**Short Summary About Current Design & Results**

**Notations:**

**T**: the length of Time Epoch

**R**: the bouncing range of mean valuation per time epochs

**N**: customer number

**sig**: the sigma in Gaussian function. Smaller the sigma is, closer valuations are.

**P**: the price of smart token at the beginning of every time slot

**1. Bancor Market**

The whole simulating time is comprised of **1000** time slots. In every time slot, Bancor Market processes orders launched by **N** customers, and in the meanwhile alters the price of smart token.

**Valuation Making:**

To begin with, based on the price of smart token, we generate the market valuation of smart token per time epoch as **Vtp**, by which the mean valuations **Vt**, i.e., blue point in below figures can be generated. After we get Vt, every customer's valuation is generated based on **Vt** (mu in Gaussian) and **sig** (sigma in Gaussian).

For instance, we first generate **Vtp** = 20 ETH, which indicates in this time epoch, customers generally regard the smart tokens' value as 20 ETH; while in different time slots of this time slot, the specific valuation **Vt** can be 19.2, 21.4, 20.8, 19.5 and so on. Then, in every time slot, **N** customers' valuations are made by **Vt**, such as 19.2, 21.4, 20.8, and etc.

The reason we use **P,** i.e.,price of smart token at the beginning of every time slot to generate **Vtp** is that we try to simulate the equal chance for **market craze** and **market crisis**, i.e., 50% probability of **Vtp** < **P -- Vtp** selected from **(P/R, P),** and 50% probability of **Vtp** > **P -- Vtp** selected from **(P, P\*R).**

Figures below shows the **Vt** in Bancor Market.

../../Bancor-Simulator/Figures/Bancor/Valuation-TE-200BG-5.0CN-1000Sig-1.0Seed-1.pdf../../Bancor-Simulator/Figures/Bancor/Valuation-TE-200BG-2.0CN-1000Sig-1.0Seed-0.pdf../../Bancor-Simulator/Figures/Bancor/Valuation-TE-50BG-2.0CN-1000Sig-1.0Seed-1.pdf

(a) T = 200, R = 5.0 (b)T = 200, R = 2.0 (c)T = 50, R = 2.0

As you can see, by Figure (b) and (c), larger the **T** is, i.e., longer the length of time epoch is, less "valuation stairs" in Bancor Market. By Figure (a) and (b), larger the **R** is, i.e., larger the valuation's bouncing range, larger the fluctuating range of valuations.

Codes about valuation generation:

'''

In the loop of "for j in range(TimeSlotNum):", we start our simulation of Bancor Market.

To begin with, based on the price of smart token, i.e., P,

we generate the market valuation of smart token per time epoch as Vtp, by which the mean valuations Vt can be generated.

After we get Vt, every customer's valuation is generated based on Vt and sig.

'''

Vt\_list = []

for j in range(TimeSlotNum):

  sychronizeMarket(MyBancorMarket)

  '''

  First of all, we randomize mean valuations per time epochs, and save in Vt\_list list.

  For instance, 0-49 time slot comprise the first time epoch.

  If the Vtp is 20 ETH, in 0 - 49 time slots,

      customers generate their orders based on 19.4 ETH, 21.2 ETH ...

  '''

  P = KennyCoin.getPrice()

  if j % T == 0:

      # reset the Vt\_list

      Vt\_list = []

      if bool(random.getrandbits(1)):

          Vtp = random.uniform(P/R, P)

      else:

          Vtp = random.uniform(P, P\*R)

      Vt\_list = np.random.normal(Vtp, 1, T)

​

  Vt = Vt\_list[j % T]

  custValuation\_list = np.random.normal(Vt, sig, N)

  ...

**Transaction Generating:**

After customers making their valuations of smart token, they will launch transaction orders with several stipulations:

1. If customers' valuation of smart token is larger than **smart token's price at the beginning of time slot,** customers will launch transaction orders to buy the smart token; otherwise, sell orders will be generated.

2. If customers do not have reserve tokens in hand, they will not launch orders to buy smart token, though their valuations might be higher than **smart token's price at the beginning of time slot**. Similarly, if customers do not have smart token to sell, they cannot generate sell orders.

3. Customers will make valuations and launch orders in every time slot; while at one time slot, single customer can only generate one order or does not generate, which indicates with **N** customers, at every time slot **<=N** transaction orders will be created. Therefore, 1000 time slots, totally **<=1000N** transaction orders will be made.

4. Customer uses all of their reserve tokens or smart tokens to buy or to sell if they have money in hand.

\* We also call 4 as all-in policy, which is implemented for simplicity.

The code for customers' transaction generation is presented as below:

# self represents a customer (customer class). we name him/her as XXX

if self.\_valuation > marketPrice and self.\_reserveBalance > 0:

  # XXX issues a buy order

  self.\_market.buy(self, self.\_reserveBalance) # all-in policy

elif self.\_valuation < marketPrice and self.\_tokenBalance> 0:

  # XXX issue a sell order

  self.\_market.sell(self, self.\_tokenBalance)

else:

  # nothing to do

  pass

After customers generating their transaction orders by valuations and money they hold, they wait to see whether their transactions could be handled by market.

**Transaction Processing:**

Bancor Market processes customers' transaction orders **one by one**. In the meanwhile, it updates the price of smart token. Here, one by one means when dealing with one of customers' transaction orders, if the current price of smart token does not meet this customer's valuation, the market will announce the customer to cancel this order and skip this transaction order to try to deal with the next customer's order.

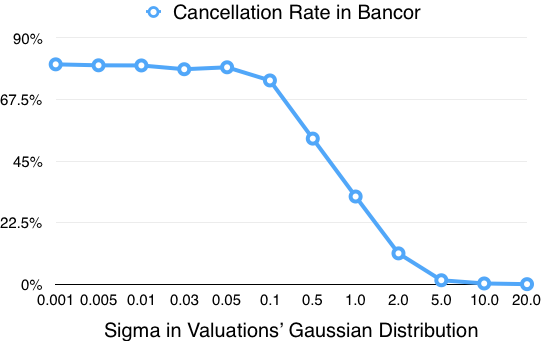
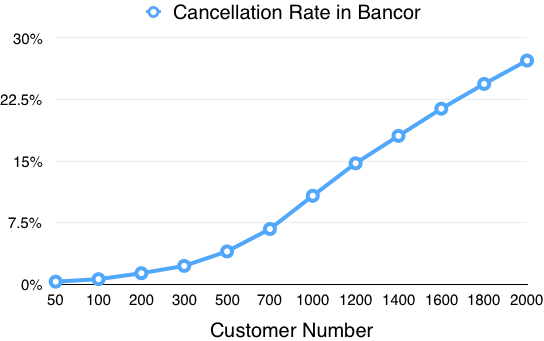
It might sound complicated, but the core idea is to simulate the limited market order in real life -- customers generate valuations of product first, and then accord to the product's price in market to decide to whether finish the transaction. In Bancor Market, customers, based on the **P**, make their valuations while decide to finish the transaction by comparing the real-time price of smart token and valuations. (If the comparison tells customer he should not finish his order, this order will be canceled.)

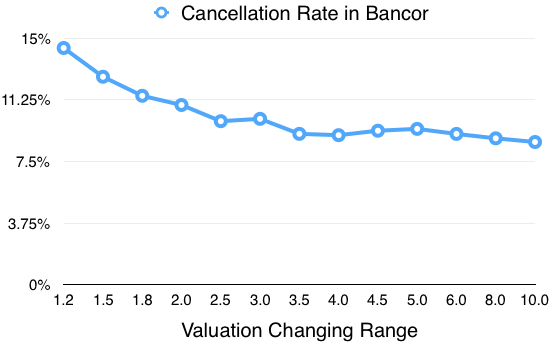
Therefore, we record the price fluctuation of smart token as an indicator of Bancor Market's performance by figure 1.

../../Oct21th-allin/Bancor/Sig/Price-TE-50BG-2.0CN-1000Sig-2.0Seed-0.pdf

Fig 1: The price change in Bancor Market when T = 50, R = 2.0, N=1000, sig=2.0.

Also, we observe the cancellation rate of transaction orders in Bancor, according to limited order. We use control experiments to see the influence of **T, N, sig** and **R** on cancellation rate respectively. (Standard setting we set is: T=50, N=1000, sig=2.0, R=2.0)

**** ****

**With smaller sigma, the transactions' cancellation rate is much higher,**

which indicates more tightly customers make their valuations, more likely they need to cancel their transaction orders.

To view mathematical proof, see slides from annex 1 to annex 9.

**With larger customer number, the transactions' cancellation rate is much higher**.

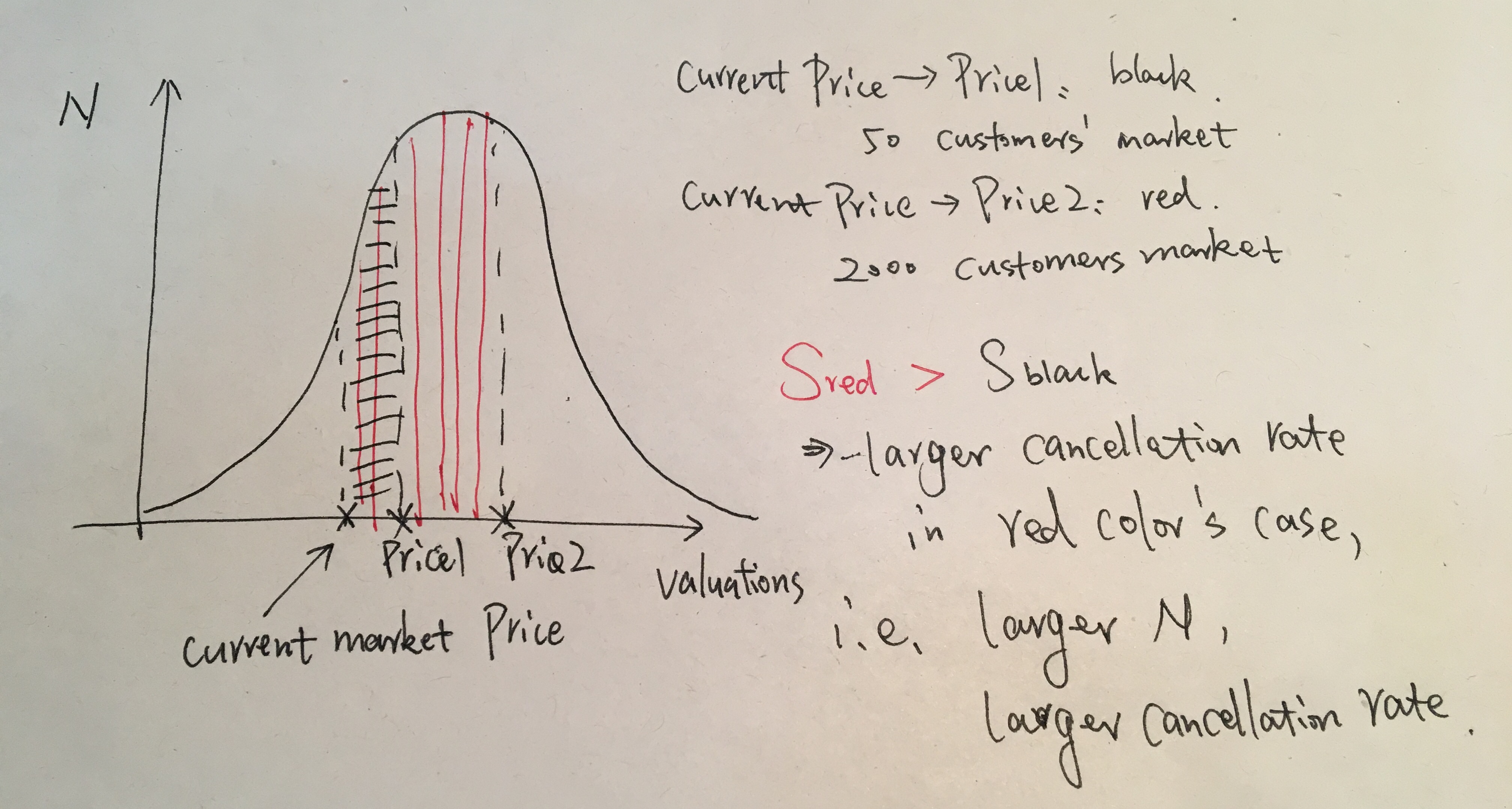
The reason is that we set every customer initially have 200 reserve tokens and 200 smart tokens.

Assuming customers all buy smart tokens:

In 50 customers’ market, 200\*50 reserve tokens are converted to smart tokens.

In 2000 customers’ market, 200\*2000 reserve tokens are converted to smart tokens.

Therefore, the price of smart token fluctuates more fiercely in 2000 customers’ market.



**2. Classic Market**

The whole simulating time is comprised of **1000** time slots. In each time slot, Classic Market processes the orders launched by **N** customers by managing the order list in the market.

**Valuation Making:**

The valuation making in Classic Market is similar with Bancor Market.

However, since smart tokens in classic market are not created or destroyed, the price of the smart token, i.e., **P** is a constant.

**Transaction Generating:**

1~4 are similar with Bancor Market.

The only difference is that in Classic Market, the customer will not launch new order if his order has not been fulfilled.

For instance, in a certain time slot, a customer launches a sell order at valuation 5 ETH to sell all of his 200 smart tokens. However, in the next time slot, he finds that only 120 smart tokens have been sold. Therefore, he will not launch new order and continue to wait for his remaining 80 smart tokens being sold at valuation 5 ETH.

In the end of simulation, i.e., 1000 time slots have passed, if this transaction order is still unfinished in market, we say this order should be canceled.

**Transaction Processing:**

Classic Market manages an order list to process all transaction orders launched by customers.

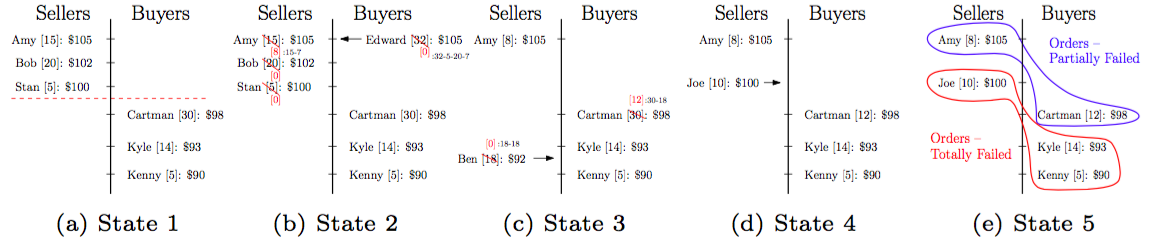
In short, all transaction orders from customers will be separated into two sub-list, named as sell list and buy list. In each list, orders will be sorted by the valuation of these orders. Then the market processes these orders by methods shown in Figure 3.

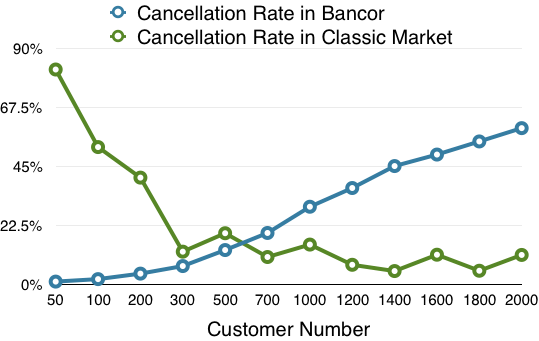
Figure 3: Five subfigures respectively correspond to five different states.

Both Partially Failed orders and Totally Failed orders will be remained in order list and expect in the time slot they can be finished.

However, if these orders can never be finished -- after 1000 time slot, they still are remained in the market, we then say these orders should be canceled.

Thus, we calculate the **cancellation rate** by: # of orders remained in market / # of all launched orders.

**3. Comparison Between Bancor and Classic's Cancellation Rate**



The reason why the cancellation rate is high when customers' number is small:

For instance, in a market craze, there are 0.5 probability for customers launching sell orders by Gaussian Distribution.

However, when customers' number is really small, there might be no one who launch sell orders. Thus, those buyers will unfortunately be stuck in market, as they have no matched sell orders.

In the case of big customers' number, there will always be a relatively stable number of customers in market launching sell orders.

**The review of Cancellation rate:**

In Bancor:

Why: The price of smart token cannot meet customer's valuation in order.

Cal: cancellation rate = # of all canceled orders / # of all launched orders

In Classic:

Why: Customer's order cannot be finished before the end of simulation.

Cal: cancellation rate = # of orders unfinished / # of all launched orders.

By Figure 5, we can know the cancellation rate in Bancor Market is much higher than in Classic Market.

**4. Summary**

1. By Figure 1, we know, if customers keep investing money in Bancor Market (never run out of reserve tokens), the market craze in time epoch can lead the price of smart token bouncing around and finally to extremely high.

2. By Figure 2, it can be viewed that the order's cancellation rate in Bancor Market can be quite high -- reaching beyond 10% in several parameter settings, especially when customers' valuations are made tightly, i.e., sigma is small.

3. The low classic cancellation rate in Figure 4 indicates the "co-incidence of double wants" might not be a problem in Classic Market.

4. By Figure 5, we know the cancellation rate in Bancor Market is always much higher than in Classic Market.

Recently, I have cleaned my code and added many comments. They are presented on:

https://github.com/Ohyoukillkenny/Bancor-Simulator

Unfortunately, up to now, what we have fully discussed in Bancor market is based on **limited order**.

Now, I am still working on modifying our simulating model in order to make it feasible in more general cases.