# Relatório Rápido com meus resultados do módulo-2

Aluno: Ruben Esteche Araujo

CPF: 109.429.904-98

Segue abaixo os gráficos plotados em sala e fora dela, contendo o material referente ao módulo 2 envolvendo simulações computacionais sobre o efeito Magnus e a força de Lorentz.

**Programa utilizado para geração dos pontos utilizados nos arquivos do trabalho do módulo 2, sobre o efeito Magnus:**

#include<stdlib.h>

#include<math.h>

#include<stdio.h>

//função que tira módulo vetorial

double modulo(double vx, double vy, double vz){

double m;

m = vx\*vx + vy\*vy + vz\*vz;

m = sqrt(m);

return (m);

}

int main(){

//Definindo constantes de movimento da bola de beisebol

float Cd=0.0006, Cm=0.00004;

int g=10;

//Definindo variáveis de movimento

int n,i;

double rx,ry,rz, vx,vy,vz, wx,wy,wz ,dt=0.0001, v;

//criando e abrindo arquivos

FILE \*Rx\_magnus;

FILE \*Ry\_magnus;

FILE \*Rz\_magnus;

Rx\_magnus = fopen("eixo x (Magnus).txt","w+");

Ry\_magnus = fopen("eixo y (Magnus).txt","w+");

Rz\_magnus = fopen("eixo z (Magnus).txt","w+");

//obtendo valores do usuário

printf("Digite as componentes da velocidade inicial (x,y,z):\n");

scanf("%lf %lf %lf",&vx,&vy,&vz);

printf("Digite as componentes da velocidade angular (x,y,z):\n");

scanf("%lf %lf %lf",&wx,&wy,&wz);

printf("Digite o número de interações:\n ");

scanf("%d",&n);

//interações numéricas

for(i=0;i<n;i++){

//preenchendo arquivos

fprintf(Rx\_magnus,"%lf\n", rx);

fprintf(Ry\_magnus,"%lf\n", ry);

fprintf(Rz\_magnus,"%lf\n", rz);

//tirando o módulo vetorial da velocidade

v=modulo(vx,vy,vz);

//calculando velocidades

vx= vx + (Cm\*(wy\*vz - wz\*vy)- (v\*vx)\*Cd)\*dt;

vy= vy + (Cm\*(wx\*vz - wz\*vx)- (v\*vy)\*Cd)\*dt;

vz= vz + (-g + Cm\*(wx\*vy - wy\*vx)- (v\*vz)\*Cd)\*dt;

//calculando posições

rx= rx + vx\*dt;

ry= ry + vy\*dt;

rz= rz + vz\*dt;

}

//fechando o arquivo

fclose(Rx\_magnus);

fclose(Ry\_magnus);

fclose(Rz\_magnus);

printf("Os arquivos Magnus foram criados com sucesso.\n");

return 0;

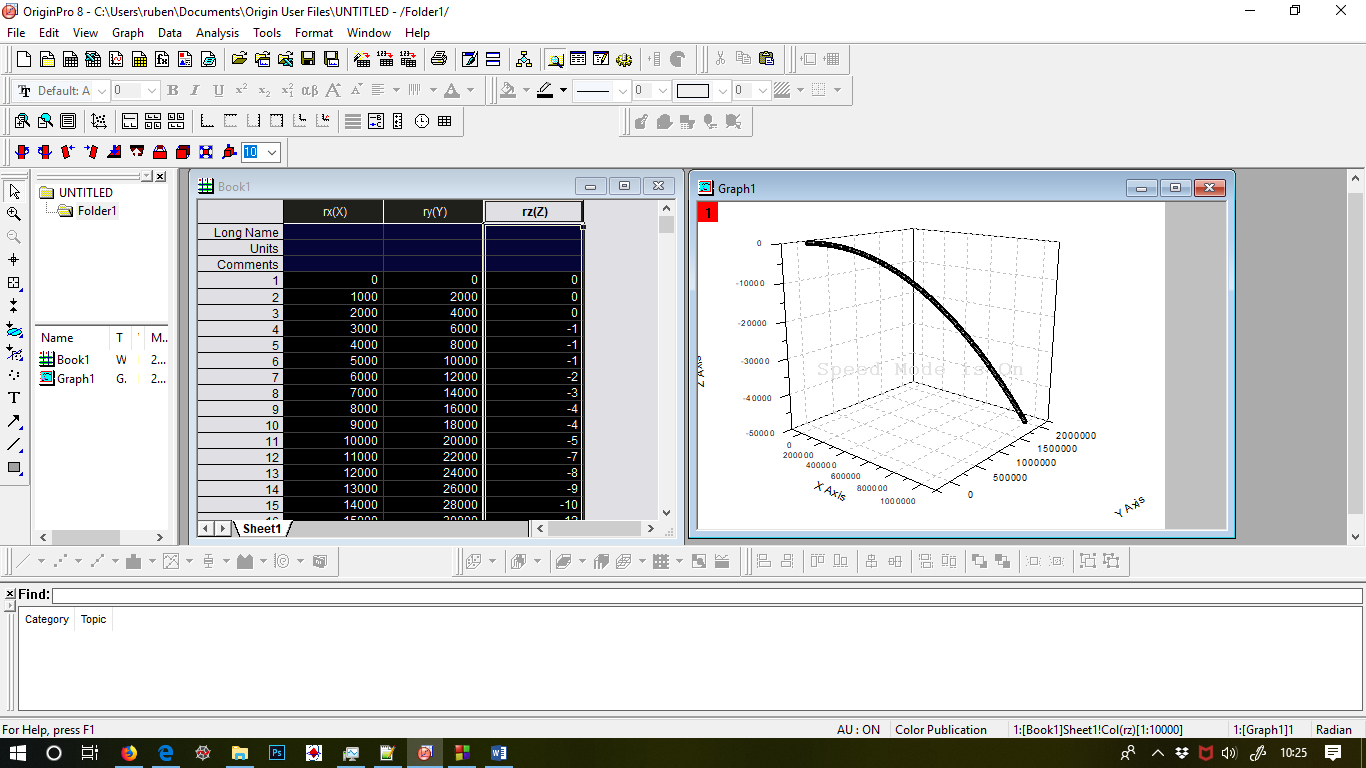
}

**Gráficos gerados para conclusão da terceira tarefa:**

**Primeiro teste**

**(w=0)**

Vox=10; Voy=20; Voz=0; || Wox=0; Woy=0; Woz=0;

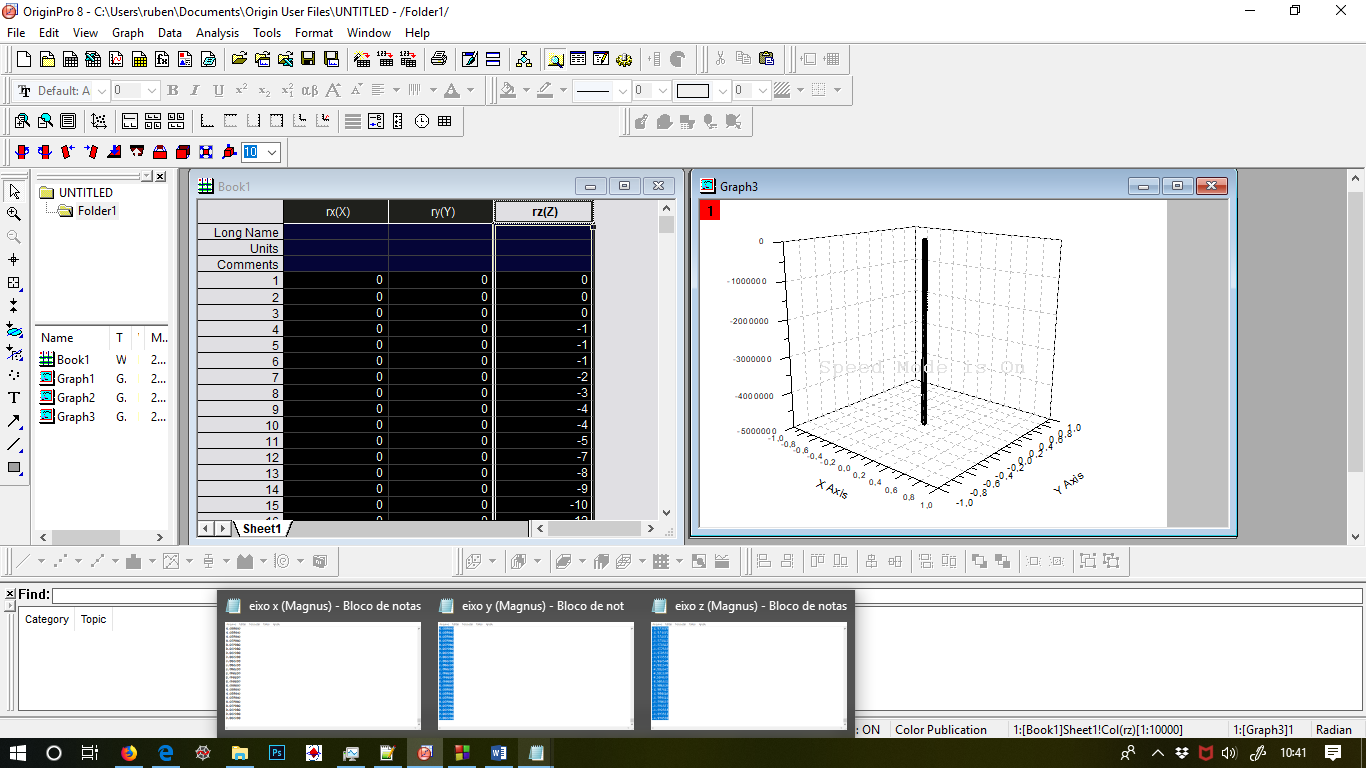


**Segundo teste**

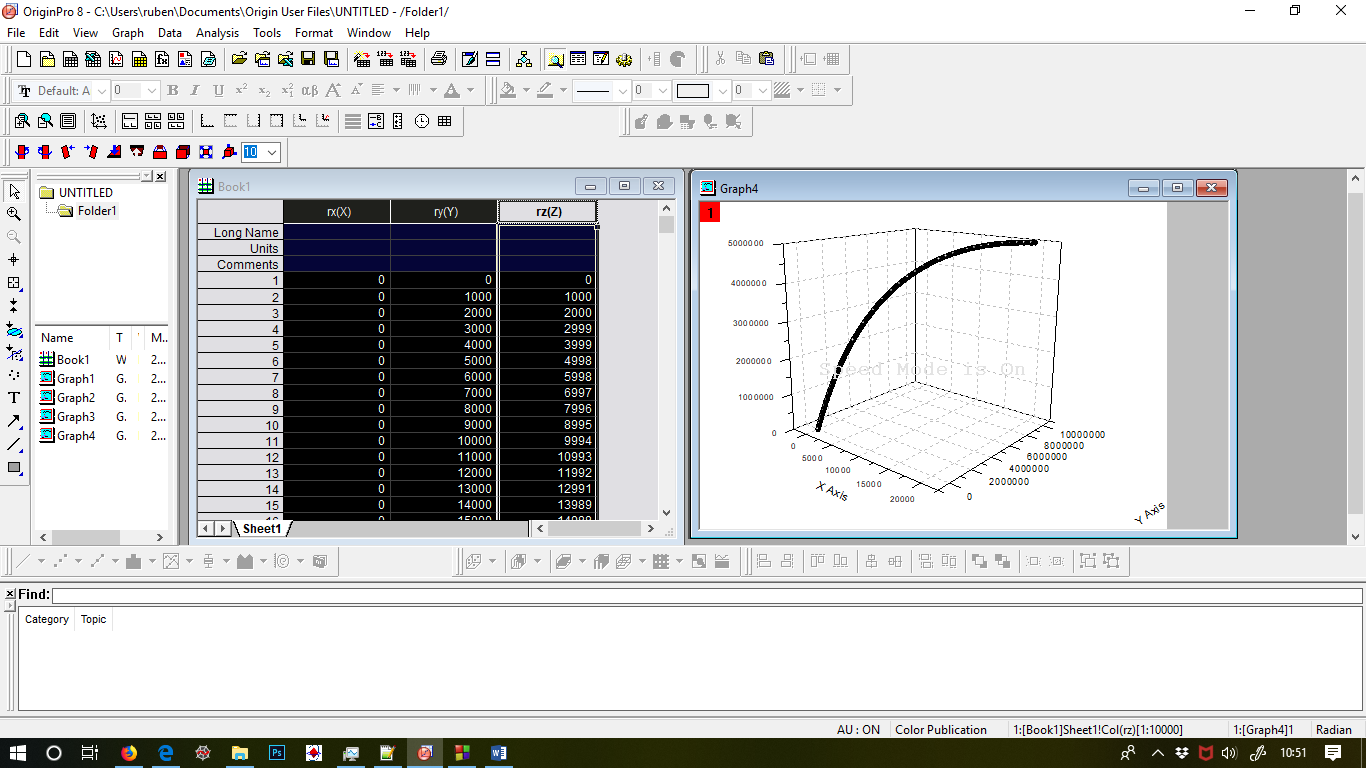
**(w||g)**

Vox=0; Voy=0; Voz=0; || Wox=0; Woy=0; Woz=-10;

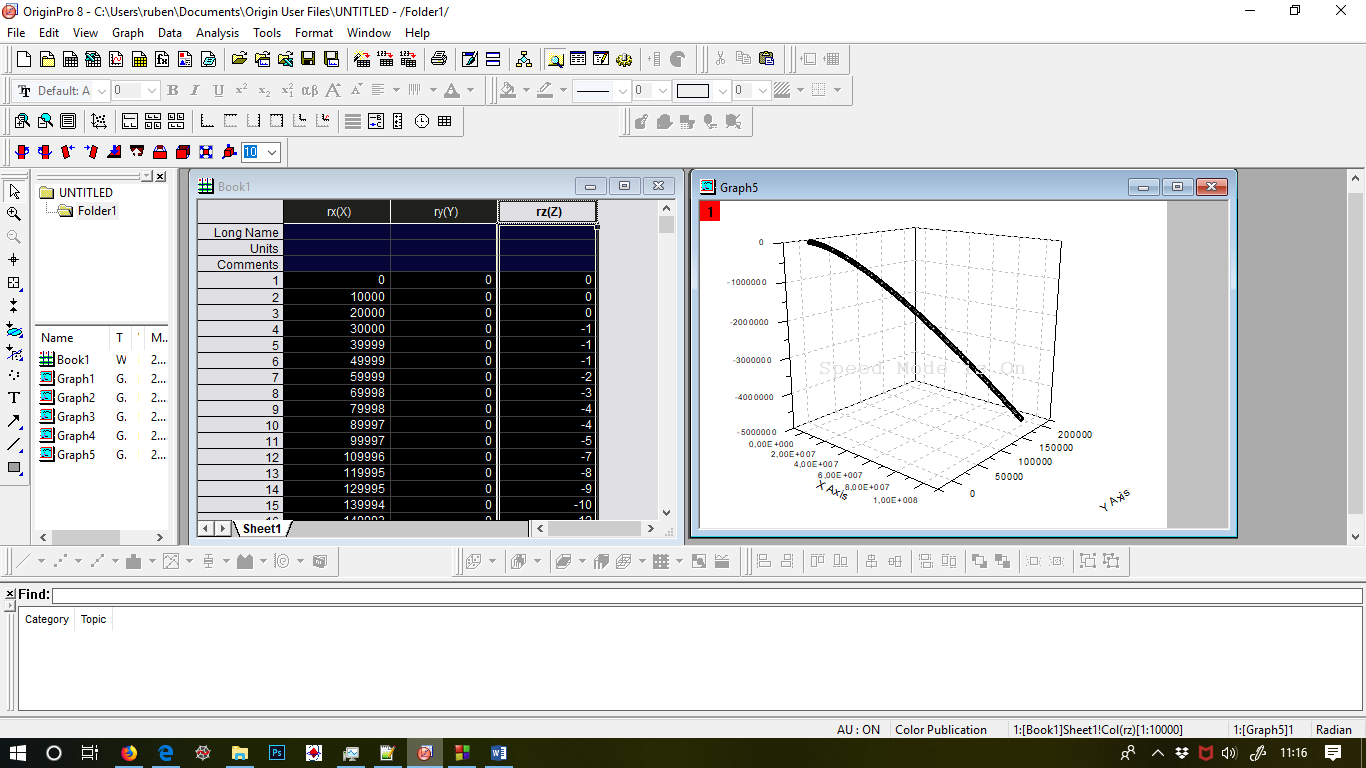
Como esperado, se não há componente de velocidade alguma para o produto vetorial, apenas a angular, o corpo cai em queda livre naturalmente.



Vox=0; Voy=10; Voz=10; || Wox=0; Woy=0; Woz=-100;



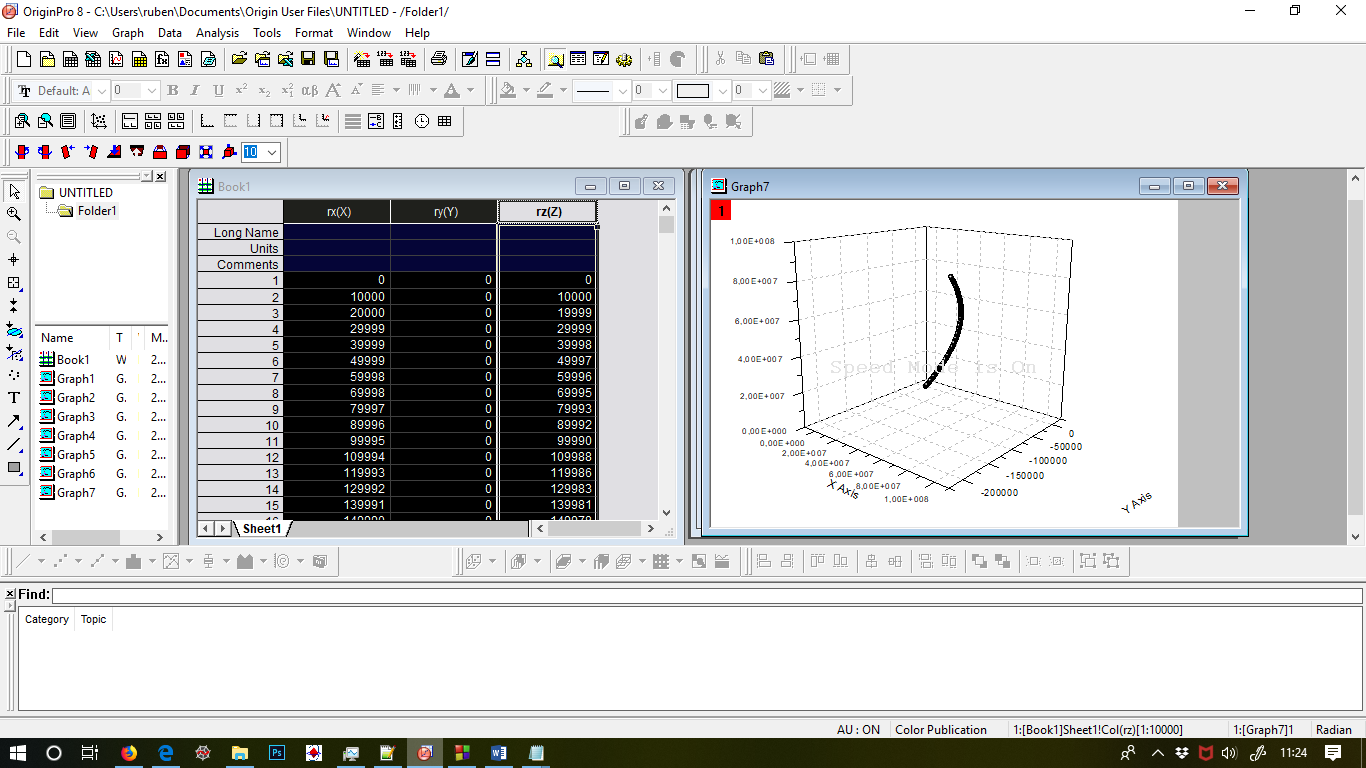
Vox=100; Voy=0; Voz=0; || Wox=0; Woy=0; Woz=-100;



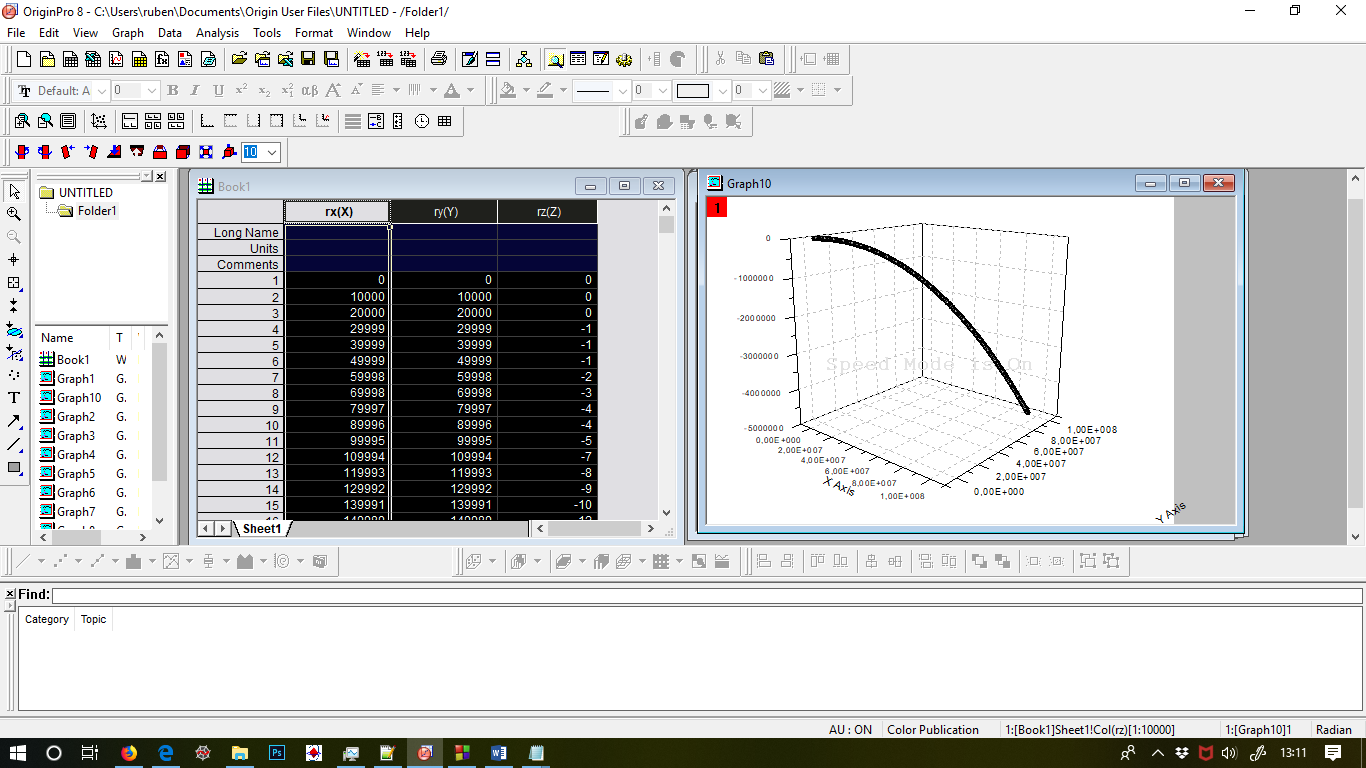
**Terceiro teste**

**(w||-g)**

Vox=100; Voy=0; Voz=100; || Wox=0; Woy=0; Woz=100;



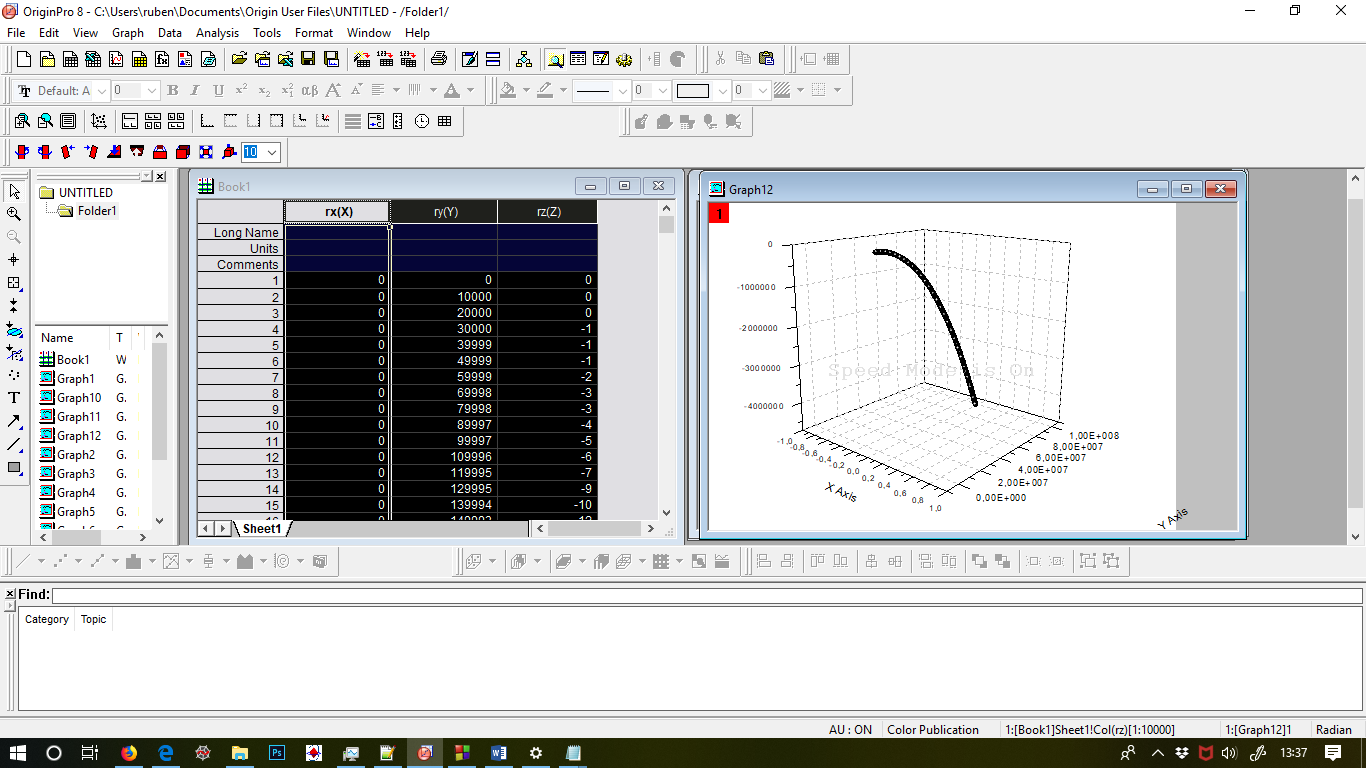
Vox=100; Voy=100; Voz=0; || Wox=0; Woy=0; Woz=100;



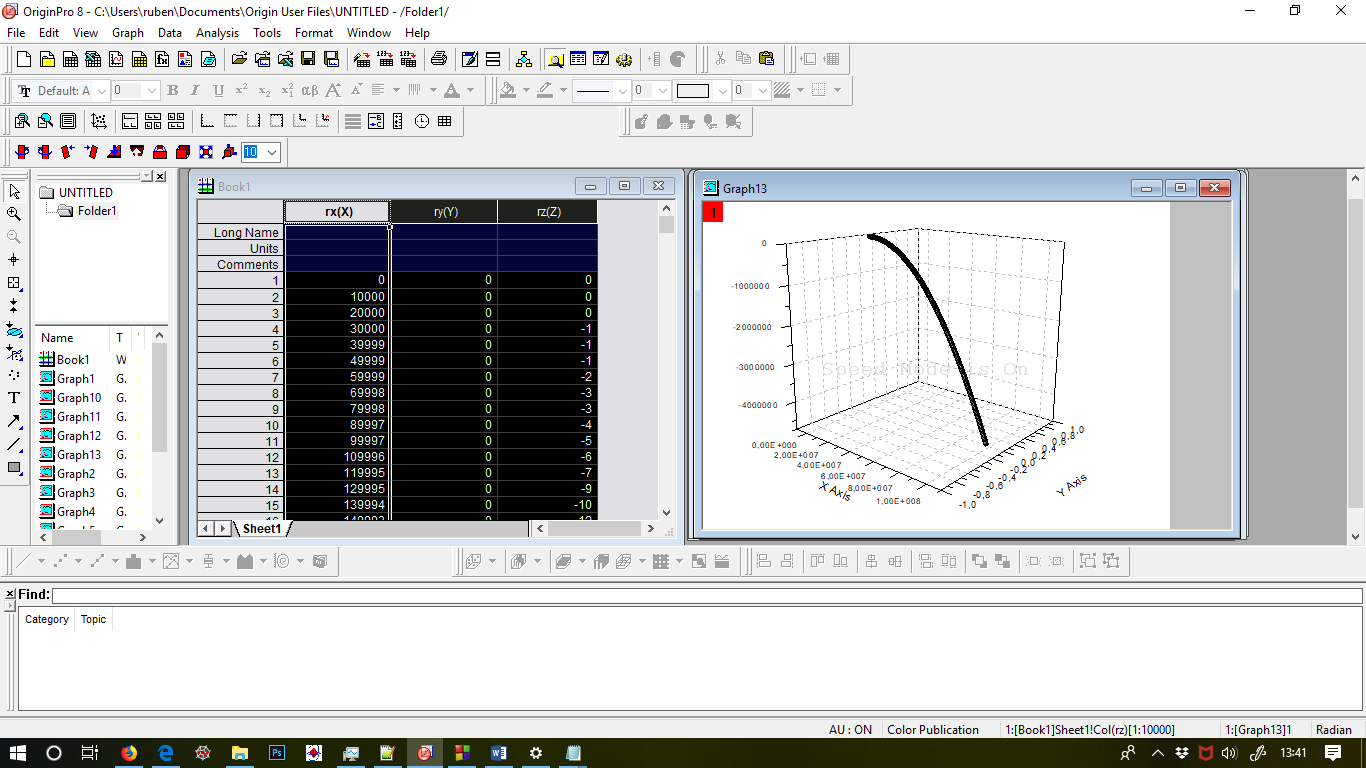
**Quarto teste**

**(w perpendicular ao plano definido por g e v0)**

Vox=0; Voy=100; Voz=0; || Wox=100; Woy=0; Woz=0;



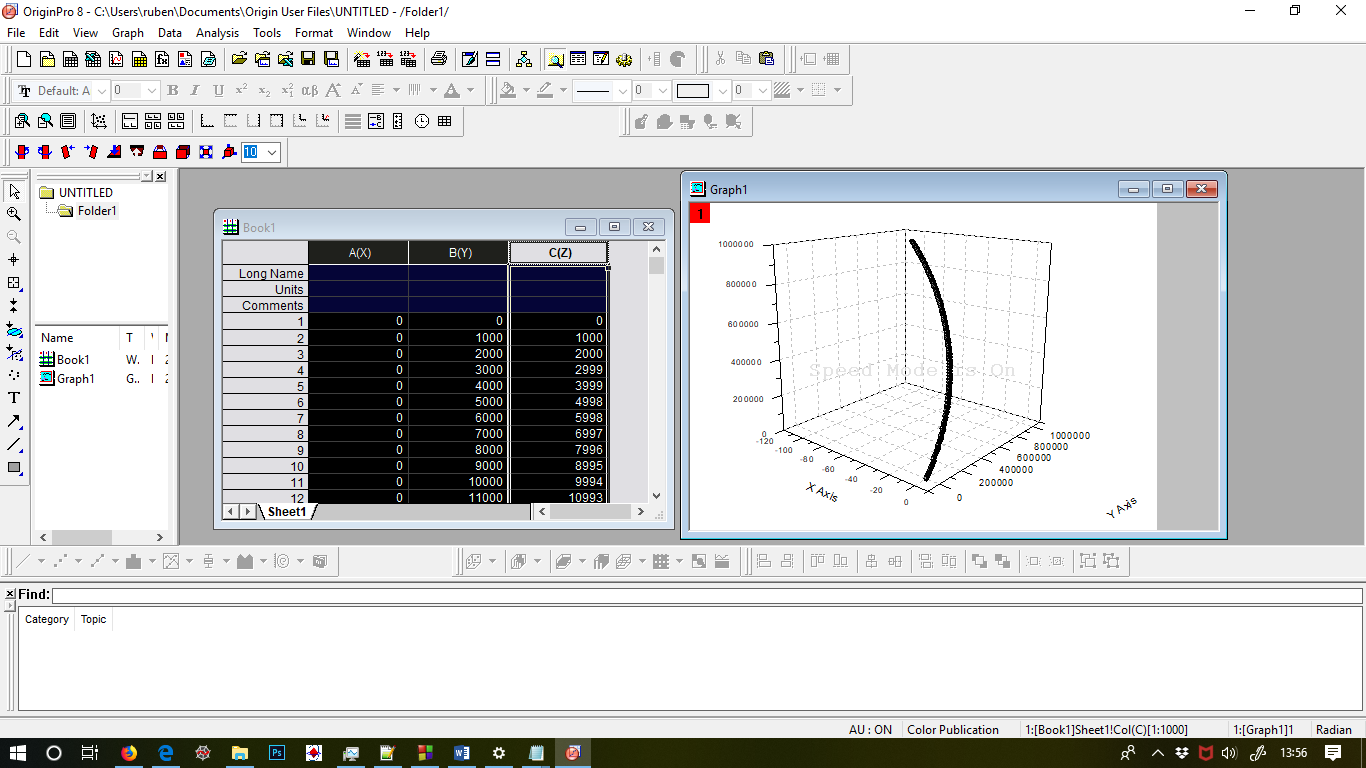
Vox=100; Voy=0; Voz=0; || Wox=0; Woy=-100; Woz=0;



**Quinto teste**

**(w arbitrário)**

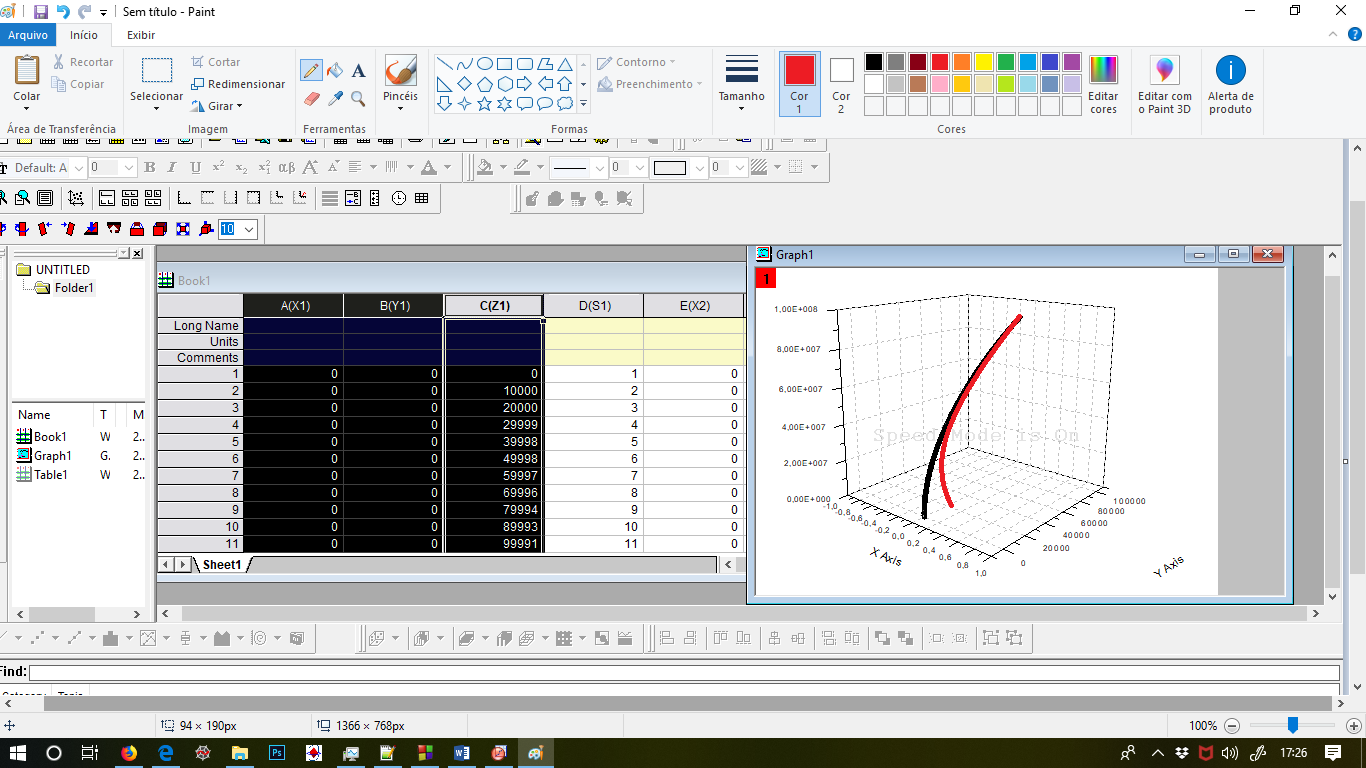
Vox=0; Voy=10; Voz=10; || Wox=25; Woy=50; Woz=100;



**Diferença gráfica: Força de Magnus Vs Parábola ideal**

**Preto – Parábola ideal**

**Vermelho – Curva da partícula sobre ação do efeito Magnus**



**Limite de trabalho feito em sala**

**Programa desenvolvido para conclusão das tarefas 4 e 5, ainda sobre o módulo 2:**

#include<stdlib.h>

#include<math.h>

#include<stdio.h>

int main(){

//Definindo variáveis de movimento

int n,i;

double rx,ry,rz, vx,vy,vz, Bx,By,Bz, Ex,Ey,Ez, dt=0.00001, m, q, K;

//criando e abrindo arquivos

FILE \*Rx\_lorentz;

FILE \*Ry\_lorentz;

FILE \*Rz\_lorentz;

Rx\_lorentz = fopen("eixo x (lorentz).txt","w+");

Ry\_lorentz = fopen("eixo y (lorentz).txt","w+");

Rz\_lorentz = fopen("eixo z (lorentz).txt","w+");

//obtendo valores do usuário

printf("Digite as componentes da velocidade inicial (x,y,z):\n");

scanf("%lf %lf %lf",&vx,&vy,&vz);

printf("Digite as componentes do Campo Magnetico inicial (x,y,z):\n");

scanf("%lf %lf %lf",&Bx,&By,&Bz);

printf("Digite as componentes do Campo Eletrico inicial (x,y,z):\n");

scanf("%lf %lf %lf",&Ex,&Ey,&Ez);

printf("Digite a massa da particula:\n ");

scanf("%lf",&m);

printf("Digite a carga da particula:\n ");

scanf("%lf",&q);

K=q/m;

printf("Digite o numero de interaçoes:\n ");

scanf("%d",&n);

//interações numéricas

for(i=0;i<n;i++){

//preenchendo arquivos

fprintf(Rx\_lorentz,"%lf\n", rx);

fprintf(Ry\_lorentz,"%lf\n", ry);

fprintf(Rz\_lorentz,"%lf\n", rz);

//calculando velocidades

vx= vx + (K\*(vy\*Bz - vz\*By)+Ex)\*dt;

vy= vy + (K\*(vx\*Bz - vz\*Bx)+Ey)\*dt;

vz= vz + (K\*(vx\*By - vy\*Bx)+Ez)\*dt;

//calculando posições

rx= rx + vx\*dt;

ry= ry + vy\*dt;

rz= rz + vz\*dt;

}

//fechando o arquivo

fclose(Rx\_lorentz);

fclose(Ry\_lorentz);

fclose(Rz\_lorentz);

printf("Os arquivos lorentz foram criados com sucesso.\n");

return 0;

}

**Plot do último gráfico com ajuste linear do origin: força de Lorentz com campos uniformes; a partícula oscila em um só plano**

