Games of Prediction

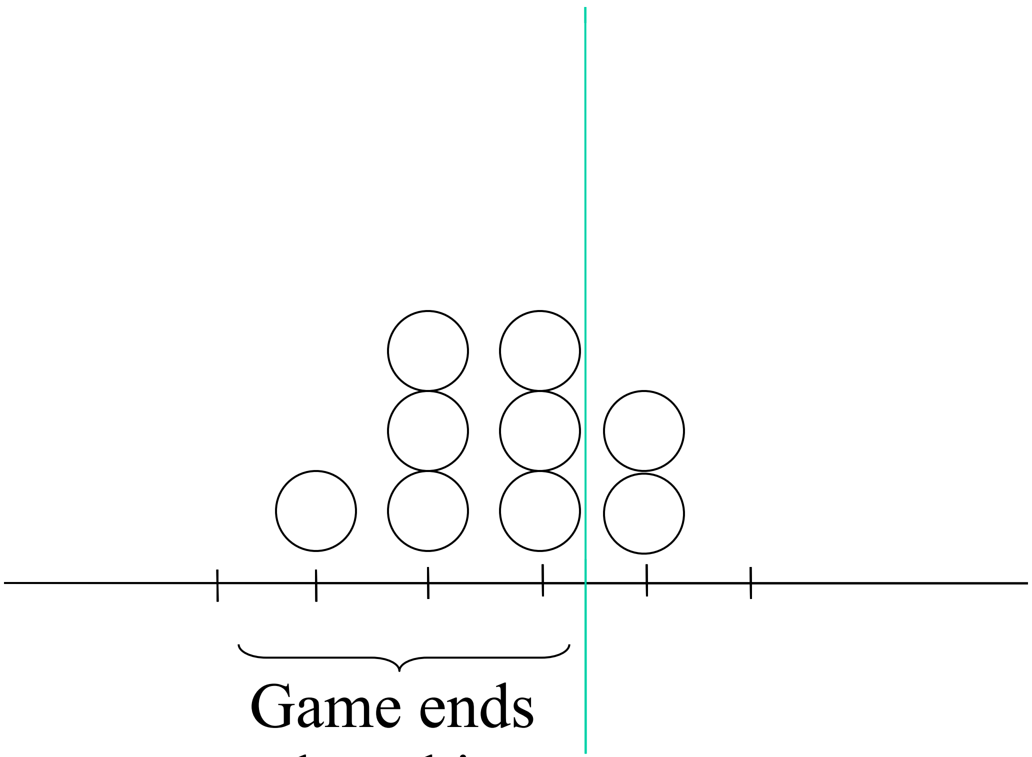
or

Things get simpler as $n \to \infty$

Yoav Freund

Banter Inc.

The Chip game



Game ends when this is empty

Plan of the talk

- Why is the chip game is interesting?
- Letting the number of chips go to infinity.
- The boosting game.
- Drifting games (let chip=sheep).
- Letting the number of rounds go to infinity
- Brownian motion and generalized boosting (nice pictures!)

Combining expert advice

- Binary sequence: 1,0,0,1,1,0,?
- N experts make predictions:

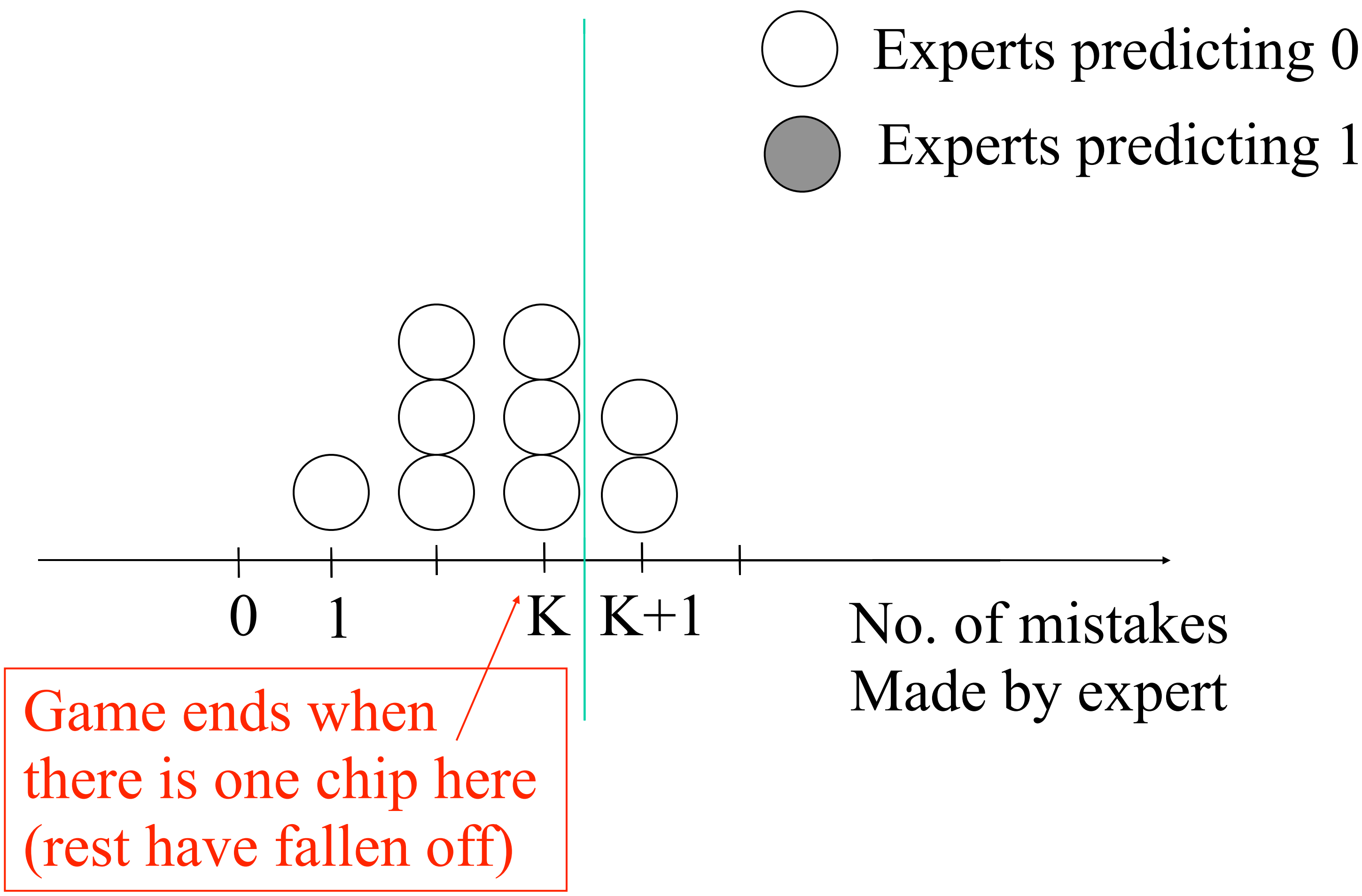
  expert 1: 1,0,1,0,1,1,1

  expert 2: 0,0,0,1,1,0,1

  …

  expert N: 1,0,0,1,1,1,0
- Algorithm makes prediction: 1,0,1,1,0,1,1
- Assumption: there exists an expert which makes at most k mistakes
- Goal: make least mistakes under the assumption (no statistics!)
- Chip = expert, bin = number of mistakes made
- Game step = algorithm's prediction is incorrect.

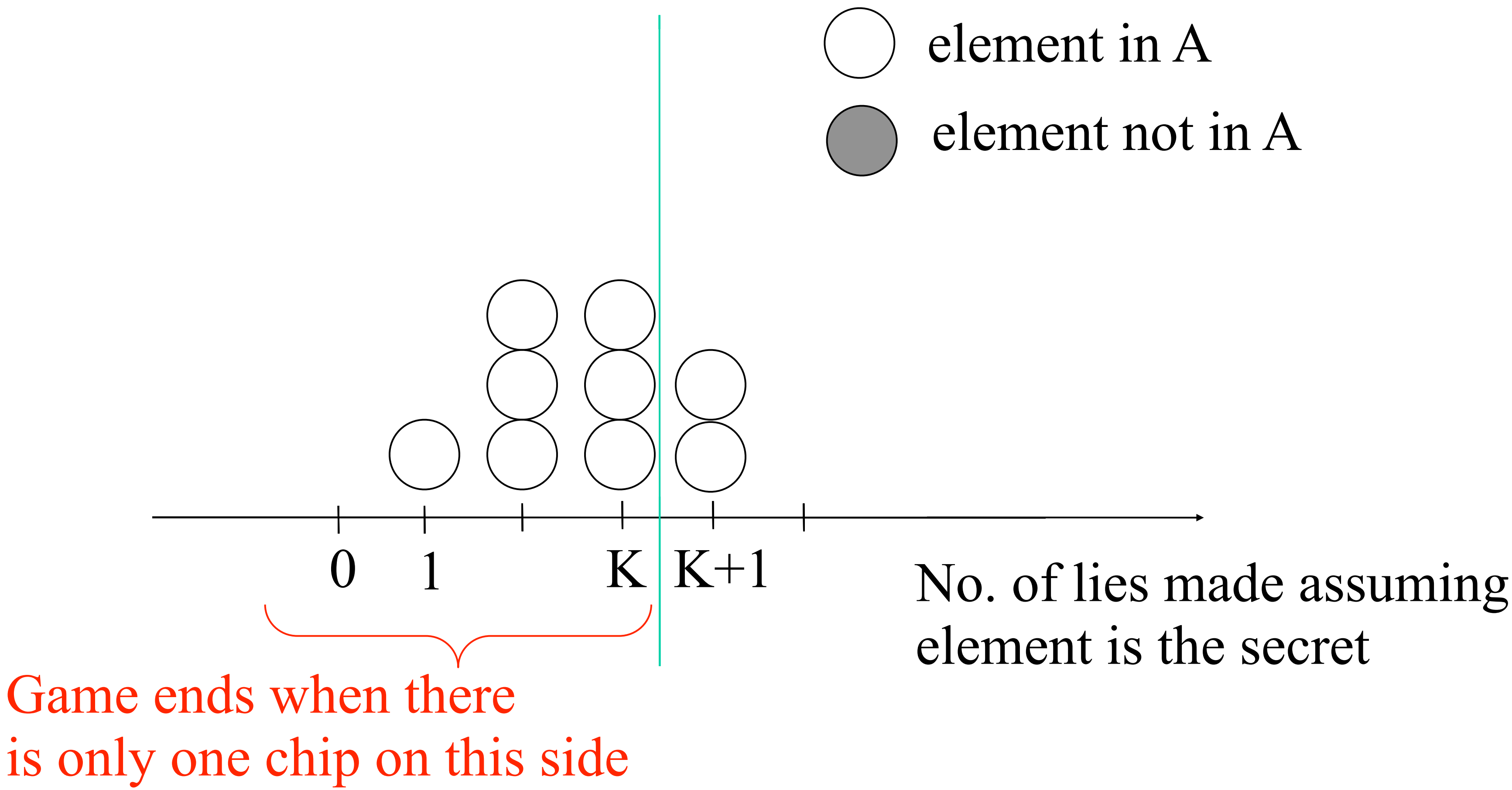
Chip game for expert advice



21 questions with k lies

- Player 1 chooses secret x from S1,S2,…,SN
- Player 2 asks "is x in the set A?"
- Player 1 can lie k times.
- Game ends when player 2 can determine x.
- A smart player 1 aims to have more than one consistent x for as long as possible.
- Chip = element (Si)
- Bin = number of lies made regarding element

Chip game for 20 questions

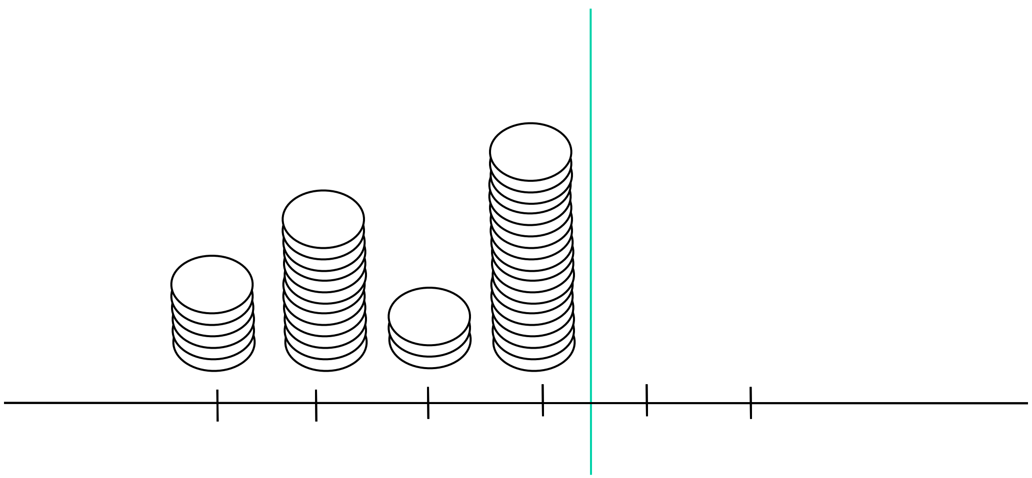


Simple case – all chips in bin 1

- 21 questions without lies
- Combining experts where one is perfect
- Splitting a cookie
- Optimal strategies:
  - Player 1: split chips into two equal parts
  - Player 2: choose larger part
- Note problem when number of chips is odd

Number of chips to infinity

Replace individual chips by chip mass



Optimal splitter strategy:
Split each bin into two equal parts

Binomial weights strategies

- What should chooser do if parts are not equal?
- Assume that on following iterations splitter will play optimally = split each bin into 2 equal parts.
- Future configuration independent of chooser's choice
- Potential: Fraction of bin i that will remain in bins 1..k when m iterations remain $$\binom{m}{\leq k-i} \doteq \frac{1}{2^m}\sum_{j=0}^{k-i}\binom{m}{j}$$
- Choose part so that next configuration will have maximal (or minimal) potential.
- Solve for m: $$m := \max\left\{q \in \mathbb{N} : \sum_{E\in\mathcal{E}}\binom{q}{\leq k - i_E} > 1\right\}$$

Optimality of strategy

- If the chips are infinitely divisible then the solution is min/max optimal
- [Spencer95] Enough for optimality if number of chips is at least

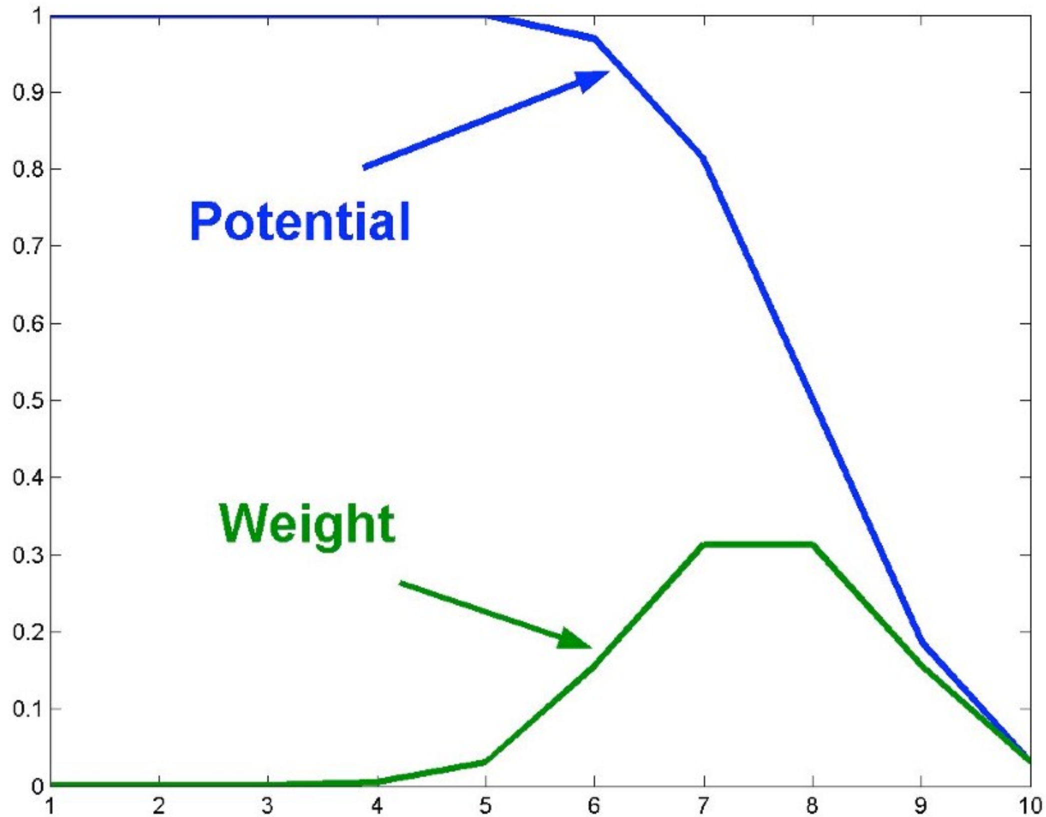
$$\Omega\left(2^{2^k}\right)$$

Equivalence to a random walk

- Both sides playing optimally is equivalent to each chip performing an independent random walk.
- Potential = probability of a chip in bin i ending in bins $1..K$ after m iteration
- Weight = difference between the potentials of a chip in its two possible locations on the following iteration.
- Chooser's optimal strategy: choose set with smaller (larger) weight

Example potential and weight

m=5; k=10



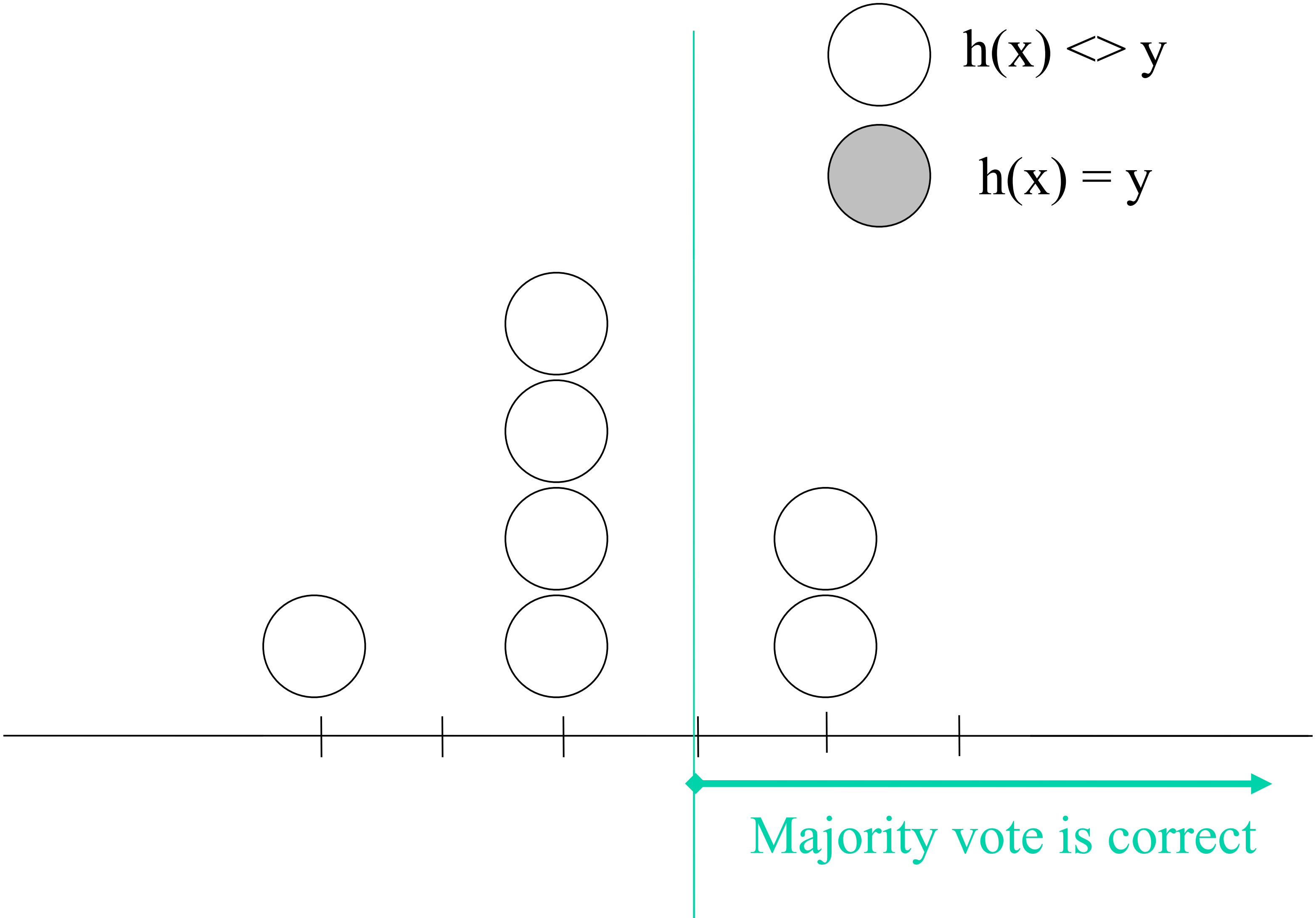
Boosting

- A method for improving classifier accuracy
- Weak Learner: a learning algorithm generating rules slightly better than random guessing.
- Basic idea: re-weight training examples to force weak learner into different parts of the space.
- Combine weak rules by a majority vote.

Boosting as a chip game

- Chip = training example
- Booster assigns a weight to each chip, weights sum to 1.
- Learner selects a subset with weight $\geq \frac{1}{2} + \gamma$
  - Selected set moves a step right (correct)
  - Unselected set moves a step left (incorrect)
- Booster wins examples on right of origin.
  - Implies that majority vote is correct.

The boosting chip game



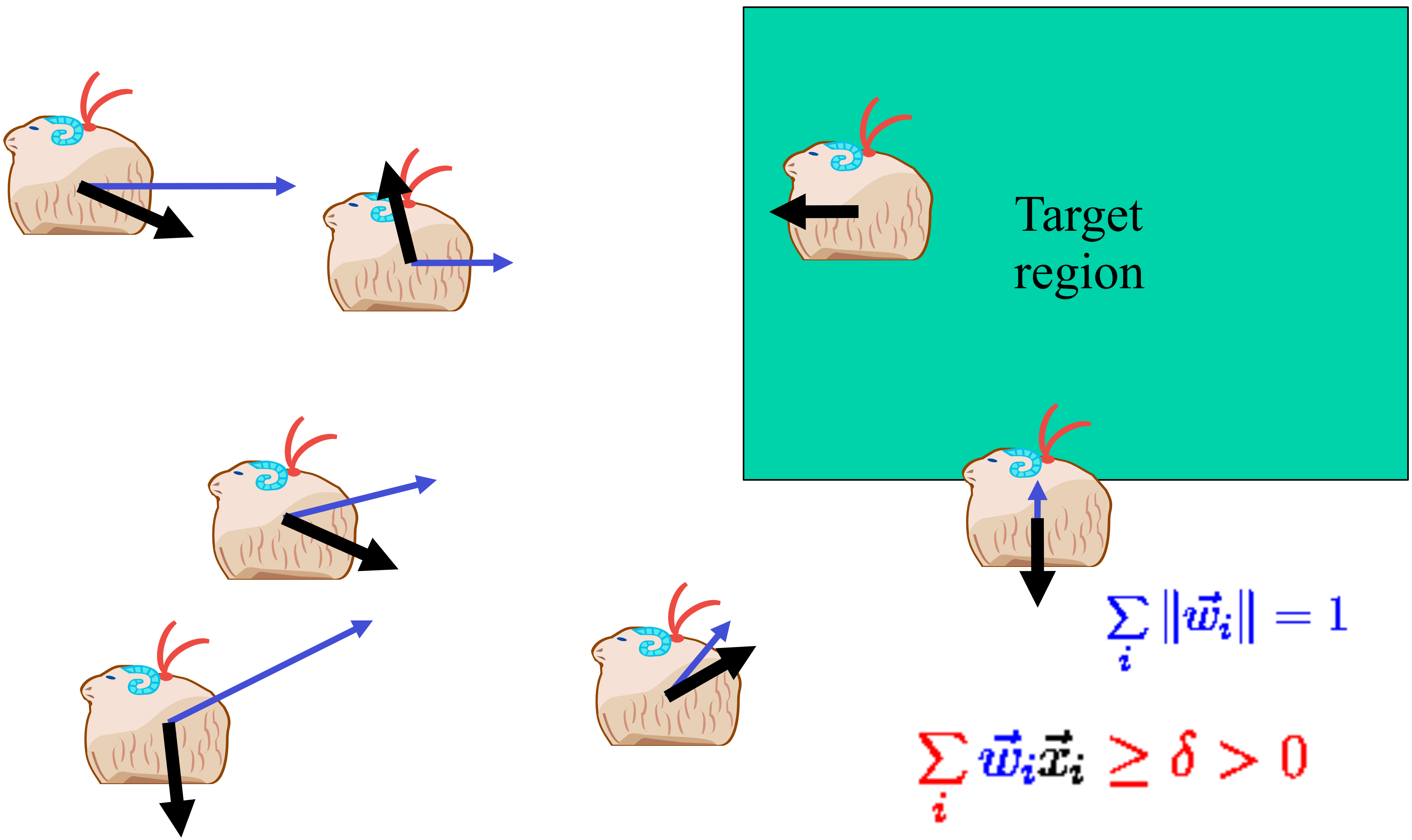
Binomial strategies for boosting

- Learner moves chip right independently with probability $\frac{1}{2}+\gamma$
- The potential of example that is correctly classified r times after i out of k iterations

$$\sum_{j=0}^{\lfloor \frac{k}{2} \rfloor - r} \binom{k-i}{j} \left(\frac{1}{2}+\gamma\right)^j \left(\frac{1}{2}-\gamma\right)^{k-i-j}$$

- Booster assigns to each example a weight proportional to the difference between its possible next-step potentials.

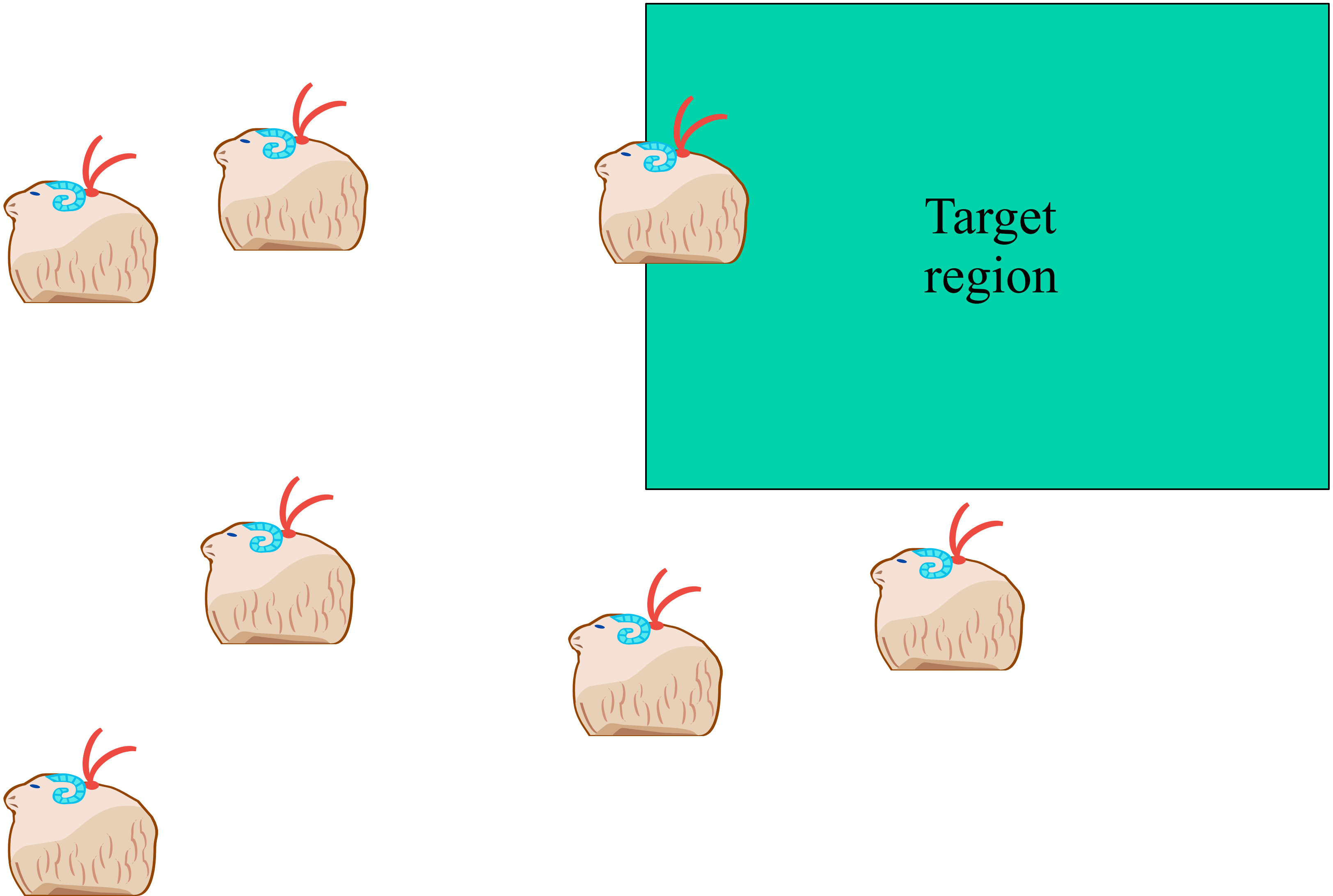
Drifting games (in 2d)



$$\sum_i \|\vec{w}_i\| = 1$$

$$\sum_i \vec{w}_i \vec{x}_i \geq \delta > 0$$

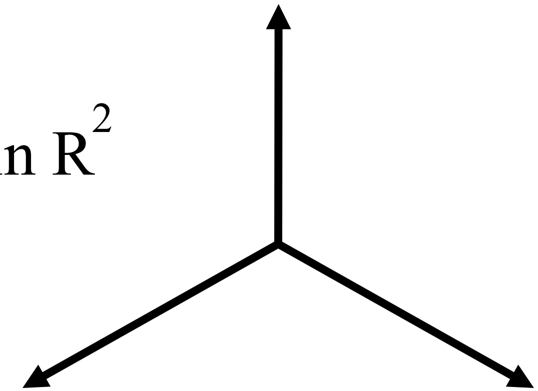
Drifting games (in 2d)



The allowable steps

B = the set of all allowable step
Normal B = minimal set that spans the space. (~basis)
Regular B = a symmetric regular set. (~orthonormal basis)

Regular step set in R^2



The min/max solution

[Schapire99]

- A potential defined by a min/max recursion

$$\phi_T(\mathbf{s}) = L(\mathbf{s})$$

$$\phi_{t-1}(\mathbf{s}) = \min_{\mathbf{w}\in\mathbb{R}^d} \sup_{\mathbf{z}\in B} \left(\phi_t(\mathbf{s}+\mathbf{z}) + \mathbf{w}\cdot\mathbf{z} - \delta\,\|\mathbf{w}\|\right)$$

$\phi_0(0) =$ the value of the game

- Shepherd´s strategy

$$\mathbf{w}_i^t = \arg\min_{\mathbf{w}} \sup_{\mathbf{z}\in B} \left(\phi_t(\mathbf{s}+\mathbf{z}) + \mathbf{w}\cdot\mathbf{z} - \delta\,\|\mathbf{w}\|\right)$$

The solution simplifies when $\delta \to 0$

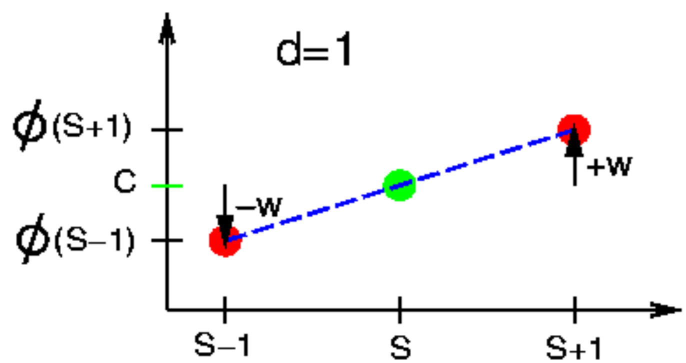
If B is normal, and δ is sufficiently small then $\exists \mathbf{w}^*$ such that

$$\phi_{t-1}(\mathbf{s}) = \phi_t(\mathbf{s}+\mathbf{z}) + \mathbf{w}^* \cdot \mathbf{z} - \delta \|\mathbf{w}^*\|$$

for all $\mathbf{z} \in B$ (and all $t = 1, 2 \ldots$, $\mathbf{s} \in \mathbb{R}^d$)

Implies that: $\mathbf{w}^*$ is the "local slope" at $\phi_t(\mathbf{s})$, i.e.

$$\phi_t(\mathbf{s}+\mathbf{z}_i) = C + \mathbf{w}^* \mathbf{z}_i \; ; \quad C \doteq \frac{\Sigma_{j=0}^{d} \phi_t(\mathbf{s}+\mathbf{z}_j)}{d+1}$$



and that

$$\phi_{t-1}(\mathbf{s}) = C - \delta \|\mathbf{w}^*\|$$

Increasing the number of steps

- Consider T steps in a unit time
- Drift δ should scale like $1/T$
- Step size $O(1/T)$ gives game to shepherd
- Step size $O\left(1/\sqrt{T}\right)$ keeps game balanced

The continuous time limit

For $t = 0, 1, 2, \ldots, T$ instantanous loss is $l_i^t \in \{-1, +1\}$

Instead of $t = 0, 1, 2, \ldots, T$ let $t = 0, \frac{1}{T}, \frac{2}{T}, \ldots, 1$

Let $T \to \infty$

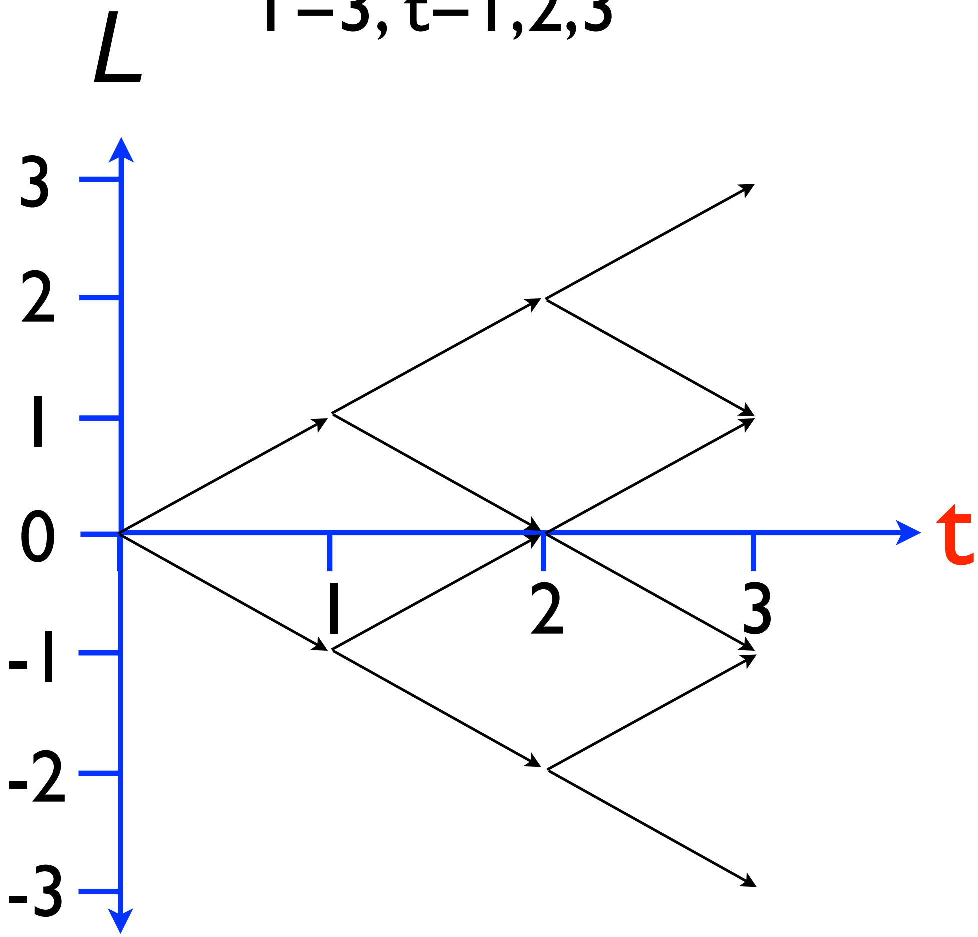
How should we set the instantanous loss?

Natural choice: $l_i^t = \pm \frac{1}{T}$?

For optimal adversary, cumulative loss of each expert defines a random walk. What is random walk in continuous time?

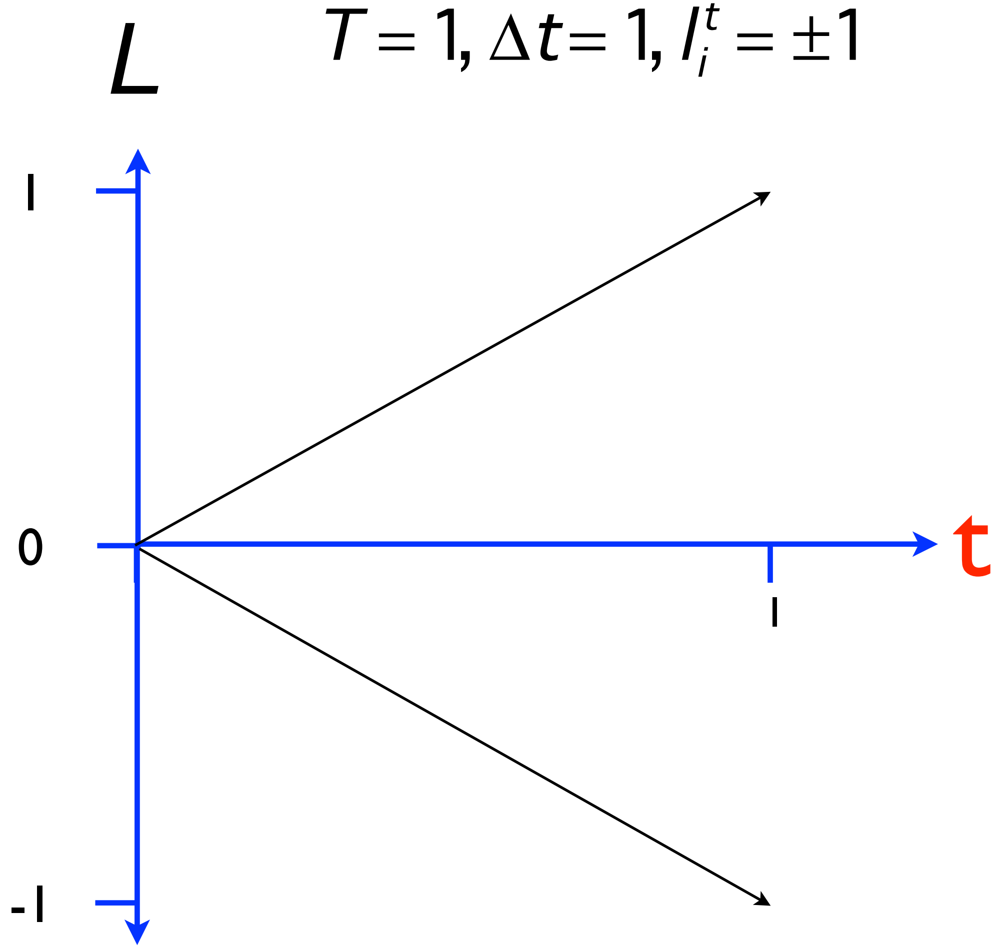
The game lattice

T=3, t=1,2,3



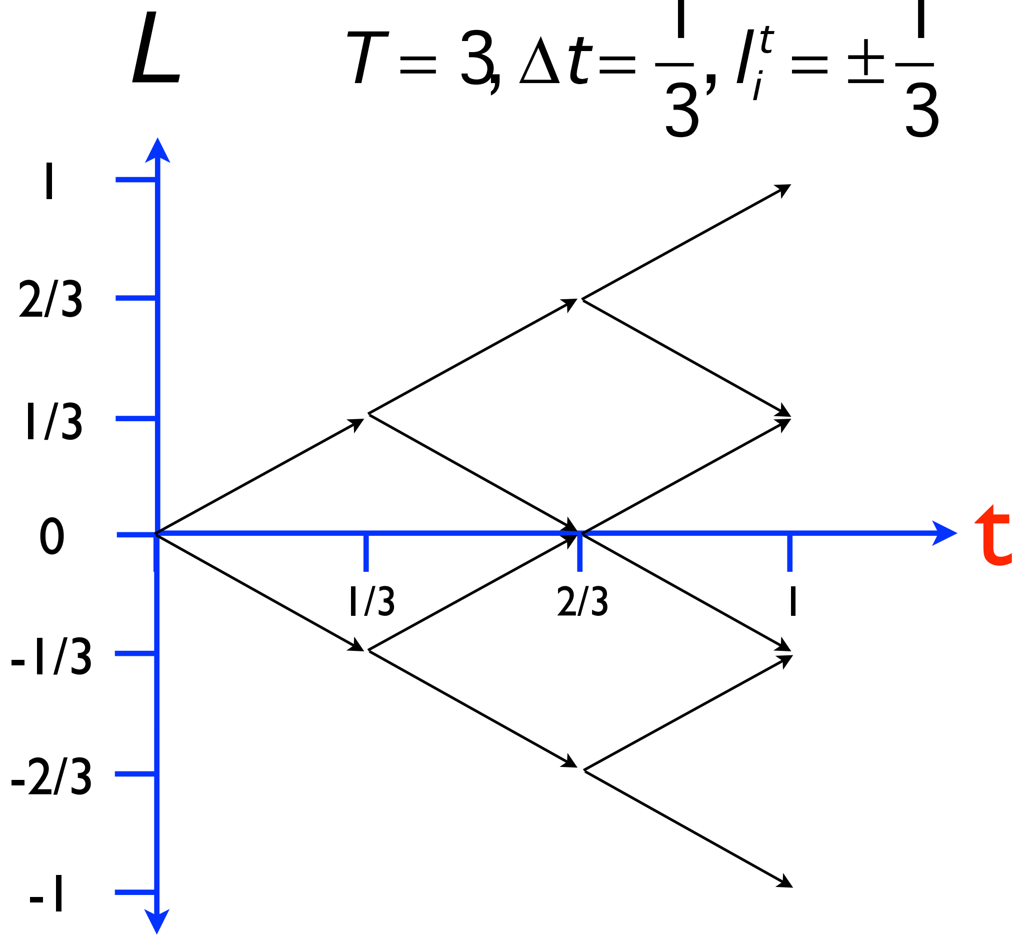
Using step $l_i^t = \pm\frac{1}{T}$

$T = 1, \Delta t = 1, l_i^t = \pm 1$



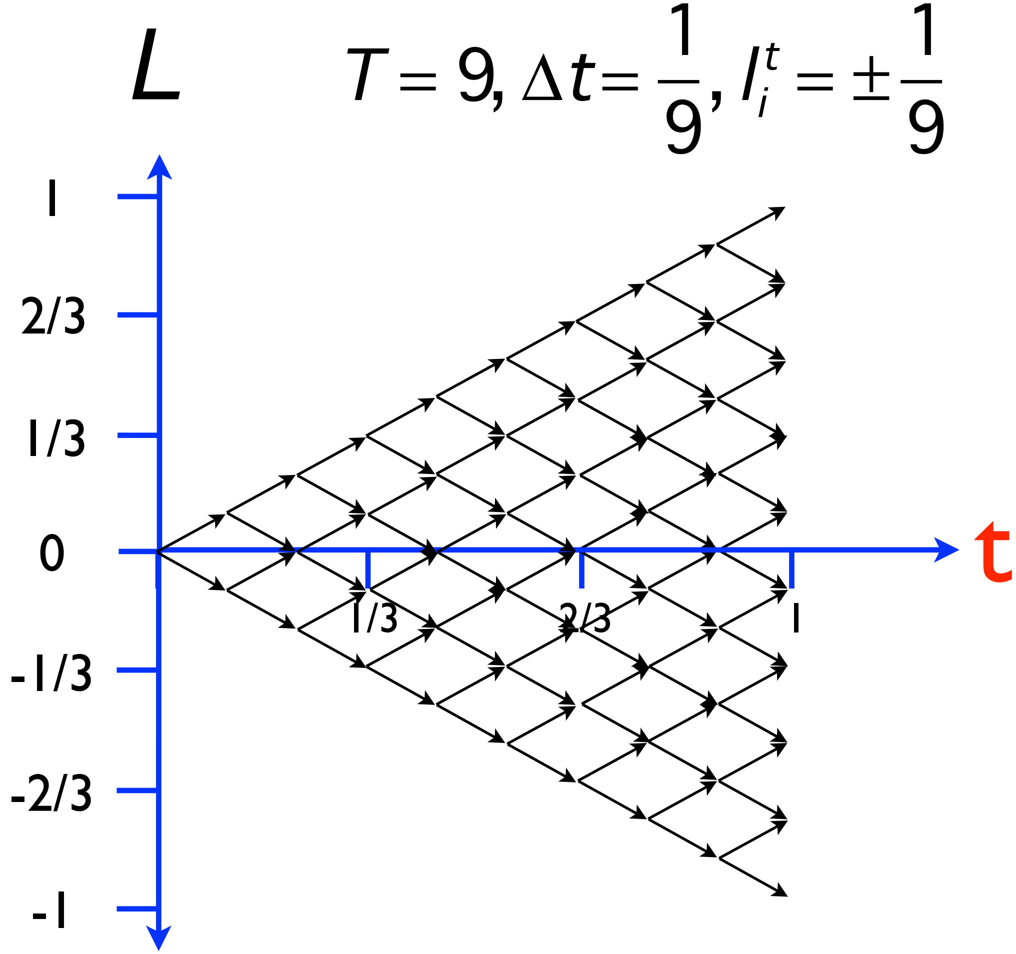
Using step $l_i^t = \pm \frac{1}{T}$

$$T = 3, \Delta t = \frac{1}{3}, l_i^t = \pm \frac{1}{3}$$



Using step $l_i^t = \pm\frac{1}{T}$

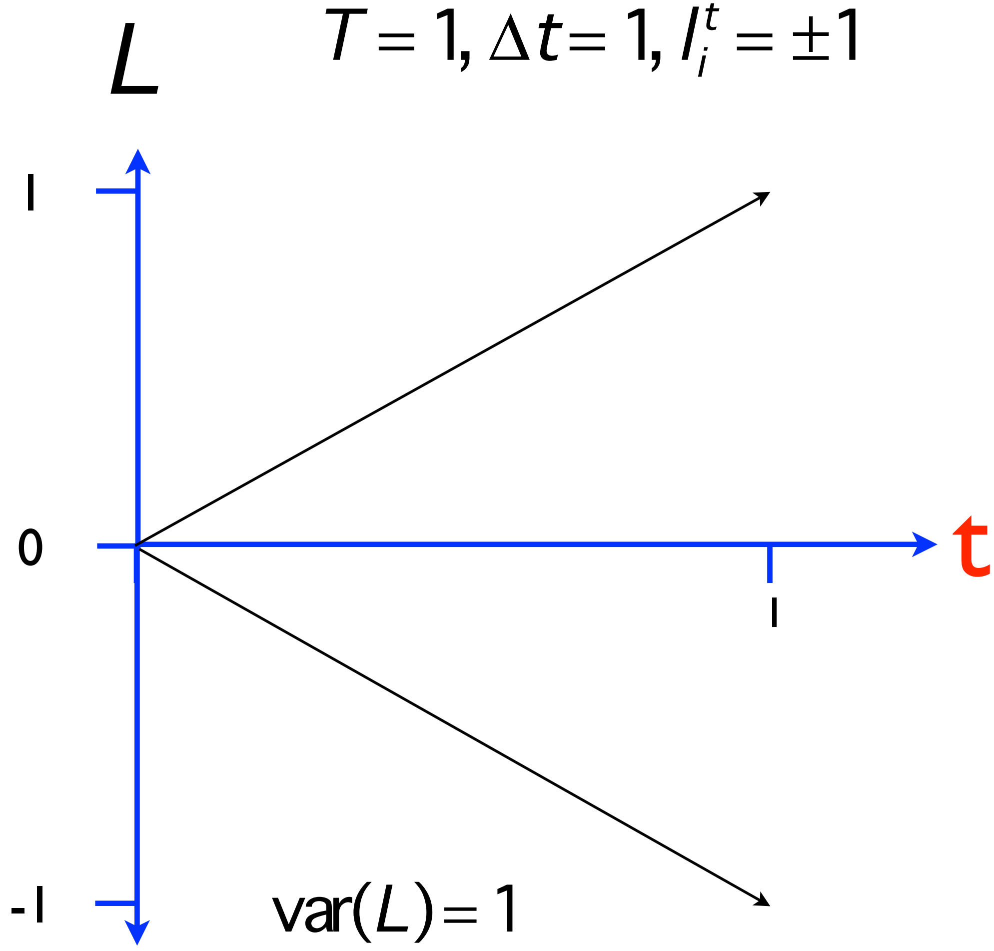
$$T = 9, \Delta t = \frac{1}{9}, l_i^t = \pm\frac{1}{9}$$



Looks fine but $\mathrm{var}(L) = T\frac{1}{T^2} = \frac{1}{T} \underset{T\to\infty}{\longrightarrow} 0$

Using step $l_i^t = \pm \frac{1}{\sqrt{T}}$

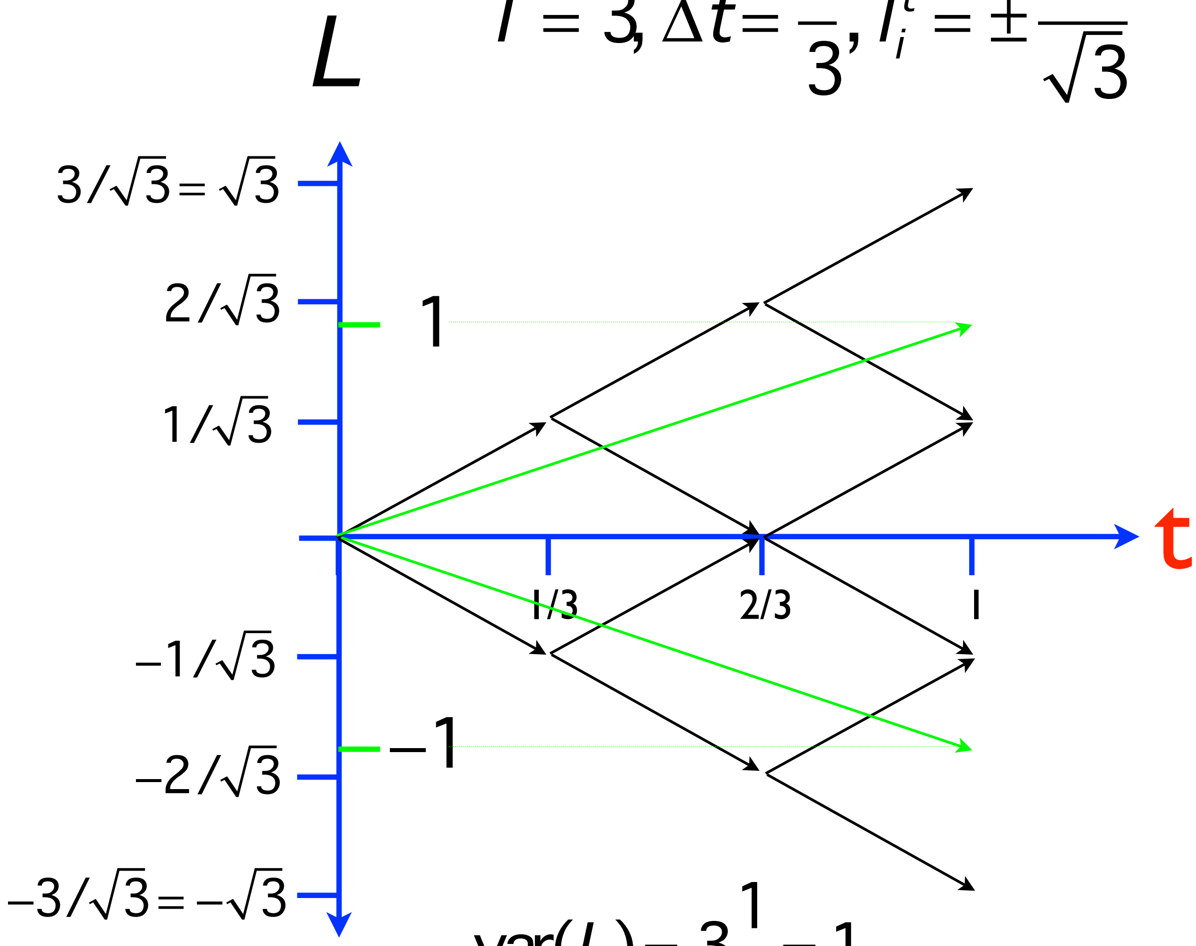
$T = 1, \Delta t = 1, l_i^t = \pm 1$



$\mathrm{var}(L) = 1$

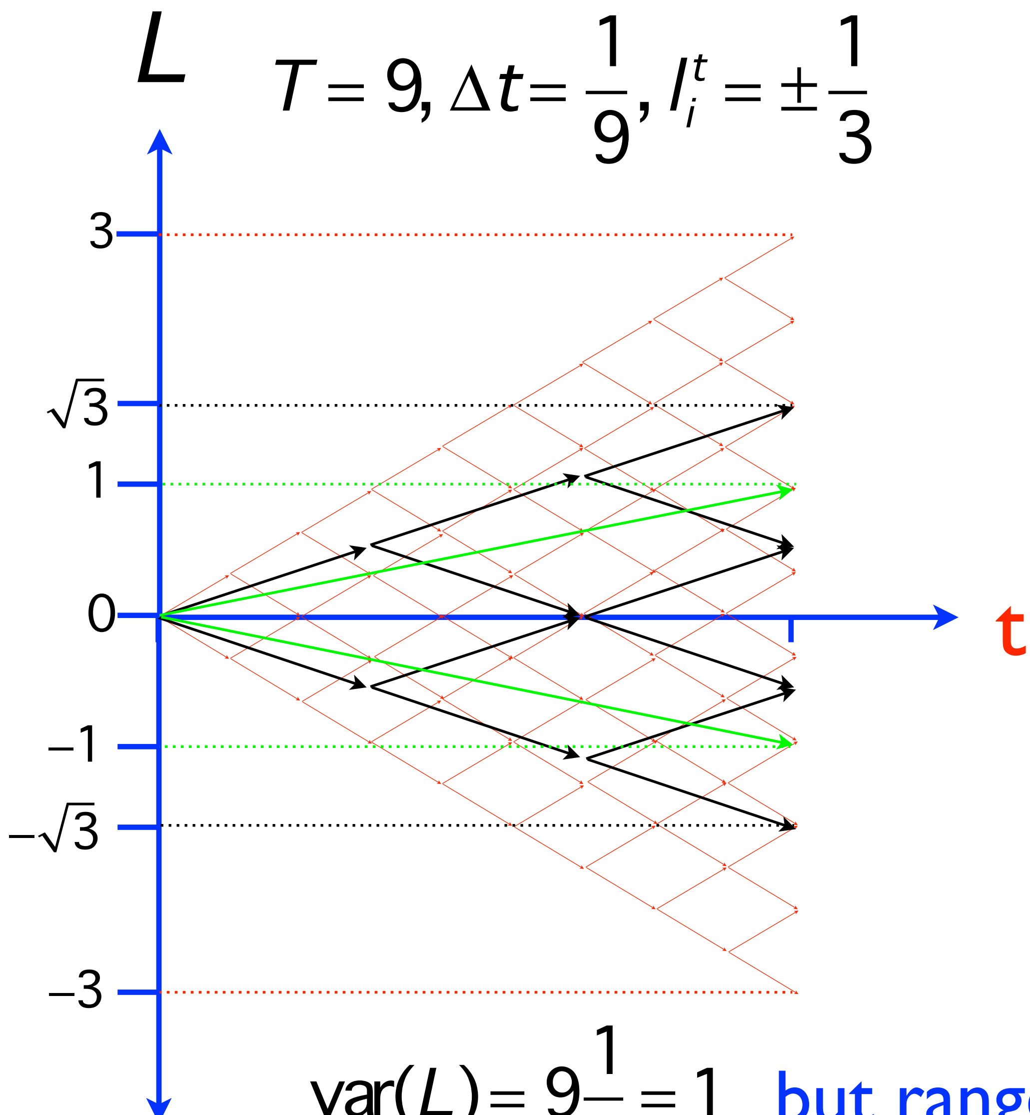
Using step $l_i^t = \pm\frac{1}{\sqrt{T}}$

$$T = 3, \Delta t = \frac{1}{3}, l_i^t = \pm\frac{1}{\sqrt{3}}$$



$$\mathrm{var}(L) = 3\frac{1}{3} = 1$$

Using step $l_i^t = \pm \frac{1}{\sqrt{T}}$



$\mathrm{var}(L) = 9\frac{1}{9} = 1$ but range(L) $\rightarrow \infty$

The solution when $T \to \infty$

The local slope becomes the gradient

$$\mathbf{w}^* = \nabla \phi_\tau(\mathbf{s})$$

The recursion becomes a PDE

$$\begin{aligned} \frac{\partial \phi_\tau(\mathbf{s})}{\partial \tau} &= -\frac{1}{2} \sum_{k=1}^{d} \frac{\partial^2 \phi_\tau(\mathbf{s})}{\partial^2 s_k} + \delta \|\mathbf{w}^*\| \\ &= -\frac{1}{2} \Delta \phi_\tau(\mathbf{s}) + \delta \|\nabla \phi_\tau(\mathbf{s})\| \end{aligned}$$

Same PDE describes time development of Brownian motion with drift

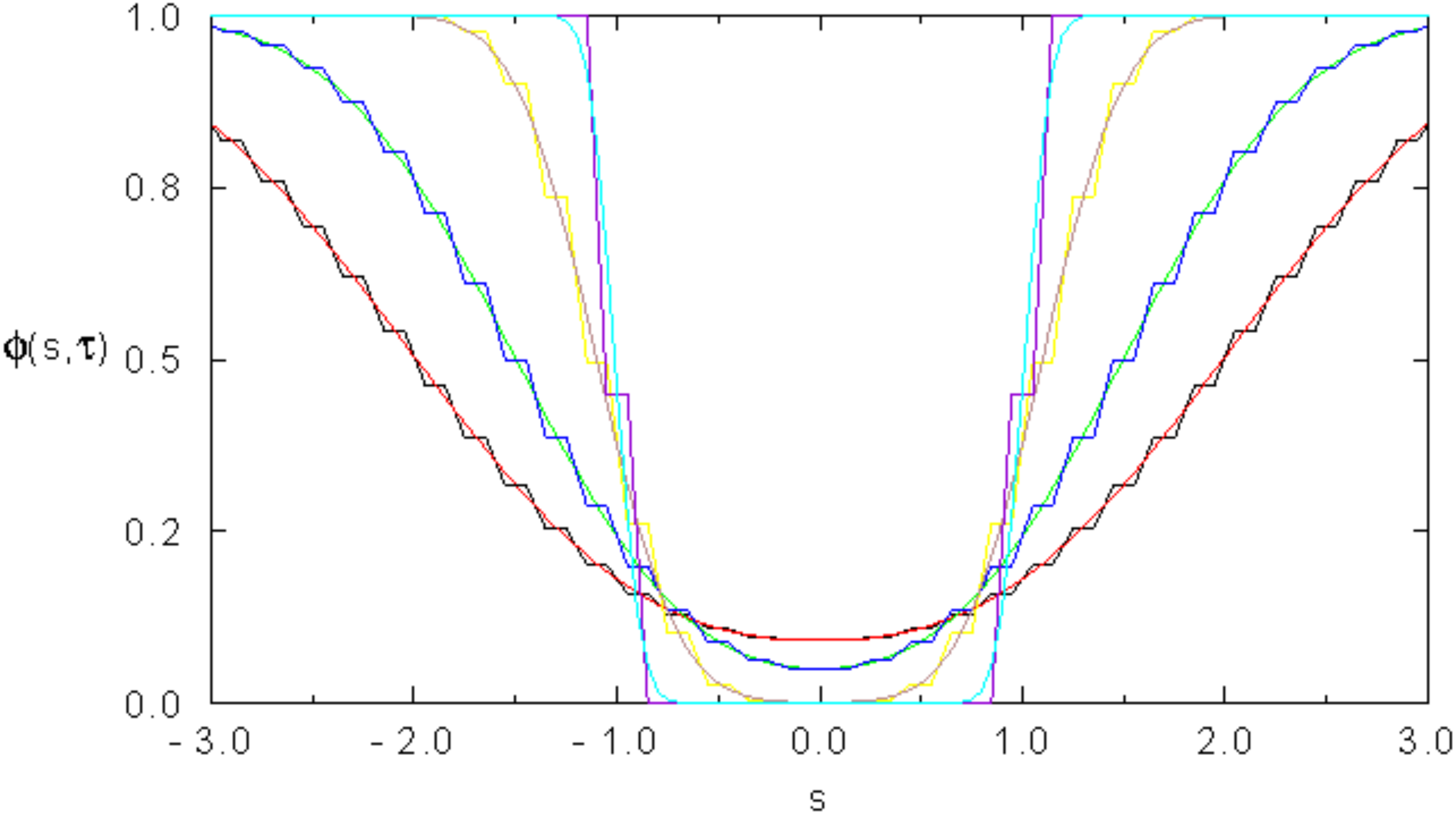
Special Cases

- Target func. = PDE boundary condition at time=1.
- Target func. = exponential
  - potential and weight are exponential at all times
  - Adaboost.
  - Exponential weights algs for online learning.
- Target func. = step function
  - Potential is the error function, weight is the normal distribution.
  - Potential and weight change with time.
  - Brownboost.

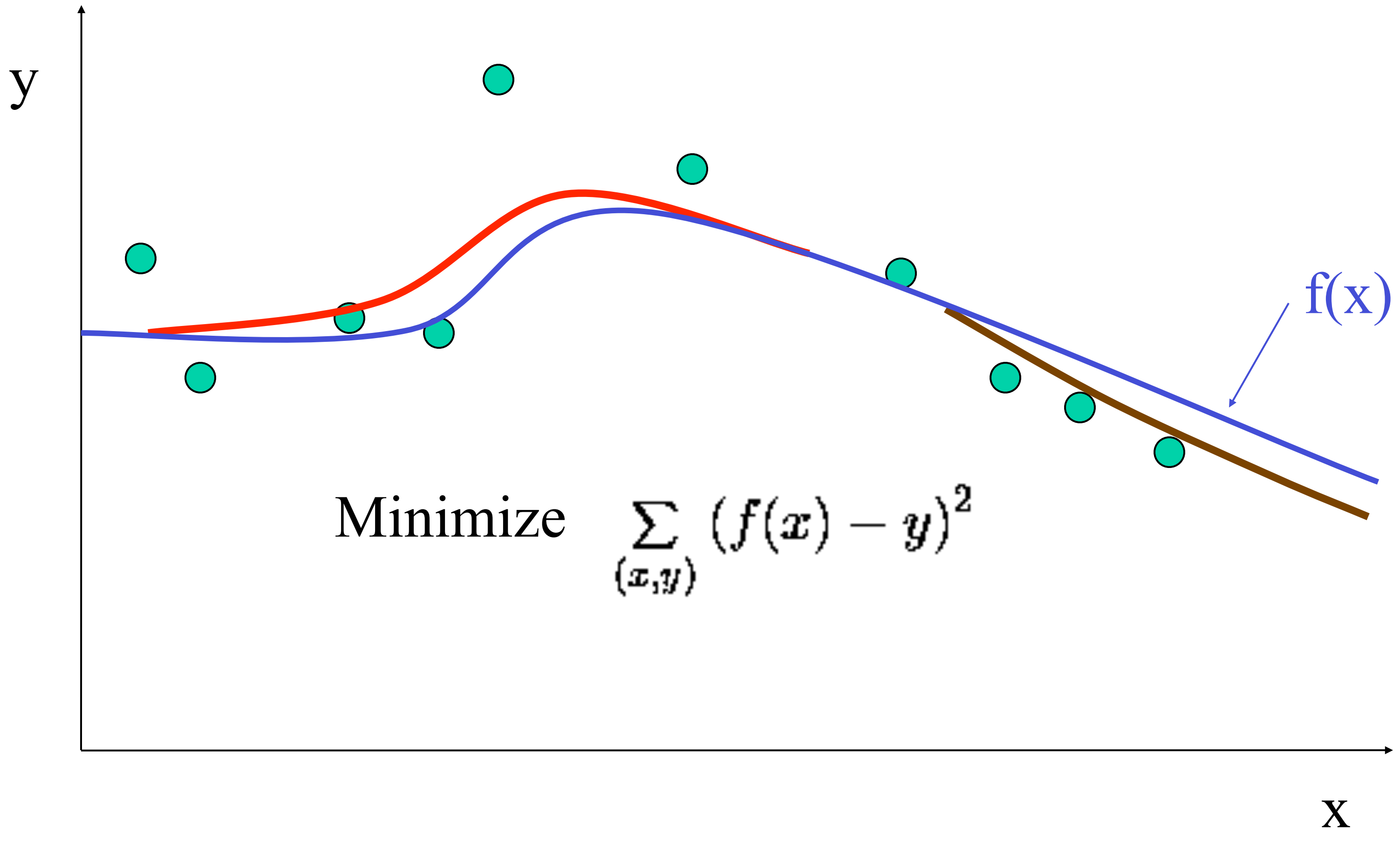
Potential applications

- Generalized boosting
  - Classification for >2 labels
  - Regression
  - Density estimation
  - Variational function fitting
- Generalized online learning
  - Continuous predictions (instead of binary)

solution for $L(s) = I_{|s|>1}$



Boosting for variational optimization



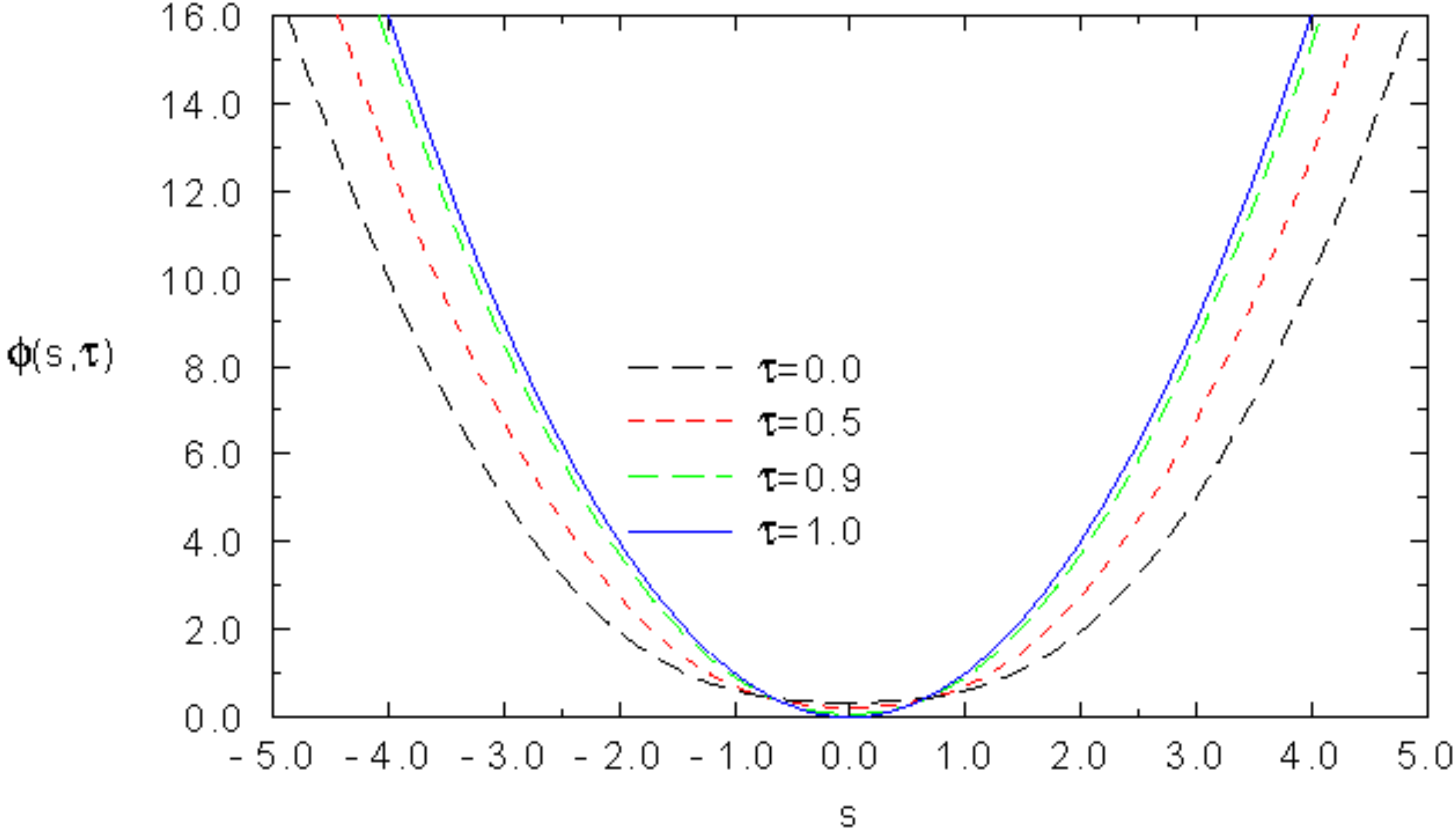


Figure 2: The potential $\phi(s, t)$ for the square loss $L(y) = y^2$.

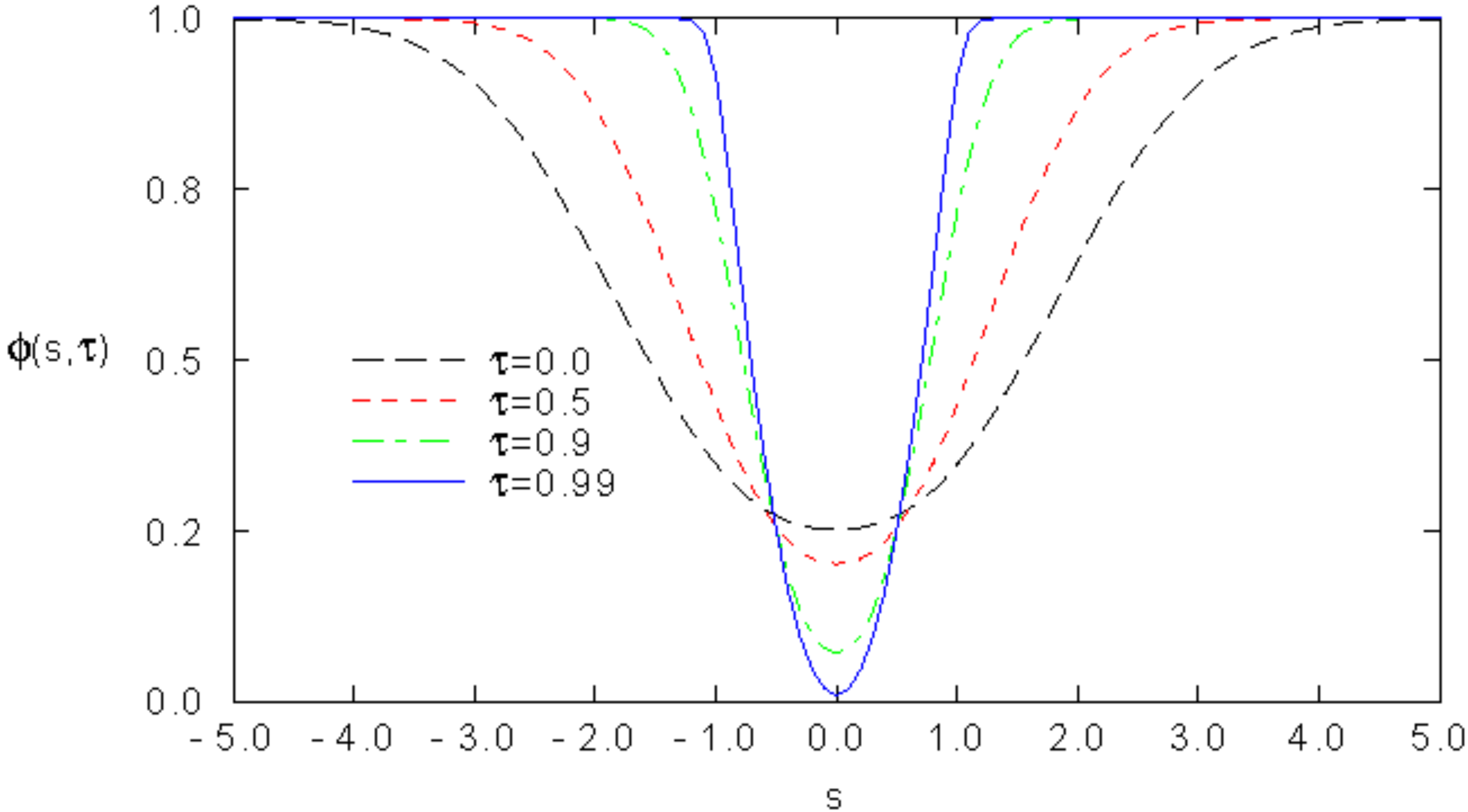


Figure 3: The potential $\phi(\mathbf{s}, \boldsymbol{t})$ for the loss $L(y) = \mathbf{min}(y^2, 1)$.

Plan of talk

- How to use expert advice,
  A quick tour of some old results.
- Decision Theoretic Online Learning
  - Hedge
  - Binomial Weights
  - Normal Hedge
  - Open problem

The Hedge Algorithm

[Freund & Schapire 1997]

based on [Littlestone and Warmuth 1989],[Cesa-Bianchi et al 1997]

Initial weights: $w^1 = \left\langle \frac{1}{N}, \ldots, \frac{1}{N} \right\rangle$

Weights update rule: $w_i^{t+1} = w_i^t e^{-\eta l_i^t} = e^{-\eta L_i^{t+1}}$

Learning rate

Alternatively: $w_i^{t+1} = \frac{1}{N} e^{-\eta L_i^t}$

Posterior probability
(un-normalized)

Prior probability

Potential-based bound

Potential: $W^{t+1} = \sum_{i=1}^{N} w_i^{t+1} = \sum_{i=1}^{N} e^{-\eta L_i^t}$

Large loss of algorithm => small potential $\qquad L_A^t \le \frac{-\log W^{t+1}}{1-e^{-\eta}}$

Good expert=> large potential $\qquad W^{t+1} \ge w_i^{t+1} = e^{-\eta L_i^t}$

Combining, we get: $\qquad \forall i, L_A^T \le \frac{\eta L_i^T + \ln N}{1-e^{-\eta}}$

Tuning the learning rate

$$\forall i, L_A^T \le \frac{\eta L_i^T + \ln N}{1 - e^{-\eta}}$$

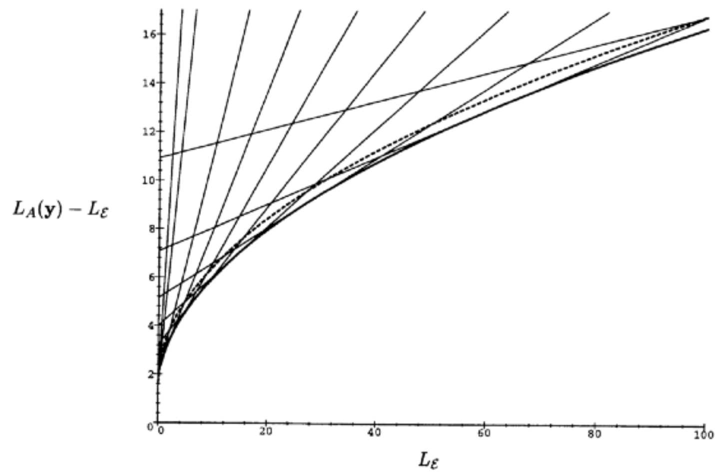
If we set $\eta = \sqrt{\frac{2\ln N}{T}}$

Then we guarantee $\mathrm{L}_A^T \le \min_i L_i^T + \sqrt{2T \ln N} + \ln N$

Equivalently $\forall i, R_i^T \le \sqrt{2T \ln N} + \ln N; \quad \lim_{T \to \infty} \frac{\sqrt{2T \ln N} + \ln N}{T} = 0$

Tuning the learning rate

$$\forall i, L_A^T \le \frac{\eta L_i^T + \ln N}{1 - e^{-\eta}}$$



If we set $\eta = \sqrt{\frac{2\ln N}{T}}$

Then we guarantee $\mathrm{L}_A^T \le \min_i L_i^T + \sqrt{2T\ln N} + \ln N$

Equivalently $\forall i, R_i^T \le \sqrt{2T\ln N} + \ln N; \quad \lim_{T\to\infty} \frac{\sqrt{2T\ln N} + \ln N}{T} = 0$

Is it possible to do better?

Plan of talk

- How to use expert advice,
  A quick tour of some old results.
- Decision Theoretic Online Learning
  - Hedge
  - Binomial Weights
  - Normal Hedge
  - Open problem

Design of NormalHedge

- BW: potential function depends on loss and number of remaining mistakes
- Normal-Hedge: Potential function based on regret and variance of the positive cumulative regrets

The NormalHedge potential

$$\text{Potential: } \psi(r,c) = \begin{cases} \exp\left(\dfrac{R^2}{2c}\right) & \text{if R} \geq 0 \\ 1 & \text{if R} \leq 0 \end{cases}$$

$$\text{Weight: } w(R,c) = \frac{\partial}{\partial R}\psi(R,c) = \begin{cases} \dfrac{R}{c}\exp\left(\dfrac{R^2}{2c}\right) & \text{if R} \geq 0 \\ 0 & \text{if R} \leq 0 \end{cases}$$

Intuition: If we play against the random walk player, then

the probability that the cumulative regret is R is approximately $e^{-\frac{R^2}{2t}}$,
to have any hope of keeping the potential constant the potential function cannot increase faster than 1/probability.

NormalHedge algorithm

for t=0,1,2,...

if $\forall i, R_i^t \le 0$ then $w_i^t = 1/N$

else

set $c(t)$ so that $\frac{1}{N}\sum_{i=1}^{N}\psi\left(R_i^t, c(t)\right) = e$

$w_i^t = w\left(R_i^t, c(t)\right)$

Incur instantanous losses: $\left\langle l_1^t, l_2^t, \ldots, l_N^t \right\rangle$

Algorithm loss: $l_A^t = \frac{\sum_{i=1}^{N} w_i^t l_i^t}{\sum_{i=1}^{N} w_i^t}$

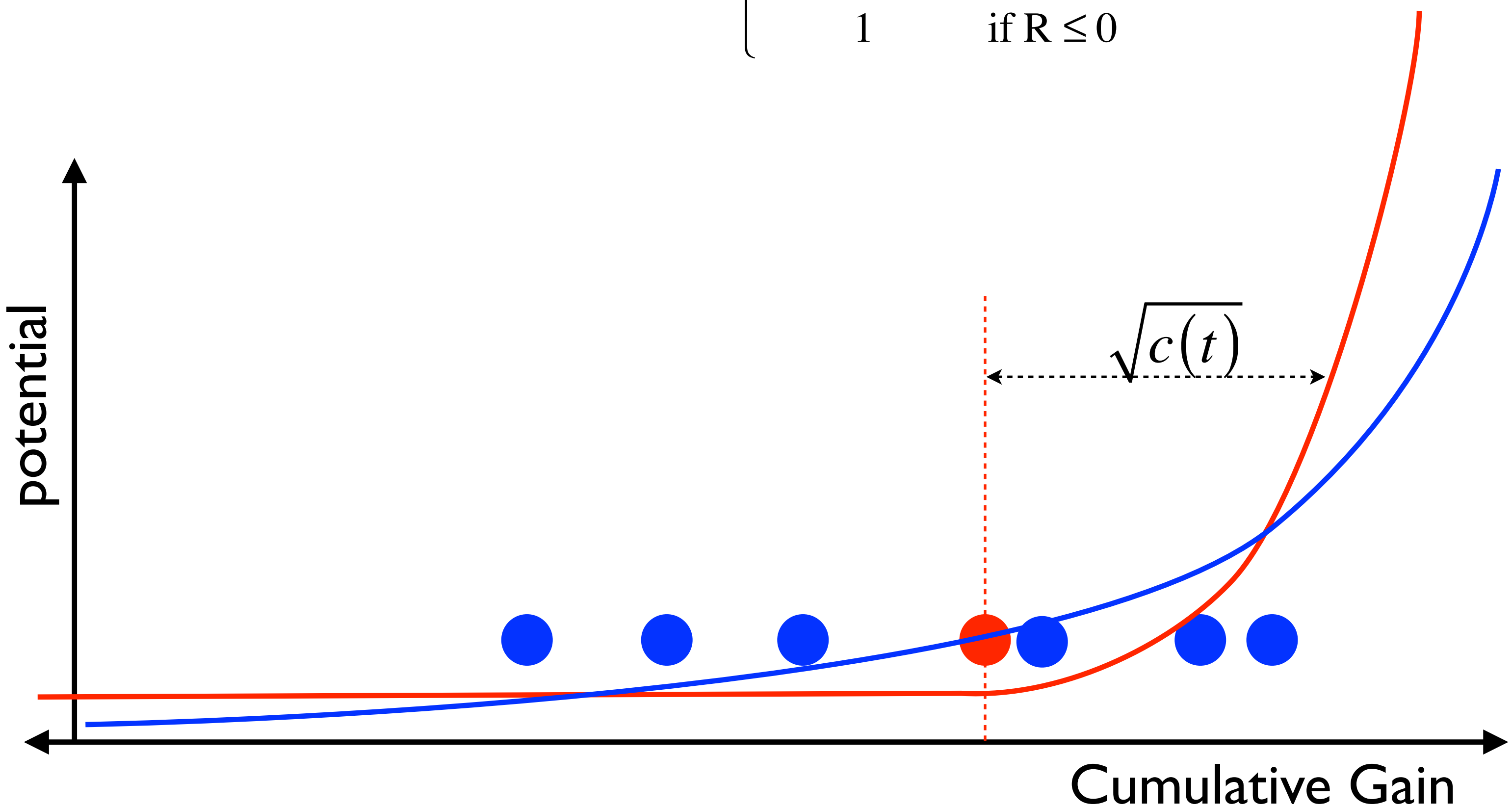
Update regrets: $R_i^{t+1} = R_i^t + l_A^t - l_i^t$

Illustrative Example

- Expert
- Algorithm

$\exp(\eta G)$

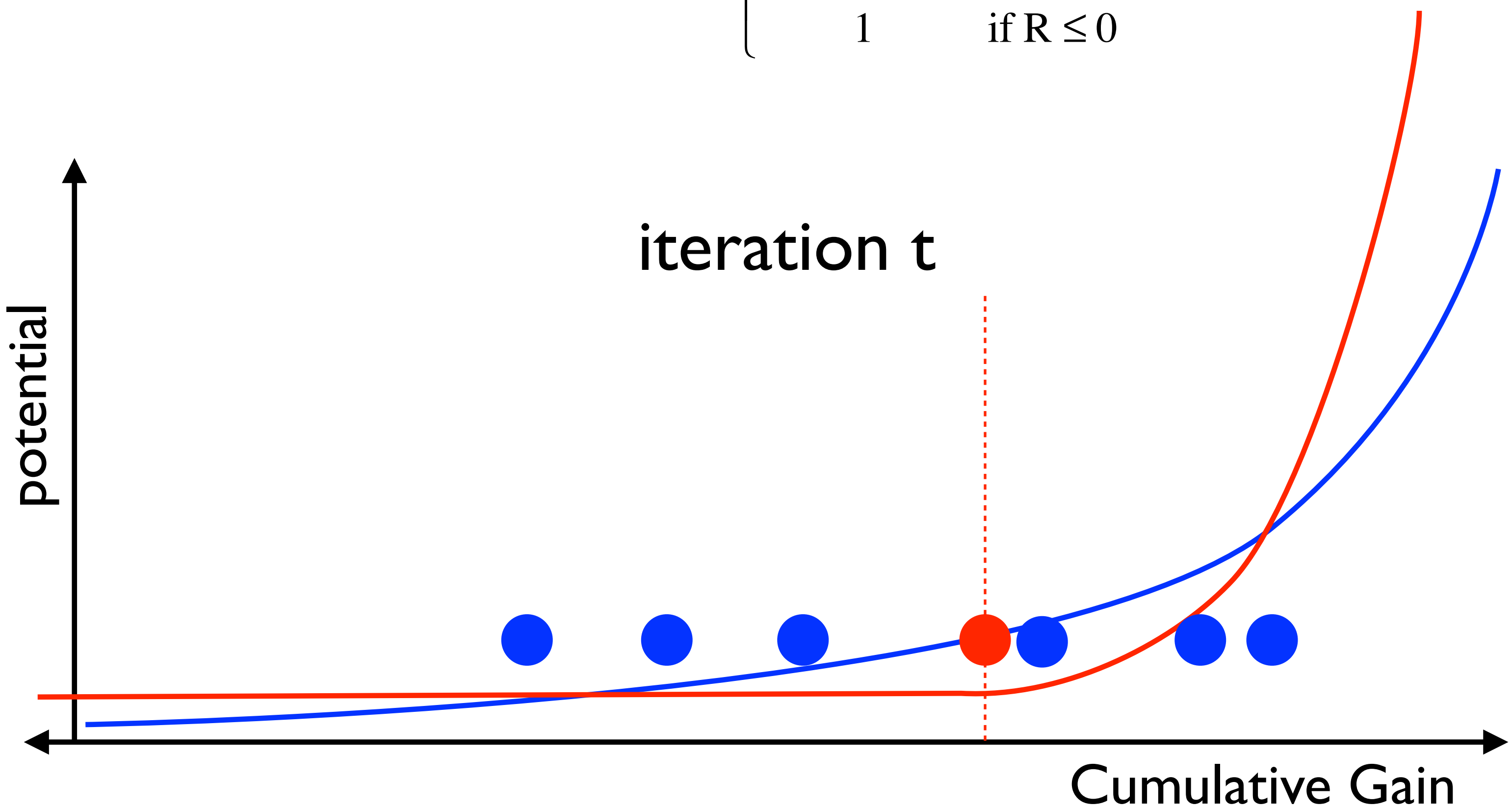
$$\begin{cases} \exp\left(\frac{R^2}{2c}\right) & \text{if } R \geq 0 \\ 1 & \text{if } R \leq 0 \end{cases}$$



Illustrative Example

Expert — $\exp(\eta G)$

Algorithm — $\begin{cases} \exp\left(\dfrac{R^2}{2c}\right) & \text{if R} \geq 0 \\ 1 & \text{if R} \leq 0 \end{cases}$



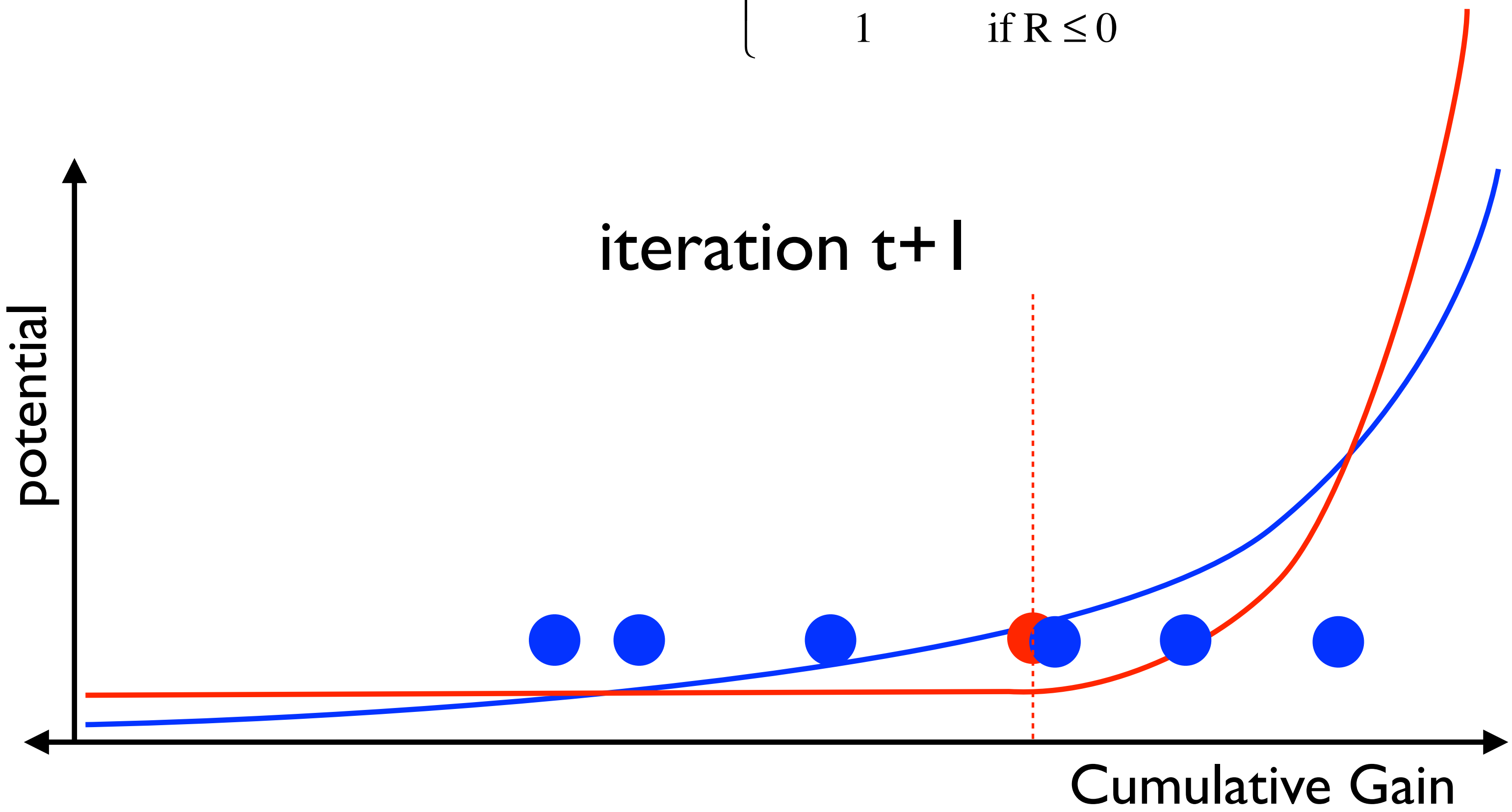
Illustrative Example

● Expert

● Algorithm

$\exp(\eta G)$

$$\begin{cases} \exp\left(\frac{R^2}{2c}\right) & \text{if } R \geq 0 \\ 1 & \text{if } R \leq 0 \end{cases}$$



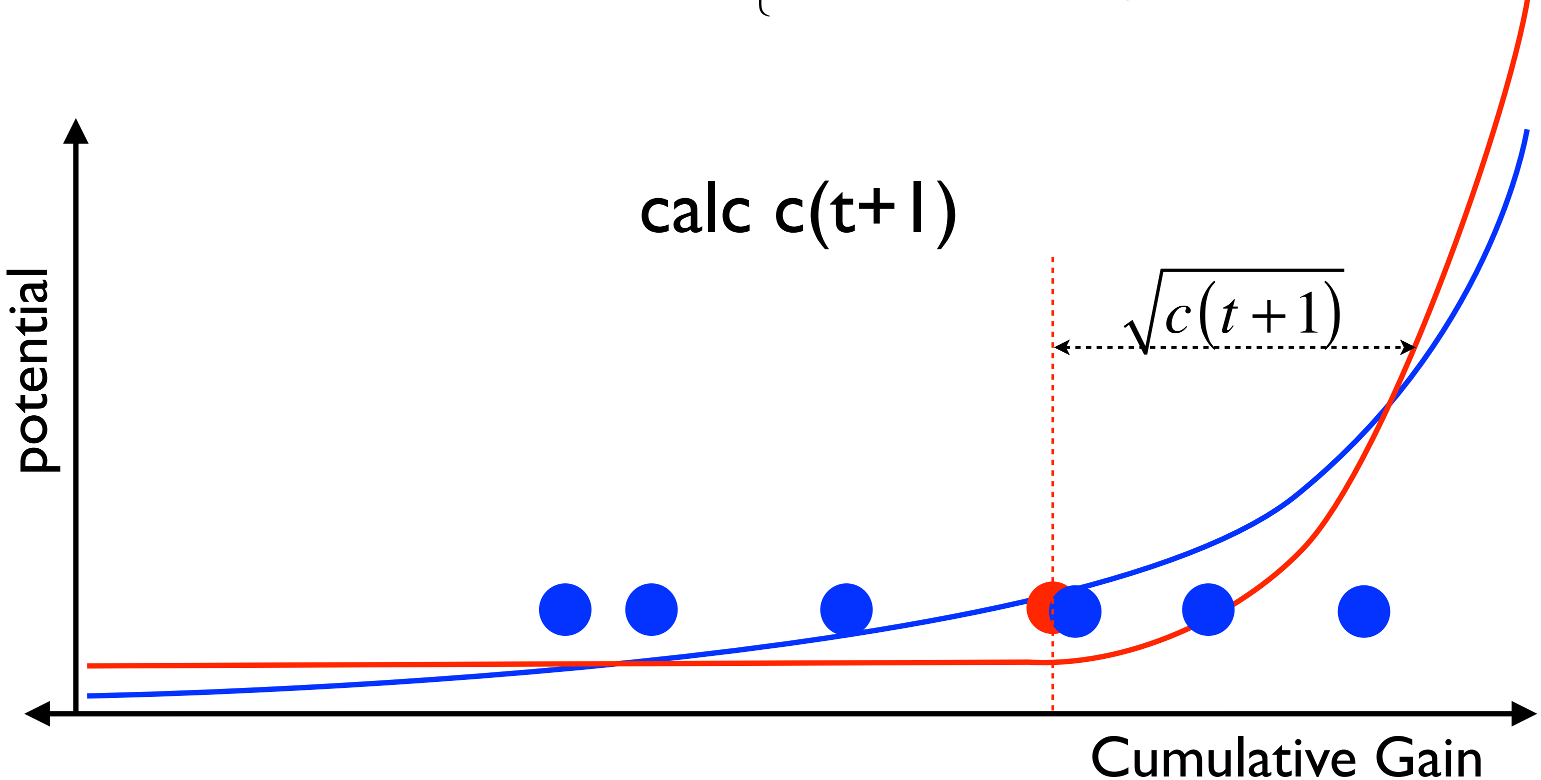
Illustrative Example

Expert

Algorithm

$$\exp(\eta G)$$

$$\begin{cases} \exp\left(\dfrac{R^2}{2c}\right) & \text{if } \mathrm{R} \geq 0 \\ 1 & \text{if } \mathrm{R} \leq 0 \end{cases}$$



Normal-Hedge Performance bound

[Chaudhuri, Freund & Hsu 2009]

Main Lemma: $c(t) \leq t + C_N$

The regret of NormalHedge is upper bounded by

$$\sqrt{3t\left(1+\ln N\right)+o(t)}$$

The regret to the top ϵ-percentile is upper bounded by

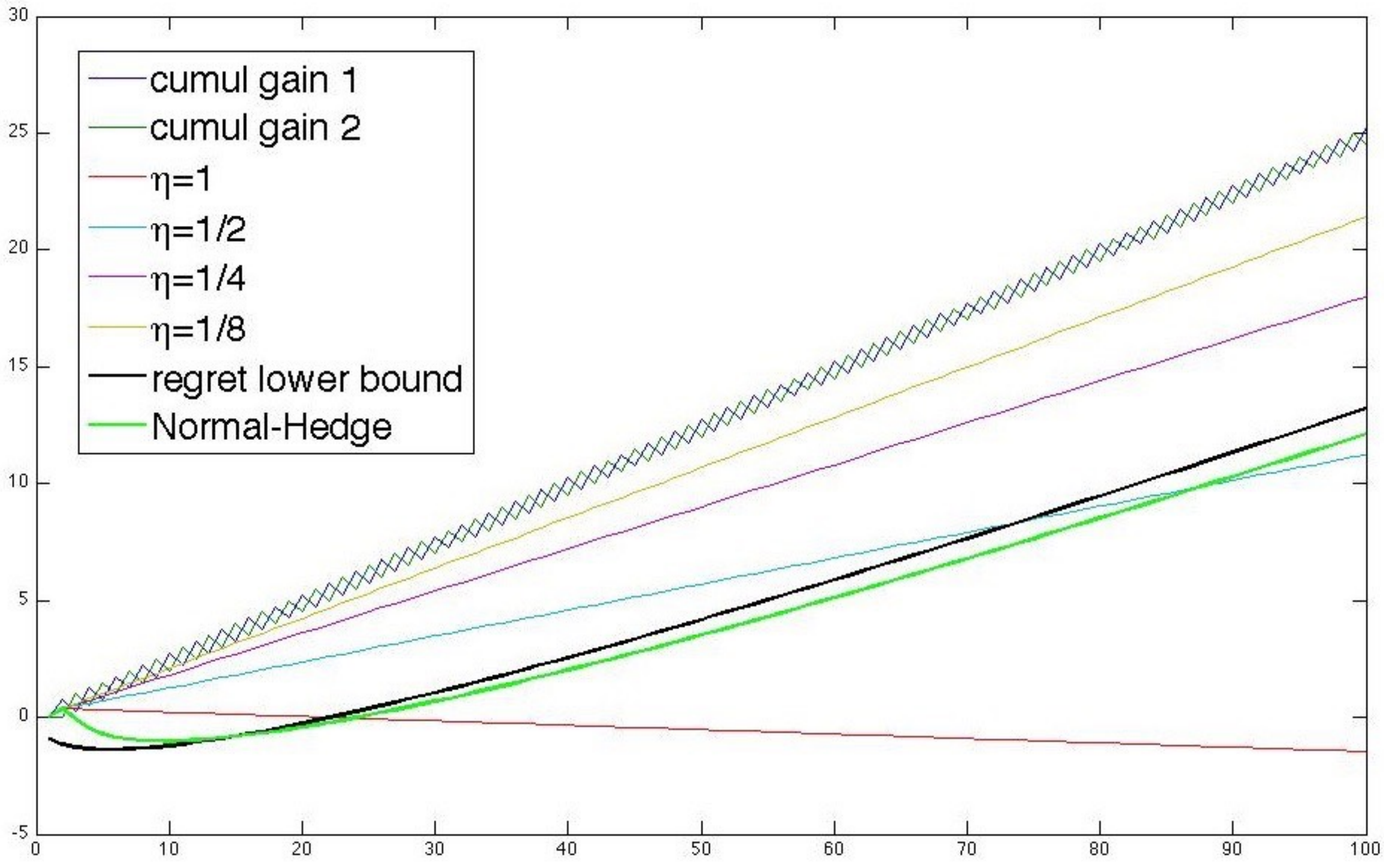
$$O\left(\sqrt{t\ln(1/\epsilon)}+\ln^2 N\right)$$

We failed to get rid of the dependence on N

BW depends only on $1/\epsilon$ not on N

Performance on flip-flop

- Worst case for <u>follow the leader</u>



NormalHedge.DT / Luo and Schapire, 2014

$$\text{Potential: } \psi(r,t) = \begin{cases} \exp\left(\dfrac{r^2}{3t}\right) & \text{if } r \geq 0 \\ 1 & \text{if } r \leq 0 \end{cases}$$

$$\text{Weight: } w(r,t) = \psi(r+1,t+1) - \psi(r+1,t-1)$$

$$\text{Recall NormalHedge weight: } w(r,c) = \frac{\partial}{\partial r}\psi(r,c)$$

A very clean application of drifting games analysis

NormalHedge.DT / analysis

Instead of keeping potential constant,

allow potential to increase like $c\log(t)$

The regret of AdaNormalHedge is bounded by

$$\sqrt{3t\ln\left(\frac{1}{2\epsilon}\left(e^{4/3}-1\right)\left(\ln t+1\right)+1\right)}=O\left(\sqrt{t\ln(1/\epsilon)+t\ln\ln t}\right)$$

for all t and for all $\epsilon>0$

Algorithm does not depend on ϵ !!!

Plan of talk

- How to use expert advice,
  A quick tour of some old results.
- Decision Theoretic Online Learning
  - Hedge
  - Binomial Weights
  - Normal Hedge
  - Open problem

Open Problem

- Luo and Schapire got rid of the dependence on N
- What about the dependence on T?
- Suppose that, although loss is bounded in [0,1], it turns out that the loss is always in [0,1/2].
- If alg. has information in advance, bound is halved
- Equivalently: $T \rightarrow \frac{T}{4}$
- Can we achieve this performance without a -priori knowledge?
- Hypothesis: The increase c(t+1)-c(t) for normal hedge can upper bounded by a function of the form

$$\int_\theta \alpha_\theta \left(r_\theta^t\right)^2 d\mu(\theta) \quad \text{(current bound is 1)}$$

- Without degrading current performance.

Recent Progress

Schapire and Liu 2015:

AdaNormalHedge, an adaptive version of NormalHedge.DT

The regret of AdaNormalHedge relative to the ϵ best experts at time T

$$\forall T, \epsilon \quad R(\epsilon, T) \leq O\left(\sqrt{T \log\left(\frac{\log T}{\epsilon}\right)}\right)$$

Open Problem

- Luo and Schapire got rid of the dependence on N
- What about the dependence on T?
- Suppose that, although loss is bounded in [0,1], it turns out that the loss is always in [0,1/2].
- If alg. has information in advance, bound is halved
- Equivalently: $T \rightarrow \frac{T}{4}$
- Can we achieve this performance without a -priori knowledge?
- Hypothesis: The increase c(t+1)-c(t) for normal hedge can upper bounded by a function of the form

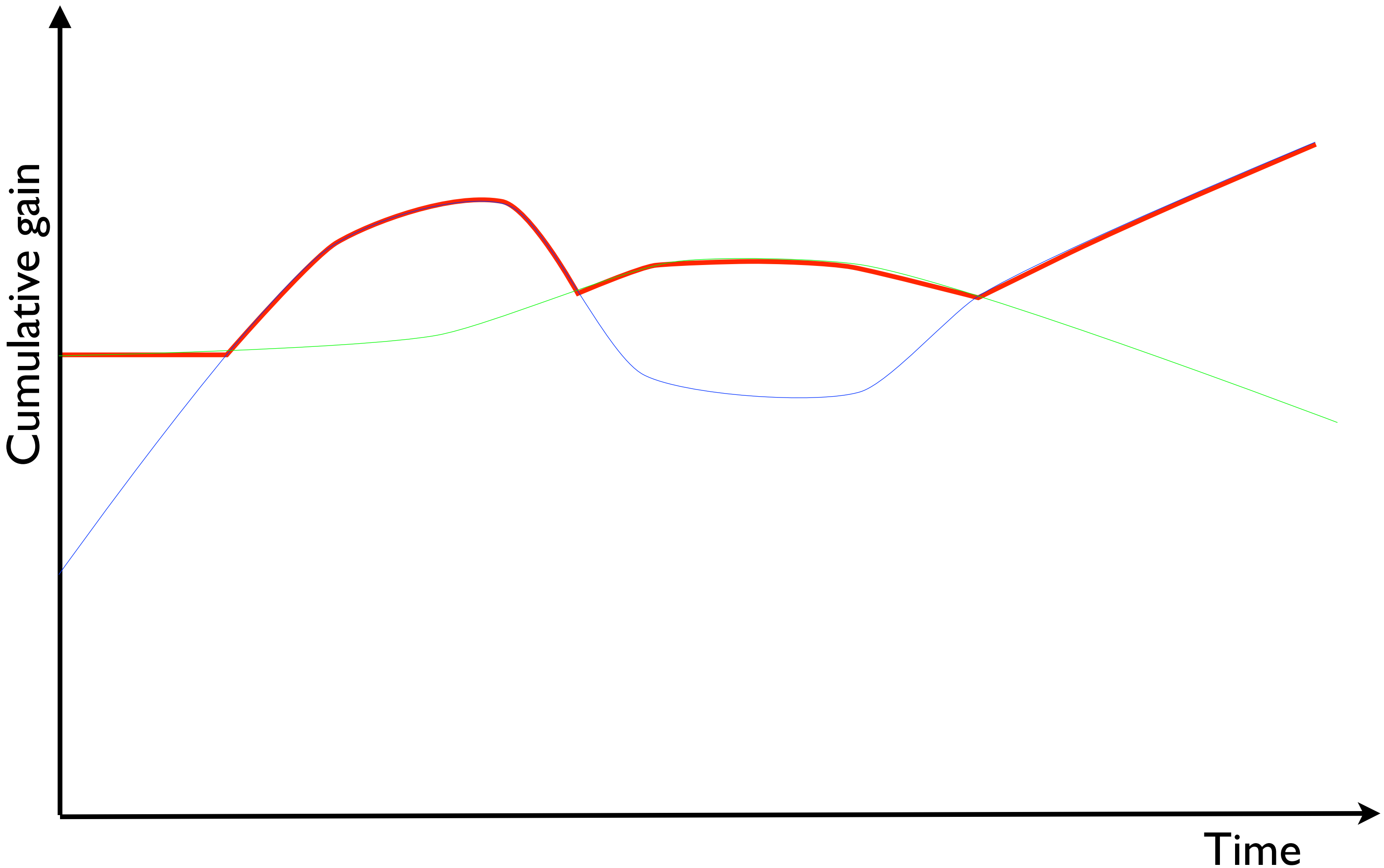
$$\int_{\theta} \alpha_{\theta} \left(r_{\theta}^{t}\right)^{2} d\mu(\theta) \quad \text{(current bound is 1)}$$

- Without degrading current performance.

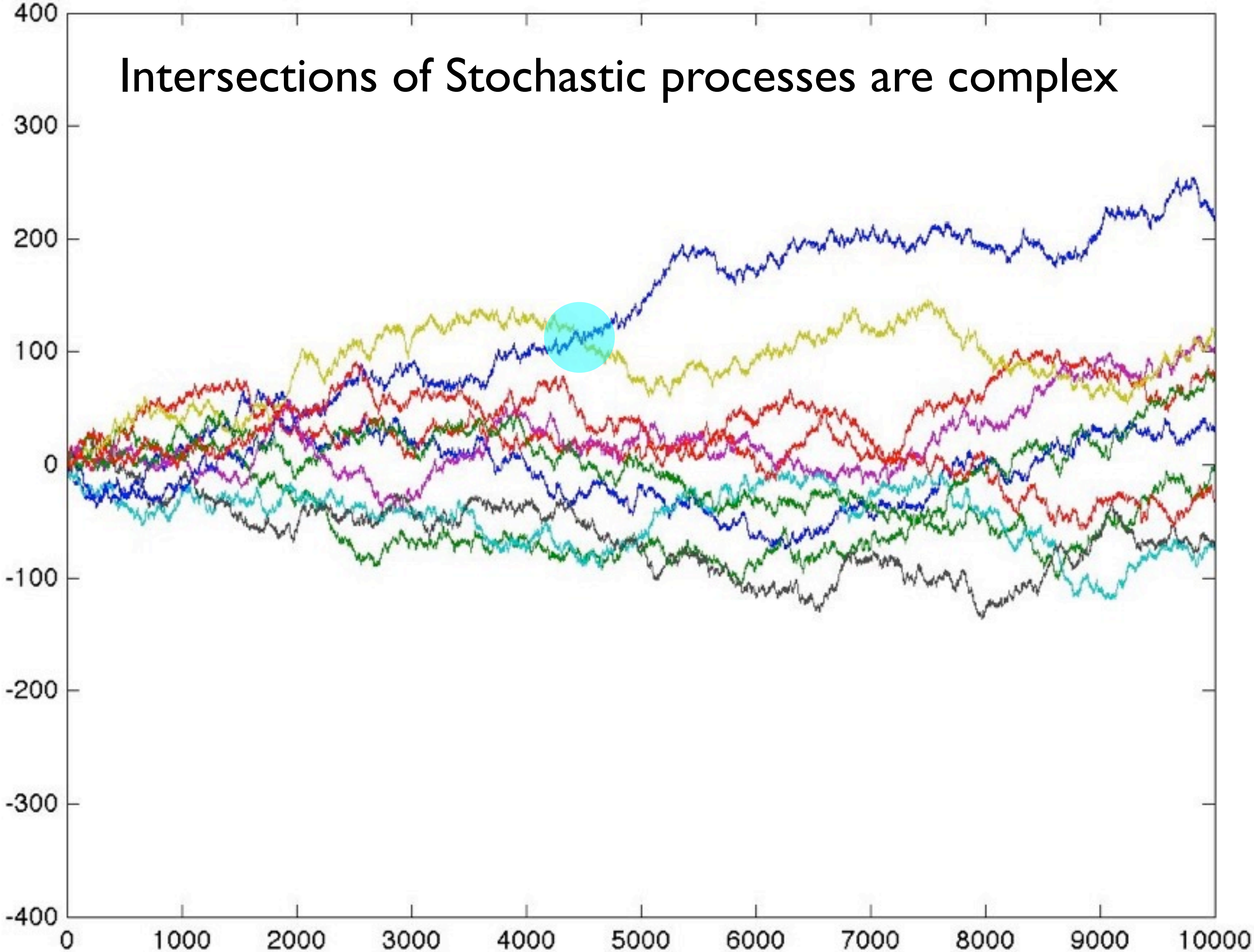
Hypothesis

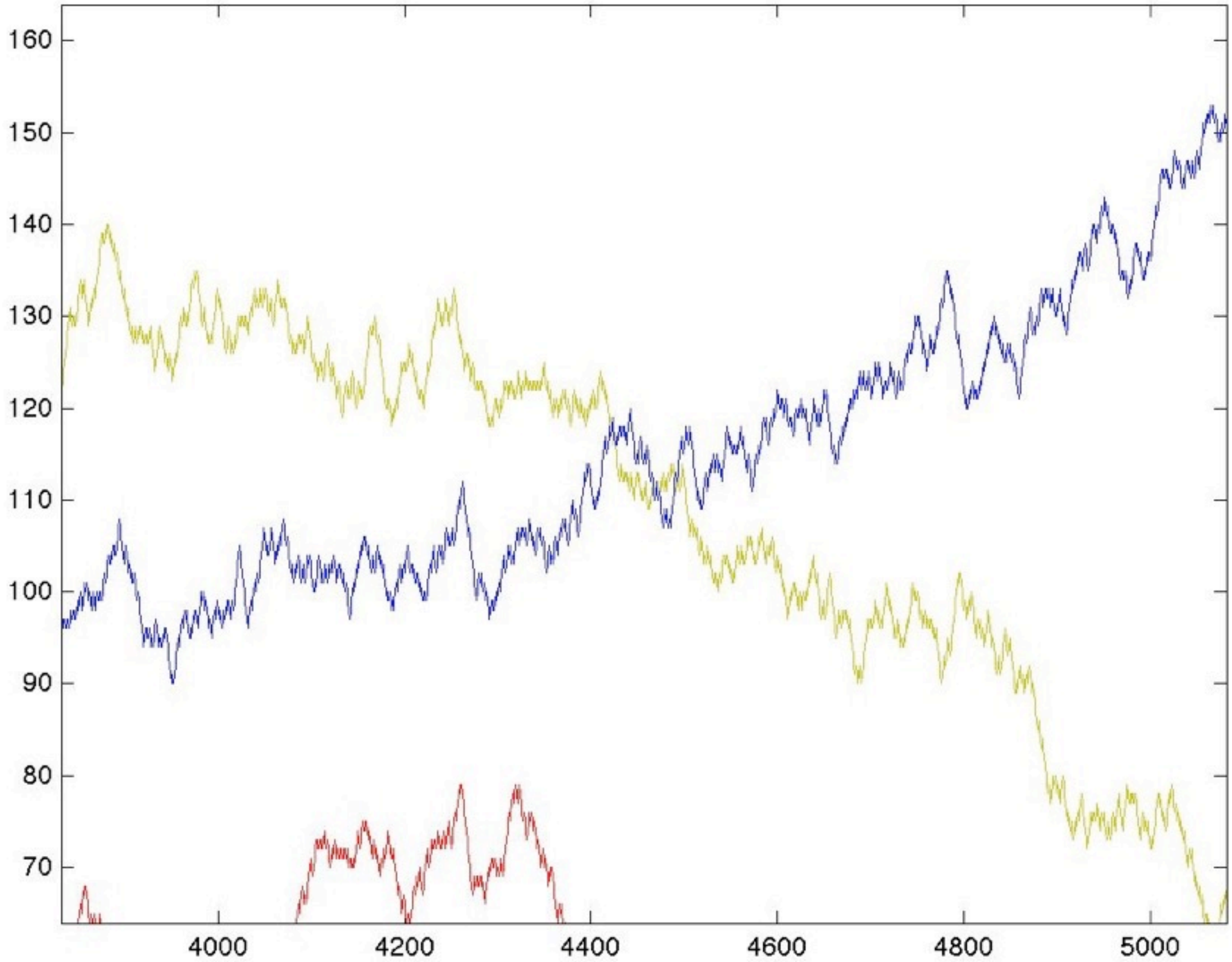
- In the bound on NormalHedge, T can be replaced by cumulative variance.
- Similar to measuring return vs. volatility in the stock market.

Hedging differentiable processes is trivial



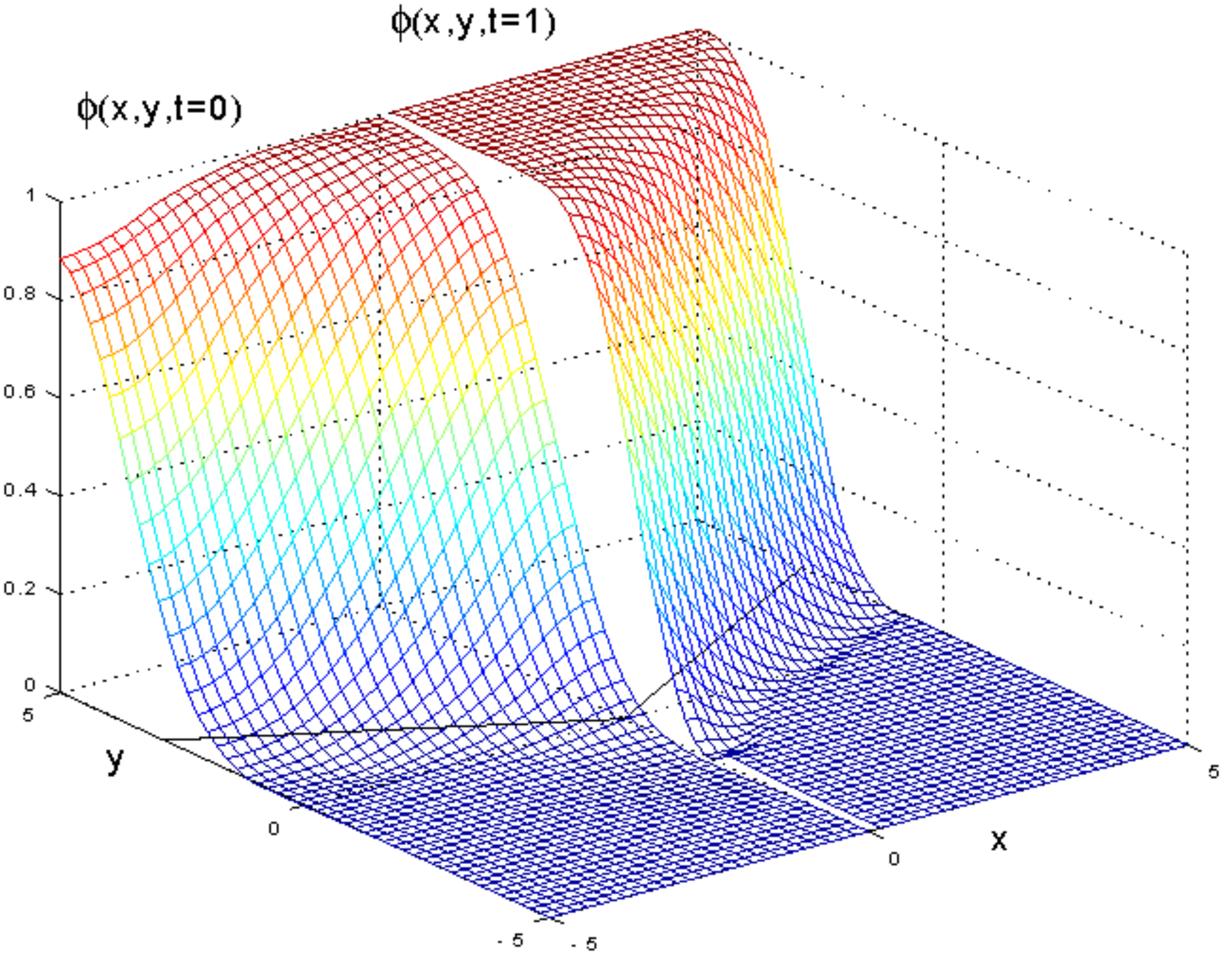
Intersections of Stochastic processes are complex



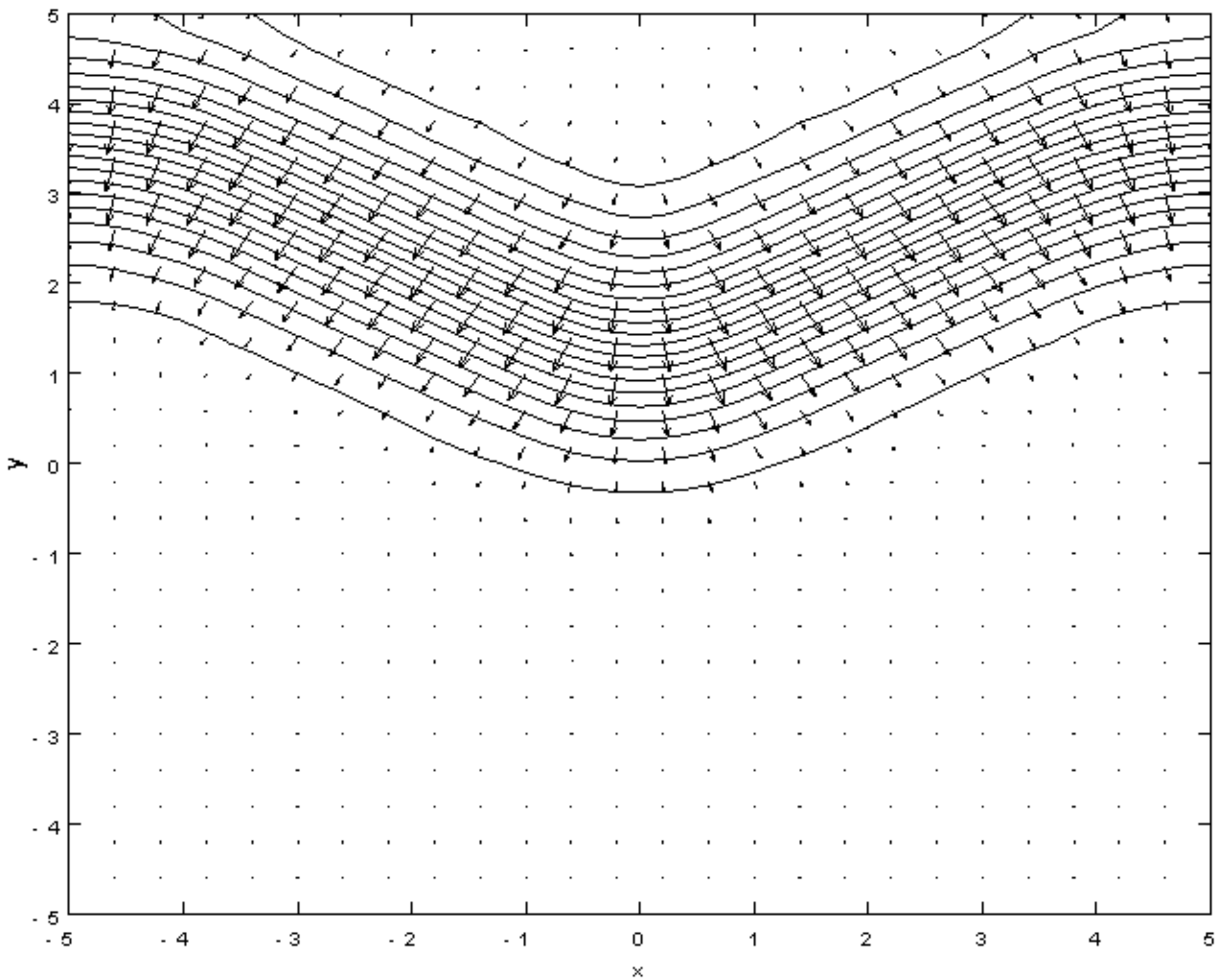

160
150
140
130
120
110
100
90
80
70
4000
4200
4400
4600
4800
5000

Thank You!

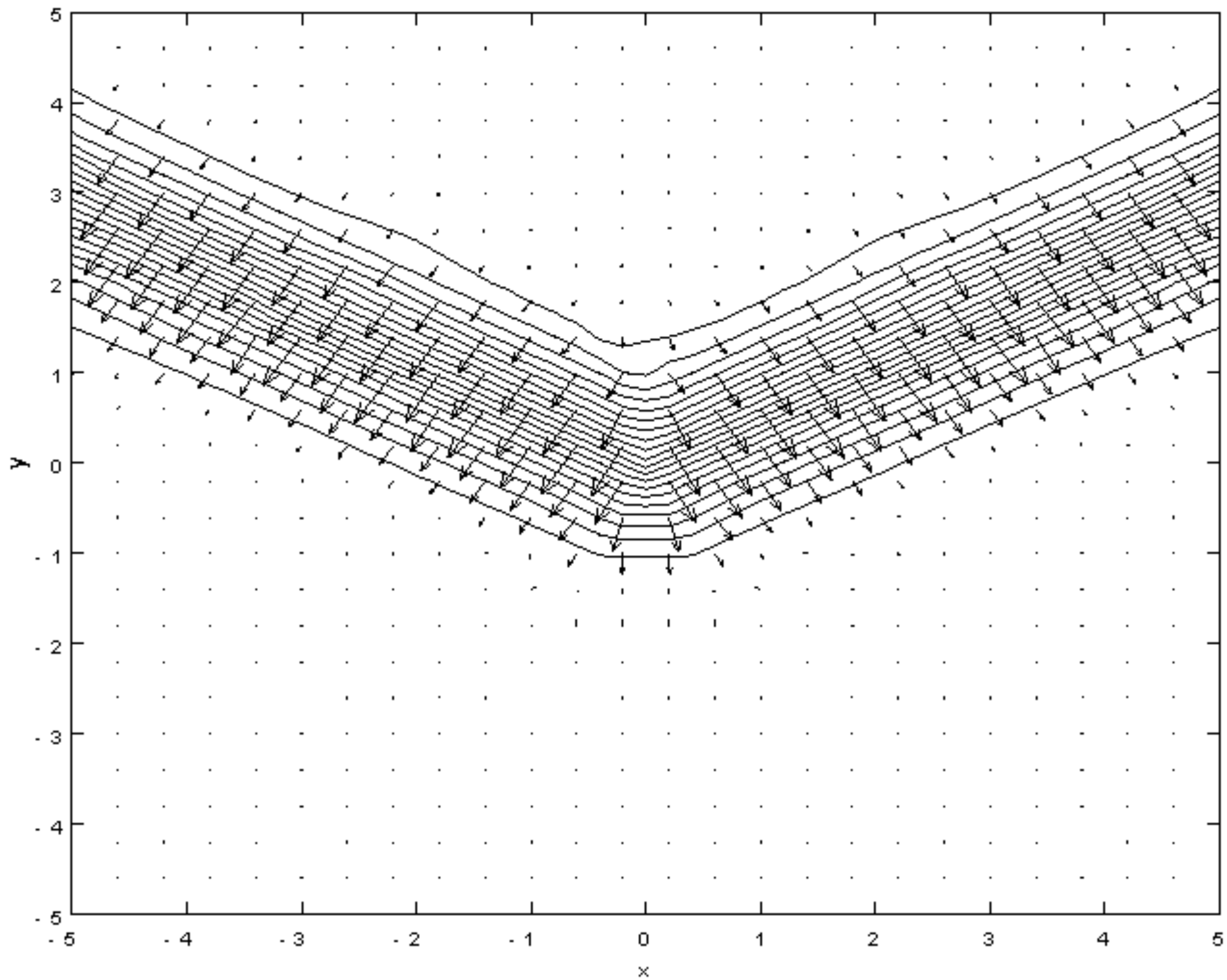
http://www.mindreaderpro.appspot.com/



φ(x,y,t=1)
φ(x,y,t=0)
1
0.8
0.6
0.4
0.2
0
5
y
0
-5
-5
0
x
5



5
4
3
2
1
0
-1
-2
-3
-4
-5
y
-5
-4
-3
-2
-1
0
1
2
3
4
5
x



5
4
3
2
1
0
-1
-2
-3
-4
-5
y
-5
-4
-3
-2
-1
0
1
2
3
4
5
x