Sparse Coding

Acknowledgement: The slides adapted from the ones prepared by M. Elad of the Technion - Israel Institute of Technology (general discussion) and L. Xu et al. of the Chinese University of Hong Kong (L0-smoothing)

Erkut Erdem
Hacettepe University
Computer Vision Lab (HUCVL)
Noise Removal?

• **Important**: (i) Practical application; (ii) A convenient platform (being the simplest inverse problem) for testing basic ideas in image processing, and then generalizing to more complex problems.

• **Many Considered Directions**: Partial differential equations, Statistical estimators, Adaptive filters, Inverse problems & regularization, Wavelets, Example-based techniques, **Sparse representations**, …
Denoising By Energy Minimization

Many of the proposed image denoising algorithms are related to the minimization of an energy function of the form

\[
f(x) = \frac{1}{2} \|x - y\|^2_2 + G(x)
\]

- \(y\): Given measurements
- \(x\): Unknown to be recovered

- This is in-fact a Bayesian point of view, adopting the Maximum-A-posteriori Probability (MAP) estimation.
- Clearly, the wisdom in such an approach is within the choice of the prior – **modeling the images** of interest.
The Evolution of $G(x)$

During the past several decades we have made all sort of guesses about the prior $G(x)$ for images:

- $G(x) = \lambda \|x\|^2_2$
- $G(x) = \lambda \|Lx\|^2_2$
- $G(x) = \lambda \|Lx\|^2_W$
- $G(x) = \lambda \rho \{Lx\}$
- $G(x) = \lambda \|\nabla x\|_1$
- $G(x) = \lambda \|Wx\|_1$
- $G(x) = \lambda \|\alpha\|^6_0$

for $x = D\alpha$

• Hidden Markov Models,
• Compression algorithms as priors,
• …
Sparse Modeling of Signals

- Every column in $D$ (dictionary) is a prototype signal (atom).
- The vector $\alpha$ is generated randomly with few (say $L$) non-zeros at random locations and with random values.
- We shall refer to this model as Sparseland.
Interesting Model:

• **Simple**: Every generated signal is built as a linear combination of few atoms from our dictionary $D$.

• **Rich**: A general model: the obtained signals are a union of many low-dimensional Gaussians.

• **Familiar**: We have been using this model in other context for a while now (wavelet, JPEG, …).
Sparse & Redundant Rep. Modeling?

Our signal model is thus: $x = D\alpha$ where $\alpha$ is sparse.
Sparse & Redundant Rep. Modeling?

Our signal model is thus: $\mathbf{x} = \mathbf{D}\alpha$ where $\alpha$ is sparse.
As $p \to 0$ we get a count of the non-zeros in the vector $\|\alpha\|_0$

Our signal model is thus: $x = D\alpha$ where $\|\alpha\|_0 \leq L$
Back to Our MAP Energy Function

- \( L_0 \) norm effectively counts the number of non-zeros in \( \alpha \).

- The vector \( \alpha \) is the representation (sparse/redundant) of the desired signal \( x \).

- The core idea: while few (\( L \) out of \( K \)) atoms can be merged to form the true signal, the noise cannot be fitted well. Thus, we obtain an effective projection of the noise onto a very low-dimensional space, thus getting denoising effect.

\[
\frac{1}{2} \| x - y \|_2^2
\]
Wait! There are Some Issues

- **Numerical Problems:** How should we solve or approximate the solution of the problem

\[
\min_{\alpha} \| D\alpha - y \|_2^2 \quad \text{s.t.} \quad \| \alpha \|_0 \leq L
\]

or

\[
\min_{\alpha} \| \alpha \|_0 \quad \text{s.t.} \quad \| D\alpha - y \|_2^2 \leq \varepsilon^2
\]

or

\[
\min_{\alpha} \lambda \| \alpha \|_0 + \| D\alpha - y \|_2^2
\]

- **Theoretical Problems:** Is there a unique sparse representation? If we are to approximate the solution somehow, how close will we get?

- **Practical Problems:** What dictionary \( D \) should we use, such that all this leads to effective denoising? Will all this work in applications?
To Summarize So Far …

Image denoising (and many other problems in image processing) requires a model for the desired image.

Use a model for signals/images based on sparse and redundant representations.

What can we do?

There are some issues:
1. Theoretical
2. How to approximate?
3. What about D?

Great! No?
Let's Start with the Noiseless Problem

Suppose we build a signal by the relation

$$D\alpha = x$$

We aim to find the signal's representation:

$$\hat{\alpha} = \text{ArgMin} \|\alpha\|_0 \quad \text{s.t.} \quad x = D\alpha$$

Why should we necessarily get $$\hat{\alpha} = \alpha$$?

It might happen that eventually $$\|\hat{\alpha}\|_0 < \|\alpha\|_0$$.
Matrix “Spark”

Definition: Given a matrix $D$, $\sigma = \text{Spark}\{D\}$ is the smallest number of columns that are linearly dependent.

Example:

\[
\begin{bmatrix}
1 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
\end{bmatrix}
\]

- Rank = 4
- Spark = 3

* In tensor decomposition, Kruskal defined something similar already in 1989.

Donoho & E. (‘02)
Uniqueness Rule

Suppose this problem has been solved somehow

$$\hat{\alpha} = \text{ArgMin} \|\alpha\|_0 \quad \text{s.t.} \quad x = D\alpha$$

If we found a representation that satisfy

$$\|\hat{\alpha}\|_0 < \frac{\sigma}{2}$$

Then necessarily it is unique (the sparsest).

This result implies that if $M$ generates signals using “sparse enough” $\alpha$, the solution of the above will find it exactly.

Donoho & E. (‘02)
Our Goal

This is a combinatorial problem, proven to be NP-Hard!

Here is a recipe for solving this problem:

\[
\min_{\alpha} \left\| \alpha \right\|_0^0 \quad \text{s.t.} \quad \left\| \mathbf{D}\alpha - y \right\|_2^2 \leq \varepsilon^2
\]

Set \( L=1 \)

Gather all the supports \( \{S_i\}_i \) of cardinality \( L \)

Solve the LS problem:

\[
\min_{\alpha} \left\| \mathbf{D}\alpha - y \right\|_2^2 \quad \text{s.t.} \quad \text{supp}(\alpha) = S_i
\]

for each support

Set \( L=L+1 \)

There are \( \binom{K}{L} \) such supports

LS error \( \leq \varepsilon^2 \) ?

No

Yes

Done

Assume: \( K=1000, \ L=10 \) (known!), 1 nano-sec per each LS

We shall need \( \sim 8e+6 \) years to solve this problem !!!!!
Let's approximate

\[
\min_{\alpha} \| \alpha \|_0^0 \quad \text{s.t.} \quad \| D\alpha - y \|_2^2 \leq \varepsilon^2
\]

Relaxation methods

Smooth the $L_0$ and use continuous optimization techniques

Greedy methods

Build the solution one non-zero element at a time
Relaxation – The Basis Pursuit (BP)

Instead of solving

\[
\min_{\alpha} \|\alpha\|_0 \quad \text{s.t.} \quad \|D\alpha - y\|_2 \leq \varepsilon
\]

Solve Instead

\[
\min_{\alpha} \|\alpha\|_1 \quad \text{s.t.} \quad \|D\alpha - y\|_2 \leq \varepsilon
\]

- This is known as the Basis-Pursuit (BP) [Chen, Donoho & Saunders ('95)].
- The newly defined problem is convex (quad. programming).
- Very efficient solvers can be deployed:
  - Interior point methods [Chen, Donoho, & Saunders ('95)] [Kim, Koh, Lustig, Boyd, & D. Gorinevsky ('07)].
  - Sequential shrinkage for union of ortho-bases [Bruce et.al. ('98)].
  - Iterative shrinkage [Figueroido & Nowak ('03)] [Daubechies, Defrise, & De-Mole ('04)] [E. ('05)] [E., Matalon, & Zibulevsky ('06)] [Beck & Teboulle ('09)] …
Go Greedy: Matching Pursuit (MP)

- The MP is one of the greedy algorithms that finds one atom at a time [Mallat & Zhang ('93)].

- Step 1: find the one atom that best matches the signal.

- Next steps: given the previously found atoms, find the next one to best fit the residual.
  - The algorithm stops when the error $\|D\alpha - y\|_2$ is below the destination threshold.

- The Orthogonal MP (OMP) is an improved version that re-evaluates the coefficients by Least-Squares after each round.
Pursuit Algorithms

\[
\min_{\alpha} \|\alpha\|_0 \quad \text{s.t.} \quad \|D\alpha - y\|_2^2 \leq \varepsilon^2
\]

There are various algorithms designed for approximating the solution of this problem:

- **Relaxation Algorithms**: Basis Pursuit (a.k.a. LASSO), Dnatzig Selector & numerical ways to handle them [1995-today].
- **Hybrid Algorithms**: StOMP, CoSaMP, Subspace Pursuit, Iterative Hard-Thresholding [2007-today].
- …
BP and MP Equivalence (No Noise)

\[ \hat{\alpha} = \text{ArgMin}_{\alpha} \| \alpha \|_0^0 \quad \text{s.t.} \quad x = D\alpha \]
Given a signal $x$ with a representation $x = D\alpha$, assuming that $\|\alpha\|_0^0 < 0.5(1 + 1/\mu)$, BP and MP are guaranteed to find the sparsest solution.

- MP and BP are different in general (hard to say which is better).
- The above result corresponds to the worst-case, and as such, it is too pessimistic.
- Average performance results are available too, showing much better bounds [Donoho (’04)] [Candes et.al. (’04)] [Tanner et.al. (’05)] [E. (’06)] [Tropp et.al. (’06)] … [Candes et. al. (’09)].
BP Stability for the Noisy Case

\[ \min_{\alpha} \lambda \| \alpha \|_1 + \| D\alpha - y \|_2^2 \]
BP Stability for the Noisy Case

Stability

Given a signal $y = D\alpha + v$ with a representation satisfying $\|\alpha\|_0^0 < 1/3\mu$ and a white Gaussian noise $v \sim N(0, \sigma^2 I)$, BP will show stability, i.e.,

$$\|\hat{\alpha}_{BP} - \alpha\|_2^2 < \text{Const}(\lambda) \cdot \log K \cdot \|\alpha\|_0^0 \cdot \sigma^2$$

Ben-Haim, Eldar & E. (‘09)

* With very high probability

- For $\sigma=0$ we get a weaker version of the previous result.
- This result is the oracle’s error, multiplied by $C \cdot \log K$.
- Similar results exist for other pursuit algorithms (Dantzig Selector, Orthogonal Matching Pursuit, CoSaMP, Subspace Pursuit, ...).
To Summarize So Far …

Image denoising (and many other problems in image processing) requires a model for the desired image. What do we do? Use a model for signals/images based on sparse and redundant representations. Problems?

We have seen that there are approximation methods to find the sparsest solution, and there are theoretical results that guarantee their success.

The Dictionary $D$ should be found somehow !!!

What next?
What Should D Be?

\[ \hat{\alpha} = \arg\min_{\alpha} \| \alpha \|_0^0 \text{ s.t. } \frac{1}{2} \| D\alpha - y \|_2^2 \leq \epsilon^2 \Rightarrow \hat{x} = D\hat{\alpha} \]

Our Assumption: Good-behaved Images have a sparse representation

D should be chosen such that it sparsifies the representations

One approach to choose D is from a known set of transforms (Steerable wavelet, Curvelet, Contourlets, Bandlets, Shearlets …)

The approach we will take for building D is training it, based on Learning from Image Examples
Each example is a linear combination of atoms from $D$

$$\begin{bmatrix} X & \ldots & \vdots \\ \vdots & \ddots & \vdots \\ \vdots & \vdots & D \end{bmatrix} \approx \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_p \end{bmatrix}$$

Min$_{D,A}$\[ \sum_{j=1}^{p} \|D\alpha_j - x_j\|_2^2 \]

s.t. \quad \forall j, \quad \|\alpha_j\|_0 \leq L

Each example has a sparse representation with no more than $L$ atoms

[Field & Olshausen ('96)]
[Engan et al. ('99)]
[Lewicki & Sejnowski ('00)]
[Cotter et al. ('03)]
[Gribonval et al. ('04)]
[Aharon, E. & Bruckstein ('04)]
[Aharon, E. & Bruckstein ('05)]
K–Means For Clustering

Clustering: An extreme sparse representation

- Initialize \( D \)
- Sparse Coding
  - Nearest Neighbor
- Dictionary Update
  - Column-by-Column by Mean computation over the relevant examples
The K–SVD Algorithm – General

[Aharon, E. & Bruckstein (‘04,’05)]

**Initialize** \( D \)

**Sparse Coding**
Use Matching Pursuit

**Dictionary Update**
Column-by-Column by SVD computation over the relevant examples

\[ X^T \]

\( D \)
K–SVD: Sparse Coding Stage

\[ \text{Min} \sum_{j=1}^{p} \left\| D_{\alpha_j} - x_j \right\|^2_2 \quad \text{s.t.} \quad \forall j, \left\| \alpha_j \right\|^p \leq L \]

D is known! For the j\textsuperscript{th} item we solve

\[ \text{Min} \left\| D\alpha - x_j \right\|^2_2 \quad \text{s.t.} \quad \left\| \alpha \right\|^p \leq L \]

Solved by A Pursuit Algorithm
K–SVD: Dictionary Update Stage

Refer only to the examples that use the column $d_k$.

Fixing all $A$ and $D$ apart from the $k^{th}$ column, and seek both $d_k$ and the $k^{th}$ column in $A$ to better fit the residual!

We should solve:

$$\min_{d_k, \alpha_k} \left\| \alpha_k d_k^T \right\|_F^2 \quad - \quad \|E\|_F^2$$
K–SVD: Dictionary Update Stage

Refer only to the examples that use the column $d_k$.

Fixing all $A$ and $D$ apart from the $k^{th}$ column, and seek both $d_k$ and the $k^{th}$ column in $A$ to better fit the residual!

We should solve:

$$\min_{d_k, \alpha_k} \| \alpha_k d_k^T - E \|_F^2$$
K–SVD: Algorithm

Task: Find the best dictionary to represent the data samples \( \{y_i\}_{i=1}^{N} \) as sparse compositions, by solving

\[
\min_{D,X} \left\{ \|Y - DX\|_F^2 \right\} \quad \text{subject to} \quad \forall i, \|x_i\|_0 \leq T_0.
\]

Initialization: Set the dictionary matrix \( D^{(0)} \in \mathbb{R}^{n \times K} \) with \( \ell^2 \) normalized columns. Set \( J = 1 \).

Repeat until convergence (stopping rule):

- **Sparse Coding Stage**: Use any pursuit algorithm to compute the representation vectors \( x_i \) for each example \( y_i \), by approximating the solution of

\[
i = 1, 2, \ldots, N; \quad \min_{x_i} \left\{ \|y_i - Dx_i\|_2^2 \right\} \quad \text{subject to} \quad \|x_i\|_0 \leq T_0.
\]

- **Codebook Update Stage**: For each column \( k = 1, 2, \ldots, K \) in \( D^{(J-1)} \), update it by
  - Define the group of examples that use this atom, \( \omega_k = \{i \mid 1 \leq i \leq N, \ x^k_i(i) \neq 0\} \).
  - Compute the overall representation error matrix, \( E_k \), by
    \[
    E_k = Y - \sum_{j \neq k} d_j x^j_k.
    \]
    - Restrict \( E_k \) by choosing only the columns corresponding to \( \omega_k \), and obtain \( E_k^{\omega_k} \).
    - Apply SVD decomposition \( E_k^{\omega_k} = U \Delta V^T \). Choose the updated dictionary column \( d_k \) to be the first column of \( U \). Update the coefficient vector \( x^k_k \) to be the first column of \( V \) multiplied by \( \Delta(1,1) \).
  - Set \( J = J + 1 \).
To Summarize So Far …

Image denoising (and many other problems in image processing) requires a model for the desired image.

What do we do?

Use a model for signals/images based on sparse and redundant representations.

What next?

We have seen that there are approximation methods to find the sparsest solution, and there are theoretical results that guarantee their success.

Will it all work in applications?

Problems?
From Local to Global Treatment

- The K-SVD algorithm is reasonable for low-dimension signals (N in the range 10-400). As N grows, the complexity and the memory requirements of the K-SVD become prohibitive.

- So, how should large images be handled?

- The solution: Force shift-invariant sparsity - on each patch of size N-by-N (N=8) in the image, including overlaps.

\[
\hat{x} = \text{ArgMin}_{x, \{\alpha_{ij}\}} \frac{1}{2} \|x - y\|^2_2 + \mu \sum_{ij} \|R_{ij}x - D\alpha_{ij}\|^2_2
\]

\[\text{s.t. } \|\alpha_{ij}\|_0^0 \leq L\]

Our prior

Extracts a patch in the ij location
What Data to Train On?

Option 1:
• Use a database of images,
• We tried that, and it works fine (~0.5-1dB below the state-of-the-art).

Option 2:
• Use the corrupted image itself!!
• Simply sweep through all patches of size N-by-N (overlapping blocks),
• Image of size $1000^2$ pixels $\rightarrow \sim 10^6$ examples to use – more than enough.
• This works much better!
K-SVD Image Denoising

\[ \hat{x} = \text{ArgMin}_{x, \{\alpha_{ij}\}} \frac{1}{2} \|x - y\|^2_2 + \mu \sum_{ij} \|R_{ij}x - D\alpha_{ij}\|^2_2 \quad \text{s.t.} \quad \|\alpha_{ij}\|_0^0 \leq L \]

- \( x = y \) and \( D \) known
- \( x \) and \( \alpha_{ij} \) known
- \( D \) and \( \alpha_{ij} \) known

Compute \( \alpha_{ij} \) per patch
\[
\alpha_{ij} = \text{Min}_{\alpha} \|R_{ij}x - D\alpha\|^2_2 \quad \text{s.t.} \quad \|\alpha\|_0^0 \leq L
\]
using the matching pursuit

Compute \( D \) to minimize
\[
\text{Min} \sum_{\alpha_{ij}} \|R_{ij}x - D\alpha\|^2_2
\]
using SVD, updating one column at a time

Compute \( x \) by
\[
\hat{x} = \left[I + \mu \sum_{ij} R_{ij}^T R_{ij}\right]^{-1} \left[y + \mu \sum_{ij} R_{ij}^T D\alpha_{ij}\right]
\]
which is a simple averaging of shifted patches
Image Denoising (Gray)  [E. & Aharon (’06)]

Source

Result 30.829dB

Noisy image \( \sigma = 20 \)

The obtained dictionary after 10 iterations
Image Denoising (Gray) [E. & Aharon (’06)]

- The results of the K-SVD algorithm compete favorably with the state-of-the-art.
- In a recent work that extended this algorithm to use joint sparse representation on the patches, the best published denoising performance are obtained [Mairal, Bach, Ponce, Sapiro & Zisserman (’09)].

\[ \sigma = 20 \]
Denoising (Color) [Mairal, E. & Sapiro (‘08)]

• When turning to handle color images, the main difficulty is in defining the relation between the color layers – R, G, and B.

• The solution with the above algorithm is simple – consider 3D patches or 8-by-8 with the 3 color layers, and the dictionary will detect the proper relations.
Denoising (Color) [Mairal, E. & Sapiro (‘08)]

Original               Noisy (20.43dB)          Result  (30.75dB)
The K-SVD algorithm leads to state-of-the-art denoising results, giving ~1dB better results compared to [Mcauley et. al. ('06)] which implements a learned MRF model (Field-of-Experts).
Image Inpainting – The Basics

• Assume: the signal $x$ has been created by $x = D\alpha_0$ with very sparse $\alpha_0$.

• Missing values in $x$ imply missing rows in this linear system.

• By removing these rows, we get $\tilde{D}\alpha = \tilde{x}$

• Now solve

$$\min_{\alpha} \|\alpha\|_0 \quad \text{s.t.} \quad \tilde{x} = \tilde{D}\alpha$$

• If $\alpha_0$ was sparse enough, it will be the solution of the above problem! Thus, computing $D\alpha_0$ recovers $x$ perfectly.
Side Note: Compressed-Sensing

- **Compressed Sensing** is leaning on the very same principal, leading to alternative sampling theorems.
- Assume: the signal $x$ has been created by $x = D\alpha_0$ with very sparse $\alpha_0$.
- Multiply this set of equations by the matrix $Q$ which reduces the number of rows.
- The new, smaller, system of equations is
  \[
  QD\alpha = Qx \Rightarrow \tilde{D}\alpha = \tilde{x} \times
  \]
- If $\alpha_0$ was sparse enough, it will be the sparsest solution of the new system, thus, computing $D\alpha_0$ recovers $x$ perfectly.
- Compressed sensing focuses on conditions for this to happen, guaranteeing such recovery.
Experiments lead to state-of-the-art inpainting results.
Experiments lead to state-of-the-art inpainting results.
Experiments lead to state-of-the-art inpainting results.

Since 1699, when French explorers landed at the great bend of the Mississippi River and celebrated the first Mardi Gras in North America, New Orleans has brewed a fascinating melange of cultures. It was French, then Spanish, then French again, then sold to the United States. Through all these years, and even into the 1900s, others arrived from everywhere: Acadians (Cajuns), Africans, indige-
The problem: Compressing photo-ID images.

General purpose methods (JPEG, JPEG2000) do not take into account the specific family.

By adapting to the image-content (PCA/K-SVD), better results could be obtained.

For these techniques to operate well, train dictionaries locally (per patch) using a training set of images is required.

In PCA, only the (quantized) coefficients are stored, whereas the K-SVD requires storage of the indices as well.

Geometric alignment of the image is very helpful and should be done [Goldenberg, Kimmel, & E. (‘05)].
Image Compression

Detect main features and warp the images to a common reference (20 parameters)

Divide the image into disjoint 15-by-15 patches. For each, compute mean and dictionary

Per each patch find the operating parameters (number of atoms \( L \), quantization \( Q \))

Warp, remove the mean from each patch, sparse code using \( L \) atoms, apply \( Q \), and dewarp

Training set (2500 images)
Image Compression Results

Original
JPEG
JPEG-2000
Local-PCA
K-SVD

Results for 820 Bytes per each file
Image Compression Results

<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>JPEG</th>
<th>JPEG-2000</th>
<th>Local-PCA</th>
<th>K-SVD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Results for 550 Bytes per each file</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>15.81</td>
<td>14.67</td>
<td>13.89</td>
<td>15.30</td>
<td>14.67</td>
</tr>
<tr>
<td></td>
<td>13.89</td>
<td>12.41</td>
<td>12.41</td>
<td>12.57</td>
<td>12.41</td>
</tr>
<tr>
<td></td>
<td>10.66</td>
<td>9.44</td>
<td>9.44</td>
<td>10.27</td>
<td>9.44</td>
</tr>
<tr>
<td></td>
<td>6.60</td>
<td>5.49</td>
<td>5.49</td>
<td>6.36</td>
<td>5.49</td>
</tr>
</tbody>
</table>
Image Compression Results

Original
JPEG
JPEG-2000
Local-PCA
K-SVD

Results for 400 Bytes per each file

18.62
12.30
7.61
16.12
11.38
6.31
16.81
12.54
7.20
Deblocking the Results [Bryt and E. (09)]

550 bytes K-SVD results with and without deblocking


Deblock (6.24)  Deblock (5.27)  Deblock (6.03)  Deblock (11.32)
Super-Resolution [Zeyde, Protter, & E. (‘11)]

• Given a low-resolution image, we desire to enlarge it while producing a sharp looking result. This problem is referred to as “Single-Image Super-Resolution”.

• Image scale-up using bicubic interpolation is far from being satisfactory for this task.

• Recently, a sparse and redundant representation technique was proposed [Yang, Wright, Huang, and Ma (‘08)] for solving this problem, by training a coupled-dictionaries for the low- and high res. images.

• We extended and improved their algorithms and results.
Super-Resolution – Results (1)

This book is about convex optimization, a special class of mathematical optimization problems, which includes least-squares and linear programming problems. It is well known that least-squares and linear programming problems have a fairly complete theory, arise in a variety of applications, and can be solved numerically very efficiently. The basic point of this book is that the same can be said for the larger class of convex optimization problems.

While the mathematics of convex optimization has been studied for about a century, several related recent developments have stimulated new interest in the topic. The first is the recognition that interior-point methods, developed in the 1980s to solve linear programming problems, can be used to solve convex optimization problems as well. These new methods allow us to solve certain new classes of convex optimization problems, such as semidefinite programs and second-order cone programs, almost as easily as linear programs.

The second development is the discovery that convex optimization problems (beyond least-squares and linear programs) are more prevalent in practice than was previously thought. Since 1990 many applications have been discovered in areas such as automatic control systems, estimation and signal processing, communications and networks, electronic circuit design, data analysis and modeling, statistics, and finance. Convex optimization has also found wide application in combinatorial optimization and global optimization, where it is used to find bounds on the optimal value, as well as approximate solutions. We believe that many other applications of convex optimization are still waiting to be discovered.

There are great advantages to recognizing or formulating a problem as a convex optimization problem. The most basic advantage is that the problem can then be solved, very reliably and efficiently, using interior-point methods or other specialized methods for convex optimization. These solution methods are reliable enough to be embedded in a computer-aided design or analysis tool, or even a real-time reactive or automatic control system. There are also theoretical or conceptual advantages of formulating a problem as a convex optimization problem. The associated dual

The training image: 717×717 pixels, providing a set of 54,289 training patch-pairs.
Super-Resolution – Results (1)

SR Result
PSNR=16.95dB

Ideal Image

An amazing variety of practical problems (design, analysis, and operation) can be formulated as optimization problems, or some variation thereof. Indeed, mathematical optimization has been widely used in engineering, in control systems, and optimal design problems in aerospace engineering. Optimization design and operation, finance, supply chain, and other areas. The list of applications is still growing.

For most of these applications, mathematics is a human decision maker, system designer, process, checks the results, and modifies when necessary. This human decision maker or the optimization problem, e.g., buying a portfolio.

Bicubic interpolation
PSNR=14.68dB

Bicubic interpolation PSNR=14.68dB

Given Image

An amazing variety of practical problems (design, analysis, and operation) can be formulated as optimization problems, or some variation thereof. Indeed, mathematical optimization has been widely used in engineering, in control systems, and optimal design problems in aerospace engineering. Optimization design and operation, finance, supply chain, and other areas. The list of applications is still growing.

For most of these applications, mathematics is a human decision maker, system designer, process, checks the results, and modifies when necessary. This human decision maker or the optimization problem, e.g., buying a portfolio.
Super-Resolution – Results (2)

Scaled-Up (factor 2:1) using the proposed algorithm, PSNR=29.32dB (3.32dB improvement over bicubic)
Super-Resolution – Results (2)

The Original  Bicubic Interpolation  SR result
Super-Resolution – Results (2)

The Original  Bicubic Interpolation  SR result
L0-Image Smoothing

General goals:

• Suppress insignificant details
• Maintain major edges
L0-Smoothing Method

A general and effective global smoothing strategy based on a sparsity measure

\[ c(f) := \# \{ p \mid |\nabla f_p| \neq 0 \} \]

which corresponds to the L0-norm of gradient
Two Features

1. **Flattening** insignificant details

By removing small non-zero gradients
Two Features

2. Enhancing prominent edges

Because large gradients receive the same penalty as small ones

\[ \# \{ p \mid | \nabla f_p | \neq 0 \} = \# \{ p \mid | \alpha | \nabla f_p | \neq 0 \} \]
Our Framework in 1D

• Constrain # of non-zero gradients
\[ c(f) = \#\{p \mid |f_p - f_{p+1}| \neq 0\} = k \]

• Make the result similar to the input
\[ \min f \sum_p (f_p - g_p)^2 \]

• Objective function
\[ \min f \sum_p (f_p - g_p)^2 \text{ s.t. } c(f) = k \]
Our Framework in 1D

• Input 1D signal $g$

$$\min_{f} \sum_{p} (f_p - g_p)^2 \quad \text{s.t.} \quad c(f) = 1$$
Our Framework in 1D

- Input 1D signal $g$

$$\min_f \sum_p (f_p - g_p)^2 \quad \text{s.t.} \quad c(f) = 2$$
Our Framework in 1D

- **Input 1D signal** $g$

\[
\min_f \sum_p (f_p - g_p)^2 \quad \text{s.t.} \quad c(f) = 5
\]
Our Framework in 1D

- Input 1D signal $g$

\[
\min_f \sum_p (f_p - g_p)^2 \quad \text{s.t.} \quad c(f) = 200
\]
Transformation

\[ \min_f \sum_p (f_p - g_p)^2 \quad \text{s.t.} \quad c(f) = k \]

\[ \min_f \sum_p (f_p - g_p)^2 + \lambda \cdot c(f) \]
Finding the global optimum is NP hard
Approximation

\[
\min_f \sum_p (f_p - g_p)^2 + \lambda \cdot c(h, \nu)
\]

\[
+ \beta \cdot \sum_p (|\partial_x f_p - h_p|^2 + |\partial_y f_p - v_p|^2)
\]

Separately estimate \( f \) and \((h, \nu)\)
Iterative Optimization

- Compute $f$ given $h, v$

$$E(f) = \sum (f_p - g_p)^2 + \beta \cdot \left( (\partial_x f_p - h_p)^2 + (\partial_y f_p - v_p)^2 \right)$$

Both the sub-problems are with closed-form solutions

$$E(h, v) = \sum \left( (\partial_x f_p - h_p)^2 + (\partial_y f_p - v_p)^2 \right) + \frac{c(h, v)}{\beta}$$

- Gradually approximate the original problem

$\beta \leftarrow 2\beta$
One Example

Converge in 15 iterations
Smoothing Strength

Input
Smoothing Strength

\[ \lambda = 0.01 \]
Smoothing Strength

$\lambda = 0.02$
Smoothing Strength

$\lambda = 0.03$
Comparison

Total Variation

L0 Smoothing