Università degli Studi di Padova
Dipartimento di Scienze Statistiche

Corso di Laurea Magistrale
in Scienze Statistiche

Relazione finale:

Un modello gerarchico bayesiano per l'analisi di geni differenzialmente espressi

Relatore:

Prof. Davide Risso

Laureando:

Davide Pagin

Matricola N. 1206985

Anno accademico 2020/2021

Indice

Elenco delle tabelle

Elenco delle figure

Introduzione

Nel corso degli ultimi 15 anni si è assistito in ambito genomico alla nascita di nuove tecnologie che hanno rivoluzionato il modo di sequenziare l'RNA dalle cellule. La novità principale è stata l'introduzione della tecnologia RNA-seq che ha rimpiazzato la precedente dei microarray e ha comportato importanti innovazioni: ad esempio è stato superato il limite dei microarray di poter studiare solamente i geni noti.
Il passo successivo nell'avanzamento tecnologico sono stati i protocolli "scRNA-seq"che hanno permesso il sequenziamento dell'RNA a livello di singola cellula. Negli ultimi anni dunque sono stati proposti una serie di modelli per analizzare le differenze di espressione genica tenendo conto dell'informazione data dalla singola cellula.
Essendo un campo relativamente nuovo la parte modellistica dell'analisi statistica è un terreno in continua esplorazione. I modelli più consolidati sono stati implementati con l'avvento dell'RNA-seq e successivamente sono stati usati in maniera efficace anche per i protocolli scRNA-seq. In questa tesi vengono presentati e utilizzati tre tra i modelli più frequenti nell'analisi statistica per dati genomici: EdgeR, Deseq2 e Limma. Il problema di questi metodi sorge quando il sequenziamento dell'RNA viene eseguito su diversi campioni biologici (uomini, topi ecc.). In questo caso i modelli non tengono conto dell'informazione gerarchica data dal diverso campione sequenziato. Di con-

seguenza in questa tesi si è cercato di proporre un modello che tenesse conto dell'informazione aggiuntiva data dal campione biologico. Per raggiungere questo fine ci si è serviti della statistica bayesiana e dell'algoritmo Markov Chain Monte Carlo (MCMC): un metodo per calcolare la distribuzione a posteriori di una variabile quando la sua distribuzione a priori non è coniugata con quella della variabile risposta.

Nel corso della tesi tutti questi concetti verranno spiegati in modo più chiaro e dettagliato; in particolare la tesi si struttura in 4 capitoli.

Nel Capitolo 1 viene introdotto il contesto biologico, che risulta essenziale per comprendere a pieno l'ambito in cui si svolge il sequenziamento dell'RNA; quest'ultimo invece viene descritto nel secondo Paragrafo insieme ai protocolli scRNA-seq. Nell'ultima parte del primo Capitolo vengono esposte le peculiarità del dataset che verrà utilizzato verso la fine della tesi per testare il modello anche sui dati reali. Questo dataset si basa su un esperimento che si è posto l'obiettivo di studiare l'effetto della privazione del sonno sui topi domestici.

Il Capitolo 2 è la sezione più sostanziale della tesi dal punto di vista della teoria statistica. Infatti vengono delineati in sequenza: i modelli gerarchici, l'inferenza bayesiana per questa categoria di modelli, i metodi MCMC, gli algoritmi di Metropolis-Hastings e Hamiltoniano Monte Carlo e infine l'inferenza variazionale (VI).

L'inferenza variazionale è un metodo con lo stesso obiettivo dell'MCMC di calcolare una distribuzione a posteriori, ma riduce di molto il tempo computazionale impiegato da quest'ultimo. Gli algoritmi illustrati nel secondo Capitolo sono quelli che vengono utilizzati dal software Stan che è stato impiegato nelle analisi. Stan viene adoperato all'interno della piattaforma R e insieme ai pacchetti del software Bioconductor, utilizzati anch'essi all'interno

di R, costituiscono la componente principale del materiale software scelto per le analisi.

Nel Capitolo 3 è proposto il modello gerarchico bayesiano per l'analisi di dati genomici sequenziati da più campioni, nel modello sono specificate le distribuzioni a priori per ogni variabile e la scelta di ogni distribuzione viene opportunamente giustificata. Prima di introdurre il modello definitivo ne viene proposto un altro che trova il suo spazio d'applicazione quando il sequenziamento dell'RNA viene eseguito su un solo campione, questo modello pone le basi metodologiche per quello finale.

Nel Capitolo 4 vengono testate le performance di entrambi i modelli su dati simulati tramite il pacchetto Splatter e successivamente vengono confrontate con le prestazioni dei 3 metodi classici tramite l'analisi delle curve ROC e dell'indicatore AUC che esprime l'area sotto la curva. L'obiettivo principale dei modelli è identificare in modo corretto i geni differenzialmente espressi e sarà possibile osservare le capacità predittive di ogni modello.

Infine nell'ultima parte del quarto Capitolo viene applicato il modello bayesiano gerarchico sui dati reali e tramite una Over Representation Analysis viene assegnato un significato biologico al gruppo di geni identificati come differenzialmente espressi.

Capitolo 1

scRNA-seq e dataset di riferimento

Nella prima parte di questo Capitolo sarà definito in maniera dettagliata il contesto biologico di riferimento tramite la spiegazione delle componenti principali della genomica, una branca della biologia molecolare che si occupa prevalentemente dell'analisi del genoma e delle sue caratteristiche quali: la struttura, il contenuto, la funzione e l'evoluzione [1]. Nel secondo Paragrafo verranno introdotti i comuni metodi di sequenziamento delle cellule con un approfondimento sulle analisi di tipo "*single-cell*". Nel terzo Paragrafo saranno spiegati i concetti biologici e statistici chiave del dataset reale analizzato in questa tesi. Pertanto verranno evidenziati il ruolo della corteccia prefrontale negli organismi, le peculiarità del topo domestico (*mus musculus*) e la finalità e la progettazione della sperimentazione.

[1] https://it.wikipedia.org/wiki/Genomica

1.1 Contesto biologico

Il fulcro della scienza genomica è la cellula, l'unità morfologica e funzionale fondamentale in tutti gli organismi viventi per lo svolgimento dei processi biologici. Nel nucleo della cellula risiede l'informazione ereditaria codificata nel DNA, che può essere trasmessa a generazioni successive di cellule durante la replicazione cellulare.
La cellula è strutturalmente definita da una membrana cellulare che racchiude il citoplasma in cui si localizzano organelli e nucleo, Il DNA in esso contenuto rappresenta l'impronta genetica di ogni organismo vivente ed è quindi la base della trasmissione ereditaria dei caratteri. Il DNA si caratterizza per una struttura tridimensionale formata da due eliche superavvolte una sull'altra. Ciascuna elica è data dalla ripetizione di una serie di nucleotidi, ossia l'insieme di una base azotata, una molecola di deossiribosio e un gruppo fosfato.
Le basi azotate sono imprescindibili per descrivere la composizione del DNA, in totale sono quattro: Adenina (A), Timina (T), Guanina (G), Citosina (C). Queste sono rivolte verso l'interno dell'elica e si accoppiano secondo lo schema A-T, G-C.
Il DNA tradotto in RNA trasporta l'informazione che viene decodificata durante la sintesi proteica per formare le proteine, le quali sono costituite da catene di amminoacidi e adempiono a una grande gamma di funzioni all'interno degli organismi viventi.
L'entità biologica più rilevante per l'analisi presentata è il gene. Il gene non è nient'altro che una porzione di DNA che codifica per una proteina, dunque sta alla base di tutte le funzioni vitali. Il meccanismo che consente il passaggio da gene a proteina può essere riassunto in due step:

1) **Trascrizione**: In questa fase un gene viene trascritto in un frammento

complementare di mRNA (RNA messaggero)

2) **Traduzione**: Il filamento di mRNA viene letto nel ribosoma attivando la sintesi della proteina associata.

Così descritto sembrerebbe che all'interno di ogni cellula ci sia un meccanismo di egual misura di trascrizione e traduzione dei geni che porterebbe ad avere per ogni cellula la produzione delle stesse proteine, in realtà la codifica delle proteine dipende dalla regolazione dell'espressione genica, che ogni cellula compie in modo tale da avere solamente le proteine che le servono per vivere e svolgere la propria funzione.
La particolarità dell'espressione genica è il meccanismo di copia-incolla nella trascrizione da DNA a RNA, infatti ogni copia del gene può produrre molte copie di RNA e di conseguenza di proteine. Questo aspetto dell'espressione genica dipende dalla composizione del gene, il quale presenta delle sequenze codificanti (dette esoni) e delle sequenze non codificanti (gli introni). Nel momento in cui il gene viene trascritto in m-RNA, avviene un processo di splicing in cui viene tenuta solamente l'informazione codificante e dunque vengono rimossi gli introni.
In alcuni casi può avvenire il fenomeno dello splicing alternativo, ovvero è possibile che ci sia un riarrangiamento degli esoni in modo tale che lo stesso gene possa produrre tipi diversi di proteine.
Le sequenze codificate a partire dallo stesso gene ma ottenute con assemblamenti diversi degli esoni vengono chiamate isoforme o trascritti del gene.

1.2 RNA-seq

La tecnologia di sequenziamento RNA-seq è un avanzamento tecnologico dei microarray i quali erano lo strumento principale per la misurazione dell'e-

spressione genica dalla meta degli anni 90. I microarray avevano diversi limiti, in particolare: la necessità di conoscere a priori i geni su cui valutare l'espressione e il "rumore di fondo"che portava a potenziali dati imprecisi e incompleti minando la specificità del procedimento, per questi motivi dal 2008 si è passati all'RNA-Seq.

Tra i primi a descrivere il sequenziamento RNA ci sono Wang et al. (2009) che nel loro articolo affermavano: "Questo metodo, denominato RNA-Seq (sequenziamento dell'RNA) presenta chiari vantaggi rispetto agli approcci esistenti e si prevede che rivoluzionerà il modo in cui vengono analizzati i trascrittomi eucarioti".

Gli autori hanno dimostrato come l'RNA-seq permetta anche lo studio di geni non conosciuti e riesca a quantificare l'espressione genica con un accuratezza maggiore, inoltre il grande vantaggio dell'RNA è la possibilità di quantificare l'espressione dei geni a livello di singola cellula dal momento che la tecnologia richiede una quantità di RNA minore rispetto i microarray. Questa novità dell'RNA-Seq è decisiva per lo studio riportato in questa tesi, laddove si è operato con dati a singola cellula sia nelle simulazioni sia nella parte conclusiva dove vengono esaminati i dati reali.

Ma più precisamente cosa si intende per sequenziamento dell'RNA?

Per sequenziamento si intende la lettura dell'esatta sequenza di nucleotidi di un acido nucleico (RNA o DNA). Dal momento che un filamento di RNA può avere fino a qualche miliardo di nucleotidi, non è possibile leggere l'intera sequenza dell'acido nucleico poiché i dispositivi di nuova generazione (NGS) hanno una capacità di lettura limitata, dunque viene letto solamente un frammento che viene denominato *read*. Le *read* sono quindi delle piccole sequenze di nucleotidi e con le nuove tecnologie arrivano a contenere fino a un massimo di cento nucleotidi. Dopo la lettura delle sequenze il passaggio

successivo è l'allineamento; questo consiste nell'individuare la posizione di origine della *read* nel genoma, in modo tale da poter assegnare ogni *read* a un gene e in seguito poter contare il numero di *read* per ogni gene. Nel prossimo Paragrafo vedremo più nel dettaglio il processo di sequenziamento nel caso dei protocolli *single cell* RNA-seq.

1.2.1 scRNA-seq

I protocolli *single-cell* RNA-seq misurano l'espressione genica sequenziando l'RNA cellula per cellula, questo metodo è stato dimostrato essere costo efficiente e molto accurato, migliorando la comprensione della complessità dell'espressione genica (Tang et al., 2009).
Questi protocolli prevedono una serie di operazioni quali: la dissociazione enzimatica, l'isolamento cellulare, la creazione di librerie ed il sequenziamento. La dissociazione enzimatica è il primo passaggio nel processo di sequenziamento e serve per poter demolire la matrice cellulare che tiene unite le cellule, dopo aver compiuto questo passo si può procedere all'isolamento cellulare. Per isolare le cellule ci sono molte possibilità, di seguito verranno elencati i tre metodi principali. Ognuno di questi si differenzia per il numero di cellule che riesce ad acquisire e per il tipo di metadati (informazioni aggiuntive) che è possibile raccogliere per ogni cellula. I tre metodi sono:

1) **piastre a micro-pozzetti**: In questo caso le cellule vengono isolate manualmente oppure attraverso uno smistamento automatico e vengono poste all'interno dei micro-pozzetti. Il vantaggio del metodo a piastre è la possibilità di controllare le cellule prima di sequenziarle, evitando ad esempio di sequenziare cellule morenti o cellule che sono rimaste attaccate (doublets). Lo svantaggio sono i tempi lunghi dovuti alla laboriosità del metodo (Picelli et al., 2014).

2) **microfluidi**: Questo metodo ha il grande vantaggio di automatizzare molti dei processi relativi alle piastre a micro-pozzetti, di conseguenza aumenta il livello di *throughput* (quantità di dati ottenuti) riducendo i costi e aumentando l'accuratezza dei risultati (Shapiro et al., 2013). Di contro i micro-pozzetti disponibili sono solo 96, quindi non è possibile processare più di 96 cellule alla volta; inoltre automatizzando il processo aumenta il rischio di doublets nei pozzetti.

3) **strumenti a gocce**: Questo strumento utilizza un dispositivo microfluidico per compartimentare "goccioline"(droplets) contenenti una singola cellula e una microsfera ricoperta di primer con dei codici a barre (barcode). Ogni primer ha al suo interno:

   (a) una sequenza oligo di 30 coppie di basi azotate per innescare la duplicazione dell'mRNA

   (b) un indice molecolare di 8 coppie di basi azotate per identificare in modo univoco ogni filamento di mRNA

   (c) un codice a barre di 12 basi azotate univoco per ciascuna cella

   (d) una sequenza universale identica su tutte le sfere.

   Il vantaggio di avere una sorta di identificativo per ogni cellula è la possibilità di riconoscere dopo il sequenziamento a quale cellula appartengono le *reads* sequenziate [2]. Inoltre l'indice molecolare unico (detto anche UMI) riconosce dopo l'amplificazione del DNA le sequenze con lo stesso barcode come frutto di una copia esatta dovuta all'amplificazione e non due copie indipendenti di RNA. Un ulteriore punto di forza di questo metodo è l'efficienza dal punto di vista economico dato

[2] https://www.illumina.com/science/sequencing-method-explorer/kits-and-arrays/drop-seq.html

il basso costo di sequenziamento per cellula che permette di sequenziare anche migliaia di cellule. Ciò nonostante spesso vengono sequenziate poche *reads* per cellula rendendo complicata la misurazione dei geni poco espressi.

Il passaggio successivo, dopo aver compiuto l'isolamento, è quello di preparare le librerie, che tecnicamente sono l'insieme dei trascritti che sono stati estratti e preparati per il sequenziamento. La creazione della librerie prevede alcuni passaggi tecnici come la retrotrascrizione da RNA a cDNA, l'amplificazione del DNA (specie quando non si ha tanto materiale a disposizione) e l'aggiunta di adattatori per permettere il sequenziamento. Infine l'ultimo step è il sequenziamento, in cui tutte le librerie precedentemente ottenute vengono raggruppate. Un aspetto molto importante del sequenziamento è la metodologia per la quantificazione dell'RNA, infatti ci sono due diversi tipi di quantificazione:

1. **quantificazione full-length**: In questo modo le *reads* vengono sequenziate in modo uniforme su tutto il gene, permettendo dunque di osservare tutta la struttura del gene espresso e di conseguenza individuare le espressioni delle diverse isoforme del gene e gli eventuali eventi di splicing alternativo.

2. **quantificazione con tag al 3' del gene**: in questo caso si mira a quantificare solamente l'estremità di un gene consentendo la possibilità di richiedere meno *reads* per gene e dunque sequenziare più cellule, ma non si ha la totale copertura delle possibili isoforme di quel gene.

Concluso il sequenziamento il passo finale è quello di allineare le *reads* al genoma, ovvero bisogna assegnare ogni *reads* a un gene o a un trascritto in maniera tale da avere un livello di espressione del gene o del trascritto. In

uno studio statistico quando si decide di fare un'analisi sulle differenze delle espressioni geniche e si decide di allineare sui geni si perde l'informazione relativa ai trascritti del gene, in compenso si evita il problema degli allineamenti multipli, ovvero la possibilità che alcune sequenze possano essere allineate su più di un trascritto. Un altro svantaggio di lavorare sui trascritti è che i dati di riferimento non sono più conteggi, di conseguenza avendo implementato un modello Poisson in questa tesi si è preferito compiere un'analisi sui geni. Le conte delle *reads* per gene rappresentano i conteggi finali che troveremo nel dataset di riferimento.

1.3 Dataset di riferimento

Il dataset su cui verrà applicato il modello proposto in questa tesi si riferisce all'analisi condotta dal laboratorio della Prof.ssa Lucia Peixoto presso il Dipartimento di Scienze Biomediche Elson S.Floyd College of Medecine dell'Università di Washington State. Le unità dell'analisi sono i topi domestici (*mus musculus*) e l'obiettivo dello studio è la comprensione di quali effetti nocivi possa avere la privazione del sonno sulle cellule situate nella corteccia prefrontale.

1.3.1 La corteccia prefrontale

La corteccia prefrontale rappresenta oltre il 30% del volume del cervello umano e risulta fondamentale in molte funzioni cognitive e comportamentali. In particolare le attività principali della corteccia sono la coordinazione dei pensieri e delle azioni in funzione dei propri obiettivi e tutte le funzioni legate all'apprendimento e alla capacità di ragionamento; pertanto danni irreversibili a questa parte del cervello possono portare a problemi di concentrazione,

di orientamento, di senso del giudizio, di risoluzione dei problemi con abilità ecc. La corteccia inoltre è anche legata agli stati emozionali e al compimento di funzioni intellettive astratte come prevedere le conseguenze di eventi o azioni. Per questo motivo sentimenti come ansia o frustrazioni vengono elaborati all'interno della corteccia e lesioni a questa area del cervello provocano difficoltà nella valutazione della relazioni temporali tra eventi (Martini et al., 2009).

La corteccia prefrontale inoltre ha una forte relazione con la durata e la qualità del sonno, beneficiando per molti aspetti da un ciclo sonno-veglia condotto con regolarità. Difatti tutte le funzioni esecutive precedentemente descritte sono regolate dalla corteccia prefrontale e possono essere compromesse dalla stanchezza indotta da uno stato prolungato di veglia. La tregua fornita dal sonno può consentire alla corteccia prefrontale di recuperare tutte le sue competenze funzionali, che diventano di importanza cruciale dopo il risveglio dell'essere umano (Muzur et al., 2002). A tal proposito sono stati condotti diversi studi che indagavano la relazione tra corteccia prefrontale e sonno, ad esempio quelli di Harrison e Horne i quali hanno esaminato i legami tra la corteccia prefrontale e la privazione del sonno in funzioni come: la fluidità verbale (studi effettuati negli anni 1988,1997,1998), la memoria temporale e le capacità decisionali (studio del 2000).

1.3.2 Mus Musculus: il topo domestico

L'esperimento descritto è stato condotto sui topi domestici, l'organismo vivente denominato scientificamente come *mus musculus*. L'importanza del *mus musculus* per lo studio dell'anatomia umana è giustificata dal fatto che i topi domestici vengono considerati sin dai primi anni del '900 come un organismo modello per gli uomini. In termini generali è possibile definire

organismo modello tutte quelle specie non umane che nel corso del tempo sono state ampiamente studiate per la comprensione di una vasta gamma di fenomeni biologici, con la speranza che le teorie e risultati derivati da questi studi siano applicabili a organismi più complessi, e solitamente all'organismo homo sapiens (Ankey e Leonelli, 2011). Date alcune caratteristiche come il breve tempo tempo di concepimento, cucciolate molto grandi, facilità di allevamento, varianti fenotipiche visibili e soprattutto per il fatto di essere dei mammiferi, il topo domestico è sempre servito come modello per studiare i fenotipi e le malattie tipici dell'essere umano (Phifer-Rixey e Nachman, 2015). In particolare i topi vengono studiati per comprendere in modo più dettagliato malattie legate al metabolismo, allo sviluppo, ai disordini neurologici, alle immunità ecc (Morse, 2007).

Ci sono vari aspetti genomici che legano il topo domestico all'uomo, in particolare i due organismi condividono praticamente lo stesso insieme di geni, infatti quasi ogni gene trovato in una specie è stato identificato in una forma volto vicina anche nell'altra specie. Inoltre dei 4000 geni che sono stati studiati meno di dieci si trovano in una specie ma non nell'altra ed è stato stimato che le porzioni di DNA che codificano per le proteine sono per l'85% identiche tra gli umani e i topi, con alcuni geni considerati identici fino al 99% mentre altri fino a solamente il 60%. Per questi motivi di natura genomica è dunque possibile compiere degli esperimenti che mimano alterazioni del DNA tipiche umane e applicarle nei topi per poterne studiare le conseguenze. (*fonte:National Human Genome Research Insitute* [3]).

Per quanto riguarda la corteccia prefrontale ci sono delle similitudini ma anche delle differenze tra gli umani e i topi. Una delle principale differenze è l'anatomia della corteccia prefrontale, nella Figura 1.1 si può notare come

[3]https://www.genome.gov/10001345/importance-of-mouse-genome

la corteccia umana presenti una parte frontale granulare, la quale è presente solamente negli organismi primati, mentre la corteccia del topo è completamente agranulare (Carlén, 2017). Ciononostante i processi funzionali del topo e dell'uomo all'interno della corteccia prefrontale possono essere equiparati, funzioni come: capacità decisionale, attenzione, capacità mnemonica sono comuni per entrambi gli organismi in quest'area del cervello. Data questa omologia delle funzioni all'interno della corteccia i risultati biologici dello studio sui topi può divenire di fondamentale importanza per gli essere umani (Chini e Hanganu-Opatz, 2020).

1.3.3 Disegno e obiettivo dell'esperimento

L'analisi condotta dalla Prof.ssa Peixoto si basa dunque sul comportamento dei topi in seguito alla privazione del sonno. Più precisamente i topi entrati a far parte dell'esperimento sono 6 topi maschi adulti di circa 8-10 settimane, a quest'ultimi sono state fatte delle operazioni chirurgiche per poter rilevare i dati di interesse: sono stati impiantati degli elettrodi elettroencefalografici (EEG) ed elettromiografici (EMG). Dopo avergli concesso il tempo necessario per il recupero dell'intervento (6 giorni) i topi sono stati preparati per la registrazione dell'attività cerebrale collegandoli ad un laccio leggero e lasciandogli altri 5 giorni per abituarsi al laccio. Successivamente sono stati registrati i segnali EEG e EMG per 24 ore a partire dall'insorgenza della luce (1^a ora). Il giorno seguente a 3 topi è stato privato il sonno per 5 ore manipolando l'insorgenza della luce e sono stati sacrificati alla 6^a ora senza consentire un sonno di recupero. Agli altri 3 topi invece, ovvero i controlli, è stato concesso un sonno regolare e in seguito anch'essi sono stati sacrificati. Infine dopo avere prelevato un campione biologico dalla corteccia di ogni topo, è stato possibile applicare il protocollo di sequenziamento a singola

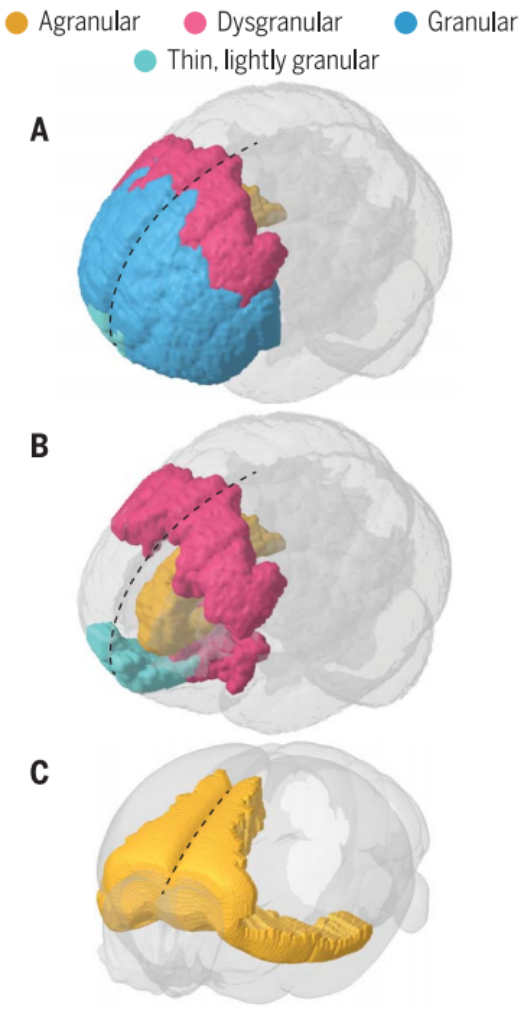

Figura 1.1: Tipi corticali nella corteccia prefrontale umana e murina

A e **B**: Vista laterale frontale inclinata del cervello umano. Illustrazione schematica dei quattro tipi corticali nella corteccia prefrontale.

C: Vista frontale inclinata del cervello del topo raffigurante la corteccia prefrontale agranulare. La linea nera tratteggiata indica la linea mediana sagittale. (*Fonte: Carlén, 2017*)

cellula, in particolare è stato scelto il metodo a gocce per l'isolamento delle cellule mentre per la quantificazione dell'espressione genica si è optato per la regola del tag al 3′ del gene.

L'obiettivo dell'esperimento è di identificare il gruppo di geni che presenta delle differenze significative in base alle due condizioni sperimentali: i tre topi controlli a cui è stato concesso il riposo e i tre topi "trattati"a cui è stato privato il sonno. In questo caso i classici metodi per studiare l'espressione differenziale dei geni nelle cellule (Capitolo 4) non tengono conto della struttura gerarchica dei dati, infatti sono stati pensati per analisi di tipo "bulk", dove non si ha nessuna informazione sulla cellula sequenziata. Nonostante questi metodi siano ampiamente usati nelle analisi differenziali anche per il sequenziamento single-cell in alcuni casi possono portare a distorsioni importanti sui falsi positivi (Lun e Marioni, 2017).

In questo caso inoltre viene introdotto un ulteriore livello di gerarchia corrispondente al topo da cui viene prelevato il campione biologico. Nel Capitolo 4 sarà possibile vedere come in quest'ultima circostanza i metodi classici fatichino maggiormente a identificare i geni differenzialmente espressi arrivando ad avere un alto numero di falsi negativi.

Capitolo 2

Modelli gerarchici Bayesiani

In questo Capitolo viene esposta la teoria statistica di riferimento per la costruzione e l'applicazione dei due modelli proposti in questa tesi. Il primo Paragrafo presenta una panoramica sui modelli lineari generalizzati (GLM) e nella seconda parte viene introdotto il caso specifico del modello Poisson appartenente alla famiglia dei GLM. In seguito viene spiegato il funzionamento di uno metodi più utilizzati per ricavare la distribuzione a posteriori di una variabile, ovvero il Markov Chain Monte Carlo (MCMC) e successivamente vengono evidenziati gli algoritmi necessari per applicare l'MCMC nel software utilizzato per l'analisi (Stan). Infine viene presentato un metodo alternativo all'MCMC, l'inferenza variazionale, che nonostante possa rivelarsi meno preciso nei risultati porta a una drastica diminuzione del tempo computazionale per applicare i modelli; nell'ultimo Paragrafo viene descritto l'algoritmo di differenziazione automatica fondamentale per implementare il modello tramite Stan.

2.1 Modelli lineari generalizzati

I modelli lineari generalizzati (GLM) vengono considerati delle estensioni dei classici modelli lineari, in particolare ci sono due assunzioni principali che vengono modificate nei GLM rispetto ai modelli lineari.
La prima ipotesi dei modelli lineari, che viene rispettata anche nei GLM, è l'indipendenza tra le realizzazioni della variabile risposta Y, mentre la seconda ipotesi ovvero l'assunzione di normalità della variabile risposta viene modificata nel caso dei GLM, infatti in questa circostanza si assume che Y si distribuisca come una variabile aleatoria con funzione di densità appartenente alla famiglia esponenziale (Salvan et al., 2020). La terza assunzione che cambia nei GLM è la funzione legame $g(\cdot)$ tra la media della variabile risposta $\mathbb{E}(Y_i)$ e il predittore lineare $\eta_i = x_i\beta$; nel caso dei modelli lineari $g(\cdot)$ corrisponde al legame identità con $g(\mu_i) = \mu_i$ mentre nei GLM la funzione legame può essere diversa e dipende dalla natura della variabile risposta.
Nei GLM dunque le funzioni di densità della variabile risposta devono sottostare a un modello parametrico che fa parte della classe delle famiglie di dispersione esponenziale. Possiamo identificare le famiglie di dispersione esponenziale esprimendo la densità di Y_i in questa forma:

$$p(y_i; \theta_i; \phi;) = \exp\{\frac{(\theta y_i - b(\theta_i))}{a_i(\phi)} + c(y_i, \phi)\}$$

Dove θ_i viene indicato come parametro naturale mentre ϕ è il parametro di dispersione; nel caso in cui $a_i(\phi) = 1$ e $c(y_i, \phi) = c(y_i)$ la formula si riconduce a una famiglia esponenziale naturale univariata, che presenta funzione di densità:

$$p(y_i, \theta_i) = \exp\{\theta y_i - b(\theta_i) + c(y_i)\}$$

Due proprietà molto importanti dei GLM sono la media e la varianza della variabile risposta Y_i, con funzione di densità appena descritta.

La media viene definita come:

$$\mathbb{E}(Y_i) = b'(\theta_i)$$

Mentre la varianza è:

$$Var(Y_i) = a_i(\phi)b''(\theta_i)$$

2.2 La distribuzione Poisson

La distribuzione Poisson fa parte delle famiglie di dispersione esponenziale ed essendo la distribuzione di riferimento del modello che verrà proposto è utile sapere le peculiarità di questa distribuzione. In questo caso $Y \sim Pois(\mu_i)$ con $\mu_i > 0$ e la funzione di probabilità definita sul supporto $S = \mathbb{N}$ corrisponde a:

$$p(y_i, \mu_i) = \exp\{y_i \log \mu_i - \mu_i\} \frac{1}{y_i!}$$

Come descritto nel Paragrafo precedente nel caso dei GLM la funzione legame non è più necessariamente la funzione identità e il legame tra la media e il predittore lineare può essere espresso in modo diverso, la funzione legame viene espressa nei GLM in questo modo:

$$g(\mathbb{E}(Y_i)) = g(\mu_i) = \eta_i = x_i\beta$$

Dove con $x_i = (x_{i,1}, ..., xi, p)$ indichiamo il vettore riga delle variabile esplicative per l'i-esima osservazione con $i = 1, ..., n$, mentre $\beta = (\beta_1, ..., \beta p)^T$ rappresenta il vettore di coefficienti della regressione.
Nel caso della distribuzione Poisson la funzione legame canonica è quella logaritmica con $\log(\mu_i) = \eta_i$. Possiamo ricapitolare quindi le assunzioni principali della distribuzione Poisson appartenente a una famiglia di dispersione esponenziale mediante le tre ipotesi principali:

1) $Y_1, ..., Y_n$ variabili casuali indipendenti

2) $\log(\mu_i) = \eta_i = x_i\beta = \beta_1 x_{i,1} + ... + \beta_p x_{i,p}$

3) $Y_i \sim Pois(\mu_i) \rightarrow Y_i \sim Pois(e^{x_i\beta})$

Infine per completare il quadro delle caratteristiche della distribuzione Poisson è utile citare l'espressione della media e della varianza, che in questo caso corrispondono:

$$Var(Y_i) = \mathbb{E}(Y_i) = e^{x_i\beta}$$

Una parte fondamentale dei modelli GLM sono le variabili da inserire nel predittore lineare e i parametri β a cui queste variabili sono associate. Uno dei casi più semplici è considerare una sola variabile esplicativa e due parametri: β_0 che corrisponde all'intercetta e β_1 che rappresenta il parametro associato alla variabile esplicativa di riferimento. In questo caso, per ogni osservazione y_i la relazione tra la media e il predittore lineare diviene:

$$\mu_i = \exp(\eta_i) = \exp(\beta_0 + \beta_1 x_i)$$

La variabile x_i può essere sia numerica che qualitativa, per il modello che verrà specificato in seguito risulta interessante l'interpretazione dei parametri considerando x_i come variabile qualitativa. Spesso x_i è una particolare tipologia di variabile qualitativa, ovvero una variabile dicotomica, la quale può assumere solamente 2 modalità che nella maggior parte dei casi vengono specificate con i valori 1 e 0. Le variabili dicotomiche possono essere utilizzate in molteplici casi, ad esempio: la presenza o l'assenza di una caratteristica, l'assegnazione o meno a un gruppo/trattamento, la divisione in due specifiche categorie etc.

Per il modello presentato nella tesi la variabile x_i si riferirà più specificatamente al caso in cui un soggetto viene sottoposto o meno a un trattamento,

nel primo caso $x_i = 1$, nel secondo caso $x_i = 0$. Risulta fondamentale capire l'interpretazione dei parametri quando viene introdotta una sola variabile dicotomica all'interno del modello; infatti in questo caso le realizzazioni della variabile risposta Y possono provenire dal gruppo uno (senza trattamento) o dal gruppo due (con trattamento). Definendo Y_1 e Y_2 le risposte provenienti dai due gruppi, il modello può essere esplicitato in quest'altra maniera:

$$Y_1 \sim Pois(\mu_1) \qquad log(\mu_1) = \beta_0$$

$$Y_2 \sim Pois(\mu_2) \qquad log(\mu_2) = \beta_0 + \beta_1$$

$$\mathbb{E}(Y_i) = \exp^{\beta_0 + \beta_1 x_i} \qquad x_i = \begin{cases} 0 & Y = Y_1 \\ 1 & Y = Y_2 \end{cases}$$

Questa forma di modellazione serve per dare un'interpretazione ai parametri β_0 e β_1, che si possono esplicitare dopo aver fatto le opportune sostituzioni:

$$\beta_0 = \log(\mu_1)$$

$$\beta_1 = \log(\mu_2) - \log(\mu_1) = \log\big(\frac{\mu_2}{\mu_1}\big)$$

Di conseguenza il parametro β_1 può essere interpretato come il log-rapporto tra la media delle osservazioni del gruppo due (soggetti trattati) e la media delle osservazioni del gruppo uno (soggetti non trattati). In sostanza β_1 esprime l'effetto del trattamento sulla popolazione in oggetto.

In un'ottica frequentista i parametri vengono stimati mediante le stime di massima verosimiglianza, e seguendo la teoria inferenziale è possibile determinare degli intervalli di confidenza e dei test d'ipotesi per i parametri. Ciò di cui la teoria frequentista non tiene conto è della possibile informazione a priori di alcuni parametri, un aspetto che può essere considerato solo lavorando in un ambiente bayesiano (Wakefield, 2013).

In un contesto biostatistico, dove molto spesso vengono condotte analisi sulla quantificazione dell'espressione genica nelle cellule, sono disponibili delle informazioni a priori che possono essere determinanti per la costruzione di un modello. Ad esempio una delle teorie principali nelle analisi genomiche è che la maggior parte dei geni non è differenzialmente espresso tra cellule diverse, questa informazione a priori può essere tradotta in linguaggio statistico ponendo la distribuzione a priori di β_1 con media 0.
La scelta che verrà fatta in seguito nel modello proposto è di assumere delle distribuzioni a priori normali per i coefficienti β_0 e β_1; lo svantaggio maggiore dell'attribuire questa tipologia di distribuzione è la perdita della proprietà di coniugatezza (Paragrafo 2.3).
In questo caso dunque si può fare uso di algoritmi ad hoc che permettono il campionamento della distribuzione a posteriori per un parametro con una distribuzione a priori non coniugata, questi metodi sono ampiamente usati attualmente e il principale algoritmo utilizzato è l'MCMC (Markov Chain MonteCarlo).

2.3 MCMC: metodi di campionamento per determinare le distribuzioni a posteriori

2.3.1 Teorema di Bayes

L'assunto principale della statistica bayesiana è che il parametro della distribuzione di riferimento viene trattato anch'esso come variabile casuale, di conseguenza il parametro assumerà una particolare distribuzione che nelle statistica bayesiana viene definita come "a priori". Il parametro, che possiamo indicare con θ, può assumere delle distribuzioni a priori differenti; a

seconda delle informazioni disponibili bisogna cercare di assegnare una distribuzione di θ in modo tale che venga assegnata una probabilità maggiore a quei valori che si ritengono più plausibili per θ. Considerando i dati osservati $y = (y_1, ..., y_n)$ e data la funzione di densità (o di probabilità) $f(y|\theta)$ possiamo definire la funzione di verosimiglianza in questo modo:

$$L(\theta) = f(y_1, ..., y_n) = \prod_{i=1}^{n} f(y_i|\theta)$$

Assumendo una specifica a priori per θ, definita come $\theta \sim \pi(\theta)$, il teorema di Bayes può essere formulato in questo modo:

$$\pi(\theta|y) = \frac{f(y|\theta)\pi(\theta)}{f(y)} = \frac{f(y|\theta)\pi(\theta)}{\int_\Theta f(y|\theta)\pi(\theta)d\theta}$$

In seguito la distribuzione a posteriori può essere approssimata tenendo conto solamente del termine al numeratore, dal momento che il denominatore può essere equiparato a una costante poiché non dipende dal parametro d'interesse. La posteriori può essere ridefinita come:

$$\pi(\theta|y) \propto f(y|\theta)\pi(\theta)$$

Il calcolo dell'integrale al denominatore viene omesso nei casi particolari in cui la distribuzione a posteriori $\pi(\theta|y)$ segue la stessa distribuzione dell'a priori $\pi(\theta)$, in questo caso ci si riferisce a distribuzioni coniugate per i parametri. Nel caso in cui non fosse possibile implementare una distribuzione coniugata e θ non si riferisse a una sola variabile casuale ma a un vettore di variabili casuali, e qualora quest'ultimo fosse di grandi dimensioni, allora sorge un problema computazionale per il fatto che l'integrale al denominatore è difficilmente calcolabile; in queste occasioni dunque risulta utile l'utilizzo dell'algoritmo Markov Chain Monte Carlo.

2.3.2 Markov Chain Monte Carlo

L'algoritmo MCMC pone le sue basi sull'utilizzo delle catene Markoviane. Una catena Markoviana è una sequenza di punti nello spazio parametrico che vengono generati a partire dalle transizioni Markoviane, queste ultime sono delle probabilità condizionali $Pr(\theta'|\theta)$, ovvero data una posizione nella catena θ si vuole esplorare una nuova posizione θ' nello spazio parametrico e tramite un'opportuna regola si decide se la nuova posizione θ' è più probabile rispetto la precedente. Ogni volta che la catena Markoviana raggiunge un nuovo stato nello spazio parametrico potrà generare un campione in base alla distribuzione della nuova posizione. Iterando questo processo per n volte si ottiene una catena Markoviana che esplora lo spazio parametrico. Il meccanismo di raggiungere ogni volta punti diversi dello spazio parametrico porta alla conclusione che, potenzialmente, da qualsiasi punto di partenza dello spazio parametrico si arrivi ad identificare una distribuzione per $\pi(\theta|Y)$ molto vicina alla reale distribuzione con un numero di iterazioni sufficientemente grande. Infatti avendo una sequenza di iterazioni infinita la catena Markoviana si concentrerà sulle posizioni dello spazio parametrico più probabili per θ, riportando una riproduzione fedele della sua distribuzione a posteriori.
Per capire meglio come funziona l'algoritmo MCMC si supponga innanzitutto di voler stimare una determinata quantità:

$$Q_\theta = \mathbb{E}(f(\theta)) = \int_\Theta f(\theta)\pi(\theta|y)d\theta$$

Considerando gli n campioni ottenuti tramite la catena Markoviana possiamo approssimare il valore atteso della quantità appena descritta in questo modo:

$$\hat{Q}_\theta = \frac{1}{N}\sum_{n=0}^{N} f(\theta_n)$$

La stima di $\hat{Q}_\theta$ converge in probabilità al vero valore secondo la legge debole dei grandi numeri:

$$\lim_{n\to\infty} \hat{Q}_\theta = \mathbb{E}_\Theta[f(\theta)]$$

Il processo di esplorazione della catena Markoviana, sotto condizioni ideali, può essere distinto in tre fasi differenti: durante la prima fase la stima di $\hat{Q}_\theta$ può essere molto distorta poiché i campioni provenienti dalla catena Markoviana non rispecchiano la vera distribuzione, nella seconda fase la catena si avvicina all'intervallo di valori dello spazio parametrico più verosimili e nella terza fase la precisione della stima MCMC aumenta poiché la catena Markoviana esplora solamente i valori dello spazio parametrico più verosimili (Betancourt, 2017). Quando la catena Markoviana entra nella cosiddetta terza fase allora soddisfa il teorema del limite centrale:

$$\hat{Q}_\theta^{MCMC} \sim \mathcal{N}(\mathbb{E}_\Theta[f(\theta)], \sigma_{MCMC})$$

Dove lo Standard Error per la catena Markoviana Monte Carlo è dato da:

$$\sigma_{MCMC} \equiv \sqrt{\frac{Var_\Theta[f(\theta)]}{ESS}}$$

Con ESS che si riferisce alla dimensione effettiva del campione e viene inteso come il numero di campioni corrispondenti alla vera distribuzione di $\hat{Q}_\theta^{MCMC}$, ESS viene così specificato:

$$ESS = \frac{N}{1 + 2\sum_{l=1}^{\inf} \rho_l}$$

dove ρ_l è l'autocorrelazione di due valori distanti l. Nella pratica verranno scelti i campioni solamente dopo una fase di warm-up della catena Markoviana, infatti solamente dopo un certo numero di iterazioni le stime per $\hat{Q}_\theta$ diventano accurate, mentre terminare prematuramente la catena Markoviana

può portare a delle distorsioni non indifferenti; un metodo empirico utilizzato per la diagnosi del campionamento MCMC è l' $\hat{R}$ che confronta l'andamento di diverse catene Markoviane inizializzate in punti diversi, se le catene sono sufficientemente mischiate tra di loro l'$\hat{R}$ tenderà ad avvicinarsi a 1 (Gelman et al., 2014)

2.3.3 L'algoritmo Metropolis-hastings

Tramite questo algoritmo è possibile descrivere il processo di formazione della transizione Markoviana che darà forma alla catena Markoviana; in particolare l'algoritmo Metropolis-Hasting utilizzando un'apposita funzione decide se accettare un nuovo vettore θ' all'interno dello spazio parametrico nella catena Markoviana o se rifiutare il vettore proposto (proprietà di reversibilità) (Hastings, 1970). Il nuovo stato esplorato della catena, θ', viene proposto a partire da una funzione $q(\theta'|\theta)$; l'algoritmo originale di Monte Carlo che viene tutt'ora utilizzato adopera come funzione proposta una normale:

$$q(\theta'|\theta) = \mathcal{N}(\theta'|\theta, \Sigma)$$

In seguito il criterio per stabilire l'accettazione della funzione proposta si base sul confronto tra un valore u generato da un uniforme (tra 0 e 1) e il valore dell'indicatore α così definito:

$$\alpha = min\left\{1, \frac{\pi(\theta'|y)q(\theta|\theta')}{\pi(\theta|y)q(\theta'|\theta)}\right\}$$

Dove il primo termine al numeratore indica la distribuzione a posteriori del nuovo modello proposto, mentre il primo termine del denominatore rappresenta la posteriori del modello corrente. Poi, se u è minore di α allora accettiamo il nuovo vettore di parametri proposto altrimenti lo rifiutiamo (Gallagher et al, 2009).

2.3.4 Hamiltonian Monte Carlo

Quando il vettore di variabili casuali θ ha una dimensionalità troppo elevata l'algoritmo di Metropolis-Hastings non è più sufficiente, poiché anche aumentando la soglia di accettabilità della funzione proposta l'algoritmo tenderà a fare "salti"nello spazio parametrico troppo piccoli e i valori prodotti saranno vicini alle code della distribuzione a posteriori effettiva del vettore θ (Betancourt, 2017); si può quindi affermare che l'algoritmo soffra della cosiddetta "maledizione della dimensionalità"(Azzalini e Scarpa, 2012). É quindi necessario un algoritmo che permetta dei "salti"più ampi nello spazio parametrico e che riesca ad esplorare delle regioni dello spazio dove si trova la zona a più alta densità, e in particolare che riesca a definire geometricamente la forma della zona con densità maggiore. L'Hamiltonian Monte Carlo prende in prestito concetti dalla fisica per utilizzarli in ambito matematico, in particolare utilizza il concetto fisico del "momentum"o quantità di moto: dato un punto materiale di massa m con velocità v, la quantità di moto p viene definita come: $p = mv$ (*fonte: Wikipedia* [1]). Considerando in questo caso θ come l'espressione della distribuzione a posteriori $\pi(\theta|y)$ (per non appesantire la notazione), viene assegnato per ogni dimensione del vettore θ_k un "momentum"ausiliare p_k, $\theta_k \rightarrow (\theta_k, p_k)$ e viene definita la probabilità congiunta come:

$$\pi(\theta, p) = \pi(p|\theta)\pi(\theta)$$

Questa densità congiunta può essere descritta utilizzando una funzione Hamiltoniana, ovvero:

$$\pi(\theta, p) = e^{-H(\theta,p)}$$

[1] https://it.wikipedia.org/wiki/Quantità_di_moto

Dopo aver specificato la densità condizionale di p dato θ possiamo ridefinire la formula esplicitando la funzione Hamiltoniana:

$$\begin{aligned} H(\theta, p) &= -\log \pi(\theta, p) \\ &= -\log \pi(p|\theta) - \log \pi(\theta) \\ &= \mathrm{T}(p|\theta) + V(\theta) \end{aligned}$$

Riprendendo l'analogia con l'ambiente fisico $\mathrm{T}(p|\theta)$ corrisponde al momento ausiliare e può essere definito come energia cinetica mentre il termine $V(\theta)$ che è completamente determinato dalla densità target che si vuole identificare, rappresenta l'energia potenziale (Betancourt e Girolami, 2015).

$$\textit{energia cinetica}: \quad \mathrm{T}(p|\theta) \equiv -\log \pi(p|\theta)$$

$$\textit{energia potenziale}: \quad V(\theta) = -\log \pi(\theta)$$

Il campionamento del momento ausiliare è il primo step per le transizioni Markoviane, con $p \sim \pi(p|\theta)$, in seguito poiché la funzione Hamiltoniana riesce a stimare la geometria della zona a più alta densità, tramite un sistema di equazione Hamiltoniane è possibile esplorare regioni ad alta densità:

$$d\theta = +\frac{\partial H}{\partial p} = +\frac{\partial \mathrm{T}}{\partial p}$$

$$\frac{dp}{dt} = -\frac{\partial H}{\partial \theta} = \frac{\partial \mathrm{T}}{\theta} - \frac{\partial V}{\partial \theta}$$

Tramite il gradiente, definito da $\frac{\partial V}{\partial \theta}$ è possibile creare un campo vettoriale allineato alla zona con densità più alta, dove per campo vettoriale si intende l'assegnazione di una direzione ad ogni punto dello spazio parametrico, se le direzioni sono allineate verso la zona a più alta densità allora possono guidare ogni transizione Markoviana verso quella determinata zona.

2.4 Inferenza Variazionale

L'inferenza variazionale è un metodo per approssimare funzioni di densità mediante ottimizzazione, l'idea di base è proporre una famiglia di densità e in seguito trovare il membro di quella famiglia più vicino alla densità obiettivo. Il grande vantaggio di utilizzare l'inferenza variazionale è la maggiore velocità di questo metodo rispetto al classico MCMC soprattutto quando le analisi vengono fatte su grandi dataset; nonostante le buone performance di questo approccio le sue proprietà non sono ancora state ben studiate e la sua implementazione nel software Stan è considerata sperimentale. Infatti, a differenza dell'MCMC che garantisce la produzione di un numero di campioni adeguato a ricoprire la funzione di densità obiettivo in modo esatto, l'inferenza variazionale garantisce solamente una densità vicina a quella reale. Di conseguenza l'inferenza variazionale presenta dei vantaggi e degli svantaggi e a seconda della velocità dei risultati o della precisione dell'inferenza potrà dipendere la scelta decisionale per uno o l'altro metodo (Blei et al., 2017).

2.4.1 Evidence Lower Bound: ELBO

L'algoritmo d'inferenza variazionale nasce con l'idea di trovare una funzione di densità il più possibile vicina a quella reale; data una famiglia di densità M, ogni densità $m(\theta)$ è una potenziale candidata come approssimazione dell'esatta funzione di densità condizionale $\pi(\theta|y)$. L'obiettivo è trovare la densità che minimizza la divergenza di Kullback-Leibler (KL) mediante la risoluzione di un problema di ottimizzazione:

$$m^*(\theta) = \underset{m(\theta)\in M}{\arg\min}\ KL(m(\theta)||\pi(\theta|y))$$

La complessità della computazione, che cerca di trovare la funzione di densità $m^*(\theta)$ che ottimizza la funzione obiettivo, dipende dalla famiglia di densità M

che viene scelta. Tuttavia la funzione obiettivo non è risolvibile poiché al suo interno contiene $\log \pi(y)$, infatti esplicitando la divergenza KL otteniamo:

$$\arg\min_{m(\theta)\in M} \; KL(m(\theta)||\pi(\theta|y)) = \mathbb{E}[\log m(\theta)] - \mathbb{E}[\log \pi(\theta, y)] + \log \pi(y)$$

É necessario quindi trovare una funzione obiettivo differente che permetta il calcolo di tutti i suoi componenti. La funzione obbiettivo implementata nell'inferenza variazionale corrisponde all' "Evidence Lower Bound (ELBO)", massimizzare l'ELBO equivale a minimizzare la divergenza KL e la funzione viene definita come:

$$ELBO(m) = \mathbb{E}[\log \pi(\theta, y)] - \mathbb{E}[\log m(\theta)]$$

Per capire il comportamento di questa funzione è utile riscriverla esplicitando il ruolo dei suoi componenti:

$$\begin{aligned} ELBO(m) &= \mathbb{E}\big[\log \pi(\theta)\big] + \mathbb{E}\big[\log \pi(y|\theta)\big] - \mathbb{E}\big[\log m(\theta)\big] \\ &= \mathbb{E}\big[\log \pi(y|\theta)\big] - KL\big(m(\theta) \;||\; \pi(\theta)\big) \end{aligned}$$

In sostanza la funzione ELBO(m) rispecchia il consueto trade-off del comportamento della posteriori, che viene influenzato sia dalla verosimiglianza che dalla a priori, infatti il primo termine rappresenta l'attesa verosimiglianza e il secondo termine spinge la densità variazionale ad essere vicina alla densità a priori.

Per completare l'algoritmo variazionale occorre infine decidere una regola per la scelta della famiglia di densità M, infatti la complessità della famiglia è decisiva per il costo computazionale dell'algoritmo. Spesso viene scelta la famiglia variazionale "mean-field" dove ogni dimensione del vettore θ è governata da un proprio fattore variazionale m_k con l'assunzione che i parametri

siano mutuamente indipendenti, di conseguenza un membro della famiglia "mean-field" viene descritto tramite:

$$m(\theta) = \prod_{k=1}^{K} m_k(\theta_k)$$

Infine il processo di ottimizzazione può essere portato a termine tramite diversi algoritmi, tra i più utilizzati vi è CAVI (Bishop, 2006). Un altro algoritmo utile per l'inferenza variazionale è la differenziazione automatica, che viene utilizzato in software come Stan (Kucukelbir et al., 2015).
Stan (insieme ad R) è il principale strumento utilizzato per l'analisi di dati in ambito bayesiano ed è un software che permette l'implementazione di modelli bayesiani, l'analisi di dati e le analisi predittive in diversi campi scientifici come biologia, ingegneria, fisica ecc. Il modello descritto nel Capitolo 4 viene implementato tramite Stan col linguaggio $C++$ e viene utilizzata l'interfaccia grafica su R mediante il pacchetto Rstan [2].

2.4.2 Inferenza Variazionale in Stan

Un aspetto che non è stato specificato nel Paragrafo precedente è la forma parametrica dei fattori variazionali individuali $m_k(\theta_k)$; la peculiarità dell'inferenza variazionale in Stan, e dunque dell'algoritmo di differenziazione automatica, è la scelta di applicare una forma parametrica Gaussiana ai fattori variazionali. Prima di applicare la distribuzione Gaussiana il software Stan compie un passaggio molto importante, ovvero trasforma il vettore di variabili latenti θ in modo tale che abbiano un supporto nello spazio reale $\mathbb{R}^K$ dove K rappresenta il numero totale di dimensioni del vettore θ .
Tramite la funzione:

$$\mathbb{T} : supp(\pi(\theta)) \to \mathbb{R}^K$$

[2] https://cran.r-project.org/web/packages/rstan/rstan.pdf

è possibile definire le variabili trasformate $\zeta = \mathbb{T}(\theta)$; in seguito si può applicare una forma parametrica Gaussiana utilizzando la nota famiglia "mean-field"per i fattori variazionali.

$$m(\zeta; \phi) = \mathcal{N}(\zeta; \mu, \sigma) = \prod_{k=1}^{K} \mathcal{N}(\zeta_k; \mu_k, \sigma_k)$$

L'inversa della trasformazione $\mathbb{T}^{-1}$ permette di ritornare al supporto delle variabili latenti, di conseguenza l'approssimazione variazionale nello spazio delle variabili latenti originali sarà $\mathcal{N}(\mathbb{T}^{-1}(\zeta); \mu, \sigma^2 | det\ \mathcal{J}_{T^{-1}}(\zeta)|)$.
Un ulteriore riparametrizzazione avviene per σ, avendo un supporto su $\mathbb{R}^+$ viene riparametrizzato tramite $\omega = \log(\sigma)$ in modo tale da avere un supporto sulle coordinate reali. Infine per utilizzare la differenziazione automatica è necessario fare un'ulteriore trasformazione denominata standardizzazione elittica. Tramite questa trasformazione i fattori variazionali invece di assumere una distribuzione Gaussiana seguono la Gaussiana standardizzata, possiamo definirla tramite $\eta = S_{\mu,\omega}(\eta) = diag(exp(\omega))^{-1}(\eta - \mu)$, e quindi la densità variazionale sarà:

$$m(\boldsymbol{\eta}) = \mathcal{N}(\boldsymbol{\eta}|\mathbf{0}, \boldsymbol{I}) = \prod_{k=1}^{K} \mathcal{N}(\eta_k, |0, 1)$$

La standardizzazione ridefinisce la funzione $ELBO$ ed è possibile riscriverla utilizzando la nuova parametrizzazione e omettendo i termini costanti (per approfondimenti sulla derivazione della formula seguente a partire dalla formula precedente di ELBO si rimanda a Kucukelbir et al., 2017):

$$\mu^*, \omega^* = \arg\max_{\mu,\omega} \mathbb{E}_{\mathcal{N}(\eta;0,I)} \left[\log \pi(Y, T^{-1}(S^{-1}_{\mu,\omega}(\eta))) + \log |det \mathcal{J}_{T^{-1}}(S^{-1}_{\mu,\omega}(\eta))| \right] + \sum_{k=1}^{K} \omega_k$$

Dove all'interno della formula $|det(\cdot)|$ rappresenta il valore assoluto del determinante. A questo punto è possibile applicare il suddetto algoritmo di differenziazione automatica:

Algoritmo di differenziazione automatica:

Input: Dataset $Y = (y_1, ...y_n)$ e modello $\pi(Y, \theta)$

Si inizializza $i = 0$ e si sceglie una dimensione del passo $\rho^{(i)}$

Si inizializza $\mu^{(0)} = \mathbf{0}$ e $\omega^{(0)} = \mathbf{0}$

Finché il cambiamento di $ELBO$ è al di sotto di una certa soglia **esegui**:

1. Crea D campioni da $\eta_d \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ da una Gaussiana standardizzata multivariata
2. Inverti la standardizzazione $\zeta_d = diag(\exp(\omega^{(i)}))\eta_d + \mu^{(i)}$
3. Approssima $\nabla_\mu ELBO$ e $\nabla_\omega ELBO$ utilizzando le integrazioni descritte nelle equazioni 1 e 2
4. Modifica $\mu^{(i+1)} \leftarrow \mu^{(i)} + \rho^{(i)} \nabla_\mu ELBO$ e $\omega^{(i+1)} \leftarrow \omega^{(i)} + \rho^{(i)} \nabla_\omega ELBO$
5. Incrementa di 1 l'iterazione

Fine

Ritorna $\mu^* \leftarrow \mu^{(i)}$ e $\omega^* \leftarrow \omega^{(i)}$

Le formule dei gradienti da inserire all'interno dell'algoritmo sono:

1)

$$\nabla_\mu ELBO = \mathbb{E}_{\mathcal{N}(\eta)}[\nabla_\theta \log p(\boldsymbol{Y}, \theta) \nabla_\zeta \mathbb{T}^{-1}(\zeta) + \nabla_\zeta \log |det \mathcal{J}_{\mathbb{T}^{-1}}(\zeta)|]$$

2)

$$\nabla_{\omega_k} ELBO = \mathbb{E}_{\mathcal{N}(\eta_k)}[(\nabla_{\theta_k} \log p(\boldsymbol{Y}, \theta) \nabla_{\zeta_k} \mathbb{T}^{-1}(\zeta) + \nabla_{\zeta_k} \log |det \mathcal{J}_{\mathbb{T}^{-1}}(\zeta)|) \eta_k \exp(\omega_k)] + 1$$

L'idea è quella di inserire i gradienti all'interno delle formule delle previsioni e trovare delle nuove previsioni utilizzando il concetto delle catene Markoviane.

Capitolo 3

Specificazione del modello

In questo Capitolo verranno descritti i due modelli che sono stati implementati in questa tesi. Il primo modello, presentato nel Paragrafo 3.1 è una particolare tipologia di modello bayesiano e può essere utile in contesti di dati a singola cellula quando si sequenziano cellule relative allo stesso campione (uomo, topo ecc.). Anche il secondo modello, descritto nel Paragrafo 3.2, è stato pensato in un'ottica bayesiana ma con l'aggiunta di un' ulteriore gerarchia relativa al campione da cui vengono sequenziate le cellule. Questo modello risulta utile quando viene applicato il sequenziamento a singola cellula nel caso in cui il materiale biologico venga prelevato da più campioni. Questo modello è una generalizzazione del primo, infatti riprende le stesse distribuzioni a priori assegnate nel modello bayesiano ma introduce un nuovo elemento di gerarchia. Nel Capitolo 4 è possibile vedere le performance dei due modelli, in particolare il modello bayesiano standard funziona bene e ottiene dei risultati leggermente migliori dei metodi comunemente usati per l'analisi differenziale dei geni. Il modello gerarchico bayesiano invece sembra superare nettamente le performance di questi metodi al costo di un tempo computazionale molto più lungo.

3.1 Modello bayesiano

Il modello è stato ipotizzato con l'idea di riuscire a individuare i geni differentemente espressi tra due particolari tipologie di cellule, ad esempio ci si può riferire a delle cellule che ricevono o meno un determinato trattamento. Di conseguenza nel modello viene inserita una sola variabile esplicativa che può corrispondere a seconda dell'analisi all'effetto del trattamento o all'effetto di gruppo (gruppo cellulare); dunque saranno presenti solamente due coefficienti. Come indicato nel Paragrafo 2.2 il primo coefficiente (β_0) rappresenta il logaritmo della media della popolazione non trattata mentre il secondo coefficiente (β_1) indica la log-differenza tra la media della popolazione trattata e quella non trattata. Il modello presentato aggiunge un ulteriore parametro α_k per catturare la differenza del numero di trascritti tra le cellule, provocata dalle diverse profondità di sequenziamento, con k che rappresenta l'indice di cellula (Risso et al., 2013). L'approccio utilizza la statistica bayesiana assegnando delle distribuzioni a priori per tutti e tre i parametri, inoltre fa uso del metodo MCMC con l'algoritmo Hamiltoniano per campionare la distribuzione a posteriori dei parametri (Paragrafo 2.3). I conteggi vengono modellati come se provenissero da una distribuzione Poisson: definiamo $Y_{j,k}$ il conteggio del gene j relativo alla cellula k, con $j = 1...J$ numero di geni e $k = 1...K$ numero di cellule. Esplicitiamo di seguito le assunzioni principali del modello:

$$Y_{j,k} \sim Pois(\mu_{j,k})$$

$$\mu_{j,k} = \exp(\beta_{0,j} + x_k\beta_{1,j} + \alpha_k)$$

Con la funzione legame che può essere riscritta come:

$$\log(E[Y_{j,k}|x, \alpha_k, \beta_{0,j}, \beta_{1,j}]) = \beta_{0,j} + x_k\beta_{1,j} + \alpha_k$$

La variabile esplicativa x è dicotomica e ha valori uguali a 0 quando ci si riferisce a una cellula non trattata e valori uguali a 1 quando la cellula ha subito il trattamento. Il modello prosegue con la specificazione delle distribuzioni a priori:

$$\beta_{0,j} \sim \mathcal{N}(\mu_0,\, \gamma_0\,)$$

$$\beta_{1,j} \sim \mathcal{N}(\mu_1,\, \exp(\delta)\,)$$

$$\alpha_k \sim \mathcal{N}(\mu_\alpha,\, \sigma_\alpha\,)$$

Con γ_0 e δ che assumono a loro volta una distribuzione a priori, le quali vengono specificate nelle formule seguenti dove $\mathcal{G}(\cdot)$ indica una distribuzione Gamma e $\mathcal{B}(\cdot)$ denota una distribuzione Bernoulliana:

$$\frac{1}{\gamma_0} \sim \mathcal{G}(\,\psi_0\,,\, \phi_0\,)$$

$$\delta = \lambda\, T_1 + (1-\lambda)\, T_2$$

$$\lambda \sim \mathcal{B}(p_0)$$

$$T_1 \sim \mathcal{N}(t_1, \sigma_{t,1}) \qquad T_2 \sim \mathcal{N}(t_2, \sigma_{t,2})$$

Successivamente vengono scelti empiricamente gli iperparametri del modello, in alcuni casi valutando le performance del modello con diversi valori per gli iperparametri, in altri casi quest'ultimi vengono scelti in base all'esperienza a priori dell'inferenza bayesiana in campo biostatistico. Per gli iperparametri μ_0 e μ_1 viene assegnato il valore 0. La media a priori di β_0 viene ipotizzata uguale a 0 rifacendosi a esempi simili presenti in letteratura. Per β_1 si ipotizza che la maggior parte dei geni non sia differenzialmente espresso tra le cellule trattate e quelle non trattate e quindi viene assegnato il valore 0 alla media a priori. Il parametro di varianza di β_0 ha una distribuzione a priori che corrisponde a una gamma inversa; assegnando agli iperparametri della gamma inversa ψ_0 e ϕ_0 lo stesso valore $0,01$ è possibile garantire una

posteriori per γ_0 abbastanza alta, in modo tale che i valori per β_0 possano essere sufficientemente alti per quei geni molto espressi e sufficientemente bassi per quelli quasi mai espressi. Per definire la distribuzione a priori della varianza di β_1 si è seguito l'approccio presentato da Risso et al. (2013). La distribuzione a priori di δ corrisponde a una mistura, dove in base al valore di p_0 verrà dato più peso a T_1 o a T_2. Entrambe le due normali hanno l'iperparametro di media uguale a 0 ($t_1 = 0$ e $t_2 = 0$) ma la prima normale ha una varianza uguale a 0.1 mentre la seconda ha una varianza uguale a 1 e di conseguenza è una normale standard ($\sigma_{t,1} = 0.1$ e $\sigma_{t,2} = 1$). Sono stati provati diversi valori di p_0 e alla fine si è osservato empiricamente che il valore migliore corrisponde a 0.1, di conseguenza la a priori assegna molto più peso alla normale standard. Infine per la a priori di α_k si è scelto di adottare una normale standard, seguendo un approccio molto comune in letteratura in modelli simili a questo.

3.2 Modello gerarchico bayesiano

Il modello come precedentemente descritto è stato generalizzato con l'idea di tenere conto dell'informazione a priori di ogni soggetto o animale da cui vengono sequenziate le cellule. Viene quindi introdotta una gerarchia molto importante: il parametro relativo alla media dell'intercetta (μ_0) assume un'altra distribuzione a priori, che sarà diversa in base al campione selezionato. Il modello quindi presenta tre differenti indici: j relativi ai geni, i per i campioni e k che rappresentano le cellule sequenziate. Anche in questo caso modelliamo i valori dei conteggi come se provenissero da una distribuzione Poisson: definendo $Y_{j,i,k}$ come il conteggio del gene j relativo al campione i e alla cellula k, poi supponiamo questa particolare conformazione per il

modello:

$$Y_{j,i,k} \sim Pois(\mu_{j,i,k})$$

$$\mu_{j,i,k} = \exp(\beta_{0,j,i} + x_k\beta_{1,j} + \alpha_k)$$

La funzione legame può essere specificata anche in quest'altra maniera:

$$\log(E[Y_{j,i,k}|x, \alpha_k, \beta_{0,j,i}, \beta_{1,j}] = \beta_{0,j,i} + x_k\beta_{1,j} + \alpha_k$$

Dove analogamente col modello precedente x_k è un vettore creato a partire dalla matrice del disegno, e assume valore 1 o 0 a seconda che le cellule siano state sottoposte o meno al trattamento. In questo caso il parametro $\beta_{0,j,i}$ rappresenta il logaritmo dell'espressione media del gene j nel campione i mentre $\beta_{1,j}$ rappresenta nuovamente la differenza logaritmica dell'espressione media del gene j tra le cellule trattate e non trattate. I due parametri β_0 e β_1 hanno delle distribuzioni a priori normali e per β_0 viene introdotta l'ulteriore gerarchia per tenere conto dell'informazione a priori del campione da cui proviene la cellula.

$$\beta_{0,j,i} \sim \mathcal{N}(\mu_{0,j,i},\, \gamma\,)$$
$$\frac{1}{\gamma} \sim \mathcal{G}(\,\psi_0\,,\, \phi_0\,)$$

$$\mu_{0,j,i} \sim \mathcal{N}(0, 1) \qquad con\ i = 1...I$$

Anche per questo modello sono stati scelti gli stessi iperparametri con $\phi_0 = 0.01$ e $\psi_0 = 0.01$; per $\mu_{0,j,i}$ viene attribuita una normale standard per ogni differente campione presente nell'analisi. Empiricamente si è osservato che con questa distribuzione a priori il modello otteneva degli ottimi risultati e quindi si è deciso di non cambiare la distribuzione, posto il fatto che qualora si avesse qualche informazione ulteriore sui campioni selezionati si potrebbe

modificare in maniera consona l'iperparametro di media della distribuzione $\mu_{0,j,i}$ per ogni i. Per $\beta_{1,j}$ viene riproposta la stessa distribuzione a priori del modello bayesiano e pure il parametro di varianza assume la stessa distribuzione a priori vista nel Paragrafo 3.1. Rimangono indifferenti le considerazioni sugli iperparametri fatte per il modello bayesiano, alla stessa maniera rimane uguale la distribuzione a priori per α_k.

3.3 Test statistico per l'espressione differenziale dei geni

Dopo aver specificato il modello è necessario adottare una statistica test che riesca ad individuare i geni differenzialmente espressi (DE); i geni vengono selezionati come DE tramite una procedura basata sul Bayesian False Discovery Rate (BFDR), che risulta più adeguata nel caso di a priori informative le quali portano ad avere delle posteriori di $\beta_{1,j}$ più concentrate attorno allo 0 (Van De Wiel et al., 2013). Inoltre mentre l'FDR (False Discovery Rate) viene utilizzato in un contesto frequentista in cui vale il principio del campionamento ripetuto, il BFDR è stato implementato appositamente per ambiti bayesiani (Ventrucci et al., 2011). Per prima cosa occorre trovare una statistica test adeguata per il contesto di riferimento e in seguito applicare il BFDR alla statistica test. Utilizziamo a tal proposito l'informazione derivata dalla distribuzione a posteriori dei parametri $\beta_{1,j}$ per ogni gene j. Le distribuzioni a posteriori saranno centrate in 0 se $\beta_{1,j}$ si riferisce a un gene non differenzialmente espresso, mentre saranno centrate su un valore leggermente maggiore o minore di 0 qualora $\beta_{1,j}$ si riferisse a un gene DE.

Definiamo con m ogni singolo valore della distribuzione a posteriori di $\beta_{1,j}$

con $m = 1...M$ dove M rappresenta il totale dei valori ottenuti col campionamento della posteriori tramite l'MCMC. Quello che risulta interessante conoscere è la percentuale di valori che si discostano in modo considerevole da 0. Dato un valore T leggermente maggiore di 0 la statistica test di partenza può essere definita come un'approssimazione del Local False Discovery Rate(lfdr) (Efron, 2007):

$$lfdr_j = \frac{\sum_{m=0}^{M} \mathbf{1}_{(-T<=m_i<=T)}}{M}$$

Con **1** che rappresenta la funzione caratteristica e risulta 1 quando il valore m_i è compreso tra le due soglie $-T$ e T o 0 nel caso contrario.
Sostanzialmente $lfdr$ tenderà ad assumere dei valori distanti da 0 per i geni non DE, mentre sarà vicino allo 0 per geni differenzialmente espressi, poiché in quel caso la maggior parte dei valori sarà al di fuori delle due soglie e la funzione caratteristica sarà uguale a 0 per tutti quei valori. Per applicare il BFDR alla statistica test sul software R bisogna eseguire determinati passaggi che verranno di seguito descritti.
Innanzitutto creiamo un vettore t composto da 100 valori equidistanti di 0.01 da 0.01a 1:

$$t = \begin{pmatrix} 0.01 \\ 0.02 \\ 0.03 \\ 0.04 \\ . \\ . \\ . \\ 0.99 \\ 1 \end{pmatrix}$$

Il passo successivo è creare una matrice D con numero di righe uguale al numero di geni e numero di colonne uguale a 100, ovvero una colonna per ogni distinto valore di t. Per ogni colonna verrà confrontato il valore di $lfdr$ col valore di t_i. Per ogni cella $[j, i]$ se il valore di $lfdr_j$ è minore di t_i allora verrà assegnato il valore 1, ad indicare che per la soglia t_i il gene j è DE, se invece $lfdr_j$ è maggiore di t_i allora sarà assegnato il valore 0 e dunque per la soglia t_i il gene j viene considerato come non differenzialmente espresso.
Di conseguenza la matrice D indica per ogni colonna il numero di geni per cui rifiuteremo l'ipotesi nulla dato il valore della colonna t_i. La struttura della matrice quindi è quella di avere per ogni riga il valore 1 sulla prima colonna in cui $lfdr_j$ sarà minore del valore t_i corrispettivo alla colonna i, poi assumerà all'interno della riga valori 0 per tutte le colonne antecedenti alla colonna i, e valore 1 per tutte le colonne conseguenti.
La costruzione di questa matrice risulta necessaria per calcolare i diversi valori di BFDR per ogni soglia t_i. Infatti il BFDR verrà calcolato per ogni diverso valore di t in questa maniera (Ventrucci et al., 2011):

$$BFDR_{t_i} = \frac{\sum_{j=1}^{J} lfdr_j * D_{j,i}}{\sum_{j=1}^{J} D_{j,i}}$$

Per concludere la procedura sul $BFDR$ occorre individuare il più grande valore t_i che consente di avere $BFDR_{t_i} <= 0.05$. Dopo aver individuato il giusto t_i, che chiameremo t^*, allora identifichiamo i geni DE nel caso in cui $lfdr_j < t^*$.
A questo punto, tenendo fissata la soglia del 5% sul BFDR, l'unica cosa che rimane da stimare è il valore di T della formula di partenza da utilizzare per il calcolo del $lfdr$. Utilizzando il metodo MCMC non è possibile sapere a priori quale sarà il corretto valore T da scegliere per individuare i geni DE. Quando si procede con le simulazioni invece, conoscendo in partenza quali

sono i reali geni differenzialmente espressi e dati i valori a posteriori di β_1, è ragionevole sapere per quali valori di T il modello riesce ad identificare in maniera più efficace i geni DE. Si è quindi scelto di utilizzare il metodo della curva ROC per analizzare le prestazioni del modello in base a diversi valori di T. Il valore di T ottimale sarà quello che garantisce il miglior trade-off tra falsi positivi e falsi negativi (questi ultimi saranno descritti in maniera dettagliata nel Capitolo 4).

3.3.1 Test statistico per l'inferenza variazionale

Per i risultati inferenziali ottenuti con l'inferenza variazionale non è stato possibile implementare la stessa tipologia di test descritta nel precedente Paragrafo. Infatti, come si avrà modo di vedere nel prossimo Capitolo, le stime dei parametri tendono ad essere distorte, in particolare la distribuzione a posteriori per il parametro $\beta_{1,j}$ tende ad essere traslata verso la destra di 0 (si veda il Capitolo 4 per ulteriori approfondimenti). Per questo motivo il precedente test che valutava quanto fossero vicini a 0 i valori della posteriori di $\beta_{1,j}$ perde di significato in questo caso dal momento che lo 0 non è più il centro della distribuzione.

Per sopperire a questo problema si è scelto di lavorare sui quantili della distribuzione a posteriori ottenuta tramite l'inferenza variazionale. Più precisamente per ogni distribuzione a posteriori di un gene viene scelta la media della distribuzione, dunque la media di tutti i valori campionati tramite l'inferenza variazionale. Successivamente viene creato l'istogramma relativo alle medie della posteriori per ogni gene. Gli istogrammi presentati nel prossimo Capitolo hanno tutti una distribuzione simile al comportamento di una normale, con molti valori vicini alla media della distribuzione e pochi valori sulle code. Infine vengono selezionati come differenzialmente espressi quei geni di

cui la media a posteriori si trova nelle code dell'istogramma utilizzando i quantili. Ad esempio prendendo il quantile del 5%, i geni DE sono quelli col valore di media a posteriori maggiore del 95esimo percentile o minore del quinto percentile, e dunque otterremo un totale di 10% di geni DE sul totale dei geni. Come sarà possibile osservare nel prossimo Capitolo, l'utilizzo di questo test statistico porta a risultati altrettanto soddisfacenti del test statistico basato sul BFDR. Anche per questo test verranno scelti diverse soglie dei quantili e per ogni soglia verrà testata la capacità predittiva del modello.

Capitolo 4

Risultati

Nel Paragrafo 4.1 di questo Capitolo vengono descritti i parametri con cui sono state implementate le simulazioni, queste ultime vengono effettuate per testare il buon funzionamento dei due modelli.
Successivamente nei Paragrafi 4.2 e 4.3 vengono esposti i risultati delle simulazioni dei due modelli: bayesiano e bayesiano gerarchico. Nell'ultima parte invece vengono mostrati i risultati relativi ai dati reali, cioè quelli rilevati dallo studio sui topi domestici. Durante il periodo di preparazione della tesi sono stati specificati diversi modelli e sono state fatte numerose simulazioni per analizzarne il comportamento; alla fine dopo aver identificato i modelli con le prestazioni migliori (i due riportati nella tesi) si è deciso di preparare un piano di simulazione che comprendesse diverse situazioni plausibili nella realtà. Per simulare i dati è stato utilizzato il software Bioconductor della piattaforma R e in particolare ci si è serviti del pacchetto Splatter [1]. Questo pacchetto consente di simulare i dati tenendo conto di diversi parametri, pertanto le simulazioni sono state fatte variando di volta in volta il valore di alcuni parametri. I valori simulati poi vengono confrontati coi risultati dei

[1] https://www.bioconductor.org/packages/release/bioc/html/splatter.html

pacchetti EdgeR, Deseq2 e Limma; anche questi tre pacchetti possono essere scaricati dal software Bioconductor e sono quelli più scaricati in assoluto per l'analisi dell'espressione differenziale dei geni [2]. Possiamo quindi definire gli esiti ottenuti mediante questi metodi dei *gold standard*. Per le simulazioni proposte saranno analizzate le prestazioni dei due modelli sia utilizzando l'algoritmo MCMC sia facendo uso dell'inferenza variazionale. Inoltre per tutte le simulazioni effettuate è stato calcolato il tempo computazionale necessario ai due algoritmi per ottenere i risultati.

4.1 Simulazioni con Splatter

Per comprendere l'andamento dei due modelli sono state condotte molteplici simulazioni che ricalcassero la struttura di dataset reali derivanti da esperimenti a singola cellula. Per il modello bayesiano i dati simulati si riferiscono a un solo campione mentre per il modello bayesiano gerarchico i dataset generati contengono l'informazione relativa ai diversi campioni da cui provengono i dati.

Per ognuno dei due modelli vengono riportati i risultati di 4 simulazioni, effettuate variando di volta in volta il valore di un parametro (nella nota 1 è disponibile il sito internet con tutta la documentazione del pacchetto Splatter, si rimanda al sito per ottenere informazioni sui parametri di default che vengono utilizzati per le simulazioni di Splatter).

Oltre al valore di alcuni parametri variabili tra le simulazioni si è optato per modificare a priori due parametri di default in modo analogo per tutte le simulazioni. Il primo valore che è stato modificato è il parametro di gruppo, che può essere inteso anche come trattamento. Splatter infatti ipotizza

[2] https://www.bioconductor.org/packages/release/BiocViews.html#___StatisticalMethod

che tutte le cellule provengano da un unico gruppo, e quindi da un unico trattamento, mentre per lo scopo della tesi è necessario che le cellule derivino da due distinti trattamenti. Per questo motivo sono stati simulati due diversi gruppi (o trattamenti) con l'ipotesi che ogni cellula abbia il 50% di probabilità di appartenere a uno dei due. Operando diverse simulazioni è stato notato che con un numero di cellule non troppo alto (20/30 cellule) Splatter le divide tra i due gruppi con una probabilità molto distante dal 50% (ad esempio con 20 cellule ne assegna 15 al gruppo 1 e 5 al gruppo 2); questa divisione distorta rimane identica anche cambiando il seme casuale da cui vengono simulati i dati. Invece quando viene simulato un numero più elevato (maggiore di 30) la probabilità del 50% viene rispecchiata. In ogni caso per le simulazioni proposte, molte delle quali composte da 20 cellule totali, lo sbilanciamento delle cellule appartenenti a uno dei due gruppi non risulta essere un problema. Il secondo parametro su cui viene applicata una modifica in tutte le simulazioni è quello relativo alla vicinanza tra i gruppi in termini di espressione genica. Prima di spiegare in che modo è stata attuata l'operazione di rendere i due gruppi più dissimili tra loro è importante chiarire il sistema con cui Splatter li crea e il modo con cui identifica i geni come differenzialmente espressi.

Il procedimento per creare insiemi di cellule differenti avviene tramite la simulazione di un espressione genica differenziale tra un gruppo e una cellula base fittizia. Di conseguenza quando si simulano due gruppi questi si differenziano maggiormente se viene aumentata la differenziazione tra il gruppo e la cellula base di partenza. Il grado di differenziazione dipende dai fattori differenziali, che Spatter produce per ogni gene. Quando il gene non è differenzialmente espresso (DE) questo fattore è uguale a 1 mentre quando è DE il fattore è maggiore di 1 se il gene è sovraespresso in quel gruppo di cellule

rispetto la cellula base, oppure minore di 1 se risulta sottoespresso. Conseguentemente per rendere un gruppo maggiormente diverso dalla cellula base ci sono due possibilità: aumentare il numero di geni DE nel gruppo rispetto alla cellula base o incrementare in valore assoluto il fattore differenziale per i geni DE. Mentre la prima opzione viene modificata in modo differente nelle simulazioni, la seconda alternativa cambia in principio in tutti i casi.
Il parametro di riferimento in Splatter per modificare i fattori differenziali per i geni DE è *"de.facLoc"*, più il valore di quest'ultimo è vicino a 0 minore sarà il fattore differenziale in valore assoluto, ottenendo quindi dei fattori differenziali per i geni DE molto vicini a 1. Il default per questo parametro è 0.1, per le simulazioni è stato mutato aumentandolo a 1.2, assicurandosi dei fattori differenziali per i geni DE distanti da 1; in questo modo sia i modelli proposti sia i metodi classici hanno delle performance molto migliori nell'identificare i reali geni DE. Lasciando il parametro col valore assunto di default invece tutti i metodi hanno delle prestazioni molto più basse, un risultato prevedibile dal momento che i gruppi risultano essere molto più vicini.
Ma utilizzando queste simulazioni quali sono i reali geni differenzialmente espressi tra i due gruppi?
In questo caso infatti Splatter non calcola i geni DE tra i due gruppi, ma mostra quelli differenzialmente espressi tra un gruppo e la cellula base; dunque occorrerà compiere delle operazioni preliminari per poter identificare i geni DE tra i due gruppi. Riprendendo le notazioni del Capitolo 2.2 possiamo scrivere il coefficiente di gruppo per ogni gene in questa maniera:

$$\beta_{1,j} = log(\mu_{2,j}) - log(\mu_{1,j})$$

$$\beta_{1,j} = log(\frac{\mu_{2,j}}{\mu_{1,j}})$$

Dove $\mu_{1,j}$ e $\mu_{2,j}$ possono essere riscritte in funzione di $\mu_{0,j}$, che rappresenta l'espressione media del gene j della cellula base fittizia, e in funzione di $FD_{1,j}$ e $FD_{2,j}$ che rappresentano i fattori differenziali tra il gene j nelle cellule dei due gruppi e il gene j nella cellula base fittizia.

$$\mu_{1,j} = \mu_{0,j} \cdot FD_{1,j}$$

$$\mu_{2,j} = \mu_{0,j} \cdot FD_{2,j}$$

In seguito per entrambe lo formule possiamo esplicitare $\mu_{2,j}$:

$$\mu_{2,j} = \mu_{1,j} \cdot e^{\beta_{1,j}}$$

$$\mu_{2,j} = \mu_{1,j} \cdot \frac{FD_{2,j}}{FD_{1,j}}$$

Risolvendo le due equazioni possiamo trovare i veri valori che dovrebbe assumere il coefficiente di gruppo $\beta_{1,j}$ per ogni gene j:

$$\beta_{1,j} = \log(\frac{FD_{2,j}}{FD_{1,j}})$$

Il vettore di coefficienti teorico β_1 servirà poi nelle simulazioni per giudicare la capacità predittiva di un dato modello, all'interno del vettore i geni non differenzialmente espressi assumono valore 0 mentre quelli DE hanno un valore maggiore o minore di 0. Infatti quando un gene non è differenzialmente espresso tra le cellule di un gruppo e la cellula base allora il suo fattore differenziale è uguale a 1, in questo modo per i geni che non sono differenzialmente espressi né per il gruppo 1 né per il gruppo 2 il valore di $\beta_{1,j}$ sarà uguale a 0 poiché diventa uguale a logaritmo di 1.

A questo punto è possibile mostrare le simulazioni compiute. In primis vengono esposte nella tabella seguente le simulazioni per dati non gerarchici dove i parametri che variano di volta in volta sono: il numero di geni, il numero di cellule o la percentuale di geni DE.

simulazioni	numero di geni	numero di cellule	% geni DE
1	300	20	≈ 10
2	500	20	≈ 10
3	500	20	≈ 20
4	300	40	≈ 20

Tabella 4.1: Parametri delle simulazioni con dati non gerarchici

La percentuale totale di geni differenzialmente espressi è data dalla somma delle frazioni di geni DE tra un gruppo e la cellula base. Ad esempio se si fissa per ogni gruppo un 10% di geni DE rispetto alla cellula base, quelli differenzialmente espressi tra i due gruppi saranno circa il 20%. In realtà la percentuale risulta probabilmente leggermente inferiore poiché ci saranno dei geni che sono differenzialmente espressi sia tra il gruppo 1 e la cellula base, sia tra il gruppo 2 e la cellula base; questa situazione è stata riscontrata nelle simulazioni. Per tutte le simulazioni la bontà del modello verrà valutata analizzando la capacità di identificare nel miglior modo possibile i geni DE.

4.2 Risultati modello bayesiano

Per analizzare i risultati del modello bayesiano implementato con l'utilizzo dell'MCMC occorre indagarne il comportamento al variare della soglia T scelta per il modello (Capitolo 3.3). In base al cambiamento del valore soglia verranno confrontati i veri positivi e i falsi positivi, e successivamente verrà creata una curva ROC per capire la capacità predittiva del modello al variare della soglia.

Per poter creare una curva ROC occorrono due vettori contenenti i valori 0 o 1, dove lo 0 indica un gene che non è differenzialmente espresso mentre

l'1 rappresenta un gene DE. Il vettore dei reali geni DE ovviamente rimane identico per ogni soglia, mentre per ogni valore di T avremo un diverso vettore dei geni previsti DE.

In questa situazione l' "evento positivo "e dunque l'evento di interesse è considerato l'identificazione di un gene come DE, quindi i veri positivi (VP) sono tutti i geni realmente differenzialmente espressi che il modello riesce a riconoscere come tali (Azzalini e Scarpa, 2012). Al contrario i veri negativi (VN) sono tutti i geni non DE identificati nello stesso modo anche dal modello. Di conseguenza i falsi positivi (FP) sono i geni realmente non differenzialmente espressi ma che il modello identifica come DE; mentre i falsi negativi (FN) sono i geni realmente differenzialmente espressi ma considerati non DE dal modello.

Il grafico della curva ROC presenta sull'asse delle ascisse il tasso di falsi positivi (TFP) mentre sull'asse delle ordinate si trova il tasso di veri positivi (TVP). I due tassi vengono calcolati in questa maniera:

$$TVP = \frac{VP}{VP+FN}$$
$$TFP = \frac{FP}{FP+VN}$$

In particolare il TVP rappresenta la sensibilità del modello e quindi la capacità da parte di quest'ultimo di riconoscere i veri geni DE. Mentre il TFP equivale a $1 - specificità$, dove la specificità è un indicatore dell'abilità del modello di riconoscere i veri geni non DE. Infine per ogni valore della soglia viene posizionato un punto sul grafico corrispondente alle coordinate relative al TFP e al TVP e in seguito vengono uniti tutti i punti per formare la curva ROC. Un indicatore dell'abilità del modello di identificare correttamente i geni DE è l'area sotto la curva ROC, infatti più la curva è orientata verso la parte in alto a sinistra del grafico maggiore sarà la capacità predittiva del

modello e maggiore sarà l'area sotto la curva.

Prima di presentare i risultati della curva ROC è interessante analizzare il tempo computazionale necessario ai modelli per ottenere dei risultati, come descritto nel Capitolo 2 il metodo MCMC ha un tempo computazionale molto più elevato rispetto all'inferenza variazionale, ma a vantaggio dell'MCMC c'è l'accuratezza dei risultati. L'aspetto più evidente della Tabella 4.2 è come l'inferenza variazionale riduca drasticamente il processo computazionale lungo tipico dall'MCMC, ma vedremo negli istogrammi presentati che le distorsioni generate dall'inferenza variazionale non possono essere trascurate. Analizzando le tempistiche dell'MCMC invece risalta l'aumento del tempo computazionale incrementando i geni da 300 (simulazione 1) a 500 (simulazioni 2 e 3), con i tempi che diventano poco più del doppio. Nella simulazione 4, dove le cellule vengono ampliate passando da 20 a 40 ma sono tenuti i 300 geni di partenza il tempo computazionale tende ad assestarsi intorno all'ora e mezza. Quindi il tempo computazionale dell'MCMC sembra essere molto più sensibile ad un aumento di cellule rispetto all'incremento di geni. Per quanto riguarda le tempistiche dell'inferenza variazionale rispecchiano quello ottenute dall'MCMC ma in una scala molto più piccola.

simulazioni	**tempo MCMC**	**tempo VI**
1	42 min	34 sec
2	1 h 36 min	70 sec
3	1 h 27 min	60 sec
4	1 h 38 min	87 sec

Tabella 4.2: Tempi computazionali del modello bayesiano per MCMC e VI

Negli istogrammi nelle Figure 4.1, 4.2, 4.3 e 4.4 presentati nelle pagine seguenti è possibile osservare il comportamento delle medie delle distribuzioni

a posteriori di β_1 per ogni gene. Più precisamente per i parametri $\beta_{1,j}$ le medie a posteriori vengono calcolate tramite la media di tutti i valori delle 1000 iterazioni finali dell'MCMC (ottenute dopo le 1000 iniziali del warm-up).
Gli istogrammi nelle Figure 4.5, 4.6, 4.7, 4.8 si riferiscono invece alle medie a posteriori di β_1 per ogni gene delle simulazioni effettuate con l'inferenza variazionale. In questo caso gli istogrammi non sono più centrati in 0 ma presentano una distorsione evidente, il centro della distribuzione delle medie a posteriori infatti è sempre un valore a destra di 1. In letteratura non sono stati trovati delle spiegazioni esaustive inerenti a questa particolare tipologia di distorsione, ma è stato constatato che l'inferenza variazionale può avere dei problemi nel replicare correttamente la distribuzione a posteriori. Un indicatore per giudicare se l'approssimazione sta funzionando bene è il k di Pareto, in linea teorica il valore di k dovrebbe essere inferiore a 1 per ottenere delle distribuzioni a posteriori sufficientemente buone (Yao et al., 2018). Nella pratica per le simulazioni compiute il valore di k è sempre stato maggiore di 1 e probabilmente proprio per questo motivo gli istogrammi presentano una distribuzione distorta. Ciononostante è stato osservato empiricamente che i geni DE continuano a trovarsi sulle code della distribuzione, pertanto è stato possibile utilizzare anche i risultati dell'inferenza variazionale avvalendosi della tecnica dei quantili per scovare i geni DE.

A questo punto è interessante capire il funzionamento dei due diversi algoritmi nell'identificare correttamente i geni DE. L'MCMC e l'inferenza variazionale vengono quindi confrontati coi metodi più noti e più applicati in letteratura per l'analisi di geni differenzialmente espressi, ovvero: EdgeR, Deseq2 e Limma. Questi tre metodi sono dei pacchetti disponibili sul sito di Bioconductor e i primi due sono metodi basati sulla distribuzione binomiale

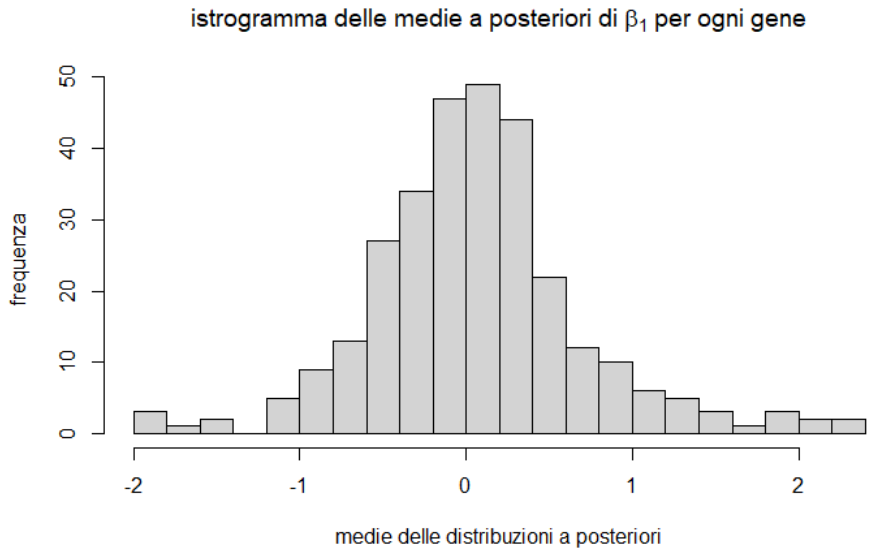

Figura 4.1: istogramma delle medie a posteriori per la simulazione 1 (MCMC)

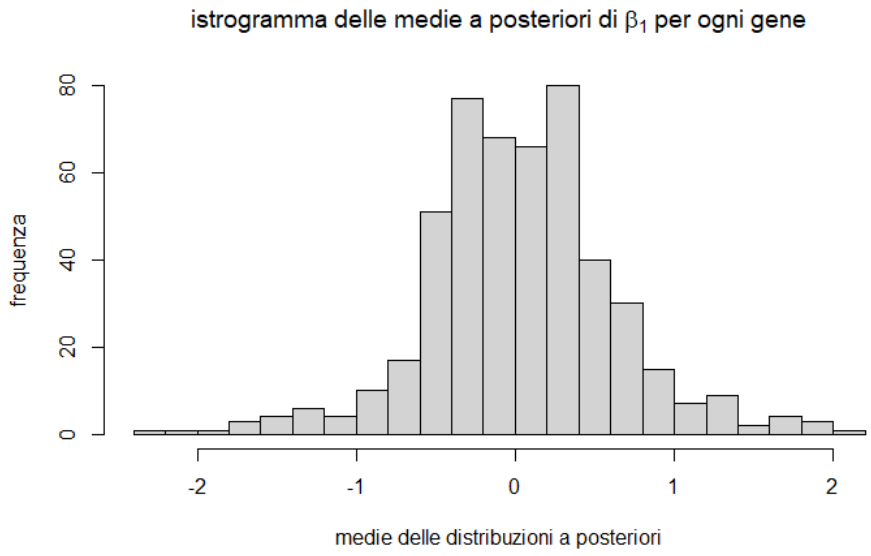

Figura 4.2: istogramma delle medie a posteriori per la simulazione 2 (MCMC)

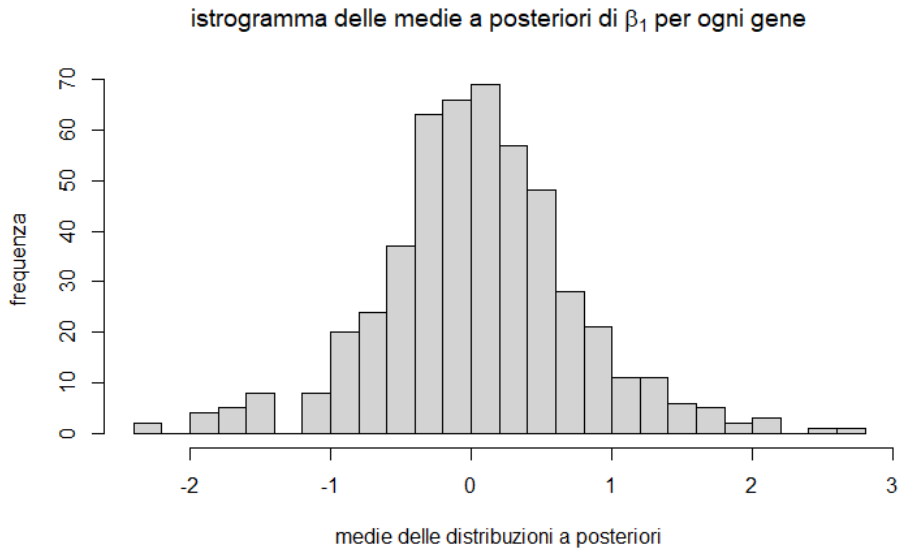

Figura 4.3: istogramma delle medie a posteriori per la simulazione 3 (MCMC)

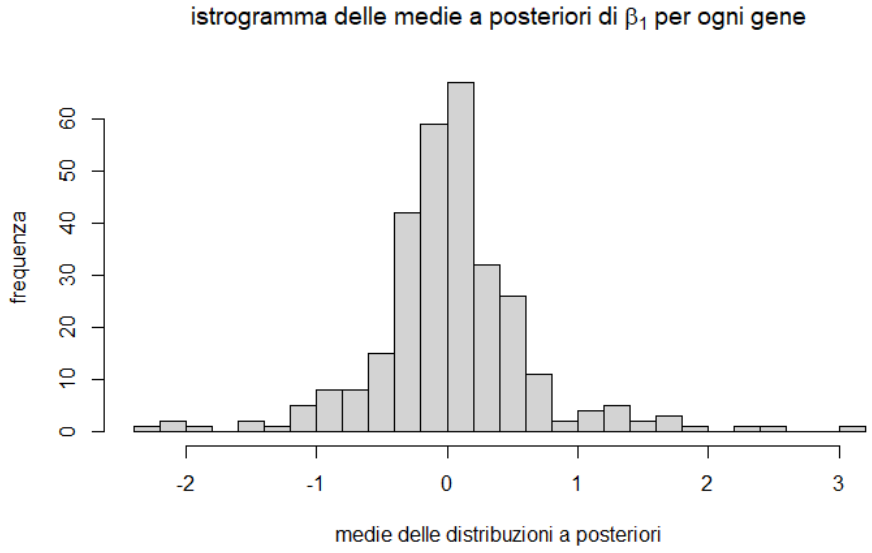

Figura 4.4: istogramma delle medie a posteriori per la simulazione 4 (MCMC)

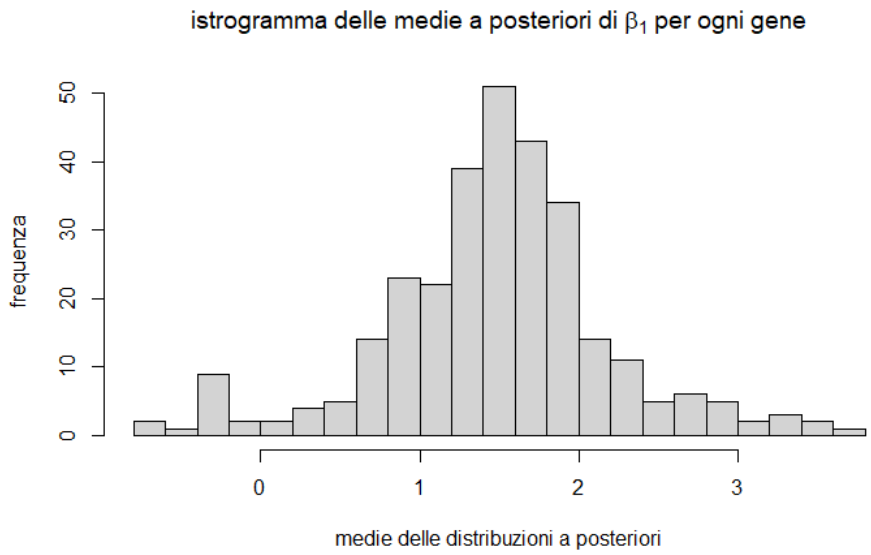

Figura 4.5: istogramma delle medie a posteriori per la simulazione 1 (VI)

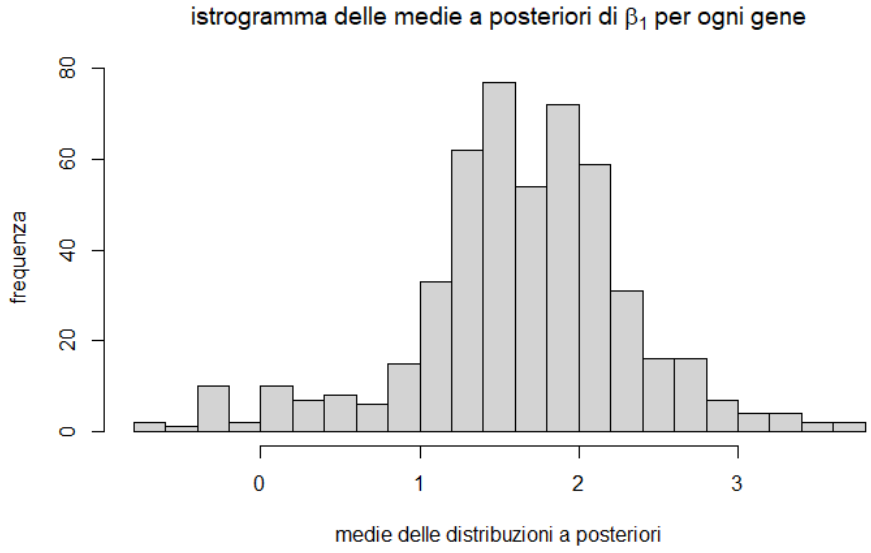

Figura 4.6: istogramma delle medie a posteriori per la simulazione 2 (VI)

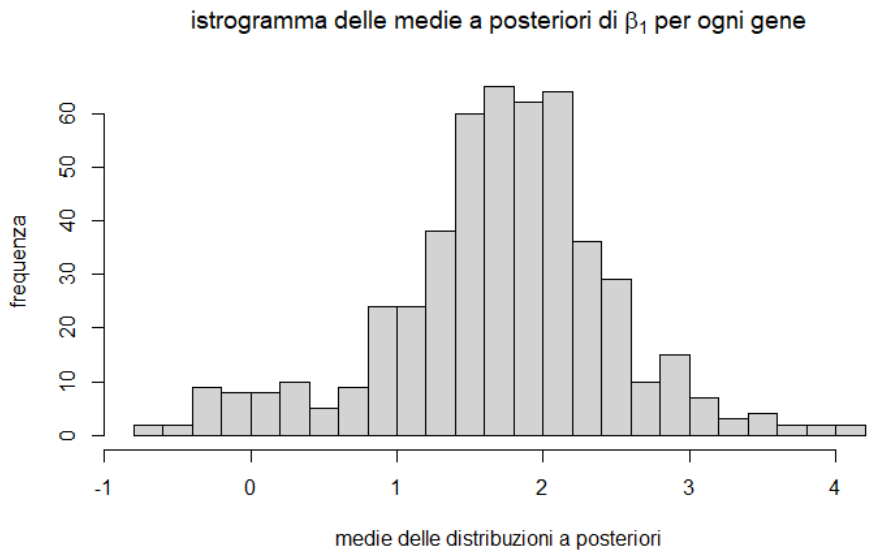

Figura 4.7: istogramma delle medie a posteriori per la simulazione 3 (VI)

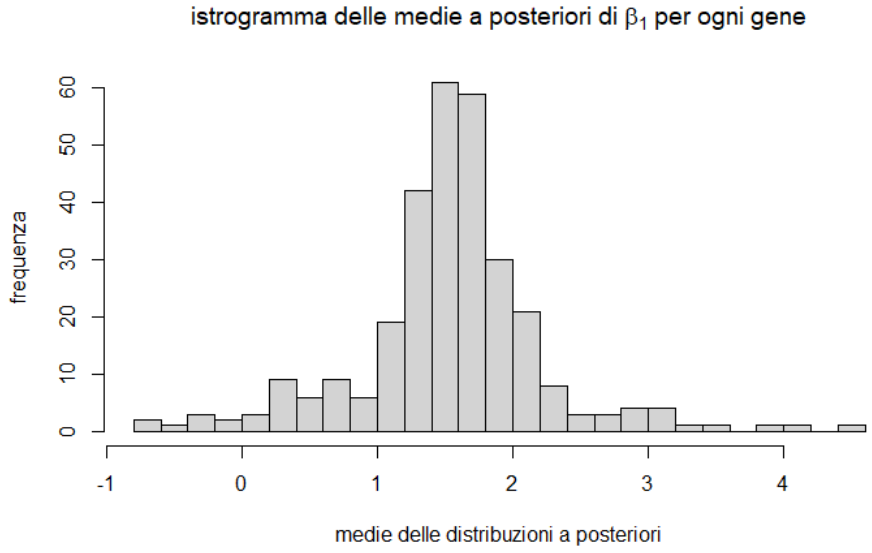

Figura 4.8: istogramma delle medie a posteriori per la simulazione 4 (VI)

negativa [3] [4], mentre Limma si basa su un modello lineare con pesi [5]. Questi approcci utilizzano la procedura del False Discovery Rate (Benjamini e Hochberg, 1995) per decretare un certo numero di geni come differenzialmente espressi. Infatti il False Discovery Rate (FDR) controlla il valore atteso dei falsi positivi e viene definito come la frazione attesa di falsi positivi tra le ipotesi che sono state dichiarate significative; riprendendo i concetti dei FP e VP spiegati precedentemente e definendo P come il totale delle ipotesi dichiarate significative e dunque corrispondenti a "eventi positivi", il FDR può essere definito in questa maniera:

$$FDR = \mathbb{E}[\frac{FP}{FP+VP}] = \mathbb{E}[\frac{FP}{P}]$$

Solitamente poi viene scelta una soglia del 5% per il FDR, in modo tale che tra le ipotesi dichiarate significative in media solamente il 5% saranno falsi positivi. Questo procedura è essenziale in un contesto biostatistico dove esiste un problema di molteplicità dei test, infatti si vanno a testare J ipotesi indipendenti col livello di significatività α (con J numero totale di geni). Con $\alpha = 0.05$ se il numero di geni è alto cresce il numero di falsi positivi e diventa il 5% delle ipotesi testate, di conseguenza su un dataset con 10000 geni 500 di questi circa potrebbero essere falsi positivi.

Per la creazione delle curve ROC si è scelto di applicare un ciclo sul valore del FDR per questi metodi. In particolare si è deciso di vedere le performance dei 3 metodi variando la percentuale del FDR dall'1% al 40%, facendo crescere di volta in volta il valore del FDR di un'unità percentuale, permettendo dunque 40 distinti punti nella curva ROC; pertanto la percentuale di falsi positivi tra le ipotesi significative potrà crescere fino al 40%. Nonostan-

[3] http://bioconductor.org/packages/release/bioc/html/edgeR.html

[4] https://bioconductor.org/packages/release/bioc/html/DESeq2.html

[5] https://bioconductor.org/packages/release/bioc/html/limma.html

te nella pratica comune difficilmente si sceglie un valore diverso dal 5% in questo caso la decisione di attribuire un intervallo di valori così ampio per il FDR è giustificata da una resa grafica della curva ROC più comprensibile, e soprattutto permette a questi metodi di avere un valore dell'AUC più elevato (area sotto la curva ROC) per poterli confrontare in modo adeguato con l'MCMC e l'inferenza variazionale.

Per questi ultimi invece sono state condotte due diverse tipologie di test statistico (Paragrafo 3.3 e Paragrafo 3.3.1) e di conseguenza anche la scelta su come costruire l'intervallo di valori per la curva ROC è stata differente. Come illustrato precedentemente per l'MCMC, tenendo fissata la soglia sul BFDR del 5%, la soglia fondamentale per identificare i geni DE deriva dal valore di T, infatti se T è piccolo avremo molti più geni differenzialmente espressi e quindi una quantità maggiore di falsi positivi, al crescere di T invece diminuiscono i geni DE pertanto può aumentare la probabilità di falsi negativi poiché il modello perde in sensibilità. Per esaminare diverse possibilità relative alla soglia si è stabilito di compiere un ciclo con valori di T da 0.3 fino a 1.3, incrementando i valori di un centesimo alla volta e avendo in totale 101 punti a formare la curva ROC.

Infine per l'ultimo metodo, ovvero l'inferenza variazionale, la soglia decisiva risulta essere quella inerente alla scelta del quantile della distribuzione delle medie a posteriori dei parametri $\beta_{1,j}$. Sono stati scelti come geni DE quelli presenti nelle code: come primo valore sono stati selezionati l'1% di quelli sulla coda di sinistra e l'1% dei geni sulla coda destra; poi aumentando le percentuali di un'unità alla volta il ciclo è proseguito fino all'identificazione del 20% di geni DE sulla coda sinistra e ugualmente sulla coda destra. Quindi sono state scelte 20 soglie differenti sui quantili, partendo da un 2% totale di geni differenzialmente espressi (soglia per la quale risulta un problema coi

falsi negativi) fino a un 40% (valore per il quale ci sono troppi falsi positivi). Per la simulazione numero 1 si è deciso di mostrare sia le curve ROC dei vari modelli formate dall'unione dei punti creati tramite i vari cicli (Figura 4.9), sia le curve ROC con le linee prolungate fino ai due punti nel grafico (0,0) e (1,1), in modo tale da poter calcolare più agevolmente l'indicatore AUC e avere una resa grafica migliore (Figura 4.10). Per le restanti simulazioni (comprese quelle del prossimo Capitolo) vengono mostrati solamente i grafici con le linee prolungate considerati più opportuni per fini interpretativi. I quattro grafici delle curve ROC (Figure 4.10, 4.11, 4.12, 4.13) mettono in evidenza le ottime performance del modello bayesiano implementato con l'MCMC, infatti l'AUC di questo modello è il più alto in tutte e quattro le simulazioni. I metodi EdgeR e Deseq2, assumendo entrambi la stessa distribuzione binomiale negativa per il modello, presentano delle curve che sembrano comportarsi in maniera molto simile. Ambedue raggiungono inizialmente gli stessi livelli di sensibilità conseguiti dall'MCMC ma quest'ultimo riesce ad avere per determinate soglie dei livelli di sensibilità maggiori a patto di un numero di falsi positivi leggermente più alto. Rispetto a Deseq2 però le prestazioni di EdgeR sembrano essere leggermente superiori, soprattutto nella Figura 4.12 relativa alla simulazione 3, ossia quella creata con più geni. In questo caso spicca la capacità di EdgeR di raggiungere una sensibilità più alta verso le ultime soglie disponibili, risultando in un AUC lievemente più elevato. Per quanto concerne il metodo Limma, quest'ultimo sembra essere in generale quello con le performance peggiori in tutte le simulazioni. Infine il comportamento dell'inferenza variazionale risulta essere un po' anomalo rispetto gli altri metodi. In generale è quello che presenta più problematiche riguardo i falsi positivi, infatti fin dai primi valori soglia sui quantili ottiene dei falsi positivi, ma aumentando la percentuale di geni DE sulle co-

de l'inferenza variazionale riesce a raggiungere i livelli di sensibilità toccati dall'MCMC e raramente anche a superarli (Figura 4.10) ottenendo nel complesso dei valori di AUC molto simili a quelli di EdgeR e Deseq2. La tabella Tabella 4.3 mostra un quadro complessivo dei valori degli AUC per le prime 4 simulazioni.

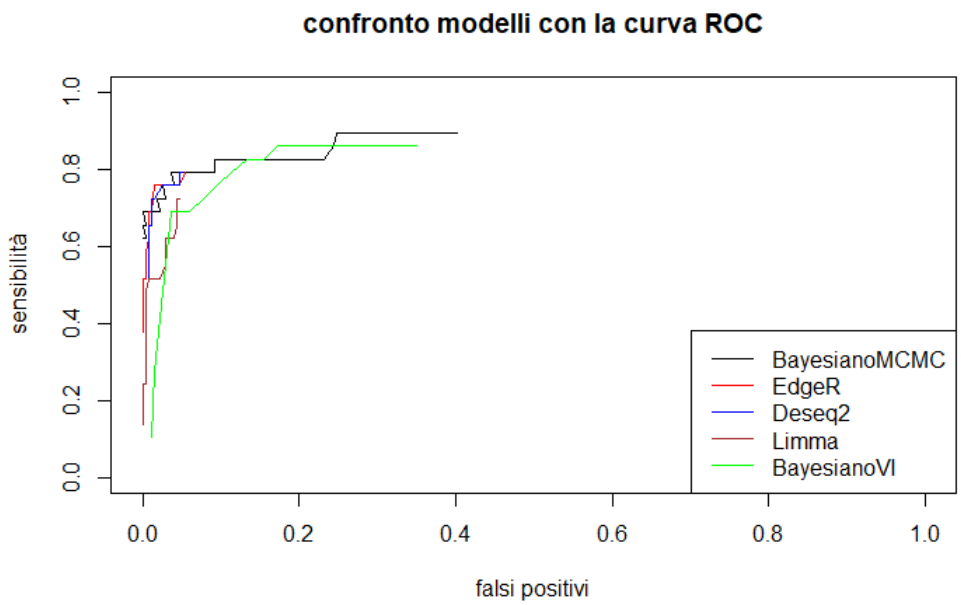

Figura 4.9: Curva ROC simulazione 1

simulazioni	MCMC	EdgeR	Deseq2	Limma	VI
1	0.905	0.887	0.89	0.847	0.874
2	0.865	0.835	0.818	0.789	0.829
3	0.866	0.837	0.799	0.79	0.828
4	0.93	0.911	0.906	0.899	0.911

Tabella 4.3: AUC dei modelli per le simulazioni 1-4

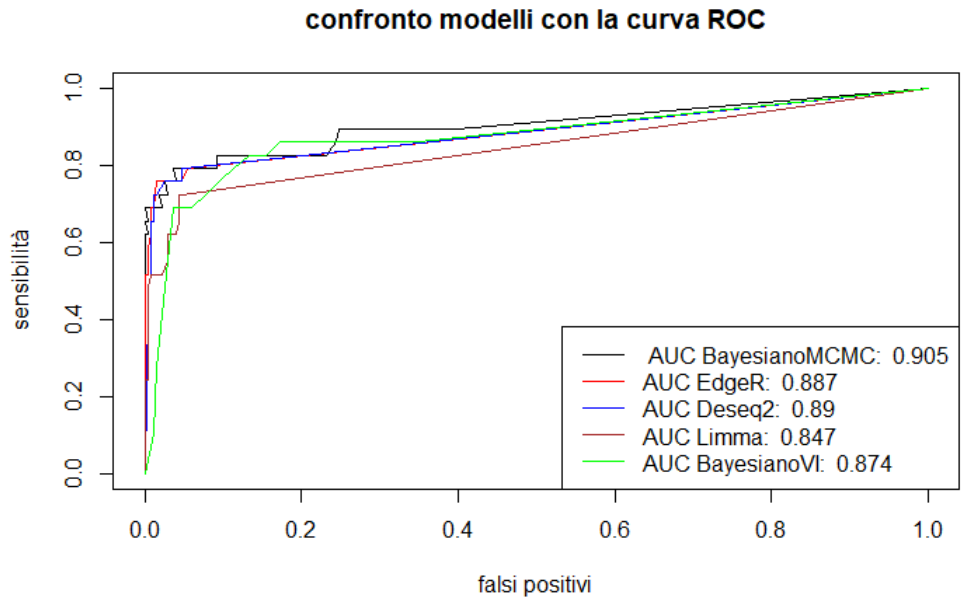

Figura 4.10: AUC simulazione 1

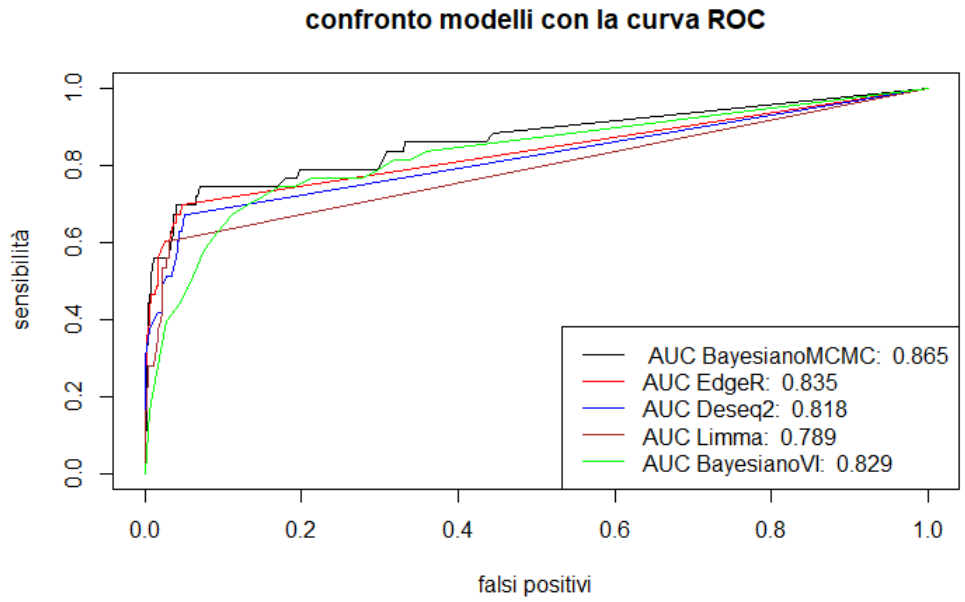

Figura 4.11: AUC simulazione 2

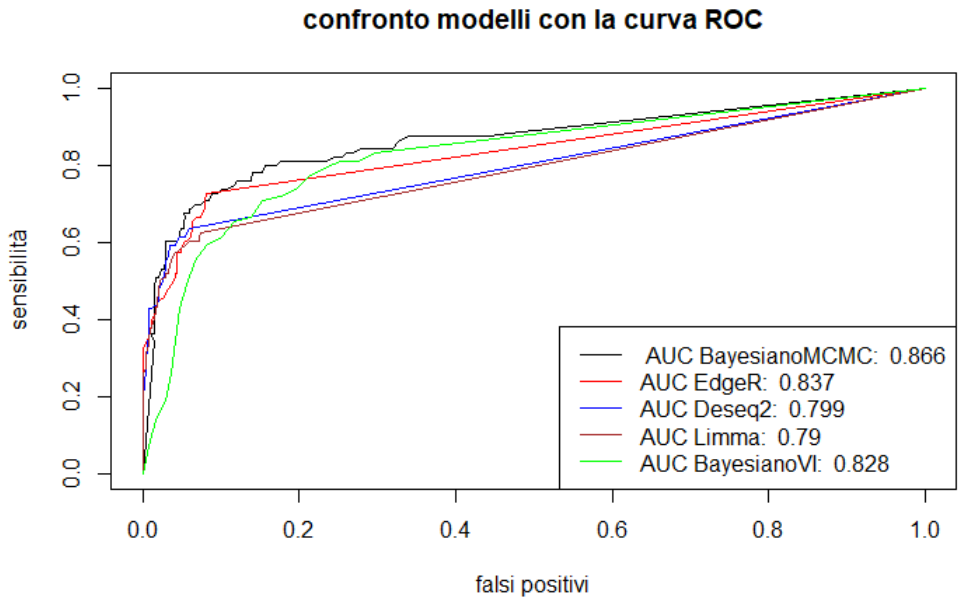

Figura 4.12: AUC simulazione 3

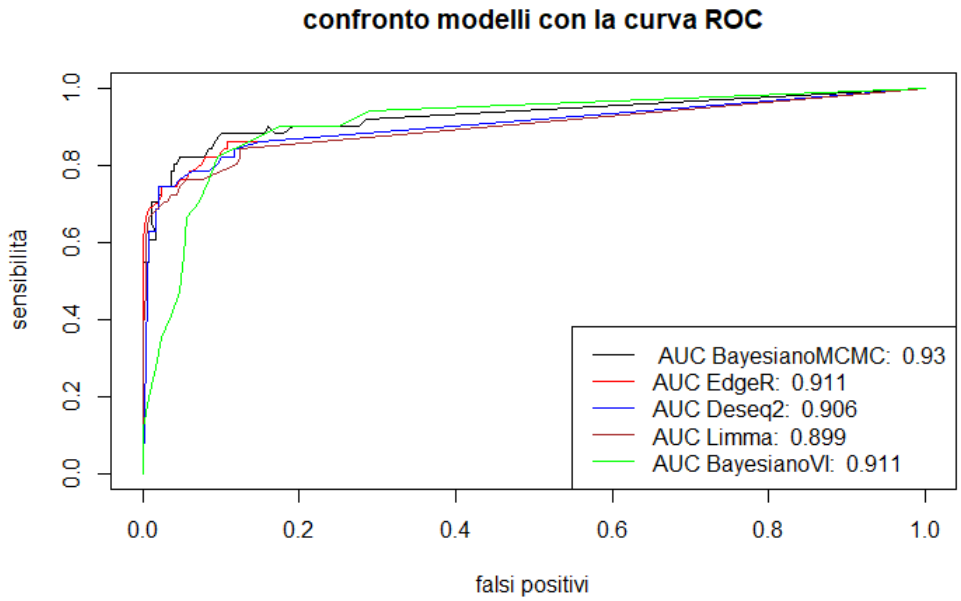

Figura 4.13: AUC simulazione 4

Le ultime considerazioni utili per finire il paragone sui modelli presentati per dati a singola cellula riguardano le performance dei modelli con una soglia prestabilita a priori. Nella Tabella 4.4 è possibile osservare i falsi positivi e falsi negativi dei vari metodi con le soglie prestabilite. Per i metodi classici si è scelta la percentuale del 5% sul FDR, che è quella comunemente usata per studi analoghi. Per il modello bayesiano che utilizza l'MCMC è stata decisa la soglia di $T = 0.71$, che è stata ricavata tramite la media delle soglie ottimali per le 4 simulazioni; dove queste ultime sono date da quel valore di T che permette il trade-off migliore tra falsi positivi e falsi negativi. Per l'inferenza variazionale invece sono state scelte due diverse soglie: vengono presi il 10% o il 20% di geni differenzialmente espressi a seconda che la simulazione di riferimento sia stata implementata con un 10% o un 20% di geni DE.
Nella tabella si può osservare come per l'appunto l'MCMC cerchi sempre di fare un trade-off favorevole tra i due tipi di errore mentre i 3 metodi classici controllano in modo ottimale l'errore di primo tipo ma presentano un valore di falsi negativi spesso molto alto (soprattutto Limma). Anche l'inferenza variazionale sembra controllare molto bene i valori dei falsi positivi (a parte il caso curioso della simulazione 3) concedendo però meno falsi negativi in confronto ai metodi classici. Nonostante non venga mostrato direttamente nella tesi si è riscontrato che per alcuni valori delle soglie sia l'MCMC che l'inferenza variazionale riescono a controllare in modo molto positivo entrambi gli errori. La flessibilità della decisione sulla soglia può essere un punto a favore di questi metodi (soprattutto per l'inferenza variazionale dato l'accettabile tempo computazionale), di contro però non avere un regola precisa sulla scelta di quest'ultima può comportare qualche difficoltà nell'identificarla quando si deve trattare con dati reali. Al contrario per i metodi classici la rigidità della soglia può essere un problema in termini di falsi negativi ma

	MCMC		**EdgeR**		**Deseq2**		**Limma**		**VI**	
Soglia	T = 0,71		FDR= 5%		FDR= 5%		FDR= 5%		geni DE: 10% o 20%	
Simulazioni	*FP*	*FN*	*FP*	*FN*	*FP*	*FN*	*FP*	*FN*	*FP*	*FN*
1	10%	17%	0%	55%	0,01%	35%	0%	62%	0,03%	31%
2	10%	25%	0,01%	60%	0%	70%	0%	86%	0,06%	49%
3	11%	26%	0,01%	65%	0,01%	64%	0%	76%	10%	38%
4	0,02%	25%	0,02%	27%	0,02%	31%	0%	43%	0,08%	23%

Tabella 4.4: Confronto tra falsi positivi e falsi negativi per i modelli con soglie fissate

permette di identificare i geni DE con più sicurezza quando si lavora coi dati reali.

4.3 Risultati modello gerarchico bayesiano

Anche per il secondo modello implementato nella tesi vengono condotte delle simulazioni per analizzarne le performance, nella Tabella 4.5 è possibile osservare le 4 compiute per questa occasione. Rispetto a quelle del precedente Paragrafo viene introdotto il parametro relativo al numero di campioni, poiché il modello bayesiano gerarchico è stato costruito proprio per contesti in cui ci sono più campioni da cui vengono sequenziate le cellule. Nuovamente il modello viene testato sia applicando il metodo MCMC sia utilizzando l'inferenza variazionale e successivamente le performance di questi due procedimenti vengono confrontate con EdgeR, Deseq2 e Limma. Anche per dati gerarchici l'inferenza variazionale presenta la distorsione vista negli istogrammi del Paragrafo 4.2, pertanto viene riproposto il procedimento basato sui

quantili per identificare i geni DE.

simulazioni	**numero di geni**	**numero di cellule**	**numero di campioni**	**% di geni DE**
5	300	24	2	10
6	300	24	4	10
7	500	24	4	20
8	350	36	6	20

Tabella 4.5: Parametri delle simulazioni con dati gerarchici

Nella Tabella 4.6 vengono esposte le tempistiche dell'MCMC e della VI per le simulazioni effettuate. In questo caso i tempi computazionali crescono ulteriormente per l'MCMC, questo dipende dall'incremento del numero di parametri da stimare. Infatti se per il modello bayesiano di base occorreva stimare j parametri per β_0 (con j numero di geni) in questa occasione bisogna stimare $j \cdot i \cdot 2 - j$ parametri aggiuntivi (con i numero di campioni) e di conseguenza l'onere computazionale aumenta in maniera considerevole. Ciononostante l'inferenza variazionale anche in questo caso semplifica di molto il processo computazionale necessario all'MCMC, passando nel caso della simulazione 3 da quasi 8 ore a meno di 20 minuti.

Tutti i metodi sono stati confrontati nuovamente con l'utilizzo della curva ROC, nelle Figure 4.14, 4.15, 4.16 e 4.17 sono osservabili le curve relative alle 4 simulazioni. Per la decisione delle soglie da includere nel ciclo sono state mantenute le scelte applicate per dati senza gerarchia anche in questa circostanza.

La cosa più sorprendente è l'ottima capacità predittiva dell'inferenza variazionale per queste tipologie di dati. Il metodo VI segue molto da vicino la prestazioni dell'MCMC ed è possibile affermare che con dati gerarchici l'in-

simulazioni	tempo MCMC	tempo VI
5	3h 32 min	6 min 37 sec
6	3 h 41 min	6 min 34 sec
7	7 h 38 min	18 min 15 sec
8	5 h 30 min	13 min 23 sec

Tabella 4.6: Tempi computazionali del modello bayesiano gerarchico per MCMC e VI

ferenza variazionale sia molto più conveniente rispetto all'MCMC, dal momento che a fronte di un tempo computazionale decisamente inferiore offre risultati perfettamente comparabili. Con l'applicazione dei metodi classici si riscontra ancora una volta un problema coi falsi negativi, e vengono raggiunti livelli di sensibilità abbastanza alti solamente per le ultime soglie del FDR, ovvero per valori difficilmente utilizzabili per dati reali.

Nella Tabella 4.7 vengono esposti i valori degli AUC per tutte le simulazioni con dati gerarchici. Come sottolineato precedentemente l'MCMC e la VI hanno risultati molto simili per tutte le simulazioni, gli AUC più alti di questi metodi sono permessi anche dalla maggior possibilità di trade-off tra falsi positivi e falsi negativi. I metodi classici invece controllando prevalentemente i falsi positivi non formano delle vere e proprie curve, infatti per la maggior parte delle soglie applicate presentano lo stesso numero sia di falsi positivi che di falsi negativi.

Infine, per completare la panoramica dei risultati per i modelli effettuati sui dati gerarchici, è interessante osservare il comportamento dei modelli con una soglia prestabilita a priori. In questa circostanza si è preferito mostrare un grafico differente rispetto alla tabella coi falsi positivi e falsi negativi presentata nel Paragrafo precedente. La Figura 4.18 infatti rappresenta l'in-

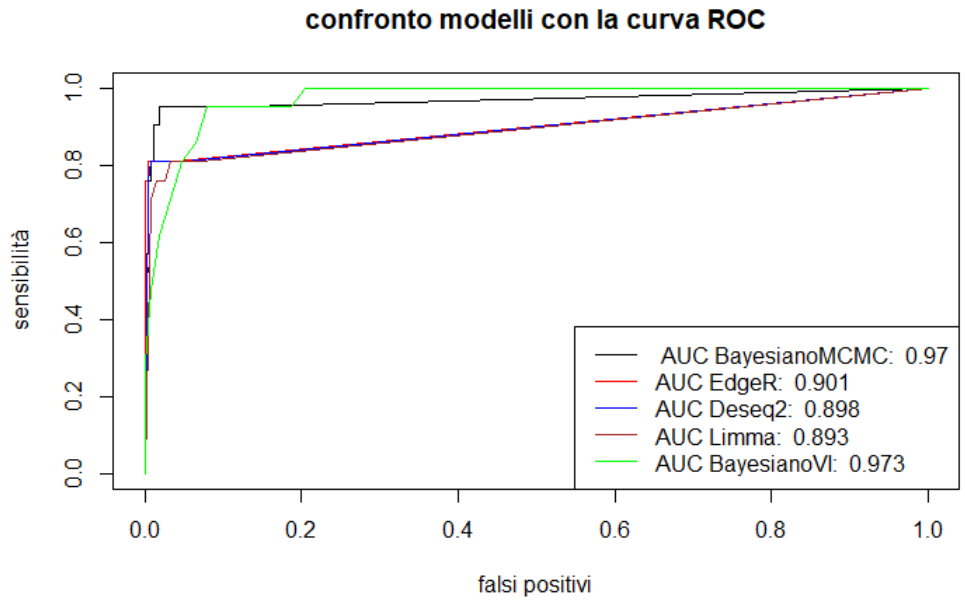

Figura 4.14: AUC simulazione 5

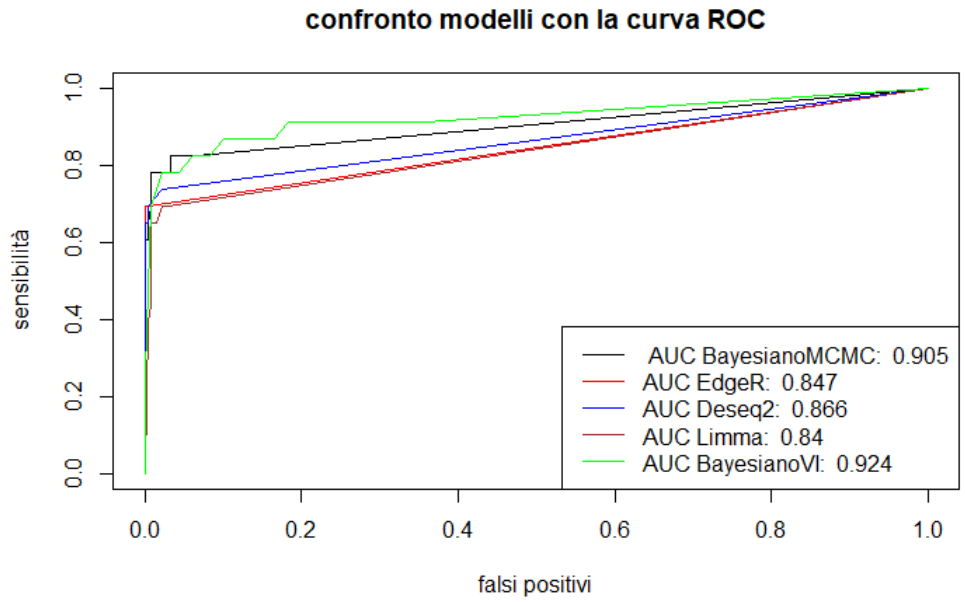

Figura 4.15: AUC simulazione 6

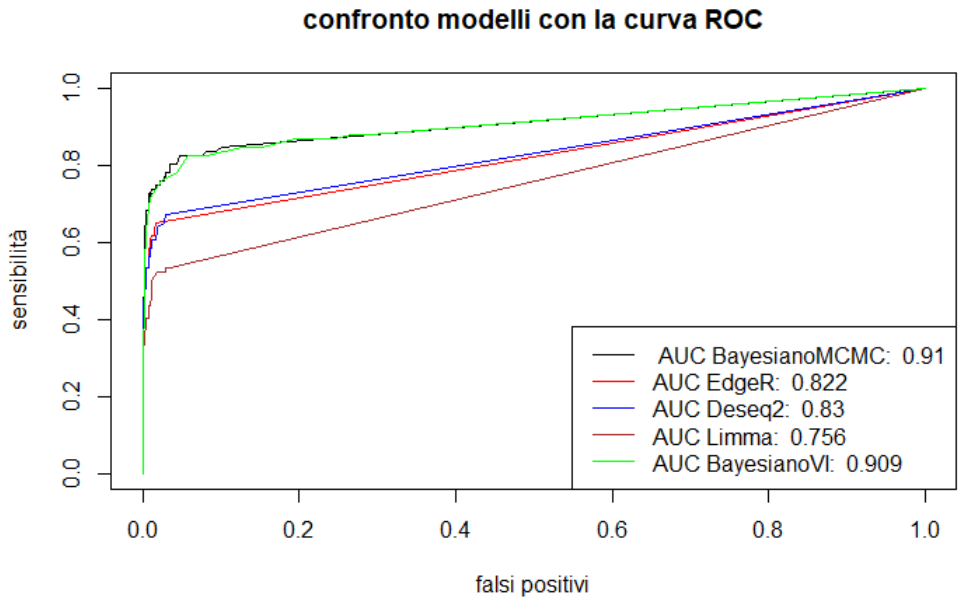

Figura 4.16: AUC simulazione 7

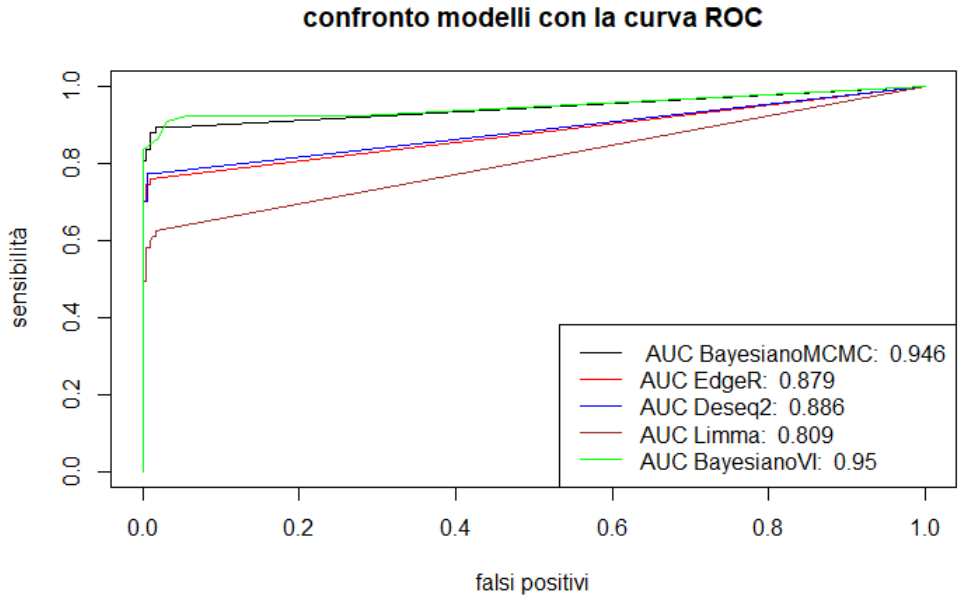

Figura 4.17: AUC simulazione 8

simulazioni	MCMC	EdgeR	Deseq2	Limma	VI
5	0.97	0.901	0.898	0.893	0.973
6	0.905	0.847	0.866	0.84	0.924
7	0.91	0.822	0.83	0.756	0.909
8	0.946	0.879	0.886	0.809	0.95

Tabella 4.7: AUC dei modelli per le simulazioni 5-8

tersezione dei geni identificati come DE tra i vari modelli e i reali geni differenzialmente espressi; per costruire il grafico viene considerata solamente la simulazione numero 7, ovvero quella costituita con un numero maggiore di geni. Per i metodi EdgeR, Deseq2 e Limma è stata optata la classica soglia del 5% sul FDR, per il metodo MCMC in questa occasione si è scelto di utilizzare la soglia ottimale, che in questo caso corrispondeva a $T = 0.46$; infine per l'inferenza variazionale si è deciso di prendere il 20% di geni presenti sulle code e identificarli come differenzialmente espressi rispecchiando la percentuale impostata a priori dalla simulazione, inoltre la percentuale del 20% corrisponde in questo caso pure alla soglia ottimale. Dal grafico si può notare come 36 geni dei 92 geni realmente DE vengono identificati da tutti i modelli, poi altri 31 vengono riconosciuti come DE solamente dall'MCMC e dalla VI e 16 geni non vengono mai considerati differenzialmente espressi da nessun modello. Complessivamente si può notare che utilizzando l'MCMC si riescono a identificare 76 dei 92 geni realmente DE, arrivando a una sensibilità decisamente alta, mentre sono solamente 19 i falsi positivi riconducibili a questo metodo. Anche l'inferenza variazionale ottiene dei risultati simili, individuando lo stesso numero di veri positivi dell'MCMC (76) e avendo un numero di falsi positivi leggermente superiore (24). Le performance di EdgeR e Deseq2 sono identiche, controllano ottimamente i falsi positivi (1 solo) con

una specificità quasi del 100% ma dimostrano una sensibilità decisamente bassa riconoscendo solamente 44 geni DE dei 92 reali (meno del 50%). Infine Limma è nuovamente in assoluto quello con le prestazioni peggiori: ottiene una sensibilità di circa il 40% (36 geni DE su 92), e anche un numero leggermente maggiore di falsi positivi rispetto ai due metodi classici (3 in totale).

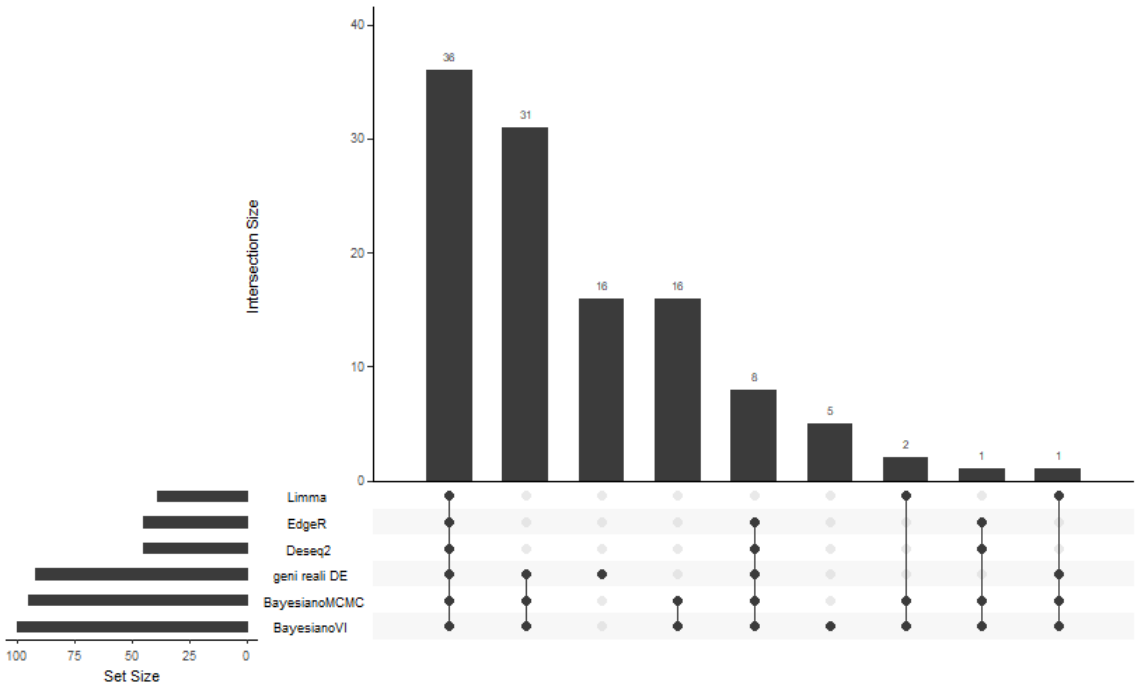

Figura 4.18: Intersezione tra i geni DE trovati dai modelli e i geni DE reali

4.4 Risultati dati reali

Lo scopo di questa tesi non è prettamente lo studio completo del dataset reale a disposizione valutandone la qualità ed esaminando tutte le sue componenti, ma piuttosto quello di validare i due modelli ideati, specie il modello che tiene conto della struttura gerarchica dei dati. Per uno studio più approfondito

sul dataset a disposizione si rimanda alla tesi di Zuin (2020) dalla quale è stato preso il pre-processamento dei dati come riferimento. Per ottenere il dataset finale infatti si è preso spunto dalle operazioni compiute dalla collega. Il dataset iniziale conteneva al suo interno circa 41mila geni e 61mila cellule. L'idea è di concentrarsi sulla sottopopolazione cellulare SST, ovvero una particolare tipologia di cellule neuronali che operano un'attività cerebrale variabile durante il ciclo sonno-veglia. In particolare le cellule SST sono un sottoinsieme degli interneuroni GABAergici che si interfacciano con la somatostasina, un ormone prodotto dall'ipotalamo e da altri organi del corpo umano, che ricopre una funzione cruciale all'interno dell'ipotalamo nell'inibire la secrezione di vari ormoni quali: l'ormone della crescita (GH), l'ormone tireostimolante (TSH), l'ormone adrenocorticotropo (ACTH) e la prolattina [6]. É stato dimostrato in letteratura che la privazione del sonno può comportare delle conseguenze importanti sulla regolazione della somastasina, dove un'assenza di sonno prolungata porta un incremento della produzione del GH (Toppilla et al., 1997).

La sottopopolazione cellulare SST comprende 761 cellule delle oltre 61mila di partenza. Per quanto riguarda i geni invece, si è deciso di concentrarsi sui 1000 con varianza più elevata. L'obiettivo è l'identificazione dei geni differenzialmente espressi nelle cellule SST tra i topi controlli e i topi trattati a cui è stato privato il sonno. Per questo scopo si è scelto di utilizzare il modello bayesiano gerarchico, che quindi considera la gerarchia dovuta al sequenziamento di cellule provenienti da più topi; inoltre viene impiegato l'algoritmo di inferenza variazionale poiché l'utilizzo del MCMC avrebbe comportato un tempo computazionale decisamente troppo elevato. Durante il proseguimento dell'analisi però sono stati riscontrati dei problemi iniziali;

[6] https://it.wikipedia.org/wiki/Somatostatina

infatti per poter applicare il modello con l'inferenza variazionale utilizzando il pacchetto "rstan"è necessario costruire una matrice con numero di righe corrispondenti al prodotto tra numero di geni e numero di cellule e numero di colonne uguale al numero di geni. La matrice in questione si rivela essere in un primo momento "troppo pesante"per essere supportata da R, ma successivamente aumentando il limite della memoria RAM utilizzabile da R è stato possibile creare con successo la matrice. Ciononostante non è stato comunque possibile far partire l'algoritmo d'inferenza variazionale poiché si presentava un problema di allocazione della matrice quando si provava a lanciare il comando per l'inferenza variazionale, di conseguenza la pesantezza della matrice continuava a bloccare l'esecuzione corretta dell'algoritmo. Date queste problematiche è stato necessario effettuare altre operazioni preliminari per permettere l'avvio della VI, si è deciso quindi di fare un filtro sulla media di riga dei 1000 geni con varianza più alta. Sono stati selezionati geni con un'espressione media maggiore di uno, rimanendo con un totale di 558 geni. A questo punto c'è stato un ulteriore tentativo di far partire l'algoritmo ma si è osservato empiricamente un blocco di R, che anche a distanza di ore non proseguiva coi normali passi richiesti dall'algoritmo. Come ultima possibilità si è optato per ridurre le cellule facenti parte dell'analisi, è stato operato un filtro sulle cellule con varianza più bassa eliminando dal dataset poco più di 100 cellule e arrivando a un totale di 647 cellule. Solamente dopo queste due ulteriori operazioni l'algoritmo d'inferenza variazionale è riuscito a compiere i i consueti passi e ad ottenere dei risultati. Il tempo computazionale richiesto da VI per portare a compimento l'algoritmo è stato di ben 7 ore e 49 minuti. Considerando le tempistiche della VI per i dati simulati con struttura gerarchica (Tabella 4.6) notiamo un incremento davvero notevole, questo aumento così marcato è attribuile soprattutto all'elevato numero di

cellule rispetto a quelle usate per le simulazioni.

Dopo aver ottenuto dei risultati da parte dell'algoritmo si è cercato di esplorare la distribuzione delle medie a posteriori per i parametri $\beta_{1,j}$ per decidere il miglior taglio possibile sulle code della distribuzione. Osservando la Figura 4.19 è stato deciso di prendere come geni DE quelli con una media a posteriori minore di 0 o maggiore di 0.55, in modo tale da non prendere geni troppo al centro della distribuzione e rischiare quindi di avere un valore troppo elevato di falsi positivi.

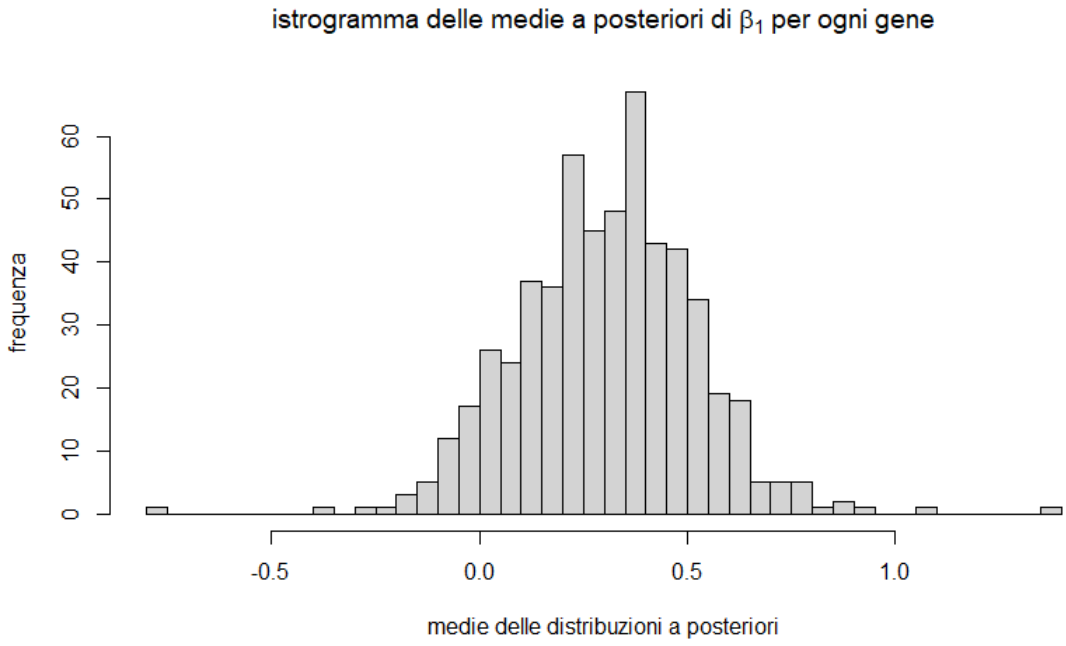

Figura 4.19: Istogramma delle medie a posteriori per il dataset reale

In totale i geni DE identificati sono 99: di cui 58 sovraespressi e 41 sottoespresssi. La confidenza di non aver selezionato un numero troppo alto di geni DE viene data dal numero di geni differenzialmente espressi trovati dai metodi classici. Difatti Deseq2 ne trova 246, EdgeR ne seleziona 223 e Limma arriva a 142. Questi numeri decisamente alti e quasi a livello del 50% di

geni totali per EdgeR e Deseq2 sono dovuti all'ampia operazione preliminare di filtraggio dei geni, che porta già in partenza a concentrarsi sui geni più espressi all'interno del dataset. Di conseguenza si sarebbe potuto procedere a una selezione più ampia per scegliere i geni DE derivanti dall'inferenza variazionale, ma si è preferito essere più conservativi per garantire la specificità del metodo e quindi per arrivare ad avere un numero di falsi positivi molto basso al netto di tralasciare qualche vero positivo.
Il problema che sorge a questo punto dell'analisi è: come appurare che l'insieme di geni DE selezionati sia effettivamente importante dal punto di vista biologico per lo studio in questione?
Per rispondere a questa domanda occorre condurre una "Over Representation Analysis".

4.4.1 Over Representation Analysis

Per dare un'interpretazione ai geni ricavati tramite l'inferenza variazionale sono disponibili due possibilità. La prima strada consiste nell'esaminare la funzione biologica di ogni gene riconosciuto come differenzialmente espresso. A tale scopo sono reperibili in internet dei database che codificano le caratteristiche principali della maggior parte dei geni come i due siti: *Gene Ontology Resource* [7] e *Molecular Signatures Database* [8]. Queste ontologie diventano particolarmente utili quanto l'interesse principale è indagare in modo approfondito le caratteristiche di un gene. Lo svantaggio di soffermarsi solamente su questi metodi è la mancanza di informazioni aggiuntive riguardanti le interazioni tra i geni riconosciuti come DE.
La seconda alternativa si riferisce all'identificazione di "pathway"comuni tra

[7] http://geneontology.org/
[8] http://www.gsea-msigdb.org/gsea/msigdb/index.jsp

i geni DE, dove per pathway si intende un gruppo di geni organizzati in un ordine specifico per eseguire una determinata funzione biologica. Per scoprire se alcuni pathway sono sovrarappresentati nella lista di geni DE rispetto i 41mila geni di partenza viene condotta una Over Representation Analysis (ORA), dove l'idea di base è per l'appunto di confrontare il gruppo di geni DE (gene set) con il totale iniziale dei geni. Adoperando il Test esatto di Fisher per le tabelle $2 \cdot 2$ è possibile determinare se la frazione di geni DE che risultano annotati per una determinata funziona biologica corrisponde a una frequenza maggiore di quella che ci si aspetterebbe sotto ipotesi di casualità; per comprendere se questa frequenza è statisticamente significativa può essere calcolato un p-value per ogni pathway utilizzando la distribuzione ipergeometrica:

$$p = 1 - \sum_{0}^{k-1} \frac{\binom{M}{i}\binom{N-M}{n-i}}{\binom{N}{i}}$$

In questa equazione N rappresenta il numero totale di geni di partenza, M sono i geni all'interno di N annotati per quel determinato pathway, n è il numero totale dei geni DE e k è la frazione di questi geni coinvolta per quella funzione biologica (Boyle et al., 2004). Successivamente i p-value calcolati per ogni pathway, essendo in una situazione di test multipli, vengono aggiustati con il metodo di Benjamini e Hochberg (1995).

Alla fine selezionando i p-value aggiustati con valore minore di 0.05 si otterrà una lista di fenotipi, ovvero una serie di funzionalità biologiche per cui il gene set risulta significativamente sovrarappresentato rispetto l'insieme iniziale dei geni. L'analisi ORA viene condotta in questa tesi utilizzando il pacchetto di Bionconductor "clusterProfiler". I risultati dell'analisi sono visibili nella Figura 4.20, dove le funzioni biologiche vengono ordinate in base al grado di significatività iniziando da quella più significativa. In seguito i risultati ottenuti tramite la Over Representation Analysis sono stati confrontati con le

conoscenze riguardanti gli effetti della privazione del sonno sui topi presenti in letteratura. I pathway biologici presenti nel grafico sembrano essere tutti associati con caratteristiche comuni alla privazione del sonno. È stato dimostrato infatti che la privazione del sonno nei topi porta a un indebolimento delle sinapsi gluttamatergiche (Liu et al., 2016), inoltre è stato provato che l'interruzione del ritmo circadiano regolare comporta un aumento delle malattie cardiovascolari, di cui fanno parte anche le cardiomiopatie (Lefta et al., 2012). Anche il sistema serotoninergico è legato alla regolazione del sonno, infatti durante la fase REM le funzionalità dei recettori serotoninergici sono minime, e la privazione del sonno porta ad aumento dell'attività di questi recettori (Adrien, 2002). Non godere del ristoro del sonno inoltre risulta essere un fattore di rischio anche per la depressione (Xu et al., 2020).
In conclusione, sembra che i pathway più significativi siano correlati con l'attività del sonno. Per questo motivo i risultati ottenuti con l'inferenza variazionale possono essere ritenuti soddisfacenti, una controprova dell'efficienza del metodo è data dai pathway molti simili notati per i 3 metodi classici quando viene condotta lo stesso tipo di analisi ORA.

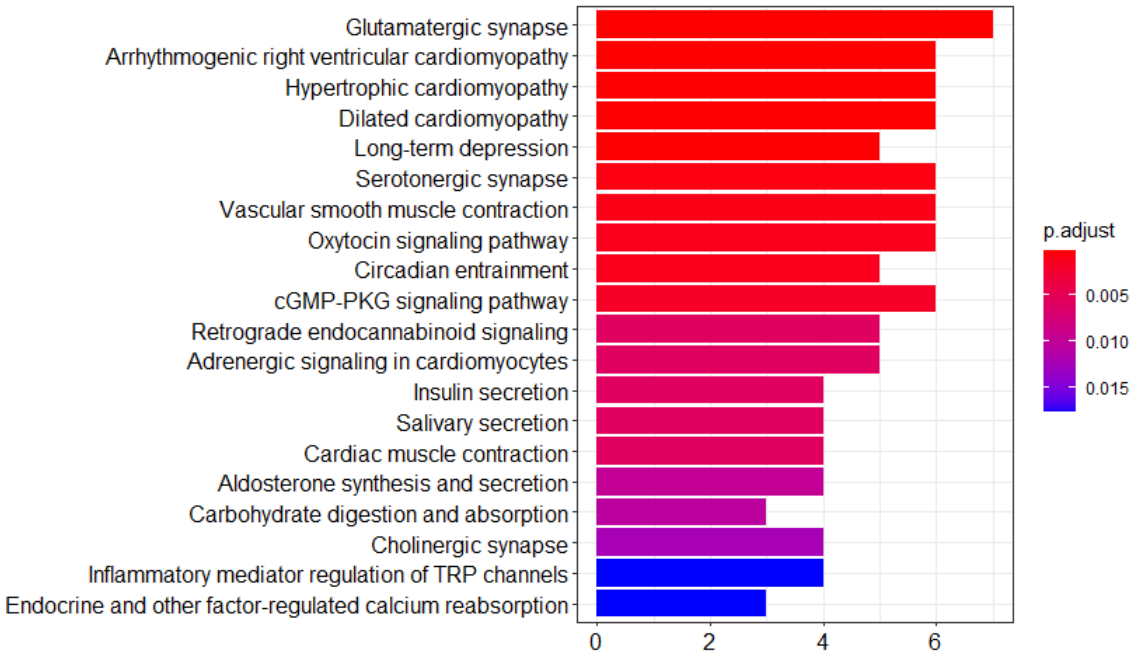

Figura 4.20: Pathway biologici sovrarappresentati nella lista di geni DE

Conclusioni

A questo punto bisogna tirare le fila di tutto il lavoro che è stato svolto. Il principale scopo della tesi era proporre un nuovo modello che tenesse conto della struttura gerarchica dei dati, e in particolare che includesse l'informazione a priori del campione da cui viene prelevato il materiale biologico.

Si può affermare che l'obiettivo preposto è stato raggiunto con successo e che il modello funziona in modo ottimale con le simulazioni compiute, identificando con precisione i geni DE. Utilizzando l'algoritmo MCMC e avendo un elevato numero di parametri da stimare si è riscontrato però un tempo computazionale particolarmente oneroso. Per questo motivo si è cercato di percorrere la strada dell'inferenza variazionale, la quale riduce estremamente le tempistiche dell'MCMC.
Nel contesto dei dati con struttura gerarchica, introducendo il test statistico basato sui quantili, si è potuto sopperire alle distorsioni evidenti presenti nella VI e si è riusciti ad ottenere delle prestazioni ugualmente soddisfacenti a quelle dell'MCMC.
Nei dati a singola cellula provenienti da un unico campione è stato implementato un modello differente che non include la gerarchia sul parametro β_0, in questo caso le performance dell'MCMC risultano leggermente migliori dell'inferenza variazionale.

Quest'ultimo modello è stato cronologicamente generalizzato prima rispetto al modello gerarchico, ed è servito poi come base metodologica per passare al modello più complicato.

Questa tesi cerca anche parallelamente di esaminare i vantaggi e gli svantaggi dei metodi più noti in letteratura: EdgeR, Deseq2 e Limma. Un risultato interessante riscontrato con le simulazioni sono le prestazioni peggiori di Limma rispetto agli altri metodi.

Infine per testare in modo completo il modello ideato occorreva analizzarne le performance su un dataset reale. Come dataset reale si è scelto l'esperimento condotto sui topi domestici da parte del laboratorio della Prof.ssa Lucia Peixoto. Lo studio inglobava al suo interno 6 differenti topi ed è stato prelevato un campione biologico da ogni topo. L'obiettivo dell'esperimento era comprendere l'effetto di un trattamento (privazione del sonno) sull'espressione genica dei topi. Pertanto il modello gerarchico bayesiano è risultato adeguato per questa sperimentazione.
Dopo essere riusciti a superare qualche ostacolo computazionale si è arrivati ad ottenere un gruppo di geni considerati differenzialmente espressi. Tramite la Over Representation Analysis poi è stato dato un significato a questo insieme di geni e sono state identificate un'insieme di funzioni biologiche per cui questi geni sono significativamente sovrarappresentati rispetto al totale dei geni di partenza.

Un ulteriore sviluppo futuro del modello potrebbe essere quello di renderlo più veloce dal punto di vista computazionale per consentire delle analisi su un insieme più ampio di geni o di cellule. Un'idea che è stata pensata

durante la tesi è quella di avvalersi di tecniche di machine learning, le quali utilizzano le schede grafiche per eseguire le istruzioni e anziché elaborare un comando per volta in maniera seriale portano avanti migliaia di operazioni contemporaneamente (in parallelo).
Il modello proposto dunque si può ritenere decisamente soddisfacente ma modificabile per avere delle tempistiche più brevi; un lavoro futuro quindi potrebbe sfruttare tecniche di machine learning per diminuire il procedimento computazionale (come ad esempio l'uso di TensorFlow).

Bibliografia

Adrien, J. (2002). Neurobiological bases for the relation between sleep and depression. *Sleep medicine reviews*, 6(5), 341-351.

Ankeny, R. A., e Leonelli, S. (2011). What's so special about model organisms?. *Studies in History and Philosophy of Science Part A*, 42(2), 313-323.

Azzalini, A., e Scarpa, B. (2012). *Data analysis and data mining: An introduction*. OUP, New York, USA.

Benjamini, Y., e Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1), 289-300.

Betancourt, M. (2017). A conceptual introduction to Hamiltonian Monte Carlo. *arXiv preprint arXiv*:1701.02434.

Betancourt, M., e Girolami, M. (2015). Hamiltonian Monte Carlo for hierarchical models. *Current trends in Bayesian methodology with applications*, 79(30), 2-4.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer, New York, USA.

Blei, D. M., Kucukelbir, A., e McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518), 859-877.

Boyle, E. I., Weng, S., Gollub, J., Jin, H., Botstein, D., Cherry, J. M., e Sherlock, G. (2004). GO:: TermFinder—open source software for accessing Gene Ontology information and finding significantly enriched Gene Ontology terms associated with a list of genes. *Bioinformatics*, 20(18), 3710-3715.

Carlén, M. (2017). What constitutes the prefrontal cortex?. *Science*, 358(6362), 478-482.

Chini, M., e Hanganu-Opatz, I. L. (2020). Prefrontal cortex development in health and disease: lessons from rodents and humans. *Trends in Neurosciences*, 44(3), 227-24.

Efron, B. (2007). Size, power and false discovery rates. *The Annals of Statistics*, 35(4), 1351-1377.

Gallagher, K., Charvin, K., Nielsen, S., Sambridge, M., e Stephenson, J. (2009). Markov chain Monte Carlo (MCMC) sampling methods to determine optimal models, model resolution and model choice for Earth Science problems. *Marine and Petroleum Geology*, 26(4), 525-535.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., e Rubin, D. B. (2013). *Bayesian data analysis*. CRC press, Boca Raton, USA.

Harrison, Y., e Horne, J. A. (2000). Sleep loss and temporal memory. *The Quarterly Journal of Experimental Psychology: Section A*, 53(1), 271-279.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1), 97-109.

Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., e Blei, D. M. (2017). Automatic differentiation variational inference. *The Journal of Machine Learning Research*, 18(1), 430-474.

Kucukelbir, A., Ranganath, R., Gelman, A., e Blei, D. M. (2015). Automatic variational inference in Stan. *arXiv preprint arXiv*:1506.03431.

Lefta, M., Campbell, K. S., Feng, H. Z., Jin, J. P., e Esser, K. A. (2012). Development of dilated cardiomyopathy in Bmal1-deficient mice. *American Journal of Physiology-Heart and Circulatory Physiology*, 303(4), H475-H485.

Liu, Z., Wang, Y., Cai, L., Li, Y., Chen, B., Dong, Y., e Huang, Y. H. (2016). Prefrontal cortex to accumbens projections in sleep regulation of reward. *Journal of Neuroscience*, 36(30), 7897-7910.

Lun, A. T., e Marioni, J. C. (2017). Overcoming confounding plate effects in differential expression analyses of single-cell RNA-seq data. *Biostatistics*, 18(3), 451-464.

Martini, F., Timmons, M. J. e Tallitsch, R. B., (2009). *Human anatomy.* Pearson Benjamin Cummings, San Francisco, USA.

Morse III, H. C. (2007). Building a better mouse: One hundred years of genetics and biology. *The mouse in biomedical research*, 1, 1-11.

Muzur, A., Pace-Schott, E. F., e Hobson, J. A. (2002). The prefrontal cortex in sleep. *Trends in cognitive sciences*, 6(11), 475-481.

Phifer-Rixey, M., e Nachman, M. W. (2015). The Natural History of Model Organisms: Insights into mammalian biology from the wild house mouse Mus musculus. *Elife*, 4, e05959.

Picelli, S., Faridani, O. R., Björklund, Å. K., Winberg, G., Sagasser, S., e Sandberg, R. (2014). Full-length RNA-seq from single cells using Smart-seq2. *Nature protocols*, 9(1), 171-181.

Risso, D., Sales, G., Romualdi, C., e Chiogna, M. (2013). A hierarchical Bayesian model for RNA-Seq data. *Complex Models and Computational Methods in Statistics*, 215-227

Salvan, A., Sartori, N., e Pace, L. (2020). *Modelli lineari generalizzati.* Springer-Verlag, Milano, Italia.

Shapiro, E., Biezuner, T., e Linnarsson, S. (2013). Single-cell sequencing-based technologies will revolutionize whole-organism science. *Nature Reviews Genetics*, 14(9), 618-630.

Stan Development Team (2020). RStan: the R interface to Stan. R package version 2.21.2. http://mc-stan.org/.

Tang, F., Barbacioru, C., Wang, Y., Nordman, E., Lee, C., Xu, N., ... e Surani, M. A. (2009). mRNA-Seq whole-transcriptome analysis of a single cell. *Nature methods*, 6(5), 377-382.

Toppila, J., Alanko, L., Asikainen, M., Tobler, I., Stenberg, D., e PORKKA-HEISKANEN, T. A. R. J. A. (1997). Sleep deprivation increases somatostatin and growth hormone-releasing hormone messenger RNA in the rat hypothalamus. *Journal of sleep research*, 6(3), 171-178.

Van De Wiel, M. A., Leday, G. G., Pardo, L., Rue, H., Van Der Vaart, A. W., e Van Wieringen, W. N. (2013). Bayesian analysis of RNA sequencing data by estimating multiple shrinkage priors. *Biostatistics*, 14(1), 113-128.

Ventrucci, M., Scott, E. M., e Cocchi, D. (2011). Multiple testing on standardized mortality ratios: a Bayesian hierarchical model for FDR estimation. *Biostatistics*, 12(1), 51-67.

Wakefield, J. (2013). *Bayesian and frequentist regression methods*. Springer Science & Business Media, New York, USA.

Wang, Z., Gerstein, M., e Snyder, M. (2009). RNA-Seq: a revolutionary tool for transcriptomics. *Nature reviews genetics*, 10(1), 57-63.

Xu, X., Zheng, P., Zhao, H., Song, B., e Wang, F. (2020). Effect of Electroacupuncture at GV20 on Sleep Deprivation-Induced Depression-Like Behavior in Mice. *Evidence-Based Complementary and Alternative Medicine*, 2020, 1-11.

Yao, Y., Vehtari, A., Simpson, D., e Gelman, A. (2018). Yes, but did it work?: Evaluating variational inference. *International Conference on Machine Learning*, 80, 5581-5590.

Zuin, E. (2020). Identificazione di geni differenzialmente espressi per dati di scRNA-seq: un caso studio sulle cellule della corteccia prefrontale in assenza di sonno. *Tesi di Laurea Magistrale in Scienze Statistiche, Università degli Studi di Padova, Anno Accademico 2019-2020.*

Appendice

Modello bayesiano in Stan

```
data {
  int<lower=0> N;  //numero totale di dati (celle*geni)
  int<lower=1> G; //numero di geni
  int<lower=1> C; //numero di cellule
  int<lower=1> P; //numero di variabili
  int y[N];    //conteggi
  int<lower=1, upper=G> gene[N]; //identificatore di gene
  int<lower=1, upper=C> cell[N]; identificatore di cellule
  vector[P] x[N];
  real<lower=0,upper=1> p0; //parametro di una Bernoulli
}

parameters {
  matrix[G, P] beta_mu;
  matrix[C, 1] gamma_mu;
  real<lower=0> varzbeta1;
  real alpha;
}
```

```
model {
  row_vector[1] mu;
  real eta;
  eta = log_mix(p0, normal_lpdf(alpha | 0, 0.1),
                    normal_lpdf(alpha | 0, 1));
  1/varzbeta1~ gamma(0.01,0.01);
  to_vector(beta_mu[,1]) ~ normal(0, varzbeta1);
  to_vector(beta_mu[,2]) ~ normal(0, exp(eta));
  to_vector(gamma_mu) ~ normal(0, 1);

  for (n in 1:N){
    mu = exp(beta_mu[gene[n]] * x[n] + gamma_mu[cell[n]]);
   target += poisson_lpmf(y[n] | mu);

  }
}
```

Modello bayesiano gerarchico in Stan

```
data {
  int<lower=0> N;  //numero totale di dati (celle*geni)
  int<lower=1> G; //numero di geni
  int<lower=1> C; //numero di cellule
  int<lower=1> J; //numero di persone/topi
  int y[N];    //conteggi
  int<lower=1, upper=G> gene[N]; //identificatore di gene
  int<lower=1, upper=C> cell[N]; identificatore di cellule
  int<lower=1,upper=J> jj[N]; //identificatore di persone/topo
```

```
  matrix[N, G] x;
  int z[N];
  real<lower=0,upper=1> p0; //parametro di una Bernoulli
}

parameters {
  vector[G] beta1[J];
  vector[G] beta2;
  vector[G] gamma1[J];
  matrix[C, 1] gamma_mu;
  real<lower=0> varzbeta1;
  real alpha;
}

model {
  row_vector[1] mu;
  real eta;
  eta = log_mix(p0, normal_lpdf(alpha | 0, 0.1),
                    normal_lpdf(alpha | 0, 1));
  1/varzbeta1~ gamma(0.01,0.01);
  beta_mu2 ~ normal(0, exp(eta));
  gamma1[J] ~ normal(0, 1);
  to_vector(gamma_mu) ~ normal(0, 1);
  beta_mu1[J]~ normal(gamma1[J], varzbeta1);

for (n in 1:N) {
  mu = exp(x[n]*(beta_mu1[jj[n]])+ z[n]*beta_mu2[gene[n]]+ gamma_mu[cell[n]]);
  target += poisson_lpmf(y[n] | mu);
  }
```

```
}
```

Codice R per simulare dati non gerarchici e applicare gli algoritmi MCMC e VI

```
  suppressPackageStartupMessages({
    library(splatter)
    library(scater)
  })
  library(magrittr)
  library(ISwR)
  library(rstan)
  library(StanHeaders)
  library(inline)
  options(mc.cores = parallel::detectCores())
  rstan_options(auto_write = TRUE)

#INIZIO SIMULAZIONE
  set.seed(3)
  sce <- mockSCE(ncells = 20,ngenes = 500)
  params<- splatEstimate(sce)
  primosim<- splatSimulate(params,
  group.prob = c(0.5,0.5), de.prob=c(0.1,0.1),
  method = "groups", de.facLoc=1.2, verbose=F)

  yy <- counts(primosim)
  y <- as.vector(yy)
  xx <- model.matrix(~primosim$Group)
  ngenes <- nrow(yy)
  ncells <- ncol(yy)
```

```
x <- matrix(data = c(rep(1, length(y)),
                     rep(xx[,2], each=ngenes)), ncol=2)

data_low <- list(N = length(y),
                 P = 2,
                 Q = 2,
                 G = ngenes,
                 C = ncells,
                 x = x,
                 w = x,
                 y = y,
                 gene = rep(seq_len(ngenes), ncells),
                 cell = rep(seq_len(ncells), each = ngenes),
                 p0 = 0.1)

fileName <- "modellobayesiano.stan"
stan_code <- readChar(fileName, file.info(fileName)$size)
cat(stan_code)

#MCMC
timeMCMC <-system.time( MCMC.bayes <- stan(model_code = stan_code,
data = data_low,chains = 4, iter = 2000, warmup = 1000, thin = 1))

#INFERENZA VARIAZIONALE
provavb <-stan_model(fileName)
timeVI <-system.time(VI.bayes <-vb(provavb,data=data_low,
tol_rel_obj = 1e-3,iter=10000))
```

Codice R per simulare dati gerarchici e applicare gli algoritmi MCMC e VI

```
  suppressPackageStartupMessages({
    library("splatter")
    library("scater")
    library("VariantAnnotation")
    library("ggplot2")
  })
  library(magrittr)
  library(ISwR)
  library(rstan)
  library(StanHeaders)
  library(inline)
  options(mc.cores = parallel::detectCores())
  rstan_options(auto_write = TRUE)

#INIZIO SIMULAZIONE
  vcf <- mockVCF(n.samples = 4)
  gff <- mockGFF(n.genes = 500)

  sim.means <- splatPopSimulateMeans(vcf = vcf, gff = gff,
                     params = newSplatPopParams(de.facLoc = 1.2,
                                                de.facScale = 0.5,
                                                de.prob = c(0.05,0.05),
                                                group.prob = c(0.5, 0.5)))
  sim.means$key
  sim.means.rep2 <- splatPopSimulateMeans(vcf = vcf, key=sim.means$key,
                                  params = newSplatPopParams())
```

```
sim.sc <- splatPopSimulateSC(params=newSplatPopParams(batchCells=6,
                                                      de.facLoc = 1.2,
                                                      de.facScale = 0.5,
                                                      de.prob = c(0.05,0.05),
                                                      group.prob = c(0.5, 0.5)),
                                                      key = sim.means$key,
                                                      sim.means=sim.means$means))


colData(sim.sc)$Group
colData(sim.sc)$Sample
colData(sim.sc)
table(coldat$Group,coldat$Sample)


ngroups<-2
nsamples<-dim(vcf)[2]
yy <- counts(sim.sc)
y <- as.vector(yy)
xx <- model.matrix(~sim.sc$Group)
J=dim(vcf)[2]
partizioni<-ngroups*nsamples
ngenes <- nrow(yy)
ncells <- ncol(yy)/(ngroups*nsamples)
x<-matrix(rep(diag(c(rep(1,ngenes))),partizioni*ncells),
nrow=ngenes*partizioni*ncells,byrow=T)
x1 <- matrix(data = c(rep(1, length(y)),
                      rep(xx[,2], each=ngenes)), ncol=2)
x1<-x1[,2]
# jj nel caso di 4 campioni, modificabile in base al numero di
campioni considerato
```

```
jj<-c(rep(c(rep(1,ngenes*ncells),rep(2,ngenes*ncells),rep(3,ngenes*ncells)
,rep(4,ngenes*ncells)),ngroups))
cells<-ncells*ngroups*nsamples
data_low <- list(N = length(y),
                 G = ngenes,
                 C = cells,
                 x = x,
                 z=x1,
                 y = y,
                 jj=jj,
                 J=J,
                 gene = rep(seq_len(ngenes), ncells*partizioni),
                 cell = rep(seq_len(cells), each = ngenes),
                 p0 = 0.1)

fileName <- "modellobayesgerarchico.stan"
system.time(stan_code <- readChar(fileName, file.info(fileName)$size))
cat(stan_code)
#MCMC
time<-system.time(MCMC <- stan(model_code = stan_code, data = data_low,
chains = 4, iter = 2000, warmup = 1000, thin = 1))
#INFERENZA VARIAZIONALE
provavb<-stan_model(fileName)
timeVI<-system.time(VI<-vb(provavb,data=data_low,tol_rel_obj = 1e-4,iter=10000))
```