**Note 1**

* In the present lesson, we will introduce a memory effect into the dealer model presented in the previous lesson to obtain a more realistic description for the stock price returns.

**Note 2**

* As usual, we begin by importing the necessary numerical and graphics libraries.
* We will compare the results of the dealer model with real stock data, so we again need to import the pandas and datareader libraries.

**Note 3**

* Here you will recognize the helper functions used in the previous two lessons to analyze the stock prices.
* Also, we retrieve the daily price for the Toyota stock for the previous 20 years and compute the one-day price return.

**Note 4**

* Let us briefly recall the dealer model introduced in the previous lesson.
* We have two dealers buying and selling a stock with each other.
* We characterize their positions through their mid-price, which is the average of their bid and ask prices.
* The price of each dealer follows an independent 1D random walk in time, until the transaction condition, Eq.(L1), is satisfied.
* At this point, they exchange one unit of stock at a market price given by the average of their mid-prices, as defined in Eq.(L2)
* Their mid-prices are then reset to this new Market price, and they start a new random walk.
* We saw that we could obtain realistic looking stock price dynamics.
* Unfortunately, upon closer inspection we realized that the distribution of the price returns, given in Eq.(L3), followed an exponential decay, instead of the universal power-law behavior of real stocks.
* The main problem with the simple model we used is the fact that the price dynamics is purely random, with no memory.
* In real situations, this is not the case, we know how a given stock has behaved in the past, and this biases our prediction for what it will do in the future.

**Note 5**

* In their original paper, Yamada and coworkers recognized the problem with the basic model and proposed to include a "trend-following" effect as a solution.
* Thus, when deciding on how to update their mid-prices, dealers use their knowledge of the previous price increases.
* One response can be to follow the recent trend, and push the price upward if it was recently increasing, or downward if it was decreasing. These dealers are called trend-followers.
* The other possibility is that the dealers go against the recent history, increasing a price that is going down, or decreasing a price that was increasing. These dealers are called contrarians.
* This "trend-following" behavior can be easily incorporated into the random-walk model we are considering by adding a memory term into the dynamical equations. This is represented in Eq.(L4).
* As you can see, the only difference with the original model is the addition of a term proportional to a running average of the previous changes in market price. The constant of proportionality *d*, will determine whether the dealers are trend-followers or contrarians, and how strong this effect will be.
* The running average *<Delta P>\_M* is a weighted average of the previous *M* price changes.
* This weighted average is computed over the market price, and as such will only change when a transaction occurs. During the random-walk process this is a constant that introduces a drift into the trajectories.
* In what follows, we will refer to this model as model 2, and to the original model with no trend-following dynamics as model 1.
* In particular, notice that if either *d*or *M* are equal to zero, then model 2 is reduced to model 1.

**Note 6**

* As before, we can rewrite these two random walk processes as a single 2D random walk, by changing variables from p1 and p2 to the price difference D and average A.
* The dynamics for D are exactly the same as before. This is because *d* is assumed to be a constant, such that the trend-following behavior is exactly the same for both dealers and cancels out when taking the difference in price.
* The dynamical equations for A, on the other hand, acquire the extra drift or memory term.
* The transaction criterion is unchanged, so we still have two absorbing boundaries at D=+/- L.

**Note 7**

* Let us define a function that performs the random walk process for a single trade.
* Compared to the example for model 1, we have two new parameters, *d* and *M*.
* Recall that *d* determines whether the dealers are trend-followers or contrarians and *M* is the number of points that are used in the running average of the previous market price increases, as defined in Eq. (L6).
* The code to reproduce the RW is exactly the same one we used before, we have just one extra line to add the term proportional to *d* times the running average to the price.
* However, since this running average is a constant during one random-walk, we do not need to calculate it, we can simply specify it as one of the function parameters. We are free to set its value arbitrarily.
* Finally, remember that this function will return the trajectory in the 2D (A,D) space, not the values of the individual mid-prices for the two dealers.

**Note 8**

* Here we perform two random walks, one using model 1, and the other using model 2. For model 2 we consider the dealers are trend-followers, with *d = 1.25*, and we set *M=1*, so that dealers only consider the latest price increase to calculate the current trend.
* Apart from this, all other parameters are the same ones used in the previous lesson for model 1. In particular, we also take care to use the same random number sequence, by explicitly setting the seed before each RW. This will ensure that the sequence of up/down left/right steps are exactly he same.
* We can use the same code for performing both simulations. All we have to change is the value of the running average. For model 1 we set it to zero, as this term does not appear, and for model 2 set it to an arbitrary value of 0.003. This means that the dealers are under the assumption that the price previously increased by this amount.
* The trajectory data for model 1 is given in red, that of model 2 in blue.
* As you can see, both trajectories start from the same point, marked as a square, and take exactly the same number of steps to reach the top boundary.
* The reason for this is straightforward: the dynamics of *D* is the same in both model 1 and model 2. Thus, the transaction interval will be the same, since it is *D* that determines the transaction condition.
* The difference then, lies in the motion along the horizontal axis *A*. Here, the extra memory or drift term adds a "flow" in the positive *A* direction, essentially stretching the trajectories.
* In the end, for this simulation, it results in the dealers making a transaction that increases the stock-price, whereas model 1 gave a decrease of the price.
* Since we simulated trend-followers under the assumption that the market price previously increased, the result is probably not at all surprising.

**Note 9**

* Now let us perform a simulation of model 2 over many transactions.
* For this, we modify the code used to simulate model 1, in order to include the trend-following term in the price dynamics.
* Comparing with the model 1 code of the previous lesson, we have added a helper function 'avgprice' that will calculate the running average of the price increases for the previous M ticks, as given in Eq. (L6).
* To simplify the code we have also added an additional array 'dmktprice', that will contain the trajectory data of the changes in market price.
* While this information can be obtained from the market price information itself, it will simplify the code to have it as a separate variable.
* The main simulation code is exactly the same as before, except for the price update, which now includes a term proportional to the running average.
* At the end of each random walk, when the transaction is executed, we save the data for the change in market price and we update the running average.
* To avoid waiting for the three or four minutes it takes for the simulation to finish, we have previously run the code and saved the results to a file for the analysis that follows.

**Note 10**

* We load the saved data for model 2 as well as that of model 1, which was presented in the previous lesson.
* As before, we are mainly interested in the price return *G\_1*and the time interval between two transactions, and we compute them using the helper functions defined at the beginning.
* The three data sets, corresponding to the price, the price return, and the time intervals are plotted here for both models.
* The data for the new model 2 is plotted in red, that of model 1 is in blue.
* Looking at the evolution of the price in time, we can start to identify the trend-following behavior of the dealers. This is reflected in the regions of sustained increase or decrease in the price, which can last over several hundreds of ticks.
* From the price returns, we see that much larger returns are now possible. If model 1 could show returns of 5 times the standard deviation, with model 2 we have returns that are 15 times the standard deviation. If you recall the analysis we performed on real stock data you will understand how important this is.
* Finally, it will not surprise you to learn that the transaction interval is exactly the same for both models. We have already explained this when we analyzed the 2D random walk corresponding to this process.

**Note 11**

* Now, let us check whether this new dealer model, with trend-following behavior, gives a better prediction for the distribution of the price returns.
* For this, we reload the data for the Toyota stock and plot the probability distribution function for both data sets.
* The simulation results are represented in orange and the Toyota data in blue. As a guide to the eye, we have also plotted an exponential distribution and a power law, with exponent of −3.
* The previous model 1, resulted in an exponential distribution.
* We see that the new model 2 indeed captures the behavior of real stock-market data.
* I will stress that this is not just a happy accident from choosing this particular stock. Recall that we analyzed the returns for several different Japanese and American stocks, as well as two market indexes, and they all showed the same power law decay.
* This behavior is universal.
* Thus, simply adding a memory term into the dynamical equations has allowed us to reproduce the non-trivial behavior of real market data, and this with a model that only has two dealers!
* You might remark that this comparison is not fair. The returns from the simulation are over tick time, whereas those for the real data are computed across a trading day. However, this behavior is quite robust to the change in time scale, as you can easily check for yourselves.

**Note 12**

* To end this lesson, I want to give you a better idea of how this trend-following behavior affects the dynamics of the stock price. For example, what is the difference between a trend-follower and a contrarian? Can just seeing how the stock is moving identify them?
* For this, we will slightly modify our simulation code and consider that the *d* value is changing in time.
* For the first 1000 ticks or transactions, we will set d<0, which gives contrarian dealers.
* For the second 1000 ticks, we set d=0, which reduces to model 1, and gives simple random walk behavior for the price changes.
* For the third 1000 ticks, we set d>0, which gives trend-following dealers.
* Finally, for the remaining ticks, we will set d depending on the current running average.
* If the running average is positive, which means the stock was increasing, we set d>0 to model trend-followers.
* If the running average is negative, which means the stock was decreasing, we set d<0 to model contrarians.
* To accomplish all this, we define an additional function that will return this time varying d value, given the current tick time and running average. This is the 'dtime' function shown here
* We have to be careful to make sure 'd' is recomputed at every tick, but other than that the code is unchanged.
* In contrast to the previous cases, here we will take *M =20*for the calculation of the running average. This means dealers take into account the history of the previous 20 ticks when determining the current trend.

**Note 13**

* Let us analyze the simulation we previously described, with the trend-following nature of the dealers changing in time.
* As before, we do not evaluate this in real-time, because it would take too long, but load the previously saved simulation data.
* The top plot shows the evolution of the price as a function of tick time, while the bottom plot shows the price return *G\_1*. The data are color-coded depending on the value of *d* used.
* In the red region, the dealers are contrarian, resulting in a stable price.
* In the blue region, we turn of any trend-following behavior and the changes in price are given by a random walk.
* In the purple region, the dealers are trend-followers, which results in a highly unstable price variation over time.
* Finally, in the gray region, the instantaneous *d* value depends on the running average at that point. The dealers react as trend-followers when presented with an increasing price, whereas they are contrarians if the price is decreasing. This will results in an almost linear increase in the price at long times.
* A superficial glance at the returns seems to indicate that nothing really changes.
* However, we know that the distributions are not the same and clear evidence of time correlations can be observed in the data. This is not surprising, since the dynamics over regions with non-zero *d* values contain memory effects caused by the trend-following/contrarian behavior of the dealers.