Università degli Studi di Padova

Dipartimento di Scienze Statistiche

Corso di Laurea Magistrale in

Scienze Statistiche

Hamiltonian Monte Carlo discontinuo per parametri discreti e continui: una versione efficiente

Relatore Prof. Bruno Scarpa

Dipartimento di Scienze Statistiche

Laureando: Manildo Paolo

Matricola N 2072679

Anno Accademico 2023/2024

Indice

Introduzione

Gli algoritmi Markov Chain Monte Carlo, abbreviati con la sigla MCMC, sono strumenti potenti ma fragili. Sia la loro efficacia che la loro efficienza variano in base al problema in esame e dipendono da un numero di parametri specifici al tipo di algoritmo utilizzato. Una delle classi più promettenti tra questi metodi è Hybrid Monte Carlo (HMC, spesso indicato come Hamiltonian Monte Carlo, Duane et al., 1987), che prende in prestito concetti dalla geometria differenziale, dalla fisica delle particelle e dalla meccanica statistica per esplorare in modo efficiente densità di probabilità caratterizzate da zone ad elevata curvatura. La sua popolarità è aumentata negli ultimi anni grazie allo sviluppo di STAN (Gelman et al., 2015), un linguaggio di programmazione probabilistico per l'inferenza bayesiana che ha superato in popolarità i suoi concorrenti che facevano affidavano su algoritmi di tipo Metropolis-within-Gibbs.

La potenza di questo tipo di algoritmi risiede nella particolare struttura geometrica implicata dai sistemi hamiltoniani, che assume la differenziabilità delle funzioni che li caratterizzano. Questo presenta l'importante svantaggio di non renderli applicabili a superfici non lisce. In un contesto statistico, ciò può accadere quando alcuni degli oggetti di inferenza desiderati non sono continui. In un contesto bayesiano, accade ad esempio quando la natura di alcuni parametri è quantitativa discreta o qualitativa. In risposta a questa mancanza, in Nishimura et al. (2020) viene proposta una versione particolare di HMC, che incorpora tali parametri in uno spazio continuo e assegna loro una distribuzione a priori a gradini. Sebbene ovviamente ancora non differenziabile, è comunque possibile sfruttare le proprietà delle dinamiche hamiltoniane per esplorare la risultante densità a posteriori, grazie ad un

concetto, preso in prestito ancora una volta dalla fisica, relativo alla rifrazione della luce.

L'obiettivo di questa tesi è quello di discutere tale metodo in dettaglio e proporne una variante algoritmica, più efficiente da un punto di vista computazionale, simile a quella utilizzata da STAN, resa disponibile sotto forma di libreria R con il nome XDNUTS. Tale variante utilizza dei criteri di terminazione (Betancourt, 2016*b*) in grado di identificare di volta in volta la lunghezza ottimale per l'approssimazione delle traiettorie definite dalle equazioni di Hamilton, in maniera tale da lasciare che sia la geometria della legge di probabilità a stabilire quanto a fondo esplorare il relativo spazio per ogni iterazione dell'algoritmo. Inoltre, viene proposta, sempre in analogia con STAN, una procedura di ottimizzazione stocastica per calibrare adattivamente i parametri del metodo in maniera quasi automatica, anche se la validità e l'ottimalità di quest'ultima richiedono ulteriori approfondimenti.

Nel Capitolo 1 viene fatta una breve introduzione ai principali metodi MCMC noti in letteratura, con il principale scopo di evidenziare, per ciascuno di essi, gli aspetti critici che motivano il successo di Hybrid Monte Carlo nei loro confronti. Nel Capitolo 2 si introduce Hybrid Monte Carlo: le prime due sezioni riassumono i concetti principali di fisica necessari a comprenderne il funzionamento a livello teorico, la terza sezione ne discute il reale funzionamento, mentre la quarta introduce la sua variante più famosa, alla base del linguaggio STAN e da cui la libreria XDNUTS trae spunto. Il Capitolo 3 vede una suddivisione simile al precedente: nelle prime due sezioni si discutono, da un punto di vista prettamente teorico, le problematiche e le soluzioni possibili per l'applicazione di Hybrid Monte Carlo a leggi di probabilità discontinue. Nella terza sezione invece viene presentato il metodo proposto da Nishimura et al. (2020), mentre nella quarta si descrive la variante efficiente definita in questa sede. Per quest'ultima si forniscono motivazioni teoriche ma anche pratiche, risultato di uno studio di simulazione, riguardo la sua efficacia in questi contesti. Il suo funzionamento viene messo alla prova ulteriormente nel Capitolo 4, dove il metodo è applicato a due problemi di interesse: un problema di imputazione di dati mancanti discreti e un modello con due punti di cambio. Per

ciascuno di questi, l'algoritmo è confrontato con alcune possibili alternative già note in letteratura. Infine, nelle Conclusioni, si traggono le somme riguardo la procedura implementata e quella descritta in Nishimura et al. (2020) da cui, per forza di cose, eredita le principali proprietà.

Capitolo 1

Markov Chain Monte Carlo, una breve introduzione

I metodi MCMC producono una sequenza di valori realizzazione di un particolare processo stocastico. Lo scopo ultimo è quello di stimare, attraverso questi campioni simulati, una data quantità di interesse. Seppure la loro applicabilità riguardi sia l'approccio frequentista che quello bayesiano, quest'ultimo è quello che ne è maggiormente interessato dal momento che, trattando anche i parametri come variabili aleatorie, la loro inferenza è sempre basata su una legge di probabilità.

In ambito frequentista sono di uso comune i semplici metodi Monte Carlo, utilizzati ad esempio nei modelii ad effetti casuali, per l'integrazione di questi, o nella risoluzione del passo di *Expectation* dell'algoritmo EM (Laird et al., 1987), dove tipicamente si è facilmente in grado di simulare dalla legge di probabilità desiderata. Ogni qualvolta non si conosce un metodo per simulare *indipendentemente* dalla legge di probabilità di interesse, in maniera efficiente, un possibile strumento consiste nell'abbandonare tale proprietà, che garantisce una più veloce convergenza delle stime grazie alla legge dei grandi numeri, e simulare campioni non più indipendenti e identicamente distribuiti ma solo con distribuzione marginale identica. In tal caso, il teorema ergodico, si veda ad esempio Robert et al. (1999, paragrafo 6.7.1), assicura sotto opportune condizioni di tale sequenza di campioni, una convergenza

delle stime, che però è generalmente più lenta[1]. Per simulare questa sequenza di valori dipendenti, ma marginalemnte identicamente distribuiti, tipicamente si utilizzano delle Catene di Markov, vedere ad esempio Brémaud (2013), la cui teoria è vasta e le applicazioni molteplici e solo una branca di queste riguarda le procedure qui di interesse.

In ambito bayesiano, a differenza di quello frequentista, l'utilizzo di questi metodi di simulazione risulta molto spesso necessario per almeno due motivi:

- la difficoltà nel calcolare/approssimare il denominatore della distribuzione a posteriori, noto anche come verosimiglianza marginale o evidenza bayesiana;
- la difficoltà di calcolare una data quantità di interesse dalla distribuzione a posteriori (anche se normalizzata) nota.

Sia X una variabile aleatoria con legge di probabilità π definita sul suo supporto $\mathcal{X}$, l'idea di fondo che accomuna quanto visto precedentemente è la stima tramite una media pesata di un integrale:

$$I = \int_{\mathcal{X}} f(x) d\pi \tag{1.1}$$

Definendo N la *numerosità Monte Carlo*, ovvero il numero di campioni simulati dalla distribuzione a posteriori, è possibile stimare la misura di probabilità utilizzata in (1.1) per integrare $f(x)$ attraverso le frequenze relative empiriche di ciascun evento in $\mathcal{X}$, osservate nei campioni simulati: $x_1, \dots, x_N \overset{id}{\sim} \pi$. Lo stimatore risultante è

$$\hat{I} = \int_{\mathcal{X}} f(x) \frac{N_x}{N},$$

dove N_x è la frequenza relativa di x nei campioni Monte Carlo. L'integrazione per ogni elemento di $\mathcal{X}$ viene sostituita da una più semplice sommatoria, dal momento che solo per alcuni x si osserva un N_x diverso da zero. È facile dimostrare che in questo modo $\hat{I}$ corrisponde esattamente alla media campionaria della sequenza di

[1] Un caso particolare si riscontra quando i campioni risultano essere anti-correlati, tuttavia ciò comporta un miglioramento nelle stime solo se la quantità su cui si vuole fare inferenza è una funzione monotona della variabile in esame.

valori simulati:

$$\hat{I} = \frac{1}{N} \sum_{i=1}^{N} f(x_i) \tag{1.2}$$

Tale procedura, nota con il nome di *Integrazione Monte Carlo*, non è l'unico modo con cui è possibile approssimare (1.1). Tra le alternative più quotate ci sono ad esempio i metodi di integrazione numerica attraverso quadratura. Questi, riscrivono I come una somma pesata della funzione calcolata in una griglia di punti,

$$\int_{\mathcal{X}} f(x)dx \approx \sum_{j=1}^{n} w_j f(x_j),$$

dove il numero di punti di quadratura: n, la loro ubicazione: $x_1, \ldots, x_n$, e i rispettivi pesi assegnati: $w_1, \ldots, w_n$, definiscono il metodo e la sua efficacia. La sensibilità a quanto fine è la griglia di punti su cui la funzione è calcolata, rende questi metodi degli strumenti non affidabili nei contesti caratterizzati da alta dimensionalità, questa è una conseguenza del fenomeno noto come *maledizione della dimensionalità* che in questo caso ha un impatto duplice sull'efficienza dello stimatore $\hat{I}$:

- al crescere di d, la dimensione dello spazio di integrazione, il numero di punti in cui calcolare $f(x)$ per garantirne una bontà di approssimazione analoga cresce esponenzialmente, aumentando irrimediabilmente il costo computazionale;

- al crescere di d, la densità di probabilità definita su tale spazio, π, si fa sempre più concentrata rendendo irrilevante il contributo che zone remote dello spazio hanno nel calcolo dell'integrale. Questo comportamento degli spazi di probabilità è formalizzato in Betancourt (2017, paragrafo 1.4) dal concetto di *Typical Set*. Riassumibile come quel sottospazio dello spazio di probabilità considerato in cui si concentra la maggior parte della massa di probabilità.

  Definendo come $TS(\mathcal{X})$ tale insieme, si osserva che:

$$\int_{\mathcal{X}} f(x)d\pi \approx \int_{TS(\mathcal{X})} f(x)d\pi$$

È interessante notare come il secondo punto sembri contraddire il primo, ma questo è dovuto al fatto che esso vale soltanto per le leggi di probabilità, mentre il

primo è una caratteristica generale della *maledizione della dimensionalità*. Infatti, i metodi di quadratura o approssimazione numerica applicati in questi contesti non perderebbero di efficienza se i punti venissero posizionati solamente in corrispondenza del *Typical Set*. Dal momento che il volume di questo tende a ridursi con la dimensionalità, non è più necessaria una griglia più fine, ma è sufficiente comprimere la sua estensione nello spazio più grande. La vera problematica deriva dal fatto che questa contromisura risulterebbe applicabile solo se si conoscesse l'ubicazione e l'ampiezza di tale insieme. Il vantaggio dei metodi di simulazione risiede esattamente in questo: simulando da π, essi forniscono allo stesso tempo un modo per identificare l'ubicazione del *Typical Set*, esplorare tale insieme e ottenere una stima $\hat{I}$ che calcola la quantità di interesse, $f(x)$, solo all'interno di esso, facendone poi la media.

La convergenza al crescere di N dello stimatore definito in (1.2) alla quantità di riferimento, enunciata in (1.1), viene garantita dalla validità del teorema ergodico, le cui assunzioni riguardano le proprietà di ricorrenza di Harris e aperiodicità, come riportato ad esempio in Robert et al. (1999, Capitolo 6). Tali proprietà dipendono da:

- la legge di probabilità marginale di partenza, π_0;
- la complessità della distribuzione obiettivo, π;
- il *nucleo* di transizione markoviano, $K(\cdot \,|\, \cdot)$, immaginabile come la legge che definisce la probabilità di transitare da un punto ad un altro di $\mathcal{X}$, e il suo rapporto con π.

Ciascuna iterazione della catena definisce una nuova legge di probabilità marginale come l'integrale del nucleo di transizione rispetto all'attuale distribuzione:

$$\pi_{j+1}(A) = \int_{\mathcal{X}} K(A|x) d\pi_j \, , \, \forall A \in \mathcal{B}$$

dove $\mathcal{B}$ è la σ-algebra dello spazio campionario ed A un suo elemento. $K(\cdot \,|\, \cdot)$ viene tipicamente assunto essere costante ad ogni iterazione[2]. Tale aggiornamento

[2]In tal caso si parla di catene omogenee.

va ripetuto fino a quando la distribuzione marginale non risulti essere *invariante* rispetto al nucleo di transizione. Attraverso un opportuna scelta di $K(\cdot \mid \cdot)$ si è in grado di far corrispondere tale distribuzione invariante alla distribuzione obiettivo.

Concretamente, salvo casi particolari come ad esempio il filto di Kalman gaussiano (Kalman, 1960), non è possibile calcolare in forma esplicita π_{j+1} ed è qui che entra in gioco l'approccio simulativo, che mette insieme le catene di Markov con i metodi Monte Carlo. Definendo:

$$x \sim \pi_j \quad \text{e} \quad x' \sim K(x' \mid x)$$

la loro legge congiunta è data da $\pi_j(x', x) = K(x' \mid x)\pi_j(x)$ mentre, per definizione, la legge marginale di x' è data da:

$$\pi_{j+1}(x') = \int_{\mathcal{X}} K(x' \mid x)\pi_j(x)dx$$

Tuttavia, non è necessario risolvere nessun integrale per ottenere un campione da tale legge di probabilità. Infatti, dal momento che (x, x') è a tutti gli effetti una realizzazione della legge congiunta $\pi_j(x', x)$, ne consegue che x' è una realizzazione della sua legge marginale. Se tale legge corrisponde a π, allora si è in grado di simulare da essa.

Conseguentemente, quello che si fa è generare dei valori provenienti da questo processo stocastico caratterizzato da 3 fasi:

1. una prima fase in cui la catena ricerca il *Typical Set* all'interno dello spazio di probabilità e il processo generatore converge marginalmente alla distribuzione obiettivo. Questa è anche detta fase di *warm-up* ed i campioni che produce sono inutilizzabili in quanto non rappresentativi della distribuzione invariante;

2. una seconda fase in cui la catena esplora il *Typical Set* e la varianza delle stime Monte Carlo risultanti decresce ad un tasso molto veloce;

3. una terza fase in cui la catena rifinisce l'esplorazione del *Typical Set*, caratterizzata da una riduzione della varianza delle stime Monte Carlo più lenta;

L'efficienza del risultante stimatore Monte Carlo dipende dal numero di iterazioni della catena e dalla sua capacità di esplorare lo spazio campionario, più precisamente il *Typical Set*. La qualità della procedura Monte Carlo dipende quindi dalla quantità di informazione che riesce ad estrarre ad ogni sua iterazione, dove con informazione si intende una misura di quanto ricevere un nuovo dato aumenti la conoscenza riguardo la legge di probabilità. Proseguendo il campionamento l'informazione estraibile man mano si riduce e la velocità con cui la catena si sposta nello spazio ad ogni iterazione stabilisce precisamente di quanto. Una misura dell'efficienza di un algoritmo MCMC è quindi fornita dall'auto-correlazione (di primo ordine) della funzione di interesse:

$$\rho(f) = \frac{\int_{\mathcal{X}} f(x) \cdot f(x') d\pi(x', x) - \int_{\mathcal{X}} f(x) d\pi(x) \cdot \int_{\mathcal{X}} f(x') d\pi(x')}{\int_{\mathcal{X}} f^2(x) d\pi(x) - (\int_{\mathcal{X}} f(x) d\pi(x))^2}$$

Tale quantità è sempre compresa tra -1 e 1 ed è un indice della velocità di esplorazione. Valori di ρ vicini a 0 implicano una velocità elevata, paragonabile ad un campionamento indipendente. Valori vicini a 1 implicano una lenta esplorazione, mentre valori negativi si osservano in presenza di campioni anti-correlati. In tal caso le proprietà dello stimatore dipendono dalla funzione f in questione. Siccome le prestazioni dei metodi MCMC vanno valutate per una f generica, un approccio conservativo non vede di buon occhio scostamenti sia positivi che negativi dallo zero.

Come mostrato in Betancourt et al. (2017, paragrafo 1.2), i metodi utilizzati per definire un nucleo di transizione, che sia invariante alla distribuzione obiettivo, generalmente partono dalla definizione di una collezione di mappe preservatrici della misura di probabilità in questione, ovvero che applicano una trasformata biettiva allo spazio di probabilità originale senza alterare il volume di un qualsiasi sottoinsieme dello spazio originale[3] e la probabilità ad esso associata:

$$t_\theta : \mathcal{X} \to \mathcal{X}, \forall t \in \Gamma$$

dove Γ è da interpretare come lo spazio campionario di tali mappe, a cui è possibile associare una σ-algebra $\mathcal{G}$ e una misura di probabilità $\gamma(t; \theta)$. Il pedice θ segnala una

[3]Nel caso continuo, se il determinante dello Jacobiano della trasformata è pari ad 1.

qualche forma di dipendenza della mappa rispetto ad alcuni parametri. Indicando con $\mathcal{I}(\cdot)$ la funzione identità:

$$\mathcal{I}(E) = \begin{cases} 1 & \text{se si verifica l'evento } E \\ 0 & \text{altrimenti} \end{cases}$$

è possibile definire il nucleo di transizione come:

$$K(A \,|\, x) = \int_{\Gamma} \gamma(t;\theta)\mathcal{I}(t(x) \in A)dt \tag{1.3}$$

Siccome questa è una convoluzione di mappe preservatrici della misura di probabilità originale, allora anche il nucleo risultante preserverà tale misura. I diversi algoritmi MCMC differiscono dunque nella scelta di t_θ e, all'interno della famiglia di mappe risultanti, del valore del parametro θ, il cui scopo è principalmente quello di regolare la velocità con cui la catena di Markov generata esplorerà lo spazio campionario (più precisamente il *Typical Set*).

Sotto questa dicitura è possibile riassumere brevemente i principali metodi MCMC noti in letteratura.

1.1 Metropolis-Hastings

Questo metodo, descritto originariamente in Hastings (1970), racchiude la maggior-parte delle procedure MCMC basate su di un nucleo di transizione reversibile, ovvero la maggior parte di quelle descritte in questa sede. È riassumibile dal seguente schema:

$$\begin{aligned} t_{q,\eta} : x \to x'\mathcal{I}\left(\eta < \frac{\pi(x')}{\pi(x)} \cdot \frac{q(x|x')}{q(x'|x)}\right) \\ x' \sim q(x'|x) \\ \eta \sim Unif(0,1) \end{aligned}$$

Di particolare interesse è la quantità:

$$\alpha(x, x') = \min\left\{1, \frac{\pi(x')}{\pi(x)} \cdot \frac{q(x|x')}{q(x'|x)}\right\} \tag{1.4}$$

nota come *tasso di accettazione Metropolis*, che rappresenta la probabilità di transitare dal valore corrente al valore proposto. Per come è costruita, se $q(\cdot|\cdot)$ è simmetrica, la transizione è sempre accettata se fatta verso zone a più alta densità di probabilità, altrimenti lo è solo in maniera probabilistica.

Il nucleo risultante dall'applicazione di (1.3) allo schema definito qui sopra è tale da garantire la proprietà di *dettaglio bilanciato* secondo cui

$$K(x'|x)\pi(x) \stackrel{d}{=} K(x|x')\pi(x'), \tag{1.5}$$

la dimostrazione si può trovare in appendice A.1, questa proprietà assicura l'invarianza del nucleo di transizione rispetto alla distribuzione obiettivo

$$\begin{aligned} \int_{\mathcal{X}} K(x'|x)\pi(x)dx &\stackrel{d}{=} \int_{\mathcal{X}} K(x|x')\pi(x')dx \\ \int_{\mathcal{X}} K(x'|x)\pi(x)dx &\stackrel{d}{=} \pi(x'). \end{aligned} \tag{1.6}$$

La generalità di questa procedura non aiuta a capire però quali siano le mancanze dei metodi classici che hanno fatto nascere la necessità di svilupparne di più sofisticati. Nei paragrafi successivi quindi, sono discusse brevemente alcune più specifiche procedure MCMC, le quali possono essere tutte ricondotte alla fine ad un caso particolare di questa, dove a cambiare è solo la scelta di $q(x'|x)$, detta densità *proposal*.

1.2 Markov Chain Indipendent Sampler

Da non confondere con un vero e proprio Indipenent Sampler, caratterizzato da un $\alpha = 1$ e dalla generazione di campioni indipendenti, è definito da una densità *proposal* che non dipende dal valore corrente della catena. La procedura è riassumibile dal seguente schema:

$$\begin{aligned} t_{q,\eta} : x \to x' \mathcal{I}&\left(\eta < \frac{\pi(x')}{\pi(x)} \cdot \frac{q(x)}{q(x')}\right) \\ x' &\sim q(x') \\ \eta &\sim Unif(0,1) \end{aligned}$$

Una scelta comune per $q(\cdot)$ ricade sulla distribuzione gaussiana, o t multivariata, centrata nel picco della densità π e con delle code più pesanti di quest'ultima. Per ottenere ciò, si può porre la matrice di varianze e covarianze della densità proposta pari ad una stima inflazionata del corrispettivo relativo alla distribuzione obiettivo. Questo risulta necessario se non si vuole inserire della distorsione nelle stime Monte Carlo risultanti dall'applicazione del metodo, conseguenza di una esplorazione inadeguata delle code di π. Lo svantaggio principale di un tale approccio riguarda l'inefficacia del metodo nei contesti in cui l'approssimazione gaussiana della distribuzione obiettivo viene meno e quindi la sua limitatezza a problemi relativamente semplici.

1.3 Random Walk Metropolis

Originariamente discusso in Metropolis et al. (1953), ne esistono due versioni principali. La prima consiste in un aggiornamento congiunto delle coordinate della catena:

$$
\begin{aligned}
t_{\epsilon,\eta} :\ & x \to x + \epsilon \mathcal{I}\left(\eta < \frac{\pi(x+\epsilon)}{\pi(x)}\right) \\
& \epsilon \sim N_d(0, \Sigma) \\
& \eta \sim Unif(0,1)
\end{aligned}
$$

La seconda consiste in un aggiornamento coordinata per coordinata

$$
\begin{aligned}
t_{i,\epsilon,\eta} :\ & x_i \to x_i + \epsilon \mathcal{I}\left(\eta < \frac{\pi(x+\epsilon \cdot e_i)}{\pi(x)}\right) \\
& i \sim Unif(\{1, \dots, d\}) \\
& \epsilon \sim N(0, \sigma_i) \\
& \eta \sim Unif(0,1)
\end{aligned}
$$

dove d indica la dimensionalità dello spazio campionario. In quest'ultimo caso, sia per ridurre l'auto-correlazione nelle catene che per evitare un allocazione eccessiva di memoria, si è soliti scartare tutte le iterazioni intermedie e tenere soltanto l'ultima.

Questo tuttavia presuppone un campionamento casuale senza reinserimento dei pedici i oppure una loro sequenza deterministica. Nel primo caso è stato dimostrato che questo non altera la proprietà (1.5), mentre nel secondo caso, invece, questa non è rispettata ma resta comunque valida l'invarianza rispetto alla distribuzione obiettivo, come riportato in Gilks et al. (1995, 51-52).

Il problema principale di questo tipo di MCMC riguarda la vicinanza del valore proposto da $q(\cdot|x)$ rispetto a quello corrente della catena. Come riportato sopra, la distribuzione proposta tipicamente corrisponde ad una densità gaussiana centrata nel valore corrente della catena e con una matrice di varianze e covarianze Σ da specificare. Nel migliore dei casi, quando la distribuzione obiettivo rispetta alcune condizioni di regolarità ed è approssimabile da una gaussiana multivariata, la scelta di Σ per $q(\cdot|x)$ ricade sulla controparte per π stimata. Così facendo, il *Typical Set* della *proposal* seguirà la forma di quello della distribuzione obiettivo, ma risulterà essere centrato nel valore corrente della catena. Tale matrice viene poi riscalata per un fattore, tipicamente minore di 1, che rende più locale la *proposal*. Questo risulta necessario per garantire un tasso di accettazione (1.4) vicino al valore ottimo nominale. Tale valore si aggira intorno a 0.4 per le distribuzioni obiettivo multivariate regolari (Roberts e Rosenthal, 2001) e rappresenta un compromesso tra una catena estremamente locale, che garantisce valori alti di α ad ogni iterazione ma lenta ad esplorare il *Typical Set* ed una che estende maggiormente il raggio di ricerca di valori plausibili, ma che è tipicamente caratterizzata da valori più bassi di α. In entrambi gli estremi, o per l'esplorazione estremamente locale, o per una ricerca talmente diffusa che risulta in un comportamento stanziale, l'auto-correlazione della catena risulta alta e l'informazione effettiva contenuta in essa molto minore di quella che ci si aspetterebbe da un campione di valori indipendenti di egual dimensione. Tutto ciò va ad inflazionare la varianza delle stime Monte Carlo.

1.4 Guided Walk Metropolis

Una delle tante versioni alternative del Random Walk Metropolis, proposta in Gustafson (1998): applica una *data augmentation* dello spazio campionario con delle variabili che tengono conto dell'orientamento, invertendolo rispetto a ciascuna coordinata ogni qualvolta venga rifiutata la rispettiva proposta fatta da $q(\cdot|x)$.

Definendo $x' = (x, \rho)$ con ρ un vettore aleatorio d-dimensionale inizializzato casualmente come $\rho_0 \sim Unif(\{-1, 1\}^d)$, è possibile riassumere la procedura come:

$$t_{i,\epsilon,\eta} : x'_i \to \left(x + \epsilon\rho_i \mathcal{I}\left(\eta < \frac{\pi(x + \epsilon\rho_i \cdot e_i)}{\pi(x)}\right), \rho_i \cdot (-1)^{\mathcal{I}\left(\eta \geq \frac{\pi(x+\epsilon\rho_i \cdot e_i)}{\pi(x)}\right)} \right)$$

$$i \sim Unif(\{1, \dots, d\})$$

$$\epsilon \sim N^+(0, \sigma_i)$$

$$\eta \sim Unif(0, 1)$$

Seppure ne esista anche una versione che aggiorna le x in blocco, come riportato dall'autore, non risulta essere significativamente migliore della controparte Random Walk.

Questo metodo MCMC è di interesse in questa sede perché rappresenta una versione embrionale dell'Hybrid Monte Carlo, che vedremo in seguito, e fornisce anche un incipit al metodo utilizzato per trattare le leggi di probabilità discontinue, di cui si parlerà sempre successivamente. L'idea di fondo consiste nel fornire alla catena sullo spazio campionario originale, attraverso un espansione di questo, un metodo per guidare la *proposal* verso le zone a più alta densità di probabilità. Anche in questo caso la reversibilità dell'algoritmo è garantita dal fatto che è riconducibile ad un Metropolis-Hastings sullo spazio aumentato. In particolare, è immaginabile come una sequenza di procedure a due passi, con il primo che aggiorna $(x_i, \rho_i) \to (x'_i, -\rho_i)$ in maniera probabilistica e il secondo che nega il segno di ρ_i in maniera deterministica.

1.5 Gibbs Sampler

Originariamente proposto in Geman e Geman (1984), prende il nome dal noto scienziato Americano, considerato uno dei padri della meccanica statistica. È caratterizzato dalla seguente struttura:

$$t_{i,\eta} : x_i \to F_i^{-1}(\eta)$$
$$i \sim Unif(\mathfrak{I})$$
$$\eta \sim Unif(0,1)$$

Dove $F_i(x_i) = \int_{-\infty}^{x_i} \pi(x_i|x_{-i})dx_i$ è la funzione di ripartizione della *full conditional* di x_i rispetto alle altre coordinate x_{-i}, mentre $\mathfrak{I}$ rappresenta un insieme di possibili partizioni dei rispettivi indici: $1, \dots, d$. Con i quindi, si è soliti indicare anche interi blocchi di coordinate e non solo coordinate singole. In genere più questi blocchi sono grandi, più l'efficienza delle stime Monte Carlo è alta. Il caso limite è dato da $i = (1, \dots, d)$ che fa corrispondere il Gibbs Sampler ad un metodo di campionamento indipendente. Anche in questo caso si è soliti definire una sequenza deterministica con cui aggiornare ciascuna coordinata e/o scartare tutti i campioni intermedi.

Qualora non fosse disponibile una formula esplicita per $F_i^{-1}(\cdot)$ è sempre possibile sostituire la corrispondente mappa $t_{i,\eta}$ con quella di un Metropolis-Hastings generico $t_{i,q,\eta}$. In questi casi si parla di algoritmo di tipo Metroplis-within-Gibbs.

È bene notare che le mappe relative al passo Gibbs puro, utilizzate per costruire il nucleo di transizione Markoviano, non dipendono da alcun parametro di regolazione ed hanno un tasso di accettazione α sempre pari ad 1. Questo non è da considerarsi una cosa sempre positiva perché può comportare un'esplorazione lenta della distribuzione congiunta a cui non è possibile porre rimedio se non attraverso una riparametrizzazione del modello. Inoltre, a differenza del Guided Walk Metropolis, il campionamento di η ad ogni iterazione fa perdere l'informazione relativa all'orientamento con cui guidare la *proposal* verso zone a più alta massa di pro-

babilità per cui il movimento della catena nello spazio di probabilità seguirà un andamento in larga parte dettato dal caso.

Capitolo 2

Hybrid Monte Carlo

Hybrid Monte Carlo (Duane et al., 1987), meglio noto come Hamiltonian Monte Carlo, è un tipo di algoritmo MCMC reso popolare nell'ambito del Machine Learning da Neal (2011) e successivamente in contesti più statistici grazie allo sviluppo di STAN (Gelman et al., 2015), un linguaggio di programmazione probabilistico per l'inferenza bayesiana che deve il suo nome a Stanisław Marcin Ulam, uno scienzato polacco che contribuì al progetto Manhattan con l'invenzione dei metodi di integrazione Monte Carlo per la risoluzione degli integrali presenti nella teoria delle reazioni nucleari. Fu lui a coniare il peculiare nome a tali procedure, ispirato dalla passione di suo zio per il gioco d'azzardo. In realtà questi metodi erano stati già utilizzati da Enrico Fermi una decade prima ma all'interno di articoli mai pubblicati. Da questo momento in poi si farà riferimento ad Hybrid (Hamiltonian) Monte Carlo con la sigla HMC.

Questo metodo di MCMC nasce dalla necessità di definire una distribuzione *proposal* che a differenza delle precedenti, a meno del Markov Chain Indipendent Sampler, non sia locale rispetto al valore corrente della catena. Come riportato in Betancourt et al. (2017, paragrafo 1.2), l'idea innovativa dell'HMC consiste nel considerare una particolare mappa t_θ con cui construire il nucleo di transizione Markoviano, che goda della particolare proprietà di allontanarsi dal valore iniziale. Una particolare famiglia di mappe caratterizzate da queste proprietà sono i *flussi*.

Un flusso,

$$\phi_t : \mathcal{X} \to \mathcal{X} \quad \forall t \in R$$

è una famiglia di isomorfismi parametrizzata dal tempo che gode delle seguenti importanti proprietà:

- associatività:

  $$\phi_s \circ \phi_t = \phi_{s+t}$$

  dove $f \circ g$ indica la funzione composta $f(g(\cdot))$. Ovvero seguire il flusso per un tempo s e poi continuare a seguirlo per un tempo t equivale a seguire il flusso per un tempo $t + s$ partendo dall'inizio;

- reversibilità:

  $$\phi_t^{-1} = \phi_{-t}$$

  ovvero è possibile ripercorrere il flusso all'indietro applicandolo con il tempo negato di segno;

- identità:

  $$\phi_0 = Id_{\mathcal{X}}$$

  questa, combinata alla precedente, ci comunica che è possibile ritornare allo stato iniziale applicando il flusso con tempo inverso.

A partire da ciò, un possibile algoritmo MCMC potrebbe essere dunque definito come:

$$t_{\phi,T,\eta} : x \to \begin{cases} \phi_T(x) & \text{se } \eta < \frac{\pi(\phi_T(x))}{\pi(x)} \cdot \frac{\mathbb{P}[\phi_T \circ \phi_T(x) = x]}{\mathbb{P}[\phi_T(x) = \phi_T(x)]} \\ x & \text{altrimenti} \end{cases}$$

$$\eta \sim Unif(0, 1)$$

Un primo problema che insorge riguarda il tasso di accettazione Metropolis:

$$\alpha = \frac{\pi(\phi_T(x))}{\pi(x)} \cdot \frac{\mathbb{P}[\phi_T \circ \phi_T(x) = x]}{\mathbb{P}[\phi_T(x) = \phi_T(x)]}$$

Essendo il flusso, per un dato tempo T fissato, una mappa deterministica, è facile notare come:

$$\mathbb{P}[\phi_T \circ \phi_T(x) = x] = 0 \quad \text{mentre} \quad \mathbb{P}[\phi_T(x) = \phi_T(x)] = 1 \tag{2.1}$$

Ciò porta (1.4) ad essere pari a 0, tuttavia, tale problematica è facilmente aggiragibile definendo una mappa $t_{\phi,T,\eta}$ particolare, la quale, al termine del flusso, inverte il segno del tempo da T a $-T$. Questo è più facile da comprendere sfruttando il parallelismo con il Guided Walk Metropolis descritto nel paragrafo 1.4. Se si considera T come un dato aumentato che tiene conto della direzione del flusso e il cui segno è cambiato deterministicamente ad ogni iterazione della catena, si ottiene:

$$\alpha = \frac{\pi(\phi_T(x))}{\pi(x)} \cdot \frac{\mathbb{P}[\phi_T \circ \phi_{-T}(x) = x]}{\mathbb{P}[\phi_T(x) = \phi_T(x)]} = \frac{\pi(\phi_T(x))}{\pi(x)} \cdot \frac{1}{1}$$

La vera problematica di un approccio simile riguarda la specificazione di un flusso adeguato. Infatti questo dovrebbe

- essere una mappa che preserva la misura di probabilità originale, altrimenti non è assicurata l'invarianza della distribuzione obiettivo rispetto al nucleo di transizione markoviano risultante, formalizzata dall'espressione (1.6);
- garantire una probabilità di accettazione Metropolis: $\alpha = \frac{\pi(\phi_T(x))}{\pi(x)}$, più alta possibile. In questo caso infatti, siccome il flusso per costruzione generalmente si distanzia dal valore di partenza, non ha senso preoccuparsi di avere un tasso di accettazione troppo alto perché qui non è più sinonimo di un'esplorazione locale.

Entrambe queste proprietà sono garantite dall'HMC, ma per farlo, prima, è necessario elevare lo spazio di probabilità originale ad uno spazio di fase.

2.1 Cenni di meccanica classica

Con il termine meccanica, si intende quella branca della fisica che studia il movimento di determinati corpi nello spazio (vedere ad esempio Mazzoldi et al., 1991).

In particolare è qui che si definiscono i concetti di posizione, velocità ed accelerazione di un oggetto e la loro relazione secondo le leggi naturali macroscopiche. Tali entità sono cruciali nel definire la traiettoria che un corpo, soggetto ad alcune forze, è costretto a seguire. Ed è da questi concetti di fisica generale che si deve partire per cercare un flusso che soddisfi le proprietà desiderate per la costruzione dell'HMC.

Definendo Θ uno spazio d-dimensionale continuo, per esempio $\mathbb{R}^d$, in cui ogni punto è identificabile univocamente da un sistema di coordinate definito da un vettore θ, sempre d-dimensionale. Una traiettoria all'interno di questo spazio è una funzione che fa evolvere le coordinate di un punto nel tempo: $\theta(t)$. A tale funzione, che viene assunta essere continua e derivabile almeno 2 volte, $\theta(t) \in \mathbb{C}^2$, si associano spesso 2 quantità di interesse che sono:

- velocità, definita come la derivata prima della posizione: $v(t) = \frac{d\theta(t)}{dt}$;
- accelerazione, definita come la derivata prima della velocità: $a(t) = \frac{dv(t)}{dt}$, ovvero la derivata seconda della posizione.

La risoluzione delle leggi del moto è dunque fornita come la soluzione di un'equazione differenziale del secondo ordine, riscrivibile come un sistema di due equazioni del primo:

$$d\theta = v(t)dt$$
$$dv = a(t)dt$$

Tale soluzione è facilmente trovabile integrando da entrambi i lati e specificando un punto di partenza per la traiettoria e la relativa velocità:

$$\theta(t) = \theta(t_0) + \int_{t_0}^{t} v(s)ds$$
$$v(t) = v(t_0) + \int_{t_0}^{t} a(s)ds$$

Una terza quantità, che tornerà utile successivamente, è quella di momento, o quantità di moto del corpo in un dato istante, definito come la velocità moltiplicata per la massa:

$$m(t) = Mv(t) \tag{2.2}$$

Con M si definisce una matrice $d \times d$ di massa, che per semplicità può essere pensata diagonale, ma in seguito verrà considerato il caso più generale, denso.

Risulta comodo sfuttare la rappresentazione duale che un sistema di coordinate può avere in uno spazio vettoriale. In particolare, conviene pensare alla posizione dell'oggetto in un dato istante come un punto nello spazio Θ, mentre alla velocità ed al momento come a dei vettori che indicano una direzione che evolve anch'essa nel tempo definita rispettivamente nello spazio tangente e cotangente, ovvero quell'insieme di iperpiani rispettivamente tangenti e cotangenti alla traiettoria nel punto definito dalla posizione.

2.1.1 I tre formalismi della meccaninca classica

Con gli strumenti a disposizione è possibile derivare le leggi del moto attraverso 3 formalismi differenti, i quali partono da premesse diverse ma arrivano al medesimo risultato.

Il formalismo newtoniano: parte dal *principio di inerzia*, che stabilisce che un corpo non soggetto a forze, o soggetto ad una somma di forze che si annullano, non subisce variazioni in velocità, ovvero resta fermo se si trova in uno stato di quiete e si muove secondo un moto rettilineo uniforme (con velocità costante) se invece era già in movimento. Definendo F come la somma di tutte le forze che agiscono su un dato corpo, la seconda legge di Newton afferma che queste hanno un effetto solo sull'accelerazione dell'oggetto in questione:

$$F(t) = Ma(t) = M\frac{d^2\theta(t)}{dt} = M\frac{dv(t)}{dt} \tag{2.3}$$

Questa è un equazione differenziale del secondo ordine rispetto alla posizione del corpo, pertanto la legge afferma che se si conoscono la sua massa e tutte le forze che agiscono su di esso, allora è possibile derivare la traiettoria attraverso la risoluzione di tale equazione.

I successivi formalismi spostano il focus del problema non più in termini di for-

ze ma in termini di energia del sistema, E, intesa come la capacità di compiere lavoro. Il lavoro è da intendere quindi come la quantità di energia ΔE, che viene trasferita ad un corpo applicando ad esso un insieme di forze. Formalmente è definito come l'integrale della forza totale lungo la traiettoria nello spazio:

$$W = \Delta E = \int_{\theta(t_0)}^{\theta(t)} F(x)dx \tag{2.4}$$

L'*energia cinetica*, indicata con K, rappresenta l'energia del corpo imputabile al suo essere in movimento. Per la legge di Newton, sappiamo che tale quantità viene modificata ogni qualvolta che si applica ad esso una forza e tale variazione nell'energia cinetica è misurata dal lavoro:

$$W = \Delta K \tag{2.5}$$

Assumendo che ciascuna delle forze agenti sia esclusivamente una *forza conservativa*, ovvero che applica un lavoro nullo, allora è possibile definire il concetto di *energia potenziale*, indicata con U. Tale energia dipende esclusivamente dalla posizione del corpo nello spazio ed è definita come segue:

$$\Delta U = U(t) - U(t_0) = -W = -\int_{\theta(t_0)}^{\theta(t)} F(x)dx$$

Derivando rispetto al tempo sia a sinistra che a destra ne otteniamo un'espressione implicita come soluzione della sequente equazione differenziale:

$$\frac{dU}{d\theta}(t) = -F(t) \tag{2.6}$$

la quale afferma che la forza tende ad opporsi alla direzione nello spazio in cui il corpo aquisisce energia potenziale. Un classico esempio esplicativo è fornito dalla gravità: un oggetto sospeso in aria è dotato di un elevata energia potenziale, ovvero pronta ad essere trasformata in lavoro. Man mano che scende, seguendo il vettore definito dalla forza di gravità che lo spinge verso il basso, tale potenzialità si dissipa.

Un'importante conseguenza implicata da queste definizioni viene formalizzata dalla legge di conservazione dell'energia, imputabile a Kelvin. La quale afferma che:

Legge di conservazione dell'energia: *In un sistema isolato, non soggetto a forze esterne che compiono lavoro sul sistema, e soggetto esclusivamente a forze*

conservative, l'energia è conservata.

Questa è una diretta conseguenza del fatto che il lavoro totale è nullo: $W = 0$ e pertanto la perturbazione dell'energia dovuta alle forze applicate risulta essere nulla: $\Delta E = 0$.

Una più importante conseguenza riguarda il fatto che è possibile sfruttare la relazione comune che energia cinetica e potenziale hanno con il lavoro per ottenere la seguente equazione:

$$\begin{aligned} \Delta E = W = \Delta K = -\Delta U \\ = (K(t) + U(t)) - (K(t_0) + U(t_0)) \end{aligned}$$

e definire l'energia totale come la somma di energia cinetica e potenziale:

$$E = K + U \tag{2.7}$$

Un esempio propedeutico riguarda il moto oscillatorio che uno *skateboarder* fa in una *half-pipe*, una rappresentazione grafica è presente in Figura 2.1.

Figura 2.1: Skater sopra una Halfpipe

Lo *skater* parte da fermo da una delle due estremità in cima. Siccome non è in movimento non possiede energia cinetica, tuttavia dal momento che si trova in una posizione sopraelevata è dotato di energia potenziale, pronta ad essere utilizzata. Non appena inizia a scendere, come si può osservare nella figura di sinistra, la forza di gravità lo farà scivolare verso il centro della *half-pipe* riducendo l'energia potenziale che verrà trasformata in energia cinetica dovuta al suo movimento. Quando arriva al centro (figura centrale) ha esaurito tutta l'energia potenziale, ma l'energia cinetica che ha guadagnato gli permette di continuare il percorso in salita (figura di destra). La legge di conservazione dell'energia ci dice che se si assume il sistema

isolato e caratterizzato solo da forze conservative, dalle quali va ovviamente escluso l'attrito, allora lo scambio tra i due tipi di energia sarà tale da compensarsi sempre e lo *skater* proseguirà la sua traiettoria all'infinito, oscillando su e giù lungo la *half-pipe*. L'evoluzione della posizione dello *skater* nel tempo, seppur ugualmente derivabile dalla legge di Newton basata sulla somma delle forze che agiscono su di esso, in questo caso gravità e resistenza del suolo, risulta talvolta più facile da studiare considerando quest'altro paradigma fondato sull'energia, attraverso il quale sono definiti i seguenti due formalismi:

Il formalismo lagrangiano: parte dalla definizione di *Lagrangiana* del sistema:

$$L(t, \theta, \theta') = K(\theta') - U(\theta)$$

intesa come la differenza tra energia cinetica ed energia potenziale. Tale quantità naturalmente evolve lungo la traiettoria. In questo caso, per semplicità. si è assunto che l'energia cinetica non dipenda dalla posizione, ma solo dalla velocità, cosa non per forza vera. Definendo l'*azione* che il corpo fa nel percorrere la traiettoria dall'inizio alla fine come:

$$A = \int_{t_0}^{T} L(t, \theta(t), \theta'(t))dt$$

tale quantità è un funzionale, ovvero una funzione di funzioni, che si presta ad essere utilizzato come criterio per stabilire la soluzione di un problema *variazionale*. Ovvero un problema di ottimizzazione dove l'argomento non è uno scalare ma una funzione.

Il formalismo lagrangiano afferma che il moto di un corpo in un sistema che conserva l'energia è definito dalla traiettoria che minimizza l'azione, ovvero la somma delle oscillazioni temporali tra l'energia cinetica e quella potenziale. E quindi è ottenibile come:

$$\hat{\theta}(t) = \underset{\theta(t)}{argmin}\, A(\theta(t))$$

Si può definire una qualsiasi traiettoria $\theta(t)$ come un modello additivo funzionale:

$$\theta(t) = \hat{\theta}(t) + \epsilon \cdot \eta(t)$$

dove ϵ è un termine erratico che fa discostare la traiettoria dalla sua versione ottima in base ad una funzione erratica $\eta(t)$. Date le seguenti assunzioni:

- $\eta(t_0) = \eta(T) = 0$, ovvero che si ricerca la funzione ottima tra tutte le traiettorie che partono in $\theta(t_0)$ e arrivano in $\theta(T)$;
- η è una funzione derivabile almeno 2 volte;
- A possiede un punto stazionario in un intorno di $\epsilon = 0$,

per trovare la soluzione a questo problema variazionale è necessario derivare il funzionale rispetto ad ϵ e porlo uguale a zero:

$$\left.\frac{dA}{d\epsilon}\right|_{\epsilon=0} = \left.\frac{d}{d\epsilon}\right|_{\epsilon=0} \int_{t_0}^{T} L(t, \theta(t), \theta'(t)))dt = 0 \tag{2.8}$$

Il risultato finale viene espresso come la soluzione dalla seguente equazione differenziale nota come *equazione di Eulero-Lagrange*:

$$\frac{\partial L}{\partial \hat{\theta}(t)} = \frac{d}{dt}\frac{\partial L}{\partial \hat{\theta}'(t)} \tag{2.9}$$

I passaggi per la sua derivazione sono presenti in appendice A.2.

Dal momento che la *Lagrangiana* è la sottrazione dell'energia potenziale rispetto alla cinetica, si ottiene:

$$-\nabla U(\theta) = \frac{d\nabla K(\theta'(t))}{dt}$$

dove con $\nabla f(x)$ si indica il gradiente di $f(x)$. La soluzione di questa equazione differenziale del secondo ordine restituisce la traiettoria del moto che minimizza l'azione.

Da qui risulta facile ricondursi alla legge di Newton osservando che:

- $-\nabla U(\theta) = F$, come noto da (2.6);
- partendo dalle condizioni (2.3), (2.4) e (2.5), possiamo definire

$$\begin{aligned}\Delta K = K(t) - K(t_0) = \int_{\theta(t_0)}^{\theta(t)} Ma(x)dx = \int_{\theta(t_0)}^{\theta(t)} M\frac{dv}{dx}\frac{dx}{ds}dx = \int_{v(t_0)}^{v(t)} Mvdv \\ = \left.\frac{v^T M v}{2}\right|_{v(t_0)}^{v(t)}\end{aligned}$$

da cui deriva che $K(\theta'(t)) = \frac{\theta'(t)^T M \theta'(t)}{2}$ per cui

$$\frac{d\nabla K(\theta'(t))}{dt} = \frac{dM\theta'(t)}{dt} = Ma(t).$$

Pertanto, il formalismo newtoniano è un caso particolare del formalismo lagrangiano in cui l'energia cinetica ha la seguente forma:

$$K(\theta'(t)) = \frac{\theta'(t)^T M \theta'(t)}{2}$$

Il formalismo hamiltoniano: è del tutto analogo a quello lagrangiano ma invece di definire la traiettoria $\theta(t)$ sullo spazio originale Θ, la definisce sullo spazio di fase $\Theta \times \mathcal{M}$, dove con $\mathcal{M}$ si indica lo spazio campionario dei momenti. Il nuovo spazio aumentato rappresenta l'insieme di tutti i possibili stati in cui il corpo si può trovare in un dato istante di tempo se, partendo dal punto $\theta(t_0)$ seguisse la traiettoria stabilita dalle equazioni del moto cominciando con un velocità iniziale $\theta'(t_0)$ aleatoria, a cui si fa corrispondere il momento $m(t_0)$. Tali traiettorie sono derivabili applicando la *trasformata di Legendre* alle equazioni di Eulero-Lagrange, che ottiene la risoluzione di un equazione differenziale di grado maggiore scrivendola come la soluzione di un sistema di equazioni di grado minore: ridefinendo il momento del corpo nello spazio come

$$m = \frac{\partial L}{\partial \theta'(t)}.$$

È importante notare che, per la definizione della Lagrangiana, esso diventa:

$$m = \frac{\partial K}{\partial \theta'(t)}$$

e una volta definita tale quantità, è possibile sostituirla alle equazioni di Eulero-Lagrange per ottenere il sistema di dimensione aumentata:

$$\begin{cases} \frac{dm}{dt} &= \frac{\partial L}{\partial \theta(t)} \\ m &= \frac{\partial K}{\partial \theta'(t)} \end{cases} = \begin{cases} \frac{dm}{dt} &= -\nabla U(\theta) \\ \frac{d\theta}{dt} &= \nabla K^{-1}(m) \end{cases}$$

Tale definizione di momento risulta uguale a quella data precedentemente in (2.2), solamente nel caso particolare in cui il formalismo di Newton concorda con quello

di Lagrange. In questo caso si ha inoltre che $\theta'(t) = M^{-1}m$ e, a partire da qui, è possibile ridefinire K direttamente in funzione del momento come:

$$K(m) = \frac{m^T M^{-1} m}{2}$$

e le equazioni del moto, che prendono il nome di *equazioni di Hamilton*, diventano le seguenti:

$$\begin{cases} \frac{dm}{dt} &= -\nabla U(\theta) \\ \frac{d\theta}{dt} &= \nabla K(m) \end{cases} \tag{2.10}$$

Da qui, è possibile definire una nuova quantità caratterizzante del sistema in cui il corpo soggetto a forze si muove: l'*Hamiltoniana* del sistema:

$$H(\theta, m) = K(m) + U(\theta) \tag{2.11}$$

che, per (2.7), altro non è che l'energia di questo. Sfruttando il fatto che se questa è conservata allora sarà costante lungo tutta la traiettoria, è possibile verificare la validità delle equazioni del moto trovate, studiando la sua derivata rispetto al tempo:

$$\begin{aligned} \frac{dH(\theta, m)}{dt} &= \frac{dU(\theta)}{d\theta}^T \frac{d\theta}{dt} + \frac{dK(m)}{dm}^T \frac{dm}{dt} \\ &= \nabla U(\theta)^T \nabla K(m) - \nabla K(m)^T \nabla U(\theta) = 0 \end{aligned}$$

Tale risultato ci conferma che seguendo la traiettoria definita dalle equazioni di Hamilton l'energia del sistema (l'Hamiltoniana) è conservata in ogni istante di tempo.

2.2 Cenni di meccanica statistica

La meccanica statistica nasce come tentativo di collegare la meccanica classica alla termodinamica. Le sue fondamenta si basano principalmente sui lavori pubblicati da Maxwell e Boltzmann nella seconda metà del XIX secolo. Successivamente, furono poi Gibbs (1902), e parallelamente Einstein in una serie di articoli, a riorganizzarne i concetti salienti nei primi anni del XX secolo. Il primo per costruire una teoria più

generale, che inglobasse le due materie. Il secondo per dimostrare le fallacie della teoria classica nei contesti microscopici.

Le discrepanze tra la meccanica classica e la termodinamica sono da imputare alla differenza dei sistemi studiati. Nei sistemi classici si presuppone la conoscenza di posizione e velocità di ogni singola entità presente. Nello studio dei sistemi termodinamici si ha a che fare tipicamente con moli di gas, le quali contengono un numero, n, di particelle dell'ordine di 6×10^{23}, noto come numero di Avogadro. Lo studio del loro movimento richiederebbe la risoluzione di n equazioni differenziali. Ciò risulta proibitivo, anche assumendo che le collisioni tra particelle avvengano solamente a coppie o, non realisticamente, non avvengano proprio. Conseguentemente, la termodinamica si limita a studiare sistemi in equilibrio, identificati dalle loro coordinate macroscopiche, quali volume, temperatura, pressione, etc... e descrive come queste entità siano in relazione tra loro attraverso le *equazioni di stato*. Inoltre, al concetto di *lavoro*, noto dalla teoria classica, viene accostato quello di *calore scambiato* tra sistemi in equilibrio. Ovvero, quella quantità che misura l'ammontare di energia trasmessa tra di essi che non è imputabile a movimenti macroscopici, descritti già dalla meccanica classica dal concetto di *lavoro* per l'appunto.

Ad uno stato termodinamico, quindi macroscopico, il quale è identificato dalle sue coordinate, è possibile far corrispondere un numero smisurato di stati meccanici, microscopici. Risulta necessario quindi un approccio statistico, basato sul calcolo delle probabilità, per legare questi due mondi, ed è qui che entra in gioco la meccanica statistica. Per una trattazione più dettagliata sulla meccanica statistica e la sua storia è possibile consultare Darrigol e Renn (2003) mentre, per la comprensione dei concetti alla base dell'HMC, è sufficiente analizzarne gli aspetti salienti solo di una branca di tale materia. In particolare, il focus è incentrato sullo studio dei sistemi chiusi ed in equilibrio statistico, dove si assume che questi siano circondati da un ambiente molto più grande che funge da termostato, mantenendo la temperatura costante e con cui sono esclusi scambi di materia ma sono ammessi solo scambi di energia. Il concetto chiave, coniato in Gibbs (1902), è quello di *ensemble*, che rappresenta l'insieme di tutti i possibili stati in cui il sistema può trovarsi ed

è caratterizzato da una legge di probabilità. Il concetto di *equilibrio statistico* si traduce in una stazionarietà di tale legge, in contrasto con l'*equilibrio meccanico* che invece assumerebbe l'assenza di forze all'interno del sistema e quindi di movimento delle particelle. Tra le possibili leggi di probabilità, due sono quelle importanti per i nostri scopi, il cui utilizzo, assiomatico, è giustificato anche dal principio di massima entropia: la distribuzione canonica e la distribuzione microcanonica.

2.2.1 La distribuzione canonica

Caratterizzante l'*ensemble* canonico, assume che le variazioni di energia del sistema, dovuta ad un ambiente circostante che funge da termostato, siano tali da assegnare ad essa una legge esponenziale:

$$\pi(E) = \frac{1}{Z} e^{-E\beta} \tag{2.12}$$

dove Z è detta *funzione di partizione* e corrisponde alla costante di normalizzazione di $\pi(E)$. Essa ricopre un ruolo molto importante dal momento che, attraverso i suoi cumulanti, è possibile ricavare le coordinate del corrispettivo sistema termodinamico. Inoltre, dal momento che questa è una famiglia esponenziale univariata, questi sono facilmente ottenibili attraverso le derivate rispetto al parametro canonico. Esso, β, è definito come:

$$\beta = \frac{1}{K_B \cdot Temperatura} \quad , \quad \text{con } K_B \text{ costante di Boltzmann}$$

Ragionevolmente, questo ci comunica che sistemi con energia più alta sono più probabili in corrispondenza di temperature alte, inoltre tale parametro è strettamente collegato al concetto di *tempering*, utilizzato anche in statistica bayesiana per svariati scopi quali:

- migliorare le procedure di stima della verosimiglianza marginale, che altro non è che la funzione di partizione, utile per confrontare modelli in ambito bayesiano. Data la vastità di questo argomento, si rimanda ad una *review* dettagliata sui possibili metodi, non per forza basati soltanto sul *tempering*, fornita in Llorente et al. (2023);

- migliorare la convergenza degli algoritmi di MCMC, soprattutto se in presenza di multimodalità. L'idea alla base consiste nell'applicare una potenza alla propria legge di probabilità che, se minore di 1 e quindi in corrispondenza di temperature alte, renda la corrispondente densità più platicurtica e riduca la *barriera energetica* (una definizione è data in seguito a pagina 37) che impedirebbe alle catene di esplorare le diverse mode. Viceversa, l'utilizzo di una potenza maggiore di 1 implica l'esplorazione di una distribuzione più leptocurtica in corrispondenza di ciascuna moda, molto facile da esplorare ma solo a livello locale. Facendo variare il valore di β è possibile ottenere entrambi i benefici, muovendo le catene tra le diverse mode ed esplorando ciascuna di queste al meglio. Esistono diversi possibili metodi per farlo, i quali differiscono principalmente nello schema che determina il percorso per β. Uno dei metodi più semplici è noto come *parallel tempering*, originariamente proposto in Geyer et al. (1991), trovabile spiegato bene in Watanabe (2018, paragrafo 7.1.3), nel quale ogni iterazione aggiorna diverse catene a temperature differenti, permettendo lo scambio tra queste con una probabilità Metropolis classica. Altri metodi più complicati, invece, prevedono l'utilizzo di una sola catena in cui la temperatura viene fatta variare internamente in una griglia prestabilita (simulated tempering, Marinari e Parisi, 1992) o nel continuo (adiabatic monte carlo, Betancourt, 2014). La complicazione principale di questi due metodi rispetto al precedente è data dalla necessità di dover calcolare il valore della funzione di partizione al variare della temperatura.

Un'ulteriore legame interessante, messo in luce sempre da β, tra la distribuzione canonica e la statistica bayesiana è decritto in LaMont e Wiggins (2019). È facile notare che, ponendo $E = -\log \pi(\theta|y)$ e $\beta = 1$ si ottiene che $\pi(E)$ corrisponde alla distribuzione a posteriori. Alla luce di ciò, è possibile pensare a β come alla numerosità campionaria, la cui crescita rende $\pi(\theta|y)$ sempre più leptocurtica. Conseguentemente, la distribuzione a posteriori di un modello bayesiano è immaginabile come la distribuzione dell'energia di un sistema chiuso la cui temperatura è inversamente proporzionale alla numerosità campionaria, sancendone così un legame stretto con

la termodinamica che permette di ottenere delle interpretazioni concrete rispetto alle quantità derivabili da $\pi(\theta|y)$ e al suo comportamento asintotico.

2.2.2 La distribuzione microcanonica

Caratterizzante l'*ensemble* microcanonico, che assume un insieme isolato, dove non avvengono né scambi di materia né di energia, dove si assegna una distribuzione di probabilità uniforme a tutti gli stati possibili. Tale definizione si presta molto alla descrizione delle traiettorie definite dalle leggi di Hamilton, dal momento che queste si muovono nello spazio di fase lasciando inalterata l'energia, per cui si avrà:

$$\pi(\theta, m|E) = \frac{1}{\int_{H^{-1}(E)} d\theta dm} \tag{2.13}$$

dove $H^{-1}(E) = \{\theta, m : H(\theta, m) = E\}$

2.3 HMC nel concreto

Dato un modello statistico bayesiano:

$$\mathcal{F} = \left\{ y \in \mathcal{Y}, \theta \in \Theta \subseteq \mathbb{R}^d : \mathbb{P}(y|\theta) = \pi(y|\theta)\,,\, \mathbb{P}(\theta) = \pi(\theta) \right\}$$

l'obiettivo è calcolare:

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

di cui il denominatore è spesso ignoto. Indicando con $\pi(\theta|y) = \pi(y|\theta)\pi(\theta)$ la densità a posteriori non normalizzata, è possibile definire l'energia potenziale come:

$$U(\theta) = -\log \pi(\theta|y)$$

La trasformata logaritmica viene applicata per semplificare i calcoli analitici e prevenire problemi computazionali dovuti ad *overflow* e *underflow*. Similmente a quanto accade per la verosimiglianza, essendo $\log(x)$ una funzione monotona crescente, quindi biettiva, è sempre possibile riportarsi alla scala originale senza alterare l'inferenza. L'applicazione del segno negativo invece risulta interessante perché permette di interpretare l'energia potenziale come una funzione di θ che assume valori grandi

quando la probabilità a posteriori è bassa e viceversa se è alta. Dal formalismo newtoniano sappiamo che la forza applicata ad un corpo in uno spazio lo fa muovere nella direzione opposta in cui l'energia potenziale cresce, dunque orientato verso i punti di minimo di $U(\theta)$, che corrispondono ai punti di massimo di $\bar{\pi}(\theta|y)$. Si può immaginare allora che il valore del parametro θ definisca le coordinate di una particella nello spazio di probabilità Θ su cui agisce un'energia potenziale $U(\theta)$, la quale è interpretabile come una forza attrattiva che dipende dalla sua posizione. Il parallelismo più comune, nel caso in cui $\Theta = \mathbb{R}^2$, è quello di un paesaggio montano dove l'altitudine in una data coordinata θ è data da $-\log \pi(\theta|y)$ e la forza di gravità, combinata alla resistenza del suolo, costituisce un vettore che agisce nella direzione opposta alla pendenza del terreno: $\nabla U(\theta)$. Qui è possibile pensare alla particella come ad una slitta sulle montagne innevate che segue le leggi del moto sotto l'assunzione di assenza di attrito. L'obiettivo dell'HMC è quello di sfruttare la simulazione di questo sistema fisico per definire una *proposal*. In questo modo, si è in grado di distanziare il valore proposto del proprio algoritmo MCMC da quello corrente per ogni iterazione della catena, quindi di esplorare rapidamente il *Typical Set*. Come riportato in Betancourt (2017, paragrafo 3.1), questo funziona grazie al fatto che il campo vettoriale definito dalle equazioni del moto è allineato ad esso. A differenza del campo vettoriale definito solo da $\nabla U(\theta)$, che punta in direzione solo esclusivamente della/e moda/e a posteriori, la traiettoria così definita permette alla slitta di continuare a muoversi anche una volta raggiunto il fondo della vallata. Questa è una diretta conseguenza della legge di conservazione dell'energia citata in 2.1.1, che impone all'energia cinetica di crescere quando quella potenziale diminuisce e viceversa. Ciò permette allo slittino di avere il pieno di velocità quando si trova a valle e gli consente così di sfruttarla per salire sull'altro versante.

Per studiare il moto di questa particella nello spazio, attraverso le equazioni di Hamilton, è necessario conoscere oltre alla posizione θ, la sua velocità iniziale, o meglio il suo momento: m. Siccome l'obiettivo inferenziale non riguarda quest'ultimo, esso può essere generato in maniera arbitraria da una *funzione di disintegrazione*:

$$m \sim \xi$$

e, in maniera analoga rispetto a θ, si definisce la funzione per l'energia cinetica come il negativo del logaritmo di questa:

$$K(m) = -\log \xi(m)$$

Adesso lo spazio di probabilità originale, Θ, è stato sopra-elevato ad uno spazio di fase: $\Theta \times \mathcal{M}$, dove $\mathcal{M}$ è il dominio del momento, posto pari a $\mathbb{R}^d$.

La nuova legge di probabilità su tale spazio è data da:

$$\pi(m)\bar{\pi}(\theta|y) \propto \pi(m)\pi(\theta|y) = e^{-[U(\theta)+K(m)]}$$

che per la definizione di *Hamiltoniana* data in (2.11) si può riscrivere come:

$$\pi(\theta, m|y) \propto e^{-H(\theta,m)}$$

Ricordandosi che $H(\theta, m)$ indica l'energia del sistema, E, ci si accorge di stare facendo inferenza sulla distribuzione canonica di un sistema termodinamico chiuso (2.12), dove si assume che la temperatura sia pari all'inversa della costante di Boltzmann, ovvero che $\beta = 1$. Inoltre, dal momento che lungo le traiettorie definite dalle equazioni di Hamilton l'energia è conservata, la legge di probabilità assume valore costante in questi tratti e partiziona così la $\pi(E)$ in tante distribuzioni condizionate ai livelli di energia, $\pi(\theta, m|E)$, per costruzione uniformi. Tali densità corrispondono quindi a delle distribuzioni microcanoniche di un sistema termodinamico isolato (2.13).

Questa partizione è osservabile in Figura 2.2, dove sono evidenziate le curve di livello della distribuzione canonica corrispondente ad una distribuzione a posteriori bimodale univariata con funzione di disintegrazione gaussiana. Queste definiscono delle traiettorie, caratterizzate da energia costante lungo lo spazio di fase, che la particella sarebbe costretta a seguire se si trovasse a giacere su una di loro. È possibile classificarne di due tipi differenti:

- traiettorie critiche, in rosso, caratterizzate da un andamento non ciclico e da energie molto alte (basse probabilità a posteriori per (θ, m));
- traiettorie non critiche, in blu, cicliche e con livelli di energia medio bassi.

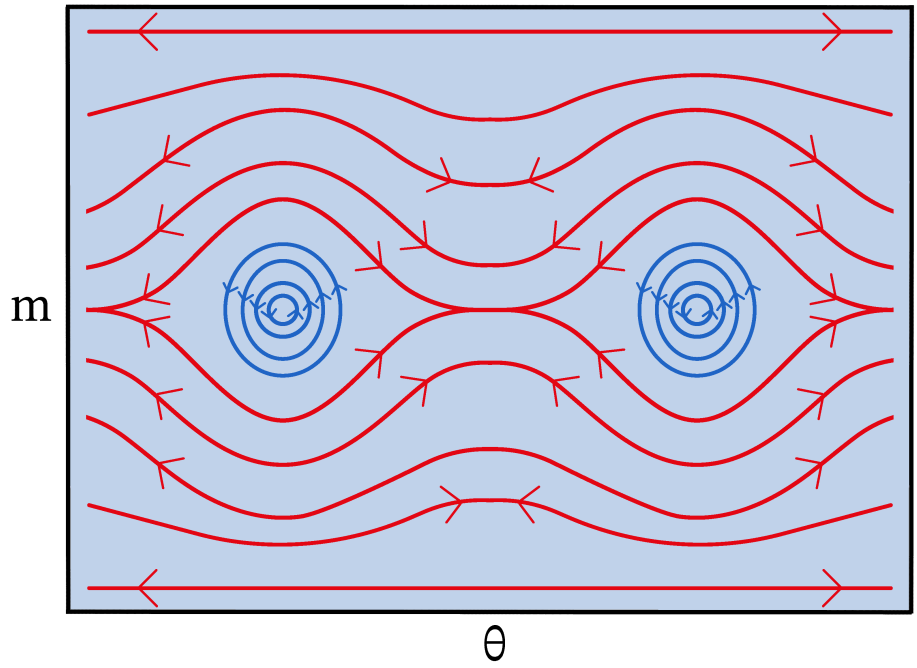

Figura 2.2: Livelli di energia costante nello spazio di fase definito da un energia potenziale bimodale.

Alla luce di ciò è ancora più chiara l'affermazione secondo cui *le equazioni di Hamilton definiscono un campo vettoriale allineato al Typical Set* e, in particolare, come questa risulti vera solo per le traiettorie non critiche. Infatti, se se ne imbocca una critica, questa allontanerà la particella dalle zone ad alta densità di probabilità per un tempo indefinito. Fortunatamente, tali curve di livello sono rese accessibili soltanto se il sistema microcanonico è dotato di un'energia elevata. Tornando al parallelismo della slitta, questo accadrebbe solo se questa partisse da un punto estremamente alto (con bassisima probabilità a posteriori) o con una velocità di partenza elevata. Il primo caso tipicamente si incontra solamente durante la fase di *warm-up*, cioè quando non si è ancora raggiunto il *Typical Set* e la catena prosegue nella ricerca. Il secondo caso invece dipende dalla funzione di disintegrazione scelta. Un altro aspetto interessante che si deduce da questo grafico riguarda la presenza di una *barriera energetica* che separa le due mode. Questa incomunicabilità tra le mode rappresenta uno dei più grandi difetti dell'HMC, a tal punto che in Mangoubi

et al. (2018) è stato dimostrato avere delle prestazioni addirittura peggiori del Random Walk Metropolis se in presenza di densità a posteriori multimodali. Alcune soluzioni applicabili sono:

- utilizzare i metodi di *tempering* per migliorare il *mixing* delle catene, già descritti nel paragrafo 2.2.1;

- utilizzare una funzione di disintegrazione con code particolarmente pesanti, per favorire il campionamento di momenti elevati che permettano il superamento delle barriere energetiche. Nell'esempio della slitta questo equivale a fornire ogni tanto un momento sufficientemente grande che permetta di risalire una collina, superare un altopiano e poi riscendere per un altra vallata. Il problema di questo approccio, come puntualizzato in Betancourt (2016*a*), risiede nel fatto che la risultante funzione di energia cinetica è caratterizzata da un gradiente tendenzialmente più piccolo. Questo, per come sono definite le equazioni di Hamilton, comporta una più lenta esplorazione delle distribuzioni microcanoniche, a vantaggio di una più veloce esplorazione della distribuzione canonica, che impatta l'efficienza dell'HMC. Inoltre, le traiettorie critiche spesso non sono veramente esplorabili perché il calcolo necessario alla loro risoluzione può essere afflitto da errori numerici anche molto grandi;

- modificare il sistema di riferimento. Una variante interessante, proposta in Vishwanath e Tak (2024), consiste nel considerare, per ogni iterazione dell'algoritmo, un sistema fisico caratterizzato da una forza di attrito inizialmente repellente, che faccia accelerare ulteriormente la slitta man mano che scende, aumentando l'energia del sistema e permettendo così di superare le barriere energetiche, e successivamente attraente, che faccia rallentare la slitta in discesa, riducendo l'energia del sistema come farebbe un normale attrito. Viene permesso così all'HMC di far saltare la particella da una moda ad un altra ad ogni iterazione.

2.3.1 Proprietà del flusso hamiltoniano

Definiamo con ϕ_t^H il flusso hamiltoniano, ovvero quella mappa che delinea la traiettoria che parte da $z_0 = (\theta_0, m_0)$ ed arriva fino ad un altro punto nello spazio di fase indicato con $z_t = (\theta_t, m_t)$. Le equazioni di Hamilton che lo descrivono sono riscrivibili in forma compatta come

$$\frac{dz_t}{dt} = \Omega \nabla_z H(z_t) \quad \text{, dove } \Omega = \begin{bmatrix} 0 & I_d \\ -I_d & 0 \end{bmatrix} \text{ e } \nabla_z H(z_t) = \begin{pmatrix} \nabla U(\theta_t) \\ \nabla K(m_t) \end{pmatrix}.$$

Analogamente a quanto discusso a pagina 22, essendo questo una trasformata deterministica, per assicurare che il tasso di accettazione di Metropolis della risultante *proposal* risulti diverso da zero, è necessario applicare alla conclusione del flusso un operatore che ne inverta la direzione. Nel caso dell'HMC la direzione in cui si muove la particella in ogni istante di tempo è definita dal suo vettore dei momenti m_t. Per invertirla, dunque, è sufficiente applicare ad esso una trasformata che ne cambi il segno: $-I_d$. Qui si può notare un parallelismo con il Guided Walk Metropolis descritto nel paragrafo 1.4, dove anche qui ad ogni coordinata nello spazio originale veniva associata una variabile ausiliaria che tenesse conto dell'orientamento e a cui era applicato un cambiamento di segno deterministico per ricondurlo ad un nucleo Metropolis-Hastings. La differenza sostanziale è che nell'HMC queste variabili ausiliarie sono definite in $\mathbb{R}^d$ anziché $\{-1, 1\}^d$, permettendo così non solo di specificare l'orientamento in cui la catena deve muoversi, ma anche la direzione, che viene stabilita seguendo le leggi del moto definite in (2.10). .

Definendo l'operatore *cambio di segno* come

$$F = \begin{bmatrix} I_d & 0 \\ 0 & -I_d \end{bmatrix}, \tag{2.14}$$

allora la mappa nello spazio di fase che pone le fondamenta per il nucleo di transizione markoviano dell'HMC è scrivibile come

$$t : z_0 \to F \circ \phi_t^H(z_0).$$

È di interesse sapere se tale mappa rispetti gli obiettivi preposti alla fine del paragrafo 2, ovvero se preservi la misura e se sia dotato di un tasso di accettazione sufficientemente alto. La risposta è si ad entrambe, come conseguenza di quattro sue proprietà degne di nota:

1. **Reversibilità:** Grazie alle proprietà delle equazioni differenziali, che una volta risolte definiscono la dinamica della traiettoria sia in avanti che all'indietro nel tempo, si ha che $\phi^H_{-t} \circ \phi^H_t = Id_{\Theta \times \mathcal{M}}$. Questo garantisce che

$$(F \circ \phi^H_t) \circ (F \circ \phi^H_t)(z) = z \quad , \forall z \in \Theta \times \mathcal{M}$$

   e che quindi $\alpha(z, z') \neq 0$, assicurando la condizione (1.5) del corrispettivo nucleo nella sua rappresentazione alla Metropolis-Hastings.

2. **Conservazione dell'Energia:** Per defizione, la traiettoria delle equazioni di Hamilton si muove su curve di livello ad energia costante. Questo ha l'importante conseguenza che, siccome la nostra distribuzione obiettivo è diventata la distribuzione canonica, il rapporto tra la densità del valore proposto e la densità del valore corrente è la stessa. Questo comporta un $\alpha(z, z') = 1$, che è il più alto possibile. Ciò rende l'HMC simile ad un Gibbs Sampler dove la procedura di campionamento avviene in tre fasi:

   (a) Simulazione del momento: $m \sim \xi$, dove ξ è interpretabile come la *full conditional* di $m_t|\theta_{t-1}, E_{t-1}$, che in realtà è incondizionata.

   (b) Definizione del livello di energia corrente $E_t = H(\theta_{t-1}, m_t)$, interpretabile come una *full conditional* deterministica per $E_t|m_t, \theta_{t-1}$.

   (c) Campionamento uniforme o deterministico lungo la traiettoria di $\phi_t(\theta_{t-1}, m_t)$, interpretabile come la *full conditional* per $\theta_t, m'_t|E_t$.

   Sfortunatamente questo richiede la risoluzione esatta delle equazioni di Hamilton, che vedremo non essere quasi mai davvero disponibile.

3. **Preservazione del Volume:** Ovvero che tale mappa t implichi una trasformazione che non altera il volume dello spazio, quindi con un determinante

dello Jacobiano pari ad 1. Questo è dimostrabile in diversi modi, ad esempio sfruttando il concetto di divergenza nulla del campo vettoriale generato dalle equazioni di Hamilton, ma un approccio più semplice è considerato in Neal (2011) e parte da un'espansione di Taylor al primo ordine per ϕ_t^H:

$$\begin{pmatrix} \theta_{t+\epsilon} \\ m_{t+\epsilon} \end{pmatrix} = \begin{pmatrix} \theta_t \\ m_t \end{pmatrix} + \epsilon \begin{pmatrix} d\theta_t/dt \\ dm_t/dt \end{pmatrix} + O(\epsilon^2) \tag{2.15}$$

La dimostrazione è riportata in appendice A.3

4. **Simpletticità:** Tale proprietà caratterizza i flussi di ogni sistema hamiltoniano ed implica la preservazione del volume. È formalizzabile dalla seguente condizione:

$$J^T \Omega^{-1} J = \Omega^{-1}$$

In particolare Ω può essere una qualsiasi matrice con determinante non nullo e per cui $\Omega^{-1} = \Omega^T$. Questa generalizzazione serve, ad esempio, per incorporare l'attrito all'interno del sistema fisico, come proposto in Vishwanath e Tak (2024) e modificare le proprietà dell'HMC. I veri benefici della simpletticità riguardano i casi in cui non è possibile risolvere esattamente le equazioni del moto. In tal caso, utilizzarne una versione approssimata che gode di simpletticità, assicura che l'errore di approssimazione resti piccolo anche per traiettorie molto lunghe, come dimostrato in McLachlan et al. (2004)

2.3.2 Gradi di libertà dell'HMC

L'HMC descritto nel paragrafo precedente nella realtà richiede la specificazione di diversi gradi di libertà. Ciò deriva principalmente dall'impossibilità di risolvere esattamente le equazioni di Hamilton per una generica energia potenziale (negativo del logaritmo della distribuzione a posteriori), che può essere affrontato in vari modi, più o meno efficienti.

1. **Scelta della funzione di disintegrazione:** In Betancourt (2017) viene motivato l'utilizzo di una distribuzione gaussiana:

$$m \sim N_d(0, M)$$

dove M è la matrice di massa che compariva in (2.2). L'argomento a favore è basato sul fatto che alla fine l'HMC può essere visto come un *Random Walk* sui livelli di energia, le cui prestazioni sono imputabili esclusivamente alla scelta di ξ e di come questa interagisce con $\pi(\theta|y)$. L'asintoto a cui ogni algoritmo di MCMC tende è di comportarsi come un *Indipendent Sampler*. Ovvero che la *proposal* sia uguale alla distribuzione obiettivo. In questo caso, si vorrebbe quindi che la distribuzione del livello di energia condizionata al valore corrente di θ,

$$\pi(E|\theta) = \int_{\mathcal{M}} \pi\left(\ H(\theta, m) = E \ \mid\ \theta, m\right) d\pi(m),$$

risultasse più simile possibile alla distribuzione marginale dell'energia, $\pi(E)$, ottenuta marginalizzando anche θ rispetto alla sua distribuzione a posteriori. Dal momento che l'energia è ottenuta da un agglomerato di tutti i parametri dello spazio di fase, questa, soprattutto in contesti di alta dimensionalità, tende a seguire il teorema del limite centrale, assumendo una forma gaussiana.

In Betancourt (2016*a*) sono proposti dei criteri per diagnosticare una scelta sub-ottimale di ξ.

- Confrontare le densità di $\pi(E) - \mathbb{E}[E]$ e di $\pi(\Delta E)$ dove:

  $$\pi(\Delta E) = \int_{\Theta} \pi(\Delta E|\theta) d\pi(\theta|y)$$

  Da qui si nota come sia esclusivamente ξ ad essere rilevante, dal momento che $\Delta E|\theta = \Delta K$, per via del fatto che l'energia potenziale non cambia ne prima ne dopo. Seppure non direttamente calcolabili ne $\pi(E)$ ne $\pi(\Delta E)$, queste sono facilmente stimabili a partire dai campioni Monte Carlo. Se la densità marginale risulta essere platicurtica rispetto alla condizionata, si osserva una lenta esplorazione della catena che rende difficile raggiungere i livelli di energia nelle code.

- Riassumere in una statistica la differenza in variabilità tra le due densità appena citate.

Applicando la legge della varianza totale si ottiene:

$$\mathbb{V}(E) = \mathbb{E}_\theta[\mathbb{V}(E|\theta)] + \mathbb{V}_\theta[\mathbb{E}(E|\theta)]$$

e per le proprietà della varianza si ha che:

$$\mathbb{V}(E|\theta) = \mathbb{V}(E - E_\theta|\theta) = V(\Delta E|\theta)$$

dove E_θ rappresenta il valore dell'energia al passo precedente, che si aveva prima di ricampionare il momento, e pertanto costante rispetto alla distribuzione $E|\theta$ per cui si calcola la varianza. Unendo i due risultati precedenti, si può ricavare un indice riassuntivo noto come *Bayesian Fraction of Missing Information*

$$BFMI = \frac{\mathbb{E}_\theta[\mathbb{V}(\Delta E|\theta)]}{\mathbb{V}(E)} \quad \in [0,1] \tag{2.16}$$

Valori grandi indicano una buona copertura del supporto di E da parte della *proposal* $E|\theta$ e quindi una veloce esplorazione dei livelli di energia. Se è piccolo significa che è la variabilità di θ a determinare la distribuzione marginale dell'energia e che quindi ξ non interagisce bene con $\pi(\theta|y)$.

Lo stimatore empirico per (2.16) è ottenibile dai campioni Monte Carlo come:

$$\widehat{BFMI} = \frac{\sum_{i=2}^{N}(E_i - E_{i-1})^2}{\sum_{i=1}^{N}(E_i - \bar{E})^2}$$

il quale spesso assume valori maggiori di 1 per via dell'errore di stima. In genere si considerano problematici valori inferiori a 0.2.

Le problematiche di queste diagnostiche si basano sul fatto che assumono la convergenza delle catene, che è essa stessa inficiata, tipicamente, da una scelta sub-ottimale della funzione di disintegrazione. Una possibile soluzione consisterebbe nella scelta di una ξ con code più pesanti della gaussiana, tuttavia, come già citato all'inizio del capitolo, questo comporta una più rapida esplorazione della distribuzione canonica a discapito però di una più lenta esplorazione della distribuzione microcanonica su ogni traiettoria, rallentando

di molto il tempo di esecuzione dell'algoritmo. Un esempio esplicativo consiste nel confrontare la distribuzione di Laplace con quella di una gaussiana: La velocità della traiettoria di θ è misurata da $\frac{d\theta}{dt} = \nabla K(m)$: nel caso gaussiano questa è pari a m, nel caso di Laplace invece è $\text{sign}(m)$.

All'interno della famiglia gaussiana, c'è ancora possibilità di manovra attraverso la specificazione della matrice di massa M. Questo è un aspetto cruciale perché permette di applicare indirettamente una riparametrizzazione alla distribuzione a posteriori originale: $\pi(\theta|y)$.

È risaputo che il *mixing* dei metodi MCMC in ambito bayesiano è fortemente influenzato dalla parametrizzazione adottata. Classici esempi sono la parametrizzazione non centrata nei modelli gerarchici (Gelfand et al., 1995) o la trasformazione logaritmica delle componenti di varianza e non solo (Roberts e Rosenthal, 2009).

Una riparametrizzazione generalmente comoda consiste nel porre $\theta' = L^T\theta$, dove $\Sigma = LL^T$ è la matrice di varianze e covarianze della distribuzione a posteriori, stimabile in una fase di *warm-up* prolungata in cui si scartano le iterazioni che non hanno ancora raggiunto il *Typical Set*. La nuova energia potenziale è scrivibile come

$$U'(\theta') = U(L^T\theta) + \log\det(L) \tag{2.17}$$

e la traiettoria dei momenti nel nuovo spazio, implicata dalle equazioni di Hamilton diventa

$$\frac{dm'}{dt} = -\frac{dU(L^T\theta)}{d\theta'}$$

mentre quella nello spazio originale è scrivibile come:

$$\frac{dm}{dt} = -\frac{dU(L^T\theta)}{d\theta'}\frac{d\theta'}{d\theta} = -L^T\frac{dm'}{dt}$$

Conseguentemente:

$$\frac{dm'}{dt} = L^{-T}\frac{dm}{dt} \implies \frac{dm'}{dm} = L^{-T}$$

questo significa che possiamo ottenere lo stesso risultato applicando la trasformata inversa all'argomento della funzione di disintegrazione, che, per le proprietà della normale multivariata, corrisponde a fissare $M = \Sigma^{-1}$. L'idea di fondo si basa sul fatto che, invece di riadattare lo spazio di probabilità alla forma della distribuzione a posteriori, è possibile far si che siano i vettori dei momenti ad essere allineati ad essa. Così la traiettoria di θ diventa:

$$\frac{d\theta}{dt} = \nabla K(m) = M^{-1}m = \Sigma m \tag{2.18}$$

In questo modo si assegna maggiore velocità nelle direzioni in cui la distribuzione a posteriori si estende e minore velocità in quelle in cui si contrae. Si riescono così ad esplorare entrambe in maniera ottimale, a patto che l'approssimazione normale di $\pi(\theta|y)$ sia buona ovviamente. Qualora non lo fosse, questo non introduce distorsione alcuna nelle stime Monte Carlo dal momento che la distribuzione obiettivo resta invariata, ma risulterà soltanto un metodo sub-ottimale, spesso comunque efficace.

Un'altra possibilità, circoscritta sempre al caso gaussiano, consiste nel fissare la matrice di massa come l'Hessiano del negativo del logaritmo della distribuzione a posteriori nel valore corrente del parametro:

$$M(\theta) = \nabla^2 U(\theta) \approx \Sigma(\theta)^{-1}$$

In questo modo si ottiene una precisione ancora migliore, data da un movimento con velocità della particella che si adatta alla geometria locale, e si è in grado di esplorare al meglio ogni angolo di Θ in maniera adattiva. L'algoritmo risultante, discusso in Betancourt (2013*b*), prende il nome di *Riemannian*-HMC, o più semplicemente RHMC. Tale titolo deriva dal fatto che $\Sigma(\theta)$ è una metrica riemanniana utilizzabile per misurare la distanza tra i momenti, vettori cotangenti sulla varietà definita dall'energia potenziale. Un accorgimento pratico consiste nell'assicurarsi che tale metrica sia definita positiva, un possibile metodo è descritto in Betancourt (2013*a*). Ciononostante, RHMC risulta estremamente costoso se confrontato con HMC, dal momento

che richiede dei metodi numerici più onerosi per l'approssimazione delle equazioni di Hamilton, basati su sviluppi di Taylor di ordine maggiore. Questa è una conseguenza del fatto che l'energia cinetica, $K(\theta, m)$, adesso dipende anche da θ.

2. **Scelta dell'integratore simplettico:** Un integratore è uno strumento con cui è possibile approssimare la soluzione di un'equazione differenziale per un dato periodo di tempo T. Questi dividono il tempo di integrazione in L intervalli tipicamente equispaziati di ampiezza ϵ, cosicché $T = L\epsilon$. Siccome la soluzione è affetta da una componente erratica, c'è il rischio che questa si protragga e si accumuli man mano che si va avanti ad integrare. Fortunatamente, se l'integratore è caratterizzato da una struttura simplettica, tale fenomeno è contenuto. Le possibili scelte in questo ambito riguardano il grado di approssimazione, basato su uno sviluppo di Taylor di ciascuna equazione differenziale, e sull'ordine con cui si aggiornano i valori della traiettoria. Dal momento che nell'HMC l'obiettivo è costruire una mappa reversibile, tali integratori dovranno soddisfare questa proprietà.

   Il più utilizzato, perché efficace ed efficiente nella maggior parte dei contesti, è noto come *Leapfrog* Φ_ϵ^L. Esso approssima il flusso hamiltoniano protratto per un tempo ϵ, attraverso uno sviluppo di Taylor del primo ordine, in tre passaggi:

$$
\begin{aligned}
m_{t+\frac{\epsilon}{2}} &= m_t - \frac{\epsilon}{2}\nabla U(\theta_t) \\
\theta_{t+\epsilon} &= \theta_t + \epsilon \nabla K\left(m_{t+\frac{\epsilon}{2}}\right) \\
m_{t+\epsilon} &= m_{t+\frac{\epsilon}{2}} - \frac{\epsilon}{2}\nabla U(\theta_{t+\epsilon})
\end{aligned}
$$

   L'errore di uno di questi integratori è stato dimostrato in Leimkuhler e Reich (2004) essere dell'ordine di $O(\epsilon^3)$, pertanto si avrà:

$$
\Phi_\epsilon^L(z_t) = \phi_\epsilon^H(z_t) + O(\epsilon^3)
$$

   Esso è interpretabile come la successione di tre mappe applicate a z_t in sequenza, la prima e la terza che aggiornano solo il momento in funzione di se

stesso e della posizione, mentre la seconda che aggiorna la posizione in funzione di se stessa e del momento. Il ripetersi della prima e della terza serve per garantire l'esatta reversibilità della traiettoria. Inoltre è facile dimostrare che ciascuna di queste preserva esattamente il volume nello spazio di fase.

$$\Phi_1 : \begin{pmatrix} \theta_t \\ m_t \end{pmatrix} \rightarrow \begin{pmatrix} \theta_t \\ m_{t+\frac{\epsilon}{2}} \end{pmatrix} \implies J_1 = \begin{bmatrix} 1 & 0 \\ -\frac{\epsilon}{2}\nabla^2 U(\theta_t) & 1 \end{bmatrix}$$

$$\Phi_2 : \begin{pmatrix} \theta_t \\ m_{t+\frac{\epsilon}{2}} \end{pmatrix} \rightarrow \begin{pmatrix} \theta_{t+\epsilon} \\ m_{t+\frac{\epsilon}{2}} \end{pmatrix} \implies J_2 = \begin{bmatrix} 1 & \epsilon\nabla^2 K(m_{t+\frac{\epsilon}{2}}) \\ 0 & 1 \end{bmatrix}$$

$$\Phi_3 : \begin{pmatrix} \theta_{t+\epsilon} \\ m_{t+\frac{\epsilon}{2}} \end{pmatrix} \rightarrow \begin{pmatrix} \theta_{t+\epsilon} \\ m_{t+\epsilon} \end{pmatrix} \implies J_3 = \begin{bmatrix} 1 & 0 \\ -\frac{\epsilon}{2}\nabla^2 U(\theta_{t+\epsilon}) & 1 \end{bmatrix}$$

Siccome $\det(J_1) = \det(J_2) = \det(J_3) = 1$ allora anche il determinante dello Jacobiano di $\Phi_\epsilon^L = \Phi_3 \circ \Phi_2 \circ \Phi_1$ sarà pari ad 1. Per le stesse motivazioni lo stesso discorso vale per la traiettoria integrata per un tempo $T = L\epsilon$:

$$\Phi_T^L = \bigcirc_{i=1}^{L} \Phi_\epsilon^L$$

dove l'operatore $\bigcirc_{i=1}^{L}$ applica una composizione iterata di L funzioni. L'errore complessivo di questo integratore, e di una qualsiasi quantità derivante da esso, come ad esempio l'*Hamiltoniana* del sistema, è stato dimostrato in Leimkuhler e Reich (2004) essere dell'ordine di $O(\epsilon^2)$. Questo garantisce l'affidabilità dell'approssimazione della traiettoria anche per lunghi periodi di tempo. Un'altra peculiarità riguarda invece quando questa risulta particolarmente ripida, un comportamento che si può riscontrare in alcune zone dello spazio di probabilità per modelli particolarmente complicati. In questi casi, a differenza degli altri metodi, gli integratori simplettici tendono a divergere fortemente, facendo schizzare la particella agli estremi dello spazio di fase, in corrispondenza di livelli di energia molto alti. Tale peculiarità permette di fare una diagnosi di quali siano le zone dello spazio di probabilità ad essere particolamente difficili da esplorare. In questi casi, le stime Monte Carlo risulteranno distorte dal fatto che non riescono a rappresentare con la giusta

frequenza queste zone patologiche del *Typical Set*. I rimedi ad un problema simile consistono nel cercare di ridurre l'errore di approssimazione, rimpicciolendo ϵ, oppure riparametrizzare il modello, in modo tale da renderlo più facile da esplorare.

Per costruire il nucleo di transizione markoviano a partire da tali integratori simplettici è necessario applicare l'operatore per il cambio di segno dei momenti, che per costruzione preserva il volume, ed appoggiarsi ad un passo di accettazione Metropolis per compensare il fatto che non si si stia più esplorando esattamente la distribuzione microcanonica. Il risultato è il seguente:

$$
\begin{aligned}
& t_{L,\epsilon,\eta} : z_0 \to z_{L\epsilon} \\
& z_{L\epsilon} = z_0 \cdot \mathcal{I}(\eta \geq \alpha) + \left[F \circ \Phi_T^L(z_0)\right] \cdot \mathcal{I}(\eta < \alpha) \\
& \alpha = \min\left\{1, \frac{\pi(F \circ \Phi_T^L(z_0))}{\pi(z_0)}\right\} \\
& \eta \sim Unif(0,1)
\end{aligned}
$$

È facile dimostrare che α corrisponde al minimo tra 1 e l'esponenziale della differenza dei 2 livelli di energia: $H(z_0) - H(F \circ \Phi_T^L(z_0)))$. Questo ad indicare che, coerentemente con l'assunzione fatta sulla distribuzione canonica, i livelli a bassa energia sono più probabili di quelli ad alta energia, quindi, muoversi verso un punto a più alta densità di probabilità è sempre una mossa bene accetta, mentre il viceversa lo è solo con probabilità α. Questo conferma che alla fine l'HMC altro non è che un tipo particolare di Metropolis-Hastings.

3. **Scelta del passo di integrazione:** questo in parte accomuna l'HMC ad un Random Walk Metropolis, dal momento che valori piccoli di ϵ si traducono in un alto tasso di accettazione perché ci si avvicina a risolvere esattamente la traiettoria del flusso e quindi a campionare dalla distribuzione microcanonica, mentre si ottiene un risultato opposto per valori di ϵ grandi. La sostanziale differenza risiede nel fatto che questo non ha nessun impatto sulla località della *proposal*, che invece è regolata da T e quindi da L. Ciononostante, per motivi di costo computazionale, in Betancourt et al. (2014) viene discus-

so come stabilire un limite inferiore e superiore per il tasso di accettazione Metropolis, funzione di ϵ, affinché il costo computazionale complessivo non risulti essere troppo grande. Il risultato finale dei loro conti suggerisce un valore ottimo di α tra 0.6 e 0.8. Conseguentemente, si può definire una funzione obiettivo: $\alpha^*(\epsilon) = \alpha(\epsilon) - \delta$, dove δ rappresenta il valore nominale che si vuole ottenere, verosimilmente contenuto all'interno di questo intervallo, e trovarne lo zero attraverso un algoritmo di ottimizzazione stocastica Nesterov (2009) in maniera automatizzata.

4. **Scelta del tempo di integrazione:** i precedenti gradi di libertà sono dotati di valori di base dimostrati ottimali sotto assunzioni pressoché generali. Ciò ne giustifica un utilizzo a scatola nera e solo in caso di prestazioni scadenti, opportunamente diagnosticabili, è possibile indagare su una più efficiente configurazione di questi. Per quanto riguarda il tempo di integrazione invece le cose si complicano. L'unico approccio generale per l'HMC consiste nel provare attraverso un griglia, non troppo sovra-dispersa per ovvi motivi di costo computazionale, una serie di valori per L e selezionare quello che garantisce il campionamento più efficiente, calcolabile attraverso il numero di campioni indipendenti effettivamente estratti in una data unità di tempo (Geyer, 2011). Tuttavia, tale valore trovato è ottimo solo a livello globale. Ciò vuol dire che in alcune zone dello spazio di probabilità potrebbero essere necessari valori più grandi di L e in altre più piccoli. Questo si capisce se si applica il concetto di *Typical Set* a ciascuna distribuzione microcanonica (2.13): ad ogni iterazione della catena si assegna un momento alla posizione corrente e si procede con un'esplorazione approssimata di un dato livello di energia. Se questa non è una traiettoria critica significa che dopo un po' ritornerà su se stessa, e quindi un'ulteriore esplorazione diventa ridondante ed eccessivamente costosa. In base al tipo di HMC adoperato, ad esempio si può scegliere se usare l'ultimo valore della traiettoria o campionare in maniera uniforme da essa, le cose peggiorano. Infatti, nel primo caso, c'è il rischio di esplorare la traiettoria ma per

via della sua ciclicità il valore proposto finale finisce per trovarsi in prossimità del punto di partenza. Per queste motivazioni, il tempo di integrazione L è uno dei fattori più importanti di cui tenere conto, a tal punto che ha motivato l'implementazione di una variante dell'HMC, leggermente più complicata, su cui è fondato STAN e da cui ne deriva il suo successo.

2.4 Il No U-Turn Sampler di Gelman & Hoffman e lo stato attuale di STAN

STAN (STAN development team, 2021) è un programma reso disponibile gratuitamente a partire dal 2012 che permette di esplorare la distribuzione a posteriori di modelli statistici bayesiani anche molto complicati. Similmente ad alcuni suoi concorrenti, quali ad esempio BUGS Gilks et al. (1994) e JAGS Plummer et al. (2003), è strutturato in maniera tale da lasciare che sia l'utente a descrivere le caratteristiche del modello attraverso una sintassi ben precisa. Una volta superata questa barriera linguistica iniziale, è impressionante la varietà di modelli che è possibile adattare. Ciò permette una maggiore flessibilità e trasparenza riguardo le assunzioni fatte e una minore dipendenza da librerie o programmi già impacchettati, atti a svolgere azioni specifiche, per i quali incombe spesso il rischio di un utilizzo improprio, a scatola nera. A differenza dei suoi concorrenti tuttavia, STAN non basa il suo algoritmo di punta sul Metropolis-within-Gibbs, ma su di un tipo particolare di HMC. Questo gli garantisce una minore vicinanza dei valori proposti ad ogni iterazione della catena rispetto a quelli correnti e quindi una più rapida esplorazione del *Typical Set*. Il prezzo da pagare in termini di costo computazionale riguarda la necessita di calcolare, ad ogni iterazione, oltre al valore della densità a posteriori quello del suo gradiente. Il vantaggio che ne deriva, in termini di numero di campioni indipendenti effettivi prodotti dalla catena per unità di tempo, la rende però una scelta spesso preferibile.

Il particolare algoritmo di HMC su cui STAN è fondato è stato descritto per la prima volta in Hoffman et al. (2014), un articolo pubblicato successivamente all'u-

scita ufficiale del programma nel 2012. Una spiegazione più dettagliata e coerente con la versione attuale invece è presentata in Betancourt (2017). Il nome per la variante all'HMC proposto dagli autori è *No U-Turn Sampler*, abbreviato con la sigla *NUTS*, esplicativo di ciò che fa. Come visto nel paragrafo precedente, infatti, l'unico grado di libertà dell'HMC per cui non è possibile avere un valore di base opportunamente giustificato è L, il numero di passi di integrazione che, insieme ad ϵ, la lunghezza di tali passi, determina il tempo di integrazione di ciascuna traiettoria T. *NUTS* tenta di porre rimedio proprio a questa carenza.

2.4.1 Criteri di terminazione

Come riportato in Betancourt (2016*b*, paragrafo 1.2), le traiettorie che seguono il flusso hamiltoniano sono caratterizzate da una certa ciclicità che ne rende l'esplorazione ridondante se protratta per un tempo T eccessivamente lungo. Un possibile *criterio di terminazione* per il tempo di integrazione riguarda l'identificazione dell'istante t^* in cui la direzione della traiettoria di θ_{t^*}, riassunta dal suo rispettivo momento m_{t^*}, collassa su se stessa, implicando un ritorno sui suoi passi. Sfruttando la seguente identità, derivante direttamente dalla legge (2.10):

$$\theta_{t^*} - \theta_0 = \theta_0 + \int_0^{t^*} \frac{d\theta_t}{dt} dt - \theta_0 = \int_0^{t^*} \nabla K(m_t) dt$$

è possibile, per certe particolari scelte di K, definire questa quantità come un vettore, riscalato per un'opportuna metrica, appartenente allo spazio vettoriale dei momenti, che indichiamo sempre con $\mathcal{M}$. Essendo l'integrazione una somma infinitesimale, e gli elementi di uno spazio vettoriale chiusi alla somma, tale quantità si può visualizzare come un'applicazione iterata della regola del parallelogramma, che definisce la direzione complessiva della traiettoria come la somma della direzioni che la percorrono. Questo permette di definire un prodotto interno tra questa quantità e la direzione infinitesimale prossima alla traiettoria, espressa da m_t^*. Tale prodotto misura la concordanza tra questi due vettori, in particolare, se negativo, identifica un'inversione ad U *(U-Turn)* della traiettoria e con ciò la possibilità di

fermare l'esplorazione della distribuzione microcanonica all'istante t^*:

$$t^* = \left\{ t \in \mathbb{R} : \left\langle \int_0^t \nabla K(m_s) ds \, , \, m_t \right\rangle < 0 \right\}$$

Dal momento che la traiettoria nel concreto è approssimata, lo sarà anche tale integrale attraverso una sommatoria dei gradienti dell'energia cinetica calcolati in corrispondenza dei momenti stimati dall'integratore simplettico:

$$L^* = \left\{ L \in \mathbb{N} : \left\langle \sum_{i=1}^{L} \nabla K(m_{i \cdot \epsilon}) \, , \, m_{L \cdot \epsilon} \right\rangle < 0 \right\}$$

Il NUTS utilizza $\xi \sim N_d(0, M)$, ciò permette di definire la distanza tra due momenti in $\mathcal{M}$ attraverso una semplice distanza euclidea dotata di metrica M^{-1}:

$$\int_0^{t^*} M^{-1} m_t dt = M^{-1} m_0^{t^*} \implies \left\langle m_0^{t^*}, m_{t^*} \right\rangle_{M^{-1}} = \left(m_0^{t^*} \right)^T M^{-1} m_{t^*}$$

dove $m_0^{t^*}$ rappresenta la somma cumulata del momento lungo la traiettoria.

Una procedura generale per definire un criterio di terminazione è descritta in Betancourt (2016*b*) con il nome di *esaurimento*. In particolare, se si definisce uno scalare, u, funzione del flusso hamiltoniano, tale da essere in esso limitato uniformemente:

$$|u \circ \phi_t(z) - u(z)| < \infty \, , \forall t \in \mathbb{R}$$

allora il valore atteso temporale della suo tasso di cambio nel tempo tenderà, in maniera oscillatoria, per forza a zero:

$$\lim_{T \to \infty} \frac{1}{T} \int_0^T \frac{du(\phi_s^H(z))}{ds} ds \to 0$$

Sempre in Betancourt (2016*b*), viene proposto di utilizzare il *viriale*:

$$G = \theta_s^T m_s$$

come quantità scalare di cui sorvegliare la media temporale. Tale scelta viene giustificata dall'autore affermando come sia l'unica non banale e canonica ai sistemi hamiltoniani per cui sia assicurata la condizione di limitatezza se calcolata su traiettorie non critiche.

Siccome le traiettorie t sono approssimate, anche in questo caso tale integrale verrà approssimato da una sommatoria a cui bisognerà applicare una correzione per compensare la distorsione dovuta all'errore di integrazione:

$$\begin{aligned}\rho(t) &= \frac{1}{|t|}\sum_{z\in t}\mathbb{P}(z|t)\frac{dG}{ds}(z) \\ &= \frac{1}{|t|}\sum_{z\in t}\mathbb{P}(z|t)\left[\frac{dG}{d\theta}\frac{d\theta}{ds}(z)+\frac{dG}{dm}\frac{dm}{ds}(z)\right] \\ &= \frac{1}{|t|}\sum_{(\theta,m)\in t}\mathbb{P}(\theta,m|t)\left[m^T\nabla K(m)-\theta^T\nabla U(\theta)\right]\end{aligned}$$

dove per $\mathbb{P}(z|t)$ si utilizza l'operatore *softmax* con lo scopo di applicare tale correzione:

$$\mathbb{P}(z|t) = \frac{e^{-H(z)}}{\sum_{z'\in t} e^{-H(z')}}$$

In realtà, visto che questa è comunque un'approssimazione, il valore di $\frac{dG}{ds}$ può essere semplicemente approssimato come:

$$\frac{dG}{ds} \approx \frac{\theta_{s+\epsilon}^T m_{s+\epsilon} - \theta_s^T m_s}{\epsilon}$$

dove ϵ corrisponde al passo di integrazione. Infine, per definire un criterio di terminazione, è sufficiente stabilire una soglia, τ, per la media temporale del tasso di cambio del viriale e fermare l'integrazione ogni qualvolta

$$|\rho(t)| < \tau \tag{2.19}$$

L'algoritmo di HMC che risulta dall'applicazione di questo criterio di terminazione è stato nominato dall'autore come *Exhaustive Hamiltonian Monte Carlo*, abbreviato con XHMC. Lo svantaggio di un tale metodo rispetto al precedente deriva dal fatto che il valore di τ va ricercato all'interno di una griglia in maniera tale da minimizzare il costo computazionale, inteso come numero di campioni effettivamente indipendenti generati dalla catena per unità di tempo. Il vantaggio che si ottiene rispetto all'approccio che si limita a calibrare L, riguarda il fatto che adesso il tempo di integrazione è libero di adattarsi al meglio alla geometria dello spazio

di probabilità. Il suo punto di forza rispetto all'altro criterio di terminazione invece risiede nella sua applicabilità a qualsiasi tipologia di HMC, dal momento che non richiede la specificazione di un prodotto interno, e quindi di una metrica, per misurare l'anti-correlazione tra vettori nello spazio vettoriale dei momenti.

Segue qui un esempio esplicativo per comprendere il funzionamento di tali criteri. Si consideri la seguente legge di probabilità bivariata:

$$X \sim N_2(0, \Sigma) \quad , \text{ con } \Sigma = \begin{pmatrix} 1.0 & 0.2 \\ 0.2 & 1.0 \end{pmatrix}$$

per cui la rispettiva energia potenziale è data da:

$$U(X) \propto \frac{X^T \Sigma^{-1} X}{2} \tag{2.20}$$

Per quanto riguarda l'energia cinetica, si è scelto l'utlizzo di una funzione di disintegrazione gaussiana bivariata con matrice di varianze e covarianze pari alla matrice identità.

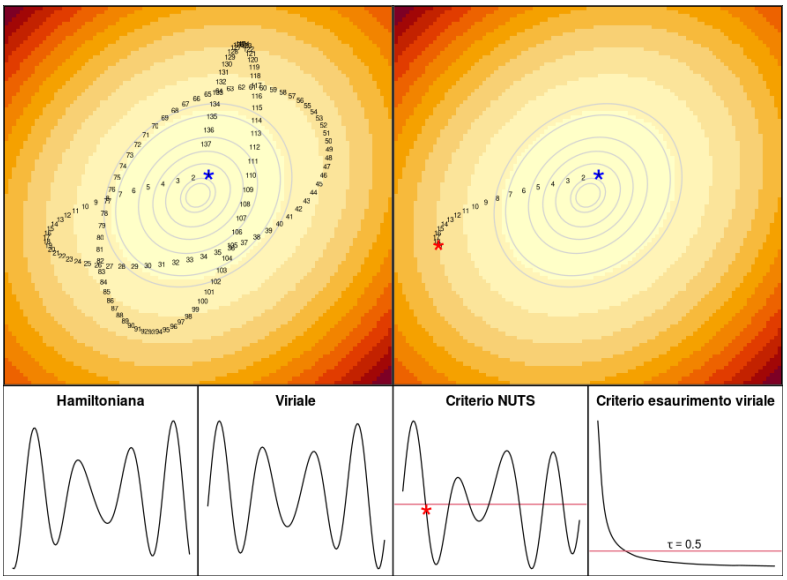

Figura 2.3: Traiettoria del *leapfrog* per $U(x)$ definito in (2.20) con $K(m) \propto \frac{\|m\|^2}{2}$

In Figura 2.3 è riportata la traiettoria approssimata risultante e alcune sue statistiche, in particolare, in alto a sinistra questa è tracciata su di un grafico mostrante le curve di livello di $U(x)$ ed è fatta continuare fino a quando sembra

ritornare al punto di partenza, indicato dall'asterisco blu. Nei due grafici sottostanti di sinistra sono mostrate le rispettive serie storiche di due sue statistiche: l'energia del sistema (Hamiltoniana) e il viriale. Come prevedibile, la natura oscillatoria della prima è da imputare all'incapacità di approssimare esattamente la traiettoria. Il secondo sotto-grafico mostra invece come, per traiettorie non critiche, il viriale sia una quantità limitata uniformemente. Nel grafico in alto a destra viene mostrata la medesima traiettoria sulle curve di livello di $U(x)$, ma questa volta terminata nell'istante in cui il criterio NUTS segnala uno *U-Turn*. Nei due grafici sottostanti sono riportati rispettivamente l'andamento del criterio NUTS (basato sul prodotto interno) e quello basato sull'esaurimento del viriale per un dato valore di τ. È importante notare la natura decrescente di quest'ultimo, che è assicurata dal fatto che il viriale sia una quantità limitata lungo il flusso.

2.4.2 Reversibilità della *proposal*

L'inserimento di un criterio di terminazione fa venire meno la condizione (1.5), che garantisce l'invarianza del nucleo di transizione alla distribuzione obiettivo, formalizzata in (1.6). Questo è una conseguenza del fatto che il flusso risultante non è più reversibile, infatti non è detto che il momento m_0, che si incontra integrando all'indietro, segnali uno *U-Turn* rispetto alla traiettoria $\theta_0 - \theta_{t^*}$. Come conseguenza di ciò (1.4) torna ad essere 0 come accadeva in (2.1), nonostante l'utilizzo dell'operatore di cambio di segno, rendendo il nucleo di transizione un nucleo di tipo Metropolis Hastings non valido. A questo problema si può porre rimedio in diversi modi, la maggior parte di questi considera il campionamento non più solo dalla traiettoria esplorata ma anche tra le diverse possibili traiettorie, scomponendo il nucleo di transizione in due passi:

1. Campionare una traiettoria t da un insieme di diverse possibili, $\mathcal{T}_z$, tutte contenenti il punto di partenza z, definito sullo spazio di fase $\Theta \times \mathcal{M}$. Per le proprietà dei flussi e dei rispettivi integratori simplettici, queste traiettorie

sono tutte una frazione della macro-traiettoria, esatta o approssimata, che passa per z.

2. Campionare uno stato qualsiasi da questa: $z' \sim K(z'|t)$.

Il nucleo di transizione è dunque definito da

$$K(z'|z) = \sum_{t \in \mathcal{T}_z \cap \mathcal{T}_{z'}} K(z'|t)\mathbb{P}(t|z)$$

Ovviamente non tutte le scelte di $K(z'|t)$ e $\mathbb{P}(t|z)$ assicurano l'invarianza rispetto alla distribuzione obiettivo. Il primo passo necessario consiste nell'assicurare la condizione:

$$\mathbb{P}(t|z) = \mathbb{P}(t|z') \quad \forall z' : \mathcal{T}_z \cap \mathcal{T}_{z'} \neq \emptyset \tag{2.21}$$

Un'opzione consiste nel costruire la traiettoria in maniera progressiva, raddoppiandone la lunghezza ad ogni passo, decidendo casualmente se integrare in avanti o all'indietro nel tempo, rispetto agli estremi z_- e z_+ correnti, e fermando la procedura non appena si incontri un criterio di terminazione. Se sono risultati necessari N raddoppi casuali per incappare in tale evento, questo vuol dire che la traiettoria ottenuta, t, è solo una delle 2^N possibili, ognuna con probabililtà 2^{-N}. Pertanto, $\mathbb{P}(t|z) = 2^{-N}$. Partendo da un valore $z' \in t$ arbitrario, la probabilità di ricostruire esattamente lo stesso t sarà sempre uguale a 2^{-N}. Per assicurare questa proprietà, è necessario che i controlli per il criterio di terminazione siano effettuati con gli estremi di ogni possibile sottotraiettoria di ampiezza 2^n per $n = 1, \ldots, N$, altrimenti potrebbe capitare di costruire delle traiettorie che hanno probabilità 0 di essere ricostruite se si partisse da un qualche punto in esse contenuto. Il funzionamento si capisce meglio se si considera ogni possibile t così generato come un albero binario bilanciato, dove ogni nodo identifica un sotto albero da cui si diramano due biforcazioni con un altro nodo ciascuna. La struttura ricorsiva si ripete fino ad arrivare al livello più basso, il 2^{N-1}-esimo, in cui sono ubicate le foglie dell'albero, corrispondenti ai 2^N valori della traiettoria. Questi sono indicizzabili dal tempo di integrazione, discreto che per convenzione si fa cominciare in 0, che può assumere anche valori negativi.

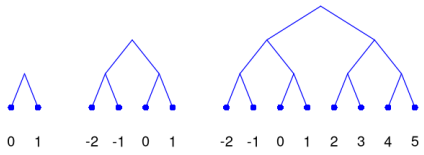

Figura 2.4: Costruzione progressiva della traiettoria tramite raddoppio.

In Figura 2.4 è mostrato un esempio in cui, a sinistra, si parte dall'istante in 0 e si integra avanti nel tempo ottenendo il valore in 1. Successivamente si raddoppia la traiettoria all'indietro nel tempo calcolando prima la sotto-traiettoria in avanti, ottenendo il valore in -1 e poi la sottotraiettoria all'indietro, ottenendo il valore in -2. Infine, a destra, si applica un raddoppio in avanti, che è scomponibile a sua volta nella costruzione ricorsiva del sotto albero con 2, 3 e del sotto albero con 4, 5.

Dal momento che l'albero è binario e bilanciato, si osserva che i possibili valori con cui z_i è confrontabile sono solo quelli definiti dalle seguenti successioni:

$$\{z_{i-2^k}\}_{k=0}^{N}\,,\ \{z_{i-1}\}\,,\ \{z_{i+1}\}\,,\ \{z_{i+2^k}\}_{k=0}^{N}$$

Questo comporta una sub-ottimalità nell'individuazione di *U-Turn*, che però fa risparmiare molto sia in termini di costo computazionale che di complessità nella costruzione di un tale nucleo di transizione. Inoltre, l'intrecciarsi di queste successioni con il variare di i, e il fatto che la traiettoria sia una funzione continua, rendono del tutto irrilevante questo malus lasciando invece intaccati i bonus appena citati. Nell'esempio in Figura 2.4 i confronti tra gli estremi di tutte le possibili sotto-traiettorie riguardano le seguenti coppie:

$$(0,1)\quad(-1,-2)\quad(-2,1)\quad(2,3)\quad(4,5)\quad(2,5)\quad(-2,5)$$

Lo schema funziona perché c'è un solo modo per ricostruire esattamente t a partire da un qualsiasi $z' \in t$ e, indipendentemente da dove si parte, i confronti tra i valori estremi delle sotto-traiettorie sono sempre gli stessi per t. Questo assicura che tale modo non incorra mai in un criterio di terminazione prematuro. L'utilizzo di questa struttura ad albero, poi, garantisce anche un'implementazione ricorsiva efficiente, in cui il principale vantaggio è dato dal fatto che, se in un dato sotto ramo di un

sotto albero si incontra un criterio di *stop*, allora l'intero sotto albero in questione verrà scartato e si risparmierà così il prezioso tempo computazionale necessario per il suo completamento. Nell'esempio in Figura 2.4, se si incorre in un criterio di terminazione durante il confronto $(2, 3)$, allora non sarà necessario calcolare i valori in 4 e 5 e la traiettoria proposta sarebbe quella che va da -2 a 1. In questo caso sono state risparmiate solo 2 integrazioni ma se, ad esempio, si incontrasse uno stop ad una profondità di 7, si risparmierebbero ben $2^7 = 128$ integrazioni. Altri vantaggi, non meno rilevanti, riguardano l'allocazione di memoria, dal momento che muovendosi dai livelli più bassi ai livelli più alti dell'albero sono poche le informazioni che ci si deve portare dietro: ovvero gli estremi e il valore proposto per questa sotto-traiettoria. Siccome il raddoppio della traiettoria fa crescere il costo computazionale esponenzialmente con il proseguimento di questa, è bene porre un limite superiore alla profondità dell'albero. Un valore ragionevole è 10, che corrisponde ad un massimo di $2^{10} - 1 = 1023$ integrazioni. Se una traiettoria finisce perché raggiunge questo numero massimo non è necessario scartare nessun campione ed è possibile scegliere un qualsiasi valore da essa. Un criterio di terminazione aggiuntivo che viene considerato si ha quando ci si imbatte in una *transizione divergente*, ovvero un'integrazione della traiettoria che comporta una spropositata variazione del livello di energia, sintomo che l'approssimazione non è più buona, anche in questo caso bisogna scartare l'intera sottotraiettoria che la contiene per garantire la reversibilità. Come già citato nel paragrafo 2.3.2, tali fenomeni possono occorrere in presenza di zone molto strette dell'energia potenziale, sono evitabili attraverso la specificazione di un passo di integrazione ϵ più piccolo e, se trascurati, producono distorsione nelle stime Monte Carlo perché sotto-rappresentative di una zona dello spazio di probabilità dove potenzialmente risiede una massa rilevante.

Questa procedura di raddoppio deve essere accompagnata da un nucleo particolare $K(z'|t)$, che tenga conto, ad ogni passo, della composizione della traiettoria corrente come $t = t_{old} \cup t_{new}$ e che proponga ogni volta un solo valore z' da questa. Dove t_{old} è la traiettoria ottenuta fino al passo precedente mentre t_{new} rappresenta l'aggiunta avvenuta al passo corrente. Uno schema comodo consiste nel considerare

una mistura:

$$\mathbb{K}(z'|t) = p \cdot K(z'|t_{old}) + (1-p) \cdot K(z'|t_{new})$$

Dove p rappresenta la probabilità di tenere il valore proposto per la traiettoria vecchia e $1-p$ invece di accettare quello proveniente da t_{new}. Se si vuole ottenere un campionamento multinomiale dalla traiettoria completa al termine della procedura di raddoppio, è sufficiente porre:

$$\begin{aligned} p &= \frac{w_{old}}{w_{old} + w_{new}} \\ K(z'|t_{old}) &= \frac{e^{-H(z')}}{w_{old}} \cdot \mathcal{I}(z' \in t_{old}) \\ K(z'|t_{new}) &= \frac{e^{-H(z')}}{w_{new}} \cdot \mathcal{I}(z' \in t_{new}) \end{aligned}$$

dove $w_{old} = \sum_{z^* \in t_{old}} e^{-H(z^*)}$ è la somma delle densità canoniche non normalizzate per la vecchia traiettoria che ha lo scopo di rendere $K(z'|t_{old})$ un operatore *soft-max*, mentre $w_{new} = \sum_{z^* \in t_{new}} e^{-H(z^*)}$ è l'analogo ma per la traiettoria aggiuntiva. È facile dimostrare che questa operazione di raddoppio preserva la distribuzione canonica rispetto alla traiettoria considerata, i passaggi sono riportati in appendice A.4. Questo campionamento multinomiale serve a correggere l'errore di approssimazione del flusso hamiltoniano, nel caso esatto infatti corrisponde al campionamento uniforme da esso e quindi direttamente dalla distribuzione microcanonica.

In realtà, una qualsiasi scelta di p per cui vale l'invarianza è valida. In particolare è possibile favorire lo sviluppo di anti-correlazione lungo la traiettoria considerata aumentando la probabilità di campionare da t_{new}, allontanando così la *proposal* finale dai valori iniziali. Lo schema proposto è il seguente:

$$p = 1 - \min\left\{1, \frac{w_{new}}{w_{old}}\right\}$$

Ancora una volta, la dimostrazione dell'invarianza della distribuzione canonica sulla traiettoria considerata è disponibile in appendice A.4. È facile notare che, qualora il flusso hamiltoniano fosse esattamente ricostruito dall'integratore simplettico, la *proposal* risultante di questo secondo metodo corrisponderebbe all'estremo, destro o sinistro, della traiettoria considerata, come tipico dell'HMC classico. Quest'ultimo,

non è da considerarsi come il metodo a cui tendere, dal momento che, come argomentato in Neal (2011), la selezione deterministica di z' da t comporta una maggiore sensibilità all'errore ciclico che caratterizza gli integratori simplettici, che si traduce in una probabilità di accettazione (1.4) minore. Campionare dalla traiettoria, invece, permette di smussare tale quantità lungo tutta la traiettoria ottenendo delle prestazioni migliori in media. Questo è il motivo alla base del perché l'algoritmo NUTS ha un tasso di accettazione nominale consigliato maggiore dell'HMC classico.

Entrambi i nuclei di transizione per un'operazione di raddoppio visti, dunque, preservano la distribuzione canonica. Conseguentemente anche una qualsiasi loro sequenza deterministica lo farà, e quindi anche il nucleo di transizione per la traiettoria completa:

$$t : t_1 \to t_2 \to \ldots \to t_N$$

Indicando tale nucleo, per semplicità di notazione, con $K(z'|t)$, si avrà che:

$$\sum_z K(z'|t)\pi(z)\mathcal{I}(z \in t) = \pi(z')\mathcal{I}(z' \in t)$$

Tuttavia, t è una sequenza stocastica, dove però si sa che, qualsiasi sia il valore di N, la probabilità di ottenere t tra tutte le traiettorie possibili è 2^{-N}, ma non è questo l'aspetto importante, quanto il fatto che l'identità (2.21) è rispettata e riscrivibile come:

$$\mathbb{P}(t|z)\mathcal{I}(z \in t) = \mathbb{P}(t|z')\mathcal{I}(z \in t) \quad , \forall z' \in t$$

Questa garantisce il seguente risultato:

$$\begin{aligned}
\sum_z K(z'|z)\pi(z) &= \sum_z \sum_t K(z'|t)\mathbb{P}(t|z)\pi(z)\mathcal{I}(z \in t) \\
&= \sum_t \sum_z K(z'|t)\mathbb{P}(t|z')\pi(z)\mathcal{I}(z \in t) \\
&= \sum_t \mathbb{P}(t|z') \sum_z K(z'|t)\pi(z)\mathcal{I}(z \in t) \\
&= \sum_t \mathbb{P}(t|z')\pi(z')\mathcal{I}(z' \in t) \\
&= \pi(z')
\end{aligned}$$

e dimostra l'invarianza del nucleo di transizione markoviano dell'algoritmo NUTS.

2.4.3 Specifiche di STAN

STAN mette a disposizione diversi algoritmi per fare inferenza bayesiana, ma quello di punta è il NUTS appena descritto, con $\xi \sim N_d(0, M)$ e criterio di terminazione basato sul prodotto interno. Alcuni accorgimenti riguardano la fase di *warm-up* che è divisa in tre parti:

1. Una prima parte, breve, in cui si utilizza una procedura adattiva per stimare il valore ottimo della lunghezza del passo di integrazione. Tali procedure, se applicate ai metodi MCMC, richiedono un criterio vanescente (Andrieu e Thoms, 2008), cioè che annulli il suo effetto asintoticamente, per evitare che l'adattamento distorca le stime Monte Carlo. Questo in realtà non sarebbe del tutto necessario dal momento che i campioni prodotti in questa fase sono alla fine scartati e la procedura adattiva non viene applicata durante la fase di campionamento vero e proprio. Quest'ultimo approccio, inoltre, è reso necessario dal fatto che la vanescenza spesso è più lenta di quanto non ci metta ϵ a convergere, comportando altrimenti un numero maggiore di campioni da scartare per niente. Il metodo usato da STAN è basato su un algoritmo di ottimizzazione stocastica, in particolare il *Dual averaging* di Nesterov (2009), caratterizzato dall'alternarsi di due statistiche temporali della catena:

$$\log \epsilon_{t+1} = \mu - \frac{\sqrt{t}}{\gamma} \frac{1}{t + t_0} \sum_{i=1}^{t} H_i$$

$$\overline{\log \epsilon}_{t+1} = \eta_t \log \epsilon_{t+1} + (1 - \eta_t) \overline{\log \epsilon}_t$$

   Ciascuno di questi parametri contenuto in queste formule ha un significato ben preciso e la loro calibrazione, attraverso studi di simulazione condotti in Hoffman et al. (2014), è già arrivata a definirne dei valori validi pressoché universalmente per il NUTS. L'idea di fondo consiste nel trovare lo zero della media di una componente stocastica che risulta funzione monotona rispetto ad ϵ. Questa componente nel caso del NUTS è rappresentata da:

$$H_i = \frac{1}{|t_i|} \sum_{z \in t_i} \min \left\{ 1, e^{H(z_0) - H(z)} \right\} - \delta \tag{2.22}$$

ovvero è la media degli scarti dei tassi di accettazione (1.4) degli elementi sulla traiettoria rispetto al valore δ nominale. Questa è funzione decrescente rispetto ad ϵ perché per suoi valori piccoli l'approssimazione del flusso hamiltoniano migliora rendendo il livello di energia iniziale, di z_0, sempre più vicino a quello di un qualsiasi punto, z, sulla traiettoria approssimata. Siccome ϵ deve essere un valore positivo allora si considera la sua trasformata logaritmica e ad ogni passo lo si comprime sempre di più verso il valore di μ ad un tasso regolato da γ. Successivamente si aggiorna la media cumulata di queste quantità, in scala logaritmica, con un peso che decresce nel tempo:

$$\eta_t = t^{-\kappa} \quad \text{, con } \kappa \in (0.5, 1] \quad \text{, tale per cui } \sum_{t=1}^{\infty} \eta_t = \infty \text{ e } \sum_{t=1}^{\infty} \eta_t^2 = 0$$

tale peso è ciò che rende la procedura vanescente, dal momento che, con il crescere di t, nuovi valori di $\log \epsilon_t$ avranno sempre meno impatto nel determinare $\overline{\log \epsilon_t}$. Dopo un numero di iterazioni, N_{init}, sufficiente si pone $\epsilon = \bar{\epsilon}_{N_{init}}$.

I valori utilizzati di default da STAN sono i seguenti:

- $\mu = \log 10 + \log \epsilon_0$ con ϵ_0 un valore iniziale trovabile attraverso un semplice algoritmo che dimezza/raddoppia ϵ monotonicamente fino a che l'errore di approssimazione non suggerisce invece di aumentarne/ridurne il valore;
- $\gamma = 0.05$;
- $t_0 = 10$, ha lo scopo di desensibilizzare la procedura rispetto alle prime iterazioni, spesso troppo rumorose;
- $\kappa = 0.75$;
- $\delta = 0.8$, la giustificazione per tale valore ottimo è euristica, in Hoffman et al. (2014) si suggeriva il valore 0.6 giustificato da studi di simulazione, tuttavia quello correntemente utilizzato è 0.8, da ritenersi più affidabile data l'evoluzione che l'algoritmo NUTS ha esperito in questi anni;
- $N_{init} = 50$;

2. Dopo la prima taratura del passo di integrazione ottimo, è prevista una fase di *warm up* nella quale si esplora il *Typical Set* per un numero prestabilito di iterazioni, N_{adapt}, con l'obiettivo di produrre una stima per la matrice di massa:

$$\widehat{M} = \left[\frac{N_{adapt} \cdot \widehat{\mathbb{C}(\theta|y)} + 5 \times 10^{-3} \cdot I_d}{N_{adapt} + 5} \right]^{-1}$$

   Questo passaggio non è obbligatorio ed è importante che venga fatto senza stimare adattivamente anche ϵ. Nei contesti di alta dimensionalità, inoltre, può risultare preferibile stimare M diagonale invece che densa, sia per ottenere delle stime migliori per le componenti di interesse, sia per alleggerire il costo computazionale di ogni singola iterazione.

3. Se si è stimata la matrice di massa, allora il valore ottimo per ϵ sarà generalmente più grande di quello trovato precedentemente. Di conseguenza è possibile ripetere il primo passo ma con il valore di M stimato. Il default di STAN utilizza una numerosità leggermente più alta questa volta: $N'_{init} = 75$ e inizializza ϵ_0 al valore trovato durante la prima fase.

Sia dopo la prima che l'eventuale terza fase, come suggerito in Neal (2011), è buona norma non tenere un valore per ϵ fisso, ma campionarlo uniformemente in un piccolo intorno, ad esempio $[0.9\hat{\epsilon}, 1.1\hat{\epsilon}]$. Questo non altera la proprietà di invarianza del nucleo e permette di rendere meno patologico il problema relativo alla presenza di zone della distribuzione a posteriori caratterizzate da alta curvatura, dove sarebbe necessaria una precisione maggiore. Anche se, per tali problemi, le uniche soluzioni veramente efficaci riguardano la riduzione forzata di ϵ, ottenibile specificando un tasso di accettazione nominale δ più alto, oppure un'opportuna riparametrizzazione del modello.

Capitolo 3

Hamiltonian Monte Carlo discontinuo

Una grave mancanza di STAN, rispetto ai linguaggi probabilistici alternativi, basati su algoritmi di tipo Metropolis-within-Gibbs, riguarda l'impossibilità di trattare parametri discreti, dal momento che le dinamiche hamiltoniane sono ben definite solo su spazi continui. Da qui in avanti con il termine *dicreto* ci si riferirà sia a variabili aleatorie quantitative discrete che qualitative, ordinali o sconnesse. Come suggerito in Neal (2011), un approccio *NUTS-within-Gibbs* è pur sempre considerabile. Esso sfrutterebbe il flusso hamiltoniano solo per la distribuzione *full conditional* dei parametri continui rispetto a quelli discreti e successivamente aggiornerebbe il valore di questi ultimi con un passo Gibbs o Metropolis. Ciononostante, STAN non permette un tale approccio e incoraggia l'utente a scrivere il proprio modello, attraverso un'opportuna marginalizzazione dei parametri discreti, solo in funzione di quelli continui.

3.1 Marginalizzazione dei parametri discreti

La procedura suggerita nel manuale di (STAN development team, 2021) consiste in una iniziale partizione dello spazio parametrico in:

$$\Theta = \Theta_I \times \Theta_J \quad \text{, con } \Theta_I \subseteq \mathbb{R}^{d-k} \text{ e } \Theta_J \subseteq \mathbb{N}^k$$

dove le corrispondenti coordinate θ_I e θ_J sono dei vettori aleatori rispettivamente continui e discreti. Successivamente, i parametri discreti vengono trattati come *parametri di disturbo* da marginalizzare[1], definendo così la distribuzione a posteriori esclusivamente per θ_I:

$$\pi(\theta_I|y) = \sum_{x \in \Theta_J} \pi(\theta_I|x, y)\mathbb{P}(\theta_J = x)$$

Questa è una funzione continua nei suoi argomenti[2], pertanto trattabile dall'HMC e dal NUTS. I vantaggi della marginalizzazione si manifestano in due istanze: una durante la fase di esplorazione dello spazio di probabilità da parte della catena e una successivamente per ottenere uno stimatore Monte Carlo con varianza minore.

- Nella fase di campionamento, l'utilizzo della distribuzione a posteriori marginale, come distribuzione invariante per la catena, è una procedura che prende il nome di *Collapsed Gibbs Sampler* (Liu, 1994) e, sotto alcune condizioni non troppo stringenti, assicura un migliore *mixing* da parte delle catene e una convergenza più veloce delle stime Monte Carlo. I campioni simulati per θ_J, potenzialmente necessari per il calcolo di queste stime, possono essere ottenuti in una fase successiva a quella di MCMC, a partire dalla legge condizionata $\theta_J|\theta_I, y$ attraverso un operatore *softmax*. Per farlo è sufficiente calcolare, per ogni valore θ_I della catena, la probabilità a posteriori in corrispondenza di ogni modalità presente in Θ_J:

  $$\mathbb{P}(\theta_J = x|\theta_I, y) = \frac{\pi(x|\theta_I, y)\mathbb{P}(\theta_J = x)}{\sum_{x \in \Theta_J} \pi(x|\theta_I, y)\mathbb{P}(\theta_J = x)} \quad , \forall x \in \Theta_J$$

  e generare un valore da questa legge attraverso un campionamento multinomiale. Infine, per le proprietà dei metodi di simulazione, l'unione dei campioni simulati da $\theta_J|\theta_I, y$ e da $\theta_I|y$ produce campioni provenienti dalla legge congiunta $\theta|y$.

[1] Questa è una conseguenza di essere in un contesto bayesiano che tratta tutti i parametri come variabili aleatorie.

[2] Il caso in cui è la densità a posteriori ad essere discontinua nonostante i parametri siano continui è un altro discorso, non trattato in questa sede.

- È possibile ottenere uno stimatore Monte Carlo alternativo, caratterizzato da una varianza minore:

$$\tilde{I} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}\left[f(\theta_I^i, \theta_J) | \theta_I^i \right] \tag{3.1}$$

  Questo risultato è una diretta conseguenza della legge dei valori attesi iterati

$$\mathbb{E}\left[\hat{I}\right] = \mathbb{E}_{\theta_I}\left[\mathbb{E}[\hat{I}|\theta_I]\right],$$

  e della legge della varianza totale

$$\mathbb{V}(\hat{I}) = \mathbb{V}_{\theta_I}(\mathbb{E}[\hat{I}|\theta_I]) + \mathbb{E}_{\theta_I}[\mathbb{V}(\hat{I}|\theta_I)] \geq \mathbb{V}_{\theta_I}(\mathbb{E}[\hat{I}|\theta_I]).$$

  Definendo $\hat{I}$ come in (1.2), è facile notare che $\tilde{I} = \mathbb{E}[\hat{I}|\theta_I]$ e quindi, per le due leggi appena enunciate, l'utilizzo di (3.1) non introduce nessuna ulteriore distorsione nello stimatore Monte Carlo originale, che sotto le condizioni del teorema ergodico è noto essere non distorto, mentre la varianza viene sicuramente ridotta, migliorandone l'efficienza, o lasciata invariata.[3]

Questo approccio, nonostante i vantaggi appena citati, ha il grosso difetto di dover fare una sommatoria rispetto a tutti gli stati in Θ_J. La fasi post-MCMC non sono di grande preoccupazione perché fattibili in maniera vettorizzata con i campioni generati di θ_I, mentre è l'MCMC effettivo ad esserne maggiormente affetto e in particolar modo l'HMC, che necessita di calcolare, oltre tale sommatoria, anche il suo gradiente per ciascun passaggio dell'integratore simplettico in ogni iterazione. Ultima problematica, ma non per questo meno importante, riguarda la stabilità numerica di tale operazione di somma. Per questo genere di conti, questa è solitamente garantita attraverso una trasformata logaritmica, che in statistica, come sottoprodotto, trasforma la produttoria della distribuzione a posteriori in una più

[3] Come riportato in Robert e Roberts (2021), questa procedura viene spesso erroneamente chiamata *Rao-Blackwellizzazione*, dal momento che come l'omonimo teorema (Rao, 1945; Blackwell, 1947) è basato sulla legge dei valori attesi iterati e della varianza totale. Tuttavia, il teorema di Rao-Blackwell riguarda l'efficienza di stimatori ottenuti facendo il valore atteso condizionato ad una statistica sufficiente minimale, cosa che in questo contesto non è presente visto che θ_I, non essendo è una funzione dei dati, non può essere considerata tale.

comoda sommatoria. Tuttavia, la densità a posteriori marginale è una sommatoria di produttorie e ciò fa perdere tale proprietà, a cui bisogna porre rimedio attraverso l'utilizzo dell'operatore *LogSumExp*: se $x_1, \ldots, x_n$ sono una sequenza di numeri positivi, come lo sono le densità di probabilità, per definizione allora vale la seguente identità:

$$\begin{aligned}
\log\left(x_1 + \ldots + x_n\right) &= \log\left(\sum_{i=1}^{n} e^{\log x_i}\right) \\
&= \log\left(e^{\log\max x} \cdot e^{-\log\max x} \sum_{i=1}^{n} e^{\log x_i}\right) \\
&= \log\left(e^{\log\max x} \sum_{i=1}^{n} e^{\log x_i - \log\max x}\right) \\
&= \log\max x + \log\left(\sum_{i=1}^{n} e^{\log x_i - \log\max x}\right) \qquad (3.2)
\end{aligned}$$

Nel caso la sommatoria non sia di valori solo positivi le cose si complicano ulteriormente ma è ancora possibile sfruttare un trucco simile per minimizzare il rischio di *underflow* o *overflow*. Tale caso viene ad esempio riscontrato nell'implementazione del criterio di terminazione basato sul viriale (2.19). Infatti, i suoi tassi di cambio lungo la traiettoria possono avere sia segno positivo che negativo e la procedura richiede di farne la media rispetto ai rispettivi pesi multinomiali $\mathbb{P}(z|t) \propto e^{-H(z)}$. Questa operazione viene fatta ad ogni raddoppio di una sottotraiettoria e, per evitare problemi di approssimazione numerica dovuti all'esponenziale presente in $\mathbb{P}(z|t)$, è da svolgere in scala logaritmica. Riprendendo l'esempio più generale, questa volta con valori di $x_1, \ldots, x_n$ a segno variabile, per farlo è necessario salvarsi il segno di ogni singolo x_i e replicare la stessa operazione applicando il logaritmo ai valori assoluti, lo pseudocodice per tale operazione è presente in Algoritmo 1

Algoritmo 1: Algoritmo per il logaritmo della somma a segno variabile di esponenziali

input : vettore dei logaritmi dei valori assoluti: $\log|x_1|, \ldots, \log|x_n|$, vettore dei segni: $\text{sign}(x_1), \ldots, \text{sign}(x_n)$
output: logaritmo del valore assoluto della somma degli esponenziali: $LogAbsSumExp$, segno del valore assoluto della somma degli esponenziali: $SignSumExp$

$M = \max(\log|x_1|, \ldots, \log|x_n|)$; `// valore massimo`

$SumExp = \sum_{i=1}^{n} \text{sign}(x_i) e^{\log|x_i| - \log M}$; `// somma riscalata`

$LogAbsSumExp = \log M + \log SumExp$; `// scala logaritmica`

$SignSumExp = \text{sign}(SumExp)$; `// tenere conto del segno`

return $LogAbsSumExp$, $SignSumExp$; `// output`

3.2 Problematiche dell'HMC per distribuzioni a posteriori discontinue

Con discontinuità di una funzione si intende quando, in un dato punto, il limite da destra e il limite da sinistra non concordano. Un classico esempio si osserva nelle funzioni a gradini in corrispondenza di ciascun salto. Questo può accadere anche se l'argomento della funzione è definito in un aperto di $\mathbb{R}^d$, ovvero, in ottica bayesiana, quando tutti i parametri sono continui ma la verosimiglianza e/o la priori sono discontinue in alcuni punti dello spazio parametrico. In tali contesti l'HMC non sarebbe propriamente applicabile a livello teorico, questo perché i sistemi hamiltoniani si basano sulla geometria differenziale che, come suggerisce il nome, richiede la differenziabilità, assente nei punti di discontinuità della funzione. Tuttavia, se lo si legge come uno strumento per generare una *proposal* di un algoritmo Metropolis-Hastings non c'è nessun divieto, ma si perdono molte delle buone proprietà che lo caratterizzano. Il seguente esempio, tratto da Nishimura et al. (2020) ne fa capire il perché.

Assumendo $\theta \in \mathbb{R}$ e $U(\theta)$ un'energia potenziale discontinua, caratterizzata da

uno scalino in θ_0 per cui:

$$U(\theta) = \begin{cases} a & \text{se } \theta \in [\theta_0 - \delta, \theta_0) \\ b & \text{se } \theta \in [\theta_0, \theta_0 + \delta] \\ f(\theta) & \text{altrimenti} \end{cases} \tag{3.3}$$

dove $f(\theta)$ è una qualche funzione continua e differenziabile con soluzioni di continuità in $\theta_0 - \delta$ e $\theta_0 + \delta$. Un esempio grafico è riportato in Figura 3.1. Ne segue

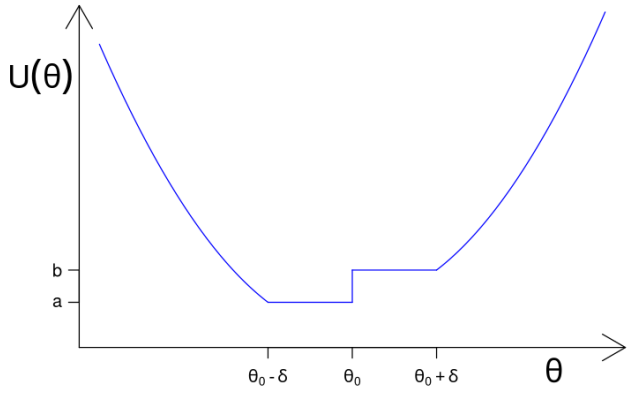

Figura 3.1: Grafico di una possibile energia potenziale definita come in (3.3).

che nell'intorno $[\theta_0 - \delta, \theta_0 + \delta]/\{\theta_0\}$ il gradiente dell'energia potenziale è pari a zero e quindi, seguendo le leggi di Hamilton (2.10), il momento coniugato per θ resterà costante lungo la traiettoria mentre il flusso per θ lo farà muovere secondo un moto rettilineo uniforme (visto che il momento non varia). Questo fa si che, dentro questo intorno, l'energia cinetica resti costante mentre quella potenziale cambi, passando da a a b o viceversa a seconda della direzione. Il venir meno della proprietà di preservazione dell'energia per le traiettorie esatte (ammesso che così definite abbiano un senso in questi contesti) è pienamente condiviso dai relativi integratori simplettici, comportando un limite superiore per il tasso di accettazione (1.4), che nell'HMC dipende dalla capacità delle traiettorie di conservare l'energia, per cui non è più vero che per $\epsilon \to 0$ ci si avvicina ad un campionamento indipendente dalla distribuzione microcanonica (2.13).

Per venire fuori da questo grattacapo, in Mohasel Afshar e Domke (2015) viene proposta una soluzione giustificata a livello fisico dal fenomeno di rifrazione e ri-

flessione della luce. Un fascio di luce emesso da un laser modifica la sua traiettoria in base alle superfici che incontra. Se puntato su un oggetto opaco allora la luce si riflette in maniera diffusa o speculare, dove la differenza tra queste due dipende dalle proprietà dell'oggetto in questione: un libro tenderà a deviare le particelle di luce lungo tutte le direzioni intorno alla sua superficie, risultandone illuminato, mentre uno specchio le rifletterà nel loro verso opposto. Si parla di rifrazione quando l'oggetto illuminato è trasparente e riesce a farsi attraversare dal fascio di luce, tipicamente alterando la traiettoria di ciascuna particella. Un classico esempio, osservabile nella vita di tutti i giorni, è dato dalla deformazione che un'immagine subisce se filtrata da una qualsiasi lente o, meglio ancora, da un bicchiere pieno d'acqua.

In Nishimura et al. (2020) viene proposta una teoria per le dinamiche hamiltoniane su spazi discontinui. Le assunzioni fatte dagli autori riguardano la possibilità di partizionare lo spazio di probabilità Θ in una serie di sottoinsiemi aperti Θ_k in cui l'energia potenziale risulti differenziabile, separati da delle frontiere di discontinuità $\partial\Theta_k$ in cui è possibile approssimare il gradiente tramite l'applicazione di una *derivata distribuzionale*:

$$\nabla U(\theta) = \lim_{\delta \to 0} \int \nabla U(\eta) \cdot \phi_\delta(\theta - \eta) d\eta$$

dove ϕ_δ è un qualsiasi nucleo la cui dispersione è regolata da δ. Tale gradiente segnala la direzione in cui l'energia potenziale cresce, la quale sarà per costruzione ortogonale alla frontiera $\partial\Theta_k$ nel punto in cui si trova. Questo comporta che, come suggerito in Mohasel Afshar e Domke (2015), il momento m in un punto su tale frontiera possa essere decomposto nella somma della sua componente perpendicolare alla frontiera $m_\perp$ con la sua componente parallela $m_\parallel$. Le leggi di Hamilton suggeriscono che il cambiamento infinitesimale di m avvenga nella direzione di $-\nabla U$, e quindi che vada ad intaccare m esclusivamente nella sua componente $m_\perp$. Il parallelismo con la rifrazione della luce risiede qui: immaginando la frontiera di discontinuità come la superficie di un oggetto su cui la particella di luce, la cui posizione è rappresentata da θ, arriva con un dato momento m. Solo se la componente

perpendicolare di quest'ultimo risulta sufficientemente grande, rispetto alla composizione di tale superficie, allora la particella è in grado di oltrepassarla, modificando la sua direzione perpendicolare ad essa, mentre in caso contrario viene riflessa nella direzione opposta.

Per capire in quale dei due casi ci si trova, e di quanto modificare il momento lungo la direzione perpendicolare, è sufficiente sfruttare la legge di conservazione dell'energia, già descritta nel paragrafo 2.1, per cui, indicando t_e l'istante in cui si incontra la discontinuità, dovrà valere la seguente relazione:

$$K(m(t_e^+)) - K(m(t_e^-)) = U(\theta(t_e^-)) - U(\theta(t_e^+))$$

indicando con $v = \frac{\nabla U}{\|\nabla U\|}$ la direzione in cui cresce l'energia potenziale si può definire l'aggiornamento infinitesimale per il momento come

$$m(t_e^+) = m(t_e^-) - \gamma \cdot v$$

con $\gamma > 0$ costante scalare da calcolare risolvendo:

$$K(m(t_e^-) - \gamma \cdot v) - K(t_e^-) = -\Delta U$$

con un po' di algebra si ottiene che questa corrisponde al rapporto tra v e

$$\left[m(t_e^-) - K^{-1}(\,-\Delta U + K(\,m(t_e^-)\,)\,)\right]$$

dove per costruzione si è assunto che tali vettori abbiano la stessa direzione.

Siccome $K^{-1} : \mathbb{R}^+ \to \mathbb{R}^d$, viene da sé che tale soluzione esista solamente se è rispettata la condizione:

$$K(m(t_e^-)) > \Delta U$$

che, coerentemente con quanto riportato sopra, sottende un'impossibilità della particella di superare la discontinuità se la sua energia cinetica prima del salto è troppo piccola da non poter compensare l'aumento in termini di energia potenziale che questo comporterebbe. In quest'ultimo caso, la particella dovrebbe rimbalzare sulla superficie determinata dalla frontiera di discontinuità e ciò sembra essere ancora una volta coerente con i calcoli, dal momento che se θ non si muove allora m ha

solo due opzioni: restare fermo, sancendo una irrealistica fine della traiettoria, oppure invertire il segno del proprio momento nella sua componente perpendicolare che, per la simmetria in zero tipicamente assunta per $K(m)$, assicura che il livello di energia resti inalterato e permette così di proseguire la traiettoria. In realtà, per come è stato formalizzato il problema, esistono infinite soluzioni, questo perché $K^{-1} : \mathbb{R}^+ \to \mathbb{R}^d$ è una *funzione multi-valore* che mappa un valore in $\mathbb{R}^+$ in un elemento qualsiasi della curva di livello corrispondente definita da $K(\cdot)$, ma solo una è coerente con l'assunzione che solo la componente $m_\perp$, che deve restare ortogonale a $m_\parallel$, venga modificata e l'unico modo per lasciare $\|m_\parallel + m_\perp\|^2$ inalterato è quello di cambiare il segno a $m_\perp$.

3.3 DHMC

Con la sigla DHMC si indica la tipologia di HMC proposta in Nishimura et al. (2020) a cui è stato assegnato il nome *Discontinous Hamiltonian Monte Carlo*. Tale approccio si applica ai casi in cui $\pi(\theta|y)$ è una funzione non continua per θ, ciò permette indirettamente di gestire anche la presenza di parametri discreti senza dover ricorrere ad una marginalizzazione, ma attraverso la diretta esplorazione di una densità a posteriori leggermente modificata. Questo avviene trasformando la legge di probabilità a priori per θ_J in una densità di probabilità definita come una funzione a gradini, ovvero costante a tratti, discontinua in corrispondenza dei salti.

Per semplicità possiamo assumere che i parametri in θ_J siano a priori indipendenti:

$$\mathbb{P}(\theta_J = x) = \prod_{j=1}^{k} \mathbb{P}(\theta_j = x_j)$$

e trattare la trasformazione nel continuo di queste leggi una ad una. Per ciascuna θ_j si può quindi immaginare un sottoinsieme aperto di $\mathbb{R}$, fittizio, in cui definire la corrispondente variabile aleatoria θ_j^* continua. La relazione tra queste due è stabilita da una sequenza di valori, $\{a_n\}$, che definisce gli intervalli (a_n, a_{n+1}) in cui trasferire la massa di probabilità dell'n-esima modalità di θ_j. Per calcolare la densità uniforme dentro tale intervallo è sufficiente dividere tale massa per il suo

volume: $a_{n+1} - a_n$. La legge di probabilità a priori per la nuova variabile risulta essere quindi:

$$\pi(\theta_j^*) = \sum_n \mathbb{P}(\theta_j = n) \cdot \mathcal{I}(a_n < \theta_j^* \leq a_{n+1}) \cdot \frac{1}{a_{n+1} - a_n} \tag{3.4}$$

Un modo analogo per visualizzare meglio questa trasformazione passa attraverso la funzione di ripartizione, che per la distribuzione a priori originale è a gradini, mentre per la versione trasformata sarà una spezzata con soluzione di continuità nei punti definiti dai vari $\{a_n\}$, i cui segmenti avranno un'inclinazione che dipende dall'ampiezza dell'intervallo $(a_n, a_{n+1}]$ considerato e dalla relativa massa di probabilità originale: $\mathbb{P}(\theta_j = n)$. In Figura 3.2 è riportato un esempio per la seguente legge di probabilità discreta:

$$\mathbb{P}(X = 1) = \mathbb{P}(X = 3) = 0.25\,,\, \mathbb{P}(X = 2) = 0.5$$

in cui si sceglie, a scopo esplicativo, la seguente sequenza:

$$a_{-1} = -\infty\ ,\ a_0 = 0\ ,\ a_1 = 2\ ,\ a_2 = 4\ ,\ a_3 = 5\ ,\ a_4 = 8\ ,\ a_5 = \infty$$

Si può notare come, nel grafico di destra, l'intervallo $(4, 5]$ sia caratterizzato da

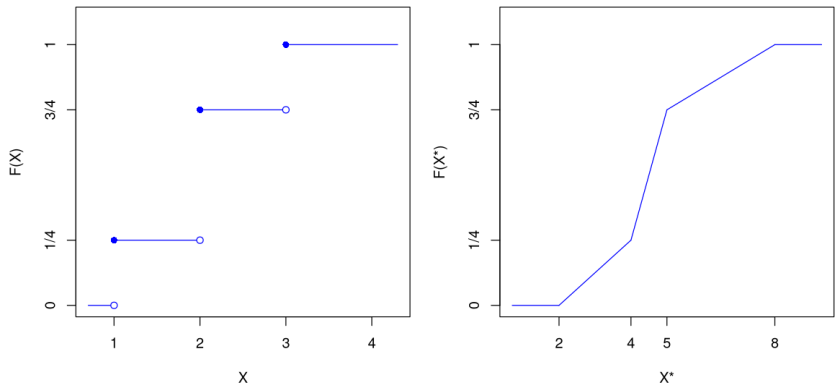

Figura 3.2: Funzioni di ripartizione per X e X^*.

una pendenza maggiore sia perché la massa di probabilità della sua modalità di riferimento, 3, è la più grande, sia perché si è scelto un intervallo più stretto rispetto alle altre modalità di X. Tale approccio, quindi, dipende fortemente dalla scelta della sequenza $\{a_n\}$ che mappa i due domini, inoltre, seppur funzionante, è meno

giustificabile se le modalità di θ_j non hanno un ordinamento ben preciso, dal momento che le prestazioni dell'algoritmo potrebbero essere influenzate dalla selezione di uno o di un altro. Come si nota dal grafico di destra, nella specificazione di $\{a_n\}$, particolare attenzione va posta sugli estremi, dal momento che qualsiasi intervallo con limite superiore o inferiore pari a $\pm\infty$ risulterà avere un volume infinito. Come conseguenza di ciò, θ_j^* non potrà essere una variabile con densità diversa da zero su tutto $\mathbb{R}$, a meno che il suo supporto originario non sia $\mathbb{Z}$, ma questo è facilmente aggirabile applicando un ulteriore cambio di variabile, ad esempio sfruttando la trasformazione logistica per cui:

$$\begin{aligned} x \in (a,b) \quad x \sim \pi_x(x) \\ x' = \log\left(\frac{x-a}{b-x}\right) \implies x = a + \frac{b-a}{1+e^{-x'}} \\ \pi_{x'}(x') = \pi_x(x') \cdot \left(\frac{1}{1+e^{-x'}}\right) \cdot \left(\frac{e^{-x'}}{1+e^{-x'}}\right) \end{aligned}$$

In questo caso è stata utilizzata la regola del cambio di variabile per leggi di probabilità assolutamente continue, che passa attraverso l'utilizzo del determinante dello Jacobiano. Tale approccio è giustificato dal fatto che, anche se la densità di probabilità incontra delle discontinuità, la sua funzione di ripartizione è comunque continua sia da destra che da sinistra.

3.3.1 Integratore simplettico gaussiano

Per approssimare le dinamiche hamiltoniane discontinue, in Mohasel Afshar e Domke (2015) viene proposta una funzione di disintegrazione gaussiana con matrice di massa M identità, per cui si ottiene che la relativa energia cinetica è scrivibile come:

$$K(m) = \frac{\|m\|^2}{2}$$

In corrispondenza di una discontinuità, l'aggiornamento della componente $m_\perp$ viene dettato dalla legge di conservazione dell'energia, secondo cui:

$$\frac{\|m'\|^2}{2} - \frac{\|m\|^2}{2} = -\Delta U$$
$$\|m_\parallel + m'_\perp\|^2 = \|m_\parallel + m_\perp\|^2 - 2\Delta U$$
$$\|m_\parallel\|^2 + \|m'_\perp\|^2 = \|m_\parallel\|^2 + \|m_\perp\|^2 - 2\Delta U$$
$$\|m'_\perp\|^2 = \|m_\perp\|^2 - 2\Delta U$$

Siccome $m'_\perp = \frac{m_\perp}{\|m_\perp\|} \cdot k$, ne segue che:

$$\frac{\|m_\perp\|^2}{\|m_\perp\|^2} \cdot k^2 = \|m_\perp\|^2 - 2\Delta U$$

per cui

$$m'_\perp = \frac{m_\perp}{\|m_\perp\|} \cdot \sqrt{\|m_\perp\|^2 - 2\Delta U}$$

Questo rappresenta l'aggiornamento, sulla frontiera, della componente perpendicolare del momento nel caso di rifrazione, possibile solamente se $\|m_\perp\|^2 > 2\Delta U$. In caso contrario, per poter continuare la traiettoria, è necessario negare tale momento: $m'_\perp = -m_\perp$, ottenendo così una riflessione.

L'integratore simplettico proposto dagli autori è riassumibile dallo pseudo-codice mostrato in Algoritmo 2.

Algoritmo 2: Algoritmo proposto da Mohasel Afshar e Domke (2015)

input : valore corrente della catena: θ, energia potenziale: U, numero di passi di integrazione: L, ampiezza dei passi di integrazione: ϵ
output: valore successivo per la catena: θ'

$m \sim N(0, I_d)$;
$H \leftarrow U(\theta) + \|m\|^2/2$; `// livello di energia corrente`
for $i = 1$ **to** L **do**
  $m \leftarrow m - \epsilon \nabla U(\theta)/2$; `// mezzo passo momento`
  $t \leftarrow 0$; `// tempo di integrazione`
  while $t < \epsilon$ **do**
    $t_e \leftarrow$ tempo per raggiungere la prossima discontinuità;
    if $t + t_e < \epsilon$ **then**
      `// nessuna discontinuità incontrata`
      $\theta' \leftarrow \theta + (\epsilon - t)m$; `// restante passo posizione`
      $t \leftarrow \epsilon$; `// termina passo`
    else
      if then
        `// discontinuità incontrata`
        $\theta' \leftarrow \theta + t_e m$; `// passo oltre discontinuità`
        $\Delta U \leftarrow U(\theta') - U(\theta)$; `// differenza potenziale`
        $m_\parallel, m_\perp \leftarrow decomponi(m, \partial\Theta_k)$; `// decomponi momento`
        if $\|m_\perp\|^2 > 2\Delta U$ **then**
          `// rifrazione`
          $m_\perp \leftarrow \sqrt{\|m_\perp\|^2 - 2\Delta U} \cdot \frac{m_\perp}{\|m_\perp\|}$;
        else
          `// riflessione`
          $m_\perp \leftarrow -m_\perp$;
        end
        $m \leftarrow m_\parallel + m_\perp$; `// ricomponi momento`
      end
    end
    $\theta \leftarrow \theta'$;
    $t \leftarrow t_e$;
  end
  $m \leftarrow m - \epsilon \nabla U(\theta')/2$; `// mezzo passo momento`
end
$H' \leftarrow U(\theta') + \|m\|^2/2$; `// nuovo livello di energia`
if $Unif(0, 1) < e^{H - H'}$ **then**
  return θ'; `// accetta proposal`
else
  return θ; `// rifiuta proposal`
end

3.3.2 Integratore simplettico laplaciano

In Nishimura et al. (2020) viene mostrato come l'utilizzo di una funzione di disintegrazione di Laplace comporti un importante vantaggio nel contesto della risoluzione delle dinamiche hamiltoniane discontinue. Il caso gaussiano necessita di calcolare la differenza potenziale prima e dopo ciascuna frontiera di discontinuità che viene attraversata dalla particella in un dato passo di integrazione ϵ. Questi istanti, a seconda dell'ampiezza di Θ_J, dell'estensione del *Typical Set* della distribuzione a posteriori su esso e del valore di ϵ, potrebbero essere molteplici, incrementando il costo computazionale di ogni singola iterazione. Il metodo proposto dagli autori permette di svolgere un tale calcolo al più un volta per ogni passaggio dell'integratore simplettico considerato, come fosse un normale HMC.

Si assuma che i momenti siano a priori indipendenti, ciascuno con distribuzione di Laplace centrata in zero e con una scala potenzialmente differente:

$$m_1 \dots, m_J \text{ indipendenti} \quad \pi(m_j) \propto e^{-\frac{|m_j|}{\sqrt{M_{jj}}}}$$

dove M è sempre interpretabile come la matrice di massa per le variabili discrete, che viene assunta essere diagonale. In questo caso le equazioni di Hamilton sono le seguenti:

$$\begin{cases} \frac{dm}{dt} & = -\nabla U(\theta) \\ \frac{d\theta}{dt} & = M^{-1/2} \frac{m}{|m|} \end{cases} \tag{3.5}$$

Questo vuol dire che le variazioni nella traiettoria della posizione non dipendono dall'ampiezza del momento ma solo dal suo segno e che quindi θ segue un moto rettilineo uniforme fintanto che questo non cambia. All'interno di un intervallo $[t, t+\epsilon]$ in cui $\frac{m}{|m|}$ è costante, indipendentemente dal numero di discontinuità in esso contenuto, l'integrazione da t a $t+\epsilon$ porta al medesimo risultato rispetto alla sequenza di integrazioni che si soffermano su ciascun punto di discontinuità. Ovvero la traiettoria approssimata risulta essere la stessa sia per la posizione che per il momento, dove quest'ultimo viene calcolato ogni volta sfruttando la legge di conservazione dell'energia.

L'integratore simplettico proposto da Nishimura et al. (2020) applica tale modifica alla versione gaussiana per densità a posteriori discontinue vista nel paragrafo precedente. In particolare, viene preferito un approccio *coordintate-wise* che ha un diretto collegamento con il Guided Walk Metropolis descritto nel paragrafo 1.4. Tale scelta è ragionevole se si pensa di applicare il metodo a spazi di probabilità resi continui attraverso l'utilizzo di distribuzioni a priori a gradini, dal momento che le frontiere di discontinuità sono tutte perpendicolari agli assi cartesiani e quindi $m_\perp$ corrisponderà sempre ad una data coordinata di m, salvo nel caso in cui θ finisca in un'intersezione di due o più frontiere. Gli autori dimostrano che quest'ultima situazione non è verificabile in quanto l'insieme di questi punti è interpretabile come un'unione finita di iperpiani di dimensione inferiore allo spazio di probabilità originale e, pertanto, con una misura di Lebesgue pari a zero. Qualora si abbia a che fare con una distribuzione non a gradini, ma comunque discontinua, il metodo è comunque un valido generatore di proposte Metropolis-Hastings, perde solamente una parte della sua giustificazione teorica. Questa validità generale lo rende appetibile anche al trattamento dei parametri continui, θ_I, ma, come suggerito dagli autori stessi, ciò non produrrebbe risultati migliori dell'HMC classico, che aggiorna le coordinate in blocco.

Per generare una *proposal*, viene proposto un algoritmo che è dato dall'unione del *Leapfrog* per le componenti continue e di un integratore simplettico specifico per quelle discrete. Per assicurarne la reversibilità, quest'ultimo viene incastrato tra due aggiornamenti di θ_I, m_I e, per evitare di ripetere tale schema a specchio anche per θ_J, m_J, data la loro natura *coordinte-wise*, si preferisce permutarne l'ordine di aggiornamento, definendo così una mappa reversibile solo in probabilità, ovviamente dopo che questa è stata dotata dell'apposito operatore (2.14) per il cambio di segno dei momenti.

L'aggiornamento di una coppia θ_j, m_j, per $j = 1, \dots, k$, è basato sulle equazioni di Hamilton (2.10) per la posizione e sulla legge di conservazione dell'energia (2.7) per il momento coniugato: l'applicazione di tali leggi in questo contesto fornisce i

seguenti risultati:

$$\theta_j(t+\epsilon) = \theta_j(t) + \epsilon \cdot \frac{\text{sign}(m_j(t))}{\sqrt{M_{jj}}} \tag{3.6}$$

$$\frac{|m_j(t+\epsilon)|}{\sqrt{M_{jj}}} - \frac{|m_j(t)|}{\sqrt{M_{jj}}} = -\Delta U$$

$$|m_j(t+\epsilon)| = |m_j(t)| - \sqrt{M_{jj}} \cdot \Delta U$$

$$m_j(t+\epsilon) = \text{sign}(m_j(t)) \cdot \left[|m_j(t)| - \sqrt{M_{jj}} \cdot \Delta U\right] \tag{3.7}$$

dove, per arrivare a (3.7), si è sfruttato il fatto che $m_j(t+\epsilon)$ e $m_j(t)$ devono avere lo stesso segno. Questo è coerente con le dinamiche hamiltoniane per cui è solo l'energia potenziale a modificare m, ovvero che un ΔU negativo implica una fase di discesa nella valle definita dall'energia potenziale e quindi una crescita di momento m_j lungo il suo orientamento (positivo o negativo), mentre un valore positivo di ΔU è sintomo di una fase di salita che avvicinerà m_j a zero. Anche in questo caso la condizione necessaria per modificare la posizione richiede che l'energia cinetica prima del passo sia maggiore della differenza potenziale generata da questo. Qualora non lo fosse, l'unica soluzione che fa proseguire la traiettoria è data dalla negazione del momento in questione. Qui risiede il parallelismo con il *Guided Walk Metropolis* descritto nel paragrafo 1.4, infatti è possibile immaginare la condizione $K > \Delta U$ come una versione deterministica, condizionatamente alla traiettoria, della probabilità di accettazione (1.4) che, se rifiutata, comporta una non modifica del parametro *originale* θ_j, e un'inversione del segno di quello *aumentato* m_j, che guiderà la catena, o in questo caso la traiettoria, verso zone a più alta densità a posteriori.

In Figura 3.3 è presentato un esempio grafico: per un'energia potenziale discontinua, la particella nel grafico di sinistra inizia il suo passo dotata di un notevole momento, rappresentato dalla freccia rossa, che le permette di oltrepassare ben 3 frontiere di discontinuità con un solo salto. Questo perché la differenza potenziale generata è minore del gradiente della sua energia cinetica. Adesso però, la particella ha perso troppa energia cinetica e il suo momento è diminuito a tal punto che nel passo successivo, mostrato nel grafico di destra, non riesce a compensare la

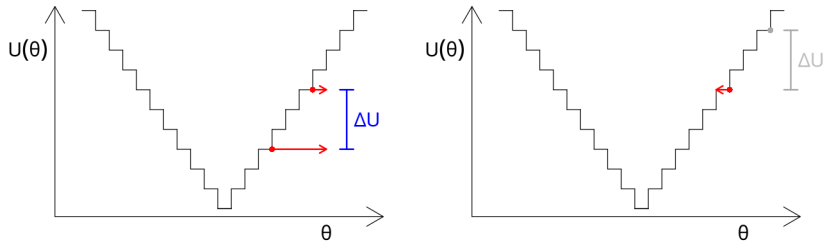

Figura 3.3: Particella che si muove in uno spazio discontinuo secondo la procedura proposta da Nishimura et al. (2020).

differenza potenziale ed è costretta a riflettere il suo momento sulla discontinuità. Nel passo successivo ΔU sarà una quantità negativa, perciò la rifrazione risulterà assicurata e la particella accrescerà il suo momento prendendo velocità in discesa.

Come puntualizzato dagli autori, l'approccio *coordinate-wise* è solo uno dei tanti *metodi di splitting* con cui è possibile approssimare la soluzione di un sistema di equazioni differenziali (McLachlan e Quispel, 2002), la cui bontà di approssimazione è inversamente proporzionale ad ϵ. In questo caso tuttavia c'è una differenza rilevante rispetto al metodo discusso in Mohasel Afshar e Domke (2015), riassunto dall'Algoritmo 2. Quest'ultimo, indipendentemente da ciò che accade nel mentre, continua l'approssimazione della traiettoria fino al tempo ϵ prestabilito, mentre il metodo qui esposto non riesce a mimare lo stesso comportamento, ad esempio: se tra t e $t + \epsilon$ sussistono tre punti di discontinuità equispaziati, e in corrispondenza del secondo l'energia cinetica non è sufficiente a garantire il salto, comportando una negazione del momento, un buon integratore dovrebbe effettuare le seguenti transizioni:

$$\theta_j(t) \to \theta_j\left(t + \epsilon/3\right) \to \theta_j\left(t + 2\epsilon/3\right) \to \theta_j\left(t + \epsilon/3\right)$$

mentre il metodo basato sulla disintegrazione laplaciana si limiterà a negare il momento alla prima transizione, restando in $\theta_j(t)$ e non muovendosi in $\theta_j(t + \epsilon/3)$ come dovrebbe. L'unico modo per risolvere questo problema sarebbe quello di considerare il movimento in corrispondenza di ogni discontinuità, ma questo farebbe perdere le buone proprietà di risparmio computazionale che ne motivano l'utilizzo. Conseguentemente tale integratore, salvo per ϵ molto piccoli che limitano il numero

di possibili discontinuità al loro interno, non è da considerarsi del tutto affidabile se si incombe in un cambio di momento, in quanto sfaserà i tempi con cui si approssima la soluzione della traiettoria. Ciononostante, siccome è costruito in maniera tale da assicurare la conservazione dell'energia, resterà sempre e comunque all'interno del livello di energia relativo a questa. Inoltre, gli autori hanno dimostrato la *simpletticità* di tale integratore, a differenza della controparte gaussiana proposta in Mohasel Afshar e Domke (2015), e ciò ne fa ereditare un errore di integrazione locale e globale che resta piccolo anche per lunghi periodi di integrazione. In particolare, l'errore locale di un singolo passo di ampiezza ϵ, se la traiettoria non incontra discontinuità, è $O(\epsilon^3)$ come il *Leapfrog*, mentre in caso di discontinuità è $O(\epsilon^2)$. L'errore globale, dopo L passi, risulta essere $O(\epsilon^2 \cdot D)$ dove D è il numero di discontinuità incontrate. Similmente al *Leapfrog*, tale errore non dipende direttamente da L, tuttavia c'è una dipendenza indiretta, visto che D verosimilmente è proporzionale ad esso.

Lo pseudo codice per il metodo proposto in Nishimura et al. (2020) è riportato in Algoritmo 3, mentre in appendice A.5 è riportata la dimostrazione del fatto che questa procedura è una mappa che non altera il volume e quindi la misura originaria.

Algoritmo 3: Algoritmo proposto da Nishimura et al. (2020)

input : valore corrente della catena: θ, energia potenziale: U, numero di passi di integrazione: L, ampiezza dei passi di integrazione: ϵ, numero di parametri discreti: k
output: valore successivo per la catena: θ'

$m_I \sim N_{d-k}(0, M_I)$; // momenti gaussiani
$m_J \sim Laplace_k(0, M_J)$; // momenti laplaciani
$H \leftarrow U(\theta) + \|m_I\|^2_{M_I}/2 + M_J^{-1/2}\|m_J\|_1$; // energia corrente
for $i = 1$ **to** L **do**
  $m_I \leftarrow m_I - \epsilon \nabla_I U(\theta)/2$; // mezzo passo momento continuo
  $\theta_I \leftarrow \theta_I + \epsilon M_I^{-1} m_I/2$; // mezzo passo posizione continuo
  // integrazione parametri discreti
  for j **in** $permuta(k)$ **do** // permuta l'ordine
    $\theta^* \leftarrow \theta$;
    $\theta^*_j \leftarrow \theta^*_j + \text{sign}(m_j) M_{jj}^{-1/2} \epsilon$; // passo posizione discreto
    $\Delta U \leftarrow U(\theta^*) - U(\theta)$; // differenza potenziale
    if $M_{jj}^{-1/2}|m_j| > \Delta U$ **then**
      // rifrazione particella
      $\theta_j \leftarrow \theta^*_j$;
      $m_j \leftarrow m_j - \text{sign}(m_j) M_{jj}^{1/2} \Delta U$; // momento discreto
    end
    // riflessione particella
    $m_j \leftarrow -m_j$;
  end
  $\theta_I \leftarrow \theta_I + \epsilon M_I^{-1} m_I/2$; // mezzo passo posizione continuo
  $m_I \leftarrow m_I - \epsilon \nabla_I U(\theta)/2$; // mezzo passo momento continuo
end
$H' \leftarrow U(\theta) + \|m_I\|^2_{M_I}/2 + M_J^{-1/2}\|m_J\|_1$; // energia nuova
if $Unif(0, 1) < e^{H-H'}$ **then**
  return θ'; // accetta proposal
else
  return θ; // rifiuta proposal
end

3.3.3 Specifiche DHMC

In Nishimura et al. (2020) sono discusse alcune ulteriori specifiche per l'Algoritmo 3, in particolare viene suggerito, in accordo con la teoria classica dell'HMC, di sfruttare la matrice di massa per facilitare l'esplorazione della distribuzione a posteriori. Per quanto riguarda i parametri continui, basati sul momento gaussiano, questo effettivamente equivale all'applicazione di una riparametrizzazione dello spazio di probabilità originale come in (2.17), mentre per i parametri discreti si perde questo parallelismo perché il momento è funzione di ΔU che, al contrario di ∇U, è invariante rispetto a riparametrizzazioni, tuttavia vale ancora l'interpretazione di (2.18) secondo cui la matrice di massa sancisce la velocità di movimento lungo ciascuna coordinata. Viene discussa l'opzione di una matrice densa, invece che diagonale, anche per i parametri discreti, tuttavia questo richiede di comporre ciascun momento m_j come una somma pesata di variabili di Laplace con parametro di scala pari agli autovalori della matrice di massa e con pesi dati dai rispettivi autovettori, aumentando sia la complessità dei conti che il costo computazionale dell'algoritmo derivante.

Un altro aspetto di interesse riguarda la specificazione del passo di integrazione ϵ. Tale quantità dovrebbe essere la stessa per entrambi i gruppi di parametri, se si vuole conservare l'interpretazione fisica, ma non c'è alcun divieto nella costruzione di una *proposal* che li consideri differenti. Un modo analogo per farlo, che conserva l'aspetto teorico, è attraverso la specificazione della matrice M_J che, moltiplicata per un ϵ costante, definisce un passo più o meno lungo per ogni coordinata. Siccome l'aggiornamento dei parametri discreti non altera il livello di energia, nel caso in cui ci fossero solo questi si avrebbe un tasso di accettazione Metropolis costante pari ad 1. Si perderebbe così un'importante diagnostica sulla sua calibrazione, aspetto di vitale importanza per determinare la velocità di esplorazione e l'accuratezza delle traiettorie. Gli autori propongono di utilizzare come euristica il tasso di rifrazione che si incontra per ciascun parametro, dal momento che il discrimine con la

riflessione è interpretabile come un passo di Random Walk Metropolis:

$$M_{jj}^{-1/2}|m_j| > U(\theta') - U(\theta)$$
$$e^{-M_{jj}^{-1/2}|m_j|} < e^{U(\theta)-U(\theta')}$$
$$\eta < \frac{\pi(\theta')}{\pi(\theta)}$$

dove è facile dimostrare, attraverso un cambio di variabile, che

$$m_j \sim Laplace(0, \sqrt{M_{jj}}) \implies \eta = e^{-M_{jj}^{-1/2}|m_j|} \sim Unif(0,1)$$

Pertanto, coerentemente con quanto detto nel paragrafo 1.3, sembra sensato assicurare un tasso tra 0.2 e 0.4. Infatti, siccome tale tasso cresce man mano che ϵ diminuisce, un valore di ϵ troppo piccolo non è desiderabile, in quanto rallenterebbe l'esplorazione, ma neanche uno troppo grande e in questo caso il motivo è duplice: da un lato perché l'approssimazione delle dinamiche hamiltoniane nel discontinuo peggiora inevitabilmente, dall'altro perché è possibile trovarsi nella spiacevole situazione in cui la particella si incastra in una *vallata* dell'energia potenziale, da cui non riesce ad uscire perché il passo di integrazione gli propone salti troppo lunghi che determinano una differenza potenziale che non è in grado di colmare.

Infine, viene osservato come l'aggiornamento per la posizione, definito in (3.6), comporti un movimento ad intervalli regolari dei parametri discreti, ovvero una riducibilità della catena risultante da questo algoritmo che ne inficia l'inferenza. Fortunatamente, questo aspetto è aggirabile considerando una campionamento casuale uniforme di ϵ in un intorno del suo valore desiderato, tale rimedio è applicato anche per altri motivi, già discussi nel paragrafo 2.4.3.

3.4 Implementazione efficiente: libreria XDNU-TS

L'algoritmo DHMC definisce già la funzione di disintegrazione e l'integratore simplettico da utilizzare. Gli autori, poi, forniscono suggerimenti su come costruire un metodo per ottenere un passo di integrazione ottimale e consigliano l'uso di una

matrice di massa densa, se il costo computazionale lo permette, per i parametri continui e diagonale per quelli discreti. Nulla di interessante viene detto riguardo al numero di passi di integrazione L se non di ricercarlo all'interno di una griglia. Come già descritto alla fine del paragrafo 2.3.2, questa soluzione è nota essere sub-ottimale perché L dovrebbe poter essere libero di variare a seconda della geometria della distribuzione a posteriori e un approccio euristico utilizzabile consiste nel campionare anche L uniformemente intorno al valore ottimo trovato, in maniera analoga a come si fa per ϵ. Questa però è una soluzione sicuramente meno efficiente di lasciare che sia la distribuzione obiettivo a comunicarne i loro valori ottimi di volta in volta: nel caso classico continuo, per L si ricorreva al NUTS (Hoffman et al., 2014), già discusso nel paragrafo 2.4, mentre per ϵ una possibilità è data dall'utilizzo del RHMC Betancourt (2013*b*) brevemente accennato nel paragrafo 2.3.2.

In questa sede si vuole proporre un metodo per identificare il tempo di integrazione ottimale per ogni traiettoria approssimata dal DHMC e implementare a partire da esso un algoritmo simile a quello utilizzato da STAN, basato sulla costruzione progressiva della traiettoria attraverso il raddoppio della sua cardinalità, dove ogni passo verrebbe svolto utilizzando l'Algoritmo 3 invece che il *Leapfrog*. Il metodo MCMC risultante sarebbe così applicabile anche a densità a posteriori discontinue e quindi al caso in cui una porzione dei parametri è di natura discreta, attraverso la trasformazione descritta in (3.4). Tale algoritmo è reso disponibile in forma di libreria R (Team, 2024) scritta prevalentemente in Rcpp (Eddelbuettel, 2013) chiamata *XDNUTS*, la cui sigla sta per *eXhaustive-Discontinuous-No-U-Turn-Sampler*, ad indicare il tipo di criteri di terminazione disponibili. Le opzioni sono 3:

1. utilizzare una lunghezza prefissata L, da specificare, come fosse un HMC classico. Come appena ricordato, conviene in realtà campionare da un intervallo uniforme centrato in questa, la cui ampiezza è anch'essa specificabile;

2. utilizzare il metodo basato sull'esaurimento del *viriale*, il quale necessita della calibrazione della soglia τ, che deve essere svolta dall'utente;

3. utilizzare il metodo basato sul prodotto interno, quello proposto in questa sede è il seguente, riassumibile dalle seguenti due condizioni

$$\left[\sum_{i\in st} M_I^{-1} m_i^I\right]^T m_{st_+}^I + \left[\sum_{i\in st} M_J^{-1/2} \frac{m_i^J}{|m_i^J|}\right]^T \frac{m_{st_+}^J}{|m_{st_+}^J|} < 0$$
$$\left[\sum_{i\in st} M_I^{-1} m_i^I\right]^T m_{st_-}^I + \left[\sum_{i\in st} M_J^{-1/2} \frac{m_i^J}{|m_i^J|}\right]^T \frac{m_{st_-}^J}{|m_{st_-}^J|} > 0 \tag{3.8}$$

dove st indica una sotto-traiettoria qualsiasi, costruita con il metodo spiegato nel paragrafo 2.4.1, mentre st_- e st_+ sono rispettivamente i suoi estremi sinistro e destro. Quando entrambe le condizioni sono rispettate per una qualsiasi traiettoria la procedura di raddoppio viene fermata restituendo il valore successivo della catena.

Per farlo rientrare nella definizione di prodotto interno, si è dovuto prendere il segno dei momenti m_J, anziché i veri e propri momenti, in questo modo è facile dimostrare che $m' = (m_I, \text{sign}(m_J))$ è un elemento di uno spazio vettoriale euclideo a cui è possibile assegnare la seguente metrica:

$$M'^{-1} = \begin{pmatrix} M_I^{-1} & 0 \\ 0 & M_J^{-1/2} \end{pmatrix}$$

con $M_J^{-1/2}$ matrice diagonale. L'utilizzo di $\text{sign}(m_J)$, anziché m_J, risulta essere coerente anche con il funzionamento dell'Algoritmo 3 e, più in generale, con le traiettorie hamiltoniane caratterizzate da funzione di disintegrazione di Laplace, dal momento che, per le componenti discontinue, la direzione prossima è definita esclusivamente dall'orientamento del vettore dei suoi momenti, opportunamente riscalato per la sua metrica $M_J^{-1/2}$. Definito il prodotto interno per lo spazio di questi nuovi momenti come:

$$\langle x', y' \rangle_{M'^{-1}} = x'^T M'^{-1} y'$$

siamo in grado di identificare l'istante in cui la traiettoria collassa su se stessa in maniera analoga a quanto descritto nel paragrafo 2.4.1 e in (3.8). Come supporto

per visualizzarne il funzionamento, sono qui riportati dei grafici analoghi a quelli presenti in Figura 2.3, che mostrano l'andamento della traiettoria per la stessa energia potenziale definita in (2.20) partendo dal medesimo punto ma questa volta utilizzando l'Algoritmo 3 e considerando tutte le coordinate come portatrici di discontinuità. Le due traiettorie non potranno mai essere considerate come diverse approssimazioni dello stesso flusso, questo perché fanno riferimento a sistemi hamiltoniani differenti per cui la diversa specificazione di $K(m)$ implica delle diverse leggi del moto per le particelle.

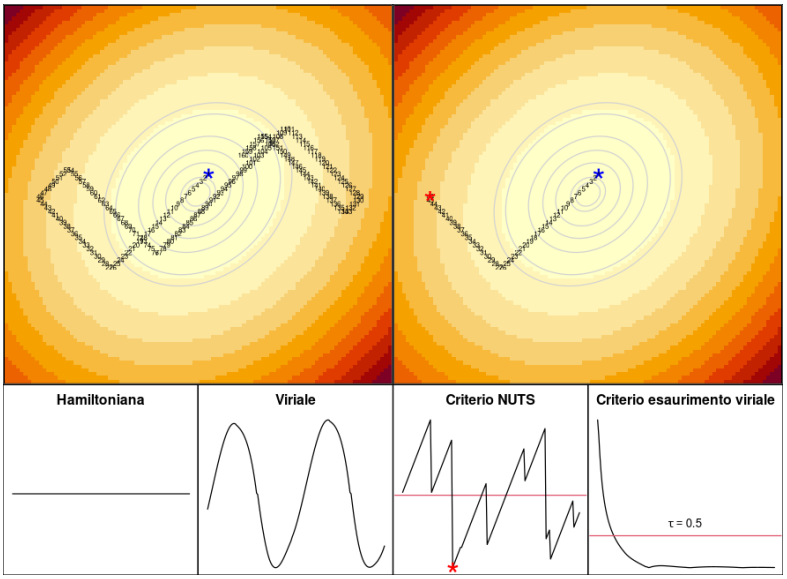

Figura 3.4: Traiettoria dell'Algoritmo 3 per $U(x)$ definito in (2.20) con tutte le componenti discontinue.

In Figura 3.4, in alto a sinistra, è riportata la traiettoria approssimata sulle curve di livello di (2.20), portata avanti fino a che non sembra ritornare al punto di partenza, utilizzando l'Algoritmo 3 e trattando entrambe le componenti come discontinue. Al disotto di questo grafico sono disegnate le serie storiche del livello di energia e del viriale. Per come è concepito l'algoritmo, l'Hamiltoniana del sistema resta sempre costante, tuttavia, per quanto mostrato in pagina 80, questo non implica che il flusso sia approssimato esattamente. Per quanto riguarda il viriale, nonostante la natura *dicreta* della traiettoria, risulta comunque seguire un andamento continuo e uniformemente limitato. Nel grafico in alto a destra è mostrata la

medesima traiettoria, ma portata avanti soltanto fino al raggiugimento del criterio di terminazione (3.8). Al disotto di questo grafico sono riportate le serie storiche del valore che ciascun criterio assume con il procedere dell'integrazione delle equazioni.

L'interpretazione di tali criteri resta invariata: per quanto riguarda la rilevazione di uno *U-Turn*, il fatto che ∇K sia costante in ogni coordinata, comporta che ogni riflessione determini un annullamento nella direzione cumulata di un passo non riflesso precedente. Questo fa si che, ad esempio, una traiettoria che procede a *zig-zag* in avanti, alternando una riflessione a una rifrazione per una data coordinata mentre un'altra si vede procedere dritta, non venga considerata erroneamente come una che sta collassando su se stessa. Affinché ciò avvenga, è necessario che si osservi globalmente un cambiamento di rotta: ad esempio se $\text{sign}\left(\int_t \nabla K\right)_j = -\text{sign}(m_t^j)\ \forall j$ è sicuro che il prodotto interno sarà negativo, mentre nei casi più generali, nel determinare la somma delle componenti, sarà rilevante anche il numero di rifrazioni fatte per ciascuna coordinata lungo la traiettoria. Per quanto concerne il criterio basato sull'esaurimento del viriale, questo fa sempre leva sul fatto che per traiettorie non critiche la media temporale del suo tasso di cambio tende a zero, per cui è possibile specificare una soglia τ in maniera analoga a quanto descritto nel paragrafo 2.4.1.

In Figura 3.5 sono mostrate le medesime quantità presenti nelle Figure 2.3 e 3.4, ma questa volta applicando l'Algoritmo 3 che considera soltanto la seconda componente come portatrice di discontinuità. Anche in questo caso valgono i risultati visti precedentemente e si nota come il criterio di terminazione basato su (3.8) sembri conciliare bene entrambe le componenti: continua e discontinua. Il codice R per riprodurre questi grafici è presente in appendice C.1.

Per la calibrazione del passo di integrazione ϵ, come suggerito dagli autori, è possibile utilizzare, in analogia a quanto fatto in STAN, un metodo di ottimizzazione

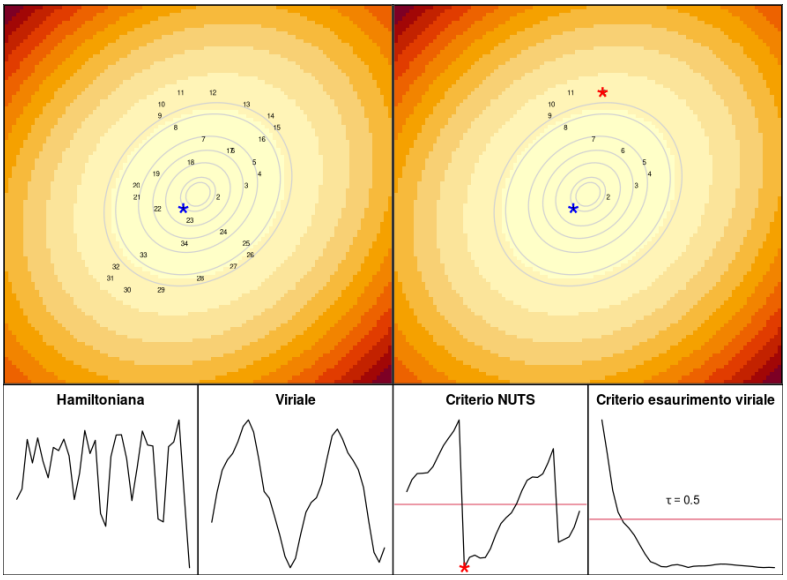

Figura 3.5: Traiettoria dell'Algoritmo 3 per $U(x)$ definito in (2.20) con solo una componente discontinua.

stocastica atto a trovare lo zero della media Monte Carlo della seguente funzione:

$$H_i = \left[\frac{1}{|t_i|} \sum_{z \in t_i} \min\left\{1, e^{H(z_0)-H(z)}\right\} - \delta_1 \right] + \left[\frac{1}{k} \sum_{j=1}^{k} \frac{1}{|t_i|} \sum_{z \in t_i} \mathcal{I}\left(M_{jj}^{-1/2} |m_j| > \Delta_j U(z) \right) - \delta_2 \right] \tag{3.9}$$

come ad esempio il *Dual Averaging* proposto in Nesterov (2009), dove:

- δ_1 rappresenta il tasso desiderato per l'accettazione complessiva delle *proposal* e viene confrontato con la media delle probabilità di accettazione Metropolis lungo la traiettoria t_i-esima. Sembra ragionevole lasciare il valore 0.8 utilizzato dal NUTS come base, dal momento che l'aggiornamento delle componenti θ_J avviene conservando esattamente l'energia;

- δ_2 rappresenta il tasso desiderato di rifrazione per ciascun parametro discreto, e viene confrontato con la sua controparte empirica calcolata lungo tutta la traiettoria. Dal momento che, come mostrato in precedenza, questo è interpretabile come il tasso di accettazione di un passo Metropolis, sembra ragionevole fissare un valore di base pari a 0.4, tuttavia, siccome si considera un campionamento dalla traiettoria approssimata, per le medesime ragioni descritte in

2.4.2, ci si aspetta di ottenere prestazioni migliori utilizzando un valore più grande. L'osservazione dei grafici delle catene di tutti i modelli dove è stato applicato il metodo suggerisce di fissare δ_1 intorno a 0.6.

L'obiettivo è quello di trovare un valore di ϵ che assicuri prestazioni analoghe per ciascuna componente discreta e un tasso di accettazione globale vicino a quello nominale. Un possibile problema che può insorgere riguarda il fatto che potrebbero esistere delle configurazioni per cui il tasso di rifrazione empirico sia tale da decrescere in maniera differente per le diverse componenti in θ_J o che non sia in accordo con il tasso nominale globale δ_1. Per questa ragione, come suggerito in Neal (2011), è possibile sfruttare le proprietà del flusso hamiltoniano per assegnare dei valori di ϵ differenti a ciascuna componente attraverso la specificazione di una matrice di massa diagonale, in questo caso solo per le componenti θ_J. Seguendo questo suggerimento, una procedura alternativa proposta consiste nell'adattamento in parallelo dei diversi ϵ applicando il metodo *Dual Averaging* alle seguenti funzioni stocastiche:

$$H_i^0 = \frac{1}{|t_i|} \sum_{z \in t_i} \min\left\{1, e^{H(z_0)-H(z)}\right\} - \delta_1$$
$$H_i^j = \frac{1}{|t_i|} \sum_{z \in t_i} \mathcal{I}\left(M_{jj}^{-1/2}|m_j| > \Delta_j U(z)\right) - \delta_2 \quad j = 1, \ldots, k$$

Ad ogni iterazione della procedura adattiva, vengono proposti dei valori $\epsilon_0, \epsilon_1, \ldots \epsilon_k$ e, per ricondursi al caso con uno ϵ unico, è sufficiente applicare le seguenti trasformazioni:

$$M_{j,j}^{-1/2} = \frac{\epsilon_j}{\epsilon} \quad j = 1, \ldots, k \tag{3.10}$$

dove $\epsilon = \epsilon_0$. Questo non è la stessa cosa rispetto ad applicare effettivamente dei valori di ϵ diversi, infatti l'aggiornamento dei momenti, che non dipende da M, viene fatto utilizzando il valore di ϵ globale. Se così non fosse il metodo continuerebbe a produrre una *proposal* valida, ma si perderebbero le proprietà teoriche che garantiscono l'allineamento della traiettoria al *Typical Set* e l'efficienza della procedura ne risentirebbe. Nel caso in cui il numero di componenti che implicano una discontinuità, k, corrisponda alla dimensionalità dello spazio parametrico, d, il tasso di accettazione globale è sempre pari ad 1, perché l'energia è conservata esattamente,

precludendo la possibilità di trovare un valore ϵ_0 con il metodo appena descritto. In questo caso è necessario scegliere un valore di riferimento: un'opzione, che empiricamente si è osservato funzionare meglio su una serie di modelli, consiste nel prendere il più piccolo ϵ_j. Una seconda alternativa all'utilizzo di (3.9), applicabile solo quando $k < d$, consiste nell'ignorare i tassi di rifrazione durante la procedura adattiva, limitandosi all'applicazione di (2.22), e lasciare che sia la successiva stima di M_J a stabilire gli ϵ_j ottimi.

Data la crucialità della fase di *warm-up*, per migliorarne l'efficienza, viene considerato l'utilizzo di un *riciclaggio* dei campioni simulati per ciascuna traiettoria secondo l'approccio descritto in Nishimura e Dunson (2020). Di fatto, se si applica un'opportuna correzione attraverso una probabilità di accettazione (1.4), ogni elemento della traiettoria è rappresentativo della distribuzione microcanonica e quindi utilizzabile per costruire uno stimatore Monte Carlo non distorto. Ovviamente, due valori attaccati risulteranno pedissequi riguardo l'informazione sulla distribuzione obiettivo, quindi un tale approccio è sensato e conveniente solamente se pochi campioni vengono riciclati. Gli autori mostrano come già solo con un campione riciclato in più si ottenga un miglioramento del 40% nella quantità di informazione estraibile per unità di tempo. Inoltre, l'implementazione all'interno della struttura ad albero binario bilanciato viene ancora più naturale, dal momento che si applicava già un campionamento multinomiale dalla traiettoria, comportando solo un leggero aumento del costo computazionale dovuto ad una maggiore allocazione di memoria. All'interno della libreria XDNUTS, tale metodo è utilizzato di base con un numero $K = 3$ di campioni riciclati solo durante la fase di *warm-up*, in modo tale da ottenere più velocemente una buona stima della matrice di massa, riducendo il tempo necessario per poter poi velocizzare la fase di campionamento attraverso un'esplorazione più agiata. Durante la fase di campionamento, caratterizzata solitamente da una numerosità dei campioni Monte Carlo maggiore, di base si preferisce non applicare il riciclaggio, dal momento che il vantaggio che ne comporta si fa man mano più flebile conservando però costante il leggero aumento nel costo computazionale di ciascuna iterazione. Sia il valore di K che quest'ultima opzione sono modificabili a

piacere.

Per una più dettagliata spiegazione del funzionamento della libreria R, è possibile consultare la rispettiva documentazione (Manildo, 2024).

3.4.1 Studio di simulazione

Per testare l'efficacia dei criteri di terminazione proposti, nel contesto di distribuzioni a posteriori in parte discontinue, è stato condotto uno studio di simulazione. Seppure in questa tesi sia adottato un approccio prevalentemente bayesiano, in questa sezione si è preferito andare oltre le differenze ideologiche che lo contraddistinguono da quello frequentista e servirsi anche di quest'ultimo per rendere le analisi più semplici. Concretamente, quello che si è fatto è analizzare alcune statistiche della distribuzione a posteriori di un modello bayesiano, stimate via MCMC, sotto l'ottica del principio del campionamento ripetuto. Sia $X \sim N_d(\mu, \Sigma)$, dove si assume di conoscere Σ, alla quale è assegnata la seguente struttura di equi-correlazione

$$\Sigma_{ii} = 1 \quad , \quad \Sigma_{ij} = \rho, \quad i = 1, \dots, d \, , \, j \neq i$$

L'oggetto di inferenza è μ, d-dimensionale, che viene assunto essere un vettore aleatorio in parte continuo ed in parte discreto. Alle componenti continue, μ_I, è stata assegnata una distribuzione a priori impropria

$$\pi(\mu_I) \propto 1,$$

per cui è facile dimostrare che la distribuzione a posteriori risultante invece è propria, ma non è di reale interesse per gli scopi di questo studio. La discontinuità di alcune sue componenti è stata forzata tramite l'utilizzo di una distribuzione a priori discreta, uniforme nel supporto

$$\{-30.00, -29.99, -29.98, \dots, 0, \dots, 29.98, 29.99, 30.00\},$$

che viene resa continua attraverso la trasformazione descritta in (3.4) e mappata in $\mathbb{R}$ grazie alla trasformazione logistica inversa.

Definendo con k il numero di componenti discontinue, sono considerati diversi scenari al crescere di d e di ρ:

- $d = \{10, 30, 50\}$;
- $\frac{k}{d} = \{0, 0.4, 0.8, 1\}$;
- $\rho = \{0.2, 0.4, 0.8\}$;

per un totale di 36 configurazioni. Per ciascuna di queste, si misura l'efficacia dei criteri di terminazione con il numero di campioni effettivamente indipendenti estratti a parità di integrazioni della traiettoria. Data una catena di N campioni, l'idea è che: se l'applicazione di (3.8) o (2.19) risulta in un numero di integrazioni della traiettoria pari ad L_{tot}, allora l'utilizzo del DHMC classico, con tempo di integrazione fissato pari a $\left\lceil \frac{L_{tot}}{N} \right\rceil \cdot \epsilon$, dovrebbe risultare in un'esplorazione meno efficiente a parità di costo computazionale, dove quest'ultimo è rappresentato proprio dal numero di integrazioni. Inoltre, per avere dei risultati comparabili, è necessario che i diversi algoritmi condividano gli stessi ϵ e M. Per ognuno dei 36 problemi è stata adottata la seguente procedura:

1. stimare il valore ottimo di ϵ attraverso l'applicazione della procedura adattiva, descritta a pagina 89, utilizzando un DHMC classico[4] con numero di integrazioni fissato a $L = 100 \cdot \rho$;
2. applicare il DHMC dotato di criterio di terminazione, con valore di ϵ trovato al passo precedente, M_I pari a Σ_I^{-1} e M_J posto uguale alla matrice identità;
3. calcolare il numero di integrazioni effettuate, $L_{tot} = \sum_{i=1}^{N} L_i$;
4. applicare il DHMC classico con numero di integrazioni fissato a $L = \left\lceil \frac{L_{tot}}{N} \right\rceil$, l'arrotondamento per eccesso è una scelta conservativa, che assegna un leggero vantaggio al DHMC classico;
5. confrontare le due procedure in termini di numero di campioni effettivamente indipendenti estratti.

[4] L'utilizzo di un metodo rispetto ad un altro non dovrebbe essere rilevante ma, per evitare favoritismi rispetto alle procedure con criterio di terminazione, si è preferito utilizzare il DHMC classico

Il secondo passaggio, nel caso del metodo basato sul viriale, richiede la specificazione di una soglia τ. Tale scelta viene guidata da una procedura a due passi, atta a massimizzare il numero di campioni effettivamente indipendenti estratti per numero totale di integrazioni: nel primo passo si trova il massimo di questa funzione in una griglia sufficientemente ampia, ingrandendola se questo giace sui suoi estremi, nel secondo passo si ricerca all'interno dell'intervallo definito dai valori limitrofi al primo punto di ottimo trovato nella griglia.

Con una numerosità Monte Carlo pari a $N = 1000$, ottenuta da un'unica catena fatta partire dai veri valori usati nella simulazione per i parametri μ, questi 5 punti sono stati iterati 30 volte per ciascuna configurazione, ottenendo in questo modo un campione delle quantità di interesse con cui poter fare inferenza. In ogni iterazione, il numero di campioni effettivamente indipendenti per l'i-esimo parametro è stato misurato dalla seguente quantità:

$$ESS_i = \frac{N}{1 + 2\sum_{k=1}^{K} \rho_i(k)} \tag{3.11}$$

dove la sigla ESS sta per *Effective Sample Size*, mentre $\rho_i(k)$ è la funzione di autocorrelazione empirica della catena ottenuta per μ_i e K un valore in $\mathbb{N}$, minore di N ma sufficientemente grande. Tramite un'analisi grafica, si è osservato come il rapporto tra ESS_i^C [5] e ESS_i restasse circa costante al variare di $i = 1, \dots, d$, pertanto si è deciso di aggregare questi indici in uno solo facendone la media aritmetica. Così facendo, per ognuno dei 36 modelli, si dispone di una coppia di campioni $\left(\overline{ESS}_1, \overline{ESS}_1^C\right), \dots, \left(\overline{ESS}_{30}, \overline{ESS}_{30}^C\right)$ per cui si vuole verificare l'ipotesi

$$H_0 : \mathbb{E}[\overline{ESS}^C] \leq \mathbb{E}[\overline{ESS}]$$

contro l'alternativa

$$H_1 : \mathbb{E}[\overline{ESS}^C] > \mathbb{E}[\overline{ESS}]$$

Attraverso un'analisi esplorativa di questi campioni, si è notato come l'assunzione di normalità, nonostante l'aggregazione di variabili aleatorie dovrebbe incentivarla, non è quasi mai rispettata: applicando il test di Shapiro e Wilk (1965) e fissando

[5] L'apice C si riferisce ai metodi che sfruttano un qualche criterio di terminazione.

una soglia che tenga conto della molteplicità attraverso la correzione di Bonferroni (Dunn, 1961), solo il 30% delle volte non si rifiuta l'ipotesi di normalità. Inoltre, si osserva un evidente presenza di correlazione tra le coppie di unità statistiche dei due campioni, in particolar modo per i dati relativi al criterio di terminazione basato sul viriale, confermando la sensibilità delle procedure alle fluttuazioni aleatorie che portano ad una diversa selezione di ϵ. Si è scelto, quindi, di considerare questi campioni come dei dati appaiati e di utilizzare un test non parametrico che, invece di concentrarsi sulle medie delle due distribuzioni, si concentra sulle loro mediane. Il sistema di ipotesi risultante è il seguente:

$$\begin{cases} H_0 : Me\left(\overline{ESS}^C\right) \leq Me\left(\overline{ESS}\right) \\ H_1 : Me\left(\overline{ESS}^C\right) > Me\left(\overline{ESS}\right) \end{cases}, \tag{3.12}$$

dove con $Me(x)$ si indica la mediana di x.

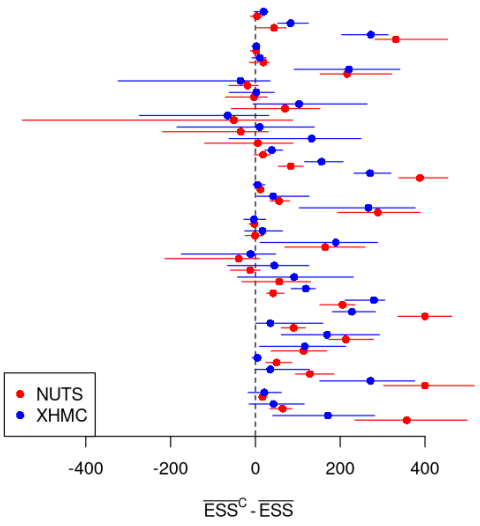

Figura 3.6: $\overline{ESS}^C - \overline{ESS}$ osservati per ciascuna delle 36 configurazioni.

In Figura 3.6 sono riassunte le differenze tra gli *Effective Sample Size* medi per ciascuna delle 36 configurazioni considerate, in rosso il confronto è fatto col il metodo che usa il criterio di terminazione basato sul prodotto interno (3.8), in blu con quello che utilizza il viriale (2.19). Ogni punto rappresenta la mediana delle differenze per quella configurazione, mentre la linea orizzontale che lo sovrasta corrisponde all'intervallo inter-quantilico $[q_{0.1}, q_{0.9}]$ empirico. Come si può nota-

re, spesso la mediana risulta spostata verso destra, segnalando un'asimmetria che spiega il venir meno dell'assunzione di normalità, inoltre si nota che le differenze sono prevalentemente positive. Date le circostanze, per verificare questa ipotesi si è scelto di adottare il test dei segni per ranghi (Wilcoxon, 1945) che, nel caso di un'ipotesi alternativa unilaterale destra, utilizza come statistica test la somma dei ranghi delle differenze positive osservate, in questo caso:

$$W = \sum_{j=1}^{30} \text{rank}\left(|\overline{ESS}_j^C - \overline{ESS}_j|\right) \cdot \mathcal{I}\left(\overline{ESS}_j^C > \overline{ESS}_j\right). \tag{3.13}$$

L'idea di fondo è che, se fosse vera l'ipotesi nulla, i ranghi delle differenze positive sarebbero piccoli o distribuiti casualmente intorno al valore medio $\mu_W = \frac{n(n+1)}{4}$, per cui è naturale rifiutare H_0 per valori grandi di W. Per ciascuno modello con-

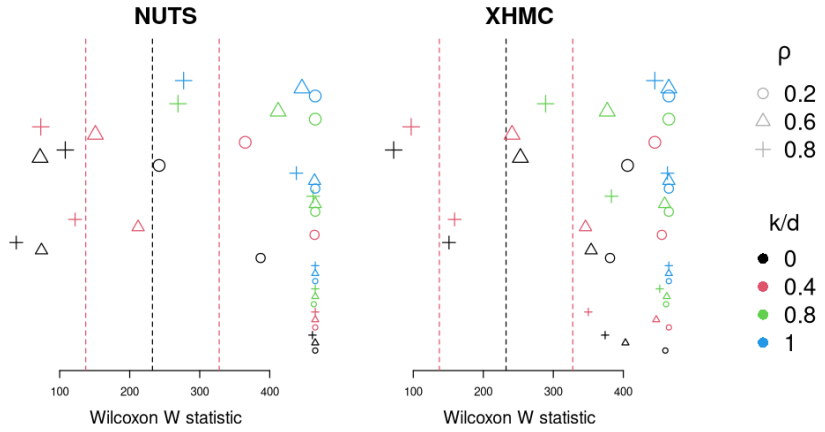

Figura 3.7: Statistiche test (3.13) per i diversi modelli dello studio di simulazione.

siderato nello studio di simulazione, in Figura 3.7 è riportato, sotto forma di un simbolo colorato per cui la dimensione cresce con d, il valore della statistica test definita in (3.13). La linea verticale in μ_W è accompagnata da due bande tratteggiate rosse in corrispondenza dei quantili $0.05, 0.95$ approssimati della distribuzione sotto l'ipotesi di uguali mediane. Tali bande hanno lo scopo di far capire dove la statistica dovrebbe trovarsi in assenza di un effetto sia positivo che negativo, anche se l'interesse in questa sede è solo rispetto a scostamenti positivi. Dal grafico si nota chiaramente che l'utilizzo dei criteri di terminazione in questione comporti un significativo aumento in termini di *Effective Sample Size* per una buona fetta dei

modelli. A supporto di quest'affermazione, in Figura 3.8, sono riportati i valori p dei rispettivi test statistici. Siccome in corrispondenza delle statistiche che in Figura 3.7 sono minori di μ_W questi assumono valori vicino ad 1, e l'interesse riguarda la soglia minore, si è preferito adottare, per questioni di chiarezza, una scala logaritmica cambiata di segno, per cui si rifiuta l'ipotesi nulla per valori grandi di questa nuova statistica test. Nel grafico è riportato anche il valore critico adottato, calcolato come il negativo del logaritmo del livello di significatività, $\alpha = 0.05$, a cui è stata applicata la già citata correzione di Bonferroni per i test multipli. Per i due

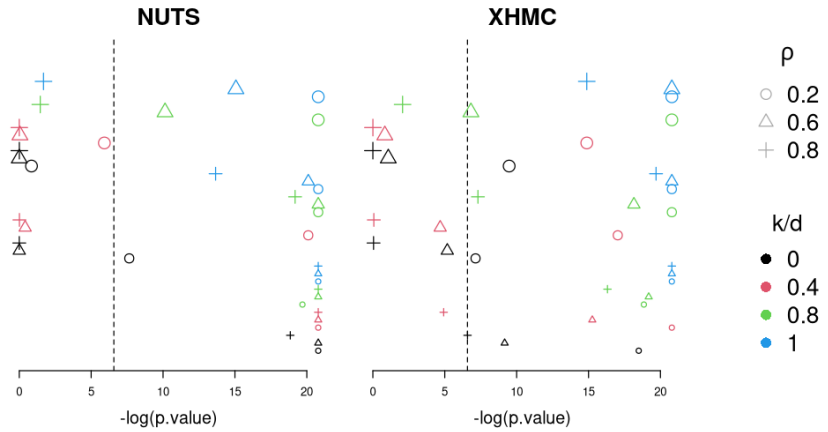

Figura 3.8: Logaritmo dei valori p per l'ipotesi alternativa (3.12) cambiati di segno, nei diversi modelli dello studio di simulazione.

criteri proposti, quello basato sul prodotto interno e quello basato sull'esaurimento del viriale, sono rifiutate rispettivamente 24 e 26 ipotesi nulle (3.12), su un totale di 36.

In entrambi i casi, le configurazioni per cui c'è un effetto nullo o leggermente negativo sembrano essere quelle caratterizzate da un'alta dimensionalità/correlazione nei dati, accompagnata da un basso rapporto di componenti discontinue. Attraverso un'analisi esplorativa delle differenze medie si osserva come non sembrino esserci interazioni rilevanti da giustificare una complicazione interpretativa rispetto ad un modello che considera solo gli effetti principali di d, k/d e ρ per spiegare il fenomeno. Per misurare questi effetti, è possibile ricorrere ad una regressione sulla mediana,

ottenuta come la soluzione della seguente funzione di perdita:

$$Loss(\beta) = \sum_{j=1}^{36} \sum_{i=1}^{30} \epsilon_{ij}(\beta) \cdot (0.5 - \mathcal{I}(\epsilon_{ij}(\beta) < 0))$$

dove $\epsilon_{ij}(\beta) = \left(\overline{ESS}_{ij}^{C} - \overline{ESS}_{ij}\right) - \beta_0 - \beta_1 d_j - \beta_2 \frac{k_j}{d_j} - \beta_3 \rho_j$, con $i = 1, \ldots, 30$ e $j = 1, \ldots, 36$. In Tabella 3.1 sono riportati i coefficienti stimati e i relativi *Standard Errors* calcolati tramite *Bootstrap* (Efron, 1979), dove si è ricampionato all'interno di ciascuna configurazione indicizzata da j. Per entrambi i metodi sono riportati, a sinistra, i coefficienti stimati dal modello additivo che considera tutte e 3 le esplicative, mentre a destra i rispettivi valori marginali, ottenuti facendo una regressione con una sola esplicativa alla volta. La concordanza di segno di ogni riga suggerisce l'assenza di fenomeni quali il paradosso di Simpson (1951) all'interno dei dati osservati. È possibile inoltre calcolare un corrispettivo dell'indice R^2 come:

	NUTS		XHMC	
β_0	333.98 (60.73)	50.07 (20.50)	254.19 (30.19)	54.28 (21.67)
β_1	-3.28 (0.71)	-2.52 (1.14)	-1.56 (0.57)	-1.84 (1.19)
β_2	111.91 (31.18)	81.61 (68.30)	103.64 (23.78)	93.91 (51.93)
β_3	-374.28 (78.55)	-347.19 (105.20)	-327.14 (40.52)	-333.86 (51.20)

Tabella 3.1: Coefficienti della regressione quantilica sulle differenze tra $\overline{ESS}$. Tra parentesi sono riportati gli *Standard Errors*

$$R^1 = 1 - \frac{Loss(\hat{\beta}^1)}{Loss(\hat{\beta}^0)}$$

dove $\hat{\beta}^1$ è la stima dei coefficienti del modello completo, mentre $\hat{\beta}^0$ è l'equivalente per il modello con sola intercetta. Tale indice è pari a 0.41 e 0.36 rispettivamente per i modelli NUTS e XHMC. Questi valori non molto alti sono da imputare principalmente all'asimmetria dei dati e alla presenza di alcuni valori anomali, infatti anche considerando un modello con interazioni tra le esplicative, non si nota un aumento rilevante.

Le stime ottenute riassumono bene quanto osservabile con un po' di attenzione dai grafici precedenti, tuttavia non spiegano affatto le motivazioni dietro tale fenomeno, inoltre, per situazioni con una dimensionalità così alta, non è possibile aiutarsi graficamente osservando il percorso svolto da ciascuna traiettoria, come si faceva nelle Figure 2.3, 3.4 e 3.5. Una possibile interpretazione del fenomeno sottostante è data dal fatto che per le componenti discontinue non si è in grado di tenere conto della loro correlazione a posteriori, perché si utilizza una matrice di massa diagonale, ciò comporta una più difficile esplorazione dello spazio di probabilità che rende il criterio di terminazione più utile per muoversi in maniera efficiente. Per valori di $\frac{k}{d}$ bassi invece, data la relativa semplicità del modello, gaussiano e opportunamente ruotato da M_I, le differenze tendono ad essere meno rilevanti, a maggior ragione al crescere di d e ρ, per cui la complessità tende a ridurre contemporaneamente sia $\overline{ESS}^C$ che $\overline{ESS}$ e con essi la loro differenza assoluta.

Il codice R per replicare lo studio di simulazione è riportato in appendice C.2.

Capitolo 4

Esempi e applicazioni

Per quanto accennato nel paragrafo 3.3, la trasformazione dei parametri discreti in continui non è molto giustificata se questi non presentano un ordinamento, pertanto modelli a fattori latenti discreti, come ad esempio gli Hidden Markov Models non sono stati considerati. Una menzione particolare riguarda il contesto della selezione di variabili, dove il DHMC potrebbe essere applicato, visto che la natura dicotomica delle variabili indicatrici ha un ordinamento ben definito: da 0 a 1. Tuttavia il metodo non apporta nessun contributo effettivo a quelli già esistenti, come ad esempio lo Shotgun Stochastic Search di Hans et al. (2007), dal momento che, per come è costruito l'Algoritmo 3, il movimento nello spazio di probabilità dei modelli richiederebbe comunque di calcolare la verosimiglianza marginale del nuovo modello per stabilire se spostarsi in quella configurazione o meno. Inoltre, la mancanza di un ordinamento all'interno dello spazio dei modelli, rende il ruolo dei momenti m del tutto irrilevante rispetto ad un campionamento casuale in un intorno del modello corrente. Alla luce di ciò, nei paragrafi successivi sono state considerate delle applicazioni in contesti più consoni.

4.1 Imputazione di dati mancanti discreti

Un possibile contesto applicativo riguarda l'imputazione di valori mancanti per variabili discrete. Un semplice esempio è fornito in McElreath (2018, capitolo 15,

sezione 3) in cui l'autore originariamente ricorreva alla marginalizzazione per poter stimare il modello via STAN, in questa sede invece, attraverso la libreria XDNUTS, si stimerà il modello completo. I dati riguardano un quartiere dove ogni casa possiede un canarino. Si prende un campione casuale tra queste e si registra il numero di note cantate dal pennuto in un minuto. Inoltre, in alcune abitazioni, è nota la presenza di un gatto, mentre per altre tale informazione è mancante. I parametri su cui si vuole fare inferenza riguardano il tasso di note cantate in un minuto, l'effetto medio che la presenza di gatti in casa ha su questa quantità, la probabilità che nell'abitazione ci siano gatti e, infine, se in ciascuna delle case in cui l'informazione è assente vi sia un gatto o meno. Siano $N_1, \ldots, N_n$ il numero di note in un minuto osservate per le diverse case e $C_1, \ldots, C_n$ le variabili indicatrici riguardanti la presenza di gatti, di cui una porzione C_J^*, di cardinalità k, è incognita. Il modello proposto dall'autore per questi dati, assumendo l'indipendenza degli N e dei C, è il seguente:

$$
\begin{aligned}
N_i &\sim Poisson(\lambda_i) \\
\log \lambda_i &= \alpha + \beta C_i \\
C_i &\sim Bern(p) \\
\alpha &\sim N(0, s_\alpha^2) \\
\beta &\sim N(0, s_\beta^2) \\
p &\sim Beta(a, b)
\end{aligned}
$$

Per facilitare l'esplorazione dello spazio parametrico si applica successivamente una trasformazione logistica inversa a p per portarlo in $\mathbb{R}$, aggiungendo il determinante dello Jacobiano dell'inversa di questa alla sua distribuzione a priori. Successivamente, per ciascuna variabile dicotomica mancante, si applica la trasformazione descritta in (3.4), prima mappando le masse di probabilità di 0 e 1 rispettivamente negli intervalli aperti $(0, 1]$ e $(1, 2]$, e poi applicando la trasformata logistica inversa per riportarsi in $\mathbb{R}^k$. Tutti i conti sono descritti in appendice B.1.1. Questo *dataset* è stato proposto in McElreath (2018, capitolo 15, sezione 3) come esempio di modello generativo, pertanto i dati stessi sono simulati, permettendo così di

testare la capacità del modello di imputare i dati mancanti conoscendo la verità. Seguendo tale approccio sono stati simulati 100 valori di N da una Poisson di tasso $\lambda_i = e^{\alpha+\beta C_i}$ con $\alpha = 2$ e $\beta = -1$. Ciascun C è stato simulato da una Bernoulliana di parametro $p = 0.5$ e con probabilità 0.2 è stato messo mancante. Per la scelta degli iper-parametri sono state adottate delle a priori meno informative di quelle proposte originariamente, in particolare $s^2_\alpha = s^2_\beta = 10$ e $a = b = 1$. Successivamente, attraverso la libreria XDNUTS sono state eseguite 10 catene in parallelo di 500 e 1000 iterazioni ciascuna, rispettivamente per la fase di *warm-up* e per il campionamento vero e proprio, per un tempo di esecuzione totale di 15 secondi circa su un computer portatile Lenovo con sistema operativo Linux. Il criterio di terminazione applicato per questo esempio è quello basato sul viriale, dove, dopo un po' di tentativi, si è scelto un valore ragionevole per τ pari a 1.5. A seguire, sono riportati alcuni dei grafici più importanti, mentre il codice R per riprodurre le analisi è disponibile in appendice C.3.

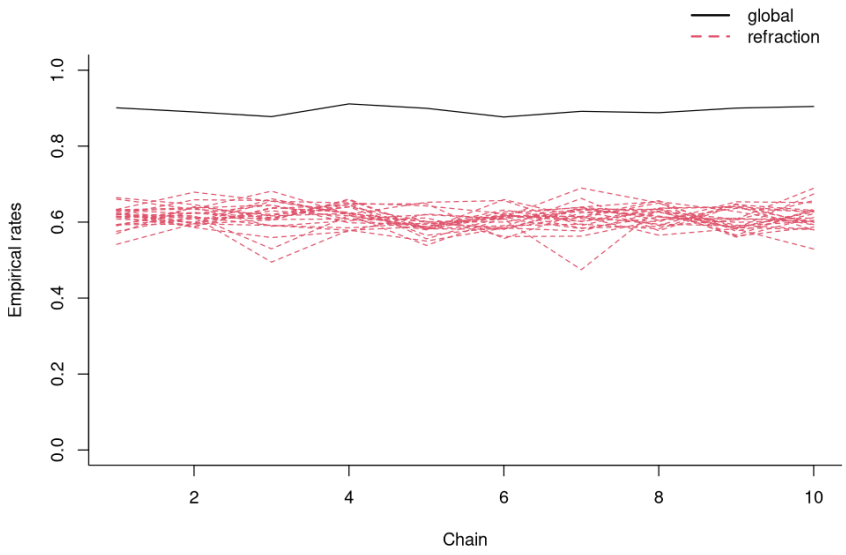

Figura 4.1: Tassi di accettazione Metropolis globale e di rifrazione per le componenti discontinue per il modello Poisson-Bernoulli con imputazione di dati mancanti.

In Figura 4.1 sono riportate le spezzate, rispetto alle 10 catene, dei tassi di rifrazione per le componenti continue, in rosso tratteggiato, e il tasso di accettazione

Metropolis globale, in nero. L'algoritmo di ottimizzazione stocastica implementato riesce con successo a portare il tasso di rifrazione di tutti i parametri discreti vicino al valore nominale desiderato, 0.6, e a garantire il tasso di accettazione globale per la procedura HMC vicino al valore nominale, 0.8. Inoltre, tali valori sembrano variare poco tra le diverse catene, nonostante siano stati scelti punti di partenza in una griglia diffusa.

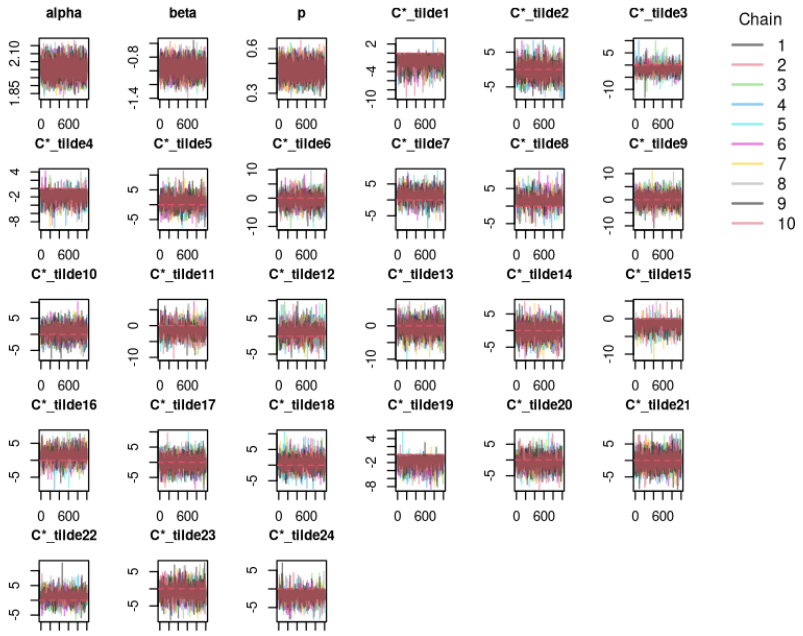

Figura 4.2: Catene di Markov di ogni parametro per il modello Poisson-Bernoulli con imputazione di dati mancanti.

In Figura 4.3 è riportata la catena di Markov dei livelli di energia del modello, questo grafico serve a riassumere la diagnostica relativa alla convergenza delle diverse catene in una sola, anche se talvolta può nasconderne un comportamento anomalo di qualche singola catena. In questo caso tutte le catene sembrano comunque convergere e mostrano un buon *mixing*, come mostrato in Figura 4.2.

In Figura 4.4 si sovrappongono le frequenze assolute del numero di integrazioni fatte da ciascuna traiettoria delle varie catene. Oltre che ad un andamento comune, si può notare che, attraverso l'uso del criterio di terminazione (2.19), la maggior parte delle traiettorie si interrompe prima di raggiungere 10 integrazioni. Questo risultato, se accompagnato da una bassa auto-correlazione delle catene, giustifica

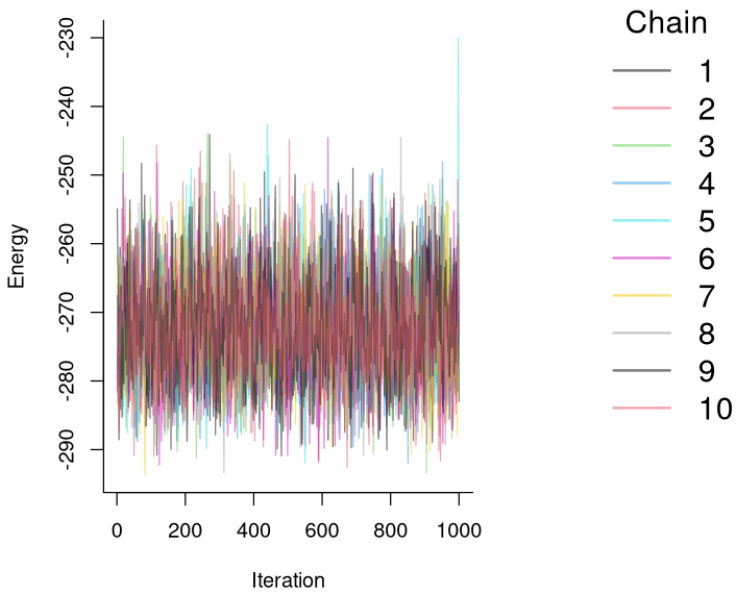

Figura 4.3: Catene di Markov per i livelli di energia per il modello Poisson-Bernoulli con imputazione di dati mancanti.

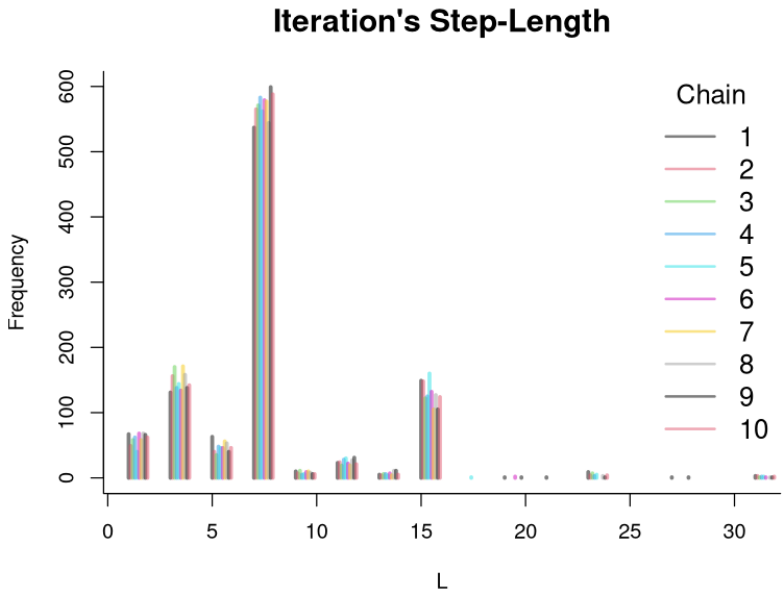

Figura 4.4: Frequenze assolute del numero di integrazioni per traiettoria per il modello Poisson-Bernoulli con imputazione di dati mancanti.

empiricamente la validità di tale criterio di terminazione anche per le densità a posteriori discontinue.

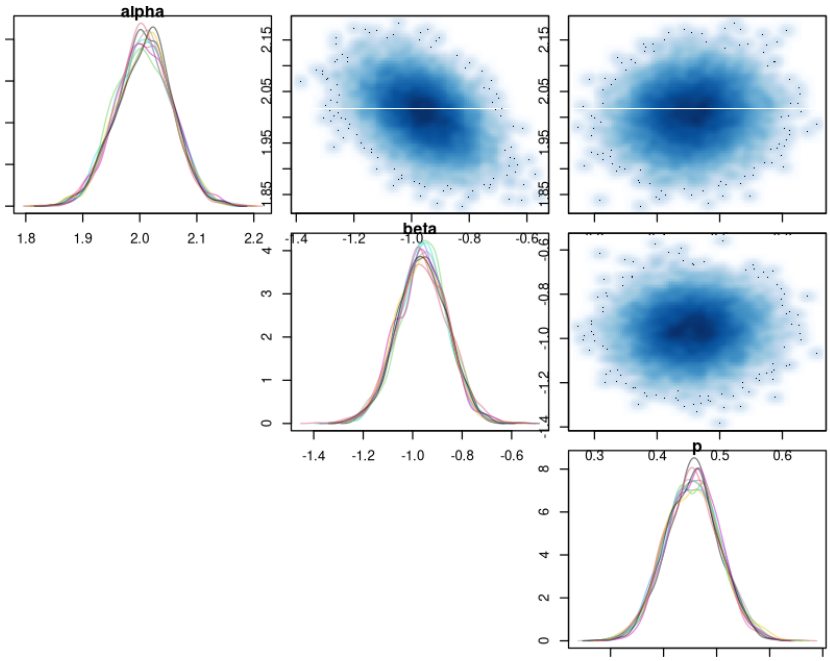

Figura 4.5: Densità univariate e bivariate delle componenti continue per il modello Poisson-Bernoulli con imputazione di dati mancanti.

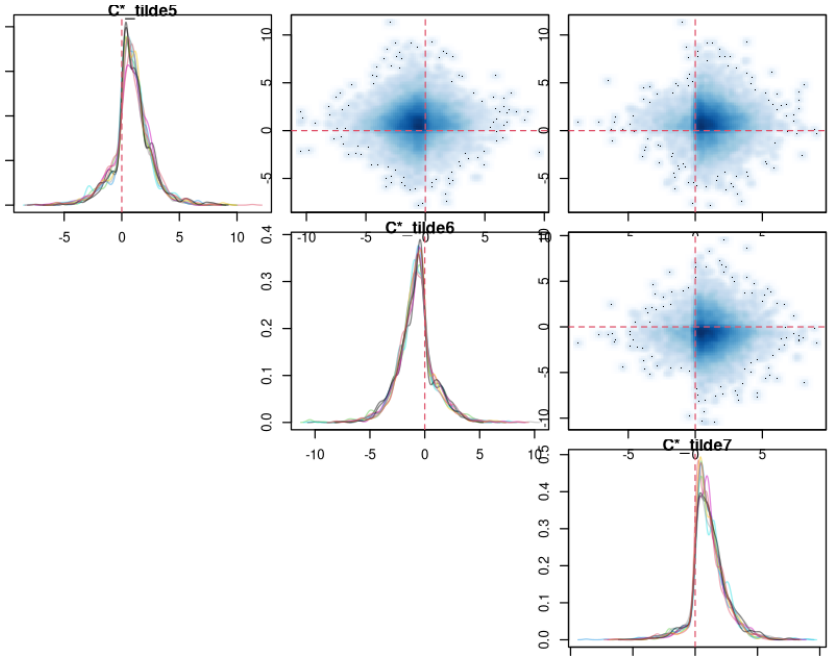

Figura 4.6: Densità univariate e bivariate di tre componenti discontinue per il modello Poisson-Bernoulli con imputazione di dati mancanti.

In Figura 4.5 sono riportate le densità a posteriori marginali e bivariate per i parametri continui. In particolare si può notare come queste siano centrate nei veri

valori dei parametri fissati dalla simulazione: $\alpha = 2$, $\beta = -1$ e $p = 0.5$. In Figura 4.6 invece è riportato il medesimo grafico ma per tre componenti discontinue, tali densità non hanno un reale significato visto che vivono in uno spazio di probabilità fittizio, surrogato di quello discreto originale, tuttavia è interessante notare il loro comportamento in prossimità della frontiera di discontinuità presente in 0.

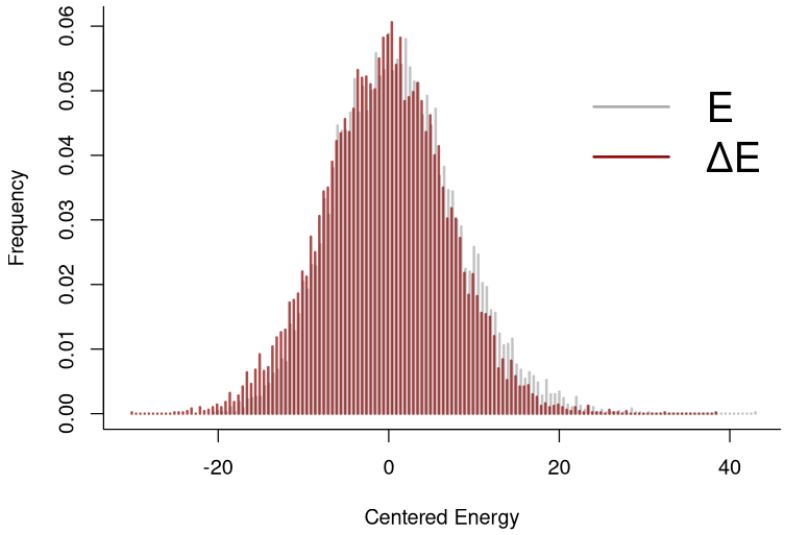

Figura 4.7: Densità marginale e differenziata prima dell'energia per il modello Poisson-Bernoulli con imputazione di dati mancanti.

In Figura 4.7 è riportato il grafico descritto nel paragrafo 2.3.2 a pagina 43, la buona copertura della densità delle differenze prime sulla densità marginale dei livelli di energia non mostra evidenza di una scelta sub-ottimale per la funzione di disintegrazione, in parte gaussiana e in parte laplaciana.

In Figura 4.8 sono mostrate le funzioni di auto-correlazione empiriche per ciascuna catena, le componenti continue sono caratterizzate da una dipendenza quasi nulla, addirittura leggermente negativa, questo non è da considerarsi per forza una buona cosa ma è testimone del fatto che lo spazio di probabilità viene esplorato agilmente. Le componenti discontinue tendono ad essere leggermente più auto-correlate, con un valore per il primo ritardo sotto 0.4 per tutte e che decresce rapidamente. Il *summary* del modello comunica che il test per il *Potential Scale*

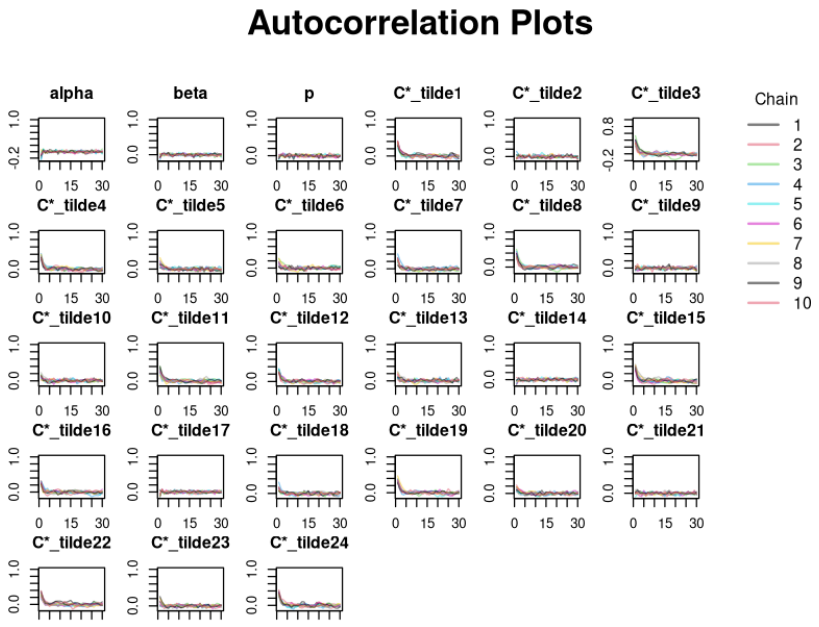

Figura 4.8: Funzioni di auto-correlazione empiriche di ciascuna catena per il modello Poisson-Bernoulli con imputazione di dati mancanti.

Reduction Factor (Gelman e Rubin, 1992), sia per ogni catena che multivariato, non mostra evidenza contro la convergenza delle catene. Il numero di campioni effettivamente indipendenti per le componenti continue è superiore a 10000, ma questo per via dell'anti-correlazione, per quanto riguarda quelle discrete, questo si aggira mediamente intorno ai 7000. Successivamente, attraverso la trasformata inversa descritta in appendice B.1.1, è possibile ottenere la distribuzione a posteriori dei parametri discreti e prevedere i dati mancanti utilizzando come classificatore la seguente funzione:

$$\hat{C}_j = \mathcal{I}\left(\frac{1}{N_{mcmc}}\sum_{i=1}^{N_{mcmc}} \mathcal{I}(C_j^i = 1) > \hat{p}\right) \tag{4.1}$$

dove $C_j^i, \ldots, C_j^{N_{mcmc}}$ sono i campioni a posteriori di C_j e $\hat{p}$ è la media a posteriori della probabilità di osservare un gatto. Per questi dati simulati solo 2 dati mancanti su 24 sono scorrettamente imputati, tuttavia questo dipende dall'effetto della covariata, β, che si è scelto: aumentarlo faciliterebbe l'imputazione, se non c'è effetto invece il metodo non potrà estrarre informazione dove non ce n'è e tenderà a classificare a caso.

4.1.1 Confronto con altri algoritmi

Per valutare la validità della procedura utilizzata, DHMC con criterio di terminazione basato sul viriale (XDHMC), la si è confrontata con altri possibili algoritmi, in termini di numero di campioni effettivamente indipendenti (3.11), in grado di estrarre per unità di tempo. In particolare, sono stati considerati i seguenti metodi:

- Exhaustive Hamiltonian Monte Carlo (XHMC), sul modello che marginalizza i parametri discreti, i passaggi per la derivazione della relativa energia potenziale e del suo gradiente sono presenti in appendice B.1.2;
- Markov Chain Indipendent Sampler (MCID), sul modello *completo*, utilizzando come *proposal* per i parametri continui una t multivariata con parametri di scala e posizione stimati dai campioni a posteriori ottenuti con il XDHMC, mentre per quelli discreti una proposta casuale tra 0 e 1;
- Markov Chain Indipendent Sampler (MCID_M), sul modello che marginalizza i parametri discreti, con la stessa *proposal* del precedente ma ristretta solo ai parametri continui;
- Gibbs Sampler, che sostituisce le a priori non informative adoperate sui parametri α e β, inizialmente proposte in McElreath (2018, capitolo 15, sezione 3), con delle versioni coniugate che permettano di scrivere la *full conditional* di ogni parametro del modello in forma esplicita, i passaggi per la derivazione sono riportati in appendice B.1.3.

Siccome le varie procedure hanno diversi modi per calibrare i loro parametri, si è preferito confrontare i metodi solamente per la fase di campionamento, facendo partire tutte le catene dalla media a posteriori dei parametri trovata inizialmente dall'algoritmo XDHMC. Per ciascun algoritmo sono state eseguite 10 catene in parallelo, ciascuna di numerosità 1000 ed è stato registrato il tempo di esecuzione. Il codice per riprodurre i confronti è reso disponibile in appendice C.3.1.

In Tabella 4.1 sono riportate le quantità di interesse, per ogni parametro continuo, ottenute dai 5 metodi a confronto. Tutti producono delle stime Monte Carlo

paragonabili per la media e deviazione standard a posteriori di ciascun parametro, l'unica anomalia è riscontrata da parte di XHMC, riconducibile all'alta autocorrelazione nelle sue catene. Quest'ultimo fatto sembrerebbe contraddire quanto affermato nella sezione 3.1 riguardo il Collapsed Gibbs Sampler (Liu, 1994), ma in realtà ha più a che fare con problemi di stabilità numerica che si ottengono nel calcolo del gradiente dell'energia potenziale relativa al modello marginalizzato, che costringe l'utilizzo di un ϵ molto piccolo per evitare di incorrere in transizioni divergenti dell'integratore simplettico. Tale comportamento anomalo, infatti, non si osserva nel Markov Chain Indipendent Sampler, per cui la marginalizzazione dei parametri discreti comporta un incremento notevole in termini di *Effective Sample Size*. Tra tutti gli algoritmi implementati, XDHMC è quello con ESS maggiore per ogni parametro, tuttavia il Gibbs Sampler, grazie al fatto che si è in grado di simulare facilmente dalle *full conditional* per ogni parametro, è quello con le prestazioni migliori se si tiene conto anche del tempo di esecuzione.

Dal momento che i parametri discreti sono 24, e una misura come (3.11) non è molto utile visto che il loro supporto ha cardinalità 2, si è preferito evitare di riproporre la medesima tabella anche in questa sede. In alternativa, in Figura 4.9 sono riportate le probabilità a posteriori per ciascun dato mancante di ogni algoritmo, ordinate in senso decrescete rispetto al valore osservato per la variabile risposta: quando N_j è grande ci si aspetta che C_j sia zero. La linea verticale tratteggiata in rosso delimita i casi in cui il vero valore di C era 0 da quelli in cui invece era 1, le linee orizzontali colorate invece rappresentano la media a posteriori della probabilità p stimata dai vari algoritmi, con cui si costruisce il classificatore (4.1). Tutti gli algoritmi sembrano comportarsi allo stesso modo e a classificare correttamente 22 volte su 24. Come si nota dal grafico, l'imputazione scorretta di due dati mancanti è dovuta ad un valore anomalo di N_j, più piccolo di quanto sarebbe dovuto essere condizionatamente al fatto che $C_j = 1$, che si posiziona sulla frontiera di classificazione.

	XDHMC	XHMC	MCID	MCID_M	Gibbs
$\bar{\alpha}$	2.01	1.94	2.01	2.01	2.01
S_α	0.05	0.06	0.05	0.05	0.05
ESS_α	**12780.67**	35.81	31.98	5715.43	9241.87
ESS_α/s	1664.15	0.14	8.79	2259.06	**5567.39**
$\bar{\beta}$	-0.95	-0.92	-0.96	-0.96	-0.96
S_β	0.10	0.10	0.10	0.10	0.11
ESS_β	**10552.95**	47.78	30.61	5837.71	9153.74
ESS_β/s	1374.08	0.18	8.41	2307.40	**5514.30**
$\bar{p}$	0.46	0.47	0.46	0.46	0.46
S_p	0.05	0.02	0.05	0.05	0.05
ESS_p	**10233.74**	45.15	30.20	5581.31	9070.37
ESS_p/s	1332.52	0.17	8.30	2206.05	**5464.08**

Tabella 4.1: Statistiche descrittive dei diversi algoritmi applicati al modello Poisson-Bernoulli per l'imputazione di dati mancanti. Per ogni parametro continuo sono riportati: media e deviazione standard a posteriori, *Effective Sample Size*, puro e diviso per il tempo trascorso in secondi. Il pedice M indica che l'algoritmo è stato applicato al modello che marginalizza i parametri discreti.

4.2 Modelli per punti di cambio

Il modello descritto nel paragrafo precedente, a meno di problemi numerici, è facile da gestire attraverso la marginalizzazione dei dati mancanti e così facendo il tempo di esecuzione risulta essere minore e le stime potenzialmente migliori per le proprietà del Collapsed Gibbs Sampler dscritto a pagina 64. Un contesto in cui questo approccio non risulta conveniente è dato, ad esempio, dai modelli per punti di cambio. In questo caso il parametro discreto sancisce la differenza tra due verosimiglianze diverse per i dati e, a seconda della numerosità campionaria, la

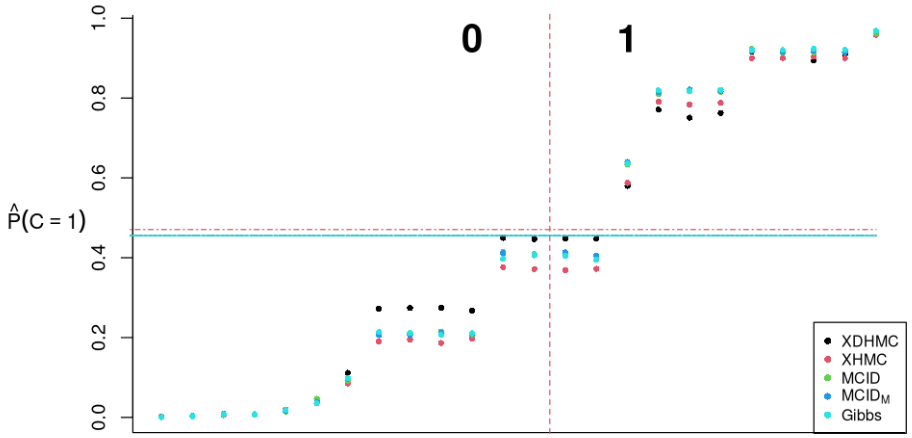

Figura 4.9: Probabilità stimata per ciascun dato mancante da parte dei cinque algoritmi per il modello Poisson-Bernoulli con imputazione di dati mancanti.

cardinalità di Θ_J può diventare molto grande. In questa sede si vuole sfruttare le potenzialità del DHMC di Nishimura et al. (2020) per muoversi velocemente nello spazio discreto di un modello con due punti di cambio. Ispirato dal famoso *dataset* relativo al numero di incidenti minerari nel regno unito descritto in Maguire et al. (1952), si vogliono simulare dei dati analoghi che per 300 anni registrino il numero di eventi rari, immaginando che vi siano stati due cambiamenti di regime. Il primo cambiamento viene posizionato all'anno 100 e porta gli eventi a verificarsi con un tasso più basso, passando da 4 a 1. Il secondo cambiamento è posto in 200, da cui in poi il tasso torna ad essere grande, passando da 1 a 5. Siccome i punti di cambio sono 2, il numero di modalità da marginalizzare sarebbe $O(T^3)$, dove T è il numero di istanti di tempo osservati, in questo caso 300, e ciò motiva l'utilizzo del DHMC.

Il modello probabilistico proposto per questo problema è il seguente:

$$
\begin{aligned}
Y_1, \ldots, Y_{t_1} &\sim Pois(\lambda_1) \\
Y_{t_1+1}, \ldots, Y_{t_2} &\sim Pois(\lambda_2) \\
Y_{t_2+1}, \ldots, Y_T &\sim Pois(\lambda_3) \\
\lambda_1 &\sim Gamma(a_1, b_1) \\
\lambda_2 &\sim Gamma(a_2, b_2) \\
\lambda_3 &\sim Gamma(a_3, b_3) \\
t_1 &\sim Unif(\{t_{inizio}, \ldots, t_{fine} - 1\}) \\
t_2 &\sim Unif(\{t_1 + 1, \ldots, t_{fine}\})
\end{aligned}
$$

Per rendere il modello identificabile si è scelto di utilizzare delle a priori assiomatiche per i parametri t_1 e t_2, che ne limitino il supporto e impongano un ordinamento nei punti di cambio, non permettendo la sovrapposizione dei loro valori. Gli iper-parametri t_{inizio} e t_{fine} servono per specificare una data zona in cui ricercarli, anche se in questo esempio si è optato per la scelta meno informativa possibile: $t_{inizio} = 1$ e $t_{fine} = T - 1$. Anche gli iper-parametri delle a priori coniugate per i diversi tassi sono stati posti in maniera da minimizzarne la sensibilità della legge a posteriori: $a_1 = a_2 = a_3 = b_1 = b_2 = b_3 = 0.001$. Dopo l'applicazione della trasformata (3.4) e il calcolo analitico dell'energia potenziale derivante e del suo gradiente per le componenti continue, entrambi disponibili in appendice B.2.1, è possibile utilizzare la libreria XDNUTS per esplorare la legge a posteriori di questo modello. Tale densità, per costruzione, sarà caratterizzata dalla presenza di molte piccole mode con massa di probabilità del tutto irrilevante in un loro intorno, tuttavia, siccome non si conosce la costante di normalizzazione, se nella fase di *warm-up* per qualche catena si inizia in una di queste configurazioni, c'è il rischio che essa ne rimanga incastrata, adattando i suoi parametri ϵ e M, in maniera da restare all'interno di questo *Typical Set* locale. In Figura 4.10 è mostrato il grafico dei livelli di energia ottenuti con una fase di *warm-up* di sole 500 iterazioni. Tutte le catene raggiungono il *Typical Set* tranne la 4 e la 5, che si incagliano in una moda locale, mentre la

numero 9 inizia la fase di campionamento in un punto lontano ma in poche iterazioni si unisce alle altre nell'esplorazione della zona con più massa di probabilità. Quest'ultimo comportamento può essere spiegato in parte da quanto discusso nel paragrafo 2.3: dal momento che la legge di probabilità di Laplace è a code pesanti, talvolta può generare dei momenti sufficientemente grandi da oltrepassare le barriere energetiche. In questo caso è probabile che, superato così lo scoglio iniziale, la catena 9 abbia trovato delle frontiere di discontinuità molto facili da oltrepassare, ad esempio con differenza potenziale negativa, sfruttando le proprietà dell'Algoritmo 3. Siccome l'energia è proporzionale al negativo del logaritmo della distribuzione a posteriori, catene con un energia più bassa esplorano zone a più alta probabilità e la scala logaritmica permette di interpretare la differenza energetica come il rapporto tra le probabilità, che non dipende dalla costante di normalizzazione. Appurato che tale differenza si aggira intorno a 100, le catene 4 e 5 potrebbero essere tranquillamente scartate, conservando le altre, tuttavia questo lavoro manuale può essere risparmiato attraverso una fase di *warm-up* più decorosa, ad esempio incrementandone la lunghezza, anche se, come suggerito dalla Figura 4.10, con una fase di campionamento sufficientemente lunga tutte le catene dovrebbero raggiungere il *Typical Set*. Inoltre, l'*output* del modello ci avverte della presenza di 193 transizioni divergenti che, come discusso nei paragrafi 2.3 e 2.4.2, sono sintomo di una legge di probabilità più difficile da esplorare, per cui è necessario aumentare la precisione dell'integratore simplettico.

Conseguentemente, sono state fatte girare 10 catene in parallelo con 1000 iterazioni ciascuna sia per la fase di *warm-up* che per quella di campionamento effettivo e fissando il valore nominale desiderato per il tasso globale di accettazione pari a 0.9. Il tempo totale di esecuzione è di circa 8 secondi su un computer portatile Lenovo con sistema operativo Linux, il codice R per riprodurre le analisi è reso disponibile in appendice C.4.

In Figura 4.11 sono riportati il tasso di accettazione globale, in nero, e i tassi di riflessione per $\tilde{t}_1^*$ e $\tilde{t}_2^*$, in rosso tratteggiato. Il tasso di accettazione globale empirico si avvicina al valore nominale desiderato, in questo caso scelto pari a 0.9 per evitare

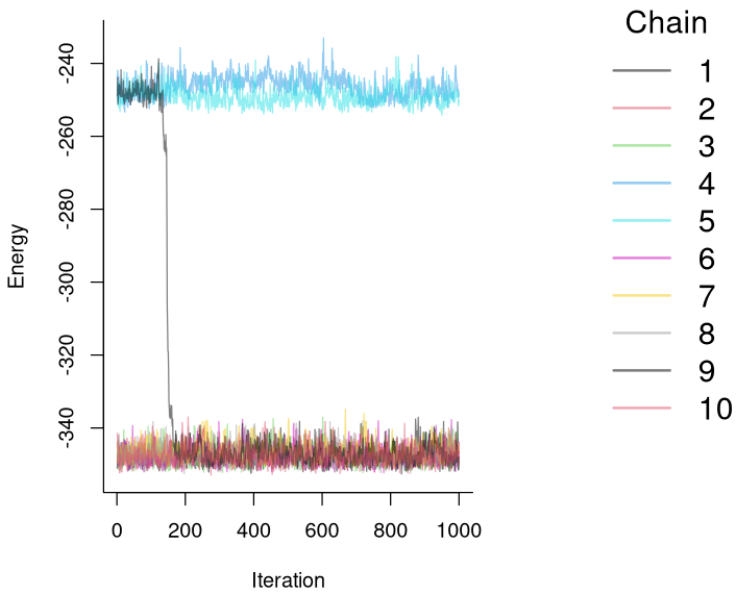

Figura 4.10: Catene di Markov per i livelli di energia del modello con punti di cambio con una fase di warm up non sufficientemente prolungata.

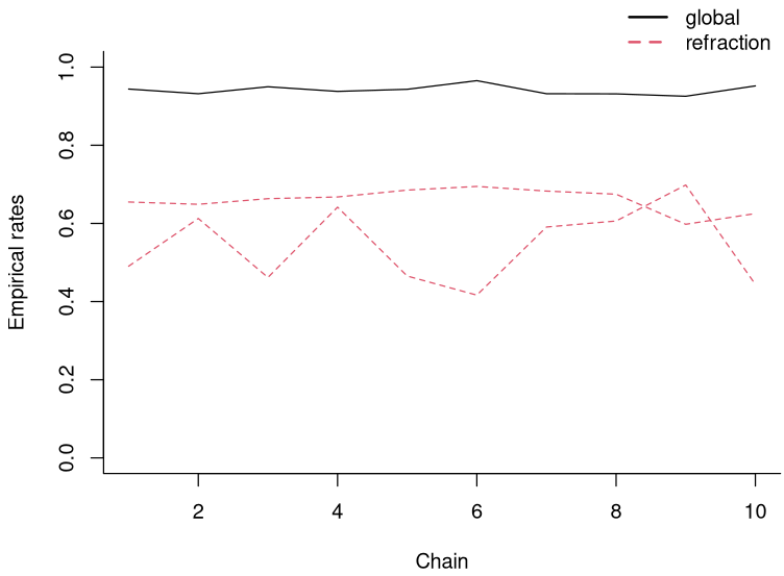

Figura 4.11: Tassi di accettazione Metropolis globale e di rifrazione per le componenti discontinue del modello con punti di cambio.

di incorrere in transizioni divergenti, mentre i tassi di rifrazione osservati si aggirano intorno a 0.6 come previsto. In questo esempio tuttavia, l'adattamento di un

ϵ diverso per i θ_J non ha fornito risultati soddisfacenti e si è preferito concentrare l'adattamento solo sul tasso globale, penalizzando scostamenti da quello di riflessione, come mostrato in (3.9). Le motivazioni di questi imprevisti, che rendono un approccio preferibile ad un altro, necessitano di uno studio più approfondito per comprenderne la natura.

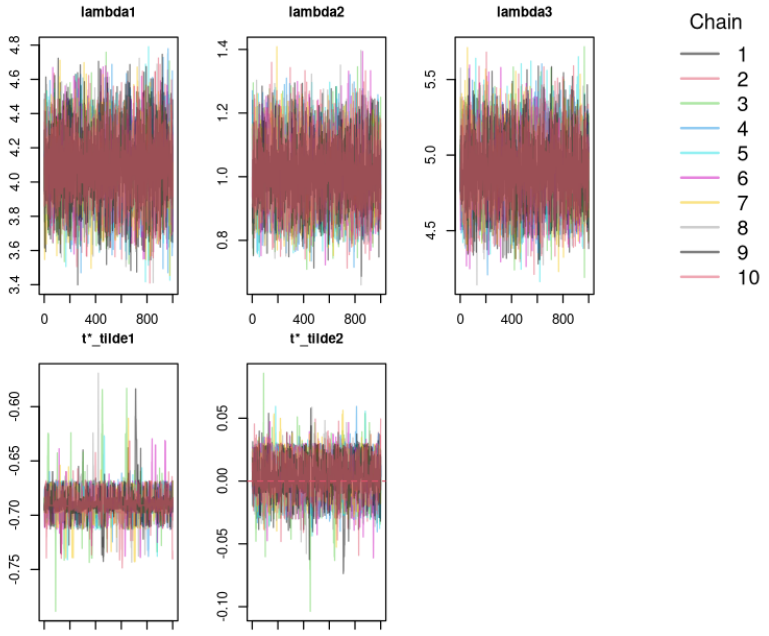

Figura 4.12: Catene di Markov per ogni parametro del modello con punti di cambio.

Nelle Figure 4.12 e 4.13 sono mostrate le catene di Markov per ogni parametro e per i livelli di energia, le quali evidenziano in generale un buon *mixing*. Per quanto riguarda i parametri $\tilde{t}_1^*$ e $\tilde{t}_2^*$, è normale aspettarsi dei salti così repentini di tanto in tanto, perché significa che il processo riesce a spostare saltuariamente il punto di cambio in zone meno probabili, sfondando una o più frontiere di discontinuità. La Figura 4.13 conferma che tutte le catene esplorano lo stesso *Typical Set* adesso.

In Figura 4.14 sono riportate le densità univariate e bivariate a posteriori, senza rimappare sulla scala originale i parametri discreti, mentre in Figura 4.15 tale trasformazione inversa è stata applicata. Come si può notare, il modello inferisce correttamente i valori dei tassi e dei punti di cambio.

In Figura 4.16 sono mostrate le frequenze assolute della lunghezza delle traiettorie per ciascuna catena. Anche in questo caso si nota un andamento comune ca-

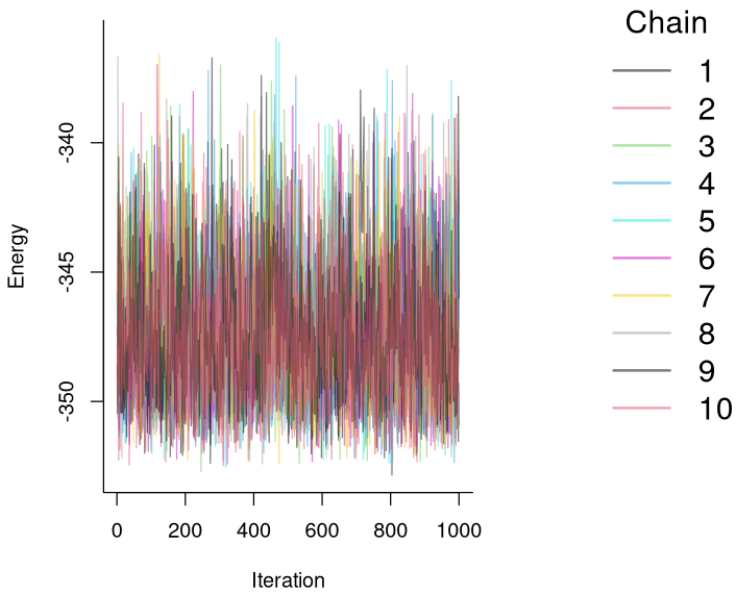

Figura 4.13: Catene di Markov per i livelli di energia del modello con punti di cambio con una fase di warm up adeguata.

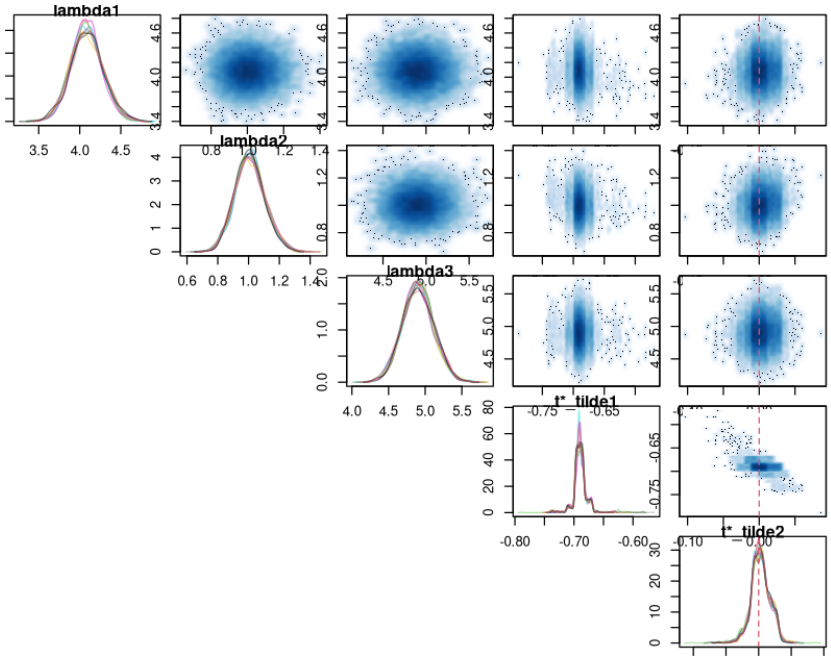

Figura 4.14: Densità univariate e bivariate del modello con punti di cambio con parametri discontinui nella riparametrizzazione per le componenti discrete.

ratterizzato da un numero di passi maggiormente concentrato sotto i 10 ma talvolta eccedente.

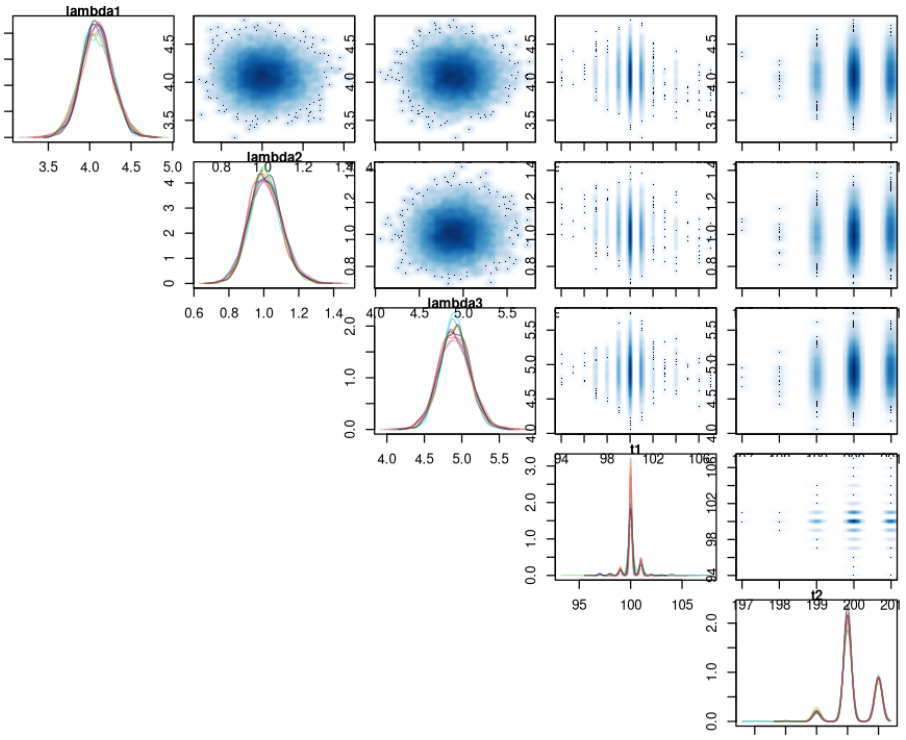

Figura 4.15: Densità univariate e bivariate del modello con punti di cambio nella parametrizzazione originale.

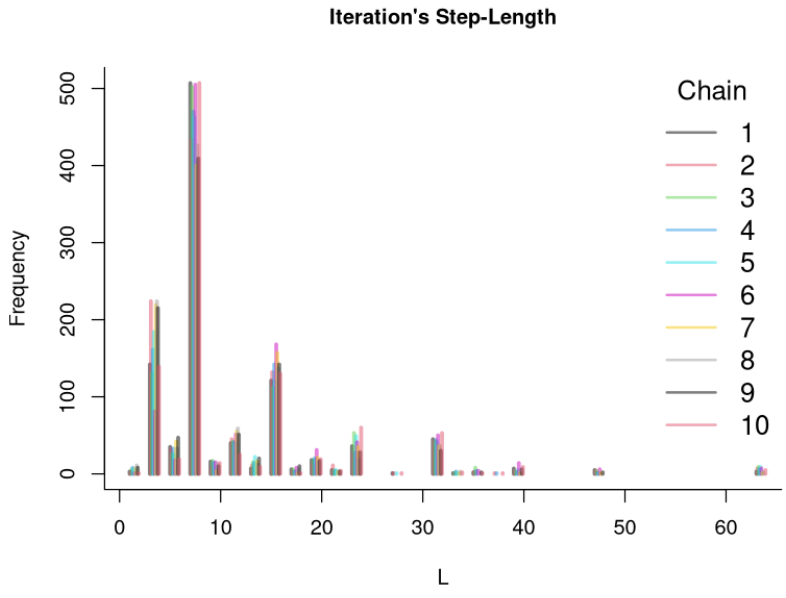

Figura 4.16: Frequenze assolute del numero di integrazioni per traiettoria nel modello con punti di cambio.

La Figura 4.17 presenta i grafici delle auto-correlazioni empiriche, le quali mostrano ancora una volta una leggera anti-correlazione per le componenti continue e una più marcata auto-correlazione per le componenti discontinue che non eccede mai 0.6 e descresce rapidamente.

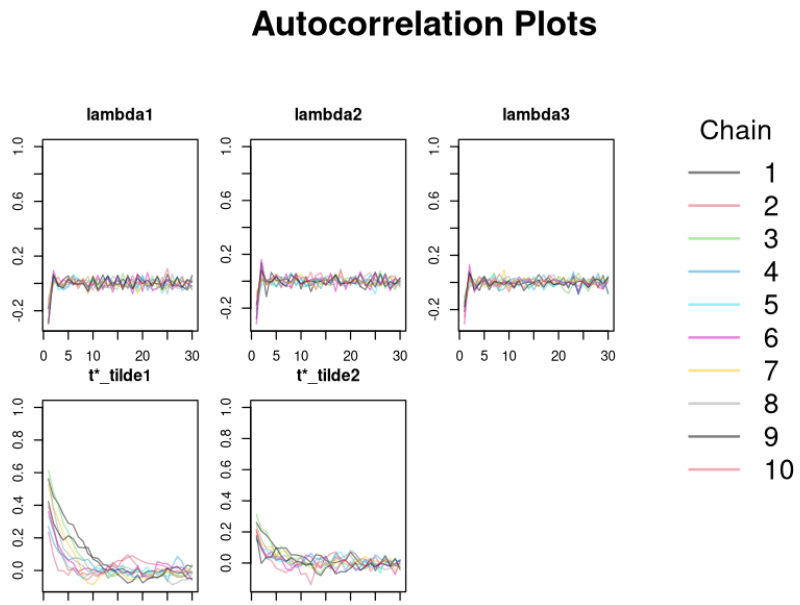

Figura 4.17: Funzioni di auto-correlazioni empiriche di ciascuna catena del modello con punti di cambio.

Tutte le diagnostiche descritte nel paragrafo precedente sono rispettate anche in questo caso. Il numero di campioni effettivamente indipendenti per le componenti continue è circa 15000, ma è una quantità sovrastimata dalla presenza di anti-correlazione, mentre per $\tilde{t}_1^*$ e $\tilde{t}_2^*$ tale valore si aggira rispettivamente intorno a 3000 e 5000. Data la natura generativa del modello, è possibile valutarne la bontà di adattamento attraverso una simulazione dalla distribuzione predittiva, i cui campioni sono confrontabili con i dati osservati. In Figura 4.18 sono riportati gli intervalli di credibilità all'80% per il numero di eventi rari verificatisi predetti in questo modo. Come aspettato, tali intervalli sembrano seguire correttamente l'andamento della variabile risposta nel tempo.

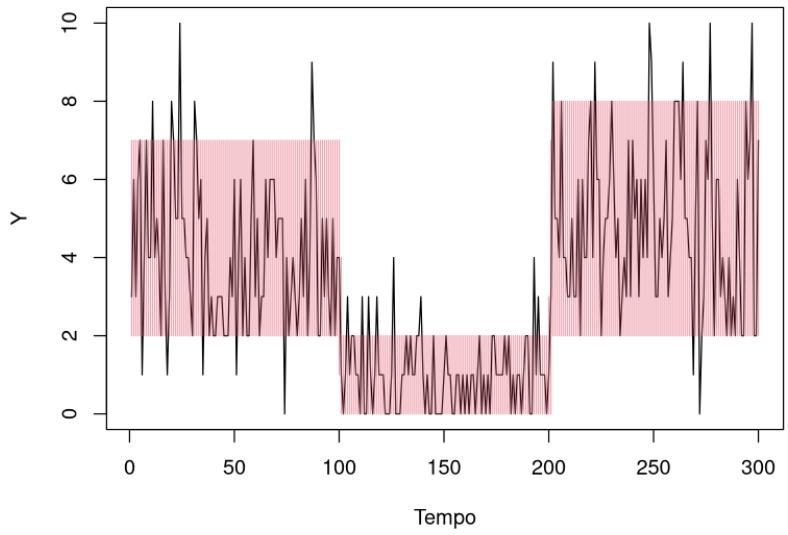

Figura 4.18: Intervalli di credibilità per le previsioni del modello con punti di cambio sui dati originali.

4.2.1 Confronto con altri algoritmi

Anche in questo caso si è voluto confrontare il DHMC, con criterio di terminazione basato sul prodotto interno (DNUTS), con altri possibili algoritmi. Il metro di paragone, analogamente a quello utilizzato in 4.1.1, è sempre dato dal numero di campioni effettivamente indipendenti (3.11), estratti per unità di tempo. Le procedure alternative considerate sono le seguenti:

- No U-Turn Sampler (NUTS), sul modello che marginalizza i parametri discreti, la derivazione della relativa energia potenziale e del suo gradiente sono disponibili in appendice B.2.2;
- Random Walk Metropolis (RWM), sul modello *completo* con *proposal* gaussiana per i parametri continui e che propone un passo avanti o indietro per quelli discreti;
- Random Walk Metropolis (RWM_M), sul modello che marginalizza i parametri discreti con *proposal* gaussiana.

- Gibbs Sampler, i passaggi per trovare le *full conditional* di ogni parametro sono riportati in appendice B.2.3;

Anche in questo caso, come nella sezione 4.1.1, i confronti sono stati fatti considerando solo la fase di campionamento, con dei parametri tarati manualmente e facendo partire tutte le 10 catene dalla media a posteriori, trovata precedentemente con l'esecuzione di DNUTS. Ogni catena è stata eseguita in parallelo con una numerosità Monte Carlo pari a 1000 e si registrato il tempo di esecuzione totale. Anche per questi confronti, il codice per replicare i seguenti risultati è reso disponibile in appendice C.4.1.

In Tabella 4.2 sono riportate le statistiche ricavate dei campioni simulati con i vari algoritmi. Anche in questo caso, le stime Monte Carlo di media e deviazione standard a posteriori sono concordi. Il DNUTS non sembra beneficiare troppo della marginalizzazione dei parametri discreti, mantiene comunque delle prestazioni comparabili in termini di ESS, ma non si può dire lo stesso se si tiene conto anche del tempo computazionale. Infatti, la procedura di marginalizzazione in questo contesto risulta essere estremamente dispendiosa, richiedendo decine di minuti per completare ciascuna catena. Il Random Walk Metropolis, seppur molto rapido, ha delle prestazioni molto scarse, mentre la sua versione che marginalizza i parametri discreti risulta essere migliore sotto entrambe le metriche. Il Gibbs Sampler produce catene molto poco auto-correlate, ma anche il suo tempo di esecuzione è rallentato dal calcolo della *full conditional* per (t_1, t_2), la cui formula è riportata in appendice B.2.3. Questo nonostante le statistiche sufficienti, per ogni configurazione di queste due variabili, siano state salvate in memoria precedentemente e non si necessiti di calcolarle ogni volta. Per quanto riguarda i parametri continui, sia in termini di ESS, che di ESS al secondo, il DNUTS risulta essere il metodo con prestazioni migliori, salvo per l'ESS di λ_1, dove è superato dalla sua versione marginalizzata, mentre per gli altri parametri la situazione premia NUTS per quanto riguarda ESS, e RWM per quanto riguarda ESS/s anche se, come citato nei confronti precedenti, una statistica come l'*Effective Sample Size* non è molto giustificata in un contesto discreto con poche modalità effettivamente visitate da ogni catena.

	DNUTS	NUTS	RWM	RWM_M	Gibbs
$\bar{\lambda}_1$	4.08	4.08	4.09	4.08	4.09
S_{λ_1}	0.19	0.18	0.21	0.20	0.20
ESS_{λ_1}	13951.89	**14531.89**	0.05	1207.29	10117.51
ESS_{λ_1}/s	**2233.02**	0.42	0.03	4.76	23.72
$\bar{\lambda}_2$	1.01	1.09	1.00	1.10	1.09
S_{λ_2}	0.09	0.10	0.10	0.11	0.11
ESS_{λ_2}	**16315.34**	8673.85	0.02	376.80	7187.68
ESS_{λ_3}/s	**4350.76**	0.25	0.01	1.49	16.85
$\bar{\lambda}_3$	4.90	4.96	4.91	4.97	4.95
S_{λ_3}	0.20	0.21	0.23	0.22	0.22
ESS_{λ_3}	**16682.41**	16126.98	0.06	1278.26	10719.83
ESS_{λ_3}/s	**4448.64**	0.47	0.04	5.04	25.13
$\bar{t_1}$	100.06	100.01	100.08	100.01	100.02
S_{t_1}	0.62	0.72	0.55	0.76	0.77
ESS_{t_1}	1636.13	**10000.00**	2089.85	5948.80	9551.28
ESS_{t_1}/s	436.30	2.89	**1331.12**	23.46	22.39
$\bar{t_2}$	200.21	201.40	200.21	201.40	201.40
S_{t_2}	0.54	0.58	0.56	0.58	0.58
ESS_{t_2}	2733	**10000.00**	3766.19	2380.12	8096.20
ESS_{t_2}/s	728.98	2.89	**2398.85**	9.39	18.98

Tabella 4.2: Statistiche descrittive dei diversi algoritmi applicati al modello per due punti di cambio. Per ogni parametro sono riportati: media e deviazione standard a posteriori, *Effective Sample Size*, puro e diviso per il tempo trascorso in secondi. Il pedice M indica che l'algoritmo è stato applicato al modello che marginalizza i parametri discreti.

Conclusioni

Attraverso l'applicazione dei criteri di terminazione enunciati in 3.4 è possibile ottenere un'implementazione più efficiente dell'originale algoritmo DHMC (Nishimura et al., 2020), capace di far terminare l'integrazione della traiettoria in un tempo variabile, che dipende dalla geometria locale della densità a posteriori. Tale risultato è stato giustificato sia a livello teorico che con uno studio di simulazione. Per quanto riguarda l'algoritmo nel suo complesso, un aspetto cruciale, che richiede approfondimento, riguarda la validità delle procedure adattive per il passo di integrazione, in particolare si ricerca un metodo flessibile che riesca a conciliare il tasso di accettazione globale dell'HMC con i vari tassi di rifrazione per le componenti discontinue. All'interno della libreria XDNUTS, sono proposte tre opzioni, descritte nella sezione 3.4, la cui efficacia dipende dal contesto di applicazione e non è detto che siano esaustive. Inoltre, i valori dei parametri della procedura adattiva sono basati sui risultati prodotti in Hoffman et al. (2014), calibrati per l'algoritmo NUTS, per cui è necessario appurare se siano ancora validi in questo contesto o se possano esistere delle configurazioni migliori.

Vale la pena, inoltre, spendere due parole riguardo i pregi e difetti che il DHMC possiede rispetto ad alti metodi MCMC applicabili in questi contesti, dal momento che la libreria XDNUTS erediterà, per forza di cose, queste sue proprietà. Nel caso di leggi di probabilità discontinue, un approccio utilizzato nell'ambito del Machine Learning consiste nell'approssimare la legge nei punti di discontinuità attraverso delle funzioni apposite, come ad esempio una sigmoide con parametro di scala tendente a zero. Ciò permette il calcolo del gradiente anche per queste componenti e quindi di utilizzare l'HMC/NUTS classico, tuttavia, come sottolineato da Michael

Betancourt in un dibattito sul forum di STAN[1], ciò comporta due problemi principalmente: la bontà di approssimazione degli integratori simplettici, e quindi anche la validità dei metodi di ottimizzazione stocastica per la taratura di ϵ, viene meno e le traiettorie hamiltoniane risultanti saranno allineate al *Typical Set* della legge modificata, facendo dipendere l'efficienza del metodo dal tipo di funzione adoperata. Da questo punto di vista, il DHMC non è affetto dal primo problema, perché rinuncia all'utilizzo del gradiente delle componenti discontinue, mentre risulta essere affetto dal secondo solamente quando lo si utilizza per leggi di probabilità discrete a cui si applica la trasformazione (3.4). In quest'ultimo contesto, alternative possibili riguardano l'utilizzo di metodi che alternano passi di HMC/NUTS per le componenti continue a passi Gibbs o Metropolis-within-Gibbs per quelle discrete. Gli svantaggi di tali approcci sono riconducibili a quanto detto nei paragrafi 1.3 e 1.5, ovvero alla vicinanza rispetto ai valori correnti da parte di quelli proposti e/o alla mancanza di un criterio che guidi le *proposal* verso le zone a più alta massa di probabilità per le componenti discrete. Inoltre, se si utilizza l'HMC per le componenti continue, il campo vettoriale definito dalle equazioni di Hamilton sarà allineato soltanto ad un *Typical Set* locale, condizionato al valore corrente dei parametri discreti, e non a quello dell'intera distribuzione. Il DHMC alla fine è riconducibile a uno di questi metodi appena citati, dal momento che per le componenti discrete nel concreto si applica un passo di accettazione Metropolis mascherandolo con i termini riflessione e rifrazione. Vi è una differenza fondamentale però, che può essere visualizzata considerando il Guided Walk Metropolis descritto nel paragrafo 1.4: attraverso l'uso di una *data augmentation*, il DHMC supera il problema della limitatezza delle *proposal* anche per le componenti discrete, riuscendo a guidarle verso le zone a più alta massa di probabilità, ammesso che esistano dei percorsi da intraprendere, ovvero non vi siano *barriere energetiche* troppo grandi che separino le diverse mode della distribuzione portata nel continuo. I dati aumentati utilizzati, poi, tengono conto non solo dell'orientamento ma anche della direzione e, attraverso l'applicazione del-

[1] https://discourse.mc-stan.org/t/what-exactly-happens-for-only-piecewise-continuous-posterior-densities-gradients-or-wrong-gradients/26798

la legge di conservazione dell'energia, la procedura allinea anche le traiettorie per le componenti discrete al loro *Typical Set*, anche se queste risultano in parte snaturate dal fenomeno discusso a pagina 80 che si osserva in presenza di riflessioni al loro interno.

Infine, è bene puntualizzare come quello che forse era il reale obiettivo che ha motivato lo sviluppo di DHMC, cioè estendere l'HMC anche per parametri discreti, non sia stato realmente raggiunto, dal momento che, in assenza di un ordinamento nelle modalità questi, la validità del metodo è dubbia. Le conclusioni che si possono trarre sono quindi che si è ancora distanti dall'avere una procedura generale con cui fare inferenza bayesiana e l'utilizzo di procedure *ad hoc* resta sempre la scelta migliore.

Ringraziamenti

Si ringrazia il professor Bruno Scarpa per avere avuto la pazienza di seguirmi in questo progetto, il professor Mauro Bernardi per la grande disponibilità nel dispensare chiarimenti che hanno portato allo sviluppo della libreria XDNUTS, i professori Annalisa Cesaroni e Tommaso Dorigo per i loro utili consigli, Giorgio Craparo per i disegni delle Figure 2.1, 2.2 e per ultimo, ma non per questo meno importante, Marco Taffarello, per il grande aiuto fornitomi nel recuperare le nozioni di fisica necessarie alla comprensione dell'HMC.

Bibliografia

Andrieu, C. e Thoms, J. (2008), 'A tutorial on adaptive mcmc', *Statistics and computing* **18**, 343–373.

Betancourt, M. (2013*a*), A general metric for riemannian manifold hamiltonian monte carlo, *in* 'International Conference on Geometric Science of Information', Springer, pp. 327–334.

Betancourt, M. (2013*b*), 'Generalizing the no-u-turn sampler to riemannian manifolds', *arXiv preprint arXiv:1304.1920* .

Betancourt, M. (2014), 'Adiabatic monte carlo', *arXiv preprint arXiv:1405.3489* .

Betancourt, M. (2016*a*), 'Diagnosing suboptimal cotangent disintegrations in hamiltonian monte carlo', *arXiv preprint arXiv:1604.00695* .

Betancourt, M. (2016*b*), 'Identifying the optimal integration time in hamiltonian monte carlo', *arXiv preprint arXiv:1601.00225* .

Betancourt, M. (2017), 'A conceptual introduction to hamiltonian monte carlo', *arXiv preprint arXiv:1701.02434* .

Betancourt, M., Byrne, S. e Girolami, M. (2014), 'Optimizing the integrator step size for hamiltonian monte carlo', *arXiv preprint arXiv:1411.6669* .

Betancourt, M., Byrne, S., Livingstone, S. e Girolami, M. (2017), 'The geometric foundations of hamiltonian monte carlo', *arXiv preprint arXiv:1701.02434* .

Blackwell, D. (1947), 'Conditional expectation and unbiased sequential estimation', *The Annals of Mathematical Statistics* pp. 105–110.

Brémaud, P. (2013), *Markov chains: Gibbs fields, Monte Carlo simulation, and queues*, Vol. 31, Springer Science & Business Media.

Darrigol, O. e Renn, J. (2003), La nascita della meccanica statistica, *in* 'Storia della scienza. Vol. VII: L'ottocento', Istituto della Enciclopedia Italiana, pp. 496–507.

Duane, S., Kennedy, A. D., Pendleton, B. J. e Roweth, D. (1987), 'Hybrid monte carlo', *Physics letters B* **195**(2), 216–222.

Dunn, O. J. (1961), 'Multiple comparisons among means', *Journal of the American statistical association* **56**(293), 52–64.

Eddelbuettel, D. (2013), *Seamless R and C++ Integration with Rcpp*, Springer, New York. ISBN 978-1-4614-6867-7.

Efron, B. (1979), 'Computers and the theory of statistics: thinking the unthinkable', *SIAM review* **21**(4), 460–480.

Gelfand, A. E., Sahu, S. K. e Carlin, B. P. (1995), 'Efficient parametrisations for normal linear mixed models', *Biometrika* **82**(3), 479–488.

Gelman, A., Lee, D. e Guo, J. (2015), 'Stan: A probabilistic programming language for bayesian inference and optimization', *Journal of Educational and Behavioral Statistics* **40**(5), 530–543.

Gelman, A. e Rubin, D. B. (1992), 'Inference from iterative simulation using multiple sequences', *Statistical science* **7**(4), 457–472.

Geman, S. e Geman, D. (1984), 'Stochastic relaxation, gibbs distributions, and the bayesian restoration of images', *IEEE Transactions on pattern analysis and machine intelligence* (6), 721–741.

Geyer, C. J. (2011), 'Introduction to markov chain monte carlo', *Handbook of markov chain monte carlo* **20116022**(45), 22.

Geyer, C. J. et al. (1991), 'Computing science and statistics: Proceedings of the 23rd symposium on the interface', *American Statistical Association, New York* **156**.

Gibbs, J. W. (1902), *Elementary principles in statistical mechanics: developed with especial reference to the rational foundations of thermodynamics*, C. Scribner's sons.

Gilks, W. R., Richardson, S. e Spiegelhalter, D. (1995), *Markov chain Monte Carlo in practice*, CRC press.

Gilks, W. R., Thomas, A. e Spiegelhalter, D. J. (1994), 'A language and program for complex bayesian modelling', *Journal of the Royal Statistical Society: Series D (The Statistician)* **43**(1), 169–177.

Gustafson, P. (1998), 'A guided walk metropolis algorithm', *Statistics and computing* **8**, 357–364.

Hans, C., Dobra, A. e West, M. (2007), 'Shotgun stochastic search for "large p" regression', *Journal of the American Statistical Association* **102**(478), 507–516.

Hastings, W. K. (1970), 'Monte carlo sampling methods using markov chains and their applications'.

Hoffman, M. D., Gelman, A. et al. (2014), 'The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo.', *J. Mach. Learn. Res.* **15**(1), 1593–1623.

Kalman, R. E. (1960), 'A new approach to linear filtering and prediction problems', *Transactions of the ASME-Journal of Basic Engineering* **82**, 35–45.

Laird, N., Lange, N. e Stram, D. (1987), 'Maximum likelihood computations with repeated measures: application of the em algorithm', *Journal of the American Statistical Association* **82**(397), 97–105.

LaMont, C. H. e Wiggins, P. A. (2019), 'Correspondence between thermodynamics and inference', *Physical Review E* **99**(5), 052140.

Leimkuhler, B. e Reich, S. (2004), *Simulating hamiltonian dynamics*, number 14, Cambridge university press.

Liu, J. S. (1994), 'The collapsed gibbs sampler in bayesian computations with applications to a gene regulation problem', *Journal of the American Statistical Association* **89**(427), 958–966.

Llorente, F., Martino, L., Delgado, D. e Lopez-Santiago, J. (2023), 'Marginal likelihood computation for model selection and hypothesis testing: an extensive review', *SIAM review* **65**(1), 3–58.

Maguire, B. A., Pearson, E. S. e Wynn, A. (1952), 'The time intervals between industrial accidents', *Biometrika* **39**(1/2), 168–180.

Mangoubi, O., Pillai, N. S. e Smith, A. (2018), 'Does hamiltonian monte carlo mix faster than a random walk on multimodal densities?', *arXiv preprint arXiv:1808.03230* .

Manildo, P. (2024), *XDNUTS: Discontinuous Hamiltonian Monte Carlo with Varying Trajectory Length*. R package version 1.1.
URL: https://github.com/paolomanildo/XDNUTS

Marinari, E. e Parisi, G. (1992), 'Simulated tempering: a new monte carlo scheme', *Europhysics letters* **19**(6), 451.

Mazzoldi, P., Nigro, M. e Voci, C. (1991), *Fisica. Volume 1: Meccanica-Termodinamica: C. Voci*, EdiSES.

McElreath, R. (2018), *Statistical rethinking: A Bayesian course with examples in R and Stan*, Chapman and Hall/CRC.

McLachlan, R. I., Perlmutter, M. e Quispel, G. (2004), 'On the nonlinear stability of symplectic integrators', *BIT Numerical Mathematics* **44**, 99–117.

McLachlan, R. I. e Quispel, G. R. W. (2002), 'Splitting methods', *Acta Numerica* **11**, 341–434.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. e Teller, E. (1953), 'Equation of state calculations by fast computing machines', *The journal of chemical physics* **21**(6), 1087–1092.

Mohasel Afshar, H. e Domke, J. (2015), 'Reflection, refraction, and hamiltonian monte carlo', *Advances in neural information processing systems* **28**.

Neal, R. M. (2011), *Handbook of Markov Chain Monte Carlo, capitolo 5: MCMC using Hamiltonian Dynamics*, CRC press.

Nesterov, Y. (2009), 'Primal-dual subgradient methods for convex problems', *Mathematical programming* **120**(1), 221–259.

Nishimura, A. e Dunson, D. (2020), 'Recycling intermediate steps to improve hamiltonian monte carlo', *Bayesian Analysis* **15**(4).
URL: *http://dx.doi.org/10.1214/19-BA1171*

Nishimura, A., Dunson, D. B. e Lu, J. (2020), 'Discontinuous hamiltonian monte carlo for discrete parameters and discontinuous likelihoods', *Biometrika* **107**(2), 365–380.

Plummer, M. et al. (2003), Jags: A program for analysis of bayesian graphical models using gibbs sampling, *in* 'Proceedings of the 3rd international workshop on distributed statistical computing', Vol. 124, Vienna, Austria, pp. 1–10.

Rao, C. R. (1945), 'Information and the accuracy attainable in the estimation of statistical parameters', *Bulletin of the Calcutta Mathematical Society* **37**, 81–91.

Robert, C. P., Casella, G. e Casella, G. (1999), *Monte Carlo statistical methods*, Vol. 2, Springer.

Robert, C. P. e Roberts, G. (2021), 'Rao–blackwellisation in the markov chain monte carlo era', *International Statistical Review* **89**(2), 237–249.

Roberts, G. O. e Rosenthal, J. S. (2001), 'Optimal scaling for various metropolis-hastings algorithms', *Statistical science* **16**(4), 351–367.

Roberts, G. O. e Rosenthal, J. S. (2009), 'Examples of adaptive mcmc', *Journal of computational and graphical statistics* **18**(2), 349–367.

Shapiro, S. S. e Wilk, M. B. (1965), 'An analysis of variance test for normality (complete samples)', *Biometrika* **52**(3-4), 591–611.

Simpson, E. H. (1951), 'The interpretation of interaction in contingency tables', *Journal of the Royal Statistical Society: Series B (Methodological)* **13**(2), 238–241.

STAN development team (2021), 'Stan modelling language users guide and reference manual v. 2.27'.

Team, R. C. (2024), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.
URL: https://www.R-project.org/

Vishwanath, S. e Tak, H. (2024), 'Repelling-attracting hamiltonian monte carlo', *arXiv preprint arXiv:2403.04607* .

Watanabe, S. (2018), *Mathematical theory of Bayesian statistics*, chapman and hall/cRc.

Wilcoxon, F. (1945), 'Individual comparisons by ranking methods', *Biometrics bullettin* **1**(6), 80–83.

Appendice A

Dimostrazioni

A.1 Dettaglio bilanciato del nucleo di transizione Metropolis-Hastings

Seguendo l'approccio utilizzato in Watanabe (2018, capitolo 7, sezione 1), il nucleo definito in 1.1 è scrivibile come

$$K(x'|x) = q(x'|x)\alpha(x, x') + \mathcal{I}(x' = x)\int_{\mathcal{X}} q(y|x)\{1 - \alpha(x, y)\}dy,$$

per cui è possibile scrivere:

$$\begin{aligned}
K(x'|x)\pi(x) =& q(x'|x)\min\left\{1, \frac{\pi(x')q(x|x')}{\pi(x)q(x'|x)}\right\}\pi(x) \\
&+ \mathcal{I}(x' = x)\int_{\mathcal{X}} q(y|x)\min\left\{0, 1 - \frac{\pi(y)q(x|y)}{\pi(x)q(y|x)}\right\}dy\pi(x) \\
=& \min\left\{\pi(x)q(x'|x), \frac{\pi(x')q(x|x')}{1}\right\} \\
&+ \mathcal{I}(x' = x)\pi(x')\int_{\mathcal{X}} q(y|x')\min\left\{0, 1 - \frac{\pi(y)q(x'|y)}{\pi(x')q(y|x')}\right\}dy \\
=& q(x|x')\min\left\{\frac{\pi(x)q(x'|x)}{\pi(x')q(x|x')}, \frac{1}{1}\right\}\pi(x') \\
&+ \mathcal{I}(x = x')\int_{\mathcal{X}} q(y|x')\min\left\{0, 1 - \frac{\pi(y)q(x'|y)}{\pi(x')q(y|x')}\right\}dy\pi(x') \\
=& \left[q(x|x')\alpha(x', x) + \mathcal{I}(x = x')\int_{\mathcal{X}} q(y|x')\{1 - \alpha(x', y)\}dy\right]\pi(x') \\
=& K(x|x')\pi(x')
\end{aligned}$$

A.2 Minimizzazione dell'azione da parte delle traiettorie lagrangiane

Partendo da (2.8), sfruttando la *regola della catena*, le proprietà della derivata totale e dello scambio dell'ordine di derivazione e integrazione si ottiene:

$$\begin{aligned}\left.\frac{dA}{d\epsilon}\right|_{\epsilon=0} &= \int_{t_0}^{T}\left[\frac{\partial L}{\partial t}\cdot\frac{\partial t}{\partial \epsilon}+\frac{\partial L}{\partial \theta(t)}\cdot\frac{\partial \theta(t)}{\partial \epsilon}+\frac{\partial L}{\partial \theta'(t)}\cdot\frac{\partial \theta'(t)}{\partial \epsilon}\right]\Bigg|_{\epsilon=0} dt = 0 \\ &= \int_{t_0}^{T}\left[\frac{\partial L}{\partial t}\cdot 0+\frac{\partial L}{\partial \theta(t)}\cdot\eta(t)+\frac{\partial L}{\partial \theta'(t)}\cdot\eta'(t)\right]\Bigg|_{\epsilon=0} dt = 0\end{aligned}$$

Siccome per $\epsilon \to 0$ si ha che $\theta(t) \to \hat{\theta}(t)$ e $\theta'(t) \to \hat{\theta}'(t)$, possiamo scrivere:

$$\begin{aligned}\left.\frac{dA}{d\epsilon}\right|_{\epsilon=0} &= \int_{t_0}^{T}\left[\frac{\partial L}{\partial \hat{\theta}(t)}\cdot\eta(t)+\frac{\partial L}{\partial \hat{\theta}'(t)}\cdot\eta'(t)\right] dt = 0 \\ &= \int_{t_0}^{T}\frac{\partial L}{\partial \hat{\theta}(t)}\cdot\eta(t)dt+\frac{\partial L}{\partial \hat{\theta}'(t)}\cdot\eta(t)\Bigg|_{t_0}^{T}-\int_{t_0}^{T}\frac{d}{dt}\frac{\partial L}{\partial \hat{\theta}'(t)}dt = 0\end{aligned}$$

Sfruttando il fatto che $\eta(t_0) = \eta(T) = 0$ allora

$$\left.\frac{dA}{d\epsilon}\right|_{\epsilon=0} = \int_{t_0}^{T}\left(\frac{\partial L}{\partial \hat{\theta}(t)}-\frac{d}{dt}\frac{\partial L}{\partial \hat{\theta}'(t)}\right)\cdot\eta(t)dt = 0$$

Siccome la funzione $\eta(t)$ è arbitraria, affinché tale integrale risulti pari a zero è necessario che l'argomento che moltiplica tale funzione lo sia sempre, per ogni possibile t. Questa condizione è quindi espressa dall'equazione (2.9).

Inoltre, ogni poossibile soluzione definisce una funzione stazionaria rispetto al funzionale. Per stabilire la natura di questa stazionarietà è necessario calcolare la derivata seconda rispetto ad ϵ, che, ripetendo gli stessi passaggi un'altra volta, fornisce il seguente risultato:

$$\left.\frac{d^2A}{d^2\epsilon}\right|_{\epsilon=0} = \int_{t_0}^{T}\left[\eta(t)^2\cdot\frac{\partial^2 L}{\partial^2\theta(t)}+2\eta(t)\eta'(t)\cdot\frac{\partial^2 L}{\partial\theta(t)\partial\theta'(t)}+\eta'(t)^2\cdot\frac{\partial^2 L}{\partial^2\theta(t)}\right]dt$$

La funzione $\theta(t)$ minimizzerà l'azione se tale integrale, interpretabile come una forma quadratica in uno spazio infinito dimensionale, è sempre positivo, indipendentemente dalle possibili $\eta(t)$. Tale condizione, spesso difficile da dimostrare, è tipicamente assunta essere vera.

A.3 Preservazione del volume da parte del flusso hamiltoniano

Seguendo l'approccio mostrato in Neal (2011), se si sostituiscono i risultati delle equazioni di Hamilton enunciati in (2.10) all'equazione (2.15), si ottiene una mappa il cui Jacobiano è dato da:

$$J = \begin{bmatrix} 1 + \epsilon \frac{\partial^2 H}{\partial m \partial \theta^T} & \epsilon \frac{\partial^2 H}{\partial m \partial m^T} \\ -\epsilon \frac{\partial^2 H}{\partial \theta \partial \theta^T} & 1 - \epsilon \frac{\partial^2 H}{\partial \theta \partial m^T} \end{bmatrix} + O(\epsilon^2)$$

Il suo determinante risulta quindi:

$$\begin{aligned} \det(J) &= 1 - \epsilon \frac{\partial^2 H}{\partial \theta \partial m^T} + \epsilon \frac{\partial^2 H}{\partial m \partial \theta^T} \\ &\quad - \epsilon^2 \frac{\partial^2 H}{\partial \theta \partial m^T} \cdot \frac{\partial^2 H}{\partial m \partial \theta^T} + \epsilon^2 \frac{\partial^2 H}{\partial \theta \partial \theta^T} \cdot \frac{\partial^2 H}{\partial m \partial m^T} + O(\epsilon^2) \\ &= 1 - \epsilon \frac{\partial^2 H}{\partial \theta \partial m^T} + \epsilon \frac{\partial^2 H}{\partial m \partial \theta^T} + O(\epsilon^2) \end{aligned}$$

Dal momento che $K(m)$ non dipende da θ e $U(\theta)$ non dipende da m allora:

$$\det(J) = 1 + O(\epsilon^2)$$

Siccome il flusso hamiltoniano gode di proprietà associativa, è possibile scomporlo come la successione di un infinito numero di piccole traiettorie che si protraggono per un tempo $\epsilon \to 0$:

$$\phi_t^H = \bigcirc_{i=1}^{\infty} \phi_\epsilon^H$$

Ciascuna di queste ha determinate pari ad 1 e per le proprietà del determinante $\det(J)$ è pari alla loro produttoria. Successivamente, è banale dimostrare che $\det(F) = \pm 1$. Quindi la mappa $t = F \circ \phi_t^H$ nel complesso preserva il volume originario.

A.4 Invarianza delle operazioni di raddoppio della traiettoria rispetto alla distribuzione canonica

Dimostrazione presa direttamente da Betancourt (2017), leggermente rivisitata.

Nel primo caso, con $p = \frac{w_{old}}{w_{old}+w_{new}}$, la dimostrazione è la seguente:

$$\begin{aligned}
\sum_z K(z'|z)\pi(z)\mathcal{I}(z \in t) &= \sum_z K(z'|t)\pi(z)\mathcal{I}(z \in t) \\
&= \sum_z \frac{e^{-H(z')}}{w_{old}+w_{new}}\mathcal{I}(z' \in t)\pi(z)\mathcal{I}(z \in t) \\
&= \frac{e^{-H(z')}}{w_{old}+w_{new}}\mathcal{I}(z' \in t)\sum_{z \in t}\pi(z) \\
&= \frac{e^{-H(z')}}{w_{old}+w_{new}}\mathcal{I}(z' \in t)\frac{w_{old}+w_{new}}{Z} \\
&= \frac{e^{-H(z')}}{Z}\mathcal{I}(z' \in t) \\
&= \pi(z')\mathcal{I}(z' \in t)
\end{aligned}$$

dove ricordiamo che $Z = \int_{\Theta \times \mathcal{M}} e^{-H(z)}dz$ corrisponde alla funzione di partizione della distribuzione canonica.

Nel secondo caso, con $p = 1 - \min\left\{1, \frac{w_{new}}{w_{old}}\right\}$, si avrà che:

$$\begin{aligned}
\mathbb{K}(z'|t) = &\left(1 - \min\left\{1, \frac{w_{new}}{w_{old}}\right\}\right) \cdot \frac{e^{-H(z')}}{w_{old}} \cdot \mathcal{I}(z' \in t_{old}) \\
&+ \min\left\{1, \frac{w_{new}}{w_{old}}\right\} \cdot \frac{e^{-H(z')}}{w_{new}} \cdot \mathcal{I}(z' \in t_{new})
\end{aligned}$$

Come prima, per dimostrare che l'invarianza alla distribuzione canonica venga rispettata, bisogna pensare a questa procedura iterativa di raddoppio come una sequenza di operazioni a cui la distribuzione obiettivo, condizionatamente alla corrente traiettoria $t = t_{old} \cup t_{new}$, risulta essere invariante:

$$\begin{aligned}
\sum_z K(z'|z)\pi(z)\mathcal{I}(z \in t) &= \sum_z K(z'|z)\pi(z)\mathcal{I}(z \in t_{old}) + \sum_z K(z'|z)\pi(z)\mathcal{I}(z \in t_{new}) \\
&= \mathbb{K}(z'|t_{old})\sum_z \pi(z)\mathcal{I}(z \in t_{old}) + \mathbb{K}(z'|t_{new})\sum_z \pi(z)\mathcal{I}(z \in t_{new}) \\
&= \mathbb{K}(z'|t_{old}) \cdot \frac{w_{old}}{Z} + \mathbb{K}(z'|t_{new}) \cdot \frac{w_{new}}{Z}
\end{aligned}$$

combinando i risultati precedenti si ottiene:

$$\begin{aligned}
\mathbb{K}(z'|t_{old}) \cdot \frac{w_{old}}{Z} = &\left(1 - \min\left\{1, \frac{w_{new}}{w_{old}}\right\}\right) \cdot \frac{e^{-H(z')}}{Z} \cdot \mathcal{I}(z' \in t_{old}) \\
&+ \min\left\{\frac{w_{old}}{w_{new}}, 1\right\} \cdot \frac{e^{-H(z')}}{Z} \cdot \mathcal{I}(z' \in t_{new}) \\
\mathbb{K}(z'|t_{new}) \cdot \frac{w_{new}}{Z} = &\left(1 - \min\left\{1, \frac{w_{old}}{w_{new}}\right\}\right) \cdot \frac{e^{-H(z')}}{Z} \cdot \mathcal{I}(z' \in t_{new}) \\
&+ \min\left\{\frac{w_{new}}{w_{old}}, 1\right\} \cdot \frac{e^{-H(z')}}{Z} \cdot \mathcal{I}(z' \in t_{old})
\end{aligned}$$

sommando i due termini ci si ritrova con:

$$\begin{aligned}
\sum_z K(z'|z)\pi(z)\mathcal{I}(z \in t) = \quad &\mathcal{I}(z' \in t_{old}) \cdot \pi(z') \cdot \left(1 - \min\left\{1, \frac{w_{new}}{w_{old}}\right\} + \min\left\{\frac{w_{new}}{w_{old}}, 1\right\}\right) \\
&+\mathcal{I}(z' \in t_{new}) \cdot \pi(z') \cdot \left(\min\left\{\frac{w_{old}}{w_{new}}, 1\right\} + 1 - \min\left\{1, \frac{w_{old}}{w_{new}}\right\}\right) \\
= \quad &\pi(z') \cdot [\mathcal{I}(z' \in t_{old}) + \mathcal{I}(z' \in t_{new})] \\
= \quad &\pi(z') \cdot \mathcal{I}(z' \in t)
\end{aligned}$$

A.5 Preservazione del volume dell'integratore simplettico proposto nell'Algoritmo 3

In Nishimura et al. (2020) si dimostra la preservazione del volume dell'integratore proposto nell'Algoritmo 3 per prima cosa dividendolo come una sequenza di mappe:

$$\Phi_{L,\epsilon} = F \circ \bigcirc_{i=1}^{L} \left[\Phi^I_{\frac{\epsilon}{2}} \circ \left(\bigcirc_{j=1}^{k} \Phi^J_{j,\epsilon}\right) \circ \Phi^I_{\frac{\epsilon}{2}}\right]$$

dove F e $\Phi^I_{\frac{\epsilon}{2}}$ sono le stesse mappe dell'HMC classico, descritte nella sezione 2.3.2, e pertanto si sa che il loro determinante è pari ad 1. La mappa dell'integratore per i parametri discreti è essa stessa una sequenza di mappe, quindi la dimostrazione di tale proprietà per una sola di queste la farà ereditare anche all'integratore

complessivo.

$$\Phi^J_{j,\epsilon}: \begin{pmatrix} \theta_1 \\ \vdots \\ \theta_j \\ \vdots \\ \theta_J \\ m_1 \\ \vdots \\ m_j \\ \vdots \\ m_J \end{pmatrix} \to \begin{pmatrix} \theta_1 \\ \vdots \\ \theta^*_j \\ \vdots \\ \theta_J \\ m_1 \\ \vdots \\ m^*_j \\ \vdots \\ m_J \end{pmatrix} \implies J_j = \begin{bmatrix} 1 & & & & & & & & & \\ & \ddots & & & & & & & & \\ & & \frac{\partial \theta^*_j}{\partial \theta_j} & & & & & \frac{\partial \theta^*_j}{\partial m_j} & & \\ & & & \ddots & & & & & & \\ & & & & 1 & & & & & \\ & & & & & 1 & & & & \\ & & & & & & \ddots & & & \\ & & \frac{\partial m^*_j}{\partial \theta_j} & & & & & \frac{\partial m^*_j}{\partial m_j} & & \\ & & & & & & & & \ddots & \\ & & & & & & & & & 1 \end{bmatrix}$$

Tale Jacobiano sarà diverso a seconda dei casi.

- Rifrazione:

$$\theta^*_j = \theta_j + sign(m_j) M_{jj}^{-1/2} \epsilon$$
$$m^*_j = m_j - sign(m_j)\sqrt{M_{jj}}\Delta U$$

$$\frac{\partial \theta^*_j}{\partial \theta_j} = 1 \qquad \frac{\partial \theta^*_j}{\partial m_j} = \delta_0(m_j) M_{jj}^{-1/2}\epsilon$$
$$\frac{\partial m^*_j}{\partial \theta_j} = sign(m_j)\sqrt{M_{jj}}\frac{\partial U}{\partial \theta_j} \qquad \frac{\partial m^*_j}{\partial m_j} = 1$$

  Siccome $\mathbb{P}(m_j = 0) = 0$ ne segue che $\frac{\partial \theta^*_j}{\partial m_j} = 0$ e J_j diventa una matrice triangolare con tutti 1 sulla diagonale, pertanto $\det(J_j) = 1$. Inoltre, nel caso di densità a posteriori a scalini per θ_j, anche $\frac{\partial U}{\partial \theta_j} = 0$, rendendo J_j a tutti gli effetti una matrice diagonale.

- Riflessione:

$$\theta^*_j = \theta_j$$
$$m^*_j = -m_j$$

$$\frac{\partial \theta_j^*}{\partial \theta_j} = 1 \qquad \frac{\partial \theta_j^*}{\partial m_j} = 0$$
$$\frac{\partial m_j^*}{\partial \theta_j} = 0 \quad \frac{\partial m_j^*}{\partial m_j} = -1$$

Il determinante in questo caso è -1, per cui non implica un'alterazione del volume da parte della trasformata ma solo dell'orientamento, il quale non è problematico per le leggi di probabilità perché si è soliti prenderne il valore assoluto.

Appendice B

Modelli del capitolo 4

B.1 Modello per l'imputazione di dati mancanti discreti

B.1.1 Modello completo

Le 2 verosimiglianze per i dati completi sono le seguenti:

- $\pi(N_1, \ldots, N_n | \alpha, \beta, C_1, \ldots, C_n) \propto \prod_{i=1}^{n} e^{-e^{\alpha+\beta C_i}} \cdot e^{N_i(\alpha+\beta C_i)}$;
- $\pi(C_1, \ldots, C_n | p) = p^{\sum C_i} \cdot (1-p)^{n-\sum C_i}$.

Le a priori invece sono:

- $\pi(\alpha) \propto e^{-\frac{\alpha^2}{2s_\alpha^2}}$;
- $\pi(\beta) \propto e^{-\frac{\beta^2}{2s_\beta^2}}$;
- $\pi(p) \propto p^{a-1} \cdot (1-p)^{b-1}$;

Il cambio di variabile per $\omega = \log \frac{p}{1-p}$ ha uno Jacobiano per la trasformata inversa pari a

$$\left(\frac{1}{1+e^{-\omega}}\right) \cdot \left(\frac{e^{-\omega}}{1+e^{-\omega}}\right)$$

che fa si che la densità per ω sia la seguente:

$$\pi(\omega) \propto \left(\frac{1}{1+e^{-\omega}}\right)^a \cdot \left(\frac{e^{-\omega}}{1+e^{-\omega}}\right)^b.$$

La trasformazione per C_j descritta in (3.4) che lo porta nell'aperto $(0, 2]$ è

$$\pi\left(C_j^*|p\right) = \frac{\pi(C_j = 0|p)}{1-0}\mathcal{I}(0 < C_j^* \leq 1) + \frac{\pi(C_j = 1|p)}{2-1}\mathcal{I}(1 < C_j^* \leq 2).$$

Successivamente, è possibile applicare la trasformazione logistica: $\tilde{C}_j^* = \log \frac{C_j^*}{2-C_j^*}$ che da luce alla seguente a priori dopo aver applicato la regola del cambio di variabile

$$\pi(\tilde{C}_j^*|\omega) = \left(\frac{1}{1+e^{-\tilde{C}_j^*}}\right)\cdot\left(\frac{e^{-\tilde{C}_j^*}}{1+e^{-\tilde{C}_j^*}}\right)\cdot\left(\frac{1}{1+e^{-\omega}}\right)^{\mathcal{I}(-\infty<\tilde{C}_j^*\leq 0)}\cdot\left(\frac{e^{-\omega}}{1+e^{-\omega}}\right)^{\mathcal{I}(0<\tilde{C}_j^*<\infty)}.$$

Definendo la trasformata inversa che lo riporta sullo spazio originale come

$$C_j = \left\lceil \frac{2}{1+e^{-\tilde{C}_j^*}} \right\rceil - 1,$$

è chiaro che l'apporto della trasformazione (3.4) è solo

$$\left(\frac{1}{1+e^{-\tilde{C}_j^*}}\right)\cdot\left(\frac{e^{-\tilde{C}_j^*}}{1+e^{-\tilde{C}_j^*}}\right),$$

mentre la restante parte

$$\left(\frac{1}{1+e^{-\omega}}\right)^{C_j}\cdot\left(\frac{e^{-\omega}}{1+e^{-\omega}}\right)^{1-C_j},$$

entra a far parte della verosimiglianza per i dati completi.

Portando tutto sulla scala logaritmica e cambiano di segno si ottengono le seguenti componenti che, se sommate, definiscono l'energia potenziale:

- $-\log\pi(N_1,\ldots,N_n|\alpha,\beta,C_1,\ldots,C_n) \propto e^{\alpha}(n_0+n_1e^{\beta}) - \sum N_i(\alpha+\beta C_i)$;
- $-\log\pi(\omega|C_1,\ldots,C_n) \propto \omega(n_0+b)+(n+a+b)\log(1+e^{-\omega})+\sum_{j\in J}\left[\tilde{C}_j^* + 2\log(1+e^{-\tilde{C}_j^*})\right]$;
- $-\log\pi(\alpha) \propto \frac{\alpha^2}{2s_\alpha^2}$;
- $-\log\pi(\beta) \propto \frac{\alpha^2}{2s_\beta^2}$.

dove $n_0 = n - n_1$ e $n_1 = \sum C_i$. Derivando l'energia potenziale, $U(\theta)$, rispetto a ciascun parametro continuo otteniamo:

- $\frac{\partial U}{\partial \alpha} = e^{\alpha}(n_0+n_1e^{\beta}) - \sum N_i + \frac{\alpha}{s_\alpha^2}$;
- $\frac{\partial U}{\partial \beta} = e^{\alpha}(n_1e^{\beta}) - \sum N_iC_i + \frac{\beta}{s_\beta^2}$;
- $\frac{\partial U}{\partial \omega} = (n_0+b) - (n+a+b)\frac{e^{-\omega}}{1+e^{-\omega}}$.

B.1.2 Modello ridotto

Seguendo l'approccio presente in McElreath (2018, capitolo 15, sezione 3), è possibile pensare a tre verosimiglianze per i dati, scrivibili in forma vettoriale[1] come:

- $N_{oss}|C_{oss} \sim Pois(e^{\alpha+\beta C_{oss}})$;
- $N_{lat}|p \sim pPois(e^{\alpha+\beta}) + (1-p)Pois(e^{\alpha})$;
- $C_{oss} \sim Bern(p)$;

dove N_{oss} è il vettore di note per le case in cui si conosce la presenza di gatti e N_{lat} invece è il vettore per le case in cui tale informazione è mancante (latente). Applicando la stessa trasformazione $\omega = \log \frac{p}{1-p}$ e utilizzando le stesse distribuzioni a priori, si ottiene che:

$$\begin{aligned}\pi(\theta|N, C_{oss}) \propto & \prod_{i:C_i \in C_{oss}} e^{-e^{\alpha+\beta C_i + N_i(\alpha+\beta C_i)}} \cdot \left(\frac{1}{1+e^{-\omega}}\right)^{C_i} \left(\frac{e^{-\omega}}{1+e^{-\omega}}\right)^{1-C_i} \\ & \prod_{i:C_i \notin C_{oss}} \left(\frac{1}{1+e^{-\omega}}\right)^{C_i} e^{N_i\alpha} \left(e^{-e^{\alpha+\beta}+N_i\beta} + e^{-\omega-e^{\alpha}}\right) \\ & e^{-\frac{\alpha^2}{2s_\alpha^2} - \frac{\beta^2}{2s_\beta^2}} \cdot \left(\frac{1}{1+e^{-\omega}}\right)^{a} \left(\frac{e^{-\omega}}{1+e^{-\omega}}\right)^{b}\end{aligned}$$

Siano $n_0 = \sum_{i:C_i \in C_{oss}} C_i$ e $n_1 = \sum_{i:C_i \notin C_{oss}} C_i$, da qui si può ottenere l'energia potenziale come

$$\begin{aligned}-\log \pi(\theta|N, C_{oss}) \propto & e^{\alpha}(n_0 + n_1 e^{\beta}) - \alpha \sum_{i=1}^{n} N_i - \beta \sum_{i:C_i \in C_{oss}} N_i C_i + \frac{\alpha^2}{2s_\alpha^2} + \frac{\beta^2}{2s_\beta^2} \\ & + (n+a+b)\log(1+e^{-\omega}) + (n_0+b)\omega \\ & - \sum_{i:C_i \notin C_{oss}} \text{LogSumExp}\left\{-e^{\alpha+\beta} + \beta N_i\,,\ -\omega - e^{\alpha}\right\}\end{aligned}$$

dove si è utilizzato l'operatore LogSumExp definito in (3.2). Il gradiente dell'energia potenziale è dato dalle seguenti derivate:

[1] Per cui non vanno intese come leggi congiunte ma come l'applicazione della marginale ad ogni elemento del vettore

$$
\begin{aligned}
\frac{\partial - \log \pi(\theta|N, C_{oss})}{\partial \alpha} &= e^{\alpha}(n_0 + n_1 e^{\beta}) - \sum_{i=1}^{n} N_i + \frac{\alpha}{s_\alpha^2} \\
&\quad - \sum_{i:C_i \notin C_{oss}} \frac{\left(e^{-e^{\alpha+\beta}+\beta N_i}\right)\left(-e^{\alpha+\beta}\right) + \left(e^{-\omega-e^{\alpha}}\right)}{e^{-e^{\alpha+\beta}+N_i\beta} + e^{-\omega-e^{\alpha}}} \\
\frac{\partial - \log \pi(\theta|N, C_{oss})}{\partial \beta} &= e^{\alpha}(n_1 e^{\beta}) - \sum_{i:i \in C_{oss}} C_i N_i + \frac{\beta}{s_\beta^2} \\
&\quad - \sum_{i:C_i \notin C_{oss}} \frac{\left(e^{-e^{\alpha+\beta}+\beta N_i}\right)\left(-e^{\alpha+\beta} + N_i\right)}{e^{-e^{\alpha+\beta}+N_i\beta} + e^{-\omega-e^{\alpha}}} \\
\frac{\partial - \log \pi(\theta|N, C_{oss})}{\partial \omega} &= n_0 + b - (n + a + b) \cdot \frac{e^{-\omega}}{1 + e^{-\omega}} \\
&\quad - \sum_{i:C_i \notin C_{oss}} \frac{\left(e^{-\omega-e^{\alpha}}\right)(-\omega)}{e^{-e^{\alpha+\beta}+N_i\beta} + e^{-\omega-e^{\alpha}}}
\end{aligned}
$$

B.1.3 Gibbs sampler

Ridefinendo il modello della sezione 4.1 nel seguente modo:

$$
\begin{aligned}
N_i|\lambda, \gamma, C_i &\sim Pois(\lambda \gamma^{C_i}) \\
C_i &\sim Bern(p) \\
\lambda &\sim Gamma(a_1, b_1) \\
\gamma &\sim Gamma(a_2, b_2) \\
p &\sim Beta(a, b)
\end{aligned}
$$

La posteriori è scrivibile come

$$
\begin{aligned}
\pi(\theta|N, C_{oss}) &\propto e^{-\sum_i \lambda \gamma^{C_i}} \lambda^{\sum_i N_i} \gamma^{C_i N_i} \cdot p^{\sum_i C_i} \cdot (1-p)^{n - \sum_i C_i} \\
&\quad \lambda^{a_1 - 1} e^{-\lambda b_1} \cdot \gamma^{a_2 - 1} e^{-\gamma b_2} \cdot p^{a-1} \cdot (1-p)^{b-1} \\
&= \lambda^{\sum_i N_i + a_1 - 1} e^{-\lambda(n_0 + n_1 \gamma + b_1)} \cdot \gamma^{\sum_i C_i N_i + a_1 - 1} e^{-\gamma(b_2)} \\
&\quad p^{n_1 + a - 1} \cdot (1-p)^{n_0 + b - 1},
\end{aligned}
$$

dove $n_1 = \sum_{i=1}^{n} C_i$ e $n_0 = n - n_1$, da cui è facile ricondursi alle *full conditional*:

- $\lambda|\gamma, p, C, N \sim Gamma(\sum_i N_i + a_1, n_0 + n_1\gamma + b_1)$;

- $\gamma|\lambda, p, C, N \sim Gamma(\sum_i C_i N_i + a_2, n_1\lambda + b_2)$;
- $p|\lambda, \gamma, C, N \sim Beta(n_1 + a, n_0 + b)$;
- $1 - \mathbb{P}(C_j = 0|p, \lambda, \gamma, C_{-j}, N) = \mathbb{P}(C_j = 1|p, \lambda, \gamma, C_{-j}, N) = \frac{p \cdot \gamma^{N_j} e^{-\lambda\gamma}}{p \cdot \gamma^{N_j} e^{-\lambda\gamma} + (1-p) \cdot e^{-\lambda}}$ per ogni $j : C_j \notin C_{oss}$.

B.2 Modello per punti di cambio

B.2.1 Modello completo

La verosimiglianza del modello è data da

$$\pi(Y_1, \ldots, Y_T|\lambda_1, \lambda_2, \lambda_3, t_1, t_2) = \prod_{i=1}^{t_1} \frac{\lambda_1^{y_i}}{y_i!} e^{-\lambda_1} \cdot \prod_{i=t_1+1}^{t_2} \frac{\lambda_2^{y_i}}{y_i!} e^{-\lambda_2} \cdot \prod_{i=t_2+1}^{T} \frac{\lambda_3^{y_i}}{y_i!} e^{-\lambda_3}$$

Le a priori nella parametrizzazione originale sono le seguenti:

- $\pi(\lambda_j) \propto \lambda_j^{a_j - 1} e^{-b_j \lambda_j} \quad , \quad j = 1, 2, 3;$
- $\mathbb{P}(t_1 = t) = \frac{1}{t_{fine} - t_{inizio}} \mathcal{I}(t_{inizio} \leq t < t_{fine})$;
- $\mathbb{P}(t_2 = t) = \frac{1}{t_{fine} - t_1} \mathcal{I}(t_1 < t \leq t_{fine})$.

I parametri continui possono essere mappati in $\mathbb{R}^3$ attraverso la trasformazione logistica ottenendo la seguente nuova distribuzione a priori per $\omega_j = \log \lambda_j$:

$$\pi(\omega_j) \propto e^{a_j \omega_j} e^{-b_j e^{\omega_j}} \quad , \quad j = 1, 2, 3.$$

Le a priori sui parametri discreti possono essere rese continue seguendo l'approccio descritto in (3.4) ottenendo:

- $t_1^* = \sum_{t=t_{inizio}}^{t_{fine}-1} \frac{1}{t_{fine} - t_{inizio}} \mathcal{I}(t < t_1^* \leq t + 1) \quad \stackrel{d}{=} \quad Unif(\,(t_{inizio}, t_{fine}]\,)$;
- $t_2^* = \sum_{t=t_1+1}^{t_{fine}} \frac{1}{t_{fine} - t_1} \mathcal{I}(t < t_2^* \leq t + 1) \quad \stackrel{d}{=} \quad Unif(\,(t_1 + 1, t_{fine} + 1]\,)$.

Successivamente, è possibile applicare la trasformazione logistica per mappare i parametri dal prodotto dei rispettivi supporti aperti limitati in $\mathbb{R}^2$:

- $\tilde{t}_1^* = \log\left(\frac{t_1^* - t_{inizio}}{t_{fine} - t_1^*}\right) \quad \Longrightarrow \quad t_1 = \left\lceil t_{inizio} + \frac{t_{fine} - t_{inizio}}{1 + e^{-\tilde{t}_1^*}} \right\rceil - 1;$
- $\tilde{t}_2^* = \log\left(\frac{t_1^* - t_1 - 1}{t_{fine} + 1 - t_2^*}\right) \quad \Longrightarrow \quad t_2 = \left\lceil t_1 + 1 + \frac{t_{fine} - t_1}{1 + e^{-\tilde{t}_2^*}} \right\rceil - 1.$

Per costruzione lo Jacobiano della mappa $\tilde{t}_1^*, \tilde{t}_2^* \to t_1^*, t_2^*$ è diagonale inferiore, pertanto il suo determinante sarà sempre pari al prodotto dei suoi elementi diagonali

$$\frac{\partial t_1^*}{\partial \tilde{t}_1^*} = (t_{fine} - t_{inizio}) \cdot \left(\frac{1}{1 + e^{-\tilde{t}_1^*}}\right) \cdot \left(\frac{e^{-\tilde{t}_1^*}}{1 + e^{-\tilde{t}_1^*}}\right)$$
$$\frac{\partial t_2^*}{\partial \tilde{t}_2^*} = (t_{fine} - t_1) \cdot \left(\frac{1}{1 + e^{-\tilde{t}_2^*}}\right) \cdot \left(\frac{e^{-\tilde{t}_2^*}}{1 + e^{-\tilde{t}_2^*}}\right)$$

È banale dimostrare che anche i nuovi parametri sono a priori indipendenti e le densità marginali sono scrivibili come

- $\pi(\tilde{t}_1^*) = \left(\frac{1}{1+e^{-\tilde{t}_1^*}}\right) \cdot \left(\frac{e^{-\tilde{t}_1^*}}{1+e^{-\tilde{t}_1^*}}\right)$
- $\pi(\tilde{t}_2^*) = \left(\frac{1}{1+e^{-\tilde{t}_2^*}}\right) \cdot \left(\frac{e^{-\tilde{t}_2^*}}{1+e^{-\tilde{t}_2^*}}\right)$

Applicando la trasformata logaritmica e cambiando di segno alle leggi di probabilità appena elencate si ottiene l'energia potenziale per il modello per punti di cambio

$$\begin{aligned} U(\theta) \propto & -\omega_1 \left(a_1 + \sum_{i=1}^{t_1} Y_i\right) + (t_1 + b_1)e^{\omega_1} \\ & -\omega_1 \left(a_2 + \sum_{i=t_1+1}^{t_2} Y_i\right) + (t_2 - t_1 + b_2)e^{\omega_2} \\ & -\omega_1 \left(a_3 + \sum_{i=t_2+1}^{T} Y_i\right) + (T - t_1 + b_3)e^{\omega_3} \\ & + (\tilde{t}_1^* + \tilde{t}_2^*) + 2\log\left(1 + e^{-\tilde{t}_1^*}\right) + 2\log\left(1 + e^{-\tilde{t}_2^*}\right). \end{aligned}$$

Il gradiente di quest'energia potenziale rispetto alle componenti continue è composto dalle seguenti derivate:

- $\frac{\partial U}{\partial \omega_1} = (t_1 + b_1)e^{\omega_1} - \left(a_1 + \sum_{i=1}^{t_1} Y_i\right);$
- $\frac{\partial U}{\partial \omega_2} = (t_2 - t_1 + b_2)e^{\omega_2} - \left(a_2 + \sum_{i=t_1+1}^{t_2} Y_i\right);$
- $\frac{\partial U}{\partial \omega_3} = (T - t_1 + b_3)e^{\omega_3} - \left(a_3 + \sum_{i=t_2+1}^{T} Y_i\right).$

B.2.2 Modello ridotto

In questo caso la distribuzione a posteriori marginale per $\theta_I = (\omega_1, \omega_2, \omega_3)$ è proporzionale a

$$\prod_{j=1}^{3} e^{a_j\omega_j - b_j e^{\omega_j}} \cdot \sum_{t_1=t_{inizio}}^{t_{fine}-1} \frac{1}{t_{fine}-t_{inizio}} \sum_{t_2=t_1+1}^{t_{fine}} \frac{1}{t_{fine}-t_1} \pi(Y_1, \ldots, Y_T | \omega_1, \omega_2, \omega_3, t_1, t_2),$$

dove $\pi(Y_1, \ldots, Y_T | \omega_1, \omega_2, \omega_3, t_1, t_2)$ è la verosimiglianza del modello completo definita nel paragrafo precedente. Passando alla scala logaritmica e cambiano di segno si ottiene:

$$-\log \pi(\theta_I|Y) \propto \sum_{j=1}^{3}[-\omega_j a_j + b_j e^{\omega_j}] - \underset{t_{inizio} \le t_1 < t_2 \le t_{fine}}{\text{LogSumExp}} \begin{bmatrix} -\log(t_{fine}-t_{inizio}) - \log(t_{fine}-t_1) \\ -t_1 e^{\omega_1} + \omega_1 \sum_{i=1}^{t_1} y_i \\ -(t_2-t_1)e^{\omega_2} + \omega_2 \sum_{i=t_1+1}^{t_2} y_i \\ -(T-t_2)e^{\omega_3} + \omega_3 \sum_{i=t_2+1}^{T} y_i \end{bmatrix}$$

Definendo

$$A_{t_1,t_2} = \frac{\exp\left\{-t_1 e^{\omega_1} + \omega_1 \sum_{i=1}^{t_1} y_i - (t_2-t_1)e^{\omega_2} + \omega_2 \sum_{i=t_1+1}^{t_2} y_i - (T-t_2)e^{\omega_3} + \omega_3 \sum_{i=t_2+1}^{T} y_i\right\}}{t_{fine}-t_1},$$

il gradiente del negativo del logaritmo della distribuzione a posteriori è fornito dalle seguenti derivate:

- $\frac{\partial -\log \pi(\theta_I|y)}{\partial \omega_1} = -a_1 + b_1 e^{\omega_1} - \frac{\sum_{t_{inizio} \le t_1 < t_2 \le t_{fine}} A_{t_1,t_2} \cdot \left(\sum_{i=1}^{t_1} y_i - t_1 e^{\omega_1}\right)}{\sum_{t_{inizio} \le t_1 < t_2 \le t_{fine}} A_{t_1,t_2}}$
- $\frac{\partial -\log \pi(\theta_I|y)}{\partial \omega_2} = -a_2 + b_2 e^{\omega_2} - \frac{\sum_{t_{inizio} \le t_1 < t_2 \le t_{fine}} A_{t_1,t_2} \cdot \left(\sum_{i=t_1+1}^{t_2} y_i - (t_2-t_1) e^{\omega_2}\right)}{\sum_{t_{inizio} \le t_1 < t_2 \le t_{fine}} A_{t_1,t_2}}$
- $\frac{\partial -\log \pi(\theta_I|y)}{\partial \omega_3} = -a_3 + b_3 e^{\omega_3} - \frac{\sum_{t_{inizio} \le t_1 < t_2 \le t_{fine}} A_{t_1,t_2} \cdot \left(\sum_{i=t_2+1}^{T} y_i - (T-t_2) e^{\omega_3}\right)}{\sum_{t_{inizio} \le t_1 < t_2 \le t_{fine}} A_{t_1,t_2}}$

La frazione a destra di ciascuna di queste, per questioni di stabilità numerica, necessita di essere trattata con l'operatore LogSumExp per il denominatore e con l'Algoritmo 1 per il numeratore.

B.2.3 Gibbs sampler

Riscrivendo il numeratore della distribuzione a posteriori definito all'inizio del paragrafo B.2.1, si ottine che

$$\pi(\theta|y) \propto \lambda_1^{\sum_{i=1}^{t_1} y_i + a_1 - 1} e^{-\lambda_1(t_1+b_1)} \cdot \lambda_2^{\sum_{i=t_1+1}^{t_2} y_i + a_2 - 1} e^{-\lambda_2(t_2-t_1+b_2)} \lambda_3^{\sum_{i=t_2+1}^{T} y_i + a_3 - 1} e^{-\lambda_3(T-t_2+b_3)} \cdot \frac{1}{t_{fine} - t_1},$$

per cui è possibile ottenere in forma esplicita tutte le *full conditional* come segue

- $\lambda_1|\lambda_2, \lambda_3, t_1, t_2, y_1, \ldots, y_T \sim Gamma\left(\sum_{i=1}^{t_1} y_i + a_1\ ,\ t_1 + b_1\right)$
- $\lambda_2|\lambda_1, \lambda_3, t_1, t_2, y_1, \ldots, y_T \sim Gamma\left(\sum_{i=t_1+1}^{t_2} y_i + a_2\ ,\ t_2 - t_1 + b_2\right)$
- $\lambda_3|\lambda_1, \lambda_2, t_1, t_2, y_1, \ldots, y_T \sim Gamma\left(\sum_{i=t_2+1}^{T} y_i + a_3\ ,\ T - t_2 + b_3\right)$
- $\mathbb{P}(t_1 = h, t_2 = k|\lambda_1, \lambda_2, \lambda_3, y_1, \ldots, y_T) =$

$$\frac{\mathcal{I}(t_{inizio} \leq h < k \leq t_{fine}) \cdot \left[\lambda_1^{\sum_{i=1}^{h} y_i + a_1 - 1} \lambda_2^{\sum_{i=h+1}^{k} y_i + a_2 - 1} \lambda_3^{\sum_{i=k+1}^{T} y_i + a_3 - 1} \cdot e^{-\lambda_1(h+b_1) - \lambda_2(k-h+b_2) - \lambda_3(T-k+b_3)} \cdot \frac{1}{t_{fine} - h}\right]}{\sum_{t_{inizio} \leq t_1 < t_2 \leq t_{fine}} \left[\lambda_1^{\sum_{i=1}^{t_1} y_i + a_1 - 1} \lambda_2^{\sum_{i=t_1+1}^{t_2} y_i + a_2 - 1} \lambda_3^{\sum_{i=t_2+1}^{T} y_i + a_3 - 1} \cdot e^{-\lambda_1(t_1+b_1) - \lambda_2(t_2-t_1+b_2) - \lambda_3(T-t_2+b_3)} \cdot \frac{1}{t_{fine} - t_1}\right]}$$

Quest'ultima legge di probabilità, per essere calcolata in maniera stabile numericamente, deve essere fatta applicando l'operatore LogSumExp al denominatore.

Appendice C

Codice R

C.1 Grafici delle traiettorie

```
### TRAIETTORIE

#funzione per il leapfrog
leapfrog <- function(theta,m,eps,nlp,args, type = 1){

  if(type == 1){
    #hmc normale, caso matrie di massa diagonale

    #aggiorna il momento
    m <- m - 0.5*eps*nlp(theta,args,FALSE)

    #aggiorna la posizione
    theta <- theta + args$M_inv %*% m

    #aggiorna il momento
    m <- m - 0.5*eps*nlp(theta,args,FALSE)

  }else if(type == 2){
    #dhmc con entrambe le componenti discontinue

    #dimensionalita'
    d <- length(theta)

    #potenziale
    U <- nlp(theta,args,TRUE)

    for(j in sample(d)){
      #vecchio valore
      theta_old <- theta[j]

      #valore aggiornato
      theta[j] <- theta[j] + sign(m[j]) * sqrt(args$M_inv[j,j]) * eps

      #differenza potenziale
      delta_U <- nlp(theta,args,TRUE) - U

      #riflessione o rifrazione?
      if(sqrt(args$M_inv[j,j])*abs(m[j]) > delta_U ){

        #rifrazione
        m[j] <- m[j] - sign(m[j])*1/sqrt(args$M_inv[j,j]) * delta_U
        U <- delta_U + U
```

```
      }else{

        #riflessione

        theta[j] <- theta_old

        #inverti il momento
        m[j] <- -m[j]
      }

    }

  }else{
    #versione mista

    #aggiorna il momento continuo
    m[1] <- m[1] - 0.5*eps*nlp(theta,args,FALSE)[1]

    #aggiorna la posizione continuo
    theta[1] <- theta[1] + 0.5*args$M_inv[1,1] * m[1]

    #potenziale
    U <- nlp(theta,args,TRUE)

    #vecchio valore
    theta_old <- theta[2]

    #valore aggiornato
    theta[2] <- theta[2] + sign(m[2]) * sqrt(args$M_inv[2,2]) * eps

    #differenza potenziale
    delta_U <- nlp(theta,args,TRUE) - U

    #riflessione o rifrazione?
    if(sqrt(args$M_inv[2,2])*abs(m[2]) > delta_U ){

      #rifrazione
      m[2] <- m[2] - sign(m[2])*1/sqrt(args$M_inv[2,2]) * delta_U
      U <- delta_U + U

    }else{

      #riflessione

      theta[2] <- theta_old

      #inverti il momento
      m[2] <- -m[2]
    }

    #aggiorna la posizione
    theta[1] <- theta[1] + 0.5*args$M_inv[1,1] * m[1]

    #aggiorna il momento
    m[1] <- m[1] - 0.5*eps*nlp(theta,args,FALSE)[1]

  }

  #resitutisci la lista
  list(theta = theta, m = m)
}

#calcola l'hamiltoniana del sistema
H <- function(theta,m,nlp,args,type){

  if(type == 1){
    #hamiltoniana gaussiana
    return(nlp(theta,args,TRUE) + 0.5 * sum(diag(args$M_inv) * m^2 ))

  }else if(type == 2){
```

```
    #hamiltoniana laplaciana
    return(nlp(theta,args,TRUE) + sum(diag(args$M_inv) * abs(m)))
  }else{
    #hamiltoniana mista
    return(nlp(theta,args,TRUE) + 0.5 * args$M_inv[1,1] * m[1]^2 +
      sum(args$M_inv[2,2] * abs(m)))
  }
}

#fai L passi leapfrog e salva le quantita' di interesse
hmc <- function(theta0,m0,eps,L,nlp,args, type = 1){

  #costruiamo la matrice contente tutte le traiettorie

  #dimensionalita'
  d <- length(theta0)

  #flusso della posizione
  theta_flow <- matrix(NA,L,d)

  #flusso del momento
  m_flow <- matrix(NA,L,d)

  #hamiltoniana
  H1 <- numeric(L)

  #viriale
  viriali <- numeric(L)

  #tasso di cambio empirico
  delta_viriali <- numeric(L)
  delta_viriali[1] <- 0

  #inizializzazione dei flussi
  theta_flow[1,] <- theta0
  m_flow[1,] <- m0

  #inizializzazione dell'hamiltoniana
  H1[1] <- H(theta0,m0,nlp,args,type)

  #inizializzazione del viriale
  viriali[1] <- drop(crossprod(theta0,m0))

  #pesi multinomiali
  pesi_mltn <- matrix(0,L,L)
  pesi_mltn[1,1] <- 1

  #integrazione traiettoria
  for(i in 2:L){
    m <- m_flow[i-1,]
    theta <- theta_flow[i-1,]

    #passo leapfrog
    passo <- leapfrog(theta,m,eps,nlp,args,type)
    theta <- passo$theta
    m <- passo$m

    #nuovo valore dell'Hamiltoniana
    H1[i] <- H(theta,m,nlp,args,type)

    #viriale
    viriali[i] <- drop(crossprod(theta,m))

    #delta viriale
    delta_viriali[i] <- (viriali[i] - viriali[i-1])/eps

    #pesi multinomiali
    pesi_mltn[1:i,i] <- exp(-H1[1:i] - matrixStats::logSumExp(-H1[1:i]))

    #aggiorna i flussi
```

```
    theta_flow[i,] <- theta
    m_flow[i,] <- m
  }

  #calcoliamo le quantita' di interesse

  #cumula i momenti
  if(type == 1){
    #caso gaussiano
    rho <- t(args$M_inv %*% t(apply(m_flow,2,cumsum)))
  }else if(type == 2){
    #caso laplaciano
    rho <- t(args$M_inv %*% t(apply(sign(m_flow),2,cumsum)))
  }else{
    #caso misto
    rho <- t(args$M_inv %*% t(cbind(cumsum(m_flow[,1]),cumsum(sign(m_flow[,2])))))
  }

  #costruiamo i vari criteri di terminazione
  nuts <- numeric(L)
  exhaustion <- numeric(L)

  for(i in 1:L){
    #ci interessa una sola direzione

    #nuts
    if(type == 1){
      nuts[i] <- crossprod(rho[i,],m_flow[i,])
    }else if(type == 2){
      nuts[i] <- crossprod(rho[i,],sign(m_flow[i,]))
    }else{
      nuts[i] <- crossprod(rho[i,],c(m_flow[i,1],sign(m_flow[i,2])))
    }

    #esauriemento approssimato
    exhaustion[i] <- abs(sum(delta_viriali[1:i]*pesi_mltn[1:i,i])) / i

  }

  #restituisci una lista
  list(theta = theta_flow,
       m = m_flow,
       rho = rho,
       viriali = viriali,
       delta_viriali = delta_viriali,
       H1 = H1,
       nuts = nuts,
       exhaustion = exhaustion)
}

grafico <- function(nlp,args,type = 1,tau = 0.05, eps = 0.01,
                    L = 500, seed = NULL, sigma_q = 0.3,
                    sigma_p = 0.1, contour = TRUE, center = TRUE){

  #salva l'aspetto corrente della finestra grafica
  op <- par(no.readonly = TRUE)

  #calcola la griglia
  ll <- 101
  mu1 <- seq(-5,5,l = ll)
  mu2 <- seq(-5,5,l = ll)
  nlp_val <- matrix(apply(expand.grid(mu1,mu2),1,
                          nlp,args = args, eval_nlp = TRUE), ll,ll)

  #genera il momento
  set.seed(seed)
  if(center){
    theta0 <- rnorm(2,0,sigma_q)
  }else{
    theta0 <- rnorm(2,2,sigma_q)
```

```
}
m0 <- rnorm(2,0,sigma_p)

#calcola le statistiche della traiettoria
lstep <- hmc(theta0,m0,eps,L,nlp,args,type = type)

#calcola il primo istante di U-TURN
U_turn <- which(lstep$nuts < 0)[1]

#partizione della finestra grafica
layout(mat = matrix(c(1,1,2,2,1,1,2,2,3,4,5,6),3,4,byrow = TRUE))
par(mar = c(0.1,0.1,0.1,0.1))

#TRAIETTORIA COMPLETA
image(mu1,mu2,nlp_val, xaxt = "n", yaxt = "n")
box(which = "figure")
if(contour){
  contour(mu1,mu2,exp(-nlp_val - min(nlp_val)),
          levels = c(0.05,0.1,0.3,0.5,0.7,0.9,0.95),
          add = TRUE, col = "lightgray", drawlabels = FALSE)
}

#punto di partenza
points(lstep$theta[1,1],lstep$theta[1,2],
       col = "blue", cex = 3, pch = "*")

#traiettoria
for(i in 1:L){
  text(lstep$theta[i,1],lstep$theta[i,2],
       labels = paste(i), col = 1, cex = .6)
}

#TRAIETTORIA TRONCATA
image(mu1,mu2,nlp_val, xaxt = "n", yaxt = "n")
box(which = "figure")
if(contour){
  contour(mu1,mu2,exp(-nlp_val - min(nlp_val)),
          levels = c(0.05,0.1,0.3,0.5,0.7,0.9,0.95),
          add = TRUE, col = "lightgray", drawlabels = FALSE)
}

#punto di partenza
points(lstep$theta[1,1],lstep$theta[1,2],
       col = "blue", cex = 3, pch = "*")

#traiettoria
for(i in 1:U_turn){
  text(lstep$theta[i,1],lstep$theta[i,2],
       labels = paste(i), col = 1, cex = .6)
}

points(lstep$theta[U_turn,1],lstep$theta[U_turn,2],
       col = "red", cex = 3, pch = "*")

#personalizza la finestra grafica
par(mar = c(0.1,0.1,2.1,0.1))

#HAMILTONIANA
plot(NULL,main = "Hamiltoniana",xaxt = "n", yaxt = "n",
     xlim = c(0,L), ylim = range(lstep$H1), bty = "n")
box(which = "figure")
lines(1:L,lstep$H1)

#VIRIALE
plot(NULL,main = "Viriale",xaxt = "n", yaxt = "n",
     xlim = c(0,L), ylim = range(lstep$viriali), bty = "n")
box(which = "figure")
lines(1:L,lstep$viriali)

#CRITERIO DI TERMINAZIONE NUTS
```

```
  plot(NULL,main = "Criterio NUTS",xaxt = "n", yaxt = "n",
       xlim = c(0,L), ylim = range(lstep$nuts), bty = "n")
  box(which = "figure")
  abline(h = 0, col = 2)
  lines(1:L,lstep$nuts)
  points(U_turn,lstep$nuts[U_turn],
         pch = "*", col = "red", cex = 3)

  #CRITERIO DI TERMINAZIONE BASATO SUL VIRIALE
  plot(NULL,main = "Criterio esaurimento viriale",
       xaxt = "n", yaxt = "n", xlim = c(1,L),
       ylim = range(lstep$exhaustion), bty = "n")
  box(which = "figure")
  lines(2:L,lstep$exhaustion[-1])
  text(round(L/2),tau + 0.2, bquote(tau == .(tau)))
  abline(h = tau, col = 2)

  #risettiamo la finsetra grafica
  par(op)

  invisible(lstep)
}


### ---------------------------------- gaussiana bivariata ---------------------------------- ###

#crea la lista da fornire in input alla funzione per l'energia potenziale
{
  rho <- 0.2
  d <- 2
  Sigma <- function(d, rho = 0.8){
    out <- matrix(rho,d,d)
    diag(out) <- 1
    out
  }
  arglist <- list(Prec = solve(Sigma(d,rho)), M_inv = diag(d))
}

#definisci la funzione per l'energia potenziale ed il suo gradiente
nlp <- function(par,args,eval_nlp = TRUE){
  if(eval_nlp){
    #solo la nlp
    return(0.5 * crossprod(par,crossprod(args$Prec,par)))
  }else{
    #solo il gradiente
    return(crossprod(args$Prec,par))
  }
}

#grafico leapfrog classico
grafico(nlp,arglist, type = 1, seed = 18, sigma_p = 0.25,
        L = 137, tau = 0.5, center = TRUE)

#grafico dhmc puro
grafico(nlp,arglist, type = 2, seed = 18, sigma_p = 5, eps = 0.1,
        L = 160, tau = 0.5, center = TRUE)

#grafico dhmc misto
grafico(nlp,arglist, type = 3, seed = 29, sigma_p = 4, eps = 0.31,
        L = 34, tau = 0.5, center = TRUE)
```

C.2 Studio di simulazione

```
# METODO SOLO PARAMETRI CONTINUI A PRIORI

#funzione che fa la log posteriori negativa e il suo gradiente
nlp_only_cont <- function(par,args,eval_nlp = TRUE){

  #distinguiamo i due casi
  if(eval_nlp){
    #restituisci solo la log posteriori negativa

    drop( (0.5 * args$n * t(par) %*% args$Prec - args$SUM_X_Prec) %*% par)

  }else{
    #restituisci solo il gradiente della log posteriori negativa

    args$Prec %*% (args$n * par) - t(args$SUM_X_Prec)
  }
}

# METODO CON SOLO PARAMETRI DISCONTINUI A PRIORI

#ridefiniamo la log posteriori negativa
nlp_only_disc <- function(par,args,eval_nlp = TRUE){

  #restituisci la log posteriori negativa

  #overflow
  if(any(abs(par) > 30)) return(Inf)

  #trasformiamo i parametri
  mu <- (ceiling(args$hyp[1] + (args$hyp[2] - args$hyp[1] + 1) * plogis(par)) - 1)/100
  drop( (0.5 * args$n * t(mu) %*% args$Prec - args$SUM_X_Prec) %*% mu) + #verosimiglianza
    sum(par + 2*log(1+exp(-par)))
}

# METODO MISTO

nlp_mixed <- function(par,args,eval_nlp = TRUE){

  #distinguiamo i due casi
  if(eval_nlp){
    #restituisci solo la log posteriori negativa

    #controlliamo l'overflow
    if(any(abs(par[args$idx_disc]) > 30)) return(Inf)

    #trasformiamo i parametri discreti
    mu <- par
    mu[args$idx_disc] <- (ceiling(args$hyp[1] + (args$hyp[2] - args$hyp[1] + 1) *
                                  plogis(par[args$idx_disc])) - 1)/100

    drop( (0.5 * args$n * t(mu) %*% args$Prec - args$SUM_X_Prec) %*% mu) +
      sum(par[args$idx_disc] + 2*log(1+exp(-par[args$idx_disc])))

  }else{
    #restituisci solo il gradiente della log posteriori negativa

    #controlliamo l'overflow
    if(any(abs(par[args$idx_disc]) > 30))
      return(rep(Inf,length(par) - args$idx_disc))

    #trasformiamo i parametri discreti
    mu <- par
    mu[args$idx_disc] <- (ceiling(args$hyp[1] + (args$hyp[2] - args$hyp[1] + 1) *
                                    plogis(par[args$idx_disc])) - 1)/100

    (args$Prec %*% (args$n * mu) - t(args$SUM_X_Prec))[-args$idx_disc]
  }
```

```
}

#FUNZIONE DI RICERCA DI TAU PER L'XHMC
ricerca_tau <- function(nlp,arglist,d,k,eps,seed,M_cont,M_disc,
                        N = 100,tau_start = 0.001,tau_end = 5.001,
                        grid_len = 10,maxit = 10){

  #crea la griglia
  tau_grid <- seq(tau_start,tau_end,l = grid_len)

  #inizializza la funzione obiettivo: ESS/L
  target <- numeric(grid_len)

  #continua fino a che non ottieni una funzione convessa
  out <- FALSE
  start <- 0
  it <- 0

  while( (!out) && it < maxit ){
    #sostituisci 1
    tau_grid[tau_grid == 1] <- 1.001

    #calcola per ciascun elemento della griglia il valore ottimo
    for(i in (start + seq_len(grid_len))){

      #fai una catena con XHMC per il valore corrente di tau
      tmp <- xdnuts(lapply(1, function(x) c(rep(1,d-k),
                                            rep(0.067,k))),
                    nlp = nlp,
                    args = arglist,
                    k = k,
                    parallel = FALSE,
                    hide = TRUE,
                    method = "XHMC",
                    N = N,
                    tau = tau_grid[i],
                    control = set_parameters(N_init1 = 0,
                                             N_adapt = 0,
                                             N_init2 = 0,
                                             M_type = "dense",
                                             M_cont = M_cont,
                                             M_disc = M_disc,
                                             l_eps_init = log(eps)))
      #salva ESS/L
      target[i] <- mean((purrr::quietly(summary))(tmp)$result$stats$ESS /
        sum(tmp$chains[[1]]$step_length))
    }

    #controlla che il massimo non sia nelle due estremita' della griglia
    idx_max <- which.max(target)

    if(idx_max == (start + 1)){
      #ingrandisci la griglia a sinistra
      start <- 0
      target <- c(rep(NA,grid_len),target)
      tau_grid <- c(seq(0,tau_grid[1], l = grid_len+2)[1 + c(1:grid_len) ], tau_grid)

    }else if(idx_max == (start + grid_len)){
      #ingrandisci la griglia a destra
      start <- length(target)
      target <- c(target,rep(0,grid_len))
      tau_grid <- c(tau_grid,tau_grid[length(tau_grid)] + seq(tau_start,tau_end,l = grid_len))

    }else{
      #esci
      out <- TRUE
    }

    it <- it + 1
```

```
  }

  #finita la prima ricerca piu' grossolana, svolgere una ricerca piu' fine
  #in un intorno del valore trovato

  #ingrandisci il vettore con la funzione obiettivo da massimizzare
  target <- c(target[1:(idx_max - 1)],
              rep(NA,floor(grid_len/2)),
              target[idx_max],
              rep(NA,floor(grid_len/2)),
              target[1 + idx_max:( length(target) - 1)])

  #ridefinisci i valori di inizio,fine e corrente della griglia
  tau_start <- tau_grid[idx_max-1]
  tau_end <- tau_grid[idx_max+1]
  tau_current <- tau_grid[idx_max]

  #ingrandisci la griglia
  tau_grid <- c(tau_grid[1:(idx_max - 1)],
                rep(NA,floor(grid_len/2)),
                tau_grid[idx_max],
                rep(NA,floor(grid_len/2)),
                tau_grid[1 + idx_max:( length(tau_grid) - 1)])

  #calcola i valori della nuova griglia
  tau_grid[idx_max + 0:floor(grid_len/2 - 1)] <-
    seq(tau_start,tau_current, l = floor(grid_len/2)+2)[1+(1:floor(grid_len/2))]
  tau_grid[idx_max + floor(grid_len/2) + 1:floor(grid_len/2)] <-
    seq(tau_current,tau_end,l = floor(grid_len/2) + 2)[1 + (1:floor(grid_len/2))]

  #calcola gli indici di questi nuovi valori su cui ciclare
  ii <- c(idx_max + 0:floor(grid_len/2 - 1),idx_max + floor(grid_len/2) + 1:floor(grid_len/2))

  #ricerca piu' fine
  for(i in ii){
    #fai una catena con XHMC per il valore corrente di tau
    tmp <- xdnuts(lapply(1, function(x) c(rep(1,d-k),
                                          rep(0.067,k))),
                  nlp = nlp,
                  args = arglist,
                  k = k,
                  parallel = FALSE,
                  hide = TRUE,
                  method = "XHMC",
                  N = N,
                  tau = tau_grid[i],
                  control = set_parameters(N_init1 = 0,
                                           N_adapt = 0,
                                           N_init2 = 0,
                                           M_type = "dense",
                                           M_cont = M_cont,
                                           M_disc = M_disc,
                                           l_eps_init = log(eps)))
    #salva ESS/L
    target[i] <- mean((purrr::quietly(summary))(tmp)$result$stats$ESS /
      sum(tmp$chains[[1]]$step_length))
  }

  #calcola il valore maxinmo
  idx_max <- which.max(target)

  #restituisci il tutto
  list(tau = tau_grid[idx_max],
       tau_grid = tau_grid,
       target = target,
       n_it = it)
}

#FUNZIONE CHE CALCOLA LE STATISTICHE PER I QUATTRO METODI
stat_fun <- function(n = 1e3, d = 10, k = 0,rho = 0.2, seed1 = 123,
```

```
                  seed2 = 123,hyp = c(-3000,3000), eps = NULL,
                  delta = c(0.9,0.6),N_search = 100,
                  tau_start = 0.00001,tau_grid_len = 5,
                  tau_end = 5.00001, maxit = 15){
require(XDNUTS)

### SIMULAZIONE DEL DATASET

#fissa il seme
set.seed(seed1)

#fissa la matrice di varianza e covarianza
Sigma <- matrix(rho,d,d)
diag(Sigma) <- 1

#fissa i veri valori dei parametri
mu <- rep(1,d)

#simula i dati
X <- mvtnorm::rmvnorm(n,mu,Sigma)

#indici da sostituire
idx_disc <- setdiff(seq_len(d),seq_len(d-k))

#creiamo la lista da fornire in input
arglist <- list(n = n, #numerosita' campionaria
                SUM_X_Prec = apply(X,2,sum) %*% solve(Sigma), #stat suff,
                Prec = solve(Sigma), #stat suff
                hyp = hyp, #intervallo di ricerca per mu nel caso discontinuo
                idx_disc = idx_disc
               )

#inizializza la matrice con gli ESS
ESS <- matrix(NA,d,4)
rownames(ESS) <- paste0("theta",seq_len(d))
colnames(ESS) <- c("NUTS","HMC1","XHMC","HMC2")

#seleziona la funzione da utilizzare
nlp <- ifelse(k == 0,nlp_only_cont,ifelse(k == d,nlp_only_disc, nlp_mixed) )

#fissa i gradi di liberta' degli algoritmi
idx_disc <- setdiff(seq_len(d),seq_len(d-k))
idx_cont <- setdiff(seq_len(d),idx_disc)
M_cont <- Sigma[idx_cont,idx_cont]
M_disc <- diag(Sigma)[idx_disc]*10

#se eps e' NULL lo stimiamo via HMC
if(is.null(eps)){
  tmp <- xdnuts(lapply(1, function(x) c(rep(1,d-k),
                                        rep(0.067,k))),
                nlp = nlp,
                args = arglist,
                k = k,
                parallel = FALSE,
                hide = TRUE,
                method = "HMC",
                L = 100*rho,
                control = set_parameters(delta = delta,
                                         N_init1 = 50,
                                         N_adapt = 0,
                                         N_init2 = 0,
                                         M_type = "dense",
                                         M_cont = M_cont,
                                         M_disc = M_disc))
  eps <- mean(tmp$chains[[1]]$step_size)
}

#esegui il NUTS
set.seed(seed2)
NUTS <- xdnuts(lapply(1, function(x) c(rep(1,d-k),
```

```
                                    rep(0.067,k))),
              nlp = nlp,
              args = arglist,
              k = k,
              parallel = FALSE,
              hide = TRUE,
              method = "NUTS",
              control = set_parameters(N_init1 = 0,
                                       N_adapt = 0,
                                       N_init2 = 0,
                                       M_type = "dense",
                                       M_cont = M_cont,
                                       M_disc = M_disc,
                                       l_eps_init = log(eps)))

#salva gli ESS per il NUTS
ESS[,1] <- (purrr::quietly(summary))(NUTS)$result$stats$ESS

#calcola il numero di integrazioni fatte per il NUTS
LL_NUTS <- sum(NUTS$chains[[1]]$step_length)

#trova il valore per l'HMC arrotondato per eccesso
L_NUTS <- ceiling(LL_NUTS / 1000)

#esegui l'HMC da confrontare al NUTS
set.seed(seed2)
HMC_NUTS <- xdnuts(lapply(1, function(x) c(rep(1,d-k),
                                   rep(0.067,k))),
             nlp = nlp,
             args = arglist,
             k = k,
             parallel = FALSE,
             hide = TRUE,
             method = "HMC",
             L = L_NUTS,
             control = set_parameters(N_init1 = 0,
                                      N_adapt = 0,
                                      N_init2 = 0,
                                      M_type = "dense",
                                      M_cont = M_cont,
                                      M_disc = M_disc,
                                      l_eps_init = log(eps),
                                      L_jitter = 0))

#salva gli ESS per il primo HMC
ESS[,2] <- (purrr::quietly(summary))(HMC_NUTS)$result$stats$ESS

#ricerca il valore di tau ottimo per l'XHMC
taus <- ricerca_tau(nlp,arglist,d,k,eps,seed2,M_cont,M_disc,
                            N_search,tau_start,tau_end,tau_grid_len,maxit)
#esegui l'XHMC
set.seed(seed2)
XHMC <- xdnuts(lapply(1, function(x) c(rep(1,d-k),
                                   rep(0.067,k))),
              nlp = nlp,
              args = arglist,
              k = k,
              parallel = FALSE,
              hide = TRUE,
              method = "XHMC",
              tau = taus$tau,
              control = set_parameters(N_init1 = 0,
                                       N_adapt = 0,
                                       N_init2 = 0,
                                       M_type = "dense",
                                       M_cont = M_cont,
                                       M_disc = M_disc,
                                       l_eps_init = log(eps)))

#salva gli ESS per l'XHMC
```

```
  ESS[,3] <- (purrr::quietly(summary))(XHMC)$result$stats$ESS

  #calcola il numero di integrazioni fatte per l'XHMC
  LL_XHMC <- sum(XHMC$chains[[1]]$step_length)

  #trova il valore per l'HMC arrotondato per eccesso
  L_XHMC <- ceiling(LL_XHMC / 1000)

  #esegui l'HMC da confrontare al XHMC
  set.seed(seed2)
  HMC_XHMC <- xdnuts(lapply(1, function(x) c(rep(1,d-k),
                                     rep(0.067,k))),
                     nlp = nlp,
                     args = arglist,
                     k = k,
                     parallel = FALSE,
                     hide = TRUE,
                     method = "HMC",
                     L = L_XHMC,
                     control = set_parameters(N_init1 = 0,
                                              N_adapt = 0,
                                              N_init2 = 0,
                                              M_type = "dense",
                                              M_cont = M_cont,
                                              M_disc = M_disc,
                                              l_eps_init = log(eps),
                                              L_jitter = 0))

  #salva gli ESS per il secondo HMC
  ESS[,4] <- (purrr::quietly(summary))(HMC_XHMC)$result$stats$ESS

  return(list(ESS = ESS, d = d, k = k, rho = rho,
              LL = c(NUTS = LL_NUTS, HMC1 = L_NUTS,
                     XHMC = LL_XHMC, HMC2 = L_XHMC),
              tau = taus,L = Ls, eps = eps))
}

#STUDIO DI SIMULAZIONE

#dimensionalita' parametrica
d_grid <- c(10,30,50)

#percentuale di componenti discontinue
k_ratio_grid <- c(0,0.4,0.8,1)

#correlazione tra i parametri
rho_grid <- c(0.2,0.6,0.8)

#griglia totale
config_grid <- expand.grid(rho_grid,k_ratio_grid,d_grid)[,3:1]

#nome di ciascun elemento della griglia
nomi <- apply(config_grid,1,function(x)
  paste0("d=",x[1],"k=",x[1]*x[2],"rho=",x[3]))

#numero di replicazioni per ciascuna configurazione
N_rep <- 30

#creazione della lista con i risultati
output <- vector("list",length(nomi))
names(output) <- nomi

for(i in seq_along(nomi)){

  #lista contenente le iterazioni della stessa configurazione
  tmp <- vector("list",N_rep)

  #ripetizioni della stessa configurazione
  for(j in seq_len(N_rep)){
    tmp[[j]] <- stat_fun(n = 1e3,
```

```
                        d = config_grid[i,1],
                        k = config_grid[i,1]*config_grid[i,2],
                        rho = config_grid[i,3],
                        seed1 = 123,
                        seed2 = j,
                        hyp = c(-3000,3000),
                        eps = NULL,
                        N_search = 100,
                        tau_start = 1e-5,
                        tau_end = 5.00001,
                        tau_grid_len = 5,
                        maxit = 10)
  }

  #salva i risultati nella lista finale
  output[[nomi[i]]] <- tmp

  #stampa in output
  cat(i," su ",length(nomi)," completati")

  #salva la lista fino ad adesso
  save(output, file = "simulazioni.RData")
}

### TEST DI NORMALITA'
p.values <- t(sapply(output, function(x){
  apply(sapply(x,function(y) colMeans(y$ESS)),1,
        function(z) shapiro.test(z)$p.value)
}))
mean(-log(unlist(p.values)) < -log(0.05/length(unlist(ESS))))
#rifiuto H0 il 70% delle volte

### GRAFICO DELLE DIFFERENZE
#estraimo i vari ESS
ESS <- lapply(output,function(x) {
  (sapply(x,function(y) colMeans(y$ESS)))
})

{
  par(mar = c(4.1,1.1,0.1,2.1), xpd = TRUE)
  x <- t(sapply(ESS,function(x)
    c(quantile(x[1,] - x[2,],c(0.1,0.5,0.9)),
      quantile(x[3,] - x[4,],c(0.1,0.5,0.9)) )))
  plot(range(x),c(1,36), bty = "n", type = "n",
       yaxt = "n", ylab = "", xlab = expression(bar(ESS)^C - bar(ESS)))
  lines(c(0,0),c(-0.3,36), lty = 2)
  for(i in 1:36){
    points(x[i,c(2,5)],c(i,i+0.4), col = c("red","blue"), pch = 16)
    lines(x[i,c(1,3)],c(i,i),col = "red")
    lines(x[i,c(4,6)],c(i,i)+0.4,col = "blue")
  }
  legend("bottomleft",legend = c("NUTS","XHMC"),col = c("red","blue"), pch = 16)
}

### TEST NON PARAMETRICI

#calcoliamo le statistiche W e il loro p.value
statistiche <- as.data.frame(t(sapply(ESS,function(x){
  c(
    #statistica t per il NUTS-HMC
    NUTS = unlist(c(wilcox.test(x[1,],x[2,],
      paired = TRUE, alternative = "greater" )[c("statistic","p.value")])),

    #statistica t per il XHMC-HMC
    XHMC = unlist(c(wilcox.test(x[3,],x[4,],
      paired = TRUE, alternative = "greater" )[c("statistic","p.value")]))
  )
})))

#estraiamo le configurazioni
```

```
statistiche$d <- as.numeric(lapply(strsplit(
  stringr::str_extract(rownames(statistiche),"d=[0-9][0-9]"),"="),"[",2))
statistiche$k <- as.numeric(lapply(strsplit(
  stringr::str_extract(rownames(statistiche),"k=[0-9]*"),"="),"[",2))
statistiche$rho <- as.numeric(lapply(strsplit(
  stringr::str_extract(rownames(statistiche),"rho=[0-9]\\.[0-9]"),"="),"[",2))

#grafico con le statistiche test
{
  #partizione della finestra grafica
  layout(matrix(1:3,1),widths = c(0.4,0.4,0.2))
  par(mar = c(4.1,2.1,3.1,0.1), xpd = FALSE)

  #grafico vuoto
  plot(range(statistiche[,c(1,3)]),c(0,40),
       xlab = "Wilcoxon W statistic",
       ylab = "", bty = "n", yaxt = "n", main = "NUTS",
       type = "n", cex.lab = 1.5, cex.main = 2)

  #quantili della distribuzione nulla
  abline(v = 30*31/4 + qnorm(c(0.025/1,0.5,1 - 0.025/1)) *
           sqrt(30*31*61/24), lty = 2, col = c(2,1,2))

  #valori delle statistiche per il NUTS
  points(statistiche[,1],1:36,
         cex = (statistiche$d/10)^(1/2),
         pch = round(statistiche$rho*4),
         col = round(statistiche$k/statistiche$d*3 + 1))

  #grafico vuoto
  plot(range(statistiche[,c(1,3)]),c(0,40),
       xlab = "Wilcoxon W statistic",
       ylab = "", bty = "n", yaxt = "n",
        main = "XHMC", type = "n", cex.lab = 1.5, cex.main = 2)

  #quantili della distribuzione nulla
  abline(v = 30*31/4 + qnorm(c(0.025/1,0.5,1 - 0.025/1)) *
           sqrt(30*31*61/24), lty = 2, col = c(2,1,2))

  #valori delle statistiche per XHMC
  points(statistiche[,3],1:36,
         cex = (statistiche$d/10)^(1/2),
         pch = round(statistiche$rho*4),
         col = round(statistiche$k/statistiche$d*3) + 1)

  #legenda
  plot.new()
  legend("top",title = expression(rho),
         legend = c(0.2,0.6,0.8), col = "darkgray",
         pch = unique(round(statistiche$rho*4)),
         cex = 2, bty = "n")
  legend("bottom",title = "k/d",
         legend = c(0,0.4,0.8,1),
         col = unique(round(statistiche$k/statistiche$d*3) + 1),
         pch = 16,cex = 2, bty = "n")
}

#grafico dei log p.value negativi
{
  #partiziona la finestra grafica
  layout(matrix(1:3,1),widths = c(0.4,0.4,0.2))
  par(mar = c(4.1,2.1,3.1,0.1), xpd = FALSE)

  #grafico vuoto
  plot(range(-log(statistiche[,c(2,4)])),c(0,40),
       xlab = "-log(p.value)", ylab = "", bty = "n",
       yaxt = "n", main = "NUTS", type = "n", cex.lab = 1.5, cex.main = 2)

  #valore critico della statistica test
  abline(v = -log(0.05/36), lty = 2)
```

```
  #valori statistiche test per il NUTS
  points(-log(statistiche[,2]),1:36,
         cex = (statistiche$d/10)^(1/2),
         pch = round(statistiche$rho*4),
         col = round(statistiche$k/statistiche$d*3 + 1))

  #grafico vuoto
  plot(range(-log(statistiche[,c(2,4)])),c(0,40),
       xlab = "-log(p.value)", ylab = "", bty = "n",
       yaxt = "n", main = "XHMC", type = "n",cex.lab = 1.5, cex.main = 2)

  #valore critico della statistica test
  abline(v = -log(0.05/36), lty = 2)

  #valori statistice test per XHMC
  points(-log(statistiche[,4]),1:36,
         cex = (statistiche$d/10)^(1/2),
         pch = round(statistiche$rho*4),
         col = round(statistiche$k/statistiche$d*3 + 1))

  #legenda
  plot.new()
  legend("top",title = expression(rho),
         legend = c(0.2,0.6,0.8),col = "darkgray",
         pch = unique(round(statistiche$rho*4)),
         cex = 2, bty = "n")
  legend("bottom",title = "k/d",
         legend = c(0,0.4,0.8,1),
         col = unique(round(statistiche$k/statistiche$d*3) + 1),
         pch = 16,cex = 2, bty = "n")
}

### REGRESSIONE SULLA MEDIANA

#creazione del data.frame
NUTS <- sapply(ESS,function(x) x[1,] - x[2,])
XHMC <- sapply(ESS,function(x) x[3,] - x[4,])

dati <- NULL
for(i in 1:36){
  dati <- rbind(dati,cbind(config_grid[i,1],
                           config_grid[i,2],
                           config_grid[i,3],
                           NUTS[,nomi[i]],
                           XHMC[,nomi[i]],
                           id = i))
}
dati <- data.frame(NUTS = dati[,4],
                   XHMC = dati[,5],
                   d = dati[,1],
                   k_ratio = dati[,2],
                   rho = dati[,3],
                   id = as.factor(dati[,6]))

library(quantreg)

#modello con sola intercetta
m0_NUTS <- rq(NUTS ~ 1, data = dati, tau = 0.5)
m0_XHMC <- rq(XHMC ~ 1, data = dati, tau = 0.5)

#modello additivo
m1_NUTS <- rq(NUTS ~ d + k_ratio + rho, data = dati, tau = 0.5)
m1_XHMC <- rq(NUTS ~ d + k_ratio + rho, data = dati, tau = 0.5)

#modello con interazioni
m2_NUTS <- rq(NUTS ~ d * k_ratio * rho, data = dati, tau = 0.5)
m2_XHMC <- rq(NUTS ~ d * k_ratio * rho, data = dati, tau = 0.5)

#modello con effetti marginali
```

```
m_d_NUTS <- rq(NUTS ~ d, data = dati, tau = 0.5)
m_d_XHMC <- rq(XHMC ~ d, data = dati, tau = 0.5)

m_k_ratio_NUTS <- rq(NUTS ~ k_ratio, data = dati, tau = 0.5)
m_k_ratio_XHMC <- rq(XHMC ~ k_ratio, data = dati, tau = 0.5)

m_rho_NUTS <- rq(NUTS ~ rho, data = dati, tau = 0.5)
m_rho_XHMC <- rq(XHMC ~ rho, data = dati, tau = 0.5)

#calcolo dell'indice R^1
R1 <- function(m1,m0){
  1 - sum(m1$residuals*(m1$tau - (m1$residuals < 0))) /
    sum(m0$residuals*(m0$tau - (m0$residuals < 0)))
}

R1(m1_NUTS,m0_NUTS);R1(m2_NUTS,m0_NUTS)
R1(m1_XHMC,m0_XHMC);R1(m2_XHMC,m0_XHMC)

#crea la tabella con l'output dei coefficienti
tabella <- matrix(NA,4,8)

tabella[,1:2] <- summary(m1_NUTS,se = "boot",
                         bsmethod = "xy", cluster = dati$id)$coef[,1:2]
tabella[2,3:4] <- summary(m_d_NUTS,se = "boot",
                          bsmethod = "xy", cluster = dati$id)$coef[2,1:2]
tabella[3,3:4] <- summary(m_k_ratio_NUTS,se = "boot",
                          bsmethod = "xy", cluster = dati$id)$coef[2,1:2]
tabella[4,3:4] <- summary(m_rho_NUTS,se = "boot",
                          bsmethod = "xy", cluster = dati$id)$coef[2,1:2]

tabella[,5:6] <- summary(m1_XHMC,se = "boot",
                         bsmethod = "xy", cluster = dati$id)$coef[,1:2]
tabella[2,7:8] <- summary(m_d_XHMC,se = "boot",
                          bsmethod = "xy", cluster = dati$id)$coef[2,1:2]
tabella[3,7:8] <- summary(m_k_ratio_XHMC,se = "boot",
                          bsmethod = "xy", cluster = dati$id)$coef[2,1:2]
tabella[4,7:8] <- summary(m_rho_XHMC,se = "boot",
                          bsmethod = "xy", cluster = dati$id)$coef[2,1:2]

round(tabella,digits = 3)
```

C.3 Modello per l'imputazione di dati mancanti discreti

```
#seed
set.seed(9)

#numero di casee
N_case <- 100L

#intercetta modello log lineare delle note cantate dai canarini
alpha <- 2

#effetto della presenza dei gatti
beta <- -1 #-1.7

#probabilita' che il gatto sia presente
k <- 0.5

#probabilita' di non sapere se il gatto e' presente o meno
r <- 0.2

#generazione di C
gatti_full <- gatti <- rbinom(N_case,1,k)

#generazione di N
note <- rpois(N_case, exp(alpha + beta*gatti))

#generazione valori mancanti
idx_na <- rbinom(N_case,1,r) == 1
gatti[idx_na] <- NA

#sistemazione dei dati
dati <- data.frame(x = gatti,y = note,idx = is.na(gatti))

#numero di dati mancanti
n_disc <- sum(is.na(dati$x)) #24

#1) MODELLO COMPLETO

#log posteriori negativa per il modello intero
nlp_completo <- function(par,args,eval_nlp = TRUE){

  if(eval_nlp){
    #teniamo conto dell'overflow
    if(any(abs(par) > 30)) return(Inf)

    #convertiamo i parametri discontinui nei valori mancanti
    x <- args$data$x
    x[args$data$idx] <- as.numeric(2*plogis(par[-(1:3)]) > 1)

    #numero di case senza gatti
    n0 <- sum(x == 0)

    #numero di case con gatti
    n1 <- sum(x == 1)

    #calcoliamo la log posteriori negativa
    return( exp(par[1])*(n0+n1*exp(par[2])) -
              sum( (par[1] + par[2]*x)*args$data$y) +
              sum(par[1:2]^2/args$hyp[1:2])/2 + #parte del modello poisson

              (n0 + n1 + args$hyp[3] + args$hyp[4])*log(1+exp(-par[3])) +
              (n0+args$hyp[4])*par[3] + #parte del modello bernoulliano

              #parte della priori per i dati mancanti
              sum(par[-(1:3)] + 2*log(1+exp(-par[-(1:3)]))) )

  }else{

    #teniamo conto dell'overflow
    if(any(abs(par) > 30)) return(rep(Inf,3))

    #convertiamo i parametri discontinui nei valori mancanti
    x <- args$data$x
    x[args$data$idx] <- as.numeric(2*plogis(par[-(1:3)]) > 1)

    #numerosita' campionaria
    n <- length(x)

    #numero di case senza gatti
    n0 <- sum(x == 0)
```

```
    #numero di case con gatti
    n1 <- sum(x == 1)

    #calcoliamo il gradiente dei parametri continui
    c(
      exp(par[1])*(n0+n1*exp(par[2])) -
        sum(args$data$y) + par[1]/args$hyp[1], #derivata per l'intercetta

      n1*exp(par[1] + par[2]) - sum(x*args$data$y) +
        par[2]/args$hyp[2], #derivata per il coefficiente angolare

      (n0 + args$hyp[4]) -
        (n0+n1+args$hyp[3]+args$hyp[4])*(1-plogis(par[3]))
      #derivata per la probabilita' in scala logit
    )
  }

}

#iperparametri
hyp0 <- c(10,10,1,1)

#lista da fornire in input
arglist_completo = list(data = dati, hyp = hyp0)

#carica il pacchetto XDNUTS
library(XDNUTS)

#carica il pacchetto per contare il tempo
library(tictoc)

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#MCMC
XHMC_completo <- xdnuts(theta0 = lapply(1:10,function(x) {
  out <- rnorm(3+n_disc,0,1)
  names(out) <- c("alpha","beta","logit_p",paste0("C*_tilde",1:n_disc))
  out
}),nlp = nlp_completo,
args = arglist_completo,
k = n_disc,
N = 1e3,
K = 3,
thin = 1,
method = "XHMC",
tau = 1.5,
parallel = TRUE,
control = set_parameters(delta = c(0.8,0.6),
                         different_stepsize = TRUE,
                         max_treedepth = 5,
                         burn_adapt_ratio = 0.5,
                         N_adapt = 500))

#smetti di contare il tempo
tempo_XHMC_completo <- toc()

#trasformazione del parametro omega in p
b <- xdtransform(XHMC_completo,which = 3,plogis,new.names = "p")

#grafico dei tassi di accettazione/riflessione
plot(b, type = 7)

#grafico delle catene
plot(b, cex.legend = 1.2)

#grafico della catena dell'energia
plot(b, type = 3, cex.legend = 1.5)

#grafico delle frequenze dei passi di integrazione
plot(b, type = 4, cex.legend = 1.3, cex.titles = 1.5)

#grafici delle densita'
plot(b, type = 2, which = 1:3, cex.titles = 1.2)
plot(b, type = 2, which = c(8,9,10), cex.titles = 1.2)

#grafico BFMI
plot(b,type = 5, cex.legend = 2)

#grafico autocorrelazioni
plot(b,type = 6, cex.legend = 1.2, cex.titles = 2.5)

#summary
```

```r
summary(b)

### salvataggio delle quantita' necessarie per il confronto tra
### i diversi metodi
theta_start <- apply(xdextract(XHMC_completo,collapse = TRUE),2,mean)
mu <- theta_start[1:3]
Sigma <- cov(xdextract(XHMC_completo,
                       which = "continuous", collapse = TRUE))

#step-size
eps_completo <- mean(sapply(XHMC_completo$chains,
                            function(x) mean(x$step_size)))

#matrice di massa per le componenti continue
M_cont <- lapply(XHMC_completo$chains,function(x) x$M_cont)
M_cont <- Reduce("+",M_cont) / length(M_cont)

#matrice di massa per la componenti discontinue
M_disc <- rowMeans(do.call(cbind,lapply(
  XHMC_completo$chains,function(x) x$M_disc)))
```

C.3.1 Confronti con altri algoritmi

```r
#carica il pacchetto per contare il tempo
library(tictoc)

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#MCMC
XHMC_completo <- xdnuts(theta0 = lapply(1:10,function(x) theta_start),
     nlp = nlp_completo,
     args = arglist_completo,
     k = n_disc,
     N = 1e3,
     K = 3,
     thin = 1,
     method = "XHMC",
     tau = 1.5,
     parallel = TRUE,
     control = set_parameters(N_init1 = 0,
                              N_adapt = 0,
                              N_init2 = 0,
                              M_type = "dense",
                              M_cont = M_cont,
                              M_disc = M_disc,
                              l_eps_init = log(eps_completo)))

#smetti di contare il tempo
tempo_XHMC_completo <- toc()

#trasformazione del parametro omega in p
b <- xdtransform(XHMC_completo,which = 3,plogis,new.names = "p")

#summary
ESS_XHMC_completo <- summary(b)$stats$ESS

#2) MODELLO CON MARGINALIZZAZIONE DEI PARAMETRI DISCRETI
arglist_ridotto <- list(n = NROW(dati),
                n0_oss = sum(dati$x == 0, na.rm = TRUE),
                n1_oss = sum(dati$x == 1, na.rm = TRUE),
                sum_y = sum(dati$y),
                y_lat = dati$y[dati$idx],
                sum_xy_oss = sum(dati$x[!is.na(dati$x)]*
                                 dati$y[!is.na(dati$x)]),
                hyp = c(10,10,1,1))

#funzione che calcola la log posteriori negativa ed il suo
#gradiente per il modello che marginalizza i parametri discreti
nlp_ridotto <- function(par,args,eval_nlp = TRUE){

  if(eval_nlp){
    #calcola solo il negativo del logaritmo della posteriori

    #teniamo conto dell'overflow
    if(any(abs(par) > 30)) return(Inf)

    #calcoliamo le componenti di cui fare il log_sum_exp
```

```
    A <- cbind(-exp(par[1]+par[2]) + par[2]*args$y_lat,
               -par[3]-exp(par[1]))

    #calcoliamo la log posterior negativa

    #parte del modello di poisson
    exp(par[1]) * (args$n0_oss + args$n1_oss*exp(par[2])) -
      par[1]*args$sum_y - par[2]*args$sum_xy_oss +
      sum(par[1:2]^2 / args$hyp[1:2])/2 +

      #parte del modello bernoulliano
      log(1+exp(-par[3]))*(args$n + args$hyp[3] + args$hyp[4]) +
      par[3]*(args$n0_oss + args$hyp[4]) -

      #parte relativa alla marginalizzazione
      sum(apply(A,1,matrixStats::logSumExp))

  }else{
    #restituisci solo il gradiente

    #teniamo conto dell'overflow
    if(any(abs(par) > 30)) return(rep(Inf,3))

    #inizializza il gradiente
    out <- numeric(3)

    #calcoliamo le componenti di cui fare il log_sum_exp
    A <- cbind(-exp(par[1]+par[2]) + par[2]*args$y_lat,
               -par[3]-exp(par[1]))

    #derivata per alpha
    out[1] <- exp(par[1])*(args$n0_oss + args$n1_oss*exp(par[2])) -
      args$sum_y + par[1]/args$hyp[1]

    #aggiunta della componente log_sum_exp
    B <- cbind(A[,1] + exp(par[1]+par[2]),A[,2] + par[1])
    out[1] <- out[1] + sum(exp(apply(B,1,matrixStats::logSumExp) -
                               apply(A,1,matrixStats::logSumExp)))

    #derivata per beta
    out[2] <- exp(par[1] + par[2])*args$n1_oss -
      args$sum_xy_oss + par[2]/args$hyp[2]

    #aggiunta della componente log_sum_exp con segno
    B <- exp(par[1] + par[2]) - args$y_lat
    out[2] <- out[2] + sum(sign(B) *
      exp(A[,1] + log(abs(B)) - apply(A,1,matrixStats::logSumExp)))

    #derivata per omega
    out[3] <- args$n0_oss + args$hyp[4] - (1 - plogis(par[3])) *
      (args$n + args$hyp[3] + args$hyp[4])

    #aggiunta della componente log_sum_exp con segno
    out[3] <- out[3] + sum(sign(par[3]) *
      exp(A[,2] + log(abs(par[3])) - apply(A,1,matrixStats::logSumExp)))

    #restituisci il gradiente
    out
  }

}

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#MCMC con XHMC
XHMC_ridotto <- xdnuts(lapply(1:10,function(x) theta_start[1:3]),
   nlp_ridotto,
   arglist_ridotto,
   k = 0,
   method = "XHMC",
   tau = 1.5,
   parallel = TRUE,
   control = set_parameters(N_init1 = 0,
                            N_adapt = 0,
                            N_ini2 = 0,
                            M_type = "dense",
                            M_cont = M_cont,
                            l_eps_init = log(eps_completo*0.0001)))

#salva il tempo trascorso
tempo_XHMC_ridotto <- toc()
```

```r
#salva l'Effective Sample Size
b <- xdtransform(XHMC_ridotto,3,plogis,new.names = "p")
ESS_XHMC_ridotto <- summary(b)$stats$ESS

#3) GIBBS SAMPLER
arglist_gibbs <- list(data = dati,
                      hyp = c(0.0001,0.0001,0.0001,0.0001,1,1))

mcmc_gibbs <- function(theta0,args,N_adapt = 1000,N = 1000){

  #pre allocazione
  theta <- matrix(NA,N_adapt + N,length(theta0))
  colnames(theta) <- c("alpha","beta","p",
                       paste0("C_",seq_len(length(theta0)-3)))
  theta[1,] <- theta0
  #inizializziamo i valori mancanti
  x <- args$data$x
  x[args$data$idx] <- theta0[-(1:3)]

  #inizializzazione della matrice di probabilita' dei dati mancanti
  mat_prob <- matrix(NA,length(theta0)-3,2)

  for(i in 2:(N_adapt + N)){

    #numero di case senza gatti
    n0 <- sum(x == 0)

    #numero di case con gatti
    n1 <- sum(x == 1)

    #aggiornamento di lambda
    theta[i,1] <- rgamma(1,
                         sum(args$data$y) + args$hyp[1],
                         n0 + n1*theta[i-1,2] + args$hyp[2])

    #aggiorna gamma
    theta[i,2] <- rgamma(1,
                          sum(args$data$y*x) + args$hyp[3],
                          n1*theta[i,1] + args$hyp[4])

    #aggiorna p
    theta[i,3] <- rbeta(1,
                        n1 + args$hyp[5],
                        n0 + args$hyp[6])

    #aggiorna le probabilita' dei dati mancanti
    mat_prob[,1] <- (1 - theta[i,3])* exp(-theta[i,1])
    mat_prob[,2] <- theta[i,3] *
      theta[i,2]^args$data$y[args$data$idx] * exp(-theta[i,2]*theta[i,1])

    #aggiorna i dati mancanti
    theta[i,-(1:3)] <- apply(mat_prob,1,function(x) sample(c(0,1),1,prob = x))
    x[args$data$idx] <- theta[i,-(1:3)]
  }

  #trasforma lambda e gamma sulla scala originale (alpha e beta)
  theta[,1:2] <- log(theta[,1:2])

  #restituisci i campioni dopo la fase di burn in
  if(N_adapt > 0){
    return(theta[-(seq_len(N_adapt)),])
  }else{
    return(theta)
  }
}

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#fai catene parallele
seeds <- rexp(10,0.001)

#cluster
cl <- parallel::makeCluster(10)

f <- function(i,theta0,mcmc,args,N_adapt,N,seeds){
  #fissa il seme di questo cluster
  set.seed(seeds[i])

  #fai mcmc
  coda::mcmc(mcmc(theta0[[i]],args,N_adapt,N))
}
```

```r
#MCMC via gibbs
gibbs <- coda::as.mcmc.list(
  parallel::parLapply(cl = cl,
    X = seq_len(10),
    fun = f,
    mcmc = mcmc_gibbs,
    theta0 = lapply(1:10, function(x)
      c(exp(2),exp(-1),0.5,
        ceiling(2*plogis(theta_start[-(1:3)])) - 1)),
    args = arglist_gibbs,
    N_adapt = 0,
    N = 1e3,
    seeds = seeds))

#ferma i cluster
parallel::stopCluster(cl)

#ferma il tempo
tempo_gibbs <- toc()

#salva l'effective sample size
ESS_gibbs <- coda::effectiveSize(gibbs)
}
#4) RANDOM WALK METROPOLIS CON IL MODELLO COMPLETO

#utilizziamo i campioni della posteriori ottenuti con XDNUTS
#per costruire una proposal
arglist_completo_rwm <- arglist_completo
arglist_completo_rwm$mu <- mu
arglist_completo_rwm$sigma <- Sigma

#funzione che calcola la log posteriori negativa considerando
#i parametri come discreti, senza trasformarli
nlp_completo_rwm <- function(par,args){

  #teniamo conto dell'overflow
  if(any(abs(par) > 30)) return(Inf)

  #convertiamo i parametri discontinui nei valori mancanti
  x <- args$data$x
  x[args$data$idx] <- par[-(1:3)]

  #numero di case senza gatti
  n0 <- sum(x == 0)

  #numero di case con gatti
  n1 <- sum(x == 1)

  #calcoliamo la log posteriori negativa
  return( exp(par[1])*(n0+n1*exp(par[2])) -
            sum( (par[1] + par[2]*x)*args$data$y) +
            sum(par[1:2]^2/args$hyp[1:2])/2 + #parte del modello poisson

            (n0 + n1 + args$hyp[3] + args$hyp[4])*log(1+exp(-par[3])) +
            (n0+args$hyp[4])*par[3]) #parte del modello bernoulliano
}

#funzione per fare il Random Walk Metropolis
mcmc_rwm_completo <- function(theta0,nlp,args,N_adapt = 1000,N = 1000, eps = 0.1){

  #pre allocazione
  n_disc <- sum(args$data$idx)
  out <- matrix(NA,N_adapt + N,length(theta0))
  colnames(out) <- c("alpha","beta","p",
                     paste0("C_",seq_len(length(theta0)-3)))
  alphas <- numeric(N_adapt + N)

  #inizializziamo i valori mancanti
  x <- args$data$x
  x[args$data$idx] <- theta0[-(1:3)]

  #verifichiamo che la nlp sia finita
  U0 <- nlp(theta0,args)

  if(!is.finite(U0)){
    stop("Starting value not admissible!")
  }

  #MCMC
  for(i in 1:(N_adapt + N)){

    #inizializzazione
    theta <- theta0

    #proponi da una t multivariata per i parametri di interesse
    theta[1:3] <- args$mu + mvtnorm::rmvt(1,args$sigma*eps,df = 1)
```

```
    #calcola il valore dell'energia potenziale
    U1 <- nlp(theta,args)

    #calcola il tasso di accettazione Metropolis-Hastings
    alphas[i] <- min(1,exp(U0 - U1 +
                mvtnorm::dmvt(theta0[1:3]-args$mu,
                              sigma = args$sigma*eps,df = 1, log = TRUE) -
                mvtnorm::dmvt(theta[1:3] - args$mu,
                              sigma = args$sigma*eps,df = 1, log = TRUE)))

    #accettazione della proposal?
    if(runif(1) < alphas[i]){
      theta0 <- theta
      U0 <- U1
    }

    #aggiorna i parametri di disturbo
    for(j in seq_len(n_disc)){

      #inizializzazione
      theta <- theta0

      #aggiorna il j-esimo valore mancante
      theta[3+j] <- sample(0:1,1)

      #calcola la nuova energia potenziale
      U1 <- nlp(theta,args)

      #accettazione della proposal?
      if(runif(1) < exp(U0 - U1)){
        theta0 <- theta
        U0 <- U1
      }
    }

    #aggiorna il valore della catena
    out[i,] <- theta0

  }

  #trasforma p sulla scala originale
  out[,3] <- plogis(out[,3])

  #restituisci l'output
  if(N_adapt > 0){
    return(list(values = out[-(1:N_adapt),],
                alpha = mean(alphas[-(1:N_adapt)])))
  }else{
    return(list(values = out,
                alpha = mean(alphas)))
  }
}

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#fai catene parallele
seeds <- rexp(10,0.001)

#cluster
cl <- parallel::makeCluster(10)

f <- function(i,theta0,mcmc,nlp,args,N_adapt,N,eps,seeds){
  #fissa il seme di questo cluster
  set.seed(seeds[i])

  #fai mcmc
  coda::mcmc(mcmc(theta0[[i]],nlp,args,N_adapt,N,eps)$values)
}

#MCMC via Random Walk Metropolis
rwm_completo <- coda::as.mcmc.list(
  parallel::parLapply(cl = cl,
                      X = seq_len(10),
                      fun = f,
                      mcmc = mcmc_rwm_completo,
                      theta0 = lapply(1:10, function(x)
                        c(theta_start[1:3],
                          ceiling(2*plogis(theta_start[-(1:3)])) - 1)),
                      nlp = nlp_completo_rwm,
                      args = arglist_completo_rwm,
                      N_adapt = 0,
```

```
                    N = 1e3,
                    eps = 1,
                    seeds = seeds))

#ferma i cluster
parallel::stopCluster(cl)

#ferma il tempo
tempo_rwm_completo <- toc()

#salva l'effective sample size
ESS_rwm_completo <- coda::effectiveSize(rwm_completo)

### RWM VERSIONE MARGINALIZZATA
arglist_ridotto_rwm <- arglist_ridotto
arglist_ridotto_rwm$sigma <- Sigma
arglist_ridotto_rwm$mu <- mu

mcmc_rwm_ridotto <- function(theta0,nlp,args,N_adapt = 1000,N = 1000, eps = 0.1){

  #pre allocazione
  n_disc <- sum(args$data$idx)
  out <- matrix(NA,N_adapt + N,length(theta0))
  colnames(out) <- c("alpha","beta","p")
  alphas <- numeric(N_adapt + N)

  #verifichiamo che la nlp sia finita
  U0 <- nlp(theta0,args)

  if(!is.finite(U0)){
    stop("Starting value not admissible!")
  }

  #MCMC
  for(i in 1:(N_adapt + N)){

    #inizializzazione
    theta <- theta0

    #proponi da una t multivariata per i parametri di interesse
    theta <- args$mu + mvtnorm::rmvt(1,args$sigma*eps,df = 3)

    #calcola il valore dell'energia potenziale
    U1 <- nlp(theta,args)

    #calcola il tasso di accettazione Metropolis-Hastings
    alphas[i] <- min(1,exp(U0 - U1 +
                           mvtnorm::dmvt(theta0-args$mu,
                                         sigma = args$sigma*eps,df = 3, log = TRUE) -
                           mvtnorm::dmvt(theta - args$mu,
                                         sigma = args$sigma*eps,df = 3, log = TRUE)))

    #accettazione della proposal?
    if(runif(1) < alphas[i]){
      theta0 <- theta
      U0 <- U1
    }

    #aggiorna il valore della catena
    out[i,] <- theta0

  }

  #trasforma p sulla scala originale
  out[,3] <- plogis(out[,3])

  #restituisci l'output
  if(N_adapt > 0){
    return(list(values = out[-(1:N_adapt),],
                alpha = mean(alphas[-(1:N_adapt)])))
  }else{
    return(list(values = out,
                alpha = mean(alphas)))
  }

}

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#fai catene parallele
seeds <- rexp(10,0.001)
```

```r
#cluster
cl <- parallel::makeCluster(10)

f <- function(i,theta0,mcmc,nlp,args,N_adapt,N,eps,seeds){
  #fissa il seme di questo cluster
  set.seed(seeds[i])

  #fai mcmc
  coda::mcmc(mcmc(theta0[[i]],nlp,args,N_adapt,N,eps)$values)
}

#MCMC via Random Walk Metropolis
rwm_ridotto <- coda::as.mcmc.list(
  parallel::parLapply(cl = cl,
                      X = seq_len(10),
                      fun = f,
                      mcmc = mcmc_rwm_ridotto,
                      theta0 = lapply(1:10, function(x)
                        theta_start[1:3]),
                      nlp = nlp_ridotto,
                      args = arglist_ridotto_rwm,
                      N_adapt = 0,
                      N = 1e3,
                      eps = 0.8,
                      seeds = seeds))

#ferma i cluster
parallel::stopCluster(cl)

#ferma il tempo
tempo_rwm_ridotto <- toc()

#salva l'effective sample size
ESS_rwm_ridotto <- coda::effectiveSize(rwm_ridotto)

#creazione della tabella finale
tabella <- matrix(NA,12,5)
colnames(tabella) <- c("DXHMC","XHMC",
                       "MCID","MCID:M","Gibbs")
rownames(tabella) <- c(paste0(c("mean","sd","ESS","ESS/s"),"_alpha"),
                       paste0(c("mean","sd","ESS","ESS/s"),"_beta"),
                       paste0(c("mean","sd","ESS","ESS/s"),"_p"))

#medie e deviazioni standard
tabella[c(1,2,5,6,9,10),1] <- c(apply(xdextract(xdtransform(
  XHMC_completo,which = 3,plogis), collapse = TRUE)[,1:3], 2,
  function(x) c(mean(x),sd(x))))
tabella[c(1,2,5,6,9,10),2] <- c(apply(xdextract(xdtransform(
  XHMC_ridotto,which = 3,plogis), collapse = TRUE)[,1:3] ,2,
  function(x) c(mean(x),sd(x))))
tabella[c(1,2,5,6,9,10),3] <- c(apply(do.call(rbind,rwm_completo)[,1:3],2,
  function(x) c(mean(x),sd(x))))
tabella[c(1,2,5,6,9,10),4] <- c(apply(do.call(rbind,rwm_ridotto)[,1:3],2,
  function(x) c(mean(x),sd(x))))
tabella[c(1,2,5,6,9,10),5] <- c(apply(do.call(rbind,gibbs)[,1:3] ,2,
  function(x) c(mean(x),sd(x))))

#ESS e ESS per secondo
tabella[c(3,4,7,8,11,12),1] <- c(sapply(ESS[,1],function(x)
  c(x*(tempo_XHMC_completo$toc - tempo_XHMC_completo$tic),x)) )
tabella[c(3,4,7,8,11,12),2] <- c(sapply(ESS[,2],function(x)
  c(x*(tempo_XHMC_ridotto$toc - tempo_XHMC_ridotto$tic),x)) )
tabella[c(3,4,7,8,11,12),3] <- c(sapply(ESS[,4],function(x)
  c(x*(tempo_rwm_completo$toc - tempo_rwm_completo$tic),x)) )
tabella[c(3,4,7,8,11,12),4] <- c(sapply(ESS[,5],function(x)
  c(x*(tempo_rwm_ridotto$toc - tempo_rwm_ridotto$tic),x)) )
tabella[c(3,4,7,8,11,12),5] <- c(sapply(ESS[,3],function(x)
  c(x*(tempo_gibbs$toc - tempo_gibbs$tic),x)) )

#mostra la tabella
knitr::kable(tabella, digits = 3)


### IMPUTAZIONE DATI MANCANTI

imputazioni <- data.frame(verita = gatti_full[dati$idx])

#XHMC completo

#trasformazione sulla scala originale
f <- function(x) c(x[1:2],plogis(x[3]),ceiling(2*plogis(x[-(1:3)])) - 1)
b <- xdtransform(XHMC_completo,NULL,f,
                 new.names = c("alpha","beta","p",paste0("C*",1:n_disc)))
theta <- xdextract(b, collapse = TRUE)
imputazioni$XHMC_completo <- apply(theta[,-(1:3)],2,mean)
```

```
#XHMC ridotto
b <- xdtransform(XHMC_ridotto,3,plogis,new.names = "p")
theta <- xdextract(b, collapse = TRUE)

#funzione che simula i dati mancanti
f <- function(x,args){
  #matrice di probabilita' per la generazione post-MCMC
  mat_prob <- matrix(NA,length(args$y_lat),2)
  mat_prob[,1] <- (1 - x[3]) * exp(-exp(x[1]))
  mat_prob[,2] <- x[3] * exp(x[2]*args$y_lat) * exp(-exp(x[1] + x[2]))
  #genera i valori mancanti
  apply(mat_prob,1,function(x) sample(c(0,1),1,prob = x))
}
imputazioni$XHMC_ridotto <-
  apply(apply(theta,1,f,arglist_ridotto),1,mean)

#gibbs
theta <- do.call(rbind,gibbs)
imputazioni$gibbs <- apply(theta[,-(1:3)],2,mean)

#rwm completo
theta <- do.call(rbind,rwm_completo)
imputazioni$rwm_completo <- apply(theta[,-(1:3)],2,mean)

#rwm ridotto
theta <- do.call(rbind,rwm_ridotto)
imputazioni$rwm_ridotto <-
  apply(apply(theta,1,f,arglist_ridotto_rwm),1,mean)

imputazioni <- imputazioni[,c(1,2,3,5,6,4)]

#media della probabilita' a posteriori
medie <- c(mean(xdextract(xdtransform(XHMC_completo,3,plogis),collapse = TRUE)[,3]),
           mean(xdextract(xdtransform(XHMC_ridotto,3,plogis),collapse = TRUE)[,3]),
           mean(do.call(rbind,rwm_completo)[,3]),
           mean(do.call(rbind,rwm_ridotto)[,3]),
           mean(do.call(rbind,gibbs)[,3]))

#ordina i dati mancanti in base al valore osservato di N
idx <- order(dati$y[dati$idx], decreasing = TRUE)

{
  par(xpd = TRUE, mar = c(1.1,6.1,1.1,1.1))
  plot(c(1,24),range((imputazioni[,-1])), type= "n",
       bty = "L", xlab = "", ylab = "", xaxt = "n")
  text(-2.7,0.5,expression(hat(P)(C == 1)), font = 2, cex = 1.1)
  text(11,0.95,"0", font = 2, cex = 2)
  text(16,0.95,"1", font = 2, cex = 2)
  lines(c(13.5,13.5),c(-0.04,1.01), col = 2, lty = 2)
  for(i in 1:24){
    points(rep(i,5),(imputazioni[idx[i],-1]),
           col = 1:5, pch = 16, cex = 0.7)
  }
  for(i in 1:5){
    lines(c(0,24),rep(medie[i],2), col = i, lty = i+2, lwd = 1)
  }
  legend(22,0.238,col = 1:5, pch = 16,
         legend = c("XDHMC","XHMC","MCID",bquote(MCID[M]),"Gibbs"),
         cex = 0.8, bg = "transparent")
}
```

C.4 Modello per punti di cambio

```
#seed
set.seed(123)

#veri valori dei parametri:

#tasso nel primo periodo
lambda1 <- 4

#tasso nel secondo periodo
lambda2 <- 1

#tasso nel terzo periodo
lambda3 <- 5

#primo punto di cambio
t1 <- 100

#secondo punto di cambio
t2 <- 200

#numerosita' campionaria
N <- 300

#simualzione dei dati
Y <- rpois(N,c(rep(lambda1,t1),rep(lambda2,t2-t1),rep(lambda3,N-t2)))

#grafico
ts.plot(Y)

#1) MODELLO COMPLETO

#log posteriori negativa e relativo gradiente
nlp_completo <- function(par,args,eval_nlp = TRUE){

  if(eval_nlp){
    #restituisci solo la log posteriori negativa

    #teniamo conto dell'overflow
    if(any(abs(par) > 30) ) return(Inf)

    #ritrasformazione dei parametri discreti:
    p1 <- plogis(par[4])
    p2 <- plogis(par[5])

    #primo punto di cambio
    t1 <- ceiling(args$hyp[7] + (args$hyp[8] - args$hyp[7])*p1) - 1

    #secondo punto di cambio
    t2 <- ceiling(t1 + 1 + (args$hyp[8] - t1)*p2) - 1

    #numerosita' campionaria
    tt <- length(args$data)

    #restituisci la log posteriore negativa
    return(
      #contributo del primo periodo
      exp(par[1]) * (t1 + args$hyp[2]) -
        par[1]*(sum(args$data[1:t1] + args$hyp[1])) +

        #contribuoto del secondo periodo
        exp(par[2]) * (t2 - t1 + args$hyp[4]) -
        par[2]*(sum(args$data[(t1+1):t2] + args$hyp[3])) +

        #contributo del terzo periodo
        exp(par[3]) * (tt - t2 + args$hyp[6]) -
        par[3]*(sum(args$data[-(1:t2)] + args$hyp[5])) +

        #contributo delle priori dei parametri discreti
        (par[4] + par[5]) + 2*log(1+exp(-par[4])) + 2*log(1+exp(-par[5]))
    )

  }else{
    #restituisci solo il gradiente della log posteriori negativa

    #teniamo conto dell'overflow
    if(any(abs(par) > 30) ) return(rep(Inf,3))

    #ritrasformazione dei parametri discreti

    #primo punto di cambio
    t1 <- ceiling(args$hyp[7]+(args$hyp[8]-args$hyp[7])*plogis(par[4]))-1

    #secondo punto di cambio
```

```
    t2 <- ceiling(t1 + 1 + (args$hyp[8] - t1)*plogis(par[5])) - 1

    #numerosita' campionaria
    tt <- length(args$data)

    #restituiamo il gradiente
    c(
      #derivata per il log-tasso del primo periodo
      exp(par[1]) * (t1 + args$hyp[2]) -
        (sum(args$data[1:t1] + args$hyp[1])),

      #derivata per il log-tasso del secondo periodo
      exp(par[2]) * (t2 - t1 + args$hyp[4]) -
        (sum(args$data[(t1+1):t2] + args$hyp[3])),

      #derivata per il log tasso del terzo periodo
      exp(par[3]) * (tt - t2 + args$hyp[6]) -
        (sum(args$data[-(1:t2)] + args$hyp[5]))
    )
  }

}

#iperparametri
hyp0 <- c(rep(0.001,6), #per la gamma
          1, #valore iniziale in cui ricercare i punti di cambio
          299#valore finale in cui ricercare i punti di cambio
)

#lista da fornire in input
arglist_completo = list(data = Y, hyp = hyp0)

#carica il pacchetto XDNUTS
library(XDNUTS)

#caso con fase di warm up troppo breve

#fissa il seme
set.seed(5)

#MCMC
NUTS_completo <- xdnuts(lapply(1:10, function(x) {
  out <- rnorm(5)
  names(out) <- c(paste0("log_lambda",1:3),"t*_tilde1","t*_tilde2")
  out
}),
nlp = nlp_completo,
args = arglist_completo,
k = 2,
N = 1e3,
K = 3,
method = "NUTS",
thin = 1,
parallel = TRUE,
control = set_parameters(delta = c(0.8,0.3),
                         N_adapt = 500,
                         burn_adapt_ratio = 0.5))

#grafico della catena dell'energia
plot(NUTS_completo, type = 3, cex.legend = 1.5)

#campionamento effettivo

#carica il pacchetto per contare il tempo
library(tictoc)

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#MCMC
NUTS_completo <- xdnuts(lapply(1:10, function(x) {
  out <- rnorm(5)
  names(out) <- c(paste0("log_lambda",1:3),"t*_tilde1","t*_tilde2")
  out
}),
nlp = nlp_completo,
args = arglist_completo,
k = 2,
N = 1e3,
K = 3,
method = "NUTS",
thin = 1,
parallel = TRUE,
```

```
control = set_parameters(delta = c(0.9,0.6),
                         keep_warm_up = TRUE,
                         N_adapt = 1e3,
                         burn_adapt_ratio = 0.7))

#smetti di contare il tempo
tempo <- toc()

#trasformazione del parametro omega in p
b <- xdtransform(NUTS_completo,which = 1:3,
                 exp,new.names = "lambda")

#grafico dei tassi di accettazione/riflessione
plot(b, type = 7)

#grafico delle catene
plot(b, cex.legend = 1.5)

#grafico della catena dell'energia
plot(b, type = 3, cex.legend = 1.5)

#grafico delle frequenze dei passi di integrazione
plot(b, type = 4, cex.legend = 1.3)

#grafici delle densita'
plot(b, type = 2, which = "continuous", cex.titles = 1.2)
plot(b, type = 2, which = "discontinuous", cex.titles = 1.2)
plot(b, type = 2, cex.titles = 1.2)

#grafico BFMI
plot(b,type = 5, cex.legend = 2)

#grafico autocorrelazioni
plot(b,type = 6, cex.legend = 2, cex.titles = 2.5)

#summary
summary(b)

#campioni sulla scala originale
f <- function(x,args) {
  #primo punto di cambio
  t1 <- ceiling(args$hyp[7]+(args$hyp[8]-args$hyp[7])*plogis(x[4]))-1

  #secondo punto di cambio
  t2 <- ceiling(t1 + 1 + (args$hyp[8] - t1)*plogis(x[5])) - 1

  #trasformate
  c(exp(x[1:3]),t1,t2)
}

#trasformata
b <- xdtransform(NUTS_completo,NULL,f,args = arglist_completo,
                 new.names = c(paste0("lambda",1:3),"t1","t2"))
#grafico densita'
plot(b, type = 2)

#vediamo se le previsioni per i vari periodi sono buone
y_hat <- function(x, args){
  #primo punto di cambio
  t1 <- ceiling(args$hyp[7]+(args$hyp[8]-args$hyp[7])*plogis(x[4]))-1

  #secondo punto di cambio
  t2 <- ceiling(t1 + 1 + (args$hyp[8] - t1)*plogis(x[5])) - 1

  #simula i dati nuovamente
  c(
    rpois(t1,exp(x[1])),
    rpois(t2-t1,exp(x[2])),
    rpois(300-t2,exp(x[3]))
  )
}

#estrai i dati sulla scala originale
omega <- xdextract(NUTS_completo,collapse = TRUE)

#simula dalla distribuzione previsiva
y_hat_post <- t(apply(omega,1,y_hat, args = arglist_completo))

#calcola gli intervalli di credibilita'
y_hat_ic <- t(apply(y_hat_post,2,quantile,c(0.1,0.9)))

#grafico delle coperture
ts.plot(Y, xlab = "Tempo")
for(i in 1:300){
  lines(rep(i,2),y_hat_ic[i,], col = adjustcolor(2,0.5))
}
```

```
### salvataggio delle quantita' necessarie per il confronto tra
### i diversi metodi
theta_start <- apply(xdextract(NUTS_completo,collapse = TRUE),2,mean)

#step-size
eps_completo <- mean(NUTS_completo$chains[[1]]$step_size)

#matrice di massa per le componenti continue
M_cont <- NUTS_completo$chains[[1]]$M_cont

#matrice di massa per la componenti discontinue
M_disc <- NUTS_completo$chains[[1]]$M_disc
```

C.4.1 Confronto con altri algoritmi

```
#carica il pacchetto per contare il tempo
library(tictoc)

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#MCMC
NUTS_completo <- xdnuts(lapply(1:10, function(x) theta_start),
  nlp = nlp_completo,
  args = arglist_completo,
  k = 2,
  N = 1e3,
  K = 3,
  method = "NUTS",
  thin = 1,
  parallel = TRUE,
  control = set_parameters(N_init1 = 0,
                           N_adapt = 200,
                           N_init2 = 0,
                           M_type = "dense",
                           M_cont = M_cont,
                           M_disc = M_disc,
                           l_eps_init = log(eps_completo)))

#smetti di contare il tempo
tempo <- toc()

#trasformazione del parametro omega in p
b <- xdtransform(NUTS_completo,which = 1:3,
                 exp,new.names = "lambda")
#summary
ESS_NUTS_completo <- summary(b)$stats$ESS

#2) MODELLO CON PARAMETRI DISCRETI MARGINALIZZATI

#creazione della lista con le statistiche sufficienti
t_inizio <- 1
t_fine <- 299
t1_grid <- t_inizio:(t_fine-1)
tt_grid <- do.call(rbind,lapply(t1_grid,function(x) cbind(x,(x+1):t_fine)))

#per ciascun elemento possiamo calcolare gia' le statistiche sufficienti
#in maniera da alleggerire il costo computazionale
stat_suff <- matrix(NA,NROW(tt_grid),7)
for(i in seq_len(NROW(tt_grid))){

  #numero di tempi considerati da ciascun intervallo
  stat_suff[i,1:3] <- c(tt_grid[i,1],diff(tt_grid[i,]),300 - tt_grid[i,2])

  #somma della variabile risposta lungo i primi tre intervalli
  stat_suff[i,4] <- sum(Y[1:tt_grid[i,1]])
  stat_suff[i,5] <- sum(Y[(tt_grid[i,1]):tt_grid[i,2] ])
  stat_suff[i,6] <- sum(Y[(tt_grid[i,2]):300])

  #logaritmo della probabilita' uniforme per la coppia
  #dei due punti di cambio
  stat_suff[i,7] <- log(t_fine - t_inizio) + log(t_fine - tt_grid[i,1])
}

#lista di argomenti per il modello ridotto
arglist_ridotto <- list(data = stat_suff, hyp = hyp0)
```

```r
#funzione add_sign_log_sum_exp
#necessaria per il calcolo analitico del gradiente
add_sign_log_sum_exp <- function(log_abs_x,sign_x){

  #prendiamo il valore massimo
  max_val <- max(log_abs_x)

  #facciamo la somma sulla scala originale riscalandola
  sum_exp <- sum(sign_x * exp(log_abs_x - max_val))

  #retituiamo il segno e il logaritmo di questa sulla scala originale
  c(max_val + log(abs(sum_exp)),
    sign(sum_exp))
}

#log posteriori negativa per il modello ridotto
nlp_ridotto <- function(par,args,eval_nlp = TRUE){

  if(eval_nlp){
    #restituisci solo il negativo della posteriori negativa

    #contributo della priori
    sum(-par*args$hyp[c(1,3,5)] + args$hyp[c(2,4,6)]*exp(par) ) -

      #log_sum_exp
      matrixStats::logSumExp(
      apply(args$data,1,function(x)
        sum(par*x[4:6] - x[1:3]*exp(par)) - x[7])
    )

  }else{
    #restituisci solo il gradiente

    #calcola i pesi di ciascun gradiente e i
    #valori da pesare per le tre derivate
    lsw <- t(apply(args$data,1,function(x)
      c(
        sum(par*x[4:6] - x[1:3]*exp(par)) - x[7],
        x[4:6] -x[1:3]*exp(par)
      )
    ))

    #calcola i segni
    sign_val <- apply(lsw[,-1],2,sign)

    #calcola i logaritmi dei valori assoluti dei
    #valori dei gradienti da pesare e sommaci
    #il loro peso logaritmico
    lsw[,-1] <- log(abs(lsw[,-1])) + lsw[,1]

    #log_sum_exp con segno del numeratore
    num <- c(add_sign_log_sum_exp(lsw[,2],sign_val[,1]),
             add_sign_log_sum_exp(lsw[,3],sign_val[,2]),
             add_sign_log_sum_exp(lsw[,4],sign_val[,3]))


    #procedi con il calcolo dei gradienti
    args$hyp[c(2,4,6)]*exp(par) - args$hyp[c(1,3,5)] -
      num[c(2,4,6)] * exp(num[c(1,3,5)] -
                          matrixStats::logSumExp(lsw[,1]) )
  }

}

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

NUTS_ridotto <- xdnuts(lapply(1,function(x) theta_start[1:3]),
           nlp_ridotto,
           arglist_ridotto,
           k = 0,
           method = "NUTS",
           parallel = FALSE,
           control = set_parameters(max_treedepth = 5,
                                    l_eps_init = log(0.07),
                                    M_type = "dense",
                                    N_init1 = 0,
                                    N_init2 = 0,
                                    N_adapt = 0,
                                    M_cont = M_cont))
#salva il tempo trascorso
tempo_NUTS_ridotto <- toc()
```

```r
#salva l'Effective Sample Size
b <- xdtransform(NUTS_ridotto,NULL,exp,new.names = "lambda")
ESS_NUTS_ridotto <- summary(b)$stats$ESS

### GIBBS

#matrice di indici per accedere velocemente alle statistiche
#sufficienti di ogni configurazione
idx_matrix <- matrix(NA,t_fine - t_inizio + 1,
                     t_fine - t_inizio + 1)
for(i in seq_len(NROW(tt_grid))){
  idx_matrix[tt_grid[i,1],tt_grid[i,2]] <- i
}

#nuova lista di elementi per l'MCMC
arglist_gibbs <- list(data = stat_suff,
                      idx_matrix = idx_matrix,
                      hyp = hyp0)

mcmc_gibbs <- function(theta0,args,N_adapt = 1e3,N = 1e3){

  #pre allocazione
  theta <- matrix(NA,N_adapt + N,length(theta0))
  log_prob <- prob <- numeric(NROW(args$data))

  for(i in 1:(N_adapt + N)){

    #calcolo delle correnti statistiche sufficienti
    ss <- args$data[args$idx_matrix[theta0[4],theta0[5]],1:6]

    #simula dai vari tassi
    theta[i,1:3] <- rgamma(3,
                           ss[4:6] + args$hyp[c(1,3,5)],
                           ss[1:3] + args$hyp[c(2,4,6)])

    #aggiorna il vettore di probabilita'
    log_prob <- apply(args$data,1,function(x)
      sum(-theta[i,1:3]*(x[1:3] + args$hyp[c(2,4,6)]) +
        log(theta[i,1:3])*(x[4:6] + args$hyp[c(1,3,5)] - 1)) - x[7])
    log_prob <- log_prob - matrixStats::logSumExp(log_prob)
    prob <- exp(log_prob)

    #campiona un indice
    idx <- sample(seq_len(NROW(args$data)),1,prob = prob)

    #aggiorna il valore dei punti di cambio
    theta[i,4:5] <- c(args$data[idx,1],sum(args$data[idx,1:2]))

    #aggiorna i valori di theta0
    theta0 <- theta[i,]

    cat(i," ")
  }

  #restituisci i campioni dopo la fase di burn in
  if(N_adapt > 0){
    return(theta[-(seq_len(N_adapt)),])
  }else{
    return(theta)
  }
}

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#fai catene parallele
seeds <- rexp(10,0.001)

#cluster
cl <- parallel::makeCluster(10)

f <- function(i,theta0,mcmc,args,N_adapt,N,seeds){
  #fissa il seme di questo cluster
  set.seed(seeds[i])

  #fai mcmc
  coda::mcmc(mcmc(theta0[[i]],args,N_adapt,N))
}

#MCMC via gibbs
gibbs <- coda::as.mcmc.list(
  parallel::parLapply(cl = cl,
                      X = seq_len(10),
```

```
                    fun = f,
                    mcmc = mcmc_gibbs,
                    theta0 = lapply(1:10, function(x)
                      c(theta_start[1:3],t1 = 100, t2 = 200)),
                    args = arglist_gibbs,
                    N_adapt = 0,
                    N = 1e3,
                    seeds = seeds))

#ferma i cluster
parallel::stopCluster(cl)

#ferma il tempo
tempo_gibbs <- toc()

#salva l'effective sample size
ESS_gibbs <- coda::effectiveSize(gibbs)

### RANDOM WALK METROPOLIS

#riscrittura della nlp del modello completo ma con
#i parametri discreti non portati nel continuo
nlp_rwm_completo <- function(par,args){

  #teniamo conto dell'overflow
  if(any(abs(par[1:3]) > 30)) return(Inf)

  return(
    #contributo del primo periodo
    exp(par[1]) * (par[4] + args$hyp[2]) -
      par[1]*(sum(args$data[1:par[4]] + args$hyp[1])) +

      #contribuoto del secondo periodo
      exp(par[2]) * (par[5] - par[4] + args$hyp[4]) -
      par[2]*(sum(args$data[(par[4]+1):par[5]] + args$hyp[3])) +

      #contributo del terzo periodo
      exp(par[3]) * (300 - par[5] + args$hyp[6]) -
      par[3]*(sum(args$data[-(1:par[5])] + args$hyp[5])) -

      #contributo delle priori dei parametri discreti
      log(args$hyp[8] - args$hyp[7]) - log(args$hyp[8] - par[4])
  )
}

mcmc_rwm_completo <- function(theta0,nlp,args,N_adapt = 1000,
                              N = 1000, eps = 0.1){

  #pre allocazione
  out <- matrix(NA,N_adapt + N,length(theta0))
  alphas <- matrix(NA,N_adapt + N,3)

  #verifichiamo che la nlp sia finita
  U0 <- nlp(theta0,args)

  if(!is.finite(U0)){
    stop("Starting value not admissible!")
  }

  #MCMC
  for(i in 1:(N_adapt + N)){

    #inizializzazione
    theta <- theta0

    #proponi da una normale multivariata per i parametri di interesse
    theta[1:3] <- rnorm(3,theta0[1:3],eps[1])

    #calcola il valore dell'energia potenziale
    U1 <- nlp(theta,args)

    #calcola il tasso di accettazione Metropolis
    alphas[i,1] <- min(1,exp(U0 - U1))

    #accettazione della proposal?
    if(runif(1) < alphas[i,1]){
      theta0 <- theta
      U0 <- U1
    }

    #aggiorna gli istanti di tempo
    for(j in 1:2){
      theta <- theta0

      theta[3+j] <- theta[3+j] + (-1)^rbinom(1,1,0.5)*round(eps[1+j])
```

```
      #calcola la nuova energia potenziale
      if(theta[4] >= theta[5]){
        #stato non ammissibile
        U1 <- Inf
      }else{
        U1 <- nlp(theta,args)
      }

      #calcola il tasso di accettazione Metropolis
      alphas[i,1+j] <- min(1,exp(U0 - U1))

      #accettazione della proposal
      if(runif(1) < alphas[i,1+j]){
        theta0 <- theta
        U0 <- U1
      }
    }

    #aggiorna il valore della catena
    out[i,] <- theta0

  }

  #trasforma i parametri continui sulla scala originale
  out[,1:3] <- exp(out[,1:3])

  #restituisci l'output
  if(N_adapt > 0){
    return(list(values = out[-(1:N_adapt),],
                alpha = mean(alphas[-(1:N_adapt)])))
  }else{
    return(list(values = out,
                alpha = mean(alphas)))
  }
}

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#fai catene parallele
seeds <- rexp(10,0.001)

#cluster
cl <- parallel::makeCluster(10)

f <- function(i,theta0,mcmc,nlp,args,N_adapt,N,eps,seeds){
  #fissa il seme di questo cluster
  set.seed(seeds[i])

  #fai mcmc
  coda::mcmc(mcmc(theta0[[i]],nlp,args,N_adapt,N,eps)$values)
}

#MCMC via Random Walk Metropolis
rwm_completo <- coda::as.mcmc.list(
  parallel::parLapply(cl = cl,
                      X = seq_len(10),
                      fun = f,
                      mcmc = mcmc_rwm_completo,
                      theta0 = lapply(1:10, function(x)
                        c(theta_start[1:3],t1 = 100,t2 = 200)),
                      nlp = nlp_completo_rwm,
                      args = arglist_completo,
                      N_adapt = 0,
                      N = 1e3,
                      eps = c(0.06,1,1),
                      seeds = seeds))

#ferma i cluster
parallel::stopCluster(cl)

#ferma il tempo
tempo_rwm_completo <- toc()

#salva l'effective sample size
ESS_rwm_completo <- coda::effectiveSize(rwm_completo)

### RWM VERSIONE MARGINALIZZATA

t_inizio <- 1
t_fine <- 299
t1_grid <- t_inizio:(t_fine-1)
tt_grid <- do.call(rbind,lapply(t1_grid,function(x)
```

```
  cbind(x,(x+1):t_fine)))

#per ciascun elemento possiamo calcolare gia' le statistiche sufficienti
#questo e' un effettivo vantaggio

mcmc_rwm_ridotto <- function(theta0,nlp,args,N_adapt = 1000,N = 1000, eps = 0.1){

  #pre allocazione
  out <- matrix(NA,N_adapt + N,length(theta0))
  alphas <- numeric(N_adapt + N)

  #verifichiamo che la nlp sia finita
  U0 <- nlp(theta0,args)

  if(!is.finite(U0)){
    stop("Starting value not admissible!")
  }

  #MCMC
  for(i in 1:(N_adapt + N)){

    #inizializzazione
    theta <- theta0

    #proponi da una normale multivariata per i parametri di interesse
    theta <- rnorm(3,theta0,eps[1])

    #calcola il valore dell'energia potenziale
    U1 <- nlp(theta,args)

    #calcola il tasso di accettazione Metropolis
    alphas[i] <- min(1,exp(U0 - U1))

    #accettazione della proposal?
    if(runif(1) < alphas[i]){
      theta0 <- theta
      U0 <- U1
    }

    #aggiorna il valore della catena
    out[i,] <- theta0
    cat(" ",i)
  }

  #riporta i parametri continui nello spazio originale
  out <- exp(out)

  #restituisci l'output
  if(N_adapt > 0){
    return(list(values = out[-(1:N_adapt),],
                alpha = mean(alphas[-(1:N_adapt)])))
  }else{
    return(list(values = out,
                alpha = mean(alphas)))
  }
}

#inizia a contare il tempo
tic()

#fissa il seme
set.seed(123)

#fai catene parallele
seeds <- rexp(10,0.001)

#cluster
cl <- parallel::makeCluster(10)

f <- function(i,theta0,mcmc,nlp,args,N_adapt,N,eps,seeds){
  #fissa il seme di questo cluster
  set.seed(seeds[i])

  #fai mcmc
  coda::mcmc(mcmc(theta0[[i]],nlp,args,N_adapt,N,eps)$values)
}

#MCMC via Random Walk Metropolis
rwm_ridotto <- coda::as.mcmc.list(
  parallel::parLapply(cl = cl,
                      X = seq_len(10),
                      fun = f,
                      mcmc = mcmc_rwm_ridotto,
                      theta0 = lapply(1:10, function(x)
                        theta_start),
                      nlp = nlp_ridotto,
```

```
                    args = arglist_ridotto,
                    N_adapt = 0,
                    N = 1e3,
                    eps = 0.07,
                    seeds = seeds))

#ferma i cluster
parallel::stopCluster(cl)

#ferma il tempo
tempo_rwm_ridotto <- toc()

#salva l'effective sample size
ESS_rwm_ridotto <- coda::effectiveSize(rwm_ridotto)

#CONFRONTI IN TERMINI DI ESS per secondo
ESS <- cbind(ESS_NUTS_completo[1:3]/
               (tempo_NUTS_completo$toc - tempo_NUTS_completo$tic),
             ESS_NUTS_ridotto/
               (tempo_NUTS_ridotto$toc - tempo_NUTS_ridotto$tic),
             ESS_gibbs[1:3]/
               (tempo_gibbs$toc - tempo_gibbs$tic),
             ESS_rwm_completo[1:3]/
               (tempo_rwm_completo$toc - tempo_NUTS_completo$tic),
             ESS_rwm_ridotto/
               (tempo_rwm_ridotto$toc - tempo_rwm_ridotto$tic))

#creiamo la tabella finale
tabella <- matrix(NA,12,5)
colnames(tabella) <- c("DNUTS","NUTS",
                       "RWM","RWM_M","Gibbs")
rownames(tabella) <- c(paste0(c("mean","sd","ESS","ESS/s"),"_lambda1"),
                       paste0(c("mean","sd","ESS","ESS/s"),"_lambda2"),
                       paste0(c("mean","sd","ESS","ESS/s"),"_lambda3"))

#medie e deviazioni standard a posteriori
tabella[c(1,2,5,6,9,10),1] <- c(apply(xdextract(xdtransform(
  NUTS_completo,which = 1:3,exp), collapse = TRUE)[,1:3],2,
  function(x) c(mean(x),sd(x))))
tabella[c(1,2,5,6,9,10),2] <- c(apply(xdextract(xdtransform(
  NUTS_ridotto,which = 1:3,exp), collapse = TRUE)[,1:3],2,
  function(x) c(mean(x),sd(x))))
tabella[c(1,2,5,6,9,10),3] <- c(apply(do.call(rbind,rwm_completo)[,1:3],2
, function(x) c(mean(x),sd(x))))
tabella[c(1,2,5,6,9,10),4] <- c(apply(do.call(rbind,rwm_ridotto)[,1:3],2,
  function(x) c(mean(x),sd(x))))
tabella[c(1,2,5,6,9,10),5] <- c(apply(do.call(rbind,gibbs)[,1:3],2,
  function(x) c(mean(x),sd(x))))

#ESS e ESS/s
tabella[c(3,4,7,8,11,12),1] <- c(sapply(ESS[,1],function(x)
  c(x*(tempo_NUTS_completo$toc - tempo_NUTS_completo$tic),x)) )
tabella[c(3,4,7,8,11,12),2] <- c(sapply(ESS[,2],function(x)
  c(x*(tempo_NUTS_ridotto$toc - tempo_NUTS_ridotto$tic),x)) )
tabella[c(3,4,7,8,11,12),3] <- c(sapply(ESS[,4],function(x)
  c(x*(tempo_rwm_completo$toc - tempo_rwm_completo$tic),x)) )
tabella[c(3,4,7,8,11,12),4] <- c(sapply(ESS[,5],function(x)
  c(x*(tempo_rwm_ridotto$toc - tempo_rwm_ridotto$tic),x)) )
tabella[c(3,4,7,8,11,12),5] <- c(sapply(ESS[,3],function(x)
  c(x*(tempo_gibbs$toc - tempo_gibbs$tic),x)) )

#stampa la tabella
knitr::kable(tabella, digits = 3)

#funzione per simulare i dati discreti
simula <- function(theta,args){
  #aggiorna il vettore di probabilita'
  log_prob <- apply(args$data,1,function(x)
    sum(-theta*(x[1:3] + args$hyp[c(2,4,6)]) +
          log(theta)*(x[4:6] + args$hyp[c(1,3,5)] - 1)) - x[7])
  log_prob <- log_prob - matrixStats::logSumExp(log_prob)
  prob <- exp(log_prob)

  #campiona un indice
  idx <- sample(seq_len(NROW(args$data)),1,prob = prob)

  t1 <- args$data[idx,1]

  t2 <- args$data[idx,2] + t1

  return(c(t1,t2))
}

#simula per il NUTS
NUTS_ridotto_espanso <- t(apply(xdextract(xdtransform(NUTS_ridotto,NULL,exp),
                                  collapse = TRUE),1,simula,args = arglist_gibbs))
```

```
#simula per il RWM_M
rwm_ridotto_espanso <- t(apply(do.call(rbind,rwm_ridotto),1,simula,args = arglist_gibbs))

#trasforma l'output di DNUTS
b <- xdtransform(NUTS_completo,NULL,function(x,args){
  #primo punto di cambio
  t1 <- ceiling(args$hyp[7]+(args$hyp[8]-args$hyp[7])*plogis(x[4]))-1

  #secondo punto di cambio
  t2 <- ceiling(t1 + 1 + (args$hyp[8] - t1)*plogis(x[5])) - 1

  return(c(lambda = exp(x[1:3]) , t1 = t1, t2 = t2))
},args = arglist_completo)

#crea la tabella per i parametri discreti
tabella2 <- matrix(NA,8,5)
colnames(tabella2) <- c("DNUTS","NUTS",
                        "RWM","RWM_M","Gibbs")
rownames(tabella2) <- c(paste0(c("mean","sd","ESS","ESS/s"),"t1"),
                        paste0(c("mean","sd","ESS","ESS/s"),"t2"))

#aggiungi media e standard deviation a posteriori
tabella2[c(1,2,5,6),1] <- c(apply(xdextract(xdtransform(
  NUTS_completo,NULL,function(x,args){
    #primo punto di cambio
    t1 <- ceiling(args$hyp[7]+(args$hyp[8]-args$hyp[7])*plogis(x[4]))-1

    #secondo punto di cambio
    t2 <- ceiling(t1 + 1 + (args$hyp[8] - t1)*plogis(x[5])) - 1

    return(c(lambda = exp(x[1:3]) , t1 = t1, t2 = t2))
  },args = arglist_completo), collapse = TRUE)[,4:5] ,2,function(x) c(mean(x),sd(x))))

tabella2[c(1,2,5,6),2] <- c(apply(NUTS_ridotto_espanso,2,function(x) c(mean(x),sd(x))))

tabella2[c(1,2,5,6),3] <- c(apply(do.call(rbind,rwm_completo)[,4:5] ,2,function(x) c(mean(x),sd(x))
    ))

tabella2[c(1,2,5,6),4] <- c(apply(rwm_ridotto_espanso,2,function(x) c(mean(x),sd(x))))

tabella2[c(1,2,5,6),5] <- c(apply(do.call(rbind,gibbs)[,4:5] ,2,function(x) c(mean(x),sd(x))))

#aggiugni ESS e ESS/s
tabella2[c(3,4,7,8),1] <- c(sapply(summary(b)$stats$ESS[4:5],function(x)
  c(x,x/(tempo_NUTS_completo$toc - tempo_NUTS_completo$tic))) )

tabella2[c(3,4,7,8),2] <- c(sapply(coda::effectiveSize(NUTS_ridotto_espanso),function(x)
  c(x,x/(tempo_NUTS_ridotto$toc - tempo_NUTS_ridotto$tic)*10)) )

tabella2[c(3,4,7,8),3] <- c(sapply(coda::effectiveSize(rwm_completo)[4:5],function(x)
  c(x,x/(tempo_rwm_completo$toc - tempo_rwm_completo$tic))) )

tabella2[c(3,4,7,8),4] <- c(sapply(coda::effectiveSize(rwm_ridotto_espanso),function(x)
  c(x,x/(tempo_rwm_ridotto$toc - tempo_rwm_ridotto$tic))) )

tabella2[c(3,4,7,8),5] <- c(sapply(coda::effectiveSize(gibbs)[4:5],function(x)
  c(x,x/(tempo_gibbs$toc - tempo_gibbs$tic))) )

#stampa la tabella
knitr::kable(tabella2, digits = 3)
```