UMEÅ UNIVERSITY
Department of Physics
Astrophysics

May 10, 2015

The Chandrasekhar Limit for White Dwarfs

Janis Erdmanis
akels14@gmail.com

Abstract

In this report white dwarf at hydrostatic equilibrium is considered numerically at it's mid stages. Governing equations are derived with assumption that white dwarf is cool enough so only degenerate electron gas gives significant pressure contribution. Density distribution dependance on radius as well as white dwarf mass dependance on radius is ploted.

Contents

1 Theory **2**

2 Calcuations **3**

3 Code **4**

1 Theory

At the hydrostatic equilibrium white dwarf satisfies a following equation system:

$$\frac{dP}{dr} = -\frac{Gm\rho}{r^2} \tag{1}$$

$$\frac{dm}{dr} = 4\pi r^2 \rho \tag{2}$$

At mid and late stages white dwarf internal pressure from degenerate electron gas is much larger than from radiation and ion presure so $P \approx P_{degenrate\ gas}$. According to the [Phillips] it is given as:

$$P = \frac{8\pi m_e^4 c^5}{3h^3} \int_0^{x_F} \frac{x^4}{(1+x^2)^{1/2}} dx \tag{3}$$

$$x_F = \frac{p_F}{m_e c} = \left(\frac{3n_e}{8\pi}\right)^{1/3} \frac{h}{m_e c} = \left(\frac{3Y_e \rho}{8\pi m_H}\right)^{1/3} \frac{h}{m_e c} \tag{4}$$

where I used $n_e = Y_e \rho / m_H$ with Y_e as average number of nucleon.

For the pressure derivative we can use chain rule:

$$\frac{dP}{dr} = \frac{dP}{d\rho}\frac{d\rho}{dr} \tag{5}$$

and so:

$$\frac{dP}{d\rho} = \frac{dP}{dx_F}\frac{dx_F}{d\rho} = \frac{8\pi m_e^4 c^5}{3h^3} \frac{x_F^4}{(1+x_F^2)^{1/2}} x_F \frac{d\log(x_F)}{d\rho} = \frac{8\pi m_e^4 c^5}{9h^3} \frac{x_F^5}{(1+x_F)^{1/2}} \frac{1}{\rho} \tag{6}$$

where one sees that if ρ is eliminated with x_F then pressure derivative as in the problem is obtained.

The system therefore becomes:

$$\frac{d\rho}{dr} = -\frac{Gm\rho}{r^2}\left(\frac{dP}{d\rho}\right)^{-1} \tag{7}$$

$$\frac{dm}{dr} = 4\pi r^2 \rho \tag{8}$$

$$x_F = \left(\frac{3Y_e\rho}{8\pi m_H}\right)^{1/3} \frac{h}{m_e c} \tag{9}$$

$$\frac{dP}{d\rho} = \frac{8\pi m_e^4 c^5}{9h^3} \frac{x_F^5}{(1+x_F)^{1/2}} \frac{1}{\rho} \tag{10}$$

$$\tag{11}$$

and is solved with initial conditions:

$$\rho(\varepsilon) = \rho_c \qquad\qquad m(\varepsilon) = \frac{4}{3}\pi\varepsilon^3\rho_c \tag{12}$$

where $\varepsilon \to 0$.

2 Calcuations

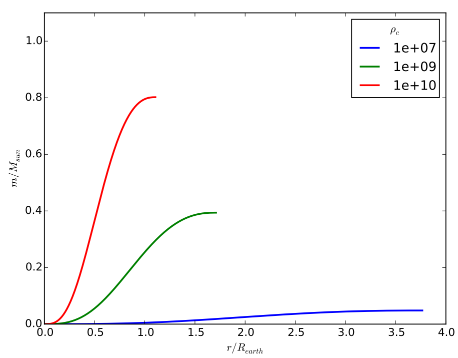

Figure 1: Shell mass dependance on radius for different dead star masses. For example red line corresponds to the mass $M = 0.8M_{sun}$ but its radius $R = 1.1R_{earth}$

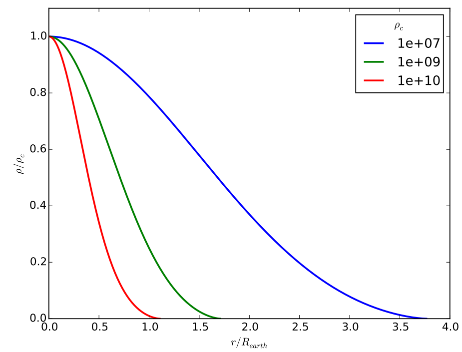

Figure 2: Density distribution for different White Dwarf masses

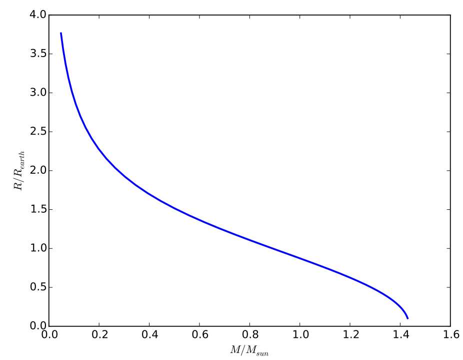

Figure 3: White Dwarf radius dependance on its mass. We see that radius of dead star goes to 0 as it's mass goes to $M_{CH} = 1.43 M_{sun}$ which is supported by [Philips] aprroximate value $1.4 M_{sun}$.

3 Code

Listing 1: Python code used for caclulations

```
from scipy.integrate import ode
import scipy.constants as C
from math import pi, sqrt

Y_e = 0.5
```

```
R_earth = 6371e3
M_sun = 1.989e30


def f(r, y):

    rho = y[0]
    m = y[1]

    x_F = (3*Y_e *rho/8/pi/C.m_p)**(1./3) * C.h/C.m_e/C.c
    DP = 8*pi*C.m_e**4 *C.c**5/9/C.h**3 * x_F**5/sqrt(1+ x_F**2) / rho

    Drho = - C.G*m*rho/r**2 / DP
    Dm = 4*pi*r**2*rho

    return [Drho,Dm]

import pylab as plt
fig1 = plt.figure(1)
fig3 = plt.figure(3)

for rho_c in [1e7,1e9,1e10]:
#    rho_c = 1e7
    eps = 0.01
    m_c = 4./3*C.pi * eps**3 * rho_c

    r = ode(f)
    r.set_initial_value([rho_c,m_c],eps)

    dt = R_earth/1e3

    x = []
    m = []
    rho = []
    while r.successful() and r.y[0] > 0:

        r.integrate(r.t + dt)

        x.append(r.t)
        rho.append(r.y[0])
        m.append(r.y[1])

    x = plt.array(x)
    rho = plt.array(rho)
    m = plt.array(m)
```

```
    plt.figure(1)
    plt.plot(x/R_earth,rho/rho_c,lw=2,label="{:.3g}".format(rho_c))

    plt.figure(3)
    plt.plot(x/R_earth,m/M_sun,lw=2,label="{:.3g}".format(rho_c))

plt.figure(1)
plt.xlabel(r"$r/R_{earth}$")
plt.ylabel(r"$\rho/\rho_c$")
plt.legend(loc=1,title=r"$\rho_c$")
plt.ylim(0,1.1)
plt.show()

plt.figure(3)
plt.xlabel(r"$r/R_{earth}$")
plt.ylabel(r"$m/M_{sun}$")
plt.legend(loc=1,title=r"$\rho_c$")
plt.ylim(0,1.1)
plt.show()



### Chandesekhar mass

def chande(rho_c):

    eps = 0.01
    m_c = 4./3*C.pi * eps**3 * rho_c

    r = ode(f)
    r.set_initial_value([rho_c,m_c],eps)

    dt = R_earth/1e3

    while r.successful() and r.y[0] > 0:

        r.integrate(r.t + dt)

    R = r.t
    M = r.y[1]
    return [R/R_earth,M/M_sun]

rho_0 = 1e7
rho_f = 1e14
```

```
xf = 100
a = plt.log(rho_f/rho_0)/xf
x = plt.linspace(0,xf)
rho = rho_0*plt.exp(a*x)

R = []
M = []
for rho_i in rho:
    Ri,Mi = chande(rho_i)
    R.append(Ri)
    M.append(Mi)

fig2 = plt.figure()
plt.plot(M,R,lw=2)
plt.xlabel(r"$M/M_{sun}$")
plt.ylabel(r"$R/R_{earth}$")
plt.show()


### saving to pdf

from matplotlib.backends.backend_pdf import PdfPages
with PdfPages("results.pdf") as pdf:
    pdf.savefig(fig1)
    pdf.savefig(fig2)
    pdf.savefig(fig3)
```