check the distance to f values corresponding to a cluster since these will also be either 0 or 1. So, the algorithm will not have moves to heights other than the height of cluster neighbors. Finally, the algorithm will stop when the minimum number of clusters, minq, is reached. The value of minq depends on whether the boundaries are fixed (minq= 2) or free (minq= 1).

In Figure 5.11, we show the value of $\min G$ vs λ for two different examples using Algorithm 5.3. The top plot is from Example 6.2, a simple noisy signal with two jumps of differing heights. The bottom plot is from Example 6.6, a signal composed of the sum of sines and gaussian stationary additive noise. These plots are obtained by finding the G value after each iteration. The points in Figure 5.11 are the points $(\lambda, \min G)$. Because the minimizer will be the same until λ drops below the λ_{cut} for the next iteration and since G is linear in λ we plot the line between λ_k and λ_{k+1} using the minimizer for λ_k to compute G.

The greatest benefit to Algorithm 5.3, for both the general and the binary problems, is that we are able to solve the $\lambda = 0$ problem and in the process get the solutions $\forall \lambda > 0$. That is, the computational task of getting a solution for all $\lambda > 0$, is the same as solving only one problem. We state this result in the next theorem.

Theorem 5.2. *Algorithm 5.3 finds a solution to L^1TV for every $\lambda > 0$.*

Proof. Algorithm 5.3 iterates by increasing the effective clusters size, thus decreasing λ beginning with the largest possible λ so that at least one cluster will move. Because the problem is discrete, the effective cluster sizes are positive integers between 0 and m (the length of the signal). The effective cluster size (and thus, λ) does not change until no cluster of this effective cluster size will move. By the results of Subsection 5.2.4 we know that, for each λ, Algorithm 5.3 finds descent

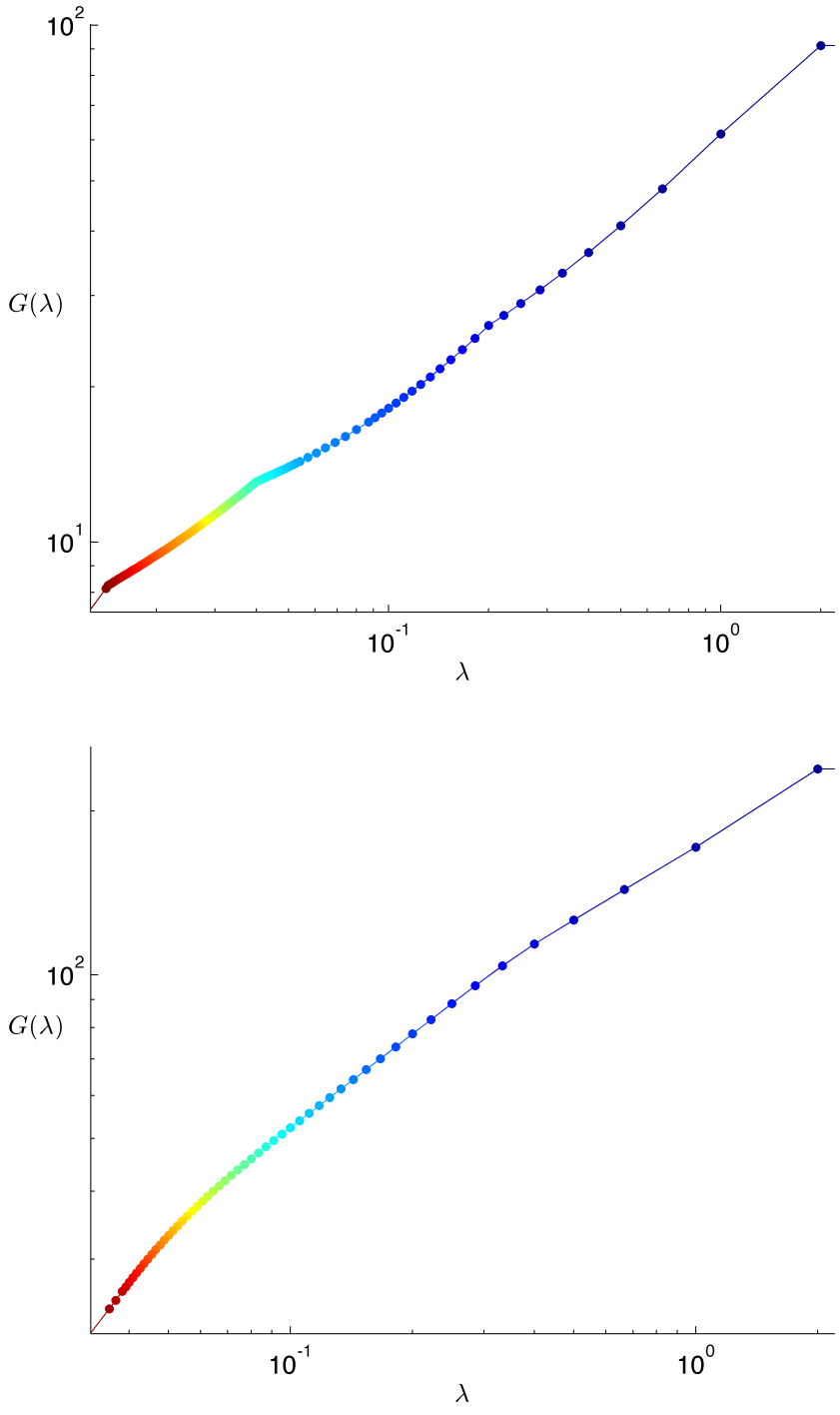


Figure 5.11: $(\lambda, \min G)$ corresponding to Exs 6.2 (top) and 6.6 (bottom)

whenever descent exists so, at each iteration, the algorithm minimize L^1TV for the current λ. And by Lemma 5.14, we know that iterating on the effective cluster size does not skip a particular effective cluster size that might need to be revisited later since the effective cluster size never decreases. Thus, for each $\lambda > 0$, we find a minimizer to the corresponding L^1TV problem. □

5.2.6 ht Algorithm in Higher Dimensions

It is worth noting that neither algorithm will extend to higher dimensions. The issue lies in the neighborhood structure that occurs at higher dimensions. In higher dimensional problems, such as imaging problems, it becomes beneficial to break up clusters when there is a data point that has more neighbors outside of the cluster than inside. We see this occurring in images with L^1TV when parts of object edges having high curvature are rounded. We conjecture that an adjustment to the algorithm to allow cluster break up only when the number of neighbors outside the cluster is not less than the number inside will give similar results for higher dimensional data.

5.2.7 Time Trials for L^1TV ht Algorithm

Finally, we show timing results for both the general and binary cases as well as for fixed and free boundary conditions.

First, we start with fixed boundary conditions. We ran Algorithm 5.3 on 100 random signals of size N. In Table 5.4 we have recorded the average number of initial clusters, the average number of λ solutions, and the average time in seconds that it takes to perform the main loop of the ht algorithm. The algorithm has a set up that is of order N, but then the main loop depends on the number of initial clusters.

We then ran 100 random binary signals of length N. We recorded the average time to complete the main loop of the ht algorithm, the average number of initial clusters, and average number of λ solutions in Table 5.5.

Next, we looked at some time trials, but this time with free boundary conditions. We ran

N	Ave. # of initial clusters	Ave. # of λ Solutions	Ave. time in seconds
5	5	2.61	0.0008
10	10	3.65	0.0015
20	20	5.93	0.0031
40	40	9.21	0.0061
80	80	12.62	0.0114
160	160	23.58	0.0229
320	320	39.27	0.0455
640	640	63.76	0.0941
1280	1280	135.61	0.2153
2560	2560	283.39	0.5410
5120	5120	418.02	1.4511

Table 5.4: Time Trials for Algorithm 5.3 for general random signals, of size N, with fixed boundary conditions.

N	Ave. # of initial clusters	Ave. # of λ Solutions	Ave. time in seconds
5	2.1	1.39	0.0001
10	3.9	2	0.0002
20	7.4	2.51	0.0004
40	13.8	2.93	0.0007
80	26.9	3.47	0.0012
160	52.2	3.93	0.0020
320	104.4	4.43	0.0035
640	209.1	4.91	0.0067
1280	419.1	5.34	0.0138
2560	836.2	5.91	0.0305
5120	1677.1	6.42	0.0762

Table 5.5: Time Trials for Algorithm 5.3 for random binary signals, of size N, with fixed boundary conditions.

100 random signals of length N. In Figures 5.6 (general random signals) and 5.7 (binary random signals), we recorded the average time to complete the main loop, the average number of initial clusters, and the average number of λ solutions.

N	Ave. # of initial clusters	Ave. # of λ Solutions	Ave. time in seconds
5	5	3.26	0.0010
10	10	4.28	0.0020
20	20	5.93	0.0035
40	40	8.11	0.0063
80	80	11.15	0.0117
160	160	15.15	0.0217
320	320	20.58	0.0417
640	640	30.22	0.0857
1280	1280	41.72	0.1874
2560	2560	58.94	0.4587
5120	5120	84.78	1.2875

Table 5.6: Time Trials for Algorithm 5.3 for general random signals, of size N, with free boundary conditions.

N	Ave. # of initial clusters	Ave. # of λ Solutions	Ave. time in seconds
5	2.3	1.85	0.0002
10	4	2.33	0.0003
20	6.7	2.71	0.0005
40	13.8	3.19	0.0008
80	26.8	3.51	0.0012
160	53.8	4.03	0.0020
320	106.7	4.54	0.0036
640	211.4	4.87	0.0068
1280	420.8	5.36	0.0138
2560	835.8	5.85	0.0305
5120	1678.3	6.40	0.07771

Table 5.7: Time Trials for Algorithm 5.3 for random binary signals, of size N, with free boundary conditions.

From our time trials, it appears that the computational complexity of the main loop of our ht algorithm is of order $o(M)$, where M is the number of initial clusters in our signal. An initial computation of order $o(N)$ is performed on each signal to catalogue the clusters, where N is the length of the signal. Putting these together, we believe that the computational complexity of this

algorithm is of order $o(aN + M)$ signals of length N, where a is small compared to 1. We know that for a binary signal, the cluster that moves will meet up with both its neighbors. So, after each iteration, the number of clusters decreases by 2 for every cluster that moves. This means that there are exactly $\frac{M}{2}$ cluster moves for any binary signal. The worst case scenario, is the binary signal given by

$$u^{(0)} = (0, 1, 0, \ldots).$$

Here the number of initial clusters is N, the signal length. Thus, the number of cluster moves is equal to $\frac{M}{2} = \frac{N}{2}$. In a random binary signal, we expect less than N initial clusters.

5.3 L^1pTV ($p < 1$) in 1-Dimension

In this section, we discuss the discrete formulation for L^1pTV,

$$\min \int_\Omega |u'|^p + \lambda |f - u| \, dx, \quad \text{for } 0 < p < 1. \tag{5.67}$$

In [8], Chartrand uses (5.67) to denoise cartoon (piecewise constant) images. He found that boundaries and pixel intensity of objects in the image are preserved even in areas of high curvature such as corners. We use an our ht algorithm on one dimensional signals to see the same preservation in objects.

Here we consider a regular fixed partition for Ω, $x_0, x_1, \ldots, x_m, x_{m+1}$ and again, consider piecewise linear functions u. We use $u' = \frac{u_{i+1} - u_i}{x_{i+1} - x_i}$ to get

$$\min \left\{ G = \sum_{i=0}^{m} |u_{i+1} - u_i|^p + \lambda \sum_{i=1}^{m} |f_i - u_i| : u \in \mathcal{F} \right\}. \tag{5.68}$$

Notice we can eliminate the x dependence for the same reason we were able to eliminate the x dependence in Section 5.2. Notice also that if u is binary data, the problem is p-independent. Thus, for binary data, this problem reduces to the L^1TV problem.

For this problem, we seek a minimizer in the set Y (Definition 5.2). Because G is concave in the regions separated by the hyperplanes of the form $\{u_i = u_{i+1}\}$, $\{u_i = u_{i-1}\}$, or $\{u_i = f_i\}$ (See Figure 5.12), we know that minimizers will be in Y. We use an ht algorithm similar to the above which stays in Y to find local minimizers. The difference in the algorithm is that it is dependent on the neighborhood structure, that is, whether the neighbors are both above, both below, or one above and one below the cluster being moved.

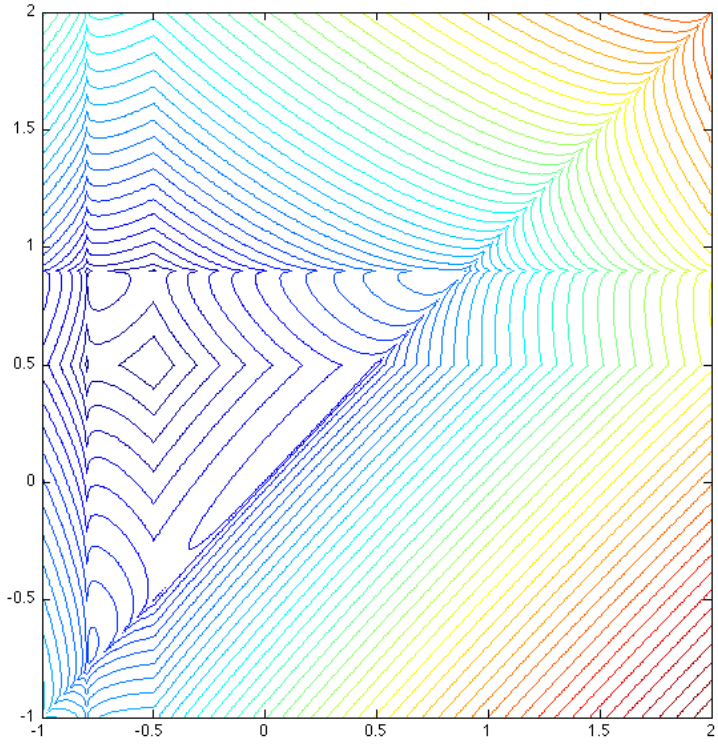


Figure 5.12: Level lines of a simple example of G for $p = .5$ and $\lambda = 1$

5.3.1 ht Algorithm for L^1pTV

Again, we use the proof of Lemmas 5.12 and 5.13 to write an algorithm to find minimizers of (5.68). We let $C_1^{(k)}, \ldots, C_{q_k}^{(k)}$ be the unique clusters at iteration k. Using the lemmas, we write the conditions on λ for a cluster to move at iteration k. We let q_g, q_e, and q_ℓ be as we defined them in

Subsection 5.2.5. To move a cluster that has one neighbor above and one neighbor below, we let η_a be the distance from the cluster to the neighbor above the cluster and η_b be the distance from the cluster to the neighbor below the cluster. And we consider moving the cluster up a distance $\eta = \min\{\eta_a, \{f_i - u_i : \text{ when } f_i > u_i, u_i \in C_i\}\}$. The algorithm will find descent in this case if

$$0 < \lambda < \frac{\eta_a^p - (\eta_a - \eta)^p + \eta_b^p - (\eta_b + \eta)^p}{\eta(q_g - q_\ell + q_e)}. \tag{5.69}$$

Notice that since $q_g - q_\ell + q_e > 0$, we need

$$0 < \eta_a^p - (\eta_a - \eta)^p + \eta_b^p - (\eta_b + \eta)^p. \tag{5.70}$$

This happens only when $\eta_a < \eta_b$. Now we consider moving the cluster up a distance $\eta = \min\{\eta_b, \{u_i - f_i : \text{ when } f_i < u_i, u_i \in C_i\}$. The algorithm will find descent in this case if

$$0 < \lambda < \frac{\eta_a^p - (\eta_a + \eta)^p + \eta_b^p - (\eta_b - \eta)^p}{\eta(-q_g + q_\ell + q_e)}. \tag{5.71}$$

Similar to the previous case, we need that $\eta_b < \eta_a$ for this to make sense. To move a cluster that has both neighbors above, we let η_ℓ be the distance from the cluster to its left neighbor and η_r be the distance from the cluster to its right neighbor. And we consider moving the cluster up a distance $\eta = \min\{\eta_\ell, \eta_r, \{f_i - u_i : \text{ when } f_i > u_i, u_i \in C_i\}$. The algorithm will find descent in this case if

$$0 < \lambda < \frac{\eta_r^p - (\eta_r - \eta)^p + \eta_\ell^p - (\eta_\ell - \eta)^p}{\eta(q_g - q_\ell + q_e)}. \tag{5.72}$$

Finally, To move a cluster that has both neighbors below, we let η_ℓ be the distance from the cluster to its left neighbor and η_r be the distance from the cluster to its right neighbor. And we consider moving the cluster up a distance $\eta = \min\{\eta_\ell, \eta_r, \{u_i - f_i : \text{ when } f_i < u_i, u_i \in C_i\}$. The algorithm

will find descent in this case if

$$0 < \lambda < \frac{\eta_r^p - (\eta_r + \eta)^p + \eta_\ell^p - (\eta_\ell + \eta)^p}{\eta(-q_g + q_\ell + q_e)}. \tag{5.73}$$

Using the above information, we see that our algorithm should only check the conditions for moving a cluster up when the closest neighbor is above the cluster. Similarly, we should only check the condition for moving a cluster down when the closest neighbor is below the cluster. As in Algorithm 5.3, we let g_i, e_i, ℓ_i be q_g, q_e, q_ℓ for cluster, i. In the algorithm below, we let $Q_{up} = \{g_i - \ell_i + e_i : 1 \le i \le m\}$ and $Q_{dwn} = \{-g_i + \ell_i + e_i : 1 \le i \le m\}$.

Algorithm 5.4. *(L^1pTV)*
Given $f = (f_1, \ldots, f_m)$*;*
Set $u^{(0)} = (u_1^{(0)}, \ldots, u_m^{(0)}) = (f_1, \ldots f_m)$*;*
Find $C_1^{(0)}, C_2^{(0)}, \ldots, C_{q_0}^{(0)}$*;*
Compute Set $k \leftarrow 1$*;*
for $i = 1 : m$
    Compute λ_i *according to Eqs. (5.69), (5.71), (5.72), and (5.73));*
end
do
    Find $mvcl = \arg\max_i\{\lambda_i\}$*;*
    Set $\lambda_k^* \leftarrow \lambda_{mvcl}$*;*
    if $\lambda_k^* < 0$*, stop* ***do*** *loop*
    Set $\eta_r \leftarrow |u_r - u|, \eta_\ell \leftarrow |u_\ell - u|$*;*
    $\eta \leftarrow \min\left\{\eta_r, \eta_\ell, \min_{f_j, u_j \in C_{mvcl}}\left\{|f_j - u_j| : \alpha(f_j - u_j) > 0\right\}\right\}$*;*
    Compute α_{mvcl} *according to 5.5 using* $u_r, u_\ell, Q_{up_i}, Q_{dwn_i}, \eta_r, \eta_\ell$*;*
    Move C_{mvcl} *the distance* $\eta\alpha_{mvcl}$
    $k \leftarrow k + 1$
    Update list of clusters, $[C_1^{(k)}, \ldots, C_{q_k}^{(k)}]$*;*
    Compute λ_i *for clusters that change;*
    if $\lambda_k^* < \lambda_{k-1}^*$
        Append list of solutions with $[C_1, \ldots, C_{q_k}]$*;*
        Append list of λ^* *with* λ_k^**;*
    end
until *no descent exists.*

Table 5.8: L^1pTV algorithm

Algorithm 5.5. *(α)*
Given $u, u_r, u_\ell, Q_{up}, Q_{dwn}, \eta_r, \eta_\ell$*;*
if $u_r, u_\ell > u$
    $\alpha \leftarrow 1$*;*
elseif $u_r > u$ *and* $\eta_r < \eta_\ell$
    $\alpha \leftarrow 1$*;*
elseif $u_\ell > u$ *and* $\eta_\ell < \eta_r$
    $\alpha \leftarrow 1$*;*
elseif $\eta_\ell = \eta_r$
    if $u_\ell > u$ *or* $u_r > r$
        if $0 \leq Q_{up} < Q_{dwn}$ *or* $Q_{dwn} < 0 \leq Q_{up}$
            $\alpha \leftarrow 1$*;*
        elseif $Q_{up} = Q_{dwn}$
            $\alpha \leftarrow$ *preferred direction (up=1,down=-1);*
        end
    end
else
    $\alpha \leftarrow -1$*;*
end

Table 5.9: L^1pTV algorithm

In words, this algorithm starts at $u^{(0)} = f$ and finds all of the clusters. The cluster with the maximum λ_{cut} (according to Equations (5.69), (5.71), (5.72), and (5.73)) is moved in the appropriate direction to the nearest neighbor or nearest f_i for $i \in \{i : u_i \in C_i\}$. We continue this until stopping conditions are reached. As in the L^1TV algorithm, stopping conditions depend upon the type of boundary conditions, that is, whether they are fixed or free boundary conditions.

Notice that since the iterations of Algorithm 5.4 stay in Y (see Definition 5.2), we know that it is also a finite ht algorithm. From Lemmas 5.12 and 5.13 we know that at each iteration, the algorithm finds strict decent. $G \geq 0$, so is bounded below. We can see that G is coercive. Indeed, we have

$$G = \sum_{i=0}^{m} |u_{i+1} - u_i|^p + \lambda \sum_{i=1}^{m} |f_i - u_i| \leq \lambda \sum_{i=1}^{m} |f_i - u_i| \leq \lambda \sum_{i=1}^{m} (|u_i| - |f_i|) \to \infty \text{ as } |u| \to \infty. \quad (5.74)$$

We do not have a convexity condition so we are only able to conclude that this algorithm finds

local minima.

Like Algorithm 5.1, Algorithm 5.4 steps to the hyperplanes where G is nonsmooth and stays in the lower dimensional space. So, this algorithm finds minimizers of lower and lower dimensional problems. The clusters also only get larger, therefore because we operate on clusters rather than the signal, the algorithm increases in efficiency as we progress through the iterations. Because we cannot guarantee that the effective cluster size is nondecreasing, we cannot be assured that we get a local minimizer for every $\lambda > 0$. We can also see from Figures 5.13, 5.14, 5.15, and 5.16 that

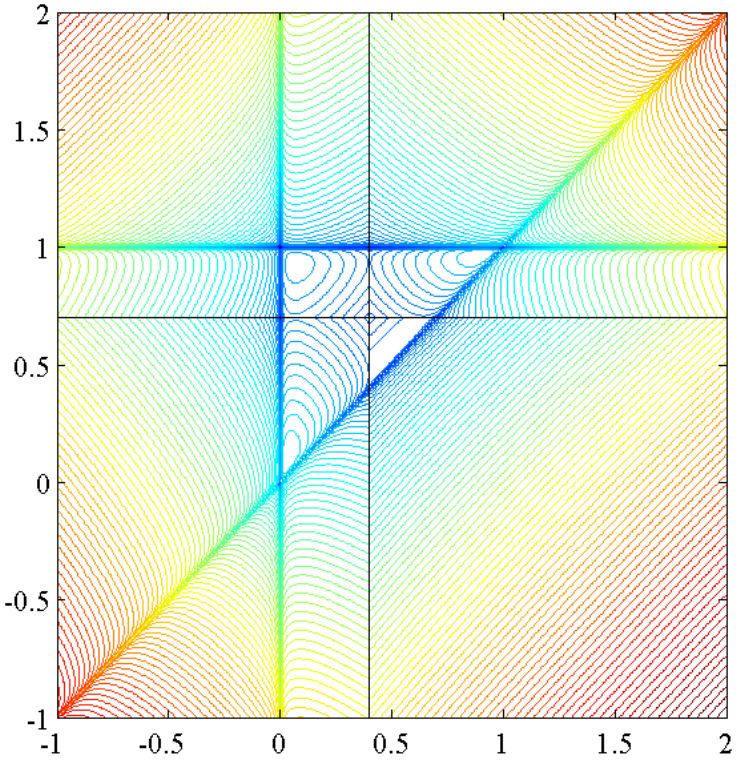


Figure 5.13: Level lines for the discretized L^1pTV, $G(u_1, u_2)$, with $\lambda = 1$ and $p = .1$

in Y, there are local minimizers and saddle points (we marked the $u_2 = f_2, u + 3 = f_3$ with black lines to make them more visible). We can also see that as $p \nearrow 1$, the plot looks more and more like Figure 5.1.

Also for Algorithm 5.4, we don't get a minimizers at the end of each iteration. This happens

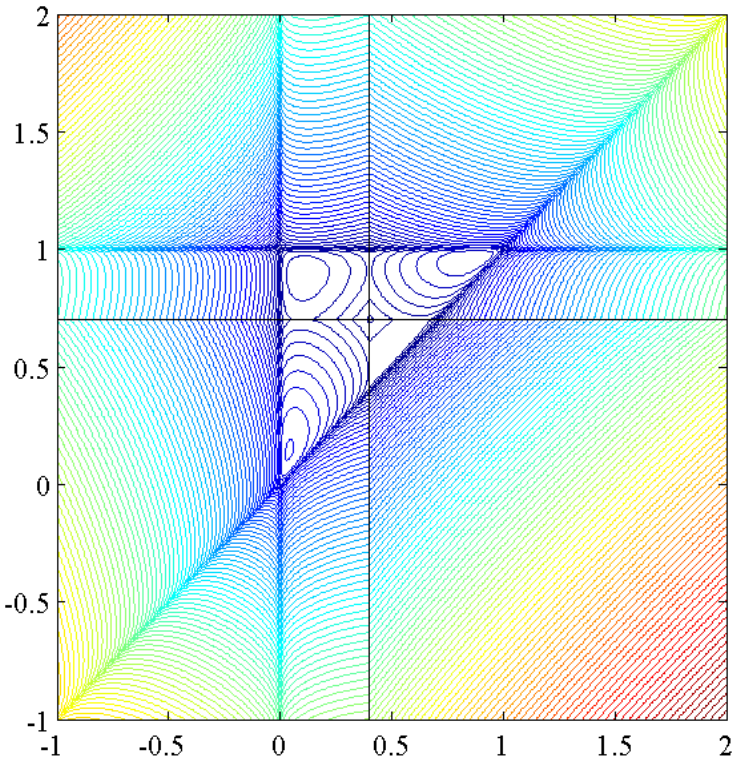


Figure 5.14: Level lines for the discretized L^1pTV, $G(u_1, u_2)$, with $\lambda = 1$ and $p = .5$

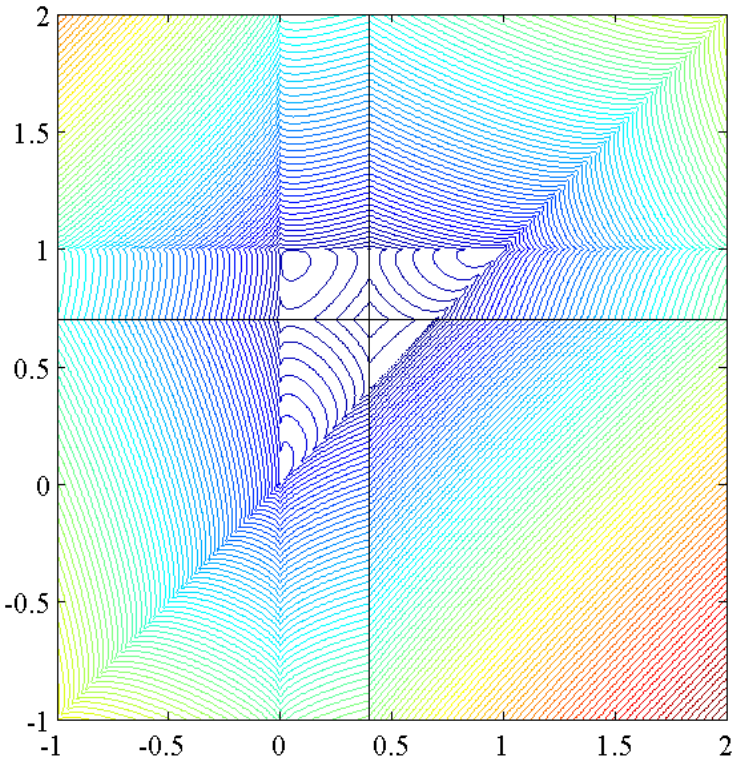


Figure 5.15: Level lines for the discretized $L^1 pTV$, $G(u_1, u_2)$, with $\lambda = 1$ and $p = .7$

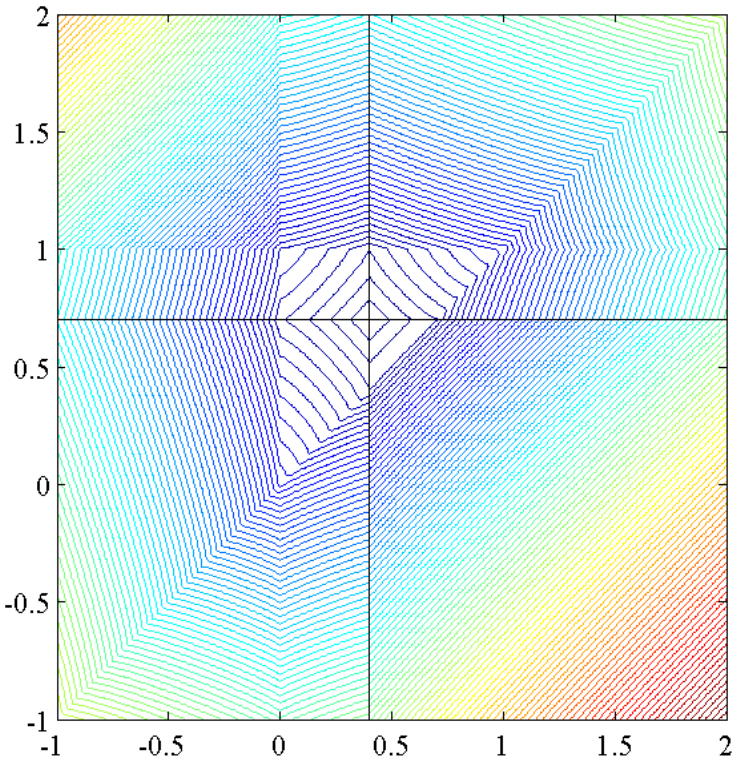


Figure 5.16: Level lines for the discretized $L^1 pTV$, $G(u_1, u_2)$, with $\lambda = 1$ and $p = .9$

because the effective cluster size can increase or decrease. The iterations that end with minimizers are those for which λ decreases in the next iteration. Notice if λ increases in the next iteration, we are still finding descent for the current λ_{cut} value. For example, if $\lambda_{cut}^{(k)} = 10$ we know that at iteration k, we find descent whenever $\lambda < 10$. If the maximum λ that will give descent at iteration $k+1$ is larger than 10, then we know that we will find descent for $\lambda < 10$ since $\lambda_{cut}^{(k+1)} > 10$.

Figures 5.17, 5.18, 5.19, and 5.20 show $(\lambda, \min G)$ from Algorithm 5.4 for $p = .1, .4, .7, .9$, respectively on a simple signal with two flat bumps (See Example 6.2. For the case of Algorithm 5.4, since G is linear in λ, we draw these lines for each of the λ_{cut} and the corresponding minimizing solution. The points plotted are at those λ for which the lines corresponding to consecutive iterations intersect. We also notice that as $p \to 1$, the plot looks more and more like the plots in Figure 5.11.

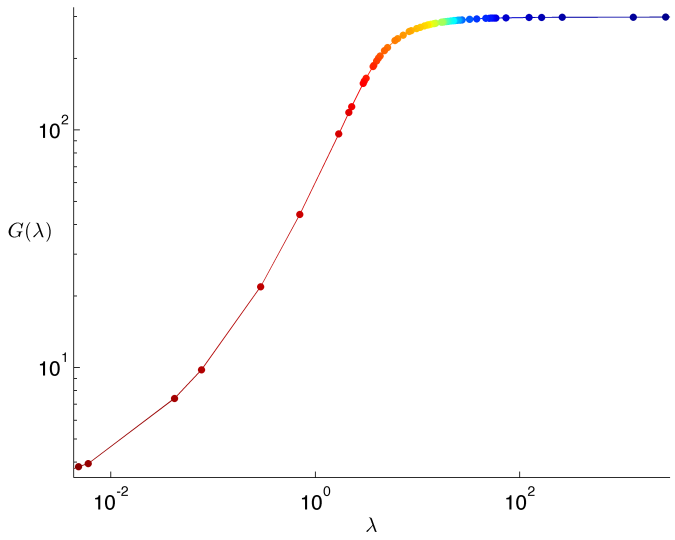


Figure 5.17: $\min G$ vs λ given by Algorithm 5.4, for $p = .1$

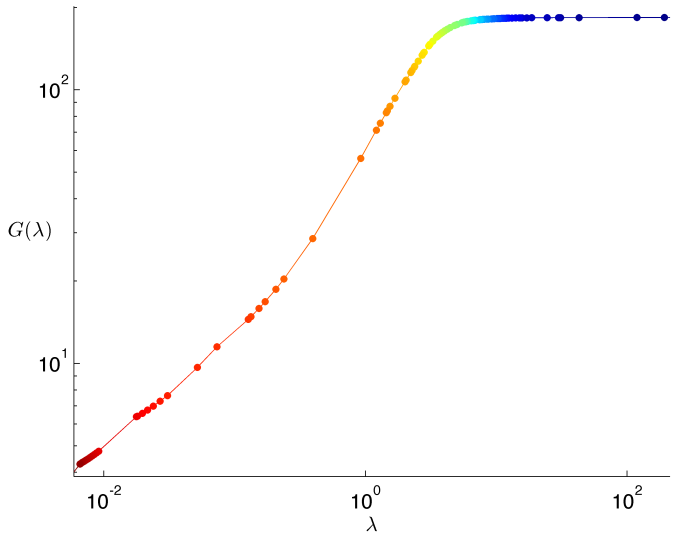


Figure 5.18: $\min G$ vs λ given by Algorithm 5.4, for $p = .4$

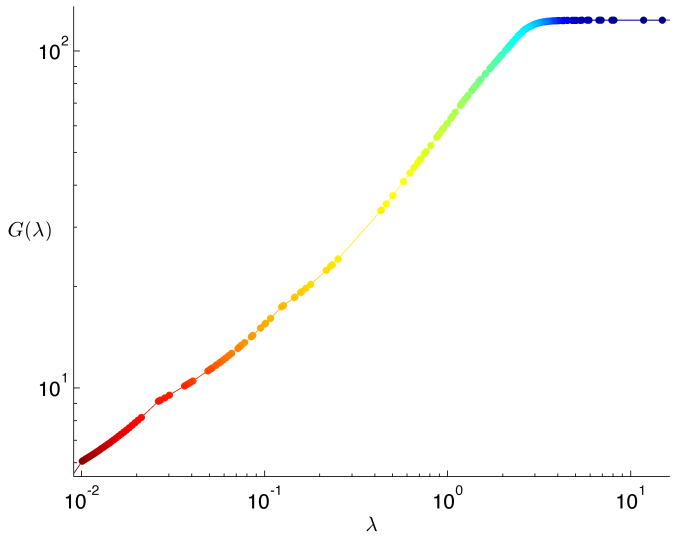


Figure 5.19: $\min G$ vs λ given by Algorithm 5.4, for $p = .7$

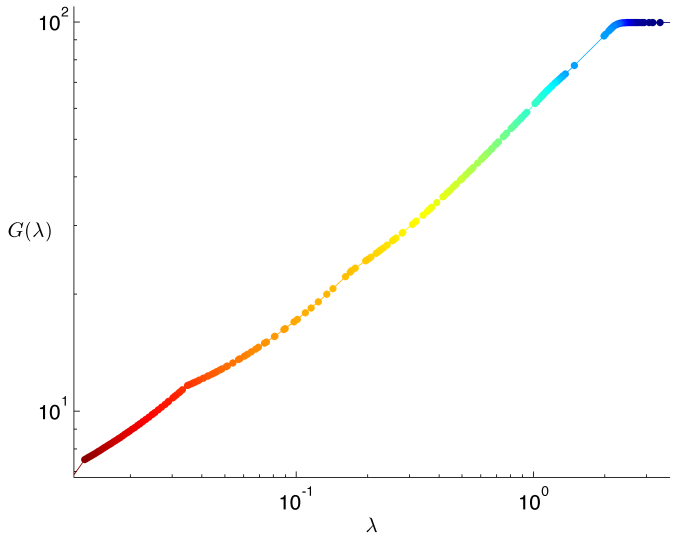


Figure 5.20: $\min G$ vs λ given by Algorithm 5.4, for $p = .9$

5.4 p-Variation, the $\lambda = 0$ case

In this section, we formulate the $\beta = 0$ version of (2.1),

$$\min \int_\Omega |u'|^p \, dx, \quad \text{for } 0 < p < 1, \tag{5.75}$$

as a discrete problem. We do this by setting $\lambda = 0$ in (5.2) to get

$$\min_{u \in \mathcal{F}, x} \left\{ G = \sum_{i=0}^{m} |u_{i+1} - u_i|^p \, (x_{i+1} - x_i)^{1-p} : x_i \le x_{i+1} \right\}, \tag{5.76}$$

where $\mathcal{F}$ is the set of piecewise linear continuous functions. We now introduce another hyperplane traversal algorithm. Here we have a partition $x_1, \ldots, x_m$ of Ω with $x_{i+1} > x_i$ and $x_0 = a, x_{m+1} = b, u_0 = \alpha, u_{m+1} = \beta$ are fixed boundary values. Notice that G is nonsmooth (or not differentiable) at any point (x, u) where at least one of $u_i = u_{i+1}$ or $u_i = u_{i-1}$ holds. Where G is smooth, we compute

the gradient of G at $(x, u) = (x_1, \ldots, x_m, u_1, \ldots, u_m)$ to be

$$\nabla G(x,u) = \begin{bmatrix} (1-p)\left(\frac{|u_1-u_0|^p}{(x_1-x_0)^p} - \frac{|u_2-u_1|^p}{(x_2-x_1)^p}\right) \\ \vdots \\ (1-p)\left(\frac{|u_m-u_{m-1}|^p}{(x_m-x_{m-1})^p} - \frac{|u_{m+1}-u_m|^p}{(x_{m+1}-x_m)^p}\right) \\ p\left(\frac{(x_1-x_0)^{1-p}}{|u_1-u_0|^{1-p}}\operatorname{sgn}(u_1-u_0) - \frac{(x_2-x_1)^{1-p}}{|u_2-u_1|^{1-p}}\operatorname{sgn}(u_2-u_1)\right) \\ \vdots \\ p\left(\frac{(x_m-x_{m-1})^{1-p}}{|u_m-u_{m-1}|^{1-p}}\operatorname{sgn}(u_m-u_{m-1}) - \frac{(x_{m+1}-x_m)^{1-p}}{|u_{m+1}-u_m|^{1-p}}\operatorname{sgn}(u_{m+1}-u_m)\right) \end{bmatrix}. \tag{5.77}$$

Now we give some lemmas about the function G. First, we show that G has no local minima. That is, since G is nonconvex, if G has no local minima, all minima of G must be global minima

Lemma 5.15. *If $(x_*, u_*) = (x_1, \ldots, x_m, u_1, \ldots, u_m)$ is not a global minimizer for G defined in Equation (5.76), then there exists a descent direction for G at (x_*, u_*).*

Proof. First, we refer to Equation (5.77). We have 3 possible cases.

1. $\nabla G \neq 0$ exists at (x_*, u_*):

   Then (x_*, u_*) is not a minimum (local or global) and thus $-\nabla G(x_*, u_*)$ is a descent direction.

2. $\nabla G = 0$:

   Then

$$\frac{|u_1-u_0|^p}{(x_1-x_0)^p} = \ldots = \frac{|u_{m+1}-u_m|^p}{(x_{m+1}-x_m)^p}.$$

$$\Rightarrow \frac{|u_1-u_0|}{x_1-x_0} = \ldots = \frac{|u_{m+1}-u_m|}{x_{m+1}-x_m}. \tag{5.78}$$

   And

$$\frac{(x_1-x_0)^{1-p}}{|u_1-u_0|^{1-p}}\operatorname{sgn}(u_1-u_0) = \ldots = \frac{(x_{m+1}-x_m)^{1-p}}{|u_{m+1}-u_m|^{1-p}}\operatorname{sgn}(u_{m+1}-u_m). \tag{5.79}$$

Using (5.78) and (5.79), we get

$$\operatorname{sgn}(u_1 - u_0) = \ldots = \operatorname{sgn}(u_{m+1} - u_m).$$

And, so

$$\Rightarrow \frac{u_1 - u_0}{x_1 - x_0} = \ldots = \frac{u_{m+1} - u_m}{x_{m+1} - x_m}.$$

Which implies that the function defined by $u(x_*) = u_*$ is the line, $u(x) = \frac{u_{m+1}-u_0}{x_{m+1}-x_0}(x - x_0) + u_0$.

A descent direction at (x_*, u_*) is given by

$$v = (0, \ldots, 0, 1, 0, \ldots, 0) \in \mathbb{R}^{2m},$$

where the 1 is in the $(m + k)^{\text{th}}$ position $(0 < k \leq m)$. That is, if $\epsilon > 0$ then we need to show that v is a descent direct, that is,

$$G((x_*, u_*) + \epsilon v) < G(x_*, u_*), \tag{5.80}$$

for $\epsilon > 0$ small enough.

First, let C be the slope of the line through the points $(x_i, u_i), 0 \leq i \leq m + 1$. That is,

$$\frac{u_{i+1} - u_i}{x_{i+1} - x_i} = C, \quad \forall\ 0 \leq i \leq m + 1.$$

Assume also, without loss of generality that $C > 0$. Then $u_{i+1} - u_i > 0$ for each i and we have

$$G(x_*, u_*) = \sum_{i=0}^{m} (u_{i+1} - u_i)^p (x_{i+1} - x_i)^{1-p} = \sum_{i=0}^{m} \left(\frac{u_{i+1} - u_i}{x_{i+1} - x_i} \right)^p (x_{i+1} - x_i) = C^p (x_{m+1} - x_0).$$

And,

$$G((x_*, u_*) + \epsilon v) = \sum_{i=0}^{k-2} (u_{i+1} - u_i)^p (x_{i+1} - x_i)^{1-p} + \left(\frac{u_k + \epsilon - u_{k-1}}{x_k - x_{k-1}} \right)^p (x_k - x_{k-1}) + \left(\frac{u_{k+1} - \epsilon - u_k}{x_{k+1} - x_k} \right)^p (x_{k+1} - x_k) + \sum_{k+1}^{m} (u_{i+1} - u_i)^p (x_{i+1} - x_i)^{1-p}$$

$$= C^p (x_{k-1} - x_0) + C^p (x_{m+1} - x_{K+1}) + \left(\frac{u_k + \epsilon - u_{k-1}}{x_k - x_{k-1}} \right)^p (x_k - x_{k-1}) + \left(\frac{u_{k+1} - \epsilon - u_k}{x_{k+1} - x_k} \right)^p (x_{k+1} - x_k)$$

$$= G(x_*, u_*) + C^p (x_k - x_{k-1}) \left(\left(\frac{u_k + \epsilon - u_{k-1}}{u_k - u_{k-1}} \right)^p - 1 \right) + C^p (x_{k+1} - x_k) \left(\left(\frac{u_{k+1} - \epsilon - u_k}{u_{k+1} - u_k} \right)^p - 1 \right). \quad (5.81)$$

Using a Taylor expansion, centered about $\epsilon = 0$, we can write this as

$$G((x_*, u_*) + \epsilon v) = G(x_*, u_*) + C^p \left(\frac{p}{C} \epsilon + \frac{p(p-1)}{2C(u_k - u_{k-1})} \epsilon^2 + O(\epsilon^3) \right) + C^p \left(-\frac{p}{C} \epsilon + \frac{p(p-1)}{2C(u_{k+1} - u_k)} \epsilon^2 + O(\epsilon^3) \right)$$

$$= G(x_*, u_*) + C^p \left(\frac{p(p-1)}{2C} \left(\frac{1}{(u_k - u_{k-1})} + \frac{1}{(u_{k+1} - u_k)} \right) \epsilon^2 + O(\epsilon^3) \right)$$

Now, $0 < p < 1$ gives us a negative coefficient on ϵ^2. Thus, we can find an ϵ small enough so that (5.80) holds.

3. $\nabla G(x_*, u_*)$ does not exist:

   Let's define the indexing sets

$$\mathcal{A} = [2, m] \cap \mathbb{Z}, \quad \mathcal{I} = \{i \in \mathcal{A} : x_i = x_{i-1}\} \quad \text{and} \quad \mathcal{J} = \{j \in \mathcal{A} : u_j = u_{j-1}\}.$$

We can see from ∇G in equation (5.77), that if

$$\mathcal{I} \neq \emptyset, \quad \mathcal{J} \neq \emptyset, \quad x_1 = x_0, \quad x_m = x_{m+1}, \quad u_1 = u_0, \quad u_m = u_{m+1},$$

or some combination of these, then ∇G does not exist. For now, let us consider only the case where $\mathcal{I} \neq \emptyset$ and $\mathcal{J} \neq \emptyset$. Say

$$|\mathcal{I}| = l \quad \text{and} \quad |\mathcal{J}| = k,$$

where $|\cdot|$ denotes cardinality.

Let

$$\mathcal{H} = \bigcup_{i \in \mathcal{I}} \{x_i = x_{i-1}\} \quad \cup \quad \bigcup_{j \in \mathcal{J}} \{u_j = u_{u-1}\}$$

be the union of hyperplanes where ∇G does not exist. We find descent of G in $\mathcal{H}$. To do this, we consider the function, $\tilde{G}$,

$$\tilde{G} : \mathbb{R}^{2m-l-k} \to \mathbb{R} \text{ defined by } \tilde{G}(\tilde{x}, \tilde{u}) = G(\hat{x}, \hat{u}),$$

where $\tilde{x} \in \mathbb{R}^{m-l}$ is the point formed from the components of x_* whose index is not in $\mathcal{I}$, $\tilde{u} \in \mathbb{R}^{m-k}$ is the point formed from the components of u_* whose index is not in $\mathcal{J}$, $\hat{x} \in \mathbb{R}^m$ and $\hat{u} \in \mathbb{R}^m$ are points in the union, $\mathcal{H}$, of the hyperplanes that correspond to $\tilde{x}$ and $\tilde{u}$ respectively. The gradient, $\nabla \tilde{G}$, exists and gives a descent direction in $\mathbb{R}^{2m-l-k}$. Since $G|_{\mathcal{H}}$ and $\tilde{G}$ are equivalent, we can construct the vector, v, in $\mathcal{H}$ corresponding to ∇G. We form v from ∇G but changing the components corresponding to coordinates whose indices are in $\mathcal{I}$ or $\mathcal{J}$ so that v points in $\mathcal{H}$. That is, if $i \in \mathcal{I}$, then $x_i = x_{i-1}$ and we find that $\frac{\partial G}{\partial x_i}$ and $\frac{\partial G}{\partial x_{i-1}}$ do not exist. So, we replace both of these components in ∇G with $\frac{\partial \tilde{G}}{\partial x_{i-1}}$. Similarly, if $j \in \mathcal{J}$, we replace the components, $\frac{\partial G}{\partial u_j}$ and $\frac{\partial G}{\partial u_{j-1}}$ in ∇G with $\frac{\partial \tilde{G}}{\partial u_{j-1}}$.

Now, to deal with the cases when $x_1 = x_0$, $x_m = x_{m+1}$, $u_1 = u_0$, $u_m = u_{m+1}$, we replace the

components of ∇G,

$$\frac{\partial G}{\partial x_1}, \quad \frac{\partial G}{\partial x_m}, \quad \frac{\partial G}{\partial u_1}, \quad \frac{\partial G}{\partial u_m},$$

with 0, respectively. We then see that $-v$ is a descent direction for G which points in $\mathcal{H}$.

□

For clarity in part (c) of the above argument, we consider the example:

Example 5.2. Let $m = 5, \mathcal{I} = \{4\}, \mathcal{J} = \{2\}, u_5 = u_6$. Recall, x_0, u_0, x_6, u_6 are all fixed. So we write $\tilde{G}$ as

$$\tilde{G}(x_1, x_2, x_3, x_5, u_1, u_3, u_4) = G(x_1, x_2, x_3, x_3, x_5, u_1, u_1, u_3, u_4, u_6)$$
$$= |u_1 - u_0|^p (x_1 - x_0)^{1-p} + |u_3 - u_2|^p (x_3 - x_2)^{1-p} + |u_6 - u_4|^p (x_5 - x_3)^{1-p}.$$

We construct v:

$$v = \begin{bmatrix} \frac{\partial G}{\partial x_1}(x_*, u_*) \\ \frac{\partial G}{\partial x_2}(x_*, u_*) \\ \frac{\partial \tilde{G}}{\partial x_3}(x_1, x_2, x_3, x_5, u_1, u_3, u_4) \\ \frac{\partial \tilde{G}}{\partial x_3}(x_1, x_2, x_3, x_5, u_1, u_3, u_4) \\ \frac{\partial G}{\partial x_5}(x_*, u_*) \\ \frac{\partial \tilde{G}}{\partial u_1}(x_1, x_2, x_3, x_5, u_1, u_3, u_4) \\ \frac{\partial \tilde{G}}{\partial u_1}(x_1, x_2, x_3, x_5, u_1, u_3, u_4) \\ \frac{\partial G}{\partial u_3}(x_*, u_*) \\ \frac{\partial G}{\partial u_4}(x_*, u_*) \\ 0 \end{bmatrix}$$

Now we characterize the set of stationary points of the function G. It turns out that the set of stationary points are points (x, u) so that $u(x)$ is affine.

Lemma 5.16. *Define*

$$\mathcal{S} = \{(x, u) | \nabla G(x, u) = 0\},$$

with G, defined as in Equation (5.76), and fix $(x, u) = (x_1, \ldots, x_m, u_1, \ldots, u_m)$*, with* $x_{i+1} > x_i$ *then* $(x, u) \in \mathcal{S}$ *if and only if* $\exists$ *a continuous affine function* $u : \mathbb{R} \to \mathbb{R}$ *so that* $u(x_i) = u_i$ *and* $u(a) = \alpha, u(b) = \beta$ *for real numbers* $\alpha \neq \beta$*.*

Proof. Let G be defined as in Equation (5.76).

We start by assuming the existence of the linear function u so that $u(x_i) = u_i$ and we will show that $(x, u) = (x_1, \ldots, x_m, u_1, \ldots, u_m) \in \mathcal{S}$. We know that, since u is linear,

$$\frac{u(x) - u(y)}{x - y} = C,$$

Since $C \neq 0$, $x_{i+1} > x_i$ tells us that $\operatorname{sgn}(u_{i+1} - u_i)$ is either always 1 or always -1. Without loss of generality, let $\operatorname{sgn}(u_{i+1} - u_i) = 1, \ \forall i$. As in the definition of f, we use the notation $x_0 = a, x_{m+1} = b, u_0 = \alpha, u_{m+1} = \beta$. Thus

$$\left| \frac{u_i - u_{i-1}}{x_i - x_{i-1}} \right| = \left| \frac{u(x_i) - u(x_{i-1})}{x_i - x_{i-1}} \right| = |C|,$$

and

$$(1 - p) \left[\left| \frac{u_i - u_{i-1}}{x_i - x_{i-1}} \right|^p - \left| \frac{u_{i+1} - u_i}{x_{i+1} - x_i} \right|^p \right] = (1 - p) \left(|C|^p - |C|^p \right) = 0,$$

for $i = 1 \ldots m$. Also,

$$p \left[\left| \frac{x_i - x_{i-1}}{u_i - u_{i-1}} \right|^{1-p} \operatorname{sgn}(u_i - u_{i-1}) - \left| \frac{x_{i+1} - x_i}{u_{i+1} - u_i} \right|^{1-p} \operatorname{sgn}(u_{i+1} - u_i) \right] = p \left(\frac{1}{|C|^{1-p}} - \frac{1}{|C|^{1-p}} \right) = 0,$$

for $i = 1 \ldots m$. Thus, by Equation (5.77), $\nabla G(x, u) = 0$ and so $(x, u) \in \mathcal{S}$.

Now, we assume $(x, u) \in \mathcal{S}$. Then $\nabla f(x, u) = 0$. That is,

$$\left| \frac{u_i - u_{i-1}}{x_i - x_{i-1}} \right|^p - \left| \frac{u_{i+1} - u_i}{x_{i+1} - x_i} \right|^p = 0 \text{ and } \left| \frac{x_i - x_{i-1}}{u_i - u_{i-1}} \right|^{1-p} \operatorname{sgn}(u_i - u_{i-1}) - \left| \frac{x_{i+1} - x_i}{u_{i+1} - u_i} \right|^{1-p} \operatorname{sgn}(u_{i+1} - u_i) = 0.$$

First, notice that the terms in the equations above must be nonzero because

$$\text{if } \left|\frac{u_i - u_{i-1}}{x_i - x_{i-1}}\right|^p = 0, \quad \Longrightarrow \quad \left|\frac{x_i - x_{i-1}}{u_i - u_{i-1}}\right|^{1-p} \operatorname{sgn}(u_i - u_{i-1}) \text{ is undefined.}$$

Notice, also, that in order for equality to hold, $\operatorname{sgn}(u_{i+1} - u_i)$ must always be 1 or always -1 for otherwise, we could not add two negative or two positive real numbers and get 0. So, let us again, without losing generality, choose $\operatorname{sgn}(u_{i+1} - u_i) = 1, \ \forall i$. So

$$\frac{|u_i - u_{i-1}|}{x_i - x_{i-1}} = \frac{|u_{i+1} - u_i|}{x_{i+1} - x_i}$$

That is,

$$\frac{u_i - u_{i-1}}{x_i - x_{i-1}} = \frac{u_{i+1} - u_i}{x_{i+1} - x_i} \ \forall i = 1 \ldots m.$$

To complete the proof, we define $u : \mathbb{R} \to \mathbb{R}$ to be the linear function, $u(x) = C(x - a) + \alpha$, where C is defined by

$$C \equiv \frac{u_i - u_{i-1}}{x_i - x_{i-1}}.$$

□

Note: There are special cases, not covered in the above statement, where u is a line, but ∇G does not exist. These cases are when $\alpha = \beta$ or when $x_{i+1} = x_i$ for some i.

Notice that Lemma 5.16 gives us stationary points for (5.76) when $x_i < x_{i+1}$, but if we consider allowing $x_i = x_{i+1}$, we can see that $G = 0$ whenever (x, u) is a step function. This is easily seen because for a step function, in any interval, either $x_i = x_{i+1}$ or $u_i = u_{i+1}$. So, as we saw in Chapter 2, step functions are minimizers of G.

With this understanding of G and, again, recognizing that G is nonsmooth on the hyperplanes where $u_i = u_{i+1}$, $u_i = u_{i-1}$, $x_i = x_{i-1}$, or $x_i = x_{i+1}$ for some $1 \leq i \leq m - 1$, we consider an algorithm to find minimizers of G.

5.4.1 ht Algorithm for p-Variation

In Figures 5.22 and 5.24, we plot some level curves for simple examples of G with $m = 2$. In these figures, dark blue represents the lowest values of G and crimson represents the highest values. We see in these figures that G attains its minimizers along the hyperplanes where G is nonsmooth. This suggests that a good algorithm for finding the minimizer of (5.76) is one that steps to a hyperplane and in the hyperplane steps to another hyperplane always remaining on hyperplanes to which we have stepped.

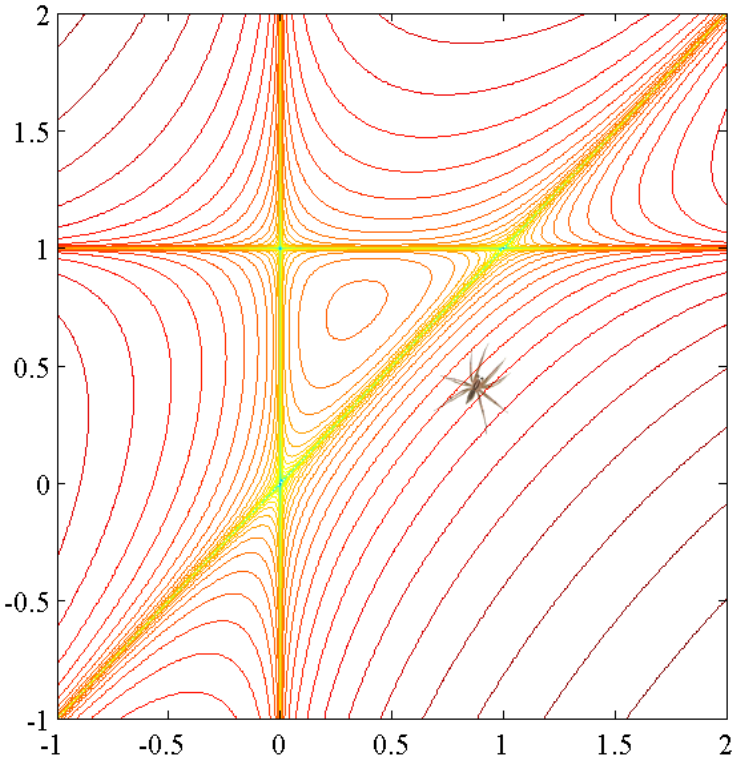


Figure 5.21: Level lines for the discretized p-variation, $G(u_1, u_2) = |u_1|^{.5} + |u_2 - u_1|^{.5} + |1 - u_2|^{.5}$.

In words Algorithm 5.6 starts with a point where ∇G exists and is nonzero. Otherwise, we take a small step in a direction to get to a point where ∇G exists and is nonzero. From this point, we step to the closest point where ∇G is undefined, in the direction of $-\nabla G$. The points to which we step are those where $u_i = u_{i+1}$, $u_i = u_{i-1}$, $x_i = x_{i+1}$, or $x_i = x_{i+1}$ for some $1 \leq i \leq i - 1$. Using

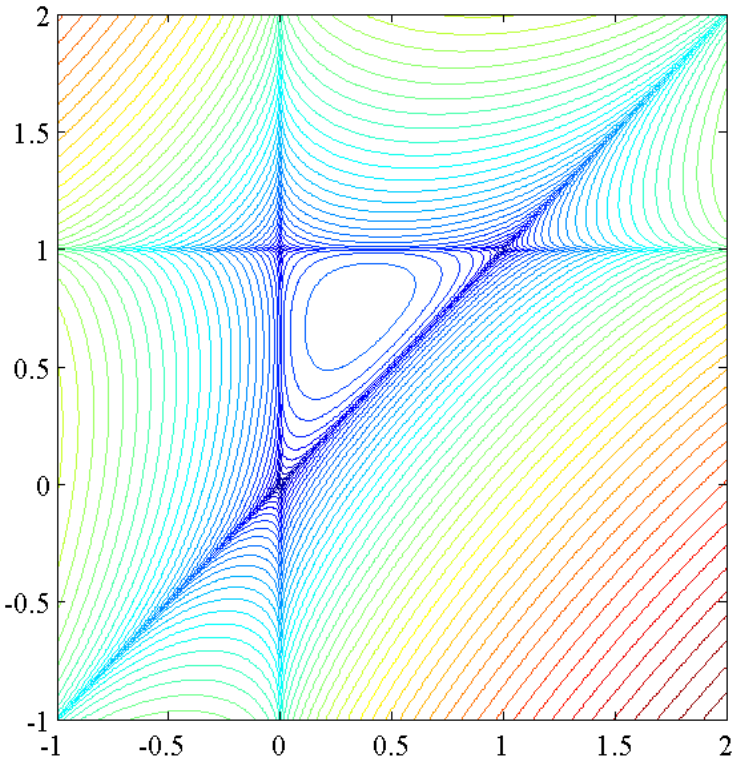


Figure 5.22: Level lines for the discretized p-variation, $G(u_1, u_2) = |u_1|^{.4} + |u_2 - u_1|^{.4} + |1 - u_2|^{.4}$

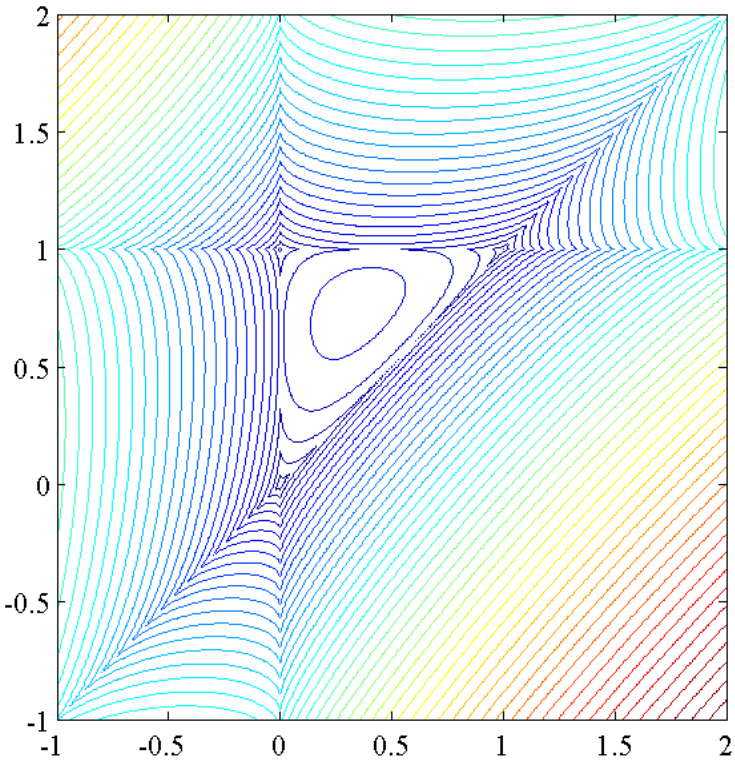


Figure 5.23: Level lines for the discretized p-variation, $G(u_1, u_2) = |u_1|^{.7} + |u_2 - u_1|^{.7} + |1 - u_2|^{.7}$

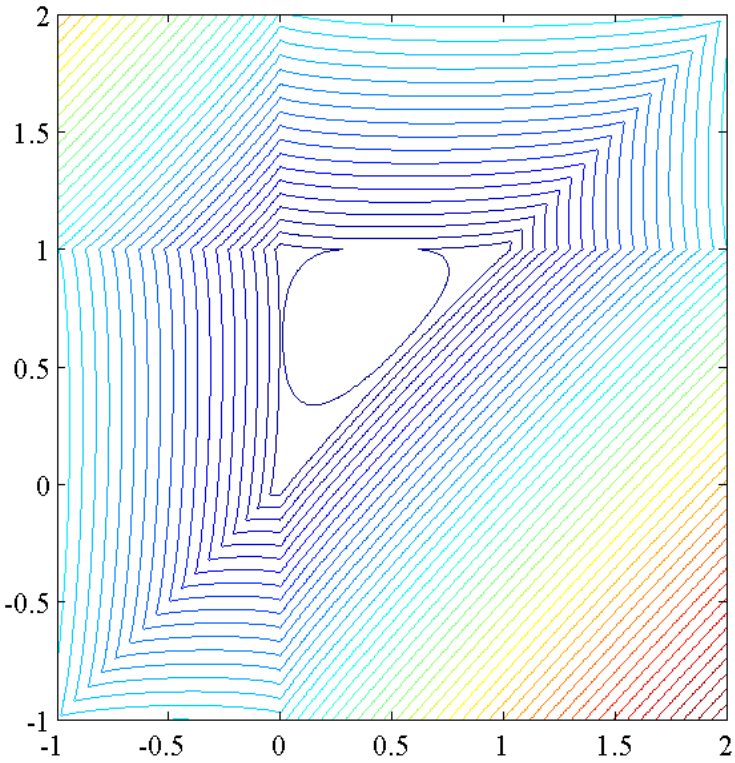


Figure 5.24: Level lines for the discretized p-variation, $G(u_1, u_2) = |u_1|^{.9} + |u_2 - u_1|^{.9} + |1 - u_2|^{.9}$

Algorithm 5.6. *(p-Variation)*
Choose $(x^{(0)}, u^{(0)}) = (x_1^{(0)}, \ldots, x_m^{(0)}, u_1^{(0)}, \ldots, u_m^{(0)}) \in \mathbb{R}^{2m}$ *so that* ∇G *exists and is nonzero.;*
Set $n_0 \leftarrow 2m$*;*
Set $G^{(0)} \leftarrow G$*;*
Set $d^{(0)} \leftarrow \nabla G^{(0)}(x^{(0)}, u^{(0)})$*;*
Set $k \leftarrow 0$*;*
do
$\alpha_k \leftarrow \min\{\alpha : \mathcal{I}((x^{(k)}, u^{(k)}) + \alpha d^{(k)}) \cup \mathcal{J}((x^{(k)}, u^{(k)}) + \alpha d^{(k)}) \neq \emptyset\}$*;*
$(x^{(k+1)}, u^{(k+1)}) \leftarrow (x^{(k)}, u^{(k)}) + \alpha_k d^{(k)}$*;*
$k \leftarrow k + 1$*;*
$n_k \leftarrow n_{k-1} - |\mathcal{I}(x^{(k)}, u^{(k)}) \cup \mathcal{J}(x^{(k)}, u^{(k)})|$*;*
Reformulate $G^{(k-1)}$ *to get* $G^{(k)}$*;*
Set $d^{(k)} \leftarrow \nabla G^{(k)}$*;*
until $n_k = 0$

this information, we can reformulate the problem to a lower dimensional problem. With the new problem, we start again. It should be noted that there is a chance that in stepping to a point where G is nonsmooth, we may also reach a saddle point so that in the reformulated problem, $\nabla G = 0$. In this case, we do as above and move in any direction along the hyperplane in which we stepped ($u_i = u_{i+1}$ or $x_i = x_{i+1}$). In the reformulation step, we consider a new function $\tilde{G}$ whose domain is now the intersection of the hyperplanes to which we have already stepped, but whose function values are the same as those of G.

In this algorithm, we start away where G is smooth, so that the gradient of G exits. But, we still step to the hyperplanes where G is nonsmooth, looking for solutions where in Y. Since we stay in Y and we find descent at each iteration, we find $\min_{u \in Y} G$. Recall that $|Y| < \infty$ so this algorithm is also finite.

5.5 Algorithms Summary

In this chapter we reformulated the functional

$$\min \int_\Omega |\nabla u|^p + \lambda |f - u| \, dx \tag{5.82}$$

into a discrete function. We recognize the structure of this function, more specifically that the function is nonsmooth on hyperplanes of the form $\{u_i = u_j\}$ and $\{u_i = f_i\}$, where $u = (u_1, \ldots, u_m)$ and $f = (f_1, \ldots, f_m)$. We then use this structure to formulate what we call hyperplane traversal algorithms to quickly, and in finitely many iterations, find minimizers for L^1TV and local minimizers for L^1pTV for all $\lambda > 0$. The most remarkable thing about our L^1TV algorithm is that it solves the minimization problem with the computational cost of only solving the $\lambda = 0$ problem since it finds all other λ solutions along the way. In this chapter, we proved that our L^1TV ht algorithms are finite and converge to a minimizer. We also discuss the convergence and finiteness of our L^1pTV ht algorithm to a local minimizer. Finally, we introduce an ht algorithm to solve the p-variation minimization problem.

We ran some time trials that seem to indicate that Algorithm 5.3 is of order $o(aN + m)$, where N is the signal length and M is the number of initial clusters. Our time tends to deviate from this estimate for large N, but we suspect this is due to memory usage with in Matlab.

CHAPTER SIX

DENOISING AND SCALE SIGNATURES FOR 1-DIMENSIONAL SIGNALS

In this chapter we show the results of denoising 1-dimensional signals using both Algorithm 5.3 for L^1TV and Algorithm 5.4 for L^1pTV. We compare results from our Hyperplane Traversal (ht) algorithms to the expected results discussed in [4, 24, 6] for denoising with L^1TV and in [8] for denoising using L^1pTV. Inspired by the results of [30] we use L^1TV signatures using Algorithm 5.3 to find scales in 1-dimensional signals.

6.1 Denoising

In the next two subsections, we will denoise synthetic signals using our ht algorithms to show that our algorithms produce expected results based on the analysis of [4, 6, 24, 8].

6.1.1 L^1TV Examples

We begin our denoising examples using our Algorithm 5.3 on several test signals.

Example 6.1. (Stationary Gaussian noise) First, we observe that Algorithm 5.3 removes stationary Gaussian noise for large λ (small scales). See Figure 6.1. We can see that the noise is removed, but there is a change in signal intensity.

Example 6.2. (Simple denoising) Here, we consider a simple example of a noisy signal (See Figure 6.2. Using L^1TV, we expect to lose small scales in the data first. In this example it looks

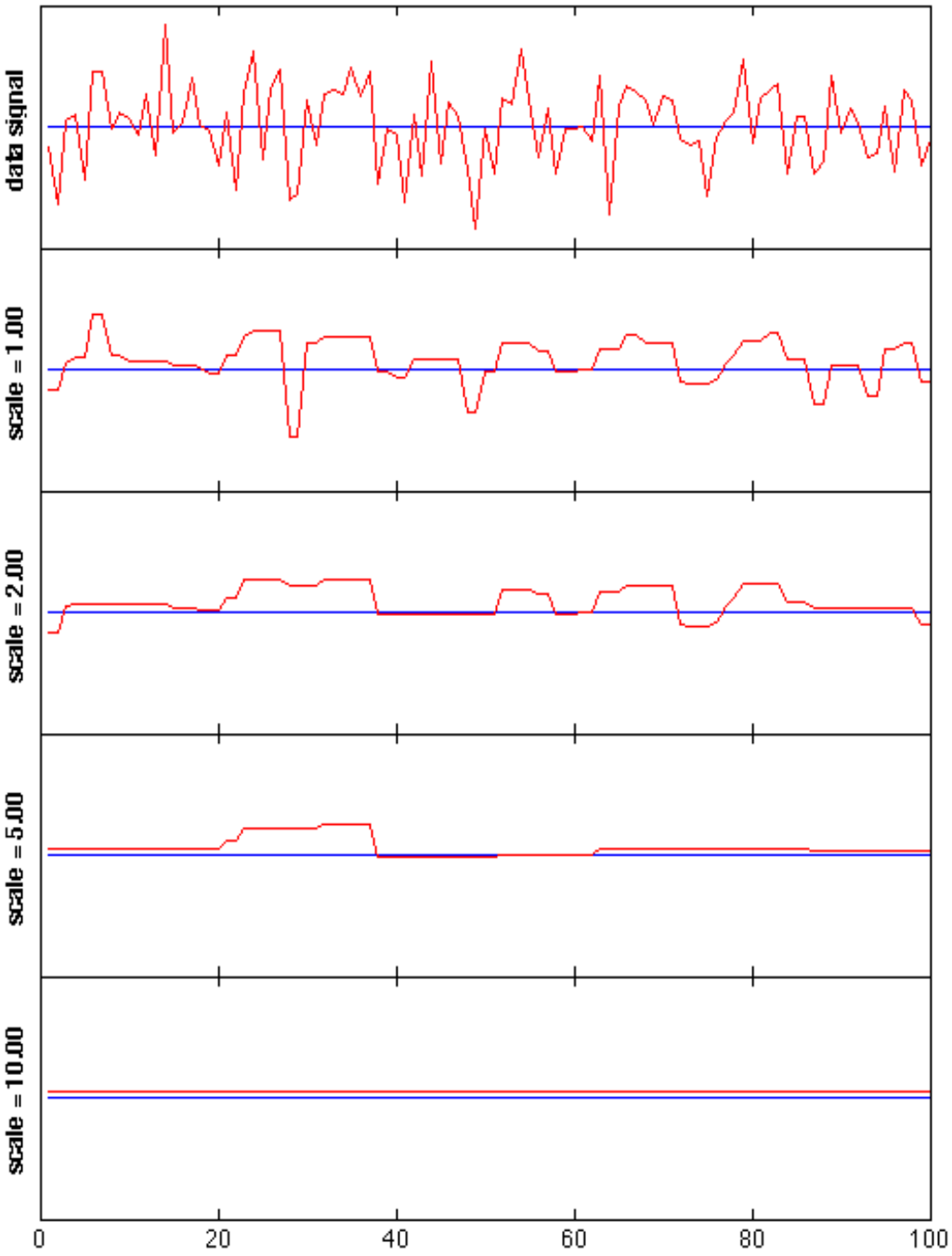


Figure 6.1: Denoising, using Alg 5.3 for L^1TV, a signal of stationary Gaussian noise. Blue: ground truth.

like the noise is of a scale close to 1 or 2 data points. We look at the L^1TV minimizers given by our algorithm for $\lambda_{cut} < \frac{2}{s}$. We choose λ to be the values $\frac{2}{3}$, $\frac{1}{4}$, and $\frac{1}{5}$. For $\lambda = \frac{2}{3}$, scales of size 3 are removed and we see that the noise is mostly gone, with some effects still present. For $\lambda = \frac{1}{4}$, we see that scales of size 8 are removed and almost all effects of the noise are gone. We do lose small scales if we set λ to be too small as can be seen when $\lambda = \frac{1}{5}$, we lose the small gap between the two bumps. This gap has scale 10. We also see that intensities at the jump discontinuities are not completely preserved as is expected with L^1TV [6, 30]. We also see that for scale 5, the result is rather close to recovering the original signal.

In Figure 6.3 we show the effects of denoising the same signal with more noise. We see similar results are obtained even in this higher noise case. We see that we remove most noise as $\lambda \searrow 0$ just before losing the small scale (size 10) feature. But, we see that the effects of noise in the flat regions are still more visible, than the lower noise example, just before we lose this gap between the two bumps. Also, at jump discontinuities there is even less preservation of signal intensity due. It is more clear with higher noise that there is some intensity loss as well.

In these two examples, using Algorithm 5.3, we were able to recover the original signal in the condition that we would expect using L^1TV. In comparing to the known results for L^1TV, we observe the same loss of small scale objects in the signal by decreasing λ.

Example 6.3. (Denoising a sinusoidal signal) In this example, we consider a sinusoidal signal with additive Gaussian noise (see Figure 6.4). We use Algorithm 5.3 to denoise this signal using a free boundary conditions (allowing the boundaries to move). As $\lambda \searrow 0$ (scale$\nearrow \infty$) we observe that the small scale (noise) is removed, but at the expense of losing signal intensity. We see also that as the boundaries move, we lose some of the periodicity of the signal.

Example 6.4. (Denoising a sinusoidal signal with fixed boundaries) As in Example 6.3, we consider a sinusoidal signal with additive Gaussian noise, but in this example we fix the boundaries. Notice (see Figure 6.5) that with fixed boundaries, we see better preservation of the periodicity even for larger scales. The effects of signal intensity is the same as in Example 6.3.

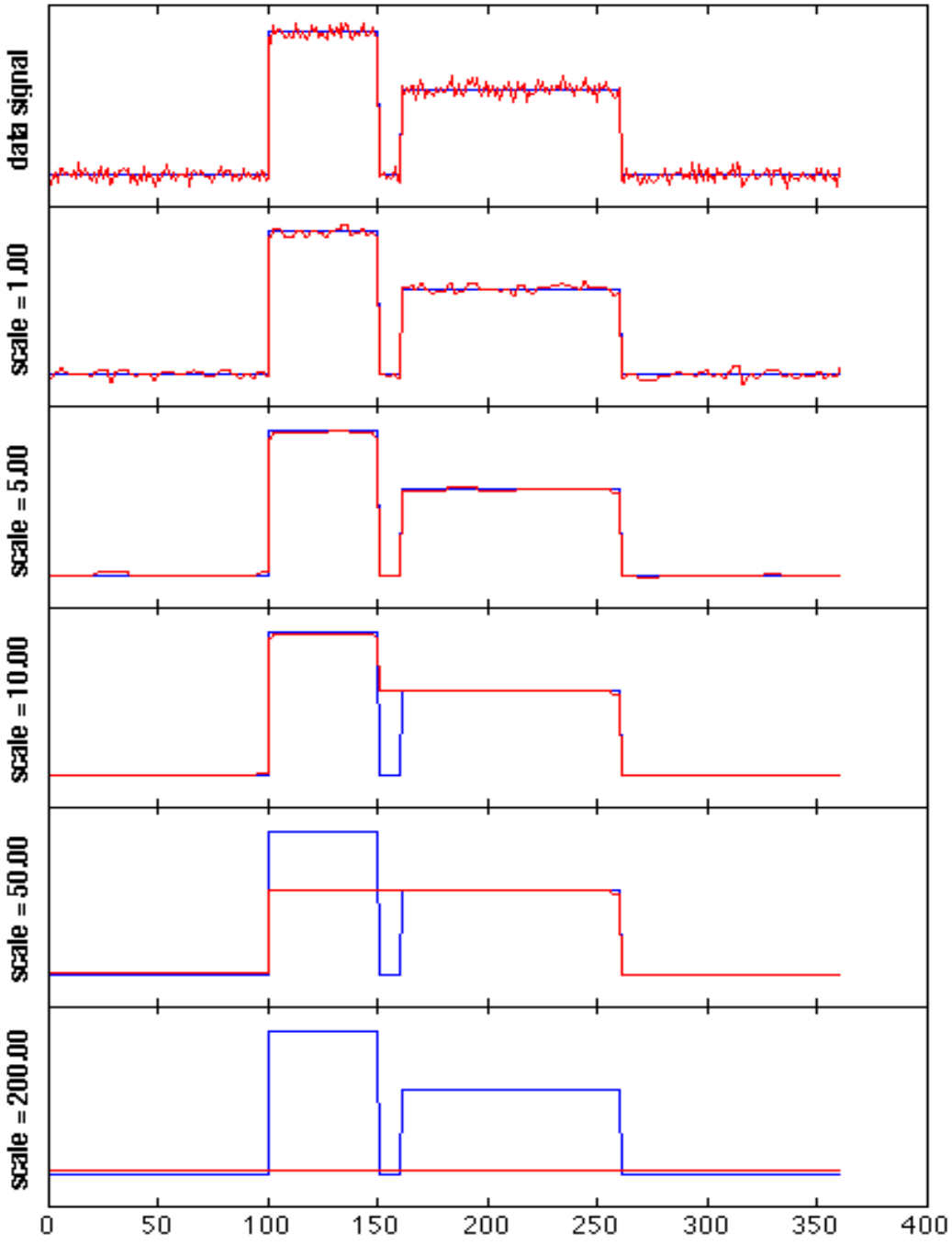


Figure 6.2: Denoising, using Alg 5.3 for L^1TV, on a simple noisy signal. Blue: ground truth.

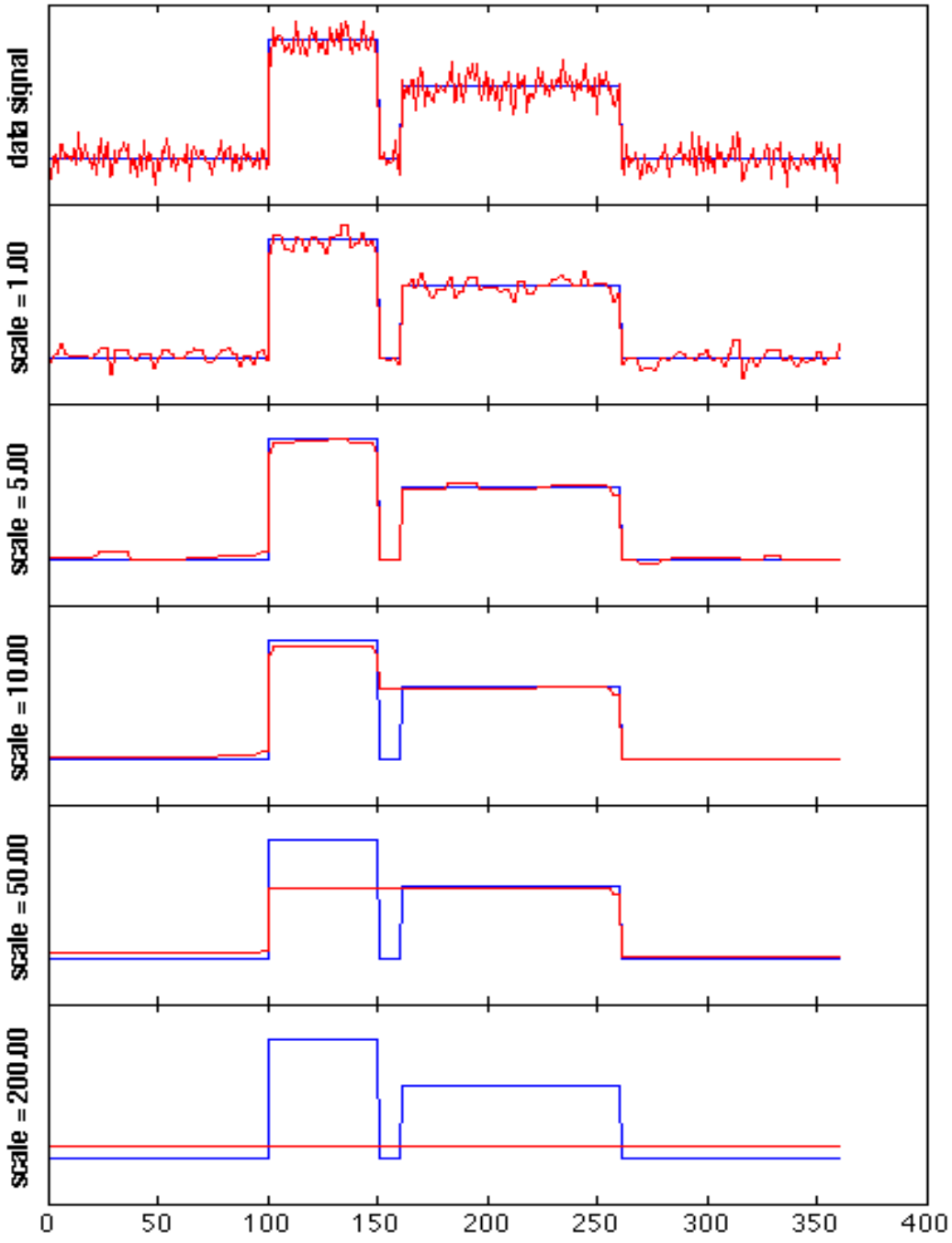


Figure 6.3: Denoising, using Alg 5.3 for L^1TV, on a simple noisy signal with higher noise. Blue: ground truth.

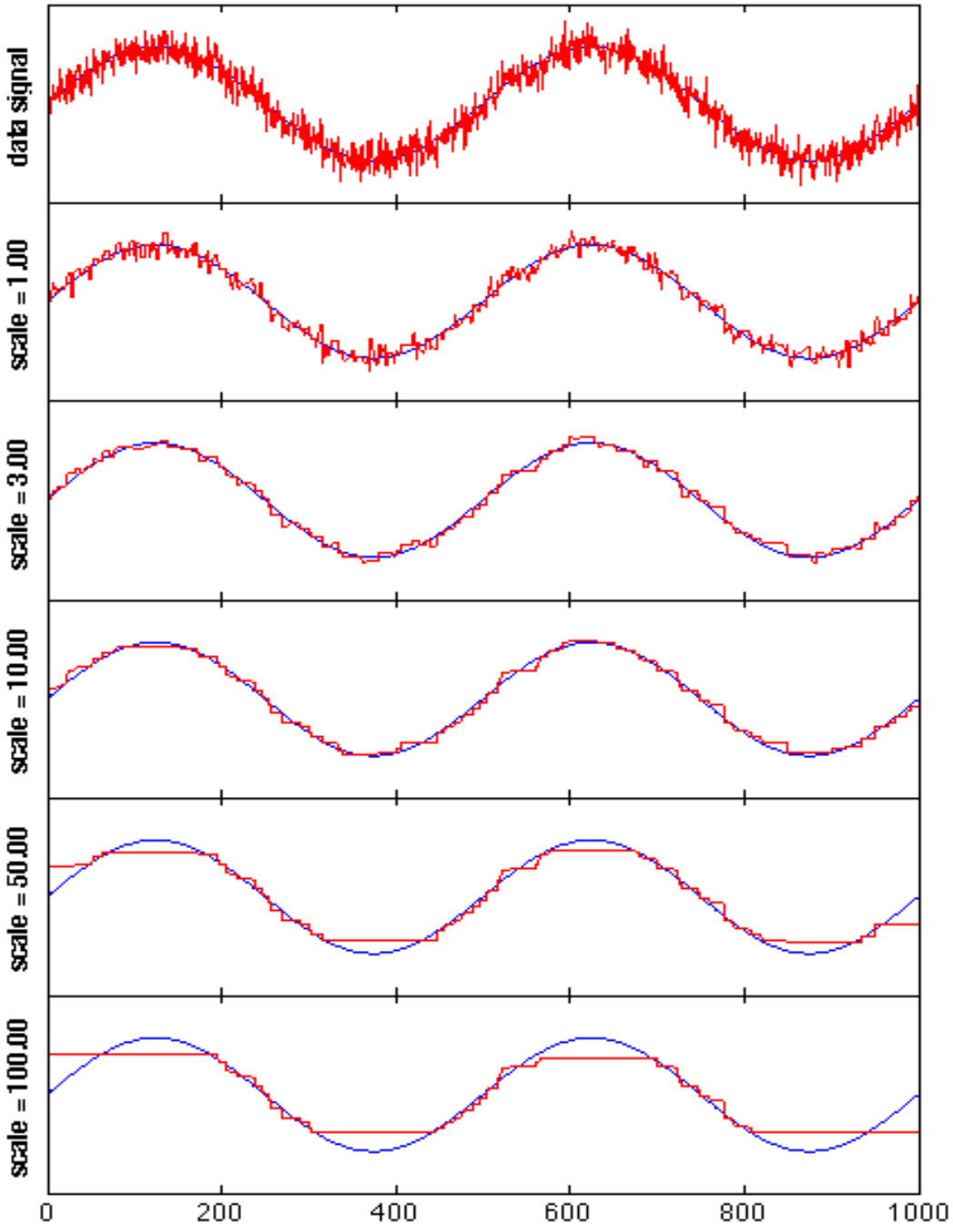


Figure 6.4: Denoising, using Alg 5.3 for L^1TV, a signal composed of noise and a single sine. Blue: ground truth.

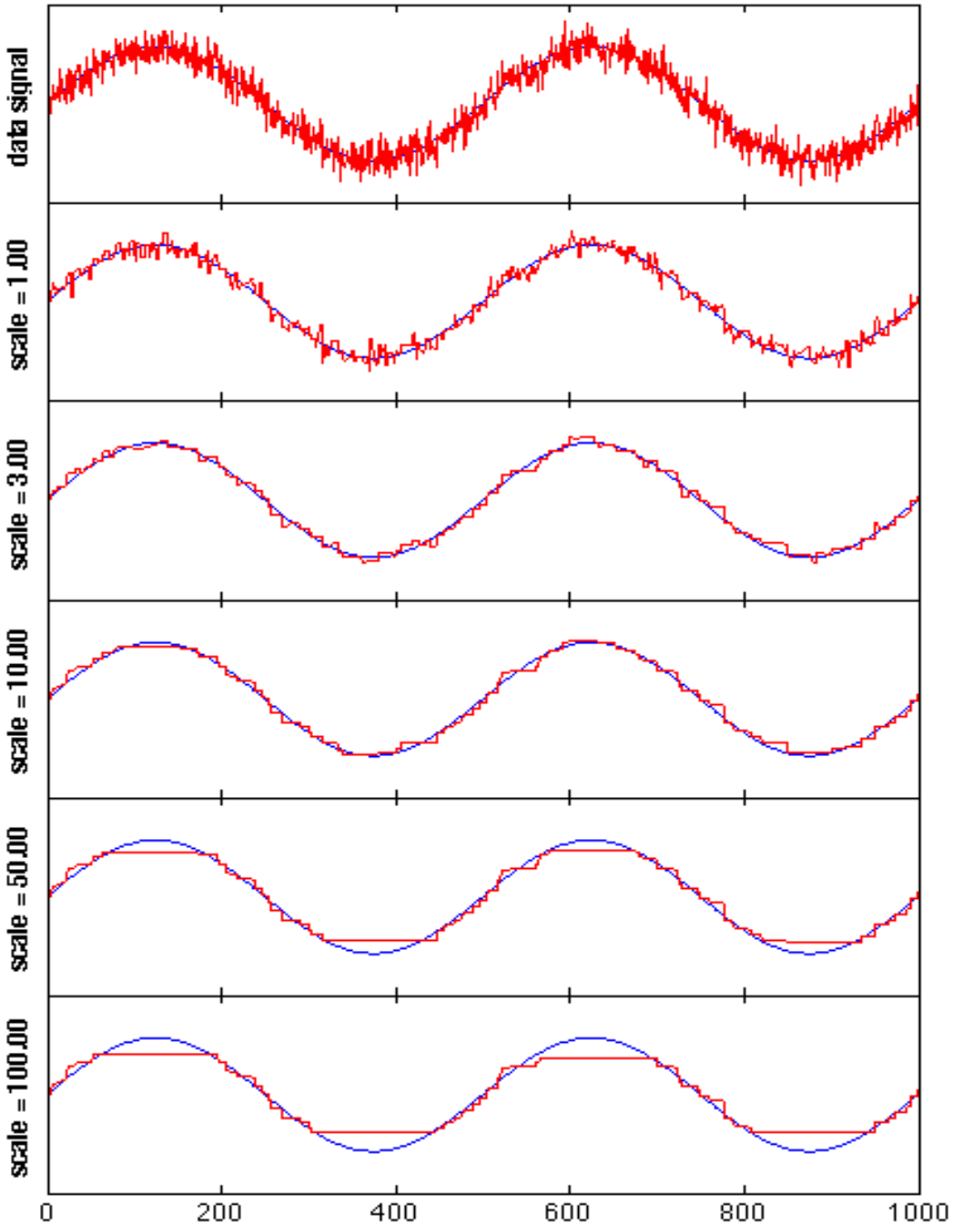


Figure 6.5: Denoising, using Alg 5.3 for L^1TV (with fixed boundaries), a signal composed of noise and a single sine. Blue: ground truth.

Example 6.5. (Denoising) Finally, we test Algorithm 5.3 on a sinusoidal signal with three frequencies and with additive Gaussian noise (see Figure 6.6). In this example, we again see for large λ that most of the small scale noise is removed. We also see that as we decrease λ, more of the smaller scale (higher frequency sines) features are removed. And, as in Example 6.3, the height of the signal drops as $\lambda \searrow 0$.

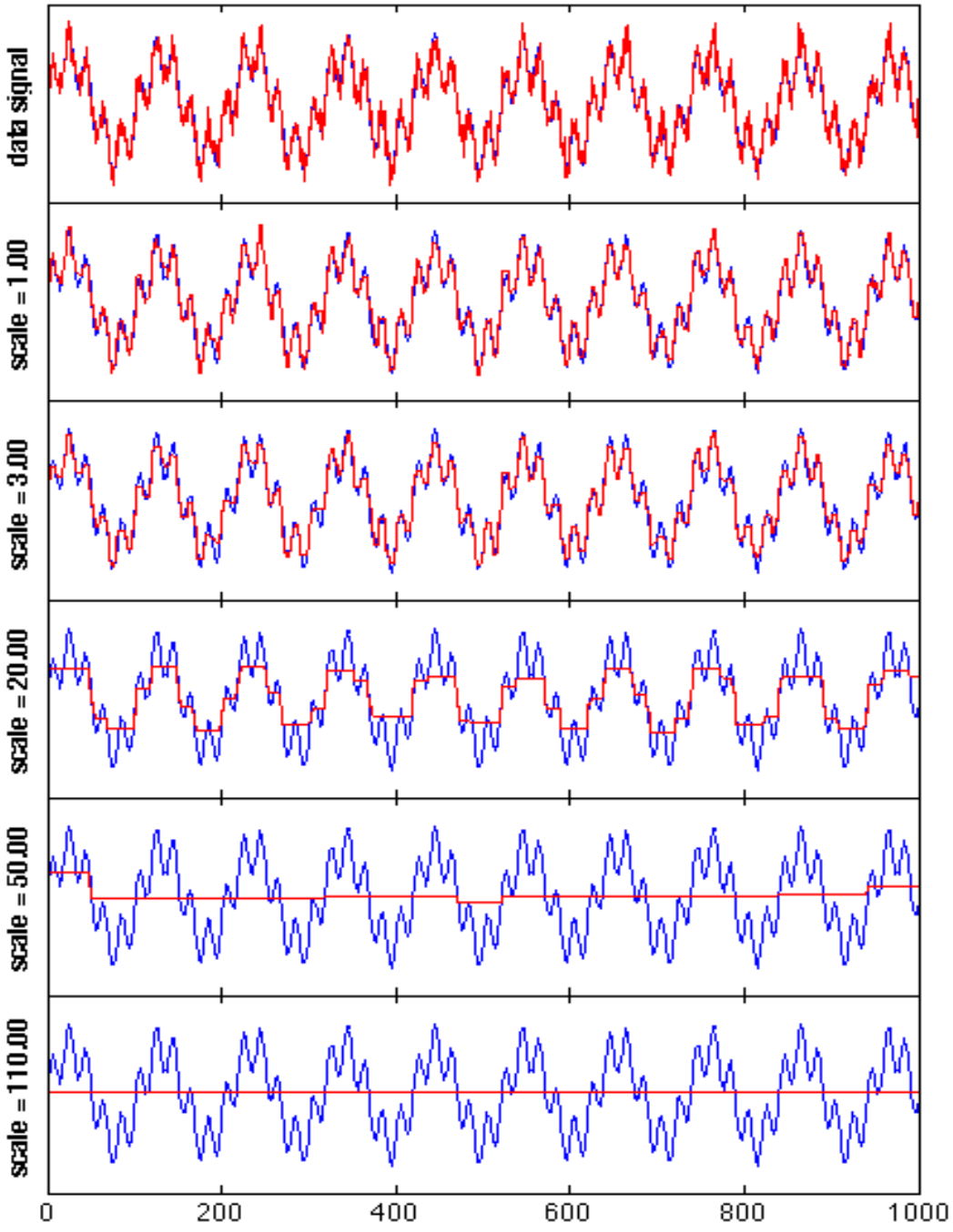


Figure 6.6: Denoising, using Alg 5.3 for L^1TV, a signal composed of noise and the sum of sines. Blue: ground truth.

6.1.2 L^1pTV Examples

In this subsection, we show our ht algorithm for L^1pTV denoising the same signals from Subsection 6.1.1. In these examples we call $2/\lambda$ scale. This does not imply that these values are scales in our signal as they are not, in general.

Example 6.6. (Stationary Gaussian noise) First, consider a signal made of only noise. We observe that, as expected, Algorithm 5.3 removes small scale stationary Gaussian noise for large λ with very little change in signal intensity. See Figure 6.7.

Example 6.7. (Simple denoising) Next, we consider the signal used in Example 6.2 having two steps close together. We used Algorithm 5.4 to denoise this signal. Notice in Figure 6.8 that the small scale noise is removed at scale= 5 and this happened before losing the gap between the two steps. Here, we see that the signal is denoised without losing corners and signal intensity as expected using L^1pTV [8].

As in Subsection 6.1.1, we added more noise to this simple signal. In Figure 6.9 we see that the result when $\lambda = 2/5$ (scale= 5) is a denoised signal that almost exactly matches the original. We also see that for λ much smaller (scale= 50) we begin to lose small scale features. As we expected, using L^1pTV, we preserve the features of the signal better than with L^1TV even with higher noise.

Example 6.8. (Denoising a sinusoidal signal) We continue with the signal given in Example 6.3 (see Figure 6.10). We use Algorithm 5.67 to denoise this signal. We see that as λ decreases, the signal becomes blocky, but the noise is reduced before $\lambda = 2/3$.

Example 6.9. (Denoising a sinusoidal signal with fixed boundaries) As in Example 6.4, we denoise the noisy sinusoidal signal using fixed boundaries. We see, like we saw with L^1TV that the periodicity is preserved for more values of λ.

Example 6.10. (Denoising a signal made of the sum sines) Here we denoise the noisy sine wave from Example 6.5 (see Figure 6.12). We see that as λ decreases, the noise is reduced quickly as we

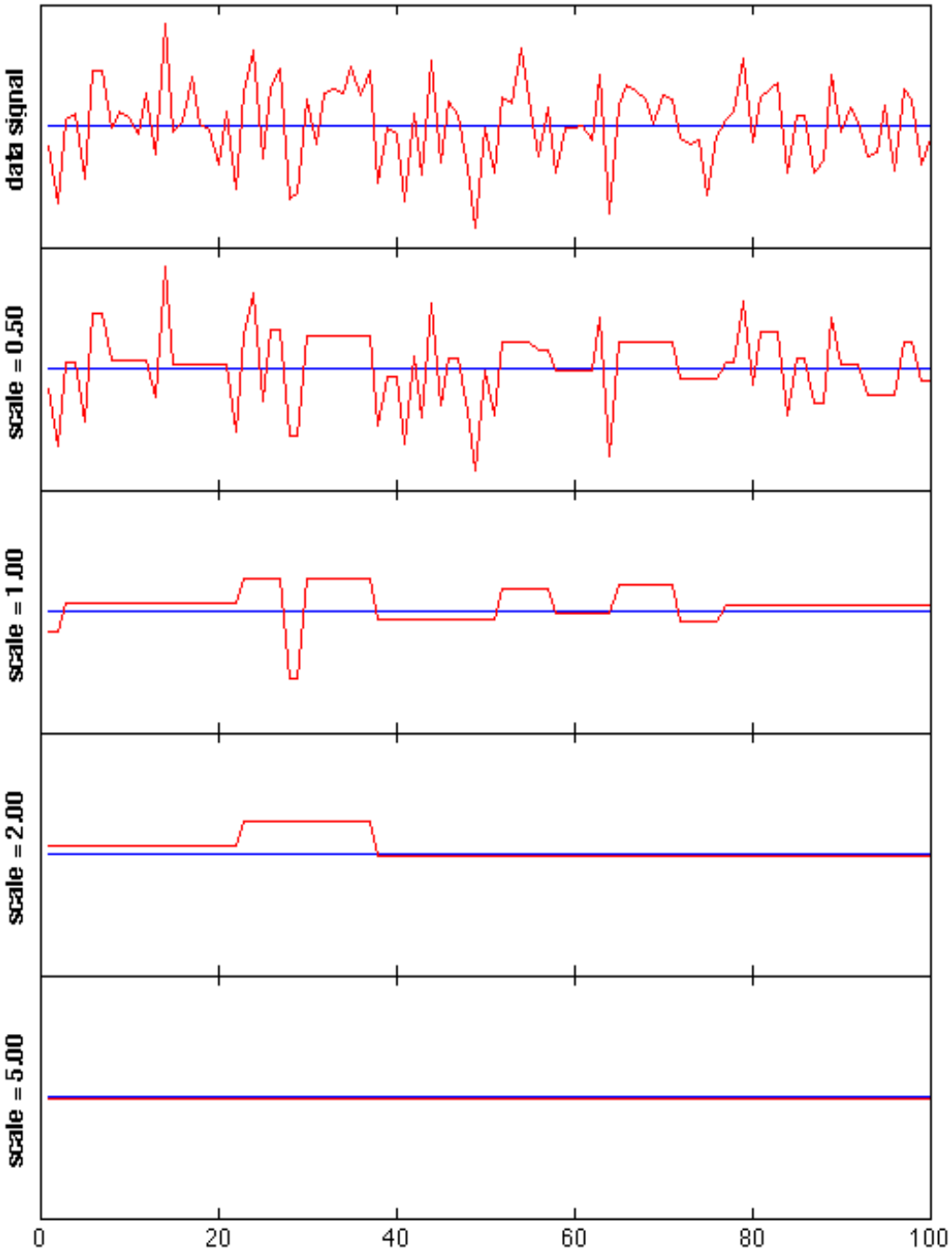


Figure 6.7: Denoising, using Alg 5.4 for L^1pTV, a signal of stationary Gaussian noise. Blue: ground truth.

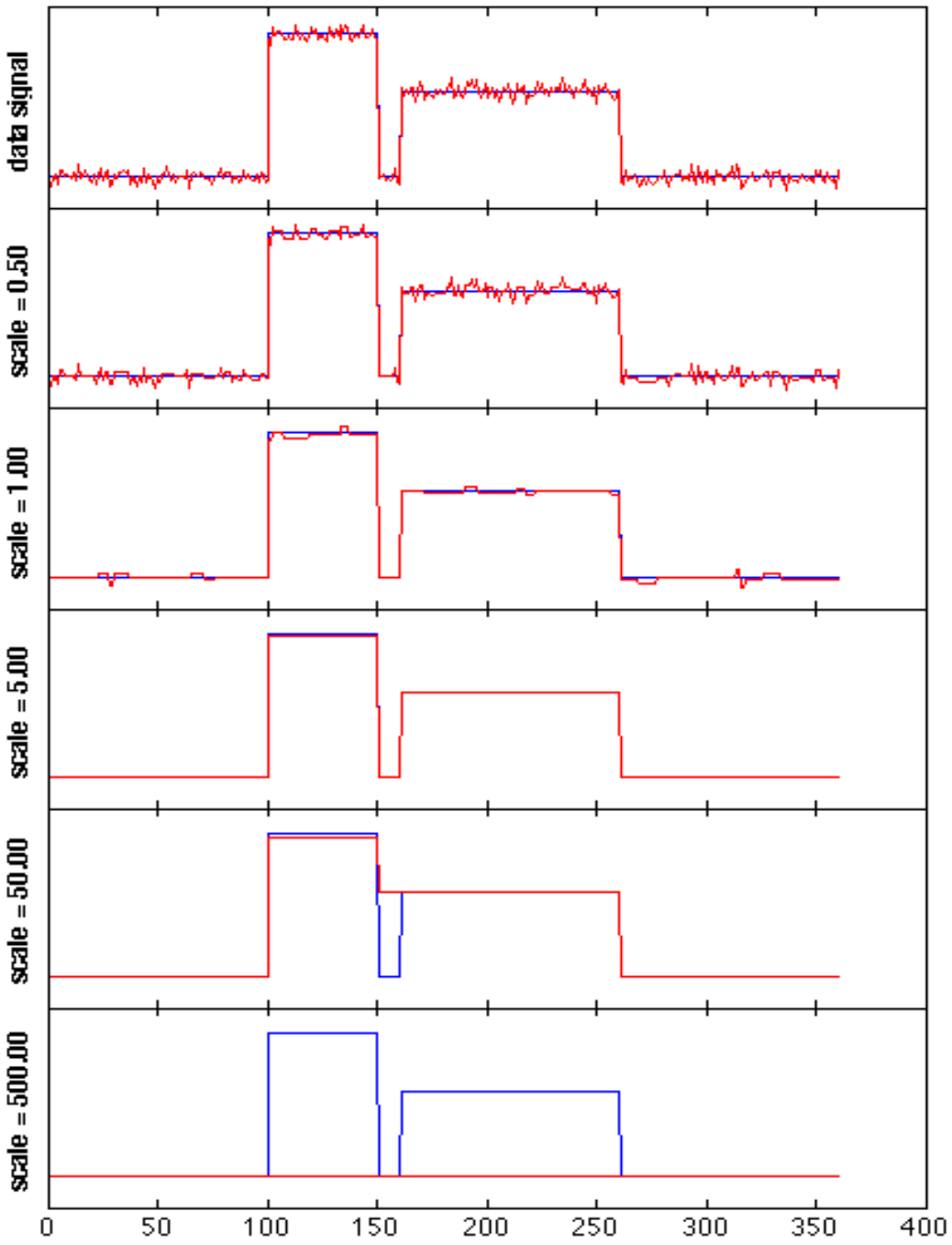


Figure 6.8: Denoising, using Alg 5.4 for L^1pTV, on a simple noisy signal with higher noise. Blue: ground truth.

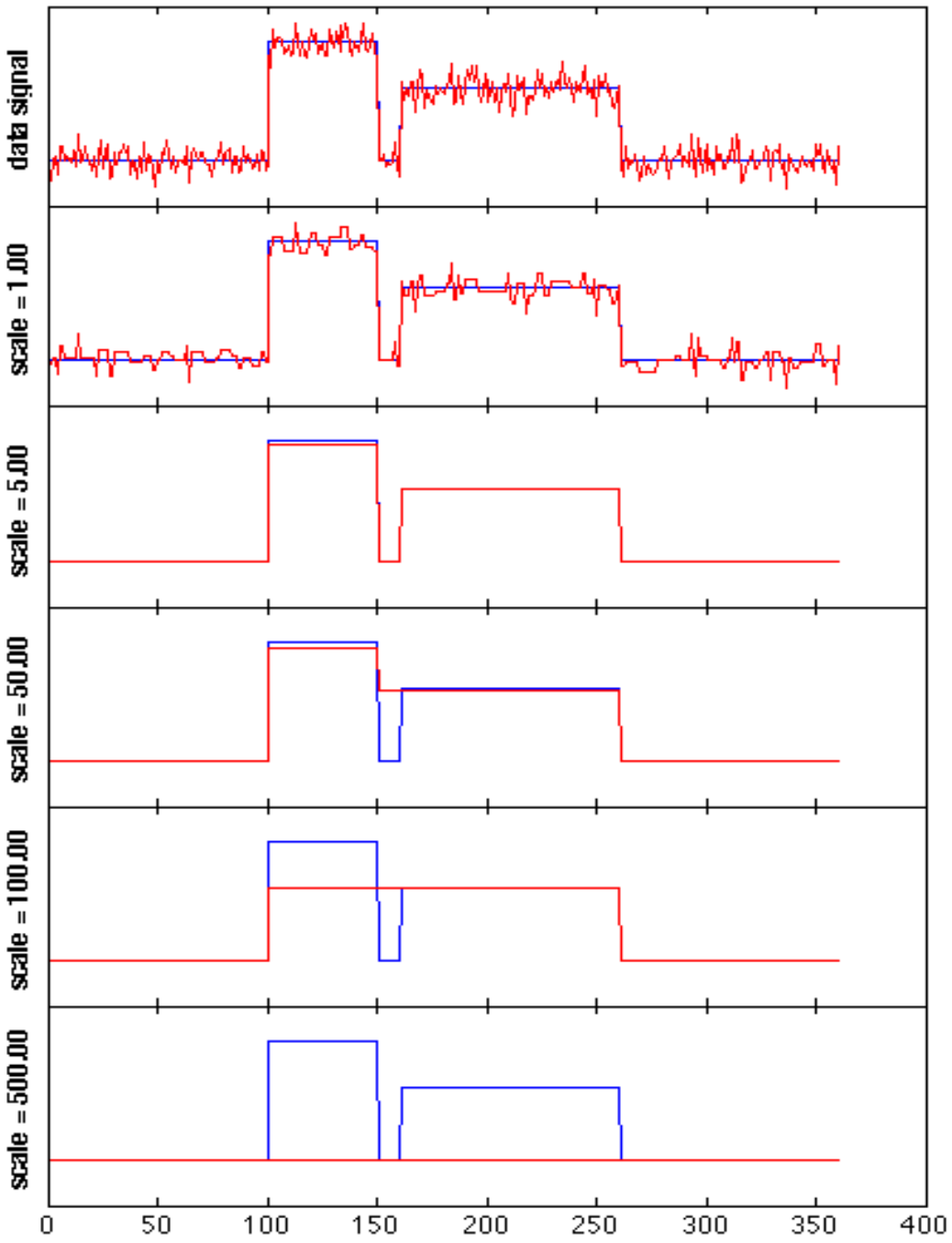


Figure 6.9: Denoising, using Alg 5.4 for L^1pTV, on a simple noisy signal with higher noise. Blue: ground truth.

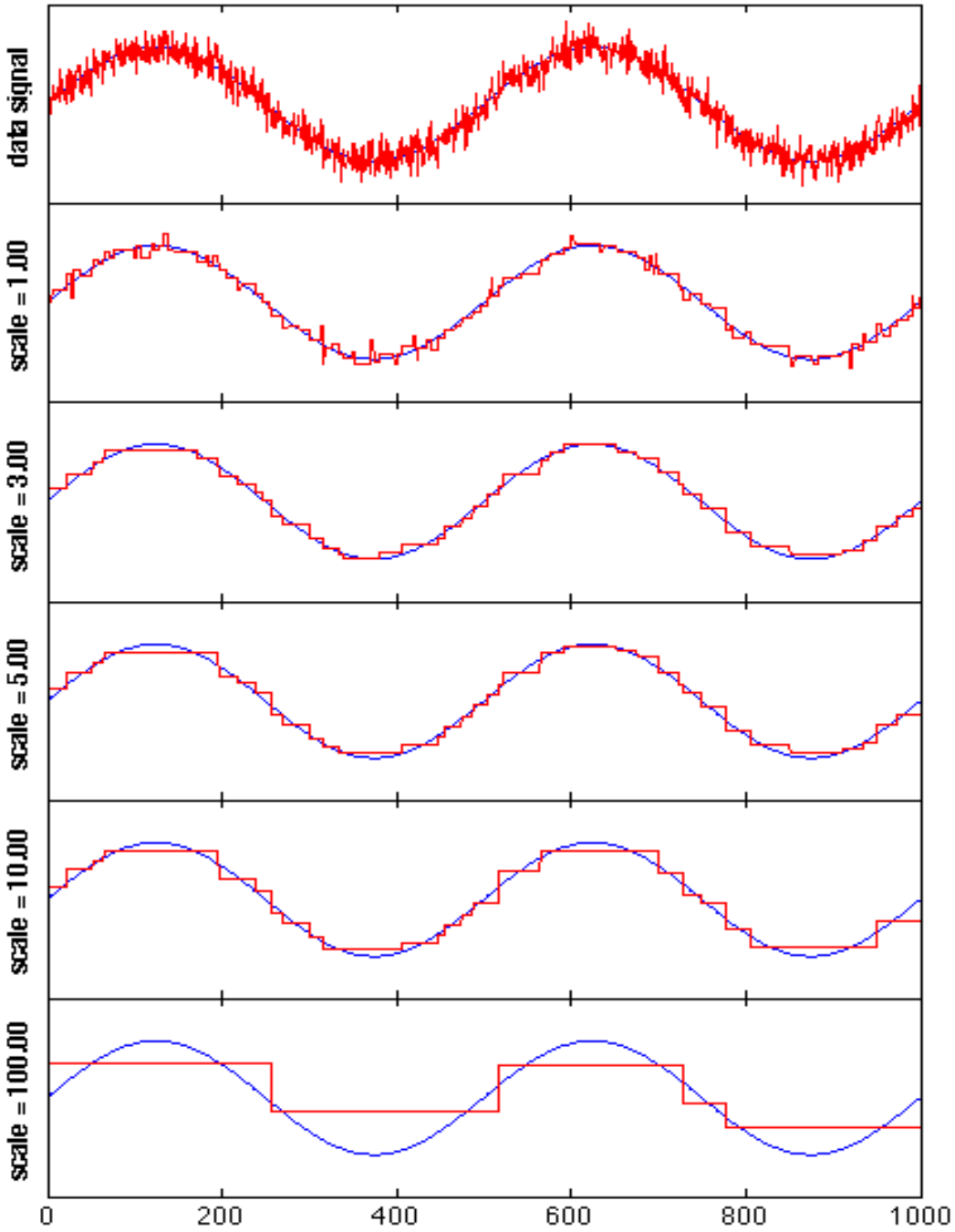


Figure 6.10: Denoising, using Alg 5.4 for L^1pTV, a signal composed of noise and a single sine. Blue: ground truth.

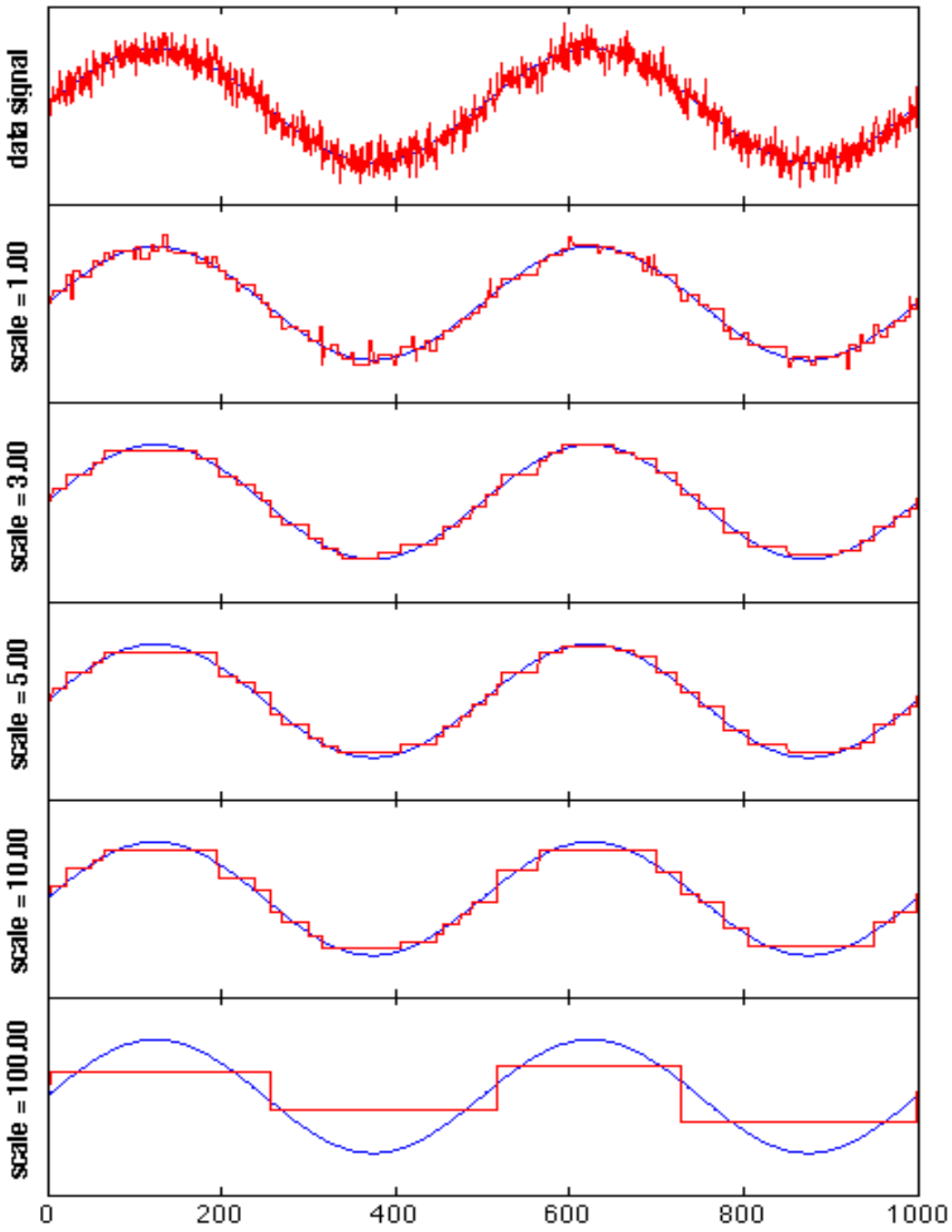


Figure 6.11: Denoising, using Alg 5.4 for L^1pTV (with fixed boundaries), a signal composed of noise and a single sine. Blue: ground truth.

saw with L^1pTV. We also see that as we let $\lambda \searrow 0$ smaller scale features are lost, but the height of the signal is preserved longer.

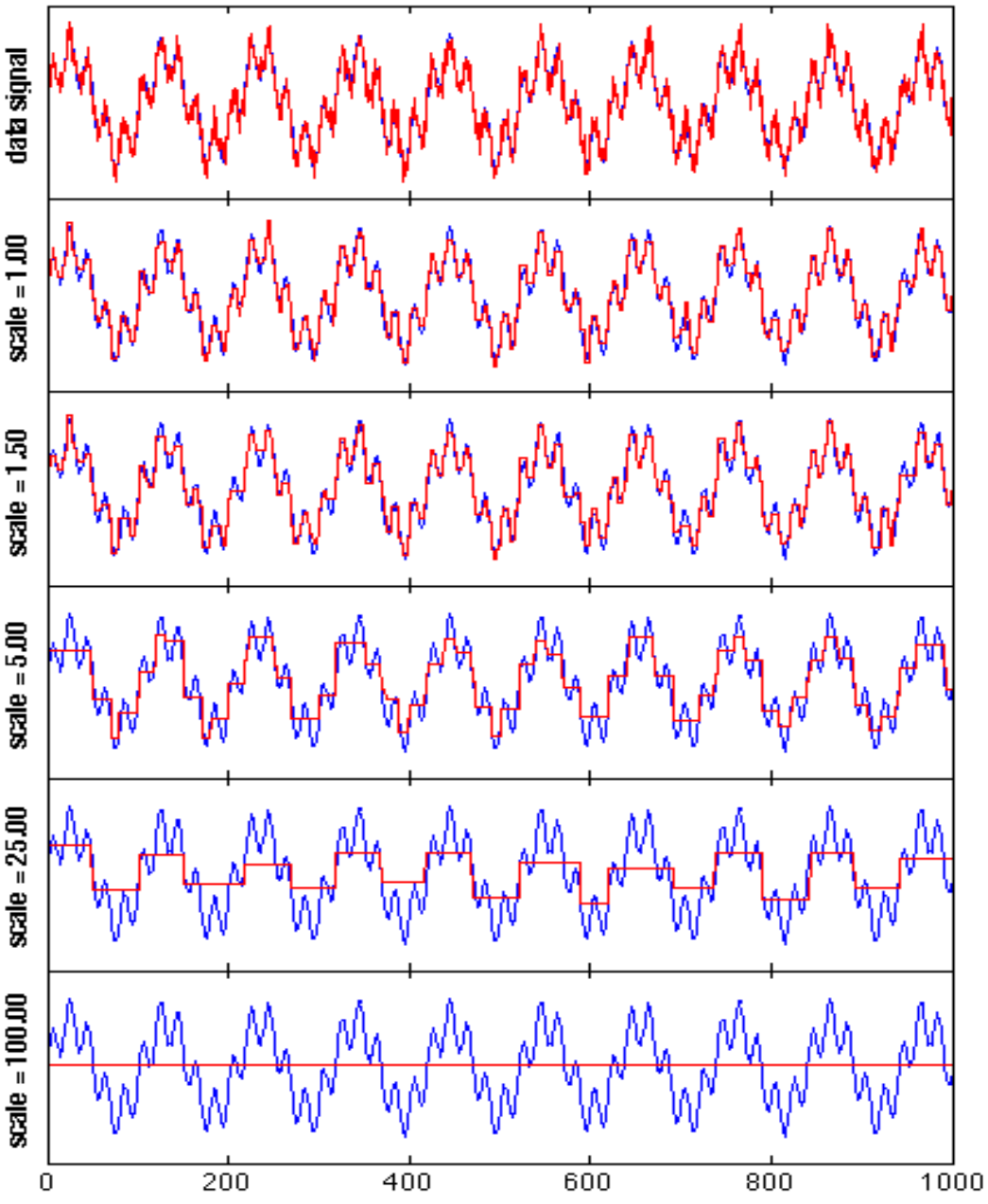


Figure 6.12: Denoising, using Alg 5.4 for L^1pTV, a signal composed of noise and the sum of sines. Blue: ground truth.

6.2 Scale Signatures

In [30], the authors use L^1TV signatures to find scale information in images. We now look at several examples using L^1TV signatures to show that with Algorithm 5.3 we also find scale information in 1-dimensional signals. With our algorithm, we have the added benefit of not needing to choose at which λ we want to use to create the signature because we have the minimizers for all $\lambda > 0$ so we can use them all.

Example 6.11. (Simple Scale information) First we consider a simple signal with four scales, $50, 150, 500$, and 1000. We compute the variation, $s_v(\lambda)$ and data fidelity, $s_f(\lambda)$, terms for all $0 < \lambda < 2$. We find the discrete derivative (using simple forward differencing) of s_v and s_f to get scale signature plots. In the scale signature plots of Figure 6.13, we see peaks corresponding to the scales in our simple signal. As L^1TV finds scales, our algorithm will also find simple scales in a 1-dimensional signal.

Example 6.12. (Single Sine Wave) In this example, we consider another simple signal, this time of a sine wave (see Figure 6.14) with a period of 50. Because the peaks gradually disappear and are symmetric, the variation and the fidelity do not change on every other scale size. Taking this into account, we see that there is really only one peak in the signature near 25 which is half the period.

Example 6.13. (Random Signal) For this example we consider a general random signal. For random signals, we expect to find only small scales (near 1) because there are no objects or structure in the signal. That is, since every data point of the signal is not linked to another, we expect that the only scale in the signal to be of size 1. In Figure 6.15, we see that the scale signature plots indicate that this is the case.

Example 6.14. (Random Binary) In this example, we consider a random binary signal. Like the random signal in Example 6.13, we expect the scale signatures for a random binary signal to indicate only small scale features. Figure 6.16 shows this to be the case.

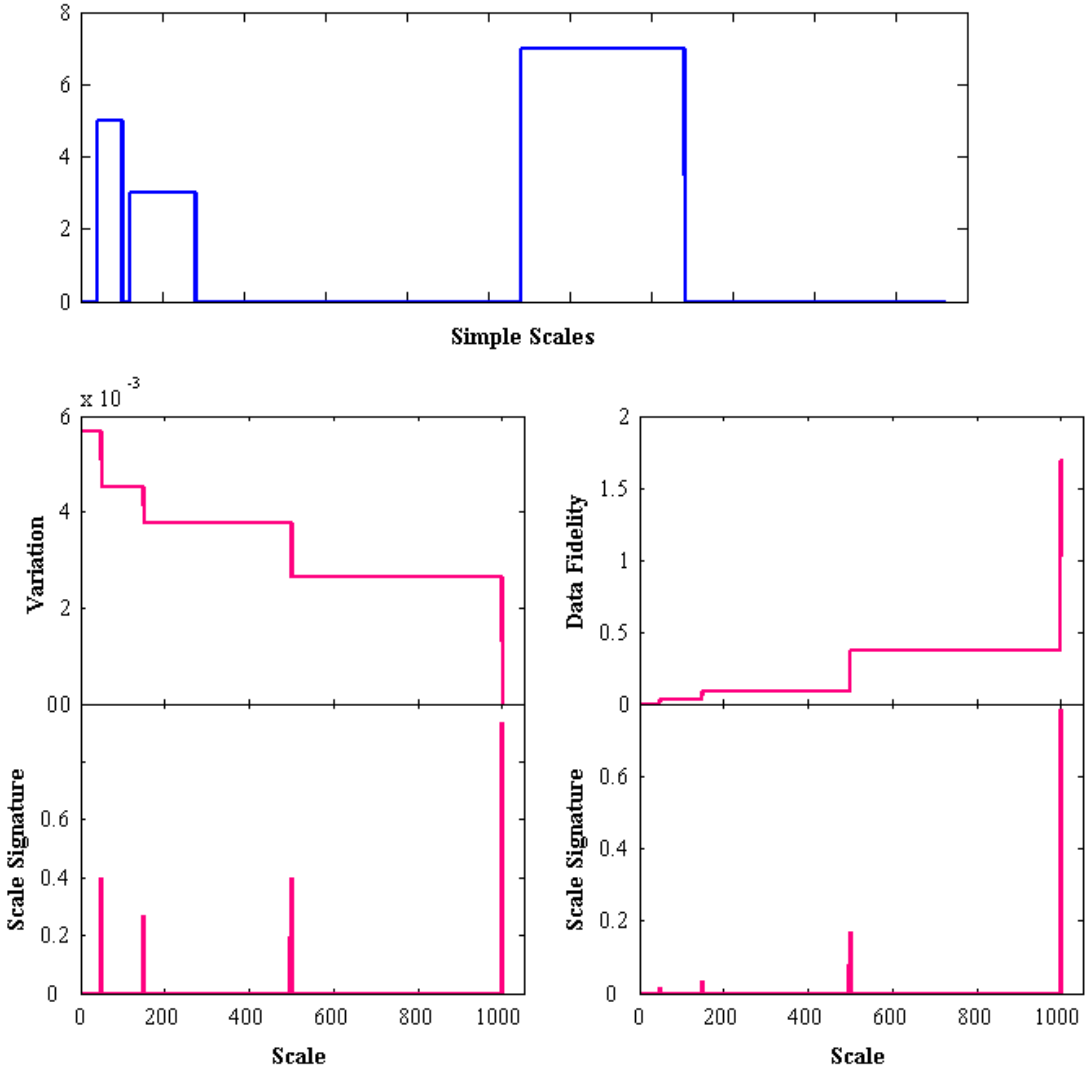


Figure 6.13: Finding scales using Alg 5.3 for a signal with 4 scales (top). Signatures are discrete derivatives of the variation (left) and fidelity (right) terms.

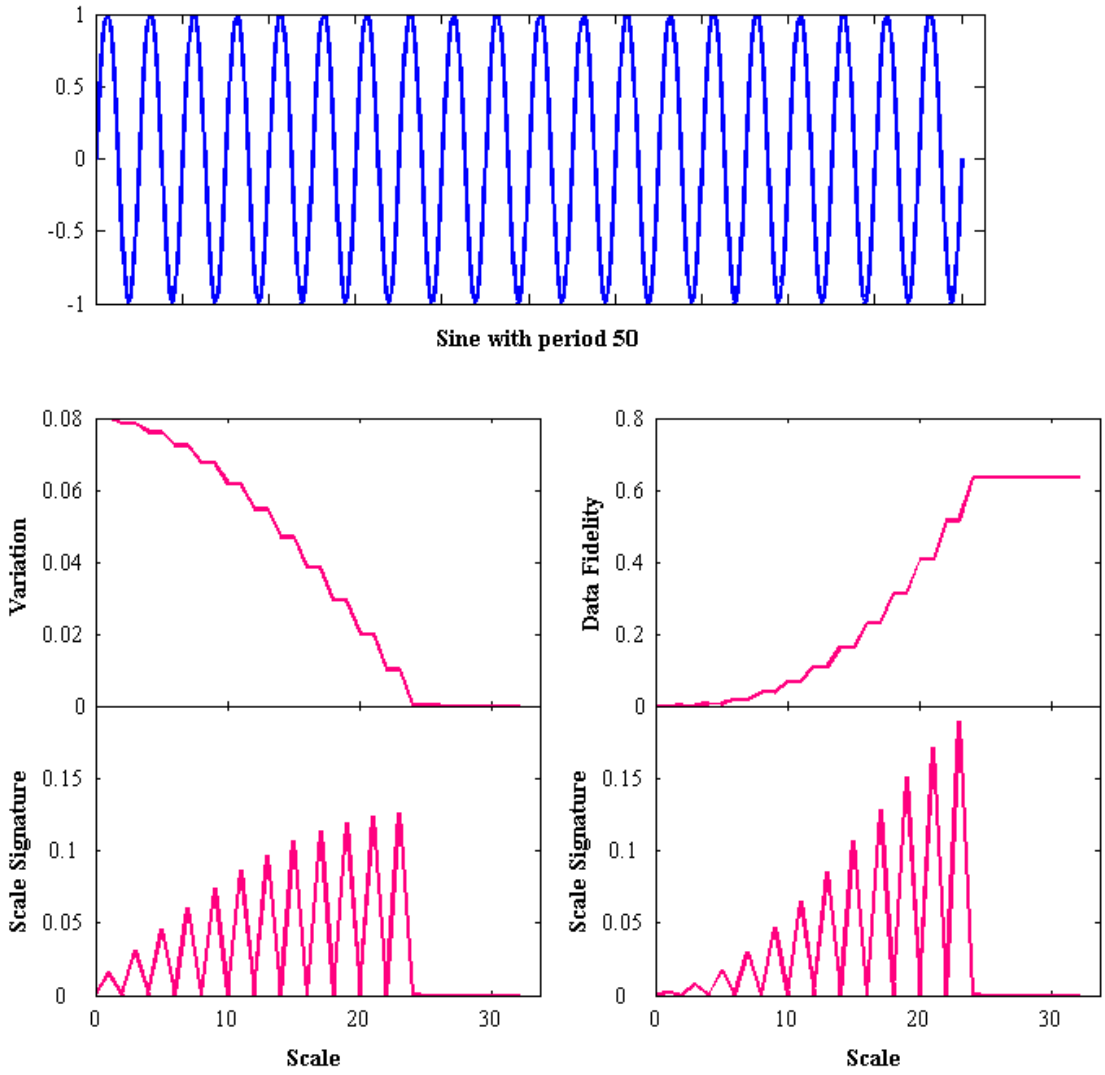


Figure 6.14: Finding scales using Alg 5.3 for a sinusoidal signal with a period of 50. Signatures are discrete derivatives of the variation (left) and fidelity (right) terms.

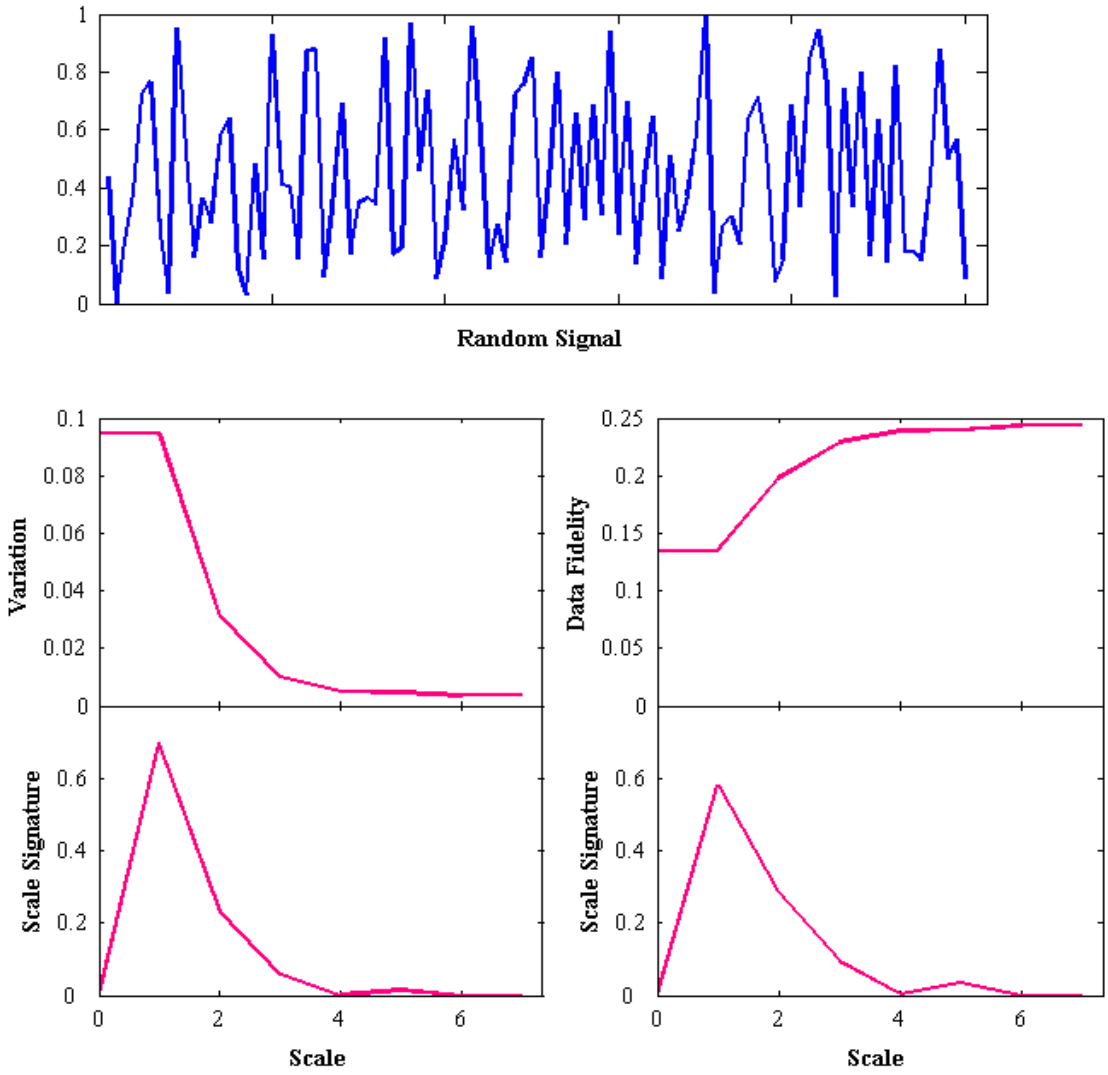


Figure 6.15: Finding scales using Alg 5.3 for a random signal. Signatures are discrete derivatives of the variation (left) and fidelity (right) terms.

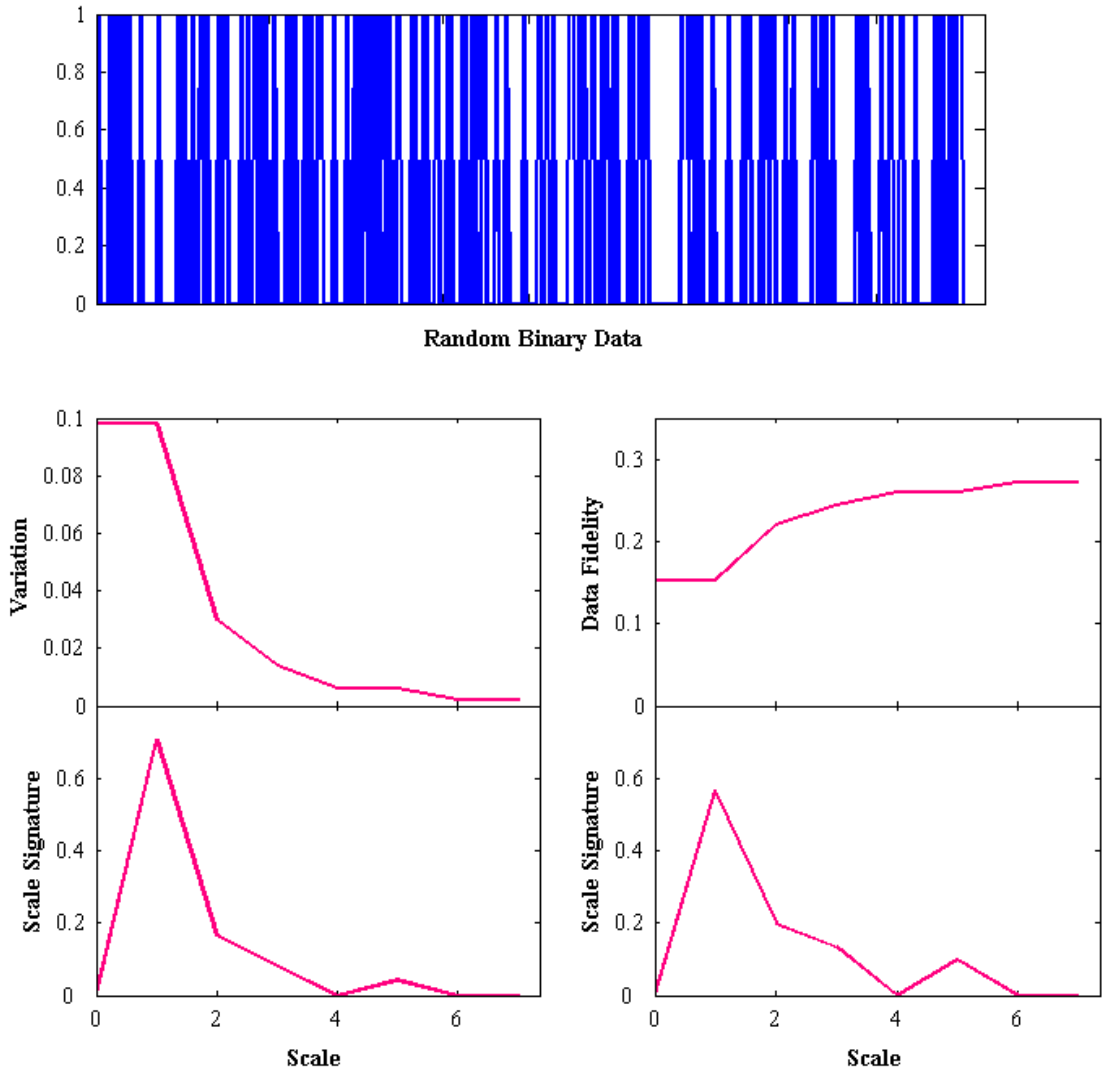


Figure 6.16: Finding scales using Alg 5.3 for a random binary signal. Signatures are discrete derivatives of the variation (left) and fidelity (right) terms.

Example 6.15. (Noisy Sine Wave) In this example, we revisit the noisy sine wave of Example 6.3 (see Figure 6.17. Here we expect to see the small scales from the noise, but also something in the signature to indicate the scale related to the period of the sine wave. In the signature corresponding to the data fidelity term, we see a spike near 1 indicating the small scale noise. We also see something indicating a scale near 40. In this case, the sine function we used is $\sin\left(\frac{\pi}{50}x\right)$. So we expected to see an indication of a scale near 50.

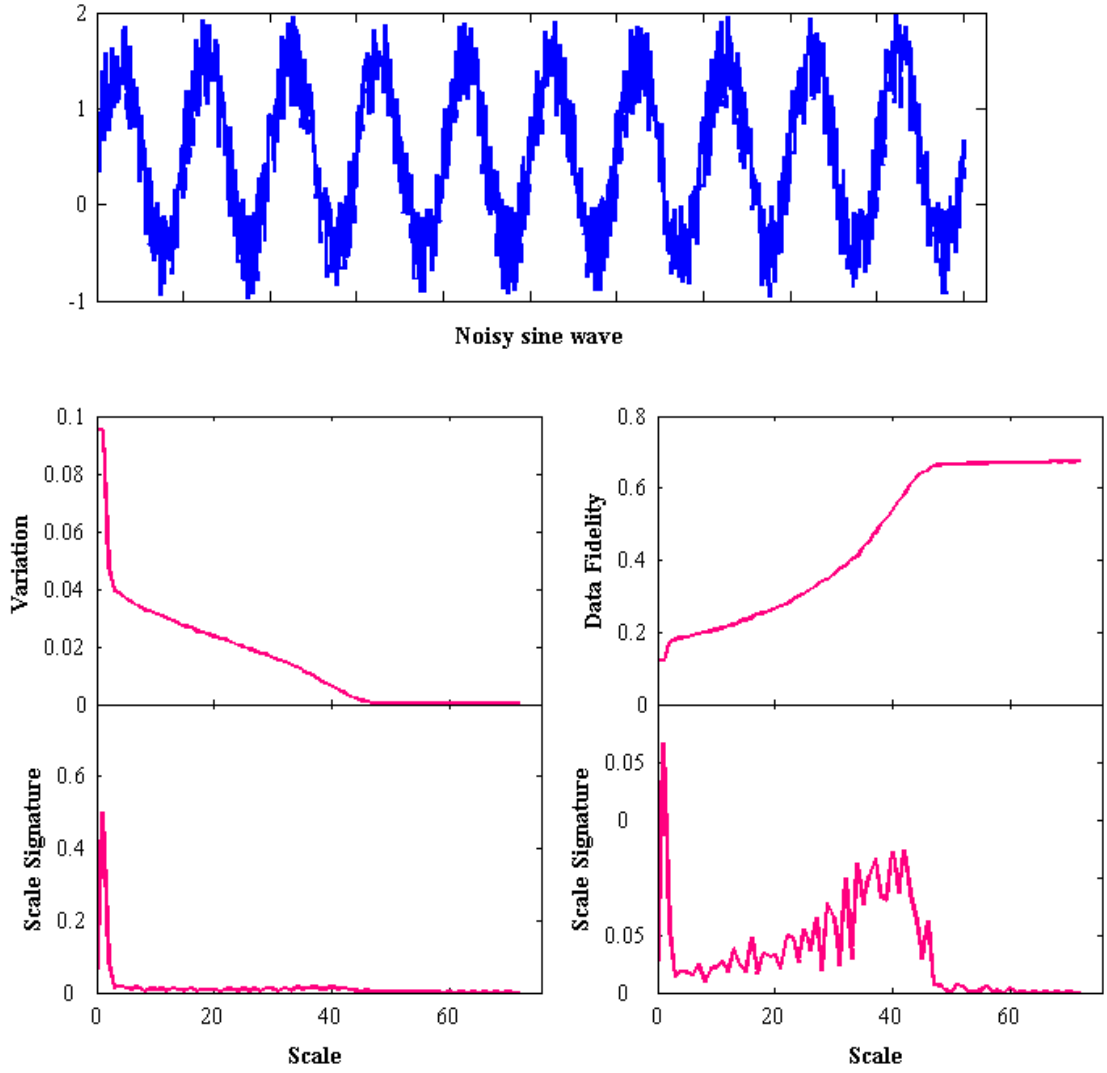


Figure 6.17: Finding scales using Alg 5.3 for a noisy sinusoidal signal. Signatures are discrete derivatives of the variation (left) and fidelity (right) terms.

6.3 Examples Summary

In this chapter we showed several examples that give the expected results for both L^1TV and L^1pTV. We were able to denoise signals using our ht algorithms for L^1TV and L^1pTV That is, for L^1TV, we see that we can eliminate small scale noise by decreasing λ and finding minimizers. We see that there is a loss of signal height as $\lambda \searrow 0$ as is discussed in [6, 24, 4]. We see that steps are preserved when denoising using L^1pTV as expected from [8]. We also showed that our algorithm for L^1TV will give signatures that pick out the scales in 1-dimensional data as suggested in [30].

CHAPTER SEVEN

Conclusion

In this work, we studied problems of the form

$$\min \int_\Omega |\nabla u|^p + \lambda |f - u| \, dx, \quad \text{for } 0 < p \leq 1. \tag{7.1}$$

We considered the L^1TV ($p = 1$), L^1pTV ($0 < p < 1$), and p-variation ($\lambda = 0$ and $0 < p < 1$) cases. Our goals were to find and understand minimizers for each of the cases.

We began by looking at the p-variation case. In this case, we showed that the set of minimizers is the convex set of step functions whose jump set has finite area. We also showed that this set is neither open nor closed. Because the functional $\int_\Omega |\nabla u|^p$ for $p < 1$ is nonconvex we are not guaranteed to find minimizers by finding solutions to the Euler-Lagrange equation. Still, we know that stationary points for $\int_\Omega |\nabla u|^p$ are stationary solutions to the corresponding Euler-Lagrange equation, i.e. the p-Laplacian equation,

$$0 = \nabla \cdot \left(|\nabla u|^{p-2} \nabla u \right) = |\nabla u|^{p-1} \left(\nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) + \frac{p-1}{|\nabla u|} \frac{\nabla u}{|\nabla u|}^T D^2 u \frac{\nabla u}{|\nabla u|} \right). \tag{7.2}$$

and its parabolic counterpart. Because of the singularity when $0 < p \leq 1$, there are difficulties in finding weak and viscosity solutions when $0 < p < 1$. We do not discuss viscosity solutions when the domain has higher dimension than 1 because there still needs to be a discussion about a maximum principle first. Because we are unable to get a bound, using an energy estimate, on the derivatives, we do not consider weak solution techniques either. To avoid the singularity, we

introduce and characterize the class of Normal Monotone functions whose domains are in $\mathbb{R}^n$, where $n \geq 2$. We use the curvature interpretation of the right-hand side of Equation (7.2) to find classical stationary solutions that are Normal Monotone. We found the following families of classical stationary solutions.

- Affine family: For $x \in \mathbb{R}^n$, $u(x) = a \cdot x + b$, where $a \in \mathbb{R}^n, b \in \mathbb{R}$ are constants.

And in n dimensional spherical coordinates $(r, \theta_1, \ldots, \theta_{n-1})$

- Radial family: $u(r) = ar^m + b$, where $m = \frac{n-p}{1-p}$,
- Polar family: $u(\theta_j) = a \int_b^{\theta_j} \csc^{m_j} \theta \, d\theta$, where $m_j = \frac{n-j-1}{p-1}$, where $1 \leq j \leq n-2$ and $0 < \theta_j < \pi$ and
- Azimuthal family: $u(\theta_{n-1}) = a\theta_{n-1} + b$ where $0 < \theta_{n-1} \leq \frac{\pi}{2}$.

In each of the cases for (7.1) that we studied, we formulated a discretization, G, for the problem. We noticed that this discrete function has a nice structure in that it is smooth in convex regions of its domain and is nonsmooth on hyperplanes. Inspired by this structure, we introduced the finite Hyperplane Traversal (ht) algorithms. We recognize that G has (local) minimizers in the set of intersections of $m-1$ of these hyperplanes. Exploiting this, the ht algorithms step to points on the hyperplanes and then remain in these hyperplanes, thus reducing the dimension of the problem and increasing the efficiency as the algorithm progresses.

For the L^1TV problem, our ht algorithm finds a global minimizer for every $\lambda > 0$ with the same computational cost of solving the problem for $\lambda = 0$ by picking up these minimizers at each iteration along the way. Using time trials to estimate the computational cost, we estimate that our algorithm is of order $o(aN + M)$ for a signal of length N, having M initial clusters, and where $0 < a \ll 1$. In the binary case, we show that the algorithm is of order $o(N) + o(M)$ and in the worst case is of order $o(N)$. We also tested our algorithm on some test cases to show that the results match up with those we expect for L^1TV. That is, we see that we can use an L^1TV signature to pick out

scales from the data. In signal denoising, we see that for low noise and the right λ ($\lambda < 2/r$), small scales are preserved [30, 6, 1, 2, 3].

For L^1pTV, we also introduce an efficient ht algorithm similar to the algorithm for L^1TV. The difference between this algorithm and the L^1TV is that we are not guaranteed to find a global minimizer. Instead, we find local minimizers. The algorithm finds a set of local minimizers for all $\lambda > 0$, again, with the same computational cost of only solving one problem, the $\lambda = 0$ problem. We used this algorithm on some test signals for denoising to verify that we see same results as in [8].

Still there are many problems left to consider. In the case of algorithms, it would be nice to extend the idea of ht algorithms to higher dimensions. We discussed in this work the reason the current algorithms do not extend, but it seems that the structure of the function G should give algorithms that are computationally fast. In studying the p-Laplacian evolution equation, we still desire to find paths through the evolution from an initial function to a stationary solution. A more in depth study of weak and viscosity solutions is also needed. One should be able to prove a maximum principle or that it does not exist. There is also much to be explored using the notion of cone monotonicity in finding minimizers of the p-variation problem and stationary solutions to the p-Laplacian evolution equation.

APPENDIX ONE

CALCULUS OF ORTHOGONAL COORDINATE SYSTEMS

A.1 Gradients

In this section we give general formulas for the gradient. Given any orthogonal coordinate system $(q^1, q^2, \ldots, q^n)$ in $\mathbb{R}^n$ we compute basis vectors $\hat{\mathbf{e}}_\mathbf{1}, \hat{\mathbf{e}}_\mathbf{2}, \ldots, \hat{\mathbf{e}}_\mathbf{n}$. We begin with a position vector in $\mathbb{R}^n$

$$\mathbf{p} = x_1\mathbf{e_1} + \ldots + x_n\mathbf{e_n}. \tag{A.1}$$

To find $\hat{\mathbf{e}}_\mathbf{i}$, we find how the position changes as q^i changes, that is

$$\hat{\mathbf{e}}_\mathbf{i} = \frac{\frac{\partial \mathbf{p}}{\partial q^i}}{|\frac{\partial \mathbf{p}}{\partial q^i}|} = \frac{\sum_{j=1}^{n} \frac{\partial x_j}{\partial q^i}\mathbf{e_i}}{\left(\sum_{j=1}^{n}\left(\frac{\partial x_j}{\partial q^i}\right)^2\right)^{1/2}}. \tag{A.2}$$

Notice that if $\{\hat{\mathbf{e}}_\mathbf{1}, \hat{\mathbf{e}}_\mathbf{2}, \ldots, \hat{\mathbf{e}}_\mathbf{n}\}$ is an orthonormal basis, we must have that $\hat{\mathbf{e}}_\mathbf{i} \cdot \hat{\mathbf{e}}_\mathbf{j} = 0$ whenever $i \neq j$ and $\hat{\mathbf{e}}_\mathbf{i} \cdot \hat{\mathbf{e}}_\mathbf{i} = 1$. By constructions, we can see that $|\hat{\mathbf{e}}_\mathbf{i}| = 1$. In order for $\hat{\mathbf{e}}_\mathbf{i} \cdot \hat{\mathbf{e}}_\mathbf{j} = 0$, we require that

$$\sum_{k=1}^{n} \frac{\partial x_k}{\partial q^i} \cdot \frac{\partial x_k}{\partial q^j} = 0. \tag{A.3}$$

Next we compute scale factors $h_1, h_2 \ldots h_n$ by recognizing that the distance between two points is preserved no matter the coordinate system in which we are working. So, we consider an in-

finitesimal distance ds between two points. Using the pythagorean theorem, we can write

$$ds^2 = dx_1^2 + dx_2^2 + \ldots + dx_n^2. \tag{A.4}$$

Using the chain rule we write $dx_1, \ldots, dx_n$ in terms of $dq^1, \ldots, dq^n$ as

$$\begin{aligned}
dx_1 &= \frac{\partial x_1}{\partial q^1} dq^1 + \frac{\partial x_1}{\partial q^2} dq^2 + \ldots + \frac{\partial x_1}{\partial d^n} && \text{(A.5)}\\
dx_2 &= \frac{\partial x_2}{\partial q^1} dq^1 + \frac{\partial x_2}{\partial q^2} dq^2 + \ldots + \frac{\partial x_2}{\partial d^n} && \text{(A.6)}\\
&\vdots && \text{(A.7)}\\
dx_n &= \frac{\partial x_n}{\partial q^1} dq^1 + \frac{\partial x_n}{\partial q^2} dq^2 + \ldots + \frac{\partial x_n}{\partial d^n} && \text{(A.8)}\\
& && \text{(A.9)}
\end{aligned}$$

Putting these into the above equation gives

$$\begin{aligned}
ds^2 &= \left(\sum_{i=1}^{n} \frac{\partial x_1}{\partial q^i} dq^i\right)^2 + \left(\sum_{i=1}^{n} \frac{\partial x_2}{\partial q^i} dq^i\right)^2 + \ldots + \left(\sum_{i=1}^{n} \frac{\partial x_n}{\partial q^i} dq^i\right)^2 && \text{(A.10)}\\
&= \sum_{i=1}^{n} \left(\frac{\partial x_i}{\partial q^2}\right)^2 \left(dq^1\right)^2 + \sum_{i=1}^{n} \left(\frac{\partial x_i}{\partial q^2}\right)^2 \left(dq^2\right)^2 + \ldots + \sum_{i=1}^{n} \left(\frac{\partial x_i}{\partial q^n}\right)^2 (dq^n)^2 && \text{(A.11)}\\
&\equiv h_1^2 \left(dq^1\right)^2 + h_2^2 \left(dq^2\right)^2 + \ldots + h_n \left(dq^n\right)^2 . && \text{(A.12)}
\end{aligned}$$

Note: The mixed terms with $dq^i dq^j$ will cancel since the coefficient of $dq^i dq^j$ is $\sum_{k=1}^{n} \frac{\partial x_k}{\partial q^i} \cdot \frac{\partial x_k}{\partial q^j}$ which, by the orthogonal condition (A.3), is zero.

We call the coefficients $h_1, h_2, \ldots, h_n$ the scale factors because for a point to move an infinitesimal distance in the q^i direction, the distance $ds = h_1 dq^i$ rather than dq^i. This leads us to the gradient in a general orthogonal coordinate system,

$$\nabla_{q^1 q^2 \ldots q^n} f = \sum_{j=1}^{n} \frac{1}{h_j} \frac{\partial f}{\partial q^j} \hat{e}_j, \tag{A.13}$$

where

$$h_j = \left(\sum_{i=1}^{n} \left(\frac{\partial x_i}{\partial q^j} \right)^2 \right)^{1/2}. \tag{A.14}$$

We write $\nabla_{q^1 q^2 \ldots q^n}$ to indicate that the gradient is in the $(q^1, q^2, \ldots, q^n)$ coordinate system.

For polar coordinates, we use the following change of variables from rectangular coordinates compute the scale factors h_1, h_2.

$$x = r\cos\theta \tag{A.15}$$

$$y = r\sin\theta. \tag{A.16}$$

$$dx = \cos\theta dr - r\sin\theta d\theta \tag{A.17}$$

$$dy = \sin\theta dr + r\cos\theta d\theta. \tag{A.18}$$

Putting these into the above equation gives

$$ds^2 = (\cos\theta dr - r\sin\theta d\theta)^2 + (\sin\theta dr + r\cos\theta d\theta)^2 \tag{A.19}$$

$$= \left(\cos^2\theta + \sin^2\theta\right)(dr)^2 + (-2r\cos\theta\sin\theta + 2r\cos\theta\sin\theta) \tag{A.20}$$

$$+r^2\left(\cos^2\theta + \sin^2\theta\right)(d\theta)^2 \tag{A.21}$$

$$= (dr)^2 + r^2(d\theta)^2. \tag{A.22}$$

This gives us the scale factors $h_1 = 1, h_2 = r$. Thus, the gradient in polar coordinates is

$$\nabla_{r,\theta} = \begin{pmatrix} \frac{\partial}{\partial r} \\ \frac{1}{r}\frac{\partial}{\partial\theta} \end{pmatrix} \quad \text{or} \quad \nabla_{r,\theta} f = \begin{pmatrix} f_r \\ \frac{1}{r} f_\theta \end{pmatrix}. \tag{A.23}$$

For n-dimensional spherical coordinates $(r, \theta_1, \theta_2, \ldots, \theta_{n-1})$, we use the change of variables

$$\begin{aligned} x_1 &= r\cos\theta_1 \\ x_2 &= r\sin\theta_1\cos\theta_2 \\ x_3 &= r\sin\theta_1\sin\theta_2\cos\theta_3 \\ &\vdots \\ x_{n-1} &= r\sin\theta_1\ldots\sin\theta_{n-2}\cos\theta_{n-1} \\ x_n &= r\sin\theta_1\ldots\sin\theta_{n-2}\sin\theta_{n-1} \end{aligned} \tag{A.24}$$

For ease of notation, let $S_i \equiv \sin\theta_i$ and $C_i \equiv \cos\theta_i$. With these, we compute $h_1, \ldots, h_n$ as

$$\begin{aligned} h_1 &= \left(C_1^2 + S_1^2C_2^2 + S_1^2S_2^2C_2^2 + \ldots + S_1^2\cdots S_{n-2}^2C_{n-1}^2 + S_1^2\cdots S_{n-2}^2S_{n-1}^2\right)^{1/2} = 1. \\ h_2 &= \left(-rS_1^2 + rC_1^2C_2^2 + rC_1^2S_2^2C_2^2 + \ldots + rC_1^2\cdots S_{n-1}^2C_{n-1}^2 + rC_1^2\cdots S_{n-1}^2S_{n-1}^2\right)^{1/2} \\ &= r. \\ h_3 &= \left(-rS_1^2S_2^2 + rS_1^2S_2^2C_2^2 + r\ldots + S_1^2C_2^2\cdots S_{n-2}^2C_{n-1}^2 + rS_1^2C_2^2\cdots S_{n-1}^2S_{n-1}^2\right)^{1/2} \\ &= r\sin\theta_1. \end{aligned} \tag{A.25}$$

$$\vdots \tag{A.26}$$

$$\begin{aligned} h_k &= \Big(-rS_1^2S_2^2\cdots S_{k-1} + rS_1^2S_2^2S_2^2\cdots S_{k-2}^2C_{k-1}^2 \\ &\quad + \ldots + rS_1^2\cdots S_{k-2}^2S_{k-1}^2\cdots S_{n-2}^2C_{n-1}^2 + rS_1^2\cdots S_{k-2}^2C_{k-1}^2\cdots S_{n-2}^2C_{n-1}^2 \\ &\quad + S_1^2S_2^2\cdots S_{k-2}^2C_{k-1}^2\cdots S_{n-2}^2S_{n-1}^2\Big)^{1/2} = r\sin\theta_1\cdots\sin\theta_{k-2}. \end{aligned}$$

We then get that $\nabla_{r\theta_1\theta_2\ldots\theta_{n-1}} f$ is given by

$$\nabla_{r\theta_1\theta_2\ldots\theta_{n-1}} f = \begin{pmatrix} f_r \\ \frac{1}{r} f_{\theta_1} \\ \frac{1}{r\sin\theta_1} f_{\theta_2} \\ \frac{1}{r\sin\theta_1\sin\theta_2} f_{\theta_3} \\ \frac{1}{r\sin\theta_1\sin\theta_2\sin\theta_3} f_{\theta_4} \\ \vdots \\ \frac{1}{r\sin\theta_1\cdots\sin\theta_{n-2}} f_{\theta_{n-1}} \end{pmatrix}. \tag{A.27}$$

A.2 Hessians

We now use that the Hessian of a function f in any orthogonal coordinate system $(q^1, \ldots, q^n)$ is computed as the outer product of $\nabla_{q^1\ldots q^n}$ and $\nabla_{q^1\ldots q^n} f$. That is,

$$D^2_{q^1q^2\ldots q^n}(f) = \left(\nabla_{q^1q^2\ldots q^n}\right)\left(\nabla_{q^1q^2\ldots q^n} f\right)^T = \left(\sum_{j=1}^n h_j \frac{\partial}{\partial q^j}\hat{e}_j\right)\left(\sum_{j=1}^n h_j \frac{\partial f}{\partial q^j}\hat{e}_j\right)^T \tag{A.28}$$

$$= \begin{pmatrix} h_1\frac{\partial}{\partial q^1} \\ \vdots \\ h_n\frac{\partial}{\partial q^n} \end{pmatrix}\begin{pmatrix} h_1\frac{\partial f}{\partial q^1} & \ldots & h_n\frac{\partial f}{\partial q^n} \end{pmatrix} \tag{A.29}$$

$$= \begin{pmatrix} h_1\frac{\partial}{\partial q^1}\left(h_1\frac{\partial f}{\partial q^1}\right) & \ldots & h_1\frac{\partial}{\partial q^1}\left(h_i\frac{\partial f}{\partial q^i}\right) & \ldots & h_1\frac{\partial}{\partial q^1}\left(h_n\frac{\partial f}{\partial q^n}\right) \\ \vdots & \ddots & & & \vdots \\ h_i\frac{\partial}{\partial q^i}\left(h_1\frac{\partial f}{\partial q^1}\right) & & h_i\frac{\partial}{\partial q^i}\left(h_i\frac{\partial f}{\partial q^i}\right) & & h_i\frac{\partial}{\partial q^i}\left(h_n\frac{\partial f}{\partial q^n}\right) \\ \vdots & & & \ddots & \vdots \\ h_n\frac{\partial}{\partial q^n}\left(h_1\frac{\partial f}{\partial q^1}\right) & \ldots & h_n\frac{\partial}{\partial q^n}\left(h_i\frac{\partial f}{\partial q^i}\right) & \ldots & h_n\frac{\partial}{\partial q^n}\left(h_n\frac{\partial f}{\partial q^n}\right) \end{pmatrix}. \tag{A.30}$$

In polar coordinates, we get

$$D^2_{r\theta}(f) = \begin{pmatrix} f_{rr} & \frac{\partial}{\partial r}\left(\frac{1}{r} f_\theta\right) \\ \frac{1}{r} f_{\theta r} & \frac{1}{r^2} f_{\theta\theta} \end{pmatrix}. \tag{A.31}$$

In n-dimensional spherical coordinates we get

$$D^2_{r\theta_1 \dots \theta_{n-1}}(f) = \tag{A.32}$$

$$\begin{pmatrix} \frac{\partial^2 f}{\partial r^2} & \dots & \frac{\partial}{\partial r}\left(r\Pi_{k=1}^{j-2} \sin\theta_k \frac{\partial f}{\partial \theta_j}\right) & \dots & \frac{\partial}{\partial r}\left(r\Pi_{k=1}^{n-2} \sin\theta_k \frac{\partial f}{\partial \theta_{n-1}}\right) \\ \vdots & \ddots & & & \vdots \\ r\Pi_{k=1}^{i-2} \sin\theta_k \frac{\partial}{\partial \theta_i}\left(\frac{\partial f}{\partial r}\right) & & \left(r\Pi_{k=1}^{i-2} \sin\theta_k\right)^2 \frac{\partial^2 f}{\partial (\theta_i)^2} & & r\Pi_{k=1}^{i-2} \sin\theta_k \frac{\partial}{\partial \theta_i}\left(r\Pi_{k=1}^{n-2} \sin\theta_k \frac{\partial f}{\partial \theta_{n-1}}\right) \\ \vdots & & & \ddots & \vdots \\ r\Pi_{k=1}^{n-2} \sin\theta_k \frac{\partial^2 f}{\partial r \theta_{n-1}} & \dots & & \dots & \left(r\Pi_{k=1}^{n-2} \sin\theta_{n-1}\right)^2 \frac{\partial^2 f}{(\partial \theta_{n-1})^2} \end{pmatrix}. \tag{A.33}$$

A.3 Divergence

In [5] the divergence of $F = (F_1, \dots, F_n)$ in any orthogonal coordinate system $(q^1, \dots, q^n)$ is given by

$$\nabla_{q^1 \dots q^n} \cdot F = \frac{1}{h_1 \cdots h_n}\left(\frac{\partial}{\partial q^1}(h_2 \cdots h_n F_1) + \dots + \frac{\partial}{\partial q^n}(h_1 \cdots h_{n-1} F_n)\right) \tag{A.34}$$

In polar coordinates, this is

$$\nabla_{r,\theta} \cdot F = \frac{1}{r}\left(\frac{\partial}{\partial r}(rF_1) + \frac{\partial}{\partial \theta}(F_2)\right). \tag{A.35}$$

In n-dimensional spherical coordinates the divergence is

$$\nabla_{r\theta_1 \dots \theta_n} \cdot F = \frac{\frac{\partial}{\partial r}\left(r^{n-1} \sin^{n-2}\theta_1 \dots \sin\theta_{n-2} F_1\right) + \dots + \frac{\partial}{\partial \theta_n}\left(r^{n-2} \sin^{n-3}\theta_1 \dots \sin\theta_{n-3} F_n\right)}{r^{n-1} \Pi_{i=1}^{n-2} \sin^{n-i-1}\theta_i} \tag{A.36}$$

A.4 Computing: $\nabla|\nabla u|$

In this section, we show that $\nabla|\nabla u| = \frac{1}{|\nabla u|} D^2 u \nabla u$, where $D^2 u$ is the Hessian of the function u. Note that $|\nabla u|$ is a scalar valued function. So to find its gradient we need to compute the first partials of $|\nabla u|$.

$$\begin{aligned}\frac{\partial}{\partial x_i}|\nabla u| &= \frac{\partial}{\partial x_i}\left(\sum_{j=1}^{n}\left(\frac{\partial u}{\partial x_j}\right)^2\right)^{1/2} \\ &= \frac{1}{2}\left(\sum_{j=1}^{n}\left(\frac{\partial u}{\partial x_j}\right)^2\right)^{-1/2}\sum_{j=1}^{n}\left(2\frac{\partial u}{\partial x_j}\frac{\partial^2 u}{\partial x_j \partial x_i}\right) \\ &= \frac{1}{|\nabla u|}\sum_{j=1}^{n}\left(\frac{\partial u}{\partial x_j}\frac{\partial^2 u}{\partial x_j \partial x_i}\right) \end{aligned} \tag{A.37}$$

Putting this together, we write the gradient of $|\nabla u|$ as

$$\nabla|\nabla u| = \frac{1}{|\nabla u|}\begin{pmatrix} \sum_{j=1}^{n}\left(\frac{\partial u}{\partial x_j}\frac{\partial^2 u}{\partial x_j \partial x_1}\right) \\ \vdots \\ \sum_{j=1}^{n}\left(\frac{\partial u}{\partial x_j}\frac{\partial^2 u}{\partial x_j \partial x_n}\right) \end{pmatrix}. \tag{A.38}$$

Notice, when writing out the sums above, we can see that this is really the same as a scalar multiple of the product of the Hessian and the gradient.

$$\nabla|\nabla u| = \frac{1}{|\nabla u|}\begin{pmatrix} \frac{\partial u}{\partial x_1}\frac{\partial^2 u}{\partial x_1^2} + \frac{\partial u}{\partial x_2}\frac{\partial^2 u}{\partial x_2 \partial x_1} + \ldots + \frac{\partial u}{\partial x_n}\frac{\partial^2 u}{\partial x_n \partial x_1} \\ \vdots \\ \frac{\partial u}{\partial x_1}\frac{\partial^2 u}{\partial x_1 \partial x_n} + \frac{\partial u}{\partial x_2}\frac{\partial^2 u}{\partial x_2 \partial x_n} + \ldots + \frac{\partial u}{\partial x_n}\frac{\partial^2 u}{\partial x_n^2} \end{pmatrix} = \frac{1}{|\nabla u|} D^2 u \nabla u.$$

Using the above, we can compute $\nabla|\nabla u|_\beta$, where the regularized gradient is given by

$$|\nabla u|_\beta = \sqrt{|\nabla u|^2 + \beta^2} \tag{A.39}$$

Again, we start with the partial derivatives of $|\nabla u|_\beta$.

$$\begin{aligned}\frac{\partial}{\partial x_i}|\nabla u|_\beta &= \frac{\partial}{\partial x_i}\left(\sum_{j=1}^{n}\left(\frac{\partial u}{\partial x_j}\right)^2+\beta^2\right)^{1/2} \\ &= \frac{1}{2}\left(\sum_{j=1}^{n}\left(\frac{\partial u}{\partial x_j}\right)^2+\beta^2\right)^{-1/2}\sum_{j=1}^{n}\left(2\frac{\partial u}{\partial x_j}\frac{\partial^2 u}{\partial x_j \partial x_i}\right) \\ &= \frac{1}{|\nabla u|_\beta}\sum_{j=1}^{n}\left(\frac{\partial u}{\partial x_j}\frac{\partial^2 u}{\partial x_j \partial x_i}\right) \end{aligned} \tag{A.40}$$

So,

$$\nabla|\nabla u|_\beta = \frac{1}{|\nabla u|_\beta}D^2 u \nabla u \tag{A.41}$$

BIBLIOGRAPHY

[1] William K Allard. Total variation regularization for image denoising, i. geometric theory. *SIAM Journal on Mathematical Analysis*, 39(4):1150–1190, 2007.

[2] William K Allard. Total variation regularization for image denoising, ii. examples. *SIAM Journal on Imaging Sciences*, 1(4):400–417, 2008.

[3] William K Allard. Total variation regularization for image denoising, iii. examples. *SIAM Journal on Imaging Sciences*, 2(2):532–568, 2009.

[4] S. Alliney. A Property of the Minimum Vectors of a Regularizing Functional Defined by Means of the Absolute Norm. *IEEE Trans. Signal Process.*, 45:913–917, 1997.

[5] G. B. Arfken and H. J. Weber. *Mathematical Methods For Physics*. Academic Press, London, fourth edition, 1995.

[6] T. Chan and Esedoḡlu. Aspects of Total Variation Regularized L^1 Function Approximation. *SIAM J. Appl. Math.*, 65(5):1817–1837, 2005.

[7] R. Chartrand and H. A. Van Dyke. private communication, 2010.

[8] Rick Chartrand. Nonconvex regularization for shape preservation. In *IEEE International Conference on Image Processing (ICIP)*, 2007.

[9] Francis H Clarke, Yuri S Ledyaev, Ronald J Stern, and Peter R Wolenski. *Nonsmooth analysis and control theory*, volume 178. Springer, 1997.

[10] Michael G Crandall. Viscosity solutions: a primer. In *Viscosity solutions and applicatons*, pages 1–43. Springer, 1997.

[11] B. Dacorogna. *Introduction to the Calculus of Variations*. Imperial College Press, London, 2004.

[12] B. Dacorogna. *Direct Methods in the Calculus of Variations, Second ed.* Springer, 2008.

[13] E. DiBenedetto. *Degenerate Parabolic Equations*. Springer-Verlag, New York, 1993.

[14] L. C. Evans. *Partial Differential Equations*. American Mathematical Society, Providence, 2002.

[15] L.C Evans and R.L. Gariepy. *Measure theory and fine properties of functions*. Studies in Advanced Mathematics, CRC Press, 1992.

[16] D. Hartenstine and Matthew Rudd. Asymptotic Statistical Characterizations of p-Harmonic Functions of Two Variables. *Rocky Mountain Journal of Mathematics*, 2011.

[17] R. Iagar and A. Sánchez. Radial equivalence and study of self-similarity for two very fast diffusion equations. *J. Math. Anal. Appl.*, 351:635–652, 2009.

[18] R. Iagar, A. Sánchez, and J.L. Vázquez. Radial equivalence for the two basic nonlinear degenerate diffusion equations. *J. Math. Pures Appl.*, 89(1):1–34, 2008.

[19] P. Juutinen, P. Lindqvist, and J. J. Manfredi. On the Equivalence of Viscosity Solutions and Weak Solutions for a Quasi-Linear Equation. *SIAM J. Math. Anal.*, 33(3):699–717, 2001.

[20] S.G. Krantz and H.R. Parks. *Geometric integration theory*. Birkhäuser Boston, 2008.

[21] H. Lebesgue. Sur le problème de Dirichlet. *Rend. Circ. Palermo.*, 27:371–402, 1907.

[22] J. Manfredi. Weakly Monotone Functions. *The J. of Geom. Anal..*, 4(2):393–402, 1994.

[23] G. Mostow. Quasiconformal mappings in n-space and the rigidity of hyperbolic space forms. *Publ. Math. Inst. Hautes Études Sci.*, 34:53–104, 1968.

[24] M Nikolova. Minimizers of cost-functions involving nonsmooth data-fidelity terms. *SIAM Journal on Numerical Analysis*, 40:965–994, 2003.

[25] M. B. Rudd and H. A. Van Dyke. Median Values, 1-Harmonic Functions, and Functions of Least Gradient. *Communications in Pure and Applied Analysis*, 12(2):711–719, March 2013.

[26] L. Rudin, S. Osher, and E. Fatemi. Nonlinear Total Variation Based Noise Removal Algorithms. *Physica D*, 60(1-4):259–268, November 1992.

[27] James Albert Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge university press, 1999.

[28] D. Strong and T. Chan. Edge-Preserving and Scale-Dependent Properties of Total Variation Regularization. *Inverse Problems*, 19, 2003.

[29] A. N. Tikhonov and V. Y. Arsenin. *Solutions of ill-posed problems*. Winston, 1977.

[30] Kevin R Vixie, Keith Clawson, Thomas J Asaki, Gary Sandine, Simon P Morgan, and Brandon Price. Multiscale flat norm signatures for shapes and images. *Applied Mathematical Sciences*, 4(14):667–680, 2010.

[31] S. K. Vodop'yanov and Gol'dshtein V. M. Quasiconformal mappings and spaces of functions with generalized first derivatives. *Siberian Math J.*, 17:399–411, 1976.