# 1. Gravity Tunnel in the Earth

## 1.1 Densities definition

### 1.1.1 Effective density function

First, we define the value of the constants that we are going to use: Acceleration of gravity, radius of the Earth, mean density of the Earth and gravity constant:

In[1]:=
```
g = 9.8156; (*m/s2*)
R = 6.371 × 10^6; (*m*)
ρ0 = 5513 ; (*kg/m3*)
G = 6.67 × 10^(-11) ;(* N m2/kg^2*)
```

We want to build a function of the following form

$$\rho 1\,(r;\ b,\ c,\ d) = \rho_0\, b \left(1 - c\,\frac{r}{R}\right)^d$$

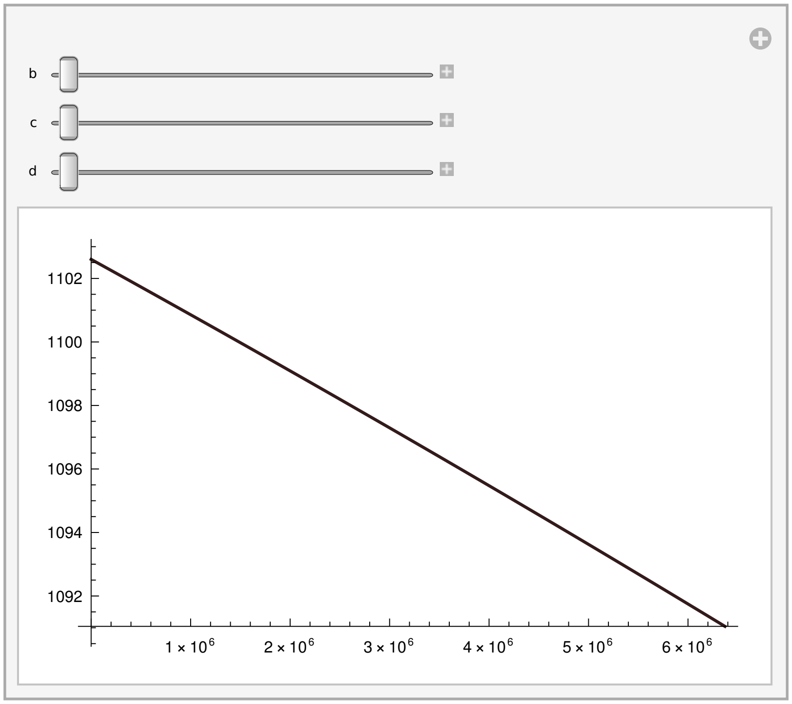
being b,c,d parameters to determine. It can be written in the code as the function

In[5]:=
```
ρ_1[r_, b_, c_, d_] := ρ0 b Power[1 - c r / R, d]
```

this density have the following forms

In[6]:=
```
Manipulate[Plot[ρ_1[r, b, c, d], {r, 0, R}, PlotStyle → RGBColor[b, c, d]],
 {b, 0.2, 4, 0.2}, {c, 0.1, 1, 0.1}, {d, 0.1, 1, 0.1}]
```

Out[6]=




#### 1.1.1.1 Effective density functions from PREM

Now, we have the next conditions to determine parameters b,c,d:

**1.** Density in the center of the Earth: $\rho(0) = \rho_0\, b = 13\,088.5$

then

In[ ]:=
```
b = 13 088.5 / ρ0 ;
Print["b = ", b]
```

```
b = 2.37325
```

**2.** Density at the surface must be that of the water $\left(1000 \text{ kg}/m^2\right)$:

$$\rho(r = R) = b\,\rho_0\,(1 - c)^d = \rho_R$$

then

$$(1 - c)^d = \frac{0.1813}{2.37325} = 0.076393$$

**3.** The third conditions is given from the total mass, we must have

$$M_T = 4\pi \int_0^R \rho(r)\, r^2\, dr = 4\pi\,\rho_0\, b \int_0^R \left(1 - c\,\frac{r}{R}\right)^d r^2\, dr$$

Doing the integral in Mathematica gives the analytical expression

```
Clear[b, R]
Integrate[b * x ^ 2 * (1 - c * (x / R)) ^ d, {x, 0, R}]
```

Out[ ]=

$$\frac{b\left(2+(1-c)^{d}(-1+c)(2+c(1+d)(2+c(2+d)))\right)R^{3}}{c^{3}(1+d)(2+d)(3+d)} \quad \text{if} \quad \mathrm{Re}[c] \le 1 \,||\, c \notin \mathbb{R}$$

so we can express the third condition as:

```
7.1197 * (2 - (1 - x) ^ (y + 1) (2 + 2 x (1 + y) + x ^ 2 (2 + 3 y + y ^ 2))) / (x ^ 3 * (6 + 11 * y + 6 * y ^ 2 + y ^ 3)) - 1
```

$$3\,b\left[2-\ (1-c)^{d+1}\left(2+2\,c\,(1+d)+c^{2}\left(2+3\,d\,+\,d^{2}\right)\right)\right]=\ c^{3}\left(6\,+\,11\,d\,+\,6\,d^{2}\,+\,d^{3}\right) \tag{1}$$

To solve the system of equations (1) and (3), this would be the code on Mathematica to solve the system, but it takes too long

In[ ]:=

```
Remove[x, y];
NSolve[{(1 - x) ^ y == 0.0763 , 7.1197 (2 - (1 - x) ^ (y + 1) * (2 + 2 * x * (1 + y) + x ^ 2 * (2 + 3 * y + x ^ 2))) /
    (x ^ 3 * (6 + 11 * y + 6 * y ^ 2 + y ^ 3)) == 1}, {x, y}]
```

On the other hand, the next code on Python finds the answer in a few seconds:

```
from scipy.optimize  import  root

def equations(p):
    x, y  = p
    eq1 = (1-x)**y  - 0.0763
    eq2 = 7.12*(2  - (1 - x)**(y  + 1)*(2  + 2*x*(1  + y)  + x**2*(2  + 3*y  + y**2)))  /
(x**3*(6  + 11*y  + 6*y**2  + y**3))  - 1
    return  ( eq1  , eq2  )

sol =  root(equations,  (0.1, 0.1), method='lm',  jac=None,  tol=None,
callback=None,  options={'col_deriv':  0, 'xtol':  1.49012e-08,  'ftol':  1.49012e-08,
'gtol':  0.0,  'maxiter':  0, 'eps': 0.0, 'factor':  100,  'diag':  None})


x = sol.x[0]
y = sol.x[1]

#print(equations((x,  y)))
print('x  = ', x)
print('y  = ', y)
```

```
x = 0.9875151202043234
y = 0.5867499795022912
```

For simplicity, the density function to work from here on is then

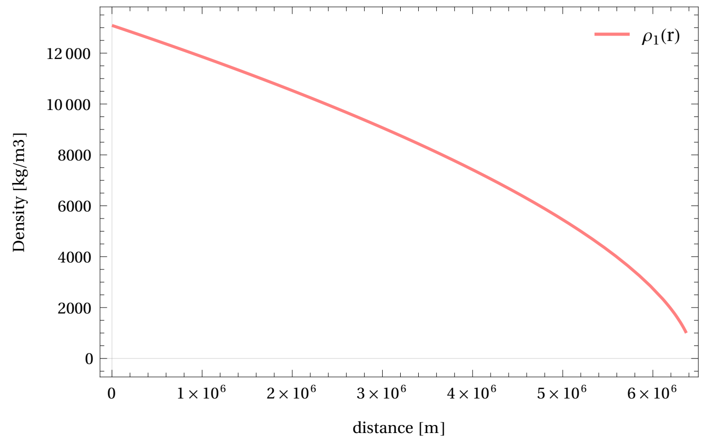
In[261]:=

```
b = 2.37325;
c = 0.9875151202043234 ;
d = 0.5867499795022912 ;
ρ[r_] := ρ_1[r, b, c, d]
```

Plotting with this result gives

```
In[ ]:= EffctiveFunction = Plot[{ρ[x]}, {x, 0, R},
PlotLegends → Placed[{"ρ₁(r)"}, {Right, Top}],
PlotStyle → {Thickness[0.005], Pink},
Frame → True,
FrameLabel → {{"Density[kg/m3]", None}, {"distance[m]", None}},
  PlotTheme → "Scientific"]
```

Out[ ]=

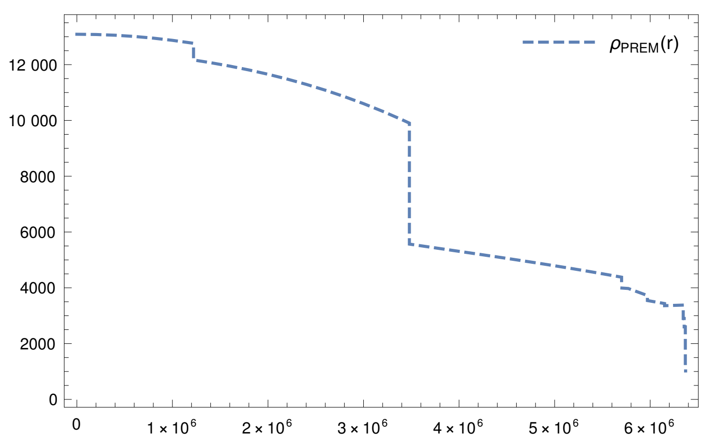



## 1.1.1.2 Numerical density from PREM

Now, let's import PREM data and plot to see how this distribution looks like

```
data = Import["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth
     Gravity Tunnel/Numerical Data/prem-density.csv", "Table"];
density_prem = ListLinePlot[data,
  PlotLegends → Placed[{"ρPREM(r)"}, {Right, Top}],
  PlotStyle → Dashed,
  Frame → True]
```

Out[ ]=

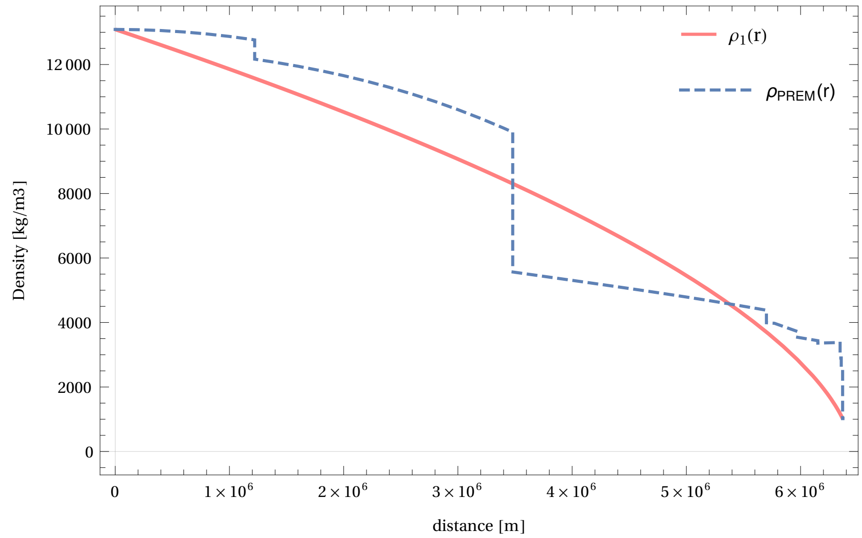


```
Export["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth
     Gravity Tunnel/Plots/density_prem.pdf", density_prem];
```

To compare, we can plot them together

In[ ]:=
```
densities = Show[EffctiveFunction, density_prem]
```

Out[ ]=




we see that actually the proposed functions fit between the data.

In[ ]:=
```
Export["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth
     Gravity Tunnel/Plots/1-densities.pdf", densities];
```

## 1.1.2 Numerical Checks for the Effective Density Function

### 1.1.2.1 Masses

Now, we want to determine the mass as a function of the radius. At r=R we would like to have M(r) = M ~ 6 x 10^15,

In[ ]:=
$$M[r\_] := 4\,\pi\ \text{NIntegrate}\left[\rho[x]\,x^2,\ \{x,\ 0,\ r\}\right]$$

the output at the radius is

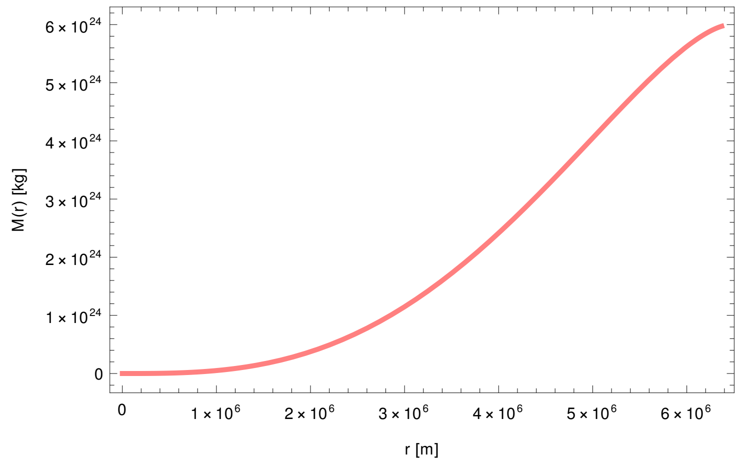
In[ ]:=
```
Print[Style["M_T = ", 20], Style[M[R], 20], Style[" kg", 20, Italic]]
```

$$M_T\ =\ 5.97237\times 10^{24}\ \text{kg}$$

the expected one. To see the behaviour at any other point let’s plot this functions

In[ • ]:=
```
Masses = Plot[{M[r]}, {r, 0, R},
  PlotStyle → {{Thickness[0.008], Pink}},
  Frame → True,
  FrameLabel → {{"M(r) [kg]", None}, {"r [m]", None}}]
```

Out[ • ]=




In[ • ]:=
```
Export["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth
    Gravity Tunnel/Plots/2-masses.pdf", Masses];
```

### 1.1.2 Gravity

**From density** functions, we can compute it as

$$a(r) = \frac{4\pi G}{r^2}\int_0^r \rho(r')\, r'^2\, dr'$$

As this is the same integral of the mass, we have already compute analytically with Mathematica, ‘til any point r, this gives

In[ • ]:=
```
Clear[b, c, d, R]
Integrate[x^2*(1 - c*(x/R))^d, {x, 0, r}]
```

Out[ • ]=
$$\frac{2R^3 + \left(1 - \frac{c\,r}{R}\right)^d (c\,r - R)\left(c^2 (1+d)(2+d) r^2 + 2c(1+d) r R + 2R^2\right)}{c^3 (1+d)(2+d)(3+d)} \quad \text{if} \quad \textit{condition}$$

Now, let’s define the function as

In[265]:=
```
a_analytic[r_] := 3 g b (2 R^2 - (1 - c r/R)^(d + 1) (c^2 (1 + d) (2 + d) r^2 + 2 c (1 + d) r R + 2 R^2)) /
    (c^3 r^2 (6 + 11 d + 6 d^2 + d^3))
```

At the radius the gravity is

In[ • ]:=
```
Print[Style["g_analytic = ", 20], Style[a_analytic[R], 20], Style[" m/s^2", 20, Italic]]
```

$$g_{analytic} = 9.81567\ \text{m/s}^2$$

As it should be. In terms of the binomial theorem, this can be made also analytically as

$$a(r) = \frac{4\pi G}{r^2}\rho_0\, b\int_0^r \left(1 - c\,\frac{r'}{R}\right)^d r'^2\, dr' = \frac{4\pi G}{r^2}\,\frac{3M}{4\pi R^3}\sum_{n=0}^{\infty}\binom{d}{n}\frac{(-c)^n}{R^n}\int_0^r r^{n+2}\, dr$$

$$= 3\, g\, b\sum_{n=0}^{\infty}\binom{d}{n}\frac{(-c)^n}{n+3}\left(\frac{r}{R}\right)^{n+1}$$

in the code

In[ ]:=
```
a[r_] := 3 b g Sum[QBinomial[d, n, 1] (-c)^n (r / R)^(n + 1) / (n + 3), {n, 0, Infinity}]
```

at the surface we have

```
Print[Style["g_Binom = ", 20], Style[a[R], 20], Style[" m/s^2", 20, Italic]]
```

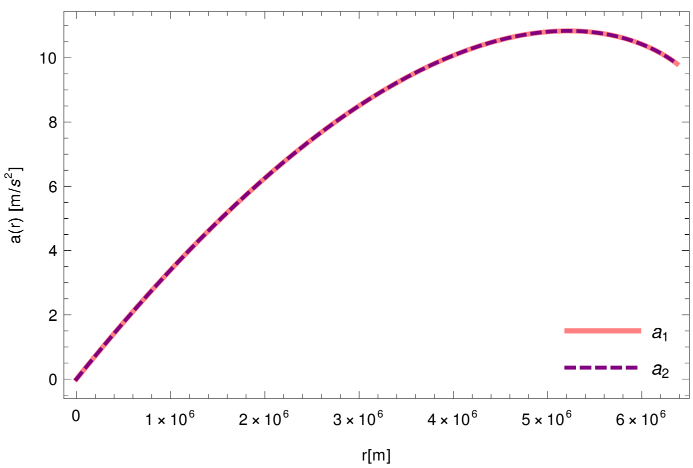
$$g = 9.81667 \ \ m/s^2$$

which is a little above the expected value.

We can both functions in the hole range by making plots

In[ ]:=
```
AccelerationFunctions = Plot[{a_analytic[r], a[r]}, {r, 0, R},
PlotLegends → Placed[{"a_1", "a_2"}, {Right, Bottom}],
PlotStyle → {{Thickness[0.008], Pink}, {Purple, Thickness[0.006], Dashed}},
Frame → True,
FrameLabel → {{"a(r) [m/s^2]", None}, {"r[m]", None}}]
```

Out[ ]=




We see now that they are indeed the same.

**From Prem**

Here we write the official data in the pertinent units and interpolate it to have a continuous function

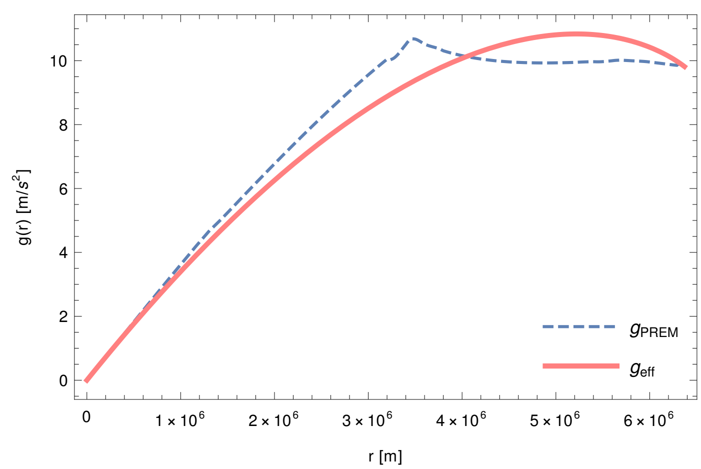
In[11]:=
```
gravity_prem =
  Import["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth Gravity
     Tunnel/Numerical Data/gravity_prem.csv", "Table"];
g_prem = Interpolation[gravity_prem, InterpolationOrder → 5]
gravity_prem_ad =
  Import["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth
     Gravity Tunnel/Numerical Data/prem-grav-ad.csv", "Table"];
g_prem_ad = Interpolation[gravity_prem_ad, InterpolationOrder → 5]
```

Out[12]=
```
InterpolatingFunction[ Domain: {{0., 6.37 × 10^6}}  Output: scalar ]
```

Out[14]=
```
InterpolatingFunction[ Domain: {{0., 1.}}  Output: scalar ]
```

Now, we can plot the three functions together

In[ ]:=
```
accelerations = Plot[{g_prem[r], a[r]}, {r, 0, R},
PlotLegends → Placed[{"g_PREM", "g_eff"}, {Right, Bottom}],
PlotStyle → {Dashed, {Thickness[0.008], Pink}},
Frame → True,
FrameLabel → {{"g(r) [m/s^2]", None}, {"r [m]", None}}]
```

Out[ ]=



In[ ]:=
```
Export["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth
    Gravity Tunnel/Plots/3-accelerations.pdf", accelerations];
```

# 1.2 Numerical Predictions

## 1.2.1 Velocity Profiles

Next, we are going to compute the velocities of the already found density profiles, in principle this should be made according to

$$v(r) = \sqrt{8\pi G \int_r^R \int_0^y \rho(x)\, x^2\, dx \frac{1}{y^2}\, dy}$$

or in terms of the integral of the acceleration. For the PREM profile, we have

In[40]:=
```
vprem[r_?NumberQ] := Sqrt[2 * NIntegrate[g_prem[x], {x, r, R}]]
vprem_ad[r_?NumberQ] := Sqrt[2/3 * NIntegrate[g_prem_ad[x], {x, r, 1}]]
```

For the case of the effective density function, this will be the direct integration of the analytical acceleration

In[17]:=
```
v_analytic[r_] :=
 Sqrt[8 Pi G ρ0 NIntegrate[b Power[1 - c x / R, d] x^2 / y^2, {y, r, R}, {x, 0, y}]]
```

while this is an still analytical expression using the binomial theorem expression:

In[51]:=
```
v_ad[r_] := Sqrt[2 b Sum[QBinomial[d, k, 1] * (-c)^k (1 - r^(k + 2)) / ((k + 2) (k + 3)), {k, 0, 20}]]
v[r_] := Sqrt[8 Pi G ρ0 R^2] ×
  Sqrt[b Sum[QBinomial[d, k, 1] * (-c)^k (1 - r^(k + 2)) / ((k + 2) (k + 3)), {k, 0, 10}]]
```

We can see that they exactly the same and very close to the value predicted by PREM distribution

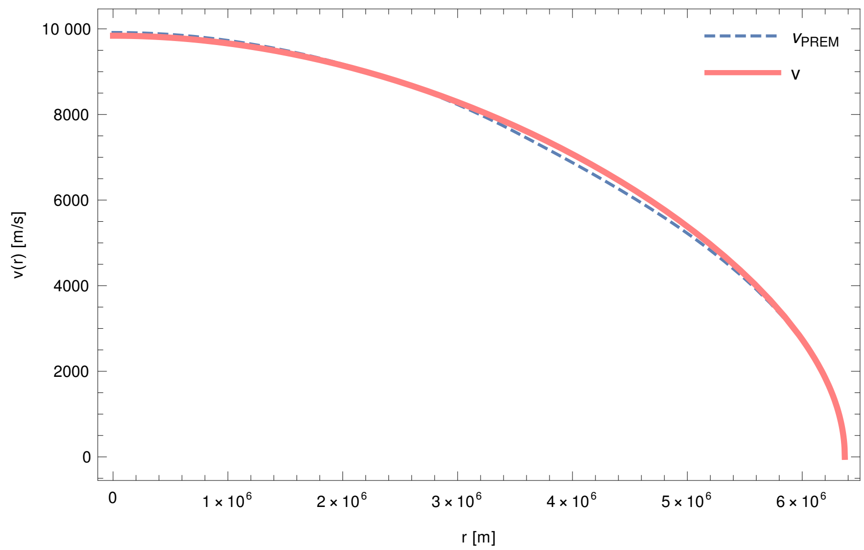
In[ ]:=
```
Print[Style["vPREM (0)= ", 20], Style[vprem[0], 20], Style[" m/s", 20, Italic], " , ",
 Style["vanalytic (0)= ", 20], Style[v_analytic[0], 20], Style[" m/s", 20, Italic], " , ",
 Style["vBinom (0)= ", 20], Style[v[0], 20], Style[" m/s", 20, Italic]]
```

$v_{PREM}$ (0)= 9914.72 m/s , $v_{analytic}$ (0)= 9840.17 m/s , $v_{Binom}$ (0)= 9840.17 m/s

In the hole range, the distribution of velocities looks as

```
In[ ]:= velocities = Plot[{vprem[x], v_analytic[x]}, {x, 0, R},
        PlotLegends → Placed[{"vPREM", "v"}, {Right, Top}],
        PlotStyle → {{ColorData[97][1], Dashed}, {Thickness[0.008], Pink}},
        Frame → True,
        FrameLabel → {{"v(r) [m/s]", None}, {"r [m]", None}}]
```

Out[ ]=




```
In[ ]:= Export["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth
          Gravity Tunnel/Plots/4-velocities.pdf", velocities];
```

## 1.2.2 Times chord path

We have arrived to the most important part of this work, the prediction of the traversal time of the train through the chord path. In the case between antipodes, the times are given by

$$T = 2\int_0^R \frac{1}{v(r)}\,dr.$$

in a more general chord path, characterized by a parameter d (distance from the polar axis), they are

$$T(d) = 2\int_d^R \frac{r}{v(r)}\,\frac{1}{\sqrt{r^2-d^2}}\,dr.$$

With the velocity profiles founded above, we can define three functions for computing times; one for PREM case and two for the effective density function. The reason for having taken all this time the two methods is that *analytic* functions were faster to plot above, but the integration of *binomial* case in this case results better, so we will define them but keep from here on, only the second case

In[88]:=
```
Tprem[d_] := NIntegrate[x / (vprem[x] * Sqrt[x ^ 2 - d ^ 2 ]), {x, d, R}]
Tprem_ad[d_] := NIntegrate[x / (vprem_ad[x] * Sqrt[x ^ 2 - d ^ 2 ]), {x, d, 1}]
T1[d_] := NIntegrate[x / (v_analytic[x] * Sqrt[x ^ 2 - d ^ 2 ]), {x, d, R}]
T_ad[d_] := NIntegrate[x / (v_ad[x] * Sqrt[x ^ 2 - d ^ 2 ]), {x, d, 1}]
T[d_] := R NIntegrate[x / (v[x] * Sqrt[x ^ 2 - d ^ 2 ]), {x, d, 1}]
```

Now, we can compute the output of this functions in minutes:

In[ ]:=
```
Print[Style["T_PREM = ", 20], Style[Tprem[0] / 30, 20], Style[" min", 20, Italic],
  "  ,  ", Style["T_Eff = ", 20], Style[T[0] / 30, 20], Style[" min", 20, Italic]]
```

$T_{PREM}$ = 38.1878 min , $T_{Eff}$ = 37.7624 min

this means an error of

In[ ]:=
```
Print[Style["Error = ", 20], Style[(Tprem[0] - T[0]) / Tprem[0] * 100, 20], Style[" %", 20]]
```

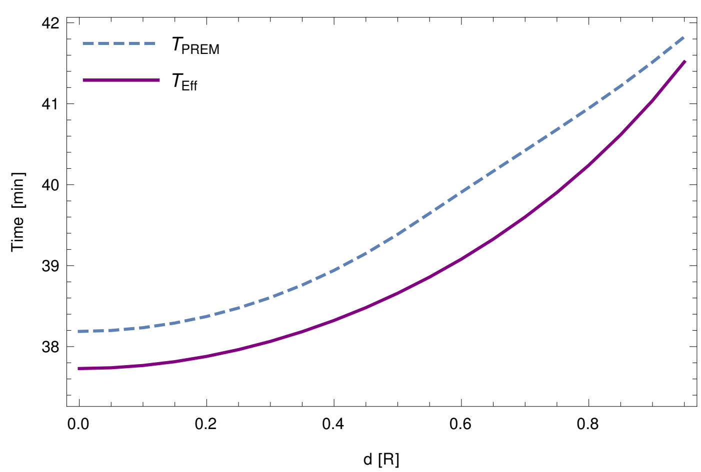
Error = 1.11406 %

But, again, the best way to compare the results is with a plot

In[114]:=
```
ListTimesPrem = Table[{d, Tprem_ad[d] * Sqrt[1 / 3 * R / g] / 30}, {d, 0, 0.95, 0.05}]
ListTimesEff = Table[{x, T_ad[x] * Sqrt[1 / 3 * R / g] / 30}, {x, 0., 0.95, 0.05}]
```

In[221]:=
```
chordtimes = ListLinePlot[{ListTimesPrem, ListTimesEff},
   PlotLegends → Placed[{"T_PREM", "T_Eff"}, {Left, Top}],
  PlotStyle → {{ColorData[97][1], Dashed}, {Purple, Thickness[0.005]}},
  Frame → True,
  FrameLabel → {{"Time [min]", None}, {"d [R]", None}}]
```

Out[221]=




In[222]:=
```
Export["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth
    Gravity Tunnel/Plots/5-chordtimes.pdf", chordtimes];
```

# 1.3 Further Analysis of the Gravity Tunnel

## 1.3.1 Shape of the brachistochrone curves

The other important part of this work is the time for the path of fastest descent and its shape. To find it we will need to perform the following integration of the trajectory:

$$\theta(r) = \int_d^r I(x, d)\, dx$$

with the integrand

$$I(r, d) = \left[\frac{r^4}{d^2}\left(\frac{v(d)}{v(r)}\right)^2 - r^2\right]^{-1/2}$$

The next is the code to define auxiliary functions for the integration,

In[66]:=
```
I1[x_, d_] := 1/(Sqrt[x^4/d^2 *(v_ad[d]/v_ad[x])^2 - x^2])
I_prem[x_, d_] := 1/(Sqrt[x^4/d^2 *(vprem_ad[d]/vprem_ad[x])^2 - x^2])
```

With them we can proceed to do the integration in a valid range

In[76]:=
```
θ[r_?NumericQ, d_?NumericQ] := If[r > d, NIntegrate[I1[x, d], {x, d, r}], 0.0]
θ_prem[r_?NumericQ, d_?NumericQ] :=
 Piecewise[{{0.0, r ≤ d}, {NIntegrate[I_prem[x, d], {x, d, r}], r > d}}]
(*If[r>d,NIntegrate[I_prem[x,d],{x,d,r}],0.0]*)
```

In order to plot the trajectories and visualise the motion in a better way, we should make a polar plot. As this is not so immediate, even in Mathematica, we have to make some lists with the data points, which is what the next cell does

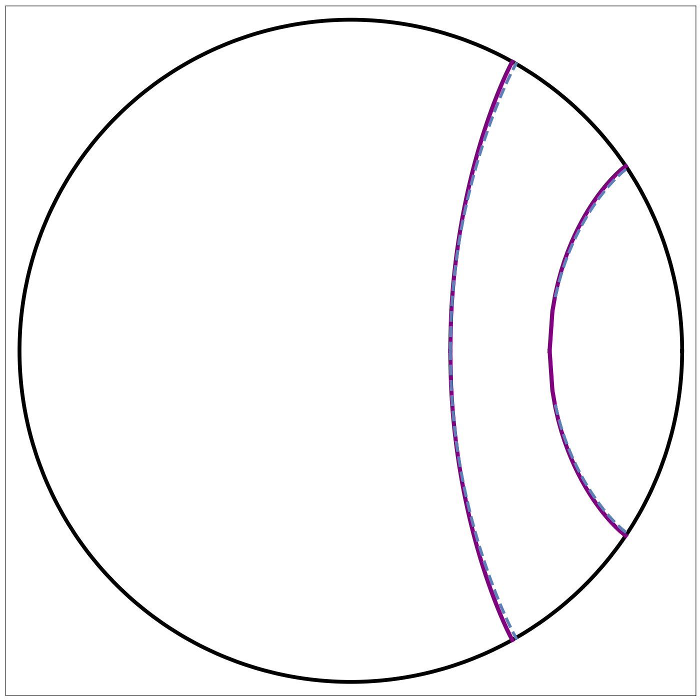
In[195]:=
```
Step = 0.02;
Theta3 = Table[{θ[i, 0.3], i}, {i, 0.3, 1, Step}];
Theta3_minus = Table[{- θ[i, 0.3], i}, {i, 0.3, 1, Step}];
Theta6 = Table[{θ[i, 0.6], i}, {i, 0.6, 1, Step}];
Theta6_minus = Table[{- θ[i, 0.6], i}, {i, 0.6, 1, Step}];
ThetaPrem3 = Table[{θ_prem[i, 0.3], i}, {i, 0.3, 1, Step}];
ThetaPrem3_minus = Table[{- θ_prem[i, 0.3], i}, {i, 0.3, 1, Step}];
ThetaPrem6 = Table[{θ_prem[i, 0.6], i}, {i, 0.6, 1, Step}];
ThetaPrem6_minus = Table[{- θ_prem[i, 0.6], i}, {i, 0.6, 1, Step}];
```

With that data, we plot the Earth as the Black circle, and the trajectories inside it.

In[137]:=
```
Circ = PolarPlot[1, {t, 0, 2 Pi},
   PlotStyle → {Thick, Black}, Frame → True, FrameTicks → None, Axes → None];
```

In[204]:=
```
Trajectories =
  ListPolarPlot[{Theta3, Theta3_minus, Theta6, Theta6_minus}, Joined → True,
   PlotStyle → {{Purple, Thickness[0.006]}}, Frame → True,
   FrameTicks → None, Axes → None];
TrajectoriesPrem = ListPolarPlot[{ThetaPrem3, ThetaPrem3_minus,
    ThetaPrem6, ThetaPrem6_minus}, Joined → True,
   PlotStyle → {{ColorData[97][1], Dashed}}, Frame → True,
   FrameTicks → None, Axes → None];
Brachistochornes = Show[Circ, Trajectories, TrajectoriesPrem]
```

Out[206]=



In[295]:=
```
Export["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth
    Gravity Tunnel/Plots/6-braqShapes.pdf", Brachistochornes];
```

### 1.3.2 Times for the brachistochrone curves

In this subsection we are going to find the times for the brachistochrone paths plotted above. As this are by definition the paths of fastest descent, the times should be the minimal times. The formula to compute this times is better given by

$$T_{\mathrm{BRAQ}} = \int_d^r \frac{\sqrt{1 + x^2\,\omega^2(x)}}{v(x)}\,dx$$

Where $\omega(r) = \partial_r\,\theta(r, d)$. As $\theta$ itself is an integral, by the fundamental theorem of calculus, $\omega$ = I(r,d):

In[208]:=
```
ω[r_, d_] := I1[r, d]
ω_prem[r_, d_] := I_prem[r, d]
```

and then the times for a given maximum approaching point *d*

In[210]:=
```
Tbraq[d_?NumericQ] := NIntegrate[Sqrt[1 + r^2 ω[r, d]^2] / v_ad[r], {r, d, 1}]
Tbraq_prem[d_?NumericQ] :=
 NIntegrate[Sqrt[1 + r^2 ω_prem[r, d]^2] / vprem_ad[r], {r, d, 1}]
```

let’s check that the times taken when d=0 are the same as for the chord path

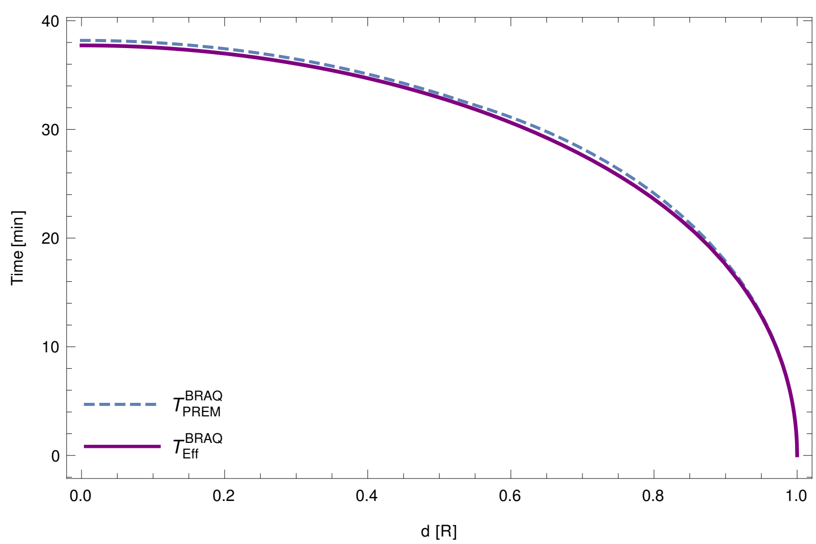
In[217]:=
```
Print[Style["T^BRAQ_PREM =", 20],
 Style[Tbraq_prem[0] × Sqrt[1/3 * R/g]/30, 20], Style[" min", 20, Italic],
 "  ,  ", Style["T^BRAQ_Eff =", 20], Style[Tbraq[0] × Sqrt[1/3 * R/g]/30, 20],
 Style[" min", 20, Italic]]
```

$T^{BRAQ}_{PREM}$ =38.1875 min , $T^{BRAQ}_{Eff}$ =37.7288 min

They are almost the same, maybe it’s a question of precision, but we can trust in this results. We can see the complete dependence of the times on the position of the tunnel in the following plot

In[228]:=
```
BraqTimes =
 Plot[{Tbraq_prem[x] × Sqrt[1/3 * R/g]/30, Tbraq[x] × Sqrt[1/3 * R/g]/30}, {x, 0, 1},
  PlotLegends → Placed[{"T^BRAQ_PREM", "T^BRAQ_Eff"}, {Left, Bottom}],
PlotStyle → {{ColorData[97][1], Dashed}, {Purple, Thickness[0.005]}},
Frame → True,
FrameLabel → {{"Time[min]", None}, {"d [R]", None}}]
```

Out[228]=




In[296]:=
```
Export["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth
   Gravity Tunnel/Plots/7-braqTimes.pdf", BraqTimes];
```

## 1.3.2 Accelerations

In this last section, we want give a dynamical reason for the difference between chord path times and brachistochrone path times. First, define some auxiliary functions

In[266]:=
```
dist[d_] := Sqrt[1 - (Sin[θ[1, d]])^2]
dist_prem[d_] := Sqrt[1 - (Sin[θ_prem[1, d]])^2]
```

with the radial accelerations and the these auxiliary functions we can define the acceleration in the direction of motion for the chord path

In[276]:=
```
a_chord[r_, d_] := Re[a_analytic[r R] / g × Sqrt[r^2 - (dist[d])^2] / r]
a_prem_chord[x_, d_] := Re[g_prem_ad[x] * Sqrt[x^2 - (dist_prem[d])^2] / x]
```

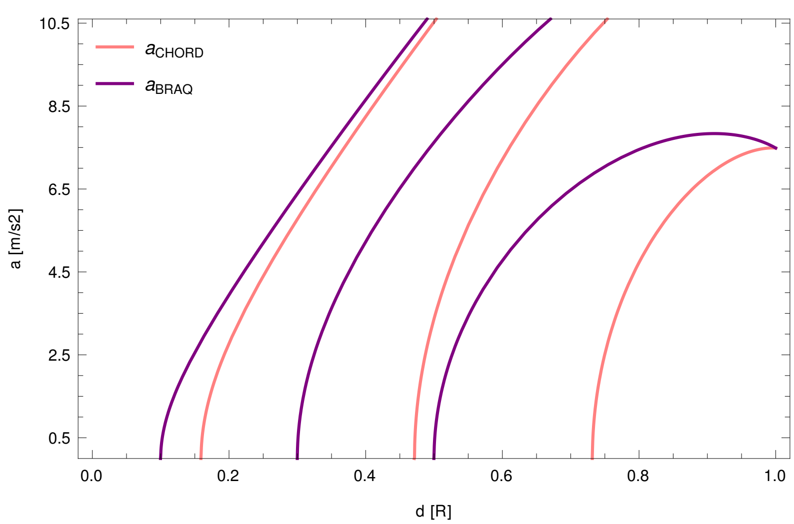
as well as the acceleration in the direction of motion for the brachistochrone paths

In[283]:=
```
a_braq[x_, d_] := Re[a_analytic[x R] / g * Sin[θ[x, d]]]
a_braq_prem[x_, d_] := Re[g_prem_ad[x] * Sin[θ_prem[x, d]]]
```

Now, let’s plot

In[292]:=
```
Accelerations_chord_braq = Plot[{Table[9.81 * a_chord[x, d], {d, 0.1, 0.5, 0.2}],
    Table[9.81 * a_braq[x, d], {d, 0.1, 0.5, 0.2}]}, {x, 0, 1},
   PlotRange → {0.0001, 10.6},
   PlotLegends → Placed[{"a_CHORD", "a_BRAQ"}, {Left, Top}],
   PlotStyle → {Pink, Purple},
  Frame → True,
  FrameLabel → {{"a [m/s2]", None}, {"d [R]", None}}]
```

Out[292]=




In[297]:=
```
Export["/home/nicolas/Documents/Physics/Bachelors-Dissertation/1-Earth Gravity
     Tunnel/Plots/8-Acceleration-chord-braq.pdf", Accelerations_chord_braq ];
```

Author: Nicolás Gómez Cruz