

Indice

1	Introduzione	1
2	Stabilità	1
3	Curva massa raggio	2
4	Potenziale gravitazionale	3
A	RK4	4
B	Risoluzione numerica di $\hat{P}(\rho)$	4

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 gravitazionale 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

$$\left\{ \begin{array}{l} \frac{dP(r)}{dr} = -G \frac{m(r)\epsilon(r)}{r^2 c^2} \left(1 + \frac{P(r)}{\epsilon(r)} \right) \left(1 + \frac{4\pi r^3 P(r)}{m(r)c^2} \right) \left(1 - \frac{2Gm(r)}{rc^2} \right)^{-1} \end{array} \right. \quad (1a)$$

$$\left\{ \begin{array}{l} \frac{dm(r)}{dr} = 4\pi r^2 \frac{\epsilon(r)}{c^2} \end{array} \right. \quad (1b)$$

$$\left\{ \begin{array}{l} \frac{d\Phi(r)}{dr} = -\frac{1}{P(r) + \epsilon(r)} \frac{dP(r)}{dr} \end{array} \right. \quad (1c)$$

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):

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

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

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

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

$$P = -\frac{dE}{dV} \Rightarrow \left\{ \begin{array}{l} P = (\alpha - 1)a \left(\frac{n}{n_0} \right)^\alpha + (\beta - 1)b \left(\frac{n}{n_0} \right)^\beta \end{array} \right. \quad \text{per } \epsilon_1 \quad (4a)$$

$$\left\{ \begin{array}{l} n = \left(\frac{P}{K(\Gamma - 1)c^2} \right)^{1/\Gamma} \end{array} \right. \quad \text{per } \epsilon_{1/2} \quad (4b)$$

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.

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

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

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

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

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.

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

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 equazione di stato più complessa (eq. 2a) viene risolta numericamente l'equazione 4a per trovare ρ da P (Appendice B).

Scegliamo 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.

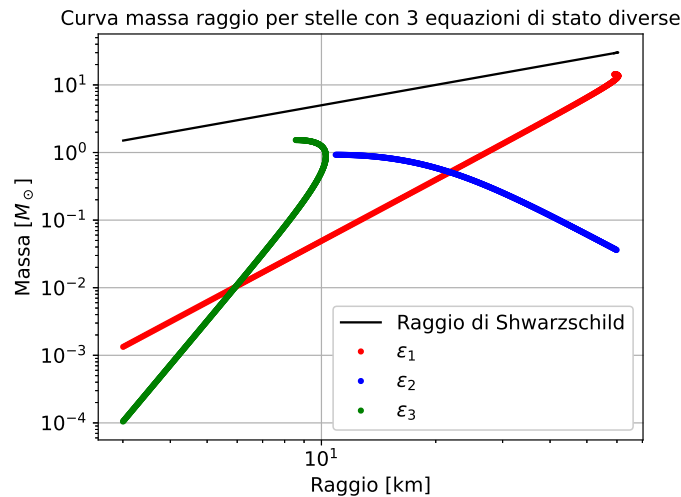


Figura 1: Curva massa raggio per stelle di equazioni di stato $\epsilon_{1/2/3}$. La prima equazione di stato, quella più realistica, prevede stelle di neutroni più massive.

Durante l'esecuzione del programma abbiamo inoltre smesso di incrementare la pressione centrale iniziale quando la condizione di stabilità $\frac{dM}{dr} > 0$ 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**.

	$P_0 \left[\frac{\text{MeV}}{c^2 \text{fm}} \right]$	$R \text{ [km]}$	$M \text{ [} M_\odot \text{]}$
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'equazione 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**.

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

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

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

Dati i valori di $\hat{P}(\hat{r})$ e $\hat{m}(\hat{r})$ che si ottengono risolvendo le equazioni di stabilità possiamo quindi disegnare il grafico del potenziale gravitazionale, mostrato in figura **2**.

Figures/Phi-eps-converted-to.pdf

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

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.

```

1 void rungeKutta4(double h, double r, double *P, double *m, int tipo_politropica){
2
3     double k1, k2, k3, k4, l1, l2, l3, l4;
4
5     k1 = h * fun_m(r, *P, tipo_politropica);
6     l1 = h * fun_P(r, *P, *m, tipo_politropica);
7
8     k2 = h * fun_m(r + h / 2, *P + l1 / 2, tipo_politropica);
9     l2 = h * fun_P(r + h / 2, *P + l1 / 2, *m + k1 / 2, tipo_politropica);
10
11    k3 = h * fun_m(r + h / 2, *P + l2 / 2, tipo_politropica);
12    l3 = h * fun_P(r + h / 2, *P + l2 / 2, *m + k2 / 2, tipo_politropica);
13
14    k4 = h * fun_m(r + h, *P + l3, tipo_politropica);
15    l4 = h * fun_P(r + h, *P + l3, *m + k3, tipo_politropica);
16
17    *m += (k1 + 2 * k2 + 2 * k3 + k4) / 6;
18    *P += (l1 + 2 * l2 + 2 * l3 + l4) / 6;
19 }
```

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

```

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

```

4 }
5
6
7 // fun_P = - (P + E)(m + r^3 P)/(r^2 - 2mr)
8 double fun_P(double r, double P, double m, int tipo_politropica){
9     if (m == 0)
10         return 0;
11     return (P + fun_E(P, tipo_politropica)) * (m + pow(r, 3) * P) / ((2 * m - r) * r);
12 }

```

Dove la funzione `fun_E()` è definita come

```

1 double fun_E(double P, int tipo_politropica){
2     // Politropica quasi realistica (a*rho^alpha + b*rho^beta)
3     if (tipo_politropica == 0){
4         double rho = findRho(P);
5         return A * pow(rho, ALPHA) + B * pow(rho, BETA);
6     }
7
8     double lambda, K;
9
10    // Politropiche semplici
11    if (tipo_politropica == 1){
12        lambda = 5. / 3.;
13        K = 0.05;
14    } else if (tipo_politropica == 2){
15        lambda = 2.54;
16        K = 0.01;
17    } else {
18        printf("Tipo politropica non riconosciuto\n");
19        return 0;
20    }
21
22    double a1 = P / (lambda - 1);
23    return a1 + pow(a1 / K, 1. / lambda);
24 }

```

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

$$P = (\alpha - 1)a \left(\frac{n}{n_0} \right)^\alpha + (\beta - 1)b \left(\frac{n}{n_0} \right)^\beta. \quad (10)$$

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

```

1 double findRho(double P){
2
3     // Cominciamo prima con il metodo di Newton-Raphson
4     if (P > 0.01){
5
6         double rho = pow(P / (BETA1 * B), 1 / BETA); // buona approx. iniziale
7         while (fabs(P - P_of_rho(rho)) > 1e-6){
8             rho -= (P_of_rho(rho) - P) / DP_of_rho(rho);
9         }
10        return rho;
11    }
12
13    // Per P piccoli meglio usare bisezione
14    double rho_sx = 0.7, rho_dx = 1.;
15    double rho = (rho_sx + rho_dx) / 2;
16    while (fabs(P - P_of_rho(rho)) > 1e-6){
17        if (P_of_rho(rho) > P)

```

```

18         rho_dx = rho;
19     else
20         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 **10** e la sua derivata. In questo modo possiamo sempre trasformare $\epsilon(\rho)$ in $\epsilon(P)$.