Revista Brasileira de Ensino de Física
https://doi.org/10.1590/1806-9126-RBEF-2024-0372

Material Suplementar para "Construção numérica de diagramas conformes em relatividade geral"

Apêndice A. Diagrama conforme numérico Schwarzschild

A.1. Código numérico dos valores de entrada

Ressaltamos que aqui constam os códigos para coordenadas de tempo avançado.

A.1.1. Região em que $r > 2M$

O que difere na versão desse código para o das coordenadas de tempo retardado é que as funções "Newton_FotonIngoing" e "FuncaoRaiz_Newton_FotonIngoing" se tornam "Newton_FotonOutgoing" e "FuncaoRaiz_Newton_FotonOutgoing" respectivamente. Além do mais, nele as funções usadas nas linhas 37 e 38 são invertidas (bem como nas linhas 49 e 50) e trocamos "GeodesicaSinalPositivo_coneFuturo"por "GeodesicaSinalNegativo_coneFuturo" e "GeodesicaSinalPositivo_conePassado" por "GeodesicaSinalNegativo_conePassado".

```
1  N = 2000;
2  passo_inicial = 0.001;
3  erro = 0.001;
4  #intervalo v coordenada
5  v1 = -300; #150
6  v2 = 600; #400
7  v = linspace(v1,v2,N+1);
8  #constantes espaciais
9  constanteEspacial = [2.06 2.2 2.4 2.5 2.6 3 6 150]; #vetor com constantes espaciais (r)
10 Cr = length(constanteEspacial);
11
12 #Encontrando as coordenadas para curvas de espaco constante
13 #Aqui serao guardados os pontos de r para geodesicas do cone futuro construidas com as
       trajetorias progredindo (indo de baixo para cima)
14 r1_geodesicaConeFuturo_espacoConstante = [];
15 r2_geodesicaConeFuturo_espacoConstante = [];
16 auxLinha_I_espacoConstante = 0;
17 auxColuna_I_espacoConstante = 0;
18 qtd_Colunas_I = []; #vetor que guarda a quantidade de colunas (nao vazias) em cada linha
19 #Aqui serao guardados os pontos de r para geodesicas do cone passado construidas com as
       trajetorias regredindo (indo de cima para baixo)
20 r1_geodesicaConePassado_espacoConstante = [];
21 r2_geodesicaConePassado_espacoConstante = [];
22 auxLinha_II_espacoConstante = 0;
23 auxColuna_II_espacoConstante = 0;
24 qtd_Colunas_II = []; #vetor que guarda a quantidade de colunas (nao vazias) em cada
       linha
25
26
27
28 for i = 1:Cr
29
30   if(constanteEspacial(i) > 2)
31
32     auxLinha_I_espacoConstante = auxLinha_I_espacoConstante + 1;
```

```
33      auxColuna_I_espacoConstante = 0;
34      for j = 1:length(v)
35        if(v(j) <= FuncaoRaiz_Newton_FotonIngoing(0,constanteEspacial(i)))
36          auxColuna_I_espacoConstante = auxColuna_I_espacoConstante + 1;
37          r1_geodesicaConeFuturo_espacoConstante(auxLinha_I_espacoConstante,
      auxColuna_I_espacoConstante) = Newton_FotonIngoing(v(j),constanteEspacial(i)); #
      geodesica ingoing
38          r2_geodesicaConeFuturo_espacoConstante(auxLinha_I_espacoConstante,
      auxColuna_I_espacoConstante) = GeodesicaSinalPositivo_coneFuturo(v(j),
      constanteEspacial(i),passo_inicial,erro); #geodesica outgoing
39        endif
40      endfor
41      qtd_Colunas_I(i) = auxColuna_I_espacoConstante;
42
43
44      auxLinha_II_espacoConstante = auxLinha_II_espacoConstante + 1;
45      auxColuna_II_espacoConstante = 0;
46      for j = 1:length(v)
47        if(v(j) >= FuncaoRaiz_Newton_FotonIngoing(0,constanteEspacial(i)))
48          auxColuna_II_espacoConstante = auxColuna_II_espacoConstante + 1;
49          r1_geodesicaConePassado_espacoConstante(auxLinha_II_espacoConstante,
      auxColuna_II_espacoConstante) = GeodesicaSinalPositivo_conePassado(v(j),
      constanteEspacial(i),passo_inicial,erro); #geodesica outgoing
50          r2_geodesicaConePassado_espacoConstante(auxLinha_II_espacoConstante,
      auxColuna_II_espacoConstante) = Newton_FotonIngoing(v(j),constanteEspacial(i)); #
      geodesica ingoing
51        endif
52      endfor
53      qtd_Colunas_II(i) = auxColuna_II_espacoConstante;
54
55    endif
56
57  endfor
58
59
60  #Guardando as matrizes, vetores e valores obtidos como dados em arquivo txt
61  arqbin_r1_coneFut_espacoConst = fopen("
      FotonIngoing_rMaiorDoQue2_matriz_r1_geodesicaConeFuturo_espacoConstante.txt","w");
62  for i = 1:size(r1_geodesicaConeFuturo_espacoConstante,1)
63    fprintf(arqbin_r1_coneFut_espacoConst,'%.10f ',r1_geodesicaConeFuturo_espacoConstante(
      i,:));
64    fprintf(arqbin_r1_coneFut_espacoConst,'\n');
65  endfor
66  fclose(arqbin_r1_coneFut_espacoConst);
67
68  arqbin_r2_coneFut_espacoConst = fopen("
      FotonIngoing_rMaiorDoQue2_matriz_r2_geodesicaConeFuturo_espacoConstante.txt","w");
69  for i = 1:size(r2_geodesicaConeFuturo_espacoConstante,1)
70    fprintf(arqbin_r2_coneFut_espacoConst,'%.10f ',r2_geodesicaConeFuturo_espacoConstante(
      i,:));
71    fprintf(arqbin_r2_coneFut_espacoConst,'\n');
72  endfor
73  fclose(arqbin_r2_coneFut_espacoConst);
74
75  arqbin_r1_conePas_espacoConst = fopen("
      FotonIngoing_rMaiorDoQue2_matriz_r1_geodesicaConePassado_espacoConstante.txt","w");
76  for i = 1:size(r1_geodesicaConePassado_espacoConstante,1)
77    fprintf(arqbin_r1_conePas_espacoConst,'%.10f ',r1_geodesicaConePassado_espacoConstante
      (i,:));
78    fprintf(arqbin_r1_conePas_espacoConst,'\n');
79  endfor
80  fclose(arqbin_r1_conePas_espacoConst);
```

```
81
82 arqbin_r2_conePas_espacoConst = fopen("
       FotonIngoing_rMaiorDoQue2_matriz_r2_geodesicaConePassado_espacoConstante.txt","w");
83 for i = 1:size(r2_geodesicaConePassado_espacoConstante,1)
84   fprintf(arqbin_r2_conePas_espacoConst,'%.10f ',r2_geodesicaConePassado_espacoConstante
       (i,:));
85   fprintf(arqbin_r2_conePas_espacoConst,'\n');
86 endfor
87 fclose(arqbin_r2_conePas_espacoConst);
88
89 arqbin_auxLinha_I_espacoConst = fopen("
       FotonIngoing_rMaiorDoQue2_valor_auxLinha_I_espacoConstante.txt","w");
90 for i = 1:size(auxLinha_I_espacoConstante,1)
91   fprintf(arqbin_auxLinha_I_espacoConst,'%.10f ',auxLinha_I_espacoConstante(i,:));
92   fprintf(arqbin_auxLinha_I_espacoConst,'\n');
93 endfor
94 fclose(arqbin_auxLinha_I_espacoConst);
95
96 arqbin_qtd_Colunas_I = fopen("FotonIngoing_rMaiorDoQue2_vetor_qtd_Colunas_I.txt","w");
97 for i = 1:size(qtd_Colunas_I,1)
98   fprintf(arqbin_qtd_Colunas_I,'%.10f ',qtd_Colunas_I(i,:));
99   fprintf(arqbin_qtd_Colunas_I,'\n');
100 endfor
101 fclose(arqbin_qtd_Colunas_I);
102
103 arqbin_auxLinha_II_espacoConst = fopen("
       FotonIngoing_rMaiorDoQue2_valor_auxLinha_II_espacoConstante.txt","w");
104 for i = 1:size(auxLinha_II_espacoConstante,1)
105   fprintf(arqbin_auxLinha_II_espacoConst,'%.10f ',auxLinha_II_espacoConstante(i,:));
106   fprintf(arqbin_auxLinha_II_espacoConst,'\n');
107 endfor
108 fclose(arqbin_auxLinha_II_espacoConst);
109
110 arqbin_qtd_Colunas_II = fopen("FotonIngoing_rMaiorDoQue2_vetor_qtd_Colunas_II.txt","w");
111 for i = 1:size(qtd_Colunas_II,1)
112   fprintf(arqbin_qtd_Colunas_II,'%.10f ',qtd_Colunas_II(i,:));
113   fprintf(arqbin_qtd_Colunas_II,'\n');
114 endfor
115 fclose(arqbin_qtd_Colunas_II);
```

A.1.2. Região em que $r < 2M$

O que difere na versão desse código para o das coordenadas de tempo retardado é que nele são construídas as geodésicas que definem o cone de luz futuro. Sendo assim as funções "FuncaoRaiz_Newton_FotonIngoing", "Newton_FotonIngoing" e "GeodesicaSinalPositivo_conePassado" se tornam "FuncaoRaiz_Newton_FotonOutgoing", "Newton_FotonOutgoing"e "GeodesicaSinalNegativo_coneFuturo". Além do mais, na linha 27, temos $u <=$ FuncaoRaiz_Newton_FotonOutgoing.

```
1 N = 2600;
2 passo_inicial = 0.001;
3 erro = 0.001;
4 #intervalo v coordenada
5 v1 = -250; #150
6 v2 = 300; #400
7 v = linspace(v1,v2,N+1);
8 #constantes espaciais
9 constanteEspacial = [0.002 0.4 1.2 1.6 1.98]; #vetor com constantes espaciais (r)
10 Cr = length(constanteEspacial);
11
12 #Encontrando as coordenadas para curvas de espaco constante
13 #Aqui serao guardados os pontos de r para geodesicas do cone passado construidas com as
       trajetorias regredindo (indo de cima para baixo)
14 r1_geodesicaConePassado_espacoConstante = [];
```

```
r2_geodesicaConePassado_espacoConstante = [];
auxLinha_espacoConstante = 0;
auxColuna_espacoConstante = 0;
qtd_Colunas = []; #vetor que guarda a quantidade de colunas (noo vazias) em cada linha

for i = 1:Cr

  if(constanteEspacial(i) < 2)

    auxLinha_espacoConstante = auxLinha_espacoConstante + 1;
    auxColuna_espacoConstante = 0;
    for j = 1:length(v)
      if(v(j) >= FuncaoRaiz_Newton_FotonIngoing(0,constanteEspacial(i)))
        auxColuna_espacoConstante = auxColuna_espacoConstante + 1;
        r1_geodesicaConePassado_espacoConstante(auxLinha_espacoConstante,
    auxColuna_espacoConstante) = GeodesicaSinalPositivo_conePassado(v(j),
    constanteEspacial(i),passo_inicial,erro) #geodesica outgoing
        r2_geodesicaConePassado_espacoConstante(auxLinha_espacoConstante,
    auxColuna_espacoConstante) = Newton_FotonIngoing(v(j),constanteEspacial(i)) #
    geodesica ingoing
      endif
    endfor
    qtd_Colunas(i) = auxColuna_espacoConstante;

  endif

endfor

#Guardando as matrizes, vetores e valores obtidos como dados em arquivo txt
arqbin_r1_conePas_espacoConst = fopen("
    FotonIngoing_rMenorDoQue2_matriz_r1_geodesicaConePassado_espacoConstante.txt","w");
for i = 1:size(r1_geodesicaConePassado_espacoConstante,1)
  fprintf(arqbin_r1_conePas_espacoConst,'%.10f ',r1_geodesicaConePassado_espacoConstante
    (i,:));
  fprintf(arqbin_r1_conePas_espacoConst,'\n');
endfor
fclose(arqbin_r1_conePas_espacoConst);

arqbin_r2_conePas_espacoConst = fopen("
    FotonIngoing_rMenorDoQue2_matriz_r2_geodesicaConePassado_espacoConstante.txt","w");
for i = 1:size(r2_geodesicaConePassado_espacoConstante,1)
  fprintf(arqbin_r2_conePas_espacoConst,'%.10f ',r2_geodesicaConePassado_espacoConstante
    (i,:));
  fprintf(arqbin_r2_conePas_espacoConst,'\n');
endfor
fclose(arqbin_r2_conePas_espacoConst);

arqbin_auxLinha_espacoConst = fopen("
    FotonIngoing_rMenorDoQue2_valor_auxLinha_espacoConstante.txt","w");
for i = 1:size(auxLinha_espacoConstante,1)
  fprintf(arqbin_auxLinha_espacoConst,'%.10f ',auxLinha_espacoConstante(i,:));
  fprintf(arqbin_auxLinha_espacoConst,'\n');
endfor
fclose(arqbin_auxLinha_espacoConst);

arqbin_qtd_Colunas = fopen("FotonIngoing_rMenorDoQue2_vetor_qtd_Colunas.txt","w");
for i = 1:size(qtd_Colunas,1)
  fprintf(arqbin_qtd_Colunas,'%.10f ',qtd_Colunas(i,:));
  fprintf(arqbin_qtd_Colunas,'\n');
endfor
fclose(arqbin_qtd_Colunas);
```

A.2. Geodésicas

A.2.1. Sinal positivo

Por sinal positivo nos referimos ao fato de que a geodésica nula que será construída é definida pela equação diferencial maior do que zero (equação **??**). Sua construção se dá através do método de Runge-Kutta de 4ª ordem adaptativo, onde "RungeKutta_DerivadaPositiva" é o método mais comumente conhecido sem adaptação que utiliza dessa equação diferencial. Aqui trazemos o código que constrói essa geodésica do cone de luz futuro: "GeodesicaSinalPositivo_coneFuturo". O que difere para o código da mesma geodésica positiva do cone de luz passado é que nele, na linha 12 com o laço, temos FuncaoRaiz_Newton_FotonIngoing$(0, y(i)) <= x(i)$. Além do mais, os limites do passo são negativos e, portanto, nas linhas 29 e 40 temos o módulo desses valores, assim como de h.

```
1  function [y1] = GeodesicaSinalPositivo_coneFuturo(x0,y0,passo_inicial,erro_relativo)
2    #A funcaoo constroi numericamente a geodesica com sinal positivo do cone futuro
       fazendo o caminho progredingo
3
4    y = [(y0)]; #vetor que guarda todos os valores tracados numericamente de y
5    x = [(x0)]; #vetor que guarda todos os valores de x
6    h = passo_inicial;
7    h_min = 0.00000001; #menor valor que o passo adaptativo pode assumir
8    h_max = 0.1; #maior valor que o passo adaptativo pode assumir
9    y_teste = []; #vetor usado na logica do passo adaptativo
10   i = 1;
11
12   while(FuncaoRaiz_Newton_FotonIngoing(0,y(i)) >= x(i)) #laco de repeticao para enquanto
       os valores estiverem no mesmo quadrado, ou seja, antes de cruzar a superficie
13   h_val = [h (h/2)]; #vetor usado na logica do passo adaptativo
14
15     for j = 1:2
16       if(j == 1)
17         y_teste(j) = RungeKutta_DerivadaPositiva(x(i),y(i),h_val(j));
18       elseif(j == 2)
19         y_intermed = RungeKutta_DerivadaPositiva(x(i),y(i),h_val(j));
20         x_intermed = x(i) + h_val(j);
21         y_teste(j) = RungeKutta_DerivadaPositiva(x_intermed,y_intermed,h_val(j));
22       endif
23     endfor
24
25     dif = abs(y_teste(2) - y_teste(1));   #diferenca entre os dois valores do mesmo
       ponto para comparacao com o erro
26     e = abs(erro_relativo*y_teste(1));    #erro
27
28     if(dif > e)      #condicao para caso a diferenca entre os valores seja maior do que
       o erro permitido
29       if(h > (2*h_min))
30         h = h/2;
31         i = i;
32       else
33         h = h_min;
34         y(i+1) = RungeKutta_DerivadaPositiva(x(i),y(i),h);
35         x(i+1) = x(i) + h;
36         h = 2*h;
37         i = i + 1;
38       endif
39     else    #condicao para caso a diferenca entre os valores seja menor do que o erro
       permitido, ou seja, dentro do limite
40       if(h >= h_max)
41         h = h_max;
42         y(i+1) = RungeKutta_DerivadaPositiva(x(i),y(i),h);
43         x(i+1) = x(i) + h;
44         i = i + 1;
45       else
46         y(i+1) = y_teste(1);
```

```
47          x(i+1) = x(i) + h;
48          h = 2*h;
49          i = i + 1;
50        endif
51      endif
52
53    endwhile
54
55    C = length(y);
56    y1 = (y(C) + y(C-1))./2;
57
58    return;
59  endfunction
```

A.2.2. Sinal negativo

Por sinal negativo nos referimos ao fato de que a geodésica nula que será construída é definida pela equação diferencial menor do que zero (equação **??**). Sua construção se dá através do método de Runge-Kutta de 4ª ordem adaptativo, onde "RungeKutta_DerivadaNegativa" é o método mais comumente conhecido sem adaptação que utiliza dessa equação diferencial. Aqui trazemos o código que constrói essa geodésica do cone de luz futuro: "GeodesicaSinalNegativo_coneFuturo". O que difere para o código da mesma geodésica negativa do cone de luz passado é que nele, na linha 12 com o laço, temos FuncaoRaiz_Newton_FotonOutgoing$(0, y(i)) <= x(i)$. Além do mais, os limites do passo e o próprio passo dessa vez são negativos e, portanto, nas linhas 29 e 40 temos o módulo desses valores.

```
1  function [y1] = GeodesicaSinalNegativo_coneFuturo(x0,y0,passo_inicial,erro_relativo)
2    #A funcao constroi numericamente a geodesica com sinal negativo do cone futuro fazendo
       o caminho progredindo
3
4    y = [(y0)]; #vetor que guarda todos os valores tracados numericamente de y
5    x = [(x0)]; #vetor que guarda todos os valores de x
6    h = passo_inicial;
7    h_min = 0.00000001; #menor valor que o passo adaptativo pode assumir
8    h_max = 0.1; #maior valor que o passo adaptativo pode assumir
9    y_teste = []; #vetor usado na logica do passo adaptativo
10   i = 1;
11
12   while(FuncaoRaiz_Newton_FotonOutgoing(0,y(i)) >= x(i)) #laco de repeticao para
       enquanto os valores estiverem no mesmo quadrado, ou seja, antes de cruzar a
       superficie
13   h_val = [h (h/2)]; #vetor usado na logica do passo adaptativo
14
15     for j = 1:2
16       if(j == 1)
17         y_teste(j) = RungeKutta_DerivadaNegativa(x(i),y(i),h_val(j));
18       elseif(j == 2)
19         y_intermed = RungeKutta_DerivadaNegativa(x(i),y(i),h_val(j));
20         x_intermed = x(i) + h_val(j);
21         y_teste(j) = RungeKutta_DerivadaNegativa(x_intermed,y_intermed,h_val(j));
22       endif
23     endfor
24
25     dif = abs(y_teste(2) - y_teste(1));   #diferenca entre os dois valores do mesmo
       ponto para comparacao com o erro
26     e = abs(erro_relativo*y_teste(1));    #erro
27
28     if(dif > e)      #condicao para caso a diferenca entre os valores seja maior do que
       o erro permitido
29       if(h > (2*h_min))
30         h = h/2;
31         i = i;
32       else
```

```
        h = h_min;
        y(i+1) = RungeKutta_DerivadaNegativa(x(i),y(i),h);
        x(i+1) = x(i) + h;
        h = 2*h;
        i = i + 1;
      endif
    else    #condicao para caso a diferenca entre os valores seja menor do que o erro permitido, ou seja, dentro do limite
      if(h >= h_max)
        h = h_max;
        y(i+1) = RungeKutta_DerivadaNegativa(x(i),y(i),h);
        x(i+1) = x(i) + h;
        i = i + 1;
      else
        y(i+1) = y_teste(1);
        x(i+1) = x(i) + h;
        h = 2*h;
        i = i + 1;
      endif
    endif

  endwhile

  C = length(y);
  y1 = (y(C) + y(C-1))./2;

  return;
endfunction
```

A.3. Método de Newton–Raphson

Sendo o objetivo do método, aqui encontramos o valor r onde $f(r) = v(r) - C = 0$, onde C é uma constante, para coordenadas de tempo avançado.

A.3.1. Interação que encontra a raiz

No código de entrada de variáveis, essa função é chamada "Newton_FotonIngoing". O que difere na versão desse código para o das coordenadas de tempo retardado é que as funções "FuncaoRaiz_Newton_FotonIngoing", "DerivadaFuncaoRaiz_Newton_FotonIngoing" e "N_FotonIngoing" se tornam "FuncaoRaiz_Newton_FotonOutgoing", "DerivadaFuncaoRaiz_Newton_FotonOutgoing" e "N_FotonOutgoing".

```
function [x] = Newton_FotonIngoing(c,x)
  #c = constante de v
  #x = valor de r (espaco) - relacionado com a funcao v(x) = v(r)
  a = x;
  b = c;

  x0 = ((b - a))/2;
  if (x0 == 2)
    x0 = 3;
  elseif(x0 == 0)
    x0 = 5;
  else
    x0 = x0;
  endif

  x1 = x0 - ((FuncaoRaiz_Newton_FotonIngoing(c,x0))./(
    DerivadaFuncaoRaiz_Newton_FotonIngoing(c,x0)));
  e = 10^(-6); #precisao
  x1 = N_FotonIngoing(x0,x1,e,c);

  z = 0;
```

```
21
22    while(x1 < 2)
23      if(x0 < 2)
24        x0 = 2.4 - z;
25        x1 = x0 - ((FuncaoRaiz_Newton_FotonIngoing(c,x0))./(
      DerivadaFuncaoRaiz_Newton_FotonIngoing(c,x0)));
26        x1 = N_FotonIngoing(x0,x1,e,c);
27        z = z + 0.01;
28      else
29        x0 = x0 - 0.01;
30        x1 = x0 - ((FuncaoRaiz_Newton_FotonIngoing(c,x0))./(
      DerivadaFuncaoRaiz_Newton_FotonIngoing(c,x0)));
31        x1 = N_FotonIngoing(x0,x1,e,c);
32      endif
33    endwhile
34
35    x = x1;
36
37    return;
38  endfunction
```

A.3.2. Interação para cálculo do valor seguinte de r

O que difere na versão desse código para o das coordenadas de tempo retardado é que são substituídas as funções respectivas Outgoing.

```
1  function [x1] = N_FotonIngoing(x0,x1,e,c)
2    while (abs(x1 - x0) > e)
3      x0 = x1;
4      x1 = x0 - ((FuncaoRaiz_Newton_FotonIngoing(c,x0))./(
       DerivadaFuncaoRaiz_Newton_FotonIngoing(c,x0)));
5    endwhile
6     x1 = x1;
7    return;
8  endfunction
```

A.3.3. Definição da função

O que difere na versão desse código para o das coordenadas de tempo retardado é que nele todos os termos após a igualdade são negativos e não somente c. As funções "DerivadaFuncaoRaiz_Newton_FotonInggoing" e "DerivadaFuncaoRaiz_Newton_FotonOutgoing" são simplesmente as derivadas dessas equações com relação a r (ou x no código).

```
1  function [v] = FuncaoRaiz_Newton_FotonIngoing(c,x)
2    v = x + 2*log(abs((x/2) - 1)) - c;
3    return;
4  endfunction
```

A.4. Plotagem

A.4.1. Região em que $r > 2M$

O código que segue tem como principal função plotar o diagrama da região $r > 2M$ coberta em coordenadas avançadas. O que difere na versão desse código para o das coordenadas de tempo retardado é que nele entramos com os valores encontrados no respectivo código das variáveis de entrada.

```
1  #Importando os dados dos arquivos txt
2  r1_geodesicaConeFuturo_espacoConstante = [];
3  i = 1;
4  arqbin_r1_coneFut_espacoConst = fopen("
       FotonIngoing_rMaiorDoQue2_matriz_r1_geodesicaConeFuturo_espacoConstante.txt","r");
```

```
while(!feof(arqbin_r1_coneFut_espacoConst))
  linha=fgets(arqbin_r1_coneFut_espacoConst);
  r1_geodesicaConeFuturo_espacoConstante(i,:)=str2num(linha);
  i = i + 1;
endwhile
fclose(arqbin_r1_coneFut_espacoConst);

r2_geodesicaConeFuturo_espacoConstante = [];
i = 1;
arqbin_r2_coneFut_espacoConst = fopen("
    FotonIngoing_rMaiorDoQue2_matriz_r2_geodesicaConeFuturo_espacoConstante.txt","r");
while(!feof(arqbin_r2_coneFut_espacoConst))
  linha=fgets(arqbin_r2_coneFut_espacoConst);
  r2_geodesicaConeFuturo_espacoConstante(i,:)=str2num(linha);
  i = i + 1;
endwhile
fclose(arqbin_r2_coneFut_espacoConst);

r1_geodesicaConePassado_espacoConstante = [];
i = 1;
arqbin_r1_conePas_espacoConst = fopen("
    FotonIngoing_rMaiorDoQue2_matriz_r1_geodesicaConePassado_espacoConstante.txt","r");
while(!feof(arqbin_r1_conePas_espacoConst))
  linha=fgets(arqbin_r1_conePas_espacoConst);
  r1_geodesicaConePassado_espacoConstante(i,:)=str2num(linha);
  i = i + 1;
endwhile
fclose(arqbin_r1_conePas_espacoConst);

r2_geodesicaConePassado_espacoConstante = [];
i = 1;
arqbin_r2_conePas_espacoConst = fopen("
    FotonIngoing_rMaiorDoQue2_matriz_r2_geodesicaConePassado_espacoConstante.txt","r");
while(!feof(arqbin_r2_conePas_espacoConst))
  linha=fgets(arqbin_r2_conePas_espacoConst);
  r2_geodesicaConePassado_espacoConstante(i,:)=str2num(linha);
  i = i + 1;
endwhile
fclose(arqbin_r2_conePas_espacoConst);

arqbin_auxLinha_I_espacoConst = fopen("
    FotonIngoing_rMaiorDoQue2_valor_auxLinha_I_espacoConstante.txt","r");
linha=fgets(arqbin_auxLinha_I_espacoConst);
auxLinha_I_espacoConstante=str2num(linha);
fclose(arqbin_auxLinha_I_espacoConst);

qtd_Colunas_I = [];
i = 1;
arqbin_qtd_Colunas_I = fopen("FotonIngoing_rMaiorDoQue2_vetor_qtd_Colunas_I.txt","r");
while(!feof(arqbin_qtd_Colunas_I))
  linha=fgets(arqbin_qtd_Colunas_I);
  qtd_Colunas_I(i,:)=str2num(linha);
  i = i + 1;
endwhile
fclose(arqbin_qtd_Colunas_I);

arqbin_auxLinha_II_espacoConst = fopen("
    FotonIngoing_rMaiorDoQue2_valor_auxLinha_II_espacoConstante.txt","r");
linha=fgets(arqbin_auxLinha_II_espacoConst);
auxLinha_II_espacoConstante=str2num(linha);
fclose(arqbin_auxLinha_II_espacoConst);

```

```
qtd_Colunas_II = [];
i = 1;
arqbin_qtd_Colunas_II = fopen("FotonIngoing_rMaiorDoQue2_vetor_qtd_Colunas_II.txt","r");
while(!feof(arqbin_qtd_Colunas_II))
  linha=fgets(arqbin_qtd_Colunas_II);
  qtd_Colunas_II(i,:)=str2num(linha);
  i = i + 1;
endwhile
fclose(arqbin_qtd_Colunas_II);


#constantes espaciais
#Limitando as curvas das contantes atraves do caminho futuro
coordenada_r_espacoConstante_coneFuturo = [];
coordenada_v_espacoConstante_coneFuturo = [];
for i = 1:auxLinha_I_espacoConstante
  for j = 1:(qtd_Colunas_I(i))
    r1 = r1_geodesicaConeFuturo_espacoConstante(i,j);
    r2 = r2_geodesicaConeFuturo_espacoConstante(i,j);
    coordenada_r_espacoConstante_coneFuturo(i,j) = CoordenadaEspaco(r2,r1,1,(1/pi)
    ,0.5,3);
    coordenada_v_espacoConstante_coneFuturo(i,j) = CoordenadaTempo(r2,r1,1,(1/pi),0.5,3)
    ;
  endfor
endfor
#Limitando as curvas das contantes atraves do caminho passado
coordenada_r_espacoConstante_conePassado = [];
coordenada_v_espacoConstante_conePassado = [];
for i = 1:auxLinha_II_espacoConstante
  for j = 1:(qtd_Colunas_II(i))
    r1 = r1_geodesicaConePassado_espacoConstante(i,j);
    r2 = r2_geodesicaConePassado_espacoConstante(i,j);
    coordenada_r_espacoConstante_conePassado(i,j) = CoordenadaEspaco(r1,r2,1,(1/pi)
    ,0.5,3);
    coordenada_v_espacoConstante_conePassado(i,j) = CoordenadaTempo(r1,r2,1,(1/pi)
    ,0.5,3);
  endfor
endfor


#Plotagem
figure(1)
for i = 1:auxLinha_I_espacoConstante
  j = 1;
  while(j < (qtd_Colunas_I(i)))
    plot(coordenada_r_espacoConstante_coneFuturo(i,j),
    coordenada_v_espacoConstante_coneFuturo(i,j),'k');
    hold on;
    j = j + 1;
  endwhile
endfor

for i = 1:auxLinha_II_espacoConstante
  j = 1;
  while (j < (qtd_Colunas_II(i)))
    plot(coordenada_r_espacoConstante_conePassado(i,j),
    coordenada_v_espacoConstante_conePassado(i,j),'k');
    hold on;
    j = j + 1;
  endwhile
endfor
axis([-0.2 1.0 -1.0 1.0])
```

```
#axis([0.8 1.6 -0.4 0.4])
#hold off;
#x = -5:0.4:5
#figure (1); for i = 1:length(x); plot(x,x+x(i),'k:'); hold on; endfor; axis([-0.2 1.0
    -1.0 1.0])
```

A.4.2. Região em que $r < 2M$

Código semelhante ao anterior, diferindo que cobre a região $r < 2M$, também em coordenadas avançadas. Ressalta-se que no caso em que estamos tratando de coordenadas retardadas, temos um cone de luz futuro, então as posições dos pontos $r1$ e $r2$ vão ser trocadas entre si nas funções.

```
#Importando os dados dos arquivos txt
r1_geodesicaConePassado_espacoConstante = [];
i = 1;
arqbin_r1_conePas_espacoConst = fopen("
    FotonIngoing_rMenorDoQue2_matriz_r1_geodesicaConePassado_espacoConstante.txt","r");
while(!feof(arqbin_r1_conePas_espacoConst))
  linha=fgets(arqbin_r1_conePas_espacoConst);
  r1_geodesicaConePassado_espacoConstante(i,:)=str2num(linha);
  i = i + 1;
endwhile
fclose(arqbin_r1_conePas_espacoConst);

r2_geodesicaConePassado_espacoConstante = [];
i = 1;
arqbin_r2_conePas_espacoConst = fopen("
    FotonIngoing_rMenorDoQue2_matriz_r2_geodesicaConePassado_espacoConstante.txt","r");
while(!feof(arqbin_r2_conePas_espacoConst))
  linha=fgets(arqbin_r2_conePas_espacoConst);
  r2_geodesicaConePassado_espacoConstante(i,:)=str2num(linha);
  i = i + 1;
endwhile
fclose(arqbin_r2_conePas_espacoConst);

arqbin_auxLinha_espacoConst = fopen("
    FotonIngoing_rMenorDoQue2_valor_auxLinha_espacoConstante.txt","r");
linha=fgets(arqbin_auxLinha_espacoConst);
auxLinha_espacoConstante=str2num(linha);
fclose(arqbin_auxLinha_espacoConst);

qtd_Colunas = [];
i = 1;
arqbin_qtd_Colunas = fopen("FotonIngoing_rMenorDoQue2_vetor_qtd_Colunas.txt","r");
while(!feof(arqbin_qtd_Colunas))
  linha=fgets(arqbin_qtd_Colunas);
  qtd_Colunas(i,:)=str2num(linha);
  i = i + 1;
endwhile
fclose(arqbin_qtd_Colunas);


#constantes espaciais
#Limitando as curvas das contantes atraves do caminho passado
coordenada_r_espacoConstante_conePassado = [];
coordenada_v_espacoConstante_conePassado = [];
for i = 1:auxLinha_espacoConstante
  for j = 1:(qtd_Colunas(i))
    r1 = r1_geodesicaConePassado_espacoConstante(i,j);
    r2 = r2_geodesicaConePassado_espacoConstante(i,j);
    coordenada_v_espacoConstante_conePassado(i,j) = CoordenadaEspaco(r1,r2,1,(1/pi)
    ,0.5,1);
```

```
    coordenada_r_espacoConstante_conePassado(i,j) = CoordenadaTempo(r1,r2,1,(1/pi)
    ,0.5,1);
  endfor
endfor

#Plotagem
figure(1)
for i = 1:auxLinha_espacoConstante
  j = 1;
  while(j < (qtd_Colunas(i)))
    plot(coordenada_v_espacoConstante_conePassado(i,j),
    coordenada_r_espacoConstante_conePassado(i,j),'r');
    hold on;
    j = j + 1;
  endwhile
endfor
axis([-0.2 1.0 -1.0 1.0])
#axis([0.8 1.6 -0.4 0.4])
#hold off;
```

A.5. Funções para limitar os intervalos das abscissa e ordenada

Através das funções "CoordenadaEspaco" e "CoordenadaTempo" encontramos as coordenadas correspondentes às do evento original em uma região compacta. Como já mencionado, aqui adotamos o arcotangente nessa transformação, de modo que a "CoordenadaEspaco" é

```
function x = CoordenadaEspaco(X1,X2,sinal,a,b,r)
  if(X1 < 2)
    x1 = sinal*((1/2) - (X1/4));
  else
    x1 = r_estrela(X1,sinal);
    x1 = a*atan(x1) + b;
  endif

  if(X2 < 2)
    x2 = sinal*((1/2) - (X2/4));
  else
    x2 = r_estrela(X2,sinal);
    x2 = a*atan(x2) + b;
  endif

  if(r < 2)
    x = (x2 - x1)/2;
  else
    x = (x2 + x1)/2;
  endif

endfunction
```

e a "CoordenadaTempo" é

```
function y = CoordenadaTempo(X1,X2,sinal,a,b,r)
  if(X1 < 2)
    x1 = sinal*((1/2) - (X1/4));
  else
    x1 = r_estrela(X1,sinal);
    x1 = a*atan(x1) + b;
  endif

  if(X2 < 2)
    x2 = sinal*((1/2) - (X2/4));
  else
    x2 = r_estrela(X2,sinal);
```

```
    x2 = a*atan(x2) + b;
  endif

  if(r < 2)
    y = (x1 + x2)/2;
  else
    y = (x2 - x1)/2;
  endif

endfunction
```