



UNIVERSITÀ DEGLI STUDI DI MILANO
FACOLTÀ DI SCIENZE E TECNOLOGIE

Corso di Laurea Triennale in Fisica

**Applicazioni della Regressione
Simbolica ai neutrini massivi in
cosmologia**

Relatore: **Prof. Luigi Guzzo**
Correlatore: **Dott. Marco Bonici**

Tesi di Laurea di:
Stefano Carturan
Matricola: **963991**

Anno Accademico 2022/2023

Indice

Introduzione	I
1 Cosmologia	1
1.1 Relatività Generale	1
Dinamica dell'Universo	3
Legge di Hubble	4
Costante cosmologica	5
Densità critica	6
Redshift cosmologico	7
1.2 Modelli cosmologici	9
Materia oscura	9
Big Bang e CMB	11
Modello Λ CDM	12
1.3 Formazione delle strutture	14
Instabilità di Jeans	15
Crescita delle perturbazioni	17
1.4 Spettro di potenza della densità di materia	18
Funzioni di correlazione	18
Spettore delle perturbazioni	19
Spettore primordiale	20
Fattore di crescita e neutrini massivi	20
1.5 Equazioni di Einstein-Boltzmann	21
2 Machine Learning	23
2.1 Reti Neurali	24
Allenamento di una rete neurale	26
Backpropagation	28
2.2 Regressione Simbolica	30
Algoritmi genetici	31
PySR	33

3 Costruzione dell’emulatore per lo spettro di potenza	36
3.1 Regressione Simbolica sul fattore di crescita	36
Generazione dei dataset	37
Costruzione dei modelli	39
Test sulla dipendenza dalle condizioni iniziali	40
Scelta del modello	41
3.2 Rete neurale e ricostruzione dello spettro	43
Generazione dei dataset	43
Caratteristiche e training della rete neurale	43
Ricostruzione dello spettro	44
3.3 Risultati ottenuti	45
Rilevanza della componente simbolica dell’emulatore	47
Conclusioni	49

Introduzione

L’Universo odierno presenta delle disomogeneità in termini di densità, le quali si manifestano come strutture aggregate di materia su varie scale: i pianeti, le galassie e gli ammassi di galassie ne sono alcuni esempi. Tali disomogeneità, che al giorno d’oggi si discostano di vari ordini di grandezza dalla densità di background, sono il risultato dell’evoluzione di piccole fluttuazioni originate nell’Universo primordiale, come testimoniato dalla presenza di anisotropie nella radiazione cosmica di fondo dell’ordine di $\Delta T/T \approx 10^{-5}$.

Al fine di descrivere la distribuzione statistica delle disomogeneità, così come la loro evoluzione in termini della scala spaziale e del redshift, in cosmologia si definisce lo spettro di potenza $P(k, z)$, il quale permette di studiare le fluttuazioni di densità in termini delle loro componenti di Fourier in spazio k . La dipendenza da k dello spettro di potenza è descritta dalla funzione di trasferimento $T(k)$, mentre la dipendenza temporale dei modi di Fourier su grandi scale, che in cosmologia si traduce in una dipendenza dal redshift z , viene delineata dal fattore di crescita $D(z)$. Nella determinazione di quest’ultimo i neutrini giocano un ruolo chiave: infatti, se si tiene conto del valore finito della massa di queste particelle, nel fattore di crescita viene introdotta una dipendenza da k , che smorza lo spettro di potenza sulle piccole scale.

Per il calcolo di $T(k)$ e $D(k, z)$, data l’assenza di soluzioni esatte alle equazioni di Einstein-Boltzmann che descrivono $P(k, z)$, in cosmologia sono utilizzati dei software, chiamati Einstein-Boltzmann solvers, i quali integrano numericamente tali equazioni. Recentemente, tuttavia, lo sviluppo delle tecniche di Machine Learning ha aperto la strada a metodi più efficienti per il calcolo di queste quantità. Si costruiscono cioè degli emulatori, codici basati su metodi di Machine Learning come le Reti Neurali, che apprendono la dipendenza di $T(k)$ e $D(k, z)$ dai vari parametri cosmologici forniti dai modelli teorici. Sebbene gli emulatori costruiti con le Reti Neurali siano molto efficienti e permettano di raggiungere ottimi livelli di precisione, essi sono dei modelli “black-box”, il cui output fornisce poche informazioni sulle dipendenze fisiche dai vari parametri. Esiste tuttavia un’altra tecnica di Machine Learning,

Introduzione

detta Regressione Simbolica, la quale esplora lo spazio delle funzioni analitiche dei parametri di input al fine di trovare quella che meglio rappresenti i dati, basandosi sui cosiddetti “algoritmi genetici”. L’utilizzo di tale tecnica nella costruzione degli emulatori permette di aumentare l’interpretabilità fisica dei modelli, e potenzialmente di generalizzare meglio il risultato ottenuto, costituito da una semplice equazione simbolica.

Lo scopo di questo progetto di tesi sarà dunque quello di costruire un emulatore neuro-simbolico per lo spettro di potenza delle fluttuazioni di densità $P(k, z)$ e confrontare la sua performance con quella di modelli più semplici basati sulla sola applicazione delle Reti Neurali. Perciò si utilizzerà la Regressione Simbolica al fine di trovare un’espressione analitica per la correzione $\mu(k, z)$ che i neutrini apportano al fattore di crescita. Tale espressione sarà poi utilizzata nel preprocessing del training dataset di una rete neurale, mediante la quale si ricostruirà lo spettro di potenza. Quest’ultimo verrà infine paragonato allo spettro ottenuto da un emulatore più elementare, costituito semplicemente da una rete neurale, al fine di verificare se il preprocessing effettuato mediante la Regressione Simbolica abbia apportato migliorie alla precisione a parità di condizioni di training.

Capitolo 1

Cosmologia

La **cosmologia** è la branca della Fisica che studia l'origine dell'Universo e la sua evoluzione fino al giorno d'oggi. In particolare, essa pone larga attenzione allo studio delle fluttuazioni di densità dell'Universo primordiale, le quali sono evolute nel tempo ed hanno dato vita alle strutture osservate al giorno d'oggi. Questa disciplina, dal punto di vista osservativo, studia quantità come la radiazione cosmica di fondo e la distribuzione delle galassie su grande scala per ottenere informazioni riguardanti l'evoluzione dell'Universo nel suo insieme. Per rendere ragione di tutte le osservazioni effettuate, nel corso della storia si sono messi a punto svariati modelli cosmologici, portando al cosiddetto modello standard della cosmologia noto anche come Λ CDM, che descrive formalmente l'evoluzione del Cosmo nel quadro globale della Relatività Generale. Il punto di incontro tra l'approccio teorico e quello osservativo in cosmologia è spesso costituito dalla costruzione di simulazioni numeriche, tramite le quali è possibile calcolare le osservabili cosmologiche previste da un certo modello, confrontandole con i dati sperimentali.

1.1 Relatività Generale

La cosmologia fa ampio uso della teoria della **Relatività Generale** (GR) per descrivere l'evoluzione dell'Universo. Secondo questa teoria, la metrica dell'Universo è legata al suo tensore energia-momento dalle *equazioni di campo di Einstein* [1], che in unità naturali assumono la forma:

$$G_{\mu\nu} = 8\pi T_{\mu\nu} . \quad (1.1)$$

In tale espressione, $T_{\mu\nu}$ contiene l'informazione sulla distribuzione dell'energia e il tensore di Einstein $G_{\mu\nu}$ è definito da:

$$G_{\mu\nu} \equiv R_{\mu\nu} - \frac{1}{2}R g_{\mu\nu}, \quad (1.2)$$

dove $R \equiv R^\nu_\nu$ è lo scalare di curvatura, $g_{\mu\nu}$ è il tensore metrico e $R_{\mu\nu} \equiv R^\lambda_{\mu\lambda\nu}$ è il tensore di Ricci, ottenuto contraendo il tensore di curvatura di Riemann $R^\sigma_{\mu\lambda\nu}$ sul primo e sul terzo indice. Quest'ultimo descrive la curvatura di una varietà differenziabile riemanniana o pseudo-riemanniana, quale è ad esempio lo spaziotempo in cui viviamo.

Al fine di applicare la teoria della Relatività allo studio dell'Universo, è necessario conoscere quale sia la soluzione delle equazioni (1.1) che lo descrive, i.e. occorre trovare la metrica dello spaziotempo. A priori, risolvere le equazioni di Einstein è un compito arduo, che tuttavia può essere notevolmente semplificato da alcune osservazioni sulla simmetria del nostro Universo. I dati provenienti dalle recenti survey di galassie [2], che stanno esplorando il Cosmo a redshift sempre maggiore, suggeriscono che esso sia omogeneo, ovvero presenti le stesse caratteristiche in ogni suo punto, ed isotropo, cioè privo di direzioni spaziali privilegiate. Infatti, via via che le nuove tecnologie permettono di osservare l'Universo più in profondità, la distribuzione spaziale delle galassie appare sostanzialmente identica in ogni direzione. L'isotropia dell'Universo implica che la curvatura k dello spaziotempo sia costante e, grazie al teorema di Eisenhart [3], è sufficiente conoscerne il segno per determinare la metrica. Sotto l'assunzione di omogeneità e isotropia, la quale prende il nome di *Principio Cosmologico*, la metrica dell'Universo è quella di Friedmann-Lemaître-Robertson-Walker (FLRW):

$$\begin{aligned} ds^2 &= -d\tau^2 + a^2(\tau) d\Omega_k = \\ &= -d\tau^2 + a^2(\tau) [d\psi^2 + f_k^2(\psi) (d\theta^2 + \sin^2(\theta)d\varphi^2)], \end{aligned} \quad (1.3)$$

in cui τ è il tempo proprio, $a(\tau)$ è detto fattore di scala e k è la curvatura dello spaziotempo, che può valere solamente -1 , 0 o $+1$, in quanto $a(\tau)$ assorbe la sua dipendenza temporale. A seconda del valore di k si hanno le seguenti possibili forme di $f_k(\psi)$:

$$f_k(\psi) = \begin{cases} \sinh(\psi) & \text{se } k = -1 \\ \psi & \text{se } k = 0 \\ \sin(\psi) & \text{se } k = 1 \end{cases}. \quad (1.4)$$

Sostituendo la metrica (1.3) nelle equazioni di Einstein (1.1), queste si riducono alle due *equazioni di Friedmann*, che descrivono la dinamica di un Universo omogeneo ed isotropo:

$$\begin{cases} \frac{3\dot{a}^2}{a^2} = 8\pi\rho - \frac{3k}{a^2} \\ \frac{3\ddot{a}}{a} = -4\pi(\rho + 3P) \end{cases}, \quad (1.5)$$

dove ρ e P sono rispettivamente la densità di energia e la pressione dell’Universo, modellizzato come un fluido, e il punto indica la derivata rispetto al tempo proprio τ . Esplicitando un’ultima condizione data dall’equazione di stato $P = P(\rho)$, il sistema (1.5) può essere risolto. Si noti subito che se ρ e P sono positive non esiste una soluzione costante per $a(\tau)$: in altre parole l’Universo è necessariamente in espansione o in contrazione.

Dinamica dell’Universo

Per risolvere le equazioni di Friedmann (1.5) è necessario introdurre un’*equazione di stato* che leggi P e ρ , per ognuno dei costituenti dell’Universo: materia, radiazione ed energia oscura. Se la componente principale dell’Universo è la *materia* contenuta nelle galassie, le quali sono solitamente rappresentate come una “polvere”, l’equazione di stato è $P = 0$, in quanto questa polvere ha pressione nulla. In tal caso, per uno spaziotempo piatto ($k = 0$) si ha:

$$\begin{cases} \rho_m \propto \frac{1}{a^3} \\ a(\tau) \propto \tau^{\frac{2}{3}} \end{cases}, \quad (1.6)$$

dove l’Universo è in espansione poiché il fattore di scala cresce come una potenza e la densità di materia ρ_m decresce come a^{-3} . In un Universo piatto dominato dalla *radiazione*, che in cosmologia indica l’insieme di tutte le particelle relativistiche, l’equazione di stato è $P = \rho/3$, e vale:

$$\begin{cases} \rho_r \propto \frac{1}{a^4} \\ a(\tau) \propto \tau^{\frac{1}{2}} \end{cases}, \quad (1.7)$$

in cui si nota che l’espansione è ancora descritta da una legge di potenza e la densità ρ_r di radiazione decresce più rapidamente rispetto al caso in cui domina

la materia. Se infine la componente predominante nell’Universo è l’*energia oscura*, normalmente rappresentata dalla costante cosmologica Λ (> 0), si ha come equazione di stato la relazione $P = -\rho$ e inoltre:

$$\begin{cases} \rho_\Lambda = \frac{\Lambda}{8\pi} = \text{cost.} \\ a(\tau) \propto \exp\left(\sqrt{\frac{\Lambda}{3}}\tau\right) \end{cases}, \quad (1.8)$$

da cui si evince che il fattore di scala cresce esponenzialmente.

Molti modelli evolutivi dell’Universo ne individuano l’origine nel cosiddetto Big Bang: inizialmente tutto si trovava concentrato in uno stato estremamente caldo e denso, che successivamente ha subìto una rapidissima fase di espansione e raffreddamento chiamata inflazione. Dopo questa fase, le tre componenti dell’Universo si sono alternate nella determinazione della sua dinamica. Come si può notare dalle equazioni precedenti, durante la prima fase la densità predominante è quella della radiazione ($\rho_r \propto a^{-4}$), che cede il posto alla materia ($\rho_m \propto a^{-3}$) dopo circa 47 000 anni dal Big Bang. Infine, l’epoca di prevalenza della materia termina a 9.8 miliardi di anni dal Big Bang, poiché da quel momento fino al giorno d’oggi la densità dominante è quella relativa alla costante cosmologica. Come si intuisce già dalla (1.8), l’espansione dell’Universo dominato dalla costante cosmologica Λ è accelerata, come sarà approfondito in seguito.

Legge di Hubble

L’espansione dell’Universo, ottenuta dalla risoluzione delle equazioni di Friedmann, è stata osservata per la prima volta da Edwin Hubble nel 1929 [4] durante i suoi studi sulle nebulae. Hubble osservò che le galassie distanti presentavano un redshift (si veda in seguito) più marcato rispetto a quelle vicine, ad indicare una maggiore velocità di deriva rispetto alla posizione della terra. Questo fatto è dovuto non ad un vero e proprio allontanamento, ma all’espansione dello spazio stesso. Infatti, la terra e le galassie osservate da Hubble si trovano sulla stessa ipersuperficie di tipo tempo, caratterizzata da $d\tau = 0$, e una distanza generica su tale ipersuperficie al tempo proprio τ è data dall’integrale della metrica di FLRW (1.3):

$$r(\tau) = a(\tau) \int d\Omega, \quad (1.9)$$

mentre la stessa distanza al tempo proprio $\tau + d\tau$ vale:

$$r(\tau + d\tau) = a(\tau + d\tau) \int d\Omega = a(\tau + d\tau) \frac{r(\tau)}{a(\tau)} . \quad (1.10)$$

Tramite uno sviluppo di Taylor al primo ordine della (1.10) si ottiene la legge di Hubble, che determina la velocità di espansione dell'Universo, ovvero il tasso di allontanamento di due punti presi sulla stessa ipersuperficie di tipo tempo:

$$v = \frac{dr}{d\tau} = \frac{\dot{a}}{a} r = H r . \quad (1.11)$$

Il parametro $H \equiv \dot{a}/a$ è detto *costante di Hubble* e si misura in km/s/Mpc. È conveniente definire anche la costante di Hubble ridotta, $h \equiv H_0/(100 \text{ km s}^{-1} \text{ Mpc}^{-1})$, in cui H_0 è il valore attuale della costante di Hubble, pari a circa 67.66 km s⁻¹ Mpc⁻¹ [5].

Costante cosmologica

La legge di Hubble costituisce un grande successo della teoria di Einstein, il quale tuttavia aveva concepito la *costante cosmologica* per ottenere una soluzione statica delle sue equazioni, rifiutando l'idea che l'Universo potesse espandersi o contrarsi. Dunque il fisico tedesco scartò Λ , considerandola “il più grave errore” della sua vita. Tuttavia, nel 1998, osservazioni delle supernove mostrarono che l'espansione dell'Universo è accelerata [6], al contrario di quanto dovrebbe essere in caso di prevalenza della materia (1.6) o della radiazione (1.7). Questo permise di concludere che l'espansione dell'Universo attualmente è dominata da un altro costituente, chiamato *energia oscura* e descritto proprio dalla costante cosmologica, la cui densità rappresenta il 74% di quella totale. In questo scenario, le equazioni di Einstein (1.1) assumono la forma:

$$G_{\mu\nu} = 8\pi T_{\mu\nu} + \Lambda g_{\mu\nu} , \quad (1.12)$$

e, assumendo un'equazione di stato generale:

$$P = w\rho \quad w = \begin{cases} 0 & \text{materia} \\ \frac{1}{3} & \text{radiazione} \\ -1 & \text{costante cosmologica} \end{cases} , \quad (1.13)$$

dalle equazioni di Friedmann (1.5) si ottiene:

$$a(\tau) = \left(\frac{3(1+w)}{2} \sqrt{\frac{8\pi C}{3}} (\tau - \tau_0) \right)^{\frac{2}{3(1+w)}}. \quad (1.14)$$

Nell'equazione (1.14), w , C e τ_0 sono costanti di integrazione e si è trascurato il termine di curvatura k per grandi valori del fattore di scala. Da questa relazione si nota che, per $w < -1$, il fattore di scala $a(\tau)$ diverge ad un tempo proprio finito ($\tau \rightarrow \tau_0$): questo è il cosiddetto “Big Rip”, che, secondo i dati forniti dalle osservazioni attuali, sembra il destino più probabile del nostro Universo (Figura 1.1).

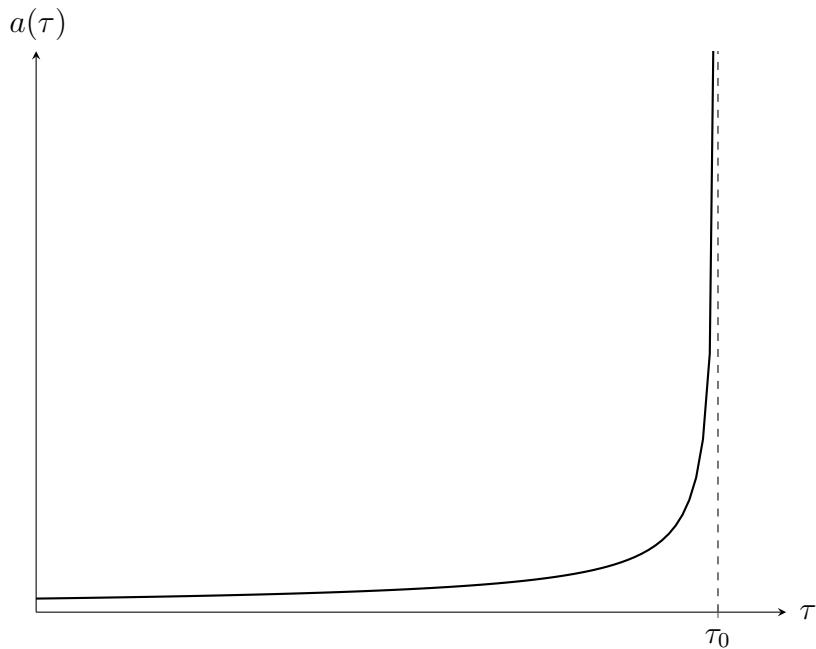


Figura 1.1: Tenendo conto della presenza di una costante cosmologica Λ con equazione di stato $P \lesssim -\rho$, la cui densità domina l'Universo attuale, le equazioni di Friedmann restituiscono un fattore di scala $a(\tau)$ che diverge in un tempo proprio finito (“Big Rip”).

Densità critica

Finora si è considerata la densità di energia associata alle varie componenti dell'Universo, tuttavia in cosmologia è utile definire un parametro, chiamato

densità critica, come:

$$\rho_c \equiv \frac{3H^2}{8\pi} . \quad (1.15)$$

Essa è utile per definire i parametri di densità adimensionali:

$$\Omega_i \equiv \frac{\rho_i}{\rho_c} \quad i \in I = \{m, r, \Lambda, k\} , \quad (1.16)$$

dove si è implicitamente definita una “densità di curvatura”:

$$\rho_k \equiv \frac{k}{a^2 H^2} . \quad (1.17)$$

È chiaro che con l'utilizzo delle Ω_i la prima delle equazioni di Friedmann (1.5) assume la forma:

$$\sum_{i \in I} \Omega_i = 1 \quad I = \{m, r, \Lambda, k\} . \quad (1.18)$$

La relazione (1.18) è anche utilizzata in cosmologia per misurare la curvatura dello spaziotempo (e in particolare il suo segno) a partire dalla conoscenza delle densità di materia, radiazione ed energia oscura.

Redshift cosmologico

Una delle osservabili più importanti nell'ambito della cosmologia è il redshift, definito dal seguente rapporto:

$$z \equiv \frac{\lambda_o - \lambda_e}{\lambda_e} = \frac{\omega_e - \omega_o}{\omega_o} , \quad (1.19)$$

dove λ_o e λ_e sono rispettivamente le lunghezze d'onda osservata ed emessa di un fotone, mentre ω_o e ω_e sono le relative frequenze. Questa grandezza fisica adimensionale ha un profondo significato in cosmologia, in quanto è strettamente connessa al fattore di scala che compare nella metrica di FLRW (1.3) [7]. Per comprendere adeguatamente la relazione tra z e $a(\tau)$, occorre introdurre una tipologia particolare di quadrivettori, detti vettori di Killing, i quali costituiscono un campo vettoriale che conserva la metrica di una varietà differenziabile

pseudo-riemanniana. Nello specifico, è possibile mostrare che, data una geodetica γ con quadrivettore tangente u^μ e ξ^μ vettore di Killing, la quantità $u_\mu \xi^\mu$ si conserva lungo γ [1]. Si consideri ora la frequenza di un fotone emesso in un certo punto p_1 ed osservato in p_2 , che viaggia lungo una geodetica di tipo luce con quadrivettore d'onda k^μ e quadrivelocità u^μ :

$$\omega_{1,2} = -k_\mu u^\mu \Big|_{p_1, p_2} . \quad (1.20)$$

Date due ipersuperfici di tipo tempo $\Sigma_1 \ni p_1$ e $\Sigma_2 \ni p_2$, è possibile mostrare che esiste sempre un vettore di Killing ξ^μ che in p_1 punta nella direzione della proiezione di k^μ su Σ_1 e in p_2 è parallelo alla proiezione di k^μ su Σ_2 . Tale vettore può essere scelto in modo tale che il quadrivettore d'onda possa essere scomposto come segue:

$$k_\mu = \alpha \left(u_\mu + \frac{\xi_\mu}{\sqrt{\xi^2}} \right) , \quad (1.21)$$

dove con ξ^2 si intende $\xi_\nu \xi^\nu$ e α è una costante moltiplicativa. Inoltre, si può calcolare la norma di ξ^μ in p_1 o in p_2 utilizzando la metrica relativa all'ipersuperficie Σ_1 o Σ_2 , e vale che:

$$\sqrt{\frac{\xi_\nu \xi^\nu \Big|_{p_1}}{\xi_\nu \xi^\nu \Big|_{p_2}}} = \frac{a(\tau_1)}{a(\tau_2)} . \quad (1.22)$$

A questo punto, moltiplicando entrambi i membri della (1.21) per k^μ e ricordando che la norma di un vettore di tipo luce è $k_\mu k^\mu = 0$, è possibile ottenere:

$$0 = u_\mu k^\mu + \frac{\xi_\mu k^\mu}{\sqrt{\xi^2}} \implies -u_\mu k^\mu = \frac{\xi_\mu k^\mu}{\sqrt{\xi^2}} \implies \omega_{1,2} = \frac{\xi_\mu k^\mu}{\sqrt{\xi^2}} \Big|_{p_1, p_2} . \quad (1.23)$$

Utilizzando la relazione (1.22) assieme al fatto che $\xi_\mu k^\mu \Big|_{p_1} = \xi_\mu k^\mu \Big|_{p_2}$ poiché ξ^μ è di Killing e k^μ è tangente ad una geodetica, si conclude che:

$$\frac{\omega_2}{\omega_1} = \frac{a(\tau_1)}{a(\tau_2)} \implies z = \frac{\omega_1}{\omega_2} - 1 = \frac{a(\tau_2)}{a(\tau_1)} - 1 . \quad (1.24)$$

Perciò, se si considera $a(\tau_2) = 1$, che corrisponde al valore odierno del fattore di scala, si ha:

$$a(\tau) = \frac{1}{1+z} , \quad (1.25)$$

e dunque una misura del redshift di un oggetto cosmologico è equivalente a misurare $a(\tau)$ al tempo proprio in cui l'oggetto ha emesso la radiazione osservata.

1.2 Modelli cosmologici

In questa sezione si discutono alcuni ingredienti fondamentali del nostro Universo, la materia oscura e la radiazione cosmica di fondo, per poi arrivare alla descrizione del modello Λ CDM, che attualmente è il modello di concordanza in cosmologia per quanto riguarda l'evoluzione del Cosmo, assumendo l'ipotesi del Big Bang e la presenza di una costante cosmologica positiva che governa l'odierna espansione accelerata.

Materia oscura

Nella sezione precedente sono stati presi in considerazione i costituenti principali dell'Universo: la materia, costituita da tutte le particelle massive con velocità molto minore di quella della luce nel vuoto, la radiazione, termine con il quale si indicano le particelle prive di massa o quelle massive che si muovono a velocità relativistiche, e la costante cosmologica. Tuttavia in cosmologia è utile effettuare un'ulteriore distinzione: quella tra la *materia barionica* e la *materia oscura* [8]. Con il primo termine si indica tutta la materia ordinaria, i.e. quella composta dalle particelle massive del modello standard (non solo dai barioni, ai quali però si deve il maggior contributo in termini di densità), mentre la materia oscura è composta per definizione da particelle che non emettono radiazione elettromagnetica, ma sono rilevabili solamente tramite i loro effetti gravitazionali. Queste particelle costituiscono circa l'85% della materia dell'Universo e la loro esistenza è stata ipotizzata a seguito di numerose evidenze provenienti dall'astrofisica osservativa, come ad esempio la curva delle velocità stellari all'interno delle galassie a spirale. Misurando la velocità di rotazione $v(r)$ delle stelle vicine al confine luminoso di molte galassie a spirale, ci si è resi conto che essa non segue la seconda legge di Keplero, che ne prevederebbe una decrescita proporzionale a $r^{-\frac{1}{2}}$ in funzione della distanza r

dal centro, ma rimane con buona approssimazione costante, come mostrato in Figura 1.2. Questo fatto può essere dovuto alla presenza di ulteriore massa, che aumenta linearmente con r via via che ci si allontana dal centro della galassia, dovuta proprio alla materia oscura.

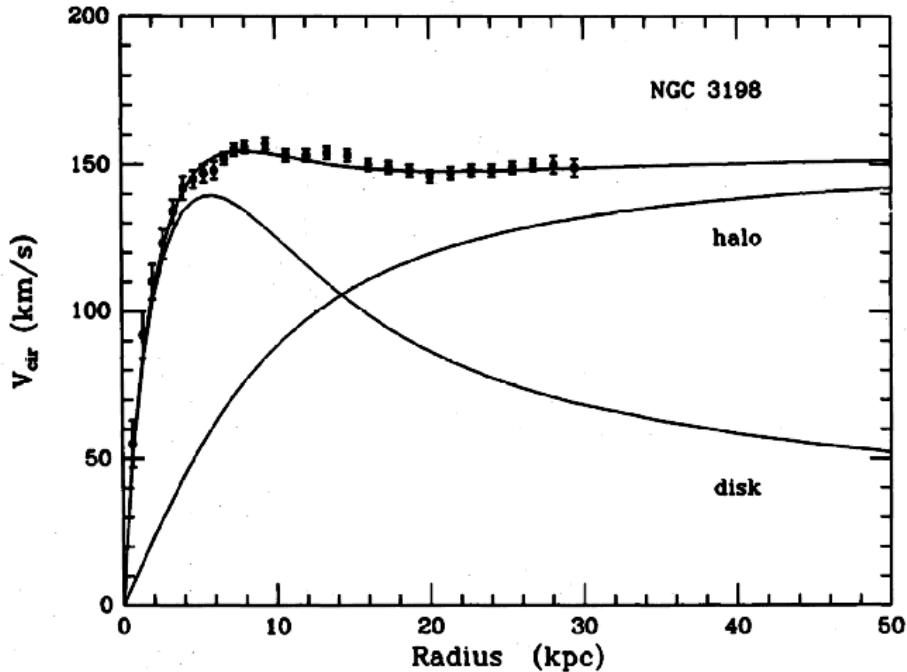


Figura 1.2: Curve di velocità di rotazione delle stelle nella periferia della galassia NGC 3198 in funzione della distanza dal centro della galassia. Invece della decrescita kepleriana proporzionale a $r^{-\frac{1}{2}}$ attesa (disk) si osserva che le velocità rimangono costanti, suggerendo la presenza di ulteriore massa (halo). Crediti: [9].

Esistono varie ipotesi sulla composizione della materia oscura, che viene genericamente distinta in barionica e non barionica. La materia oscura barionica formerebbe tipicamente oggetti compatti, come nane brune, buchi neri di massa stellare e pianeti, genericamente indicati come MACHOs (MAssive Compact Halo Objects) e cercati attraverso fenomeni di microlensing gravitazionale. Un'altra tipologia di materia oscura barionica potrebbe essere quella contenuta nel gas caldo intergalattico. Tuttavia le osservazioni sperimentali suggeriscono che la quantità di materia oscura in queste forme non è sufficiente a spiegare le curve di rotazione, dunque la maggior parte di essa deve essere di natura non barionica. Questo tipo di materia oscura si divide a sua volta in due tipologie, “calda” (Hot Dark Matter, HDM) e “fredda” (Cold Dark Matter,

CDM), formate rispettivamente da particelle relativistiche di piccola massa e da particelle lente molto massive. I neutrini, avendo una massa inferiore ad 1 eV, sono fra i principali candidati per la HDM, mentre per spiegare la presenza di CDM si ipotizza l'esistenza degli WIMPs (Weakly Interacting Massive Particles). Capire quale di queste ultime due tipologie sia la componente principale della materia oscura è molto utile al fine di costruire modelli cosmologici che spieghino l'evoluzione del nostro Universo. Ad esempio, nel caso domini la componente di HDM, si formerebbero prima le strutture su grande scala e poi per disgregazione quelle su piccola scala, mentre la CDM formerebbe prima strutture su piccola scala, che poi aggregandosi darebbero vita a quelle su scale maggiori.

Big Bang e CMB

L'espansione dell'Universo attuale suggerisce l'ipotesi che ad un certo istante nel passato tutto dovesse essere concentrato in un punto: questa è l'idea alla base del modello del Big Bang. Una delle evidenze più significative a favore di questo modello è l'esistenza di una radiazione cosmica di fondo, detta *Cosmic Microwave Background* (CMB), osservata per la prima volta da Arno Penzias e Robert W. Wilson nel 1964 [10]. Tale radiazione, anche detta radiazione fossile, è stata emessa 380 000 anni dopo il Big Bang, all'epoca di ricombinazione, e costituisce la prima luce visibile dell'Universo [11]. Infatti, prima della ricombinazione, l'Universo era composto da un plasma estremamente denso e caldo in cui materia e radiazione erano accoppiate: i fotoni erano soggetti a continui scattering altamente energetici con protoni ed elettroni, impedendo a questi ultimi di formare stati legati. Con la successiva espansione e il conseguente raffreddamento del plasma primordiale, è stata possibile la formazione dei primi atomi di idrogeno e, conseguentemente, i fotoni sono stati liberi di attraversare l'Universo, che da opaco è diventato trasparente alla luce: tali fotoni sono quelli che oggi osserviamo nella CMB.

La CMB costituisce anche una prova della correttezza del Principio Cosmologico: infatti, le osservazioni in regioni diverse del cielo mostrano come essa rimanga invariata, suggerendo l'omogeneità e l'isotropia dell'Universo primordiale. In particolare, le misure effettuate dal satellite COBE (COsmic Background Explorer) [12] mostrano che lo spettro della radiazione fossile è in perfetto accordo con quello di un corpo nero con temperatura $T = 2.7260 \pm 0.0006$ K, descritto dalla legge di Planck:

$$I(\nu) = \frac{2h\nu^3}{c^2} \frac{1}{\exp\left(\frac{h\nu}{k_B T}\right) - 1} . \quad (1.26)$$

Inoltre, le più recenti misure della missione Planck [13], hanno mostrato che le anisotropie della CMB, che si traducono in fluttuazioni di temperatura rispetto al valor medio, sono dell'ordine di $\Delta T/T \approx 10^{-5}$, come mostrato in Figura 1.3. Tali anisotropie, anche se minuscole, sono di fondamentale importanza, perché dalla loro evoluzione, come si vedrà in seguito, si sono formate tutte le strutture presenti nell'Universo attuale.

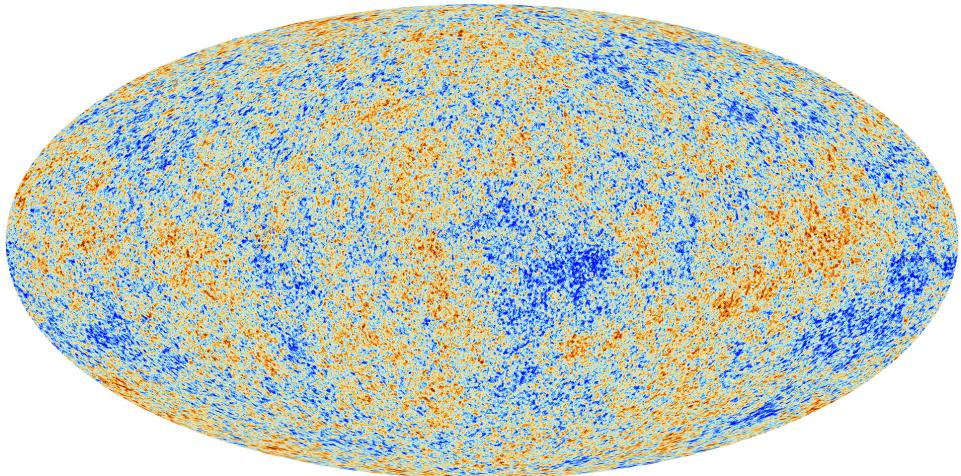


Figura 1.3: Mappa delle anisotropie della CMB realizzata dalla missione Planck. Tali anisotropie sono dell'ordine di 10^{-5} , perciò risulta valida l'assunzione del Principio Cosmologico, per cui l'Universo è omogeneo ed isotropo. Immagine reperibile sul sito ufficiale dell'ESA.

Modello Λ CDM

Il principale modello cosmologico che descrive l'origine dell'Universo è il *modello Λ CDM*, il quale è costituito principalmente da tre ingredienti: la costante cosmologica (Λ), la componente fredda della materia oscura (CDM) e l'inflazione. Questo modello assume curvatura spaziale nulla ($k = 0$) ed è in ottimo accordo con l'attuale espansione accelerata dell'Universo, dovuta al valore positivo di Λ , che a partire da Hubble è stata osservata con strumenti sempre più precisi ed è delineata nelle sue fasi principali in Figura 1.4. È rilevante sottolineare che la presenza dominante di materia oscura fredda, composta da particelle non relativistiche, sia indipendente dall'assunzione di una costante cosmologica positiva.

Secondo il modello Λ CDM, la densità di energia dell'Universo è dovuta per circa il 70% alla costante cosmologica ($\Omega_\Lambda \simeq 0.69$) e per il restante 30% alla

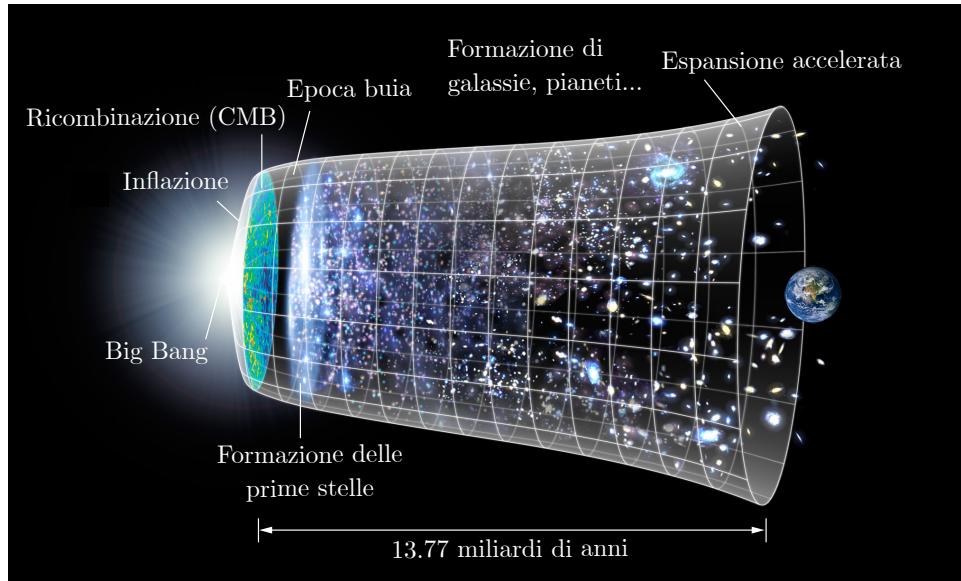


Figura 1.4: Rappresentazione schematica delle principali fasi evolutive dell'Universo secondo il modello Λ CDM. Immagine reperibile sul sito ufficiale della NASA.

materia ($\Omega_m \simeq 0.31$), mentre il contributo della radiazione e della curvatura sono trascurabili ($\Omega_r \simeq 5 \times 10^{-5}$ e $|\Omega_k| < 0.01$), in accordo con l'equazione (1.18). In particolare, il basso limite superiore posto su $|\Omega_k|$ dalle misure più recenti indica che l'Universo, in buona approssimazione, è spazialmente piatto. Inoltre, al fine di poter confrontare sistemi cosmologici su scale spazio-temporali diverse, è conveniente definire i parametri di densità ridotti, definiti mediante la costante di Hubble ridotta h come:

$$\omega_i \equiv h^2 \Omega_i \quad i \in I = \{m, r, \Lambda, k\} . \quad (1.27)$$

In Tabella 1.1 sono riportati i valori dei parametri cosmologici ottenuti dalla collaborazione Planck [5]. Di questi parametri, ω_r è stato accuratamente misurato già dalla missione COBE con il FIRAS (Far-InfraRed Absolute Spectrophotometer). La densità di curvatura Ω_k non compare poiché è deducibile dagli altri parametri attraverso la prima equazione di Friedmann nella forma (1.18). Inoltre, si noti che il parametro ω_ν , il quale indica la densità di neutrini e sarà di particolare rilievo nel presente progetto di tesi, è ottenuto come:

$$\omega_\nu = \frac{\sum_i m_{\nu_i}}{93.14 \text{ eV}} , \quad (1.28)$$

dove la sommatoria corre sui tre sapori del neutrino e può assumere valori compresi tra 0.06 e 1 eV. È importante notare che, nonostante il modello Λ CDM sia ampiamente accettato dalla comunità scientifica, non rappresenta il modello definitivo dell’evoluzione dell’Universo: esso infatti presenta delle discrepanze, come ad esempio le tensioni su misurazioni diverse dei parametri H_0 e σ_8 [14] [15], e dunque è necessario perfezionarlo ulteriormente.

Tabella 1.1: Valori della costante di Hubble e dei principali parametri cosmologici di densità, misurati dalla missione Planck [5] ed assunti dal modello Λ CDM.

Parametro	Simbolo	Valore
Costante di Hubble	H_0	$67.66 \pm 0.42 \text{ km s}^{-1} \text{ Mpc}^{-1}$
Densità di materia barionica	ω_b	0.02242 ± 0.00014
Densità di materia oscura fredda	ω_c	0.11933 ± 0.00091
Densità totale di materia	Ω_m	0.3111 ± 0.0056
Densità totale di radiazione	ω_r	2.47×10^{-5}
Densità di neutrini	ω_ν	< 0.0076
Densità di costante cosmologica	Ω_Λ	0.6889 ± 0.0056

1.3 Formazione delle strutture

In precedenza si è visto che i modelli evolutivi dell’Universo basati sulle equazioni di Friedmann assumono il Principio Cosmologico, descrivendo l’Universo come un fluido perfettamente omogeneo ed isotropo. Tuttavia in un siffatto Universo non si potrebbero formare strutture, come galassie, pianeti e stelle, poiché esse sono delle fluttuazioni di densità, ovvero delle disomogeneità [7]. Occorre dunque perfezionare questi modelli evolutivi tenendo conto di tali fluttuazioni, le quali, come mostrato dalla Figura 1.3, sono presenti già nell’Universo primordiale, e studiando come esse si evolvono nel tempo sotto l’effetto della forza di gravità.

Come si è accennato in precedenza, spesso l’Universo viene modellizzato come un fluido cosmologico, di cui le galassie sono le particelle costituenti: è necessario perciò introdurre alcune nozioni di fluidodinamica per poter comprendere l’evoluzione delle fluttuazioni di densità in tale fluido. Innanzitutto risulta conveniente porsi in un sistema di coordinate che tenga conto dell’espansione dell’Universo, le cosiddette *coordinate comoventi*. Date le coordinate reali di una particella di fluido \mathbf{r} , $\mathbf{u} = \dot{\mathbf{r}}$, le corrispondenti coordinate comoventi sono

$\mathbf{x}, \mathbf{v} = a\dot{\mathbf{x}}$, dove $a = a(t)$ è il fattore di scala della metrica (1.3), inoltre vale che:

$$\begin{cases} \mathbf{r}(t) = a(t) \mathbf{x}(t) \\ \mathbf{u} = H\mathbf{r} + \mathbf{v} \end{cases} , \quad (1.29)$$

dove H è il parametro di Hubble. Definito il sistema di coordinate, la dinamica di un fluido è descritta da tre equazioni differenziali alle derivate parziali: la prima è l'equazione di continuità, che descrive la conservazione della massa,

$$\frac{\partial \rho}{\partial t} + \rho \nabla \cdot \mathbf{v} = 0 , \quad (1.30)$$

la seconda, detta equazione di Eulero, è l'equazione del moto per un elemento di fluido,

$$\frac{\partial \mathbf{v}}{\partial t} = -\frac{1}{\rho} \nabla P - \nabla \phi , \quad (1.31)$$

mentre la terza, cioè l'equazione di Poisson, descrive il potenziale gravitazionale ϕ in presenza di sorgenti,

$$\nabla^2 \phi = 4\pi G \rho . \quad (1.32)$$

Si noti che queste tre equazioni, in particolare la (1.31), sono scritte in coordinate Lagrangiane, e le grandezze che vi compaiono sono le distribuzioni di densità ρ , velocità \mathbf{v} e pressione P . Si vanno ora a studiare le perturbazioni di un fluido omogeneo in espansione uniforme.

Instabilità di Jeans

La teoria di Jeans considera perturbazioni adiabatiche della densità, aventi lo stesso effetto sulla materia barionica e sulla materia oscura, per le quali vale:

$$\frac{\partial P}{\partial \rho} = c_s^2 , \quad (1.33)$$

con c_s velocità del suono nel fluido. Si definisce perciò il *density contrast*:

$$\Delta = \frac{\delta \rho}{\rho_0} , \quad (1.34)$$

dove ρ_0 è la densità media dell’Universo, che è pressoché costante, mentre $\delta\rho$ è una fluttuazione. Combinando le tre equazioni (1.30), (1.31) e (1.32) in spazio di Fourier [7], si ottiene per ogni \mathbf{k} fissato un’equazione per $\Delta(\mathbf{k}, t)$ della forma:

$$\ddot{\Delta} + 2H\dot{\Delta} = (4\pi G\rho_0 - k^2 c_s^2)\Delta , \quad (1.35)$$

dove il punto indica la derivata rispetto al tempo. Considerando un Universo stazionario, con $\dot{a} = 0$, il termine $2H\dot{\Delta}$ è nullo e l’equazione (1.35) ammette come soluzione un’onda piana della forma:

$$\Delta(\mathbf{k}, t) = \Delta_0 e^{i(\mathbf{k}_c \cdot \mathbf{r} - \omega t)} , \quad (1.36)$$

in cui $\mathbf{k}_c = a\mathbf{k}$ è il vettore d’onda in coordinate comoving. Per tale soluzione, la relazione di dispersione è:

$$\omega^2 = c_s^2 k^2 - 4\pi G\rho_0 = c_s^2 (k^2 - k_J^2) , \quad (1.37)$$

dove ρ_0 rappresenta ancora la densità media dell’Universo. Inoltre il parametro k_J , che rappresenta una scala tipica, è detto numero d’onda di Jeans ed è definito come segue:

$$k_J = \sqrt{\frac{4\pi G\rho_0}{c_s^2}} . \quad (1.38)$$

Si nota perciò che perturbazioni con $k > k_J$ oscillano come onde sonore, mentre per $k < k_J$ si ha $\omega^2 < 0$, generando delle soluzioni esponenzialmente crescenti (o decrescenti). Questo accade poiché le fluttuazioni di densità in un fluido tendono a collassare, e quindi ad accrescere diventando sempre più dense, per effetto della gravità, mentre la pressione si oppone a questo collasso, come risulta chiaro dall’equazione di Eulero (1.31). Tuttavia, il tempo di “risposta” della pressione dipende linearmente dalla dimensione spaziale della fluttuazione, perciò si ha un valore limite, che corrisponde alla lunghezza d’onda di Jeans,

$$\lambda_J = \frac{2\pi}{k_J} = \sqrt{\frac{\pi c_s^2}{G\rho_0}} , \quad (1.39)$$

oltre la quale la pressione non riesce a contrastare efficacemente il collasso gravitazionale. Perciò si ha che le perturbazioni su scale spaziali maggiori di λ_J , e dunque con $k < k_J$, tipicamente crescono esponenzialmente.

Crescita delle perturbazioni

Si analizzano ora le soluzioni dell'equazione (1.35) in alcuni semplici casi, considerando ora l'Universo in espansione uniforme: si noti che ciò ristabilisce la presenza del termine $2H\dot{\Delta}$, il quale modifica significativamente l'analisi. Nel caso stabile, con $\lambda < \lambda_J$, le soluzioni sono oscillanti e tale termine ne rappresenta lo smorzamento. Si considerino ora delle perturbazioni per cui è presente instabilità di Jeans, in particolare, con $\lambda \gg \lambda_J$ si può trascurare il termine $k^2 c_s^2$. Durante l'epoca di dominazione della materia, con $a \propto t^{2/3}$, si ottiene la seguente evoluzione delle perturbazioni:

$$\Delta \propto \begin{cases} t^{2/3} \\ t^{-1} \end{cases} . \quad (1.40)$$

È importante notare che, considerando l'espansione dell'Universo, le perturbazioni non crescano più esponenzialmente ma secondo leggi di potenza, ponendo un problema sulla formazione delle galassie: infatti con un simile tasso di crescita esse non avrebbero avuto tempo a sufficienza per formarsi. Nell'epoca di prevalenza della radiazione, il fluido cosmologico deve essere considerato relativistico, provocando una modifica del termine gravitazionale della (1.35), cioè la presenza di un termine $32\pi/3$ al posto di 4π , e le perturbazioni evolvono come:

$$\Delta \propto \log(t) . \quad (1.41)$$

In un Universo dominato dalla costante cosmologica, infine, le soluzioni sono della forma:

$$\Delta \propto \begin{cases} \text{cost.} \\ e^{-2Ht} \end{cases} , \quad (1.42)$$

e dunque le fluttuazioni di densità vengono sopprese esponenzialmente o rimangono costanti, impedendo completamente la formazione di strutture aggregate.

1.4 Spettro di potenza della densità di materia

Nella sezione precedente si è analizzata l’evoluzione del density contrast nel tempo, mentre qui ci si concentra sulla distribuzione spaziale delle fluttuazioni di densità, fondamentale per poter capire se i modelli teorici sono in grado di riprodurre, attraverso simulazioni numeriche, l’attuale struttura su grande scala. Il modo più naturale per studiare tale distribuzione è passare in spazio di Fourier e costruire lo spettro di potenza delle fluttuazioni di densità a partire dalla funzione di correlazione sulla posizione delle galassie. Quest’ultima può essere costruita grazie alla grande quantità di dati provenienti dalle già citate survey di galassie [2].

Funzione di correlazione

La funzione di correlazione a due punti $\xi(r)$ viene utilizzata per ricavare informazioni sulla struttura tridimensionale della distribuzione di galassie nell’Universo [7]. Essa può essere espressa in termini della probabilità di trovare una coppia di galassie separate da una distanza r nel seguente modo:

$$dN = n_0^2 [1 + \xi(r)] dV_1 dV_2 , \quad (1.43)$$

dove n_0 è la densità numerica media di galassie e dN è il numero di coppie di galassie, di cui la prima è contenuta nel volume dV_1 e la seconda in dV_2 . Ricordando la definizione del density contrast (1.34) che permette di scrivere $\rho(\mathbf{x}) = \rho_0 [1 + \Delta(\mathbf{x})]$, si può ricavare la seguente relazione tra ξ e Δ calcolando il valor medio su un vasto campione di elementi di volume:

$$\xi(r) = \langle \Delta(\mathbf{x}) \Delta(\mathbf{x} + \mathbf{r}) \rangle . \quad (1.44)$$

Perciò la conoscenza della funzione di correlazione a due punti dà informazioni sulle fluttuazioni di densità; ad esempio, su scale spaziali comprese tra 100 kpc/ h e 10 Mpc/ h , la correlazione decresce con una legge di potenza della forma:

$$\xi(r) = \left(\frac{r}{r_0} \right)^{-\gamma} , \quad (1.45)$$

dove $r_0 = 5$ Mpc/ h e $\gamma = 1.8$. Questa legge dice di come la correlazione tra le posizioni reciproche delle galassie decresca all’aumentare della scala (si noti

che per scale superiori a $10 \text{ Mpc}/h$, essa decresce ancor più rapidamente), a suggerire ancora una volta l'omogeneità della distribuzione della materia nell'Universo.

Spettro delle perturbazioni

Le fluttuazioni di densità sono scomponibili in modi di Fourier aventi diversi valori di \mathbf{k} : si vuole perciò trovare la relazione che intercorre tra lo spettro di tali fluttuazioni e la funzione di correlazione [7]. Innanzitutto occorre definire la trasformata di Fourier per Δ :

$$\begin{cases} \Delta(\mathbf{r}) = \frac{V}{(2\pi)^3} \int \Delta(\mathbf{k}) e^{-i\mathbf{k}\cdot\mathbf{r}} d^3k \\ \Delta(\mathbf{k}) = \frac{1}{V} \int \Delta(\mathbf{r}) e^{i\mathbf{k}\cdot\mathbf{r}} d^3x \end{cases}, \quad (1.46)$$

dove V è il volume occupato dalla fluttuazione $\Delta(\mathbf{r})$. Tramite il teorema di Parseval è poi possibile scrivere:

$$\frac{1}{V} \int \Delta^2(\mathbf{r}) d^3x = \frac{V}{(2\pi)^3} \int |\Delta(\mathbf{k})|^2 d^3k, \quad (1.47)$$

mettendo quindi in relazione l'ampiezza quadratica media delle fluttuazioni in spazio reale $\Delta^2(\mathbf{r})$ con lo **spettro di potenza** delle fluttuazioni, definito nel seguente modo:

$$P(k) \equiv |\Delta(\mathbf{k})|^2. \quad (1.48)$$

Per trovare la relazione che lega $P(k)$ e $\xi(r)$, è sufficiente espandere in serie di Fourier la quantità $\Delta(\mathbf{x})$ e sostituirla nella (1.44), successivamente si nota che si elidono tutti i termini eccetto quelli per cui $\mathbf{k} = \mathbf{k}'$ ed infine si passa dalla serie ad un integrale di Fourier, ottenendo:

$$\xi(r) = \frac{V}{(2\pi)^3} \int |\Delta(\mathbf{k})|^2 e^{i\mathbf{k}\cdot\mathbf{r}} d^3k. \quad (1.49)$$

Per il calcolo di questo integrale è necessaria solamente la parte reale dell'esponenziale, $\cos(\mathbf{k}\cdot\mathbf{r})$, poiché la funzione di correlazione è reale. Inoltre, sfruttando la simmetria sferica di $\xi(r)$, si può eseguire il calcolo su una distribuzione isotropa di angoli su una sfera, $\frac{1}{2} \sin(\theta) d\theta$. Infine, a partire dalla (1.47), è possibile

riscrivere lo spettro di potenza come integrale della funzione di correlazione, ottenendo:

$$P(k) = \frac{1}{V} \int_0^\infty \xi(r) \frac{\sin(kr)}{kr} 4\pi r^2 dr . \quad (1.50)$$

Da quest'ultima relazione risulta chiaro che, su scala r , le uniche fluttuazioni che contribuiscono sono quelle con numero d'onda di Fourier $k \leq r^{-1}$, poiché i contributi delle altre sono in media nulli a causa del termine $\sin(kr)/kr$.

Spettro primordiale

Le condizioni iniziali del problema delle fluttuazioni di densità su grande scala sono racchiuse nello spettro di potenza primordiale, il quale descrive le perturbazioni scalari della fase inflazionaria dell'Universo [16]. Esso è una funzione della scala k che dipende da tre parametri: A_s , che rappresenta l'ampiezza delle perturbazioni di curvatura ad un scala tipica k_p , e l'indice spettrale n_s :

$$P_{\text{prim}}(k) = \frac{2\pi}{k^3} A_s \left(\frac{k}{k_p} \right)^{n_s - 1} . \quad (1.51)$$

Questa legge di potenza suggerisce che nell'Universo primordiale le fluttuazioni fossero presenti a tutte le scale, formando uno spettro molto ampio. I parametri dello spettro primordiale sono stati misurati da Planck [5], che ha ottenuto $\ln(10^{10} A_s) = 3.047 \pm 0.014$ ad una scala $k_p = 0.05$ Mpc e $n_s = 0.9665 \pm 0.0038$.

Fattore di crescita e neutrini massivi

Per introdurre il fattore di crescita, che sarà uno dei principali oggetti di indagine in questo progetto di tesi, è necessario considerare l'evoluzione dello spettro di potenza delle fluttuazioni. Si può mettere in relazione lo spettro al giorno d'oggi con le fluttuazioni di curvatura primordiali, per grandi scale spaziali, nel seguente modo:

$$P(k, z) = P_{\text{prim}}(k) T^2(k) D^2(z) . \quad (1.52)$$

In questa espressione P_{prim} rappresenta le condizioni iniziali ed è ottenuto dalla (1.51), mentre T e D descrivono l'evoluzione dello spettro. $T(k)$ è detta

funzione di trasferimento e descrive la distribuzione statistica dei modi di Fourier su scale spaziali diverse, mentre $D(z)$ è il **fattore di crescita**, che cattura l’evoluzione temporale (cioè in funzione del redshift) di ciascun modo k . La conoscenza dell’espressione analitica di queste ultime due funzioni implicherebbe la possibilità di predire con esattezza lo spettro, tuttavia ad oggi esistono solo soluzioni analitiche approssimate per la funzione di trasferimento ed il fattore di crescita, e la conoscenza della forma di $P(k, z)$ si deve soprattutto alla grande precisione dei codici sviluppati.

Si vuole ora approfondire il ruolo che i neutrini giocano nella determinazione dello spettro delle fluttuazioni. I neutrini sono particelle neutre con una massa estremamente piccola - ma non nulla - compresa tra 0.06 e 1 eV. Questa loro caratteristica fa sì che essi si muovano a velocità molto elevate, e dunque possano essere considerati come particelle non relativistiche solamente su grandi scale spaziali. In questo caso essi contribuiscono allo spettro di potenza come materia, mentre su scale più piccole contribuiscono come radiazione, andando incontro al cosiddetto *free streaming* e smorzando le fluttuazioni di densità. Tale smorzamento si traduce sullo spettro di potenza in una dipendenza da k del fattore di crescita, che diviene $D(k, z)$. In letteratura esistono alcuni tentativi di trovare la forma esplicita della correzione al fattore di crescita dovuta all’effetto dei neutrini massivi, in particolare si riporta l’espressione ricavata da Kiakotou, Elgarøy e Lahav [17], mostrata anche in Figura 1.5:

$$D(k, z) \approx \bar{D}(z) [1 - A(k)\Omega_\Lambda f_\nu + B(k)f_\nu^2 - C(k)f_\nu^3] , \quad (1.53)$$

in cui $\bar{D}(z)$ è l’approssimazione del fattore di crescita su grandi scale, Ω_Λ è la densità di energia oscura e $f_\nu = \Omega_\nu/\Omega_m$ è la frazione di neutrini.

1.5 Equazioni di Einstein-Boltzmann

La cosmologia studia l’Universo a grande scala, e perciò risulta essere una combinazione di Relatività Generale e Meccanica Statistica [16], governata da due set di equazioni fondamentali: le equazioni di Einstein (1.1), che descrivono la gravità, e quelle di Boltzmann, che governano la fisica statistica [18]. Come si è già accennato, le attuali tecnologie permettono di eseguire misurazioni con una precisione mai vista prima, perciò insorge il problema di costruire modelli cosmologici che predicano le varie osservabili altrettanto precisamente, risolvendo numericamente le equazioni di Einstein-Boltzmann. Per questo sono stati implementati numerosi codici, chiamati “Einstein-Boltzmann solvers” che risolvono le versioni linearizzate di queste equazioni [19]. Tra questi, due di

particolare rilevanza sono **CAMB** [20] e **CLASS** [21]: per questo progetto di tesi si è scelto di utilizzare il secondo per la sua grande flessibilità.

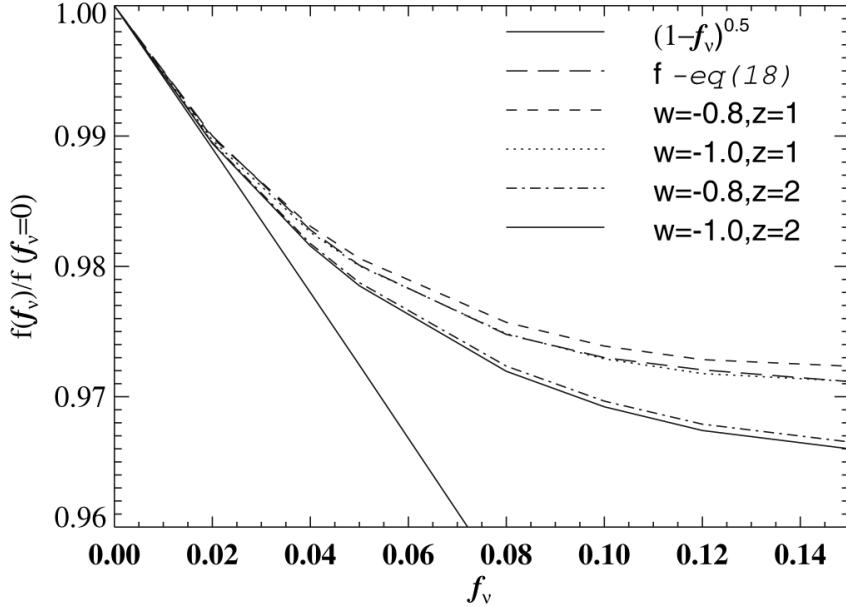


Figura 1.5: Grafico dell'espressione (1.53), riportata in funzione della frazione di neutrini f_ν , ad una scala di 8 Mpc/ h [17]. Il fattore di crescita, indicato con f , è normalizzato rispetto al suo limite a grandi scale e $f -eq(18)$ corrisponde all'espressione (1.53) valutata a $z = 1$. Sono riportati i grafici per diverse combinazioni di redshift z e parametri di stato w (si veda la Sezione 1.1), mentre gli altri parametri cosmologici assumono i seguenti valori: $\Omega_m = 0.3$, $\Omega_b = 0.04$ e $h = 0.7$.

Capitolo 2

Machine Learning

Con **Machine Learning** (ML) si intende quel ramo dell’Intelligenza Artificiale (IA) che contiene algoritmi in grado di svolgere determinate operazioni senza essere stati esplicitamente programmati per farlo. Tali algoritmi sono in grado di apprendere da un insieme di dati, chiamato *training dataset*, eventuali patterns e regole che possono essere utilizzati per compiere predizioni su nuovi dati. In varie discipline scientifiche le tecnologie correnti permettono di raccogliere enormi quantità di dati, con una precisione sempre maggiore, tanto che sempre più spesso si parla della nostra come l’epoca dei *big data*. I metodi di ML hanno generalmente l’obiettivo di estrarre da questi dati correlazioni e regole in modo da poter fare predizioni. Ad esempio, le più recenti survey di galassie stanno fornendo ai cosmologi una mole enorme di dati, dai quali si ottengono misure sempre più precise dei parametri cosmologici. Da ciò nasce l’interesse per lo sviluppo di algoritmi di ML, i quali hanno la capacità di apprendere automaticamente, creando modelli che catturano i patterns e le correlazioni presenti nei dati in modo più efficiente di quanto possa fare un essere umano.

Il Machine Learning si suddivide in due principali categorie a seconda dell’approccio seguito: *apprendimento supervisionato* e *non supervisionato* [22]. Nel caso di apprendimento supervisionato si fornisce all’algoritmo un training set in cui ad ogni input \mathbf{x}_i è associato un output y_i :

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N , \quad (2.1)$$

dove N è il numero di esempi utilizzati per l’allenamento del programma. In generale gli input \mathbf{x}_i possono essere dei vettori di dimensione d , ma anche oggetti strutturati, come ad esempio immagini o frasi. Similmente gli output y_i possono assumere sia valori nominali, nel qual caso si parla di problemi

di classificazione, sia valori numerici nei cosiddetti problemi di regressione. L’obiettivo di questo tipo di apprendimento è quello di costruire un modello che catturi quanto più precisamente possibile la dipendenza dell’output dall’input. Se l’approccio è invece quello non supervisionato, il training dataset assume semplicemente la forma di un insieme di input:

$$\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^N . \quad (2.2)$$

In questo caso l’algoritmo cerca autonomamente patterns e regolarità all’interno di \mathcal{D} , il quale può essere costituito da dati di natura numerica o più complessa.

2.1 Reti Neurali

Uno dei modelli di Machine Learning più utilizzato è quello delle **Reti Neurali Artificiali** (ANN), che stanno assumendo sempre più un ruolo di spicco a causa della grande varietà di ambiti a cui possono essere applicate, dal riconoscimento di immagini [23] alla predizione della struttura tridimensionale delle proteine [24]. Al livello più essenziale, le ANN sono funzioni non lineari di più variabili che dipendono da molti parametri, tuttavia possono anche essere viste come degli approssimatori di funzioni che vengono allenati con vari esempi [25].

Le ANN hanno strutture complesse, la cui architettura si ispira al modello biologico del cervello umano, composto da miliardi unità di elaborazione dei segnali interconnesse, i neuroni. La struttura di una ANN, rappresentata in Figura 2.1, prevede delle unità di calcolo fondamentali, dette *neuroni*, organizzate in *strati* (o layers) e connesse tra loro. Nel caso delle cosiddette ANN dense, ciascun neurone riceve input da tutti quelli del layer precedente. Il primo strato prende il nome di *strato di input*, mentre l’ultimo è detto *strato di output*; tra di essi sono presenti uno o più *strati nascosti* ad aumentare la complessità (e dunque la potenza espressiva) della rete.

Ciascun neurone di una ANN prende in input una combinazione *lineare* degli output provenienti dallo strato precedente, alla quale viene applicata una funzione *non lineare*, detta *funzione di attivazione*, il cui risultato (scalare) costituisce l’output del neurone. Inoltre, ogni connessione è caratterizzata da un peso w , perciò il valore di input dell’ i -esimo neurone nell’ l -esimo strato non è altro che la somma degli output del layer $l - 1$, pesata dai corrispettivi w :

$$x_i^l = \sum_{j=1}^{n_l} w_{ij}^l y_j^{l-1} + b_i^l , \quad (2.3)$$

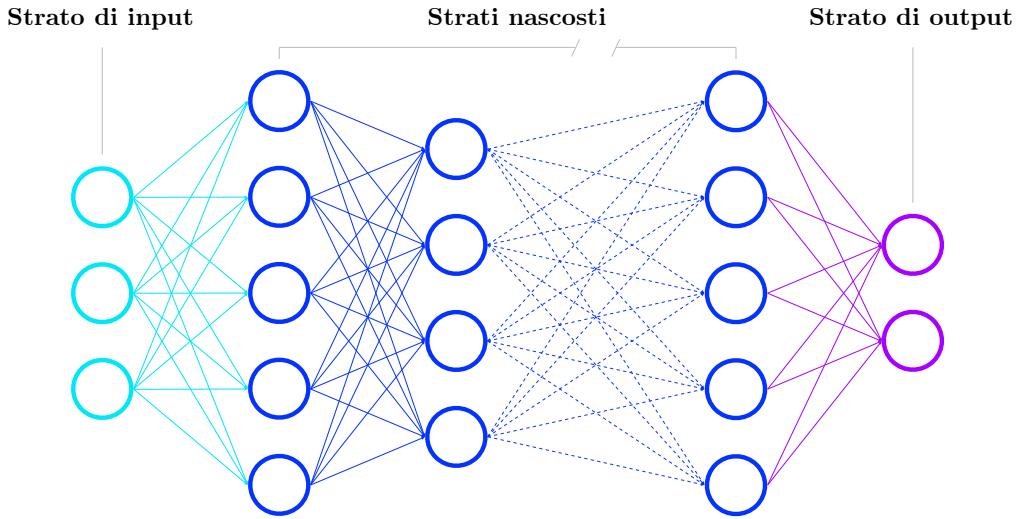


Figura 2.1: Schema della struttura di una rete neurale densa dotata di vari strati nascosti.

dove b_i^l è detto *bias* del neurone i , y_j^{l-1} è l'output del j -esimo neurone del layer precedente e n_l è il numero di neuroni del layer l . L'equazione (2.3) può essere riscritta utilizzando il formalismo matriciale come:

$$\mathbf{x} = W\mathbf{y} + \mathbf{b} . \quad (2.4)$$

Ogni neurone processa l'input ricevuto tramite una *funzione di attivazione* f , che costituisce la non linearità del sistema: questa caratteristica è fondamentale in quanto permette alle ANN di approssimare un'enorme varietà di funzioni, che sarebbe notevolmente ridotta se l'output dei neuroni fosse una combinazione lineare degli input. Le funzioni di attivazione più comuni sono la *sigmoide*,

$$s(x) = \frac{1}{1 + e^{-x}} , \quad (2.5)$$

che deve il suo nome alla sua forma di “S” allungata ed è sostanzialmente una funzione a gradino resa liscia, la *ReLU* (Rectified Linear Unit),

$$r(x) = \max(0, x) , \quad (2.6)$$

e la *tangente iperbolica*,

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} . \quad (2.7)$$

Soltamente si costruiscono reti neurali utilizzando la medesima funzione di attivazione per tutti i neuroni, tuttavia non esiste una scelta ottimale a priori. Ad esempio la derivata della ReLU è monotona e molto semplice da calcolare, e questo fa sì che una rete che utilizzi la (2.6) come funzione di attivazione sia meno soggetta a loop durante il training dovuti a minimi locali del gradiente (come risulterà chiaro in seguito). Questa funzione, tuttavia, mappa tutti gli input negativi in zero, e questo può portare a problemi detti di “dying ReLU”, che si verificano quando un elevato numero di neuroni non si attiva mai e quindi non apprende nulla. In generale dunque la scelta della funzione di attivazione dipende dal tipo di problema che si sta affrontando e spesso è frutto di tentativi ed errori.

Come accennato in precedenza, le ANN possono essere pensate come approssimatori di funzioni arbitrariamente complesse. A tal proposito si vuole riportare un importante teorema, la cui dimostrazione si deve a George Cybenko [26], il quale afferma che una rete neurale con un singolo strato nascosto ed una generica funzione di attivazione non lineare può approssimare con precisione arbitraria una qualunque funzione continua, a condizione di avere abbastanza neuroni. Questo teorema è noto come *Teorema di approssimazione universale* e costituisce un fondamento teorico per l'utilizzo delle reti neurali in una vasta gamma di applicazioni.

Allenamento di una rete neurale

Una volta definite la struttura e la funzione di attivazione di una rete neurale, occorre calibrare i restanti parametri, ovvero i pesi w_{ij} , caratteristici di ogni connessione della rete, e i bias b_i di ciascun neurone. Questa fase è detta *allenamento* della rete e gioca un ruolo fondamentale per l'efficienza di quest'ultima. In questa sezione ci si concentrerà principalmente sull'allenamento nel caso di apprendimento supervisionato, in cui, avendo associato un output ad ogni input del training dataset, si può quantificare l'errore commesso dal modello.

Se si indica con θ una generica configurazione di tutti i parametri (pesi e bias) della rete, si può esprimere il suo output complessivo come una funzione $\mathbf{F}_x(\theta)$, che dipende parametricamente dall'input x ricevuto. Si può quindi definire una funzione costo specifica, ad esempio:

$$C_x(\theta) = \|\mathbf{F}_x(\theta) - \mathbf{y}_x^{\text{out}}\|^2 , \quad (2.8)$$

che è sostanzialmente la distanza tra il valore predetto dalla rete dato l'input \mathbf{x} e il corrispettivo valore reale $\mathbf{y}_x^{\text{out}}$ assegnato nel training set. Se si calcola la media della (2.8) su tutto il dataset di allenamento si ottiene la funzione costo complessiva:

$$C(\theta) = \langle \|\mathbf{F}_{\mathbf{x}}(\theta) - \mathbf{y}_{\mathbf{x}}^{\text{out}}\|^2 \rangle_{\mathbf{x}} . \quad (2.9)$$

La funzione costo, detta anche *loss function*, definisce il problema di ottimizzazione che verrà risolto in fase di allenamento della rete: essa non è determinata a priori, ma varia a seconda del compito per cui la ANN è stata progettata.

Uno degli algoritmi di ottimizzazione più intuitivi e semplici da implementare, valido se la funzione costo è differenziabile, è la **Gradient Descent** (GD). In questo algoritmo iterativo il valore dei parametri viene aggiornato sottraendo di volta in volta il gradiente della (2.9) moltiplicato per uno scalare η detto tasso di apprendimento (*learning rate*):

$$\theta_{i+1} = \theta_i - \eta \nabla C(\theta_i) , \quad (2.10)$$

dove θ_i è la configurazione dei parametri al passo i -esimo. In tal modo si segue la direzione della massima “pendenza” della funzione costo fino ad arrivare, al termine del training, alla configurazione ottimale dei parametri, che corrisponde al minimo globale di $C(\theta)$. In questo processo il valore di η è di fondamentale importanza, poiché quantifica l'ampiezza di ciascun passo. Come mostrato in Figura 2.2, se il learning rate è troppo piccolo l'algoritmo potrebbe impiegare troppe iterazioni per convergere, oppure potrebbe bloccarsi in un minimo locale della funzione costo; d'altro canto, un tasso di apprendimento troppo grande non permetterebbe di individuare il minimo globale con precisione sufficiente o, nel peggiore dei casi, potrebbe causare la divergenza dell'algoritmo.

Solitamente i dataset utilizzati per allenare le reti neurali sono molto estesi, perciò il calcolo della funzione costo (2.9), dove la media è eseguita su tutto il training set, risulta spesso troppo costoso dal punto di vista computazionale per essere eseguito ad ogni iterazione della GD. Per ovviare a questo problema la media viene calcolata rispetto a sottoinsiemi casuali del dataset di allenamento, chiamati *mini-batches*, e la funzione costo assume la seguente forma approssimata:

$$C(\theta) \approx \frac{1}{N} \sum_{k=1}^N \|\mathbf{F}_{\mathbf{x}_k}(\theta) - \mathbf{y}_{\mathbf{x}_k}^{\text{out}}\|^2 = \langle C_{\mathbf{x}}(\theta) \rangle|_{\text{batch}} , \quad (2.11)$$

dove N è il numero di dati nel batch. Questo metodo, chiamato **Stochastic Gradient Descent** (SGD), permette di ridurre notevolmente i tempi di calcolo, fornendo però un risultato meno preciso, in quanto ad ogni step $C(\theta)$ viene stimata e non calcolata esattamente.

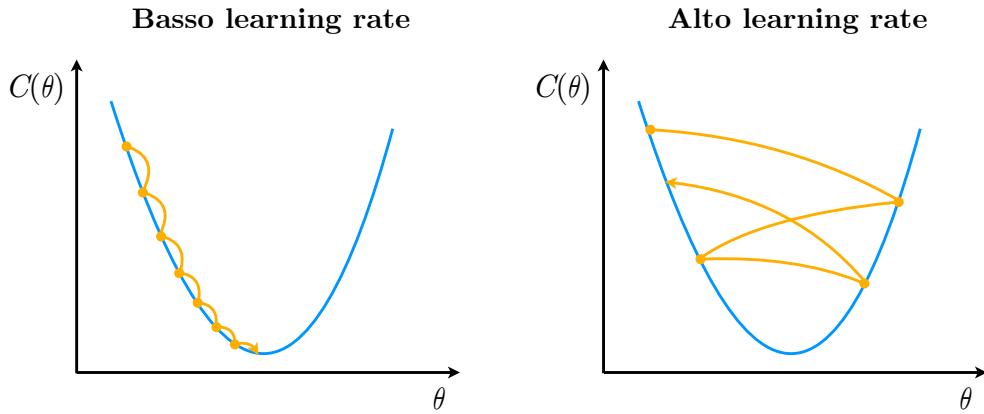


Figura 2.2: Rappresentazione grafica dell'effetto che il learning rate comporta sull'andamento della GD nel caso monodimensionale. A sinistra: un valore di η piccolo comporta la necessità di un maggior numero di iterazioni per raggiungere il minimo. A destra: un elevato tasso di apprendimento può causare una scarsa precisione nell'individuazione del valore ottimale di θ .

Backpropagation

Se con l'introduzione della SGD si è risolto il problema di dover calcolare la media della loss function su tutto il training dataset, rimane comunque da affrontare la difficoltà nel calcolo del singolo gradiente. Infatti, la funzione costo dipende spesso da molti parametri θ , che nei casi più semplici possono essere dell'ordine delle centinaia, ma arrivano anche ai miliardi nelle reti più complesse (come ad esempio ChatGPT). È impensabile calcolare tale gradiente per ogni step utilizzando algoritmi quali, ad esempio, quello delle differenze finite, perciò si ricorre ad un metodo più efficiente, chiamato **backpropagation**, il quale calibra i parametri della rete partendo dallo strato di output e procedendo verso quello di input.

Prima di mostrare il funzionamento di questo algoritmo, è necessario fare chiarezza sulla notazione utilizzata [27]. Si indicherà con θ_{ij}^l il parametro associato alla connessione tra l' i -esimo neurone al layer $l - 1$ e il j -esimo

neurone dello strato l e si noti subito che il bias b_j^l del neurone j corrisponderà al termine θ_{0j}^l . Inoltre si ricordi che l'input x_i^l dell' i -esimo neurone del layer l è dato dalla (2.3) e che l'output y_i^l del medesimo neurone è $f(x_i^l)$, dove f è la funzione di attivazione scelta per la rete. Infine, per rispettare la definizione dell'input di un neurone e la scelta di notazione relativa ai bias, si definisce $y_0^l = 1$, avendo esteso la sommatoria nell'equazione (2.3) da 0 a n_l .

L'obiettivo della backpropagation è quello di avvicinarsi in maniera efficiente al minimo della funzione costo per la SGD, calcolando il suo gradiente rispetto ai parametri θ_{ij}^l :

$$\frac{\partial C(\theta)}{\partial \theta_{ij}^l} \approx \frac{\partial}{\partial \theta_{ij}^l} \left(\langle C_{\mathbf{x}}(\theta) \rangle \Big|_{\text{batch}} \right) = \frac{1}{N} \sum_{k=1}^N \frac{\partial C_k}{\partial \theta_{ij}^l}, \quad (2.12)$$

dove si ricorda che N è il numero di elementi in un batch, mentre con θ si indica una determinata configurazione dei parametri. Inoltre si può notare che la (2.12) è valida per una loss function generica, a patto di calcolarne la media su un batch di dimensione finita, perciò il seguente discorso è generalizzabile ad altre scelte di $C(\theta)$. Matematicamente, la backpropagation si basa sull'applicazione della *chain rule* per le funzioni differenziabili, perciò:

$$\frac{\partial C_k}{\partial \theta_{ij}^l} = \frac{\partial C_k}{\partial x_j^l} \frac{\partial x_j^l}{\partial \theta_{ij}^l}, \quad (2.13)$$

dove il primo termine viene detto errore, e lo si indica con $\delta_j^{k,l}$, mentre il secondo è calcolato come segue:

$$\frac{\partial x_j^l}{\partial \theta_{ij}^l} = \frac{\partial}{\partial \theta_{ij}^l} \left(\sum_{m=0}^{n_l} \theta_{im}^l y_m^{l-1} \right) = y_i^{l-1}. \quad (2.14)$$

Dunque l'equazione (2.13) assume la forma:

$$\frac{\partial C_k}{\partial \theta_{ij}^l} = \delta_j^{k,l} y_i^{l-1}. \quad (2.15)$$

Applicando ora la chain rule per le funzioni in più variabili a $\delta_j^{k,l}$, si ottiene:

$$\delta_j^{k,l} \equiv \frac{\partial C_k}{\partial x_j^l} = \sum_{m=0}^{n_{l+1}} \frac{\partial C_k}{\partial x_m^{l+1}} \frac{\partial x_m^{l+1}}{\partial x_j^l}. \quad (2.16)$$

Ricordando la definizione di x_m^{l+1} e il fatto che $y_r^l = f(x_r^l)$, si ha:

$$\frac{\partial x_m^{l+1}}{\partial x_j^l} = \frac{\partial}{\partial x_j^l} \left(\sum_{r=0}^{n_l+1} \theta_{mr}^{l+1} f(x_r^l) \right) = \theta_{mj}^{l+1} f'(x_j^l), \quad (2.17)$$

che, sostituita nell'equazione (2.16) assieme alla definizione di $\delta_m^{k,l+1}$ restituisce l'espressione:

$$\delta_j^{k,l} = f'(x_j^l) \sum_{m=0}^{n_l+1} \theta_{mj}^{l+1} \delta_m^{k,l+1}. \quad (2.18)$$

Quest'ultima è una formula ricorsiva che, se iterata fino al layer di output ($l = o$), permette infine di calcolare tramite la (2.15) il gradiente della loss function rispetto a tutti i parametri della rete:

$$\frac{\partial C_k}{\partial \theta_{ij}^l} = y_i^{l-1} f'(x_j^l) \sum_{m=0}^{n_l+1} \theta_{mj}^{l+1} \delta_m^{k,l+1}. \quad (2.19)$$

La backpropagation è un algoritmo di fondamentale importanza per il training delle ANN poiché ha un costo computazionale paragonabile a quello del “forward pass”, ovvero del calcolo dell'output della rete.

2.2 Regressione Simbolica

Le ANN sono strumenti potenti e versatili, che permettono di cogliere svariati patterns e rapporti di causalità nascosti nei dati, tuttavia sono dei modelli “black-box”: esse associano input e output utilizzando forme funzionali di difficile interpretazione fisica. Questo apre la strada ad un'altra applicazione del Machine Learning, chiamata **Regressione Simbolica**: una tecnica di apprendimento supervisionato che è in grado, a partire dai dati, di ricavare un'equazione esplicita dei parametri scelti per descrivere il modello [28].

I vantaggi di questo approccio sono principalmente due. Il primo è la maggior facilità a condividere i risultati ottenuti: infatti avendo come modello un'espressione analitica è sufficiente implementarla in un nuovo codice, senza bisogno di importare l'intera struttura dei parametri di una rete neurale. Inoltre la Regressione Simbolica produce modelli esplicativi che potrebbero mettere in luce rapporti di causalità mai notati prima, suggerendo nuove leggi agli scienziati che le interpretano. Un problema di regressione tradizionale, che si applica a

sistemi di cui si possiede una solida conoscenza fisica, consiste nell'ottimizzare i parametri di un'equazione proveniente dai modelli teorici per meglio adattarla ai dati sperimentali. Per esempio, nota la terza legge di Keplero $T^2 = \frac{4\pi^2}{GM}a^3$, una volta misurati il periodo e il semiasse maggiore dell'orbita, si può usare la regressione per ricavare il termine di proporzionalità, da cui si può estrarre la massa della stella. Nel caso della Regressione Simbolica, invece, la legge fisica non è nota a priori e l'obiettivo è proprio quello di determinarla, esplorando lo spazio delle espressioni matematiche che meglio rappresentano i dati [29]. Perciò questa tecnica è utilizzata in svariati ambiti per avere indizi sulla fisica del sistema oppure per catturare leggi non lineari piuttosto complesse [30].

Uno dei principali rischi della Regressione Simbolica è quello di *overfitting*, che si verifica quando il modello impara a riprodurre tanto fedelmente il training set da catturarne le fluttuazioni statistiche nell'espressione simbolica, risultando di conseguenza poco preciso nel momento in cui lo si estende ad un validation dataset. In generale la tecnica migliore è combinare l'utilizzo della Regressione Simbolica, che fornisce un risultato facilmente interpretabile, con la precisione di calcolo data dalle ANN [31].

Algoritmi genetici

La Regressione Simbolica deve il suo successo alla capacità di esplorare un vasto numero di espressioni analitiche, combinandole tra loro in maniera efficiente. Questo tipo di funzionamento è possibile grazie agli **algoritmi genetici**, che sono dei metodi per evolvere delle variabili (in questo caso equazioni) ispirati all'evoluzione biologica [32]. Quest'ultima, attraverso incroci e mutazioni casuali del codice genetico, ha infatti prodotto organismi estremamente complessi a partire da esseri viventi monocellulari, e rappresenta dunque un ottimo modello a cui ispirarsi.

Il primo passo degli algoritmi genetici è la generazione casuale di alcuni insiemi di equazioni, detti *popolazioni*, i cui *individui* sono le candidate soluzioni al problema (si veda la Figura 2.3). Essendo generati casualmente, alcuni individui riproducono il training dataset meglio di altri, e dunque vengono selezionati e ricombinati fra loro (come mostrato in Figura 2.4) per formare nuove popolazioni, mentre gli altri sono scartati. Inoltre, nel passaggio alla popolazione successiva, c'è una probabilità di mutazione del singolo individuo e di crossover tra individui diversi al fine di aumentare ulteriormente lo spazio delle funzioni esplorate (Figura 2.5). In ultimo, il processo viene *iterato* fino al raggiungimento di una soluzione soddisfacente, che riproduca fedelmente il dataset e non sia eccessivamente complessa.

Due sono le caratteristiche degli algoritmi genetici che li rendono adatti a cercare soluzioni di problemi complessi: l'adattabilità a situazioni non stazionarie e la possibilità di eseguire facilmente ricerche parallelizzate [33]. Essi infatti ottengono alte prestazioni anche in condizioni variabili, poiché il loro meccanismo evolutivo è molto semplice - essenzialmente si tratta di mutazioni casuali - ed è combinato con una “selezione naturale”, che dunque introduce nella casualità un bias verso il modello che meglio si adatta ai dati sperimentali. Inoltre l’evoluzione delle varie popolazioni può facilmente essere implementata in parallelo, permettendo un’efficienza di gran lunga maggiore rispetto allo studio di una “specie” alla volta.

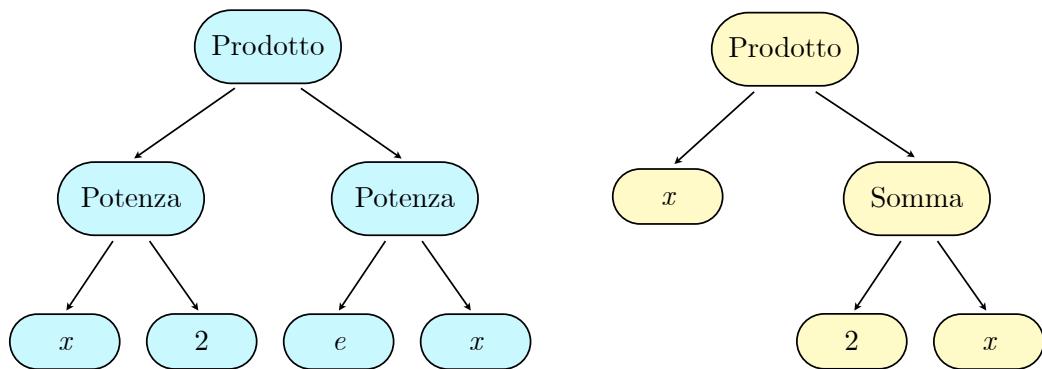


Figura 2.3: Esempio di rappresentazione ad albero di due individui rappresentanti le espressioni “ $x^2 \cdot e^x$ ” e “ $x \cdot (2 + x)$ ”.

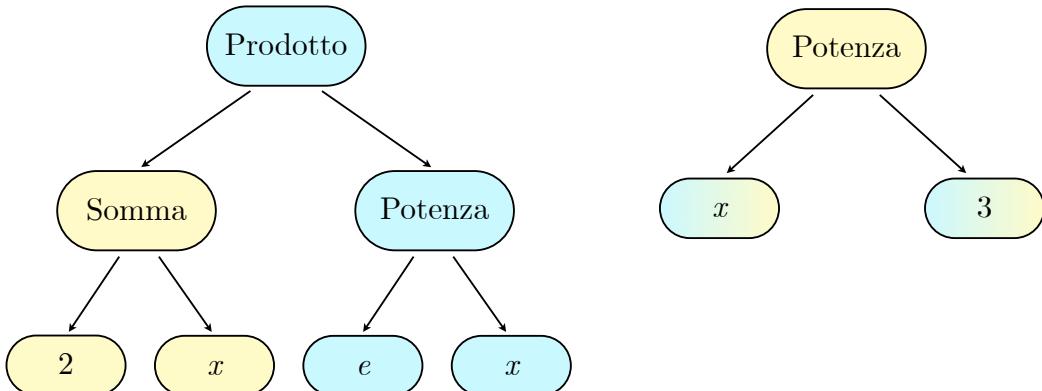


Figura 2.4: I due individui sono stati combinati tra loro, scambiandosi i blocchi “ x^2 ” e “ $2 + x$ ”. Il blocco “ x^2 ”, mantenendo l’operazione “Prodotto” proveniente dal secondo individuo, ha generato l’elemento “ x^3 ”

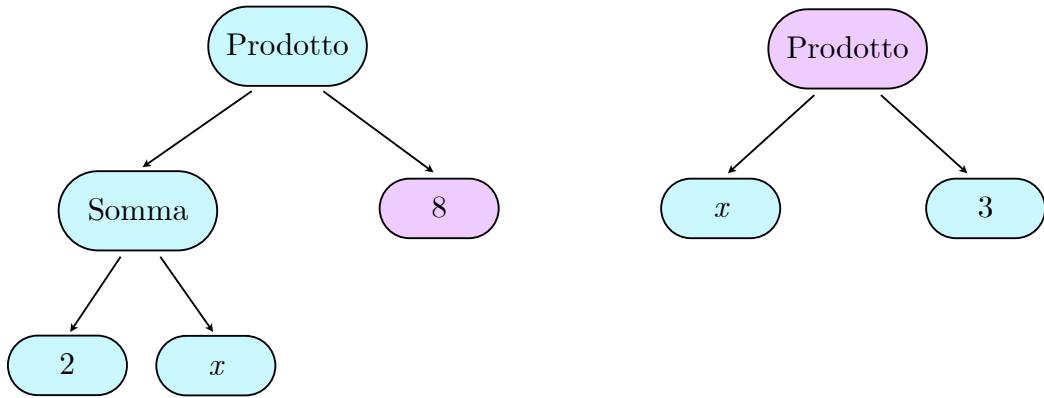


Figura 2.5: I due individui combinati hanno subito una mutazione: il blocco “ e^x ” del primo individuo è diventato “8”, mentre l’operazione del secondo individuo da “Potenza” è diventata “Prodotto”.

PySR

Nel presente progetto di tesi la Regressione Simbolica è stata implementata utilizzando PySR [🔗](#), una libreria open-source scritta in Python che permette di svolgere la ricerca di espressioni analitiche in maniera ottimizzata [34]. PySR si basa su un backend scritto in Julia, che consiste in una libreria chiamata `SymbolicRegression.jl` [🔗](#), utilizzabile anche in maniera autonoma.

PySR implementa una variante del classico algoritmo genetico, basato su un ciclo evoluzione-semplificazione-ottimizzazione in cui le varie popolazioni vengono processate parallelamente. La fase di evoluzione è analoga a quella descritta nella sezione precedente, in cui gli individui di ogni popolazione vengono selezionati attraverso un “torneo”, successivamente le espressioni simboliche trovate vengono semplificate algebricamente ed infine, nella fase di ottimizzazione, vengono utilizzati algoritmi di minimizzazione (come ad esempio il metodo BFGS [35]) per determinare i valori ottimali delle costanti. Rispetto ad un algoritmo genetico standard, PySR presenta varie differenze e possibilità di personalizzazione, di cui le fasi di semplificazione e ottimizzazione sopracitate sono un esempio, in quanto sono applicabili soltanto alla classe di individui considerati da questo algoritmo (le equazioni). Un altro esempio di variante implementata da PySR è la “age-regularization” presente nella fase evolutiva, che consiste nello scartare automaticamente gli individui più vecchi, per fare in modo che l’algoritmo non converga troppo in fretta, rischiando di individuare un minimo locale dello spazio di ricerca.

Si illustrano ora alcune caratteristiche di questa libreria, in particolare del modulo `PySRegressor`, utilizzato per costruire i modelli, tra cui definizioni utili e parametri modificabili dall'utente. Per informazioni più dettagliate si consulti la documentazione ufficiale [36].

- Complessità: è il numero di nodi nel diagramma ad albero di un'espressione analitica. Questo parametro è utile nella selezione del modello finale dell'algoritmo, che di default seleziona la soluzione che combina il maggior potere predittivo con la minor complessità. Il suo valore di default è dato dal parametro `maxsize` e vale 20.
- Popolazioni: sono gli insiemi di individui che l'algoritmo evolve indipendentemente, di default sono `populations = 15` e ciascuna è formata da 33 elementi (regolabile attraverso il parametro `population_size`).
- Iterazioni: sono i cicli evoluzione-semplificazione-ottimizzazione a cui l'algoritmo sottopone ciascuna popolazione. Il valore di default è 40, ed è modificabile tramite il parametro `niterations`.
- Funzione costo: è il quantificatore della bontà predittiva del modello, detta anche loss function. La loss function predefinita è `loss = 'L2DistLoss()`, che corrisponde all'errore quadratico medio, ma può essere scelta tra una serie di possibilità oppure definita a piacere.
- Operatori: sono gli elementi che PySR può combinare per ottenere varie espressioni. Oltre agli operatori binari, che di default sono le comuni operazioni aritmetiche, è possibile inserire anche degli operatori unari (ad esempio l'esponenziale) oppure definirne a piacere.
- Batching: è la divisione delle popolazioni in sottoinsiemi di dimensione inferiore, sui quali viene eseguita la selezione, in modo che l'algoritmo converga in un tempo minore. La divisione è effettuata se il parametro `batching` è impostato su `True`, e la dimensione dei sottoinsiemi è data dal parametro `batch_size`.

Al fine di comprendere le potenzialità e i limiti di questa libreria, si è deciso, prima di iniziare il lavoro vero e proprio, di svolgere la ricerca di una semplice espressione analitica, una funzione $f : \mathbb{R} \rightarrow \mathbb{R}$ definita come:

$$y = x^2 + 10 \cos(x) - 2 . \quad (2.20)$$

Per generare il training dataset, si sono estratti 100 valori di x da una distribuzione gaussiana di media 0 e deviazione standard 5, mentre le y sono state

ottenute a partire dalla (2.20), aggiungendo un rumore gaussiano di ampiezza $\sigma = 3$. Per questa ricerca di test si è scelto di utilizzare come parametri del modello i valori forniti di default da PySR, in particolare 40 iterazioni, 15 popolazioni ed una complessità pari a 20, inoltre come operatori sono state scelte le operazioni aritmetiche standard e la funzione coseno. In Figura 2.6 si mostra il risultato ottenuto, e si può notare come, nonostante il noise, la regressione sia in grado di trovare un'espressione analitica quasi perfetta:

$$y_{\text{pred}} = x^2 + 9.99 \cos(x) - 2.15 . \quad (2.21)$$

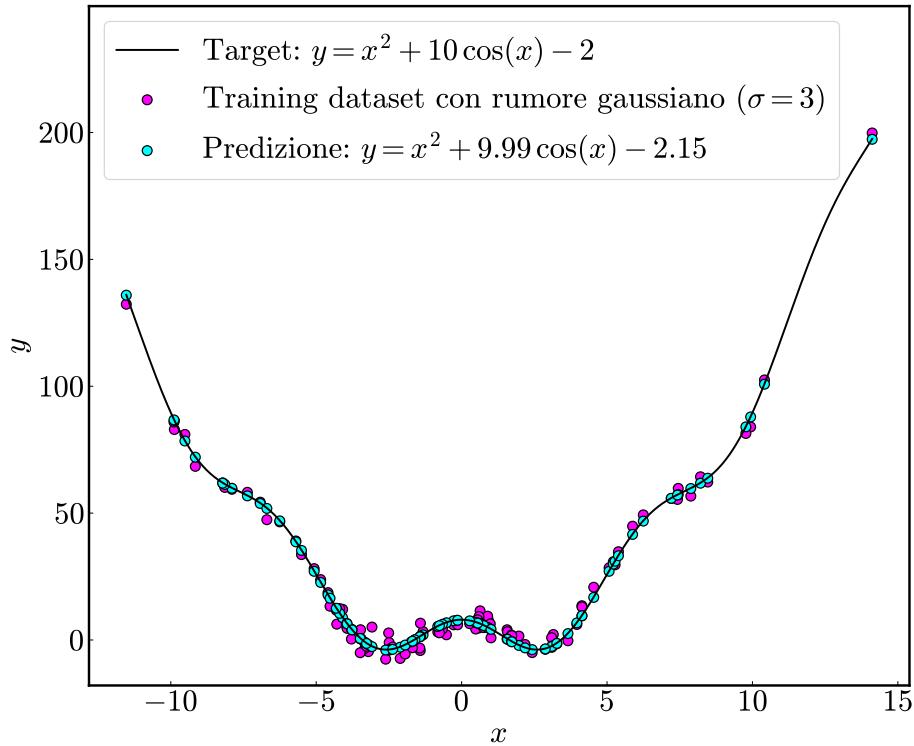


Figura 2.6: Risultati del test di ricerca dell'espressione $y = x^2 + 10 \cos(x) - 2$ tramite l'utilizzo di PySR. I valori in output nel training dataset presentano un rumore gaussiano con deviazione standard $\sigma = 3$. L'espressione trovata è $y_{\text{pred}} = x^2 + 9.99 \cos(x) - 2.15$.

Capitolo 3

Costruzione dell'emulatore per lo spettro di potenza

L'obiettivo del mio progetto di tesi è costruire un **emulatore neuro-simbolico** dello spettro di potenza delle fluttuazioni di densità $P(k, z)$ combinando due tecniche di Machine Learning: la Regressione Simbolica e le Reti Neurali. La prima parte del lavoro è stata la ricerca di un'espressione simbolica per il fattore correttivo $\mu(k, z)$ che la massa dei neutrini provoca sul fattore di crescita, valutando diversi set di parametri cosmologici per descriverlo. Nella seconda parte, invece, ho allenato una rete neurale per ricostruire l'intero spettro di potenza, utilizzando l'espressione simbolica ottenuta in precedenza nel preprocessing del training dataset. In questa fase mi sono servito anche dell'espressione simbolica ottenuta per la funzione di trasferimento $T(k)$ da Bartlett et al. [37]. Infine ho confrontato la precisione del mio emulatore con quella di un modello più semplice, costituito solamente da una rete neurale, al fine di verificare se la procedura di preprocessing effettuata mediante la Regressione Simbolica permetta di ottenere una performance migliore.

3.1 Regressione Simbolica sul fattore di crescita

In questa Sezione illustro la prima parte del lavoro svolto, ovvero l'applicazione della regressione simbolica al fattore correttivo sul fattore di crescita dovuto alla presenza di neutrini massivi. Ho costruito il training dataset per la regressione utilizzando il l'interfaccia Python di **CLASS** (Cosmic Linear Anisotropy Solving System)  [21], da cui ho estratto lo spettro di potenza ottenuto risolvendo numericamente le equazioni di Einstein-Boltzmann linearizzate. Ho poi utilizzato

la libreria PySR  [34] per ottenere i modelli di regressione simbolica.

Generazione dei dataset

Per ottenere il training dataset su cui allenare i modelli di regressione simbolica, ho utilizzato il Python wrapper di **CLASS**, il quale non è in grado di calcolare direttamente il fattore di crescita in funzione sia del redshift che della scala, tuttavia fornisce sia lo spettro di potenza completo che l’approssimazione $\bar{D}(z)$ su grandi scale. Perciò per la creazione del training dataset ho sfruttato la fattorizzazione di $P(k, z)$ già discussa alla fine della Sezione 1.4:

$$P(k, z) = P_{\text{prim}}(k) T^2(k) D^2(k, z) . \quad (3.1)$$

Successivamente, ho riscritto il fattore di crescita $D(k, z)$ come:

$$D(k, z) = \bar{D}(z) \bar{\mu}(k, z) , \quad (3.2)$$

dove $\bar{D}(z)$ rappresenta l’approssimazione di D su grandi scale e $\bar{\mu}(k, z)$ è il fattore correttivo dovuto alla massa di questi ultimi. Perciò, dividendo per lo spettro di potenza al giorno d’oggi ($z = 0$), ho ottenuto:

$$\mathcal{D}(k, z) = \sqrt{\frac{P(k, z)}{P(k, 0)}} = \frac{D(k, z)}{D(k, 0)} = \frac{\bar{D}(z) \bar{\mu}(k, z)}{\bar{D}(0) \bar{\mu}(k, 0)} . \quad (3.3)$$

Infine, normalizzando per il fattore $\bar{D}(z)/\bar{D}(0)$, ho ricavato l’espressione:

$$\mu(k, z) = \frac{\bar{\mu}(k, z)}{\bar{\mu}(k, 0)} , \quad (3.4)$$

che è il termine correttivo al fattore di crescita cercato attraverso la regressione simbolica. Il training dataset risulta così costituito da un array bidimensionale le cui colonne contengono valori di k [h/Mpc], z e vari parametri cosmologici; inoltre l’ultima colonna contiene il valore di $\mu(k, z)$ corrispondente a ciascuna combinazione degli altri parametri.

I valori di k presenti nel dataset sono $n_k = 100$, distribuiti nel range $[10^{-4}, 3]$ ad intervalli di ampiezza logaritmica, mentre quelli di z sono $n_z = 20$, estratti ad intervalli regolari nel range $[0, 5]$. Per quanto riguarda invece i parametri cosmologici, ho costruito vari dataset (e di conseguenza vari modelli) per testare la dipendenza di μ da essi. In Tabella 3.1 riporto le combinazioni di questi

Tabella 3.1: Combinazioni dei parametri cosmologici utilizzate nei vari modelli costruiti con rispettivi range di variazione e numeri di valori assunti da ciascun parametro. I modelli $mbch$, $VBCh$ e $VBMh$ sono risultati essere i migliori in termine di precisione, in quanto utilizzano il maggior numero di parametri. Per questi tre modelli i valori dei parametri sono stati scelti utilizzando due Latin Hypercube a 4 entrate ciascuno, contenenti 200 combinazioni.

Modello	Parametri	Range	Estrazioni
m	m_ν	$[0.06, 1]$ eV	10
man	m_ν	$[0.06, 1]$ eV	10
	$\ln(10^{10} A_s)$	$[2.5, 3.5]$	20
	n_s	$[0.9, 1]$	20
mb	m_ν	$[0.06, 1]$ eV	10
	ω_b	$[0.020, 0.024]$	20
mc	m_ν	$[0.06, 1]$ eV	10
	ω_{cdm}	$[0.10, 0.14]$	20
mh	m_ν	$[0.06, 1]$ eV	10
	h	$[0.6, 0.8]$	20
$mbch$	m_ν	$[0.06, 1]$ eV	200
	ω_b	$[0.020, 0.024]$	200
	ω_{cdm}	$[0.10, 0.14]$	200
	h	$[0.6, 0.8]$	200
$VBCh$	Ω_ν	$[0.001, 0.030]$	200
	Ω_b	$[0.04, 0.06]$	200
	Ω_{cdm}	$[0.23, 0.29]$	200
	h	$[0.6, 0.8]$	200
$VBMh$	Ω_ν	$[0.001, 0.030]$	200
	Ω_b	$[0.04, 0.06]$	200
	Ω_m	$[0.27, 0.40]$	200
	h	$[0.6, 0.8]$	200

parametri utilizzate nei vari dataset, il range in cui variano e il numero di valori assunti da ciascun parametro. Per i primi modelli ho selezionato i parametri su griglie equispaziate come per k e z , mentre per i modelli $mbch$, $VBCh$ e $VBMh$ ho utilizzato dei Latin Hypercubes per coprire lo spazio dei parametri utilizzando un numero ridotto di combinazioni. Ad esempio, per il modello mb ho in totale $10 \cdot 20 = 200$ combinazioni dei due parametri, avendo estratto 10

valori per la massa del neutrino e 20 per la densità barionica, ma nel modello *VBC_h* ho scelto di avere lo stesso numero di combinazioni ($n_c = 200$), estratte appunto utilizzando un Latin Hypercube, nonostante abbia 4 parametri, per evitare di avere tempi di training eccessivi. Il piccolo numero di combinazioni dei parametri cosmologici negli ultimi modelli non costituisce un problema, infatti, come si vede anche dall'esempio riportato in Figura 2.6, la regressione simbolica è in grado di raggiungere una buona precisione anche con training dataset di piccole dimensioni. Inoltre la componente simbolica dell'emulatore che ho costruito effettua un preprocessing del dataset su cui viene poi allenata la rete neurale, dunque la precisione del modello complessivo è affidata più alla componente neurale che a quella simbolica.

Oltre ai dataset per il training dei modelli, ho generato anche un validation dataset per ciascun modello, estraendo i parametri cosmologici con gli stessi criteri utilizzati per in precedenza. Per campionare il range dei parametri in modo diverso, ho scelto come numero di estrazioni per i validation dataset $2n + 1$, dove n è il numero di estrazioni per il training dataset. Per i modelli in cui ho utilizzato i Latin Hypercubes, ho estratto 301 combinazioni di parametri.

Costruzione dei modelli

Per la costruzione dei modelli di regressione simbolica, PySR fornisce un modulo, chiamato `PySRRegressor()`, di cui ho di cui ho descritto i parametri fondamentali nella Sezione 2.2. Nello specchietto Codice 3.1 ho illustrato la modalità con cui ho utilizzato questo modulo per la creazione dei vari modelli.

Il trainig di questi modelli, lanciato dalla funzione `model_mu.fit()`, è governato da tre parametri: il numero di iterazioni `niterations`, il numero di popolazioni `populations` e la complessità massima dell'espressione `maxsize`. Aumentare questi tre parametri significa effettuare allenamenti più lunghi, che spesso restituiscono espressioni simboliche più precise. Dunque la prima cosa che ho fatto è stata effettuare molti training del modello più semplice (i.e. il modello m) variando uno alla volta questi parametri e calcolando la funzione di costo al termine dell'allenamento. Tuttavia in tutti e tre i casi la loss function decresce molto lentamente e con ampie fluttuazioni, dovute alla stocasticità dell'evoluzione durante l'algoritmo di training. Perciò ho scelto come valori minimi per la creazione e il confronto dei primi modelli 200 iterazioni, 50 popolazioni ed una complessità massima di 20.

```

1 model_mu = PySRRegressor(
2     # operatori
3     binary_operators = ['+', '- ', '*', '/ ', '^ '],
4     unary_operators=['exp', 'log'],
5
6     # parametri per regolare la precisione del
7     # modello
8     niterations = iter,
9     populations = pop,
10    maxsize = comp,
11
12    batching = True,
13    batch_size = 50,
14
15    loss = 'L2DistLoss()',
16    model_selection = 'best',
17    equation_file = path
)

```

Codice 3.1: Dichiarazione di un modello per la regressione simbolica utilizzando PySR. Gli operatori sono le classiche operazioni aritmetiche, l'elevamento a potenza, l'esponenziale e il logaritmo naturale, inoltre è presente batching su campioni di 50 elementi ciascuno. Come loss function ho utilizzato l'errore quadratico medio e la modalità di selezione dell'espressione finale `best` permette di ottenere un modello che abbia valori piccoli per la loss e bassa complessità dell'equazione. Il training di questo modello genera tre files: un csv, un pkl e un file di backup, i quali vengono salvati al percorso specificato da `path`. I parametri che regolano la precisione del modello sono: il numero di iterazioni `niterations`, il numero di popolazioni da evolvere `populations` e la complessità massima delle espressioni simboliche `maxsize`.

Test sulla dipendenza dalle condizioni iniziali

Come discusso nella Sezione 1.4, la dipendenza dalle condizioni iniziali A_s e n_s dello spettro di potenza è racchiusa nella componente primordiale $P_{\text{prim}}(k)$ e dunque il fattore $\mu(k, z)$ non dovrebbe risentire di eventuali variazioni di questi parametri. Perciò ho costruito il modello *man*, utilizzando proprio A_s e n_s come parametri oltre alla massa del neutrino, per verificare che la regressione simbolica non trovi dipendenze in $\mu(k, z)$. Eseguendo un training con parametri `niterations = 200`, `populations = 50`, `maxsize = 20`, ho

ottenuto la seguente espressione per μ :

$$\mu(k, z) = \left[\left(0.988^{0.170^{\frac{k}{m_\nu}}} \right)^{m_\nu} \right]^{z + z^{0.550^z}}, \quad (3.5)$$

che, come si può vedere, non dipende da A_s e n_s . Perciò l'espressione ottenuta dalla regressione simbolica rispecchia la conoscenza fisica che noi abbiamo del fattore di crescita, essendo priva di dipendenze da A_s e n_s nonostante il modello sia stato allenato utilizzando proprio questi ultimi come parametri liberi.

Scelta del modello

Come accennato in precedenza, ho costruito vari modelli simbolici per $\mu(k, z)$ variando la base di parametri cosmologici utilizzata. Osservando la Tabella 3.1, si nota che, dei modelli realizzati, quelli che utilizzano tutti i parametri da cui dipende μ sono: *mbch*, *VBCCh*, *VBMh*. Di questi tre ho scelto il modello *VBCCh*, allenato sui parametri $\{\Omega_\nu, \Omega_b, \Omega_{cdm}, h\}$, per due motivi. Innanzitutto è utile avere un modello per $\mu(k, z)$ allenato su parametri di densità non ridotti, perché l'emulatore complessivo sfrutta anche l'espressione simbolica per $T(k)$ di Bartlett et al. [37], la quale era stata ottenuta allenando un modello in assenza di neutrini sulla seguente base di parametri: $\{\Omega_b, \Omega_m, h\}$. Infatti, poiché anche h è un parametro che varia, utilizzare $\omega_i = \Omega_i \cdot h^2$ non permetterebbe di coprire la stessa zona dello spazio delle combinazioni coperta utilizzando Ω_i . Inoltre, mentre Bartlett et al. [37] hanno considerato $\Omega_m^B = \Omega_b + \Omega_{cdm}$, nel mio caso la densità totale di materia comprende anche quella dovuta ai neutrini, perciò $\Omega_m = \Omega_b + \Omega_{cdm} + \Omega_\nu$. Ho quindi scelto una base in cui i contributi alla densità di materia rimanessero ben distinti, per evidenziare l'impatto generato dai neutrini.

Utilizzando dunque il modello *VBCCh*, con parametri `niterations = 4000`, `populations = 120`, `maxsize = 40`, ho ottenuto la seguente espressione per $\mu(k, z)$:

$$\mu_{\text{PySR}}(k, z) = \exp \left(\frac{\Omega_\nu z}{-1.13z + k \Omega_\nu^{-\frac{0.532}{h}} - \log(\Omega_b + \Omega_{cdm}) - 1.08} \right), \quad (3.6)$$

la quale, per k e z positivi, è sempre minore o uguale a 1, evidenziando il damping prodotto dai neutrini sullo spettro di potenza, e tende a 1 per grandi k

e piccoli z . Inoltre possiamo notare che Ω_b e Ω_{cdm} appaiono solamente sommati tra di loro, mentre Ω_ν dà un contributo indipendente. Per valutare la bontà di questo modello, in Figura 3.1 ho rappresentato le differenze percentuali tra i dati predetti tramite la (3.6) e i dataset di training e validation costruiti con CLASS. Come si può notare, il 99% degli errori è inferiore a 0.5% e il modello generalizza molto bene il suo potere predittivo al validation dataset: infatti il profilo degli errori percentuali, come anche i loro valori effettivi, sono sostanzialmente identici nei due casi.

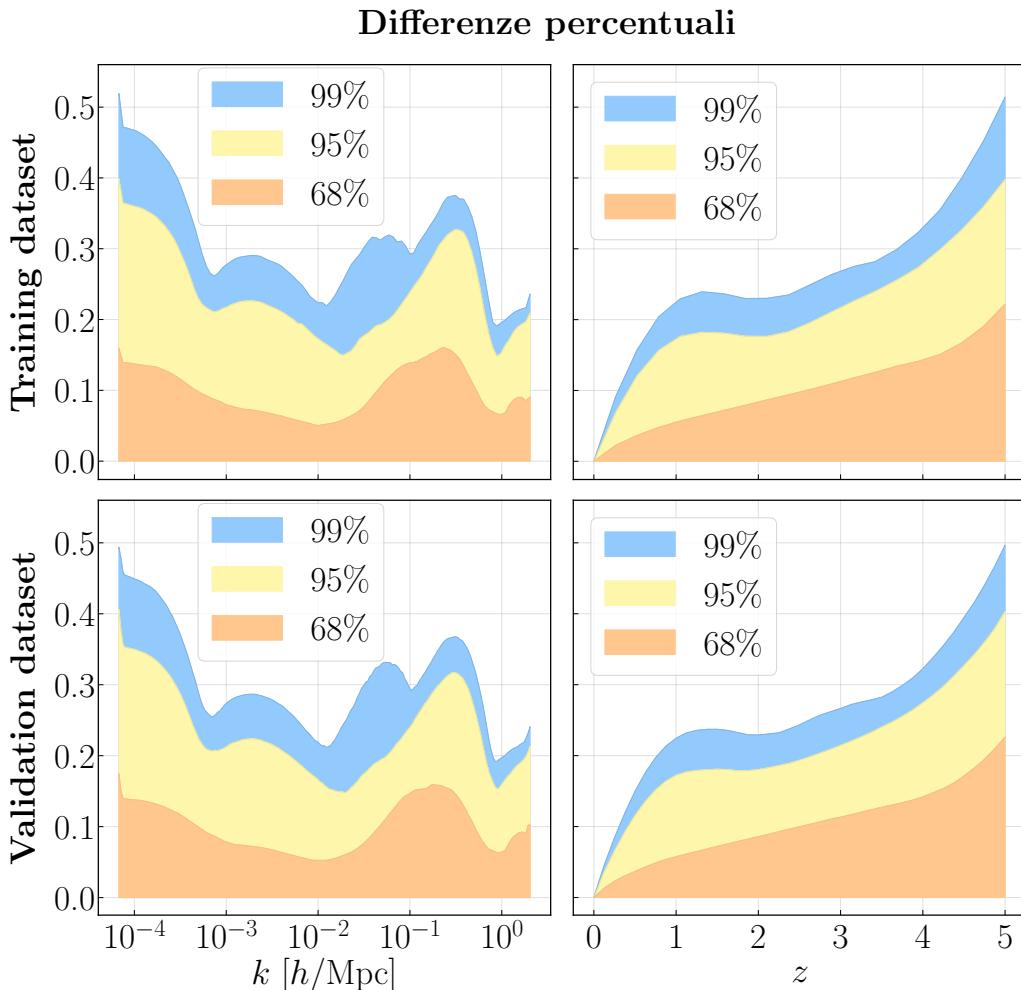


Figura 3.1: Differenze percentuali tra l'espressione simbolica per $\mu(k, z)$ (3.6) e i dataset di training e di validation, al variare di k e z , calcolate come $100 \cdot |1 - \mu_{\text{PySr}}/\mu_{\text{CLASS}}|$. Le aree colorate racchiudono gli errori relativi al 68%, 95% e 99% dei dati. L'accordo del modello simbolico con i dati è ottimo, con errori percentuali che non superano mai lo 0.5%.

3.2 Rete neurale e ricostruzione dello spettro

Di seguito descrivo la seconda parte del progetto, ossia la costruzione della rete neurale, scritta utilizzando la libreria Lux  [38] di Julia, e la ricostruzione dello spettro di potenza complessivo $P(k, z)$. L'obiettivo è quello di indagare se, inserendo la componente simbolica nella rete neurale, la sua performance migliora.

Generazione dei dataset

Al fine di ricostruire $P(k, z)$, ho allenato la rete neurale a predire la seguente quantità:

$$NN(k, z) \equiv \frac{P_{\text{CLASS}}(k, z)}{P_{\text{prim}}(k) T_{\text{SR}}^2(k) \mu_{\text{SR}}^2(k, z)}, \quad (3.7)$$

dove $P_{\text{CLASS}}(k, z)$ è lo spettro di potenza estratto da CLASS, $P_{\text{prim}}(k)$ è lo spettro primordiale dato dalla (1.51), mentre gli altri due termini rappresentano il contributo simbolico al preprocessing della rete. In particolare, ho utilizzato il codice sviluppato da Bartlett et al.  [37] per calcolare la funzione di trasferimento $T_{\text{SR}}(k)$ simbolica e $\mu_{\text{SR}}(k, z)$ è dato dall'espressione simbolica trovata in precedenza (3.6).

Ho quindi costruito un training dataset costituito da due array bidimensionali. Il primo rappresenta gli “input”, ed è un Latin Hypercube contenente $n_c = 2000$ combinazioni (*samples*) dei 4 parametri cosmologici (*features*) $\{\Omega_\nu, \Omega_b, \Omega_{cdm}, h\}$, estratti nei range specificati in Tabella 3.1. Il secondo invece ha $n_k \cdot n_z$ righe e n_c colonne e contiene, su ogni colonna, la quantità (3.7) calcolata per una fissata combinazione dei parametri cosmologici, al variare di k e z sulla stessa griglia avente $n_k = 100$ e $n_z = 20$ utilizzata nella sezione precedente. Come per i modelli di regressione simbolica, anche per la rete neurale ho creato un validation dataset, mantenendo n_k e n_z invariati e selezionando 1000 combinazioni dei parametri cosmologici.

Caratteristiche e training della rete neurale

La componente neurale del mio emulatore è costituita da un multi-layer perceptron di Lux con 5 strati nascosti da 64 neuroni ciascuno, dunque è una rete neurale di tipo feed-forward, in cui ogni nodo riceve input da tutti quelli del

layer precedente. Questa rete prende in input un array bidimensionale di 4 features (i parametri cosmologici $\{\Omega_\nu, \Omega_b, \Omega_{cdm}, h\}$) e N samples (le combinazioni dei parametri) e restituisce in output un array contenente la quantità (3.7) al variare di k e z per ciascuna delle N combinazioni. La funzione di attivazione scelta per ciascun neurone è la tangente iperbolica (2.7), mentre come loss function per il training ho utilizzato lo scarto quadratico medio tra i valori predetti dalla rete e quelli del training dataset (2.9). Infine, per eseguire la backpropagation mi sono servito dell'ottimizzatore Adam [39].

Per effettuare il training della rete neurale ho inizializzato il learning rate a 0.001, poi ho costruito 20 cicli di allenamento da 20 000 epoche ciascuno. Ad ogni epoca lo stato del modello viene aggiornato secondo l'algoritmo di Stochastic Gradient Descent, tuttavia il modello viene salvato solamente nel caso in cui la loss function valutata sul validation dataset diminuisca. In tal modo ho ottenuto un modello che minimizza la loss sul training dataset evitando il rischio di overfitting.Terminate le 20 000 epoche di ogni ciclo, il learning rate viene diviso per 5, per ottenere una maggior precisione nell'individuazione del minimo della loss function.

Ricostruzione dello spettro

Una volta allenata la rete neurale, l'emulatore è completato e lo si può usare per ricostruire lo spettro di potenza complessivo. La fase finale del mio lavoro, dunque, è stata la ricostruzione di $P(k, z)$ con l'emulatore neuro-simbolico e il confronto con quello fornito da **CLASS**. Ho quindi invertito l'espressione (3.7) a dare:

$$P_{\text{CLASS}}(k, z) = NN(k, z) P_{\text{prim}}(k) T_{\text{SR}}^2(k) \mu_{\text{SR}}^2(k, z), \quad (3.8)$$

dove si ricorda che $NN(k, z)$ è la predizione della rete neurale, $P_{\text{prim}}(k)$ è l'espressione analitica (esatta) dello spettro primordiale, $T_{\text{SR}}(k)$ è la funzione di trasferimento simbolica ricavata da Bartlett et al. [37], $\mu_{\text{SR}}(k, z)$ è la correzione al fattore di crescita ricavata da me tramite la regressione simbolica. Lo spettro di potenza ottenuto è in ottimo accordo con quello restituito da **CLASS**, come ho mostrato in Figura 3.2 per una singola combinazione di parametri cosmologici.

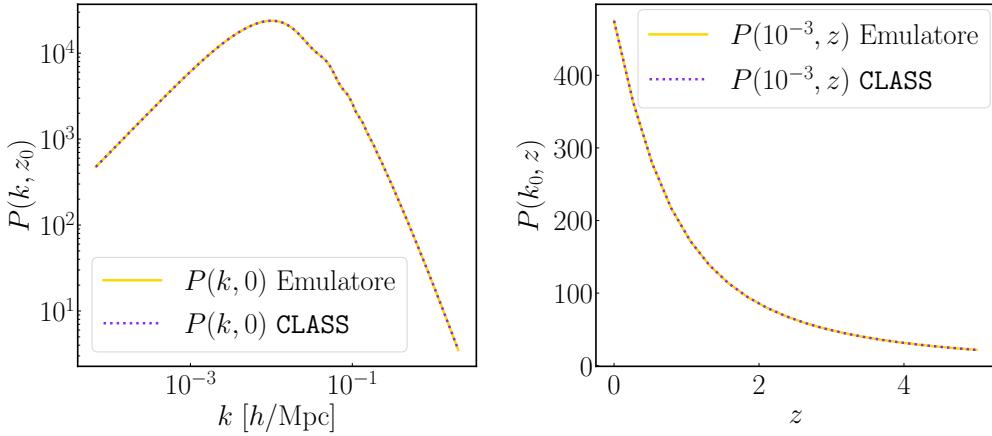


Figura 3.2: Ricostruzione dello spettro di potenza ottenuto dall'emulatore e confronto con il risultato fornito da **CLASS**. Nel plot a sinistra ho fissato il valore del redshift a $z = 0$, mentre in quello a destra k è fissato a 10^{-3} . Come si può facilmente notare, l'accordo tra i dati e il modello è ottimo, infatti gli errori percentuali massimi al variare di k e z sono di circa lo 0.15%.

3.3 Risultati ottenuti

La ricostruzione dello spettro di potenza per le fluttuazioni di densità attraverso l'emulatore neuro-simbolico è avvenuta con successo. Mediante la regressione simbolica, ho ottenuto un'espressione per il termine correttivo $\mu(k, z)$ che la massa dei neutrini provoca sul fattore di crescita $D(k, z)$, con un errore percentuale al variare di k e z sempre inferiore allo 0.5%, come si evince dalla Figura 3.1. Ho quindi utilizzato tale espressione, insieme a quella per la funzione di trasferimento $T(k)$ ricavata da Bartlett et al. [37], per eseguire il preprocessing del dataset su cui ho allenato una rete neurale. Tale rete restituisce in output la predizione $NN(k, z)$ data dalla (3.7), che ho utilizzato per ricostruire lo spettro complessivo, ottenendo un'ottima compatibilità, come mostrato in Figura 3.3. Gli errori percentuali sullo spettro di potenza predetto sono inferiori allo 0.4% su tutto il range di k e z considerato, ad eccezione di una piccola regione di k attorno a 10^{-2} . In tale regione l'errore aumenta drasticamente fino ad arrivare allo 0.5% rispetto al training dataset e allo 0.8% rispetto al validation dataset. Tuttavia, come si può notare dalla Figura 3.1, l'emulatore simbolico da me costruito a $k = 10^{-2}$ presenta un errore che ammonta circa allo 0.3%, senza mostrare comportamenti anomali. Posso quindi dedurre che questo picco di errore sia dovuto ad una carenza di precisione che l'emulatore simbolico costruito da Bartlett et al. [37] sembra avere per questi

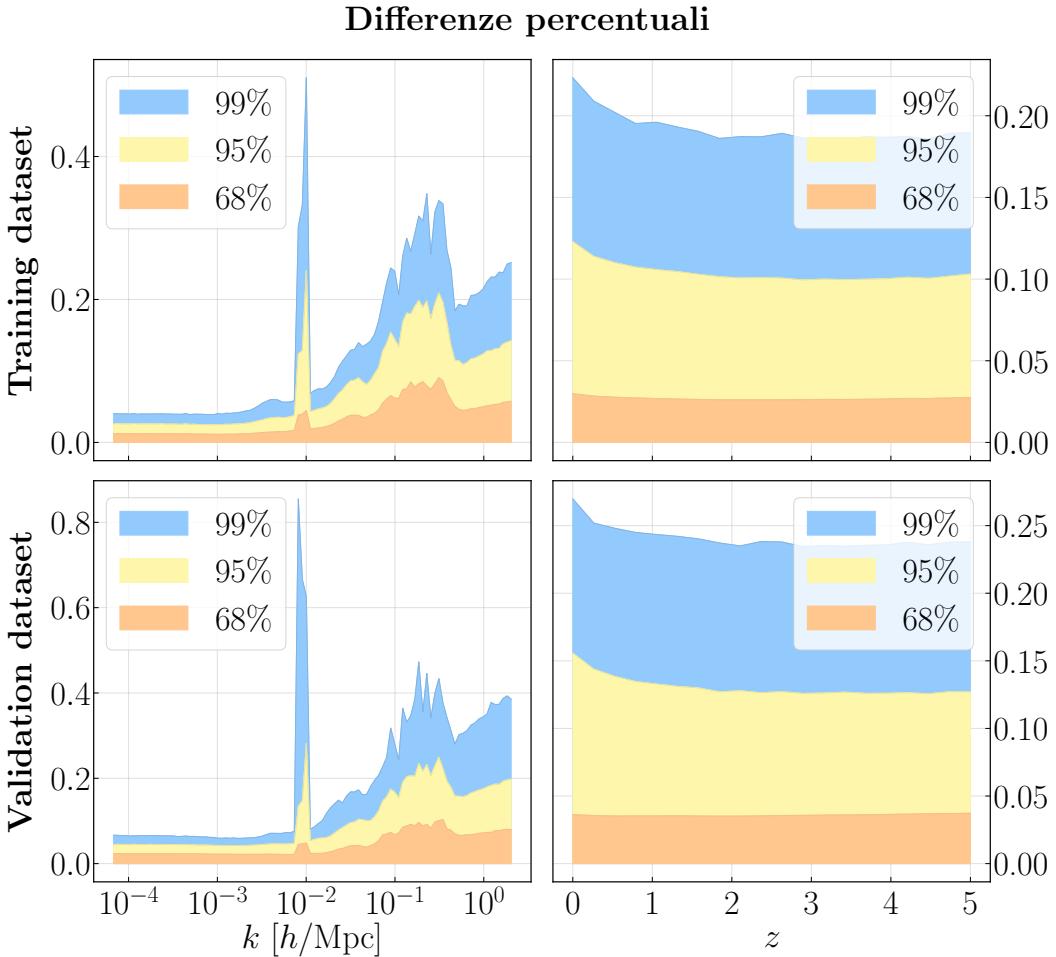


Figura 3.3: Differenze percentuali tra lo spettro di potenza predetto dall'emulatore neuro-simbolico e i dataset di training e validation, al variare di k e z , calcolate come $100 \cdot |1 - P_{\text{Emulatore}}/P_{\text{CLASS}}|$. Le aree colorate racchiudono gli errori relativi al 68%, 95% e 99% dei dati. Gli errori sono inferiori allo 0.4% per tutto il range di k e z considerato, ad eccezione di una piccola regione attorno a $k = 10^{-2}$, nella quale l'errore dell'emulatore simbolico di Bartlett et al. [37] genera un picco a 0.5% per il training dataset e 0.8% per il validation dataset.

valori di k . Ad ogni modo, il risultato è incoraggiante, perché la precisione di questo emulatore neuro-simbolico è molto elevata nonostante la piccola dimensione del training dataset, che contiene solamente 100 valori di k , 20 valori di z e 2000 combinazioni degli altri parametri cosmologici, valori che sono di molto inferiori a quelli comunemente utilizzati in letteratura per scopi analoghi [40] [41].

Rilevanza della componente simbolica dell'emulatore

In ultimo, ho confrontato i risultati dell'emulatore neuro-simbolico con quelli di un emulatore più semplice, costituito da una rete neurale senza componente simbolica, al fine di verificare se l'aggiunta di quest'ultima induca miglioramenti o peggioramenti nella performance. Confrontando i risultati ho riscontrato che, a parità di training dataset, complessità della rete neurale e durata dell'allenamento, l'emulatore in cui ho eseguito il preprocessing mediante la regressione simbolica presenta un errore inferiore rispetto alla semplice rete neurale di un fattore che va da 2 a 10. Per facilitare il confronto tra i due emulatori ho riportato le mappe degli errori al variare di k e z in Figura 3.4, da cui risulta chiaro che il modello neuro-simbolico ha una precisione maggiore soprattutto agli estremi dei range di k e z , con particolare riguardo ai bassi valori di k . Sottolineo che il training dataset di entrambi i modelli è costituito da un Latin Hypercube di 2000 combinazioni di parametri cosmologici, tuttavia, per il training dell'emulatore puramente neurale, ai 4 parametri su cui è allenato l'emulatore neuro-simbolico vanno aggiunti anche A_s e n_s , poiché sappiamo che lo spettro di potenza dipende da essi attraverso $P_{\text{prim}}(k)$. Questi due parametri però non sono necessari per allenare l'emulatore neuro-simbolico, perché la dipendenza analitica dallo spettro primordiale è stata tolta calcolando il rapporto (3.7) e la funzione di trasferimento di Bartlett et al. [37] dipende dalle condizioni iniziali solo per quanto riguarda la normalizzazione. Inoltre, ho verificato in precedenza che il fattore correttivo $\mu(k, z)$ non dipende da A_s e n_s , come mostrato nella relazione (3.5), e per questo non è necessario aggiungere tali parametri al training dataset.

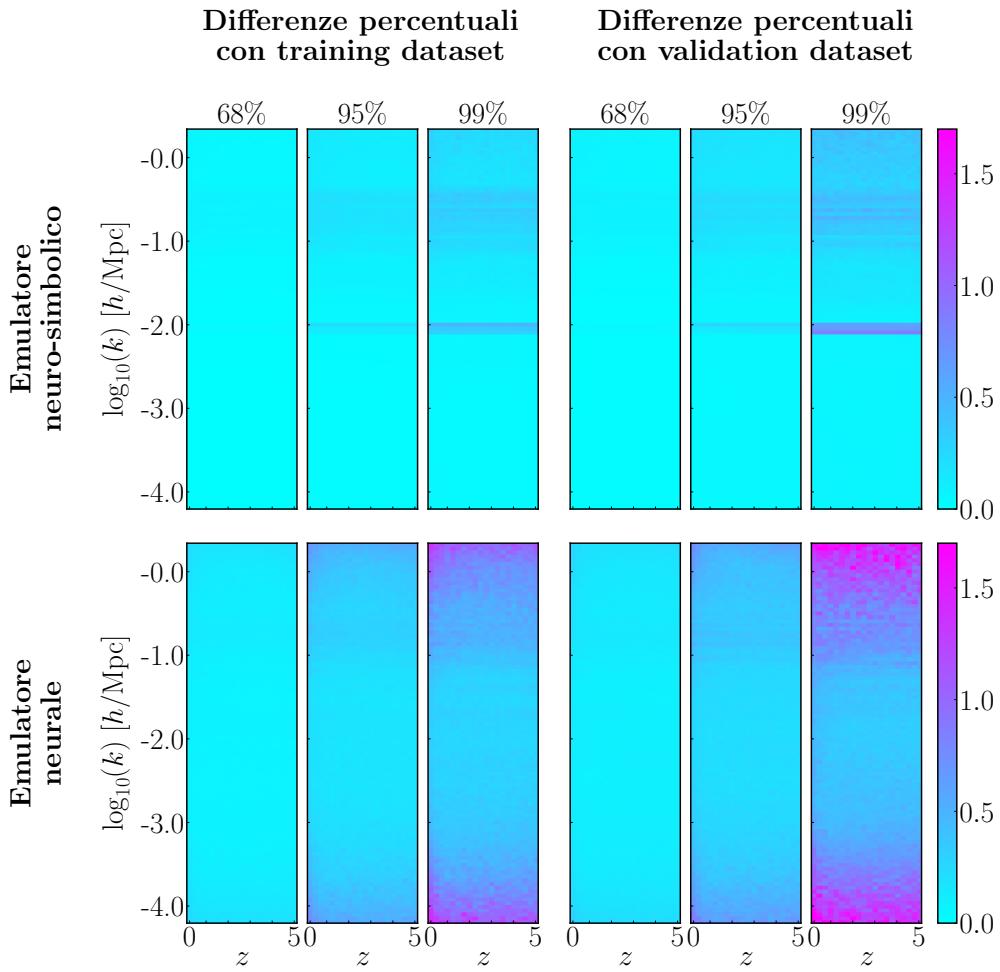


Figura 3.4: Mappe degli errori percentuali dei due emulatori in funzione di k e z , calcolati come $100 \cdot |1 - P_{\text{Emulatore}}/P_{\text{CLASS}}|$. I plot in alto si riferiscono all'emulatore neuro-simbolico, mentre quelli in basso all'emulatore puramente neurale, inoltre, da sinistra verso destra sono rappresentati gli errori del 68%, 95% e 99% dei dati e la scala di colore è la medesima per tutti i grafici. Come si evince dai plot, l'emulatore neuro-simbolico ha una precisione superiore rispetto alla sola rete neurale, eccezion fatta per la zona a $k = 10^{-2}$, dove gli errori dei due modelli sono confrontabili. Si noti che la differenza dei due modelli è particolarmente evidente agli estremi dei range di definizione di k e z (in particolare a bassi k).

Conclusioni

In questo progetto di tesi ho applicato delle tecniche di Machine Learning allo studio statistico delle fluttuazioni di densità dell’Universo, che descrivono la sua struttura a grande scala. Tali fluttuazioni sono descritte, tramite la loro decomposizione in modi di Fourier, dallo spettro di potenza $P(k, z)$, in cui la dipendenza da k è racchiusa nella funzione di trasferimento $T(k)$ mentre quella da z è descritta dal fattore di crescita $D(k, z)$. A sua volta $D(k, z)$ è scomponibile in un termine dipendente unicamente dal redshift $\bar{D}(z)$, ottenuto come limite a grandi scale, e in un fattore correttivo $\mu(k, z)$ dovuto al valore finito della massa dei neutrini. Lo spettro di potenza e le sue componenti sono governati dalle equazioni di Einstein-Boltzmann, di cui tuttavia non esistono soluzioni esatte; ciò rende necessario l’utilizzo di metodi numerici per integrare tali equazioni, chiamati Einstein-Boltzmann solvers, i quali sono precisi ma poco efficienti. Sfruttando i recenti emulatori che utilizzano alcune tecniche di Machine Learning è possibile calcolare lo spettro di potenza in maniera più efficiente di quanto fatto finora. In particolare, le Reti Neurali possono essere allenate per apprendere come riprodurre $P(k, z)$, tuttavia il loro output è di difficile interpretazione fisica, in quanto tali modelli utilizzano relazioni funzionali complesse per esprimere la dipendenza dello spettro dai parametri cosmologici.

In questo contesto ho sfruttato, oltre alle Reti Neurali, un’altra tecnica di Machine Learning, chiamata Regressione Simbolica, per costruire un emulatore dello spettro di potenza che avesse una maggiore interpretabilità fisica e fosse più efficiente di una semplice Rete Neurale. Ho dunque utilizzato la libreria PySR per eseguire la Regressione Simbolica sulla componente $\mu(k, z)$ del fattore di crescita, in dipendenza dai seguenti parametri cosmologici: $\{\Omega_\nu, \Omega_b, \Omega_{cdm}, h\}$. Dal training del modello simbolico ho ottenuto l’espressione (3.6), i cui errori massimi al variare di k e z sono inferiori allo 0.5%, come mostrato in Figura 3.1. Successivamente ho utilizzato questa espressione simbolica per effettuare il preprocessing del training dataset di una Rete Neurale, la quale costituisce la componente neurale del mio emulatore, secondo l’equazione (3.7). Infine,

Conclusioni

sfruttando l'output della Rete Neurale allenata, ho ricostruito lo spettro di potenza complessivo, ottenendo errori percentuali massimi inferiori allo 0.4% per tutto il range di k e z , ad eccezione di una piccola zona attorno a $k = 10^{-2}$, come si evince dalla Figura 3.3.

In ultimo, ho confrontato il risultato ottenuto attraverso l'emulatore neuro-simbolico da me costruito con quello ottenuto da un altro emulatore, costituito da una semplice Rete Neurale. A parità di complessità della rete, training dataset e durata dell'allenamento, l'emulatore neuro-simbolico presenta errori percentuali più bassi di un fattore tra 2 e 10 rispetto a quello puramente neurale, ad indicare il fatto che il preprocessing effettuato mediante la Regressione Simbolica ha permesso di incrementare notevolmente l'efficienza del modello.

La combinazione della Regressione Simbolica alle Reti Neurali risulta dunque essere una tecnica promettente nell'affrontare problemi di cosmologia e in futuro molto probabilmente verrà utilizzata per risolvere problemi ancora più complessi. Un ulteriore sviluppo di questo progetto di tesi potrebbe essere ad esempio il miglioramento della precisione dell'emulatore simbolico per la funzione di trasferimento $T(k)$, al fine di abbattere l'errore sullo spettro di potenza complessivo anche nella zona attorno a $k = 10^{-2}$, nella quale l'emulatore neuro-simbolico presenta errori paragonabili a quelli del modello puramente neurale. Un altro interessante caso potrebbe essere l'utilizzo del fattore di crescita indipendente dalla scala $\bar{D}(z)$ nel preprocessing del training dataset della Rete Neurale che ricostruisce $P(k, z)$.

Bibliografia

- [1] R. M. Wald. *General Relativity*. University of Chicago Press, 2010.
- [2] B. R. Granett, L. Guzzo et al. *The power spectrum from the angular distribution of galaxies in the CFHTLS-Wide fields at redshift ~ 0.7 : The galaxy power spectrum at $z \sim 0.7$* . Gen. 2012. DOI: 10.1111/j.1365-2966.2011.20297.x.
- [3] L. P. Eisenhart. *Riemannian Geometry*. Vol. 51. Princeton University Press, 1997.
- [4] E. Hubble. *A relation between distance and radial velocity among extra-galactic nebulae*. 1929. DOI: 10.1073/pnas.15.3.168.
- [5] Planck Collaboration. *Planck 2018 results - VI. Cosmological parameters*. 2020. DOI: 10.1051/0004-6361/201833910.
- [6] S. Perlmutter e B. P. Schmidt. *Measuring Cosmology with Supernovae*. 2003. DOI: 10.1007/3-540-45863-8_11.
- [7] M. S. Longair. *Galaxy Formation*. Springer Science & Business Media, 2008.
- [8] G. Bertone e D. Hooper. *History of dark matter*. Ott. 2018. DOI: 10.1103/revmodphys.90.045002.
- [9] URL: <https://physicsanduniverse.com/galaxy-rotation-curve-dark-matter/>.
- [10] A. A. Penzias e R. W. Wilson. *A Measurement of Excess Antenna Temperature at 4080 Mc/s*. Lug. 1965. DOI: 10.1086/148307.
- [11] D. Tong. *Cosmology*. University of Cambridge, 2009.
- [12] D. J. Fixsen et al. *The Cosmic Microwave Background Spectrum from the Full COBE FIRAS Data Set*. Dic. 1996. DOI: 10.1086/178173.
- [13] Planck Collaboration. *Planck 2018 results - I. Overview and the cosmological legacy of Planck*. 2020. DOI: 10.1051/0004-6361/201833880.

Bibliografia

- [14] A. G. Riess et al. *Large Magellanic Cloud Cepheid Standards Provide a 1% Foundation for the Determination of the Hubble Constant and Stronger Evidence for Physics beyond Λ CDM*. *Mag.* 2019. DOI: 10.3847/1538-4357/ab1422.
- [15] E. Abdalla et al. *Cosmology intertwined: A review of the particle physics, astrophysics, and cosmology associated with the cosmological tensions and anomalies*. *Giu.* 2022. DOI: 10.1016/j.jheap.2022.04.002.
- [16] S. Dodelson e F. Schmidt. *Modern Cosmology*. Academic Press, 2020.
- [17] Angeliki Kiakotou, Øystein Elgarøy e Ofer Lahav. *Neutrino mass, dark energy, and the linear growth factor*. Mar. 2008. DOI: 10.1103/physrevd.77.063005. URL: <http://dx.doi.org/10.1103/PhysRevD.77.063005>.
- [18] H. A. Winther. *Cosmology II*. URL: https://cmb.wintherscoming.no/theory_perturbations.php.
- [19] E. Bellini et al. *Comparison of Einstein-Boltzmann solvers for testing general relativity*. *Gen.* 2018. DOI: 10.1103/physrevd.97.023520.
- [20] A. Lewis, A. Challinor e A. Lasenby. *Efficient Computation of Cosmic Microwave Background Anisotropies in Closed Friedmann-Robertson-Walker Models*. Ago. 2000. DOI: 10.1086/309179.
- [21] D. Blas, J. Lesgourgues e T. Tram. *The Cosmic Linear Anisotropy Solving System (CLASS). Part II: Approximation schemes*. Lug. 2011. DOI: 10.1088/1475-7516/2011/07/034.
- [22] K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT Press, 2012.
- [23] R. Wu et al. *Deep Image: Scaling up Image Recognition*. 2015. arXiv: 1501.02876 [cs.CV].
- [24] J. Tan e Y. Zhang. *ExplainableFold: Understanding AlphaFold Prediction with Explainable AI*. 2023. arXiv: 2301.11765 [cs.AI].
- [25] F. Marquardt. *Machine learning and quantum devices*. 2021. DOI: 10.21468/SciPostPhysLectNotes.29.
- [26] G. Cybenko. *Approximation by superpositions of a sigmoidal function*. 1989. DOI: 10.1007/BF02551274.
- [27] J. McGonagle, G. Shaikouski e C. Williams. *Backpropagation*. URL: <https://brilliant.org/wiki/backpropagation>.
- [28] D. Angelis, F. Sofos e T. E. Karakasidis. *Artificial Intelligence in Physical Sciences: Symbolic Regression Trends and Perspectives*. 2023. DOI: 10.1007/s11831-023-09922-z.

Bibliografia

- [29] I. A. Abdellaoui e S. Mehrkanoon. *Symbolic regression for scientific discovery: an application to wind speed forecasting*. 2021. arXiv: 2102.10570.
- [30] S. Stijven et al. *Prime-Time: Symbolic Regression Takes Its Place in the Real World*. Dic. 2016. DOI: 10.1007/978-3-319-34223-8_14.
- [31] P. Lemos et al. *Rediscovering orbital mechanics with machine learning*. 2022. arXiv: 2202.02306 [astro-ph.EP].
- [32] S. Forrest. *Genetic algorithms*. 1996. URL: <https://scholar.google.com/scholar?q=Forrest+S+1996+ACM+Comput.+Surv.+28+77%2E2%80%9380>.
- [33] M. Mitchell. *Genetic algorithms: An overview*. 1995. DOI: <https://doi.org/10.1002/cplx.6130010108>.
- [34] M. Cranmer. *Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl*. 2023. arXiv: 2305.01582 [astro-ph.IM].
- [35] C. G. Broyden. *The Convergence of a Class of Double-rank Minimization Algorithms 1. General Considerations*. Mar. 1970. DOI: 10.1093/imamat/6.1.76.
- [36] M. Cranmer. *PySRRegressor Reference*. URL: <https://astroautomata.com/PySR/api/>.
- [37] D. J. Bartlett et al. *A precise symbolic emulator of the linear matter power spectrum*. 2023. arXiv: 2311.15865 [astro-ph.CO].
- [38] A. Pal. *Lux: Explicit Parameterization of Deep Neural Networks in Julia*. Ver. v0.5.0. If you use this software, please cite it as below. Apr. 2023. DOI: 10.5281/zenodo.7808904. URL: <https://doi.org/10.5281/zenodo.7808904>.
- [39] D. P. Kingma e J. Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].
- [40] M. Bonici et al. *Capse.jl: efficient and auto-differentiable CMB power spectra emulation*. Gen. 2024. DOI: 10.21105/astro.2307.14339.
- [41] A. Spurio Mancini et al. *CosmoPower: emulating cosmological power spectra for accelerated Bayesian inference from next-generation surveys*. Gen. 2022. DOI: 10.1093/mnras/stac064.

