In the classical multi-armed bandit problem, a gambler is presented with a set of k probability distributions $D_1, D_2, \cdots D_k$ corresponding to arms on a slot machine. At each turn $t = 1, 2, ..., n$, the gambler selects an arm with index j and receives a reward drawn from D_j.

In typical online advertising contexts, we can make the simplifying assumption that the users are independent and either click on the ad or do not, such that each of the probability distributions $D_i = \text{Bernoulli}(p_i)$. However, in real-world applications we often observe that global trends cause the probabilities p_i to depend on t in a correlated way such that at certain times all of the arms have significantly increased or reduced reward rates. In other words:

$$D_i = \text{Bernoulli}(p_i \alpha(t)) \tag{1}$$

where α is a slowly varying function of t.

The most straightforward way to adjust a policy for selecting arms is to attempt to extract the p_js from the observed success rates as follows

```
data {
int<lower=0> i; //number of iterations
int<lower=0> j; //number of timebins
int<lower=0> k; // number of arms

int<lower=0,upper=1> y[i]; //success or failure of each iteration
int<lower=1,upper=j> t[i]; //timebin of each iteration
int<lower=1,upper=k> a[i]; //arm chosen at each iteration
}

parameters {
real theta[j]; // models alpha(t)
real eta[k]; // models p_j
real<lower=0> S; // random walk distance for alpha(t)
}

model {
S ~ gamma(1,0.5); // prior on random walk distance

for (kk in 1:k) {
eta[kk] ~ normal(-3,10); // weak prior on arm probabilities
}

theta[1] <- 0.0;

for (mm in 2:j) {
theta[mm] ~ normal(theta[mm-1],S); //thetas are a gaussian random walk
}

for (nn in 1:i) {
```

```
y[nn] ~ bernoulli(inv_logit(eta[a[nn]]+theta_temp[t[nn]]));
}
}
```

Let $S_{i,m}$ and $N_{i,m}$ be the number of successes and trials for arm i in timebin m. Then we estimate p_i as

$$\hat{p}_i = \frac{\sum_m S_{i,m} e^{-\theta(m)}}{\sum_m N_{i,m}} \tag{2}$$

where the θ_ms are as computed above. Using these estimates for p_j we can compute our favorite index for example the upper confidence bound (UCB), choosing the arm that maximizes

$$(UCB)_j = \hat{p}_j + \sqrt{\frac{2\log(N)}{N_j}} \tag{3}$$