# Dominik Ciesiołkiewicz 44289 – Sprawozdanie lab 5 – poprawione

W porównaniu do mojego wcześniejszego kodu uległa zmianie jedna zmienna w wyznaczaniu AM która była tam niepoprawnie, usunąłem phi z sygnału informacyjnego oraz zmieniłem częstotliwość fs oraz F dla zwiększenia czytelności. Moja odpowiedź do zadania 3 znajduje się w zakomentowanej sekcji kodu.

**Kod:**

#include <iostream>

#include <complex>

#include <fstream>

#define \_USE\_MATH\_DEFINES

double pi = 3.14159265359;

using namespace std;

complex<double>\* DFT(const double\* tab, int N)

{

complex<double>\* tab2 = new complex<double>[N];

for (int k = 0; k < N; k++)

{

tab2[k] = 0;

complex<double> WN = cos(tab[k]) + 1i \* sin(tab[k]);

for (int n = 0; n < N; n++)

{

tab2[k] += tab[n] \* pow(WN, -k \* n);

}

}

return tab2;

}

double ton\_prosty(double a, double F, double t)

{

//cout << a << " " << F << " " << t << endl;

return a \* sin(2 \* pi \* F \* t);

}

int main()

{

double a = 1;//volty

double A = 2;

double F = 0.5;

double phi = 2 \* pi;

double fs = 300;// (?)

double Ts = 1 / fs;

//double kA = 0.5, kp = 1.5; //(a)

//double kA = 10, kp = 3; //(b)

double kA = 90, kp = 99; //(c)

ofstream saveOX("zad1OX.txt");

ofstream saveTonProsty("zad1sig.txt");

ofstream saveM("zad1M.txt");

ofstream saveZa("zad1Za.txt");

ofstream saveZp("zad1Zp.txt");

int count = 0;

for (double i = 0; i < A; i = i + Ts)

{

count++;

}

double\* sig = new double[count];

double\* Za = new double[count];

double\* Zp = new double[count];

int ilosc = count;

count = 0;

double fn = 50 / double(ilosc);

//cout << fn << endl;

for (double i = 0; i < A; i = i + Ts)

{

sig[count] = ton\_prosty(a, F, i);

saveOX << i << endl;

saveTonProsty << sig[count] << endl;

Za[count] = (kA \* sig[count] + 1) \* cos(2 \* pi \* fn \* count);

saveZa << Za[count] << endl;

Zp[count] = cos(2 \* pi \* fn \* i + kp \* sig[count]);

saveZp << Zp[count] << endl;

count++;

}

//edit do zad 3

/\*

double famin, famax, Wa;

for (double i = 0; i < A; i = i + Ts)

{

sig[count] = ton\_prosty(a, F, i);

saveOX << i << endl;

saveTonProsty << sig[count] << endl;

Za[count] = (kA \* sig[count] + 1) \* cos(2 \* pi \* fn \* count);

if (Za[count] < -3)

Za[count] = -3;

saveZa << Za[count] << endl;

if (count == 0)

{

famin = Za[count];

famax = Za[count];

}

else

{

if (Za[count] < famin)

{

famin = Za[count];

}

if (Za[count] > famax)

{

famax = Za[count];

}

}

Zp[count] = cos(2 \* pi \* fn \* i + kp \* sig[count]);

saveZp << Zp[count] << endl;

count++;

}

Wa = famax - famin;

cout << "Szerokosc pasma sygnalu: " << Wa << endl;

//a: Zmodulowana amplituda: 2.99901

//b: Zmodulowana amplituda: 13.9803

//c: Zmodulowana amplituda: 93.8224 \*/

//zad2

complex<double>\* DFTvalues = DFT(Za, count);

ofstream saveSpectrum("zad2Spectrum.txt");

ofstream saveMprim("zad2Mprim.txt");

double\* M = new double[ilosc];

double\* Mprim = new double[ilosc];

for (int i = 0; i < count; i++)

{

M[i] = sqrt(pow(real(DFTvalues[i]), 2) + pow(imag(DFTvalues[i]), 2));

saveSpectrum << M[i] << endl;

Mprim[i] = 10 \* log10(M[i]);

saveMprim << Mprim[i] << endl;

}

//zamkniecie strumieni

saveSpectrum.close();

saveMprim.close();

saveOX.close();

saveTonProsty.close();

saveM.close();

saveZa.close();

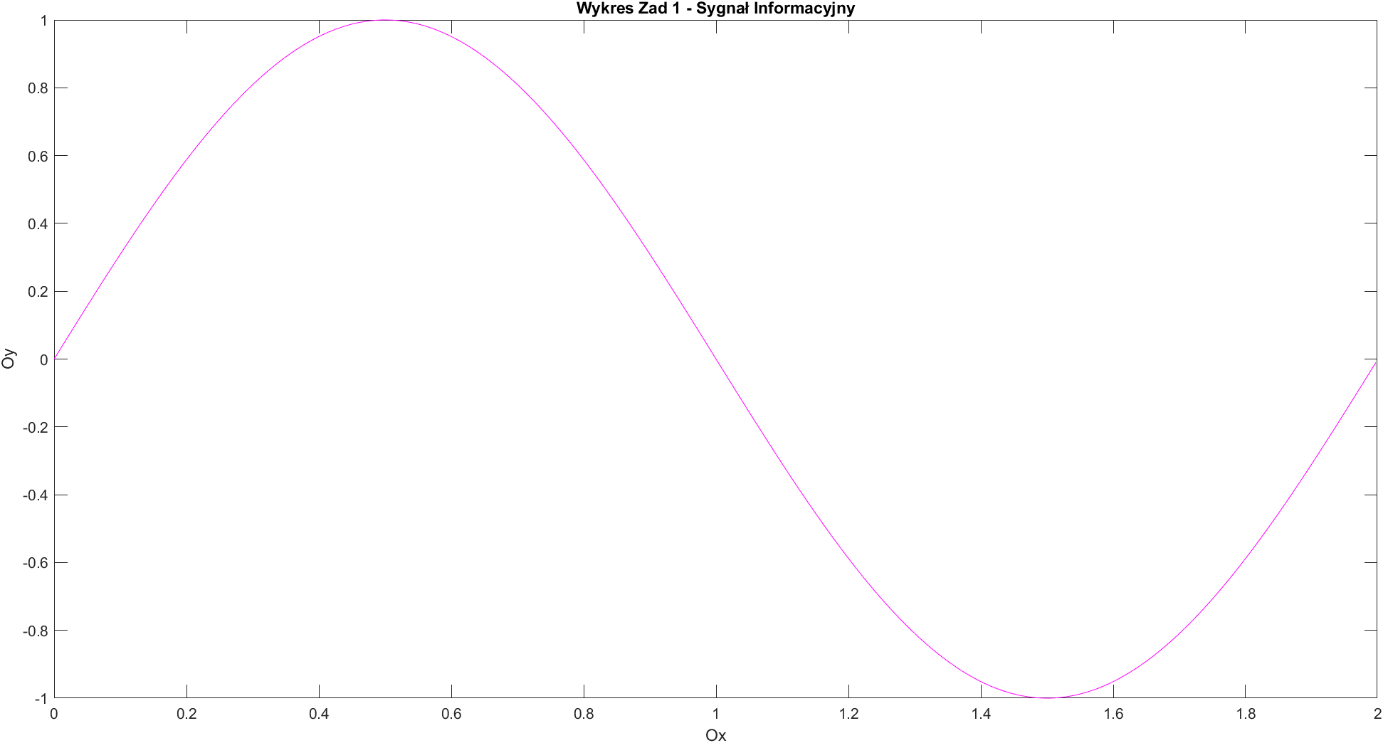
saveZp.close();

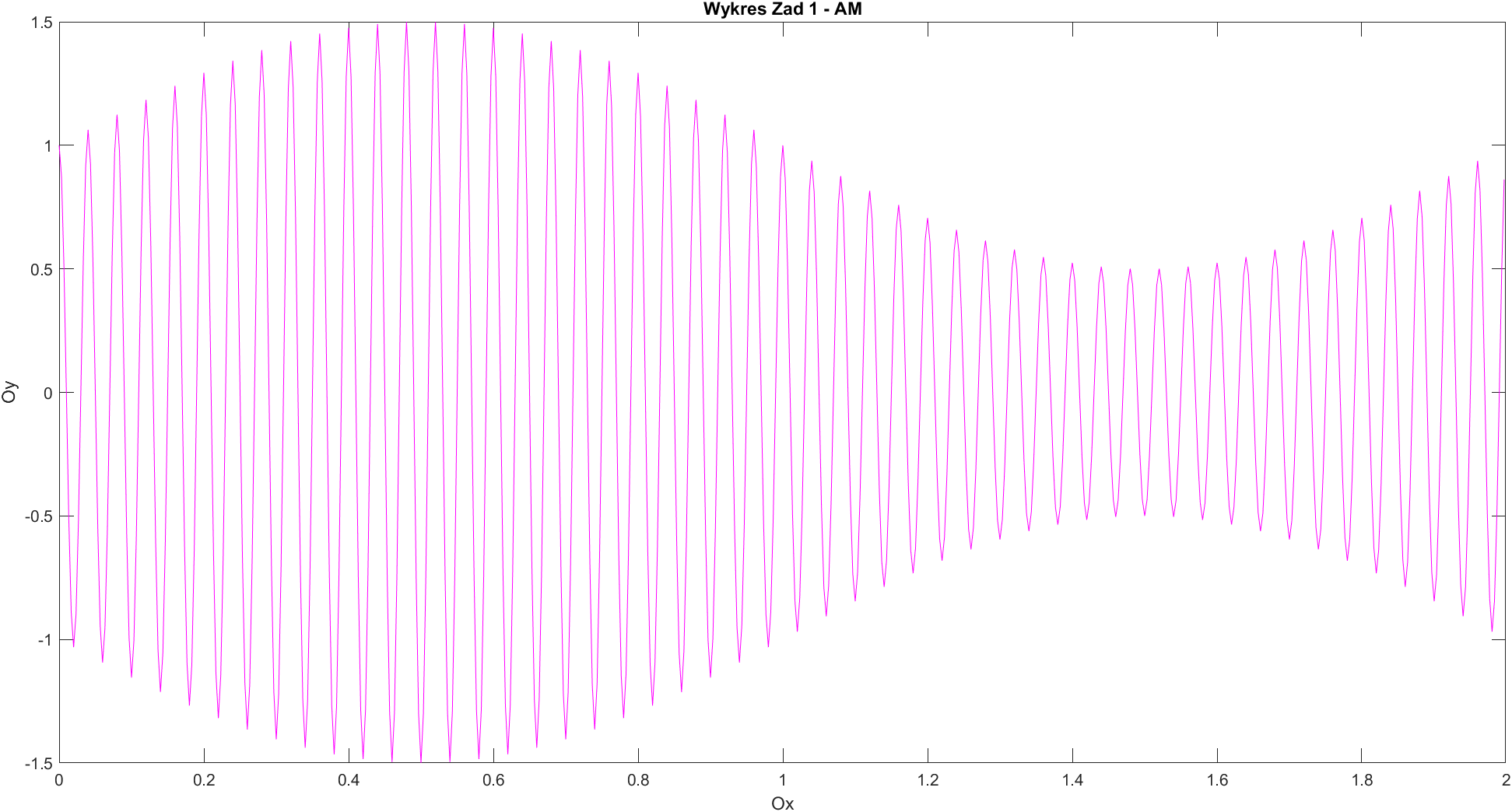
return 0;

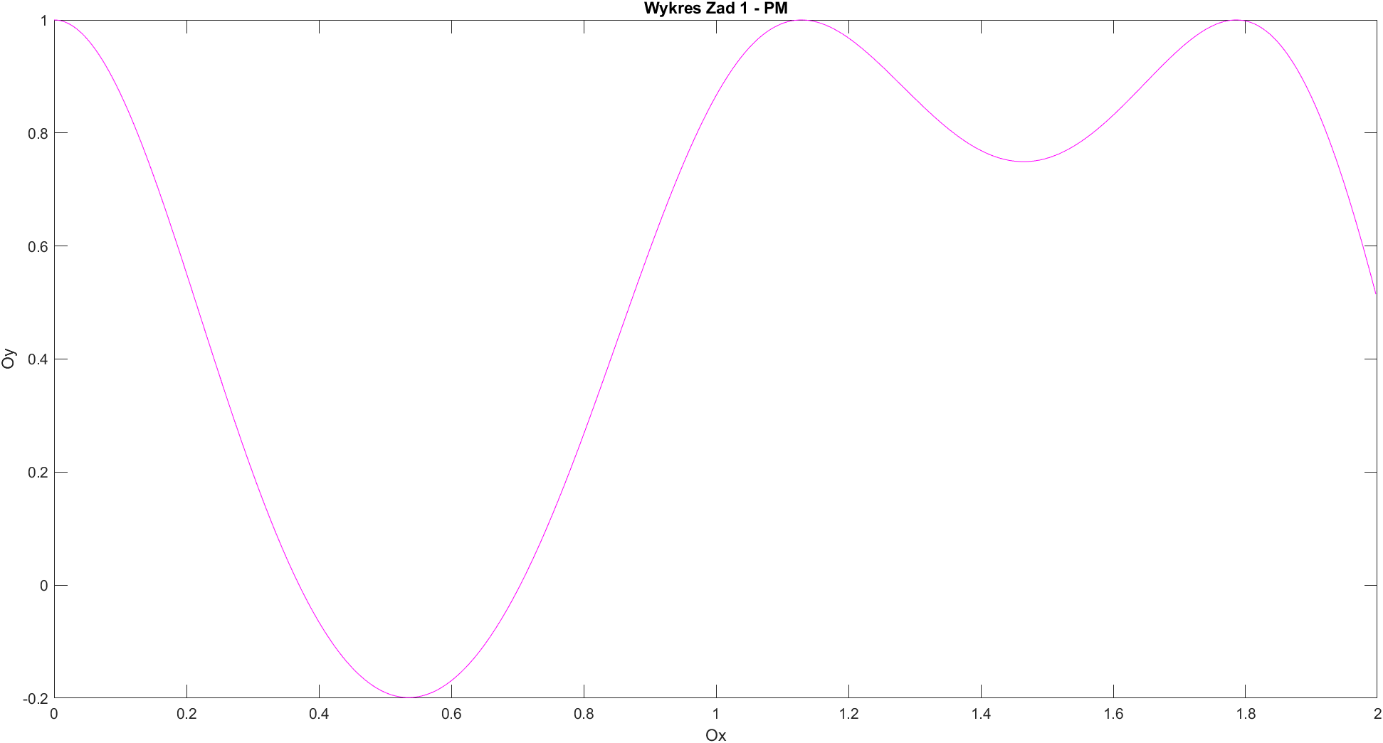
}

**Wykresy:**

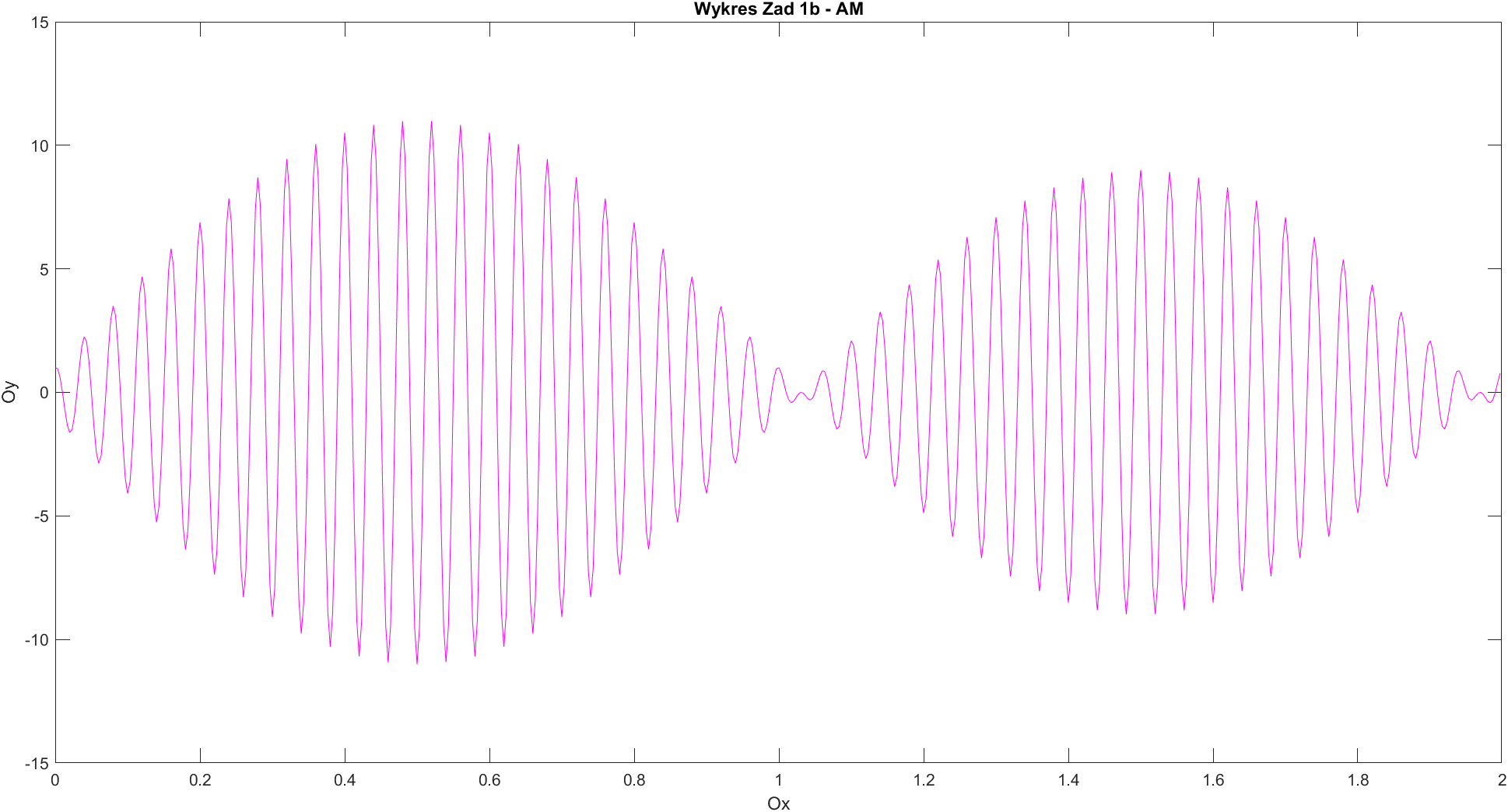
**1a)**

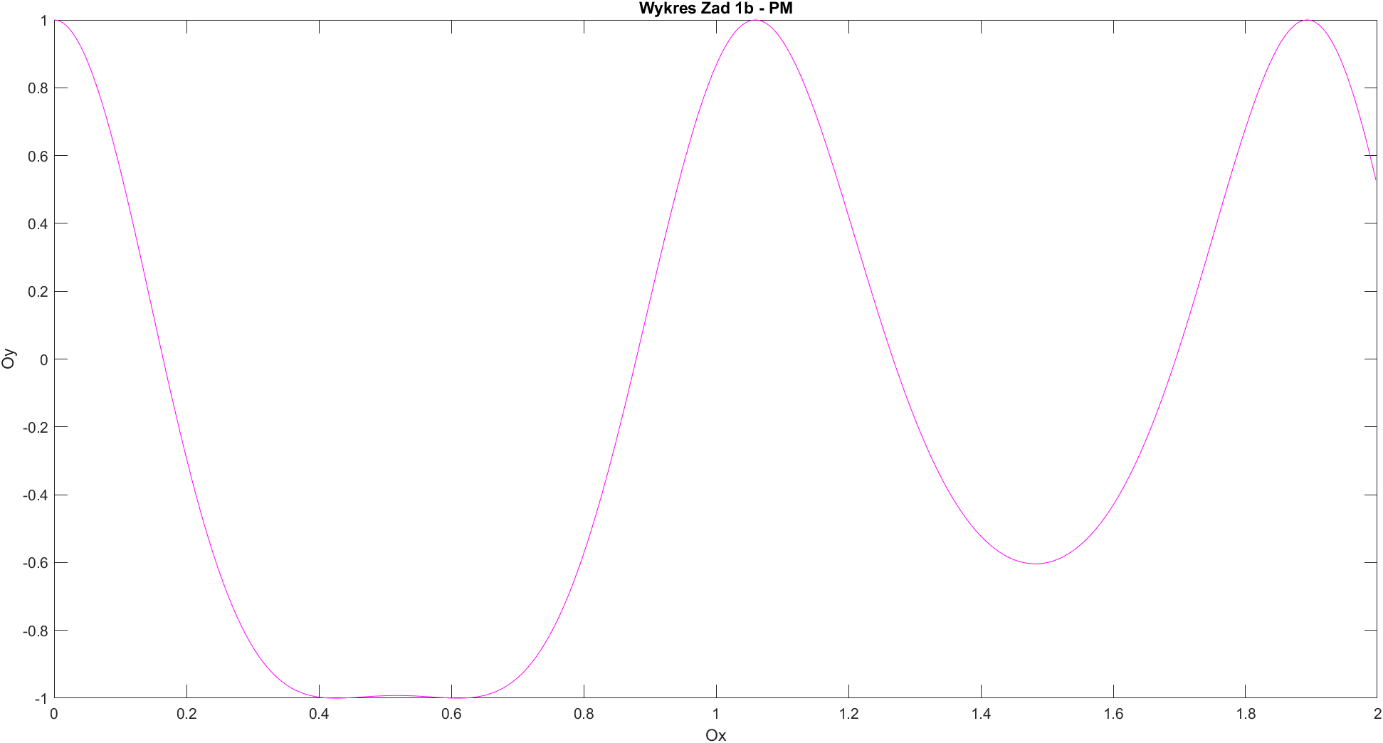
****

****

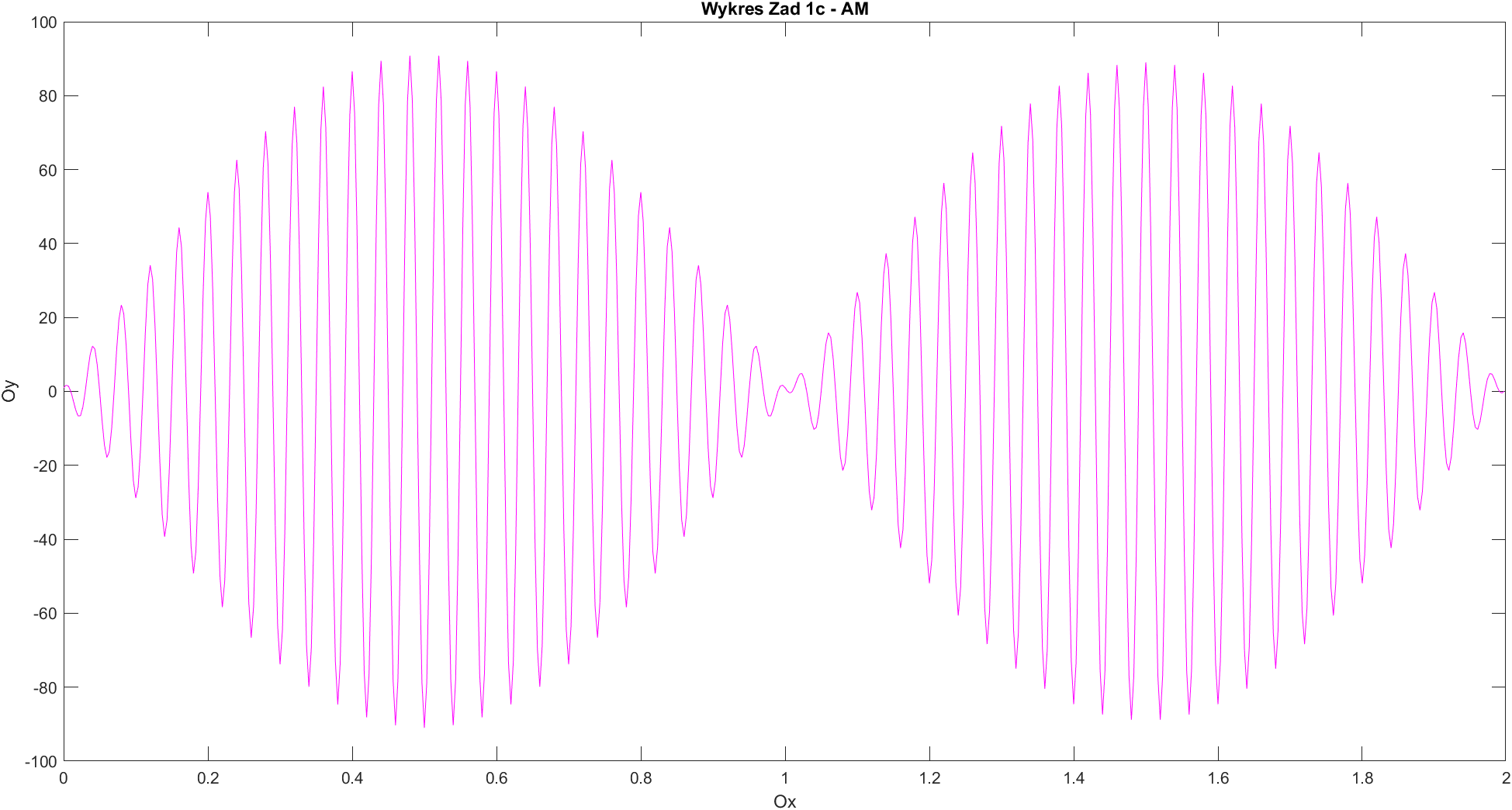
****

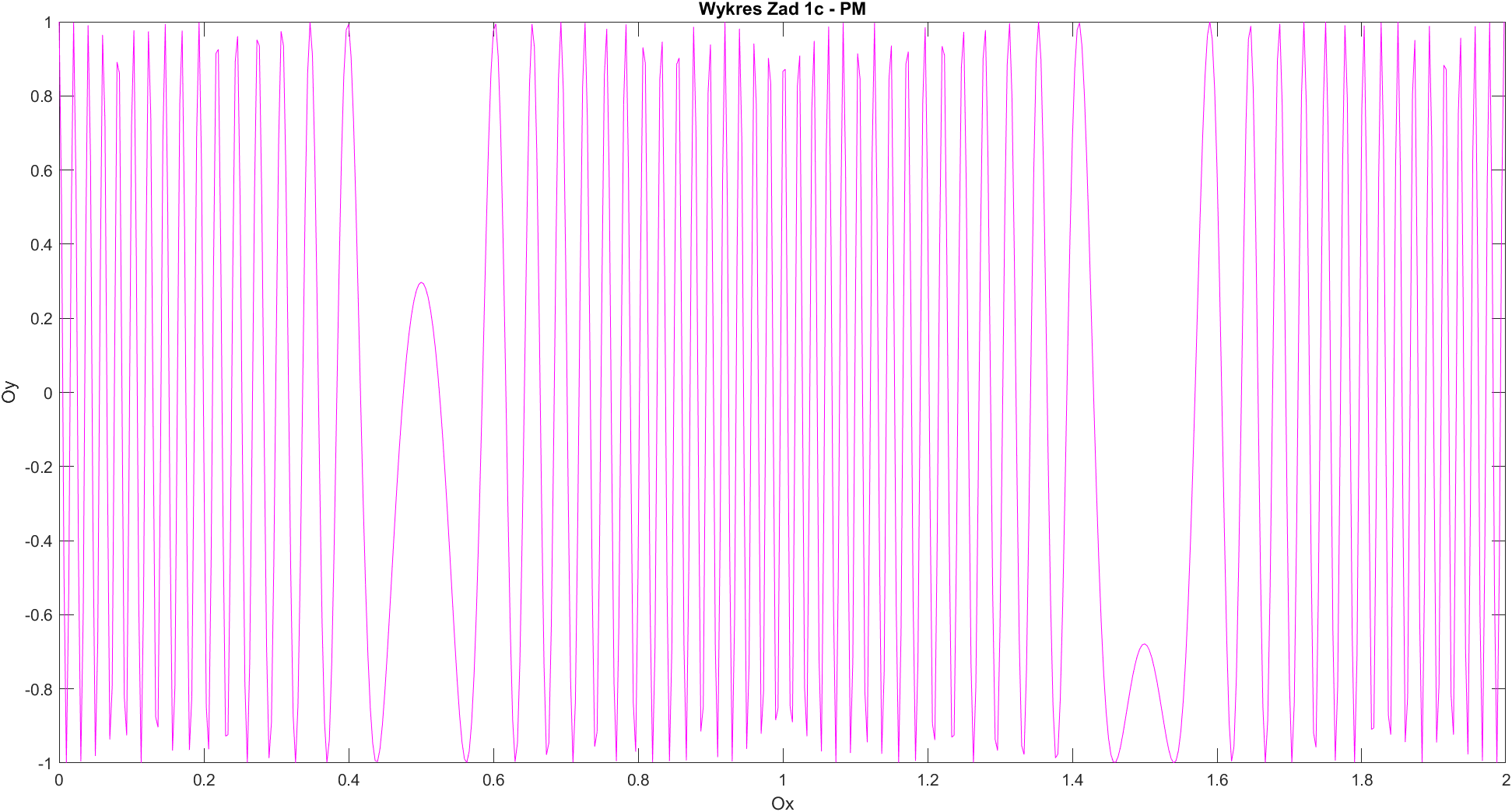
**1b)**

****

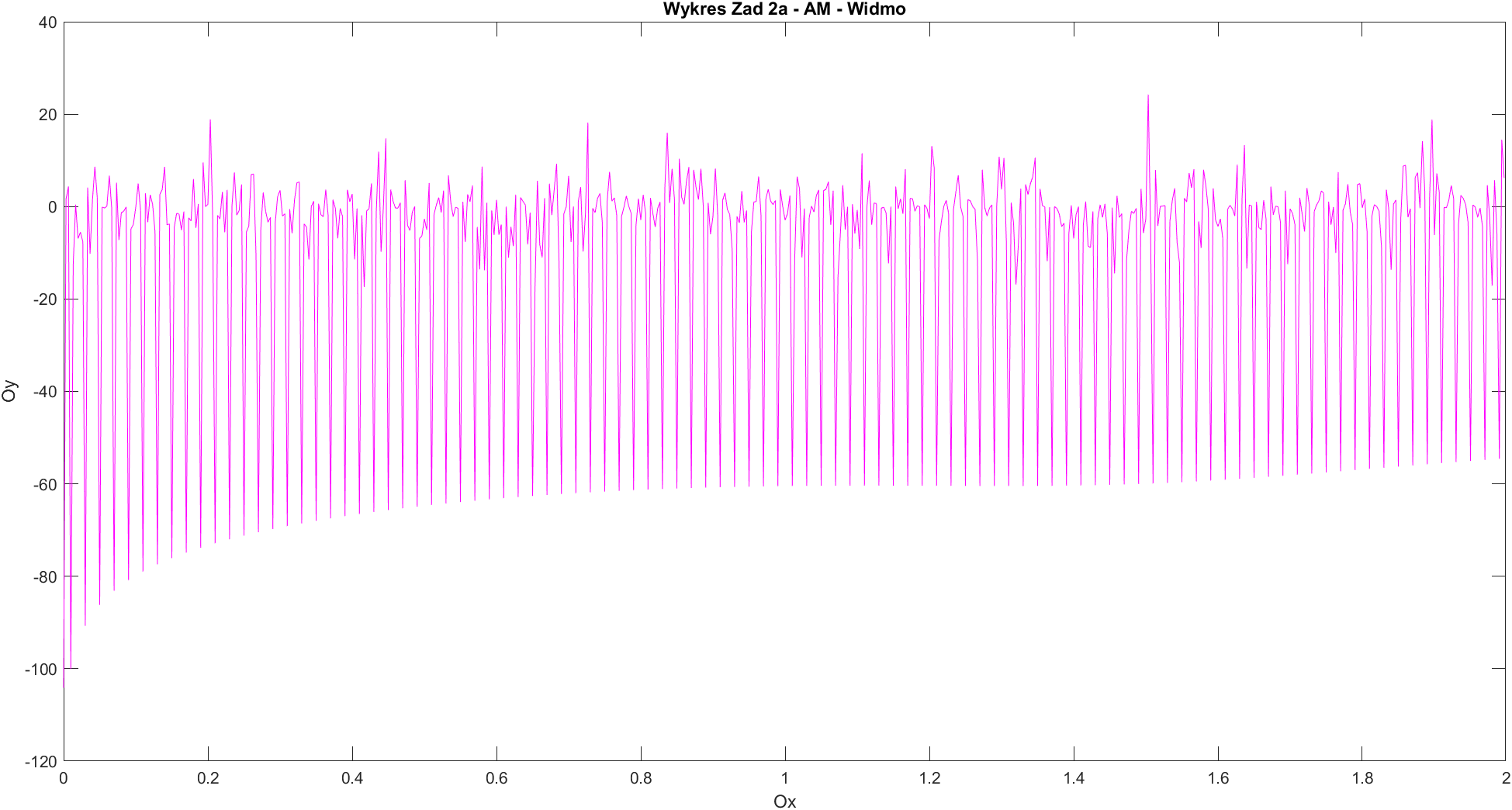
****

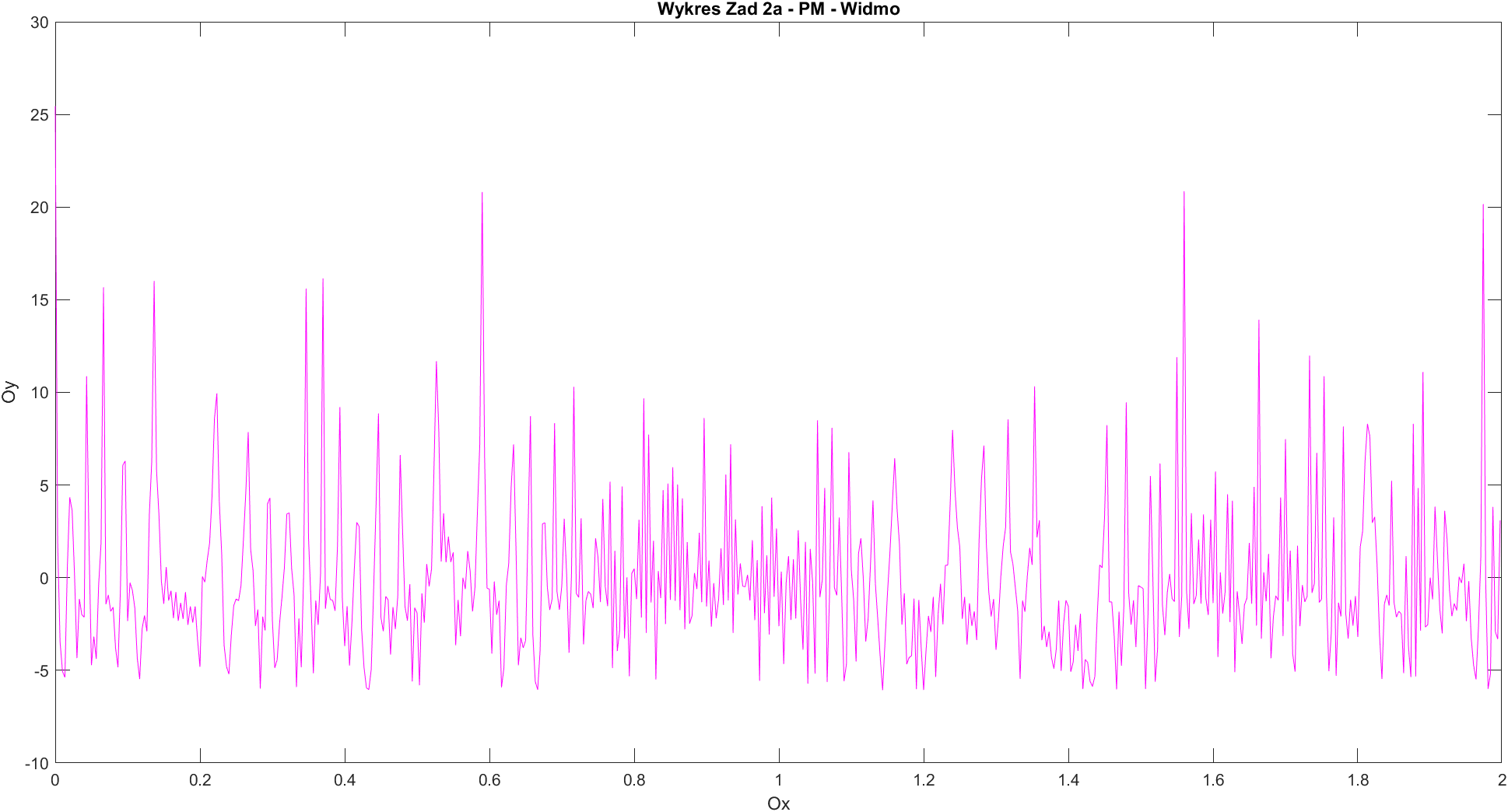
**1c)**

****

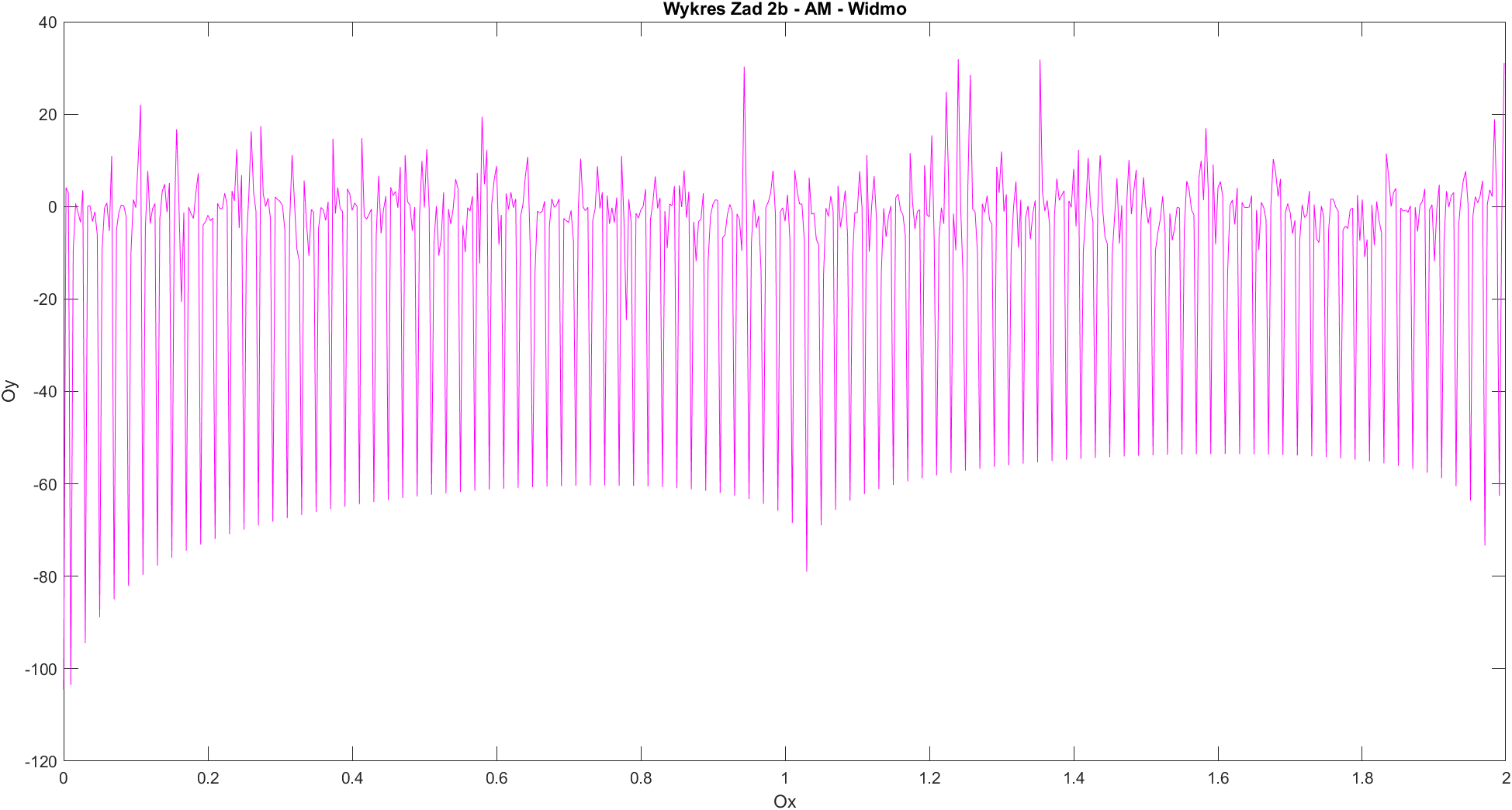
****

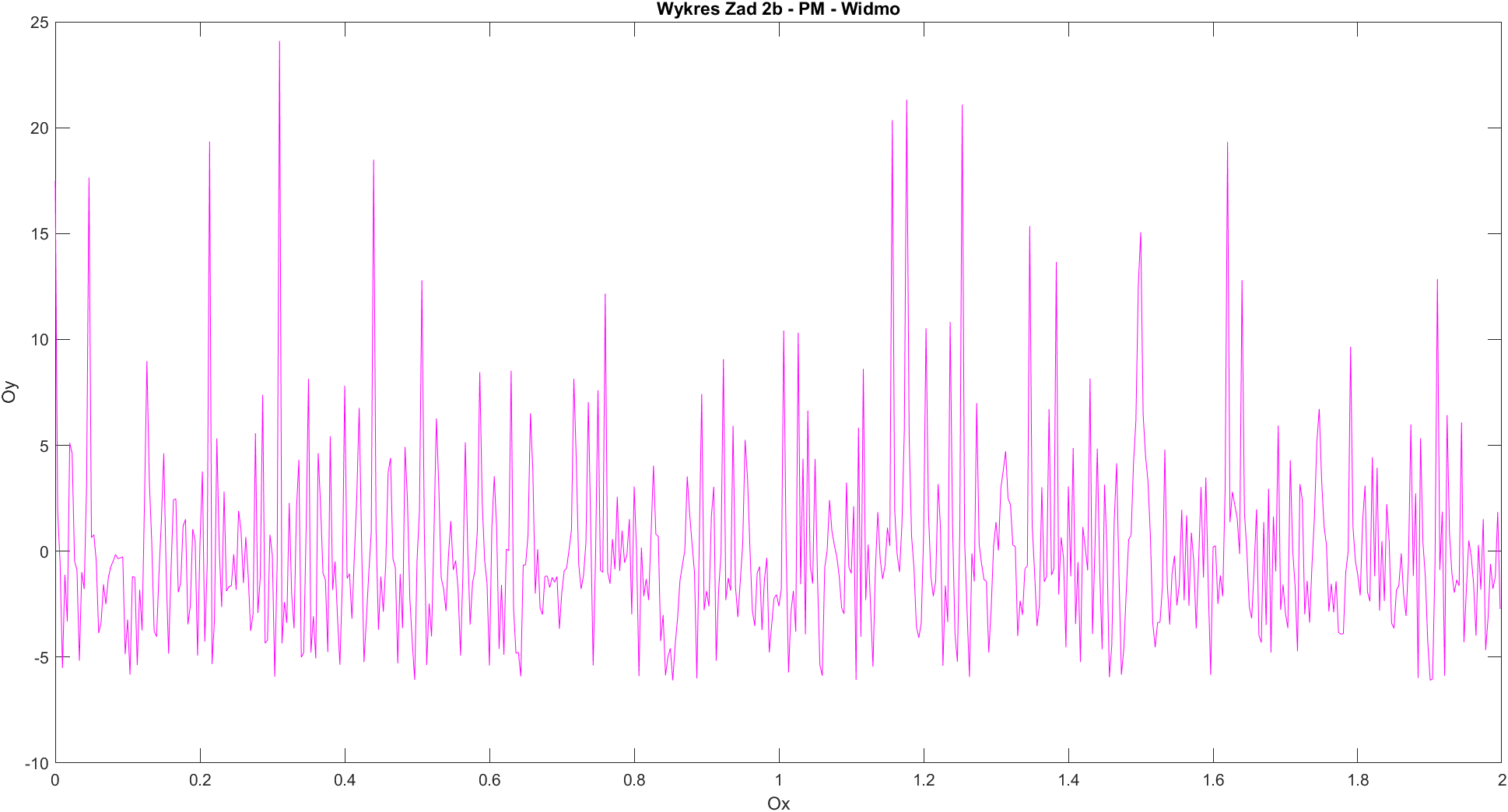
**2a)**

****

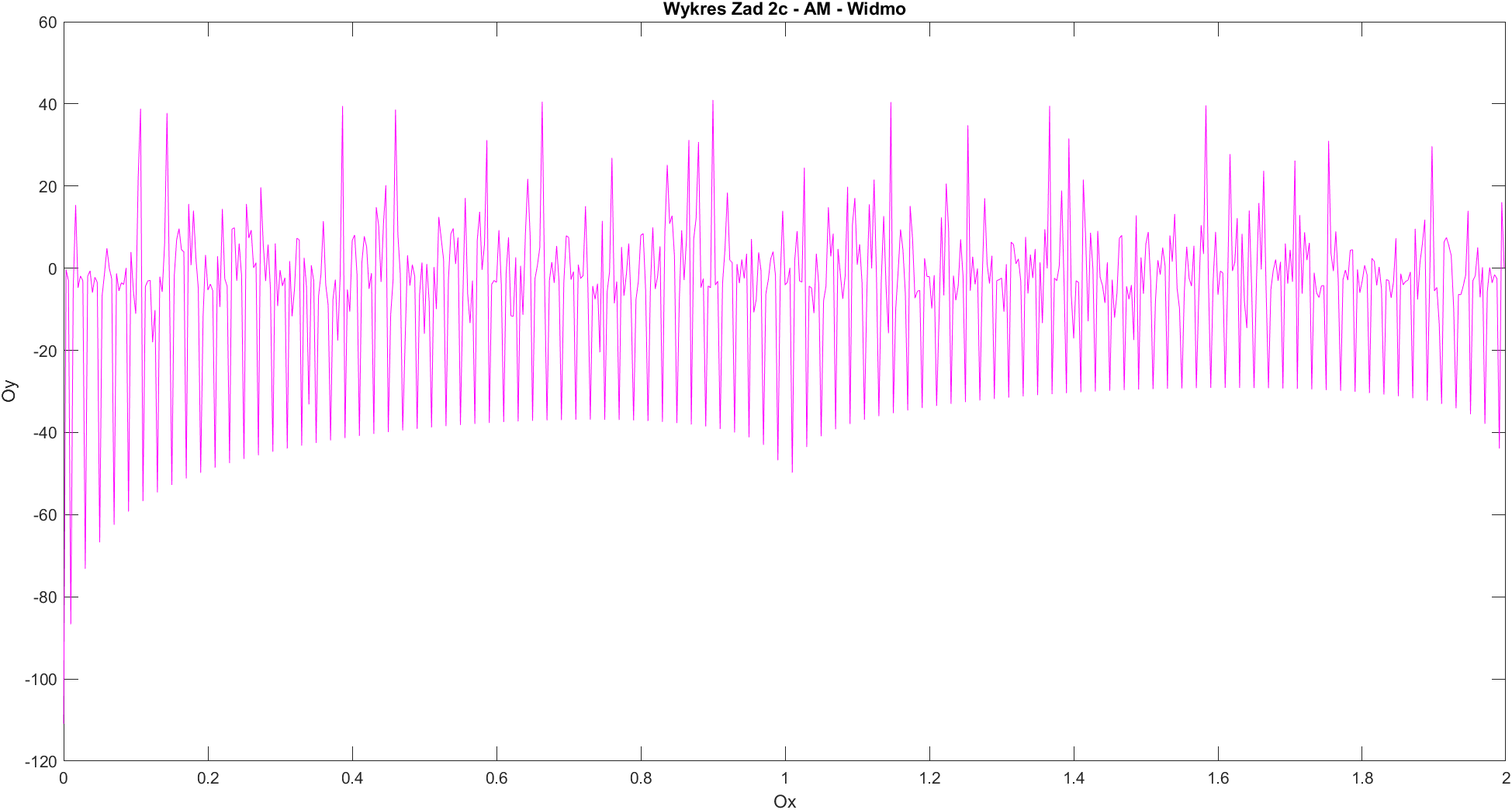
****

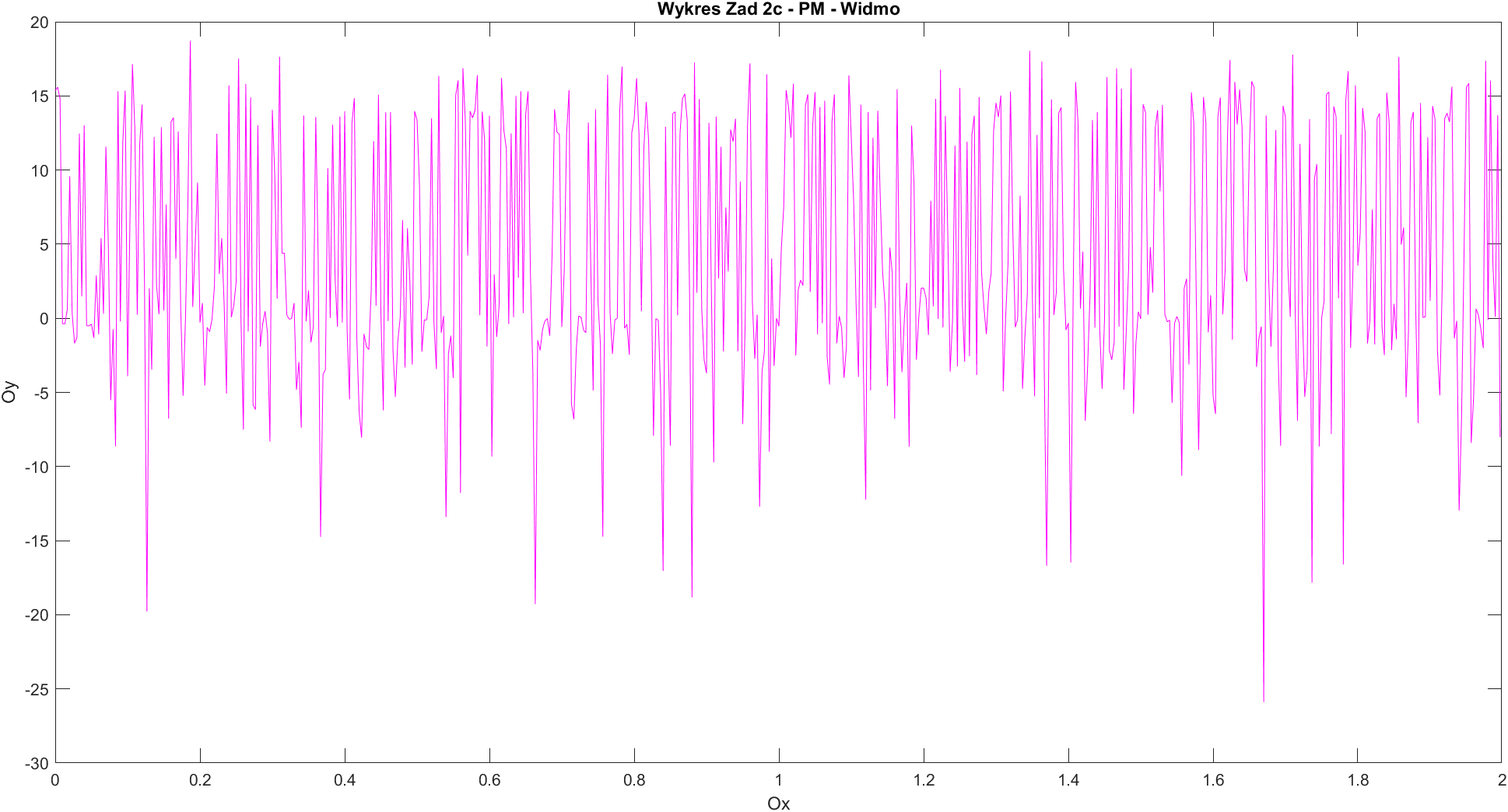
**2b)**

****

****

**2c:**

****

****