Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék

Spektrumbecslés adatvesztés esetén

Diplomaterv

Készítette
Palkó András

Konzulens
Dr. Sujbert László

2019. december 19.

Tartalomjegyzék

HALLGATÓI NYILATKOZAT

Alulírott *Palkó András*, szigorló hallgató kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy autentikált felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Budapest, 2019. december 19.

Palkó András
hallgató

Kivonat

A hagyományos kommunikációs, mérési módszerek mellett napjainkban a szenzorhálózatok, az IoT megjelenésével kezdenek terjedni a kevésbé megbízható, ám olcsóbb kommunikációs protokollok, mérőeszközök. Ezekben a rendszerekben előfordulhat adatvesztés, azaz, hogy néhány minta hibás legyen.

A villamosmérnöki gyakorlat egyik alapvető eszköze a spektrumanalízis, melyet hagyományosan a DFT-vel végzünk el. Adatvesztés esetén a DFT nem számítható ki pontosan, továbbá jellemzően nem várhatjuk meg, hogy összegyűljön egy adatvesztés nélküli DFT-blokk.

A dolgozat áttekinti az adatvesztés jelenségét és két különböző megközelítését, ezek az indikátorfüggvény alkalmazása és a nem egyenletes mintavétel. Ismertetésre kerülnek az adatvesztés alapvető mennyiségei, függvényei.

A dolgozat ismertet három alapvető megközelítési módot az adatvesztéssel terhelt jelek spektrumának becslésére. Az első irány a DFT használata némi kiegészítő számítással együtt. A második irány a jel modellezése és ez alapján egy becslő kialakítása. A harmadik irány az adatvesztéssel terhelt jelet nem egyenletesen mintavételezett jelként kezeli, és a nem egyenletesen mintavételezett jelek spektrumának becslésére kifejlesztett eljárásokat tartalmazza. A dolgozat újdonságként mutatja be a vezérelt adaptív Fourier-analizátor, a harmonikus DCDFT, CLEANEST és SLICK eljárásokat.

Az egyes ismertetett eljárások elméleti és szimulációs összehasonlításra kerülnek. A dolgozat elméleti oldalról vizsgálja az egyes eljárások számítás- és memóriaigényét, az általuk használt frekvenciarácsot, az inkoherens mintavétel kezelését, a potenciális numerikus problémákat, valamint a valós idejű megvalósíthatóságot.

Az ismertetett eljárások szimulációs összehasonlítása azzal a kérdésfeltevéssel történt, hogy az adatvesztés mennyiben befolyásolja az eljárások által adott becslőket, illetve azzal is, hogy a becslőből az eredeti jel komponensei milyen pontosan azonosíthatók. A dolgozatban ismertetett szimulációs összehasonlítás mindkét kérdéssel foglalkozik. Az eredmények alapján javaslatokat teszek adott mérési helyzetekben az alkalmazandó eljárásra.

Abstract

Nowadays with the emergence of the IoT, less reliable and cheaper communication protocols, measurement devices are appearing besides the traditional communication and measurement methods. In these systems data loss is a possibility, this means that some of the samples may be incorrect.

Spectral analysis is one of the common tools in electrical engineering. Traditionally, DFT is used for this calculation, but it cannot be evaluated precisely when data loss occurs. Usually, is is not possible to wait for a full DFT block.

This paper reviews the phenomenon of data loss and two modeling approaches: the application of the indicator function and the non-equidistant sampling. The basic properties, functions of the data loss are described.

Three different approaches are presented for the spectral estimation of a signal with missing samples. The first one is the usage of DFT with some additional calculations. The second one is to model the signal and create an estimator based on this model. The third one is to view the signal as a non-equidistantly sampled one and use a method developed for the spectral estimation of the non-equidistantly sampled signals. The paper describes new methods, namely the controlled AFA, the harmonic DCDFT, CLEANEST and SLICK.

The above mentioned methods are analyzed from a theoretical point of view and tested by simulations. The thesis investigates the methods from the following aspects: computational complexity, required memory, used frequency grid, handling of the incoherent sampling, potential numerical problems and the possibility of a real-time implementation.

Simulations have been conducted to answer two questions. The first one is, how does the data loss affects the estimator calculated by the different methods. The second question is, how precisely the estimator identify the components of the original signal. The paper deals with both approaches. Recommendations are given for the choice of the spectral estimation method in different measurement circumstances.

1. fejezet

Bevezetés

Napjainkban egyre több eszköz lesz okos: telefon, tévé, hűtő, mosógép, és még folytathatnánk. Ezeket ráadásul az interneten keresztül össze is kapcsoljuk egymással, így alakul ki az IoT, avagy az Internet of Things. Ilyen módon kialakulóban van egy hatalmas szenzorhálózat, amellyel a fizikai világban mérhetünk, illetve abba beavatkozhatunk, távolról.

A hagyományos kommunikációs, mérési módszerek gyors, pontos és megbízható adatátvitelt tesznek lehetővé. A szenzorhálózatokban a mobilitási, egyszerűségi, olcsósági és energiahatékonysági követelmények miatt kevésbé megbízható kommunikációt alkalmaznak. Ez azt jelentheti, hogy például néhány mérési adat sérül az átvitel során, akár csak részlegesen is. Kezdetben csupán adattovábbításról beszélhettünk egy szenzorhálózatban, manapság már a jeltovábbítás is egy elterjedt alkalmazás, ez indokolja a vizsgálatot jelfeldolgozási szempontból.

Az adatvesztés tekinthető egy mérési, kommunikációs hibának, melyet a nem megbízható eszközök, eljárások okoznak. Interferencia gátolhatja egy rádiójel helyes vételét, csomagok veszhetnek el internetes kapcsolatokon vagy akár külső körülmények akadályozhatják a mérést. Egy elosztott rendszerben az elosztott órajel miatt fellépő szinkronizációs hibák is okozhatják adatok kiesését vagy megduplázódását.

A villamosmérnöki eszköztár egyik alapvető eleme a frekvenciatartománybeli analízis, melyre gyakran alkalmazzuk a diszkrét Fourier-transzformációt (DFT-t):

$$X_k = \sum_{n=0}^{N-1} x_n \mathrm{e}^{\mathrm{j}\frac{2\pi}{N}kn} = \mathrm{DFT}\left(x_n\right), \tag{1.1}$$

ahol x_n az időtartománybeli DFT-blokk n. eleme, X_k a DFT k. pontjának értéke, ez a spektrum becslője, N a DFT-blokk hossza, j pedig a képzetes egység. Látható, hogy a DFT minden pontjának kiszámításához szükséges a blokk összes mintája, ezért amennyiben néhány minta elvész, a DFT-t már csak hibával tudjuk kiszámítani. Kézenfekvő az ötlet, hogy várjuk meg, amíg egy teljes DFT-blokk összegyűlik, de az ehhez szükséges minták száma már gyenge adatvesztés esetén is elfogadhatatlanul nagynak adódhat.

Az adatvesztéssel terhelt jelek spektrumának becslését több irányból is megközelíthetjük, így beszélhetünk a módszerek három típusáról. Az első típusba a DFT-alapú módszerek tartoznak, melyeknél a DFT-t használjuk a spektrum kiszámítására, a jel időtartománybeli előfeldolgozása után. A második típus a modellalapú módszereké, melyeknél modellezzük a jelet, és ez alapján próbálunk spektrumot becsülni. A harmadik típus azt használja ki, hogy az adatvesztés felfogható a nem egyenletes mintavétel speciális eseteként, így a spektrumszámításra alkalmazhatók a nem egyenletesen mintavételezett jelekre kidolgozott eljárások.

A 2. fejezetben először körüljárjuk az adatvesztés jelenségét. Megismerjük matematikai leírását az indikátorfüggvénnyel, az adatvesztés alapvető mennyiségeit és függvényeit, valamint néhány adatvesztési modellt. A fejezetet az adatvesztés és a nem egyenletes mintavétel kapcsolatával zárjuk.

A 3. fejezet tartalmazza a vizsgált eljárások áttekintését, rövid leírásukat. Megismerjük a DFT-alapú eljárások két megközelítését az előfeldolgozás ideje szerint. Bemutatásra kerül a vezérelt rezonátoros struktúra származtatása, amelyhez hasonló gondolatmenettel vezetjük fel a vezérelt adaptív Fourier-analizátort. Ezután körüljárjuk a nem egyenletes mintavételre kifejlesztett eljárásokat, melyeket a projektív szemlélet szerint fogunk vizsgálni. Végül újdonságként kerülnek ismertetésre a harmonikus DCDFT, CLEANEST és SLICK eljárások.

A 4. fejezetben a bemutatott eljárásokat elméleti oldalról hasonlítjuk össze. Megvizsgáljuk az eljárások számítás- és memóriaigényét. Szót ejtünk a vizsgált frekvenciákról, az inkoherens mintavétel kezeléséről, valamint potenciális numerikus problémákról az implementáció során. Az említett szempontok mind befolyásolják az eljárások valós idejű megvalósításának lehetőségét, így összefoglalásként ezt a szempontot vesszük szemügyre.

Az 5. fejezet az eljárások szimulációs vizsgálatát tárgyalja. Az első részben megismerjük a szimulációs környezetet, részletezzük a vizsgált jeleket, adatvesztési modelleket és az eljárások paraméterezését. Bemutatásra kerül két különböző hibakritérium, melyek alapján az eljárások pontossága értékelésre kerül. A fejezetet a szimulációs eredmények áttekintésével folytatjuk, majd eljáráskiválasztási javaslatok ismertetésével zárjuk.

A 6. fejezet röviden összefoglalja a dolgozatot, valamint értekezik a továbbfejlesztési lehetőségekről. Végül az F.1. függelék a főszövegben nem megjelenített szimulációs eredmények ábráit tartalmazza.

2. fejezet

Az adatvesztés

2.1. Az adatvesztés jelensége

Az adatvesztés sokoldalú jelenség. Általánosan elmondható, hogy ez egy kommunikációs, mérési hiba, melyet a nem megbízható eljárásaink, csatornáink, protokolljaink okoznak. Jellemzően véletlenszerű, de előfordul determinisztikus változata is. A feldolgozó algoritmus is dönthet úgy, hogy egyes adatokat eldob, mivel például a kezelésük már túlterhelné a rendszert. A decimálás egy speciális, tervezett adatvesztés.

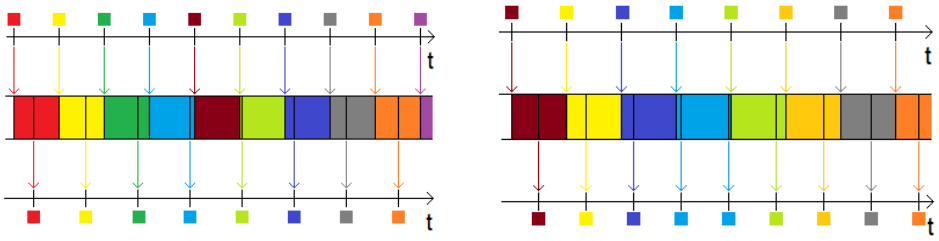
Általánosságban elmondható, hogy annak ellenére, hogy mintavételezésről és mintákról beszélünk, ez fizikailag nem feltétlenül takar egy AD-átalakítót. Talán az a legcélszerűbb, ha úgy gondolunk a mintákra, mint számok sorozatára, és az adatvesztés ebben a sorozatban bekövetkező hibákat ír le. Viszont mivel a feldolgozó eljárásainkat digitális rendszerekkel valósítjuk meg, az adatvesztést a dolgozatban diszkrét időben vizsgáljuk.

Három alapvető adatvesztési módról beszélhetünk. Az első az érvénytelen minták esete: az adott mintavételi pontban jön adat, de tudjuk, hogy az hibás, vagyis a számsorozat egy eleme érvénytelen. Egy tipikus példa egy túlvezérelt AD-átalakító: valamit kiad a kimenetén, de jelzi, hogy hiba történt, ez nem a helyes érték. Másik példa lehet egy rádiós üzenet, amit interferencia miatt hibásan veszünk, és az ellenőrző összegből kiderül, hogy az üzenet sérült. Ilyen interferenciát okozó hatás lehet például egy villámcsapás, zavarhatják egymást a szenzorhálózat egyes elemei, vagy lehet valamilyen másik kommunikáló eszköz is a közelben.

Az adatvesztés második módja a hiányzó minták esete: az adott mintavételi pontban egyáltalán nem jön adat, vagyis a számsorozatból egy elem kimarad (és erről tudomást szerzünk). Erre a jellemző példa egy csomag elvesztése az interneten. UDP kapcsolat esetén ez például úgy vehető észre, ha a csomagokat valamilyen egyedi azonosítóval látjuk el – például egy csomagban küldünk el egy adott időponthoz tartozó mérési eredményeket az időponttal együtt, és tudjuk, hogy milyen gyakran történik meg a mérés. Hiányzó mintának tekinthető az is, amikor valami időnként akadályozza a mérést. Lehet ez egy csillagászati mérés, amit minden nap éjfélkor kellene elvégezni, de időnként az időjárás nem teszi ezt lehetővé; vagy akár azt is modellezhetjük így, amikor valaki influenzásan óránként megméri a lázát, kivéve alvás közben.

Az adatvesztés harmadik módját alkotják a szinkronizációs problémák. Ha egy elosztott rendszerről beszélünk, az órajel is elosztott, azaz minden komponensnek saját órajele van. Hiába azonos az órajelek névleges frekvenciája, a gyakorlatban némi eltérés lesz közöttük a kvarckristályok különbözősége, illetve az eltérő környezeti feltételek miatt. Így órajeltartományok között kell adatokat átvinnünk, amit a legegyszerűbben úgy modellezhetünk, hogy felveszünk egy egy minta kapacitású tárolót a határra, amelybe az egyik oldalról írunk, és az aktuálisan bent lévő adatot a másik oldalról kiolvassuk. Így ha a

beírás gyorsabb, mint a kiolvasás, minták eshetnek ki (2.1.a. ábra), míg fordított esetben ugyanazt kétszer kiolvasva adatok duplázódhatnak meg (2.1.b. ábra).



(a) Minta kiesése **(b)** Minta duplázódása

2.1. ábra. Szinkronizációs problémák

2.2. Alapvető kezelési lehetőségek

A rendszereink tervezésekor készülhetünk az adatvesztésre: implementálhatunk különböző hibajavító kódolásokat, hibatűrő átviteli protokollokat, az elveszett mintákat újraküldhetjük, a hibás méréseket elvégezhetjük újra. Ezek mind lehetnek járható utak, de a nagyobb megbízhatóság nagyobb számítás-, idő-, energia-, és/vagy sávszélességigénnyel jár együtt. Ezek miatt adott hardver mellett a rendszerünk teljesítménye csökken, illetve ugyanazt a teljesítményt csak egy drágább eszközzel tudjuk elérni.

Továbbá, amikor egy valós idejű rendszerről beszélünk, akkor véges időn belül nem tudjuk garantálni az üzenet megérkezését, így mindig fel kell készülni az adatvesztésre. A megbízható átvitel megnövelheti a kommunikáció késleltetését, ami szabályozási körökben nemkívánatos, mivel instabilitáshoz vezethet.

Ezen kívül, amennyiben egy jel pillanatértékét mérjük, akkor is előfordulhat, hogy a mérés sikertelen, ilyenkor ez az információ nem pótolható. Ekkor az alkalmazott kommunikációtól függetlenül szembesülünk az adatvesztéssel.

A blokkos feldolgozó eljárások esetén – például DFT – természetes ötlet, hogy megvárjuk, amíg egy teljes blokknyi minta összegyűlik, és ekkor végezzük el a számítást. Adott esetben ez is tökéletesen működőképes lehet, de mivel az ilyen blokkok mérete akár több ezer minta is lehet, már gyenge adatvesztés esetén is előfordulhat, hogy elfogadhatatlan mérési időre vezet egy ilyen megoldás. Ráadásul jellemzően nem csak egy mérést végzünk, hanem a mérési eredményeket szeretnénk átlagolni, és ekkor az előző hatás még erősebben jelentkezik.

2.3. Leírás az indikátorfüggvénnyel

Az adatvesztés behatóbb vizsgálatához szükségünk van a modellezésére. Ez a modellezés megtehető egy K_n úgynevezett indikátorfüggvénnyel, mely minden ütemre megmutatja, hogy az adott minta elérhető-e vagy elveszett:

$$K_n = \begin{cases} 1 & \text{ha az adat elérhető az } n. \text{ ütemben} \\ 0 & \text{ha az adat elveszett az } n. \text{ ütemben} \end{cases}. \tag{2.1}$$

Vagyis beszélhetünk elérhető, illetve elveszett mintákról. Egymás utáni mintákra sorozatként, fix hosszúságú sorozatra blokként fogunk hivatkozni. Csak elérhető mintákat tar-

talmazó sorozat (blokk) a teljes sorozat (blokk), és hasonlóan csupán elveszett mintákat tartalmaz az üres sorozat, illetve blokk.

Két természetesen felmerülő mutató a γ adatvesztési, illetve a μ adatelérhetőségi arány, melyek rendre az elveszett és az elérhető minták valószínűségeit mutatják meg:

$$\gamma = \mathbb{P}\left(K_n = 0\right), \tag{2.2}$$

$$\mu = \mathbb{P}\left(K_n = 1\right) = 1 - \gamma, \tag{2.3}$$

ahol $\mathbb{P}\left(\cdot\right)$ a valószínűségi operátor. Az adatvesztés időbeli eloszlását jellemzi az $R\left(L\right)$ megbízhatósági, illetve az $R'\left(L\right)$ komplementer megbízhatósági függvény, mely megmutatja egy L minta hosszú teljes, illetve üres sorozat valószínűségét:

$$R\left(L\right) = \mathbb{P}\left(\forall i \in \{1, \ldots, L\} : K_i = 1\right), \tag{2.4}$$

$$R'\left(L\right) = \mathbb{P}\left(\forall i \in \{1, \ldots, L\} : K_i = 0\right). \tag{2.5}$$

Amíg az adatvesztési arány egy magától értetődő mutató, addig a megbízhatósági függvények némi magyarázatra szorulnak. Például a megbízhatósági függvény mutatja meg, hogy egy adott méretű blokk milyen valószínűséggel lesz teljes, azaz mi az esélye, hogy a klasszikus DFT-vel hibátlanul fel lehessen dolgozni. Két különböző adatvesztés esetén a megbízhatósági függvények akkor is lehetnek különbözők, ha az adatvesztési arányok azonosak. Például ha minden minta a többitől függetlenül veszik el γ valószínűséggel, vagy a minták tízesével vesznek el γ valószínűséggel, akkor az adatvesztési arányok nyilvánvalóan azonosak lesznek. Az utóbbi esetben azonban valószínűbb, hogy egy adott hosszúságú sorozat teljes legyen, mint az előbbi esetben.

Ezzel a megközelítéssel úgy modellezhetünk egy adatvesztéssel terhelt jelet, hogy az eredeti $x_{\mathrm{e},n}$ jelet minden ütemben megszorozzuk az indikátorfüggvény adott üteméhez tartozó értékével:

$$x_n = K_n x_{\mathrm{e},n}. \tag{2.6}$$

Mivel időtartományban szorzással modelleztük az adatvesztést, az a frekvenciatartományban egy konvolúciónak felel meg:

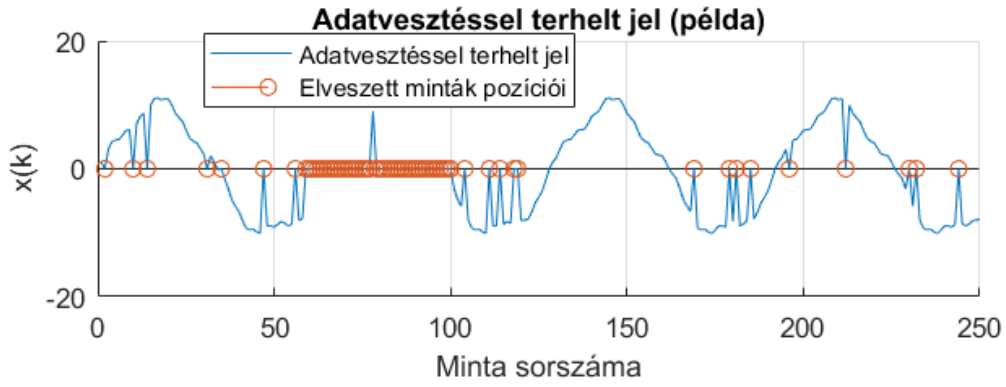
$$X\left(k\right) = \mathrm{DFT}\left(K_n\right) * \mathrm{DFT}\left(x_{\mathrm{e},n}\right), \tag{2.7}$$

ahol $*$ a konvolúciós operátor.

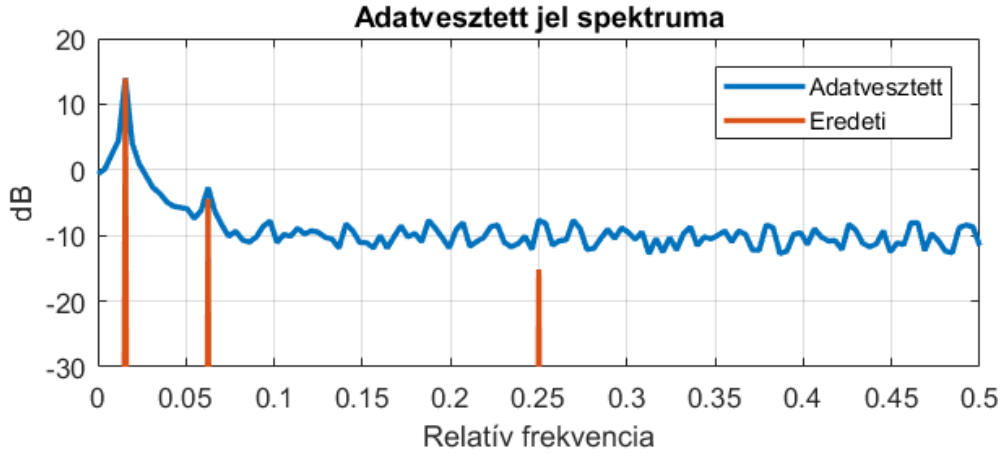
A 2.2.a. ábrán egy adatvesztéssel terhelt jelet láthatunk. A jel három szinuszos komponens összege, és a jel néhány mintája elveszett. Az elveszett mintákat kis piros körök jelzik, és a (2.6) egyenlettel összhangban az elveszett minták helyén a jel nulla. A 2.2.b. ábra mutatja az eredeti és az adatvesztéssel terhelt jel spektrumát. A vízszintes tengelyen a frekvenciát relatív frekvenciában ábrázoltam, azaz az $\frac{f}{f_s}$ arány látható, ahol f_s a mintavételi frekvencia. Megfigyelhető, hogy eredetileg három szinuszos komponens összegéről van szó, de adatvesztés esetén fellép egy szivárgás jellegű jelenség. Ez nem az inkoherens mintavételből adódó szivárgás, nem is zaj, ez az indikátorfüggvény spektrumának megjelenése a jel összes komponensén. Megfigyelhető, hogy ez a hatás a gyengébb komponensek precíz mérését, illetve detektálását megnehezíti, esetenként lehetetlenné teszi.

2.4. Adatvesztési modellek

Miután megismertük az indikátorfüggvénnyel való leírás módszerét, természetesen vetődik fel a kérdés, hogy honnan is szerezzünk indikátorfüggvényt? Az első ötlet az lehet, hogy megpróbáljuk megmérni: végzünk egy mérési sorozatot és vizsgáljuk, mikor kaptunk érvénytelen adatokat; megvizsgálunk egy kommunikációs csatornát, hogy mikor sikeres az



(a) Időtartomány



(b) Frekvenciatartomány

2.2. ábra. Példa adatvesztéssel terhelt jelre és a spektrumára

átvitel stb. Nyilvánvalóan következik ebből az igény, hogy identifikáljuk ezeket a helyzeteket, azaz adjunk egy modellt, hogy mégis milyen az adott környezetben az adatvesztés. Ez az első motiváció az adatvesztési modellek bevezetésére.

A második motiváció onnan származik, hogy szeretnénk tudni, mennyire sikerült robusztus eljárást alkotnunk: milyen adatvesztést visel el? Így a szimulációs vizsgálatokhoz rengeteg különféle adatvesztési modellt „kitalálhatunk”, melyek közül a tervező kedvére válogathat. A két irány egy tervezési feladat során összeérhet: identifikáljuk az adatvesztési folyamatot, majd az így kapott modellt használjuk fel arra, hogy a felhasználói igényeket kielégítő algoritmust, protokollt válasszunk, fejlesszünk ki.

Az adatvesztési modellek sokféleségét illusztrálandó, szeretném itt is bemutatni a 2018-as TDK-dolgozatomban [1] ismertetett áttekintést.

Véletlen független modell

A véletlen, független adatvesztés a legegyszerűbb modell. Minden minta a többitől függetlenül, azonos γ valószínűséggel veszik el.

Ilyen indikátorfüggvényre látunk példát a 2.3. ábrán. A látszat ellenére az ábra egydimenziós, hibás az ábra alapján indikátormátrixra következtetni. Az ábra a következő gondolatmenettel származtatható:

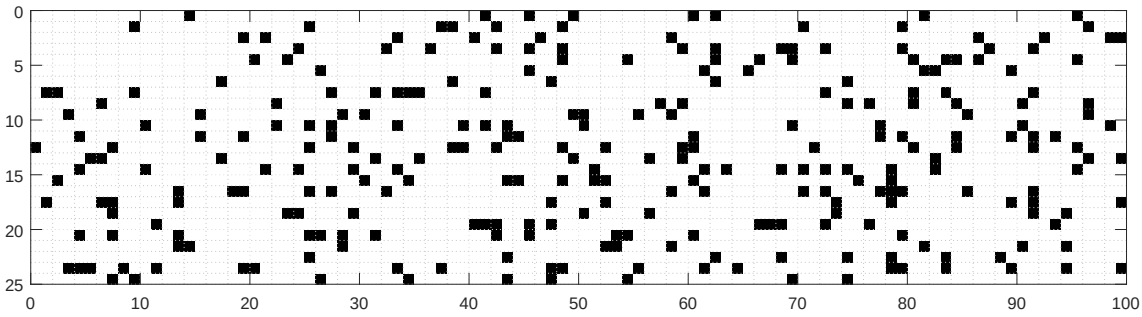
1. Generáljunk egy 2500 mintás sorozathoz tartozó indikátorfüggvényt.

2. A generált értékeket szemléltessük egy számegyenes mellett: amelyik indexnél a minta elérhető, oda fehér, ahol elveszett, oda rajzoljuk fekete négyzetet. Így kapunk egy „négyzetekből álló sorvektort”.

3. Az előbb kapott vektor túlságosan hosszú ahhoz, hogy változatlan formában ábrázoljuk, ha minden részletét szeretnénk bemutatni. Ezért a vektort daraboljuk fel 100 mintás blokkokra, és ezeket a blokkokat illesszük egymás alá. Így megkapjuk az indikátorfüggvényt szemléltető ábrát.

Vagyis az ábrát úgy kell értelmezni, mintha olvasnánk: balról jobbra, majd fentről lefelé; az egyes karaktereknek pedig a fekete és a fehér négyzetek felelnek meg. Mindennek megfelelően a vízszintes tengelyen a 100-as blokkon belüli pozíció, a függőleges tengelyen pedig a 100-as blokk sorszáma látható.

Látható, hogy az adatvesztés véletlenszerűen történik, időnként előfordulnak egymás utáni elveszett minták, de a hosszú üres sorozatok ritkák.

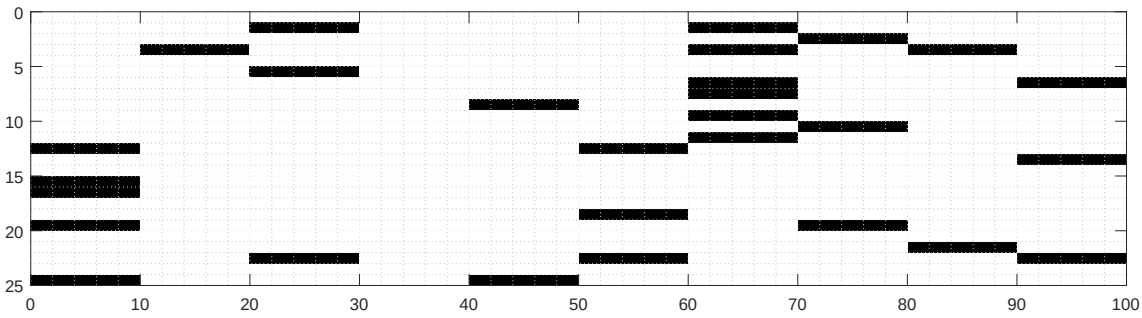


2.3. ábra. Példa a véletlen független adatvesztésre (értelmezés: soronként, balról jobbra)

Blokkos, független modell

Blokkos adatvesztés esetén a minták fix méretű blokkokban elérhetők vagy vesznek el. Blokkos, független adatvesztés esetén az egyes blokkok egymástól függetlenül, azonos γ valószínűséggel vesznek el.

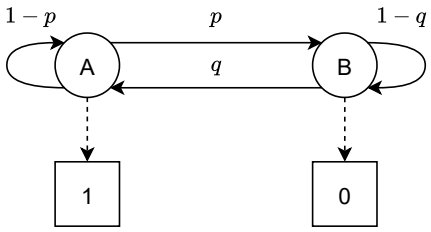
A blokkos független adatvesztésre mutat példát a 2.4. ábra. A blokkméret jól azonosíthatóan 10 minta, az egyes blokkok elérhetőségéről pedig ugyanaz mondható el, mint a véletlen független modell esetén a mintákéról.



2.4. ábra. Példa a blokkos független adatvesztésre (értelmezés: soronként, balról jobbra)

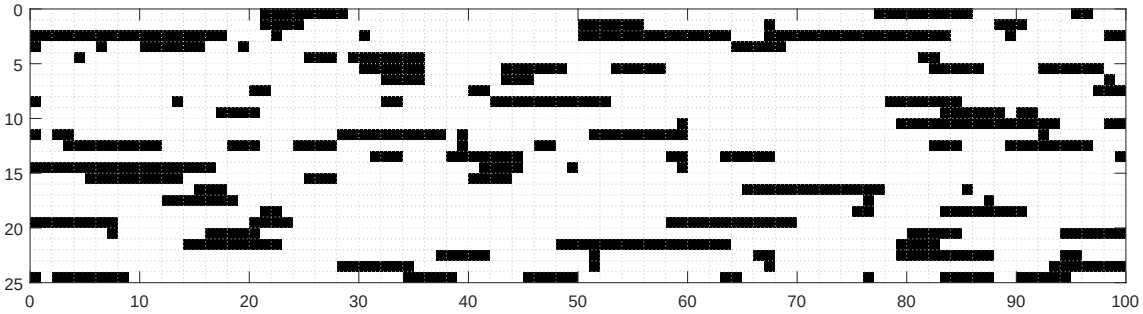
Kétállapotú Markov-modell

A kétállapotú Markov-lánc alapú adatvesztés (2.5. ábra) a legegyszerűbb a memóriával rendelkező adatvesztések közül. Adott egy kétállapotú Markov-lánc, az egyik állapotban a minta elérhető, a másikban elveszett. Vagyis a következő minta elérhetőségének a valószínűsége attól függ, hogy az aktuális minta elérhető-e vagy sem.



2.5. ábra. Kétállapotú Markov-modell alapú adatvesztés

A kétállapotú Markov-modell indikátorfüggvényét illusztrálja a 2.6. ábra. Az eddigiekkel ellentétben itt a jellemző viselkedés az, hogy változó hosszúságú teljes és üres sorozatok követik egymást.



2.6. ábra. Példa a kétállapotú Markov-modell alapú adatvesztésre (értelmezés: soronként, balról jobbra)

Gilbert-modell

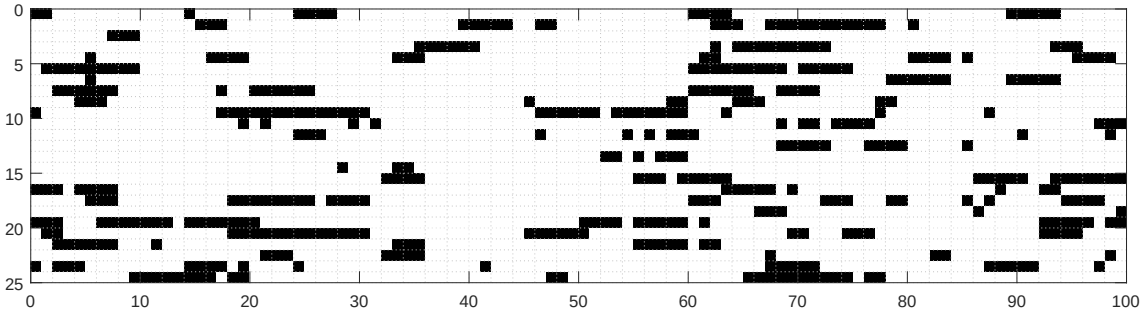
A Gilbert-modell [2] a kétállapotú Markov-modell egy általánosítása. A Markov-lánc egyik állapotában a minta determinisztikusan elérhető, míg a másikban véletlen független adatvesztés történik.

Az indikátorfüggvényt szemlélteti a 2.7. ábra. Hasonlít a kétállapotú Markov-modellhez, a lényeges különbség az, hogy a teljesen üres sorozatok helyett csupán többnyire üres sorozatokat találunk. Itt már nem tudjuk egyértelműen meghatározni a modell állapotait az indikátorfüggvény alapján.

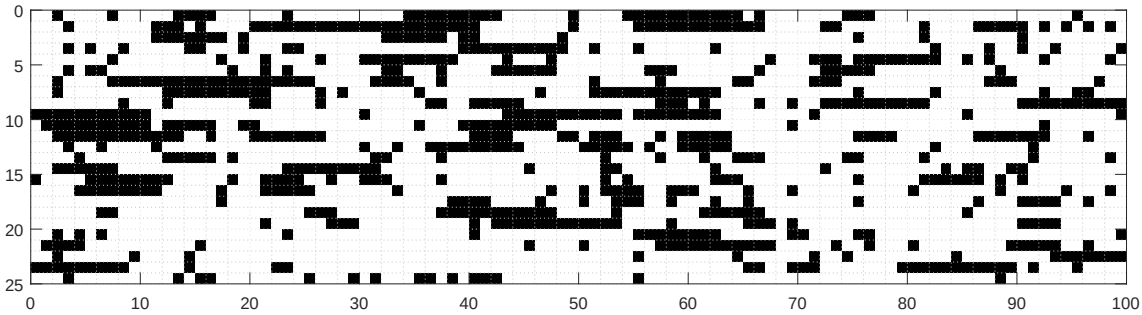
Komplementer Gilbert-modell

A komplementer Gilbert-modell nagyon hasonló a Gilbert-modellhez, a fő különbség az, hogy itt az egyik állapotban a minta nem determinisztikusan elérhető, hanem determinisztikusan elveszett.

A 2.8. ábrán láthatunk egy példát az indikátorfüggvényre. Ez is hasonlít a kétállapotú Markov-modellhez, viszont most a teljes sorozatok helyett csak többnyire elérhető sorozatokkal találkozunk. Az állapotok itt sem állíthatók vissza egyértelműen.



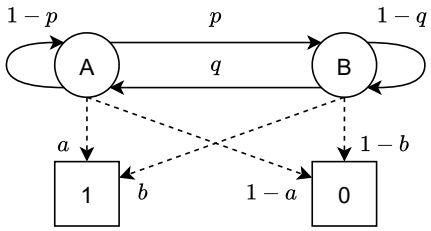
2.7. ábra. Példa a Gilbert-modell alapú adatvesztésre (értelmezés: soronként, balról jobbra)



2.8. ábra. Példa a komplementer Gilbert-modell alapú adatvesztésre (értelmezés: soronként, balról jobbra)

Gilbert–Elliott-modell

A Gilbert–Elliott-modell [3] (2.9. ábra) a Gilbert-modell további általánosítása, itt mindkét állapotban véletlen független adatvesztés lép fel, általánosan más adatvesztési aránnyal.



2.9. ábra. Gilbert–Elliott-modell alapú adatvesztés

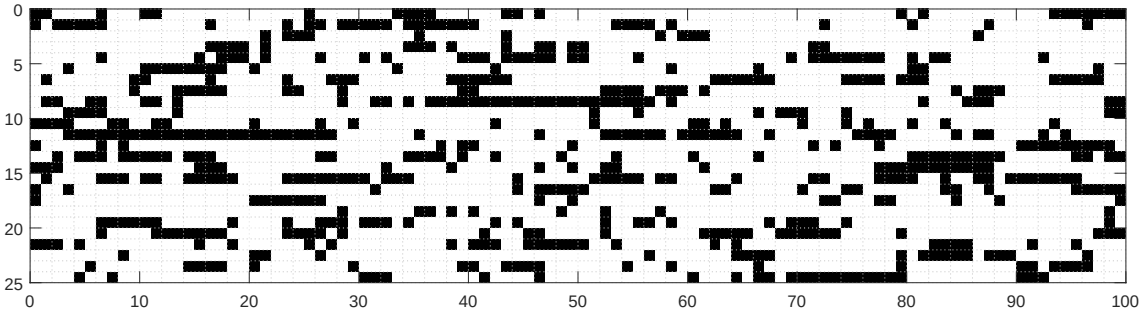
A modell indikátorfüggvényét mutatja be a 2.10. ábra. Most a jellemzően elérhető és a jellemzően elveszett sorozatok váltják egymást.

Általános blokkalapú modell

Ha az adatok blokkokban vesznek el, az nemcsak úgy képzelhető el, hogy a blokkok a többitől függetlenül, azonos valószínűséggel elérhetők. Valójában a blokkokra tetszőleges adatvesztési modellt alkalmazhatunk, és így egy blokkosított adatvesztést tudunk elérni.

Általános kétállapotú Markov-modell

[4] ismertet egy általános kétállapotú Markov-modellt. Ez a modell annyiban több, mint a



2.10. ábra. Példa a Gilbert-Elliott–modell alapú adatvesztésre (értelmezés: soronként, balról jobbra)

Gilbert-Elliott, hogy bevezet két új valószínűséget. A kimenet nemcsak az aktuális állapot függvénye, hanem a következőé is. Tehát először meghatározza, hogy mi lesz a következő állapot, de még nem vált állapotot, hanem az aktuális és a következő állapot alapján valamilyen valószínűséggel lesz elérhető a minta. Csak a kimenet meghatározása után történik meg az állapot átváltása az először meghatározott következő állapotra.

Többállapotú Markov-modell

A kétállapotú Markov-modell egy másik lehetséges általánosítási lehetősége, hogy bevezetünk további állapotokat. Ez esetben több állapotban lesznek a minták elérhetők vagy elveszettek, ez azonban még mindig az állapot determinisztikus függvénye. [5] mutat egy példát négyállapotú Markov-modellre.

Többállapotú rejtett Markov-modell

A Gilbert–Elliott modell másik lehetséges általánosítása a több állapot bevezetése. Így egy többállapotú rejtett Markov-modellhez jutunk, a háttérben egy többállapotú Markov-modellel, minden állapotban az állapottól függő adatvesztési arányú véletlen, független adatvesztéssel.

Hierarchikus rejtett Markov-modell

[6] ismerteti a hierarchikus rejtett Markov-modellt. Ez egy rekurzív modell, az alapja a rejtett Markov-modell. Ebből úgy kapunk hierarchikus modellt, hogy egyes állapotokhoz hozzárendelünk egy új rejtett Markov-modellt, és ebben az állapotban a kimenetet ennek a modellnek a kimeneti sorozata határozza meg.

Természetesen az állapotokba helyettesített rejtett Markov-modellek állapotait is ugyanúgy tovább lehet osztani. Minden ilyen helyettesítésnél az új rejtett Markov-modell kiegészül egy visszatérési állapottal, amely zárja a modell kimeneti sorozatát. Így gyakorlatilag egy állapotfán ugrál a rendszer állapota, a fa leveleiben képződnek a kimenetek, és egy állapotváltás vagy egy adott állapot közvetlen leszármazottai között, vagy egy adott állapotból egy szinttel felfelé történik.

Hosszabb memóriájú (magasabbrendű Markov-) modellek

[7] bemutatja, hogy hogyan lehet bináris sztochasztikus folyamatokat generálni, úgy, hogy a spektrumuk csak pólusokat tartalmazzon. Ehhez felhasznál egy olyan Markov-láncot, amely állapota az utolsó néhány minta. Azt, hogy az aktuális állapot az utolsó néhány minta, magasabbrendű Markov-modellnek nevezzük, például, ha az utolsó két állapottól függ a következő állapot, akkor ez egy másodrendű Markov-modell.

A magasabbrendű Markov-modell valójában a többállapotú Markov-modell speciális esete. A többállapotú Markov-modell állapotai a magasabbrendű modell memóriájának lehetséges tartalmai, a magasabbrendű modell átmeneti valószínűségeit kölcsönösen egyértelműen meg tudjuk feleltetni a többállapotú modell átmeneti valószínűségei egy részhalmazának. A többállapotú modell többi átmeneti valószínűsége 0, hiszen például 11 memóriatartalomból egy minta alatt nem lehet 00 memóriatartalom.

Az ötlet, hogy a következő állapotot nemcsak az aktuális, hanem még néhány korábbi állapot határozza meg, alkalmazható a többi adatvesztési modellre is.

A modellezett rendszerhez illesztett modellek

Amennyiben elég ismerettel rendelkezünk az adatvesztési folyamatról, készíthetünk egy olyan modellt, amely a folyamat fizikai működéséhez illeszkedik. Elképzelhető, hogy többféle modellből tudjuk összerakni ezt a rendszerhez illesztett modellt, például az egyik működési állapotában nincs adatvesztés, egy másikban véletlen független, egy harmadikban másodrendű Markov-modell stb. a megfelelő modell. Ezeket a működési állapotokat és átmeneteiket modellezhetjük egy Markov-lánccal, és így kialakul egy hierarchikus modell.

Várhatóan a rendszerhez illesztett modellek írják le a legjobban az adott adatvesztést, de előállításuk sok a priori ismeretet igényel. Például UDP-s kapcsolatok egy ilyen modelljét adja meg [8].

Időfüggő (paraméterű) modellek

Amennyiben tudjuk, hogy az adatvesztés időben változik, és ennek miként jéről van elképzelésünk, akkor azt figyelembe vehetjük úgy, hogy ezt a változást az eddig megismert modellekbe, az eddig megismert technikákkal beépítjük. A változások kezelésének másik módja, hogy választunk egy modellt, ami mindvégig jól írja le az adatvesztést, csupán más és más paraméterekkel, és a paramétereket nem konstansokként, hanem időtől függő változókként kezeljük.

Az is elképzelhető, hogy nemcsak a modell paraméterei, hanem a struktúrája is változik az idővel. [9] közlekedési témában mutat egy példát az időfüggő modellek alkalmazására.

Soros, párhuzamos modellek

A soros (párhuzamos) modellek nem mások, mint adatvesztési modellek (komponensek) soros (párhuzamos) kapcsolásai. Soros modell esetén a minta pontosan akkor elérhető, amennyiben a modellt alkotó összes komponens indikátorfüggvénye 1. Vagyis például egymás után több kommunikációs csatornán kell átvinnünk az adatot, és bármelyikben történhet adatvesztés. Az adatátvitel pontosan akkor sikeres, amennyiben az összes csatornán sikeres az átvitel. Soros modelleket vizsgál [5].

Párhuzamos modell esetén a minta pontosan akkor elérhető, ha legalább az egyik komponens indikátorfüggvénye 1. Egy lehetséges példa: van több, egyenként rossz minőségű átviteli lehetőségünk, és ezért az összesen elküldjük az adatot. Vagy, több, egymástól független eseménynek kell bekövetkeznie ahhoz, hogy adatvesztés lépjen fel.

2.5. Az adatvesztés mint a nem egyenletes mintavétel speciális esete

Amikor mintavételezésről beszélünk, általában az ideális, egyenletes mintavételezésre gondolunk. Ez azt jelenti, hogy a mintákat a következő időpontokban vesszük:

$$t_n = t_0 + nt_s, n \in \mathbb{Z} \cap [n_1, n_2], \tag{2.8}$$

ahol t_s a mintavételi periódusidő (a mintavételi frekvencia reciproka), n a minták indexe, n_1 és n_2 pedig a mérés időbeli határait jelöli ki.

A gyakorlati megvalósításban a mintavételi időpontok nem egyeznek meg tökéletesen a (2.8) egyenlet által adottakkal, továbbá a mintavételi periódusidő is kis mértékben változhat. Ezek a hatások azonban elég kicsik ahhoz, hogy az ideális mintavétel egy jó közelítés legyen.

Ha az ideális mintavétel feltételezését elhagyjuk, akkor a skála másik végén megtaláljuk a tetszőleges nem egyenletes mintavételt: a mintákat tetszőleges t_n időpontokban vesszük. Ezzel gyakran találkozunk a csillagászatban, ahol a méréseket a nappal-éjszaka ritmus, illetve az időjárás is befolyásolja. Elképzelhetők olyan mérőrendszerek, amelyeknél valamilyen külső esemény indítja el a mérést, ekkor nem feltétlenül jogos azt feltételezni erről az eseményről, hogy periodikus lenne.

Az adatvesztés egy speciális nem egyenletes mintavétel. A mintákat a

$$t_n = t_0 + nt_s, n \in Z \subset \mathbb{Z} \cap [n_1, n_2], \tag{2.9}$$

időpontokban vesszük. Amikor mintát veszünk, azt egy olyan időpontban tesszük, amikor az ideális mintavétel szerint is vennénk, viszont nem veszünk mintát minden ilyen időpontban.

A nem egyenletes mintavételként való modellezés alapvetően különbözik az indikátorfüggvény megközelítésétől. Mindkét esetben van két számsorozatunk, melyből az egyik a „hasznos jel”. Az indikátorfüggvény esetén a másik számsorozat megmutatja, hogy a hasznos jel érvényes-e, míg a nem egyenletes mintavételnél a másik számsorozat a mérések időpontja. Természetesen a kétféle modell könnyen átalakítható egymásba.

3. fejezet

Spektrumbecslési eljárások

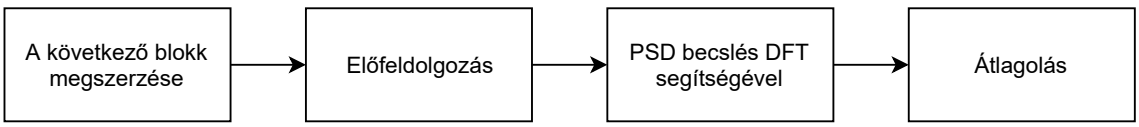
Az általam megismert eljárások három fő csoportra oszthatók, a probléma megközelítése alapján. Egyes eljárások a spektrum kiszámítására továbbra is a DFT-t alkalmazzák, és ezen kívül a jelen még egyszerű műveleteket hajtanak végre. Egy másik csoport a jelet modellezi, és ez alapján készít valamilyen becslőt. A harmadik csoportba tartoznak a nem egyenletesen mintavételezett jelek spektrumának becslésére kidolgozott eljárások. A spektrumbecslés alatt a dolgozatban a teljesítménysűrűség-spektrum (PSD) becslését vagy a harmonikus komponensek meghatározását értjük, valós jelekre.

3.1. DFT-alapú eljárások

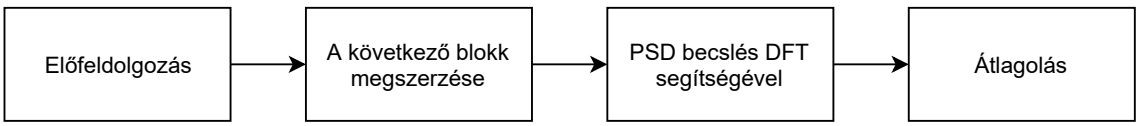
A DFT-alapú eljárásokban közös, hogy a spektrumot DFT-vel számítják ki. Ezen kívül végezhetnek még egyszerű kiegészítő előfeldolgozást, mellyel a felbontást növelni, vagy a szivárgást csökkenteni lehet.

A DFT-alapú eljárásokkal gyakorlati lehetőség van mind online (valós idejű), mind offline feldolgozásra. Online feldolgozás esetén a mintákat folyamatosan kapjuk a mérőrendszerből, így gyakorlatilag egy adatfolyam áll rendelkezésünkre, míg offline feldolgozás esetén rendelkezésünkre áll egy korábbi mérési regisztrátum, azaz egy véges méretű adatsorral dolgozunk. Az online alkalmazást az eljárások kis számítás- és memóriaigénye teszi lehetővé. Az eljárások általános hátránya a rögzített frekvenciarács.

A 3.1. ábra mutatja a DFT-alapú eljárások általános sablonjait. Az első típusban az előfeldolgozás a blokkosítás után, a másodikban a blokkosítás előtt történik. A továbbiakban bemutatásra kerülnek az egyes lépések.



(a) Előfeldolgozás a blokkosítás után



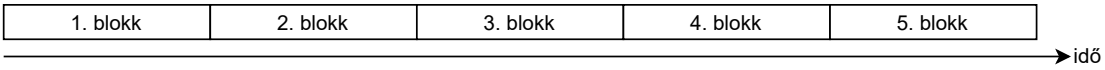
(b) Előfeldolgozás a blokkosítás előtt

3.1. ábra. A DFT-alapú eljárások sablonjai

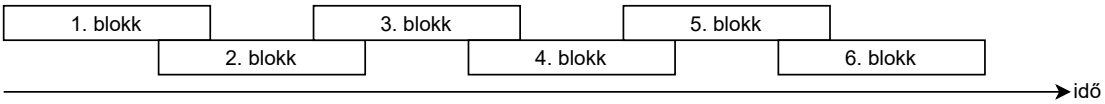
3.1.1. A következő blokk megszerzése

A DFT-alapú eljárások blokkos feldolgozó eljárások, amely azt jelenti, hogy az adatokat blokkokban dolgozzák fel. Online feldolgozás esetén a blokkokat úgy kapjuk, hogy megvárjuk, amíg egy blokknyi minta összegyűlik, míg offline feldolgozáskor az adatsorból vágunk ki megfelelő méretű részeket.

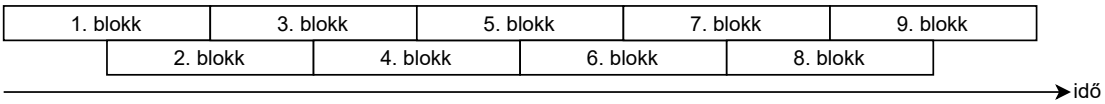
A blokkok előállításánál alkalmazhatunk átfedést is. Átfedés nélkül a blokkok éppen egymás után kezdődnek, minden minta pontosan egy blokkba tartozik (3.2.a. ábra). Az átfedés használata azt jelenti, hogy az aktuális blokk vége alkotja a következő blokk elejét. Például 25%-os átfedés (3.2.b. ábra) esetén az aktuális blokk utolsó negyede egybeesik a következő blokk első negyedével. 50%-os átfedést javasol [10], továbbá [11] alapján akár 75% is megengedhető.



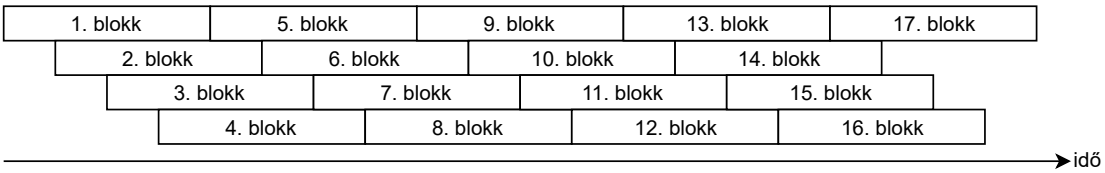
(a) Nincs átfedés



(b) 25%-os átfedés



(c) 50%-os átfedés



(d) 75%-os átfedés

3.2. ábra. Átfedő blokkok használata

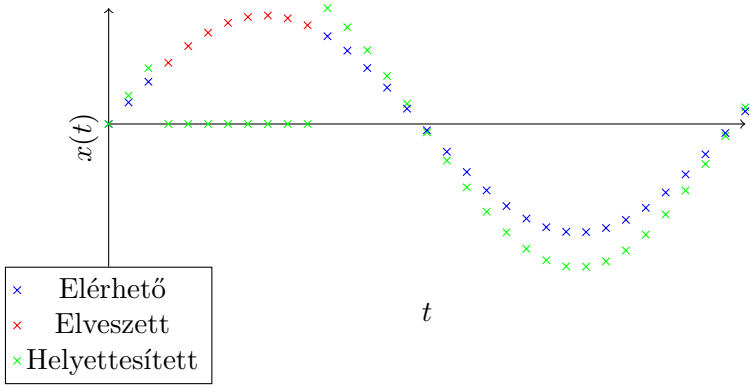
3.1.2. Előfeldolgozás

A DFT-alapú eljárások közötti lényegi különbségek az előfeldolgozásban jelennek meg, így az eljárásokat ez alapján különböztetjük meg.

3.1.2.1. Előfeldolgozás a blokkosítás után

Nullával helyettesítés
Az elveszett adatok nullákkal való helyettesítése az indikátorfüggvény definíció szerinti alkalmazását jelenti. Egyik oldalról mondhatjuk, hogy ekkor nem csinálunk semmilyen kiegészítő számítást. Másik oldalról viszont, általánosan semmi sem garantálja, hogy az elveszett minták helyére nulla kerüljön, ezt valahol a hardverben, a protokollban vagy az algoritmusban implementálni kell, így nevezhető egy önálló eljárásnak.

Továbbá, a teljesítményszint megtartása érdekében a blokkot még skálázni kell. Ha csupán L minta elérhető a blokk N mintájából, akkor az ebből számított PSD becslő a valós $\frac{L}{N}$-ed része lesz. Ennek kiküszöbölésére a blokk minden mintáját $\frac{N}{L}$-lel meg kell szorozni. Az eljárást a 3.3. ábrán láthatjuk (az ábra egy DFT blokkot mutat).



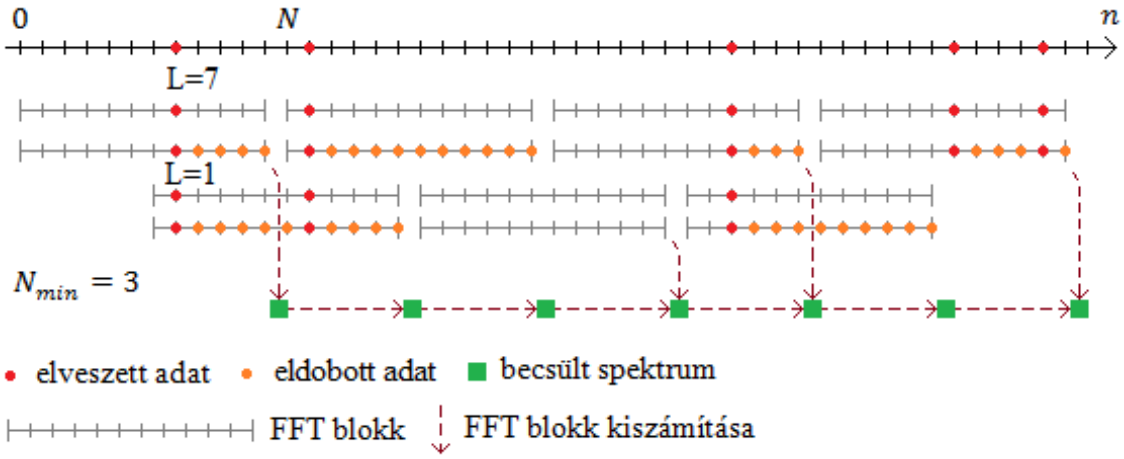
3.3. ábra. A nullával helyettesítés előfeldolgozás

A nullával való helyettesítés módszerére láttunk példát a 2.2. ábrán. A példa alapján is látszik, hogy ez az eljárás nem túl hatékony, de gyenge adatvesztésnél előfordulhat, hogy elegendő.

Zero padding

[12] ismerteti a Zero padding eljárást, melynek az alapötlete, hogy a DFT-blokkban keressük meg az első elveszett pont helyét, és azzal kezdődően nullázzuk ki a DFT-blokkot. Legyen L az így módosított blokkban lévő elérhető minták száma. Ha $L < L_{\min}$, akkor a blokkból nem számítunk spektrumot. Ha $L \geq L_{\min}$, akkor a blokk mintáit $\frac{N}{L}$ szeresére növeljük a spektrumszámítás előtt.

Ez azt jelenti, hogy az eredetileg elveszett adatokon kívül továbbiakat is eldobunk. Az eljárást a 3.4. ábra szemlélteti.



3.4. ábra. A zero padding eljárás vázlata

A további adatok eldobása kontraintuitív, de az eljárás mélyebb vizsgálatával megérthető. Egyrészt, amennyiben túlságosan sok mintát dobunk el, a blokkból nem számítunk

DFT-t. Továbbá, a megmaradó részt úgy súlyozzuk, hogy a jel teljesítményéből az eldobással ne veszítsünk, és csak a megmaradó részt ablakozzuk. Ezekkel kettős hatást érünk el: egyrészt, a csúcsoktól távol a szivárgás csökken, viszont a csúcsok kiszélesednek.

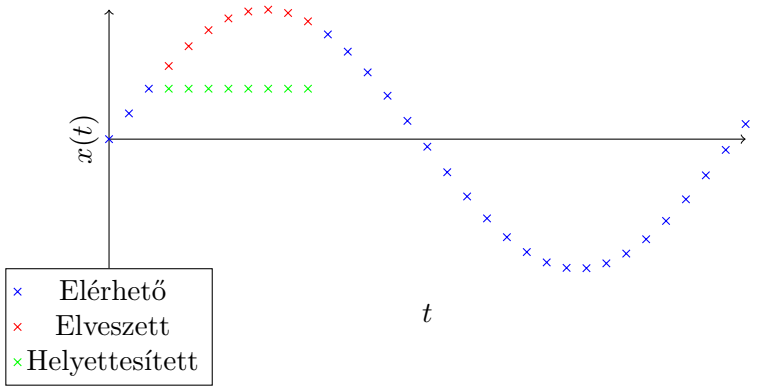
3.1.2.2. Előfeldolgozás a blokkosítás előtt – interpolációs eljárások

Az interpolációs eljárások általános jellemzője, hogy a hiányzó mintákat a közeli elérhető minták alapján próbálják meg pótolni. A cél az, hogy ha elveszett a minta, próbáljuk meg megbecsülni, vajon mi lehetett. Ezt tehetjük például nulladrendű tartóval, elsőrendű tartóval, szomszédos mérések közötti lineáris interpolációval, vagy akár bonyolultabb módszerekkel is. Az interpolációnak sajnos a mért spektrumra nézve is hatása van, például a nulladrendű tartó sinc jelleggel torzítja a spektrumot.

Nulladrendű tartó
A nulladrendű tartó eljárása az alábbi egyenlettel fogalmazható meg:

$$y_n = \begin{cases} x_n & \text{ha } K_n = 1 \\ y_{n-1} & \text{ha } K_n = 0 \end{cases}, \tag{3.1}$$

vagyis az ismeretlen mintákat a legutóbbi ismert mintával helyettesítjük. Az eljárást a 3.5. ábrán láthatjuk.

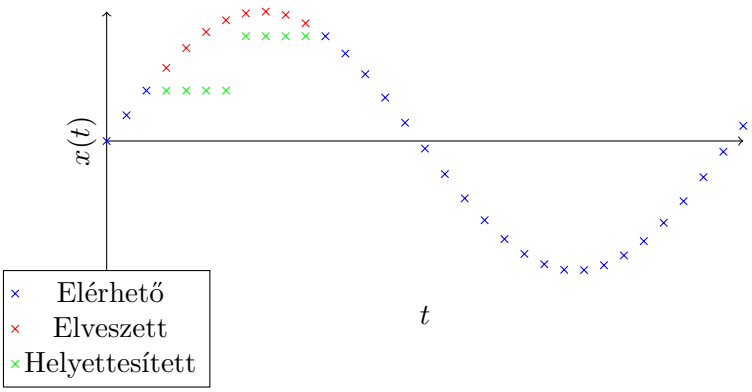


3.5. ábra. A nulladrendű tartó előfeldolgozás

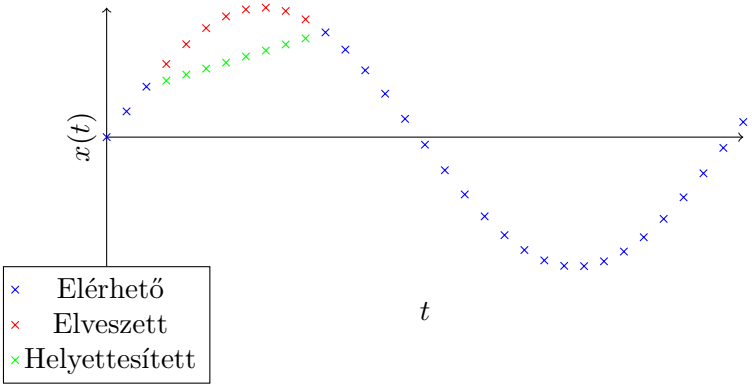
Az eljárás továbbfejlesztéseként [13] ismertet egy eljárást, amely nulladrendű tartót alkalmaz, majd a spektrumból a torzítást dekonvolúcióval próbálja meg eltávolítani.

Legközelebbi szomszéd A legközelebbi szomszéd előfeldolgozási eljárás először megvárja, amíg befejeződik egy üres sorozat, majd az elemeit a hozzájuk időben legközelebbi elérhetővel helyettesíti. Mivel az előfeldolgozáshoz meg kell várni az üres sorozat végét, ezért online feldolgozás esetén ez az eljárás nehézkes. Az eljárást a 3.6. ábra szemlélteti.

Lineáris interpoláció A lineáris interpoláció alkalmazása esetén is egészben kezeljük az üres sorozatokat. Amint egy üres sorozat befejeződött, az elemeit a határoló két elérhető minta közötti lineáris interpoláció szerinti értékkel helyettesítjük. Ez az eljárás is nehézkes online feldolgozás esetén. Az eljárást a 3.7. ábra mutatja.



3.6. ábra. A legközelebbi szomszéd előfeldolgozás



3.7. ábra. A lineáris interpoláció előfeldolgozás

3.1.3. PSD becslés a DFT segítségével

A PSD becslő a DFT ismeretében könnyen kiszámítható. Ha a DFT értéke a k. pontban X_k, akkor a PSD becslő értéke a k. pontban

$$S_k = \frac{1}{N f_s} |X_k|^2 , \tag{3.2}$$

ahol f_s a mintavételi frekvencia.

A PSD fizikai jelentése az, hogy egy adott f frekvencia $\mathrm{d}f$ környezetében a jel mekkora energiát hordoz. Egy adott frekvenciasávbeli energia számításához a PSD-t arra a sávra integrálni kell. A DFT frekvenciafelbontása arányos a mintavételi frekvenciával ($\Delta f = \frac{f_s}{N}$), így minél nagyobb a mintavételi frekvencia, annál nagyobb sávból (DFT „bin") gyűjt be egy pont energiát. Mivel a PSD sűrűségjellegű mennyiség, ezért a bin-be eső energia kiszámításakor a bin szélességével megszorozzuk a PSD értékét. Ezért szükséges a mintavételi frekvenciával való osztás.

Az $\frac{1}{N}$-es szorzás azért szükséges, mert a DFT-nek nem az ortonormált változatát használjuk. (1.1)-ben a szummázott értékek szorzója 1, és az inverz transzformációnál van egy $\frac{1}{N}$-es szorzó. Az ortonormált DFT-ben mindkét irányban $\frac{1}{\sqrt{N}}$-es szorzót kell alkalmazni. A transzformáció ortonormáltságából következik, hogy az energiát megtartja. Most az ortonormált transzformáció helyett olyat használunk, ami az energiát N-szeresére növeli (az $\frac{1}{\sqrt{N}}$ helyett 1 szerepel szorzóként, ez $\sqrt{N}$-es arány, és energia esetén ezt négyzetesen kell venni). A PSD számításánál az $\frac{1}{N}$-es szorzó ezt a hatást ellensúlyozza.

Ha arra vagyunk kíváncsiak, hogy egy frekvencia környezetében mekkora a jel átlagos teljesítménye, akkor a PSD értékét a mérési idővel kell még elosztani. Ekkor a mintavételi frekvencia kiesik a szorzóból, a szorzó $\frac{1}{N^2}$ értéket vesz fel. Az irodalomban a PSD további normalizálásai is előfordulnak, elsősorban statisztikai próbák céljára.

3.1.4. Átlagolás

Az egyes blokkokból készített becslőket a variancia csökkentése érdekében célszerű átlagolni. Az átlagolásra a két legelterjedtebb módszer az exponenciális és a lineáris átlagolás.

3.1.4.1. Exponenciális átlagolás

Az exponenciális átlagolás paramétere az $\alpha \in (0, 1]$ együttható, amely azt mutatja meg, hogy az új becslőt milyen súllyal vesszük figyelembe az átlagolásban. Az átlagolási egyenlet:

$$S_{n,\text{átlagolt}} = \alpha S_{n,\text{új}} + (1-\alpha)\, S_{n-1,\text{átlagolt}} = S_{n-1,\text{átlagolt}} + \alpha \left(S_{n,\text{új}} - S_{n-1,\text{átlagolt}}\right), \quad (3.3)$$

ahol n az időindex, és az átlagolást minden frekvenciára külön-külön kell elvégezni.

Az exponenciális átlagolás onnan kapta a nevét, hogy egy konstans bemeneti szintet az átlag $(1-\alpha)^n$ jellegű hibával közelít az időben. Minél nagyobb α értéke, annál gyorsabb a beállás, ám annál kisebb a zajelnyomás mértéke.

Az exponenciális átlagolás könnyen implementálható mind online, mind offline feldolgozás esetén.

3.1.4.2. Lineáris átlagolás

A lineáris vagy más néven csúszóablakos átlagolás esetén az átlagolt értéket az utolsó M becslő átlaga adja:

$$S_{n,\text{átlagolt}} = \frac{1}{M} \sum_{i=n-L+1}^{n} S_i. \quad (3.4)$$

Ehhez az utolsó M becslőt el kell tárolni, ami – tekintve, hogy ezek vektorok – memóriaigényes feladat, így online megvalósítása nehézkes lehet. Ennek ellenére a módszer számításigénye nem lényegesen nagyobb, mint az exponenciális átlagolásé. Itt – az exponenciális átlagolással ellentétben – egy adott időpontban készített becslő elméletileg is csak véges ideig van hatással az átlagra.

A módszer változata a végtelen lineáris átlagolás, amely esetén az összes eddigi becslőt átlagoljuk. Ez online is könnyen megvalósítható rekurzív átlagolással:

$$S_{n,\text{átlagolt}} = \frac{1}{M_n} \left((M_n - 1)\, S_{n-1,\text{átlagolt}} + S_n \right), \quad (3.5)$$

ahol M_n az eddig kiszámított blokkok száma.

3.2. Modellalapú eljárások

A modellalapú eljárások jellemzője, hogy felállítanak egy jelmodellt, és ennek a paramétereit becslik a minták alapján.

3.2.1. Vezérelt rezonátoros megfigyelő

3.2.1.1. A Fourier-jelmodell

A rezonátoros megfigyelő [14] a periodikus jelek modellezéséből indul ki. Minden periodikus jel felírható a komplex Fourier-sorával:

$$y_n = \sum_{k=-\infty}^{\infty} Y_k \mathrm{e}^{\mathrm{j}2\pi f_0 k n}, \tag{3.6}$$

ahol Y_k a k. Fourier-együttható, f_0 a relatív alapfrekvencia és j a képzetes egység. A relatív frekvencia használata azért kényelmes, mert így diszkrét időben a mintavételi frekvenciától függetlenül beszélhetünk frekvenciákról.

Feltételezhetjük, hogy a mintavételezéskor jó minőségű átlapolásgátló szűrőt alkalmaztak, így elegendő a $(-0{,}5, 0{,}5]$ relatív frekvenciatartományt vizsgálni. Ekkor jelnek csupán a $k \in \{-L, \ldots, L\}$ sorszámú komponensei maradnak meg diszkrét időben, így a szumma az alábbira módosul:

$$y_n = \sum_{k=-L}^{L} Y_k \mathrm{e}^{\mathrm{j}2\pi f_0 k n}. \tag{3.7}$$

Ezt az összegzést megvalósító rendszer a Fourier-jelmodell, melynek állapotváltozós leírása:

$$\mathbf{x}_{n+1} = \mathbf{A}\mathbf{x}_n, \tag{3.8}$$

$$y_n = \mathbf{c}^T \mathbf{x}_n, \tag{3.9}$$

$$\mathbf{A} = \langle \mathbf{z} \rangle, \tag{3.10}$$

$$\mathbf{c} = \begin{bmatrix} 1 & \ldots & 1 \end{bmatrix}^T, \tag{3.11}$$

$$\mathbf{v} = \begin{bmatrix} 0 & 1 & -1 & 2 & -2 & \ldots & L & -L \end{bmatrix}^T, \tag{3.12}$$

$$z_i = \mathrm{e}^{\mathrm{j}2\pi f_0 v_i}, \tag{3.13}$$

$$x_{i,0} = Y_{v_i}, \tag{3.14}$$

ahol $\mathbf{x}$ az állapotváltozók oszlopvektora, $\mathbf{A}$ a rendszermátrix, $\mathbf{c}$ a kimeneti vektor, amely jelen esetben az összegzést valósítja meg, $\cdot^T$ a transzponálási operátor, $\langle \cdot \rangle$ a diagonálmátrix képzés, $\mathbf{v}$ a Fourier-együtthatók indexeit az állapotváltozók indexeire leképező vektor, $\mathbf{z}$ az egyes harmonikus komponensek forgatását megvalósító pólusok, és $x_{i,0}$ az i. állapotváltozó kezdeti értéke.

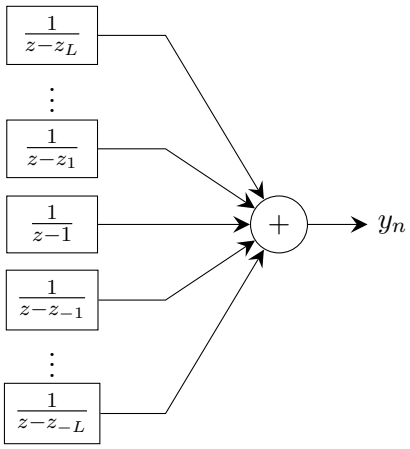
Vagyis a rendszer nem más, mint megfelelő sebességgel forgó állapotváltozók összegéből képzett kimenet. Egy állapotváltozót megvalósító ilyen komponenst nevezünk rezonátornak. A Fourier-jelmodellt ábrázolja a 3.8. ábra.

Amennyiben az alapfrekvencia valamely $f_0 = \frac{1}{N}$ érték, ahol N páros szám, akkor a 0,5 és $-0{,}5$ relatív frekvenciájú komponensek közül csak az egyiket kell a jelmodellbe bevenni, ekkor a $\mathbf{v} = \begin{bmatrix} 0 & 1 & -1 & 2 & -2 & \ldots & -(L-1) & L \end{bmatrix}^T$ helyzet áll elő, ahol $L = \frac{N}{2}$.

3.2.1.2. A rezonátoros megfigyelő

A Fourier-jelmodellre épített megfigyelő a rezonátoros megfigyelő (RBO), melynek egyenletei:

$$\mathbf{x}_{n+1} = \mathbf{A}\mathbf{x}_n + \mathbf{g}e_n, \tag{3.15}$$



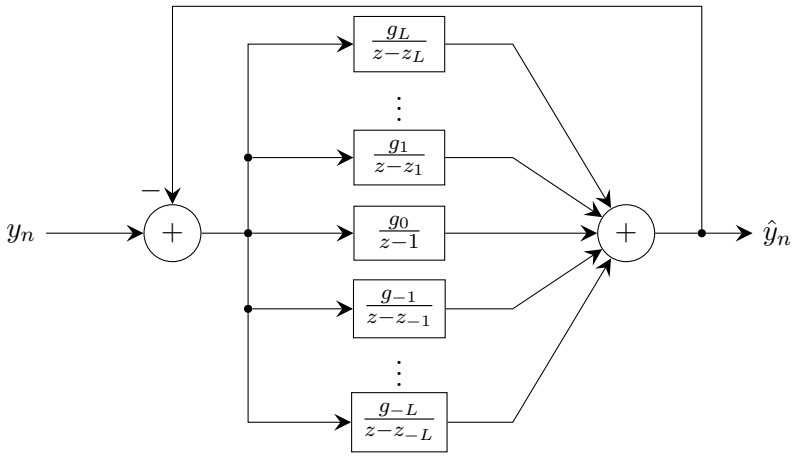
3.8. ábra. A Fourier-jelmodell

$$\hat{y}_n = \mathbf{c}^T \mathbf{x}_n, \tag{3.16}$$

$$e_n = y_n - \hat{y}_n, \tag{3.17}$$

$$\mathbf{g} = \frac{\alpha}{2L+1}\mathbf{A}\mathbf{c}, \tag{3.18}$$

ahol $\hat{y}_n$ a becsült jel, e_n a hibajel, $\mathbf{g}$ a visszacsatoló vektor és $\alpha \in (0, 1]$ a csillapítási együttható. A megfigyelőt a 3.9. ábra illusztrálja.



3.9. ábra. A rezonátoros megfigyelő

Az alapfrekvencia és az α érték függvényében három alapesetet különböztetünk meg:

- $f_0 = \frac{1}{N}$ (N egész), $\alpha = 1$: a rezonátorpozíciók az egységkörön egyenletesen helyezkednek el. Ha N páros, akkor a Nyquist-frekvenciához tartozó rezonátort is használjuk, és a (3.18) egyenletben $2L+1$ helyett N-nel kell osztani. A rendszer ortogonális transzformációkat végez, a DFT értékeit számítja ki, és N lépésben beáll. A beállítás előnye a gyors beállás, hátránya a rossz zajelnyomás.

- $f_0 = \frac{1}{N}$ (N egész), $\alpha \in (0, 1)$: bár a rezonátorpozíciók az egységkörön egyenletesen helyezkednek el, a beállás már nem véges. A rendszer exponenciális átlagolást végez $\beta = 1 - (1-\alpha)^N$ átlagolási paraméterrel. A megfigyelő pólusai $(1-\alpha)\,\mathrm{e}^{\mathrm{j}\frac{2\pi}{N}v_i}$.

- f_0 tetszőleges, $\alpha \in (0, 1]$: a beállás végtelen még $\alpha = 1$ esetén is. A rendszer beállása viszonylag gyors, ha a Nyquist-frekvenciáig az összes komponenst megfigyeljük.

A rezonátoros megfigyelőnek számos előnye van: a megfigyelőben megvalósított visszacsatolás miatt robusztus, így nem lépnek fel numerikus problémák az implementációban. A becsült frekvenciák lehetnek a DFT frekvenciái, de nem kell, hogy azok legyenek: inkoherens mintavétel esetén is képes pontos becslésre, ablakozás nélkül. Természetesen, ha a harmonikus komponensek amplitúdója változik, a struktúra képes követni. A paraméterekkel egyszerűen beállítható egy exponenciális átlagolás is. A módszer hátránya, hogy szükséges az alapfrekvencia ismerete, illetve, hogy nagyobb a számításigénye. További nehézséget okoz, ha ablakozni szeretnénk, mivel ezt utólag, a frekvenciatartományban kell megtennünk.

3.2.1.3. PSD becslés a rezonátoros struktúrával

A rezonátoros struktúra a komplex exponenciálisok együtthatóit becsli, ezért a PSD becslésére a harmonikus frekvenciákon ad lehetőséget, a becslő Dirac-delta sorozat lesz. Mivel a becsült értékek komplex exponenciálisok amplitúdói, ezért az effektívérték-négyzetek könnyen kiszámíthatók:

$$P_i = |x_i|^2 \quad \forall i \in \{1, \ldots, 2L + 1\}, \tag{3.19}$$

ahol P_i az i. állapotváltozó által leírt komplex exponenciális teljesítménye. Az energiához ezeket még a T mérési idővel kell megszorozni, így a PSD becslő:

$$S(f) = T \sum_{i=-L}^{L} P_i \delta(f - f_i), \tag{3.20}$$

ahol $\delta(\cdot)$ a Dirac-delta és f_i az i. komponens frekvenciája.

Arra kell még figyelni, hogy amikor az egyes harmonikus frekvenciákra eső (nem DC) teljesítményt (energiát) számítjuk, akkor a hozzá tartozó negatív és pozitív irányba forgó komplex exponenciálisok teljesítményét (energiáját) össze kell adni.

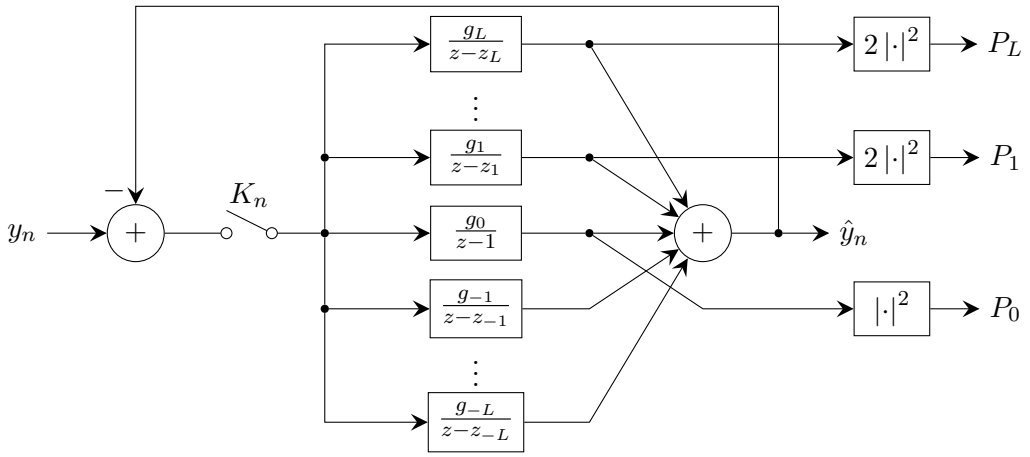
3.2.1.4. A vezérelt rezonátoros megfigyelő

A rezonátoros megfigyelő kiegészítésre került, hogy adatvesztés esetén is képes legyen a spektrumbecslésre [15]. Az így kapott struktúra a vezérelt rezonátoros megfigyelő, a továbbiakban ezt értjük RBO alatt. Az RBO csupán az állapotegyenletében különbözik az eredetitől:

$$\mathbf{x}_{n+1} = \mathbf{A}\mathbf{x}_n + \mathbf{g}K_n e_n, \tag{3.21}$$

vagyis a hibajelet az indikátorfüggvény aktuális értékével megszoroztuk. A struktúrát a 3.10. ábra szemlélteti. Itt már feltüntetésre kerültek a PSD becslés műveletei is, az ábrán a P_k indexek a harmonikus komponensek teljesítményére vonatkoznak. Az adatvesztést a struktúra úgy kezeli, hogy az időlépéshez tartozó frissítést minden ütemben elvégzi, míg az új bemeneti adathoz tartozó frissítést csupán akkor, amikor van minta, vagyis nem történt adatvesztés. Ezt szimbolizálja a K_n-nel vezérelt kapcsoló az ábrán. Fontos, hogy ez nem azt jelenti, hogy adatvesztés esetén megpróbálnánk kitalálni, hogy mi lehetett a minta, hanem kihasználjuk a lehetőséget arra, hogy módosítás nélkül a következő ütemre léptessük a modellt.

Amennyiben az adatvesztés véletlenszerű, a struktúra konvergens [15]. A beállást az α értéke és a μ adatelérhetőségi arány adja meg, a beállás olyan jellegű, mintha $\alpha' = \alpha\mu$



3.10. ábra. A vezérelt rezonátoros megfigyelő

csillapítású lenne. A vezérelt struktúra is megtartja a robusztusságát, és az alapfrekvencia szabad beállítását.

3.2.2. Vezérelt adaptív Fourier-analizátor

A precíziós mérések esetén sem az alapfrekvencia pontos ismeretét, sem az állandóságát nem tételezhetjük fel. Az RBO minden csatornája a rá jellemző frekvencia környékén egy keskeny sáváteresztő szűrőt valósít meg. Így amennyiben egy olyan jelet szeretnénk az RBO-val feldolgozni, amelynek a frekvenciája egy névleges érték körül kicsit ingadozik (például a hálózati feszültség), akkor bár az alapharmonikust valószínűleg kis hibával tudjuk mérni, de a felharmonikusok frekvenciája több RBO-frekvencián keresztül is változhat, ezzel a felharmonikusok pontos mérését ellehetetlenítve. Erre az igényre született meg az adaptív Fourier-analizátor (AFA) [16].

3.2.2.1. Jelmodell

Az adaptív Fourier-analizátor esetén is a jelmodell-megfigyelő utat járjuk be. A jelmodellben az alapfrekvenciát is egy állapotváltozónak tekintjük. Továbbá, most időfüggőnek írjuk fel az egyes komplex exponenciálisok együtthatóit is (technikailag eddig is lehettek azok, csak erre nem hívtuk fel a figyelmet).

$$y_n = \sum_{k=-\infty}^{\infty} x_{k,n}, \tag{3.22}$$

$$x_{k,n} = x_{k,n-1} \mathrm{e}^{\mathrm{j}2\pi f_{0,n} k n}, \tag{3.23}$$

ahol $Y_{k,n}$ a k. Fourier-együttható az n. ütemben és $f_{0,n}$ az alapfrekvencia az n. ütemben.

A mintavételezést figyelembe véve, mivel az alapfrekvencia változhat, ezért az aktuálisan megmaradó komponensek L_n száma is időfüggő lesz.

3.2.2.2. Megfigyelő: adaptív Fourier-analizátor

Az előbb ismertetett jelmodellre tervezett megfigyelő az adaptív Fourier-analizátor, melynek frissítési algoritmusa:

1. A figyelembe vett harmonikusok száma:

$$L_n = \left\lfloor \frac{0{,}5}{f_n} \right\rfloor. \tag{3.24}$$

2. A komplex exponenciálisok száma és változása:

$$N_n = 2L_n + 1, \tag{3.25}$$
$$\Delta N_n = N_n - N_{n-1}. \tag{3.26}$$

3. A Fourier-együtthatók $\mathbf{x}$ oszlopvektorának bővítése/szűkítése az aktuális méretre:

$$\mathbf{x}_n := \begin{cases} \begin{bmatrix} \mathbf{x}_n^T & \mathbf{0}^{1\times\Delta N_n} \end{bmatrix}^T & \text{ha } \Delta N_n > 0 \\ \begin{bmatrix} x_1 & \dots & x_{N_n} \end{bmatrix}^T & \text{ha } \Delta N_n \leq 0 \end{cases}, \tag{3.27}$$

ahol $\mathbf{0}^{1\times\Delta L_n}$ ΔL_n nullából álló sorvektor.

4. A kimeneti szorzók $\mathbf{r}$ oszlopvektorának bővítése/szűkítése az aktuális méretre:

$$\mathbf{r}_n := \begin{cases} \begin{bmatrix} \mathbf{r}_n^T & \mathbf{1}^{1\times\Delta N_n} \end{bmatrix}^T & \text{ha } \Delta N_n > 0 \\ \begin{bmatrix} r_1 & \dots & r_{N_n} \end{bmatrix}^T & \text{ha } \Delta N_n \leq 0 \end{cases}, \tag{3.28}$$

ahol $\mathbf{1}^{1\times\Delta L_n}$ ΔL_n egyesből álló sorvektor.

5. Az aktuális visszacsatoló vektor számítása:

$$\mathbf{g}_n = \frac{\alpha}{N_n}\mathbf{r}_n^*, \tag{3.29}$$

ahol $(\cdot)^*$ a komplex konjugálás. Az $\alpha \in (0, 1]$ együttható itt is exponenciális átlagolást eredményez.

6. Az $\hat{y}$ becsült kimenet és az e hibajel:

$$\hat{y}_n = \mathbf{r}_n^T \mathbf{x}_n. \tag{3.30}$$
$$e_n = y_n - \hat{y}_n. \tag{3.31}$$

7. A Fourier-együtthatók következő értékének kiszámítása:

$$\mathbf{x}_{n+1} = \mathbf{x}_n + e_n \mathbf{g}_n. \tag{3.32}$$

8. A frekvenciabecslő frissítése:

$$f_{n+1} = f_n + \frac{\angle(x_{2,n+1}) - \angle(x_{2,n})}{2\pi N_n}, \tag{3.33}$$

ahol $\angle(\cdot)$ egy komplex szám szögét adja meg radiánban, $x_{2,n+1}$ pedig a 2-es indexű (pozitív alapharmonikus) állapotváltozó $n+1$. ütembeli értéke.

A frekvencia követéséhez kihasználjuk, hogy amennyiben a jel becsült alapfrekvenciája eltér a valóstól, akkor az alapharmonikus becsült együtthatóvektora forog, mégpedig a frekvenciahibával arányos sebességgel. Tehát, ha a forgási sebesség nulla értékére szabályozunk, azzal a frekvenciát követni fogjuk.

9. Az állapotváltozók és a Fourier-együtthatók összerendelése:

$$\mathbf{v}_n = \begin{bmatrix} 0 & 1 & -1 & 2 & -2 & \dots & L_n & -L_n \end{bmatrix}^T . \tag{3.34}$$

Érdemes észrevenni, hogy minél nagyobb rendszámú egy felharmonikus, annál később helyezkedik el. Ez összhangban van azzal, hogy az állapotváltozó-vektor és a kimenetiszorzó-vektor méretének aktualizálását a végéről való levágással vagy a végéhez való illesztéssel végezzük el: a be- vagy kiléptetett harmonikus tagok az éppen figyelembe vett legnagyobb rendszámú tagok.

10. A kimeneti szorzók frissítése:

$$\mathbf{r}_{n+1} = \mathbf{r}_n \mathrm{e}^{\mathrm{j}2\pi f_{n+1}\langle \mathbf{v}_n \rangle} \tag{3.35}$$

Az alapfrekvencia f_0 kezdeti becslése értelmes határok között tetszőleges, jó gyakorlat például a névleges frekvencia.

A megfigyelő egyenleteit más megközelítésben írjuk fel, mint a Fourier-jelmodellt. A rezonátoros struktúrát forgó állapotváltozókkal írtuk le, vagyis az állapotváltozókat a rendszermátrix forgatta. Itt az állapotváltozók a mindenkori Fourier-együtthatók értékével egyeznek meg, és ezeket megfelelően forgatott komplex exponenciálisokkal szorozva, majd összegezve kapjuk meg a jel aktuális értékét. Ezen kívül figyelnünk kell rá, hogy az alapfrekvencia változhat, ezért rezonátorokat kell ki- vagy beléptetni, illetve a kimeneti szorzók forgási sebessége változó.

A gyakorlati megvalósításnál érdemes a kimeneti szorzók fázisait tárolni és ütemenként léptetni. Ekkor nem léphet fel az a probléma, hogy a sorozatos szorzások miatt a kimeneti szorzók az egységkörtől egyre inkább eltávolodnak. A fázis tárolása esetén a kimeneti szorzók legfeljebb kis szöghibát tartalmaznak, de ezt a frekvenciakövetés ellensúlyozza. Bár az RBO-nál az állapotváltozókat szorzással forgatjuk, ott ez a hibajelenség a visszacsatolás miatt nem léphet fel.

Az AFA-ban két csatolt szabályozás működik: egyrészt az alapfrekvenciát becsüljük a becsült alapharmonikus együttható forgásából, másrészt az egyes komponensek becslését csak a helyes frekvenciabecslő esetén tudjuk elvégezni.

Létezik az AFA-nak olyan változata is, amely képes hiba nélkül követni egy lineáris, logaritmikus vagy hiperbolikus sweep-et is [17]. Az előbb bemutatott algoritmus képes arra, hogy egy állandó frekvenciára hiba nélkül ráálljon. Ezt az algoritmust módosítottam ugyanolyan elvek mentén, mint ahogy a vezérelt rezonátoros megfigyelőt származtattuk, így alakult ki a vezérelt adaptív Fourier-analizátor algoritmusa.

3.2.2.3. Vezérelt adaptív Fourier-analizátor

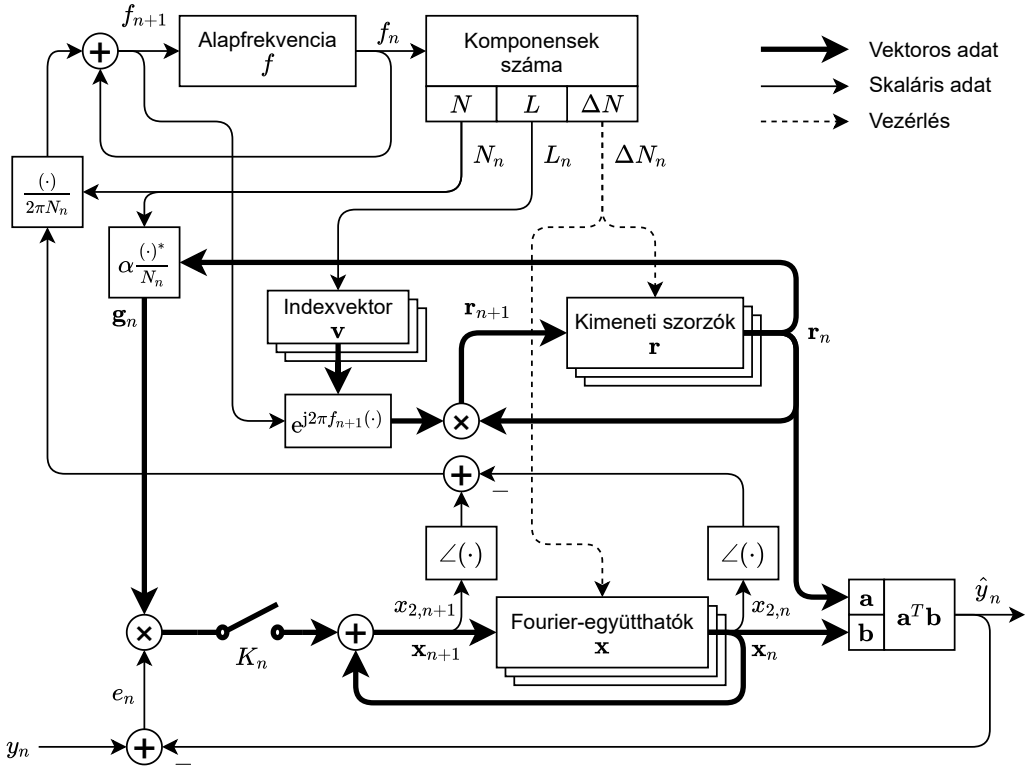
A vezérelt adaptív Fourier-analizátor algoritmusa egyetlen helyen különbözik az eddigitől: a Fourier-együtthatók frissítésénél (3.32) helyett az

$$\mathbf{x}_{n+1} = \mathbf{x}_n + e_n \mathbf{g}_n K_n \tag{3.36}$$

egyenletet kell használni.

A gondolatmenet ugyanaz, mint a vezérelt rezonátoros struktúra esetén: ha nincs minta, akkor csak az időléptetést hajtjuk végre. Jogosan vetődik fel a kérdés, hogy miért csak egy helyen módosítunk az algoritmuson, ha az két szabályozási kört tartalmaz. A válasz a frekvenciabecslő frissítési módszerében rejlik: ha csak az időlépés alapján léptetjük a modellt, akkor a Fourier-együtthatók nem változnak, így a becsült frekvencia is változatlan marad. Vagyis kihasználtuk, hogy két csatolt szabályozásunk van.

A vezérelt Fourier-analizátor blokkvázlatát mutatja a 3.11. ábra.



3.11. ábra. A vezérelt adaptív Fourier-analizátor

A vezérelt AFA algoritmus teljesen új, a dolgozat keretein belül ennek csupán „proof-of-concept" jellegű ismertetése a cél. Az algoritmus esetleges [17] szerinti kiterjesztése, további javítása jövőbeli feladat lehet. A dolgozatban a továbbiakban AFA alatt a vezérelt adaptív Fourier-analizátort értjük.

3.3. A nem egyenletes mintavétel eljárásai

A nem egyenletes mintavételezés gyakran előfordul például a csillagászatban. Mivel a nem egyenletesen mintavételezett jelek esetén is értelmes feladat a frekvenciatartománybeli analízis, illetve a csillagászoknak is szükségük volt rá, kidolgoztak többféle eljárást a nem egyenletesen mintavételezett jelek spektrumának becslésére.

Alapesetben a nem egyenletes mintavétel a tetszőleges időpontokban való mintavételezést jelenti, itt azonban a mintákat csak egy adott mintavételi frekvencia szerinti időpontokban vehetjük (de nem veszünk mintát az összes ilyenben). A nem egyenletes mintavételezés felveti a Nyquist-frekvencia létezésének kérdését, ugyanis az az egyenletes mintavételezésnél bekövetkező frekvenciatartománybeli átlapolódás meghatározó frekvenciája. [18] foglalkozik vele, hogy hogyan kell a Nyquist-frekvenciát meghatározni nem egyenletes mintavétel esetén. A cikk a nem egyenletes mintavételezés és az adatvesztés kapcsolatára világít rá: meg kell keresni a legnagyobb olyan időtartamot, amellyel a jel tekinthető adatvesztéssel terhelt, egyenletesen mintavételezett jelnek. Ha létezik ilyen, akkor ebből az időtartamból – mintha ezzel az időtartammal mintavételeztünk volna egyenletesen – kiszámítható a Nyquist-frekvencia.

Esetünkben a keresett időtartam pont a mintavételi időtartam, tehát az adatvesztés a Nyquist-frekvenciát nem változtatja meg akkor sem, ha nem egyenletes mintavételként tekintünk rá. [18] eredménye pont arra világít rá, hogy a nem egyenletes mintavételezés és az adatvesztés tekinthetők ugyanannak a jelenségnek a két ekvivalens leírásának.

3.3.1. A projektív szemlélet

Sokféle, széleskörűen elterjedt eljárás alapszik azon, hogy a megfigyelt adatokat próbafüggvények lineáris kombinációjaként állítsa elő. Ilyenek például a Fourier-analízis, ahol a harmonikus viszonyban lévő trigonometrikus függvényeket használjuk, vagy a polinomiális regresszió, ahol az 1, x, x^2, ... függvényeket.

A hamarosan ismertetendő eljárások megértéséhez a projektív szemléletet hívjuk segítségül, melyet [19] mutat be.

3.3.1.1. A projektív feldolgozás általános folyamata

Általánosan fogalmazva, adott véges sok (N) mérési adat (mintavételi időpont és érték). Az értékekre tekinthetünk úgy, mint egy vektorra egy N dimenziós vektortérben, ez lesz az adatvektor:

$$\mathbf{x} = \begin{bmatrix} x(t_1) & \dots & x(t_N) \end{bmatrix}^T. \tag{3.37}$$

Választunk néhány $\phi_\alpha(t)$ próbafüggvényt, melyeket a mintavételi időpontokban behelyettesítünk, így ϕ_α próbavektorokat kapunk:

$$\phi_\alpha = \begin{bmatrix} \phi_\alpha(t_1) & \dots & \phi_\alpha(t_N) \end{bmatrix}^T. \tag{3.38}$$

Megjegyzendő, hogy itt a mintavételi időpontok alatt csak a sikeres mintavételek időpontjait értjük. Ez érvényes a szakasz összes eljárására.

A próbavektorokat egymás mellé írva megkapjuk a Φ próbamátrixot:

$$\Phi = \begin{bmatrix} \phi_\alpha & \phi_\beta & \dots \end{bmatrix}. \tag{3.39}$$

Az szeretnénk, hogy az adatvektort előállítsuk a próbavektorok lineáris kombinációjaként:

$$\mathbf{x} = \sum_\alpha c_\alpha \phi_\alpha = \Phi \mathbf{c}. \tag{3.40}$$

Mivel több egyenletünk van, mint ismeretlenünk, az előző egyenlet általánosan nem oldható meg. Ezért azt a $\mathbf{y}$ modellvektort keressük, mely a legkisebb négyzetes hibával közelíti az adatvektort:

$$\mathbf{y} = \Phi \mathbf{c}. \tag{3.41}$$

A legkisebb négyzetes hibájú közelítés a pszeudoinverz segítségével adható meg:

$$\mathbf{c} = \left(\Phi^T \Phi\right)^{-1} \Phi^T \mathbf{x}, \tag{3.42}$$

ahol $\Phi^T\Phi$ a metrikus tenzor.

Érdemes megjegyezni, hogy a metrikus tenzor invertálását kifejezett precizitással kell elvégezni. Ez az invertálás elkerülhető, amennyiben a modellvektoraink ortonormált bázist alkotnak, mivel ekkor a metrikus tenzor egységmátrix, így az előbbi egyenlet lényegesen leegyszerűsödik:

$$\mathbf{c} = \Phi^T \mathbf{x}. \tag{3.43}$$

Az együtthatóvektor ismeretében a meghatározható a modellfüggvény:

$$y(t) = \sum_{\alpha} c_\alpha \phi_\alpha(t). \tag{3.44}$$

[19] az együtthatóvektor meghatározásának képletét úgy vezeti le, hogy tekinti a próbavektorok által kifeszített alteret, és ebbe vetíti az adatvektort. A két vektor különbsége a hibavektor, amely merőleges az előbbi altérre.

3.3.1.2. A DFT értékelése projektív szempontból

Az N pontos DFT esetén a próbafüggvények azok a komplex exponenciálisok, melyek az N minta alatt egész egész sok fordulatot tesznek. Diszkrét időben N különböző ilyen komplex exponenciális létezik, ezek páronként ortogonálisak.

Érdemes kiemelni a konstansfüggvényt a próbafüggvények közül, ez bármilyen időpontokban mintavételezve az egy-vektort adja eredményül:

$$\mathbf{1} = \begin{bmatrix} 1 & \dots & 1 \end{bmatrix}^T. \tag{3.45}$$

Ismételten, az egy-vektor ortogonális az összes többi próbavektorra.

Mivel a próbavektorok páronként ortogonálisak és azonos hosszúak, (de nem feltétlenül normáltak), ezért a metrikus tenzor egy konstanssal való szorzásnak felel meg. Ezt a konstanst kell ellensúlyozni a DFT értékeiből történő PSD számítás során.

Adatvesztés esetén a próbavektorokat csak ott értékeljük ki, ahol sikerült mintát venni. Ezzel rögtön kevesebb eleműek lesznek a próbavektorok, mint ahány darab van belőlük, ezért már nem lehetnek páronként ortogonálisak. Az ortogonalitás sérülése miatt az egyes frekvenciákon lévő jelek más frekvenciákhoz tartozó próbafüggvények esetén is nullától különböző együtthatót eredményeznek, ez egy másik magyarázata a spektrum adatvesztés esetén történő torzulásának.

A próbafüggvények ekvivalens meghatározása a periodikus határfeltételt teljesítő szinusz- és koszinuszfüggvények használata.

3.3.2. Lomb–Scargle-eljárás

A Lomb-Scargle eljárás [20], [21] két próbafüggvényt használ: egy adott frekvenciájú szinusz-, illetve koszinuszfüggvényt. Adatvesztés vagy a periodikus határfeltételt nem teljesítő próbafüggvények esetén (inkoherens mintavétel) a szinusz-, illetve koszinuszfüggvények egymásra és a konstansfüggvényre való ortogonalitása megszűnik, ezért a spektrumbecslő torzul.

A Lomb-Scargle eljárás lépései:

1. A mérési adatok centralizálása:

$$\mathbf{x} := \mathbf{x} - \bar{\mathbf{x}}, \tag{3.46}$$

   ahol a felülvonás az átlagképzés.

2. Egy adott f frekvencián a két próbafüggvény:

$$\phi_c(t) = \cos(2\pi f t), \tag{3.47}$$
$$\phi_s(t) = \sin(2\pi f t), \tag{3.48}$$

a próbavektorok pedig:

$$\phi_c = \begin{bmatrix}\cos(2\pi f t_1) & \dots & \cos(2\pi f t_N)\end{bmatrix}^T, \tag{3.49}$$

$$\phi_s = \begin{bmatrix}\sin(2\pi f t_1) & \dots & \sin(2\pi f t_N)\end{bmatrix}^T, \tag{3.50}$$

ahol $t_1, \dots, t_N$ a mintavételi időpontok.

3. A két próbavektort ortogonalizáljuk a Gram-Schmidt eljárással, és úgy alakítjuk ki a Φ próbamátrixot.

4. Az együtthatóvektor kiszámítása:

$$\mathbf{c} = \Phi^T \mathbf{x}. \tag{3.51}$$

5. A vizsgált frekvencián a modellvektorunk a projekció után:

$$\mathbf{y} = \Phi\mathbf{c} = \Phi\left(\Phi^T\Phi\right)^{-1}\Phi^T\mathbf{x} = \Phi\Phi^T\mathbf{x}, \tag{3.52}$$

ahol az utolsó egyenlőség azért igaz, mert ortogonalizáltuk a próbavektorokat, így a metrikus tenzor az egységmátrix.

6. A frekvenciára eső energiát skaláris szorzattal határozhatjuk meg:

$$E_f = \mathbf{y}^T\mathbf{y} = \left(\Phi\Phi^T\mathbf{x}\right)^T\Phi\Phi^T\mathbf{x} = \mathbf{x}^T\Phi\left(\Phi^T\Phi\right)\Phi^T\mathbf{x} = \mathbf{x}^T\Phi\Phi^T\mathbf{x}. \tag{3.53}$$

7. Ezt az energiát a modellünk szerint egy f frekvenciájú szinuszos jel hordozta, melyből T_s mintavételi időköz mellett sikerült N mintát vennünk. Így az effektív mérési idő $T = NT_s$, ezzel elosztva az energiát megkapjuk a komponensünk teljesítményét:

$$P_f = \frac{E_f}{T} = \frac{\mathbf{y}^T\mathbf{y}}{NT_s}. \tag{3.54}$$

A Lomb–Scargle-eljárás az előbbi lépések végrehajtását jelenti előre meghatározott, ám tetszőleges frekvenciarácson. Mivel az eljárást offline feldolgozásra szokás használni, a frekvenciarács jóval sűrűbb lehet, mint egy szokásosan használt DFT rács. Cserében az eljárással nagyobb számításigény felé kezdtünk elindulni.

Az eljárás gyakorlatilag egy háromparaméteres szinuszillesztést hajt végre, az alábbi függvényt illeszti LS közelítésben a jel mintáira:

$$x(t) = A\cos(2\pi f t) + B\sin(2\pi f t), \tag{3.55}$$

ahol A, B, és f a három változó paraméter. Tehát az illesztett szinuszjel amplitúdója, frekvenciája és fázisa változó, a középértéket nullának feltételezzük.

Az illesztést az alábbi egyenlettel is felírhatjuk:

$$P_{LS}(f) = \frac{1}{2}\left(\frac{\left(\sum_n x_n\cos\left(2\pi f\left(t_n-\tau\right)\right)\right)^2}{\sum_n\cos^2\left(2\pi f\left(t_n-\tau\right)\right)} + \frac{\left(\sum_n x_n\sin\left(2\pi f\left(t_n-\tau\right)\right)\right)^2}{\sum_n\sin^2\left(2\pi f\left(t_n-\tau\right)\right)}\right), \tag{3.56}$$

$$\tau = \frac{1}{4\pi f}\arctan\left(\frac{\sum_n\cos\left(4\pi f t_n\right)}{\sum_n\sin\left(4\pi f t_n\right)}\right), \tag{3.57}$$

ahol τ biztosítja a két próbavektor ortogonalitását.

Az eljárás kiterjeszthető négyparaméteres szinuszillesztésre is (amikor a középérték is tetszőleges), illetve több harmonikus komponens illesztésére is [22].

3.3.3. DCDFT – Date Compensated DFT

A Lomb–Scargle-eljárás nem oldotta meg azt a problémát, hogy a szinusz-, illetve koszinuszfüggvény nem feltétlenül ortogonális a konstansra. Az adatok centralizálása nem oldja meg ezt a problémát, az egyes komponensek erősségét ettől még hibásan mérjük. Ennek kiküszöbölésére a DCDFT eljárás a próbafüggvények közé a konstansfüggvényt is beveszi.

A DCDFT eljárás lépsei:

1. A mérési adatok centralizálása:

$$\mathbf{x} := \mathbf{x} - \bar{\mathbf{x}}. \tag{3.58}$$

2. A három próbafüggvény egy adott f frekvencián:

$$\phi_1(t) = 1, \tag{3.59}$$
$$\phi_c(t) = \cos(2\pi f t), \tag{3.60}$$
$$\phi_s(t) = \sin(2\pi f t). \tag{3.61}$$

3. A mérési időpontok behelyettesítésével kapott próbavektorok:

$$\phi_1 = \mathbf{1}, \tag{3.62}$$
$$\phi_c = \begin{bmatrix}\cos(2\pi f t_1) & \dots & \cos(2\pi f t_N)\end{bmatrix}^T, \tag{3.63}$$
$$\phi_s = \begin{bmatrix}\sin(2\pi f t_1) & \dots & \sin(2\pi f t_N)\end{bmatrix}^T. \tag{3.64}$$

4. A három próbavektor Gram-Schmidt ortogonalizálásával megkapjuk a Φ próbamátrixot.

5. Az együtthatóvektor – mivel a próbamátrix oszlopai ortogonális vektorrendszert alkotnak, a metrikus tenzor egységmátrix –:

$$\mathbf{c} = \begin{bmatrix}c_1 & c_c & c_s\end{bmatrix}^T = \Phi^T \mathbf{x}. \tag{3.65}$$

6. Az együtthatóvektor alapján meghatározható az adott frekvenciára eső energia:

$$E_f = c_c^2 + c_s^2. \tag{3.66}$$

7. Az energiát osztva az effektív mérési idővel megkapjuk a teljesítményt:

$$P_f = \frac{E_f}{T} = \frac{c_c^2 + c_s^2}{N T_s}. \tag{3.67}$$

A DCDFT eljárás – a Lomb–Scargle-eljáráshoz hasonlóan – is az előbbi műveletek egy előre adott, ám tetszőleges frekvenciarácson való elvégzését takarja. A számításigénye még nagyobb, mint a Lomb–Scargle-módszeré, ezt is offline feldolgozásra szokás használni.

Fizikai jelentést tulajdoníthatunk az alábbi módon kiszámított amplitúdónak:

$$A = \sqrt{\frac{2}{N}\left(\mathbf{y}^T\mathbf{y} - (\mathbf{1}^T\mathbf{y})^2\right)}. \tag{3.68}$$

Azaz vesszük a modellfüggvény konstansra ortogonális részének effektív értékét; a $\sqrt{2}$-es szorzó pedig azért szerepel, hogy egy $\sin(2\pi f t)$ jel körülbelül egységnyi amplitúdót eredményezzen.

Ha az adatok elérhetősége hasonló frekvenciával változik, mint az adatok egy jellemző frekvenciája, akkor a Lomb–Scargle-eljárás pontatlan becslést eredményez, míg a DCDFT nem. A konstansfüggvény kihagyása hibát eredményez, kifejezetten akkor, ha több frekvenciára illesztünk modellt egyszerre.

3.3.3.1. Harmonikus szűrés

A DCDFT becslője jellemzően elég pontos ahhoz, hogy azonosítsuk vele a jelben lévő legnagyobb teljesítményű komponenst, és kivonjuk azt a jelből [23]. Így egy iteratív eljárás keretében kereshetünk több szinuszos komponenst, melyeknek nem feltétlenül kell harmonikus viszonyban lenniük egymással. Az eljárás lépései:

1. Domináns komponens keresése DCDFT segítségével, az eredmény az $\mathbf{y}$ modellvektor.
2. Az adatvektorból az azonosított komponens eltávolítása:

$$\mathbf{x} := \mathbf{e} = \mathbf{x} - \mathbf{y}, \tag{3.69}$$

   vagyis a továbbiakban a hibavektorral dolgozunk.
3. A hibavektor elemzése DCDFT-vel, hogy van-e még komponens.
4. Ha van még komponens, levonás és ismétlés.

3.3.3.2. Többfrekvenciás DCDFT

A DCDFT kiterjeszthető olyan módon, hogy egyszerre illesszünk szinuszfüggvényt több különböző frekvenciára. Az eljárás a DCDFT természetes kiterjesztése, $\mathbf{f} = \begin{bmatrix} f_1 & \dots & f_N \end{bmatrix}$ a vizsgált frekvenciák vektora. Az algoritmus:

1. A mérési adatok centralizálása:

$$\mathbf{x} := \mathbf{x} - \bar{\mathbf{x}}. \tag{3.70}$$

2. A próbafüggvények a konstansfüggvény, illetve minden frekvenciához egy-egy szinusz- és koszinuszfüggvény:

$$\phi_1(t) = 1, \tag{3.71}$$
$$\phi_{c,i}(t) = \cos(2\pi f_i t) \quad \forall i \in \{1, \dots, N\}, \tag{3.72}$$
$$\phi_{s,i}(t) = \sin(2\pi f_i t) \quad \forall i \in \{1, \dots, N\}. \tag{3.73}$$

3. A mérési időpontok behelyettesítésével kapott próbavektorok:

$$\phi_1 = \mathbf{1}, \tag{3.74}$$
$$\phi_{c,i} = \begin{bmatrix} \cos(2\pi f t_1) & \dots & \cos(2\pi f t_N) \end{bmatrix}^T \quad \forall i \in \{1, \dots, N\}, \tag{3.75}$$
$$\phi_{s,i} = \begin{bmatrix} \sin(2\pi f t_1) & \dots & \sin(2\pi f t_N) \end{bmatrix}^T \quad \forall i \in \{1, \dots, N\}. \tag{3.76}$$

4. A $2N+1$ próbavektor Gram-Schmidt ortogonalizálásával megkapjuk a Φ próbamátrixot.
5. Az együtthatóvektor:

$$\mathbf{c} = \begin{bmatrix} c_1 & c_{c,1} & c_{s,1} & c_{c,2} & \dots & c_{c,N} & c_{s,N} \end{bmatrix}^T = \Phi^T \mathbf{x}. \tag{3.77}$$

6. Az együtthatóvektor alapján meghatározható az adott f_i frekvenciára eső energia:

$$E_{f,i} = c_{c,i}^2 + c_{s,i}^2. \tag{3.78}$$

7. Az energiát osztva az effektív mérési idővel, megkapjuk a teljesítményt:

$$P_{f,i} = \frac{E_{f,i}}{T} = \frac{c_{c,i}^2 + c_{s,i}^2}{NT_s}. \tag{3.79}$$

Itt is kiszámíthatjuk a (3.68) szerinti amplitúdót, mely itt egy azonos teljesítményt szállító szinuszjel amplitúdóját eredményezi. Ezt nevezhetjük i-frekvenciás amplitúdónak, és jelölhetjük A_i-vel.

3.3.4. CLEANEST

Sajnos, mivel a DCDFT eljárással a keresett szinuszos komponenst nem találjuk meg tökéletesen (az amplitúdót a zaj miatt, míg a frekvenciát a rács miatt sem), ezért harmonikus szűrés esetén a kivonással ezt nem tudjuk teljesen eltüntetni a jelből. Így egy hibát viszünk be, és a bevitt hibák a további iterációk során halmozódnak. Erre a problémára kínál megoldást a CLEANEST algoritmus [19], amely egy iteratív eljárás.

Az algoritmus egy DCDFT végrehajtásával indul, melynek az eredményét kivonjuk a jelből. Végrehajtunk egy második DCDFT-t, a kivonás előtt azonban az első és a második DCDFT által kapott eredményeket mint kezdeti értékeket felhasználva megkeressük az eredeti jel optimális kétszinuszos közelítését, és az így kapott szinuszjelek összegét vonjuk ki az eredeti jelből. Ez után hajtjuk végre a harmadik DCDFT-t, és mindaddig folytatjuk az iterációkat, amíg még elfogadható nagyságú csúcsot kapunk a DCDFT-vel. Amikor már nem kapunk ilyet, az utolsó DCDFT eredményét mint „maradék spektrum” fogadjuk el.

A CLEANEST által adott spektrum valójában két részből tevődik össze: a megtalált diszkrét komponensek, illetve a maradék spektrum. Az algoritmus lépései:

1. $k = 1$ az iterációk száma, K az iterációk maximális száma; i, $\mathbf{d} = \left[\ \right]$ és $\mathbf{f}_d = \left[\ \right]$ a megtalált komponensek száma, amplitúdói és frekvenciái (kezdetben üres vektorok).

2. DCDFT végrehajtása az adatvektoron, a legnagyobb amplitúdójú komponens kiválasztása, ennek a frekvenciája f_1; $i := 1$.

3. Így a megtalált komponensek frekvenciái:

$$\mathbf{f}_d := \left[f_1\right]. \tag{3.80}$$

4. Az újonnan megtalált kiválasztott komponenst valószínűleg mind frekvenciában, mind amplitúdóban hibásan becsüljük. A frekvenciát azért, mert a DCDFT-t egy adott frekvenciarácson hajtjuk végre, az amplitúdót pedig a tetőesés és a zaj miatt. Ezért a megtalált komponensek frekvenciáival mint kezdeti értékkel végrehajtunk egy maximumkeresést. Így a frekvenciabecslőket „rá tudjuk húzni” a valós frekvenciákra, aminek következtében a tetőesést is kiküszöböljük. Az optimalizálási feladat:

   (a) Optimalizálási változó: az $\mathbf{f}_d$ frekvenciavektor, az előbb meghatározott kezdeti értékkel.

   (b) Célfüggvény:

$$A_i \to \max, \tag{3.81}$$

   ahol A_i az i-frekvenciás amplitúdó. Ez – zajmentes esetben – akkor lesz maximális, ha a tetőesést kiküszöböltük, azaz a frekvenciabecslők pontosak.

A célfüggvényt egyszerűsíthetjük, ha elhagyjuk a gyökvonást és a szorzókat:

$$\left(\mathbf{y}^T\mathbf{y} - \left(\mathbf{1}^T\mathbf{y}\right)^2\right) \to \max. \tag{3.82}$$

(c) Kényszerek:

i. A frekvenciák a 0 és a Nyquist-frekvencia között helyezkedhetnek el:

$$0 \le f_j \le 0{,}5 \quad \forall j \in \{1, \ldots, i\}, \tag{3.83}$$

az egyenlőtlenségeket relatív frekvenciában felírva.

ii. A frekvenciák alkossanak növekvő sorozatot:

$$f_j \le f_{j+1} \quad \forall j \in \{1, \ldots, i-1\}. \tag{3.84}$$

Itt azért engedtük meg az egyenlőséget, mert ez az optimalizációs problémák általánosan elterjedt alakja.

Az alkalmazandó optimalizálási módszerről a szakirodalom – tudomásom szerint – nem értekezik. A szimulációs megvalósításban a MATLAB optimalizációs toolbox-át alkalmaztam, mely egy gradiens alapú eljárást használt. Az alkalmazott optimalizáció konvergenciáját vizsgálni kell, különösen közeli frekvenciák azonosítása esetén. [19] szerint az eljárás képes megtalálni egymáshoz közel eső frekvenciákat, így szétválasztva interferáló komponenseket. Ez a képesség azonban az adott adatvesztéstől és optimalizációs algoritmustól is függhet, ennek vizsgálata nem célja a dolgozatnak.

Az optimalizálás eredménye a maximális amplitúdót biztosító $\mathbf{f}_d$ frekvenciavektor. Amennyiben ebben vannak azonos frekvenciák, akkor az optimalizálás valószínűleg azért állt meg, mert elértük a kényszerek biztosította határt, nem pedig azért, mert (lokális) maximumot találtunk. Ekkor minden frekvenciából csak egyet hagyunk meg ($\mathbf{f}_d := \text{különböző}(\mathbf{f}_d)$), és ennek megfelelően módosítjuk a megtalált frekvenciák számát ($i := \dim \mathbf{f}_d$). Amennyiben ilyen beavatkozásra volt szükség, akkor $k < K$ esetén $k := k + 1$ és újraindítjuk a maximumkeresést, a módosított frekvenciavektorral, $k \ge K$ esetén pedig a következő lépéssel folytatjuk.

5. A megtalált frekvenciákhoz tartozó komponensek amplitúdóinak kiszámítása i-frekvenciás DCDFT segítségével, az eredmény $\mathbf{d}$. Eközben előállt az új $\mathbf{y}$ modellvektor.

6. Az új hibavektor számítása:

$$\mathbf{e} := \mathbf{x} - \mathbf{y}, \tag{3.85}$$

majd DCDFT végrehajtása a hibavektoron, az eredmény a maradék spektrum.

7. Amennyiben a maradék spektrumban nem találunk más komponenst (elégséges nagyságú csúcsot), leállunk, a CLEANEST eredménye a megtalált komponensek és a maradék spektrum. Leállunk továbbá, ha $k \ge K$ vagy teljesítjük az alkalmazási leállási feltételt.

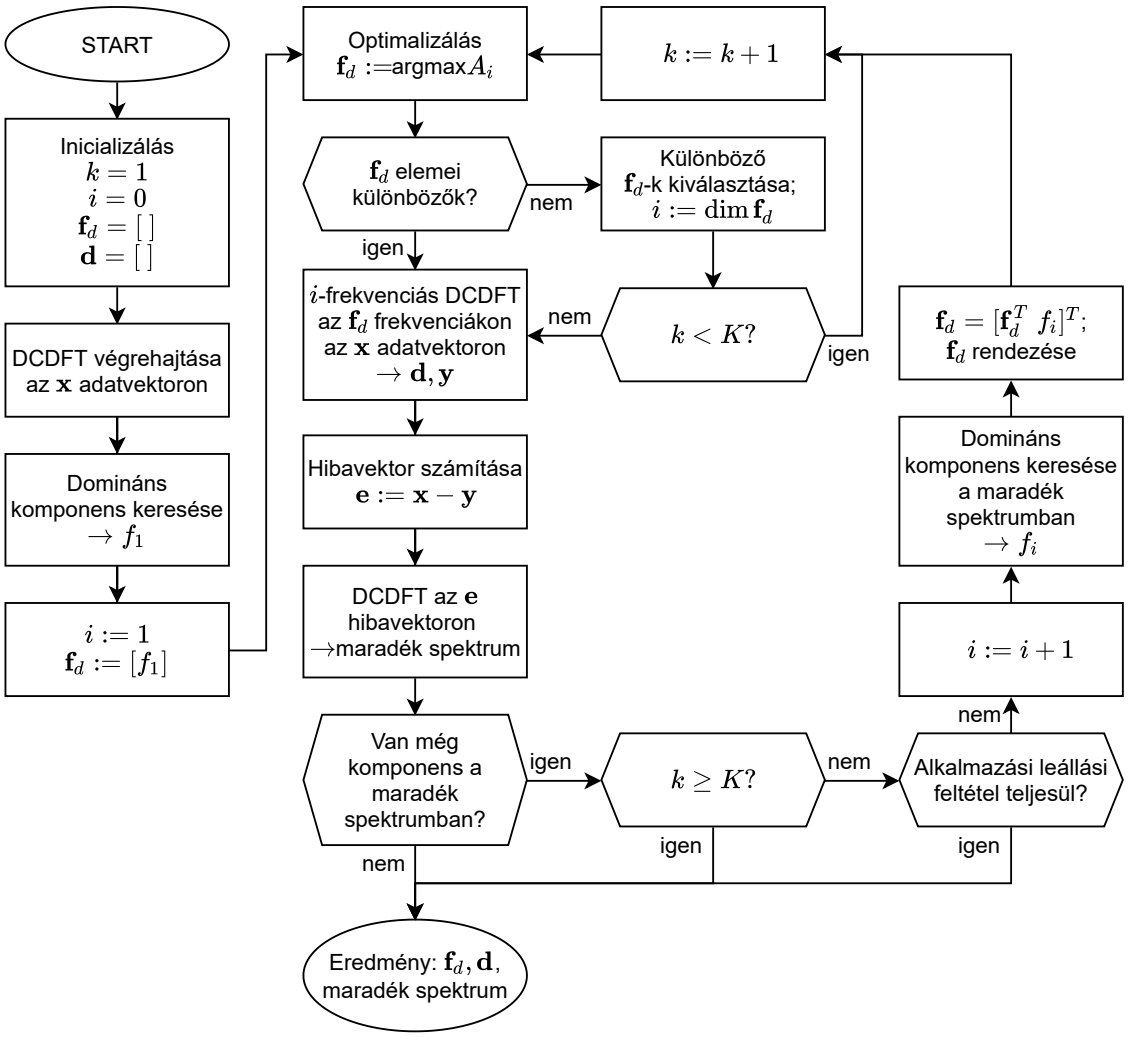
Az alkalmazási leállási feltétel az adott alkalmazás konkrét céljának teljesülését jelenti, mely többféle kritériumként jelentkezhet: adott amplitúdónál nagyobb komponensek megtalálása, a hibavektor energiája legyen az adatvektor energiájának egy adott százaléka alatt, a modellvektor energiája legyen az adatvektor energiájának adott százaléka fölött, már nem tudunk lényegesen javítani a hibavektor vagy a modellvektor energiáján, kifutottunk a számítási időből, ….

Amennyiben a maradék spektrum tartalmaz még komponenst, akkor ebből a legnagyobb kiválasztása, a megtalált komponensek számának növelése ($i := i + 1$), a komponens frekvenciája f_i, a frekvenciát felvesszük a megtalált komponensek frekvenciái közé:

$$\mathbf{f}_d := \begin{bmatrix} \mathbf{f}_d^T & f_i \end{bmatrix}^T, \tag{3.86}$$

majd $\mathbf{f}_d$-t növekvő sorba rendezzük. Növeljük az iterációk számát ($k := k + 1$), majd a 4. lépéssel folytatjuk.

Az eljárás lépéseit szemléltei a 3.12. ábra.



3.12. ábra. A CLEANEST eljárás folyamatábrája

Az eljárás elején nem feltétlenül kell az egyetlen legnagyobb amplitúdójú komponens frekvenciájával indítani az optimalizációt, használható több is (például egy adott amplitúdó fölött; annyit, amennyit a számítási kapacitás elbír). Ha néhány csúcs hamis volt (csak az adatvesztés terméke), akkor ezek amplitúdója nagy valószínűséggel elhanyagolhatónak fog adódni. Hasonlóan, a maradék spektrumból is bevehetünk egyszerre több komponenst, ezzel csökkentve az optimalizációk szükséges számát.

Az optimalizált frekvenciákon való DCDFT elvégzése után a túl kicsi amplitúdójú komponenseket az optimalizált frekvenciák közül eltávolíthatjuk, ezzel csökkentve a szükséges számításigényt.

Ez az eljárás lokális maximumot keres, ezért lehetséges, hogy így nem a legjobb i frekvenciát tartalmazó modellt kapjuk eredményül. Elméletileg megtehető, hogy a teljes $0 \leq f_1 < \cdots < f_i \leq 0{,}5$ frekvenciateret megvizsgáljuk, ennek számításigénye azonban annyira magas, hogy ez az irány csupán kivételes esetekben indokolt.

3.3.5. SLICK

A CLEANEST eljárás során nem használtuk ki, hogy a hibavektor ortogonális a modellvektorra. Ezt megtehetjük, ha a maradék spektrumot nem a hibavektor DCDFT-vel való feldolgozásával állítjuk elő. A SLICK eljárás [19] majdnem megegyezik a CLEANEST-tel, csupán a 6. lépését az alábbira módosítja:

- Az új hibavektor számítása:
$$\mathbf{e} := \mathbf{x} - \mathbf{y}. \tag{3.87}$$
- Egy adott frekvenciarács f_m elemeire (a frekvenciákra, ahol a DCDFT-t végrehajtanánk):
  - A megtalált komponensek frekvenciái és az aktuális f_m érték alkotják a vizsgált frekvenciák vektorát:
$$\mathbf{f}_S = \begin{bmatrix} \mathbf{f}_d^T & f_m \end{bmatrix}^T. \tag{3.88}$$
  - $i+1$-frekvenciás DCDFT végrehajtása az $\mathbf{f}_S$ frekvenciákon, az $\mathbf{x}$ adatvektoron.
  - Az f_m frekvenciához tartozó energia (teljesítmény) lesz a maradék spektrum értéke ezen a frekvencián.

Vagyis felveszünk egy futó frekvenciát, és azt vizsgáljuk, hogy mekkora energia esik a futó frekvenciára úgy, hogy mellette meghagytuk a megtalált komponensek frekvenciáit. Mivel a DCDFT-ben a próbavektorokat ortogonalizáljuk, ezzel kihasználtuk, hogy a hibavektor ortogonális a modellvektorra.

3.3.6. Harmonikus DCDFT

Amennyiben valamilyen periodikus jel jelenlétét feltételezzük a vizsgált adatsorban, akkor érdemes lehet az alapharmonikus és a felharmonikusok becslését egyszerre elvégezni. Egy adott frekvenciájú periodikus jel komponenseinek becslése könnyen elvégezhető a többfrekvenciás DCDFT algoritmus végrehajtásával az alábbi frekvenciavektoron:

$$\mathbf{f} = \begin{bmatrix} f_1 & 2f_1 & 3f_1 & \dots & Nf_1 \end{bmatrix}^T. \tag{3.89}$$

Vagyis a frekvenciavektort az alapharmonikus és a felharmonikusok frekvenciái alkotják. A figyelembe vett frekvenciák számát korlátozhatjuk egyrészt a Nyquist-frekvenciával, másrészt adhatunk egy fix korlátot a figyelembe vett felharmonikusok számára. Kivételes eseteket leszámítva jellemzően elegendő az első néhány tíz komponenst vizsgálni.

Amennyiben egy ismert frekvenciájú periodikus jel komponenseit szeretnénk becsülni, az előbbi vizsgálatot elegendő arra a frekvenciára elvégezni. Ekkor megkapjuk a komponensek nagyságát és frekvenciáját. Amennyiben a vizsgálatot egy frekvenciarácson szeretnénk végrehajtani, célszerű egy olyan indikátort eredményül választani, amely az összes vizsgált frekvencia által hordozott energiát/teljesítményt összesítve tartalmazza. A (3.68) egyenlet szerinti amplitúdó megfelelő ilyen indikátor. Ekkor ezt az amplitúdót a frekvenciarács minden pontján kiszámítva, majd a maximumhoz tartozó frekvenciát kiválasztva megkapjuk a periodikus jel becsült frekvenciáját, és ezen a frekvencián már úgy tudjuk elvégezni a harmonikus DCDFT-t, hogy az egyes komponensek nagyságára vagyunk kíváncsiak.

Az előbbi eljárás során figyelembe kell venni azt, hogy egy f_a frekvenciájú periodikus jel nemcsak az f_a frekvenciájú harmonikus DCDFT végrehajtásánál eredményezi a komponensek becslését, hanem az $\frac{f_a}{2}$, $\frac{f_a}{3}$, ... frekvenciák esetén is. Ez a jelenség az előbbi eljárás során egy periodikus jel jelenlétét az alapharmonikus frekvencián kívül – hibásan – annak szubharmonikusaira is diagnosztizálja. Mivel ekkor az alapharmonikus – a várhatóan legnagyobb amplitúdójú komponens – is járulékot ad az amplitúdóba, ezeknek a hamis csúcsoknak a nagysága jelentősen megközelítheti a valós csúcs nagyságát.

Ennek a hibának a kiküszöbölésére egy primitív módszer az, hogy a kiszámított amplitúdót érvénytelenítjük, amennyiben az alapharmonikus – a többi harmonikushoz képest – elhanyagolható. Ez a módszer a $\sin(2\pi 2 f_0 t) + \sin(2\pi 3 f_0 t)$ jellegű, f_0 alapfrekvenciájú periodikus jeleket hibásan elveti.

Egy jobb érvényességvizsgálati módszert kaphatunk, ha figyelembe vesszük, hogy amikor egy f_a alapfrekvenciájú periodikus jelet $\frac{f_a}{N}$ alapfrekvenciájú jelként vizsgálunk, akkor – zajmentes esetben – csak minden N. harmonikuson kapunk nullától különböző amplitúdót. Így ha vesszük azokat a harmonikus rendszámokat, ahol értelmes nagyságú komponenst találtunk, akkor ezek N többszörösei lesznek. Ezért a kapott rendszámok legnagyobb közös osztója egytől különböző (N) lesz. Amennyiben a kapott rendszámok legnagyobb közös osztójára egyet kapunk, akkor a vizsgált alapfrekvencia helyesnek tekinthető.

A szubharmonikusokon való hamis megjelenésen kívül a felharmonikusok is adnak hamis csúcsokat. Ez a hiba is kiküszöbölhető: miután egy frekvencián megtaláltunk egy periodikus jelet, ezt kivonhatjuk az eredeti jelből, ezzel a DCDFT harmonikus szűréséhez hasonló eljárást megvalósítva. Másik lehetőség, hogy miután a frekvenciarácson elvégeztük a harmonikus DCDFT-t, a kiszámított amplitúdókon a kisebb frekvenciák felől végigmegyünk, és ahol találunk egy jelet, annak a frekvenciának a többszörösein lévő csúcsokat lenullázzuk.

Ha egy adott frekvenciát vizsgálunk, akkor könnyen előfordulhat, hogy egyes komponensek amplitúdóira elhanyagolható értéket kapunk – például egy 50% kitöltésű négyszögjel esetén minden páros komponensre. Ezért a további feldolgozás esetén a túl kis amplitúdójú komponenseket érdemes lehet kizárni az eredményből.

Az egyszerre vizsgált komponensek számának nemcsak a számításigényre van jelentős hatása – hiszen az próbavektorok ortogonalizálása számításigényes feladat –, hanem a zajszintre is. Ha az eljárást egy frekvenciarácson hajtjuk végre fehér zaj bemeneten, lépcsőzetes zajszint alakul ki. Ennek az oka egyszerűen az, hogy egy adott frekvenciához – ahol egyébként nincs jelkomponens – annak felharmonikusairól is hozzászámítjuk a zajteljesítményt. Így kis alapfrekvenciánál – ahol az összes vizsgált felharmonikus frekvencia még a Nyquist-frekvencia alatt van, akkor az adott frekvencián közelítőleg az ott lévő zajteljesítmény N-szerese jelenik meg. Ahogy növekszik az alapfrekvencia, egyre több felharmonikus kerül a Nyquist-frekvencia fölé, ezzel a zajszintre vonatkozó „szorzót” csökkentve.

A harmonikus DCDFT eljárással eddig nem találkoztam az irodalomban, a dolgozat keretei között végzett megvalósítása – a vezérelt adaptív Fourier-analizátorhoz hasonlóan – „proof-of-concept” jellegű. Így nem cél az eljárás teljes finomítása, csupán az ötlet működőképességének ellenőrzése.

3.3.6.1. Multiharmonikus DCDFT

Ahogy az „egyszerű” DCDFT esetén is lehetőség van több frekvenciára való kiterjesztésre, a harmonikus DCDFT esetén is vizsgálhatunk egyszerre több alapfrekvenciát. Ezt nevezhetjük multiharmonikus DCDFT-nek.

A multiharmonikus DCDFT esetén arra kell ügyelni, hogy az egyes alapfrekvenciák közül egyik kettő se legyen egymással harmonikus viszonyban. Másképp megfogalmazva, bármelyik két alapfrekvencia hányadosa legyen irracionális.

3.3.6.2. Harmonikus CLEANEST és harmonikus SLICK

A harmonikus DCDFT ugyanúgy alkalmas iteratív eljárás kidolgozására, mint az egyszerű DCDFT. A harmonikus CLEANEST és a harmonikus SLICK eljárások lépései ugyanazok, mint „egyszerű” változataiké, a következő különbségekkel:

- A frekvenciavektor, amit az optimalizáláshoz használunk, csak az alapfrekvenciákat tartalmazza. (Ezért a megtalált komponensek frekvenciáinak vektora itt különböző lesz az optimalizálási frekvenciáktól.)
- A kezdeti iterációs lépés során javasolt csupán egy alapfrekvenciát kiválasztani, a kezdeti harmonikus DCDFT során a legnagyobb amplitúdót eredményezőt. Hasonlóan akkor is, amikor újabb frekvenciák bevételéről döntünk a maradék spektrumból. A további frekvenciák bevétele már egyszerű DCDFT alapján is történhet a számítás gyorsítása érdekében.
- Miután kiléptünk az iterációból, a maradék spektrumot célszerű a hibavektorból az egyszerű DCDFT segítségével újraszámítani. Így a harmonikus DCDFT okozta lépcsőzetes zajszint elkerülhető.

4. fejezet

Elméleti összehasonlítás

A fejezetben elméleti oldalról hasonlítjuk össze az előbbiekben megismert eljárásokat. Szót ejtünk az eljárások számítás- és memóriaigényéről, a vizsgált frekvenciákról, az eljárások viselkedéséről inkoherens mintavétel esetén, potenciális numerikus problémákról és a valós idejű implementáció lehetőségéről.

4.1. Számítás- és memóriaigény

Az egyes eljárások számításigénye alapvetően meghatározza az alkalmazási területüket. Egy offline feldolgozás esetén előfordulhat, hogy lényegében tetszőleges számítási kapacitás rendelkezésre áll, adott esetben akár napokat is hajlandóak vagyunk várni, míg egy pontos eredmény elkészül. Ellenben egy beágyazott, valós idejű rendszerben szigorú határidőt kell betartani, és a számítási kapacitás is erősen korlátozott lehet, ezek komoly megkötéseket támasztanak az alkalmazott algoritmussal szemben.

A számítás- és memóriaigényről tett megfontolások során az összehasonlíthatóság kedvéért feltételezzük, hogy adott egy L mintából álló regisztrátum, melyet az adott eljárással szeretnénk feldolgozni. A frekvenciarács elemszámát N-nel, az egyszerre vizsgált frekvenciák számát F-fel jelöljük. Az adatelérhetőségi arány a regisztrátumban μ, ez a nem egyenletes mintavétel eljárásainál lesz lényeges, mivel ezeket az eljárásokat csak az elérhető mintákon hajtjuk végre.

A DFT-alapú eljárások esetén – mivel a DFT-t megtartottuk a spektrumszámításra – a számítás hatékonyságának növeléséhez alkalmazható az FFT. Ezen kívül még egyszerű kiegészítő műveleteket (előfeldolgozás, PSD számítás, átlagolás) végzünk. Egy DFT-blokk FFT-vel való feldolgozása $\mathcal{O}(N \log(N))$ lépést igényel. A további feldolgozási műveletek a blokkban $\mathcal{O}(N)$ művelettel elvégezhetők (blokkosítás előtti előfeldolgozás esetén pedig az egész regisztátumra $\mathcal{O}(L)$ lépés alatt). L minta esetén $\mathcal{O}\left(\frac{L}{N}\right)$ DFT-blokkunk van, így a számításigény $\mathcal{O}(L \log(N))$.

A DFT-alapú eljárások memóriaigénye – ha csak a DFT-t tekintjük – a DFT hosszával arányos. Mivel most L mintánk van, ezeket el kell tárolni, ezért a memóriaigény legalább $\mathcal{O}(L)$. Valós idejű feldolgozás esetén a blokkosítás utáni előfeldolgozást alkalmazó eljárások memóriaigénye csupán $\mathcal{O}(N)$. Ha olyan interpolációt alkalmazunk, amely az üres sorozatokat egyben kezeli – előfeldolgozás a blokkosítás előtt –, akkor elméletileg tetszőlegesen sok minta kimaradhat egymás után; adott adatvesztések mellett az üres sorozatok hossza a DFT-blokk többszöröse is lehet, ez a valós idejű feldolgozást kétségessé teszi.

A modellalapú eljárások esetén mintáról mintára frissíteni kell a modellt, ez egy L-es szorzót jelent. Mind az RBO, mind az AFA esetén a legtöbb számítást igénylő műveletek az egyszerű vektorműveletek, a skaláris szorzás, illetve vektor szorzása diagonálmátrix-

szal, így mintánként $\mathcal{O}(F)$-es lépésszám szükséges (természetesen az AFA esetén nagyobb szorzóval). Tehát az eredő számításigény a modellalapú eljárások esetén $\mathcal{O}(LF)$.

A modellalapú eljárások esetén szükség van néhány F elemű tömbre: állapotváltozók, ki- és bemeneti szorzók, rendszermátrix főátlója. Ennél fogva a memóriaigény $\mathcal{O}(F)$ – lenne online feldolgozás esetén. Mivel azonban a bemeneti mintákat el kell tárolni, a memóriaigény $\mathcal{O}(L)$.

A nem egyenletes mintavétel eljárásaiban csak azokkal a mintákkal dolgozunk, amelyek elérhetők. A Lomb-Scargle eljárás során egy adott frekvencia esetén elő kell állítani a két próbavektort az elérhető minták időpontjain, ez $\mathcal{O}(\mu L)$ művelet. Ezek ortogonalizálása, az együtthatóvektor és az energia/teljesítmény további $\mathcal{O}(\mu L)$ lépést igényelnek. Mindezt a frekvenciarács összes frekvenciáján el kell végezni, így a számításigény $\mathcal{O}(\mu LN)$. A memóriaigény pedig – a próbamátrix miatt – $\mathcal{O}(\mu L)$.

Az adott frekvenciarácson végrehajtott DCDFT számításigény szempontjából lényegében abban különbözik a Lomb–Scargle-eljárástól, hogy kettő helyett három próbavektort alkalmaz. Ezért a számításigénye $\mathcal{O}(\mu LN)$, csupán nagyobb szorzóval. Hasonló okokból a memóriaigénye is $\mathcal{O}(\mu L)$.

Ha többfrekvenciás DCDFT-t hajtunk végre, azt nem egy frekvenciarácson tesszük, hanem az F darab vizsgált frekvencián egyszerre. Ezért a próbavektorok generálása és az együtthatóvektor számítása $\mathcal{O}(\mu LF)$ művelet, viszont az ortogonalizációjuk $\mathcal{O}\left(\mu LF^2\right)$ lépést igényel. Ezért a számításigény $\mathcal{O}\left(\mu LF^2\right)$. A memóriaigényt nagyságrendileg a próbamátrix adja meg, ezért az $\mathcal{O}(\mu LF)$.

A CLEANEST eljárás esetén először egyszerű DCDFT-t, majd két- három, és így tovább frekvenciásat optimalizálunk maximális amplitúdóra. Amennyiben mindegyik DCDFT-ből csupán egyet kellene végrehajtani, akkor ez $\mathcal{O}\left(\mu LF^3\right)$ lépést igényelne, itt F az utolsó iterációban az illesztett frekvenciák száma. Jelöljük I-vel az egy iteráció során történő DCDFT-számítások korlátját (ilyet jellemzően tartalmaznak az optimalizációs szoftverek), és tegyük fel, hogy egyik iterációban sem találtuk meg a maximumot, vagyis az előbbi korlát miatt állt le az optimalizálás. Így a számításigény $\mathcal{O}\left(\mu LF^3 I\right)$. A memóriaigény a próbamátrix miatt $\mathcal{O}(\mu LF)$. Megjegyzendő, hogy egy adott optimalizációs algoritmus számítás-és memóriaigénye ettől eltérő lehet.

A SLICK esetében az iterációk végén a maradék spektrumot nem egy egyszerű DCDFT-vel, hanem egy többfrekvenciással határozzuk meg a frekvenciarácson. Ez minden iterációban hozzáad még N többfrekvenciás DCDFT-számítást, így a számításigény $\mathcal{O}\left(\mu LF^3(I+N)\right)$. A memóriaigény szintén $\mathcal{O}(\mu LF)$.

A harmonikus DCDFT-ben F-frekvenciás DCDFT végzünk egy frekvenciarácson. Ezért a számításigény $\mathcal{O}\left(\mu LF^2 N\right)$. A memóriaigény a próbamátrix miatt $\mathcal{O}(\mu LF)$. Multiharmonikus DCDFT esetén – K alapfrekvencia együttes vizsgálatakor – KF a vizsgált frekvenciák száma, ezért a számításigény $\mathcal{O}\left(\mu LK^2F^2N\right)$, míg a memóriaigény $\mathcal{O}(\mu LKF)$.

Jelölje K a harmonikus CLEANEST és a harmonikus SLICK eljárások során a végzett iterációk számát (mivel mindegyik iterációban csak egy újabb alapfrekvenciát veszünk be, ez a multiharmonikus DCDFT-nél bemutatott K-val analóg). Így az iterációk során 1, 2, ..., K különböző periodikus jelet keresünk, mindegyiket legfeljebb F komponenssel. Ezek egyszeri számítás mellett összesen $\mathcal{O}\left(\mu LK^3F^3\right)$ műveletet eredményeznénk, figyelembe vége az egy iteráción belüli I korlátot is, eddig $\mathcal{O}\left(\mu LK^3F^3I\right)$ lépésünk van. Továbbá, minden iterációban a harmonikus DCDFT-t kiszámítjuk a maradék spektrumon. ez újabb $\mathcal{O}\left(\mu LF^2N\right)$. Ezt a mind a harmonikus CLEANEST, mind a harmonikus SLICK estén megtesszük, a különbség, hogy a CLEANEST esetén egyetlen harmonikussal számolunk ($\mathcal{O}\left(\mu LF^2N\right)$ művelet iterációnként), míg a SLICK-nél az eddig megtalált frekvenciákhoz veszünk hozzá egy futót ($\mathcal{O}\left(\mu LK^2F^2N\right)$ a K. iterációban). Ezért a harmonikus CLEANEST $\mathcal{O}\left(\mu LK^3F^3I\right)$, míg a harmonikus SLICK $\mathcal{O}\left(\mu LK^3F^3(I+N)\right)$ számításigényű. A memóriaigény mindkét eljárás esetén a próbamátrixok miatt $\mathcal{O}(\mu LKF)$.

A 4.1. táblázat összefoglalja a hardverigényről tett megállapításainkat.

Eljárás	**Számításigény**	**Memóriaigény**
DFT-alapú, előfeldolgozás blokkosítás után	$\mathcal{O}\left(L\log\left(N\right)\right)$	$\mathcal{O}\left(L\right)$
DFT-alapú, előfeldolgozás blokkosítás után	$\mathcal{O}\left(L\log\left(N\right)\right)$	$\mathcal{O}\left(L\right)$
RBO	$\mathcal{O}\left(LF\right)$	$\mathcal{O}\left(L\right)$
AFA	$\mathcal{O}\left(LF\right)$	$\mathcal{O}\left(L\right)$
Lomb-Scargle	$\mathcal{O}\left(\mu LN\right)$	$\mathcal{O}\left(\mu L\right)$
DCDFT frekvenciarácson	$\mathcal{O}\left(\mu LN\right)$	$\mathcal{O}\left(\mu L\right)$
Többfrekvenciás DCDFT	$\mathcal{O}\left(\mu LF^2\right)$	$\mathcal{O}\left(\mu LF\right)$
CLEANEST	$\mathcal{O}\left(\mu LF^3I\right)$	$\mathcal{O}\left(\mu LF\right)$
SLICK	$\mathcal{O}\left(\mu LF^3\left(I+N\right)\right)$	$\mathcal{O}\left(\mu LF\right)$
Harmonikus DCDFT	$\mathcal{O}\left(\mu LF^2N\right)$	$\mathcal{O}\left(\mu LF\right)$
Multiharmonikus DCDFT	$\mathcal{O}\left(\mu LK^2F^2N\right)$	$\mathcal{O}\left(\mu LKF\right)$
Harmonikus CLEANEST	$\mathcal{O}\left(\mu LK^3F^3I\right)$	$\mathcal{O}\left(\mu LKF\right)$
Harmonikus SLICK	$\mathcal{O}\left(\mu LK^3F^3\left(I+N\right)\right)$	$\mathcal{O}\left(\mu LKF\right)$

4.1. táblázat. Összehasonlítás hardverigény szempontjából

4.2. Frekvenciák, inkoherens mintavétel

Összehasonlíthatjuk az eljárásokat az alapján is, hogy velük milyen frekvenciákon vizsgálhatjuk a jeleket. Ez magában foglalja a frekvenciafelbontást, de több annál: a vizsgált frekvenciák változhatnak is, ahogy a feldolgozó eljárást kiértékeljük. Itt beszélhetünk arról is, hogy az eljárások működőképesek-e inkoherens mintavétel esetén. Ez a gyakorlatban fontos szempont, hiszen valószínűleg a mintavételi frekvenciát nem tudjuk pontosan a jelhez igazítani.

A DFT-alapú eljárások esetén a DFT pontszámából egyértelműen következik a vizsgált frekvenciák értéke és a frekvenciafelbontás. A frekvenciarácsot csupán a DFT pontszámával tudjuk befolyásolni, így azt fixnek tekinthetjük. Továbbá, a vizsgált frekvenciák az időben előrehaladva nem változhatnak, ezért a frekvenciarács statikus. A DFT-alapú eljárások esetén az inkoherens mintavételt a hagyományos módon – ablakozással – tudjuk kezelni.

Az RBO és az AFA estén a vizsgált frekvenciákat a rezonátorpozíciók határozzák meg. Az RBO esetén ezeket előre meg kell adni, viszont tetszőleges frekvenciák beállíthatók. Ezek a frekvenciák az algoritmus futása alatt változatlanok, így egy tetszőleges, statikus frekvenciaráccsal dolgozunk. Az AFA használatakor a frekvenciák a futás során tetszőlegesen változhatnak, ezért itt dinamikus frekvenciarácsot használunk. Az RBO pontos működéséhez szükséges a mért jel frekvenciájának előzetes ismerete, hogy a rezonátorokat rá tudjuk állítani. Az AFA esetén erre nincs szükség, a helyes beállítást elvégzi magától.

A Lomb-Scargle és a DCDFT esetén a vizsgált frekvenciák tetszőlegesek, bármilyen sűrű – akár egyenetlen – frekvenciarácson futtathatjuk az algoritmusokat. A rács az algoritmus futása alatt nem változik. Amennyiben találunk egy csúcsot, a helyének és a nagyságának a becslését a környezetében egy finomabb frekvenciarácson való vizsgálattal pontosíthatjuk.

A CLEANEST és a SLICK eljárások esetén a kezdeti és a maradék spektrum kiszámításánál használt frekvenciarács tetszőleges, de statikus. Az optimalizáció során a frekvenciák természetesen változnak, ez biztosítja az inkoherens mintavétel kezelését.

A harmonikus DCDFT, CLEANEST és SLICK eljárások ilyen tekintetben ugyanolyanok, mint az „egyszerű" társaik. A vizsgált frekvenciákról és az inkoherens mintavétel kezeléséről tett megállapításainkat a 4.2. táblázat foglalja össze.

Eljárás	**Frekvenciarács**	**Inkoherens mintavétel kezelése**
DFT-alapú	fix, statikus	ablakozással
RBO	tetszőleges, statikus	előzetes ismerettel
AFA	dinamikus	adaptációval
Lomb-Scargle	tetszőleges, statikus	frekvenciarács finomításával
DCDFT	tetszőleges, statikus	frekvenciarács finomításával
CLEANEST	dinamikus (komponensek) + tetszőleges, statikus (maradék)	optimalizációval
SLICK	dinamikus (komponensek) + tetszőleges, statikus (maradék)	optimalizációval
Harmonikus DCDFT	tetszőleges, statikus	frekvenciarács finomításával
Harmonikus CLEANEST	dinamikus (komponensek) + tetszőleges, statikus (maradék)	optimalizációval
Harmonikus SLICK	dinamikus (komponensek) + tetszőleges, statikus (maradék)	optimalizációval

4.2. táblázat. Az eljárások összehasonlítása a frekvenciarács és az inkoherens mintavétel kezelése szerint

4.3. Numerikus problémák

Az egyes algoritmusok implementálásakor az alkalmazott platformtól függően felléphetnek numerikus problémák. Az alábbiakban ezeket a potenciális numerikus problémákat vesszük számba.

A DFT-alapú eljárások esetén a DFT-t FFT használatával szoktuk kiszámítani. Ahhoz, hogy az FFT optimális legyen, ezt az adott platformra optimalizálni kell. Mivel az FFT a jelfeldolgozás egy alapvető eszköze, amikor DFT-alapú eljárást szeretnénk implementálni, akkor az FFT-t mint könyvtári rutint érhetjük el, amelyről feltételezhetjük, hogy numerikus problémáktól mentes.

Az RBO esetén az állapotváltozók ismételt forgatása okozhatna problémákat, hiszen az ismételt szorzások miatt a kvantált együtthatókból következő hibák halmozódnak. Ezt a jelenséget a visszacsatolással kiküszöböljük, a hibajel ezt az eltérést is kompenzálja. Mivel az elveszett minták feldolgozása során a hibajelet kinullázzuk, ekkor ez a kompenzáció nem történik meg. Így ha hosszú üres sorozatok vannak, akkor közben felléphet ez a hibajelenség. Az állapotváltozók amplitúdó+fázis alakban való tárolása pedig az elérhető minták melletti hibajel alkalmazását teszi nehézkessé. Ekkor a struktúrát megvalósíthatjuk álló állapotváltozók és forgó együtthatók segítségével (mint az AFA-t). Az együtthatókat ilyenkor amplitúdó+fázis alakban kell tárolni, mivel ezek értékét nem szabályozzuk, csak vezéreljük.

Az AFA-t álló állapotváltozókkal és forgó együtthatókkal mutattuk be. Az együtthatókat amplitúdó+fázis alakban célszerű tárolni (vagy időnként normálni kell őket).

A nem egyenletes mintavétel eljárásaiban kritikus lépés a próbamátrix ortogonalizációja. Ezt nagy pontossággal kell elvégezni, különben a kapott mátrix oszlopai nem ugyanazt a teret fogják kifeszíteni, mint az eredeti próbavektorok. Az adatvesztéstől függően előfordulhatnak közel párhuzamos próbavektorok, melyek pontos kezeléséhez nagy számábrázolási pontosság vagy speciális algoritmus szükséges. Amennyiben pedig nem ortogonalizáljuk a próbavektorokat, akkor a pszeudoinverz számítása során ütközhetünk numerikus problémákba, különösen nagyobb számú próbavektor esetén.

Az iteratív eljárások – CLEANEST, SLICK – során a közel párhuzamos vektorok problémája előkerülhet, ha az optimalizáció úgy áll le, hogy két megtalált frekvencia (szinte) azonos. Továbbá, adott jelek, adatvesztések és optimalizációs eljárások mellett a konvergencia problémás lehet. Ezért is kell ilyenkor újraindítani az optimalizációt. A multiharmonikus DCDFT esetén a különböző alapfrekvenciák egyes felharmonikusai kerülhetnek egymáshoz igen közel, ezzel problémákat okozva.

A numerikus problémákról tett megállapításokat a 4.3. táblázat tartalmazza.

Eljárás	**Numerikus problémák okai**	**Okozott nehézség**
DFT-alapú	nincs	nincs
Modellalapú	komplex számok ismételt szorzása	kisebb
Nem egyenletes mintavétel	ortogonalizáció, pszeudoinverz, konvergencia	nagyobb

4.3. táblázat. Az eljárások összehasonlítása a numerikus problémák szerint

4.4. Összegzés, valós idejű megvalósítás

Most, hogy több szempontból vizsgáltuk az eljárásokat, érdemes az eddig tett megállapításinkat összegezni. Így választ adhatunk arra a kérdésre, hogy mely eljárásokat lehet valós időben megvalósítani.

A vizsgált eljárások közül a DFT-alapúak alkotják a legkisebb számítás- és memóriaigényű csoportot, továbbá nincsenek kitéve numerikus problémáknak. Hátrányuk, hogy csak egy fix, statikus frekvenciarácsot kínálnak. A valós idejű megvalósítás elsősorban attól függ, hogy milyen előfeldolgozást alkalmazunk. Ha blokkosítás utánit, akkor egyértelműen lehetséges a valós idejű implementáció. Ha a blokkosítás előtt történik az előfeldolgozás, akkor csak a nulladrendű tartó használatával lesz valós idejű az eljárás.

A rezonátoros struktúra számításigénye már egy fokkal magasabb, de még mindig alacsonynak tekinthető. Ezért cserében azt kapjuk, hogy a frekvenciarácsunk tetszőleges lesz, de még mindig statikus. Az algoritmus valós idejű megvalósításának komoly akadálya nincs, viszont adott hardverrel kevesebb frekvencia becslését tudjuk elvégezni, mint DFT segítségével.

Az adaptív Fourier-analizátor esetén a frekvenciák is dinamikusan változhatnak, némi extra számításért cserében. Ezzel a hardverigény még az alacsony kategóriában marad. Numerikus problémák tekintetében hasonló az RBO-hoz, és valós idejű megvalósítása is lehetséges.

A Lomb-Scargle és a DCDFT eljárások egy újabb lépcsőt jelentenek hardverigény szempontjából, ezeket a közepes kategóriába sorolhatjuk. Tetszőleges, statikus frekvenciarácson elvégezhetők, és mivel egyszerre csak egyetlen frekvenciát vizsgálnak, az ortogonalizációból következő numerikus problémák valószínűsége elhanyagolható. A valós idejű megvalósítást elsősorban a számításigény korlátozza, de korlátozott számú vizsgált frekvencián elképzelhető.

A hardverigény következő lépcsőjén helyezkednek el CLEANEST és a SLICK eljárások. Ezeken egyrészt egy dinamikus frekvenciarácsot használunk a komponensek keresésére, másrészt egy tetszőleges, statikus rácsot a maradék spektrum kiszámítására. Az ortogonalizáció során előfordulhatnak numerikus problémák, a valós idejű megvalósítás a magas számításigény miatt kétséges.

A harmonikus DCDFT, CLEANEST és SLICK eljárások tulajdonságaikban megegyeznek az „egyszerű" CLEANEST-tel. Az elméleti összehasonlítást foglalja össze a 4.4. táblázat.

Eljárás	**Hardverigény**	**Frekvenciarács**	**Numerikus problémák**	**Valós időben megvalósítható**
DFT-alapú, előfeldolgozás blokkosítás után	minimális	fix, statikus	nincs	igen
DFT-alapú, előfeldolgozás blokkosítás előtt	minimális	fix, statikus	nincs	csak nulladrendű tartóval
RBO	alacsony	tetszőleges, statikus	tervezéskor kezelhetők	igen
AFA	alacsony	dinamikus	tervezéskor kezelhetők	igen
Lomb-Scargle	közepes	tetszőleges, statikus	gyakorlatilag nincs	kevesebb frekvencián
DCDFT	közepes	tetszőleges, statikus	gyakorlatilag nincs	kevesebb frekvencián
CLEANEST	magas	dinamikus + tetszőleges, statikus	lehetséges	kétséges
SLICK	magas	dinamikus + tetszőleges, statikus	lehetséges	kétséges
Harmonikus DCDFT	magas	tetszőleges, statikus	lehetséges	kétséges
Harmonikus CLEANEST	magas	dinamikus + tetszőleges, statikus	lehetséges	kétséges
Harmonikus SLICK	magas	dinamikus + tetszőleges, statikus	lehetséges	kétséges

4.4. táblázat. Az eljárások elméleti összehasonlítása

5. fejezet

Szimulációs vizsgálat

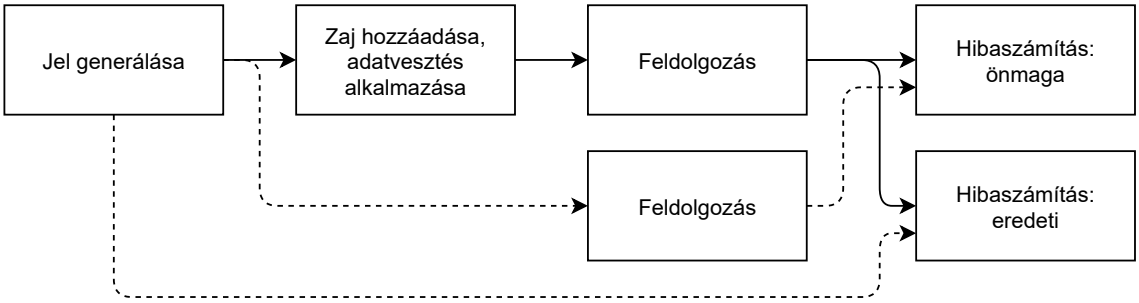
A dolgozatban ismertetett eljárásokat megvalósítottam MATLAB környezetben, és szimulációk segítségével összehasonlítottam. Az alábbiakban először áttekintjük a szimulációs környezetet, a vizsgált jeleket, adatvesztéseket és az eljárásokat, ezek egységes értékelését, majd bemutatjuk a szimulációk eredményeit. Végül javaslatot teszek adott mérési helyzetekben a feldolgozó eljárásra, melyet táblázatos és döntési fa formában is megadok.

5.1. A szimulációs környezet

A szimulációk a következő hipotetikus mérési szituációból származó adatok feldolgozását végezték el: adott N potenciális mintavételi pont, melyek közül a sikeres mintavételeket az alkalmazott adatvesztési modell határozza meg. Ez annak a gyakorlati esetnek felel meg, amikor adott egy jelet adott ideig (például eddig tekinthető állandónak), adott mintavételi frekvenciával (ilyen mérőeszközünk van) tudunk mérni.

5.1.1. Áttekintés

A szimuláció során többféle jelet, adatvesztést és feldolgozási eljárást kipróbáltam. Egy adott jel, adatvesztés és eljárás hármasra a szimulációs folyamatot az 5.1. ábra mutatja. Itt folytonos vonal jelöli a fő feldolgozást, a szaggatott az eredményül kapott spektrum „jóságának” meghatározásához szükséges adatáramlást.



5.1. ábra. Egy adott jel, adatvesztés és eljárás vizsgálatának folyamata

A folyamat lépései az alábbiak:

1. Előállítjuk a jelet: meghatározzuk egyenletes mintavétel szerint a mintavételi időpontokat, valamint ezekben az időpontokban a jel értékét. Itt a jel még zajmentes.

2. A jelhez adott jel-zaj viszony mellett fehér zajt adunk hozzá, előállítjuk az indikátorfüggvényt és alkalmazzuk a jelre az adatvesztést.

3. A zajjal és adatvesztéssel terhelt jelet a kiválasztott eljárással feldolgozzuk.

4. Az eredeti (adatvesztés- és zajmentes) jelet is feldolgozzuk a kiválasztott eljárással.

5. Az így kapott két spektrum eltéréséből kiszámítjuk az önmagához viszonyított hibát.

6. Az adatvesztéssel terhelt jel spektrumának és az eredeti jel – ismert – spektrumának összehasonlításával megkapjuk az eredetihez viszonyított hibát.

A továbbiakban először szemügyre vesszük a vizsgált jelek, adatvesztések, eljárások halmazát, majd a kétféle hibaszámítási módot.

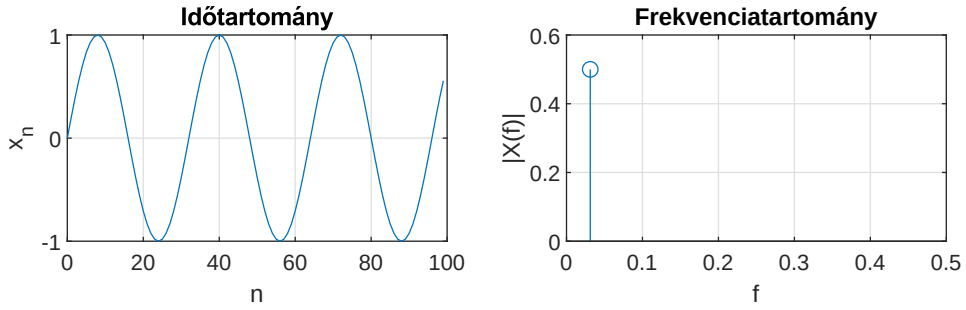
5.1.2. Vizsgált jelek, adatvesztések, eljárások

A szimulációkat az alábbiakban leírt jelek, adatvesztések és eljárások összes lehetséges kombinációjára elvégeztem.

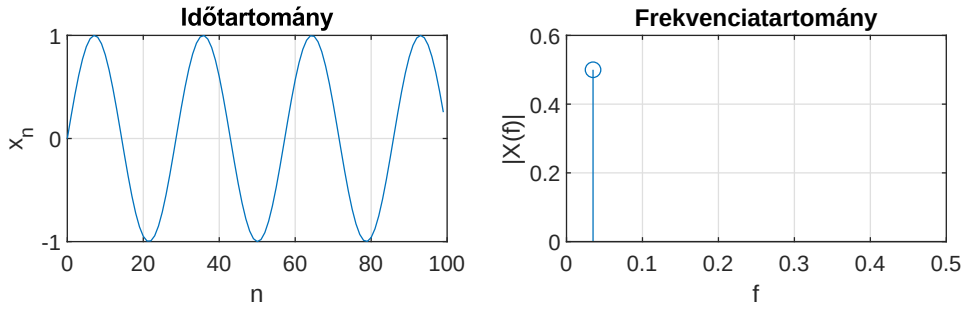
5.1.2.1. Jelek

A szimulációhoz különböző periodikus jeleket használtam. A jelekhez kétféle alapfrekvenciát használtam: a $\frac{1}{32}$ és a $\frac{\sqrt{5}}{64}$ relatív frekvenciákat. A két alapfrekvencia hányadosa irracionális, így a felharmonikusaik sem eshetnek egybe.

Az első két jel egy-egy szinuszos komponenst tartalmaz, a frekvenciáik a két alapfrekvencia (5.2. és 5.3. ábrák).

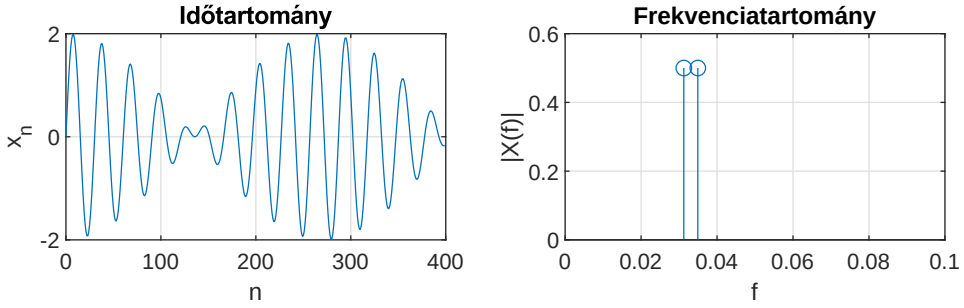


5.2. ábra. 1. vizsgált jel: szinuszjel, $f = \frac{1}{32}$



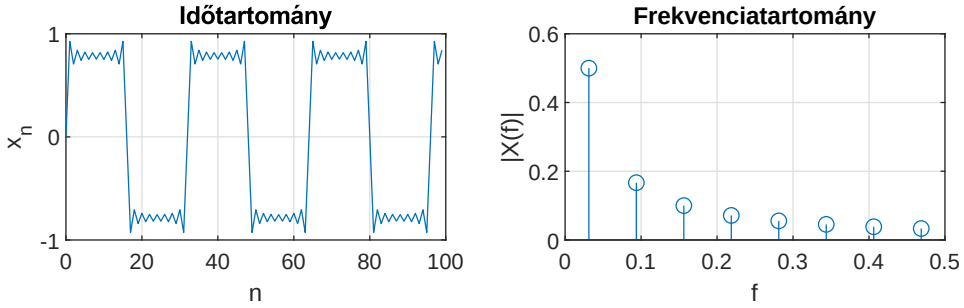
5.3. ábra. 2. vizsgált jel: szinuszjel, $f = \frac{\sqrt{5}}{64}$

A harmadik jel az első két jel összege. Ennek a jelnek a kihívása a két, egymáshoz közel lévő frekvencia, melyek egymással nincsenek harmonikus viszonyban. A jelet és a spektrumát az 5.4. ábrán láthatjuk. Az ábra több mintát mutat, mint az előzők, a jel jobb illusztrálására. Továbbá a frekvenciatartományi ábrán a $[0; 0{,}1]$ relatív frekvenciatartományra ránagyítottunk, hogy a két komponens elkülönüljön. A jelben más komponens nincs.



5.4. ábra. 3. vizsgált jel: kettős szinuszjel, $f_1 = \frac{1}{32}$, $f_2 = \frac{\sqrt{5}}{64}$

A negyedik és ötödik jel közelítő négyszögjel. A négyszögjelet alkotó harmonikus komponensekből azokat vettem figyelembe, amelyek a Nyquist-frekvencia alatt helyezkednek el. Lényegében ugyanazt a logikát alkalmazhatjuk itt is, mint a rezonátoros struktúra származtatásánál: ideális átlapolásgátló szűrőt feltételezve ezek a komponensek maradnak a spektrumban. A két jelet az 5.5. és az 5.6. ábra szemlélteti. Az utóbbi négyszögjelnél megfigyelhető, hogy a mintavétel inkoherens volta miatt a Gibbs-oszcilláció az egyes periódusokban máshogy jelentkezik.

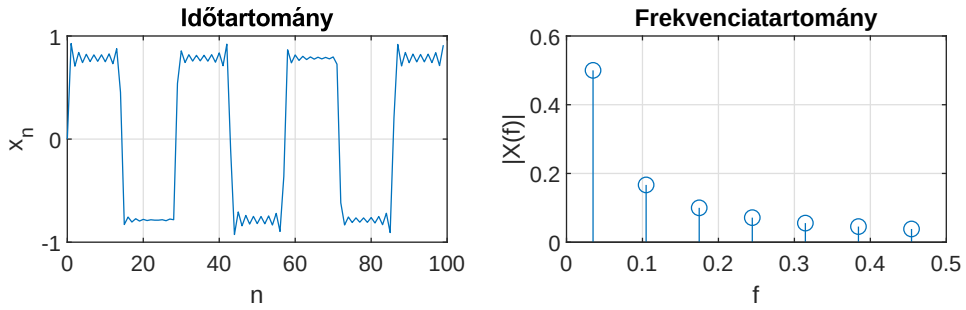


5.5. ábra. 4. vizsgált jel: négyszögjel, $f = \frac{1}{32}$

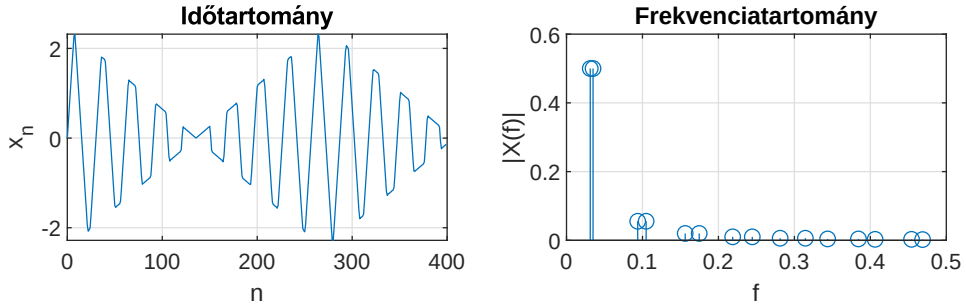
A hatodik jel (5.7. ábra) két közelítő háromszögjel összege (itt is csak a Nyquist-frekvenciáig maradtak meg a komponensek). Az alapfrekvenciák közelsége miatt itt is megjelenik egy lebegésszerű jelenség.

A hetedik jel egy négyszög- és egy háromszögjel összege, a jelet az 5.8. ábrán látjuk. A jelen megfigyelhető zajszerű jelenség a négyszögjelből származó Gibbs-oszcilláció.

A zajmentes jelekhez az egyes szimulációkban 80 vagy 20 dB-es jel-zaj viszony mellett fehér zajt adtam hozzá. A vizsgált adatvesztések és eljárások nagy száma miatt az egyes szimulációs menetekben az eredeti jel $N = 10000$ mintáját állítottam elő.



5.6. ábra. 5. vizsgált jel: négyszögjel, $f = \frac{\sqrt{5}}{64}$



5.7. ábra. 6. vizsgált jel: kettős háromszögjel, $f_1 = \frac{1}{32}$, $f_2 = \frac{\sqrt{5}}{64}$

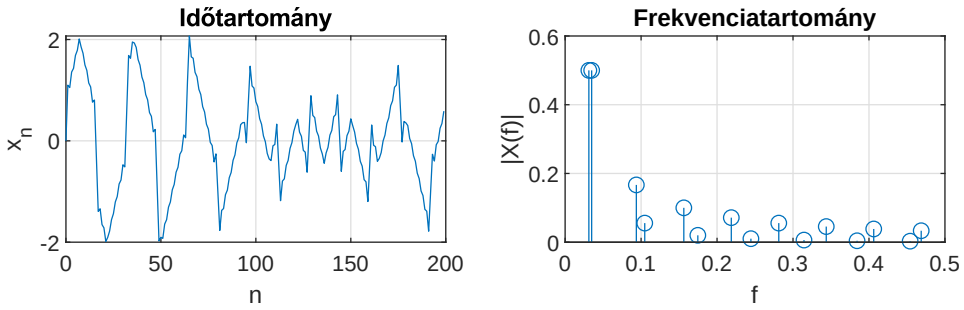
5.1.2.2. Adatvesztések

A dolgozatban bemutatott adatvesztési modellek közül a véletlen független, a kétállapotú Markov, a Gilbert és a Gilbert-Elliott modelleket, illetve ezekre épülő blokkos adatvesztést teszteltem.

Az 5.1. táblázat mutatja a szimulációk során alkalmazott adatvesztési modelleket. Az egyes modellek és alkalmazott paramétereik mellett feltüntetésre kerültek az elméleti adatelérhetőségi arányok is. A táblázatban szereplő összes adatvesztés 1, 10 és 100 mintás blokkok esetén is kipróbálásra került.

A véletlen független adatvesztések esetén az egyetlen paraméter az adatvesztési arány. 1 és 99 százalékos adatvesztés között „szimmetrikus 1-2-5-ös lépésekben” a teljes paramétertéret lefedtem. Ezekkel a gyengétől az extrém mértékű adatvesztésig vizsgálható az eljárások pontossága.

A kétállapotú Markov, a Gilbert és a Gilbert-Elliott modellek esetén úgy történt a kiválasztás, hogy a modellek alapját képező Markov-lánc állapoteloszlásai között legyen olyan, ami az egyik vagy a másik állapotot részesíti előnyben, illetve olyan is, ami nagyjából fele-fele. Így alakultak ki a $(p;q)$ értékekre a három modellben közösen használt $(0{,}1;0{,}9)$, $(0{,}5;0{,}5)$ és $(0{,}9;0{,}1)$ értékek. Továbbá ugyanilyen állapoteloszlások mellett szerettem volna tesztelni az eljárásokat úgy is, hogy az egyes állapotokban töltött idő hosszabb, ezért mindhárom $(p;q)$ pár mellé a tizedére vett valószínűségekkel rendelkező $(p;q)$ párt is felvettem. A Gilbert, illetve a Gilbert-Elliott modellek esetén az egyes állapotokban az adatelérhetőségi arányt úgy választottam meg, hogy a modellek jellegzetes viselkedése érvényesüljön, de a két állapot még lényegesen különböző legyen.



5.8. ábra. 7. vizsgált jel: négyszög- és háromszögjel összege, $f_{\text{négyszög}} = \frac{1}{32}$, $f_{\text{háromszög}} = \frac{\sqrt{5}}{64}$

5.1.2.3. Eljárások

A dolgozatban bemutatott eljárásokat a szimulációs vizsgálat céljából implementáltam. Az eljárásokat az 5.2. táblázat foglalja össze. Az eljárások azonosítója a későbbi hivatkozások egyértelműségét szolgálja.

A DFT-alapú eljárásoknál exponenciális átlagolást használtam α átlagolási tényezővel, továbbá a blokkokat β átlapolással vettem. Itt N a DFT pontszámát jelöli. A zero padding előfeldolgozást használó DFT-k esetén $L_{\min}$ az előfeldolgozás után a blokkban maradó minták minimális száma, hogy még éppen számítsunk belőle spektrumot.

A DFT-alapú eljárásokban az összehasonlíthatóság végett 64, 256 és 1024 pontos DFT-t alkalmaztam. Érdemes észrevenni, hogy így a vizsgált jelek egy része épp a DFT-rácsra esik, míg a másik részük nem. A teljesség kedvéért a DFT-alapú eljárásokat ablakozás nélkül és Hanning-ablak használata mellett is vizsgáltam. A DFT-alapú eljárások esetén közös átlapolási mértéket és exponenciális átlagolási tényezőt használtam szintén az összehasonlíthatóság céljából. Az 5.2. táblázat azonosító oszlopában a zárójel nélküli elem az ablakozás nélküli, a zárójelben lévő a Hanning-ablakot használó eljárásra utal.

A modellalapú eljárásoknál f_0 jelöli a (kezdeti) alapfrekvenciát, míg α az exponenciális átlagolási tényezőt. A rezonátoros struktúrát több alapfrekvenciával is megvalósítottam: egyrészt a 64 és 256 pontos DFT-nek megfelelően, másrészt úgy, hogy az $\frac{\sqrt{5}}{64}$ frekvenciájú jelek is rezonátorpozíciókra essenek. Az AFA esetén – mivel az alapfrekvencia is változik – csak egyetlen kezdeti alapfrekvenciát használtam. Az egyes eljárásokat közös exponenciális átlagolási tényezővel működtettem.

A nem egyenletes mintavétel eljárásaihoz egy sűrűbb, egyenletes frekvenciarácsot használtam, a 8196 pontos DFT frekvenciarácsát (természetesen például a CLEANEST ezek közé a pontok közé is képes optimalizációval frekvenciát állítani), ezt jelöltem az f_0 paraméterrel. Szimmetriaokokból a Nyquist-frekvenciánál nagyobb frekvenciákon a számítást nem végeztem el.

A CLEANEST és a SLICK eljárások esetén a kezdeti (és a maradék) spektrumot e szerint a 8192 pontos frekvenciarács szerint számítottam ki. A kezdeti spektrumból a kétszeres szórás feletti csúcsokat, de legfeljebb 10 legnagyobbat választottam ki az első optimalizációra. A maradék spektrum kiszámítása után az optimalizált frekvenciák közül azokat, amelyekhez tartozó teljesítmény a szórás alatt van, eltávolítottam az optimalizációból, és a kétszeres szórás feletti, de legfeljebb 5 legnagyobb csúcsot vettem be a következő optimalizált frekvenciák közé. Az iterációk maximális száma 15 volt. Alkalmazási leállási feltételnek azt választottam, ha a modellvektor energiája legalább az adatvektor energiájának 0,9999-szerese (ekkor feltételezhetőleg jó a modell), vagy az aktuális iterációban a modellvektor energiát sikerült növelni, de legfeljebb 0,2%-kal (ekkor pedig nem azért

Modell	**Paraméterek**	**Elméleti adatelérhetőség**
Véletlen független	$\gamma = 0{,}01$	$\mu = 0{,}99$
Véletlen független	$\gamma = 0{,}02$	$\mu = 0{,}98$
Véletlen független	$\gamma = 0{,}05$	$\mu = 0{,}95$
Véletlen független	$\gamma = 0{,}1$	$\mu = 0{,}9$
Véletlen független	$\gamma = 0{,}2$	$\mu = 0{,}8$
Véletlen független	$\gamma = 0{,}5$	$\mu = 0{,}5$
Véletlen független	$\gamma = 0{,}8$	$\mu = 0{,}2$
Véletlen független	$\gamma = 0{,}9$	$\mu = 0{,}1$
Véletlen független	$\gamma = 0{,}95$	$\mu = 0{,}05$
Véletlen független	$\gamma = 0{,}98$	$\mu = 0{,}02$
Véletlen független	$\gamma = 0{,}99$	$\mu = 0{,}01$
Kétállapotú Markov	$p = 0{,}1$, $q = 0{,}9$	$\mu = 0{,}9$
Kétállapotú Markov	$p = 0{,}01$, $q = 0{,}09$	$\mu = 0{,}9$
Kétállapotú Markov	$p = 0{,}5$, $q = 0{,}5$	$\mu = 0{,}5$
Kétállapotú Markov	$p = 0{,}05$, $q = 0{,}05$	$\mu = 0{,}5$
Kétállapotú Markov	$p = 0{,}9$, $q = 0{,}1$	$\mu = 0{,}1$
Kétállapotú Markov	$p = 0{,}09$, $q = 0{,}01$	$\mu = 0{,}1$
Gilbert	$p = 0{,}1$, $q = 0{,}9$, $b = 0{,}1$	$\mu = 0{,}91$
Gilbert	$p = 0{,}01$, $q = 0{,}09$, $b = 0{,}1$	$\mu = 0{,}91$
Gilbert	$p = 0{,}5$, $q = 0{,}5$, $b = 0{,}1$	$\mu = 0{,}55$
Gilbert	$p = 0{,}05$, $q = 0{,}05$, $b = 0{,}1$	$\mu = 0{,}55$
Gilbert	$p = 0{,}9$, $q = 0{,}1$, $b = 0{,}1$	$\mu = 0{,}19$
Gilbert	$p = 0{,}09$, $q = 0{,}01$, $b = 0{,}1$	$\mu = 0{,}19$
Gilbert-Elliott	$p = 0{,}1$, $q = 0{,}9$, $a = 0{,}9$, $b = 0{,}1$	$\mu = 0{,}82$
Gilbert-Elliott	$p = 0{,}01$, $q = 0{,}09$, $a = 0{,}9$, $b = 0{,}1$	$\mu = 0{,}82$
Gilbert-Elliott	$p = 0{,}5$, $q = 0{,}5$, $a = 0{,}9$, $b = 0{,}1$	$\mu = 0{,}5$
Gilbert-Elliott	$p = 0{,}05$, $q = 0{,}05$, $a = 0{,}9$, $b = 0{,}1$	$\mu = 0{,}55$
Gilbert-Elliott	$p = 0{,}9$, $q = 0{,}1$, $a = 0{,}9$, $b = 0{,}1$	$\mu = 0{,}18$
Gilbert-Elliott	$p = 0{,}09$, $q = 0{,}01$, $a = 0{,}9$, $b = 0{,}1$	$\mu = 0{,}18$

5.1. táblázat. Az alkalmazott adatvesztési modellek

nem nőtt az energia, mert nem sikerült az optimum felé mozdulni az optimalizáció során, hanem mert az újabb komponensek már „elég kicsik” voltak).

A harmonikus DCDFT esetén először a (3.68) szerinti amplitúdót számítottam ki a frekvenciarácson, majd ezek közül a maximális értékhez tartozó frekvencián a harmonikus komponensek értékét. A legnagyobb komponens $0,1\%$-ánál kisebb amplitúdójú komponenseket elhanyagolhatónak tekintettem.

A harmonikus CLEANEST és SLICK eljárásoknál a frekvenciákat az optimalizációba egyesével vettem be. Az iterációk maximális száma szintén 15 volt, az alkalmazási leállási feltétel pedig, hogy a modellvektor energiája legalább az adatvektor energiájának 98%-a, vagy a modellvektor energiáját sikerült emelni, de kevesebb, mint 1%-kal. Ezek a kritériumok azért megengedőbbek, mint az „egyszerű” CLEANEST és SLICK esetén, mert itt teljes periodikus jeleket veszünk be a modellfüggvénybe, nem csak ezek egy-egy komponensét. A legnagyobb szinuszos komponens $0,1\%$-ánál kisebb amplitúdójú komponenseket elhanyagolhatónak tekintettem.

Azonosító	**Eljárás**	**Paraméterek**
ZST-DFT64 (ZST-DFTwH64)	DFT + helyettesítés nullával	$N = 64$, $\alpha = 0{,}2$, $\beta = 0{,}5$
ZST-DFT256 (ZST-DFTwH256)	DFT + helyettesítés nullával	$N = 256$, $\alpha = 0{,}2$, $\beta = 0{,}5$
ZST-DFT1024 (ZST-DFTwH1024)	DFT + helyettesítés nullával	$N = 1024$, $\alpha = 0{,}2$, $\beta = 0{,}5$
ZOH-DFT64 (ZOH-DFTwH64)	DFT + nulladrendű tartó	$N = 64$, $\alpha = 0{,}2$, $\beta = 0{,}5$
ZOH-DFT256 (ZOH-DFTwH256)	DFT + nulladrendű tartó	$N = 256$, $\alpha = 0{,}2$, $\beta = 0{,}5$
ZOH-DFT1024 (ZOH-DFTwH1024)	DFT + nulladrendű tartó	$N = 1024$, $\alpha = 0{,}2$, $\beta = 0{,}5$
NN-DFT64 (NN-DFTwH64)	DFT + legközelebbi szomszéd	$N = 64$, $\alpha = 0{,}2$, $\beta = 0{,}5$
NN-DFT256 (NN-DFTwH256)	DFT + legközelebbi szomszéd	$N = 256$, $\alpha = 0{,}2$, $\beta = 0{,}5$
NN-DFT1024 (NN-DFTwH1024)	DFT + legközelebbi szomszéd	$N = 1024$, $\alpha = 0{,}2$, $\beta = 0{,}5$
LERP-DFT64 (LERP-DFTwH64)	DFT + lineáris interpoláció	$N = 64$, $\alpha = 0{,}2$, $\beta = 0{,}5$
LERP-DFT256 (LERP-DFTwH256)	DFT + lineáris interpoláció	$N = 256$, $\alpha = 0{,}2$, $\beta = 0{,}5$
LERP-DFT1024 (LERP-DFTwH1024)	DFT + lineáris interpoláció	$N = 1024$, $\alpha = 0{,}2$, $\beta = 0{,}5$
ZP-DFT64 (ZP-DFTwH64)	DFT + zero padding	$N = 64$, $\alpha = 0{,}2$, $\beta = 0{,}5$, $L_{\min} = 16$
ZP-DFT256 (ZP-DFTwH256)	DFT + zero padding	$N = 256$, $\alpha = 0{,}2$, $\beta = 0{,}5$, $L_{\min} = 64$
ZP-DFT1024 (ZP-DFTwH1024)	DFT + zero padding	$N = 1024$, $\alpha = 0{,}2$, $\beta = 0{,}5$, $L_{\min} = 256$
RBO-1/64	RBO	$f_0 = 1/64$, $\alpha = 0{,}1$
RBO-1/256	RBO	$f_0 = 1/256$, $\alpha = 0{,}1$
RBO-sqrt(5)/128	RBO	$f_0 = \sqrt{5}/128$, $\alpha = 0{,}1$
RBO-sqrt(5)/512	RBO	$f_0 = \sqrt{5}/512$, $\alpha = 0{,}1$
AFA-1/64	AFA	$f_0 = 1/64$, $\alpha = 0{,}1$
LS8192	Lomb-Scargle	$f_0 = 1/8192$
DCDFT8192	DCDFT	$f_0 = 1/8192$
CLEANEST8192	CLEANEST	$f_0 = 1/8192$, $K = 15$
SLICK8192	SLICK	$f_0 = 1/8192$, $K = 15$
H-DCDFT8192	harmonikus DCDFT	$f_0 = 1/8192$
H-CLEANEST8192	harmonikus CLEANEST	$f_0 = 1/8192$, $K = 15$
H-SLICK8192	harmonikus SLICK	$f_0 = 1/8192$, $K = 15$

5.2. táblázat. A szimulációk során vizsgált eljárások

5.1.3. Hibaszámítás

Az eljárások pontosságának értékeléséhez szükségünk van valamilyen hibakritérium felállítására. Alapvetően két kérdést tehetünk fel: az adatvesztés milyen módon befolyásolja az eljárások által adott becslőt (függetlenül attól, hogy adatvesztés nélkül milyen pontos), illetve, hogy az eljárások milyen pontosan képesek megbecsülni a vizsgálójeleink komponenseit. Előbbi hibát önmagához viszonyított, utóbbi eredetihez viszonyított hibának fogjuk nevezni.

5.1.3.1. Önmagához viszonyított hiba

Az önmagához viszonyított hiba kiszámítása viszonylag egyszerű, hiszen bármilyen formában is adjon számunkra eredményt a vizsgált eljárás, ez a forma nem változik az adatvesztés hatására.

A DFT-alapú eljárásoknál a kimenet fix frekvenciákon áll elő. Ezért az adatvesztéssel terhelt és az eredeti jel spektrumának különbségét könnyen képezhetjük, majd ennek négyzetösszege adja a hiba energiáját. Ezt az energiát az eredeti kimenet négyzetösszegéhez viszonyítva megkapjuk a használt – relatív – hibát:

$$h_{\text{ö}} = \frac{\sum_k \left(P_{v,k} - P_{e,k}\right)^2}{\sum_k P_{e,k}^2}, \tag{5.1}$$

ahol $P_{v,k}$ és $P_{e,k}$ az adatvesztéssel terhelt és az eredeti jelek spektrumának k. frekvenciához tartozó komponense. Mivel itt a spektrumot úgy értelmeztük, mint adott frekvenciákon lévő energia/teljesítmény, nem szükséges abszolút érték használata a különbségben. A hiba értéke 0, ha a két spektrum megegyezik, és 1, ha az adatvesztéssel terhelt jel spektruma konstans nulla.

A rezonátoros struktúra esetén a hibát szintén számíthatjuk (5.1) szerint, csak itt az egyes komponenseket az egyes állapotváltozóknak feleltetjük meg. Az AFA esetén is használható ez a képlet, kis meggondolás után. Amennyiben a frekvencia rossz értékre áll be, akkor a becsült amplitúdók sem lesznek helyesek. Vagyis csak a helyes frekvencia esetén lehetnek pontosak az amplitúdóbecslők. Ezért elegendő az amplitúdóbecslőket összehasonlítani; ha a becsült frekvencia eltérése miatt más az eredeti és az adatvesztés utáni jel becsült komponenseinek száma, akkor addig kell venni a különbséget, amíg mindkét esetben van komponens.

A Lomb-Scargle és a DCDFT esetén is használhatjuk ugyanazt a hibaszámítási módszert, mint a DFT esetén. A CLEANEST és a SLICK eljárásoknál azonban nem – mivel a becslést a megtalált diszkrét komponensek hordozzák, melyek frekvenciái adatvesztés mellett nem feltétlenül egyeznek meg az eredeti jelből becsült frekvenciákkal. Viszont mindkét eljárás előállít egy modellvektort, mely a becsült komponenseket tartalmazza időtartományban, így – Parseval-tétel figyelembe vételével – az időtartományban is számolhatunk energiát: az (5.1) egyenlet szerint, a modellvektorok behelyettesítésével. Ugyanezekkel a módszerekkel számíthatjuk a hibát a harmonikus DCDFT, CLEANEST és SLICK eljárások esetén.

5.1.3.2. Eredetihez viszonyított hiba

Az eredetihez viszonyított hiba esetén is vonzó lehetőség az eredeti jel és a modellvektor összehasonlítása. Sajnos adatvesztés esetén előfordulhat olyan eset, hogy egy adott komponenshalmaz az elérhető mintákon jól modellezi a jelet, viszont ez a halmaz különbözik a jel valódi komponenseitől [19]. Ezért az eredetihez viszonyított hiba kiszámítását frekvenciatartományban fogjuk elvégezni.

Ez a hibaszámítás két lépésből áll. Mivel a vizsgált eljárásaink különféle „formátumú” kimenetekkel rendelkeznek, először az adott eljárás kimenetéből meg kell határoznunk a becsült komponenseket. Másodszor pedig a becsült komponenseket a jel komponenseihez kell viszonyítanunk, itt figyelembe kell vennünk mind a frekvencia-, mind az amplitúdóhibát is.

A becsült komponensek meghatározása

Az eredeti jel komponenseit egy frekvencia-amplitúdó párokból álló listaként ismerjük. Mivel a fázisok most érdektelenek, ezt a listát átalakítjuk frekvencia-teljesítmény párokra, és ilyen listát készítünk az egyes eljárások eredményeiből is.

A DFT-alapú eljárások esetén a spektrumban levő csúcsokhoz tartozó frekvencia- és teljesítményértékek adják a becsült listát. Először a spektrum Nyquist- frekvenciánál nagyobb részét – mivel valós jeleket vizsgálunk – elhagyjuk, és a $(0; 0{,}5)$ relatív frekvenciatartományba eső teljesítményt megduplázzuk. Ebben a csúcskeresést lokális maximum kereséseként valósítjuk meg. Érdemes valamilyen minimális kiemelkedési szintet szabni, hogy a zajban lévő hullámzásokat ne érzékeljük komponensnek. A szimuláció során ez a minimális kiemelkedési szint a legnagyobb csúcs nagyságának 1%-a volt.

Az RBO és az AFA esetén az állapotváltozók alapján készíthetjük el a becsült listát. A DC komponens teljesítménye a 0. állapotváltozó abszolútérték-négyzete, míg az i. harmonikus komponens esetén a $\pm i$-s állapotváltozók abszolútértéknégyzet-összege. A konjugált szimmetria miatt elegendő a pozitív frekvenciához tartozó állapotváltozóból számított teljesítmény kétszeresét venni. A 3.10. ábrán már láttuk ezt a számítást is.

A Lomb–Scargle-eljárás és a DCDFT esetén a DFT-hez hasonlóan adott frekvenciarácson kapjuk a teljesítménybecslőt, így egy hasonló csúcskereséssel itt is célba érhetünk. Mivel ezek az eljárások szinuszfüggvényt illesztenek, itt nincs szükség a teljesítmény duplázására.

A CLEANEST és a SLICK eljárások egyrészt eredményül adják a diszkrét komponenseket, másrészt a maradék spektrumot. Elsőre gondolhatnánk azt, hogy ekkor a becsült listát alkossák a diszkrét komponensek, viszont előfordulhat, hogy a diszkrét komponensek között maradtak elhanyagolható amplitúdójúak, illetve hogy a maradék spektrum tartalmaz még nem elhanyagolható csúcsokat. Ezért itt a listában a diszkrét komponensekből és a maradék spektrumból azok az elemek kerülnek be, amelyek nagysága legalább a legnagyobb diszkrét komponens nagyságának 1%-a.

A harmonikus DCDFT esetén közvetlenül ezt a becsült listát kapjuk eredményül. A harmonikus CLEANEST és SLICK eljárások hasonlóan kezelhetők, mint az „egyszerű” társaik.

A komponensek összehasonlítása

Az előbbiekben meghatároztuk az eredeti és a becsült frekvencia-teljesítmény listákat. Az alábbiakban ismertetem azt a számítási eljárást, amellyel a hibát – a két lista különbözőségét – meghatároztam a szimulációk során. Legyen az eredeti listában M darab (f_k, P_k) pár ($k \in \{1, \ldots, M\}$), míg a becsültben $\hat{M}$ darab $\left(\hat{f}_k, \hat{P}_k\right)$ pár ($k \in \left\{1, \ldots, \hat{M}\right\}$). Jelölje $\mathbf{P} = \begin{bmatrix} P_1 & \ldots & P_M \end{bmatrix}^T$ az eredeti, míg $\hat{\mathbf{P}} = \begin{bmatrix} \hat{P}_1 & \ldots & \hat{P}_{\hat{M}} \end{bmatrix}^T$ a becsült teljesítményekből alkotott vektorokat.

Itt is egy olyan hibamérték kialakítása a cél, amely értéke 0, ha a két lista megegyezik és 1, ha például a becsült lista üres. Egy ilyen hibakritérium az alábbi módon kiszámított:

1. Skálatényező meghatározása, hogy az eredeti lista maximális teljesítménye 0,5 legyen, majd ennek alkalmazása az összes teljesítményre.

$$\alpha = \frac{0{,}5}{\max_k P_k}, \tag{5.2}$$

$$\mathbf{P} := \alpha\mathbf{P}, \tag{5.3}$$

$$\hat{\mathbf{P}} := \alpha\hat{\mathbf{P}}. \tag{5.4}$$

   Ezt azért tesszük, hogy később a frekvencia- és a teljesítményhibát azonos súllyal vegyük figyelembe.

2. A hibát a következő változóba gyűjtjük:

$$h_\mathrm{e} := 0. \tag{5.5}$$

3. Megpróbáljuk minél kisebb hibával párosítani az eredeti és a becsült komponenseket. Csökkenő teljesítmény szerint egyesével sorra vesszük a becsült lista elemeit, és az alábbi lépéseket elvégezzük. Az aktuális elem frekvenciája legyen $\hat{f}_k$, teljesítménye pedig $\hat{P}_k$.

   (a) A hiba, ha nem találunk a kiválasztott elemnek párt:

$$h_\mathrm{új} := \hat{P}_k^2, \tag{5.6}$$

$$I_\mathrm{új} := 0, \tag{5.7}$$

   ahol $I_\mathrm{új}$ az eredeti komponensekből a hozzá talált legjobb pár indexét fogja tartalmazni, ha értéke nullától különböző lesz (ha nulla marad, akkor nem találtunk hozzá párt).

   (b) A még párosítatlan eredeti komponenseket egyesével tetszőleges sorrendben véve, rájuk az alábbi lépések végrehajtása. Az éppen kiválasztott elem frekvenciája és teljesítménye legyen rendre f_i és P_i.

      i. A hiba, ha a kiválasztott eredeti komponenssel állítjuk párba:

$$h_\mathrm{komponens} = \left(\hat{P}_k - P_i\right)^2 + \left(\hat{f}_k - f_i\right)^2. \tag{5.8}$$

      Vagyis a frekvenciát és a teljesítményt a sík két koordinátájának feleltetjük meg, és távolságnégyzetet számítunk. Így a páratlanul maradt becsült komponens hibája úgy értelmezhető, mintha felvettünk volna egy fiktív eredeti komponenst, nulla teljesítménnyel és egyező frekvenciával.

      ii. Ha $h_\mathrm{komponens} < h_\mathrm{új}$, akkor a kiválasztott eredeti komponens az eddigi legjobb párja az épp vizsgált becsültnek. Ekkor elmentjük a találat indexét ($I_\mathrm{új} := i$) és az eddigi legjobb hibát ($h_\mathrm{új} := h_\mathrm{komponens}$).

   (c) Ha találtunk párt ($I_\mathrm{új} \neq 0$), akkor az $I_\mathrm{új}$ indexű eredeti komponenst megjelöljük, hogy már párosítottuk.

   (d) A hibához hozzáadjuk az aktuális becsült komponens legjobb párosítás szerinti hibáját:

$$h_\mathrm{e} := h_\mathrm{e} + h_\mathrm{új}. \tag{5.9}$$

4. Miután az összes becsült komponenst párosítottuk, maradhattak még páratlan eredeti komponensek. Ezeket ugyanúgy vesszük hozzá a hibához, mint a párosítatlan

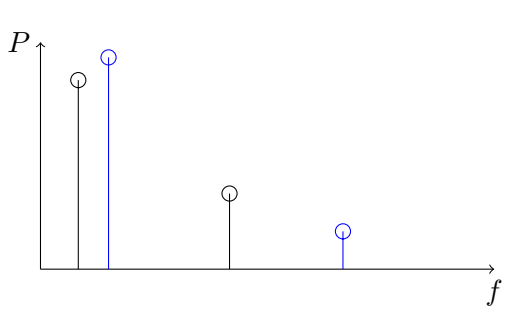
becsült komponenseket, vagyis mindegyik (f_j, P_j) párosítatlan eredeti komponensre:

$$h_\mathrm{e} := h_\mathrm{e} + P_j^2. \tag{5.10}$$

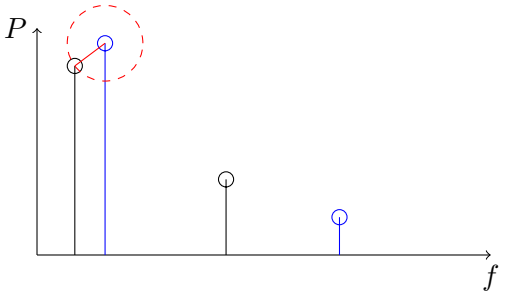
5. A hibát normáljuk az eredeti komponensek összteljesítményével:

$$h_\mathrm{e} := \frac{h_\mathrm{e}}{\mathbf{P}^T\mathbf{P}}. \tag{5.11}$$

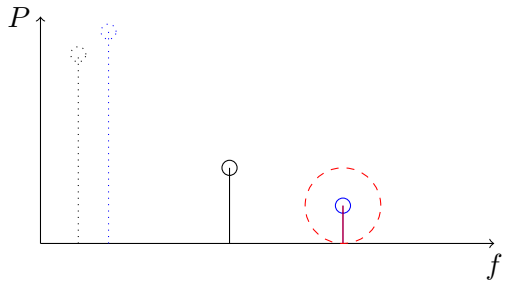
A hibaszámítás működését az 5.9. ábra szemlélteti. Az ábrán az eredeti komponenseket fekete, a becsülteket kék színnel jelöltem. Az 5.9.a. ábra a skálázás utáni állapotot mutatja. Kiválasztva a legnagyobb becsült komponenst, ezt egy közeli eredetihez tudjuk párosítani (5.9.b. ábra). Itt a piros vonal hosszának négyzete a hozzáadott hiba, és a piros szaggatott kör az ezt a hibajárulékot adó pontok halmazát mutatja. Mivel a piros körön belül nincs másik eredeti komponens, illetve ez nem metszi a frekvenciatengelyt, ehhez a komponenshez ez a legjobb párosítás. Miután ezeket párosítottuk, folytatjuk a következő legnagyobb (itt az egyetlen másik) becsült komponenssel (5.9.c. ábra). A már párosított komponenseket pontozott vonal jelöli, mivel ezek a továbbiakban nem vesznek részt a hibaszámításban. Itt az a legkedvezőbb, ha a vizsgált komponenst nem párosítjuk eredetihez, hanem önmagában hagyjuk. Így a párosítatlanul maradt eredeti komponenst önmagában kell a hibába beszámítani (5.9.d. ábra).



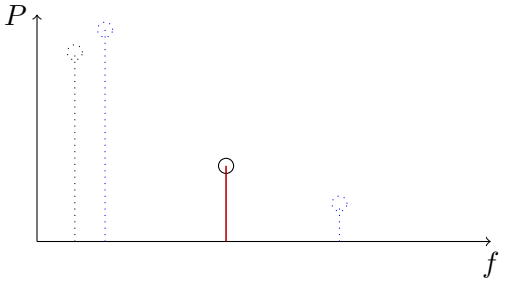
(a) Skálázás utáni állapot



(b) A nagyobb becsült komponens párosítása



(c) A kisebb becsült komponens párosítása



(d) A nem párosított eredeti komponensek számításba vétele

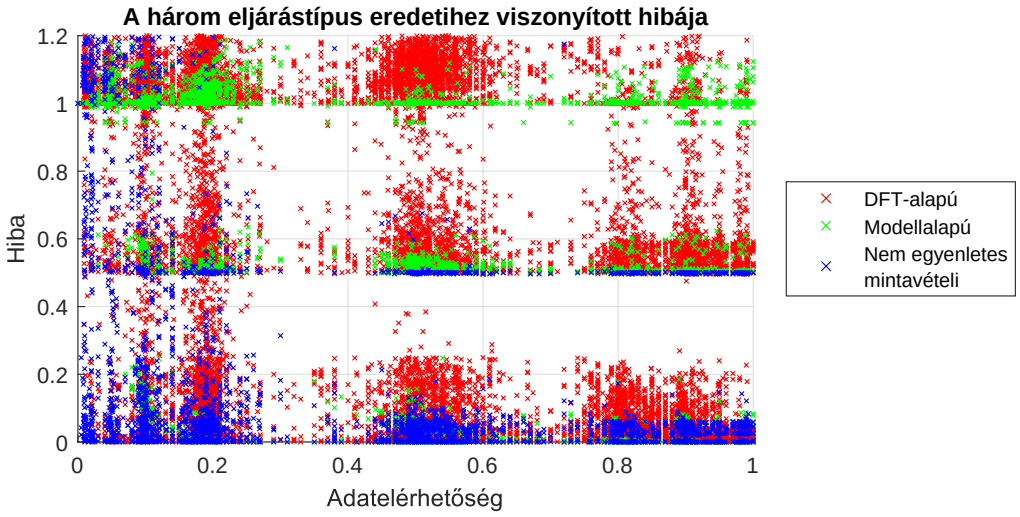
5.9. ábra. Példa az eredetihez viszonyított hibaszámítási eljárásra

5.2. Szimulációs eredmények

Az eredményeket a következők szerint vesszük sorra: először az eredetihez viszonyított hiba vizsgálatával választ kapunk arra, hogy a komponensek becslésének pontosságát mennyire befolyásolja az adatvesztés. Majd az önmagához viszonyított hiba alapján az eljárások adatvesztésre való érzékenységét vesszük szemügyre. Végül a tanulságokat összegezzük és eljáráskiválasztási javaslatokat teszünk.

5.2.1. Eredetihez viszonyított hiba

Az adatelérhetőség függvényében ábrázolja az eredetihez viszonyított hibát az egyes eljárástípusok esetén az 5.10. ábra. Minden kis „x” egy mérési eredményt jelöl: adott jel adott adatvesztés melletti feldolgozását adott eljárással. Az ábra nagyított, mivel extrém mértékű hibák is adódtak kis adatelérhetőség esetén. A teljes ábrát és két, fokozatosan nagyított változatát – a teljesség kedvéért – a függelék tartalmazza (F.1.1, F.1.2, F.1.3. ábrák).



5.10. ábra. Az eredetihez viszonyított hiba az eljárástípusok szerinti bontásban

Az ábrát három vízszintes sáv dominálja: a sávok a 0, 0,5 és 1 hibaértékekre „épülnek rá”. Észrevehetünk függőleges sávokat is: $0\cdots\approx 0{,}3$, $\approx 0{,}3\cdots\approx 0{,}7$ és $\approx 0{,}7\ldots 1$, ezeket rendre erős, közepes és gyenge adatvesztésnek fogom nevezni. A gyenge adatvesztés 1-hez közeli adatelérhetőségi tartományára nagyon gyenge adatvesztésként fogok hivatkozni. A függőleges sávokat a választott adatvesztési modellek okozták.

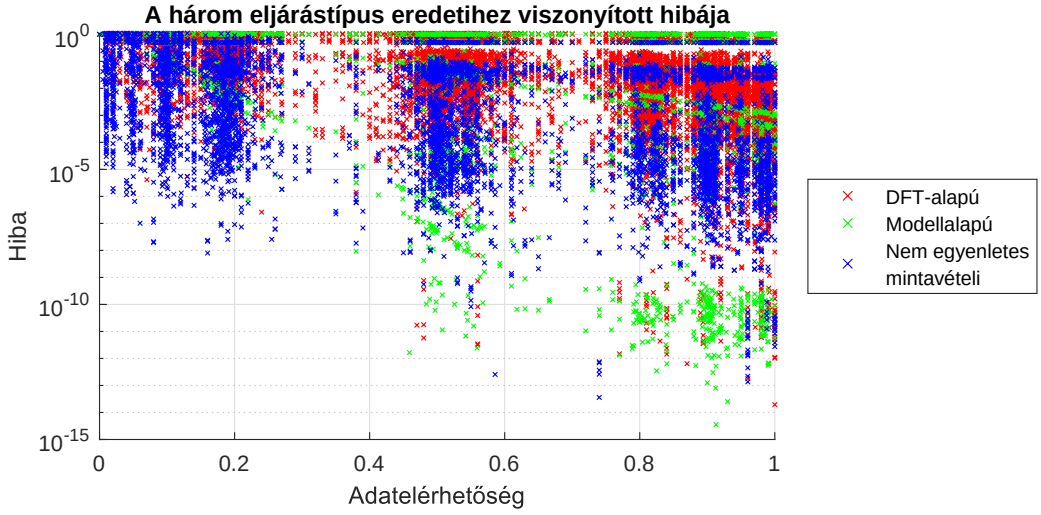
Erős adatvesztés esetén a három vízszintes sáv összemosódik. A felső és a középső sávokban azokat az eseteket láthatjuk, amikor az eljárások nem vagy csak részlegesen adtak helyes becslőt.

A felső sáv egyrészt a DFT-alapú eljárásokból, másrészt a hibásan – inkoherensen – beállított rezonátoros struktúrából, harmadrészt a kváziperiodikus jelek AFA-val való feldolgozásából adódott (F.1.4. ábra). A DFT-alapú eljárások általánosan csak gyenge adatelérhetőség esetén működőképesek: különben nullával helyettesítés és interpoláció használatakor a spektrum erősen torzul, a zero padding esetén pedig nem marad elég minta, így a blokk használhatatlan lesz. Az inkoherens RBO azt jelenti, hogy az alapfrekvencia beállítása hibás, így a struktúra képtelen a jelben lévő komponensek becslésére. A kváziperiodikus jelek esetén nincs alapfrekvencia, amit az AFA megtalálhatna.

A középső sáv léte abból adódik, hogy egyes vizsgálójeleink kváziperiodikusak voltak, ez jól látható az F.1.5. ábrán. A sáv összetevőkre bontását mutatja az F.1.6. ábra. Mivel a kváziperiodikus vizsgálójelek két alapfrekvenciája túl közel van ahhoz, hogy a 64 és a 256 pontos DFT-vel elkülönítsük őket, a velük számított becslők hibásak lesznek. A nagy hiba abból adódik, hogy a két alapfrekvenciához tartozó teljesítmény közelítőleg összeadódik, majd ez az egyetlen becsült komponens hasonlítódik a két – jelen esetben – azonos teljesítményű alapfrekvenciához tartozó eredeti komponenshez. Így egy nagy amplitúdóhiba keletkezik a párosításból, illetve a másik alapfrekvenciához tartozó eredeti csúcs valószínűleg páratlan marad, szintén nagy hibajárulékot okozva.

Az RBO esetén az alapfrekvencia a kváziperiodikus jelek egyik komponensére volt beállítva. Ezért a vizsgálójel egyik periodikus tagjának komponenseit helyesen tudtuk becsülni, a másik jel pedig zajként hatott a becslésre. Így a jel komponenseinek körülbelül felét sikerül megtalálni és megbecsülni. Ez teljesítményben nem feltétlenül a felét jelenti, például a négyszög- és háromszögjel összege esetén ha a háromszögjel frekvenciájára van állítva az RBO, akkor csak a teljesítmény kisebb részét fogjuk megkapni. Ez az oka az F.1.6. ábra jobb felső sarkában lévő értéksorozatnak. Továbbá, a H-DCDFT8192 eljárással csupán egy periodikus komponenst kerestem, ezért a kváziperiodikus vizsgálójeleink esetén szintén csak a komponensek felét vesszük figyelembe (a nagyobb teljesítményű tag komponenseit).

Az alsó sáv hordozza számunkra a legtöbb információt: amennyiben működnek az eljárások, mennyire pontosak? A viszonyok jobb áttekintése érdekében érdemes az 5.10. ábrát újrarajzolni úgy, hogy a hibát logaritmikus skálán ábrázoljuk, ezt mutatja az 5.11. ábra.



5.11. ábra. Az eredetihez viszonyított hiba az eljárástípusok szerinti bontásban, logaritmikus hibával

Az ábrán a modellalapú eljárások eredményei részben egyenesek mentén látszódnak rendeződni. Ezeket az RBO eredményei alkotják: az adatelérhetőség adott kiinduló mintaszám mellett egy effektív mintaszámként szolgál. Így, mivel a beállás exponenciális, a hiba logaritmikus skálán a feldolgozott minták számának függvényében lineárisan csökken.

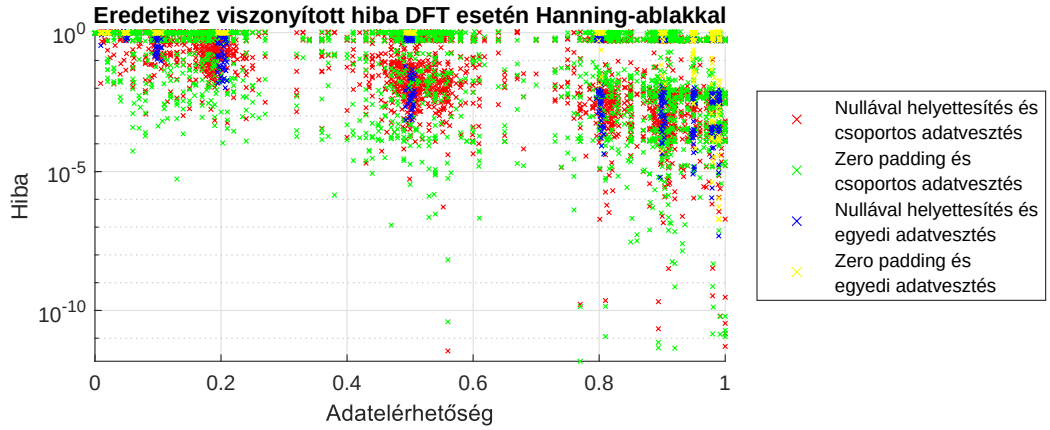
Az ábráról látható, hogy amennyiben a lehető legpontosabb becslőt szeretnénk elérni, akkor gyenge vagy közepes adatvesztés és periodikus mérendő jel esetén érdemes modellalapú eljárást választani, míg erős adatvesztésnél a nem egyenletes mintavételi eljárások kínálják a legjobb lehetőséget. Ezek a trendek általánosan jelen vannak adatvesztési modellektől, blokkmérettől és zajszinttől függetlenül.

5.2.1.1. A DFT-alapú eljárások összehasonlítása

A DFT esetén a pontosságot adatvesztés esetén is elsősorban az befolyásolja, hogy alkalmazunk-e ablakfüggvényt. Ezt mutatja az F.1.7. ábra: inkoherens mintavétel esetén ablakozással egyértelműen pontosabb eredményt lehet elérni, mint ablakozás nélkül. Mivel általános esetben a koherens mintavétel nem biztosítható, a DFT-alapú eljárásokat célszerű ablakozással használni (a zero padding előfeldolgozás használatakor pedig még koherens mintavétel esetén is ablakozni kell a megmaradó részt). Ezért a továbbiakban csak az ablakozást alkalmazó DFT-s eredményeket elemezzük.

Az előfeldolgozási eljárások típusa szerint ábrázolva (F.1.8. ábra) azt állapíthatjuk meg, hogy csupán igen gyenge adatvesztés esetén ad várhatóan jobb eredményt a blokkosítás előtti előfeldolgozás, mint a blokkosítás utáni. Ilyen gyenge adatvesztés esetén a különböző interpolációs eljárások pontossága között számottevő különbség nincs.

A blokkosítás utáni előfeldolgozás eredményei (5.12. ábra) különböznek attól függően, hogy az adatvesztés egyedi vagy csoportos. Egyedi adatvesztés esetén egyértelműen a nullával helyettesítés, csoportosnál pedig a zero padding módszer ad pontosabb eredményt. A zero padding eljárás használatakor mind egyedi, mind csoportos adatvesztés esetén fennáll az a lehetőség, hogy az elveszett minták eloszlása olyan legyen, hogy miattunk ne maradjon elég minta a spektrumszámításra a blokkban az előfeldolgozás után. Ez különösen érvényesül a hosszabb DFT-k esetén: a felső sávot nagyrészt az 1024 pontú DFT ilyen esetei alkotják.



5.12. ábra. Az eredetihez viszonyított hiba a blokkosítás utáni előfeldolgozás esetén, egyedi és csoportos adatvesztés mellett

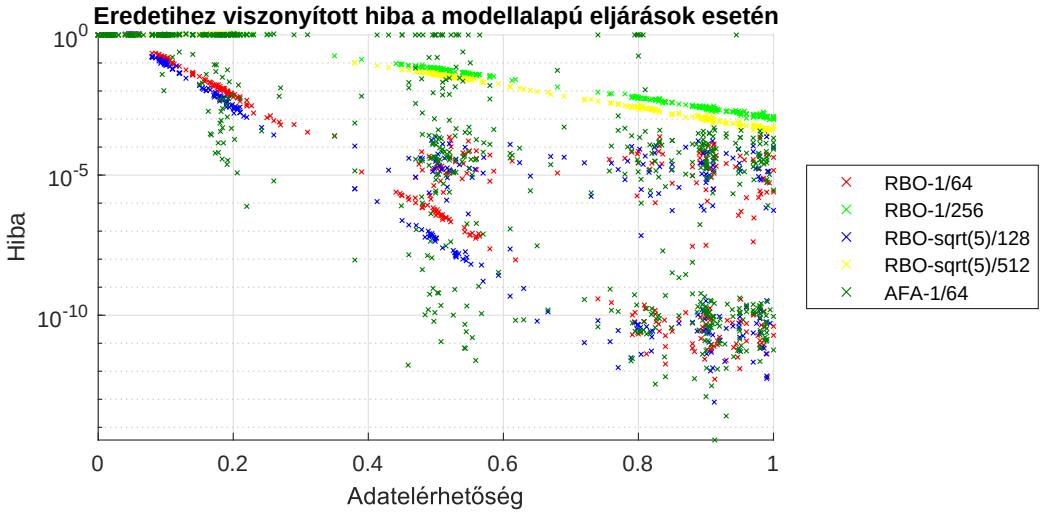
A DFT-s eljárások esetén az egy intuitív várakozás, hogy az egyedi adatvesztés mellett (amikor nem jellemzők az üres sorozatok) pontosabb az eredmény, mint a csoportos adatvesztésnél (amikor az üres sorozatok jellemzők). Ugyanis minél távolabb vagyunk egy elérhető mintától, feltehetőleg annál pontatlanabbak az interpolációs módszerek. Ez a várakozás megfigyelhető az F.1.9. ábrán is: az egyedi adatvesztés eredményei kisebb hibánál sűrűsödnek, mint a csoportosé. Közepes és erős adatvesztés mellett gyakran előfordult azonban, hogy a csoportos adatvesztés pontosabb eredményt produkáljon, mint az egyedi, ez jellemzően a zero padding előfeldolgozás mellett jelentkezett. A többi ilyen esetet az adta, hogy 100 mintás blokkokból 64 pontú DFT-t számítottunk: ekkor egy elérhető blokkba teljes egészében belefért egy DFT-blokk, így azt előfeldolgozás nélkül ki tudtuk értékelni.

A DFT pontszáma szerint vizsgálódva (F.1.10. ábra) azt vehetjük észre, hogy erős adatvesztésnél a 64, míg máskor a 256 pontú DFT hozta a legjobb eredményt. Ennek magyarázataként egyrészt megfigyelhetjük, hogy az 1024 pontú DFT pontossága „alulról korlátos”, ugyanis csak kevesebb blokk áll rendelkezésre az átlagoláshoz. Másrészt, erős adatvesztés esetén a a 64 pontú DFT legjobb eredményei a zero padding eljárás használatával születtek (v.ö. F.1.9. ábra): a legtöbb esetben a blokkban nem maradt elég minta a spektrumszámításhoz, de amikor igen, akkor viszonylag pontos becslőt sikerült adni.

5.2.1.2. A modellalapú eljárások értékelése

A modellalapú eljárások részletes vizsgálatánál a helytelen működést okozó konfigurációkat figyelmen kívül hagyjuk. Ezek a kváziperiodikus vizsgálójel alkalmazás (jelmodell nem teljesül), valamint az inkoherens RBO (alapfrekvencia hibás beállítása). Ha a hozzájuk tartozó eredményeket ábrázoljuk (F.1.11. ábra), akkor láthatjuk, hogy ez a megkülönböztetés jogos.

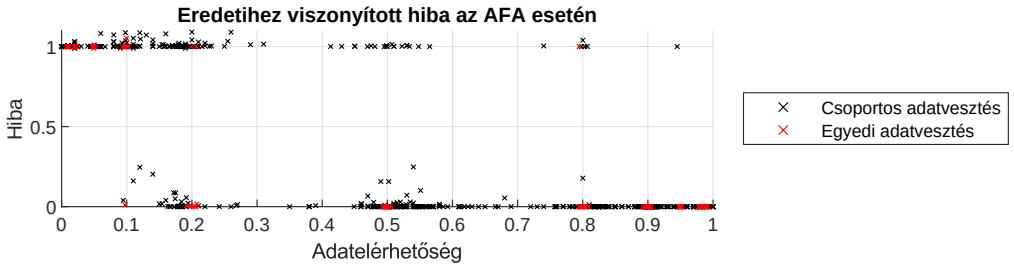
A helyes működést vizsgáló konfigurációkat láthatjuk az 5.13. ábrán. Rögtön észrevehető, hogy az RBO eredményei nagyrészt vonalakba rendeződnek, míg az AFA-nál ez nem tapasztalható. A nagyobb felbontású RBO-k kisebb pontosságot tudtak elérni, mint a kisebb felbontású társaik, ennek az az oka, hogy a nagyobb felbontás esetén a beállás lassabb – több minta szükséges ugyanakkora pontosság eléréséhez.



5.13. ábra. Az eredetihez viszonyított hiba a különböző modellalapú feldolgozó eljárások esetén

A kisebb felbontású RBO-k esetén a hiba nagyságát a jel-zaj viszony alapján két részre tudjuk választani, ezt szemlélteti az F.1.12. ábra. Ez választ ad arra a kérdésre is, hogy mi befolyásolja az RBO-k egyenesének „törését”: az exponenciális átlagolás az átlagolási együtthatótól függő mértékű zajelnyomást biztosít. Mivel ez az együttható a szimulációk során állandó volt, az állandósult hiba a zajszint adott része lesz. A két alkalmazott jel-zaj viszony közötti 60 dB-es különbség az állandósult hibában 6 nagyságrendnyi különbségnek felel meg, ez látható az ábrán is.

Az jel-zaj viszony szerinti szétválasztás az AFA esetén is megtehető (F.1.13. ábra), erről elmondható mindaz, ami az RBO-ról. Látható, hogy az AFA nem mindig működőképes, viszont amikor igen, akkor pontos eredményt ad. Az 5.14. ábra alapján az AFA közepes adatvesztésig is működőképes marad, feltéve ha az adatvesztés egyedi.

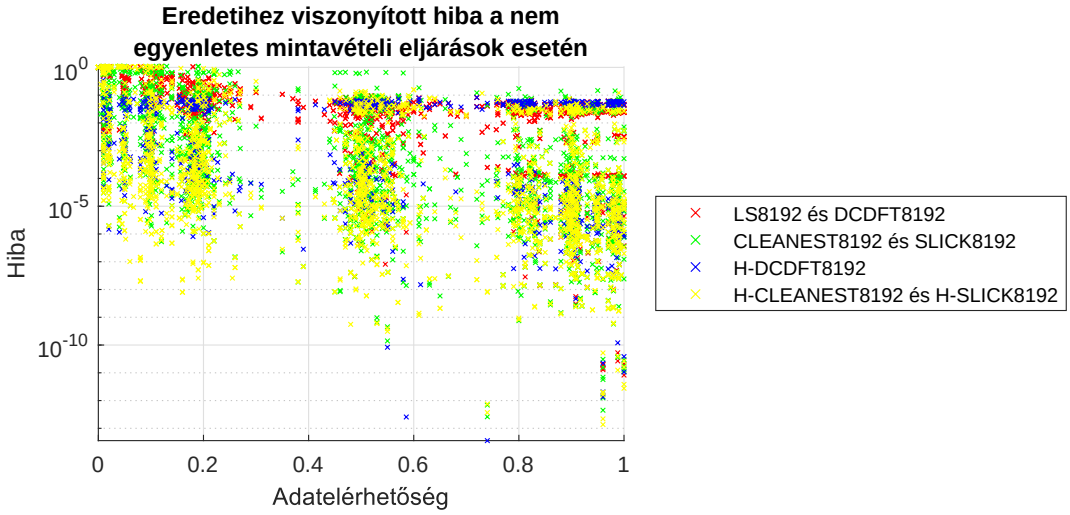


5.14. ábra. Az eredetihez viszonyított hiba az AFA esetén egyedi és csoportos adatvesztés esetén

5.2.1.3. A nem egyenletes mintavétel eljárásainak értékelése

A nem egyenletes mintavétel eljárásainak részletesebb vizsgálatánál is kizárjuk a helytelen működés adatait (F.1.14. ábra). Itt azokról az eredményekről van szó, amelyeket harmonikus DCDFT-vel számítottunk ki úgy, hogy egy periodikus komponenst kerestünk, holott a jel kváziperiodikus volt.

A helyes működéshez tartozó adatokat válogattam szét eljárások szerint az 5.15. ábrán. A nem iteratív eljárások helyenként vonalakat alkotnak, míg az iteratívak hibája széles tartományt fog közre.



5.15. ábra. Az eredetihez viszonyított hiba a különböző nem egyenletes mintavételi feldolgozó eljárások esetén

A Lomb-Scargle és a DCDFT eredményeit áttekintve (F.1.15. ábra) láthatunk több vonalat. Figyelembe véve, hogy az eljárásokat a 8192 pontos DFT frekvenciarácsán számítottuk ki és a kapott csúcsok körül nem alkalmaztunk rácsfinomítást, ismét az inkoherens mintavétel problémájával szembesülünk. Így a jeleket koherens, inkoherens és kváziperiodikus csoportokra osztva a vonalak szétválaszthatók (F.1.16. ábra). Ez a példa jól illusztrálja, hogy érdemes a kapott csúcsok környékén finomítani a rácsot, ezzel lényegesen jobb becslőt nyerhetünk. A következőkben ezért csak a koherensen mintavételezett jelek eredményeit vizsgáljuk.

Az F.1.17. ábra alapján a Lomb–Scargle-eljárás és a DCDFT eredménye a vizsgálatainkban gyakorlatilag azonos. Még itt is látunk egy vonalat. Ez a hibaszámítási mód

következménye: mivel komponensnek csak a legalább a legnagyobb csúcs nagyságának 1%-a kiemelkedésű csúcsokat választottuk, ezért a négyszögjel 11, 13 és 15-ös rendszámú komponenseit nem vettük figyelembe. Ez az elhanyagolás egy szükségszerű hibát okoz, ezt láthatjuk is az F.1.18. ábrán.

A CLEANEST és a SLICK esetén – mivel optimalizációval a frekvenciát a kezdeti rácspontok közé is be tudják vinni – az összes jelet vizsgáljuk. Az F.1.19. ábra alapján a SLICK legalább olyan jó eredményt produkál, mint a CLEANEST. Kisebb zaj esetén a hiba is kisebb, de ez látványos szétválasztást az eredmények között nem okoz. Ugyanígy nem okoznak szétválasztást az adatvesztési modellek, viszont a vizsgálójelek igen: kváziperiodikus jelek esetén a hiba nagyobb, mint periodikus jelek esetén (F.1.20. ábra). Figyelemre méltó az, hogy – periodikus jelek esetén – még erős adatvesztés mellett is van lehetőség viszonylag kis hibájú becslő elérésére.

A harmonikus DCDFT esetén is fontos, hogy a talált csúcsok környékén a frekvenciarácsot finomítsuk, ezt szemléleti az F.1.21. ábra. Rácsfinomítás nélkül az inkoherens mintavétel akkora hibát okoz, hogy amellett az adatvesztés nem számít. Ugyanis ilyenkor a frekvenciahiba a rendszámmal növekszik, így a felharmonikusokat már teljesen máshol keressük, mint ahol vannak. Ellenben elmondható, hogy amennyiben az alapfrekvencia helyes, akkor az eddigiek közül ez az egyik legstabilabb, adatvesztésre legkevésbé érzékeny becslő. A zaj növeli a hibát, de önálló sávot nem képez az eredményekből. Hasonlóképpen nincs sávképződés jelek vagy adatvesztési modellek szerint.

A harmonikus CLEANEST és a harmonikus SLICK eredményeit – az összes vizsgálójel esetén – mutatja be az F.1.22. ábra. Ez is szétválasztható a periodikus-kváziperiodikus vizsgálójelek mentén (F.1.23. ábra). A harmonikus és az egyszerű CLEANEST és SLICK eljárások ábrái szinte megegyeznek, ennek az oka az, az eljárások közötti lényegi különbség csak az új frekvenciák felvételének módjában található.

5.2.2. Önmagához viszonyított hiba

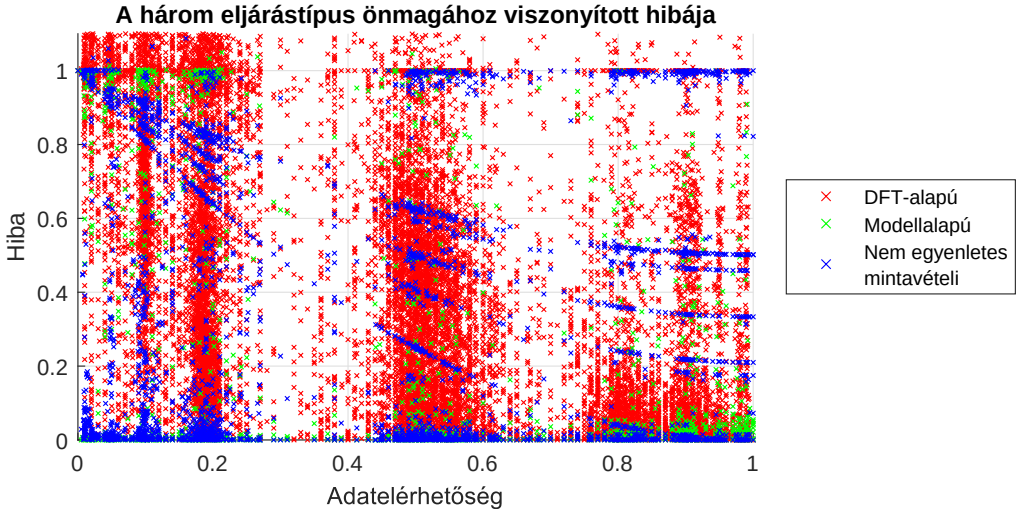
Az eljárások pontosságának vizsgálata után áttérünk arra a kérdésre, hogy milyen mértékben befolyásolja az eljárások kimenetét az adatvesztés. Az 5.16. ábrán az önmagához viszonyított hibát láthatjuk az adatelérhetőség függvényében. Ez az ábra is nagyított, a teljes ábrát és két gyengébb nagyítását a függelékben láthatjuk (F.1.24, F.1.25, és F.1.26. ábrák). Nem látunk az eredetihez viszonyított hibához hasonló sávszerkezetet, viszont itt egyes nem egyenletes mintavételes eredmények vonalakba rendeződnek.

Logaritmikus skálán ábrázolva (5.17. ábra) rögtön levonhatjuk a következtetést, hogy a modellalapú eljárások kimenetét befolyásolja a legkevésbé az adatvesztés.

5.2.2.1. A DFT-alapú eljárások értékelése

Az eredetihez viszonyított hibával szemben most a DFT-alapú eljárások eredményei között nem tesz lényegi különbséget az, hogy ablakoztunk-e, vagy sem (F.1.27. ábra).

A DFT-alapú eljárások között most az tesz különbséget, hogy milyen előfeldolgozási algoritmust alkalmaznak. A blokkosítás előtti előfeldolgozásokat összehasonlítva (F.1.28. ábra) azt kapjuk, hogy gyenge és közepes adatvesztés esetén a nullával helyettesítés, míg erős adatvesztés esetén a zero padding eljárás a kevésbé érzékeny. Másképp fogalmazva, a zero padding érzékenysége az adatvesztésre kevésbé változik. Ennek az az oka, hogy a zero padding esetén egy elveszett minta továbbiak eldobását okozhatja, így gyenge adatvesztés esetén több mintát dobunk el, mint amennyi elveszett. Erős adatvesztés esetén már nem marad annyi minta a DFT-blokkban, hogy abból a zero padding esetén spektrumot számítsunk, illetve ekkor a nullával helyettesítés nagymértékben tor-



5.16. ábra. Az önmagához viszonyított hiba az eljárástípusok szerinti bontásban

zítja a spektrumot. A blokkosítás előtti előfeldolgozás eljárásai között lényegi különbség nincs (F.1.29. ábra).

5.2.2.2. A modellalapú eljárások értékelése

A modellalapú eljárások érzékenységét csekély mértékben befolyásolja az adatvesztés (F.1.30.ábra). Vagyis legyen jó vagy rossz az adott eljárás által adott becslő, ezt az adatvesztés lényegesen nem módosítja.

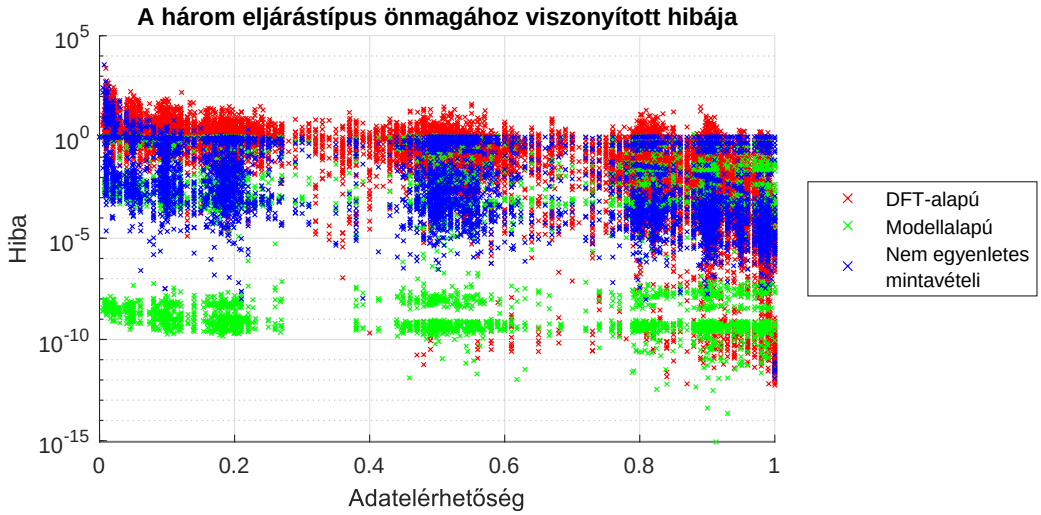
Ha csak a helyes működést ábrázoljuk (F.1.31. ábra), akkor hasonló elrendezésű ábrát kapunk, mint amikor az eredetihez viszonyított hibát vizsgáltuk (5.13. ábra). Itt azonban a vonalakat nem a különböző felbontású RBO-k okozzák, hanem a kétféle zajszint (F.1.32. ábra) – ugyanis az önmagához viszonyított hiba esetén a referencia a zaj- és adatvesztés nélküli jel. A két vonal között itt is megjelenik a 60 dB-es különbség.

Mivel az AFA adatvesztés nélkül – periodikus jelre – képes nagy pontossággal meghatározni a jel komponenseit, ezért az önmagához viszonyított hibája jó közelítéssel megegyezik az eredetihez viszonyítottal. Vagyis ha elkészítenénk hozzá az önmagához viszonyított hibára az 5.14. ábra megfelelőjét, az az előbb említett ábrától gyakorlatilag nem különbözne.

5.2.2.3. A nem egyenletes mintavételi eljárások értékelése

A nem egyenletes mintavételi eljárások 5.16. ábrán látott vonalai közül néhányat a kváziperiodikus jel harmonikus DCDFT-vel való feldolgozása ad (F.1.33. ábra). A teljes adatelérhetőségnél ezek a vonalak nem nulla hibához tartanak, hanem valamilyen véges értékhez, ennek kettős oka van. Egyrészt, a mintavétel (részben) inkoherens, ezért a felharmonikusokat rossz helyen keressük. Másrészt, a hibaszámítást itt időtartományban végeztem: a harmonikus DCDFT modellvektorát hasonlítottam az eredeti jelhez. Így a maximális 1%-ánál kisebb teljesítményű komponensek kihagyása okozta szükségszerű hibát látjuk az inkoherens felharmonikusok becslési hibájával együtt – ezeket nem befolyásolja az adatvesztés.

Ha a kváziperiodikus jel és harmonikus DCDFT kombináció kivételével ábrázoljuk az önmagához viszonyított hibát, az F.1.34. és az F.1.35. ábrákhoz jutunk. Itt rögtön



5.17. ábra. Az önmagához viszonyított hiba az eljárástípusok szerinti bontásban, logaritmikus skálán

láthatjuk, hogy további vonalakat adnak a harmonikus DCDFT eredményei, illetve a hiba szempontjából viszonylag elkülönülnek az egyes eljárások.

A Lomb-Scargle és a DCDFT között az önmagához viszonyított hiba szempontjából nincs különbség, a mintavétel koherenciája alapján nem képződnek jól elválasztható vonalak (F.1.36. ábra). Az előző ábra a két eljárás adatvesztésre való érzékenységének jellegét jól illusztrálja.

A CLEANEST és SLICK önmagához viszonyított hibája (F.1.37. ábra) nagyon hasonlít az eredetihez viszonyított hibájukra (F.1.19. ábra). A hasonlóság oka az, hogy itt is az eljárás modellvektora alapján történik az összehasonlítás. A SLICK hibája a kisebb a kettő közül. A két nagyobb hibás vonal egy részét a kváziperiodikus jelek okozzák.

A harmonikus DCDFT eredményeit a vizsgálójelek szerint csoportosítva (F.1.38. ábra) kiderül, hogy a nem nulla maradó hibájú vonalakat az inkoherens mintavétel okozta.

A harmonikus CLEANEST és SLICK esetén is hasonló az önmagához viszonyított hiba (F.1.39. ábra) és az eredetihez viszonyított hiba (F.1.22. ábra), szintén a modellvektoron alapuló hibaszámítás miatt. Az önmagához viszonyított hiba itt is szétbontható a kváziperiodikus és a periodikus jelek szerint (F.1.40. ábra).

5.2.3. Összegzés

A szimulációk tanulságai az alábbiakban foglalhatók össze:

- A vizsgált eljárások közül periodikus jelekre közepes adatvesztésig a modellalapú eljárások; erős adatvesztés vagy kváziperiodikus jel esetén minden jelre a nem egyenletes mintavétel eljárásai adták a legpontosabb becslőket.

- A DFT-alapú eljárások közepes vagy erős adatvesztés esetén valószínűleg nem fognak jó becslőt szolgáltatni.

- A túl kicsi DFT pontszám esetén különböző komponensek megkülönböztetése lesz lehetetlen, míg a túl nagy DFT pontszám esetén kevés blokkot tudunk átlagolni.

- A DFT használata esetén adatvesztés mellett is ablakozni kell.

- A DFT esetén az egyedi adatvesztés mellett várhatóan pontosabb becslőt kapunk, mint a csoportosnál.
- A DFT használata esetén igen gyenge adatvesztés mellett az interpolációs eljárások adják a legjobb eredményt, ekkor a különböző interpolációk pontosságbeli különbségei elhanyagolhatók.
- A DFT alkalmazásakor legalább gyenge, egyedi adatvesztés esetén a nullával helyettesítés, míg legalább gyenge, csoportos adatvesztés esetén a zero padding hozta a legjobb eredményt.
- Az RBO csak periodikus jelek esetén alkalmazható, akkor is csak ismert alapfrekvenciával. Ellenkező esetben a becslése helytelen.
- Minél nagyobb az RBO felbontása, állandó α együttható mellett annál tovább tart a beállása.
- Az RBO alkalmazható kváziperiodikus jelek egyes ismert frekvenciájú tagjai komponenseinek meghatározására.
- Az AFA csak periodikus jelek esetén alkalmazható, ellenkező esetben a becslése helytelen (legfeljebb egy tagot képes becsülni, ha annak rááll az alapfrekvenciájára).
- Az AFA egyedi, közepes adatvesztést is elvisel.
- A Lomb–Scargle-eljárás és a DCDFT gyakorlatilag azonos eredményeket hoztak.
- A Lomb-Scargle-eljárás, a DCDFT és a harmonikus DCDFT esetén ha megtaláltunk egy komponenst, érdemes a környezetét finomabb ráccsal újraszámolni, hogy pontosítsuk a becslést. Ez különösen érvényes a harmonikus DCDFT-re.
- Helyes alapfrekvencia mellett periodikus jel keresésére a harmonikus DCDFT volt a legalkalmasabb.
- (Kvázi)periodikus jelek esetén érdemes a harmonikus CLEANEST és SLICK eljárásokat alkalmazni az „egyszerű” társaik helyett.

Az 5.3–5.6. táblázatokban az eddigieket eljáráskiválasztási szempontból foglaltuk össze. Az, hogy az adatvesztés egyedi vagy csoportos, illetve, hogy valós idejű feldolgozást szeretnénk-e vagy sem, meghatározza a megfelelő táblázatot. Ha nem tudjuk, hogy az adatvesztés csoportos-e, akkor érdemes a csoportos táblázatokat választani.

Ha kiválasztottuk a megfelelő táblázatot, akkor a jel típusától és az adatvesztés mértékétől függően kiválaszthatjuk a megfelelő feldolgozó eljárást. A táblázatokban a H-SLICK(f_0) eljárás az f_0 frekvenciáról indított harmonikus SLICK algoritmust jelenti, a többi rövidítés értelmezéséhez az 5.2. táblázat ad segítséget. Az adatvesztés mértékére adott kvalitatív meghatározások értelmezésében a szimulációs eredmények elején ismertetett tartományok az irányadók.

Az eljáráskiválasztást nemcsak táblázatos formában ábrázolhatjuk. Készíthető belőle döntési fa is (5.18. ábra), amely a négy táblázat adatait együtt tartalmazza.

Jel	**Egyedi adatvesztés**			
	Nagyon gyenge	Gyenge	Közepes	Erős/ismeretlen mértékű
Periodikus, f_0 ismert, állandó	RBO	RBO	RBO	H-SLICK(f_0)
Periodikus, f_0 ismeretlen	AFA	AFA	AFA	H-SLICK
Kváziperiodikus	H-SLICK	H-SLICK	H-SLICK	H-SLICK
Ismeretlen	SLICK	SLICK	SLICK	SLICK

5.3. táblázat. Eljárás kiválasztása egyedi adatvesztés és nem valós idejű feldolgozás esetére

Jel	**Egyedi adatvesztés**			
	Nagyon gyenge	Gyenge	Közepes	Erős/ismeretlen mértékű
Periodikus, f_0 ismert, állandó	RBO	RBO	RBO	ZST-DFT
Periodikus, f_0 ismeretlen	AFA	AFA	AFA	ZST-DFT
Kváziperiodikus	ZOH-DFT	ZST-DFT	ZST-DFT	ZST-DFT
Ismeretlen	ZOH-DFT	ZST-DFT	ZST-DFT	ZST-DFT

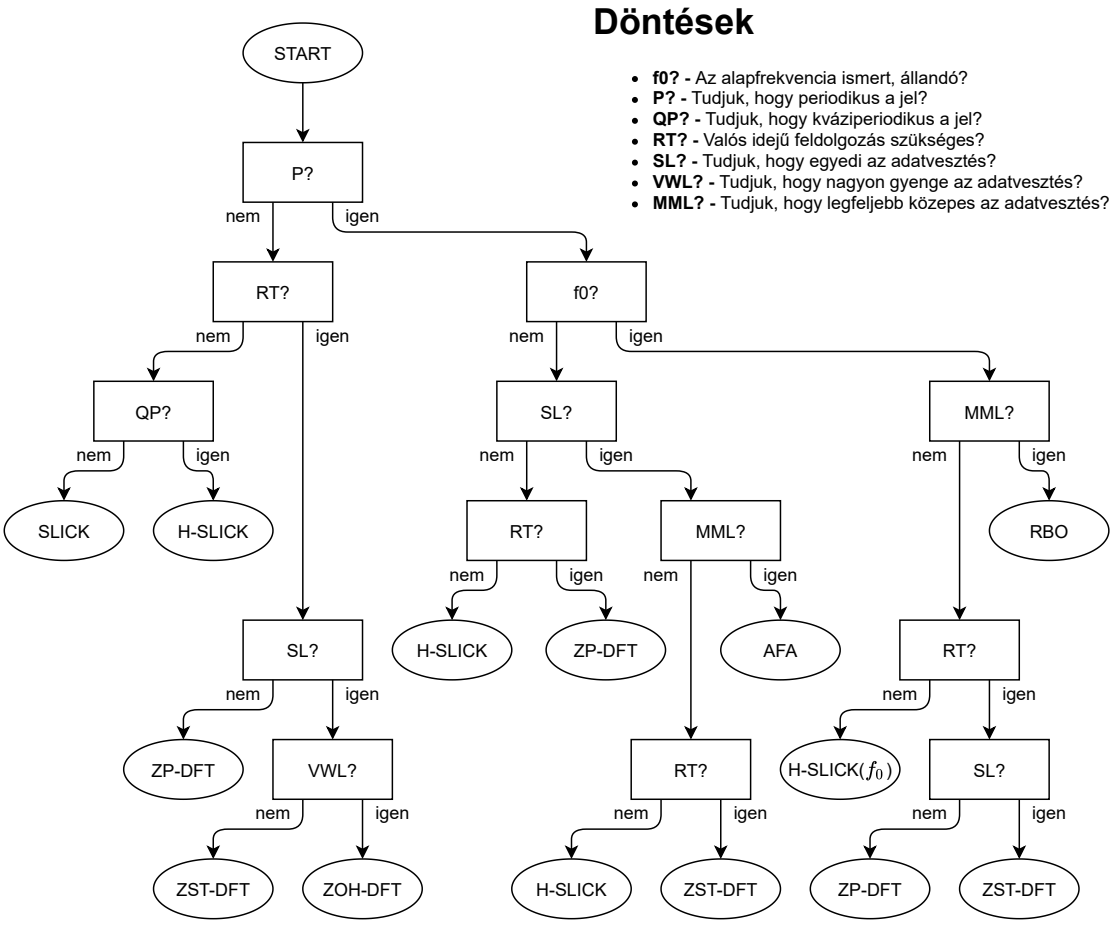
5.4. táblázat. Eljárás kiválasztása egyedi adatvesztés és valós idejű feldolgozás esetére

Jel	**Csoportos adatvesztés**			
	Nagyon gyenge	Gyenge	Közepes	Erős/ismeretlen
Periodikus, f_0 ismert, állandó	RBO	RBO	RBO	H-SLICK(f_0)
Periodikus, f_0 ismeretlen	H-SLICK	H-SLICK	H-SLICK	H-SLICK
Kváziperiodikus	H-SLICK	H-SLICK	H-SLICK	H-SLICK
Ismeretlen	SLICK	SLICK	SLICK	SLICK

5.5. táblázat. Eljárás kiválasztása csoportos vagy ismeretlen adatvesztés és nem valós idejű feldolgozás esetére

Jel	**Csoportos adatvesztés**			
	Nagyon gyenge	Gyenge	Közepes	Erős/ismeretlen
Periodikus, f_0 ismert, állandó	RBO	RBO	RBO	ZP-DFT
Periodikus, f_0 ismeretlen	ZP-DFT	ZP-DFT	ZP-DFT	ZP-DFT
Kváziperiodikus	ZP-DFT	ZP-DFT	ZP-DFT	ZP-DFT
Ismeretlen	ZP-DFT	ZP-DFT	ZP-DFT	ZP-DFT

5.6. táblázat. Eljárás kiválasztása csoportos vagy ismeretlen adatvesztés és valós idejű feldolgozás esetére



5.18. ábra. Eljáráskiválasztási döntési fa

6. fejezet

Összefoglalás, kitekintés

Munkám során áttekintettem az adatvesztés jelenségét, a spektrumbecslési feladatot és meglévő eljárásokat a probléma megoldására.

Az adatvesztés sokoldalú jelenség: jelentkezhet hiányzó mintákként, érvénytelen mintákként, vagy szinkronizációs problémákként is. Tipikus oka lehet a rádiós interferencia, egy csomag elvesztése az interneten, vagy valamilyen külső körülmény, amely befolyásolja a mérést.

Az adatvesztés leírható az indikátorfüggvény segítségével, mely minden mintára megmutatja, hogy érvényes-e, továbbá megközelíthető a nem egyenletes mintavétel irányából: az adatvesztés egy speciális nem egyenletes mintavételezés. A kétféle megközelítés észrevehető a spektrumbecslési eljárásokban is.

A spektrumbecslési eljárások alapvetően három csoportra oszthatók: DFT-alapú eljárások, melyek estén az indikátorfüggvény megközelítését használjuk, és jellemzően DFT-vel számítjuk a spektrumot; modellalapú módszerek, melyek a jelet valahogyan modellezik, és ezt használják fel az eljárás kiindulási alapjaként; és a nem egyenletes mintavételre kifejlesztett eljárások.

Az eljárásokat összehasonlítottam elméleti oldalról: hardverigény, vizsgált frekvenciák, numerikus problémák. Ezek alapján a valós idejű megvalósíthatóságról tettem megállapításokat. Általános tapasztalat, hogy minél nagyobb hardverigényű egy eljárás, annál pontosabb becslőt képes szolgáltatni, de annál nagyobb a numerikus problémák kialakulásának esélye is.

Az eljárásokat megvalósítottam, és egy MATLAB-alapú szimulációs környezet segítségével vizsgáltam. A vizsgálatot különböző jelek és adatvesztések mellett végeztem el, kétféle hibakritériummal. Az eredmények értékelésével választ kaptam arra, hogy mikor melyik eljárást érdemes alkalmazni. A tapasztalatok alapján eljáráskiválasztási javaslatokat tettem.

A dolgozat újdonságként mutatta be a vezérelt adaptív Fourier-analizátor, a harmonikus DCDFT, CLEANEST és SLICK eljárásokat. Mindnél a cél az elv „proof-of-concept” jellegű bemutatása volt, az eljárások továbbfejlesztése a jövő feladata.

Szintén további feladat lehet a vizsgálatba még több eljárás beemelése, például a DFT-s eljárások ismertetésénél említett ARFIL [24]. Ez az ARMA-ML [25] eljárás továbbfejlesztése, vele a spektrumbeli lejtőket lehet megtalálni (a dolgozat fókusza a spektrumbeli csúcsok megtalálásán volt).

Ábrák jegyzéke

Táblázatok jegyzéke

Irodalomjegyzék

[1] A. Palkó, „Adatvesztési modellek vizsgálata idő- és frekvenciatartományban," *Budapesti Műszaki és Gazdaságtudományi Egyetem*, TDK-dolgozat, 2018. [Online]. Available: https://tdk.bme.hu/VIK/ViewPaper/Adatvesztesi-modellek-vizsgalata-ido-es

[2] E. N. Gilbert, „Capacity of a burst-noise channel," *Bell System Technical Journal*, vol. 39, no. 5, pp. 1253–1265, 1960. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1538-7305.1960.tb03959.x

[3] E. O. Elliott, „Estimates of error rates for codes on burst-noise channels," *Bell System Technical Journal*, vol. 42, pp. 1977–1997, Sept. 1963.

[4] G. Haßlinger and O. Hohlfeld, „Analysis of random and burst error codes in 2-state Markov channels," in *2011 34th International Conference on Telecommunications and Signal Processing (TSP)*, Aug 2011, pp. 178–184.

[5] J. Rachwalski, *Analysis of Packet Loss Pattern for Concatenated Transmission Channesl Using Burst Ratio Parameter.* AGH University of Science and Technology, PhD thesis, 2016.

[6] S. Fine, Y. Singer, and N. Tishby, „The hierarchical hidden Markov model: Analysis and applications," *Machine Learning*, vol. 32, no. 1, pp. 41–62, Jul 1998. [Online]. Available: https://doi.org/10.1023/A:1007469218079

[7] P. Boufounos, „Generating binary processes with all-pole spectra," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07*, vol. 3, April 2007, pp. III–981–III–984.

[8] M. Ellis, D. P. Pezaros, T. Kypraios, and C. Perkins, „A two-level markov model for packet loss in UDP/IP-based real-time video applications targeting residential users," *Computer Networks*, vol. 70, pp. 384 – 399, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128614002205

[9] S. Lawlor and M. G. Rabbat, „Time-varying mixtures of Markov chains: An application to road traffic modeling," *IEEE Transactions on Signal Processing*, vol. 65, no. 12, pp. 3152–3167, June 2017.

[10] P. Welch, „The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms," *IEEE Transactions on Audio and Electroacoustics*, vol. 15, no. 2, pp. 70–73, June 1967.

[11] F. J. Harris, „On the use of windows for harmonic analysis with the discrete Fourier transform," *Proceedings of the IEEE*, vol. 66, no. 1, pp. 51–83, Jan 1978.

[12] L. Sujbert and G. Orosz, „FFT-based spectrum analysis in the case of data loss," *IEEE Transactions on Instrumentation and Measurement*, pp. 968–976, Jan. 2016.

[13] G. Plantier, S. Moreau, L. Simon, J.-C. Valière, A. L. Duff, and H. Bailliet, „Nonparametric spectral analysis of wideband spectrum with missing data via sample-and-hold interpolation and deconvolution," *Digital Signal Processing*, vol. 22, no. 6, pp. 994 – 1004, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1051200412001376

[14] G. Péceli, „A common structure for recursive discrete transforms," *Circuits and Systems, IEEE Transactions on*, vol. 33, pp. 1035 – 1036, 11 1986.

[15] G. Orosz, L. Sujbert, and G. Péceli, „Analysis of resonator-based harmonic estimation in the case of data loss," *Instrumentation and Measurement, IEEE Transactions on*, vol. 62, pp. 510–518, 02 2013.

[16] F. Nagy, „Measurement of signal parameters using nonlinear observers," *Instrumentation and Measurement, IEEE Transactions on*, vol. 41, pp. 152–155, 03 1992.

[17] F. Nagy, „An adaptive Fourier analysis algorithm," in *5th International Conference on Signal Processing Applications and Technology*, 1994, pp. 414–418.

[18] L. Eyer and P. Bartholdi, „Variable stars: Which nyquist frequency?" *Astronomy and Astrophysics Supplement Series*, vol. 135, no. 1, p. 1–3, Feb 1999. [Online]. Available: http://dx.doi.org/10.1051/aas:1999102

[19] G. Foster, „The cleanest Fourier spectrum," *ASTRONOMICAL JOURNAL*, vol. 109, pp. 1889–1902, Apr. 1995.

[20] N. R. Lomb, „Least-squares frequency analysis of unequally spaced data," *Astrophysics and Space Science*, vol. 39, no. 2, pp. 447–462, Feb 1976. [Online]. Available: https://doi.org/10.1007/BF00648343

[21] J. D. Scargle, „Studies in astronomical time series analysis. III - Fourier transforms, autocorrelation functions, and cross-correlation functions of unevenly spaced data," *Astrophysical Journal, Part 1*, vol. 343, pp. 874–887, Aug. 1989.

[22] G. Bretthorst, *Bayesian Spectrum Analysis and Parameter Estimation*. Lecture Notes in Statistics (Berlin: Springer), 1988.

[23] S. Ferraz-Mello, „Estimation of Periods from Unequally Spaced Observations," *ASTRONOMICAL JOURNAL*, vol. 86, p. 619, Apr. 1981.

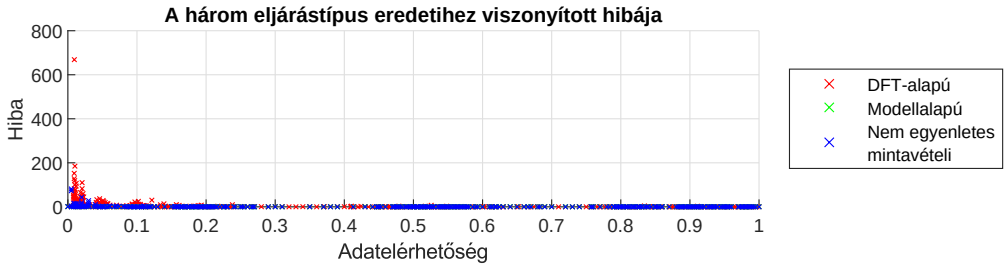
[24] P. M. T. Broersen, S. de Waele, and R. Bos, „Estimation of autoregressive spectra with randomly missing data," in *Proceedings of the 20th IEEE Instrumentation Technology Conference (Cat. No.03CH37412)*, vol. 2, May 2003, pp. 1154–1159 vol.2.

[25] R. H. Jones, „Maximum likelihood fitting of ARMA models to time series with missing observations," *Technometrics*, vol. 22, no. 3, pp. 389–395, 1980. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/00401706.1980.10486171

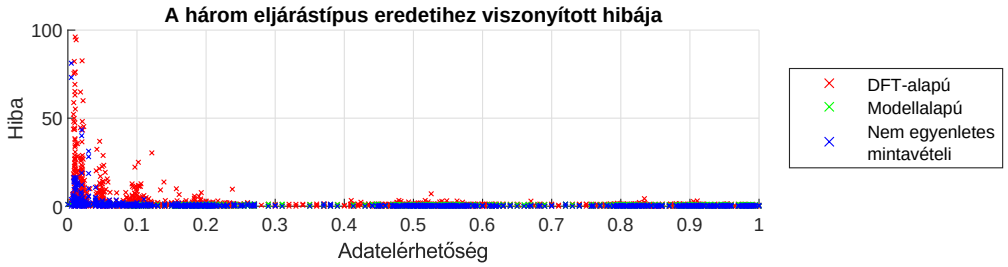
Függelék

F.1. Szimulációs eredmények ábrái

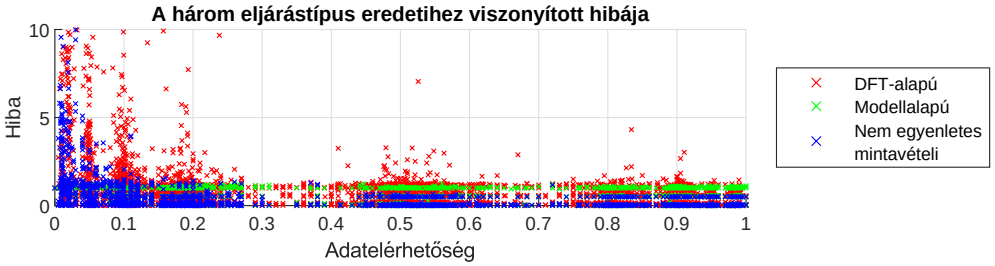
F.1.1. Eredetihez viszonyított hiba



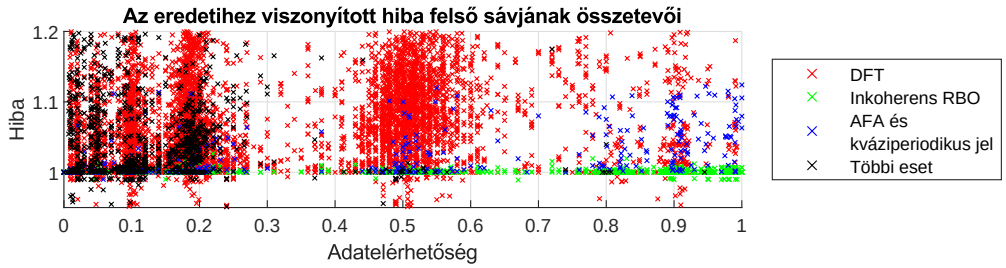
F.1.1. ábra. Az eredetihez viszonyított hiba az eljárástípusok szerinti bontásban (teljes)



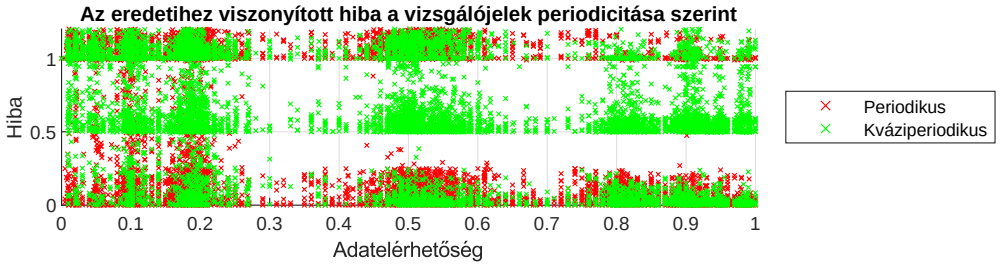
F.1.2. ábra. Az eredetihez viszonyított hiba az eljárástípusok szerinti bontásban (nagyítás)



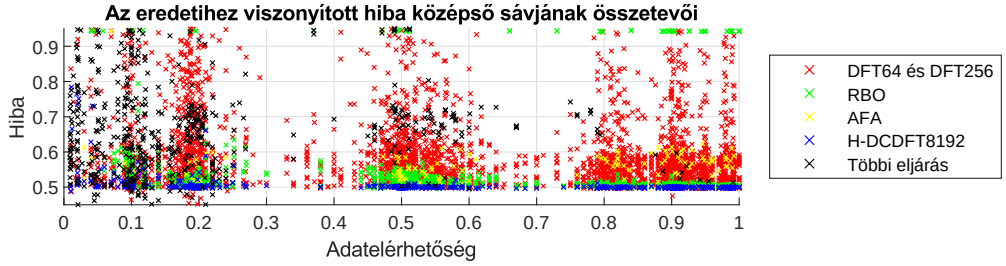
F.1.3. ábra. Az eredetihez viszonyított hiba az eljárástípusok szerinti bontásban (erősebb nagyítás)



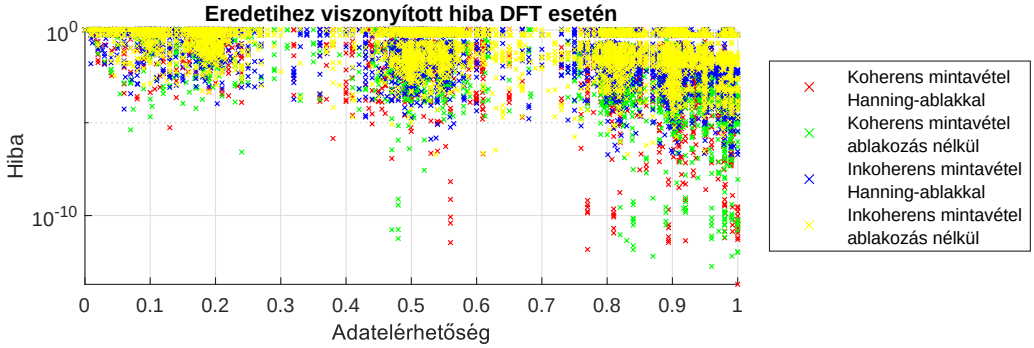
F.1.4. ábra. Az eredetihez viszonyított hiba felső sávjának összetevői



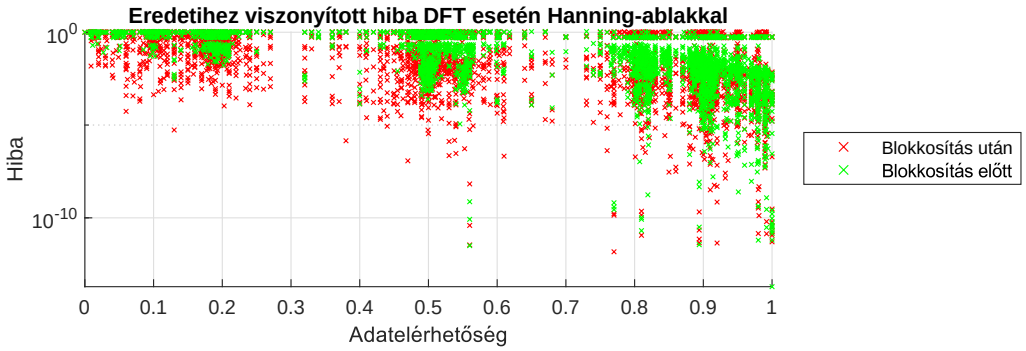
F.1.5. ábra. Az eredetihez viszonyított hiba a vizsgálójelek periodicitása szerint



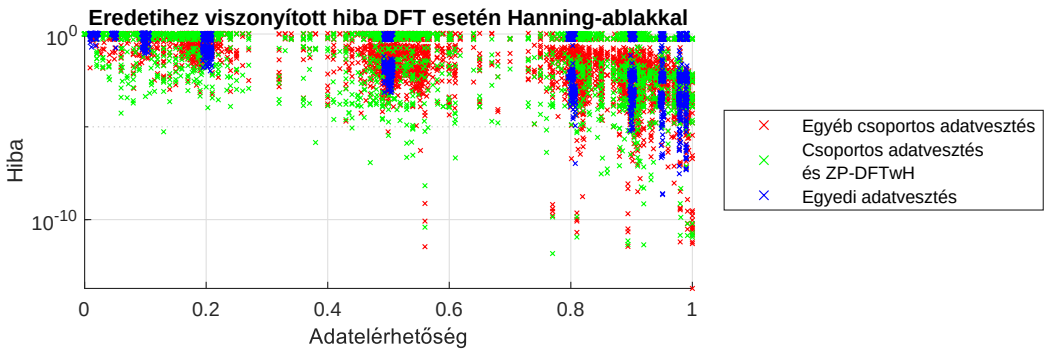
F.1.6. ábra. Az eredetihez viszonyított hiba középső sávjának összetevői



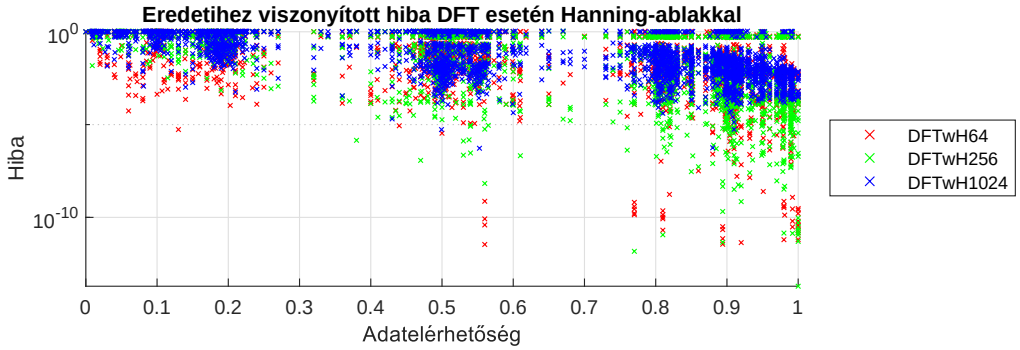
F.1.7. ábra. Az eredetihez viszonyított hiba a DFT esetén: ablakozás hatása



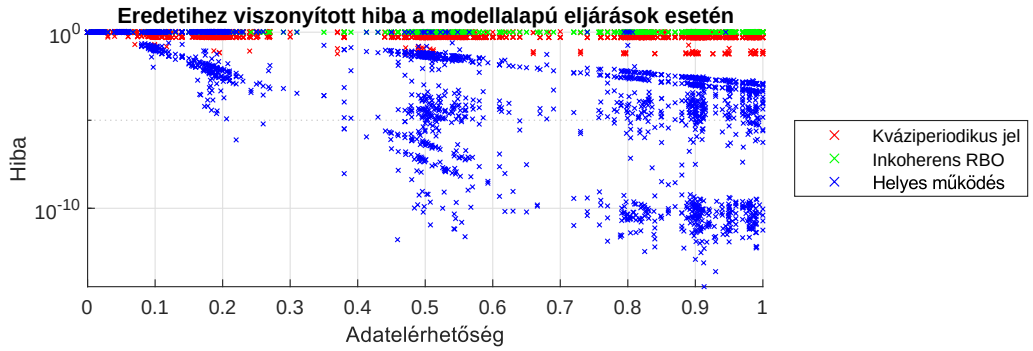
F.1.8. ábra. Az eredetihez viszonyított hiba az előfeldolgozás típusa szerint



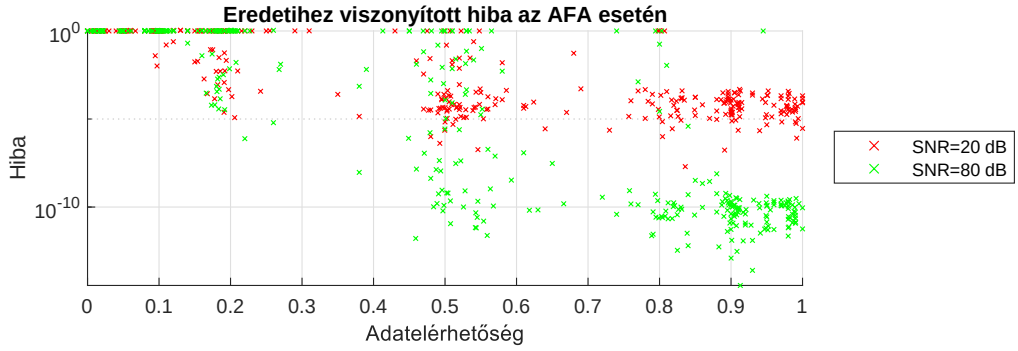
F.1.9. ábra. Az eredetihez viszonyított hiba az adatvesztés jellege (egyedi, csoportos) szerint



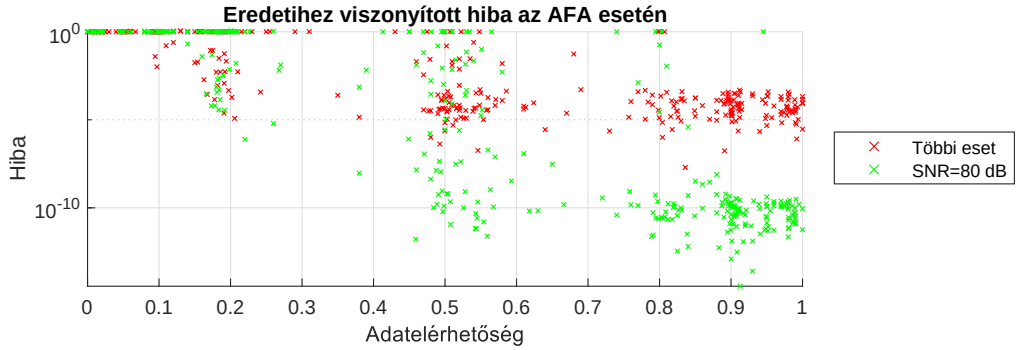
F.1.10. ábra. Az eredetihez viszonyított hiba a DFT pontszáma szerint



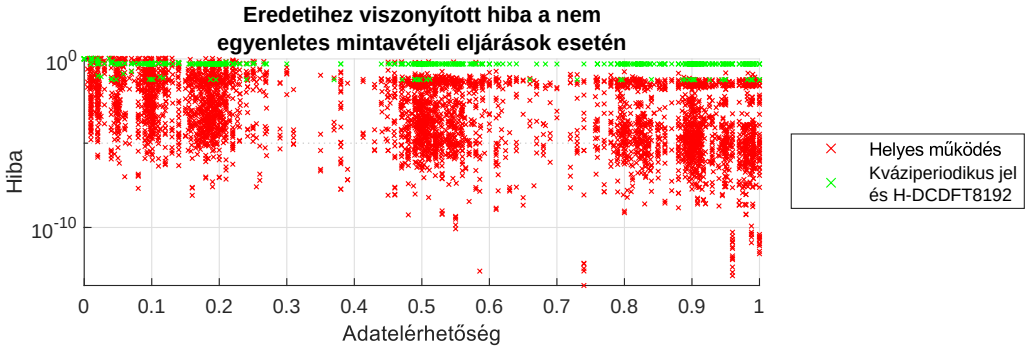
F.1.11. ábra. Az eredetihez viszonyított hiba a modellalapú eljárások esetén: a működésképtelenség okai



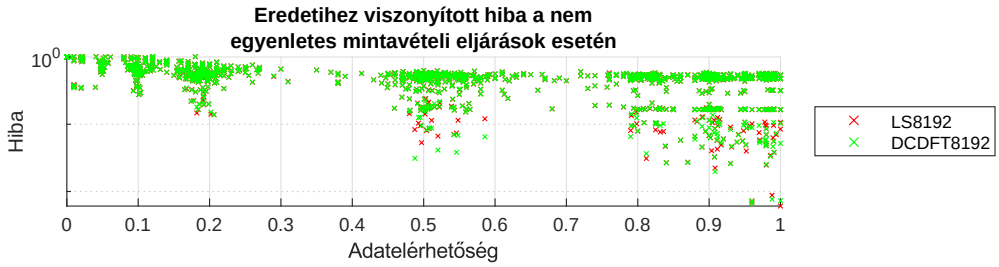
F.1.12. ábra. Az eredetihez viszonyított hiba a kisebb felbontású RBO-k esetén a jel-zaj viszony szerint



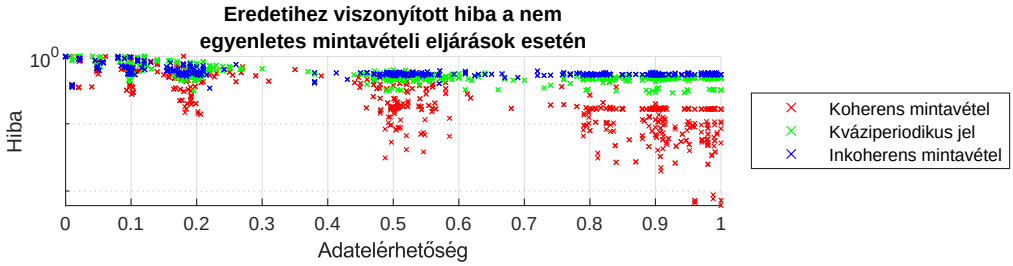
F.1.13. ábra. Az eredetihez viszonyított hiba az AFA esetén a jel-zaj viszony szerint



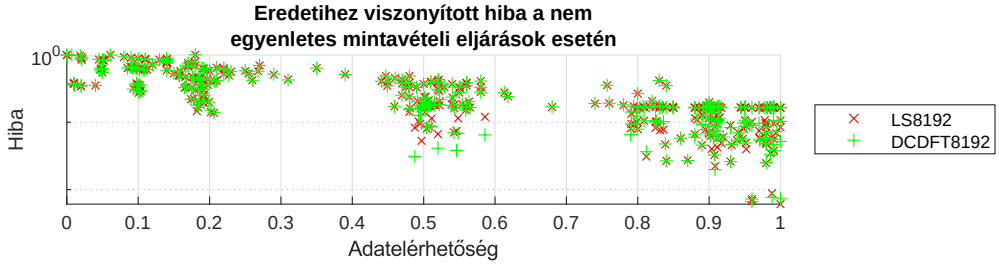
F.1.14. ábra. Az eredetihez viszonyított hiba a nem egyenletes mintavételi eljárások esetén: a működésképtelenség okai



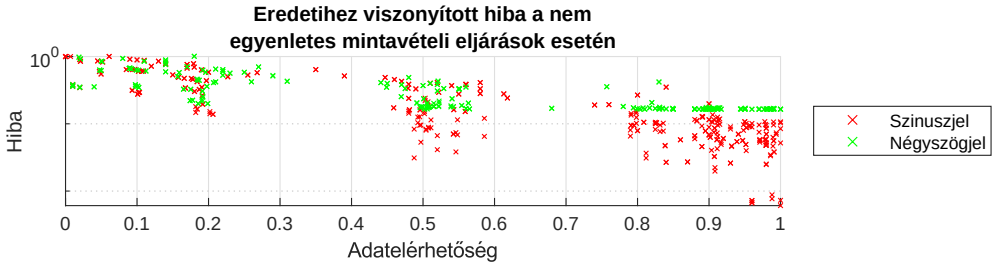
F.1.15. ábra. Az eredetihez viszonyított hiba a Lomb-Scargle és a DCDFT feldolgozó eljárások esetén



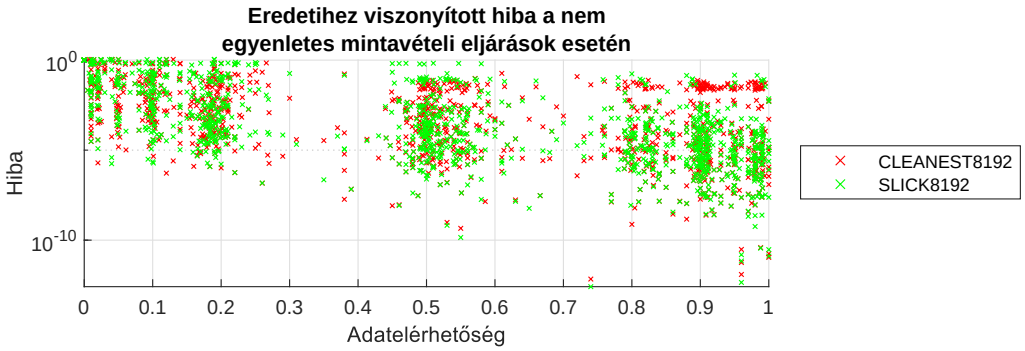
F.1.16. ábra. Az eredetihez viszonyított hiba a Lomb-Scargle és a DCDFT feldolgozó eljárások esetén: koherencia



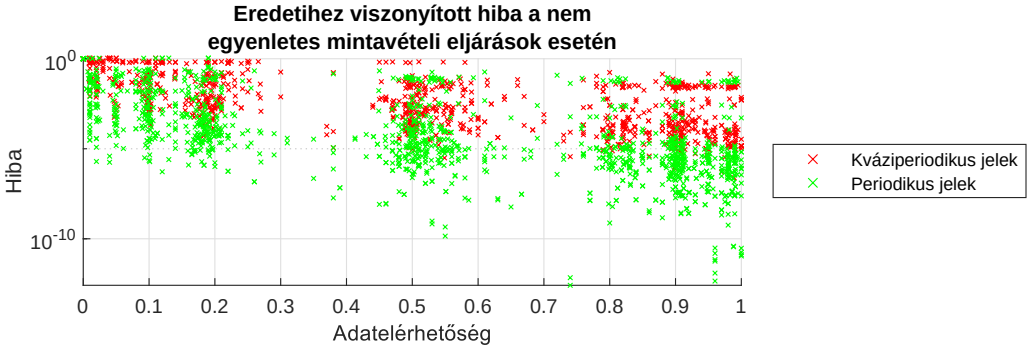
F.1.17. ábra. Az eredetihez viszonyított hiba a Lomb-Scargle és a DCDFT feldolgozó eljárások használatával, koherens mintavétel esetén



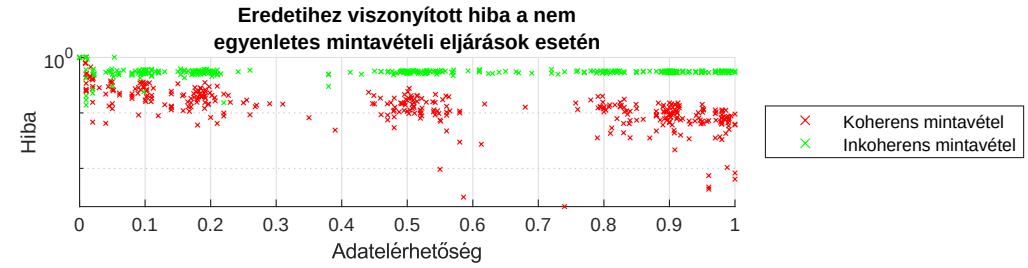
F.1.18. ábra. Az eredetihez viszonyított hiba a Lomb-Scargle és a DCDFT feldolgozó eljárások használatával, koherens mintavétel esetén: felbontás a vizsgálójelek szerint



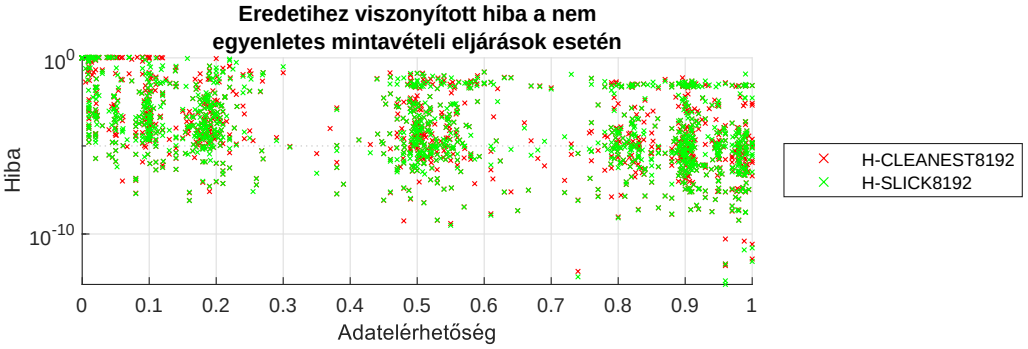
F.1.19. ábra. Az eredetihez viszonyított hiba a CLEANEST és a SLICK feldolgozó eljárások esetén



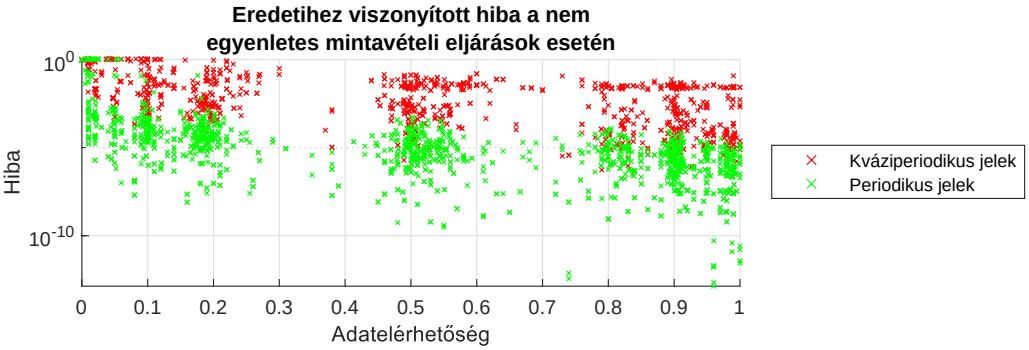
F.1.20. ábra. Az eredetihez viszonyított hiba a CLEANEST és a SLICK feldolgozó eljárások esetén



F.1.21. ábra. Az eredetihez viszonyított hiba a harmonikus DCDFT eljárás használatával, koherens és inkoherens mintavétel esetén

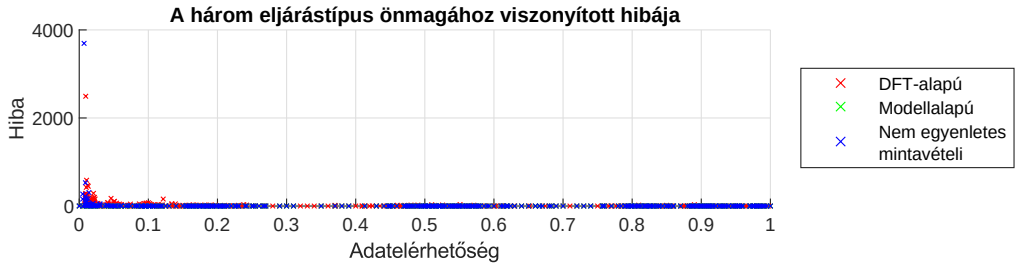


F.1.22. ábra. Az eredetihez viszonyított hiba a harmonikus CLEANEST és a harmonikus SLICK feldolgozó eljárások esetén

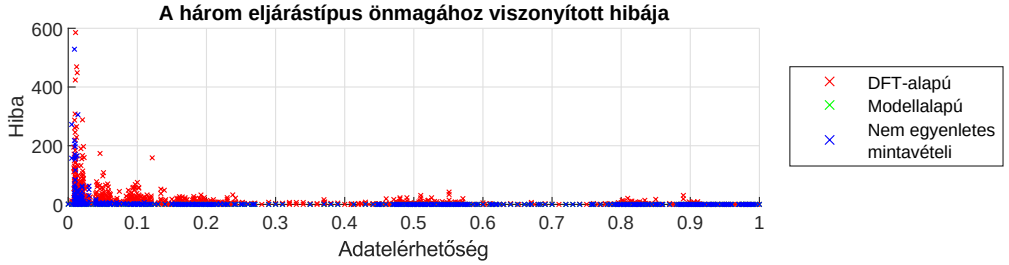


F.1.23. ábra. Az eredetihez viszonyított hiba a harmonikus CLEANEST és a harmonikus SLICK feldolgozó eljárások esetén

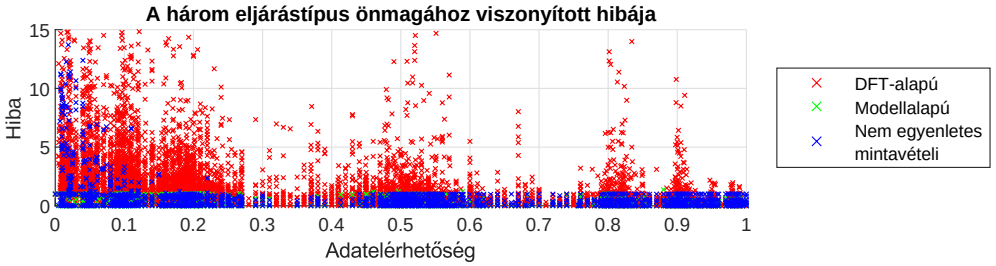
F.1.2. Önmagához viszonyított hiba



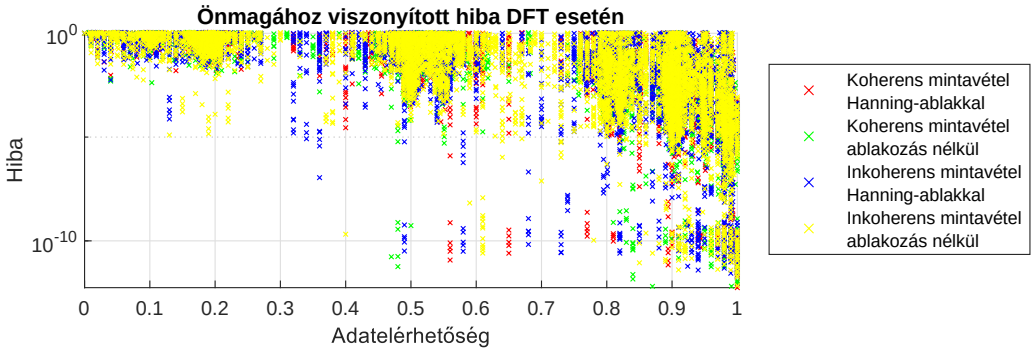
F.1.24. ábra. Az önmagához viszonyított hiba az eljárástípusok szerinti bontásban (teljes)



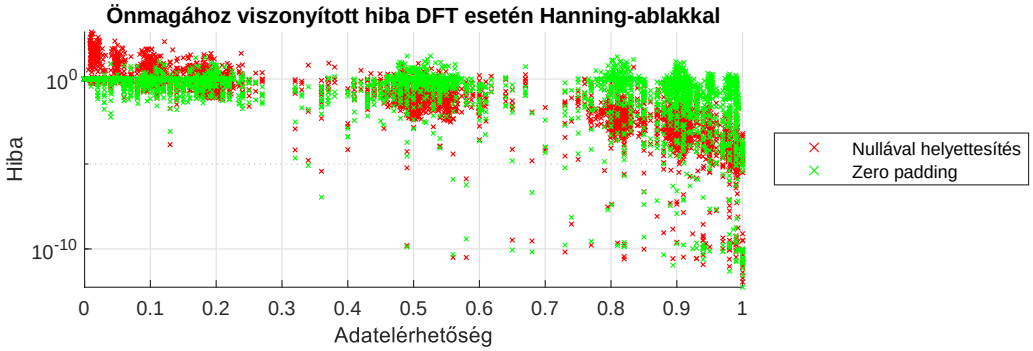
F.1.25. ábra. Az önmagához viszonyított hiba az eljárástípusok szerinti bontásban (nagyítás)



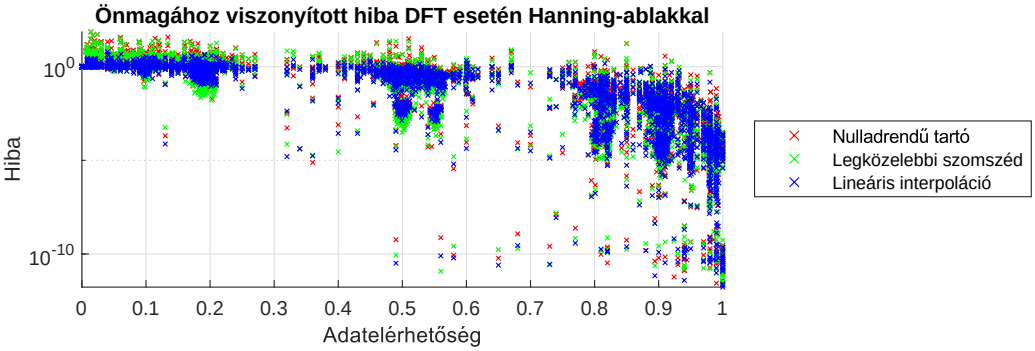
F.1.26. ábra. Az önmagához viszonyított hiba az eljárástípusok szerinti bontásban (erősebb nagyítás)



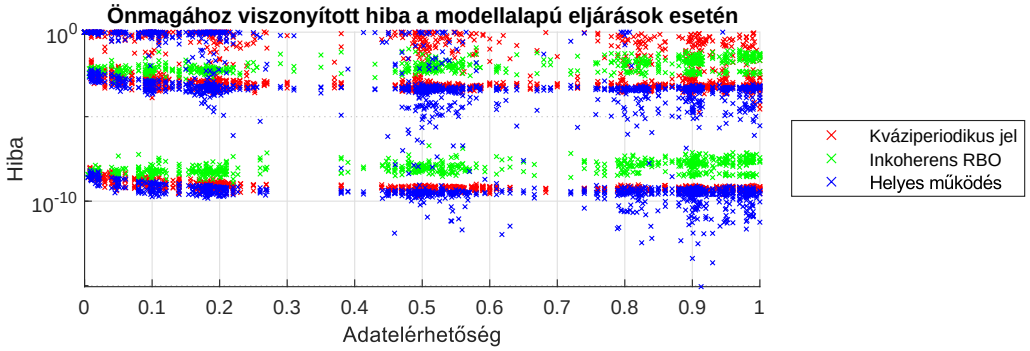
F.1.27. ábra. Az önmagához viszonyított hiba a DFT esetén



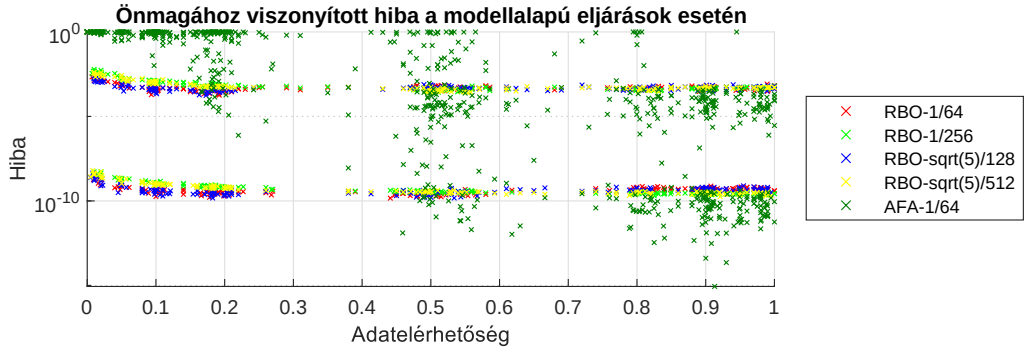
F.1.28. ábra. Az önmagához viszonyított hiba a DFT esetén ablakozás nélkül, a blokkosítás utáni előfeldolgozások használatával



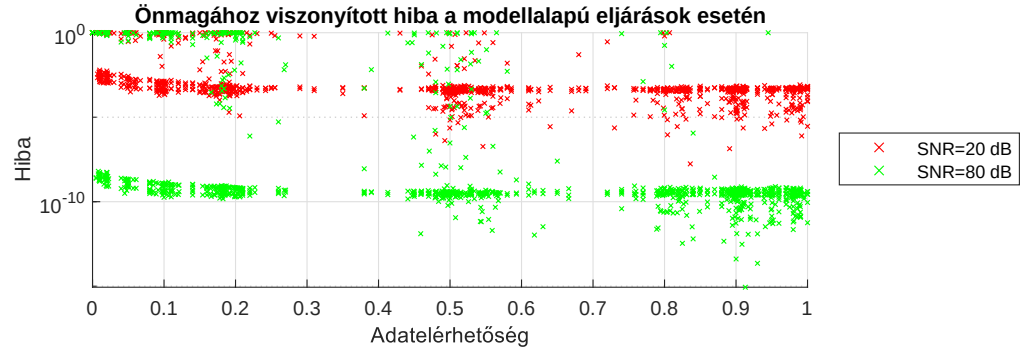
F.1.29. ábra. Az önmagához viszonyított hiba a DFT esetén ablakozás nélkül, a blokkosítás előtti előfeldolgozások használatával



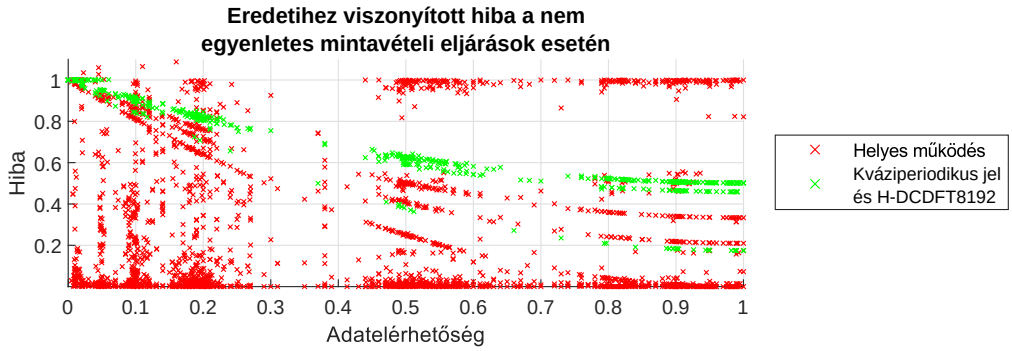
F.1.30. ábra. Az önmagához viszonyított hiba a modellalapú eljárások esetén, különböző vizsgálójelek mellett



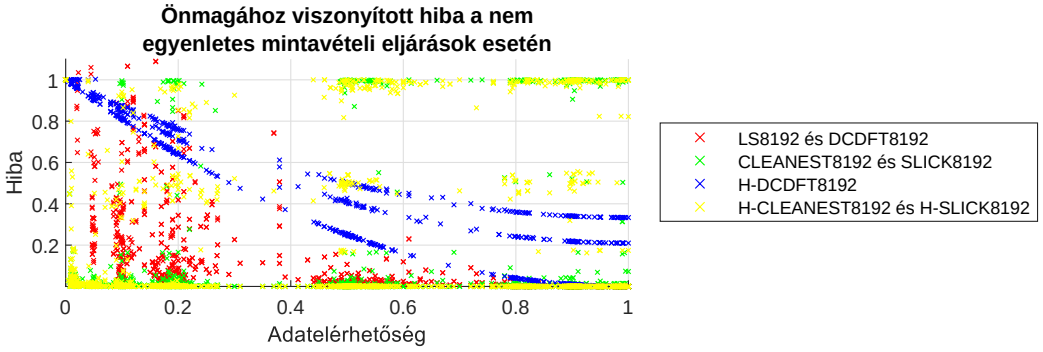
F.1.31. ábra. Az önmagához viszonyított hiba a modellalapú eljárások helyes működése esetén, eljárások szerint



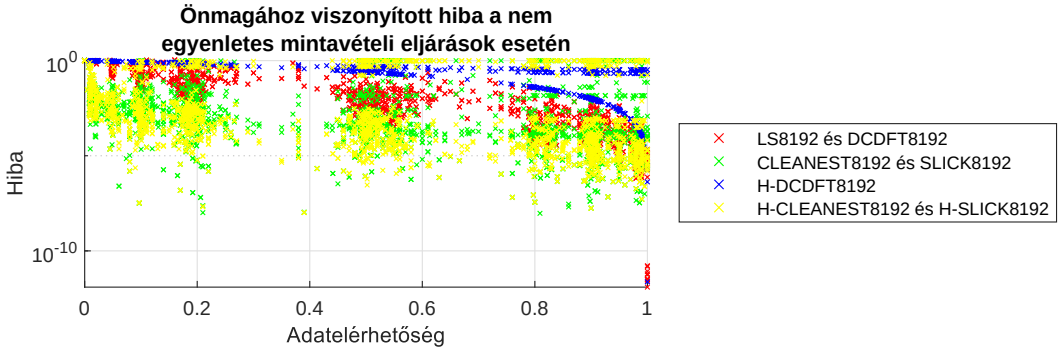
F.1.32. ábra. Az önmagához viszonyított hiba a modellalapú eljárások helyes működése esetén, jel-zaj viszony szerint



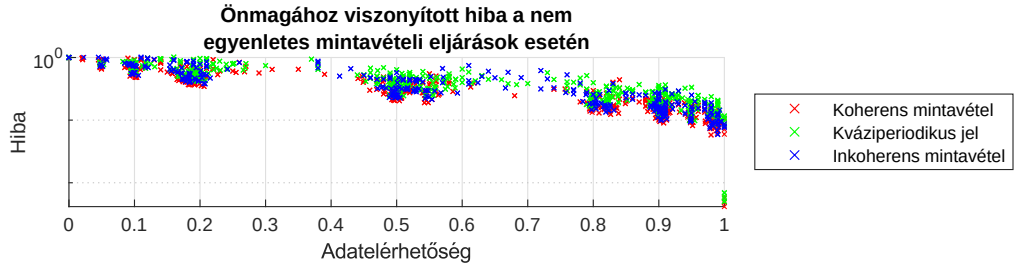
F.1.33. ábra. Az önmagához viszonyított hiba a nem egyenletes mintavételi eljárások esetén, különböző vizsgálójelek mellett



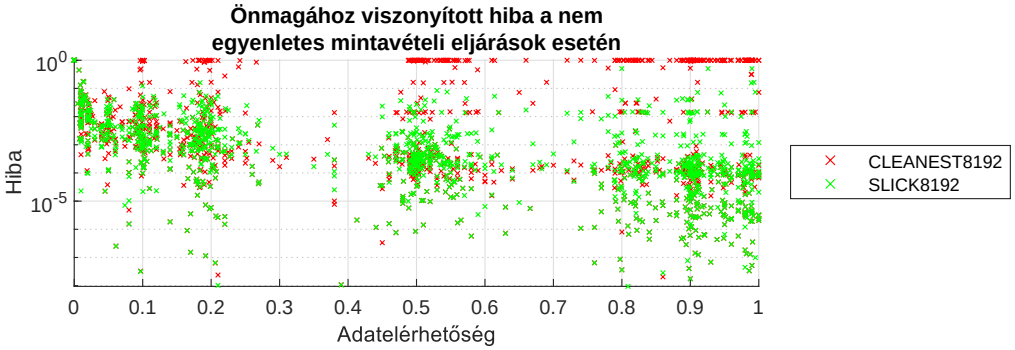
F.1.34. ábra. Az önmagához viszonyított hiba a különböző nem egyenletes mintavételi feldolgozó eljárások esetén



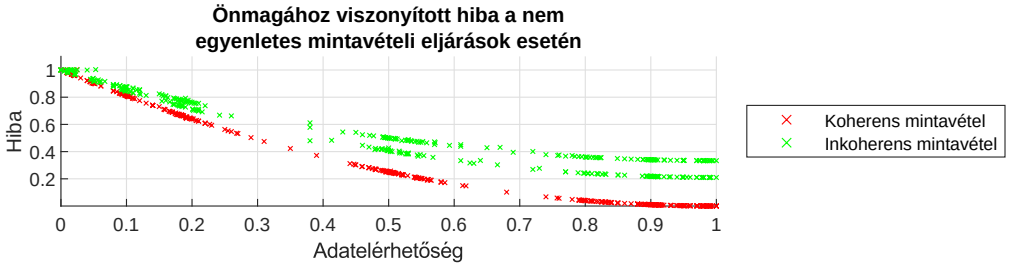
F.1.35. ábra. Az önmagához viszonyított hiba a különböző nem egyenletes mintavételi feldolgozó eljárások esetén, logaritmikus skálán



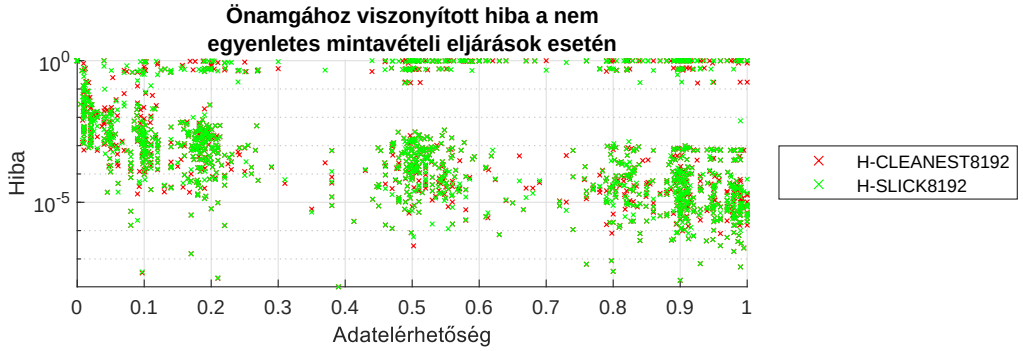
F.1.36. ábra. Az önmagához viszonyított hiba a Lomb-Scargle és a DCDFT feldolgozó eljárások esetén: koherencia



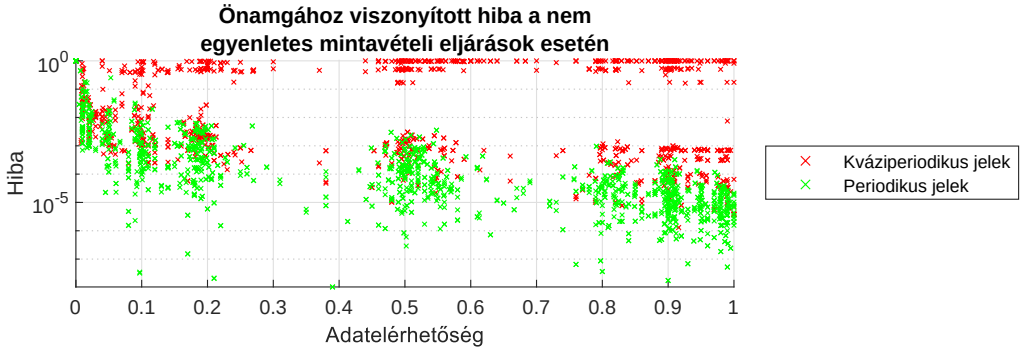
F.1.37. ábra. Az önmagához viszonyított hiba a CLEANEST és a SLICK feldolgozó eljárások esetén



F.1.38. ábra. Az önmagához viszonyított hiba a harmonikus DCDFT eljárás használatával, koherens és inkoherens mintavétel esetén



F.1.39. ábra. Az önmagához viszonyított hiba a harmonikus CLEANEST és a harmonikus SLICK feldolgozó eljárások esetén



F.1.40. ábra. Az önmagához viszonyított hiba a harmonikus CLEANEST és a harmonikus SLICK feldolgozó eljárások esetén