The Algorithmic Foundations of Adaptive Data Analysis November, 2017

Lecture 15-16

Lecturer: Aaron Roth *Scribe: Aaron Roth*

The Multiplicative Weights Algorithm

In this lecture, we define and analyze a classic, powerful learning algorithm known as "Multiplicative Weights". Its uses in machine learning, game theory, and algorithms are legion: it can be used to solve linear programs, to prove von Neumann's min-max theorem, to compute Nash equilibria in zero sum games, and (coarse) correlated equilibria in arbitrary games, amongst many other applications. For us, we will use it for its ability to "guess" the answers to arbitrary linear functions of a data-set while promising that it won't make too many mistakes. This is exactly the functionality we need to pair with the AboveThreshold algorithm (either the deterministic version, analyzed via transcript compressibility, or the randomized version, analyzed via differential privacy) in order to design a statistical estimator for adaptively chosen statistical queries. As compared to the median oracle that we designed, the oracle we get from multiplicative weights will be faster to run, and will make a number of mistaken guesses that is smaller by a $\log k$ factor (which in turn will save a $\log k$ factor in the final accuracy theorem). The disadvantage will be that it is tailored for linear queries, whereas the differentially private version of the median oracle can be used to answer large numbers of arbitrary low sensitivity queries.

We start by presenting the algorithm as a means to choose sequentially amongst d discrete "actions", each of which may have a different "cost" at each round. The algorithm must make its choice at each round before the cost at that round is revealed. The goal of the algorithm is to learn to perform as well as if it had known the best (fixed) action ahead of time, and played that action at every round. This would be easier if we made assumptions about where the losses came from — for example, that they were drawn i.i.d. from some fixed distribution at each day. But for our purposes, we will want a stronger guarantee: that the algorithm can perform as well as the best action in hindsight even if the losses can be chosen adaptively by an adversary. The setting is as follows:

1. In rounds $1, \ldots, T$, the algorithm chooses some action $i^t \in \{1, \ldots, d\}$.
2. Each action i incurs a loss $\ell_i^t \in [0, 1]$. The algorithm experiences the loss of the action it chooses: $\ell_A^t = \ell_{i^t}^t$.
3. The total loss of action i is $L_i^T = \sum_{t=1}^T \ell_i^t$, and the total loss of the algorithm is $L_A^T = \sum_{t=1}^T \ell_A^t$. The goal of the algorithm is to obtain loss not much worse than that of the best action in hindsight: $\min_i L_i^T$.

What should such an algorithm do? It is tempting to be aggressive: just play the action that has minimized loss so far. You might call this strategy the "follow the leader" strategy, formalized below.

Algorithm 1 Follow the Leader

```
FtL:
  for $t = 1$ to $T$ do
    Pick action $i^t = \arg\min_i \sum_{j=1}^{t-1} \ell_i^j$ breaking ties arbitrarily.
  end for
```

Unfortunately, this algorithm does quite poorly. Consider the following sequence of losses in a two action instance:

$$\ell^1 = (0, 1) \quad \ell^2 = (1, 0) \quad \ell^3 = (0, 1) \quad \ell^4 = (1, 0) \quad \ldots \ell^{2j-1} = (0, 1) \quad \ell^{2j} = (1, 0), \ldots$$

Follow the leader will always choose the wrong action, racking up cumulative loss $L_A^T = T$. This is twice the loss of playing either fixed action: $L_1^T = L_2^T = T/2$.

Why doesn't the algorithm work? The problem is that in a sense that we will formalize, Follow the Leader is too sensitive to small changes in its loss history — it makes large changes in its action distribution in response to insignificant changes in the cumulative loss of various actions. In other words, it is over-fitting the historical losses! Fortunately, we now have a collection of techniques to deal with this. Lets try the simplest thing we can — just add a little bit of noise.

Algorithm 2 Follow the Perturbed Leader

```
FtPL($\epsilon$):
  for $t = 1$ to $T$ do
    Sample $Z_1^t, \ldots, Z_d^t \sim \text{Lap}(2/\epsilon)$.
    Pick action $i^t = \arg\min_i \left(\sum_{j=1}^{t-1} \ell_i^j + Z_i^t\right)$ breaking ties arbitrarily.
  end for
```

We will show that "Follow the Perturbed" leader is able to obtain loss that is approaching that of the best fixed action in hindsight. Specifically:

Theorem 1 *Setting $\epsilon = \sqrt{\frac{\log d}{T}}$: For any sequence of losses, and any expert k:*

$$\frac{1}{T}\,\mathbb{E}[L_{FtPL}^T] \leq \frac{1}{T} L_k^T + O\left(\sqrt{\frac{\log d}{T}}\right)$$

Proof We introduce some notation. Write $\ell^{1:t}$ to denote the summed vector $\ell^{1:t} = \sum_{j=1}^t \ell^j$. Write $M : \mathbb{R}^d \to \mathbb{R}^d$ to denote the function such that $M(v)_{i^*} = 1$ where $i^* = \arg\min_i v_i$ and $M(v)_i = 0$ otherwise. In this notation, at each round t, "Follow the Leader" obtains loss $M(\ell^{1:t-1}) \cdot \ell^t$ and "Follow the Perturbed Leader" obtains loss $M(\ell^{1:t-1} + Z^t) \cdot \ell^t$. At the end of play, at time T, the best action in hindsight obtains cumulative loss $M(\ell^{1:T}) \cdot \ell^{1:T}$.

The proof of this theorem will go through a thought experiment. Consider an imaginary algorithm called "be the leader", which at round t plays according to $M(\ell^{1:t})$. (We can't use this algorithm, since we don't know ℓ^t at the time at which we choose action t.) We will first show that this imaginary algorithm obtains loss that is only lower than that of the best action in hindsight.

Lemma 2

$$\sum_{i=1}^{T} M(\ell^{1:t}) \cdot \ell^t \leq M(\ell^{1:T}) \cdot \ell^{1:T}$$

Proof This follows by a simple induction on T. For $T = 1$, it holds with equality. Now assume it holds for general T – we show it holds for the next time step:

$$\sum_{i=1}^{T+1} M(\ell^{1:t})\cdot\ell^t \leq M(\ell^{1:T})\cdot\ell^{1:T} + M(\ell^{1:T+1})\cdot\ell^{T+1} \leq M(\ell^{1:T+1})\cdot\ell^{1:T} + M(\ell^{1:T+1})\cdot\ell^{T+1} = M(\ell^{1:T+1})\cdot\ell^{1:T+1}$$

■

Great! Next, we show that "be the perturbed leader", which at round t plays according to $M(\ell^{1:t} + Z^t)$, doesn't do much worse.

Lemma 3 *For any set of loss vectors $\ell^1, \ldots, \ell^T$ and any set of perturbation vectors $Z^0 \equiv 0, Z^1, \ldots, Z^T$:*

$$\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot \ell^t \leq M(\ell^{1:T}) \cdot \ell^{1:T} + 2\sum_{t=1}^{T} ||Z^t - Z^{t-1}||_\infty$$

Proof Define $\hat{\ell}^t = \ell^t + Z^t - Z^{t-1}$. Note that $\hat{\ell}^{1:t} = \ell^{1:t} + Z^t$, since the sum telescopes. Thus, we can apply Lemma 2 on the sequence $\hat{\ell}$ to conclude:

$$\begin{aligned}\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot (\ell^t + Z^t - Z^{t-1}) &\leq M(\ell^{1:T} + Z^T) \cdot (\ell^{1:T} + Z^T) \\ &\leq M(\ell^{1:T}) \cdot (\ell^{1:T} + Z^T) \\ &= M(\ell^{1:T}) \cdot \ell^{1:T} + \sum_{t=1}^{T} M(\ell^{1:T}) \cdot (Z^t - Z^{t-1})\end{aligned}$$

Subtracting from both sides, we have:

$$\sum_{t=1}^{T} M(\ell^{1:t}+Z^t)\cdot\ell^t \leq M(\ell^{1:T})\cdot\ell^{1:T}+\sum_{t=1}^{T}(M(\ell^{1:T})-M(\ell^{1:t}+Z^t))\cdot(Z^t-Z^{t-1}) \leq M(\ell^{1:T})\cdot\ell^{1:T}+\sum_{t=1}^{T} 2||Z^t-Z^{t-1}||_\infty$$

■

We will use this lemma and a trick to compute the expected regret of "be the perturbed leader". Since expectations distribute over sums, we have:

$$\mathbb{E}[\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot \ell^t] = \sum_{t=1}^{T} \mathbb{E}[M(\ell^{1:t} + Z^t) \cdot \ell^t]$$

Hence, the expectation remains unchanged in the thought experiment under which the Laplace perturbation is not resampled at every step, and instead $Z^1 = \ldots = Z^t \sim \text{Lap}(1/\epsilon)$. Applying Lemma 3 to this version of be the perturbed leader, we obtain:

$$\mathbb{E}[\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot \ell^t] \leq M(\ell^{1:T}) \cdot \ell^{1:T} + 2\,\mathbb{E}[||Z^1||_\infty] \leq M(\ell^{1:T}) \cdot \ell^{1:T} + O(\frac{\log d}{\epsilon}).$$

Finally, we will analyze "follow the perturbed leader" by considering the multiplicative distance of its distribution over play from the distribution over play of "be the perturbed leader". Fix a sequence $\ell^1, \ldots, \ell^T$ of loss functions. Let $A_1, \ldots, A_T$ and $B_1, \ldots, B_T$ be random variables describing the action sequence of follow the perturbed leader and be the perturbed leader respectively, on this action sequence. Then we have:

Lemma 4 *For each round t, $d_\diamond(A_t, B_t) \leq \epsilon$.*

Proof We can view follow the leader as selecting an action using "report noisy max", over the dataset consisting of the $t-1$ loss functions observed so far. Similarly, be the leader consists of the same algorithm run on a *neighboring* dataset — i.e. the same dataset, but with the addition of one additional data point, ℓ^t. The result then follows from our analysis of report noisy max. ■

In other words, we can view each step of follow the perturbed leader as an ϵ-differentially private algorithm where the vectors of losses correspond to the entries in the "data set". We therefore have that for each t:

$$\mathbb{E}[M(\ell^{1:t-1} + Z^t) \cdot \ell^t] \leq e^\epsilon\, \mathbb{E}[M(\ell^{1:t} + Z^t) \cdot \ell^t]$$

Combining this bound with the regret bound we have proven for "be the perturbed leader" yields:

$$\sum_{t=1}^{T} \mathbb{E}[M(\ell^{1:t-1} + Z^t) \cdot \ell^t] \leq e^\epsilon\, \mathbb{E}[\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot \ell^t] \leq e^\epsilon \left(M(\ell^{1:T}) \cdot \ell^{1:T} + O(\frac{\log d}{\epsilon}) \right) \leq$$

$$(1+2\epsilon) \left(M(\ell^{1:T}) \cdot \ell^{1:T} + O(\frac{\log d}{\epsilon}) \right) \leq M(\ell^{1:T}) \cdot \ell^{1:T} + O(\frac{\log d}{\epsilon}) + \epsilon \cdot T + O(\log d)$$

Setting $\epsilon = \sqrt{\frac{\log d}{T}}$ and dividing by T yields the theorem[1]:

$$\frac{1}{T}\,\mathbb{E}[L_{FtPL}^T] \leq \frac{1}{T} L_k^T + O\left(\sqrt{\frac{\log d}{T}}\right)$$

■

Ok – that was a randomized algorithm that sampled from a distribution that would be a little tricky to get a handle on. What if we are in a setting in which rather than selecting a single action, we can propose a probability distribution over actions, and directly observe our "expected" reward. In other words, the algorithm can choose at each round a distribution $p^t \in [0,1]^d$, and obtains loss at each round $\ell_A^t = p^t \cdot \ell^t$[2]. Here, we analyze a different algorithm that obtains the same asymptotic regret bound. The advantage of this algorithm is that it explicitly maintains a distribution p^t at each round, and it is easier to get a handle on the constants in the regret bound. The algorithm is called "Multiplicative Weights"

Algorithm 3 The Multiplicative Weights Algorithm (MW)

```
MW($\eta$):
  Set weights $w_i^1 \leftarrow 1$ for all experts $i$.
  for $t = 1$ to $T$ do
    Let $W^t = \sum_{i=1}^N w_i^t$.
    Define a probability distribution $p^t$ such that $p_i^t = w_i^t / W^t$.
    For each $i$, set $w_i^{t+1} \leftarrow w_i^t \cdot (1 - \eta \ell_i^t)$.
  end for
```

Theorem 5 *For any sequence of losses, and any expert k:*

$$\frac{1}{T} L_{MW}^T \leq \frac{1}{T} L_k^T + \eta + \frac{\ln(d)}{\eta \cdot T}$$

In particular, setting $\eta = \sqrt{\frac{\ln(d)}{T}}$ we get:

$$\frac{1}{T} L_{MW}^T \leq \frac{1}{T} \min_k L_k^T + 2\sqrt{\frac{\ln(d)}{T}}$$

A couple of things to note. First, this algorithm obtains the same asymptotic regret bound as follow the perturbed leader. Second, it actually isn't so different: the distribution it is maintaining can be viewed as a variant of the "exponential mechanism" from differential privacy, so we can also view this as an algorithm that is attempting to play the best action in hindsight at each round, but selected via a differentially private mechanism. However, we will give a direct analysis (and the way we will use the algorithm will be by using the distribution p^t at each round, rather than sampling from it, so the algorithm in our view will be deterministic.

Ok, on to the proof:

Proof Let F^t denote the loss of the multiplicative weights algorithm at time t: $L_{MW}^T = \sum_{t=1}^T F^t$. We also know that:

$$F^t = p^t \cdot \ell^t = \frac{\sum_{i=1}^d w_i^t \ell_i^t}{W^t}$$

How does W^t change between rounds? We know that $W^1 = d$, and looking at the algorithm we see:

$$W^{t+1} = W^t - \sum_{i=1}^d \eta w_i^t \ell_i^t = W^t(1 - \eta F^t)$$

[1]We also note that for $T > \log d$, $\sqrt{\frac{\log d}{T}} > \frac{\log d}{T}$. For $T \leq \log d$, the regret can be trivially bounded by 1.

[2]This would be the algorithm's expected loss if it played an action sampled from p^t

So by induction, we can write:

$$W^{T+1} = d\prod_{t=1}^{T}(1-\eta F^t)$$

Taking the log, and using the fact that $\ln(1-x) \le -x$, we can write:

$$\begin{aligned}\ln(W^{t+1}) &= \ln(d) + \sum_{t=1}^{T}\ln(1-\eta F^t)\\ &\le \ln(d) - \eta\sum_{t=1}^{T}F^t\\ &= \ln(d) - \eta L_{MW}^T\end{aligned}$$

Similarly (using the fact that $\ln(1-x) \ge -x - x^2$ for $0 < x < \frac{1}{2}$), we know that for every action k:

$$\begin{aligned}\ln(W^{T+1}) &\ge \ln(w_k^{T+1})\\ &= \sum_{t=1}^{T}\ln(1-\eta\ell_k^t)\\ &\ge -\sum_{t=1}^{T}\eta\ell_k^t - \sum_{t=1}^{T}(\eta\ell_k^t)^2\\ &\ge -\eta L_k^T - \eta^2 T\end{aligned}$$

Combining these two bounds, we get:

$$\ln(d) - \eta L_{MW}^T \ge -\eta L_k^T - \eta^2 T$$

for all k. Dividing by η and rearranging, we get:

$$L_{MW}^T \le \min_k L_k^T + \eta T + \frac{\ln(N)}{\eta}$$

■

A simple corollary is that the algorithm performs nearly as well as any fixed *distribution* p^* over actions in hindsight.

Corollary 6 *For any sequence of losses, and any distribution p^* over actions k, setting $\eta = \sqrt{\frac{\ln(d)}{T}}$ we get:*

$$\frac{1}{T}L_{MW}^T \le \frac{1}{T}\sum_{t=1}^{T}p^*\cdot\ell^t + 2\sqrt{\frac{\ln(d)}{T}}$$

Proof Observe that:

$$\frac{1}{T}\sum_{t=1}^{T}p^*\cdot\ell^t = \sum_{k=1}^{d}p_k^*L_k^T$$

We know that for each of the d possible choices of action k,

$$\frac{1}{T}L_{MW}^T \le \frac{1}{T}\min_k L_k^T + 2\sqrt{\frac{\ln(d)}{T}}$$

Taking a weighted sum of these inequalities, assigning weight p_k^* to inequality k, and noting that $\sum_k p_k^* = 1$ yields the corollary. ■

We want to use this guarantee to give a method for "guessing" the answers to statistical queries, up to some error tolerance α, such that we are guaranteed to make only a bounded number of mistakes. This will allow us to apply the machinery we developed in both the compressibility and stability sections to obtain bounds for statistical estimators for answering many statistical queries in the adaptive setting.

The idea will be to think of the data set S itself as a distribution over the data universe $\mathcal{X}$ — just the empirical distribution that is uniform over the n elements of the dataset. We can think of such a distribution as a vector $p^S \in [0,1]^{|\mathcal{X}|}$ such that it takes value $p^S_j = |\{x \in S : x = j\}|/n$. Note that $||p_S||_1 = 1$. Similarly, we can think of a statistical query ϕ as represented by a vector $q^\phi \in [0,1]^{|\mathcal{X}|}$, such that for a coordinate x, $q^\phi_x = \phi(x)$. With this representation, note that $\phi(S) = \langle q^\phi, p^S \rangle$. Here is how we will use the multiplicative weights mechanism (or any other no-regret algorithm from which we can extract a distribution over play) as a sub-routine for answering queries with a bounded number of mistakes. We assume that we have a way of determining whether our guess $\hat{a}_t$ was either *too big* — i.e. $\hat{a}_t \geq \phi(S) + \alpha$, or *too small* — i.e. $\hat{a}_t \leq \phi_t(S) - \alpha$. The guarantee we will give is that the number of times these checks will trigger will be bounded.

```
MW-Guesser($\alpha$):
  Initialize a copy of $MW$.
  for Each query $\phi_t$ do
    Extract the current distribution $p^t$ maintained by $MW$.
    Guess the answer $\hat{a}_t = \langle q^{\phi_t}, p^t \rangle$.
    if TooBig($\hat{a}_t, S$) then
      Feed $MW$ the loss function $\ell_t = q^{\phi_t}$.
    else if TooSmall($\hat{a}_t, S$) then
      Feed $MW$ the loss function $\ell_t = 1 - q^{\phi_t}$.
    else
      Do not update $MW$.
    end if
  end for
```

When either TooBig($\hat{a}_t, S$) or TooSmall($\hat{a}_t, S$) triggers, we will say that MW-Guesser *made a mistake*. We can prove:

Theorem 7 *For any T, and for any sequence of statistical queries $\phi_1, \ldots, \phi_T$, MW-Guesser makes at most M mistakes for:*

$$M = \frac{4 \log |\mathcal{X}|}{\alpha^2}$$

Proof The proof strategy will be to show that at each mistake, the MW algorithm MW-Guesser is running accumulates additional regret α. After too many mistakes, this will contradict the regret bound we proved for MW.

We divide the mistakes into their two types:

$$B = \{t : \text{TooBig}(\hat{a}_t, S)\} \qquad L = \{t : \text{TooSmall}(\hat{a}_t, S)\}$$

We separately bound the regret for each type of mistake. We use p^S, the distribution corresponding to the dataset S.

$$\begin{aligned}
\sum_{t \in B} \langle p^t, \ell^t \rangle - \langle p^S, \ell^t \rangle &= \sum_{t \in B} \langle p^t, q^{\phi_t} \rangle - \langle p^S, q^{\phi_t} \rangle \\
&= \sum_{t \in B} \hat{a}_t - \phi_t(S) \\
&\geq \alpha \cdot |B|
\end{aligned}$$

where the last inequality holds because of the definition of TooBig. Similarly:

$$\begin{aligned}\sum_{t\in L}\langle p^t,\ell^t\rangle-\langle p^S,\ell^t\rangle &= \sum_{t\in L}\langle p^t,1-q^{\phi_t}\rangle-\langle p^S,1-q^{\phi_t}\rangle\\ &= \sum_{t\in L}\langle p^S,q^{\phi_t}\rangle-\langle p^t,q^{\phi_t}\rangle\\ &= \sum_{t\in L}\phi_t(S)-\hat{a}_t\\ &\geq \alpha\cdot|L|\end{aligned}$$

Combining these two bounds, we see:

$$\frac{1}{|B|+|L|}\sum_{t\in B\cup L}\langle p^t,\ell^t\rangle\geq\frac{1}{|B|+|L|}\sum_{t\in B\cup L}\langle p^S,\ell^t\rangle+\alpha$$

On the other hand, we know from the regret bound of multiplicative weights:

$$\frac{1}{|B|+|L|}\sum_{t\in B\cup L}\langle p^t,\ell^t\rangle\leq\frac{1}{|B|+|L|}\sum_{t\in B\cup L}\langle p^S,\ell^t\rangle+2\sqrt{\frac{\ln(|\mathcal{X}|)}{|B|+|L|}}$$

Together, these two bounds imply:

$$\alpha\leq 2\sqrt{\frac{\ln(|\mathcal{X}|)}{|B|+|L|}}$$

Solving, we get:

$$M=|B|+|L|\leq 4\frac{\ln|\mathcal{X}|}{\alpha^2}$$

■

Now recall from Lecture 12: for any $\epsilon,\delta>0$, we can use the differentially private version of GuessAndCheck (which we can now analyze with our advanced composition theorem for differential privacy, from Lecture 13) to answer any k statistical queries with empirical error that is with probability $1-\beta$ at most $\alpha=O(\frac{\ln(k/\beta)}{n\epsilon})$, and so long as our queries are accompanied by guesses, of which at most m have empirical error more than α, will satisfy (ϵ',δ)-differential privacy for:

$$\epsilon'\leq\epsilon\sqrt{2m\ln(1/\delta)}+m\epsilon\frac{e^\epsilon-1}{e^\epsilon+1}$$

Thus, we can satisfy (ϵ',δ)-differential privacy for any level of ϵ we like, by setting $\epsilon=O(\epsilon'/\sqrt{2m\ln(1/\delta)})$. Now recall our transfer theorem from the last lecture: if we have a statistical estimator that is simultaniously:

1. $(\alpha,\alpha\beta)$-differentially private, and
2. (α,β)-sample accurate,

then it is $(O(\alpha),O(\beta))$-accurate on the underlying distribution.

We can optimize over our choice of ϵ. Recall that we have (α,β)-sample accuracy for $\alpha=O(\frac{\sqrt{2m\ln(1/\alpha\beta)}\ln(k/\beta)}{n\epsilon'})$ (setting $\delta=\alpha\beta$). Recall also that we can guess all but m answers correctly up to error α using MW-Guesser, for $m=\frac{4\log|\mathcal{X}|}{\alpha^2}$. Plugging this in, and setting $\alpha=\epsilon'$, we get:

$$\alpha=\epsilon'=\tilde{O}\left(\frac{\left(\ln\frac{k}{\beta}\right)^{1/3}(\log|\mathcal{X}|\ln(1/\beta))^{1/6}}{n^{1/3}}\right)$$

Applying our transfer theorem, we immediately obtain an (α, β)-accurate mecahanism for answering any k adaptively chosen set of statistical queries, for the above value of α.

Compared to the bound we got from the median mechanism, the improvement in accuracy is quite small (a fractional power of a $\log k$). However, the multiplicative weights algorithm runs in time per-iteration that is only linear in $|\mathcal{X}|$, rather than superpolynomial in $|\mathcal{X}|$, which is a substantial improvement. It can be feasibly run when $|\mathcal{X}|$ is small.

Bibliographic Information Follow the perturbed leader was given and analyzed in [KV05]. The multiplicative weights algorithm is ubiquitious across computer science, and has been rediscovered multiple times — see [AHK12] for a survey and collection of applications. The multiplicative weights algorithm for privately answering statistical queries was given by Hardt and Rothblum [HR10].

References

[AHK12] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[HR10] Moritz Hardt and Guy Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *Proc. 51st Foundations of Computer Science (FOCS)*, pages 61–70. IEEE, 2010.

[KV05] Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.