Arytmetyka komputerowa

Zadanie numer 7

Cel zadania:

Wyznaczyć kolejne elementy ciągu $x_{k+1} = x_k + 3x_k\,(1 - x_k)$, $x_0 = 0.1$, i porównać otrzymane wartości dla różnej precyzji zmiennych (*float*, *double*, *long double*). Powtórzyć doświadczenie dla przekształconej postaci wzoru: $x_{k+1} = 4x_k - 3x_k x_k$. Spróbować wyjaśnić otrzymane wyniki.

Rozwiązanie w języku c++:

Funkcja główna:

```
int main() {
    int iter = 200;
    float xpf = 0.1, xmf = 0.1;
    double xpd = 0.1, xmd = 0.1;
    long double xpld = 0.1, xmld = 0.1;
    auto start :time_point<...> = chrono::steady_clock::now();
    for(int i = 1; i <= iter; i++){
        xpf = float_eq( x: xpf);
        xpd = double_eq( x: xpd);
        xpld = double_long_eq( x: xpld);
        xmf = float_eq_simple( x: xmf);
        xmd = double_eq_simple( x: xmd);
        xmld = double_long_eq_simple( x: xmld);
        cout << i << "float: " << xpf << " double: "<< xpd << " long double: " << xpld << endl;
        cout << i << "float: " << xmf << " double: "<< xmd  << " long double: " << xmld << endl;
```

Funkcje obliczające i zwracające kolejne elementy ciągu:

```
float float_eq(float x){
    return x + 3 * x * (1 - x);
}

double double_eq(double x){
    return x + 3 * x * (1 - x);
}

long double double_long_eq(long double x){
    return x + 3 * x * (1 - x);
}

float float_eq_simple(float x){
    return 4 * x - 3 * x * x;
}

double double_eq_simple(double x){
    return 4 * x - 3 * x * x;
}

long double double_long_eq_simple(long double x){
    return 4 * x - 3 * x * x;
}
```

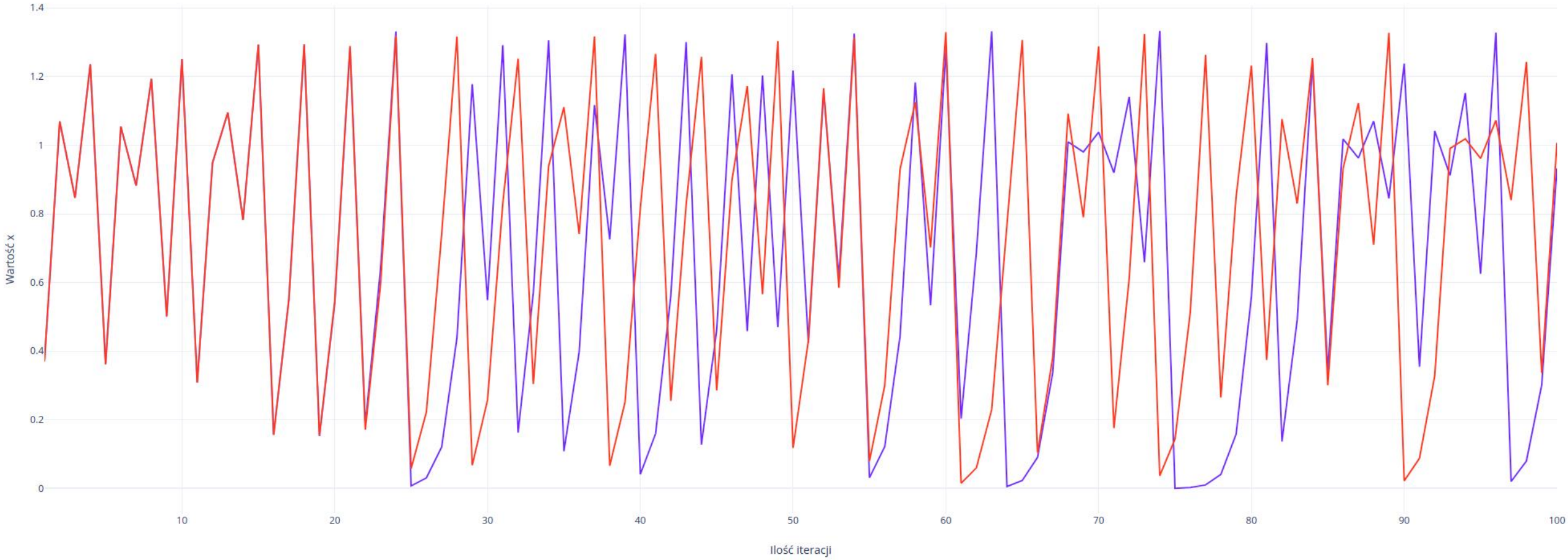
Wyniki:

Numer iteracji	Wzór pierwszy			Wzór po modyfikacji		
	float	double	long double	float	double	long double
1	0.37	0.37	0.37	0.37	0.37	0.37
2	1.0693	1.0693	1.0693	1.0693	1.0693	1.0693
3	0.846993	0.846993	0.846993	0.846993	0.846993	0.846993
4	1.23578	1.23578	1.23578	1.23578	1.23578	1.23578
5	0.36166	0.36166	0.36166	0.36166	0.36166	0.36166
15	**1.29301**	**1.2932**	**1.2932**	**1.29307**	**1.2932**	**1.2932**
16	0.156427	0.155706	0.155706	0.156202	0.155706	0.155706
17	0.5523	0.550092	0.550092	0.551612	0.550092	0.550092
18	1.29409	1.29256	1.29256	1.29362	1.29256	1.29256
19	0.152338	0.15809	0.15809	0.15412	0.15809	0.15809
20	0.539731	0.557382	0.557382	0.545222	0.557382	0.557382

	Wzór pierwszy			Wzór po modyfikacji		
Numer iteracji	float	double	long double	float	double	long double
99	**0.301088**	**0.635468**	**0.388615**	**0.33674**	**1.29**	**0.0191609**
100	0.932389	1.33041	1.1014	1.00678	0.167684	0.0755422
101	1.12151	0.011655	0.766367	0.986305	0.586382	0.285049
102	0.712692	0.0462124	1.30351	1.02683	1.314	0.896437
103	1.32698	0.178443	0.116613	0.944187	0.0762256	1.17495
104	0.0252993	0.618245	0.425656	1.10228	0.287471	0.558279
196	**0.00699961**	**0.000328681**	**1.15679**	**0.0221337**	**1.07719**	**1.33286**
197	0.0278515	0.0013144	0.612663	0.0870651	0.827757	0.00187435
198	0.109079	0.00525241	1.32458	0.325519	1.25548	0.00748684
199	0.40062	0.0209269	0.0347677	0.984189	0.293219	0.0297792
200	1.12099	0.0823937	0.135445	1.03087	0.914944	0.116456

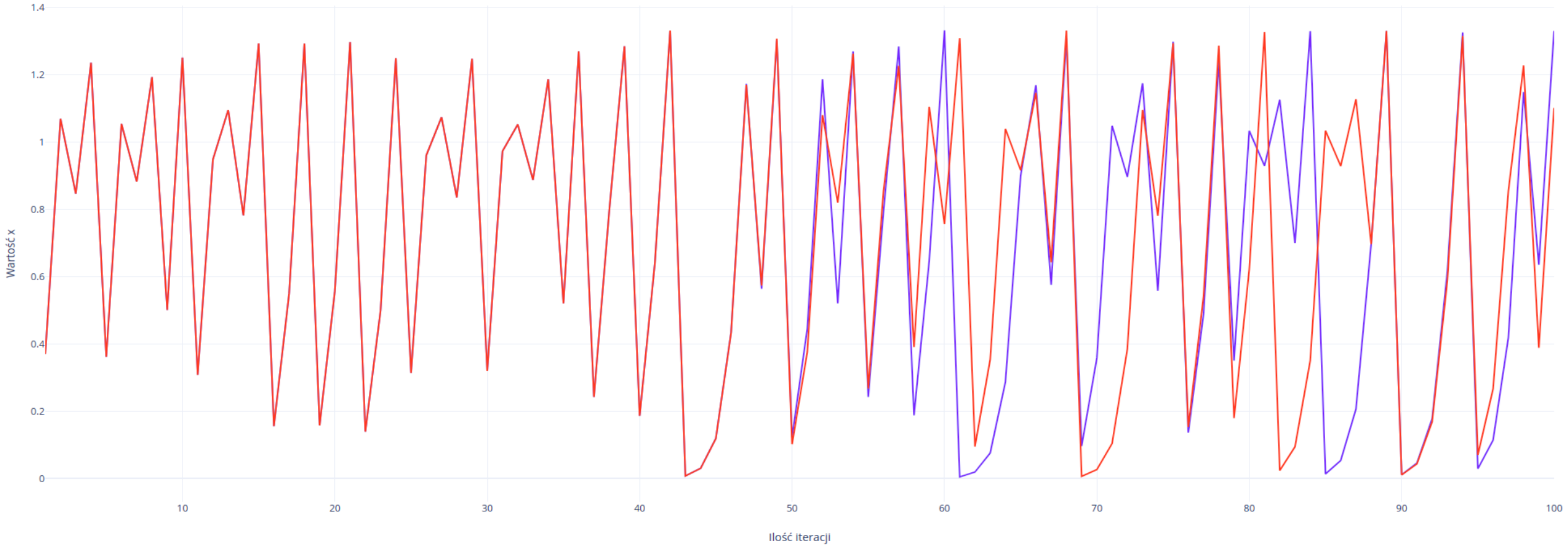
Wyniki obliczone dla pierwszych 14 elementów ciągu są takie same, jednak już od 15 elementu zaczynają się nieznacznie różnić dla obu użytych wzorów. Już przy 100 elemencie możemy dostrzeć wyraźną różnice w wynikach. Dla wzoru po modyfikacji, wyniki szybciej zaczynają się znacząco różnić. Przy 200 elemencie wyniki stają się już zupełnie rozbieżne, więc dalsza ich analiza nie ma wnosi nic nowego.

Wykres przedstawia wyniki w pierwszych stu iteracjach dla typu Float



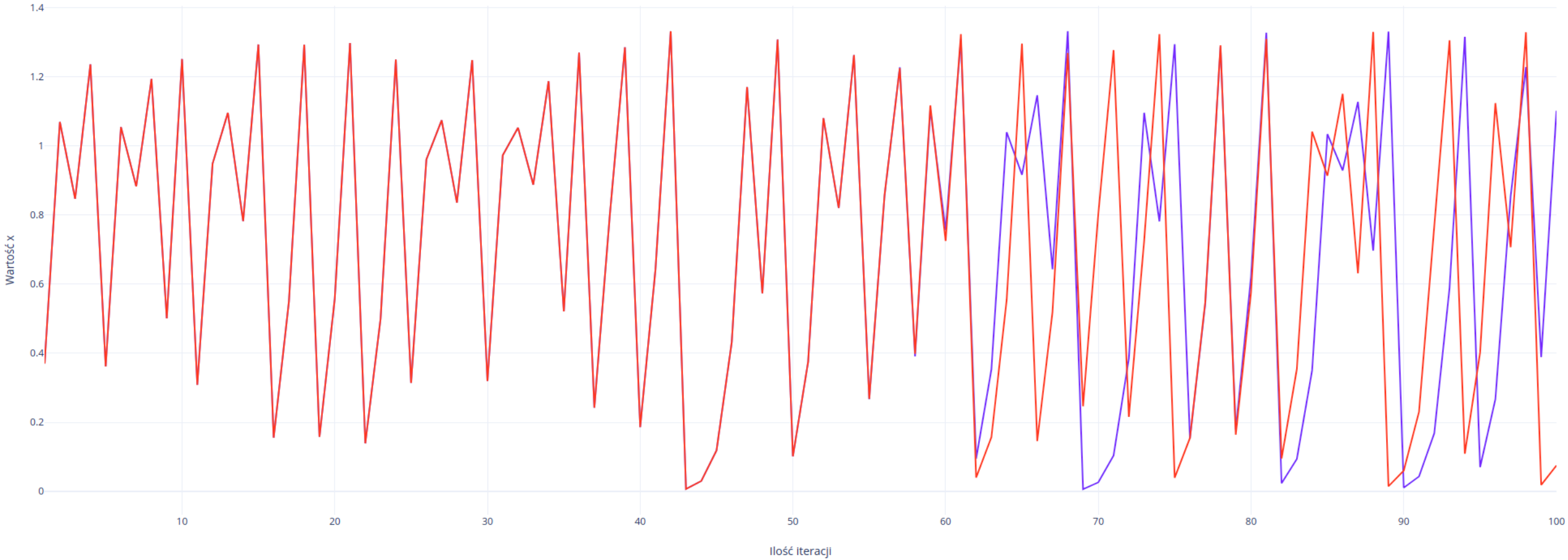
Wzór pierwszy Wzór po modyfikacji

Wykres przedstawia wyniki w pierwszych stu iteracjach dla typu Double



wzór pierwszy wzór po modyfikacji

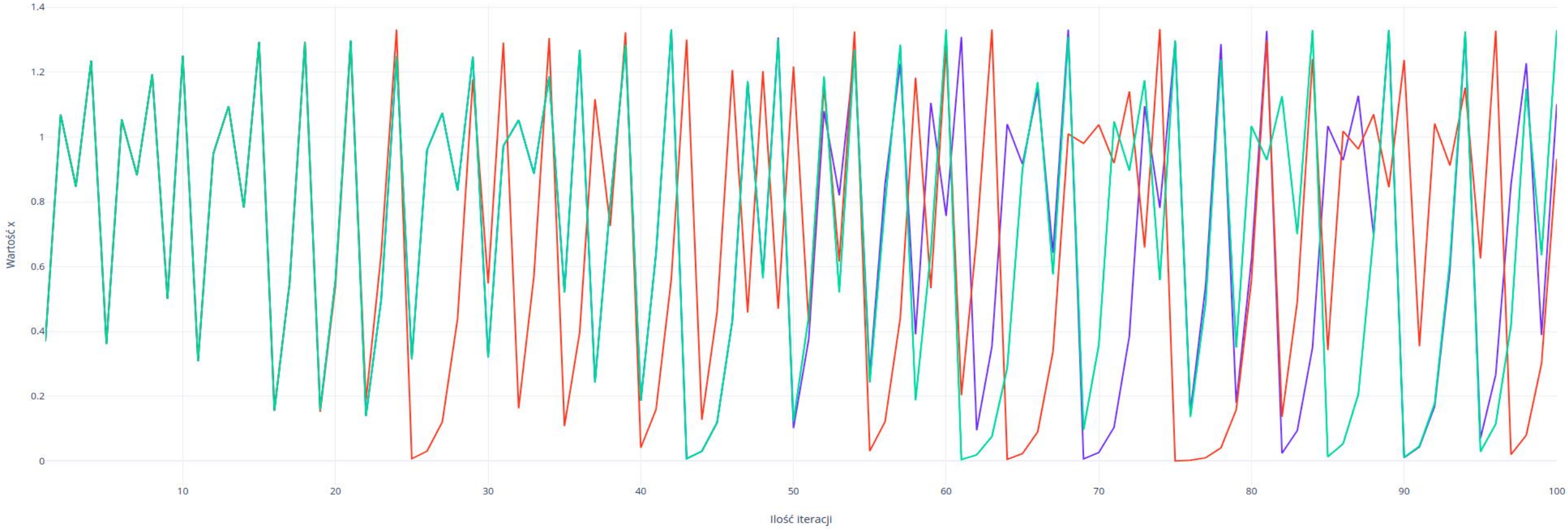
Wykres przedstawia wyniki w pierwszych stu iteracjach dla typu Long Double



wzór pierwszy wzór po modyfikacji

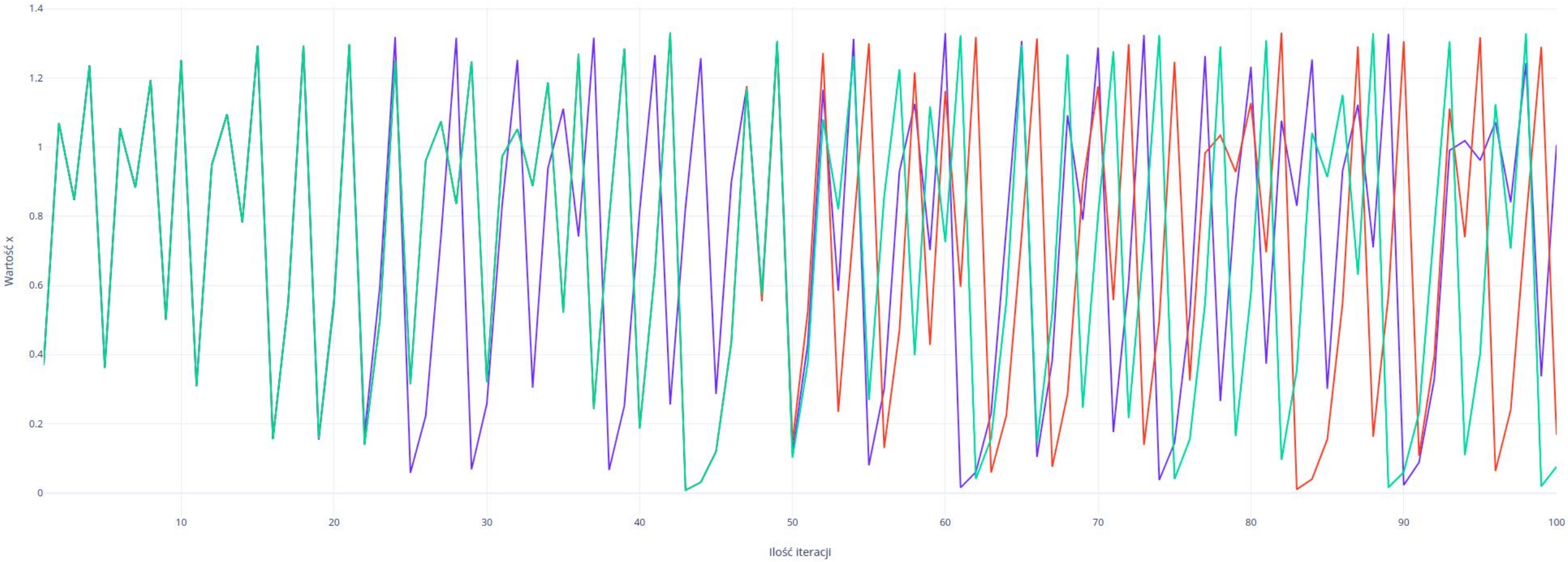
Dla typu float elementy ciągu zaczynają się różnić przy około 25 elemencie. Obliczając coraz większe elementy ciągu ta różnica staje się coraz bardziej wyrażniejsza.Dla typu long obliczone elemnety zaczynają być rozbieżne przy 50 elemencie, a dla long double 62. Różnice te wynikają z dokładności każdej z tych zmiennych, która jest dla float najmniejsza, a dla long double największa.

Wykres przedstawia wyniki z wykorzystaniem pierwszego wzoru w pierwszych stu iteracjach



Long Double Float Double

Wykres przedstawia wyniki z wykorzystaniem wzoru po modyfikacji w pierwszych stu iteracjach



Float Long Double Double

We wzorze po modyfikacji różnice są większe i częstsze, wynika to z tego, że występuje tam więcej operacji mnożenia niż we wzorze podstawowym, więc overflow i underflow występuje szybciej.

Zmierzone czasy w sekundach dla różnych typów i ilości obliczonych elementów ciągu

Typ Danych		Wzór pierwszy	Wzór po modyfikacji
Float	10000000	0.151	0.153
	100000000	1.478	1.537
	1000000000	14.690	15.204
Double	10000000	0.146	0.152
	100000000	1.482	1.524
	1000000000	14.627	15.227
Long Double	10000000	0.213	0.223
	100000000	2.090	2.201
	1000000000	20.742	21.931

Podsumowanie i wnioski

- Dla typu o większej ilości bitów czas wykonywania operacji jest dłuższy.
- Typy zmiennych mają różne precyzje, zatem nie zawsze można obliczyć dokładny wynik I nalezy dopasowywać typ do konkretnego zadania
- Różne działania matematyczne mogą dawać inne wyniki dla różnych typów

- Rozwiązanie zostało napisane za pomocą języka C++, kompilator gcc.
- Do wygenerowania wykresów wykorzystano stronę: https://chart-studio.plotly.com
- Intel Corei3-4030U CPU @ 1.90GHz × 4 , RAM: 8GB
- System Operacyjny: Ubuntu 20.04.4 LTS

Martyna Olszewska