Indice

1 Introduzione

Studiamo la stabilità delle stelle di neutroni in regime relativistico considerando 3 possibili equazioni di stato per la materia. Una volta risolte le equazioni è possibile ottenere l'espressione del potenziale gravitazione della stella e calcolare l'effetto sulla radiazione emessa dalla stella.

Viene calcolata la radianza per ogni stella a 3 distanze diverse e la potenza totale di emissione in funzione della distanza dalla stella. Viene quindi calcolata la temperatura apparente delle 3 stelle più massive in funzione di quella effettiva e poi viene studiata la temperatura apparente in funzione della pressione centrale della stella.

2 Stabilità

Le equazioni che descrivono la stabilità di una stella in funzione della massa (m) e della pressione (P) sono quelle di Tolman-Oppenheimer-Volkoff

$$\begin{cases} \dfrac{\mathrm{d}P(r)}{\mathrm{d}r} = -G\dfrac{m(r)\epsilon(r)}{r^2c^2}\left(1+\dfrac{P(r)}{\epsilon(r)}\right)\left(+\dfrac{4\pi r^3P(r)}{m(r)c^2}\right)\left(1-\dfrac{2Gm(r)}{rc^2}\right)^{-1} & \text{(1a)} \\ \dfrac{\mathrm{d}m(r)}{\mathrm{d}r} = 4\pi r^2\dfrac{\epsilon(r)}{c^2} & \text{(1b)} \\ \dfrac{\mathrm{d}\Phi(r)}{\mathrm{d}r} = -\dfrac{1}{P(r)+\epsilon(r)}\dfrac{\mathrm{d}P(r)}{\mathrm{d}r} & \text{(1c)} \end{cases}$$

Dove la terza equazione è l'equazione disaccoppiata e descrive il potenziale gravitazionale della stella. Usiamo 3 diverse densità di energia per la materia della stella (eq. 2b viene presa con due coppie di valori diversi di Γ e K):

$$\epsilon_1(n) = a\left(\frac{n}{n_0}\right)^{\alpha} + b\left(\frac{n}{n_0}\right)^{\beta} \tag{2a}$$

$$\epsilon_{2/3}(n) = \mu c^2 n + Kc^2 n^{\Gamma} \tag{2b}$$

$$\text{con} \quad a = 13.4\text{MeV fm}^{-3}\,, \quad \alpha = 0.514, \quad b = 5.62\text{MeV fm}^{-3}\,, \quad \beta = 2.436\,, \quad n_0 = 0.16\text{fm}^{-3} \tag{3}$$

dove n è la densità numerica, μ la massa di una singola particella e quindi $\rho = \mu n$ è la densità di massa.

Visto che la densità di energia è in funzione di ρ e le incognite del sistema 1 sono P e m possiamo scrivere la densità di energia in funzione di P e m partendo dalla relazione termodinamica

$$P = -\tfrac{\mathrm{d}E}{\mathrm{d}V} \Rightarrow \begin{cases} P = (\alpha-1)a\left(\dfrac{n}{n_0}\right)^{\alpha} + (\beta-1)b\left(\dfrac{n}{n_0}\right)^{\beta} & \text{per } \epsilon_1 \quad \text{(4a)} \\ n = \left(\dfrac{P}{K(\Gamma-1)c^2}\right)^{1/\Gamma} & \text{per } \epsilon_{1/2} \quad \text{(4b)} \end{cases}$$

Nel primo caso (eq. 4a) non è stato possibile invertire l'equazione per trovare n in funzione di P e m quindi utilizzeremo un metodo numerico per trovare n di volta in volta.

Facciamo le seguenti sostituzioni per rendere le variabili adimensionali e con valori più vicini a 0.

$$m = M_0\hat{m}, \quad r = R_0\hat{r}, \quad P = P_0\hat{P}, \quad \rho = \rho_0\hat{\rho}, \quad K = \hat{K}\frac{\mu^{\Gamma}}{\rho_0^{\Gamma-1}},$$

$$\begin{cases} \dfrac{\mathrm{d}\hat{P}}{\mathrm{d}\hat{r}} = -\dfrac{(\hat{P}+\hat{\epsilon})(\hat{m}+\hat{r}^3\hat{P})}{\hat{r}^2-2\hat{m}\hat{r}} & \text{(5a)} \\ \dfrac{\mathrm{d}\hat{m}}{\mathrm{d}\hat{r}} = \hat{r}^2\hat{\epsilon} & \text{(5b)} \\ \dfrac{\mathrm{d}\Phi}{\mathrm{d}\hat{r}} = -\dfrac{1}{\hat{P}+\hat{\epsilon}}\dfrac{\mathrm{d}\hat{P}}{\mathrm{d}\hat{r}} & \text{(5c)} \end{cases}$$

Otteniamo il sistema 5 dove, grazie alle equazioni in 4, $\hat{m}$, $\hat{P}$ e $\hat{\epsilon}$ sono funzioni di $\hat{r}$. Come valori delle costanti sono stati usati

$$M_0 = 12.655756 M_\odot \quad R_0 = 20.06145\text{km} \quad \epsilon = P_0 = \rho_0 c^2 = 150.174 \frac{\text{MeV}}{c^2 \text{fm}^3} \tag{6}$$

Per il potenziale gravitazionale Φ si può inoltre trovare una soluzione analitica all'esterno della stella che possiamo mettere in forma adimensionale (eq. 7), dove $\hat{M}$ e $\hat{R}$ sono rispettivamente la massa totale e il raggio della stella in forma adimensionale.

$$\Phi_{\text{ext}}(r) = \frac{1}{2}\log\left(1 - \frac{2GM}{rc^2}\right) \implies \Phi_{\text{ext}}(\hat{r}) = \frac{1}{2}\log\left(1 - \frac{2\hat{M}}{\hat{r}}\right) \qquad \hat{r} \geq \hat{R} \tag{7}$$

3 Curva massa raggio

Cominciamo con il risolvere le prime due equazioni 5a e 5b del sistema adimensionale con il metodo `RK4` (appendice A). Per il caso con equzione di stato più complessa (eq. 2a) viene risolta numericamente l'equazione 4a per trovare ρ da P (Appendice B).

Segliamo massa iniziale 0 e pressioni iniziali differenti in modo da trovare soluzioni con R compreso tra i 3 e i 60 km. Il grafico massa raggio trovato viene riportato in figura 1.

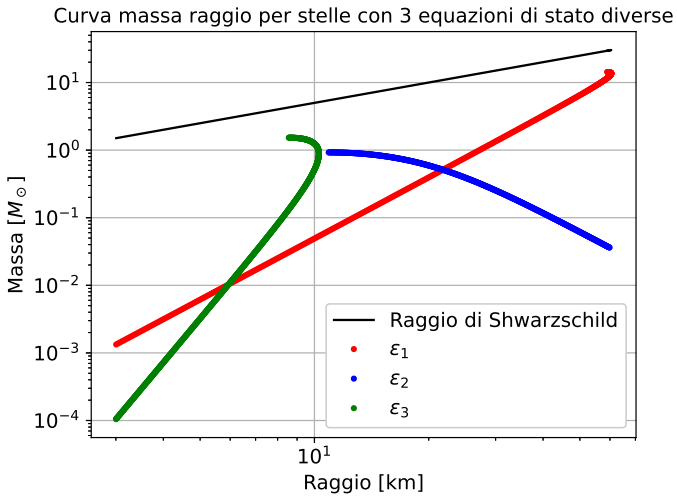

Figura 1: Curva massa raggio per stelle di equzioni di stato $\epsilon_{1/2/3}$. La prima equazione di stato, quella più realistica, prevede stelle di neutroni più massive.

Durante l'esecuzione del programma abbiamo inoltre smesso di incrementare la pressione centrale iniziale quando la condizione di stabilità $\frac{\mathrm{d}M}{\mathrm{d}r} > 0$ veniva meno.

Notiamo subito che con l'equazione di stato 2a il modello prevede stelle con massa e raggio maggiore rispetto al limite previsto dagli altri modelli.

I valori della stella più massiva per ogni equazione di stato diversa vengono riportati nella tabella 1.

	P_0 $[\frac{\text{MeV}}{c^2\text{fm}^3}]$	R [km]	M $[M_\odot]$
ϵ_1	43.31065	59.03824	14.29963
ϵ_2	217.0675	10.90280	0.9252994
ϵ_3	947.5339	8.559218	1.528782

Tabella 1: Valori della pressione iniziale P_0, massa totale M e raggio R della stella più massiva per ogni equazione di stato utilizzata.

4 Potenziale gravitazionale

Come mostrato nel sistema 1, l'equzione che determina il potenziale gravitazionale è disaccoppiata dalle altre due e, per $r \geq R$, può anche essere risolta analiticamente. Utilizzando le equazioni 5c e 7 possiamo trovare l'espressione generale per il potenziale all'interno della stella riportata in eq. 8.

$$\Phi_{\text{int}}(\hat{r}) = \Phi_{\text{ext}}(\hat{R}) - \int_{\hat{r}}^{\hat{R}} \frac{\mathrm{d}\Phi}{\mathrm{d}x}\,\mathrm{d}x = \frac{1}{2}\log\left(1 - \frac{2\hat{M}}{\hat{R}}\right) + \int_{\hat{r}}^{\hat{R}} \frac{1}{\hat{P}(x) + \hat{\epsilon}(x)} \frac{\mathrm{d}\hat{P}(x)}{\mathrm{d}x}\,\mathrm{d}x \tag{8}$$

dove siamo stati attenti a rendere $\Phi(r)$ continuo per ogni $r \geq 0$ usando il valore di Φ_{ext} in $\hat{R}$. Infine sostituendo alla derivata della pressione l'espressione in 5a otteniamo

$$\Phi_{\text{int}}(\hat{r}) = \frac{1}{2}\log\left(1 - \frac{2\hat{M}}{\hat{R}}\right) + \int_{\hat{r}}^{\hat{R}} \frac{\hat{m}(x) + x^3\hat{P}(x)}{2\hat{m}(x)x - x^2}\,\mathrm{d}x \tag{9}$$

Dati i valori di $\hat{P}(\hat{r})$ e $\hat{m}(\hat{r})$ che si ottengono risolvendo le equazioni di stabilità possiamo quindi risolvere l'integrale con il metodo dei trapezi per ottenere il valore di Phi a ogni r (Appendice C). Il grafico del potenziale gravitazionale ottenuto è mostrato in figura 2.

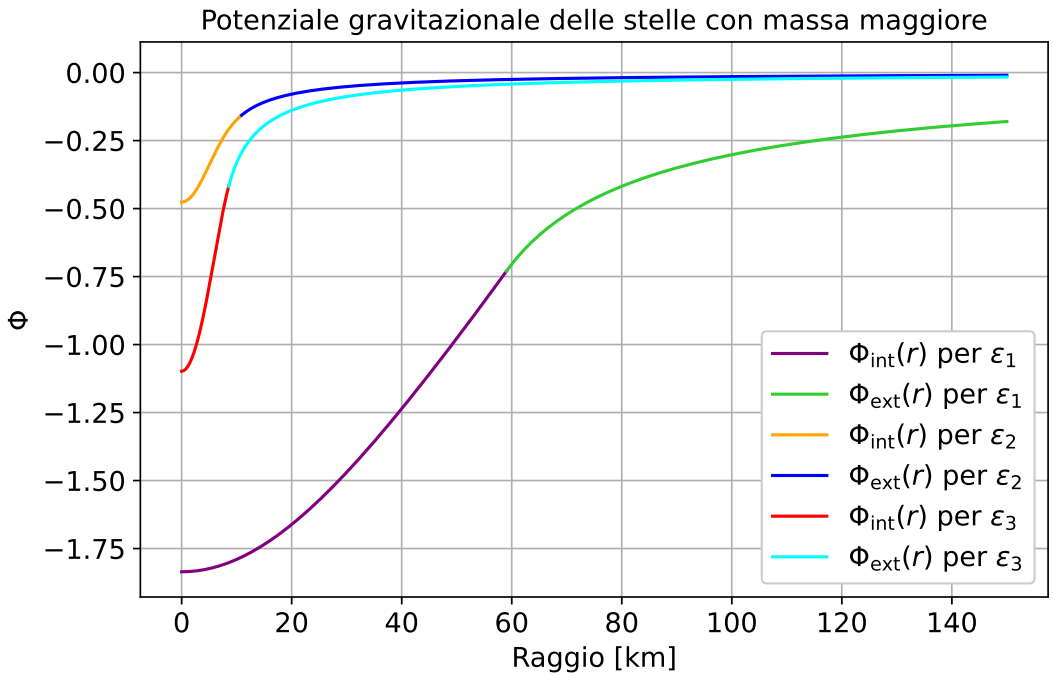

Figura 2: Grafico del potenziale gravitazionale all'interno e all'esterno della stella. Per $r < R$ è stato ottenuto integrando con il metodo dei trapezi l'eq. 9, per $r \geq R$ è stata plottata l'eq. 7

Vediamo dalla figura 2 che in tutti e 3 i casi $\Phi(r)$ è continuo, grazie alla condizione imposta.

Il potenziale Φ (che fa parte del termine $e^{\Phi(r)}c^2\mathrm{d}t^2$ della metrica $\mathrm{d}s^2$, che descrive la geometria dello spazio vicino alla stella) assume un andamento famigliare: raggiunge valori più bassi per le stelle più massive al diminuire di r e tende a 0 per r grandi.

5 Radianza

In metrica di Schwarzschild un fotone emesso a distanza r con frequenza ν_{em} viene ricevuto da un osservatore a distanza r' con una frequeza ν_{ric} data da

$$\frac{\nu_{\text{ric}}}{\nu_{\text{em}}} = e^{\Phi(r)-\Phi(r')}\,. \tag{10}$$

Dove $\Phi(r)$ è propio il potenziale gravitazionale mostrato in figura 2.
La radianza di una stella si può esprimere con l'equazione di Plank per il corpo nero

$$B(\nu, T) = \frac{2h\nu^3}{c^2}\frac{1}{e^{h\nu/(k_B T)} - 1} \tag{11}$$

dove T è la temperatura della stella e h la costante di Plank. Per fare i conti considerando solo il caso con $K_B T = 1\text{MeV}$ e mettiamo l'equzione 11 in funzione di variabili adimensionali. Otteniamo

$$\hat{B}(\hat{\nu}) = \frac{\hat{\nu}^3}{e^{\hat{\nu}} - 1}$$

dove abbiamo definito

$$\nu_0 = \frac{\nu}{\hat{\nu}} = \frac{h}{1\text{MeV}} \simeq 2.417\,989 \times 10^{20}\text{Hz}$$
$$B_0 = \frac{B}{\hat{B}} = \frac{2h\nu_0^3}{c^2} = \frac{2\text{MeV}}{(hc)^2} \simeq 1.301\,059 \times 10^{-6}\frac{\text{MeV}}{\text{fm}^2}$$

Infine, se consideriamo l'effetto doppler dovuto al potenziale gravitazionale descritto in eq. 10 e utilizziamo la formula analitica per il potenziale gravitazionale all'esterno della stella (eq. 7), otteniamo

$$\hat{\nu}_{\text{em}} = \hat{\nu}_{\text{ric}}\left(\frac{1-\frac{2\hat{M}}{r}}{1-\frac{2\hat{M}}{\hat{R}}}\right)^{1/2} \quad\Longrightarrow\quad \hat{B}(\hat{\nu}_{\text{ric}}, \hat{r}) = \frac{\hat{\nu}_{\text{ric}}^3\left(1-\frac{2\hat{M}}{\hat{r}}\right)^{3/2}\left(1-\frac{2\hat{M}}{\hat{R}}\right)^{-3/2}}{\exp\left(\hat{\nu}_{\text{ric}}\left(1-\frac{2\hat{M}}{\hat{r}}\right)^{1/2}\left(1-\frac{2\hat{M}}{\hat{R}}\right)^{-1/2}\right)-1} \tag{12}$$

Nel figure 3, 4 e 5 viene studiato lo spettro della radiazione emessa per le 3 stelle massive di cui abbiamo studiato il potenziale in 2 a distanze diverse dalla stella. In blu è plottata l'eq. 11, ovvero la radianza senza correzioni relativistiche.

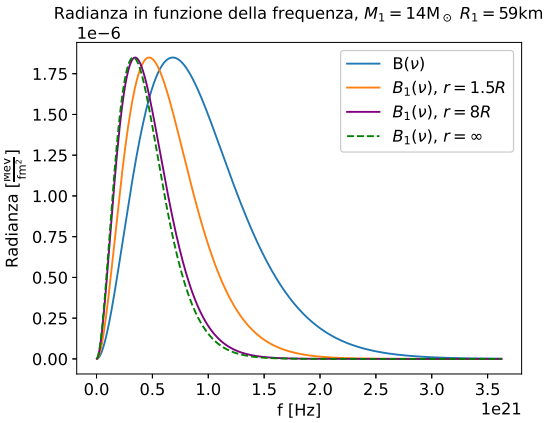

Figura 3: Radianza di una stella con $M = 14M_\odot$ e $R = 59$km, percepita a 88.6km, 472km e a distanza infinita. Essendo la stella con massa maggiore è anche quella in cui il redshift è più visibile.

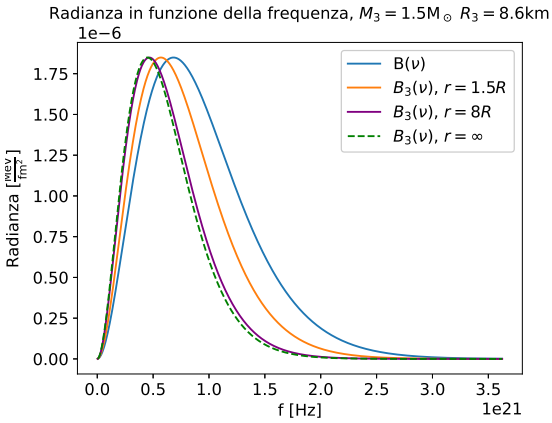

Figura 4: Radianza da una stella con $M = 1.5M_\odot$ e $R = 8.6$km, percepita a 12.8km, 68.5km e a distanza infinita. In blu la radianza senza correzioni relativistiche.

La curva in tutti e 3 i casi subisce uno spostamento verso le frequenze più basse, *redshift* per l'appunto, e l'effetto è tanto maggiore quanto più la stella è massiva e l'osservatore distante. Già a una distanza di 8 volte il raggio della stella la curva della radianza è molto simile a quella all'inifinito, in cui il termine $\Phi(r')$ in eq. 10 è trascurabile.

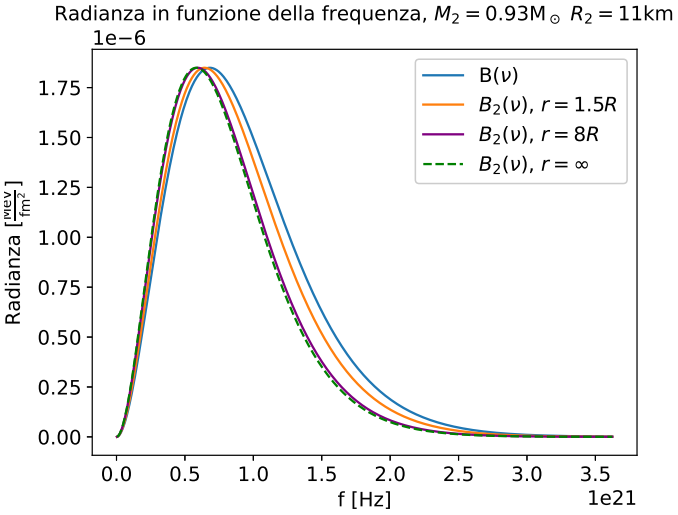

Figura 5: Radianza da una stella con $M = 0.93M_\odot$ e $R = 11$km, percepita a 16.4, 87.2 km e a distanza infinita. In blu la radianza senza correzioni relativistiche.

6 Potenza emessa

La potenza totale emessa dalla stella per un elemento della sua superficie, si può ottenere integrando la radianza $B(\nu, T)$ sulle frequenze e sull'angolo solido, con l'accortezza di moltiplicare per un fattore $\cos(\theta)$ (Legge di Lambert) per proiettare la radiazione lungo la normale alla superficie sottesa da $\mathrm{d}\Omega$.

$$\mathcal{P} = \int_0^\infty \mathrm{d}\nu \int B(\nu, T)\cos(\theta)\ \mathrm{d}\Omega = \int_0^{Ak_BT} B(\nu, T)\ \mathrm{d}\nu \int_0^{2\pi} \mathrm{d}\phi \int_0^1 \mathrm{d}(\cos(\theta))\ \cos(\theta) \tag{13}$$

$$= \pi \int_0^{Ak_BT} B(\nu, T)\ \mathrm{d}\nu \tag{14}$$

Utilizzando la stessa temperatura della sezione 5, $k_BT = 1\mathrm{MeV}$, e le unità adimensionali, l'integrale in 14 può essere riscritto come

$$\hat{\mathcal{P}}(\hat{r}) = \int_o^{\frac{A\,\mathrm{MeV}}{\nu_0}} \hat{B}(\hat{\nu}_{\mathrm{ric}}, \hat{r})\ \mathrm{d}\hat{\nu} \tag{15}$$

dove $\hat{B}(\hat{\nu}_{\mathrm{ric}}, \hat{r})$ è definita in 12 e

$$\mathcal{P}_0 = \frac{\mathcal{P}}{\hat{\mathcal{P}}} = \pi B_0 \nu_0 \simeq 9.883\,290 \times 10^{14}\ \frac{\mathrm{MeV}}{\mathrm{s\,fm}^2} \tag{16}$$

6.1 Convergenza dell'integrale

Dal momento che numericamente non è possibile integrare sulle frequenze fino a $\nu = +\infty$ si sceglie A sufficientemente grande da poter approssimare in modo ragionevole l'integrale. Facendo riferimento alla figura 4, possiamo vedere che per $\nu > 3.5 \times 10^{21}$Hz la radianza è quasi nulla. In quell'occasione era stata calcolata $\hat{B}(\hat{\nu})$ tra 0 e 15. Come stima iniziale possiamo quindi integrare fino a $\hat{\nu} = 20$ che corrisponde a

$$\frac{A\,\mathrm{MeV}}{\nu_0} = 20 \quad \Longleftrightarrow \quad A = 20\,\nu_0\,\mathrm{MeV}^{-1} \simeq 4.835\,978 \times 10^{21}\,\mathrm{MeV}^{-1}\,\mathrm{s}^{-1} \tag{17}$$

Possiamo ottenere un valore migliore valutando la convergenza dell'integrale. Per prima cosa verifichiamo per quale `N` (numero di step nell'integrazione) il valore di $\mathcal{P}$ converge (il codice si trova in Appendice D).

Studiamo il caso in cui è necessario un `N` maggiore, ovvero quello in cui la curva è più ripida: $r = \infty$, $M_1 = 14.3M_\odot$ e con $\hat{\nu}_{\max} = 20$. Otteniamo il grafico in figura 6. I valori sono degli scarti rispetto a

$\mathcal{P}_{N=1e7}$ (la potenza calcolata con $N = 1 \times 10^7$) e sono normalizzati rispetto alla stessa. Al contriario di quello che succede di solito, il metodo Simpson (che ha un costo computazionale maggiore), non presenta vantaggi rispetto a quello dei trapezi. Dal codice (Appendice D) che genera i dati mostrati in figura 6 otteniamo un errore relativo minore di 1×10^{-5} per `N_trap = 62` e `N_simp = 82`.

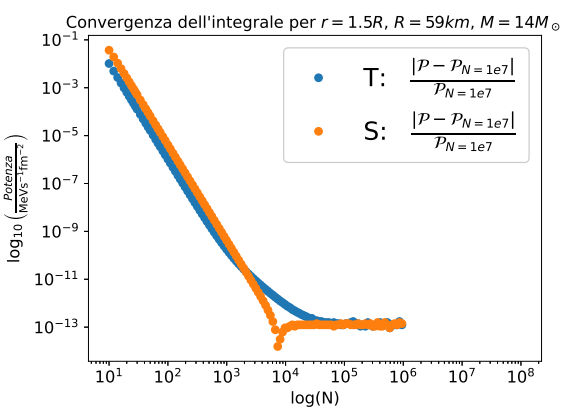

Figura 6: Errore relativo sul calcolo della potenza per diversi `N` (numero di step nell'integrazione con i trapezi **T** e con simpson **S**).

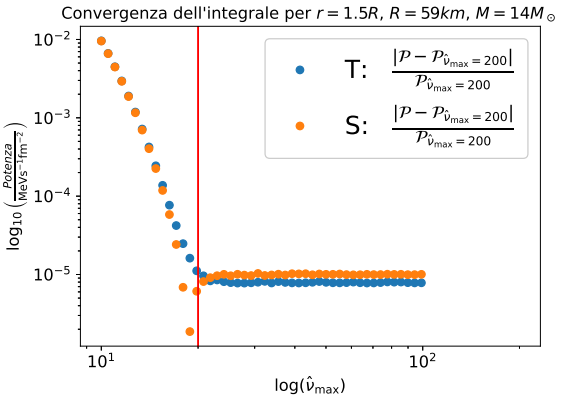

Figura 7: Errore relativo sul calcolo della potenza per diversi valori di $\hat{\nu}_{\max}$. Per $\hat{\nu}_{\max} > 20$ (line verticale rossa) otteniamo una precisione maggiore di 1×10^{-8}, il valore di A trovato in 17 è quindi una buon upperboud.

Fissati `N_trap` e `N_simp` per un intervallo $\Delta\hat{\nu} = 20$ proponiamo quindi in figura 7 lo stesso tipo di grafico degli scarti fatto però in funzione della scelta di $\nu_{\max}$, dove abbiamo tenuto costante il valore `N / nu_max` apena ottenuto. Questa volta si studia il caso in cui $r = R_1$, ovvero quello in cui la curva è più spostata verso destra e serve un upper bound maggiore.

Dallo script per generare i dati mostrati in figura 7 otteniamo un errore di 1×10^{-5} per

	N	$\hat{\nu}_{\max}$
Trapezi	64	20.7892818
Simpson	78	17.9585633

Per comodità decidiamo di utilizzare lo stesso valore di $\hat{\nu}_{\max} = 21$ per entrambi i metodi [1] e quindi, per eccesso,

$$\texttt{N\_trap} = \frac{64}{20.7892818} \times 21 = 67 \quad \text{e} \quad \texttt{N\_simp} = \frac{78}{17.9585633} \times 21 = 92 \tag{18}$$

Di conseguenza possiamo valutare

$$A = 21\,\nu_0\,\mathrm{MeV}^{-1} \simeq 5.077\,769 \times 10^{21}\,\mathrm{MeV}^{-1}\,\mathrm{s}^{-1} \tag{19}$$

6.2 Potenza totale in funzione della distanza

Ottenuti i parametri per integrare numericamente l'equazione 15 possiamo usarli per generare i grafico di $\mathcal{P}$ in funzione di r (distanza dal centro della stella) per le stelle più massive descritte in 1.

Figures/Pot.eps

Figura 8: FINALMENTE

[1] Si noti che Simpson tende a convergere prima al variare di `nu_max` perché, in questo caso, approssima la curva per eccesso e quindi compensa il tratto di funzione che viene escluso dall'integrazione. La scelta di aumentare `nu_max` anche per questo metodo mantiene comunque l'errore minore di 1×10^{-5} vista figura 6.

Stella	r[km]	$\mathcal{P}(r)$[MeV s^{-1} fm^{-2}]	Metodo
$M_1 = 14.3 M_\odot$ $R_1 = 59.0$km	88.6	$4.426\,067\,202 \times 10^{15}$	Trapezi
		$4.426\,067\,212 \times 10^{15}$	Simpson
	472.3	$3.252\,180\,370 \times 10^{15}$	Trapezi
		$3.252\,180\,396 \times 10^{15}$	Simpson
	∞	$3.092\,162\,593 \times 10^{15}$	Trapezi
		$3.092\,162\,623 \times 10^{15}$	Simpson
$M_2 = 0.925 M_\odot$ $R_2 = 10.9$km	16.4	$6.057\,276\,926 \times 10^{15}$	Trapezi
		$6.057\,276\,930 \times 10^{15}$	Simpson
	87.2	$5.581\,867\,348 \times 10^{15}$	Trapezi
		$5.581\,867\,353 \times 10^{15}$	Simpson
	∞	$5.487\,198\,980 \times 10^{15}$	Trapezi
		$5.487\,198\,986 \times 10^{15}$	Simpson
$M_3 = 1.53 M_\odot$ $R_3 = 8.56$km	12.8	$5.357\,438\,539 \times 10^{15}$	Trapezi
		$5.357\,438\,544 \times 10^{15}$	Simpson
	68.5	$4.384\,963\,175 \times 10^{15}$	Trapezi
		$4.384\,963\,185 \times 10^{15}$	Simpson
	∞	$4.226\,926\,032 \times 10^{15}$	Trapezi
		$4.226\,926\,043 \times 10^{15}$	Simpson

Tabella 2: Risultati integrazione numerica

A RK4

Codice con cui è stato implementato il metodo `RK4`. `fun_P()` e `fun_m()` sono le funzioni presenti a destra dell'uguale nella prima e nella seconda riga del sistema 5.

```
void rungeKutta4(double h, double r, double *P, double *m, int tipo_politropica){

    double k1, k2, k3, k4, l1, l2, l3, l4;

    k1 = h * fun_m(r, *P, tipo_politropica);
    l1 = h * fun_P(r, *P, *m, tipo_politropica);

    k2 = h * fun_m(r + h / 2, *P + l1 / 2, tipo_politropica);
    l2 = h * fun_P(r + h / 2, *P + l1 / 2, *m + k1 / 2, tipo_politropica);

    k3 = h * fun_m(r + h / 2, *P + l2 / 2, tipo_politropica);
    l3 = h * fun_P(r + h / 2, *P + l2 / 2, *m + k2 / 2, tipo_politropica);

    k4 = h * fun_m(r + h, *P + l3, tipo_politropica);
    l4 = h * fun_P(r + h, *P + l3, *m + k3, tipo_politropica);

    *m += (k1 + 2 * k2 + 2 * k3 + k4) / 6;
    *P += (l1 + 2 * l2 + 2 * l3 + l4) / 6;
}
```

B Risoluzione numerica di $\hat{P}(\rho)$

Quando si calcola il valore della derivata di P o m ad un dato r serve anche il valore dell'energia interna, infatti

```
// fun_m = r^3 E
double fun_m(double r, double P, int tipo_politropica){
    return r * r * fun_E(P, tipo_politropica);
}


```

```
// fun_P = - (P + E)(m + r^3 P)/(r^2 - 2mr)
double fun_P(double r, double P, double m, int tipo_politropica){
    if (m == 0)
        return 0;
    return (P + fun_E(P, tipo_politropica)) * (m + pow(r, 3) * P) / ((2 * m - r) * r);
}
```

Dove la funzione `fun_E()` è definita come

```
double fun_E(double P, int tipo_politropica){
    // Politropica quasi realistica (a*rho^alpha + b*rho^beta)
    if (tipo_politropica == 0){
        double rho = findRho(P);
        return A * pow(rho, ALPHA) + B * pow(rho, BETA);
    }

    double lambda, K;

    // Politropiche semplici
    if (tipo_politropica == 1){
        lambda = 5. / 3.;
        K = 0.05;
    } else if (tipo_politropica == 2){
        lambda = 2.54;
        K = 0.01;
    } else {
        printf("Tipo politropica non riconosciuto\n");
        return 0;
    }

    double a1 = P / (lambda - 1);
    return a1 + pow(a1 / K, 1. / lambda);
}
```

Per le politropiche semplici (eq. 2b) abbiamo potuto trovare un'espressione analitica per $\rho(P)$ (eq. 4b). L'energia può quindi essere calcolata in modo diretto come viene fatto nelle righe 22-23 del codice sopra riportato.

Per la politropica 2a, la relazione tra ρ e P che si trova è data in 4a, che riportiamo

$$P = (\alpha - 1)a \left(\frac{n}{n_0}\right)^{\alpha} + (\beta - 1)b \left(\frac{n}{n_0}\right)^{\beta} . \tag{20}$$

Data una certa pressione P bisopgna quindi risolvere numericamente l'equazione per trovare il valore n che la soddisfa. Per fare ciò utiliziamo la funzione `findRho()` così definita

```
double findRho(double P){

    // Cominciamo prima con il metodo di Newton-Raphson
    if (P > 0.01){

        double rho = pow(P / (BETA1 * B), 1 / BETA);     // buona approx. iniziale
        while (fabs(P - P_of_rho(rho)) > 1e-6){
            rho -= (P_of_rho(rho) - P) / DP_of_rho(rho);
        }
        return rho;
    }

    // Per P piccoli meglio usare bisezione
    double rho_sx = 0.7, rho_dx = 1.;
    double rho = (rho_sx + rho_dx) / 2;
    while (fabs(P - P_of_rho(rho)) > 1e-6){
        if (P_of_rho(rho) > P)
            rho_dx = rho;
        else
            rho_sx = rho;
```

```
21          rho = (rho_sx + rho_dx) / 2;
22      }
23      return rho;
24  }
```

dove `P_odf_rho()` e `DP_of_rho()` sono rispettivamente la funzione 20 e la sua derivata. In questo modo possiamo sempre trasformare $\epsilon(\rho)$ in $\epsilon(P)$.

C Risoluzione dell'integrale del potenziale gravitazionale

Per ognuna delle 3 stelle trovate (la più massiva per ogni diversa equazione di stato) salviamo ogni valore di r, m e P in un file che successivamente importiamo (riga 3) e utiliziamo per calcolare l'integrale

```
1  double r[lenfile], P[lenfile], m[lenfile], Phi[lenfile];
2
3  read_maxM_data(tipo_politropica, lenfile, r, P, m);
4
5  double R = r[lenfile - 1];
6  double M = m[lenfile - 1];
7  double Phi_ext = fun_Phi_ext(R, M);
8  double integral = 0;
9  double h = 1e-5;
10
11 // Partiamo a calcolare Phi dalla fine (r = R) perche' e' quando l'integrale e' piu' piccolo
12 for (int i = lenfile - 1; i > 0; i--){
13     integral += h / 2 * (fun_to_integrate(r[i], m[i], P[i]) + fun_to_integrate(r[i - 1], m[i - 1], P[i - 1]));
14     Phi[i] = Phi_ext + integral;
15 }
16
17 char Phi_int_filename[50];
18 sprintf(Phi_int_filename, "../data/Phi_int_%d.csv", tipo_politropica);
19 FILE *f_Phi_int = fopen(Phi_int_filename, "w");
20 fprintf(f_Phi_int, "r,Phi\n");
21 for (int i = 0; i < lenfile; i++)
22     fprintf(f_Phi_int, "%.10e,%.10e\n", r[i] * R0, Phi[i]);
23 fclose(f_Phi_int);
24
25 // Calcoliamo anche Phi_ext(r) per r > R
26 double r_ext = R;
27
28 char Phi_ext_filename[50];
29 sprintf(Phi_ext_filename, "../data/Phi_ext_%d.csv", tipo_politropica);
30 FILE *f_Phi_ext = fopen(Phi_ext_filename, "w");
31 fprintf(f_Phi_ext, "r,Phi\n");
32
33 while (r_ext < 150 / R0){
34 // for (int i = 0; i < lenfile; i++){
35     fprintf(f_Phi_ext, "%.10e,%.10e\n", r_ext * R0, fun_Phi_ext(r_ext, M));
36     r_ext += h;
37 }
38
39 fclose(f_Phi_ext);
```

L'integrale (calcolato esplicitamente nella riga 13) può essere valutato solo nei punti r che sono stati utilizzati durante la risoluzione delle equazioni di stabilità della stella.

Per ottimizzare il codice, invece che calcolare l'intero integrale per ogni punto r del grafico di $\Phi(r)$, partiamo da $r = R$ (ovvero quando l'integrale è nullo) e aggiungiamo il valore di 1 trapezio per volta alla variabile `integral`. In contemporanea, durante una iterazione del ciclo `for` possiamo utilizzare il valore (parziale) di `integral` per calcolare Φ_{int} in r.

Infine dalla riga 25 calcoliamo Φ_{ext} utilizzando la funzione analitica.

D Integrali con trapezi e simpson

Le funzioni che fanno l'integrale con i due metodi sono

```
// Metodo Trapezi
double integrale_trapezio(double a, double b, int N, double r, double R, double M,
    double (*fun)(double, double, double, double)){
    // assume a < b
    double h = (b - a)/ (double)N;
    double integral = ((*fun)(a, r, R, M) + (*fun)(b, r, R, M)) * h / 2;

    for (int i = 1; i < N; i++) {
    integral += h * (*fun)(a + i * h, r, R, M);
    }

    return integral;
}

// Metodo Simpson
double integrale_simpson(double a, double b, int N, double r, double R, double M,
    double (*fun)(double, double, double, double)){
    // assume a > b
    double h = (b - a)/N;
    double integral = ((*fun)(a, r, R, M) + (*fun)(b, r, R, M)) * h / 3;

    // assume N pari
    for (int i = 1; i <= N/2 - 1; i++) {
        integral += (2 * h /3) * (*fun)(a + 2 * i * h, r, R, M);
        integral += (4 * h / 3) * (*fun)(a + (2 * i - 1) * h, r, R, M);
    }

    integral += (4 * h / 3 ) * (*fun)(a + (N - 1) * h, r, R, M);

    return integral;
}
```

La funzione da integrare (la radianza) è stata definita come

```
double funB_corrected(double nu, double r, double R, double M){
    double corr;
    // To simulate r = infinity
    if (r < 0)
        corr = pow(1. - 2 * M / R, - 1. / 2);
    else
        corr = pow((1. - 2 * M / r) / (1. - 2 * M / R), 1. / 2);
    nu *= corr;
    return pow(nu, 3) / (exp(nu) - 1.);
}
```

Per controllare la convergenza del valore della potenza usiamo inizialmente `nu_max` (ovvero $\hat{\nu}_{\max}$) uguale a 20. Verifichiamo per quale `N` si ottiene un errore minore di quello voluto prima per i trapezi e poi per Simpson e stampiamo i valori sul terminale.

```
double R[3] = {59.03824 / R0, 10.90280 / R0, 8.559218 / R0};        // Raggi delle 3
    stelle
double M[3] = {14.29963 / M0, 0.9252994 / M0, 1.528782 / M0};       // Masse delle 3
    stelle
int N = 10;
int N_trap, N_simp;
double Pot, nu_max = 20.;
double r = R[0];
double Pot_cvg = integrale_trapezio(1e-10, nu_max, 1e7,  r, R[0], M[0], &
    funB_corrected); // Aggiunto dopo, e' quello ottenuto con N = 1e6
double errore_max = 1e-5;
int kk = 0;

```

```
// Dati per grafico cvg per N trapezi
FILE *f0 = fopen("../data/potenza/test_cvg_N_trap.csv", "w");
fprintf(f0, "Pot,N,nu_max\n");
while(N <= 1e6){
    Pot = integrale_trapezio(1e-10, nu_max, N, r, R[0], M[0], &funB_corrected);
    fprintf(f0, "%.13e,%d,%.13e\n", Pot * POT0, N, nu_max * nu0);

    if ((fabs(Pot - Pot_cvg) / Pot_cvg) < errore_max && kk == 0){
        printf("Per trapezi N = %d e' sufficiente\n", N);
        N_trap = N;
        kk++;
    }

    N *= 1.1;
    if (N % 2 != 0) N += 1;
}
fclose(f0);


N = 10;
kk = 0;
// Dati per grafico cvg per N simpson
FILE *f1 = fopen("../data/potenza/test_cvg_N_simp.csv", "w");
fprintf(f1, "Pot,N,nu_max\n");
while(N <= 1e6){
    Pot = integrale_simpson(1e-12, nu_max, N, r, R[0], M[0], &funB_corrected);
    fprintf(f1, "%.13e,%d,%.13e\n", Pot * POT0, N, nu_max * nu0);

    if ((fabs(Pot - Pot_cvg) / Pot_cvg) < errore_max && kk == 0){
        printf("Per Simpson N = %d e' sufficiente\n", N);
        N_simp = N;
        kk++;
    }

    N *= 1.1;
    if (N % 2 != 0) N += 1;
}
fclose(f1);

// Decidiamo che va bene N_trap e N_simp risultati del codice sopra
Pot_cvg = integrale_trapezio(1e-12, 1000, 1e7, R[0], R[0], M[0], &funB_corrected);
nu_max = 20.;
int Nrel_trap = (double)N_trap / nu_max; // Teniamo la stella densita'
int Nrel_simp = (double)N_simp / nu_max; // di N / nu_max

// partiamo da
nu_max = 10.;
kk = 0;
// Test cvg per A (ovvero nu_max)
FILE *f2 = fopen("../data/potenza/test_cvg_A_trap.csv", "w");
fprintf(f2, "Pot,N,nu_max[ad]\n");
while(nu_max <= 1e2){
    N = Nrel_trap * nu_max;
    if (N % 2 != 0) N += 1;

    Pot = integrale_trapezio(1e-10, nu_max, N, R[0], R[0], M[0], &funB_corrected);
    fprintf(f2, "%.13e,%d,%.13e\n", Pot * POT0, N, nu_max);

    if ((fabs(Pot - Pot_cvg) / Pot_cvg) < errore_max && kk == 0){
        printf("Per trapezi N = %d, nu_max = %.2f sono sufficienti\n", N, nu_max);
        kk++;
    }

    nu_max *= 1.1;
}
fclose(f2);

nu_max = 10.;
kk = 0;
// Test cvg per A (ovvero nu_max)
```

```
81 FILE *f3 = fopen("../data/potenza/test_cvg_A_simp.csv", "w");
82 fprintf(f2, "Pot,N,nu_max[ad]\n");
83 while(nu_max <= 1e2){
84     N = Nrel_simp * nu_max;
85     if (N % 2 != 0) N += 1;
86
87     Pot = integrale_simpson(1e-10, nu_max, N, R[0], R[0], M[0], &funB_corrected);
88     fprintf(f3, "%.13e,%d,%.13e\n", Pot * POT0, N, nu_max);
89
90     if ((fabs(Pot - Pot_cvg) / Pot_cvg) < errore_max && kk == 0){
91         printf("Per Simpson N = %d, nu_max = %.2f sono sufficienti\n", N, nu_max);
92         kk++;
93     }
94
95     nu_max *= 1.1;
96 }
97 fclose(f3);
```

In `stdout` otteniamo

```
Per trapezi N = 62 e' sufficiente
Per Simpson N = 82 e' sufficiente
Per trapezi N = 62, nu_max = 20.7892818 sono sufficienti
Per Simpson N = 72, nu_max = 17.9585633 sono sufficienti
```

Dalla riga 50 viene utilizzato un codice molto simile a quello sopra controllare la convergenza per diversi valori di `nu_max`, in più bisogna assicurarsi che `N / nu_max` rimanga costante al variare di `nu_max`. Inoltre invece che integrare per $r = \infty$ integriamo per $r = R_1$, ovvero il caso in cui la curva della radianza è più spostata verso destra e che quindi richiede di integrare fino a un `nu_max` maggiore.