POLITECNICO DI MILANO

Scuola di Ingegneria dei Sistemi

Corso di Studi in Ingegneria Matematica

Tesi di Laurea Magistrale

METODI NUMERICI E STATISTICI PER LA SIMULAZIONE E VALIDAZIONE DI ECG

RELATORE: Prof. Anna Maria Paganoni
CORRELATORE: Dott. Ing. Francesca Ieva

Tesi di laurea di
Nicholas Tarabelloni

Matr. n. 782187

ANNO ACCADEMICO 2012-2013

A papà.

Sommario

Il presente lavoro di tesi combina metodi statistici e di analisi numerica per simulare segnali elettrocardiografici (ECG) e confrontarli con quelli di una popolazione di riferimento composta da tracciati raccolti in un contesto clinico reale. La simulazione di ECG viene svolta considerando il sistema di equazioni a derivate parziali del modello bidominio per il potenziale elettrico cardiaco e toracico, con modello ionico di chiusura di Mitchell-Schaeffer. Un'opportuna tecnica di analisi statistica non-parametrica per l'analisi degli ECG, basata sul concetto di profondità per dati funzionali multivariati, viene definita e implementata in una libreria di statistica computazionale. Essa viene applicata al confronto di due popolazioni di soggetti sani e patologici. Infine viene applicata alla validazione, rispetto alla popolazione reale di riferimento, di ECG costruiti a partire dalla soluzione del problema elementi finiti dell'elettrofisiologia cardiaca.

Abstract

The present work combines statistical and numerical methods to simulate realistic electrocardiograms (ECG) and compare them with signals belonging to a reference population, collecting data coming from a real, clinical context. The simulation of ECGs is carried out solving the PDEs arising from the bidomain model for the toracic and cardiac electric potential, completed with the Mitchell-Schaeffer ionic model. A proper statistical, non-parametric technique, based on the concept of depth for multivariate functional data, is devised to analyse the ECGs and implemented inside an ad-hoc library of high-performance computational statistics. It is tested on the comparison of two populations of normal and pathological signals. Finally, it is applied to the validation of normal ECGs computed from the solution of the discrete, finite-element bidomain problem with respect to the real, reference population.

Ringraziamenti

L'altra volta mi ero ripromesso di cominciarli per tempo. Invece, come al solito, mi sono ridotto all'ultimo momento e dovrò frustare dita e cervello.

Oggi si chiude il mio percorso di Laurea Magistrale. Due anni fa lo guardavo da fuori, ma in retrospettiva è stato un guizzo. No, scherziamo?, è stato infinito e così intenso e così denso che pensavo avrebbe scucito lo spazio-tempo (e in quel caso, tornare indietro e correggere le risposte di ARF non sarebbe stato poi così male).
Non sentirlo più nella coscienza è tutto e solo merito delle persone magnifiche che mi hanno accompagnato, che devo ringraziare veramente dal profondo del cuore (e dato che ci ho fatto su la tesi, mi credano).

Per primi ringrazio Anna, Francesca e Andrea, trio delle meraviglie dei modelli & metodi che, a parte un sacco di metodi, insegnano a tutti quelli che entrano in contatto con loro molti più modelli di quanto forse sappiano. Mi hanno mostrato il lato umano del fare matematica, dietro alle lavagne, sotto ai tomi, oltre i cividì, e il lato buono del lato umano. Per questo li ringrazio profondamente.

Di seguito vorrei ringraziare il prof. Gerbeau ed Elisa per l'opportunità che mi hanno dato di collaborare allo studio di un tema così interessante come l'elettrofisiologia cardiaca.

Ringrazio Bianca, autentico salvavita con braccia, gambe e facoltà di parola. Stefano, compagno di viaggio, di merende e di correzioni. Camilla, che si è guadagnata un cesto natalizio per sempre grazie all'appropriata capacità di giudizio. Teo, per "la gente coi probblemi ", per i "dammi TYR che ti passo questi qua ", per la birra, i gusti, il supporto morale. Giada, per i panini che non mi ha mai chiesto. Marco, per la fantastica simpatia, che poteva pure non portare in Bovisa. Teresa e Alice per aver soffiato sul fuoco della statistica. Giacomo, per la solida, reciproca stima. Anna e le sue mani, che hanno confezionato per me sontuosi lipidi. Stefano, per i tanti consigli. Rachele, per non avermi ucciso piantandomi nel cranio una chiavetta USB con "l'ultima versione del codice ". Tia, Marco, Ale e Rosanna per aver sopportato la mia latitanza e alleggerito le serate più dure.

Non sarei vivo se non fosse per Giorgia. Purtroppo si è trovata nel posto sbagliato al momento sbagliato. Un ingegnere matematico al suo ultimo semestre è un pericolo che insegnano ad evitare fin dall'asilo. Ma lei niente. E meno male, non ce l'avrei davvero fatta senza. La ringrazio

per la curiosità, la leggerezza, l'intelligenza e il suo dolce, semplice e costante supporto.

Ringrazio di cuore la mia famiglia, che in questi due anni per me difficili ha avuto il difficile compito di sopportarmi. Ringrazio mia madre, per credere alle mie speranze e per essere pronta ad assecondarle quando si concretizzano all'improvviso. Ringrazio le mie due sorelle, Arianna e Ambra, per la pazienza e la gentilezza di fronte ad un fratello così noioso. Ringrazio mio papà, che mi ha sostenuto in tutte le mie scelte e non è più qui con me per condividerne l'esito.

Ringrazio infine quella folla di persone che le mie pigre sinapsi non si sono ancora ricordate ma che con un sorriso, un gesto o ancor più graditamente un caffè mi hanno rallegrato la giornata. Tra questi c'è sicuramente qualcuno di importante con cui farò presto o tardi una grande gaffe. Mi scusino tutti. Vi ringrazio tutti.

Indice

Elenco delle figure

Capitolo 1

Introduzione e motivazione

Nel presente lavoro di tesi vengono studiati metodi statistici e di analisi numerica utilizzati per la simulazione di elettrocardiogrammi umani (ECG) e la loro validazione rispetto a dati realii. L'elettrocardiogramma è un esame non invasivo dell'attività elettrica del cuore consistente nella misurazione in dodici specifici punti del corpo umano delle differenze di potenziale elettrico che si propagano nei tessuti a seguito del battito cardiaco. L'ECG a dodici derivazioni è uno degli esami clinici più diffusi e viene utilizzato con ottimi risultati dal personale medico per monitorare lo stato di salute dei pazienti o l'eventuale insorgenza di patologie cardiovascolari.
Negli ultimi anni l'impiego della modellistica matematico-numerica e dell'analisi statistica nello studio di fenomeni cardiaci ha riscosso un crescente interesse in centri di ricerca e società cliniche. Lo testimoniano vari progetti, di cui un esempio è PROMETEO (PROgetto sull'area Milanese Elettrocardiogrammi Teletrasferiti dall'ExtraOspdaliero), una collaborazione tra il 118 Milano e il MOX, laboratorio di modellistica e calcolo scientifico del Dipartimento di Matematica del Politecnico di Milano, con lo scopo di applicare tecniche statistiche all'analisi di patologie cardiache rilevabili dall'esame dell'elettrocardiogramma; un altro esempio è costituito da EuHEART, un progetto di ricerca Europeo incentrato sullo sviluppo di metodi per la modellazione personalizzata del cuore e delle principali malattie cardiovascolari a partire da misurazioni cliniche, frutto di una collaborazione tra 16 partners accademici, clinici ed industriali, tra cui INRIA e Philips Research/Philips Healthcare; un terzo esempio è il progetto MATHCARD, finanziato dall'Unione Europea e coordinato dai centri CMCS dell'EPFL e MOX con lo scopo di sviluppare, analizzare ed implementare modelli matematici per il sistema cardiovascolare, con enfasi su fenomeni di trasporto, elettrofisiologia e meccanica cardiaca.
Generalmente il problema dello studio dell'attività del cuore viene affrontato separatamente con strumenti statistici o numerici. Questo lavoro di tesi vuole porsi come ponte tra i due, proponendosi come possibile primo passo verso la combinazione dei due aprocci, al fine di beneficiare dei punti di forza reciproci. In particolare, l'obiettivo di questa tesi è quello di utilizzare opportuni metodi statistici per confrontare gli ECG simulati a partire dalla soluzione numerica del problema dell'elettrofisiologia cardiaca con quelli realmente registrati nella pratica clinica.
La riproduzione numerica di ECG realistici ha molti vantaggi: da una parte permette di legare l'espressione sintetica dello stato cardiaco dei soggetto a una dinamica elettrica sottostante direttamente controllabile, dall'altra parte è un utile passo avanti nel tentativo di adattare i modelli numerici generali a pazienti particolari ed è anche utile per costruire delle banche dati di ECG,

sia fisiologici che patologici, su cui testare e affinare algoritmi di trattamento del segnale alla base dei dispositivi medici.
L'utilizzo di metodi statistici per la validazione delle soluzioni di modelli numerici rappresenta un punto chiave per calibrare i modelli rispetto ad un output reale in cui la variabilità sia naturalmente e intrinsecamente legata al fenomeno d'interesse. Questo è il caso, tra gli altri, dell'elettrofisiologia cardiaca, in cui l'output di interesse è costituito dai segnali ECG.

Il lavoro di tesi è organizzato come segue. Gli elementi di elettrofisiologia cardiaca alla base del meccanismo di propagazione del segnale elettrico sono descritti nel Capitolo 2, mentre nel Capitolo 3 vengono presentati i principali modelli e metodi matematici più promettenti presenti in letteratura per la sua soluzione numerica, determinando l'evoluzione temporale del potenziale elettrico nel cuore e nel torso, da cui poi costrire gli ECG sintetici. Nel Capitolo 4 viene descritto l'approccio statistico all'analisi di dati funzionali multivariati, quali possono essere modellati gli ECG, in particolare mettendo in evidenza il concetto di *profondità* di un dato rispetto al campione, che costituirà la base su cui costruire la tecnica di validazione statistica con cui determinare se i segnali reali e artificiali possano essere ritenuti provenienti da una stessa popolazione. Nel Capitolo 5 viene risolto il problema agli elementi finiti che esprime l'evoluzione del potenziale elettrico nel corpo umano, viene descritta la libreria di statistica computazionale appositamente creata per attuare le analisi di validazione sui segnali sintetici e infine vengono presentati i risultati ottenuti. Il Capitolo 6 raccoglie le conclusioni e gli sviluppi futuri.

Capitolo 2

Elementi di elettrofisiologia cardiaca

Il cuore umano è un muscolo cavo formato da fibre arrotolate ed ha la funzione di raccogliere e pompare continuamente il sangue nelle arterie. Da un punto di vista macroscopico esso è composto da quattro cavità, due atrii e due ventricoli, dotate di opportune valvole per impedire il reflusso sanguigno. La circolazione sanguigna nel corpo umano è sostenuta attraverso la continua contrazione (sistole) e il rilassamento (diastole) del tessuto cardiaco (o *miocardio*) durante i battiti. Il comportamento meccanico del cuore è governato dalle interazioni elettriche delle singole cellule, che subiscono variazioni nel potenziale elettrico transmembranale quando siano stimolate con un opportuno potenziale d'azione, innescando fenomeni di depolarizzazione e ripolarizzazione attraverso flussi di alcune specie ioniche (Ca^{++}, Na^{+}, K^{+}, ...) a cavallo della membrana cellulare. A livello del tessuto, quindi, si determina un fronte di propagazione della differenza di potenziale che percorre l'intera struttura del cuore, producendo un movimento meccanico globale, regolare e periodico.

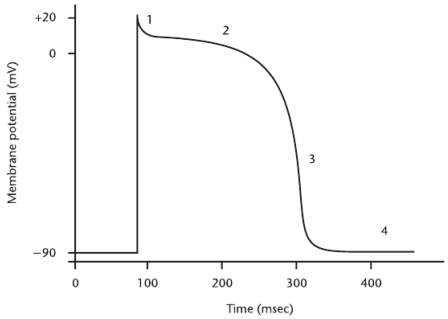

Figura 2.1: Andamento tipico del potenziale d'azione di una cellula del miocardio ventricolare.

Le cellule del miocardio in stato di riposo possiedono un potenziale transmembranale tipico di circa -80, $-90mV$ rispetto al fluido extracellulare circostante. La membrana cellulare che racchiude la cellula è permeabile ad un certo numero di ioni (ad esempio sodio e potassio), che la attraversano in corrispondenza di canali attivabili sia per mezzo di un potenziale suffi-

cientemente elevato (cioè che superi una *soglia* fisiologica caratteristica), sia dall'effetto di un recettore. Il flusso di tali ioni attraverso i canali attivati influisce sulla dinamica del potenziale transmembranale, il cui andamento tipico è mostrato in Figura 2.1. In un primo momento lo stimolo produce una rapida depolarizzazione della cellula, che superare la soglia di $-70mV$ fino ad un picco di potenziale, cui segue poi una iniziale ripolarizzazione dovuta all'attivazione dei canali di potassio. Il plateau seguente esprime un sostanziale bilancio tra il flusso di potassio in uscita e quello di sodio in ingresso, che si tramuta infine in una completa ripolarizzazione fino allo stato di riposo.

2.1 Il battito cardiaco

Il ciclo cardiaco di depolarizzazione-ripolarizzazione delle cellule del miocardio, che è alla base del battito cardiaco, viene inizializzato dalla propagazione di un potenziale d'azione che origina nel *Nodo Seno-Atriale* (NSA), una piccola regione dell'atrio destro formata da cellule che si comportano come *pacemaker* naturale per il cuore, sfruttando la loro proprietà di *automaticità*, ossia di depolarizzarsi automaticamente. Cellule di questo tipo si trovano anche in corrispondenza del *Nodo Atrio-Ventricolare* (NAV) e in certi sistemi di conduzione specializzati all'interno di atrii e ventricoli.
Nelle cellule automatiche lo stato con potenziale a riposo non è stabile e il potenziale transmembranale si innalza spontaneamente in conseguenza al formarsi di flussi di ioni attraverso i canali della membrana cellulare. Quando il potenziale supera la soglia critica si genera un potenziale d'azione che si propaga alle cellule circostanti fino alla completa depolarizzazione della cellula, sfruttando la speciale proprietà delle cellule miocardiche di diffondere direttamente il fronte d'onda. Il tempo caratteristico dell'attività delle cellule automatiche influenza la periodicità del battito cardiaco. La frequenza di depolarizzazione-ripolarizzazione intrinseca del NSA è la più alta (circa 60-100 battiti al minuto), seguita da quella del NAV (40-50 battiti al minuto), e dai muscoli ventricolari (20-40 battiti al minuto). In condizioni normali la frequenza del NSA determina il ritmo del battito cardiaco, che tuttavia può essere modificato dall'azione dei fasci nervosi del sistema simpatico e parasimpatico che innervano il muscolo cardiaco. Il sistema nervoso simpatico viene attivato in momenti di stress aumentando la frequenza del NSA, mentre il nervo vago, attraverso cui il sistema parasimpatico comunica col cuore, si attiva per diminuire il battito cardiaco, agendo sul NAV.

Il fronte di depolarizzazione originato nel NSA attraversa le cellule miocardiche degli atrii, determinandone la contrazione, fino a pervenire al NAV, dove subisce un ritardo tipico prima di essere trasmesso ai ventricoli, il che permette di separarne la contrazione da quella degli atrii. Il segnale raggiunge quindi il *fascicolo di His* e i due fascicoli della branca destra e sinistra situati nel setto interventricolare, che servono rispettivamente il ventricolo destro e sinistro del cuore. Infine il segnale giunge all'apice del cuore e alle fibre di Purkinje, che costituiscono la parte terminale del sistema di conduzione del cuore. Esse sono in grado di propagare il segnale elettrico molto velocemente e penetrano in profondità direttamente nel miocardio, inoltre possiedono una propria spontanea frequenza di pulsazione, circa 15-40 battiti al minuto, che tuttavia non impone il ritmo del battito cardiaco. Avvenuta la depolarizzazione del miocardio ventricolare si verifica un periodo di plateau, in cui non si propaga nessun potenziale d'azione, in seguito le singole cellule iniziano a ripolarizzarsi e un altro fronte d'onda attraversa il cuore, partendo dai dipoli

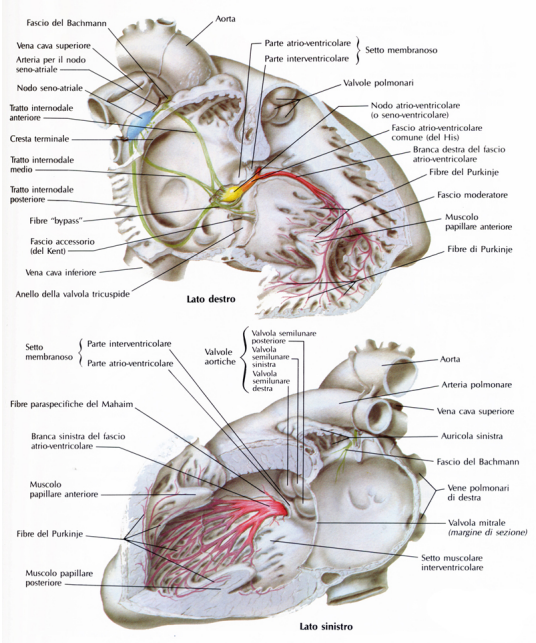

Figura 2.2: Descrizione anatomica del cuore umano.

generati all'interfaccia del tessuto depolarizzato e ripolarizzato e con direzione di propagazione opposta rispetto a quella dell'onda di depolarizzazione (ripolarizzazione ventricolare). Infine il cuore torna nello stato di riposo, in attesa di un successivo stimolo elettrico che faccia ripartire il ciclo.

2.2 L'Elettrocardiogramma (ECG)

L'attività elettrica delle cellule cardiache determina un flusso di correnti all'interno del cuore e si manifesta con una variazione della differenza di potenziale sulla superficie della pelle che può essere misurata con opportuni apparati sperimentali. L'andamento in tempo di tali differenze di potenziale costituisce l'elettrocardiogramma (ECG).

2.2.1 Sistema standard a 12 derivazioni

Da un punto di vista pratico, l'ECG è un insieme di 12 tracciati temporali di altrettante differenze di potenziale misurate tra punti differenti della superficie del corpo umano.
Esso è la proiezione su 12 direzioni nello spazio tridimensionale del vettore cardiaco risultante dei momenti dei dipoli elettrici determinati nel cuore dall'avanzamento del fronte d'onda. Ognuna delle 12 *derivazioni* dell'ECG mostra il modulo del vettore cardiaco (e quindi l'attività elettrica) nella corrispondente direzione per ogni istante di tempo. Le 12 direzioni sono scelte in maniera tale da individuare una suddivisione dello spazio che esprima l'attività nelle orientazioni destro-sinistro, superiore-inferiore, anteriore-posteriore del corpo.

Le tre derivazioni indicate con I, II e III sono chiamate *bipolari*, le tre derivazioni denominate aVL, aVR e aVF sono dette *unipolari aumentate*, mentre le sei derivazioni V1, V2, V3, V4, V5 e V6 sono chiamate *precordiali*.
Si nota subito c usare le proiezioni lungo 12 direzioni per rappresentare un vettore in uno spazio tridimensionale si abbia un'evidente ridondanza e correlazione nei tracciati delle derivazioni che rappresenta un interessante oggetto di studio.

Si può vedere in Figura 2.3 la disposizione superficiale degli elettrodi relativi alle 12 derivazioni.

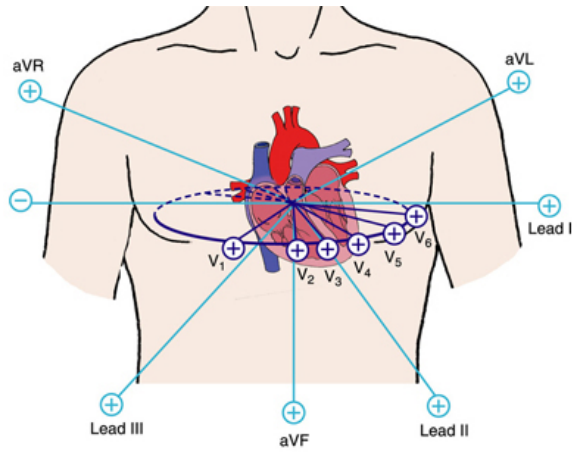

Figura 2.3: Disposizione degli elettrodi delle derivazioni precordiali e rappresentazione delle direzioni di quelle bipolari e unipolari aggiunte dell'ECG a 12 derivazioni.

Le tre derivazioni bipolari sono determinate misurando il potenziale elettrico in corrispondenza dei quattro arti, e sono definite come:

(I) $V_I = V_{LA} - V_{RA}$,

(II) $V_{II} = V_{LL} - V_{RA}$,

(III) $V_{III} = V_{LL} - V_{LA}$,

dove si sono indicati con V_{LA}, V_{RA} i valori di potenziale in corrispondenza del braccio destro e sinistro, e con V_{LL} il valore corrispondente alla gamba sinistra (si veda la Figura 2.4). Dalle definizioni si può notare come la derivazione I fornisca informazione sull'attività lungo l'asse laterale del corpo (destra-sinistra), mentre II e III siano relative all'attività lungo l'asse verticale (superiore-inferiore)
Dalla definizione precedente risulta che

$$V_I + V_{III} = V_{II},$$

di conseguenza solo due delle tre derivazioni sono indipendenti.

Le tre derivazioni unipolari aumentate, invece, sono definite a partire dagli elettrodi posti sugli arti come:

(aVF) $V_{aVF} = V_{LL} - \frac{V_{RA} + V_{LA}}{2}$,

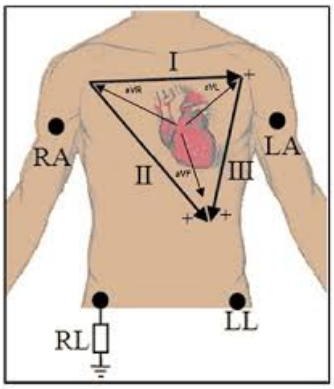

Figura 2.4: Rappresentazione della disposizione di elettrodi che definisce le derivazioni unipolari aggiunte e bipolari dell'ECG a 12 derivazioni.

(aVR) $V_{aVR} = V_{RA} - \dfrac{V_{LL} + V_{LA}}{2}$,

(aVL) $V_{aVL} = V_{LA} - \dfrac{V_{RA} + V_{LL}}{2}$.

La definizione indica che la derivazione aVF coglie l'attività relativa dell'area inferiore del corpo rispetto a quella dell'area superiore. Allo stesso modo la aVR coglie l'attività relativa destra e la aVL l'attività relativa sinistra.

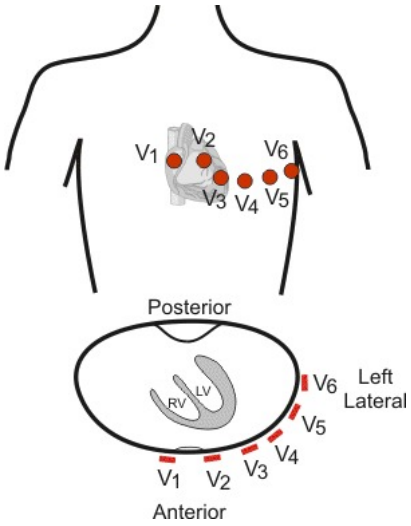

Figura 2.5: Disposizione degli elettrodi relativi alle derivazioni precordiali dell'ECG standard a 12 derivazioni.

Infine, le sei derivazioni precordiali sono determinate posizionando gli elettrodi sul torso come in Figura 2.5.

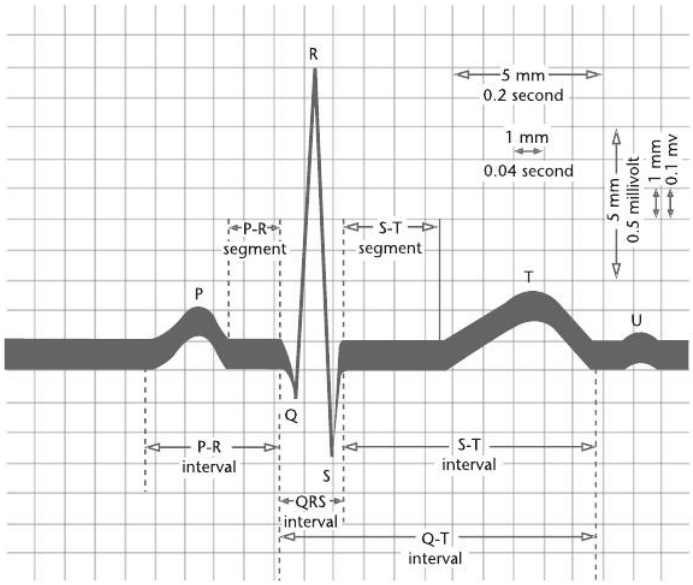

Figura 2.6: Tracciato della derivazione I in condizioni normali. Si può notare la suddivisione del segnale in segmenti corrispondenti a diverse fasi durante il battito cardiaco. Il supporto graduato permette di identificare facilmente la durata dei tratti caratteristici, la cui variabilità può essere indice di un comportamento anormale e viene analizzata dai medici.

Data la loro disposizione, esse rappresentano la proiezione del vettore cardiaco sul piano orizzontale individuato nel torso all'altezza del cuore. Le prime tre (V1, V2, V3) colgono l'attività dei lati anteriore-posteriore del corpo, mentre le seconde tre (V4, V5, V6) forniscono informazioni sull'attività lungo l'asse laterale (lati destro-sinistro).

2.2.2 Il tracciato ECG

Il principale utilizzo dell'ECG è l'analisi dei tracciati a scopo di monitoraggio o diagnosi, ricavando informazioni circa lo stato del ciclo elettrico (e in parte anche meccanico) del cuore.
Il tracciato ECG viene suddiviso in segmenti corrispondenti alle diverse fasi del battito cardiaco, come mostrato nella Figura 2.6.

Nell'ordine, i punti caratteristici del tracciato sono denominati P, Q, R, S, T ed U, e corrispondono:

(P) alla depolarizzazione atriale,

(QRS) alla depolarizzazione dei ventricoli,

(T) alla ripolarizzazione dei ventricoli,

(U) generalmente assente, è provocata da fattori meccanici relativi al rilassamento del miocardio.

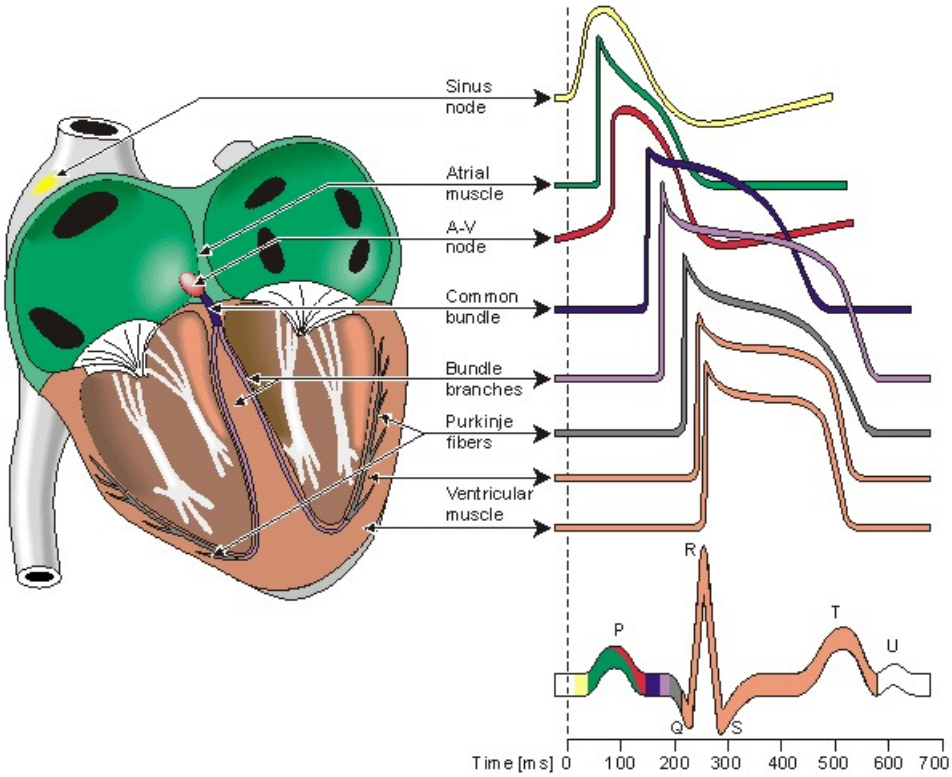

Figura 2.7: Contributi all'ECG degli elementi che partecipano alla dinamica elettrica del cuore.

In Figura 2.7 si può notare il contributo all'ECG delle varie parti del cuore in termini della loro singola evoluzione temporale, influenzata quindi dal ruolo svolto durante il ciclo elettrico del battito.

L'ECG superficiale ha delle naturali limitazioni rispetto al tentativo di ricostruire l'attività elettrica del cuore a partire dalla distribuzione di potenziale superficiale. In particolare, la ricostruzione dettagliata del comportamento elettrofisiologico della sorgente costituisce un problema mal posto (come dimostrato da Helmholtz in [Hel53]).
Di conseguenza l'analisi più approfondita dell'attività elettrica richiede un esame più diretto (e invasivo). Tuttavia, l'esame dei segmenti del tracciato ECG può fornire una prima evidenza per la presenza di certe patologie elettriche. A tal proposito, i tratti maggiormente significativi sono l'onda P, il complesso QRS e l'onda T, le cui forme, estensioni e altezze vengono subito indentificate dai medici studiando un ECG.

Le anormalità dell'attività elettrica riscontrabili attraverso l'esame dell'ECG si possono dividere, in prima approssimazione, in *bradicardia* (il cuore batte troppo lentamente o irregolarmente) e *tachicardia* (il cuore batte troppo velocemente). Una frequenza di battito troppo lenta può essere causata dall'invecchiamento del NSA o da patologie riguardanti la conducibilità dei segnali da parte del NAV. Problemi di conducibilità dei fronti d'onda atriali verso i ventricoli possono affliggere anche i fascicoli di fibre della branca destra (blocco di branca destra) o sinistra

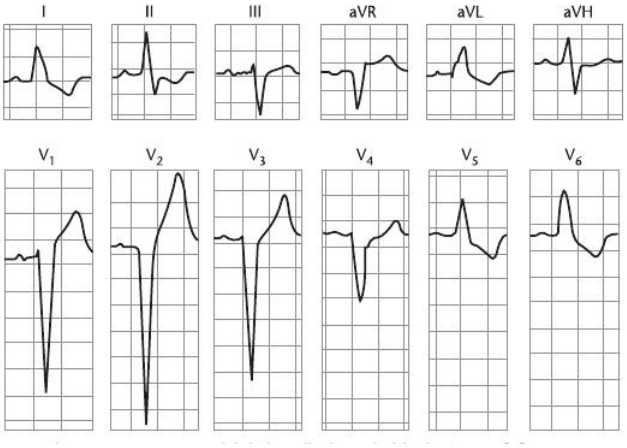

Figura 2.8: Tracciato ECG in condizioni di blocco di branca sinistra, si può notare, ad esempio, il prolungamento del complesso QRS e l'inversione dell'onda T rispetto all'ECG in condizioni normali.

(blocco di branca sinistra), che non sono più in grado di trasmettere i segnali elettrici, i quali non avendo a disposizione le fibre di trasmissione veloci si propagano con il canale più lento muscolo-muscolo.
Tali blocchi hanno in genere un effetto minimo sul pompaggio del sangue nel sistema circolatorio, tuttavia hanno un effetto molto rilevante sul comportamento del vettore cardiaco, e quindi sul tracciato ECG, come evidenziato in Figura 2.8. Il blocco di branca sinistra, ad esempio, ha l'effetto di allargare il complesso QRS dell'ECG, e di causare l'inversione dell'onda T, per cui questa viene deflessa nel verso opposto rispetto a quello della deflessione del QRS.

Capitolo 3

Modellistica numerica dell'Elettrofisiologia Cardiaca

3.1 Modelli matematici del problema dell'EFC

La modellistica matematica del problema dell'elettrofisiologia cardiaca (EFC) deve considerare sin da principio la doppia scala spaziale caratteristica alla quale avvengono i fenomeni d'interesse: come descritto nel Capitolo 2, da un lato si ha la scala microscopica cellulare, alla quale si verificano i meccanismi di eccitazione e scambio di specie ioniche alla base del ciclo elettrico cellulare; dall'altro si ha la scala macroscopica degli organi, che vengono attraversati dal segnale elettrico dando luogo all'accoppiamento elettro-meccanico del battito cardiaco.
I modelli differenziali che si occupano di queste due scale sono, rispettivamente, quelli ionici e quelli macroscopici.

3.1.1 Scala microscopica: modelli ionici

I modelli ionici hanno come obiettivo la descrizione della dinamica alla scala miscroscopica e vengono generalmente suddivisi in due categorie, a seconda della loro complessità e del loro realismo: i modelli *fisiologici* e i modelli *fenomenologici*.
In generale l'obiettivo di un modello ionico è di caratterizzare attraverso equazioni differenziali, riguardanti opportune variabili di stato, le correnti ioniche a cavallo della membrana cellulare. Generalmente si ha $I_{\text{ion}} = I_{\text{ion}}(V_m, \mathbf{w})$, dove V_m indica il potenziale transmembranale, definito come $V_m := u_i - u_e$, e $\mathbf{w}$ è un vettore di concentrazioni di altre specie ioniche e variabili da cui dipende il gating. L'evoluzione di I_{ion} nella singola cellula è descritta ad una ODE della forma:

$$\frac{\partial V_m}{\partial t} + I_{\text{ion}} + i_{\text{app}} = 0, \tag{3.1}$$

in cui i_{app} è una corrente applicata dall'esterno, mentre $\mathbf{w}$ soddisfa una ODE della forma:

$$\frac{\partial \mathbf{w}}{\partial t} + \mathbf{g}(V_m, \mathbf{w}) = 0. \tag{3.2}$$

Modelli fisiologici

Il primo modello per l'attività elettrica delle cellule eccitabili è stato proposto da Hodgkin e Huxley [HH52] ed esprime attraverso un insieme di equazioni differenziali nonlineari la dinamica delle correnti ioniche che danno luogo ai potenziali d'azione nell'assone gigante del calamaro. Negli anni seguenti tale modello è stato adattato alle cellule cardiache, in particolare quelle delle fibre di Purkinje e alle cellule pacemaker [Nob62]. La modellazione delle cellule ventricolari è stata introdotta nel 1977 [BR77]. Ulteriori miglioramenti sono stati apportati nel 1991 [LR91] e nel 1994 [LR94]. Gli sforzi recenti si concentrano verso l'adattamento dei modelli esistenti a condizioni fisiologiche tipiche di certi stati patologici e dell'attività delle cellule ventricolari.
Questi modelli sono comunque generalmente più complessi dei modelli fenomenologici, dato che si introduce un numero maggiore di variabili di stato nella modellazione di $I_{\text{ion}}(V_m, w)$, questo pone dei limiti al loro utilizzo in contesti applicativi, ad esempio quando si debba considerare l'accoppiamento con una dinamica di macroscala a livello degli organi.

Modelli fenomenologici

Un'alternativa alla complessità dei modelli fisiologici è costituita dai modelli fenomenologici. Questi usano generalmente un ridotto numero di variabili di stato e descrivono un'approssimazione della dinamica elettrica.
Un primo modello, avanzato negli anni '60, è quello di Fitzhugh-Nagumo [Fit61], una versione semplificata delle equazioni di Hodgkin-Huxley in cui si considerano solo due variabili di stato:

$$\begin{cases} I_{\text{ion}}(v,w) = v(v-\alpha)(v-1) + w \\ g(v,w) = -\epsilon(\gamma v - w) \end{cases}$$

in cui $0 < \alpha < 1$, $\epsilon \ll 1$ and $\gamma > 0$.

Successivi modelli a due variabili di stato adattati alle cellule cardiache sono, ad esempio, quello di Roger-McCulloch [RM94]:

$$\begin{cases} I_{\text{ion}}(v,w) = v(v-\alpha)(v-1) + wv \\ g(v,w) = -\epsilon(\gamma v - w) \end{cases}$$

quello di Aliev-Panfilov [AP96]:

$$\begin{cases} I_{\text{ion}}(v,w) = v(v-\alpha)(v-1) + wv \\ g(v,w) = \epsilon(\gamma v(v-1-\alpha) + w) \end{cases}$$

Un modello più complicato, con tre variabili di stato (una per il potenziale e due variabili di *gating*) e le relative equazioni di evoluzione, è quello di Fenton-Karma (si veda [FK98]), determinato con l'intento di sintetizzare un modello di complessità minimale che riproducesse quantitativamente la restituzione (cioè la relazione tra la durata del potenziale d'azione (Action Potential Duration, APD) e l'intervallo diastolico (Diastolic Interval, DI) che precede il potenziale d'azione) ottenibile da modelli fisiologici più complessi. Tale modello contiene tre correnti, corrispondenti a sodio, calcio e potassio.
Proseguendo nel lavoro di semplificazione, Mitchell e Schaeffer [MS03] hanno proposto un modello con due sole correnti, che tuttavia riproduce almeno dal punto di vista qualitativo il

comportamento di restituzione del modello Fenton-Karma.

$$I_{\text{ion}}(v,w) = \frac{w}{\tau_{\text{in}}} v^2 (v-1) - \frac{v}{\tau_{\text{out}}} \tag{3.3}$$

$$g(v,w) = \begin{cases} \dfrac{w-1}{\tau_{\text{open}}} & \text{se} \quad v \leq v_{\text{gate}} \\ \dfrac{w}{\tau_{\text{close}}} & \text{se} \quad v > v_{\text{gate}} \end{cases} \tag{3.4}$$

w, come variabile di gating adimensionale, è tale che $w = 0$ e $w = 1$ corrispondano a stati in cui i canali ionici sono, rispettivamente, chiusi e aperti. La variabile v, adimensionale, rappresenta ancora il potenziale transmembranale nella cellula.
Tra i pregi del modello di Mitchell-Schaeffer vi è sicuramente la sua semplicità, che permette di studiare analiticamente l'influenza dei vari parametri, di determinare una forma chiusa per la curva di restituzione (importante poichè usata per studiare certi tipi di aritmia) e di facilitare il calcolo durante simulazioni intensive di tessuti spazialmente estesi. Anche per queste ragioni esso è stato utilizzato in diversi lavori mirati alla simulazione realistica dell'elettrofisiologia cardiaca di macroscala (si vedano, ad esempio, [Zem09], [Bou+10], [BSG12]).

Insieme alla versione adimensionale, il modello di Mitchell-Schaeffer può essere scritto anche nella versione riscalata, nella quale si esplicita il ruolo delle costanti fisiche che caratterizzano le cellule cardiache (anch'esse oggetto di studio e modellazione):

$$I_{\text{ion}}(v,w) = -\frac{w}{\tau_{\text{in}}} \frac{(V_m - V_{\text{min}})^2 (V_{\text{max}} - V_m)}{V_{\text{max}} - V_{\text{min}}} + \frac{1}{\tau_{\text{out}}} \frac{V_m - V_{\text{min}}}{V_{\text{max}} - V_{\text{min}}} \tag{3.5}$$

$$g(V_m,w) = \begin{cases} \dfrac{w}{\tau_{\text{open}}} - \dfrac{1}{\tau_{\text{open}} (V_{\text{max}} - V_{\text{min}})^2} & \text{se} \quad V_m \leq v_{\text{gate}} \\ \dfrac{w}{\tau_{\text{close}}} & \text{se} \quad V_m > v_{\text{gate}} \end{cases} \tag{3.6}$$

dove τ_{in}, τ_{out}, τ_{open}, τ_{close} e V_{gate} sono parametri dati, mentre V_{min} e V_{max} sono costanti di riscalamento (tipicamente -80 e 20mV rispettivamente).

In [Bou+07] e [Bou+10] gli autori hanno mostrato come segnali ECG realistici possano essere ottenuti con il modello di Mitchell-Schaeffer, mentre sembra che non sia così per il caso con modello ionico di Fitzhugh-Nagumo.

3.1.2 Scala macroscopica: modelli per cuore e torso

La modellistica del tessuto cardiaco è storicamente divisa in due filoni: il filone dei modelli discreti, in cui la granularità del tessuto viene caratterizzata da una rappresentazione esplicita delle singole cellule, e quello di modelli continui, che trattano il tessuto cardiaco come un *sincizio funzionale*, cioè un reticolo molto fitto di cellule interconnesse che permette la propagazione dei potenziali d'azione come in un mezzo continuo.
Nei primi modelli discreti le cellule erano descritte come automi cellulari con un numero finito di stati, con un'evoluzione ad ogni istante temporale dipendente dal loro stato precedente e da quello delle cellule circostanti.
L'approccio continuo, chiamato *bidominio*, è stato formulato matematicamente da Tung [Tun78],

ed è divenuto uno standard per la maggior parte del lavoro di ricerca sulla modellazione dell'attività elettrica del cuore. Esso rappresenta il tessuto cardiaco come un sincizio formato dal dominio intracellulare (l'insieme delle cellule cardiache) e dal dominio extracellulare (lo spazio che le separa). Le assunzioni sono che entrambi i domini siano sovrapposti e continui, ma separati dalla membrana cellulare, e che i mezzi intra ed extra-cellulari siano dei conduttori ohmici. Dopo un processo di omogeneizzazione (si vedano [NK93] e [PSCF05]), possibile dietro l'ipotesi di una struttura periodica del miocardio, si può ritenere che i domini intracellulare ed extracellulare occupino l'intero volume del cuore Ω_H.

Seguendo [Zem09], [Bou+10] e [Cla+11], si denotino con Ω_{H_i} e Ω_{H_e}, rispettivamente, i domini intracellulare ed extracellulare, separati dalla membrana cellulare Γ_m, cosicché si abbia

$$\overline{\Omega}_H = \overline{\Omega}_{H_i} \cup \overline{\Omega}_{H_e}, \qquad \Gamma_m = \partial\Omega_{H_i} \cap \partial\Omega_{H_e}$$

Siano inoltre $\mathbf{j}_e$, $\mathbf{j}_i$, $\mathbb{G}_e$, $\mathbb{G}_i$, u_e e u_i, rispettivamente, la densità di corrente elettrica, i tensori di conducibilità elettrica e i potenziali elettrici extracellulare ed intracellulare.
Allora, dalla legge di Ohm si ottiene:

$$\begin{aligned} \mathbf{j}_i &= -\mathbb{G}_i \nabla u_i, \\ \mathbf{j}_e &= -\mathbb{G}_e \nabla u_e. \end{aligned} \tag{3.7}$$

Dalla conservazione della carica e della corrente, e assumendo che le sorgenti di carica nei due domini siano localizzate solo sulla membrana Γ_m, si ha:

$$I_m = \nabla \cdot \mathbf{n} = -\mathbf{j}_e \cdot \mathbf{n} \tag{3.8}$$

dove $\mathbf{n}$ è la normale uscente a $\partial\Omega_{H_i}$ e I_m è la densità di corrente per unità di volume attraverso Γ_m, composta da una parte capacitiva e una parte ionica, I_{ion}, dovuta all'azione di canali, pompe e scambiatori nella membrana cellulare.

$$I_m = I_{\text{ion}} + C_m \frac{\partial V_m}{\partial t} + i_{\text{app}}, \tag{3.9}$$

In cui C_m è la capacità membranale per unità di superficie e V_m è il potenziale trans-membrana, che abbiamo già definito come:

$$V_m := u_i - u_e,$$

e i_{app} è la corrente applicata da una sorgente esterna. Come menzionato nella Sottosezione 3.1.1, $\mathbf{w}$ è una variabile vettoriale che esprime le concentrazioni di altre specie ioniche e alcune variabili di gating, e obbedisce ad una relazione della forma:

$$\frac{\partial \mathbf{w}}{\partial t} + \mathbf{g}\left(V_m, \mathbf{w}\right) = 0. \tag{3.10}$$

La scelta tra i modelli ionici disponibili (Sottosezione 3.1.1) determina la forma di I_{ion} e delle equazioni di chiusura su $\mathbf{w}$.

Dopo aver proceduto all'omogeneizzazazione, sfruttando la (3.8) e considerando le (3.7) e la definizione di V_m, si ha:

$$\nabla \cdot \left(\mathbb{G}_i \nabla u_i + \mathbb{G}_e \nabla u_e\right) = 0, \qquad \text{in} \quad \Omega_H$$

da cui si ottiene:

$$\nabla \cdot ((\mathbb{G}_i + \mathbb{G}_e)\, \nabla u_e) = -\nabla \cdot (\mathbb{G}_i \nabla V_m) \quad \text{in } \Omega_H.$$

Infine inglobando le correnti ioniche:

$$A_m \left(C_m \frac{\partial V_m}{\partial t} + I_{\text{ion}}\left(V_m, \mathbf{w}\right) \right) - \nabla \cdot (\mathbb{G}_i \nabla V_m) = \nabla \cdot (\mathbb{G}_i \nabla u_e) + I_{\text{app}} \qquad \text{in } \Omega_H \tag{3.11}$$

dove A_m è una costante geometrica che rappresenta la quantità media di superficie membranale per unità di volume.

Le condizioni al bordo vanno poste sulla superficie esterna del cuore, che è formata da due parti: l'*endocardio* (Γ_{endo}), cioè la superficie interna, e l'*epicardio* (Γ_{epi}), quella esterna. Chiaramente è $\partial\Omega_H := \Sigma = \Gamma_{\text{epi}} \cup \Gamma_{\text{endo}}$.
Viene generalmente assunto che al di fuori di tali superfici la corrente intracellulare non scorra, quindi si impongono condizioni al bordo della forma:

$$\mathbb{G}_i \nabla u_i \cdot \mathbf{n} = 0 \qquad \text{su } \Sigma, \tag{3.12}$$

dove $\mathbf{n}$ è la normale uscente a Σ. Considerando la definizione di V_m, si può scrivere:

$$\mathbb{G}_i \nabla u_e \cdot \mathbf{n} = -\mathbb{G}_i \nabla V_m \cdot \mathbf{n} \qquad \text{su } \Sigma. \tag{3.13}$$

Le condizioni al bordo su u_e, invece, sono differenti a seconda di quale interazione si ipotizza tra cuore e torace (cioè il resto del corpo umano). Se si considera il cuore come mezzo elettricamente isolato, disaccoppiando la dinamica cuore e torso, si ha:

$$\mathbb{G}_e \nabla u_e \cdot \mathbf{n} = 0 \qquad \text{su } \Sigma \tag{3.14}$$

Il modello così ottenuto è detto *Modello bidominio isolato*:

$$\begin{cases} A_m \left(C_m \dfrac{\partial V_m}{\partial t} + I_{\text{ion}}\left(V_m, \mathbf{w}\right) \right) - \nabla \cdot (\mathbb{G}_i \nabla V_m) = \nabla \cdot (\mathbb{G}_i \nabla u_e) + I_{\text{app}} & \text{in } \Omega_H \times (0,T) \\ \nabla \cdot ((\mathbb{G}_i + \mathbb{G}_e)\, \nabla u_e) = -\nabla \cdot (\mathbb{G}_i \nabla V_m) & \text{in } \Omega_H \times (0,T) \\ \dfrac{\partial \mathbf{w}}{\partial t} + \mathbf{g}\left(V_m, \mathbf{w}\right) = 0 & \text{in } \Omega_H \times (0,T) \\ \mathbb{G}_i \nabla u_e \cdot \mathbf{n} = -\mathbb{G}_i \nabla V_m \cdot \mathbf{n} & \text{su } \Sigma \times (0,T) \\ (\mathbb{G}_i + \mathbb{G}_e)\, \nabla u_e \cdot \mathbf{n} = -\mathbb{G}_i \nabla V_m \cdot \mathbf{n} & \text{su } \Sigma \times (0,T) \end{cases} \tag{3.15}$$

Per poter simulare un elettrocardiogramma serve misurare il potenziale in specifici punti del torace, quindi è necessario modellare anche il comportamento del torace e accoppiarlo con quello cardiaco.
Siano Ω_T e u_T, rispettivamente, il dominio definito dal torace e il suo potenziale elettrico. Il torace viene considerato un conduttore passivo, quindi dalla legge di Ohm si ha:

$$\mathbf{j}_T = -\mathbb{G}_T \nabla u_T \qquad \text{in } \Omega_T.$$

Allora la conservazione della carica diventa

$$\nabla \cdot (\mathbb{G}_T \nabla_T) = 0 \qquad \text{in } \Omega_T$$

Chiaramente questa equazione viene completata dalla naturale richiesta che

$$\mathbb{G}_T \nabla_T \cdot \mathbf{n}_T = 0 \qquad \text{su } \Gamma_{\text{ext}},$$

dove $\mathbf{n}_T$ è il vettore normale esterno a Γ_{ext}. Da ultimo, bisogna imporre delle condizioni di trasmissione di potenziale e corrente all'interfaccia tra cuore e torace, cioè tra Ω_H e Ω_T. Si suppone la continuità di potenziale e corrente a cavallo di Σ:

$$u_T = u_e \quad e \quad \mathbb{G}_i \nabla u_i \cdot \mathbf{n} = \mathbb{G}_e \nabla u_T \cdot \mathbf{n}_T \quad \text{su } \Gamma_m \tag{3.16}$$

Dunque il *modello bidominio accoppiato* è:

$$\begin{cases} A_m \left(C_m \dfrac{\partial V_m}{\partial t} + I_{\text{ion}}(V_m, \mathbf{w}) \right) - \nabla \cdot (\mathbb{G}_i \nabla V_m) = \nabla \cdot (\mathbb{G}_i \nabla u_e) + I_{\text{app}} & \text{in } \Omega_H \times (0,T) \\ \nabla \cdot ((\mathbb{G}_i + \mathbb{G}_e) \nabla u_e) = -\nabla \cdot (\mathbb{G}_i \nabla V_m) & \text{in } \Omega_H \times (0,T) \\ \dfrac{\partial \mathbf{w}}{\partial t} + \mathbf{g}(V_m, \mathbf{w}) = 0 & \text{in } \Omega_H \times (0,T) \\ \nabla \cdot (\mathbb{G}_T \nabla u_T) = 0 & \text{in } \Omega_T \times (0,T) \end{cases} \tag{3.17}$$

con le condizioni al bordo:

$$\begin{cases} \mathbb{G}_i \nabla u_e \cdot \mathbf{n} = -\mathbb{G}_i \nabla V_m \cdot \mathbf{n} & \text{su } \Sigma \times (0,T) \\ (\mathbb{G}_i + \mathbb{G}_e) \nabla u_e \cdot \mathbf{n} = -\mathbb{G}_i \nabla V_m \cdot \mathbf{n} & \text{su } \Gamma_{\text{endo}} \times (0,T) \\ \mathbb{G}_T \nabla_T \cdot \mathbf{n}_T = 0 & \text{su } \Gamma_{\text{ext}} \times (0,T) \end{cases} \tag{3.18}$$

e le condizioni di trasmissione (3.16).

Osservazione 3.1. Per ragioni di complessità computazionale, in alcuni casi le condizioni (3.16) vengono sostituite da

$$\begin{cases} u_e = u_i & \text{su } \Sigma \times (0,T) \\ \mathbb{G}_i \nabla u_e \cdot \mathbf{n} = 0 & \text{su } \Sigma \times (0,T) \end{cases}$$

Questo equivale a trascurare il feedback elettrico del torso sul cuore, ed ha il pregio, dal punto di vista computazionale, di disaccoppiare il calcolo di (V_m, u_e) da quello di u_T. Per contro, tuttavia, questa scelta impedisce di descrivere i fenomeni per cui il torso influenza il cuore.

Nel seguito il modello di riferimento sarà il modello bidominio con accoppiamento caradio-toracico e modello ionico di Mitchell Schaeffer.

3.1.3 Buona posizione del problema dell'EFC

Consideriamo il problema dell'EFC con accoppiamento cardio-toracico:

$$\begin{cases} A_m\left(C_m\dfrac{\partial V_m}{\partial t}+I_{\text{ion}}\left(V_m,\mathbf{w}\right)\right)-\nabla\cdot\left(\mathbb{G}_i\nabla V_m\right)=\nabla\cdot\left(\mathbb{G}_i\nabla u_e\right)+I_{\text{app}} & \text{in } \Omega_H\times(0,T)\\ \nabla\cdot\left(\left(\mathbb{G}_i+\mathbb{G}_e\right)\nabla u_e\right)=-\nabla\cdot\left(\mathbb{G}_i\nabla V_m\right) & \text{in } \Omega_H\times(0,T)\\ \dfrac{\partial\mathbf{w}}{\partial t}+\mathbf{g}\left(V_m,\mathbf{w}\right)=0 & \text{in } \Omega_H\times(0,T)\\ \nabla\cdot\left(\mathbb{G}_T\nabla u_T\right)=0 & \text{in } \Omega_T\times(0,T)\\ \mathbb{G}_i\nabla u_e\cdot\mathbf{n}=-\mathbb{G}_i\nabla V_m\cdot\mathbf{n} & \text{su } \Sigma\times(0,T)\\ \left(\mathbb{G}_i+\mathbb{G}_e\right)\nabla u_e\cdot\mathbf{n}=-\mathbb{G}_i\nabla V_m\cdot\mathbf{n} & \text{su } \Gamma_{\text{endo}}\times(0,T)\\ \mathbb{G}_T\nabla_T\cdot\mathbf{n}_T=0 & \text{su } \Gamma_{\text{ext}}\times(0,T) \end{cases}$$

completato dalle condizioni iniziali:

$$V_m\left(0,\mathbf{x}\right)=V_0\left(\mathbf{x}\right),\quad \mathbf{w}\left(0,\mathbf{x}\right)=\mathbf{w}_0\left(\mathbf{x}\right)\qquad\forall\,\mathbf{x}\in\Omega_H.$$

Si nota da subito che i potenziali u_e e u_T sono definiti a meno di una costante, che può essere fissata a meno di una costante chiedendo che

$$\int_{\Omega_H}u_e=0.$$

Un'analisi di buona posizione del modello bidominio è stata portata avanti sotto l'ipotesi di modelli ionici semplificati, come il modello Fitzhugh-Nagumo o un modello con sola dipendenza dal potenziale transmembrana V_m.
In [Bou+08], invece, gli autori affrontano la buona posizione del problema (3.1.3) per una classe astratta di modelli ionici nei quali ricadono Fitzhugh-Nagumo, Aliev-Panfilov, Roger-McCulloch e una versione regolarizzata del modello di Mitchell-Schaeffer:

$$\begin{cases} I_{\text{ion}}\left(V_m,w\right)=\dfrac{w}{\tau_{\text{in}}}f_1\left(V_m\right)-\dfrac{V_m}{\tau_{\text{out}}},\\ g\left(V_m,w\right)=\left(\dfrac{1}{\tau_{\text{close}}}+\dfrac{\tau_{\text{close}}-\tau_{\text{open}}}{\tau_{\text{close}}\tau_{\text{open}}}h_\infty(V_m)\right)\left(w-h_\infty(V_m)\right) \end{cases}$$

dove f_1 è una funzione reale (sulla quale vengono specificate in seguito delle condizioni) e

$$h_\infty\left(V_m\right)=\left[1-\tanh\left(\frac{V_m-V_{\text{gate}}}{\eta_{\text{gate}}}\right)\right],$$

con η_{gate} costante.

Il risultato principale dell'analisi di buona posizione, ottenuto con una tecnica di tipo Faedo-Galerkin, mostra che, sotto ulteriori ipotesi secondarie per cui si rimanda direttamente a [Bou+08], esiste una soluzione debole al problema (3.1.3) con modello ionico di Mitchell-Schaeffer, dalla

seguente regolarità:

$$
\begin{aligned}
&V_m \in L^\infty\left(0,T;H^1\left(\Omega_H\right)\right) \cap H^1\left(0,T;L^2\left(\Omega_H\right)\right),\\
&u \in L^\infty\left(0,T;\mathcal{V}\right),\\
&w \in H^1\left(0,T;L^2\left(\Omega_H\right)\right) \cap W^{1,\infty}\Big(0,T;L^\infty\left(\Omega_H\right)\Big),
\end{aligned}
$$

dove

$$
u := \begin{cases} u_e & \text{in } \Omega_H,\\ u_T & \text{in } \Omega_T \end{cases}
$$

e

$$
\mathcal{V} := \left\{\phi \in H^1\Big(\Omega\Big) : \int_{\Omega_H} \phi = 0\right\}, \qquad \Omega := \Omega_H \cup \Omega_T.
$$

3.2 Soluzione numerica del problema dell'EFC

Seguendo l'approccio delineato in [Bou+07], [Zem09] e [BSG12], la soluzione numerica del problema (3.1.3) viene svolta attraverso un metodo Galerkin-FEM.

A tal fine è necessaria una formulazione debole del problema, che conformemente a quanto mostrato in [Bou+08] e ai risultati di regolarità espressi nella Sottosezione 3.1.3, diventa:

Per ogni $t > 0$, trovare $V_m(\cdot,t) \in H^1(\Omega_H)$, $w(\cdot,t) \in L^\infty(\Omega_H)$ e $u(\cdot,t) \in H^1(\Omega)$, con $\int_{\Omega_H} u = 0$, tale che:

$$
\begin{cases}
A_m \displaystyle\int_{\Omega_H} \left(C_m + \frac{\partial V_m}{\partial t} + I_{\text{ion}}\left(V_m, w\right)\right)\phi + \int_{\Omega_H} \mathbb{G}_i \nabla\left(V_m + u\right)\cdot\nabla\phi \\
\qquad\qquad = A_m \displaystyle\int_{\Omega_H} I_{\text{app}}\phi, \\
\displaystyle\int_{\Omega_H} \left(\mathbb{G}_i + \mathbb{G}_e\right)\nabla u\cdot\nabla\psi + \int_{\Omega_H} \mathbb{G}_i \nabla V_m\cdot\nabla\psi + \int_{\Omega_T} \mathbb{G}_T \nabla u\cdot\nabla\psi = 0, \\
\qquad\qquad \dfrac{\partial w}{\partial t} + g\left(V_m, w\right) = 0
\end{cases}
\tag{3.19}
$$

per tutte le coppie $(\phi,\psi) \in H^1(\Omega_H) \times H^1(\Omega)$, con $\int_{\Omega_H} \psi = 0$.

Osservazione 3.2. I potenziali u_e e u_T vengono determinati ponendo $u_e = u|_{\Omega_H}$ e $u_T = u|_{\Omega_T}$. Si noti anche come la condizione di accoppiamento (3.16) sia incorporata in maniera naturale nella formulazione.

La formulazione semi-discreta in spazio è determinata introducendo gli spazi di approssimazione finito-dimensionali di funzioni affini, continue a tratti $V_h \subset H^1(\Omega_H)$ e $W_h \subset H^1(\Omega)$. L'avanzamento in tempo sfrutta uno schema BDF implicito del second'ordine con trattamento esplicito della corrente ionica I_{ion}.
Sia dunque $N \in \mathbb{N}$, e si consideri una partizione dell'intervallo $[0,T]$, $\{[t_n, t_{n+1}]\}$, con $t_0 = 0$ e $t_n := N\delta t$, e un passo temporale $\delta t = T/N$.

Allora, per ogni $0 \leq n \leq N-1$, la soluzione $\left(V_m^{n+1}, u^{n+1}, w^{n+1}\right)$ è determinata a partire da quella al passo precedente, $\left(V_m^n, u^n, w^n\right)$ come segue:

1. Estrapolare (al secondo ordine)

$$\widetilde{V_m^{n+1}} = 2V_m^n - V_m^{n-1}$$

2. Risolvere rispetto a $w^{n+1} \in V_h$

$$\frac{1}{\delta t}\left(\frac{3}{2}w^{n+1} - 2w^n + \frac{1}{2}w^{n-1}\right) + g\left(\widetilde{V_m^{n+1}}, w^{n+1}\right) = 0$$

3. Valutare la corrente ionica $I_{\text{ion}}\left(\widetilde{V_m^{n+1}}, w^{n+1}\right)$

4. Risolvere rispetto a $\left(V_m^{n+1}, u^{n+1}\right) \in V_h \times W_h$, con $\int_{\Omega_H} u^{n+1} = 0$

$$\begin{cases} A_m \displaystyle\int_{\Omega_H} \frac{C_m}{\delta t}\left(\frac{3}{2}V_m^{n+1} - 2V_m^n + \frac{1}{2}V_m^{n-1}\right)\phi + \int_{\Omega_H} \mathbb{G}_i \nabla\left(V_m^{n+1} + u^{n+1}\right)\cdot\nabla\phi \\ \qquad\qquad = A_m \displaystyle\int_{\Omega_H}\left(I_{\text{app}}(t_{n+1}) - I_{\text{ion}}\left(\widetilde{V_m^{n+1}}, w^{n+1}\right)\right)\phi, \\ \displaystyle\int_{\Omega_H}\left(\mathbb{G}_i + \mathbb{G}_e\right)\nabla u^{n+1}\cdot\nabla\psi + \int_{\Omega_H}\mathbb{G}_i\nabla V_m^{n+1}\cdot\nabla\psi + \int_{\Omega_T}\mathbb{G}_T\nabla u^{n+1}\cdot\nabla\psi = 0 \end{cases} \tag{3.20}$$

   per tutti $(\phi, \psi) \in V_h \times W_h$ con $\int_{\Omega_H}\psi = 0$.

5. Infine, aggiornare $u_e^{n+1} = u^{n+1}|_{\Omega_H}$ e $u_T^{n+1} = u^{n+1}|_{\Omega_T}$

Il suddetto algoritmo, semi-implicito, ha il vantaggio, attraverso il passo di estrapolazione, di disaccoppiare la soluzione della ODE riguardante la variabile di recupero w dalla soluzione del potenziale nel cuore e nel torso.

Osservazione 3.3. Ad ogni time-step, il problema (3.20) richiede la soluzione accoppiata del potenziale transmembranale V_m e del potenziale u, cardio-toracico. Questo può essere effettuato risolvendo in blocco l'intero problema algebrico generato, che però può avere una dimensione anche molto grande su mesh fisiologicamente realistiche.
Un'alternativa può essere quella di partizionare la soluzione sfruttando un algoritmo di *domanin-decomposition* iterativo che permetta di separare la soluzione nel cuore da quella nel torso. Una formulazione di questo tipo, che sfrutta l'algoritmo di Dirichlet-Neumann con un'opportuna accelerazione è riportato in [Bou+10].

Capitolo 4

Analisi statistica non-parametrica degli ECG

Una possibile modellizzazione di carattere statistico definisce l'insieme di curve dell'ECG standard a 12 derivazioni come *dato funzionale multivariato* con componenti fortemente correlate tra loro, come già menzionato nella Sezione 2.2.1.

Si consideri una quantità di interesse $\mathbf{f} = (f_1, \ldots, f_s)$ e un dominio $I \subset \mathbb{R}$, spazio di variazione dei parametri, in corrispondenza dei quali vengono misurate le diverse realizzazioni della quantità $\mathbf{f}$,

$$(\mathbf{f}_1, x_1), \ldots, (\mathbf{f}_P, x_P).$$

Quando la misurazione della quantità $\mathbf{f}$ rispetto ad $x \in I$ dia luogo ad una dipendenza sufficientemente regolare, vi siano ragioni legate al fenomeno in esame per ritenere tale dipendenza continua e si voglia sfruttare un numero P molto grande di misurazioni discrete che rendono di fatto impraticabile l'analisi multivariata standard, conviene applicare ad $\mathbf{f}$ il modello di dato funzionale (se $s > 1$ multivariato):

$$\mathbf{f} : I \subset \mathbb{R} \longrightarrow \mathbb{R}^s.$$

La forza di questa modellizzazione sta nella possibilità di analizzare la dipendenza continua di $\mathbf{f}$ da x, specialmente in un contesto stocastico, tipico dell'analisi statistica, in cui le misurazioni incorporino una variabilità a priori incognita, e nella possibilità di considerare nelle analisi quantità differenziali naturalmente messe a disposizione dall'assunzione funzionale, come primitive o derivate, che possono evidenziare facilmente informazioni altrimenti inaccessibili.
Le tradizionali tecniche di indagine statistica, a partire dalla sintesi del dato attraverso media campionaria e covarianza, fino alla riduzione dimensionale, ai modelli di regressione e alla classificazione, possono essere opportunamente estese al contesto di dati funzionali per ottenere informazioni di carattere continuo, anziché discreto, sui dati. Per la trattazione di queste estensioni si veda, ad esempio, [RS05].
La trasformazione dei dati grezzi in dati funzionali processabili statisticamente passa da una prima fase di filtraggio e *smoothing*, in cui contemporaneamente si elimina l'eventuale rumore

dovuto alle misurazioni e si ricostruisce il dato funzionale a partire dalle osservazioni puntuali, a cui può seguire la separazione della cosiddetta variabilità di *fase*, cioè la variabilità orizzontale delle curve, dalla variabilità di *ampiezza*, ossia quella verticale, tramite la *registrazione* dei dati.

Nel caso degli ECG a 12 derivazioni si considera $s = 12$, dove ad x si sostituisce il tempo t e $I \subset \mathbb{R}$ è un sottoinsieme dell'asse temporale su cui si misura l'ECG.

4.1 Misure di profondità per dati funzionali

La banda di profondità statistica (BD) di un dato funzionale è una misura non-parametrica che esprime la *centralità* di quell'osservazione rispetto ad un fissato dataset funzionale o alla distribuzione di una popolazione. Essa può essere utilizzata per ordinare il dataset a partire dai segnali più centrali (o profondi), verso quelli meno centrali, e permette:

1. di costruire *boxplot funzionali*, cioè l'analogo funzionale dei boxplot tradizionali, con cui visualizzare la dispersione del dataset e identificare eventuali *outlier* (si vedano, ad esempio, [SG11] e [SG12]);
2. di estendere al contesto funzionale le statistiche d'ordine e poter applicare i metodi non-parametrici univariati basati su di esse (si vedano, ad esempio, [LPR09] e [IP13]).

Il concetto statistico di profondità è stato creato per generalizzare al caso di osservazioni multivariate idee come la mediana, le statistiche d'ordine o i ranghi, ben definite per i dati univariati. In letteratura sono state avanzate ed analizzate varie definizioni, che permettono di costruire robuste tecniche di inferenza non-parametriche per osservazioni finito-dimensionali (si vedano, ad esempio, [Tuk74], [Liu90], [LS93], [ZS00] e [Zuo03]).
In [FM01], [LPR06] e [LPR09] vengono proposte delle estensioni al caso funzionale, esaminando le relazioni coni il caso vettoriale, insieme alle proprietà asintotiche.

4.1.1 Il caso univariato

Si consideri una collezione di funzioni reali, definite su un sottoinsieme compatto $I \subset \mathbb{R}$,

$$(f_1, \dots, f_N) \quad \text{t.c.} \quad f_i : I \subset \mathbb{R} \text{ compatto, e } f_i \in C^0(I) \quad \forall i = 1 \dots N.$$

Si indica con $G(f) = \{(t,\ f(t)),\ \ t \in I\}$ il grafico della funzione f, e con

$$B(f_{i_1}, f_{i_2}, \dots, f_{i_k}) = \left\{ (t,y) : t \in I,\ \min_{r=1,\dots,k} f_{i_r}(t) \le y \le \max_{p=1,\dots,k} f_{i_p}(t) \right\}, \quad k \le N$$

l'inviluppo delle funzioni $f_{i_1}, f_{i_2}, \dots, f_{i_k}$. Per ogni funzione $f \in (f_1, \dots, f_N)$, scelto un valore j tale che $2 \le j \le N$, si definisce

$$\mathrm{BD}_N^{(j)}(f) = \binom{N}{j}^{-1} \sum_{1 \le i_1 < i_2 < \dots < i_j \le N} \mathbb{I}\{G(f) \subseteq B(f_{i_1}, \dots, f_{i_j})\}, \tag{4.1}$$

dove $\mathbb{I}(A)$ rappresenta la funzione indicatrice di A. La 4.1 esprime la proporzione di bande $B(f_{i_1}, \dots, f_{i_j})$, determinate a partire da j elementi della collezione, che contengono il grafico di f.

Definizione 4.1. Sia $J \in \mathbb{N}$ un valore fissato, con $2 \leq J \leq N$, allora si definisce la banda di profondità di ciascuna curva della collezione $(f_1, \ldots, f_N)$:

$$\mathrm{BD}_N^J(f) = \sum_{j=2}^{J} \mathrm{BD}_N^{(j)}(f), \tag{4.2}$$

se $F_1, F_2, \ldots, F_N$ sono copie indipendenti del processo stocastico F che genera le osservazioni $(f_1, f_2, \ldots, f_N)$, le corrispondenti versioni per popolazioni sono:

$$\mathrm{BD}^{(j)}(f, \mathbb{P}) = \mathbb{P}\left(G(f) \subseteq B(F_1, F_2, \ldots, F_j)\right), \tag{4.3}$$

$$\mathrm{BD}^{J}(f, \mathbb{P}) = \sum_{j=2}^{J} \mathrm{BD}^{(j)}(f, \mathbb{P}). \tag{4.4}$$

Nella determinazione di $\mathrm{BD}_N^{(j)}(f)$ si ha che l'indicatrice produce un contributo non nullo per il fissato inviluppo $B(f_{i_1}, \ldots, f_{i_j})$ solo se il grafico della curva f vi è contenuto lungo l'intero intervallo I. In generale, calcolando le profondità in casi applicativi, questo causa l'attribuzione di valori di profondità bassi e prossimi tra loro a molte curve, dando luogo al formarsi di ampie *code*.
Per evitare questo fenomeno si può modificare la definizione appena introdotta proponendo una misura di profondità *modificata* che tenga conto della frazione dell'intervallo I in cui il grafico di f è contenuto negli inviluppi.
Si consideri $j \in \mathbb{N}$, $2 \leq j \leq N$ e un elemento f del campione $(f_1, \ldots, f_N)$, allora si costruisce l'insieme

$$\begin{aligned} A_j(f) &= A\left(f;\, f_{i_1}, f_{i_2}, \ldots, f_{i_j}\right) \\ &= \left\{ t \in I : \min_{r=i_1,\ldots,i_j} f_r(t) \leq f(t) \leq \max_{p=i_1,\ldots,i_j} f_p(t) \right\} \end{aligned}$$

che esprime il sottoinsieme di I in cui il grafico di f è contennuto nell'inviluppo della j-pla $\left(f_{i_1}, f_{i_2}, \ldots, f_{i_j}\right)$.
Indicando con λ la misura di Lebesgue sull'intervallo I,

$$\lambda_r\left(A_j(f)\right) = \frac{\lambda\left(A_j(f)\right)}{\lambda\left(I\right)}$$

è la misura di $A_j(f)$ relativa a I.
Allora si può dare la seguente definizione:

Definizione 4.2. Sia $J \in \mathbb{N}$ un valore fissato, con $2 \leq J \leq N$, si definisce la misura di profondità

modificata di ciascuna curva della collezione $(f_1, \ldots, f_N)$:

$$\mathrm{MBD}_N^{(j)}(f) = \binom{N}{j}^{-1} \sum_{1 \le i_1 < i_2 < \ldots < i_j \le N} \lambda_r \left(A(f; f_{i_1}, f_{i_2}, \ldots, f_{i_j}) \right),$$

$$\mathrm{MBD}_N^{J}(f) = \sum_{j=2}^{J} \mathrm{MBD}_N^{(j)}(f).$$

se $F_1, F_2, \ldots, F_N$ sono copie indipendenti del processo stocastico F che genera le osservazioni $(f_1, f_2, \ldots, f_N)$, le corrispondenti versioni per popolazioni sono:

$$\mathrm{MBD}^{J}(f, \mathbb{P}) = \sum_{j=2}^{J} \mathrm{MBD}^{(j)}(f, \mathbb{P}),$$

$$\mathrm{MBD}^{(j)}(f, \mathbb{P}) = \mathbb{E}\left[\lambda_r \left(A(f; F_1, F_2, \ldots, F_j) \right) \right].$$

Osservazione 4.1. Le misure di profondità (non modificate) sono maggiormente dipendenti dalla forma delle curve considerate, e sono più restrittive delle versioni modificate, causando di conseguenza la presenza di molte curve con stessa profondità. Le misure di profondità modificate, invece, sono più sensibili all'ampiezza o alla grandezza del segnale che alla loro forma. Un'altra differenza rilevante tra loro è rappresentata dal comportamento rispetto alle curve che presentano tratti irregolari, in cui il segnale si allontana bruscamente, ma per poco tempo, dal comportamento medio del dataset. Questo fa sì che la curva si trovi al di fuori della regione centrale per un breve lasso di tempo. La misura di profondità non modificata associerà alla curva un valore molto basso (curva poco centrale), mentre la misura di profondità modificata può ancora essere grande (curva centrale), in base al comportamento nella restante parte di dominio.
L'uso della corretta misura di profondità dipende dal tipo di funzioni considerate durante l'analisi e dall'obiettivo finale. Se le curve sono molto irregolari può convenire usare la versione modificata, per evitare di avere molte code poco profonde e poiché potrebbe non esserci una forma rappresentativa; altrimenti, se le curve sono regolari e si desidera cogliere le variazioni rispetto ad una particolare forma la versione non modificata è più adeguata.

4.1.2 Il caso multivariato

Una possibile estensione, che verrà adottata nel seguito, delle misure di profondità di dati funzionali al caso multivariato è proposta in [IP13].

Definizione 4.3. Sia $\mathbf{F}$ un processo stocastico con legge $\mathbb{P}$ a valori nello spazio $C^0(I; \mathbb{R}^s)$ delle funzioni continue $\mathbf{f} = (f_1, \ldots, f_s) : I \longrightarrow \mathbb{R}^s$, $J \in \mathbb{N}$. Definiamo misura di profondità multivariata la quantità:

$$\mathrm{BD}_{P_{\mathbf{F}}}^{J}(\mathbf{f}) = \sum_{k=1}^{s} p_k \, \mathrm{BD}_{P_{F_{\cdot,k}}}^{J}(f_k), \quad p_k > 0 \ \ \forall k = 1, \ldots, s, \quad \sum_{k=1}^{s} p_k = 1 \tag{4.5}$$

dove viene utilizzata la definizione

$$\mathrm{BD}^J_{P_{F_{\cdot,k}}}(f_k) = \sum_{j=2}^{J} \mathbb{P}\left\{G(f_k) \subset B\left(F_{1,k}, F_{2,k}, \ldots, F_{j,k}\right)\right\} \tag{4.6}$$

in cui $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_j$ sono copie indipendenti del processo stocastico $\mathbf{F}$.

Questa è una delle possibili estensioni del concetto di banda di profontià al caso multivariato funzionale. Tale definizione permette di prendere in considerazione tutte le componenti del dato multivariato. In generale, tuttavia, la scelta dei pesi $\{p_k\}_{k=1}^s$ è influenzata dal problema in esame. In particolare i pesi dovrebbero incorporare le informazioni a priori sul fenomeno, e le eventuali relazioni di interdipendenza e la struttura di correlazione tra le componenti. La libertà nella scelta dei pesi causa una mancanza di unicità nella definizione (4.5), ma assicura una grande adattabilità al problema reale in studio.

Sia ora $\mathbf{F}$ un processo aleatorio multivariato tale che

$$\mathbb{P}\left(\min_{k=1,\ldots,s} \|F_k\| > M\right) \longrightarrow 0 \quad \text{se} \quad M \longrightarrow +\infty.$$

Nelle ipotesi precedenti vale il seguente teorema, per la cui dimostrazione si rinvia a [IP13]:

Teorema 4.1. *Sia* $\mathbf{f} : I \longrightarrow \mathbb{R}^s$, *con* $\mathbf{f} \in C^0(I; \mathbb{R}^s)$, *allora:*

(i) *Sia* $T(\mathbf{f}) = \mathbb{A}(t)\mathbf{f}(t) + \mathbf{b}(t)$, *dove* $\forall t \in I$ $\mathbf{A}(t)$ *è una matrice diagonale* $s \times s$ *tale che* $\mathbf{A}_{kk}(t)$ *siano funzioni continue in* I, $\mathbf{A}_{kk}(t) \neq 0 \ \forall t \in I$, *e dove* $\mathbf{b}(t) \in C^0(I; \mathbb{R}^s)$. *Allora:*

$$BD^J_{P_{T(\mathbf{F})}}(T(\mathbf{f})) = BD^J_{P_{\mathbf{F}}}(\mathbf{f}).$$

(ii)
$$BD^J_{P_{\mathbf{F}(g(t))}}(\mathbf{f}(g(t))) = BD^J_{P_{\mathbf{F}(t)}}(\mathbf{f}(t)),$$

dove $g : I \longrightarrow I$ *è una funzione biunivoca.*

(iii)
$$\sup_{\min_{k=1,\ldots,s} \|f_k\|_\infty > M} BD^J_{P_{\mathbf{F}}}(\mathbf{f}) \longrightarrow 0 \quad se \quad M \longrightarrow +\infty.$$

(iv) *Se* $\forall k = 1, \ldots, s$ *la distribuzione di probabilità* P_{F_k} *su* $C^0(I)$ *ha distribuzioni marginali assolutamente continue, allora* $BD^J_{P_{\mathbf{F}}}$ *è un funzionale continuo su* $C^0(I; \mathbb{R}^s)$

Se $\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_N$ sono copie indipendenti del processo stocastico $\mathbf{F}$, è possibile introdurre la versione campionaria di (4.5), utile per condurre analisi descrittive ed inferenziali su un insieme di dati funzionali multivariati $\mathbf{f}_1, \ldots, \mathbf{f}_N$ generati dal processo $\mathbf{F}$.
Per ogni $\mathbf{f}$ nel campione, calcoliamo la sua misura di profondità come:

$$\mathrm{BD}^J_N(\mathbf{f}) = \sum_{k=1}^{s} p_k \mathrm{BD}^J_{N,k}(f_k), \tag{4.7}$$

in cui a sua volta si definisce

$$\mathrm{BD}^J_{N,k}(f_k) = \sum_{j=2}^{J} \binom{N}{j}^{-1} \sum_{1\le i_1<i_2<\ldots<i_j\le N} \mathbb{I}\left\{G(f_k) \subset B\left(f_{i_1,k},\ldots,f_{i_j,k}\right)\right\}. \tag{4.8}$$

Rispetto alla versione campionaria, vale la seguente proposizione (vedi [IP13]):

Proposizione 4.2. *La versione campionaria della profondità multivariata funzionale è consistente, cioè*

$$|\mathrm{BD}^J_N(\mathbf{f}) - \mathrm{BD}^J_{P_\mathbf{F}}(\mathbf{f})| \longrightarrow 0, \quad q.c.\ se\ \ N \longrightarrow +\infty.$$

Analogamente a quanto fatto per le misure di profondità univariate, è possibile modificare la (4.7) introducendo le misure di profondità multivariate modificate, per mitigare la richiesta di inclusione delle $\mathbf{f}$ all'interno degli inviluppi.

Definizione 4.4. Con riferimento allo stesso contesto della definizione (4.7), definiamo la profondità multivariata modificata di una curva $\mathbf{f} : I \longrightarrow \mathbb{R}^s$:

$$\mathrm{MBD}^J_{N,k}(f_k) = \sum_{j=2}^{J} \binom{N}{j}^{-1} \sum_{1\le i_1<i_2<\ldots<i_j\le N} \lambda_r\left\{A(f_k; f_{i_1,k},\ldots,f_{i_j,k})\right\},$$

$$\mathrm{MBD}^J_N(\mathbf{f}) = \sum_{k=1}^{s} p_k\, \mathrm{MBD}^J_{N,k}(f_k)$$

dove, analogamente al caso delle bande di profondità univariate, si è indicato:

$$\lambda_r(A) = \frac{\lambda(A)}{\lambda(I)}, \quad \forall A \subseteq I \quad \text{Lebesgue misurabile}$$

$$A(f;\, f_{i_1}, f_{i_2},\ldots,f_{i_j}) = \left\{t \in I : \min_{r=i_1,\ldots,i_j} f_r(t) \le f(t) \le \max_{p=i_1,\ldots,i_j} f_p(t)\right\}$$

Come nel caso delle misure di profondità univariate modificate, anche le profondità multivariate modificate evitano la comparsa nel dataset di code dalla profondità molto bassa e sono più indicate nel caso di analisi che oltre alla forma delle funzioni debbano considerare anche la loro ampiezza.

4.1.3 Boxplot funzionali

La possibilità fornita dalle misure di profondità BD e MBD (campionarie) di ordinare un dataset funzionale dal centro agli estremi permette di estendere l'uso dei boxplot anche a dati funzionali, utilizzando quindi come variabile di ordinamento un indice di sintesi di natura intrinsecamente funzionale che misura in maniera naturale proprio quanto una curva si allontani dalla parte centrale dei dati (si vedano, ad esempio, [SG11] e [SG12]).
Insieme alla costruzione di boxplot, le profondità possono essere usate anche per determinare la presenza di outliers e quindi irrobustire un dataset in vista, ad esempio, di ulteriori analisi di

classificazione.

La procedura di costruzione di un boxplot funzionale, nel caso generale di dataset funzionale multivariato, $(\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_N)$, è la seguente:

1. Calcolare $\mathrm{MBD}^J(\mathbf{f}_i) \quad \forall i = 1, 2, \ldots, N$
2. Ordinare le funzioni $\mathbf{f}_i$ rispetto al valore delle MBD, e marcare come outlier le curve che per almeno un $t \in I$ escono dalla regione ottenuta dilatando di un fattore h l'inviluppo determinato dall'α% dei dati più profondi. In particolare, la α%-regione dei dati più profondi nella componente k è definita come

$$\mathcal{C}_\alpha = \left\{ (t, y(t)) : \min_{r=1,\ldots,\lceil \alpha N \rceil} f_{r;k} \leq y(t) \leq \max_{s=1,\ldots,\lceil \alpha N \rceil} f_{s;k} \right\}.$$

   Tipicamente si pone $\alpha = 50$ e $h = 1.5$.
3. Visualizzare il boxplot funzionale in ognuna delle s componenti, costruendo l'inviluppo dell'α% dei dati più profondi scelto, e marcando opportunamente gli outliers.
   Si noti che, rispetto alla definizione qui adottata, le curve marcate outlier possono non uscire dalla regione centrale in alcuni degli s boxplot marginali, poiché l'etichetta di outlier viene attribuita secondo un criterio globale che tenga in considerazione il comportamento in tutte le componenti.

Si può osservare un esempio del risultato di tale procedura su un dataset artificiale in Figura 4.1.

Osservazione 4.2. Si osservi che, mentre nel caso univariato numerico, o anche multivariato vettoriale, la caratterizzazione degli outlier è piuttosto naturale, nell'ambito funzionale essa non è univoca e viene adattata ai vari contesti, essendo possibili diversi comportamenti strutturalmente differenti per cui una curva possa essere distante dalla regione centrale:

(a) distanza in termini di ampiezza,

(b) distanza in termini di fase,

(c) distanza generica dovuta ad una morfologia differente.

Inoltre, nel caso di misure di profondità per dati multivariati la denominazione di outlier deve essere conforme alla scelta dei suddetti pesi $\{p_k\}_{k=1}^s$, che devono appunto essere individuati per pesare in maniera opportuna i comportamenti, anche anomali, nelle varie componenti. Se si assume che le esse siano tutte ugualmente significative, ad esempio, i pesi possono essere scelti uniformi, $p_k = 1/s \; \forall k = 1, \ldots, s$, e una curva può essere considerata un outlier *multivariato* se almeno una delle componenti è un outlier *univariato*.

4.1.4 Test dei ranghi di Wilcoxon

Seguendo l'approccio delineato in [IP13], è possibile sfruttare i ranghi indotti dalle MBD per generalizzare il test di Wilcoxon a dataset funzionali in cui i dati provengano da distribuzioni possibilmente differenti.

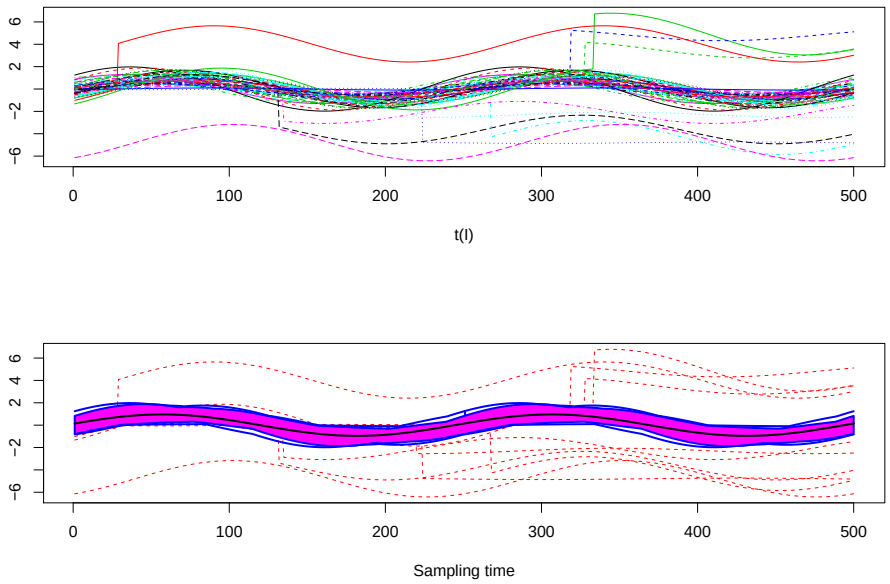

Figura 4.1: Esempio di boxplot funzionale per una famiglia di segnali con variabilità di ampiezza e fase, si noti la presenza di alcuni outliers di ampiezza.

Si consideri un campione $(\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_N)$ generato secondo una legge $\mathbb{P}_{\mathbf{X}}$, e un secondo campione, $(\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_M)$ generato a partire da una distrubuzione $\mathbb{P}_{\mathbf{Y}}$, e si voglia testare la differenza tra le due popolazioni, avendo a disposizione un terzo campione considerato come riferimento, $(\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_L)$, facente parte di una delle due popolazioni, ad esempio $\mathbb{P}_{\mathbf{X}}$.
Si calcolino le profondità e i ranghi delle popolazioni $\{\mathbf{f}\}$ e $\{\mathbf{g}\}$ rispetto alla popolazione $\{\mathbf{h}\}$ e si denoti con $R(\mathbf{f}_i)$, $i = 1, \ldots, N$ la proporzione delle $\{\mathbf{h}\}$ con profondità minore o uguale a quella di $\mathbf{f}_i$ e si proceda analogamente per $R(\mathbf{g}_j)$, $j = 1, \ldots, M$.
Si consideri allora l'insieme

$$\mathcal{R} = \{R(\mathbf{f}_1), \ldots, R(\mathbf{f}_N), R(\mathbf{g}_1), \ldots, R(\mathbf{g}_M)\}$$

e lo si ordinini in maniera crescente, attribuendo ai segnali un rango compreso tra 1 e $N + M$. Questo induce un ordinamento sul campione di dati $(\mathbf{f}_1, \ldots, \mathbf{f}_N, \mathbf{g}_1, \ldots, \mathbf{g}_M)$. Come illustrato in [LS93], si può applicare il test di Wilcoxon sui ranghi indotti da questo ordinamento. Si noti che il rango di un elemento del dataset è tanto più basso quanto più bassa è la sua profondità. La statistica test considerata è la somma dei ranghi indotti sul secondo campione, $\{R(\mathbf{g}_1), \ldots, R(\mathbf{g}_M)\}$. Come ipotesi nulla ($H_0$) si considera che non ci siano differenze tra le distribuzioni che generano i dati, e viene rifiutata per valori bassi della statistica test.
Per valori grandi di N e M si può usare un'approssimazione normale (si veda [LL04]), mentre la presenza di code viene trattata come spiegato in [LS93] e [LPR09].

4.2 Analisi di robustezza delle MBD

Nel calcolo delle profondità si deve scegliere il valore del parametro J, che regola la dimensione della tupla di elementi del dataset che forma gli inviluppi utilizzati nella definizione.
Consideriamo l'espressione della profondità modificata:

$$\mathrm{MBD}_N^{(j)}(f) = \binom{N}{j}^{-1} \sum_{1 \le i_1 < i_2 < \ldots < i_j \le N} \lambda_r \left(A(f; f_{i_1}, f_{i_2}, \ldots, f_{i_j}) \right),$$

$$\mathrm{MBD}_N^{J}(f) = \sum_{j=2}^{J} \mathrm{MBD}_N^{(j)}(f).$$

Da essa si può dedurre subito che l'entità delle profondità è influenzata direttamente dalla scelta di J, in particolare vale:

$$\forall f \in (f_1, \ldots, f_N) \qquad \mathrm{MBD}_N^{J_1}(f) \le \mathrm{MBD}_N^{J_2}(f) \qquad \forall J_1 \le J_2, \;\; J_1 \ge 2.$$

Il valore di J non influisce solo sulla grandezza delle profondità del campione, ma anche sullo sforzo computazionale richiesto per il loro calcolo, la cui complessità è crescente in maniera proporzionale a $\binom{N}{J}$ quando J cresce.
In [LPR09] viene sostenuto che l'ordinamento indotto dalle profondità sia stabile rispetto alla scelta del parametro J, autorizzando così a scegliere nelle applicazioni $J = 2$ e abbattere la complessità computazionale. Di seguito viene data prova della validità di questa affermazione attraverso due esempi.

Primo esempio

Nel primo esempio viene studiato l'andamento dei ranghi in un dataset artificiale composto da $N = 30$ funzioni univariate sinusoidali costruite artificialmente traslando verticalmente in maniera incrementale una funzione base, in maniera tale che il campione risulti costituito da funzioni i cui grafici siano impilati uno sull'altro (si veda la Figura 4.2).

Questo fa sì che le profondità siano determinabili a priori: le funzioni alla base e in cima al grafico saranno sempre escluse dagli inviluppi delle restanti, quindi avranno profondità nulla; le altre funzioni, invece, saranno incluse in un numero sempre maggiore di inviluppi mano a mano che si avvicineranno alla regione centrale del grafico, quindi avranno una profondità crescente, fino a raggiungere un massimo. Riportiamo in Figura 4.3 l'andamento delle profondità calcolate per questo dataset, come si può notare esse rispecchiano le previsioni teoriche.
Per quanto riguarda i ranghi indotti, essi sono deducibili visivamente dai grafici delle profondità, che essendo dilatate verso l'alto in maniera progressiva danno luogo a ranghi costanti al variare di J.

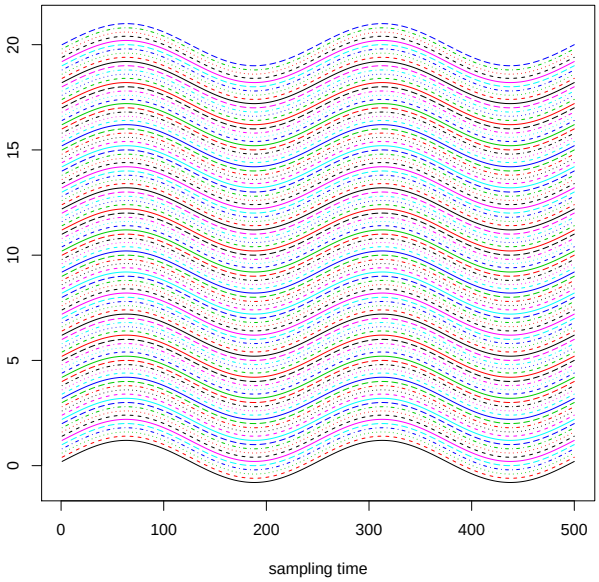

Figura 4.2: Grafico del primo dataset utilizzato per testare la robustezza dei ranghi indotti dalle misure di profondità. Si noti come i grafici degli elementi del campione siano tutti determinabili come traslazioni di un unico grafico base.

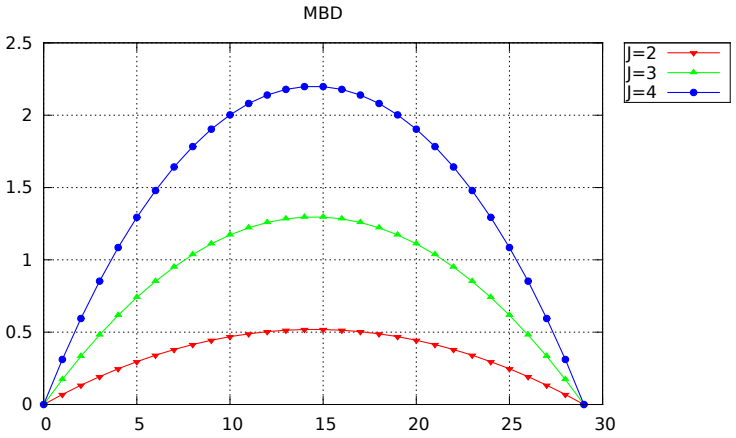

Figura 4.3: Misure di profondità determinate in corrispondenza del primo dataset di esempio, per valori di J crescenti. Si noti come i profili di profondità, nulli in corrispondenza dei segnali estremi, siano morfologicamente equivalenti, dando luogo ai ranghi costanti al variare di J.

Secondo esempio

Come secondo esempio, consideriamo un dataset artificiale formato da $N = 50$ segnali sinusoidali bivariati, con una variabilità congiunta localizzata sia in ampiezza che in fase, e affetti da

rumore gaussiano additivo. Assumiamo per essi un modello:

$$f_1(t) = (1 + X_1)\sin(2\pi t + Y_1) + N(t)$$
$$f_2(t) = (1 + X_2)\sin(2\pi t + Y_2) + N(t)$$

dove $\mathbf{X} = (X_1, X_2) \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}\right)$ e $\mathbf{Y} = (Y_1, Y_2) \sim \mathcal{N}\left(\mathbf{0}, \begin{pmatrix} 1 & \mu \\ \mu & 1 \end{pmatrix}\right)$, $t \in I = [0,1]$ e $N(t)$ tale che $\forall\, t \in [0,1]\ \ N(t) \sim \mathcal{N}(0, \sigma)$.

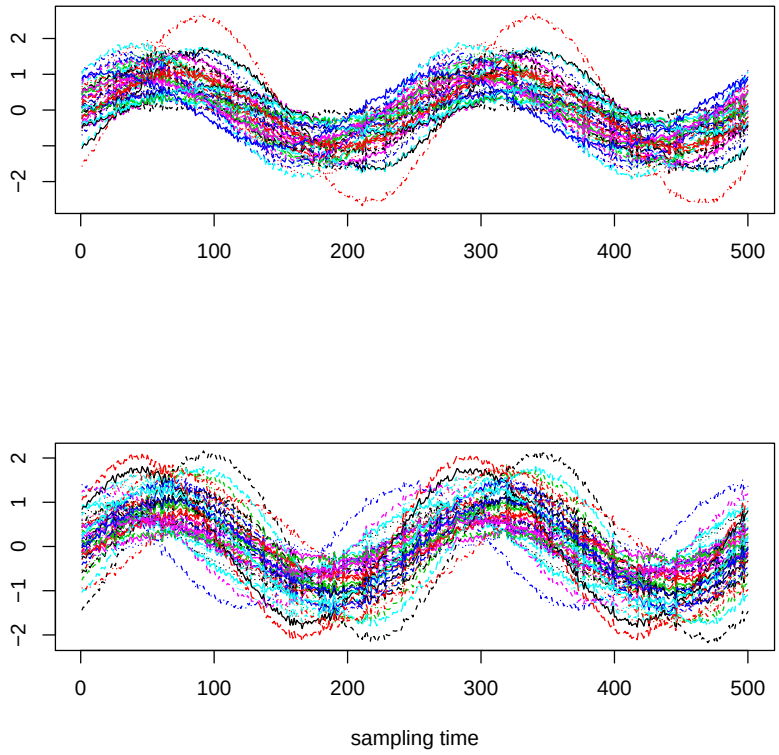

Figura 4.4: Grafico delle due dimensioni del dataset artificiale bivariato considerato nello studio della robustezza dell'ordinamento indotto dalle profondità.

Nell'esempio considereremo una popolazione di $N = 50$ segnali generata attraverso questo modello scegliendo $\rho = \mu = 0.4$ e $\sigma = 0.3$. Si può vedere in Figura 4.4 una rappresentazione grafica di questo dataset.
Si calcolano per questo dataset le misure di profondità modificate multivariate, determinate con pesi uniformi ($p_1 = p_2 = 1/2$), in corrispondenza dei valori di J crescenti $J = 2, 3, 4$.
Si riassumono in Figura 4.5 i risultati del calcolo. Come si può notare, i profili di profondità

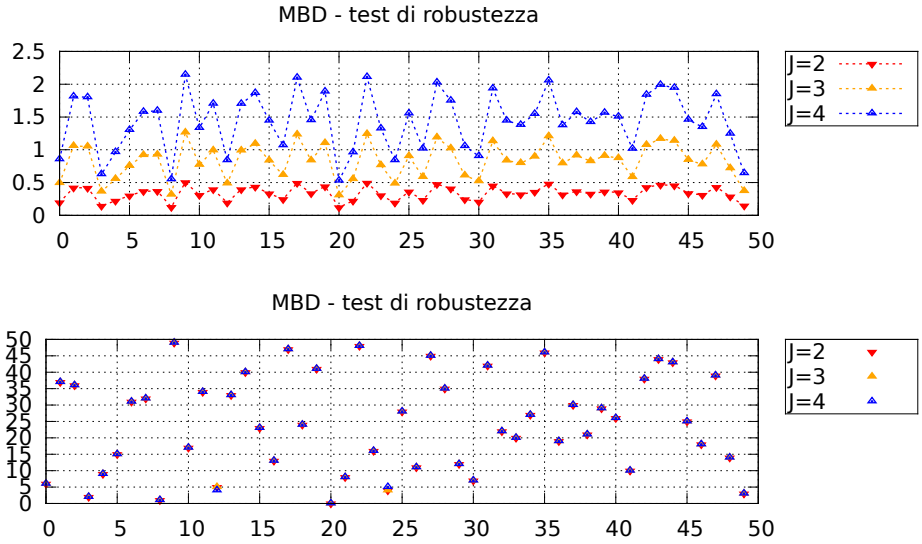

Figura 4.5: Risultato dell'analisi di robustezza delle MBD bivariate sul secondo dataset artificiale. In alto sono rappresentate le MBD funzionali bivariate, mentre in basso i ranghi ottenuti. Si noti la grande stabilità nei ranghi calcolati al crescere di J.

corrispondenti a valori di J differenti sono ben separati in una successione monotona crescente. Inoltre si può notare come tali insiemi siano anche morfologicamente corrispondenti, il che fa in modo che i ranghi calcolati a partire dall'ordinamento delle profondità siano molto stabili. Dal grafico si può osservare che soltanto due segnali hanno subito una modifica del rango corrispondente, al crescere di J.

Anche altri esperimenti confermano queste caratteristiche generali sulla sensitività delle profondità rispetto a J. Possiamo dunque concludere che i ranghi indotti dalle misure di profondità sul dataset siano stabili rispetto alla scelta del parametro J, autorizzando a porre per convenienza computazionale $J = 2$. Essi possono essere ritenuti come indicatori statistici robusti, quindi adatti all'impiego in analisi non-parametriche.

4.3 Applicazione agli ECG

I modelli e i metodi illustrati nella Sezione 4.1 trovano un significativo caso applicativo nell'analisi morfologica dei tracciati ECG.
In particolare, l'obiettivo che si pone è quello di studiare gli ECG reali e ideare una procedura di classificazione statistica per confrontare tra loro segnali differenti e separare tra di essi quelli corrispondenti a stati patologici da quelli relativi a pazienti sani.
Nel seguito si è studiato un dataset composto da ECG provenienti da pazienti sani ed affetti da blocco di branca sinistra (LBBB), una patologia cardiaca causata da un'anormalità nel sistema di conduzione cardiaco per cui, a causa di un blocco nel fascicolo di branca sinistro, il ventricolo sinistro si contrae più tardi rispetto a quello destro, causando un'asimmetria nel battito cardiaco.

Come menzionato nella Sottosezione 2.2.2, il LBBB ha effetti visibili sulla forma del tracciato ECG, ampliando l'intero complesso QRS e, nei casi *non-complicati*, producendo un'onda T con deflessione discordante rispetto a quella del QRS. L'impatto della malattia sulla forma dell'ECG rende tale patologia adatta ad un'analisi statistica basata sulla morfologia dei segnali.

4.3.1 Descrizione del data set

Il dataset utilizzato proviene dai dati raccolti nell'ambito del progetto PROMETEO (PROgetto sull'area Milanese Elettrocardiogrammi Teletrasferiti dall'Extra Ospedaliero), una collaborazione iniziata nel 2008 tra Azienda Regionale Emergenza Urgenza (AREU), Abbott Vascular and Mortara-Rangoni Europe s.r.l. [1] con l'intento di diffondere l'uso intensivo degli ECG come strumento diagnostico pre-ospedaliero e di costruire un nuovo database di ECG dalle caratteristiche mai registrate prima in altre banche dati sui disturbi cardiaci.
Sfruttando questa collaborazione, sono stati montati su tutte le Basic Rescue Units (BRU) dell'area urbana Milanese degli apparecchi per acquisire, registrare e trasferire via GSM gli ECG dei cittadini soccorsi dalle unità del 118, indipendentemente dai sintomi.

Ogni file di PROMETEO è in corrispondenza con altri tre files:

- *Details*, contenente informazioni utili all'analisi dei dati, come i tempi delle onde di depolarizzazione o ripolarizzazione, l'indicazione dei *landmark* e una diagnosi automatica effettuata con l'algoritmo Mortara-Rangoni VERITAS^{TM}.
- *Rhythm*, contenente la vera e propria registrazione dell'ECG, con una durata di 10s con campionamento ad 1kHz.
- *Median*, costruito a partire da *Rhythm*, riportante un battito di riferimento con la durata di 1.2 secondi (ad 1kHz).

Il dataset usato è costituito dai segnali contenuti nei files *Median*, di cui si sono considerate solo le 8 derivazioni I, II, V1, V2, V2, V3, V4, V5, V6, e raccoglie un totale di $N = 149$ pazienti, di cui 101 in stato fisiologico e 48 affetti da LBBB.
Al fine di prepararlo per le analisi statistiche di stampo funzionale, trasformando i singoli segnali da insiemi di misurazioni puntuali a funzioni regolari, è stato preprocessato rimuovendo il rumore e separando la variabilità di ampiezza da quella di fase. Questa è una variabilità, tipica dei dati funzionali di provenienza biomedica, riscontrabile nella fase dei dati e dovuta al fatto che gli stessi eventi biologici possono presentarsi in tempi assoluti a priori differenti in soggetti differenti. Di conseguenza è necessario ricondurre i segnali ad un unico tempo caratteristico rispetto al quale poter effettuare i confronti temporali della morfologia delle curve.

Entrambe le operazioni sono state svolte in [Iev12], in cui l'autrice ha sfruttato un algoritmo di smoothing funzionale multivariato basato su wavelet e una tecnica di registrazione non-lineare attraverso i landmark forniti naturalmente dalla suddivisione dell'ECG in segmenti.
Al termine di questa fase i 149 gruppi di 8 segnali ricostruiti sono definiti su 1024 punti, come conseguenza dell'algoritmo di smoothing via wavelet, che sfrutta griglie unidimensionali di 2^K punti, con $K \in \mathbb{N}$. In [Iev12] è stato scelto $K = 10$, considerando che riducendo il segnale in *Median* da 1200 a 1024 nodi di osservazione non si causasse una perdita di informazione, essendo

[1] Si ringrazia Mortara-Rangoni Europe s.r.l. per la gentile concessione dei dati

le caratteristiche importanti dell'ECG localizzate nel tratto centrale del tracciato.
Come ulteriore raffinamento, si sono eliminati i primi 149 e gli ultimi 174 punti di osservazione, riducendo i 1024 valori a 700 per ogni curva, poichè i tratti iniziale e finale riportavano degli andamenti ricostruiti non significativi dal punto di vista biologico. Anche questa riduzione non comporta una perdita di informazione caratterizzante, visto il minore potere discriminante di quei tratti nella separazione di soggetti sani da LBBB.

Ultimato il pre-processing, il dataset raccoglie gli ECG di 149 soggetti, di cui sono state considerate le lead I, II, V1, V2, V3, V4, V5, V6, con 700 misurazioni ad 1kHz che danno un tempo totale della dinamica osservata di 700ms (si veda la Figura 4.6).

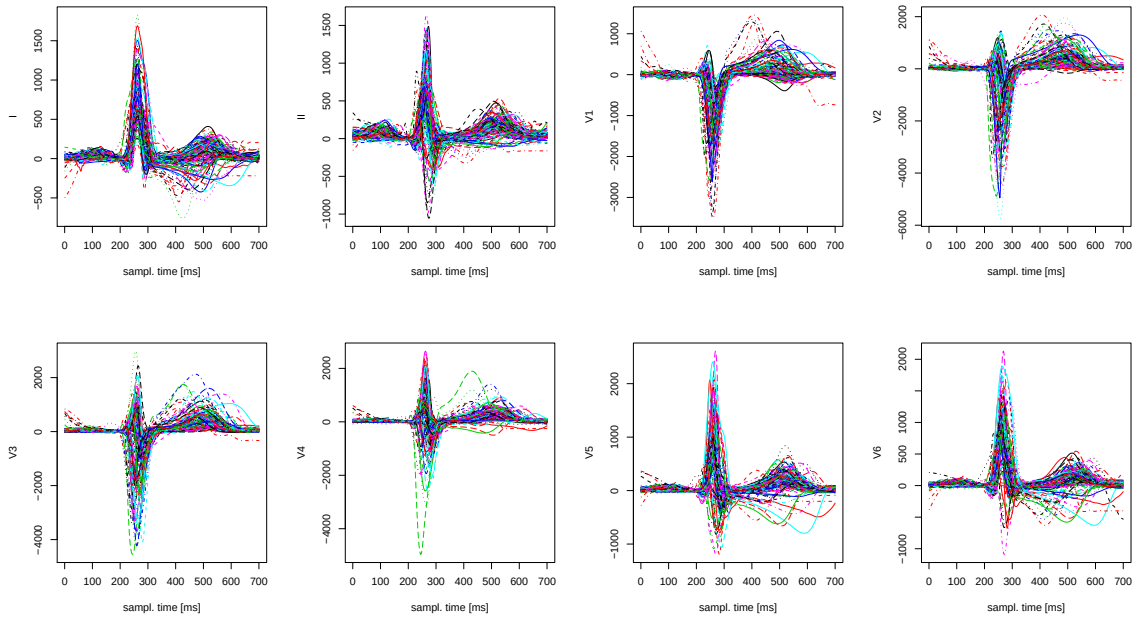

Figura 4.6: Segnali ECG provenienti dal database PROMETEO, dopo le procedure di smoothing e registrazione non-lineare.

4.3.2 Classificazione utilizzando le MBD

Le misure di profondità multivariate possono ora essere applicate al dataset misto di soggetti sani e in stato patologico (LBBB) al fine di costruire un classificatore per separare le due popolazioni in base a caratteristiche morfologiche dei segnali.

Da un esame grafico esplorativo risulta evidente che il comportamento dei segnali di soggetti sani e affetti da LBBB sia qualitativamente differente (si veda la Figura 4.7), e che un'analisi discriminante basata su caratteristiche morfologiche differenti delle due famiglie di segnali possa portare a buoni risultati.

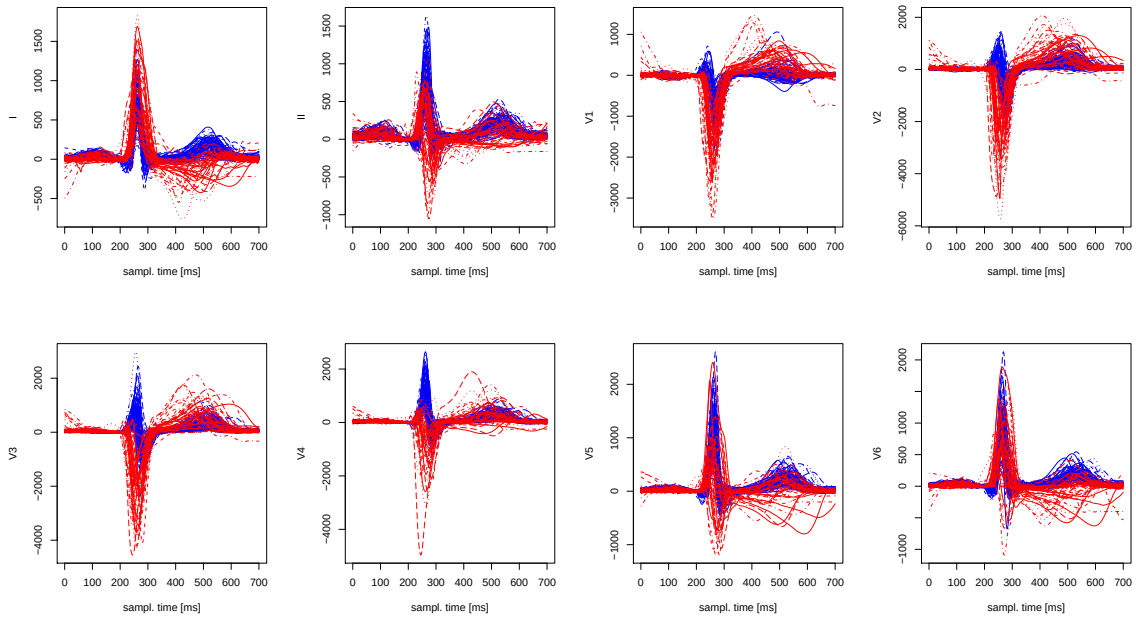

Figura 4.7: Segnali ECG provenienti dal database PROMETEO, dopo le procedure di smoothing e registrazione non-lineare, suddivisi in popolazione sana (colore blu) e LBB (colore rosso).

Il classificatore che viene proposto sfrutta le misure di profondità multivariate del campione di ECG come regressore di un modello logistico in cui si vuole prevedere la probabilità di appartenere alla popolazione sana.
Si supponga dunque di voler studiare $p \in [0,1]$, probabilità di appartenere alla popolazione sana. Si vuole utilizzare il modello lineare generalizzato:

$$p_i \sim \frac{e^{\beta_0+\beta_1 x_i}}{1+e^{\beta_0+\beta_1 x_i}} \tag{4.9}$$

ossia:

$$\text{logit}(p) \sim \beta_0 + \beta_1 x_i$$

dove $x_i = \text{MBD}(\mathbf{f}_i)$, profondità multivariate del campione. Si noti che la scelta di p come probabilità di appartenere alla popolazione sana sia conforme al fatto, riscontrato durante il calcolo delle profondità, che i segnali sani siano più centrali dei patologici.

Osservazione 4.3. Al fine di misurare la profondità relativa dei segnali patologici rispetto a quelli sani, e quindi pensare di aumentare la potenza del classificatore logistico, si sceglie di modificare la strategia di calcolo delle profondità in maniera simile a quanto fatto per la generalizzazione del test dei ranghi di Wilcoxon al caso di dati funzionali (Sottosezione 4.1.4). In particolare, anziché calcolare le profondità dei segnali del dataset testando ciascuno di essi con *tutti* i rimanenti segnali, si decide di estrarre dalla popolazione di sani un campione da considerare come riferimento, rispetto al quale calcolare le profondità sia dei restanti segnali

sani, sia di tutti i segnali patologici.
Questo adattamento è una conseguenza della scelta di utilizzare la profondità come indice non-parametrico della dissimilarità tra i soggetti sani e malati. Nel caso in cui i segnali vengano confrontati con tutti gli altri elementi del dataset, i pattern patologici fungono sia da oggetto di test, sia da riferimento rispetto al quale misurare la centralità dei segnali sani; questo può dar luogo ad un mascheramento dell'anormalità dei dati patologici rispetto a quelli sani rilevabile attraverso la profondità, e in ogni caso non rappresenta nella pratica l'intento di misurare quanto una popolazione sia morfologicamente differente rispetto all'altra. Invece, scegliendo un sottoinsieme della popolazione sana come riferimento e misurando le profondità rispetto a questo insieme di dati, si trova proprio un'espressione di quanto siano profondi i dati patologici rispetto a quelli sani.
Per questo motivo nel seguito di questa sezione utilizzeremo questa procedura per determinare le MBD.

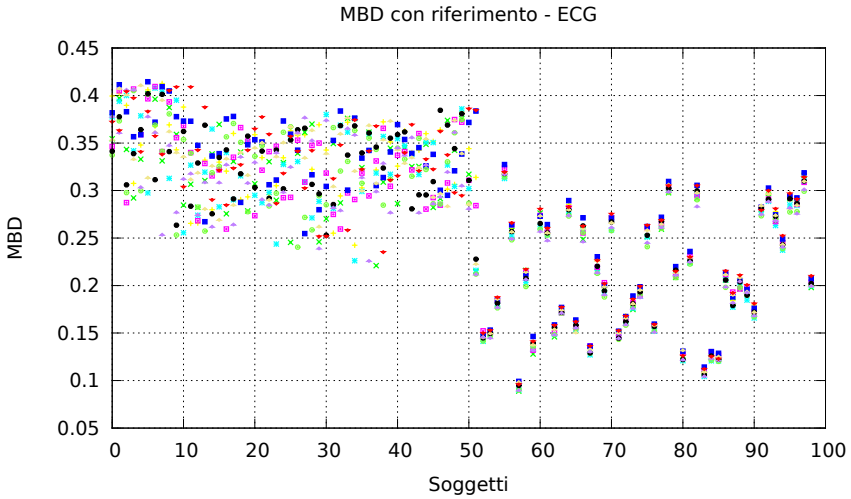

Figura 4.8: MBD determinate testando i segnali rispetto a 10 differenti campionamenti di $N_{\text{ref}} = 50$ soggetti dalla popolazione di sani.

Scegliamo di estrarre in maniera casuale $N_{ref} = 50$ soggetti dai 101 sani e considerarli come riferimento per il calcolo delle MBD; rimangono così $N = 99$ segnali da testare, di cui 51 sani e 48 patologici. Per il calcolo delle MBD multivariate adottiamo la scelta di pesi uniformi, cioè $p_k = 1/8 \ \ \forall k = 1, \ldots, 8$. In un successivo sviluppo delle analisi si potrebbe operare una scelta differente, che tenga conto ed evidenzi attraverso i pesi la struttura di covarianza tra le derivazioni dell'ECG. La procedura di campionamento dalla popolazione di soggetti sani viene ripetuta più volte, per evitare che le considerazioni successive dipendano dall'esito particolare del campionamento. Si riportano in Figura 4.8 le MBD determinate a partire da 10 differenti campionamenti, esse mostrano un pattern molto stabile che evidenzia come le profondità dei dati sani siano quasi dominanti rispetto a quelle dei patologici.
Le MBD così determinate vengono introdotte nel modello di regressione logistica (4.9), i cui valori predetti vengono confrontati con la soglia di cut-off $\bar{p} = 0.5$, attribuendo le osservazioni con $p_i > \bar{p}$ alla popolazione di sani, e le restanti alla popolazione di soggetti patologici. Il risulta-

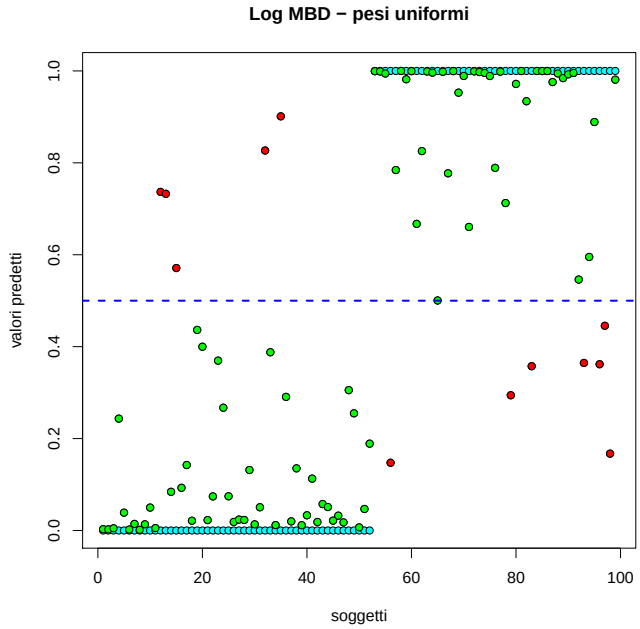

Figura 4.9: Esito del classificatore di regressione logistica in corrispondenza del primo campionamento. In verde sono riportati i soggetti classificati correttamente, mentre in rosso i soggetti misclassificati. In ciano è mostrata la suddivisione del dataset nelle due sottopopolazioni. La soglia di cut-off è posta a $\bar{p} = 0.5$.

to di questa procedura in uno dei 10 campionamenti, insieme al confronto tra la popolazione attribuita e quella effettiva di provenienza, viene riportato in Figura 4.9.
La significatività statistica del modello proposto è attestata in ciascuna delle prove svolte, in cui si ottengono *p-values* molto bassi relativi sia al coefficiente β_0 che β_1 del modello (4.9). Come esempio, in corrispondenza del primo campionamento si ottiene un p-value per β_0 pari a $1.20e-05$ e uno per β_1 di $6.66e-06$.

Ripetendo le analisi per tutti i campionamenti estratti dalla popolazione di soggetti sani si valuta la misclassificazione risultante in corrispondenza di ognuno di essi, ottenendo per ciascuno dei valori molto buoni (si riporta nella Tabella 4.1, a titolo di esempio rappresentativo, la matrice di confusione per il primo campionamento). Per questo motivo il classificatore proposto sembra adatto a separare le due popolazioni del campione.

		Pop. vera		
		Sani	**LBBB**	**Totale**
Pop. attr.	**Sani**	46	7	53
	LBBB	5	41	46
	Totale	51	48	

Tabella 4.1: Matrice di confusione per la classificazione dei 99 soggetti sani e patologici in corrispondenza del primo campionamento.

Capitolo 5

Validazione statistica di ECG sintetici

In questo capitolo vengono introdotti e combinati gli strumenti necessari per procedere alla validazione degli ECG sintetici sfruttando le tecniche di analisi statistica non-parametrica basate sulle profondità dei dati funzionali multivariati illustrate nel Capitolo 4.
In particolare, l'obiettivo è quello di verificare se gli ECG determinati a partire da simulazioni numeriche dell'attività elettrica del cuore, condotta con i modelli e le tecniche descritte nel Capitolo 3, siano una riproduzione accurata di quelli riscontrabili in popolazioni reali, tenendo contemporaneamente conto della loro variabilità naturale. Questo viene fatto applicando le tecniche di analisi statistica non parametrica su un dataset composto sia da dati reali che da dati simulati. Per semplicità ci si limita a considerare il caso di tracciati corrispondenti a soggetti sani.

Per raggiungere questo scopo bisognerà da una parte considerare un solutore per il problema elementi finiti, antecedente alla determinazione di ECG sintetici, sia considerare un opportuno strumento con cui condurre le analisi di profondità .
Nel seguito verranno illustrati entrambi.

5.1 Libreria FELiScE

Il solutore utilizzato per simulare l'attività elettrica del cuore rientra nel progetto FELiScE [2], una libreria di metodi elementi finiti dedicata alla soluzione di problemi ingegneristici e appartenenti all'ambito delle scienze della vita. Essa è sviluppata dai team Macs e Reo del centro di ricerca di INRIA - Paris, Rocquencourt.
Al suo interno si trovano moduli per l'impostazione e la soluzione del modello bidominio descritto nel Capitolo 3, considerando sia il problema accoppiato (3.15) sia quello isolato (3.1.3), in presenza di diversi modelli ionici per la dinamica elettrica membranale.
La libreria è parallelizzata sfruttando il protocollo di comunicazione MPI per suddividere il costo computazionale delle simulazioni in più processori e permettere l'applicazione di tecniche di domain decomposition (attraverso l'algoritmo di Dirichlet-Neumann) utili a velocizzare la soluzione del problema accoppiato.

[2]Si ringrazia il team Reo di INRIA-Paris, Rocquencourt per avere gentilmente messo a disposizione il codice.

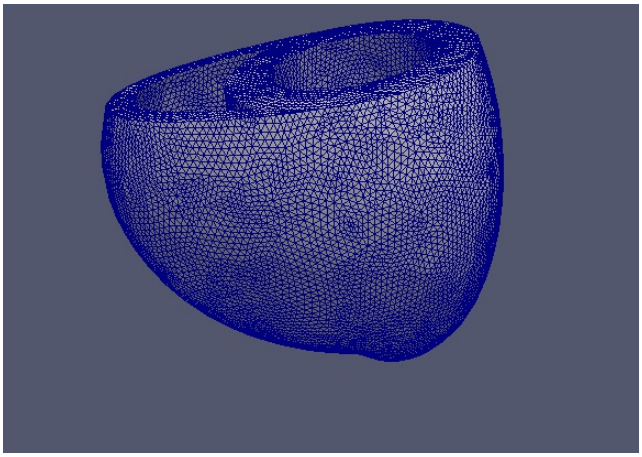

Figura 5.1: Rappresentazione della geometria cardiaca semplificata, costituita dai soli ventricoli.

Una volta determinato il potenziale elettrico definito sul cuore e nel torso che risolve il modello bidominio, è possibile sfruttare FELiScE per costruire l'ECG corrispondente applicando al potenziale trovato la definizione standard delle 12 derivazioni, come descritto nella Sottosezione 2.2.1.

Osservazione 5.1. Ad oggi il solutore permette di risolvere il modello bidominio accoppiato solo su una geometria cardiaca semplificata, costituita dai soli ventricoli, come mostrato in Figura 5.1. L'assenza degli atrii fa sì che gli elettrocardiogrammi costruiti a partire dalle simulazioni siano mancanti della parte iniziale del segnale (si veda la Figura 2.7), limitandosi a riprodurre il complesso QRS e l'onda T. Questo sarà importante quando essi verranno testati rispetto ai segnali reali.

5.2 Libreria HPCS

Lo strumento utilizzato per condurre le analisi statistiche in questa sezione (e nel resto del lavoro di Tesi) è stato il codice appositamente sviluppato e raccolto nella libreria HPCS.
HPCS è una libreria *open-source* scritta in `C++` pensata per applicazioni di statistica computazionale, specialmente se ad alte prestazioni, che non possono essere raggiunte dal software di alto livello per analisi statistiche attualmente utilizzato in ambito accademico e professionale, R.

La libreria contiene un modulo che permette di calcolare la profondità di dati funzionali multivariati in maniera efficiente, beneficiando da una parte della robustezza del linguaggio compilato e dall'altra dell'efficienza messa a disposizione dalla possibilità di eseguire il codice in parallelo, sfruttando il protocollo di parallelizzazione `MPI`.
Nella creazione di `HPCS` sono stati presi in considerazione i seguenti obiettivi:

- La libreria deve essere in grado di rispondere ad esigenze concrete di rapidità di esecuzione e scalabilità rispetto alle dimensioni dei compiti assegnati, quindi la parallelizzazione del codice sottostante deve essere nativa e disegnata in maniera tale da integrarsi profondamente con le strutture necessarie al calcolo.

- In fase di design del codice si deve decidere per l'implementazione di oggetti di utilizzabilità trasversale, che realizzino una suddivisione modulare del compito da eseguire incapsulandone le funzionalità singole.
- Il codice deve essere organizzato in maniera tale da fornire all'utilizzatore la maggior flessibilità disponibile, favorendo la controllabilità esterna al sorgente, e possibilmente a run-time, degli oggetti utilizzati per i vari compiti.
- Il codice deve poter essere distribuito e fruito anche da altri utilizzatori, inoltre deve poter essere facilmente esteso inglobando eventuali contributi provenienti dai collaboratori.

L'obiettivo di flessibilità viene raggiunto permettendo il setup degli oggetti delle varie classi attraverso il parser `GetPot` e combinandolo, quando necessario, con l'utilizzo di opportune *abstract factories* che permettano di raccordare l'esigenza del sorgente di conoscere a compile-time il tipo di tutti gli oggetti creati con quella dell'utilizzatore di specificarne alcuni a run-time.
`GetPot` permette l'input nel codice di informazioni disponibili in un file `data` esterno al sorgente, organizzato con una struttura a dizionario in cui ad ogni keyword corrisponde il valore da leggere durante l'esecuzione. I file `data` possono essere modificati arbitrariamente senza richiedere la ricompilazione dei codici sorgenti, necessaria invece se non si fosse utilizzato il parser per rompere la dipendenza del setup degli oggetti dalla conoscenza a compile-time dei parametri. Questo ha l'importante vantaggio di ridurre l'esigenza di ricompilazione del codice solo ai casi in cui sia effettivamente necessario, rendendolo contemporaneamente più semplice e veloce da maneggiare, il che è particolarmente importante per una libreria scritta in un linguaggio come il `C++`: molto robusto ma che richiede tempi di sviluppo maggiori rispetto ai linguaggi di alto livello.
L'obiettivo semplificare la distribuzione di `HPCS` è stato attuato creando un repository online su Github da cui `HPCS` può essere scaricato e installato (`https://github.com/larvatus/HPCS`).
Il codice può anche essere clonato facendo uso di git, un sistema avanzato di versioning molto diffuso, utile per tenere traccia della storia di un progetto e gestire in maniera apposita sia contributi dall'esterno che fusioni di rami di sviluppo differenti del singolo programmatore.
La fruibilità di `HPCS` è stata estesa utilizzando `CMake` , un software multipiattaforma che permette di configurare e generare `Makefile` nativi al sistema in uso, sfruttando compilatori e percorsi delle librerie locali e rendendo così molto più facile la diffusione del codice tra gli sviluppatori.
Oltre a questo la libreria è accompagnata da un'apposita documentazione creata con Doxygen, un utile strumento per generare facilmente la documentazione associata ad una libreria. Una volta introdotti nel sorgente dei commenti preceduti da opportune keyword, `Doxygen` può essere lanciato per leggere il codice e generare l'elenco delle classi implementate, i loro diagrammi di collaborazione e la descrizione di metodi ed attributi.

Dal punto di vista macroscopico il codice è organizzato in una struttura modulare e gerarchica. In una prima directory `source` vengono raccolti i sorgenti `.hpp` e `.cpp` in cui sono state ripartite ed implementate le singole funzionalità. Nella cartella `synthetic` sono riportati i dataset creati artificialmente che possono essere usati nei test e nello studio delle misure di profondità. I sorgenti dei test, invece, sono collocati nella directory `test`.

5.2.1 Strumenti di sviluppo

CMake

Il funzionamento di `CMake` è basato sull'utilizzo di opportuni files `CMakeLists.txt` contenuti nelle directory della libreria. Nel file CMakeLists.txt della directory principale viene impostata la dipendenza dalla versione di CMake richiesta, il nome del progetto corrente e si estende il path di compilazione includendo le directory dei sorgenti. Oltre a questo vengono cercate le librerie esterne utilizzate dai sorgenti, che sono:

- `boost`, di cui vengono usati in maniera capillare gli shared pointers, robusti rispetto ai problemi di scoping dei puntatori raw, e le `ublas`, che sono il fondamento della rappresentazione matriciale dei dataset;
- `mpi`, che viene utilizzato per la parallelizzazione del codice;
- `Doxygen`, che viene utilizzato per generare la documentazione.

Per usare `CMake` creare dei `Makefile` nativi con cui compilare il codice sorgente è necessario creare una directory di `build`, in cui il codice verrà compilato, e lanciare `CMake` nel seguente modo (si presuppone l'uso di un sistema Linux):

```
cd <build_dir>
cmake <source_dir>
```

`CMake` esaminerà le dipendenze dalle librerie esterne e imposterà variabili di path e obiettivi specifici secondo quando richiesto nel file `CMakeLists.txt`. `CMake` procederà anche ad esaminare le sottocartelle `source` e `test` e, grazie alla presenza in esse di altri file `CMakeLists.txt` che definiscono le ulteriori sottocartelle e obiettivi, esaminerà in maniera ricorsiva tutto l'albero della directory.

```
CMAKE_MINIMUM_REQUIRED(VERSION 2.8)

PROJECT(BandDepth)

INCLUDE_DIRECTORIES(${CMAKE_CURRENT_SOURCE_DIR})


#Require Boost for this project
FIND_PACKAGE(Boost)
INCLUDE_DIRECTORIES(${Boost_INCLUDE_DIR})

# Require MPI for this project:
FIND_PACKAGE(MPI REQUIRED)
SET( CMAKE_CXX_COMPILE_FLAGS ${CMAKE_CXX_COMPILE_FLAGS} ${MPI_COMPILE_FLAGS})
SET( CMAKE_CXX_LINK_FLAGS ${CMAKE_CXX_LINK_FLAGS} ${MPI_LINK_FLAGS})
SET( CMAKE_BUILD_TYPE ${CMAKE_CXX_FLAGS_DEBUG})
INCLUDE_DIRECTORIES(${MPI_INCLUDE_PATH})

#Require Doxygen for documentation
FIND_PACKAGE(Doxygen)
IF (DOXYGEN_FOUND)
SET(DOXYGEN_INPUT ${CMAKE_CURRENT_SOURCE_DIR}/Doxyfile.in )
SET(DOXYGEN_OUTPUT ${CMAKE_BINARY_DIR}/doc)
ADD_CUSTOM_COMMAND(OUTPUT ${DOXYGEN_OUTPUT}
    COMMAND ${CMAKE_COMMAND} -E echo ${CMAKE_BINARY_DIR}
    COMMAND ${CMAKE_COMMAND} -E echo_append "Building␣API␣Documentation..."
    COMMAND ${DOXYGEN_EXECUTABLE} ${DOXYGEN_INPUT}
```

```
    COMMAND ${CMAKE_COMMAND} -E echo "Done."
    WORKING_DIRECTORY ${CMAKE_CURRENT_SOURCE_DIR}
    DEPENDS ${DOXYGEN_INPUT}
   )

ADD_CUSTOM_TARGET(doc DEPENDS ${DOXYGEN_OUTPUT})
ENDIF (DOXYGEN_FOUND)

# Adding test subdirectory
ADD_SUBDIRECTORY(source)
ADD_SUBDIRECTORY(test)
```

Nelle cartelle di test i file `CMakeLists.txt` definiscono gli obiettivi relativi ai sorgenti dei test e creano i collegamenti simbolici ai files di appoggio contenuti in `test` che vengono usati durante l'esecuzione. Questi sono principalmente files `data` usati da `GetPot` e script `Gnuplot` per la visualizzazione dei risultati.

```
EXECUTE_PROCESS( COMMAND ln -s ${CMAKE_CURRENT_SOURCE_DIR}/data ${
    CMAKE_CURRENT_BINARY_DIR}/data  )
EXECUTE_PROCESS( COMMAND ln -s ${CMAKE_CURRENT_SOURCE_DIR}/multiDepthMeasure.plot ${
    CMAKE_CURRENT_BINARY_DIR}/multiDepthMeasure.plot )
EXECUTE_PROCESS( COMMAND ln -s ${CMAKE_CURRENT_SOURCE_DIR}/depthMeasure.plot ${
    CMAKE_CURRENT_BINARY_DIR}/depthMeasure.plot )

# A first executable

ADD_EXECUTABLE(main_depthMeasureAll.exe main_depthMeasureAll )
TARGET_LINK_LIBRARIES(main_depthMeasureAll.exe ${MPI_LIBRARIES} mbd)


ADD_EXECUTABLE(main_depthMeasureRef.exe main_depthMeasureRef )
TARGET_LINK_LIBRARIES(main_depthMeasureRef.exe ${MPI_LIBRARIES} mbd)


ADD_EXECUTABLE(main_multiDepthMeasureAll.exe main_multiDepthMeasureAll )
TARGET_LINK_LIBRARIES(main_multiDepthMeasureAll.exe ${MPI_LIBRARIES} mbd)
EXECUTE_PROCESS( COMMAND ln -s ${CMAKE_CURRENT_SOURCE_DIR}/weightsAll.dat ${
    CMAKE_CURRENT_BINARY_DIR}/weightsAll.w  )



ADD_EXECUTABLE(main_multiDepthMeasureRef.exe main_multiDepthMeasureRef )
TARGET_LINK_LIBRARIES(main_multiDepthMeasureRef.exe ${MPI_LIBRARIES} mbd)
EXECUTE_PROCESS( COMMAND ln -s ${CMAKE_CURRENT_SOURCE_DIR}/weightsRef.dat ${
    CMAKE_CURRENT_BINARY_DIR}/weightsRef.w  )
```

Nella directory `source` viene aggiunta la creazione di una libreria, `mbd`, a partire dai sorgenti contenuti.

Doxygen

Per potere utilizzare `Doxygen` e creare la documentazione di `HPCS` è necessario il file `Doxyfile.in`, contenuto nella directory principale, che definisce alcune variabili di configurazione importanti quali le cartelle in cui cercare i sorgenti, l'estensione dei files da considerare e la directory in cui creare la documentazione.

```
PROJECT_NAME           = HPCS
OUTPUT_DIRECTORY       = ./build/doc
INPUT                  = ./source ./test/
FILE_PATTERNS          = *.cpp *.hpp
SEARCH_ENGINE          = YES
HAVE_DOT               = YES
COLLABORATION_GRAPH    = YES
HIDE_UNDOC_RELATIONS   = NO
```

Per creare la documentazione, dopo aver lanciato `CMake`, è sufficiente eseguire:

```
cd <build_dir>
make doc
```

Al termine dell'esecuzione verrà creata una directory `build/doc` che conterrà la documentazione LaTeXe `html`.

5.2.2 Strategia implementativa

Di seguito viene descritta la strategia implementativa adottata nell'implementazione del calcolo delle misure di profondità.

Si consideri la definizione di misura di profondità, ad esempio univariata modificata, come riportata nella Definizione 4.2:

$$\mathbf{U} \quad \begin{cases} \mathrm{MBD}_N^{(j)}(f) = \binom{N}{j}^{-1} \sum_{1 \le i_1 < i_2 < \ldots < i_j \le N} \lambda_r \left(A(f; f_{i_1}, f_{i_2}, \ldots, f_{i_j}) \right), \\ \mathrm{MBD}_N^{J}(f) = \sum_{j=2}^{J} \mathrm{MBD}_N^{(j)}(f). \end{cases}$$

Il calcolo di $\mathrm{MBD}_N^J(f)$ è stato suddiviso macroscopicamente in tre passi:

(U1) Acquisizione del dataset.

(U2) Calcolo delle bande di profondità $\mathrm{MBD}_N^{(j)}(f)$, $j = 1, 2, \ldots, J$.

(U3) Sintesi e calcolo delle misure di profondità univariate $\mathrm{MBD}_N^J(f)$.

Si considerino ora le misure di profondità multivariate modificate, esplicitando l'espressione introdotta nella Definizione 4.4:

$$\mathbf{M} \quad \begin{cases} \mathrm{MBD}_{N,k}^{(j)}(f_k) = \binom{N}{j}^{-1} \sum_{1 \le i_1 < i_2 < \ldots < i_j \le N} \lambda_r \left\{ A(f_k; f_{i_1,k}, \ldots, f_{i_j,k}) \right\}, \\ \mathrm{MBD}_{N,k}^{J}(f_k) = \sum_{j=2}^{J} \mathrm{MBD}_{N,k}^{(j)}(f_k), \\ \mathrm{MBD}_N^J(\mathbf{f}) = \sum_{k=1}^{s} p_k \, \mathrm{MBD}_{N,k}^J(f_k) \end{cases}$$

Ai tre passi precedenti, nei quali si calcolano le profondità delle singole componenti, si aggiunge un quarto passo per il calcolo della misura di profondità globale del dato come media pesata con i pesi $\{p_k\}$ delle profondità di tutte le componenti.

(M1) Acquisizione del dataset.

(M2) Calcolo delle bande di profondità $\mathrm{MBD}^{(j)}_{N,k}(f_k)$, $k = 1, \ldots, s$ e $j = 1, \ldots, J$.

(M3) Sintesi e calcolo delle misure di profondità $\mathrm{MBD}^{J}_{N,k}(f_k)$, $k = 1, \ldots, s$ di ogni componente del segnale multivariato.

(M4) Sintesi e calcolo delle misure di profondità multivariate di ogni segnale $\mathrm{MBD}^{J}_{N}(\mathbf{f})$.

Osservazione 5.2. Oltre al calcolo normale delle misure di profondità, svolto considerando ogni funzione del dataset e misurandone la profondità rispetto a tutte le altre, si è voluto fornire anche l'implementazione del metodo basato sulla suddivisione del dataset in due parti: una di riferimento (*reference-set*), che raccoglie i segnali rispetto ai quali calcolare le profondità, e un'altra di test (*test-set*) che contiene i dati di cui si vuole calcolare la profondità rispetto al reference set. Come già menzionato nell'Osservazione 4.3, questa procedura è molto utile per avere una misura non-parametrica efficace della dissimilarità tra segnali appartenenti a differenti sottopopolazioni di uno stesso campione. In particolare i dati della prima sottpopolazione possono costituire il reference-set, che rappresenta un metro di misura con cui confrontare i segnali della seconda popolazione (che formerà il test-set). Nel seguito la procedura standard, in cui i dati sono confrontati in maniera combinatoria con tutti gli altri, viene indicata con la keyword `All`, mentre la procedura che separa il test-set dal reference-set è indicata con la keyword `Reference`.

A partire dalla suddivisione in passi del calcolo delle profondità e dati gli obiettivi ricorrenti, si è suddivisa l'implementazione in maniera da incapsulare i diversi passi in alcuni oggetti:

- **(U1),(M1)** $\rightsquigarrow$ `DataSet`, `DataSetLevelled`
- **(U2),(M2)** $\rightsquigarrow$ `BandDepth`, `BandDepthRef`.
- **(U3),(M3)** $\rightsquigarrow$ `DepthMeasure`.
- **(M4)** $\rightsquigarrow$ `MultiDepthMeasure`.

Nel seguito verranno forniti alcuni dettagli sull'implementazione di queste classi.

5.2.3 `DataSet`

La classe `DataSet` è la struttura base che rappresenta il dataset funzionale. La sua interfaccia pubblica deve permettere una costruzione flessibile, che si adatti alle varie forme nelle quali i dati possano essere disponibili nel codice, e contemporaneamente la sua implementazione deve garantire una grande efficienza, sia rispetto alla memorizzazione dei dati, sia rispetto all'accesso dall'esterno, che presumibilmente avverrà molto frequentemente.
In vista del calcolo delle bande di profondità rispetto ad una sottopopolazione di riferimento, oltre al concetto di di dataset funzionale normale si è implementato il concetto di dataset sudddiviso in livelli, ossia una partizione in sottoinsiemi corrispondenti alle sottopopolazioni. La classe corrispondente è `DataSetLevelled`.

DataSet

La classe DataSet memorizza i dati sfruttando le funzionalità messe a disposizione dal modulo ublas delle librerie boost. Nelle ublas si trovano implementati molti oggetti tipici delle librerie di algebra lineare, tra cui matrici, vettori e proxies. La loro robustezza, l'efficienza di gestione e la presenza di molte funzioni specializzate per l'accesso a righe o colonne o loro insiemi ne fanno un ambiente ideale da cui trarre la struttura dati alla base della classe DataSet, che è una matrix< Real >.
L'interfaccia pubblica di DataSet permette l'utilizzo di diversi costruttori, che richiedono di specificare il numero di campioni nel dataset, nbSamples, il numero di punti dei segnali, nbPts, e i dati veri e propri. Essi verranno salvati in uno shared_pointer ad un oggetto matrix con nbSamples righe e nbPts colonne.

```
class DataSet
{
public:
  typedef boost::numeric::ublas::matrix< Real > data_Type;
  typedef boost::shared_ptr< data_Type > dataPtr_Type;

  DataSet( const UInt & nbSamples, const UInt & nbPts );
  DataSet( const Real * data, const UInt & nbSamples, const UInt & nPts );
  DataSet( const std::vector< Real > & data, const UInt & nbSamples, const UInt & nbPts )
      ;
  DataSet( const data_Type & data );
  DataSet( const dataPtr_Type & dataPtr );

protected:
  dataPtr_Type M_data;
};
```

L'accesso agli elementi del dataset è possibile grazie all'overload dell'operatore ():

```
 Real operator()( const UInt & sample, const UInt & pt ) const;
```

Le altre funzionalità della classe sono fornite da setters & getters, un metodo ShowMe per l'output dello stato interno dell'oggetto e uno per scrivere i dati su file. Particolarmente utile è il setter:

```
void readData( const std::string & filename );
```

DataSetLevelled

La classe DataSetLevelled implementa il concetto astratto, ma comune in ambito statistico, di data set con livelli, cioè gruppi in cui i campioni sono raggruppati in base a qualche criterio. Questa struttura è alla base del calcolo delle misure di profondità di un dataset rispetto ad un campione di riferimento. In particolare, considerando due livelli, un primo definirà la parte di campione da cui verrà costruita la sotto-popolazione di riferimento (reference set), mentre l'altro definirà la parte di campione di cui calcolare le profondità rispetto al riferimento (test set).

La classe, derivata da `DataSet` è pensata per gestire un numero qualunque di livelli, che vengono determinati specificando gli ID dei dati relativi. Visto che a priori il numero di livelli è sconosciuto, che i singoli ID dei dati corrispondenti ai vari livelli possono avere una disposizione irregolare nel dataset, e soprattutto vista la possibile necessità di fare ricerche negli insiemi di ID di un dato gruppo, si è scelto di usare le `std::map` per rappresentare gli insiemi ordinati di ID dei gruppi.

```
class DataSetLevelled : public DataSet
{
public:
  typedef std::set< UInt > IDContainer_Type;
  typedef std::vector< IDContainer_Type > levelsContainer_Type;
  typedef boost::shared_ptr< levelsContainer_Type > levelsContainerPtr_Type;

  DataSetLevelled( const UInt & nbSamples, const UInt & nbPts, const UInt & nbLevels );

private:
  levelsContainerPtr_Type M_levelsPtr;
};
```

I costruttori di DataSetLevelled riprendono quelli di `DataSet`, aggiungendo la variabile `nbLevels`. Per brevità ne è stato riportato solo uno.

Il setup dei livelli viene effettuato sfruttando i setters disponibili in due modi: è possibile passare alla classe direttamente un oggetto `levelsContainerPtr_Type`, oppure è possibile specificare sottoinsiemi contigui del dataset che costituiscono i vari gruppi.

```
void setLevels( const levelsContainerPtr_Type & levelsPtr );
void setLevels( const std::vector< UInt > & linearExtrema );
void setLevelsFromExtrema( const std::string & levelsExtremaFilename );
```

Gli estremi dei livelli (`linearExtrema`) sono gli ID che definiscono il dato livello. Così il vettore di estremi $(0, 10, 20)$ identificherà i due livelli $L_0 = (0, ..., 9)$ e $L_1 = (10, ..., 20)$.
L'impostazione di tali estremi è possibile sia passando direttamente il vettore degli ID estremi a `setLevels`, sia specificando a `setLevelsFromExtrema` un file esterno che li contiene.
Completano la classe i setters & getters delle nuove variabili e l'override del metodo `showMe` della classe base.

5.2.4 `Band Depth`

Le classi deputate al calcolo dei contributi $\mathrm{MBD}_N^{(j)}(f)$ sono `BandDepth<_J>` e `BandDepthRef<_J>`, template rispetto al parametro intero `_J` che definisce la dimensione della tupla considerata nella costruzione del sottoinsieme $A(f; f_{i_1}, f_{i_2}, \ldots, f_{i_j})$. `BandDepth<_J>` corrisponde al calcolo dei contributi nel caso normale, mentre `BandDepthRef<_J>` corrisponde al calcolo rispetto ad un sottoinsieme di riferimento.

Entrambe le classi derivano da una classe base, `BandDepthBase<BDPolicy _policy >`, template rispetto ad una policy `BDPolicy`, definita come tipo enumerativo, che esprime se il calcolo debba essere effettuato in maniera standard o rispetto al sottoinsieme di riferimento.

`BandDepth<_J>` deriva dalla specializzazione `BandDepthBase<All>`, mentre `BandDepthRef<_J>` deriva da `BandDepthBase<Reference>`.

```
typedef unsigned int UInt;

enum BDPolicy {Reference, All };

template < BDPolicy _policy >
class BandDepthBase
{...};

template < UInt _J >
class BandDepth : public BandDepthBase< All >
{...};

template < UInt _J >
class BandDepthRef : public BandDepthBase< Reference >
{...};
```

Nota 5.3. La scelta, presa in fase di design del codice, di definire entrambe le classi `BandDepth<_J>` e `BandDepthRef<_J>` template rispetto al parametro `_J` ha due motivi: da una parte si vuole imporre un rigido controllo direttamente a compile-time sul valore di questo parametro, cruciale per l'implementazione. Dall'altro questa scelta permette di specializzare i metodi responsabili del calcolo effettivo delle profondità, che almeno nel caso di `_J=2` possono sfruttare un algoritmo più semplice e molto più veloce rispetto a quello standard.

Nota 5.4. La classe base `BandDepthBase<_policy>` viene completamente specializzata per i valori di `_policy=All` e `_policy=Reference` per fornire ai tipi derivati un'interfaccia pubblica utilizzabile in maniera polimorfica che tenga conto della diversa gesione esterna tra il caso `All` e `Reference`.
Essendo classi base con interfaccia pubblica virtuale, costituita principalmente da setters & getters e dal metodo `computeBDs()`, il codice duplicato è minimo.
La presenza di queste classi base, indipendenti dal parametro `_J`, ha anche il vantaggio di permettere l'istanziazione di oggetti `BandDepth<_J>` e `BandDepthRef<_J>` attraverso un'apposita factory che, insieme all'uso di `GetPot`, incrementa la flessibilità e la gestibilità a run-time del codice.

Inizializzazione: `BandDepthData` e `BandDepthRefData`

La gestione delle informazioni necessarie al setup degli oggetti `BandDepth<_J>` e `BandDepthRef<_J>` è esterna e viene demandata alle classi `BandDepthData` e `BandDepthRefData`.
La classe `BandDepthData` contiene variabili che esprimono:

- il numero di segnali nel dataset e numero di punti per segnale;
- l'eventuale file di input da cui leggere il dataset;
- l'offset destro e sinistro nella lettura del dataset;
- il livello di verbosità;
- il valore di `_J`;

- il file di output nel quale verranno stampate le profondità determinate.

La classe `BandDepthRefData`, derivata da `BandDepthData`, è pensata per contenere i dati degli oggetti che calcolano le bande di profondità rispetto ad una sottopopolazione di riferimento contenuta nel dataset. Come spiegato nella sezione riguardante la classe `DataSetLevelled`, la creazione di un reference-set è possibile dividendo il dataset in livelli ed estraendo i segnali di riferimento da un particolare livello (che di default è il livello con ID $= 0$).

La gestione dei livelli richiede ad un oggetto `BandDepthRefData` di disporre di altre informazioni oltre a quelle della classe base:

- il numero degli elementi del reference set, che deve essere minore della cardinalità del livello corrispondente;
- il numero di elementi del test-set, che viene determinato sottraendo ai segnali totali il numero di elementi del reference-set;
- un'eventuale stringa contenente il nome del file che riporta gli estremi dei livelli considerati.

Gli oggetti `BandDepthData` e `BandDepthRefData` sono costruibili specificando direttamente il valore delle variabili che essi contengono, oppure passando un oggetto `GetPot` e la sezione di cui fare il parsing.

```
class BandDepthData
{
public:
  typedef GetPot data_Type;

   BandDepthData( const UInt & nbPz, const UInt & nbPts, ..., const UInt & J, ... );
   BandDepthData( const data_Type & dataFile, const std::string & section );
};

class BandDepthRefData : public BandDepthData
{
public:
  BandDepthRefData( const UInt & nbPz, const UInt & nbPts, ...  );
  BandDepthRefData( const data_Type & dataFile, const std::string & section );
};
```

La separazione tra costruzione e setup di `BandDepth<_J>` e `BandDepthRef<_J>` ha il vantaggio di rendere più semplice l'inizializzazione degli oggetti, che sono costruibili, rispettivamente, ricevendo variabili `BandDepthData` e `BandDepthRefData`, per copia o attraverso `shared_pointer`.

```
template < UInt _J >
class BandDepth : public BandDepthBase< All >
{
public:
     BandDepth( const bdData_Type & bdData );
};



template < UInt _J >
class BandDepthRef : public BandDepthBase< Reference >
{
```

```
public:
 BandDepthRef( const bdRefData_Type & bdRefData );
};
```

Setup di reference/test set per `BandDepthRef<_J>`

La classe `BandDepthRef<_J>` ha un'interfaccia pubblica differente da quella di `BandDepth<_J>`, dovendo fornire delle funzionalità per il setup di reference-set e test-set. Questo, tra l'altro, è anche il motivo per cui esse devono derivare da classi `BandDepthBase<_policy>` specializzate diversamente.

Come anticipato, la separazione in sottoinsiemi di riferimento e di test viene effettuata sfruttando il dataset a livelli `DataSetLevelled`. A tal proposito `BandDepthRef<_J>` mette a disposizione due metodi principali, la cui azione combinata (e ordinata) permette di impostare la divisione in sottopopolazioni.

```
template < UInt _J >
class BandDepthRef : public BandDepthBase< Reference >
{
public:
  void addToReferenceSet( const UInt & levelID, const UInt & size, const UInt & seed = 1
      );
  void setTestSet();
  void clearReferenceSet();
};
```

`addToReferenceSet` permette di aggiungere un numero pari a `size` di elementi, provenienti dal livello `levelID`, all'insieme di riferimento. Se `size` è minore del numero di elementi del livello specificato, essi vengono estratti sfruttando un generatore di numeri pseudo-casuali inizializzato con `seed`.
Una volta che all'insieme di riferimento sono stati aggiunti tutti i segnali desiderati (che comunque devono essere in numero minore a quelli espressi nell'oggetto `BandDepthRefData`), si può finalizzarne la costruzione e creare l'insieme di test attraverso `setTestSet()`, che vi aggiunge tutti i restanti elementi del dataset.
Il reference-set può essere resettato con `clearReferenceSet()`.

Il metodo `computeBDs()`

Il metodo più importante dell'interfaccia pubblica di `BandDepth<_J>` e `BandDepthRef<_J>` è `computeBDs()`, con cui vengono effettivamente calcolati i contributi $\mathrm{MBD}_N^{(j)}(f)$, e che rappresenta il punto computazionalmente più oneroso del calcolo delle misure di profondità, presentando una complessità crescente rispetto all'aumento del parametro `_J` proporzionale alle combinazioni di N dati a gruppi di `_J`.
Come anticipato nella Nota 5.3, esso viene specializzato rispetto al valore di `_J` poiché nel caso `_J= 2` è possibile sfruttare un algoritmo molto più efficiente, proposto da Genton in [SGN12].

Nel caso di `_J> 2` il calcolo di $\mathrm{MBD}_N^{(j)}(f)$ segue la definizione:

$$\mathrm{MBD}_N^{(j)}(f) = \binom{N}{j}^{-1} \sum_{1 \le i_1 < i_2 < \ldots < i_j \le N} \lambda_r \left(A(f; f_{i_1}, f_{i_2}, \ldots, f_{i_j}) \right),$$

Per ogni segnale nel dataset e per ogni punto del dominio viene calcolato il numero di combinazioni dei valori delle restanti funzioni che includono il valore del segnale corrente, sommando sui punti del dominio I e standardizzando infine per la sua misura.
L'idea dell'algoritmo di Genton, proposto per `_J= 2`, è quella di ordinare i segnali ad ogni istante di tempo fissato e sfruttare le informazioni deducibili dai ranghi. Considerando il valore in $f_i(t_0)$ del segnale i-esimo al tempo t_0 , dato il suo rango, $k(t_0)$, rispetto ai valori delle altre funzioni, allora esso sarà maggiore di $n_i(t_0) = k(t_0) - 1$ valori e minore di $m_i(t_0) = N - k(t_0)$, dove N indica il numero di funzioni nel dataset.
Nella formula che definisce $\mathrm{MBD}_N^2(f)$:

$$\mathrm{MBD}_N^2(f) = \binom{N}{2}^{-1} \sum_{1 \le i_1 < i_2 <= N} \lambda_r \left(A(f; f_{i_1}, f_{i_2}) \right),$$

il sottoinsieme $A(f; f_{i_1}, f_{i_2} \subset I$ degli istanti per cui $f_i(t) \in \left(\min\{f_{i_1}(t), f_{i_2}(t)\}, \max\{f_{i_1}(t), _{i_2}(t)\} \right)$ conterrà t_0 solo per un numero limitato delle possibili coppie del campione, in particolare $n_i(t_0) \cdot m_i(t_0)$.
Si ha dunque a disposizione la formula esatta:

$$\mathrm{MBD}_N^2(f) = \binom{N}{2} \int_I \frac{n(t) \cdot m(t)}{\lambda(I)} \mathrm{d}t \approx \binom{N}{2} \sum_{i=0}^{M} \frac{n(t_i) \cdot m(t_i)}{\lambda(I)},$$

dove $\{t_0, t_1, \ldots, t_M\}$ è la suddivisione dell'intervallo I rispetto alla quale si conoscono i valori dei segnali del campione.

La strategia appena delineata può, in linea di principio, essere estesa anche per alcuni valori di `_J> 2`, determinando delle formule esplicite analoghe a quella per `_J= 2`; questo sarà parte di una successiva estensione di `HPCS`. Si noti come anche in questo caso la scelta di rendere le classi `BandDepth<_J>` e `BandDepthRef<_J>` template rispetto al parametro `_J` per poi specializzare il metodo `computeBDs()` sia appropriata

Parallelizzazione di `computeBDs()`

Rappresentando il metodo che richiede maggior sforzo computazionale, in particolare quando `_J= 2` oppure se il numero di elementi del dataset è grande, in `computeBDs()` il carico di lavoro viene partizionato tra tutti i processi disponibili. La strategia di parallelizzazione è uguale per `BandDepth<_J>` e `BandDepthRef<_J>` nei casi di `_J> 2`, ossia quando l'algoritmo per il calcolo delle profondità implementa la definizione. Nel caso di `_J= 2`, cioè quando viene utilizzato l'algoritmo di Genton, la strategia viene opportunamente modificata.
Nel caso `_J`$\neq$ `2` si deve considerare che:

- per il calcolo di un singolo termine $\mathrm{MBD}_N^{(j)}(f_i)$ è necessario confrontare il segnale i-esimo con tutti i restanti segnali da processare, quindi la struttura dati che li raccoglie deve essere

accessibile da tutti i processi. Per diminuire la comunicazione con granularità bassa, ma a discapito della memoria impiegata da ogni processo, il dataset viene letto durante il setup dell'oggetto `BandDepth<_J>` o `BandDepthRef<_J>` da tutti i processi.

- l'obiettivo di scalabilità interessante non riguarda il valore di `_J`, che influenza molto la complessità del calcolo ma è pensato per rimanere comunque basso (nella pratica, visto lo sforzo richiesto, si utilizza `_JJ= 2, 3, 4`), piuttosto riguarda il numero di segnali da analizzare. Quindi in presenza di dataset di dimensioni anche molto differenti, fissando il valore di `_J`, idealmente si può trovare il numero di processi minimo affinché il calcolo richieda un tempo minore di una soglia desiderata per ognuno di essi.

In seguito a queste considerazioni si è pensato di suddividere il numero di segnali da processare (cioè tutti nel caso normale e quelli del test-set nel caso con riferimento) tra i processi a disposizione in maniera bilanciata, attribuendo eventuali residui al master. Ogni processore calcola la profondità dei segnali corrispondenti e al termine un broadcast collettivo allinea le informazioni di tutti i processi.

Nel caso di `_J= 2`, a causa dell'algoritmo utilizzato, la strategia che porta ad un'esecuzione meno complessa è differente: nel caso di `BandDepth<_J>` i processi suddividono inizialmente il dataset rispetto all'asse temporale, e ognuno di essi produce una struttura che riporta l'ordinamento del dataset per ogni istante di tempo. Al termine, una comunicazione collettiva trasmette a tutti i processi le strutture costruite, che vengono poi assemblate. Nel passo seguente il dataset viene partizionato rispetto al numero di segnali e viene calcolata la profondità di ciascuno, con successiva comunicazione collettiva finale per scambiare le informazioni di ciascun processo.
Nel caso di `BandDepthRef<_J>`, invece, è più conveniente dal punto di vista implementativo partizionare unicamente il dataset temporale, e costruire in ogni processo l'ordinamento del chunk di reference set corrispondente, con cui poi testare tutti gli elementi del test-set. Al termine viene svolta una riduzione collettiva tra tutti i processi.

Utilizzo tramite `BDFactory<BDPolicy _policy>`

L'istanziazione di un oggetto `BandDepth<_J>` o `BandDepthRef<_J>` richiede la conoscenza del valore del parametro `_J`a compile-time, mentre è preferibile che questa informazione sia esprimibile a run-time, e impostabile esternamente attraverso un opportuno file data letto da `GetPot`, così da avere la massima flessibilità nell'utilizzo del codice. La struttura gerarchica di queste classi, che derivano da `BandDepthBase<_policy>`, classe template indipendente dal valore di `_J`, permette di conciliare le due esigenze costruendo un'apposita abstract factory (si veda ad esempio []), con cui delegare a run-time la costruzione dell'oggetto richiesto. La factory `BDFactory<BDPolicy _policy>` deriva da una classe base che implementa una factory generica, e il costruttore viene specializzato per i valori `_policy=All` e `_policy=Reference`, visto che le classi `BandDepth<_J>` e `BandDepthRef<_J>` hanno in comune una classe padre che ha dovuto necessariamente subire una differente specializzazione dell'interfaccia pubblica:

```
template < class ProductType,
                 typename KeyType,
                 typename CreatorType
           >
class Factory
{
```

```
public:
  bool registerProduct( const key_Type & key, const creator_Type & rule );
  bool unregisterProduct( const key_Type & key );
  product_Type * create( const key_Type & key );
};

template < BDPolicy _policy >
class BDFactory
:
public Factory< BandDepthBase< _policy >, UInt, CreationRulePtrWrapper< BandDepthBase<
    _policy > > >
{...};

template <>
BDFactory< All >::
BDFactory()
 :
 Factory< BandDepthBase< All >, UInt,  CreationRulePtrWrapper< BandDepthBase< All > > >()
{...}

template <>
BDFactory< Reference >::
BDFactory()
:
Factory< BandDepthBase< Reference >, UInt,  CreationRulePtrWrapper< BandDepthBase<
    Reference > > >()
{...}
```

Le regole di creazione dei prodotti nella factory sono implementate in una semplice gerarchia di funtori struct, il che, se emergesse la necessità, permetterebbe di adattare la regola allo specifico oggetto creato:

```
template < typename _returnType >
struct CreationRule
{
      CreationRule(){}

      virtual _returnType * operator()() const
      {
          return new _returnType();
      }
};

struct
CreateBDAll2 : public CreationRule< BandDepthBase< All > >
{
    CreateBDAll2(){}

    BandDepthBase< All > * operator()() const
    {
      return new BandDepth< 2 >();
    }
};

struct
CreateBDRef2 : public CreationRule< BandDepthBase< Reference > >
{
    CreateBDRef2(){}

    BandDepthBase< Reference > * operator()() const
    {
       return new BandDepthRef< 2 >();
     }
};
```

Sono fornite le implementazioni per i principali valori di `_J` utilizzabili, `_J= 2, 3, 4, 5`. Poiché nella sua dichiarazione la factory `BDFactory<_policy>` deve specificare un solo tipo di creatore, viene disegnato un semplice wrapper, usato come `Creator_Type` in `BDFactory<_policy>`, che accettando uno `shared_pointer` ad una `CretionRule` permette di usare in maniera polimorfica le regole di creazione, adattando la struttura delle regole di creazione dei prodotti all'esigenza di univocità del tipo creatore della factory.

```
template < typename _returnType >
 class CreationRulePtrWrapper
{
public:
  CreationRulePtrWrapper( const creationRulePtr_Type & rulePtr );
  _returnType * operator()() const {  return (*(this->M_creationRulePtr))(); }
};
```

5.2.5 Depth Measure

Il calcolo delle misure di profondità univariate modificate $\mathrm{MBD}_N^J(f)$ viene svolto dalla classe `DepthMeasure<_J,_policy>`. Considerando la definizione,

$$\mathrm{MBD}_N^J(f) = \sum_{j=2}^{J} \mathrm{MBD}_N^{(j)}(f).$$

essa si deve interfacciare con le classi `BandDepth<_J>` e `BandDepthRef<_J>` per calcolare i contributi $\mathrm{MBD}_N^J(f)$, e accumularne i valori per creare le misure di profondità univariate.

Dal punto di vista implementativo, essa è una classe template con due parametri, `_J` e la `_policy` di tipo `BDPolicy` che specifica se si sfrutti la procedura normale oppure la suddivisione in test-set/training-set.
`DepthMeasure<_J,_policy>` deriva da una classe base `DepthMeasureBase<_policy>`, indipendente dal parametro `_J`, che fornisce un'interfaccia pubblica essenziale, utile per usare gli oggetti `DepthMeasure<_J,_policy>` in maniera polimorfica. Essi possono essere costruiti ricevendo uno `shared_pointer` ad un oggetto `BandDepthData` o `BandDepthRefData` a seconda del valore di `_policy`. La conformità tra il valore di `_policye` l'oggetto corrispondente (`BandDepth<_J>` o `BandDepthRef<_J>`) è garantita dai seguenti `typedef`, che permettono di acquisire questa informazione direttamente dalle classi `BandDepthBase<_policy>`.

```
template < UInt _J, BDPolicy _policy >
class DepthMeasure : public DepthMeasureBase< _policy >
{
public:
  typedef BandDepthBase< _policy > bdBase_Type;
  typedef boost::shared_ptr< bdBase_Type > bdBasePtr_Type;
  typedef typename bdBase_Type::bdData_Type bdData_Type;
  typedef boost::shared_ptr< bdData_Type > bdDataPtr_Type;
};
```

I principali metodi pubblici di `DepthMeasure<_J,_policy>` sono `computeDepths()` e `computeRanks()`. Mentre `computeDepths()` è responsabile del calcolo delle profondità, `computeRanks()` permette di calcolare i ranghi corrispondenti all'ordinamento indotto sul dataset dal calcolo delle profondità. I dati con profondità più bassa, e quindi meno centrali nel campione, avranno un rango basso, mentre i dati con profondità alta, cioè i più centrali, avranno un rango alto.

Il metodo `computeDepths()`

Il metodo che calcola le profondità dei dati deve creare progressivamente oggetti di tipo `BandDepth<_J>` o `BandDepthRef<_J>` con valori di `_J` crescenti fino a quello dell'oggetto di tipo `DepthMeasure<_J,_policy>` corrente, inizializzarli e chiamare i loro metodi `computeBDs()`. Gli oggetti delle classi `BandDepth<_J>` e `BandDepthRef<_J>` creati in `computeDepths()`, tuttavia, devono subire inizializzazioni differenti che non possono essere ricondotte ad un'unica struttura comune, dovendo esplicitamente usare nel secondo caso i metodi per il setup di reference-set e test-set. Per questo motivo il metodo `computeDepths()` deve avere definizioni differenti nei due casi. Poiché la classe `DepthMeasure<_J,_policy>` ha due parametri template non è possibile specializzare parzialmente rispetto al valore di `_policy` solo `computeDepths()`, senza specializzare l'intera classe, cosa da evitare essendo piuttosto corposa.
Di conseguenza si è pensato di creare un nuovo oggetto, `computeImplement<_J,_policy>`, a cui affidare l'implementazione del metodo `computeDepths()`. Visto che questo metodo dipende sia dalla `_policy` che da `_J`, `computeImplement` dovrà essere una classe template con gli stessi due parametri, ma la sua specializzazione nei due casi `All` e `Reference` sarà molto più agevole, essendo una classe di limitata estensione.

```
template < UInt _J, BDPolicy _policy >
class computeImplement
{
public:
   computeImplement( const bdDataPtr_Type & bdDataPtr );
   void compute();
};
```

Utilizzo tramite `DMFactory<BDPolicy _policy>`

In maniera analoga al caso di `BandDepth<_J>` e `BandDepthRef<_J>`, anche gli oggetti della classe `DepthMeasure<_J,_policy>` possono essere utilizzati attraverso una factory apposita, `DMFactory<BDPolicy _policy>`, derivata dal tipo generico di abstract factory di `HPCS`. La sua definizione e la struttura ricalcano quanto visto per `BDFactory<_policy>`.

```
template < BDPolicy _policy >
struct CreateDM2 : public CreationRule< DepthMeasureBase< _policy > >
{
  CreateDM2(){}

  DepthMeasureBase< _policy > * operator()() const
  {
      return new DepthMeasure< 2, _policy >();
    }
};
```

```
template < BDPolicy _policy >
class DMFactory
:
public Factory< DepthMeasureBase< _policy >, UInt, CreationRulePtrWrapper<
    DepthMeasureBase< _policy > > >
{...};
```

5.2.6 Multivariate Depth Measure

Nel caso di dataset funzionale multivariato le misure di profondità corrispondenti sono determinate a partire dalle profondità univariate delle componenti attraverso una media con opportuni pesi:

$$\mathrm{MBD}_N^J(\mathbf{f}) = \sum_{k=1}^{s} p_k \, \mathrm{MBD}_{N,k}^J(f_k)$$

Il loro calcolo è possibile attraverso la classe `MultiDepthMeasure<_J,_policy>`, template rispetto ai parametri `_J` e `_policy`, derivata dalla classe base `MultiDepthMeasureBase<_policy>`.

```
template < BDPolicy _policy >
class MultiDepthMeasureBase
{...};

template < UInt _J, BDPolicy _policy >
class MultiDepthMeasure : public MultiDepthMeasureBase< _policy >
{...};
```

La presenza di più componenti nei dati funzionali è gestita abbinando ad un costruttore `void` il metodo `addDimension`, che permette di aggiungere una dimensione a quelle già presenti.

```
void addDimension( const getPot_Type & dataFile, const std::string & section );
```

Non dovendo interagire con le altre dimensioni in maniera diretta, ognuna di esse è trattata come un dataset funzionale a sé stante di cui calcolare le profondità univariate. In seguito esse vengono combinate per dar luogo alle MBD_N^J. Questo permette di riciclare le strutture usate per processare i dataset univariati già presenti nel codice. Inoltre, le varie dimensioni sono univocamente caratterizzabili attraverso i corrispondenti oggetti di dati, `BandDepthData` o `BandDepthRefData`, che vengono salvati in una lista di `shared_pointer`(per limitare i duplicati), memorizzata come membro privato della classe. Il metodo `addDimension` agisce proprio aumentando il contatore interno delle dimensioni e creando, a partire dalla variabile `GetPot` ricevuta, un oggetto data da aggiungere alla lista. Quando sarà necessario costruire l'oggetto `DepthMeasure<_J,_policy>` corrispondente alla dimensione desiderata, verrà selezionato dalla lista l'oggetto data relativo.

Un'altra funzionalità importante dell'interfaccia pubblica di `MultiDepthMeasure<_J,_policy>` è deputata all'impostazione dei pesi da usare nel calcolo effettivo delle profondità multivariate. È possibile specificare tali pesi sfruttando dei setters che li ricevano sotto forma di un contenitore

(tali setters sono template rispetto al contenitore o ai suoi iteratori), oppure usando un setter che riceva un oggetto `GetPot`da cui dedurre il file esterno in cui essi sono salvati, per poi poterli leggere.

I metodi più importanti di questa classe sono `computeMultiDepths()` e `computeMultiRanks()`, omologhi dei metodi `computeDepths()` e `computeRanks()` della classe `DepthMeasure<_J,_policy>`.

```
void computeMultiDepths();
void computeMultiRanks();
```

Il primo metodo, che calcola le misure di profondità multivariate, prende in considerazione sequenzialmente ogni dimensione e crea il corrispondente oggetto `DepthMeasure<_J,_policy>`, di cui chiama il metodo `computeDepths()` dopo averne fatto il setup attraverso l'oggetto data corrispondente a quella dimensione. Sempre iterativamente, le profondità vengono poi sommate con il peso apposito a quelle già ottenute. Il metodo `computeMultiRanks()` attribuisce i ranghi agli elementi del dataset in maniera concorde alla loro loro profondità.

Nota 5.5. Mentre nella classe `DepthMeasure<_J,_policy>`si è dovuta distingurere l'implementazione del metodo `computeDepths()` tra il caso `All` e `Reference`, qui non è necessario, poiché gli oggetti `DepthMeasure<_J,_policy>`stessi, creati nel metodo `computeMultiDepths()`, hanno un'interfaccia pubblica identica.

Utilizzo tramite `MDMFactory<BDPolicy _policy>`

Analogamente alle classi precedenti, anche questa sfrutta la propria classe base indipendente dal parametro template `_J`, `DepthMeasureBase<_policy>`, per realizzare una factory che renda molto flessibile l'utilizzo degli oggetti di tipo `MultiDepthMeasure<_J,_policy>`all'interno di files sorgente manipolabili esternamente con `GetPot`.
Dunque, similmente ai casi precedenti, si è implementata la classe `MDMFactory<BDPolicy _policy>`:

```
template < BDPolicy _policy >
struct CreateMultiDM2 : public CreationRule< MultiDepthMeasureBase< _policy > >
{
  CreateMultiDM2(){}

  MultiDepthMeasureBase< _policy > * operator()() const
  {
      return new MultiDepthMeasure< 2, _policy >();
    }
};

template < BDPolicy _policy >
class MultiDepthMeasureFactory
:
public Factory< MultiDepthMeasureBase< _policy >, UInt, CreationRulePtrWrapper<
    MultiDepthMeasureBase< _policy > > >
{...};
```

5.3 Risultati

In questa sezione vengono riportati i risultati della sintesi di elettrocardiogrammi a partire dalla soluzione numerica del problema bidominio e del loro test statistico rispetto ad una popolazione reale di riferimento.

5.3.1 Simulazione degli ECG numerici

Per la soluzione numerica del problema bidominio si decide di applicare lo schema (3.20), combinato con il modello ionico di Mitchell-Schaeffer, ad una discretizzazione agli elementi finiti che sfrutti la geometria cardiaca semplificata riportata in Figura 5.1 e una geometria toracica di riferimento, in cui la soluzione del problema lineare è calcolata sfruttando un'opportuna matrice di trasferimento. Come anticipato nella Sezione 3.1.1, tale scelta è stata applicata con successo nel tentativo di simulare correttamente i principali elementi di un ECG sano (si vedano, ad esempio, [Zem09], [Bou+10] e [BSG12]).

I parametri che definiscono il modello sono i tensori di conducibilità e i parametri ionici di Mitchell-Schaeffer. Per i primi si è scelto, conformemente a [Zem09] e [Bou+10], di considerare i tensori isotropi e si sono impostati i valori ivi riportati. Per quanto riguarda il modello di Mitchell-Schaeffer, i parametri caratterizzanti sono: τ_{in}, τ_{out}, τ_{open}, τ_{close}, V_{gate}, V_{min} e V_{max}. Ad esse si devono aggiungere le costanti generate durante il processo di omogeneizzazione nella formulazione numerica del problema bidominio, A_m e C_m.
I valori di queste costanti adatti ad una corretta simulazione dell'attività elettrica del cuore sono stati proposti in [Zem09] e [Bou+10], e sono riportati nella Tabella 5.1.
Si noti come all'unico parametro τ_{close} originariamente prescritto dal modello di Mitchell-Schaeffer si siano sostituiti altri quattro valori, $\tau_{\text{close}}^{\text{RV}}$, $\tau_{\text{close}}^{\text{endo}}$, $\tau_{\text{close}}^{\text{mcell}}$ e $\tau_{\text{close}}^{\text{epi}}$. Essi tengono conto dell'eterogeneità nella durata del potenziale d'azione (Action Potential Duration - APD) riscontrabile in differenti parti del miocardio (ad esempio tra base e apice, tra lato settale e posteriore e in direzione transmurale), uno dei fattori che influenzano la forma normale dell'onda T dell'ECG, in particolare la sua polarità. In [MS03] si trova che l'ordine dominante del massimo valore di APD fornito dal modello di Mitchell-Schaeffer è proporzionale al parametro τ_{close}, dunque l'eterogeneità nell'APD in direzione transmurale è espressa dalla scelta dei quatro valori locali di τ_{close}, assumendo che le cellule dell'epicardio abbiano la minore APD e quelle dell'endocardio abbiano una APD intermedia tra quella del miocardio medio (M-cells) e delle cellule epicardiali. In particolare, $\tau_{\text{close}}^{\text{endo}}$ è il valore relativo alle cellule vicine all'endocardio, $\tau_{\text{close}}^{\text{mcell}}$ è il valore nelle M-cells e $\tau_{\text{close}}^{\text{epi}}$ è quello vicino all'epicardio. Nel ventricolo destro si è considerato un valore costante e pari a $\tau_{\text{close}}^{\text{RV}}$.

A_m	C_m	τ_{in}	τ_{out}	τ_{open}	$\tau_{\text{close}}^{\text{endo}}$	$\tau_{\text{close}}^{\text{epi}}$	$\tau_{\text{close}}^{\text{mcell}}$	$\tau_{\text{close}}^{\text{RV}}$	V_{gate}	V_{min}	V_{max}
200 cm^{-1}	10^{-3} mF	8	180	300	130	140	90	120	-67	-80	20

Tabella 5.1: Valore delle costanti del modello ionico considerato nelle simulazioni.

Osservazione 5.6. Come descritto nel Capitolo 2, lo stimolo elettrico che si propaga nel tessuto cardiaco origina dal Nodo Seno-Atriale, nell'atrio destro, e si propaga ai ventricoli attraverso il

sistema di conduzione cardiaca. Nella geometria utilizzata per le simulazioni non sono presenti gli atrii, di conseguenza lo stimolo deve essere imposto esternamente. A tal fine viene applicata una data densità di corrente, per un breve periodo di tempo, ad un sottile strato sub-endocardiale in entrambi i ventricoli.

L'obiettivo di generare una popolazione di segnali ECG artificiali viene raggiunto risolvendo più volte il problema bidominio in corrispondenza di set di parametri differenti, ma tali che il risultato in termini di morfologia dell'ECG sia il più possibile realistico. Visto il grande numero di parametri che influenzano il modello, il costo piuttosto elevato della soluzione numerica del problema e l'esigenza di produrre un dataset artificiale di dimensioni sufficientemente grandi da consentire le analisi statistiche, si è scelto di variare un sottoinsieme di 2 parametri, scelti tra quelli che influenzano maggiormente l'ECG.
In [Zem09] viene eseguita un'analisi di sensitività dell'ECG rispetto ai parametri ionici del modello di Mitchell-Schaeffer, e si trova che tra i parametri più importanti vi sono τ_{in} e τ_{out}, che di conseguenza sono stati presi in considerazione per le analisi seguenti. Anche τ_{close} presenta una buona sensitività ma, come menzionato precedentemente, esso non può essere considerato omogeneo sulla mesh per avere una simulazione realistica dell'onda T (infatti, se fosse omogeneo, l'onda T dell'ECG avrebbe polarità invertita), e quindi dovrebbe essere considerato nei quattro valori locali.
I valori di τ_{in} e τ_{out} scelti per creare i campionamenti dello spazio dei parametri sono riportati nella Tabella 5.2, essi vengono accoppiati in maniera cartesiana e forniscono un insieme di $N = 30$ segnali ECG, che vengono riportati nella Figura 5.2.

τ_{in}	6	7	8	9	10	11
τ_{open}	170	175	180	185	190	

Tabella 5.2: Valore dei parametri τ_{in} e τ_{out} usati nelle simulazioni.

5.3.2 Validazione degli ECG sintetici

In questa sezione riportiamo i risultati del confronto degli ECG simulati con quelli reali provenienti dal database PROMETEO.
Per poter operare il confronto, gli $N = 30$ segnali simulati sono stati re-interpolati sulla griglia temporale dei segnali reali e riscalati alla stessa scala di potenziale. I sgnali reali, a loro volta, sono stati privati del tratto iniziale, comprendente l'onda P, espressione dell'attività elettrica atriale, che è mancante nei segnali simulati. Per fare questo ci si è avvalsi della presenza di landmark naturali, rispetto ai quali i dati reali sono stati allineati durante il pre-processing, che indicano in maniera precisa l'inizio del complesso QRS (si veda la Sezione 4.3).
Nella Figura 5.3 viene riportata la sovrapposizione dei segnali reali e quelli simulati, suddivisa nelle differenti derivazioni. A prima vista si può notare come i segnali simulati siano pienamente conformi a quelli reali da un punto di vista morfologico macroscopico, presentando tutti i tratti tipici delle derivazioni corrispondenti, ad eccezione di parte della derivazione II. Questo conferma la scelta della formulazione bidominio con modello di chiusura di Mitchell-Schaeffer come un ottimo strumento per riprodurre, attraverso la simulazione numerica, le caratteristiche principali dell'elettrocardiogramma.
Una discrepanza si può notare invece sui valori delle ampiezze dei segnali simulati rispetto a quelli sani, che sono generalmente minori (ad eccezione del picco QRS nella derivazione I) di

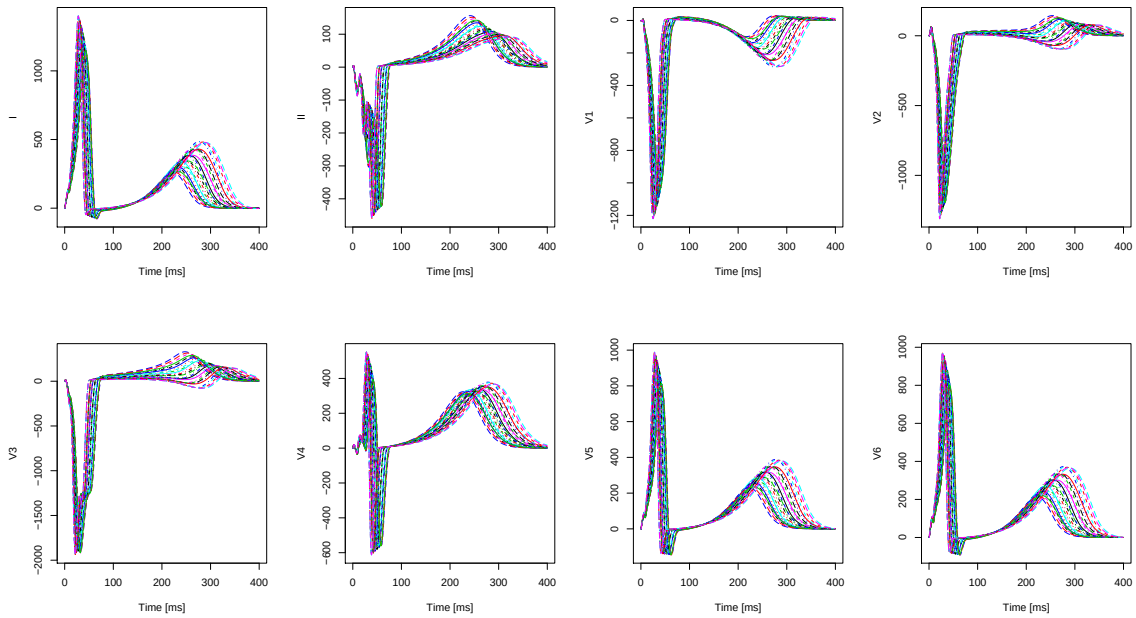

Figura 5.2: Popolazione di ECG sintetici determinati risolvendo il problema con la scelta dei parametri riportata in Tabella 5.2.

quelle dei segnali reali, specialmente in corrispondenza dell'onda T, e anche sulla localizzazione dell'onda T, che non sempre (ad esempio nella deriazione I) sembra essere allineata in fase rispetto ai segnali reali.

Per condurre un'analisi più quantitativa si è costruito un dataset composto sia da segnali reali che simulati. Ad esso si è applicata il test di Wilcoxon descritto nella Sottosezione 4.1.4, adatto a determinare se due popolazioni siano differenti sulla base delle loro misure di profondità.
Seguendo, appunto, la procedura delineata nella Sottosezione 4.1.4, si è scelto un campionamento di $N = 70$ segnali dai 101 ECG reali sani e lo si è utilizzato come sottoinsieme di riferimento per calcolare le profondità multivariate (con pesi uniformi) della restante parte dei segnali reali sani e degli ECG simulati.
La procedura è stata ripetuta 10 volte, per garantire che il risultato non dipendesse dal particolare campionamento del dataset di ECG reali (si riportano come esempio in Figura 5.4 le profondità univariate determinate in corrispondenza della prima prova). Come si può vedere la distribuzione delle profondità relative degli ECG simulati rispetto a quelli reali assume valori inferiori, una caratteristica che si riscontra anche nelle restanti prove.

Successivamente, si sono calcolate le profondità multivariate del solo sottoinsieme di riferimento e si è calcolata la proporzione di quelle minori o uguali a quelle del sottoinsieme di test, come da definizione. Il test di Wilcoxon è stato applicato ai ranghi indotti dall'ordinamento delle proporzioni determinate, ottenendo nelle dieci prove dei *p-values* compresi tra 1.85e-05 e 1.01e-09, che costringono a rifiutare l'ipotesi nulla, ossia che i segnali appartengano alla stessa popolazione.

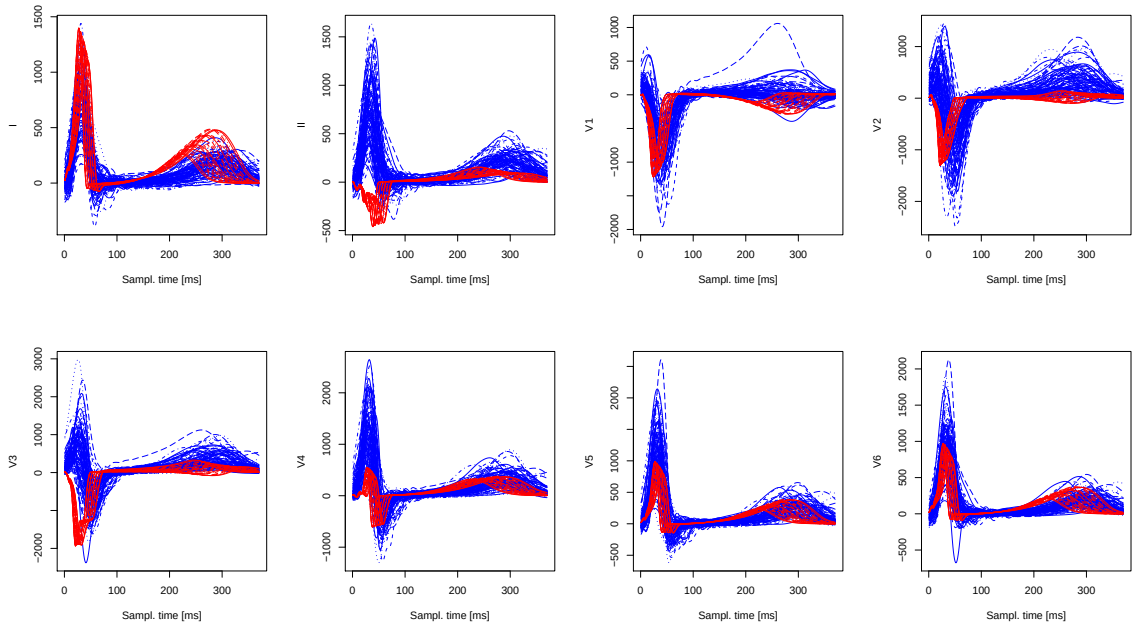

Figura 5.3: Popolazione di ECG reali (colore blu) e simulati in corrispondenza dei parametri della tabella 5.2 (colore rosso).

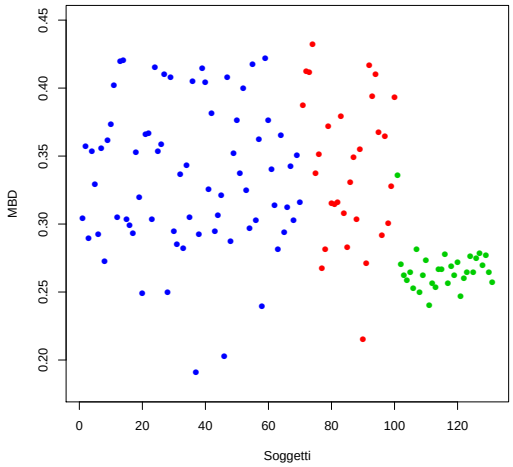

Figura 5.4: Rappresentazione delle MBD del dataset composito relative al sottoinsieme di $N = 70$ ECG reali sani estratti durante la prima delle 10 prove. In blu sono rappresentate le MBD calcolate nei 70 segnali di riferimento, in rosso e in verde, rispettivamente, le MBD dei restanti soggetti sani e degli ECG artificiali.

Si deduce dunque che, nonostante l'accordanza qualitativa tra segnali reali e fisiologici, il metodo statistico utilizzato non sembra fornire evidenza per ritenere gli ECG simulati come appartenenti alla stessa popolazione di quelli reali, almeno sulla base di indicatori morfologici quali le MBD. L'esame della dispersione delle profondità degli ECG reali rispetto a quelli simulati sulla base di grafici analogi a quello in Figura 5.4, in ognuno dei 10 casi, sembra suggerire che il motivo sia una differenza non tanto tra le medie delle profondità dei gruppi, quanto a una differenza tra le loro variabilità, che nel caso di dati sintetici è molto minore.

Capitolo 6

Conclusioni e Sviluppi Futuri

In questo lavoro di tesi si è cercato di determinare se l'uso di promettenti tecniche di analisi numerica per simulare l'elettrofisiologia cardiaca umana sia in grado di fornire dei segnali ECG artificiali confrontabili con quelli effettivamente registrabili nella pratica clinica, in particolare limitandosi a considerare il caso di tracciati normali (cioè non patologici).
A tal fine, da una parte è stata sfruttata la combinazione del modello bidominio per il potenziale elettrico cardiaco, accoppiato con il modello ionico di Mitchell-Schaeffer, che rappresenta la migliore descrizione dell'attività elettrica del cuore in ambito numerico; dall'altra si è costruita un'opportuna tecnica statistica di carattere non-parametrico, basata sul concetto di profondità di dati funzionali multivariati recentemente proposto in letteratura, per permettere il confronto tra una popolazione di dati artificiali e una di dati reali, tenendo opportunamente conto della variabilità tipica del fenomeno in esame.

I risultati trovati mostrano che la simulazione numerica degli ECG, attuata con le tecniche descritte, riesce a riprodurre efficacemente la struttura morfologica dei segnali appartenenti alla popolazione reale di riferimento, mentre non sembra essere ancora calibrata a sufficienza per poter generare pattern realistici anche nei dettagli, almeno limitatamente ai segnali generati a partire dai set di parametri scelti. In particolare, si sono notate delle discrepanze nell'allineamento orizzontale del segnale (specialmente riguardo la localizzazione dell'onda T in alcune delle derivazioni) e nell'ampiezza media dei segnali artificiali, le principali cause del rifiuto nel test di Wilcoxon attuato nella parte finale del lavoro.
La tecnica numerica utilizzata rimane un valido punto di partenza per sviluppare delle analisi grazie a cui allineare meglio l'output della simulazione al profilo degli ECG reali.

A tal proposito gli sviluppi futuri del presente lavoro saranno diretti ad arricchire e adattare sia le tecniche numeriche di riproduzione dei segnali, che quelle statistiche di analisi, oltre che ad affrontare le principali criticità emerse.
Sul fronte dell'analisi numerica bisognerà estendere il solutore numerico e considerare una geometria cardiaca completa anche degli atrii, così da avere un tracciato elettrocardiografico completo da confrontare interamente con quello dei soggetti reali. Dall'altra si dovrà esplorare in maniera più sistematica la dipendenza degli ECG sintetici dai parametri ionici del modello, trovando le combinazioni che permettano di produrre dei profili ECG maggiormente aderenti a

quelli reali. Questo potrà essere fatto sfruttando un approccio greedy, oppure integrando anche a questo livello opportuni metodi statistici e campionare nello spazio dei parametri a partire da distribuzioni di probabilità assegnate a priori. Un esame più dettagliato della dipendenza dai parametri degli ECG, anche in corrispondenza di mesh realistiche, richiederà inoltre di valutare l'applicazione di specifiche tecniche per la riduzione del costo computazionale della soluzione del problema elementi finiti anche da un lato metodologico, e non soltanto beneficiando dell'esecuzione parallela del solutore.
Sul fronte dell'analisi statistica bisognerà affinare l'uso delle misure di profondità multivariate, trovando un criterio per la scelta dei pesi usati per calcolarle che inglobi informazioni quantitative sulla struttura di covarianza delle derivazioni, arrivando dunque ad un adattamento molto maggiore della tecnica al problema in esame e influenzando positivamente l'esito dei test non-parametrici che su di esse si basano.

Una volta che si sarà riusciti a simulare ECG sintetici con profili simili, nel preciso senso statistico dell'esito di un opportuno test non-parametrico (ad esempio quello di Wilcoxon), sarà possibile concentrarsi sulla sintesi di una popolazione artificiale che non solo sia della stessa famiglia di quella reale, ma anche che abbia la stessa struttura di varianza-covarianza, garantendo a tutti gli effetti che la procedura numerica riesca a simulare correttamente i segnali concretamente riscontrabili nella pratica clinica.

Bibliografia

[AP96] R.R. Aliev e A. V. Panfilov. "A simple two-variable model of cardiac excitation". In: *Chaos, Solitons and Fractals* 7.3 (1996), pp. 293 –301.

[Bou+07] M. Boulakia et al. "Towards the Numerical Simulation of Electrocardiograms." In: *FIMH*. A cura di F.B. Sachse e G. Seemann. Vol. 4466. Lecture Notes in Computer Science. Springer, 26 lug. 2007, pp. 240–249.

[Bou+08] M. Boulakia et al. "A Coupled System of PDEs and ODEs Arising in Electrocardiograms Modeling". In: *Applied Mathematics Research eXpress* 2008.abn002 (2008). ISSN: 1687-1200.

[Bou+10] M. Boulakia et al. "Mathematical Modeling of Electrocardiograms: A Numerical Study". In: *Annals of Biomedical Engineering* 38.3 (2010). RR-6977 RR-6977 CardioSense3D, pp. 1071–1097.

[BR77] G. W. Beeler e H Reuter. "Reconstruction of the action potential of ventricular myocardial fibres". In: *The Journal of Physiology* 268.1 (1977), pp. 177–210.

[BSG12] M. Boulakia, E. Schenone e J-F. Gerbeau. "Reduced-order modeling for cardiac electrophysiology. Application to parameter identification". In: *International Journal for Numerical Methods in Biomedical Engineering* 28.6-7 (2012), pp. 727–744.

[Cla+11] RH Clayton et al. "Models of cardiac tissue electrophysiology: progress, challenges and open questions". In: *Progress in Biophysics & Molecular Biology* 104.1-3 (2011), pp. 22–48.

[Fit61] R. Fitzhugh. "Impulses and Physiological States in Theoretical Models of Nerve Membrane". In: *Biophysical Journal* 1.6 (1961), pp. 445–466.

[FK98] F. Fenton e A. Karma. "Vortex dynamics in three-dimensional continuous myocardium with fiber rotation: Filament instability and fibrillation". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 8.1 (1998), pp. 20–47.

[FM01] R. Fraiman e G. Muniz. "Trimmed means for functional data". In: *Test* 10.2 (2001), pp. 419–440.

[Hel53] H. Helmholtz. "Ueber einige Gesetze der Vertheilung elektrischer Ströme in körperlichen Leitern mit Anwendung auf die thierisch-elektrischen Versuche". In: *Annalen der Physik* 165.6 (1853), pp. 211–233. ISSN: 1521-3889.

[HH52] A.L. Hodgkin e A.F. Huxley. "A quantitative description of membrane current and its application to conduction and excitation in nerve." In: *The Journal of physiology* 117.4 (1952), pp. 500–544.

[Iev12] F. Ieva. "Statistical Methods for classification in cardiovascular healthcare". Tesi di dott. Politecnico di Milano, 2012.

[IP13] F. Ieva e A. M. Paganoni. "Depth Measures for Multivariate Functional Data". In: *Communications in Statistics - Theory and Methods* 42.7 (2013), pp. 1265–1276.

[Liu90] R. Y. Liu. "On a notion of data depth based on random simplices". In: *The Annals of Statistics* 18 (1990), pp. 405–414.

[LL04] J. Li e R. Liu. "New nonparametric tests of multivariate locations and scales using data depths". In: *Statistical Science* 19 (2004), pp. 686–696.

[LPR06] S. López-Pintado e J. Romo. "Depth-based inference for functional data". In: *Computational Statistics and Data Analysis* 51.10 (2006), pp. 4957 –4968.

[LPR09] S. López-Pintado e J. Romo. "On the Concept of Depth for Functional Data". In: *Journal of the American Statistical Association* 104.486 (2009), pp. 718–734.

[LR91] C. H. Luo e Y. Rudy. "A model of the ventricular cardiac action potential. Depolarization, repolarization, and their interaction." In: *Circulation Research* 68.6 (1991), pp. 1501–1526.

[LR94] C. H. Luo e Y. Rudy. "A dynamic model of the cardiac ventricular action potential. I. Simulations of ionic currents and concentration changes." In: *Circulation Research* 74.6 (1994), pp. 1071–1096.

[LS93] R. Y. Liu e K. Singh. "A Quality Index Based on Data Depth and Multivariate Rank Tests". In: *Journal of the American Statistical Association* 88.421 (1993), pp. 252–260.

[MS03] C.C. Mitchell e D.G. Schaeffer. "A two-current model for the dynamics of cardiac membrane". In: *Bulletin of Mathematical Biology* 65.5 (2003), pp. 767–793.

[NK93] J.C. Neu e W. Krassowska. "Homogenization of syncytial tissues". In: *Critical Reviews in Biomedical Engineering* 21 (2 1993), pp. 137–199.

[Nob62] D. Noble. "A modification of the Hodgkin-Huxley equation applicable to Purkinje fiber action and pacemaker potentials". In: *J. Physiol.* 160 (1962), pp. 317–352.

[PSCF05] M. Pennacchio, G. Savaré e P. Colli Franzone. "Multiscale Modeling for the Bioelectric Activity of the Heart". In: *SIAM J. Math. Analysis* 37.4 (2005), pp. 1333–1370.

[RM94] J. M. Rogers e A. D. McCulloch. "A collocation-Galerkin finite element model of cardiac action potential propagation". In: *IEEE Transactions on Biomedical Engineering* 41.8 (1994), pp. 743–757.

[RS05] J. O. Ramsay e B. W. Silverman. *Functional Data Analysis*. Springer, 2005.

[SG11] Y. Sun e M. G. Genton. "Functional Boxplots". In: *Journal of Computational and Graphical Statistics* 20.2 (2011), pp. 316–334.

[SG12] Y. Sun e M. G. Genton. "Adjusted functional boxplots for spatio-temporal data visualization and outlier detection". In: *Environmetrics* 23.1 (2012), pp. 54–64.

[SGN12] Y. Sun, M.G. Genton e D. Nychka. "Exact fast computation of band depth for large functional datasets: How quickly can one million curves be ranked?" In: *Stat* 1 (2012), pp. 68–74.

[Tuk74] J.W. Tukey. "Mathematics and the Picturing of Data". In: *International Congress of Mathematicians 1974*. A cura di R.D. James. Vol. 2. Vancouver, 1974, pp. 523–532.

[Tun78] L. Tung. "A bi-domain model for describing ischemic myocardial D-C potentials". Tesi di dott. MIT, Cambridge, MA, 1978.

[Zem09] N. Zemzemi. "Étude théorique et numérique de l'activité électrique du cœur: Applications aux électrocardiogrammes". Tesi di dott. Université Paris-Sud XI, 2009.

[ZS00] Y. Zuo e R. Serfling. "General notions of statistical depth function". In: *The Annals of Statistics* 28 (2000), pp. 461–482.

[Zuo03] Y. Zuo. "Projection Based Depth Functions and Associated Medians". In: *The Annals of Statistics* 31 (2003), pp. 1460–1490.