Efficient Projection Algorithms onto the L_1 Ball for Learning Sparse Representations from High Dimension Data

Yoram Singer
Google

BASED ON JOINT WORK WITH:
JOHN DUCHI & TUSHAR CHANDRA (GOOGLE)
SHAI SHALEV-SHWARTZ (TTI)

MMDS WORKSHOP, STANFORD, JUNE 27, 2008

1

Feature Selection & Learning

THE HIGHER MINIMUM WAGE THAT WAS SIGNED INTO LAW ... WILL BE WELCOME RELIEF OF WORKERS ... THE 90 CENT-AN-HOUR INCREASE...

REGULATIONS
LABOUR
ECONOMICS

- Common approach to topic classification:
  - Select relevant features / tokens
  - Assign weights to tokens in order to achieve low classification error rate

Portfolio Design & Selection

- A large collections of investments tools (stocks, bonds, ETFs, cash, options, ...)
- Select a subset of the assets
- Distribute investments among selected assets, not necessarily evenly

Learning & Representation

- Many learning problems benefit from compact representation of the input space:
  spam classification, advertisements placement, web ranking, audio reconstruction, ...
- Often the learning is divides into two phases:
  - Find compact representation (CR)
  - Build a prediction mechanism from (on top) CR
- *Perform selection of features and learning a predictor <u>simultaneously</u>*

Two Forms of ERM

- Support vector machine learning

$$\arg\min_{\mathbf{w}} \sigma\|\mathbf{w}\|^2 + \frac{1}{m}\sum_{i=1}^{m}[1 - y_i(\mathbf{w}\cdot\mathbf{x}_i)]_+$$

PENALIZED

EMPIRICAL RISK

$$\arg\min_{\mathbf{w}\in S} \sigma\|\mathbf{w}\|^2 + \mathbf{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\ell(\mathbf{w};(\mathbf{x},y))\right]$$

- Portfolio design

$$\sum_{t=1}^{T}\log\left(\mathbf{w}_t\cdot\mathbf{x}_t\right) \text{ s.t. } \mathbf{w}_t\in\Delta$$

DOMAIN CONSTRAINED

EMPIRICAL RISK

$$\sum_{t=1}^{T}\ell\left(\mathbf{w}_t;(\mathbf{x}_t,y_t)\right) \text{ s.t. } \mathbf{w}_t\in S$$

Stoc. Grad. & Domain Constraints

$$\min_{\mathbf{w}} L(\mathbf{w}) \text{ s.t. } \|\mathbf{w}\|_1 \leq z$$

STOCHASTIC GRADIENT

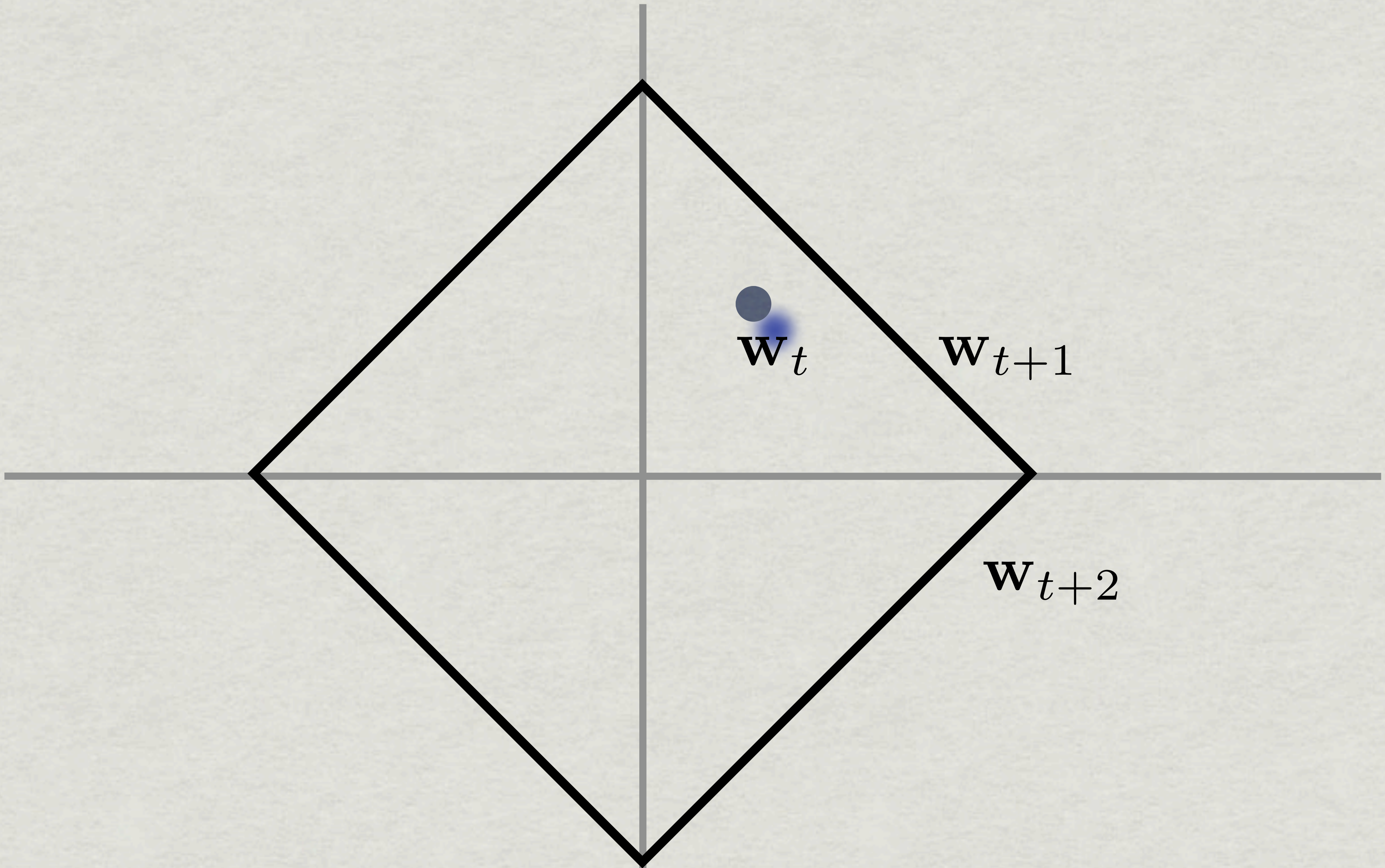
$$\mathbf{w}_{t+1} = \Pi_X(\mathbf{w}_t - \eta_t \nabla_t L)$$

LEARNING RATE / STEP SIZE

$$\Pi_X(\mathbf{w}) = \arg\min \{\|\mathbf{w} - \mathbf{v}\| \mid \mathbf{v} \in X\}$$

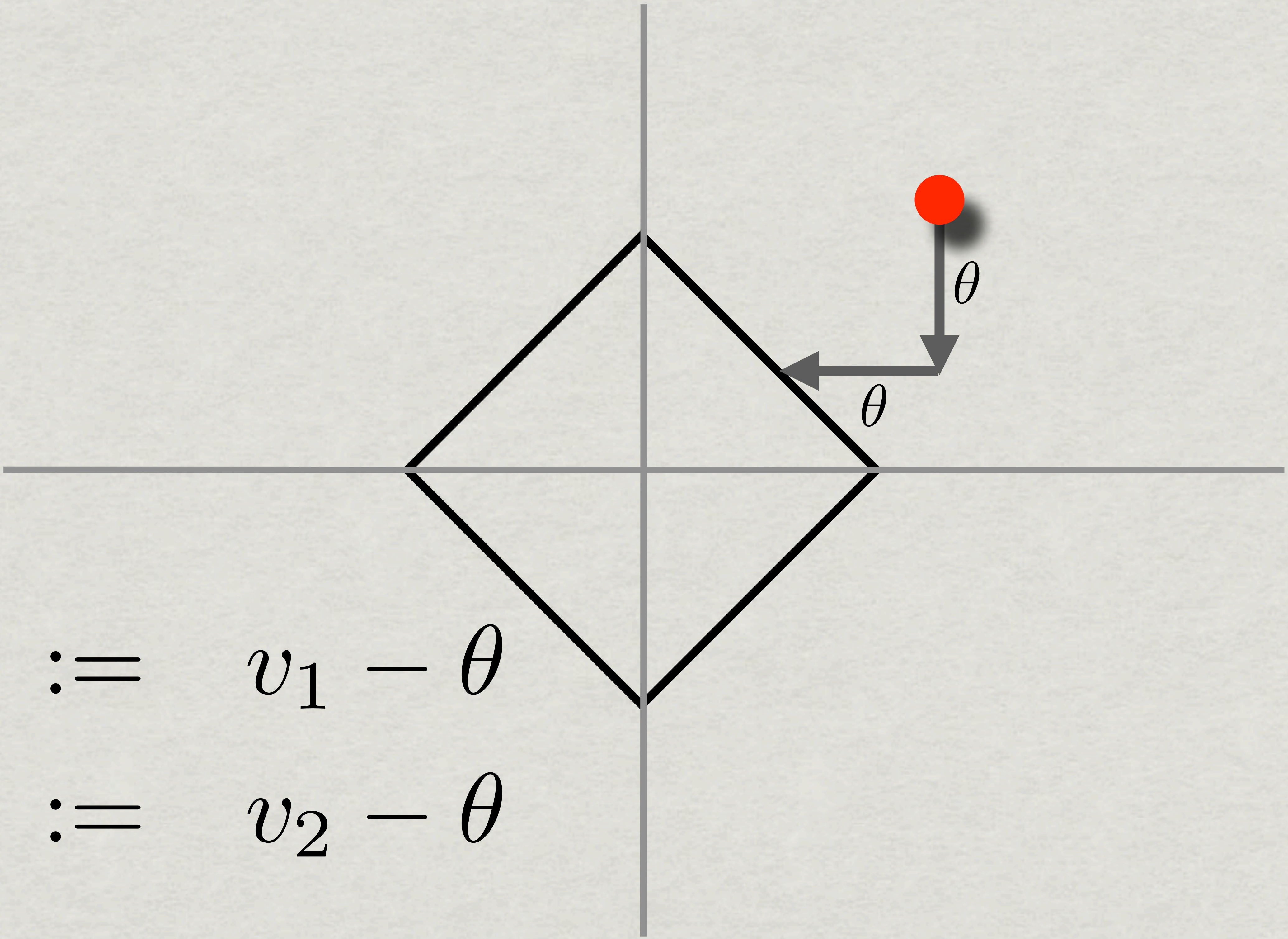
$$X = \{\mathbf{w} \mid \|\mathbf{w}\|_1 \leq z\}$$

Stoc. Grad. with ℓ_1 Constraints



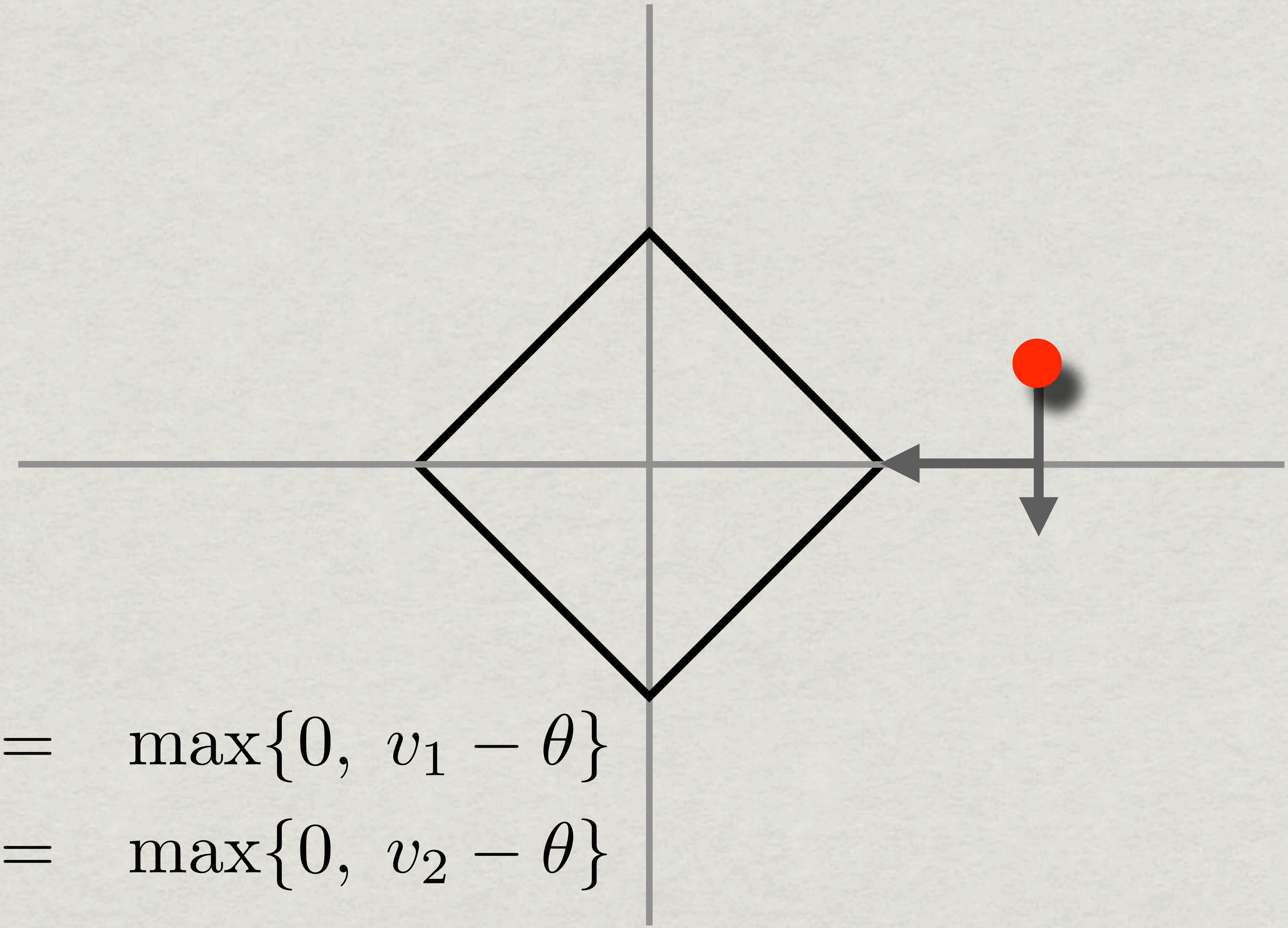
FOCUS MOSTLY ON EFFICIENT ALGORITHMS FOR EUCLIDEAN PROJECTIONS ONTO THE L_1 BALL IN HIGH DIMENSIONS

Projection onto ℓ_1 Ball



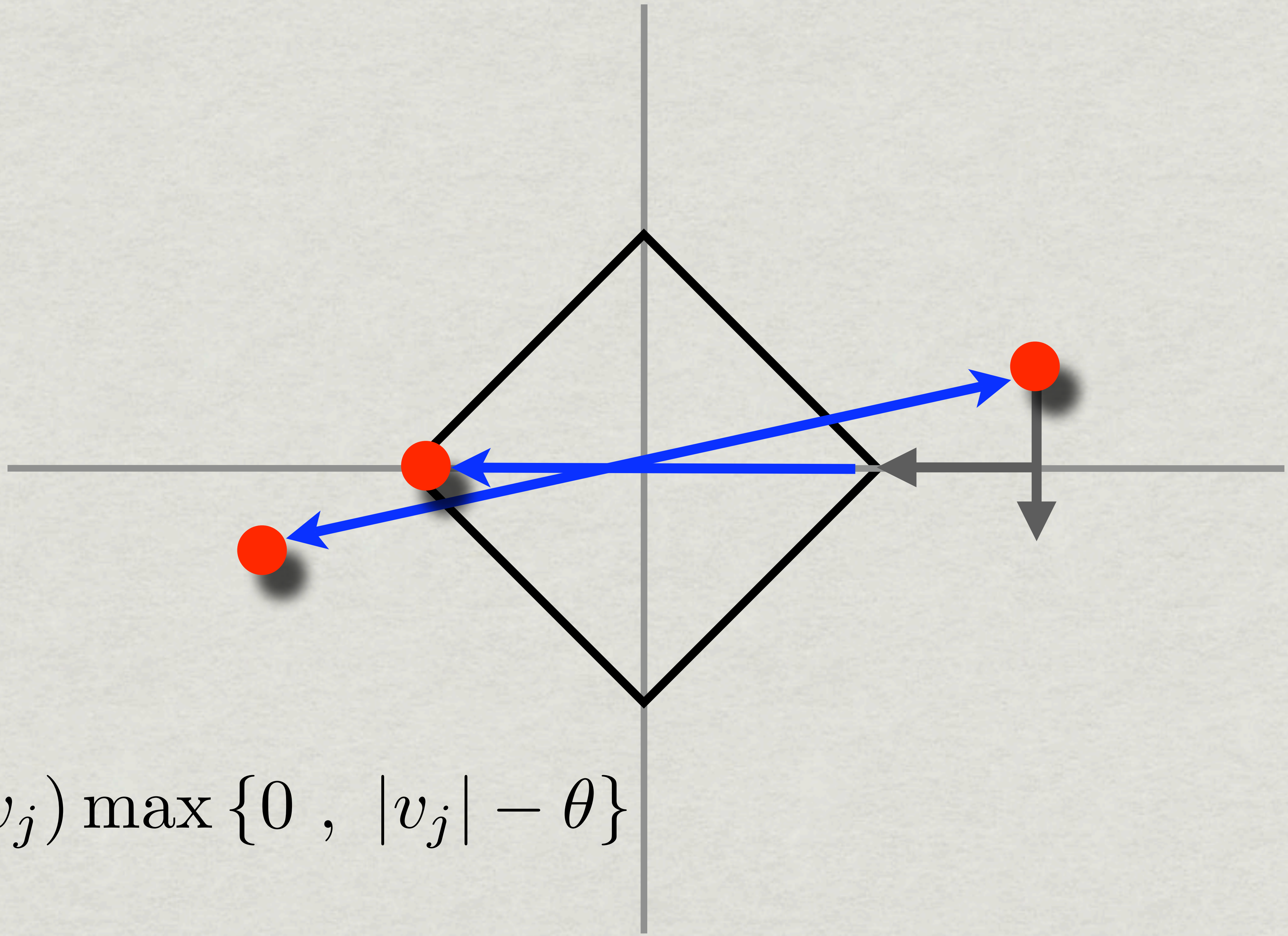
$$
\begin{aligned}
v_1 &:= v_1 - \theta \\
v_2 &:= v_2 - \theta
\end{aligned}
$$

Projection onto ℓ_1 Ball



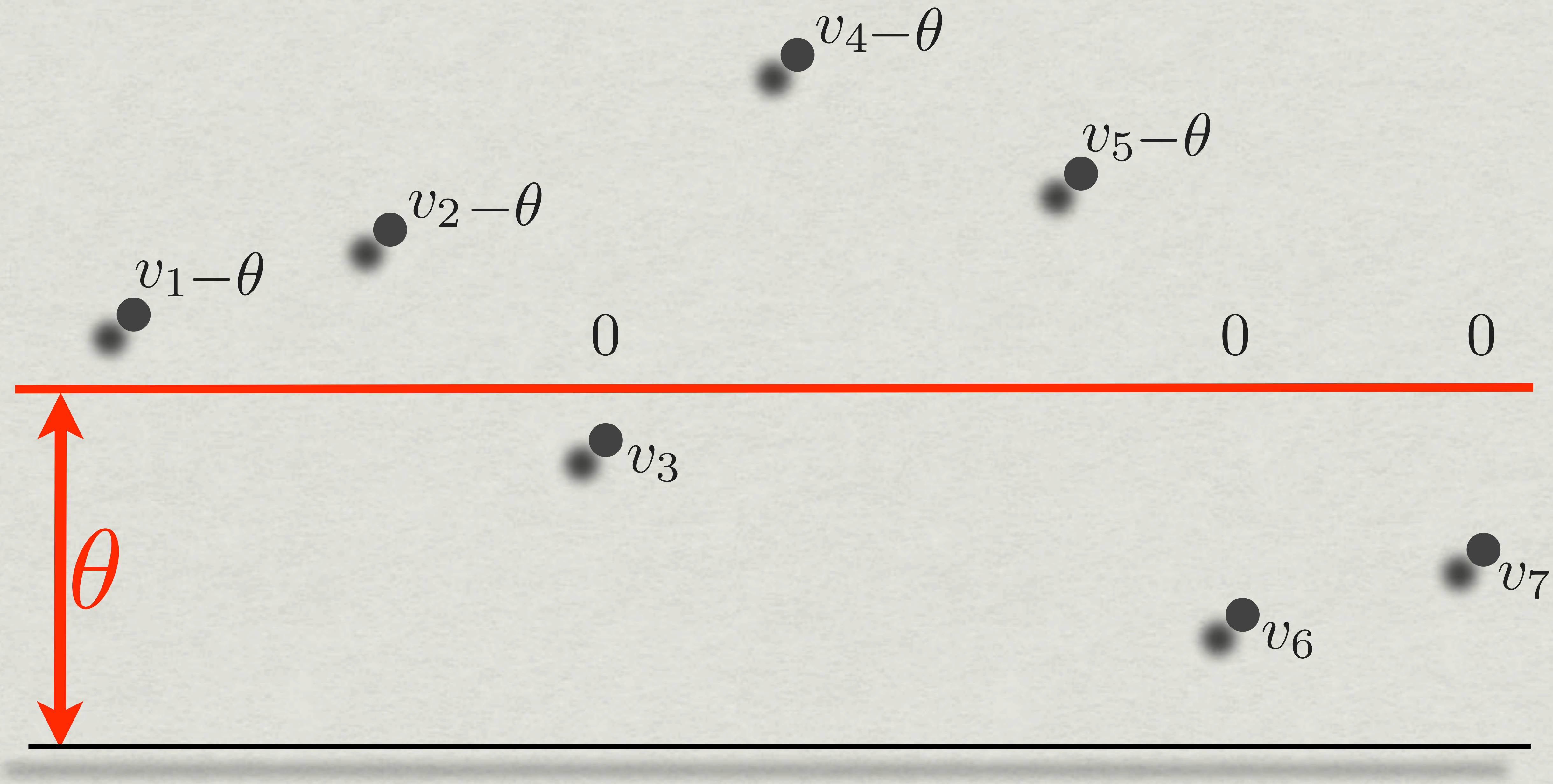
$$v_1 \quad := \quad \max\{0,\ v_1 - \theta\}$$
$$v_2 \quad := \quad \max\{0,\ v_2 - \theta\}$$

Projection onto ℓ_1 Ball (cont)

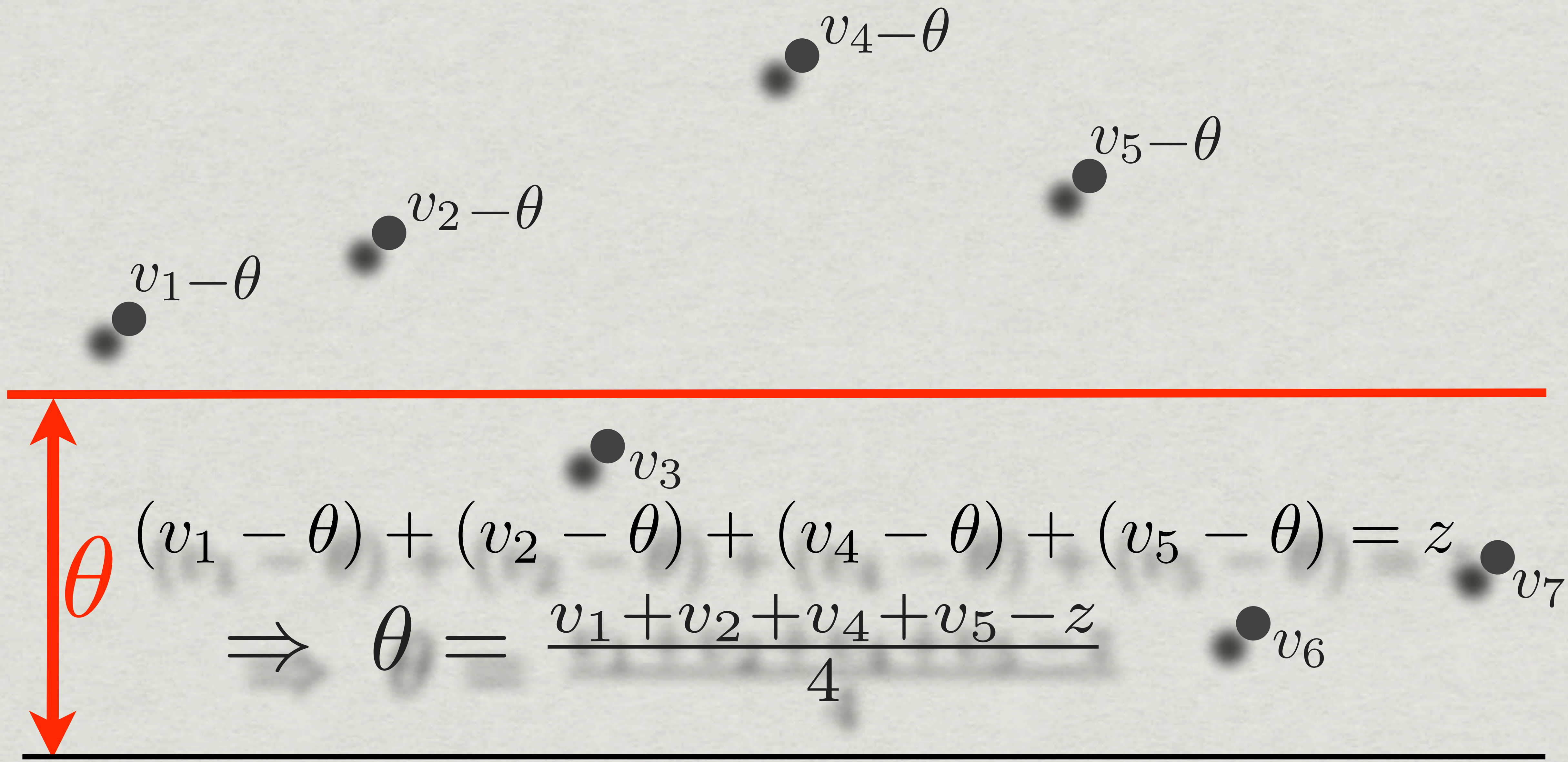


$$\mathrm{sign}(v_j)\max\{0\ ,\ |v_j| - \theta\}$$

Algebraic-Geometric View



Algebraic-Geometric View



$$(v_1 - \theta) + (v_2 - \theta) + (v_4 - \theta) + (v_5 - \theta) = z$$

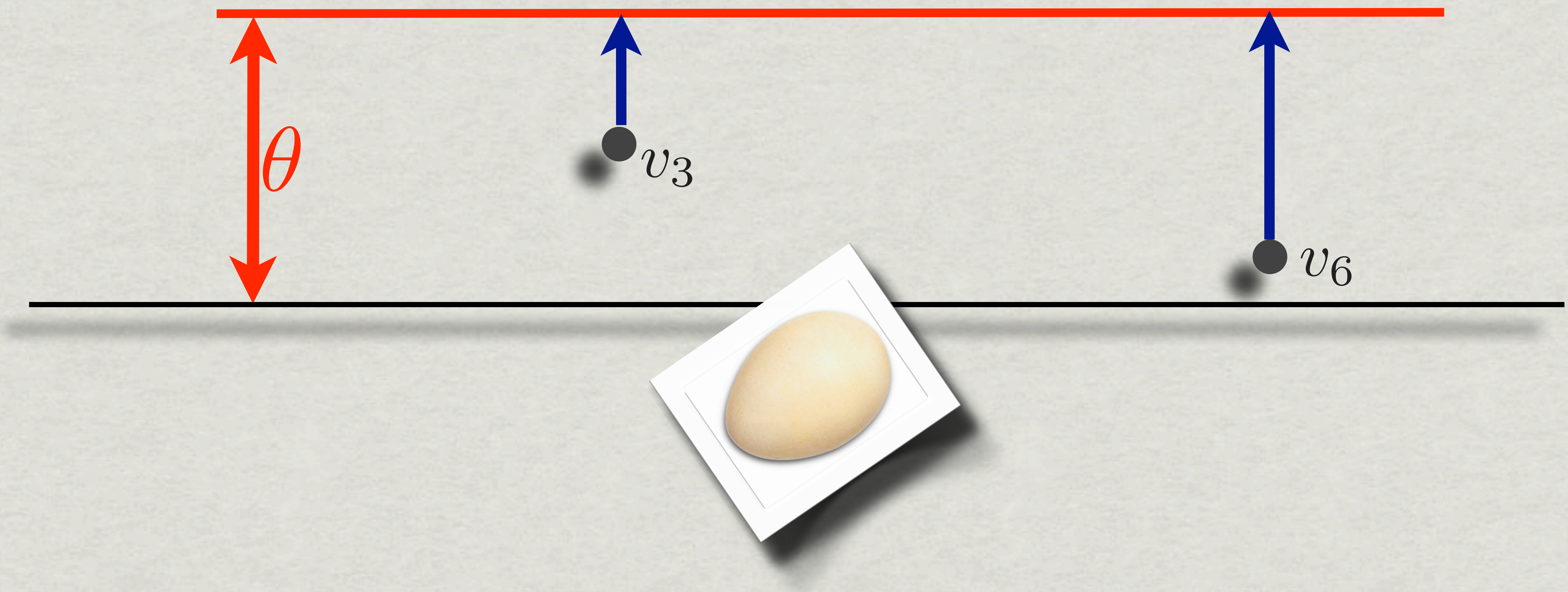
$$\Rightarrow \ \theta = \frac{v_1 + v_2 + v_4 + v_5 - z}{4}$$

Chicken and Egg Problem

- Had we known the threshold we could have found all the zero elements
- Had we known the elements that become zero we could have calculated the threshold

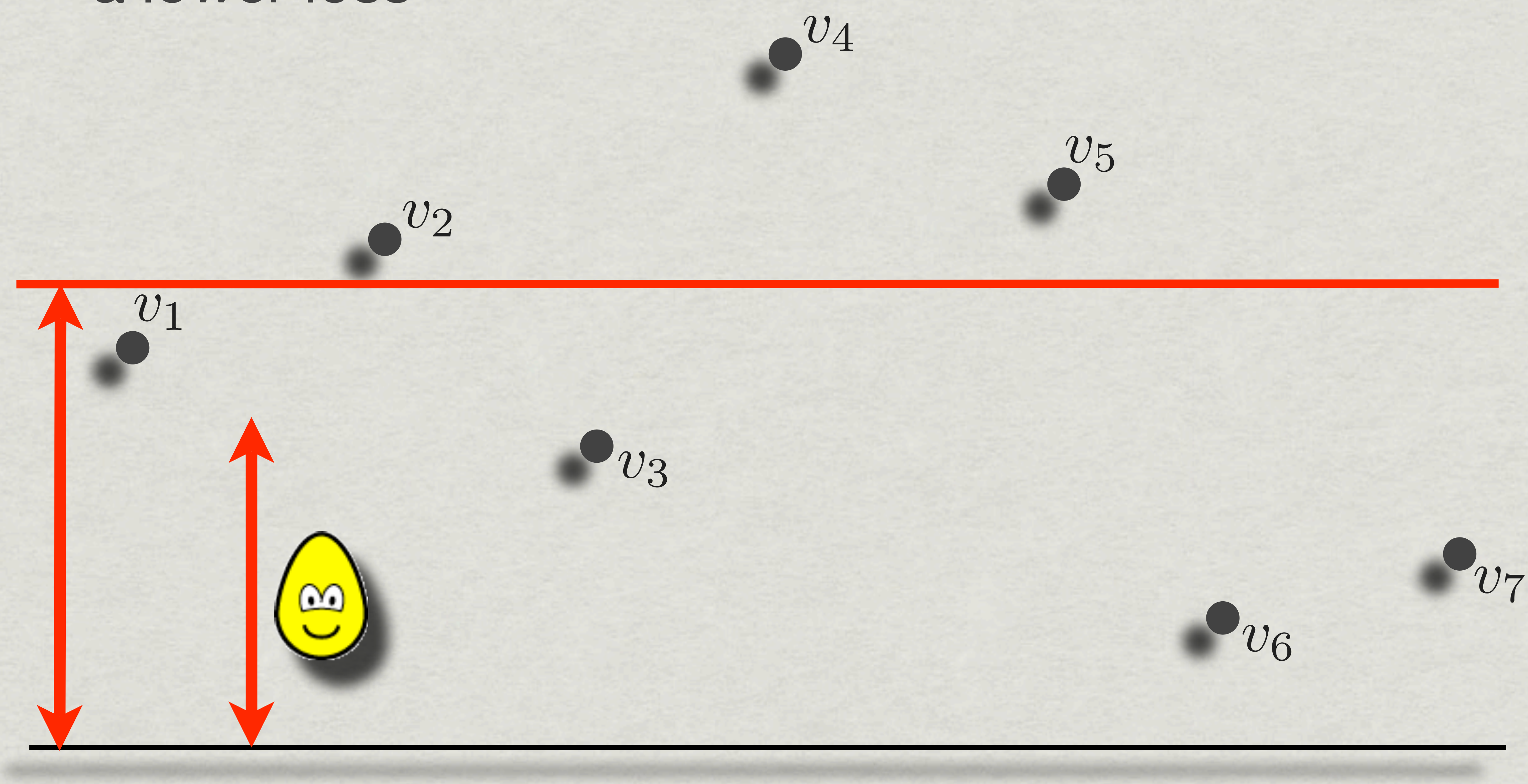
From Eggs to Omelette

If $v_j < v_k$ then if after the projection the k'th component is zero, the j'th component must be zero as well



From Eggs to Omelette

If two feasible solutions exist with k and k+1 non-zero elements then the solution with k+1 elements attains a lower loss



Calculating ℓ_1 Projection

- Sort vector to be projected

$$\Rightarrow \quad \mu_1 \geq \mu_2 \geq \mu_3 \geq \dots \geq \mu_n$$

- If j is a feasible index then

$$\mu_j > \theta \quad \Rightarrow \quad \mu_j > \underbrace{\frac{1}{j}\left(\sum_{r=1}^{j} \mu_r - z\right)}_{\theta}$$

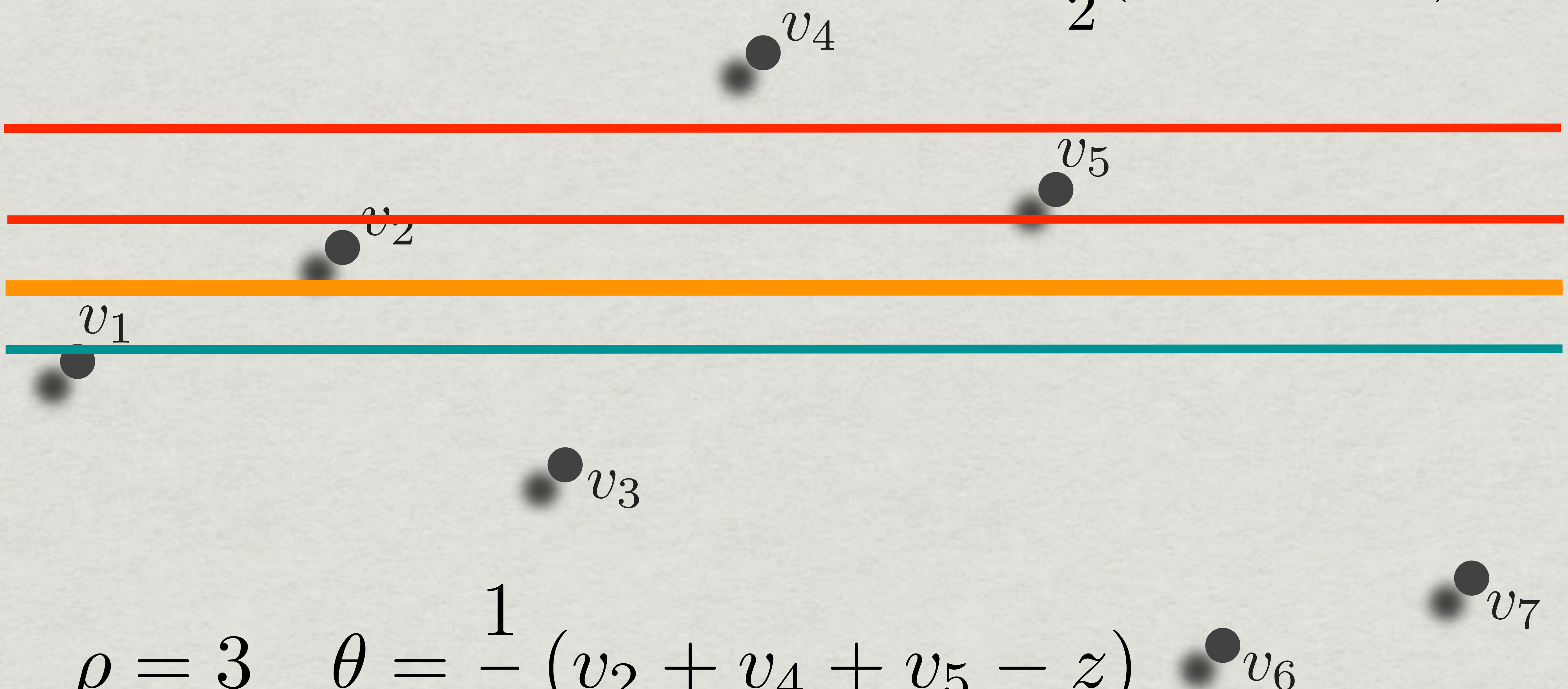
- Number of non-zero elements ρ

$$\rho \;=\; \max\left\{ j \;:\; \mu_j - \frac{1}{j}\left(\sum_{r=1}^{j} \mu_r - z\right) > 0 \right\}$$

Calculating Projection (G)

$$v_4 - (v_4 - z) > 0$$

$$v_5 - \frac{1}{2}(v_4 + v_5 - z) > 0$$



$$\rho = 3 \quad \theta = \frac{1}{3}\left(v_2 + v_4 + v_5 - z\right)$$

Efficient Projection Alg.

- Assume we know number of elements greater than v_j

$$\rho(v_j) = |\{v_i : v_i \geq v_j\}|$$

- Assume we know the sum of elements great than v_j

$$s(v_j) = \sum_{i:v_i \geq v_j} v_i$$

- Then, we can check in constant time the status of v_j

$$v_j > \theta \Leftrightarrow v_j > \frac{1}{\rho(v_j)}(s(v_j) - z) \Leftrightarrow s(v_j) - \rho(v_j)v_j < z$$

- Randomized median-like search [O(n) instead O(n log(n))]

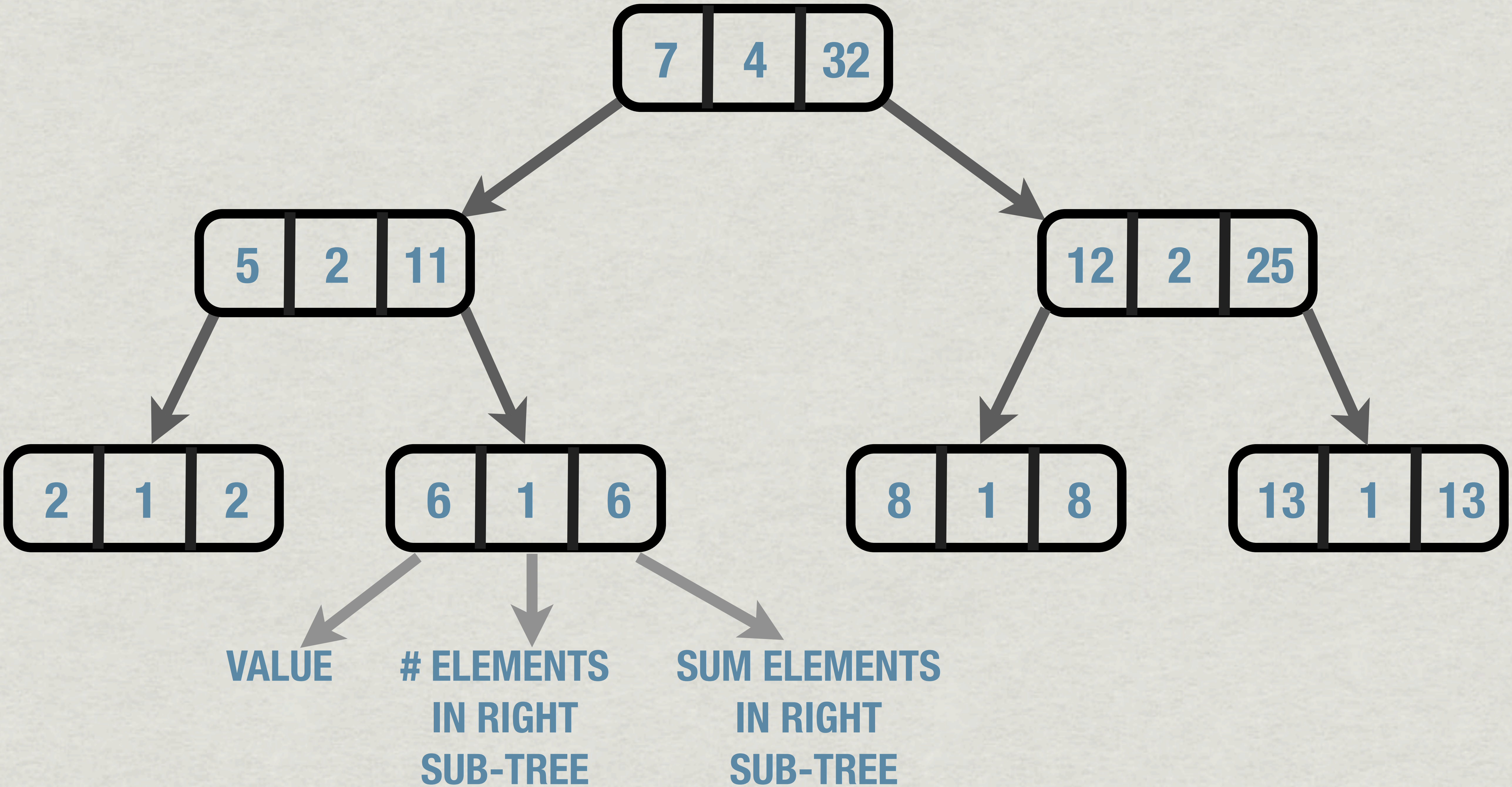
Working in High Dimensions

- In many applications the dimension is very high
  [text application: 2 million tokens]
  [web/ads data: often > 10^8]
- Small number of non-zero elements in each example
  [text application: ~ thousand tokens per document]
  [web/ads data: often < 10^{11}]
- Online/stochastic updates only modify the weights corresponding to non-zero features in example
- Goals:
  - linear time in the number of non-zero features
  - sub-linear in the full dimension

Efficient Alg. for High Dim

- Use **red-black** (**RB**) tree to store only the non-zero components of the weight vector. Non-zero components are stored w/o global shift $\Theta_t = \sum_{s \leq t} \theta_t$
- Each online/stochastic update deletes & then inserts non-zero elements of an example in O(k log(n)) time
- Store in each node of **RB** additional information that facilitates efficient search for “pivot” θ_t
- Upon projection, removal of a whole sub-tree is performed in logarithmic time using Tarajan’s (83) algorithm for splitting **RB** tree

RB Tree for Efficient Proj.



Pivot Search with RB Tree

PROCEDURE PIVOTSEARCH($\mathcal{T}, v, \rho, s$)

Co

IF

I $= \hat{s}$

I $)/\rho^\star$

C

EL

I $)/\rho^\star$

C $s)$

ENDIF

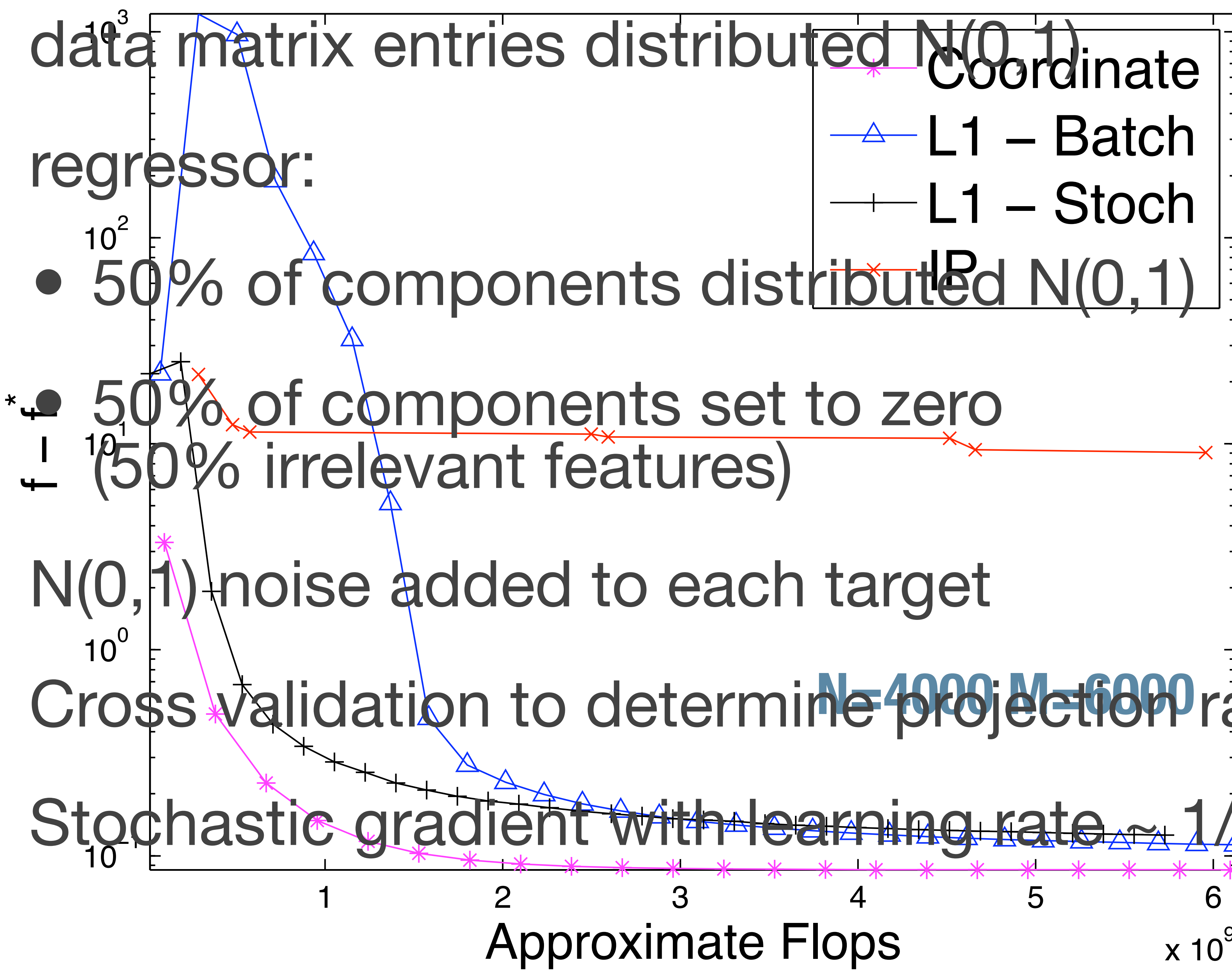
END

Empirical Results

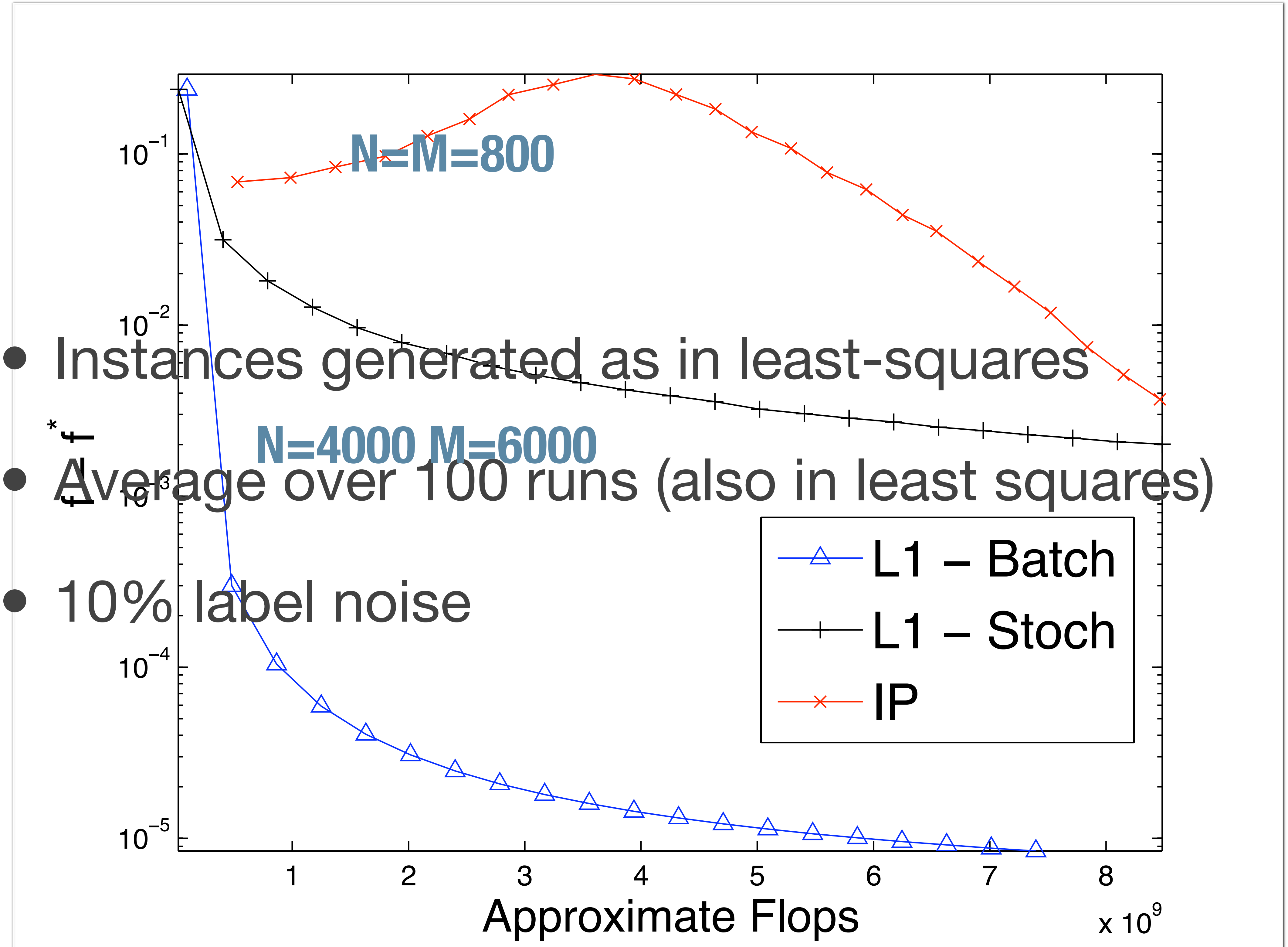
- Losses:
  - squared error
  - logistic regression (binary & multiclass)
- Datasets: synthetic, MNIST, Reuters Corpus Vol. 1
- Algorithms for comparison:
  - Specialized coordinate descent for SE (FHT'07)
  - Interior Point (IP) method with L_1 Boundary Const.
  - Mirror (entropic) descent & Exponentiated Gradient

Synthetic: Least Squares

- data matrix entries distributed N(0,1)
- regressor:
  - 50% of components distributed N(0,1)
  - 50% of components set to zero (50% irrelevant features)
- N(0,1) noise added to each target
- Cross validation to determine projection radius
- Stochastic gradient with learning rate ~ 1/sqrt(t)



Synt.: Logistic Regression



- Instances generated as in least-squares
- Average over 100 runs (also in least squares)
- 10% label noise

Results for MNIST Data

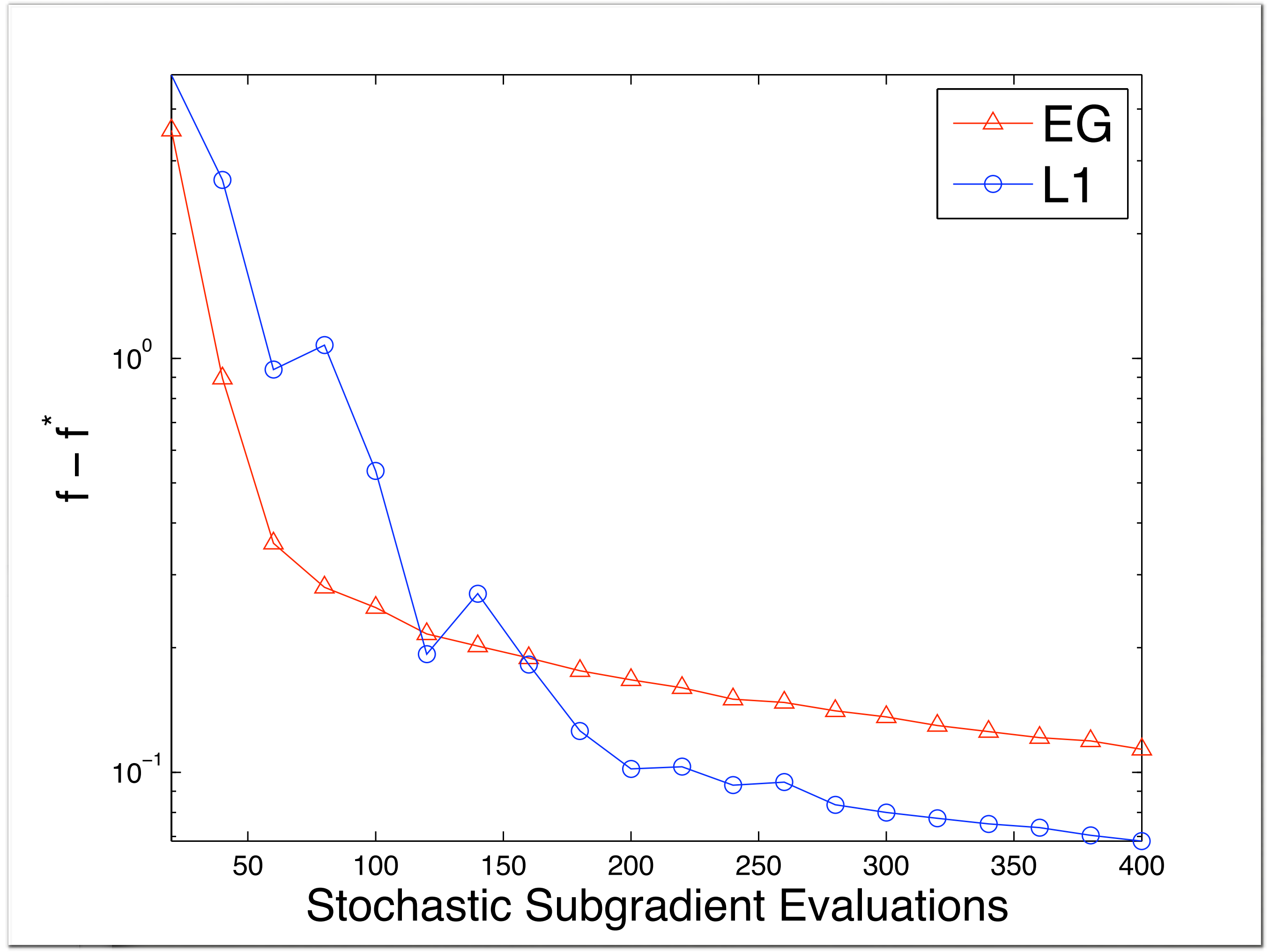
- Learned predictor of the form

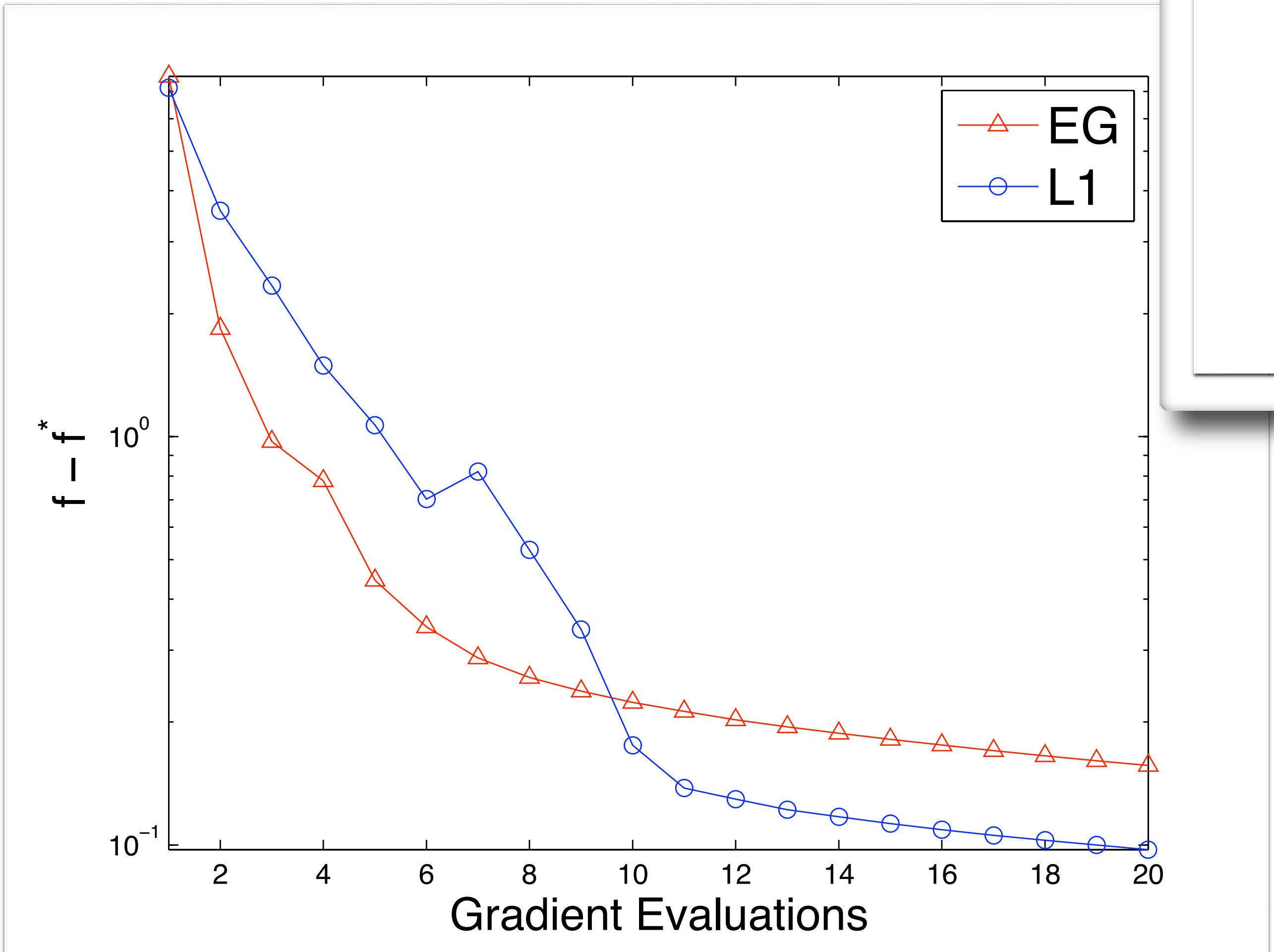
$$k(\mathbf{x}, j) = \sum_{i \in S} w_{ji} \sigma_{ji} K(\mathbf{x}_i, \mathbf{x}), \quad \sigma_{ji} = \begin{cases} 1 & \text{if } y_i = j \\ -1 & \text{otherwise.} \end{cases}$$

- S: support-set, found using multicass Perceptron
- 60,000 training examples, 28x28 pixel images
- Multiclass logistic regression with L_1

$$\begin{aligned} \min_{\mathbf{w}} \quad & \tfrac{1}{m} \textstyle\sum_{i=1}^{m} \log \left(1 + \textstyle\sum_{r \neq y_i} e^{k(\mathbf{x}_i, r) - k(\mathbf{x}_i, y_i)}\right) \\ \text{s.t.} \quad & \|\mathbf{w}_j\|_1 \leq z, \mathbf{w}_j \succeq 0. \end{aligned}$$

Results for MNIST Data





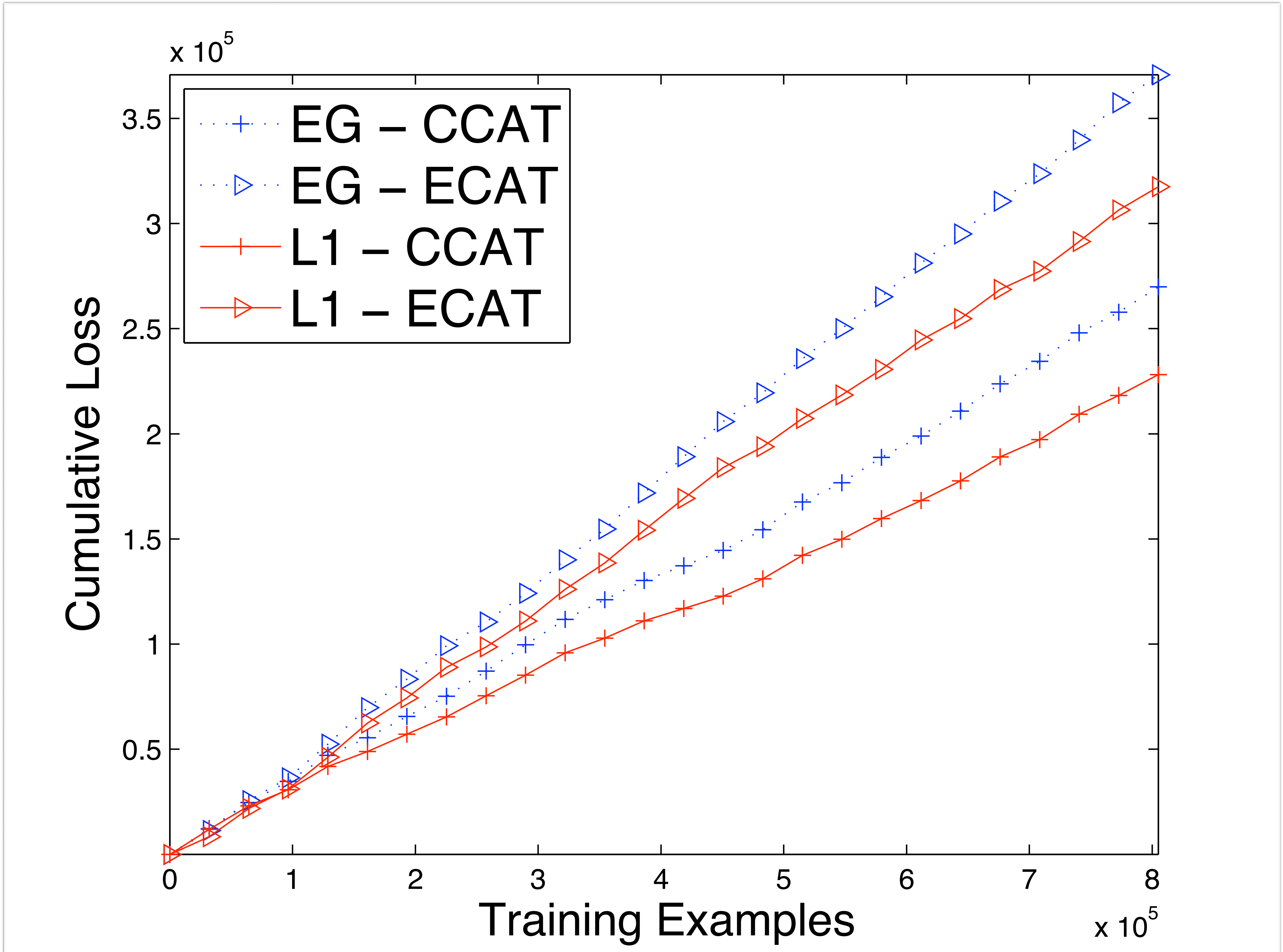
EG (Mirror Descent):

$$w_j^{(t+1)} = \frac{w_j^{(t)} e^{-\eta_t \nabla_j (\mathbf{w}^{(t)})}}{Z_t}$$

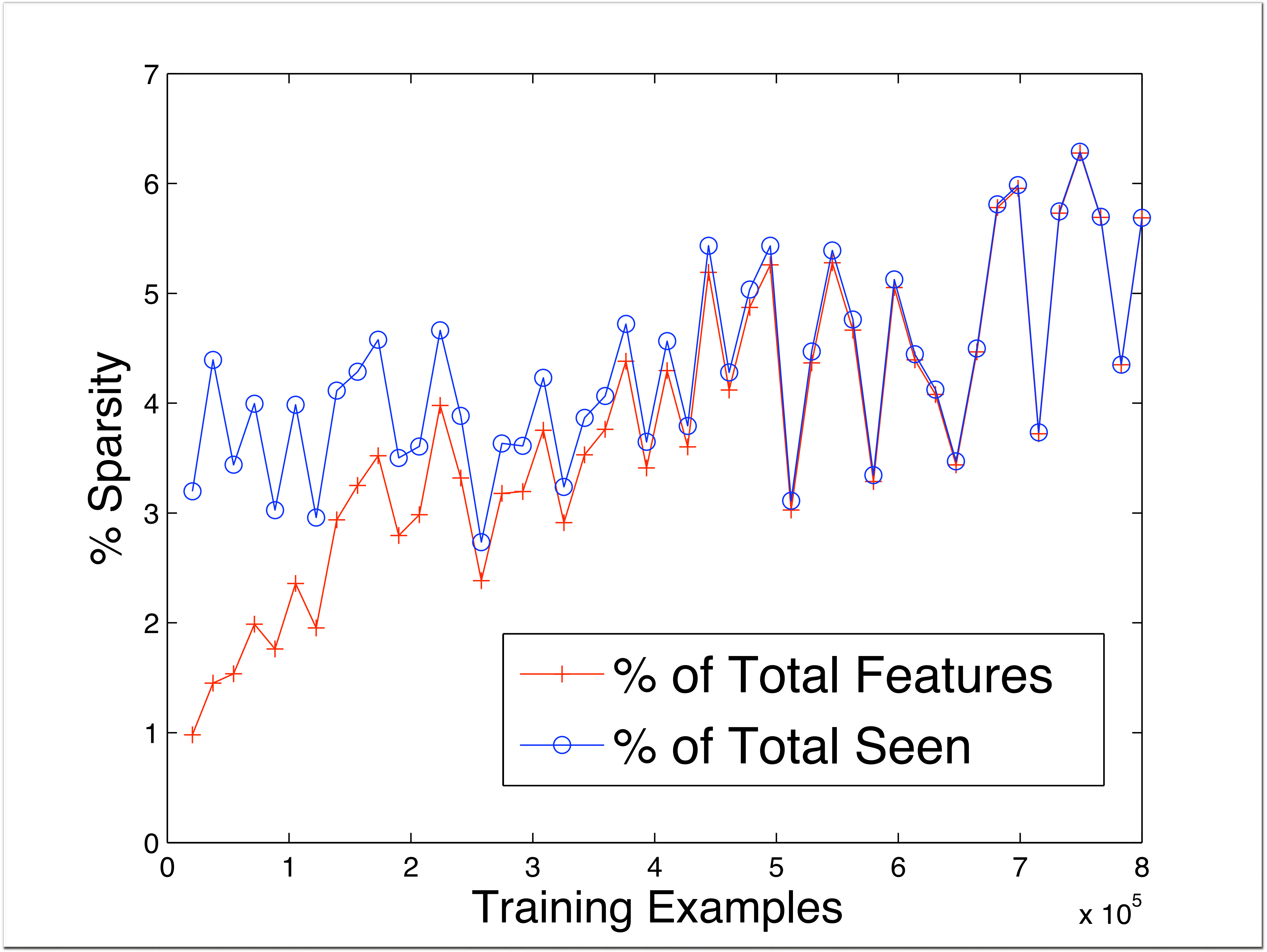
Reuters Corpus Vol. 1

- 804,414 articles, 1,946,684 word bigrams
- Each article includes ~0.26% of bigrams
- Compared with Exponentiated Gradient (KW'97) [extension with positive & negative weights]
- Both algorithms used the same domain constraints
- Learning rate ~ 1/sqrt(t)

L_1 Proj. vs. EG on RCV1



L_1 Magic with RCV1



Concluding Remarks

- Bertsekas first described Euclidean projection onto the simplex (see also [Gafni & Bertsekas, 84]) using sorting (O(n log(n)) time)
- Similar algorithms rediscovered and used as dual solvers for multiclass SVM, ranking problems (CS'01, CS'02, SS'06, Hazan'06)
- Efficient L_1-like experts tracking: Herbster & Warmuth'01
- First efficient L_1 algorithms for high dimensional settings
- Part of my work on design, analysis, and implementation of provably correct & efficient learning algorithms for very large scale problems
- Extensions and other related work:
  - Adding hyper-box constraints, non-Euclidean projections
  - *Infusing AdaBoost with L_1 regularization*
  - New algorithm for L_1 regularization through projections