## Toll/Congestion

Given: $G = (V, E)$.
Given $(s_1, t_1) \ldots (s_k, t_k)$.

Problem: Route path for each pair and minimize maximum congestion.

Congestion is maximum number of paths that use any edge.

Note: Number of paths is exponential.
Can encode in polysized linear program, but large.

## Toll/Congestion

Given: $G = (V, E)$.
Given $(s_1, t_1) \ldots (s_k, t_k)$.
Row: choose routing of all paths. (Exponential)
Column: choose edge.
Row pays if column chooses edge on any path.

Matrix:
row for each routing: $r$
column for each edge: $e$

$A[r, e]$ is congestion on edge $e$ by routing $r$

**Offense: (Best Response.)**
Router: route along shortest paths.
Toll: charge most loaded edge.

**Defense:** Toll: maximize shortest path under tolls.
Route: minimize max congestion on any edge.

## Two person game.

Row is router.

An exponential number of rows!

Two person game with experts won't be so easy to implement.

Version with row and column flipped may work.

$A[e, r]$ - congestion of edge $e$ on routing $r$.

$m$ rows. Exponential number of columns.

Multiplicative Weights only maintains $m$ weights.

Adversary only needs to provide best column each day.

Runtime only dependent on $m$ and $T$ (number of days.)

## Congestion minimization and Experts.

Will use gain and $[0, \rho]$ version of experts:

$G \geq (1 - \varepsilon)G^* - \frac{\rho \log n}{\varepsilon}$.

Let $T = \frac{k \log n}{\varepsilon^2}$

1. Row player runs multiplicative weights on edges:
   $w_i = w_i(1 + \varepsilon)^{g_i/k}$.
2. Column routes all paths along shortest paths.
3. Output the average of all routings: $\frac{1}{T}\sum_t f(t)$.

**Claim:** The congestion, $c_{max}$ is at most $C^* + 2k\varepsilon$.

Proof:
$G \geq G^*(1 - \varepsilon) - \frac{k \log n}{\varepsilon T} \rightarrow G^* - G \leq \varepsilon G^* + \frac{k \log n}{\varepsilon}$

$G^* = T * c_{\max}$ – Best row payoff against average routing (times $T$).

$G \leq T \times C^*$ – each day, gain is avg. congestion $\leq$ opt congestion.

$T = \frac{k \log n}{\varepsilon^2} \rightarrow Tc^*_{\max} - TC \leq \varepsilon TC^* + \frac{k \log n}{\varepsilon} \quad \rightarrow$
$c_{max} - C^* \leq \varepsilon C^* + \varepsilon$ □

## Better setup.

Runtime: $O(km \log n)$ to route in each step (using Dijkstra's)

$O(\frac{k \log n}{\varepsilon^2})$ steps
to get $c_{\max} - C^* < \varepsilon C^*$ (assuming $C^* > 1$) approximation.

To get constant $c$ error.
$\rightarrow O(k^2 m \log n/\varepsilon^2)$ to get a constant approximation.
Exercise: $O(km \log n/\varepsilon^2)$ algorithm ! ! !

## Fractional versus Integer.

Did we (approximately) solve path routing?
Yes? No?

No! Average of $T$ routings.
We approximately solved fractional routing problem.

No solution to the path routing problem that is $(1 + \varepsilon)$ optimal!

Decent solution to path routing problem?

For each $s_i, t_i$, choose path $p_i$ at random from "daily" paths.

Congestion $c(e)$ edge has expected congestion, $\tilde{c}(e)$, of $c(e)$.

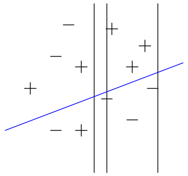
"Concentration" (law of large numbers)
$c(e)$ is relatively large ($\Omega(\log n)$)
$\rightarrow \tilde{c}(e) \approx c(e)$.

Concentration results? later.

## Learning

Learning just a bit.

Example: set of labelled points, find hyperplane that separates.



Looks hard.

$1/2$ of them? Easy.
Arbitrary line. And Scan.

Useless. A bit more than $1/2$ Correct would be better.

Weak Learner: Classify $\ge \frac{1}{2}+\varepsilon$ points correctly.

Not really important but ...

## Weak Learner/Strong Learner

Input: $n$ labelled points.

Weak Learner:
produce hypothesis correctly classifies $\frac{1}{2}+\varepsilon$ fraction

Strong Learner:
produce hyp. correctly classifies $1+\mu$ fraction

That's a really strong learner!

Strong Learner:
produce hypothesis correctly classifies $1-\mu$ fraction

Same thing?

Can one use weak learning to produce strong learner?

Boosting: use a weak learner to produce strong learner.

## Poll.

Given a weak learning method (produce ok hypotheses.)
produce a great hypothesis.

Can we do this?

(A) Yes

(B) No

If yes. How?

The idea: Multiplicative Weights.
Standard online optimization method reinvented in many areas.

## Boosting/MW Framework

Points lose when classified correctly.
The little devils want to fool the learner.

Learner classifies weighted majority of points correctly.

Strong learner algorithm from many weak learners!

**Initialize:** all points have weight 1.

Do $T = \frac{2}{\varepsilon^2}\ln\frac{1}{\mu}$ rounds

1. Find $h_t(\cdot)$ correct on $1/2+\gamma$ of weighted points.
2. Multiply each point that is correct by $(1-\varepsilon)$.

Output hypotheses $h(x)$: majority of $h_1(x), h_2(x), \ldots, h_T(x)$.

**Claim:** $h(x)$ is correct on $1-\mu$ of the points ! ! !

Cool!

Really? Proof?

## Logarithm

$ln(1-x) = (-x - x^2/2 - x^3/3 \ldots.)$ Taylors formula for $|x|<1$.

Implies: for $x \le 1/2$, that $-x-x^2 \le \ln(1-x) \le -x$.

The first inequality is from geometric series.

$x^3/3 + \ldots = x^2(x/3 + x^2/4 + ..) \le x^2(1/2)$ for $|x| < 1/2$.

The second is from truncation.

Second implies: $(1-\varepsilon)^x \le e^{-\varepsilon x}$, by exponentiation.

## Adaboost proof.

**Claim:** $h(x)$ is correct on $1-\mu$ of the points!

Let $S_{bad}$ be the set of points where $h(x)$ is incorrect.

majority of $h_t(x)$ are wrong for $x \in S_{bad}$.

point $x \in S_{bad}$ is winning – loses less than $\frac{1}{2}$ the time.

$W(T) \ge (1-\varepsilon)^{\frac{T}{2}}|S_{bad}|$

Each day $t$, weak learner penalizes $\ge \frac{1}{2}+\gamma$ of the weight.

Loss $L_t \ge (1/2+\gamma)$

$\rightarrow W(t+1) \le W(t)(1-\varepsilon(L_t)) \le W(t)e^{-\varepsilon L_t}$

$\rightarrow W(T) \le ne^{-\varepsilon \sum_t L_t} \le ne^{-\varepsilon(\frac{1}{2}+\gamma)T}$

Combining

$|S_{bad}|(1-\varepsilon)^{T/2} \le W(T) \le ne^{-\varepsilon(\frac{1}{2}+\gamma)T}$

## Calculation..

$|S_{bad}|(1-\varepsilon)^{T/2} \leq ne^{-\varepsilon(\frac{1}{2}+\gamma)T}$

Set $\varepsilon = \gamma$, take logs.

$\ln\left(\frac{|S_{bad}|}{n}\right) + \frac{T}{2}\ln(1-\gamma) \leq -\gamma T(\frac{1}{2}+\gamma)$

Again, $-\gamma - \gamma^2 \leq \ln(1-\gamma)$,

$\ln\left(\frac{|S_{bad}|}{n}\right) + \frac{T}{2}(-\gamma-\gamma^2) \leq -\gamma T(\frac{1}{2}+\gamma) \to \ln\left(\frac{|S_{bad}|}{n}\right) \leq -\frac{\gamma^2 T}{2}$

And $T = \frac{2}{\gamma^2}\log\mu$,

$\to \ln\left(\frac{|S_{bad}|}{n}\right) \leq \log\mu \to \frac{|S_{bad}|}{n} \leq \mu$.

The misclassified set is at most $\mu$ fraction of all the points.

The hypothesis correctly classifies $1-\mu$ of the points !

**Claim:** Multiplicative weights: $h(x)$ is correct on $1-\mu$ of the points!

## Some details...

Weak learner learns over distributions of points not points.

Make copies of points to simulate distributions.

Used often in machine learning.
Blending learning methods.

## Theme:Good on average, hyperplane.

“Duality”

$\min cx, Ax \geq b, x \geq 0$.

Linear combination of constraints: $y^T Ax \geq y^T c$
Find a solution for just one constraint!!!
Best response.

Multiplicative weights: two person games (linear programs)
$y$ is exponential weights on “how unsatisfied” each equation is.
$y_i \propto \sum_t (1+\varepsilon)^{(a_i x^{(t)} - b_i)}$
$y$ “wins” $\equiv$ unsatisfiable linear combo of constraints.
Otherwise, $x$ eventually “wins”.
Or pair that are pretty close.

(Apologies: switched $x$ and $y$ in game setup.)

“Separating” Hyperplane?
$y^T$ “separates” affine subspace $Ax$ from $\geq y^T c$.
Or doesn’t and $x$ responds.

The math: $e = \lim_{n\to\infty}(1+1/n)^n$.

## A step closer.

Another Algorithm.

Finding a feasible point: $x^*$ for constraints.
If $x^{(t)}$ point violates constraint by $> \varepsilon$
move toward constraint.
Closer.

The Math:
Wrong side, angle to correct point is less than $90^\circ$

This is the idea in perceptron. But can do analysis directly.

## Multiplicative weights and a step closer.

The solution is a distribution: $p^*$.

Every day each strategy loses (or not), $\ell_i^{(t)}$.
Assumption: Solution doesn’t lose (much).

MW: keeps a distribution.

Closer?
Distance (divergence) from $q$ to $p$ $\sum_i p_i^* \log(p_i^*/q_i)$.
Step in MW gets closer to $p^*$ with this distance.
Idea: $p^*$ loses less,
so new distribution plays losers less.
Move toward playing losers less.
Thus closer to $p^*$.

The math:
linear (and quadratic) approximation of $e^x$.
Advantage?
Distributions have entropy at most $O(\log n)$.

## Reinforcement learning == Bandits.

Multiplicative Weights framework:
Update all experts.

Bandits.
Only update expert you choose.
No information about others.
(Named after one-armed bandit slot machine.)

Idea: “Learn” which expert is best.
Prof. Dragan’s mantra: formulation as optimization.

Exploration: choose new bandit to get “data”.
Exploitation: choose best bandit.

Strategy:
Multiplicative weights.
Update by $(1+\varepsilon)$.

Big $\varepsilon$.
Exploit or explore more? Exploit.

Perceptron also like bandits. One point at a time.

Online optimization: limited information.

## Next up: convex optimization.

Analysis of previous.
Get closer to a feasible point.

Idea: infeasible gives direction to step toward a feasible point.
violation of hyperplane for perceptron.
loss function for multiplicative weights.

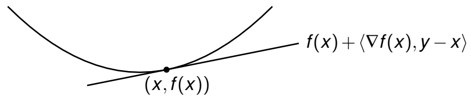
Next: Get closer to an optimal point for function.

## Convex optimization

Slides: Thanks to Di Wang.

$$\min_{x \in Q} f(x)$$
$$f(x) - f(y) \le \langle \nabla f(x), x - y \rangle$$

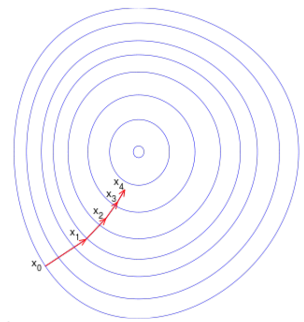
$Q$: feasible space, convex.




First-order Iterative Methods

- Query $x \in Q$, update using $\nabla f(x)$
- Low per-iteration cost, $\text{poly}(\frac{1}{\varepsilon})$ convergence.
- Methods of choice in large-scale regime.

## Gradient Descent




- Moves in down-hill direction.
- Improve objective function value each iteration.
- Output final point.

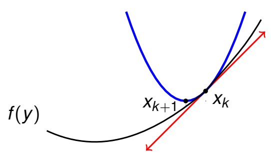
## Gradient Descent

### $L$-Lipschitz continuous

$$\|\nabla f(x) - \nabla f(y)\|_* \le L\|x - y\| \quad \forall x, y \in Q$$

- Global linear lower bound and quadratic upper bound:

$$\forall y \quad f(x) + \langle \nabla f(x), y - x \rangle \le f(y) \le f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2}\|y - x\|^2$$




- Minimize using quadratic bound

$$x_{k+1} = \text{Grad}(x_k) = \underset{x \in Q}{\text{argmin}}\{\langle \nabla f(x_k), x - x_k \rangle + \frac{L}{2}\|x - x_k\|^2\}$$

## Gradient Descent: one dimensional intuition.

Convexity:
$f(x^*) \ge f(x) + \nabla f(x)^T(x^* - x). \implies f(x) \le f(x^*) + \nabla f(x)^T(x - x^*)$

Also: $f(x) - f(x^*) \le \nabla f(x)^T(x - x^*) = gR$

$L$-Lipschitz, $R = \|x_0 - x^*\|$:
$x^+ = x - \frac{1}{L}\nabla f(x) \qquad f(x) - f(x^+) \ge \frac{1}{2L}\|\nabla f(x)\|_*^2$

In one dimension: $\nabla f(x) = g$.

Gap: $gR$. Progress/step: Roughly $g^2/2$.

Idea: Gap/(progress/step) $\implies$ roughly $2LR/g$ steps.

Convexity: $g \ge (f(x) - f(x^*))/R \implies 2LR^2/(f(x) - f(x^*))$ steps.

While gap $f(x) - f(x^*) \ge \varepsilon$ we have $g \ge \varepsilon/R$.

$\implies O(LR^2/\varepsilon)$ steps reduce gap by $1/2$.

## Gradient Descent: convergence in $\ell_2$

Convexity:
$f(x^*) \ge f(x) + \nabla f(x)^T(x^* - x). \implies f(x) \le f(x^*) + \nabla f(x)^T(x - x^*)$

$L$-Lipschitz, $R = \|x_0 - x^*\|$:
$x^+ = x - \frac{1}{L}\nabla f(x) \qquad f(x) - f(x^+) \ge \frac{1}{2L}\|\nabla f(x)\|_*^2$

$f(x^+) \le f(x^*) + \nabla f(x)^T(x - x^*) - \frac{1}{2L}\|\nabla f(x)\|_2^2$

$\implies f(x^+) - f(x^*) \le \frac{L}{2}(\frac{2}{L}\nabla f(x)^T(x - x^*) - \frac{1}{L^2}\|\nabla f(x)\|_2^2)$

$\le \frac{L}{2}(\frac{2}{L}\nabla f(x)^T(x - x^*) - \frac{1}{L^2}\|\nabla f(x)\|_2^2 - \|x - x^*\|_2^2 + \|x - x^*\|_2^2)$ Add 0

$\le \frac{L}{2}(\|x - x^*\|_2^2 - \|(x - x^*) - \frac{1}{L}\nabla f(x)\|_2^2)$
$\le \frac{L}{2}(\|x - x^*\|_2^2 - \|x^+ - x^*\|_2^2)$

$\sum_k^T f(x_k) - f(x^*) \le \sum_k^T \frac{L}{2}(\|x_{k-1} - x^*\|_2^2 - \|x_k - x^*\|_2^2)$
$\le \frac{L}{2}(\|x_0 - x^*\|_2^2 - \|x_T - x^*\|_2^2) \le \frac{L}{2}\|x_0 - x^*\|_2^2$

$f(x_k)$ is decreasing, we have $f(x_T) \le \frac{1}{T}\sum_k f(x_k)$.
$\implies f(x_T) - f(x^*) \le \frac{LR^2}{2T}$ where $R = \|x_0 - x^*\|$.

Also: $T = O(LR^2/\varepsilon)$ iterations for $f(x_T) - f(x^*) \le \varepsilon$.

## Gradient Descent

**Primal progress**

$$f(x_k) - f(x_{k+1}) \geq \frac{1}{2L} \|\nabla f(x)\|_*^2$$

**Convergence**

$L$-Lipschitz, $R = \max_{x:f(x) \leq f(x_0)} \|x - x^*\|$:

$$f(x_T) - f(x^*) \leq O(\frac{LR^2}{T})$$

To get $\varepsilon$-approximation, need

$$T = O(\frac{LR^2}{\varepsilon})$$

## Relationship?

What is relationship to move closer to feasible?

If wrong side of hyperplane by at least something.
Move to other side.

What is the "hyperplane" here?

$\nabla f(x)$ Maybe.