# Техническое задание

на разработку версии 0.1 алгоритма биржевой торговли. Рабочее название текущей модификации алгоритма: ‘alg0’.

## Общее описание

Работа алгоритма основана на принципе регулирования и принятия решения в каждый момент времени на основе текущей ситуации на рынке. Решение принимается конечным автоматом с тремя состояниями: купить, продать, бездействовать.

Алгоритм не имеет дело с прогнозированием цены и работает только на основании исторических данных, не пытаясь предсказать поведение рынка.

## Содержание алгоритма

Алгоритм работает пошагово, в цикле. На ***i***-том шаге он берет на входе с значения цены ***p*** и тренда ***mu***, и по ним выдает решение (Buy/Sell/Nothing) и значения величин , (прибыль в моменте), (инвестиция/изменение депозита в моменте). Значение тренда mu определяется как первая разность массива цены:

Кумулятивные величины депозита и прибыли могут быть посчитаны суммированием моментных величин по всем сделкам. Значение ***K*** всегда ограничено,

1. На нулевом шаге величины задаются как начальные условия, они необходимы для корректной работы всех задействованных реккурентных формул. Начальные данные:
2. Также требуется задать мгновенную инвестицию на первом шаге .

Т.е. на каждом шаге работы алгоритма (c номером i) мы по значениям на входе , вычисляем значения на выходе , .

1. Начиная с первого шага (***i***=1), сначала делаем проверку. Если рынок стагнирует, ничего не делаем: если , где =0.1. (настроечное значение, может меняться), то  
   , , а решение – Nothing (ничего не делаем). Иначе выполняются пункты 3-6.
2. Берем значение и постоянный параметр C этими вводными находим минимум функции

где sign(x) – функция знака аргумента.   
sign(x) = 1, если x>0, 0, если х=0 и -1, если x<0.

Функция зависит от одной переменной , минимум ищем на отрезке [-10, 10]. Функция является строго монотонной (либо возрастающей, либо убывающей) на этом отрезке, поэтому ее минимум будет достигаться на границах отрезка: либо при , либо при . Т.е. нужно просто выбрать либо 10, либо -10 так, чтобы было наименьшим.

1. Прибыль в моменте считается по формуле . Если найденная прибыль получается , (настроечное значение, может меняться), то сбрасываем найденные значения: , и устанавливаем т.к. мы близко к 0 по прибыли или имеем отрицательную прибыль – страхуемся и не делаем никаких движений. Решение Nothing.
2. Если прибыль оказывается больше допустимого нижнего порога (п.4), то выбираем направление сделки по знаку найденного : если , то покупаем (решение Buy), а если , то продаем (решение Sell).
3. Считаем инвестицию в моменте по формуле:  
   Теперь мы получили значения , .
4. Переходим к шагу 3, увеличив счетчик ***i*** на единицу.

### Настроечные параметры

– ограничение для абсолютной величины тренда .

– ограничение минимальной прибыли.

## Тестирование

Тестирование производится на пяти датасетах: модельные данные в виде ряда ARMA(1, 4) и данных четырех исторических биржевых дней USD-RUB 14, 16, 17, 21 февраля 2022 года.

### Построение ARMA(1, 4)-ряда

Скрипт на Python для построения ряда:

n\_max=100  
a = [10., 8., 6., 2., .1] # Параметры   
d = [1., .6, .4, .1, .06] # Параметры   
p = [-1., 3., .1, -.8, .5] # Фиктивные (до-исторические) значения цены для формирования данных   
  
for i in range(5, n\_max+5):  
 p\_next = a[0]\*p[4] + a[1]\*p[3] + a[2]\*p[2] + a[3]\*p[1] + a[4]\*p[0] + .1\*p[i-1]  
 xi = np.random.normal(0., 10., 5) # Массив из 5 случайных величин, каждая распределена как  
 # N(0., 10.) – нормальное распределение, мат. ожидание 0.,  
 # дисперсия 10  
 p\_next += sum([d[i]\*xi[i] for i in range(5)]) # Считаем следующий член ряда  
 p.append(p\_next) # Добавляем его в ряд  
p\_arr\_arma = np.array(p[5:]) # Отрезаем первые 5 элементов (до-исторические   
 # данные)

mu\_arr\_arma = np.diff(p\_arr\_arma) # Дифференцируем ряд p\_arr\_arma (создаем   
# новый ряд размером на 1 меньше исходного, в него   
# пишутся разности p\_arr\_arma[i+1] – p\_arr\_arma[i].

mu\_arr\_arma = np.append(0., mu\_arr\_arma) # Добавляем спереди (нулевым членом) в ряд 0, # чтобы mu\_arr\_arma[0] было равно 0.

# done!

### Извлечение биржевых данных

Биржевые данные по сделкам (USD-RUB) лежат в основной папке репозитория и имеют имена:

trades-16-02-22.csv, trades-17-02-22.csv, trades-si-14-02-22.csv, trades-si-21-02-22.csv.

Из каждого мы берем только колонку #4 (цены).

Для тестового датасета используем выборку: берем каждую 5000-ную сделку, начиная с нулевой. Таким образом, каждый датасет содержит разное число сделок. Оно лежит около значений 100-200.