Tracking the best Expert

Yoav Freund

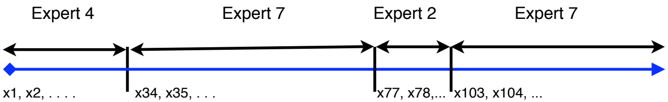
February 17, 2025

Based on “Tracking the best linear predictor” and “Tracking the best expert” by Herbster and Warmuth. Also, section 11.5 in Prediction learning and Games.

Tracking Linear Experts

- Usually: compare algorithm's total loss to total loss of the best expert.
- drifting experts: Compare with a sequence of experts that change over time.
- The amount of change is measured using total bregman divergence.
- Regret depends on $\sum_t \Delta_F(\mathbf{u}_{t-1}, \mathbf{u}_t)$
- The Projection Update After computing the unconstrained update, project the $\mathbf{w}_{t+1}$ onto a convex set.
- Does not allow the algorithm to over-commit to an extreme vector from which it is hard to recover.

Switching experts setup

- Usually: compare algorithm's total loss to total loss of the best expert.
- Switching experts: compare algorithm's total loss to total loss of best expert sequence with k switches.
- 

An inefficient algorithm

- Fix:
  - l - sequence length
  - k - number of switches
  - n - number of experts
- Consider one partition-expert per sequence of switching experts.
- No. of partition-experts : $\binom{l}{k-1} n(n-1)^k = O\left(n^{k+1}\left(\frac{el}{k}\right)^k\right)$
- The log-loss regret is at most $(k+1)\log n + k \log \frac{l}{k} + k$
- Requires maintaining $O\left(n^{k+1}\left(\frac{el}{k}\right)^k\right)$ weights.

generalization to mixable losses

- In this lecture we assume loss function is mixable.
- There is an exponential weights algorithm with learning rate η that achieves (in the non-switching case) a bound

$$L_A \leq \min_i L_i + \frac{1}{\eta} \log n$$

- Then using the partition-expert algorithm for the switching-experts case we get a bound on the regret $\frac{1}{\eta}\left((k+1)\log n + k \log \frac{l}{k} + k\right)$

Weight sharing algorithms

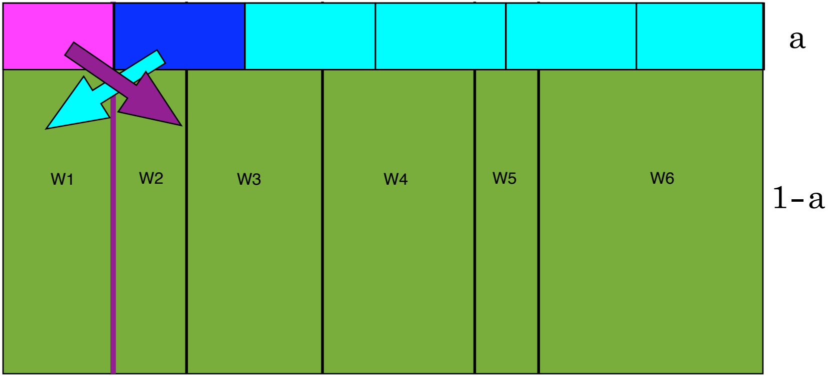
- Update weights in two stages: loss update then share update.
- Prediction uses the normalized s weights $w^s_{t,i} / \sum_j w^s_{t,j}$
- Loss update is the same as always, but defines intermediate m weights:

$$w^m_{t,i} = w^s_{t,i} e^{-\eta L(y_t, x_{t,i})}$$

- Share update: redistribute the weights
- Fixed-share:

$$\begin{aligned} pool &= \alpha \sum_{i=1}^{n} w^m_{t,i} \\ w^s_{t+1,i} &= (1-\alpha) w^m_{t,i} + \frac{1}{n-1} (pool - \alpha w^m_{t,i}) \end{aligned}$$

The fixed-share algorithm



Proving a bound on the fixed-share

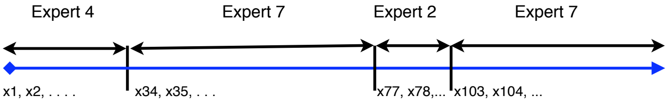
- The relation between algorithm loss and total weight does not change because share update does not change the total weight.
- Thus we still have

$$L_A \leq \frac{1}{\eta} \sum_{i=1}^{n} w_{l+1,i}^{s}$$

- The harder question is how to lower bound $\sum_{i=1}^{n} w_{l+1,i}^{s}$

Lower bounding the final total weight

- Fix some switching experts sequence:



- "follow" the weight of the chosen expert i_t.
- The loss update reduces the weight by a factor of $e^{-\eta \ell_{t,i_t}}$.
- The share update reduces the weight by a factor larger than:
  - $1-\alpha$ on iterations with no switch.
  - $\frac{\alpha}{n-1}$ on iterations where a switch occurs.

Bound for arbitrary α

- Combining we lower bound the final weight of the last expert in the sequence

$$w^s_{l+1,e_k} \geq \frac{1}{n} e^{-\eta L_*} (1-\alpha)^{l-k-1} \left(\frac{\alpha}{n-1}\right)^k$$

Where L_* is the cumulative loss of the switching sequence of experts.

- Combining the upper and lower bounds we get that for any sequence

$$L_A \leq L_* + \frac{1}{\eta}\left(\ln n + (l-k-1)\ln\frac{1}{1-\alpha} + k\left(\ln\frac{1}{\alpha} + \ln(n-1)\right)\right)$$

Tuning α

- let k^* be the best number of switches (in hind sight) and $\alpha^* = k^*/l$
- Suppose we use $\alpha \approx \alpha^*$ then the bound that we get is

$$L_A \leq L_* + \frac{1}{\eta}((k+1)\ln n + (l-1)(H(\alpha^*) + D_{\mathrm{KL}}(\alpha^* || \alpha)))$$

Where

$$H(\alpha^*) = -\alpha^* \ln \alpha^* - (1-\alpha^*)\ln(1-\alpha^*)$$

$$D_{\mathrm{KL}}(\alpha^* || \alpha) = \alpha^* \ln \frac{\alpha^*}{\alpha}(1-\alpha^*)\ln \frac{1-\alpha^*}{1-\alpha}$$

- This is very close to the loss of the computationally inefficient algorithm.
- For the log loss case this is essentially optimal.
- Not so for square loss!

What can we hope to improve?

- In the fixed-share algorithm, the weight of a suboptimal expert never decreases below α/n.
- The algorithm does not concentrate only on the best expert, even if the last switch is in the distant past.
- The regret depends on the length of the sequence.

The idea of variable-share

- Let the fraction of the total weight given to the best expert get arbitrarily close to 1.
- we can get a regret bound that depends only on the number of switches, not on the lenght of the sequence.
- Requires that the loss be bounded.
- Works for square loss, but not for log loss!

Variable-share

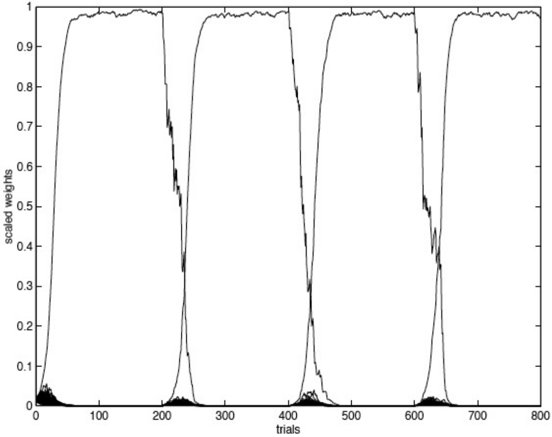
$$
\begin{aligned}
pool &= \sum_{i=1}^{n} \left(1 - (1-\alpha)^{\ell_{t,i}}\right) w^m_{t,i} \\
w^s_{t+1,i} &= (1-\alpha)^{\ell_{t,i}} w^m_{t,i} + \frac{1}{n-1}\left(pool - \left(1 - (1-\alpha)^{\ell_{t,i}}\right) w^m_{t,i}\right)
\end{aligned}
$$

If $\ell_{t,i} = 0$, then expert i does not contribute to the pool.
Expert can get fraction of the total weight arbitrarily close to 1.
Shares the weight quickly if $\ell_{t,i} > 0$

Bound for variable share

- $$\frac{1}{\eta}\ln n+\left(1+\frac{1}{(1-\alpha)\eta}\right)L_*+k\left(1+\frac{1}{\eta}\left(\ln n-1+\ln\frac{1}{\alpha}+\ln\frac{1}{1-\alpha}\right)\right.$$
- α should be tuned so that it is (close to) $\frac{k}{2k+L_*}$

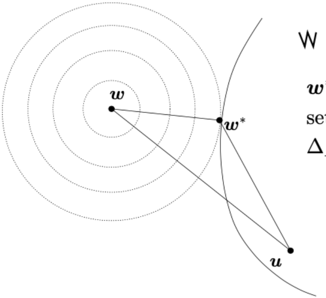
An experiment using variable share



Switching within a small subset

- Suppose the best switching sequence is repeatedly switching among a small subset of the experts $n' \ll n$
- In the context of speech recognition - the speaker repeatedly uses a small number of phonemes.
- If we know the subset, we can pay $\ln n'$ per switch rather than $\ln n$
- Can track switches much more closely.
- Easy to describe an inefficient algorithm (consider all $\binom{n}{n'}$ subsets.)
- Switching to Slides from Manfred Warmuth.

A Pythagorean Theorem [Br,Cs,A,HW]



$\boldsymbol{w}^*$ is projection of $\boldsymbol{w}$ onto convex set $\mathcal{W}$ w.r.t. Bregman divergence Δ_F:

$$\boldsymbol{w}^* = \operatorname*{argmin}_{\boldsymbol{u} \in \mathcal{W}} \Delta_F(\boldsymbol{u}, \boldsymbol{w})$$

Theorem:

$$\Delta_F(\boldsymbol{u}, w) \geq \Delta_F(\boldsymbol{u}, \boldsymbol{w}^*) + \Delta_F(\boldsymbol{w}^*, \boldsymbol{w})$$

Key Idea: Projection onto a convex set reduces divergence.

Bounding Regret Using the Pythagorean Inequality

Using the inequality:

$$D_R(w^* \| w_t) \geq D_R(w^* \| w_{t+1}) + D_R(w_{t+1} \| w_t),$$

we obtain:

$$\sum_{t=1}^{T} \langle w_t - w^*, z_t \rangle \leq D_R(w^* \| w_1) + \sum_{t=1}^{T} D_R(w_t \| w_{t+1}).$$

Interpretation:

- Switching to a better expert is controlled by $D_R(w_t \| w_{t+1})$.
- The total regret depends on the regularizer $R(w)$.

Application: Multiplicative Weights

For entropy-based regularization $R(w) = \sum_i w_i \log w_i$, the update rule is:

$$w_{t+1}^i = \frac{w_t^i e^{-\eta z_t^i}}{\sum_j w_t^j e^{-\eta z_t^j}}.$$

KL Divergence and Pythagorean Inequality:

$$D_{KL}(w^* \| w_t) \geq D_{KL}(w^* \| w_{t+1}) + D_{KL}(w_{t+1} \| w_t).$$

Conclusion:

- Exponential weight updates minimize regret.
- Smaller switching cost improves performance.

Fixed Share Algorithm

Fixed Share is an online learning strategy that allows switching between experts.
Update Rule:

$$w_{t+1}^{i} = (1-\alpha)\frac{w_t^i e^{-\eta z_t^i}}{\sum_j w_t^j e^{-\eta z_t^j}} + \frac{\alpha}{N}$$

where:

- α is the fixed share parameter controlling switching frequency.
- N is the number of experts.

Key Property: Distributes small weight to all experts, preventing early commitment.

Variable Share Algorithm

Variable Share improves upon Fixed Share by dynamically adjusting the switching rate.
Update Rule:

$$w_{t+1}^{i} = (1 - \alpha_t) \frac{w_t^i e^{-\eta z_t^i}}{\sum_j w_t^j e^{-\eta z_t^j}} + \alpha_t S_t^i$$

where:

- α_t is an **adaptive** switching rate.
- S_t^i redistributes weight based on past performance.

Key Property: More efficient than Fixed Share for non-stationary environments.

Comparison: Fixed vs. Variable Share

- **Fixed Share:** Constant switching rate α.
- **Variable Share:** Adaptive switching rate α_t based on history.
- **Fixed Share is simpler** but **can be suboptimal** in dynamic settings.
- **Variable Share performs better** in **changing environments**.

Summary

- The **Generalized Pythagorean Inequality** helps bound regret in expert learning.
- The **divergence term** controls switching cost.
- **Fixed Share**: Constant switching rate improves robustness.
- **Variable Share**: Adaptive switching rate handles non-stationarity.
- **Smooth switching** leads to better regret bounds.

Effect on Expert Learning

Standard Mirror Descent:

$$w_{t+1} = \arg\min_{w} \left[\eta \sum_{s=1}^{t} \langle w, z_s \rangle + D_R(w \| w_1) \right]$$

Regret Bound:

$$\sum_{t=1}^{T} \langle w_t - w^*, z_t \rangle \leq D_R(w^* \| w_1) + \sum_{t=1}^{T} D_R(w_t \| w_{t+1}).$$

The Pythagorean inequality controls regret by bounding divergence.

Fixed Share Algorithm

Fixed Share Update:

$$w_{t+1}^i = (1-\alpha)\frac{w_t^i e^{-\eta z_t^i}}{\sum_j w_t^j e^{-\eta z_t^j}} + \frac{\alpha}{N}.$$

Impact on Pythagorean Inequality:

$$D_R(w^* \| w_t) \geq D_R(w^* \| w_{t+1}) + D_R(w_{t+1} \| w_t).$$

Modification: The divergence $D_R(w_{t+1} \| w_t)$ increases due to the uniform mixing factor α.

Regret Bound:

$$\sum_{t=1}^{T} \langle w_t - w^*, z_t \rangle \leq D_R(w^* \| w_1) + \sum_{t=1}^{T} \left[D_R(w_t \| w_{t+1}) + \alpha D_{KL}(w_t \| u) \right].$$

Variable Share Algorithm

Variable Share Update:

$$w_{t+1}^i = (1 - \alpha_t) \frac{w_t^i e^{-\eta z_t^i}}{\sum_j w_t^j e^{-\eta z_t^j}} + \alpha_t S_t^i.$$

Impact on Pythagorean Inequality:

$$D_R(w^* \| w_t) \geq D_R(w^* \| w_{t+1}) + D_R(w_{t+1} \| w_t) + \alpha_t D_{KL}(w_t \| u).$$

Modification: The divergence term now depends on α_t, making it adaptive rather than constant.

Regret Bound:

$$\sum_{t=1}^{T} \langle w_t - w^*, z_t \rangle \leq D_R(w^* \| w_1) + \sum_{t=1}^{T} \left[D_R(w_t \| w_{t+1}) + \alpha_t D_{KL}(w_t \| u) \right].$$

Comparison: Fixed Share vs. Variable Share

Algorithm	Effect on Pythagorean Inequality	Switchin
Fixed Share	Constant divergence increase	$\alpha D_{KL}(w_t \| u$
Variable Share	Adaptive divergence term	$\alpha_t D_{KL}(w_t \| u)$

Key Differences:

- **Fixed Share:** Adds a fixed switching cost, increasing divergence uniformly.
- **Variable Share:** Dynamically adjusts the divergence penalty, reducing unnecessary switching.

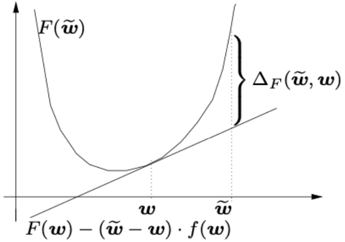
Summary

- The **Pythagorean Inequality** ensures **divergence decreases over time**.
- **Fixed Share**: Constant switching cost increases divergence.
- **Variable Share**: Adaptive switching cost reduces unnecessary updates.
- **Fixed Share is simpler**, but **Variable Share performs better in non-stationary settings**.

Bregman Divergences [Br,CL,Cs]

For any differentiable convex function F

$$\begin{aligned}\Delta_F(\widetilde{\boldsymbol{w}}, \boldsymbol{w}) &= F(\widetilde{\boldsymbol{w}}) - F(\boldsymbol{w}) - (\widetilde{\boldsymbol{w}} - \boldsymbol{w}) \cdot \underbrace{\nabla_{\boldsymbol{w}} F(\boldsymbol{w})}_{f(\boldsymbol{w})} \\ &= F(\widetilde{\boldsymbol{w}}) - \begin{array}{c}\text{supporting hyperplane} \\ \text{through } (\boldsymbol{w}, F(\boldsymbol{w}))\end{array}\end{aligned}$$



Bregman Divergences: Simple Properties

1. $\Delta_F(\widetilde{\boldsymbol{w}}, \boldsymbol{w})$ is convex in $\widetilde{\boldsymbol{w}}$
2. $\Delta_F(\widetilde{\boldsymbol{w}}, \boldsymbol{w}) \geq 0$
   If F convex equality holds iff $\widetilde{\boldsymbol{w}} = \boldsymbol{w}$
3. Usually not symmetric: $\Delta_F(\widetilde{\boldsymbol{w}}, \boldsymbol{w}) \neq \Delta_F(\boldsymbol{w}, \widetilde{\boldsymbol{w}})$
4. Linearity (for $a \geq 0$):
   $\Delta_{F+a\,H}(\widetilde{\boldsymbol{w}}, \boldsymbol{w}) = \Delta_F(\widetilde{\boldsymbol{w}}, \boldsymbol{w}) + a\,\Delta_H(\widetilde{\boldsymbol{w}}, \boldsymbol{w})$
5. Unaffected by linear terms ($a \in \mathbf{R}$, $\boldsymbol{b} \in \mathbf{R}^n$):
   $\Delta_{H+a\widetilde{\boldsymbol{w}}+\boldsymbol{b}}(\widetilde{\boldsymbol{w}}, \boldsymbol{w}) = \Delta_H(\widetilde{\boldsymbol{w}}, \boldsymbol{w})$

Bregman Divergences: more properties

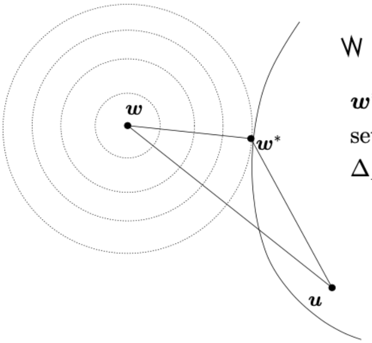
6. $\nabla_{\widetilde{\boldsymbol{w}}} \Delta_F(\widetilde{\boldsymbol{w}}, \boldsymbol{w})$

$$
\begin{aligned}
&= \nabla F(\widetilde{\boldsymbol{w}}) - \nabla_{\widetilde{\boldsymbol{w}}}(\widetilde{\boldsymbol{w}} \nabla_{\boldsymbol{w}} F(\boldsymbol{w})) \\
&= f(\widetilde{\boldsymbol{w}}) - f(\boldsymbol{w})
\end{aligned}
$$

7. $\Delta_F(\boldsymbol{w}_1, \boldsymbol{w}_2) + \Delta_F(\boldsymbol{w}_2, \boldsymbol{w}_3)$

$$
\begin{aligned}
&= F(\boldsymbol{w}_1) - F(\boldsymbol{w}_2) - (\boldsymbol{w}_1 - \boldsymbol{w}_2) f(\boldsymbol{w}_2) \\
&\quad F(\boldsymbol{w}_2) - F(\boldsymbol{w}_3) - (\boldsymbol{w}_2 - \boldsymbol{w}_3) f(\boldsymbol{w}_3) \\
&= \Delta_F(\boldsymbol{w}_1, \boldsymbol{w}_3) + (\boldsymbol{w}_1 - \boldsymbol{w}_2) \cdot (f(\boldsymbol{w}_3) - f(\boldsymbol{w}_2))
\end{aligned}
$$

A Pythagorean Theorem [Br,Cs,A,HW]



$\boldsymbol{w}^*$ is projection of $\boldsymbol{w}$ onto convex set $\mathcal{W}$ w.r.t. Bregman divergence Δ_F:

$$\boldsymbol{w}^* = \operatorname*{argmin}_{\boldsymbol{u} \in \mathcal{W}} \Delta_F(\boldsymbol{u}, \boldsymbol{w})$$

Theorem:

$$\Delta_F(\boldsymbol{u}, w) \geq \Delta_F(\boldsymbol{u}, \boldsymbol{w}^*) + \Delta_F(\boldsymbol{w}^*, \boldsymbol{w})$$

Unnormalized Relative entropy

- prediction, outcome $\mathbf{p}, \mathbf{q}$ are n dimensional vectors with non-negative coordinates.
- Loss is RE extended to non-negative vectors:

$$\mathrm{RE}\,(\mathbf{p} \;\|\; \mathbf{q}) = \sum_{i=1}^{n} p_i \log \frac{p_i}{q_i} - \sum_{i=1}^{n} (q_i - p_i)$$

  Coincides with RE when $\sum_{i=1}^{n} p_i = \sum_{i=1}^{n} q_i = 1$

- Unnormalized RE is the Bregman divergence corresponding to the unnormalized entropy:

$$F(\mathbf{p}) = \sum_{i=1}^{n} p_i \log p_i - \sum_{i=1}^{n} p_i$$

Inequalities for Unnormalized Relative entropy

- No triangle inequality
  $\exists \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3 \;\; \mathrm{RE}(\mathbf{p}_1 \parallel \mathbf{p}_3) > \mathrm{RE}(\mathbf{p}_1 \parallel \mathbf{p}_2) + \mathrm{RE}(\mathbf{p}_2 \parallel \mathbf{p}_3)$
- Generalized Pythagorean inequality For any closed convex set S and any point $\mathbf{p}_1 \notin S$, define the projection of $\mathbf{p}_1$ on S to be $\mathbf{p}_2 = \mathrm{argmin}_{\mathbf{u} \in S} \mathrm{RE}(\mathbf{p}_1 \parallel \mathbf{u})$, then:

$$\forall \mathbf{p}_3 \in S; \;\; \mathrm{RE}(\mathbf{p}_1 \parallel \mathbf{p}_3) \geq \mathrm{RE}(\mathbf{p}_1 \parallel \mathbf{p}_2) + \mathrm{RE}(\mathbf{p}_2 \parallel \mathbf{p}_3)$$

half squared euclidean distance

- prediction, outcome $\mathbf{u}, \mathbf{v} \in R^n$
- $$\lambda_{\text{sq}}(\mathbf{u}, \mathbf{v}) = \frac{1}{2}\|\mathbf{u} - \mathbf{v}\|^2 = \frac{1}{2}\sum_{i=1}^{n}(u_i - v_i)^2$$
- Bregman divergence with respect to the square euclidean norm
$$\|\mathbf{v}\|_2$$
- Triangle inequality does not hold.
- Pythagoras inequality : For any closed convex set S and any point $\mathbf{v}_1 \notin S$, define the projection of $\mathbf{v}_1$ on S to be $\mathbf{v}_2 = \text{argmin}_{\mathbf{u} \in S}\|\mathbf{v}_1 - \mathbf{u}\|^2$, then:
$$\forall \mathbf{v}_3 \in S; \quad \|\mathbf{v}_1 - \mathbf{v}_3\|^2 \geq \|\mathbf{v}_1 - \mathbf{v}_2\|^2 + \|\mathbf{v}_2 - \mathbf{v}_3\|^2$$

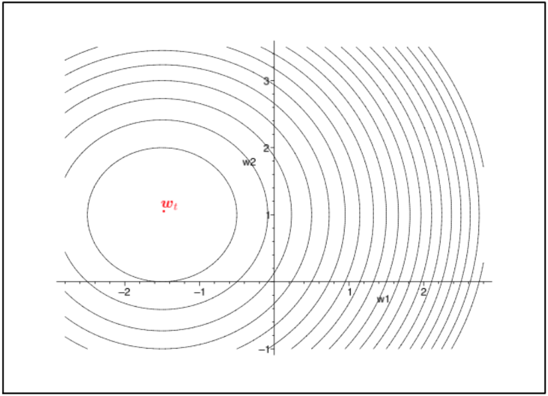
Divergence: Euclidean Distance Squared

$$\Delta_F(\boldsymbol{w}, \boldsymbol{w}_t) = \|\boldsymbol{w} - \boldsymbol{w}_t\|_2^2/2$$

$\boldsymbol{w}_t = (-3/2, 1)$

$\boldsymbol{x}_t = (1, -0.5)$

$y_t = 1$



Bregman divergence regularization

- Idea: Set $\mathbf{w}_{t+1}$ to be $\mathbf{u}$ that minimizes:

$$\Delta_F(\mathbf{w}_t, \mathbf{u}) + \alpha \ell_t(\mathbf{u})$$

- In general, hard to compute the minimum.
- Efficient approximation Mirror Descent. Will be covered later.

General Motivation of Updates [KW]

Trade-off between two term:

$$w_{t+1} = \underset{w}{\operatorname{argmin}}\left(\underbrace{\Delta_F(w, w_t)}_{\textit{weight domain}} + \eta_t \underbrace{L_t(w)}_{\textit{label domain}} \right)$$

$\Delta_F(w, w_t)$ is "regularization term" and serves as measure of progress in the analysis.

When loss L is convex (in w)

$$\nabla_w(\Delta_F(w, w_t) + \eta_t L_t(w)) = 0$$

iff

$$f(w) - f(w_t) + \eta_t \underbrace{\nabla L_t(w)}_{\approx \nabla L_t(w_t)} = 0$$

$$\Rightarrow \quad w_{t+1} = f^{-1}\left(f(w_t) - \eta_t \nabla L_t(w_t)\right)$$

How to prove relative loss bounds?

Loss: $L_t(\boldsymbol{w}) = L((\boldsymbol{x}_t, y_t), \boldsymbol{w})$ convex in $\boldsymbol{w}$

Divergence: $\Delta_F(\boldsymbol{u}, \boldsymbol{w}) = F(\boldsymbol{u}) - F(\boldsymbol{w}) - (\boldsymbol{u} - \boldsymbol{w}) \cdot f(\boldsymbol{w})$

Update: $f(\boldsymbol{w}_{t+1}) - f(\boldsymbol{w}_t) = -\eta \nabla_{\boldsymbol{w}} L_t(\boldsymbol{w}_t)$

$$
\begin{aligned}
L_t(\boldsymbol{u}) &\overbrace{\geq}^{\text{convexity}} L_t(\boldsymbol{w}_t) + (\boldsymbol{u} - \boldsymbol{w}_t) \cdot \underbrace{\nabla_{\boldsymbol{w}} L_t(\boldsymbol{w}_t)}_{\text{update}} \\
&= L_t(\boldsymbol{w}_t) - \frac{1}{\eta} \underbrace{(\boldsymbol{u} - \boldsymbol{w}_t) \cdot (f(\boldsymbol{w}_{t+1}) - f(\boldsymbol{w}_t))}_{\text{prop. 7 of } \Delta_F} \\
&= L_t(\boldsymbol{w}_t) + \frac{1}{\eta} (\Delta_F(\boldsymbol{u}, \boldsymbol{w}_{t+1}) - \Delta_F(\boldsymbol{u}, \boldsymbol{w}_t) - \Delta_F(\boldsymbol{w}_t, \boldsymbol{w}_{t+1}))
\end{aligned}
$$