Estimating future bitcoin mining income

Thursday, 6th February 2014

Previous posts in series

0. Introduction
Websites such as mining.thegenesisblock.com provide some very handy tools to estimate future mining income, and I use them if I need to do some quick calculations away from home. But if you use these tools without knowing how or what they calculate, you can end up misinterpreting results, using unlikely forecast values, or seeing correlations where there are none.

If you look back at some of the previous "ASIC choices" posts, you'll notice that while I gave details on how to accurately estimate mining earnings, I never provided a nice simple method for anyone to make a forward projection.

In this post I'll do that - show you a simplified way to estimate bitcoin mining income, step-by-step, and using at most a spreadsheet (groan) - no fancy R code necessary.

At the end of this post you will:

1. Know a simple method for estimating future changes in difficulty, and be able to calculate confidence intervals for your forecast.
2. Understand the relationship between the time taken for a retarget to occur and the change in difficulty.
3. Be able to do your own difficulty change calculation.
4. Calculate your expected income per Ghps (ignoring electricity costs).

Well, if I explain it well enough, you'll be able to do the above. If not, feel free to lambast my poor pedagogical skills.

1. Estimating a range of future changes in difficulty
Estimating how much the network mining difficulty will change even in the short term is very hard with any degree of accuracy. My recommendation is to either:

- Guess what the changes will be based on estimated hashrate delivery dates, using information such as this thread on bitcointalk.org.

- Estimate a range of future changes based on past changes.

If you want something that is more likely to be accurate, choose method 1 and do lots of reading, and try to estimate what the difficulty changes might be based on hashes added and dates. It's tricky and complicated.

Forecasting future data from historical data can also be tricky and complicated. I've previously posted forecasts using ARIMA and exponential smoothing, and I don't think those methods are easily spreadsheetable. I was able to do a regression forecast model (with confidence intervals) using a google docs spreadsheet, but it was neither simple nor easy.

Instead I decided on a so-called *naive* method of forecasting. The concept is simple - the next forecast data will be equal to the last one. In this case, the quantity of interest is the change in difficulty rather than the network difficulty itself.

When difficulty is increasing or decreasing at a constant rate, this method has been more reliable than I've expected. When difficulty is increasing or decreasing at an accelerating or decelerating rate (such as now), the naive method will tend to undershoot or overshoot the mark - something to keep in mind when you're planning with your forecast.

How to forecast difficulty using the naive forecast method:

Step 1: Decide how many retargets ahead you want to forecast. We'll call that number *'n'*.

Step 2: Get historical difficulty data (use the spreadsheet below if you like - it's up to date as of today). I only use post-July 2011 data for my difficulty forecasts. Before this there were fewer miners and so a greater variance in the network hashrate and in difficulty, especially as MTGOX came online and as the GPU revolution hit. At the moment there are many miners and while the network hashrate is increasing exponentially, it is not doing so in a discontinuous fashion. Prior to July 2011 there were several abrupt changes in difficulty that simply cannot happen now.

Step 3: Starting from the most recent retarget, calculate the ratio between the first difficulty and the last difficulty for the last *n* retargets. For example, if you want to forecast six retargets ahead:

```
2,621,404,453 / 908,350,862 = 2.88589
```

This means that we are forecasting a 2.9 times increase in difficulty over the next six difficulty changes.

Step 4: Convert the six week forecast to a per retarget figure by taking the nth root of the estimate:

```
 2.88589 ^ (1/6) = 1.1932 (forecast that network difficulty
increases by a factor of 1.1932 each retarget).
```

That's easy enough, so far, and if you like you can skip the next step and go to the next section. However, if you want to learn how to calculate forecast intervals, just stick with it for another step.

Step 5: Calculate forecast intervals. To do this, you'll need to repeat steps *3* and *4* for your historical difficulty data. This is column C in the spreadsheet below. Next, copy the values from column C to column D, increasing the row number by *n* - if you're forecasting six retargets ahead, as in the spreadsheet, C8 is copied to D2, C14 to D8, and so on. The log of column D / column C are the "residuals" of your forecast.

Residuals are required to calculate forecast intervals (residuals are also required to be normally distributed - you'll have to take my word for it that they are). A forecast interval is a range of values that a forecast could be - it's only the mean value that we're forecasting. A 95% confidence interval would mean that, were the forecast to be run concurrently in one hundred different universes, 95% of the time the actual data would be within the forecast interval.

However, as miners. we're not really interested in lower an upper bounds - we just want worst case scenarios. In the spreadsheet I have "75%" and "90%" figures - these are forecasts which will be greater than the actual figure 75% of the time and 90% of the time respectively. If you inspect the cells, you'll see that they are calculated as:

```
75%: exp(log(forecast) + 0.6744898*stdev(residuals))
90%: exp(log(forecast) + 1.644854*stdev(residuals))
```

If the idea of "forecast intervals" is confusing you, just think of these forecasts as your "bad luck" estimate (75%) and your "worse luck" estimate (90%).

unixtime	difficulty		Per six retargets	Per retarget
		forecast=	2.885894	1.1932
		75% =	3.493007	1.23178
		90% =	4.597185	1.289482
1391584456	2,621,404,453	2.885894	3.737339	0.258539
1390570126	2,193,847,870			
1389583220	1,789,546,951			
1388624318	1,418,481,395			
1387617112	1,180,923,195			
1386684686	908,350,862			
1385742648	707,408,283	3.737339	3.979893	0.062881
1384699499	609,482,680			
1383681123	510,929,738			
1382754272	390,928,788			
1381925788	267,731,249			
1381070552	189,281,249			
1380118146	148,819,200	3.979893	2.571867	-0.43662
1379202248	112,628,549			
1378268460	86,933,018			
1377353319	65,750,060			
1376417490	50,810,339			
1375527115	37,392,766			
1374515827	31,256,961	2.571867	2.307771	-0.10835
1373502163	26,162,876			

1372515725	21,335,329			
1371418654	19,339,258			
1370442318	15,605,633			
1369499746	12,153,412			
1368386123	11,187,257	2.307771	1.465909	-0.45381
1367296471	10,076,293			
1366218134	8,974,296			
1365183643	7,673,000			
1364126425	6,695,826			
1363249946	4,847,647			
1362159764	4,367,876	1.465909	1.103304	-0.28417
1361148470	3,651,012			
1360063146	3,275,465			
1358966487	2,968,775			
1357641634	3,249,550			
1356530740	2,979,637			
1355162613	3,370,182	1.103304	1.534587	0.329952
1353928229	3,438,909			
1352743186	3,368,767			
1351556195	3,304,356			
1350428168	3,072,322			
1349226660	3,054,628			
1348092851	2,864,141	1.534587	1.160988	-0.27899
1346955037	2,694,048			
1345859199	2,440,643			
1344772855	2,190,866			
1343647577	2,036,671			
1342537166	1,866,391			
1341401841	1,751,455	1.160988	1.143708	-0.015
1340208964	1,726,567			
1339099525	1,583,178			
1337883029	1,591,075			
1336565313	1,733,208			
1335512370	1,508,590			
1334246689	1,577,913	1.143708	1.09663	-0.04203
1332999707	1,626,553			
1331885394	1,498,294			
1330676736	1,496,979			
1329564255	1,376,302			
1328351561	1,379,647			
1327204504	1,307,728	1.09663	0.666479	-0.49799
1326047176	1,250,758			0.28307
1324925005	1,159,929			
1323718955	1,155,038			
1322576420	1,090,716			
1321253770	1,192,498			
1320032534	1,203,462	0.666479		
1318556675	1,468,195			
1317163624	1,689,334			
1315906316	1,755,425			

1314681303	1,777,774
1313451894	1,805,701
1312186279	1,888,787
1311103389	1,690,896
1309984546	1,563,028

2. Calculate expected earnings per Ghps

Plug your data into a spreadsheet - go ahead, give it a try. Either make your own copy of the spreadsheet, or wait your turn if a bunch of people are using it. Once you've played with it for a bit, scroll down and I'll explain how it works.

Date of retarget	Difficulty	Estimated percentage change next Difficulty	Seconds left until next retarget left	Bitcoin reward per block	Estimated BTC income per Ghps	Estimated cumulative BTC income per Ghps
7/28/2014 0:00:00	4250000000	120.10%	274907	25	0.00037651	0.00037651
7/31/2014 4:21:47	5104250000	120.00%	1008000	25	0.00114950	0.00152601
8/11/2014 20:21:47	6125100000	120.00%	1008000	25	0.00095792	0.00248393
8/23/2014 12:21:47	7350120000	120.00%	1008000	25	0.00079826	0.00328219
9/4/2014 4:21:47	8820144000	120.00%	1008000	25	0.00066522	0.00394741
9/15/2014 20:21:47	10584172800	120.00%	1008000	25	0.00055435	0.00450176
9/27/2014 12:21:47	12701007360	120.00%	1008000	25	0.00046196	0.00496372
10/9/2014 4:21:47	15241208832	120.00%	1008000	25	0.00038497	0.00534868
10/20/2014 20:21:47	18289450598	120.00%	1008000	25	0.00032080	0.00566949
11/1/2014 12:21:47	21947340718	120.00%	1008000	25	0.00026734	0.00593682
11/13/2014 4:21:47	26336808862	120.00%	1008000	25	0.00022278	0.00615960
11/24/2014 20:21:47	31604170634	120.00%	1008000	25	0.00018565	0.00634525
12/6/2014 12:21:47	37925004761	120.00%	1008000	25	0.00015471	0.00649996
12/18/2014 4:21:47	45510005713	120.00%	1008000	25	0.00012892	0.00662889
12/29/2014 20:21:47	54612006856	120.00%	1008000	25	0.00010744	0.00673632
1/10/2015 12:21:47	65534408227	120.00%	1008000	25	0.00008953	0.00682585
1/22/2015 4:21:47	78641289872	120.00%	1008000	25	0.00007461	0.00690046

2/2/2015 20:21:47	94369547846	120.00%	1008000	25	0.00006217	0.00696264
2/14/2015 12:21:47	1.13243E+11	120.00%	1008000	25	0.00005181	0.00701445
2/26/2015 4:21:47	1.35892E+11	120.00%	1008000	25	0.00004318	0.00705762
3/9/2015 20:21:47	1.63071E+11	120.00%	1008000	25	0.00003598	0.00709361
3/21/2015 12:21:47	1.95685E+11	120.00%	1008000	25	0.00002998	0.00712359
4/2/2015 4:21:47	2.34822E+11	120.00%	1008000	25	0.00002499	0.00714858
4/13/2015 20:21:47	2.81786E+11	120.00%	1008000	25	0.00002082	0.00716940

3. Understanding the income spreadsheet

The percentage change in difficulty is the key to understanding the spreadsheet. If you can forecast difficulty changes, you can forecast not just the next changes in difficulty, but how long each set of 2016 blocks will take.

Column B: Difficulty = last difficulty * percentage change forecast

Column D: Time until next difficulty = 14 days / percentage change forecast

Column F: Income per ghps = 10^9 / 2^32 * time until next retarget / difficulty * reward per block

Column G: Cumulative income per Ghps: cumulative sum of column F

4. Summary

Calculating an estimate for your earnings is not hard if you have the right spreadsheet, but the real skill lies in forecasting difficulty changes. If you don't mind the lack of forecast intervals, then you could try plotting the rate of acceleration/deceleration of difficulty (the change in the rate of change of difficulty) and forecast using a trend-line of some sort.

Alternatively you could just take your historical income, and bypass the whole step function funkiness that comes with difficulty forecasts - which is what I'll do next post - except without a spreadsheet (oh happy day!).