Sparse Methods

Sanjiv Kumar, Google Research, NY

EECS-6898, Columbia University - Fall, 2010

What is Sparsity?

When a data item such as a vector or a matrix has only a few nonzero entries

- Sparsity in features x
- Sparsity in parameters α

What is Sparsity?

When a data item such as a vector or a matrix has only a few nonzero entries

- Sparsity in features x
- Sparsity in parameters α

Learning methods that utilize sparsity

- Kernel methods → want only a few coefficients nonzero in kernel sums
- Feature selection in linear methods using L_1 penalty e.g., Lasso
- Group feature selection e.g., Group Lasso
- Sparse Coding → Linear reconstruction of the data using a few elements from large dictionaries
- Sparse PCA and Sparse SVD
  - Find sparse vectors "closest" to the true singular vectors
- Many others…

Sparse methods lead to parsimonious (and interpretable) models in addition to being efficient for large scale learning

Sparse penalties

Total Loss $J(\alpha) = \sum_{i=1}^{n} l(x_i, y_i, \alpha) + R(\alpha)$

Lasso or L_1 $R(\alpha) = \lambda \|\alpha\|_1$ leads to sparse α

Elastic net $R(\alpha) = \lambda_1 \|\alpha\|_1 + \lambda_2 \|\alpha\|_2^2$ allows important correlated variables to be picked

Group Lasso $R(\alpha) = \lambda \sum_{l=1}^{L} \|\alpha_l\|_2$ note: no square in the norm
allows groups of variables to be sparse

Sparse Group Lasso $R(\alpha) = \lambda_1 \sum_{l=1}^{L} \|\alpha_l\|_2 + \lambda_2 \|\alpha\|_1$ allows elements within groups to be sparse in addition to sparse groups

Multitask Lasso $R(\alpha) = \lambda \sum_{l=1}^{L} \|\alpha_l\|_\infty$ encourages entire groups to have zero elements
more aggressive than group lasso

Lasso

Linear regression with L_1 penalty

$$\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} (y_i - \alpha^T x_i)^2 + \lambda \|\alpha\|_1 \qquad x, \alpha \in \Re^d$$

equivalent problem $\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} (y_i - \alpha^T x_i)^2 \text{ s.t.} \|\alpha\|_1 \leq C$

Lasso

Linear regression with L_1 penalty

$$\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} (y_i - \alpha^T x_i)^2 + \lambda \|\alpha\|_1 \qquad x, \alpha \in \Re^d$$

equivalent problem $$\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} (y_i - \alpha^T x_i)^2 \text{ s.t. } \|\alpha\|_1 \leq C$$

- Quadratic objective function with linear constraints → Quadratic Programming !
  - Does not scale well with problem size
- Can be solved using iterative (sub)gradient methods e.g., coordinate descent
- Leads to sparse α, higher C → less sparse α
- Related method: Forward Stagewise Procedure (Homotopy method)

Lasso

Linear regression with L_1 penalty

$$\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} (y_i - \alpha^T x_i)^2 + \lambda \|\alpha\|_1 \qquad x, \alpha \in \Re^d$$

equivalent problem $\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} (y_i - \alpha^T x_i)^2 \text{ s.t.} \|\alpha\|_1 \leq C$

- Quadratic objective function with linear constraints → Quadratic Programming !
  - Does not scale well with problem size
- Can be solved using iterative (sub)gradient methods e.g., coordinate descent
- Leads to sparse α, higher C → less sparse α
- Related method: Forward Stagewise Procedure (Homotopy method)
  - Suppose data X and prediction y are centered, and $\mu_t = X^T \alpha_t$
  - Given current estimate of α_t (starting with $\alpha_0 = 0$), compute residual $r_t = (y - \mu_t)$
  - Compute data correlation with the residual vector $c_t = X r_t$
  - Find the direction of greatest correlation and take a small step

$$j = \arg\max_{l}(c_{lt}) \Rightarrow \mu_{t+1} \leftarrow \mu_t + \varepsilon.\text{sgn}(c_{jt}).(X^j)^T$$

j^{th} row of X

Group Lasso

Linear regression with L_2 (non-squared) penalty

- Suppose the features can be divided into L groups
- Let X_l be the data submatrix and α_l be the parameter chunk corresponding to l^{th} group

$$\hat{\alpha} = \arg\min_{\alpha} \left\| y - \sum_{l=1}^{L} X_l^T \alpha_l \right\|_2^2 + \lambda \sum_{l=1}^{L} \|\alpha_l\|_2$$

Group Lasso

Linear regression with L_2 (non-squared) penalty

- Suppose the features can be divided into L groups
- Let X_l be the data submatrix and α_l be the parameter chunk corresponding to l^{th} group

$$\hat{\alpha} = \arg\min_{\alpha} \left\| y - \sum_{l=1}^{L} X_l^T \alpha_l \right\|_2^2 + \lambda \sum_{l=1}^{L} \|\alpha_l\|_2$$

Subgradient equation

$$-X_l\,(y - \sum_{l=1}^{L} X_l^T \alpha_l) + \lambda s_l = 0 \quad l = 1,\ldots,L$$

$$s_l = \alpha_l / \|\alpha_l\| \quad if\ \alpha_l \neq 0, \text{any vector with} \|s_l\|_2 < 1 \text{otherwise}$$

Group Lasso

Linear regression with L_2 (non-squared) penalty

- Suppose the features can be divided into L groups
- Let X_l be the data submatrix and α_l be the parameter chunk corresponding to l^{th} group

$$\hat{\alpha} = \arg\min_{\alpha} \left\| y - \sum_{l=1}^{L} X_l^T \alpha_l \right\|_2^2 + \lambda \sum_{l=1}^{L} \|\alpha_l\|_2$$

Subgradient equation

$$-X_l\,(y - \sum_{l=1}^{L} X_l^T \alpha_l) + \lambda s_l = 0 \quad l = 1,\ldots,L$$

$$s_l = \alpha_l / \|\alpha_l\| \quad if\ \alpha_l \neq 0, \text{any vector with} \|s_l\|_2 < 1 \text{ otherwise}$$

if $\left\| X_l\,(y - \sum_{k \neq l} X_k^T \alpha_k) \right\| < \lambda \Rightarrow \alpha_l = 0$

otherwise $\alpha_l = (X_l\,X_l^T + \lambda / \|\hat{\alpha}_l\|)^{-1} X_l\,r_l$ where $r_l = y - \sum_{k \neq l} X_k^T \alpha_k$

Group Lasso

Linear regression with L_2 (non-squared) penalty

- Suppose the features can be divided into L groups
- Let X_l be the data submatrix and α_l be the parameter chunk corresponding to l^{th} group

$$\hat{\alpha} = \arg\min_{\alpha} \left\| y - \sum_{l=1}^{L} X_l^T \alpha_l \right\|_2^2 + \lambda \sum_{l=1}^{L} \|\alpha_l\|_2$$

Subgradient equation

$$-X_l\,(y - \sum_{l=1}^{L} X_l^T \alpha_l) + \lambda s_l = 0 \quad l = 1,\ldots,L$$

$$s_l = \alpha_l / \|\alpha_l\| \quad \text{if } \alpha_l \neq 0, \text{any vector with} \|s_l\|_2 < 1 \text{ otherwise}$$

if $\left\| X_l\,(y - \sum_{k \neq l} X_k^T \alpha_k) \right\| < \lambda \Rightarrow \alpha_l = 0$

otherwise $\alpha_l = (X_l X_l^T + \lambda / \|\hat{\alpha}_l\|)^{-1} X_l r_l$ where $r_l = y - \sum_{k \neq l} X_k^T \alpha_k$

if $X_l X_l^T = I$ and $v_l = X_l r_l$

$$\boxed{\alpha_l = (1 - \lambda / \|v_l\|) v_l}$$

Block-coordinate descent

Multitask Penalty

Want to learn K parameter vectors, one for each task $k = 1,\ldots,K$

$$\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} l(x_i, \alpha) + \lambda \|\alpha\|_{1,\infty} \qquad \alpha \in \Re^{d \times K} \quad x \in \Re^{d}$$

$$\|\alpha\|_{1,\infty} = \sum_{j=1}^{d} \max_{k} |\alpha_{jk}|$$

sum of max absolute value in each row of α

Multitask Penalty

Want to learn K parameter vectors, one for each task $k = 1,\ldots,K$

$$\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} l(x_i, \alpha) + \lambda \|\alpha\|_{1,\infty} \qquad \alpha \in \Re^{d \times K} \quad x \in \Re^{d}$$

$$\|\alpha\|_{1,\infty} = \sum_{j=1}^{d} \max_{k} |\alpha_{jk}|$$

sum of max absolute value in each row of α

equivalent

$$\hat{\alpha} = \arg\min_{\alpha} \sum_{i=1}^{n} l(x_i, \alpha) \text{ s.t. } \|\alpha\|_{1,\infty} \leq C$$

Tends to make entire row of matrix α zero

Choice of loss function

Regression for k[th] task $l(x_i, y_i, \alpha_k) = \left\| y_i - \alpha_k^T x_i \right\|_2^2$

Classification for k[th] task $l(x_i, y_i, \alpha_k) = \max(0, 1 - y_i \alpha_k^T x_i)$

Can be solved directly using block coordinate descent or via projected subgradient

Projected Subgradient Method

Let's focus on classification case with hinge loss

$$\alpha_{t+1} = P_{\Omega}(\alpha_t - \eta_t g_t)$$

projection on convex set Ω: $\|\alpha\|_{1,\infty} \le C$

subgradient of $\max(0, 1 - y_i \alpha_k^T x_i)$

step size $\eta_0 / \sqrt{t}$

Projected Subgradient Method

Let's focus on classification case with hinge loss

$$\alpha_{t+1} = P_{\Omega}(\alpha_t - \eta_t g_t)$$

projection on convex set Ω: $\|\alpha\|_{1,\infty} \le C$

subgradient of $\max(0, 1 - y_i \alpha_k^T x_i)$

step size $\eta_0 / \sqrt{t}$

Subgradient for k[th] task $g_t^k = \sum_{i: x_i \in k, y_i \alpha_k^T x_i < 1} y_i x_i$

Projection on constraint set $\min_{\alpha \in \Omega} \|\alpha - \alpha'\|^2 = \min_{\alpha \in \Omega} \sum_{jk} (\alpha_{jk} - \alpha'_{jk})^2$

How to efficiently do projection of a matrix to a $L_{1,\infty}$ ball ?

Efficient Projection

Equivalent projection problem

- assuming all entries of α' are positive (can be relaxed)

$$\min_{\alpha,\mu} \sum_{jk} (\alpha_{jk} - \alpha'_{jk})^2$$

$$\text{s.t. } \forall j,k \quad \alpha_{jk} \le \mu_j$$

$$\sum_j \mu_j = C$$

$$\forall j \quad \mu_j \ge 0$$

Efficient Projection

Equivalent projection problem

- assuming all entries of α' are positive (can be relaxed)

$$\min_{\alpha,\mu} \sum_{jk} (\alpha_{jk} - \alpha'_{jk})^2$$

$$\text{s.t. } \forall j,k \quad \alpha_{jk} \le \mu_j$$

$$\sum_j \mu_j = C$$

$$\forall j \quad \mu_j \ge 0$$

Lagrangian

$$L = \sum_{jk} (\alpha_{jk} - \alpha'_{jk})^2 + \sum_{jk} \rho_{jk} (\alpha_{jk} - \mu) + \theta(\sum_j \mu_j - C) - \sum_{jk} \beta_{jk} \alpha_{jk} - \sum_j \gamma_j \mu_j$$

Efficient Projection

Equivalent projection problem

- assuming all entries of α' are positive (can be relaxed)

$$\min_{\alpha,\mu} \sum_{jk} (\alpha_{jk} - \alpha'_{jk})^2$$

$$\text{s.t. } \forall j,k \quad \alpha_{jk} \le \mu_j$$

$$\sum_j \mu_j = C$$

$$\forall j \quad \mu_j \ge 0$$

Lagrangian

$$L = \sum_{jk} (\alpha_{jk} - \alpha'_{jk})^2 + \sum_{jk} \rho_{jk} (\alpha_{jk} - \mu) + \theta(\sum_j \mu_j - C) - \sum_{jk} \beta_{jk} \alpha_{jk} - \sum_j \gamma_j \mu_j$$

Suppose $\hat{\mu}$ is the optimal value of μ

$$\alpha'_{jk} \ge \hat{\mu}_j \Rightarrow \alpha_{jk} = \mu_j$$

$$\alpha'_{jk} \le \hat{\mu}_j \Rightarrow \alpha_{jk} = \alpha'_{jk}$$

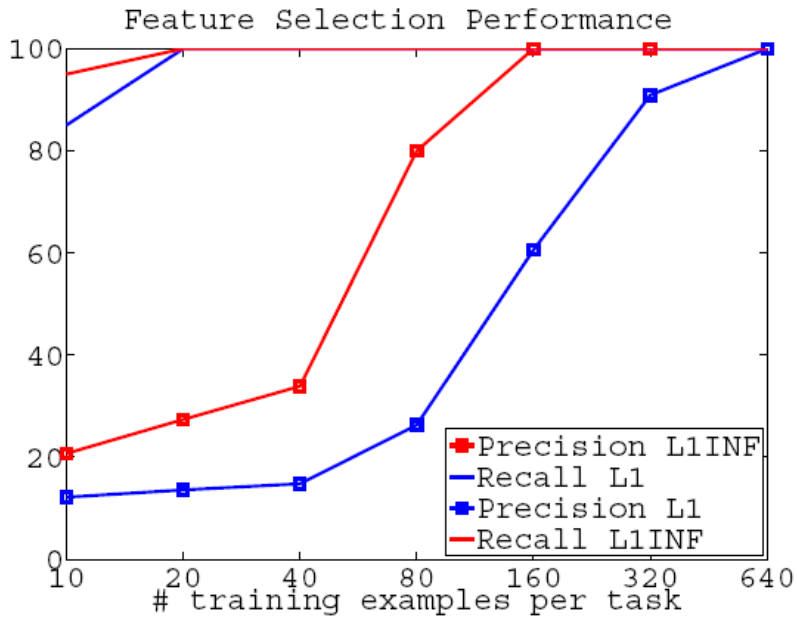
$$\hat{\mu}_j = 0 \Rightarrow \alpha_{jk} = 0$$

Optimal value of μ can be obtained in $O(dKlogdK)$ time !

Experiment

Synthetic Data, only 10% features relevant for any task prediction

$K = 60,\ d = 200$



Quattoni et al. [14]

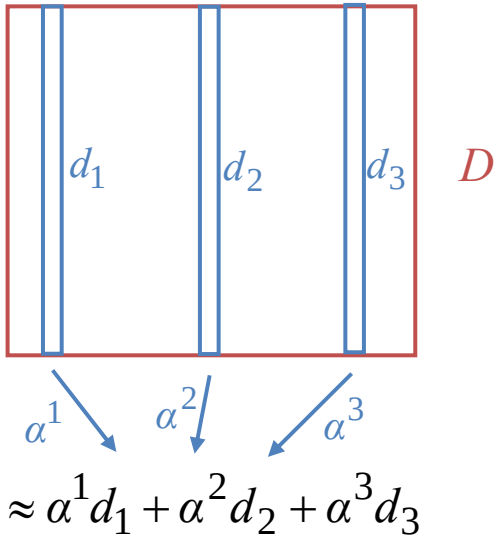
Sparse Coding

Linear reconstruction of data using a few elements of a dictionary

- A data vector $x \in \Re^d$ is reconstructed using a dictionary D containing k elements (sometimes called "basis" vectors or "atoms")

$$x \approx \sum_{j=1}^{k} \alpha^j d_j = D\alpha \qquad D \in \Re^{d \times k}, \alpha \in \Re^k$$

number of nonzero elements $\|\alpha\|_0 << k$



Sparse Coding

Linear reconstruction of data using a few elements of a dictionary

- A data vector $x \in \Re^d$ is reconstructed using a dictionary D containing k elements (sometimes called "basis" vectors or "atoms")

$$x \approx \sum_{j=1}^{k} \alpha^j d_j = D\alpha \qquad D \in \Re^{d \times k}, \alpha \in \Re^k$$

number of nonzero elements $\|\alpha\|_0 << k$

Properties

- The dictionary elements usually not orthogonal (unlike PCA)
- Dictionary may be overcomplete, i.e., number of elements are more than the data dimensionality d → elements are not linearly independent
- Learned dictionaries usually lead to more more compact representation than predefined ones e.g., based on wavelets

Sparse Coding

Linear reconstruction of data using a few elements of a dictionary

- A data vector $x \in \Re^d$ is reconstructed using a dictionary D containing k elements (sometimes called "basis" vectors or "atoms")

$$x \approx \sum_{j=1}^{k} \alpha^j d_j = D\alpha \qquad D \in \Re^{d \times k}, \alpha \in \Re^k$$

number of nonzero elements $\|\alpha\|_0 << k$

Properties

- The dictionary elements usually not orthogonal (unlike PCA)
- Dictionary may be overcomplete, i.e., number of elements are more than the data dimensionality d → elements are not linearly independent
- Learned dictionaries usually lead to more more compact representation than predefined ones e.g., based on wavelets

Learning Problem

- Offline: Given a dataset X, learn dictionary D
- Online: Given a dictionary D and input vector x, learn coefficients α

Sparse Coding Formulation

Given a training set $X = [x_1, \ldots x_n], \; x_i \in \Re^d$ learn dictionary D

$$X \approx D\alpha \qquad X \in \Re^{d \times n}, D \in \Re^{d \times k}, \alpha \in \Re^{k \times n}$$

Total penalized loss

$$\min_{D,\alpha} \sum_{i=1}^{n} [(1/2)\|x_i - D\alpha_i\|_2^2 + \lambda \underbrace{\|\alpha_i\|_1}_{L_1 \text{ penalty}}] \qquad \alpha_i \in \Re^k$$

Sparse Coding Formulation

Given a training set $X = [x_1, \ldots x_n],\ x_i \in \Re^d$ learn dictionary D

$$X \approx D\alpha \qquad X \in \Re^{d \times n}, D \in \Re^{d \times k}, \alpha \in \Re^{k \times n}$$

Total penalized loss

$$\min_{D,\alpha} \sum_{i=1}^{n} [(1/2)\|x_i - D\alpha_i\|_2^2 + \lambda \underbrace{\|\alpha_i\|_1}_{L_1 \text{ penalty}}] \quad \alpha_i \in \Re^k$$

- Given D, obtaining best α equivalent to solving Lasso
- Nonconvex in both D and α but convex in one if the other is fixed
- Since scales of D and α are arbitrary, dictionary elements are constrained

$$C = \{D \in \Re^{d \times k},\ \text{s.t. } d_j^T d_j \leq 1 \quad \forall\ j = 1, \ldots, k\}$$

Sparse Coding Formulation

Given a training set $X = [x_1, \ldots x_n]$, $x_i \in \Re^d$ learn dictionary D

$$X \approx D\alpha \qquad X \in \Re^{d \times n}, D \in \Re^{d \times k}, \alpha \in \Re^{k \times n}$$

Total penalized loss

$$\min_{D,\alpha} \sum_{i=1}^{n} [(1/2)\|x_i - D\alpha_i\|_2^2 + \lambda \underbrace{\|\alpha_i\|_1}_{L_1 \text{ penalty}}] \quad \alpha_i \in \Re^k$$

- Given D, obtaining best α equivalent to solving Lasso
- Nonconvex in both D and α but convex in one if the other is fixed
- Since scales of D and α are arbitrary, dictionary elements are constrained

$$C = \{D \in \Re^{d \times k}, \text{ s.t. } d_j^T d_j \leq 1 \quad \forall\ j = 1, \ldots, k\}$$

- Matrix form $\min_{D \in C, \alpha} [(1/2)\|X - D\alpha\|_2^2 + \lambda \|\alpha\|_{1,1}]$ $\quad \alpha_{1,1} = \sum_{ij} |\alpha_{ij}|$
- Usually solved via alternate minimization of D and α
  - For large-scale problems (i.e., large n), most time spent in getting α

Dictionary Learning

Suppose α is given, then how to learn best D?

- Due to scale constraints, iterative methods for solving D
- For large-scale learning, projected stochastic gradient decent

$$D_t = P_C[D_{t-1} - \eta_t \nabla l(x_t, D_{t-1})] \qquad l(x_t, D_{t-1}) = (1/2)\|x_t - D_{t-1}\alpha\|_2^2$$

projection on constraint set
if $\|d_j\|_2 > 1$ then $d_j \leftarrow d_j / \|d_j\|_2$

step size
$a/(t+b)$

random sample from training set, can be extended to mini-batch

Dictionary Learning

Suppose α is given, then how to learn best D?

- Due to scale constraints, iterative methods for solving D
- For large-scale learning, projected stochastic gradient decent

$$D_t = P_C[D_{t-1} - \eta_t \nabla l(x_t, D_{t-1})] \qquad l(x_t, D_{t-1}) = (1/2)\|x_t - D_{t-1}\alpha\|_2^2$$

projection on constraint set
if $\|d_j\|_2 > 1$ then $d_j \leftarrow d_j / \|d_j\|_2$

step size
$a/(t+b)$

random sample from training set, can be extended to mini-batch

- Need to tune the step size
- May have slow convergence
- Can one exploit the structure of the sparse coding problem to get faster learning without much dependence on step size
- Alternative: use second order information in stochastic updates

Online Dictionary Learning

Key Idea: The loss function at time t aggregates information from samples drawn up to time t

best α for point x_i using the dictionary $D_{i\text{-}1}$

$$l_t(D) = (1/2t)\sum_{i=1}^{t} \|x_i - D\alpha_i\|_2^2$$

averaging over previous sparse reconstructions

Online Dictionary Learning

Key Idea: The loss function at time t aggregates information from samples drawn up to time t

best α for point x_i using the dictionary D_{i-1}

$$l_t(D) = (1/2t)\sum_{i=1}^{t} \|x_i - D\alpha_i\|_2^2$$

averaging over previous sparse reconstructions

$$= (1/2t)(Tr[D^T D A_t] - Tr[D^T B_t])$$

$$A_t = \sum_{i=1}^{t} \alpha_i \alpha_i^T \quad B_t = \sum_{i=1}^{t} x_i \alpha_i^T$$

Online Dictionary Learning

Key Idea: The loss function at time t aggregates information from samples drawn up to time t

best α for point x_i using the dictionary D_{i-1}

$$l_t(D) = (1/2t)\sum_{i=1}^{t} \|x_i - D\alpha_i\|_2^2$$

averaging over previous sparse reconstructions

$$= (1/2t)(Tr[D^T D A_t] - Tr[D^T B_t])$$

$$A_t = \sum_{i=1}^{t} \alpha_i \alpha_i^T \quad B_t = \sum_{i=1}^{t} x_i \alpha_i^T$$

- At each step, given D_{t-1}, α_t is obtained using x_t via any method, e.g., coordinate descent with soft thresholding, LARS with Cholesky etc.
- Dictionary learning based on block coordinate descent
- Does not require learning rate tuning
- With increasing t, the storage cost of x_i and α_t, $i < t$ can be expensive
  - No need to store these explicitly, only need to store A_t and B_t

Complete Algorithm

Given training set X, coefficient λ, initial dictionary D_0 and iterations T

$A_0 = 0, B_0 = 0$

For t = 1,…,T

1. Draw a random point x_t from X
2. Find best α_t, for x_t given D_{t-1}

$$\alpha_t = \arg\min_{\alpha} [(1/2)\|x_t - D_{t-1}\alpha\|_2^2 + \lambda\|\alpha\|_1]$$

use Lasso / LARS / iterative thresholding

3. Update $A_t \leftarrow A_{t-1} + \alpha_t \alpha_t^T \quad B_t \leftarrow B_{t-1} + x_t \alpha_t^T$

Complete Algorithm

Given training set X, coefficient λ, initial dictionary D_0 and iterations T

$A_0 = 0, B_0 = 0$

For t = 1,…,T

1. Draw a random point x_t from X
2. Find best α_t, for x_t given D_{t-1}

$$\alpha_t = \arg\min_{\alpha} [(1/2)\|x_t - D_{t-1}\alpha\|_2^2 + \lambda\|\alpha\|_1]$$

use Lasso / LARS / iterative thresholding

3. Update $A_t \leftarrow A_{t-1} + \alpha_t \alpha_t^T \quad B_t \leftarrow B_{t-1} + x_t \alpha_t^T$
4. Update dictionary using block coordinate descent (starting with D_{t-1})

$$d_j \leftarrow \frac{1}{A_{jj}}(b_j - Da_j) + d_j \qquad \text{if } \|d_j\|_2 > 1 \Rightarrow d_j \leftarrow d_j / \|d_j\|_2$$

or using Newton's method by first creating a Lagrangian using constraints

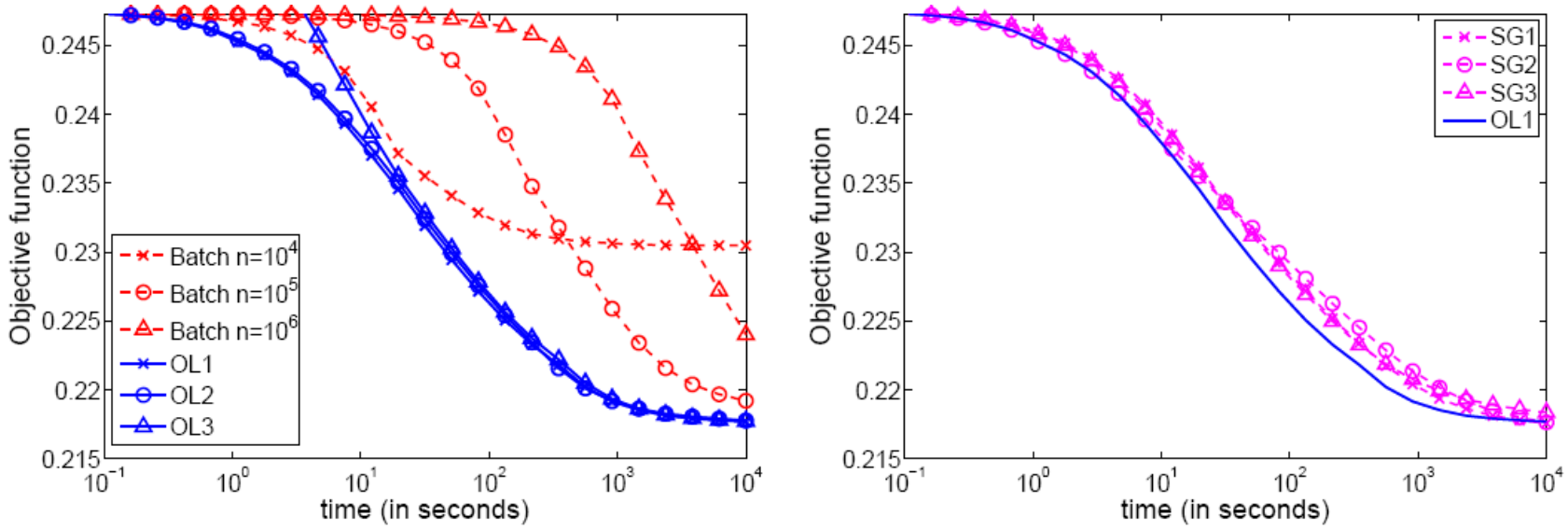
- Need to invert $k \times k$ matrix (fine for small k)

What happens if T > n ? Various heuristics to remove the contribution of older α's !

Dictionary Learning Experiment

$n = 1.25M$ image patches

$d = 256, \; k = 1024$



Mairal et al. [13]

Dictionary Learning Experiment

Inpainting, e.g., remove text from images

$n = 7M$ patches, $d = 432$, $k = 256$

Training time: 8 mins, 2.4GHz/8 cores

Mairal et al. [13]

Sparse PCA

Key Idea: Find sparse vectors that maximize the variance of the projected data or minimize mean squared error of reconstruction

Benefits: Less storage and computation, Interpretability

Formulation 1

- Maximize variance subject to sparsity constraints

Sparse PCA

Key Idea: Find sparse vectors that maximize the variance of the projected data or minimize mean squared error of reconstruction

Benefits: Less storage and computation, Interpretability

Formulation 1

- Maximize variance subject to sparsity constraints

suppose data matrix X is centered, i.e., zero mean

$$\hat{D} = \arg\max_{D} Tr[D^T XX^T D] \qquad X \in \Re^{d \times n}, D \in \Re^{d \times k}$$

$$\text{subject to } D^T D = I$$

$$\sum_{j=1}^{k} \left\| d_j \right\|_1 \leq C$$

- Usually each d_j is learned incrementally by enforcing orthonormality and sparsity constraints
- problem is non-convex and computationally expensive

Sparse PCA

Key Idea: Find sparse vectors that maximize the variance of the projected data or minimize mean squared error of reconstruction

Formulation 2

- Minimize squared reconstruction error subject to sparsity constraints

Recall PCA formulation $\hat{D} = \arg\min_{D} \left\| X - DD^T X \right\|_2^2$ subject to $D^T D = I$

Sparse PCA

Key Idea: Find sparse vectors that maximize the variance of the projected data or minimize mean squared error of reconstruction

Formulation 2

- Minimize squared reconstruction error subject to sparsity constraints

Recall PCA formulation $\hat{D} = \arg\min_{D} \left\| X - DD^T X \right\|_2^2$ subject to $D^T D = I$

Sparse PCA formulation $\hat{B}, \hat{D} = \arg\min_{B,D} \left\| X - BD^T X \right\|_2^2 + \lambda_1 \sum_{j=1}^{k} \left\| d_j \right\|_2^2 + \lambda_2 \sum_{j=1}^{k} \left\| d_j \right\|_1$

$D \in \Re^{d \times k}, B \in \Re^{d \times k}$

subject to $B^T B = I$

elastic net penalty

- If no L_1 penalty, solution the same as PCA
- Solved via alternate minimization, using LARS and SVD
- Much more expensive than PCA

Sparse PCA

Key Idea: Find sparse vectors that maximize the variance of the projected data or minimize mean squared error of reconstruction

Formulation 3

- Minimize square reconstruction via dictionary learning view

$$\min_{D,\alpha} \sum_{i=1}^{n} [(1/2)\|x_i - D\alpha_i\|_2^2 + \lambda\|\alpha_i\|_1] \qquad \text{sparse } \alpha$$

$$\text{subject to } \|d_j\|_2^2 + \gamma\|d_j\|_1 \le 1 \quad j = 1,\ldots,k \qquad \text{sparse } d_j$$

Sparse PCA

Key Idea: Find sparse vectors that maximize the variance of the projected data or minimize mean squared error of reconstruction

Formulation 3

- Minimize square reconstruction via dictionary learning view

$$\min_{D,\alpha} \sum_{i=1}^{n} [(1/2)\|x_i - D\alpha_i\|_2^2 + \lambda\|\alpha_i\|_1] \qquad \text{sparse } \alpha$$

$$\text{subject to } \|d_j\|_2^2 + \gamma\|d_j\|_1 \le 1 \quad j = 1,\ldots,k \qquad \text{sparse } d_j$$

- Solved via alternate minimization as for sparse coding
- Dictionary learning step is modified as

$$d'_j \leftarrow \frac{1}{A_{jj}}(b_j - Da_j) + d_j$$

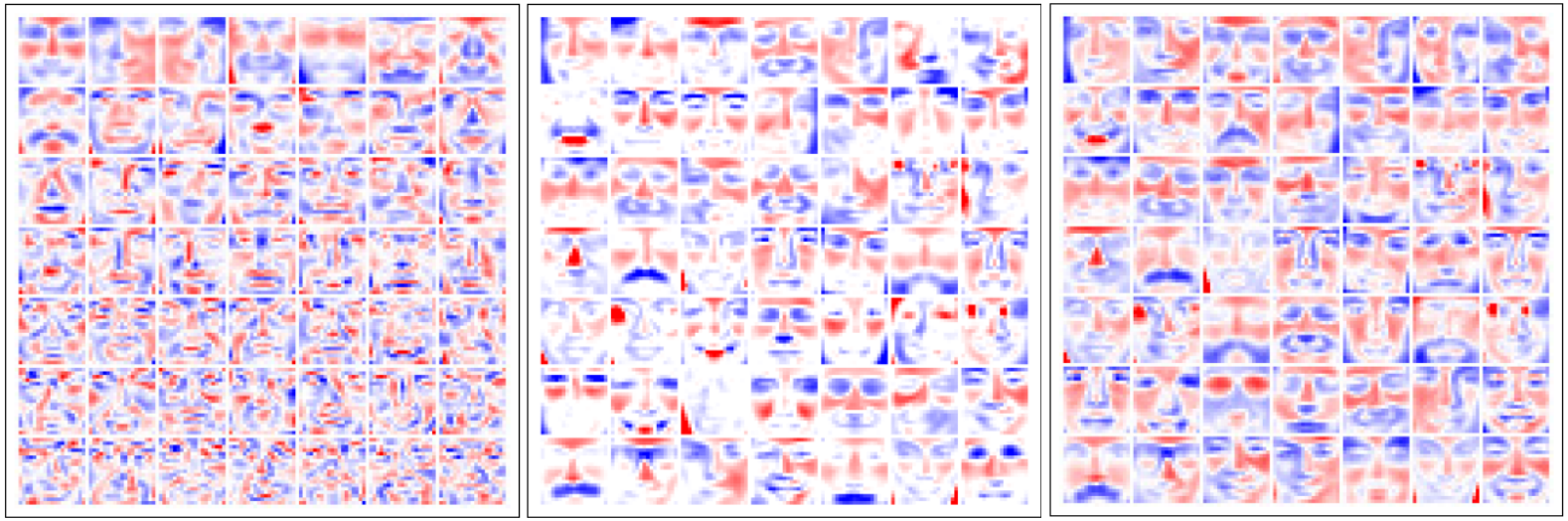
$$\text{subject to } \quad d_j \leftarrow \arg\min_d \|d'_j - d\|_2^2 \text{ s.t. } \|d\|_2^2 + \gamma\|d\|_1 \le 1 \quad j = 1,\ldots,k$$

No orthonormality constraints on d_j !

Dictionary Learning Experiment

$n = 2414$ face images

$d = 256,\ k = 49$



(a) PCA (b) SPCA, $\tau = 70\%$ (e) Dictionary Learning

Mairal et al. [13]

References

1. Tibshirani, R. Regression shrinkage and selection via the lasso. J. Royal. Statist. Soc B., Vol. 58, No. 1, 267-288, 1996.
2. B. A. Olshausen and D. J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37:3311–3325, 1997.
3. B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals Statistics*, 32(2):407–499, 2004.
4. H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B*, 67(2):301–320, 2005.
5. B. Moghaddam, Y. Weiss and S. Avidan. Spectral Bounds for Sparse PCA: Exact and Greedy Algorithms. NIPS, 2005.
6. H. Zou, T. Hastie, and R. Tibshirani. Sparse principal component analysis. *Journal of Computational and Graphical Statistics*, 15(2):265–286, 2006.
7. M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B*, 68:49–67, 2006.
8. H. Lee, A. Battle, R. Raina and A. Ng. Efficient Sparse Coding Algorithms. NIPS, 2006.
9. T. T. Wu and K. Lange. Coordinate descent algorithms for Lasso penalized regression. *Annals of Applied Statistics*, 2(1):224–244, 2008.
10. K. Kavukcuoglu, M. Ranzato and Y. LeCun. *Fast Inference in Sparse Coding Algorithms with Applications to Object Recognition*, Courant Institute, NYU, Tech Report CBLL-TR-2008-12-01, 2008.
11. A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imag. Sci.*, 2(1):183–202, 2009.
12. J. Mairal, F. Bach, J. Ponce, G. Sapiro. Online Learning for Matrix Factorization and Sparse Coding. *Journal of Machine Learning Research*, 11, 10-60, 2010.
13. Karol Gregor and Yann LeCun: Learning Fast Approximations of Sparse Coding, *Proc. International Conference on Machine learning (ICML). 2010*
14. A. Quattoni, X. Carreras, M. Collins and T. Darrell, “An Efficient Projection for L1∞ Regularization,” ICML, 2010.
15. H. Liu, M. Palatucci, J. Zhang, “Blockwise Coordinate Descent Procedures for the Multitask Lasso, with Applications to Neural Semantic Basis Discovery,” ICML, 2010.