Metody numeryczne – projekt 1
“wskaźnik giełdowy MACD”

Michał Soja

175793
Informatyka
semestr 4
grupa 5

28.03.2022r

1. Wstęp

Projekt polega na stworzeniu programu wyliczającego wskaźnika giełdowego MACD oraz analizie wskaźnika i otrzymanych wyników.

W moim projekcie został wykorzystany język C++, a wizualizacja danych wykonałem za pomocą programu LibreOffice Calc.

Wykorzystane zostały dane giełdowe podane przez WIG20 z dni między 20.03.2018r, a 22.03.2022r. Do programu można podać dowolne dane w formacie zgadzającym się z danymi z WIG20. Oprócz tego działa on na dowolnej ilości danych. W tym przypadku jest to 1000 rekordów.

2. Implementacja

Program do obliczeń wykorzystuje głównie kolekcję deque będącą częścią bibliotek szablonów STL.

```
struct dataContainer {
    float fund = 0;
    float wallet = 0;
    std::deque<float> value;
    std::deque<std::string> date;
    std::deque<float> macd;
    std::deque<float> signal;
};
```

Program jako dane wejściowe przyjmuje dane, które musi przerobić na dwie zmienne – dzień i wartość otwarcia. Pozostałe dane są niepotrzebne i są odrzucane. Podając plik danych może być więcej i można je ograniczyć do zadanej przez nas ilości podając parametr numberOfRecords.

```
for (int i = 0; inputFile >> inputBuffor && (i < numberOfRecords || numberOfRecords == -1); i++) {
    auto row = splitString(inputBuffor, ',');

    auto date = row[0];
    data.date.push_back(date);

    auto value = std::stof(row[1]);
    data.value.push_back(value);
```

Następnie program wylicza zmienne ema12 I ema26 wykorzystując wykładniczą średniąk kroczącą. Jako parametry przyjmuje ona ilość kolekcje oraz ilość ostatnich dni, które ma brać pod uwagę. W przypadku liczenia ema12 trzeba wziąc 12 poprzednich wartości i aktualną czyli razem 13. Do pobrania ostatnich X elementów służy funkcja getLastElements.

```
std::deque<float> getLastElements(std::deque<float> container, int number) {
    auto result = std::deque<float>();

    for (int i = (int)container.size() - 1; i >= 0 && i >= (int)container.size() - number; i--) {
        result.push_front(container[i]);
    }
    return result;
}
```

Wykonanie funkcji służącej do wyliczenia średniej wykładniczej polegało na zaimplementowaniu wzoru podanego w instrukcji.

```
float calcExpAverage(std::deque<float> input) {
    auto numerator = 0.0f;
    auto denominator = 0.0f;
    auto alpha = 2 / (float)(input.size() + 1);

    for (int i = (int)input.size() - 1; i >= 0; i--) {
        auto pi = input[i];
        auto n = input.size() - i - 1;
        auto power = (float)pow(1 - alpha, n);

        numerator += pi * power;
        denominator += power;
    }
    return numerator / denominator;
}
```

Następnie program wylicza wskaźnik MACD, który jest różnicą EMA12 I EMA26. MACD wykorzystujemy później do wyliczenia lini sygnału. Do tego również wykorzystujemy funkcję calcExpMovingAverage.

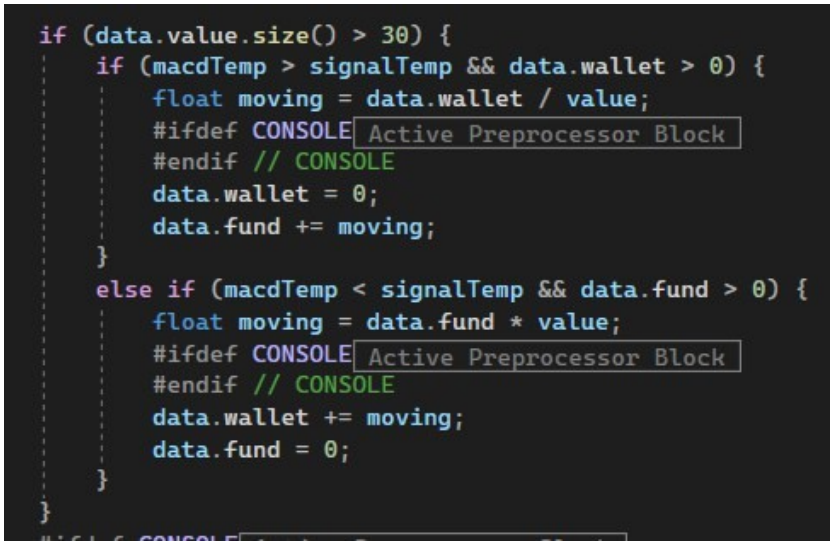
```
auto ema12 = calcExpMovingAverage(data.value, 12);
auto ema26 = calcExpMovingAverage(data.value, 26);
auto macdTemp = ema12 - ema26;
data.macd.push_back(macdTemp);

auto signalTemp = calcExpMovingAverage(data.macd, 9);
data.signal.push_back(signalTemp);
```

W celu przyspieszenia działania programu oraz oszczędniejszej pracy pamięci usuwane są niepotrzebne stare rekordy.

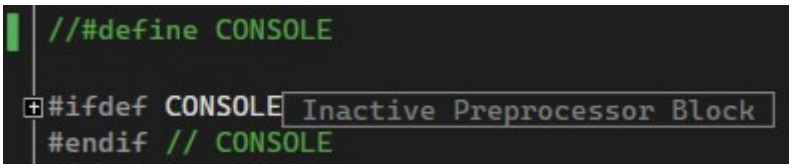
```
void removeUnusedData(dataContainer* temp) {
    if (temp->date.size() > 40) {
        temp->date.pop_front();
        temp->value.pop_front();
        temp->macd.pop_front();
        temp->signal.pop_front();
    }
}
```

Otrzymane wyniki program zapisuje do innego pliku o rozszerzeniu .csv.

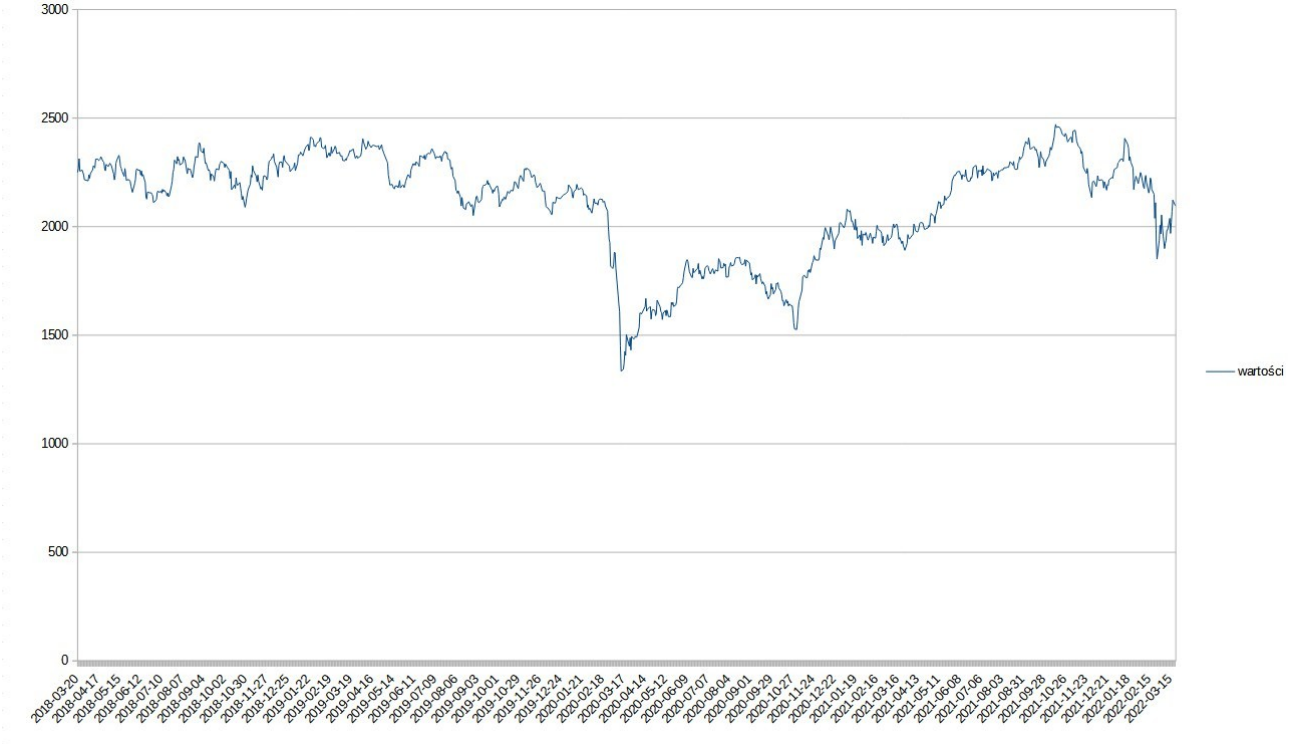
Dodatkową funkcjonalnością programu jest użycie tego algorytmu do powiększenia kwoty zadanej. Program korzystając z wskaźnika MACD I SIGNAL szacuje czy opłaca się kupić akcje czy też należy sprzedać. Jeśli MACD jest większe od SIGNAL – akcje są kupowane, jeśli MACD jest mniejsze od SIGNAL – akcje są sprzedawane.



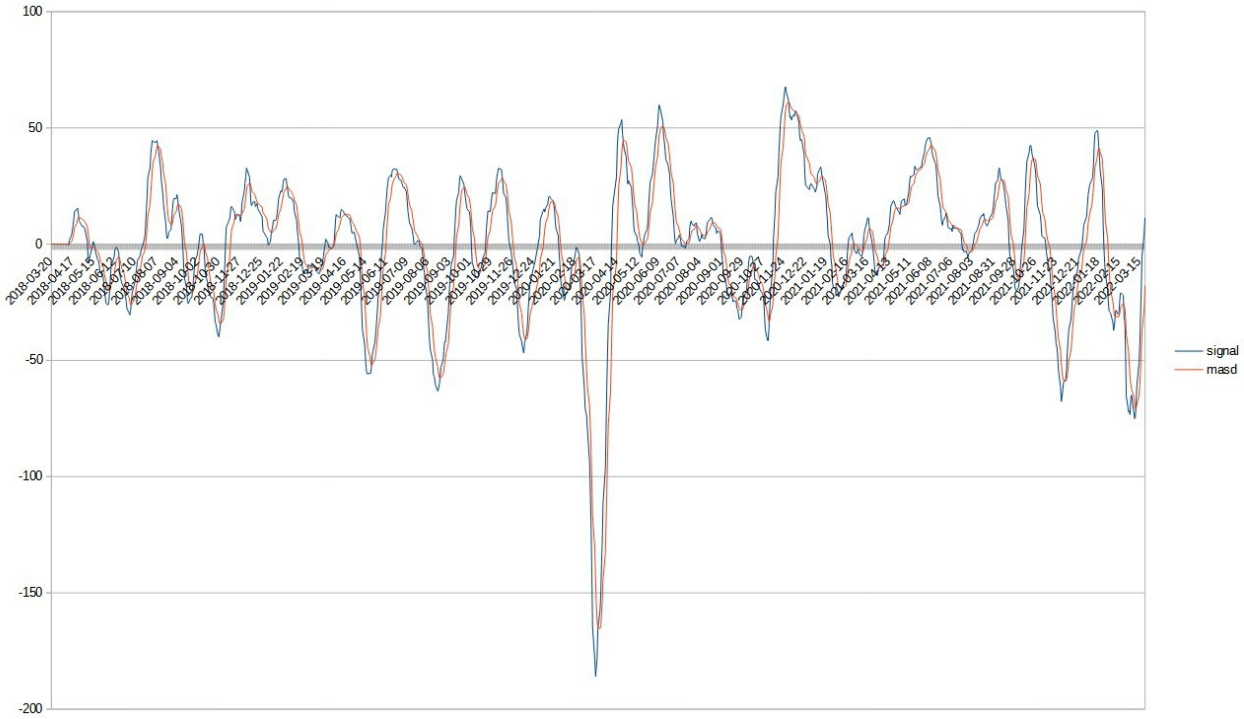
Dodatkowo w celach sprawdzenia, w których dokładnie momentach program sprzedaje akcje, udostępnione jest wypisywanie danych na konsolę. Aby ją włączyć należy odkomentować dyrektywę #define CONSOLE



3. Wyniki



Wykres 1. Historyczne wartości giełdowe



Wykres 2. Wykres wskaźnika MACD i linii SIGNAL

4. Analiza algorytmu

Wyniki algorytmu dla tych danych są zadowalające gdyż zostanie odnotowany zysk. Jednakże wskaźnik MACD pokazuje zmiany giełdowę z opóźnieniem co skutkuje nie do końca trafionymi decyzjami dotyczącymi kupna i sprzedaży. Algorytm ten dobrze się sprawdza dla danych, które rzadko zmieniają monotoniczność. Dla danych o bardziej gwałtownych zmianach przecięcia linii MACD I SIGNAL nie zawsze pokażą optymalne rozwiązanie.

Korzystając tylko z przecięć zyski nie byłyby duże. Większe możliwości zyskujemy podczas obserwacji zmiany wskaźnika MACD. Im szybciej on rośnie tym więcej możemy zyskać. Niestety późno on reaguje na zmiany przez co podczas gwałtownego spadku ceny akcji można również dużo stracić.

5. Wnioski

Wskaźnik ten nie nadaje się do giełdy o dużej dynamice. W giełdach o mniejszej dynamice natomiast pokazuje zadowalające dane. Dzięki temu można stwierdzić, że wskaźnik ten może być przydatny do analizy technicznej.