POLITECNICO DI TORINO

Corso di Laurea Magistrale
in Ingegneria Matematica

Tesi di Laurea Magistrale

Modellizzazione Bayesiana a Tempo Variabile dei Fischi-Firma

Relatore

Prof. Gianluca MASTRANTONIO

Candidato

Riccardo RACCA

Anno Accademico 2024-2025

Abstract

I fischi-firma costituiscono l'elemento centrale nel sistema comunicativo dei delfini tursiopi. Sebbene siano stati tradizionalmente considerati solo segnali di identificazione individuale, recenti studi suggeriscono la presenza di modulazioni temporali volontarie, che potrebbero veicolare informazioni aggiuntive. Alla luce di queste ipotesi, si propone un modello statistico bayesiano MCMC, in grado di gestire tali variazioni temporali come un fenomeno desiderato, e non come una manifestazione rumorosa.

Nello specifico, dopo una rassegna sulla bioacustica dei delfini (Cap. 1), vengono introdotti i principi teorici della statistica bayesiana e delle serie temporali, fondamenta dei modelli implementati (Cap. 2). Successivamente, viene formalizzato il concetto di riscalamento temporale, per modellare variazioni non-lineari nel tempo (Cap. 3). A seguire, viene descritto in modo dettagliato il modello gerarchico bayesiano, prima nella sua formalizzazione matematica, poi, nella sua implementazione (Cap. 4 e 5). Si procede con la descrizione dettagliata del processo di scelta dei modelli analizzati, mostrando le loro prestazioni su dati sintetici (Cap. 6); prima di verificare l'affidabilità dei modelli su dati reali (Cap. 7). Si conclude con l'analisi dei risultati e le possibili applicazioni (Cap. 8).

Table of Contents

Elenco delle figure

Capitolo 1

Fenomeno Naturale

1.1 I Delfini Tursiopi

I Delfini Tursìopi (Tursiops truncatus) sono una specie animale acquatica che ha suscitato l'interesse della comunità scientifica sin dalla prima metà del XX secolo. I primi lavori a riguardo consistono in report di natura descrittiva, dove si elencano alcune caratteristiche fisiche e comportamentali della specie, con considerazioni come: *"Vision and hearing are well-developed, and vocalizations are produced in the form of "jaw-snapping", whistling, and barking."* [1]. Questo fino al 1968, quando viene pubblicato: *"Vocalization of naive captive dolphins in small groups"* da parte di: Caldwell, M. C. e Caldwell, D. K. che si consacra come un lavoro pionieristico nell'ambito dello studio della vocalità dei delfini tursiopi e che sposta l'attenzione verso l'apparato comunicativo della specie. Nonostante la pubblicazione di questo articolo, per quasi due decenni furono condotte poche ricerche sul tema. Ciò fu dovuto in gran parte al fatto che i delfini non forniscono in modo evidente ed affidabile segnali visibili associati alla vocalizzazione. Pertanto, l'assenza della capacità di identificazione dell'individuo che produce un suono, requisito essenziale per lo studio di qualsiasi sistema di comunicazione animale, rappresentava un ostacolo insormontabile nello studio dei suoni [2]. Col passare degli anni, e con l'avvento di sistemi più efficienti nella rilevazione di suoni ed identificazione dell'emittente, i biologi sono riusciti a sostenere centinaia di sessioni di registrazione, con la conseguente collezione dei suoni, avendo assoluta certezza sull'esemplare emittente [2], favorendo dunque lo studio e l'indagine del fenomeno sonoro.

I motivi che hanno spinto gli scienziati a focalizzarsi su questa specie sono diversi; dal punto di vista strutturale i delfini tursiopi presentano valori promettenti di due indici comunemente usati come misura indiretta di intelligenza e capacità cognitiva, quali: "brain-to-body mass ratio" e "surface-to-volume brain ratio" [3][4]. Nel caso dei delfini tursiopi, questi indici rappresentanti le potenzialità cognitive, sono supportati da evidenti dimostrazioni di capacità avanzate. Tra queste, spiccano la creazione di complesse strutture sociali di tipo "fissione-fusione", in cui gruppi di individui si uniscono o si separano per svolgere attività altamente specifiche; la formazione di "super-alleanze" durante le fasi riproduttive; e un'elevata abilità di problem-solving, evidente in strategie sofisticate di foraggiamento come il *"Mud plume feeding"* [5][4][6].

Per comprendere come i delfini siano in grado di svolgere attività così complesse, è fondamentale esplorare il loro sofisticato apparato comunicativo. I delfini Tursiopi emettono tre tipologie di suoni su un ampio spettro di frequenze: i "clicks", segnali di breve durata a banda larga che non fanno parte della comunicazione in senso stretto, bensì servono alla ecolocalizzazione usata per la navigazione sott'acqua. Poi ci sono i "burst-pulse", che hanno la stessa natura dei clicks, ma vengono riprodotti ad un tasso più elevato, e vengono usati per trasmettere degli stati di stress-emotivo. Infine, vi sono i "fischi", denominati

”fischi-firma” nel 1968 da Caldwell M.C., Caldwell D.K., i quali sono la parte principale dell’apparato comunicativo dei delfini e su cui si è concentrata la maggior parte della produzione scientifica inerente ai suoni di questa specie. Si tratta di suoni continui a banda stretta, modulati in frequenza e possono contenere diverse armoniche. [4][7][8][9].

1.2 Il fischio-firma

Secondo quanto riportato da V. M. Janik e L. S. Sayigh in *"Communication in bottlenose dolphins: 50 years of signature whistle research"* [10], il fischio-firma viene definito come "un tipo di fischio appreso ed individualmente distintivo nel repertorio di un delfino, usato per trasmettere l’identità del proprietario del fischio". Altre fonti specificano, inoltre, che gli individui di questa specie tendono a sviluppare il proprio fischio-firma caratteristico nei primi mesi di vita, mantenendolo quasi invariato nel tempo [11][12]. Nei primi anni 2000, K. C. Buckstaff, ha indagato le caratteristiche fisiche del fenomeno, analizzando oltre 80 ore di registrazioni acustiche di 69 delfini. Da questi studi è emerso che la frequenza dei fischi distintivi varia tra 2,91 e 23,48 kHz, con una durata compresa tra 0,10 e 4,11 secondi e una media di 0,80 secondi [13]. A questo lavoro, si sono aggiunte pubblicazioni più recenti, che indicano intervalli di frequenza ancora più ampi, da 1 a oltre 30 kHz [14][15].

Oltre all’importanza semantica del fenomeno, i fischi-firma dei delfini rappresentano segnali particolarmente promettenti per lo studio dell’apprendimento vocale della specie, poiché ogni individuo sviluppa un fischio distintivo unico [11] caratterizzato da stabilità e predominanza nel suo repertorio [10]. Questo connubio permette dunque ai ricercatori di approfondire i meccanismi cognitivi alla base della comunicazione, storicizzando le informazioni.

All’atto pratico, i fischi-firma non solo servono a mantenere o ristabilire la coesione tra animali temporaneamente separati, ma vengono anche utilizzati per scambiare informazioni sull’identità quando gruppi distinti di individui si incontrano in mare, presumibilmente anche per salutarsi dopo una separazione [3]. L’importanza di questi segnali nelle interazioni sociali è evidenziata in *“Bottlenose dolphins exchange signature whistles when meeting at sea”* [16], dove si sottolinea come lo scambio di informazioni sia un elemento cruciale quando i gruppi si uniscono: solo il 10% delle unioni avviene senza scambio vocale, dimostrando così che i fischi-firma rappresentano il principale mezzo di trasmissione dell’identità.

Questa capacità di riconoscimento tra conspecifici si estende per decenni. Studi hanno dimostrato che anche delfini in cattività, separati dai loro simili per oltre dieci anni, sono in grado di riconoscere fischi-firma noti da lungo tempo [3]. Inoltre, esistono evidenze di utilizzo referenziale del fischio: la maggior parte dei delfini riproduce occasionalmente i fischi-firma di altri individui, spesso introducendo modifiche improvvise per differenziare la copia dall’originale. Questo comportamento sembra avere una funzione comunicativa specifica, in quanto viene utilizzato per attirare l’attenzione del destinatario del richiamo [16][3].

1.2.1 La struttura del fischio-firma

Come di consueto nello studio dei fischi-firma dei delfini turisiopi, quando si parla di fischio ci si riferisce alla frequenza fondamentale di esso, che può essere individuata nello spettrogramma come il contorno di frequenza più basso; mentre i multipli di questa frequenza fondamentale sono chiamate armoniche [2]. Questa semplificazione è tipica, siccome è stato mostrato che tale frequenza fondamentale è sufficiente per trasmettere l’identità dell’emittente [17]. Come indicato in *The Sarasota Dolphin Whistle Database: A unique long-term resource for understanding dolphin communication”* [2] per "tipo di fischio-firma" si indica un fischio che è stato identificato come un fischio-firma di uno specifico delfino. I vari tipi di fischi-firma possono consistere in: un singolo elemento stereotipato, chiamato "loop" o "single loop whistle", oppure in loop multipli stereotipati con o senza spazio tra loop (rispettivamente struttura sconnessa e connessa). Inoltre viene registrata grande variazione del numero di loop anche per lo stesso

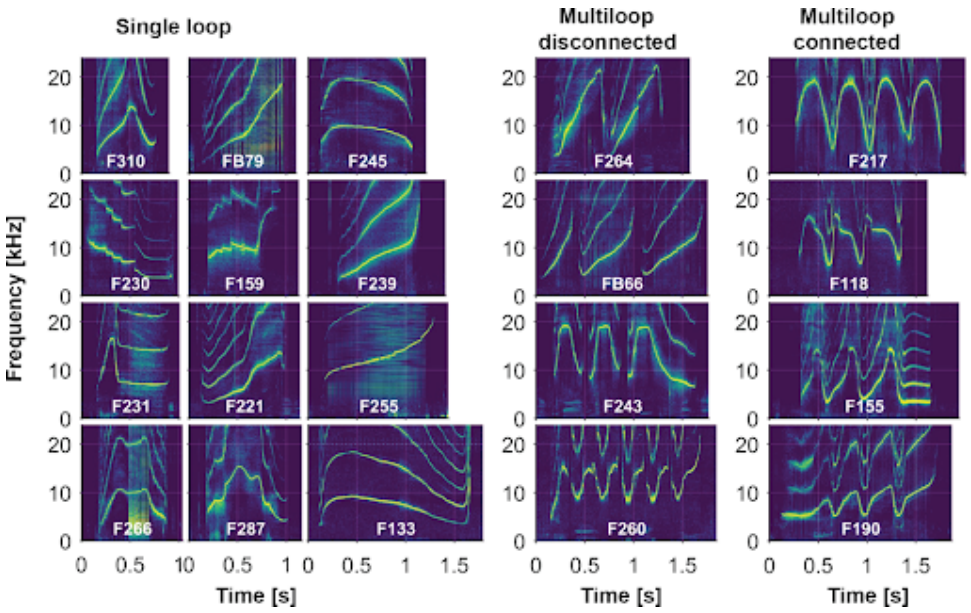


Figura 1.1: Esempio di alcuni tipi di fischi-firma del Sarasota Dataset. La prima colonna rappresenta fischi-firma a singolo loop, la seconda e terza colonna di fischi mostrano rispettivamente una struttura multi-loop sconnessa e connessa.

esemplare. In aggiunta, i contorni della frequenza fondamentale di tutti i tipi di fischi-firma possono subire troncamenti (deletions), o aggiunte (additions) a cui si sommano occasionalmente altre modifiche ad alcune caratteristiche del contorno stesso [2]. Malgrado le variazioni, l'identificazione dei fischi-firma risulta generalmente semplice, poiché i delfini tursiopi mantengono stabili alcune caratteristiche del loro fischio, modificandone solo alcune. Questo equilibrio permette ai suoni di rimanere altamente stereotipati e facilmente distinguibili [2][11]. Il giudizio umano nella distinzione e categorizzazione dei fischi-firma è molto accurato, e come riportato da [2], fino al 2022, produceva risultati migliori dei metodi computerizzati. Questo non stupisce, viste le variazioni più o meno marcate a cui i fischi-firma sono soggetti e le possibili difficoltà incontrate dagli algoritmi applicati. Un metodo citato nel 2006 da V. B. Deecke e V. M. Janik [18] ha cercato di applicare il "Dynamic Time-Warping", un algoritmo per il riconoscimento vocale automatico, che permette (in modo limitato) la compressione ed espansione sull'asse temporale dello spettrogramma, per massimizzare la somiglianza tra due suoni. Questo perché le misure di similarità applicate non sono flessibili e necessitano di standardazioni temporali, che se applicate imprudentemente, causano dilatazioni o compressioni capaci di portare due suoni a non essere più compatibili, quando in realtà in partenza lo erano. Quest'idea appena citata dà indizi su come in questa tesi verrà affrontato il trattamento dei fischi-firma e la loro modulazione; ovvero tramite riscalamento temporale.

Il repertorio sonoro dei delfini tusiopi sembra essere molto inferiore alla loro abilità comunicativa, perciò si solleva la questione di come sia possibile che riescano a mantenere una società così sofisticata, effettuando e dirigendo azioni complesse, avendo come mezzo di comunicazione principale semplicemente l'emissione del loro fischio-firma caratteristico ripetuto. Nella letteratura tradizionale si è considerato il fischio-firma come un'unità atomica, ma forse c'è altro trasmesso al suo interno [4]. A sostegno di questa ipotesi vi è un articolo del 1994 [19], dove vengono riportati i risultati di alcune analisi che dimostrano come gli aspetti contestuali possono influenzare alcune caratteristiche dei fischi-firma, suggerendo quindi che altre informazioni oltre all'identità dell'animale sono trasmesse. Questo costringe ad interrogarsi sulla natura di queste variazioni. La prima ipotesi suggerita considera questa non-uniformità dei segnali come una manifestazione di rumore biofisico. Se fosse così, però, il fenomeno variazionale si dovrebbe

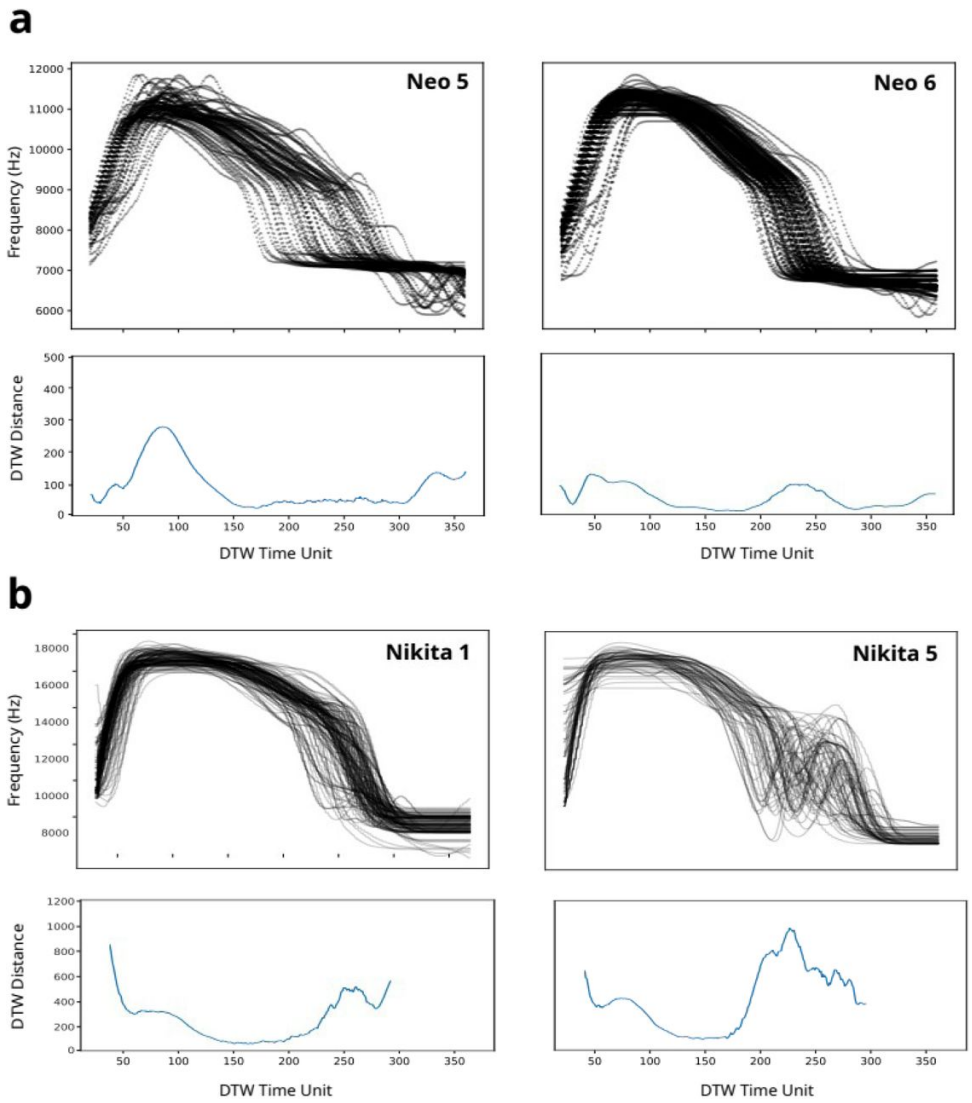


Figura 1.2: Immagine che mostra la sovrapposizione di alcuni fischi-firma degli esemplari Neo e Nikita, categorizzati attraverso un algoritmo di clustering in 4 categorie. I contorni di frequenza sono stati allineati usando il DTW, il quale è stato anche usato per determinare un "rappresentante" da cui calcolare la "DTW distance" che si vede sotto i grafici dei fischi sovrapposti.

evidenziare su tutto il segnale, mentre, invece, si osserva solamente su alcune parti ben precise.

Si può notare, infatti, come nella figura 1.2 i due raggruppamenti dei delfini Neo e Nikita presentano una modulazione stereotipata, nella misura in cui Neo modula principalmente il picco massimo del suo fischio-firma, mentre Nikita la parte finale. Se fosse frutto di rumore si dovrebbe avere discostamento ovunque. Per queste ragioni viene esclusa la prima ipotesi. La seconda ipotesi, invece, suggerisce di considerare la variabilità come legata all'identità dell'emittente, valutando questa non-uniformità quindi come una sorta di "accento" per ogni individuo. Per verificare questa ipotesi è stato etichettato manualmente un sottogruppo dei dati, colorando in maniera diversa il contorno di frequenza del fischio-firma, in base all'individuo che emetteva il suono.

Se fosse vera l'ipotesi, si dovrebbe osservare una separazione netta nelle figure, con tutte le istanze raggruppate insieme aventi un unico colore, associato a chi pronuncia il suono. Quanto atteso chiaramente non accade nella figura 1.3 e ciò fa cadere la seconda ipotesi.

Come ultima ipotesi proposta vi è la supposizione che i fischi-firma uniti alle variazioni, possano contenere altre informazioni oltre all'identità dell'emittente. Nello specifico non si considera più il fischio come un'unità atomica, ma potrebbe essere più simile al "segnale portante" delle radio, dove il contorno

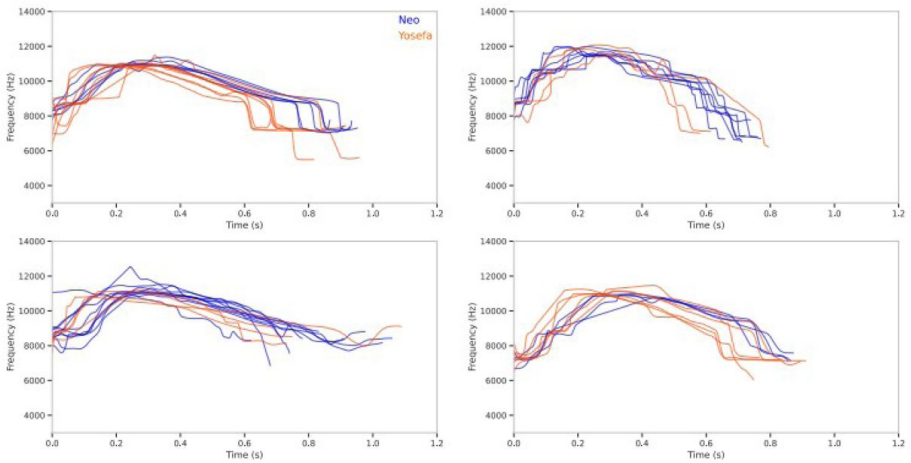


Figura 1.3: Immagine che mostra la sovrapposizione di alcuni fischi-firma, categorizzati attraverso un algoritmo di clustering in 4 categorie e colorati in modo diverso in base all'esemplare che li pronuncia. In "blu" viene mostrato il fischio emesso da Neo ed in "arancione" quello emesso da Yosefa.

distintivo della frequenza è il segnale portante, ed il modo in cui viene modulato è il mezzo con cui si trasmettono informazioni. Questo suggerisce che la modulazione non è un fenomeno casuale, bensì una parte integrante del fenomeno acustico, che necessita di una modellizzazione e che dunque giustifica la stesura di questa tesi.

1.3 I Dati

I dati analizzati nel presente studio comprendono un totale di 721 registrazioni di fischi firma emessi da delfini tursiopi. La raccolta è stata condotta tra il 2019 e il 2023 nel Mar Tirreno centrale (Mar Mediterraneo, Italia) nell'ambito di campagne di osservazione in mare su base giornaliera. Per una descrizione dettagliata della metodologia di campionamento e delle procedure adottate, si rimanda a [20] [21]. L'identificazione dei fischi firma è stata effettuata mediante il software Raven Pro 1.6, seguendo il protocollo SIGID descritto in [22]. Ogni segnale è stato analizzato attraverso l'estrazione del contorno della frequenza di picco utilizzando gli strumenti di misura forniti da Raven Pro. Successivamente, i dati spettrali così ottenuti sono stati importati e sottoposti a pre-processamento in ambiente R, al fine di eliminare eventuali valori anomali derivanti da interferenze di rumore. La rappresentazione funzionale dei contorni spettrali è stata realizzata tramite l'interpolazione con spline cubiche, implementata attraverso la funzione `smooth.spline` di R, utilizzando i parametri predefiniti del software [23].

Capitolo 2

Principi Teorici

2.1 Introduzione e Terminologia Generale

Per la modellizzazione dei fischi-firma e la stesura del modello, è adottata un'impostazione statistica di tipo bayesiano. Perciò ora viene fornita una spiegazione dettagliata degli strumenti utilizzati, in modo tale da comprendere meglio i passaggi e le finalità dell'algoritmo, nonché la sua implementazione; dimostrando anche le garanzie teoriche fondamentali dei due algoritmi usati: Gibbs Sampler e Metropolis-Hastings.

La statistica bayesiana prende il nome dal teorema di Bayes, il quale postula che date due variabili aleatorie X, Y, allora $f(x|y) = \frac{f(x,y)}{f(y)} = \frac{f(y|x)f(x)}{f(y)}$. In generale questa formula viene usata in modo iterativo cercando di fare inferenza su dei parametri $\boldsymbol{\theta}$, avendo osservato la realizzazione di una variabile aleatoria (tipicamente il vettore delle realizzazioni del fenomeno di interesse) che dipende da essi, $\boldsymbol{y}$. In questo ambito la distribuzione $f(\boldsymbol{\theta}|\boldsymbol{y})$ è chiamata distribuzione *a-posteriori* di $\boldsymbol{\theta}$ e, seguendo il postulato del teorema di Bayes, è composta dai seguenti tre elementi [24]:

- $f(\boldsymbol{y}|\boldsymbol{\theta})$: la distribuzione congiunta delle osservazioni, scritta esplicitando la dipendenza da $\boldsymbol{\theta}$. Ci si riferisce ad essa anche col nome: "verosimiglianza della a-posteriori".
- $f(\boldsymbol{\theta})$: la distribuzione *a-priori* dei parametri di interesse, i.e. l'informazione che si ha sui parametri, prima di osservare la realizzazione di $\boldsymbol{y}$.
- $f(\boldsymbol{y})$: la costante di normalizzazione, necessaria affinché valga $\int_{\Omega_\theta} f(\boldsymbol{\theta}|\boldsymbol{y})d\boldsymbol{\theta} = 1$. Siccome non dipende dai parametri su cui si fa inferenza è tipicamente trascurabile.

L'indagine bayesiana è dunque finalizzata al campionamento dalla distribuzione a-posteriori dei parametri $f(\boldsymbol{\theta}|\boldsymbol{y})$. Questo purtroppo non è immediato siccome, tipicamente, la forma della distribuzione a-posteriori non è tra quelle note, oppure risulta difficile il campionamento diretto [25]. Tuttavia, vi sono delle configurazioni della coppia "prior", "verosimiglianza" che restituiscono delle posterior in forma nota, della stessa famiglia della prior. Quando questo accade si dice che prior e verosimiglianza sono "coniugate" [26]. Non ricadere in questo caso fortuito, non risulta comunque in un problema insormontabile, infatti, come verrà mostrato dagli algoritmi Gibbs e Metropolis-Hastings, non è necessario avere esplicitamente la distribuzione a-posteriori dei parametri $\boldsymbol{\theta}$ per poter campionare da essa.

2.2 Le distribuzioni come mezzo di informazione

Come accennato con $f(\boldsymbol{\theta})$, le distribuzioni possono essere interpretate come delle informazioni sulla variabile aleatoria che descrivono. Consistenti con questa intuizione $f(\boldsymbol{\theta}|\boldsymbol{y})$ può quindi essere interpretata come "l'informazione sui parametri $\boldsymbol{\theta}$ a seguito della realizzazione di $\boldsymbol{y}$". Ragionevolmente, osservare una

variabile aleatoria che dipende dai parametri $\boldsymbol{\theta}$ permette di avere più informazioni su essi, rispetto a quante se ne avevano a-priori. Questo concetto di informazione, come si potrebbe immaginare, è iterativo e mantiene la coerenza anche quando si hanno più fasi di realizzazione e quindi più fasi di aggiornamento dell'informazione. Infatti, si ipotizzi di avere un campione x_1 di una variabile aleatoria che dipende dai parametri θ, come già detto la sua distribuzione a-posteriori sarebbe: $f(\theta|x_1) = \frac{f(x_1|\theta)f(\theta)}{f(x_1)}$. Se inoltre si ottenesse un altro campione x_2 sempre dipendente da θ allora, ragionevolmente, l'informazione fornita da questa realizzazione può andare ad aggiornare l'informazione appena appresa con x_1, infatti:

$$f(\theta|x_1, x_2) = \frac{f(x_2|x_1, \theta)f(x_1|\theta)f(\theta)}{f(x_1, x_2)} = \\ = \frac{f(x_2|x_1, \theta)f(x_1|\theta)f(\theta)}{f(x_2|x_1)f(x_1)} = \frac{f(x_2|x_1, \theta)f(\theta|x_1)}{f(x_2|x_1)}$$

Perciò la a-posteriori $f(\theta|x_1)$ può essere vista come la nuova a-priori quando si osserva un campione x_2.

I modelli statistici bayesiani, assumendo distribuzioni sulle variabili, si fondano su astrazioni matematiche della realtà. Di conseguenza questi metodi sono talvolta considerati particolarmente soggettivi, poiché dipendono dalla scelta della distribuzione a priori sui parametri θ [27]. Questo rende la scelta di tale distribuzione a-priori un argomento molto delicato. Tuttavia, in questo lavoro, vista la natura complessa del fenomeno in atto, vengono scelte prevalentemente distribuzioni a-priori di tipo: *non-informativo*, le quali, generalmente, rispecchiano una situazione in cui non si hanno a disposizione molte informazioni sui parametri, o non se ne conosce esattamente la natura. D'altro canto, quando si manifesta una conoscenza del fenomeno in atto, sufficiente a poter affermare una preferenza o una probabilità preliminare su specifici valori dei parametri, si opta per le distribuzioni a-priori *informative* [28].

2.3 Il Kernel di una Distribuzione

Si ipotizzi di avere una densità di questo tipo $f(x|y)$ dove, quindi, la variabile aleatoria x può essere scritta in funzione della variabile aleatoria y. Allora è possibile riscrivere la densità in questo modo: $f(x|y) = \frac{K(x|y)}{C(y)}$. In cui:

- $K(x|y)$ è il Kernel della distribuzione, la parte principale della funzione di densità, quella che rimane quando si tiene conto solo di x [29][30].
- $C(y)$ è invece la costante di normalizzazione, che si distingue perché non dipende dalla x [30].

Questa distinzione è comoda per notare come una distribuzione di probabilità è in realtà totalmente descritta dal suo Kernel, infatti:

$$1 = \int_\chi f(x|y)dx = \int_\chi \frac{K(x|y)}{C(y)}dx = \frac{1}{C(y)}\int_\chi K(x, y)dx \implies C(y) = \int_\chi K(x, y)dx$$

Il che implica: $f(x|y) \propto K(x|y)$.
Nelle applicazioni che vengpono mostrate in questo lavoro è necessario riconoscere il kernel di tre distribuzioni, ovvero: la normale uni- / multi-variata e la inverse-gamma. Esplicitamente:

Normale univariata

La funzione di densità della normale univariata con media μ e varianza σ^2 è data da:

$$f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

dove il termine di normalizzazione $\frac{1}{\sqrt{2\pi\sigma^2}}$ è una costante rispetto a x, quindi il kernel è:

$$\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

espandendo il quadrato:

$$(x-\mu)^2 = x^2 - 2\mu x + \mu^2$$

il kernel diventa:

$$\exp\left(-\frac{x^2 - 2\mu x + \mu^2}{2\sigma^2}\right)$$

ignorando i termini che non dipendono da x per il kernel:

$$\exp\left(-\frac{x^2 - 2\mu x}{2\sigma^2}\right)$$

Normale multivariata

La funzione di densità della normale multivariata di dimensione d, con media $\boldsymbol{\mu}$ e matrice di covarianza $\boldsymbol{\Sigma}$, è:

$$f(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)$$

Anche qui, il termine di normalizzazione è una costante rispetto a $\boldsymbol{x}$, quindi il kernel è:

$$\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)$$

Espandendo il termine al quadrato:

$$(\boldsymbol{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) = \boldsymbol{x}^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{x} - 2\boldsymbol{x}^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \boldsymbol{\mu}^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$$

Il kernel diventa:

$$\exp\left(-\frac{1}{2}\boldsymbol{x}^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \boldsymbol{x}^\top \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}\right)$$

Inverse-Gamma

La funzione di densità della variabile aleatoria inverse-gamma con parametri a e b è:

$$f(x|a,b) = \frac{b^a}{\Gamma(a)} x^{-(a+1)} \exp\left(-\frac{b}{x}\right), \quad x > 0$$

Il termine di normalizzazione $\frac{b^a}{\Gamma(a)}$ è costante rispetto a x, quindi il kernel è:

$$x^{-(a+1)} \exp\left(-\frac{b}{x}\right), \quad x > 0$$

2.4 I metodi Markov chain Monte Carlo (MCMC)

Per introdurre i due algoritmi usati in questo lavoro, è necessario definire alcune entità che permettono di dimostrare la loro convergenza. Perciò segue una breve introduzione alla teoria delle Markov Chain.

Processo Stocastico

Un **processo stocastico** è una collezione di variabili aleatorie $\{X_i\}_{i\in I}$ per qualche insieme di indici temporali I (continuo o discreto), in cui le variabili aleatorie condividono lo stesso spazio degli stati S [31][32]. Nella trattazione di questa tesi l'insieme dei tempi I sarà discreto.

Markov Chain

Una **catena di Markov** è un processo stocastico tale che la distribuzione di probabilità sullo spazio degli stati S delle realizzazioni future, considerata all'istante attuale, è completamente determinata dallo stato presente ed è indipendente da qualsiasi informazione della storia passata del processo [33].

Proprietà fondamentale delle Markov Chain del 1° ordine

I processi stocastici definiti come catene di Markov di ordine primo godono della seguente proprietà [33]:

$$\mathbb{P}(X_{j+1} = s_{j+1}|X_j = s_j, ..., X_0 = s_0) = \mathbb{P}(X_{j+1} = s_{j+1}|X_j = s_j)$$

che ha parole si traduce in: la probabilità che il processo stocastico $\{X_i\}_{i\in I}$ vada, al tempo $j+1 \in I$, nello stato s_{j+1}, sapendo tutta la storia dei passi effettuati fino al tempo j dal processo $\{X_i\}_{i\in I}$, coincide con la stessa probabilità ma avendo solamente noto lo stato attuale della catena, i.e. $X_j = s_j$. Questa proprietà induce un'altra definizione.

Probabilità di Transizione

La probabilità $\mathbb{P}(X_{j+1} = s_h|X_j = s_k)$ è definita **probabilità di transizione** dallo stato s_k al tempo $j \in I$ allo stato s_h al tempo $j+1 \in I$. La funzione che descrive la probabilità di transizione da uno stato $s_k \in S$ ad uno stato $s_h \in S$ generici al tempo $j \in I$, è denotata: $T^j(s_h|s_k)$. Se la funzione delle probabilità di transizione non dipende dal tempo, allora si parla di "caso temporalmente omogeneo" [33].

Caratterizzazione Completa di un Processo Stocastico

Un processo stocastico $\{X_i\}_{i\in I}$ si dice **caretterizzato completamente** quando sono state fornite tutte le informazioni che descrivono il comportamento probabilistico del sistema $\forall i \in I$ e lo spazio degli stati S.

Proposizione

Per caratterizzare completamente una catena di Markov è necessario conoscere:

- L'insieme degli stati S su cui è definito il processo, i.e. i valori che può assumere;
- La distribuzione iniziale $\mathbb{P}(X_0 = s) = \alpha$;
- La funzione $T^j(s_h|s_k)$ che descrive le probabilità di transizione dallo stato s_k allo stato s_h $\forall j \in I$.

Distribuzione sullo Spazio degli Stati

Si denota con π^j la **distribuzione definita sullo spazio degli stati** S, tale che, per uno stato $s \in S$ e un tempo $j \in I$, essa rappresenta la probabilità che la catena si trovi nello stato s all'istante j.

Distribuzione Stazionaria

Una **distribuzione stazionaria** π, è una distribuzione sullo spazio degli stati S tale che se $X_0 \sim \pi$ (X_0 è distribuita secondo π) allora $X_j \sim \pi$. Intuitivamente questo dice che, indipendentemente da dove il processo si trovi al tempo $j \in I$, la probabilità che il processo finisca su un qualsiasi stato generico $s \in S$ al tempo $j+1 \in I$, è identica a quella che si aveva all'istante iniziale. Questo può essere scritto come [34]:

$$\sum_{s_k \in S} \pi(s_k) T^j(s_h \mid s_k) = \pi(s_h), \quad \forall s_h \in S, \quad j \in I$$

Detailed Balanced

Una distribuzione è detta **detailed balanced** se $\forall s_h, s_k \in S$ e $\forall j \in I$ vale [34]:

$$\pi(s_k) T^j(s_h|s_k) = \pi(s_h) T^j(s_k|s_h)$$

Questo significa che il flusso "in entrata" nello stato s_h (cioè la probabilità di essere in s_k moltiplicata per la probabilità di transizione verso s_h) è uguale al flusso "in uscita" dallo stato s_h verso s_k.

Proposizione
Una distribuzione di probabilità che rispetta il detailed balanced è una distribuzione stazionaria. Ovvero π detailed balanced $\implies$ π stazionaria [34].
Dimostrazione:
Partendo dalla definizione di detailed balanced:

$$\pi(s_k) T^j(s_h|s_k) = \pi(s_h) T^j(s_k|s_h), \quad \forall s_h, s_k \in S, \quad \forall j \in I$$

Sommando sullo spazio degli stati, indicizzati da s_k:

$$\sum_{s_k \in S} \pi(s_k) T^j(s_h|s_k) = \sum_{s_k \in S} \pi(s_h) T^j(s_k|s_h)$$

Siccome $\pi(s_h)$ non dipende da s_k può essere portato fuori dalla sommatoria:

$$\sum_{s_k \in S} \pi(s_k) T^j(s_h|s_k) = \pi(s_h) \sum_{s_k \in S} T^j(s_k|s_h)$$

Infine, visto che $T^j(s_k|s_h)$ rappresenta la funzione delle probabilità di transizione e siccome si sta sommando su tutti gli stati di "arrivo" partendo dallo stesso stato s_h, allora vale:

$$\sum_{s_k \in S} T^j(s_k|s_h) = 1$$

Che conduce a:

$$\sum_{s_k \in S} \pi(s_k) T^j(s_h|s_k) = \pi(s_h), \quad \forall s_h \in S, \quad j \in I$$

Ovvero la definizione di distribuzione stazionaria nel caso discreto. □

La Teoria nelle Applicazioni

Le definizioni sopracitate forniscono formalità alla teoria che viene usata, ma nella successiva applicazione degli algoritmi non è rilevate quale sia la distribuzione iniziale. Inoltre la funzione che descrive la probabilità di transizione non dipende dall'istante temporale, ovvero, ci si riduce al caso temporalmente

Algorithm 1 Gibbs Sampler

Input: Il vettore delle realizzazioni $\boldsymbol{\psi}$, lo stato iniziale della catena $\boldsymbol{x}^0 = x_1^0, x_2^0, ..., x_p^0$ ed il numero di iterazioni B.
Output: La catena di campioni, dipendenti, dalla distribuzione a-posteriori $\{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^B\}$.
Inizializzazione: $\boldsymbol{x}^0$ e $b \leftarrow 0$
while $b < B$ **do**
  $b \leftarrow b + 1$
  for $i = 1$ to p **do**
    Campiona $x_i^b \sim X_i^b \mid x_1^b, \ldots, x_{i-1}^b, x_{i+1}^{b-1}, \ldots, x_p^{b-1}, \boldsymbol{\psi}$
  end for
  Salva il campione: $\boldsymbol{x}^b \leftarrow (x_1^b, \ldots, x_p^b)$
end while
return $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^B$, B-campioni dalla a-posteriori.

omogeneo. L'idea alla base degli algoritmi è quella di specificare una funzione per le probabilità di transizione $T(.|.)$ in modo tale che i campioni della catena, una volta raggiunta la convergenza, provengano da una specifica distribuzione stazionaria, la quale deve coincidere con la distribuzione di probabilità a-posteriori da cui si desidera campionare. Il metodo con cui gli algoritmi garantiscono che la distribuzione a-posteriori è stazionaria avviene mediante il soddisfacimento della proprietà di detailed balanced, la quale viene fatta rispettare scegliendo delle opportune funzioni per le probabilità di transizione.

Gibbs Sampler

Si consideri il problema di voler generare campioni da una densità $f_{\boldsymbol{X}}(\boldsymbol{x} \mid \boldsymbol{\psi})$, dove il vettore $\boldsymbol{x} = (x_1, \ldots, x_p)$ rappresenta un insieme di parametri di interesse e $\boldsymbol{\psi}$ è il vettore di realizzazioni. L'algoritmo Gibbs Sampler afferma che, mediante un campionamento iterativo dalla distribuzione full-conditional delle singole variabili aleatorie x_i, i.e. dalla distribuzione di x_i condizionata a tutte le altre variabili aleatorie $\boldsymbol{x_{-1}}$ e $\boldsymbol{\psi}$, dopo essere giunti a convergenza, si riesce ad ottenere un vettore di campioni dipendenti dalla distribuzione a-posteriori dei parametri, $f_{\boldsymbol{X}}(\boldsymbol{x} \mid \boldsymbol{\psi})$ [25]. Questo approccio si traduce nell'algoritmo 1 [35]. La legittimità dell'algoritmo Gibbs sampler si dimostra attraverso la costruzione di una catena di Markov, in cui l'aggiornamento avviene tramite l'utilizzo delle full-conditional. Questa catena converge alla distribuzione a-posteriori, che si rivela stazionaria, soddisfacendo la condizione di detailed balance.

Detailed Balanced

Per semplicità notazionale si assuma di avere solamente due variabili e di voler campionare da: $f(x_1, x_2|\boldsymbol{\psi})$, che per brevità verrà denotata $f(x_1, x_2)$. Per ottenere il campione b-esimo, dato il precedente, si utilizza la procedura con le full-conditional:

- Campionamento di x_1^b dato $x_1^{b-1} \mid x_2^{b-1}$
- Campionamento di x_2^b dato $x_2^{b-1} \mid x_1^b$

Si indica con $f(x_1, x_2)$ la distribuzione a-posteriori, che, nella teoria delle Markov chain, veniva rappresentata dalla distribuzione sullo spazio degli stati π; mentre la funzione delle probabilità di transizione è sempre denotata come:

$$T(x_1^b, x_2^b | x_1^{b-1}, x_2^{b-1})$$

Ora si deve dimostrare che questi due passaggi rispettano il detailed balanced. Innanzitutto il passaggio nello spazio degli stati:

$$(x_1^{b-1}, x_2^{b-1}) \rightarrow (x_1^b, x_2^b)$$

può essere scomposto in

$$(x_1^{b-1}, x_2^{b-1}) \rightarrow (x_1^b, x_2^{b-1}) \rightarrow (x_1^b, x_2^b),$$

da cui ne consegue che le funzione delle probabilità di transizione si suddivide come segue:

$$T(x_1^b, x_2^b \mid x_1^{b-1}, x_2^{b-1}) = T_1(x_1^b, x_2^{b-1} \mid x_1^{b-1}, x_2^{b-1}) T_2(x_1^b, x_2^b \mid x_1^b, x_2^{b-1}).$$

A questo punto la condizione di datiled balanced diventa:

$$f(x_1^{b-1}, x_2^{b-1}) T_1(x_1^b, x_2^{b-1} \mid x_1^{b-1}, x_2^{b-1}) = f(x_1^b, x_2^{b-1}) T_1(x_1^{b-1}, x_2^{b-1} \mid x_1^b, x_2^{b-1}) \tag{2.1}$$

$$f(x_1^b, x_2^{b-1}) T_2(x_1^b, x_2^b \mid x_1^b, x_2^{b-1}) = f(x_1^b, x_2^b) T_2(x_1^b, x_2^{b-1} \mid x_1^b, x_2^b) \tag{2.2}$$

Ora si impone l'aggiornamento full-conditional che definisce il Gibbs Sampler:

$$\begin{aligned} T_1(x_1^b, x_2^{b-1} \mid x_1^{b-1}, x_2^{b-1}) &= f(x_1^b \mid x_2^{b-1}) \\ T_2(x_1^b, x_2^b \mid x_1^b, x_2^{b-1}) &= f(x_2^b \mid x_1^b) \end{aligned}$$

La transizione da uno stato (b-1)-esimo allo stato b-esimo diventa:

$$\begin{aligned} T(x_1^b, x_2^b \mid x_1^{b-1}, x_2^{b-1}) &= T_1(x_1^b, x_2^{b-1} \mid x_1^{b-1}, x_2^{b-1}) T_2(x_1^b, x_2^b \mid x_1^b, x_2^{b-1}) \\ &= f(x_1^b \mid x_2^{b-1}) f(x_2^b \mid x_1^b). \end{aligned}$$

La dimostrazione della prima uguaglianza del detailed (2.1) è:

$$\begin{aligned} &f(x_1^{b-1}, x_2^{b-1}) T_1(x_1^b, x_2^{b-1} \mid x_1^{b-1}, x_2^{b-1}) = f(x_1^{b-1}, x_2^{b-1}) f(x_1^b \mid x_2^{b-1}) = \\ &= f(x_1^{b-1} \mid x_2^{b-1}) f(x_2^{b-1}) f(x_1^b \mid x_2^{b-1}) = f(x_1^{b-1} \mid x_2^{b-1}) f(x_1^b, x_2^{b-1}) = \\ &= f(x_1^b, x_2^{b-1}) T_1(x_1^{b-1}, x_2^{b-1} \mid x_1^b, x_2^{b-1}) \end{aligned}$$

In modo analogo (2.2):

$$\begin{aligned} &f(x_1^b, x_2^{b-1}) T_2(x_1^b, x_2^b \mid x_1^b, x_2^{b-1}) = f(x_1^b, x_2^{b-1}) f(x_2^b \mid x_1^b) = \\ &= f(x_2^{b-1} \mid x_1^b) f(x_1^b) f(x_2^b \mid x_1^b) = f(x_2^{b-1} \mid x_1^b) f(x_1^b, x_2^b) = \\ &= f(x_1^b, x_2^b) T_2(x_1^b, x_2^{b-1} \mid x_1^b, x_2^b) \end{aligned}$$

Come volevasi dimostrare. □

La dimostrazione appena svolta si può generalizzare al caso n-variato, definendo di conseguenza le n-funzioni di transizione necessarie e conservando la stessa logica iterativa.

Metropolis-Hastings Algorithm

Il Gibbs sampler non è sempre applicabile, poiché richiede la derivazione delle full-conditional per ogni variabile o gruppo di variabili nel modello. Questo significa che tali distribuzioni devono essere espresse in forma chiusa o essere facilmente campionabili, ma ciò non è sempre possibile. Inoltre, anche quando le full-conditional sono note, l'algoritmo può risultare inefficiente in presenza di variabili fortemente correlate. In questi casi, il Gibbs sampler necessita di molti passi per raggiungere la convergenza, poiché l'aggiornamento delle variabili una alla volta rallenta l'esplorazione dello spazio delle soluzioni. Per risolvere alcune di queste problematiche viene presentato l'algoritmo Metropolis-Hastings.

Con il termine algoritmo di Metropolis-Hastings si indica una famiglia di metodi di simulazione Markov chain usati per il campionamento dalla distribuzione posterior. Questa famiglia comprende anche il Gibbs sampler, che può essere considerato un caso particolare del Metropolis-Hastings. L'algoritmo Metropolis-Hastings è un'adattazione ad un *random walk* con meccanismo di accettazione/rifiuto che permette di convergere alla distribuzione desiderata [27] e viene implementato nell'algoritmo 2.

Algorithm 2 Metropolis-Hastings

Input: Il vettore delle realizzazioni $\boldsymbol{\psi}$, lo stato iniziale della catena $\boldsymbol{x}^0$, una funzione di densità $q(.)$ chiamata *proposta* ed il numero di iterazioni B.
Output: La catena di campioni, dipendenti, dalla distribuzione a-posteriori $\{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^B\}$.
Inizializzazione: $\boldsymbol{x}^0$ e $b \leftarrow 0$
while $b < B$ **do**
  $b \leftarrow b + 1$
  $\boldsymbol{x}^* \sim q(\boldsymbol{x}^*|\boldsymbol{x}^{b-1}, .)$
  $\alpha(\boldsymbol{x}^*, \boldsymbol{x}) \leftarrow \min\left\{\frac{f_{\boldsymbol{X}}(\boldsymbol{x}^*|\boldsymbol{\psi})q(\boldsymbol{x}^{b-1}|\boldsymbol{x}^*,.)}{f_{\boldsymbol{X}}(\boldsymbol{x}^{b-1}|\boldsymbol{\psi})q(\boldsymbol{x}^*|\boldsymbol{x}^{b-1},.)}, 1\right\}$
  $u \sim U(0,1)$
  if $u \leq \alpha$ **then**
    $\boldsymbol{x}^{(b)} \leftarrow \boldsymbol{x}^*$
  else
    $\boldsymbol{x}^b \leftarrow \boldsymbol{x}^{b-1}$
  end if
end while
return $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^B$, B-campioni dalla a-posteriori.

L'algortimo Metropolis-Hastings introduce il concetto fondamentale di *distribuzione proposta* [28] sui parametri $\boldsymbol{x}$, indicata con $q(.|.)$, che viene utilizzata per esplorare lo spazio degli stati, generando nuovi candidati che vengono accettati o rifiutati sulla base di un criterio di accettazione. Il criterio di accettazione vede una prima fase di selezione del minimo tra il cosiddetto *rapporto Metropolis* ed il valore 1, il minimo tra questi due valori è chiamato *acceptance ratio* e viene indicato con $\alpha(\boldsymbol{x}^*, \boldsymbol{x})$ (riga 7 dell'algoritmo 2). Nella seconda fase si campiona una variabile aleatoria $u \sim U(0,1)$. Se la variabile u campionata soddisfa la disuguaglianza $u \leq \alpha(\boldsymbol{x}^*, \boldsymbol{x})$ allora si accetta la proposta, altrimenti si rifiuta. La distribuzione $q(.|.)$ viene lasciata volutamente con uno spazio finale nell'algoritmo 2, per fare intendere che la proposta può dipendere da vari fattori.

Detailed Balanced

Anche in questo caso la legittimità dell'algoritmo si dimostra attraverso la costruzione di una catena di Markov convergente alla distribuzione a-posteriori. Come per il Gibbs sampler si limita la dimostrazione al caso bi-variato: x_1, x_2, concentrandosi esclusivamente sul campionamento di x_1. L'obiettivo rimane quello di campionare da $f(x_1, x_2|\boldsymbol{\psi})$, che per brevità viene denotata $f(x_1, x_2)$. La condizione di detailed balanced per x_1 è:

$$f(x_1^{b-1}, x_2^{b-1})T_1(x_1^b, x_2^{b-1}|x_1^{b-1}, x_2^{b-1}) = f(x_1^b, x_2^{b-1})T_1(x_1^{b-1}, x_2^{b-1}|x_1^b, x_2^{b-1}) \tag{2.3}$$

dove in questo caso vale:

$$T_1(x_1^b, x_2^{b-1}|x_1^{b-1}, x_2^{b-1}) = q(x_1^b|x_1^{b-1}, x_2^{b-1})\alpha(x_1^b, x_1^{b-1})$$

con $q(.)$ la distribuzione proposta e $\alpha(x_1^b, x_1^{b-1})$ il rapporto Metropolis. Sostituendo in 2.3 si ottiene la nuova riformulazione di detailed balanced:

$$f(x_1^{b-1}, x_2^{b-1})q(x_1^b|x_1^{b-1}, x_2^{b-1})\alpha(x_1^b, x_1^{b-1}) = f(x_1^b, x_2^{b-1})q(x_1^{b-1}|x_1^b, x_2^{b-1})\alpha(x_1^{b-1}, x_1^b) \tag{2.4}$$

Si noti, inoltre, come sicuramente un termine tra: $\alpha(x_1^b, x_1^{b-1})$ e $\alpha(x_1^{b-1}, x_1^b)$ è unitario, mentre l'altro coincide col rapporto Metropolis nella definizione dell'acceptance ratio, infatti:

$$\alpha(x_1^b, x_1^{b-1}) = min\left\{\frac{f(x_1^b|x_2^{b-1})q(x_1^{b-1}|x_1^b, x_2^{b-1})}{f(x_1^{b-1}|x_2^{b-1})q(x_1^b|x_1^{b-1}, x_2^{b-1})}, 1\right\} = min\{r, 1\}$$

$$\alpha(x_1^{b-1}, x_1^b) = min\left\{\frac{f(x_1^{b-1}|x_2^{b-1})q(x_1^b|x_1^{b-1}, x_2^{b-1})}{f(x_1^b|x_2^{b-1})q(x_1^{b-1}|x_1^b, x_2^{b-1})}, 1\right\} = min\left\{\frac{1}{r}, 1\right\}$$

si ipotizzi $\alpha(x_1^{b-1}, x_1^b) = 1$ e $\alpha(x_1^b, x_1^{b-1}) = r$, allora il seguente rapporto è ben definito:

$$\frac{\alpha(x_1^b, x_1^{b-1})}{\alpha(x_1^{b-1}, x_1^b)} = \frac{f(x_1^b|x_2^{b-1})q(x_1^{b-1}|x_1^b, x_2^{b-1})}{f(x_1^{b-1}|x_2^{b-1})q(x_1^b|x_1^{b-1}, x_2^{b-1})} \tag{2.5}$$

Ma questo implica il soddisfacimento del detailed balanced, infatti, moltiplicando ambo i membri dell'equazione 2.5 per i due denominatori, si ottiene:

$$f(x_1^{b-1}|x_2^{b-1})q(x_1^b|x_1^{b-1}, x_2^{b-1})\alpha(x_1^b, x_1^{b-1}) = f(x_1^b|x_2^{b-1})q(x_1^{b-1}|x_1^b, x_2^{b-1})\alpha(x_1^{b-1}, x_1^b)$$

che coincide con l'equazione 2.4, e questo è sufficiente per concludere la dimostrazione, precisando che per x_2 i calcoli sono analoghi. □

Riformulazione del Rapporto Metropolis

L'acceptance ratio nell'algoritmo 2, ha la seguente struttura:

$$\min\left\{\frac{f_{\boldsymbol{X}}(\boldsymbol{x}^* \mid \boldsymbol{\psi})q(\boldsymbol{x}^{b-1}|\boldsymbol{x}^*, .)}{f_{\boldsymbol{X}}(\boldsymbol{x}^{b-1} \mid \boldsymbol{\psi})q(\boldsymbol{x}^*|\boldsymbol{x}^{b-1}, .)}, 1\right\}$$

Dove sia a numeratore che a denominatore viene usata la distribuzione full-conditional congiunta. Nel caso in cui non fosse possibile ottenere esplicitamente quella densità è possibile riformulare il rapporto usando la distribuzione congiunta, $f(\boldsymbol{x}, \boldsymbol{\psi})$ invece della condizionata, $f_{\boldsymbol{X}}(\boldsymbol{x} \mid \boldsymbol{\psi})$, infatti nota la seguente relazione tra probabilità condizionate e congiunta:

$$f(\boldsymbol{x}, \boldsymbol{\psi}) = f_{\boldsymbol{X}}(\boldsymbol{x} \mid \boldsymbol{\psi})f_{\boldsymbol{\Psi}}(\boldsymbol{\psi})$$

se si sostituisce la congiunta alla condizionata nel rapporto Metropolis si ottiene:

$$\frac{f(\boldsymbol{x}^*, \boldsymbol{\psi})q(\boldsymbol{x}^{b-1}|\boldsymbol{x}^*, .)}{f(\boldsymbol{x}^{b-1}, \boldsymbol{\psi})q(\boldsymbol{x}^*|\boldsymbol{x}^{b-1}, .)} = \frac{f_{\boldsymbol{X}}(\boldsymbol{x}^* \mid \boldsymbol{\psi})f_{\boldsymbol{\Psi}}(\boldsymbol{\psi})q(\boldsymbol{x}^{b-1}|\boldsymbol{x}^*, .)}{f_{\boldsymbol{X}}(\boldsymbol{x}^{b-1} \mid \boldsymbol{\psi})f_{\boldsymbol{\Psi}}(\boldsymbol{\psi})q(\boldsymbol{x}^*|\boldsymbol{x}^{b-1}, .)}$$

che coincide con il rapporto Metropolis dopo la semplificazione del termine $f_{\boldsymbol{\Psi}}(\boldsymbol{\psi})$ tra numeratore e denominatore.

Metropolis-Within-Gibbs

I due algoritmi analizzati possono essere usati congiuntamente per sopperire alle situazioni in cui, ad esempio, non si è in grado di ottenere la distribuzione full-conditional esplicita per un gruppo di parametri, ma si vuole mantenere comunque l'impostazione iterativa dell'algoritmo Gibbs sampler, senza usare un puro algoritmo Metropolis-Hastings di cui magari è difficile scovare una distribuzione proposta opportuna. In questi casi è possibile quindi mantenere l'algoritmo Gibbs, inserendo dei *passi Metropolis* per campionare singole o multiple variabili aleatorie [27]. Uno pseudo-codice generico che illustra l'idea, in uno scenario in cui la j-esima componente del vettore dei coefficienti $\boldsymbol{x} = x_1, x_2, ..., x_p$, viene proposta con un'impostazione Metropolis-Hastings nell'algoritmo 3.

2.5 Le Serie Temporali

Il concetto di serie temporale è cruciale nella modellizzazione scelta per i fischi-firma in questa tesi, perciò segue una breve trattazione delle nozioni che verranno sfruttate nella stesura del modello.

Algorithm 3 Metropolis-Within-Gibbs

Input: Il vettore delle realizzazioni $\boldsymbol{\psi}$, lo stato iniziale della catena $\boldsymbol{x}^0 = x_1^0, x_2^0, ..., x_p^0$ ed il numero di iterazioni B.
Output: La catena di campioni, dipendenti, dalla distribuzione a-posteriori $\{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^B\}$.
Inizializzazione: $\boldsymbol{x}^0$ e $b \leftarrow 0$
while $b < B$ **do**
  $b \leftarrow b + 1$
  for $i = 1$ to p **do**
    if $i == j$ **then**
      $x_j^* \sim q(x_j^* | x_j^{b-1}, .)$
      $\alpha(x_j^*, x_j) \leftarrow \min\left\{ \frac{f_{X_j}(x_j^* | \boldsymbol{\psi}) q(x_j^{b-1} | x_j^*, .)}{f_{X_j}(x_j^{b-1} | \boldsymbol{\psi}) q(x_j^* | x_j^{b-1}, .)}, 1 \right\}$
      $u \sim U(0,1)$
      if $u \leq \alpha$ **then**
        $x_j^{(b)} \leftarrow x_j^*$
      else
        $x_j^b \leftarrow x_j^{b-1}$
      end if
    else
      Campiona $x_i^b \sim X_i^b \mid x_1^b, \ldots, x_{i-1}^b, x_{i+1}^{b-1}, \ldots, x_p^{b-1}, \boldsymbol{\psi}$
    end if
  end for
  Salva il campione: $\boldsymbol{x}^b \leftarrow (x_1^b, \ldots, x_p^b)$
end while
return $\boldsymbol{x}^1, \ldots, \boldsymbol{x}^B$, B-campioni dalla a-posteriori.

Serie Temporale

Per **serie temporale** si intende una realizzazione di un insieme di variabili casuali, indicizzate in base all'ordine in cui vengono ottenute nel tempo [36]. Siccome, in generale, una collezione di variabili aleatorie $\{X_i\}_{i \in I}$ per qualche insieme di indici temporali I è definito come un processo stocastico, si può definire la serie temporale come la realizzazione di un processo stocastico [37].

Stazionarietà Forte

Un processo stocastico si dice **stazionario in senso forte** se $\forall n \in \mathbb{N}$, ogni n-upla $(t_1, \ldots, t_n) \in I$ e $\forall h : t_j + h \in I \ \forall j = 1, 2, \ldots, n$ vale:

$$\mathbb{P}(X_{t_1} = x_1, X_{t_2} = x_2, \ldots, X_{t_n} = x_n) = \mathbb{P}(X_{t_1+h} = x_1, X_{t_2+h} = x_2, \ldots, X_{t_n+h} = x_n) \tag{2.6}$$

Quindi 2.6 afferma che, anche traslando il processo temporale di un tempo h, la distribuzione (il carattere) del processo non cambia. In altre parole, se una serie temporale è stazionaria in senso forte, allora tutte le funzioni di densità multivariata, per un qualsiasi sottoinsieme di variabili, devono essere identiche rispetto alle loro controparti nell'insieme traslato, per tutti i valori del parametro di traslazione h [36] [37].

Stazionarietà Debole

Un processo stocastico si dice **stazionario in senso debole** di ordine $m \in \mathbb{N}$ se $\forall\ n \in \mathbb{N} : n \leq m$, ogni n-upla $(t_1, \ldots, t_n) \in I$ e $\forall h : t_j + h \in I \ \forall j = 1, 2, \ldots, n$ vale:

$$\mathbb{E}(X_{t_1} X_{t_2} \ldots X_{t_n}) = \mathbb{E}(X_{t_1+h} X_{t_2+h} \ldots X_{t_n+h}) \tag{2.7}$$

[37] definisce la stazionarietà debole di ordine 2, ma la generalizzazione ad m è immediata.

Processo Gaussiano

Un processo stocastico, $\{X_i\}_{i\in I}$, si dice **processo gaussiano** se i vettori n-dimensionali: $\mathbf{X} = (X_{t_1}, X_{t_2}, \ldots, X_{t_n})'$, per ogni insieme di punti temporali $t_1, t_2, \ldots, t_n \in I$, seguono una distribuzione normale multivariata [36].

Caratterizzazione di un Processo Gaussiano

Dato un vettore aleatorio $\boldsymbol{X} \sim N_n(\boldsymbol{\mu}, \Sigma)$, distribuito secondo una normale multi-variata, ogni suo sotto-vettore di dimensione $p \leq n$, $\boldsymbol{X}_p$ soddisfa: $\boldsymbol{X}_p \sim N_p(\boldsymbol{\mu}_p, \Sigma_p)$, con $\boldsymbol{\mu}_p$ dato dalle p-entrate corrispondenti del vettore delle medie $\boldsymbol{\mu}$, e matrice di varianza e covarianza Σ_p data dalla matrice ottenuta selezionando solamente le righe e le colonne di Σ corrispondenti alle p-entrate degli elementi di $\boldsymbol{X}_p$. Questo implica che tutte le variabili aleatorie X_i selezionate dal vettore $\boldsymbol{X}$ sono distribuite marginalmente secondo delle normali uni-variate di media μ_i e varianza $\psi_i^2 := \Sigma(i,i)$. Se si considera il vettore X come una processo stocastico, i.e. $\boldsymbol{X} = \{X_i\}_{i\in I}$ processo stocastico gaussiano, allora quanto appena enunciato garantisce, grazie alla definizione di caratterizzazione di processo stocastico, che per caratterizzare completamente un processo stocastico gaussiano sono sufficienti la funzione di media e di varianza-covarianza sull'insieme dei tempi I.

Proposizione

Se una serie temporale gaussiana, $\{X_i\}_{i\in I}$, è debolmente stazionaria di ordine 2, allora $\mu_t = \mu$ e $\gamma(t_1, t_2) = \gamma(|t_1 - t_2|) \quad \forall t, t_1, t_2 \in I$; ovvero, il vettore $\boldsymbol{\mu}$ e la matrice Γ sono indipendenti dal tempo. Questi fatti implicano che tutte le distribuzioni finite, del processo $\{X_i\}_{i\in I}$ dipendono solo dall'intervallo di tempo e non dai tempi effettivi, e quindi il processo stocastico deve essere strettamente stazionario [36].
Dimostrazione:
Dato un qualsiasi processo stocastico $\{X_i\}_{i\in I}$ che sia stazionario debolmente di ordine 2, vale per definizione:

$$\mathbb{E}(X_t) = \mathbb{E}(X_{t+h})$$
$$\mathbb{E}(X_t X_{t+l}) = \mathbb{E}(X_{t+h} X_{t+l+h}),\ \forall t \in I,\ l, h \in \mathbb{N} : t+h, t+l, t+h+l \in I$$

Allora la media è costante in ogni istante, mentre il momento secondo $\mathbb{E}(X_t X_{t+l})$ dipende solamente dal *lag*-temporale. Perciò, data la formula della covarianza tra due elementi dello stesso processo stocastico:

$$Cov(X_t, X_{t+l}) = \mathbb{E}(X_t X_{t+l}) - \mathbb{E}(X_t)\mathbb{E}(X_{t+l})$$

solamente il termine $\mathbb{E}(X_t X_{t+l})$ è una variabile e questo implica che anche la funzione di covarianza tra due variabili aleatorie dello stesso processo dipende solamente dal lag-temporale: $Cov(X_t, X_{t+l}) =: \gamma(l)$. Questo è sufficiente a concludere la dimostrazione, perché: preso un vettore di dimensione finita distribuito secondo una normale multi-variata rappresentante un processo stocastico debolmente stazionario di ordine 2: $(X_{t_1}, X_{t_2}, ..., X_{t_n})$ allora il loro vettore di media ha tutte le entrate con lo stesso valore, mentre la matrice di varianza e covarianza dipende esclusivamente dal lag-temporale tra le entrate, che qualunque esso sia, sarà identico a quello che avranno le stesse variabili aleatorie traslate di una componente $h \in \mathbb{N}$, i.e. $(X_{t_1+h}, X_{t_2+h}, X_{t_n+h})$. □

Autocorrelazione

Una statistica di interesse nel caso delle serie temporali è l'**autocorrelazione**, che si calcola esattamente come la correlazione tra due variabili aleatorie generiche, ma siccome le variabili in ingresso sono provenienti dalla stessa serie temporale, allora viene aggiunto il prefisso "auto-". Nel caso di interesse di processi stocastici stazionari, l'autocorrelazione dipende unicamente dal lag-temporale che distanzia le due variabili

aleatorie del processo; la formula è [37]:

$$\rho(l) = \frac{Cov(X_t, X_{t+l})}{Var(X_t)} = \frac{\gamma(l)}{\gamma(0)}$$

Questa statistica verrà usata per analizzare la dipendenza tra i valori delle catene campionate dai metodi MCMC.

Il Correlogramma

Il **correlogramma** è la rappresentazione grafica della autocorrelazione di una serie temporale [38], questa trova sull'asse delle ascisse il distanziamento temporale, i.e. i lag; sull'asse delle ordinate il valore corrispondente di autocorrelazione, insieme ad una soglia di rilevanza, sotto la quale si assume che l'autocorrelazione sia non-rilevante. Ogni punto del grafico mostra quanto la serie temporale è correlata con sé stessa ad un dato lag. Il correlogramma è utile nello capire se una serie temporale è stazionaria, in tal caso l'autocorrelazione dovrebbe diminuire all'aumentare del lag, fin sotto il valore di soglia.

Il Modello Autoregressivo

Il **modello autoregressivo** (AR) si fonda sull'assunto che l'andamento passato di una variabile contenga informazioni utili per prevederne i valori futuri. In particolare, esso descrive un processo in cui il valore assunto al tempo t è funzione dei valori passati del medesimo processo [39]. Si definisce **modello autoregressivo di ordine 1** (AR(1)) un processo in cui, condizionatamente al valore al tempo $t-1$, il valore al tempo t risulta indipendente da tutti gli altri valori passati. Formalmente con $\{X_i\}_{i\in I}$ e $t \in I$ vale:

$$X_t = \alpha X_{t-1} + \epsilon_t,\ \epsilon_t \sim N(0, \psi^2)$$

Assumendo la serie temporale di interesse "sufficientemente lunga" è possibile ricavare approssimativamente alcune caratteristiche interessanti del processo, se si pone la condizione $\mid \alpha \mid < 1$. Ad esempio è agevole il calcolo della distribuzione marginale, di un punto "sufficientemente lontano" dal principio della serie:

$$X_t = \alpha(\alpha X_{t-2} + \epsilon_{t-1}) + \epsilon_t = \cdots = \alpha^{+\infty} X_{t-\infty} \sum_{i=0}^{+\infty} \alpha^i \epsilon_{t-i}$$

chiaramente è un'approssimazione quella di usare $+\infty$, velata dal termine "sufficientemente lontano". Questa catena di uguaglianze, unito alla condizione 1 permette di trascurare il primo termine fuori dalla sommatoria, ottenendo:

$$X_t = \sum_{i=0}^{+\infty} \alpha^i \epsilon_{t-i}$$

Ora si può determinare la distribuzione marginale degli elementi del processo. Innanzitutto, siccome il processo $\{X_i\}_{i\in I}$ è una combinazione lineare di un processo gaussiano, $\epsilon_t \sim N(0, \psi^2)$, $\epsilon_t \perp\!\!\!\perp \epsilon_{t'}\ \forall t, t' \in I$ è anch'esso un processo gaussiano, perciò per definirlo completamente è necessario determinare la media e la funzione di varianza-covarianza. La media è banalmente nulla, essendo $X_t\ \forall t \in I$ somma di variabili

aleatorie a media nulla. Segue il calcolo della covarianza tra X_t e X_{t+h}:

$$\begin{aligned}\gamma_t(h) &= Cov(X_t, X_{t+h}) = Cov(\sum_{i=0}^{+\infty} \alpha^i \epsilon_{t-i}, \sum_{j=0}^{+\infty} \alpha^j \epsilon_{t+h-j}) \\ &= \sum_{i=0}^{+\infty}\sum_{j=0}^{+\infty} \alpha^i\alpha^j Cov(\epsilon_{t-i}, \epsilon_{t+h-j}) = \sum_{i=0, j=k+1}^{+\infty} \alpha^i\alpha^j Cov(\epsilon_{t-i}, \epsilon_{t+h-(h+i)}) \\ &= \psi^2 \sum_{i=0}^{+\infty} \alpha^i\alpha^{k+i} = \alpha^k\psi^2 \sum_{i=0}^{+\infty} \alpha^{2i}\end{aligned}$$

da cui si ottiene, grazie alla convergenze delle serie geometriche:

$$\gamma_t(h) = \frac{\alpha^h\psi^2}{1-\alpha^2}$$

La funzione di autocovarianza non dipende dal tempo, bensì solo dal lag, i.e. $\gamma_t(h) = \gamma(h)$ il che rende i processi stocastici descritti da questo modello, dei processi gaussiani stazionari in senso forte.

Aggiunta del Trend

Il modello autoregressivo appena formulato è vincolato a mantenere una media nulla, potrebbe essere interessante avere un modello che invece è capace di essere più flessibile, in tale caso è possibile agire direttamente sulla formulazione del modello per aggiunere una componente media, chiamata **trend**, definendo il processo autoregressivo come $(X_t - \mu_t) = \alpha(X_{t-1} - \mu_{t-1}) + \epsilon_t$. Questo nuovo processo $\{X_i - \mu_i\}_{i\in I}$ ha le stesse caratteristiche del modello autoregressivo sopracitato, ma questo implica:

$$\begin{aligned}X_t &= \mu_t + \alpha(X_{t-1} - \mu_{t-1}) + \epsilon_t = \dots \\ &= \mu_t + \alpha^{+\infty}(X_{t-\infty} - \mu_{t-\infty}) + \sum_{i=0}^{+\infty} \alpha^i \epsilon_{t-i}\end{aligned}$$

grazie alla solita condizione $\mid \alpha \mid < 1$ si ottiene la seguente uguaglianza: $X_t = \mu_t + \sum_{i=0}^{+\infty} \alpha^i \epsilon_{t-i}$, che conduce a: $\mathbb{E}(X_t) = \mu_t$ e $\gamma(h) = \frac{\alpha^h\psi^2}{1-\alpha^2}$.

Ornstein-Uhlenbeck Model

Il processo stocastico gaussiano, a cui si presta particolare attenzione nella modellazione dei fischi-firma, viene descritto dalla seguente funzione di covarianza esponenziale, introdotta nel 1930 da Ornstein e Uhlenbeck per descrivere la velocità di una particela sottoposta al moto browniano [40]:

$$Cov(X_{t_1}, X_{t_2}) = \tau^2 \exp(-\rho|t_1 - t_2|),\ t_1, t_2 \in I,\ \tau^2, \rho > 0$$

Malgrado i fischi-firma presentino una regolarità maggiore rispetto al moto browniano, la scelta di questo modello rimane comunque interessante, in quanto conserva alcune proprietà desiderabili. In particolare, a tempo discreto con punti equidistanti, il modello può essere riformulato sotto forma di processo autoregressivo di ordine 1.
Dimostrazione:
La proposizione che si vuole dimostrare è: dato la successione dei tempi I equi-distanti, ed il processo $\boldsymbol{X} = \{X_i\}_{i\in I} \sim N(0, \Sigma(\rho, \tau^2))$ con $\Sigma_{i,j}(\rho, \tau^2) := \tau^2 \exp(-\rho|t_i - tj|)$, allora $\{X_i\}_{i\in I}$ è un processo

autoregressivo di ordine 1. Infatti:

$$\begin{pmatrix} X_{t_i} \\ X_{t_j} \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \Sigma_{ii} & \Sigma_{ij} \\ \Sigma_{ji} & \Sigma_{jj} \end{pmatrix} \right)$$

implica, dalle proprietà delle normali multivariate condizionate, che:

$$X_{t_i} \mid X_{t_i} \sim N\left(\frac{\Sigma_{ij}}{\Sigma_{jj}} X_{t_j}, \Sigma_{ii} - \frac{\Sigma_{ij}^2}{\Sigma_{jj}} \right)$$

in questo caso: $\Sigma_{ii} = \Sigma_{jj} = \tau^2$ e $\Sigma_{ij} = \Sigma_{ji} = \tau^2 \exp(-\rho \mid t_i - t_j \mid)$. allora:

$$X_{t_i} \mid X_{t_i} \sim N\left(\frac{\tau^2 \exp(-\rho \mid t_i - t_j \mid)}{\tau^2} X_{t_j}, \tau^2 - \frac{\tau^4 \exp(-2\rho \mid t_i - t_j \mid)}{\tau^2} \right)$$

che riscritto nella notazione di autoregressivo diventa:

$$X_{t_i} = \alpha_{ij} X_{t_j} + \epsilon_{ij}$$

dove: $\epsilon_{ij} \sim N(0, \tau^2(1 - \exp(-2\rho \mid t_i - t_j \mid)))$ e $\alpha_{ij} = \exp(-\rho \mid t_i - t_j \mid)$ imponendo che i punti temporali siano equidistanti, a distanza h, allora si ottiene:

$$X_{t_i} = \exp(-\rho h) X_{t_j} + \epsilon_{ij},\ \epsilon_{ij} \sim N(0, \tau^2(1 - \exp(-2\rho h)))$$

e ridefinendo:

$$\alpha := \exp(-\rho h),\ \psi^2 := \tau^2(1 - \alpha^2)$$

si conclude: $X_{t_i} = \alpha X_{t_j} + \epsilon,\ \epsilon \sim N(0, \psi^2)$ ovvero il processo di partenza, con le ipotesi di equidistanza temporale e covarianza esponenziale, è un processo autoregressivo di ordine 1. □

2.6 Reti Bayesiane

Per l'implementazione dell'algoritmo Gibbs, è necessario determinare la forma esplicita delle distribuzioni full-conditional, a questo fine risulta dunque necessario introdurre il concetto di **rete bayesiana**. Questo strumento permette di evidenziare le dipendenze dirette tra le variabili aleatorie nel modello e dunque di scrivere più chiaramente le dipendenze nelle distribuzioni full-conditional.

Rete Bayesiana

Considerando n variabili casuali $X_1, X_2, \ldots, X_n$ ed un grafo diretto aciclico con n nodi numerati, supponendo che il nodo j $(1 \leq j \leq n)$ del grafo sia associato alla variabile X_j. Allora il grafo è una rete bayesiana, che rappresenta le variabili $X_1, X_2, \ldots, X_n$, se:

$$\mathbb{P}(X_1, X_2, \ldots, X_n) = \prod_{j=1}^{n} \mathbb{P}(X_j \mid \text{genitori}(X_j))$$

dove genitori(X_j) indica l'insieme di tutte le variabili X_i, tali che esiste un arco dal nodo i al nodo j nel grafo [41].

Indipendenza Condizionata

Data una terna di variabili aleatorie X, Y, Z, allora X e Y si definiscono **indipendenti condizionatamente** alla realizzazione di Z, e viene denotato con $X \perp\!\!\!\perp Y|Z$, se vale: $p(x, y|z) = p(x|z)p(y|z)$ con

x, y, z valori delle rispettive variabili aleatorie [42].

Markov Blanket

La **Markov blanket** di una variabile aleatoria in una rete bayesiana è l'insieme minimo di variabili nella rete che, una volta conosciute, rendono quella variabile aleatoria condizionatamente indipendente da tutte le altre nelle rete. A livello pratico si riconosce con questa formulazione: la Markov blanket di una variabile X è l'insieme costituito dai nodi (variabili) genitori di X, dai nodi (variabili) figli di X, i.e. quelli di cui X è genitore, e dai nodi (variabili) cogenitori, i.e. che condividono un figlio con X [43].

Struttura del Grafo

Il grafo diretto viene costruito con una semplice linea guida: se la distribuzione di una variabile aleatoria Y dipende esplicitamente da una variabile aleatoria X, allora si traccia un arco diretto tra la variabile X e la variabile Y, in questo caso si dice che X è genitore di Y, o analogamente, Y è figlio di X. In qualsiasi altro caso non si tracciano archi. Quindi tutti gli archi diretti entranti in Y provengono da variabili che esplicitamente compaiono nella distiribuzione di Y, e tutti gli archi diretti uscenti da Y sono diretti verso variabili aleatorie la cui distribuzione dipende esplicitamente da Y.

Capitolo 3

Riscalamento Temporale

3.1 Definzione del Riscalamento Temporale

Come concluso dall'introduzione del fenomeno naturale, la modulazione del fischio-firma gioca un ruolo chiave nella comunicazione dei delfini tursiopi. E' perciò necessario costruire un modelllo che sia in grado di tenere in considerazione questa modulazione, non trattandola come un discostamento rumoroso dal processo generatore, bensì come una corretta variazione rispetto ad un processo temporale genesi, che è possibile modulare. La modulazione avviene attraverso un riscalamento temporale. A questo fine si definiscono: il processo temporale $W : S \subset \mathbb{R}^+ \longrightarrow W(S) \subset \mathbb{R}^+$ (i.e. il fischio-firma generatore) ed il processo temporale $Y : T \subset \mathbb{R}^+ \longrightarrow Y(T) \subset \mathbb{R}^+$ (i.e la realizzazione). E' stato scelto di denotare il dominio dei tempi del processo generatore con S ed i suoi elementi con s, mentre per il processo Y è stato scelto il dominio T con elementi t, per evidenziare che i due processi vivono in due domini diversi. La seguente definizione di riscalamento temporale specifica la relazione tra i due domini.

Siano $S, T \subset \mathbb{R}^+$ compatti e connessi, si definisce **riscalamento temporale** la mappa $g : T \subset \mathbb{R}^+ \longrightarrow g(T) = S \subset \mathbb{R}^+$ continua, tale che $g'(t) > 0 \ \forall t \in T \setminus T^r$ con $T^r \subset T$ definito come l'insieme dei raccordi, e per cui vale $g(inf(T)) = inf(S)$ e $g(sup(T)) = sup(S)$.

Nelle applicazioni vi sono alcune semplificazioni alle condizioni introdotte nella definizione; innanzitutto i due domini combaciano ad un dominio di lunghezza unitaria, i.e. $S = T = [0, 1] \subset \mathbb{R}^+$, inoltre, siccome il riscalamento temporale $g(.)$ viene applicato secondo la seguente relazione: $g(t) = s \ \forall t \in T$, questo rende le condizioni nella definizione analoghe a: $g(0) = 0$ e $g(1) = 1$. Per flessibilità e chiarezza, è possibile scrivere il processo temporale generatore in due modi:

- $W(s)$ per $s \in S$, per indicare il suo "dominio di appartenenza naturale". Si noti che se rappresentanto in questo dominio, il processo generatore avrà la sua forma "non-modulata";
- $W(g(t)) =: W^*(t)$ per $t \in T$, per indicarlo nel dominio della realizzazione. Si noti che se invece il processo viene rappresentanto in questo dominio, il processo generatore risulta "modulato".

Il Processo di scelta della Funzione di Riscalamento

La definizione di riscalamento temporale nasce in risposta ad osservazioni ed esempi riportati in seguito; le considerazioni nelle prossime righe riassumono il processo che ha portato alla definizione finale del riscalamento temporale dove, inizialmente, non è stata sempre imposta le condizione $g(1) = 1$. Nelle applicazioni si parte sempre dalla situazione in cui i due domini sono in relazione di identità, i.e. $s = t$, quindi, per chiarezza, e mantenuto solamente il dominio temporale T, ovvero la prospettiva in cui si apprezza la modulazione del processo W.

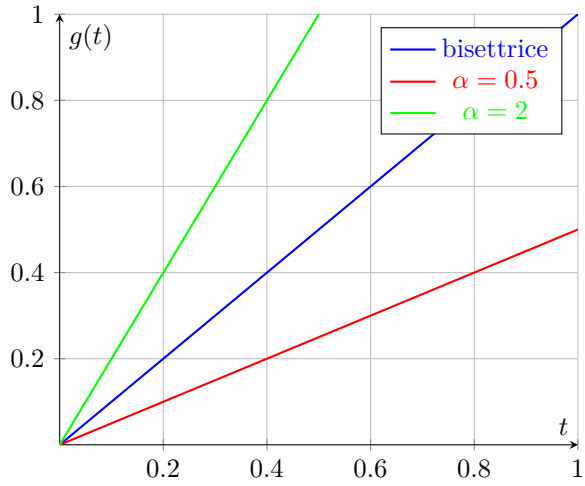


Figura 3.1: Grafico delle funzioni $g(t)$ per $\alpha = 0.5$ e $\alpha = 2$.

1. Funzione di Riscalamento con Derivata Costante minore di 1: $g(t) = \alpha t \ \forall t \in T$ con $g'(t) = \alpha < 1$ $\forall t \in T$

   Se $\alpha < 1$, la trasformazione dell'argomento di $W(.)$ da $s = t$ a $s = g(t) = \alpha t$ si traduce in una "dilatazione" dalla prospettiva del dominio T. Dove per dilatazione si intende quel fenomeno per cui per osservare ciò che accadeva a W al tempo t_1, ovvero $W(t_1)$ si deve aspettare ora un tempo $t'_1 := g^{-1}(t_1) > t_1$. Infatti, nota la seguente catena di uguaglianze: $W(t_1) = W(g(t'_1)) = W(\alpha t'_1)$ e grazie alla relazione base $t_1 = g(t'_1)$, che accompagnerà le analisi di questo tipo nel proseguio di questo capitolo, si ottiene banalmente che: $t_1 = \alpha t'_1$ con $\alpha < 1 \implies t'_1 = \frac{t_1}{\alpha} > t_1$.

   Graficamente, nella figura 3.1, tutte le funzioni di riscalamento temporale che sono "sotto" la bisettrice nel piano $(t, g(t))$ sono delle dilatazioni. Più sono al di sotto e maggiore è la dilatazione in quel punto.

2. Funzione di Riscalamento con Derivata Costante maggiore di 1: $g(t) = \alpha t \ \forall t \in T$ con $g'(t) = \alpha > 1$ $\forall t \in T$

   Se $\alpha > 1$, la trasformazione dell'argomento di $W(.)$ da $s = t$ a $s = g(t) = \alpha t$ si traduce in una "contrazione" dalla prospettiva del dominio T. Dove per contrazione si intende quel fenomeno per cui per osservare ciò che accadeva a W al tempo t_1, ovvero $W(t_1)$ si deve aspettare ora un tempo $t'_1 := g^{-1}(t_1) < t_1$. Infatti, nota la seguente catena di uguaglianze: $W(t_1) = W(g(t'_1)) = W(\alpha t'_1)$ e grazie alla relazione base $t_1 = g(t'_1)$, si ottiene banalmente che: $t_1 = \alpha t'_1$ con $\alpha > 1 \implies t'_1 = \frac{t_1}{\alpha} < t_1$.

   Graficamente, nella figura 3.1, tutte le funzioni di riscalamento temporale che sono "sopra" la bisettrice nel piano $(t, g(t))$ sono delle contrazioni. Più sono al di sopra e maggiore è la contrazione in quel punto.

3. Funzione di Riscalamento generica:

   Data una funzione di riscalamento temporale $g(.)$ applicata su $W(.)$ come $W(g(t))$, vale sempre l'interpretazione data nei due scenari precedenti, dove per osservare ciò che accadeva a $W(.)$ al tempo t_1, ovvero $W(t_1)$ è necessario aspettare un tempo t'_1 tale che: $W(t_1) = W(g(t'_1))$. Nonostante fosse già stata ripetuta l'interpretazione della relazione, è importante averla chiara, perché nei casi sopracitati è facile da capire il concetto, ma quando si usano funzioni meno semplici per riscalare il tempo, esse causano delle modulazioni che possono essere più o meno forti in base a quanto ci si distanzia dalla bisettrice, e l'interpretazione "di attesa" risulta fondamentale per capire l'intensità della variazione causata.

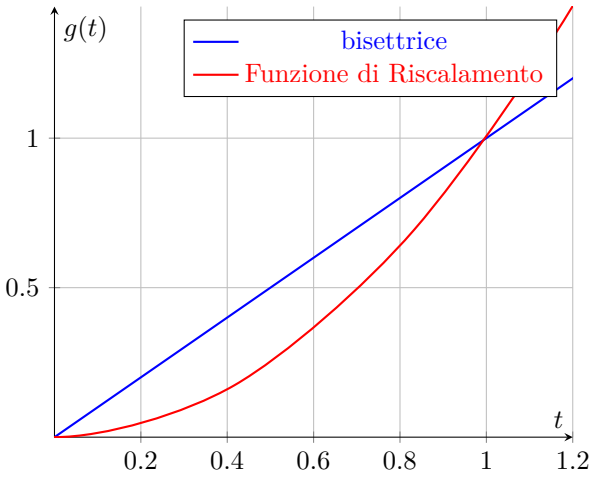


Figura 3.2: Grafico della funzione $g(t) = t^2$. Dove $t^* = 0.5$, mentre il punto temporale che fa passare la modulazione da dilatazione a contrazione è $t = 1$.

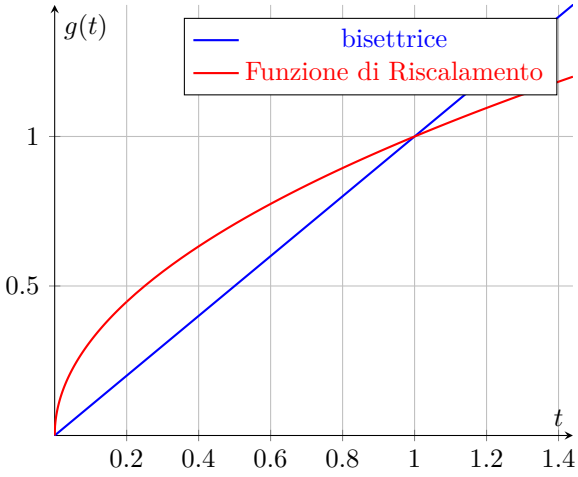


Figura 3.3: Grafico della funzione $g(t) = \sqrt{t}$. Dove $t^* = \frac{1}{4}$, mentre il punto temporale che fa passare la modulazione da contrazione a dilatazione è $t = 1$.

Alcuni esempi di Riscalamento con Funzione

Per meglio familiarizzare col concetto di "riscalamento con funzione" è conveniente analizzare alcuni esempi, che condurranno poi alla proposta finale che verrà scelta.

- Caso: $g'(t) < 1 \ \forall t < t^*$ e $g'(t) > 1 \ \forall t > t^*$

  Si osserva una dilatazione iniziale, che all'aumentare di t diminuisce, fino ad arrivare ad un punto di "identità" dopo il quale si inizia a manifestare una contrazione, che all'aumentare di t aumenta, figura 3.2.

- Caso: $g'(t) > 1 \ \forall t < t^*$ e $g'(t) < 1 \ \forall t > t^*$

  Si osserva una contrazione iniziale, che all'aumentare di t diminuisce, fino ad arrivare ad un punto di "identità" dopo il quale si inizia a manifestare una dilatazione, che all'aumentare di t aumenta, figura 3.3.

3.2 Proposte di Riscalamento Temporale

Per scegliere il riscalamento temporale si devono fare alcune considerazioni.

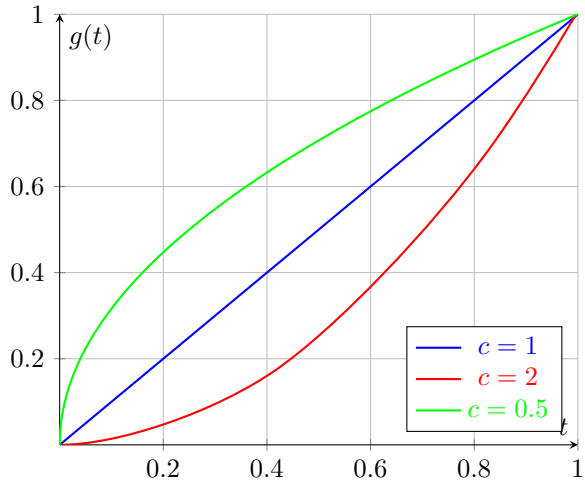


Figura 3.4: Grafico della funzione $g(t) = t^c$ con $c = 1$, $c = 0.5$ e $c = 2$

- Innanzitutto, il tempo su cui è definito il processo $W(.)$ deve sempre essere $[0, 1]$, perché i W sono definiti su quell'asse temporale. Da qui le condizioni $g(0) = 0$ e $g(1) = 1$.
- Inoltre, per mantenere l'interpretazione fisica del tempo concorde per i due processi W e Y, è necessario che la funzione $g(t)$ sia monotona strettamente crescente.
- Per come è sviluppato il modello gerarchico bayesiano, la modulazione deve dipendere da un parametro che amplifichi o riduca la modulazione, presentando quindi anche una configurazione di parametri che restituiscono la modulazione "identità", i.e. nessuna modulazione.
- Infine, il riscalamento temporale deve essere una funzione continua. Questo perché i fischi-firma del dataset a disposizione sono continui.

Da queste considerazioni è possibile proporre alcuni tipi di riscalamenti temporali.

La prima proposta

Il primo riscalamento temporale è quello più semplice in termini di numero di parametri da stimare. Esso permette di tenere in considerazione la modulazione come un fenomeno previsto, non trattandola come una manifestazione rumorsa. Tuttavia, non fornisce molta flessibilità e non rispecchia alcune delle caratteristiche della modulazione che sono state citate nella sezione del fenomeno naturale, dove si evidenziava come la modulazione non fosse la stessa su tutto il contorno del fischio-firma. La prima funzione proposta come riscalamento temporale è:

$$g(t) = t^c \text{ con } t \in [0, 1] \text{ e } c \in (0, +\infty)$$

Questa funzione rispecchia tutte le condizioni sopracitate, infatti essa è vincolata a rispettare $g(0) = 0$ e $g(1) = 1$ indipendentemente dall'intensità della modulazione, la quale viene amplificata o ridotta in base al valore del parametro c, che soddisfa le richieste per l'impostazione bayesiana. Infatti, se $c < 1$ allora si manifesta una contrazione, viceversa, se $c > 1$, si ottiene una dilatazione; il valore del parametro che restituisce la modulazione identità è $c = 1$. Nella stesura del modello bayesiano, a livello pratico, non è usato il parametro c di modulazione esplicitamente, bensì il parametro $b := \log(c)$, per permettere al modello di proporre da una distribuzione normale. Questo non apporta modifiche al meccanismo di modulazione, infatti il riscalamento temporale rimane invariato imponendo: $t^{\exp(b)}$, che è chiaramente uguale a $t^{\exp(\log(c))} = t^c$. E' bene notare come in questo caso la derivata $g'(t)$ sia ben definita su tutto l'insieme unitario e la condizione $g'(t) > 0$ sia sempre soddisfatta, infatti: $g(t) = t^c \implies g'(t) = ct^{c-1}$ ma siccome $c \in (0, +\infty)$ e $t \in [0, 1] \implies g'(t) > 0 \; \forall t \in [0, 1]$. Un esempio in figura 3.4.

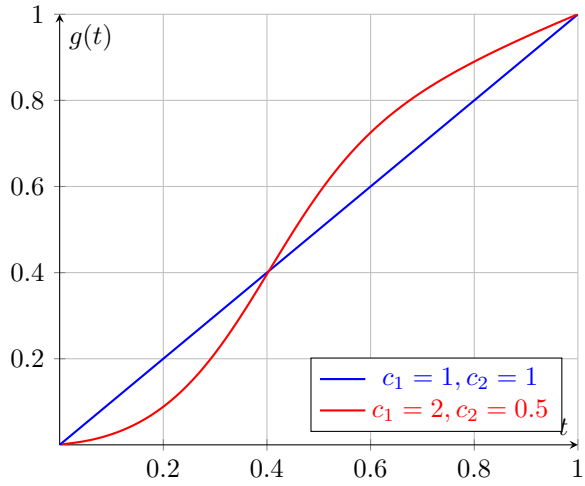


Figura 3.5: Grafico della funzione $g(t) = t^{c_1} + (t^{c_2} - t^{c_1})\frac{1}{1+e^{-\gamma(t-\delta)}}$ con $c_1 = 2$, $c_2 = 0.5$, $\delta = 0.4$, $\gamma = 10$ dove si mostra prima una fase di dilatazione, seguita da una fase di contrazione.

La seconda proposta

Come anticipato nella prima proposta, potrebbe essere interessante gestire con più libertà le fasi di dilatazione, contrazione o identità, non limitando la modulazione ad una sola di esse su tutto il contorno del fischio-firma. Si può puntare ad avere una funzione che modula (ad esempio con una dilatazione) fino ad un certo punto temporale δ, dopo il quale si ottiene un'altra modulazione che può essere diversa da quella usata fino a δ, o uguale. Perciò la seconda funzione di riscalamento temporale proposta è:

$$g(t) = t^{c_1} + (t^{c_2} - t^{c_1})\frac{1}{1+e^{-\gamma(t-\delta)}},\ t \in [0,1],\ c_1, c_2 \in (0,+\infty),\ \delta \in [0,1]\ \gamma \in (0,+\infty)$$

Malgrado questa funzione possa sembrare molto più complessa dell'altra, in realtà si basa sullo stesso principio. L'interpretazione dei parametri c_1 e c_2 è infatti analoga a prima, così come lo è la loro relazione con t. Il parametro c_1 modula la funzione nella prima sezione, mentre la seconda parte viene modulata da c_2. L'intensità della modulazione segue sempre la logica introdotta precedentemente, ossia: se $c_i < 1 \forall i \in \{1,2\}$ allora si ottiene una contrazione, viceversa se $c_i > 1 \forall i \in \{1,2\}$ si ottiene una dilatazione. Le due sezioni sono suddivise dal parametro $\delta \in [0,1]$, mentre la "velocità" di cambiamento è parametrizzata da $\gamma \in (0,+\infty)$. In breve, questa funzione $g(t)$ rappresenta una combinazione di due termini dipendenti dal tempo, con t^{c_1}, inizialmente predominante, e t^{c_2}, che emerge gradualmente con l'aumentare di t oltre un istante di tempo δ. Il parametro γ controlla la rapidità con cui avviene questa transizione. Questo tipo di funzione è più conforme al concetto di modulazione che si è visto nella parte introduttiva, non imponendo che lungo tutto il contorno vi sia lo stesso tipo di riscalamento. Purtroppo, con questa proposta, non vi sono lo stesso tipo di garanzie sulla condizione $g'(t) > 0$ che si avevano in precedenza, infatti:

$$g'(t) = c_1 t^{c_1-1} + \frac{(c_2 t^{c_2-1} - c_1 t^{c_1-1})(1+e^{-\gamma(t-\delta)}) + \gamma(t^{c_2} - t^{c_1})e^{-\gamma(t-\delta)}}{(1+e^{-\gamma(t-\delta)})^2}$$

ammette delle configurazioni di parametri c_1, c_2, δ, γ per cui non risulta essere una funzione strettamente positiva per $t \in [0,1]$. Nell'implementazione dell'algoritmo MCMC le configurazioni proposte che non rispettano la condizione $g'(t) > 0$ vengono rifiutate. Tuttavia, anche in questo caso viene garantita la differenziabilità su tutta la funzione, grazie soprattutto al parametro γ che fa transitare la funzione di riscalamento da t^{c_1} a t^{c_2} in maniera regolare. Un esempio in figura 3.5

La terza proposta

Nonostante i miglioramenti in termini di flessibilità che sono stati offerti dalla seconda proposta, si riconoscono ancora dei limiti a questa struttura, infatti non è detto che la modulazione dei fischi-firma

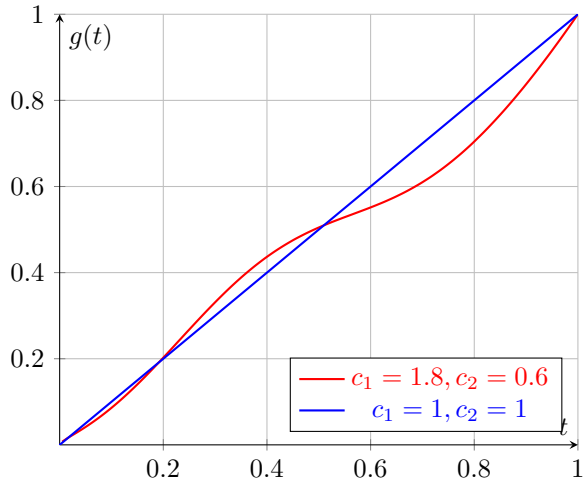


Figura 3.6: Grafico della funzione $g(t) = t^{c_1} + \frac{t^{c_2} - t^{c_1}}{1+e^{-\gamma(t-\delta)}} + \frac{t^{c_1} - t^{c_2}}{1+e^{-\gamma(t-\delta_2)}}$ con $c_1 = 1.8$, $c_2 = 0.6$, $\delta_1 = 0.2$, $\delta_2 = 0.6$, $\gamma = 7.5$ dove si mostra una fase iniziale e finale di dilatazione con una contrazione centrale.

avvenga solamente all'inizio e/o alla fine di essi. A questo scopo l'ultima modulazione proposta introduce la coppia di parametri δ_1, δ_2 tale che $0 \leq \delta_1 \leq \delta_2 \leq 1$, nella seguente funzione di riscalamento temporale:

$$g(t) = t^{c_1} + \frac{t^{c_2} - t^{c_1}}{1+e^{-\gamma(t-\delta_1)}} + \frac{t^{c_1} - t^{c_2}}{1+e^{-\gamma(t-\delta_2)}} \text{ con } t \in [0,1], c_1, c_2 \in (0, +\infty), 0 \leq \delta_1 \leq \delta_2 \leq 1, \gamma \in (0, +\infty)$$

Nuovamente si osserva un riscalamento in funzione del tempo e di altri parametri. I parametri c_1, c_2, γ hanno la medesima interpretazione precedente; δ_1 diventa il punto in cui si passa dalla modulazione offerta da c_1 a quella offerta da c_2, mentre δ_2 è il punto dopo il quale torna a prevalere t^{c_1}. Anche in questo caso la monotonia strettamente crescente non è garantita, la dinamica in cui si gestisce questa criticità nell'algoritmo MCMC è identica a quella specificata nella seconda proposta; la derivabilità è garantita anche in questo scenario sempre dall'utilizzo del parametro γ. Un esempio in figura 3.6

3.3 Senza l'Assunzione di Derivabilità Globale

Nella definizione di riscalamento temporale, si considera una condizione meno restrittiva rispetto alla differenziabilità sull'intero dominio T. Questo per lasciare aperta la possibilità di utilizzare come risorsa, per realizzare riscalamenti temporali, funzioni continue definite a tratti. Come già fatto presente, gran parte della derivabiltà delle funzioni viste fin'ora è merito della formulazione esponenziale e dell'utilizzo del coeffciente γ per il transito tra modulazioni in maniera regolare. Tuttavia, questo coefficiente porta con sé delle insidie che possono fuorivare significativamente le stime dell'algoritmo MCMC. Nello specifico si osservi la figura 3.6 nella trattazione della terza proposta. Essa mostra come i parametri δ_1 e δ_2, malgrado la loro chiara interpretabilità come punti di transizione tra una modulazione e l'altra, non rispecchino esattamente tale fenomeno. Infatti, il primo cambio di modulazione è in $t = \delta_1 = 0.2$, ma il secondo avviene significativamente sotto $\delta_2 = 0.6$. Questo meccanismo fuorviante avviene in funzione del parametro γ e delle modulazioni $\boldsymbol{c}_i$, che insieme determinano i posizionamenti dei paremtri δ, più o meno distanti dal loro posizionamento supposto. La formulazione regolare, dunque, crea una dinamica complessa che può sfociare in problemi di "non-identificabilità", in cui simili configurazioni forniscono risultati molto diversi e viceversa. Un'ulteriore criticità di questa proposta riguarda la monotonia; questa condizione viene facilmente evasa, anche a seguito di minime variazioni dei parametri di modulazione, il ché porta ad uno scenario di rifiuto frequente che impedisce all'algoritmo di ispezionare liberamente lo spazio degli stati, imponendo quindi una restrizione troppo forte alle proposte. Un esempio di configurazione, che conduce alla caduta della condizione di monotonia, è in figura 3.7, dove l'unica differenza rispetto alla figura 3.6 risiede nel parametro γ che passa da 7.5 a 12.5. Alla luce di queste criticità, è possibile adottare una

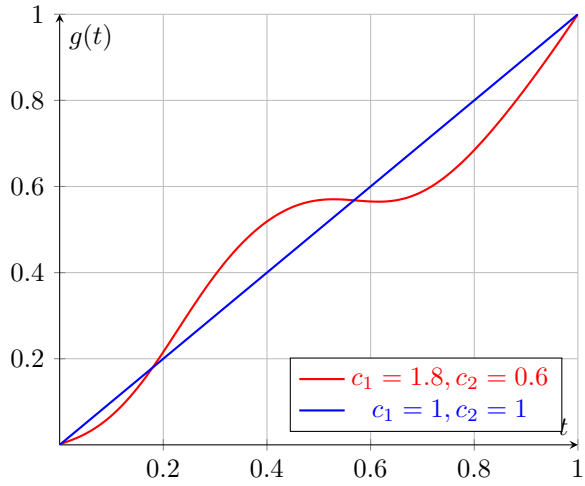


Figura 3.7: Grafico della funzione $g(t) = t^{c_1} + \frac{t^{c_2} - t^{c_1}}{1+e^{-\gamma(t-\delta)}} + \frac{t^{c_1} - t^{c_2}}{1+e^{-\gamma(t-\delta_2)}}$ con $c_1 = 1.8$, $c_2 = 0.6$, $\delta_1 = 0.2$, $\delta_2 = 0.6$, $\gamma = 12.5$ dove si mostra una fase in cui la monotonia non è rispettata.

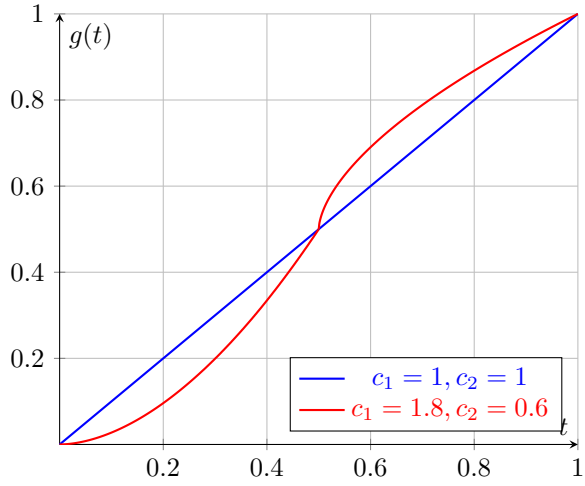


Figura 3.8: Grafico della funzione $g(t)$ definita a tratti con $c_1 = 1.8$, $c_2 = 0.6$ e $\delta = 0.5$.

formulazione con funzione di riscalamento definita a tratti, rimuovendo il parametro γ. Questa alternativa, oltre a garantire sempre la monotonia, permette anche di posizionare con precisione i coefficienti δ, in modo da riflettere il fenomeno di transizione della modulazione che rappresentano. Questo beneficio si paga al prezzo della non-regolarità della funzione $g(.)$ nei punti temporali di raccordo $t \in T^r$, ovvero gli istanti temporali di congiunzione tra una funzione in un tratto e quella definita sul tratto prossimo.

La seconda proposta senza derivabiltà globale

Tenendo presente che i coefficienti utilizzati mantengono la stessa interpretazione della formulazione regolare, la funzione $g(t)$, definita a tratti, assume la seguente espressione:

$$g(t) = \begin{cases} \frac{t^{c_1}}{\delta^{c_1-1}}, & \text{se } t \in (0, \delta), \\ \delta + \frac{(t-\delta)^{c_2}}{(1-\delta)^{c_2-1}}, & \text{se } t \in [\delta, 1]. \end{cases}$$

In questo caso la monotonia è assicurata, poiché questa formulazione rappresenta una generalizzazione della prima proposta. Dalla Figura 3.8 emerge chiaramente che la derivabilità nel punto $t = \delta = 0.5$ non è preservata; tuttavia, i benefici derivanti da questa scelta sembrano giustificare la perdita di tale proprietà.

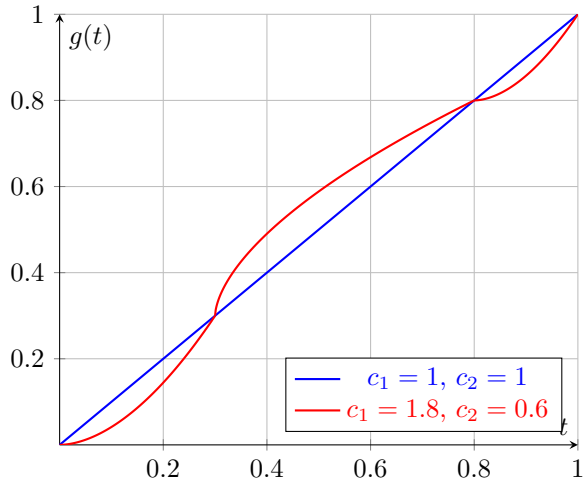


Figura 3.9: Grafico della funzione $g(t)$ definita a tratti con $c_1 = 1.8$, $c_2 = 0.6$, $\delta_1 = 0.3$ e $\delta_2 = 0.8$.

La terza proposta senza derivabilità globale

Seguendo le solite premesse di interpretabilità dei coefficienti, la terza proposta, senza la proprietà di derivabilità globale, diventa:

$$g(t) = \begin{cases} \frac{t^{c_1}}{\delta_1^{c_1-1}}, & \text{se } t \in [0, \delta_1], \\ \delta_1 + \frac{(t-\delta_1)^{c_2}}{(\delta_2-\delta_1)^{c_2-1}}, & \text{se } t \in (\delta_1, \delta_2], \\ \delta_2 + \frac{(t-\delta_2)^{c_1}}{(1-\delta_2)^{c_1-1}}, & \text{se } t \in (\delta_2, 1]. \end{cases}$$

che rappresentata si traduce in 3.9

La Formulazione Unificata

Al fine di semplificare l'esposizione e favorire la sintesi della notazione nelle sezioni successive, è opportuno osservare che le due famiglie di modulazioni possono essere espresse in forma chiusa, ammettendo, in via del tutto informale, l'uso della notazione $(0, +\infty]$ per descrivere l'intervallo di definizione di γ. Infatti, sia per la seconda proposta, che per la terza, si può sfruttare il seguente limite:

$$\lim_{\gamma \to +\infty} \frac{1}{1 + e^{-\gamma(t-\delta)}} \to 1 \ , \ \forall t > \delta$$

e

$$\lim_{\gamma \to +\infty} \frac{1}{1 + e^{-\gamma(t-\delta)}} \to 0 \ , \ \forall t < \delta$$

Questo abuso di notazione permette dunque di accomunare le modulazioni con e senza derivabilità globale come segue:

- Seconda famiglia di modulazione:

$$g(t) = t^{c_1} + (t^{c_2} - t^{c_1}) \frac{1}{1+e^{-\gamma(t-\delta)}} \text{ con } t \in [0,1], c_1, c_2 \in (0, +\infty), \delta \in [0,1], \gamma \in (0, +\infty]$$

- Terza famiglia di modulazione:

$$g(t) = t^{c_1} + \frac{t^{c_2} - t^{c_1}}{1+e^{-\gamma(t-\delta_1)}} + \frac{t^{c_1} - t^{c_2}}{1+e^{-\gamma(t-\delta_2)}} \text{ con } t \in [0,1], c_1, c_2 \in (0, +\infty), 0 \leq \delta_1 \leq \delta_2 \leq 1, \gamma \in (0, +\infty]$$

Capitolo 4

Modello

4.1 Spiegazione del Modello

L'assunzione cardine del modello vede il fischio-firma dei delfini tursiopi come la realizzazione di un processo temporale $Y_i(t) \in \mathbb{R}$, questa entità rappresenta il valore del contorno di frequenza del fischio-firma i-esimo nel punto temporale t [23]. Questo processo $Y_i(t)$, chiamato *realizzazione*, è una manifestazione rumorosa e modulata di un processo *generatore*, indicato $W_{z_i}^T(s)$, caratteristico dell'individuo z_i, in cui s evidenzia la diversa natura temporale dei due processi, in relazione mediante l'uguaglianza $s = g(t)$. Per sciogliere un piccolo nodo notazionale: z_i è utilizzato per rifersi all'individuo che ha emesso il suono i-esimo; quando non viene indicizzato il suono l'individuo generico di riferimento è indicato con k. Il processo $Y_i(t)$ valutato in questi d-punti temporali è denotato:

$$\mathbf{y}_i = (y_{i,1}, y_{i,2}, \ldots, y_{i,d})^\top$$

mentre il processo latente generatore delle istanze dell'esemplare z_i è:

$$\mathbf{w}_{z_i} = (w_{z_i,1}, w_{z_i,2}, \ldots, w_{z_i,|T^{z_i}|})^\top$$

Le Realizzazioni

I fischi-firma, seppur dello stesso esemplare, non vengono sempre riprodotti alla stessa frequenza iniziale, perciò è necessario introdurre una variabile $u_i \in \mathbb{R}$ caratteristica di ogni osservazione e che descriva questo fenomeno traslazionale. Il discostamento rumoroso, o non-spiegato, tra la realizzazione ed il processo generatore rimodulato, giustifica la scelta della distribuzione normale univariata come fonte di aleatorietà, con parametro di varianza τ_i^2, anch'esso dipendente unicamente dalla singola osservazione che lo manifesta. La formulazione completa di $Y_i(t)$ diventa:

$$Y_i(t) \mid W_{z_i}^T, u_i, \tau_i^2, z_i \sim N(u_i + W_{z_i}^T(s), \tau_i^2), \quad s = g(t), \quad t \in [0,1] \tag{4.1}$$

La quale dichiara che il valore del fischio-firma i-esimo rilevato al tempo t si comporta mediamente come una modulazione del processo generatore tipico dell'esemplare, al più di traslazioni nello spettro di frequenza.

I Generatori

Come suggerito in [23], è ragionevole richiedere che il suono latente sia continuo, siccome la forma dei fischi-firma sembra esserlo. Questo è possibile introdurlo definendo $W_k^T(.)$ come un processo gaussiano, dove k indica l'individuo generico. È noto che, senza ulteriori assunzioni, la gestione della matrice

di covarianza di un processo gaussiano (GP) è spesso computazionalmente onerosa, infatti, in modelli complessi, come quelli utilizzati nel contesto bayesiano, è richiesta una manipolazione avanzata di questa matrice, tra cui il calcolo di forme quadratiche e la sua inversione ripetuta [44]. Questo conduce a definire il GP secondo il modello Ornstein-Uhlenbeck con trend. Malgrado questo modello fosse pensato per descrivere dei fenomeni molto irregolari, gode anche di molta flessibilità ed è possibile ottenere delle configurazioni conformi allo scenario attuale [23]. La formulazione di W_k^T quindi diventa:

$$W_k^T \mid \boldsymbol{\beta}^k, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k \sim N_{|T^k|}(X^k\boldsymbol{\beta}^k, \Sigma(\alpha_k, \psi_k^2)) \tag{4.2}$$

dove le entità: $(\boldsymbol{\beta}^k, \alpha^k, \psi^{2,k})$ introdotte sono tutte esplicitamente dipendenti dal gruppo k. Il vettore dei coefficienti $\boldsymbol{\beta}^k$ agisce sulla media, mentre i coefficienti α^k, ψ^2 hanno interpretazione analoga a quelle usate nella trattazione del modello Ornstein-Uhlenbeck. Il vettore $\boldsymbol{\theta}^k$ nell'elenco delle variabili condizionanti, rappresenta tutta quella serie di parametri necessari per generare il riscalamento temporale, fulcro dell'analisi.

Il Riscalamento Temporale

I riscalamenti temporali proposti sono diversi e di conseguenza introducono grandezza diverse nel vettore $\boldsymbol{\theta}^k$. I parametri in questione subentrano implicitiamente nella distribuzione 4.2, dove il loro contributo modula l'insieme di d-tempi delle osservazioni e sono unici per ogni istanza. Questo coinvolgimento contribuisce quindi a determinare l'insieme dei tempi T^k per ogni individuo, con un meccanismo di aggregazione dei tempi modulati, che di conseguenza determina la matrice X^k, e ciò ne giustifica il suffisso di gruppo. All'atto pratico, data l'istanza i-esima, e fissato il riscalamento temporale di interesse c_i, caratteristico per questa istanza, si applica la modulazione temporale t^{c_i} all'insieme di d-tempi equidistanti in $[0, 1]$, che porta ad ottenere altri d-tempi sempre in [0,1], che probabilmente non sono più equidistanti. Indicando genericamente il numero di fischi-firma rilevati dall'individuo k-esimo come n_k, e assumendo che ogni istanza abbia una modulazione propria diversa da tutte le altre, assunzione ragionevole vista la natura continua di tutti i coefficienti di modulazione che verranno utilizzati, allora vale la seguente relazione: $\mid T^k \mid = n_k d_k - 2(n_k - 1)$. Perché, nel momento in cui si sta agendo nell'intervallo $[0, 1]$, con 0 ed 1 rispettivamente il primo ed ultimo elemento temporale di ogni rilevazione, essi, per come sono state impostante le modulazioni, rimangono invariati, mentre tutti gli altri elementi in mezzo si modificano. Quindi si avranno $n_k d_k$ nuovi tempi, ma $2n_k$ di questi coincidono perché sono n_k "zeri" ed n_k "uni", allora si prende solamente una coppia di questi e gli altri si tolgono, perciò: $n_k d_k - 2n_k + 2$.

4.2 Il Modello in Forma Esplicita

$$
\begin{aligned}
&Y_i(t) \mid W_{z_i}^T, u_i, \tau_i^2, z_i \sim N(u_i + W_{z_i}^T(s), \tau_i^2), \quad s = g(t), \quad t \in [0,1] \\
&z_i \sim Disc(\boldsymbol{\pi}) \\
&W_k^T \mid \boldsymbol{\beta}^k, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k \sim N_{|T^k|}(X^k\boldsymbol{\beta}^k, \Sigma(\alpha_k, \psi_k^2)) \\
&\alpha^k \sim U(0,1) \ \in (0,1) \\
&\psi^{2,k} \sim IG(0.1, 0.1) \ \in \mathbb{R}^+ \\
&\beta_0^k \sim N(15, 30) \ \in \mathbb{R} \\
&\beta_1^k \sim N0, 30) \ \in \mathbb{R} \\
&\sigma_u^2 \sim IG(0.1, 0.1) \ \in \mathbb{R}^+ \\
&u_i \mid \sigma_u^2 \sim N(0, \sigma_u^2) \ \in \mathbb{R} \\
&\tau_i^2 \sim IG(0.1, 0.1) \ \in \mathbb{R}^2 \\
&\boldsymbol{\theta}^k \sim D(\boldsymbol{\theta}), \\
&\boldsymbol{\theta}^k \text{ indica i coefficienti dediti alla modulazione.}
\end{aligned}
\tag{4.3}
$$

Distribuzione Categoriale

La distribuzione indicata come Disc($\boldsymbol{\pi}$) esula dalle finalità di questa analisi, tuttavia rispecchia l'applicazione principale a cui aspira questo modello: il riconoscimento degli individui ed il conseguente raggruppamento dei fischi-firma. Perciò vale la pena spendere alcune parole a riguardo. Si tratta di una distribuzione spesso utilizzata nei modelli gerarchici bayesiani per assegnare osservazioni a cluster o gruppi latenti. Questa distribuzione discreta è una generalizzazione della distribuzione categoriale, definita da un vettore di probabilità

$$\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_K),$$

dove π_k rappresenta la probabilità che un'osservazione venga assegnata al cluster k, con la condizione che

$$\sum_{k=1}^{K} \pi_k = 1.$$

Discussione sulle Priors

Le distribuzioni a-priori scelte sono tutte di tipo *non informativo*, l'unica considerazione fatta sul fenomeno fisico che ha guidato la stesura delle prior, è nella scelta della distribuzione di β_0^k e β_1^k, siccome il fischio-firma spazia in uno spettro di frequenza che va dai 0 kHz ai 30 kHz. La scelta di α^k e $\psi^{2,k}$ segue le giustificazioni del modello Ornsetein-Uhlenbeck. In generale i parametri di varianza sono stati modellati da priors $IG(.,.)$ non informative per ottenere delle distribuzioni in forma chiusa e applicare direttamente l'algoritmo Gibbs. In ultima analisi vi è la coppia:

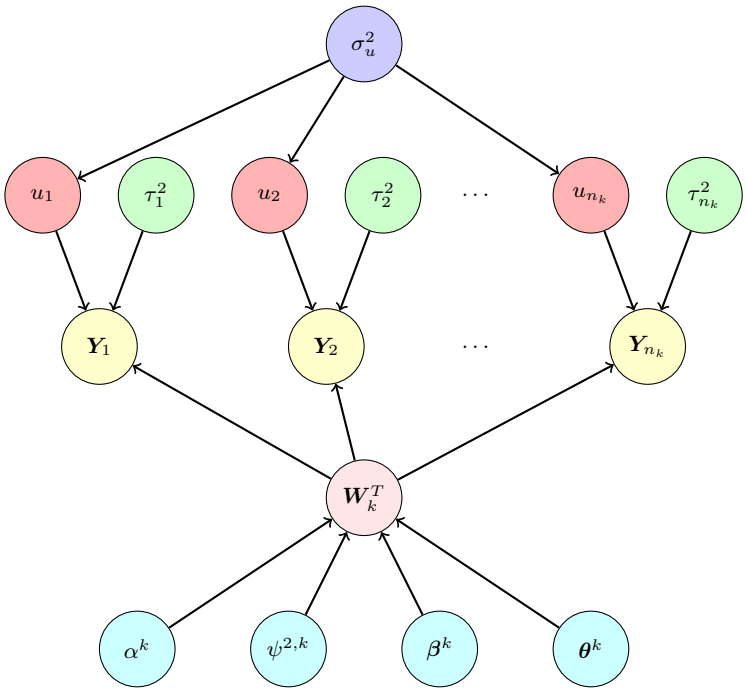
$$\sigma_u^2 \sim IG(0.1, 0.1) \ \in \mathbb{R}^+$$

$$u_i \mid \sigma_u^2 \sim N(0, \sigma_u^2) \ \in \mathbb{R}$$

questa è chiaramente la parte del modello dedita alla traslazione verticale, in cui è stato introdotto il coefficiente σ_u^2 per evidenziare che la varianza del fenomeno traslazionale non è nota ed è giusto che la scelga il modello stesso, carpendola dai dati.

4.2.1 Il Network Bayesiano

Per la corretta implementazione dell'algoritmo è necessario costruire la rete bayesiana, che fornisce le dipendenze dirette tra variabili aleatorie e agevola il calcolo delle distribuzioni full-conditional. Segue il grafo diretto aciclico che descrive le dipendenze tra le variabili nella descrizione di un gruppo, i.e. individuo, k-esimo in cui si denota con n_k il numero di fischi-firma emessi dall'esemplare k-esimo:



4.3 Le Distribuzioni Full-Conditional

Nei calcoli, e nell'implementazione dell'algoritmo, risulta particolarmente conveniente introdurre le matrici D_i^k, per poter scrivere la distribuzione del vettore multivariato $\boldsymbol{Y}_i$ come:

$$\boldsymbol{Y}_i \mid \boldsymbol{W}_{z_i}^T, u_i, \tau_i^2 \sim N_d\left(u_i \mathbf{1}_d + D_i^{z_i} \boldsymbol{W}_{z_i}^T, \tau_i^2 \mathbf{I_d}\right)$$

dove viene indicato con $\mathbf{1}_d$ il vettore di lunghezza d avente tutte le entrate pari ad 1; mentre con $\mathbf{I}_d$ la matrice diagonale quadrata con d-righe e d-colonne. La matrice $D_i^{z_i}$ in questo caso funziona da "selezionatore" delle d-entrate del vettore $\boldsymbol{W}_{z_i}^T$ tali per cui vale:

$$Y_i(t_j) \mid \boldsymbol{W}_{z_i}^T, u_i, \tau_i^2 \sim N_d\left(u_i + W_{z_i}^T\left(s_j\right), \tau_i^2\right) \ \forall t_j$$

con $s_j = g(t_j)$. Prima di ottenere le full-conditional delle variabili, sono elencate le loro dipendenze con le altre variabili del modello che sono presenti nella loro Markov blanket. Il primo insieme sono i nodi genitori, segue l'insieme dei nodi figlio ed, infine, vi è l'insieme dei nodi co-genitori.

La Full-Conditional di u_i

L'analisi delle dipendenze dalla rete bayesiana tramite Markov blanket è:

$$\text{MB}(u_i) = \left\{\sigma_u^2\right\}, \left\{\boldsymbol{Y}_i\right\}, \left\{\boldsymbol{W}_{z_i}^T, \tau_i^2\right\}$$

Perciò si cerca una distribuzione condizionata con questa sequenza di dipendenze:

$$u_i \mid \boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, \tau_i^2, \sigma_u^2$$

Segue un'analisi preliminare per determinare la full-conditional, considerando solamente il Kernel $K(.)$ ed esplicitando i termini che contribuiscono esclusivamente alla costante di normalizzazione $C(.)$ in colore blu:

$$\begin{aligned} f(u_i \mid \boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, \tau_i^2, \sigma_u^2) &= \frac{K(u_i \mid \boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, \tau_i^2, \sigma_u^2)}{C(\boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, \tau_i^2, \sigma_u^2)} \\ &= \frac{f(\boldsymbol{Y}_i \mid u_i, \boldsymbol{W}_{z_i}^T, \tau_i^2) f(\boldsymbol{W}_{z_i}^T) f(u_i \mid \sigma_u^2) f(\tau_i^2) f(\sigma_u^2)}{f(\boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, \tau_i^2, \sigma_u^2)} \end{aligned}$$

si osserva che per determinare la distribuzione full-conditional di u_i è sufficiente considerare il kernel di queste due distribuzioni:

- La distribuzione: $f(\boldsymbol{Y}_i \mid u_i, \boldsymbol{W}_{z_i}^T, \tau_i^2)$
- La prior di u_i: $f(u_i \mid \sigma_u^2)$

esplicitamente:

$$\begin{aligned} K(\boldsymbol{y}_i \mid u_i, \boldsymbol{w}, \tau_i^2) &\propto \exp\left(-\frac{1}{2}\left[\left(\boldsymbol{y}_i - \left(u_i + D_i^k \boldsymbol{w}\right)\right)^\top \left(\tau_i^2 \mathbf{I}_d\right)^{-1} \left(\boldsymbol{y}_i - \left(u_i + D_i^k \boldsymbol{w}\right)\right)\right]\right) \\ &\propto \exp\left(-\frac{1}{2}\left[-2\left(u_i \mathbf{1}_d\right)^\top \left(\left(\tau_i^2 \mathbf{I}_d\right)^{-1} \left(\boldsymbol{y}_i - D_i^k \boldsymbol{w}\right)\right) + \left(u_i \mathbf{1}_d\right)^\top \left(\tau_i^2 \mathbf{I}_d\right)^{-1} \left(u_i \mathbf{1}_d\right)\right]\right) \\ K(u_i \mid \sigma_u^2) &\propto \exp\left(-\frac{u_i^2}{2\sigma_u^2}\right) \end{aligned}$$

svolgendo il prodotto tra questi due kernel si ottiene:

$$\begin{aligned} K(u_i \mid \boldsymbol{y}_i, \boldsymbol{w}, \tau_i^2, \sigma_u^2) &\propto \exp\left(-\frac{1}{2}\left(\frac{1}{\tau_i^2}\left[-2\left(u_i \mathbf{1}_d\right)^\top \left(\boldsymbol{y}_i - D_i^k \boldsymbol{w}\right) + d\left(u_i^2\right)\right]\right) + \frac{u_i^2}{\sigma_u^2}\right) \\ &\propto \exp\left(-\frac{\tau_i^2 \sigma_u^2}{2\left(d\sigma_u^2 + \tau_i^2\right)}\left[u_i^2 - 2u_i\left[\sum_{j=1}^{d} y_{ij} - \left(D_i^k \boldsymbol{w}\right)_j\right]\frac{\sigma_u^2}{d\sigma_u^2 - \tau_i^2}\right]\right) \end{aligned}$$

che si riconosce essere il kernel di una distribuzione normale univariata, esplicitamente:

$$\begin{aligned} \text{M}_u &= \frac{\left[\sum_{j=1}^{d} y_{ij} - \left(D_i^k \boldsymbol{w}\right)_j\right]}{\tau_i^2} \frac{\tau_i^2 \sigma_u^2}{d\sigma_u^2 + \tau_i^2} \\ \text{V}_u &= \frac{\tau_i^2 \sigma_u^2}{d\sigma_u^2 + \tau_i^2} \end{aligned}$$

e perciò $u_i \mid \boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, \tau_i^2, \sigma_u^2 \sim N(\text{M}_u, \text{V}_u)$

La Full-Conditional di τ_i^2

L'analisi delle dipendenze dalla rete bayesiana tramite Markov blanket è:

$$\text{MB}(\tau_i^2) = \{\emptyset\}, \{\boldsymbol{Y}_i\}, \{\boldsymbol{W}_{z_i}^T, u_i\}$$

Perciò si cerca una distribuzione condizionata con questa sequenza di dipendenze:

$$\tau_i^2 \mid \boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, u_i$$

Segue un'analisi preliminare per determinare la full-conditional, considerando solamente il Kernel $K(.)$ ed esplicitando i termini che contribuiscono esclusivamente alla costante di normalizzazione $C(.)$ in colore blu:

$$\begin{aligned} f(\tau_i^2 \mid \boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, u_i) &= \frac{K(\tau_i^2 \mid \boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, u_i)}{C(\boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, u_i)} \\ &= \frac{f(\boldsymbol{Y}_i \mid u_i, \boldsymbol{W}_{z_i}^T, \tau_i^2) f(\boldsymbol{W}_{z_i}^T) f(u_i) f(\tau_i^2)}{f(\boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, u_i)} \end{aligned}$$

si osserva che per determinare la distribuzione full-conditional di τ_i^2 è sufficiente considerare il kernel di queste due distribuzioni:

- La distribuzione: $f(\boldsymbol{Y}_i \mid u_i, \boldsymbol{W}_{z_i}^T, \tau_i^2)$
- La prior di τ_i^2: $f(\tau_i^2)$

esplicitamente:

$$K(\boldsymbol{y}_i \mid u_i, \boldsymbol{w}, \tau_i^2) \propto \left(\tau_i^2\right)^{-\frac{d}{2}} \exp\left(-\frac{1}{2\tau_i^2}\left[\left(\boldsymbol{y}_i - \left(u_i + D_i^k \boldsymbol{w}\right)\right)^\top \left(\boldsymbol{y}_i - \left(u_i + D_i^k \boldsymbol{w}\right)\right)\right]\right)$$

$$K(\tau_i^2) \propto \left(\tau_i^2\right)^{-(a+1)} \exp\left(-\frac{b}{\tau_i^2}\right)$$

svolgendo il prodotto tra questi due kernel si ottiene:

$$\begin{aligned} &K(\tau_i^2 \mid \boldsymbol{y}_i, \boldsymbol{w}, u_i) \propto \\ &\propto \left(\tau_i^2\right)^{-(\frac{d}{2}+a+1)} \exp\left(-\frac{1}{2\tau_i^2}\left[\left(\boldsymbol{y}_i - \left(u_i + D_i^k \boldsymbol{w}\right)\right)^\top \left(\boldsymbol{y}_i - \left(u_i + D_i^k \boldsymbol{w}\right)\right)\right]\right) \exp\left(-\frac{b}{\tau_i^2}\right) \\ &\propto \left(\tau_i^2\right)^{-(\frac{d}{2}+a+1)} \exp\left(-\frac{\left(\boldsymbol{y}_i - \left(u_i + D_i^k \boldsymbol{w}\right)\right)^\top \left(\boldsymbol{y}_i - \left(u_i + D_i^k \boldsymbol{w}\right)\right) \frac{1}{2} + b}{\tau_i^2}\right) \end{aligned}$$

che si riconosce essere il kernel di una distribuzione inverse-gamma, esplicitamente:

$$\begin{aligned} \text{a}_{\tau_i^2} &= \frac{d}{2} + a \\ \text{b}_{\tau_i^2} &= \frac{\left(\boldsymbol{y}_i - \left(u_i + D_i^k \boldsymbol{w}\right)\right)^\top \left(\boldsymbol{y}_i - \left(u_i + D_i^k \boldsymbol{w}\right)\right)}{2} + b \end{aligned}$$

e perciò $\tau_i^2 \mid \boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, u_i \sim IG(\text{a}_{\tau_i^2}, \text{b}_{\tau_i^2})$

La Full-Conditional di $\boldsymbol{\beta}^k$

L'analisi delle dipendenze dalla rete bayesiana tramite Markov blanket è:

$$\text{MB}(\boldsymbol{\beta}^k) = \{\emptyset\}, \{\boldsymbol{W}_{z_i}^T\}, \{\alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k\}$$

Perciò si cerca una distribuzione condizionata con questa sequenza di dipendenze:

$$\boldsymbol{\beta}^k \mid \boldsymbol{W}_{z_i}^T, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k$$

Segue un'analisi preliminare per determinare la full-conditional, considerando solamente il Kernel $K(.)$ ed esplicitando i termini che contribuiscono esclusivamente alla costante di normalizzazione $C(.)$ in colore blu:

$$\begin{aligned} f(\boldsymbol{\beta}^k \mid \boldsymbol{W}_{z_i}^T, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k) &= \frac{K(\boldsymbol{\beta}^k \mid \boldsymbol{W}_{z_i}^T, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k)}{C(\boldsymbol{W}_{z_i}^T, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k)} \\ &= \frac{f(\boldsymbol{W}_{z_i}^T \mid \boldsymbol{\beta}^k, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k) f(\boldsymbol{\beta}^k) f(\boldsymbol{\theta}^k) f(\alpha^k) f(\psi^{2,k})}{f(\boldsymbol{W}_{z_i}^T, \alpha^k, \psi^{2,k} \boldsymbol{\theta}^k)} \end{aligned}$$

si osserva che per determinare la distribuzione full-conditional di $\boldsymbol{\beta}^k$ è sufficiente considerare il kernel di queste due distribuzioni:

- La full-conditional: $f(\boldsymbol{W}_{z_i}^T \mid \boldsymbol{\beta}^k, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k)$
- La prior di $\boldsymbol{\beta}^k$: $f(\boldsymbol{\beta}^k)$

esplicitamente:

$$\begin{aligned} K(\boldsymbol{w} \mid \boldsymbol{\beta}^k, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k) &\propto \\ &\propto \exp\left(-\frac{1}{2}\left(\boldsymbol{w} - X^k\boldsymbol{\beta}^k\right)^\top \left(\Sigma^k\right)^{-1}\left(\boldsymbol{w} - X^k\boldsymbol{\beta}^k\right)\right) \\ &\propto \exp\left(-\frac{1}{2}\left[-2\left(\boldsymbol{\beta}^k\right)^\top \left(X^k\right)^\top \left(\Sigma^k\right)^{-1}\boldsymbol{w} + \left(\boldsymbol{\beta}^k\right)^\top \left(X^k\right)^\top \left(\Sigma^k\right)^{-1} X^k\boldsymbol{\beta}^k\right]\right) \end{aligned}$$

$$\begin{aligned} K(\boldsymbol{\beta}^k) &\propto \exp\left(-\frac{1}{2}\left(\boldsymbol{\beta}^k - \mu_{\boldsymbol{\beta}}\right)^\top \left(\Sigma_{\boldsymbol{\beta}}\right)^{-1}\left(\boldsymbol{\beta}^k - \mu_{\boldsymbol{\beta}}\right)\right) \\ &\propto \exp\left(-\frac{1}{2}\left[-2\left(\boldsymbol{\beta}^k\right)^\top \left(\Sigma_{\boldsymbol{\beta}}\right)^{-1}\boldsymbol{\mu}_{\boldsymbol{\beta}} + \left(\boldsymbol{\beta}^k\right)^\top \left(\Sigma_{\boldsymbol{\beta}}\right)^{-1}\boldsymbol{\beta}^k\right]\right) \end{aligned}$$

svolgendo il prodotto tra questi due kernel si ottiene:

$$\begin{aligned} &K(\boldsymbol{\beta}^k \mid \boldsymbol{W}_{z_i}^T, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k) \propto \\ &\propto \exp\left(-\frac{1}{2}\left[-2(\boldsymbol{\beta}^k)^\top (X^k)^\top (\Sigma^k)^{-1}\boldsymbol{w} + (\boldsymbol{\beta}^k)^\top (X^k)^\top (\Sigma^k)^{-1} X^k\boldsymbol{\beta}^k - 2(\boldsymbol{\beta}^k)^\top (\Sigma_{\boldsymbol{\beta}})^{-1}\boldsymbol{\mu}_{\boldsymbol{\beta}} + (\boldsymbol{\beta}^k)^\top (\Sigma_{\boldsymbol{\beta}})^{-1}\boldsymbol{\beta}^k\right]\right) \\ &\propto \exp\left(-\frac{1}{2}\left[-2\left(\boldsymbol{\beta}^k\right)^\top \left[\left(X^k\right)^\top \left(\Sigma^k\right)^{-1}\boldsymbol{w} + \Sigma_{\boldsymbol{\beta}}^{-1}\boldsymbol{\mu}_{\boldsymbol{\beta}}\right] + \left(\boldsymbol{\beta}^k\right)^\top \left(\left(X^k\right)^\top \left(\Sigma^k\right)^{-1} X^k + \Sigma_{\boldsymbol{\beta}}^{-1}\right)\boldsymbol{\beta}^k\right]\right) \end{aligned}$$

che si riconosce essere il kernel di una distribuzione normale bivariata, esplicitamente:

$$\begin{aligned} \mathrm{V}_{\boldsymbol{\beta}}^{-1} &= \left(X^k\right)^\top \left(\Sigma^k\right)^{-1} X^k + \Sigma_{\boldsymbol{\beta}}^{-1} \\ \mathrm{M}_{\boldsymbol{\beta}} &= \mathrm{V}_{\boldsymbol{\beta}}\left[\left(X^k\right)^\top \left(\Sigma^k\right)^{-1}\boldsymbol{w} + \Sigma_{\boldsymbol{\beta}}^{-1}\boldsymbol{\mu}_{\boldsymbol{\beta}}\right] \end{aligned}$$

e perciò $\boldsymbol{\beta}^k \mid \boldsymbol{W}_{z_i}^T, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k \sim N_2(\mathrm{M}_{\boldsymbol{\beta}}, \mathrm{V}_{\boldsymbol{\beta}})$

La Full-Conditional di σ_u^2

L'analisi delle dipendenze dalla rete bayesiana tramite Markov blanket è:

$$\mathrm{MB}(\sigma_u^2) = \{\emptyset\}, \{\{u_i\}_i\}, \{\emptyset\}$$

Perciò si cerca una distribuzione condizionata con questa sequenza di dipendenze:

$$\sigma_u^2 \mid \boldsymbol{u}$$

Dove si è denotato con $\boldsymbol{u}$ il vettore degli n coefficienti traslazionali con n pari al numero di istanze totale di fischi-firma a disposizione. Segue un'analisi preliminare per determinare la full-conditional, considerando solamente il Kernel $K(.)$ ed esplicitando i termini che contribuiscono esclusivamente alla costante di normalizzazione $C(.)$ in colore blu:

$$\begin{aligned} f(\sigma_u^2 \mid \boldsymbol{u}) &= \frac{K(\sigma_u^2 \mid \boldsymbol{u})}{C(\boldsymbol{u})} \\ &= \frac{f(\sigma_u^2, \boldsymbol{u})}{f(\boldsymbol{u})} = \frac{f(\boldsymbol{u} \mid \sigma_u^2) f(\sigma_u^2)}{f(\boldsymbol{u})} = \frac{\prod_{i \in I} f(u_i \mid \sigma_u^2) f(\sigma_u^2)}{f(\boldsymbol{u})} \end{aligned}$$

si osserva che per determinare la distribuzione full-conditional di σ_u^2 è sufficiente considerare il kernel di queste due distribuzioni:

- La distribuzione condizionata: $f(u_i \mid \sigma_u^2)$
- La prior di σ_u^2: $f(\sigma_u^2)$

esplicitamente:

$$K(u_i \mid \sigma_u^2) \propto \left(\sigma_u^2\right)^{-\frac{1}{2}} \exp\left(\frac{u_i^2}{\sigma_u^2}\right)$$

$$K(\sigma_u^2) \propto \left(\sigma_u^2\right)^{-(a+1)} \exp\left(-\frac{b}{\sigma_u^2}\right)$$

svolgendo il prodotto tra questi due kernel si ottiene:

$$\begin{aligned} K(\sigma_u^2 \mid \boldsymbol{u}) &\propto \prod_{i \in I} \left(\sigma_u^2\right)^{-\frac{1}{2}} \exp\left(\frac{u_i^2}{\sigma_u^2}\right) \left(\sigma_u^2\right)^{-(a+1)} \exp\left(-\frac{b}{\sigma_u^2}\right) \\ &\propto \left(\sigma_u^2\right)^{-\left(\frac{n}{2}+a+1\right)} \exp\left(-\frac{1}{\sigma_u^2}\left[\frac{1}{2}\sum_{i \in I} u_i^2 + b\right]\right) \end{aligned}$$

che si riconosce essere il kernel di una distribuzione inverse-gamma, esplicitamente:

$$\begin{aligned} \mathrm{a}_{\sigma_u^2} &= \frac{n}{2} + a \\ \mathrm{b}_{\sigma_u^2} &= \frac{1}{2}\sum_{i \in I} u_i^2 + b \end{aligned}$$

e perciò $\sigma_u^2 \mid \boldsymbol{u} \sim IG(\mathrm{a}_{\sigma_u^2}, \mathrm{b}_{\sigma_u^2})$

La Full-Conditional di $\boldsymbol{W}_k^T$

L'analisi delle dipendenze dalla rete bayesiana tramite Markov blanket è:

$$\mathrm{MB}(\boldsymbol{W}_k^T) = \left\{\alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k, \boldsymbol{\beta}^k\right\}, \left\{\{\boldsymbol{Y}_i\}_{i \in I^k}\right\}, \left\{\{u_i\}_{i \in I^k}, \left\{\tau_i^2\right\}_{i \in I^k}\right\}$$

Dove $I^k \subset I$ è l'insieme di indici delle osservazioni che vengono generate dallo stesso individuo k-esimo. Segue l'analisi preliminare per determinare la full-conditional considerando solamente il Kernel $K(.)$ ed esplicitando i termini che contribuiscono esclusivamente alla costante di normalizzazione $C(.)$ in colore

blu:

$$f\left(\boldsymbol{W}_k^T|\{\boldsymbol{Y}_i\}_{i\in I^k},\alpha^k,\psi^{2,k},\boldsymbol{\beta}^k,\{u_i\}_{i\in I^k},\left\{\tau_i^2\right\}_{i\in I^k},\boldsymbol{\theta}^k\right)=$$
$$\frac{K\left(\boldsymbol{W}_k^T|\{\boldsymbol{Y}_i\}_{i\in I^k},\alpha^k,\psi^{2,k},\boldsymbol{\beta}^k,\{u_i\}_{i\in I^k},\left\{\tau_i^2\right\}_{i\in I^k}\right)}{C\left(\{\boldsymbol{Y}_i\}_{i\in I^k},\alpha^k,\psi^{2,k},\boldsymbol{\beta}^k,\{u_i\}_{i\in I^k},\{\tau_i^2\}_{i\in I^k}\right)}$$
$$=\frac{\prod_{i\in I^k} f\left(\boldsymbol{Y}_i\mid\boldsymbol{W}_k^T,\boldsymbol{u}_i,\tau_i^2\right) f\left(\boldsymbol{W}_k^T\mid\alpha^k,\psi^{2,k},\boldsymbol{\beta}^k,\boldsymbol{\theta}^k\right) f\left(\alpha^k\right) f\left(\psi^{2,k}\right) f\left(\{u_i\}_{i\in I^k}\right) f\left(\left\{\tau_i^2\right\}_{i\in I^k}\right) f\left(\boldsymbol{\theta}^k\right)}{f\left(\{\boldsymbol{Y}_i\}_{i\in I^k},\alpha^k,\psi^{2,k},\boldsymbol{\beta}^k,\{u_i\}_{i\in I^k},\{\tau_i^2\}_{i\in I^k},\boldsymbol{\theta}^k\right)}$$

si osserva che per determinare la distribuzione full-conditional di $\boldsymbol{W}_k^T$ è sufficiente considerare il kernel di queste due distribuzioni:

- La distribuzione: $\prod_{i\in I^k} f(\boldsymbol{Y}_i\mid u_i,\boldsymbol{W}_{z_i}^T,\tau_i^2)$
- La distribuzione condizionata: $f\left(\boldsymbol{W}_k^T\mid\alpha^k,\psi^{2,k},\boldsymbol{\beta}^k,\boldsymbol{\theta}^k\right)$

esplicitamente:

$$\begin{aligned}K(\boldsymbol{y}_i\mid u_i,\boldsymbol{w},\tau_i^2)&\propto\exp\left(-\frac{1}{2}\left[\left(\boldsymbol{y}_i-\left(u_i+D_i^k\boldsymbol{w}\right)\right)^\top\left(\tau_i^2\mathbf{I}_d\right)^{-1}\left(\boldsymbol{y}_i-\left(u_i+D_i^k\boldsymbol{w}\right)\right)\right]\right)\\&\propto\exp\left(-\frac{1}{2}\left[\left(2\left(u_i^k\mathbf{1}_d-\boldsymbol{y}_i\right)+D_i^k\boldsymbol{w}\right)^\top\left(\tau_i^2\mathbf{I}_d\right)^{-1}\left(D_i^k\boldsymbol{w}\right)\right]\right)\end{aligned}$$

$$\begin{aligned}K\left(\boldsymbol{w}\mid\alpha^k,\psi^{2,k},\boldsymbol{\beta}^k,\boldsymbol{\theta}^k\right)&\propto\exp\left(-\frac{1}{2}\left[\left(\boldsymbol{w}-X^k\boldsymbol{\beta}^k\right)^\top\left(\Sigma^k\right)^{-1}\left(\boldsymbol{w}-X^k\boldsymbol{\beta}^k\right)\right]\right)\\&\propto\exp\left(-\frac{1}{2}\left[\left(\boldsymbol{w}-2X^k\boldsymbol{\beta}^k\right)^\top\left(\Sigma^k\right)^{-1}\boldsymbol{w}\right]\right)\end{aligned}$$

svolgendo il prodotto tra questi due kernel si ottiene:

$$K\left(\boldsymbol{w}\mid\{Y_i\}_{i\in I^k},\alpha^k,\psi^{2,k},\boldsymbol{\beta}^k,\{u_i\}_{i\in I^k},\left\{\tau_i^2\right\}_{i\in I^k},\boldsymbol{\theta}^k\right)\propto$$
$$\prod_{i\in I^k}\exp\left(-\frac{1}{2}\left[\left(2\left(u_i^k\mathbf{1}_d-\boldsymbol{y}_i\right)+D_i^k\boldsymbol{w}\right)^\top\left(\tau_i^2\mathbf{I}_d\right)^{-1}\left(D_i^k\boldsymbol{w}\right)\right]\right)\exp\left(-\frac{1}{2}\left[\left(\boldsymbol{w}-2X^k\boldsymbol{\beta}^k\right)^\top\left(\Sigma^k\right)^{-1}\boldsymbol{w}\right]\right)\propto$$
$$\exp\left(-\frac{1}{2}\left[\boldsymbol{w}^\top\left(\sum_{i\in I^k}\left(D_i^k\right)^\top\frac{1}{\tau_i^2}D_i^k+\left(\Sigma^k\right)^{-1}\right)\boldsymbol{w}-2\boldsymbol{w}^\top\left(\sum_{i\in I^k}\left(D_i^k\right)^\top\frac{1}{\tau_i^2}\left(\boldsymbol{y}_i-u_i\mathbf{1}_d\right)+\left(\Sigma^k\right)^{-1}\left(X^k\boldsymbol{\beta}^k\right)\right)\right]\right)$$

che si riconosce essere il kernel di una distribuzione normale multivariate, esplicitamente:

$$\mathrm{V}_{\boldsymbol{w}}^{-1}=\sum_{i\in I^k}\left(D_i^k\right)^\top\frac{1}{\tau_i^2}D_i^k+\left(\Sigma^k\right)^{-1}$$
$$\mathrm{M}_{\boldsymbol{w}}=\mathrm{V}_{\boldsymbol{w}}\left(\sum_{i\in I^k}\left(D_i^k\right)^\top\frac{1}{\tau_i^2}\left(\boldsymbol{y}_i-u_i\mathbf{1}_d\right)+\left(\Sigma^k\right)^{-1}\left(X^k\boldsymbol{\beta}^k\right)\right)$$

e perciò $\boldsymbol{W}_k^T\mid\{Y_i\}_{i\in I^k},\alpha^k,\psi^{2,k},\boldsymbol{\beta}^k,\{u_i\}_{i\in I^k},\left\{\tau_i^2\right\}_{i\in I^k},\boldsymbol{\theta}^k\sim N_{|T^k|}\left(\mathrm{M}_{\boldsymbol{w}},\mathrm{V}_{\boldsymbol{w}}\right)$

4.4 Introduzione dei Riscalamenti Temporali

Nella rete bayesiana e nella stesura delle distribuzioni full-conditional è stato indicato genericamente con $\boldsymbol{\theta}^k$ l'insieme dei coefficienti del modello che sono dediti alla modulazione. In questa sezione vengono inseriti i coefficienti utilizzati e, laddove necessario, calcolate anche le relative full-conditional.

Primo Riscalamento

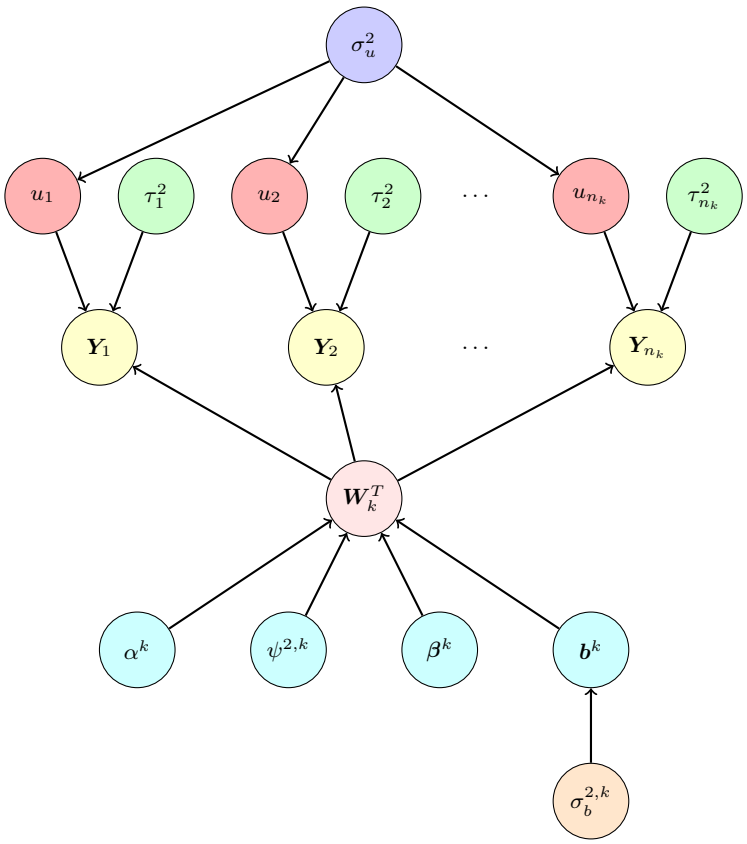
Il primo riscalamento temporale proposto è:

$$g(t) = t^c, \;\; t \in [0,1] \; , \; c \in (0, +\infty)$$

per l'algoritmo MCMC è più agevole usare una variabile che possa assumere valori su tutto $\mathbb{R}$, perciò si ridefinisce $b := \log(c) \in \mathbb{R}$. Nel modello queste variabili vengono trattate analogamente a quanto accadeva per la componente traslazionale u, infatti, ogni individuo k-esimo ha una sua propensione alla modulazione, che è funzione della varianza $\sigma_b^{2,k}$, dove più questa è alta e più tende a discostarsi dalla modulazione identità. Ogni realizzazione del fischio-firma ammette la propria modulazione intrinseca, quindi vi saranno tanti parametri di modulazione quante sono le istanze dell'esemplare, da cui la notazione b_i. La distribuzione scelta per questi coefficienti è:

$$b_i \sim N(0, \sigma_b^{2,z_i})$$

assumendo indipendenza condizionata tra i vari coefficienti di modulazione data la variabile $\sigma_b^{2,k}$. Per quanto riguarda la distribuzione a-priori di $\sigma_b^{2,k}$ si continua a seguite la scelta non-informativa rappresentata da una $IG(0.1,0.1)$. Nella seguente rappresentazione in rete bayesiana si indica con $\boldsymbol{b}^k$ il vettore degli n_k coefficienti corrispondenti alle realizzazioni dell'esemplare k-esimo:



Full-Conditional di $\sigma_b^{2,k}$

Cambiando la struttura del grafo, cambiano anche le dipendenze ed è perciò necessario calcolare la full-conditional per la varianza delle modulazioni. L'analisi delle dipendenze dalla rete bayesiana tramite Markov blanket è:

$$\mathrm{MB}(\sigma_b^{2,k}) = \{\emptyset\}, \left\{\{b_i\}_{i\in I^k}\right\}, \{\emptyset\}$$

Perciò si cerca una distribuzione condizionata con questa sequenza di dipendenze:

$$\sigma_b^{2,k} \mid \boldsymbol{b}^k$$

Dove si è denotato con $\boldsymbol{b}^k$ il vettore degli n_k coefficienti di modulazione con n_k pari al numero di istanze totale di fischi-firma a disposizione dell'individuo k-esimo e $I^k \subset I$ il sotto-insieme di indici corrispondente. Segue un'analisi preliminare per determinare la full-conditional, considerando solamente il Kernel $K(.)$ ed esplicitando i termini che contribuiscono esclusivamente alla costante di normalizzazione $C(.)$ in colore blu:

$$\begin{aligned} f(\sigma_b^{2,k} \mid \boldsymbol{b}^k) &= \frac{K(\sigma_b^{2,k} \mid \boldsymbol{b}^k)}{C(\boldsymbol{b}^k)} \\ &= \frac{f(\sigma_b^{2,k}, \boldsymbol{b}^k)}{f(\boldsymbol{b}^k)} = \frac{f(\boldsymbol{b}^k \mid \sigma_b^{2,k}) f(\sigma_b^{2,k})}{f(\boldsymbol{b}^k)} = \frac{\prod_{i\in I^k} f(b_i^k \mid \sigma_b^{2,k}) f(\sigma_b^{2,k})}{f(\boldsymbol{b}^k)} \end{aligned}$$

si osserva che per determinare la distribuzione full-conditional di σ_u^2 è sufficiente considerare il kernel di queste due distribuzioni:

- La distribuzione condizionata: $f(b_i^k \mid \sigma_b^{2,k})$
- La prior di $\sigma_b^{2,k}$: $f(\sigma_b^{2,k})$

esplicitamente: (si usano α e β anziché a e b come parametri generici delle $IG(\alpha, \beta)$ per evitare confusioni con il b delle modulazioni)

$$K(b_i^k \mid \sigma_b^{2,k}) \propto \left(\sigma_b^{2,k}\right)^{-\frac{1}{2}} \exp\left(\frac{\left(b_i^k\right)^2}{\sigma_b^{2,k}}\right)$$

$$K(\sigma_b^{2,k}) \propto \left(\sigma_b^{2,k}\right)^{-(\alpha+1)} \exp\left(-\frac{\beta}{\sigma_b^{2,k}}\right)$$

svolgendo il prodotto tra questi due kernel si ottiene:

$$\begin{aligned} K(\sigma_b^{2,k} \mid \boldsymbol{b}^k) &\propto \prod_{i\in I^k} \left(\sigma_b^{2,k}\right)^{-\frac{1}{2}} \exp\left(\frac{\left(b_i^k\right)^2}{\sigma_b^{2,k}}\right) \left(\sigma_b^{2,k}\right)^{-(\alpha+1)} \exp\left(-\frac{\beta}{\sigma_b^{2,k}}\right) \\ &\propto \left(\sigma_b^{2,k}\right)^{-\left(\frac{n_k}{2}+\alpha+1\right)} \exp\left(-\frac{1}{\sigma_b^{2,k}}\left[\frac{1}{2}\sum_{i\in I^k}\left(b_i^k\right)^2 + \beta\right]\right) \end{aligned}$$

che si riconosce essere il kernel di una distribuzione inverse-gamma, esplicitamente:

$$\begin{aligned} \alpha_{\sigma_b^{2,k}} &= \frac{n_k}{2} + \alpha \\ \beta_{\sigma_b^{2,k}} &= \frac{1}{2}\sum_{i\in I^k}\left(b_i^k\right)^2 + \beta \end{aligned}$$

e perciò $\sigma_b^{2,k} \mid \boldsymbol{b}^k \sim IG(\alpha_{\sigma_b^{2,k}}, \beta_{\sigma_b^{2,k}})$

Secondo Riscalamento

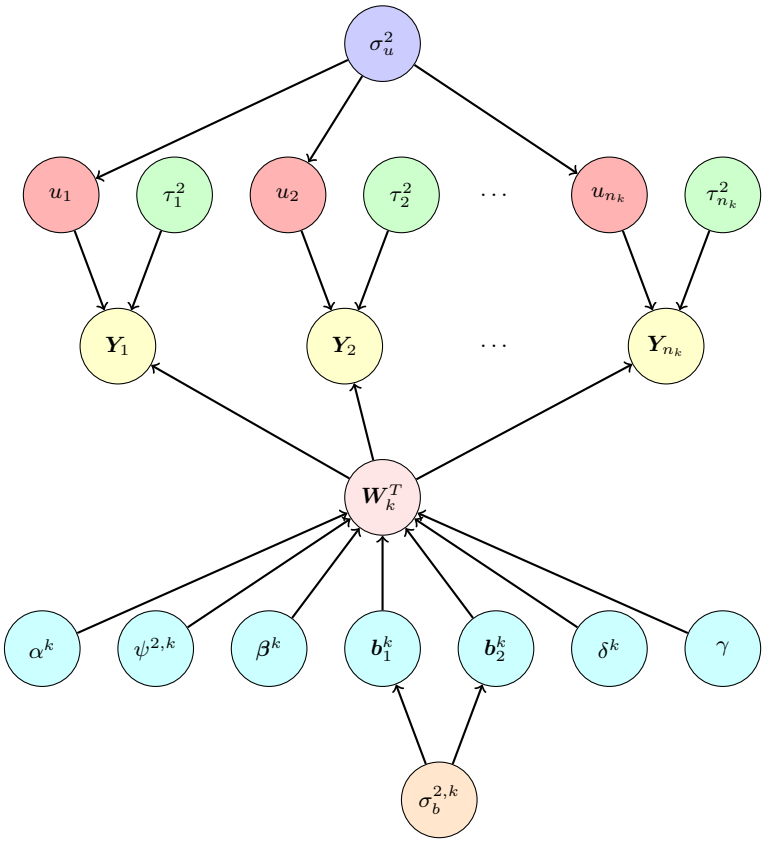
La seconda famiglia di riscalamenti temporali, con il solito abuso di notazione $\gamma \in (0, +\infty]$:

$$g(t) = t^{c_1} + (t^{c_2} - t^{c_1})\frac{1}{1 + e^{-\gamma(t-\delta)}},\ t \in [0,1],\ c_1, c_2 \in (0, +\infty), \delta \in [0,1]\ \gamma \in (0, +\infty]$$

come anticipato nella prima proposta, si effettua una trasformazione per ottenere variabili che possano assumere valori su tutto $\mathbb{R}$, perciò si ridefinisce $b_1 := \log(c_1),\ b_2 := \log(c_2) \in \mathbb{R}$. Insieme a queste variabili viene introdotta nuovamente la rappresentazione della varianza: $\sigma_b^{2,k}$. La distribuzione a-priori scelta per questi coefficienti è:

$$b_{1,i} \sim N(0, \sigma_b^{2,z_i}),\ b_{2,i} \sim N(0, \sigma_b^{2,z_i})$$

assumendo indipendenza condizionata tra i vari coefficienti di modulazione, compresi quelli inerenti alla stessa istanza, una volta data la variabile $\sigma_b^{2,k}$. Per quanto riguarda la distribuzione a-priori di $\sigma_b^{2,k}$ si continua a seguire la scelta non-informativa rappresentata da una $IG(0.1,0.1)$. Per il coefficienti δ^k si sceglie la distribuzione prior $U(0,1)$ e per γ invece una $IG(0.1, 10)$, anch'esse non-informative. Nella seguente rappresentazione in rete bayesiana si indicano, rispettivamente, con $\boldsymbol{b}_1^k$ e $\boldsymbol{b}_2^k$ i vettori degli n_k coefficienti corrispondenti alla prima modulazione e alla seconda modulazione dell'esemplare k-esimo:



Full-Conditional di $\sigma_b^{2,k}$

L'analisi delle dipendenze dalla rete bayesiana tramite Markov blanket è:

$$\mathrm{MB}(\sigma_b^{2,k}) = \{\emptyset\}, \left\{\{b_{1,i}\}_{i \in I^k}, \{b_{2,i}\}_{i \in I^k}\right\}, \{\emptyset\}$$

Perciò si cerca una distribuzione condizionata con questa sequenza di dipendenze:

$$\sigma_b^{2,k} \mid \boldsymbol{b}_1^k, \boldsymbol{b}_2^k$$

Dove si è denotato, rispettivamente, con $\boldsymbol{b}_1^k$ e $\boldsymbol{b}_2^k$, il vettore delle n_k coppie di coefficienti di prime e seconda modulazione con n_k pari al numero di istanze totale di fischi-firma a disposizione dell'individuo k-esimo e $I^k \subset I$ il sotto-insieme di indici corrispondente. Segue un'analisi preliminare per determinare la full-conditional, considerando solamente il Kernel $K(.)$ ed esplicitando i termini che contribuiscono esclusivamente alla costante di normalizzazione $C(.)$ in colore blu:

$$\begin{aligned} f(\sigma_b^{2,k} \mid \boldsymbol{b}_1^k, \boldsymbol{b}_2^k) &= \frac{K(\sigma_b^{2,k} \mid \boldsymbol{b}_1^k, \boldsymbol{b}_2^k)}{C(\boldsymbol{b}_1^k, \boldsymbol{b}_2^k)} \\ &= \frac{f(\sigma_b^{2,k}, \boldsymbol{b}_1^k, \boldsymbol{b}_2^k)}{f(\boldsymbol{b}_1^k, \boldsymbol{b}_2^k)} = \frac{f(\boldsymbol{b}_1^k, \boldsymbol{b}_2^k \mid \sigma_b^{2,k}) f(\sigma_b^{2,k})}{f\boldsymbol{b}_1^k, \boldsymbol{b}_2^k)} \\ &= \frac{\prod_{i \in I^k} f(b_{1,i}^k \mid \sigma_b^{2,k}) f(b_{2,i}^k \mid \sigma_b^{2,k}) f(\sigma_b^{2,k})}{f(\boldsymbol{b}_1^k, \boldsymbol{b}_2^k)} \end{aligned}$$

si osserva che per determinare la distribuzione full-conditional di σ_u^2 è sufficiente considerare il kernel di queste distribuzioni:

- Le distribuzioni condizionate: $f(b_{1,i}^k \mid \sigma_b^{2,k})$ e $f(b_{2,i}^k \mid \sigma_b^{2,k})$
- La prior di $\sigma_b^{2,k}$: $f(\sigma_b^{2,k})$

esplicitamente: (si usano α e β anziché a e b come parametri generici delle $IG(\alpha, \beta)$ per evitare confusioni con il b delle modulazioni)

$$K(b_{1,i}^k \mid \sigma_b^{2,k}) \propto \left(\sigma_b^{2,k}\right)^{-\frac{1}{2}} \exp\left(\frac{\left(b_{1,i}^k\right)^2}{\sigma_b^{2,k}}\right)$$

$$K(b_{2,i}^k \mid \sigma_b^{2,k}) \propto \left(\sigma_b^{2,k}\right)^{-\frac{1}{2}} \exp\left(\frac{\left(b_{2,i}^k\right)^2}{\sigma_b^{2,k}}\right)$$

$$K(\sigma_b^{2,k}) \propto \left(\sigma_b^{2,k}\right)^{-(\alpha+1)} \exp\left(-\frac{\beta}{\sigma_b^{2,k}}\right)$$

svolgendo il prodotto tra questi kernel si ottiene:

$$\begin{aligned} K(\sigma_b^{2,k} \mid \boldsymbol{b}^k) &\propto \prod_{i \in I^k} K(b_{1,i}^k \mid \sigma_b^{2,k}) K(b_{2,i}^k \mid \sigma_b^{2,k}) \left(\sigma_b^{2,k}\right)^{-(\alpha+1)} \exp\left(-\frac{\beta}{\sigma_b^{2,k}}\right) \\ &\propto \left(\sigma_b^{2,k}\right)^{-(n_k+\alpha+1)} \exp\left(-\frac{1}{\sigma_b^{2,k}}\left[\frac{1}{2}\left(\sum_{i \in I^k} \left(b_{1,i}^k\right)^2 + \left(b_{2,i}^k\right)^2\right) + \beta\right]\right) \end{aligned}$$

che si riconosce essere il kernel di una distribuzione inverse-gamma, esplicitamente:

$$\begin{aligned} \alpha_{\sigma_b^{2,k}} &= n_k + \alpha \\ \beta_{\sigma_b^{2,k}} &= \frac{1}{2}\left(\sum_{i \in I^k} \left(b_{1,i}^k\right)^2 + \left(b_{2,i}^k\right)^2\right) + \beta \end{aligned}$$

e perciò $\sigma_b^{2,k} \mid \boldsymbol{b}_1^k, \boldsymbol{b}_2^k \sim IG(\alpha_{\sigma_b^{2,k}}, \beta_{\sigma_b^{2,k}})$

Terzo Riscalamento

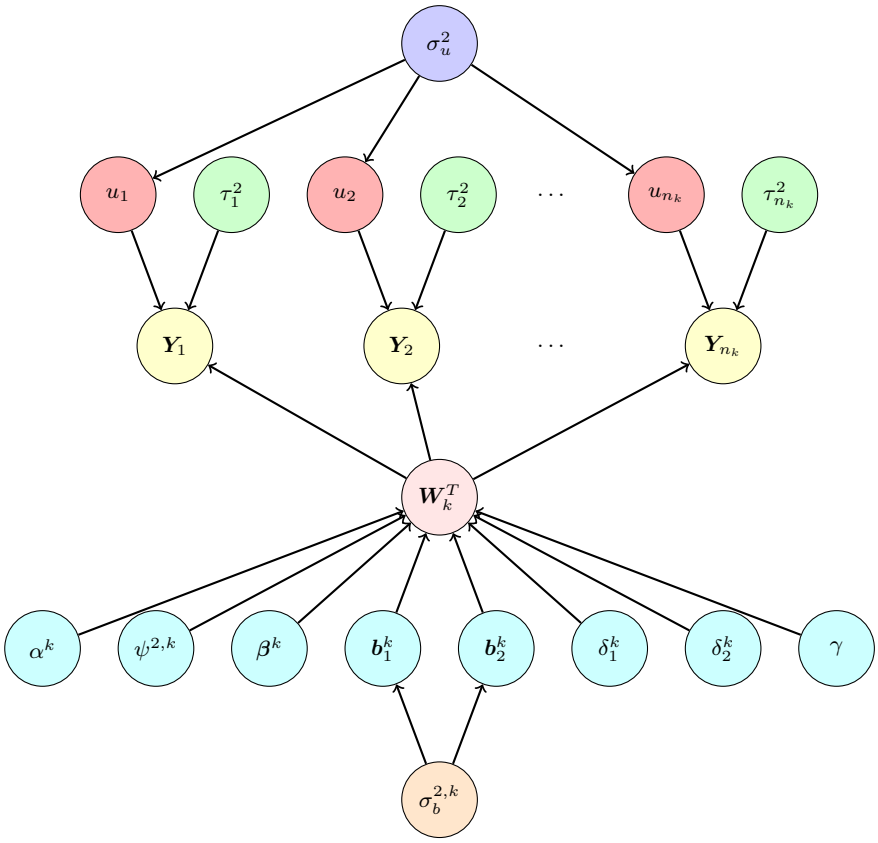
La terza famiglia di riscalamenti:

$$g(t) = t^{c_1} + \frac{t^{c_2} - t^{c_1}}{1 + e^{-\gamma(t-\delta_1)}} + \frac{t^{c_1} - t^{c_2}}{1 + e^{-\gamma(t-\delta_2)}},$$
$$t \in [0,1],\ c_1, c_2 \in (0, +\infty),\ 0 \leq \delta_1 \leq \delta_2 \leq 1,\ \gamma \in (0, +\infty]$$

La trasformazione per ottenere variabili che possano assumere valori su tutto $\mathbb{R}$, rimane invariata ai casi precedenti: $b_1 := \log(c_1),\ b_2 := \log(c_2) \in \mathbb{R}$. Così come rimane invariata la rappresentazione della varianza: $\sigma_b^{2,k}$ e le loro distribuzioni a priori. Per i coefficienti δ_1^k e δ_2^k si sceglie la coppia di distribuzioni:

$$\delta_1^k \sim U(0,1),\ \delta_2^k \mid \delta_1^k \sim U(\delta_1^k, 1)$$

per γ si sceglie nuovamente $IG(0.1, 10)$. La rappresentazione in rete bayesiana diventa:



Ai fini dell'implementazione non è necessario ricavare full-conditional per questa modulazione, siccome per $\sigma_b^{2,k}$ rimane invariata al caso della seconda proposta.

Capitolo 5

Algoritmo MCMC

5.1 Introduzione dell'Algoritmo

L'algoritmo implementato appartiene alla famiglia dei metodi Markov chain Monte Carlo (MCMC) e adotta una strategia di campionamento Metropolis-within-Gibbs, con approccio adattivo per la determinazione delle proposte Metropolis [45]. Le variabili aggiornate attraverso il campionamento Metropolis-Hastings riguardano le entità del modello che sono a stretto contatto con il processo genesi $\boldsymbol{W}_k^T$ e con il riscalamento temporale. Nello specifico vengono proposte insieme le due variabili scalari α_k, ψ_k^2 congiuntamente ad una realizzazione del processo genesi, ottenuta mediante full-conditional; per preservare coerenza tra coefficienti e processo, aumentando l'acceptance rate. Le proposte di α_k, ψ_k^2 sono simmetriche, e questo permette di semplificare i calcoli nel rapporto Metropolis. A seguire vi è un campionamento partizionato delle variabili di modulazione $\boldsymbol{\theta}^k$, congiunto anch'esso ad una realizzazione del processo genesi. Vista la natura computazionalmente complessa del problema, la gestione delle matrici di varianza e covarianza avviene mediante la decomposizione di Cholesky, metodo ben noto per la sua efficienza con matrici hermitiane (i.e. "simmetriche" se ha valori reali) e semi-definite positive [46].

5.2 Algoritmo per le Proposte

Algoritmo 4

L'algoritmo 4, descritto nell'articolo di Christophe Andrieu e Johannes Thoms [45], delinea un approccio Metropolis-Hastings (MH) con varianza adattiva. Questo algoritmo ottimizza dinamicamente la varianza della distribuzione proposta durante la simulazione, garantendo un campionamento bilanciato dalla distribuzione desiderata. Il metodo ha lo scopo di raggiungere un equilibrio tra "exploration" ed "exploitation", evitando la scelta diretta della varianza delle proposte, che potrebbe compromettere la convergenza del metodo. E' noto, infatti, che la componente chiave dell'algoritmo Metropolis-Hastings è la distribuzione proposta, $q(x^*, x)$, che genera le transizioni candidate. La sua varianza influisce significativamente sull'efficienza dell'algoritmo siccome, generalmente, se la varianza è troppo elevata questo porta a bassi acceptance rate, poiché l'esplorazione dello spazio degli stati è troppo sparsa e le proposte cadono in regioni a bassa probabilità. Invece, varianza bassa, provoca proposte molto concentrate, rallentando l'esplorazione e aumentando l'autocorrelazione tra i valori campionati. Questo rende l'utilizzo dell'algoritmo 4 particolarmente efficace nei problemi ad alta dimensionalità, dove la scelta della varianza della proposta può influenzare notevolmente le prestazioni. Lo pseudo codice di riferimento è l'algoritmo 4.

Entrando più nel dettaglio dell'algoritmo, si notano due entità fondamentali, λ_i e $\bar{\alpha}_*$. Il coefficiente λ_i moltiplica la matrice di varianza e covarianza delle proposte, rendendo più sparsa (λ_i elevato) o più

Algorithm 4 AM Algorithm with Global Adaptive Scaling

1: Inizializzazione: $\boldsymbol{X}_0$, μ_0, Σ_0, e λ_0
2: **for** $i = 0$ to $N - 1$ **do**
3: Campiona $\boldsymbol{Y}_{i+1} \sim \mathcal{N}(\boldsymbol{X}_i, \lambda_i \Sigma_i)$ e poni $\boldsymbol{X}_{i+1} \leftarrow \boldsymbol{Y}_{i+1}$, con probabilità $\alpha(\boldsymbol{X}_i, \boldsymbol{Y}_{i+1})$, altrimenti $\boldsymbol{X}_{i+1} \leftarrow \boldsymbol{X}_i$
4: Aggiorna:

$$\begin{aligned} \log(\lambda_{i+1}) &\leftarrow \log(\lambda_i) + \gamma_{i+1}[\alpha(\boldsymbol{X}_i, \boldsymbol{Y}_{i+1}) - \bar{\alpha}_*], \\ \mu_{i+1} &\leftarrow \mu_i + \gamma_{i+1}(\boldsymbol{X}_{i+1} - \mu_i), \\ \Sigma_{i+1} &\leftarrow \Sigma_i + \gamma_{i+1}[(\boldsymbol{X}_{i+1} - \mu_i)(\boldsymbol{X}_{i+1} - \mu_i)^\top - \Sigma_i]. \end{aligned}$$

5: **end for**

Algorithm 5 Componentwise AM with Componentwise Adaptive Scaling

1: Inizializzazione: $\boldsymbol{X}_0$, μ_0, Σ_0, e $\lambda_0^1, \ldots, \lambda_0^{n_x}$
2: **for** $i = 0$ to $N - 1$ **do**
3: Scegli un componente $k \sim \mathcal{U}\{1, \ldots, n_x\}$
4: Campiona $\boldsymbol{Y}_{i+1} \sim \boldsymbol{X}_i + e_k \mathcal{N}(0, \lambda_i^k [\Sigma_i]_{k,k})$ e poni $\boldsymbol{X}_{i+1} \leftarrow \boldsymbol{Y}_{i+1}$, con probabilità $\alpha(\boldsymbol{X}_i, \boldsymbol{Y}_{i+1})$, altrimenti $\boldsymbol{X}_{i+1} \leftarrow \boldsymbol{X}_i$
5: Aggiorna:

$$\begin{aligned} \log(\lambda_{i+1}^k) &\leftarrow \log(\lambda_i^k) + \gamma_{i+1}[\alpha(\boldsymbol{X}_i, \boldsymbol{Y}_{i+1}) - \bar{\alpha}_*], \\ \mu_{i+1} &\leftarrow \mu_i + \gamma_{i+1}(\boldsymbol{X}_{i+1} - \mu_i), \\ \Sigma_{i+1} &\leftarrow \Sigma_i + \gamma_{i+1}[(\boldsymbol{X}_{i+1} - \mu_i)(\boldsymbol{X}_{i+1} - \mu_i)^\top - \Sigma_i], \end{aligned}$$

e poni $\lambda_{i+1}^j \leftarrow \lambda_i^j$ per $j \neq k$
6: **end for**

concentrata (λ_i piccolo) l'esplorazione dello spazio degli stati. In fase implementativa, tale valore viene aggiornato dopo l'esecuzione di "batch" di iterazioni, nello specifico, dopo ogni batch-iterazioni si calcola l'*acceptance rate provvisorio*, $\alpha(\boldsymbol{X}_i, \boldsymbol{Y}_{i+1})$, come il valore medio dei Rapporti Metropolis calcolati durante le iterazioni all'interno del batch. Se il rapporto Metropolis medio risulta maggiore dell'*acceptance rate desiderato*, $\bar{\alpha}_*$, allora viene incrementato il valore di λ_i attraverso l'incremento del suo logaritmo, che è una funzione monotona crescente, questo invita quindi ad aumentare la sparsità delle proposte e quindi ad abbassare l'acceptance rate provvisorio; accade dunque l'opposto se $\bar{\alpha}_* > \alpha(\boldsymbol{X}_i, \boldsymbol{Y}_{i+1})$. Infine si nota il coefficiente γ, il quale modula "l'aggressività" con cui vengono apportate le modifiche alla media μ e alla matrice di varianza e covarianza Σ. Più vicino γ risulta ad 1 e meno memoria ha il modello sul valore passato della quantità aggiornata. Nel caso estremo di $\gamma = 1$ il valore medio e la matrice di varianza e covarianza cambiano ad ogni iterazione, mostrando memoria nulla rispetto ai passi precedenti. Questo parametro viene ridotto all'aumentare delle iterazioni, siccome i primi passi saranno quelli più "inesatti" e quindi è coerente pesino di meno.

Algoritmo 5

L'algoritmo appena presentato può essere migliorato, infatti, una criticità evidente è l'azione "globale" del coefficiente λ, il quale riscala media e matrice di varianza e covarianza, di tutte le variabili coinvolte nella proposta, dello stesso fattore. Anche se esso può risultare opportuno per alcune entrate del vettore, probabilmente non è ottimale per tutte. Per sopperire a questa mancanza di raffinatezza, viene introdotto l'algoritmo 5, il quale agisce in modo separato per le varie componenti del vettore proposto, ottenendo dei riscalamenti individuali per ogni variabile.

5.2.1 Le Entità Proposte nel Metropolis

Le componenti del modello per le quali non sono state esplicitamente calcolate le full-conditional sono aggiornate dall'algoritmo MCMC mediante passi Metropolis. Tuttavia, prima di descrivere l'intero algoritmo, è fondamentale sottolineare che per garantire un acceptance rate elevato, senza compromettere la capacità dell'algoritmo di esplorare efficacemente lo spazio degli stati, è necessario proporre congiuntamente alle quantità responsabili della riscalatura temporale anche un nuovo processo genesi. Nello specifico, si consideri lo scenario in cui il vettore dei coefficienti di modulazione $\boldsymbol{\theta}$ viene proposto senza un nuovo processo. In tale caso, quando i coefficienti vengono proposti separatamente al processo, essi causano un riscalamento temporale che conduce ad una ridisposizione dei valori del processo vecchio sui nuovi tempi. Il processo, tuttavia, era stato campionato su altri valori temporali che hanno definito la sua distribuzione mediante specifici valori di covarianza esponenziale, che chiaramente sono privi di significato una volta cambiata la scala temporale, siccome vale:

$$Cov(X_{t_1}, X_{t_2}) = \frac{\psi^2}{1-\alpha^2} \exp(\log(\alpha)|t_1 - t_2|),\ t_1, t_2 \in I,\ \psi^2 > 0,\ \alpha \in (0,1)$$

ed è inverosimile che la realizzazione dei coefficienti di modulazione conduca ad un riscalamento dei tempi identico a quello passato. Chiaramente questo scenario condurrebbe ad un rate di accettazione molto basso e ad un'ovvia necessità dell'algoritmo di proporre valori di modulazione praticamente identici a quelli precedenti, forzando quindi l'algoritmo MCMC a muoversi poco. Inoltre, siccome l'interazione tra la scala temporale ed il processo è così forte, la semplice ridisposizione dei valori del processo vecchio sui tempi nuovi condurrebbe ad una mancata corrispondenza con i dati, che verosimilmente erano invece descritti bene dal vecchio processo sui vecchi tempi, portando conseguentemente ad una drastica riduzione della probabilità del nuovo stato proposto, sia nella sua densità $W_k^T \mid \boldsymbol{\beta}^k, \alpha^k, \psi^{2,k}, \boldsymbol{\theta}^k \sim N_{|T^k|}(X^k\boldsymbol{\beta}^k, \Sigma(\alpha_k, \psi_k^2))$ che nella densità $Y_i(t) \mid W_{z_i}^T, u_i, \tau_i^2, z_i \sim N(u_i + W_{z_i}^T(s), \tau_i^2), \quad s = g(t), \quad t \in [0,1]$, il ché si traduce in un valore del numeratore del rapporto Metropolis sproporzionatamente piccolo rispetto al denominatore, siccome quest'ultimo rispecchia più coerenza tra dati e processo. Al contrario, proporre i coefficienti in modo congiunto consente di modificare in modo coerente la scala temporale e il processo risultante. Di conseguenza, il numeratore e il denominatore del rapporto di Metropolis sono meglio bilanciati, il che porta a una maggiore probabilità di accettazione. Generalmente, infatti, se due o più parametri sono altamente interdipendenti, è necessario assicurarsi che vengano aggiornati insieme come un "blocco". Se ciò non viene fatto, la convergenza del processo alla sua distribuzione stazionaria potrebbe richiedere un numero eccessivo di iterazioni [47].

5.3 Random Scan-Sampling

Una caratteristica distintiva dell'algoritmo MCMC proposto riguarda la scelta dell'ordine con cui le variabili del modello vengono campionate. La letteratura offre diverse interpretazioni in merito, poiché la dicotomia tra Systematic-Scan Sampling e Random-Scan Sampling non ha ancora trovato un consenso definitivo. Diaconis Persi, in un'opera del 2008 [48], evidenzia l'assenza di prove conclusive a favore di una strategia rispetto all'altra, suggerendo tuttavia che il tema non sia stato oggetto di un'analisi approfondita. Più recentemente, nel 2021, sono stati presentati esempi che mostrano come le due strategie possano determinare velocità di convergenza differenti, con particolari strutture che favoriscono l'una o l'altra metodologia di campionamento [49]. Nello specifico si parla di Systematic-Scan Sampling quando le variabili vengono campionate in modo predeterminato, scegliendo un ordine e mantenendolo per ogni iterazione. Mentre con Random-Scan Sampling la selezione della variabile da campionare avviene in modo casuale e quindi tende a cambiare per ogni iterazione. Sul tema del Random-Scan Sampling vi sono state delle interessanti indagini, che hanno mostrato come un metodo adattivo possa condurre a convergenze più rapide, impostando campionamenti più frequenti verso le variabili più complesse da stimare [50]. In questa trattazione, tuttavia, l'analisi si focalizza sulla formulazione più semplice del Random-Scan, la quale seleziona in modo uniforme l'ordine delle variabile da campionare. Come mostrato nella sezione finale di analisi dei risultati, il modello MCMC proposto sembra preferire un certo tipo di sequenzialità di campionamento. Nello specifico, per come è stato descritto il modello, il contributo medio della componente traslazionale $\boldsymbol{u}$ dovrebbe essere a media nulla. Questa intuizione sembra andare in contrasto con la forma della sua full-conditional, se il suo campionamento avviene per primo:

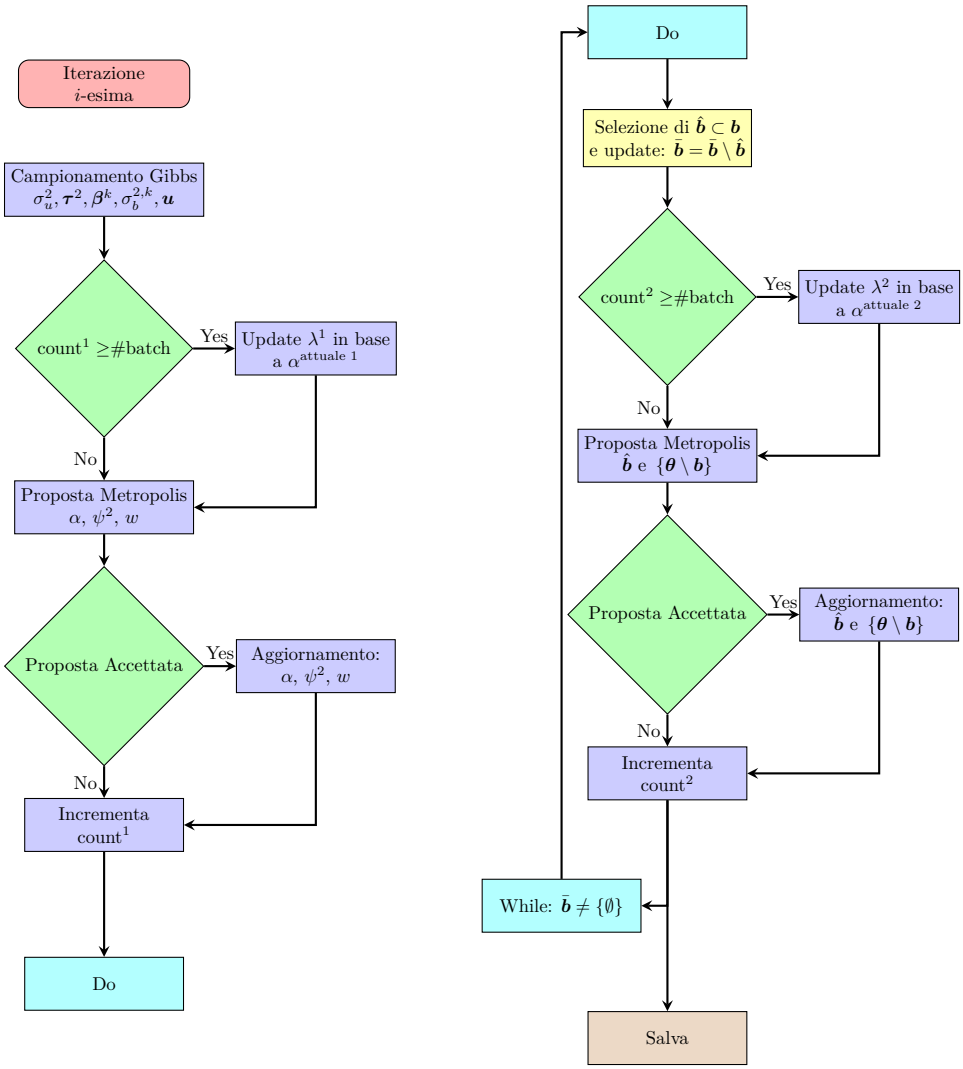
$$\mathrm{M}_u = \frac{\left[\sum_{j=1}^{d} y_{ij} - \left(D_i^k \boldsymbol{w}\right)_j\right]}{\tau_i^2} \frac{\tau_i^2 \sigma_u^2}{d\sigma_u^2 + \tau_i^2}$$
$$\mathrm{V}_u = \frac{\tau_i^2 \sigma_u^2}{d\sigma_u^2 + \tau_i^2}$$

con $u_i \mid \boldsymbol{Y}_i, \boldsymbol{W}_{z_i}^T, \tau_i^2, \sigma_u^2 \sim N(\mathrm{M}_u, \mathrm{V}_u)$.

Infatti, con inizializzazione del processo $W_{z_i}^T$ a rumore bianco, si nota come la componente $\mathrm{M}_u = \frac{\left[\sum_{j=1}^{d} y_{ij} - \left(D_i^k \boldsymbol{w}\right)_j\right]}{\tau_i^2} \frac{\tau_i^2 \sigma_u^2}{d\sigma_u^2 + \tau_i^2}$, almeno in una prima fase iniziale, si prenda carico della discrepanza tra dati e processo. Dunque, verosimilmente, la componente u_i in questione assorbirà l'effetto dell'intercetta $\beta_0^{z_i}$. Per mitigare questo effetto risulta favorevole iniziare campionando prima il vettore di coefficienti $\boldsymbol{\beta}^k$ ed in seguito il vettore $\boldsymbol{u}$. L'analisi nella sezione dedicata alla selezione del modello mostra come questo fenomeno rallenti la convergenza della catena delle $\boldsymbol{u}$ e, dunque, del processo complessivo. Siccome vi sono possibili indizi che suggeriscono l'esistenza di una sequenzialità preferibile, e visto che l'analisi condotta non verte verso l'indagine di essa, si preferisce continuare con l'approccio Random-Scan Sampling.

5.4 Diagramma di Flusso

Prima della descrizione dettagliata dell'algoritmo MCMC viene presentato il suo diagramma di flusso semplificato, privo della strategia Random-Scam sampling per agevolarne la comprensione generale.



5.5 Spiegazione dell'Algoritmo

Nell'algoritmo MCMC rappresentato nel diagramma in 5.4, si mostra una prima fase di campionamento delle variabiabili aleatorie: $\sigma_u^2, \boldsymbol{\tau}^2, \boldsymbol{\beta}^k, \sigma_b^{2,k}, \boldsymbol{u}$ mediante Gibbs Sampling, a cui segue una seconda fase di proposte tramite passi Metropolis. La proposta delle variabili $\alpha^k, \psi^{2,k}$ avviene congiuntamente ad una proposta dalla full-condtional di $\boldsymbol{W}_k^T$, ottenuta usfruendo delle nuove variabili proposte. La logica di proposta congiunta anche qui è ben giustificata, siccome le variabili $\alpha^k, \psi^{2,k}$ modellano la matrice di varianza e covarianza del processo, il quale necessita dunque di cambiare per essere coerente ad essi e poter "competere" col processo vecchio nel rapporto Metropolis. In ultimo stadio vi è la proposta delle variabili dedite alla modulazione temporale. Prima di procedere è bene rimarcare come questa descrizione sia in realtà semplificata, e che la suddivisione in fasi non è realmente mantenuta. Infatti le variabili non seguono ad ogni iterazione la stessa sequenzialità, ma vengono ridistribuite in modo casuale. Entrando nel merito della modulazione temporale, si noti innanzitutto come il vettore dei coefficienti $\boldsymbol{\theta}^k$, si può dividere in due sottoinsiemi: i coefficienti che sono in relazione esponenziale con la variabile temporale, racchiuse nel vettore $\boldsymbol{b}^k$, e tutte le altre variabili $\boldsymbol{\theta}^k \setminus \boldsymbol{b}^k$ che sono specifiche per ogni riscalamento scelto. Esplcitamente:

- Nel primo riscalamento: $g(t) = t^c$ con $t \in [0,1]$ e $c := \log(b) \in (0, +\infty)$, non vi sono altre variabili modulanti al di fuori di quelle esponenziali.
- Nella seconda famiglia di modulazione: $g(t) = t^{c_1} + (t^{c_2} - t^{c_1})\frac{1}{1+e^{-k(t-\delta)}}$, $t \in [0,1]$, $c_1 := \log(b_1), c_2 := \log(b_2) \in (0, +\infty)$, $\delta \in [0,1]$ $\gamma \in (0, +\infty]$ vi sono le variabili δ e γ.
- Mentre nell'ultima: $g(t) = t^{c_1} + \frac{t^{c_2} - t^{c_1}}{1+e^{-k(t-\delta_1)}} + \frac{t^{c_1} - t^{c_2}}{1+e^{-k(t-\delta_2)}}$ con $t \in [0,1], c_1 := \log(b_1), c_2 := \log(b_2) \in (0, +\infty), 0 \leq \delta_1 \leq \delta_2 \leq 1, \gamma \in (0, +\infty]$ compaiono δ_1, δ_2, γ.

Siccome i coefficienti modulanti b_i^k a disposizione per ogni individuo k-esimo posso risultare numerosi, non è ottimale proporre insieme il vettore $\boldsymbol{b}^k$. Questo perché, anche se il vettore proposto è complessivamente aderente a quello desiderato, è sufficiente una sola istanza fuori scala per compromettere l'intero vettore proposto, portandolo ad essere rifiutato. Insomma, più la dimensionalità della proposta aumenta e più è difficile proporre in modo convincente senza che nessun elemento assuma un valore compromettente. Questo giustifica dunque la fase di selezione ed introduce il ciclo che si vede nel diagramma di flusso. Partendo dall'assegnazione $\bar{\boldsymbol{b}}^k \leftarrow \boldsymbol{b}^k$, per ogni iterazione del ciclo si seleziona in modo casuale, senza re-inserimento, un sottoinsieme $\hat{\boldsymbol{b}}^k \subset \bar{\boldsymbol{b}}^k$, dove la dimensione di questo sottoinsieme può essere inserita come iper-parametro del modello. Una volta selezionato $\hat{\boldsymbol{b}}^k$ esso viene proposto congiuntamente a tutte le altre variabili di modulazione non-esponenziale, i.e. $\boldsymbol{\theta}^k \setminus \boldsymbol{b}^k$, ad una proposta da full-conditional del processo $\boldsymbol{W}_k^T$, e al restante insieme $\boldsymbol{b}^k \setminus \hat{\boldsymbol{b}}^k$ invariato. Come di consueto si procede con il calcolo del rapporto Metropolis ed il meccanismo di accettazione/rifiuto. Il ciclo continua fin tanto che il vettore $\bar{\boldsymbol{b}}^k$ non coincide con l'insieme nullo $\{\emptyset\}$.

L'ultimo nodo da sciogliere riguarda la dinamica di come avvengono queste proposte e del perché avvengono due fasi separate di proposta Metropolis. La dinamica di proposta risulta essere una via di mezzo tra l'algoritmo 4 e l'algoritmo 5, in cui la prima e la seconda fase di proposta vivono in due realtà separate e aggiornano il coefficiente λ in modo indipendente, risultando quindi in un metodo che non è né globale, come l'algoritmo 4, né individuale come l'algoritmo 5. Questo approccio è stato scelto per via della diversa semantica tra i due gruppi all'interno del modello, infatti, mentre α^k e $\psi^{2,k}$ servono per la struttura del processo genesi, $\boldsymbol{\theta}^k$ si occupa della sua collocazione sull'asse temporale. E, come si può immaginare, questa semantica differente implica anche una trattazione separata dei loro aggiornamenti. Insomma, la trattazione congiunta di queste due parti risulterebbe grossolana, mescolando effetti diversi e causando delle adattazioni di varianza imprecise.

Capitolo 6

Analisi Dati Sintetici e Selezione dei Modelli

6.1 Procedura di Analisi dei Modelli e Dati Sintetici

Nei passati capitoli sono stati introdotti diversi modelli per rappresentare i fischi-firma, oltre che a diverse strutture algoritmiche; questo impone la necessità di stabilire un'analisi metodologica che permetta di concentrare l'attenzione progressivamente sulle modulazioni temporali e le strutture algoritmiche più promettenti. Innanzitutto, è bene sottolineare che nella fase iniziale vi saranno diverse analisi con dati sintetici, i.e. dati che vengono simulati per evincere alcune caratteristiche sulle prestazioni del modello. Questa procedura offre il chiaro vantaggio di poter focalizzare l'attenzione su componenti dell'algoritmo in maniera specifica, confrontando le stime con la configurazione di parametri "vera" che ha generato i dati. In aggiunta, è possibile progettare simulazioni complesse per mettere alla prova l'efficacia dei metodi, evidenziandone le principali criticità. Questo approccio consente di identificare le condizioni operative ottimali e di definire i contesti specifici in cui il modello può essere considerato affidabile. E' cruciale non fondare le proprie conclusioni analizzando esclusivamente i risultati sui dati reali; in questo caso soprattuto, siccome tutti i parametri stimabili dall'algoritmo non sono noti nei dati reali, e l'unico tipo di giudizio effettuabile è sulla prossimità dei processi globali stimati, che può celare delle rilevanti problematiche. Queste valutazioni preliminari, invece, permettono di fare dei confronti tra metodi che condividono modulazione o struttura, stabilendo conseguentemente la tipologia di algoritmo con le prestazioni migliori. Dopo aver asserito la tipologia di modello potenzialmente più affidabile, si può procedere con l'applicazione su dati reali, per valutare realisticamente il loro comportamento.

Come introdotto, i dati sintetici permettono di analizzare in maniera completa le prestazioni di un modello, dunque è necessario studiare dei metodi affinché le peculiarità emergano. A tale scopo si è scelto di optare per due tipi di dati sintetici:

- Dati Sintetici Completi
- Dati Sintetici Parziali

Dati Sintetici Completi

Con il termine **Dati Sintetici Completi** si intendono tutte quelle simulazioni la cui provenienza è determinata esattamente dalla conoscenza di **ogni** parametro usato nel modello. Tradotto nel caso attuale significa che sono stati predeterminati dei valori di $\boldsymbol{u}, \boldsymbol{\tau}^2, \boldsymbol{\beta}, \sigma_u^2, \sigma_b^2, \alpha, \psi^2, W^T, \boldsymbol{\theta}$, e successivamente si è applicata una procedura di simulazione del processo genesi e dei dati, conforme alle assunzioni del modello.

L'utilità di tale scenario è chiara, avendo note le quantità esatte del processo è possibile determinare con esattezza le prestazioni del modello, confrontando le stime predette con i parametri veri. Nel caso in analisi, tuttavia, non è sufficiente questa casistica per apprezzare tutte le caratteristiche del modello, visto l'interesse nel stimare sia il proceso che la modulazione temporale. Come anticipato nella sezione dedita alla descrizione del modello Ornestein-Uhlenbeck, la modellizzazione del processo genesi nasce per descrivere fenomeni altamente irregolari, e difficilmente si riesce a trovare una configurazione di parametri in grado di offrire delle simulazioni di processi con forme simili a quelle reali dei fischi-firma, che invece risultano molto regolari. Per questa ragione l'utilizzo di dati sintetici completi verrà limitato alle prime fasi di analisi, mentre progressivamente si useranno altre soluzioni che possono più fedelmente riprodurre i processi reali dei fischi-firma. Colloquialmente, e si capirà meglio dalle immagini, i dati sintetici parziali servono a determinare quanto fedelmente il modello riesce a carpire le caratteristiche del processo; mentre, per determinare la bontà della rimodulazione temporale, è più utile considerare i dati simulati di tipo parziale.

Dati Sintetici Parziali

Con **Dati Sintetici Parziali** si intende quell'insieme di processi simulati in cui solamente alcuni parametri richiesti dal modello sono noti. Questo, nel caso in esame, si traduce nella conoscenza di: $\boldsymbol{u}, \boldsymbol{\tau}^2, \sigma_u^2, \sigma_b^2, W^T, \boldsymbol{\theta}$. La motivazione di questa conoscenza parziale è semplice, si possono scegliere delle curve a piacere senza interessarsi dei coefficienti che definiscono il processo di Ornsetein-Uhlenbeck. Dunque si è in grado di ottenere delle curve simili ai fischi-firma reali, che sono una valida soluzione per verificare l'efficacia degli algoritmi e guidare la scelta del migliore. Inizialmente può sembrare un eccesso di zelo quello di accertarsi del non utilizzo dei dati reali nella scelta del modello, ma, come mostrato dai risultati, tutti i modelli si comportano bene, rispetto alla semplice metrica di valutazione di "somiglianza" tra dati reali e stime del processo. Quindi, per garantire l'affidabilità del metodo, è necessario avere dei contesti di valutazione più severi.

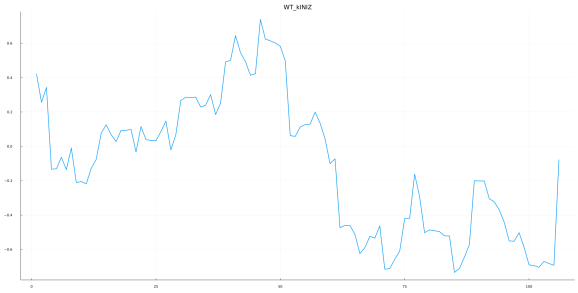
Come già anticipato, l'elevato numero di modelli trattati richiede un attento processo di selezione per orientare l'analisi dei risultati, ecco un breve riassunto su come è impostato. In primo luogo vengono presentate le misure di stima dei parametri del modello, con eventuali giustificazioni laddove non fossero già state presentate nella sezione di Principi Teorici. Si prosegue con una prima fase di selezione, in cui si sceglie la migliore struttura algoritmica tra, *Random-Scan Sampling* contro *Systematic-Scan Samlpling*. Il successivo confronto prestazionale tra modulazioni "regolari" e "definite a tratti", guida invece la selezione del tipo di modulazione che continuerà ad essere investigato. A seguire viene proposta una giustificazione empirica dei riscalamenti temporali, confrontando risultati di modelli diversi. Una volta scelti i modelli potenzialmente più affidabili, vengono verificate le effettive coerenze a livello intuitivo tra modelli, evidenziando le relazioni gerarchiche tra essi. Infine vengono mostrati i risultati completi dei modelli scelti quando applicati sui dati sintetici, per avere una panoramica più completa delle loro capacità sui dati reali.

Le Misura di Stima

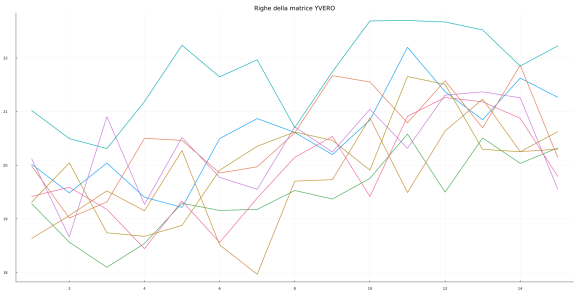
I parametri valutati si dividono in due categorie:

- quantità scalari: $\beta_1, \beta_2, \sigma_u^2, \sigma_b^2, \alpha, \psi^{2,k}$
- quantità vettoriali: $\boldsymbol{u}, \boldsymbol{\tau}^2, W^T$

La distinzione nasce come necessità per sintetizzare l'analisi, avendo comunque una visione complessiva sui parametri, senza omettere troppi dettagli. Si potrebbe argomentare, infatti, che ogni componente delle quantità vettoriali può essere studiata separatamente, diventando dunque una quantità scalare, ma questa



(a) Il processo W_k^T di partenza



(b) I Dati

Figura 6.1: I dati di partenza ed il processo iniziale

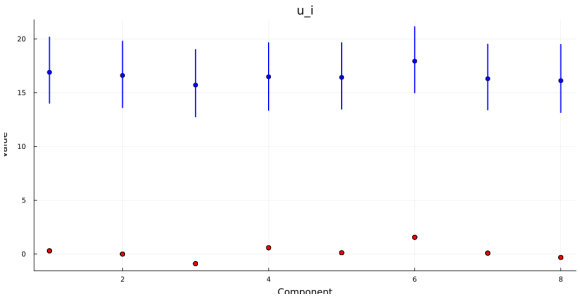
impostazione risulterebbe troppo onerosa dal punto di vista dell'analisi e potenzialmente più confusionaria complessivamente. Per le quantità scalari le misure di bontà della stima sono i *Trace Plot*, i.e. il plot della catena, e i *Correlogrammi*; mentre, per le quantità vettoriali, le *stime intervallari di credibilità* ed, eventualmente, *l'errore quadratico medio*. L'intervallo di credibilità è definito come: Un intervallo di valori che ha probabilità (1 - α) di contenere il parametro [51]. L'errore quadratico medio (MSE), invece, misura mediamente il discostamento al quadrato tra il valore stimato ed il valore vero. Questa metrica non è necessariamente indicativa e infatti non verrà usata in tutti i casi in esame, siccome non è detto che una stima di un vettore sia peggio di un'altra basandosi sul MSE. Questo perché il MSE, pur essendo una misura globale dell'errore, non tiene conto della direzione degli errori nelle diverse componenti del vettore stimato. Due stimatori potrebbero avere lo stesso MSE ma comportamenti molto diversi: ad esempio, uno potrebbe produrre errori piccoli e uniformi su tutte le componenti, mentre un altro potrebbe generare errori grandi in alcune componenti e piccoli in altre. Inoltre, il MSE penalizza maggiormente errori grandi a causa della sua natura quadratica, il che potrebbe non essere sempre desiderabile, ma dipendente dal problema. Per quanto riguarda, invece, il vettore dei coefficienti di modulazione, $\boldsymbol{\theta}$, esso viene trattato in maniera mista, con delle componenti estrapolate singolarmente e altre analizzate in modo vettoriale. I parametri tra le varie esecuzioni degli algoritmi rimangono invariati, perciò sono sempre presentati i risultati riferendosi ad esecuzioni con: 10'000 iterazioni dell'algoritmo MCMC a cui viene applicato un burnin pari a 5500 e un thin pari a 3, risultando in catene lunghe 1500 elementi. Malgrado gli algoritmo MCMC possano anche essere soggetti ad analisi del tempo di esecuzione e del costo computazionale, non è stato possibile garantire le stesse condizioni di lavoro per le diverse esecuzioni del codice, perciò tali analisi sono lasciate per studio futuri.

6.2 Random- vs Systematic- Scan Sampling

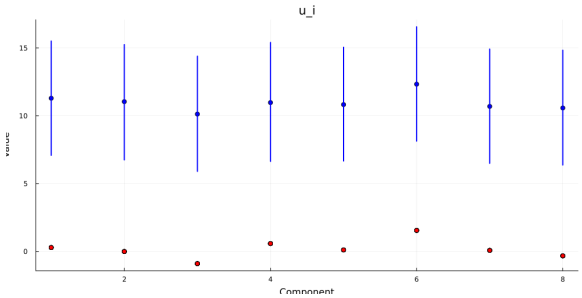
In questa sezione sono mostrati i risultati del modello con prima modulazione temporale e strutturati con diverse tipologie di algoritmi. Nello specifico i casi di interesse sono quattro:

- Systematic-Scan Sampling con ordine degli step [1,2,3,4,5,6,7];
- Random-Scan Sampling con ordine degli step iniziale [1,2,3,4,5,6,7];
- Systematic-Scan Sampling con ordine degli step [4,2,3,5,6,7,1];
- Random-Scan Sampling con ordine degli step iniziale [4,2,3,5,6,7,1].

Gli algoritmi MCMC sono stati in tutti e quattro i casi inizializzati con lo stesso esatto tipo di dati sintetici, dati iniziali e parametri iniziali. L'unica differenza risiede nella struttura algoritmica. Ora, per evitare inutili ripetizioni di grafici, seguono i principali risultati per la determinazione della configurazione migliore.

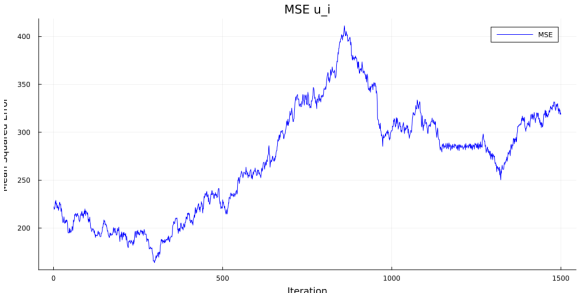


(a) Stime intervallari $\boldsymbol{u}$ per il Systematic-Scan con step ordinati

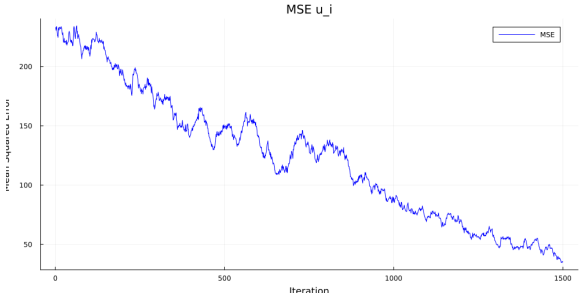


(b) Stime intervallari $\boldsymbol{u}$ per il Random-Scan con step ordinati

Figura 6.2: Confronto tra le stime intervalli $\boldsymbol{u}$ tra Systematic e Random con step ordinati



(a) MSE $\boldsymbol{u}$ per il Systematic-Scan con step ordinati

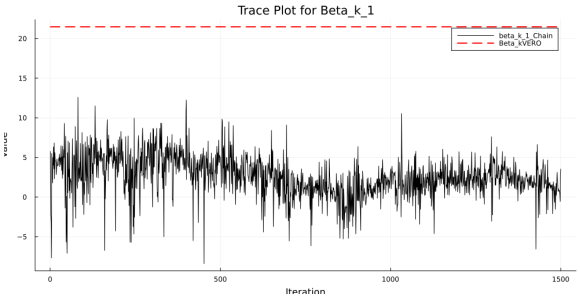


(b) MSE $\boldsymbol{u}$ per il Random-Scan con step ordinati

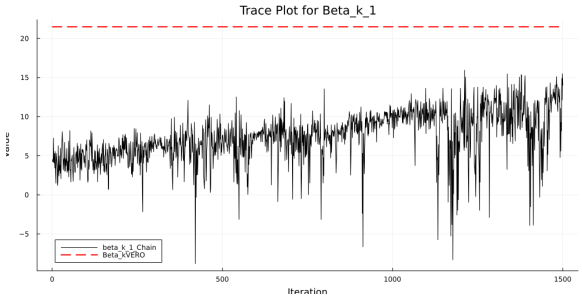
Figura 6.3: Confronto andamenti MSE tra Systematic e Random con step ordinati

Come deducibile dalla figura 6.1, si è scelto di partire con una configurazione molto generale, drasticamente diversa dai dati. Il processo W_k^T di partenza nasce da una trasformazione di rumore bianco, con componente tralsazionale nulla, che permette di ricadere nel caso critico in cui lo step 1, atto alla stima di $\boldsymbol{u}$, si prende carico di buona parte della componente traslazione che dovrebbe essere di natura ricondotta a β_1, come introdotto dettagliatamente nel capitolo dell'algoritmo MCMC. Questa inizializzazione così lontana dal valore vero del processo, unita ad una struttura intrinsicamente sfavorevole dell'algoritmo, conduce ad un errore significativo della stima di alcune quantità, che ragionevolmente compromettono la stima finale del processo W_k^T . Questo fenomeno tuttavia mostra diverse implicazioni per i due tipi di strutture confrontate, a tal proposito è sufficiente notare come l'andamento del MSE in figura 6.3 mostri due comportamente drasticamente diversi, dove il modello sistematico sembra essere arrivato a convergenza, mentre l'altro coglie che la configurazione di convergenza è diversa. Il processo di spostamento tra una configurazione errata all'altra si può apprezzare anche dal trace plot di β_1 in figura 6.4. Mentre i grafici delle figure 6.2 6.5 mostrano come le stime siano drasticamente errate, mostrando anche qua differenze significative malgrado l'errore, siccome il modello Random-Scan mostra intervalli più larghi, e unendo questa informazione con il MSE, si capisce che sta cercando di arrivare al punto di convergenza effettivo. Perciò, da questa prima analisi, risulta che la struttura Random-Scan sia la scelta più adeguata.

Per completare la panoramica sulla disposizione dei passi dell'algoritmo, si procede con una configurazione più favorevole dei campionamenti, iniziando con la stima di $\boldsymbol{\beta}$ e concludendo con $\boldsymbol{u}$. Le figure 6.6 6.7 6.8 6.9 mostrano un comportamento quasi identico tra le due strutture; non sembra esserci prevalenza dei risultati in nessuna delle stime critiche analizzate e i risultati sono generalmente accettabili, sapendo che i dati sintetici scelti sono volutamente molto stressanti e ci si aspetta di trovare alcune inesattezze di stima. Malgrado queste inesattezze evidenti, come nella stima di β_1, il risultato sull'osservazione in figura 6.10 è ottimo per entrambi i modelli. Per completezza è bene specificare come, in realtà, i Trace-Plot di β_1 sono sistematicamente insoddisfacienti, se non nella stima direttamente, spesso nella loro "forma". Questo risulta essere un tema ricorrente oltre che la principale debolezza evidente degli algoritmi

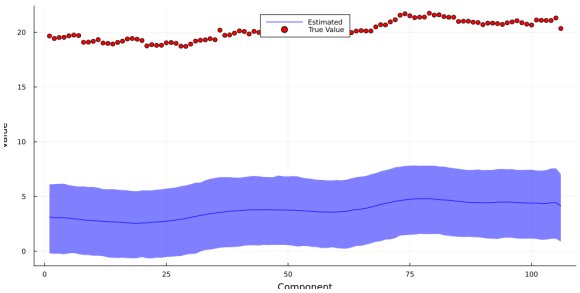


(a) Trace-Plot β_1 per il Systematic-Scan con step ordinati

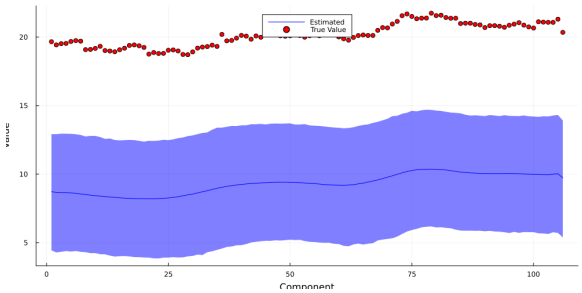


(b) Trace-Plot β_1 per il Random-Scan con step ordinati

Figura 6.4: Confronto Trace-Plot β_1 tra Systematic e Random con step ordinati



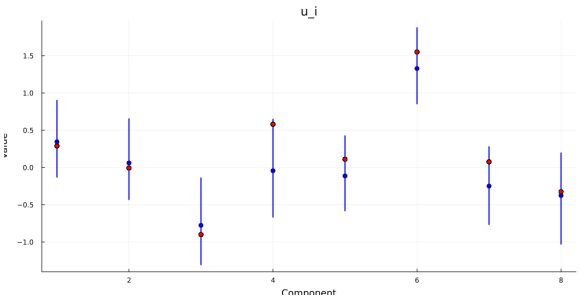
(a) Stima del processo W_k^T per il Systematic-Scan con step ordinati



(b) Stima del processo W_k^T per il Random-Scan con step ordinati

Figura 6.5: Confronto Stima del processo W_k^T tra Systematic e Random con step ordinati

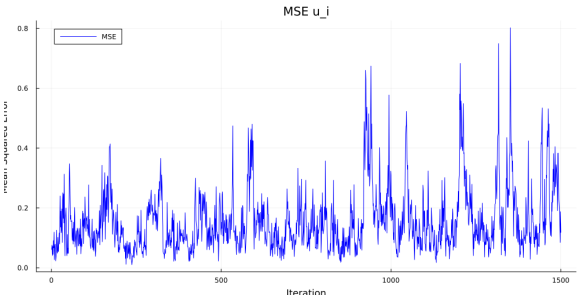
presentati. In conclusione a questa prima analisi, si sceglie di proseguire con la struttura **Random-Scan Sampling**, malgrado i risultati pressoché identici tra le due strutture con inizializzazione non-ordinata. Il fatto che il Systemic-Scan Sampling abbia dato dei risultati così diversi, in termini di prestazioni e convergenza, lascia aperta l'ipotesi che vi sia una configurazione ottimale ben specifica di parametri da campionare in modo sequenziale, la cui indagine esula dalle finalità di questa tesi, in cui, per mantenere rigorosità e logica, si sceglie la configurazione casuale dei passi, con inizializzazione non-ordinata.



(a) Stime intervallari $\boldsymbol{u}$ per il Systematic-Scan con step non-ordinati

(b) Stime intervallari $\boldsymbol{u}$ per il Random-Scan con step non-ordinati

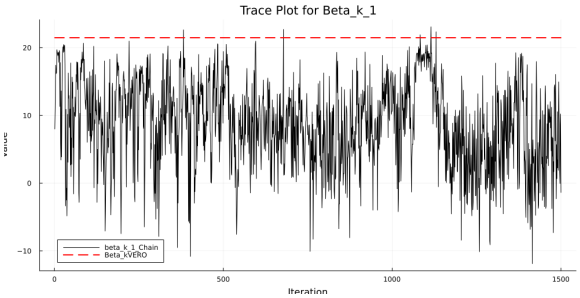
Figura 6.6: Confronto tra le stime intervalli $\boldsymbol{u}$ tra Systematic e Random con step non-ordinati



(a) MSE $\boldsymbol{u}$ per il Systematic-Scan con step non-ordinati

(b) MSE $\boldsymbol{u}$ per il Random-Scan con step non-ordinati

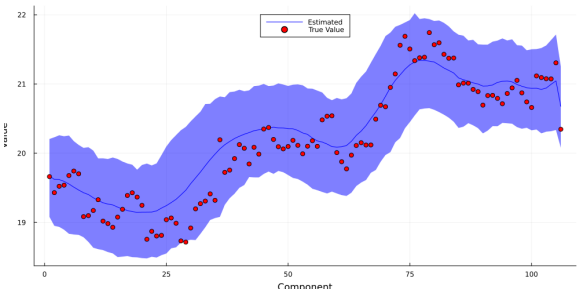
Figura 6.7: Confronto andamento MSE tra Systematic e Random con step non-ordinati



(a) Trace-Plot β_1 per il Systematic-Scan con step non-ordinati

(b) Trace-Plot β_1 per il Random-Scan con step non-ordinati

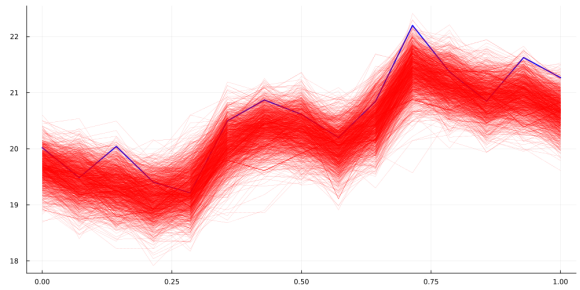
Figura 6.8: Confronto Trace-Plot β_1 tra Systematic e Random con step non-ordinati



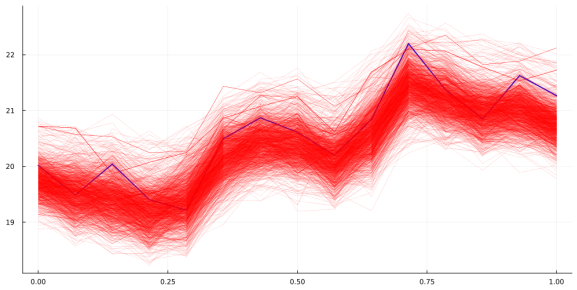
(a) Stima del processo W_k^T per il Systematic-Scan con step non-ordinati

(b) Stima del processo W_k^T per il Random-Scan con step non-ordinati

Figura 6.9: Confronto Stima del processo W_k^T tra Systematic e Random con step non-ordinati

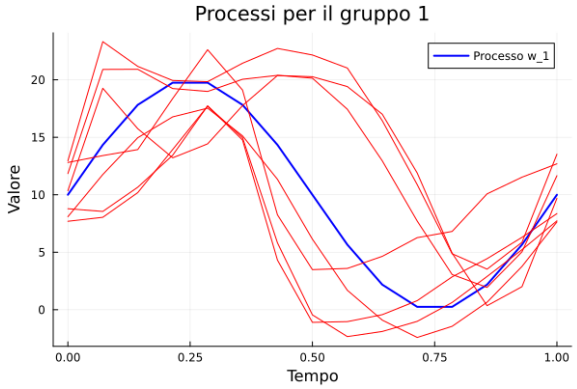


(a) Stima di un'istanza per il Systematic-Scan con step non-ordinati

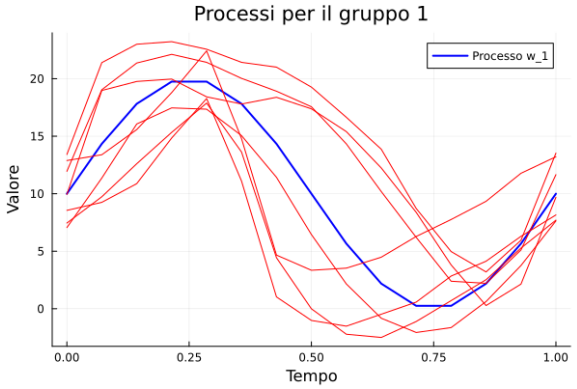


(b) Stima di un'istanza per il Random-Scan con step non-ordinati

Figura 6.10: Confronto stima di un'istanza con step non-ordinati



(a) Processi simulati per il gruppo di dati sintetici parziali 1, con modulazione 2 regolare



(b) Processi simulati per il gruppo di dati sintetici parziali 1, con modulazione 2 a tratti

Figura 6.11: Confronto di processi simulati per il gruppo di dati sintetici parziali 1, con modulazione 2

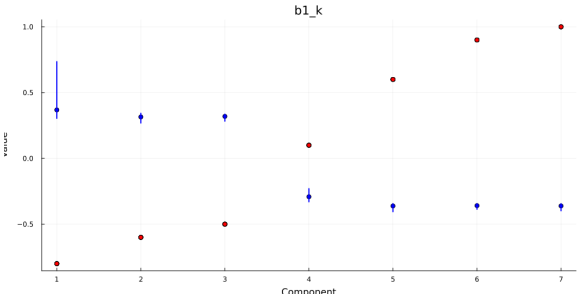
6.3 Modulazione: Regolare vs Definita a Tratti

In questa sezione si analizzano i risultati, su dati sintetici parziali, dei modelli aventi seconda e terza modulazione temporale. Nella stessa famiglia di modulazione temporale si è interessati a capire quale tipo sia più adatto ad un'indagine statistica come quella effettuata con gli algoritmi MCMC implementati, esplicitamente si vuole determinare quale riscalamento temporale sia più calzante tra quelli regolari ed i definiti a tratti. Si inizia con il confronto all'interno della seconda famiglia di riscalamenti:

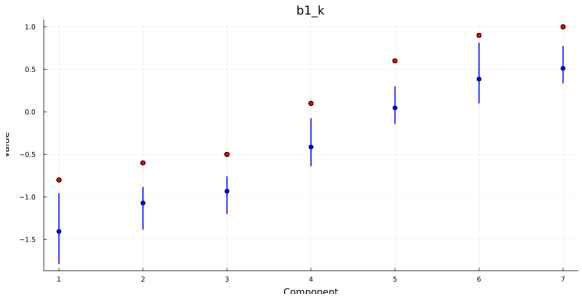
$$g(t) = t^{c_1} + (t^{c_2} - t^{c_1})\frac{1}{1+e^{-\gamma(t-\delta)}} \text{ con } t \in [0,1], c_1, c_2 \in (0,+\infty), \delta \in [0,1], \gamma \in (0,+\infty]$$

Anche in questo caso il confronto avviene esclusivamente sui parametri più di interesse per il genere di indagine che si sta facendo. La scelta dei dati sintetici parziali è figlia dell'interesse che si sta cercando di dare alla modulazione temporale in questa fase di selezione.

In via preliminare si noti come i processi di partenza sono leggermente diversi, come visibile in figura 6.11, a causa di diverse ipotesi di modulazione. Malgrado questa differenza, è comunque possibile avanzare delle conclusioni sulle prestazioni generali dei due algoritmi. Innanzitutto si può notare come le due tipologie di seconda modulazione mostrano enormi differenze in termini di qualità di stima, specie nella stima del vettore dei coefficienti di prima modulazione b_1, in figura 6.12 dove la modulazione a tratti riesce a rappresentare correttamente il vettore almeno in termini relativi tra varie osservazioni. In altra parole, riconosce correttamente una modulazione più o meno aggressiva di un'altra, mentre il caso regolare sembra addirittura manifestare relazioni opposte. Lo scenario regolare migliora nella valutazione del vettore b_2 in figura 6.13, ma allo stesso modo migliora anche per lo scenario a tratti, che in questo frangente non si limita a cogliere la modulazione relativa, infatti, riesce a stimare degli

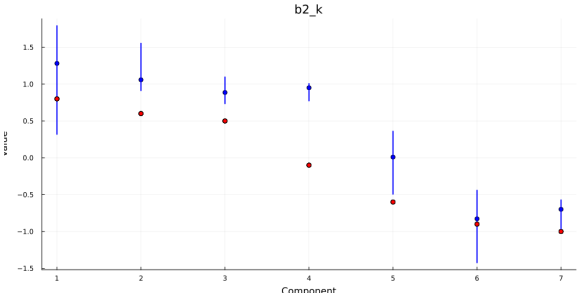


(a) Stime del vettore b_1 per la modulazione 2 regolare

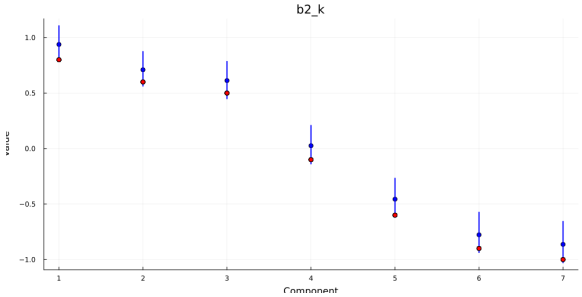


(b) Stime del vettore b_1 per la modulazione 2 a tratti

Figura 6.12: Confronto tra stime del vettore b_1 per la modulazione 2



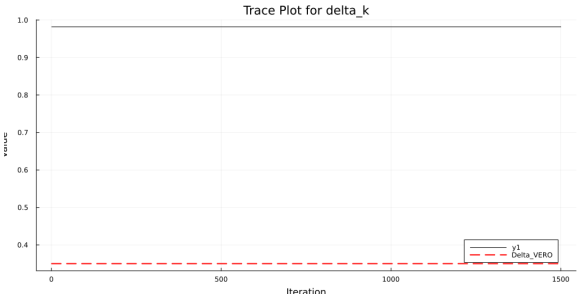
(a) Stime del vettore b_2 per la modulazione 2 regolare



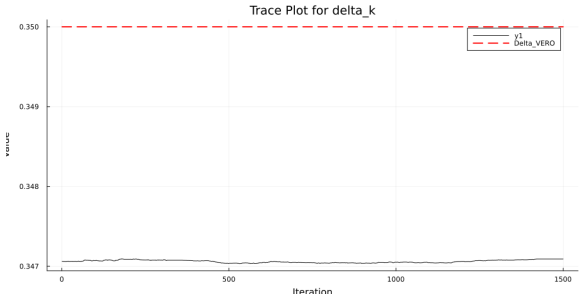
(b) Stime del vettore b_2 per la modulazione 2 a tratti

Figura 6.13: Confronto tra stime del vettore b_2 per la modulazione 2

intervalli di credibilità che contengono il valore vero. Un altro ruolo critico in questa trattazione viene ricoperto dal parametro δ e, nel caso della modulazione 1, anche da γ. Malgrado la loro importanza teorica, le figure: 6.14 6.15 mostrano che le stime vengono completamente sbagliate nel caso regolare. Ad una prima osservazione anche il caso a tratti sembra fornire pessimi risultati per il parametro δ, ma è solo un'impressione. Chiaramente il valore vero non viene mai toccato dalla traiettoria del trace-plot, ma essa oscilla sul valore 0.347 quando il valore vero è 0.35, ovvero si tratta di un'ottima stima. Infine è interessante notare come, malgrado il modello con modulazione regolare abbia delle stime deludenti rispetto ai coefficienti concettualmente cruciali, entrambi riescano con successo a stimare correttamente la realizzazione in figura 6.16. Questo risultato potrebbe indurre a pensare che quindi entrambe le modulazioni vadano bene, anche se le stime non sono proprio ideali nel caso regolare, ma questo sarebbe un grosso errore di valutazione. Si presti attenzione alla figura 6.17 per convincersi dell'importanza delle stime dei parametri. Tale figura rappresenta l'anello fondamentale che collega le stime dei parametri alle stime delle realizzazioni, mostrando come il modello vede i processo genesi. E' cruciale che il processo genesi venga ricostruito correttamente, non tanto per la loro rappresentazione nelle istanze, ma per la loro classificazione nel momento in cui si volesse applicare questo algoritmo di rimodulazione temporale a scenari più generali, come quelli del paper [23], dove una corretta rappresentazione del processo genesi è necessaria per aumentare la probabilità di assegnazione di un'osservazione all'individuo corretto. Siccome il problema di clusterizzazione rappresenta lo scopo finale di quest'analisi, per la modulazione 2, si proseguirà d'ora in avanti, con lo schema di definizione a tratti.



(a) Trace-Plot del parametro δ per la modulazione 2 regolare



(b) Trace-Plot del parametro δ per la modulazione 2 a tratti

Figura 6.14: Confronto Trace-Plot del parametro δ per la modulazione 2

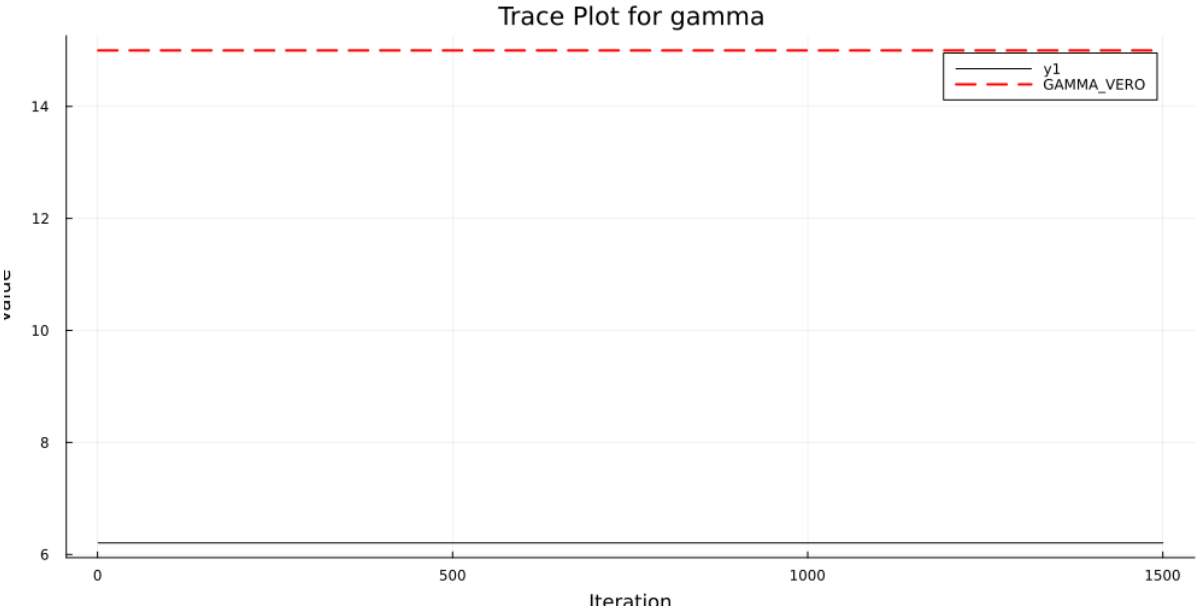
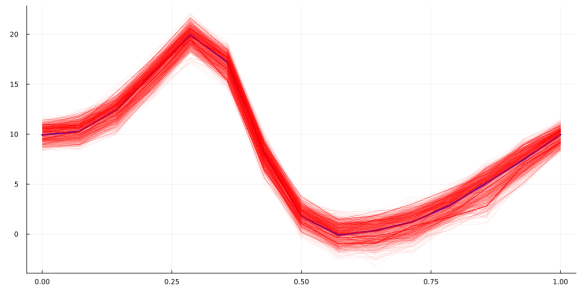


Figura 6.15: Trace-Plot γ per la modulazione 2 regolare

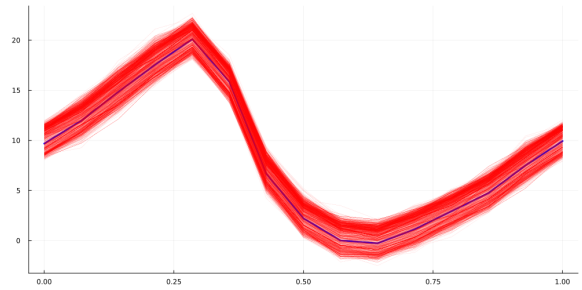
Ora che la scelta per la seconda famiglia di riscalamenti è stata effettuata, si procede con un'analisi analoga per i riscalamenti del tipo:

$$g(t) = t^{c_1} + \frac{t^{c_2} - t^{c_1}}{1+e^{-\gamma(t-\delta_1)}} + \frac{t^{c_1} - t^{c_2}}{1+e^{-\gamma(t-\delta_2)}} \text{ con } t \in [0,1], c_1, c_2 \in (0,+\infty), 0 \leq \delta_1 \leq \delta_2 \leq 1, \gamma \in (0,+\infty]$$

Anche in questo caso i risultati della modulazione a tratti sembrano condurre a modelli più affidabili. Le stime dei coefficienti b_i nelle figure: 6.19 6.20 sono chiari indicatori di come il modello regolare non sia in grado di stimare correttamente i coefficienti di modulazione. Lo stesso vale per i coefficienti δ_1, δ_2 e γ nelle figure 6.21 6.22 6.23, dove in tutti questi casi si ottengono valori gravemente insoddisfacienti da parte del modello regolare. Sarebbero sufficienti questi grafici per concludere la scelta, ma anche in questo caso risulta interessante proseguire e analizzare il comportamento nella ricostruzione del processo genesi. Senza stupore si nota come il confronto tra i due processi ricostruiti in figura 6.25 vede prevalere, in termini di precisione, il modello con riscalamento a tratti. Tuttavia, così come avvenuto nella modulazione 2, l'enorme discostamento tra il processo reale e quello ricostruito, non rispecchia la fase di stima delle realizzazioni, che mantiene risultati ottimi malgrado le stime compromettenti dei parametri, come visibile in figura 6.24. In conclusione si ottiene ulteriore conferma delle potenzialità dell'algoritmo nella stima delle realizzazioni, anche laddove i parametri non sono vicini a quelli veri. Ma, soprattutto, si determina che il modello con riscalamento a tratti è sensibilmente più affidabile della sua controparte regolare,

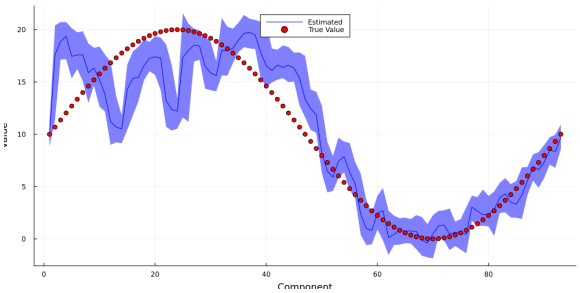


(a) Stima di una realizzazione della modulazione 2 regolare

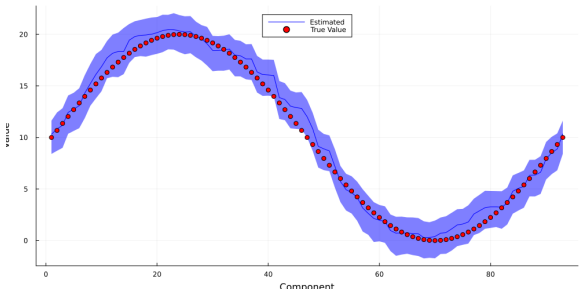


(b) Stima di una realizzazione della modulazione 2 a tratti

Figura 6.16: Confronto tra stime di realizzazioni per diverse modulazioni 2



(a) Processo stimato W_k^T nel caso di modulazione 2 regolare



(b) Processo stimato W_k^T nel caso di modulazione 2 a tratti

Figura 6.17: Confronto tra processo stimato W_k^T nel caso di modulazione 2

questo implica che per le successive analisi sulla modulazione 3 è preferibile considerare unicamente il caso a tratti.

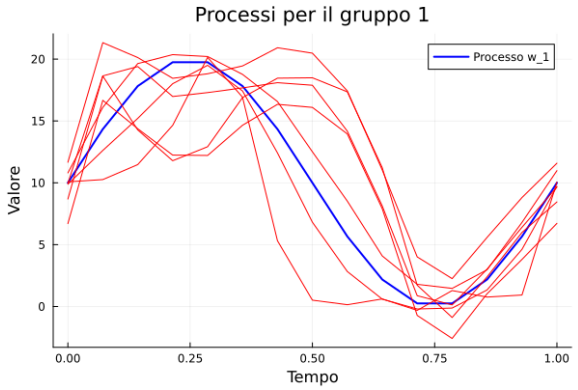
6.4 Giustificazione Empirica delle Modulazioni

I riscalamenti temporali, finora, sono stati giustificati esclusivamente dalle osservazioni degli esperti sul tema dei fischi-firma. Tuttavia, un'analisi interessante potrebbe consistere nel verificare se esistano effettive differenze prestazionali tra le diverse modulazioni, esaminandone il comportamento su stessi dati sintetici; in questo modo, la loro necessità potrebbe essere empiricamente giustificata attraverso l'algoritmo MCMC. A tal proposito viene proposto un confronto tra la prima modulazione, la più semplice, e la terza modulazione a tratti, la più complessa, entrambe con logica Random-Scan Sampling. Anche in questo caso, ragionevolmente, si valuta la bontà della modulazione, e quindi si applicano gli algoritmi su dati sintetici parziali. Osservando le figure 6.26 6.27 6.28 le conclusioni sono chiare: a parità di realizzazioni, la terza modulazione fornisce una rappresentazione del processo genesi molto più accurata. Nonostante la rappresentazione compromettente offerta dalla prima modulazione, tuttavia, le sue rappresentazioni dei dati sono fedeli, ma comunque meno aderenti rispetto a quanto offerto dal terzo riscalamento. Le immagini supportano l'ipotesi della necessità di modulazioni successive alla prima.

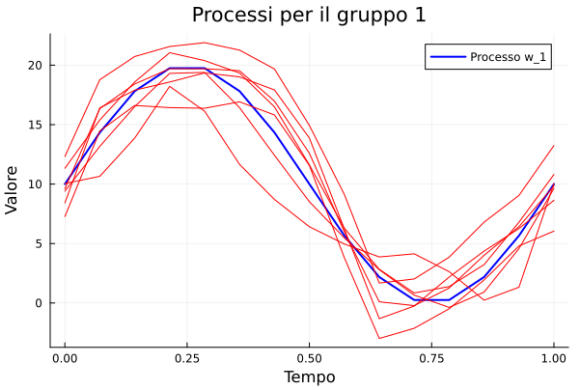
6.5 Coerenza tra Riscalamenti Diversi

I tipi di riscalamenti sono diversi, ma possono essere visti come casi particolari l'uno dell'altro, infatti, il riscalamento temporale:

$$g(t) = t^c \text{ con } t \in [0,1] \text{ e } c \in (0, +\infty)$$

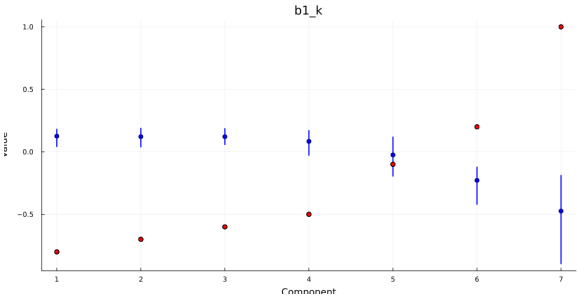


(a) Processi simulati per il gruppo di dati sintetici parziali 1, con modulazione 3 regolare

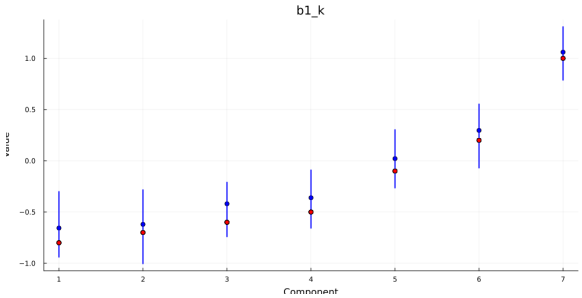


(b) Processi simulati per il gruppo di dati sintetici parziali 1, con modulazione 3 a tratti

Figura 6.18: Confronto di processi simulati per il gruppo di dati sintetici parziali 1, con modulazione 3



(a) Stime del vettore b_1 per la modulazione 3 regolare



(b) Stime del vettore b_1 per la modulazione 3 a tratti

Figura 6.19: Confronto tra stime del vettore b_1 per la modulazione 3

può essere visto come caso particolare di:

$$g(t) = t^{c_1} + (t^{c_2} - t^{c_1})\frac{1}{1+e^{-\gamma(t-\delta)}} \text{ con } t \in [0,1], c_1, c_2 \in (0,+\infty), \delta \in [0,1], \gamma \in (0,+\infty]$$

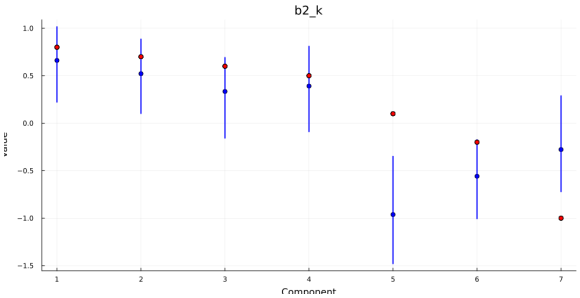
con varie configurazioni, tra cui $\delta \approx 0$ o $\delta \approx 1$. Il quale può essere visto a sua volta come caso particolare di:

$$g(t) = t^{c_1} + \frac{t^{c_2}-t^{c_1}}{1+e^{-\gamma(t-\delta)}} + \frac{t^{c_1}-t^{c_2}}{1+e^{-\gamma(t-\delta_2)}} \text{ con } t \in [0,1], c_1, c_2 \in (0,+\infty), 0 \leq \delta_1 \leq \delta_2 \leq 1, \gamma \in (0,+\infty]$$

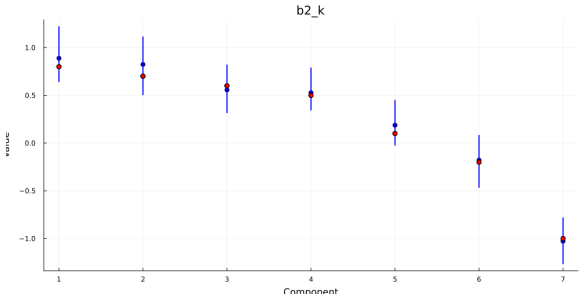
ponendo, ad esempio, $\delta_2 \approx 1$. Per verificare la buona implementazione dei metodi e la loro coerenza, dunque, un'analisi interessante potrebbe essere quella di applicare un metodo più generale a dati simulati con modelli che possono essere visti come casi particolari di esso. Se le implementazioni sono corrette e la metodologia ben posta, si dovrebbero ottenere risultati simili, con la naturale convergenza del modello generale al modello particolare. A tale scopo vengono analizzati due scenari, il primo in cui si applica la seconda modulazione a dati sintetici parziali generati seguendo la prima modulazione; a cui segue lo stesso scenario con protagonisti, rispettivamente, la seconda e la terza modulazione.

I risultati nelle figure 6.29 6.30 6.31 sono esplicativi, il comportamento del modello generale converge esattamente dove era previsto. Il contributo della prima zona di modulazione temporale viene annichilito e il vettore dei coefficienti di modulazione rimanente combacia con quello stimato dal primo riscalamento.

Per il secondo confronto si ottengono risultati altrattanto incoraggianti. Il valore di δ_2 viene stimato approssimativamente pari a 1, rappresentato in figura 6.32 il che implica una sola transizione di modulazione, che avviene in corrispondenza di δ_1, anch'esso stimato coerentemente al valore vero noto dal secondo riscalamento, visibile in 6.33. I vettori per i coefficienti di modulazione, $\boldsymbol{b}_{1,i}$ e $\boldsymbol{b}_{2,i}$ sono quasi identici tra i due modelli, 6.34 6.35. Conseguentemente i processi genesi sono rappresentati allo stesso modo tra i due riscalamenti, come mostrato dalla figura 6.36.

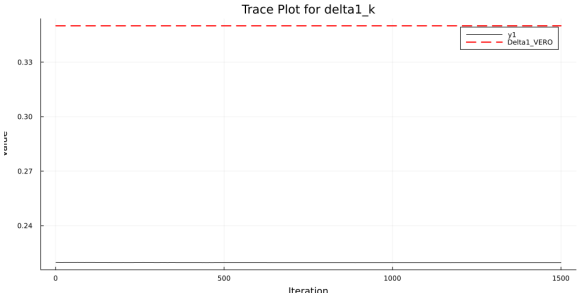


(a) Stime del vettore b_2 per la modulazione 3 regolare

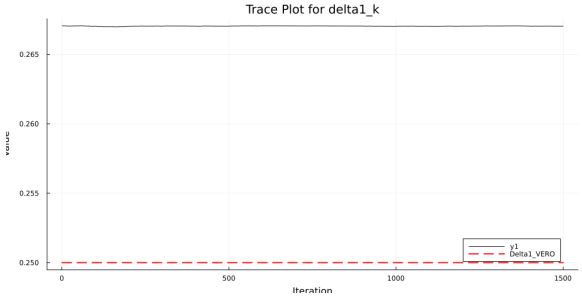


(b) Stime del vettore b_2 per la modulazione 3 a tratti

Figura 6.20: Confronto tra stime del vettore b_2 per la modulazione 3



(a) Trace-Plot del parametro δ_1 per la modulazione 3 regolare



(b) Trace-Plot del parametro δ_1 per la modulazione 3 a tratti

Figura 6.21: Confronto Trace-Plot del parametro δ_1 per la modulazione 3

6.6 Applicazioni Complete

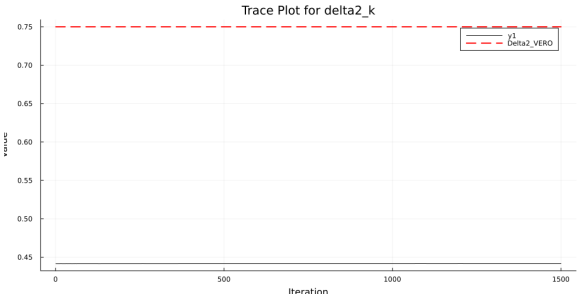
Dopo aver selezionato i metodi e giustificato la loro implementazione, è fondamentale applicare gli algoritmi a nuovi esempi di dati sintetici, valutandone il comportamento in modo globale. Questo include l'analisi delle stime dei parametri finora non considerati. L'obiettivo è formulare ipotesi sul loro comportamento su dati reali e, se necessario, individuare eventuali criticità. A tal proposito sono mostrati e commentati i risultati dei modelli:

- prima modulazione con random-scan su dati sintetici totali e parziali;
- seconda modulazione a tratti con random-scan su dati sintetici totali e parziali;
- terza modulazione a tratti con random-scan su dati sintetici totali e parziali.

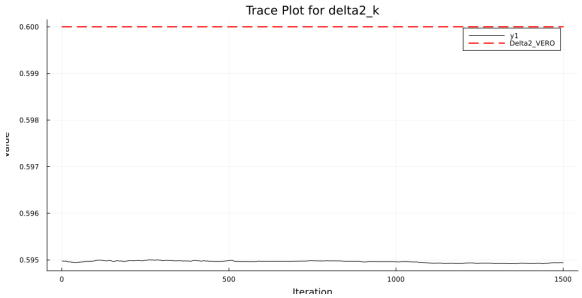
Si è scelto di rappresentare questi risultati a coppie in parallelo: dati sintetici totali, dati sintetici parziali.

Prima Modulazione

Come di consueto il processo iniziale, in figura 6.37, è stato scelto volutamente molto diverso dai dati, 6.38, per effettivamente valutare sotto condizioni stressanti il modello. I due flussi parelleli iniziano con le stime del vettore $\boldsymbol{u}$, che risultano eccellenti per entrambi, figura 6.39, come altrettanto buone sono le stime per $\boldsymbol{\tau}^2$, specialmente per i dati sintetici parziali in figura 6.40. Come anticipato, il trace-plot del coefficiente β_1 è un tema critico, sia in termini di accuratezza che di autocorrelazione 6.41, mentre la stima del coefficiente β_2 risulta buona sotto entrambi gli aspetti 6.42. Anche il trace-plot di σ_u^2 non è ideale 6.43, ma riesce ad aggirarsi attorno al valore vero in maniera discreta, seppur con varianza non stabile, mostrando un buon correlogramma 6.44. Per quanto riguardo il coefficiente $\sigma_b^{2,k}$ i due trace-plot presentano comportamenti diversi, così come i due correlogrammi, figure 6.45 e 6.46. Nello specifico si



(a) Trace-Plot del parametro δ_2 per la modulazione 3 regolare



(b) Trace-Plot del parametro δ_2 per la modulazione 3 a tratti

Figura 6.22: Confronto Trace-Plot del parametro δ_2 per la modulazione 3

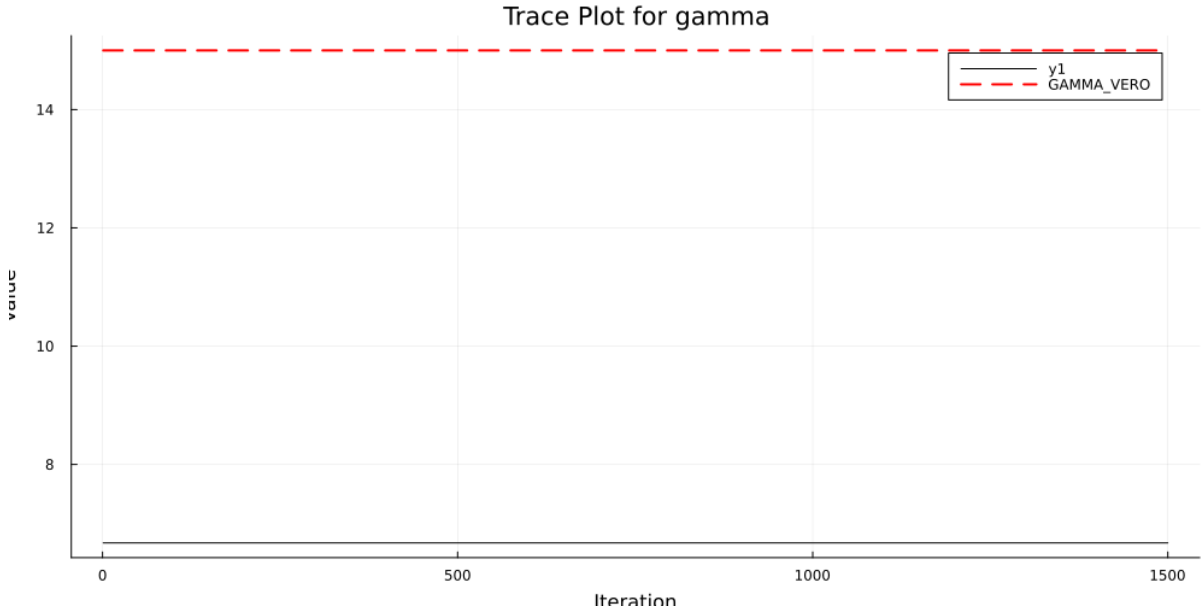
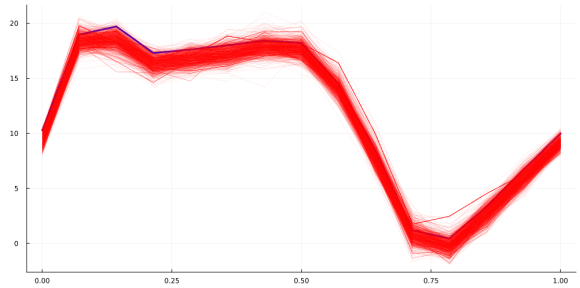
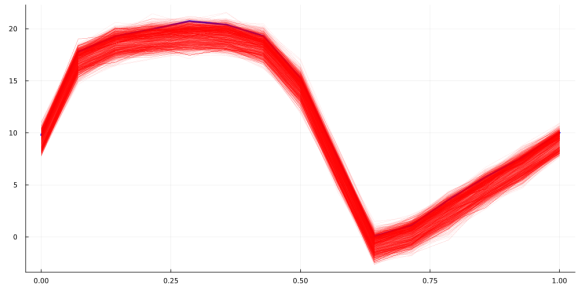


Figura 6.23: Trace-Plot γ per la modulazione 3 regolare

evidenzia un comportamento instabile, ma comunque vicino al valore vero, per il trace-plot per i dati totali con conseguente autocorrelazione non trascurabile per i primi lag; mentre per i dati parziali si hanno sia trace-plot che correlogramma ideali. Le stime dei coefficienti b_i sono ragionevolmente grossolane per i dati sintetici totali, essendo il processo da modellare molto irregolare. Nonostante ciò, tutti gli intervalli di credibilità stimati contengono il valore vero, seppur con intervalli troppo ampi per poter fare inferenza, 6.47.a. D'altro canto, per i dati sintetici parziali, si osserva un'eccellente prestazione in fase di stima intervallare per i parametri b_i, in figura 6.47.b. Infine vengono mostrate le modellizzazioni per i processi genesi, che risultano buone, seppur nel caso a. l'intervallo stimato risulti un po' lasso, 6.48. Vengono anche mostrate due coppie di realizzazioni stimate per ciascun tipo di dato. Queste stime risultano ottime per gli intenti e le difficoltà dell'applicazione, 6.49 e 6.50. Complessivamente si ritengono soddisfacienti questi test e si ha conferma di poter applicare questo riscalamento ai dati reali.

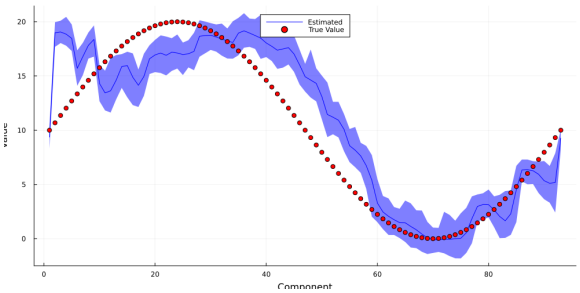


(a) Stima di una realizzazione della modulazione 3 regolare

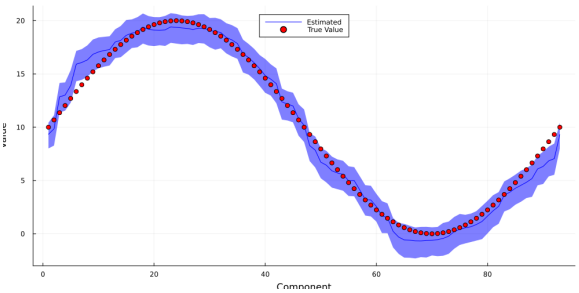


(b) Stima di una realizzazione della modulazione 3 a tratti

Figura 6.24: Confronto tra stime di realizzazioni per diverse modulazioni 3

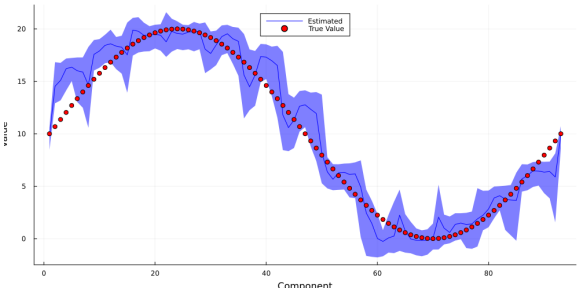


(a) Processo stimato W_k^T nel caso di modulazione 3 regolare

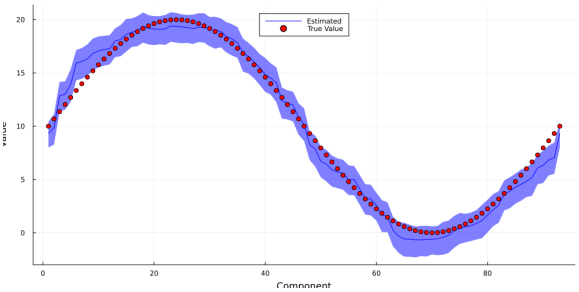


(b) Processo stimato W_k^T nel caso di modulazione 3 a tratti

Figura 6.25: Confronto tra processo stimato W_k^T nel caso di modulazione 3

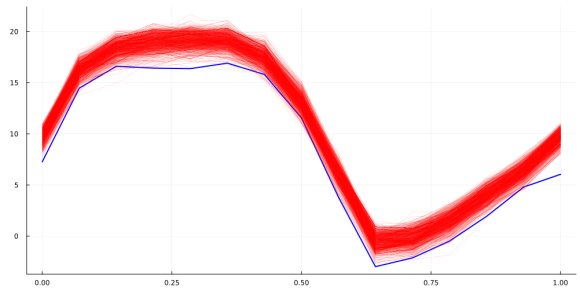


(a) Rappresentazione del processo W_k^T nella prima modulazione

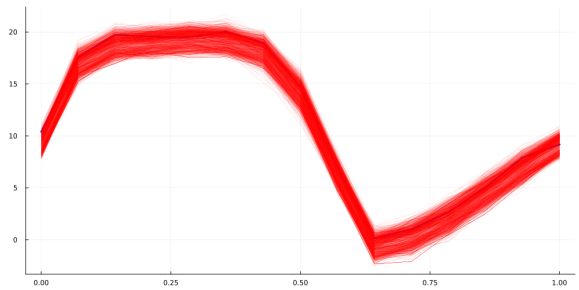


(b) Rappresentazione del processo W_k^T nella terza modulazione

Figura 6.26: Rappresentazione del processo W_k^T prima modulazione contro terza modulazione

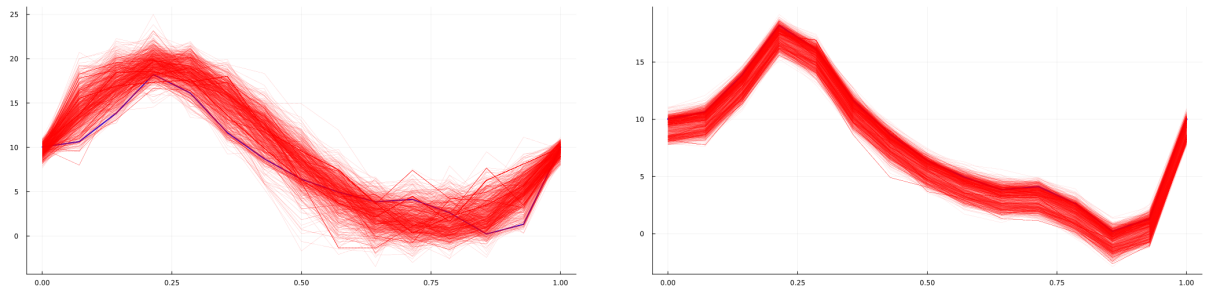


(a) Realizzazione 2 secondo la prima modulazione



(b) Realizzazione 2 secondo la terza modulazione

Figura 6.27: Realizzazione 2 prima modulazione contro terza modulazione



(a) Realizzazione 7 secondo la prima modulazione

(b) Realizzazione 7 secondo la terza modulazione

Figura 6.28: Realizzazione 7 prima modulazione contro terza modulazione

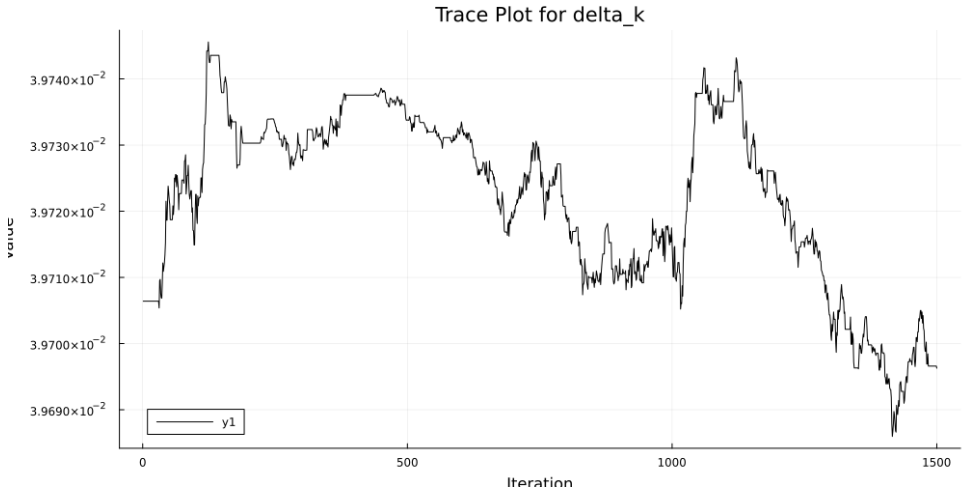
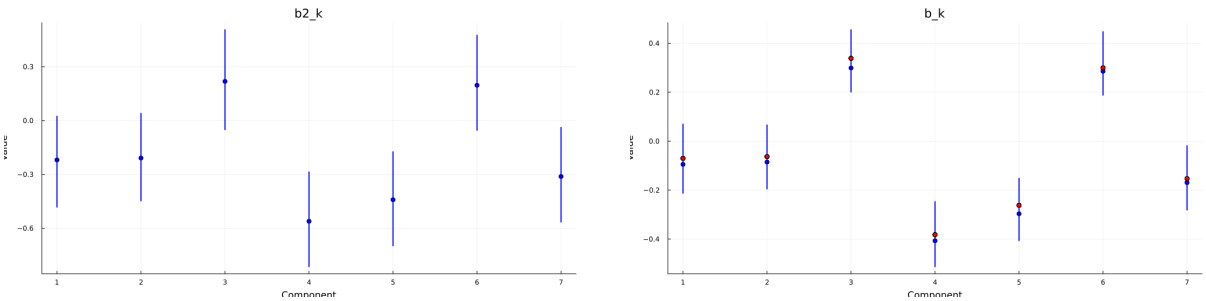


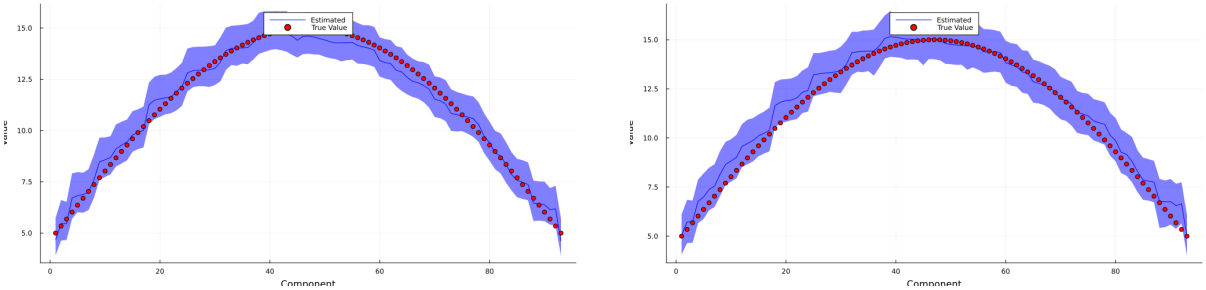
Figura 6.29: parametro δ a cui il modello con la seconda modulazione converge, con i dati sintetici simulati con la prima modulazione



(a) Coefficienti b_i rilevanti nel secondo riscalamento

(b) I valori stimati di b_i stimati dal primo riscalamento

Figura 6.30: Coefficienti di modulazione stimati a confronto tra seconda e prima modulazione



(a) Il processo W_k^T stimato nel secondo riscalamento

(b) Il processo W_k^T stimato dal primo riscalamento

Figura 6.31: Processi genesi a confronto tra seconda e prima modulazione

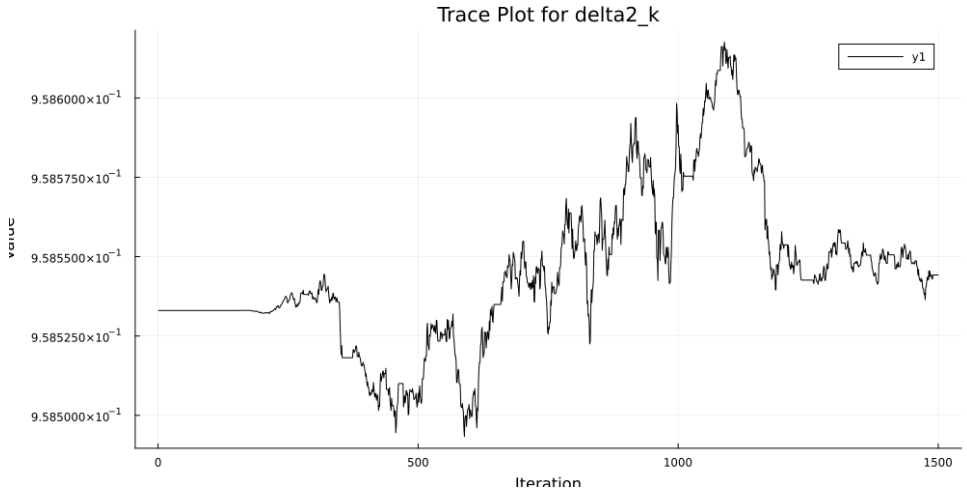
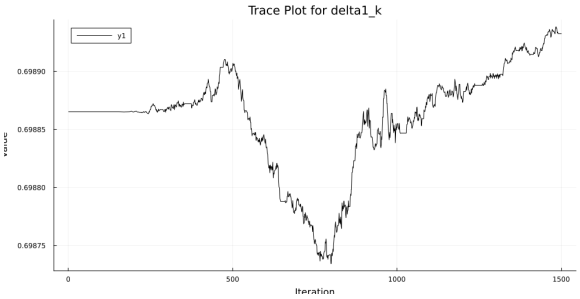
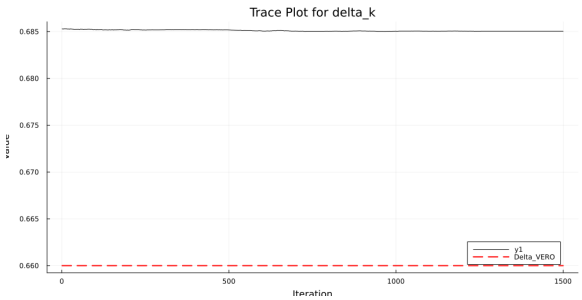


Figura 6.32: parametro δ_2 a cui il modello con la terza modulazione converge, con i dati sintetici simulati con la seconda modulazione

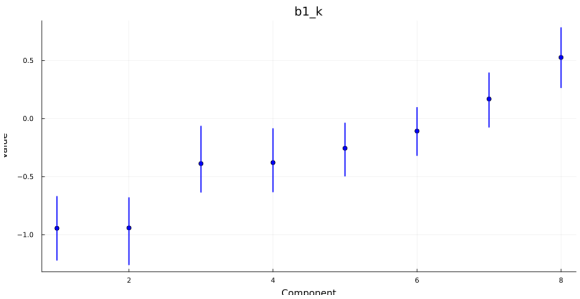


(a) Coefficiente δ rilevante a cui il terzo riscalamento converge

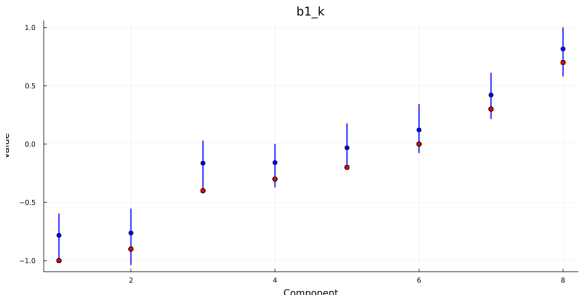


(b) I valori di δ stimati dal primo riscalamento

Figura 6.33: Coefficienti δ stimati a confronto tra terza e seconda modulazione

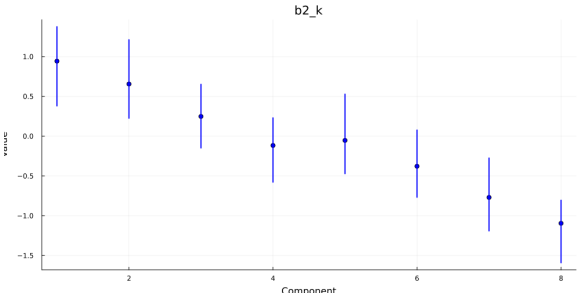


(a) Coefficienti $b_{1,i}$ a cui il terzo riscalamento converge

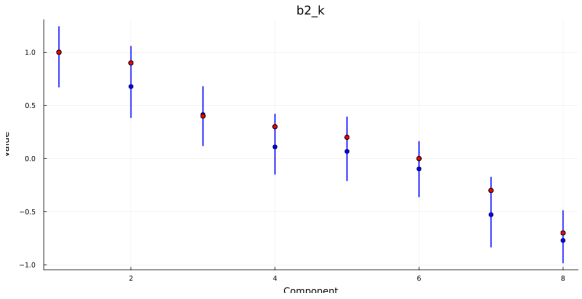


(b) I valori di $b_{1,i}$ stimati dal secondo riscalamento

Figura 6.34: Coefficienti b_1 stimati a confronto tra terza e seconda modulazione

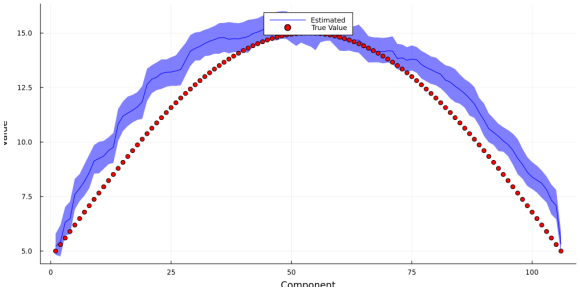


(a) Coefficienti $b_{2,i}$ a cui il terzo riscalamento converge

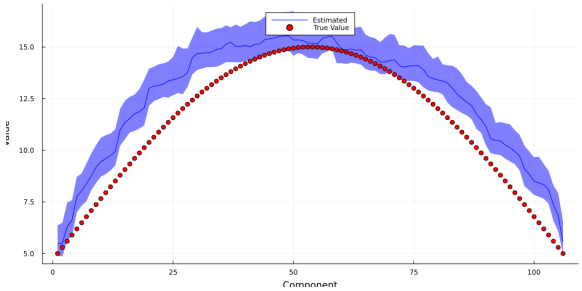


(b) I valori di $b_{2,i}$ stimati dal secondo riscalamento

Figura 6.35: Coefficienti b_2 stimati a confronto tra terza e seconda modulazione



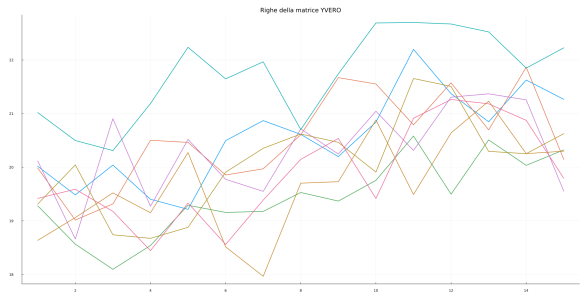
(a) I valori del processo genesi stimati dal terzo riscalamento



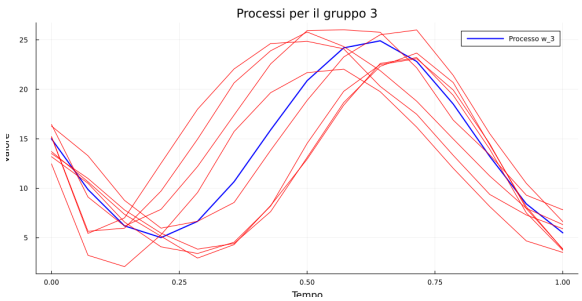
(b) I valori del processo genesi stimati dal secondo riscalamento

Figura 6.36: Processi genesi a confronto tra terza e seconda modulazione

Figura 6.37: Processo di inizio prima modulazione per dati sintetici

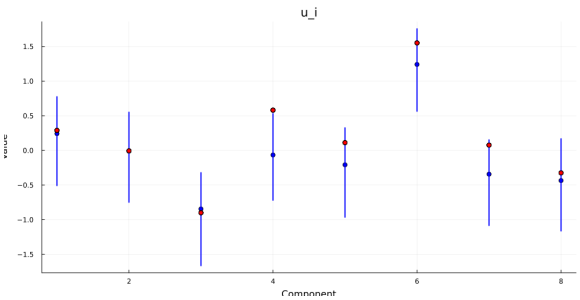


(a) Le realizzazione della prima modulazione per dati sintetici totali

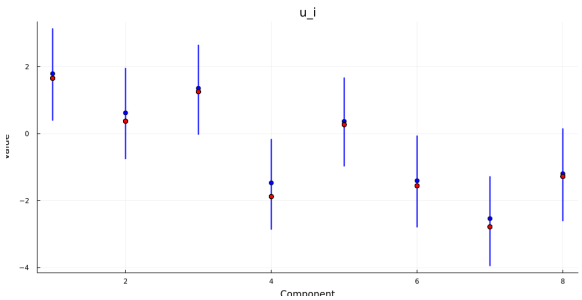


(b) Le realizzazione della prima modulazione per dati sintetici parziali

Figura 6.38: Le realizzazione della prima modulazione per dati sintetici

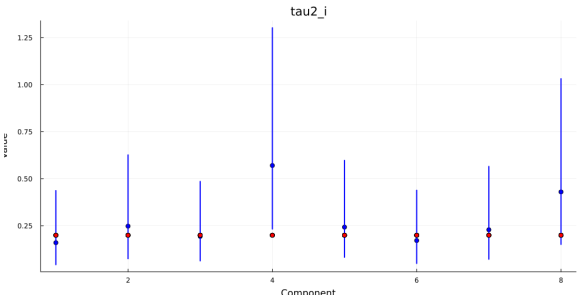


(a) Stime u per la prima modulazione su dati sintetici totali

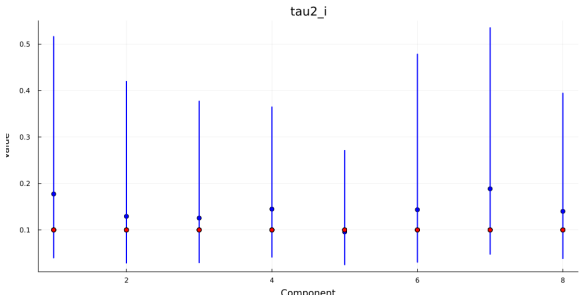


(b) Stime u per la prima modulazione su dati sintetici parziali

Figura 6.39: Stime u per la prima modulazione su dati sintetici

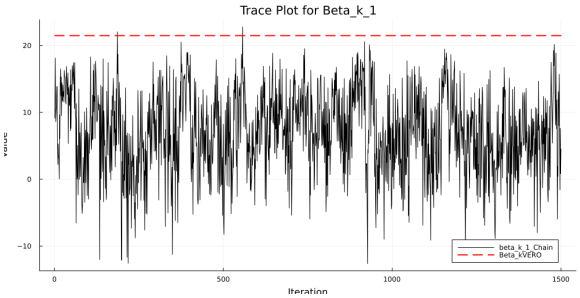


(a) Stime τ^2 per la prima modulazione su dati sintetici totali

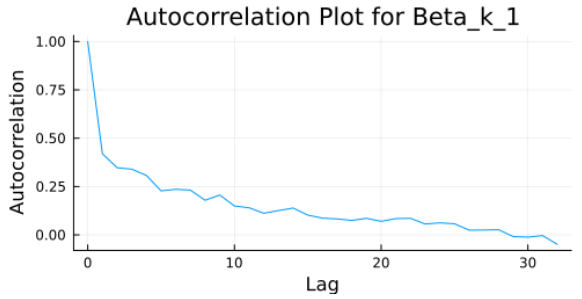


(b) Stime τ^2 per la prima modulazione su dati sintetici parziali

Figura 6.40: Stime τ^2 per la prima modulazione su dati sintetici

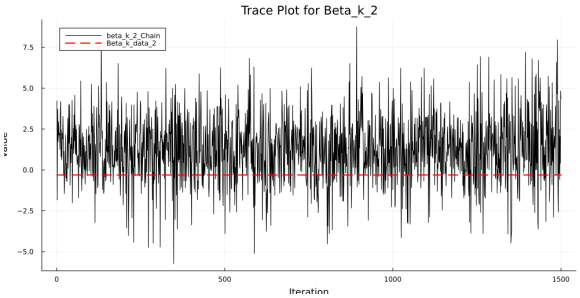


(a) Trace-Plot β_1 per la prima modulazione su dati sintetici totali

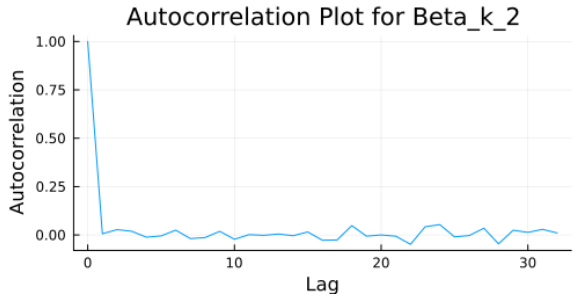


(b) Autocorrelation Plot β_1 per la prima modulazione su dati sintetici totali

Figura 6.41: Grafici β_1 per la prima modulazione su dati sintetici

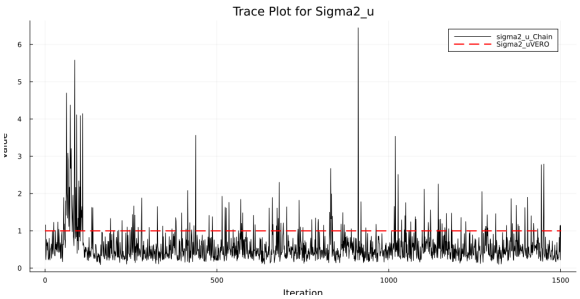


(a) Trace-Plot β_2 per la prima modulazione su dati sintetici totali

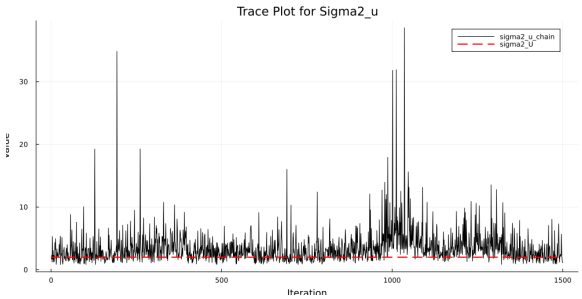


(b) Autocorrelation Plot β_2 per la prima modulazione su dati sintetici totali

Figura 6.42: Grafici β_2 per la prima modulazione su dati sintetici

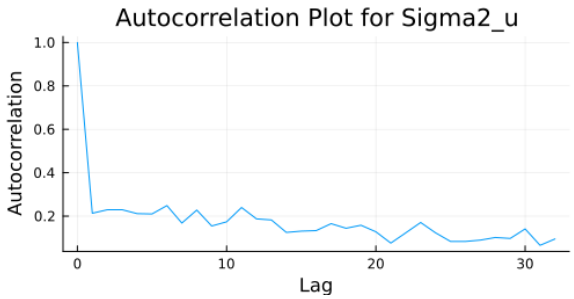


(a) Trace-Plot σ_u^2 per la prima modulazione su dati sintetici totali

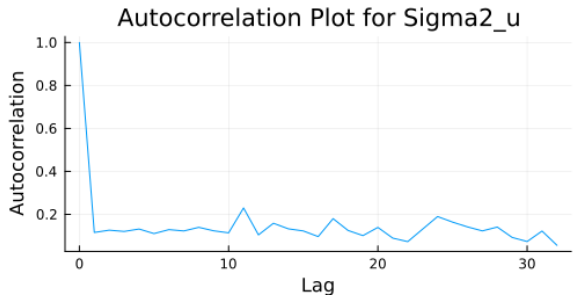


(b) Trace-Plot σ_u^2 per la prima modulazione su dati sintetici parziali

Figura 6.43: Trace-Plot σ_u^2 per la prima modulazione su dati sintetici

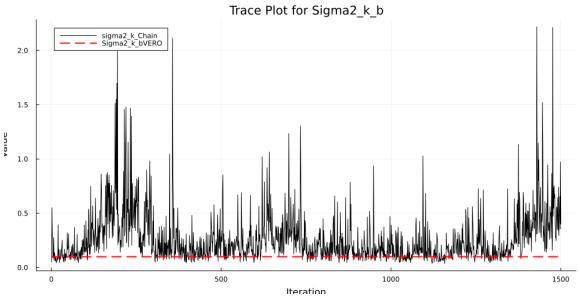


(a) Autocorrelation plot σ_u^2 per la prima modulazione su dati sintetici totali

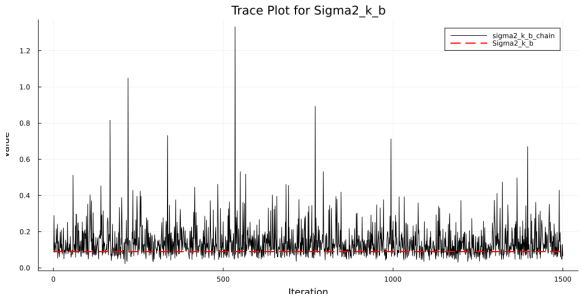


(b) Autocorrelation plot σ_u^2 per la prima modulazione su dati sintetici totali

Figura 6.44: Autocorrelation plot σ_u^2 per la prima modulazione su dati sintetici

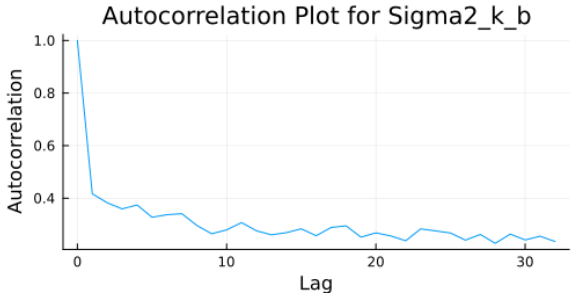


(a) Trace-Plot $\sigma_b^{2,k}$ per la prima modulazione su dati sintetici totali

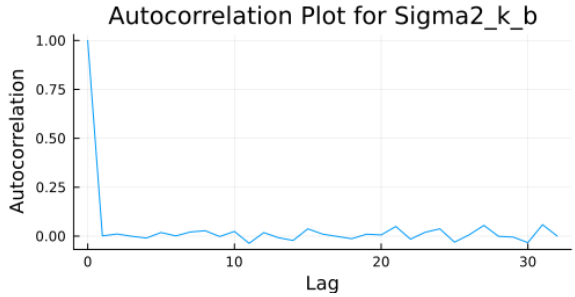


(b) Trace-Plot $\sigma_b^{2,k}$ per la prima modulazione su dati sintetici parziali

Figura 6.45: Trace-Plot $\sigma_b^{2,k}$ per la prima modulazione su dati sintetici

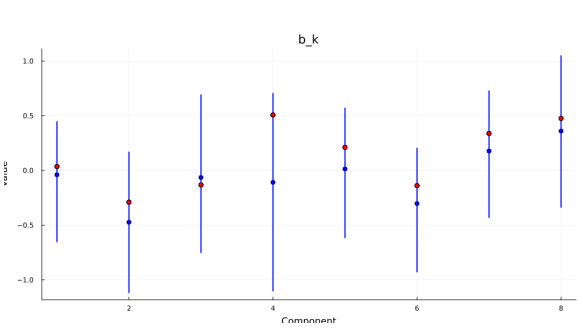


(a) Autocorrelation plot $\sigma_b^{2,k}$ per la prima modulazione su dati sintetici totali

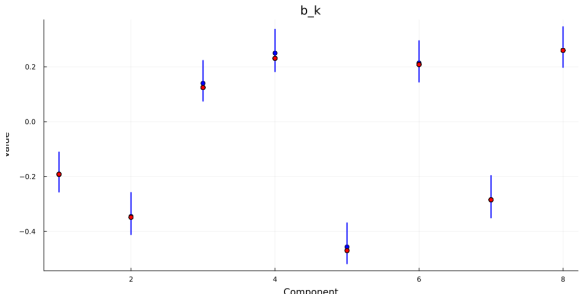


(b) Autocorrelation plot $\sigma_b^{2,k}$ per la prima modulazione su dati sintetici parziali

Figura 6.46: Autocorrelation plot $\sigma_b^{2,k}$ per la prima modulazione su dati sintetici

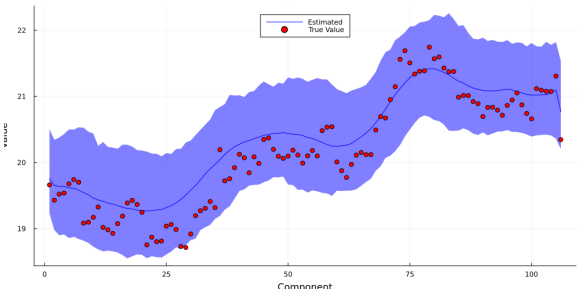


(a) Stima b_i prima modulazione su dati sintetici totali

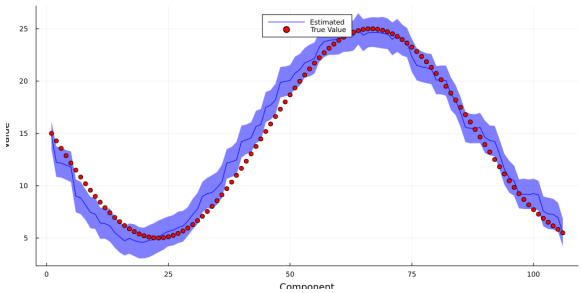


(b) Stima b_i prima modulazione su dati sintetici parziali

Figura 6.47: Stima b_i prima modulazione su dati sintetici

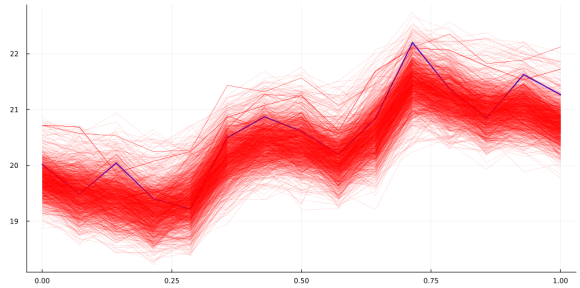


(a) Stima W_k^T prima modulazione su dati sintetici totali



(b) Stima W_k^T prima modulazione su dati sintetici parziali

Figura 6.48: Stima W_k^T prima modulazione su dati sintetici



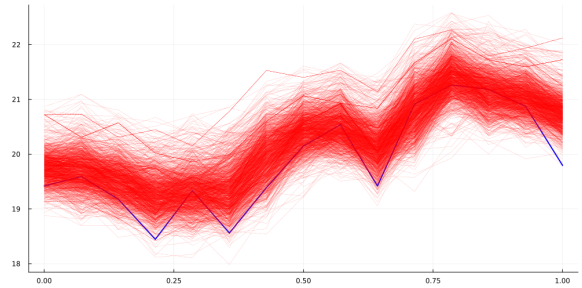
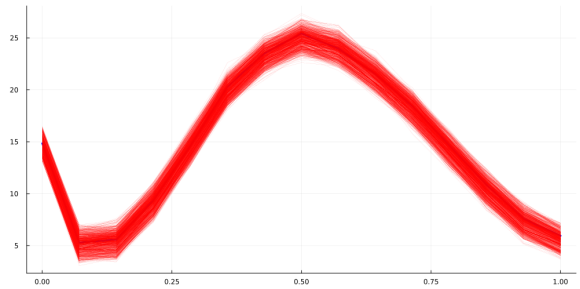


Figura 6.49: Due realizzazioni della prima modulazione per dati sintetici totali



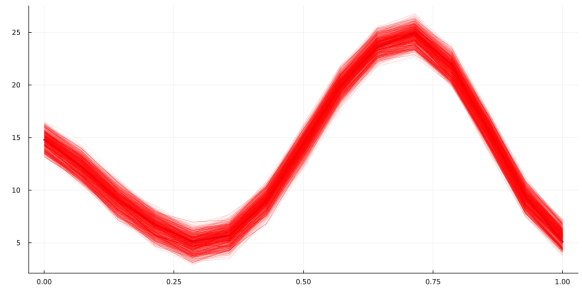


Figura 6.50: Due realizzazioni della prima modulazione per dati sintetici parziali

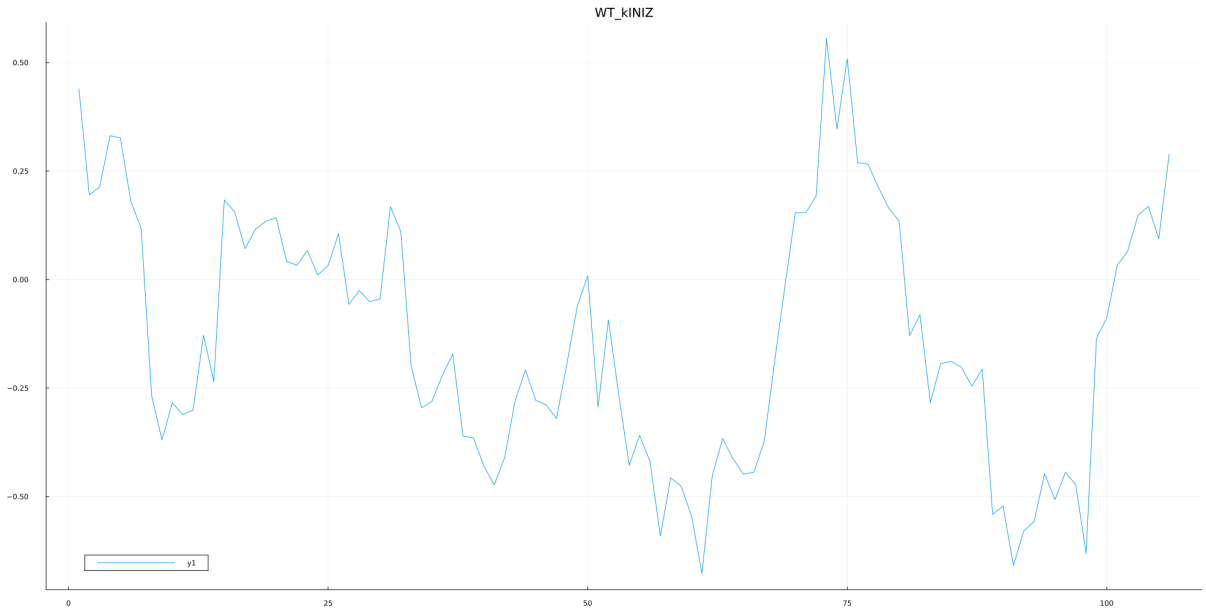
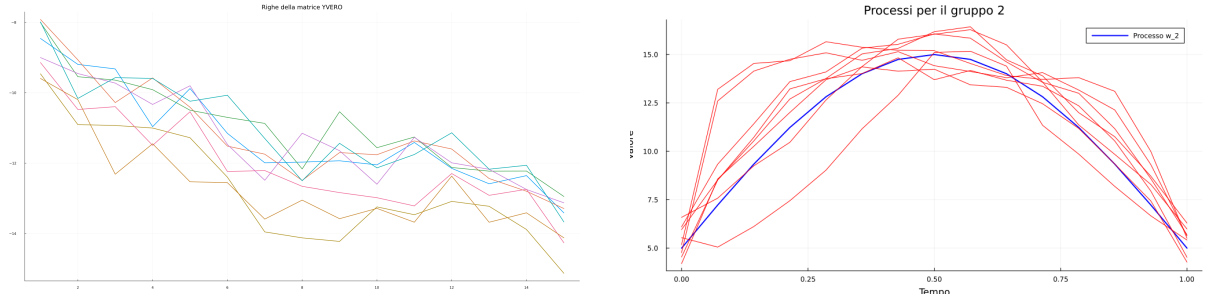


Figura 6.51: Processo di inizio seconda modulazione per dati sintetici



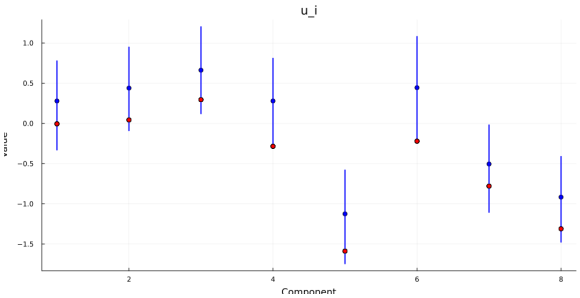
(a) Le realizzazione della seconda modulazione per dati sintetici totali

(b) Le realizzazione della seconda modulazione per dati sintetici parziali

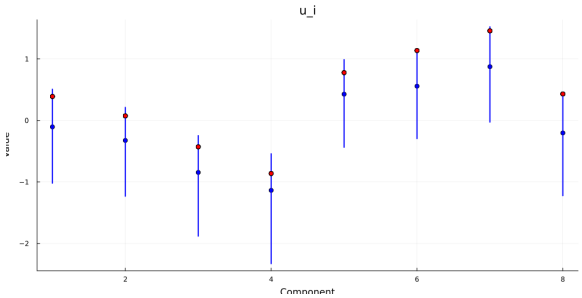
Figura 6.52: Le realizzazione della seconda modulazione per dati sintetici

Seconda Modulazione

I grafici 6.51, 6.52 mostrano nuovamente situazioni complesse di partenza; nonostante ciò, le stime di $\boldsymbol{u}$ e $\boldsymbol{\tau}^2$ risultano buone, 6.53, 6.54. Per β_1 e β_2, 6.55 6.56 i risultati sembrano timidamente migliorati rispetto a quelli visti per la prima modulazione, anche se β_1 continua ad essere problematico, almeno in termini di autocorrelazione. I risultati di σ_u^2, sono buoni per i dati sintetici totali, meno per quelli parziali, che invece presentano un trace-plot meno preciso ed un correlogramma con valori non trascurabili anche dopo alcuni lag, 6.57 6.58. Per $\sigma_b^{2,k}$ si osservano due comportamenti diversi per le due applicazioni: per i dati sintetici totali si ha una stima più accurata ma autocorrelazione non trascurabile; mentre per i dati parziali si ottiene un trace-plot meno preciso, ma con una forma migliore, supportata dal correlogramma, 6.59 6.60. Anche in questo caso le stime dei coefficienti $b_{1,i}$ e $b_{2,i}$ sono ottime, e nuovamente risultano particolarmente buone per il modello applicato ai dati parziali 6.61 6.62. Il trace plot del coefficiente δ è molto stabile, forse troppo, ma concede una stima accurata in entrambi i casi, seppur non raggiungendo mai il valore vero 6.63. Ne segue dunque che le modellizzazioni del processo W_k^T riescono ad emulare il loro corrispettivo reale abbastanza fedelmente nella forma, anche se alcuni valori non vengono racchiusi nell'intervallo di credibilità stimato 6.64. Malgrado la non perfetta inclusione del valore vero nell'intervallo di credibilità dei processi genesi, come spesso accade, le realizzazioni ricostruite sono molto aderenti a quelle vere 6.65 6.66. I risultati complessivi sono soddisfacienti, seppur migliorabili. Si considerano questi

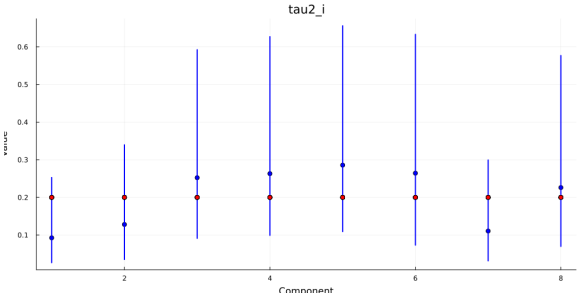


(a) Stime u per la seconda modulazione su dati sintetici totali

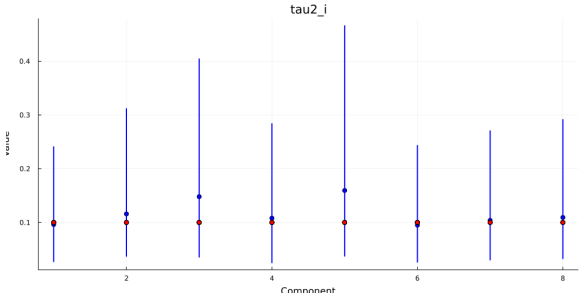


(b) Stime u per la Seconda modulazione su dati sintetici parziali

Figura 6.53: Stime u per la seconda modulazione su dati sintetici



(a) Stime τ^2 per la seconda modulazione su dati sintetici totali



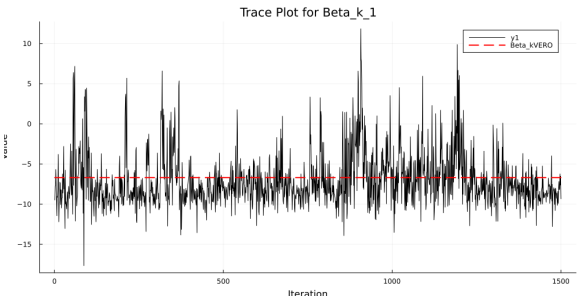
(b) Stime τ^2 per la seconda modulazione su dati sintetici parziali

Figura 6.54: Stime τ^2 per la seconda modulazione su dati sintetici

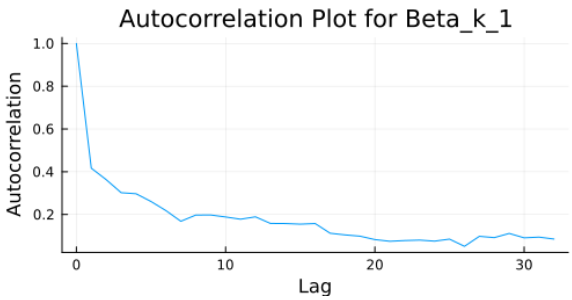
risultati postivi e, dunque, i modelli valevoli per le applicazioni su dati reali.

Terza Modulazione

L'impostazione per l'analisi su dati sintetici del terzo riscalamento risulta del tutto analoga alle due precedenti: le figure 6.67, 6.68 mostrano, rispettivamente, lo stato iniziale e i dati di riferimento. Le quantità rappresentate nelle figure: 6.69 6.70 6.73 6.74 sono molto incoraggianti e forniscono ottimi risultati sia in termini di stime che di autocorrelazione. Anche i grafici 6.71 6.72, per β_1 e β_2 sono ottimi, mostrando un ulteriore miglioramento rispetto ai precedenti due casi, persino nel trace plot di β_1 che è sistematicamente problematico. Mantengono anche ottimi risultati le stime per i coefficienti di modulazione $b_{1,i}$ e $b_{2,i}$ accompagnati da altrettanto buoni trace-plot delle quantità δ_1 e δ_2, seppur troppo stabili, 6.77 6.78 6.79 6.80. Risultati meno promettenti sono quelli mostrati in 6.75 e 6.76, che offrono, rispettivamente, andamenti e stime compromettenti per i dati sintetici totali e sintetici parziali. Infine le stime del processo sono eccellenti, almeno in termini di forma, a cui si aggiunge, nel caso totale, anche una buona centratura con conseguente inclusione di quasi tutti i valori veri, 6.81. Le rappresentazioni del processo latente rimodulato, messo a confronto con la realizzazione vera sono ideali 6.82 6.83. Anche questo riscalamento viene proposto per ulteriori indagini sui dati reali, essendo fortemente supportato dai risultati su dati sintetici.

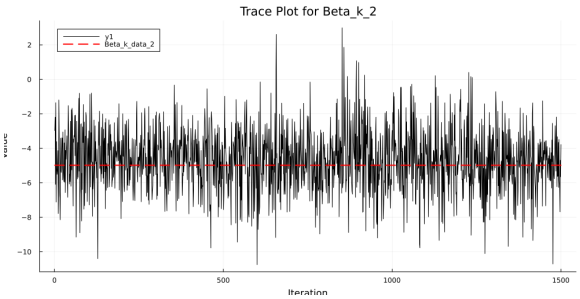


(a) Trace-Plot β_1 per la seconda modulazione su dati sintetici totali

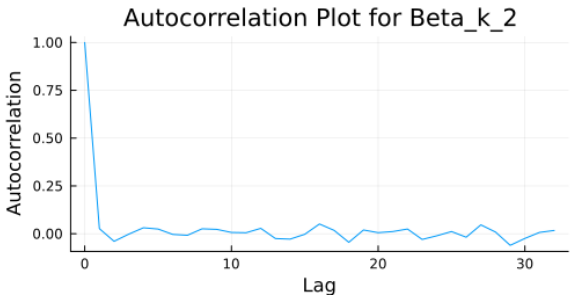


(b) Autocorrelation Plot β_1 per la seconda modulazione su dati sintetici totali

Figura 6.55: Grafici β_1 per la seconda modulazione su dati sintetici

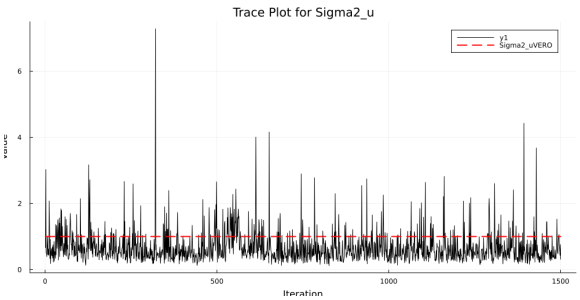


(a) Trace-Plot β_2 per la seconda modulazione su dati sintetici totali

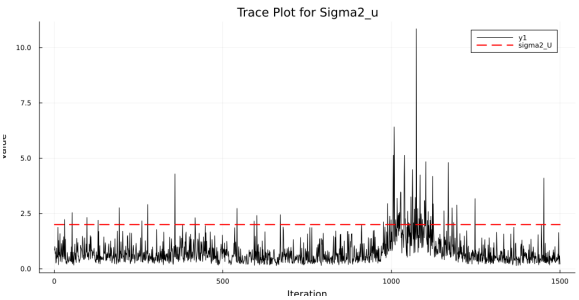


(b) Autocorrelation Plot β_2 per la seconda modulazione su dati sintetici totali

Figura 6.56: Grafici β_2 per la seconda modulazione su dati sintetici

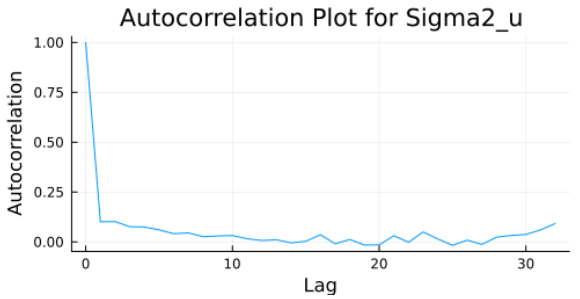


(a) Trace-Plot σ_u^2 per la seconda modulazione su dati sintetici totali

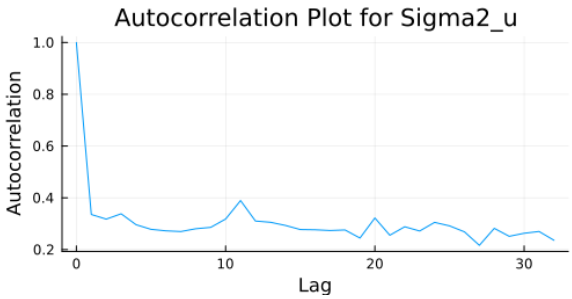


(b) Trace-Plot σ_u^2 per la seconda modulazione su dati sintetici parziali

Figura 6.57: Trace-Plot σ_u^2 per la seconda modulazione su dati sintetici

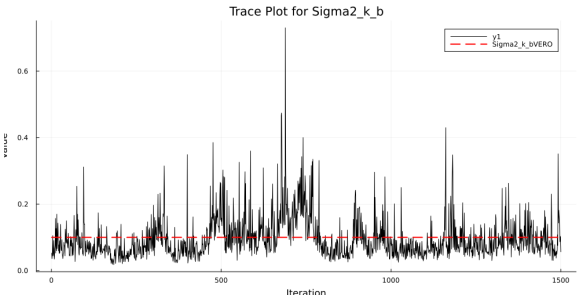


(a) Autocorrelation plot σ_u^2 per la seconda modulazione su dati sintetici totali

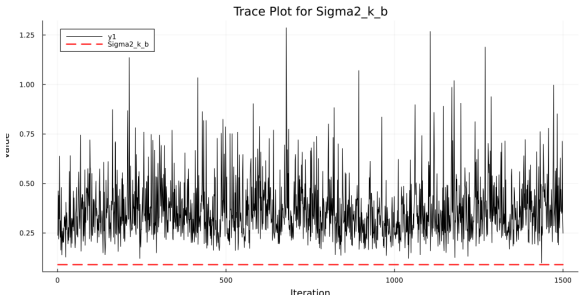


(b) Autocorrelation plot σ_u^2 per la seconda modulazione su dati sintetici totali

Figura 6.58: Autocorrelation plot σ_u^2 per la seconda modulazione su dati sintetici

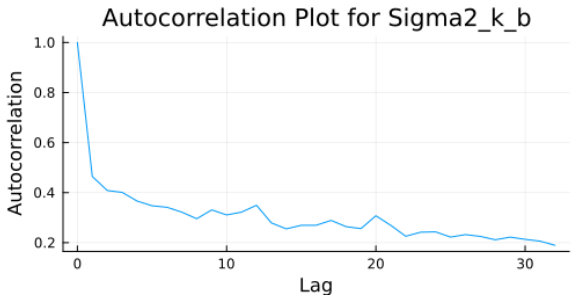


(a) Trace-Plot $\sigma_b^{2,k}$ per la seconda modulazione su dati sintetici totali

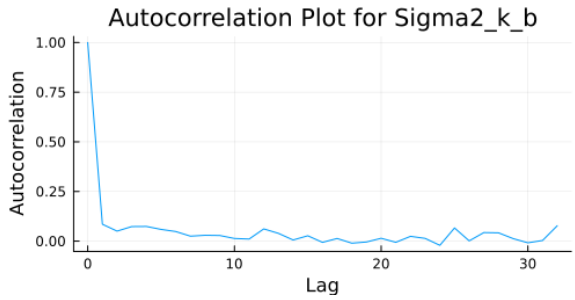


(b) Trace-Plot $\sigma_b^{2,k}$ per la seconda modulazione su dati sintetici parziali

Figura 6.59: Trace-Plot $\sigma_b^{2,k}$ per la seconda modulazione su dati sintetici

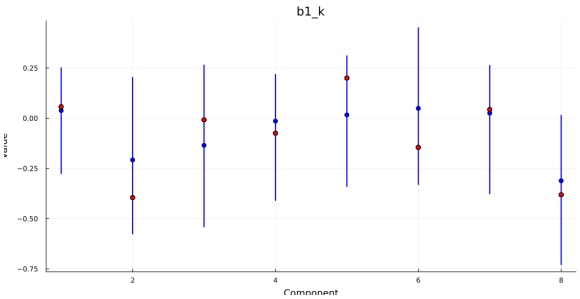


(a) Autocorrelation plot $\sigma_b^{2,k}$ per la seconda modulazione su dati sintetici totali

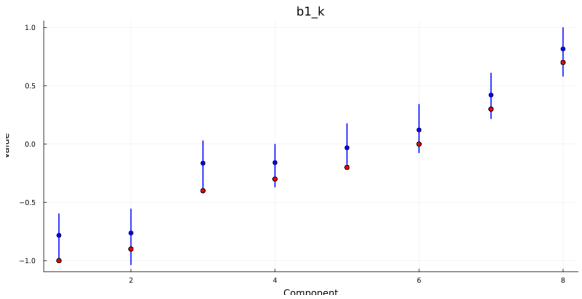


(b) Autocorrelation plot $\sigma_b^{2,k}$ per la seconda modulazione su dati sintetici parziali

Figura 6.60: Autocorrelation plot $\sigma_b^{2,k}$ per la seconda modulazione su dati sintetici

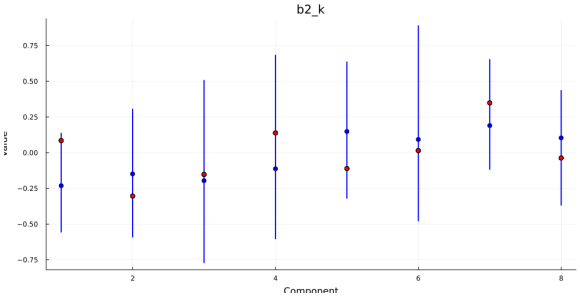


(a) Stima $b_{1,i}$ seconda modulazione su dati sintetici totali

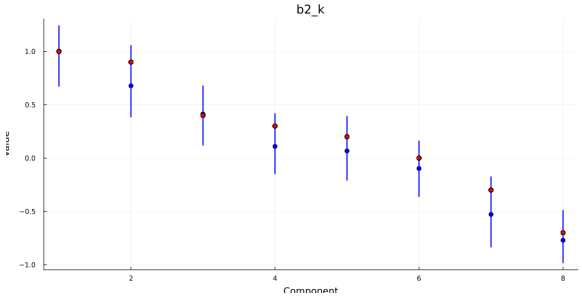


(b) Stima $b_{1,i}$ seconda modulazione su dati sintetici parziali

Figura 6.61: Stima $b_{1,i}$ seconda modulazione su dati sintetici

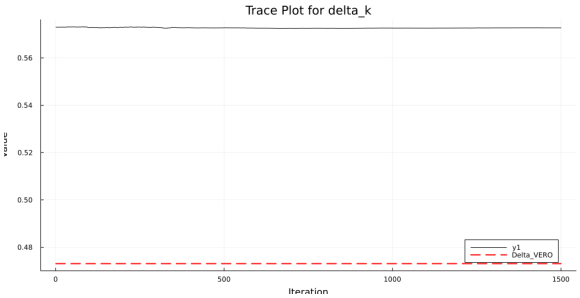


(a) Stima $b_{2,i}$ seconda modulazione su dati sintetici totali

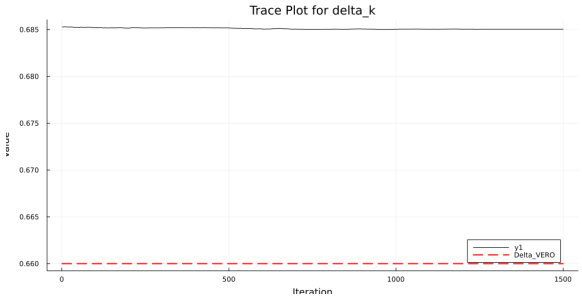


(b) Stima $b_{2,i}$ seconda modulazione su dati sintetici parziali

Figura 6.62: Stima $b_{2,i}$ seconda modulazione su dati sintetici

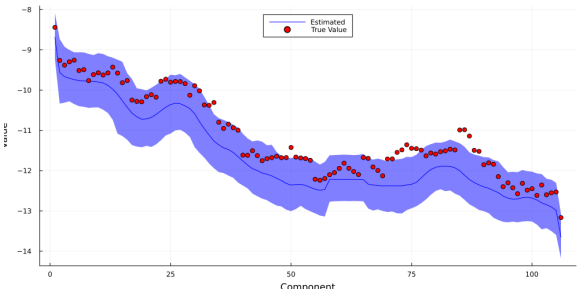


(a) Trace-Plot δ seconda modulazione su dati sintetici totali

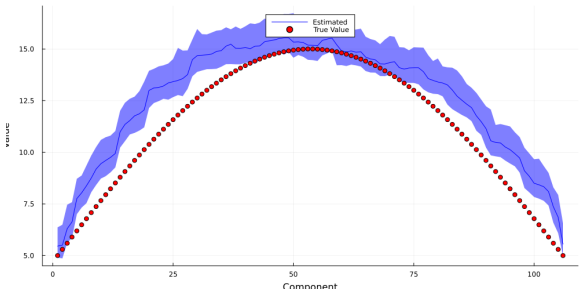


(b) Trace-Plot δ seconda modulazione su dati sintetici parziali

Figura 6.63: Trace-Plot δ seconda modulazione su dati sintetici

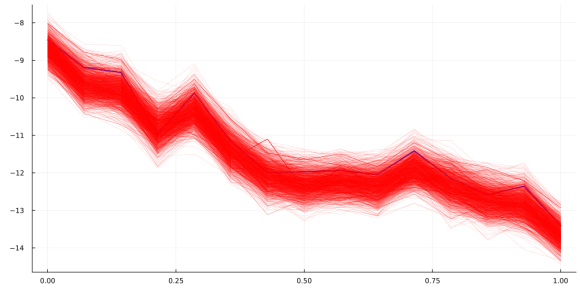


(a) Trace-Plot W_k^T seconda modulazione su dati sintetici totali



(b) Stima W_k^T seconda modulazione su dati sintetici parziali

Figura 6.64: Stima W_k^T seconda modulazione su dati sintetici



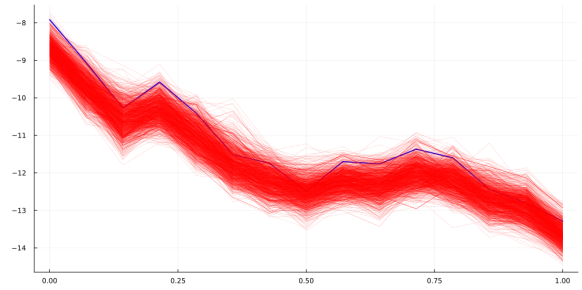
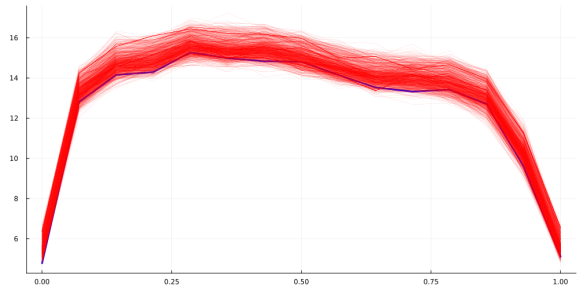


Figura 6.65: Due realizzazioni della seconda modulazione per dati sintetici totali



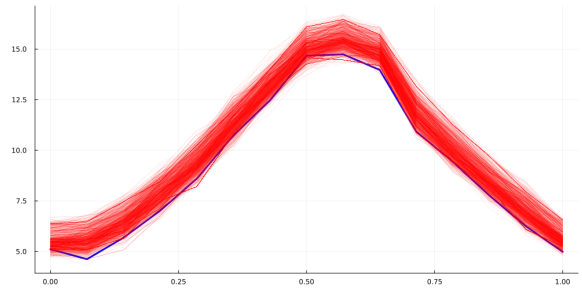


Figura 6.66: Due realizzazioni della seconda modulazione per dati sintetici parziali

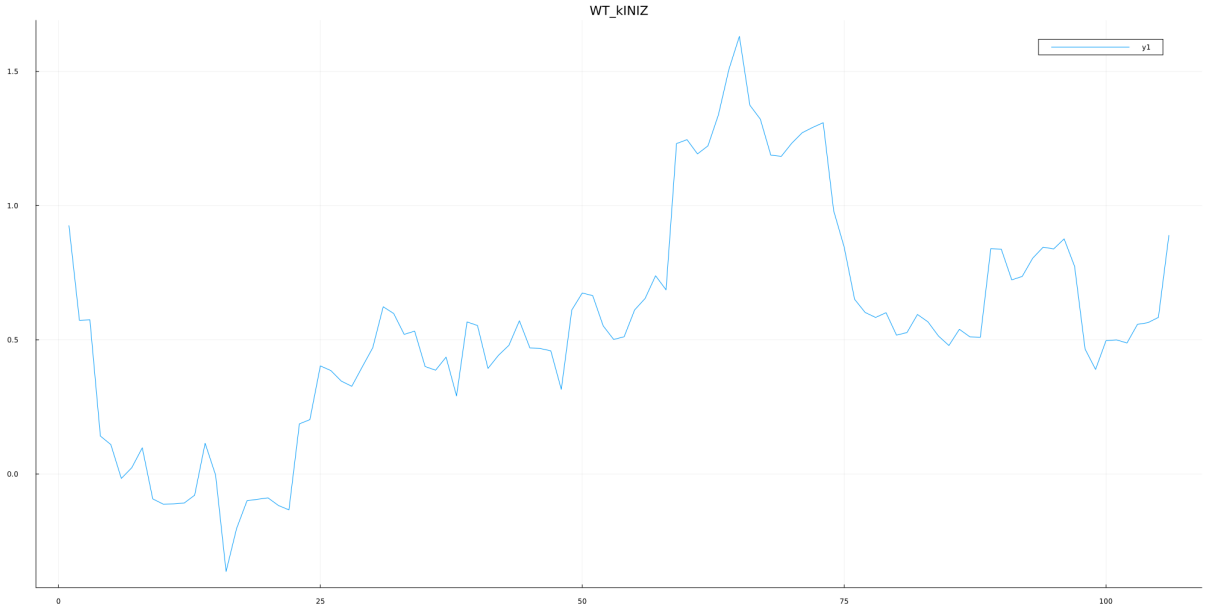
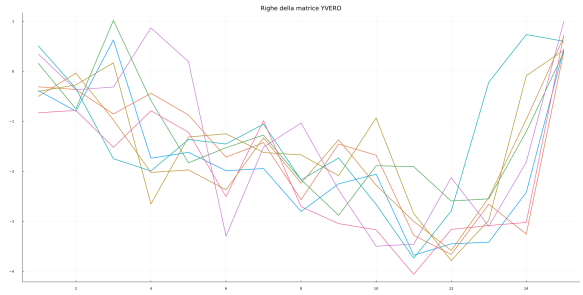
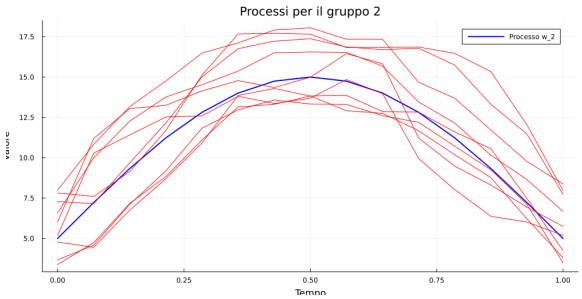


Figura 6.67: Processo di inizio terza modulazione per dati sintetici

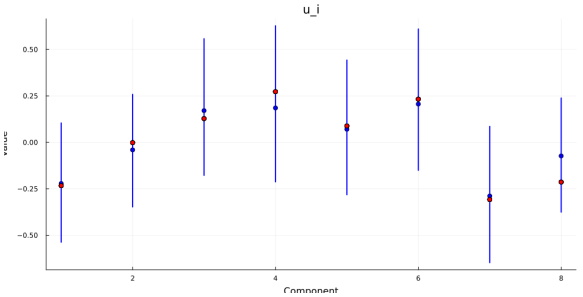


(a) Le realizzazione della terza modulazione per dati sintetici totali

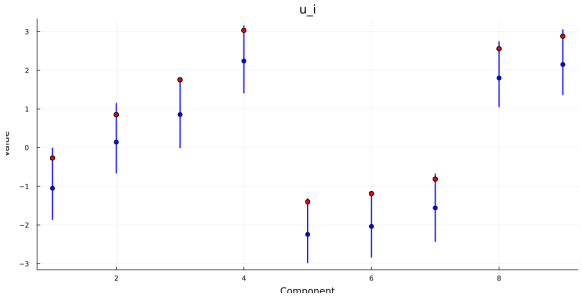


(b) Le realizzazione della terza modulazione per dati sintetici parziali

Figura 6.68: Le realizzazione della terza modulazione per dati sintetici

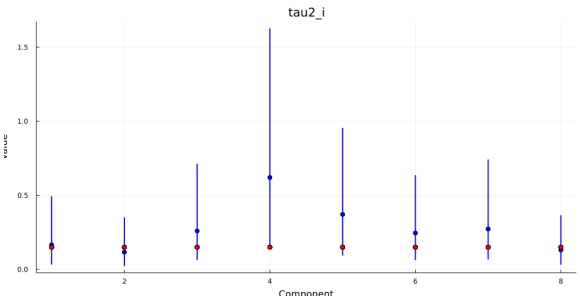


(a) Stime u per la terza modulazione su dati sintetici totali

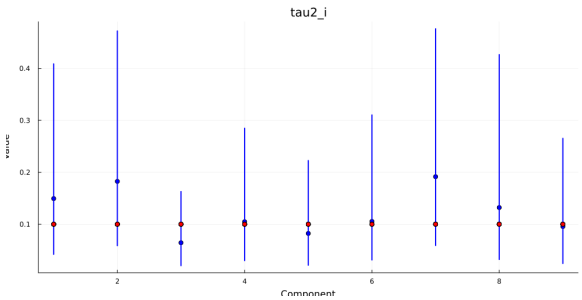


(b) Stime u per la terza modulazione su dati sintetici parziali

Figura 6.69: Stime u per la terza modulazione su dati sintetici

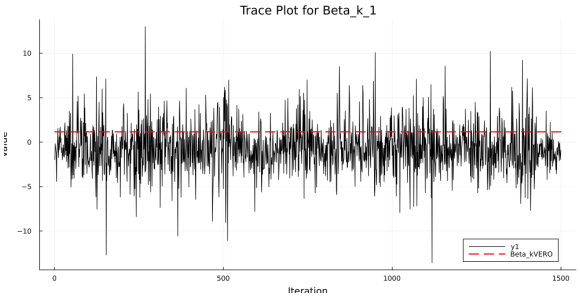


(a) Stime τ^2 per la terza modulazione su dati sintetici totali

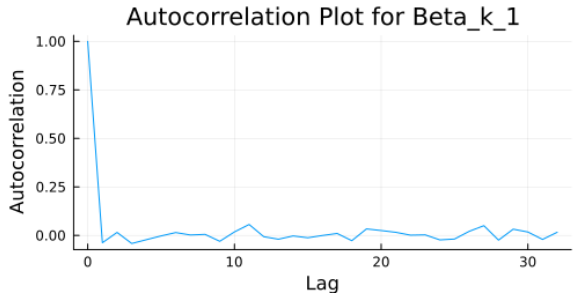


(b) Stime τ^2 per la terza modulazione su dati sintetici parziali

Figura 6.70: Stime τ^2 per la terza modulazione su dati sintetici

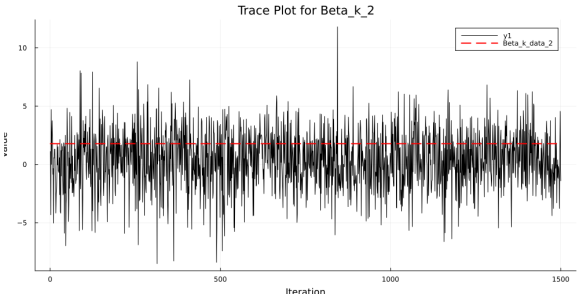


(a) Trace-Plot β_1 per la terza modulazione su dati sintetici totali

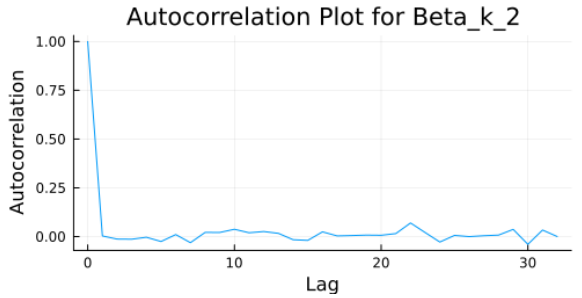


(b) Autocorrelation Plot β_1 per la terza modulazione su dati sintetici totali

Figura 6.71: Grafici β_1 per la terza modulazione su dati sintetici

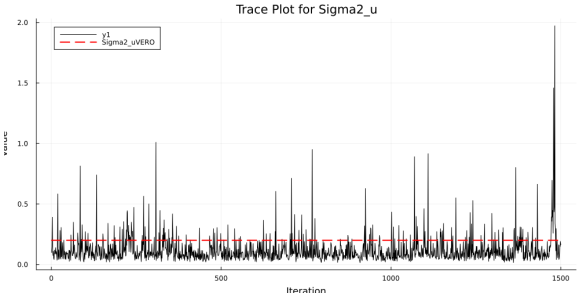


(a) Trace-Plot β_2 per la terza modulazione su dati sintetici totali

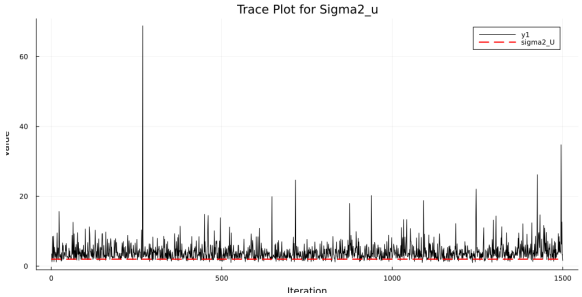


(b) Autocorrelation Plot β_2 per la terza modulazione su dati sintetici totali

Figura 6.72: Grafici β_2 per la terza modulazione su dati sintetici

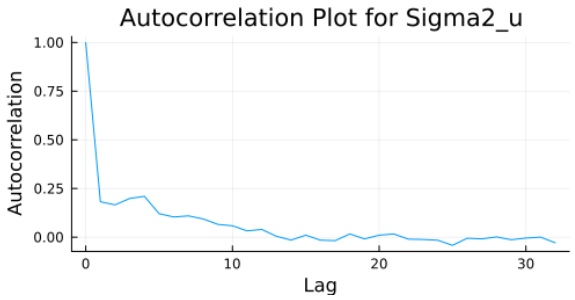


(a) Trace-Plot σ_u^2 per la terza modulazione su dati sintetici totali

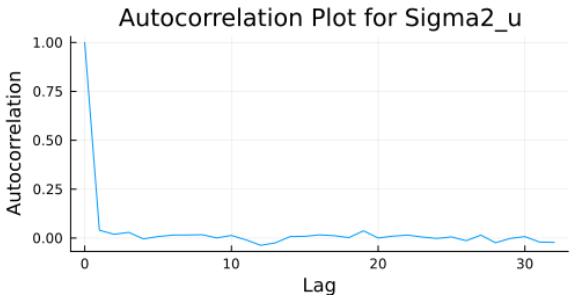


(b) Trace-Plot σ_u^2 per la terza modulazione su dati sintetici parziali

Figura 6.73: Trace-Plot σ_u^2 per la terza modulazione su dati sintetici

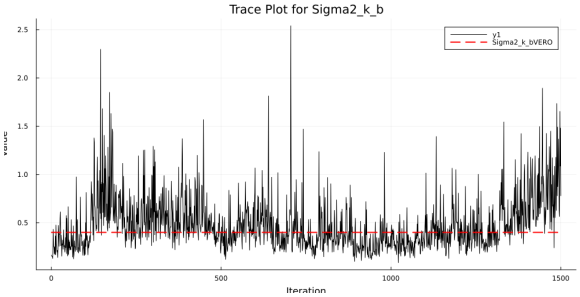


(a) Autocorrelation plot σ_u^2 per la terza modulazione su dati sintetici totali

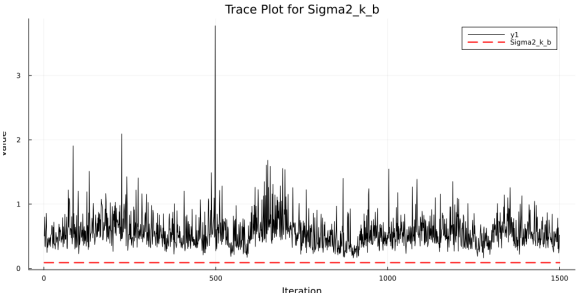


(b) Autocorrelation plot σ_u^2 per la terza modulazione su dati sintetici totali

Figura 6.74: Autocorrelation plot σ_u^2 per la terza modulazione su dati sintetici

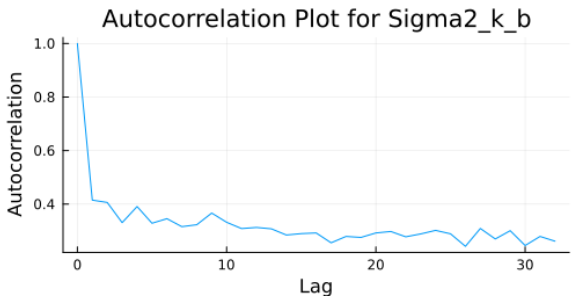


(a) Trace-Plot $\sigma_b^{2,k}$ per la terza modulazione su dati sintetici totali

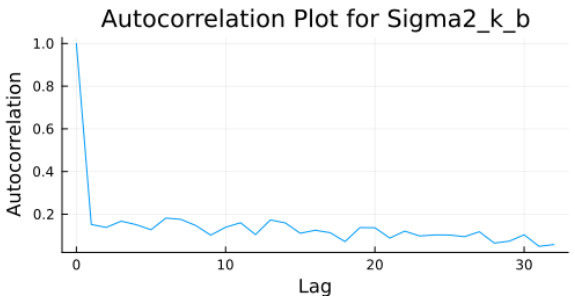


(b) Trace-Plot $\sigma_b^{2,k}$ per la terza modulazione su dati sintetici parziali

Figura 6.75: Trace-Plot $\sigma_b^{2,k}$ per la terza modulazione su dati sintetici

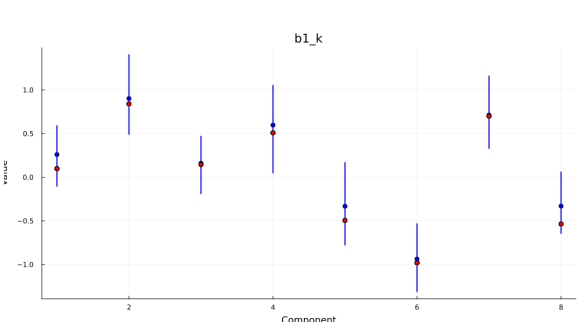


(a) Autocorrelation plot $\sigma_b^{2,k}$ per la terza modulazione su dati sintetici totali

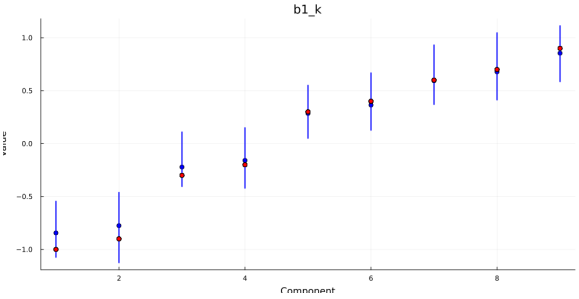


(b) Autocorrelation plot $\sigma_b^{2,k}$ per la terza modulazione su dati sintetici parziali

Figura 6.76: Autocorrelation plot $\sigma_b^{2,k}$ per la terza modulazione su dati sintetici

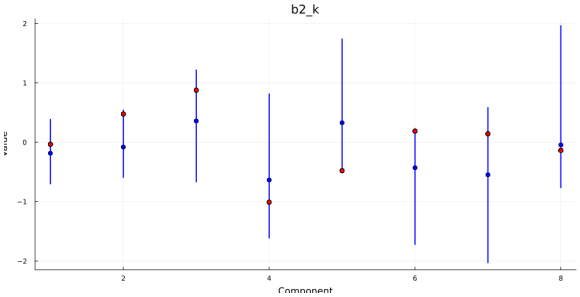


(a) Stima $b_{1,i}$ terza modulazione su dati sintetici totali

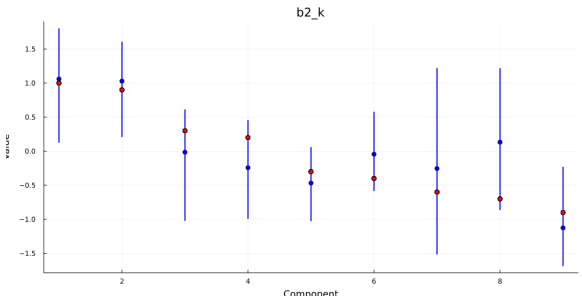


(b) Stima $b_{1,i}$ terza modulazione su dati sintetici parziali

Figura 6.77: Stima $b_{1,i}$ terza modulazione su dati sintetici

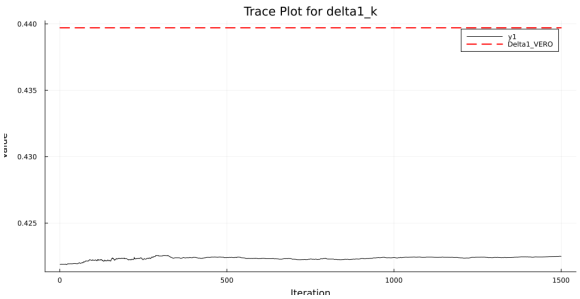


(a) Stima $b_{2,i}$ terza modulazione su dati sintetici totali

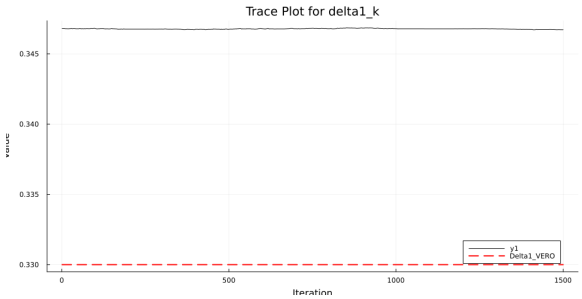


(b) Stima $b_{2,i}$ seconda modulazione su dati sintetici parziali

Figura 6.78: Stima $b_{2,i}$ terza modulazione su dati sintetici

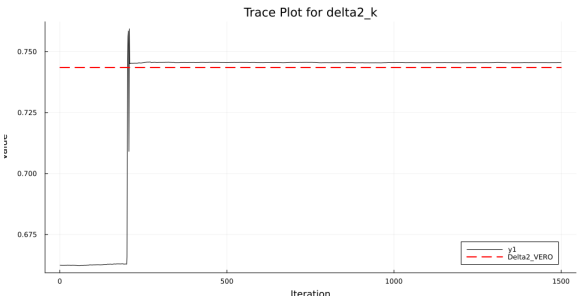


(a) Trace-Plot δ_1 terza modulazione su dati sintetici totali

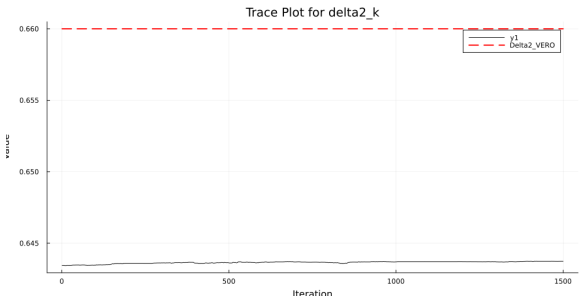


(b) Trace-Plot δ_1 terza modulazione su dati sintetici parziali

Figura 6.79: Trace-Plot δ_1 terza modulazione su dati sintetici

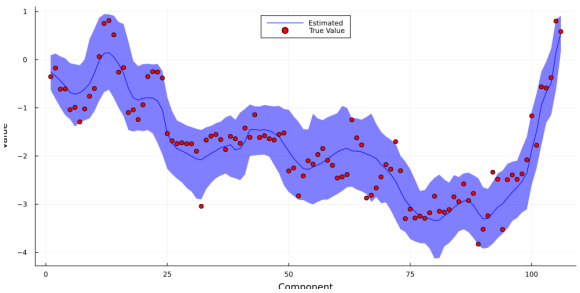


(a) Trace-Plot δ_2 terza modulazione su dati sintetici totali

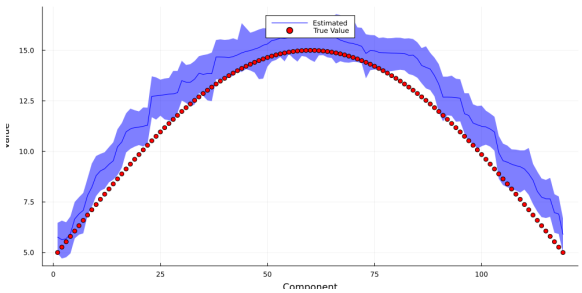


(b) Trace-Plot δ_2 terza modulazione su dati sintetici parziali

Figura 6.80: Trace-Plot δ_2 terza modulazione su dati sintetici



(a) Trace-Plot W_k^T seconda modulazione su dati sintetici totali



(b) Stima W_k^T seconda modulazione su dati sintetici parziali

Figura 6.81: Stima W_k^T terza modulazione su dati sintetici

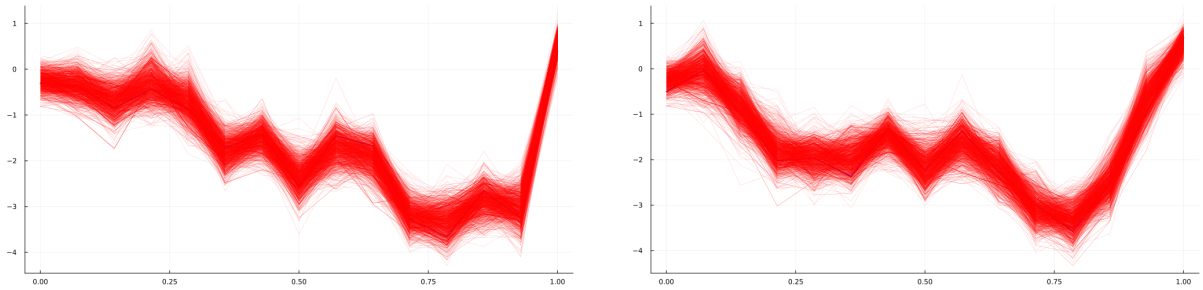


Figura 6.82: Due realizzazioni della terza modulazione per dati sintetici totali

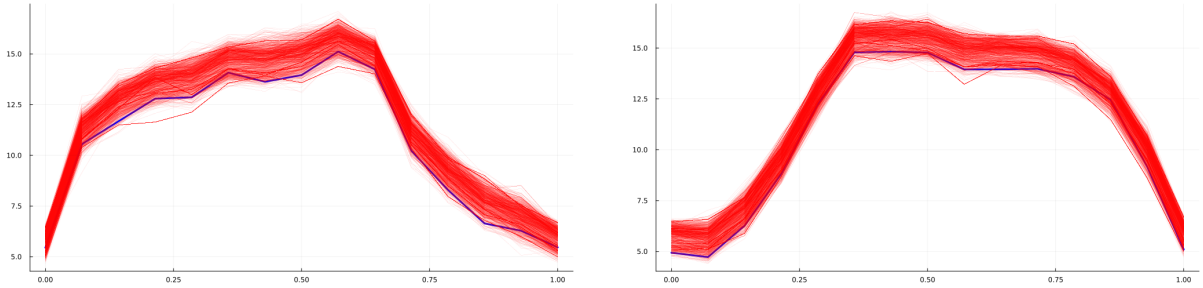


Figura 6.83: Due realizzazioni della terza modulazione per dati sintetici parziali

Capitolo 7

Applicazione sui Dati Reali

Grazie all'approfondita analisi condotta sulla qualità delle stime e sull'affidabilità dei metodi, come mostrato nel capitolo dedicato ai dati sintetici, è ora possibile offrire una panoramica chiara e concisa su diverse realizzazioni reali. Infatti, per garantire un'esposizione efficace e non dispersiva, ci si focalizza su un numero limitato di aspetti chiave.

In particolare, l'attenzione verrà rivolta a tre elementi principali:

- la modellizzazione del processo di generazione W_k^T secondo l'algoritmo impiegato;
- il grado di aderenza del processo latente rimodulato rispetto alla realizzazione osservata;
- i coefficienti di transizione da una modulazione all'altra per il secondo ed il terzo modello.

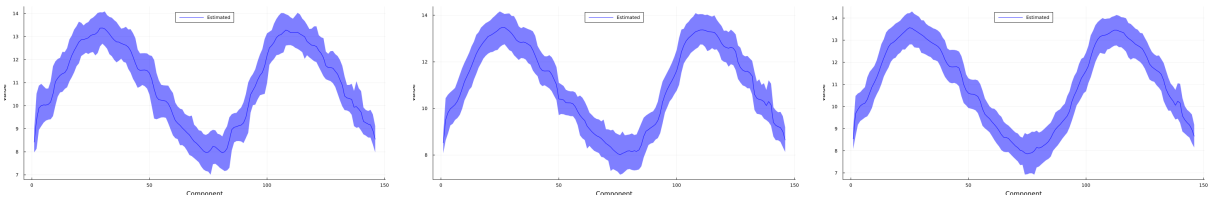
Di seguito, viene presentata una selezione di grafici accompagnati da commenti esplicativi, relativi ai risultati ottenuti applicando i modelli scelti a sei fischi-firma reali. Per evitare ridondanze e migliorare la leggibilità, sono state incluse unicamente le istanze più significative e informative per ciascun esemplare, privilegiando quelle che meglio rappresentano gli aspetti di maggiore interesse.

SW 032

Tra le modellizzazioni del processo W_k^T, in figura 7.1, non vi è sostanziale differenza, al di là dell'andamento più liscio nella seconda parte del grafico .c, che suggerisce forse una migliore rappresentazione del processo da parte della terza modulazione. I trace-plot per i coefficienti δ in 7.2 evidenziano una modulazione intorno alla metà del processo, anche se con risultati così simili è difficile apprezzare la differenza di tale contributo. Le due rappresentazioni più interessanti sono visibili nelle figure 7.3 e 7.4. Nello specifico, la figura 7.3 viene scelta come rappresentante di un comportamento ottimo che viene confermato anche nelle altre istanze (non rappresentate per sintesi, ma disponibili su GitHub); invece l'esempio 7.4 esplicita il comportamento dei modelli su un'istanza meno canonica, in cui si possono apprezzare le rappresentazioni che con l'aumentare della complessità dei modelli sembrano migliorare, seppur non riuscendo a contenere completamente la realizzazione.

SW 042

Per il fischio-firma 042, invece, si evidenzia una divisione netta in termini di qualità di stima del processo tra il primo modello e gli altri due. Nella figura 7.5 si nota una forma meno liscia per il processo stimato, ed il conseguente intervallo di credibilità risulta altrattanto frenetico. Le stime del primo modello per i dati, nelle figure 7.7 e 7.8 rispecchiano la "brutta" stima del modello, mostrando risultati inferiori. D'altro canto, seppure con delle stime dei coefficienti δ diverse, figura 7.6, i risultati delle altre due

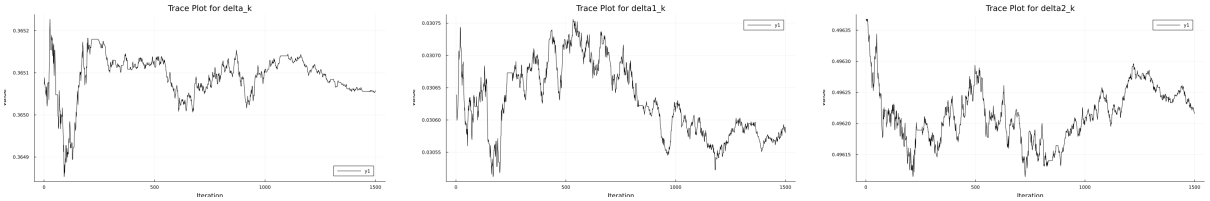


(a) Modellizzazione del fischio-firma SW 032, con prima rimodulazione temporale

(b) Modellizzazione del fischio-firma SW 032, con seconda rimodulazione temporale

(c) Modellizzazione del fischio-firma SW 032, con terza rimodulazione temporale

Figura 7.1: Modellizzazione del fischio-firma SW 032



(a) Trace-Plot del parametro δ per la seconda modulazione

(b) Trace-Plot del parametro δ_1 per la seconda modulazione

(c) Trace-Plot del parametro δ_2 per la seconda modulazione

Figura 7.2: Trace-Plot parametro δ SW 032

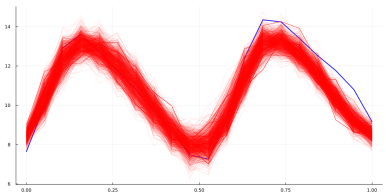
modulazioni sembrano eque. Eque, ma non uguali, visto che vi sono alcune differenze sottili sia nella stima del massimo del processo W_k^T che nella rimodulazione temporale del processo latente della realizzazione 5, in figura 7.8, che sembrerebbe mostrare risultati leggeremente migliori per il terzo modello, emulando più fedelmente la curvatura della parte centrale del fischio.

SW 044

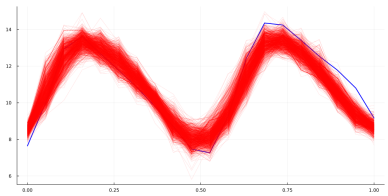
Nel caso della realizzazione 044, si vede un comportamento peculiare, che rispecchia quanto osservato nel paragrafo di coerenza tra modelli. Esplicitamente il modello più generale, il terzo, converge spontaneamente al secondo, evidente dalla vicinanza del coefficiente δ_2 ad 1 e dalla vicinanza dei valori di δ e δ_1 nei trace plot nella figura 7.10. A livello di stime, nuovamente, si riscontrano ottimi risultati da parte di tutti e tre i modelli, con una chiara supremazia dei due più complessi rispetto al primo, evidenziata dalla inferiore regolarità nella stime del processo genesi, figura 7.9, e dalla rappresentazione inferiore delle realizzazioni in 7.11 e 7.12, seppur buone di per sé.

SW 056

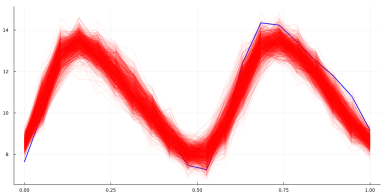
Questo caso porta alla luce un'eventualità ancora mai incontrata, in cui il terzo modello si discosta dal secondo, malgrado scelga anch'esso il riscalamento unico δ, imponendo l'altro $d_2 \approx 1$, figura 7.14. Questo suggerisce che uno dei due modelli non è riuscito a giungere a convergenza nelle iterazioni fornite, presumibilmente il terzo, vista la superiorità della rimodulazione offerta dal secondo modello in figura 7.17. Riguardo alla prima modulazione, essa manifesta in maniera evidente un'inferiorità rispetto alle altre due, chiara dalle irregolarità in figura 7.13 e dalla realizzazione inferiore in 7.16. Può valere la pena rimarcare che si sta parlando di ottimi risultati anche per la prima modulazione, come giustamente mostrato da 7.15 e in tutte le altre istanze non riportate per facilitare la visualizzazione. Tuttavia, la bontà dei risultati complessiva costringe ad evidenziare piccole inferiorità laddove possibile.



(a) Prima rimodulazione temporale del processo latente per la realizzazione 2 di SW 032

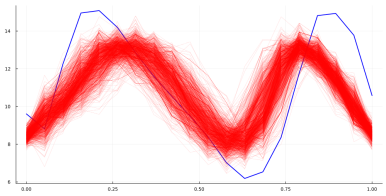


(b) Seconda rimodulazione temporale del processo latente per la realizzazione 2 di SW 032

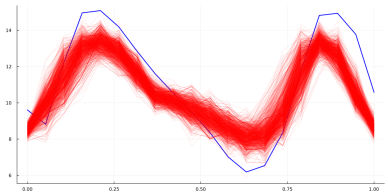


(c) Terza rimodulazione temporale del processo latente per la realizzazione 2 di SW 032

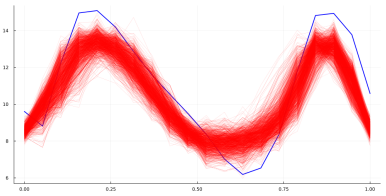
Figura 7.3: Rimodulazione temporale del processo latente per la realizzazione 2 di SW 032



(a) Prima rimodulazione temporale del processo latente per la realizzazione 8 di SW 032



(b) Seconda rimodulazione temporale del processo latente per la realizzazione 8 di SW 032



(c) Terza rimodulazione temporale del processo latente per la realizzazione 8 di SW 032

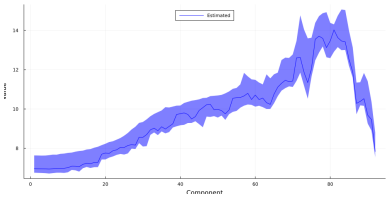
Figura 7.4: Rimodulazione temporale del processo latente per la realizzazione 8 di SW 032

SW 067

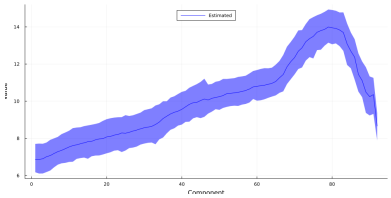
Per queste realizzazioni si osserva una situazione simile a quanto visto per il SW 044, dove il terzo modello collassa nel secondo, presentando un valore di δ_2 prossimo ad 1, figura 7.19. Tuttavia i due modelli complessi, in questo frangente, non sembrano surclassare il primo modello, se non per contorni dei fischi più aderenti, che però si basano su modellizzazioni del processo generatore, praticamente identici. Tutto visibile nelle figure 7.18, 7.20, 7.21

SW 078

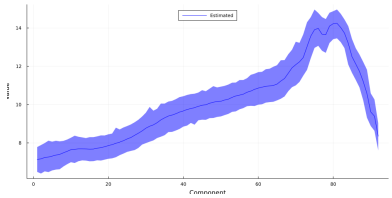
Anche con realizzazioni più complesse il risultato rimane pressoché invariato e questo tipo di fischio-firma ne è la prova; le stime sono ottime malgrado la forma non banale e anche le figure non riportate mostrano essenzialmente lo stesso comportamento. La stima del processo genesi è eccellente. I trace plot dei coefficienti di transizione sono diversi, ma conducono praticamente allo stesso risultato; che per essere precisi, risulta vagamente meglio per il terzo riscalamento, il quale mostra leggermente maggiore regolarità rispetto ai primi due. Tutto riassunto dalle figure 7.22 7.24 7.23



(a) Modellizzazione del fischio-firma SW 042, con prima rimodulazione temporale

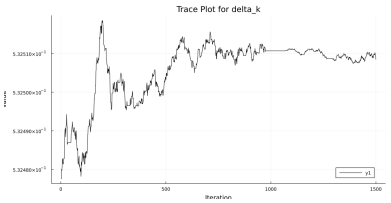


(b) Modellizzazione del fischio-firma SW 042, con seconda rimodulazione temporale

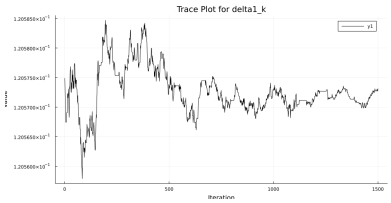


(c) Modellizzazione del fischio-firma SW 042, con terza rimodulazione temporale

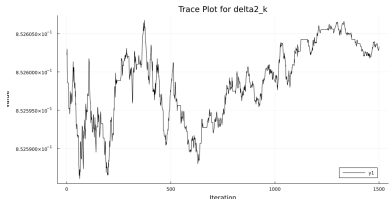
Figura 7.5: Modellizzazione del fischio-firma SW 042



(a) Trace-Plot del parametro δ per la seconda modulazione

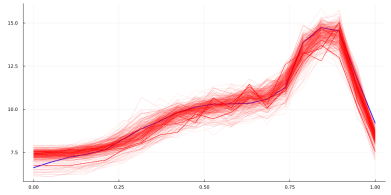


(b) Trace-Plot del parametro δ_1 per la seconda modulazione

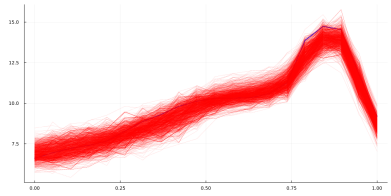


(c) Trace-Plot del parametro δ_2 per la seconda modulazione

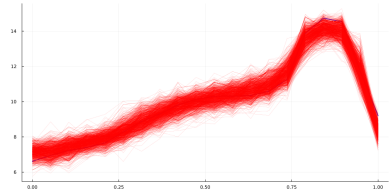
Figura 7.6: Trace-Plot parametro δ SW 042



(a) Prima rimodulazione temporale del processo latente per la realizzazione 1 di SW 042

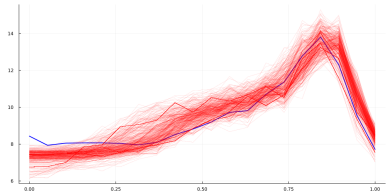


(b) Seconda rimodulazione temporale del processo latente per la realizzazione 1 di SW 042

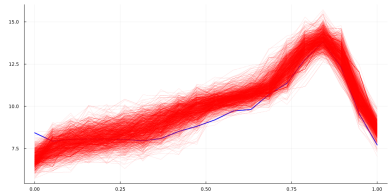


(c) Terza rimodulazione temporale del processo latente per la realizzazione 1 di SW 042

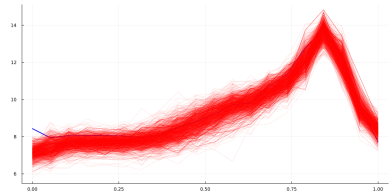
Figura 7.7: Rimodulazione temporale del processo latente per la realizzazione 1 di SW 042



(a) Prima rimodulazione temporale del processo latente per la realizzazione 5 di SW 042

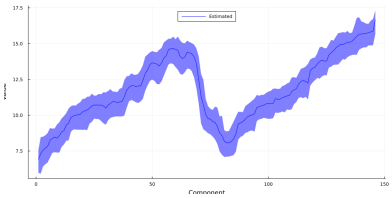


(b) Seconda rimodulazione temporale del processo latente per la realizzazione 5 di SW 042

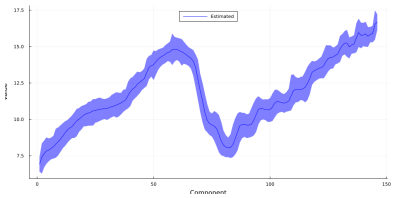


(c) Terza rimodulazione temporale del processo latente per la realizzazione 5 di SW 042

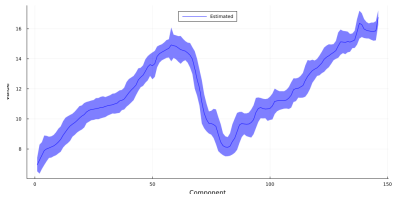
Figura 7.8: Rimodulazione temporale del processo latente per la realizzazione 5 di SW 042



(a) Modellizzazione del fischio-firma SW 044, con prima rimodulazione temporale

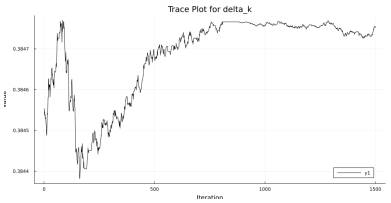


(b) Modellizzazione del fischio-firma SW 044, con seconda rimodulazione temporale

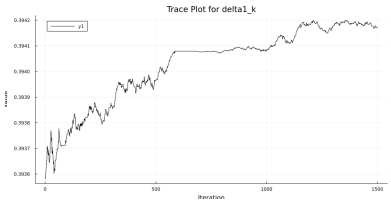


(c) Modellizzazione del fischio-firma SW 044, con terza rimodulazione temporale

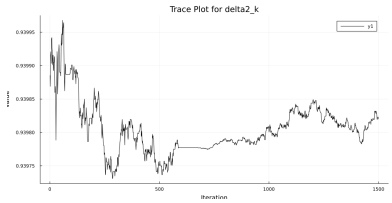
Figura 7.9: Modellizzazione del fischio-firma SW 044



(a) Trace-Plot del parametro δ per la seconda modulazione

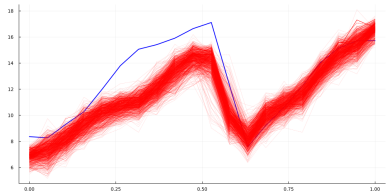


(b) Trace-Plot del parametro δ_1 per la seconda modulazione

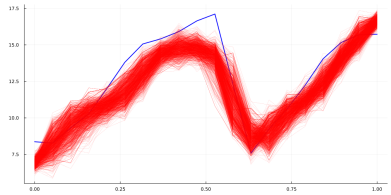


(c) Trace-Plot del parametro δ_2 per la seconda modulazione

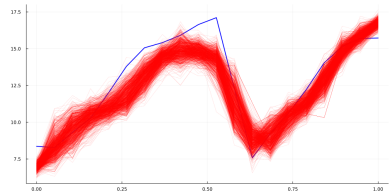
Figura 7.10: Trace-Plot parametro δ SW 044



(a) Prima rimodulazione temporale del processo latente per la realizzazione 1 di SW 044

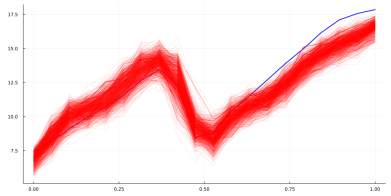


(b) Seconda rimodulazione temporale del processo latente per la realizzazione 1 di SW 044

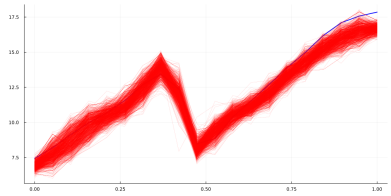


(c) Terza rimodulazione temporale del processo latente per la realizzazione 1 di SW 044

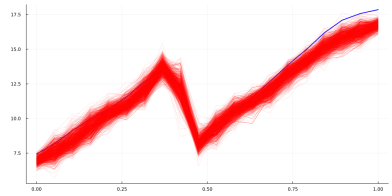
Figura 7.11: Rimodulazione temporale del processo latente per la realizzazione 1 di SW 044



(a) Prima rimodulazione temporale del processo latente per la realizzazione 6 di SW 044

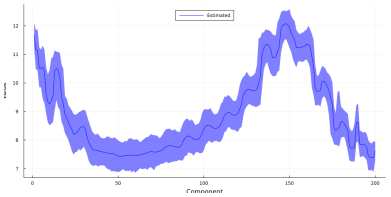


(b) Seconda rimodulazione temporale del processo latente per la realizzazione 6 di SW 044

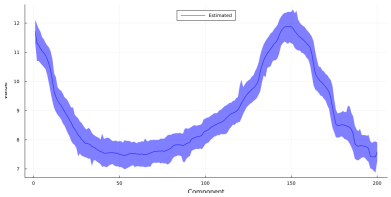


(c) Terza rimodulazione temporale del processo latente per la realizzazione 6 di SW 044

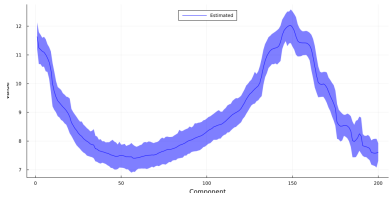
Figura 7.12: Rimodulazione temporale del processo latente per la realizzazione 6 di SW 044



(a) Modellizzazione del fischio-firma SW 056, con prima rimodulazione temporale

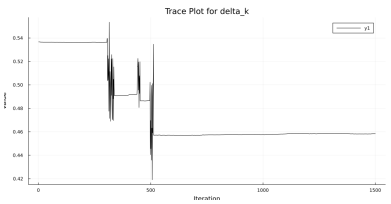


(b) Modellizzazione del fischio-firma SW 056, con seconda rimodulazione temporale

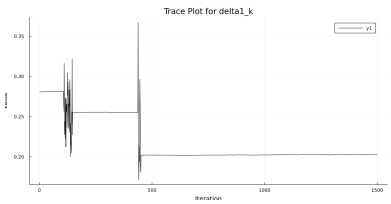


(c) Modellizzazione del fischio-firma SW 056, con terza rimodulazione temporale

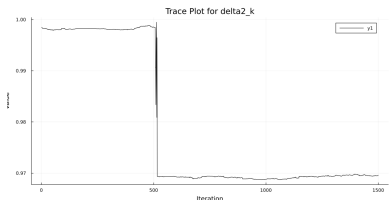
Figura 7.13: Modellizzazione del fischio-firma SW 056



(a) Trace-Plot del parametro δ per la seconda modulazione

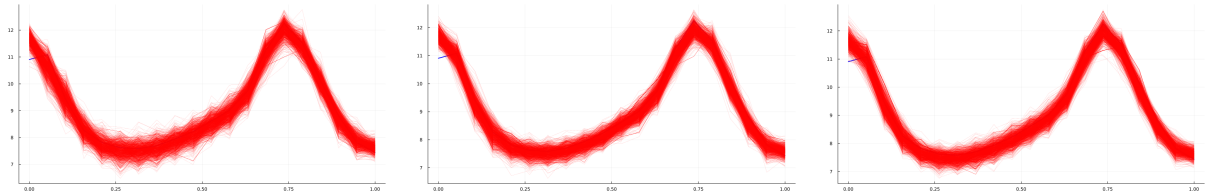


(b) Trace-Plot del parametro δ_1 per la seconda modulazione



(c) Trace-Plot del parametro δ_2 per la seconda modulazione

Figura 7.14: Trace-Plot parametro δ SW 056

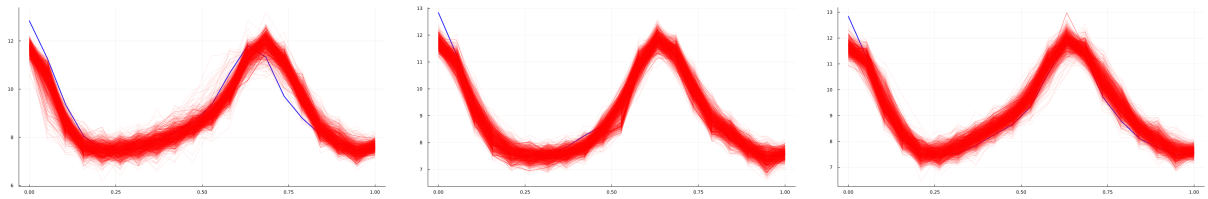


(a) Prima rimodulazione temporale del processo latente per la realizzazione 2 di SW 056

(b) Seconda rimodulazione temporale del processo latente per la realizzazione 2 di SW 056

(c) Terza rimodulazione temporale del processo latente per la realizzazione 2 di SW 056

Figura 7.15: Rimodulazione temporale del processo latente per la realizzazione 2 di SW 056

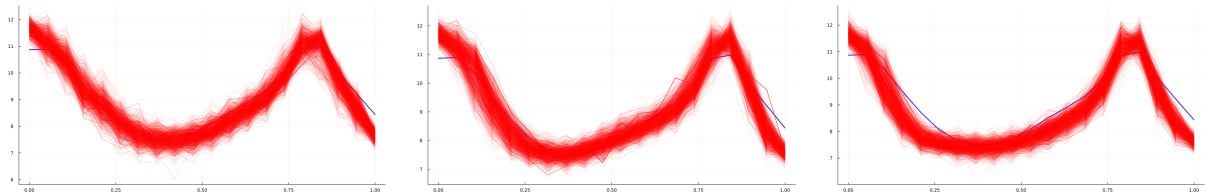


(a) Prima rimodulazione temporale del processo latente per la realizzazione 7 di SW 056

(b) Seconda rimodulazione temporale del processo latente per la realizzazione 7 di SW 056

(c) Terza rimodulazione temporale del processo latente per la realizzazione 7 di SW 056

Figura 7.16: Rimodulazione temporale del processo latente per la realizzazione 7 di SW 056

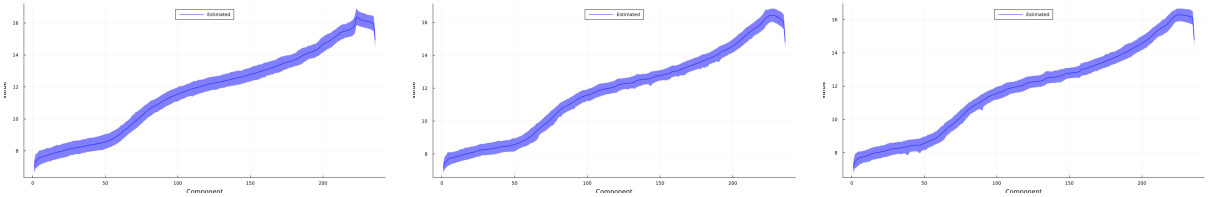


(a) Prima rimodulazione temporale del processo latente per la realizzazione 8 di SW 056

(b) Seconda rimodulazione temporale del processo latente per la realizzazione 8 di SW 056

(c) Terza rimodulazione temporale del processo latente per la realizzazione 8 di SW 056

Figura 7.17: Rimodulazione temporale del processo latente per la realizzazione 8 di SW 056

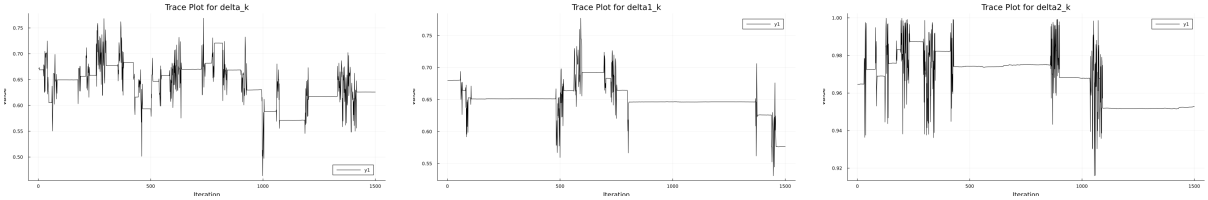


(a) Modellizzazione del fischio-firma SW 067, con prima rimodulazione temporale

(b) Modellizzazione del fischio-firma SW 067, con seconda rimodulazione temporale

(c) Modellizzazione del fischio-firma SW 067, con terza rimodulazione temporale

Figura 7.18: Modellizzazione del fischio-firma SW 067

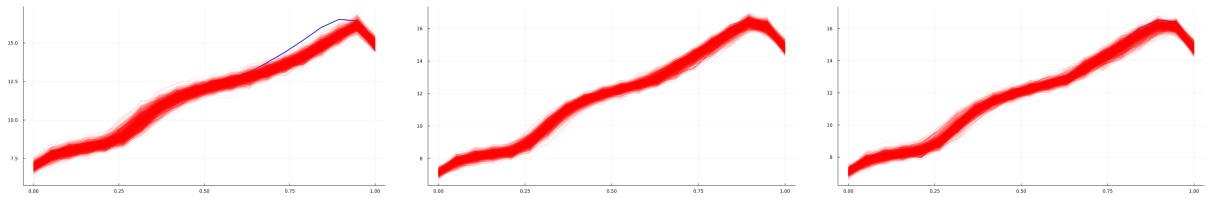


(a) Trace-Plot del parametro δ per la seconda modulazione

(b) Trace-Plot del parametro δ_1 per la seconda modulazione

(c) Trace-Plot del parametro δ_2 per la seconda modulazione

Figura 7.19: Trace-Plot parametro δ SW 067

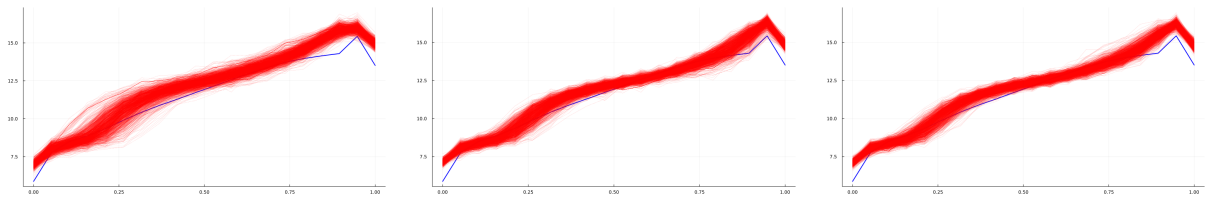


(a) Prima rimodulazione temporale del processo latente per la realizzazione 3 di SW 067

(b) Seconda rimodulazione temporale del processo latente per la realizzazione 3 di SW 067

(c) Terza rimodulazione temporale del processo latente per la realizzazione 3 di SW 067

Figura 7.20: Rimodulazione temporale del processo latente per la realizzazione 3 di SW 067

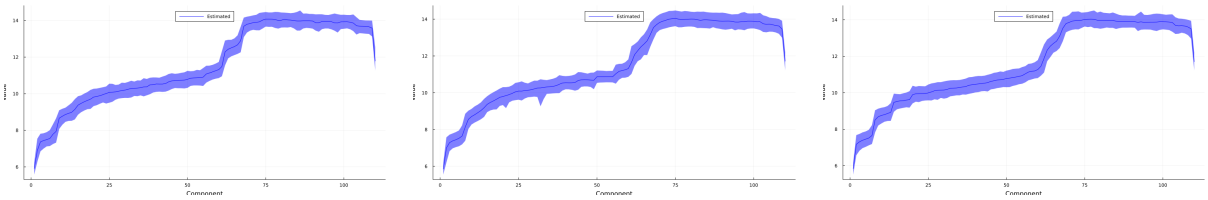


(a) Prima rimodulazione temporale del processo latente per la realizzazione 6 di SW 067

(b) Seconda rimodulazione temporale del processo latente per la realizzazione 6 di SW 067

(c) Terza rimodulazione temporale del processo latente per la realizzazione 6 di SW 067

Figura 7.21: Rimodulazione temporale del processo latente per la realizzazione 6 di SW 067

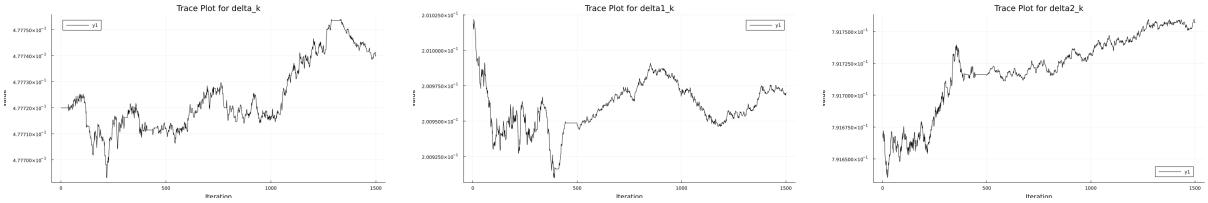


(a) Modellizzazione del fischio-firma SW 078, con prima rimodulazione temporale

(b) Modellizzazione del fischio-firma SW 078, con seconda rimodulazione temporale

(c) Modellizzazione del fischio-firma SW 078, con terza rimodulazione temporale

Figura 7.22: Modellizzazione del fischio-firma SW 078

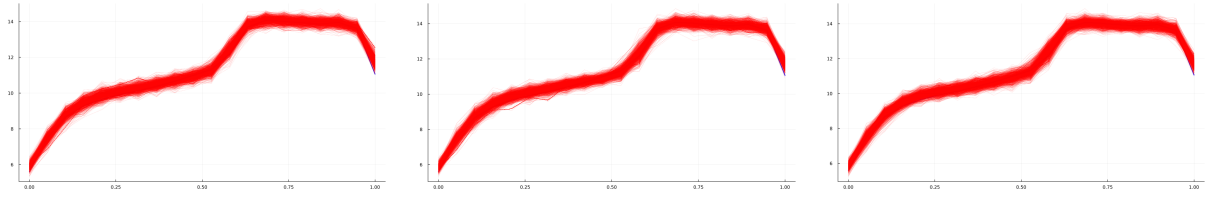


(a) Trace-Plot del parametro δ per la seconda modulazione

(b) Trace-Plot del parametro δ_1 per la seconda modulazione

(c) Trace-Plot del parametro δ_2 per la seconda modulazione

Figura 7.23: Trace-Plot parametro δ SW 078



(a) Prima rimodulazione temporale del processo latente per la realizzazione 1 di SW 078

(b) Seconda rimodulazione temporale del processo latente per la realizzazione 1 di SW 078

(c) Terza rimodulazione temporale del processo latente per la realizzazione 1 di SW 078

Figura 7.24: Rimodulazione temporale del processo latente per la realizzazione 1 di SW 078

Capitolo 8

Conclusioni

Questa tesi fornisce un'analisi introduttiva dettagliata del fenomeno naturale in esame, mettendo in evidenza l'interesse scientifico nei confronti dei delfini tursiopi e del loro apparato comunicativo. In particolare, il fischio-firma è stato identificato come elemento centrale del repertorio vocale, per poi approfondirne le caratteristiche fisiche e strutturali. L'analisi preliminare evidenzia, inoltre, il contrasto tra l'apparente semplicità della comunicazione e la complessità della struttura sociale della specie. Questo contrasto motiva un'indagine più approfondita sul fenomeno della rimodulazione, interpretandolo come una potenziale strategia volontaria, finalizzata a rendere la comunicazione più informativa e sofisticata. L'ipotesi di una modulazione volontaria dei segnali giustifica la stesura di questo lavoro, il quale modifica il modello introdotto in [23], che implicitamente considera tale rimodulazione come parte di un fenomeno rumoroso generale. Dopo l'attenta analisi sulle considerazioni degli esperti riguardo alla modulazione dei segnali, viene formalizzato il concetto di riscalamento temporale, culminando nella proposta di tre famiglie di funzioni non-lineari.

Il modello gerarchico bayesiano, incorporante tali riscalamenti, è analizzato nel dettaglio, illustrando inizialmente il ruolo di ciascuna componente per poi calcolare esplicitamente le full-conditional laddove possibile. Nella descrizione dei metodi MCMC viene approfondita sia la scelta delle proposte nei passi Metropolis, che la sequenzialità dell'algoritmo stesso. In particolare, vengono prima riepilogati due algoritmi adattivi per la varianza delle proposte e, successivamente, presentata un'implementazione specifica per il caso in esame, sfruttando i concetti precedentemente introdotti. L'algoritmo MCMC mostra sensibilità nella trattazione delle proposte nei passi Metropolis, separando in due distinte fasi di proposta le entità agenti sulla forma del processo genesi, e quelle atte alla rimodulazione temporale.

L'analisi dettagliata delle diverse strutture algoritmiche viene completata dalla loro valutazione prestazionale su dati sintetici. Un primo confronto riguarda la sequenzialità dei passi nell'algoritmo MCMC, mettendo in evidenza criticità significative nel Systematic-Scan Sampling, che mostra netta preferenza tra due ordinamenti predeterminati dei passi dell'algoritmo, aprendo dunque la possibilità a configurazioni più efficienti. A parità di inizializzazione, il Random-Scan Sampling mostra maggiore consistenza nei risultati, il che porta alla scelta di questo approccio per le fasi successive dello studio, evitando il problema della scelta ottimale della sequenzialità. Nella seconda fase del processo decisionale, si confronta la modulazione regolare con quella definita a tratti. Anche in questo caso l'esito del confronto è chiaro: la modulazione a tratti risulta prevalere nettamente sulla sua controparte regolare. Ciò evidenzia come le potenziali problematiche di non-identificabilità, unite alla difficoltà intrinseca nel garantire la condizione di monotonia, rendano la gestione del problema eccessivamente complessa per l'algoritmo. Dopo aver selezionato i modelli più affidabili, si procede con una giustificazione empirica delle modulazioni. In particolare, si mostra come l'aumento della complessità modellistica sia adeguatamente motivato dal miglioramento nella stima del processo di genesi. Come ultima analisi preliminare, si verifica la coerenza tra diversi riscalamenti, evidenziando come i modelli più complessi, in specifiche condizioni ad hoc, siano

in grado di ridursi efficacemente a sotto-casistiche.

Sulla base di questi risultati incoraggianti, i modelli selezionati vengono successivamente applicati a nuovi dati sintetici, confermando la loro efficacia. Ciò conclude la fase di validazione dei metodi MCMC proposti e conduce all'applicazione su dati reali. Nella sezione ad essi dedicata, i modelli vengono applicati a sei tipologie di fischi-firma differenti. Complessivamente, il loro comportamento risulta soddisfacente: i risultati ottenuti confermano pienamente le aspettative teoriche e, in alcuni casi, superano persino quelli derivanti dai dati sintetici.

In tutte le istanze analizzate, sia quelle riportate direttamente nella tesi che quelle disponibili su GitHub, il metodo di stima non evidenzia criticità significative, il ché ne conferma l'affidabilità. Un'analisi più approfondita rivela una prevalenza prestazionale delle modulazioni più complesse rispetto alla più semplice, in particolare nella stima del processo generatore. Questo aspetto è evidenziato nelle figure 7.5, 7.9 e 7.13. In fase di applicazione sulle realizzazioni, tuttavia, non emergono differenze prestazionali significative tra le tre modulazioni, rendendo dunque la scelta del riscalamento più adeguato dipendente dalla specifica indagine da condurre. Il riscalamento più semplice si dimostra affidabile, ma può generare stime meno regolari del processo di genesi, pur avendo il vantaggio di richiedere circa la metà dei coefficienti rispetto alle altre due famiglie. Tra queste, la terza modulazione, caratterizzata dai parametri δ_1 e δ_2, offre prestazioni leggermente superiori, ed è facilmente riconducibile al caso con un solo δ. La terza configurazione risulta, quindi, particolarmente adatta per le possibili applicazioni nel riconoscimento dei fischi-firma e nel raggruppamento per individui emittenti.

La modellizzazione gerarchica bayesiana, a tempo variabile, risulta ben giustificata a livello teorico ed empirico. Le ottime stime confermano l'affidabilità del metodo, validandolo come strumento per l'analisi avanzata dei fischi-firma e concludendo, dunque, la tesi.

Appendice A

Implementazione del Codice

A.1 Julia

L'informatica scientifica ha tradizionalmente richiesto le massime prestazioni, ma gli esperti del settore si sono ampiamente spostati verso linguaggi dinamici più lenti per il lavoro quotidiano. La progettazione moderna dei linguaggi e le attuali tecniche di compilazione consentono di eliminare in gran parte il compromesso sulle prestazioni e di fornire un singolo ambiente sufficientemente produttivo per la prototipazione e sufficientemente efficiente per l'implementazione di applicazioni ad alta intensità di prestazioni. Il linguaggio di programmazione **Julia** ricopre questo ruolo. Si tratta di un linguaggio dinamico flessibile, appropriato per l'informatica scientifica e numerica, con prestazioni paragonabili ai tradizionali linguaggi staticamente tipizzati. Julia offre tipizzazione opzionale, dispatch multiplo e buone prestazioni, ottenute tramite inferenza di tipo e compilazione just-in-time (JIT) (e compilazione ahead-of-time opzionale), implementate tramite LLVM. È multi-paradigma, combinando funzionalità di programmazione imperativa, funzionale e orientata agli oggetti. Julia offre semplicità ed espressività per l'elaborazione numerica di alto livello, allo stesso modo di linguaggi come R, MATLAB e Python, ma supporta anche la programmazione generale [52].

A.2 Il codice

In questo paragrafo segue l'implementazione delle principali componenti dell'algoritmo MCMC che ha fornito i risultati discussi. Per la natura del problema affrontato diverse sono le implementazioni, come diverse erano le modulazioni, perciò, per evitare ripetizioni di codice e alleggerire il contenuto, verrà riportata l'implementazione di un solo scenario di modulazione. Insieme alle componenti principali dell'algoritmo MCMC saranno riportate anche le implementazioni per la creazione dei dati sintetici, l'impostazione dello stato iniziale della catena ed alcune funzioni utilizzate nei passaggi precedenti, che sfruttano i concetti di ottimizzazione per ridurre lo sforzo computazionale. Il codice completo è reperibile su GitHub.

Implementazione delle Full-Conditional

Listing A.1: Full-Conditional $\boldsymbol{u}$

```
### STEP 1 ###
function step1_u(;WT_k::Vector{Float64}, indici::Matrix{Int64},
    Dati::Matrix{Float64}, tau2_k::Vector{Float64}, sigma2_u::Float64,
    n_time::Int64, n_instances::Int64)::Vector{Float64}
    # Dichiarazione variabili step 1:
```

```
    w_k_i=Vector{Float64}(undef, n_time)  # Anche per step 2
    u_ik=Vector{Float64}(undef, n_instances)
    MU_u::Float64=0.0
    SIGMA2_u::Float64=0.0
    h_u::Float64=0.0

    ### STEP 1) MCMC su u^k_i: dalla full-conditional
    u_ik.=fill(0.0, n_instances)
    for i in 1:n_instances
        w_k_i.=WT_k[indici[i, :]]
        MU_u=0.0
        SIGMA2_u=0.0
        MU_u = sum(Dati[i, :] .- w_k_i[:])/tau2_k[i]  # Somma stabile
        h_u = (n_time*sigma2_u + tau2_k[i])/(tau2_k[i]*sigma2_u)
        MU_u /= h_u
        SIGMA2_u = 1/h_u
        u_ik[i] = rand(Normal(MU_u, sqrt(SIGMA2_u)))
    end
    return u_ik
end
```

Listing A.2: Full-Conditional τ^2

```
### STEP 2 ###
function step2_tau2(;WT_k::Vector{Float64}, indici::Matrix{Int64},
    par_inv_gamma_a::Float64, par_inv_gamma_b::Float64,
    Dati::Matrix{Float64}, u_k::Vector{Float64}, n_time::Int64,
    n_instances::Int64)::Vector{Float64}
    # Dichiarazione delle variabili:
    tau2_ik=Vector{Float64}(undef, n_instances)
    new_par_a_tau::Float64=0.0
    new_par_b_tau::Float64=0.0

    ### STEP 2) MCMC su \tau2^k_i: dalla full-conditional
    for i in 1:n_instances
        w_k_i = WT_k[indici[i, :]]
        new_par_a_tau = n_time / 2 + par_inv_gamma_a
        new_par_b_tau = (Dati[i, :].-(u_k[i].+w_k_i))'*(Dati[i,
    :].-(u_k[i].+w_k_i))/2 + par_inv_gamma_b

        tau2_ik[i] = rand(InverseGamma(new_par_a_tau, new_par_b_tau))
    # Genera valore dalla distribuzione Inverse Gamma
    end
    return tau2_ik
end
```

Listing A.3: Full-Conditional β

```
    ### STEP 3 ###
function step3_beta(;n_time_trasformati::Int64, sigma2_beta::Float64,
    SIGMA_k::Symmetric{Float64, Matrix{Float64}},
    X_k::Matrix{Float64}, WT_k::Vector{Float64},
    mu_beta::Vector{Float64})::Vector{Float64}
    # Dichiarazione delle variabili:
```

```
    beta_k=Vector{Float64}(undef, 2)
    inv_SIGMA2_beta=Diagonal(Vector{Float64}(undef, 2))
    L_SIGMA_k=LowerTriangular(Matrix{Float64}(undef,
    n_time_trasformati, n_time_trasformati))
    prod_invL_X=Matrix{Float64}(undef, n_time_trasformati, 2)
    inv_V_beta=Symmetric(Matrix{Float64}(undef, 2, 2))
    prod_invL_WT=Vector{Float64}(undef, n_time_trasformati)
    M_beta=Vector{Float64}(undef, 2)

    ### STEP 3) MCMC su \beta^k: dalla full-conditional
    inv_SIGMA2_beta.diag .= fill(1 / sigma2_beta, 2)
    L_SIGMA_k.= cholesky(SIGMA_k).L
    prod_invL_X.= LowerTriangular(L_SIGMA_k) \ X_k
    copyto!(parent(inv_V_beta),Symmetric(prod_invL_X' * prod_invL_X +
    inv_SIGMA2_beta))
    prod_invL_WT.= LowerTriangular(L_SIGMA_k) \ WT_k
    M_beta.= inv_V_beta \ (prod_invL_X' * prod_invL_WT +
    inv_SIGMA2_beta * mu_beta)
    beta_k.= rmvnorm_nuova(1; mean=vec(M_beta), invSigma=inv_V_beta)

    return beta_k
end
```

Listing A.4: Full-Conditional σ_u^2

```
    ### STEP 4 ###
function step4_sigma2_u(;n_instances::Int64, par_inv_gamma_a::Float64,
    par_inv_gamma_b::Float64, u_k::Vector{Float64})::Float64
    # Dichiarazione variabili step 4:
    new_par_a_sigma2_u::Float64=0.0
    new_par_b_sigma2_u::Float64=0.0
    sigma2_u::Float64=0.0

    ### STEP 4) MCMC su \sigma2_u: dalla full-conditional
    new_par_a_sigma2_u = n_instances / 2 + par_inv_gamma_a
    new_par_b_sigma2_u = par_inv_gamma_b + (u_k'*u_k) / 2
    sigma2_u = rand(InverseGamma(new_par_a_sigma2_u,
    new_par_b_sigma2_u))

    return sigma2_u
end
```

Listing A.5: Full-Conditional $\sigma_b^{2,k}$

```
    ### STEP 5 ###
function step5_sigma2_k_b(;n_instances::Int64,
    par_inv_gamma_a::Float64, par_inv_gamma_b::Float64,
    b_k::Vector{Float64})::Float64

    # Dichiarazione variabili step 5:
    new_par_a_sigma2_k_b::Float64=0.0
    new_par_b_sigma2_k_b::Float64=0.0

    ### STEP 5) MCMC su \sigma2_k_b: dalla full-conditional
```

```
    new_par_a_sigma2_k_b = n_instances/2 + par_inv_gamma_a
    new_par_b_sigma2_k_b = par_inv_gamma_b + (b_k'*b_k) / 2
    sigma2_k_b = rand(InverseGamma(new_par_a_sigma2_k_b,
    new_par_b_sigma2_k_b))

    return sigma2_k_b
end
```

Listing A.6: Full-Conditional processo W_k^T

```
function FC_WT(;n_time_trasformati::Int64, T_k::Vector{Float64},
    alpha_k::Float64, psi2_k::Float64, beta_k::Vector{Float64},
    tau2_k::Vector{Float64}, indici::Matrix{Int64}, n_time::Int64,
    n_instances::Int64, u_k::Vector{Float64},
    Dati::Matrix{Float64})::Tuple{Vector{Float64}, Vector{Float64},
    Symmetric{Float64}, UpperTriangular{Float64}, Vector{Float64},
    Symmetric{Float64}, UpperTriangular{Float64}, Matrix{Float64}}

    # Dichiarazione:
    X_k=Matrix{Float64}(undef, n_time_trasformati, 2)
    SIGMA_k=Symmetric(Matrix{Float64}(undef, n_time_trasformati,
    n_time_trasformati))
    X_k_times_beta_k=Vector{Float64}(undef, n_time_trasformati)
    U_SIGMA_k=UpperTriangular(Matrix{Float64}(undef,
    n_time_trasformati, n_time_trasformati))
    SIGMA_k_inv=Symmetric(Matrix{Float64}(undef, n_time_trasformati,
    n_time_trasformati))
    V_inv=Symmetric(Matrix{Float64}(undef, n_time_trasformati,
    n_time_trasformati))
    U_V_inv=UpperTriangular(Matrix{Float64}(undef, n_time_trasformati,
    n_time_trasformati))
    vettore=Vector{Float64}(undef, n_time_trasformati)
    L_V_inv=LowerTriangular(Matrix{Float64}(undef, n_time_trasformati,
    n_time_trasformati))
    M=Vector{Float64}(undef, n_time_trasformati)
    WT_k=Vector{Float64}(undef, n_time_trasformati)

    # Full-Conditional WT_k
    X_k.=hcat(ones(n_time_trasformati), T_k)

    copyto!(parent(SIGMA_k), Symmetric(creare_varcov_OU(T_k=T_k,
    alpha=alpha_k, psi2=psi2_k)))
    X_k_times_beta_k.=X_k*beta_k

    U_SIGMA_k.=UpperTriangular(cholesky(SIGMA_k).U)
    copyto!(parent(SIGMA_k_inv), Symmetric(chol2inv(U_SIGMA_k)))

    copyto!(parent(V_inv),
    Symmetric(somma_matrice_diagonale(B=SIGMA_k_inv, v=tau2_k,
    indici_matrix=indici, n_time=n_time, n_instances=n_instances)))
    U_V_inv.=UpperTriangular(cholesky(V_inv).U)

```

```
    vettore.=ottenere_vettore(n_time_transformed=n_time_trasformati,
    n_instances=n_instances, n_time=n_time, indici=indici, Y=Dati,
    u_k=u_k, tau2_k=tau2_k)
    vettore .+= SIGMA_k_inv*X_k_times_beta_k

    L_V_inv.=LowerTriangular(transpose(U_V_inv))
    M.=UpperTriangular(U_V_inv) \ (L_V_inv \ vettore)

    # Il processo WT_k_prop:
    WT_k.=rmvnorm_nuova(1; mean=M, invSigma=V_inv, U_invSigma=U_V_inv)

    return WT_k, X_k_times_beta_k, SIGMA_k, U_SIGMA_k, M, V_inv,
    U_V_inv, X_k
end
```

Implementazione dei rapporti Metropolis

Listing A.7: Rapporto metropolis per $\alpha\ \psi^2\ W_k^T$.

```
    function Metropolis_Ratio_alpha_psi_WT(;n_instances::Int64,
    n_time_trasformati::Int64, WT_kprop::Vector{Float64},
    indici_prop::Matrix{Int64}, tau2_kMCMC::Vector{Float64},
    n_time::Int64, Dati::Matrix{Float64}, u_kMCMC::Vector{Float64},
    SIGMA_kprop::Symmetric{Float64},
    U_SIGMA_kprop::UpperTriangular{Float64}, alpha_kprop::Float64,
    par_inv_gamma_a::Float64, par_inv_gamma_b::Float64,
    psi2_kprop::Float64, WT_kMCMC::Vector{Float64},
    M_MCMC::Vector{Float64}, V_inv_MCMC::Symmetric{Float64},
    U_V_inv_MCMC::UpperTriangular{Float64},
    indici_MCMC::Matrix{Int64}, X_kMCMC::Matrix{Float64},
    beta_kMCMC::Vector{Float64}, SIGMA_kMCMC::Symmetric{Float64},
    U_SIGMA_kMCMC::UpperTriangular{Float64}, alpha_kMCMC::Float64,
    psi2_kMCMC::Float64, M_prop::Vector{Float64},
    V_inv_prop::Symmetric{Float64},
    U_V_inv_prop::UpperTriangular{Float64})::Float64

    # Dichiarazione
    logRatioMetropolis::Float64=0.0
    numeratore::Float64=0.0
    vett_num=Vector{Float64}(undef, 5)
    w_k_i_prop=Vector{Float64}(undef, n_time)
    TAU2_k_i=Symmetric(Matrix{Float64}(undef, n_time, n_time))
    delta2_numeratore::Float64=0.0
    delta3_numeratore::Float64=0.0
    delta4_numeratore::Float64=0.0
    num_q_w::Float64=0.0
    denominatore::Float64=0.0
    vett_denom=Vector{Float64}(undef, 5)
    w_k_i_MCMC=Vector{Float64}(undef, n_time)
    X_kMCMC_times_beta_kMCMC::Vector{Float64}=X_kMCMC*beta_kMCMC
    delta2_denominatore::Float64=0.0
    delta3_denominatore::Float64=0.0
    delta4_denominatore::Float64=0.0
```

```
20      denom_q_w::Float64=0.0
21
22      ### Il rapporto Metropolis:
23      logRatioMetropolis=0.0
24
25      ### Numeratore:
26      numeratore=0.0
27      vett_num.=zeros(5)
28
29      # Densita' delle Y_i sul nuovo processo WT_k proposto:
30      for i in 1:n_instances
31          w_k_i_prop.=WT_kprop[indici_prop[i,:]]
32          copyto!(parent(TAU2_k_i),
        Symmetric(Diagonal(fill(tau2_kMCMC[i], n_time))))
33          numeratore += dmvnorm_nuova(Dati[i, :]; mean=w_k_i_prop .+
        u_kMCMC[i], Sigma=TAU2_k_i, logflag=true)  # function
        dmvnorm_nuova(x; mean=nothing, Sigma=nothing, invSigma=nothing,
        U_Sigma=nothing, U_invSigma=nothing, logflag=false)
34      end
35      vett_num[1] = numeratore
36
37      # Densita' del nuovo processo WT_kprop:
38      delta2_numeratore = dmvnorm_nuova(WT_kprop;
        mean=X_kMCMC_times_beta_kMCMC, Sigma=SIGMA_kprop,
        U_Sigma=U_SIGMA_kprop, logflag=true)
39      numeratore += delta2_numeratore
40      vett_num[2] = delta2_numeratore
41
42      # Densita' della nuova alpha_kprop:
43      delta3_numeratore = logpdf(Uniform(0, 1), alpha_kprop)
44      numeratore += delta3_numeratore
45      vett_num[3] = delta3_numeratore
46
47      # Densita' della nuova psi2_kprop:
48      delta4_numeratore = logpdf(InverseGamma(par_inv_gamma_a,
        par_inv_gamma_b), psi2_kprop)
49      numeratore += delta4_numeratore
50      vett_num[4] = delta4_numeratore
51
52      # Densita' delle proposte: (non considero quelle simmetriche)
53      num_q_w = dmvnorm_nuova(WT_kMCMC; mean=M_MCMC,
        invSigma=V_inv_MCMC, U_invSigma=U_V_inv_MCMC, logflag=true)
54      numeratore += num_q_w
55      vett_num[5] = num_q_w
56
57      ### Denominatore:
58      denominatore = 0.0
59      vett_denom.=zeros(5)
60
61      # Densita' delle Y_i sul nuovo processo WT_k MCMC:
62      for i in 1:n_instances
63          w_k_i_MCMC.=WT_kMCMC[indici_MCMC[i,:]]
```

```
64          copyto!(parent(TAU2_k_i),
    Symmetric(Diagonal(fill(tau2_kMCMC[i], n_time))))
65          denominatore += dmvnorm_nuova(Dati[i, :]; mean=w_k_i_MCMC .+
    u_kMCMC[i], Sigma=TAU2_k_i, logflag=true)
66      end
67      vett_denom[1] = denominatore
68
69      # Densita' del nuovo processo WT_kMCMC:
70      delta2_denominatore = dmvnorm_nuova(WT_kMCMC,
    mean=X_kMCMC_times_beta_kMCMC, Sigma=SIGMA_kMCMC,
    U_Sigma=U_SIGMA_kMCMC, logflag=true)
71      denominatore += delta2_denominatore
72      vett_denom[2] = delta2_denominatore
73
74      # Densita' della nuova alpha_kMCMC:
75      delta3_denominatore = logpdf(Uniform(0, 1), alpha_kMCMC)
76      denominatore += delta3_denominatore
77      vett_denom[3] = delta3_denominatore
78
79      # Densita' della nuova psi2_kMCMC:
80      delta4_denominatore = logpdf(InverseGamma(par_inv_gamma_a,
    par_inv_gamma_b), psi2_kMCMC)
81      denominatore += delta4_denominatore
82      vett_denom[4] = delta4_denominatore
83
84      # Densita' delle proposte: (non considero quelle simmetriche)
85      denom_q_w = dmvnorm_nuova(WT_kprop; mean=M_prop,
    invSigma=V_inv_prop, U_invSigma=U_V_inv_prop, logflag=true)
86      denominatore += denom_q_w
87      vett_denom[5] = denom_q_w
88
89      # Stampa confronto
90      println("confronto numeratore e denominatore")
91
92      # Stampa parallela
93      for i in 1:5
94          println("Elemento $i: $(vett_num[i]) | $(vett_denom[i])")
95      end
96
97      ### Il rapporto Metropolis:
98      logRatioMetropolis = numeratore - denominatore
99
100     ### Campione di uniforme per accettare o rifiutare la proposta:
101     RatioMetropolis = min(1, exp(logRatioMetropolis))
102
103     return RatioMetropolis
104 end
```

Listing A.8: Rapporto metropolis per b_k W_k^T.

```
1
```

```
function Metropolis_Ratio_b_WT(;n_instances::Int64,
    n_time_trasformati::Int64, WT_kprop::Vector{Float64},
    indici_prop::Matrix{Int64}, tau2_kMCMC::Vector{Float64},
    n_time::Int64, Dati::Matrix{Float64}, u_kMCMC::Vector{Float64},
    X_kprop_times_beta_kMCMC::Vector{Float64},
    SIGMA_kprop::Symmetric{Float64},
    U_SIGMA_kprop::UpperTriangular{Float64}, b_kprop::Vector{Float64},
    WT_kMCMC::Vector{Float64}, M_MCMC::Vector{Float64},
    V_inv_MCMC::Symmetric{Float64},
    U_V_inv_MCMC::UpperTriangular{Float64},
    indici_MCMC::Matrix{Int64}, X_kMCMC::Matrix{Float64},
    beta_kMCMC::Vector{Float64}, SIGMA_kMCMC::Symmetric{Float64},
    U_SIGMA_kMCMC::UpperTriangular{Float64}, sigma2_k_bMCMC::Float64,
    b_kMCMC::Vector{Float64}, M_prop::Vector{Float64},
    V_inv_prop::Symmetric{Float64},
    U_V_inv_prop::UpperTriangular{Float64})::Float64

    # Dichiarazione
    logRatioMetropolis::Float64=0.0
    numeratore::Float64=0.0
    vett_num=Vector{Float64}(undef, 4)
    w_k_i_prop=Vector{Float64}(undef, n_time)
    TAU2_k_i=Symmetric(Matrix{Float64}(undef, n_time, n_time))
    delta2_numeratore::Float64=0.0
    delta5_numeratore::Float64=0.0
    num_q_w::Float64=0.0
    denominatore::Float64=0.0
    vett_denom=Vector{Float64}(undef, 4)
    w_k_i_MCMC=Vector{Float64}(undef, n_time)
    X_kMCMC_times_beta_kMCMC=Vector{Float64}(undef, n_time_trasformati)
    delta2_denominatore::Float64=0.0
    delta5_denominatore::Float64=0.0
    denom_q_w::Float64=0.0

    ### Il rapporto Metropolis:
    logRatioMetropolis=0.0

    ### Numeratore:
    numeratore=0.0
    vett_num.=zeros(4)

    # Densita' delle Y_i sul nuovo processo WT_k proposto:
    for i in 1:n_instances
        w_k_i_prop.=WT_kprop[indici_prop[i,:]]
        copyto!(parent(TAU2_k_i),
    Symmetric(Diagonal(fill(tau2_kMCMC[i], n_time))))
        numeratore += dmvnorm_nuova(Dati[i, :]; mean=w_k_i_prop .+
    u_kMCMC[i], Sigma=TAU2_k_i, logflag=true)  # function
    dmvnorm_nuova(x; mean=nothing, Sigma=nothing, invSigma=nothing,
    U_Sigma=nothing, U_invSigma=nothing, logflag=false)
    end
    vett_num[1] = numeratore

```

```
36      # Densita' del nuovo processo WT_kprop:
37      delta2_numeratore = dmvnorm_nuova(WT_kprop;
        mean=X_kprop_times_beta_kMCMC, Sigma=SIGMA_kprop,
        U_Sigma=U_SIGMA_kprop, logflag=true)
38      numeratore += delta2_numeratore
39      vett_num[2] = delta2_numeratore
40
41      # Densita' dei nuovi b_kprop:
42      delta5_numeratore=dmvnorm_nuova(b_kprop; mean=fill(0.0,
        n_instances), Sigma=Symmetric(Diagonal(fill(sigma2_k_bMCMC,
        n_instances))), logflag=true)
43      numeratore += delta5_numeratore
44      vett_num[3] = delta5_numeratore
45
46      # Densita' delle proposte: (non considero quelle simmetriche)
47      num_q_w = dmvnorm_nuova(WT_kMCMC; mean=M_MCMC,
        invSigma=V_inv_MCMC, U_invSigma=U_V_inv_MCMC, logflag=true)
48      numeratore += num_q_w
49      vett_num[4] = num_q_w
50
51      ### Denominatore:
52      denominatore = 0.0
53      vett_denom.=zeros(4)
54
55      # Densita' delle Y_i sul nuovo processo WT_k MCMC:
56      for i in 1:n_instances
57          w_k_i_MCMC.=WT_kMCMC[indici_MCMC[i,:]]
58          copyto!(parent(TAU2_k_i),
        Symmetric(Diagonal(fill(tau2_kMCMC[i], n_time))))
59          denominatore += dmvnorm_nuova(Dati[i, :]; mean=w_k_i_MCMC .+
        u_kMCMC[i], Sigma=TAU2_k_i, logflag=true)  # function
        dmvnorm_nuova(x; mean=nothing, Sigma=nothing, invSigma=nothing,
        U_Sigma=nothing, U_invSigma=nothing, logflag=false)
60      end
61      vett_denom[1] = denominatore
62
63      X_kMCMC_times_beta_kMCMC.=X_kMCMC*beta_kMCMC
64      # Densita' del nuovo processo WT_kMCMC:
65      delta2_denominatore = dmvnorm_nuova(WT_kMCMC,
        mean=X_kMCMC_times_beta_kMCMC, Sigma=SIGMA_kMCMC,
        U_Sigma=U_SIGMA_kMCMC, logflag=true)
66      denominatore += delta2_denominatore
67      vett_denom[2] = delta2_denominatore
68
69      # Densita' dei nuovi b_kMCMC:
70      delta5_denominatore=dmvnorm_nuova(b_kMCMC; mean=fill(0.0,
        n_instances), Sigma=Symmetric(Diagonal(fill(sigma2_k_bMCMC,
        n_instances))), logflag=true)
71      denominatore += delta5_denominatore
72      vett_denom[3] = delta5_denominatore
73
74      # Densita' delle proposte: (non considero quelle simmetriche)
```

```
    denom_q_w = dmvnorm_nuova(WT_kprop; mean=M_prop,
    invSigma=V_inv_prop, U_invSigma=U_V_inv_prop, logflag=true)
    denominatore += denom_q_w
    vett_denom[4] = denom_q_w

    # Stampa confronto
    println("confronto numeratore e denominatore")

    # Stampa parallela
    for i in 1:4
        println("Elemento $i: $(vett_num[i]) | $(vett_denom[i])")
    end

    ### Il rapporto Metropolis:
    logRatioMetropolis = numeratore - denominatore

    ### Campione di uniforme per accettare o rifiutare la proposta:
    RatioMetropolis = min(1, exp(logRatioMetropolis))

    return RatioMetropolis
end
```

Algoritmo MCMC completo

```
### ALGORITMO MCMC ###

function MCMC_STEP_1_2_3_4_5_6(;alpha_k::Union{Float64,
    Nothing}=nothing, psi2_k::Union{Float64, Nothing}=nothing,
    b_k::Union{Vector{Float64}, Nothing}=nothing,
    beta_k::Union{Vector{Float64}, Nothing}=nothing,
    u_k::Union{Vector{Float64}, Nothing}=nothing,
    tau2_k::Union{Vector{Float64}, Nothing}=nothing,
    sigma2_u::Union{Float64, Nothing}=nothing,
    sigma2_k_b::Union{Float64, Nothing}=nothing,
    par_inv_gamma_a::Union{Float64, Nothing}=nothing,
    par_inv_gamma_b::Union{Float64, Nothing}=nothing,
    mu_beta::Union{Vector{Float64}, Nothing}=nothing,
    sigma2_beta::Union{Float64, Nothing}=nothing,
    Dati::Union{Matrix{Float64}, Nothing}=nothing, iter::Union{Int64,
    Nothing}=nothing, burnin::Union{Int64, Nothing}=nothing,
    thin::Union{Int64, Nothing}=nothing, WT_k::Union{Vector{Float64},
    Nothing}=nothing, T_k::Union{Vector{Float64}, Nothing}=nothing,
    X_k::Union{Matrix{Float64}, Nothing}=nothing,
    SIGMA_k::Union{Symmetric{Float64}, Nothing}=nothing,
    U_SIGMA_k::Union{UpperTriangular{Float64}, Nothing}=nothing,
    M::Union{Vector{Float64}, Nothing}=nothing,
    V_inv::Union{Symmetric{Float64}, Nothing}=nothing,
    U_V_inv::Union{UpperTriangular{Float64}, Nothing}=nothing,
    indici::Union{Matrix{Int64}, Nothing}=nothing,
    size_batch::Int64=1, n_indici::Int64=1)
n_instances::Int64, n_time::Int64 = size(Dati)
```

```
time_points::Vector{Float64} = LinRange(0, 1, n_time)
n_time_trasformati::Int64=(n_time + (n_time - 2) * (n_instances - 1))

# Dichiarazione:

### Numero di campioni a posteriori che devo salvare
NsampleSave::Int64    = Int(floor((iter-burnin)/thin))

### Oggetti che ritorna la funzione
alpha_kSave::Vector{Float64}     = zeros(NsampleSave)
psi2_kSave::Vector{Float64}      = zeros(NsampleSave)
b_kSave::Matrix{Float64}         = zeros(NsampleSave, n_instances)
beta_kSave::Matrix{Float64}      = zeros(NsampleSave, 2)
u_kSave::Matrix{Float64}         = zeros(NsampleSave, n_instances)  # Ogni osservazione ha la sua u
tau2_kSave::Matrix{Float64}      = zeros(NsampleSave, n_instances)  # Ogni osservazione ha la sua varianza
sigma2_uSave::Vector{Float64}    = zeros(NsampleSave) # Un'unica varianza per tutti gli u
sigma2_k_bSave::Vector{Float64} = zeros(NsampleSave)  # Ogni gruppo ha la propria varianza sulla modulazione
WT_kSave::Matrix{Float64}        = zeros(NsampleSave, (n_time + (n_time - 2) * (n_instances - 1)))
IndiciSave::Array{Int64, 3}    = zeros(NsampleSave, n_instances, n_time)

### definisco i valori correnti.
alpha_kMCMC       = copy(alpha_k)
psi2_kMCMC        = copy(psi2_k)
b_kMCMC           = copy(b_k)
beta_kMCMC        = copy(beta_k)
u_kMCMC           = copy(u_k)
tau2_kMCMC        = copy(tau2_k)
sigma2_uMCMC      = copy(sigma2_u)
sigma2_k_bMCMC    = copy(sigma2_k_b)

indici_MCMC       = copy(indici)
WT_kMCMC          = copy(WT_k)
T_kMCMC           = copy(T_k)
X_kMCMC           = copy(X_k)
SIGMA_kMCMC       = copy(SIGMA_k)
U_SIGMA_kMCMC     = copy(U_SIGMA_k)
V_inv_MCMC        = copy(V_inv)
U_V_inv_MCMC      = copy(U_V_inv)
M_MCMC            = copy(M)


# Dichiarazione variabili step 6:
A::Int64=2000
B::Int64=4000
epsilon::Float64=1e-13
gamma_i=A/B

# alpha e psi
```

```
alpha_batch_1::Float64=0.0
alpha_star_1::Float64=0.234
log_lambda_i_1::Float64=0.0
Prop_i_1=Vector{Float64}(undef, n_instances)
X_i_1::Vector{Float64}=[alpha_kMCMC, psi2_kMCMC]
mu_i_1::Vector{Float64}=zeros(Float64, 2)
iterazione_aggiornamento_1::Int64=0
Sigma_i1_ = Matrix{Float64}(undef, 2, 2)
fill!(Sigma_i1_, 0.0)
Sigma_i1_[diagind(Sigma_i1_)] .= 1.0
Sigma_i_1 = Symmetric(Sigma_i1_)
skippa_1::Bool=false

# b
alpha_batch_2::Float64=0.0
alpha_star_2::Float64=0.234
log_lambda_i_2::Float64=0.0
Prop_i_2=Vector{Float64}(undef, n_instances)
X_i_2::Vector{Float64}=copy(b_kMCMC)
mu_i_2::Vector{Float64}=zeros(Float64, n_instances)
iterazione_aggiornamento_2::Int64=0
Sigma_i2_ = Matrix{Float64}(undef, n_instances, n_instances)
fill!(Sigma_i2_, 0.0)
Sigma_i2_[diagind(Sigma_i2_)] .= 1.0
Sigma_i_2 = Symmetric(Sigma_i2_)
skippa_2::Bool=false

alpha_kprop::Float64=0.0
psi2_kprop::Float64=0.0
b_kprop=Vector{Float64}(undef, n_instances)
T_kprop=Vector{Float64}(undef, n_time_trasformati)
indici_prop=Matrix{Int64}(undef, n_instances, n_time)

n_time_trasformatiprop::Int64=0

WT_kprop=Vector{Float64}(undef, n_time_trasformati)
X_kprop_times_beta_kMCMC=Vector{Float64}(undef, n_time_trasformati)
X_kMCMC_times_beta_kMCMC=Vector{Float64}(undef, n_time_trasformati)
SIGMA_kprop=Symmetric(Matrix{Float64}(undef, n_time_trasformati,
    n_time_trasformati))
U_SIGMA_kprop=UpperTriangular(Matrix{Float64}(undef,
    n_time_trasformati, n_time_trasformati))
M_prop=Vector{Float64}(undef, n_time_trasformati)
V_inv_prop=Symmetric(Matrix{Float64}(undef, n_time_trasformati,
    n_time_trasformati))
U_V_inv_prop=UpperTriangular(Matrix{Float64}(undef,
    n_time_trasformati, n_time_trasformati))
X_kprop=Matrix{Float64}(undef, n_time_trasformati, 2)

RatioMetropolis_alpha_psi2_WT::Float64=0.0
RatioMetropolis_b_WT::Float64=0.0
u::Float64=0.0

```

```
appSamp::Int64 = burnin

iterazione::Int64=0
num_accettati_1::Int64=0
num_accettati_2::Int64=0
iterazioni_saltate_per_PosDef_or_BoundsError::Int64=0
iterazioni_saltate_per_alpha::Int64=0
iterazioni_saltate_per_psi2::Int64=0
iterazioni_saltate_per_tempi::Int64=0
ordine::Vector{Int64}=[4, 2, 3, 5, 6, 7, 1]
for iMCMC in 1:NsampleSave
for jMCMC in 1:appSamp
iterazione+=1
println("iterazione: ", iterazione)
for step in ordine
    if step == 1
        ### STEP 1) MCMC su u^k_i: dalla full-conditional
        u_kMCMC .= step1_u(WT_k=WT_kMCMC, indici=indici_MCMC, Dati=Dati, tau2_k=tau2_kMCMC, sigma2_u=sigma2_uMCMC, n_time=n_time, n_instances=n_instances)
        u_kSave[iMCMC, :] .= u_kMCMC # Memorizza i valori salvati in u_kSave
    end

    if step == 2

        ### STEP 2) MCMC su \tau2^k_i: dalla full-conditional
        tau2_kMCMC.=step2_tau2(WT_k=WT_kMCMC, indici=indici_MCMC, par_inv_gamma_a=par_inv_gamma_a, par_inv_gamma_b=par_inv_gamma_b, Dati=Dati, u_k=u_kMCMC, n_time=n_time, n_instances=n_instances)::Vector{Float64}
        tau2_kSave[iMCMC, :] .= tau2_kMCMC
    end

    if step == 3
        ### STEP 3) MCMC su \beta^k: dalla full-conditional
        beta_kMCMC.=step3_beta(n_time_trasformati=n_time_trasformati, sigma2_beta=sigma2_beta, SIGMA_k=SIGMA_kMCMC, X_k=X_kMCMC, WT_k=WT_kMCMC, mu_beta=mu_beta)
        beta_kSave[iMCMC, :] .= beta_kMCMC
    end

    if step == 4
        ### STEP 4) MCMC su \sigma2_u: dalla full-conditional
        sigma2_uMCMC = step4_sigma2_u(n_instances=n_instances, par_inv_gamma_a=par_inv_gamma_a, par_inv_gamma_b=par_inv_gamma_b, u_k=u_kMCMC)
        sigma2_uSave[iMCMC] = sigma2_uMCMC
    end

    if step == 5
        ### STEP 5) MCMC su \sigma2_k_b: dalla full-conditional
```

```
144         sigma2_k_bMCMC = step5_sigma2_k_b(n_instances=n_instances,
    par_inv_gamma_a=par_inv_gamma_a, par_inv_gamma_b=par_inv_gamma_b,
    b_k=b_kMCMC)
145         sigma2_k_bSave[iMCMC] = sigma2_k_bMCMC
146     end
147
148     if step == 6
149         ### STEP 6)
150         ### STEP 6.a) MCMC su \alpha_k, \psi2_k, WT_k: passo Metropolis
151         gamma_i=A/(B+iterazione)
152         skippa_1=false
153         try
154             Prop_i_1, log_lambda_i_1, iterazione_aggiornamento_1,
    alpha_batch_1 = algoritmo4_alpha_psi2(X_i=X_i_1, gamma_i=gamma_i,
    iterazione_aggiornamento=iterazione_aggiornamento_1,
    size_batch=size_batch, alpha_batch=alpha_batch_1,
    alpha_star=alpha_star_1, log_lambda_i=log_lambda_i_1,
    Sigma_i=Sigma_i_1)
155
156             alpha_kprop=Prop_i_1[1]
157             psi2_kprop=Prop_i_1[2]
158
159             # alpha #
160             println("alpha_kMCMC:", alpha_kMCMC)
161             println("alpha_kprop:", alpha_kprop)
162             # psi #
163             println("psi2_kMCMC:", psi2_kMCMC)
164             println("psi2_kprop:", psi2_kprop)
165             # informazioni ulteriori #
166             println("alpha_batch_1: ", alpha_batch_1)
167             println("iterazione_aggiornamento_1: ",
    iterazione_aggiornamento_1)
168
169             if alpha_kprop < 0.001 || alpha_kprop > 0.999
170                 println("Rifiutato per alpha")
171                 iterazioni_saltate_per_alpha+=1
172                 skippa_1=true
173             end
174
175             if psi2_kprop <= 0
176                 println("Rifiutato per psi2")
177                 iterazioni_saltate_per_psi2+=1
178                 skippa_1=true
179             end
180
181             # WT_k - 1 #
182             # Il processo WT_k_prop:
183             if skippa_1==false
184                 WT_kprop[:], X_kMCMC_times_beta_kMCMC[:], SIGMA_kprop,
    U_SIGMA_kprop[:,:], M_prop[:], V_inv_prop, U_V_inv_prop[:,:],
    X_kMCMC[:,:]=
```

```
185                 FC_WT(n_time_trasformati=n_time_trasformati,
    T_k=T_kMCMC, alpha_k=alpha_kprop, psi2_k=psi2_kprop,
    beta_k=beta_kMCMC, tau2_k=tau2_kMCMC, indici=indici_MCMC,
    n_time=n_time, n_instances=n_instances, u_k=u_kMCMC, Dati=Dati)
186
187                 RatioMetropolis_alpha_psi2_WT=
188                 Metropolis_Ratio_alpha_psi_WT(n_instances=n_instances,
    n_time_trasformati=n_time_trasformati, WT_kprop=WT_kprop,
    indici_prop=indici_MCMC, tau2_kMCMC=tau2_kMCMC, n_time=n_time,
    Dati=Dati, u_kMCMC=u_kMCMC, SIGMA_kprop=SIGMA_kprop,
    U_SIGMA_kprop=U_SIGMA_kprop, alpha_kprop=alpha_kprop,
    par_inv_gamma_a=par_inv_gamma_a, par_inv_gamma_b=par_inv_gamma_b,
    psi2_kprop=psi2_kprop, WT_kMCMC=WT_kMCMC, M_MCMC=M_MCMC,
    V_inv_MCMC=V_inv_MCMC, U_V_inv_MCMC=U_V_inv_MCMC,
    indici_MCMC=indici_MCMC, X_kMCMC=X_kMCMC, beta_kMCMC=beta_kMCMC,
    SIGMA_kMCMC=SIGMA_kMCMC, U_SIGMA_kMCMC=U_SIGMA_kMCMC,
    alpha_kMCMC=alpha_kMCMC, psi2_kMCMC=psi2_kMCMC, M_prop=M_prop,
    V_inv_prop=V_inv_prop, U_V_inv_prop=U_V_inv_prop)
189
190                 println("RatioMetropolis_alpha_psi2_WT: ",
    RatioMetropolis_alpha_psi2_WT)
191                 alpha_batch_1+=RatioMetropolis_alpha_psi2_WT/size_batch
192
193                 u = rand()
194                 if u < RatioMetropolis_alpha_psi2_WT
195                     num_accettati_1+=1
196                     println("Proposta accettata")
197                     X_i_1          .= Prop_i_1
198                     alpha_kMCMC     = X_i_1[1]
199                     psi2_kMCMC      = X_i_1[2]
200                     WT_kMCMC       .= WT_kprop
201                     SIGMA_kMCMC    .= SIGMA_kprop
202                     U_SIGMA_kMCMC  .= U_SIGMA_kprop
203                     # Da controllare cosa devi aggiornare
204                     M_MCMC         .= M_prop
205                     V_inv_MCMC     .= V_inv_prop
206                     U_V_inv_MCMC   .= U_V_inv_prop
207                 end
208             end
209
210         catch e
211             if isa(e, PosDefException)
212                 iterazione_aggiornamento_1+=1
213                 iterazioni_saltate_per_PosDef_or_BoundsError+=1
214                 println("iterazioni_saltate_per_PosDef_or_BoundsError:
    ", iterazioni_saltate_per_PosDef_or_BoundsError)
215             else
216                 rethrow(e)
217             end
218         end
219
220         alpha_kSave[iMCMC] = alpha_kMCMC
221         psi2_kSave[iMCMC]  = psi2_kMCMC
```

```
222         copyto!(parent(Sigma_i_1),
223         Symmetric(Sigma_i_1+gamma_i*((X_i_1-mu_i_1)*(X_i_1-mu_i_1)'
224         -Sigma_i_1)+epsilon*I))
225         mu_i_1.+=gamma_i*(X_i_1-mu_i_1)
226     end
227 
228     if step == 7
229         ### STEP 7) MCMC su \b_k, WT_k: passo Metropolis
230         skippa_2=false
231         # b_k #
232         vettore = 1:n_instances
233         while !isempty(vettore)
234             skippa_2=false
235             try
236                 k = min(n_indici, length(vettore))
237                 selected_indices = randperm(length(vettore))[1:k]
238                 selected_elements = vettore[sort(selected_indices)]
239                 vettore = vettore[setdiff(1:end, selected_indices)]
240                 try
241                     Prop_i_2[:], T_kprop[:], indici_prop[:,:],
    log_lambda_i_2, iterazione_aggiornamento_2, alpha_batch_2 =
    algoritmo4_b_T_indici_separated2(;X_i=X_i_2, gamma_i=gamma_i,
    iterazione_aggiornamento=
242                     iterazione_aggiornamento_2,
    n_instances=n_instances, n_time=n_time,
    n_time_trasformati=n_time_trasformati, time_points=time_points,
    size_batch=size_batch, alpha_batch=alpha_batch_2,
    alpha_star=alpha_star_2, log_lambda_i=log_lambda_i_2,
    Sigma_i=Sigma_i_2, indici=selected_elements)
243 
244                     b_kprop.=Prop_i_2
245                     # informazioni ulteriori #
246                     println("alpha_batch_2: ", alpha_batch_2)
247                     println("iterazione_aggiornamento_2: ",
    iterazione_aggiornamento_2)
248 
249                 catch e
250                     if isa(e, DimensionMismatch)
251                         iterazione_aggiornamento_2+=1
252                         println("Dimensione dei tempi sbagliata")
253                         iterazioni_saltate_per_tempi+=1
254                         b_kSave[iMCMC,:]    .= b_kMCMC
255                         WT_kSave[iMCMC,:]   .= WT_kMCMC
256                         println("numero di accettati_2: ",
    num_accettati_2)
257                         skippa_2=true
258                     end
259                 end
260 
261                 if skippa_2==false
262                     n_time_trasformatiprop = length(T_kprop)
263                     # WT_k - 2 #
264                     # Il processo WT_k_prop:
```

```
                    WT_kprop[:], X_kprop_times_beta_kMCMC[:],
    SIGMA_kprop, U_SIGMA_kprop[:,:], M_prop[:], V_inv_prop,
    U_V_inv_prop[:,:], X_kprop[:,:]=
                    FC_WT(n_time_trasformati=n_time_trasformatiprop,
    T_k=T_kprop, alpha_k=alpha_kMCMC, psi2_k=psi2_kMCMC,
    beta_k=beta_kMCMC, tau2_k=tau2_kMCMC, indici=indici_prop,
    n_time=n_time, n_instances=n_instances, u_k=u_kMCMC, Dati=Dati)

                    # Metropolis_Ratio:
                    RatioMetropolis_b_WT=
                    Metropolis_Ratio_b_WT(n_instances=n_instances,
    n_time_trasformati=n_time_trasformati, WT_kprop=WT_kprop,
    indici_prop=indici_prop, tau2_kMCMC=tau2_kMCMC, n_time=n_time,
    Dati=Dati, u_kMCMC=u_kMCMC,
    X_kprop_times_beta_kMCMC=X_kprop_times_beta_kMCMC,
    SIGMA_kprop=SIGMA_kprop, U_SIGMA_kprop=U_SIGMA_kprop,
    b_kprop=b_kprop, WT_kMCMC=WT_kMCMC, M_MCMC=M_MCMC,
    V_inv_MCMC=V_inv_MCMC, U_V_inv_MCMC=U_V_inv_MCMC,
    indici_MCMC=indici_MCMC, X_kMCMC=X_kMCMC, beta_kMCMC=beta_kMCMC,
    SIGMA_kMCMC=SIGMA_kMCMC, U_SIGMA_kMCMC=U_SIGMA_kMCMC,
    sigma2_k_bMCMC=sigma2_k_bMCMC, b_kMCMC=b_kMCMC, M_prop=M_prop,
    V_inv_prop=V_inv_prop, U_V_inv_prop=U_V_inv_prop)

                    println("RatioMetropolis_b_WT: ",
    RatioMetropolis_b_WT)
                    alpha_batch_2+=RatioMetropolis_b_WT/size_batch

                    u = rand()
                    if u < RatioMetropolis_b_WT
                        num_accettati_2+=1
                        println("Proposta accettata")
                        X_i_2          .= Prop_i_2
                        b_kMCMC        .= X_i_2
                        WT_kMCMC       .= WT_kprop
                        SIGMA_kMCMC    .= SIGMA_kprop
                        U_SIGMA_kMCMC  .= U_SIGMA_kprop
                        # Da controllare cosa devi aggiornare
                        indici_MCMC    .= indici_prop
                        X_kMCMC        .= X_kprop
                        T_kMCMC        .= T_kprop
                        M_MCMC         .= M_prop
                        V_inv_MCMC     .= V_inv_prop
                        U_V_inv_MCMC   .= U_V_inv_prop
                    end
                end
            catch e
                if isa(e, PosDefException) || isa(e, BoundsError)
                    iterazione_aggiornamento_2+=1
                    iterazioni_saltate_per_PosDef_or_BoundsError+=1

    println("iterazioni_saltate_per_PosDeforBoundsError:"
                    ,iterazioni_saltate_per_PosDef_or_BoundsError)
                else
```

```
                    rethow(e)
                end
            end
            copyto!(parent(Sigma_i_2),
    Symmetric(Sigma_i_2+gamma_i*((X_i_2-mu_i_2)*
            (X_i_2-mu_i_2)'-Sigma_i_2)+epsilon*I))
            mu_i_2.+=gamma_i*(X_i_2-mu_i_2)
            println("numero di accettati_2: ", num_accettati_2)
        end

        println("X_i_2: ", X_i_2)

        b_kSave[iMCMC,:]         .= b_kMCMC
        WT_kSave[iMCMC,:]        .= WT_kMCMC
        IndiciSave[iMCMC, :, :] .= indici_MCMC
    end
end
ordine=shuffle(1:7)
println("ordine: ", ordine)
end
appSamp = thin
end
println("iterazioni_saltate_per_alpha: ", iterazioni_saltate_per_alpha)
println("iterazioni_saltate_per_psi2: ", iterazioni_saltate_per_psi2)
println("iterazioni_saltate_per_tempi: ", iterazioni_saltate_per_tempi)
println("iterazioni_saltate_per_PosDef_or_BoundsError: ",
    iterazioni_saltate_per_PosDef_or_BoundsError)
return Dict(
:Alpha_k => alpha_kSave,
:Psi2_k => psi2_kSave,
:b_k => b_kSave,
:Beta_k => beta_kSave,
:U_k => u_kSave,
:Tau2_k => tau2_kSave,
:Sigma2_u => sigma2_uSave,
:Sigma2_k_b => sigma2_k_bSave,
:WT_K => WT_kSave,
:INDICI => IndiciSave)

end
```

Generazione Dati Sintetici

Vi saranno due codici, il primo per la generazione della prima famiglia di dati sintetici, con natura molto irregolare che serve per determinare la capacità del modello nel apprendere la forma del processo. Segue una seconda generazione di dati sintetici più regolari, che invece permette di apprezzare maggiormente la capacità dell'algoritmo di carpire le informazioni inerenti alla modulazione.

Listing A.9: Prima famiglia di dati sintetici

```
    ### LIBRERIE ###
using Distributions
using Random
```

```
using LinearAlgebra
using Plots

p0::Plots.Plot=plot()
p00::Plots.Plot=plot()

### SEED ###
Random.seed!(103)

### DATI ###

# Dichiarazione delle variabili:
# Parametri
n_instances::Int64=8
n_time::Int64=15
time_points::Vector{Float64}=range(0, stop=1, length=n_time)  # Crea un vettore di punti temporali da 0 a 1


### CREAZIONE DATI DAL MODELLO CORRETTO ###
# Valori da stimare
alpha_kVERO::Float64=0.85
psi2_kVERO::Float64=3.5
tau2_kVERO::Vector{Float64}=ones(Float64, n_instances).*0.2
sigma2_uVERO::Float64=1
sigma2_k_bVERO::Float64=0.1

# Parametri per beta
mu_beta::Vector{Float64}=zeros(Float64, 2) # Fisso a (0,0)
sigma2_beta::Float64=30  # Fisso a 30

# Ulteriori entita' derivanti da quelle sopra:
beta_kVERO::Vector{Float64}=rmvnorm_nuova(1; mean=mu_beta, Sigma=Symmetric(Diagonal([sigma2_beta, sigma2_beta])))
u_kVERO::Vector{Float64}=rmvnorm_nuova(1, mean=fill(0.0, n_instances), Sigma=Symmetric(Diagonal(fill(sigma2_uVERO, n_instances))))
b_kVERO::Vector{Float64}, T_kVERO::Vector{Float64}, indici_VERO::Matrix{Int64}, n_time_transformedVERO::Int64=
controllo_tempi_trasformati(n_time=n_time, n_instances=n_instances, time_points=time_points, sigma2_k_b=sigma2_k_bVERO)

X_kVERO::Matrix{Float64}=hcat(ones(n_time_transformedVERO), T_kVERO)

SIGMA_kVERO::Symmetric{Float64}=creare_varcov_OU(T_k=T_kVERO, alpha=alpha_kVERO, psi2=psi2_kVERO)
X_kVERO_times_beta_kVERO::Vector{Float64}=X_kVERO*beta_kVERO
# Generazione del proceso:
WT_kVERO::Vector{Float64}=rmvnorm_nuova(1, mean=X_kVERO_times_beta_kVERO, Sigma=SIGMA_kVERO)
# Creazione della matrice YVERO:
YVERO::Matrix{Float64}=zeros(Float64, n_instances, n_time)
w_k_i::Vector{Float64}=zeros(Float64, n_time)

```

```
# Ciclo attraverso gli istanti
for i in 1:n_instances
    # Estrai w_k_i da WT_kVERO usando gli indici
    w_k_i.=WT_kVERO[indici_VERO[i, :]]
    # Crea la distribuzione normale multivariata
    YVERO[i, :]=rmvnorm_nuova(1, mean=u_kVERO[i] .+ w_k_i,
    Sigma=Symmetric(Diagonal(tau2_kVERO[i]*ones(n_time))))
end

# Creiamo un plot delle righe di YVERO
gr(size = (2000, 1000)) # use gr as the plot backend
p0::Plots.Plot=plot(YVERO', xlabel = "Tempo", ylabel = "Valore", title
    = "Righe della matrice YVERO", lw = 2, legend=false)
YVERO_U = YVERO .- u_kVERO
plot(YVERO_U', xlabel = "Tempo", ylabel = "Valore", title = "Righe
    della matrice YVERO senza il valore di spostamento", lw = 2,
    legend=false)
p00::Plots.Plot=plot(WT_kVERO, xlabel = "Tempo", ylabel = "Valore",
    title = "WT_kVERO", lw = 2, legend=false)
```

Listing A.10: Seconda famiglia di dati sintetici

```
### LIBRARIES ###
using Pkg
using Distributions
using Random
using LinearAlgebra
using Plots
using Interpolations

 # Funzione per generare numeri interi casuali
 function genera_lista_interi(n_instances, n_groups)::Vector{Int64}
     return rand(1:n_groups, n_instances)
 end

 # Funzione per simulare i processi e plottare
 function simula_processi_e_plot(;
     FUNC::Vector{Function} = [identity], # Lista di funzioni,
    default: identity
     n_groups::Int64=1,
     n_instances::Union{Int64, Nothing}=nothing,
     n_time::Union{Int64, Nothing}=nothing,
     B::Union{Vector{Float64}, Nothing}=nothing, # Vettore delle
    modulazioni "base", una per ogni gruppo.
     U::Union{Vector{Float64}, Nothing}=nothing,
     TAU2::Union{Vector{Float64}, Nothing}=nothing
 )::Tuple{Matrix{Float64}, Vector{Int}, Matrix{Float64}}

    # Dichiarazione delle variabili:
    time_points = range(0, stop=1, length=n_time)
    w_k = Matrix{Float64}(undef, n_groups, n_time)
    Y = Matrix{Float64}(undef, n_instances, n_time)
    b_i::Float64=0.0
    u_i::Float64=0.0
```

```
    tau2_i::Float64=0.0

    # Inizializza w_k
    for i in 1:n_groups
        w_k[i, :] .= FUNC[i].(time_points)
    end

    # Assegna ogni istanza a un gruppo casualmente
    indici_gruppo::Vector{Int} = genera_lista_interi(n_instances, n_groups)

    # Calcola Y
    for i in 1:n_instances
        indice_gruppo_i = indici_gruppo[i]
        b_i = B[i]
        u_i = U[i]
        tau2_i = TAU2[i]
        Y[i, :] = rmvnorm_nuova(1;mean=u_i.+FUNC[indice_gruppo_i].(time_points .^ exp(b_i)), Sigma=Symmetric(Diagonal(fill(tau2_i, n_time))))
    end

    return w_k, indici_gruppo, Y
end

### CHIAMATA ###
using Random
using Distributions

# 1) Entita' iniziali:
Random.seed!(731)

n_instances::Int64 = 21
n_time::Int64=15
n_gruppi::Int64 = 3
time_points::Vector{Float64}=range(0, stop=1, length=n_time)  # Crea un vettore di punti temporali da 0 a 1

# 2) Coefficienti di definizione della U:
sigma2_U::Float64=2.0
U::Vector{Float64}=rand(Normal(0, sqrt(sigma2_U)), n_instances)

# 3) Coefficienti di definizione della TAU2:
TAU2::Vector{Float64}=fill(0.1, n_instances)

# 4) Coefficienti di definzione della B:
sd_B::Float64=0.30
B::Vector{Float64}=rand(Normal(0, sd_B), n_instances)

# Definizione delle funzioni
a_k::Vector{Float64}=[1, 0.5, -1.2]
f1(t) = sin(2 * 3.1415 * t * a_k[1])*10+10
f2(t) = sin(2 * 3.1415 * t * a_k[2])*10+5
```

```
f3(t) = sin(2 * 3.1415 * t * a_k[3])*10+15

# Lista delle funzioni
FUNC = [f1, f2, f3]

WT_k::Matrix{Float64}, indici_gruppo::Vector{Int64},
    Y::Matrix{Float64}=simula_processi_e_plot(
     FUNC=FUNC,
     n_groups=n_gruppi,
     n_instances=n_instances,
     n_time=n_time,
     B=B,
     U=U,
     TAU2=TAU2
)

### PLOT ###
gruppo::Int64=1

## WT_kVERO ##
WT_kVERO::Vector{Float64}=deepcopy(WT_k[gruppo, :])

## U_VERO ##
U_VERO::Vector{Float64}=U[indici_gruppo .== gruppo]

## TAU2_VERO ##
TAU2_VERO::Vector{Float64}=TAU2[indici_gruppo .== gruppo]

## B_VERO ##
B_VERO::Vector{Float64}=B[indici_gruppo .== gruppo]

## n_instances_VERO ##
n_instances_VERO::Int64 = length(B_VERO)

## YVERO ##
p0::Plots.Plot=plot()
p0::Plots.Plot=plot(time_points, WT_k[gruppo, :], label="Processo
    w_$gruppo", lw=2, color=:blue, legend=:topright)
YVERO::Matrix{Float64} = Y[indici_gruppo .== gruppo, :]
for row in eachrow(YVERO)
    plot!(p0, time_points, row, label="", lw=1, color=:red, alpha=1)
end

# Aggiungere titolo e etichette
p0 = title!(p0, "Processi per il gruppo $gruppo")
p0 = xlabel!(p0, "Tempo")
p0 = ylabel!(p0, "Valore")


p00::Plots.Plot = plot(time_points, WT_k[gruppo, :], label="Processo
    w_$gruppo", lw=2, color=:blue, legend=:topright)
# Calcola Y_gruppo_meno_u
Y_gruppo_meno_u::Matrix{Float64} = YVERO .- U[indici_gruppo .== gruppo]
```

```
# Aggiungi le serie di Y_gruppo_meno_u al grafico esistente
for row in eachrow(Y_gruppo_meno_u)
    plot!(p00, time_points, row, label="", lw=1, color=:red, alpha=1)
end

# Aggiungere titolo e etichette
p00 = title!(p00, "Processi per il gruppo $gruppo, senza la componente traslazionale u")
p00 = xlabel!(p00, "Tempo")
p00 = ylabel!(p00, "Valore")

## WT_kVERO_interpolated ##
new_length::Int64=n_time * n_instances_VERO - 2 * n_instances_VERO + 2

# Creazione dell'interpolazione
x_original = 1:n_time  # Non so come tipizzarla
x_interpolated = range(1, n_time, length=new_length)  # Non so come tipizzarla
spline = CubicSplineInterpolation(x_original, WT_kVERO)  # Non so come tipizzarla
WT_kVERO_interpolated::Vector{Float64} = spline.(x_interpolated)
```

Inizializzazione

```
using Distributions
using Random
using LinearAlgebra
using Plots

p000::Plots.Plot=plot()

Random.seed!(1000)

n_instances::Int64=length(YVERO[:,1])
n_time::Int64=length(YVERO[1,:])
time_points::Vector{Float64}=range(0, stop=1, length=n_time)  # Crea un vettore di punti temporali da 0 a 1


### Solo per il Test ###
mu = 0     # Media
sigma = 1 # Deviazione standard

# Creazione della matrice casuale Y_NOISE
Y_NOISE = rand(Normal(mu, sigma), size(YVERO))
### Fine ###

plot(title="Dati", Y_NOISE', legend=false)

# Parametri per beta
mu_beta::Vector{Float64}=zeros(Float64, 2) # Fisso a (0,0)
```

```
sigma2_beta::Float64=30  # Fisso a 30

### INIZIALIZZAZIONE ###

### STEP 2 ###
tau2_kINIZ::Vector{Float64} = ones(Float64, n_instances)

### STEP 3 ###
beta_kINIZ::Vector{Float64}=rmvnorm_nuova(1; mean=mu_beta,
    Sigma=Symmetric(Diagonal([sigma2_beta, sigma2_beta])))

### STEP 4 & STEP 1 ###
sigma2_uINIZ::Float64=1
u_kINIZ::Vector{Float64}=rmvnorm_nuova(1; mean=fill(0.0, n_instances),
    Sigma=Symmetric(Diagonal(fill(sigma2_uINIZ, n_instances))))

### STEP 5 ###
sigma2_k_bINIZ::Float64=0.15

### STEP 6 ###
alpha_kINIZ::Float64=0.5

psi2_kINIZ::Float64=1

### STEP 7 ###
b_kINIZ::Vector{Float64}, T_kINIZ::Vector{Float64},
    indici_INIZ::Matrix{Int64}, n_time_transformedINIZ::Int64 =
    controllo_tempi_trasformati(n_time=n_time,
    n_instances=n_instances, time_points=time_points,
    sigma2_k_b=sigma2_k_bINIZ)

# La matrice X modificata
X_kINIZ::Matrix{Float64}=hcat(ones(n_time_transformedINIZ), T_kINIZ)

SIGMA_kINIZ::Symmetric{Float64}=creare_varcov_OU(T_k=T_kINIZ,
    alpha=alpha_kINIZ, psi2=psi2_kINIZ)

X_kINIZ_times_beta_kINIZ::Vector{Float64}=X_kINIZ*beta_kINIZ

U_SIGMA_kINIZ::UpperTriangular{Float64}=cholesky(SIGMA_kINIZ).U
SIGMA_kINIZ_inv::Symmetric{Float64}=chol2inv(U_SIGMA_kINIZ)

V_inv_INIZ::Symmetric{Float64}=somma_matrice_diagonale(B=SIGMA_kINIZ_inv,
    v=tau2_kINIZ, indici_matrix=indici_INIZ, n_time=n_time,
    n_instances=n_instances)
U_V_inv_INIZ::UpperTriangular{Float64}=cholesky(V_inv_INIZ).U

vettore::Vector{Float64}=
ottenere_vettore(n_time_transformed=n_time_transformedINIZ,
    n_instances=n_instances, n_time=n_time, indici=indici_INIZ,
    Y=Y_NOISE, u_k=u_kINIZ, tau2_k=tau2_kINIZ)

vettore .+= SIGMA_kINIZ_inv*X_kINIZ_times_beta_kINIZ
```

```
69
70 L_V_inv_INIZ::LowerTriangular{Float64}=
71 LowerTriangular(transpose(U_V_inv_INIZ))
72 M_INIZ::Vector{Float64}=UpperTriangular(U_V_inv_INIZ) \ (L_V_inv_INIZ
       \ vettore)
73
74 # Il processo WT_k_INIZ:
75 WT_kINIZ::Vector{Float64}=rmvnorm_nuova(1; mean=M_INIZ,
       invSigma=V_inv_INIZ, U_invSigma=U_V_inv_INIZ)
76 p000::Plots.Plot=plot(WT_kINIZ, xlabel = "Tempo", ylabel = "Valore",
       title = "WT_kINIZ", lw = 2, legend=false)
```

Funzioni Utili

```
1  using LinearAlgebra
2  using Distributions
3
4  ### FUNZIONI UTILI ###
5
6  function chol2inv(U::UpperTriangular{Float64})::Symmetric{Float64}
7      return Symmetric(UpperTriangular(U \ I(size(U, 1))) *
       transpose(UpperTriangular(U \ I(size(U, 1)))))
8  end
9
10 function dmvnorm_nuova(x::Union{Float64, Vector{Float64}};
       mean::Union{Float64, Vector{Float64}, Nothing}=nothing,
       Sigma::Union{Float64, Symmetric{Float64}, Nothing}=nothing,
       invSigma::Union{Symmetric{Float64}, Nothing}=nothing,
       U_Sigma::Union{UpperTriangular{Float64}, Nothing}=nothing,
       U_invSigma::Union{UpperTriangular{Float64}, Nothing}=nothing,
       logflag::Bool=false)::Float64
11
12     # Dichiarazione delle variabili:
13     n::Int64=length(mean)
14     log_det_Sigma::Float64=0.0
15     log_det_invSigma::Float64=0.0
16     mah_dist::Float64=0.0
17     log_density::Float64=0.0
18
19     if  x isa Number
20         if logflag
21             log_density=logpdf(Normal(mean, sqrt(Sigma)), x)
22             return log_density
23         else
24             density=pdf(Normal(mean, sqrt(Sigma)), x)
25             return density
26         end
27     end
28
29     # Corpo del codice:
30     if mean === nothing
```

```
31          throw(ArgumentError("La media non puo' essere nulla."))
32      end
33
34      if Sigma === nothing && invSigma === nothing
35          throw(ArgumentError("Devi fornire almeno Sigma o invSigma."))
36      end
37
38      # Caso 1: Viene fornita Sigma
39      if Sigma !== nothing && invSigma === nothing
40          if U_Sigma === nothing
41              U_Sigma = UpperTriangular(cholesky(Sigma).U)  # Calcola la
    fattorizzazione Cholesky
42          end
43          # Supponiamo che U sia la matrice triangolare superiore dalla
    decomposizione Cholesky
44          invSigma = Symmetric(chol2inv(U_Sigma))
45
46          log_det_Sigma = 2 * sum(log.(diag(U_Sigma)))  #
    Log-determinante di Sigma
47      end
48
49      # Caso 2: Viene fornita invSigma
50      if Sigma === nothing && invSigma !== nothing
51          if U_invSigma === nothing
52              U_invSigma = UpperTriangular(cholesky(invSigma).U)  #
    Calcola la fattorizzazione Cholesky di invSigma
53          end
54          log_det_invSigma = 2 * sum(log.(diag(U_invSigma)))  #
    Log-determinante di invSigma
55          log_det_Sigma = -log_det_invSigma  # Log-determinante di Sigma
56      end
57
58      # Calcolo della distanza di Mahalanobis
59      mah_dist = (x - mean)' * invSigma * (x - mean)
60
61      # Calcolo della densita' o del log-densita'
62      if logflag
63          log_density = -0.5 * (n * log(2 * 3.1415) + log_det_Sigma +
    mah_dist)
64          return log_density  # Restituisce il log-densita' come numero
65      else
66          density = exp(-0.5 * (n * log(2 * 3.1415) + log_det_Sigma +
    mah_dist))
67          return density  # Restituisce la densita' come numero
68      end
69  end
70
71  function rmvnorm_nuova(n::Int64; mean::Union{Vector{Float64},
    Nothing}=nothing, Sigma::Union{Symmetric{Float64},
    Nothing}=nothing, invSigma::Union{Symmetric{Float64},
    Nothing}=nothing, U_Sigma::Union{UpperTriangular{Float64},
    Nothing}=nothing, U_invSigma::Union{UpperTriangular{Float64},
    Nothing}=nothing)::Union{Matrix{Float64}, Vector{Float64}}
```

```
72      p::Int64=length(mean)
73      if n>1
74          Z1::Matrix{Float64}=randn(n, p)
75          if Sigma === nothing && invSigma === nothing
76              error("Devi fornire almeno Sigma o invSigma.")
77          end
78
79          if Sigma !== nothing && U_Sigma === nothing
80              U_Sigma=UpperTriangular(cholesky(Sigma).U)
81          end
82
83          if Sigma===nothing && invSigma!==nothing
84
85              if U_invSigma===nothing
86                  U_invSigma=UpperTriangular(cholesky(invSigma).U)
87              end
88              U_Sigma=UpperTriangular((UpperTriangular(U_invSigma) \
    I(size(U_invSigma, 2)))')
89          end
90          ris1::Matrix{Float64}=((Z1 * U_Sigma) .+ mean')'
91          return  ris1
92      else
93          Z2::Vector{Float64}=randn(p)
94
95          if Sigma === nothing && invSigma === nothing
96              error("Devi fornire almeno Sigma o invSigma.")
97          end
98
99          if Sigma !== nothing && U_Sigma === nothing
100             U_Sigma=UpperTriangular(cholesky(Sigma).U)
101         end
102
103         if Sigma===nothing && invSigma!==nothing
104
105             if U_invSigma===nothing
106                 U_invSigma=UpperTriangular(cholesky(invSigma).U)
107             end
108             U_Sigma=UpperTriangular(UpperTriangular(U_invSigma) \
    I(size(U_invSigma, 2)))'
109         end
110         ris2::Vector{Float64}=vec((Z2' * U_Sigma) .+ mean')
111         return ris2
112     end
113 end
114
115 function somma_matrice_diagonale(;B::Union{Symmetric{Float64},
    Nothing}=nothing, v::Union{Vector{Float64}, Nothing}=nothing,
    indici_matrix::Union{Matrix{Int64}, Nothing}=nothing,
    n_time::Union{Int64, Nothing}=nothing, n_instances::Union{Int64,
    Nothing})::Symmetric{Float64}
116     A = Symmetric(deepcopy(B))
117     # Dichiarazione delle variabili:
118     dim_A::Int64=n_time + (n_time - 2) * (n_instances - 1)
```

```
119
120     # Controlli:
121     # Controllo che A non sia nulla
122     if A === nothing
123         throw(ArgumentError("La matrice A non puo' essere nulla."))
124     end
125     # Controllo per dimensioni consistenti
126     if size(A, 1) != dim_A || size(A, 2) != dim_A
127         throw(ArgumentError("Le dimensioni della matrice A non sono
    corrette."))
128     end
129
130     # Corpo della funzione:
131     for i in 1:n_instances
132         for j in 1:n_time
133             A[indici_matrix[i, j], indici_matrix[i, j]] +=  1 / v[i]
    # Aggiungi 1/v[i] all'elemento diagonale
134         end
135     end
136     return Symmetric(A)
137 end
138
139 function ottenere_T_k_e_indici(
140     ; b_k::Union{Vector{Float64}, Nothing}=nothing,
141     time_points::Union{Vector{Float64}, Nothing}=nothing,
142     n_instances::Union{Int64, Nothing}=nothing
143 )::Tuple{Vector{Float64}, Matrix{Int64}}
144     # Dichiarazione delle variabili
145     time_points_transformed_i::Vector{Float64} = zeros(Float64,
    length(time_points))
146     T_k::Vector{Float64} = Float64[]
147     indici::Matrix{Int64} = zeros(Int64, n_instances,
    length(time_points))
148
149     # Corpo della funzione
150     for i in 1:n_instances
151         # Trasforma i punti temporali
152         time_points_transformed_i .= time_points.^exp(b_k[i])
153         # Aggiorna T_k con i nuovi valori unici e ordinati
154         T_k = sort(unique(vcat(T_k, time_points_transformed_i)))
155     end
156
157     # Calcola gli indici usando T_k generato
158     for i in 1:n_instances
159         # Trasforma i punti temporali per l'indice i
160         time_points_transformed_i .= time_points.^exp(b_k[i])
161         # Trova gli indici dei valori trasformati in T_k
162         indici[i, :] .= [findfirst(x -> x == tp, T_k) for tp in
    time_points_transformed_i]
163     end
164
165     return T_k, indici
166 end
```

```
167
168 function creare_varcov_OU(; T_k::Union{Vector{Float64},
        Nothing}=nothing, alpha::Union{Float64, Nothing}=nothing,
        psi2::Union{Float64, Nothing}=nothing, epsilon::Float64=1e-13) ::
        Symmetric{Float64}
169     # Dichiarazione delle variabili
170     n_time_transformed::Int64 = length(T_k)
171     SIGMA_k::Matrix{Float64} = zeros(Float64, n_time_transformed,
        n_time_transformed)
172     rho::Float64 = -log(alpha)
173     tau2::Float64 = psi2 / (1 - alpha^2)
174
175     # Corpo della funzione:
176     for i in 1:n_time_transformed
177         for j in i:n_time_transformed
178             SIGMA_k[i, j] = tau2 * exp(-rho * abs(T_k[i] - T_k[j]))
179             SIGMA_k[j, i] = SIGMA_k[i, j]  # Imposta la simmetria
180         end
181         SIGMA_k[i, i]+=epsilon
182     end
183     # Converti in una matrice simmetrica
184     return Symmetric(SIGMA_k)
185 end
186
187 function ottenere_vettore(;n_time_transformed::Union{Int64,
        Nothing}=nothing, n_instances::Union{Int64, Nothing}=nothing,
        n_time::Union{Int64, Nothing}=nothing,
        indici::Union{Matrix{Int64}, Nothing}=nothing,
        Y::Union{Matrix{Float64}, Nothing}=nothing,
        u_k::Union{Vector{Float64}, Nothing}=nothing,
        tau2_k::Union{Vector{Float64}, Nothing}=nothing)::Vector{Float64}
188
189     vettore::Vector{Float64} = zeros(Float64, n_time_transformed)
190
191     for i in 1:n_instances
192         vettore[indici[i, :]].+=(Y[i, :].-u_k[i])/(tau2_k[i])
193     end
194     return vettore
195 end
196
197 function controllo_tempi_trasformati(;n_time::Union{Int64,
        Nothing}=nothing, n_instances::Union{Int64, Nothing}=nothing,
        time_points::Union{Vector{Float64}, Nothing}=nothing,
        sigma2_k_b::Union{Float64,
        Nothing}=nothing)::Tuple{Vector{Float64}, Vector{Float64},
        Matrix{Int64}, Int64}
198     # Dichiarazione delle variabili:
199     b_k::Vector{Float64}=rand(MvNormal(fill(0, n_instances),
        Symmetric(Diagonal(fill(sigma2_k_b, n_instances)))))
200     T_k::Vector{Float64},
        indici::Matrix{Int64}=ottenere_T_k_e_indici(b_k=b_k,
        time_points=time_points, n_instances=n_instances)
201     n_time_transformed::Int64=length(T_k)
```

```
202     # Restituisci i valori
203     return b_k, T_k, indici, n_time_transformed
204 end
```

Bibliografia

[1] A. F. McBride e D. O. Hebb. "Behavior of the captive bottle-nose dolphin, Tursiops truncatus". In: *Journal of Comparative and Physiological Psychology* 41 (1948), pp. 111–123 (cit. a p. 1).

[2] L. S. Sayigh, V. M. Janik, F. H. Jensen, M. D. Scott, P. L. Tyack e R. S. Wells. "The Sarasota Dolphin Whistle Database: A unique long-term resource for understanding dolphin communication". In: *Frontiers in Marine Science* 9 (2022). A cura di Rebecca Dunlop, p. 923046. DOI: 10.3389/fmars.2022.923046. URL: https://doi.org/10.3389/fmars.2022.923046 (cit. alle pp. 1–3).

[3] F. Jensen. *Decoding Communication in Nonhuman Species II*. Conference presentation streamed live on YouTube. Simons Institute for the Theory of Computing. Giu. 2023. URL: https://www.youtube.com/watch?v=ko0-rZ_hWPI (cit. alle pp. 1, 2).

[4] C. Semenzin. *Decoding Communication in Nonhuman Species III*. Conference presentation streamed live on YouTube. Simons Institute for the Theory of Computing. Giu. 2023. URL: https://www.youtube.com/watch?v=YiYkdsO6J4k (cit. alle pp. 1–3).

[5] R. C. Connor, M. R. Heithaus e L. M. Barre. "Complex social structure, alliance stability and mating access in a bottlenose dolphin 'super-alliance'". In: *Proceedings of the Royal Society of London. Series B: Biological Sciences* 268.1464 (2001), pp. 263–267. DOI: 10.1098/rspb.2000.1357 (cit. a p. 1).

[6] J.S. Lewis. "Mud plume feeding, a unique foraging behavior of the bottlenose dolphin in the Florida Keys". In: *Gulf of Mexico Science* 21 (2003), pp. 92–97 (cit. a p. 1).

[7] A. N. Popper. "Sound Emission and Detection by Delphinids". In: *Cetacean Behavior: Mechanisms and Functions*. A cura di L. M. Herman. New York: Wiley-Interscience, 1980, pp. 1–52 (cit. a p. 2).

[8] M. C. Caldwell e D. K. Caldwell. "Vocalization of naive captive dolphins in small groups". In: *Science* 159 (1968), pp. 1121–1123. DOI: 10.1126/science.159.3819.1121 (cit. a p. 2).

[9] F. Mustun et al. "Whistle variability and social acoustic interactions in bottlenose dolphins". In: *Cold Spring Harbor Laboratory* (2024). DOI: 10.1101/2024.10.15.618471 (cit. a p. 2).

[10] V. M. Janik e L. S. Sayigh. "Communication in bottlenose dolphins: 50 years of signature whistle research". In: *Behavioral Ecology and Sociobiology* 67.1 (gen. 2013), pp. 1–18. DOI: 10.1007/s00265-013-1561-6 (cit. a p. 2).

[11] L. S. Sayigh, P. L. Tyack, R. S. Wells e M. D. Scott. "Signature whistles of free-ranging bottlenose dolphins, Tursiops truncatus: mother-offspring comparisons". In: *Behavioral Ecology and Sociobiology* 26 (1990), pp. 247–260. DOI: 10.1007/BF00178318 (cit. alle pp. 2, 3).

[12] H. C. Eskelinen e B. L. Jones. "Acoustic Characteristics of Bubblestream-Associated Whistles Produced by Atlantic Bottlenose Dolphins (Tursiops truncatus) During the First Thirty Days of Life". In: *Aquatic Mammals* 47.4 (2021), pp. 337–348. DOI: 10.1578/AM.47.4.2021.337 (cit. a p. 2).

[13] K. C. Buckstaff. “Effects of watercraft noise on the acoustic behavior of bottlenose dolphins, Tursiops truncatus, in Sarasota Bay, Florida”. In: *Marine Mammal Science* 20 (2004), pp. 709–725. DOI: 10.1111/j.1748-7692.2004.tb01192.x (cit. a p. 2).

[14] O. Boisseau. “Quantifying the acoustic repertoire of a population: the vocalizations of free-ranging bottlenose dolphins in Fiordland, New Zealand”. In: *Journal of the Acoustical Society of America* 117.4 (2005), pp. 2318–2329. DOI: 10.1121/1.1850471 (cit. a p. 2).

[15] L. S. Sayigh e V. M. Janik. “Signature whistles”. In: *Encyclopedia of Animal Behavior*. A cura di M. D. Breed e J. Moore. Oxford: Academic Press, 2010 (cit. a p. 2).

[16] V.M. Janik e N.J. Quick. “Bottlenose dolphins exchange signature whistles when meeting at sea”. In: *Proceedings of the Royal Society B: Biological Sciences* 279.1738 (lug. 2012), pp. 2539–2545. DOI: 10.1098/rspb.2011.2537 (cit. a p. 2).

[17] V. M. Janik, L. S. Sayigh e R. S. Wells. “Signature Whistle Shape Conveys Identity Information to Bottlenose Dolphins”. In: *Proceedings of the National Academy of Sciences* 103 (2006), pp. 8293–8297. DOI: 10.1073/pnas.0509918103 (cit. a p. 2).

[18] V. B. Deecke e V. M. Janik. “Automated categorization of bioacoustic signals: avoiding perceptual pitfalls”. In: *Journal of the Acoustical Society of America* 119 (2006), pp. 645–653 (cit. a p. 3).

[19] V. M. Janik, G. Dehnhardt e D. Todt. “Signature whistle variations in a bottlenosed dolphin, *Tursiops truncatus*”. In: *Behavioral Ecology and Sociobiology* 35 (1994), pp. 243–248 (cit. a p. 3).

[20] D. S. Pace et al. “Capitoline Dolphins: Residency Patterns and Abundance Estimate of Tursiops truncatus at the Tiber River Estuary (Mediterranean Sea)”. In: *Biology* 10 (2021). DOI: 10.3390/biology10040275 (cit. a p. 5).

[21] D. S. Pace et al. “Resources and population traits modulate the association patterns in the common bottlenose dolphin living nearby the Tiber River estuary (Mediterranean Sea)”. In: *Frontiers in Marine Science* 9 (2022). DOI: 10.3389/fmars.2022.935235 (cit. a p. 5).

[22] V. M. Janik, S. L. King, L. S. Sayigh e R. S. Wells. “Identifying signature whistles from recordings of groups of unrestrained bottlenose dolphins (*Tursiops truncatus*)”. In: *Marine Mammal Science* 29 (2013), pp. 109–122. DOI: 10.1111/j.1748-7692.2011.00549.x (cit. a p. 5).

[23] G. Mastrantonio, G. Jona Lasinio, A Pollice, P. O. Pammer, G. Pedrazzi, D. S. Pace e M. S. Labriola. “Clustering Dolphin Signature Whistles with Dirichlet Process Mixtures”. Submitted to the Annals of Applied Statistics. 2024 (cit. alle pp. 5, 29, 30, 56, 86).

[24] G. E. P. Box e G. C. Tiao. *Bayesian Inference in Statistical Analysis*. Wiley Classics Library Edition. New York: Wiley, 1992 (cit. a p. 6).

[25] P.D. Hoff. *A First Course in Bayesian Statistical Methods*. Springer Texts in Statistics. Dordrecht, Heidelberg, London, New York: Springer, 2009. ISBN: 978-0-387-92299-7. DOI: 10.1007/978-0-387-92407-6 (cit. alle pp. 6, 11).

[26] C. P. Robert. *The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation*. Second. Springer Texts in Statistics. New York: Springer, 2001. ISBN: 978-0387952314 (cit. a p. 6).

[27] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari e D. B. Rubin. *Bayesian Data Analysis*. Third. Updated with errors fixed as of 15 February 2021. Boca Raton, FL: CRC Press, 2021. ISBN: 978-1439840955 (cit. alle pp. 7, 12, 14).

[28] P. Müller, C. D. Paulino e M. A. A. Turkman. *Computational Bayesian Statistics: An Introduction*. Vol. 11. Institute of Mathematical Statistics Textbooks. Cambridge, UK: Cambridge University Press, 2019. ISBN: 978-1-108-48103-8. DOI: 10.1017/9781108646185 (cit. alle pp. 7, 13).

[29] G. Casella e R.L. Berger. *Statistical Inference*. 2nd. Pacific Grove, CA: Duxbury, 2002. ISBN: 978-0534243128 (cit. a p. 7).

[30] J.M. Bernardo e A.F.M. Smith. *Bayesian Theory*. Wiley Series in Probability and Statistics. Chichester, UK: John Wiley & Sons, 1994. ISBN: 978-0471924166. DOI: 10.1002/9780470316870 (cit. a p. 7).

[31] M. Franzese e A. Iuliano. *Hidden Markov Models*. Institute for Applied Mathematics "Mauro Picone". Napoli, Italy: Elsevier Inc., 2019 (cit. a p. 9).

[32] O. Knill. *Probability Theory and Stochastic Processes with Applications*. New Delhi, India: Overseas Press India Private Limited, 2009. ISBN: 978-81-89938-40-6 (cit. a p. 9).

[33] Henk C. Tijms. *A First Course in Stochastic Models*. Chichester, England: John Wiley & Sons, 2003. ISBN: 978-0-471-49880-1 (cit. a p. 9).

[34] R. Durrett. *Essentials of Stochastic Processes*. 2nd. New York, NY: Springer, 2012. ISBN: 978-1-4614-3614-0 (cit. a p. 10).

[35] C. P. Robert e G. Casella. *Monte Carlo Statistical Methods*. 2nd. Springer Texts in Statistics. New York, NY: Springer, 2004. ISBN: 978-0-387-21239-5. DOI: 10.1007/978-1-4757-4145-2 (cit. a p. 11).

[36] Robert H. Shumway e David S. Stoffer. *Time Series Analysis and Its Applications: With R Examples*. 3rd. New York: Springer, 2011. ISBN: 978-1-4419-7864-6. DOI: 10.1007/978-1-4419-7865-3 (cit. alle pp. 15, 16).

[37] R. Douc, E. Moulines e D. Stoffer. *Nonlinear Time Series: Theory, Methods, and Applications with R Examples*. Chapman & Hall/CRC Texts in Statistical Science. Boca Raton, FL: Chapman & Hall/CRC, 2014. ISBN: 978-1466502253 (cit. alle pp. 15, 17).

[38] R. J. Hyndman. *Forecasting: Principles and Practice*. Leader: Rob J. Hyndman, 23–25 September 2014, University of Western Australia. 2014. URL: https://robjhyndman.com/uwa (cit. a p. 17).

[39] A. Nielsen. *Practical Time Series Analysis: Prediction with Statistics and Machine Learning*. Sebastopol, CA: O'Reilly Media, 2019. ISBN: 978-1492041658 (cit. a p. 17).

[40] G. E. Uhlenbeck e L. S. Ornstein. "On the theory of the Brownian motion". In: *Physical Review* 36 (1930), pp. 823–841 (cit. a p. 18).

[41] O. Pourret, P. Naim e B. Marcot. *Bayesian Networks: A Practical Guide to Applications*. Electricité de France, France; ELSEWARE, France; USDA Forest Service, USA, 2008 (cit. a p. 19).

[42] A. P. Dawid. "Conditional Independence in Statistical Theory". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 41.1 (1979). Accessed 10 Jan. 2025, pp. 1–31. URL: http://www.jstor.org/stable/2984718 (cit. a p. 20).

[43] T. Koski e J. M. Noble. *Bayesian Networks: An Introduction*. Wiley Series in Probability and Statistics. First published: 25 September 2009. John Wiley & Sons, Ltd, 2009. ISBN: 9780470743041. DOI: 10.1002/9780470684023 (cit. a p. 20).

[44] G. Jona Lasinio, G. Mastrantonio e A. Pollice. "Discussing the "big n problem"". In: *Journal of the Italian Statistical Society* 22.1 (2013), pp. 97–112. DOI: 10.1007/s10260-012-0207-2 (cit. a p. 30).

[45] C. Andrieu e J. Thoms. "A Tutorial on Adaptive MCMC". In: *Statistics and Computing* 18.4 (2008), pp. 343–373. DOI: 10.1007/s11222-008-9110-y. URL: https://doi.org/10.1007/s11222-008-9110-y (cit. a p. 43).

[46] Bau III D. e Trefethen L. N. *Numerical Linear Algebra*. Philadelphia, PA: Society for Industrial e Applied Mathematics, 1997 (cit. a p. 43).

[47] L. Marshall, D. Nott e A. Sharma. “A Comparative Study of Markov Chain Monte Carlo Methods for Conceptual Rainfall-Runoff Modeling”. In: *Water Resources Research* 40.2 (2004), W02501. DOI: 10.1029/2003WR002378. URL: https://www.researchgate.net/publication/248808832_A_Comparative_Study_of_Markov_Chain_Monte_Carlo_Methods_for_Conceptual_Rainfall-Runoff_Modeling (cit. a p. 45).

[48] P. Diaconis. “Some things we’ve learned (about Markov chain Monte Carlo)”. In: *Stanford University* (2008) (cit. a p. 46).

[49] B. He, C. De Sa, I. Mitliagkas e C. Ré. “Scan Order in Gibbs Sampling: Models in Which it Matters and Bounds on How Much”. In: *Proceedings of Stanford University*. Stanford University. 2021 (cit. a p. 46).

[50] R. A. Levine e G. Casella. “Comment: On Random Scan Gibbs Samplers”. In: *Statistical Science* 5.1 (1990), pp. 85–88 (cit. a p. 46).

[51] W. M. Bolstad e J. M. Curran. *Introduction to Bayesian Statistics*. 3rd. Wiley, 2016 (cit. a p. 51).

[52] The Julia Developers. *Julia 1.11 Documentation*. Accessed: 2025-01-18. 2025. URL: https://docs.julialang.org/en/v1/ (cit. a p. 89).