Statistical Techniques in Robotics (16-831, F10) Lecture#12 (Monday October 10)

Random fields for Ladar data and estimation

Lecturer: Drew Bagnell *Scribe: Jonathan Butzke*[1]

1 Portfolio Optimization Continued - No Regret Portfolio

Note: Do not actually use the investment algorithm we discussed in the last lecture - you will lose money!

Recall the algorithm from last time: Start with

$$w_0 \leftarrow \frac{1}{n} \tag{1}$$

Then

$$w_{t+1} \leftarrow Proj[w_t + \frac{\alpha r^t}{w_t^T r_t}] \tag{2}$$

Remember, we control the w_t, the objective function is r_t^i (the return of the i^{th} investment during timestep t) and this is no regret with respect to a constant weight $w^\star$ (This means it is a constantly rebalanced portfolio, after every timestep, the assets are moved around such that their starting weight each day is the same - it is not the portfolio where you pick the best asset each day from some oracle).

So for each day $w^T r_t$ is the total return for that day and your total return at the end is $\prod_t w_t^T r_t$.

We will optimize $\sum \log w_t^T r_t$. (As an aside, it seems people actual care about money in a "sort of" logarithmic way, but we are really doing this because it is convenient mathematically).

On day 1, buy at some arbitrary set of weights as in equation 1 where w_0 can be set using any a priori knowledge available.

Then for each day calculate

$$\nabla \ell = \frac{r_t}{w_t^T r_t} \tag{3}$$

$$w_{t+1} = w_t + \alpha \nabla \ell \tag{4}$$

by restricting r such that $r_{min} \leq r_t \leq r_{max}$ we get

$$\alpha = \sqrt{\frac{F}{TG}} = \sqrt{\frac{2}{TN\frac{r_{max}}{r_{min}}}} \tag{5}$$

How can this go wrong?

[1]Notes based on work from previous scribes: Javier Hernandez Rivera and Heather L. Jones

1. We made a no junk bond assumption

$$r_{min} \leq r_t \leq r_{max} \tag{6}$$

This assumption fails if there is a catastrophic market crash.

2. How do you do the projection?

If $w^i \leq 0$, then $w^i = 0$. Now everything doesn't sum to 1. To do the projection correctly, sort the weights, smallest to largest. Then look at $\sum_i max(w_i - a, 0)) = 1$, and find an a such that this is true.

Gain function: $log(w^T r)$ (take the negative to get the loss)

L_2 Projection (minimize squared difference between point before and after projection)

$$R \leftarrow \sqrt{G} * \sqrt{T} * \sqrt{D(w_0, w*)} \tag{7}$$

3. Transaction costs are not accounted for

4. Moving the market - if you put a lot of money in, you affect the price. We had regret $\sum_t l_t(w_t) - l_t(w*)$. If these losses are a function of decisions you made, then it doesn't mean a lot if you have small regret (you just made the best expert perform worse). This shows up in other robotics applications too: the robot may see something different because it makes actions based on what it sees. For example, the robot decides that a green thing far away is bad and wants to stay as far away from it as possible. Then the robot never learns that maybe green isn't so bad.

All these problems exist in every application of no regret. You're still minimizing your regret, but maybe you aren't minimizing what you hoped you were minimizing. So, no regret is cool, but not magic (ok, maybe a little magic).

Why is projection OK? When you project onto a convex set, you get closer to every point in the convex set.

$$R_t \leq \bigtriangledown l_t^T (w_t - w*) \leq \frac{\alpha G}{2} + [D(\tilde{w}_t, w*) - D(w_{t-1}, w*)] \tag{8}$$

The term $D(\tilde{w}_t, w*) - D(w_{t+1}, w*)$ is always positive (See Figure 1).

Not only does projection not hurt, it could help. We have to get $D(\tilde{w}_t, w*)$ to go down - for this we need the L_2 distance, not L_1. Even if the set is huge, no regret with respect to $D(w_0, w*)$.

$D()$ is the L_2 distance. *Mirror descent* is for other D's besides $(w_0 - w*)^2$.

$\alpha = \sqrt{\frac{F}{GT}}$, if you don't know the horizon, use little t instead of T, α_t instead of a constant α.

$R \leq \sqrt{FGT}$

This is a strongly convex function (bounded by a quadratic) - the quadratic has some steepness.

Every strongly convex function can be written as a convex function.

$f(x) = g(x) + \frac{H}{2} * ||x||^2$, (with f(x) strong)

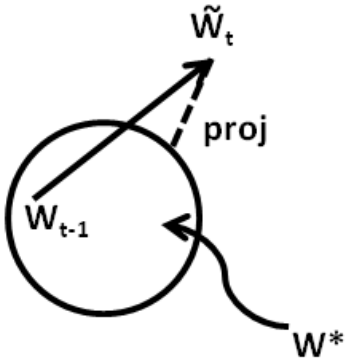

Figure 1: Projection onto a convex set

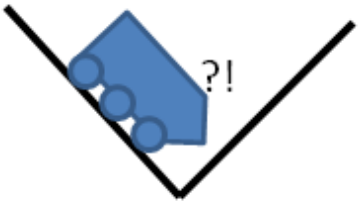

Figure 2: Change in slope is bad

$abx(x) + ||x||^2$

The market problem we just did does not have this property.

$\alpha = \frac{1}{Ht}$, and for this kind of function, $R \leq \frac{G}{2H}(1 + log(T))$, where $\sum \frac{1}{T}$ is bounded by $log(T)$ - notice no dependance on F, so regret grows only logarithmically with the number of time steps.

2 Markov Random Field Applications

2.1 Elevation Map from Laser Data

[Projected slides showing terrain as sensed from a plane]

The laser in this example has issues - sometimes it does not penetrate vegetation, and sometimes it appears to go through the ground.

We know that a change in slope is bad for a robot on the ground (See Figure 2). We can represent the ground as an MRF (See Figure 3). The blue nodes represent the height of the ground h_i, and the orange nodes represent the range readings, z_i. The potential functions are defined as:

$$\Phi_{ij} = e^{-\lambda(h_i - h_j)^2} \tag{9}$$

$$\Psi_i = e^{-\sigma(z_i - h_i)^2} \tag{10}$$

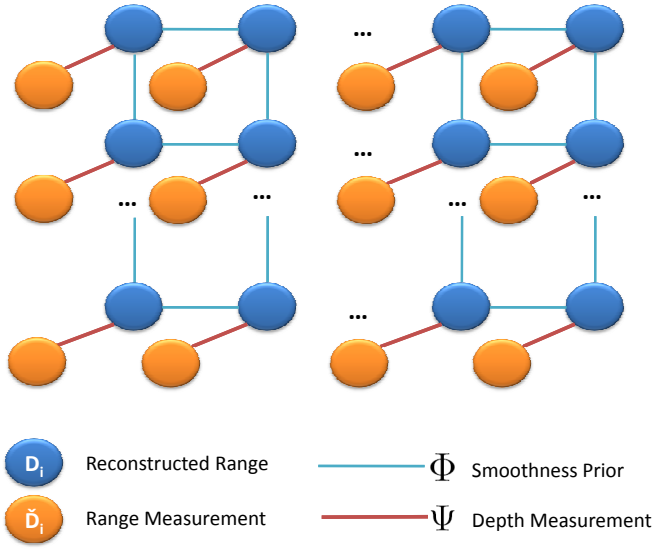

Figure 3: Markov Random Field for Range Sensing

where λ is a constant weight placed on the depth measurements.

$$Pr(h) = \frac{1}{Z} \prod_i \Psi_i \prod_{i,j \in neighboringpairs} \Psi_{ij} \tag{11}$$

With Z as the normalizing factor.

Today: $argmax_h Pr(h|z)$

Previously: $h_{sample} \tilde{} Pr(h|z), E[h_i]$

These two need not agree, but they will here because for a Gaussian distribution, mean = mode.

There is also a third thing (which we won't ever do): $argmax_{h_i} Pr(h_i|z)$, (this is the max marginal distribution)

Now take the log of today's formulation:

$log(Pr(h)) = C - \lambda \sum (h_i - h_j)^2 - \sigma_i \sum (z_i - h_i)^2$

This is convex.

$\bigtriangledown^i_{l_t} = -2\lambda \sum_{j \in neighbor i} (h_i - h_j) - 2\sigma(z_i - h_i)$, where the superscript i indicates that the gradient is with respect to h_i.

After applying this, the terrain map [on projected slide] looks smoother.

An alternative way of writing this is:

$$\min_{\bar{x}}(-\log P(\bar{x}|observations)) \rightarrow \min_{\bar{x}} \sum_i [\gamma(x_i - h_{i,min})^2 + \lambda \sum_{j \in Neighbors} (|x_i - x_j|)]$$

$$\rightarrow (\sum_i 2\gamma(x_i - h_{i,min}) + \lambda \sum_j (sign(x_i - x_j))(\gamma - \alpha)$$

where

$$sign(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ -1 & \text{if } z < 0 \end{cases}$$

2.2 Bookshelf with Image and Laser (Depth) Data

The trick in this application is to realize that the image is higher resolution than the depth data. We can use essentially the same MRF model as in Figure 3, but now the z_t nodes only connect to every third (or tenth, etc.) h_i node, but there is a camera pixel for each h_i node with color c_i.

$\delta(color(r_1), color(r_2)) = \sum(p_i - p_j)^2$ for pixels $p_i and p_j$.

From the equation we had before:

$$\Phi_{ij} = e^{-\lambda(h_i - h_j)^2} \tag{12}$$

We now set $\lambda^{ij} = (constant) * e^{-D(c_i, c_j)}$.

2.3 Terrain Mapping for a Ground Robot

The objective in this problem is to estimate the height of the terrain observed with range sensors. Let's assume the set of heights can be described by a function $Z = f(x, y)$ where $Z = 0$ corresponds to the ground level. Figure 4 illustrates a MRF to solve this problem. Note that in this case, Z is defined in 1D.

Similarly to the previous problem, we can define the potential functions as:

$$\Psi = e^{-(\tilde{D}_i - D_i)^2} \tag{13}$$

$$\Phi = e^{-w_{ij}(D_i - D_j)^2} \tag{14}$$

Moreover, we have the set of constraints $\forall D_i \leq \text{Ray height}_i$ that require a projection function to be satisfied. Note that the set of solutions is convex because it is defined with inequalities.

This assumes no reflections (a puddle could make the robot think there was a hole).

In this case, we get new heights at every time step, so we have a convex set that changes at every time step. This is still OK, because the sequence of convex sets keeps getting smaller.

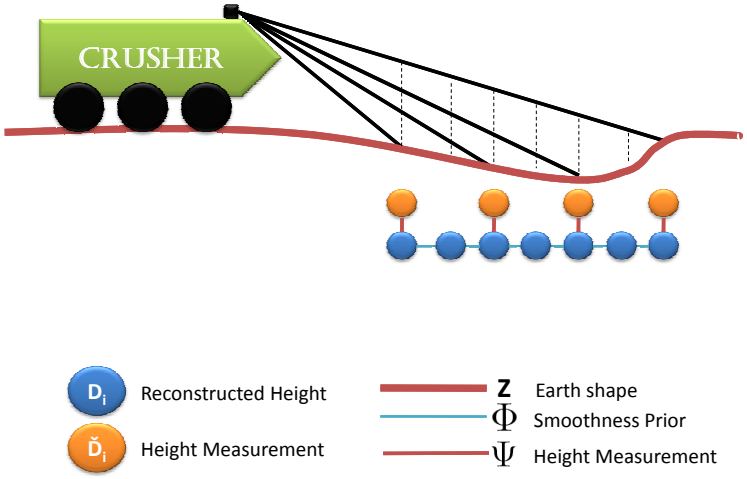

Figure 4: Markov Random Field for Terrain Mapping