Eötvös Loránd Tudományegyetem
Informatikai Kar

EKG jelek feldolgozása Hermite-függvények segítségével

BSc Szakdolgozat

Készítette: Dózsa Tamás
ELTE IK
Programtervező informatikus
BSc

Témavezető: Dr. Kovács Péter
Adjunktus
ELTE IK
Numerikus Analízis Tanszék

Budapest, 2016.11.16.

Tartalomjegyzék

1. fejezet

Bevezetés

Az információ ábrázolásának módja az informatika tudomány fontos kérdése. Természetesen annak eldöntése, hogy egy adott adat halmaz milyen módon kerül ábrázolásra erősen függ annak jellegétől. Az adatábrázolás felel az adatok hatékony felhasználhatóságáért (például egy internetes video hívásnál az adatokat gyorsan kell egymás után továbbítani), ugyanakkor biztosítania kell, hogy az adatokból kinyerhető információ nem veszik el (ha túl rossz mindőségű képeket továbbítunk, a fogadó fél nem tudja értelmezni azokat).

A szakdolgozat célja egy speciális adatábrázolás, nevezetesen EKG jelek egy ábrázolásának bemutatása. Mivel jellemzően ezeket az adatokat, későbbiekben *jelek*-et, általában nagyobb memória felhasználásával szokás ábrázolni, ezért a dolgozat az eljárásra EKG jelek *tömörítéseként* hivatkozik. EKG jelek esetén az ábrázolás minősége sok szemponttól függ. Mivel ezek a jelek fontos információkat hordoznak a szív állapotáról, különösen fontos, hogy a tömörítés során ne vesszen el fontos információ. Egy ilyen jel rögzítésekor azonban sok olyan adat is tárolásra kerül (például a végtagok mérés közbeni mozgatása miatt), amelyek nem hordoznak fontos információt. Az ilyen adatokra a dolgozat *zaj*-ként hivatkozik. Egy jó EKG ábrázolás sikeresen szűri a mérés során keletkezett zajt, miközben az orvosi szempontból fontosnak nevezhető információt megtartja. Mivel a dolgozatban egy tömörítési eljárás kerül bemutatásra, fontos szempont az EKG jelek memória takarékos ábrázolása. Gyakorlati szempontból minél kevesebb memórián történik meg a jelek ábrázolása, annál könnyebb azokat tárolni (hosszú mérések esetén fontos lehet), illetve egyszerűbb és biztonságosabb a jelek hálózaton történő továbbítása. EKG jeleknek egy igazán jónak nevezhető ábrázolása pedig az eddig említettek mellett az orvosok munkáját közvetlenül segítő információt is kódol magában. Ilyen lehet például egy olyan ábrázolás amely hatékony bemenetéül szolgál valamilyen osztályozó algoritmusnak, lehetővé téve az abnormális jelek automatikus felismerését.

A dolgozat három fő fejezetre tagolható. A Bevezetés című fejezetben található a dolgozatban bemutatott eljárás specifikációja, a jel reprezentáció matematikai modelljének ismertetése, valamint a bemutatott módszer egyéb jellemzőit ismertető alfejezetek. Ezek közé tartozik például a jel közelítésének optimalizációjához szükséges Nelder-Mead algoritmus elméleti bemutatása, illetve az EKG jel szegmentációját

elősegítő Matching-Pursuit algoritmust részletező alfejezet.

A dolgozat második fejezete az eljárás mellékelt implementációjának a fejlesztői dokumentációja. Ebben a fejezetben találhatóak a tömörítő eljárást implementáló c++ osztályok jellemzői, illetve az elérést segítő webes felület implementációjának részletes ismertetése. A fejlesztői dokumentáció fejezet tartalmazza továbbá a program logikai jellemzését elősegítő UML és egyéb osztálydiagrammokat. A fejezet igyekszik pragmatikusan és érthetően jellemezni a program felépítését, illetve kellően megindokolni az egyes implementációk mellett szóló döntéseket.

A Felhasználói dokumentáció fejezetbe, a program használatával kapcsolatos információk kerültek. Ebben a fejezetben található a felhasználói felület funkcióinak pontos ismertetése, valamint hasznos példák annak használatára. Bemutatásra kerülnek továbbá a program használatához szükséges előkészületi lépések, és az ismert rendszerkövetelmények.

A dolgozat utolsó része a függelék, melyben a Bevezetés fejezetben található matematikai állítások bizonyításai, egyéb kapcsolódó matematikai fogalmak leírásai, illetve felhasznált algoritmusok pszeudo-kódja található.

1.1. A feladat specifikációja

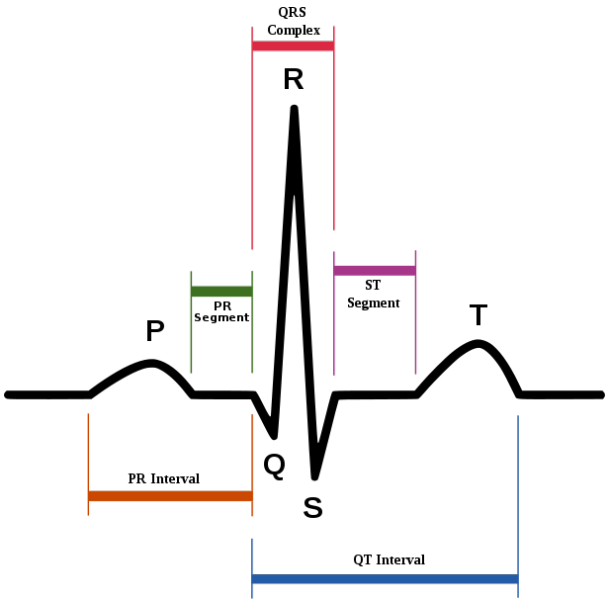
A dolgozat célja egy olyan tömörítési eljárás bemutatása, amely lehetővé teszi az EKG jelek hatékony (memória takarékos) ábrázolását, a mérések zaj szűrését, illetve az EKG hullámszegmenseinek szeparációját. Az utóbbi jellemző orvosi szempontból lehet hasznos, ugyanis sok kóros elváltozás kimutatásához szükséges az egyes hullámszegmensek széleinek ismerete. A dolgozatban bemutatott módszer hatékonysága, az irodalomban fellelhető más tömörítési eljárások [] hatékonyságának összehasonlításával igazolandó.

A dolgozatban bemutatott tömörítési eljárás implementációjának feladata, hogy az MIT-BIH adatbázisban található EKG jelek tömörítésére alkalmas legyen, ehhez pedig egy könnyen használható webes felületet biztosítson. Az implementációnak két féle bemente megengedett: tömörítés esetén az MIT-BIH adatbázisban megtalálható EKG jelek, illetve a tömörítés következtében létrejött számsorozat, melyet a tömörített jel helyreállításához használ.

1.2. A modell ismertetése

A modern orvostudományban nagy jelentőséggel bírnak a valamely élő szervezet által kibocsátott úgynevezett biológiaijelek. Ezek közé sorolható az Elektro Kardio Gram, vagy EKG, amely a szív állapotáról képes információt adni. Bár ennek a dolgozatnak nem célja az EKG jelek pontos elemzése, fontos néhány sorban ismertetni egy átlagos EKG jel meghatározó hullámait. Egyetlen szí vütés EKG reprezentációja három fő részre bontható: a szí vütés elején megjelenő P hullámra, az ezt követő QRS

komplexumra, és az ütés végén található T hullámra. Ezek rendre a pitvari összehűzódást, a kamrák depolarizációját és elektromos újratöltődését reprezentálják. Diagnosztikai szempontból a QRS komplexus a legfontosabb, ezért ezt nagy pontossággal kell tárolni. Általánosságban elmondható, hogy ezeknek a hullámoknak kezdő és végpontjai, valamint maximum és minimum értékei vesznek részt az orvosi diagnosztikában. Az említett paraméterek az 1.1 ábrán láthatóak.



1.1. ábra. Az EKG jel egy szívütése, illetve annak főbb diagnosztikai jellemzői.

Az irodalomban ismert tömörítő algoritmusokat [?] alapján három kategóriába sorolhatjuk: 1) egyszerű paraméteres becslések (pl.: interpoláció, különbségi kódolás, stb.), 2) direkt módszerek (pl.: csúcsok, meredekségek, stb. tárolása), 3) transzformációs eljárások. Az utóbbi osztály tartalmazza azokat az algoritmusokat, melyek a jelet egy előre adott függvényrendszer szerinti sorfejtéssel approximálják. Így az eredeti adatsorozat helyett csak az együtthatókat és a rendszer paramétereit kell tárolnunk. Ezen kategóriába sorolandó a dolgozatban bemutatott algoritmus is. Nevezetesen, az eredeti adatsorozatot speciális, Hermite-polinomok segítségével előállitott függvényrendszerrel fogjuk közelíti. A módszer alapját képező eljárás [?], jól ismert az irodalomban, mely nem csak a jelek tömörítéséhez, de azok modellezéséhez [?], illetve osztályozásához [?, ?] is alkalmazható. A dolgozatban az EKG jelekkel való hasonlóságuk miatt Hermite-függvényeket használunk az adatok reprezentálásához. Ezeket egy argumentum transzformáción keresztül szabad paraméterekkel egészítjük ki. Ennek köszönhetően az eredeti jelet egy adaptív bázisban írhatjuk fel. Az említett paraméterek megválasztásához a Nelder-Mead optimalizációs eljárást alkalmaztuk. Mivel az EKG jelek diszkrét adatsorozatok, ezért a módszert [?] alapján implementáltuk diszkrét ortogonális Hermite-polinomokra is. A dolgozatban különböző tesztekkel demonstráljuk az algoritmus hatékonyságát. Ehhez, több órányi, zajjal terhelt, valódi EKG felvételt használtunk. Ezen keresztül a bemutatott módszert összehasonlítottuk

több másik, az irodalomban jól ismert tömörítő algoritmussal is [?].

A tömörítő eljárást egy c++ nyelven megírt, objektum elvű alkalmazás implementálja, melyet egy webes felületen keresztül érhetünk el. A felület lehetőséget biztosít a dolgozatban jelölt tesztek újrafuttatására, valamint a teszteléskor felhasznált adatbázis további jeleinek a tömörítésére. Szintén a webes felületen keresztül nyílik alkamunk a már tömörített EKG jelek helyreállítására.

Az alkalmazás megetrvezésekor külön hangsúlyt kapott a kód újra felhasználhatósága. Ennek érdekében a felhasznált algoritmusok, illetve matematikai modellek a lehető legáltalánosabb formában lettek implementálva. Fontos szempontot jelentett továbbá a c++11-es nyelvszabvány által nyújtotta lehetőségek minél hatékonyabb kihasználása. Jó példa erre a lambda függvények alkalmazása az optimalizációs algoritmusok implementációja során. A hatékony működés mellett azonban a program igyekszik megfelelni a modern felhasználók igényeinek. Ennek érdekében a felhaszálói felület weboldalként lett implementálva. A rendszerfüggetlen, és installáció mentes elérés lehetővé teszi a gyors és egyszerű használatot, valamint az eredmények megosztását.

1.3. Matematikai háttér

1.3.1. Jelek approximációja

EKG jelek feldolgozásakor sok esetben szembesülünk gyakorlati kihívásokkal. Két sűrűn előforduló példa a hosszú mérések tárolása, valamint a zajjal terhelt mérések ábrázolása. Ezekre a nehézségekre egyszerre ad kielégítő megoldást, ha a jeleket valamely $\mathcal{H}$ Hilbert-tér sima függvényeiből álló $(\Phi_n, n \in \mathbb{N})$ ortogonális bázisában reprezentáljuk és a jelet véges sok $\Phi_0, \Phi_1, \cdots, \Phi_n$ bázisbeli elem lineáris kombinációjával közelítjük. Az $f \in \mathcal{H}$ jel legjobb közelítését a tér $\|\cdot\|$ normájában az

$$S_n f := \sum_{k=0}^{n} \langle f, \Phi_k \rangle \Phi_k$$

leképezés nyújtja, ahol $\langle \cdot, \cdot \rangle$ az $\mathcal{H}$ tér skaláris szorzatát jelöli. A jel és a közelítés eltérésének négyzete a

$$\|f - S_n f\|^2 = \|f\|^2 - \sum_{k=0}^{n} |\langle f, \Phi_k \rangle|^2$$

képplettel adható meg. Adott hibán belüli közelítést véve a jel helyett elég az $S_n f$ approximációt reprezentáló $\langle f, \Phi_k \rangle$ $(k = 0, 1, \cdots, n)$ Fourier-együtthatókat tárolni. Zajos jel esetén az ilyen típusú approximáció szűrőként is szolgál. A közelítés megvalósításához a klasszikus ortogonális rendszerek közül EKG görbék közelítésére az Hermite-féle függvények bizonyultak használhatónak. Ezt támasztják alá a [...] dolgozatok. Az Hermite függvények alkalmazása azzal is indikolható, hogy grafikonjuk hasonlít az EKG görbékre. Ezt a tulajdonságot a ?? ábra szemlélteti.

1.3.2. Hermite-függvények

A dolgozatban az $\mathbb{R}$ számegyenesen (Lebesgue-mérték szerint) négyzetesen integrálható függvények $\mathcal{H}$ Hilbert-tere helyett elegendő a szakaszonként folytonos, az $\mathbb{R}$-en négyzetesen integrálható függvények $\mathcal{F}$ euklideszi terét használni. Ebben a térben a skaláris szorzat és a norma a következő alakban írható fel:

$$\langle f, g\rangle := \int_{-\infty}^{\infty} f(t)g(t)\,dt, \quad \|f\| := \sqrt{\langle f, f\rangle} \quad (f, g \in \mathcal{F})\,. \tag{1.1}$$

Továbbá, a

$$\Phi_n(x) := H_n(x)e^{-x^2/2}/\sqrt{\pi^{1/2}2^n n!} \quad (n \in \mathbb{N})$$

normált Hermite-függvények (teljes) ortonormált rendszert alkotnak az $\mathcal{F}$ téren:

$$\langle \Phi_n, \Phi_m\rangle = \delta_{nm} \quad (m, n \in \mathbb{N}), \quad \|f - S_n f\| \to 0 \ (n \to \infty)\,.$$

Itt H_n $(n \in \mathbb{N})$ jelöli az Hermite-féle polinomokat.

Az Hermite-függvények alkalmazásának számos előnye van:

i) A Φ_n $(n \in \mathbb{N})$ rendszer zárt (teljes) az $\mathcal{F}$ téren.

ii) A $\Phi_n(x)$ függvények gyorsan tartanak 0-hoz, ha $|x| \to \infty$:

$$|\Phi_n(x)| \le M_n e^{-x^2/4} \le M_n \quad (x \in \mathbb{R}, n \in \mathbb{N}).$$

iii) A Φ_n függvények (stabil) másodrendű rekurzióval számíthatók:

$$\begin{aligned} &\Phi_0(x) := e^{-x^2/2}/\pi^{1/4}, \ \Phi_1(x) := \sqrt{2}\,xe^{-x^2/2}/\pi^{1/4} \\ &\Phi_n(x) = \sqrt{\frac{2}{n}}x\Phi_{n-1}(x) - \sqrt{\frac{n-1}{n}}\Phi_{n-2}(x) \quad (x \in \mathbb{R}, n \ge 2) \end{aligned} \tag{1.2}$$

iv) A Φ_n' deriváltak kifejezhetők a Φ_n, Φ_{n-1} függvényekkel:

$$\Phi_n'(x) = \sqrt{2n}\Phi_{n-1}(x) - x\Phi_n(x) \quad (x \in \mathbb{R}, n \in \mathbb{N}, \Phi_{-1} = 0) \tag{1.3}$$

1.3.3. Az approximáció optimalizálása

A jelek reprezentációja függ az időskála 0 pontjának és az egység megválasztásától. Ezeket a paramétereket a gyakorlatban önkényesen szoktuk megválasztani. Ezzel összefüggében felvethető a kérdés, hogyan lehet optimálisan megválaszthatani ezeket a paramétereket. Az approximáció pontossága javítható azonos együttható szám mellett, amennyiben az Hermite-függvények helyett azok

$$\Phi_n^{a,\lambda}(x) := \Phi_n(\lambda x + a) \quad (x, a \in \mathbb{R}, \lambda > 0) \tag{1.4}$$

affin transzformáltjait használjuk. A $\sqrt{\lambda}\Phi_n^{a,\lambda}$ ($n \in \mathbb{N}$) rendszer is ortonormált és teljes az $\mathcal{F}$ téren. Ebben az esetben az f legjobb approximációja az

$$S_n^{a,\lambda} f := \sum_{k=0}^{n} \langle f, \Phi_k^{a,\lambda} \rangle \Phi_k^{a,\lambda} \quad (n \in \mathbb{N}, a \in \mathbb{R}, \lambda > 0) \tag{1.5}$$

projekció és a közelítés hibája az a transzlációs és a λ dilatációs paraméter függvénye:

$$D_n^2(a,\lambda) := \|f\|^2 - \sum_{k=0}^{n} |\langle f, \Phi_k^{a,\lambda} \rangle|^2. \tag{1.6}$$

E két szabad paraméter optimalizálásával azonos együtthatószám mellett, az eredeti Hermite polinomokkal történő approximációhoz képest pontosabb közelítés érhető el. A D_n függvény minimalizálása ekvivalens az

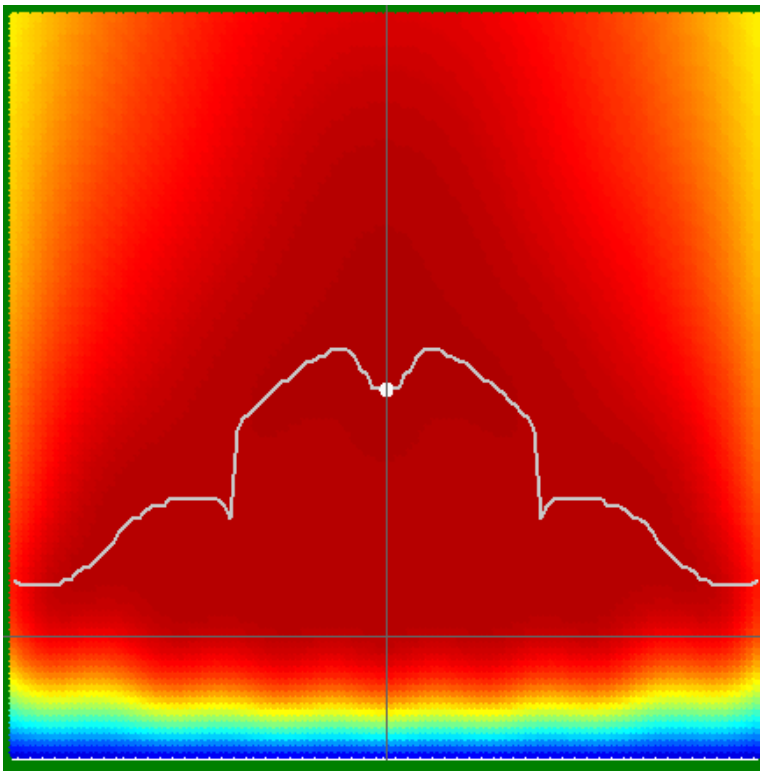
$$F_n(a,\lambda) := \sum_{k=0}^{n} |\langle f, \Phi_k^{a,\lambda} \rangle|^2$$

függvény maximumának meghatározásával. A paraméteres integrálok tulajdonságaiból következik, hogy az

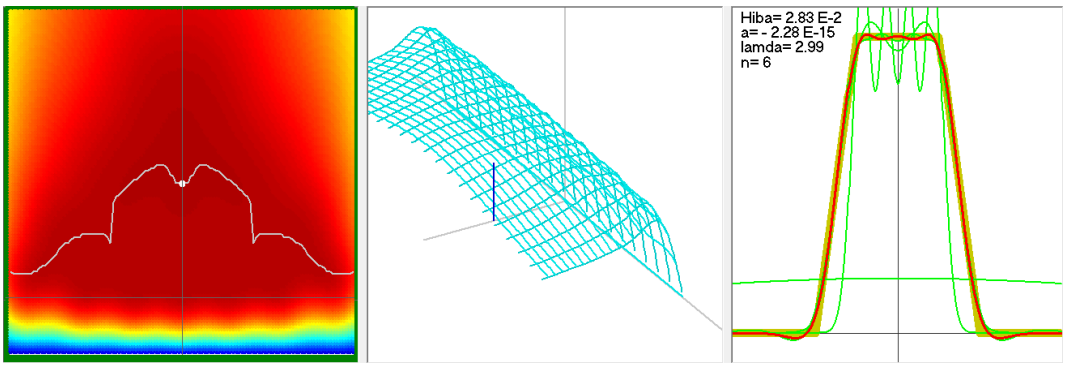
$$A_n(a,\lambda) := \langle f, \Phi_k^{a,\lambda} \rangle \quad ((a,\lambda) \in T := \mathbb{R} \times (0,\infty))$$

Fourier-együtthatók a T tartományon a paraméterek sima függvényei. Bebizonyítható, hogy $\lambda \to 0$ és $|a| + \lambda \to \infty$ esetén $F_k(a,\lambda) \to 0$, következésképpen az F_n függvénynek létezik a maximuma és a D_n függvénynek létezik a minimuma. A részleteket a Függelékben találhatók.

Az alábbi ábrák az $F_n(a,\lambda)$ függvényeket szemléltetik fényintenzitás és perspektivikus ábrázolást használva. A harmadik ablakon a jel közelí tését szemlélteti a maximum helynek megfelelő, ill. egyéb paraméter esetén.



1.2. ábra. Az F_n szí nkódos ábrázolása



1.3. ábra. Az F_n által meghatározott felület, és approximáció

1.4. A Nelder-Mead algoritmus

A $\mathsf{Nelder-Mead}$ szimplex algoritmust [] eredetileg 1965-ben fejlesztették ki azzal céllal, hogy létrehozzanak egy eljárást, amely képes meghatározni egy $f : \mathbb{R}^n \to \mathbb{R}$ nemlineáris függvény minimum (maximum) helyét a gradiens felhasználása nélkül, pusztán a függvényértékekre támszkodva. Mivel az algoritmust a bemutatott tömörítési eljárás során kétváltozós függvényekre kerül aklalmazásra, ezért ebben a speciális

esetben szemléltetjük, megjegyezve, hogy hasonló elvek szerint működik az általános n dimenziós eset is. Említendő továbbá, hogy Fejlesztői dokumentáció fejezetben bemutatott Nelder-Mead algoritmus implementációja képes kezelni az n-dimenziós esetet. A minimum meghatározásához az $f(x_1), f(x_2), f(x_3)$ adnak kiindulási pontot, melyek közül egyet lecserélendő az adott lépésben. Nevezetesen, az algoritmus

$$f(x_3) \leq f(x_2) \leq f(x_1)$$

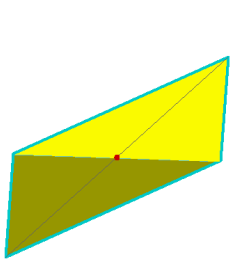
esetén olyan x' helyet keres, amelyre $f(x') \leq f(x_3) \leq f(x_2)$ teljesül és az x_3, x_2, x_1 ponthármasról az x_1-et elhagyva áttérünk az x', x_3, x_2 hármasra. Az x' pontot az előzőekből geometriai transzformációkkal származtatjuk, felhasználva az x_2x_3 szakasz $x = (x_2 + x_3)/2$ felezéspontját:

$$x' = x_1 + \alpha(x - x_1) \quad (\alpha \in \mathbb{R}) .$$

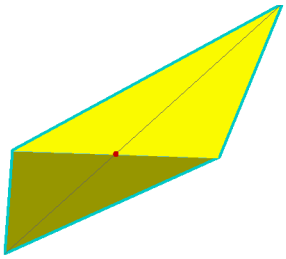
Az alábbi ábrák szemléltetik az algoritmusban használt transzformációkat. Látható, hogy $\alpha = 2$ esetén x' éppen az x_1 pont x-re vonatkozó középpontos tükrözése (T_1). Továbbá $\alpha > 2$ az eredeti háromszög tükrözése + nyújtása (T_2), az $1 < \alpha < 2$ pedig tükrözéses + zsugorításnak felel meg (T_3). A $-1 < \alpha < 0$ paraméterrel egyszerű zsugorítás adódik. Végül az 5. transzformáció (T_5) az x_3 pontból történő kicsinyítésnek feleltethető meg. Az x_1 képe ezekben a transzformációkban és a hozzá tartozó függvényértékek a következő alakban adottak:

$$x' := x_{3+i} = T_i(x_1) \quad (i = 1, 2, 3, 4), \qquad y_j = f(x_j) \quad (j = 1, 2, \cdots, 7) .$$

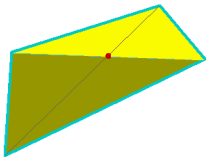
Azt, hogy mikor melyik transzformáció használandó, a Függelékben, illetve a Fejlesztői dokumentáció Nelder-Mead algoritmus implementációjával foglalkozó fejezetben található folyamatábrából látható. Az említett műveleteket a ?? ábra szemlélteti.



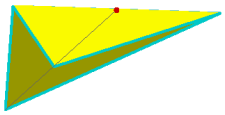
(a) Tükrözés ($T_1 : \alpha = 2$).



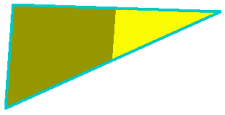
(b) T-Nyújtás ($T_2 : \alpha = 2.5$).



(c) T-Összehúzás ($T_3 : \alpha = 1.5$).



(d) Összehúzás ($T_4 : -1 < \alpha < 0$).



(e) Kicsinyítés x_3-ból (T_5).

1.4. ábra. A $\mathtt{Nelder-Mead}$ szimplex transzformációi.

Megjegyezendő, hogy a $\mathtt{Nelder-Mead}$ szimplex módszer egy determinisztikus algoritmus. Az algoritmus gyors, hiszen minden lépésben csak néhány függvénykiértékelést kell végezni. Továbbá, egyes módszerek, például a gradiens módszerrel ellentétben az olyan patologikus függvények optimumát is képes gyorsan megtalálni, mint amilyen a Rosenbrock függvény.

1.5. Kvadratúra formulák

Mivel a $\mathtt{Nelder-Mead}$ algoritmus kizárólag a függvényértékekre támaszkodik, a hibafüggvény értékeinek kiszámításához elegendő az Hermite-függvényeket valamilyen felosztás pontjaiban egyszer meghatatározni. Ez implementációs szempontból is fontos, hiszen nem kell minden $(\lambda, \mathfrak{a})$ paraméterhez kiszámolni az ettől függő, rekurzióval adott Hermite-féle függvényrendszer tagjait. Ehelyett elég az eredeti diszkrét adatsorozatot dilatálni, illetve eltolni, ami lényegesen kevesebb számítást igényel. Továbbá, az említett felosztás pontjainak valamely Hermite-függvény zérushelyeit választva, a hibafüggvényben szereplő integrálok kiszámításához a [] dolgozatban használt eljáráshoz hasonlóan kvadratúra formulát is alkalmazhatunk. Ennek előnye, hogy az alappontokban a módszer interpolál, így itt lehetséges a jel mintáinak pontos rekonstrukciója. Ugyanakkor, a kapott approximáció is pontosabb, amit a []-ben közölt numerikus tesztek is alátámasztanak.

Ahogyan ez a [] dolgozatban látható, további optimalizációs algoritmusok is alkalmazhatóak a probléma megoldásához. Egy ezek közül a leggyosrabb ereszkedés módszere, melynek az alkalmazásához szükségünk van a függvény parciális deriváltjaira. Ezek előállításával ilyen esetben külön kell foglalkozni. Az Hermite-függvények deriváltjaira vonatkozó formulák alapján a parciális deriváltakra a hibafüggvényhez hasonló ellőállítás adható. Ebben az esetben az (??) egyenletben szereplő integrálok kiszámításához ekvidisztans felosztása alkalmazandó.

1.6. A tömörítő eljárás jellemzése

A tömörítés kezdetekor az első feladat a teljes EKG mérést szívütésekre bontása. Ezt követően az egyes szívütések a tömörítő eljárás bemeneti paramétereként válnak értelmezhetővé. Inicializálni kell továbbá a tömörítő függvények által visszaadott, az aktuális szívütésre vonatkozó Fourier-együtthatókat, az approximáció hibáját, illetve az optimalizációs eljárások által meghatározott dilatációs és transzlációs paramétereket.

A tömörítés előkészítése nem ér véget a jel szívütésekre történő felbontásakor. A szívütéseket normalizálása is szükséges. Ez azt jelenti, hogy az első és utolsó helyen felvett értékeket összekötő egyenes, és az eredeti EKG különbségét tekinti az eljárás tömörítendő jelnek. Ezt az eljárást az irodalomban az alapvonal eliminálásnak nevezik. Ennek eredményeként a jel tartója a kezdő és a végpont által meghatározott intervallum. Végül, a szívütést normálásra kerül, vagyis az egyes értékeket elosztjuk az abszolút maximummal.

A tömörítés megkezdése előtt inicializálni kell a függvényrendszert, melynek során az egyes bázisfüggvények által felvett értékek egy mátrix soraiba kerülnek:

$$\Phi := [\Phi_n(\alpha_m)]_{0\le n<N,\, 0\le m<M} \cdot$$

A $\Phi \in \mathbb{R}^{N\times M}$ mátrix segítségével az $f \in \mathbb{R}^M$ diszkrét jel Fourier-együtthatói könnyen meghatározhatók:

$$c_n := \langle f, \Phi_n\rangle = \frac{1}{M}\Lambda^{-1}\Phi f \quad (0 \le n < N)\,,$$

ahol $\mathbb{R}^{N\times N} \ni \Lambda = \Phi\Phi^T$ Cristoffel-Darboux számokat tartalmazza. Az előállításban szereplő $\alpha_n \in \mathbb{R}\,(0 \le n < N)$ számok a ?? fejezetnek megfelelően kétféleképpen határozhatóak meg. Egyrészt a $\mathtt{Nelder-Mead}$ algoritmusban a Φ_N függvény gyökeit véve kvadratúra formulákat definiáltunk. Más optimalizációs algoritmusok, például a gradiens módszer esetén f tartóján egyenletes alapponterndszert alkalmazandó. Figyelmet érdemel, hogy az előbbi esetben a gyökök pontos meghatározása kritikus a feladat szempontjából. A probléma megoldásához a [?] könyv által javasolt numerikus eljárás található az implementációban. Nevezetesen, a H_n Hermite-polinom (??) rekurziójában szereplő együtthatókat egy tridiagonális mátrixba rendezzük. Könnyen belátható, hogy az α_n gyökök megegyeznek ezen tridiagonális mátrix sajátértékeivel.

Hátra van még a megfelelő (a, λ) paraméterek beállítása. Annak érdekében, hogy a reprezentáció minél adaptívabb legyen, több optimális transzláció, illetve dilatáció meghatározása szükséges. Az eredeti $f \in \mathcal{F}$ jelet tehát a következő alakban közelíti a módszer:

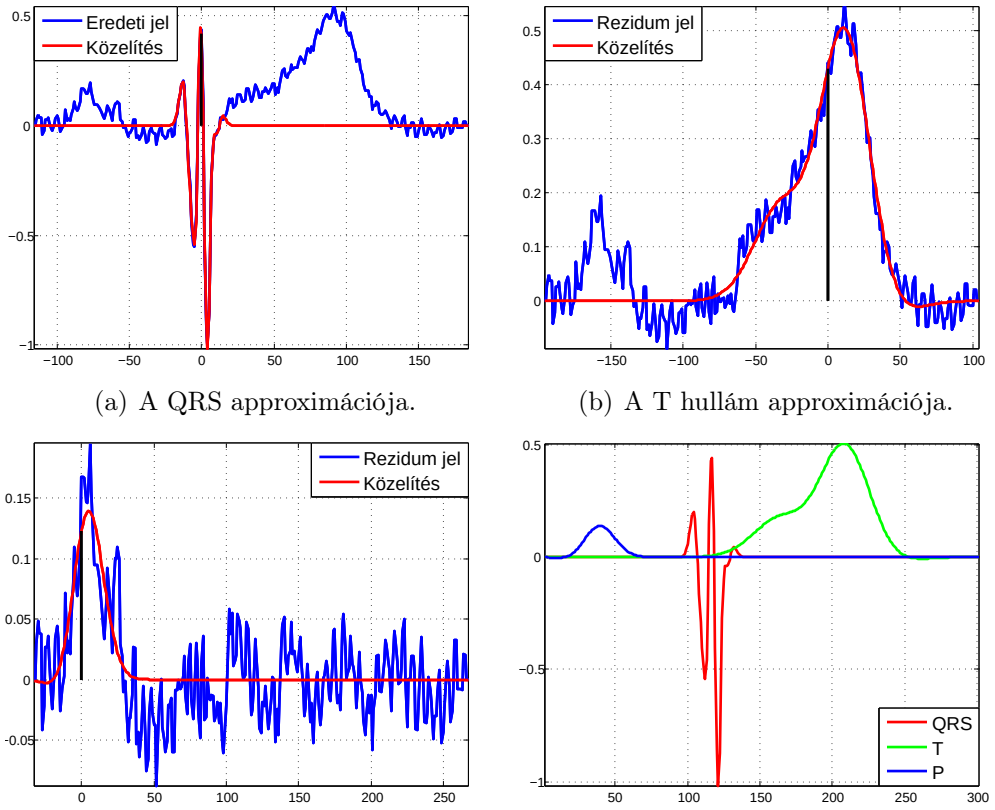
$$S_{\mathbf{n}}^{\mathbf{a},\lambda} f := \sum_{i=1}^{N}\sum_{k=0}^{n_i} \langle f^{a_i,\lambda_i}, \Phi_k\rangle \Phi_k \quad (a_k \in \mathbb{R}, \lambda_k > 0)\,,$$

ahol $\mathbf{a} = a_1, a_2, \ldots, a_N$ az alkalmazott transzlációk, $\boldsymbol{\lambda} = \lambda_1, \lambda_2, \ldots, \lambda_N$ pedig a dilatációk sorozata. Az egyes sorfejtésekhez tartozó együtthatók számát az $\mathbf{n} = n_1, n_2, \ldots, n_N$ vektor jelöli. Mivel az EKG jel alapvetően három fő hullámból áll ezért jelen probléma megoldásakor $N = 3$. Továbbá a [?] dolgozat eredményei alapján a QRS komplexumot egy heted, a T hullámot hatod, a P hullámot pedig egy másodfokú ortogonális rendszer segítségével approximálja az eljárás azaz $\mathbf{n} = 7, 6, 2$. Fontos, hogy a legjobb approximáció előállításához a (??) egyenlettel ellentétben már az eredeti függvény f^{a_i,λ_i} transzformáltja használatos. Ez nem jelent megszorítást az eredeti problémára nézve, implementációs szempontból viszont f^{a_i,λ_i} kiszámítása gyorsabb, mint a $\Phi_n^{a_i,\lambda_i}$ rendszer előállítása.

A (a_i, λ_i) paraméter párok optimalizációja egymástól függetlenül történik. Így azonban nem garantált, hogy az algoritmus a P, QRS, T hullámokat külön-külön approximálja. A problémát az irodalomban jól ismert ún. $\mathtt{Matching\ Pursuit}$ (MP) konstrukció [?] alkalmazásával oldja meg a módszer. Ez egy mohó algoritmus, mely minden lépésben a (??) egyenletben definiált $F_n(a_i, \lambda_i)$ függvény maximalizálására törekszik. Az iteráció i. lépése a következő alakban írható fel:

$$s^{(i)} = s^{(i-1)} + S_{n_i}^{a_i,\lambda_i} R^{(i-1)} \quad (1 \le i \le N)\,, \tag{1.7}$$

ahol $R^{(i)} = f - s^{(i)}$ a rezidum függvényt jelöli. Röviden tehát az $s^{(0)} = 0$, $R^{(0)} = f$ inicializálás után, az eljárás az i. lépésében megkeresi az $R^{(i-1)}$ függvény ℓ^2 norma szerinti legjobb közelítését, amit ki is von az említett rezidum vektorból. Ezt N iteráción keresztül ismétli az aktuális $R^{(i)}$ függvényre. Az MP módszer egy gyors algoritmus, mellyel megkonstruálható az $f \in \mathcal{F}$ jel ritka reprezentációja. Emellett lehetséges az EKG szívütéseinek automatikus szeparációja is, hiszen a jel különböző részeit eltérő ortogonális rendszerek segítségével közelítjük. Figyelmet érdemel, hogy az eredeti [?] módszerben ezt a lépést egy külön szegmentáló algoritmus végezte. Így a közelítés pontossága erősen függött a szegmentálás eredményétől (lsd. ?? fejezet). A kifejlesztett eljárásnál azonban ez a probléma nem áll fenn még zajos jelek esetén sem. A dolgozatban bemutatott módszer iterációs lépéseit, az $R^{(i)}$ rezidum függvények alakulását, illetve a szeparált EKG jelet az ?? ábra szemlélteti. Jól látható az is, hogy a fekete vonallal jelölt optimális transzláció általában nem az abszolút maximum helyén található. Ez indokolja, hogy a λ_i dilatáció mellett az a_i paraméter optimalizációjára is szükség van. Mivel ez utóbbi a kiindulásként használt [?] dolgozatból hiányzik, ezért a pontosság és tömörítési arány jelentős javulása várható el tőle.



(a) A QRS approximációja.

(b) A T hullám approximációja.

(c) A P hullám approximációja.

(d) Szeparált szívütés.

1.5. ábra. Az MP algoritmus lépései.

2. fejezet

Felhasználói dokumentáció

3. fejezet

Fejlesztői dokumentáció

Ebben a fejezetben található a Bevezetés-ben bemutatott eljárás implementációjának pontos (modul szintű) leírása, valamint az alkalmazott tesztek jellemzése illetve ezek eredményei. A tesztelés eredményeit a tapasztalok összegzése és néhány lehetséges jövőbeli fejlesztés jellemzése követi.

A fejlesztői dokumentáció első részében a tömörítési módszer implementációjának leírása található. Ez az alfejezet két fő részre tagolható: a webes felhasználói felület és az ehhez felhasznált könyvtárakat leíró, valamint a háttérben a konkrét tömörítést elvégző c++ program modulonkénti jellemzését. Mivel a webes felület megvalósítása rövid, és specializált scriptek összessége, ezért ezek jellemzését minden esetben egyedi módon érdemes kezelni. Ellentétben ezzel a szemlélettel, a c++ program nagyban épít a nyelv által támogatott objektum orientált megközelítésre. Ez lehetővé teszi és indokolja is egyben, hogy az implementációt jellemző dokumentáció is hasonlóan jól strukturált legyen. Az egyes modulok dokumentációja ugyanazon három szempont szerint közelíti meg a jellemzésüket: a modul feladatának, a modul interfészének (hogyan és mely egyéb modulokhoz kapcsolódik), valamint a modult felépítő osztályok implementációjának pontos megfogalmazása. A modulok leírásánál találhatóak továbbá a szemléltetést segítő UML és egyéb hasznos diagrammok.

A teszteléssel foglalkozó alfejezet szintén több részre tagolható. Az első alfelejezetben található a webes felhasználói felület funkcióinak tesztelési terve, illetve az egyes tesztek eredményeinek leírása. Ezt követi a c++ program moduljainak részletes tesztelési terve. A modulok tesztelését minden esetben egy külön tesztfájl végezte. Ennek leírása, az eredmények értékelése, illetve a tesztállomány elérési útja szintén megtalálható az alfejezetben. A tesztelés alfejezet utolsó része a bemutatott tömörítési eljárást hivatott tesztelni. Ez több szempont alapján, egyéb EKG tömörítő módszerekkel történő összehasonlítás útján valósul meg. A tesztek eredményeinek jellemzését az egyes elfajuló, különleges esetek közelebbi vizsgálata, majd egy rövid kitekintés követi.

3.1. A felhasználói felület implementációja

3.2. A tömörítő eljárás implementációja

A felhasználói dokumentáció bevezetésében leírtakkal összhangban, a tömörítő eljárás implementációjával foglalkozó fejezet minden pontja három részre tagolható. Minden alfejezet az aktuális modul feladatának pontos megfogalmazásával kezdődik. Ezt követi a modul interfészének definiálása, melynek segítségével átlátható a modul kapcsolata a program többi részével. Végül a modult alkotó osztályok és strukturák fő metódusainak, illetve adattagjainak leírása következik. A program robosztussága indokolja, hogy minden egyes metódus és függvény ne kerüljön itt jellemzésre, azonban a program forráskódjában minden metódus előtt megtalálható annak rövid jellemzése, elvárt bemenete, és esetleges visszaadott értékének típusa. Az egyes modulok dokumentációja tartalmazza ezen kívűl az ezeket leíró UML, és egyéb diagrammokat. Az egyes modulok imlpementációja külön .h, és .cpp kiterjesztésű állományokban található, ezért az egyes alfejezetek címei megegyeznek ezen állományok elnevezésével. A fejezet utolsó alfejezete leírást biztosít az implementáció által felhasznált külső könyvtárakhoz, illetve indokolja ezek választását.

3.2.1. A SigPrep modul implementációja

Az implementációt tartalmazó repository gyökerét /-vel jelölve, a SigPrep modult az /src/headers/SigPrep.h, illetve az /src/SigPrep.cpp állományok implementálják.

A modul feladata, hogy absztrakt módszert biztosítson a bemeneti jelek kezeléséhez. Mivel az eljárás EKG jelek tömörítésével foglalkozik, illetve speciálisan az MIT-BIH adatbázisban megtalálható egyedi formátumú jelek feldolgozásával, ezért egy általános jel kezelő modul implementációja nem volt szükséges a módszer megvalósításához. Annak érdekében azonban, hogy hogy a bemutatott jel tömörítő eljárást jövőbeli vizsgálódások során könnyen ki lehessen próbálni egyéb jel típusokon, fontos hogy az ehhez szükséges metódusok absztrakt módon legyenek definiálva. A SigPrep modul feladata tehát, hogy definiálja azokat az egységes metódusokat amelyek elengedhetetlenek a bemeneti jelek programon belül történő kezeléséhez. Ezen kezelési módszerek pontos megvalósítását a /src/headers/SigPrep.h állományban található SigPrep osztályból származtatatott osztályok írják le.

3.2.2. Az EcgSigPrep modul implementációja

3.2.3. Az OrtFunSys modul implementációja

Az implementációt tartalmazó repository gyökerét /-vel jelölve, az OrtFunSys modult az /src/headers/OrtFunSys.h, illetve az /src/OrtFunSys.cpp állományok implementálják.

A modul feladata, hogy definiálja azokat a konténereket illetve absztrakt metódusokat amelyek segítségével megadható egy a program által felhasználható függvényrendszer. A Bevezetés fejezetben leírt eljárás speciálisan az ott bemutatott Hermite függvényrendszert alkalmazza az EKG jelek tömörítéséhez. A bemutatott algoritmusokat azonban további, különböző tulajdonságú ortogonális (vagy akár csak lineárisan független) függvényrendszerek segítségével is érdemes lehet kipróbálni. Annak érdekében, hogy ez könnyedén implementálható legyen, fontos hogy az egyes függvényrendszereket kezelő metódusok absztrak módon legyenek definiálva. Az OrtFunSys modul feladata tehát, hogy megadja azokat az adatszerkezeteket és absztrakt metódusokat, amelyek segítségével a program képes függvényrendszerek kezelésére. Az egyes függvényrendszerek, példaul az Hermite függvények pontos implementációja a /src/headers/OrtFunSys.h állományban definiált OrtFunSys osztályból származtatatott osztályokban található.

A modul az absztrakt OrtFunSys osztályból áll, amely tartalmaz úgynevezett pure virtual (tisztán virtuális) függvényeket. Ennek következtében az osztály által definiált típusból objektumokat példányosítani nem lehet. A példányosítható objektumok hiányában az OrtFunSys modulhoz nem határozható meg konkrét interfész, melynek segítségével az OrtFunSys típusú objektumok szolgáltatásokat nyújthatnának a program többi részének. Megjegyezendő azonban, hogy az OrtFunSys osztályból származtatott osztályok objektumainak szolgáltatásait az OrtCompresser modul használja fel.

A modult alkotó OrtFunSys osztály definíciója a /src/headers/OrtFunSys.h állományban található. A bemutatott tömörítő eljárás egyik legköltségesebb művelete a felhasznált függvényrendszer értékeinek kiszámítása. Mivel a függvényértékek meghatározása bármely függvényrendszer esetében szükséges, ezért a kiszámított értékek tárolásának logikáját illetve a felhasznált adatszerkezeteket az OrtFunSys osztály határozza meg. Egyes függvényrendszerek (például az Hermite függvényrendszer) tulajdonságai lehetővé teszik, hogy a rendszert jellemző paraméterek (Hermite rendszer esetében a dilatáció illetve a transzláció) közvetlenül a jelben történő változtatását. Ilyen esetben a tömörítő eljárás során elegendő a felhasznált függvényrendszer értékeinek egyszeri meghatározása, hiszen a közelítést optimalizáló algoritmusok a rendszer paramétereit közvetlenül a bemeneti jelre alkalmazhatják. Ennek következményeként az OrtFunSys osztály a függvényrendszer értékeit a heap-en, dinamikus memóriában tárolja.

Az osztály négy protected láthatóságú adattaggal rendelkezik.

4. fejezet

Függelék