Quasiconvex Relaxations for l_0-Norm Optimization

Karthi Srinivasan, EE17B070

July 27, 2020

Reference:

An Approximate l_0-Norm Minimization Algorithm for Compressed Sensing [HM09]

Contributions:

Implementation of algorithm

Extension to image processing

Extension to other loss functions

Comparison with other methods

Contents

1 Introduction **3**

2 Compressive Sensing **4**
2.1 Compressible Signal 4
2.2 Stable Measurement Matrix 4

3 Quasiconvex Relaxation Functions **5**

4 Methods **7**
4.1 Fixed Point Iteration Algorithm 7
4.2 Projected Gradient Descent (PGD) 9
4.3 l_1 Relaxation 9
4.4 Additional Processing for Images 9

5 Visualizations **10**

6 Results **12**

7 Conclusion **14**

8 Appendices **15**
8.1 Improved Approximate L-Zero (IALZ) - MATLAB Code 15
8.2 Projected Gradient Descent - MATLAB Code 17

Bibliography **18**

1 Introduction

Finding sparse solutions to systems of linear equations is a problem that has been dealt with extensively, and has applications in various fields, including image processing, signal processing, machine learning, medical imaging, and more. The primary difficulty of solving these problems arises from the fact that this is a combinatorial problem, and even in the optimization formulation, the objective/loss function(l_0 norm) that is used is discontinuous at the origin, and constant everywhere else, thereby thwarting iterative methods that are usually employed to solve such problems exactly. The problem can be formulated quite simply as shown:

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & ||x||_0 \\ \text{subject to} \quad & Ax = b \end{aligned} \tag{1}$$

The matrix A is called the measurement matrix, b is the measured vector and x is the underlying event vector that needs to be estimated.
However, solving this exactly has been proven to be NP-hard and numerically unstable. Hence, practical methods usually use relaxed versions of the objective in place of the original so as to enable the use of conventional optimization methods. Conventional relaxations use convex functions as a replacement, such as the l_1 or the l_2 norms. Though these are convex, they do not approximate the original objective function closely, and their validity has been proved only in noiseless cases with certain constraints on the measurement matrix, A. [CRT06]
In other cases, it is merely a heuristic that seems to work well in many scenarios, despite these being outside the set for which proofs exist.
The main implicit restriction that has always remained is that the new relaxed loss function needs to be convex. However, this need not be the case as there are deterministic, efficient methods to solve non-convex problems also. The general formulation of the problem is:

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & \sum_i f(x_i) \qquad x_i : elements\ of\ x \\ \text{subject to} \quad & Ax = b, \end{aligned} \tag{2}$$

where $f(x)$ is the chosen relaxation function.
The chosen function needs to be an even function and non-negative with a unique minimum at zero, so as to approximate the original function.
This report explores quasiconvex relaxations to the l_0 problem and iterative methods to solve them.

2 Compressive Sensing

The problem of recovering sparse signals from measurements at sampling rates much below the Nyquist rate is broadly known as compressive sensing. For discrete vectors, this translates to recovering the signal from fewer observations than the size of the vector itself. A crucial step in the process is the design of the measurement matrix itself, as its structure makes the difference between being and not being able to recover the sparse signal.

2.1 Compressible Signal

Consider a real-valued finite-length, one-dimensional discrete-time signal x of size N. If there exists a basis Ψ such that, in that basis, x is known to have at most K entries non-zero, then such a signal is known as a K-sparse signal. The particular case of interest here is when $K \ll N$. The signal x is compressible if its representation in some basis has just a few large coefficients, with the majority being either very small or zero.

2.2 Stable Measurement Matrix

A stable measurement matrix is a matrix A which is fixed (does not change with the signal) such that all the information in the original signal x is recoverable from the reduced measured vector $b = Ax$. For a measurement matrix A of size M×N (K≤ M<N), some properties need to be satisfied for it to be a stable measurement matrix i.e to allow proper reconstruction [CRT06]. They are:

1. Restricted Isometry: For any $3K$-sparse vector x, and for some $\epsilon > 0$,

$$1 - \epsilon \leq \frac{||Ax||_2}{||x||_2} \leq 1 + \epsilon \tag{3}$$

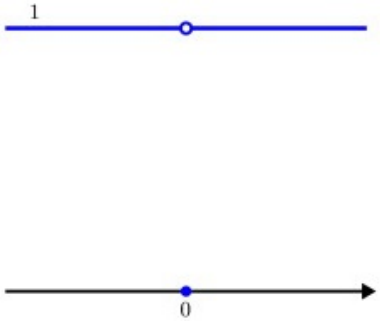
2. Incoherence: For the basis matrix (matrix with basis vectors ψ_i as columns) Ψ, the rows of A cannot sparsely represent the columns of Ψ and vice versa.

Direct construction of such a measurement matrix A requires verifying all possible combinations of K non-zero entries in the vector x of length N. However, both properties can be achieved with high probability simply by selecting A as a random matrix [Bar07]. In the case considered here, A was chosen such that A_{ij} are all independently and identically distributed (iid) random variables from a Gaussian pdf with mean zero and variance $1/N$. Such a matrix A has two useful properties:

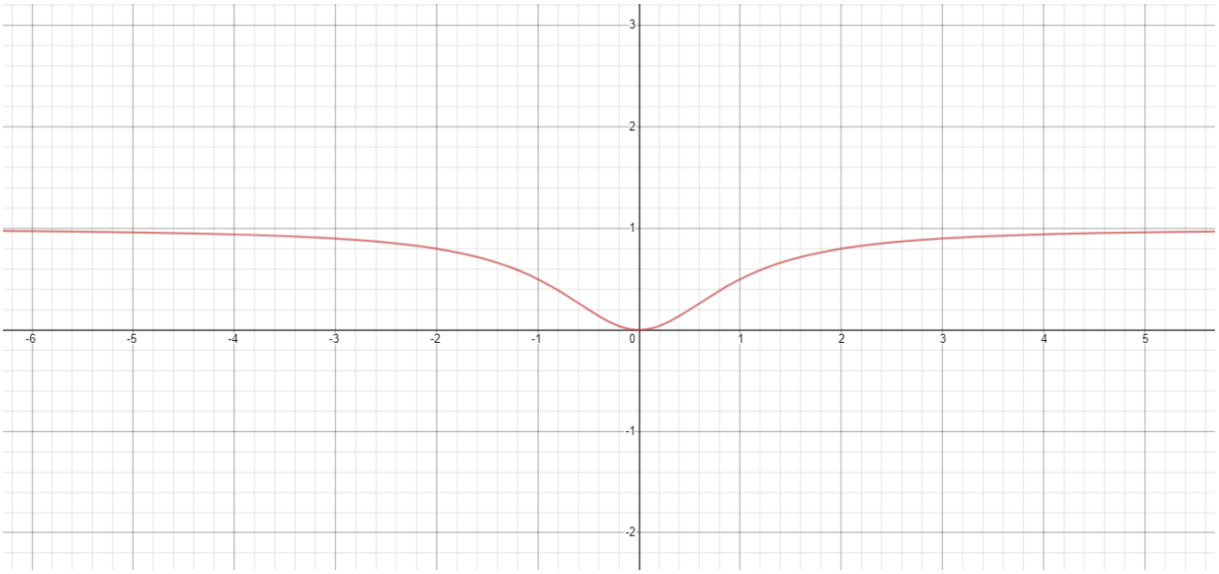
1. A is incoherent with the Cartesian basis matrix ($\Psi = I$) with high probability.
2. A is universal in the sense that $\tilde{A} = A\Psi$ will be iid Gaussian and thus will have the required properties regardless of the choice of orthonormal basis Ψ. In particular, the property will hold even if Ψ is the discrete cosine basis matrix.

3 Quasiconvex Relaxation Functions

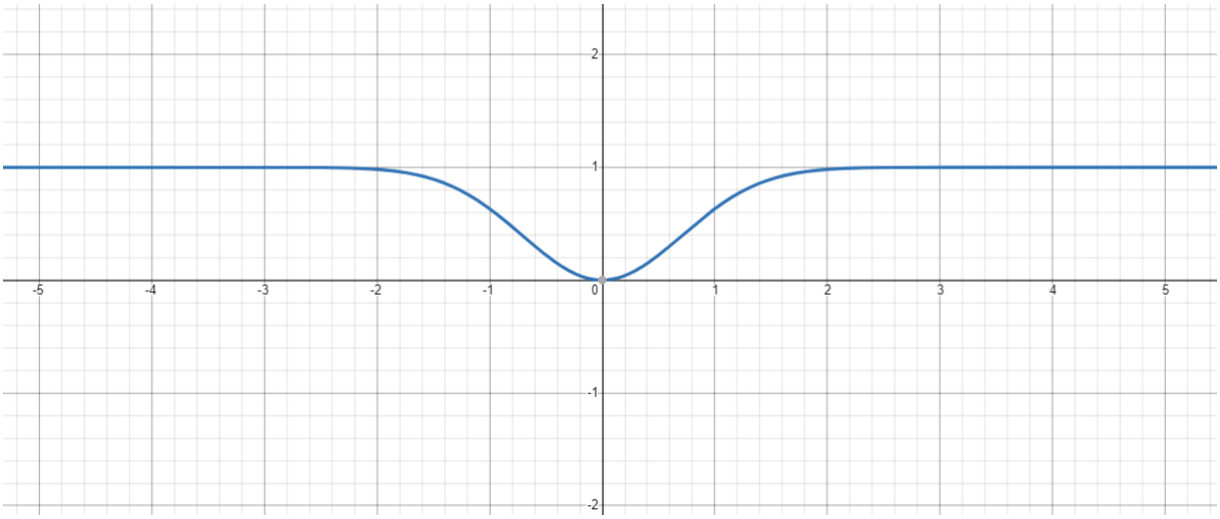
The l_0 norm function is shown below:



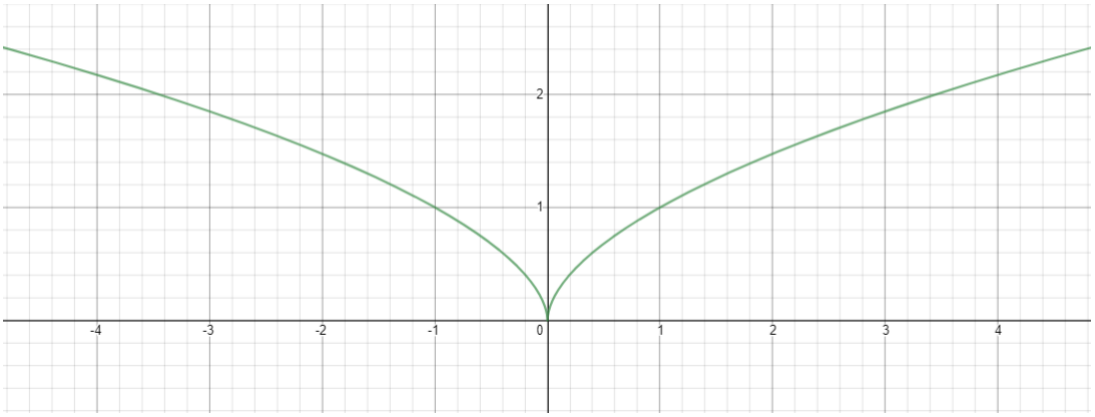
The quasiconvex functions that were considered were:



$$f(x) = 1 - \frac{s^2}{x^2 + s^2} \tag{4}$$



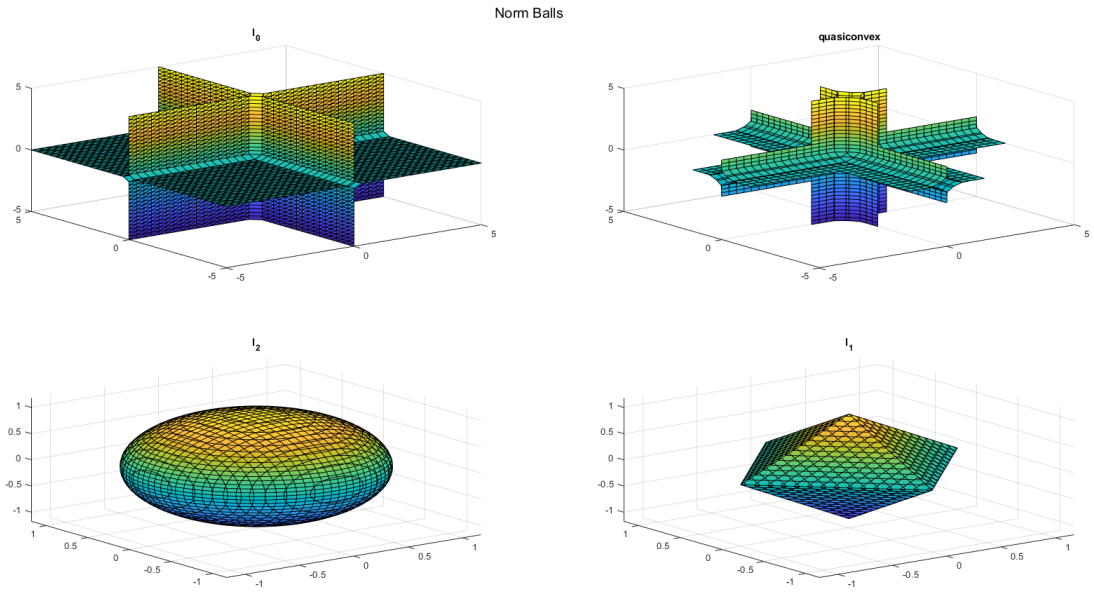
$$f(x) = 1 - e^{-\frac{x^2}{2s^2}} \tag{5}$$



$$f(x) = \lambda x^p, 0 < p < 1 \tag{6}$$

As can be seen, for varying values of the parameters, these functions closely and continuously approximate the l_0 norm function itself. Further, the first two functions are continuously differentiable everywhere.

The plot below shows the norm balls (loose terminology, since a few of them are not norms in the strict sense) of the different functions used. It is easy to see why one would expect a good approximation to the l_0 norm from quasiconvex functions.



4 Methods

4.1 Fixed Point Iteration Algorithm

Consider the minimization problem:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \qquad \sum_i f(x_i) \qquad x_i : elements\ of\ x \tag{7}$$

$$\text{subject to} \qquad Ax = b$$

Let

$$F(x) = \sum_i f(x_i) \tag{8}$$

The Lagrangian for the problem (27) is:

$$L(x, \mu) = F(x) + \mu(Ax - b) \tag{9}$$

where μ is the vector of Lagrange multipliers. Computing the stationary point of L:

$$\frac{\partial L(x_*, \mu_*)}{\partial x} = \frac{\partial F(x_*)}{\partial x} + A^T \mu_* = 0 \tag{10}$$

Further, rewriting the gradient of $F(x)$ in a matrix product form:

$$\frac{\partial F(x)}{\partial x} = D(x)x \tag{11}$$

where

$$D(x) = \frac{1}{\sigma^2} diag(e^{-\frac{x_1^2}{2s^2}}, ...e^{-\frac{x_n^2}{2s^2}}) \tag{12}$$

$$D(x) = 2s^2 diag(\frac{1}{(x_1^2 + s^2)^2}, ...\frac{1}{(x_n^2 + s^2)^2}) \tag{13}$$

depending on the choice of f. Notice that these diagonal matrices always have positive values on the diagonal, for all real values of x, and are hence positive definite, and always invertible. Substituting in (10), gives:

$$x_* = -D^{-1}(x_*)A^T \mu_* \tag{14}$$

Solving this for μ_* and substituting back in the constraint, we get:

$$\mu_* = -(AD^{-1}(x_*)A^T)^{-1} b \tag{15}$$

Substituting this back in (10), we obtain the fixed point equation for x_*:

$$x_* = D^{-1}(x_*)A^T(AD^{-1}(x_*)A^T)^{-1} b \tag{16}$$

Note that this equation is highly non-linear and hence cannot be solved exactly. An iterative method to solve (17) was proposed in [HM09]. Let

$$g(x) = D^{-1}(x)A^T(AD^{-1}(x)A^T)^{-1} b \tag{17}$$

Notice that $Ag(x) = b\ \forall x$. The central argument revolves around the fact that the gradient of F and the line joining x and $g(x)$ are always in roughly opposite direction i.e. they have negative inner product always. Hence, by changing the value of x along the gradient of F, the gap between x and $g(x)$ continually reduces,thereby approaching the optimal point, which has $g(x) = x$.
At any point which is not the optimal point i.e. $g(x) \neq x$, consider the dot product:

$$[\frac{\partial F(x)}{\partial x}]^T (g(x) - x) \tag{18}$$

$$= [D(x)x]^T[D^{-1}(x)A^T(AD^{-1}(x)A^T)^{-1}b - x]$$
$$= x^T[A^T(AD^{-1}(x)A^T)^{-1}A - D]x$$
$$= -[D^{0.5}(x)x]^T[I - D^{-0.5}(x)A^T(AD^{-1}(x)A^T)^{-1}AD^{-0.5}(x)][D^{0.5}(x)x] \quad (19)$$

Consider the matrix in the middle, in (19):

$$P = [I - D^{-0.5}(x)A^T(AD^{-1}(x)A^T)^{-1}AD^{-0.5}(x)] \quad (20)$$

Notice that the second term in P is the projection onto the range of $AD^{-0.5}(x)$. Hence, the entire matrix P, which takes a vector x and removes the projection onto the range of $AD^{-0.5}(x)$ must be the projection onto the null-space of $AD^{-0.5}(x)$. This is indeed true as $P^2 = P$ and $Px = x$ if and only if x is in the nullspace of $AD^{-0.5}(x)$. Therefore:

$$[D^{0.5}(x)x]^T P[D^{0.5}(x)x] \geq 0 \quad (21)$$

with equality only when $D^{0.5}(x)x$ is of the form $D^{-0.5}(x)A^T z$ for some vector z.

$$D^{0.5}(x)x = D^{-0.5}(x)A^T z \quad (22)$$
$$\implies \frac{\partial F(x)}{\partial x} = A^T z$$
$$\implies z^T A(g(x) - x) = 0$$
$$\implies x = g(x)$$

but our original assumption was that $x \neq g(x)$. Also, the gradient of F is zero only at the origin.

$$\implies [\frac{\partial F(x)}{\partial x}]^T(g(x) - x) < 0 \quad (23)$$

Hence the inner product between $g(x) - x$ and the gradient of $F(x)$ is always strictly negative at non-optimal points. This implies that one can always choose some point along the line joining x and $g(x)$, and obtain a strict decrease in objective function value. Hence, there exists a θ such that $0 < \theta \leq 1$ and:

$$F\{\theta g(x) + (1 - \theta)x\} < F(x) \quad (24)$$

If the iteration is started at a feasible point, then $Ax_0 = b$. Further, $Ag(x) = b\ \forall x$, hence $\theta g(x) + (1 - \theta)x$ will also be a feasible point, for all θ. Hence, there is no additional cost incurred in maintaining feasibility as part of the algorithm.

The algorithm that is based on this result is shown below:

1. Compute a feasible starting point. Typically $x_0 = A^T(AA^T)^{-1}b$
2. Set $\sigma = 1$ and choose $\rho > 1, \gamma \in (0, 1)$
3. Repeat:
   (a) Set $\theta = 1$.
   (b) Compute $x_{i+1} = \theta g(x_i) + (1 - \theta)x_i$
   (c) Backtrack: If $F(x_{i+1}) > F(x_i)$, set $\theta = \gamma\theta$ and redo (a),(b)
   (d) If $\tau = ||x_{i+1} - x_i|| < \frac{\sigma}{\rho}$, set $\sigma = \frac{\sigma}{\rho}$.
4. until $\tau \leq 10^{-8}$

Step (d) dynamically scales the width parameter of the objective function to enforce scarcity across different scales. As values get closer to zero, it makes the function steeper to push them even closer.

4.2 Projected Gradient Descent (PGD)

Projected gradient descent is essentially a generalization of the gradient descent method, in that it can deal with constrained optimization [FNW07]. Normal gradient descent works by moving along the negative gradient direction to reach the minimum point of the function. This only works when there are no constraints on the values that the variable can take, save the domain restriction of the function itself, which is implicit in the problem.
Projected gradient descent generalizes this by maintaining feasibility at every iteration. PGD starts with a feasible point, and proceeds from there. Essentially, once the local direction of the gradient has been found, the component of this direction(projection) that lies in the feasible set of the constraints is recovered and then the step size is calculated along this direction, instead of the original gradient itself.
The basic algorithm for (25) is shown below:

1. Compute a feasible starting point. Typically $x_0 = A^T(AA^T)^{-1}b$
2. Pick a step size: t, threshold: s
3. Repeat:
4. Compute $y_i = x - t\nabla f(x_i)$
5. Compute projection of gradient:
   (a) $x_{i+1} = argmin||x - y_i||_2$ such that $Ax = b$
6. until $\nabla f(x_i) \leq s$

More sophisticated additions such as dynamically changing the step size etc. can be done that will improve the convergence rate of the algorithm. PGD was also used to solve the quasiconvex problem mentioned earlier, to compare convergence rates.

4.3 l_1 Relaxation

This is the usual convex relaxation method that is used to solve the problem. The formulation is as follows:

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & ||x||_1 \\ \text{subject to} \quad & Ax = b \end{aligned} \tag{25}$$

This was solved using CVX, as usual.

4.4 Additional Processing for Images

The main results in this report were obtained through processing images. The key point to note when dealing with images is that they are usually sparse in the frequency (DCT) domain, not in the pixel value domain. Due to this, the random measurement matrix that is generated is post-multiplied by the DCT matrix so that the final measured vector also contains information represented in the frequency space. Essentially, the overall measurement matrix is:

$$\tilde{A} = A\Phi \tag{26}$$

where $A \in \mathbb{R}^{m \times n}$ is the original randomly generated measurement matrix, and $\Phi \in \mathbb{R}^{n \times n}$ is the n-dimensional DCT matrix.

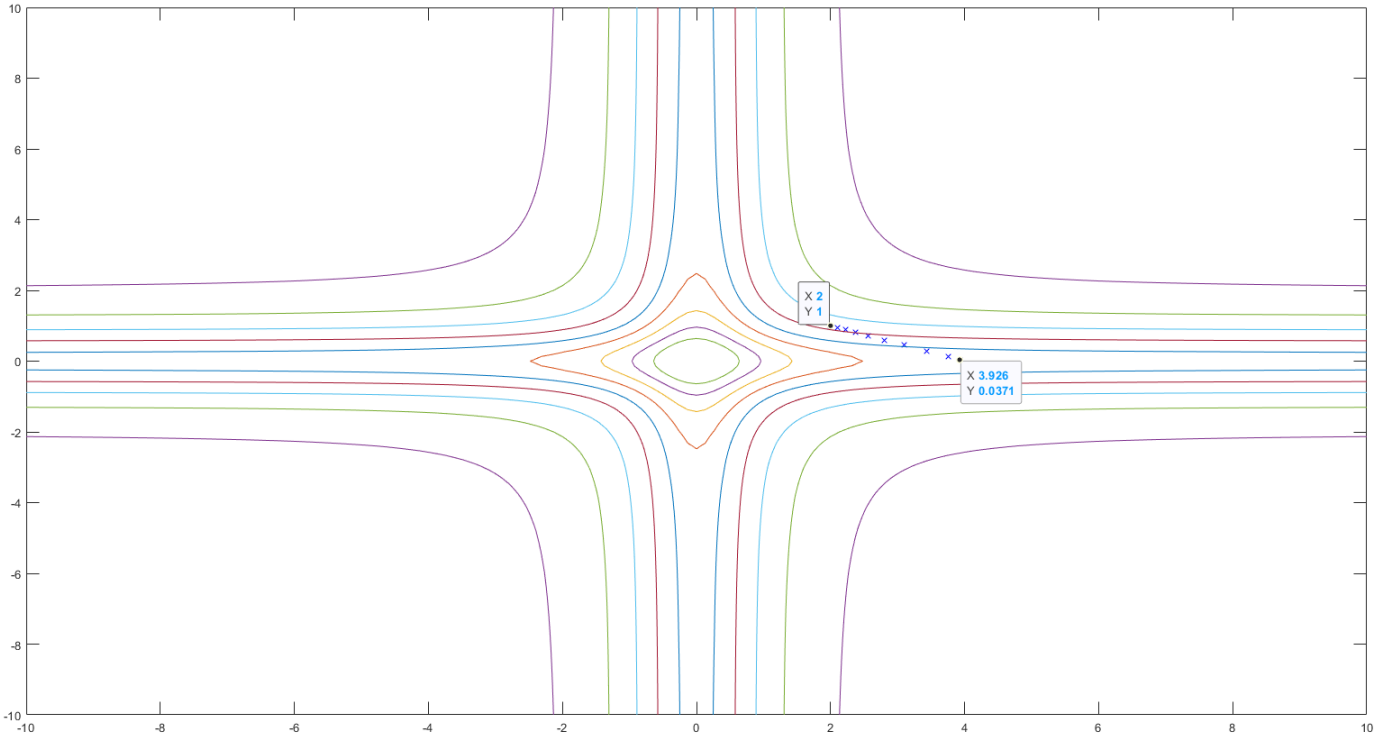
5 Visualizations

This section illustrates the iterations and the motion of the solution point, under the quasiconvex optimization algorithm (IALZ). A size 2 vector was used for this toy problem. The exact problem is:

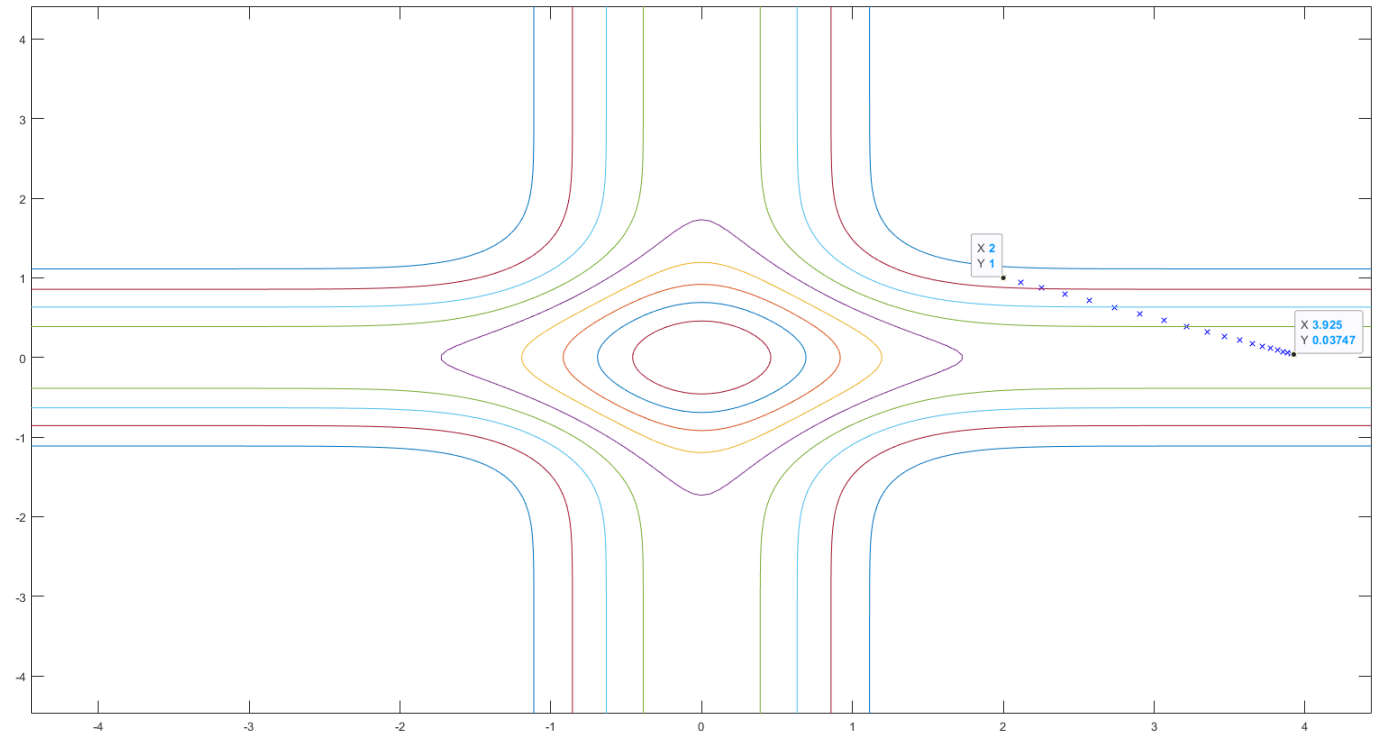
$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \qquad \sum_i f(x_i) \qquad x_i : elements\ of\ x \tag{27}$$

$$\text{subject to} \qquad [1\ 2][x_1\ x_2]^T = 4$$

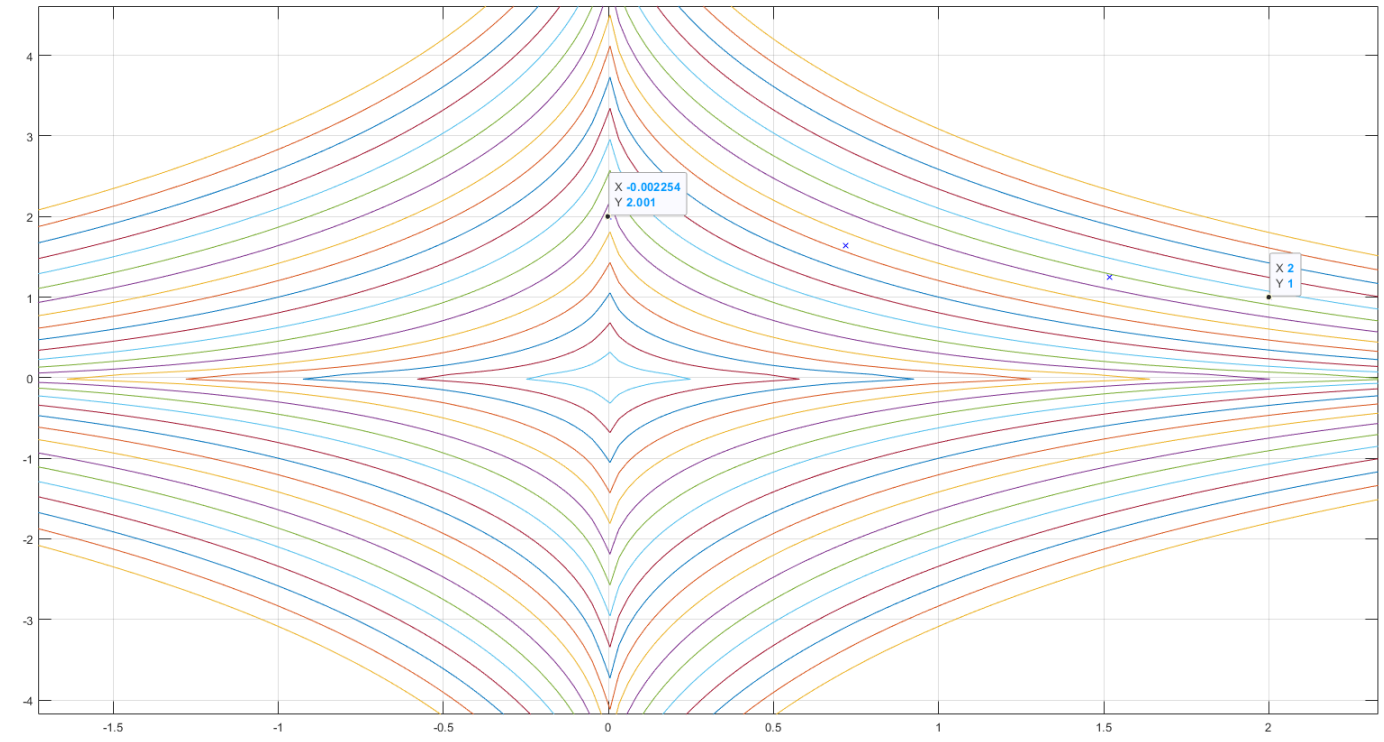
for different quasiconvex objective functions. The contours show equipotential surfaces of the functions. The main point to note is that troughs in the objective function value occur along the axes, similar to the l_0 norm, and solution points tend to move towards these troughs. Also, there isn't a unique minimum solution [(4,0) and (0,2)] to the original l_0 problem, as can be seen, and different objective functions also favor some over the other. The initial feasible point from where iteration was started was (2,1).



$$f(x) = 1 - \frac{s^2}{x^2 + s^2}$$



$$f(x) = 1 - e^{-\frac{x^2}{2s^2}}$$

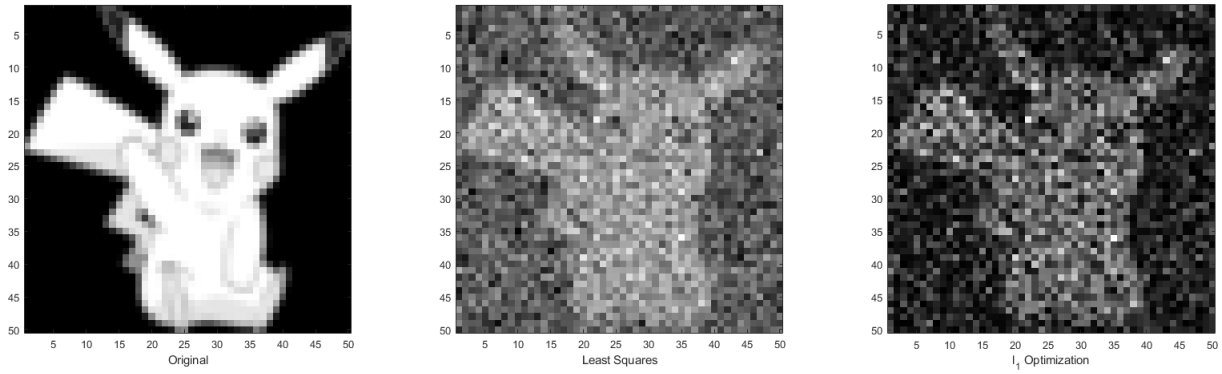


$$f(x) = \lambda x^{0.5}$$

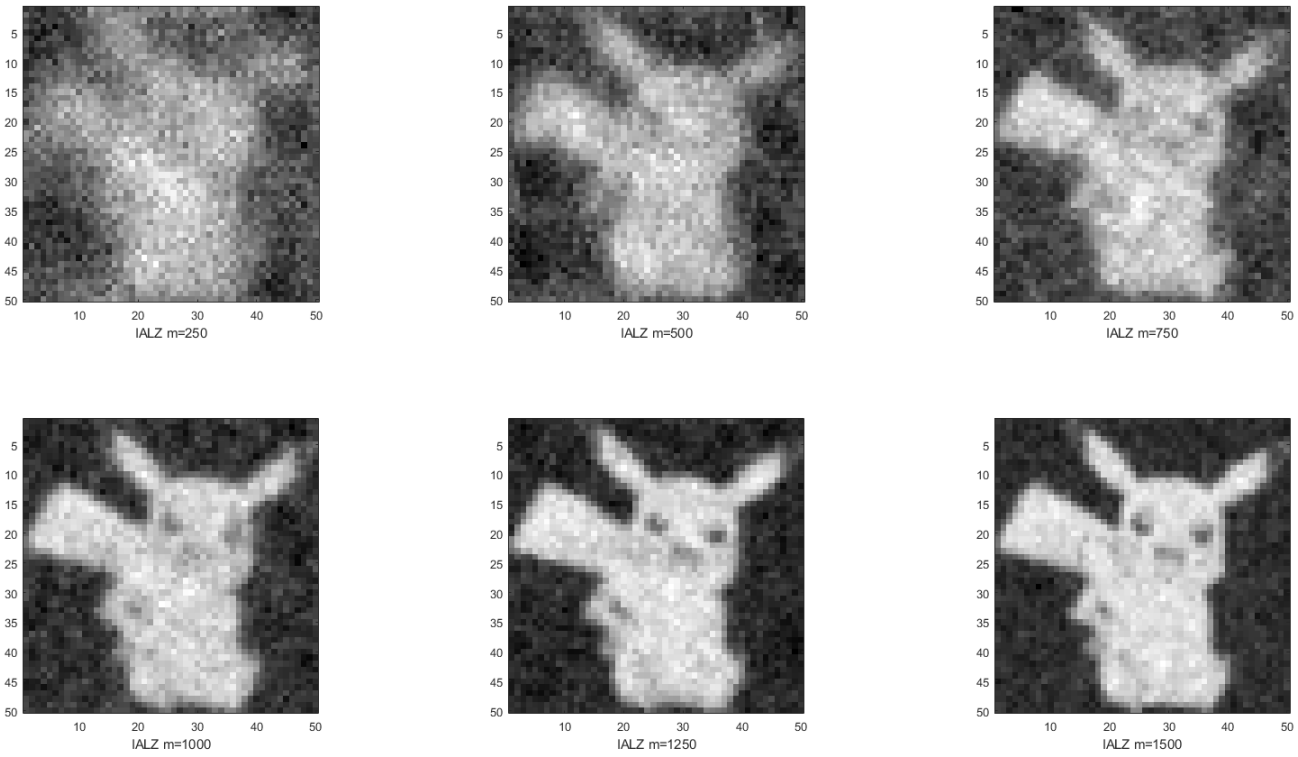
The square-root function tends to pick (0,2) over (4,0), in contrast to the other functions.

6 Results

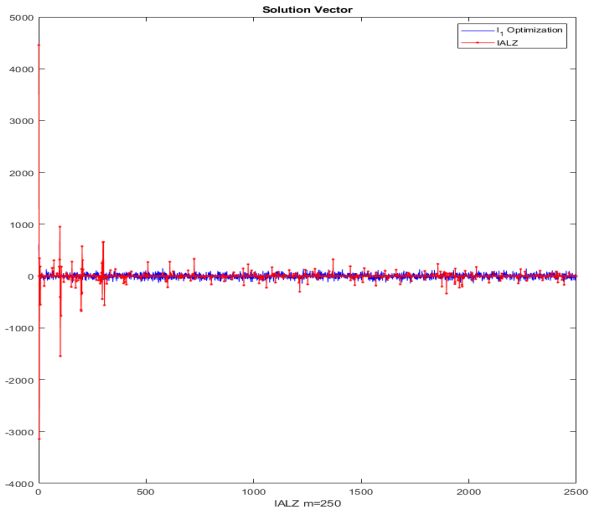
This section details the results obtained by using the different norms on the same problem. A 50×50 image was flattened into a 2500×1 vector. The vector was measured using a cascade of an m×2500 measurement matrix and a 2500×2500 DCT matrix. The image was then reconstructed using only the information from the m×1 vector. The following images show reconstructed results for different values of m.

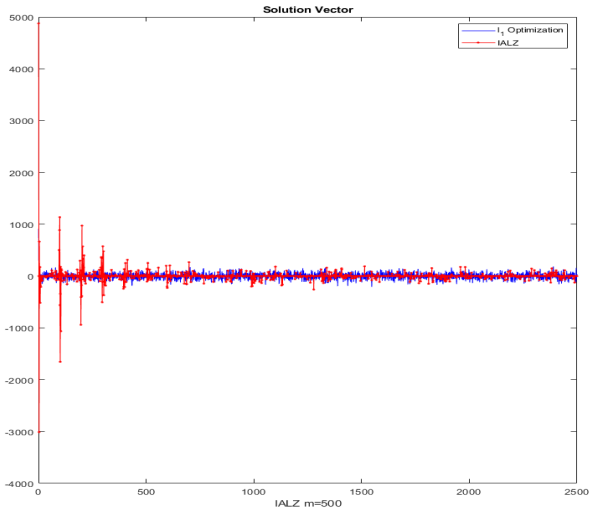


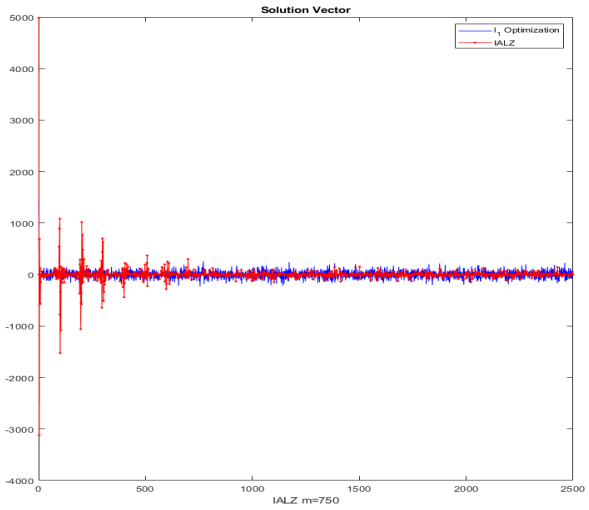
For these two methods (l_2 and l_1), a value of 750 was used for m. l_1 has darker blacks as higher values are penalized more than in l_2, close to the origin.

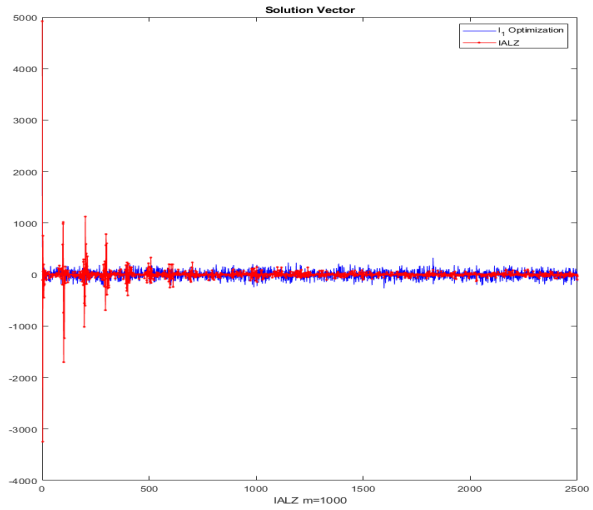


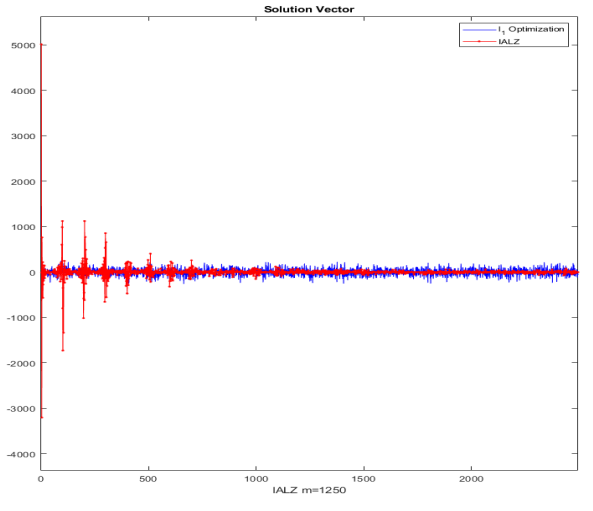
Reconstructed images for varying values of m, using the IALZ algorithm. Even for m=750, the result is considerably better than the other two methods. For m=1500, the algorithm recovers the image almost perfectly.

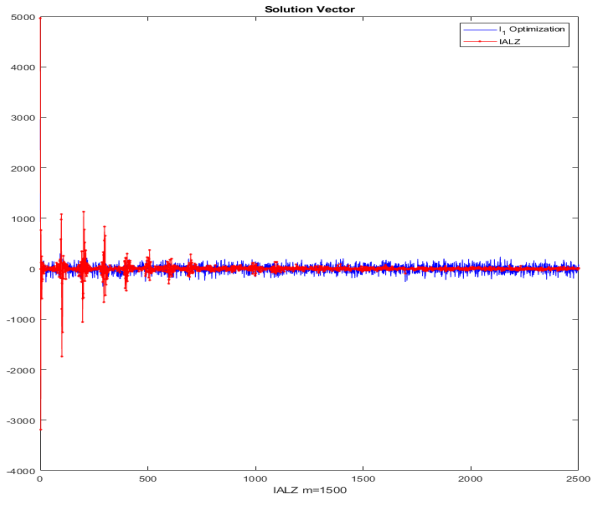




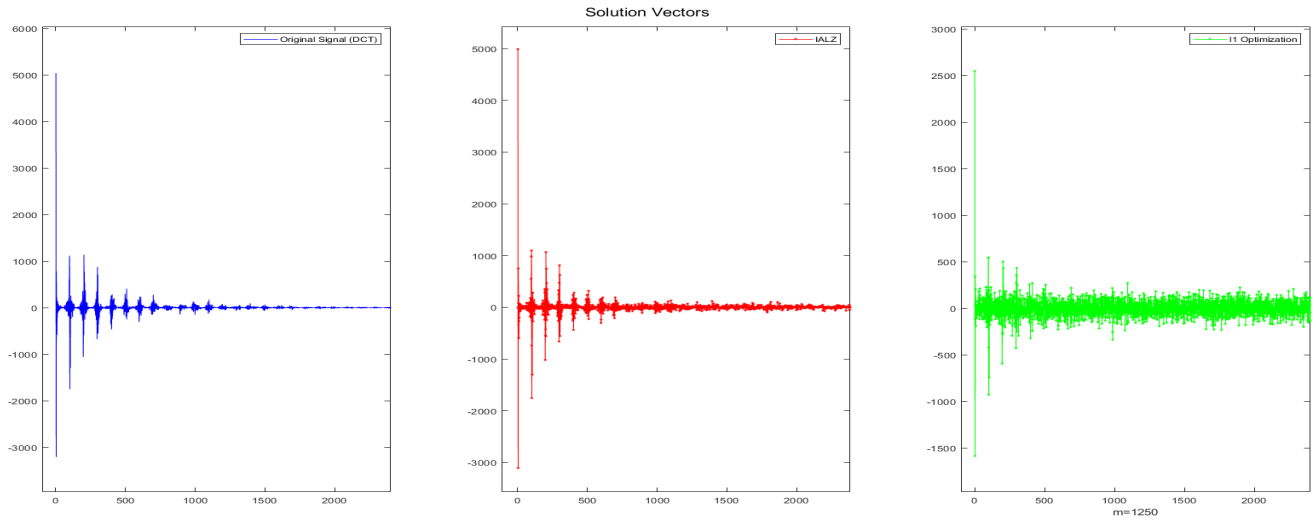




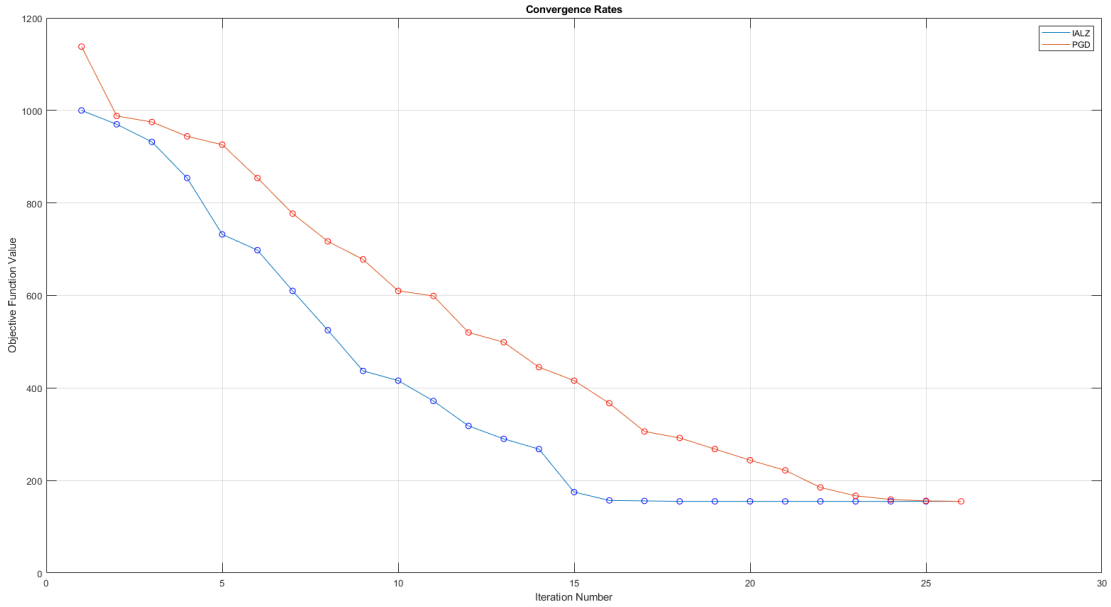




Plot of the flattened 2500×1 vector form of the image, for different values of m, in the DCT basis. As can be seen, the image is quite sparse in this basis. Also, the quasiconvex objective has lesser noise at high frequencies, due to dynamically changing the scale of the objective function to push values closer to zero.



Comparison of input and outputs due to different methods.



Comparison of convergence rates of the two methods employed. They have similar rates but IALZ performs slightly better in this case. However, the results are too close to make a hard conclusion.

7 Conclusion

The performance of quasiconvex objective functions as an effective relaxation for l_0 norm minimization problems was explored, and compared with existing convex techniques. In particular, the application of these methods to compressive sensing for images was discussed. For moderate sparsity levels, some quasiconvex functions outperform l_2-norm and l_1-norm methods, but take longer to run due to the lack of efficient solvers. This problem can be solved by incorporating the algorithm into the Disciplined Quasi-Convex Programming(DQCP) framework in CVX [AB20]. The effect of dynamic loss function scaling was also explored and was deemed to be a useful technique for inducing sparsity.

8 Appendices

8.1 Improved Approximate L-Zero (IALZ) - MATLAB Code

```
%IALZ

%Use 50x50 image for full size
P = imread('pikachu1.png');
%P=imbinarize(P);
P = P([1:50],[1:50]);
Im = double(P(:));
Im1=Im+sqrt(50)*randn(size(Im)); %Add noise
n = length(Im);

%Controls level of compression
m = 1500;

A = randn(m,n);

Theta = zeros(m,n);
for ii = 1:n
    %ii
    ek = zeros(1,n);
    ek(ii) = 1;
    psi = idct(ek)';
    Theta(:,ii) = A*psi;
end

x = zeros(n,1000);

b=A*Im1;
x(:,1)=pinv(Theta)*b;

i=1;

%IALZ Parameters
sig=0.1;
rho=10;
gamma=0.5;
tau=1;

while(tau>1e-8)
    lambda=1;
    x(:,i+1)=lambda*g(x(:,i),sig,Theta,b,n)+(1-lambda)*x(:,i);
    while(sum(f(x(:,i+1),sig))<sum(f(x(:,i),sig)))
        lambda=lambda*gamma;
        x(:,i+1)=lambda*g(x(:,i),sig,Theta,b,n)+(1-lambda)*x(:,i);
    end
    tau=norm(x(:,i+1)-x(:,i));
    disp(tau);
    if tau<sig/rho
         sig=sig/rho;
    end
```

```
    i=i+1;
end

x2 = zeros(n,1);
for ii = 1:n
    %ii
    ek = zeros(1,n);
    ek(ii) = 1;
    psi = idct(ek)';
    x2 = x2+psi*x(ii,i);
end

subplot(2,3,1), imagesc(reshape(P,50,50)), xlabel('Original'), axis image
subplot(2,3,2), imagesc(reshape(Im1,50,50)), xlabel('Noisy'), axis image
subplot(2,3,3), imagesc(reshape(x2(:),50,50)), xlabel('Reconstructed'), axis image
colormap gray

%
% plot(x(1,1),x(2,1),'rx')
% hold on
% for j = 2:i
%     plot(x(1,j),x(2,j),'rx')
%     hold on
% end
%
% disp(x(:,1:i));

function r = f(x,sig)
    r=sig^2/x.^2+sig^2;
end

function r = f2(x,sig)
    r=abs(x.^0.5);
end

function r = f1(x,sig)
    r=exp((-x.^2)/2*(sig^2));
end

function r = g(x,sig,Theta,b,n)
    Winv=eye(n);
    i=1;
    while(i<n+1)
        Winv(i,i)=1/f(x(i),sig)^2;
        i=i+1;
    end
    r=Winv*transpose(Theta)*((Theta*Winv*transpose(Theta))\b);
end
```

8.2 Projected Gradient Descent - MATLAB Code

```
x = zeros(2,1000);
A=[1 2];
b=4;
x0 = pinv(A)*b; % start point
x(:,1)=x0;
i=1;
gamma=0.5;
lambda=zeros(1,1000);
lambda(1)=gamma;
c=2;
d=[0 0]; % step
g=[0 0]; % pre-grad step
%while(norm(gradf(x(:,i)))>0.1)
while(i<10)
    g=x(:,i)-gradf(x(:,i));
    cvx_begin
        variables y(2)
        minimize 0.5*norm(y-g)
        subject to
            A*y==b;
    cvx_end

    d=y-x(:,i);
    %Find lambda
    j=1;
    while(j<100)
        if f(x(:,i)+(gamma^j)*d)-f(x(:,i))<=c*(gamma^j)*dot(gradf(x(:,i)),d)
           lambda(i)=gamma^j;
           disp(lambda(i));
           j=100;
        end
        j=j+1;
    end
    x(:,i+1)=x(:,i)+lambda(i)*d;
    i=i+1;
end

for j = 1:i
    plot(x(1,j),x(2,j),'bx')
    hold on
end

function y = f(x)
    y=sum(x.^2/(x.^2+0.5));
end

function grady = gradf(x)
    grady = 2*x./(x.^2+0.5).^2;
end
```

Bibliography

[CRT06] Emmanuel J. Candès, Justin K. Romberg, and Terence Tao. "Stable signal recovery from incomplete and inaccurate measurements". In: *Communications on Pure and Applied Mathematics* 59.8 (2006), pp. 1207–1223. DOI: 10.1002/cpa.20124. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.20124. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.20124.

[Bar07] R. G. Baraniuk. "Compressive Sensing [Lecture Notes]". In: *IEEE Signal Processing Magazine* 24.4 (2007), pp. 118–121.

[FNW07] Mário AT Figueiredo, Robert D Nowak, and Stephen J Wright. "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems". In: *IEEE Journal of selected topics in signal processing* 1.4 (2007), pp. 586–597.

[HM09] M. Hyder and K. Mahata. "An approximate L0 norm minimization algorithm for compressed sensing". In: *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. 2009, pp. 3365–3368.

[AB20] Akshay Agrawal and Stephen Boyd. "Disciplined quasiconvex programming". In: *Optimization Letters* (2020), pp. 1–15.