MLSS 2011 Boosting Tutorial
Survey of Boosting
from an Optimization Perspective

Manfred K. Warmuth - UCSC
many parts prepared jointly with S.V.N. Vishwanathan - Purdue
greatly expanded from an ICML 2009 tutorial

Updated May 8, 2013

Outline

Setup for Boosting

[Giants of field: Schapire,Freund]

- Fixed set of ± 1 labeled examples
- Easy to find "weak learners" that have error $\leq \frac{1}{2} - \epsilon$
- Can these weak learners be "boosted" to abitrary high accuracy? - question posed by Kearns in his thesis [K89]
- First recursive Boosting algorithm in Schapire's thesis [S90]
- AdaBoost: Boosting with 5 lines of code [FS97]
- Here: Fancier Boosting algorithms - big overview
- Black box setup:
  - Assume an oracle provides weak learners for weighted examples
  - The boosting algorithm "aggregates" the weak learners

Protocol of Boosting [FS97]

- Maintain distribution on N ± 1 labeled examples
- At iteration $t = 1, \ldots, T$:
  - Receive "weak" hypothesis h^t from oracle
  - Update $\mathbf{d}^{t-1}$ to $\mathbf{d}^t$: put more weights on "hard" examples
- Output convex combination of the T weak hypotheses

Goals:

- Final hypothesis highly accurate
- Small number of iterations
- No overfitting

Classifying spam

Weak/base hypotheses

- "viagra" in text
- Mailed from certain sites
- Any fancy spam classifier that can steal from friend or foe
  - base hypothesis don't have to be "weak"
  - combine anything and get an improvement
  - use large variety of base learners
    - simple decision trees
    - SVMs
    - nearest neighbor

Other example problems

- Classify pictures as to whether they contain a dog?
- Does picture contain a human face? [Viola]

Most common base learners

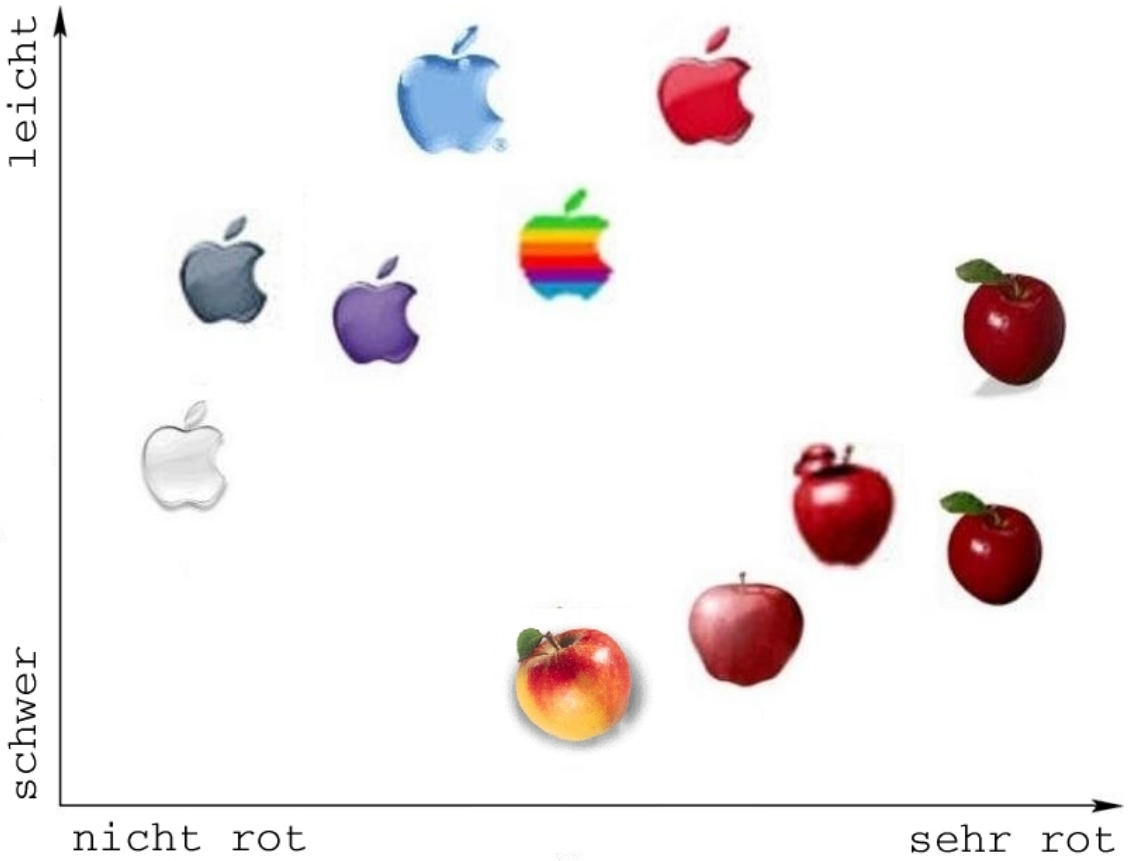
- "Good" feature from a large set of binary features that are known to work well for your problem
- Decision stumps on a large set of real features

$$h_\theta^i = (\text{feature}_i \geq \theta)$$

Boosting is simple versatile technique for building strong classifiers

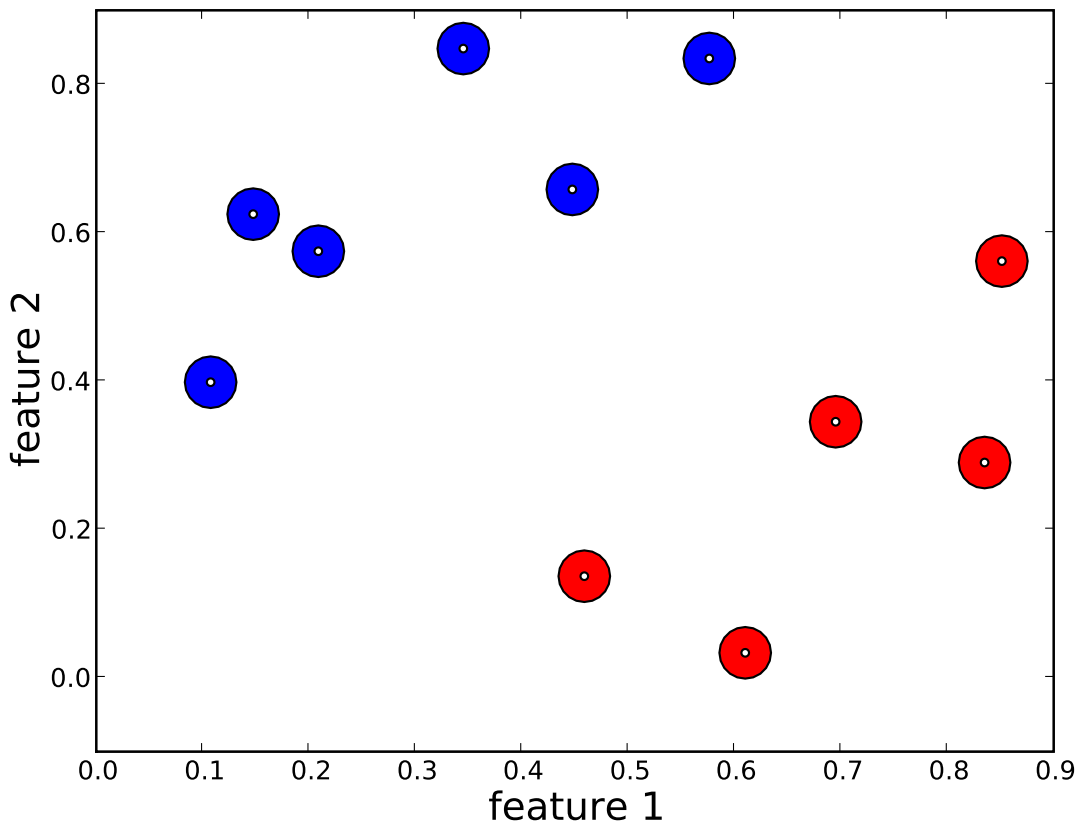
- provided you have oracle for providing good base learner
- coordinate descent
- choice is greedy - i.e. oracle iteratively provides a good feature

Our running example: Classifying apples



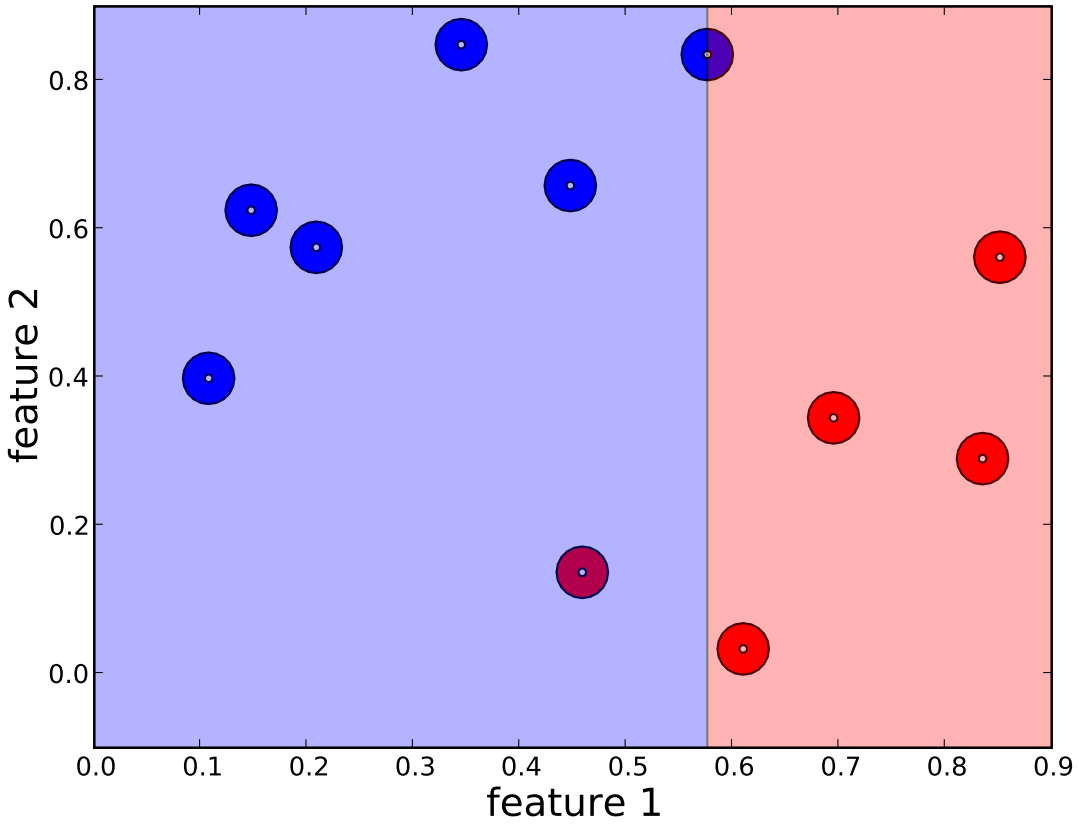
- examples: 11 apples
- $+1$ if artificial
  -1 if natural
- goal:
  accurate classification
- 2 real features:
  redness & lightness

Setup for Boosting



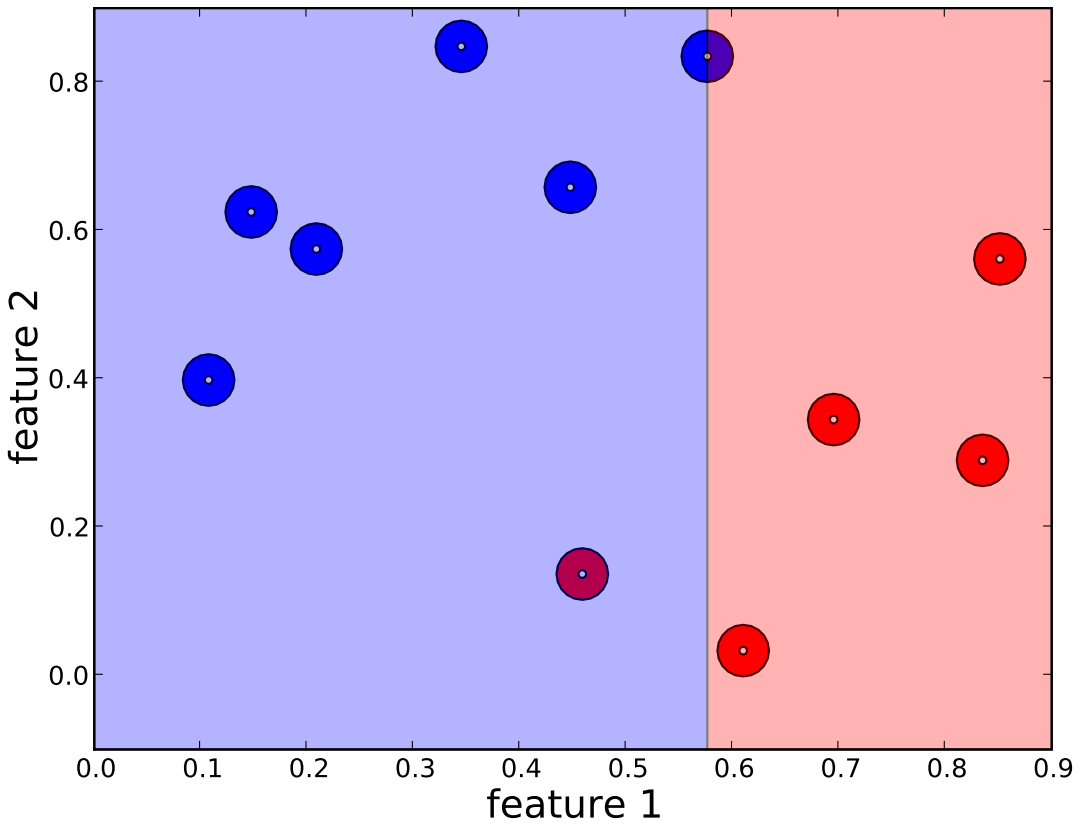
- +1/-1 examples
- weight $d_n \approx$ size
- separable

Weak hypotheses



- weak hypotheses:
  decision stumps on two features
  one can't do it
- goal:
  find convex combination of weak hypotheses that classifies all

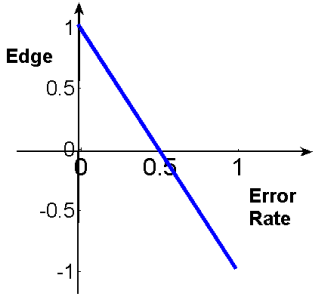
Boosting: 1st iteration



First hypothesis:

- error: $\frac{1}{11}$
- edge: $\frac{9}{11}$

  low error $=$ high edge



edge $= 1 - 2$ error

Accuracy on example / edge

	$y_n h(x_n) := u_n$
perfect	$+1$
wrong	-1
neutral	0

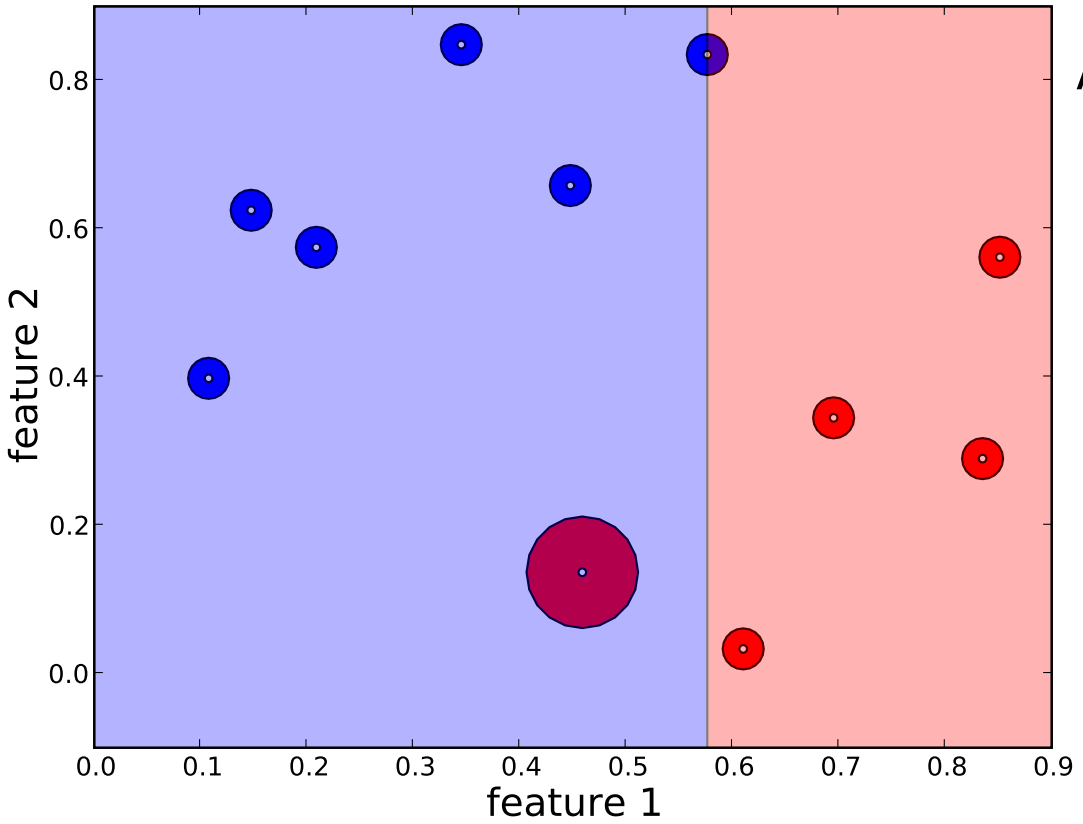
u_n is accuracy of h_n on example (x_n, y_n)
edge is accuracy on all examples weighted by distribution

$$\sum_n d_n u_n = \mathbf{d} \cdot \mathbf{u}$$

Weak hypothesis - column vector of accuracies

examples x_n	labels y_n	1st stump $h^1(x_n)$	accuracies u_n^1
	-1	-1	**1**
	-1	-1	**1**
	-1	-1	**1**
	-1	-1	**1**
	1	1	**1**
	1	1	**1**
	1	1	**1**
	1	-1	$-\mathbf{1}$

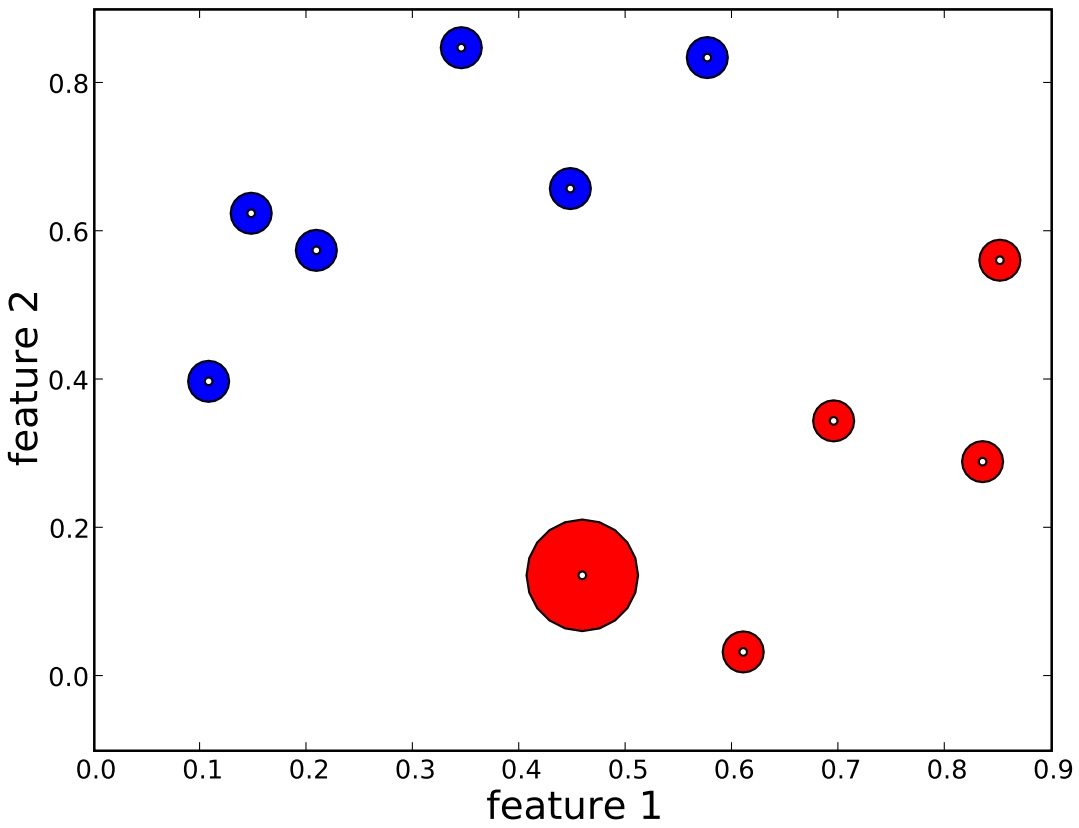
Update after 1st



Misclassified examples
- increased weights

After update
- edge of hypothesis decreased

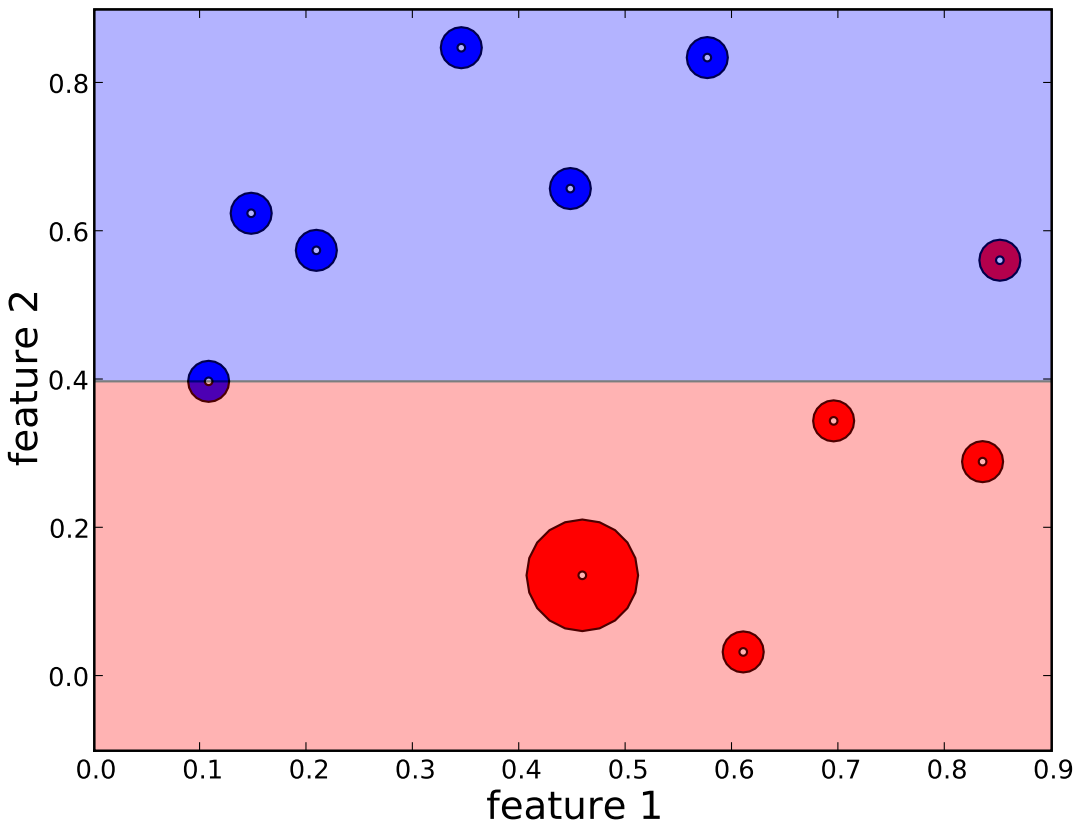
Before 2nd iteration



Hard examples

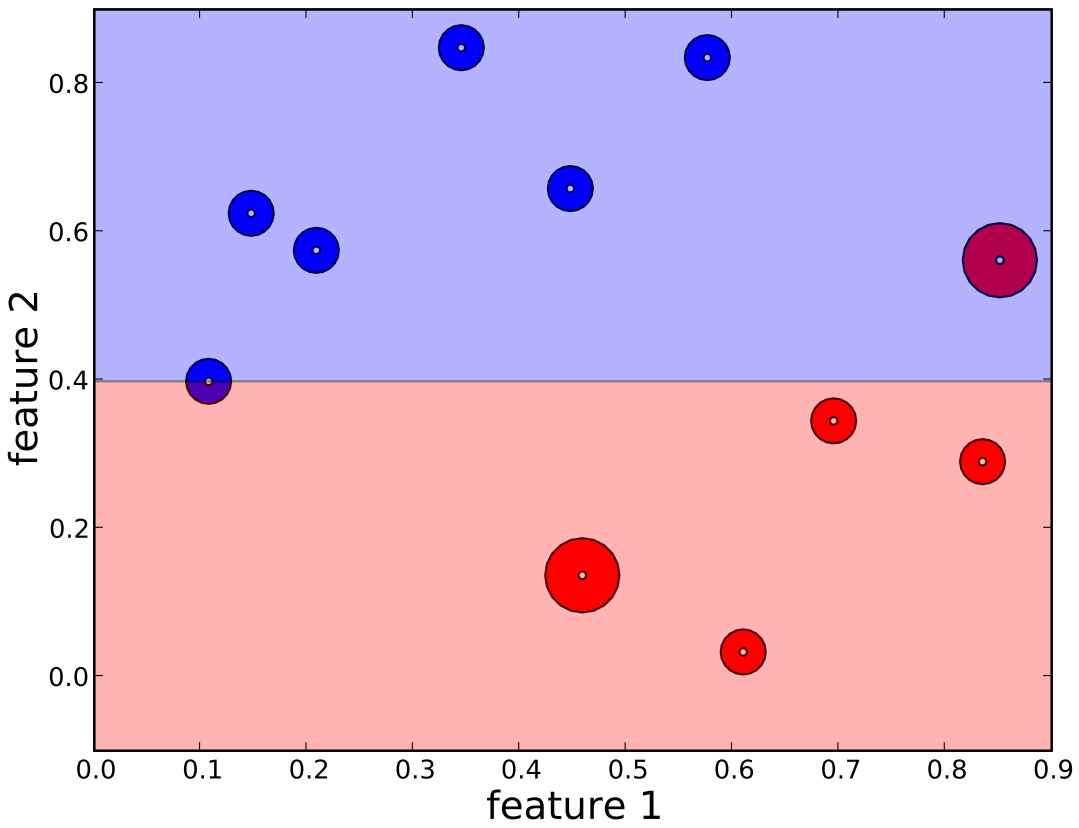
- high weight

Boosting: 2nd hypothesis



Pick hypotheses
with high (weighted) edge

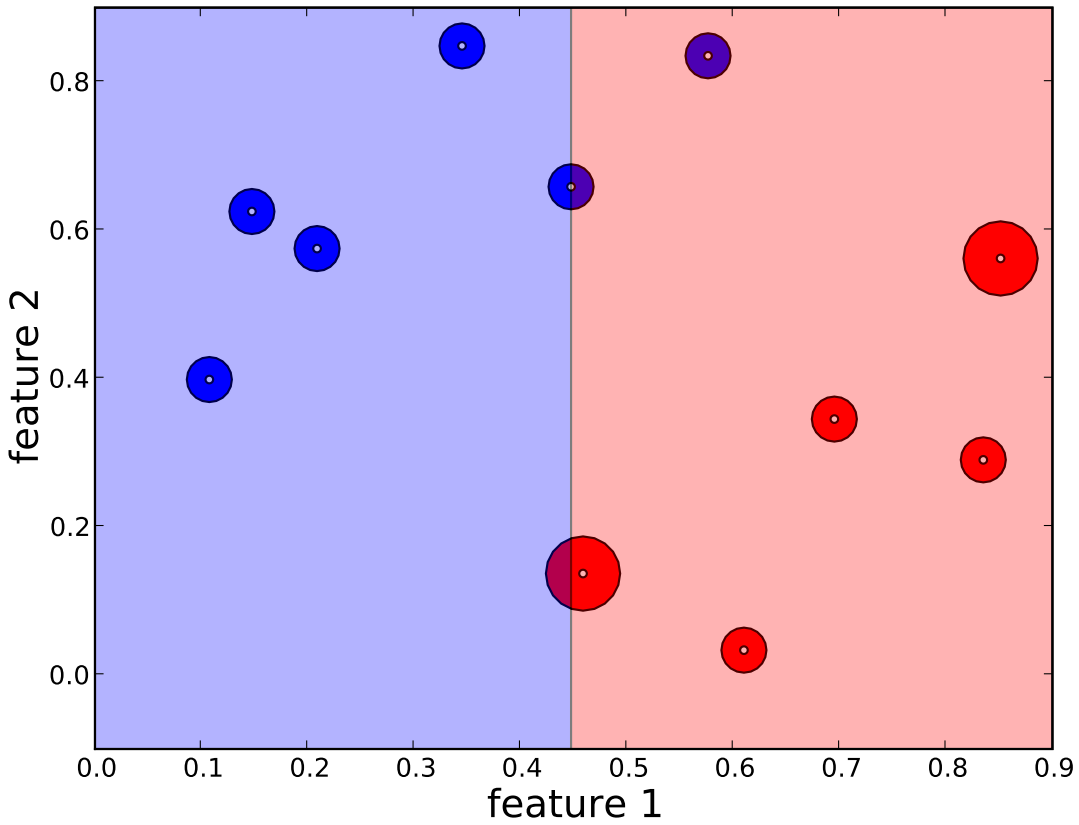
Update after 2nd



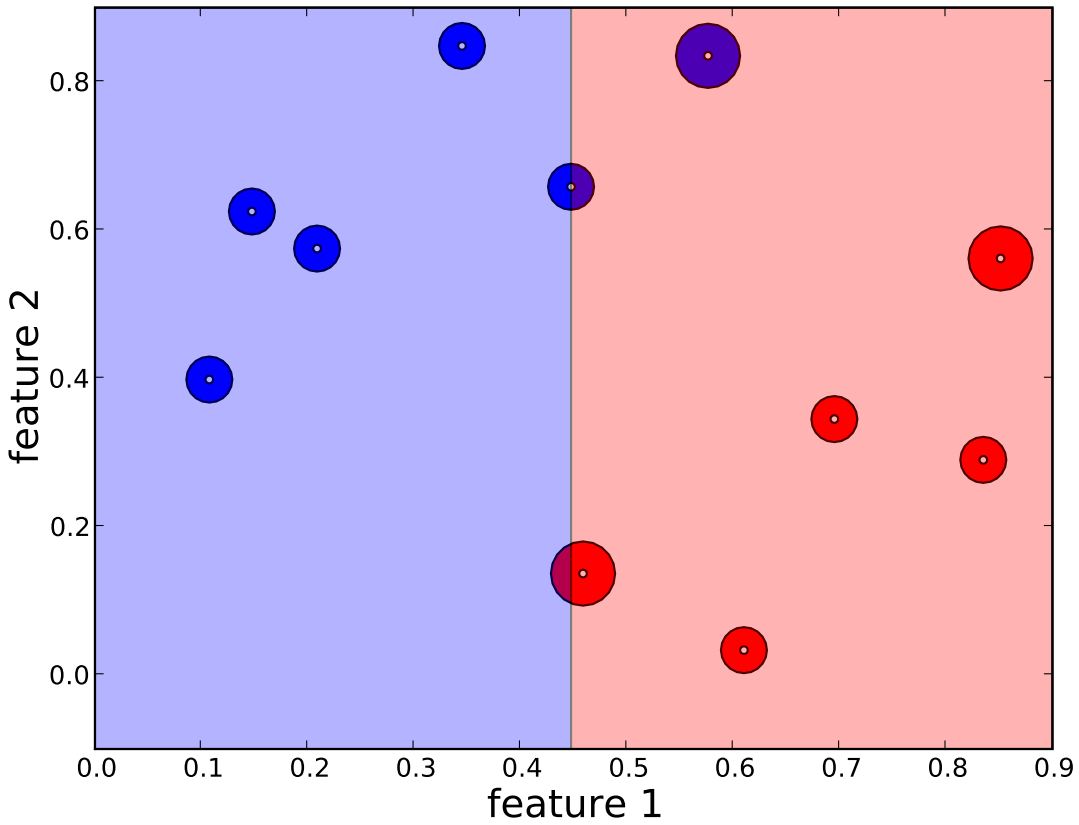
After update

- edges of all past hypotheses should be small

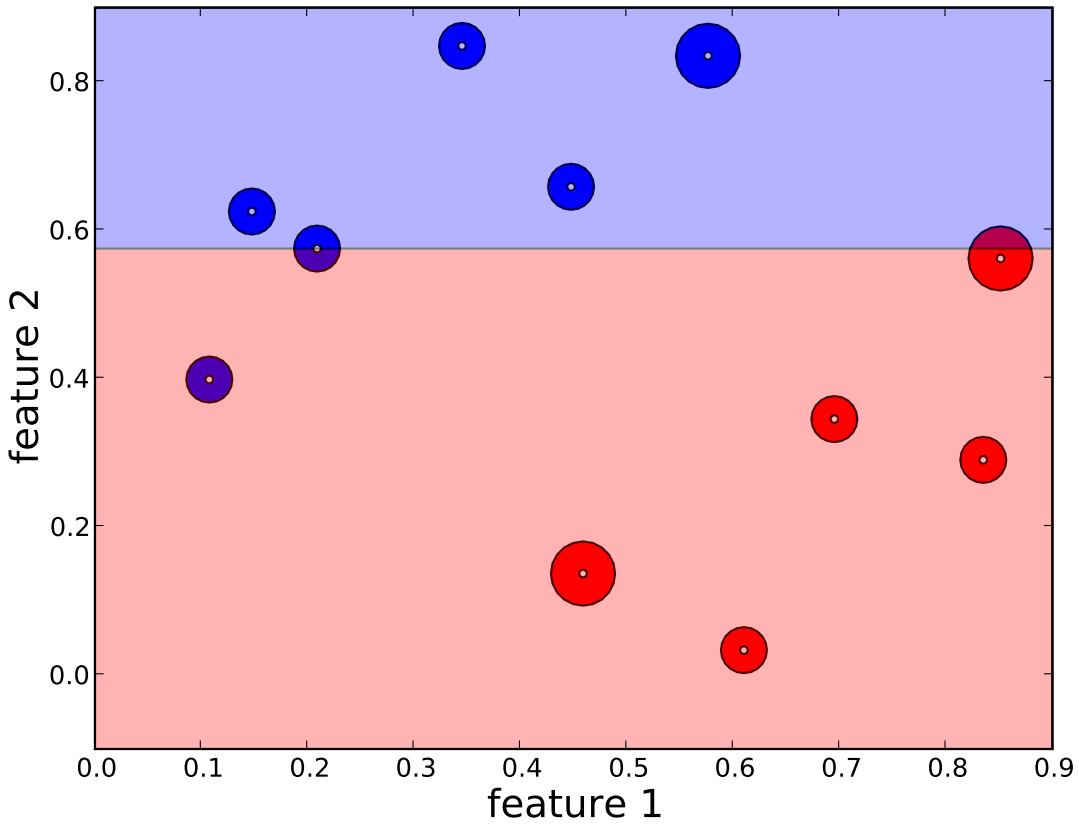
3rd hypothesis



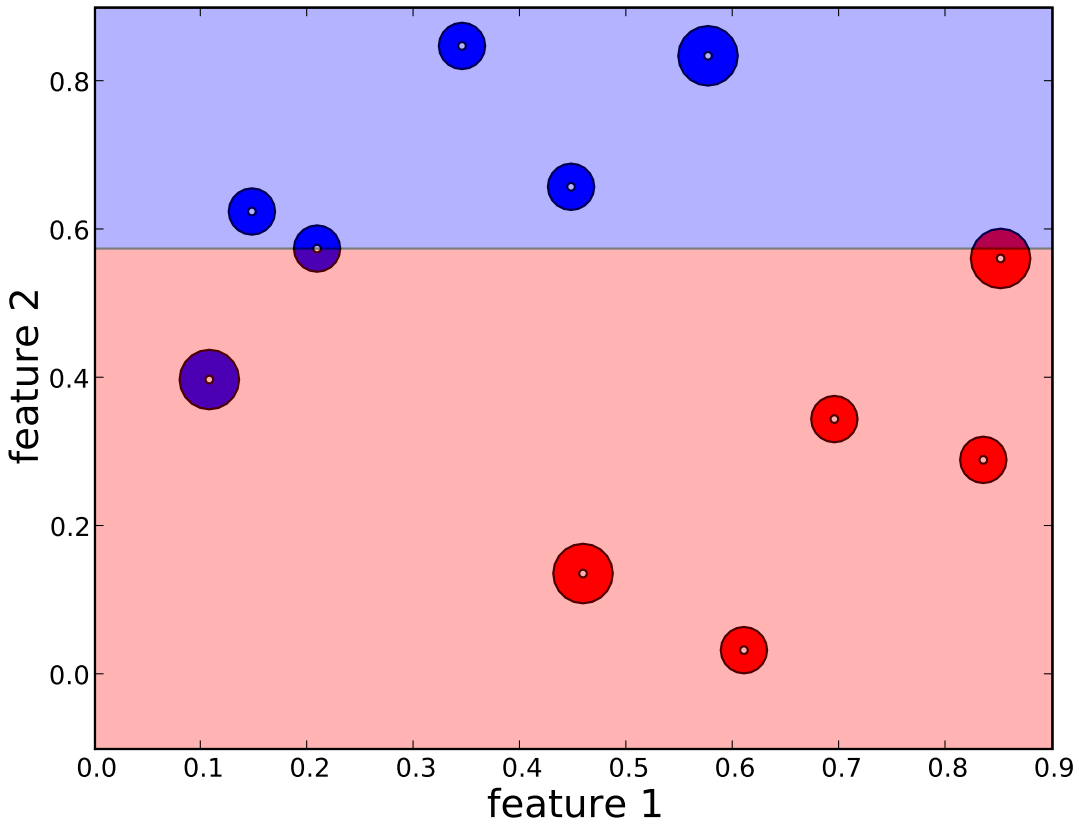
Update after 3rd



4th hypothesis

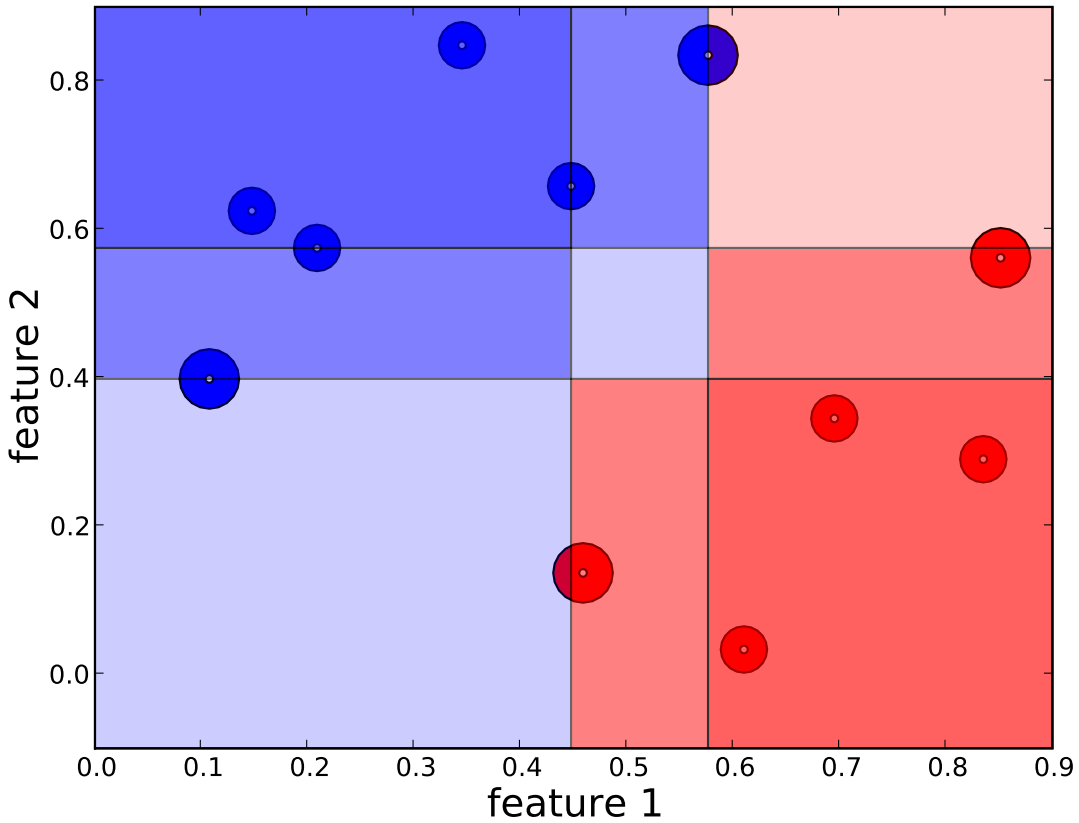


Update after 4th



Final convex combination of all hypotheses

Decision: $\sum_{t=1}^{T} w^t h^t(\mathbf{x}) \geq 0$?



Positive total weight - Negative total weight

Protocol of Boosting [FS97]

- Maintain distribution on N ± 1 labeled examples
- At iteration $t = 1, \ldots, T$:
  - Receive "weak" hypothesis h^t from oracle of high edge
  - Update $\mathbf{d}^{t-1}$ to $\mathbf{d}^t$: put more weights on "hard" examples
- Output convex combination of the weak hypotheses $\sum_{t=1}^{T} w^t \, h^t(x)$

Two sets of weights:
- distribution on $\mathbf{d}$ on examples
- distribution on $\mathbf{w}$ on hypotheses

Recall data representation

Accuracy of example

	$y_n h^t(x_n) := u_n^t$
perfect	$+1$
opposite	-1
neutral	0

examples x_n	labels y_n	$h^t(x_n)$	$\mathbf{u}^t$
	-1	-1	$\mathbf{1}$
	-1	-1	$\mathbf{1}$
	-1	-1	$\mathbf{1}$
	-1	1	$-\mathbf{1}$
	1	1	$\mathbf{1}$
	1	1	$\mathbf{1}$
	1	1	$\mathbf{1}$
	1	-1	$-\mathbf{1}$

AdaBoost update

$$d_n^t \sim d_n^{t-1} \exp(-w^t u_n^t)$$

where

- d_n^{t-1} are old weights
- $u_n^t \in \pm 1$ are accuracies
- w^t is "learning rate" / coefficient of hypothesis h^t

$$w^t = \frac{1}{2} \ln \frac{1 + \mathbf{d}^{t-1} \cdot \mathbf{u}^t}{1 - \mathbf{d}^{t-1} \cdot \mathbf{u}^t}$$

Oracle gives weak hypothesis such that $\underbrace{\mathbf{d}^{t-1} \cdot \mathbf{u}^t}_{\text{edge}} \geq \gamma \geq 0$

AdaBoost

+

- Trivial to code, provided you have oracle
- Fast update
- Has iteration bound
  consistent hypothesis in $\leq \frac{\ln n}{\gamma^2}$ iterations
- Good initial Boosting algorithm

-

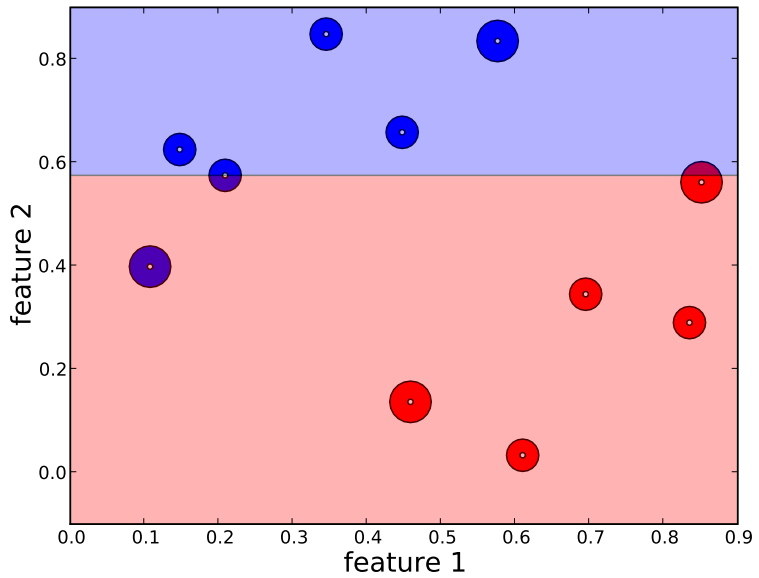
- Too many iterations
- Same hypothesis chosen multiple times
- Cycles on inseparable case

Questions

- How to motivate Boosting updates?
- What is underlying optimization problem?
- What is the regularization?
- How do we get bound on number of iterations?

Two applications of Boosting

- Build strong classifier from weak classifier
- Use Boosting as filter
  - Determine the examples that remain hard to classify after combining a set of base classifier
  - Those examples are likely to be "noisy"
  - Present expensing human agent only with hard examples



Outline

Motivations of the updates?

- Motivate additive and multiplicative updates
- Use linear regression as example problem
- Motivate all updates as minimizing

$$\text{regularization} \; + \; \eta \, \text{loss/objective}$$

  - Sometimes additional constraints
  - Sometimes no regularization

Online linear regression

For $t = 1, 2, \ldots$

- Get instance $\mathbf{x}_t \in \mathbb{R}_n$
- Predict $\hat{y}_t = \mathbf{w}_t \cdot \mathbf{x}_t$
- Get label $y_t \in \mathbb{R}$
- Incur square loss $(y_t - \hat{y}_t)^2$
- Update $\mathbf{w}_t \rightarrow \mathbf{w}_{t+1}$

Two main update families - linear regression

- Additive

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \underbrace{(\mathbf{w}_t \cdot \mathbf{x}_t - y_t)\mathbf{x}_t}_{\text{Gradient of Square Loss}}$$

  - Motivated by squared Euclidean distance
  - Weights can go negative
  - Gradient Descent (GD)
- Multiplicative

$$w_{t+1,i} = \frac{w_{t,i}\ e^{-\eta(\mathbf{w}_t \cdot \mathbf{x}_t - y_t)x_{t,i}}}{Z_t}$$

  - Motivated by relative entropy
  - Updated weight vector stays on probability simplex
  - Exponentiated Gradient (EG) [KW97]

Additive Updates

Goal

Find $\mathbf{w}_{t+1}$ close to old $\mathbf{w}_t$ that has small loss on last example OR
Minimize tradeoff between closeness and loss

$$\begin{aligned} \mathbf{w}_{t+1} &= \operatorname*{argmin}_{\mathbf{w}} U(\mathbf{w}) \\ U(\mathbf{w}) &= \underbrace{||\mathbf{w} - \mathbf{w}_t||_2^2}_{\text{divergence}} + \eta \underbrace{(\mathbf{w} \cdot \mathbf{x}_t - y_t)^2}_{\text{loss}} \end{aligned}$$

$\eta > 0$ is the *learning rate*

Additive updates

$$\frac{\partial U(\mathbf{w})}{\partial w_i}|_{w_i=w_{t+1,i}} = 2(w_{t+1,i} - w_{t,i}) + 2\eta(\mathbf{w}\cdot\mathbf{x}_t - y_t)x_{t,i} = 0$$

Therefore,

$$\begin{aligned} \text{implicit:} \quad \mathbf{w}_{t+1} &= \mathbf{w}_t - \eta(\mathbf{w}_{t+1}\cdot\mathbf{x}_t - y_t)\mathbf{x}_t \\ \text{explicit:} \quad &= \mathbf{w}_t - \eta(\mathbf{w}_t\cdot\mathbf{x}_t - y_t)\mathbf{x}_t \end{aligned}$$

Multiplicative updates

$$\begin{aligned} \mathbf{w}_{t+1} &= \operatorname*{argmin}_{\sum_i w_i = 1} U(\mathbf{w}) \\ \text{where } U(\mathbf{w}) &= \underbrace{\sum w_i \ln \frac{w_i}{w_{t,i}}}_{\text{relative entropy}} + \eta(\mathbf{w} \cdot \mathbf{x}_t - y_t)^2 \end{aligned}$$

Define Lagrangian

$$L(\mathbf{w}) = \sum w_i \ln \frac{w_i}{w_{t,i}} + \eta(\mathbf{w} \cdot \mathbf{x}_t - y_t)^2 + \lambda(\sum_i w_i - 1)$$

where λ Lagrange coeff.

Multiplicative updates

$$\begin{aligned}
\frac{\partial L(\mathbf{w})}{\partial w_i} &= \ln \frac{w_i}{w_{t,i}} + 1 + \eta(\mathbf{w} \cdot \mathbf{x}_t - y_t)x_{t,i} + \lambda = 0 \\
\ln \frac{w_{t+1,i}}{w_{t,i}} &= -\eta(\mathbf{w}_{t+1} \cdot \mathbf{x}_t - y_t)x_{t,i} - \lambda - 1 \\
w_{t+1,i} &= w_{t,i} e^{-\eta(\mathbf{w}_{t+1} \cdot \mathbf{x}_t - y_t)x_{t,i}} e^{-\lambda - 1}
\end{aligned}$$

Enforce normalization constraint by setting $e^{-\lambda-1}$ to $1/Z_t$

$$\begin{aligned}
\text{implicit:} \qquad w_{t+1,i} &= \frac{w_{t,i} e^{-\eta \mathbf{w}_{t+1} \cdot \mathbf{x}_t - y_t)x_{t,i}}}{Z_t} \\
\text{explicit:} \qquad &= \frac{w_{t,i} e^{-\eta \mathbf{w}_t \cdot \mathbf{x}_t - y_t)x_{t,i}}}{Z'_t}
\end{aligned}$$

GD + EG via differential equations

- GD

$$\dot{\mathbf{w}}_t = -\nabla L(\mathbf{w}_t)$$

Here $\dot{f}(x_t) = \partial f(x_t)/\partial t$

Two ways to discretize:

Forward Euler

$$\begin{aligned} \frac{\mathbf{w}_{t+h} - \mathbf{w}_t}{h} &= -\eta \nabla L(\mathbf{w}_t) \\ \mathbf{w}_{t+h} &= \mathbf{w}_t - \eta h \nabla L(\mathbf{w}_t) \\ \text{explicit: } \mathbf{w}_{t+1} &\overset{h=1}{=} \mathbf{w}_t - \eta \nabla L(\mathbf{w}_t) \end{aligned}$$

Backward Euler

$$\begin{aligned} \frac{\mathbf{w}_{t+h-h} - \mathbf{w}_{t+h}}{-h} &= -\eta \nabla L(\mathbf{w}_{t+h}) \\ \mathbf{w}_{t+h} &= \mathbf{w}_t - \eta h \nabla L(\mathbf{w}_{t+h}) \\ \text{implicit: } \mathbf{w}_{t+1} &\overset{h=1}{=} \mathbf{w}_t - \eta \nabla L(\mathbf{w}_{t+1}) \end{aligned}$$

GD + EG via differential equation

- EG

$$\dot{\log \mathbf{w}_t} = -\eta \nabla L(\mathbf{w}_t)$$

Two ways to discretize:

Forward Euler

$$\begin{aligned} \frac{\log w_{t+h,i} - \log w_{t,i}}{h} &= -\eta \nabla L(\mathbf{w}_t)_i \\ w_{t+h,i} &= w_{t,i} e^{-\eta h \nabla L(\mathbf{w}_t)_i} \\ \text{explicit: } w_{t+1,i} &\overset{h=1}{=} w_{t,i} e^{-\eta \nabla L(\mathbf{w}_t)_i} \end{aligned}$$

Backward Euler

$$\begin{aligned} \frac{\log w_{t+h-h,i} - \log w_{t+h,i}}{-h} &= -\eta \nabla L(\mathbf{w}_{t+h})_i \\ w_{t+h,i} &= w_{t,i} e^{-\eta h \nabla L(\mathbf{w}_{t+h})_i} \\ \text{implicit: } w_{t+1,i} &\overset{h=1}{=} w_{t,i} e^{-\eta \nabla L(\mathbf{w}_{t+1})_i} \end{aligned}$$

Derivation of normalized update more involved

Return to minimizing tradeoff

- Batch: tradeoff between regularization and total loss
- Regularization determines how parameter space is searched

Two main regularization

- squared Euclidean distance
- relative entropy

Two families of updates

$\|.\|^2$	relative entropy
Widrow Hoff update	EG update
Perceptron	Winnow
Backprop	Weighted Majority algorithm
SVMs	Baysian update
Newton's method	Boosting

Different properties

ignores small weights	smoothed 1-norm
log of a Gaussian	information theoretic
rotation invariant	not rotation invariant

Send symbol X on channel

X	$P(X = x_i)$	$-\log P(x_i)$
x_1	$\frac{1}{2}$	1
x_2	$\frac{1}{4}$	2
x_3	$\frac{1}{8}$	3
x_4	$\frac{1}{8}$	3

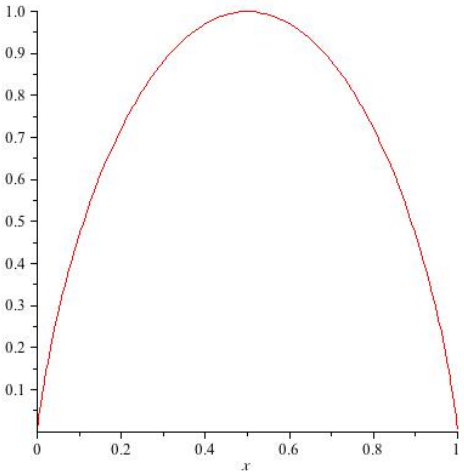
$-\log P(x_i)$ is measure of surprise in bits

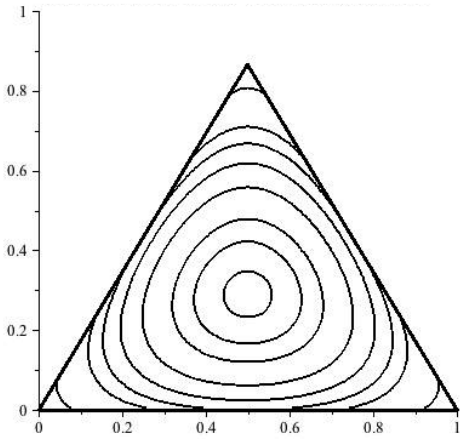
- $-\log 1 = 0$ no surprise
- $-\log 0 = \infty$ infinite surprise
- $-\log \frac{1}{2^i} = i$ i bits of surprise

Entropy equals expected surprise

$$\begin{aligned} H(X) &= \sum_i P(x_i) \log \frac{1}{p(x_i)} \\ &= \frac{1}{2}\,1 + \frac{1}{4}\,2 + \frac{1}{8}\,3 + \frac{1}{8}\,3 \\ &= 1\frac{3}{4} \text{ bits} \end{aligned}$$

$-p \log p - (1-p)\log(1-p)$ 3-dim entropy unif. in dim. n





$\log(n)$

Code

Assigns symbols a bitstring (codeword)
- any sequence of codewords must be uniquely decodable

Expected codelength

$$L(\underbrace{C}_{\text{code}}) = \sum_i p(x_i) \underbrace{\ell_C(\mathbf{x}_i)}_{\text{length of codeword for } x_i}$$

Optimal code C^*
- $L(C^*)$ is minimum
  - Thm: $H(X) \leq L(C^*) \leq H(X) + 1$
  - Thm: Huffman codes are optimal
  - More info: first five chapters of Cover & Thomas

Relative entropy between distributions $\mathbf{p}$ and $\mathbf{q}$

$$\Delta(\mathbf{p}, \mathbf{q}) = \sum_i p_i \log \frac{p_i}{q_i}$$

$$= \underbrace{\sum_i p_i \log \frac{1}{q_i}}_{\text{expected codelength of best codebook for } \mathbf{q}} - \overbrace{\underbrace{\sum_i p_i \log \frac{1}{p_i}}_{\text{expected codelength of best codebook for } \mathbf{p}}}^{H(\mathbf{p})}$$

where all expectations are wrt $\mathbf{p}$

Relative entropy to the uniform distribution

$$\begin{aligned}\Delta(\mathbf{p}, (\frac{1}{n})) &= \sum_i p_i \log \frac{p_i}{1/n} \\ &= \sum_i p_i \log p_i + p_i \log n \\ &= \log n - H(\mathbf{p}) \\ &\geq 0\end{aligned}$$

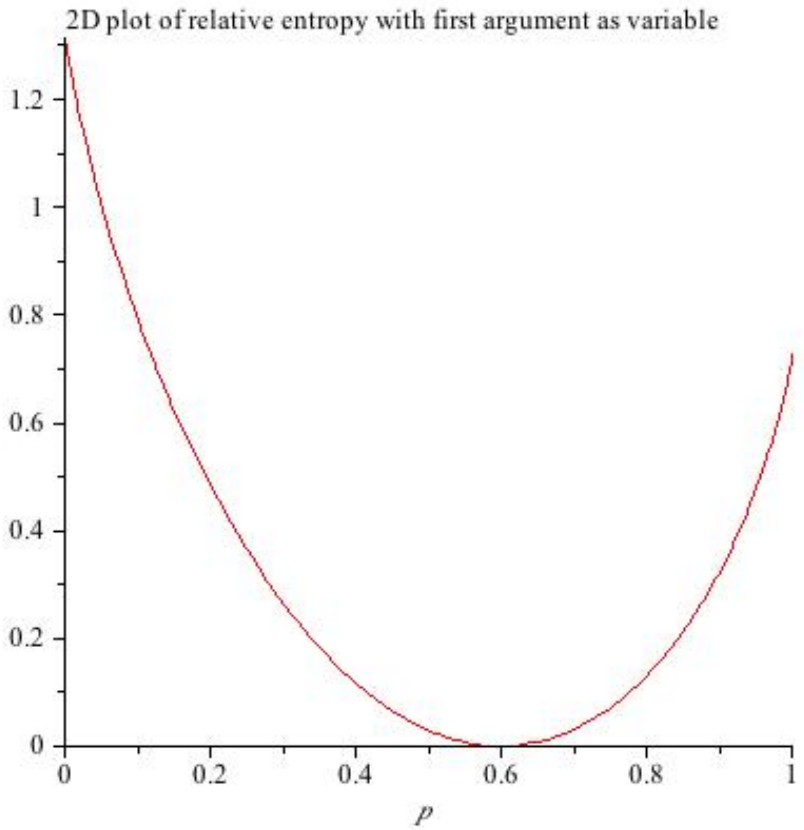
0 at center of simplex

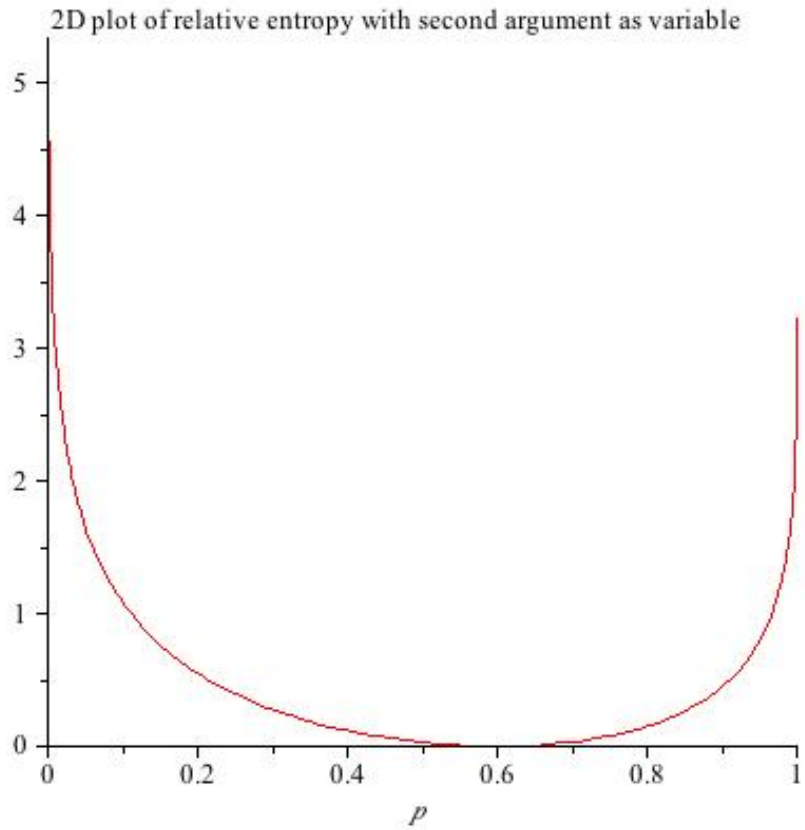
In general

$$\Delta(\mathbf{p}, \mathbf{q}) \geq 0,$$

where equality holds iff $\mathbf{p} = \mathbf{q}$

Which argument should be the variable?

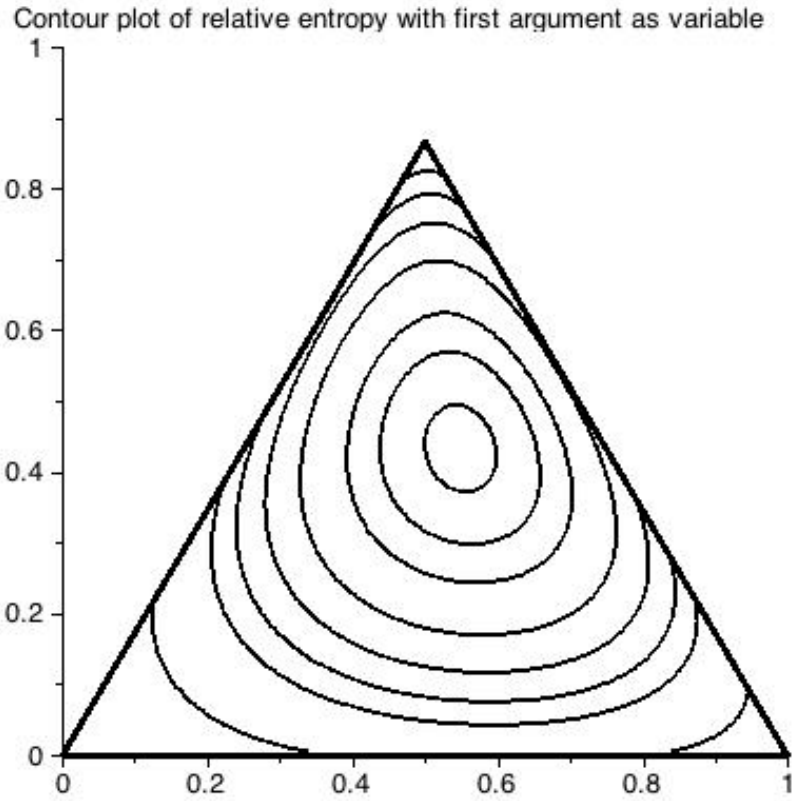


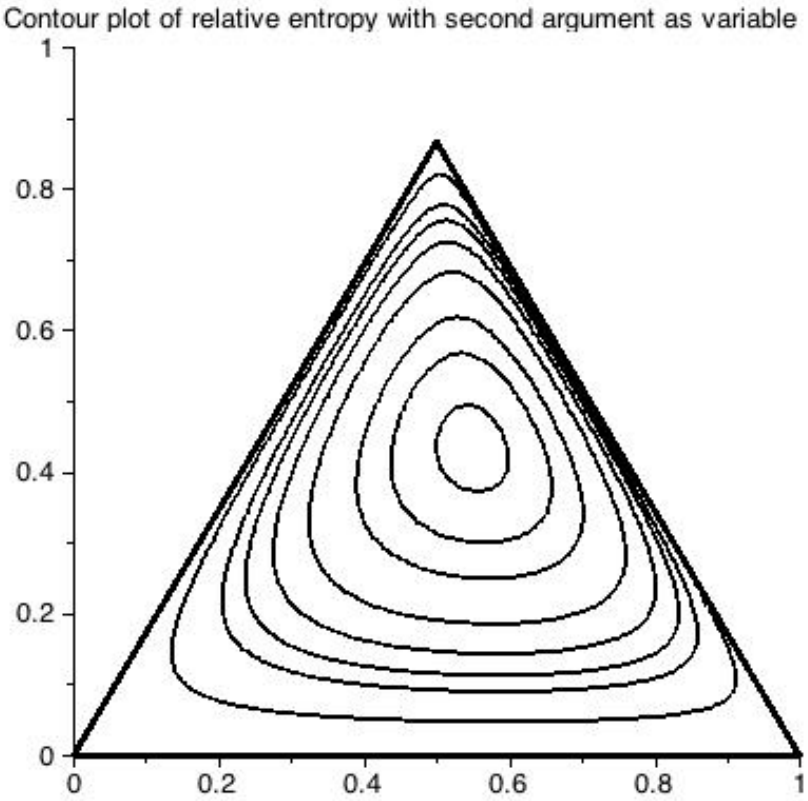


not too steep at boundary
motivates EG and Boosting

steep at boundary

Two relative entropies in 3D



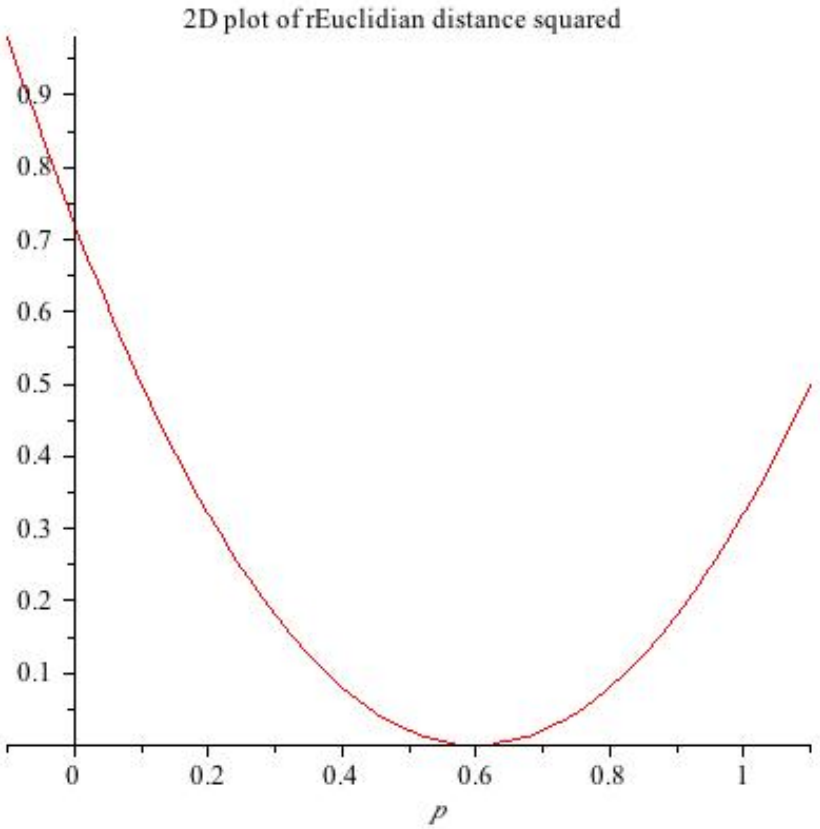


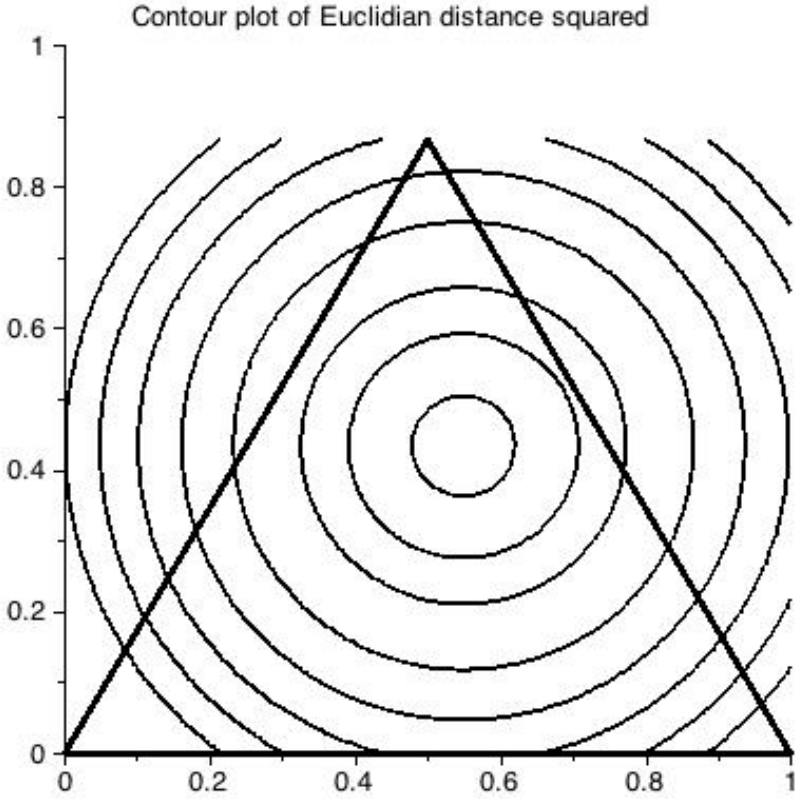
Both are barriers for simplex

Use of relative entropy (w. first argument as var.)

- As regularizer in motivation of update
- As measure of progress in analysis

Squared Euclidean distance "ignores" the simplex





Questions

- What optimization problems motivate Boosting?
- Why is relative entropy used as a regularization?
- What is the loss/objective for Boosting?
- What if no regularization is used?

Outline

What is the objective of Boosting?

- Give objective ito edges and margins
- Minimize objective without regulariaztion
  - Game theoretic interpretation of Boosting
  - Column generation interpretation
  - Minimizing objective alone: LPBoost

Recall data representation

Accuracy of
hypothesis h^t on example $(\mathbf{x}_n, y_n)$

	$y_n h^t(x_n) \quad := u_n^t$
perfect	$+1$
opposite	-1
neutral	0

examples x_n	labels y_n	$h^t(x_n)$	$\mathbf{u}^t$
	-1	-1	$\mathbf{1}$
	-1	-1	$\mathbf{1}$
	-1	-1	$\mathbf{1}$
	-1	1	$-\mathbf{1}$
	1	1	$\mathbf{1}$
	1	1	$\mathbf{1}$
	1	1	$\mathbf{1}$
	1	-1	$-\mathbf{1}$

Edge vs. margin [Br99]

Edge of a hypothesis h^t for a distribution $\mathbf{d}$ on the examples

$$\underbrace{\sum_{n=1}^{N} \overbrace{u_n^t}^{\text{accuracy on example}} d_n}_{\text{weighted accuracy of hypothesis}} \qquad \mathbf{d} \in \mathcal{P}^N$$

Margin of example n for current hypothesis weighting $\mathbf{w}$

$$\underbrace{\sum_{t=1}^{T} \overbrace{u_n^t}^{\text{accuracy of example}} w_t}_{\text{weighted accuracy of example}} \qquad \mathbf{w} \in \mathcal{P}^T$$

Edge vs. margin [Br99]

Edge of a hypothesis h^t for a distribution $\mathbf{d}$ on the examples

$$\underbrace{\sum_{n=1}^{N} \overbrace{u_n^t}^{\text{accuracy on example}} d_n}_{\text{weighted accuracy of hypothesis}} \qquad \mathbf{d} \in \mathcal{P}^N$$

Margin of example n for current hypothesis weighting $\mathbf{w}$

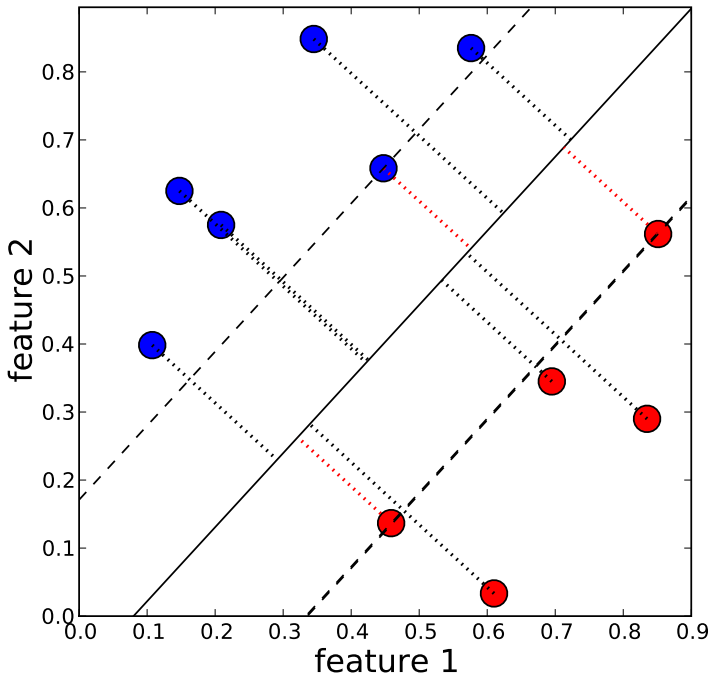
$$\underbrace{\sum_{t=1}^{T} \overbrace{u_n^t}^{\text{accuracy of example}} w_t}_{\text{weighted accuracy of example}} \qquad \mathbf{w} \in \mathcal{P}^T$$

Objective in the $\mathbf{d}$ domain

- Edges of past hypotheses should be small after update
  - More weight on hard (low accuracy examples) decreases weighted accuracy $=$ edge
- Minimize maximum edge of past hypotheses

Objective in the **w** domain

- Choose convex combination of weak hypotheses that maximizes the minimum margin of the examples



Which margin in **w** domain?

SVM	2-norm
Boosting	1-norm

Connection between objectives?

$$\text{min max edge} \quad = \quad \text{max min margin}$$

$$\min_{\mathbf{d}\in\mathcal{S}^N} \max_{q=1,2,\ldots,t-1} \underbrace{\mathbf{u}^q \cdot \mathbf{d}}_{\text{edge of hypothesis q}} \quad = \quad \max_{\mathbf{w}\in\mathcal{S}^{t-1}} \min_{n=1,2,\ldots,N} \underbrace{\sum_{q=1}^{t-1} u_n^q \, w^q}_{\text{margin of example } n}$$

Van Neumann's Minimax Theorem

Boosting as zero-sum-game [FS97]

Rock, Paper, Scissors game

			column player		
			R	P	S
			w_1	w_2	w_3
row player	R	d_1	0	1	-1
	P	d_2	-1	0	1
	S	d_3	1	-1	0

gain matrix

Row player minimizes
Column player maximizes

$$\text{payoff} = \mathbf{d}^T \mathbf{U}\, \mathbf{w} = \sum_{i,j} d_i U_{i,j} \mathbf{w}_j$$

Single row is pure strategy of row player and **d** is mixed strategy

Single column is pure strategy of column player and **w** is mixed strategy

Optimum strategy

			R	P	S
			w_1	w_2	w_3
			.33	.33	.33
R	d_1	.33	0	1	-1
P	d_2	.33	-1	0	1
S	d_3	.33	1	-1	0

- Minimax theorem:

$$\min_d \max_{\mathbf{w}} \mathbf{d}^T \mathbf{U}\mathbf{w} = \min_d \max_j \mathbf{d}^T \mathbf{U}\mathbf{e}_j$$
$$= \max_{\mathbf{w}} \min_d \mathbf{d}^T \mathbf{U}\mathbf{w} = \max_{\mathbf{w}} \min_i \mathbf{e}_i^\top \mathbf{U}\mathbf{w}$$
$$= \text{value of the game (0 in example)}$$

$\mathbf{e}_j$ is pure strategy

Connection to Boosting?

Payoff matrix $\mathbf{U}$

- Rows are the examples
- Columns $\mathbf{u}^q$ encode weak hypothesis h^q
- Row sum: margin of example
- Column sum: edge of weak hypothesis
- Value of game:

$$\min \max \text{ edge} = \max \min \text{ margin}$$

Van Neumann's Minimax Theorem

Edges/margins

			R	P	S		
			w_1	w_2	w_3	margin	
			.33	.33	.33		
R	d_1	.33	0	1	-1	0	
P	d_2	.33	-1	0	1	0	min
S	d_3	.33	1	-1	0	0	
	edge		0	0	0		
				max			

value of game 0

New column added: Boosting

Helps maximizing column player

			R	P	S			
			w_1	w_2	w_3	w_4	margin	
			.44	0	.22	.33		
R	d_1	.22	0	1	-1	1	.11	
P	d_2	.33	-1	0	1	1	.11	min
S	d_3	.44	1	-1	0	-1	.11	
	edge		.11	-.22	.11	.11		
				max				

Value of game **increases** from 0 to .11

Row added: on-line learning

Helps minimizing row player

			R	P	S		
			w_1	w_2	w_3	margin	
			.33	.44	.22		
R	d_1	0	0	1	-1	.22	
P	d_2	.22	-1	0	1	-.11	min
S	d_3	.44	1	-1	0	-.11	
	d_4	.33	-1	1	-1	-.11	
	edge		-.11	-.11	-.11		
				max			

Value of game **decreases** from 0 to -.11

Boosting: maximize margin incrementally

	w_1^1
d_1^1	0
d_2^1	1
d_3^1	-1

iteration 1

	w_1^2	w_2^2
d_1^2	0	-1
d_2^2	1	0
d_3^2	-1	1

iteration 2

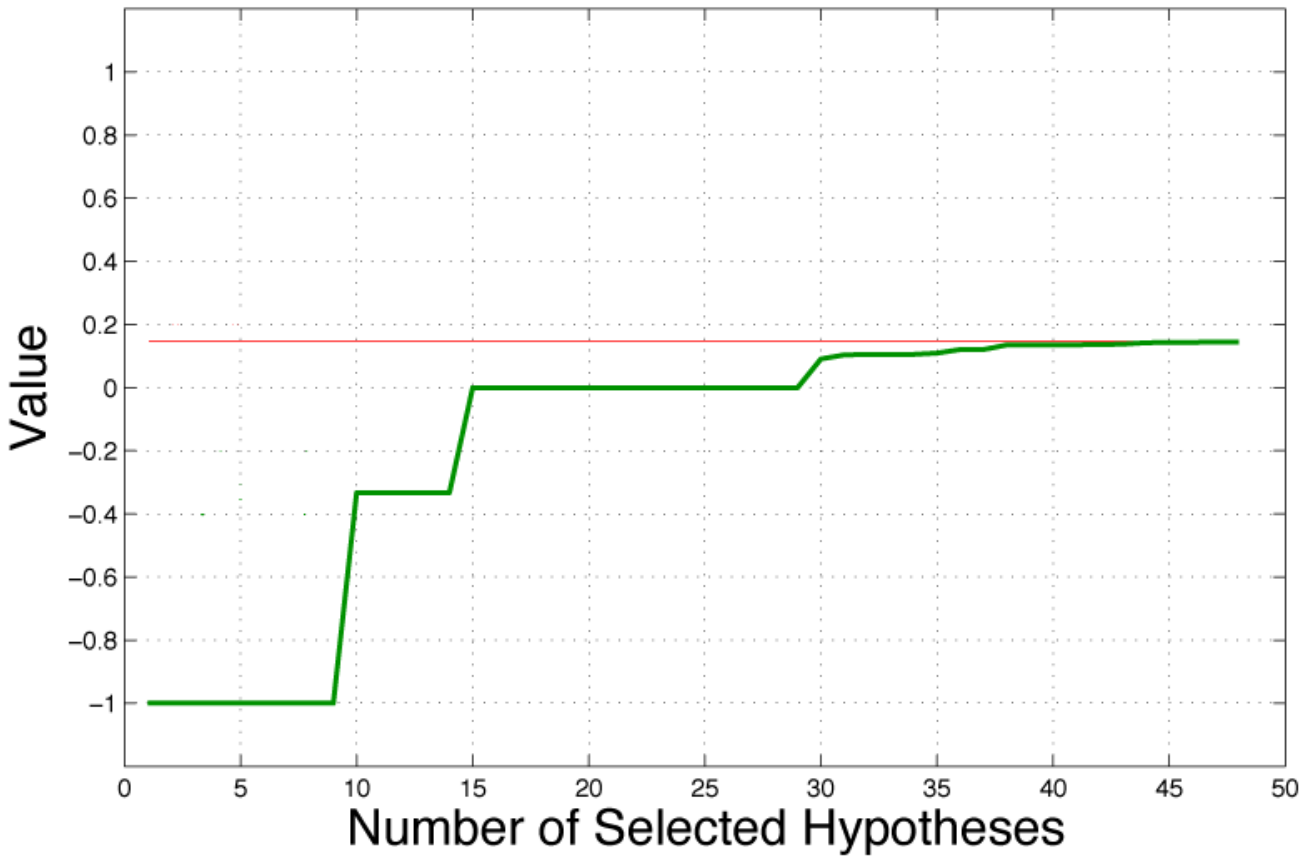
	w_1^3	w_2^3	w_3^3
d_1^3	0	-1	1
d_2^3	1	0	-1
d_3^3	-1	1	0

iteration 3

- In each iteration solve optimization problem to update d
- Column player = oracle provides new hypothesis
- Boosting is column generation method in **d** domain and coordinate ascent in **w** domain

Boosting = greedy method for increasing margin

Converges to optimum margin w.r.t. all hypotheses



Want small number of iterations

Power of the oracle

Strong oracle:

- Return hypothesis of maximum edge

Goal:

- For given ϵ, produce convex combination of weak hypotheses with soft margin $\geq$ value $- \epsilon$

Weak oracle:

- For some guarantee g, return hypothesis of edge $\geq g$

Goal:

- For given ϵ, produce convex combination of weak hypotheses with soft margin $\geq g - \epsilon$

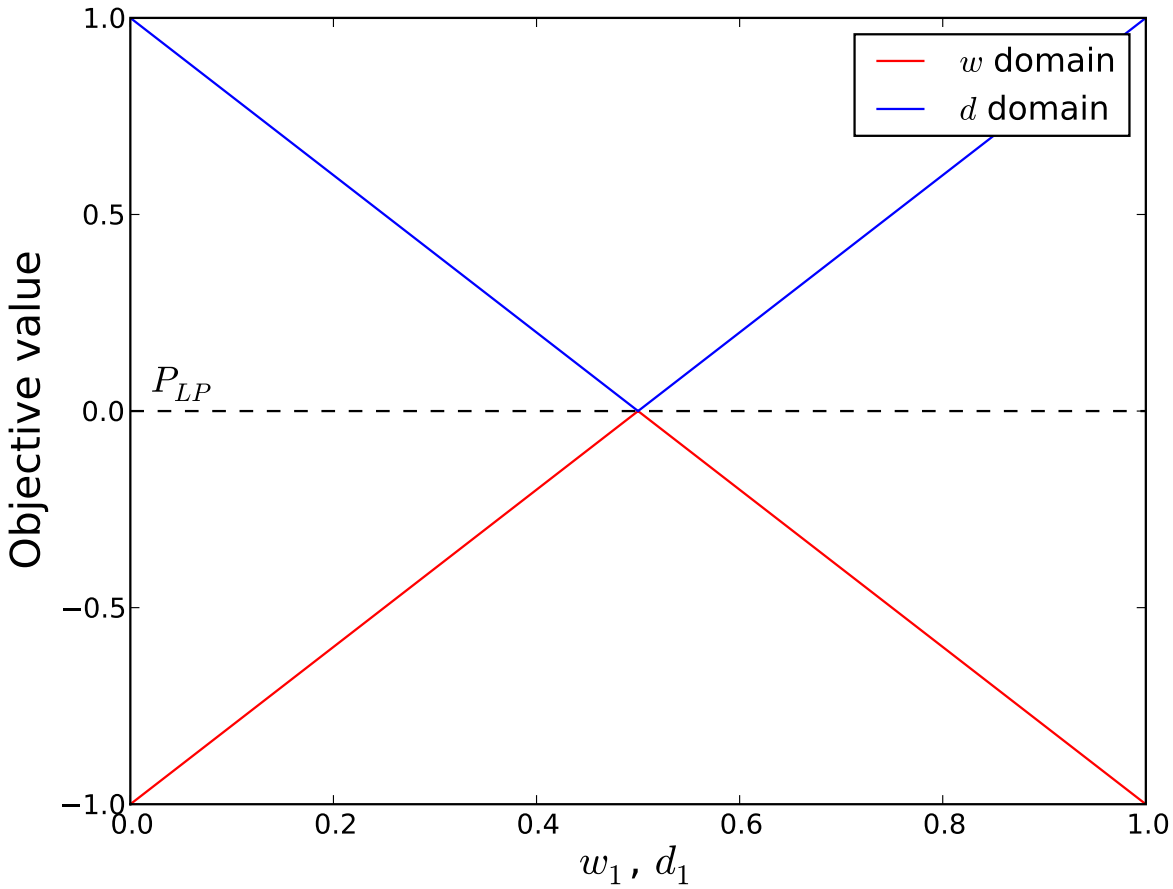
Recall minimax thm

$$\min_{\mathbf{d}\in\mathcal{S}^N} \max_{q=1,2,\ldots,t} \underbrace{\mathbf{u}^q \cdot \mathbf{d}}_{\text{edge of hypothesis q}}$$

$$= \max_{\mathbf{w}\in\mathcal{S}^t} \min_{n=1,2,\ldots,N} \underbrace{\left(\sum_{q=1}^{t} u_n^q \, w^q\right)}_{\text{margin of example } n}$$

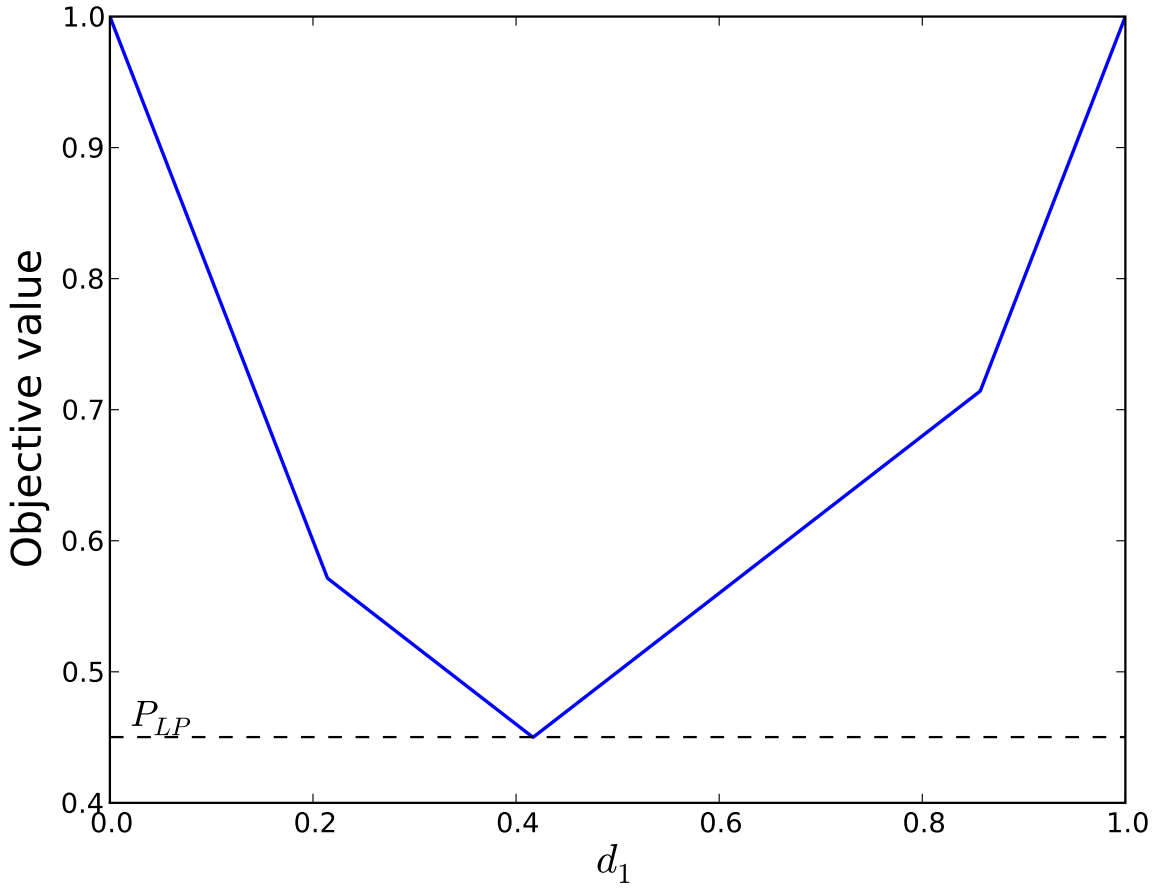
Linear Programming Duality

Optimization problems in both the **d** and the **w** domain

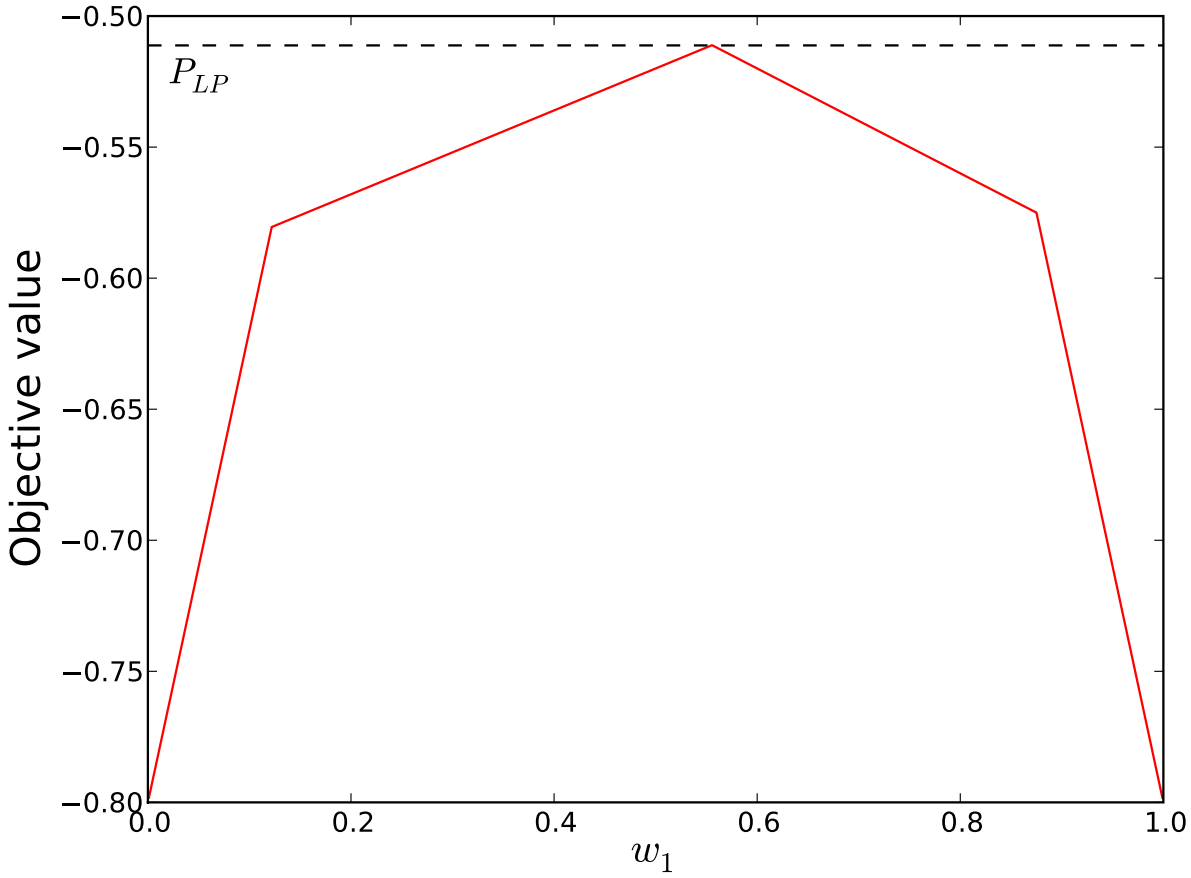


2 examples and 2 hypotheses

$$\min_{\mathbf{d}\in\mathcal{S}^N} \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d} = \max_{\mathbf{w}\in\mathcal{S}^t} \min_{n=1,2,\ldots,N} \left(\sum_{q=1}^{t} u_n^q \, w^q \right)$$



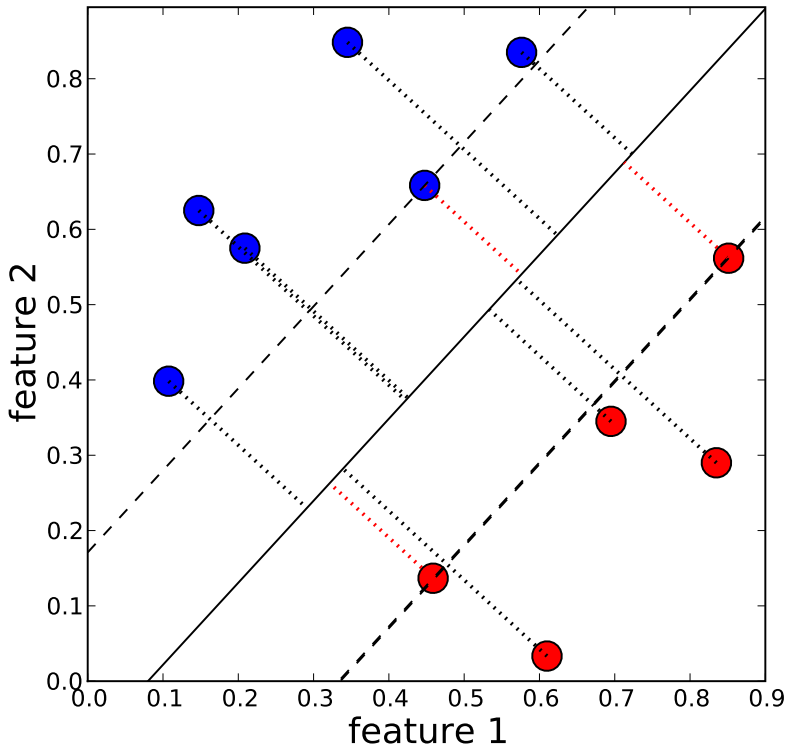
2 examples and 4 hypotheses



4 examples and 2 hypotheses

$$\textit{minmax} = \textit{maximin}$$

Visualizing the margin



Not the margin corresponding to Boosting

1-norm versus 2-norm margin

Boosting with $\mathbf{w}$ in diamond rather than simplex:

$$\min_{\mathbf{d}\in\mathcal{S}^N} \max_{q=1,2,\ldots,t} \left|\mathbf{u}^q \cdot \mathbf{d}\right| = \max_{\mathbf{w}\in\mathcal{D}^t} \min_{n=1,2,\ldots,N} \mathbf{U}_{n*}\mathbf{w}$$

$$= \max_{\mathbf{w}\in\mathbb{R}^t} \min_{n=1,2,\ldots,N} \mathbf{U}_{n*} \frac{\mathbf{w}}{||\mathbf{w}||_1}$$

2-norm margins as in SVMs:

$$\min_{\mathbf{d}\in\mathcal{S}^n} \max_{\mathbf{w}\in\mathbb{R}^t} \mathbf{d}^\top \mathbf{U} \frac{\mathbf{w}}{||\mathbf{w}||_2} = \max_{\mathbf{w}\in\mathbb{R}^t} \min_{\mathbf{d}\in\mathcal{S}^n} \mathbf{d}^\top \mathbf{U} \frac{\mathbf{w}}{||\mathbf{w}||_2}$$

$$= \max_{\mathbf{w}\in\mathbb{R}^t} \min_{n=1,2,\ldots,N} \mathbf{U}_{n*} \frac{\mathbf{w}}{||\mathbf{w}||_2}$$

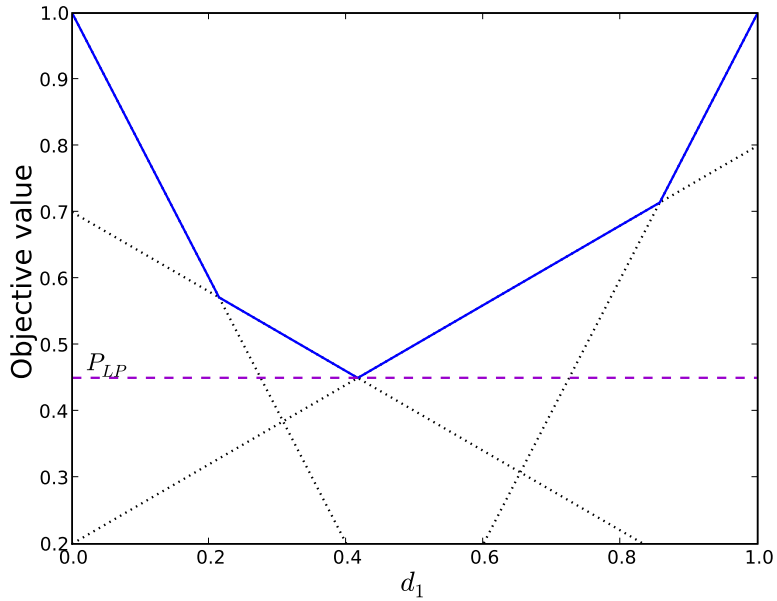
Outline

LPBoost [GS98,RSS+00,DBST02]

Choose distribution that minimizes the maximum edge of current hypotheses by solving w. LP

$$\underbrace{\min_{\sum_n d_n = 1} \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d}}_{P^t_{LP}}$$

All weight is put on examples with minimum margin



Entropy Regularized LPBoost

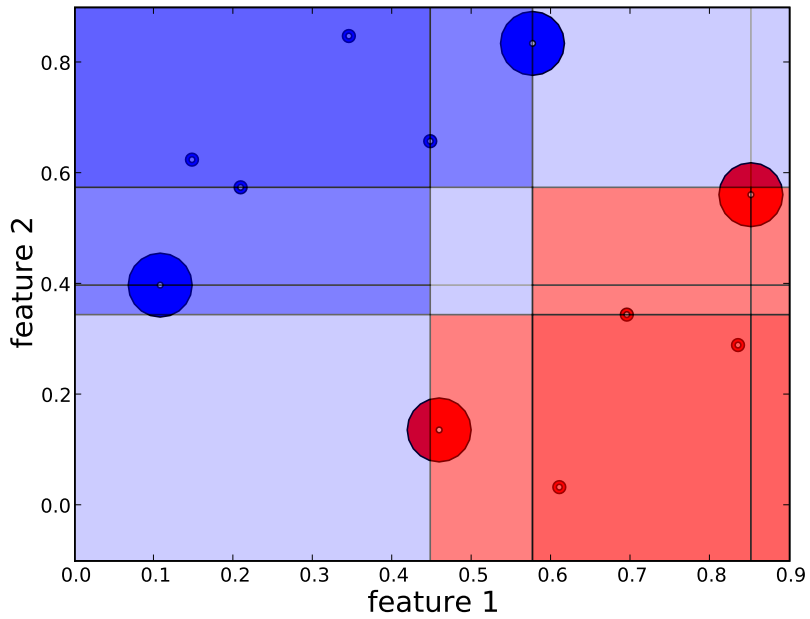
$$\min_{\sum_n d_n = 1} \left(\frac{1}{\eta} \Delta(\mathbf{d}, \mathbf{d}^0) + \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d} \right)$$

- $$\mathbf{d}_n = \frac{\exp^{-\eta \text{ margin of example } n}}{Z}$$ "soft min"

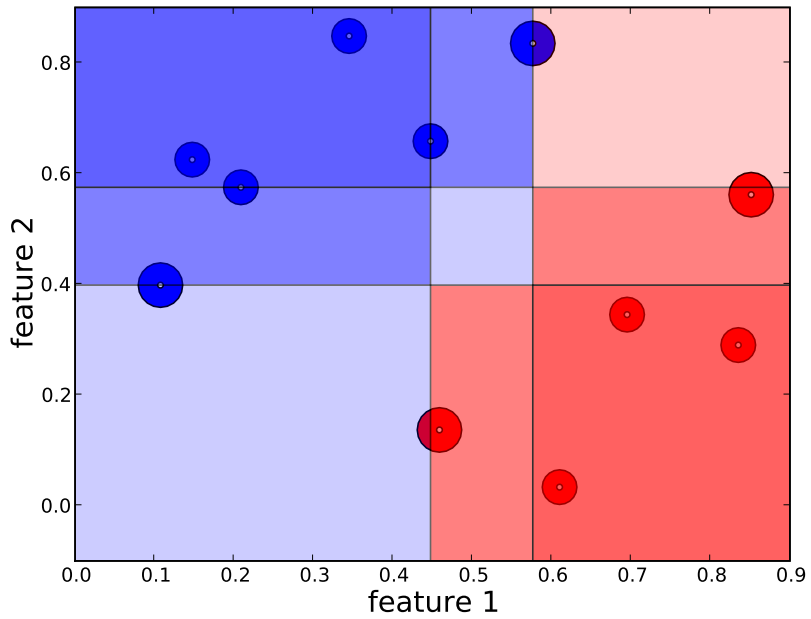
- Form of weights first in ν-Arc algorithm [RSS+00]
- Regularization in $\mathbf{d}$ domain makes problem strongly convex
- Gradient of dual Lipschitz continuous in $\mathbf{w}$ [e.g. HL93,RW97]

Effect of entropy regularization

Different distributions on the examples



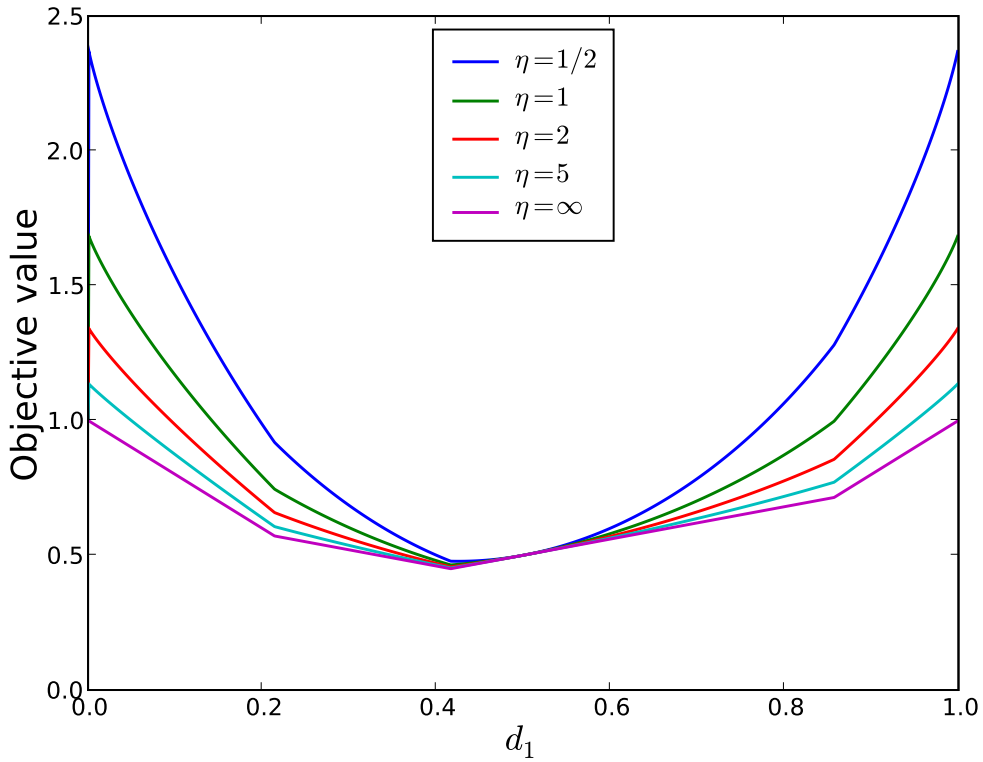
LPBoost: lots of zeros / brittle



ERLPBoost: smoother

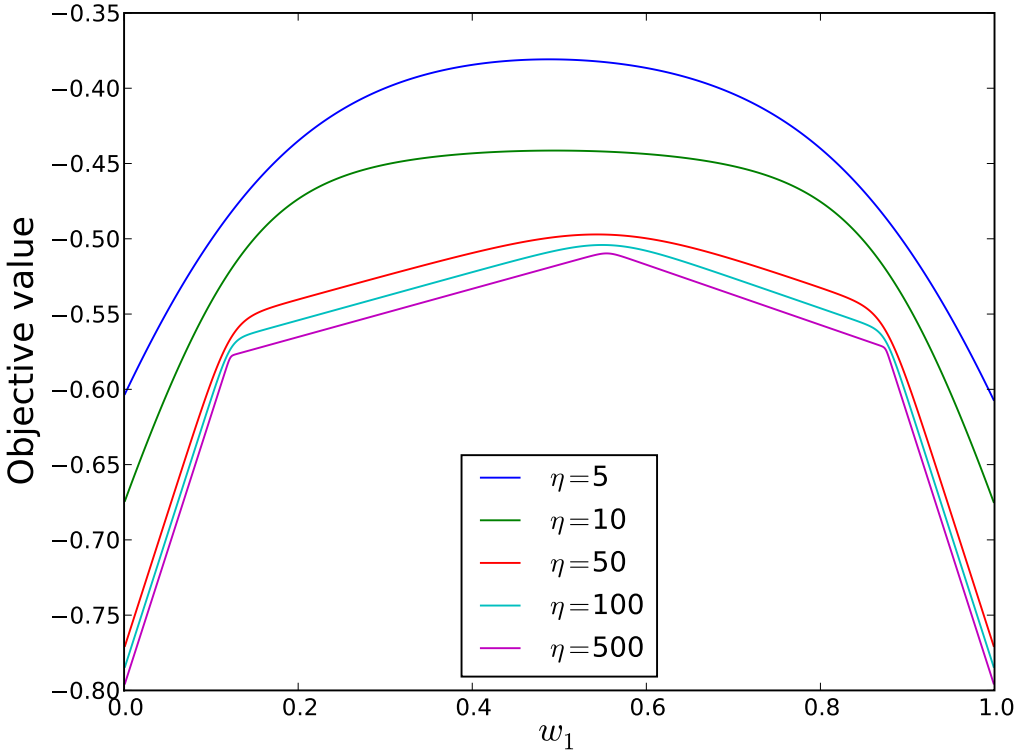
Effect of the regularization in the $\mathbf{d}$ domain

$$\min_{\sum_n d_n=1} \left(\frac{1}{\eta}\Delta(\mathbf{d}, \mathbf{d}^0) + \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d} \right)$$



Uncapped case, $\eta = \infty$ becomes LP objective

w domain



Uncapped case, $\eta = \infty$ becomes LP objective

primal value = dual value

From ERLPBoost to SVMs

Two steps removed

- Replace 1-norm margin by 2-norm margin
- Replace relative entropy regularization by $||.||^2$
-

What is the simplest path?

- minimax duality - Lagrange duality - Fenchel duality

Many things are easier for quadradic regularization:

- Slack variables ✓
- Bias term soooooon

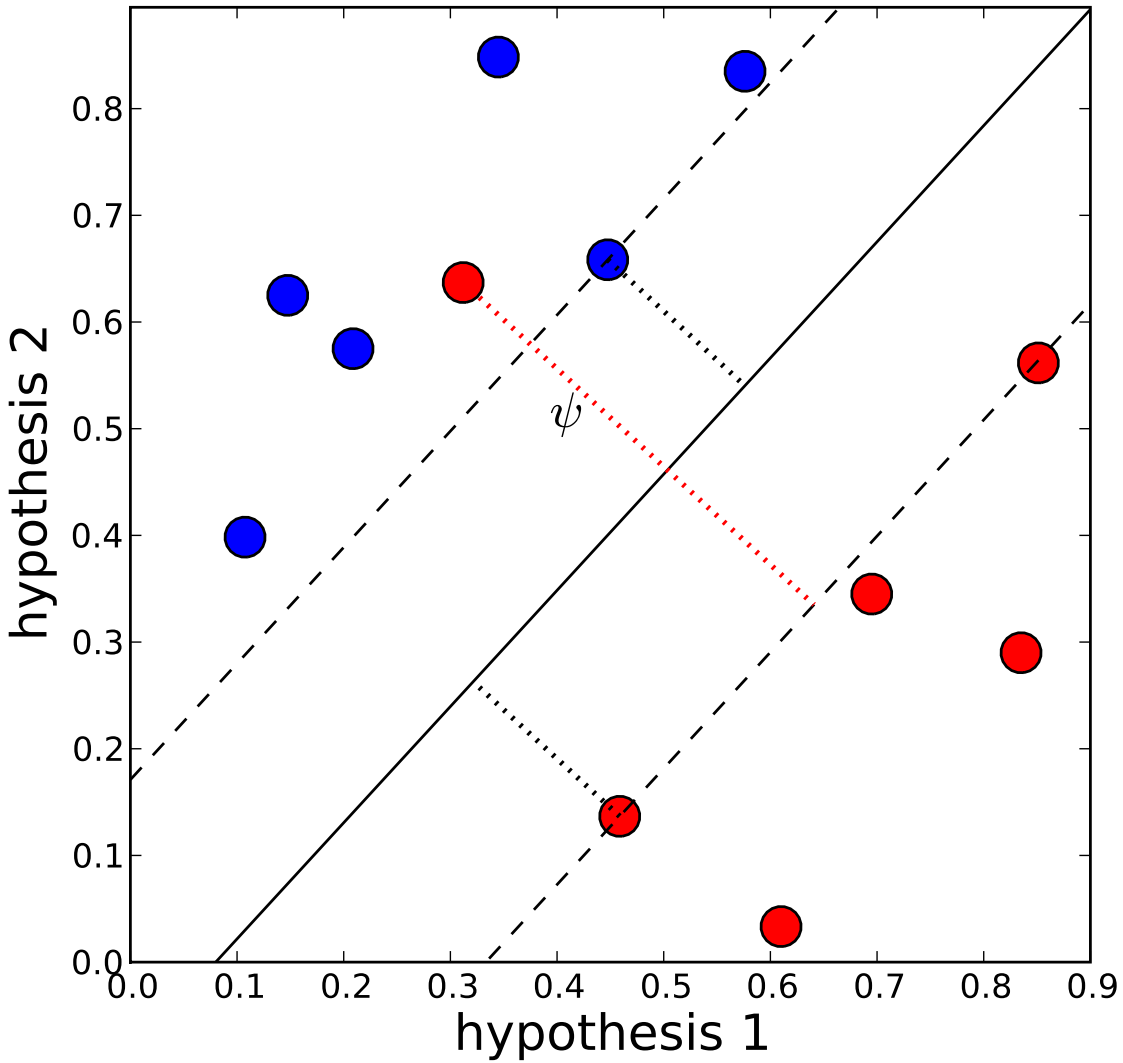
Minimax thm - inseparable case

Slack variables in $\mathbf{w}$ domain = capping in $\mathbf{d}$ domain

$$\min_{\mathbf{d}\in\mathcal{S}^N, \mathbf{d}\le\frac{1}{\nu}\mathbf{1}} \max_{q=1,2,\ldots,t} \underbrace{\mathbf{u}^q\cdot\mathbf{d}}_{\text{edge of hypothesis q}}$$

$$= \max_{\mathbf{w}\in\mathcal{S}^t, \boldsymbol{\psi}\ge\mathbf{0}} \min_{n=1,2,\ldots,N} \underbrace{\left(\sum_{q=1}^{t} u_n^q\, w^q + \psi_n\right)}_{\text{soft margin of example } n} - \frac{1}{\nu}\sum_{n=1}^{N}\psi_n$$

Visualizing the soft margin



Adding slack variables to the algorithms

LPBoost

$$\underbrace{\min_{\sum_n d_n=1, \mathbf{d} \leq \frac{1}{\nu}\mathbf{1}} \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d}}_{P^t_{LP}}$$

All weight put on examples with minimum soft margin ERLPBoost

$$\min_{\sum_n d_n=1, \mathbf{d} \leq \frac{1}{\nu}\mathbf{1}} \left(\frac{1}{\eta} \Delta(\mathbf{d}, \mathbf{d}^0) + \max_{q=1,2,\ldots,t} \mathbf{u}^q \cdot \mathbf{d} \right)$$

$$\mathbf{d}_n = \frac{\exp^{-\eta \text{ soft margin of example } n}}{Z} \qquad \text{"soft min"}$$

AdaBoost - the most common motivation [FS97]

$$d_n^t := \frac{d_n^{t-1} \exp(-w^t u_n^t)}{\sum_{n'} d_{n'}^{t-1} \exp(-w^t u_{n'}^t)},$$

where w^t s.t. $\underbrace{-\ln \sum_n d_n^{t-1} \exp(-w\ u_n^t)}_{\text{dual objective}}$ is minimized

$$\text{Choose } w \text{ s.t. } \frac{\partial -\ln \sum_{n'} d_{n'}^{t-1} \exp(-w\ u_{n'}^t)}{\partial w} = \mathbf{u}^t \cdot \mathbf{d}^t(w) = 0$$

- Easy to implement
- Adjusts distribution so that last hypothesis has edge zero
- When $h_n^t \in \{-1, +1\}$, then $w^t = \frac{1}{2} \ln \frac{1+\mathbf{d}^{t-1} \cdot \mathbf{u}^t}{1-\mathbf{d}^{t-1} \cdot \mathbf{u}^t}$
  when $h_n^t \in [-1, +1]$, then line search
- Gets within half of the optimal hard margin [RSD07]
  but only in the limit

Corrective versus totally corrective

Processing last hypothesis versus all past hypotheses

Corrective	Totally Corrective
AdaBoost	LPBoost
LogitBoost	TotalBoost
AdaBoost*	SoftBoost
SS,Colt08	ERLPBoost

From AdaBoost to ERLPBoost

AdaBoost (as interpreted in [KW99,La99])

Primal:

$$\min_{\mathbf{d}} \quad \Delta(\mathbf{d}, \mathbf{d}^{t-1})$$
$$\text{s.t.} \quad \mathbf{d} \cdot \mathbf{u}^t = 0, \ \|\mathbf{d}\|_1 = 1$$

Dual:

$$\max_{\mathbf{w}} \quad -ln \sum_n d_n^{t-1} \exp(-u_n^t w)$$

Achieves half of optimum hard margin in the limit

AdaBoost* [RW05]

Primal:

$$\min_{\mathbf{d}} \quad \Delta(\mathbf{d}, \mathbf{d}^{t-1})$$
$$\text{s.t.} \quad \mathbf{d} \cdot \mathbf{u}^t \leq \gamma_t, \ \|\mathbf{d}\|_1 = 1$$

Dual:

$$\max_{\mathbf{w}} \ -ln \sum_n d_n^{t-1} \exp(-u_n^t w) - \gamma_t \|w\|_1$$
$$\text{s.t.} \quad w \geq 0$$

where edge bound γ_t is adjusted downward by a heuristic

Good iteration bound for reaching optimum hard margin

SoftBoost [WGR07]

Primal:

$$\begin{aligned} \min_{\mathbf{d}} \quad & \Delta(\mathbf{d}, \mathbf{d}^0) \\ \text{s.t.} \quad & \|\mathbf{d}\|_1 = 1,\ \mathbf{d} \le \tfrac{1}{\nu}\mathbf{1} \\ & \mathbf{d} \cdot \mathbf{u}^q \le \gamma_t,\ 1 \le q \le t \end{aligned}$$

Dual:

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\psi}} \quad & -\ln \sum_n \mathbf{d}_n^0 \exp(-\eta \sum_{q=1}^{t} u_n^q w^q - \eta\psi_n) \\ & -\tfrac{1}{\nu}\|\boldsymbol{\psi}\|_1 - \gamma_t \|\mathbf{w}\|_1 \\ \text{s.t.} \quad & \mathbf{w} \ge 0,\ \boldsymbol{\psi} \ge 0 \end{aligned}$$

where edge bound γ_t is adjusted downward by a heuristic

Good iteration bound for reaching soft margin

ERLPBoost [WGV08]

Primal:

$$\begin{aligned} \min_{\mathbf{d},\gamma} \quad & \gamma + \tfrac{1}{\eta}\Delta(\mathbf{d}, \mathbf{d}^0) \\ \text{s.t.} \quad & \|\mathbf{d}\|_1 = 1,\ \mathbf{d} \le \tfrac{1}{\nu}\mathbf{1} \\ & \mathbf{d} \cdot \mathbf{u}^q \le \gamma,\ 1 \le q \le t \end{aligned}$$

Dual:

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\psi}} \quad & -\tfrac{1}{\eta} \ln \sum_n \mathbf{d}_n^0 \exp(-\eta \sum_{q=1}^{t} u_n^q w^q - \eta\psi_n) \\ & -\tfrac{1}{\nu}\|\boldsymbol{\psi}\|_1 \\ \text{s.t.} \quad & \mathbf{w} \ge 0,\ \|\mathbf{w}\|_1 = 1,\ \boldsymbol{\psi} \ge 0 \end{aligned}$$

where for the iteration bound η is fixed to $\max(\frac{2}{\epsilon} \ln \frac{N}{\nu}, \frac{1}{2})$

Good iteration bound for reaching soft margin

Corrective ERLPBoost [SS08]

Primal:

$$\begin{aligned} \min_{\mathbf{d}} \quad & \left(\sum_{q=1}^{t} w^q \mathbf{u}^q\right) \cdot \mathbf{d} + \tfrac{1}{\eta}\Delta(\mathbf{d}, \mathbf{d}^0) \\ \text{s.t.} \quad & \|\mathbf{d}\|_1 = 1,\ \mathbf{d} \leq \tfrac{1}{\nu}\mathbf{1} \end{aligned}$$

Dual:

$$\begin{aligned} \min_{\boldsymbol{\psi}} \quad & -\tfrac{1}{\eta} \ln \sum_n \mathbf{d}_n^0 \exp\left(-\eta \sum_{q=1}^{t} u_n^q w^q - \eta\psi_n\right) - \tfrac{1}{\nu}\|\boldsymbol{\psi}\|_1 \\ \text{s.t.} \quad & \boldsymbol{\psi} \geq 0 \end{aligned}$$

where for the iteration bound η is fixed to $\max(\frac{2}{\epsilon} \ln \frac{N}{\nu}, \frac{1}{2})$

Good iteration bound for reaching soft margin

Iteration bounds

Corrective	Totally Corrective
AdaBoost	LPBoost
LogitBoost	TotalBoost
AdaBoost*	SoftBoost
SS,Colt08	ERLPBoost

- Strong oracle: returns hypothesis with maximum edge
- Weak oracle: returns hypothesis with edge $\geq g$

- In $O(\frac{\log \frac{N}{\nu}}{\epsilon^2})$ iterations
  within ϵ of maximum soft margin for strong oracle
  or within ϵ of g for weak oracle
- Ditto for hard margin case
- When $g > 0$, in $O(\frac{\log N}{g^2})$ iterations
  consistency with weak oracle

LPBoost may require $\Omega(N)$ iterations

		w_1	w_2	w_3	w_4	w_5	margin
		0	0	0	0	0	
d_1	.125	+1	-.95	-.93	-.91	-.99	–
d_2	.125	+1	-.95	-.93	-.91	-.99	–
d_3	.125	+1	-.95	-.93	-.91	-.99	–
d_4	.125	+1	-.95	-.93	-.91	-.99	–
d_5	.125	-.98	+1	-.93	-.91	+.99	–
d_6	.125	-.97	-.96	+1	-.91	+.99	–
d_7	.125	-.97	-.95	-.94	+1	+.99	–
d_8	.125	-.97	-.95	-.93	-.92	+.99	–
edge		.0137	-.7075	-.6900	-.6725	.0000	
value	**-1**						

LPBoost may require $\Omega(N)$ iterations

		w_1	w_2	w_3	w_4	w_5	margin
		1	0	0	0	0	
d_1	0	+1	-.95	-.93	-.91	-.99	1
d_2	0	+1	-.95	-.93	-.91	-.99	1
d_3	0	+1	-.95	-.93	-.91	-.99	1
d_4	0	+1	-.95	-.93	-.91	-.99	1
d_5	1	-.98	+1	-.93	-.91	+.99	-.98
d_6	0	-.97	-.96	+1	-.91	+.99	-.97
d_7	0	-.97	-.95	-.94	+1	+.99	-.97
d_8	0	-.97	-.95	-.93	-.92	+.99	-.97
edge		-.98	1	-.93	-.91	.99	
value	-1	-.98					

LPBoost may require $\Omega(N)$ iterations

		w_1	w_2	w_3	w_4	w_5	margin
		0	1	0	0	0	
d_1	0	+1	-.95	-.93	-.91	-.99	-.95
d_2	0	+1	-.95	-.93	-.91	-.99	-.95
d_3	0	+1	-.95	-.93	-.91	-.99	-.95
d_4	0	+1	-.95	-.93	-.91	-.99	-.95
d_5	0	-.98	+1	-.93	-.91	+.99	1
d_6	1	-.97	-.96	+1	-.91	+.99	-.96
d_7	0	-.97	-.95	-.94	+1	+.99	-.95
d_8	0	-.97	-.95	-.93	-.92	+.99	-.95
edge		-.97	-.96	1	-.91	.99	
value	-1	-.98	-.96				

LPBoost may require $\Omega(N)$ iterations

		w_1	w_2	w_3	w_4	w_5	margin
		0	0	1	0	0	
d_1	0	+1	-.95	-.93	-.91	-.99	-.93
d_2	0	+1	-.95	-.93	-.91	-.99	-.93
d_3	0	+1	-.95	-.93	-.91	-.99	-.93
d_4	0	+1	-.95	-.93	-.91	-.99	-.93
d_5	0	-.98	+1	-.93	-.91	+.99	-.93
d_6	0	-.97	-.96	+1	-.91	+.99	1
d_7	1	-.97	-.95	-.94	+1	+.99	-.94
d_8	0	-.97	-.95	-.93	-.92	+.99	-.93
edge		-.97	-.95	-.94	1	.99	
value	-1	-.98	-.96	-.94			

LPBoost may require $\Omega(N)$ iterations

		w_1	w_2	w_3	w_4	w_5	margin
		0	0	0	1	0	
d_1	0	+1	-.95	-.93	-.91	-.99	-.91
d_2	0	+1	-.95	-.93	-.91	-.99	-.91
d_3	0	+1	-.95	-.93	-.91	-.99	-.91
d_4	0	+1	-.95	-.93	-.91	-.99	-.91
d_5	0	-.98	+1	-.93	-.91	+.99	-.91
d_6	0	-.97	-.96	+1	-.91	+.99	-.91
d_7	0	-.97	-.95	-.94	+1	+.99	1
d_8	1	-.97	-.95	-.93	-.92	+.99	-.92
edge		-.97	-.95	-.94	-.92	.99	
value	-1	-.98	-.96	-.94	-.92		

LPBoost may require $\Omega(N)$ iterations

		w_1	w_2	w_3	w_4	w_5	margin
		.5	.0026	0	0	.4975	
d_1	.497	+1	-.95	-.93	-.91	-.99	.0051
d_2	0	+1	-.95	-.93	-.91	-.99	.0051
d_3	0	+1	-.95	-.93	-.91	-.99	.0051
d_4	0	+1	-.95	-.93	-.91	-.99	.0051
d_5	0	-.98	+1	-.93	-.91	+.99	.0051
d_6	.490	-.97	-.96	+1	-.91	+.99	.0051
d_7	0	-.97	-.95	-.94	+1	+.99	.0051
d_8	.013	-.97	-.95	-.93	-.92	+.99	.0051
edge		.0051	.0051	.9055	.9100	.0051	
value	-1	-.98	-.96	-.94	-.92	.0051	

No ties!

LPBoost may return bad final hypothesis

How good is the master hypothesis returned by LPBoost compared to the best possible convex combination of hypotheses?

Any linearly separable dataset can be reduced to a dataset on which LPBoost misclassifies all examples by

- adding a bad hypothesis
- adding a bad example

Before adding a bad example

		w_1	w_2	w_3	w_4	w_5	margin
		.5	.0026	0	0	.4975	
d_1	.497	+1	-.95	-.93	-.91	-.99	.0051
d_2	0	+1	-.95	-.93	-.91	-.99	.0051
d_3	0	+1	-.95	-.93	-.91	-.99	.0051
d_4	0	+1	-.95	-.93	-.91	-.99	.0051
d_5	0	-.98	+1	-.93	-.91	+.99	.0051
d_6	.490	-.97	-.96	+1	-.91	+.99	.0051
d_7	0	-.97	-.95	-.94	+1	+.99	.0051
d_8	.013	-.97	-.95	-.93	-.92	+.99	.0051
d_9		-.03	-.03	-.03	-.03	-.03	
edge		.0051	.0051	.9055	.9100	.0051	
value							.0051

After adding a bad example

		w_1 .4445	w_2 .0534	w_3 .0543	w_4 .0561	w_5 .3917	margin
d_1	0	+1	-.95	-.93	-.91	-.99	−.0956
d_2	0	+1	-.95	-.93	-.91	-.99	−.0956
d_3	0	+1	-.95	-.93	-.91	-.99	−.0956
d_4	0	+1	-.95	-.93	-.91	-.99	−.0956
d_5	0	-.98	+1	-.93	-.91	+.99	−.0959
d_6	0	-.97	-.96	+1	-.91	+.99	−.0913
d_7	0	-.97	-.95	-.94	+1	+.99	−.0891
d_8	0	-.97	-.95	-.93	-.92	+.99	−.1962
d_9	1	**-.03**	**-.03**	**-.03**	**-.03**	**-.03**	−.03
edge		−.03	−.03	−.03	−.03	−.03	
value							-.03

Synopsis

- LPBoost often unstable
- For safety, add relative entropy regularization
- Corrective algs
  - Sometimes easy to code
  - Fast per iteration
- Totally corrective algs
  - Smaller number of iterations
  - Faster overall time when ϵ small
- Weak versus strong oracle makes a big difference in practice

$O(\frac{\log N}{\epsilon^2})$ iteration bounds

Good

- Bound is major design tool
- Any reasonable Boosting algorithm should have this bound

Bad

- Bound is weak

	$\frac{\ln N}{\epsilon^2} \geq N$
$\epsilon = .01$	$N \leq 1.2\ 10^5$
$\epsilon = .001$	$N \leq 1.7\ 10^7$

- Why are totally corrective algorithms much better in practice?

Lower bounds on the number of iterations

- Majority of $\Omega(\frac{\log N}{g^2})$ hypotheses for achieving consistency with weak oracle of guarantee g [Fr95]

- Easy:
  - $\Omega(\frac{1}{\epsilon^2})$ iterations to get within ϵ of hard margin w. strong oracle
  - Uses Hadamard matrix
- Harder:
  - $\Omega(\frac{\log N}{\epsilon^2})$ iterations for strong oracle
    Uses random matrices [KY99,Ne83]

Conclusion

- Adding relative entropy regularization of LPBoost leads to good boosting alg.
- Boosting is instantiation of MaxEnt and MinxEnt principles [Jaines 57,Kullback 59]
- Relative entropy regularization smoothes one-norm regularization

Open

- When hypotheses have one-sided error then $O(\frac{\log N}{\epsilon})$ iterations suffice [As00,HW03]
- Does ERLPBoost have $O(\frac{\log N}{\epsilon})$ bound when hypotheses one-sided?
- Replace geometric optimizers by entropic ones
- Compare ours with Freund's algorithms that don't just cap, but forget examples

Outline

Overview of lower bounds

- $\Omega(\frac{1}{\epsilon^2})$ for Hadamard matrices
- $\Omega(\frac{\log N}{\epsilon^2})$ for random matrices
  but $t \leq n^{1/2-\nu}$
- Conjecture: Hadamard is the hardest

Boosting = greedy method for increasing margin

	w_1^1
d_1^1	0
d_2^1	1
d_3^1	-1

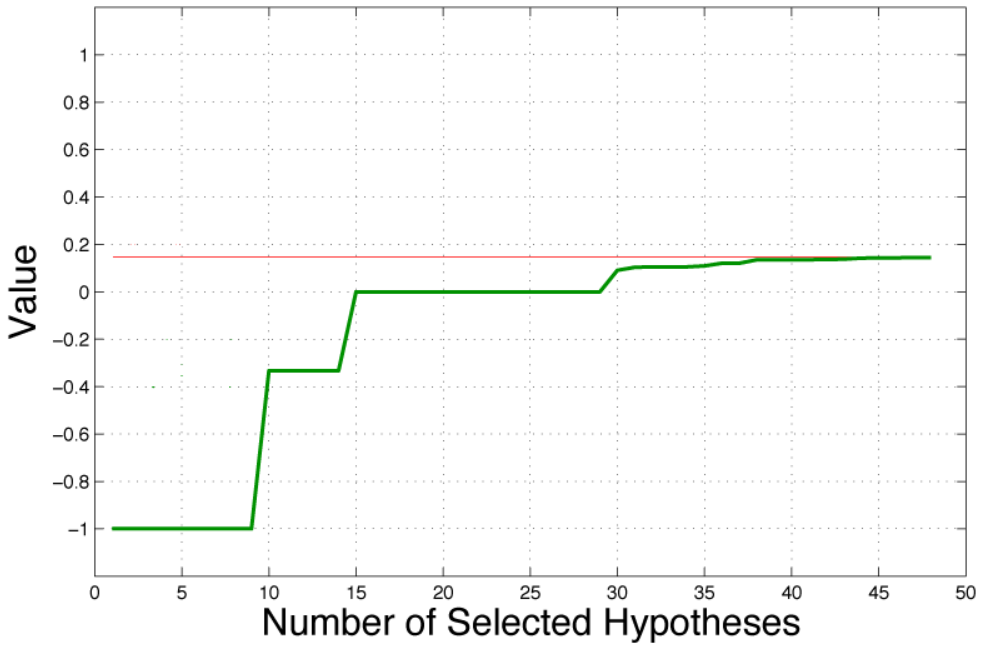
iteration 1

	w_1^2	w_2^2
d_1^2	0	-1
d_2^2	1	0
d_3^2	-1	1

iteration 2

	w_1^3	w_2^3	w_3^3
d_1^3	0	-1	1
d_2^3	1	0	-1
d_3^3	-1	1	0

iteration 3



Upper bound versus lower bounds

Upper bounds:

- Design algorithms such value increases fast
- Want value within ϵ in small number of iterations
- For our Boosting algorithms
  $O(\frac{\log N}{\epsilon^2})$ iterations suffice to get within ϵ

Lower bounds:

- Find a setup where for particular algorithm or any algorithm that curve increases slowly
- Conjecture: it requires $t = \Omega(\frac{\log N}{\epsilon^2})$ iterations to get within ϵ

$$\#\text{ of iterations } t \geq \frac{\log N}{\epsilon^2} \iff \text{gap } \epsilon \geq \sqrt{\frac{\log n}{t}}$$

Simplifying the lower bound setup

How fast the value increases depends on

- matrix $\mathbf{U}$
- which hypotheses the oracle chooses
- which distributions the algorithm chooses

Simplification: Find $\mathbf{U}$ s.t.

- $$\max_{\mathbf{U}_t \text{ any } t \text{ columns of } \mathbf{U}} \text{val}(\mathbf{U}_t)$$

  increaseses slowly as a function of t
- Conjecture: There are $\mathbf{U}$ s.t.
  it requires $\Omega(\frac{\log N}{\epsilon^2})$ iterations to get within ϵ of val($\mathbf{U}$)

$\Omega(1/\epsilon^2)$ bound with Hadamard matrices

$$\mathbf{H}_2 = \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix} \quad \mathbf{H}_4 = \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}$$

Let $\widehat{\mathbf{H}}_n$ be $\mathbf{H}_n$ with first row removed

$$\widehat{\mathbf{H}}_4 = \begin{bmatrix} +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 \end{bmatrix}$$

We will use $\mathbf{U} = \widehat{\mathbf{H}}_n$ has our hard matrix

$\mathcal{D}\,\mathcal{S}$ game value

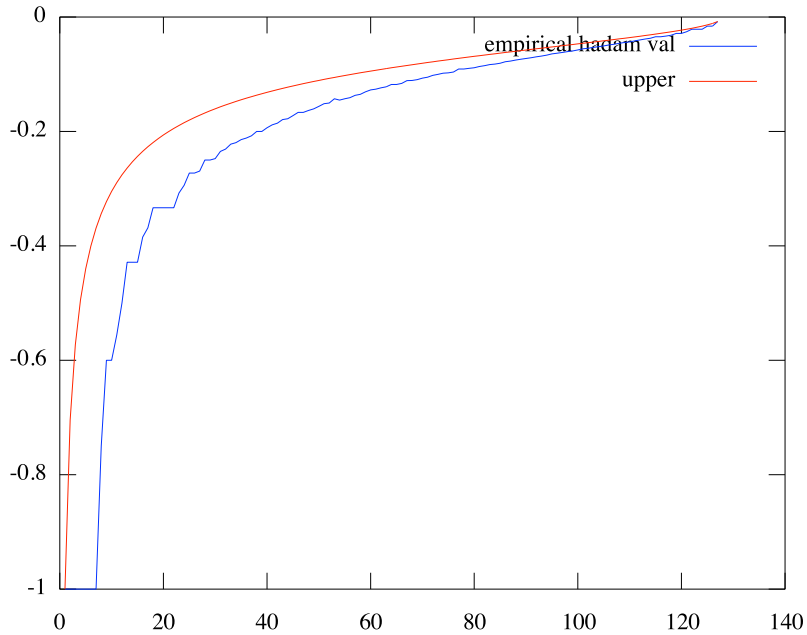
Let $\mathcal{S}^n$ denote the n dimensional probability simplex and $\mathcal{D}^n := \{\mathbf{d} \in \mathbf{R}^n : \|\mathbf{d}\|_1 \leq 1\}$.
Define the value of a matrix $\mathbf{U} \in \mathbf{R}^{r \times c}$ as

$$\begin{aligned}
\text{val}(\mathbf{U}) &= \min_{\mathbf{d} \in \mathcal{D}^r} \max_{\mathbf{w} \in \mathcal{S}^c} \mathbf{d}^\top \mathbf{U}\ \mathbf{w} \\
&\overset{\text{minimax thm}}{=} \max_{\mathbf{w} \in \mathcal{S}^c} \min_{\mathbf{d} \in \mathcal{D}^r} \mathbf{d}^\top \mathbf{U}\ \mathbf{w} \\
&= \max_{\mathbf{w} \in \mathcal{S}^c} \min_{\mathbf{r}=1,\ldots,r} \pm \underbrace{[\mathbf{U}\,\mathbf{w}]_{\mathbf{r}}}_{\text{margin}} \\
&= \max_{\mathbf{w} \in \mathcal{S}^c} - \max_{\mathbf{r}=1,\ldots,r} |\mathbf{U}\,\mathbf{w}|_{\mathbf{r}} \\
&= - \min_{\mathbf{w} \in \mathcal{S}^c} \|\mathbf{U}\,\mathbf{w}\|_\infty
\end{aligned}$$

Bound for $\mathbf{U}_t$

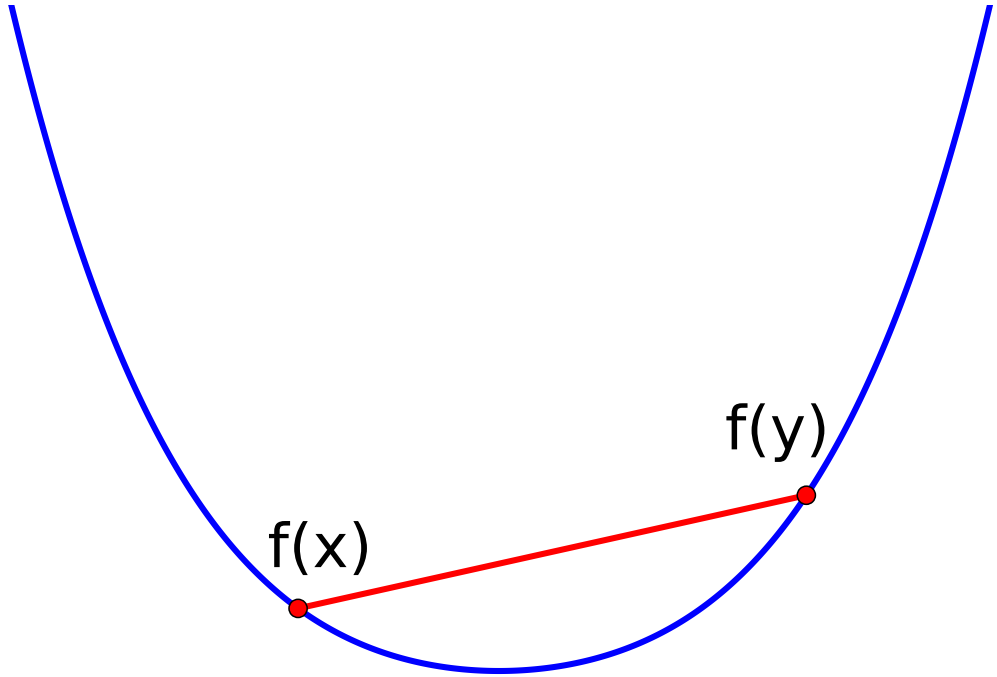
$$\forall \mathbf{u} \in \mathbf{R}^q : \| \mathbf{u} \|_\infty \geq \frac{1}{\sqrt{q}} \| \mathbf{u} \|_2$$

$$\begin{aligned}
-\min_{\mathbf{w}\in\mathcal{S}^t} \| \mathbf{U}_t \mathbf{w} \|_\infty &\leq -\frac{1}{\sqrt{n-1}} \min_{\mathbf{w}\in\mathcal{S}^t} \| \mathbf{U}_t \mathbf{w} \|_2 \\
&= -\frac{1}{\sqrt{n-1}} \min_{\mathbf{w}\in\mathcal{S}^t} \sqrt{\mathbf{w}^\top \underbrace{\mathbf{U}_t^\top \mathbf{U}_t}_{n\,\mathcal{I}_t - ones(t,t)} \mathbf{w}} \\
&= -\min_{\mathbf{w}\in\mathcal{S}^t} \sqrt{\frac{n}{n-1} \underbrace{\mathbf{w}^\top \mathbf{w}}_{\geq 1/t} - \frac{1}{n-1}} \\
&= -\sqrt{\frac{n-t}{(n-1)t}} \\
&\leq -\frac{1}{\sqrt{2t}}, \quad \text{for } 1 \leq t \leq \frac{n-1}{2}
\end{aligned}$$



- Our upper bound $\mathrm{val}(\mathbf{U}_t) \leq -\frac{1}{\sqrt{2t}}$: gives $\Omega(\frac{1}{\epsilon^2})$ iteration bound
- Conjecture $\mathrm{val}(\mathbf{U}_t) \leq -\sqrt{\frac{\log N}{t}}$: gives $\Omega(\frac{\log N}{\epsilon^2})$ iteration bound
- Conjecture: $\widehat{\mathbf{H}}_n$ curve is the lowest curve for any matrix $\mathbf{U}$ with n rows s.t. $\mathrm{val}(\mathbf{U}) = 0$
- For random matrices, $\mathrm{val}(\mathbf{U}_t) \leq -c\sqrt{\frac{\log N}{t}}$ whp, for $t \leq N^{1/2-\nu}$ [KY99,Ne83]

Convex Function - Common Definition

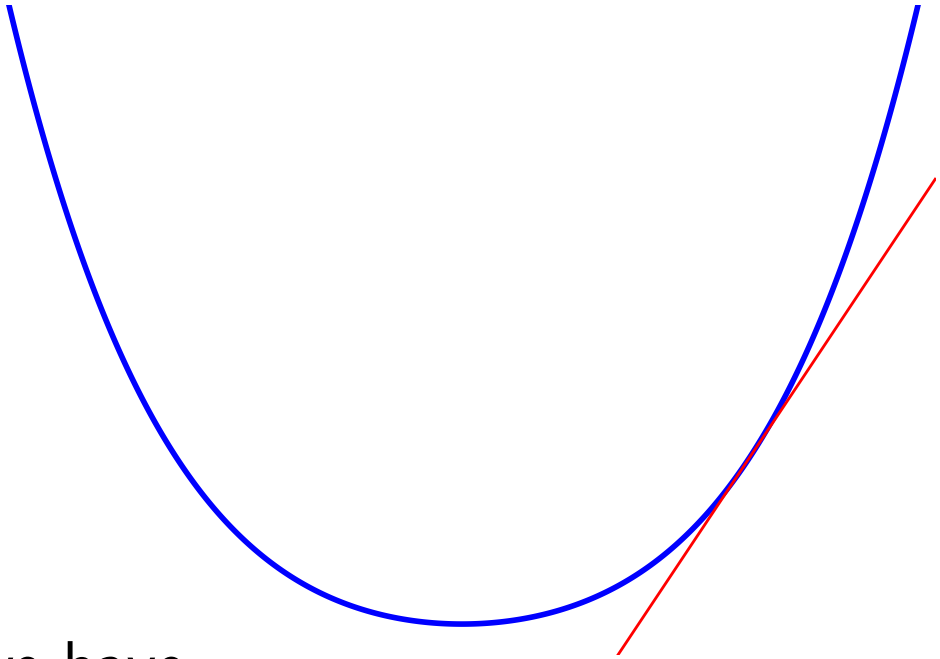


A function f is convex if, and only if, for all x, y and $\alpha \in (0, 1)$ we have

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

A Key Property of Convex Functions

The First Order Taylor approximation always lower bounds the function

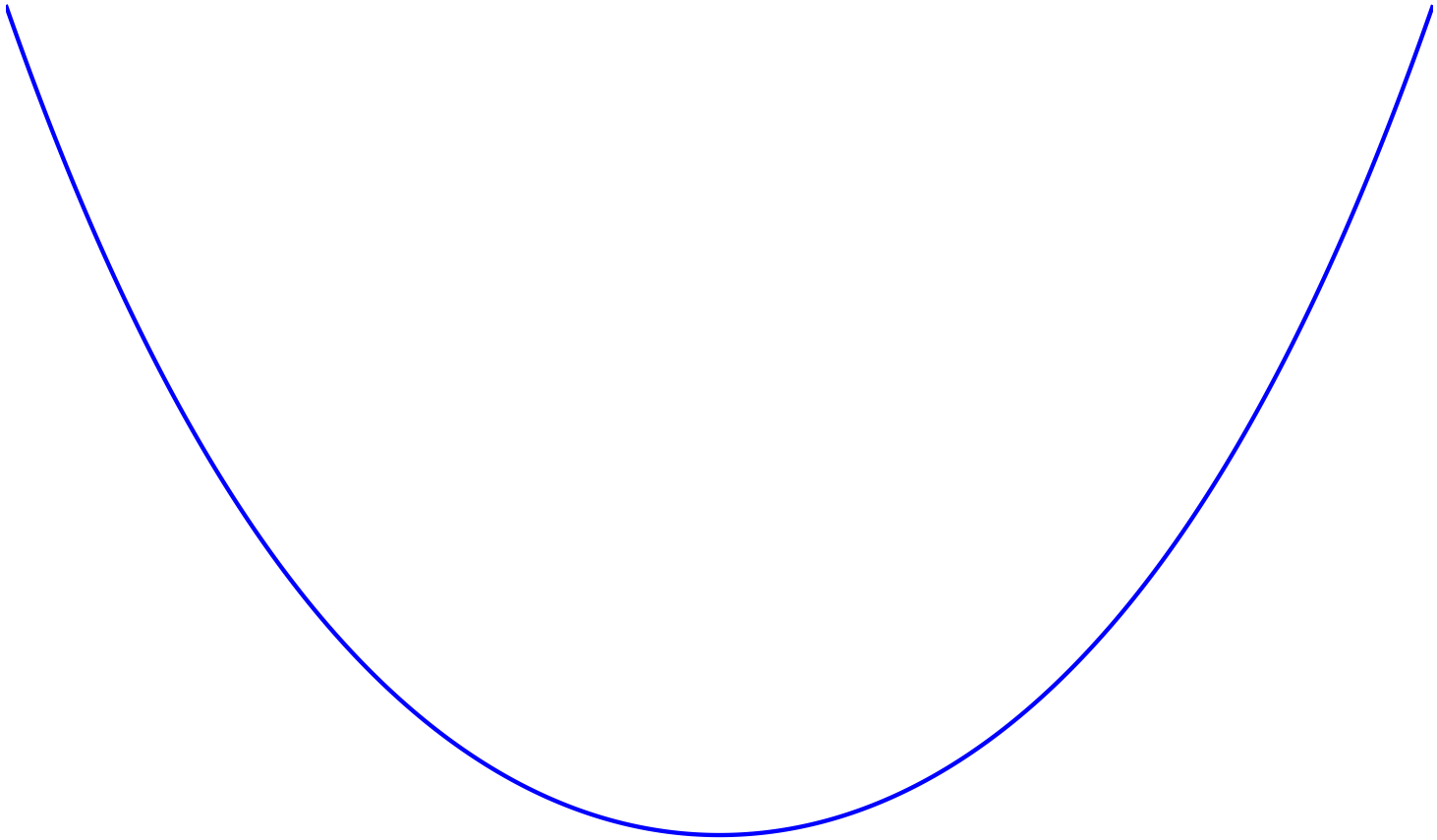


For any x and y we have

$$f(x) \geq f(y) + \langle x - y, \nabla f(y) \rangle$$

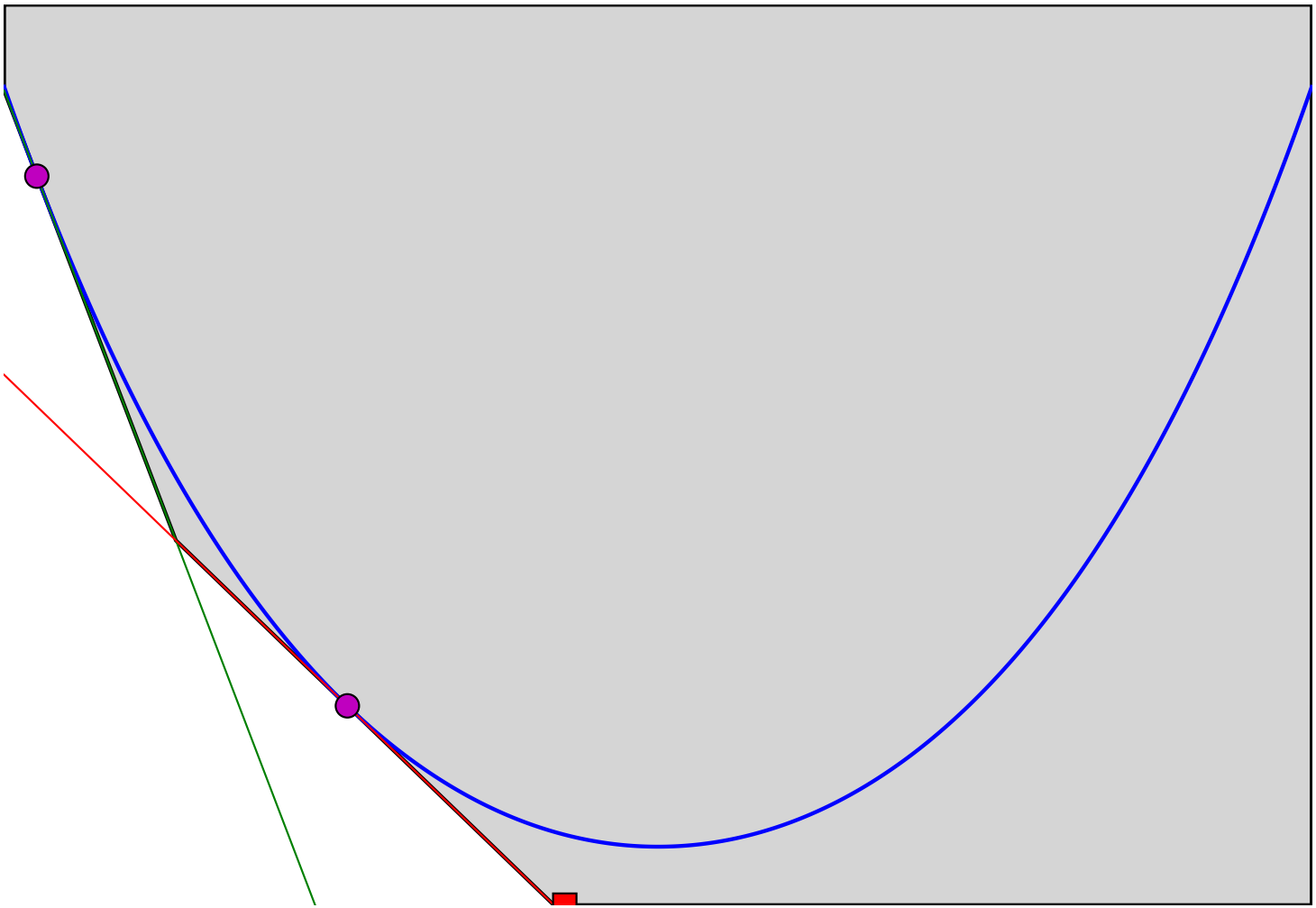
$$\Leftrightarrow \underbrace{f(x) - f(y) + \langle x - y, \nabla f(y) \rangle}_{} \geq 0$$

Cutting Plane Methods [Kelly 60]

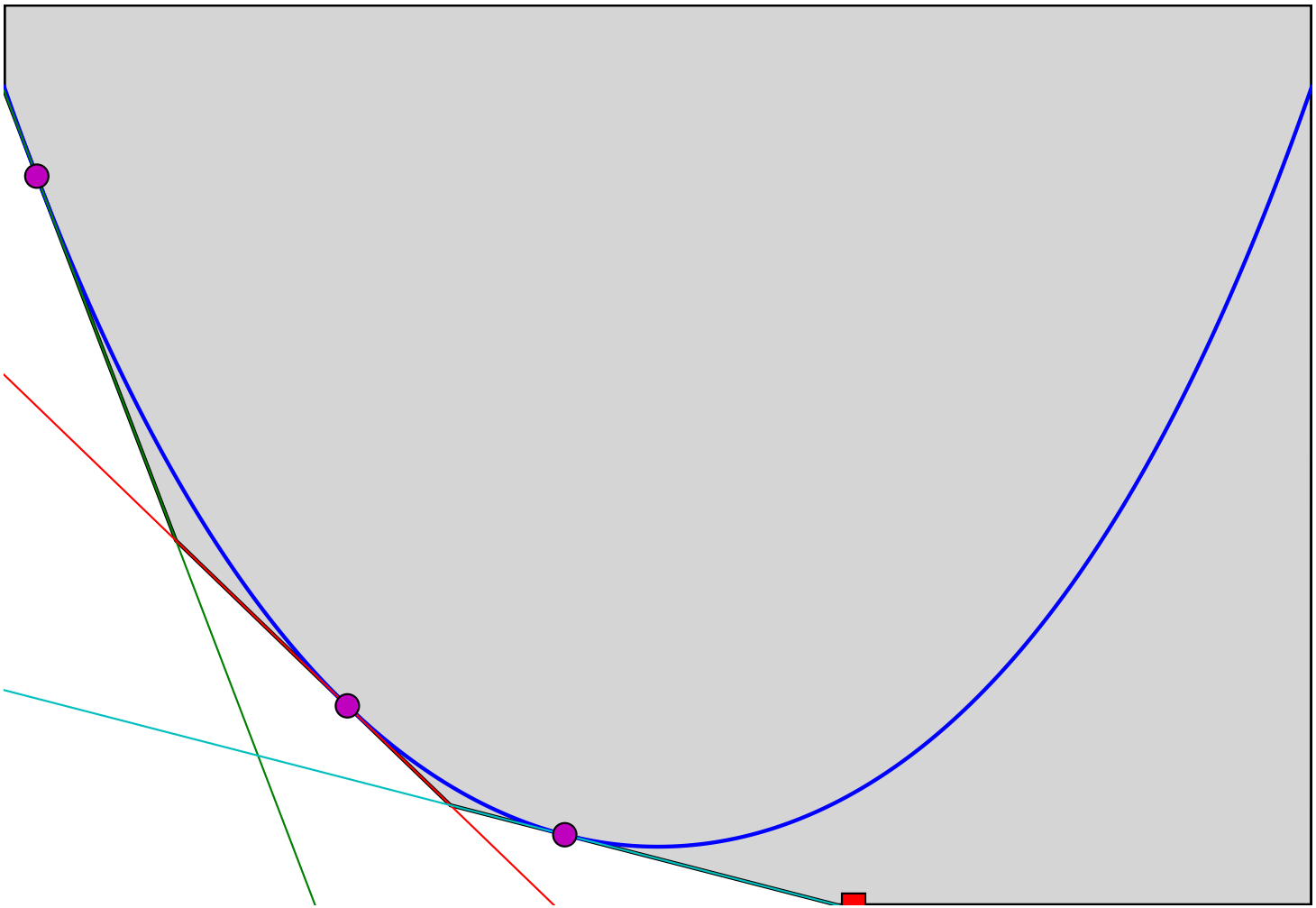


Cutting Plane Methods [Kelly 60]

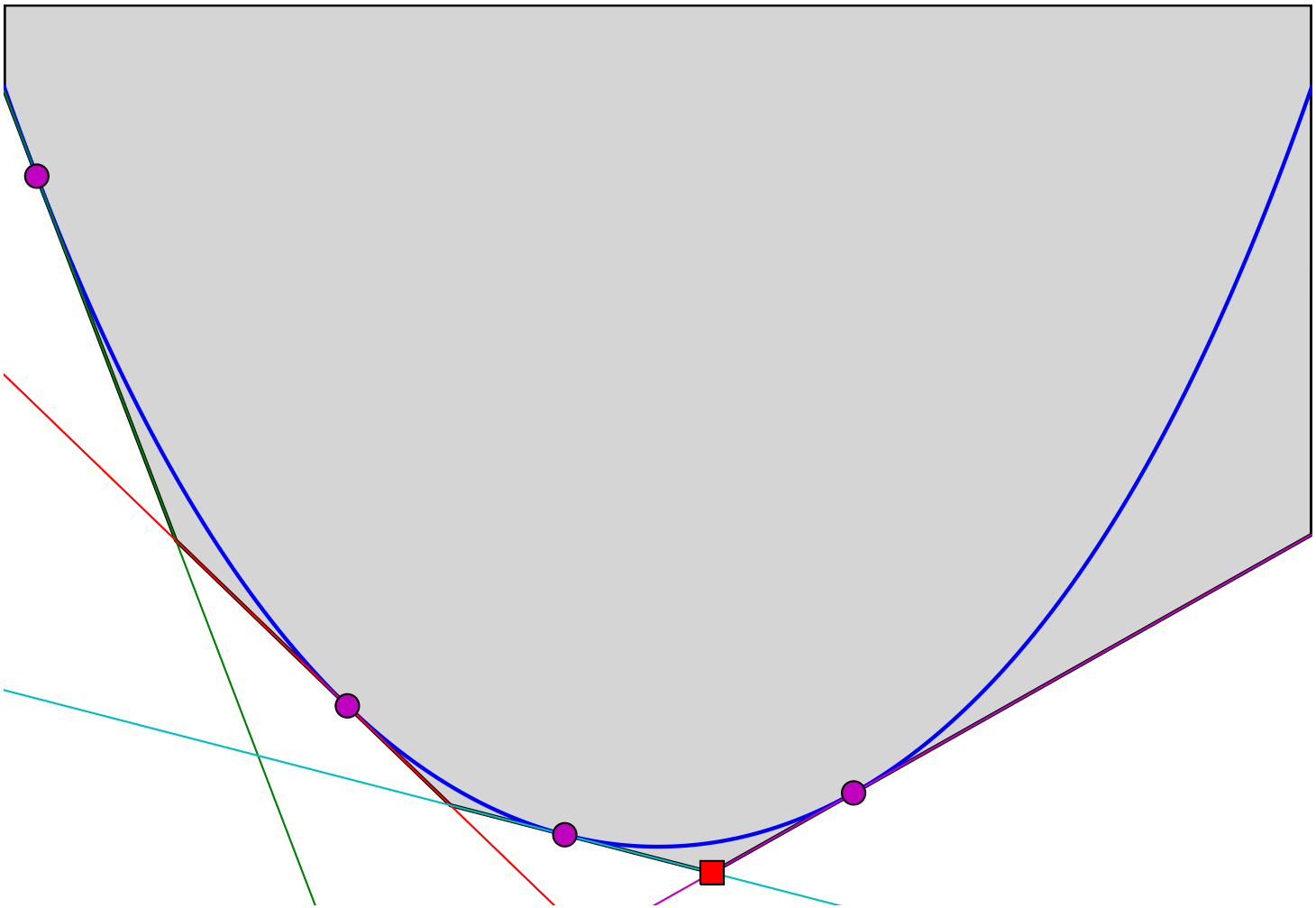
Cutting Plane Methods [Kelly 60]



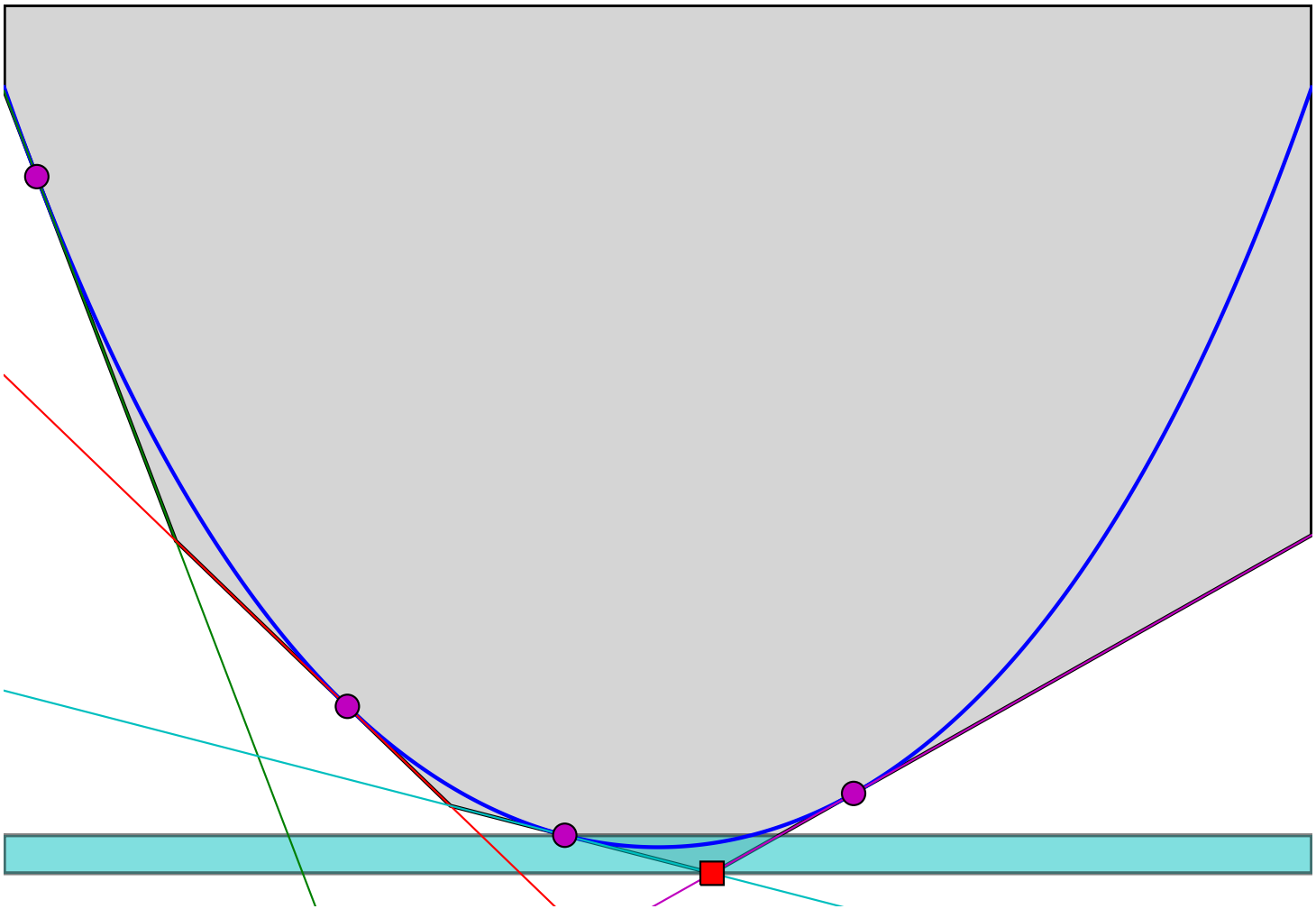
Cutting Plane Methods [Kelly 60]



Cutting Plane Methods [Kelly 60]



Monitoring Convergence



In a Nutshell

- Cutting Plane Methods work by forming the piecewise linear lower bound

$$f(x) \geq f_t^{\mathrm{CP}}(x) := \max_{1 \leq i \leq t} \{f(x_{i-1}) + \langle x - x_{i-1}, s_i \rangle\}.$$

  where s_i denote gradients $\nabla f(x_{i-1})$.

- At iteration t the set $\{x_i\}_{i=0}^{t-1}$ is augmented by

$$x_t := \operatorname*{argmin}_x f_t^{\mathrm{CP}}(x).$$

- Stop when the duality gap

$$\epsilon_t := \min_{0 \leq i \leq t} f(x_i) - f_t^{\mathrm{CP}}(x_t)$$

  falls below a pre-specified threshold ϵ.

In a Nutshell

- Cutting Plane Methods work by forming the piecewise linear lower bound

$$f(x) \geq f_t^{\mathrm{CP}}(x) := \max_{1 \leq i \leq t} \{f(x_{i-1}) + \langle x - x_{i-1}, s_i \rangle\}.$$

  where s_i denote gradients $\nabla f(x_{i-1})$.
- At iteration t the set $\{x_i\}_{i=0}^{t-1}$ is augmented by

$$x_t := \operatorname*{argmin}_x f_t^{\mathrm{CP}}(x).$$

- Stop when the duality gap

$$\epsilon_t := \min_{0 \leq i \leq t} f(x_i) - f_t^{\mathrm{CP}}(x_t)$$

  falls below a pre-specified threshold ϵ.

In a Nutshell

- Cutting Plane Methods work by forming the piecewise linear lower bound

$$f(x) \geq f_t^{\mathrm{CP}}(x) := \max_{1 \leq i \leq t} \{f(x_{i-1}) + \langle x - x_{i-1}, s_i \rangle\}.$$

  where s_i denote gradients $\nabla f(x_{i-1})$.
- At iteration t the set $\{x_i\}_{i=0}^{t-1}$ is augmented by

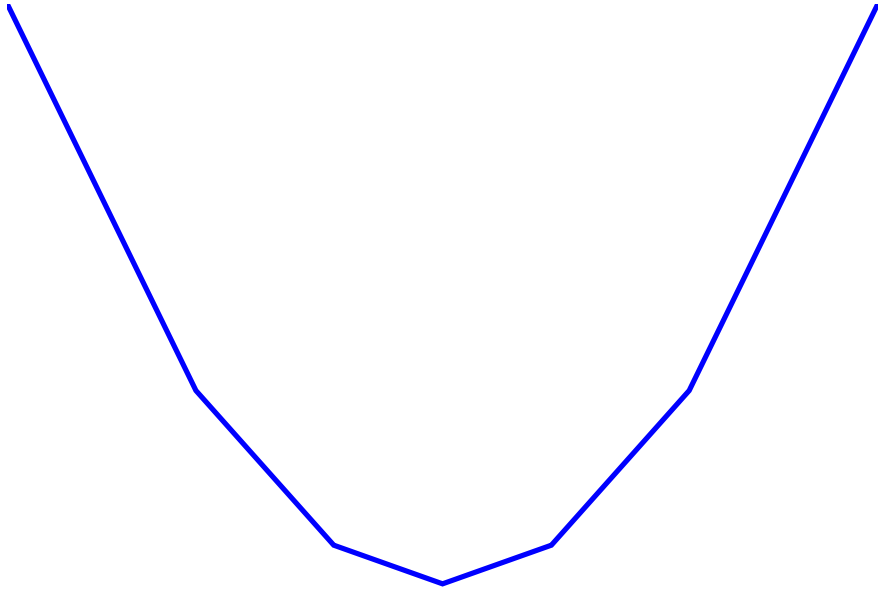
$$x_t := \operatorname*{argmin}_x f_t^{\mathrm{CP}}(x).$$

- Stop when the duality gap

$$\epsilon_t := \min_{0 \leq i \leq t} f(x_i) - f_t^{\mathrm{CP}}(x_t)$$

  falls below a pre-specified threshold ϵ.

What if the Function is NonSmooth?

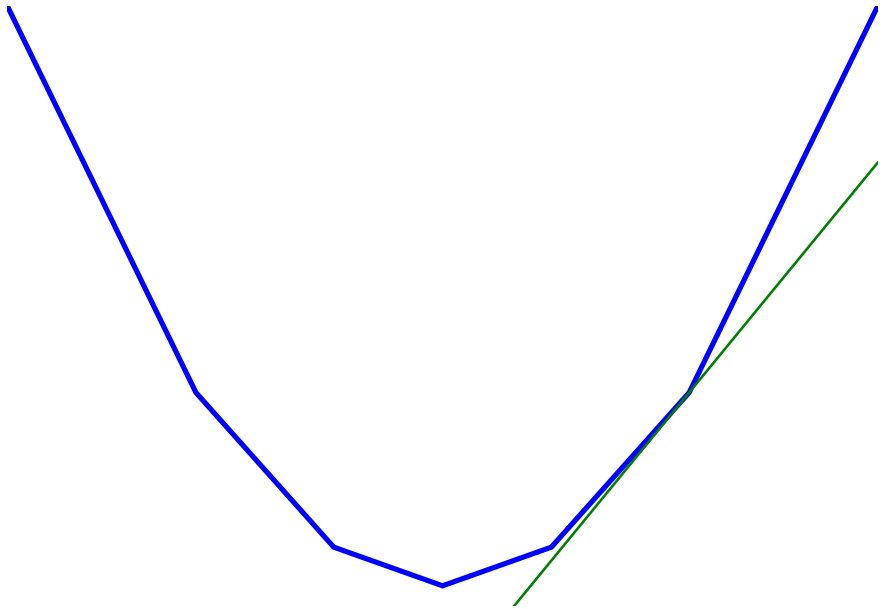


The piecewise linear function

$$f(x) := \max_i \langle u_i, x \rangle$$

is convex but not differentiable at the kinks!

Subgradients to the Rescue

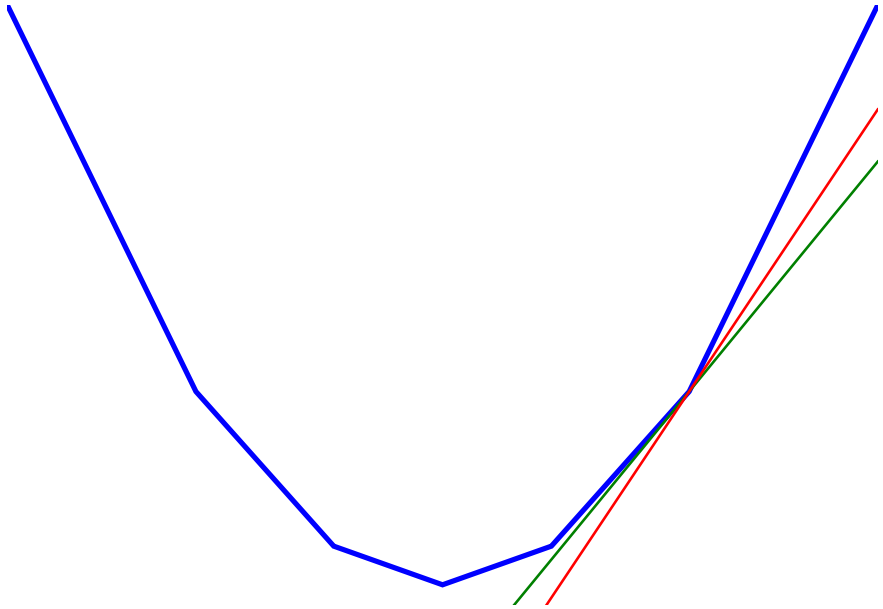


A subgradient at y is any vector μ which satisfies

$$f(x) \geq f(y) + \langle x - y, \mu \rangle \text{ for all } x$$

Set of all subgradients is denoted as $\partial f(y)$

Subgradients to the Rescue

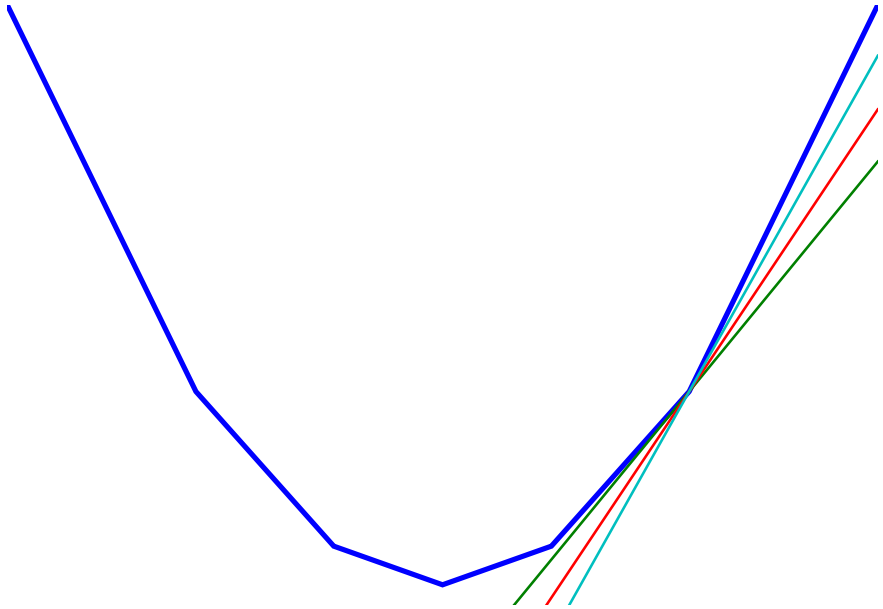


A subgradient at y is any vector μ which satisfies

$$f(x) \geq f(y) + \langle x - y, \mu \rangle \text{ for all } x$$

Set of all subgradients is denoted as $\partial f(y)$

Subgradients to the Rescue



A subgradient at y is any vector μ which satisfies

$$f(x) \geq f(y) + \langle x - y, \mu \rangle \text{ for all } x$$

Set of all subgradients is denoted as $\partial f(y)$

Good News!

Cutting Plane Methods work with subgradients

Just choose an arbitrary one

Boosting as an Optimization Problem

- Minimizing the maximum edge

$$\min_{\mathbf{d} \in \mathcal{S}^N} \underbrace{\max_{\mathbf{u} \in \mathcal{U}} \langle \mathbf{u}, \mathbf{d} \rangle}_{f(\mathbf{d})}$$

  is a convex optimization problem.

- Subgradient: $\partial f(\mathbf{d}) = \operatorname{argmax}_{\mathbf{u} \in \mathcal{U}} \langle \mathbf{u}, \mathbf{d} \rangle$

Computing Subgradient of $f(\mathbf{d})$ = Strong Oracle!

Boosting as an Optimization Problem

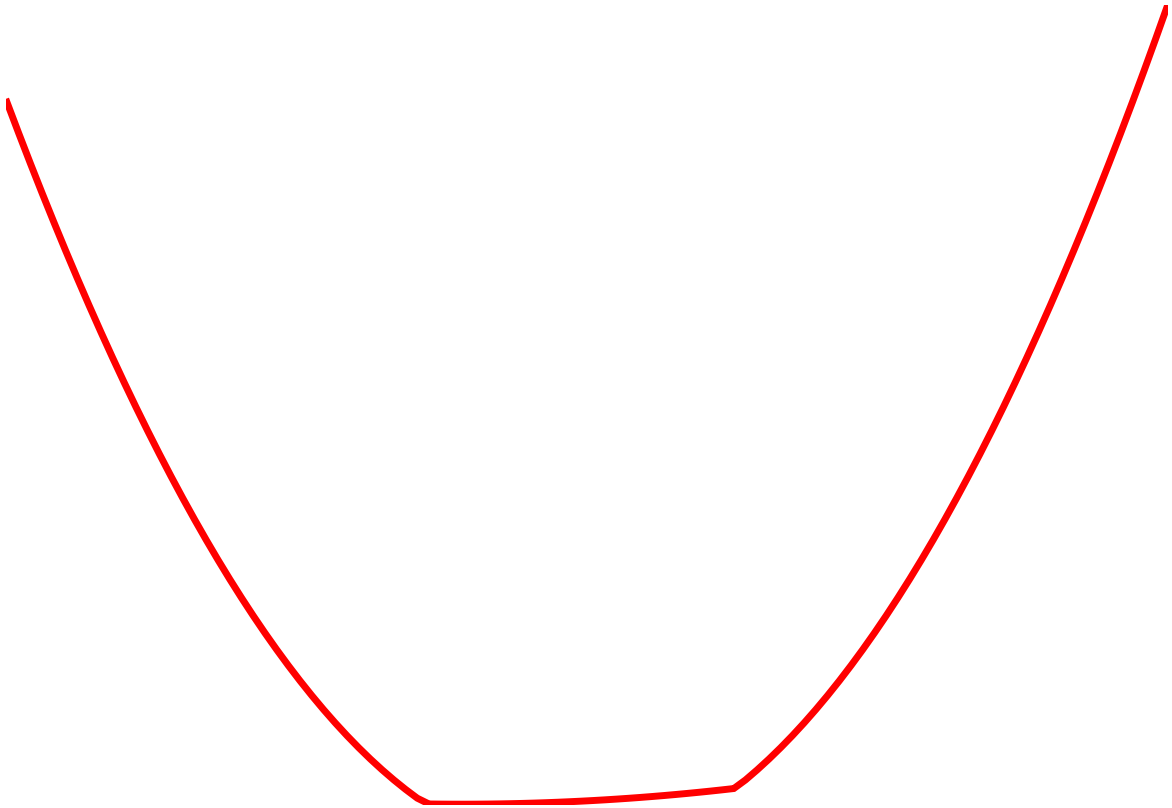
- Minimizing the maximum edge

$$\min_{\mathbf{d}\in\mathcal{S}^N} \underbrace{\max_{\mathbf{u}\in\mathcal{U}} \langle \mathbf{u}, \mathbf{d}\rangle}_{f(\mathbf{d})}$$

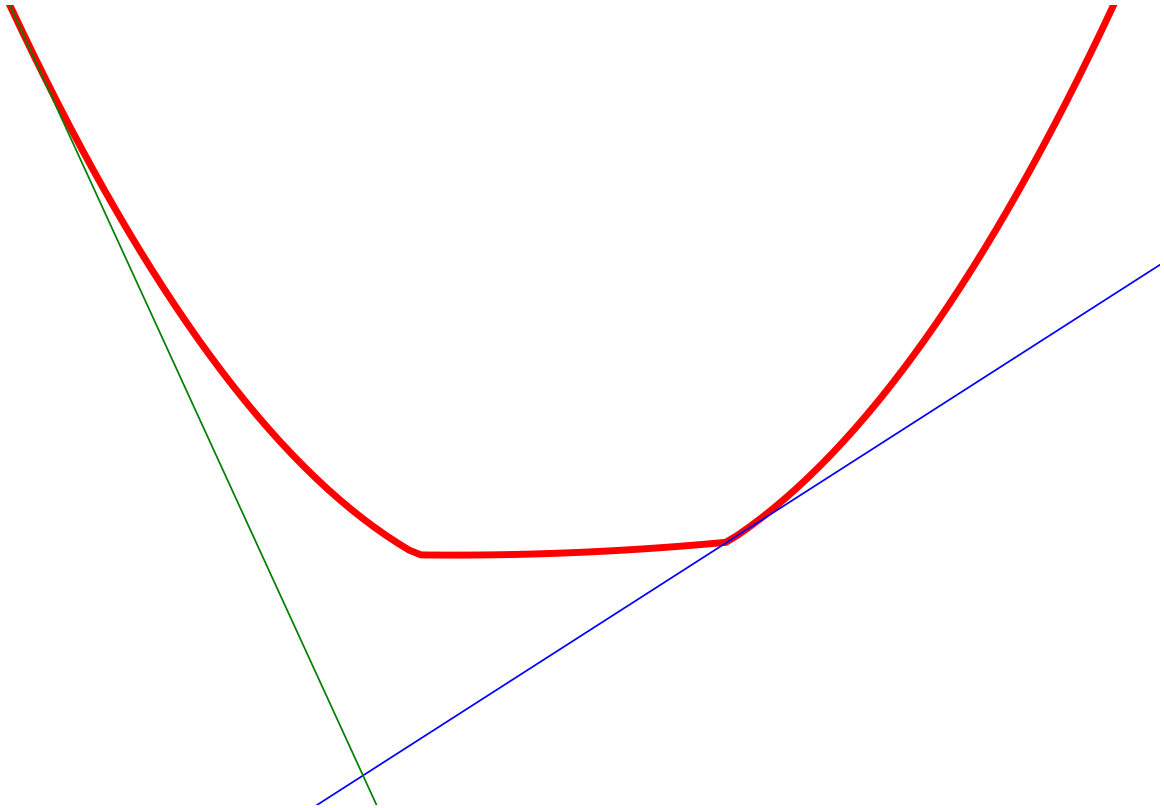
  is a convex optimization problem.
- Subgradient: $\partial f(\mathbf{d}) = \operatorname{argmax}_{\mathbf{u}\in\mathcal{U}} \langle \mathbf{u}, \mathbf{d}\rangle$

Computing Subgradient of $f(\mathbf{d})$ = Strong Oracle!

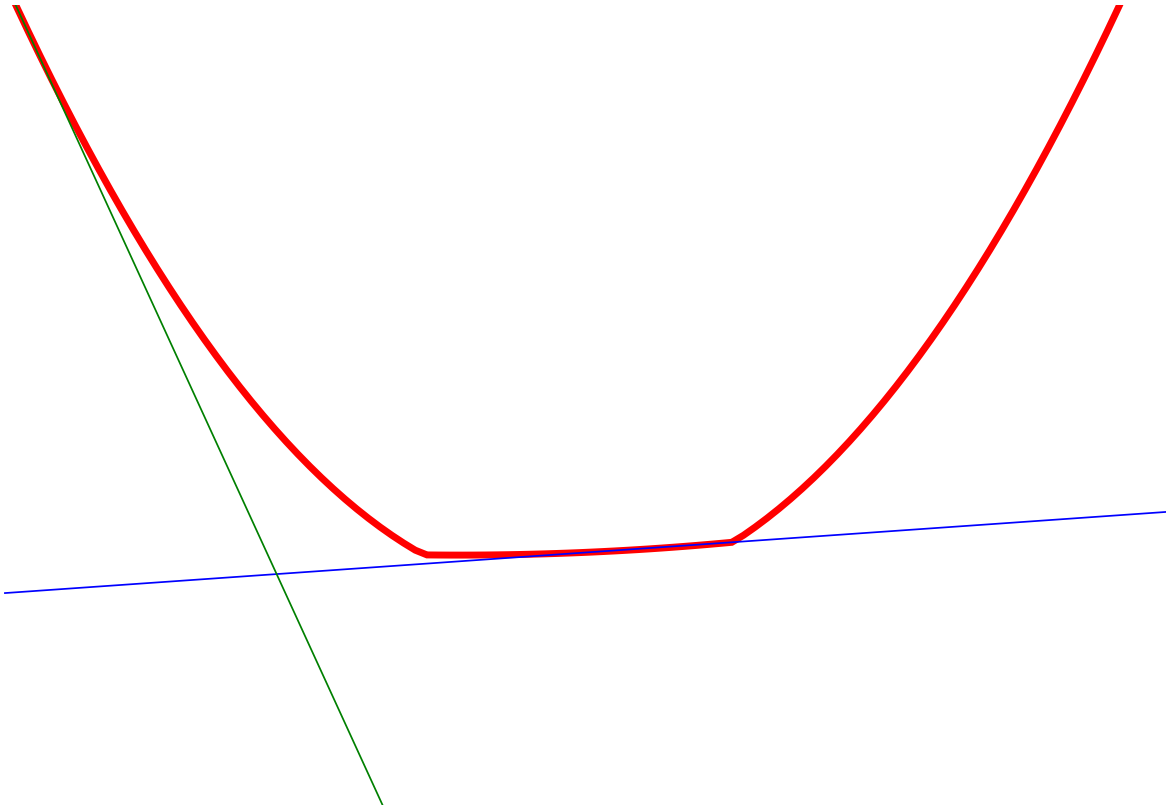
Subgradients and Stability



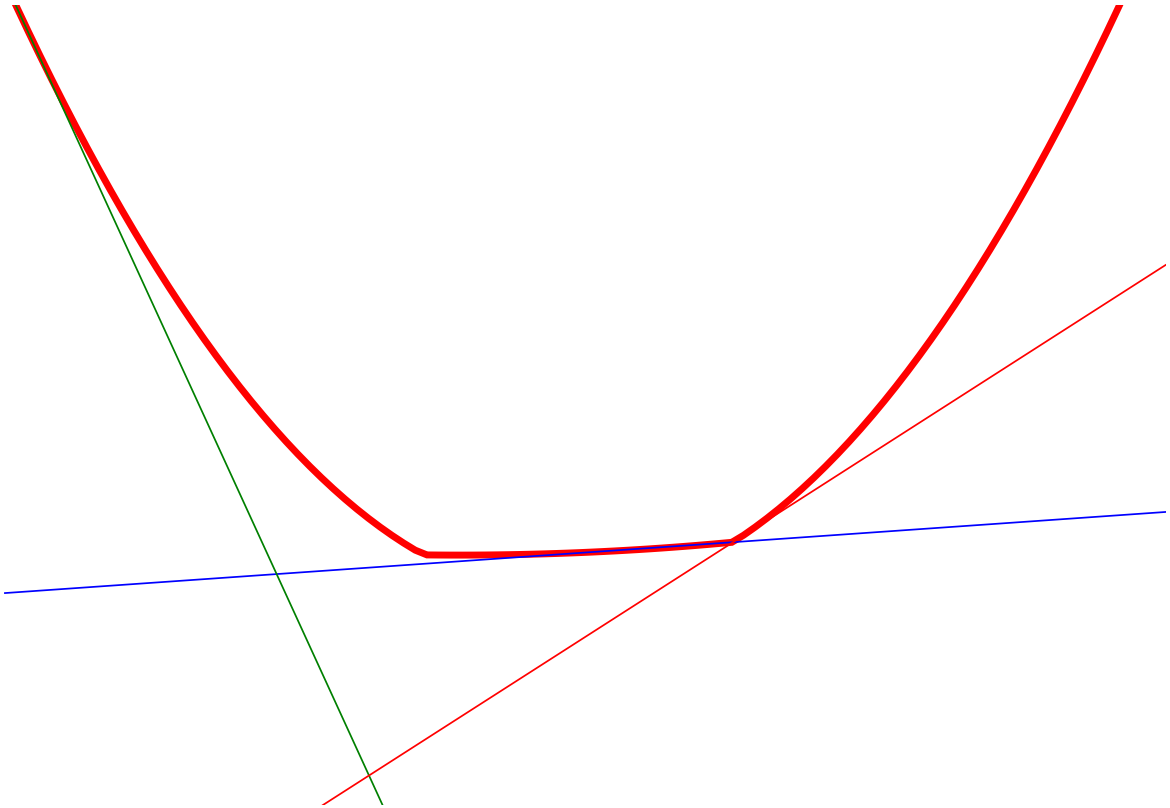
Subgradients and Stability



Subgradients and Stability

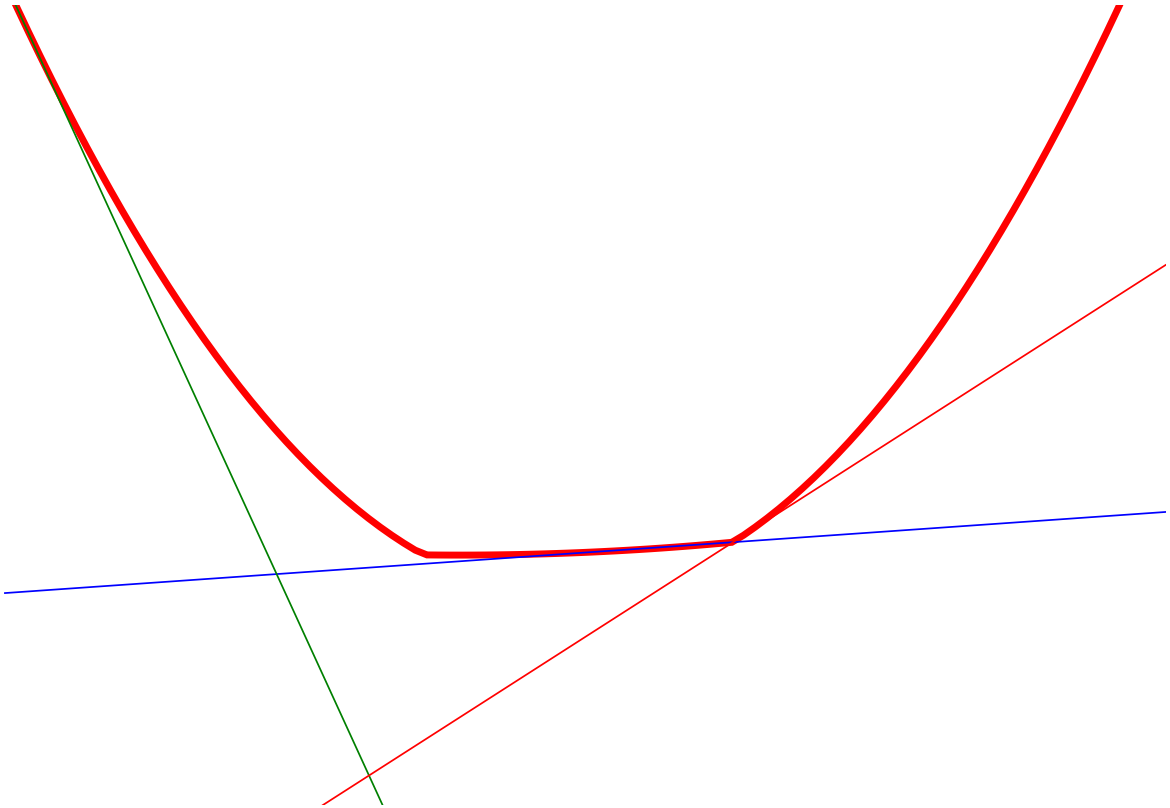


Subgradients and Stability



- Picking an arbitrary subgradient can cause stability issues

Subgradients and Stability



- Picking an arbitrary subgradient can cause stability issues

Back to Convex Analysis

Strong Convexity

A convex function f is strongly convex if, and only if, there exists a constant $\sigma > 0$ such that the function $f(x) - \frac{\sigma}{2}\|x\|^2$ is convex.

- If f is twice differentiable then the eigenvalues of the Hessian of f are lower bounded

$$\nabla^2 f(x) \succeq \sigma I.$$

- If f is strongly convex then

$$f(x') - f(x) - \langle x' - x, \mu \rangle \geq \frac{\sigma}{2}\|x' - x\|^2 \quad \forall x, x' \text{ and } \mu \in \partial f(x).$$

Back to Convex Analysis

Strong Convexity

A convex function f is strongly convex if, and only if, there exists a constant $\sigma > 0$ such that the function $f(x) - \frac{\sigma}{2}\|x\|^2$ is convex.

- If f is twice differentiable then the eigenvalues of the Hessian of f are lower bounded

$$\nabla^2 f(x) \succeq \sigma I.$$

- If f is strongly convex then

$$f(x') - f(x) - \langle x' - x, \mu \rangle \geq \frac{\sigma}{2}\|x' - x\|^2 \quad \forall x, x' \text{ and } \mu \in \partial f(x).$$

Back to Convex Analysis

Strong Convexity

A convex function f is strongly convex if, and only if, there exists a constant $\sigma > 0$ such that the function $f(x) - \frac{\sigma}{2}\|x\|^2$ is convex.

- If f is twice differentiable then the eigenvalues of the Hessian of f are lower bounded

$$\nabla^2 f(x) \succeq \sigma I.$$

- If f is strongly convex then

$$f(x') - f(x) - \langle x' - x, \mu \rangle \geq \frac{\sigma}{2}\|x' - x\|^2 \quad \forall x, x' \text{ and } \mu \in \partial f(x).$$

Bundle Methods [HL93]

- Are stabilized cutting plane methods
- At every iteration they form the model

$$f_t(x) := \Omega(x) + \max_{1 \le i \le t} \{ f(x_{i-1}) + \langle x - x_{i-1}, s_i \rangle \}.$$

  where Ω is an appropriately chosen strongly convex function, and s_i denotes a (sub)gradient of f evaluated at x_{i-1}.
- At iteration t the set $\{x_i\}_{i=0}^{t-1}$ is augmented by

$$x_t := \operatorname*{argmin}_x f_t(x).$$

- Stop when

$$\epsilon_t := \min_{0 \le i \le t} \Omega(x_i) + f(x_i) - f_t(x_t)$$

  falls below a pre-specified threshold ϵ.

Bundle Methods [HL93]

- Are stabilized cutting plane methods
- At every iteration they form the model

$$f_t(x) := \Omega(x) + \max_{1 \le i \le t} \{ f(x_{i-1}) + \langle x - x_{i-1}, s_i \rangle \}.$$

where Ω is an appropriately chosen strongly convex function, and s_i denotes a (sub)gradient of f evaluated at x_{i-1}.

- At iteration t the set $\{x_i\}_{i=0}^{t-1}$ is augmented by

$$x_t := \operatorname*{argmin}_x f_t(x).$$

- Stop when

$$\epsilon_t := \min_{0 \le i \le t} \Omega(x_i) + f(x_i) - f_t(x_t)$$

falls below a pre-specified threshold ϵ.

Bundle Methods [HL93]

- Are stabilized cutting plane methods
- At every iteration they form the model

$$f_t(x) := \Omega(x) + \max_{1 \le i \le t} \{ f(x_{i-1}) + \langle x - x_{i-1}, s_i \rangle \}.$$

  where Ω is an appropriately chosen strongly convex function, and s_i denotes a (sub)gradient of f evaluated at x_{i-1}.
- At iteration t the set $\{x_i\}_{i=0}^{t-1}$ is augmented by

$$x_t := \operatorname*{argmin}_x f_t(x).$$

- Stop when

$$\epsilon_t := \min_{0 \le i \le t} \Omega(x_i) + f(x_i) - f_t(x_t)$$

  falls below a pre-specified threshold ϵ.

Bundle Methods [HL93]

- Are stabilized cutting plane methods
- At every iteration they form the model

$$f_t(x) := \Omega(x) + \max_{1 \le i \le t} \{ f(x_{i-1}) + \langle x - x_{i-1}, s_i \rangle \}.$$

where Ω is an appropriately chosen strongly convex function, and s_i denotes a (sub)gradient of f evaluated at x_{i-1}.

- At iteration t the set $\{x_i\}_{i=0}^{t-1}$ is augmented by

$$x_t := \operatorname*{argmin}_x f_t(x).$$

- Stop when

$$\epsilon_t := \min_{0 \le i \le t} \Omega(x_i) + f(x_i) - f_t(x_t)$$

falls below a pre-specified threshold ϵ.

Rates of Convergence [SVL08]

Nonsmooth Functions

- The number of iterations to reach ϵ precision is bounded by

$$t \leq \frac{c_1}{\epsilon \cdot \sigma} + c_2$$

where c_1 and c_2 are problem dependent constants, and σ is the modulus of strong convexity of Ω.

Proving Iteration Bounds for Boosting

Lemma

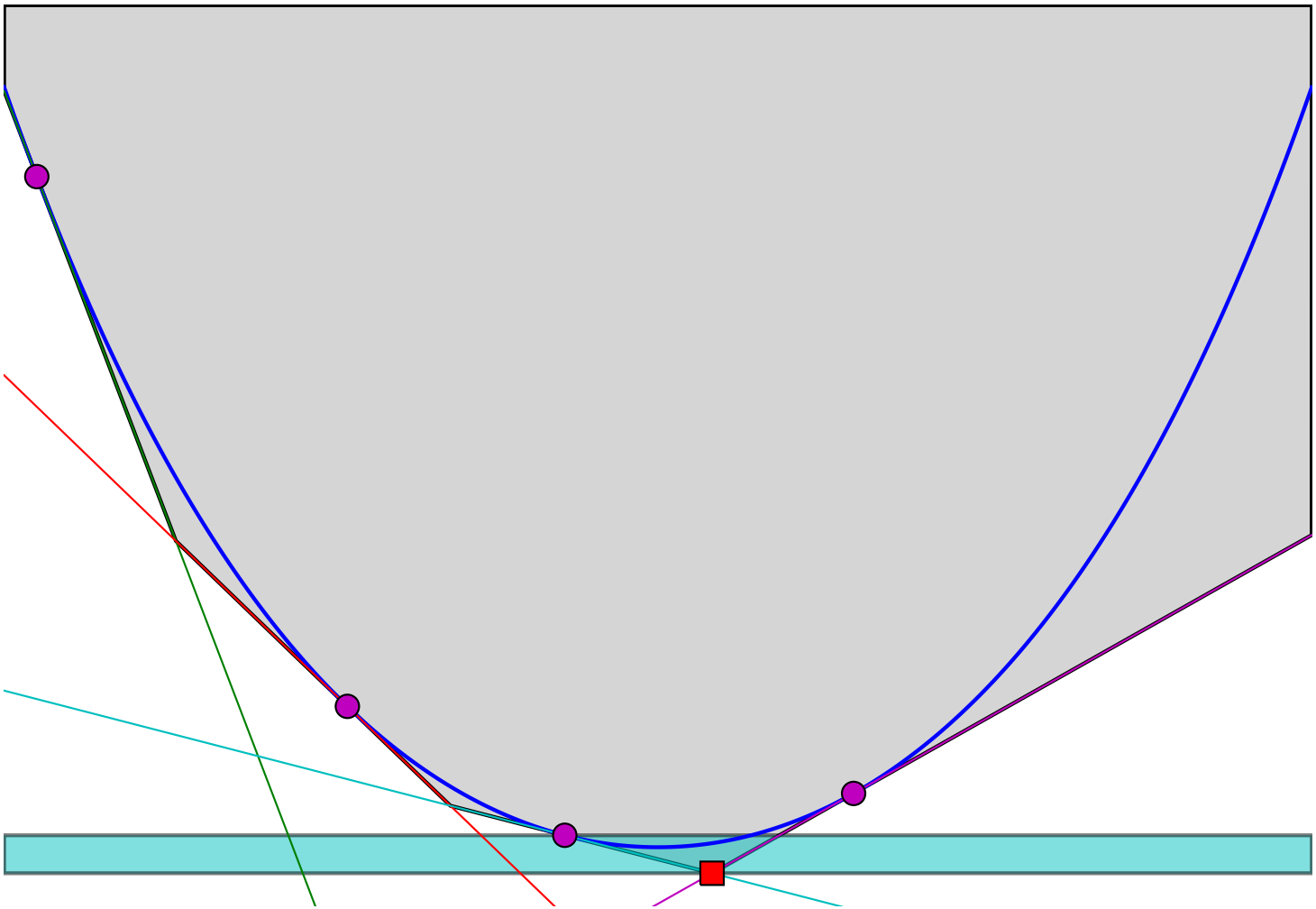
Suppose $0 \leq \Omega(\mathbf{d}) \leq \epsilon/2$ *for all* $\mathbf{d}$, *and let*

$$\min_{0 \leq i \leq t} \underbrace{\Omega(\mathbf{d}_i)}_{\leq \epsilon/2} + f(\mathbf{d}_i) - \underbrace{f_t(\mathbf{d}_t)}_{\leq f(\mathbf{d}^*)} \leq \epsilon/2.$$

Then

$$f(\mathbf{d}_t) \leq f(\mathbf{d}^*) + \epsilon$$

Recall



Proving Iteration Bounds Contd.

Entropy as a Regularizer

Suppose we set $\Omega(\mathbf{d}) = \frac{\epsilon}{2 \log N} \sum_{i=1}^{N} d_i \log d_i$ then

- $\Omega(\mathbf{d}) \leq \epsilon/2$ for all $\mathbf{d}$
- Ω is strongly convex with modulus $\frac{\epsilon}{2 \log N}$.

- Plugging this into rates of convergence of bundle methods shows that we need

$$t \leq 2 \frac{c_1 \log N}{\epsilon^2} + c_2$$

iterations to obtain an ϵ accurate solution [SSS08].

- Similar iteration bounds for
  - strong oracle: returns $\operatorname{argmax}_{\mathbf{u} \in \mathcal{U}} \langle \mathbf{u}, \mathbf{d} \rangle$
  - weak oracle: returns an $\mathbf{u}$ such that $\langle \mathbf{u}, \mathbf{d} \rangle \geq g$ [WGV08]

Proving Iteration Bounds Contd.

Entropy as a Regularizer

Suppose we set $\Omega(\mathbf{d}) = \frac{\epsilon}{2 \log N} \sum_{i=1}^{N} d_i \log d_i$ then

- $\Omega(\mathbf{d}) \leq \epsilon/2$ for all $\mathbf{d}$
- Ω is strongly convex with modulus $\frac{\epsilon}{2 \log N}$.

- Plugging this into rates of convergence of bundle methods shows that we need

$$t \leq 2 \frac{c_1 \log N}{\epsilon^2} + c_2$$

iterations to obtain an ϵ accurate solution [SSS08].

- Similar iteration bounds for
  - strong oracle: returns $\operatorname{argmax}_{\mathbf{u} \in \mathcal{U}} \langle \mathbf{u}, \mathbf{d} \rangle$
  - weak oracle: returns an $\mathbf{u}$ such that $\langle \mathbf{u}, \mathbf{d} \rangle \geq g$ [WGV08]

Proving Iteration Bounds Contd.

Entropy as a Regularizer

Suppose we set $\Omega(\mathbf{d}) = \frac{\epsilon}{2 \log N} \sum_{i=1}^{N} d_i \log d_i$ then

- $\Omega(\mathbf{d}) \leq \epsilon/2$ for all $\mathbf{d}$
- Ω is strongly convex with modulus $\frac{\epsilon}{2 \log N}$.

- Plugging this into rates of convergence of bundle methods shows that we need

$$t \leq 2 \frac{c_1 \log N}{\epsilon^2} + c_2$$

iterations to obtain an ϵ accurate solution [SSS08].

- Similar iteration bounds for
  - strong oracle: returns $\operatorname{argmax}_{\mathbf{u} \in \mathcal{U}} \langle \mathbf{u}, \mathbf{d} \rangle$
  - weak oracle: returns an $\mathbf{u}$ such that $\langle \mathbf{u}, \mathbf{d} \rangle \geq g$ [WGV08]

Towards practical algorithms for Boosting

Primal Problem

$$\min_{\substack{\mathbf{d}\cdot\mathbf{1}=1 \\ \mathbf{d}\leq\frac{1}{\nu}\mathbf{1}}} \underbrace{\frac{1}{\eta}\Delta(\mathbf{d},\mathbf{d}^0)+\max_{q=1,2,\ldots,t}\langle\mathbf{u}^q,\mathbf{d}\rangle}_{:=P^t(\mathbf{d})}.$$

Dual Problem

$$\max_{\substack{\mathbf{w}\geq\mathbf{0} \\ \langle\mathbf{w},\mathbf{e}\rangle=1 \\ \psi\geq 0}} -\frac{1}{\eta}\log\sum_{n=1}^{N} d_n^0\exp(-\eta\sum_{q=1}^{t}u_n^q w_q-\eta\psi_n)-\frac{1}{\nu}\sum_{n=1}^{N}\psi_n$$

Towards practical algorithms for Boosting

Primal Problem

$$\min_{\substack{\mathbf{d}\cdot\mathbf{1}=1 \\ \mathbf{d}\leq\frac{1}{\nu}\mathbf{1}}} \underbrace{\frac{1}{\eta}\Delta(\mathbf{d},\mathbf{d}^0)+\max_{q=1,2,\ldots,t}\langle\mathbf{u}^q,\mathbf{d}\rangle}_{:=P^t(\mathbf{d})}.$$

Dual Problem

$$\max_{\substack{\mathbf{w}\geq\mathbf{0} \\ \langle\mathbf{w},\mathbf{e}\rangle=1 \\ \boldsymbol{\psi}\geq 0}} -\frac{1}{\eta}\log\sum_{n=1}^{N} d_n^0\exp(-\eta\sum_{q=1}^{t}u_n^q w_q-\eta\psi_n)-\frac{1}{\nu}\sum_{n=1}^{N}\psi_n$$

Gradient Descent

Unconstrained

- Suppose you want to minimize

$$\min_{\mathbf{w} \in \mathbf{R}^n} f(\mathbf{w})$$

- Gradient descent produces a sequence of iterates

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla f(\mathbf{w}_t)$$

- The step-size η found by
  - Solving $\min_\eta f(\mathbf{w}_t - \eta \nabla f(\mathbf{w}_t))$ or
  - Decay schedule such as $\eta_t = \frac{1}{t}$

Gradient Descent

Unconstrained

- Suppose you want to minimize

$$\min_{\mathbf{w} \in \mathbf{R}^n} f(\mathbf{w})$$

- Gradient descent produces a sequence of iterates

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla f(\mathbf{w}_t)$$

- The step-size η found by
  - Solving $\min_\eta f(\mathbf{w}_t - \eta \nabla f(\mathbf{w}_t))$ or
  - Decay schedule such as $\eta_t = \frac{1}{t}$

Gradient Descent

Unconstrained

- Suppose you want to minimize

$$\min_{\mathbf{w} \in \mathbf{R}^n} f(\mathbf{w})$$

- Gradient descent produces a sequence of iterates

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla f(\mathbf{w}_t)$$

- The step-size η found by
  - Solving $\min_\eta f(\mathbf{w}_t - \eta \nabla f(\mathbf{w}_t))$ or
  - Decay schedule such as $\eta_t = \frac{1}{t}$

Projected Gradient Descent

Constrained

- Suppose you want to minimize

$$\min_{\mathbf{w}\in\Gamma} f(\mathbf{w})$$

- Projected Gradient descent produces a sequence of iterates

$$\mathbf{w}_{t+1} = P_\Gamma(\mathbf{w}_t - \eta\nabla f(\mathbf{w}))$$

where $P_\Gamma(\mathbf{z}) := \operatorname{argmin}_{x\in\Omega} \|\mathbf{z} - x\|^2$.

Projected Gradient Descent

Constrained

- Suppose you want to minimize

$$\min_{\mathbf{w}\in\Gamma} f(\mathbf{w})$$

- Projected Gradient descent produces a sequence of iterates

$$\mathbf{w}_{t+1} = P_\Gamma(\mathbf{w}_t - \eta\nabla f(\mathbf{w}))$$

where $P_\Gamma(\mathbf{z}) := \operatorname{argmin}_{x\in\Omega} \|\mathbf{z} - x\|^2$.

Spectral Projected Gradient Method [BMR00]

- **Step 1:** Detect if current point is stationary
- **Step 2:** Backtracking
  - Step 2.1: Compute $\mathbf{d}_t = P_\Omega(\mathbf{w}_t - \alpha_t \nabla f(\mathbf{w})) - \mathbf{w}_t$. Set $\lambda = 1$
  - Step 2.2: Set $\mathbf{w}_+ = \mathbf{w}_t + \lambda \mathbf{d}_t$
  - Step 2.3: If $f(\mathbf{w}_+) \leq \max_{0 \leq j \leq M} f(\mathbf{w}_{t-j}) + \gamma \lambda \langle \mathbf{d}_t, \nabla f(\mathbf{w}_t) \rangle$, $\mathbf{s}_t = \mathbf{w}_+ - \mathbf{w}_t$, $\mathbf{w}_{t+1} = \mathbf{w}_+$, $\mathbf{y}_t = \nabla f(\mathbf{w}_{t+1}) - \nabla f(\mathbf{w}_t)$
  - Else define $\lambda = [\eta_1 \lambda, \eta_2 \lambda]$ and go to Step 2.2
- **Step 3:** Compute $b_t = \langle \mathbf{s}_t, \mathbf{y}_t \rangle$.
  - If $b_t \leq 0$ set $\alpha_{t+1} = \alpha_{\max}$
  - Else compute $a_t = \langle \mathbf{y}_t, \mathbf{y}_t \rangle$ and set $\alpha_{t+1} = \min\{\alpha_{\max}, \max\{\alpha_{\min}, a_t / b_t\}\}$

Spectral Projected Gradient Method [BMR00]

- **Step 1:** Detect if current point is stationary
- **Step 2:** Backtracking
  - **Step 2.1:** Compute $\mathbf{d}_t = P_\Omega(\mathbf{w}_t - \alpha_t \nabla f(\mathbf{w})) - \mathbf{w}_t$. Set $\lambda = 1$
  - **Step 2.2:** Set $\mathbf{w}_+ = \mathbf{w}_t + \lambda \mathbf{d}_t$
  - **Step 2.3:** If $f(\mathbf{w}_+) \leq \max_{0 \leq j \leq M} f(\mathbf{w}_{t-j}) + \gamma \lambda \langle \mathbf{d}_t, \nabla f(\mathbf{w}_{t-j}) \rangle$
    $\lambda_t = \lambda$, $\mathbf{w}_{t+1} = \mathbf{w}_+$, $\mathbf{s}_k = \mathbf{w}_{t+1} - \mathbf{w}_t$,
    $\mathbf{y}_t = \nabla f(\mathbf{w}_{t+1}) - \nabla f(\mathbf{w}_t)$
  - Else define $\lambda \in [\sigma_1 \lambda, \sigma_2 \lambda]$ and go to Step 2.2
- **Step 3:** Compute $b_t = \langle \mathbf{s}_t, \mathbf{y}_t \rangle$.
  If $b_t \leq 0$ set $\alpha_{t+1} = \alpha_{\max}$
  Else compute $a_t = \langle \mathbf{y}_t, \mathbf{y}_t \rangle$ and set
  $\alpha_{t+1} = \min\{\alpha_{\max}, \max\{\alpha_{\min}, a_t / b_t\}\}$

Spectral Projected Gradient Method [BMR00]

- **Step 1:** Detect if current point is stationary
- **Step 2:** Backtracking
  - **Step 2.1:** Compute $\mathbf{d}_t = P_\Omega(\mathbf{w}_t - \alpha_t \nabla f(\mathbf{w})) - \mathbf{w}_t$. Set $\lambda = 1$
  - **Step 2.2:** Set $\mathbf{w}_+ = \mathbf{w}_t + \lambda \mathbf{d}_t$
  - **Step 2.3:** If $f(\mathbf{w}_+) \leq \max_{0 \leq j \leq M} f(\mathbf{w}_{t-j}) + \gamma \lambda \langle \mathbf{d}_t, \nabla f(\mathbf{w}_{t-j}) \rangle$ $\lambda_t = \lambda$, $\mathbf{w}_{t+1} = \mathbf{w}_+$, $\mathbf{s}_k = \mathbf{w}_{t+1} - \mathbf{w}_t$, $\mathbf{y}_t = \nabla f(\mathbf{w}_{t+1}) - \nabla f(\mathbf{w}_t)$
  - Else define $\lambda \in [\sigma_1 \lambda, \sigma_2 \lambda]$ and go to Step 2.2
- **Step 3:** Compute $b_t = \langle \mathbf{s}_t, \mathbf{y}_t \rangle$.
  - If $b_t \leq 0$ set $\alpha_{t+1} = \alpha_{\max}$
  - Else compute $a_t = \langle \mathbf{y}_t, \mathbf{y}_t \rangle$ and set $\alpha_{t+1} = \min\{\alpha_{\max}, \max\{\alpha_{\min}, a_t/b_t\}\}$

Spectral Projected Gradient Method [BMR00]

- **Step 1:** Detect if current point is stationary
- **Step 2:** Backtracking
  - **Step 2.1:** Compute $\mathbf{d}_t = P_\Omega(\mathbf{w}_t - \alpha_t \nabla f(\mathbf{w})) - \mathbf{w}_t$. Set $\lambda = 1$
  - **Step 2.2:** Set $\mathbf{w}_+ = \mathbf{w}_t + \lambda \mathbf{d}_t$
  - **Step 2.3:** If $f(\mathbf{w}_+) \leq \max_{0 \leq j \leq M} f(\mathbf{w}_{t-j}) + \gamma \lambda \langle \mathbf{d}_t, \nabla f(\mathbf{w}_{t-j}) \rangle$
    $\lambda_t = \lambda$, $\mathbf{w}_{t+1} = \mathbf{w}_+$, $\mathbf{s}_k = \mathbf{w}_{t+1} - \mathbf{w}_t$,
    $\mathbf{y}_t = \nabla f(\mathbf{w}_{t+1}) - \nabla f(\mathbf{w}_t)$
  - Else define $\lambda \in [\sigma_1 \lambda, \sigma_2 \lambda]$ and go to Step 2.2
- **Step 3:** Compute $b_t = \langle \mathbf{s}_t, \mathbf{y}_t \rangle$.
  - If $b_t \leq 0$ set $\alpha_{t+1} = \alpha_{\max}$
  - Else compute $a_t = \langle \mathbf{y}_t, \mathbf{y}_t \rangle$ and set
    $\alpha_{t+1} = \min\{\alpha_{\max}, \max\{\alpha_{\min}, a_t / b_t\}\}$

Spectral Projected Gradient Method [BMR00]

- **Step 1:** Detect if current point is stationary
- **Step 2:** Backtracking
  - **Step 2.1:** Compute $\mathbf{d}_t = P_\Omega(\mathbf{w}_t - \alpha_t \nabla f(\mathbf{w})) - \mathbf{w}_t$. Set $\lambda = 1$
  - **Step 2.2:** Set $\mathbf{w}_+ = \mathbf{w}_t + \lambda \mathbf{d}_t$
  - **Step 2.3:** If $f(\mathbf{w}_+) \leq \max_{0 \leq j \leq M} f(\mathbf{w}_{t-j}) + \gamma\lambda \langle \mathbf{d}_t, \nabla f(\mathbf{w}_{t-j}) \rangle$
    $\lambda_t = \lambda$, $\mathbf{w}_{t+1} = \mathbf{w}_+$, $\mathbf{s}_k = \mathbf{w}_{t+1} - \mathbf{w}_t$,
    $\mathbf{y}_t = \nabla f(\mathbf{w}_{t+1}) - \nabla f(\mathbf{w}_t)$
  - Else define $\lambda \in [\sigma_1 \lambda, \sigma_2 \lambda]$ and go to Step 2.2
- **Step 3:** Compute $b_t = \langle \mathbf{s}_t, \mathbf{y}_t \rangle$.
  - If $b_t \leq 0$ set $\alpha_{t+1} = \alpha_{\max}$
  - Else compute $a_t = \langle \mathbf{y}_t, \mathbf{y}_t \rangle$ and set
    $\alpha_{t+1} = \min\{\alpha_{\max}, \max\{\alpha_{\min}, a_t/b_t\}\}$

Spectral Projected Gradient Method [BMR00]

- **Step 1:** Detect if current point is stationary
- **Step 2:** Backtracking
  - **Step 2.1:** Compute $\mathbf{d}_t = P_\Omega(\mathbf{w}_t - \alpha_t \nabla f(\mathbf{w})) - \mathbf{w}_t$. Set $\lambda = 1$
  - **Step 2.2:** Set $\mathbf{w}_+ = \mathbf{w}_t + \lambda \mathbf{d}_t$
  - **Step 2.3:** If $f(\mathbf{w}_+) \leq \max_{0 \leq j \leq M} f(\mathbf{w}_{t-j}) + \gamma\lambda \langle \mathbf{d}_t, \nabla f(\mathbf{w}_{t-j}) \rangle$
    $\lambda_t = \lambda$, $\mathbf{w}_{t+1} = \mathbf{w}_+$, $\mathbf{s}_k = \mathbf{w}_{t+1} - \mathbf{w}_t$,
    $\mathbf{y}_t = \nabla f(\mathbf{w}_{t+1}) - \nabla f(\mathbf{w}_t)$
  - Else define $\lambda \in [\sigma_1 \lambda, \sigma_2 \lambda]$ and go to Step 2.2
- **Step 3:** Compute $b_t = \langle \mathbf{s}_t, \mathbf{y}_t \rangle$.
  - If $b_t \leq 0$ set $\alpha_{t+1} = \alpha_{\max}$
  - Else compute $a_t = \langle \mathbf{y}_t, \mathbf{y}_t \rangle$, and set
    $\alpha_{t+1} = \min\{\alpha_{\max}, \max\{\alpha_{\min}, a_t/b_t\}\}$

Spectral Projected Gradient Method [BMR00]

- **Step 1:** Detect if current point is stationary
- **Step 2:** Backtracking
  - **Step 2.1:** Compute $\mathbf{d}_t = P_\Omega(\mathbf{w}_t - \alpha_t \nabla f(\mathbf{w})) - \mathbf{w}_t$. Set $\lambda = 1$
  - **Step 2.2:** Set $\mathbf{w}_+ = \mathbf{w}_t + \lambda \mathbf{d}_t$
  - **Step 2.3:** If $f(\mathbf{w}_+) \leq \max_{0 \leq j \leq M} f(\mathbf{w}_{t-j}) + \gamma\lambda \langle \mathbf{d}_t, \nabla f(\mathbf{w}_{t-j}) \rangle$
    $\lambda_t = \lambda$, $\mathbf{w}_{t+1} = \mathbf{w}_+$, $\mathbf{s}_k = \mathbf{w}_{t+1} - \mathbf{w}_t$,
    $\mathbf{y}_t = \nabla f(\mathbf{w}_{t+1}) - \nabla f(\mathbf{w}_t)$
  - Else define $\lambda \in [\sigma_1 \lambda, \sigma_2 \lambda]$ and go to Step 2.2
- **Step 3:** Compute $b_t = \langle \mathbf{s}_t, \mathbf{y}_t \rangle$.
  - If $b_t \leq 0$ set $\alpha_{t+1} = \alpha_{\max}$
  - Else compute $a_t = \langle \mathbf{y}_t, \mathbf{y}_t \rangle$ and set
    $\alpha_{t+1} = \min\{\alpha_{\max}, \max\{\alpha_{\min}, a_t / b_t\}\}$

Spectral Projected Gradient Method [BMR00]

- **Step 1:** Detect if current point is stationary
- **Step 2:** Backtracking
  - **Step 2.1:** Compute $\mathbf{d}_t = P_\Omega(\mathbf{w}_t - \alpha_t \nabla f(\mathbf{w})) - \mathbf{w}_t$. Set $\lambda = 1$
  - **Step 2.2:** Set $\mathbf{w}_+ = \mathbf{w}_t + \lambda \mathbf{d}_t$
  - **Step 2.3:** If $f(\mathbf{w}_+) \leq \max_{0 \leq j \leq M} f(\mathbf{w}_{t-j}) + \gamma\lambda \langle \mathbf{d}_t, \nabla f(\mathbf{w}_{t-j}) \rangle$
    $\lambda_t = \lambda$, $\mathbf{w}_{t+1} = \mathbf{w}_+$, $\mathbf{s}_k = \mathbf{w}_{t+1} - \mathbf{w}_t$,
    $\mathbf{y}_t = \nabla f(\mathbf{w}_{t+1}) - \nabla f(\mathbf{w}_t)$
  - Else define $\lambda \in [\sigma_1 \lambda, \sigma_2 \lambda]$ and go to Step 2.2
- **Step 3:** Compute $b_t = \langle \mathbf{s}_t, \mathbf{y}_t \rangle$.
  - If $b_t \leq 0$ set $\alpha_{t+1} = \alpha_{\max}$
  - Else compute $a_t = \langle \mathbf{y}_t, \mathbf{y}_t \rangle$ and set
    $\alpha_{t+1} = \min\{\alpha_{\max}, \max\{\alpha_{\min}, a_t / b_t\}\}$

Spectral Projected Gradient Method [BMR00]

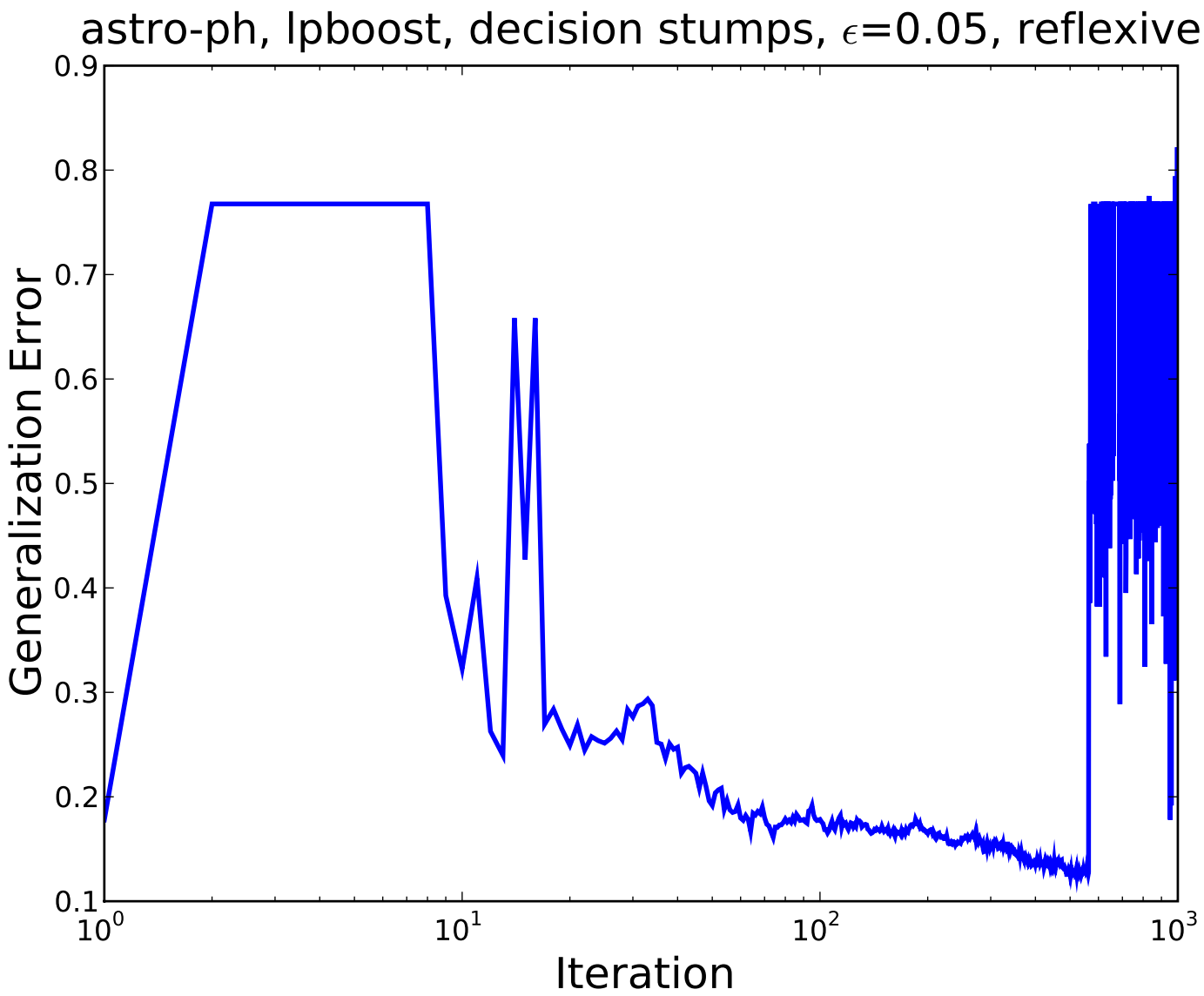
- **Step 1:** Detect if current point is stationary
- **Step 2:** Backtracking
  - **Step 2.1:** Compute $\mathbf{d}_t = P_\Omega(\mathbf{w}_t - \alpha_t \nabla f(\mathbf{w})) - \mathbf{w}_t$. Set $\lambda = 1$
  - **Step 2.2:** Set $\mathbf{w}_+ = \mathbf{w}_t + \lambda \mathbf{d}_t$
  - **Step 2.3:** If $f(\mathbf{w}_+) \leq \max_{0 \leq j \leq M} f(\mathbf{w}_{t-j}) + \gamma\lambda \langle \mathbf{d}_t, \nabla f(\mathbf{w}_{t-j}) \rangle$
    $\lambda_t = \lambda$, $\mathbf{w}_{t+1} = \mathbf{w}_+$, $\mathbf{s}_k = \mathbf{w}_{t+1} - \mathbf{w}_t$,
    $\mathbf{y}_t = \nabla f(\mathbf{w}_{t+1}) - \nabla f(\mathbf{w}_t)$
  - Else define $\lambda \in [\sigma_1 \lambda, \sigma_2 \lambda]$ and go to Step 2.2
- **Step 3:** Compute $b_t = \langle \mathbf{s}_t, \mathbf{y}_t \rangle$.
  - If $b_t \leq 0$ set $\alpha_{t+1} = \alpha_{\max}$
  - Else compute $a_t = \langle \mathbf{y}_t, \mathbf{y}_t \rangle$ and set
    $\alpha_{t+1} = \min\{\alpha_{\max}, \max\{\alpha_{\min}, a_t / b_t\}\}$

Dataset Properties

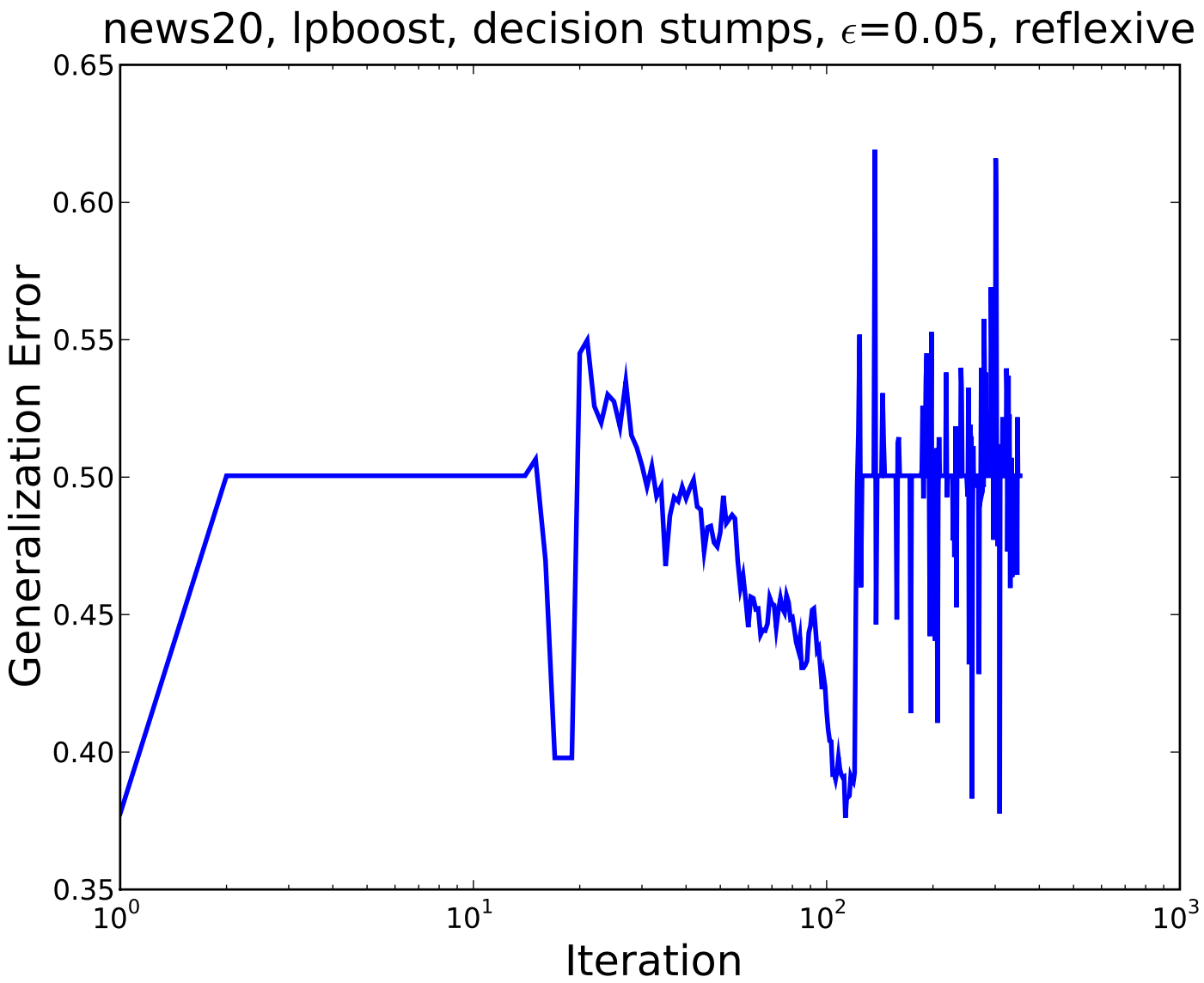
Name	Size	Dimension	Density
Astro-ph	94,856	99,757	0.008
News20	19,954	1,355,191	0.03
Real-Sim	72,201	20,958	0.25
rcv1	677,399	47,236	0.15

- Combined test and train. 60% randomly chosen for train, 20% for test and 20% for validation.
- Plot for generating the splits available.

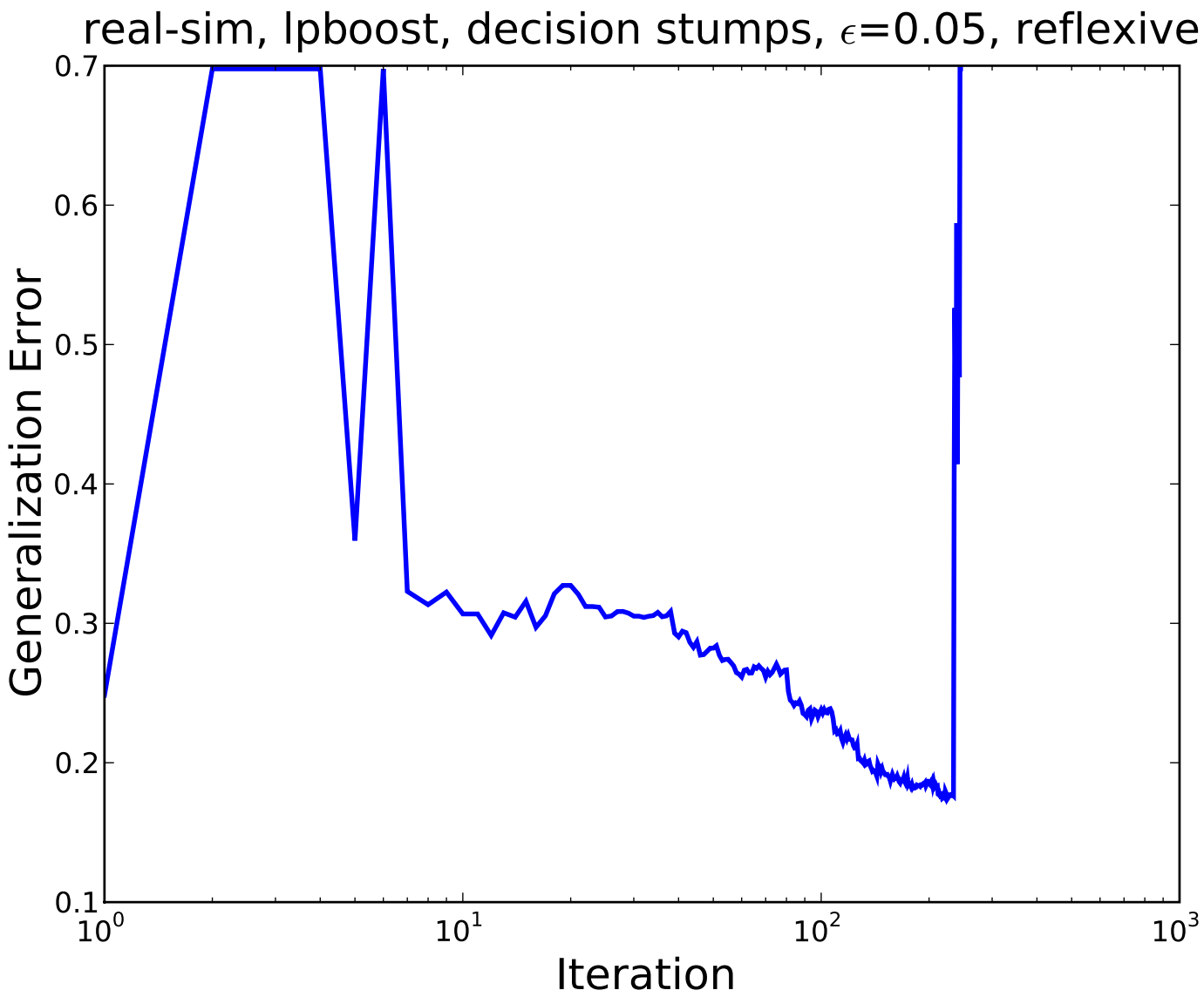
LPBoost is Brittle



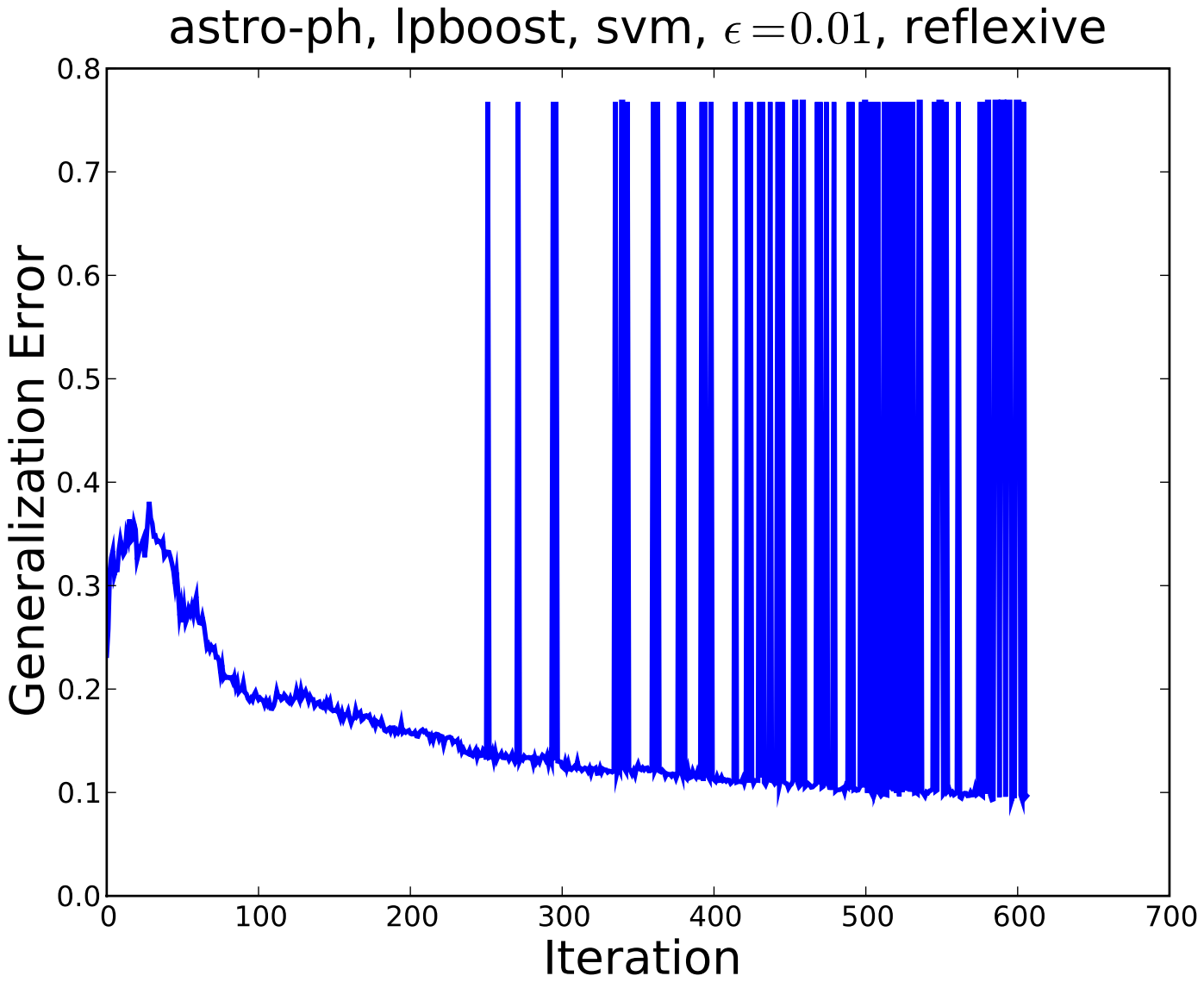
LPBoost is Brittle



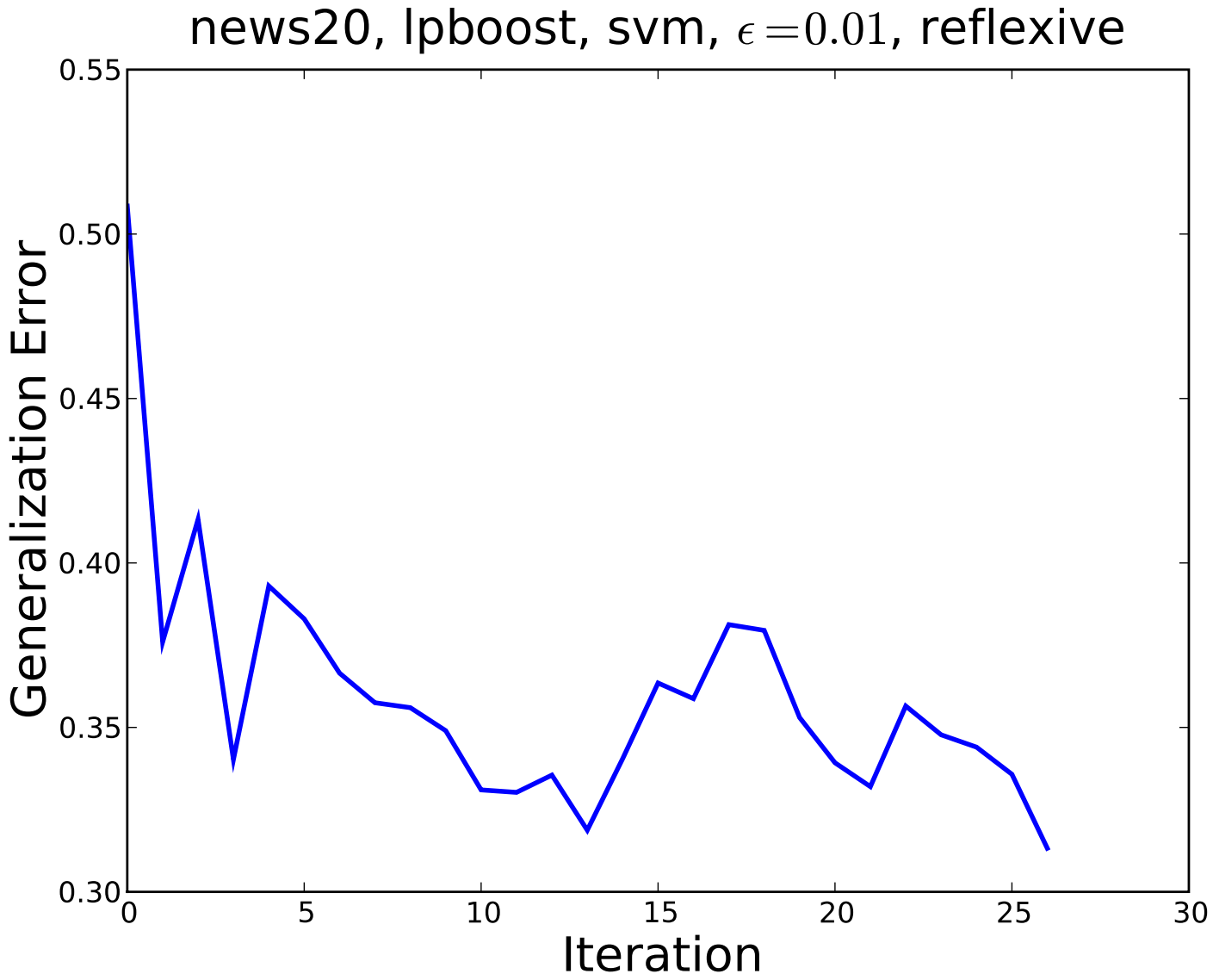
LPBoost is Brittle



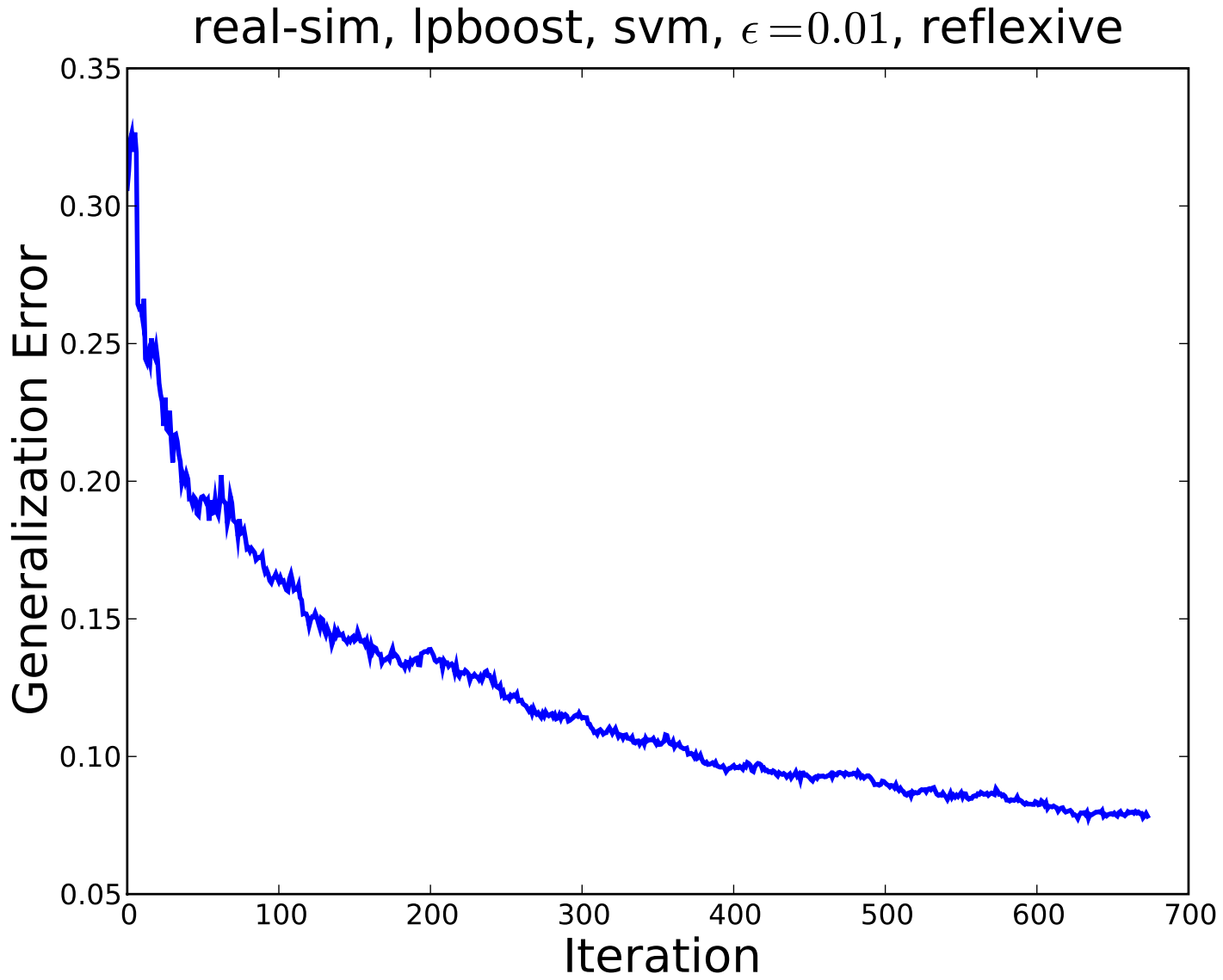
LPBoost is Brittle



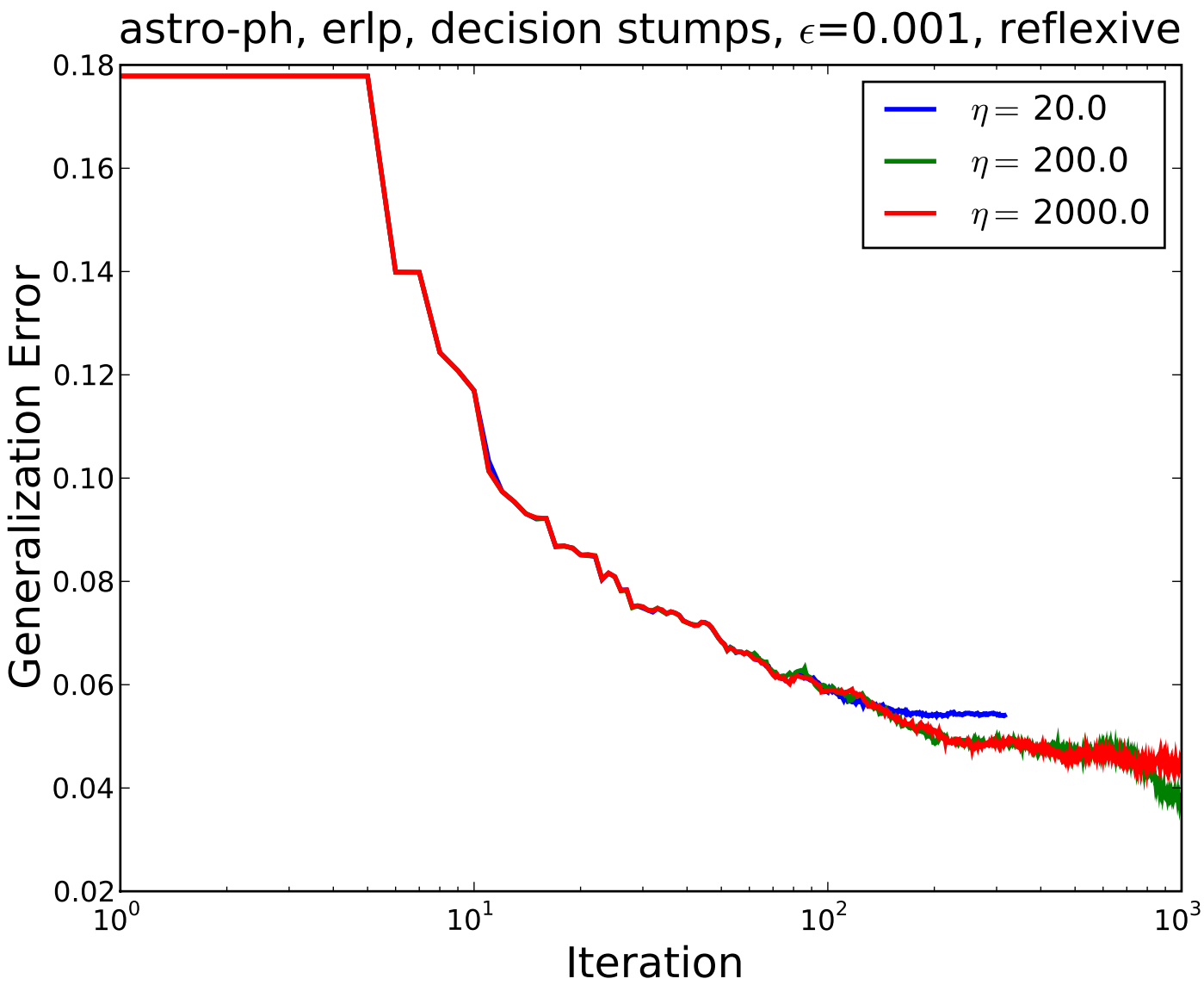
LPBoost is Brittle



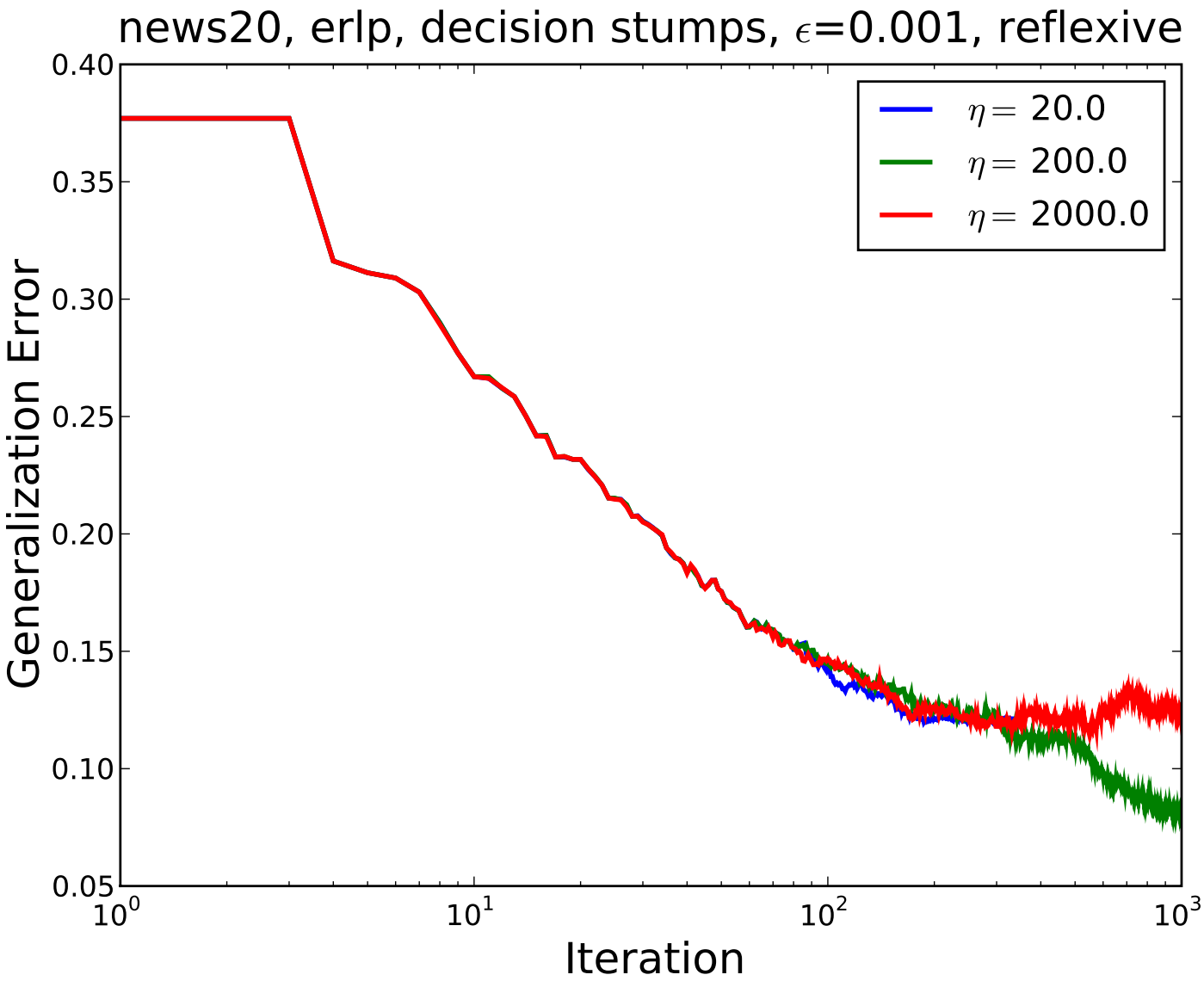
LPBoost is Brittle



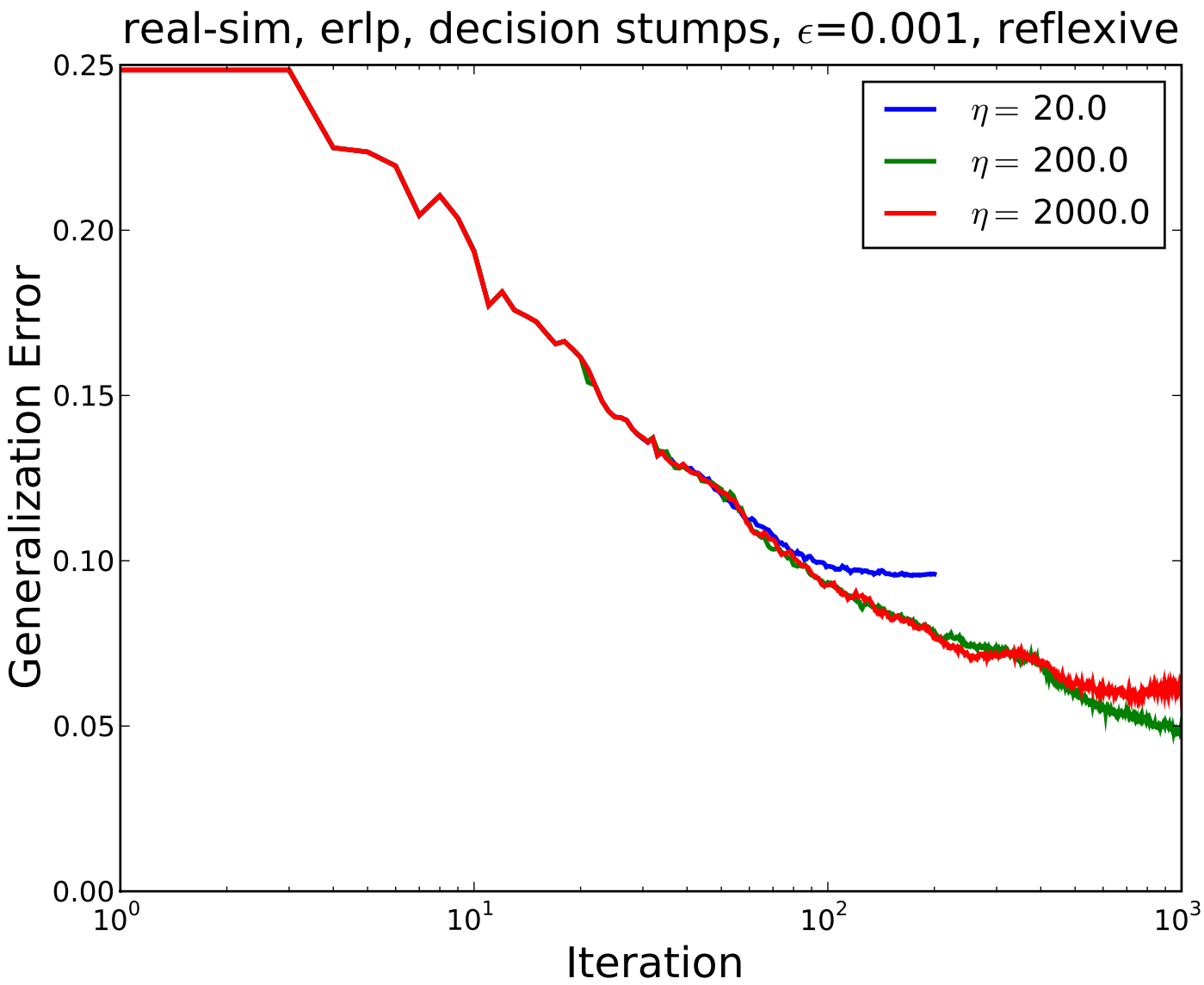
ERLPBoost fixes the problem



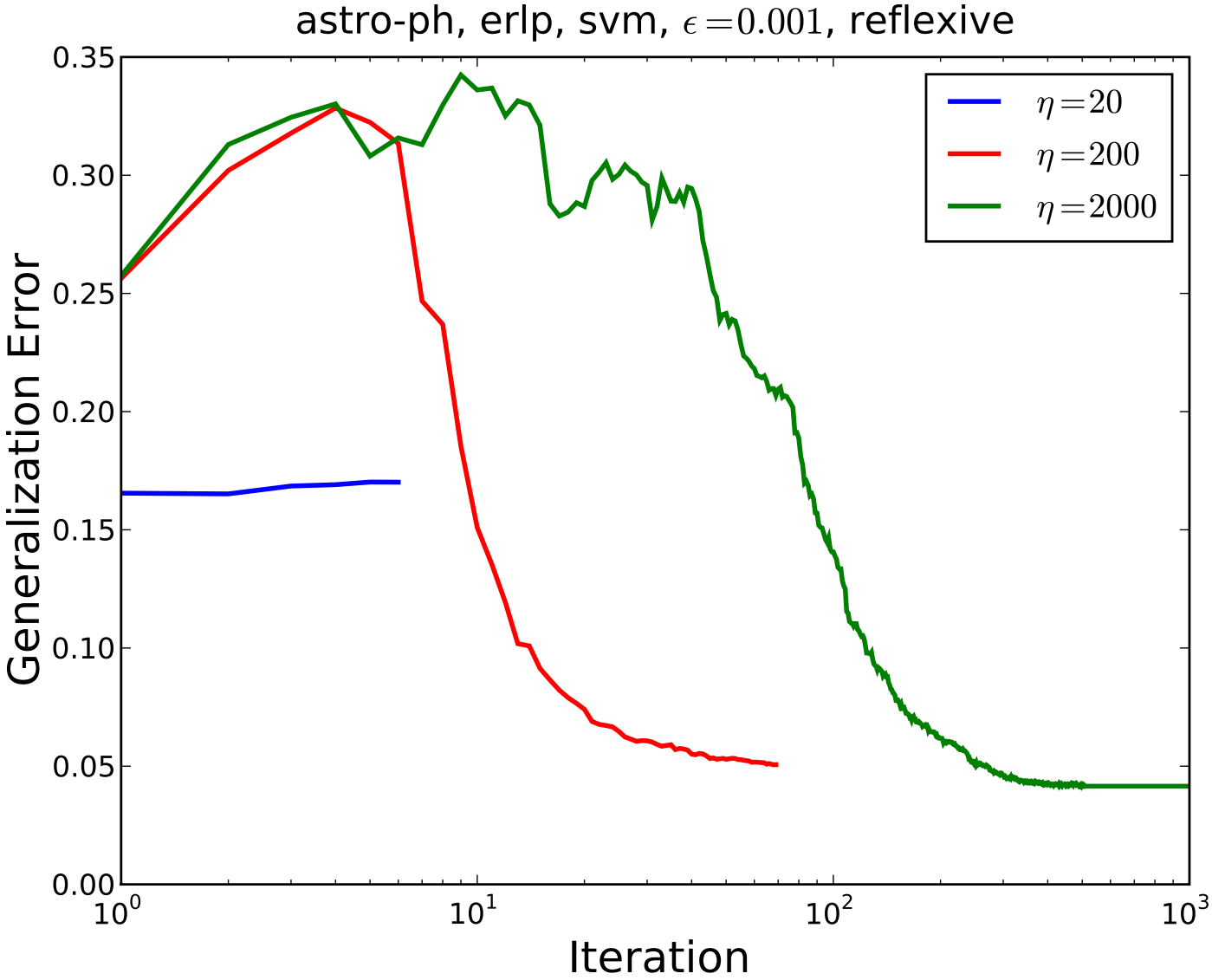
ERLPBoost fixes the problem



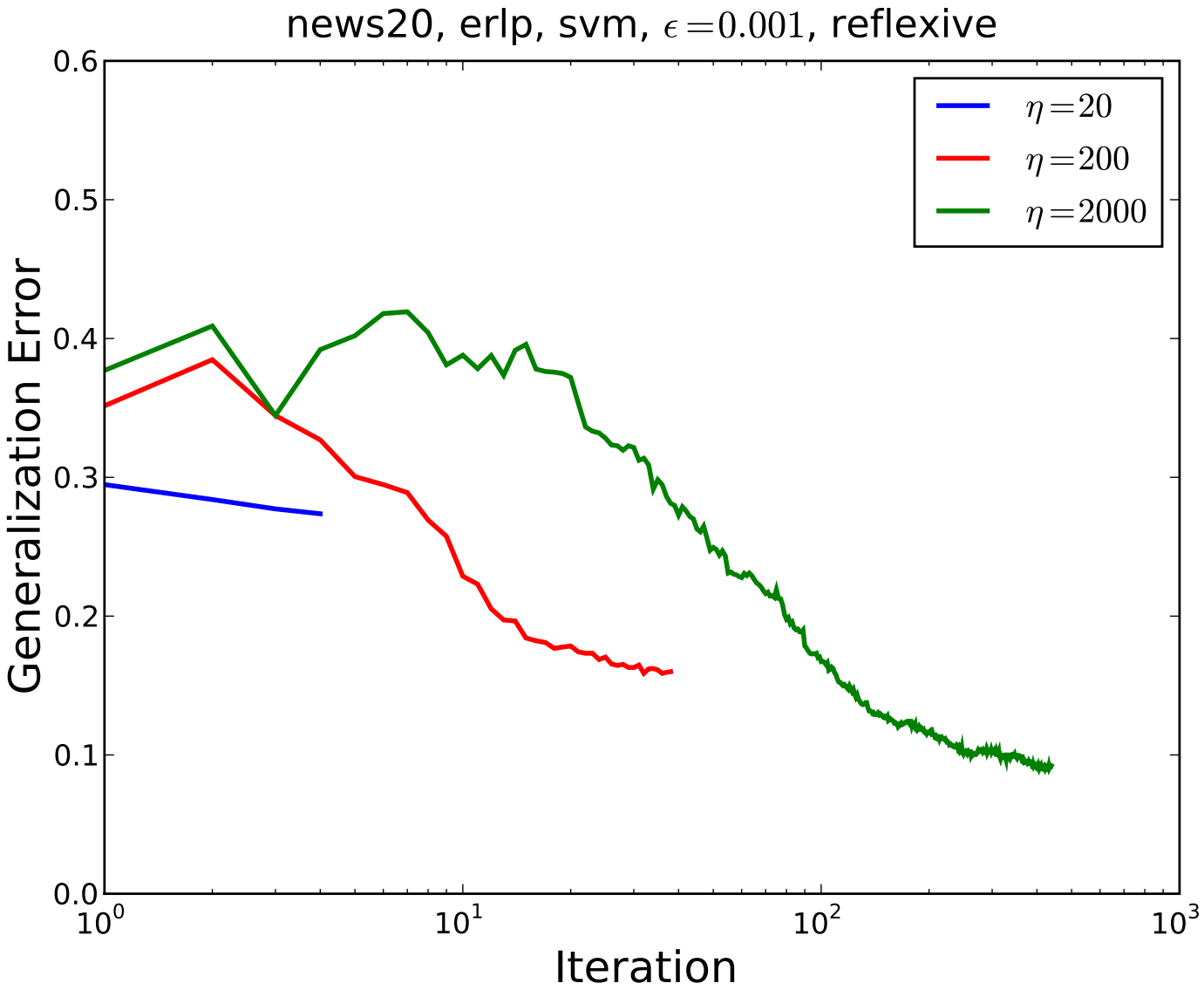
ERLPBoost fixes the problem



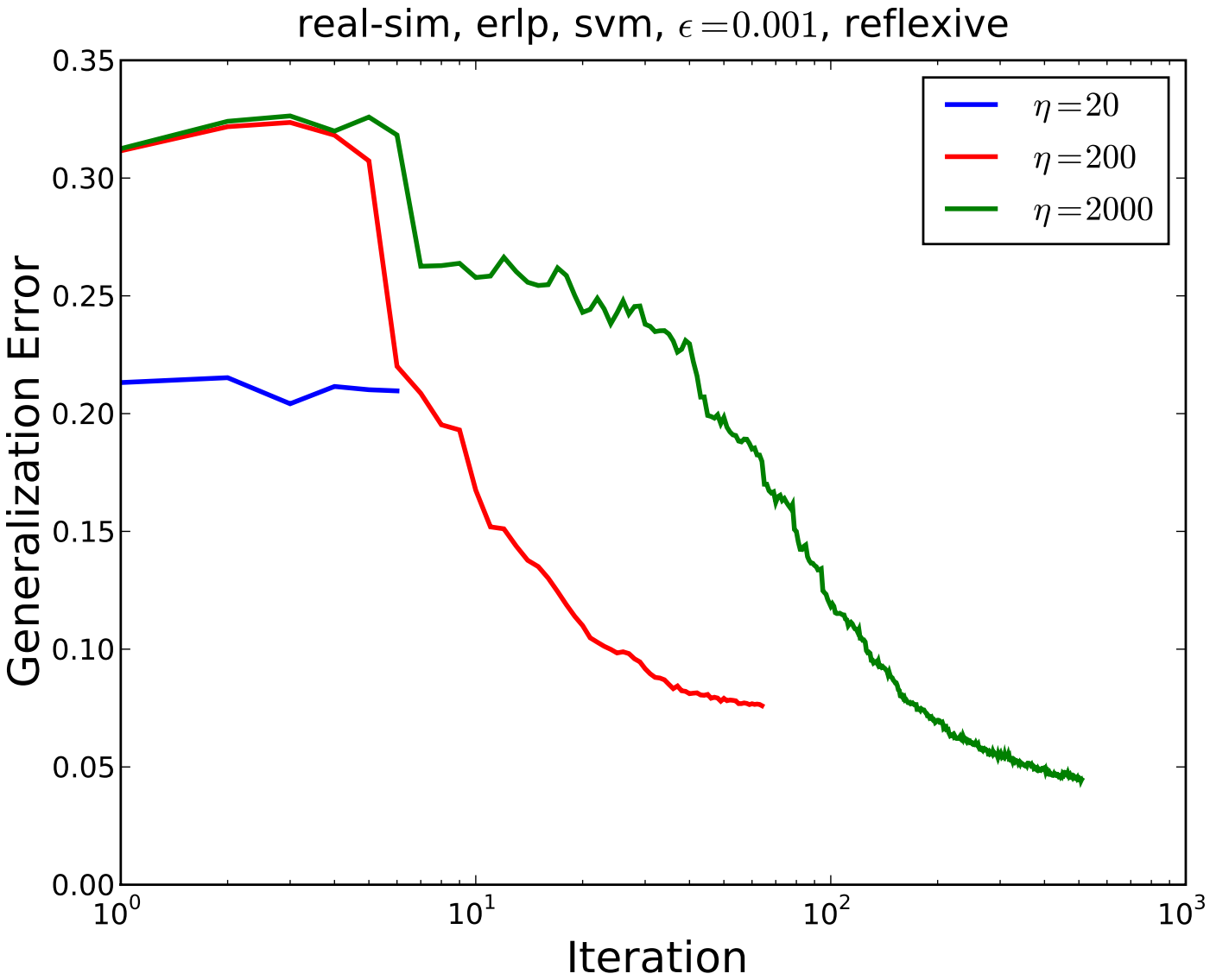
ERLPBoost fixes the problem



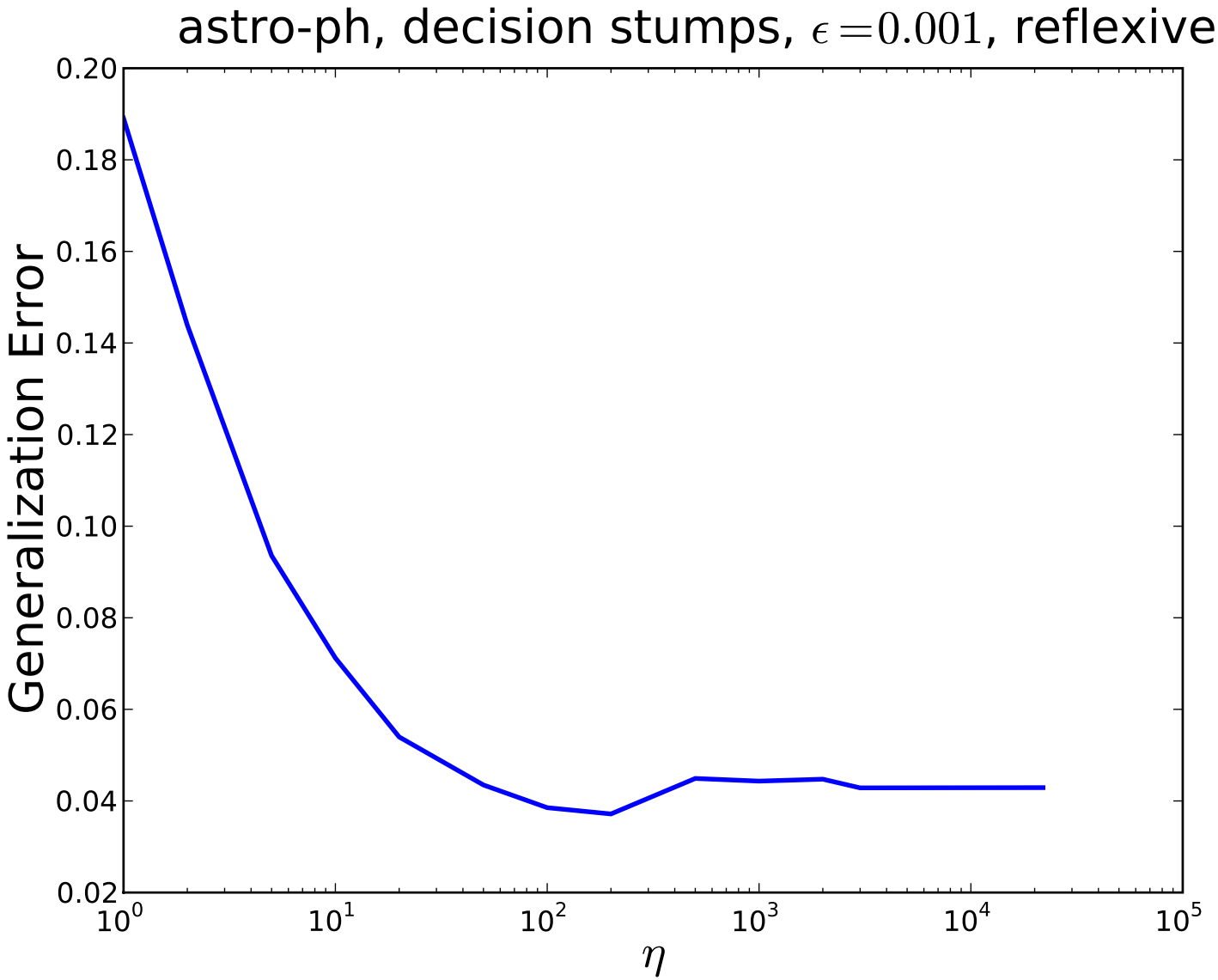
ERLPBoost fixes the problem



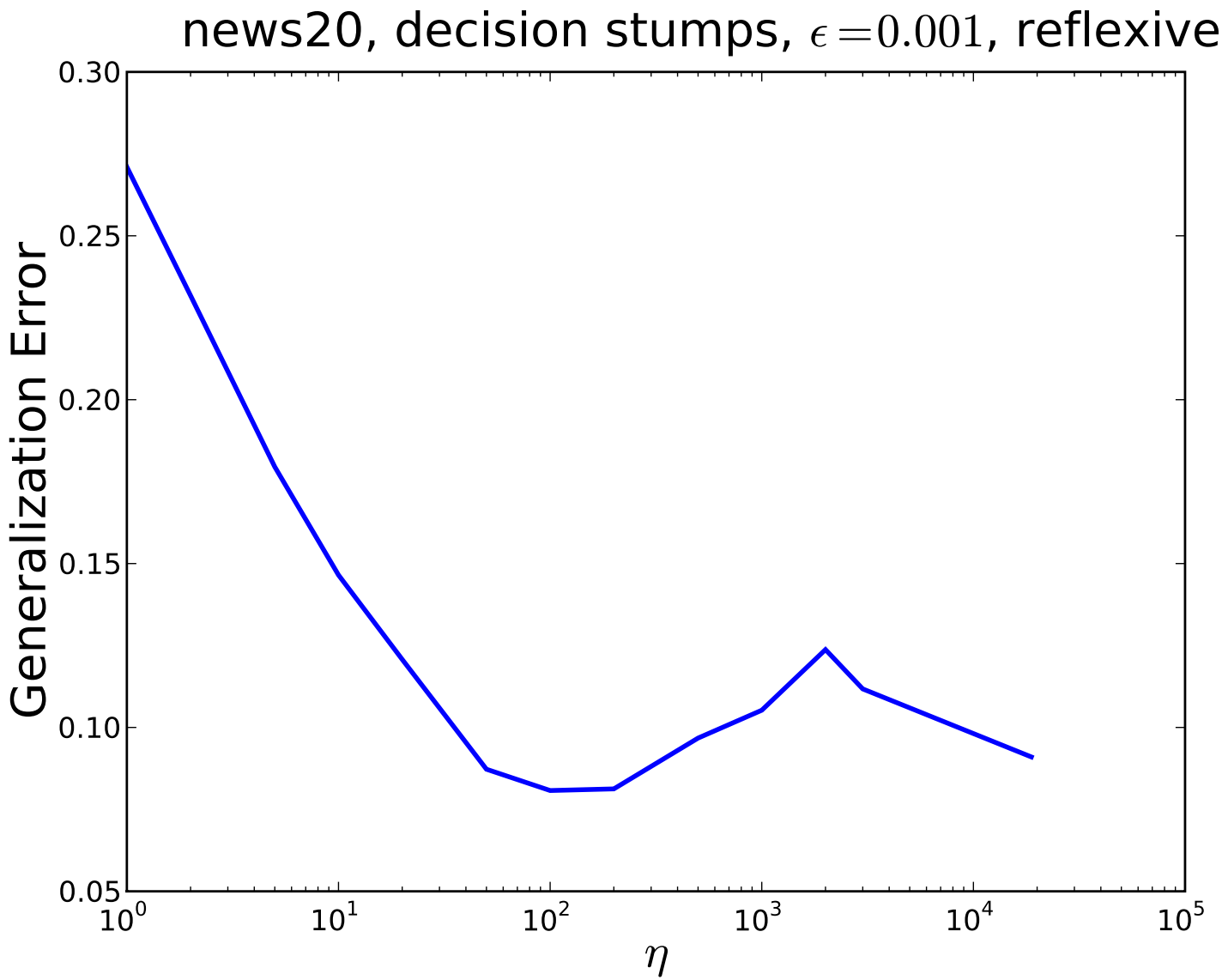
ERLPBoost fixes the problem



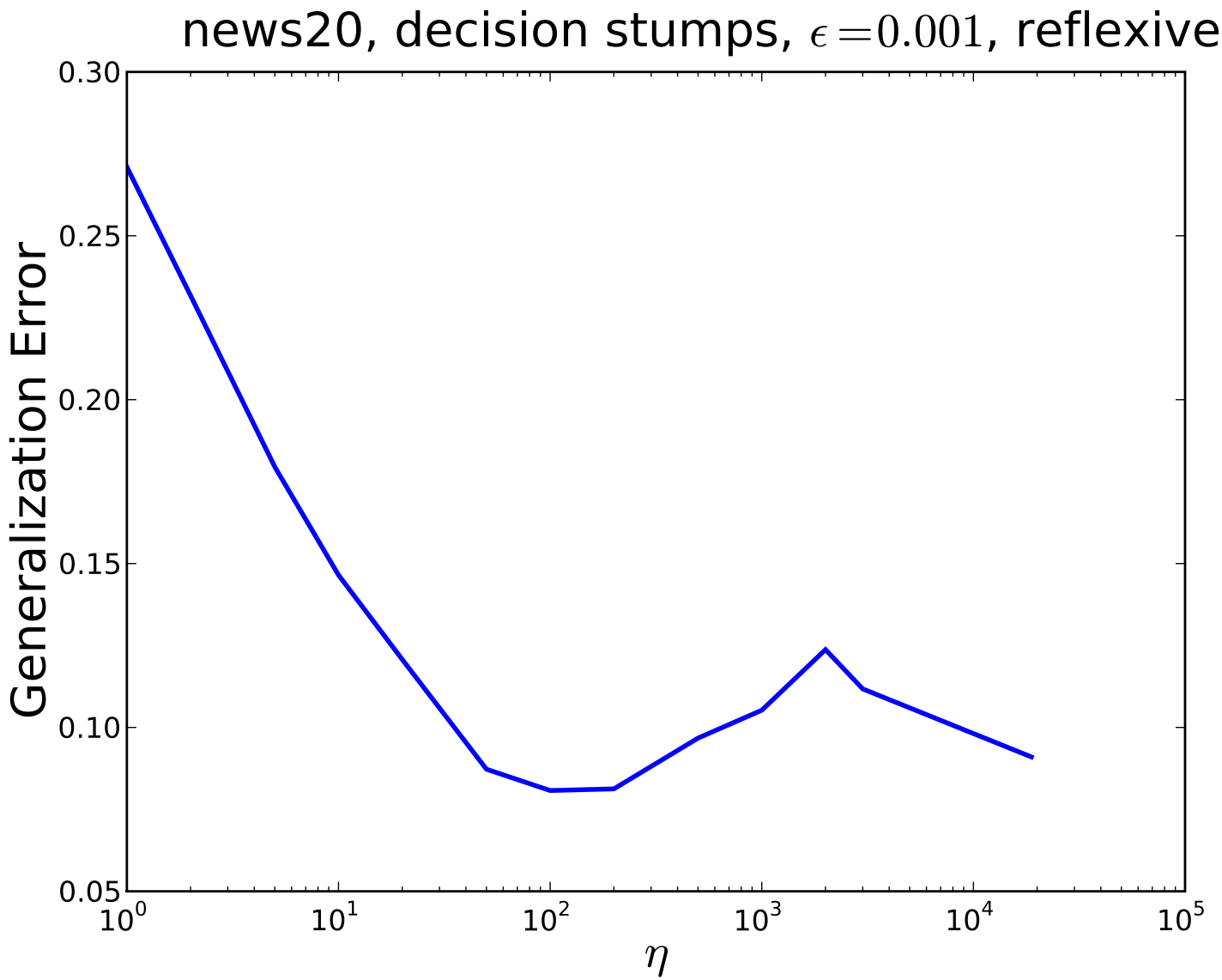
Generalization as a function of η



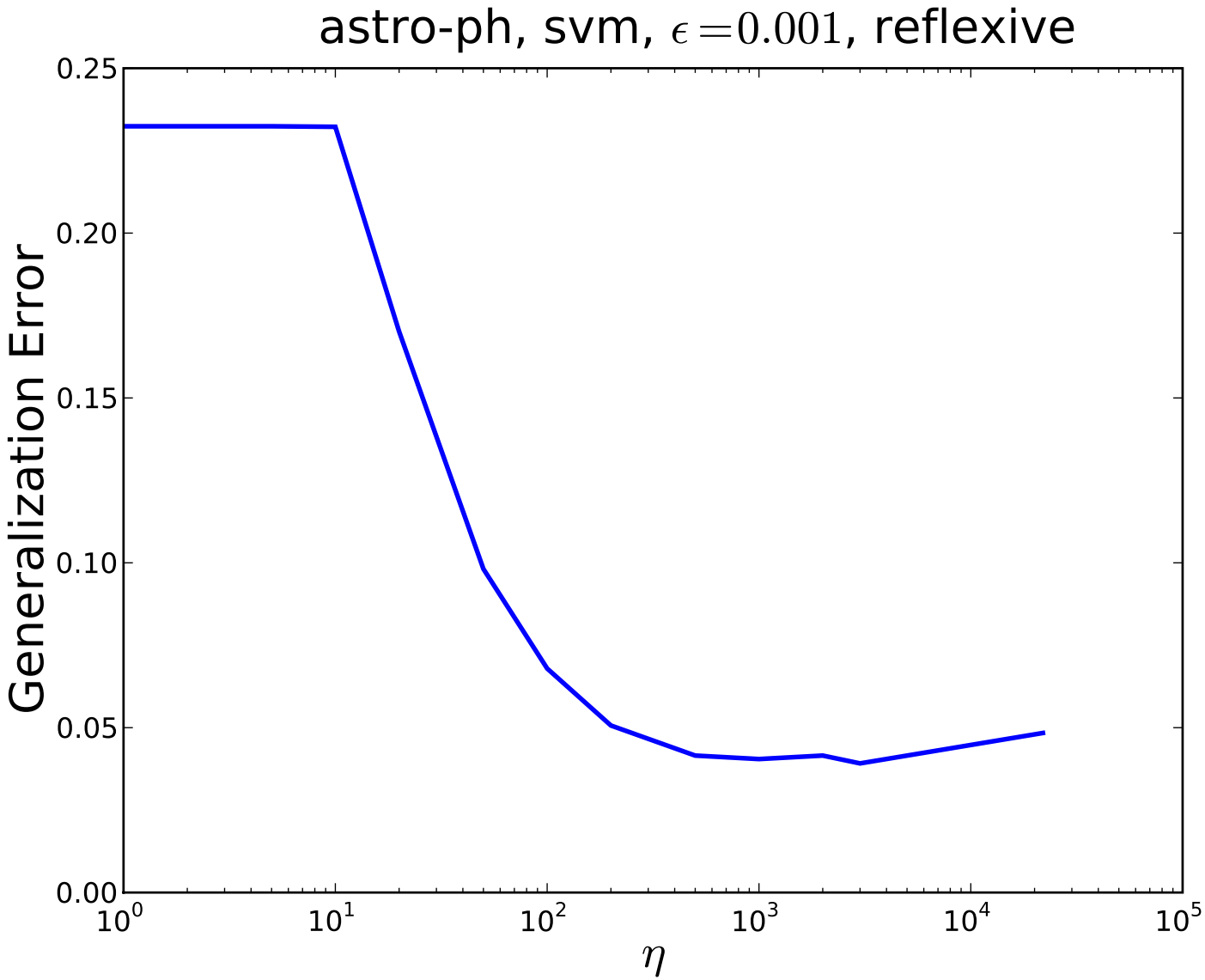
Generalization as a function of η



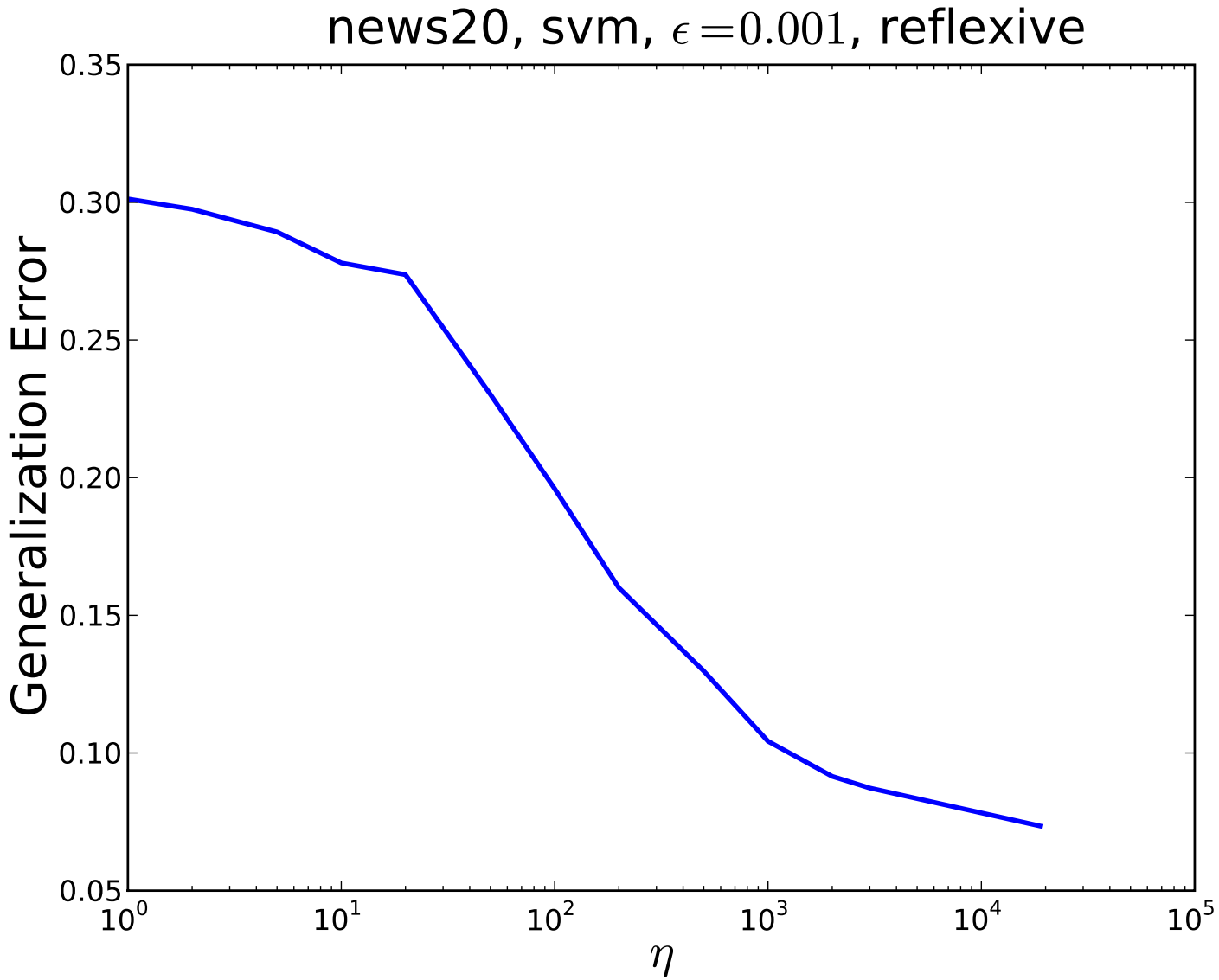
Generalization as a function of η



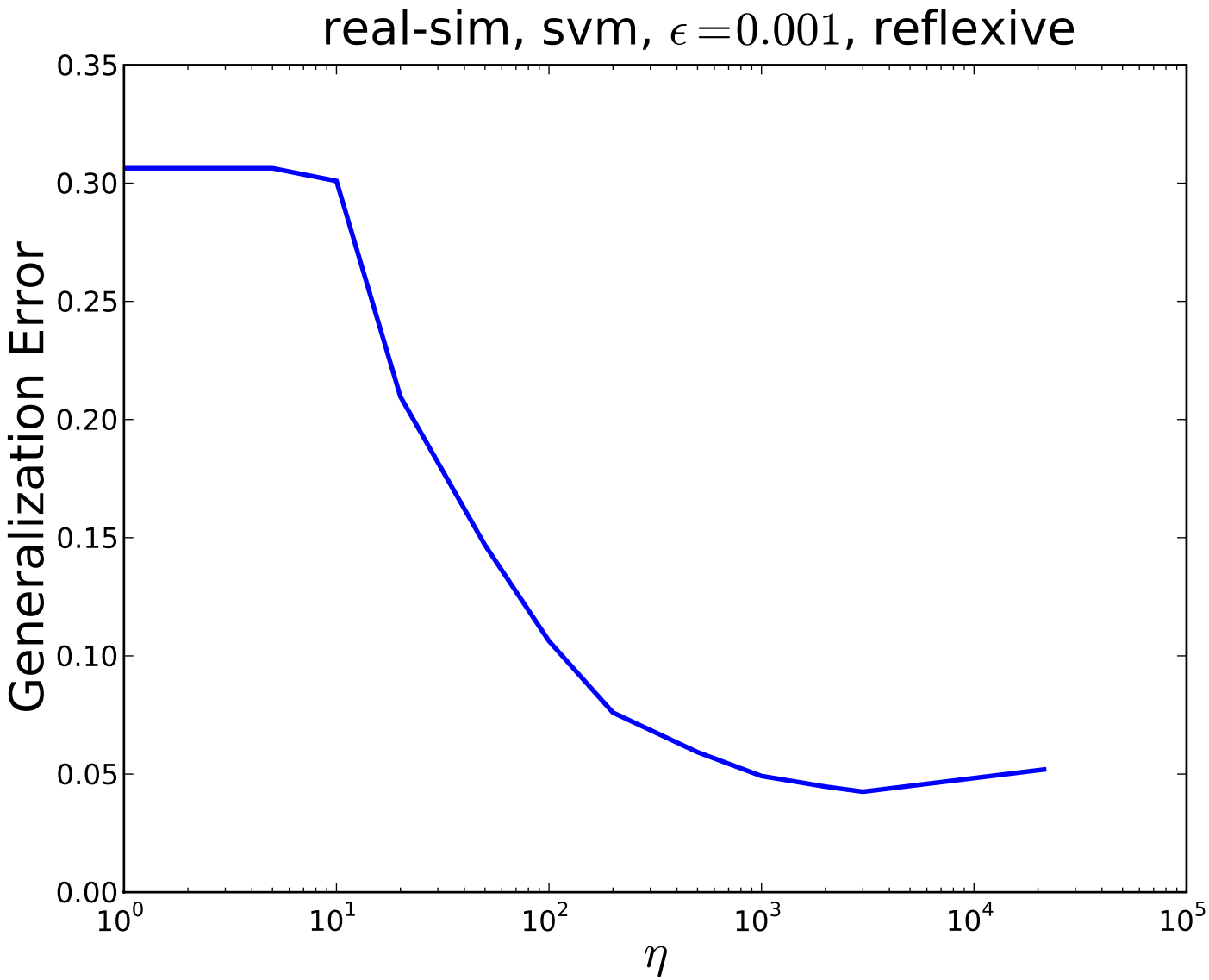
Generalization as a function of η



Generalization as a function of η

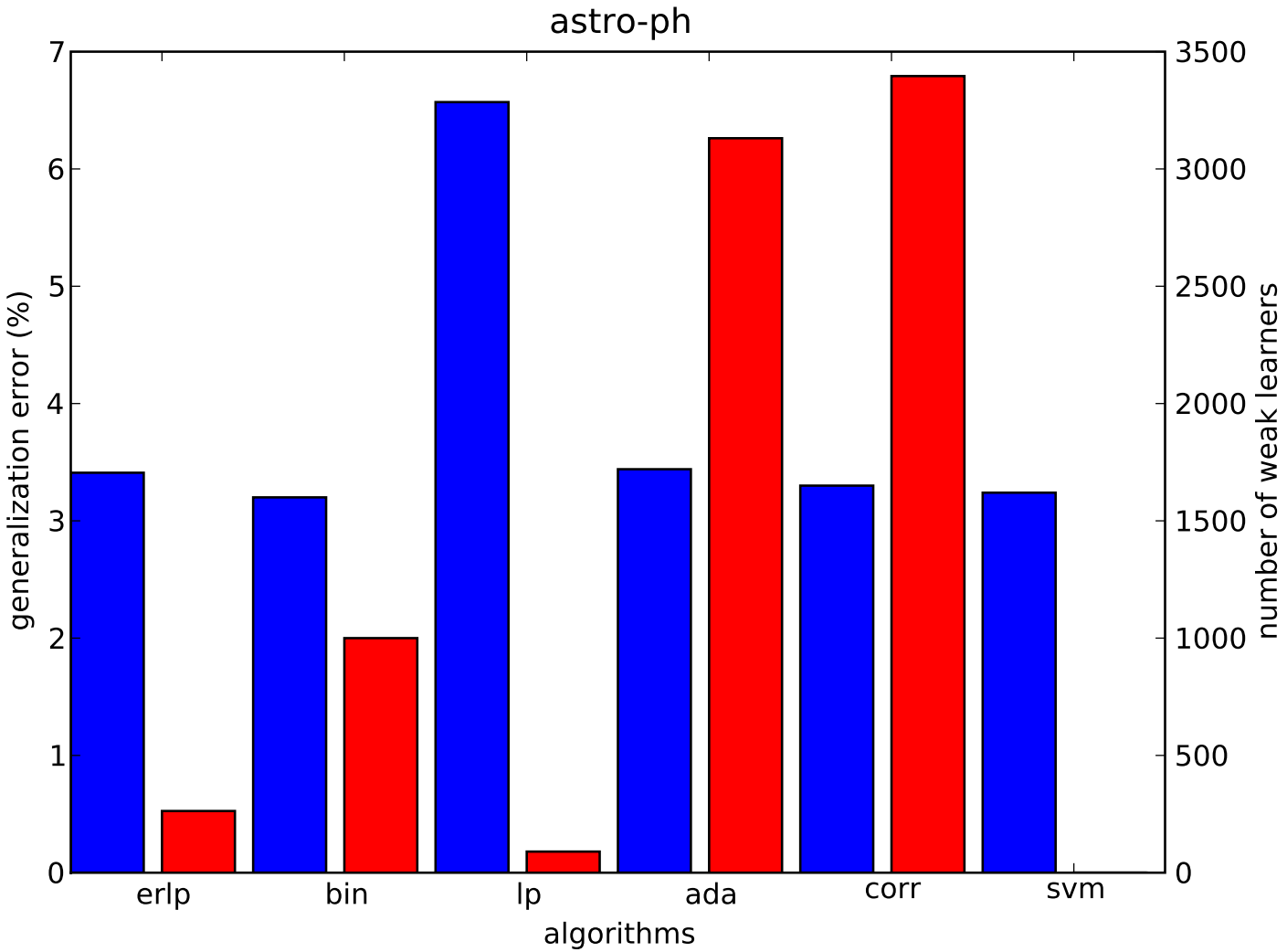


Generalization as a function of η



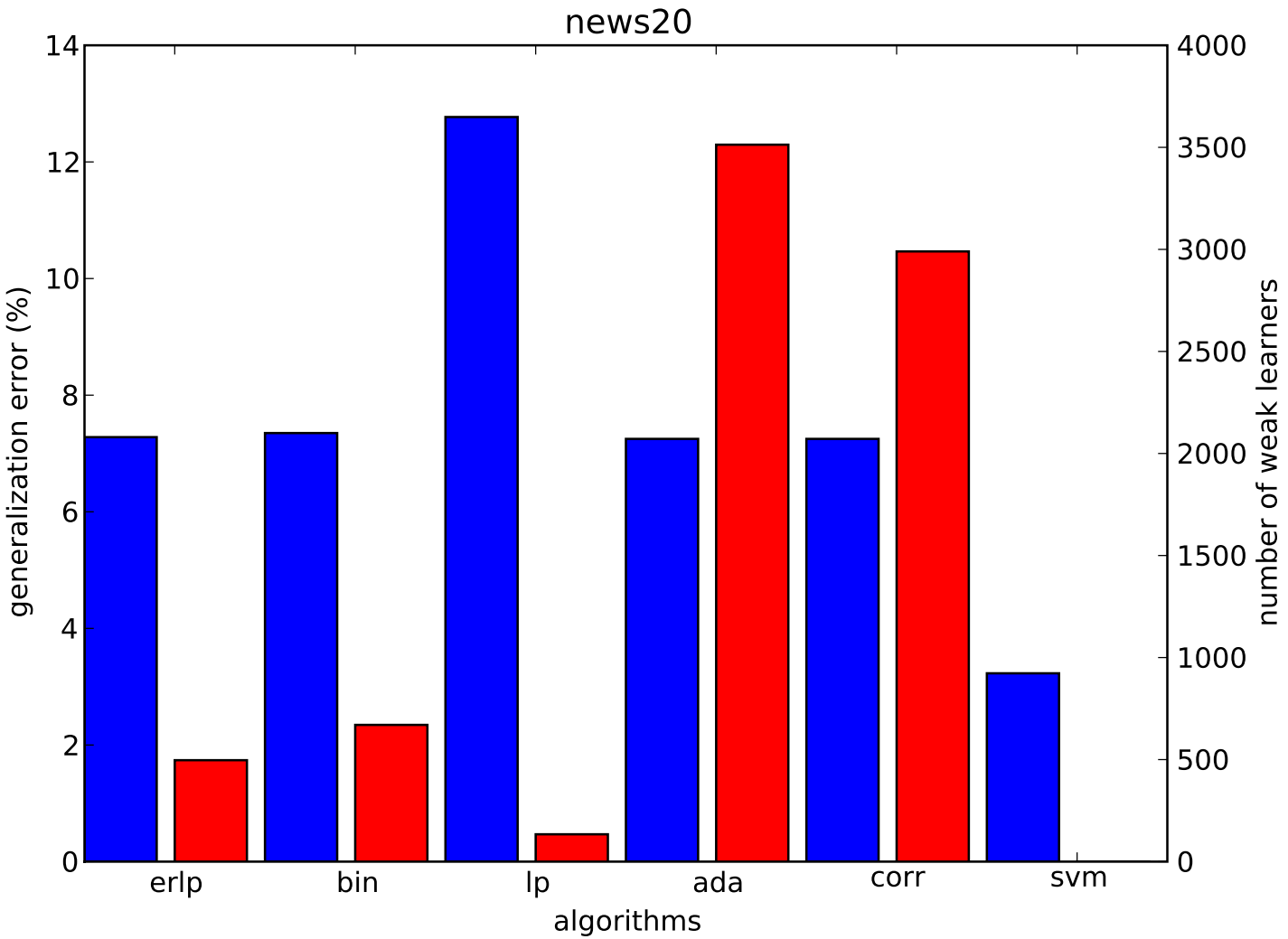
Generalization Error and # of Weak Hypothesis

Parameters tuned for best generalization performance



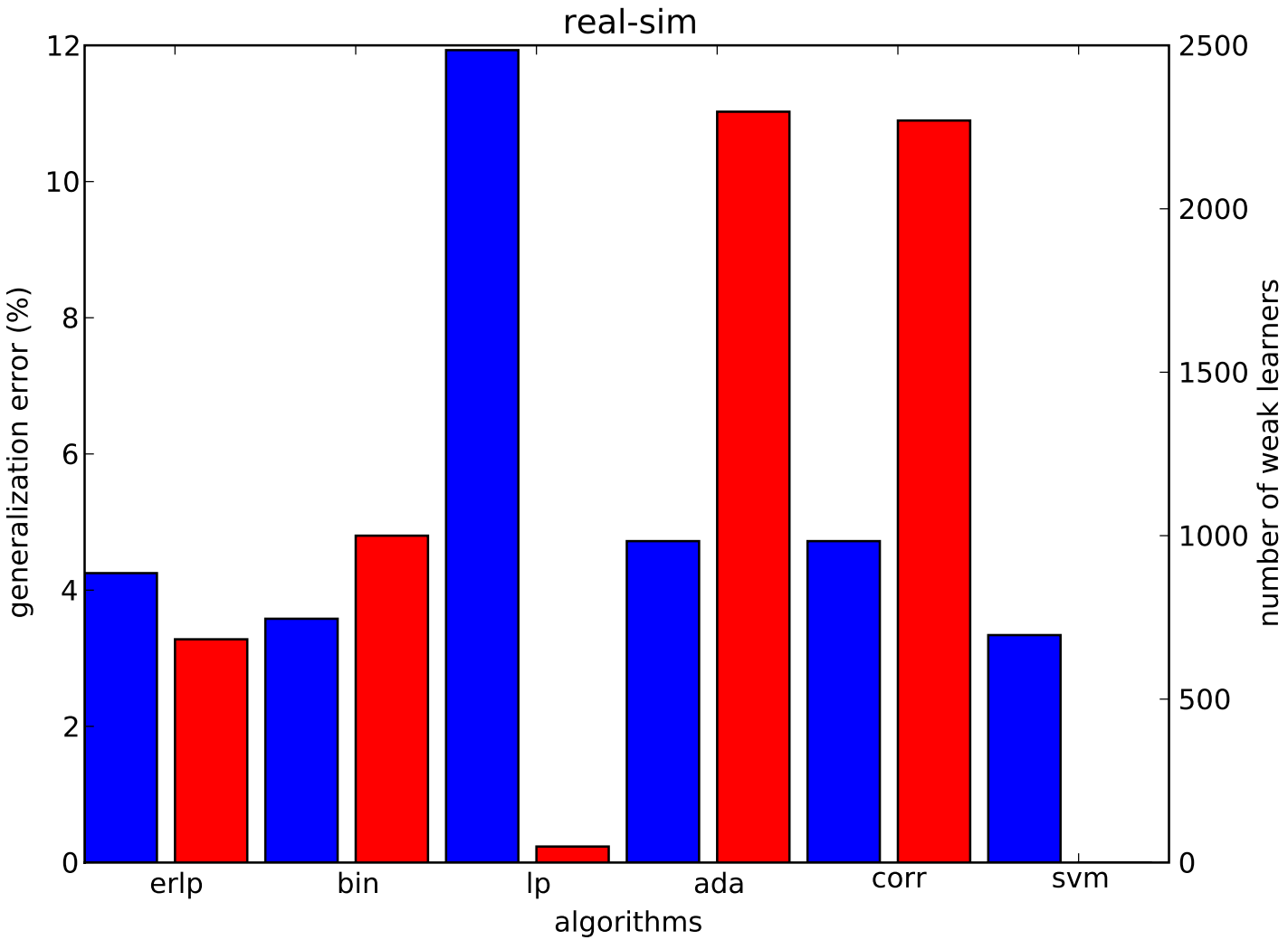
Generalization Error and # of Weak Hypothesis

Parameters tuned for best generalization performance

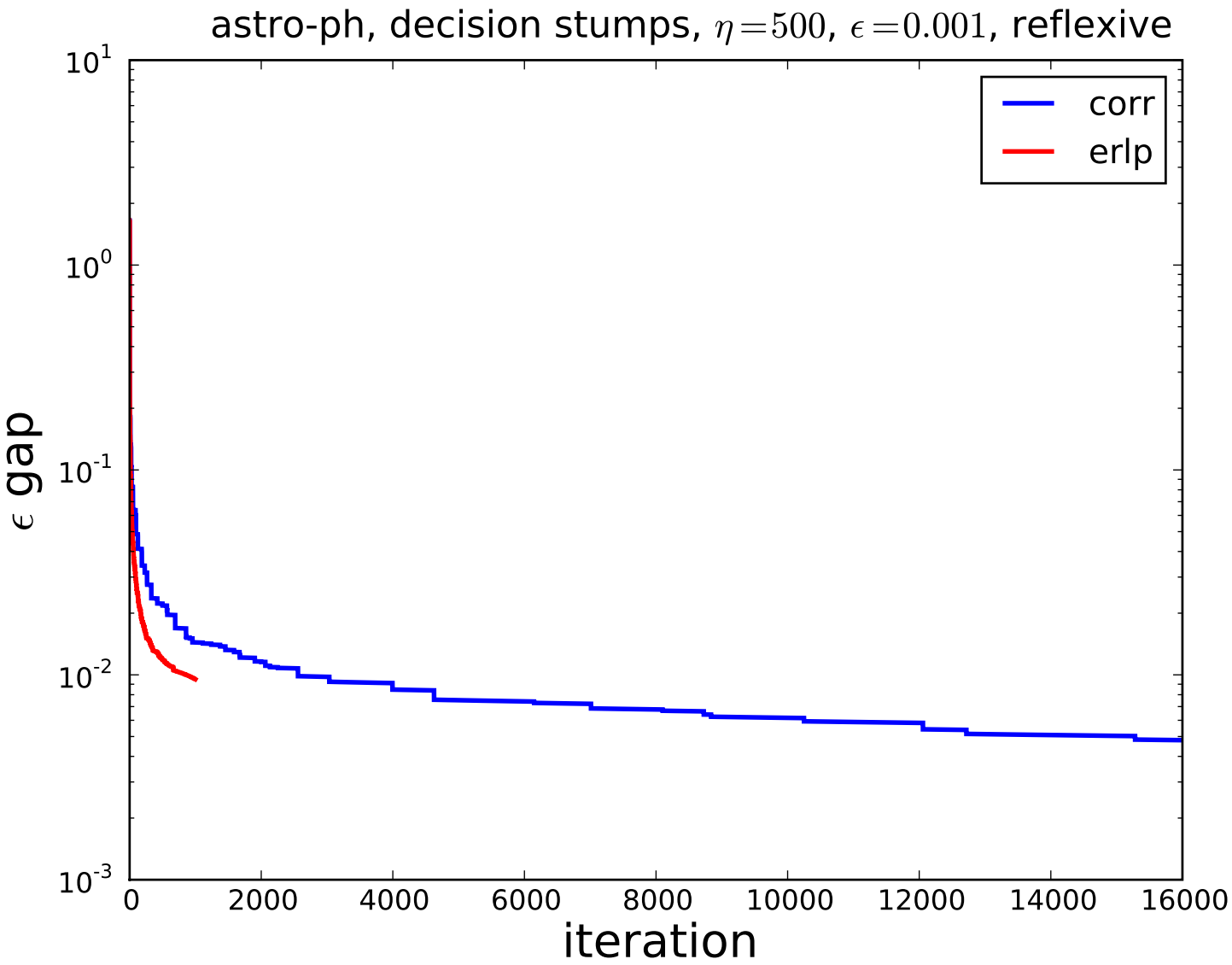


Generalization Error and # of Weak Hypothesis

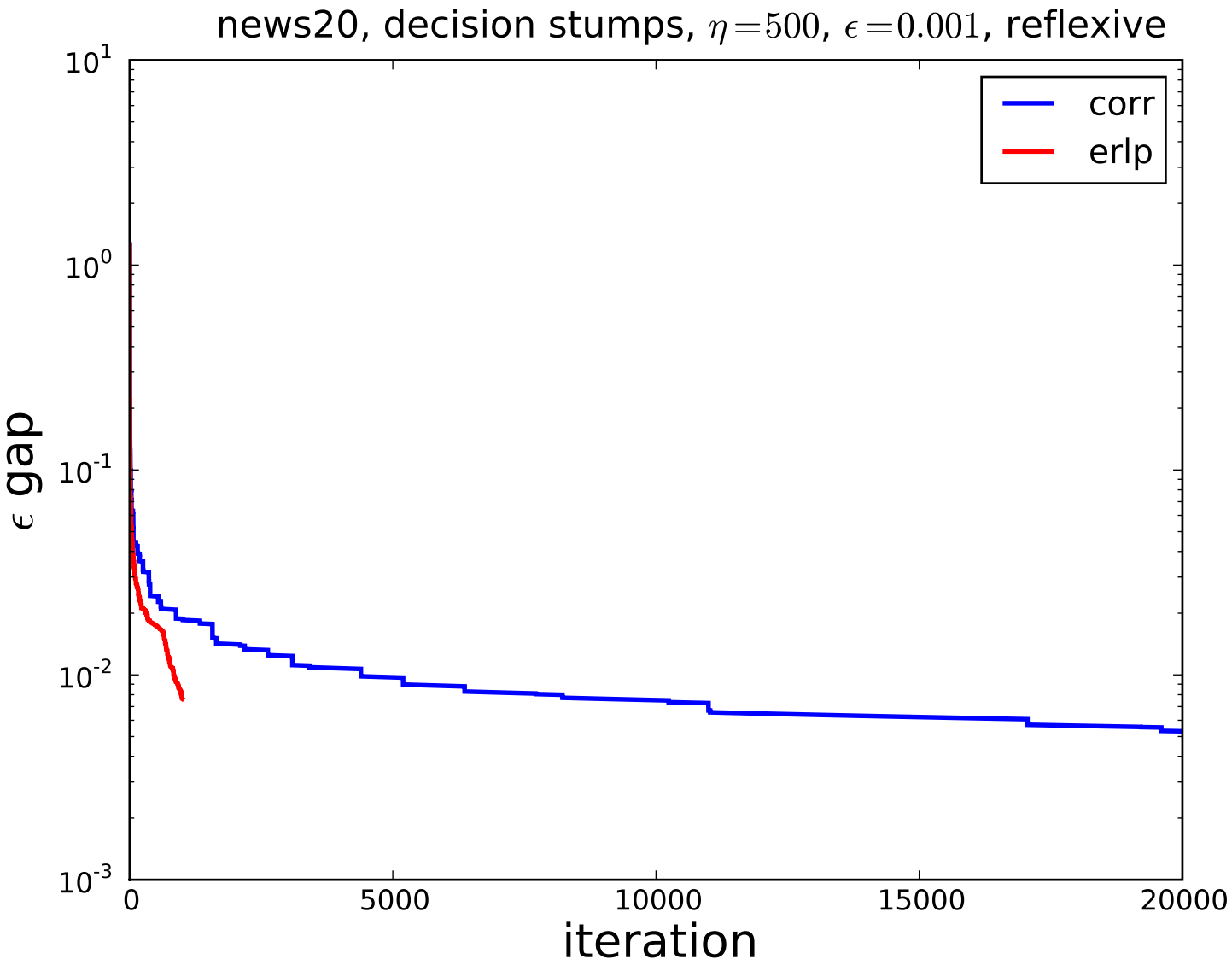
Parameters tuned for best generalization performance



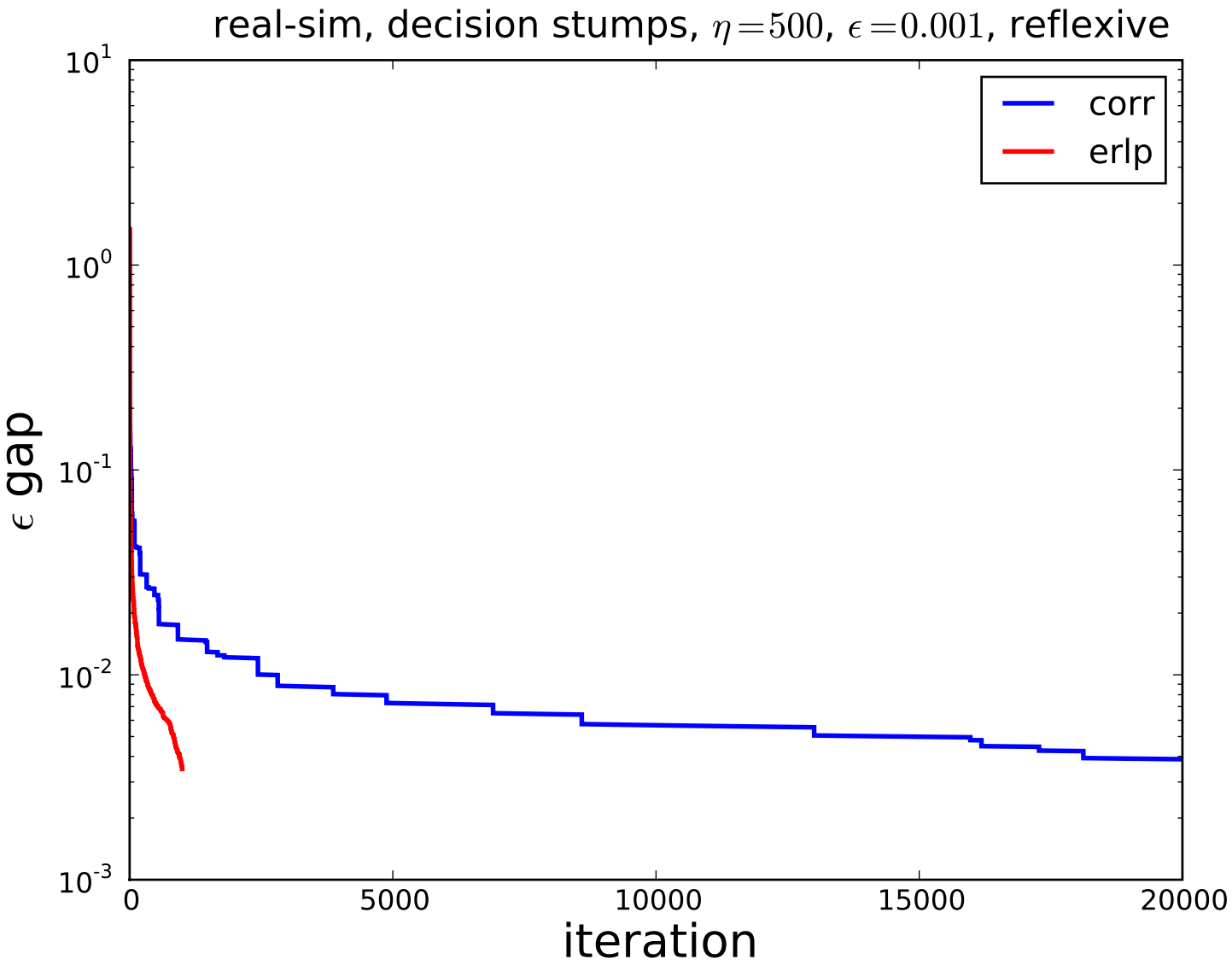
Corrective vs Totally Corrective



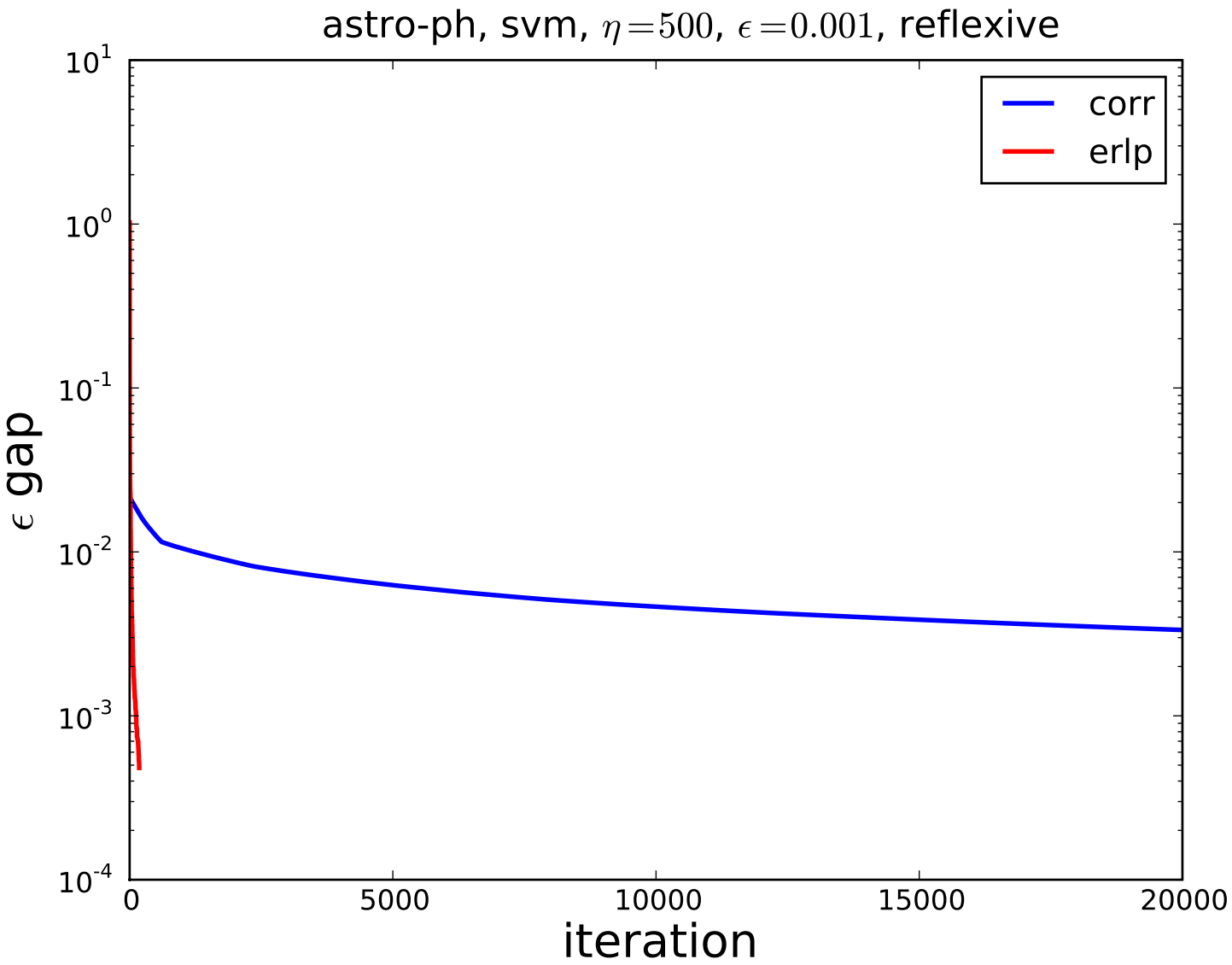
Corrective vs Totally Corrective



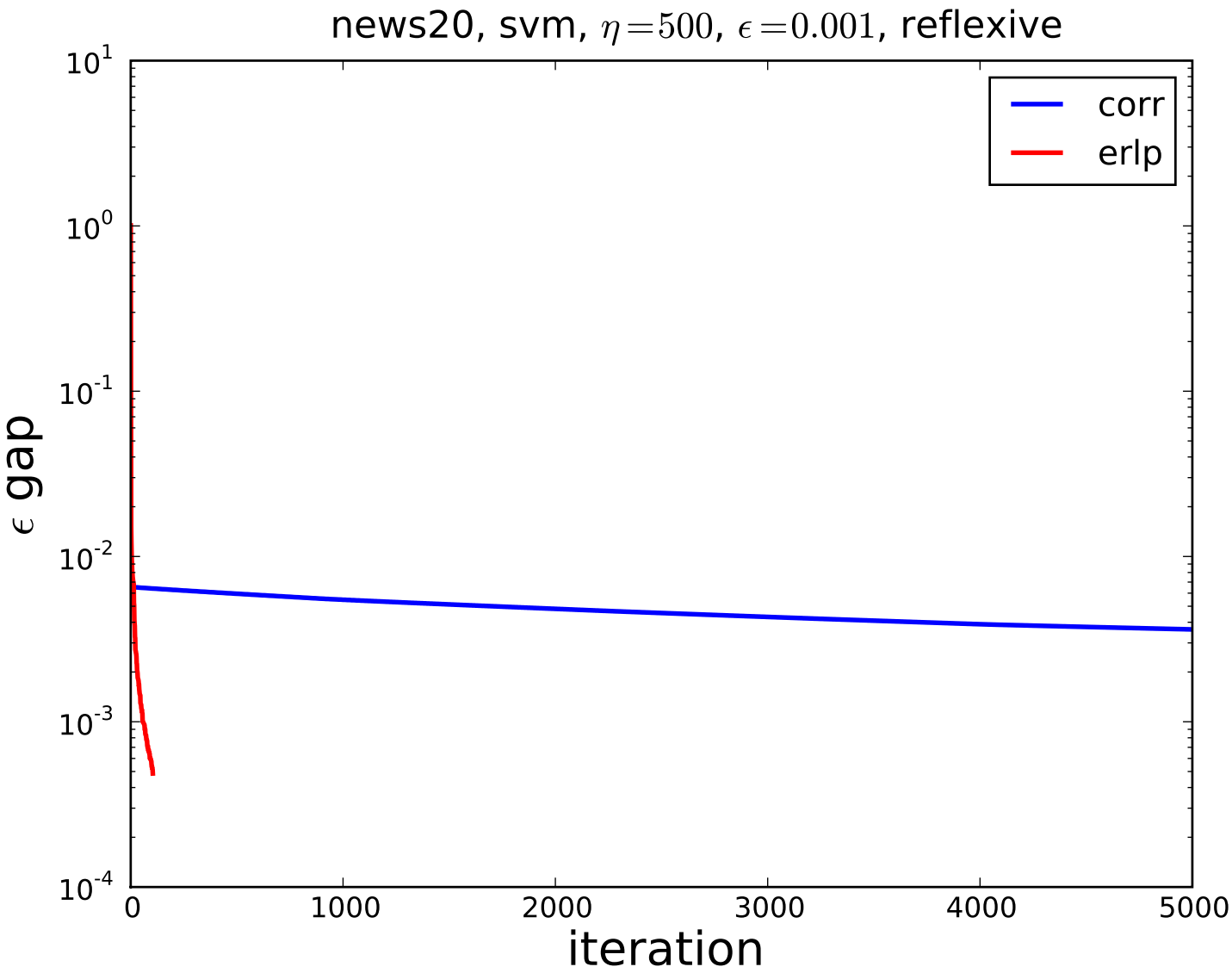
Corrective vs Totally Corrective



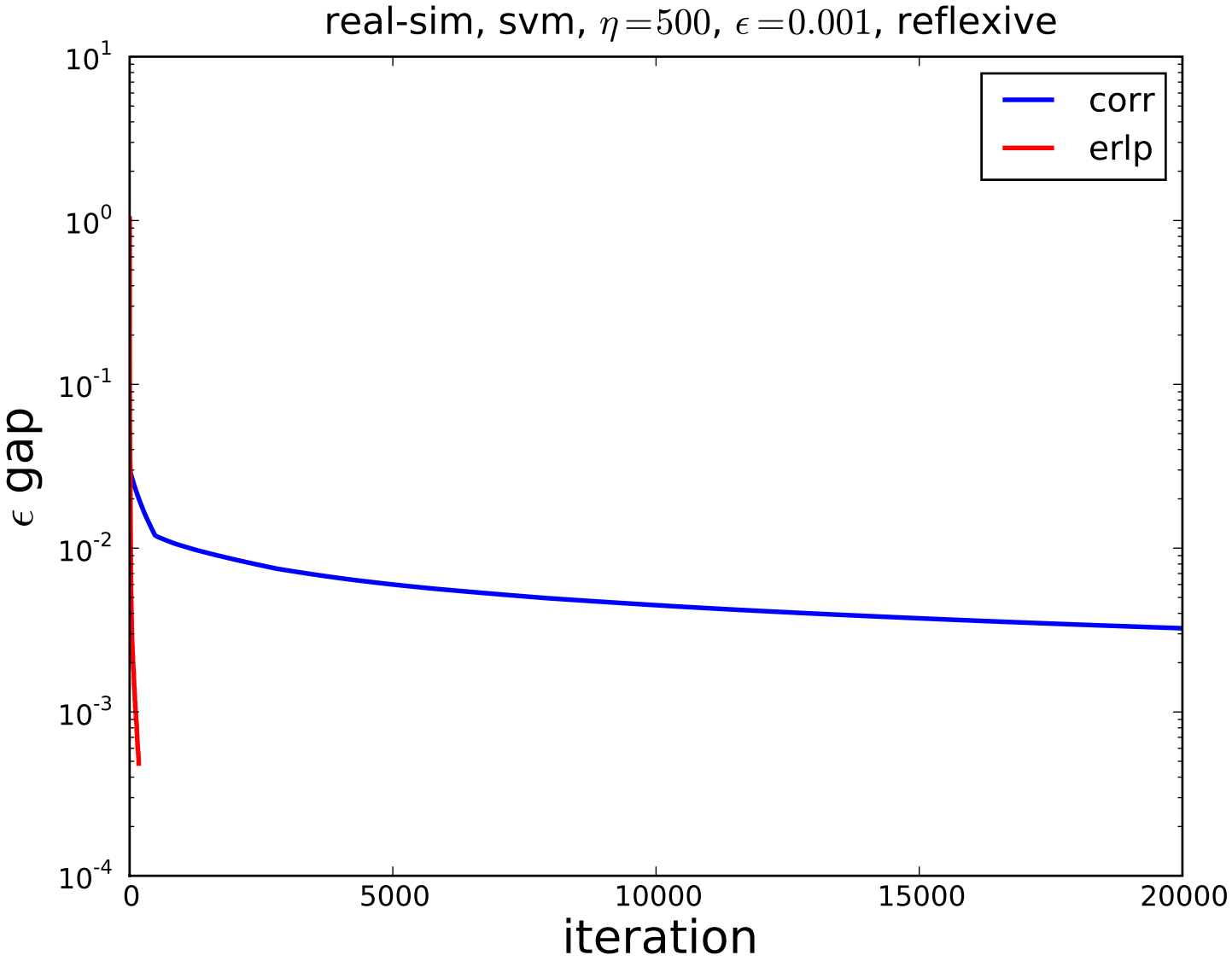
Corrective vs Totally Corrective



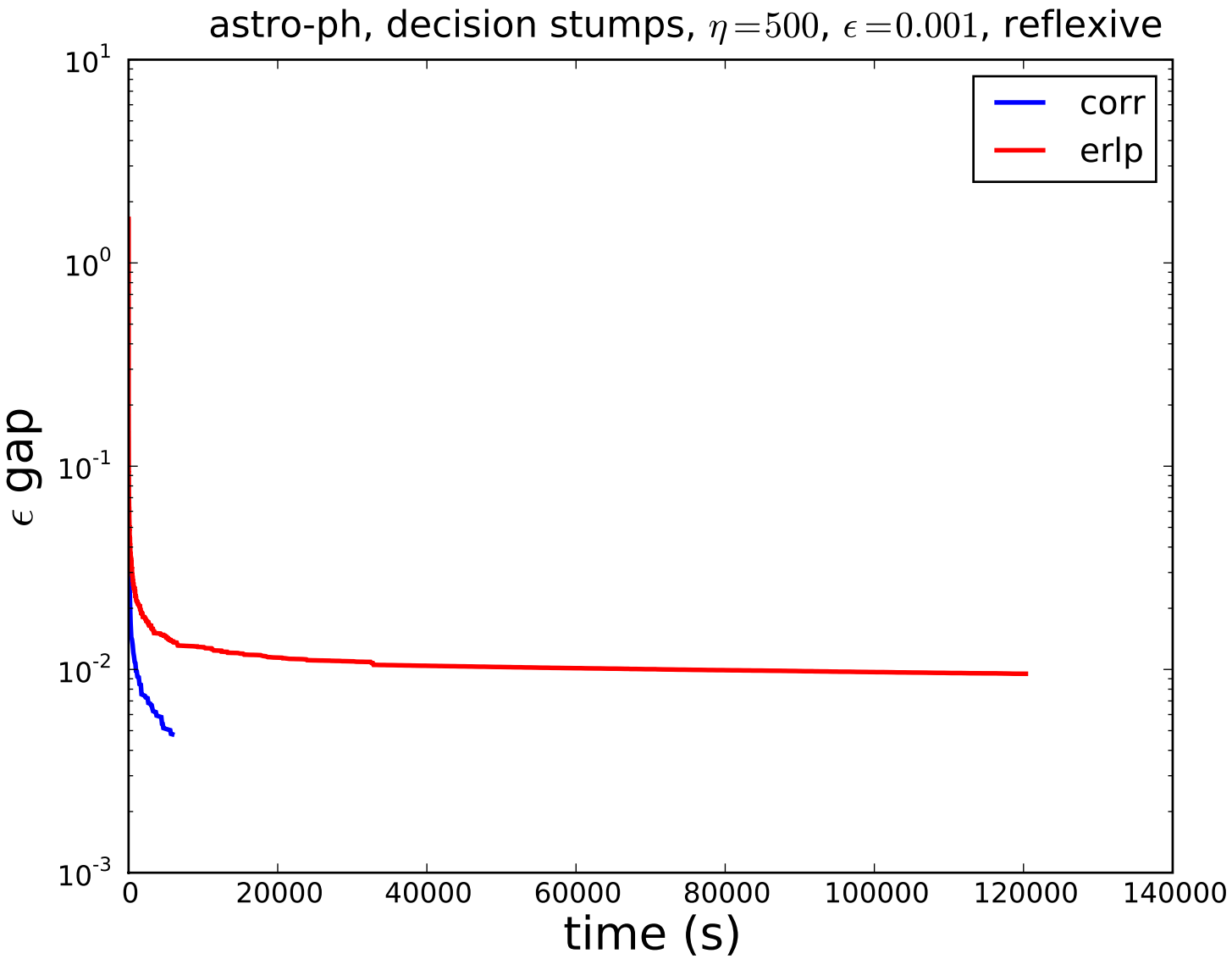
Corrective vs Totally Corrective



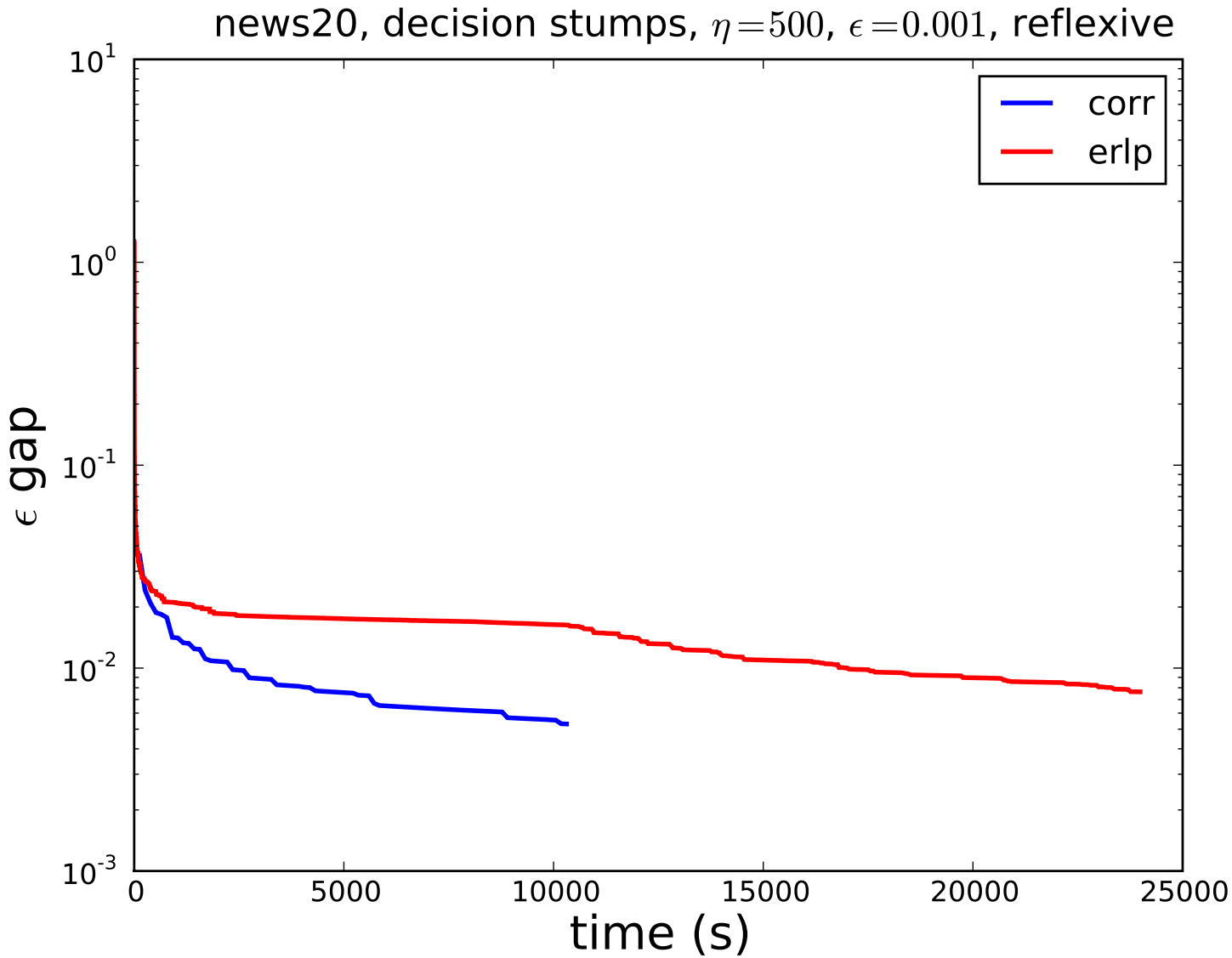
Corrective vs Totally Corrective



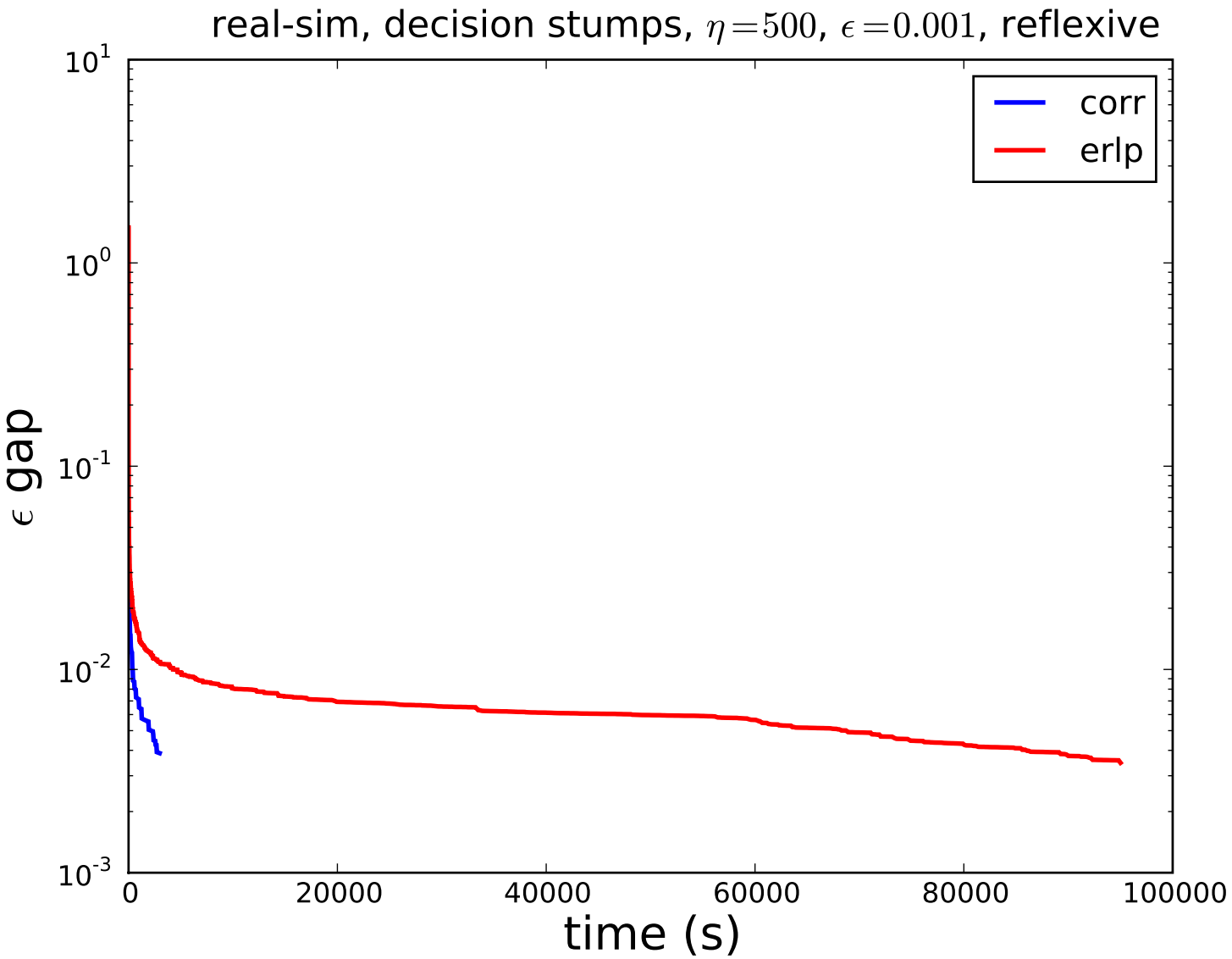
Corrective vs Totally Corrective contd.



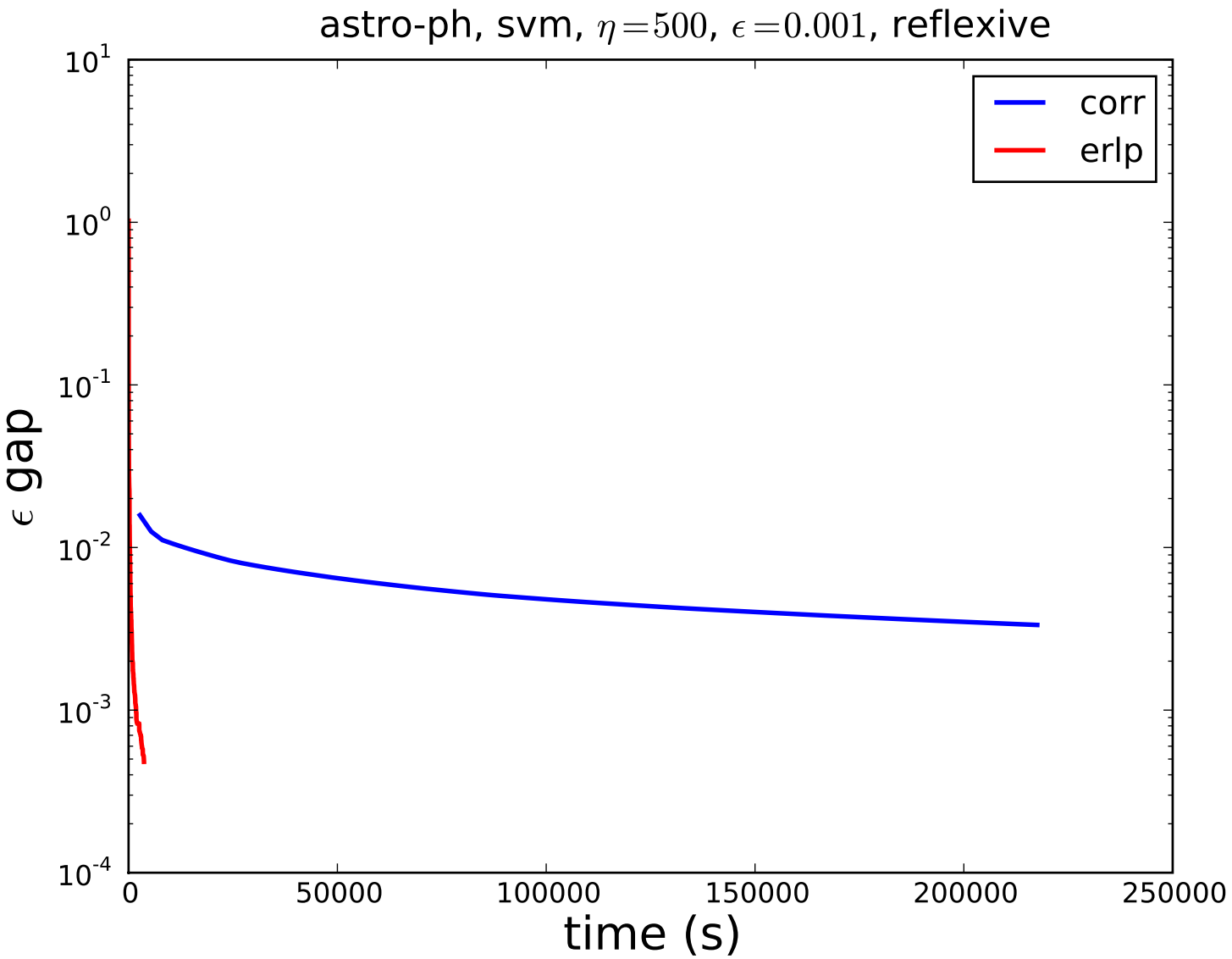
Corrective vs Totally Corrective contd.



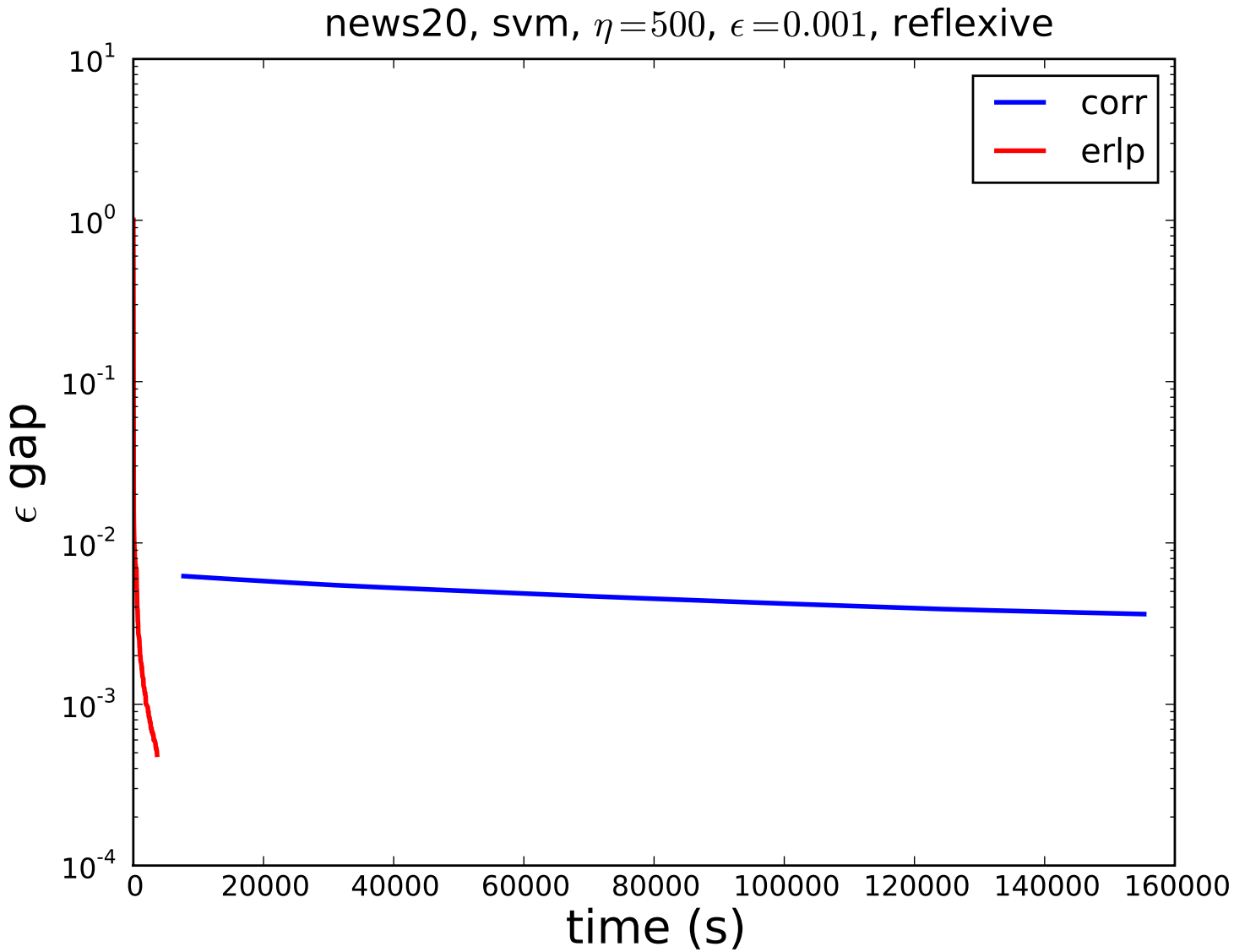
Corrective vs Totally Corrective contd.



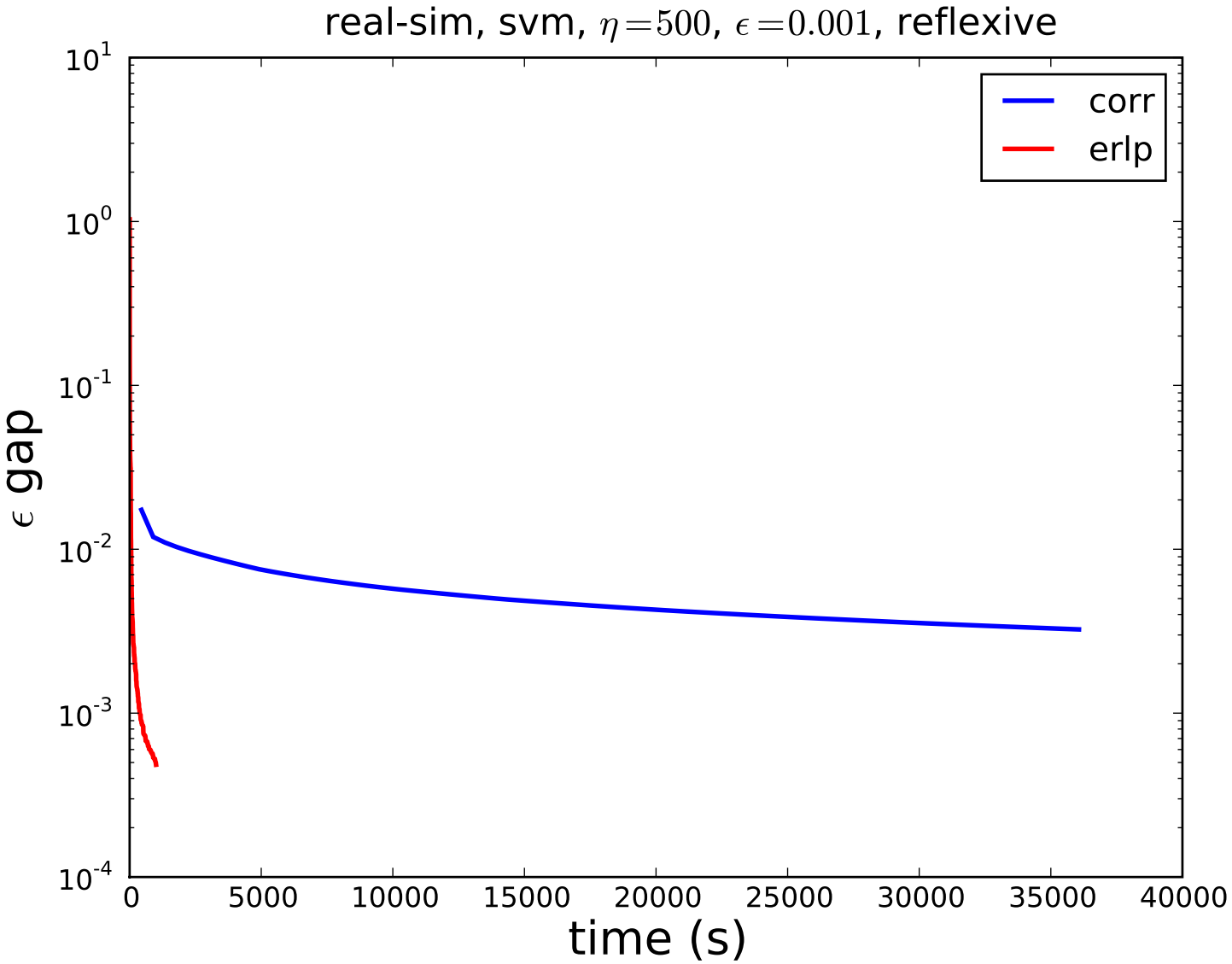
Corrective vs Totally Corrective contd.



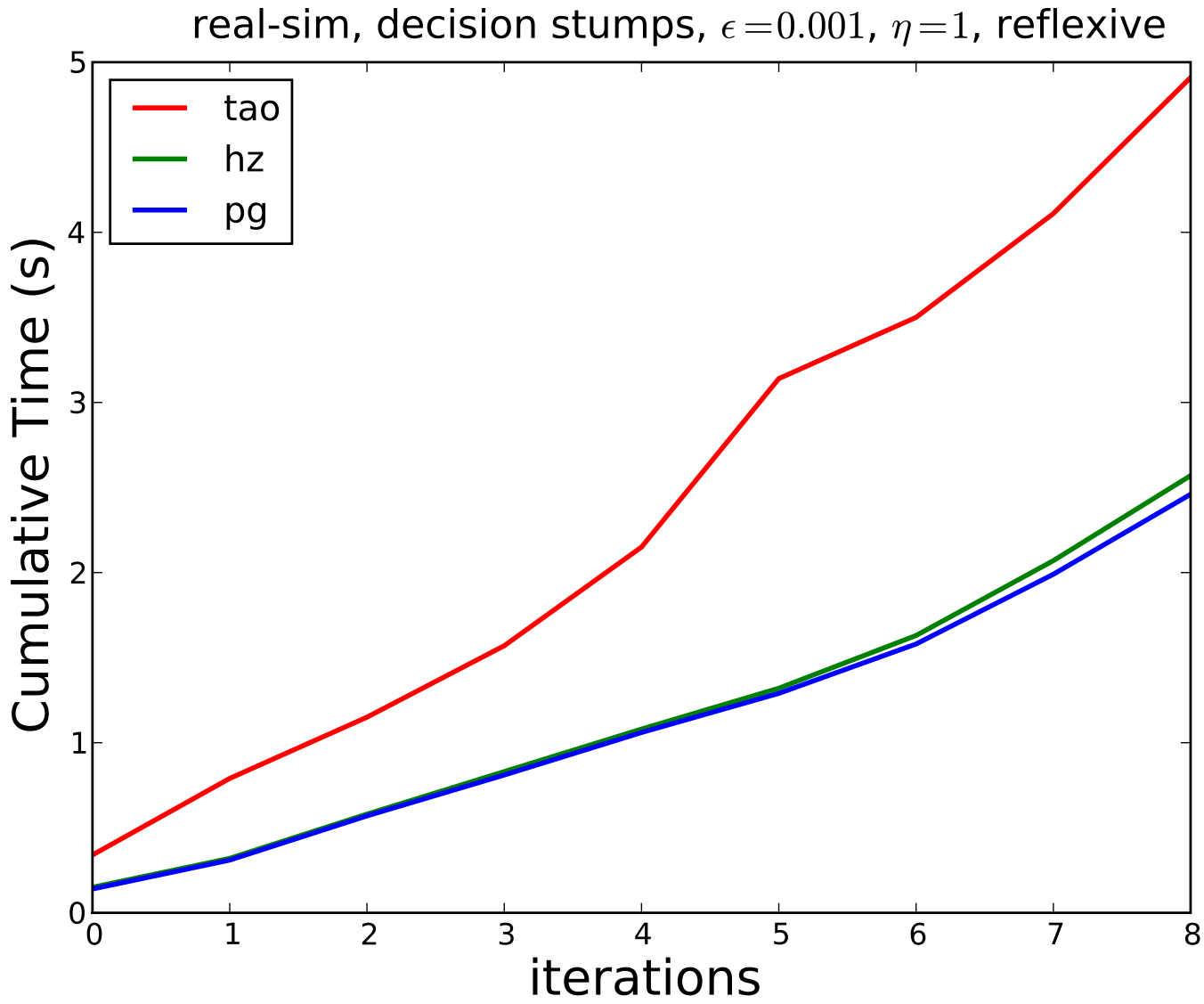
Corrective vs Totally Corrective contd.



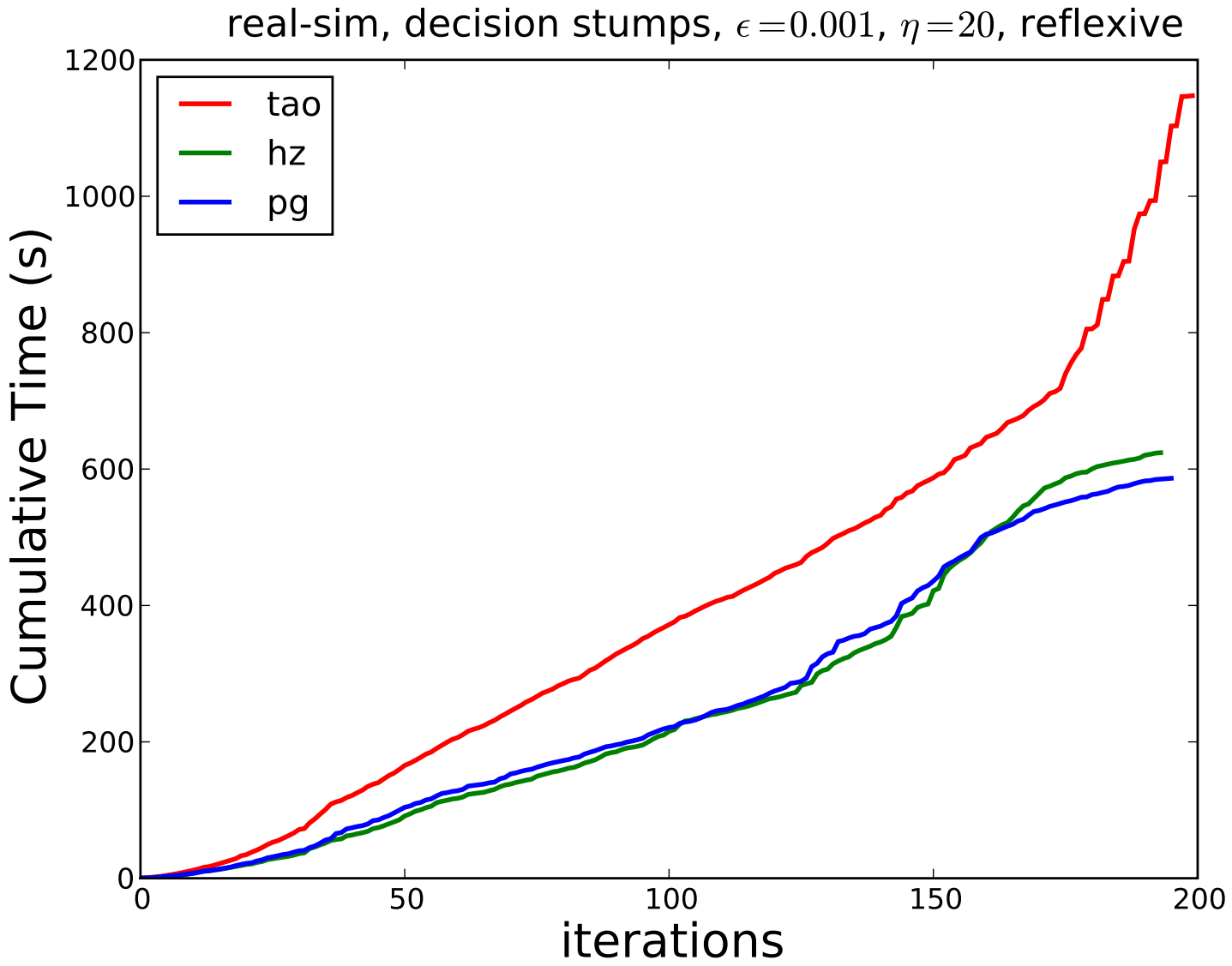
Corrective vs Totally Corrective contd.



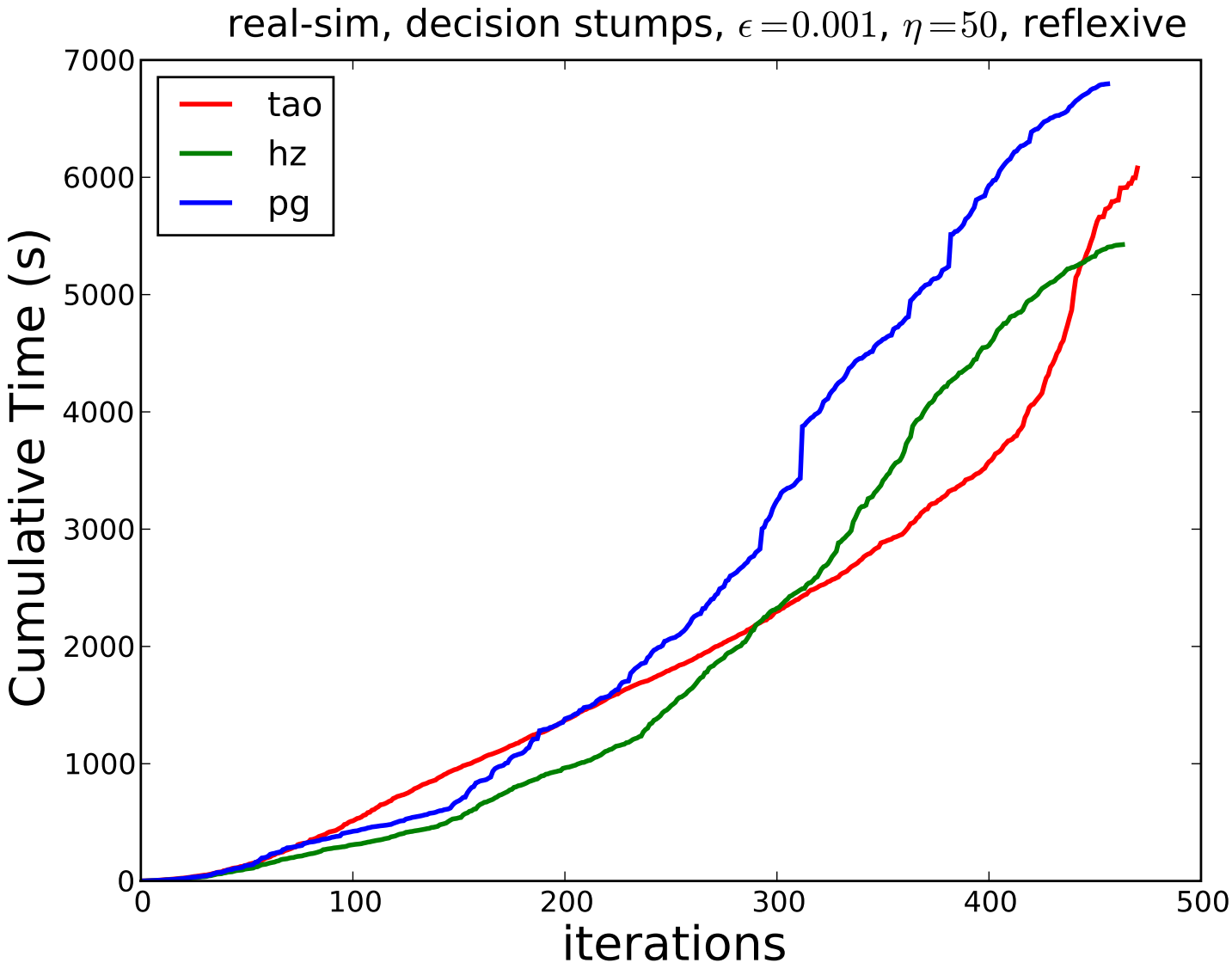
Comparing Different Optimizers



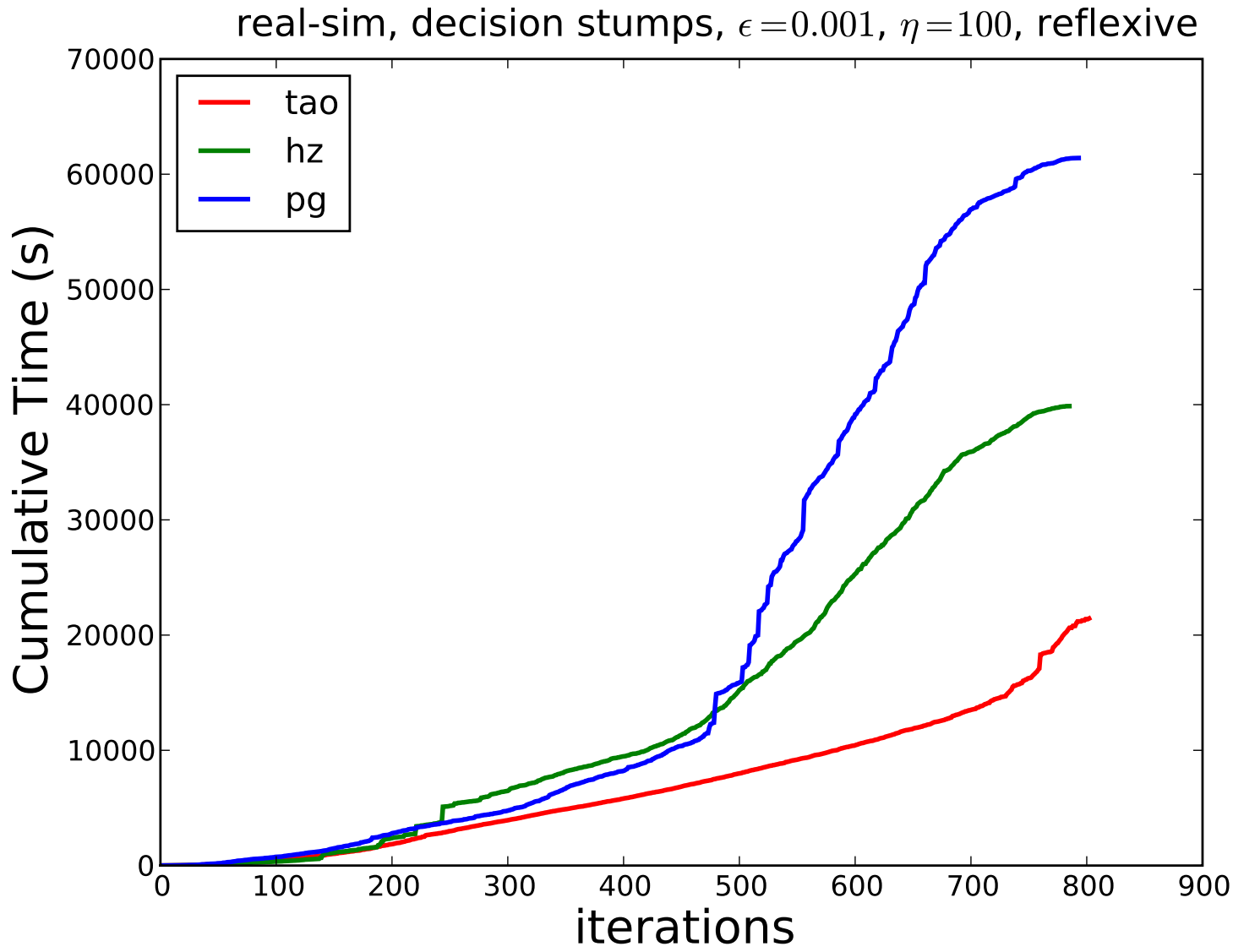
Comparing Different Optimizers



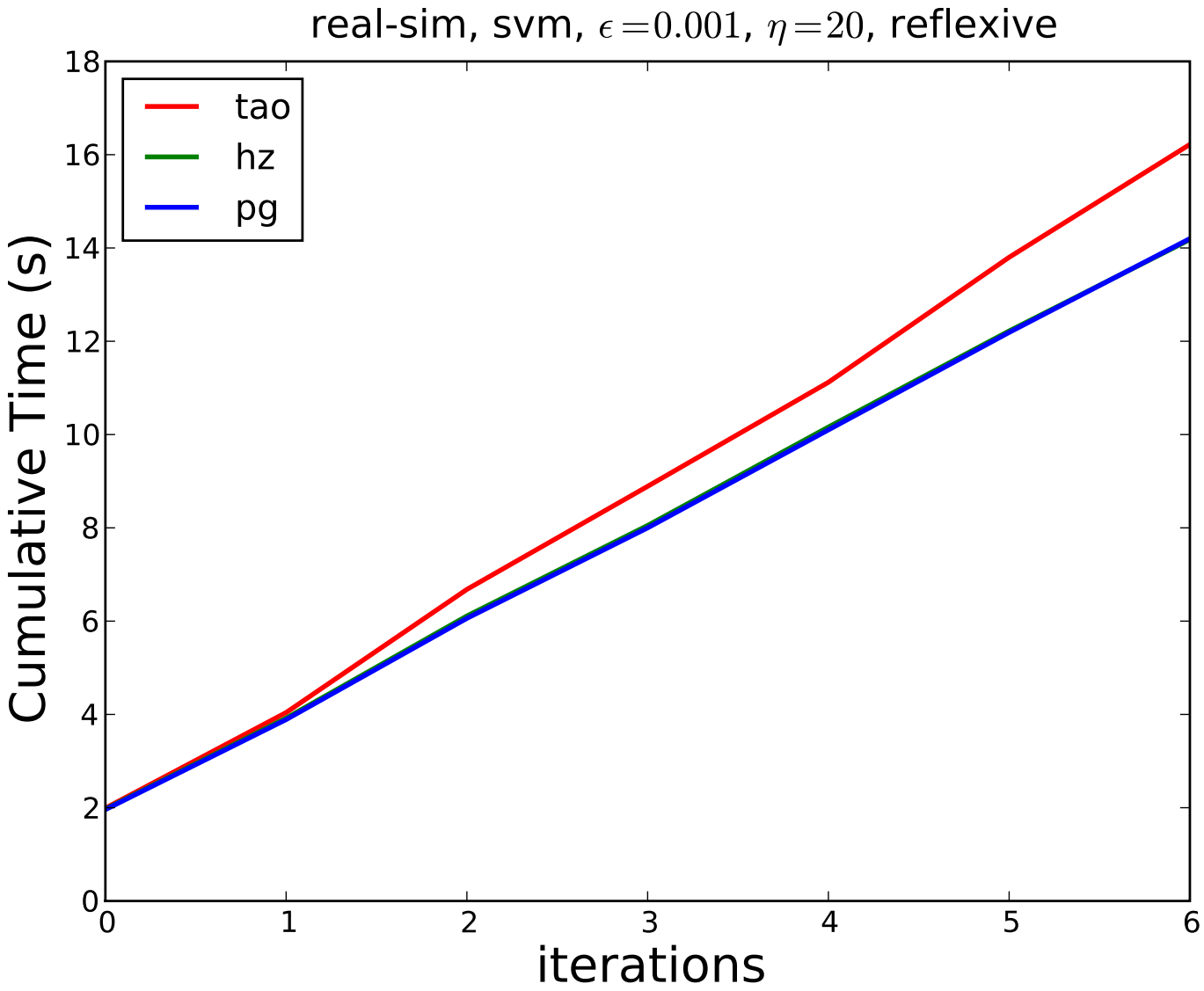
Comparing Different Optimizers



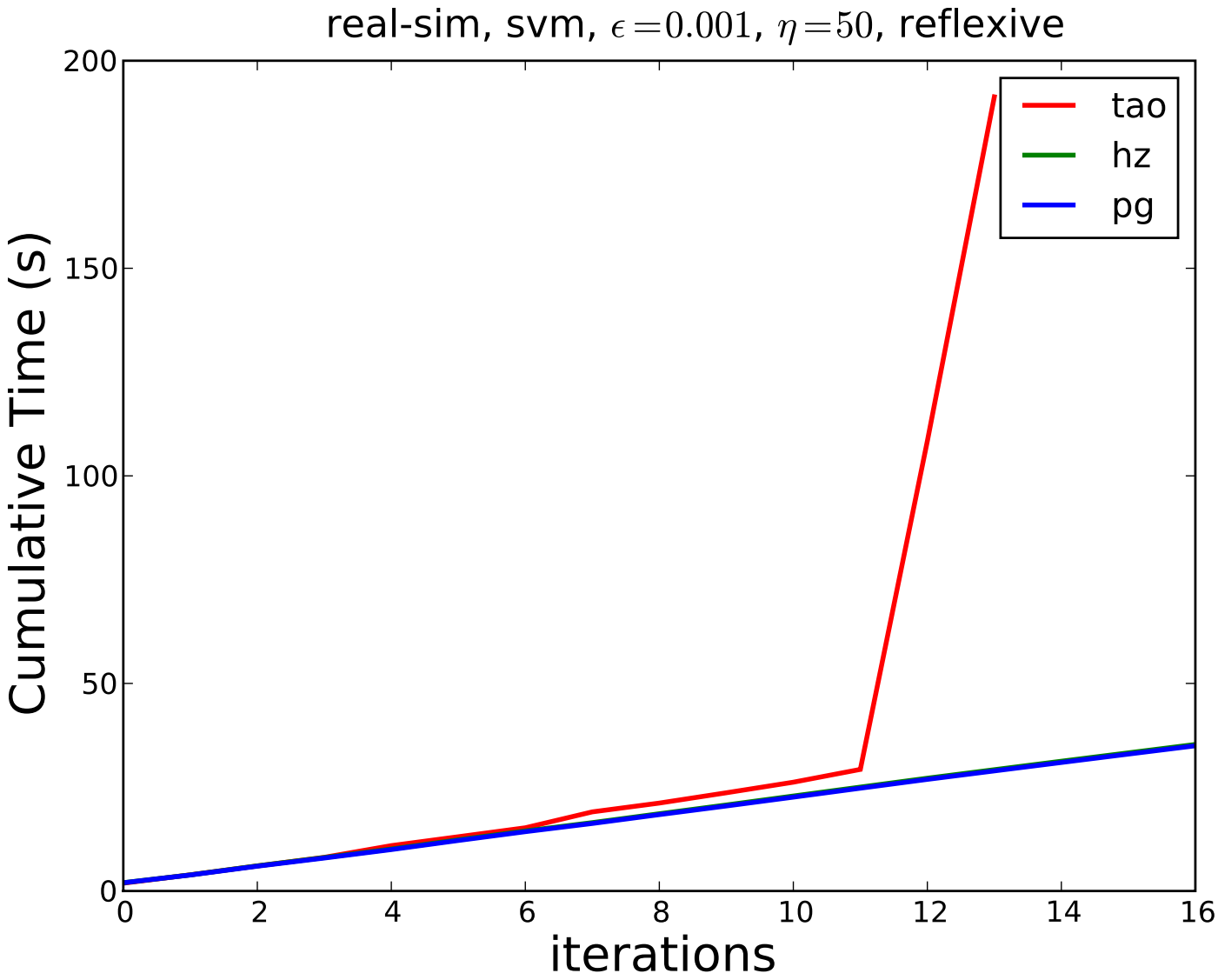
Comparing Different Optimizers



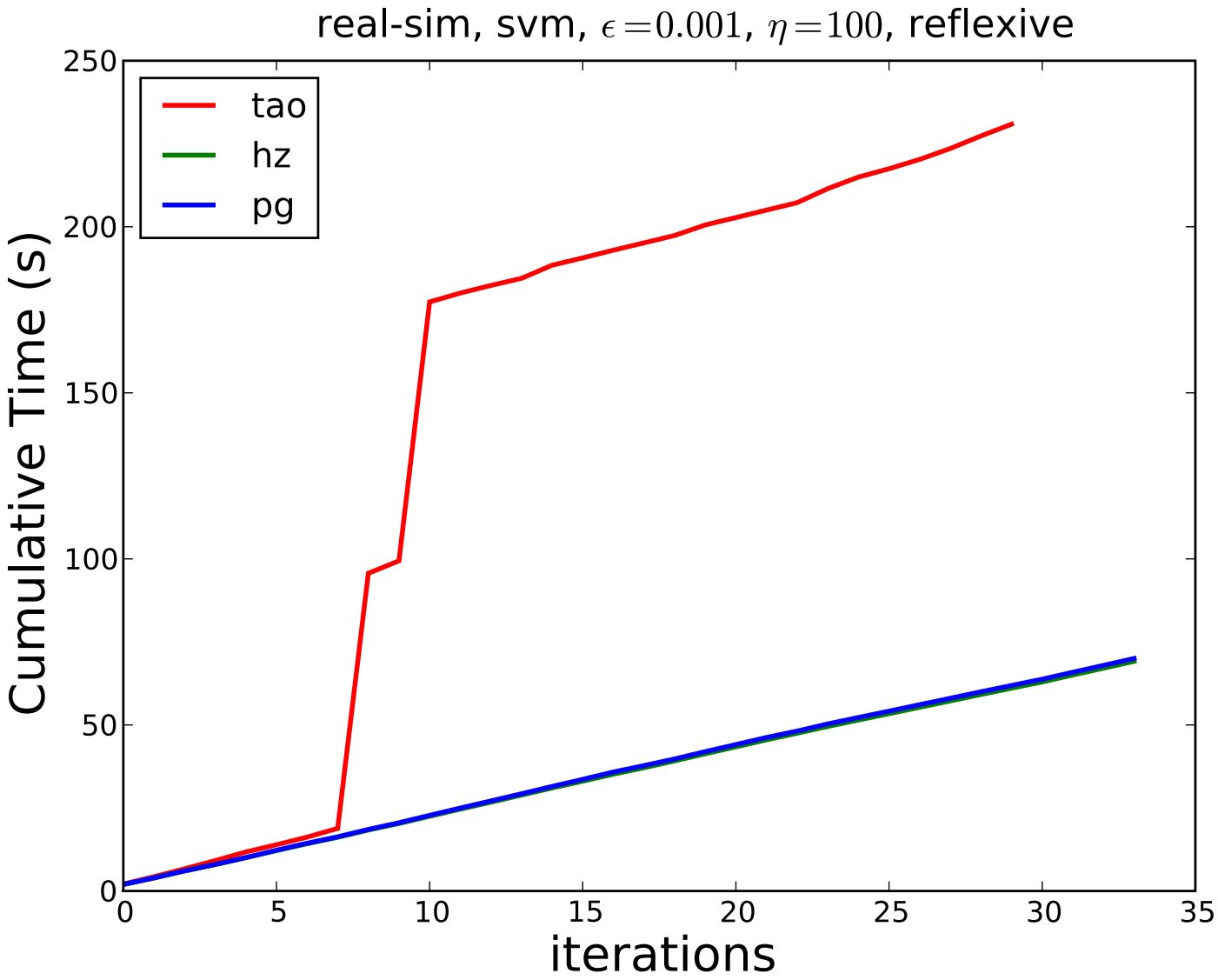
Comparing Different Optimizers



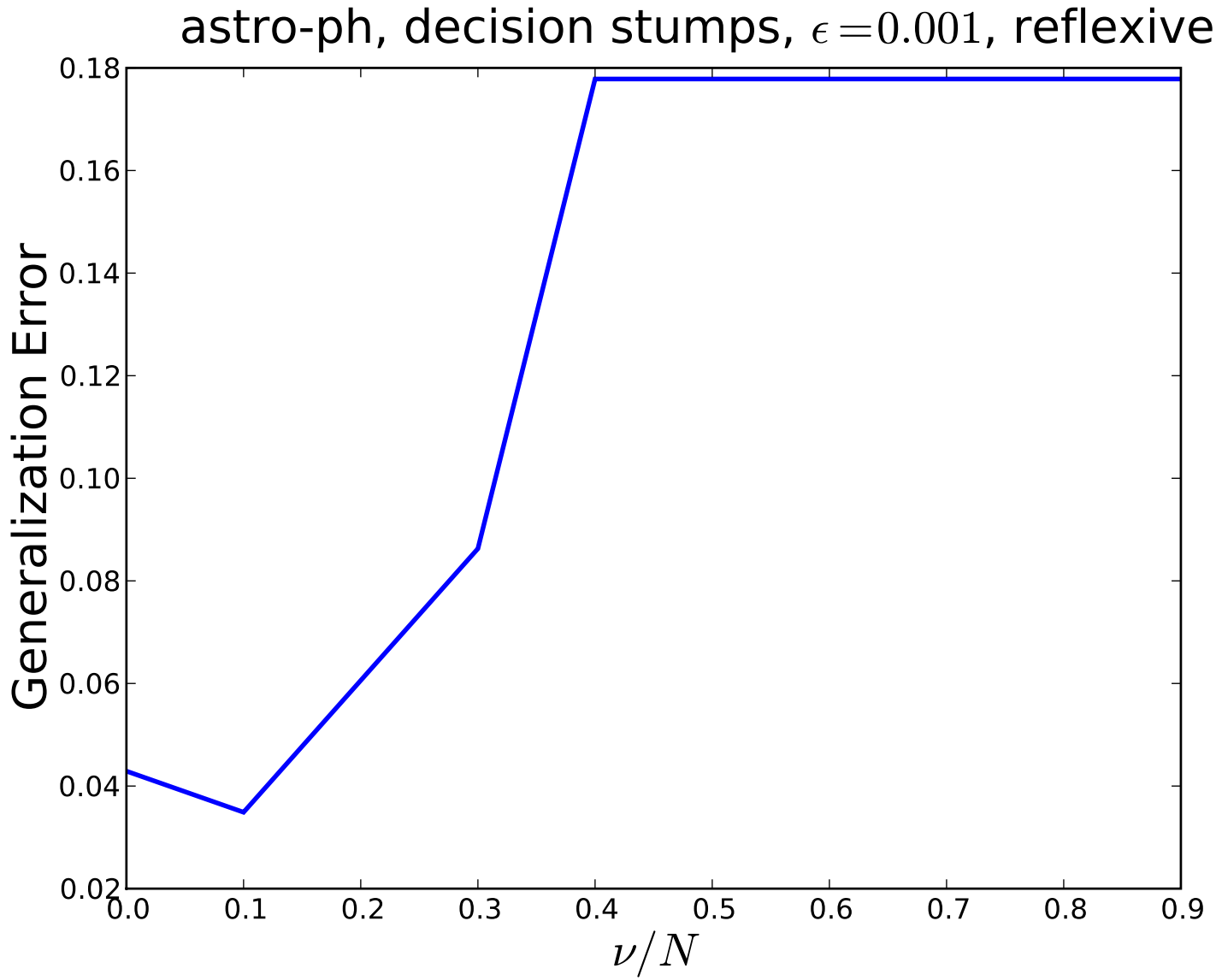
Comparing Different Optimizers



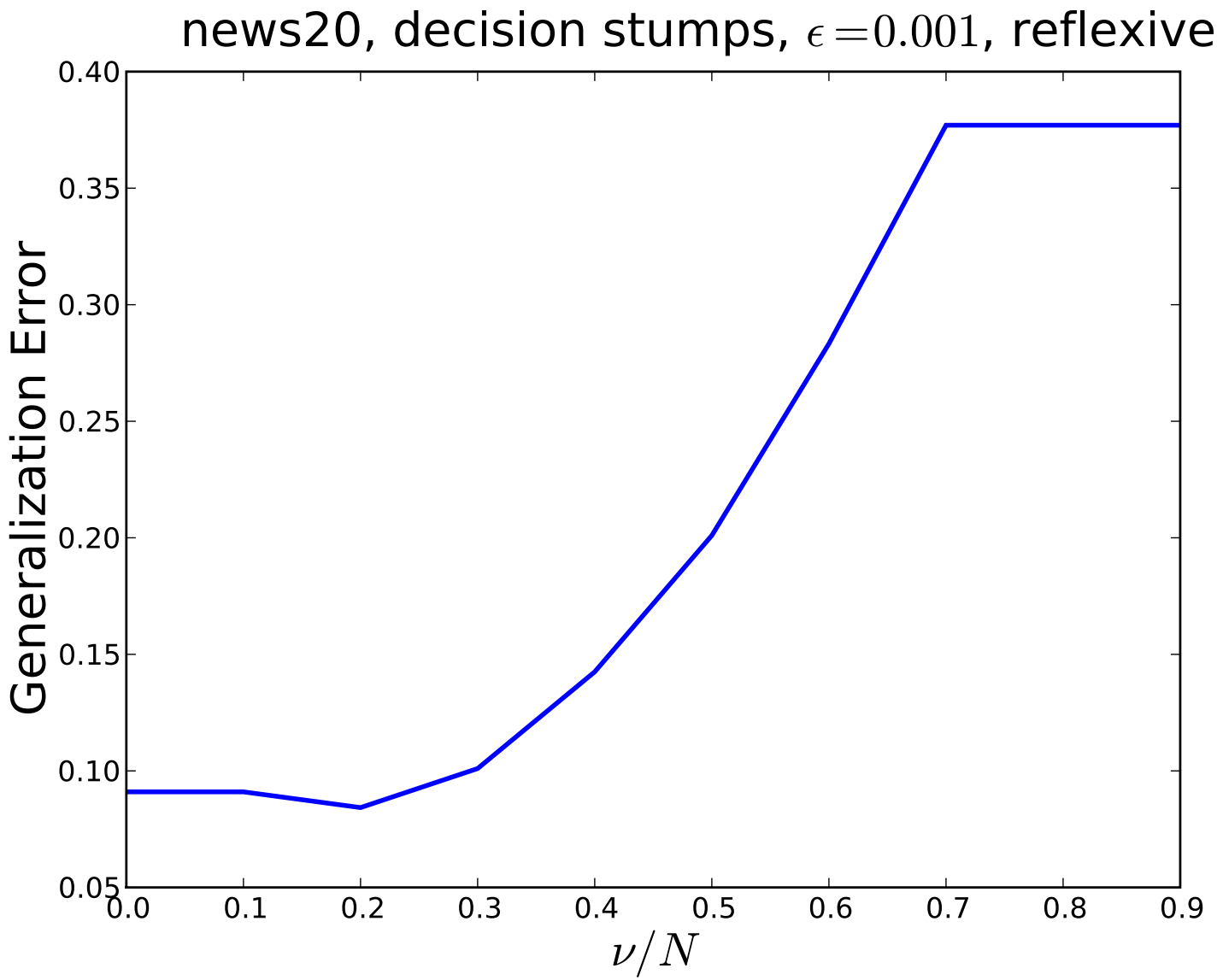
Comparing Different Optimizers



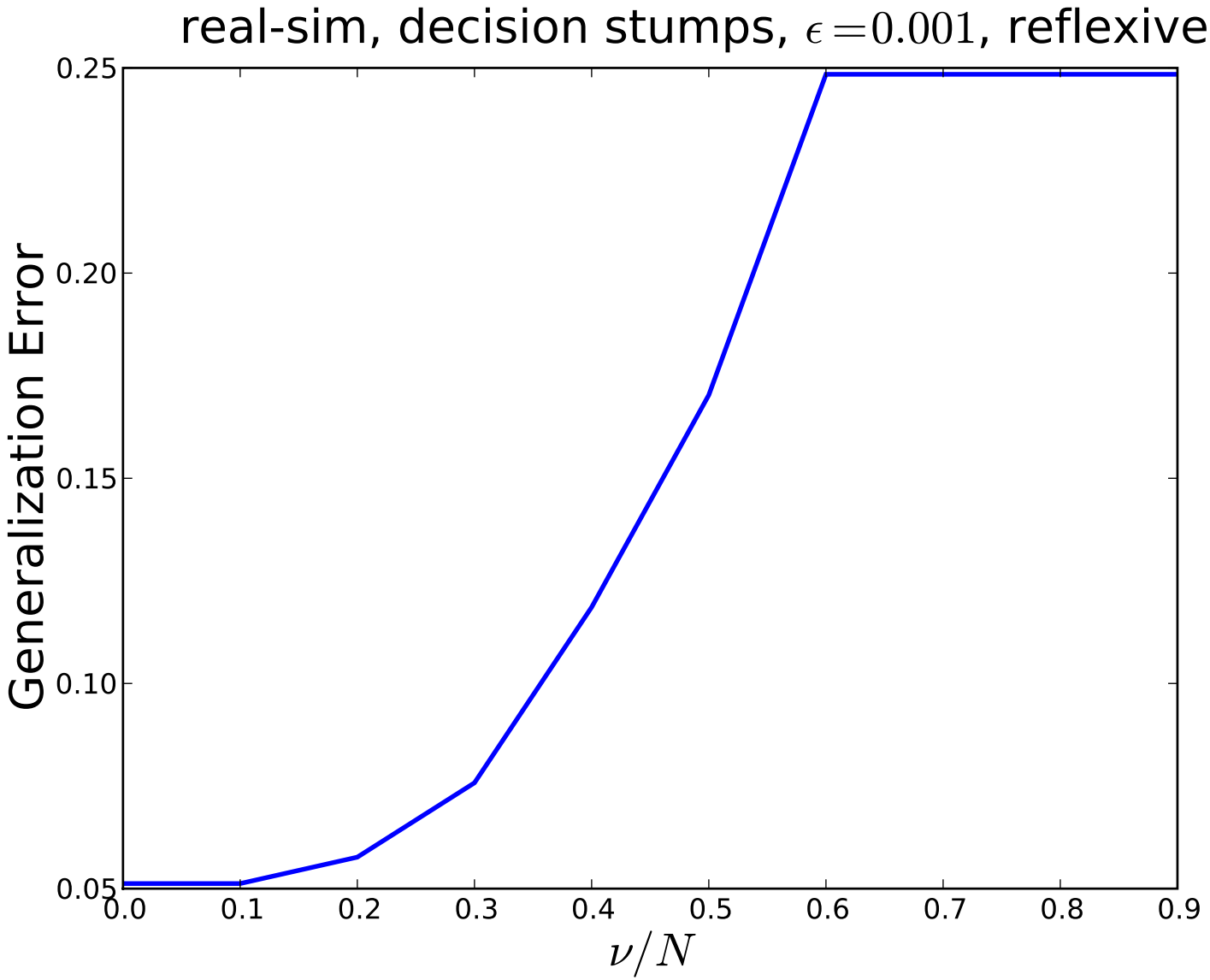
Effect of ν


astro-ph, decision stumps, $\epsilon = 0.001$, reflexive
Generalization Error
0.18
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
ν/N

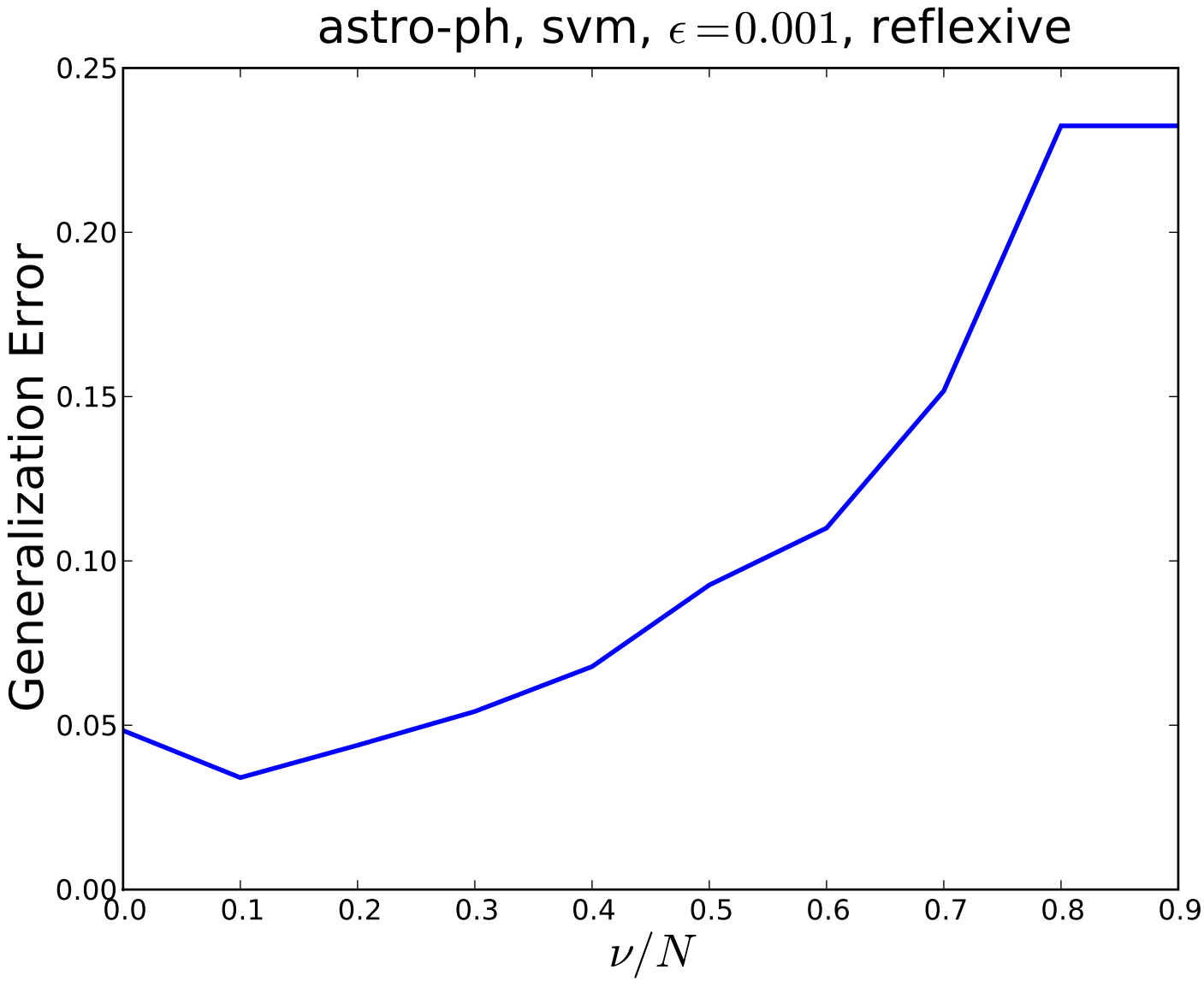
Effect of ν


news20, decision stumps, $\epsilon = 0.001$, reflexive
Generalization Error
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
ν/N

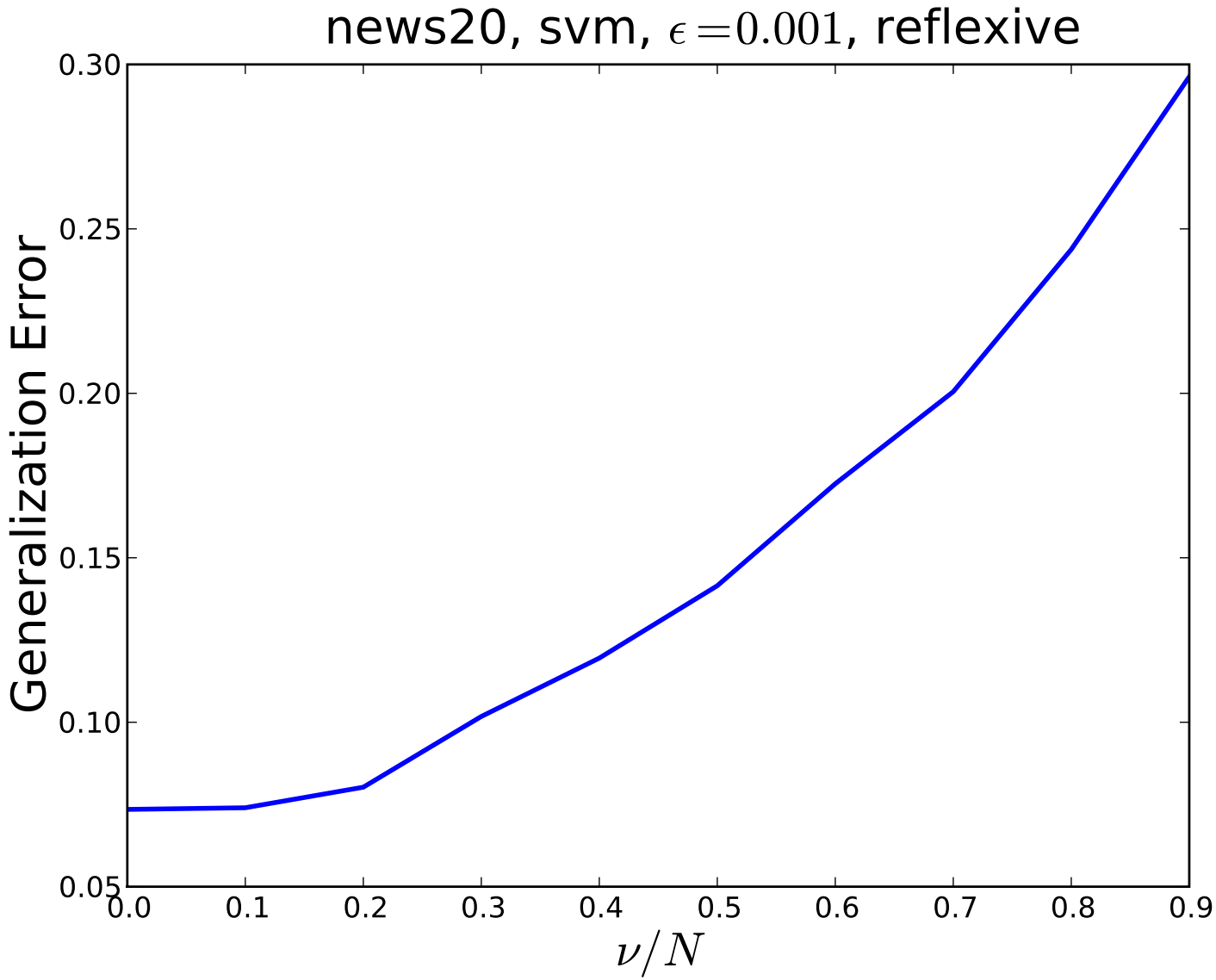
Effect of ν


real-sim, decision stumps, $\epsilon = 0.001$, reflexive
Generalization Error
0.25
0.20
0.15
0.10
0.05
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
ν/N

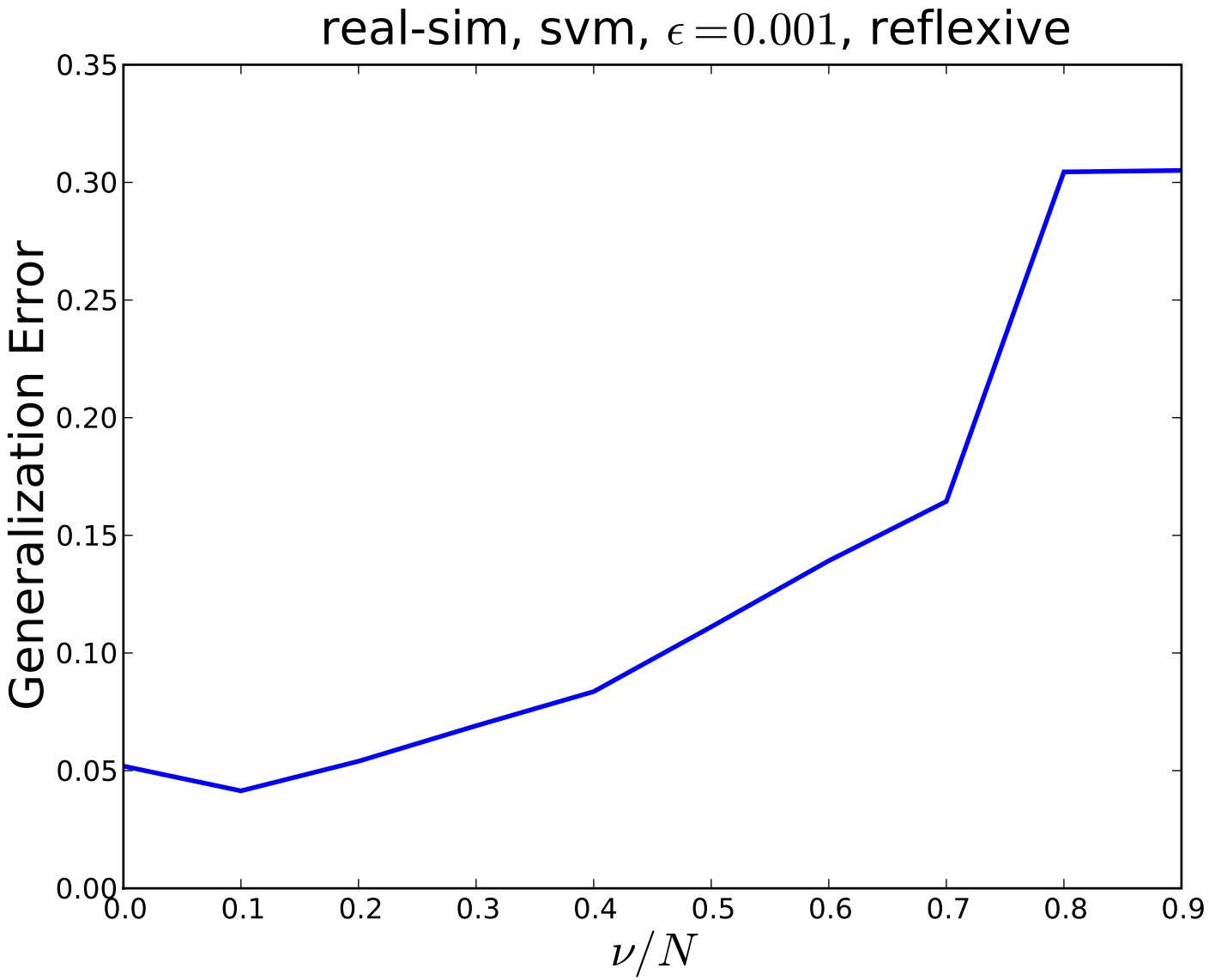
Effect of ν


astro-ph, svm, $\epsilon = 0.001$, reflexive
Generalization Error
0.25
0.20
0.15
0.10
0.05
0.00
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
ν/N

Effect of ν


news20, svm, $\epsilon = 0.001$, reflexive
Generalization Error
0.30
0.25
0.20
0.15
0.10
0.05
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
ν/N

Effect of ν


real-sim, svm, $\epsilon = 0.001$, reflexive
Generalization Error
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
ν/N

SVMs vs Boosting

SVMs

- Simple Quadratic Optimization Problem
- A decade of experience
- Can handle $\approx 10^6$ points

Boosting

- Softmax problem (log, exp, and simplex constraints)
- Just starting out
- Can handle $\approx 10^4$ points

Our Code

- Freely available under the MPL
- Scaling to large datasets (lots of room for improvement)
- Ideas on how to prune weak learners
- Still looking for large datasets where boosting shines ...

SVMs vs Boosting

SVMs

- Simple Quadratic Optimization Problem
- A decade of experience
- Can handle $\approx 10^6$ points

Boosting

- Softmax problem (log, exp, and simplex constraints)
- Just starting out
- Can handle $\approx 10^4$ points

Our Code

- Freely available under the MPL
- Scaling to large datasets (lots of room for improvement)
- Ideas on how to prune weak learners
- Still looking for large datasets where boosting shines ...

SVMs vs Boosting

SVMs

- Simple Quadratic Optimization Problem
- A decade of experience
- Can handle $\approx 10^6$ points

Boosting

- Softmax problem (log, exp, and simplex constraints)
- Just starting out
- Can handle $\approx 10^4$ points

Our Code

- Freely available under the MPL
- Scaling to large datasets (lots of room for improvement)
- Ideas on how to prune weak learners
- Still looking for large datasets where boosting shines ...

SVMs vs Boosting

SVMs

- Simple Quadratic Optimization Problem
- A decade of experience
- Can handle $\approx 10^6$ points

Boosting

- Softmax problem (log, exp, and simplex constraints)
- Just starting out
- Can handle $\approx 10^4$ points

Our Code

- Freely available under the MPL
- Scaling to large datasets (lots of room for improvement)
- Ideas on how to prune weak learners
- Still looking for large datasets where boosting shines ...

SVMs vs Boosting

SVMs

- Simple Quadratic Optimization Problem
- A decade of experience
- Can handle $\approx 10^6$ points

Boosting

- Softmax problem (log, exp, and simplex constraints)
- Just starting out
- Can handle $\approx 10^4$ points

Our Code

- Freely available under the MPL
- Scaling to large datasets (lots of room for improvement)
- Ideas on how to prune weak learners
- Still looking for large datasets where boosting shines ...

SVMs vs Boosting

SVMs

- Simple Quadratic Optimization Problem
- A decade of experience
- Can handle $\approx 10^6$ points

Boosting

- Softmax problem (log, exp, and simplex constraints)
- Just starting out
- Can handle $\approx 10^4$ points

Our Code

- Freely available under the MPL
- Scaling to large datasets (lots of room for improvement)
- Ideas on how to prune weak learners
- Still looking for large datasets where boosting shines ...

SVMs vs Boosting

SVMs

- Simple Quadratic Optimization Problem
- A decade of experience
- Can handle $\approx 10^6$ points

Boosting

- Softmax problem (log, exp, and simplex constraints)
- Just starting out
- Can handle $\approx 10^4$ points

Our Code

- Freely available under the MPL
- Scaling to large datasets (lots of room for improvement)
- Ideas on how to prune weak learners
- Still looking for large datasets where boosting shines ...

SVMs vs Boosting

SVMs

- Simple Quadratic Optimization Problem
- A decade of experience
- Can handle $\approx 10^6$ points

Boosting

- Softmax problem (log, exp, and simplex constraints)
- Just starting out
- Can handle $\approx 10^4$ points

Our Code

- Freely available under the MPL
- Scaling to large datasets (lots of room for improvement)
- Ideas on how to prune weak learners
- Still looking for large datasets where boosting shines ...

Conclusion

- Lots of exciting connections between boosting and optimization (we are only scratching the surface)
- Bring entropic regularization algorithms up to par with squared Euclidean distance regularization
- Look for datasets that exploit merits of new algorithms
- Find artificial datasets that highlight advantages of different families of algorithms
- Better lower bounds (the case of the missing $\log n$)

Acknowledgments

- Rob Schapire and Yoav Freund for pioneering Boosting
- Gunnar Rätsch for bringing in optimization
- Karen Glocer for helping with figures and plots
- Vishy for teaching me the latest optimization tricks