POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Matematica

Tesi di Laurea

Inferenza per modelli ad effetti misti descritti da equazioni differenziali stocastiche

Relatore:
Enrico Bibbona

Correlatore:
Gianluca Mastrantonio

Candidato:
Martina Amongero

Anno Accademico 2018-2019

A nonna Maria che sa tutto di questi 5 anni, come fossero stati i suoi,
a nonno Piero che dice che studio troppo e lavoro troppo poco,
a nonna Lena che non crede molto nello studio ma mi ha supportato lo stesso, quasi fino alla fine,
ed a nonno Meco che avrebbe tanto voluto esserci...

Indice

Introduzione

L'inferenza statistica per modelli ad effetti misti è motivo di interesse in numerosi campi. Viene ora utilizzata, a titolo di esempio, la farmacocinetica per chiarire il metodo statistico, oggetto di studio di questa tesi.

Si consideri una popolazione di NP pazienti a cui venga somministrato un farmaco a bolo. Si supponga di voler analizzare la metabolizzazione del farmaco e di voler ricavare una legge che ne descriva la concentrazione nel sangue allo scorrere del tempo. Si hanno a disposizione, per ogni paziente, T rilevazioni cliniche che misurano la concentrazione di farmaco nel sangue in diversi istanti di tempo. Si supponga che le misurazioni distino Δ l'una dall'altra e che possano essere affette da errori di misurazione. Ogni paziente è inoltre caratterizzato da specifiche caratteristiche fisiche che influenzano la metabolizzazione del farmaco.

Sia X il processo che misura la concentrazione del farmaco. In termini matematici, il problema può essere modellato dalle seguenti equazioni:

$$Y_{i,j\Delta} = X_{i,j\Delta} + \epsilon_{i,j\Delta},\ j = 1, ..., T, \quad i = 1, ..., NP, \quad \text{con } \epsilon_{i,j\Delta} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma^2)$$

$$dX_{it} = a(X_{it}, t, \phi_i)dt + b(X_{it}, t, \phi_i, \gamma)dB_{it}, \quad X_{i0} = 0$$

$$\phi_i \overset{\text{i.i.d}}{\sim} p(\cdot, \beta)$$

con $(B_{it})_{i=1,..,NP}$ moti Browniani indipendenti che introducono correlazione tra le misurazioni, ϵ_{ij} che ha il significato di rumore bianco e ϕ_i parametri individuali. Il processo X non è osservato, i dati raccolti sono osservazioni di $\{Y_{i,\Delta j}\}$. Il vettore dei parametri di interesse statistico è $\theta = (\beta, \gamma, \sigma)$.

L'obiettivo di questo lavoro è stimare questi parametri dalle osservazioni disponibili.

Quando si studia un modello di questo tipo si possono verificare due diverse situazioni:

- la densità di transizione del processo X è nota,
- la densità di transizione del processo X non è nota.

Nel primo caso, in letteratura, sono state proposte alcune tecniche di inferenza su θ. Una delle metodologie proposte in ambito frequentista è l'algoritmo di Expectation Maximization (sigla inglese EM), una tecnica che si utilizza in caso di osservazioni mancanti. Sono state, in seguito, proposte tecniche che fanno uso di algoritmi di tipo Stochastic Approximation Expectation Maximization (SAEM), strettamente legati al metodo EM, precedentemente introdotto. In particolare in Donnet and Samson [2014] e in Kuhn and Lavielle [2004] si uniscono metodi di tipo SAEM con metodi di tipo Markov Chain Monte Carlo (MCMC) e Particle Markov Chain Monte Carlo (PMCMC).
Nel secondo caso, poiché la funzione di densità non è nota, si ricorre ad approssimazioni numeriche. L'utilizzo di tecniche di approssimazione richiede di possedere molti dati. Siccome nella maggior parte dei casi, raccogliere numerosi dati non è possibile, si ricorre a tecniche di data augmentation che permettono di infittire artificialmente i dati reali. In Donnet and Samson [2008] sono state proposte alcune tecniche di data augmentation da integrare agli algoritmi precedentemente citati. Queste tecniche sono state poi riprese e studiate in Erhardt [2019]. Da studi simulativi (Erhardt [2019]) è stata evidenziata una difficoltà critica nella stima di γ e σ.

Per semplicità è stato scelto di studiare un modello basato sul processo di Ornstein-Uhlenbeck (OU), che è un caso in cui la densità di transizione è in realtà nota ma è stata supposta, talvolta, non nota nel corso della tesi.

Gli obiettivi della tesi sono:

- analizzare nel dettaglio condizioni di applicabilità, punti di forza e di debolezza degli algoritmi SAEM-MCMC e SAEM-PMCMC, testare i suddetti algoritmi su dataset simulati,
- analizzare l'impatto che le tecniche di data augmentation hanno sulla stima dei parametri,
- chiarire quale sia la fonte del problema già riscontrato in Erhardt [2019].

Nello specifico la tesi si compone di due parti.

La prima parte (4 capitoli) fornisce tutte le definizioni e le nozioni utili a comprendere il lavoro svolto.
Il primo capitolo è dedicato al concetto di Integrale Stocastico, di Equazione Differenziale Stocastica (sigla inglese SDE) ed al processo di Ornstein Uhlenbeck.
Il secondo capitolo è dedicato ai metodi MCMC. Viene presentata l'idea generale alla base di questi metodi e vengono analizzati nel dettaglio due particolari algoritmi di tipo MCMC: il Metropolis-Hastrings e il Gibbs Sampler.
Il terzo capitolo è dedicato interamente agli algoritmi PMCMC. Per poter comprendere appieno questi algoritmi viene fatto un breve accenno alle tecniche di Importance Sampling, Importance Resampling e Sequential Monte Carlo (SMC). Si entra poi nel dettaglio del metodo Particle Indipendent Metropolis Hastings (PIMH) e del metodo Particle Marginal Metropolis Hastings (PMMH) e si presenta un loro esempio di applicazione.
Il quarto capitolo presenta l'algoritmo EM, la sua estensione SAEM e gli algoritmi SAEM-MCMC e SAEM-PMCMC.

La seconda parte della tesi (3 capitoli) contiene i risultati del lavoro di ricerca svolto. Viene esposto il problema studiato, presentata l'implementazione usata per gli algoritmi SAEM-MCMC, SAEM-PMCMC, ed infine i

problemi che si riscontrano nella loro implementazione pratica suggerendo alcuni possibili metodi per migliorarla. Viene poi introdotto il metodo di data augmentation e sono presentate alcune simulazioni numeriche.
Nello specifico il quinto capitolo riporta gli pseudo-codici relativi a SAEM-MCMC, SAEM-PMCMC. Vengono riportati nel dettaglio la loro implementazione ed i risultati ottenuti per il modello OU.
Il sesto capitolo presenta la tecnica di data augmentation proposta per il modello in analisi, le difficoltà riscontrate e alcune soluzioni proposte.
Infine l'ultimo capitolo confronta tutti i risultati ottenuti con i vari metodi implementati.

Parte I

Teoria & letteratura

Capitolo 1

Equazioni Differenziali Stocastiche

L'intera tesi poggia sul concetto di Equazione Differenziale Stocastica (SDE), pertanto questo primo capitolo è dedicato a fornire le nozioni di base utilizzate nei capitoli successivi.
Dopo aver brevemente introdotto la nozione di Integrale Stocastico, vengono definite le SDE mettendole a confronto con le Equazioni Differenziali Ordinarie (ODE). Viene chiarito il concetto di SDE secondo Itô, il concetto di soluzione e si analizzano le condizioni necessarie per la sua esistenza. Viene poi presentato il processo di Ornstein–Uhlenbeck. Si espongono, infine, metodi per la simulazione delle SDE e delle loro soluzioni.

1.1 Integrale Stocastico

Definizione 1.1. *Il processo stocastico* $C = (C_t, t \in [0,T])$ *è detto processo semplice se esiste un partizione*

$$0 = t_0 < ... < t_n = T,$$

ed esiste una sequenza $(Z_i, i = 1, ..., n)$ *di variabili casuali tali che*

$$C_t = \begin{cases} Z_n & con \quad t = T \\ Z_i & con \quad t_{i-1} \leq t < t_i, \quad i = 1, ..., n. \end{cases} \tag{1.1}$$

L'integrale di Itô per un processo semplice poggia sulla definizione di somma di Riemann-Stieltjes.

Definizione 1.2. *L'integrale di Itô, rispetto a un moto Browniano* B_s*, del processo semplice* C*, adattato alla filtrazione indotta da* $(B_s, s \geq 0)$*, su* $[0, T]$ *è dato da*

$$\int_0^T C_s \, dB_s = \sum_{i=1}^{n} C_{t_{i-1}}(B_{t_i} - B_{t_{i-1}}) = \sum_{i=1}^{n} Z_i \Delta_i B. \tag{1.2}$$

La definizione di integrale di Itô per un generico processo è strettamente legata alla definizione data per un processo semplice.

Viene ora accennata l'idea generale che collega le due definizioni. Una dimostrazione rigorosa può essere trovata nell'appendice A4 di Mikosch [1998].

Sia P un processo stocastico tale che:

1. P è adattato ad un moto Browniano sull'intervallo $I = [0, T]$,
2. $\int_0^T \mathbb{E}[P_s]^2 \, ds < \infty$,

allora esiste una successione $(C^{(n)})$ di processi semplici per cui

$$\int_0^T \mathbb{E}[P_s - C_s^{(n)}]^2 \, ds \to 0.$$

Poiché $C^{(n)}$ è un processo semplice è possibile valutarne l'integrale stocastico. Sia quindi $(I(C^{(n)}))$ la successione di integrali di Itô associata alla successione $C^{(n)}$ di processi semplici. E' possibile dimostrare l'esistenza di un processo limite $I(P)$ a cui la successione $(I(C^{(n)}))$ converge in media quadratica, ovvero:

$$E\left[\sup_{0 \leq t \leq T} [I_t(P) - I_t(C^{(n)})]^2\right] \to 0. \tag{1.3}$$

Si può ora definire l'integrale stocastico di Itô per un generico processo P che soddisfi le assunzioni fatte.

Definizione 1.3. *Si definisce integrale di Itô del processo P*

$$I_t(P) = \int_0^t P_s dB_s\, ds, \quad t \in [0,T]. \tag{1.4}$$

L'integrale di Itô gode delle seguenti proprietà:

- $I_t(P)$ è una martingala rispetto alla filtrazione naturale $\{F_t\}_{t\in[0,t]}$ indotta dal moto Browniano,
- l'integrale di Itô ha media nulla,
- l'integrale di Itô è lineare.

Definizione 1.4. *Si definisce processo di Itô un processo X per cui valga la rappresentazione*

$$X_t = X_0 + \int_0^t A_s^{(1)}\, ds \; + \int_0^t A_s^{(2)}\, dB_s \tag{1.5}$$

con $A_s^{(1)}$ ed $A_s^{(2)}$ processi stocastici adattati al moto Browniano.

Si può dimostrare che i processi $A_s^{(1)}$ ed $A_s^{(2)}$ sono unicamente determinati.

Lemma 1.5. (Formula di Itô)
Sia X un processo di Itô e sia $g(t,x)$ una funzione con derivate parziali seconde continue, allora vale:

(1.6)

$$g(t,X_t) - g(s,X_s) = \int_s^t [g_1(y,X_y) + A_y^{(1)} g_2(y,X_y) + \frac{1}{2}[A_y^{(2)}]^2 g_{22}(y,X_y)]\, dy + \\ + \int_s^t A_y^{(2)}[g_2(y,X_y)\, dB_y], \quad s < t.$$

1.2 Equazioni differenziali Stocastiche

Si consideri l'equazione differenziale deterministica

$$dx(t) = a(t, x(t)), \quad x(0) = x_0, \tag{1.7}$$

un'equazione differenziale stocastica può essere pensata come una versione affetta da rumore dell'equazione (1.7). Il rumore può essere introdotto in diversi modi, si pensi per esempio a randomizzare la condizione iniziale oppure ad aggiungere un termine casuale all'equazione (1.7).

Nel corso della tesi saranno utilizzate equazioni SDE del tipo:

$$dX(t) = \alpha(t, X_t)dt + \beta(t, X_t)dB_t, \quad X_0(\omega) = Y(\omega) \tag{1.8}$$

dove $\{B_t\}_{t\geq 0}$ indica una famiglia di moti Browniani e le funzioni α e β sono deterministiche. La funzione α è detta coefficiente di drift (o di deriva), la funzione β è detta coefficiente di diffusione.
Si osservi che in (1.8) la casualità è stata introdotta sia attraverso la condizione iniziale, sia attraverso il moto Browniano.
L'equazione (1.8) è giustificata formalmente secondo la seguente definizione.

Definizione 1.6. *Si definisce equazione stocastica differenziale secondo Itô la seguente equazione*

$$X_t = X_0 + \int_0^t \alpha(s, X_s)\, ds + \int_0^t \beta(s, X_s)\, dB_s, \quad 0 \leq t \leq T. \tag{1.9}$$

Un processo $X = \{X_t\}_{t\in[0,T]}$ *è detto soluzione* forte *per l'equazione* (1.8) *se valgono:*

- X *è adattato al moto Browniano,*
- *tutti gli integrali nell'equazione* (1.9) *sono ben definiti,*
- X *è una funzione di* $\{B_t\}_{t\geq 0}$, α *e* β.

Si osservi che (1.9) si compone di due integrali, il primo è un integrale secondo Riemann, il secondo è un integrale di Itô della forma (1.4). La soluzione X di (1.9) è un processo di Itô della forma (1.5), dove $A_s^{(1)} = \alpha(s, X_s)$ e $A_s^{(2)} = \beta(s, X_s)$. In alcuni casi, per opportune funzioni g, il Lemma 1.5 permette di risolvere l'equazione (1.8) e calcolare esplicitamente la soluzione X. Il processo di Ornstein-Uhlenbeck ricade in questa casistica.

1.2.1 Processo di Ornstein-Uhlenbeck (OU)

Si consideri l'equazione differenziale stocastica lineare

$$X_t = X_0 - \int_0^t \left(\frac{X_t}{\tau} - k\right) dt + \gamma \int_0^t dB_t, \quad X_0 = 0, \quad \tau > 0, \tag{1.10}$$

che può essere riscritta nella forma

$$dX_t = -\left(\frac{X_t}{\tau} - k\right) dt + \gamma dB_t. \tag{1.11}$$

Si osservi che il processo X ed il moto Browniano non appaiono insieme nell'integrale di Itô: questo tipo di equazioni sono dette equazioni con rumore additivo.
L'equazione (1.11) è detta processo di Ornstein-Uhlenbeck (OU). Il processo di OU è un processo di Itô che ha rappresentazione data dall'equazione (1.5), dove $A_s^{(1)} = -\left(\frac{X_t}{\tau} - k\right)$ ed $A_s^{(2)} = \gamma$.

Sfruttando il Lemma 1.5 e la trasformazione $g(t, X_t) = e^{\frac{t}{\tau}} X_t$, è possibile risolvere (1.11).
Sostituendo, infatti, la funzione g scelta in (1.6), si ottiene

(1.12)

$$\begin{aligned} e^{\frac{t}{\tau}} X_t - e^{\frac{s}{\tau}} X_s &= \int_s^t \frac{1}{\tau} X_y e^{\frac{y}{\tau}}\, dy + \int_s^t \left(-\frac{X_y}{\tau} + k\right) e^{\frac{y}{\tau}}\, dy \;+\; \int_s^t \gamma e^{\frac{y}{\tau}}\, dB_y \\ &= \int_s^t k e^{\frac{y}{\tau}}\, dy + \int_s^t \gamma e^{\frac{y}{\tau}}\, dB_y \\ &= k\tau (e^{\frac{t}{\tau}} - e^{\frac{s}{\tau}}) \;+\; \int_s^t \gamma e^{\frac{y}{\tau}}\, dB_y \end{aligned}$$

da cui si ricava

$$
\begin{aligned}
X_t &= e^{-\frac{t}{\tau}}\left[e^{\frac{s}{\tau}}X_s + k\tau(e^{\frac{t}{\tau}} - e^{\frac{s}{\tau}}) \; + \; \int_s^t \gamma e^{\frac{y}{\tau}}\, dB_y\right] \\
&= e^{-\frac{t-s}{\tau}}X_s + k\tau(1 - e^{-\frac{t-s}{\tau}}) + e^{-\frac{t}{\tau}}\int_s^t \gamma e^{\frac{y}{\tau}}\, dB_y \\
&= e^{-\frac{\Delta_{ts}}{\tau}}X_s + k\tau(1 - e^{-\frac{\Delta_{ts}}{\tau}}) + e^{-\frac{t}{\tau}}\int_s^t \gamma e^{\frac{y}{\tau}}\, dB_y \\
&= e^{-\frac{\Delta_{ts}}{\tau}}X_s + k\tau(1 - e^{-\frac{\Delta_{ts}}{\tau}}) + \eta_{ts}
\end{aligned}
\tag{1.13}
$$

con $\eta_{ts} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \frac{\gamma^2\tau(1-e^{-2\Delta_{ts}/\tau)}}{2})$.

Utilizzando le proprietà dell'integrale di Itô riportate precedentemente (Sezione 1.1) è possibile ricavare la densità di transizione del processo e la distribuzione della soluzione.

Proposizione 1.7. *La soluzione dell'equazione* (1.11) *soddisfa*

1. $\mathbb{E}[X_t]=0$,
2. $Var(X_t) = \frac{\gamma^2\tau(1-e^{-2t/\tau)}}{2}$
3. $Cov(X_t,\ X_s) = \frac{\tau\gamma^2}{2}\left(1 - e^{-\frac{2s}{\tau}}\right)e^{-\frac{\Delta_{st}}{\tau}}, \quad s < t$
4. *la densità di transizione del processo è*

$$X_{t+\Delta}|X_t \sim \mathcal{N}\left(\tau k + (X_t - \tau k)e^{-\Delta/\tau}, \frac{\gamma^2\tau(1 - e^{-2\Delta/\tau)}}{2}\right).$$

1.2.2 Soluzione Numerica

Le SDE che ammettono soluzione esplicita sono solo un piccolo gruppo: nella maggioranza dei casi non è possibile ottenere la soluzione esatta. In questi scenari si può ricorrere ad alcune tecniche che forniscono soluzioni numeriche della SDE.
Uno dei metodi numerici più usati è l'approssimazione di Eulero-Maruyama.

Sia data la SDE (1.8). Chiamiamo $X^{(n)} = \{X_t^{(n)}, \quad t \in [0,T]\}$ la sua soluzione numerica e $X = \{X_t, \quad t \in [0,T]\}$ la sua soluzione esatta. Sia $\tau_n : \quad 0 = t_0 < t_1 < ... < t_{n-1} < t_n = T$ una partizione dell'intervallo $[0,T]$ tale che $\delta_n = \max_{i=1,...,n}(t_i - t_{i-1})$, con $\delta_n \to 0$.

Definizione 1.8. *Il Metodo di Eulero-Maruyama approssima X con la formula ricorsiva*

$$X_{t+\delta_n}^{(n)} = X_t^{(n)} + \alpha(t,\ X_t^{(n)})\delta_n + \beta(t,\ X_t^{(n)})(B_{t+\delta_n} - B_t),$$

da cui segue che

$$X_{t+\delta_n}^{(n)} | X_t^{(n)} \sim \mathcal{N}(X_t^{(n)} + \alpha(t,\ X_t^{(n)})\delta_n,\ \beta(t,\ X_t^{(n)})^2\delta_n) \tag{1.14}$$

è una approssimazione della densità di transizione della soluzione della SDE, calcolata solo sui punti della partizione τ_n.

1.2.3 Eulero-Maruyama applicato al processo OU

Viene ora dato un esempio di applicazione del metodo di Eulero-Maruyama per il processo di Ornstein-Uhlenbeck definito nella Sezione 1.2.1. Si ricordi che per questo preciso processo si può ricavare anche la soluzione esatta, ma per propositi che saranno chiariti più avanti, si introduce comunque l'approssimazione di Eulero-Maruyama.

Si consideri il processo (1.11). Il metodo di Eulero-Maruyama applicato a questo processo restituisce, per Δ_t sufficientemente piccolo,

$$X_{t+\Delta_t} = X_t - \left(\frac{X_t}{\tau} - k\right)\Delta_t + \gamma\sqrt{\Delta_t}N, \quad N \sim \mathcal{N}(0,1),$$

da cui si ottiene una approssimazione per la densità di transizione

$$X_{t+\Delta_t} | X_t \sim \mathcal{N}\left(X_t - \left(\frac{X_t}{\tau} - k\right)\Delta_t, \gamma^2\Delta_t\right). \tag{1.15}$$

Il seguente codice è stato utilizzato per simulare alcune traiettorie del processo OU sia utilizzando l'approssimazione della densità di transizione (1.15) ottenuta con il metodo di Eulero-Maruyama sia utilizzando l'esatta densità di transizione del processo.

```
###Parametri#####
T=100
n=100
Delta=T/n
k=1
tau=1.8
gm=0.05

###Inserisco condizioni iniziali e definisco la SDE###
xo=0
Alpha<-function(x){return (k-x/tau)}
Beta<-function(x){return (gm)}

###Eulero###
eulero<-function(xo,n,Delta,a,b,vettoreB)
{    X=array(dim=n+1)
     X[1]=xo
     for(p in 2:(n+1))
     {
       X[p]=X[p-1]+a(X[p-1])*Delta+b(X[p-1])*(vettoreB[p]-vettoreB
    [p-1])

     }
    return (X)
}

##Soluzione esatta###
simulazioneEsatta<- function(Xo, to, Dt, N, parametri)
{
  X <- numeric(N+1)
  X[1] <- Xo
```

```
  for(i in 2:(N+1))
  { X[i] <- (rcdistOU(1, Dt, X[i-1], parametri))}
  return(X)
}


rcdistOU<-function(n,t,CI,parametri)
{ mean=parametri[1]*parametri[2]+(CI-(parametri[1]*parametri[2]))
    *(exp(-t/parametri[2]))
  var=(parametri[3]^2)*((1-exp(-2*t/parametri[2])))/2
  return (rnorm(n,mean,sqrt(var)))
}


###Simulazione con Eulero###
vettoreB=c(0,cumsum(sqrt(Delta)*rnorm(n)))
soluzione=eulero(xo,n,Delta,Alpha,Beta,vettoreB)
vettore=array(dim=(n))
vettore[1:(n)]=1*Delta
vettoreT=c(0,cumsum(vettore))
par(mfrow=c(1,2))
plot(vettoreT,soluzione,"l",col="black",ylab = 'tempoT',xlab = '
    soluzione Eulero-Maruyama',ylim = c(1,2))
ccol=c('blue','red','violet','yellow','green','brown')
for(seme in 1:6)
{ set.seed(seme)
  vettoreB=c(0,cumsum(sqrt(Delta)*rnorm(n)))
  soluzione=eulero(xo,n,Delta,Alpha,Beta,vettoreB)
  vettore=array(dim=(n))
  vettore[1:(n)]=1*Delta
  vettoreT=c(0,cumsum(vettore))
  lines(vettoreT,soluzione,col=ccol[seme])
}

###Simulazione con soluzione Esatta###
parametri=c(k,tau,gm)
soluzione=simulazioneEsatta(xo, 0, Delta, n, parametri)
plot(vettoreT,soluzione,type='l',col='black',,ylab = 'tempoT',
```

```
    xlab = 'soluzione Esatta',ylim = c(1,2))
ccol=c('blue','red','violet','yellow','green','brown')
for(seme in 1:6)
{ set.seed(seme)
  soluzione=simulazioneEsatta(xo, 0, Delta, n, parametri)
  lines(vettoreT,soluzione,col=ccol[seme])
}
```

In Figura 1.1 sono poste a confronto alcune traiettorie simulate utilizzando la struttura esatta del processo OU con alcune traiettorie ottenute utilizzando il metodo di Eulero-Maruyama.

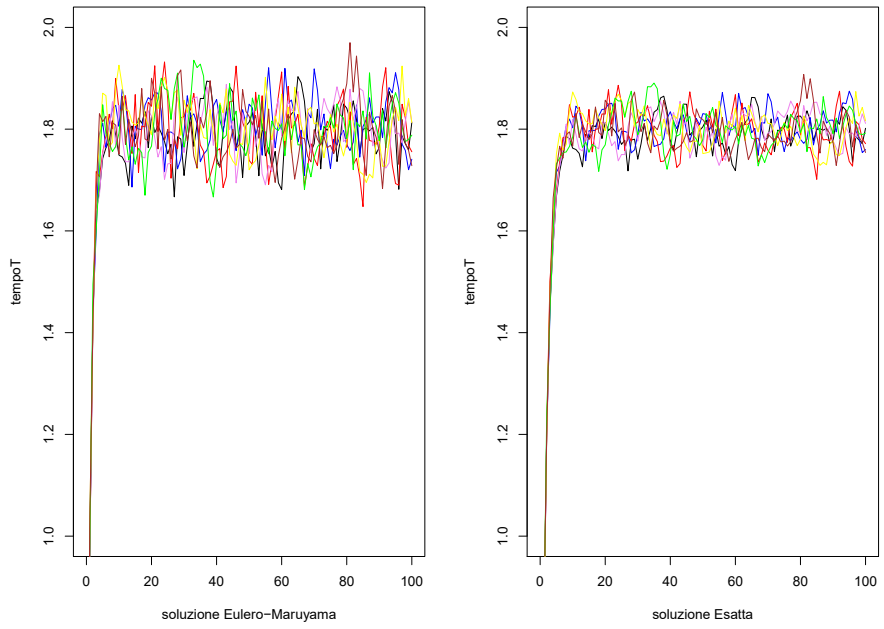

Figura 1.1: Simulazione di 10 traiettorie con il metodo di Eulero-Maruyama (sinistra) e con la densità esatta (destra).

Per ulteriori dettagli teorici sugli integrali stocastici, sulle SDE e sul metodo di Eulero-Maruyama si rimanda a Mikosch [1998] e Iacus [2009].

Capitolo 2

Markov Chain Monte Carlo

I metodi MCMC sono tra i metodi più utilizzati per campionare da distribuzioni di probabilità complesse. Questi metodi traggono vantaggio dall'utilizzo di apposite catene di Markov e dalle loro proprietà di stazionarietà.
In questo capitolo vengono analizzati nel dettaglio due algoritmi MCMC: l'algoritmo Metropolis-Hastings e l'algoritmo Gibbs Sampler.

In quanto segue le variabili aleatorie sono indicate con caratteri maiuscoli e le loro realizzazioni con caratteri minuscoli.
Sia $X = (X_1, ..., X_J)$ un vettore di variabili aleatorie di densità $f(\mathbf{x})$. Simulare dalla distribuzione di probabilità di X risulta piuttosto complicato: spesso X è composto da variabili aleatorie dipendenti o si conosce solamente la densità $\bar{f}(\mathbf{x})$, dove $f(\mathbf{x}) = \bar{f}(\mathbf{x})/B$, con B fattore di normalizzazione. In questi casi si ricorre a metodi di simulazione di tipo MCMC.
L'idea è quella di generare una catena di Markov che abbia come distribuzione stazionaria la distribuzione di probabilità di interesse. Per ottenere un campione dalla distribuzione considerata è quindi sufficiente simulare un numero adeguato di stati successivi della catena.

I metodi che vengono ora presentati sono due particolari algoritmi di tipo MCMC utilizzati per campionare da densità congiunte. Si osservi che, mentre il metodo Metropolis-Hastings può essere utilizzato anche per simulare da

densità univariate, il metodo Gibbs Sampler può essere applicato solo nel caso di densità multivariate.

2.1 Metropolis-Hastings

L'algoritmo Metropolis-Hastings è un algoritmo iterativo. Dopo aver inizializzato la variabile X con un opportuno valore $\mathrm{x}^{(0)}$, alla generica iterazione $i+1$, viene proposto un nuovo valore per X, indicato con $\mathbf{x}^*$, che può essere accettato o meno. Il valore corrente, accettato all'iterazione i-esima, viene denotato con $\mathbf{x}^{(i)}$.

L'algoritmo Metropolis-Hastings, ad ogni passo $i+1$, si compone quindi dei seguenti passi:

- proporre un nuovo valore x^* da una distribuzione $Q(\mathbf{x}^{(i)})$, di densità q. Si indica con $q(\mathbf{x}^{(i)}, \mathbf{x}^*)$ la probabilità di transizione dallo stato $\mathbf{x}^{(i)}$ allo stato $\mathbf{x}^*$;
- porre

$$\mathbf{x}^{(i+1)} = \begin{cases} \mathbf{x}^* & \text{con probabilità} \quad a(\mathbf{x}^{(i)}, \mathbf{x}^*), \\ \mathbf{x}^{(i)} & \text{altrimenti.} \end{cases} \tag{2.1}$$

Dove $a(\mathbf{x}^{(i)}, \mathbf{x}^*) = \min\left(1, \frac{\bar{f}(\mathbf{x}^*)q(\mathbf{x}^*, \mathbf{x}^{(i)})}{\bar{f}(\mathbf{x}^{(i)})q(\mathbf{x}^{(i)}, \mathbf{x}^*)}\right)$.

Si osservi che la matrice di transizione F, associata alla catena $\{\mathbf{x}^{(i)}\}_{i\geq 1}$ costruita, ha la seguente forma:

$$F(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = q(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})a(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}), \quad i \neq j \tag{2.2}$$

$$F(\mathbf{x}^{(i)}, \mathbf{x}^{(i)}) = q(\mathbf{x}^{(i)}, \mathbf{x}^{(i)}) + \sum_{k\neq i} q(\mathbf{x}^{(i)}, \mathbf{x}^{(k)})(1 - a(\mathbf{x}^{(i)}, \mathbf{x}^{(k)})). \tag{2.3}$$

Proposizione 2.1. *La catena generata con l'algoritmo di Metropolis-Hastings è reversibile ed ha distribuzione stazionaria* $f(\mathbf{x})$

Dimostrazione. Prima di dimostrare la proposizione 2.1, si richiama la definizione di stazionarietà per una catena di Markov.

Definizione 2.2. *Una catena di Markov, con matrice di transizione F, è reversibile ed ha distribuzione stazionaria $f(\mathbf{x})$ se vale*

$$f(\mathbf{x}^{(i)})F(\mathbf{x}^{(i)},\mathbf{x}^{(j)}) = f(\mathbf{x}^{(j)})F(\mathbf{x}^{(j)},\mathbf{x}^{(i)}) \quad i \neq j, \quad \forall i,j \tag{2.4}$$

Riscrivendo quindi l'equazione (2.4) per la specifica catena di interesse (equazioni (2.2) e (2.3)) si ottiene

$$f(\mathbf{x}^{(i)})q(\mathbf{x}^{(i)},\mathbf{x}^{(j)})a(\mathbf{x}^{(i)},\mathbf{x}^{(j)}) = f(\mathbf{x}^{(j)})q(\mathbf{x}^{(j)},\mathbf{x}^{(i)})a(\mathbf{x}^{(j)},\mathbf{x}^{(i)}) \tag{2.5}$$

da cui, ricordando che $f(\mathbf{x}) = \frac{\bar{f}(\mathbf{x})}{B}$, si ha

$$\bar{f}(\mathbf{x}^{(i)})q(\mathbf{x}^{(i)},\mathbf{x}^{(j)})a(\mathbf{x}^{(i)},\mathbf{x}^{(j)}) = \bar{f}(\mathbf{x}^{(j)})q(\mathbf{x}^{(j)},\mathbf{x}^{(i)})a(\mathbf{x}^{(j)},\mathbf{x}^{(i)}). \tag{2.6}$$

Si osservi ora che $a(\mathbf{x}^{(i)},\mathbf{x}^{(j)}) = \min\left(1, \frac{\bar{f}(\mathbf{x}^{(j)})q(\mathbf{x}^{(j)}),\mathbf{x}^{(i)})}{\bar{f}(\mathbf{x}^{(i)})q(\mathbf{x}^{(i)},\mathbf{x}^{(j)})}\right)$.
Quando $a(\mathbf{x}^{(i)},\mathbf{x}^{(j)}) = \left(\frac{\bar{f}(\mathbf{x}^{(j)})q(\mathbf{x}^{(j)},\mathbf{x}^{(i)})}{f(\mathbf{x}^{(i)})q(\mathbf{x}^{(i)},\mathbf{x}^{(j)})}\right)$ allora $a(\mathbf{x}^{(j)},\mathbf{x}^{(i)}) = 1$, e viceversa.

L'ultima osservazione, unita all'equazione (2.6) dà la tesi.

Il calcolo di $a(\mathbf{x}^{(i)},\mathbf{x}^{(j)})$ non richiede esplicitamente la conoscenza della costante B. □

Siccome la catena generata è strettamente legata all'inizializzazione $\mathrm{x}^{(0)}$, si utilizza una tecnica, che prende il nome di *burnin*, che consiste nello scartare le prima n iterazioni dell'algoritmo MCMC poiché strettamente dipendenti da $\mathrm{x}^{(0)}$.
Il vettore $(\mathbf{x}^{(i)},...,\mathbf{x}^{(N)})$, generato mediante le N iterazioni, è un campione dipendente dalla densità $f(\mathbf{x})$. Quando si sia interessati ad un campione *i.i.d.*, è possibile campionare solamente alcuni valori, piuttosto che selezionare l'intera sequenza. Questa tecnica prende il nome di *thinning*.

2.2 Gibbs sampler

Il Gibbs Sampler è una variante dell'algoritmo di Metropolis-Hastings. E' utilizzato per simulare un campione da una distribuzione multivariata basandosi solo sulla conoscenza e la capacità di simulare dalle singole distribuzioni condizionate. E' particolarmente utile quando la dimensione di X è piuttosto elevata.

Sia $X = (X_1, ..., X_J)$ il vettore aleatorio di interesse, sia $\mathbf{x}$ una sua realizzazione. Si indica nel seguito con x_j la componente j-esima del vettore $\mathbf{x}$ e con $\mathbf{x}_{-j}$=$(x_1, ..., x_{j-1}, x_{j+1}, ..., x_J)$ il vettore $\mathbf{x}$ privato della componente j-esima. Analogamente al metodo Metropolis-Hastings, $\mathbf{x}^{(i)}$ denota il valore accettato all'iterazione i-esima.
Siano note le densità condizionate $f(X_j|X_1, ..., X_{j-1}, X_{j+1}, ..., X_J)$ per ogni $j = 1, ..., J$.

Il metodo si compone dei seguenti passi:

- inizializzare la catena con un valore $\mathrm{x}^{(0)}$,
- all'iterazione $i+1$-esima, generare $\mathbf{x}^{(i+1)}$ da $\mathbf{x}^{(i)}$ mediante

$$\begin{aligned} & X_1^{(i+1)} \sim f(X_1|X_2^{(i)}, ..., X_J^{(i)}) \\ & X_2^{(i+1)} \sim f(X_2|X_1^{(i+1)}, X_3^{(i)} ..., X_J^{(i)}) \\ & ... \\ & X_J^{(i+1)} \sim f(X_J|X_1^{(i+1)}, ..., X_{J-1}^{(i+1)}) \end{aligned} \tag{2.7}$$

Proposizione 2.3. *Il Gibbs sampler è una variante dell'algoritmo Metropolis-Hastings in cui la densità utilizzata per proporre la j-esima variabile è la densità condizionata $f(x_j|\mathbf{x}_{-j})$. Risulta quindi che la probabilità di accettazione è pari ad uno: ad ogni iterazione il nuovo stato generato è accettato.*

Dimostrazione. Si consideri, alla generica iterazione $i+1$, la variabile $X_j^{(i)}$. L'algoritmo di Metropolis-Hastings prevede la scelta di una densità $q(\mathbf{x}^{(i)}, x_j^*)$

da cui proporre il nuovo valore x_j^*, da accettarsi con probabilità

$$a(\mathbf{x}^{(i)}, x_j^*) = \min\left(1, \frac{\bar{f}(x_j^*)q(x_j^*, \mathbf{x}^{(i)})}{\bar{f}(\mathbf{x}^{(i)})q(\mathbf{x}^{(i)}, x_j^*)}\right). \tag{2.8}$$

Sia ora $q(\mathbf{x}_j^{(i)}, x_j^*) = f(x_j^*|\mathbf{x}_{-j}^{(i)})$. L'equazione (2.8) diventa

$$a(\mathbf{x}^{(i)}, x_j^*) = \min\left(1, \frac{\bar{f}(x_j^*)f(\mathbf{x}_{-j}^{(i)}|\mathbf{x}_j^*)}{\bar{f}(\mathbf{x}_{-j}^{(i)})f(x_j^*|\mathbf{x}_{-j}^{(i)})}\right) = \min\left(1, \frac{f(\mathbf{x})}{f(\mathbf{x})}\right) = 1. \tag{2.9}$$

□

Da cui si ottiene che $x_j^{(i+1)} = x_j^*$ con probabilità 1.

Gli algoritmi presentati vengono talvolta usati congiuntamente.
Si consideri come esempio il caso in cui la dimensione di X sia molto elevata e non consenta di ottenere buoni risultati utilizzando il metodo Metropolis-Hastings. Si supponga inoltre che le densità condizionate non siano note. E' possibile allora utilizzare una strategia mista. L'algoritmo di Metropolis-Hastings è usato per simulare, all'interno dell'algoritmo Gibbs sampler, dalle densità condizionate, ottenendo infine un campione dalla densità congiunta.

Per ulteriori dettagli si rimanda a Iacus [2009], Robert et al. [2010], Wilkinson [2011].

Capitolo 3

Particle Markov Chain Monte Carlo

I metodi MCMC presentati nel Capitolo 2 sono tra gli algoritmi più usati per simulare da densità di probabilità complesse; essi godono della proprietà di stazionarietà delle catene di Markov. L'idea chiave dell'algoritmo è quella di costruire un'apposita catena di Markov reversibile che abbia come distribuzione di interesse la distribuzione stazionaria. E' quindi sufficiente generare un numero abbastanza elevato di stati successivi della catena per ottenere un sample dalla distribuzione studiata. Le proprietà della catena costruita garantiscono, a livello teorico, l'esistenza della distribuzione stazionaria. Tuttavia quando si passa al lato pratico e all'implementazione è necessario che la catena costruita permetta di raggiungere la distribuzione stazionaria con un numero adeguato di iterazioni, ovvero che la catena raggiunga il suo stato stazionario in un numero finito di passi. La performance dei metodi MCMC risulta quindi strettamente legata alla scelta della densità q e alla complessità della distribuzione di interesse f.

In letteratura sono stati proposti dei metodi per ottimizzare la scelta della distribuzione q utilizzata per simulare ai fini di migliorare le performances dei metodi MCMC.

In quanto segue vengono discussi, dopo aver posto le basi teoriche, i me-

todi Particle MCMC, proposti in Andrieu et al. [2010]. Questi metodi si basano su un algoritmo di tipo Sequential Monte Carlo (SMC).
Nello specifico vengono definiti Importance Sampling e Importance Resampling, utili alla comprensione dell'algoritmo SMC. Viene poi analizzato, mediante pseudo-codice, il metodo SMC. Infine è introdotto il concetto di metodo MCMC ideale e di PMCMC.

Vengono analizzati nel dettaglio due tipi di algoritmi PMCMC ed è presentato un esempio numerico. Quanto segue è strettamente basato sull'articolo Andrieu et al. [2010].

3.1 Sequential Monte Carlo

Prima di presentare nel dettagli il metodo SMC, vengono presentati alcuni concetti chiave utili alla sua comprensione.

3.1.1 Importance Sampling

Sia Z una variabile aleatoria continua e sia $f(x)$ la sua funzione di densità. Si supponga di voler calcolare $\mathbb{E}(g(X))$ per una qualche funzione $g(\cdot)$. Il problema si riconduce al calcolo dell'integrale

$$\mathbb{E}(g(X)) = \int_X f(X)g(X)\,dx$$

E' noto che, quando l'integrale non sia direttamente calcolabile, si può risolvere il problema mediante tecniche di integrazione Monte Carlo, simulando n realizzazioni della variabile X. L'integrale verrà approssimato con

$$\mathbb{E}(g(X)) \approx \frac{1}{n}\sum_{i=1}^{n} g(x_i).$$

La tecnica di importance sampling si rende necessaria quando non è possibile simulare dalla densità $f(\cdot)$. In tal caso si può ricorrere ad una variabile

aleatoria ausiliaria Y con densità $f^1(\cdot)$ che abbia stesso supporto di $f(\cdot)$, ottenendo

(3.1)
$$\mathbb{E}(g(X)) = \int_X f(X)g(X)\,dx \;=\; \int_X \frac{f(X)g(X)}{f^1(x)} f^1(x)\,dx \approx \frac{1}{n}\sum_{i=1}^{n} \frac{g(y_i)f(y_i)}{f^1(y_i)}.$$

Il rapporto $\frac{f(y_i)}{f^1(y_i)}$, $i = 1, ..., n$, rappresenta il peso attribuito alla particella y_i.

3.1.2 Importance Resampling

L'Importance Resampling è strettamente collegato all'Importance Sampling. Sia X la variabile di interesse di densità $f(\cdot)$, sia Y una variabile aleatoria con densità $f^1(\cdot)$ definita sullo stesso supporto di $f(\cdot)$. Sia infine $g(\cdot)$ una qualsiasi funzione. Sia $\mathbb{E}(g(X))$ la quantità da calcolare.

L'algoritmo può essere schematizzato come segue:

1. generare $\{y_i\}_{i=1}^n$ da $f^1(\cdot)$,
2. calcolare i pesi $w_i = \frac{f(y_i)}{f^1(y_i)}$ per ogni i e normalizzarli, ottenendo $\overline{w}_i$,
3. usando il vettore di probabilità $(\overline{w}_1, ..., \overline{w}_n)$, fare n estrazioni con reintroduzione $\{y_i\}_{i=1}^n$, ottenendo $\{z_i\}_{i=1}^n$,

Il campione $\{z_i\}_{i=1}^n$ è un campione approssimato da $f(\cdot)$ che può essere utilizzato in (3.1) per calcolare $\mathbb{E}(g(X))$.

3.1.3 SMC algoritmo

Si consideri il seguente problema, così definito:

- θ è un parametro incognito del sistema con densità a priori $p(\theta)$.

- sia $\{X_n; n \geq 1\}$ un processo di Markov.
  Sia $X_1 \sim \mu_0$ e $X_{n+1}|(X_n = x) \sim f_\theta(\cdot|x)$ la densità di transizione del processo. La presenza del parametro θ a pedice serve a sottolineare la dipendenza di f da questo parametro.

- Il processo $\{X_n; n \geq 1\}$ non è osservato direttamente, ma tramite un altro processo $\{Y_n; n \geq 1\}$.
  Si assume che le osservazioni siano condizionatamente indipendenti dato $\{X_n; n \geq 1\}$ e che valga per ogni $1 \leq n \leq m$

$$Y_n|(X_1, ..., X_n = x, ..., X_m) \sim g_\theta(\cdot|x)$$

Dato un numero di osservazioni $T \geq 1$ si vuole trovare la densità

$$p(\theta, x_1, ..., x_T|y_1, ..., y_T) = p(\theta, x_{1:T}|y_{1:T}). \tag{3.2}$$

Si osservi che vale

$$p(\theta, x_{1:T}|y_{1:T}) \propto p_\theta(x_{1:T}, y_{1:T})p(\theta),$$

Sotto le assunzioni fatte, $p_\theta(x_{1:T}, y_{1:T})$ può essere decomposta come

$$p_\theta(x_{1:T}, y_{1:T}) = \mu_0(x_1) \prod_{n=2}^{T} f_\theta(x_n|x_{n-1}) \prod_{n=1}^{T} g_\theta(y_n|x_n) \tag{3.3}$$

Uno scenario di questo tipo è detto Modello a Spazio degli Stati (sigla inglese SSM).
Poiché $p(\theta, x_{1:T}|y_{1:T})$ e $p_\theta(x_{1:T}, y_{1:T})$ spesso non hanno forma esplicita e l'inferenza per i parametri risulta piuttosto complessa. I metodi SMC sono una classe di algoritmi che approssimano le densità $p_\theta(x_{1:T}|y_{1:T})$ e $p_\theta(y_{1:T})$ al variare del valore M per un certo θ fissato. Nello specifico questi algoritmi approssimano sequenzialmente $p_\theta(x_1|y_1)$ e $p_\theta(y_1)$, poi $p_\theta(x_{1:2}|y_{1:2})$ e $p_\theta(y_{1:2})$ fino ad arrivare a $p_\theta(x_{1:T}|y_{1:T})$ e $p_\theta(y_{1:T})$.

Di seguito è riportato lo pseudo-codice della versione base dell'algoritmo SMC. Si indicheranno con $W_n = (W_n^1, ..., W_n^N)$ il vettore dei pesi normalizzati al passo n e con $\mathrm{M}(\cdot|W_n)$ la distribuzione di $\{1, ..., N\}$ con vettore di probabilità W_n.

procedure

for n=1 **do**

generare $X_1^k \sim q_\theta(\cdot|y_1), \quad \forall k = 1, ..., N$

calcolare i pesi $w_1(X_1^K)$ e i pesi normalizzati W_1^k come segue

$$w_1(X_1^k) = \frac{p_\theta(X_1^k, y_1)}{q_\theta(X_1^k, y_1)} = \frac{\mu_0(X_1^k) g_\theta(y_1|X_1^k)}{q_\theta(X_1^k, y_1)}$$

$$W_1^k = \frac{w_1(X_1^k)}{\sum_{k=1}^N w_1(X_1^k)}$$

end for

for n= 2:T **do**

generare $I_{n-1}^k \sim \mathrm{M}(\cdot|W_{n-1}), \forall k = 1, ..., N$

generare $X_n^k \sim q_\theta(\cdot|y_n, X_{n-1}^{I_{n-1}^k})$ e settare $X_{1:n}^k = (X_{1:n-1}^{I_{n-1}^k}, X_n^k)$

calcolare i pesi e i pesi normalizzati come segue

$$w_n(X_{1:n}^k) = \frac{p_\theta(X_{1:n}^k, y_{1:n})}{q_\theta(X_n^k|y_n, X_{n-1}^{I_{n-1}^k}) p_\theta(X_{n-1}^{I_{n-1}^k}, y_{1:n-1})} = \frac{f_\theta(X_n^k, X_{n-1}^{I_{n-1}^k}) g_\theta(y_n|X_n^k)}{q_\theta(X_n^k|y_n, X_{n-1}^{I_{n-1}^k})}$$

$$W_n^k = \frac{w_n^k(X_1^k)}{\sum_{k=1}^N w_n(X_1^k)}$$

end for

end procedure

Allo passo T l'algoritmo fornisce l'approssimazione

$$\hat{p}_\theta(dx_{1:T}|y_{1:T}) = \sum_{k=1}^N W_T^k \delta_{X_{1:T}^k}(dx_{1:T}), \tag{3.4}$$

$$\hat{p}_\theta(y_{1:T}) = \hat{p}_\theta(y_1) \prod_{n=2}^T \hat{p}_\theta(y_n|y_{1:n-1}), \tag{3.5}$$

dove

$$\hat{p}_\theta(y_n|y_{1:n-1}) = \frac{1}{N} \sum_{k=1}^N w_n(X_{1:n}^k) \tag{3.6}$$

Osservazione 3.1. *Per ottenere un campione da* $p_\theta(x_{1:T}|y_{1:T})$ *è sufficiente estrarre casualmente un indice* k *da* $\mathrm{M}(\cdot|W_T)$

3.2 PMCMC

Si supponga ora di voler fare inferenza sullo SSM utilizzando algoritmi di tipo MCMC. Si definiscono metodi MCMC *ideali*, i metodi MCMC che approssimano la densità $p(\theta, x_{1:T}|y_{1:T})$ facendo uso diretto di $p_\theta(x_{1:T}|y_{1:T})$. Nella realtà questi metodi non sono implementabili (a parte rare eccezioni) in maniera esatta. Esistono tuttavia delle versioni approssimate il cui obiettivo è riprodurre al meglio i metodi ideali. Questa classe di algoritmi è detta Particle MCMC. I metodi PMCMC fanno uso di un algoritmo SMC per approssimare $p_\theta(x_{1:T}|y_{1:T})$. La forza dei metodi PMCMC risiede nel fatto che non necessitano di ottenere dal metodo SMC un'approssimazione di $p_\theta(x_{1:T}, y_{1:T})$ ma è sufficiente un campione distribuito approssimativamente come $p_\theta(x_{1:T}, y_{1:T})$.

Proposizione 3.2. *I metodi PMCMC sono approssimazioni dei metodi ideali MCMC con distribuzioni di interesse* $p_\theta(x_{1:T}|y_{1:T})$ *o* $p(\theta, x_{1:T}|y_{1:T})$.
Per ogni numero $N \geq 1$ *di particelle usate nell'algoritmo SMC il metodo PMCMC lascia la densità di interesse invariata.*

3.2.1 PIMH

L'algoritmo Indipendent Metropolis Hastings (IMH) è un algoritmo di tipo Metropolis-Hastings che può essere utilizzato per la stima di $p_\theta(x_{1:T}|y_{1:T})$, con θ noto. Questo metodo prevede la scelta di una densità $q_\theta(x_{1:T}|y_{1:T})$ con la quale proporre, ad ogni iterazione, i valori $x^*_{1:T}$ che verranno poi accettati con probabilità

$$a(x_{,1:T}\,, x^*_{1:T}) = \min\left(1, \frac{p_\theta(x^*_{1:T}|y_{1:T})q_\theta(x_{1:T}|y_{1:T})}{p_\theta(x_{1:T}|y_{1:T})q_\theta(x^*_{1:T}|y_{1:T})}\right) \tag{3.7}$$

L'algoritmo ideale prevede che la densità usata per proporre sia $q_\theta(x_{1:T}|y_{1:T}) = p_\theta(x_{1:T}|y_{1:T})$. Questa scelta è chiaramente impraticabile nella maggior parte

dei casi. Si ricorre quindi all'approssimazione PIMH che utilizza $q_\theta(x_{1:T}|y_{1:T}) = \hat{p}_\theta(x_{1:T}|y_{1:T})$. L'algoritmo si compone come segue:

---

procedure

for i = 0 **do**

inizializzazione

usare un algoritmo SMC per approssimare $p_\theta(x_{1:T}|y_{1:T})$

con $\hat{p}_\theta(x_{1:T}|y_{1:T})$

generare $X_{1:T}^{(0)} \sim \hat{p}_\theta(\cdot|y_{1:T})$

sia $\hat{p}_\theta^{(0)}(y_{1:T})$ l'approssimazione ottenuta

end for

for i=1:Iterazioni **do**

usare un algoritmo SMC per approssimare $p_\theta(x_{1:T}|y_{1:T})$

con $\hat{p}_\theta(x_{1:T}|y_{1:T})$

generare $X_{1:T}^{*} \sim \hat{p}_\theta(x_{1:T}|y_{1:T})$

sia $\hat{p}_\theta^{*}(y_{1:T})$ l'approssimazione ottenuta

con probabilità

$$a(X_{1:T}^{(i-1)}, X_{1:T}^{*}) = \min\left(1, \frac{\hat{p}_\theta^{*}(y_{1:T})}{\hat{p}_\theta^{(i-1)}(y_{1:T})}\right)$$

settare $X_{1:T}^{(i)} = X_{1:T}^{*}$, $\hat{p}_\theta^{(i)}(y_{1:T}) = \hat{p}_\theta^{*}(y_{1:T})$

altrimenti, settare $X_{1:T}^{(i)} = X_{1:T}^{(i-1)}$, $\hat{p}_\theta^{(i)}(y_{1:T}) = \hat{p}_\theta^{(i-1)}(y_{1:T})$

end for

end procedure

Si osservi che il metodo PIMH implementato ha come densità target $\hat{p}_\theta(x_{1:T}|y_{1:T})$, così come il metodo SMC che lo compone. Il metodo PIMH non è stato pensato per migliorare il metodo SMC nella stima della densità di interesse, ma piuttosto per essere utilizzato con altre densità di transizione. Viene qui riportato nella sua forma più semplice al fine di presentarne la struttura.

Per studiare in maniera esaustiva questo algoritmo per applicazioni più

complesse, che saranno presentate in seguito, è stato riprodotto l'Esempio 3.1 presente nell'articolo Andrieu et al. [2010].

3.2.2 Esempio

Sia dato lo SSM seguente:

$$X_n = \frac{X_{n-1}}{2} + 25\frac{X_{n-1}}{1+X_{n-1}^2} + 8\cos(1.2n) + V_n, \tag{3.8}$$

$$Y_n = \frac{X_n^2}{20+W_n}, \tag{3.9}$$

con $X_1 \sim \mathcal{N}(0,5)$, $V_n \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,\sigma_v^2)$ e $W_n \overset{\text{i.i.d}}{\sim} \mathcal{N}(0,\sigma_w^2)$.

Siano Y_n le variabili osservate, X_n la variabili nascoste. Siccome il processo X è osservato solo attraverso il suo quadrato c'è incertezza sul segno di X.

Sia $\theta = (\sigma_v, \sigma_w)$ il vettore dei parametri.

E' stato implementato l'algoritmo PIMH per fare inferenza su $p_\theta(x_{1:T}|y_{1:T})$ utilizzando un dataset simulato da (3.8) e (3.9), considerando $\theta = (10, 10)$.

Le densità scelte per il metodo SMC sono $q_\theta(x_1) = \mu_0(x_1)$ e $q_\theta(x_n|y_n, x_{n-1}) = f_\theta(x_n|x_{n-1})$ per ogni $n = 2, ..., T$.

I parametri da scegliere sono il numero N di particelle ed il numero T di osservazioni da usare nell'algoritmo SMC . Per capire come questi valori influenzino la performance dell'algoritmo è stata studiata la variazione del tasso di accettazione al variare dei valori di T ed N. I risultati sono rappresentati in Figura 3.1

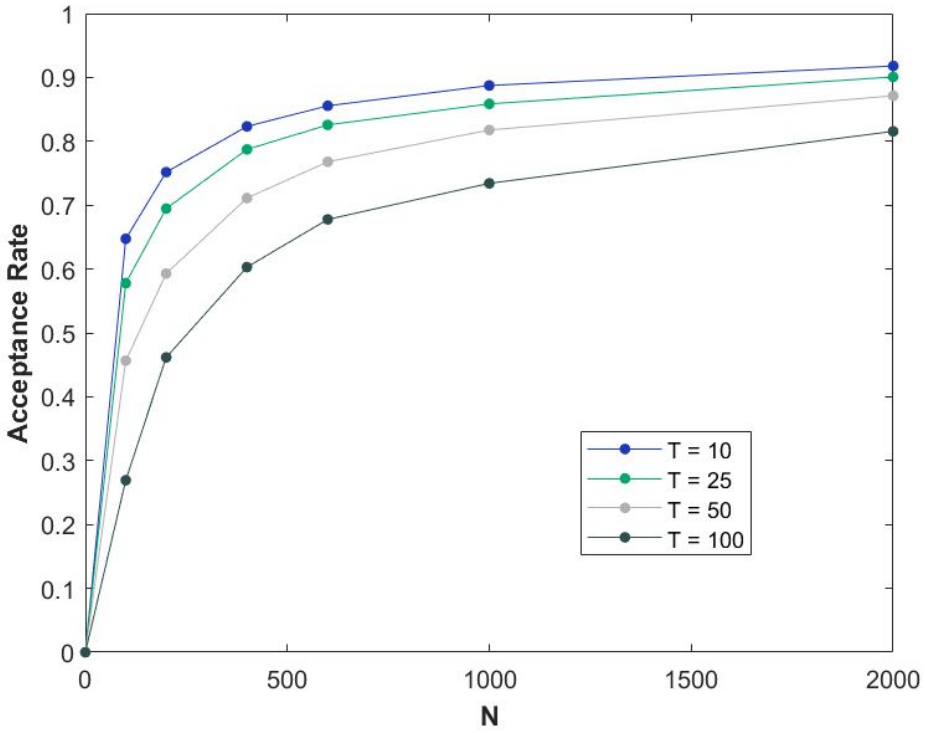

Figura 3.1: Tasso di accettazione al variare del numero di Particelle N e del numero T di osservazioni.

I risultati sono stati ottenuti con 10000 iterazioni dell'algoritmo PIMH.

Analizzando, per esempio, la curva che corrisponde a $T = 50$ si osserva che il valore di $N = 2000$ dà un tasso dell'85%, mentre $N = 200$ corrisponde ad un tasso del 60%. A livello computazionale, utilizzare $N = 200$ riduce di 10 volte la complessità dell'algoritmo, mantenendo un buon risultato in termini di rate di accettazione. Conviene quindi ridurre il numero di particelle usate, aumentando eventualmente il numero I di iterazioni PIMH.

3.2.3 PMMH

L'algoritmo Marginal Metropolis Hastings (MMH) è un algoritmo di tipo Metropolis-Hastings che può essere utilizzato per la stima di $p(\theta, x_{1:T}|y_{1:T})$. L'algortimo PMCMC associato prende il nome di PMMH.

Quando il parametri del modello non sono noti, ma sono anch'essi di interesse statistico, la densità di interesse diventa $p(\theta, x_{1:T}|y_{1:T})$. L'algoritmo MMH per la stima di $p(\theta, x_{1:T}|y_{1:T})$ prevede di scegliere una densità $q(\theta, x_{1:T}|y_{1:T})$ per proporre i valori simulati $(\theta^*, x^*_{1:T})$ che verranno poi accettati con probabilità

(3.10)
$$a((\theta, x_{1:T}), (\theta^*, x^*_{1:T})) = \min\left(1, \frac{p((\theta^*, x^*_{1:T})|y_{1:T})q((\theta^*, x^*_{1:T}), (\theta, x_{1:T}))}{p((\theta, x_{1:T})|y_{1:T})q((\theta, x_{1:T}), (\theta^*, x^*_{1:T}))}\right)$$

L'algoritmo ideale prevede di scegliere $q((\theta, x_{1:T}), (\theta^*, x^*_{1:T})) = q(\theta^*|\theta)p_{\theta^*}(x^*_{1:T}|y_{1:T})$. Questa scelta è chiaramente impraticabile nella maggior parte dei casi. Si ricorre quindi all'utilizzo del metodo PMMH, che sostituisce la funzione $p_\theta(x_{1:T}|y_{1:T})$ con una sua approssimazione $\hat{p}_\theta(x_{1:T}|y_{1:T})$ calcolata tramite l'utilizzo dell'algoritmo SMC.
Il metodo PMMH è così strutturato:

Anche questo algoritmo è stato implementato per risolvere l'esercizio precedentemente esposto (sezione 3.2.1), definito dalle equazioni (3.8) ed (3.9). E' stato utilizzato lo stesso dataset simulato ed è stata fatta inferenza anche su σ_v e σ_w, supposti non noti.

Sono state utilizzate $I = 50000$ iterazioni PMMH, $T = 50$ osservazioni, $M = 100$ particelle. Il vettore θ è stato inizializzato a $\theta_0 = (10, 10)$, il vero valore di θ, utilizzato per generare il dataset è $\theta = (\sqrt{2}, \sqrt{3})$. E' stata scelta inoltre una distribuzione Gamma Inversa di parametri $a = b = 1$ su σ_v^2 e σ_w^2, supposte indipendenti.

E' stato utilizzato un random walk per proporre $q(\theta^*|\theta^{(i-1)})$, per ogni iterazione i, di deviazione standard 0.015 e 0.08 rispettivamente su σ_v e σ_w, ovvero

(3.11)
$$\theta^*|\theta^{(i-1)} \sim \mathcal{N}\left(\begin{pmatrix} \theta_1^{(i-1)} \\ \theta_2^{(i-1)} \end{pmatrix}, \begin{pmatrix} 0.15^2 & 0 \\ 0 & 0.08^2 \end{pmatrix}\right)$$

Di seguito sono riportati i valori assunti da σ_w e σ_v ad ogni interazione. La linea rossa rappresenta i valori reali $\sigma_v = \sqrt{2}$ e $\sigma_w = \sqrt{3}$.

---

procedure

for i = 0 **do**

inizializzazione di $\theta^{(0)}$

usare un algoritmo SMC per approssimare $p_{\theta^{(0)}}(x_{1:T}|y_{1:T})$

con $\hat{p}_{\theta^{(0)}}(x_{1:T}|y_{1:T})$

generare $X_{1:T}^{(0)} \sim \hat{p}_{\theta^{(0)}}(\cdot|y_{1:T})$

sia $\hat{p}_{\theta^{(0)}}(y_{1:T})$ l'approssimazione ottenuta

end for

for i=1:Iterazioni **do**

simulare $\theta^* \sim q(\cdot|\theta^{(i-1)})$

usare un algoritmo SMC per approssimare $p_{\theta^*}(x_{1:T}|y_{1:T})$

con $\hat{p}_{\theta^*}(x_{1:T}|y_{1:T})$

generare $X_{1:T}^* \sim \hat{p}_{\theta^*}(\cdot|y_{1:T})$

sia $\hat{p}_{\theta^*}(y_{1:T})$ l'approssimazione ottenuta

con probabilità

$$a((\theta^{(i-1)}X_{1:T}^{(i-1)}),(\theta^*,X_{1:T}^*)) = \min\left(1, \frac{\hat{p}_{\theta^*}(y_{1:T})p(\theta^*)}{\hat{p}_{\theta^{(i-1)}}(y_{1:T})p(\theta^{(i-1)})}\frac{q(\theta^{(i-1)}|\theta^*)}{q(\theta^*|\theta^{(i-1)})}\right)$$

settare $\theta^{(i)} = \theta^*$, $X_{1:T}^{(i)} = X_{1:T}^*$, $\hat{p}_{\theta^{(i)}}(y_{1:T}) = \hat{p}_{\theta^*}(y_{1:T})$

altrimenti, settare $\theta(i) = \theta^{(i-1)}$, $X_{1:T}^{(i)} = X_{1:T}^{(i-1)}$,

$\hat{p}_{\theta^{(i)}}(y_{1:T}) = \hat{p}_{\theta^{(i-1)}}(y_{1:T})$

end for

end procedure

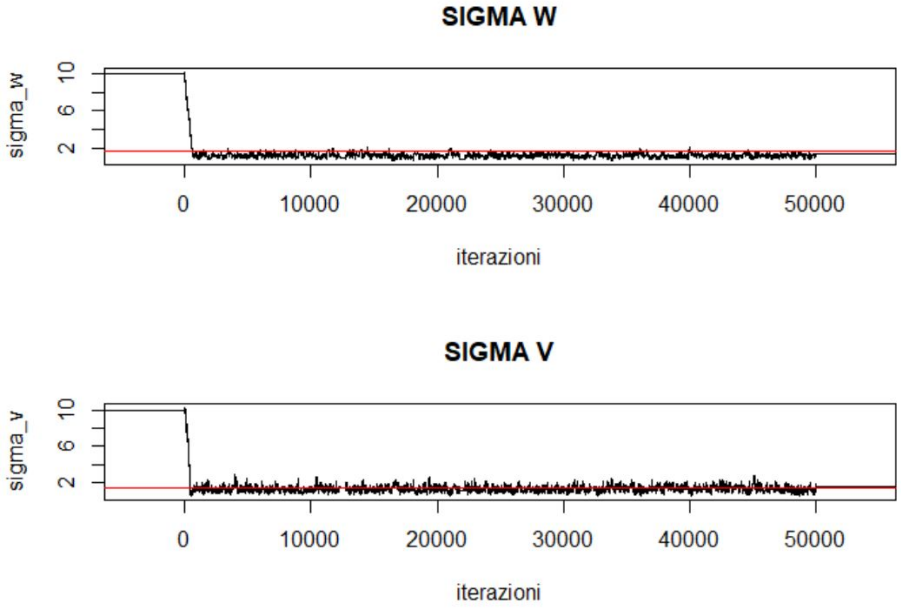

Figura 3.2: Convergenza ai valori veri.

Come si può osservare, dopo una fase iniziale di assestamento, l'algoritmo si stabilizza intorno ai valori $\theta = (\sqrt{2}, \sqrt{3})$.

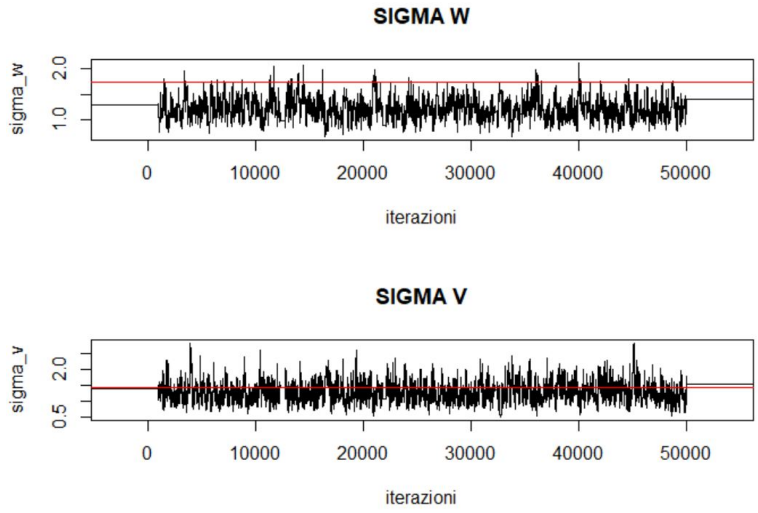

Figura 3.3: Convergenza ai valori veri con Burnin.

In questo caso i valori σ_w^*, e σ_v^* sono proposti ed accettati in coppia. Negli algoritmi che sono presentati più avanti i parametri di interesse non saranno legati in questo modo: ogni parametro è proposto ed accettato indipendentemente dagli altri parametri.
Inoltre verranno usate sia tecniche Metropolis Hasting sia tecniche Gibbs Sampler per parametri differenti all'interno dello stesso codice.

Capitolo 4

Algoritmo EM

Uno dei metodi più utilizzati in statistica per la stima dei parametri di interesse di una distribuzione è il metodo della massima verosimiglianza. Questo metodo prevede di massimizzare la funzione di verosimiglianza dei dati e ricavare il parametro che sia più in accordo coi dati raccolti. L'algoritmo Expextation Maximization (EM) fornisce un modo di procedere per calcolare lo stimatore di massima verosimiglianza quando si è in presenza di *dati incompleti*.

Il termine *dati incompleti* si riferisce alla presenza di più variabili, non tutte osservabili direttamente.

Siano y le variabili osservate e z le corrispettive variabili che possono essere osservate solo tramite le y. Si assuma quindi l'esistenza di una mappa tra le z e le y.

Nel seguito si indica con il nome *dati completi* il vettore $x = (z, y)$, con il nome *dati osservati* il vettore y e infine con il nome *dati non osservati* il vettore x. Eventuali parametri nascosti del sistema ricadono in quest'ultima categoria.

Sia $f(z, y|\xi) = f(x|\xi)$.una famiglia di densità di probabilità associate ai dati completi x, al variare del parametro ξ. Sia $g(y|\xi)$ una famiglia di densità di probabilità associate ai dati osservati y , al variare del parametro ξ. Sia $h(z|y, \xi)$ la distribuzione a posteriori dei dati non osservati, condizionata-

mente ai dati osservati y.

L'algoritmo iterativo EM massimizza la funzione $g(y|\xi)$ rispetto ad ξ facendo uso della funzione $f(x|\xi)$.

Di seguito è descritto nei dettagli l'algoritmo EM: in particolare si analizza il caso in cui le funzioni $f(x|\xi)$ siano appartenenti alla famiglia esponenziale. Segue poi una breve descrizione del caso generale.
Infine, si l'algoritmo EM è esteso alla versione SAEM, basata sull'approssimazione stocastica.

L'intera sezione è basata sull'articolo Dempster et al. [1977]

4.1 Famiglia Esponenziale

Supponiamo che la funzione $f(x|\xi)$ appartenga alla famiglia esponenziale, ovvero che esistano

- $\xi \in \mathbb{R}^{1\times r}$ vettore dei parametri ristretto ad un insieme convesso Ω sul quale $f(x|\xi)$ è una funzione di densità per ogni $\xi \in \Omega$
- $S(x) \in \mathbb{R}^{1\times r}$ vettore di statistiche sufficienti per x
- $a(\xi) = \int b(x) \exp(\xi S(x)^{\mathrm{T}})\, dx$ termine di normalizzazione

tali che valga la decomposizione

$$f(x|\xi) = \frac{b(x)\exp(\xi S(x)^{\mathrm{T}})}{a(\xi)}, \tag{4.1}$$

che equivale a

$$\log(f(x|\xi)) = \tilde{b}(x) + < \xi, S(x) > -\tilde{a}(\xi), \tag{4.2}$$

dove $\tilde{a}(\xi) = \log(a(\xi))$ e $\tilde{b}(x) = \log(b(x))$.
In termini di massimizzazione rispetto ad ξ, la formulazione (4.2) può essere semplificata in

$$\log(f(x|\xi)) = -\tilde{a}(\xi) + < \xi, S(x) > . \tag{4.3}$$

L'algoritmo EM si compone, per ogni iterazione i, di due passi; il primo prende il nome di *Expectation* step, il secondo di *Maximization* step, da cui il nome EM.
Sia $\xi^{(i)}$ l'approssimazione di ξ all'iterazione i-esima. I passi E ed M, per l'iterazione $i+1$, sono definiti come segue:

- *passo E*: si stima la statistica sufficiente $S(x)$ mediante

$$S^{(i)} = \mathbb{E}[S(x)|y, \xi^{(i)}], \tag{4.4}$$

- *passo M*: si determina $\xi^{(i+1)}$ risolvendo

$$\xi^{(i)} = \mathbb{E}[S(x)|y, \xi], \tag{4.5}$$

Proposizione 4.1. *Risolvere l'equazione* (4.5) *corrisponde a massimizzare*

$$\log(f(x|\xi)) = -\tilde{a}(\xi) + < \xi, S^{(i)} >$$

.

Dimostrazione. Nella dimostrazione viene omesso l'indice i relativo all'iterazione. Sia $L(\xi) = \log(g(y|\xi))$ la funzione da massimizzare.
Si definisca la densità condizionata dei dati x ai dai osservati y come

$$h(x|y, \xi) = \frac{f(x|\xi)}{g(y|\xi)}, \tag{4.6}$$

e si osservi che vale la rappresentazione

$$h(x|y, \xi) = \frac{b(x)\exp(\xi S(x)^{\mathrm{T}})}{a(\xi|y)}. \tag{4.7}$$

con $a(\xi|y) = \int b(x)\exp(\xi t(x)^{\mathrm{T}}))\,dx$.
Volendo quindi massimizzare $\log(g(y|\xi))$, si ottiene

$$\begin{aligned} L(\xi) &= \log(g(y|\xi)) \\ &= \log(f(x|\xi)) - \log(k(x|y,\xi)) \\ &= \tilde{b}(x) + < \xi, S(x) > -\tilde{a}(\xi) - (\tilde{b}(x) + < \xi, S(x) > -\tilde{a}(\xi|y)) \\ &= \tilde{a}(\xi|y) - \tilde{a}(\xi) \end{aligned} \tag{4.8}$$

e derivando

$$DL(\xi) = D[\tilde{a}(\xi|y) - \tilde{a}(\xi)] = -\mathbb{E}[S|\xi] + \mathbb{E}[S|y,\xi]. \tag{4.9}$$

Si osservi infatti che

$$D\tilde{a}(\xi|y) = D(\log a(\xi|y)) = D\int b(x)\exp(\xi t(x)^{\mathrm{T}})) = \mathbb{E}[S|y,\xi].$$

Analogamente si procede per $\tilde{a}(\xi)$.
Segue che le formule (4.4) ed (4.5) garantiscono che $DL(\xi) = 0$. □

4.2 Caso Generale

Si analizza ora il caso in cui $f(x|\xi)$ sia una generica funzione di densità. L'algoritmo EM, al passo $i+1$, si può riscrivere come

- *passo E*: si calcola $Q(\xi|\xi^{(i)})$ come

$$Q(\xi|\xi^{(i)}) = \mathbb{E}[\log(f(x|\xi))|y,\xi^{(i)}], \tag{4.10}$$

- *passo M*: si determina $\xi^{(i+1)}$ risolvendo

$$\xi^{(i+1)} = \max_{\xi} Q(\xi|\xi^{(i)}). \tag{4.11}$$

L'idea è quella di massimizzare $f(x|\xi)$, non disponibile o di difficile utilizzo, tramite il del suo valore atteso.
Nel caso esponenziale precedentemente analizzato si ha

(4.12)

$$\begin{aligned}\max_{\xi} Q(\xi|\xi^{(i)}) &= \max_{\xi} -\tilde{a}(x) + \mathbb{E}[b(x)|y,\xi] + \xi S^{(i)}(x)^{\mathrm{T}} \\ &= \max_{\xi} -\tilde{a}(x) + \xi S^{i}(x)^{\mathrm{T}} = \max_{\xi} -\tilde{a}(x) + < \xi, S^{(i)}(x) >,\end{aligned}$$

che riporta a quanto descritto nella Sezione precedente (proposizione (4.1)).

Per maggiori dettagli sul metodo EM si rimanda a Dempster et al. [1977].

4.3 SAEM

L'algoritmo EM appena analizzato prevede il calcolo di una attesa condizionata e la sua successiva massimizzazione.
Quando il calcolo dell'attesa risulta molto complesso o addirittura impraticabile, si ricorrere ad una versione dell'algoritmo EM basata sull'approssimazione stocastica che prende il nome di Stochastic Approximation Expectation Maximization (SAEM).
L'algoritmo SAEM prevede di dividere il passo E in due ulteriori passi: un primo passo S in cui si simulano i dati non osservati ed un secondo passo SA dove si approssima l'attesa condizionata con una tecnica di approssimazione stocastica.
Alla generica iterazione i:

- *passo S*: si simulano n realizzazioni dei dati non osservati $z_j^{(i)}$, $j = 1, ..., n$, da $h(z|y, \xi^i)$

- *passo SA*: si determina $\hat{Q}^{(i)}(\xi)$,come

$$\hat{Q}^{(i)}(\xi) = \hat{Q}^{(i-1)}(\xi) + \alpha_i \left(\frac{1}{n} \sum_{j=1}^{n} \log(f(x_j^{(i)}|\xi)) - \hat{Q}^{(i-1)}(\xi) \right) \tag{4.13}$$

  dove $\{\alpha_i, i \geq 1\}$ è un sequenza decrescente di numeri positivi e $x_j^{(i)} = (z_j^{(i)}, y)$.

- *passo M*: si determina $\xi^{(i)}$ risolvendo

$$\xi^{(i+1)} = \max_{\xi} \hat{Q}^{(i)}(\xi). \tag{4.14}$$

Per maggiori dettagli sull'algoritmo SAEM e sulle proprietà di convergenza si rimanda all'articolo Kuhn and Lavielle [2004].

4.3.1 SAEM-MCMC e SAEM-PMCMC

L'algoritmo SAEM richiede, al passo S, di simulare dalla densità di probabilità $h(x|y, \xi^{(i)})$. Simulare in maniera esatta le variabili non osservate dai dati raccolti è, nella maggior parte dei casi, impossibile. E' stato a tal proposito dimostrato in Kuhn and Lavielle [2005] che la fase di simulazione esatta può essere sostituita da una simulazione di tipo MCMC (Capitolo 2) mantenendo inalterata la convergenza dell'algoritmo SAEM. Questo tipo di algoritmo è indicato con il nome SAEM-MCMC ed è strutturato come segue:

- *passo S*: si simulano n realizzazioni dei dati non osservati $z_j^{(i)}$, $j = 1, ..., n$, da $h(z|y, \xi^{(i)})$ utilizzando un metodo MCMC con distribuzione stazionaria di interesse $h(z|y, \xi^{(i)})$

- *passo SA*: si determina $\hat{Q}^{(i)}(\xi)$,come

$$\hat{Q}^{(i)}(\xi) = \hat{Q}^{(i-1)}(\xi) + \alpha_i \left(\frac{1}{n} \sum_{j=1}^{n} \log(f(x_j^{(i)}|\xi)) - \hat{Q}^{(i-1)}(\xi) \right) \tag{4.15}$$

  dove $\{\alpha_i, i \geq 1\}$ è un sequenza decrescente di numeri positivi e $x_j^{(i)} = (z_j^{(i)}, y)$.

- *passo M*: si determina $\xi^{(i+1)}$ risolvendo

$$\xi^{(i+1)} = \max_{\xi} \hat{Q}(\xi). \tag{4.16}$$

La convergenza del metodo SAEM-MCMC è strettamente correlata alla convergenza del metodo MCMC utilizzato. Maggiori dettagli possono essere trovati in Kuhn and Lavielle [2005].
In Donnet and Samson [2014], con lo scopo di migliorare la convergenza dei metodi SAEM-MCMC, è stato proposto di utilizzare, in fase di simulazione, metodologie di tipo Particle MCMC (3) ed è stata studiata la convergenza dei metodi SAEM-PMCMC così ottenuti.

Parte II

Risultati

Motivazioni

Viene ora ripreso l'esempio, ispirato dalla farmacocinetica.

Si supponga di voler studiare un nuovo farmaco ed i suoi tempi di assorbimento. Supponiamo che il farmaco venga somministrato a bolo ad una popolazione composta da NP pazienti. In un intervallo di tempo $[0, T]$ dove $t = 0$ corrisponde al tempo in cui il farmaco viene somministrato, vengono effettuate delle analisi del sangue ai vari pazienti e viene misurata la concentrazione del farmaco nel sangue. Supponiamo che la distanza temporale tra un'osservazione e la successiva sia Δ e che le analisi vengano effettuate in contemporanea su tutti i pazienti.

Ci si aspetta che la concentrazione di farmaco nel sangue sia massima al tempo $t = 0$ e man mano decresca in maniera esponenziale per tutti i pazienti, come rappresentato il Figura 4.1.
La concentrazione di medicinale nel sangue può essere modellata come la soluzione di un'equazione differenziale ordinaria del tipo

$$dX(t) = -vX(t), \quad X(0) = C \tag{4.17}$$

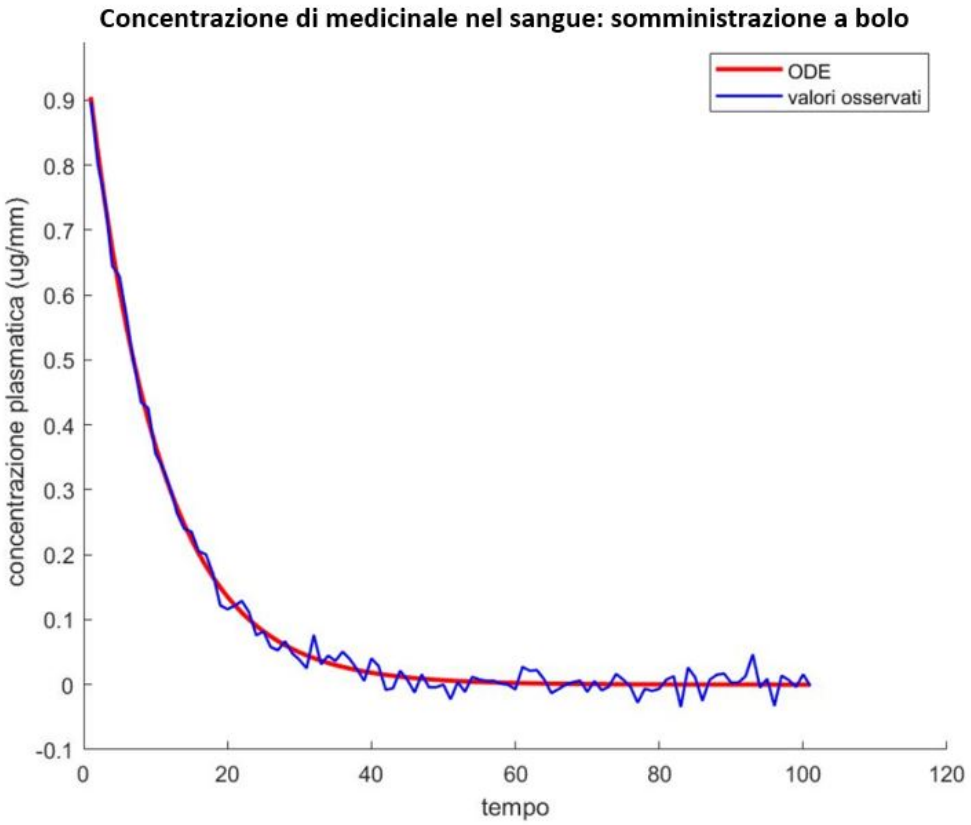

Figura 4.1: Concentrazione di farmaco nel sangue

Quello che si osserva, analizzando i dati raccolti, è un comportamento oscillatorio attorno alla soluzione di (4.17). Queste fluttuazioni possono essere interpretate come errori di misurazione e/o come fenomeni stocastici dovuti alla variabilità intrinseca del metabolismo.
Quando le osservazioni sono sufficientemente ravvicinate nel tempo, è possibile che fattori esterni, e non controllabili, vadano ad influire sulla metabolizzazione del farmaco, rendendo le osservazioni correlate. Si pensi per esempio alla digestione: da quando il paziente sotto osservazione inizia a mangiare a quando la digestione termina, vengono effettuate alcune misurazioni, queste misurazioni risentono dell'influenza dell'attività in corso. L'utilizzo di SDE permette di modellare questi fenomeni.

Le principali difficoltà di questo approccio sono:

- la concentrazione non è osservabile direttamente,

- trattandosi di misurazioni effettuate mediante analisi del sangue su pazienti, la distanza temporale Δ tra i dati raccolti è piuttosto elevata,

- ogni paziente è caratterizzato da parametri personali che influiscono sulla metabolizzazione del farmaco e che non sono osservabili,

- infine, spesso, potrebbe essere utile misurare anche l'interazione del farmaco con altri tessuti. Ogni tessuto può essere modellata mediante un compartimento, ma non tutti i compartimenti sono direttamente osservabili. E' stato scelto di non considerare questo problema nella tesi, in cui si analizzerà un unico compartimento.

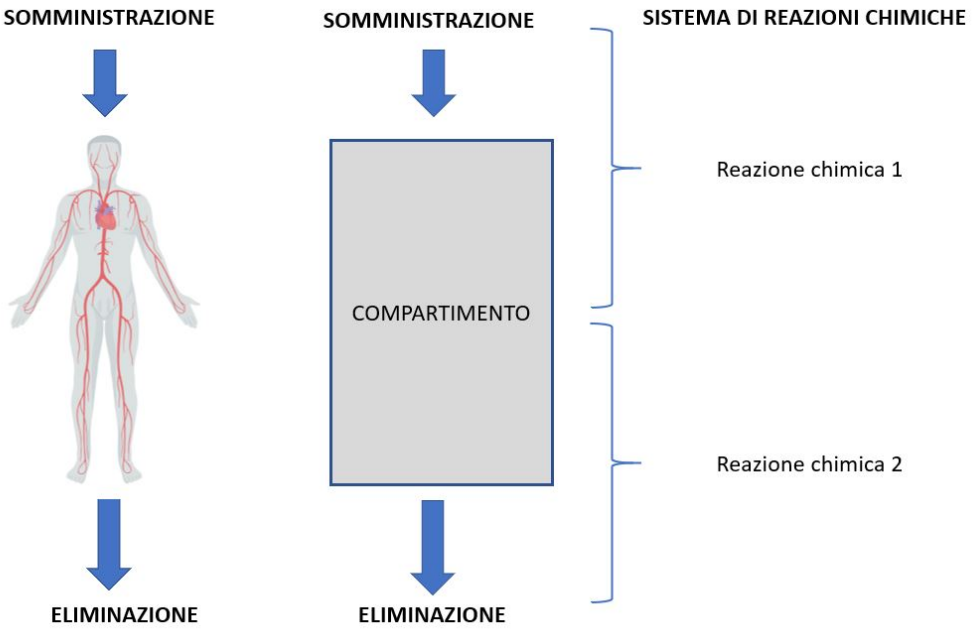

Figura 4.2: Modello uni-compartimentale.

Il problema di interesse può quindi essere tradotto, in termini matematici come segue:

(4.18)
$$Y_{i,j\Delta} = X_{i,j\Delta} + \epsilon_{i,j\Delta},\ j = 1, ..., T,\quad i = 1, ..., NP,\quad \text{con } \epsilon_{i,j\Delta} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma^2)$$

(4.19)
$$dX_{it} = a(X_{it}, t, \phi_i)dt + b(X_{it}, t, \phi_i, \gamma)dB_{it}, \quad X_{i0} = 0$$

(4.20)
$$\phi_i \overset{\text{i.i.d}}{\sim} p(\cdot, \beta)$$

con $(B_{it})_{i=1,..,N}$ moti Browniani indipendenti, ϵ_{ij} rumore bianco e ϕ_i parametri individuali. Dove il processo X di interesse è osservato mediante i dati raccolti $\{Y_{i,j\Delta}\}$.
Si osservi che $(\{X_{i,j\Delta}\}, \beta, \gamma, \sigma)$ costituiscono i valori non osservati e $\{Y_{i,j\Delta}\}$ i valori osservati.
Siano $\underline{Y} = \{Y_{i,j\Delta}\}_{i=1,...,NP}^{j=1,...,T}$, $Y_i = \{Y_{i,j\Delta}\}_{j=1,...,T}$, $\underline{X} = \{X_{i,j\Delta}\}_{i=1,...,NP}^{j=1,...,T}$, $X_i = \{X_{i,j\Delta}\}_{j=1,...,T}$, e $\Phi = \{\phi_i\}_{i=1,...,NP}$.
Il vettore che contiene i parametri di interesse statistico è $\theta = (\beta, \gamma, \sigma)$.

Per fare inferenza su θ si può usare il metodo EM, descritto precedentemente, massimizzando la funzione di verosimiglianza $\mathcal{L}(\underline{Y}; \theta)$. A tal scopo è necessario analizzare la struttura dei dati e fare alcune assunzioni sul modello.

Si supponga quindi che

1. le funzioni a e b siano sufficientemente regolari in modo da garantire l'esistenza e l'unicità della soluzione forte della SDE,

2. la SDE abbia densità di transizione nota ed esplicita $p(X_t|t-s, s, X_s, \theta)$.

Si osservi che $\mathcal{L}(\underline{Y}; \theta)$ non ha forma esplicita poiché non è possibile calcolare gli integrali nella formula (4.21) (nemmeno numericamente).

(4.21)
$$\mathcal{L}(\underline{Y}; \theta) = \int_{\underline{X}, \Phi} p(\underline{Y}, \underline{X}, \Phi; \theta)\, d\underline{X}\, d\Phi$$

L'assunzione 2 permette di decomporre la funzione di verosimiglianza dei dati completi, come

$$p(\underline{Y}, \underline{X}, \Phi; \theta) = p(\underline{Y}|\underline{X}; \sigma)p(\underline{X}|\Phi; \gamma)p(\Phi|\beta). \tag{4.22}$$

La tesi è stata ristretta allo studio di modelli con funzione di verosimiglianza Gaussiana e quindi appartenente alla famiglia esponenziale, fattorizzabile come (Equazione (4.2))

$$\log(p(\underline{Y}, \underline{X}, \Phi; \theta)) = \tilde{b}(x) + < \theta, S(\underline{Y}, \underline{X}, \Phi)) > -\tilde{a}(\theta), \tag{4.23}$$

dove $S(\underline{Y}, \underline{X}, \Phi)$ è un vettore di statistiche sufficienti.

Siccome, nella maggior parte dei casi, simulare i dati non osservati $(\underline{X}, \Phi)$ dai dati osservati $\underline{Y}$ è impraticabile, si ricorre all'approssimazione SAEM dell'algoritmo EM, unita a metodi di tipo MCMC e PMCMC per la simulazione.

Da qui in avanti si specifica la teoria introdotta nel caso del modello Ornstein–Uhlenbeck, definito dall'equazione (1.11):

(4.24)

$$Y_{i,j\Delta} = X_{i,j\Delta} + \epsilon_{i,j\Delta},\ j = 1, ..., T, \quad i = 1, ..., NP, \quad \text{con } \epsilon_{i,j\Delta} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma^2)$$

$$dX_{it} = -\left(\frac{X_{it}}{\tau_i} - k_i\right)dt + \gamma dB_{it}, \quad X_{i0} = 0 \tag{4.25}$$

con $(B_{it})_{i=1,..,NP}$ moti Browniani indipendenti, ϵ_{ij} rumore bianco e $\phi_i = (\log(\tau_i), k_i)$ parametri individuali, con distribuzione

$$\begin{pmatrix} \log(\tau_i) \\ k_i \end{pmatrix} \overset{\text{i.i.d}}{\sim} \mathcal{N}\left(\begin{pmatrix} \log(\tau) \\ k \end{pmatrix}, \begin{pmatrix} w_\tau^2 & 0 \\ 0 & w_k^2 \end{pmatrix}\right) \tag{4.26}$$

Il processo di OU è osservato solo mediante i dati raccolti $\{Y_{i,j\Delta}\}$, la cui distribuzione, condizionatamente ai valori $\{X_{i,j\Delta}\}$, è

$$Y_{i,j\Delta}|X_{i,j\Delta} \overset{\text{i.i.d}}{\sim} \mathcal{N}(X_{i,j\Delta}, \sigma^2), \quad \forall i, j.$$

Si osservi che la distribuzione $X_i = (X_{i,\Delta}, ..., X_{i,T\Delta})|\phi_i$, per il paziente i-esimo, è normale con parametri

$$\mu_i = \left(\tau_i k_i(e^{\frac{-\Delta}{\tau_i}}), ..., \tau_i k_i(e^{\frac{-T\Delta}{\tau_i}})\right)$$

$$\Sigma_{ij} = \left(\frac{\tau_i \gamma^2}{2} \left(1 - e^{\frac{-2\min(i\Delta, j\Delta)}{\tau_i}} \right) e^{\frac{-|i\Delta - j\Delta|}{\tau_i}} \right)_{1 \leq i,j \leq T}.$$

C'è indipendenza tra i valori $\{X_{i,j\Delta}\}_{j \geq 1}$ relativi a pazienti diversi.

Il vettore dei parametri di interesse statistico è $\theta = (\log \tau,\ k,\ w_\tau,\ w_k, \gamma, \sigma)$.

Come precedentemente chiarito, ogni qualvolta ci sia correlazione tra i dati, dovuta a fattori esterni e non controllabili, è giustificabile l'utilizzo di una SDE.
La scelta di utilizzare il modello di Ornstein-Ulhenbeck ha senso se

$$e^{\frac{-\Delta}{\tau_i}} >> 0 \text{ per ogni } i.$$

Si osservi infatti che se $e^{\frac{-\Delta}{\tau_i}} \sim 0$, allora le variabili $X_{i,j\Delta},\ j = 1, ..., T$, sono indipendenti tra loro.
Richiamando la soluzione del processo OU, data da (1.12), e sostituendo $e^{\frac{-\Delta}{\tau_i}} \simeq 0$ in (1.11), si ottiene

$$X_{i,j\Delta} \simeq \tau_i k_i + \eta_{i,j\Delta}, \text{ con } \eta_{i,j\Delta} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \frac{\gamma^2 \tau_i (1 - e^{-2\Delta/\tau_i})}{2}).$$

L'indipendenza della variabili $\eta_{i,j}$, garantisce l'indipendenza tra le variabili $\{X_{i,j\Delta}\}_{j \geq 1}$, per ogni j.

E' stato scelto un modello semplice e facilmente maneggiabile: tutte le densità di interesse sono note e permettono di trovare con facilità il valore di θ, senza la necessità di applicare tecniche statistiche complesse.
Sono state tuttavia applicate diverse tecniche, non di per sé necessarie, al fine di studiarne l'applicabilità, i punti di forza e di debolezza.
Avendo scelto un caso semplice, è stato possibile confrontare le performances delle tecniche più complesse, che introducono più approssimazione, con i metodi standard applicabili a questo modello e trarre quindi conclusioni sulla precisione dei metodi.

L'obiettivo è studiare metodi complessi, che andranno applicati a casi più generali in cui molte delle densità di probabilità coinvolte non siano note, applicandoli a un caso semplice, che permetta il confronto con metodi standard, con il fine ultimo di generalizzare.

Nella prima fase sono stati riprodotti risultati già presenti in letteratura in Donnet and Samson [2014], per tanto quanto segue è basato su questo articolo. Inizialmente si è implementato un algoritmo di tipo SAEM-MCMC ideale che, per questo particolare problema, è implementabile in maniera esatta. Si è poi analizzata l'approssimazione fornita dalla versione SAEM-PMCMC e si sono confrontati i risultati ottenuti.

Nella seconda fase sono state utilizzate alcune simulazioni con lo scopo di indagare le motivazioni alla base delle difficoltà riscontrate in Erhardt [2019]. In particolare è stato analizzato l'impatto della tecnica di data augmentation proposta. Infine, sono state utilizzate alcune metodologie alternative per la stima di γ.

Capitolo 5

Implementazione: SAEM-MCMC e SAEM-PMCMC

Richiamando le proprietà del processo OU (Proposizione 1.7), e le relative densità riportate nella sezione precedente, si analizzano ora alcuni algoritmi per fare inferenza sul vettore dei parametri θ.

Si osservi che non è possibile ricavare analiticamente il valore di θ che massimizzi $p(X_{i,j\Delta}, \Delta, j | X_{i,(j-1)\Delta}, \phi_i, \theta)$. Si ricorre quindi a metodi SAEM, come presentato in Donnet and Samson [2014].

L'intero capitolo è dedicato a chiarire le scelte implementative prese nel riprodurre i risultati di Donnet and Samson [2014] per il processo OU. Verranno analizzati gli pseudo-codici usati per l'algoritmo SAEM-MCMC e per l'algoritmo SAEM-PMCMC ed i relativi risultati ottenuti. Le simulazioni sono basate interamente sull'esempio 1 (Sezione 4.1) dell'articolo Donnet and Samson [2014], pertanto le scelte dei parametri utilizzati sono le medesime. L'obiettivo è quello di riprodurre i risultati presentati nell'articolo di Donnet and Samson [2014] per utilizzarli come punto di partenza per lo studio di nuove metodologie.
Per valutare gli algoritmi studiati e poterli comparare sono state fatte alcune simulazioni, con due differenti combinazioni di parametri per ognuno dei due metodi. Per la prima configurazione (C1) sono stati simulati 100 dataset

composti di 15 misurazioni per ognuno dei 20 pazienti. La distanza temporale che intercorre tra le misurazioni è $\Delta = 0.5$.
I valori utilizzati nella simulazione sono $\theta = (\log(\tau), k, w_\tau, w_k, \gamma, \sigma) = (0.6, 1, 0.1, 0.1, 0.05, 0.05)$.
L'inizializzazione usata è $\theta^{(0)} = (\log(10), 1.5, 0.5, 0.5, 0.25, 0.25)$.
Per la seconda configurazione (C2) ognuno dei 100 dataset si compone di 40 misurazioni per ognuno dei 20 pazienti considerati. La distanza temporale che intercorre tra le misurazioni è $\Delta = 2.5$.
I valori utilizzati nella simulazione sono $\theta = (\log(\tau), k, w_\tau, w_k, \gamma, \sigma) = (2.3, 1, 0.5, 0.5, 0.5, 0.5)$.
L'inizializzazione usata è $\theta^{(0)} = (2.83, 1.5, 0.5, 0.5, 5, 5)$. Si osservi che per C1 si ha $e^{\frac{-\Delta}{\tau}} = e^{-0.5/1} \simeq 0.6$ e per C2 $e^{\frac{-\Delta}{\tau}} = e^{-2.5/1} \simeq 0.08$. I valori scelti giustificano l'utilizzo del processo stocastico.

5.1 SAEM-MCMC

L'algoritmo SAEM-MCMC è stato implementato come segue.

---

procedure

for m=0 **do** Inizializzazione di $\hat{\theta}^0$ e di $\hat{S}^{(0)} = \mathbb{E}[S(\underline{Y}, \underline{X}, \Phi)|\underline{Y}, \hat{\theta}^{(0)}]$,

end for

for m = 1:Numero Iterazioni **do**

simulare la variabili non osservate $\underline{X}^{(m)} = (X_1^{(m)}, ..., X_{NP}^{(m)})$
e $\Phi^{(m)} = (\phi_1^{(m)}, ..., \phi_{NP}^{(m)})$
per ogni paziente mediante *metodo MCMC*
aggiornare $\hat{S}^{(m)}$ come

$$\hat{S}^{(m)} = \hat{S}^{(m-1)} + \alpha_m \left(S(\underline{Y}, \underline{X}^{(m)}, \Phi^{(m)}) - \hat{S}^{(m-1)} \right)$$

aggiornare $\hat{\theta}^{(m-1)}$ con $\hat{\theta}^{(m)} = \max_\theta(-\tilde{a}(\theta) + < \theta, \hat{S}^{(m)} >)$

end for

end procedure

Al passo $m+1$, dato il valore corrente di $\theta^{(m)}$, stimato alla precedente iterazione, la parte più delicata è la simulazione di $(\underline{X}^{(m+1)}, \Phi^{(m+1)})$.
Questa simulazione avviene mediante l'utilizzo di C iterazioni di un algoritmo MCMC.
Ad ogni iterazione $c+1$ del metodo MCMC, vengono aggiornati i valori di $\log^{(c)} = (\log(\tau_1^{(c)}), .., \log(\tau_{NP}^{(c)}))$, $k^{(c)} = (k_1^{(c)}, .., k_{NP}^{(c)})$ e di $\underline{X}^{(c)}$, accettati all'iterazione c-esima, come segue:

1. per ogni paziente i, i nuovi valori $\log(\tau_i^*)$ vengono proposti con una densità $q(\log(\tau_i^{(c)}), \log(\tau_i^*))$ di tipo random walk con media $\log(\tau_i^{(c)})$ e con deviazione standard pari a 0.05.
   Il nuovo valore $\log(\tau_i^*)$, condizionatamente al valore $\log(\tau_i^{(c)})$, si accetta con probabilità:
   (5.1)
   $$\begin{aligned} a(\log(\tau_i^{(c)}), \log(\tau_i^*)) &= \min\left(1,\ \frac{p_{\hat{\theta}^{(m)}}(\log(\tau_i^*)|X_i^{(c)}, Y_i, k^{(c)}) q(\log(\tau_i^*), \log(\tau_i^{(c)}))}{p_{\hat{\theta}^{(m)}}(\log(\tau_i^{(c)})|X_i^{(c)}, Y_i, k^{(c)}) q(\log(\tau_i^{(c)}), \log(\tau_i^*))}\right) \\ &= \min\left(1,\ \frac{p_{\hat{\theta}^{(m)}}(\log(\tau_i^*), X_i^{(c)}, Y_i, k^{(c)}) q(\log(\tau_i^*), \log(\tau_i^{(c)}))}{p_{\hat{\theta}^{(m)}}(\log(\tau_i^{(c)}), X_i^{(c)}, Y_i, k^{(c)}) q(\log(\tau_i^{(c)}), \log(\tau_i^*))}\right) \\ &= \min\left(1,\ \frac{p_{\hat{\theta}^{(m)}}(X_i^{(c)}|\log(\tau_i^*), k^{(c)}) p(\log(\tau_i^*))}{p_{\hat{\theta}^{(m)}}(X_i^{(c)}|\log(\tau_i^{(c)}), k^{(c)}) p(\log(\tau_i^{(c)}))}\right). \end{aligned}$$

   Si osservi che per ogni paziente i è stato utilizzato un metodo MCMC indipendente di tipo Metropolis-Hasting sul parametro individuale $\log(\tau_i)$.

2. Per i parametri k_i, per ogni paziente i, dopo aver calcolato la densità condizionata esatta $p_\theta(k_i|X_i, Y_i, \log(\tau_i))$, è stato utilizzato un aggiornamento di tipo Gibbs Sampler.
   Ricordando che c'è indipendenza tra i parametri $log(\tau_i)$ e k_i, si osservi che
   $$p_\theta(k_i|X_i, Y_i, \log(\tau_i))) \propto p_\theta(X_i|k) p_\theta(k)$$
   da cui si ottiene, per le proprietà di coniugazione delle variabili normali,

$k_i|X_i, Y_i \overset{\text{i.i.d}}{\sim} \mathcal{N}(\lambda_k, V_k^2)$, dove

$$V_k = \left(\frac{1}{w_k^2} + T \cdot \frac{(\tau_i - \tau_i e^{\frac{-\Delta}{\tau_i}})^2}{\sigma_\eta^2}\right)^{-1},$$

$$\lambda_k = V_k \cdot \left(\frac{k}{w_k^2} \frac{\sum_j (\tau_i - \tau_i e^{\frac{-\Delta}{\tau_i}})(X_{i,j\Delta} - X_{i,(j-1)\Delta} e^{\frac{-\Delta}{\tau_i}})}{\sigma_\eta^2}\right),$$

con $\sigma_\eta^2 = \frac{\gamma^2 \tau_i (1 - e^{\frac{-2\Delta}{\tau_i}})}{2}$.
Utilizzando il parametro $\log(\tau_i^{(c+1)})$ appena aggiornato, il vettore $X_i^{(c)}$ e il valore $\hat{\theta}^{(m)}$ nella densità calcolata, si simulano i valori $k^{(c+1)} = (k_1^{(c+1)}, ..., k_{NP}^{(c+1)})$.

3. Per il processo X è possibile utilizzare la densità condizionata esatta in un aggiornamento di tipo Gibbs Sampler.
Si aggiorna simulando da:
$X_{i,\Delta}^{(c+1)}|(X_0, \log(\tau_i^{(c+1)}), k_i^{(c+1)}, \hat{\theta}^{(m)}) \overset{\text{i.i.d}}{\sim} \mathcal{N}(\lambda_{i,1}, v_i)$
$X_{i,2\Delta}^{(c+1)}|(X_{i,\Delta}^{(c+1)}, \log(\tau_i^{(c+1)}), k^{(c+1)}, \hat{\theta}^{(m)}) \overset{\text{i.i.d}}{\sim} \mathcal{N}(\lambda_{i,2}, v_i)$
$\vdots$
$X_{i,T\Delta}^{(c+1)}|(X_{i,(T-1)\Delta}^{(c+1)}, \log(\tau_i^{(c+1)}), k_i^{(c+1)}, \hat{\theta}^{(m)}) \overset{\text{i.i.d}}{\sim} \mathcal{N}(\lambda_{i,T}, v_i)$
dove

$$\lambda_{i,j} = X_{i,\Delta(j-1)} e^{\frac{-\Delta}{\tau_i^{(c+1)}}} + k_i^{(c+1)} \tau_i^{(c+1)} (1 - e^{\frac{-\Delta}{\tau_i^{(c+1)}}})$$

$$\text{e } v_i = \frac{\gamma^{(m)2} \tau_i^{(c+1)} (1 - e^{\frac{-2\Delta}{\tau_i^{(c+1)}}})}{2}$$

.

Quindi, all'iterazione c+1 dell'algoritmo MCMC, si ottengono i valori $\log^{(c+1)} = (\log(\tau_1^{(c+1)}), ..., \log(\tau_{NP}^{(c+1)}))$, $k^{(c+1)} = (k_1^{(c+1)}, ..., k_{NP}^{(c+1)})$ e $X^{(c+1)}$.

I valori ottenuti all'ultima iterazione C costituiranno le approssimazioni $\log(\tau^{(m+1)})$, $k^{(m+1)}$ e $X^{(m+1)}$.

Questi valori verranno, inoltre, utilizzati al successivo passo $m+2$ per inizializzare il metodo MCMC.

Sono riportate di seguito le statistiche sufficienti utilizzate. L'approssimazione della statistica S_h all'iterazione $m+1$ è stata denotata con $\hat{S}_h^{(m+1)}$, $h = 1, 2, 3, 4$.

(5.2)

$$\hat{S}_1^{(m+1)}(\underline{Y}, \underline{X}, \Phi) = \sum_{i=1}^{NP} \phi_i^{(m+1)},$$

$$\hat{S}_2^{(m+1)}(\underline{Y}, \underline{X}, \Phi) = \sum_{i=1}^{NP} \phi_i^{(m+1)^{\mathrm{T}}} \phi_i^{(m+1)},$$

$$\hat{S}_3^{(m+1)}(\underline{Y}, \underline{X}, \Phi) = \sum_{i=1}^{NP} (Y_i - X_i^{(m+1)})(Y_i - X_i^{(m+1)})^{\mathrm{T}},$$

$$\hat{S}_4^{(m+1)}(\underline{Y}, \underline{X}, \Phi) = \sum_{i=1}^{NP} \sum_{j=1}^{\mathrm{T}} \frac{\left(X_{i,j\Delta}^{(m+1)} - X_{i,(j-1)\Delta}^{(m+1)} e^{\frac{-\Delta}{\tau_i^{(m+1)}}} - k_i^{(m+1)} \tau_i^{(m+1)} (1 - e^{\frac{-\Delta}{\tau_i^{(m+1)}}}) \right)^2}{\frac{\tau_i^{(m+1)}}{2} (1 - e^{\frac{-2\Delta}{\tau_i^{(m+1)}}})}$$

da cui si ottengono le stime per $\mu = (log(\tau), k)$, $\Omega = \mathrm{diag}(w_\tau, w_k)$, σ, e γ, come

(5.3)

$$\begin{aligned}
\hat{\mu}^{(m+1)} &= \frac{1}{NP} \hat{S}_1^{(m+1)}, \\
\hat{\Omega}^{(m+1)} &= \frac{1}{NP} \hat{S}_2^{(m+1)} - \frac{1}{NP^2} (\hat{S_1^{(m+1)}})^{\mathrm{T}} \hat{S}_1^{(m+1)} \\
\hat{\sigma}^{(m+1)} &= \sqrt{\frac{\hat{S}_3^{(m+1)}}{NP \cdot T}}, \\
\hat{\gamma}^{(m+1)} &= \sqrt{\frac{\hat{S}_4^{(m+1)}}{NP \cdot T}}
\end{aligned}$$

Sono riportati di seguito i valori ottenuti con i parametri elencati precedentemente:

		$\log(\tau)$	k	w_τ	w_k	γ	σ
Inizializzazione C1		log(10)	1.5	0.5	0.5	0.25	0.25
Valori veri C1		0.6	1.0	0.1	0.1	0.05	0.05
SAEM - MCMC C1	mean	0.578	1.018	0.091	0.098	0.051	0.050
	sd	0.032	0.029	0.025	0.024	0.005	0.003
Inizializzazione C2		2.83	1.5	0.5	0.5	5	5
Valori veri C2		2.3	1.0	0.5	0.5	0.5	0.5
SAEM - MCMC C2	mean	2.29	1.0	0.465	0.479	0.505	0.493
	sd	0.143	0.123	0.113	0.088	0.033	0.054

Tabella 5.1: Stime dei parametri per il modello OU ottenute con l'algoritmo SAEM-MCMC

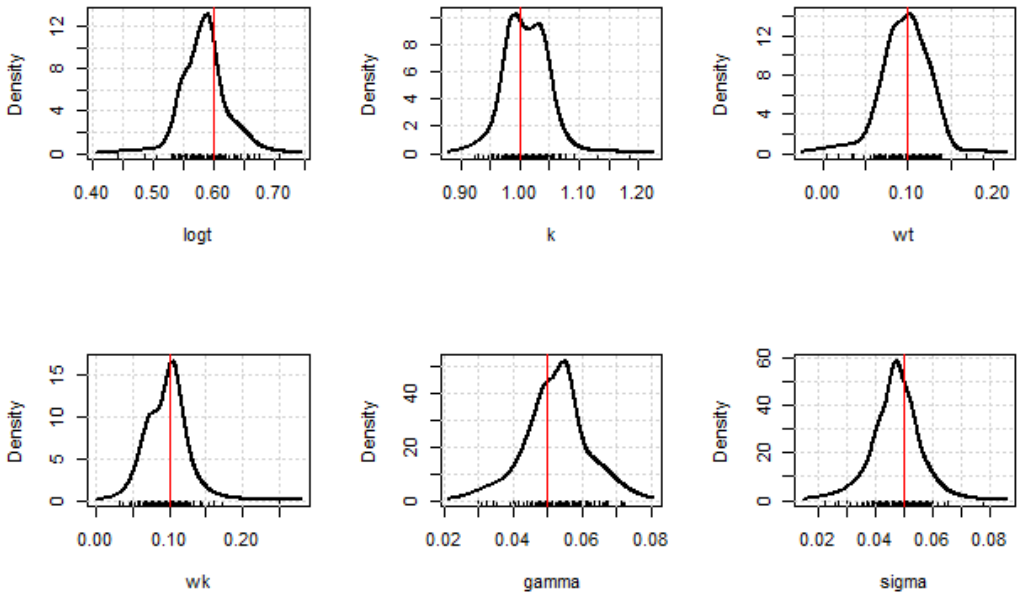

Figura 5.1: Distribuzione degli stimatori dei parametri sulle 100 simulazioni SAEM-MCMC (C1). Le distribuzioni per la configurazione C2 sono analoghe.

5.2 SAEM-PMCMC

Il processo OU analizzato risulta di particolare interesse in quanto permette di applicare il metodo SAEM-MCMC ideale, così come presentato nella Sezione 5.1, e confrontarlo con la versione approssimata che viene ora analizzata nel dettaglio.
Lo pseudo-codice relativo al metodo SAEM-PMCMC è il seguente.

---

procedure

for m=0 **do** Inizializzazione di $\hat{\theta}^0$ e di $\hat{S}^{(0)} = \mathbb{E}[S(\underline{Y}, \underline{X}, \Phi)|\underline{Y}, \hat{\theta}^{(0)}]$,

end for

for m = 1:Numero Iterazioni **do**

simulare la variabili non osservate $\underline{X}^{(m)} = (X_1^{(m)}, ..., X_{NP}^{(m)})$
e $\Phi^{(m)} = (\phi_1^{(m)}, ..., \phi_{NP}^{(m)})$
per ogni paziente mediante *metodo PMCMC*
aggiornare $\hat{S}^{(m)}$ come

$$\hat{S}^{(m)} = \hat{S}^{(m-1)} + \alpha_m \left(S(\underline{Y}, \underline{X}^{(m)}, \Phi^{(m)}) - \hat{S}^{(m-1)} \right)$$

aggiornare $\hat{\theta}^{(m-1)}$ con $\hat{\theta}^{(m)} = \max_\theta(-\tilde{a}(\theta) + < \theta, \hat{S}^{(m)} >)$

end for

end procedure

Al passo $m+1$, dato il valore corrente di $\hat{\theta}^{(m)}$, stimato alla precedente iterazione, la simulazione di $(\underline{X}^{(m+1)}, \Phi^{(m)})$ può essere effettuata con C iterazioni di un metodo PMCMC.
Ad ogni iterazione c+1 vengono aggiornati i valori di $\log^{(c)} = (\log(\tau_1^{(c)}), .., \log(\tau_{NP}^{(c)}))$, $k^{(c)} = (k_1^{(c)}, .., k_{NP}^{(c)})$ e di $\underline{X}^{(c)}$, accettati all'iterazione c-esima, come segue:

- per proporre i nuovi valori $\log(\tau_i^*)$ e k_i^*, per ogni i, si usano due densità $q(\log(\tau_i^{(c)}), \log(\tau_i^*))$ e $q(k_i^{(c)}, k_i^*)$ di tipo random walk di medie rispettivamente $\log(\tau_i^{(c)})$ e $k_i^{(c)}$ e deviazione standard 0.1.

Vengono quindi proposti i vettori $\log(\tau^*) = (\log(\tau_1^*), ..., \log(\tau_{NP}^*))$ e $k^* = (k_1^*, ..., k_{NP}^*)$.

- Per simulare il processo X_i, per ogni i, si usa un algoritmo SMC con K particelle e distribuzione target $p(X_i|Y_i, \phi_i^*, \hat{\theta}^{(m)})$; tramite questo algoritmo si calcola la densità $\hat{p}(\underline{Y}_i|\log(\tau_i^*), k_i^*, \hat{\theta}^{(m)})$, approssimazione di $p(Y_i|\log(\tau_i^*), k_i^*, \hat{\theta}^{(m)})$.
  Si propongono le traiettorie X_i^* da $\hat{p}(X_i|Y_i, \phi_i^*, \hat{\theta}^{(m)})$.

- I valori $(\log(\tau_i^*), k_i^*, X_i^*)$, per ogni i, sono accettati con probabilità

$$a((\log^{(c)}(\tau_i), k_i^{(c)}, X_i^{(c)}), (\log(\tau_i^*), k_i^*, X_i^*)) = \min\left(1, \frac{\hat{p}(Y_i|\log(\tau_i^*), k_i^*, \hat{\theta}^{(m)})p(\log(\tau_i^*)|\hat{\theta}^{(m)})p(k_i^*|\hat{\theta}^{(m)})}{\hat{p}(Y_i|\log(\tau_i^{(c)}), k_i^{(c)}, \hat{\theta}^{(m)})p(\log(\tau_i^{(c)})|\hat{\theta}^{(m)})p(k_i^{(c)}|\hat{\theta}^{(m)})}\right) \tag{5.4}$$

- $\log(\tau^{(c+1)})$, $k^{(c+1)}$ e $\underline{X}^{(c+1)}$ vengono aggiornati opportunamente.

Dopo C iterazioni PMCMC, strutturate come descritto, si ottengono le simulazioni di $\underline{X}^{(m+1)} = (X_1^{(m+1)}, ..., X_{NP}^{(m+1)})$ e $\Phi^{(m+1)} = (\phi_1^{(m+1)}, ..., \phi_{NP}^{(m+1)})$ da utilizzare all'interno del metodo SAEM.

Per poter confrontare i risultati ottenuti con SAEM-MCMC e con SAEM-PMCMC sono stati usate le stesse configurazioni di parametri, esposte precedentemente. Per entrambe le configurazioni il metodo SMC è stato implementato usando $N = 50$ particelle. Sono riportate di seguito le stime ottenute:

		$\log(\tau)$	k	w_τ	w_k	γ	σ
Inizializzazione C1		log(10)	1.5	0.5	0.5	0.25	0.25
Valori veri		0.6	1.0	0.1	0.1	0.05	0.05
SAEM - PMMH C1	mean	0.600	0.997	0.092	0.092	0.050	0.050
	sd	0.032	0.027	0.026	0.022	0.005	0.002
Inizializzazione		2.83	1.5	0.5	0.5	5	5
Valori veri C2		2.3	1.0	0.5	0.5	0.5	0.5
SAEM - PMMH C2	mean	2.306	0.983	0.471	0.483	0.500	0.500
	sd	0.143	0.116	0.109	0.085	0.032	0.047

Tabella 5.2: Stime dei parametri per il modello OU ottenute con l'algoritmo SAEM-PMCMC

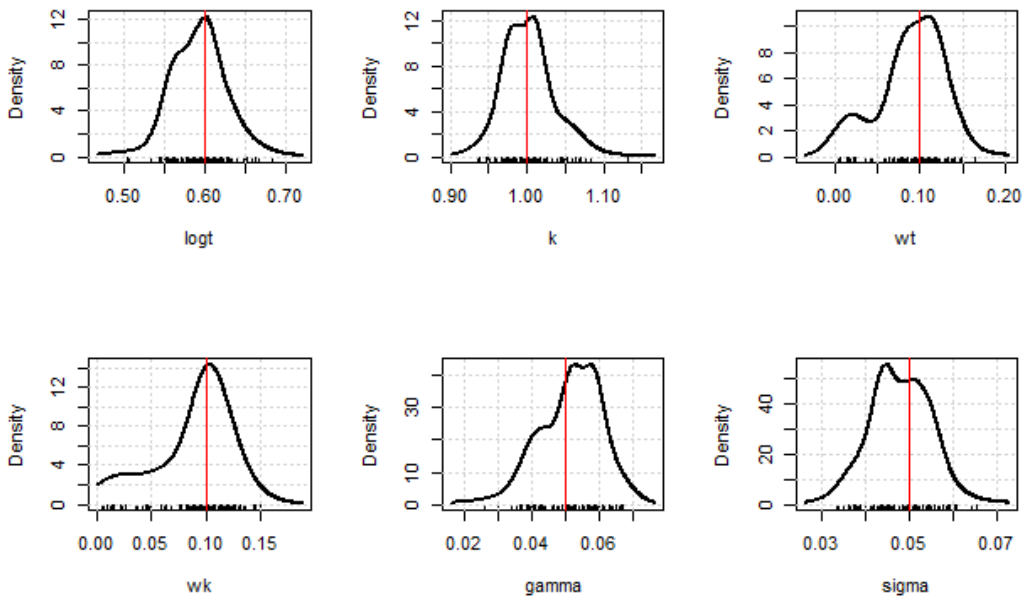

Figura 5.2: Distribuzione dei parametri sulle 100 simulazioni SAEM-PMCMC (C1). Le distribuzioni per la configurazione C2 sono analoghe.

Si osservi che i passi SA ed M sono stati implementati in maniera identica nei due casi. La differenza sostanziale tra i due metodi risiede nel passo S di simulazione, affrontato con due tecniche diverse, analizzate nella Sezione 5.1 e nella sezione corrente.

La seguente Tabella riunisce i risultati ottenuti coi due metodi, e già riportati nelle Tabelle 5.1 e 5.2, per semplificarne il confronto.

		$\log(\tau)$	k	w_τ	w_k	γ	σ
Inizializzazione C1		log(10)	1.5	0.5	0.5	0.25	0.25
Valori veri		0.6	1.0	0.1	0.1	0.05	0.05
SAEM - MCMC C1	mean	0.578	1.018	0.091	0.098	0.051	0.050
	sd	0.032	0.029	0.025	0.024	0.005	0.003
SAEM - PMMH C1	mean	0.600	0.997	0.092	0.092	0.050	0.050
	sd	0.032	0.027	0.026	0.022	0.005	0.002
Valori veri C2		2.3	1.0	0.5	0.5	0.5	0.5
SAEM - MCMC C2	mean	2.29	1.0	0.465	0.479	0.505	0.493
	sd	0.143	0.123	0.113	0.088	0.033	0.054
SAEM - PMMH C2	mean	2.306	0.983	0.471	0.483	0.500	0.500
	sd	0.143	0.116	0.109	0.085	0.032	0.047

Tabella 5.3: Stime dei parametri per il modello OU ottenute con l'algoritmo SAEM-MCMC e con l'algoritmo PMCMC

Si osservi che entrambi i metodi danno ottimi risultati e che le deviazioni standard degli stimatori sono dello stesso ordine di grandezza. Il valore reale del parametro è sempre compreso nel rispettivo intervallo di previsione.

I risultati ottenuti sono in accordo con l'articolo Donnet and Samson [2014].

E' stata riscontrata, tuttavia, una notevole differenza tra i tempi computazionali dei due metodi. Il metodo SAEM-PMCMC risulta più lento. Questo è giustificato dal fatto che ogni iterazione del metodo PMCMC utilizza un algoritmo SMC per approssimare la densità con cui proporre.

Capitolo 6

Data Augmentation

Le assunzioni (1)-(2) analizzate all'inizio della Sezione (II) sono il punto cardine su cui poggiano gli algoritmi analizzati nel Capitolo 5(SAEM-MCMC e SAEM-PMCMC). In particolare l'assunzione (2) richiede che la densità di transizione del processo sia nota. Per la maggior parte dei processi noti questo non accade ed occorre quindi ricorrere a tecniche di approssimazione. L'idea generale è quella di approssimare la densità del processo mediante l'utilizzo di opportune tecniche (vedi Eulero Maruyama 1.8).
Per poter applicare metodi di questo tipo è necessario che i dati raccolti abbiano una distanza temporale minima.
In generale la distanza temporale è tutt'altro che piccola: raccogliendo i dati mediante esami clinici su pazienti si hanno solitamente dataset caratterizzati da sparsità delle misurazioni. Le tecniche di data augmentation servono ad ovviare a questo problema: i dati raccolti vengono infittiti mediante dati generati artificialmente, per ottenere un dataset caratterizzato da numerose misurazioni temporalmente molto vicine.
Il seguente capitolo ha come fine analizzare l'impatto delle tecniche di data augmentation sulla bontà delle stime.

6.1 Data Augmentation per il modello OU

Si consideri il modello Ornstein-Uhlenbeck definito dalle equazioni (4.24), (4.25), (4.26), con $\Delta >> 0$. Si supponga la densità $p(X_{i,j\Delta}, \Delta, j | X_{i,(j-1)\Delta}, \phi_i, \theta)$ non nota.

Sotto queste assunzioni è stata sviluppata una tecnica di data augmentation, strettamente adattata al processo analizzato, che permetta di studiare l'impatto che i punti artificialmente generati hanno sulla stima di θ.

In questa sezione viene omesso l'indice i relativo al paziente per semplificare la notazione, tuttavia ciò che segue è da intendersi per ogni paziente i.

Per ogni coppia di osservazioni $(Y_{j\Delta}, Y_{(j+1)\Delta})$ si considerino M punti intermedi $\{X_{j\Delta + r\frac{\Delta}{M}}\}_{r=1,...,M}$.
Sfruttando il metodo di Eulero-Maruyama (1.8), per $r \in \{1 \cdots M\}$ vale

$$X_{j\Delta + r\frac{\Delta}{M}} = X_{j\Delta}\left(1 - \frac{\Delta}{\tau}\right)^r + \sum_{i=0}^{r-1} k\Delta\left(1 - \frac{\Delta}{\tau}\right)^i + \sum_{i=0}^{r-1} \gamma\sqrt{\Delta}\left(1 - \frac{\Delta}{\tau}\right)^i N_i^s \tag{6.1}$$

dove N_i^s sono i.i.d. normali standard e $X_{(j+1)\Delta}$ è il valore del processo X al tempo $\Delta(j+1)$.

Inoltre per ogni j vale:

$$\mathcal{L}_\theta(X_{j\Delta + \frac{\Delta}{M}}, ..., X_{(j+1)\Delta}, Y_{(j+1)\Delta} | X_{j\Delta}) = g_\theta(Y_{(j+1)\Delta} | X_{(j+1)\Delta}) \prod_{r=1}^{M} f_\theta(X_{j\Delta + r\frac{\Delta}{M}} | X_{j\Delta + (r-1)\frac{\Delta}{M}}) \tag{6.2}$$

dove g è la funzione di densità di $Y_{(j+1)\Delta} | X_{(j+1)\Delta}; \theta \sim \mathcal{N}(0, \sigma^2)$ e f è la funzione di densità di $X_{j\Delta + r\frac{\Delta}{M}} | X_{j\Delta + (r-1)\frac{\Delta}{M}}$ che deriva direttamente dell'equazione (1.14), sfruttando il metodo di Eulero-Maruyama.

Sfruttando le proprietà intrinseche del processo di Ornstein-Uhlenbeck, dalle equazioni (6.1) e (6.2) si ottiene la densità di $(X_{j\Delta + \frac{\Delta}{M}}, ..., X_{(j+1)\Delta} | Y_{(j+1)\Delta}, X_{j\Delta}; \theta)$ che ha distribuzione multivariata di parametri:

$$\mathbb{E}[X_{\overline{r}}] = X_{j\Delta}\left(1-\frac{\Delta}{\tau}\right)^{r} + \sum_{i=0}^{\overline{r}-1} k\Delta\left(1-\frac{\Delta}{\tau}\right)^{i}, \tag{6.3}$$

$$Cov(X_{\overline{r}}, X_{\overline{s}}) = \sum_{i=1}^{min(\overline{r},\overline{s})} \gamma^2\Delta\left(1-\frac{\Delta}{\tau}\right)^{(\overline{r}-i)}\left(1-\frac{\Delta}{\tau}\right)^{(\overline{s}-i)} \tag{6.4}$$

dove $\overline{r} = j\Delta + r\frac{\Delta}{M}$ e $\overline{s} = j\Delta + s\frac{\Delta}{M}$, con $r, s \in \{1, ..., M\}$.
Questa densità è stata usata per simulare $(X_{j\Delta+\frac{\Delta}{M}}, ..., X_{(j+1)\Delta})$ dati $Y_{(j+1)\Delta}$ e $X_{j\Delta}$.
Si osservi che i punti $(X_{j\Delta+\frac{\Delta}{M}}, ..., X_{(j+1)\Delta})$ vengono simulati tutti assieme, condizionatamente all'informazione contenuta nei punti $Y_{j\Delta}$ e $X_{j\Delta}$.
Se i punti fossero simulati in passi successivi, ovvero se si simulassero con la legge di $X_{j\Delta+(r+1)\frac{\Delta}{M}}|X_{j\Delta+r\frac{\Delta}{M}}$ per $r = 1, ..., M-2$ e con $X_{(j+1)\Delta}|(X_{j\Delta+\frac{(M-1)\Delta}{M}}, Y_{j\Delta})$ quando $r = M$, si otterrebbero traiettorie potenzialmente valide ma divergenti che non sfruttano l'informazione contenuta in $Y_{j\Delta}$. Si osservi ad esempio il caso riportato in Figura (6.1).

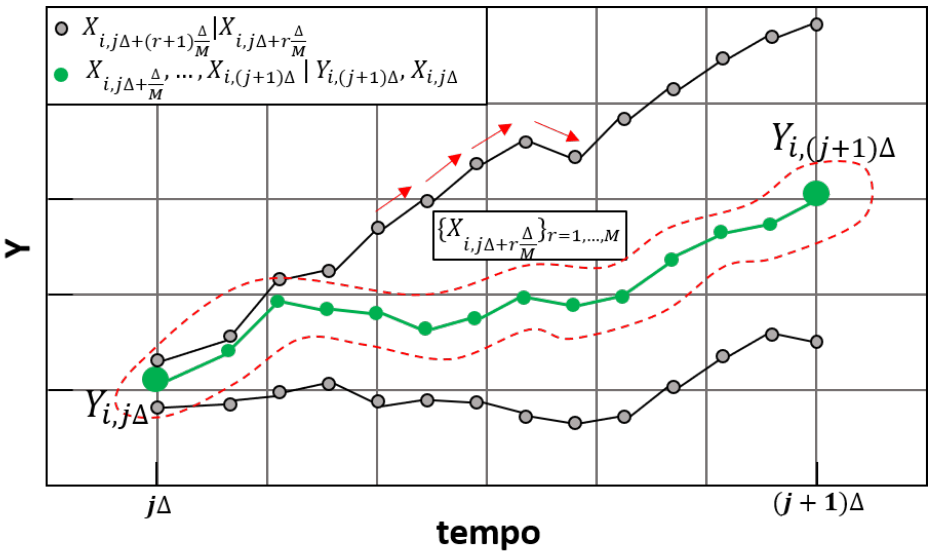

Figura 6.1: Infittimento.

6.2 SAEM-MCMC e Data Augemantation

La tecnica di data augmentation analizzata nella Sezione 6.1 è stata integrata all'interno dell'algoritmo SAEM-MCMC come segue.

---

procedure

for m=0 **do**

Inizializzazione di $\hat{\theta}^{(0)}$ e di $\hat{S}^0 = \mathbb{E}[S(\underline{Y}, \underline{X}, \Phi|\underline{Y}, \hat{\theta_0})]$,

end for

for m = 1:Numero Iterazioni **do**

simulare la variabili non osservate $\underline{\tilde{X}}^{(m)} = (\tilde{X}_1^{(m)}, ..., \tilde{X}_{NP}^{(m)})$ e $\Phi^{(m)} = (\phi_1^{(m)}, ..., \phi_{NP}^{(m)})$

per ogni paziente mediante *metodo MCMC*

aggiornare $\hat{S}^{(m)}$ come

$$\hat{S}^{(m)} = \hat{S}^{(m-1)} + \alpha^{(m)}\left(S(\underline{Y}, \underline{\tilde{X}}^{(m)}\Phi^{(m)}) - \hat{S}^{(m-1)}\right)$$

aggiornare $\hat{\theta}^{(m-1)}$ con $\hat{\theta}^{(m)} = \max_\theta(-\tilde{a}(\theta) + < \theta, \hat{S}^{(m)} >)$

end for

end procedure

---

dove $\tilde{X}_i^{(m)} = \{X_{i,j\Delta+r\frac{\Delta}{M}}\}_{r=1,...,M;j=1,...,T}$ è la traiettoria, completa di dati infittiti, per il paziente i ottenuta combinando MCMC e data augmentation.

Le statistiche sufficienti utilizzate e le loro approssimazioni dipendono da $\underline{\tilde{X}}^{(m)}$ e $\Phi^{(m)}$ e sono modificate come segue.

(6.5)

$$\hat{S}_1^{(m)}(\underline{Y},\underline{X},\Phi) = \sum_{i=1}^{NP} \phi_i^{(m)},$$

$$\hat{S}_2^{(m)}(\underline{Y},\underline{X},\Phi) = \sum_{i=1}^{NP} (\phi_i^{(m)})^{\mathrm{T}} \phi_i^{(m)},$$

$$\hat{S}_3^{(m)}(\underline{Y},\underline{X},\Phi) = \sum_{i=1}^{NP} (Y_i - X_i^{(m)})(Y_i - X_i^{(m)})^T,$$

$$\hat{S}_4^{(m)}(\underline{Y},\underline{X},\Phi) = \sum_{i=1}^{NP} \sum_{j=1}^{\mathrm{T}} \sum_{r=1}^{(m-1)} \frac{\left(X^{(m)}_{i,j\Delta+(r+1)\frac{\Delta}{M}} - X^{(m)}_{i,j\Delta+r\frac{\Delta}{M}} \left(1 - \frac{\Delta}{M\cdot\tau_i^{(m)}}\right) - k_i^{(m)} \frac{\Delta}{M} \right)^2}{\frac{\Delta}{M}}$$

da cui si ottengono le stime per $\mu = (log(\tau), k),\ \Omega = \mathrm{diag}(w_\tau, w_k),\ \sigma,\ \gamma$, come

(6.6)

$$\begin{aligned}
\hat{\mu}^{(m)} &= \frac{1}{NP}\hat{S}_1^{(m)}, \\
\hat{\Omega}^{(m)} &= \frac{1}{NP}\hat{S}_2^{(m)} - \frac{1}{NP^2}(\hat{S_1^{(m)}})^{\mathrm{T}} \hat{S}_1^{(m)} \\
\hat{\sigma}^{(m)} &= \sqrt{\frac{\hat{S}_3^{(m)}}{NP\cdot T}}, \\
\hat{\gamma}^{(m)} &= \sqrt{\frac{\hat{S}_4^{(m)}}{NP\cdot T\cdot M}}.
\end{aligned}$$

I punti artificialmente generati entrano solamente nella stima di γ. Per la stima di σ vengono invece solo utilizzate le variabili relative al processo X che corrispondono temporalmente ai dati raccolti $\underline{Y}$.

Il metodo che unisce le tecniche di data-augmentation all'interno del metodo SAEM-MCMC è indicato nel seguito con la sigla SAEM-MCMC+DA1.

6.2.1 Come scegliere il valore M

In quanto esposto finora, relativamente alla tecnica di data augmentation presentata, si è fatto riferimento al valore M che caratterizza il numero di punti $\{X_{i,j\Delta+r\frac{\Delta}{M}}\}_{r=1,\dots,M}$ che sono generati tra le osservazioni $Y_{i,j\Delta}$ e $Y_{i,(j+1)\Delta}$ per ogni j.
La scelta di M non è univoca ed è strettamente legata all'intervallo temporale Δ che intercorre tra le osservazioni nel dataset $\underline{Y}$. La distanza temporale tra i punti utilizzati per infittire viene indicata con $\Delta_M = \frac{\Delta}{M}$.

La scelta di infittire il dataset è praticabile nel caso in cui

$$1 - e^{\frac{-\Delta}{\tau_i}} \text{ non sia troppo vicino a } \frac{\Delta}{\tau_i} \text{ per ogni } i. \tag{6.7}$$

Si consideri, infatti, lo sviluppo in serie di Taylor al primo ordine per la funzione esponenziale

$$e^{-z} = 1 - z + o(z), \quad z \to 0$$

Se $\frac{\Delta}{\tau_i}$ è sufficientemente piccolo vale $e^{\frac{-\Delta}{\tau_i}} \simeq 1 - \frac{\Delta}{\tau_i}$ per ogni i. In tal caso non è necessario infittire. I punti, infatti, sono sufficientemente vicini da garantire che valga lo sviluppo di Taylor al primo ordine, su cui si basa il metodo di Eulero-Maruyama.

Nella condizione (6.7) appena descritta è quindi appropriato applicare il metodo di Eulero-Maruyama, dopo aver simulato i nuovi punti.
La scelta di Δ_M ed M, per le simulazioni, deve essere tale da garantire una distanza temporale sufficientemente piccola in modo da rendere valido il metodo di Eulero-Maruyama.

Sia τ il valore utilizzato per simulare il dataset e sia Δ la distanza temporale tra le osservazioni, tali che valga (6.7). Si sceglie allora M tale che $1 - e^{\frac{\Delta_M}{\tau}} \simeq \frac{\Delta_M}{\tau}$.

Per valutare l'impatto che il valore M ha sulla bontà delle stime sono state fatte alcune simulazioni su un dataset definito dalle equazioni (4.24),

(4.25) e (4.26).
Sono stati considerati 15 differenti dataset composti da 20 pazienti ognuno. Per ogni paziente sono state considerate 15 misurazioni con distanza temporale costante $\Delta = 2$. Il dataset è stato simulato con $\theta = (0.6, 1, 0.1, 0.1, 0.05, 0.05)$. Si osservi la validità dell'affermazione (6.7):

1. $1 - e^{\frac{-\Delta}{\tau}} = 0.8646647$ e $\frac{\Delta}{\tau} = 2$.

Sono stati utilizzati diversi valori di M, compresi tra 1 e 15, all'interno di un algoritmo di tipo SAEM-MCMC+DA1.

La seguente tabella riporta il valore del rapporto $1 - e^{\frac{\Delta_M}{\tau}}$ e di $\frac{\Delta_M}{\tau}$ per alcuni dei valori scelti per M.

M	1	3	5	7	10	15
$1 - e^{\frac{\Delta_M}{\tau}}$	0.86	0.49	0.33	0.25	0.18	0.12
$\frac{\Delta_M}{\tau}$	0.14	0.37	0.67	0.75	0.82	0.88

Tabella 6.1: Confronto tra $1 - e^{\frac{\Delta_M}{\tau}}$ e $\frac{\Delta_M}{\tau}$ al variare del valore M.

Si osservi che per ogni coppia di dati $(Y_{i,j\Delta}, Y_{i,(j+1)\Delta})$ si generano $M - 1$ nuovi punti $\{X_{i,j\Delta+r\frac{\Delta}{M}}\}_{r=1,\dots,M}$ usati per infittire. Questi punti non corrispondono ad alcuna osservazione reale. L'M-esimo punto corrisponde invece al valore osservato $Y_{i,(j+1)\Delta}$. Porre $M = 1$ corrisponde quindi a non infittire, ed utilizzare il dataset così come si presenta con $\Delta_M = \Delta$, applicando quindi il metodo di Eulero-Maruyama direttamente ai punti del dataset.

Quando vale (6.7) è ragionevole supporre che porre $M = 1$ non porti a stimare con accuratezza e precisione θ. Applicare il metodo di Eulero-Maruyama in condizioni in cui non vale lo sviluppo di Taylor al primo ordine equivale ad utilizzare un'approssimazione della densità di transizione del processo non valida.
Aumentare il numero M sembra quindi poter migliorare la stima di θ. Maggiore è il numero di punti che vengono simulati più accurata è la stima della densità di transizione, tuttavia, aumenta il rumore introdotto nel dataset.

E' quindi necessario trovare un tradeoff che garantisca una buona approssimazione della densità di transizione, introducendo meno rumore possibile all'interno del dataset.

In figura 6.2 è riportato l'errore percentuale ottenuto per ogni parametro al variare di M.

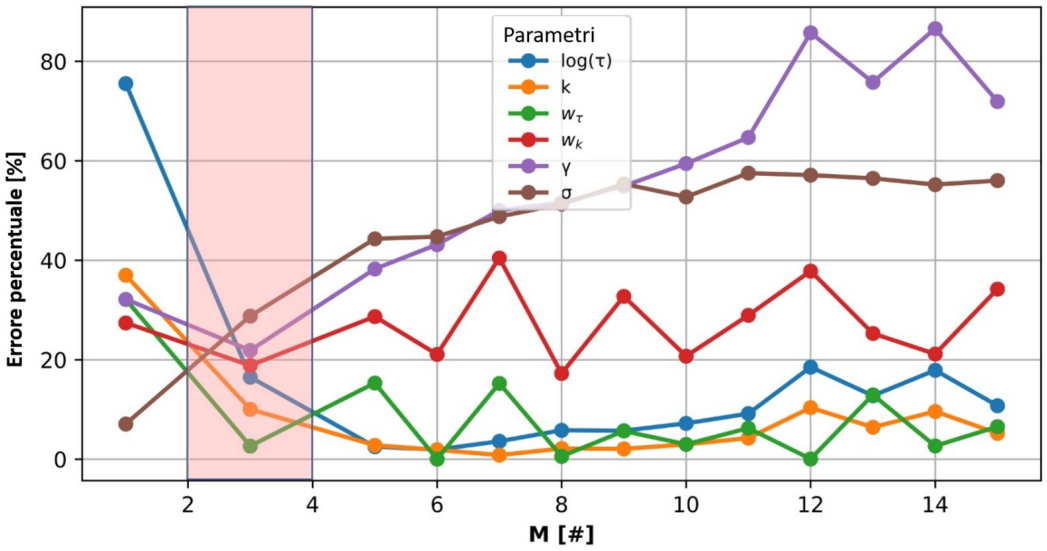

Figura 6.2: Errore percentuale commesso sulla stima dei parametri, per il modello OU, al variare del numero M di punti utilizzati per infittire.

Si osservi che per $M = 1$ l'errore percentuale commesso è molto alto per tutti i parametri. La condizione (6.7) giustifica il fatto che il metodo di Eulero-Maruyama non sia valido.
Per valori di $M > 1$ l'errore commesso su $\log(\tau)$, k, w_τ, w_k è sempre sotto al 40%. Non sembra quindi che questi parametri risentano, in modo significativo, del valore scelto per M. E' interessante osservare che sia il valore di γ, sia quello di σ sono invece strettamente legati alla quantità M. Questo non sorprende per quanto riguarda γ: nella stima di γ vengono utilizzati tutti i punti simulati. Il fatto che anche la stima di σ risenta del valore di M è giustificabile osservando che γ e σ sono due misure di variabilità inter-paziente

del sistema (ovvero due varianze che non dipendono dal paziente considerato); se una delle due viene stimata erroneamente il sistema riflette l'errore anche sull'altra. Più rumore viene introdotto nel dataset, meno l'algoritmo riesce a distinguere tra le due varianze.

Il valore ottimale risulta essere $M = 3$. Pertanto questo valore è stato utilizzato per tutte le simulazioni che seguono.

6.2.2 Stimare γ con un metodo MCMC

Nella precedente sezione è stata analizzata l'influenza che il valore M ha sulle stime dei parametri. Il parametro γ è risultato essere l'elemento più delicato da stimare. Per provare a migliorare la sua stima si è pensato di considerare anche esso come una variabile non osservata da generare quindi al passo m-esimo dell'algoritmo SAEM, con un iterazione di tipo MCMC, dopo aver simulato $\underline{X}$ ed i parametri individuali Φ.
Sono stati utilizzati due approcci differenti:

1. **Random Walk**
   All'iterazione $m+1$, successivamente alla fase di simulazione delle variabili non osservate (fase MCMC), si genera un nuovo valore $\log(\gamma^*)$ con un random walk di deviazione standard pari a 0.5. Questo valore è accettato condizionatamente al valore corrente $\log(\gamma^{(m)})$, e alle variabili $\underline{X}^{(m+1)}$ e $\Phi^{(m+1)}$ appena simulate. La probabilità di accettazione vale

   $$a(\gamma^{(m)}, \gamma^*) = \min\left(1, \frac{p(\gamma^*)}{p(\gamma^{(m)})} \frac{p(\underline{\tilde{X}}^{(m+1)}|\gamma^*, \sigma^{(m)}, \Phi^{(m+1)})}{p(\underline{\tilde{X}}^{(m+1)}|\gamma^{(m)}, \sigma^{(m)}, \Phi^{(m+1)})}\right)$$

   determinando così il nuovo valore $\gamma^{(m+1)}$, che costituisce l'approssimazione $\hat{\gamma}^{(m+1)}$.

   E' stata scelta una distribuzione Gamma Inversa su γ di parametri $a = b = 1$.

   Questo metodo viene indicato con la sigla SAEM-MCMC+DA1+RW.

2. **Gibbs sampler**

   Per sfruttare un approccio di tipo Gibbs Sampler è necessario conoscere la legge $p(\gamma|\underline{Y}, \tilde{\underline{X}}, \log(\tau), k, w_\tau, w_k, \sigma)$ da cui generare i nuovi valori di γ ad ogni iterazione m del metodo SAEM.
   Sia γ^2 una variabile di tipo Gamma Inversa di parametri $a = b = 5$. Sfruttando il teorema di Bayes si ha

$$p(\gamma|\underline{Y}, \tilde{\underline{X}}, \log(\tau), k, w_\tau, w_k, \sigma) = \frac{p(\underline{Y}, \tilde{\underline{X}}, \log(\tau), k, w_\tau, w_k, \sigma|\gamma)p(\gamma)}{p(\underline{Y}, \tilde{\underline{X}}, \log(\tau), k, w_\tau, w_k, \sigma)}$$

   da cui si ricava, sfruttando la decomposizione (4.22)

$$p(\gamma|\underline{Y}, \tilde{\underline{X}}, \log(\tau), k, w_\tau, w_k, \sigma) \propto p(\tilde{\underline{X}}| \log(\tau), k, w_\tau, w_k, \sigma, \gamma)p(\gamma).$$

   Per le proprietà di coniugazione della variabile Gamma Inversa, si ottiene che anche la variabile $\gamma^2|(\underline{Y}, \tilde{\underline{X}}, \log(\tau), k, w_\tau, w_k, \sigma)$ è una Gamma Inversa di parametri

   (6.8)

$$\overline{a} = a + \frac{NP \cdot T \cdot M}{2},$$

$$\overline{b} = b + \sum_{i=0}^{NP} \sum_{j=0}^{\mathrm{T}} \sum_{r=1}^{M} \frac{M}{2\Delta} \left(X_{i,j\Delta + r\frac{\Delta}{M}} - X_{i,j\Delta+(r-1)\frac{\Delta}{M}} \left(1 - \frac{\Delta}{\tau_i M}\right) - k_i \frac{\Delta}{M} \right)^2$$

   La distribuzione trovata si utilizza per generare al passo $m+1$ il valore di γ, condizionatamente a $\tilde{\underline{X}}^{(m+1)}$, $\Phi^{(m+1)}$ ed $\hat{\theta}^{(m)}$. Questo metodo viene indicato con la sigla SAEM-MCMC+DA1+GS.

Si osservi che entrambi i metodi riscontrano qualche difficoltà nella stima di γ e σ. Il metodo che sfrutta un aggiornamento di tipo Gibbs Sampler risente in particolar modo del rumore introdotto dai punti generati per infittire il dataset. Infatti, ogni valore di γ è generato con una densità condizionata ai dati (anche quelli artificiali). Accade quindi che all'iterazione m dell'algoritmo EM, il valore di $\hat{\gamma}^{(m)}$ generato è strettamente legato ai punti $\tilde{\underline{X}}^{(m)}$ ed al rumore che essi introducono, risultando quindi in una stima poco accurata di γ. Al passo $m+1$, condizionatamente al valore $\hat{\gamma}^{(m)}$, vengono generati i nuovi valori $\tilde{\underline{X}}^{(m+1)}$ che risentono della mancata accuratezza del valore $\hat{\gamma}^{(m)}$.

		$\log(\tau)$	k	w_τ	w_k	γ	σ
Inizializzazione		1.1	1.5	0.5	0.5	0.25	0.25
Valori veri		0.6	1.0	0.1	0.1	0.05	0.05
SAEM - MCMC	mean	0.707	0.902	0.090	0.058	0.070	0.024
DA1 + RW	sd	0.050	0.030	0.037	0.037	0.005	0.004
SAEM - MCMC	mean	0.689	0.915	0.101	0.046	0.109	0.021
DA1 + GS	sd	0.044	0.022	0.032	0.033	0.008	0.004

Tabella 6.2: Risultati ottenuti per il modello OU con un algoritmo SAEM-MCMC+DA1 che stima γ mediante metodo Bayesiano MCMC.

Il metodo che sfrutta l'aggiornamento di tipo Metropolis-Hastings genera i valori di γ con un random walk, risulta quindi che i valori generati risentono meno del rumore introdotto dai punti $\underline{\tilde{X}}^{(m)}$.

6.2.3 Migliorare la stima di γ

Come discusso, nella stima del parametro γ sono coinvolti sia i punti corrispondenti alle osservazioni del dataset, sia i punti artificialmente generati per infittire. Dalle analisi della sezione 6.2.1 è risultato che l'introduzione di questi punti sia responsabile delle difficoltà riscontrate nella stima di γ e σ. Per lo specifico processo OU sotto analisi è possibile escludere dalla stima di γ i punti artificialmente generati ed utilizzarli solamente per approssimare la densità di transizione del processo.

Le stime dei parametri $\log_\tau,\ k,\ w_\tau,\ w_k,\ \sigma$ non dipendono direttamente dai punti $\tilde{X}_i^{(m)} = \{X_{i,j\Delta+r\frac{\Delta}{M}}\}_{r=1,...,M;j=1,...,T}$ nella loro totalità.

Richiamando l'equazione (6.1)

$$X_{i,j\Delta+r\frac{\Delta}{M}} = X_{i,j\Delta}\left(1-\frac{\Delta}{\tau}\right)^r + \sum_{l=0}^{r-1} k\Delta\left(1-\frac{\Delta}{\tau}\right)^l + \sum_{l=0}^{r-1}\gamma\sqrt{\Delta}\left(1-\frac{\Delta}{\tau}\right)^l N_l^s$$

e sfruttando la distribuzione normale di $X_{i,j\Delta+r\frac{\Delta}{M}}|X_{j\Delta}$ per ogni $r = 1, ..., M$

si può utilizzare la seguente statistica sufficiente

$$\hat{S}_4^{(m)} = \sum_{i=1}^{NP} \sum_{j=1}^{T} \frac{\left[\left(X_{i,j\Delta} - X_{i,(j-1)\Delta} \left(1 - \frac{\Delta}{M \cdot \tau_i} \right) \right)^M - k_i \frac{\Delta}{M} s_0 \right]^2}{s_i \frac{\Delta}{M}} \tag{6.9}$$

$$\begin{aligned} s_0 &= \sum_{l=1}^{M} \left(1 - \frac{\Delta}{M \cdot \tau} \right)^{l-1} \\ s_i &= \sum_{l=1}^{M} \left[\left(1 - \frac{\Delta}{M \cdot \tau} \right)^{l-1} \right]^2 \end{aligned} \tag{6.10}$$

da cui si ricava

$$\hat{\gamma}^{(m)} = \sqrt{\frac{\hat{S}_4^{(m)}}{NP \cdot T}}.$$

Si osservi che il metodo di Eulero-Maruyama, utilizzato per ricavare la legge $X_{i,j\Delta} | X_{i,(j-1)\Delta}$, necessita dei punti intermedi $\{X_{i,j\Delta + r\frac{\Delta}{M}}\}_{r=1,\dots,M}$ per poter essere applicato. Tuttavia, utilizzando l'equazione (6.10) questi punti non entrano direttamente nella stima di γ.

Questo metodo è stato utilizzato all'interno del metodo SAEM-MCMC con data augmentation (DA) ed è indicato nel seguito con la sigla SAEM-MCMC+DA2.

Capitolo 7

Risultati finali

Il capitolo finale è interamente dedicato al confronto completo di tutti i metodi applicati e spiegati precedentemente e dei risultati ottenuti per il dataset simulato sotto il modello OU.

Sono stati simulati 15 dataset. Ogni dataset si compone di 20 pazienti. Per ogni paziente sono state raccolte 15 misurazioni con distanza temporale $\Delta = 2$.

Il valore del vettore θ utilizzato in fase di simulazione è

$$\theta = (0.6, 1, 0.1, 0.1, 0.05, 0.05).$$

Viene riportata di seguito la funzione utilizzata per generare i dataset. La funzione prende in input il vettore θ, il numero dei pazienti, il numero dei dati T, il numero M dei punti che serviranno all'algoritmo di data augmentation ed infine due vettori contenenti i parametri individuali per i pazienti. La funzione genera $M \cdot T$ valori del tipo $\{X_{i,j\Delta+r\frac{\Delta}{M}}\}_{j=1,...,T;r=1,...,M}$ per ogni paziente i, con $i = 1, ..., NP$, utilizzando la densità del processo OU.
Dai valori $\{X_{i,j\Delta+r\frac{\Delta}{M}}\}_{j=1,...,T;r=1,...,M}$ si ottengo i valori $\{Y_{i,j\Delta+r\frac{\Delta}{M}}\}_{j=1,...,T;r=1,...,M}$ come $Y_{i,j\Delta+r\frac{\Delta}{M}} = X_{i,j\Delta+r\frac{\Delta}{M}} + \epsilon_{ijr}$, con $\epsilon_{ijr} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma^2)$.

```
dataset<-function(teta,NP,tempoT,deltaG,N,logtp,kp)
```

```
  #teta:insieme dei valori reali dei parametri
  #NP:numero pazienti
  #tempo: numero osservazioni per paziente
  #deltaG:distanza tra le misurazioni in termini temporali
  #N: infittimento
  #logtp:vettore parametri logt individuali
  #kp:vettore parametri k individuali
{
  delta=deltaG/N #delta delle misurazioni ravvicinate
  %tempoT=tempo;
  teta[1]->logt
  teta[2]->k
  teta[3]->wt
  teta[4]->wk
  teta[5]->gm
  teta[6]->sigmaVar
  NP=pazienti
  tp=exp(logtp)


  #su ogni colonna ci sono le osservazioni per un paziente.
  # X=(Xp1|Xp2|...|XpN) matrice dei dati
  Xdata=array(dim=c((N*tempoT),NP))

  eps=array(dim=c(N*tempoT,NP))
  eta=array(dim=c(N*tempoT,NP))


 #su ogni colonna ci sono le osservazioni per un paziente.
 #y=(yp1|yp2|...|ypN) matrice dei dati
  y=array(dim=c((N*tempoT),NP))


  for(paziente in 1:NP)
  {
  mean1=tp[paziente]*kp[paziente]+(-tp[paziente]* kp[paziente])*(
    exp(-delta/tp[paziente]))
  var1=(gm*gm*tp[paziente]*(1-exp(-2*delta/tp[paziente])))/2
```

```
40    Xdata[1,paziente]=rnorm(1,mean1,sqrt(var1))
      eps[1,paziente]=rnorm(1,0,sigmaVar)
42
      y[1,paziente]=(Xdata[1,paziente]) + eps[1,paziente]
44
      for(i in 2:(N*tempoT))
46      {
        eta[i,paziente]=rnorm(1,0,sqrt(var1))
48      eps[i,paziente]=rnorm(1,0,sigmaVar)

50      Xdata[i,paziente]=tp[paziente]*kp[paziente]+(Xdata[i-1,
        paziente]-tp[paziente]*kp[paziente])* exp(-delta/ tp[paziente
        ])+eta[i,paziente]

52      y[i,paziente]=((Xdata[i,paziente])) + eps[i,paziente]
        }
54    }

56  return(list(y=y[N*(1:tempoT),],ycomplete=y,XC=Xdata,XD=Xdata[N*
        (1:tempoT),]))
    }
```

Come output vengono forniti, per ogni paziente i:

1. $\{Y_{i,j\Delta}\}_{j=1,..,T}$,
2. $\{Y_{i,j\Delta+r\frac{\Delta}{M}}\}_{j=1,...,T;r=1,...,M}$,
3. $\{X_{i,j\Delta+r\frac{\Delta}{M}}\}_{j=1,...,T;r=1,...,M}$,
4. $\{X_{i,j\Delta}\}_{j=1,..,T}$.

I dati che costituiscono il dataset sono riportati al punto (1). Gli altri dati sono stati utilizzati, in fase di studio, per verificare la somiglianza tra i punti generati, nelle varie iterazioni del metodo EM, con la tecnica di data augmentation e i punti reali. (Figura 7.1).

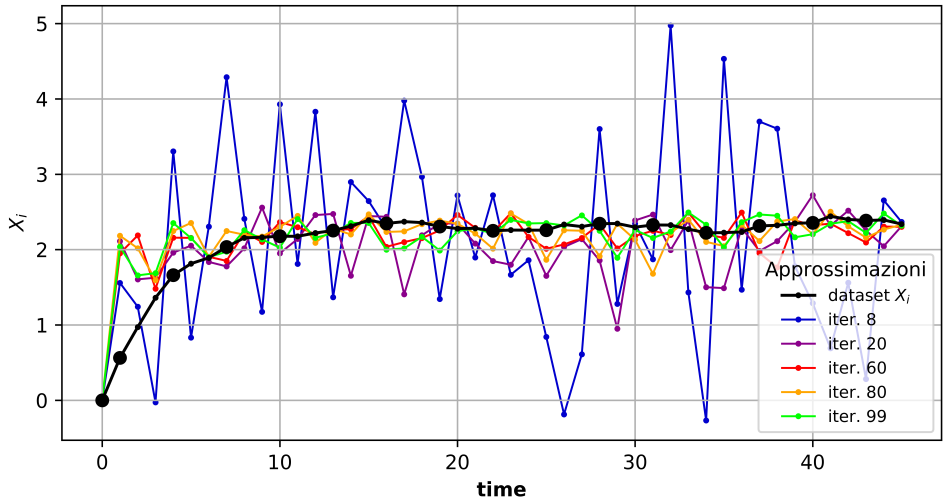

Figura 7.1: Approssimazioni successive di X_i per un generico paziente i.

Dopo avere simulato i dataset di interesse, sono stati applicati i seguenti metodi:

- SAEM-MCMC,
- SAEM-PCMC,
- SAEM-MCMC+DA1 con M=3,
- SAEM-MCMC+DA1+RW con M=3,
- SAEM-MCMC+DA1+GS con M=3,
- SAEM-MCMC+DA2 con M=3.

Ogni metodo si compone di 100 iterazioni SAEM, ognuna delle quali contiene 100 iterazioni MCMC o PMCMC. La seguente Tabella raccoglie le stime ottenute con i 6 metodi.

		$\log(\tau)$	k	w_τ	w_k	γ	σ
Inizializzazione		1.1	1.5	0.5	0.5	0.25	0.25
Valori veri		0.6	1.0	0.1	0.1	0.05	0.05
SAEM - MCMC	mean	0.584	1.013	0.097	0.098	0.053	0.047
	sd	0.039	0.040	0.030	0.031	0.009	0.009
SAEM - PMMH	mean	0.594	1.003	0.891	0.088	0.051	0.048
	sd	0.035	0.035	0.040	0.038	0.009	0.007
SAEM - MCMC	mean	0.699	0.900	0.103	0.081	0.061	0.036
DA1	sd	0.043	0.036	0.027	0.027	0.006	0.009
SAEM - MCMC	mean	0.707	0.902	0.090	0.058	0.070	0.024
DA1 + RW	sd	0.050	0.030	0.037	0.037	0.005	0.004
SAEM - MCMC	mean	0.574	1.021	0.092	0.042	0.202	0.031
DA1 + GS	sd	0.061	0.049	0.052	0.045	0.013	0.005
SAEM - MCMC	mean	0.639	0.960	0.162	0.144	0.047	0.063
DA2	sd	0.064	0.060	0.056	0.046	0.007	0.013

Tabella 7.1: Modello OU- risultati a confronto

Si osservano delle piccole differenze tra le performances dei metodi SAEM-MCMC/SAEM-PMCMC e quelle dei metodi che fanno uso di data augmentation.
I metodi che usano la densità esatta del processo risultano essere meno affetti da rumore, questo si traduce in risultati più precisi. Più si introduce rumore nel dataset mediante l'aggiunta di nuovi punti, più le stime peggiorano. Si osservi a sostegno della tesi che il metodo di tipo DA2 performa meglio dei corrispettivi metodi di tipo DA1, confermando che le stime peggiorano quando dipendono direttamente dai punti usati per infittire.
Il modello OU è uno dei pochi modelli per cui è possibile escludere i punti usati per infittire dalle stime dei parametri; l'unico scopo dei metodi DA2 è quello di aiutare a valutare le performances dei metodi più generici DA1, identificando con chiarezza la causa della loro instabilità. E' evidente che l'u-

tilizzo di tecniche di data augmentation per il modello scelto sia del tutto superflua, si è tuttavia scelto di lavorare su un modello semplice per estendere, in futuro, a modelli più complessi.

Conclusione

In questa tesi sono stati raccolti alcuni algoritmi, già studiati ed analizzati in Letteratura, per due diverse tipologie di processi stocastici: i processi a densità di transizione nota e i processi a densità di transizione non nota.

Per la prima categoria di processi, i metodi proposti in Letteratura sono soddisfacenti. Nella tesi è stato riproposta una dettagliata analisi delle performances ottenibili, utilizzando un processo Ornstein-Uhlenbeck.

Per la seconda categoria di processi non esiste tuttora un algoritmo globale di facile implementazione che dia buone performances. In Erhardt [2019] sono state evidenziate alcune difficoltà sorte implementando l'algoritmo proposto da Donnet and Samson [2008] che combina il metodo SAEM-MCMC con una tecnica di data augmentation basata su ponti Browniani. In particolare risulta difficile distinguere il valore di γ dal valore di σ.
Nella tesi, partendo dal lavoro di Erhardt [2019], si è indagata la motivazione alla base di questa difficoltà.
E' stato utilizzato un processo Ornstein-Uhlenbeck. Questo processo ricade nella prima classe di processi studiati e non necessita quindi dell'utilizzo di tecniche di data augmentation. Si è scelto tuttavia di utilizzarlo per poter confrontare gli algoritmi tra le due classi. E' stato proposto un differente metodo di data augmentation, creato appositamente per l'algoritmo OU.
Grazie a numerose simulazioni si è osservato che il rumore introdotto dai punti utilizzati per infittire il dataset di partenza copre l'informazione vera contenuta nei dati, rendendo impossibile distinguere in maniera precisa il valore dei parametri, in particolar modo γ e σ.

L'analisi proposta in questa tesi, valutando debolezze e punti di forza di numerosi algoritmi, fornisce basi utili per generalizzare gli algoritmi già esistenti a modelli più complessi. Sarà quindi obiettivo di futuri lavori estendere la tecnica di data augmentation studiata ad ogni tipo di processo, cercando di ottenere un algoritmo di tipo SAEM-MCMC applicabile per la maggior parte dei processi noti.

Bibliografia

Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342, 2010.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

Sophie Donnet and Adeline Samson. Parametric inference for mixed models defined by stochastic differential equations. *ESAIM: Probability and Statistics*, 12:196–218, 2008.

Sophie Donnet and Adeline Samson. Using PMCMC in EM algorithm for stochastic mixed models: theoretical and practical issues. 2014.

Elvira M. Erhardt. *Bayesian and Stochastic Techniquesfor Mixed Effects Modelsin Early Phase Drug Developmen*. PhD thesis, Politecnico di Torino, 2019.

Stefano M Iacus. *Simulation and inference for stochastic differential equations: with R examples*. Springer Science & Business Media, 2009.

Estelle Kuhn and Marc Lavielle. Coupling a stochastic approximation version of EM with an MCMC procedure. *ESAIM: Probability and Statistics*, 8: 115–131, 2004.

Estelle Kuhn and Marc Lavielle. Maximum likelihood estimation in nonlinear mixed effects models. *Computational Statistics & Data Analysis*, 49(4): 1020–1038, 2005.

Thomas Mikosch. *Elementary stochastic calculus with finance in view*. World scientific, 1998.

Christian P Robert, George Casella, and George Casella. *Introducing monte carlo methods with r*, volume 18. Springer, 2010.

Darren J Wilkinson. *Stochastic modelling for systems biology*. CRC press, 2011.