Matrix Exponentiated Gradient Updates for On-line Learning and Bregman Projection

---

Koji Tsuda

Max Planck Institute for Biological Cybernetics
AIST Computational Biology Research Center

Joint work with Gunnar Rätsch (FML) and Manfred Warmuth (UCSC)

Outline

- Learning theory developed for vector parameters
  - Online learning: Gradient descent, Exponentiated gradient
  - Boosting
- Generalize the framework to symmetric matrix parameters
  - Parameter matrix constrained to be *positive definite*
- Bregman divergences as main tool
  - Positive Vector: Kullback-Leibler divergence
  - Positive Definite Matrix: *Von Neumann divergence*

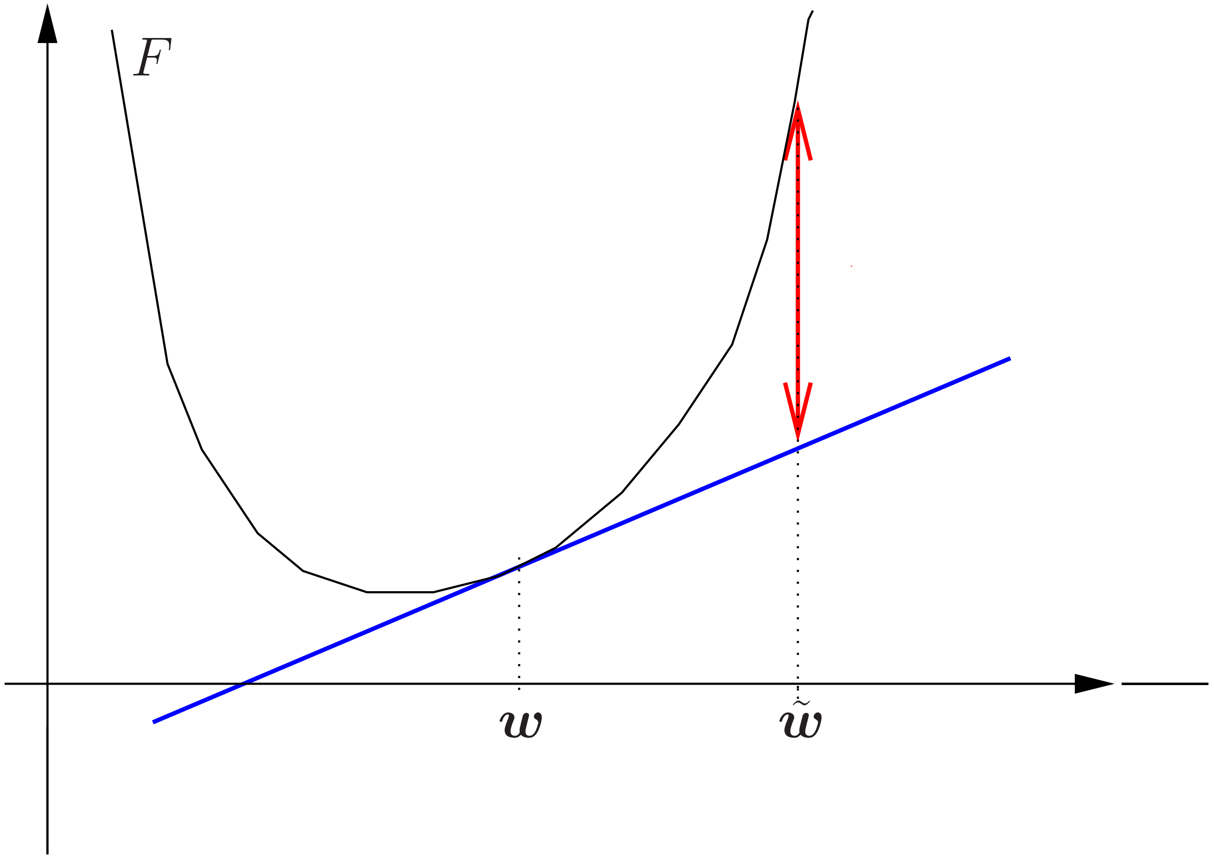
Contents of this talk

- 1. Online Learning Algorithm for Positive Definite Matrices (85%)
- 2. Bregman Projection Algorithm for Positive Definite Matrices (15%)
  - Connection to AdaBoost: *DefiniteBoost*

Bregman divergence

- For convex function $F(\boldsymbol{w})$ with gradient $\quad f(\boldsymbol{w}) := \nabla_{\boldsymbol{w}} F(\boldsymbol{w})$

$$\Delta_F(\tilde{\boldsymbol{w}}, \boldsymbol{w}) = F(\tilde{\boldsymbol{w}}) - (F(\boldsymbol{w}) + (\tilde{\boldsymbol{w}} - \boldsymbol{w}) \cdot f(\boldsymbol{w}))$$



Two main examples

Squared Euclidean Distance

$$\begin{aligned} F(\boldsymbol{w}) &= ||\boldsymbol{w}||_2^2/2 \\ f(\boldsymbol{w}) &= \boldsymbol{w} \\ \Delta_F(\widetilde{\boldsymbol{w}}, \boldsymbol{w}) &= ||\widetilde{\boldsymbol{w}}||_2^2/2 - ||\boldsymbol{w}||_2^2/2 - (\widetilde{\boldsymbol{w}} - \boldsymbol{w}) \cdot \boldsymbol{w} \\ &= ||\widetilde{\boldsymbol{w}} - \boldsymbol{w}||_2^2/2 \end{aligned}$$

(Unnormalized) Relative Entropy

$$\begin{aligned} F(\boldsymbol{w}) &= \sum_i (w_i \ln w_i - w_i) \\ f(\boldsymbol{w}) &= \ln \boldsymbol{w} \\ \Delta_F(\widetilde{\boldsymbol{w}}, \boldsymbol{w}) &= \sum_i \left(\widetilde{w}_i \ln \frac{\widetilde{w}_i}{w_i} + w_i - \widetilde{w}_i \right) \end{aligned}$$

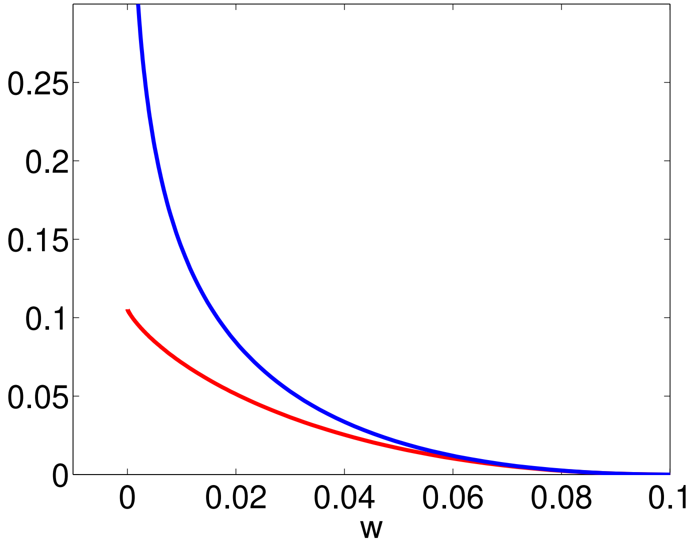
Now $w_i \geq 0$!

Kullback-Leibler divergence

- When $\sum_i w_i = 1$

$$\Delta(\tilde{\boldsymbol{w}}, \boldsymbol{w}) = \sum_i \tilde{w}_i \log \frac{\tilde{w}_i}{w_i}$$

- Asymmetric: $\Delta([w, 1-w], [0.1, 0.9]) \neq \Delta([0.1, 0.9], [w, 1-w])$



Applications of Bregman divergences

- Information geometry
  - Motivates EM
- Online Learning
  - Multiplicative Updates
  - Measure of progress for proving Relative Loss Bounds
- Boosting
  - As Bregman projection
  - Measure of progress in convergence proof

Online Learning (Vector Parameter)

- Parameter at time t: $\boldsymbol{w}_t \in \mathcal{D}$
- Iteration $t = 1, \ldots, T$
  - Receive an input $\boldsymbol{x}_t$
  - Produce a prediction $\hat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$
  - Receive an output y_t and compute loss $L_t = (\hat{y}_t - y_t)^2$
  - Update $\boldsymbol{w}_t$ to $\boldsymbol{w}_{t+1}$
- Design the update such that
  1. Total loss $L = \sum_{t=1}^{T}(\boldsymbol{w}_t \cdot \boldsymbol{x}_t - y_t)^2$ minimized
  2. Parameter vector stays in $\mathcal{D}$

Motivating the update [KW]

- New parameter $\boldsymbol{w}_{t+1}$ close to $\boldsymbol{w}_t$, and loss for $(\boldsymbol{x}_t, y_t)$ small
- Minimize sum of Bregman divergence and loss

$$\boldsymbol{w}_{t+1} = \operatorname{argmin}_{\boldsymbol{w}} \left(\Delta_F(\boldsymbol{w}, \boldsymbol{w}_t) + \eta\, (\boldsymbol{w} \cdot \boldsymbol{x}_t - y_t)^2\right)$$

- Setting derivative to zero:

$$f(\boldsymbol{w}_{t+1}) - f(\boldsymbol{w}_t) + 2\eta(\boldsymbol{w}_{t+1} \cdot \boldsymbol{x}_t - y_t)\boldsymbol{x}_t = 0$$

$$\begin{aligned} \Leftrightarrow \boldsymbol{w}_{t+1} &= f^{-1}(f(\boldsymbol{w}_t) - 2\eta(\boldsymbol{w}_{t+1} \cdot \boldsymbol{x}_t - y_t)\boldsymbol{x}_t) \\ &\approx f^{-1}(f(\boldsymbol{w}_t) - 2\eta(\boldsymbol{w}_{t} \cdot \boldsymbol{x}_t - y_t)\boldsymbol{x}_t) \end{aligned}$$

Examples of the updates

- Gradient Descent: $f(\boldsymbol{w}) = \boldsymbol{w}$

$$\boldsymbol{w}_{t+1} := \boldsymbol{w}_t - 2\eta(\boldsymbol{w}_t \cdot x_t - y_t)\boldsymbol{x}_t)$$

- Unnormalized Exponentiated gradient: $f(\boldsymbol{w}) = \ln(\boldsymbol{w})$

$$\boldsymbol{w}_{t+1,i} := \boldsymbol{w}_{t,i} \exp(-2\eta(\boldsymbol{w}_t \cdot x_t - y_t)\boldsymbol{x}_t)$$

- Exponentiated gradient

$$\boldsymbol{w}_{t+1,i} := \boldsymbol{w}_{t,i} \exp(-2\eta(\boldsymbol{w}_t \cdot x_t - y_t)\boldsymbol{x}_t)/Z_t$$

Why Bregman divergence?

- Keep parameters in domain
- Use divergence as measure of progress in analysis
- Clarify the relations to other divergence-based algorithms (e.g. Boosting)

- Update for positive definite matrices???

Von Neumann or quantum entropy

- PD matrix W

$$F(W) = \operatorname{tr}(W \log W - W)$$

- Normalized case: $\operatorname{tr}(W) =$ sum of eigenvals. $= 1$

$$F(W) = \operatorname{tr}(W \log W) - 1$$

- Using eigen decomposition
  $W = \sum_i \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^\top$,

$$F(W) = \left(\sum_i \lambda_i \ln \lambda_i \right) - 1$$

Von Neumann divergence or quantum relative entropy

- PD matrices $\tilde{W}$ and W

$$\Delta_F(\tilde{W}, W) = \mathrm{tr}(\tilde{W} \log \tilde{W} - \tilde{W} \log W - \tilde{W} + W)$$

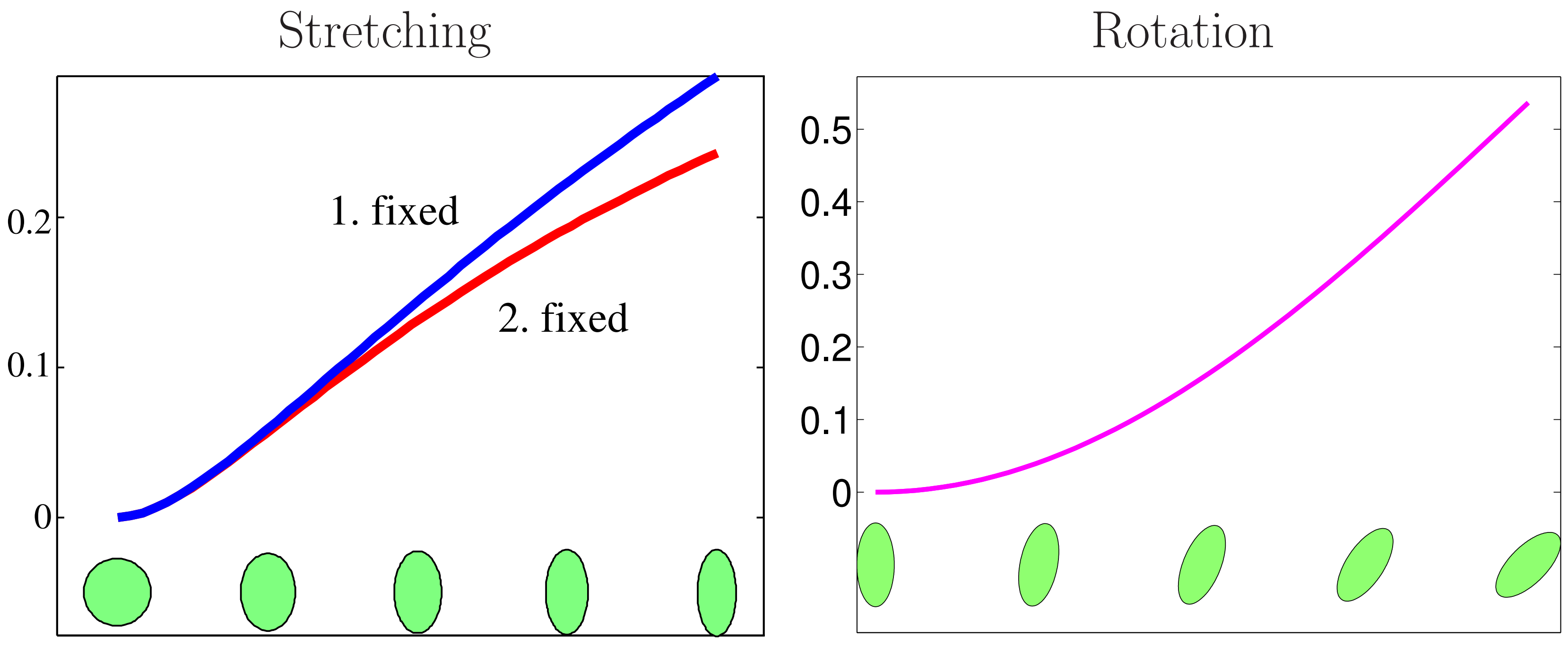
- Normalized $tr(W) = 1$

$$\Delta_F(\tilde{W}, W) = \mathrm{tr}(\tilde{W} \log \tilde{W} - \tilde{W} \log W)$$

- Quantifies the difference in the eigenvalues as well as the eigenvectors
- When $\tilde{\boldsymbol{v}}_i = \boldsymbol{v}_i$, $\Delta_F(\tilde{W}, W) = \sum_i \tilde{\lambda}_i \ln \frac{\tilde{\lambda}_i}{\lambda_i}$

Plots of $\Delta(\tilde{W}, W)$

- Eigen values $\boldsymbol{v}_i$ are directions of axes and $\sqrt{\lambda_i}$ their lengths



Same eigen vectors

Same eigen values

(Symmetric only for 2 dim.)

Online Learning with matrix parameters

- Parameter at time t: Symmetric matrix W_t
- W_t is *positive definite* and $\mathrm{tr}(W) = 1$
- Iteration $t = 1, \ldots, T$
  - Receive an input X_t
  - Produce a prediction $\hat{y}_t = \mathrm{tr}(W_t X_t)$
  - Receive an output y_t and compute loss $L_t = (\hat{y}_t - y_t)^2$
  - Update W_t to W_{t+1}

Matrix Exponentiated Gradient Updates

- General update formula derived from the Bregman divergence

$$W_{t+1} = f^{-1}(f(W_t) - 2\eta(\hat{y}_t - y_t)X_t)$$

- $F(W) = \mathrm{tr}(W \log W - W)$, thus $f(W) = \log W$ and $f^{-1}(W) = \exp W$
- Matrix exponentiated gradient (MEG) update

$$W_{t+1} = \exp(\log W_t - 2\eta(\hat{y}_t - y_t)X_t)$$

- Normalized version (trace 1)

$$W_{t+1} = \frac{1}{Z_t}\exp(\log W_t - 2\eta(\hat{y}_t - y_t)X_t)$$

Keeping Positive Definiteness

- Matrix logarithm maps PD matrix to symmetric matrix
- Matrix exponential maps any symmetric matrix back to PD matrix
- Wlog X_t symmetric since $\operatorname{tr}(\underbrace{W_t}_{\text{symm.}} X_t) = \operatorname{tr}(W_t \frac{X_t - X_t}{2})$
- η may be arbitrary large

$$W_{t+1} \quad = \quad \overbrace{\exp(\underbrace{\underbrace{\log \overbrace{W_t}^{PD}}_{Sym} - 2\eta(\hat{y}_t - y_t) \underbrace{X_t}_{Sym}}_{Sym})}^{PD}$$

Relative loss bounds

- Total loss for a fixed comparator matrix U and samples $S = \{X_t, y_t\}_{t=1}^{T}$

$$L_U(S) = \sum_{t=1}^{T} (\mathrm{tr}(U X_t) - y_t)^2$$

- $L(S)$: Total loss of our learning algorithm
- Total loss *relative to* comparator U

$$L(S) \le \alpha L_U(S) + \beta \Delta(U, W_1), \text{ for any } S, W_1 \text{ and } U$$

- When $L_U(S) = 0$

$$L(S) \le \gamma \Delta(U, W_1)$$

Relative loss bounds for MEG update

- S: Sequence of Examples (X_t, y_t)
- Difference between the largest and smallest eigenvalue of X_t is at most r
- W_1 and U: Start and comparison matrices

Theorem 1. *For any c such that $\eta = 2c/(r^2(2+c))$,*

$$L(S) \leq (1+\frac{c}{2}) Loss_U(S) + (\frac{1}{2}+\frac{1}{c}) r^2 \Delta(U, W_1). \tag{1}$$

- Algorithm and bound generalize the usual vector valued case

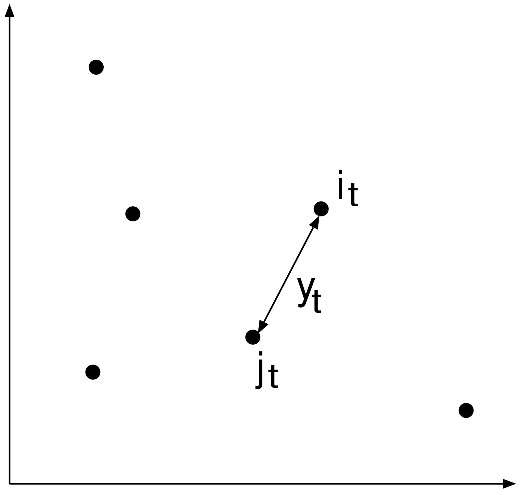
Proof?

- Same structure
- Problem: $\exp(A+B) = \exp(A)\exp(B)$ does not hold except for the special case $AB = BA$
- Rescue:
  - Golden-Thomson inequality

$$\mathrm{tr}[\exp(A+B)] \le \mathrm{tr}[\exp(A)\exp(B)]$$

Artificial Experiment: Online Learning of a Gram Matrix

- Teacher: Pick up two points randomly and tell the learner the squared Euclidean distance between them



- Example at time t

$$(i_t, j_t, y_t), \quad y_t = \|\boldsymbol{x}_{i_t} - \boldsymbol{x}_{j_t}\|^2$$

- Learner: Reconstruct the hidden Gram matrix of all samples (trace 1)

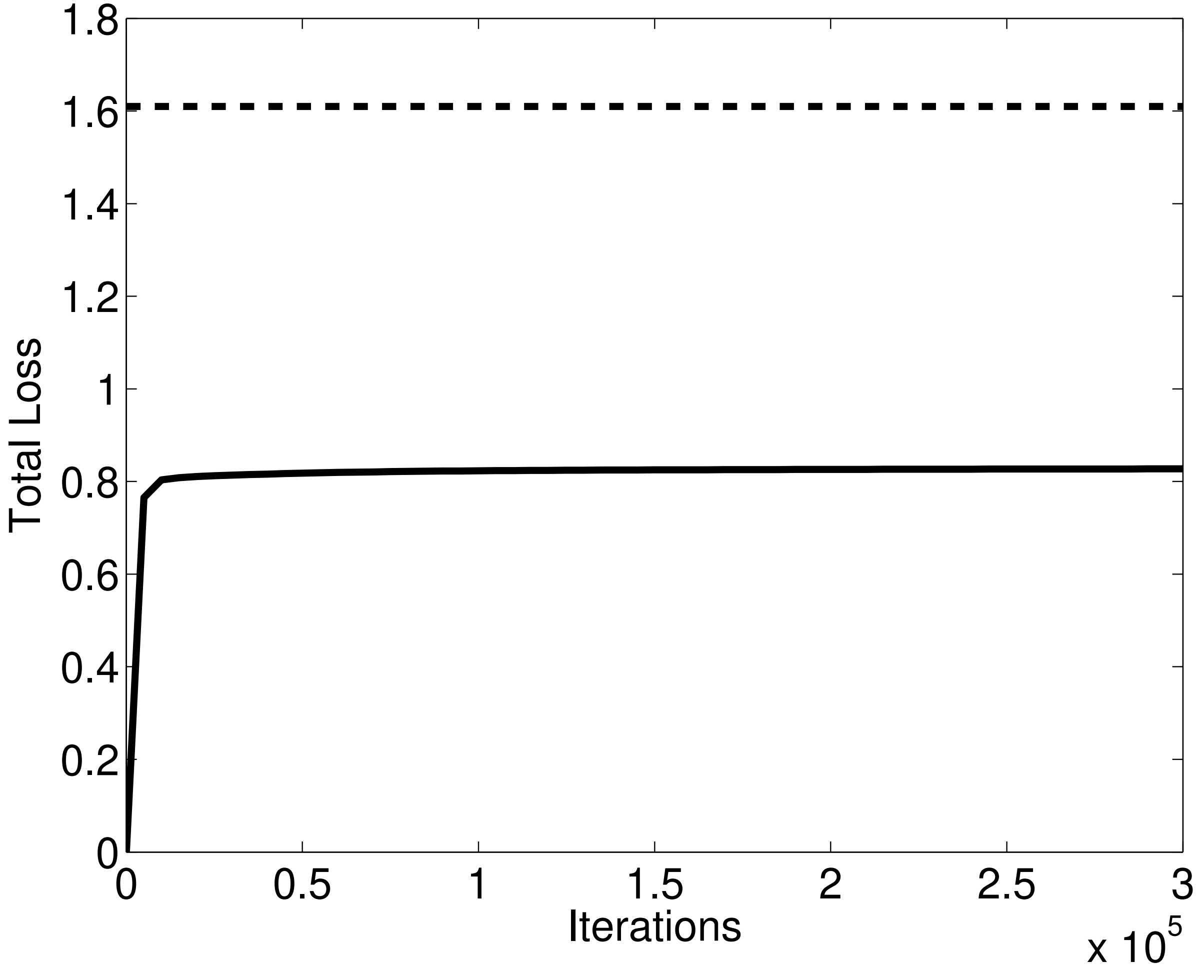
Experiment (Contd.)

- If K_0 is the teacher Gram matrix, $\|\boldsymbol{x}_{i_t} - \boldsymbol{x}_{j_t}\|^2 = \mathrm{tr}(X_t K_0)$,

$$X_t = \begin{array}{cc} & \begin{matrix} i_t & j_t \end{matrix} \\ \begin{matrix} i_t \\ j_t \end{matrix} & \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \end{array}$$

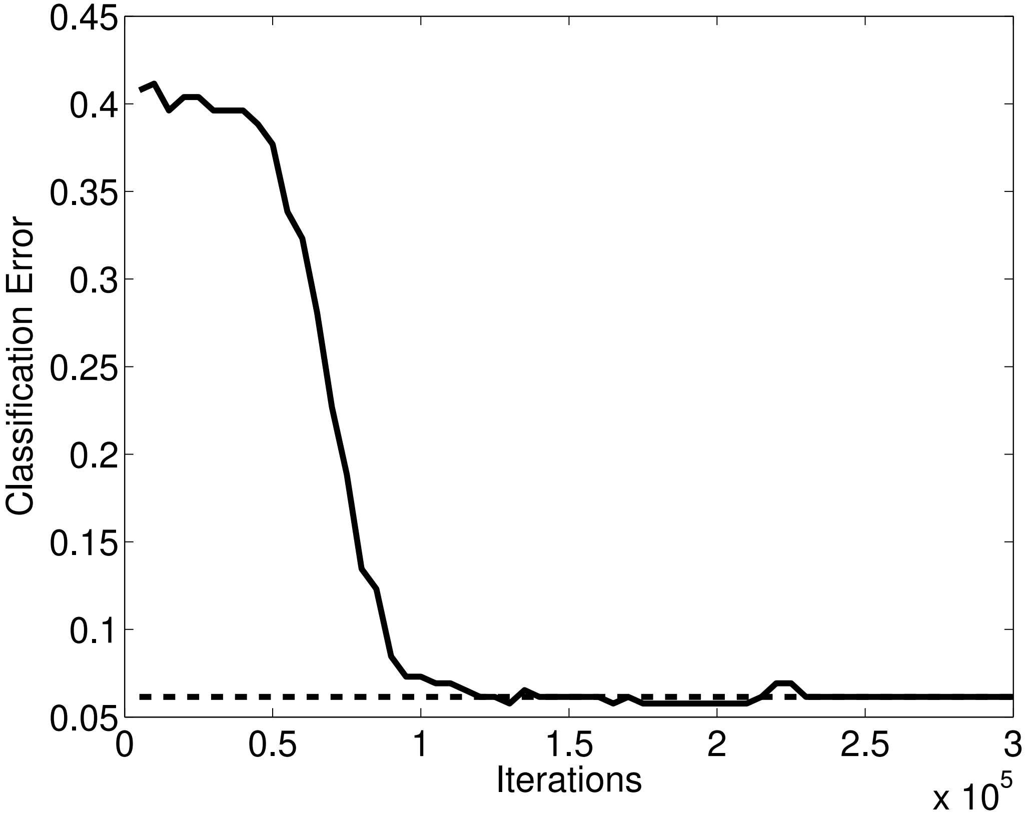
- Example at time t: matrix-scalar pair (X_t, y_t)
- Reconsruct a kernel matrix by MEG update starting from $W_1 = I/d$.
- Compute the bound setting $U = K_T$.
- Teacher matrix: 52×52, generated from bacterial amino acid sequences (3 species)

Growth of the total loss ($\eta = 2$)



Error rate of Nearest Neighbor Classifier

- Classification error for the kernel matrix at time t
- 50%-50% training/test division



Bregman Projection

- Find a PD matrix closest to W_1 subject to a number of constraints

$$\begin{aligned} \operatorname{argmin}_W \;& \Delta_{\mathcal{F}}(W, W_1) \\ & \operatorname{tr}(W) = 1,\ \operatorname{tr}(WC_j) \leq 0,\ \text{for } j = 1, \ldots, n \end{aligned}$$

- Encode available information as constraints (e.g. graph, class labels)
  - Locally constrained diffusion kernel (Tsuda and Noble, ISMB 2004)
  - Semisupervised classification (Xing et al., NIPS 2003)

Exact Successive Projections

- At the t-th step, an unsatisfied constraint is chosen
- Projection with only one constraint

$$W_{t+1} = \mathrm{argmin}_W \, \Delta(W, W_t), \quad \mathrm{tr}(W) = 1, \mathrm{tr}(WC_t) \leq 0.$$

- Dual problem: Solved by line search

$$\alpha_t = \mathrm{argmin}_\alpha \, \mathrm{tr}[\exp(\log W_t - \alpha C_t)], \quad \alpha \geq 0.$$

- Using the solution, W_t is updated as

$$W_{t+1} = \frac{1}{Z_t(\alpha_t)} \exp(\log W_t - \alpha_t C_t)$$

Successive Approximate Projection: *DefiniteBoost*

- Eigenvalues of C_t lie in the interval $[\lambda_t^{\min}, \lambda_t^{\max}]$ and $r_t = \mathrm{tr}(W_t C_t)$
- Approximate solution

$$\alpha_t = \frac{1}{\lambda_t^{\max} - \lambda_t^{\min}} \log\left(\frac{1 + r_t/\lambda_t^{\max}}{1 + r_t/\lambda_t^{\min}}\right),$$

- Still, convergence is guaranteed

Relation to AdaBoost

- When all matrices are diagonal, the DefiniteBoost degenerates to AdaBoost
- Training samples $\{\boldsymbol{x}_i, y_i\}_{i=1}^d$, Weak hypotheses $h_1(x), \ldots, h_n(x) \in [-1, 1]$
- If $C_j = \mathrm{diag}(y_1 h_j(x_1), \ldots, y_d h_j(x_d))$ and $W_1 = I/d$, the dual objective is

$$\frac{1}{d}\sum_{i=1}^{d} \exp\left(-y_i \sum_{j=1}^{n} \gamma_j h_j(x_i)\right)$$

- If $w_{ti} = [W_t]_{ii}$, the updating formula is

$$w_{t+1,i} = w_{ti} \exp(-\alpha_t y_i h_t(x_i))/Z_t(\alpha_t),$$

where $\alpha_t = \frac{1}{2}\log\frac{1+r_t}{1-r_t}$

Conclusion

- Von Neumann divergence generalizes KL divergence-based algorithms
- Relationship to Diffusion Kernels
- Variety of matrix divergences based on different convex functions
- Relationship to $\log |W|$ based algorithms