Effective business simulation: store-by-store income statements

(Henry T.H. Tu, 13-nov-2019)

We will simulate the franchise with many stores and each store has its own record of earnings. We will assume that each store has its own capacity of revenue generating and cost occurring. We assume linear growth model for each store. However, the linear coefficients are generated randomly. Cost fluctuation can be simulated in month and store level.

3.1. Stores and their capabilities

We will generate a list of stores with linear growth model over months. Each store will have revenue rk computed by rk = ak*t + bk, where (ak, bk) are two linear coefficients and t is the month. The store code column st_code is generated by picking first 7 digits of the MD5 hashed code. The base revenue (bk) and the scale factor (ak) are generated by first 7 digits of MD5 code divided by 10^7 as in table 1A.

	st_code	revenue_base	revenue_scale
1	st5001234	0.978595	0.887946
2	st1063141	0.184222	0.32669
3	st8805521	0.9272	0.154839
4	st3287523	0.836446	0.562666
5	st9383355	0.002003	0.61142
6	st6058807	0.665612	0.888074
7	st6848714	0.820366	0.139165
8	st6335186	0.25748	0.540481
9	st3293141	0.235614	0.126021
10	st2165420	0.462991	0.360291
11	st2071901	0.868066	0.119431
12	st0365824	0.382462	0.844008
13	st8807146	0.764952	0.717933
14	st3822024	0.249985	0.819235
15	st4941203	0.658129	0.852246
16	st0788077	0.849378	0.435685
17	st1380179	0.71903	0.002635

```
------- st_code
"st" + left(regex_replace(
MD5_ASCII([RowCount] + "/code")
, "\D+", ""), 7)

------- revenue_base
tonumber(left(regex_replace(
MD5_ASCII([RowCount] + "/rev-base")
, "\D+", ""), 7))/pow(10, 7)

------- revenue_scale
tonumber(left(regex_replace(
MD5_ASCII([RowCount] + "/rev-scale")
, "\D+", ""), 7))/pow(10, 7)
```

Table 3.1A. Stores and their capabilities

3.2. Historical period or time axis

Let us add time axis for each store in order to prepare for the cost and revenue over time. We multiply each store with all year and month of the historical period of FY15 to FY19. We will remove few months in the beginning and few months at the end of the historical period in order to make it more realistic. We never have full 12 months for the first year and the last year in the historical period. The similar results can be found in Table 2A

	year	month	time_ym	time_rel
1	2016	4	2016-04-01	0
2	2016	5	2016-05-01	1
3	2016	6	2016-06-01	2
4	2016	7	2016-07-01	3
5	2016	8	2016-08-01	4
6	2016	9	2016-09-01	5
7	2016	10	2016-10-01	6
8	2016	11	2016-11-01	7
9	2016	12	2016-12-01	8
10	2017	1	2017-01-01	9
11	2017	2	2017-02-01	10
12	2017	3	2017-03-01	11
13	2017	4	2017-04-01	12
14	2017	5	2017-05-01	13
15	2017	6	2017-06-01	14
16	2017	7	2017-07-01	15
17	2017	8	2017-08-01	16

------ year
Increasing sequence 2016 to 2019

------ month
Increasing sequence 1 to 12

------ time_ym
The combination of year and month

------ time_rel
The increasing sequence from 0 to the end of the month, this is used in the revenue function

Table 3.2A. Historical period

3.3. Linearly-growing revenue and cost fluctuation model

We assume that the revenue grows linearly over time as we will have more and more customers and orders. Therefore, each revenue line item is the linear function of month. The shape of the function depends on store linear coefficients. Revenue fluctuation is added in order to make the line items more realistic.

We will assume that cost is distributed around revenue in particular month. The base cost is around 80% revenue over the year, meaning that the business does not conduct cost optimization nor business model innovation. Table 3A shows the revenue and cost model

	st_code	time_ym	time_rel	year	month	st_revenue	st_cost
1	st5001234	2016-04-01	0	2016	4	497.577933	403.069567
2	st5001234	2016-05-01	1	2016	5	431.360467	358.652051
3	st5001234	2016-06-01	2	2016	6	524.256321	499.264649
4	st5001234	2016-07-01	3	2016	7	585.126978	412.731917
5	st5001234	2016-08-01	4	2016	8	431.047766	404.023834
6	st5001234	2016-09-01	5	2016	9	482.898524	406.353769
7	st5001234	2016-10-01	6	2016	10	676.964079	586.342595
8	st5001234	2016-11-01	7	2016	11	672.539782	553.466518
9	st5001234	2016-12-01	8	2016	12	516.306341	354.64689
10	st5001234	2017-01-01	9	2017	1	694.597035	474.111841
11	st5001234	2017-02-01	10	2017	2	559.552977	435.897326
12	st5001234	2017-03-01	11	2017	3	421.675843	305.0457
13	st5001234	2017-04-01	12	2017	4	577.714639	383.994064
14	st5001234	2017-05-01	13	2017	5	666.536789	543.980595
15	st5001234	2017-06-01	14	2017	6	601.504999	516.056555
16	st5001234	2017-07-01	15	2017	7	490.370119	456.409891
17	st5001234	2017-08-01	16	2017	8	658.916792	527.346787

```
------ revenue
([revenue_base]*537
+ 2.5*[revenue_scale]*[time_rel])

* (1 + 0.57*(
-0.5 + tonumber(left(regex_replace(
MD5_ASCII([time_ym] + "/rev")
, "\D+", ""), 7))/pow(10, 7)
))

------ cost
[revenue] * (0.8 + 0.31*(
-0.5 + tonumber(left(regex_replace(
MD5_ASCII([time_ym] + "/cost")
, "\D+", ""), 7))/pow(10, 7)
))
```

Table 3.3A. Revenue and cost

Figure 3B shows the profile of the data. We can display the matrix of store by time for revenue and cost to see store-by-store line items. We can also draw bar chart to see the trends of the data

	2016		2017		2018		2019	
st_code	revenue	cost	revenue	cost	revenue	cost	revenue	cost
st0246412	579.9	478.0	1,046.1	807.0	1,409.9	1,136.1	1,186.3	952.9
st0253234	2,552.7	2,108.3	3,573.5	2,753.3	3,826.5	3,073.0	2,740.5	2,196.3
st0278886	294.1	242.1	610.5	471.2	899.4	725.5	793.9	638.1
st0323421	3,730.7	3,081.4	5,192.0	4,000.1	5,521.6	4,433.8	3,931.3	3,150.4
st0338909	2,977.9	2,459.5	4,191.8	3,229.7	4,516.9	3,627.9	3,252.3	2,606.7
st0342557	3,009.3	2,485.1	4,304.0	3,316.5	4,721.5	3,793.3	3,450.2	2,765.9
st0352265	1,254.6	1,035.2	2,022.7	1,559.5	2,495.2	2,008.1	1,987.7	1,595.4
st0365824	1,929.8	1,593.0	2,948.4	2,272.7	3,462.1	2,784.3	2,665.4	2,138.3
st0380194	3,292.2	2,719.2	4,606.2	3,548.9	4,928.9	3,958.3	3,528.0	2,827.4
st0396228	785.4	648.0	1,263.4	974.1	1,555.5	1,251.8	1,237.5	993.3
st0403850	4,722.6	3,900.7	6,565.2	5,058.0	6,972.9	5,599.1	4,959.2	3,974.0
st0434522	1,723.9	1,423.5	2,487.1	1,916.5	2,754.4	2,213.2	2,028.2	1,626.2
st0450029	2,659.7	2,196.8	3,692.3	2,844.7	3,915.3	3,143.9	2,780.8	2,228.3
st0450796	224.5	184.7	489.2	377.7	740.1	597.2	661.9	532.0
st0486258	1,108.0	914.7	1,666.0	1,284.1	1,925.9	1,548.6	1,465.9	1,175.8

st_code	time_ym	st_re...	st_cost
st5001234	01-Apr-2016	497.6	403.1
st5001234	01-May-2...	431.4	358.7
st5001234	01-Jun-2016	524.3	499.3
st5001234	01-Jul-2016	585.1	412.7
st5001234	01-Aug-20...	431.0	404.0
st5001234	01-Sep-20...	482.9	406.4
st5001234	01-Oct-2016	677.0	586.3
st5001234	01-Nov-20...	672.5	553.5
st5001234	01-Dec-20...	516.3	354.6
st5001234	01-Jan-2017	694.6	474.1
st5001234	01-Feb-2017	559.6	435.9
st5001234	01-Mar-2017	421.7	305.0
st5001234	01-Apr-2017	577.7	384.0
st5001234	01-May-2...	666.5	544.0
st5001234	01-Jun-2017	601.5	516.1
st5001234	01-Jul-2017	490.4	456.4

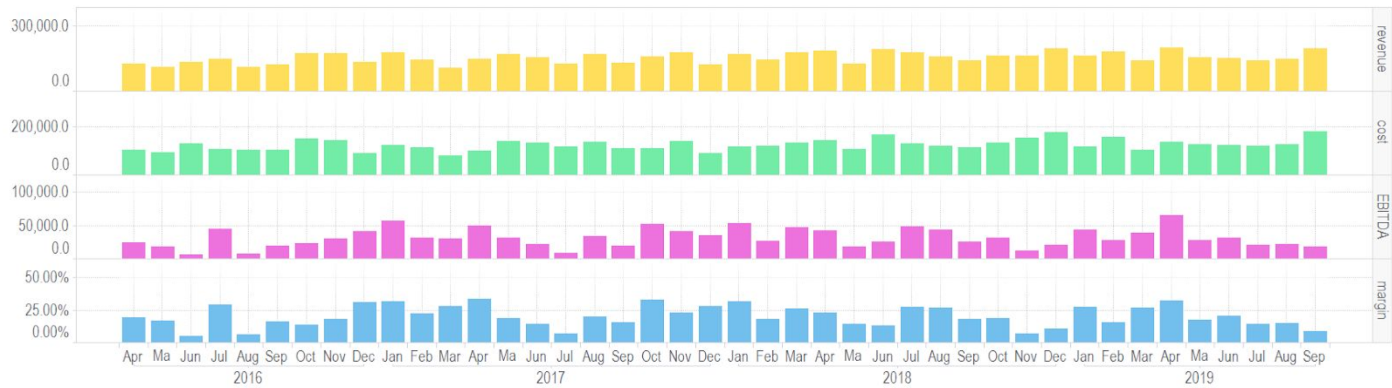

Table 3.3B. Trends and rows

3.4. Summary

Each store can have revenue line item and cost line item to show its performance over time. Revenue line item can be modeled as a linear function of time, where linear coefficients show the revenue-generating capacity of the store.

Cost can be generated in the same manner. Or we just assume the stochastic affine transformation of revenue. Without cost optimization or business model renovation, the cost/revenue rate is constant over the years.