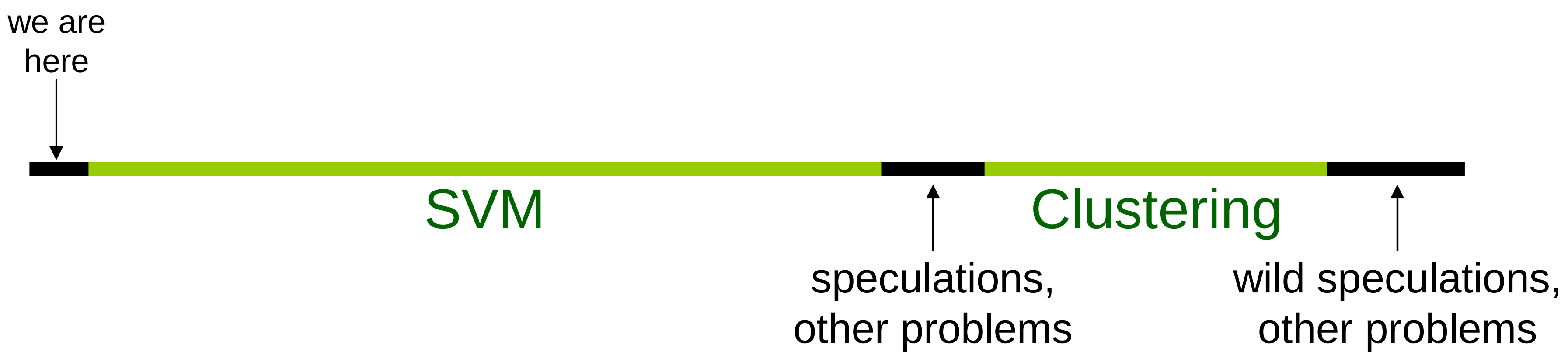
More Data, Less Work:

Runtime as a decreasing function of data set size

Nati Srebro

Toyota Technological Institute—Chicago

Outline



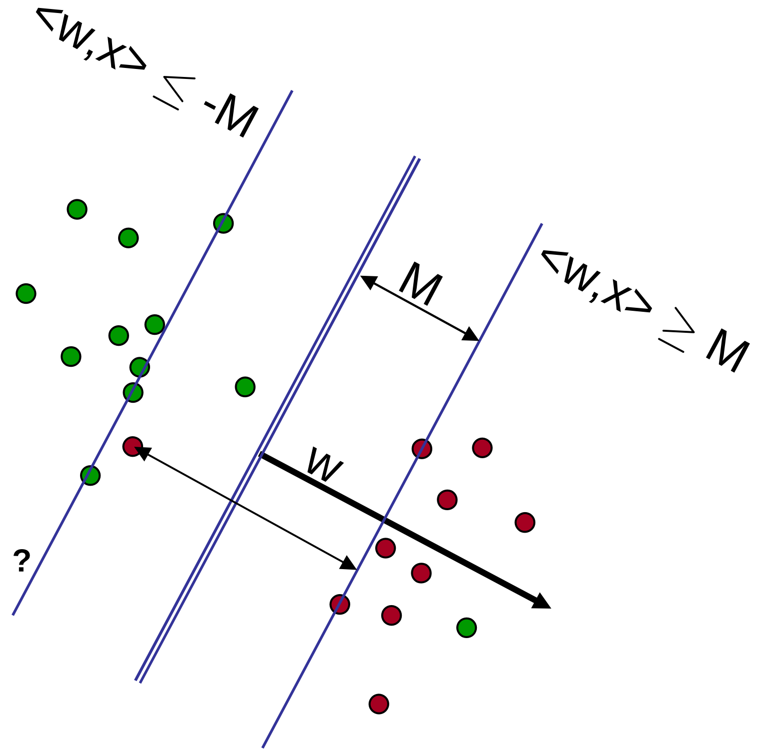
•SVM Optimization: Inverse Dependence on Training Set Size
Shai Shalev-Shwartz (TTI), N Srebro

•Fast Rates for Regularized Objectives
Karthik Sridharan (TTI), **Shai Shalev-Shwartz** (TTI), N Srebro, NIPS'08

•Pegasos: Primal Estimated sub-GrAdient SOlver for SVM
Shai Shalev-Shwartz (TTI), **Yoram Singer** (Google), N Srebro, ICML'07

•An Investigation of Comp. and Informational Limits in Gaussian Mixture Clustering
N Srebro, **Greg Shakhnarovich** (TTI), **Sam Roweis** (Google/Toronto), ICML'06

Large Margin Linear Classification
aka L_2-regularized Linear Classification
aka Support Vector Machines



$|w|=1$

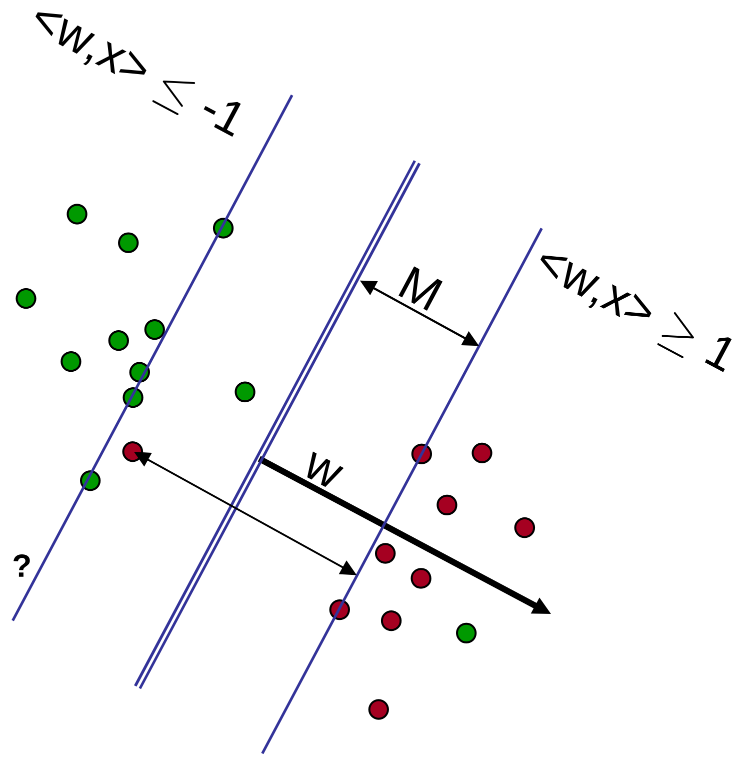
Error: $[1-y<w,x>]_+$

Large Margin Linear Classification

aka L_2-regularized Linear Classification

aka Support Vector Machines

$$f(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n}\left[1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle\right]_+$$



Margin: M = 1/|w|

Error: $[1-y\langle w,x\rangle]_+$

SVM Training as an Optimization Problem

$$f(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n}\left[1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle\right]_+$$

- IP method on dual (standard QP solver):

  $O(n^{3.5} \log \log(1/\varepsilon))$

- Dual decomposition methods (e.g. SMO):

  $O(n^2 d \log(1/\varepsilon))$ **[Platt 98][Joachims 98][Lin 02]**

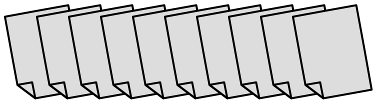
- Primal cutting plane method (SVMperf):

  $O(nd / (\lambda\varepsilon))$ **[Joachims 06][Smola et al 08]**

Runtime to get $f(w) \leq \min f(w) + \varepsilon$

More Data ⇒ More Work?

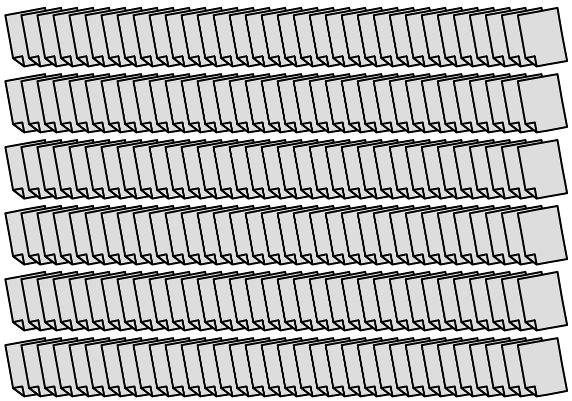
10k training examples



1 hour

2.3% error
(when using the predictor)

1M training examples



1 week (or more…)

2.29% error

Can always sample and get same runtime:

1 hour 2.3% error

Can we leverage the excess data to **reduce** runtime?

10 minutes 2.3% error

But I really care about that 0.01% gain

Study runtime increase as a function of target accuracy

My problem is so hard, I *have* to crunch 1M examples

Study runtime increase as a function of problem difficulty (e.g. small margin)

SVM Training

- Optimization objective:

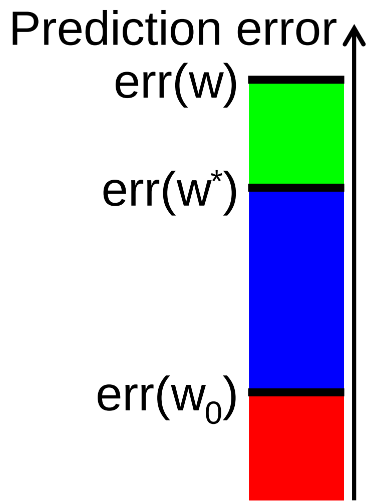
$$f(\mathbf{w}) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n}[1 - y_i \langle \mathbf{w}, \mathbf{x}_i \rangle]_+$$

- True objective: prediction error on future examples

  err(w) = $\mathbf{E}_{x,y}$[error of w'x vs. y]

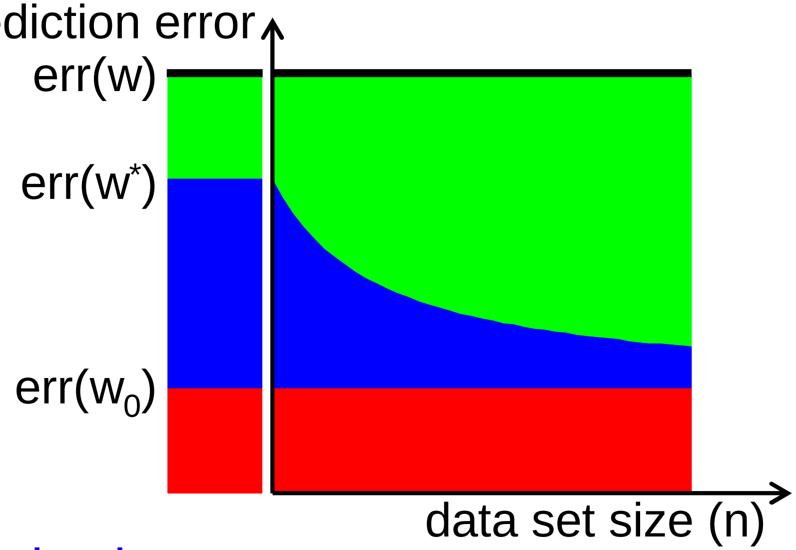
- Would like to understand computational cost in terms of:
- Increasing function of:
  - Desired generalization performance (i.e. as err(w) decreases)
  - Hardness of problem:
    margin, noise (unavoidable error)
- Decreasing function of available data set size

Error Decomposition



- Approximation error:
  - Best error achievable by large-margin predictor
  - Error of population minimizer
    $w_0 = \text{argmin } E[f(w)] = \text{argmin } \lambda|w|^2 + E[\text{loss}(w)]$
- Estimation error:
  - Extra error due to replacing E[loss] with empirical loss
    $w^* = \text{arg min } f_n(w) = \text{arg min } \lambda\,|w|^2 + \text{loss}(w \text{ on training set})$
- Optimization error:
  - Extra error due to only optimizing to within finite precision

The Double-Edged Sword
Error Decomposition



- When data set size increases:
  - **Estimation error** decreases
  - Can increase **optimization error**, i.e. optimize to within lesser accuracy ⇒ fewer iterations

  - But handling more data is expensive e.g. runtime of each iteration increases

- PEGASOS (Primal Efficient Sub-Gradient Solver for SVMs)

[Shalev-Shwartz Singer S 07]

  - Fixed runtime per iteration
  - Runtime to get fixed accuracy does not increase with n

PEGASOS: Stochastic (sub-)Gradient Descent

$$f(\mathbf{w}) = \lambda\|\mathbf{w}\|^2 + \frac{1}{n}\sum_{i=1}^{n}[1 - y_i\langle\mathbf{w},\mathbf{x}_i\rangle]_+$$

- Initialize w=0
- At each iteration t,
  with random data point (x_i,y_i): $\nabla = 2\lambda\mathbf{w} - \begin{cases} y_i\mathbf{x}_i & \text{if } y_i\langle w,\mathbf{x}_i\rangle < 1 \\ 0 & \text{otherwise} \end{cases}$

  subgradient of $\lambda|w|^2+[1-y_i<w,x_i>]_+$

  $$\mathbf{w} \leftarrow \mathbf{w} - \frac{1}{2\lambda t}\nabla$$

- **Theorem**: After at most $\tilde{O}\left(\frac{1}{\delta\,\lambda\,\epsilon}\right)$ iterations, $f(w_{PEGASOS}) \leq \min_w f(w)+\varepsilon$, with probability $\geq 1-\delta$
- With d-dimensional (or d-sparse) features, each iteration takes time O(d)
- **Conclusion**: Run-time required for PEGASOS to find ε accurate solution with constant probability:

$$\tilde{O}\left(\frac{d}{\lambda\,\epsilon}\right)$$

- Run-time does not depend on #examples

Training Time (in seconds)

	Pegasos	SVM-Perf [Joachims06]	SVM-Light [Joachims]
Reuters CCAT (800K examples, 47k features)	**2**	77	20,075
Covertype (581k examples, 54 features)	**6**	85	25,514
Physics ArXiv (62k examples, 100k features)	**2**	5	80

Runtime Analyzis

	Traditional $f(w)<f(w^*)+\varepsilon_{acc}$	Data Laden: $err(w) \leq err(w_0)+\varepsilon$
Interior Point	$\mathbf{n}^{3.5} \log(\log(1/\boldsymbol{\varepsilon}_{\mathbf{acc}}))$	$\lvert w_0\rvert^7/\varepsilon^7$
SMO	$\mathbf{n}^2\, d \log(1/\boldsymbol{\varepsilon}_{\mathbf{acc}})$	$d\, \lvert w_0\rvert^4/\varepsilon^4$
SVMPerf	$\mathbf{n}\, d / (\lambda\, \boldsymbol{\varepsilon}_{\mathbf{acc}})$	$d\, \lvert w_0\rvert^4/\varepsilon^4$
PEGASOS	$d / (\lambda\, \boldsymbol{\varepsilon}_{\mathbf{acc}})$	$d\, \lvert w_0\rvert^2/\varepsilon^2$

(ignoring log-factors)

large margin $M=1/\lvert w_0\rvert$

If there is some predictor w_0 with low $\lvert w_0\rvert$ and low $err(w_0)$, how much time to find predictor with $err(w) \leq err(w_0)+\varepsilon$

To get $err(w) \leq err(w_0)+O(\varepsilon)$:

$\lambda = O(\varepsilon/\lvert w_0\rvert^2)$

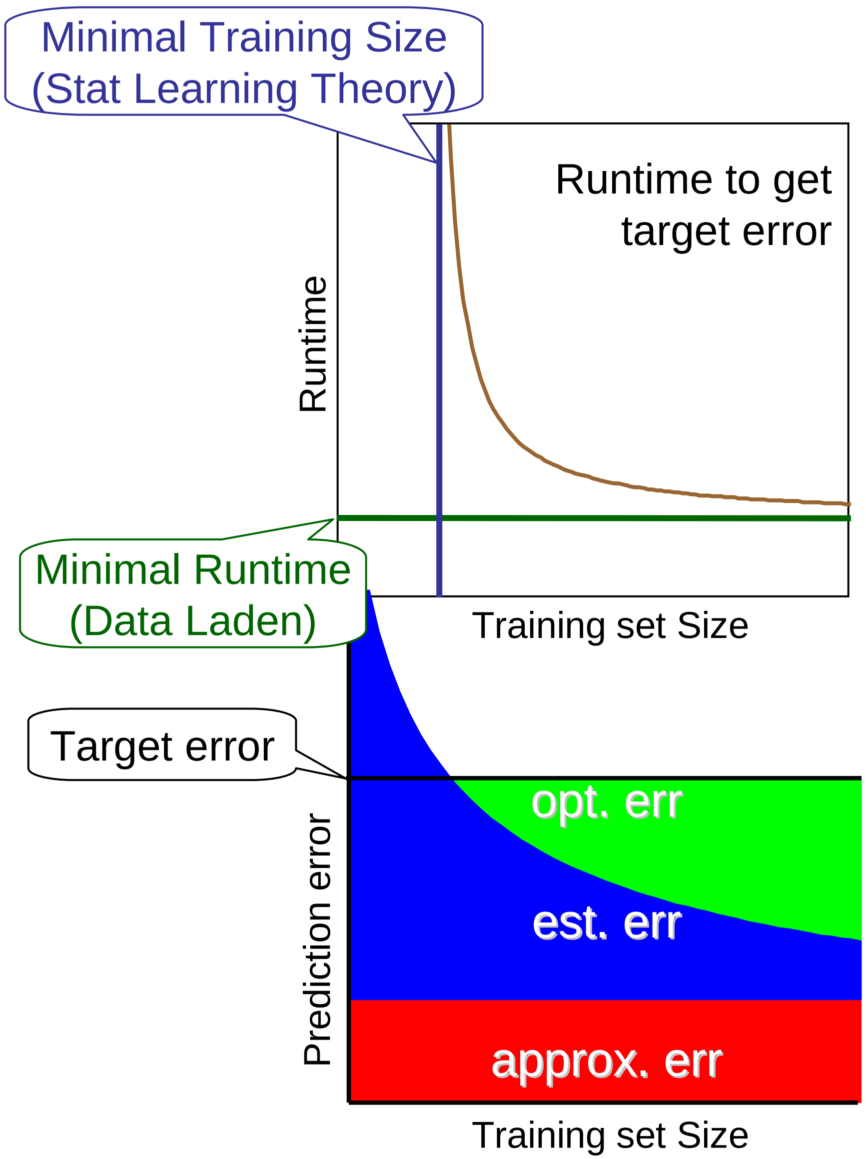
$\varepsilon_{acc} = O(\varepsilon)$

$n = \Omega(1/(\lambda\, \varepsilon)) = \Omega(\lvert w_0\rvert^2/\varepsilon^2)$

Unlimited data available, can choose working data-set size

Data Laden analysis: Restricted by computation, not data

Dependence on Data Set Size



PEGASOS guaranteed runtime to get error err(w_0)+ε with n training points:

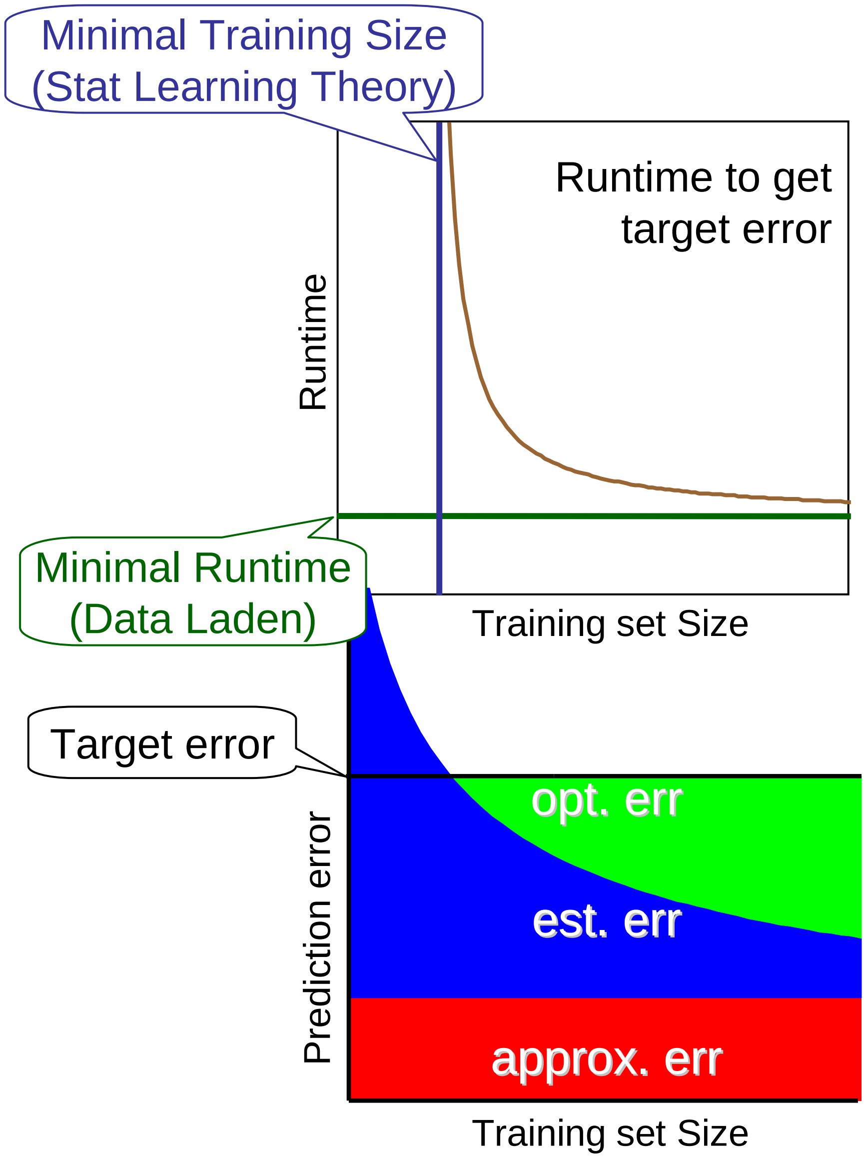
$$T = \tilde{\Omega}\left(\frac{d}{\left(\epsilon M - O\left(\frac{1}{\sqrt{n}}\right)\right)^2}\right)$$

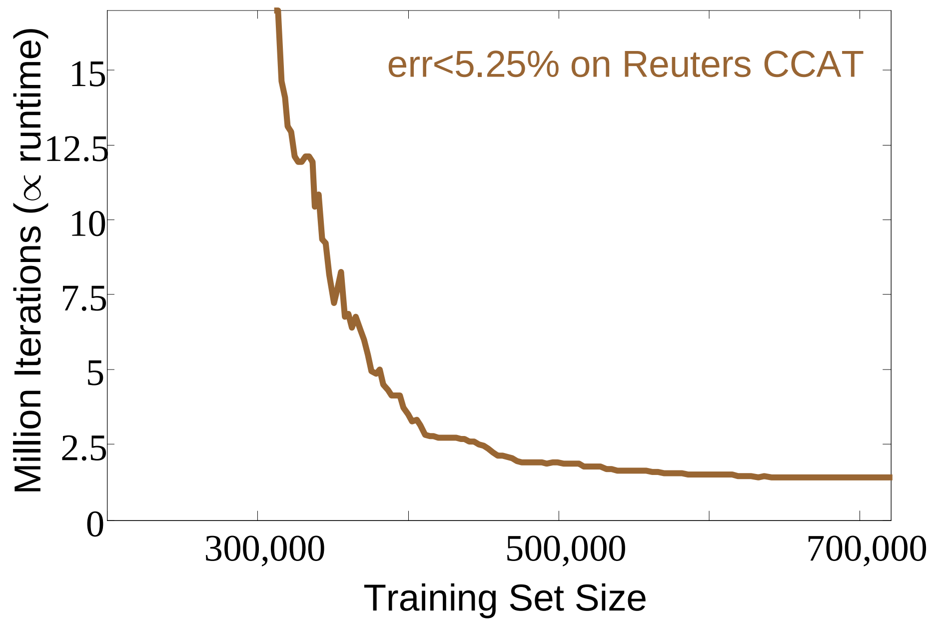
Increases for smaller target error

Increases for smaller margin

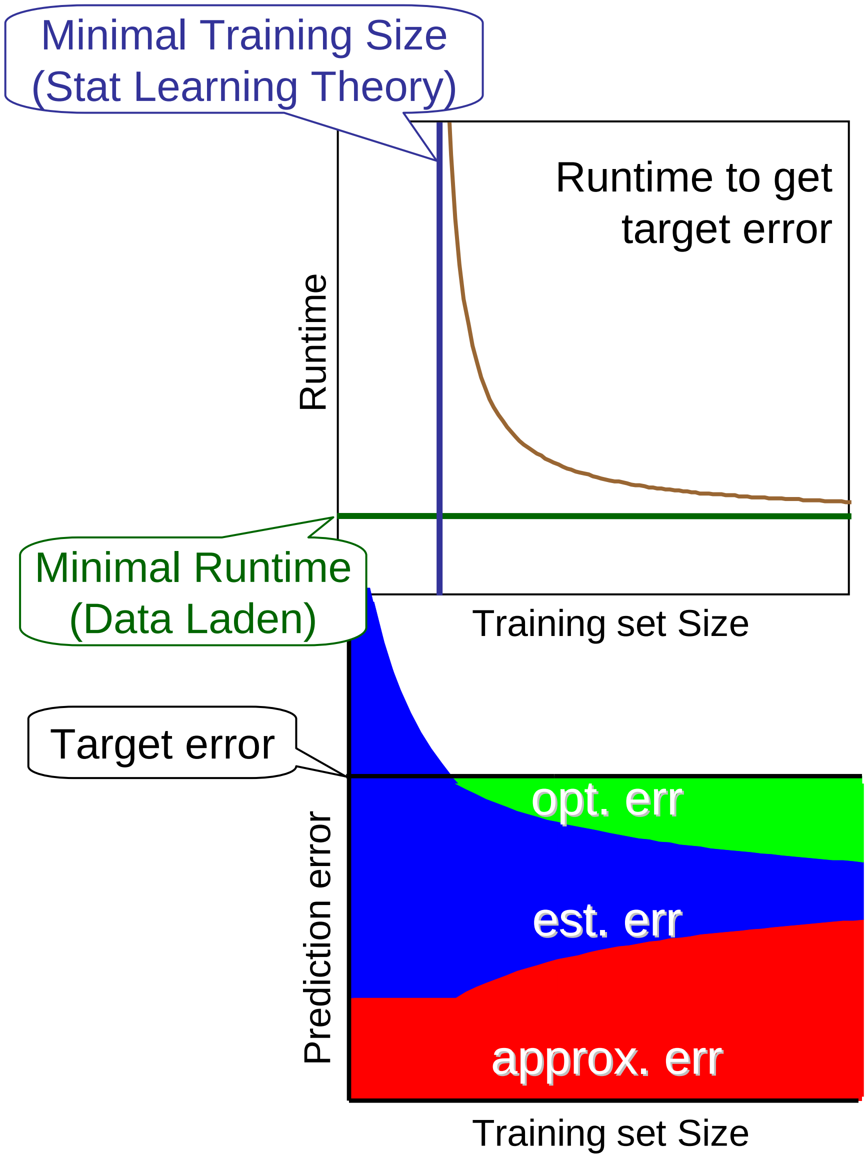
Decreases for larger data set

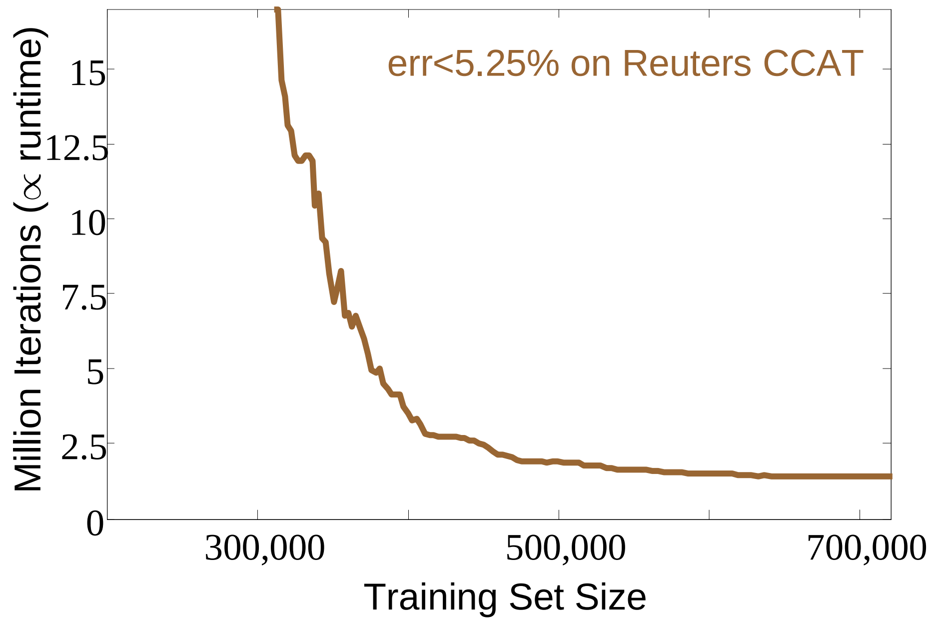
Dependence on Data Set Size





Dependence on Data Set Size



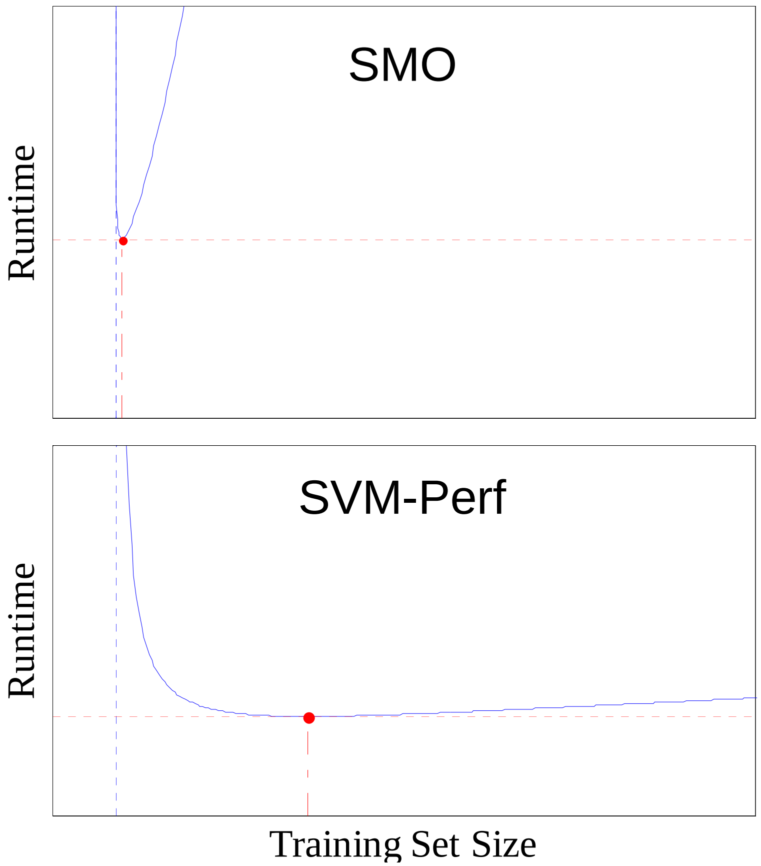


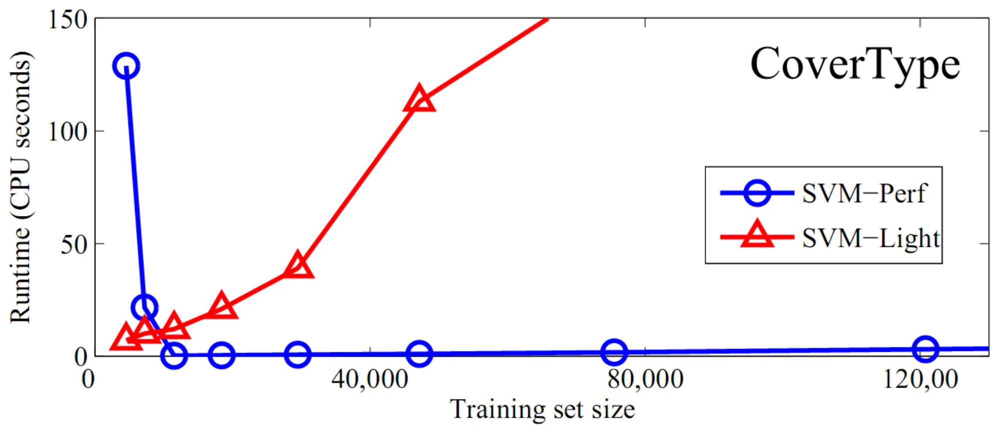
$$\text{err}(w) \leq \text{err}(w_0) + \lambda|w_0|^2 + O(1/(\lambda n)) + O(d/(\lambda T))$$

Increase λ as training size increases!

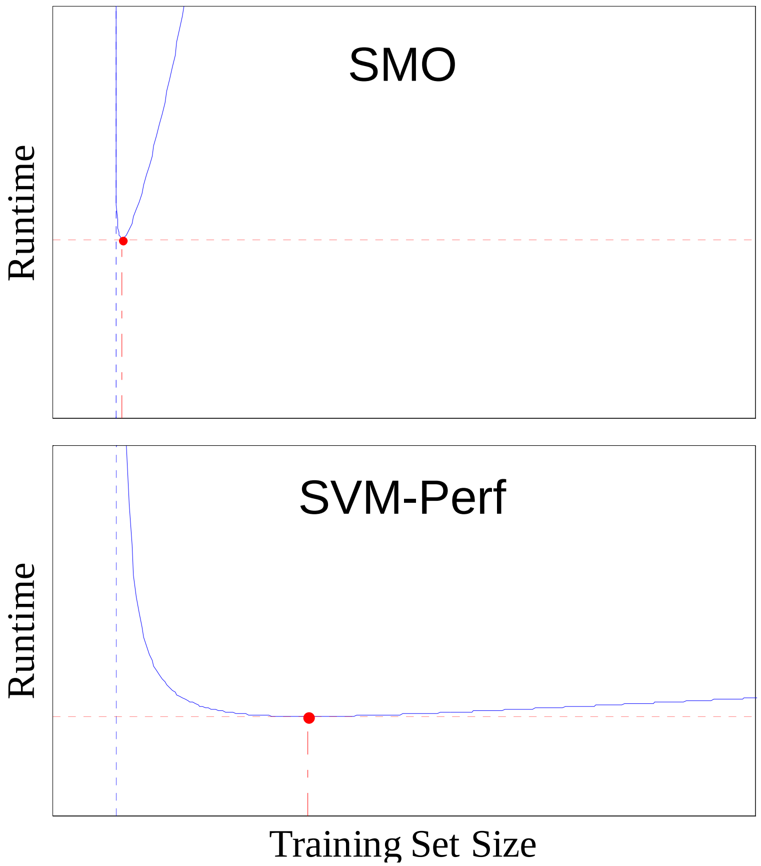
- More regularization, less predictors allowed
- Larger approximation error $\text{err}(w_0)+\lambda|w_0|^2$
- Faster runtime $T \propto 1/\lambda$

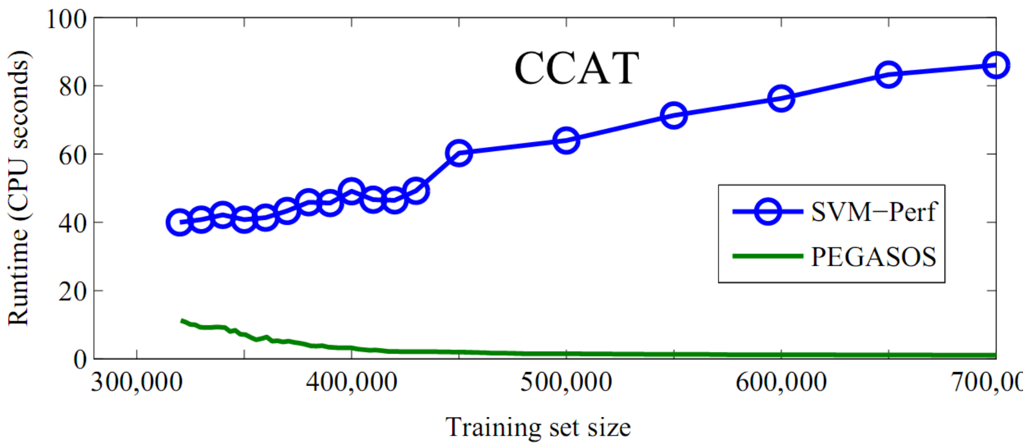
Dependence on Data Set Size: Traditional Optimization Approaches





Dependence on Data Set Size: Traditional Optimization Approaches





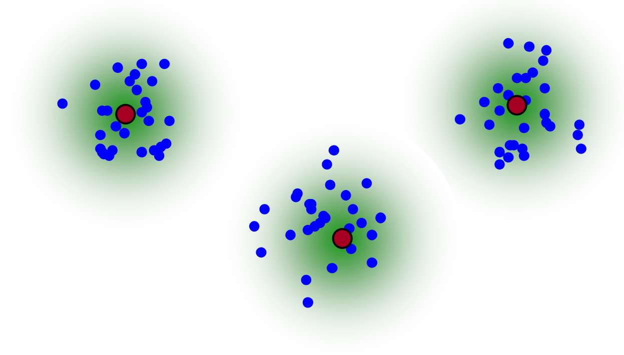
Beyond PEGASOS

- Other machine learning problems
  - Kernalizes SVMs
  - L_1-regularization, e.g. LASSO
  - Matrix/factor models (e.g. with trace-norm regularization)
  - Multilayer / deep networks

  …

- Can we more explicitly leverage excess data?
  - Playing only on the error decomposition, const $\times$ minimum-sample-complexity is enough to get to const $\times$ minimum-data-laden-runtime

Clustering
(by fitting a Gaussian mixture model)



- Find centers ($\mu_1,\ldots,\mu_k$) minimizing objective:
  - Negative log-likelihood under Gaussian mixture model:

$$-\Sigma_i \log(\Sigma_j \exp -(x_i-\mu_j)^2/2)$$

  - *k*-means objective ≈ negative log-likelihood of assignment:

$$\Sigma_i \min_j (x_i-\mu_j)^2$$

Clustering (by fitting a Gaussian mixture model)

- Clustering is hard in the worst-case
- Given LOTS of data and HUGE separation:
  - Can efficiently recover true clustering **[Dasgupta 99][Dasgupta Schulman 00][Arora Kannan 01][Vempala Wang 04] [Achliopts McSherry 05][Kannan Salmasian Vempala 05]**
  - EM works (empirically)
- With too little data, clustering is meaningless:
  - Even if we find the ML clustering, it has nothing to do with underlying distribution

"Clustering isn't hard—
it's either easy, or not interesting"

Effect of “Signal Strength”

Larger data set

Lots of data—
true solution creates
distinct peak.
Easy to find.

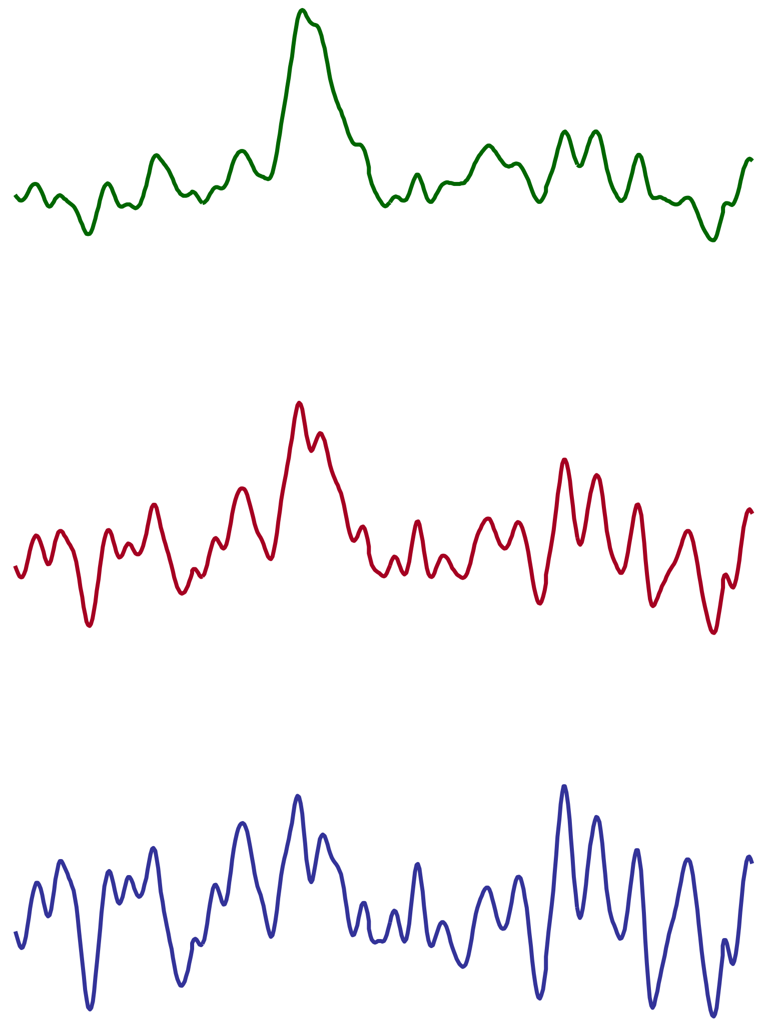
Computational
limit

Just enough data—
optimal solution is
meaningful, but hard to
find?

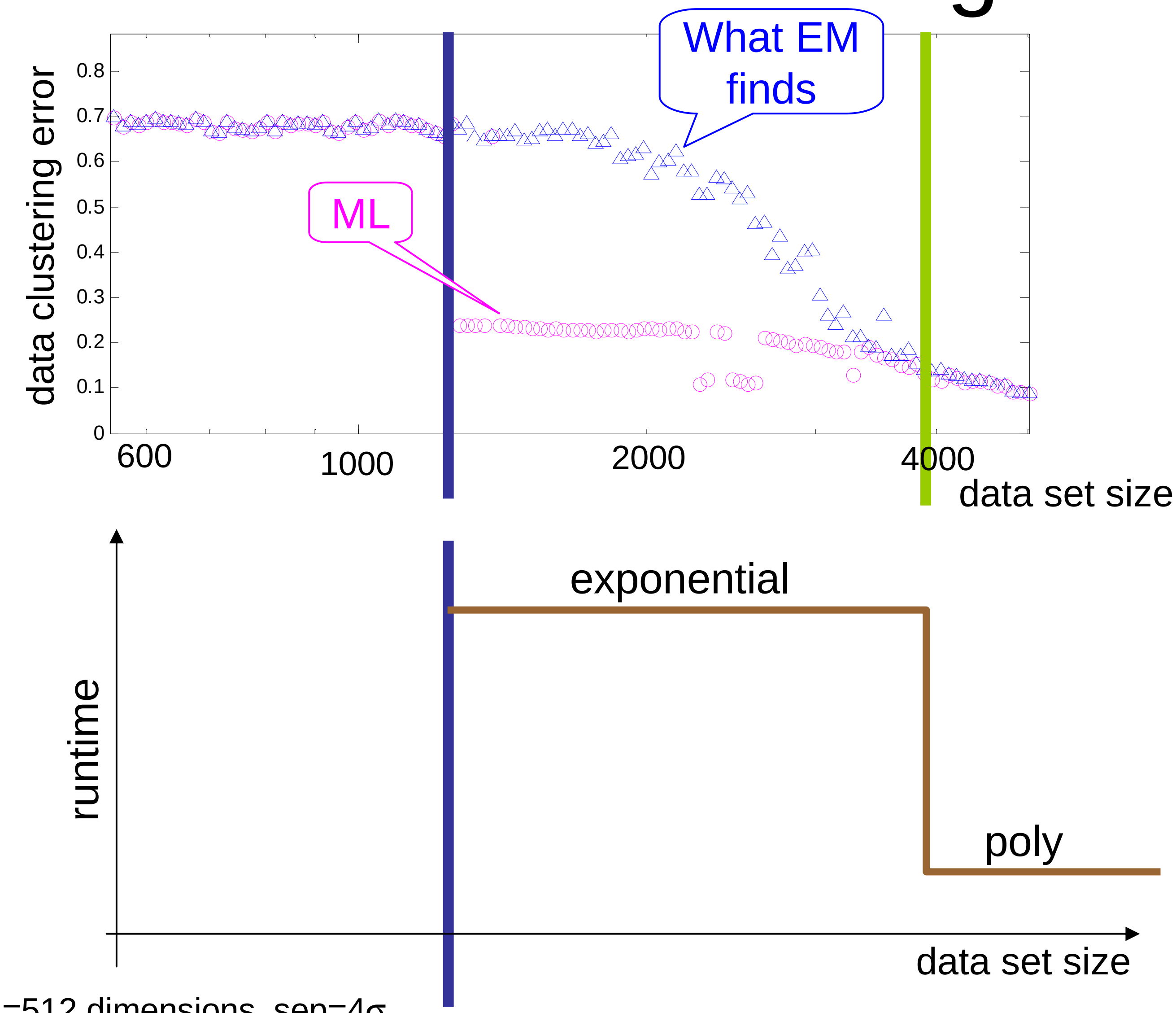
Informational
limit

Not enough data—
“optimal” solution is
meaningless.

Smaller data set

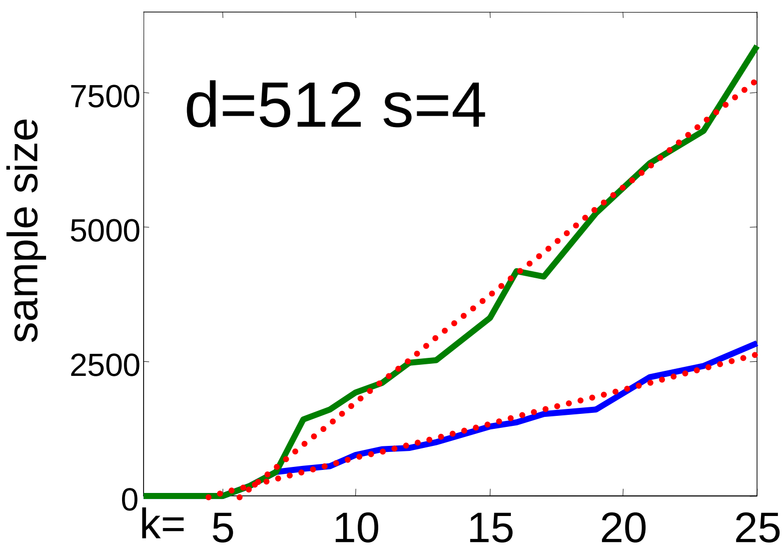


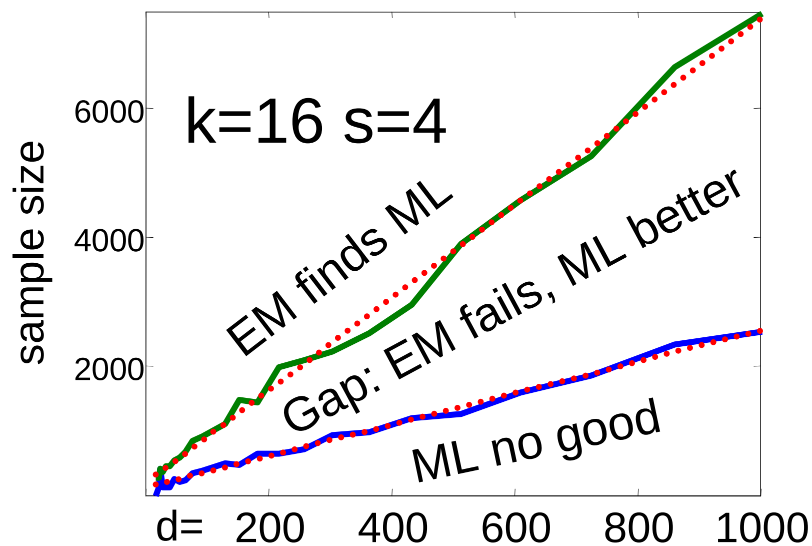
Computational and Information Limits in Clustering

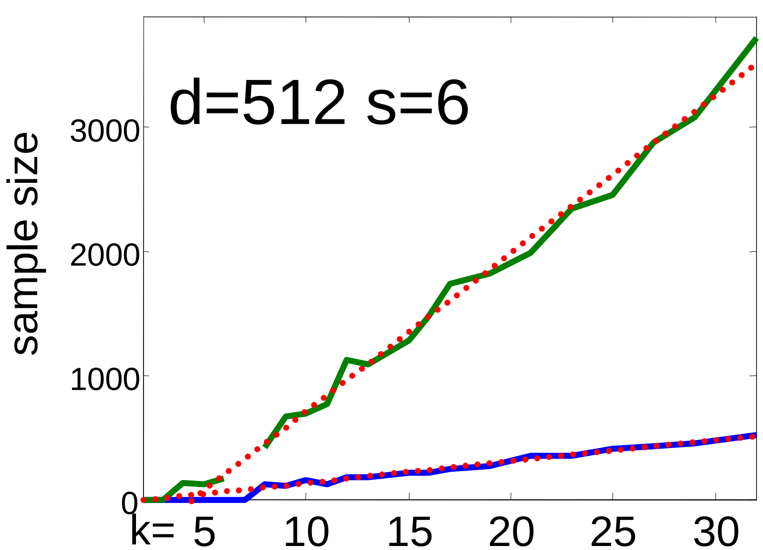


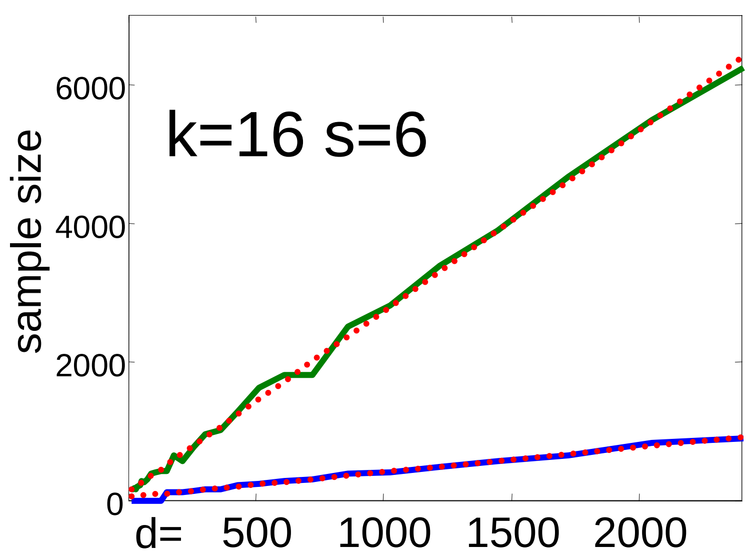
k=16 clusters in d=512 dimensions, sep=4σ

Dependence on dimensionality (d) and number of clusters (k)

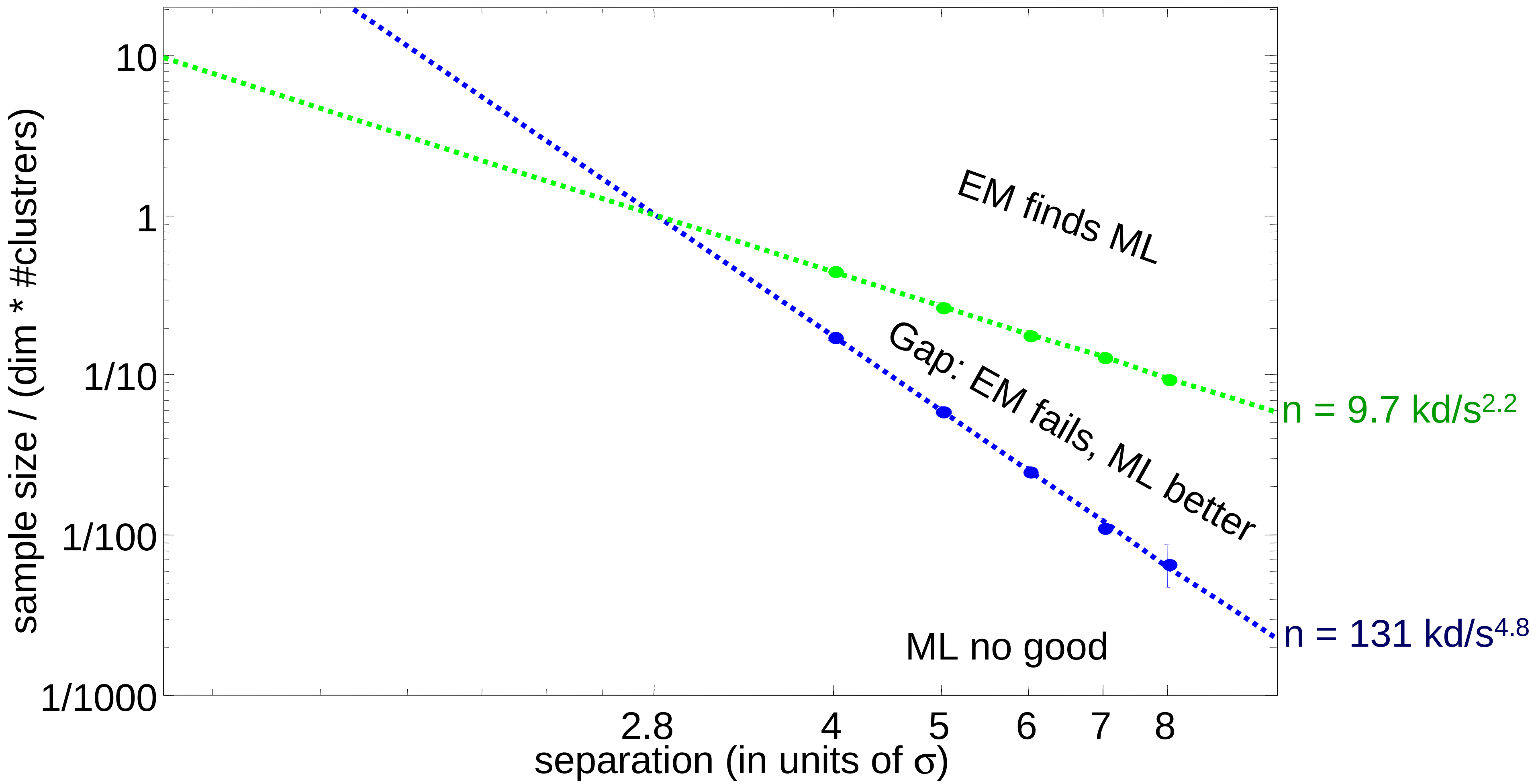








Dependence on the cluster separation



Conclusions from Empirical Study

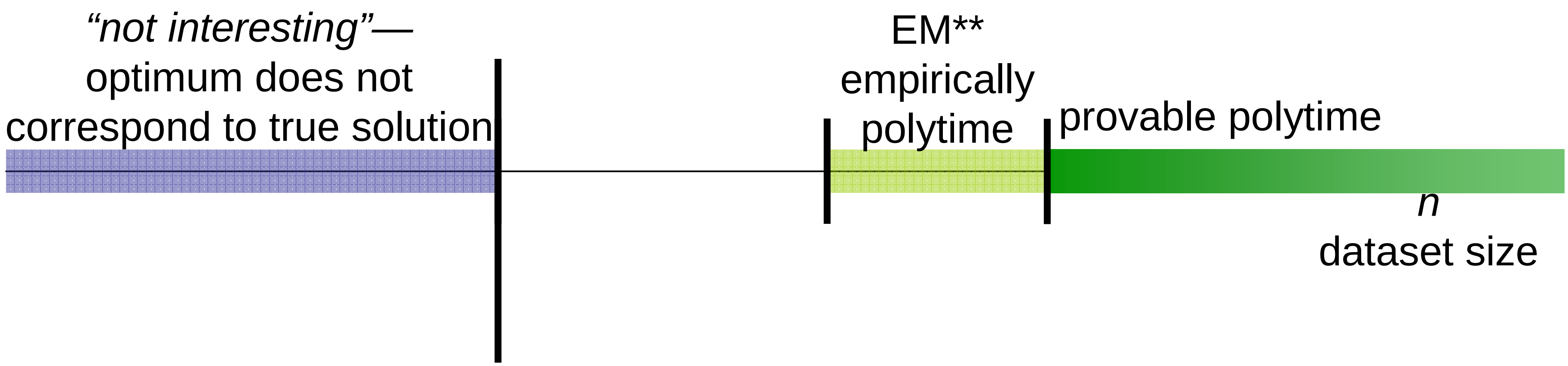
- With enough samples, EM does find global ML, even with low separation

- **There is an informational cost to tractability**
  (at least when using known methods)

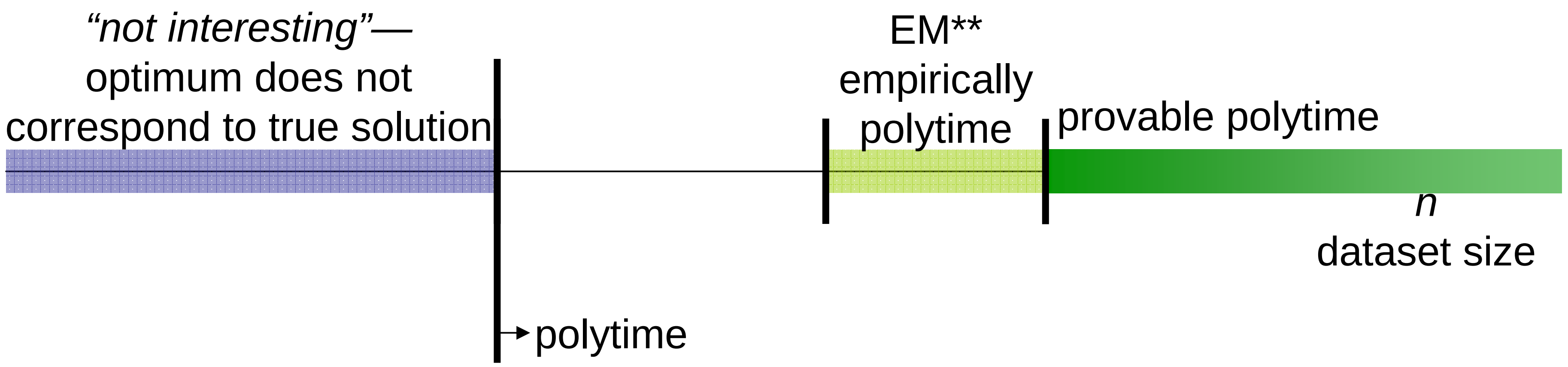
- Cost of tractability: PCA+EM+pruning (best known method) may require about **s^2 as much data** as what is statistically necessary

- **Cost increases when separation increases**

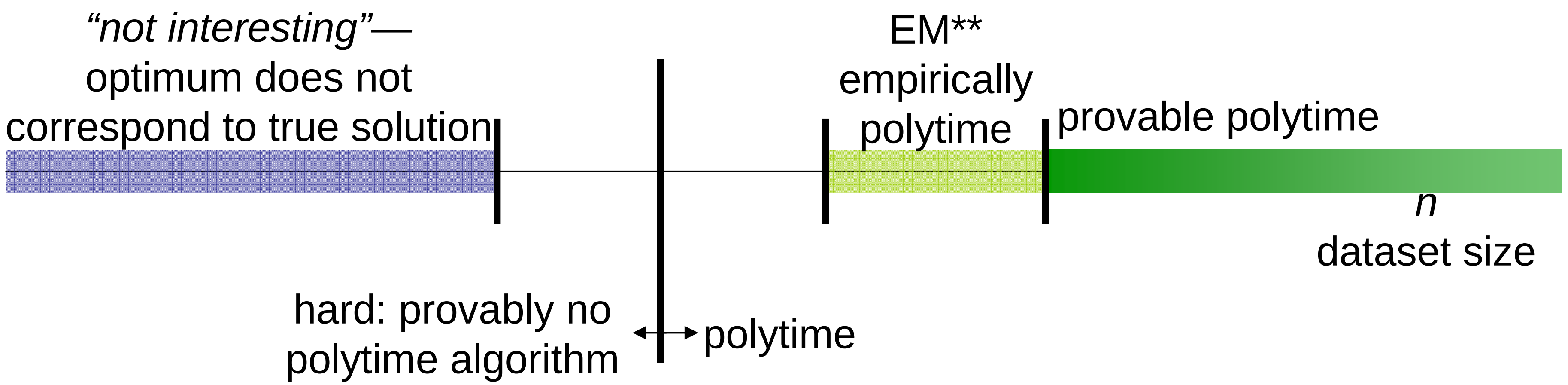
Hardness as a Function of Dataset Size



Hardness as a Function of Dataset Size



Hardness as a Function of Dataset Size



Informational Cost of Tractability?

- Gaussian Mixture Clustering
- Learning structure of dependency networks
  - Hard to find optimal (ML) structure in the worst case **[Srebro 01]**
  - Polynomial-time algorithms for the large-sample limit **[Chechetka Guestrin 07]**
- Graph partitioning (correlation clustering)
  - Hard in the worst case
  - Easy for large graphs with a “nice” partitions **[McSherry 03]**
- Finding cliques in random graphs
- Planted Noisy MAX-SAT

More Data $\Rightarrow$ Less Work

- Required runtime:
  - **increases** with complexity of the answer (separation, decision boundary)
  - **increases** with desired accuracy
  - **decreases** with amount of available data
- PEGASOS (stochastic sub-gradient descent for SVMs):
  - Runtime to get fixed optimization accuracy doesn’t depend on n

  $\rightarrow$ Best performance in data-laden regime

  $\rightarrow$ Runtime **decreases** as more data is available
- Clustering
  - Past informational limit, extra data is needed to make problem tractable
  - Cost of tractability increases quadratic with cluster seperation