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;
```

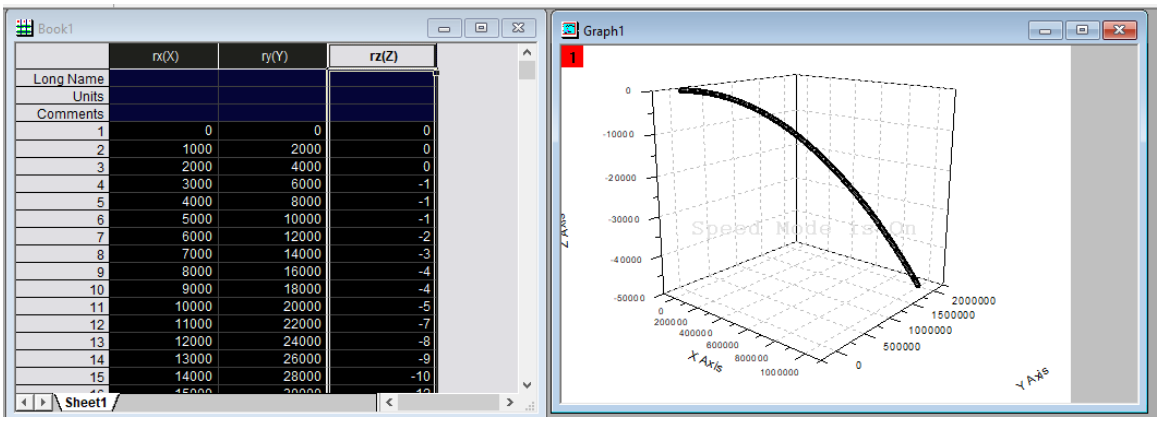
}

Gráficos gerados para conclusão da terceira tarefa:

Primeiro teste

(w=0)

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

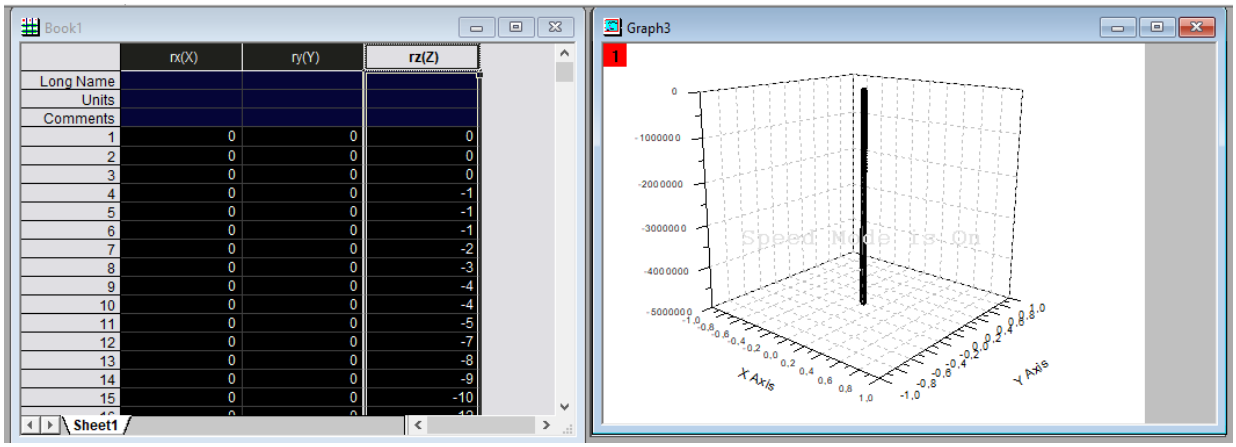


Segundo teste

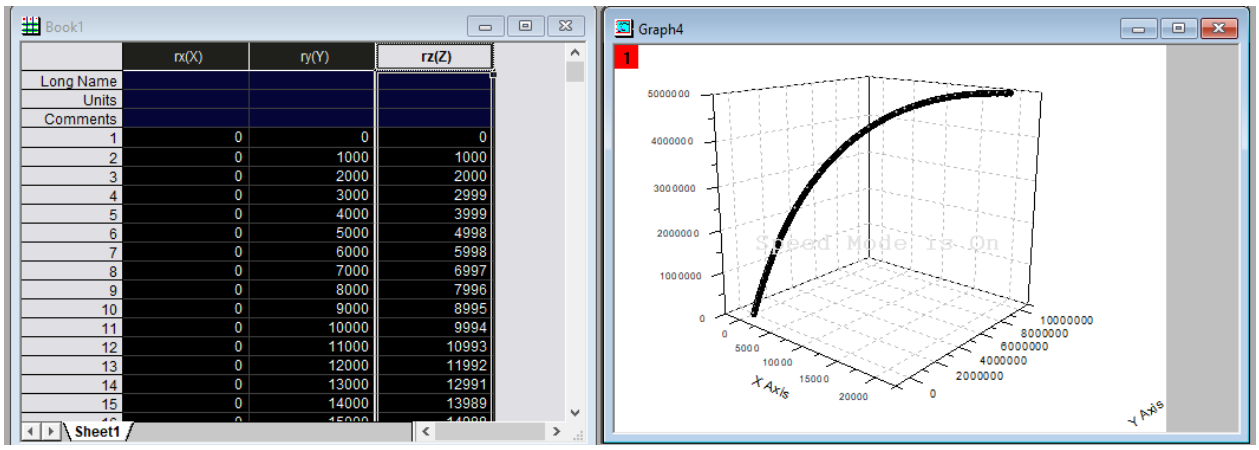
(w||g)

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

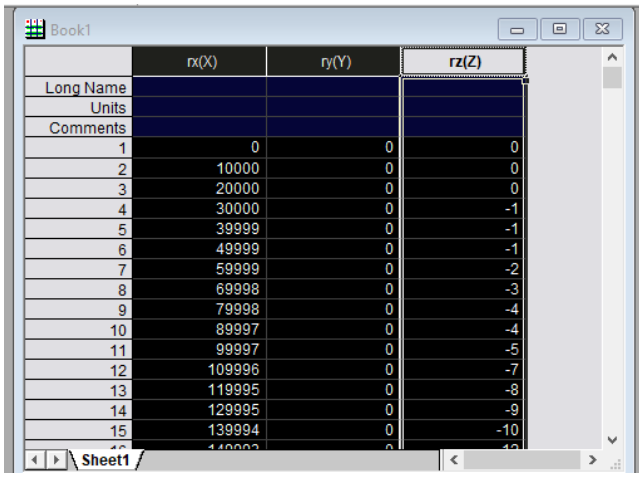
Como esperado, se não há componente de velocidade alguma para o produto vetorial, apenas a angular, o corpo cai em queda livre naturalmente.



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



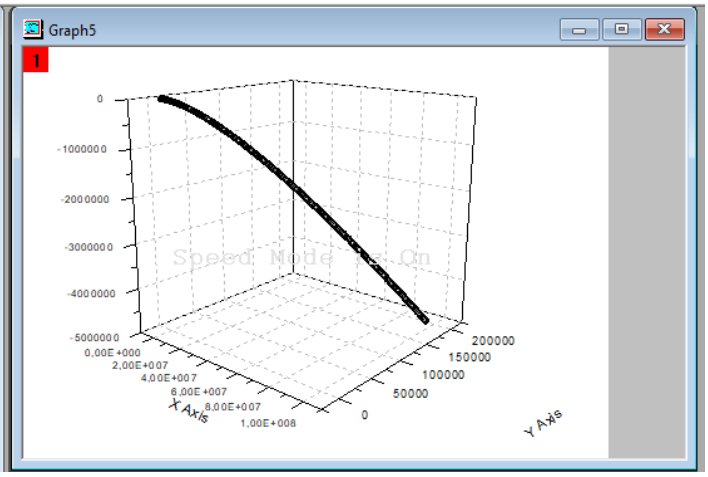
Vox=100; Voy=0; Voz=0; || Wox=0; Woy=0; Woz=-100;



Book1

	rx(X)	ry(Y)	rz(Z)
Long Name			
Units			
Comments			
1	0	0	0
2	10000	0	0
3	20000	0	0
4	30000	0	-1
5	39999	0	-1
6	49999	0	-1
7	59999	0	-2
8	69998	0	-3
9	79998	0	-4
10	89997	0	-4
11	99997	0	-5
12	109996	0	-7
13	119995	0	-8
14	129995	0	-9
15	139994	0	-10

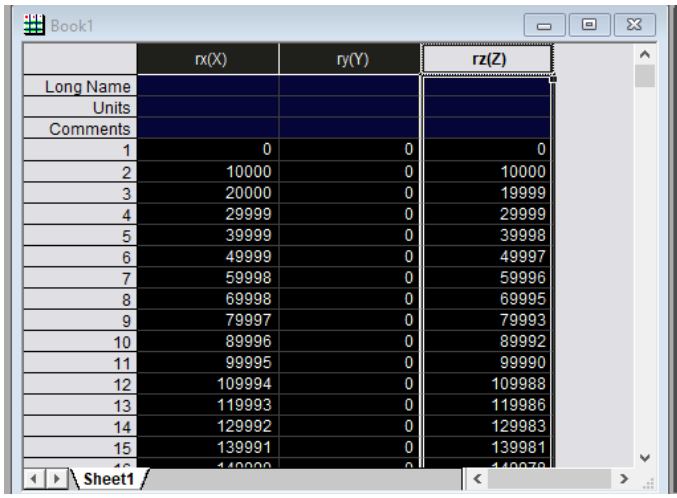
Sheet1



Terceiro teste

(w||-g)

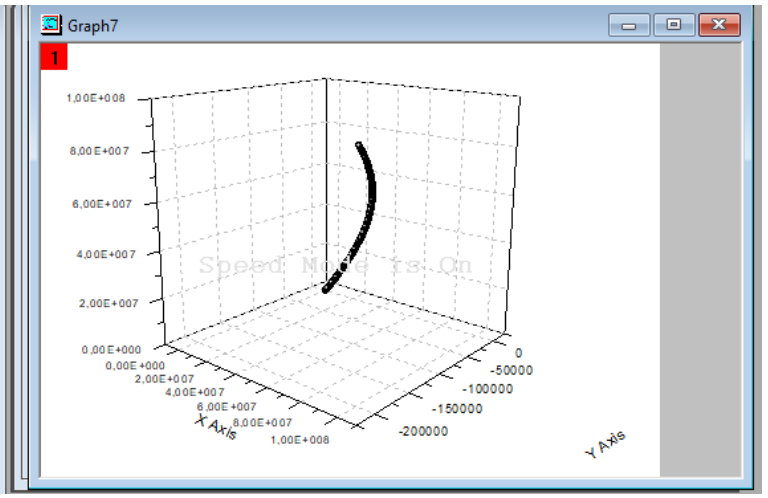
Vox=100; Voy=0; Voz=100; || Wox=0; Woy=0; Woz=100;



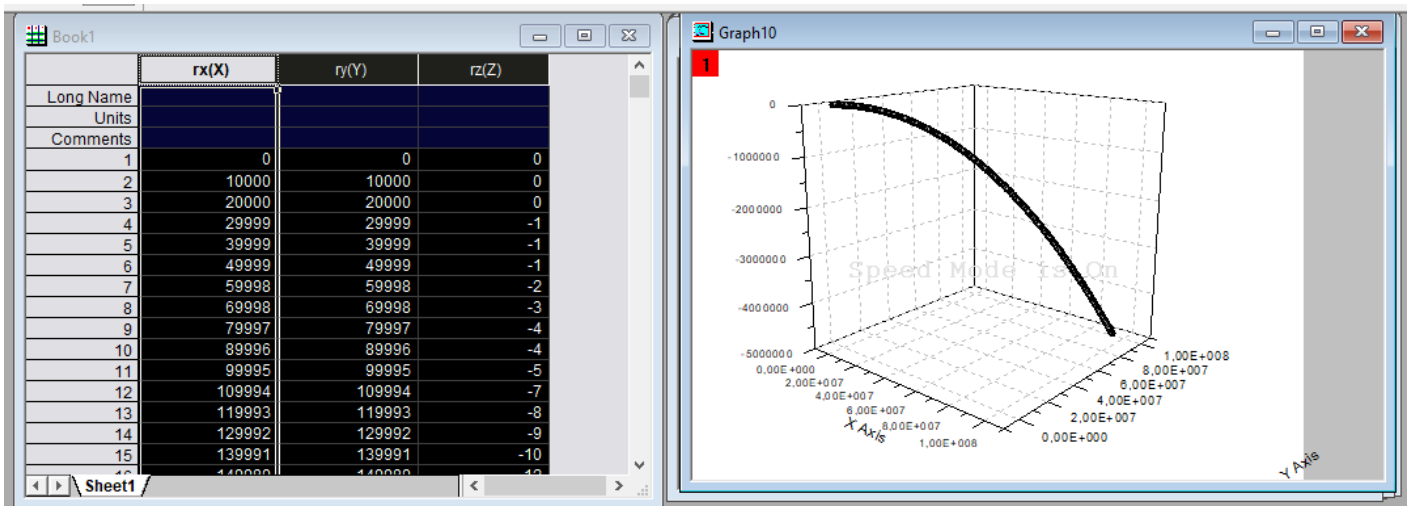
Book1

	rx(X)	ry(Y)	rz(Z)
Long Name			
Units			
Comments			
1	0	0	0
2	10000	0	10000
3	20000	0	19999
4	29999	0	29999
5	39999	0	39998
6	49999	0	49997
7	59998	0	59996
8	69998	0	69995
9	79997	0	79993
10	89996	0	89992
11	99995	0	99990
12	109994	0	109988
13	119993	0	119986
14	129992	0	129983
15	139991	0	139981

Sheet1



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

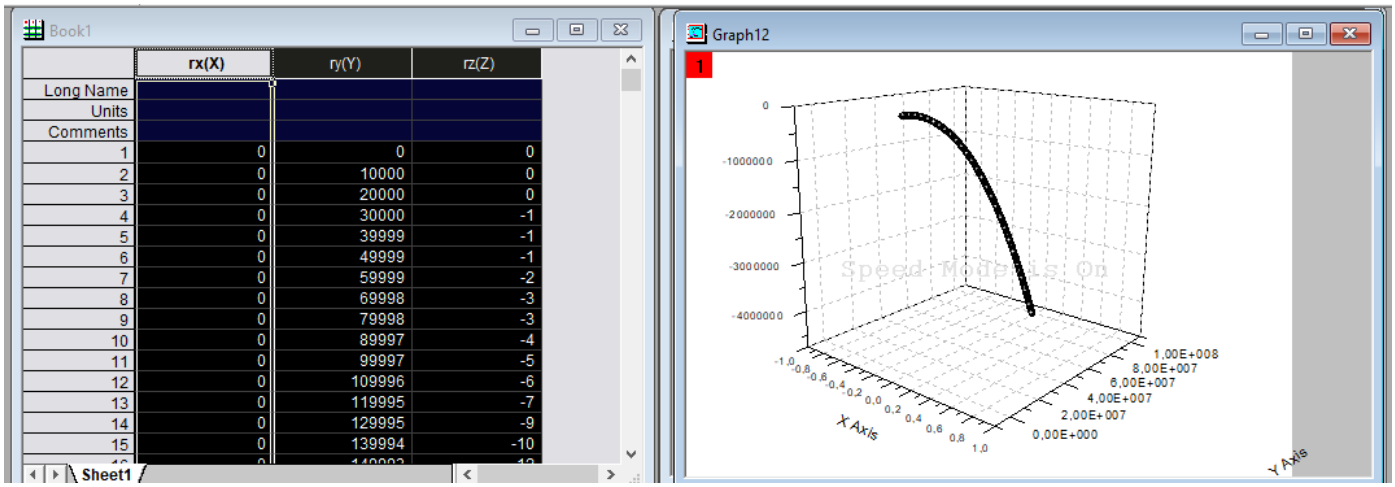


	rx(X)	ry(Y)	rz(Z)
Long Name			
Units			
Comments			
1	0	0	0
2	10000	10000	0
3	20000	20000	0
4	29999	29999	-1
5	39999	39999	-1
6	49999	49999	-1
7	59998	59998	-2
8	69998	69998	-3
9	79997	79997	-4
10	89996	89996	-4
11	99995	99995	-5
12	109994	109994	-7
13	119993	119993	-8
14	129992	129992	-9
15	139991	139991	-10

Quarto teste

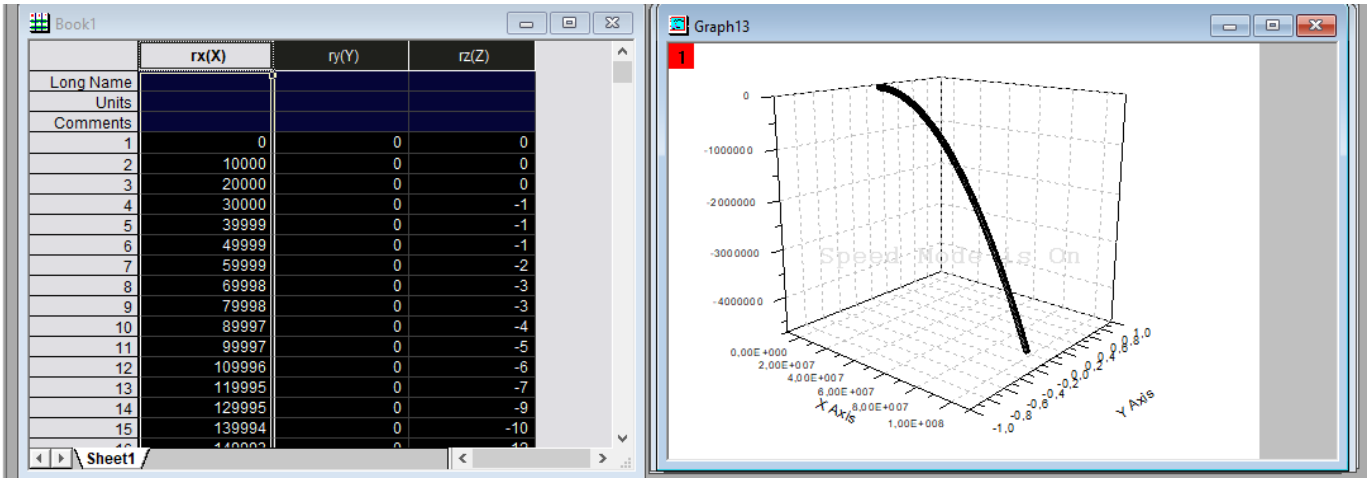
(w perpendicular ao plano definido por g e v0)

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



	rx(X)	ry(Y)	rz(Z)
Long Name			
Units			
Comments			
1	0	0	0
2	0	10000	0
3	0	20000	0
4	0	30000	-1
5	0	39999	-1
6	0	49999	-1
7	0	59999	-2
8	0	69998	-3
9	0	79998	-3
10	0	89997	-4
11	0	99997	-5
12	0	109996	-6
13	0	119995	-7
14	0	129995	-9
15	0	139994	-10

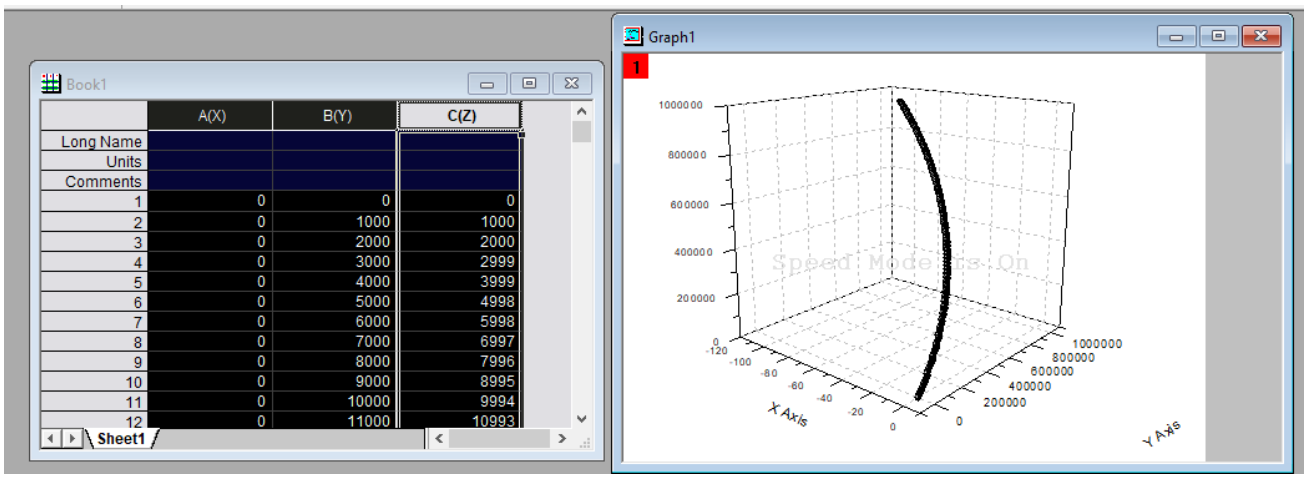
Vox=100; Voy=0; Voz=0; || Wox=0; Woy=-100; Woz=0;



Quinto teste

(w arbitrário)

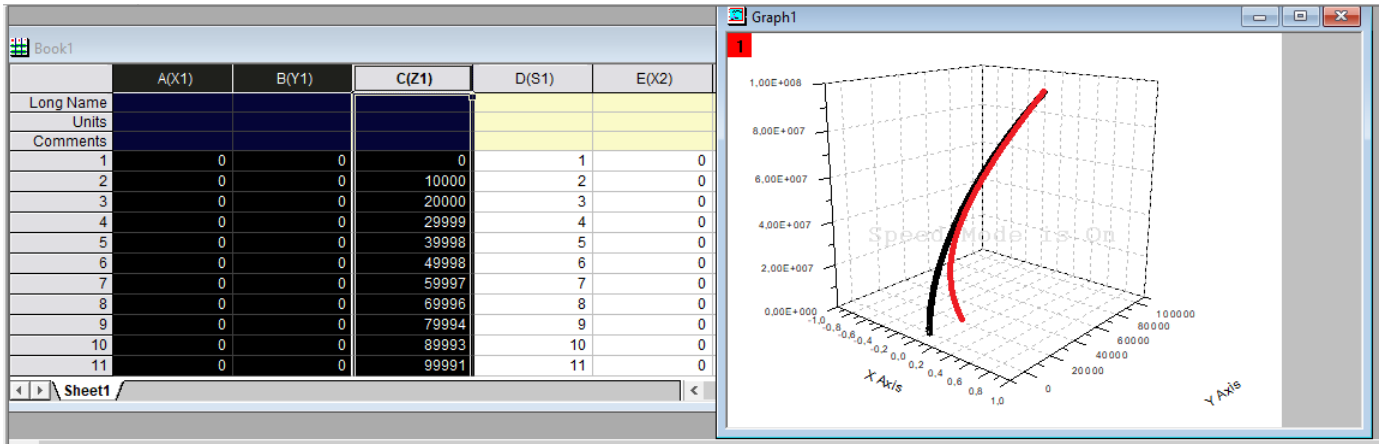
Vox=0; Voy=10; Voz=10; || Wox=25; Woy=50; Woz=100;



Diferença gráfica: Força de Magnus Vs Parábola ideal

Preto – Parábola ideal

Vermelho – Curva da partícula sobre ação do efeito Magnus



Book1

	A(X1)	B(Y1)	C(Z1)	D(S1)	E(X2)
Long Name					
Units					
Comments					
1	0	0	0	1	0
2	0	0	10000	2	0
3	0	0	20000	3	0
4	0	0	29999	4	0
5	0	0	39998	5	0
6	0	0	49998	6	0
7	0	0	59997	7	0
8	0	0	69996	8	0
9	0	0	79994	9	0
10	0	0	89993	10	0
11	0	0	99991	11	0

Sheet1

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