Topics on High-Dimensional Data Analytics

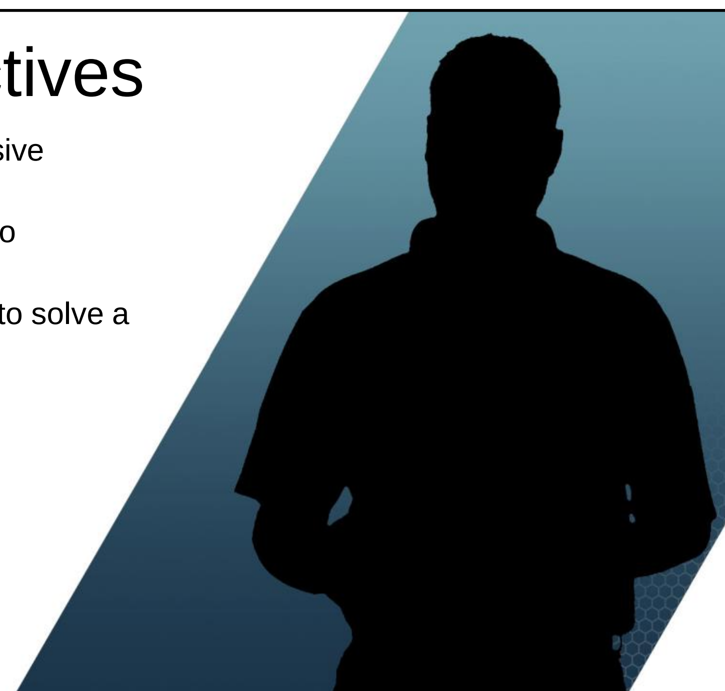
Regularization Applications

Kamran Paynabar, Ph.D.
Associate Professor
School of Industrial and Systems Engineering

Compressive Sensing I

Learning Objectives

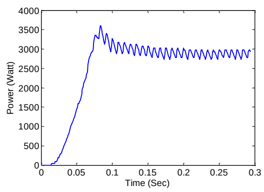
- Explain principles of compressive sensing
- Use L0 and L1 regularization to formulate a CS problem.
- Apply optimization algorithms to solve a CS problem.



Sparse Information and Regularization

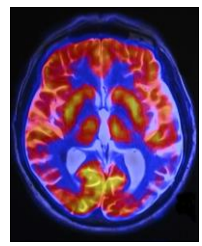
- High-dimensional data usually have a low dimensional structure.
- Important information of HD data is embedded in a few dimensions (sparse) and the rest are non-informative and noise.

Single-channel signals

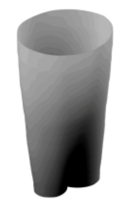


Multi-channel signals

Images



Point Cloud



- Regularization helps identify the spares informative features and remove the noise.

Sparse Representation of Image

- Image Compression-JPEG

(Patches of) … input image

A DCT basis

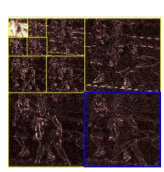
(Discrete cosine transform)

x coefficients

compression:
JPEG, JPEG2000, MPEG, ...

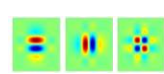
- Wavelet

x



α_i

$$x = \sum_{i=1}^{N} \alpha_i \psi_i$$



$$\alpha_i = \langle x, \psi_i \rangle$$

Image Compression

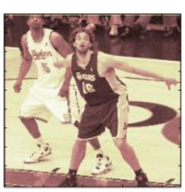
- Keep the 1% largest coefficients and set the rest to zero.

original

approximated

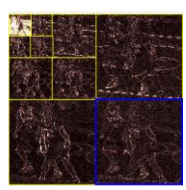
rel. error $= 0.031$

Conventional Sampling & Compression



N

High-Resolution Sensing



Compression

Storage/ Transmission

$M \ll N$

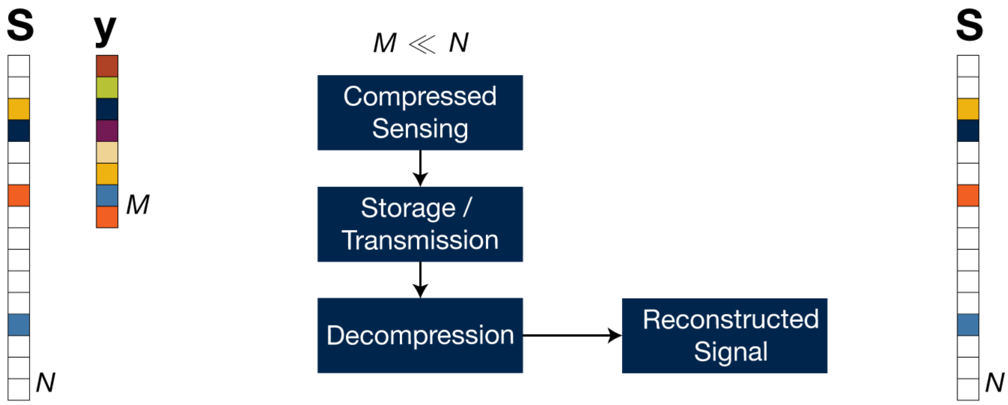
Decompression

N

Reconstructed Signal

Nyquist rate: the minimum rate at which a signal can be sampled without introducing errors, which is twice the highest frequency present in the signal.

Compressed Sampling



The original or transformed signal is sparse (a few non-zero elements).

A Simple Example

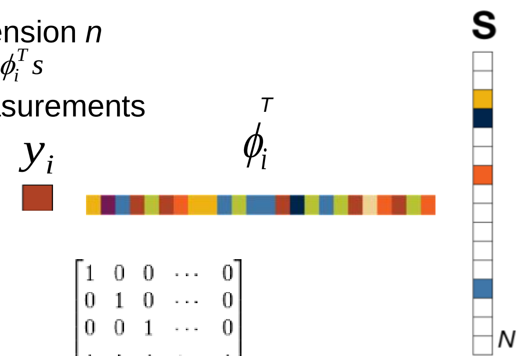
- Consider a 1-support signal (i.e., all zeros but one).
- Using Hamming Matrix the signal can be measured in a compressed way

$$\begin{bmatrix} 5 \\ 5 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 0 \\ 5 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

- Only $\log_2 n$ measurement is needed to recover the signal

Sensing Matrix

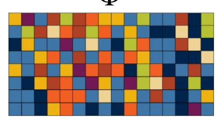
- Redefine sensing approach by using sensing matrix
- Sensing Matrix
  - Each measurement is a vector ϕ_i of dimension *n*
  - Given ϕ_i and signal, measurement $y_i = \phi_i^T s$
  - Cost is proportional to the number of measurements



$$\begin{matrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{matrix}$$

Hamming Matrix

Φ



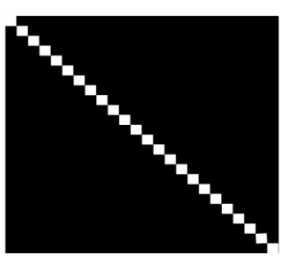
Random Matrix

$$\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

Identity Matrix

Traditional vs Compressive Sensing Matrix

$$\begin{bmatrix} Y_1 \\ \vdots \\ \vdots \\ Y_n \end{bmatrix} =$$



$$\begin{bmatrix} f_1^* \\ f_2^* \\ \vdots \\ \vdots \\ f_n^* \end{bmatrix}$$

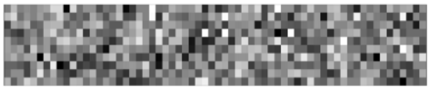
Sample Uniformly at a high rate (Nyquist Rate)

$$\begin{bmatrix} Y_1 \\ \vdots \\ Y_k \end{bmatrix} =$$

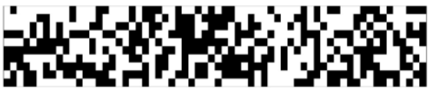
$$\begin{bmatrix} f_1^* \\ f_2^* \\ \vdots \\ \vdots \\ f_n^* \end{bmatrix}$$

Sample randomly, and Take very few samples

Examples of Random Sensing Matrices



Gaussian random matrix



Rademacher matrix
(i.i.d. of +1 and -1 entries)

Fourier random matrix
(rows are randomly selected DFT vectors)

Topics on High-Dimensional Data Analytics

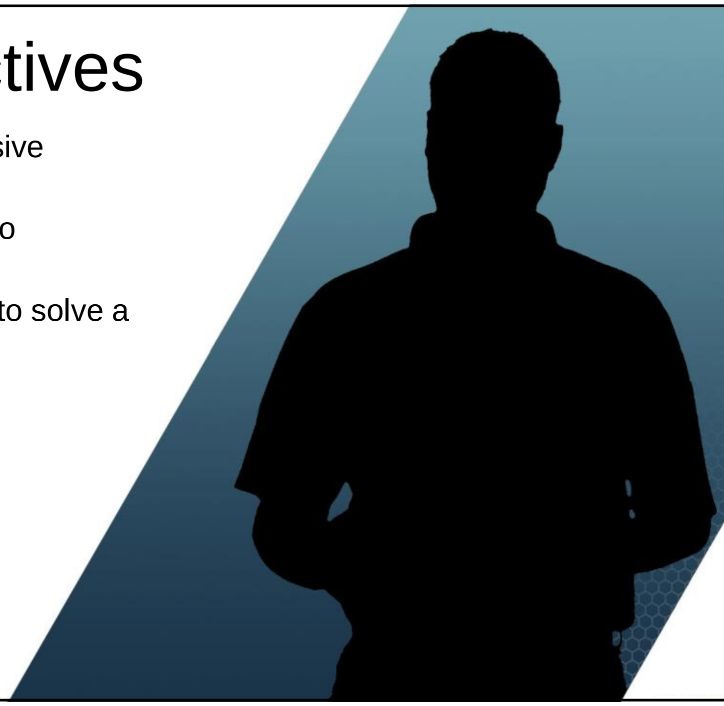
Regularization Applications

Kamran Paynabar, Ph.D.
Associate Professor
School of Industrial and Systems Engineering

Compressive Sensing II

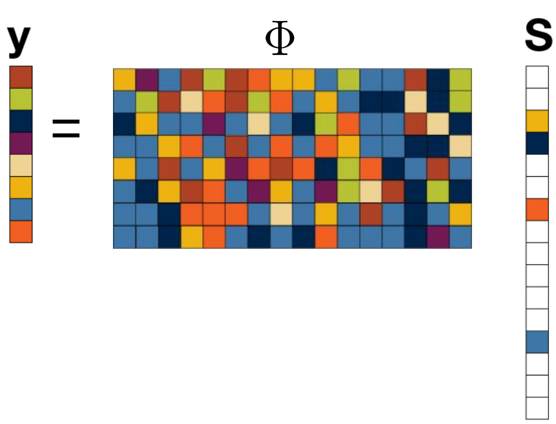
Learning Objectives

- Explain principles of compressive sensing
- Use L0 and L1 regularization to formulate a CS problem.
- Apply optimization algorithms to solve a CS problem.



Compressive Sensing Framework

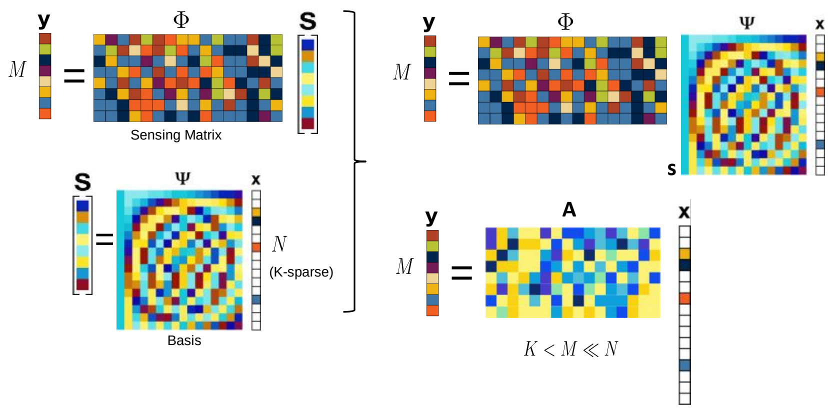
If the signal is sparse in original domain, then we use a fat sensing matrix for sampling.



Compressive Sensing Framework

s

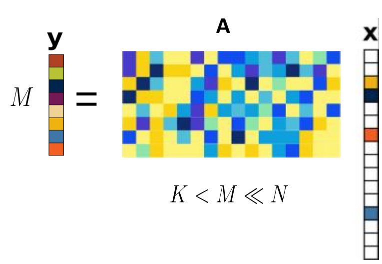
If the signal is NOT sparse in original domain, we should transform it into a domain with sparse representation.



$$y = \Psi s' = \Phi\Psi x = Ax$$

Signal Reconstruction from Measurements

- We should solve an underdetermined linear system $y = Ax$



- Search for the sparsest signal that agrees with the measurements

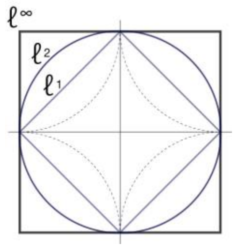
$$\text{minimize } \|\boldsymbol{x}\|_0 \quad \textit{subject to} \quad \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}.$$

$$\|\boldsymbol{x}\|_0 \doteq \#\{i \mid x_i \neq 0\}$$

Non-convex and Interactable

Relaxation of L_0 by L_2

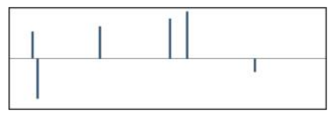
$\|\boldsymbol{x}\|_p = (\sum_i |x_i|^p)^{1/p}$



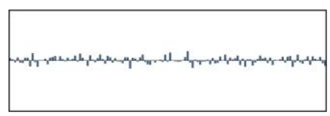
Problem is convex when $p \geq 1$

L_2 Norm: fast but inaccurate results

$$\hat{x} = \arg \min_{y=Ax} \|x\|_2$$



x

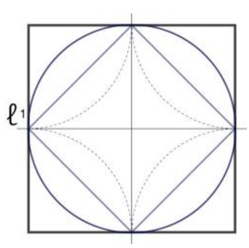


$$\hat{x} = (A^T A)^{-1} A^T y$$

Relaxation of L_0 by L_1

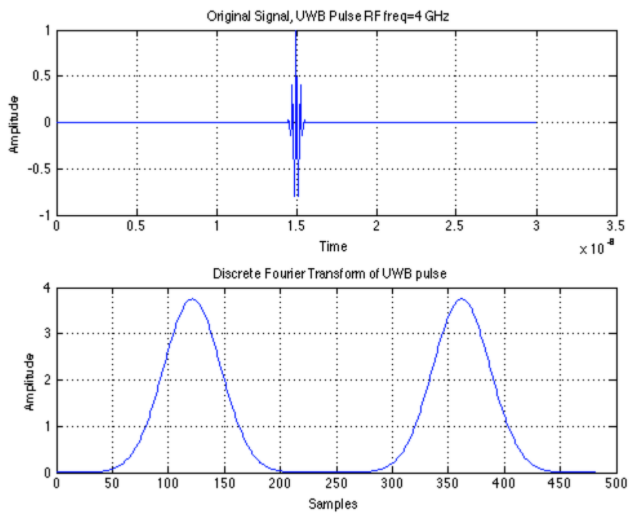
$$\text{minimize} \quad \|\boldsymbol{x}\|_1 \quad \text{subject to} \quad \boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}.$$

Efficiently solvable



Norm Used	Property	Formulation
L2	Fast, Wrong	$\hat{x} = \arg \min_{y=Ax} \|x\|_2$
L0	Slow, Correct	$\hat{x} = \arg \min_{y=Ax} \|x\|_0$
L1	Fast, Mild Oversampling	$\hat{x} = \arg \min_{y=Ax} \|x\|_1$

Example 1: Sparse in Time Domain



Generate Signal by

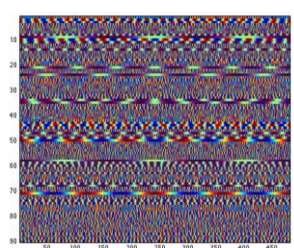
x=pulstran(t,15e-9,'gauspuls',f,0.5);

FFT analysis

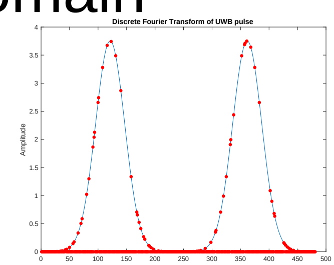
xf=fft(x);

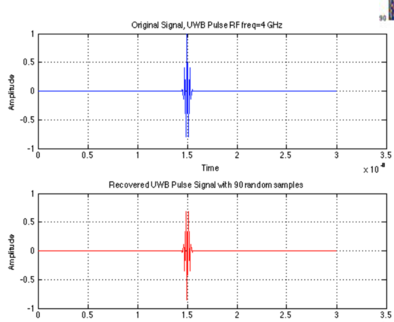
Example 1: Sparse in Time Domain

Sensing Matrix:
Randomly selecting rows
of the DFT matrix



M=90

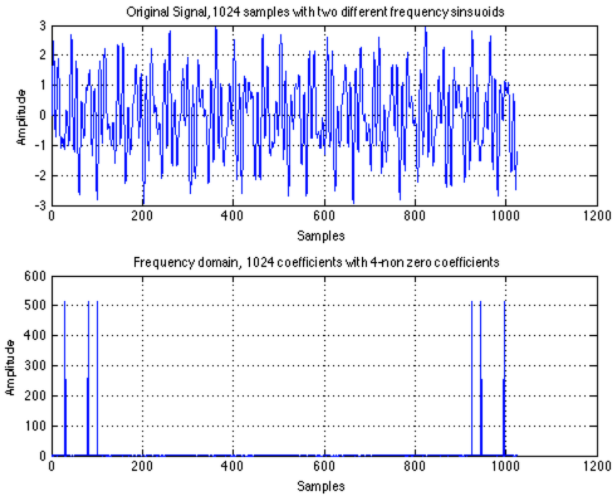





```
%creating Discrete Fourier matrix
B=dftmtx(N);
%Selecting random rows of the DFT matrix
q=randperm(N);
%creating measurement matrix
A=B(q(1:K),:);
%taking random frequency measurements
y=(A*x');
% Calculating Initial guess
x0=A'*y;
%Running the recovery Algorithm
xp=l1eq_pd(x0,A,[],y,1e-5);
```

Recovery by solving $\hat{x} = \arg\min_{y=Ax} \|x\|_1$

Example 2: Sparse in Frequency Domain



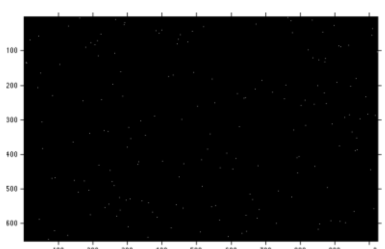
Generate Signal by

x=(sin(2*pi*(k1/N)*n)+sin(2*pi*(k2/N)*n)+sin(2*pi*(k3/N)*n))';

FFT analysis

xf=fft(x);

Example 2: Sparse in Frequency Domain

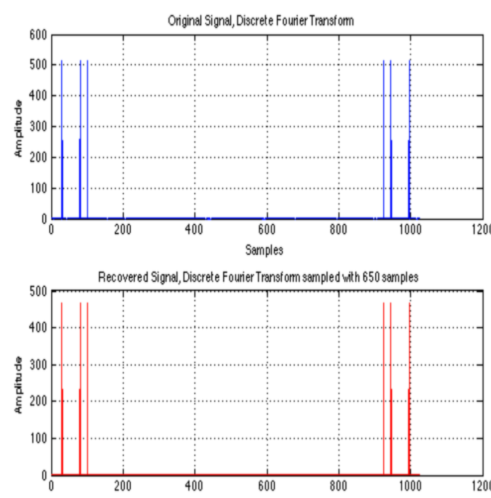


Φ Sensing Matrix:

Random rows of identity matrix with **M=650**

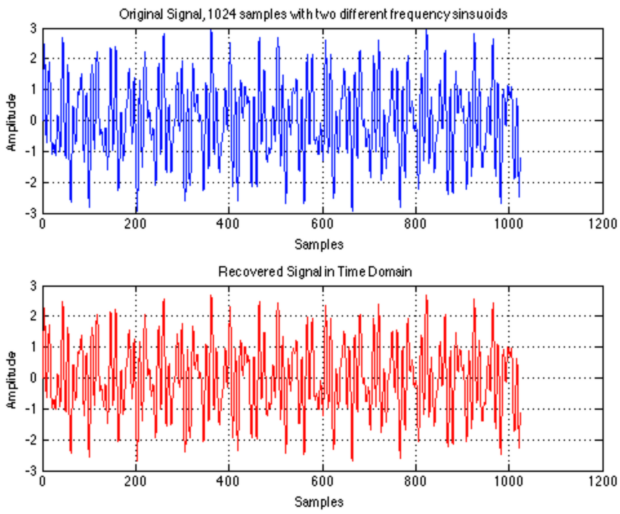
Ψ Fourier Basis $\quad y = \Psi s' = \Phi\Psi x = Ax$

```
ID = eye(N);q=randperm(N);
Phi=IN(:,q(1:K))';
Psi = dftmtx(N);
%taking random time measurements
y=(Phi*x);
%Calculating Initial guess
x0= Psi'*(Phi'*y);
%Running the recovery Algorithm
xp=l1eq_pd(x0,Phi*Psi,[],y,1e-7);
%recovered signal in time domain
xprec=real(-inv(Psi)*xp);
```



Recovery by solving $\quad \hat{x} = \arg\min_{y=Ax} \|x\|_1$

Example 2: Sparse in Frequency Domain



Exact
Recovery

```
%recovered signal in time domain
xprec=real(-inv(Psi)*xp)
```

Example 3: CS Applications for Images

Take M=48640 incoherent measurements

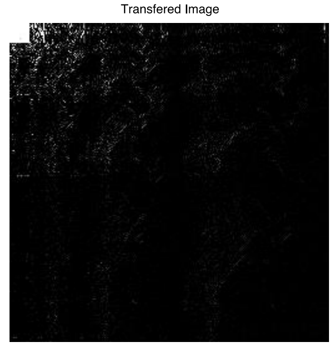
$f_a = \Psi x$ wavelet coefficients (perfectly sparse)

Solve $\min \|x\|_1 \quad s.t. \quad \Phi\Psi x = y$

original Image

256*256=65536 pixels

Transfered Image



Sparse Wavelet Approximation

Wavelet_OMP_SD_main.m

Example 3: CS Applications for Images

original Image

Recovered Image

Recover from ~49k Random Measurement

Wavelet_OMP_SD_main.m

Incoherent Sampling

- Coherence between sensing basis Φ and representation basis Ψ

$$\mu(\Phi, \Psi) = \sqrt{n} \cdot \max_{1 \le k, j \le n} |\langle \varphi_k, \psi_j \rangle|.$$

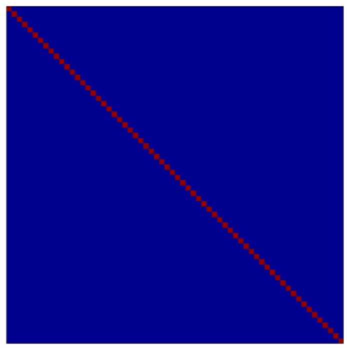
- If these matrices contain correlated elements, the coherence is large. Otherwise, it is small.

- It can be shown that

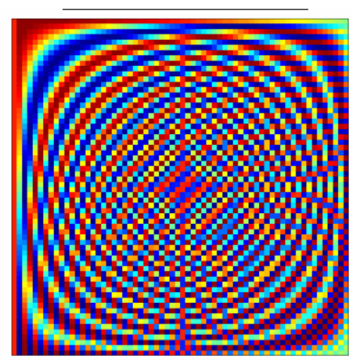
$$\mu(\Phi, \Psi) \in [1, \sqrt{n}]$$

- Compressive Sampling is mainly concerned with low coherence pairs.

Incoherent Pairs: Time-Frequency



Identity Matrix

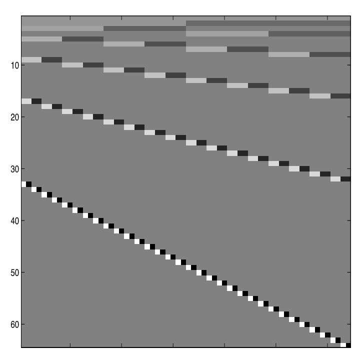


Fourier Series

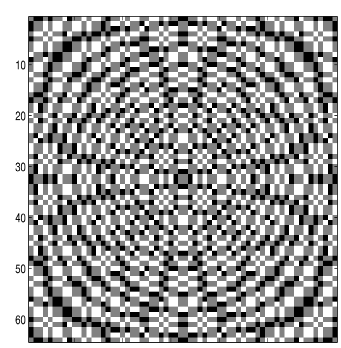
$$\mu(\Phi, \Psi) = 1$$

Maximum Incoherence

Incoherent Pairs: Wavelet and Noiselet



Haar basis



Noiselet basis

$$\mu(\Phi, \Psi) = \sqrt{2}$$

Guaranteed Recovery

- Theorem: Fix *f* and suppose that the coefficient sequence *x* of *f* in the basis Ψ is *S*-Sparse. Select *m* measurements in Φ domain uniformly at random, then if

$$m \geq C \cdot \mu^2(\Phi, \Psi) \cdot S \cdot \log n$$

  holds for some positive constant C, the solution of L_1 formulation is exact with overwhelming probability.

- Remarks
  - Role of coherence is important, smaller the coherence, the fewer samples are needed.
  - If incoherence is close to 1, only (*S log n*) samples are needed instead of *n.*

(Candès and Tao, '04; Donoho, '04):

Signal Recovery from Noisy Measurements and RIP

$$\min \|x\|_{l_1} \qquad \text{subject to} \qquad \|Ax - y\|_{l_2} \leq \epsilon$$

- If restricted isometry property (RIP) holds for the sensing matrix, the signal can be recovered from noisy measurement.

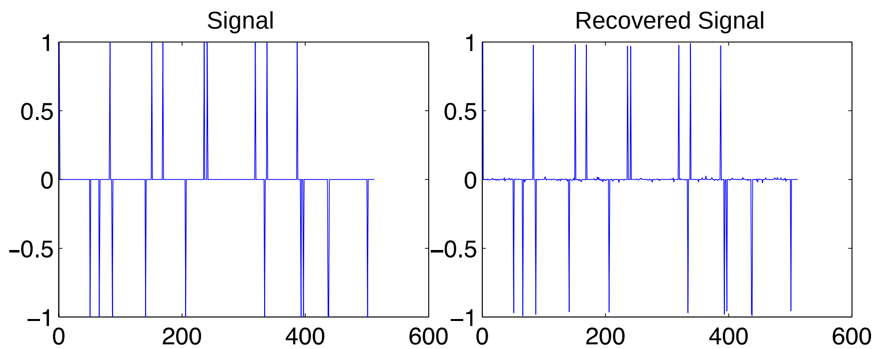
- For each integer S=1,2,…, define the isometry constant δ_S of a matrix *A* as the smallest number such that

$$(1 - \delta_S)\|x\|_{\ell_2}^2 \leq \|Ax\|_{\ell_2}^2 \leq (1 + \delta_S)\|x\|_{\ell_2}^2$$

  holds for all S-sparse vectors *x*

- What does it mean?
  - Matrix *A* Preserve the Euclidean length of S-sparse signals
  - Any *S* columns taken from *A* are in fact nearly orthogonal

Example: Noisy Signal Recovery



Recovered by solving $\min \|x\|_{l_1}$ subject to $\|Ax - y\|_{l_2} \leq \epsilon$

N=512 and M=150 1dnoise_example.m

Example: Noisy Image Recovery

Original Image

Recovered Image

Image Size: 256*256=65K Recovered from 20K measurements

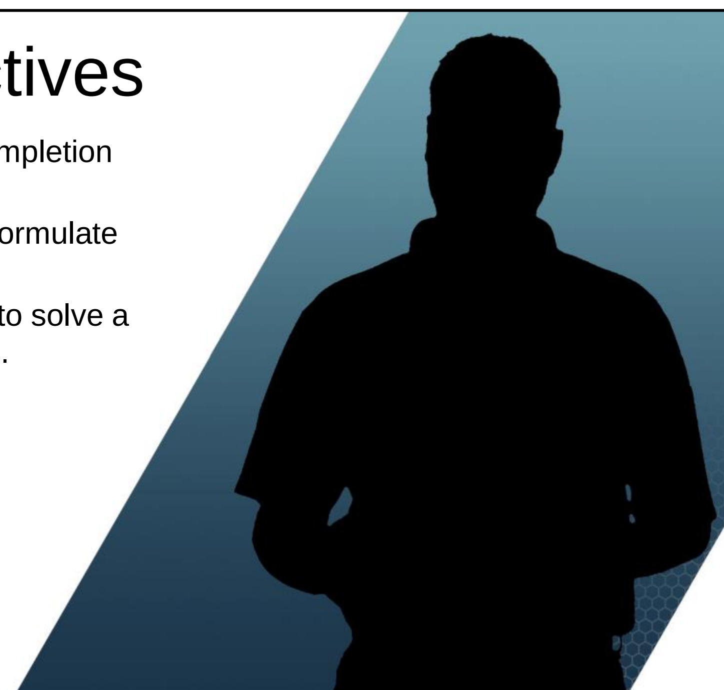
Topics on High-Dimensional Data Analytics

Regularization Applications

Kamran Paynabar, Ph.D.
Associate Professor
School of Industrial and Systems Engineering

Matrix Completion

Learning Objectives



- Explain principles of matrix completion for missing data imputation.
- Use nuclear regularization to formulate a matrix completion problem.
- Apply optimization algorithms to solve a the matrix completion problem.

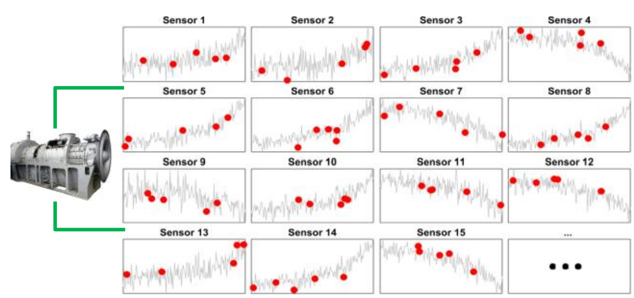
Motivation: Netflix Prize

- Training Data: ~480K users, ~18K movies, 100 M ratings (1-5), 99 % ratings missing
- $1 M prize for 10 % reduction in RMSE over Cinematch (a method developed by Netflix)
- BellKor's Pragmatic Chaos declared winners on 9/21/2009
- used ensemble of models including low-rank factorization

	The Game	Revolutionary Road	Tinker Tailor Soldier Spry	The Imitation Game	...
User 1	★★★★★	★★★	?	★★★★	...
User 2	?	?	★★★★	★★★★★	...
User 3	★★★	★★★★★	?	★★★★	...
User 4	★★★★★	?	★★★★	?	...
...	...	...	...	...	...

Motivation: Multi-Sensors Monitoring

- Hundreds of sensors are used for condition monitoring of industrial assets.
- Different sampling frequencies of sensors and/or their malfunctions will result in missing data.
- Goal is impute the missing values based on the temporal and spatial correlation of sensors.
- This is achievable if the matrix is low rank



	Sensor 1	Sensor 2	Sensor 3	Sensor 4	...
Engine 1					...
Engine 2					...
Engine 3					...
...	...	...	...	...	...

Matrix Completion Formulation

Assuming the that that the matrix is low rank, the missing values can be imputed by solving:

$$\min\{\operatorname{rank}(Z)\}$$

subject to: $Z_{ij} = X_{ij}\,;\ \ (i,j) \in \Omega_{\text{observed}}$

- Use the following projection function $P_\Omega(X)_{n\times m}$:

$$P_\Omega(X)_{i,j} = \begin{cases} X_{i,j} & \text{if } (i,j) \text{ is observed} \\ 0 & \text{if } (i,j) \text{ is missing} \end{cases}$$

$$\Downarrow$$

$$\min\{\operatorname{rank}(Z)\}$$

subject to: $P_\Omega(Z) = P_\Omega(X)$

$$\begin{bmatrix} \times & ? & ? & ? & \times & ? \\ ? & ? & \times & \times & ? & ? \\ \times & ? & ? & \times & ? & ? \\ ? & ? & \times & ? & ? & \times \\ \times & ? & ? & ? & ? & ? \\ ? & ? & \times & \times & ? & ? \end{bmatrix}$$

Matrix Completion Formulation

Assuming the that that the matrix is low rank, the missing values can be imputed by solving:

$$\min\{\operatorname{rank}(Z)\}$$

subject to: $P_\Omega(Z) = P_\Omega(X)$

Convex Relaxation $\Downarrow$

$$\min\{\|Z\|_*\}$$

subject to: $P_\Omega(Z) = P_\Omega(X)$

$$\begin{bmatrix} \times & ? & ? & ? & \times & ? \\ ? & ? & \times & \times & ? & ? \\ \times & ? & ? & \times & ? & ? \\ ? & ? & \times & ? & ? & \times \\ \times & ? & ? & ? & ? & ? \\ ? & ? & \times & \times & ? & ? \end{bmatrix}$$

Singular Value Thresholding (SVT)

Consider the singular value decomposition (SVD) of a matrix $X \in \mathbb{R}^{n_1 \times n_2}$ of rank *r*:

$$X = U\Sigma V^T,$$
$$\Sigma = \mathrm{diag}(\,\{\sigma_i\}_{1 \le i \le r}),$$

where $U \in \mathbb{R}^{n_1 \times r}$ and $V \in \mathbb{R}^{n_2 \times r}$ matrices with orthogonal columns and σ_i are positive singular values.

For each $\tau \geq 0$, the soft-thresholding operator, S_τ is defined by

$$S_\tau(X) = US_\tau(\Sigma)V^T,$$
$$S_\tau(\Sigma) = \mathrm{diag}\{(\sigma_i - \tau)_+\},$$

where $(a)_+$=max(0,a).

Matrix Completion using SVT

SVT can be used to solve the relaxed convex optimization problem:

$$\min\{\|Z\|_*\}$$

subject to: $\quad P_\Omega(Z) = P_\Omega(X)$

For a fixed $\tau > 0$ and a sequence of $\{\delta_k\}$ of positive step sizes, start with $Y_0 = 0 \in \mathbb{R}^{n_1 \times n_2}$, and for $k = 1,2, \ldots$ iteratively compute

$$\begin{cases} Z^k = S_\tau(Y^{k-1}) \\ Y^k = Y^{k-1} + \delta_k P_\Omega\left(X - Z^k\right) \end{cases}$$

Until convergence (i.e., a stopping criterion is reached).

Matrix Completion with Noisy Data

In real life, however, observations are noisy. Therefore, the equality constraint should be relaxed:

$$\min\{\|Z\|_*\}$$

subject to: $\|P_\Omega(Z) - P_\Omega(X)\|_F^2 < \epsilon$

$$\begin{bmatrix} \times & ? & ? & ? & \times & ? \\ ? & ? & \times & \times & ? & ? \\ \times & ? & ? & \times & ? & ? \\ ? & ? & \times & ? & ? & \times \\ \times & ? & ? & ? & ? & ? \\ ? & ? & \times & \times & ? & ? \end{bmatrix}$$

The Langrangian equivalent of the problem is

$$\min\left\{\lambda\|Z\|_* + \frac{1}{2}\|P_\Omega(Z) - P_\Omega(X)\|_F^2\right\},$$

For some $\lambda > 0$.

Matrix Completion with Noisy Data

$$\min\left\{\lambda\|Z\|_* + \frac{1}{2}\|P_\Omega(Z) - P_\Omega(X)\|_F^2\right\},$$

If the PFBS (Proximal Forward-Backward Splitting) method is used, the following iterative algorithm for solving this problem can be obtained:

$$\begin{cases} Z^k = S_{\lambda\delta_{k-1}}(Y^{k-1}), \\ Y^k = Z^k + \delta_k P_\Omega(X - Z^k) \end{cases}$$

Until convergence (i.e., a stopping criterion is reached).

Some theoretical results can be found in Ref. 1 and 2.

Example

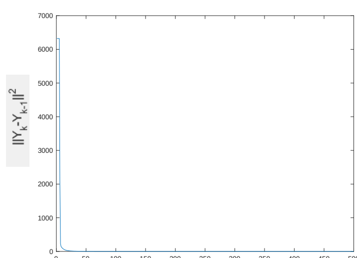
```
%Generating original matrix
n1 = 10; n2 = 8;
A = randi([-20,20],n1,n2);
r = 2;
[U, S, V] = svd(A);
if n1 < n2
    s = diag(S); s(r+1:end)=0; S=[diag(s) zeros(n1,n2-n1)];
else
    s = diag(S); s(r+1:end)=0; S=[diag(s); zeros(n1-n2,n2)];
end

X = U* S* V';
X0 = X;

%Removing 20% of the observations
A = [rand(n1,n2)>=0.80];
X(A) = 0;
m = sum(sum(A==0));
```

2.466	1.153	-7.324	1.214	-1.517	-0.304	-2.04	2.383
11.09	-7.026	0.885	-5.155	-3.33	14.58	11.08	4.589
2.521	6.269	-21.59	5.667	-3.008	-6.959	-10.53	4.991
-6.835	6.415	-6.321	4.99	1.456	-11.71	-10.29	-1.782
-12.75	2.116	15.49	0.744	5.532	-8.97	-2.851	-8.264
4.237	-4.477	5.305	-3.528	-0.759	7.909	7.21	0.853
16.03	-5.505	-11.6	-3.408	-6.143	14.99	8.305	8.966
7.485	-7.445	8.087	-5.829	-1.474	13.37	11.97	1.741
-8.605	-3.455	23.99	-3.744	5.132	0.321	6.179	-8.03
-0.696	4.11	-10.22	3.513	-0.841	-5.706	-6.784	1.553

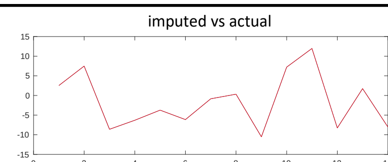
$X =$

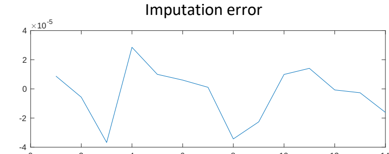
2.466	1.153	-7.324	1.214	-1.517	-0.304	-2.04	2.383
11.09	-7.026	0.885	-5.155	-3.33	14.58	11.08	4.589
0	6.269	-21.59	5.667	-3.008	-6.959	0	4.991
-6.835	6.415	0	4.99	1.456	-11.71	-10.29	-1.782
-12.75	2.116	15.49	0.744	5.532	-8.97	-2.851	0
4.237	-4.477	5.305	-3.528	-0.759	7.909	0	0.853
16.03	-5.505	-11.6	-3.408	0	14.99	8.305	8.966
0	-7.445	8.087	-5.829	-1.474	13.37	0	0
0	-3.455	23.99	0	5.132	0	6.179	0
-0.696	4.11	-10.22	3.513	0	-5.706	-6.784	1.553

```
%Initialization
Y=zeros(n1,n2);
delta = n1*n2/m;
tau = 250;

%Iterations
vec = zeros(500,1);
for i = 1:500
    [U, S, V] = svd(Y);
    S_t = (S-tau);
    S_t(S_t<0) = 0;
    Z = U*S_t*V';
    P = X-Z;
    P(A) = 0;
    Y0 = Y;
    Y = Y0 + delta*P;
    vec(i) = sum(sum((Y-Y0).^2));
    err(i)=sum(sum((X0-Z).^2))/sum(sum((X0).^2));
end

% plot the results
figure;plot(vec);
figure;plot((err));
figure;
Ar=reshape(A, n1*n2,1);
Xr=reshape(X0, n1*n2,1);Xr=Xr(Ar);
Zr=reshape(Z, n1*n2,1);Zr=Zr(Ar);
subplot(2,1,1);plot(Xr);hold on;plot(Zr,'r');
subplot(2,1,2);plot(Xr-Zr);
```

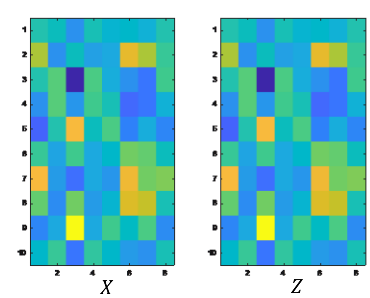


$$\begin{cases} Z^k = S_\tau(Y^{k-1}) \\ Y^k = Y^{k-1} + \delta_k P_\Omega(X - Z^k) \end{cases}$$





$$||X - Z||_2^2 = 4.6261e-09$$



References

[1] Jian-feng Cai, Emmanuel J. Candes, And Zuowei Shen. A Singular Value Thresholding Algorithm For Matrix Completion. Technical Report.

[2] Emmanuel J. Cand`es and Benjamin Recht. Exact Matrix Completion via Convex Optimization.

[3] Trevor Hastie, Rahul Mazumder, Jason D. Lee, Reza Zadeh, Matrix Completion and Low-Rank SVD via Fast Alternating Least Squares.

Topics on High-Dimensional Data Analytics

Regularization Applications

Kamran Paynabar, Ph.D.

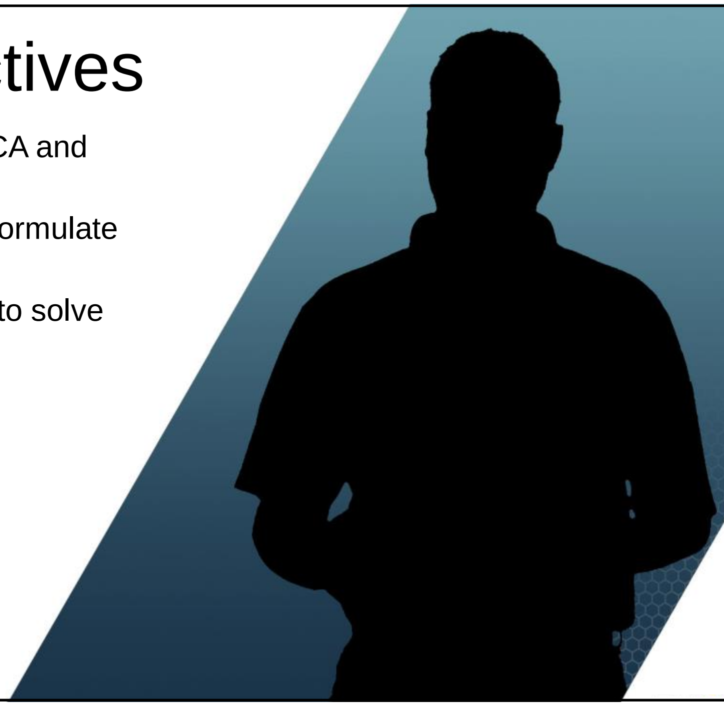
Associate Professor

School of Industrial and Systems Engineering

Robust PCA

Learning Objectives

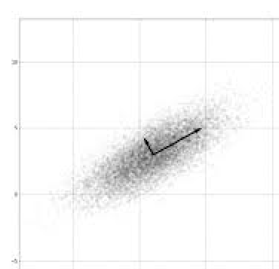
- Explain difference between PCA and Robust PCA.
- Use nuclear regularization to formulate the Robust PCA problem.
- Apply optimization algorithms to solve the Robust PCA problem.



Traditional PCA

- Principal Component Analysis reduces data dimensions by finding a low-rank representation of data.
- Suppose $X = [x_1\ x_2 \dots x_p]$ is the centered matrix data.
- A low-rank representation of X is given by $X = L + E$; where L is a low-rank matrix and E is noise.
- The low-rank subspace L can be found by solving

$$\arg\min_{L} \|X - L\|_F^2 \quad s.t. \operatorname{rank}(L) \leq k$$

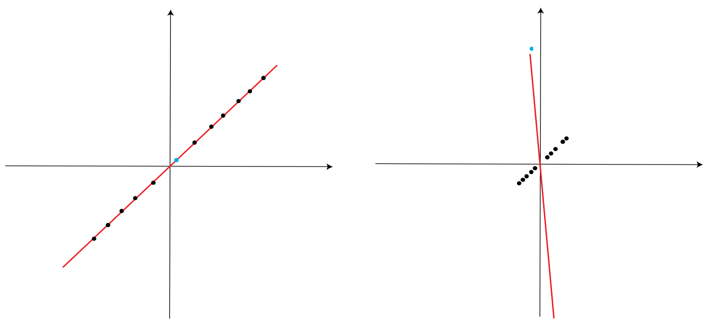


- This problem can be solved via truncated SVD:

$$L = U\Sigma V' = \sum_{i \leq k} \sigma_i \mathbf{u}_i \mathbf{v}_i'$$

PCA in Presence of Outliers

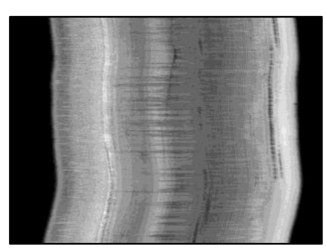
- Similar to regression, PCA is highly sensitive to outliers.



- Assuming outliers are sparse, Robust PCA can retrieve the correct low-rank structure

Robust PCA Applications

- Face recognition
- Anomaly detection
- Denoising
- Text mining
- Web mining
- Image and video repair
- Video Surveillance



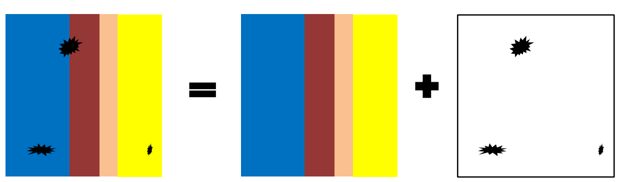
Robust PCA

$$M = L + S$$

- M is the observed data matrix
- L is a low-rank matrix (to be estimated)
- S is a matrix of sparse outliers (to be estimated)

By finding L and S, we can retrieve low-dimensional linear structure from non-ideal observations.

Robust PCA Formulation



$$M = L + S$$

- L is low-rank
- S is sparse
- L and S could be found by solving

$$\min\{\mathrm{rank}(L) + \lambda \|S\|_0\}$$

subject to: $M = L + S$

$\mathrm{rank}(L) = \#\{\sigma(L) \neq 0\}$ $\|S\|_0 = \#\{S_{ij} \neq 0\}$

Intractable!!!!

Robust PCA Formulation

$$\min\{\operatorname{rank}(L) + \lambda \|S\|_0\}$$

subject to: $M = L + S$

$\operatorname{rank}(L) = \#\{\sigma(L) \neq 0\}$

$\|S\|_0 = \#\{S_{ij} \neq 0\}$

Convex Relaxation

$$\|L\|_* = \sum_i \sigma(L)$$

Nuclear norm: sum of singular values

$$\|S\|_1 = \sum_{i,j} |S_{ij}|$$

L_1 norm: sum of absolute values

This is also known as Principal Component Pursuit (PCP)

See Chandrasekaran, Sanghavi, Parrilo, Willsky ('09)

Solving RPCA

- Review of Augmented Langrangian Multipliers Method:

$$\min f(X), \quad \text{subject to} \quad h(X) = 0,$$

$$L(X, Y, \mu) = f(X) + \langle Y, h(X) \rangle + \frac{\mu}{2}\|h(X)\|_F^2$$

(General Method of Augmented Lagrange Multiplier)

```
μ ≥ 1.
while not converged do
    Solve X_{k+1} = argmin_X L(X, Y_k, μ_k).
    Y_{k+1} = Y_k + μ_k h(X_{k+1});
    Update μ_k to μ_{k+1}.
end while
Output: X_k.
```

Solving RPCA

- Augmented Langrangian Multiplier form:

$$l(L,S,Y) = \|L\|_* + \lambda\|S\|_1 + \langle Y, M-L-S\rangle + \frac{\mu}{2}\|M-L-S\|_F^2.$$

$$l(L,S,Y;\mu) = \|L\|_* + \lambda\|S\|_1 + \frac{\mu}{2}\left\|M-L-S+\frac{Y}{\mu}\right\|_F^2 + \frac{\mu}{2}\left\|\frac{Y}{\mu}\right\|_F^2$$

Main Idea:

- Given S and Y, Update L

$$\arg\min_L \|L\|_* + \frac{\mu}{2}\left\|M-L-S+\frac{Y}{\mu}\right\|_F^2 \qquad X = M - S + \frac{Y}{\mu} \qquad L = D_{1/\mu}(X) = US_{1/\mu}(\Sigma)V^T$$

- Given L and Y, Update S

$$\arg\min_S \lambda\|S\|_1 + \frac{\mu}{2}\left\|M-L-S+\frac{Y}{\mu}\right\|_F^2 \qquad S_{ij} = S_{\frac{\lambda}{\mu}}(X) = \mathrm{sgn}(X)\max(|X| - \frac{\lambda}{\mu}, 0)$$

- Given L and S, Update Y

$$Y_{k+1} = Y_k + \mu(M-L-S)$$

- Iteration until Convergence

Solving RPCA - Algorithm

$$l(L,S,Y) = \|L\|_* + \lambda\|S\|_1 + \langle Y, M-L-S\rangle + \frac{\mu}{2}\|M-L-S\|_F^2.$$

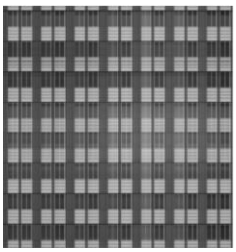
- Algorithm:

initialize: $S_0 = Y_0 = 0, \mu > 0.$
while not converged **do**
  compute $L_{k+1} = \mathcal{D}_\mu(M - S_k - \mu^{-1}Y_k)$; (soft-thresholding on singular values by $\frac{1}{\mu}$)
  compute $S_{k+1} = \mathcal{S}_{\lambda\mu}(M - L_{k+1} + \mu^{-1}Y_k)$; (soft-thresholding scalar entries by $\frac{\lambda}{\mu}$)
  compute $Y_{k+1} = Y_k + \mu(M - L_{k+1} - S_{k+1})$;
end while
output: L, S.

Example

Input

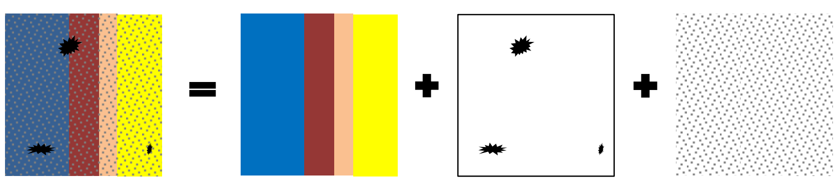
Output



```
D = double(rgb2gray(imread('building.png')));
lambda=1e-2;
[m n] = size(D); tol = 1e-7; maxIter = 1000;
% Initialize A,E,Y,u
Y = D; norm_two = norm(Y); norm_inf = norm( Y(:), inf) / lambda;
Y = Y / norm_inf;
A_hat = zeros( m, n); E_hat = zeros( m, n);
mu = 1.25/norm_two; mu_bar = mu * 1e7; rho = 1.5;        % this one can be tuned
d_norm = norm(D, 'fro');
iter = 0; total_svd = 0; converged = false; stopCriterion = 1;
while ~converged
   iter = iter + 1;
   temp_T = D - A_hat + (1/mu)*Y;
   E_hat = max(temp_T - lambda/mu, 0);
   E_hat = E_hat+min(temp_T + lambda/mu, 0);
   [U S V] = svd(D - E_hat + (1/mu)*Y, 'econ');
   diagS = diag(S);
   svp = length(find(diagS > 1/mu));
   A_hat = U(:, 1:svp) * diag(diagS(1:svp) - 1/mu) * V(:, 1:svp)';
   total_svd = total_svd + 1;
   Z = D - A_hat - E_hat;
   Y = Y + mu*Z;
   mu = min(mu*rho, mu_bar);
   % stop Criterion
   stopCriterion = norm(Z, 'fro') / d_norm;
   if stopCriterion < tol
      converged = true;
   end
end
```

Liang, Ren, Zhang, and Ma, ECCV 2012

Robust PCA for Noisy Data



$$M = L + S + E$$

- L is low-rank, S is sparse, E is noise
- L and S could be found by solving

$$\min\{\|L\|_* + \lambda\|S\|_1\}$$

subject to: $\|M - (L+S)\|_F < \delta$

References

[1] Martin J Wainwright. Structured regularizers for high-dimensional prob- lems: Statistical and computational issues. Annual Review of Statistics and Its Application, 1:233–253, 2014.

[2] Sahand Negahban, Bin Yu, Martin J Wainwright, and Pradeep K Raviku- mar. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. In Advances in Neural Information Processing Systems, pages 1348–1356, 2009.

[3] Sparse Representation and Low-Rank Representation for Biometrics -- Theory, Algorithms, and Applications, ICB 2013 Short Course

[4] John Wright, Arvind Ganesh, Shankar Rao, Yigang Peng, and Yi Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In Advances in neural information pro- cessing systems, pages 2080–2088, 2009.

[5] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust prin- cipal component analysis? Journal of the ACM (JACM), 58(3):11, 2011.

Topics on High-Dimensional Data Analytics

Regularization Applications

Kamran Paynabar, Ph.D.

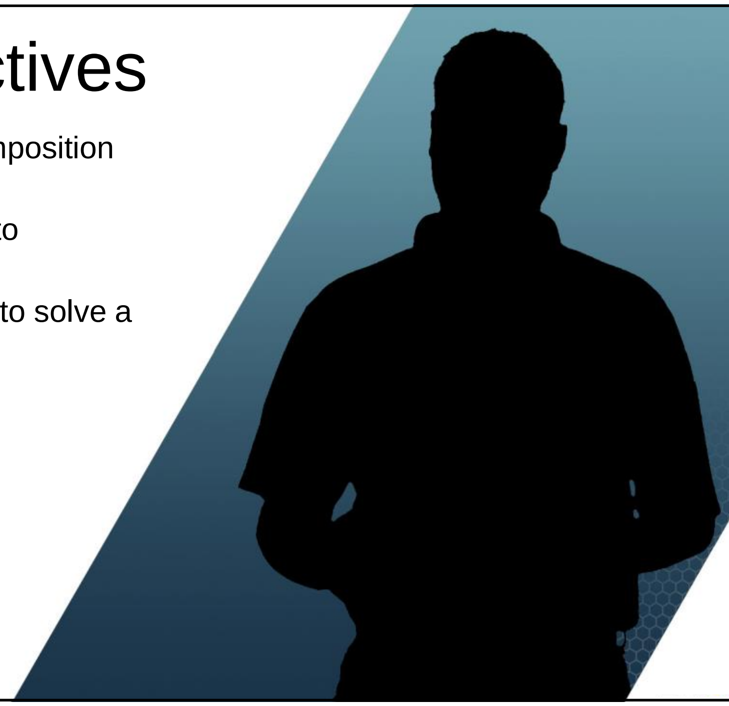
Associate Professor

School of Industrial and Systems Engineering

Sparse Smooth Decomposition

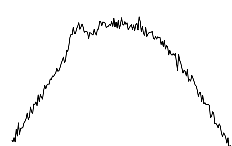
Learning Objectives

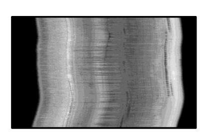
- Define Smooth Sparse Decomposition (SSD)
- Use L1 and L2 regularization to formulate the SSD problem.
- Apply optimization algorithms to solve a the SSD problem.

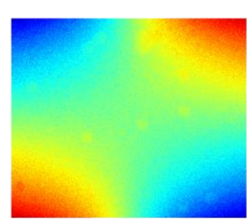


Motivation

- In practice, high-dimensional data
  - are noisy;
  - often have functional structure that makes them smooth
  - sometimes contain sparse features.
- Decomposition of an observation into these three elements have various applications, including:
  - Denoising
  - Sparse feature learning
  - Image background retrieval
  - Function estimation in presence of outliers and noise
  - Anomaly detection

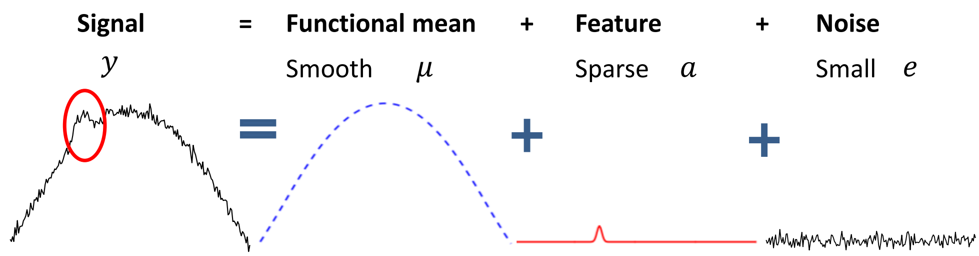






Sparse Smooth Decomposition (SSD)

- Objective: To decompose an HD noisy observation into functional mean, sparse features, and noise (Yan et al., 2017): $y = \mu + a + e$



Assumptions

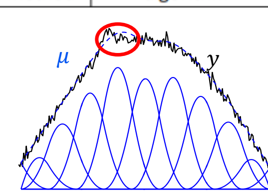
- Functional mean is smooth.
- Features are sparse features and exhibit a different structure from the functional mean

Two-step Approach: Smoothing + Detection

Methods	Edge Based	Thresholding Based	Region Based
Criterion	Look for discontinuity	Look for largest	Look for similarity
Output	Edge	Region	Region
Preprocessing	Smoothing	Smoothing	Smoothing, need a seed point to start
Postprocessing	Close the Edge to get boundary	Clean small areas	Clean small areas
Algorithms	Sobel Edge Detection[1] Jump Regression[2]	Ostu's Glocal method[3] Nick's Local method[4]	Region Growing algorithm[5]

Some issues:

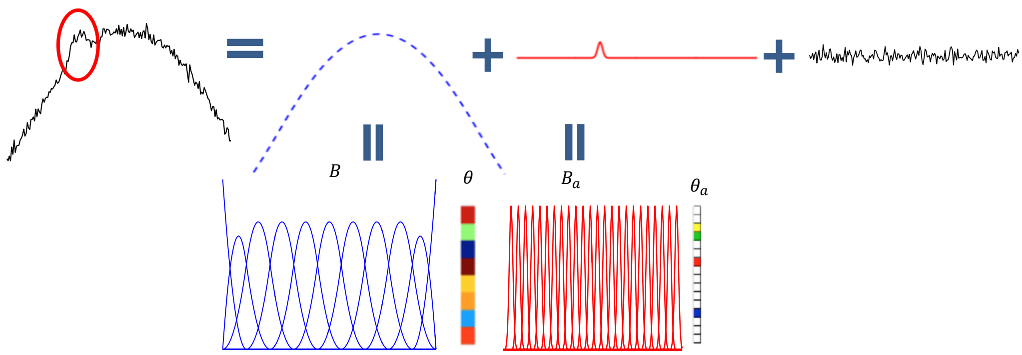
- Smoothing blurs the boundaries of defect regions;
- Smoothing is inaccurate in presence of anomaly;
- Often, a post-processing task is required to connect detected boundaries;



SSD Formulation

Let's begin with 1D case: $y = \mu + a + e$ → $y = B\theta + B_a\theta_a + e$

$\mathrm{argmin}_{\theta,\theta_a}\ \lambda\theta' R\theta\ + \gamma||\theta_a||_1\ + ||e||^2, s.t.\ y = B\theta + B_a\theta_a + e$



R is the roughness matrix that can be defined by $R = D^T D$ where D is the first difference matrix

SSD via Iterative Thresholding Algorithm

$$\mathrm{argmin}_{\theta,\theta_a}\ \lambda\theta' R\theta\ +\ \gamma||\theta_a||_1\ +\ ||e||^2, s.t.\ y = B\theta + B_a\theta_a + e$$

- Propose an optimization algorithm based on ITA (Daubechies, et al, 2004)

Iterative Thresholding Algorithm

In the k^{th} iteration, update $\mu^{(k)}$ and $\theta_a^{(k)}$ by

$\mu^{(k)} = H(y - B_a\theta_a^{(k-1)})$, $H = B\,(B'B + \lambda R)^{-1}B'$ is the projection matrix

$\theta_a^{(k)} = T_{\tau^{(k)}}(\theta_a^{(k-1)} - c^{(k)}B_a'(B_a\theta_a^{(k-1)} + \mu^{(k)} - y))$

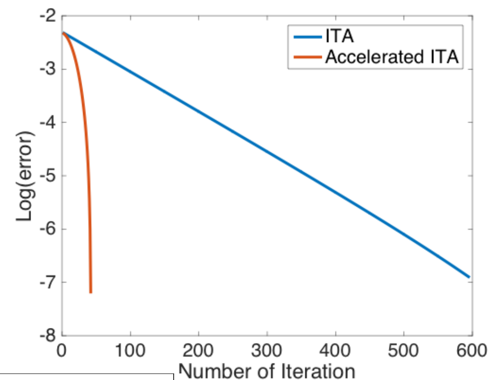
$c^{(k)} = \frac{2}{L}; \tau^{(k)} = \frac{\gamma}{L}$

- $p = 1$, convex optimization
  - $T(\cdot)$ is the soft-thresholding operator.

Accelerated Algorithm for SSD

Nesterov's acceleration (Beck and Teboulle, 2009)

- $x_a^{(k)} = T_{\tau^{(k)}}\left(\theta_a^{(k-1)}, c^{(k)}, B_a, \mu^{(k)}, y\right)$
- $t_{k+1} = (1 + \sqrt{1 + 4t_k^2})/2$
- $\theta_a^{(k+1)} = x_a^{(k)} + \frac{t_k - 1}{t_{k+1}}\left(x_a^{(k)} - x_a^{(k-1)}\right)$



$$\begin{cases} y_B^{(k)} = H(y - B_S\theta_S^{(k-1)}) \\ \theta_S^{(k)} = \text{softthreshold}(x^{(k-1)} + \frac{2}{L}B_S'(y - B_S\theta_S^{(k-1)} - y_B^{(k)}), \frac{\gamma}{L}) \end{cases}$$

Convergence Rate & Error Bound of ITA Algorithm for SSD

Restricted isometry property

- δ_{3s+1} is RIP constant which satisfies (for all s-sparse vector θ_{a1}, θ_{a2})

$(1 - \delta_{3s+1})||\theta_{a1} - \theta_{a2}||_2^2 \leq ||(I - H)(B_a\theta_{a1} - B_a\theta_{a2})||_2^2 \leq (1 + \delta_{3s+1})||\theta_{a1} - \theta_{a2}||_2^2$

Theorem

If $\rho_s = \boxed{L\delta_{3s+1}((I - H)B_a) < 1}$, *the anomaly estimation in the* k^{th} *iteration of the algorithm with* $\lambda = 0$ *is bounded by*

$$\|\theta_a^{(k)} - \theta_a^*\|_2 \leq \boxed{(\rho_s)^k \|\theta_a^{(0)} - \theta_a^*\|_2} + \boxed{\frac{L}{1 - \rho_s}\|B_a^T(I - H)\epsilon\|_2}$$

Linear convergence rate

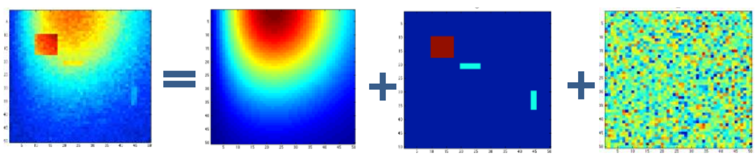
Estimation error

- Noiseless case: $||\epsilon||_2 = 0$, perfect recovery
- Noisy case: detect anomaly with $B_a \perp \epsilon$

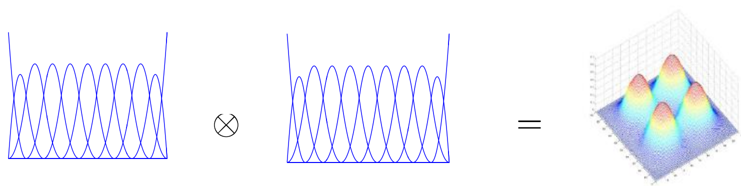
Generalization of SSD to 2D Case (Images)

Image = **Background** + **Defect** + **Noise**

Smooth Sparse Small



$$\underset{\theta,\theta_S}{argmin} \|\tilde{e}\|^2 + \theta' R\theta + \gamma|\theta_S|_1, \ s.t.\ y = (B_1 \otimes B_2)\theta + (B_{S,1} \otimes B_{S,2})\theta_S + \tilde{e}$$



Generalization of SSD to 2D Case (Images)

$$\underset{\theta,\theta_S}{argmin} \|\tilde{e}\|^2 + \theta' R\theta + \gamma|\theta_S|_1, \ s.t.\ y = (B_1 \otimes B_2)\theta + (B_{S,1} \otimes B_{S,2})\theta_S + \tilde{e}$$

y: $n_1 n_2 \times 1$

$H : k_1 k_2 \times k_1 k_2$

$$\boxed{\begin{cases} y_B^{(k)} = H(y - B_S\theta_S^{(k-1)}) \\ \theta_S^{(k)} = \text{softthreshold}(x^{(k-1)} + \frac{2}{L}B_S'(y - B_S\theta_S^{(k-1)} - y_B^{(k)}), \frac{\gamma}{L}) \end{cases}}$$

APG algorithm

Complexity $\boxed{O(k_1^3 k_2^3 + 6n_1^2 n_2^2 k_1 k_2)}$

Define $R = \lambda_1 B_2' B_2 \otimes D_1' D_1 + \lambda_2 D_2' D_2 \otimes B_1' B_1 + \lambda_1\lambda_2 D_2' D_2 \otimes D_1' D_1$

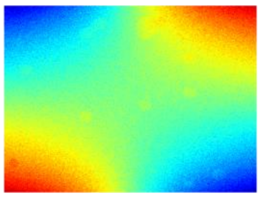
$H = H_2 \otimes H_1$ $\quad H_i = B_i(B_i' B_i + \lambda_i D_i' D_i)^{-1} B_i'$, $i = 1, 2$ size $n_i \times n_i$

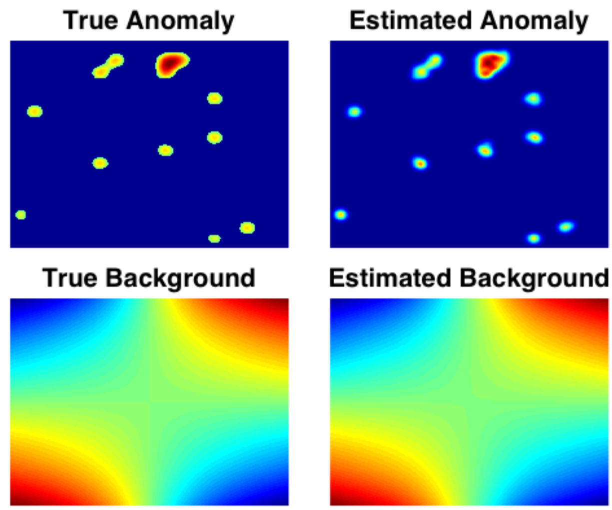
APG algorithm $\boxed{\begin{cases} Y_B^{(k)} = H_1(Y - B_{S_1}\Theta_S^{(k-1)} B_{S_2}')H_2 \\ \Theta_S^{(k)} = \text{softthreshold}(X^{(k-1)} + \frac{2}{L}B_{S_1}'(Y - B_{S_1}X^{(k-1)}B_{S_2}' - Y_B^{(k)})B_{S_2}, \frac{\gamma}{L}) \end{cases}}$

Complexity $\boxed{O(k_1^3 + k_2^3 + 6n_1^2 k_1 + 6n_2^2 k_2)}$

Example

Simulated Image (Y)





Background MSE: 1×10^{-5}
Anomaly FPR: 1%, FNR: 1%

B_a: Quartic kernel

Matlab file: SSD.m

References

[1] Yan, H., Paynabar, K., Shi, J., (2017) Anomaly Detection in Images With Smooth Background via Smooth-Sparse Decomposition, *Technomtrics,* Vol 59, Issue 1.

Topics on High-Dimensional Data Analytics

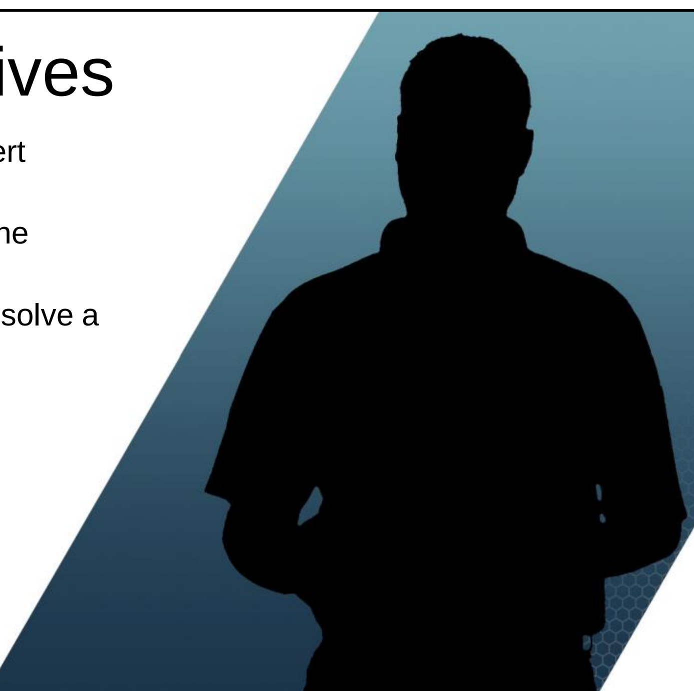
Regularization Applications

Kamran Paynabar, Ph.D.
Associate Professor
School of Industrial and Systems Engineering

RKHS Ridge Kernel Regression

Learning Objectives

- Define Reproducing Kernel Hilbert Space
- Use regularization to formulate the ridge regression problem.
- Apply optimization algorithms to solve a the ridge regression problem.



Ridge Regression

Assuming the observations are centered, Ridge estimates can be computed by

$$\arg\min \frac{1}{n}\sum_{i=1}^{n}(y_i - x_i{}^T\theta)^2 + \lambda\|\theta\|^2$$

$$\hat{\theta} = (X^TX + \lambda I)^{-1}X^Ty$$

- λ≥0 is the tuning parameter controls the amount of shrinkage
- λ is chosen based on some prediction criteria (MSE) using CV or independent validation data set

Ridge Regression

Evaluate ridge regression for a new test point *x*:

$$\hat{f}\,(x) = x^T\hat{\theta} = x^T(X^TX + \lambda I)^{-1}X^Ty = \underbrace{x^TX^T}(\underbrace{X^TX} + \lambda I)^{-1}y = y^T(XX^T + \lambda I)^{-1}Xx$$

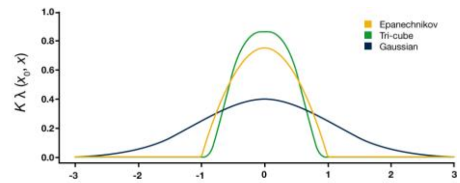
Only depends on the inner product of predictors

The equality above is true because of the Matrix inversion lemma:

$$X^T(XX^T + \lambda I)^{-1} = (X^TX + \lambda I)^{-1}X^T$$

Kernel as Inner Product

- $k(x, y)$ equivalent to first compute feature $\phi(x)$, and then perform inner product $k(x, y) = \phi(x)^\top \phi(y)$



- A dataset $D = \{x_1, x_2, x_3 \ldots x_n\}$

- Compute pairwise kernel function $k(x_i, x_j)$ and form a $n \times n$ kernel matrix (Gram matrix)
  - $K = \begin{pmatrix} k(x_1, x_2) & \ldots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \ldots & k(x_n, x_n) \end{pmatrix}$

Kernel Ridge Regression

$\hat{f}(x) = \hat{\theta}^T x = y^T (XX^T + \lambda I)^{-1} Xx$ only depends on inner products

$$XX^T = \begin{pmatrix} x_1 x_1^T & \ldots & x_1 x_n^T \\ \vdots & \ddots & \vdots \\ x_n x_1^T & \ldots & x_n x_n^T \end{pmatrix} \qquad Xx = \begin{pmatrix} x_1 x \\ \vdots \\ x_n x \end{pmatrix}$$

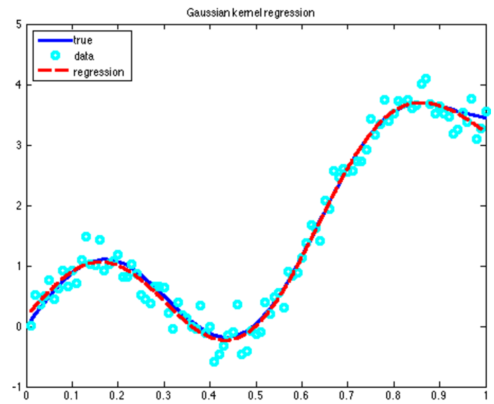
x_i is the row vector corresponding to the i^{th} row of the matrix X.

Kernel ridge regression: replace the inner products by a kernel function

$$XX^T \to K = [k(x_i, x_j)]_{n \times n}$$
$$Xx \to k_x = [k(x_i, x)]_{n \times 1}$$
$$\hat{f}(x) = \hat{\theta}^T x = y^T (K + \lambda I)^{-1} k_x$$

Kernel Ridge Regression - Example

```
clc
clear all
Xtrain = (1:100)/100;
Yt = sin(Xtrain*10)+(Xtrain*2).^2;
Ytrain = Yt + 0.2*randn(1,100);
N = 1000;
Xtest = linspace(min(Xtrain),max(Xtrain),N);
Xtrain=Xtrain(:); Ytrain=Ytrain(:); n =
length(Xtrain);Xtest=Xtest(:);

lambda = 0.04;
c = 0.04;

kernel1 = exp(-dist(Xtrain').^2 ./ (2*c));
kernel2 = exp(-pdist2(Xtrain, Xtest).^2 ./ (2*c));
yhatRBF = Ytrain' * ((kernel1 + lambda *
eye(size(kernel1))) \ kernel2);
```

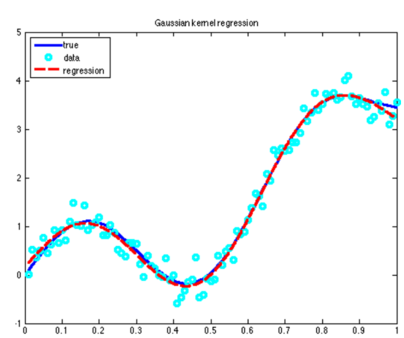


Another Look to Ridge Regression

- Suppose a set of observation, $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)$ are given. To estimate a regression function, we minimize the following empirical loss function:

$$\arg\min \frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(x_i)\right)^2 + \lambda \|f\|_{\mathcal{H}}^2$$

Where $\mathcal{H}$ is the Reproducible Kernel Hilbert Space.



Loosely speaking A **Hilbert space** is a (possibly) infinite dimensional linear space established with a dot product.

Representer Theorem

- The minimizer over the RKHS $\mathcal{H}$, of the regularized empirical loss function:

$$\arg\min_f \frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(x_i)\right)^2 + \lambda\|f\|_{\mathcal{H}}^2$$

Can be represented by the expression

$$f^\lambda(x) = \sum_{i=1}^{n}\alpha_i K(x_i, x)$$

- Therefore, minimizing over the possibly infinite dimensional Hilbert space boils down to minimizing over $\mathbb{R}^n$.

RKHS

- Loosely speaking, a **Hilbert space** is a (possibly) infinite dimensional linear space established with a dot product.
- For example, Square integrable functions $L_2[a, b]$ (i.e., the integral of the squared function is finite), is Hilbert space and the norm is calculated by

$$\langle f, g\rangle = \int_a^b f(x)g(x)dx$$

- $k(\cdot,\cdot)$ is a Reproducing Kernel Hilbert Space (RKHS) of $\mathcal{H}$ if $\forall f \in \mathcal{H}$,

$$f(x) = \langle k(x,\cdot), f(\cdot)\rangle.$$

Kernel Ridge Regression

According to the Representer Theorem $\hat{f} = \underset{f \in \mathcal{H}}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - f(x^{(i)}))^2 + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2$

Solved by: $\hat{f}(\cdot) = \sum_{j=1}^{n} \alpha_j \mathbb{K}(\cdot, x^{(j)})$

$$\|\sum_{j=1} \alpha_j \mathbb{K}(\cdot, x^{(j)})\|_{\mathcal{H}}^2 = \sum_{i,j=1} \alpha_i \alpha_j \langle \mathbb{K}(\cdot, x^{(i)}), \mathbb{K}(\cdot, x^{(j)}) \rangle_{\mathcal{H}}$$
$$= \sum_{i,j=1}^{n} \alpha_i \alpha_j \mathbb{K}(x^{(i)}, x^{(j)}) = \sum_{i,j=1}^{n} \alpha_i \alpha_j K_{ij}$$
$$= \alpha^T \mathbf{K} \alpha$$

Thus,

$$\boxed{\hat{\alpha} = \underset{\alpha \in \mathbb{R}^n}{\operatorname{argmin}} \frac{1}{2} \|y - \mathbf{K}\alpha\|_2^2 + \frac{\lambda}{2} \alpha^T \mathbf{K} \alpha}$$

$$\boxed{\hat{\alpha} = (\mathbf{K} + \lambda \mathbf{I})^{-1} y}$$