Analyzing Task Driven Learning Algorithms

Final Presentation

Mike Pekala

May 1, 2012

Advisor: Prof. Doron Levy (dlevy at math.umd.edu)
UMD Dept. of Mathematics & Center for Scientific Computation and Mathematical Modeling (CSCAMM)

Project Overview

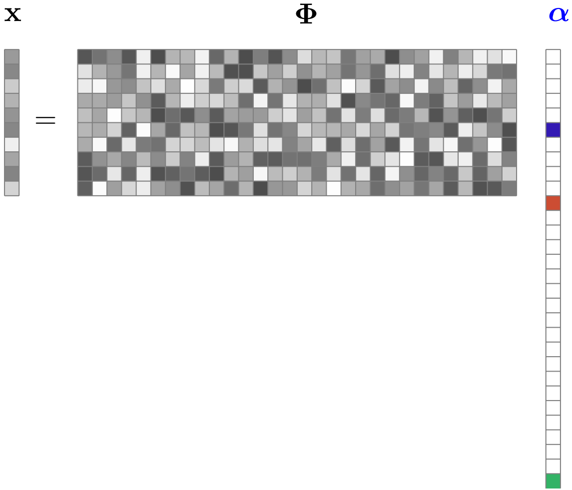
Existing Algorithm Implementation/Validation

- Sparse Reconstruction
  - Least Angle Regression (LARS) [Efron et al., 2004]
  - Feature-Sign [Lee et al., 2007]
  - Non-negative and incremental Cholesky variants
- Dictionary Learning
  - Task-Driven Dictionary Learning (TDDL) [Mairal et al., 2010]

Application/Analysis to New (Publicly Available) Datasets

- Hyperspectral Imagery
  - Urban [US Army Corps of Engineers, 2012]
  - USGS Hyperspectral Library [Clark et al., 2007]

Topic: Sparse Reconstruction



Penalized Least Squares

Recall the Lasso: given $\mathbf{\Phi} = [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_p] \in \mathbb{R}^{m \times p}, t \in \mathbb{R}_+$, solve:

$$\min_{\boldsymbol{\alpha}} \ ||\mathbf{x} - \mathbf{\Phi}\boldsymbol{\alpha}||_2^2 \ \ s.t. \ \ ||\boldsymbol{\alpha}||_1 \leq t$$

which has an equivalent unconstrained formulation:

$$\min_{\boldsymbol{\alpha}} \ ||\mathbf{x} - \mathbf{\Phi}\boldsymbol{\alpha}||_2^2 + \lambda||\boldsymbol{\alpha}||_1$$

for some scalar $\lambda \geq 0$. The L_1 penalty improves upon OLS by introducing parsimony (feature selection) and regularization (improved generality).

Many ways to solve this problem, e.g.

1. Directly, via convex optimization (can be expensive)
2. Iterative techniques
   - Forward selection (“matching pursuit”), forward stagewise, others.
   - Least Angle Regression (LARS) [Efron et al., 2004]
   - Feature-Sign [Lee et al., 2007]

LARS Properties

Full details in [Efron et al., 2004]

Why is it good?

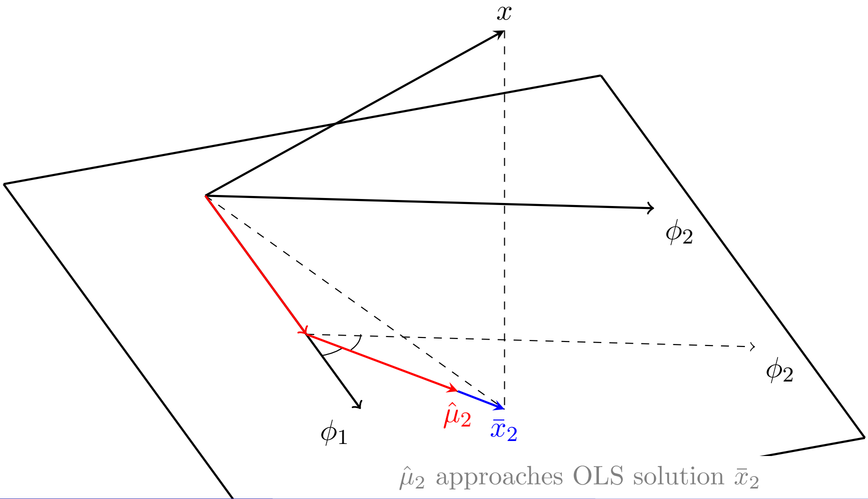
- Less aggressive than some greedy techniques; less likely to eliminate useful predictors when predictors are correlated.
- More efficient than Forward Selection, which can take thousands of tiny steps towards a final model.

Some Properties

- **(Theorem 1)** Assuming covariates added/removed one at a time from active set, complete LARS solution path yields *all* Lasso solutions.
- **(Sec. 3.1)** With a change to the covariate selection rule, LARS can be modified to solve the *Positive Lasso* problem.
- **(Sec. 7)** The cost of LARS is comprable to that of a least squares fit on m variables. The LARS sequence incrementally generates a Cholesky factorization of $\mathbf{\Phi}^T\mathbf{\Phi}$ in a very specific order.

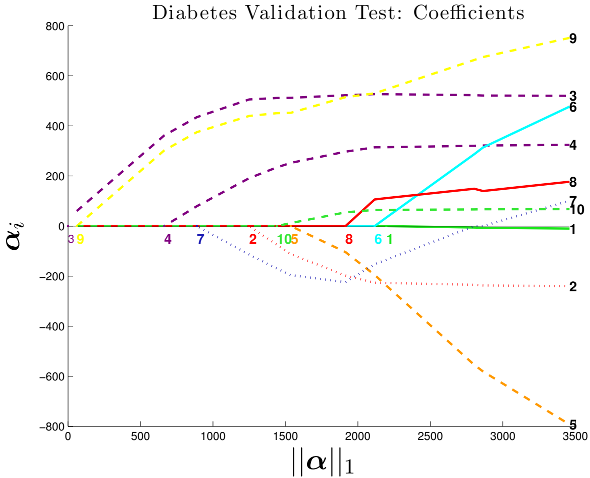
LARS Relationship to OLS

(2.22) Successive LARS estimates $\hat{\mu}_k$ always approach but never reach the OLS estimate $\bar{x}_k$ (except maybe on the final iteration).



$\hat{\mu}_2$ approaches OLS solution $\bar{x}_2$

LARS Implementation/Validation



$n = 10, m = 442$; Matches Figure 1 in [Efron et al., 2004]
Also validated by comparing orthogonal designs with theoretical result.

Feature-Sign Properties

Full details in [Lee et al., 2007]

Why is it good?

- Very efficient; reported performance gains over LARS.
- Can be initialized with arbitrary starting coefficents.
- Simple to implement.
- One half of a two-part algorithm for matrix factorization.

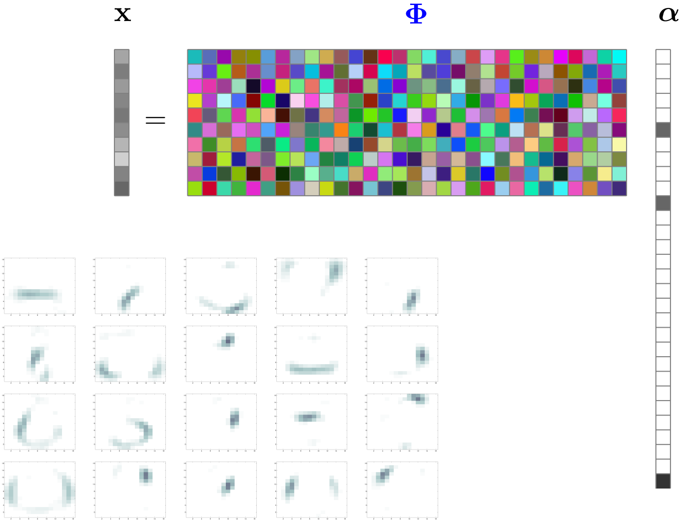
Some Properties

- Tries to search for, or "guess", signs of coefficients. Knowing signs reduces LASSO to an unconstrained quadratic program (QP) with closed form solution.
- Iteratively refines these sign guesses; involves an intermediate line search.
- Objective function strictly decreases.

Feature-Sign Implementation/Validation

- Implemented nonnegative extension. Performance hit (at least w/ my implementation) as the unconstrained QP becomes a constrained QP. Solved using Matlab's quadprog().
- Validated by comparing results with LARS

Topic: Dictionary Learning



Dictionary Learning for Sparse Reconstruction

Following the notation/development of [Mairal et al., 2010].

- Given: training data set of signals $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$ in $\mathbb{R}^{m \times n}$
- Goal: design a dictionary $\mathbf{\Phi}$ in $\mathbb{R}^{m \times p}$ (possible for $p > m$, i.e. an *overcomplete* dictionary) by minimizing the empirical cost function

$$g_n(\mathbf{D}) \triangleq \frac{1}{n} \sum_{i=1}^{n} \ell_u(\mathbf{x}_i, \mathbf{D})$$

where ℓ_u, the *unsupervised* loss function, is small when $\mathbf{\Phi}$ is "good" at representing $\mathbf{x}_i$ sparsely. In [Mairal et al., 2010], the authors use the elastic-net formulation:

$$\ell_u(\mathbf{x}, \mathbf{D}) \triangleq \min_{\boldsymbol{\alpha} \in \mathbb{R}^p} \frac{1}{2} ||\mathbf{x} - \mathbf{D}\boldsymbol{\alpha}||_2^2 + \lambda_1 ||\boldsymbol{\alpha}||_1 + \frac{\lambda_2}{2} ||\boldsymbol{\alpha}||_2^2 \tag{1}$$

Dictionary Learning for Sparse Reconstruction

- To prevent artificially improving ℓ_u by arbitrarily scaling $\mathbf{D}$, one typically constrains the set of permissible dictionaries:

$$\mathcal{D} \triangleq \{\mathbf{D} \in \mathbb{R}^{m \times p} \text{ s.t. } \forall j \in \{1, \ldots, p\}, ||\mathbf{d}_j||_2 \leq 1\}$$

- Optimizing the empirical cost g_n can be very expensive when the training set is large (as is often the case in dictionary learning problems). However, in reality, one usually wants to minimize the expected loss:

$$g(\mathbf{D}) \triangleq \mathbb{E}_{\mathbf{x}}\left[\ell_u(\mathbf{x}, \mathbf{D})\right] = \lim_{n \to \infty} g_n(\mathbf{D}) \quad \text{a.s.}$$

(where expectation is taken with respect to the unknown distribution of data objects $p(\mathbf{x})$) In these cases, online stochastic techniques have been shown to work well [Mairal et al., 2009].

Classification and Sparse Reconstruction

Consider the classification task:

- Given: a fixed dictionary $\mathbf{D}$, an observation $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^m$ and a sparse representation of the observation $\mathbf{x} \approx \boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})$
- Goal: identify the associated label $y \in \mathcal{Y}$, where $\mathcal{Y}$ is a finite set of labels (would be a subset of $\mathbb{R}^q$ for regression)

Assume $\mathbf{D}$ is fixed and $\boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})$ will be used as the features for predicting y. The classification problem is to learn the model parameters $\mathbf{W}$ by solving:

$$\min_{\mathbf{W} \in \mathcal{W}} f(\mathbf{W}) + \frac{\nu}{2} ||\mathbf{W}||_F^2$$

where

$$f(\mathbf{W}) \triangleq \mathbb{E}_{y,\mathbf{x}} \left[\ell_s(y, \mathbf{W}, \boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})) \right]$$

and ℓ_s is a convex loss function (e.g. logistic) adapted to the supervised learning problem.

Task Driven Dictionary Learning for Classification

Now, we wish to *jointly* learn $\mathbf{D}, \mathbf{W}$:

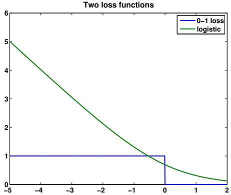
$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{W}\in\mathcal{W}} f(\mathbf{D},\mathbf{W}) + \frac{\nu}{2}||\mathbf{W}||_F^2 \tag{2}$$

where

$$f(\mathbf{D},\mathbf{W}) \triangleq \mathbb{E}_{y,\mathbf{x}}\left[\ell_s(y,\mathbf{W},\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D}))\right]$$

Example:

Binary classification: $\mathcal{Y} = \{-1,+1\}$
Linear model: $\mathbf{w} \in \mathbb{R}^p$
Prediction: $\text{sign}(\mathbf{w}^T\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D}))$
Logistic loss: $\ell_s = \log\left(1 + e^{-yw^T\boldsymbol{\alpha}^\star}\right)$



$$\min_{\mathbf{D}\in\mathcal{D},\mathbf{w}\in\mathbb{R}^p} \mathbb{E}_{y,\mathbf{x}}\left[\log\left(1 + e^{-y\mathbf{w}^T\boldsymbol{\alpha}^\star(\mathbf{x},\mathbf{D})}\right)\right] + \frac{\nu}{2}||\mathbf{w}||_2^2 \tag{3}$$

Solving the Problem

Stochastic gradient descent is often used to minimize functions whose gradients are expectations. The authors of [Mairal et al., 2010] show that, under suitable conditions, equation (2) is differentiable on $\mathcal{D} \times \mathcal{W}$, and that,

$$\nabla_{\mathbf{W}} f(\mathbf{D}, \mathbf{W}) = \mathbb{E}_{y,\mathbf{x}} \left[\nabla_{\mathbf{W}} \ell_s(y, \mathbf{w}, \boldsymbol{\alpha}^\star) \right]$$
$$\nabla_{\mathbf{D}} f(\mathbf{D}, \mathbf{W}) = \mathbb{E}_{y,\mathbf{x}} \left[-\mathbf{D}\boldsymbol{\beta}^\star \boldsymbol{\alpha}^{\star T} + (\mathbf{x} - \mathbf{D}\boldsymbol{\alpha}^\star)\boldsymbol{\beta}^{\star T} \right]$$

where $\boldsymbol{\beta}^\star \in \mathbb{R}^p$ is defined by the properties:

$$\boldsymbol{\beta}^\star \Lambda^C = 0 \text{ and } \boldsymbol{\beta}^\star \Lambda = (D_\Lambda^T D_\Lambda + \lambda_2 \mathbf{I})^{-1} \nabla_{\alpha_\Lambda} \ell_s(y, \mathbf{W}, \boldsymbol{\alpha}^\star)$$

and Λ are the indices of the nonzero coefficients of $\boldsymbol{\alpha}^\star(\mathbf{x}, \mathbf{D})$.

Algorithm: SGD for task-driven dictionary learning

[Mairal et al., 2010]

Input: $p(y, \mathbf{x})$ (a way to draw samples i.i.d. from p), $\lambda_1, \lambda_2, \nu \in \mathbb{R}$ (regularization parameters), $\mathbf{D} \in \mathcal{D}_0$ (initial dictionary), $\mathbf{W}_0 \in \mathcal{W}$ (initial model), T (num. iterations), $t_0, \rho \in \mathbb{R}$ (learning rate parameters)

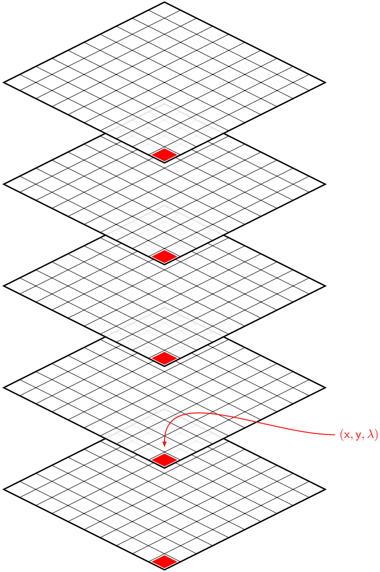
1. for $t = 1$ to T do
2. Draw $(y_t, \mathbf{x}_t)$ from $p(y, \mathbf{x})$ (mini-batch of size 200)
3. Compute $\boldsymbol{\alpha}^\star$ via sparse coding (LARS, Feature-Sign)
4. Determine active set Λ and $\boldsymbol{\beta}^\star$
5. Update learning rate ρ_t
6. Take projected gradient descent step
7. end

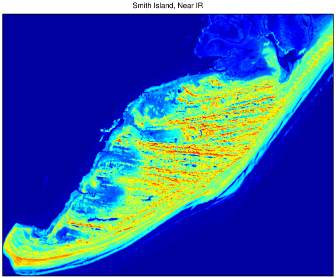
TDDL Implementation/Validation

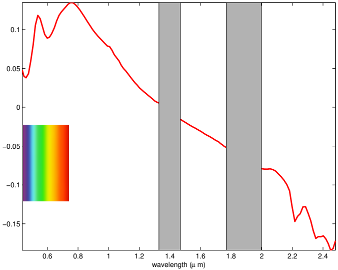
Matched experimental results on the USPS [Hastie et al., 2009] data set with those reported in [Mairal et al., 2010]

Digit	ρ	λ	# in D_0	Runtime (h)	Accuracy
0	10	.150	5	8.2	.926
1	10	.225	7	7.1	.990
2	10	.225	7	6.8	.972
3	10	.225	7	7.4	.968
4	10	.225	4	7.6	.971
5	10	.225	4	7.2	.972
6	10	.225	2	7.5	.969
7	10	.175	5	7.9	.983
8	10	.200	3	8.5	.951
9	10	.200	3	8.1	.969
mean					.967
reported					.964

Topic: Hyperspectral Imaging







Spectral Unmixing

Material heterogeneity and environmental interference mean that one never measures "pure" pixels/spectra. Instead, "spectral unmixing" is often used to determine the material present at some pixel $\mathbf{x} \in \mathbb{R}^m$,

$$\mathbf{x} = \sum_{k=1}^{n} \boldsymbol{\phi}_k \alpha_k + \epsilon$$

where $\{\phi_k\}$ is a spectral library, $\{a_k\}$ are scalar mixture coefficients and ϵ is noise. Recent results suggest *sparse coding* may apply to the spectral unmixing problem; also to infer HSI-resolution data from lower resolution measurements [Charles et al., 2011].

Mixture Element Detection

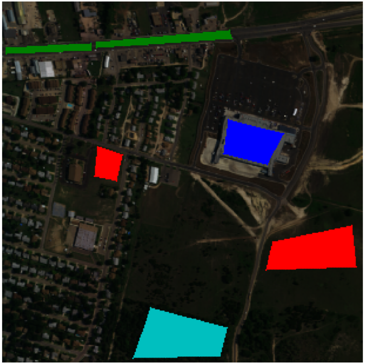
- Original plan: analysis on single pixel classification problems for objects in scene comprised of ≥ 1 pixel
  - Problem very easy in some cases (baseline algorithms have no difficulty)
  - In the opinion of one HSI expert, a more relevant problem today is sub-pixel detection
- Modified plan: "mixture element detection" problem
  - Select a single spectral signature as the target
  - Generate mixtures of s spectral "ingredients"; some containing target signature, some without
  - Binary classification problem: identify mixtures containing target signatures
- Used TDDL + nonnegative Feature-Sign solver; various baselines for comparison

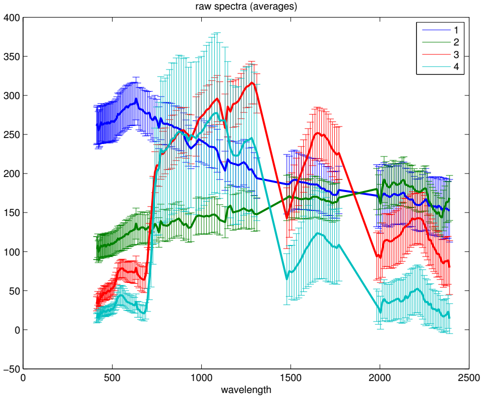
Urban

[US Army Corps of Engineers, 2012]

- Urban scene in Texas
- 307× 307 pixels
- 210 spectral bands (162 valid)
- wavelengths: 412-2390 nm
- radiance data
- no “standard” ground truth
- freely available

Manual Ground Truth





Mixture Element Detection, 1-vs-all Classification

Classification Accuracy

	LR	kNN1	kNN3	LR-SC	kNN1-SC	kNN3-SC	TD-10
M1	84.0	83.2	82.0	**86.8**	77.0	78.4	82.4
M2	82.6	79.2	76.6	75.8	72.8	77.4	81.2
M3	76.8	74.6	75.0	74.0	75.2	69.0	**81.4**
M4	86.4	87.8	82.6	84.0	77.6	78.6	**88.2**

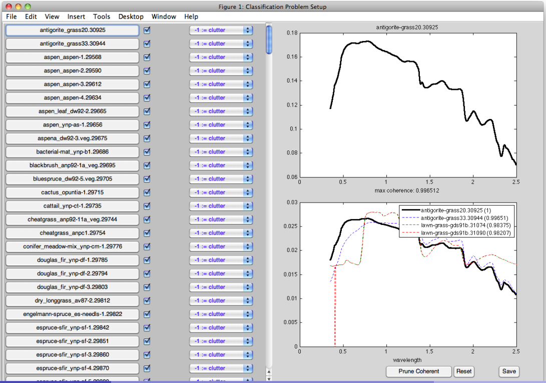
Parameter	Value
# Target Mixtures	500
# Clutter Mixtures	500
% Training	50
# Ingredients	3
Min. % Target	5
Max. % Target	25
Noise Variance	0
TDDL Iterations	10000

- Not clear any one approach significantly better
- Only 4 total ingredients in library, signatures fairly distinct

USGS Spectral Library

[Clark et al., 2007]

- Freely available library of 1365 different spectra (minerals, mixtures, coatings, volatiles, man-made, vegetation)
- Focus on a subset of 44 spectra from the vegetation category ($\sim 0.3 - 2.5\mu$m, $\sim$ 1200 valid wavelengths)



Mixture Element Detection, 1-vs-all Classification

Classification Accuracy

	LR	kNN1	kNN3	LR-SC	kNN1-SC	kNN3-SC	TD-50	TD-300
M1	60.2	63.4	65.8	67.7	57.4	61.6	**70.2**	**70.2**
M2	49.2	61.6	**63.2**	56.4	52.8	60.0	59.4	60.8
M3	52.8	56.2	54.0	56.4	52.2	50.4	54.6	55.0
M4	57.8	62.4	61.8	60.6	54.0	56.6	63.6	**70.8**
M5	55.0	67.6	68.6	66.6	59.6	63.2	64.2	**73.34**
M6	52.2	59.0	62.6	61.0	54.2	57.6	62.8	**65.2***
M7	44.8	56.4	59.6	60.2	56.8	58.4	**63.4**	**64.2***
M8	64.8	80.8	81.4	66.6	64.0	65.2	82.2	81.2

Parameter	Value
# Target Mixtures	500
# Clutter Mixtures	500
% Training	50
# Ingredients	5
Min. % Target	5
Max. % Target	25
Noise Variance	0.001
TDDL Iterations	1000

- More challenging mixture model
- LR suffers from noise; SC helps
- TDDL relatively strong performer
- kNN3 pretty good, especially when given enough data

(* := TD-200)

Processing

- Platform Load Sharing Facility (LSF) scheduler on 20 compute nodes (Intel Xeon X5650, 12 threads)
- Software includes scripts for various tasks (kfold CV, train/test)

```
$ lsload
HOST_NAME       status  r15s   r1m  r15m   ut    pg  ls    it   tmp   swp   mem
cn17                ok   0.0   0.0   0.0   0%   0.0   0 40576 8824M 2000M   22G
maul                ok   0.0   0.2   0.1   0%   0.0   7    15   19G   26G   20G
cn00                ok  12.0  12.2  11.8  99%   0.0   0 10528 8824M 2000M   22G
cn08                ok  12.0  12.5  12.0  99%   0.0   0 3e+05 8824M 2000M   22G
cn12                ok  12.0  12.2  11.7  99%   0.0   0 3e+05 8824M 2000M   22G
cn07                ok  12.0  12.3  11.8  99%   0.0   0 3e+05 8824M 2000M   22G
cn19                ok  12.0  12.2  11.7  98%   0.0   0  9040 8832M 2000M   22G
cn15                ok  12.0  12.0  11.8  98%   0.0   0 3e+05 8824M 2000M   22G
cn13                ok  12.0  12.4  11.7  99%   0.0   0  7264 8824M 2000M   22G
cn04                ok  12.0  11.6  11.6  98%   0.0   0  9016 8824M 2000M   22G
cn02                ok  12.0  12.3  11.9  99%   0.0   0 47712 8824M 2000M   22G
cn03                ok  12.1  12.4  12.1  99%   0.0   0 29312 8832M 2000M   22G
cn05                ok  12.1  12.4  11.7  99%   0.0   0  7504 8824M 2000M   22G
cn09                ok  12.2  12.1  11.8  98%   0.0   0 20240 8824M 2000M   22G
cn11                ok  12.2  12.1  11.9  98%   0.0   0  9032 8824M 2000M   22G
cn14                ok  12.2  11.9  11.6  99%   0.0   0 3e+05 8824M 2000M   22G
cn16                ok  12.3  11.6  11.7  99%   0.0   0 3e+05 8824M 2000M   22G
cn01                ok  12.3  11.9  11.8  98%   0.0   1    71 8824M 2000M   22G
cn10                ok  12.3  12.1  11.8  99%   0.0   0 30672 8824M 2000M   22G
cn18                ok  12.3  12.8  11.9  99%   0.0   0    78 8832M 2000M   22G
cn06                ok  12.6  11.5  11.5  99%   0.0   0 3e+05 8824M 2000M   22G
```

Deliverables

Software/Data Sets

- Solvers (LARS, F-S, TDDL): ∼2000 lines of Matlab
  - Diabetes data set downloaded from LARS author's website; removed header (provided)
  - Test matrices constructed on-the-fly by unit tests (provided)
  - Limited doxygen documentation (requires doxygen and "Using Doxygen with Matlab" from Matlab Central to regenerate)
- Analysis experiments: ∼1500 lines of Matlab, ∼140 lines bash
  - URLs to HSI data sets provided in references
- USGS Viewer: ∼500 lines of Matlab

Presentations (9/22/2011, 12/6/2011, 3/15/2012, 5/1/2012)

Final report and software tarball to be delivered by May 11

Doxygen

Solvers

Main Page | Files

File List | File Members

File List

Here is a list of all files with brief descriptions:

FeatureSign/fs_l1.m	Implements the sparse coding algorithm 1 from [1]
FeatureSign/Unittests/compare_to_lars.m	Compares LARS and FeatureSign solutions for consistency
FeatureSign/Unittests/fsl1_test_correlated.m	Make sure the algorithm runs with correlated atoms
LARS/lars.m	An implementation of the Least Angle Regression algorithm [1]
LARS/Unittests/fix_math_fonts.m	Improves fonts of the current plot
LARS/Unittests/plot_k_vs_cost.m	Plots objective function cost vs iteration
LARS/Unittests/plot_l1_vs_value.m	Plots LARS results for diabetes data set
LARS/Unittests/test_correlated.m	Make sure the algorithm runs with correlated atoms
LARS/Unittests/test_diabetes.m	Runs LARS on the diabetes data set provided by [1]
LARS/Unittests/test_orthogonal.m	Test the LARS algorithm using an orthogonal dictionary
LARS/Unittests/test_problemA.m	Runs a specific case that was problematic in the past
LARS/Unittests/test_problemB.m	Runs a specific case that was problematic in the past
TDDL/tddl.m	Task driven dictionary learning solver
TDDL/tddl_calc_gradients.m	Gradient calculations for the TDDL algorithm
TDDL/tddl_mpi.m	Task driven dictionary learning, MPI version (deprecated)
TDDL/unsupervised_dl.m	Dictionary learning w/o a task
TDDL/Unittests/test_grad_w.m	Some experiments with TDDL gradients of w (deprecated)

Generated on Sun Apr 29 2012 23:44:06 for Solvers by doxygen 1.7.4

Summary

Project Goals Met

- Implemented algorithms from three papers
    - (LARS, Feature-Sign, TDDL)
- Validated using data sets with existing/known results
    - (diabetes, orthogonal designs, USPS)
- Conducted new experiments with hyperspectral data sets
    - (Urban, USGS)

Thanks!!

- Dr.'s Levy, Balan, Ide, Wang, Banerjee for guidance and help throughout the course!
- AMSC663/4 for great questions and enduring four presentations on this topic ⌣̈

Bibliography I

Adam S. Charles, Bruno A. Olshausen, and Christopher J. Rozell. Learning sparse codes for hyperspectral imagery. *J. Sel. Topics Signal Processing*, 5(5):963–978, 2011.

R.N. Clark, G.A. Swayze, R. Wise, E. Livo, T. Hoefen, R. Kokaly, and S.J. S.J. Sutley. Usgs digital spectral library splib06a: U.s. geological survey, digital data series 231, 2007. `http://speclab.cr.usgs.gov/spectral.lib06/`.

Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *Annals of Statistics*, 32:407–499, 2004.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Datasets for "the elements of statistical learning", 2009. `http://www-stat-class.stanford.edu/~tibs/ElemStatLearn/data.html`.

Bibliography II

Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In *In NIPS*, pages 801–808. NIPS, 2007.

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 689–696, New York, NY, USA, 2009. ACM.

Julien Mairal, Francis Bach, and Jean Ponce. Task-Driven Dictionary Learning. Rapport de recherche RR-7400, INRIA, 2010.

Army Geospatial Center US Army Corps of Engineers, 2012. `http://www.agc.army.mil/hypercube/`.