A Scalable Modular Convex Solver for Regularized Risk Minimization (BMRM)

Choon Hui Teo

Statistical Machine Learning Program, NICTA

RSISE, Australian National University

(Joint work with Quoc V. Le, Alex Smola and S.V.N. Vishwanathan)

Regularized Risk Minimization

Many machine learning problems can be cast in the form,

$$\underset{w}{\text{minimize}} \;\; J(w) := \lambda\Omega(w) + R(w)$$

$$\text{where } R(w) := \frac{1}{m}\sum_{i=1}^{m} l(x_i, y_i, w)$$

- w: weight vector
- $\{(x_i, y_i)\}_{i=1}^{m}$: training data
- $l(x, y, w)$: convex and non-negative loss function
- $\Omega(w)$: convex and non-negative regularizer
- λ: regularization constant

Examples

Method (obj. fn.)	$\lambda\Omega(w)$	+	$R(w)$
linear SVMs	$\frac{\lambda}{2}\left\|w\right\|_2^2$	+	$\frac{1}{m}\sum_{i=1}^{m}\max\left\{0, 1 - y_i\left\langle w, x_i\right\rangle\right\}$
ℓ_1 log. reg.	$\lambda\left\|w\right\|_1$	+	$\frac{1}{m}\sum_{i=1}^{m}\log\left(1 + \exp\left(-y_i\left\langle w, x_i\right\rangle\right)\right)$
ϵ-insensitive reg.	$\frac{\lambda}{2}\left\|w\right\|_2^2$	+	$\frac{1}{m}\sum_{i=1}^{m}\max\left\{0, \left\|y_i - \left\langle w, x_i\right\rangle\right\| - \epsilon\right\}$

How to solve these problems?

1. Newton and quasi-Newton Methods
   - When the (convex) function is differentiable
2. Cutting Plane based Methods
   - When the (convex) function is continuous
   - *Meaningful* termination criterion

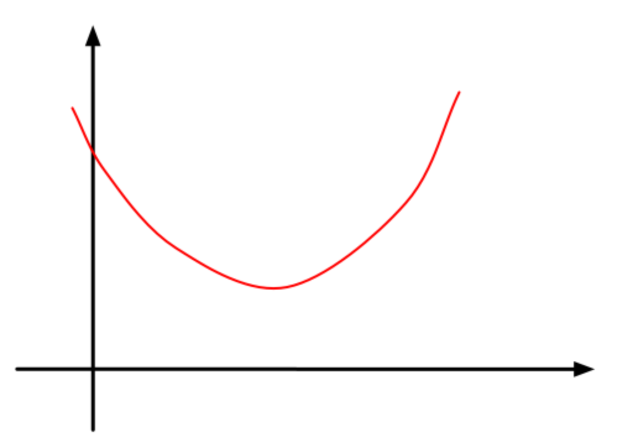
How to solve these problems?

1. Newton and quasi-Newton Methods
   - When the (convex) function is differentiable
2. Cutting Plane based Methods
   - When the (convex) function is continuous
   - *Meaningful* termination criterion

Cutting Plane Methods (CPM)

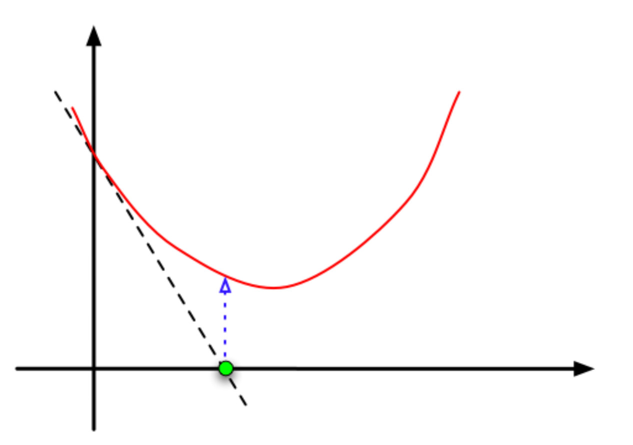
- Given: Convex (and non-negative) function $R(w)$
- Idea: First order Taylor approximation lower-bounds $R(w)$

The convex function...



- Red curve: convex non-negative function

The lower bound...

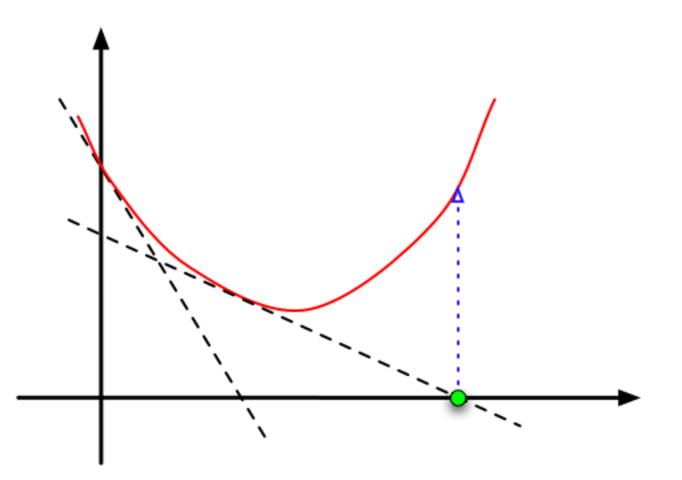


- Black dashed line: 1st-order Taylor approx. at $w = 0$
- Green dot: minimum of the lower bound
- Blue dashed line: current approximation gap ϵ_0

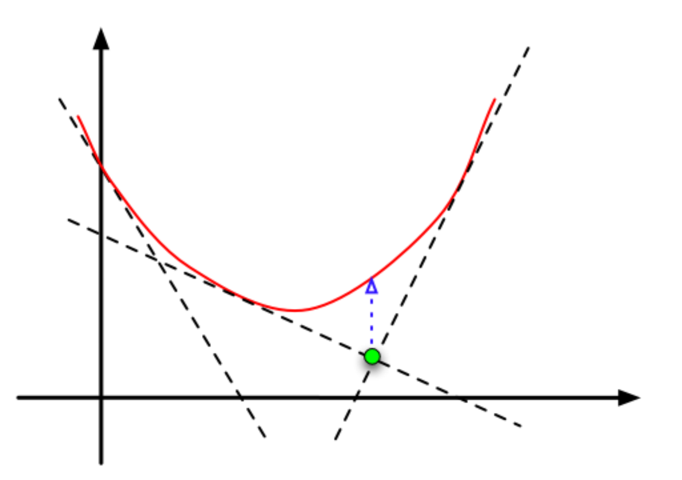
Cutting Plane Methods (CPM)

- Given: Convex, non-negative convex function $R(w)$
- Idea: First order Taylor approximation lower-bounds $R(w)$
- Fact: More approximations $\longrightarrow$ better lower bound

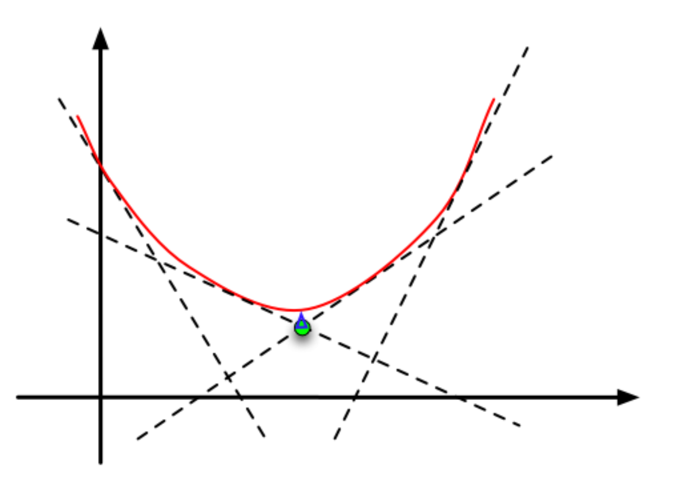
The lower bound is better...



The lower bound is better and better...



The lower bound is better and better and better...



Cutting Plane Methods (CPM)

- Given: Convex, non-negative convex function $R(w)$
- Idea: First order Taylor approximation lower-bounds $R(w)$
- Fact: More approximations $\longrightarrow$ better lower bound
- Summary: Iteratively improve the piecewise-linear lower bound and minimize it

$$\min_{w,\xi} \quad \xi$$
$$\text{s.t.} \quad \langle \partial_w R(w_i), w - w_i \rangle + R(w_i) \leq \xi \quad \forall i$$

- Note: Take any subgradient when $R(w_i)$ is not differentiable

Bundle Methods (BM)

Is basically CPM stabilized with (Moreau-Yosida) regularizer, i.e.,

$$\begin{aligned} \min_{w,\xi} \quad & \frac{\lambda}{2}\|w - \bar{w}\|_2^2 + \xi \\ \text{s.t.} \quad & \langle \partial_w R(w_i), w - w_i \rangle + R(w_i) \leq \xi \quad \forall i, \end{aligned}$$

where $\bar{w}$ is the *current* minimizer.

Point: Prevent new minimizer from moving “too” far away from the current

A Variant of BM

But, our (machine learning) problem comes with a regularizer $\Omega(w)$

$$\min_{w,\xi} \quad \lambda\Omega(w) + \xi$$
$$\text{s.t.} \quad \langle \partial_w R(w_i), w - w_i \rangle + R(w_i) \leq \xi \quad \forall i,$$

Examples of $\Omega(w)$:

- $\Omega(w) = \|w\|_1 \longrightarrow$ Linear Program
- $\Omega(w) = \|w\|_2^2 \longrightarrow$ Quadratic Program

Rate of Convergence

Question

How fast does the approximate minimizer $\bar{w}$ approach actual minimizer w^*?

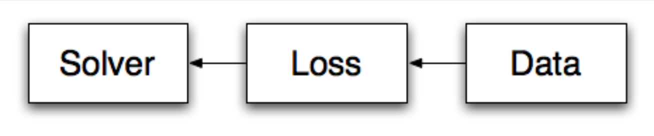
Answer

$O(\frac{1}{\epsilon})$, where $\epsilon := R(w^*) - R(\bar{w})$.

ϵ is the *meaningful* termination criterion.

Architecture of BMRM

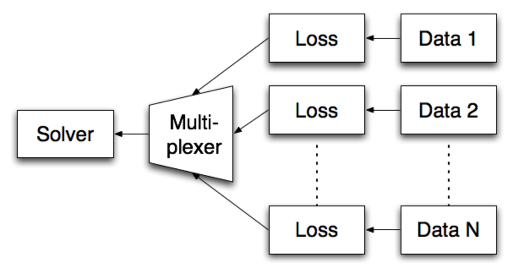
For serial computation:



- Data module manages dataset
- Loss module computes loss and (sub)gradient
- Solver module solves optimization problem ($\Omega(w)$-specific)
- Modules are loosely coupled

Architecture of BMRM (cont'd)

For parallel/distributed computation:



- For decomposable loss function
- Split dataset into sub-datasets
- Each node computes loss w.r.t. its sub-dataset
- Multiplexer aggregates the loss and (sub)gradients and broadcast new w

Experiment 1: Training time comparison

- Task: Binary classification
- Solvers:
  - Our method BMRM (in particular, ℓ_2 norm and soft-margin loss)
  - SVMPERF [Joachims, KDD'06]
- Datasets:
  - kdd99 (m=4898431, dim.=127, den.=12.86%)
  - reuters-c11 (m=23149, dim.=47236, den.=0.16%)
- Setting:
  - ϵ = 1e-5
  - $\lambda \in \{1,0.3,0.1,\ldots,3\text{e-}6\}$

BMRM is comparable to SVMPERF

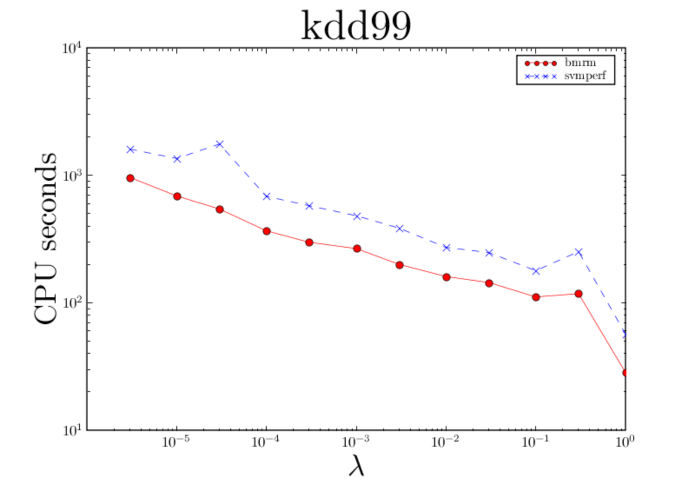


Figure: log-log plot of linear SVM training time vs. regularization constant λ on `kdd99`.

BMRM is comparable to SVMPERF (cont'd)

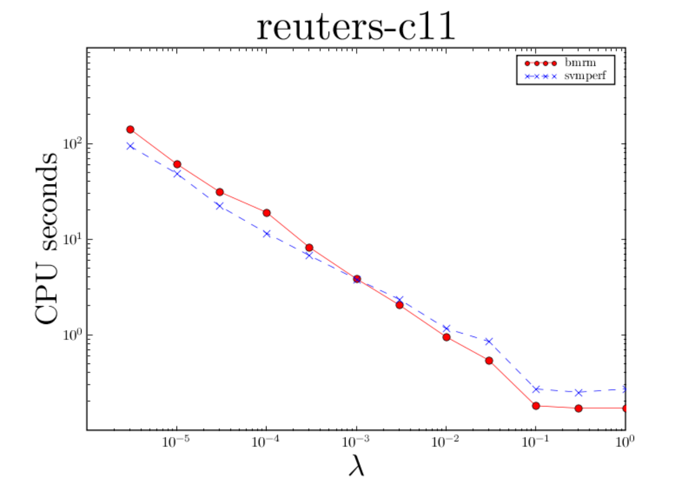


Figure: log-log plot of linear SVM training time vs. regularization constant λ on `reuters-c11`.

Experiment 2: Convergence rate

- Task: Binary classification
- Solvers: `BMRM`
- Datasets: `kdd99` and `reuters-c11`
- Setting: $\epsilon = 1\text{e-}5$, $\lambda = 3\text{e-}6$

BMRM converged under $O(1/\epsilon)$ steps

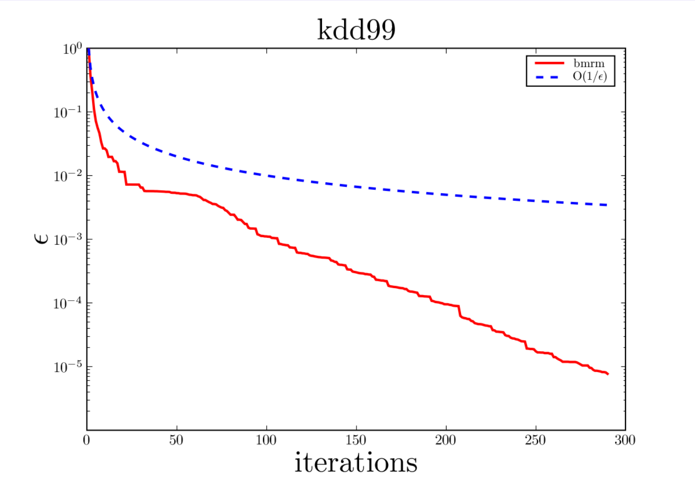


Figure: semilog-y plot of approximation gap ϵ vs. iterations

BMRM converged under $O(1/\epsilon)$ steps (cont'd)

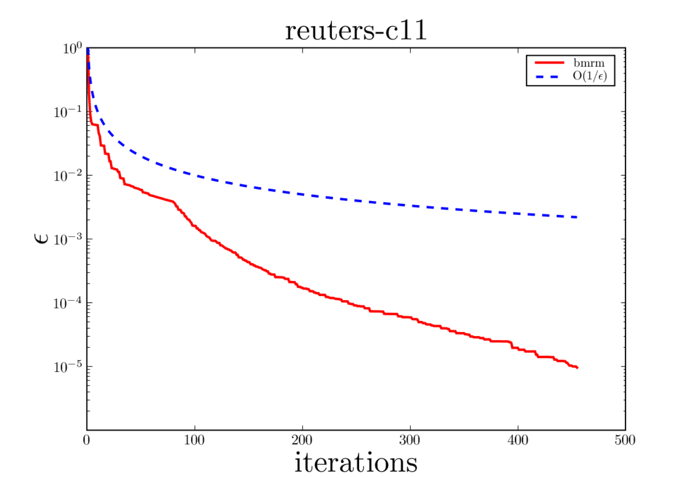


Figure: semilog-y plot of approximation gap ϵ vs. iterations

Experiment 3: Parallelization of BMRM

- Task: Ranking
- Methods:
  - Normalized Discounted Cumulative Gain (NDCG)
  - Ordinal regression
- Dataset: MSN
- $\epsilon = 1\text{e-}5$
- $\lambda \in \{10, 100\}$
- Number of computers $n \in \{1, 2, 4, \ldots, 512\}$

BMRM runtime $\propto 1/n$

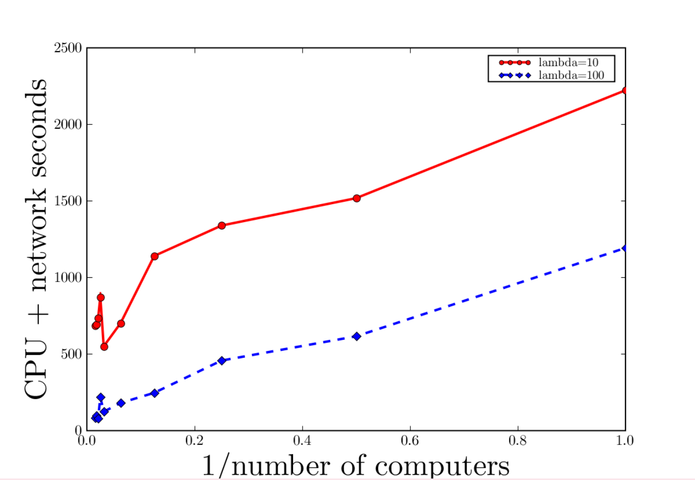


Figure: Plot of NDCG training time vs. the inverse number of computers

BMRM runtime $\propto 1/n$ (cont'd)

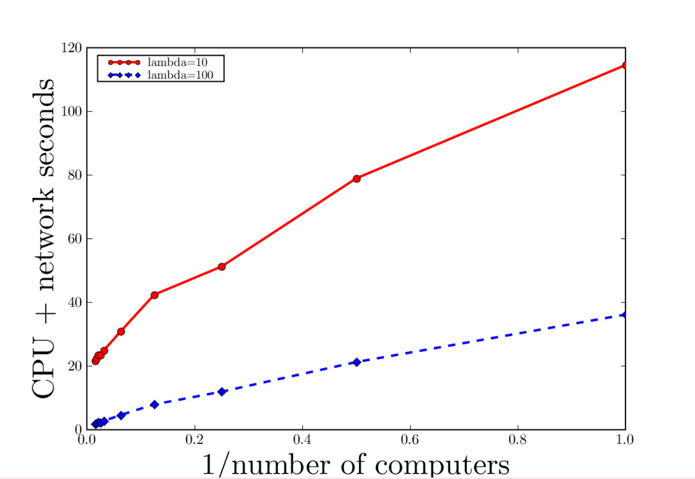


Figure: Plot of Ordinal regression training time vs. the inverse number of computers

Conclusion

- Unconstrained formulation leads to easy, modular and scalable solver design
- "Job specialization": optimization, loss, parallelization scheme

Thank you!

(Poster 23, Tuesday 14th August 07)