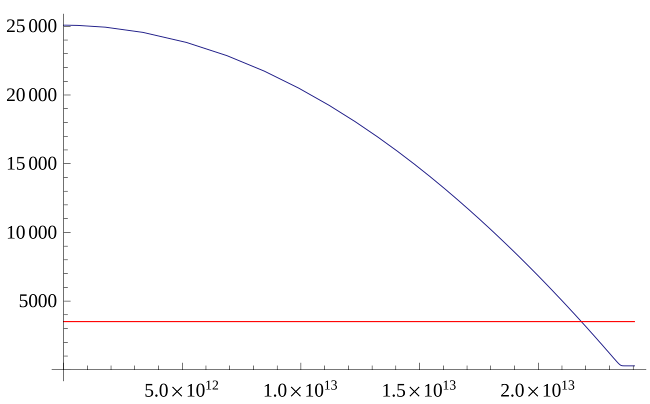
```
In[1]:= Clear["Global`*"]
kb = 1.38 × 10^-16;
mp = 1.67 × 10^-24;
μ = 0.615;
γ = 5 / 3;
G = 6.67 × 10^-8;
c = 3 × 10^10;
σ = 5.67 × 10^-5;
Msun = 2 × 10^33;

In[10]:=
M = 10^7 Msun;
κes = 0.4;
(*Calculating effective temperate based on the profile name*)
MyFile = "profile-35035-0.1-1000";
MyFileP = StringSplit[MyFile, "-"] // #[[2 ;;]] &;
MyFileP = ToExpression /@ MyFileP;
Σ = MyFileP[[1]];
Ṁ = MyFileP[[2]] 10 × 4 π G M/(c κes);
R = MyFileP[[3]] 2 G M/c^2;
ν = Ṁ/(3 π Σ);
Ω = Sqrt[G M/R^3];
Teff = ((9/8 ν Σ) Ω^2/σ)^0.25;

(*Import profile*)
profile = Import[NotebookDirectory[] <> MyFile, "Table"];
tprofile = profile[[All, {2, 4}]];
(*Maximum z for profile*)
zmax = profile[[-1, 2]];

tlow = Position[tprofile[[All, 2]], x_ /; x < Teff][[1]];
photopos = Extract[tprofile, tlow] // #[[1]] & // #/zmax &
(*Plot temperature and effective temperature for our profile*)
p1 = tprofile // ListLinePlot[#, PlotRange → All] &;
p2 = Plot[Teff, {z, 0, zmax}, PlotStyle → Directive[Red]];
Show[p1, p2]
```

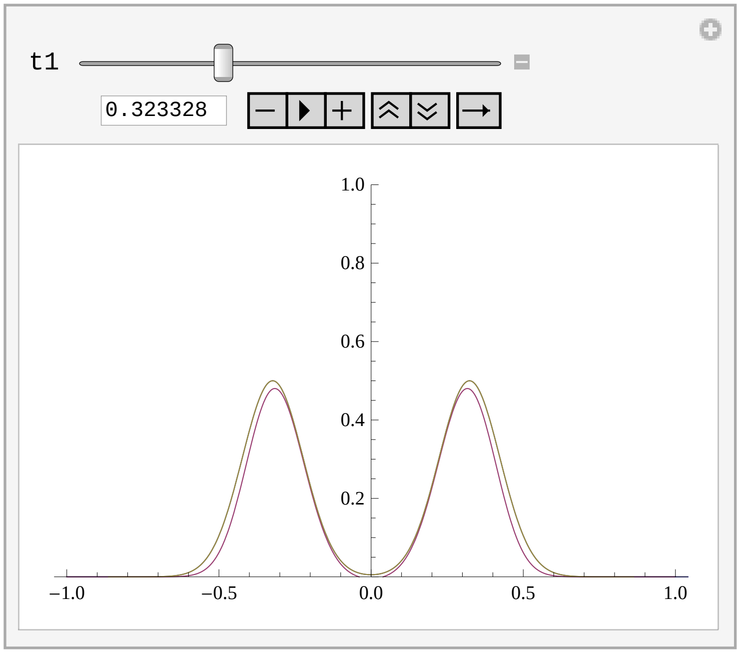
```
tprofile[[All, 1]] = tprofile[[All, 1]] / zmax;
tprofile = tprofile // Interpolation;
(*Sound  speed profile*)
cs[z_] := Sqrt[(γ kb tprofile[Abs[z]]) / (μ mp)]
(*cs[z_]:=1*)
cs2[z_] := cs[z] / cs[0]
(*Defining our perturbation*)
ρ1 = 1;
σ1 = 0.1;
(*Perturbation: contains all the info of the   analytic D'alembert solution*)
pert[z_, t_: 0] := 1/2 (ρ1 ⅇ^(-(z-t)^2/(2 σ1^2)) + ρ1 ⅇ^(-(z+t)^2/(2 σ1^2)));
(*Endpoint in time to which we would like to evolve the perturbation*)
tmax = 1;

(*Solution with non-constant sound speed, based on our temperature profile*)
pertsoln = NDSolve[{(cs2[z])^2 D[ρ[z, t], z, z] - D[ρ[z, t], t, t] == 0,
    ρ[1, t] == 0, ρ[-1, t] == 0, ρ[z, 0] == pert[z], Derivative[0, 1][ρ][z, 0] == 0},
   ρ, {z, -1, 1}, {t, 0, tmax}, Method → {"MethodOfLines",
     "SpatialDiscretization" → {"TensorProductGrid", "MaxStepSize" → 0.002}}];
(*Solution with constant sound speed*)
pertsoln2 = NDSolve[{D[ρ[z, t], z, z] - D[ρ[z, t], t, t] == 0,
    ρ[1, t] == 0, ρ[-1, t] == 0, ρ[z, 0] == pert[z], Derivative[0, 1][ρ][z, 0] == 0},
   ρ, {z, -1, 1}, {t, 0, tmax}, Method → {"MethodOfLines",
     "SpatialDiscretization" → {"TensorProductGrid", "MaxStepSize" → 0.002}}];
p1 = Table[{photopos, ρ1 * i}, {i, 0, 1}] // ListLinePlot;
p2 = Table[{-photopos, ρ1 * i}, {i, 0, 1}] // ListLinePlot;
Manipulate[
 Show[Plot[{pert[z, t1 ], ρ[z, t1] /. pertsoln, ρ[z, t1] /. pertsoln2},
   {z, -1, 1}, PlotRange → {Automatic, {0, 1}}, AxesOrigin → {0, 0}], p1, p2],
 {t1, 0 , tmax}, ContinuousAction → False, TrackedSymbols :→ {t1}
]
```

Out[25]= 0.908108

Out[28]=

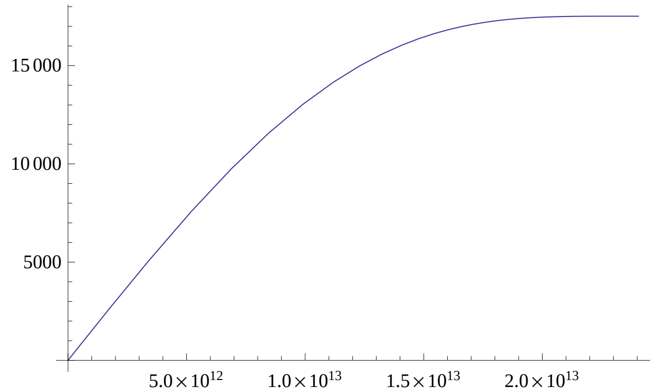


Out[41]=

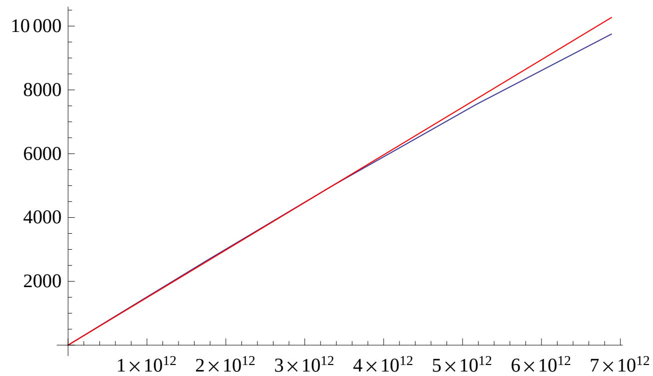


In[42]:=
```
(*Import profile*)
profile = Import[NotebookDirectory[] <> "profile-35035-0.1-1000", "Table"];
(*Plot u vs. z*)
profile[[All, {2, 1}]] // ListLinePlot[#, PlotRange → All] &
(*Maximum for which u looks linear with z just eyeballing the u vs. z graph*)
zmax = 8 × 10^12;
(*Select all the points in our profile
 that z less than the above arbitrarily chosen zmax*)
profile2 = Select[profile, #[[2]] < zmax &];
(*reset zmax to be the max z in profile2*)
zmax = profile2[[-1, 2]];
(*Find the z closest to zmax/2*)
midpoint = Nearest[profile2[[All, 2]], zmax/2] //
    Position[profile2[[All, 2]], #[[1]] ] & // #[[1, 1]] &;
(*Calculate the slope using this midpoint*)
m = profile2[[midpoint, 1]] / profile2[[midpoint, 2]];
(*m=profile[[1,3]]*)
linapprox[z_] := m z
p1 = profile2[[All, {2, 1}]] // ListLinePlot[#, PlotRange → All] &;
p2 = Plot[linapprox[z], {z, 0, zmax}, PlotStyle → Directive[Red]];
Show[p1, p2]
profile2[[All, {2, 4}]] // ListLinePlot[#, PlotRange → All] &
```

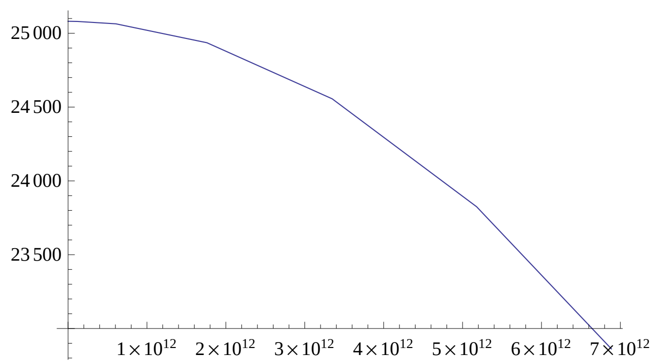
Out[43]=



Out[52]=



Out[53]=



```
(*Import profile*)
profile = Import[NotebookDirectory[] <> "profile-35035-0.1-1000", "Table"];
p1 = profile[[All, {1, 2}]] // ListLinePlot[#, PlotRange → All] &
ρc = profile[[1, 3]];
u0 = 2 profile[[-1, 1]];
```

$$z[u\_] := \frac{(u / \rho c)}{\left(1 - \left(\frac{2\,u}{u0}\right)\right)^{1/8}}$$

$$p2 = Plot\left[z[u],\ \left\{u,\ 0,\ \frac{u0}{2} - 1\right\},\ PlotStyle \rightarrow Directive[Red]\right]$$

```
Show[p1, p2]
```