Feature Selection with Annealing

Adrian Barbu

Department of Statistics

Florida State University

Acknowledgements: Yiyuan She

Motivation

Why Feature Selection?

- Understanding underlying causes
  - What genes are responsible for breast cancer?
  - What foods are responsible for heart disease?
  - What factors impact student achievement?

- Prediction
  - Balance speed vs accuracy
  - Smaller models generalize better to unseen data

Overview

- Objective:
  - Simultaneous feature selection and model learning
- Novel approach:
  - Generic setup of minimizing a loss function $L(\beta)$
    - Constraints on the number of non-zero parameters β_j
  - Start by minimizing the unconstrained loss
  - Gradually tighten the constraints according to a schedule and a variable elimination criterion
- Experiments
  - Simulations
    - Compared with many state of the art methods
  - Real Data
    - Compared with Logitboost, SVM and Rankboost

Problem Setup

- Given training set $D = \{(\boldsymbol{x}_i, y_i) \in \mathbb{R}^p \times \{-1, 1\}, i = \overline{1, N}\}$ want to optimize a differentiable loss function L_D with sparsity constraints

$$\beta = \underset{|\{i, \beta_i \neq 0\}| \leq k}{\operatorname{argmin}} L_D(\beta)$$

- Example: Logistic loss with shrinkage:

$$L_D(\beta) = \frac{1}{N} \sum_{i=1}^{N} \ln(1 + e^{-y_i \beta^T \mathbf{x}_i}) + s\|\beta\|_2^2$$

Challenges

- Computationally challenging
  - Exhaustive search takes $O(p^k N)$ which is large since $p \sim 10^5$, $N \sim 10^8$, $k \sim 1000$ for computer vision problems

Other approaches

- Replace the L_0 constraints with the L_1 penalty (Tibshirani '96, Bunea '07)
  - Pros:
    - Convex problem
    - Theoretical guarantees
  - Cons:
    - Biases the coefficients – poor prediction performance
    - Computationally expensive
    - Weak feature selection capability

Other Approaches

Use sparsity inducing non-convex penalties

- SCAD penalty (Fan&Li ’01, Zhou&Li ’08)
- MCP Penalty (Zhang ’10)

Pros:

- Better feature selection capabilities
- Theoretical guarantees
- Less biased

Cons:

- Computationally expensive
- Does not scale well to large problems
- Coefficients still slightly biased

Boosting

Logitboost (Friedman et al, ‘96)

- Optimizes the logistic loss without shrinkage
- Each boosting iteration is a Newton optimization step

Pros:

- Computationally efficient
- Can handle large problems

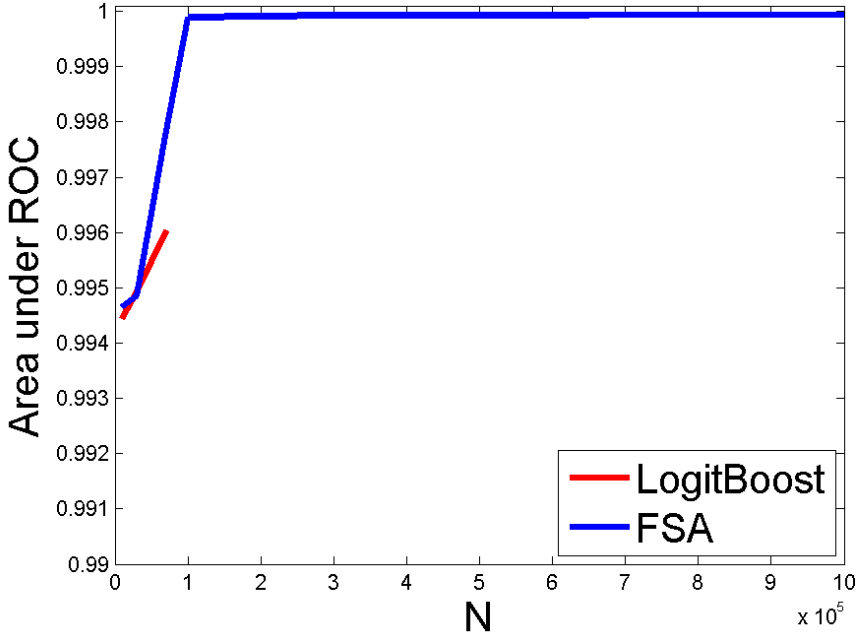
Cons:

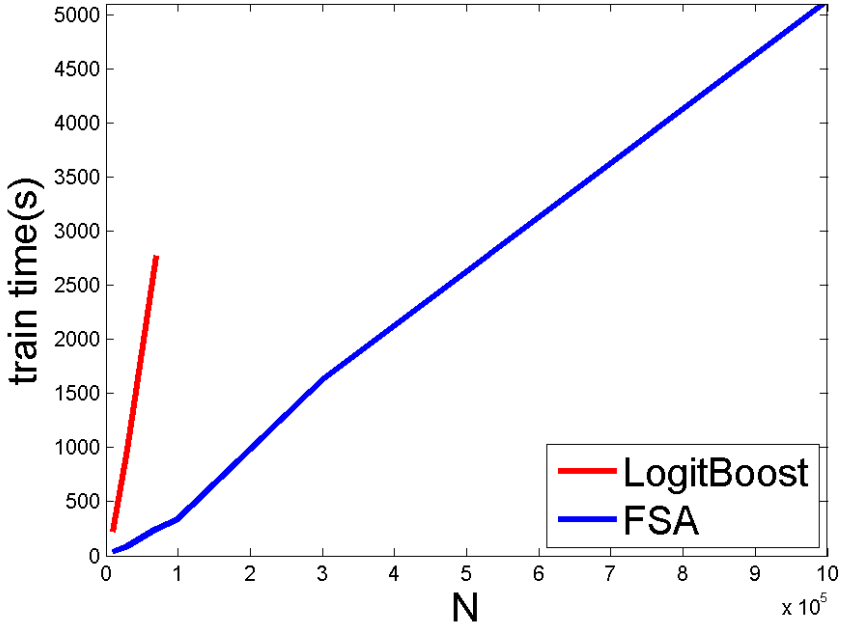
- Hard to obtain theoretical guarantees
- Learned parameters are suboptimal

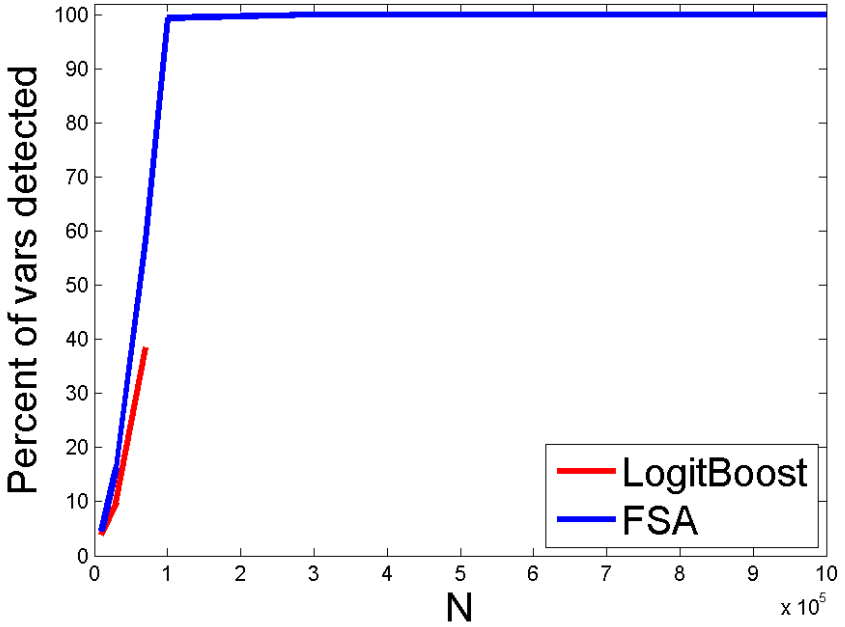
Boosting vs Proposed (FSA)

Large data:

- Total $p=10^5$ variables
- $N=10^6$ observations
- Select k=500 variables
- Outperforms Logitboost in speed, accuracy and scalability







TISP (She 2009)

Thresholding Based Iterative Selection Procedure

- Can work with L_1,L_0, SCAD and other penalties
- Iterates two steps until convergence
  - Parameter update step
  - Thresholding step on the coefficients β
- Usually converges in less than 10 steps

Pros

- Computationally efficient

Cons:

- Too greedy: thresholds almost all coefficients in the first step
  - Feature selection performance could be better

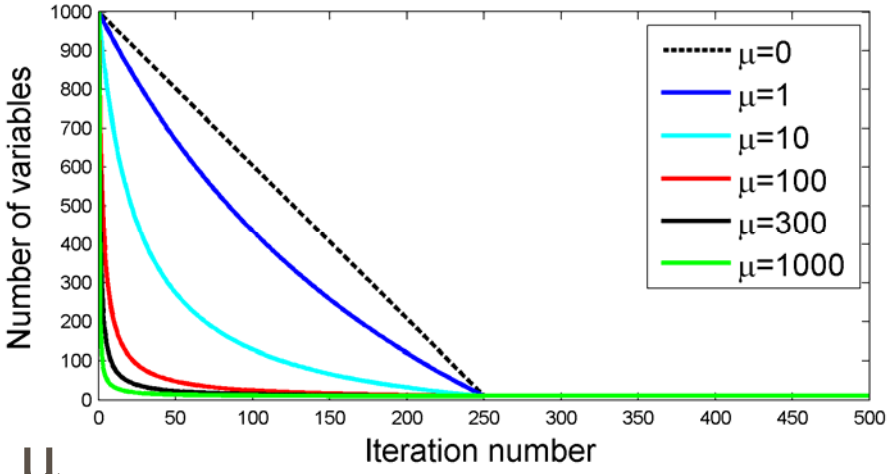
Feature Selection with Annealing

- Inspired by
  - TISP (She '09)
  - Graduated Nonconvexity (Blake & Zisserman, '87)
- Iterates two steps
  - Gradient update step to decrease loss function

$$\beta \leftarrow \beta - \eta \frac{\partial L(\beta)}{\partial \beta}$$

  - Feature selection step
    - Removes some variables according to a schedule and coefficient magnitudes
- Needs about 500 iterations to converge
- Very fast
  - Most features are removed in the first few iterations
  - Iterations are faster and faster

Variable Elimination Schedule



- A sequence M_i, i=1..N^{iter}
- M_i specifies how many variables are kept at iteration i
- Inverse schedule with parameter μ

$$M_i = k + (p-k)\max(0, \frac{N^{iter} - 2i}{2i\mu + N^{iter}})$$

Computationally equivalent to 2-10 iterations with all p variables when $\mu \geq 100$

- Also tried exponential schedule with parameter $\xi < 1$

$$M_i = \max(k, p\xi^i)$$

- Inverse schedule works better for same CPU time

FSA Algorithm

Algorithm 1 Feature Selection with Annealing (FSA)

Input: Training examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$
Output: Trained classifier parameters β.

1: Initialize $\beta = 0$.
2: **for** i=1 to N^{iter} **do**
3: Update $\beta \leftarrow \beta - \eta \frac{\partial L(\beta)}{\partial \beta}$
4: Keep the M_i variables with highest $\|\beta_j\|^2$ and renumber them $1, ..., M_i$.
5: **end for**

Convergence and Consistency

Theorem. Let M_e be a bounded and monotone annealing schedule satisfying $p \geq M_e \geq k$ and $\eta < 4/\|X\|_2^2$ where $\|X\|_2$ is the spectral norm

1. The algorithm converges in the sense that $\lim_{e\to\infty} \beta^{(e)}$ exists
2. Furthermore, assume that

$$\lim_{N\to\infty} \mathbf{X}^T \text{diag}\left\{ \frac{e^{\mathbf{x}_i^T \beta^*}}{(1+e^{\mathbf{x}_i^T \beta^*})^2} \right\}_{i=1}^{N} \mathbf{X}/N \to \mathcal{I}^*$$

which is positive definite and $k \geq \|\beta^*\|_0$, then there exists a slow enough schedule $\{M_e\}$ such that any $\beta^{(e)}$ for e sufficiently large is a consistent estimator of β^* as $N \to \infty$ and $P(\text{supp}(\beta^*) \subset \text{supp}(\beta^{(e)}) \to 1$

Barbu & She, 2013

Algorithm Parameters

- Number k of variables to select
  - Trade-off between speed and accuracy
  - Assumed known (e.g. obtained by cross-validation or with AIC/BIC)
- Learning rate η in $\beta \leftarrow \beta - \eta \frac{\partial L(\beta)}{\partial \beta}$
  - Any value up to a max value
  - Usually η=20
- Annealing parameter μ
  - Smaller μ = less greedy = better feature selection
  - Usually μ =300
- Number of iterations N^{iter}
  - Need about 200 more iterations after k variables are selected
  - Usually N^{iter}=500

Classification Loss Functions

- Given training set $D = \{(\boldsymbol{x}_i, y_i) \in \mathbb{R}^p \times \{-1, 1\}, i = \overline{1, N}\}$
- Generic loss function:

$$L_D(\beta) = \frac{1}{N} \sum_{i=1}^{N} L(y_i \beta^T \mathbf{x}_i) + s\|\beta\|_2^2$$

- Logistic loss:

$$L(x) = \log(1 + e^{-x})$$

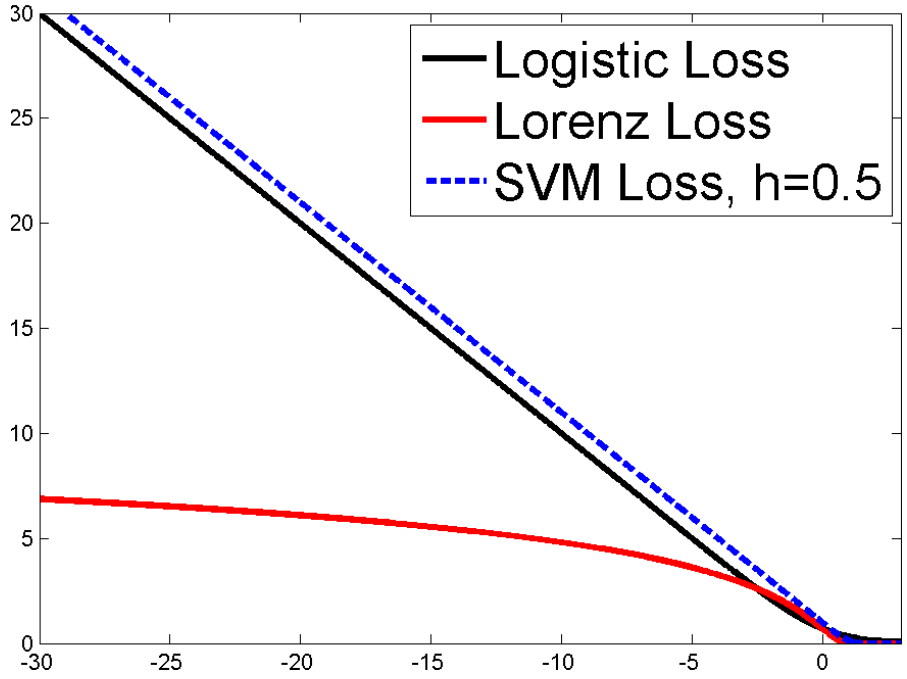
- Huberized SVM loss:

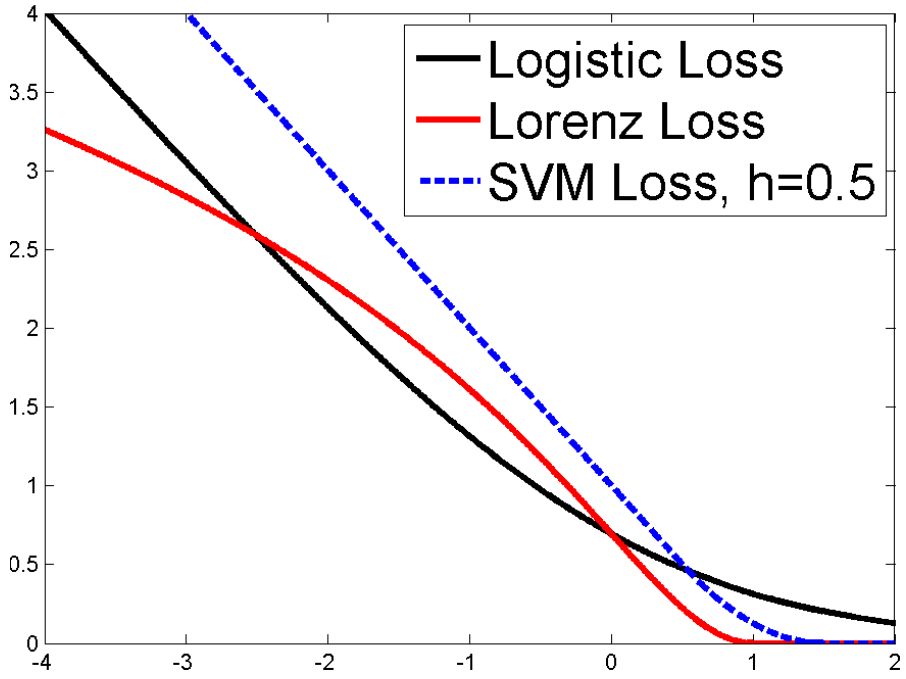
$$L(x) = \begin{cases} 0 & \text{if } x > 1 + h \\ \dfrac{(1 + h - x)^2}{4h} & \text{if } |1 - x| \leq h \\ 1 - x & \text{if } x < 1 - h \end{cases}$$

Loss Functions for Classification

- Lorenz loss – robust to label noise

$$L(x) = \begin{cases} 0 & \text{if } x > 1 \\ \ln(1 + (x-1)^2) & \text{else} \end{cases}$$





Regression Loss Function

- Given training set

$$D = \{(\boldsymbol{x}_i, y_i) \in \mathbb{R}^p \times \mathbb{R}^d, i = \overline{1, N}\}$$

- Regression loss function:

$$L_D(\beta) = \frac{1}{N} \sum_{i=1}^{N} \|y_i - \beta^T \mathbf{x}_i\|^2 + s\|\beta\|_2^2$$

Ranking Loss Function

- Given training set $\{\boldsymbol{x}_i \in \mathbb{R}^p \times, i = \overline{1, N}\}$ and pairs of true rankings

$$C \subset \{1, ..., N\} \times \{1, ..., N\}$$

with weights r_{ij}

- Ranking loss function:

$$L_C(\boldsymbol{\beta}) = \frac{1}{|C|} \sum_{(i,j) \in C} \ln(1 + \exp[\boldsymbol{\beta}^T(\boldsymbol{x}_i - \boldsymbol{x}_j)]) - \\ - \sum_{(i,j) \in C} r_{ij} \boldsymbol{\beta}^T(\boldsymbol{x}_i - \boldsymbol{x}_j) + s\|\beta\|^2$$

Simulation Experiments

- Correlated predictors for p≤10,000

$$\mathbf{x} \sim \mathcal{N}(0, \Sigma),$$
$$\Sigma_{ij} = \delta^{|i-j|}, \delta = 0.9$$

- Uniformly correlated predictors for large data

$$\mathbf{x} = \alpha z \mathbf{1}_{p\times 1} + \boldsymbol{\epsilon},$$
$$\boldsymbol{\epsilon} \sim N(0, I_M),$$
$$z \sim N(0, 1), \alpha = 0.5$$

Simulation Experiments

- Linearly separable classification data:

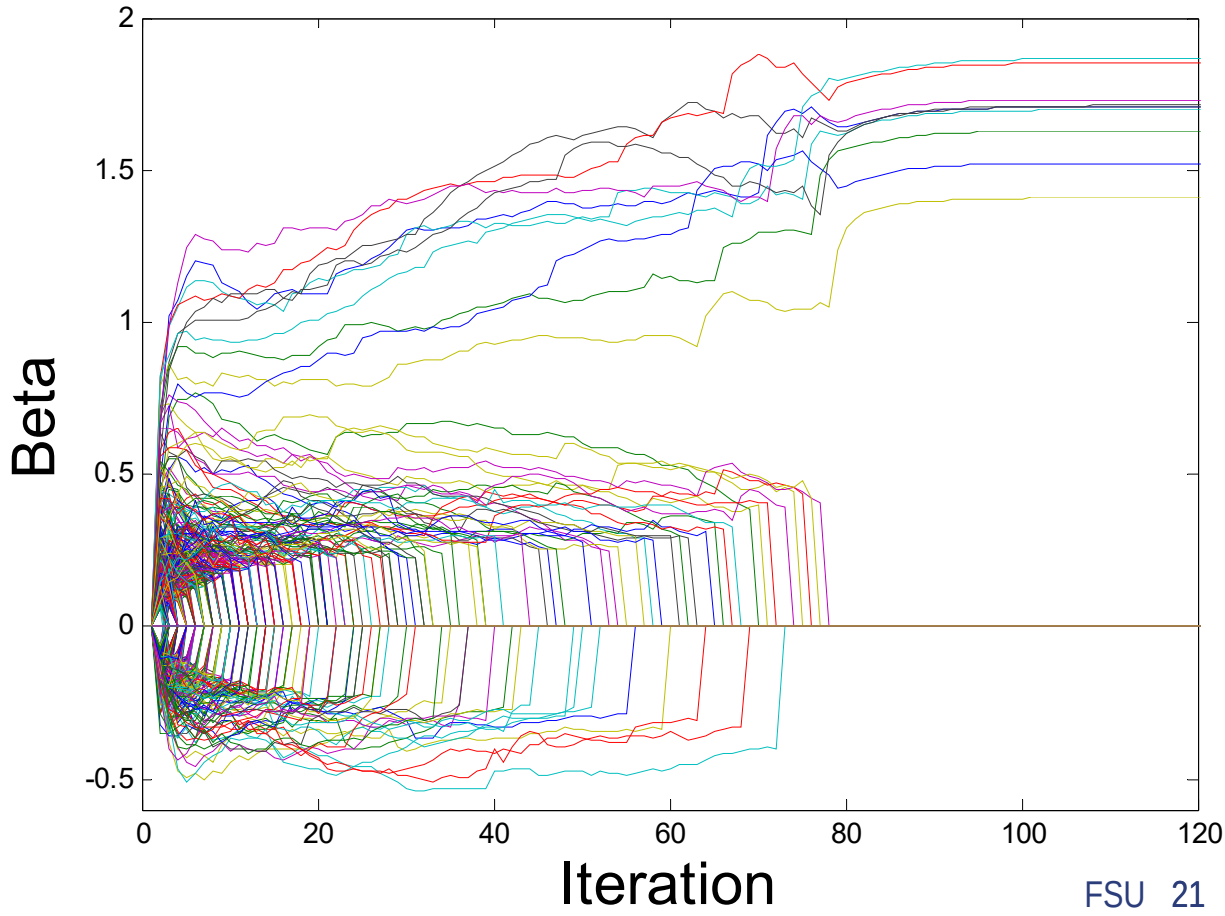
$$y = \begin{cases} 1 & \text{if } \sum_{i=1}^{k^*} x_{10i} > 0 \\ 0 & \text{otherwise} \end{cases}$$

- Noisy version
  - 10% of labels are random
  - $\approx$ 5% of labels switched
- Only variables 10i, i=1…k* are relevant
- Linear regression data

$$y = \sum_{i=1}^{k^*} x_{10i} + \epsilon, \ \epsilon \sim \mathcal{N}(0, 1)$$

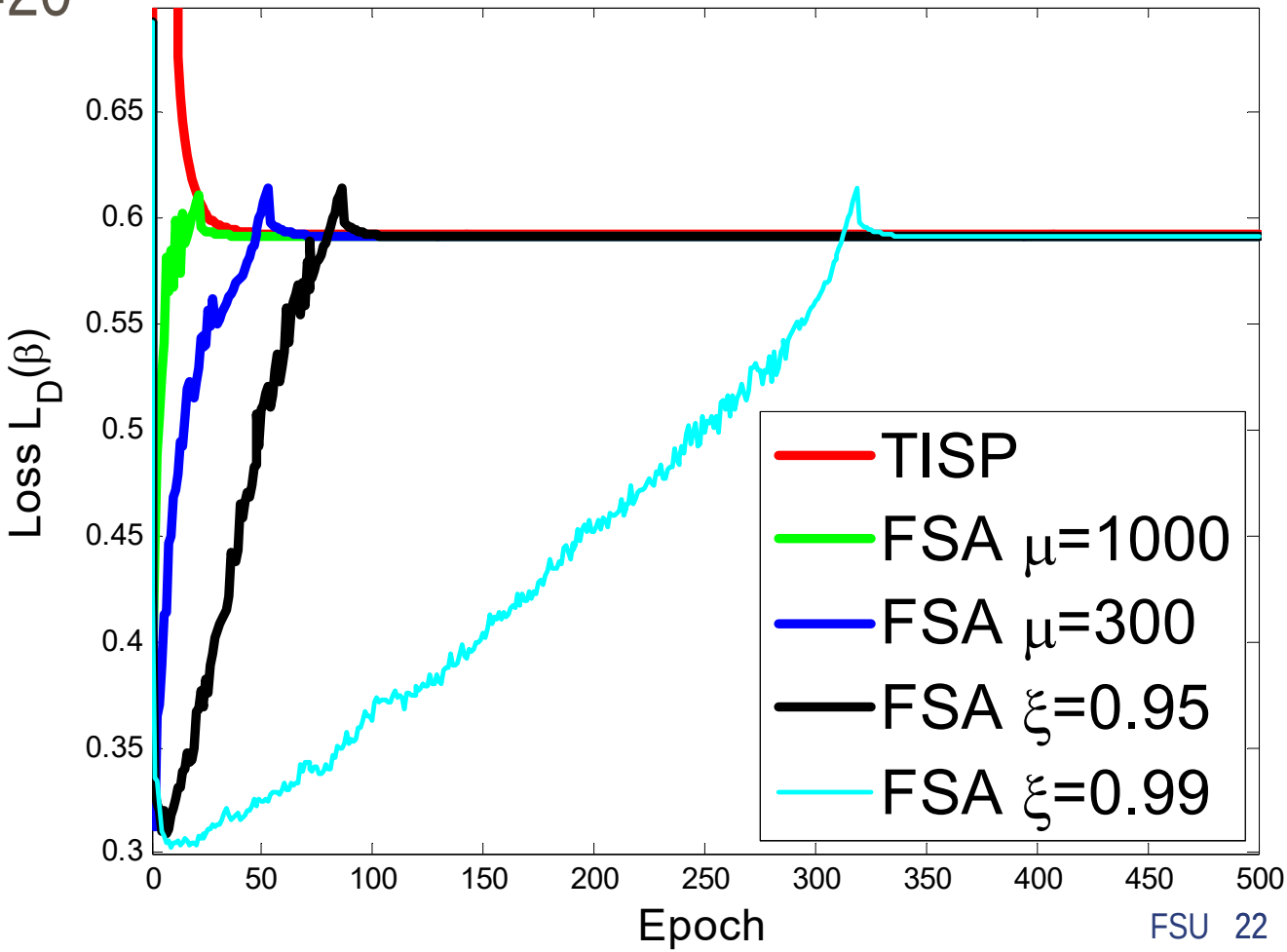
Coefficients Example

- Classification data
- N=1000, p=1000, k=k*=10 with shrinkage s=0.001
- Schedule with μ=300
- Learning rate η=20



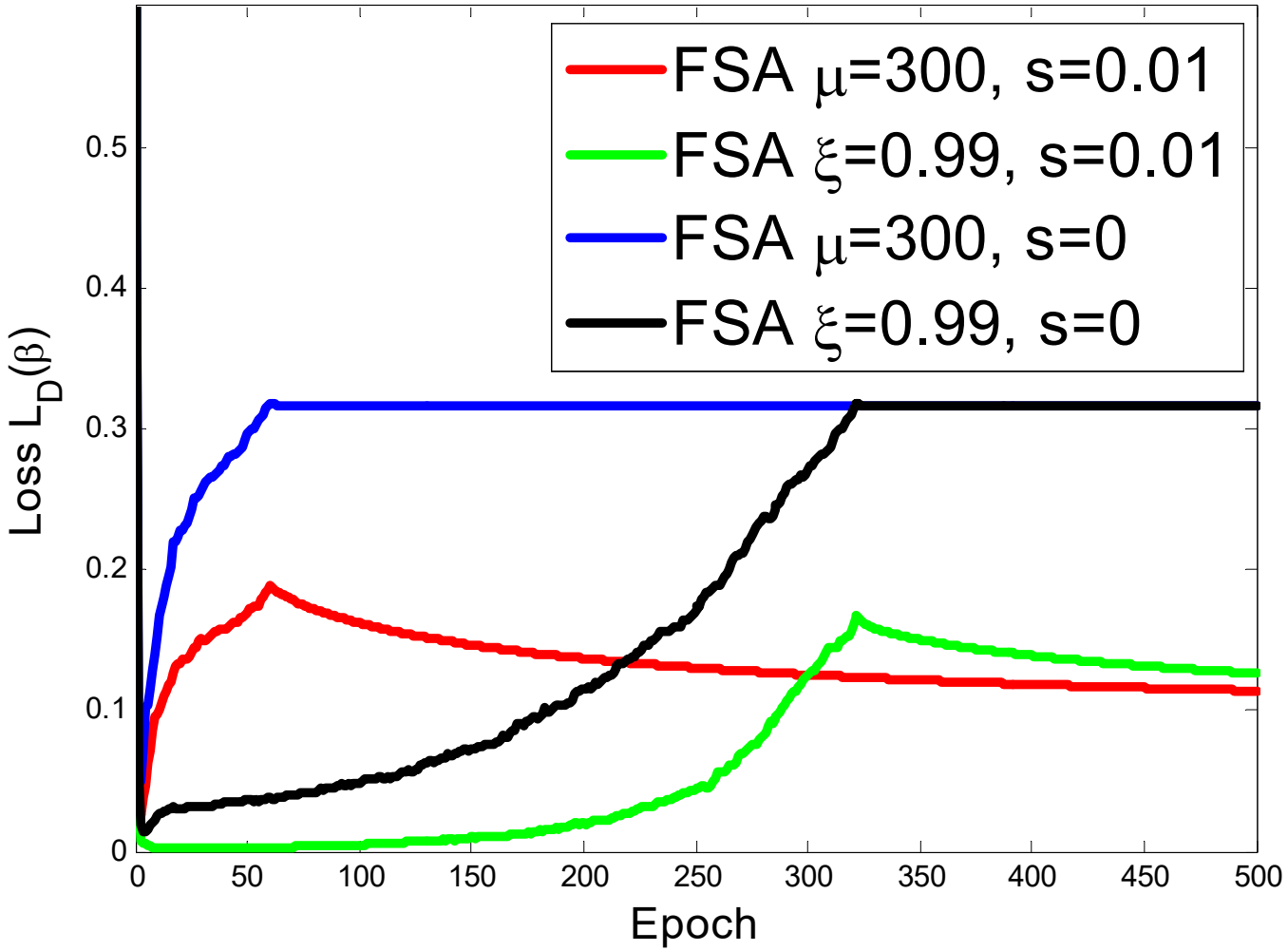
Loss Example

- Classification data,
- N=3000, p=1000, k=k*=10, s=0.01
- Learning rate η=20



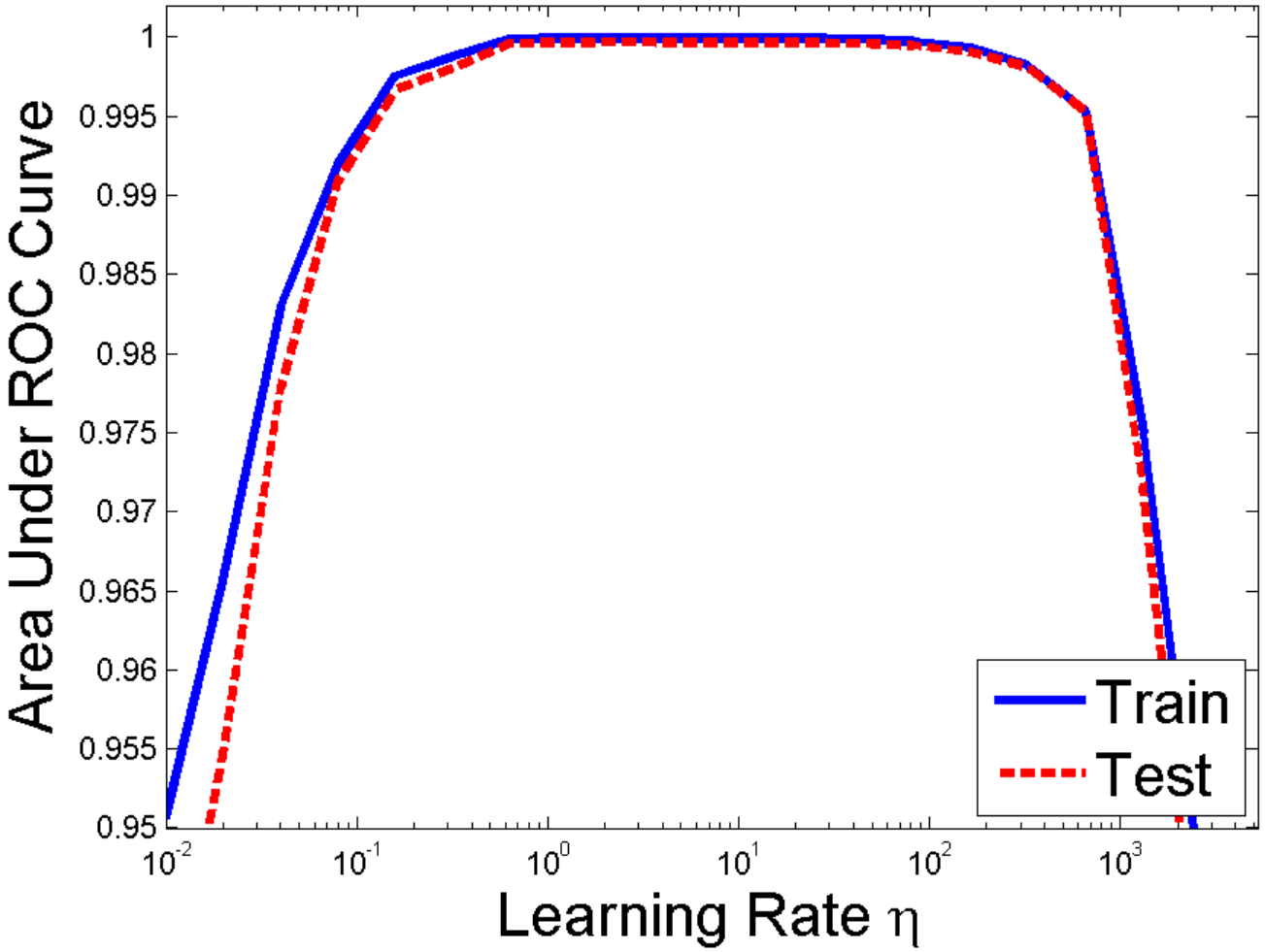
Loss Example - Regularization

- Classification data, N=3000,p=1000,k=10
- Learning rate η=20



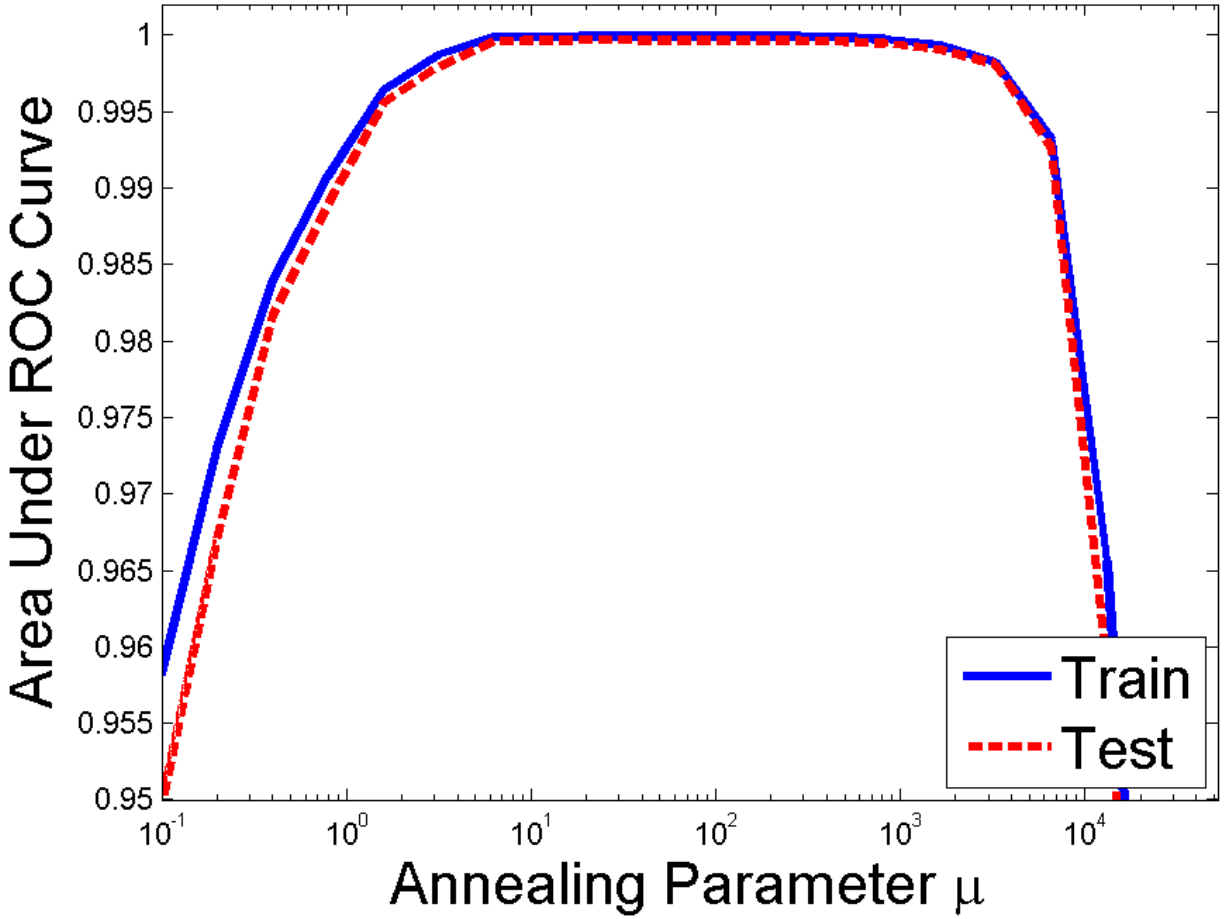
Stability to Learning Rate

- Classification data, N=1000, p=1000, k=10, μ=300
- Averaged over 10 runs



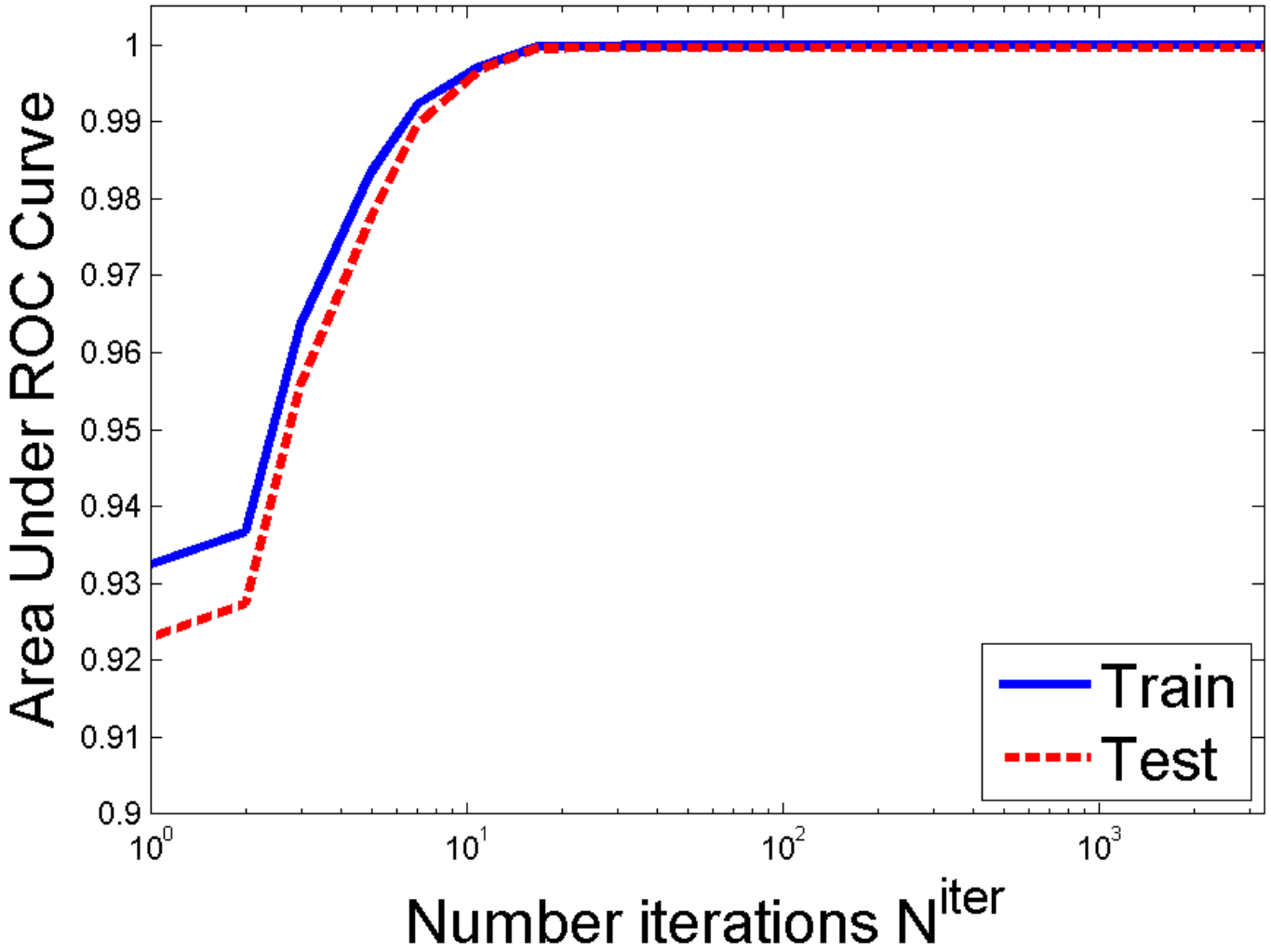
Stability to Annealing Parameter

- Classification data, N=1000, p=1000, k=10
- Learning rate η was changed so that $\eta = \mu/10$
- Averaged over 10 runs



Stability to N^{iter}

- Classification data, N=1000, p=1000, k=10, η=20, μ=300
- Averaged over 10 runs



Simulation Experiments

Algorithms

- FSA, FSV, FSR – FSA with the Logistic, SVM and Lorenz losses
  - μ=300, N^{iter}=500, s=0.01, η=20
- QTP – the Quantile TISP algorithm (She 2012)
- L_1 – L_1 - penalized logistic regression
  - Interior point method algorithm (Koh et al, JMLR 2007)
- MCP, SCAD – logistic regression with MCP/SCAD penalty
  - Based on Coordinate Descent (Breheny & Huang, 2011, Jiang & Huang 2011)
  - Implemented in two R packages: ncvreg and **cvplogistic**
- LB – Logitboost with univariate linear regressors as weak learners
- LB1 – Like LB, but selecting best learner from a random subset of 10% of all learners
- MRMR – The Maximum relevance minimum redundancy method (Peng et al, 2005)
  - Only feature selection

Results averaged over 100 runs

Simulation Experiments – Separable Data

			All-variable detection rate (DR)									
N	p	$k = k^*$	FSA	FSV	FSR	QTP	L_1	MCP	SCAD	LB	LB1	MRMR
300	1000	10	29	30	34	0	0	3	1	0	0	0
1000	1000	10	100	100	100	1	2	39	25	44	0	0
3000	1000	10	100	100	100	30	33	65	63	97	0	19
10000	1000	10	100	100	100	88	100	97	97	100	0	91
1000	1000	30	24	22	21	0	0	0	0	0	0	0
3000	1000	30	100	100	100	0	0	8	14	4	0	0
10000	1000	30	100	100	100	33	8	73	56	82	0	0
			Percent correctly detected (PCD)									
300	1000	10	86.1	84.7	86.0	37.9	42.4	64.0	55.6	61.3	23.1	31.9
1000	1000	10	100	100	100	67.8	72.4	88.5	85.7	92.3	26.3	56.8
3000	1000	10	100	100	100	91.1	91.5	95.9	95.4	99.6	29.1	85.4
10000	1000	10	100	100	100	98.8	100	99.6	99.6	100	31.8	99.1
1000	1000	30	93.8	92.4	92.6	47.4	41.5	66.8	61.2	62.4	29.0	28.9
3000	1000	30	100	100	100	78.7	68.6	91.1	91.7	90.4	37.8	52.1
10000	1000	30	100	100	100	97.2	93.9	98.3	97.3	99.3	43.8	82.5

Simulation Experiments – Separable Data

			Area under the ROC curve (AUC) - Test Set								
N	p	$k = k^*$	FSA	FSV	FSR	QTP	L_1	MCP	SCAD	LB	LB1
300	1000	10	.992	.990	.990	.899	.915	.955	.934	.950	.923
1000	1000	10	1.00	1.00	1.00	.947	.951	.965	.953	.967	.936
3000	1000	10	1.00	1.00	1.00	.987	.982	.973	.976	.971	.939
10000	1000	10	1.00	1.00	1.00	.998	.997	.978	.978	.971	.942
1000	1000	30	.996	.995	.995	.919	.923	.954	.937	.956	.936
3000	1000	30	1.00	1.00	1.00	.969	.954	.979	.976	.975	.948
10000	1000	30	1.00	1.00	1.00	.997	.985	.987	.984	.980	.956
			Training Time (sec)								
300	1000	10	0.03	0.03	0.04	0.02	17	87	68	0.13	0.01
1000	1000	10	0.06	0.07	0.15	0.05	469	352	282	0.44	0.09
3000	1000	10	0.23	0.15	0.49	0.21	705	1122	1103	1.3	0.18
10000	1000	10	1.8	1.8	1.8	1.4	2151	3738	3672	4.9	0.49
1000	1000	30	0.13	0.08	0.29	0.05	240	358	293	1.2	0.16
3000	1000	30	0.26	0.2	1.10	0.29	565	1840	1139	4.1	0.48
10000	1000	30	3.5	3.3	3.5	2.0	3914	3860	3710	14	1.5

Simulation Experiments – Noisy Data

			All-variable detection rate (DR)									
N	p	$k = k^*$	FSA	FSV	FSR	QTP	L_1	MCP	SCAD	LB	LB1	MRMR
300	1000	10	0	0	1	0	0	0	0	0	0	0
1000	1000	10	45	45	86	0	0	17	8	21	0	0
3000	1000	10	100	100	100	20	22	66	58	91	0	13
10000	1000	10	100	100	100	100	92	95	95	100	0	80
1000	1000	30	0	0	0	0	0	0	0	0	0	0
3000	1000	30	12	14	68	0	0	2	5	1	0	0
10000	1000	30	99	99	100	7	0	60	49	60	0	0
			Percent correctly detected (PCD)									
300	1000	10	44.5	38.9	43.7	30.7	41.2	46.7	45.8	47.8	21.8	29
1000	1000	10	92.5	91.4	98.5	58.8	65.3	81.2	78.9	84.4	25.4	49.3
3000	1000	10	100	100	100	88.2	87.8	95.5	94.2	99.1	29.1	82.5
10000	1000	10	100	100	100	100	99.2	99.5	99.5	100	31.4	97.9
1000	1000	30	49.5	45.0	53.7	34.9	40.0	47.5	47.3	48.8	26.7	26.4
3000	1000	30	92.4	92.3	98.7	67.5	63.7	84.0	83.9	82.8	32.9	47.6
10000	1000	30	100	100	100	93.7	90.3	97.5	96.8	98.3	40.7	78.2

Simulation Experiments – Noisy Data

			Area under the ROC curve (AUC) - Test Set								
N	p	$k = k^*$	FSA	FSV	FSR	QTP	L_1	MCP	SCAD	LB	LB1
300	1000	10	.890	.868	.885	.834	.880	0.877	0.865	.885	.863
1000	1000	10	.943	.940	.946	.890	.907	.914	.906	.915	.888
3000	1000	10	.950	.950	.950	.935	.934	.927	.923	.924	.895
10000	1000	10	.950	.950	.950	.950	.949	.933	.932	.924	.897
1000	1000	30	.905	.889	.904	.845	.885	.876	.873	.898	.887
3000	1000	30	.945	.943	.949	.906	.911	.926	.919	.925	.902
10000	1000	30	.950	.950	.950	.942	.938	.937	.933	.932	.908
			Training Time (sec)								
300	1000	10	0.04	0.04	0.04	0.04	22	67	65	0.17	0.02
1000	1000	10	0.14	0.13	0.14	0.12	412	218	211	0.53	0.06
3000	1000	10	0.49	0.46	0.48	0.36	1094	385	367	1.6	0.17
10000	1000	10	2.0	2.0	2.1	1.6	13921	653	599	5.3	0.5
1000	1000	30	0.30	0.29	0.30	0.17	791	246	239	1.6	0.17
3000	1000	30	1.0	1.0	1.0	0.55	1862	522	504	4.7	0.48
10000	1000	30	3.8	3.8	3.8	2.1	15949	763	719	16	1.6

Simulation Experiments – Regression

Data Params.			**All-variable detection rate (DR)**							**Percent correctly detected (PCD)**						
N	p	k	FSA	QTP	L1	EL	L2	MCP	SCAD	FSA	QTP	L1	EL	L2	MCP	SCAD
300	1000	30	67	0	0	0	0	0	0	98.5	33.1	37.8	37.0	-	62.1	24.9
1000	1000	30	100	0	0	0	0	1	0	100	51	63.8	62.5	-	78.5	55.6
3000	1000	30	100	0	1	1	0	8	0	100	72	88.6	87.6	-	93.4	83.4
10000	1000	30	100	0	87	82	0	95	76	100	90	99.5	99.4	-	99.8	99.2
			RMSE							**Time**(s)						
N	p	k	FSA	QTP	L1	EL	L2	MCP	SCAD	FSA	QTP	L1	EL	L2	MCP	SCAD
300	1000	30	1.11	5.08	3.68	3.72	2.58	2.41	6.63	0.25	0.09	1.24	1.1	0.05	1.56	0.15
1000	1000	30	1.02	4.22	2.79	2.88	2.45	3.17	6.51	0.18	0.1	2.6	2.3	0.07	3.5	7.6
3000	1000	30	1.01	3.06	1.87	1.94	1.22	3.80	6.36	0.52	0.3	5.7	5.6	0.16	12	41
10000	1000	30	1.00	1.91	1.05	1.07	1.05	3.94	5.01	1.8	1.0	14	14	0.33	41	79

$$\mathbf{x} \sim \mathcal{N}(0, \Sigma), \Sigma_{ij} = \delta^{|i-j|}, \delta = 0.9$$

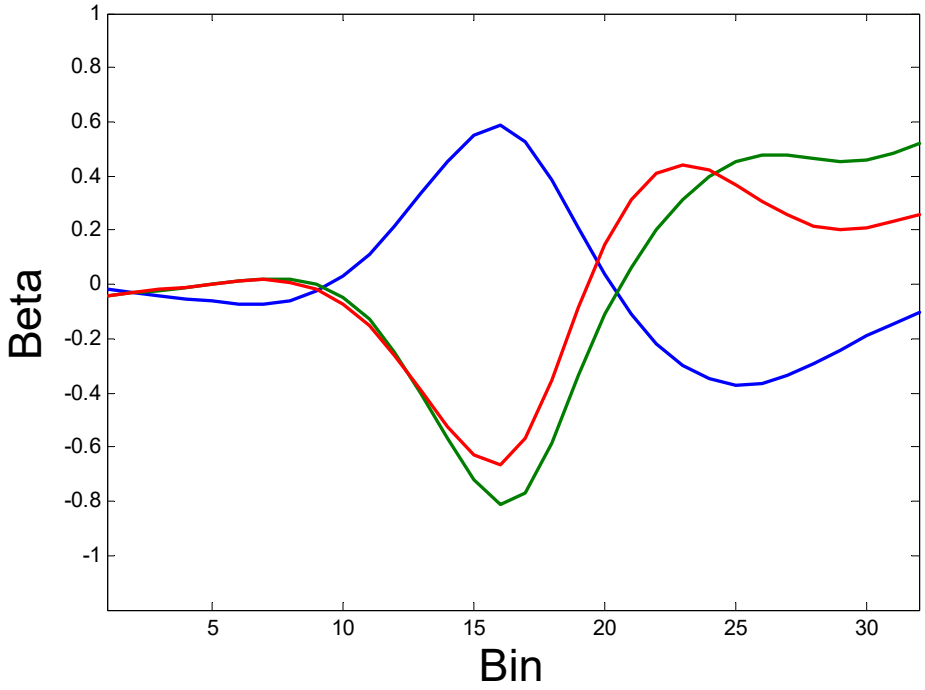
$$y = \sum_{i=1}^{k^*} x_{10i} + \epsilon, \ \epsilon \sim \mathcal{N}(0, 1)$$

Nonlinear Decision Boundary

- Linear boundary is not enough for many problems
- Use piecewise linear response functions on each variable

$$h_\beta(x) = \sum_{i=0}^{B} \beta_i u_i(x) = \beta^T \mathbf{u}(x)$$

- Use a smoothness prior



Nonlinear Decision Boundary

- Loss function

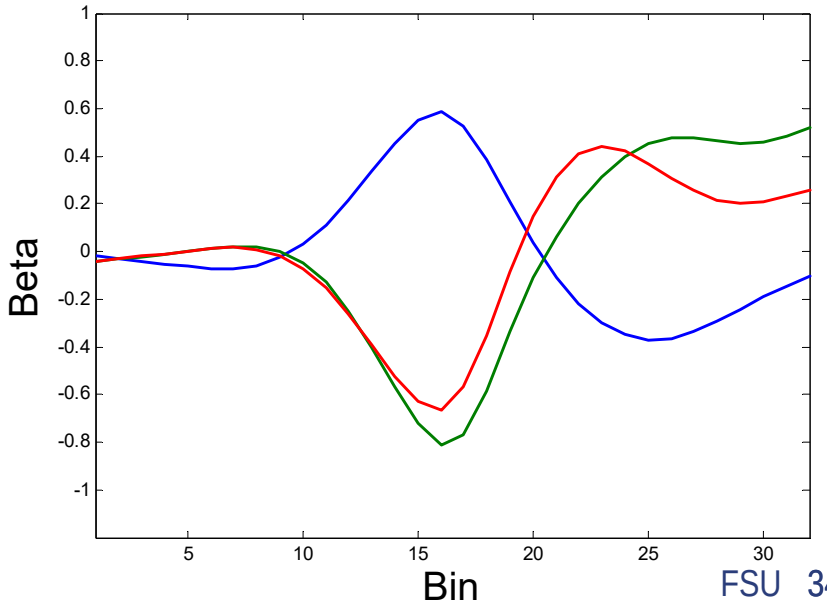
$$L_D(\beta) = \sum_{n=1}^{N} L(y_n h_\beta(\mathbf{x}_n)) + \sum_{i=1}^{p} \rho(\beta_i)$$

- Use second order smoothness prior

$$\rho(\beta_i) = c \sum_{j=2}^{B-1} (\beta_{i,j+1} + \beta_{i,j-1} - 2\beta_{ij})^2$$

- First order prior also good

$$\rho(\beta_i) = q \sum_{j=2}^{B} (\beta_{i,j} - \beta_{i,j-1})^2$$



UCI Data Experiments

- 3 UC Irvine Repository binary classification datasets

Method	Number of features k	Error train %	Error valid %	Error test %
URL_Reputation, $N^{train} = 2,000,000, N^{test} = 20,000, M = 3,200,000$				
SVM	all	-	-	1.8
Log Reg-SGD	all	-	-	1.6
Linear FSA-Lorenz	75,000	0.50	-	1.15
Confidence Weighted (Ma 2009)	500,000	0.23	-	1.0
Gisette, $N^{train} = 6,000, N^{valid} = 1,000, N^{test} = 6,500, M = 5,000$				
RFE	700	0.38	1.2	1.82
Linear FSA-Lorenz	500	0.2	0.1	1.83
MCP	1185	0.88	1.5	1.85
FSA-Lorenz, 2 bins	320	1.13	1.3	1.88
FSA-Lorenz, 5 bins	200	0.67	1.4	1.95
Parallel FS (Zhou 2014)	500	-	-	2.15
Dexter, $N^{train} = 300, N^{valid} = 300, N^{test} = 2,000, M = 20,000$				
Linear FSA-Lorenz*	300	0.	0.	8.55
Linear FSA-Lorenz*	93	2.0	1.0	8.7
Linear FSA-Lorenz	93	0.33	8.67	10.3
L1-penalized (Rosset 2003)	93	0.33	9.0	6.3

Face Data

- AFLW Dataset
- Training data: 999 images with 999 faces
  - Faces manually annotated with 21 landmarks
- Test data: 1555 images with 3861 faces
  - Different from training set

Face Keypoint Detection

- Feature pool
  - 61000 Haar features
  - 864 HOG (Histograms of Oriented Gradients)
- Gaussian pyramid
  - 4 scales / octave
- Training set:
  - About 6000 positives
  - About 1 billion negatives
- Memory limit: 16Gb
  - Can only fit 200,000 observations
  - Hard negative mining

Face Keypoint Detection

Monolithic classifier :

- 1500 features
- Hard negative mining
  - 10 iterations
  - Add about 20k negatives at each iteration

Detection Criteria:

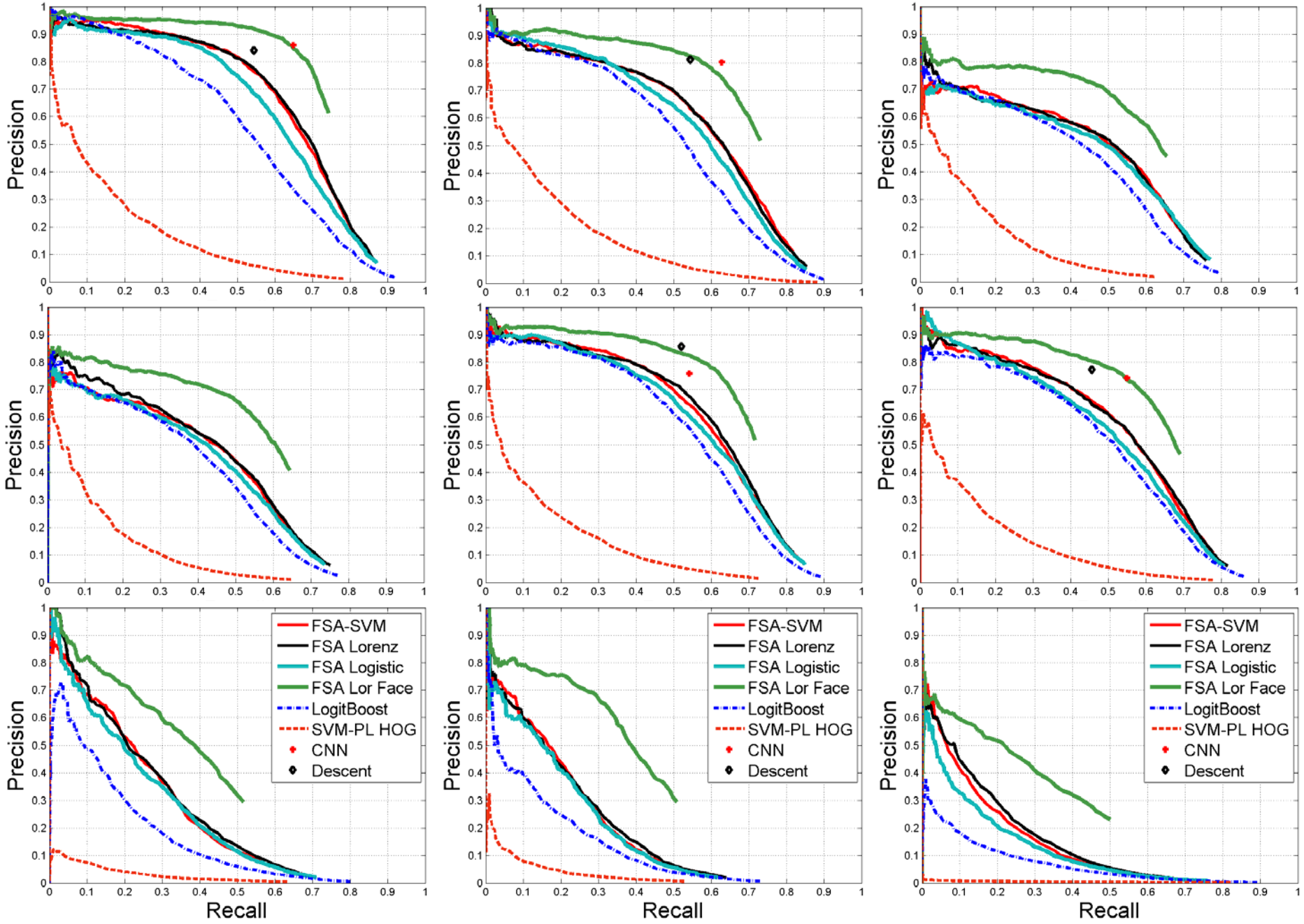
- Correct detection:
  There is a scale with a detection within 5% IED pixels from keypoint
- False positive:
  The detection is at least 10% IED from the keypoint of any face

Experiments on Real Data

Algorithms evaluated:

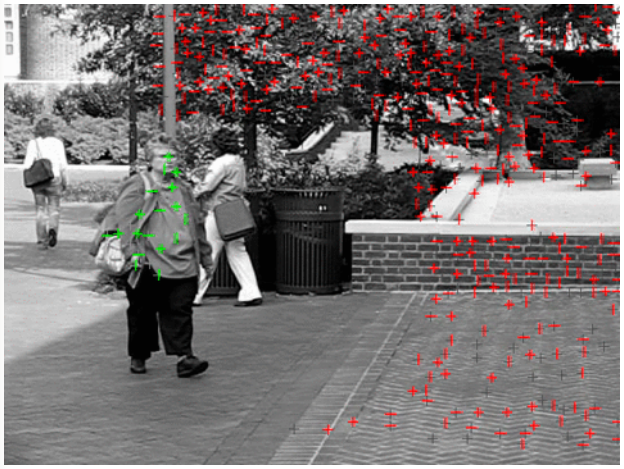
- FSA, FSA-SVM, FSA-Lorenz – FSA for the three losses
  - piecewise linear learners, μ=300
- SVM-PL HOG – SVM on HOG features
- LB – Logitboost with locally constant regressors
  - trained on 10% of weak learners at each iteration
- Descent – Supervised descent
  - Xiong & De la Torre, CVPR 2013
- CNN – Deep CNN based face point detection
  - Sun et al, CVPR 2013
- FSA Lor Face – FSA-Lorenz at locations constrained by face detection

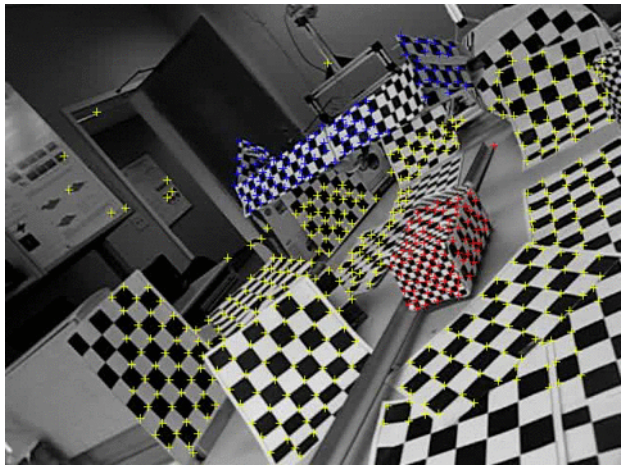
Keypoint Detection Results



Ranking for Motion Segmentation

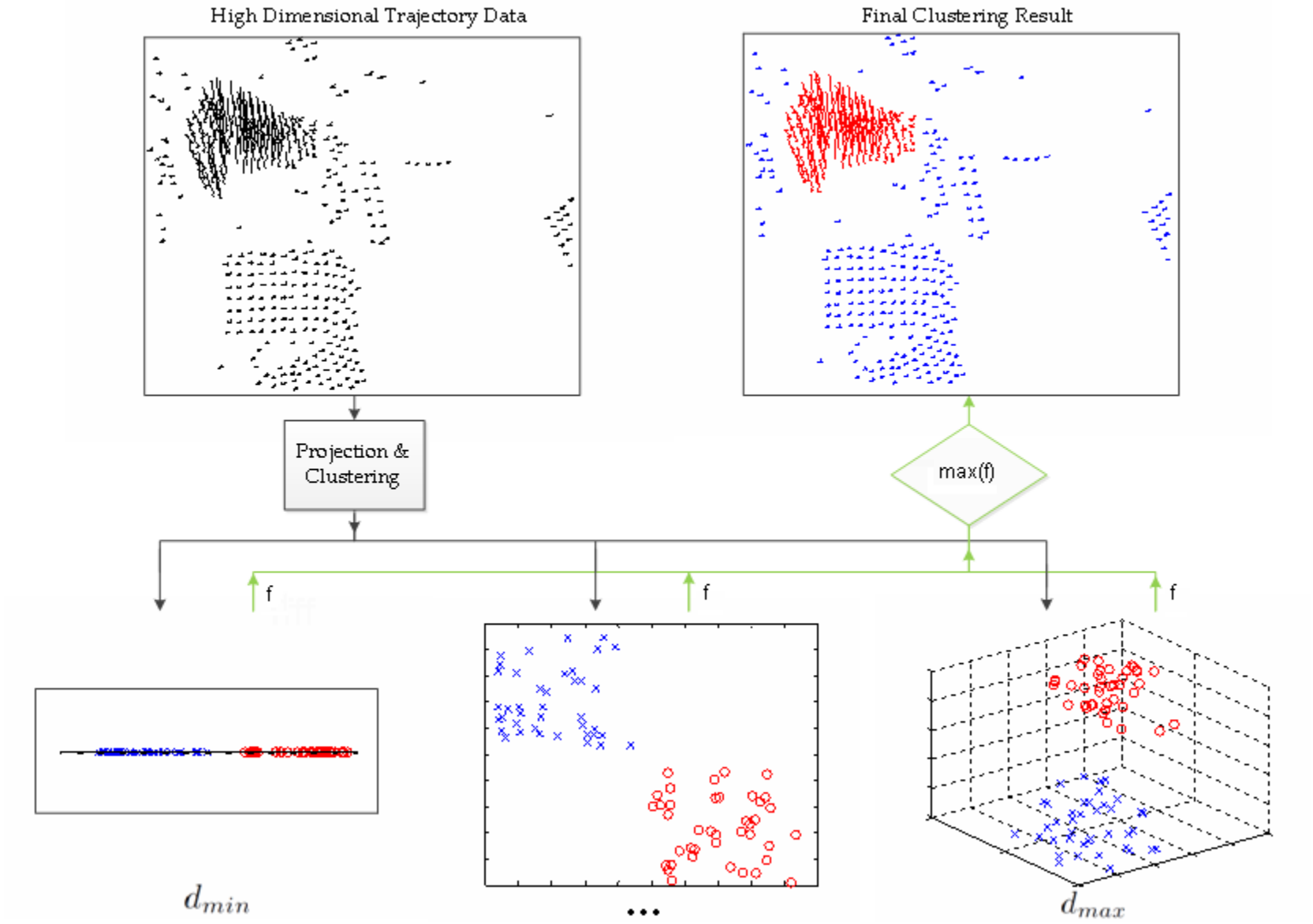
- Given a number of trajectories of feature points
- Group them by their common motion





- Many segmentation algorithms
- How to choose which is best for one sequence?

What Dimension to Project To?



Criterion for Comparing Segmentations

- Given a number of segmentations
- Compare them using ranking
  - Extract feature vector $\boldsymbol{x}$
  - Predict a ranking function

$$f_{\boldsymbol{\beta}}(\boldsymbol{x}) = \sum_{k=1}^{p} \boldsymbol{\beta}_k^T \boldsymbol{u}_k(x_k)$$

  - Select segmentation with largest value of $f_{\boldsymbol{\beta}}(\boldsymbol{x})$

- Ranking function is trained with FSA-Rank
  - Select k=40 features

Features

From each segmentation more than 2000 features are extracted

Two types of features:

- Likelihood features
  - Measure planarity based on rigid motion assumption
  - Based on distance from trajectories to fitted planes
    - Average distance, average squared distance
    - Average thresholded distance for different thresholds τ
- Prior features
  - Measure compactness of each cluster
  - Based on the kNN graph in a dimension d
  - Proportion of kNN edges that have differently labeled endpoints

$$F_G = \frac{|(i,j) \in E, L(i) \neq L(j)|}{|E|}$$

Motion Segmentation Algorithm

Algorithm 1 Motion Segmentation using Ranking

Input: The measurement matrix $W = [t_1, t_2, \ldots, t_P] \in \mathbb{R}^{2F \times P}$ whose columns are point trajectories, and the number of clusters K.

Preprocessing: Build the velocity measurement matrix $W' = (v(t_1), ..., v(t_P))$.

for $d = d_{min}$ **to** d_{max} **do**

1. Perform SVD: $W' = U\Sigma V^T$
2. Obtain P projected points as the columns of the $d \times P$ matrix

$$X_d = [v_1, ..., v_d]^T$$

where v_i is the i-th column of V.

3. Apply spectral clustering to the P points of X_d, obtaining segmentation L_d.
4. Extract feature vector $\boldsymbol{x}_d$ from segmentation L_d.
5. Compute the ranking

$$f_{\boldsymbol{\beta}}(L_d) = \sum_{k=1}^{p} \boldsymbol{u}_k^T(x_{dk})\boldsymbol{\beta}_k$$

end for

Output: The segmentation result L_d with the largest value of $f_{\boldsymbol{\beta}}(L_d)$.

Experiments

Methods Compared:

- RV – randomized voting (Jung et al, CVPR 2014)
- SC – spectral clustering (Lauer & Schnorr, ICCV 2009)
- SSC – sparse spectral clustering (Elhamifar & Vidal, CVPR 2009)
- VC – velocity clustering (Ding et al, DTCE 2013)
- RankBoost (Freund et al, JMLR 2003)
  - Same features as ours
  - Using RankBoost for ranking
  - 100 boosting iterations
- 10 fold cross-validation for Rankboost and FSA-Rank

Ranking Results

Method	RV	SC	SSC	VC	RankBoost		FSARank					
							Likelihood Features		Prior Features		All Features	
					Train	Test	Train	Test	Train	Test	Train	Test
Checkerboard (2 motion)												
Average	-	0.85	1.12	0.67	0.67	0.74	0.58	0.69	1.09	1.28	0.12	0.12
Median	-	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Traffic (2 motion)												
Average	-	0.90	0.02	0.99	0.69	0.72	0.80	0.76	4.25	4.25	0.59	0.58
Median	-	0.00	0.00	0.22	0.00	0.00	0.15	0.00	0.00	0.00	0.00	0.00
Articulated (2 motion)												
Average	-	1.71	0.62	2.94	2.05	2.26	2.30	2.27	1.32	1.32	1.32	1.32
Median	-	0.00	0.00	0.88	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
All (2 motion)												
Average	0.44	0.94	0.82	0.96	0.80	0.87	0.80	0.85	1.93	2.05	0.35	0.35
Median	-	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Checkerboard (3 motion)												
Average	-	2.15	2.97	0.74	0.85	2.60	0.74	0.74	4.10	4.22	0.49	0.49
Median	-	0.47	0.27	0.21	0.26	0.26	0.21	0.21	0.24	0.24	0.21	0.21
Traffic (3 motion)												
Average	-	1.35	0.58	1.13	4.15	4.24	1.13	1.13	4.05	4.05	1.73	1.07
Median	-	0.19	0.00	0.21	0.00	0.47	0.00	0.00	0.00	0.00	0.00	0.00
Articulated (3 motion)												
Average	-	4.26	1.42	5.65	3.66	18.09	5.32	5.32	3.19	3.19	3.19	3.19
Median	-	4.26	0.00	5.65	3.66	18.09	5.32	5.32	3.19	3.19	3.19	3.19
All (3 motion)												
Average	1.88	2.11	2.45	1.10	1.67	3.82	1.08	1.08	4.04	4.13	0.90	0.76
Median	-	0.37	0.20	0.22	0.20	0.32	0.20	0.20	0.20	0.20	0.00	0.00
All sequences combined												
Average	0.77	1.20	1.24	0.99	1.00	1.54	0.86	0.90	2.40	2.52	0.47	0.44
Median	-	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Conclusion

- Feature Selection with Annealing
  - Optimizes a constrained loss function
  - Gradually tightens the constraint
- Alternates two steps:
  - Parameter update
  - Variable removal according to a schedule
- Advantages:
  - Fast to train
  - Consistent estimator
  - Outperforms other methods in real and simulated problems
- Many applications:
  - Classification, regression, ranking

Future Work

- Other Object Detection applications
  - 3D faces, pedestrians, cars, bicycles, etc
  - Cells in microscopy images
  - Lymph nodes, organs, etc in CT, MRI
- Regression applications
  - Object segmentation
- Capturing feature interactions
  - Multi-dimensional response functions
- Clustering applications
  - Simultaneous feature selection and clustering

Discussion Points

- How does FSA work?
- What does annealing mean?
- What is the difference between TISP and FSA?
- Why is FSA faster than TISP?
- Why do we do simulations?
- How does FSA compare with penalized methods?
- How can we obtain a nonlinear model with feature selection?
- What is ranking good for?