Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Szélessávú Hírközlés és Villamosságtan Tanszék

Önálló laboratórium 2 dolgozat

Modell-redukció alkalmazása az elektromágneses térszámításban

Szilágyi Gábor

Konzulens: Dr. Bilicz Sándor

Budapest, 2022. december 1.

1. Bevezetés

A POD, vagyis a Proper Orthogonal Decomposition, egy bázistranszformációs eljárás, ami egy adott adathalmaz reprezentálásához optimális bázist keres meg. Az eljárás által meghatározott Ψ bázisban a bázisvektoroknak az a tulajdonsága, hogy a lehető legkevesebb bázisvektorral leírható az adathalmaz információtartalmának vagy energiájának lehető legnagyobb része. Ezt felhasználva a Ψ csonkításával egy Ψ' közelítő bázist lehet előállítani, ami lényegesen kisebb rendű, mint Ψ, mégis kis hibával reprezentálható benne az eredeti adathalmaz. Természetesen minél több bázisvektort hagyunk meg Ψ'-ben, annál jobban csökken a modell-redukcióból származó hiba, de a csonkítás mértékét az adott alkalmazáshoz mérten előírhatjuk. A POD egy másik előnyös tulajdonsága, hogy a gyakorlati esetek nagy részében a sorbarendezett Ψ_n bázisvektorokra eső energiatartalom a legtöbb esetben rohamosan csökken, ezért sokszor nagyságrendekkel kisebb dimenziószámú bázissal is jól leírható az adathalmaz, mint az eredeti esetben.

1.1. A módszer eredete

A módszer gyökerei az '40-es évekig nyúlnak vissza, amikor olyan tudományos problémákat próbáltak statisztikai módszerekkel megközelíteni, amelyeknek a megoldása valamilyen folytonos értékű függvény, sok esetben sztochasztikus folyamat. Egy jeles személy ebből a kezdeti időszakból Damodar Dharmananda Kosambi indiai matematikus [9], aki a POD alapját képező (Kosambi–)Karhunen–Loève Expansion, vagyis Karhunen–Loève felbontás atyja. A Karhunen–Loève felbontás sztochasztikus folyamatok olyan reprezentálását teszi lehetővé, ahol a sztochasztikus folyamatot egy végtelen összeg alakjában írjuk fel. Ha ebből az összegből csak az első véges sok n db. tényezőt hagyjuk meg (csonkítjuk a felbontást), akkor a kapott összegnek a lehető legkisebb a négyzetes hibája az eredeti sztochasztikus folyamathoz képest [2].

A POD klasszikus formájában végtelen dimenziós vektorterek, például függvényterek fölött működik. Valójában numerikusan ennek egy véges dimenziós vektortereken értelmezett megfelelőjét, a szinguláris érték szerinti felbontást (SVD) lehet használni, a féléves munkám során én is ezt az eljárást használtam. A végtelen és véges dimenziós változatok között az átjárást a Galerkin-vetítés teremti meg [7].

1.2. Felhasználási területekről általában

Az eljárást számos tudományterületen sikeresen alkalmazták már, különböző területeken különböző néven szokták emlegetni gyakorlatilag ugyanezt a módszert. Statisztikában főként Principal Component Analysis (PCA) néven fordul elő és nagy adathalmazok információtartalmának kinyerésére használják [8]. Ahogy fent említettem, találkozhatunk még vele az irodalomban Karhunen–Loève Expansion néven is, elsősorban a sztochasztikus folyamatok kapcsán [2]. A turbulens áramlások kutatásával kapcsolatban legtöbbször POD-ként említik [10].

Lényeges felhasználási területek a fentieken kívül például: szabályozástechnikában a szuboptimális, de gyorsan számítható beavatkozás; nemlineáris elektromágneses problémák (pl. motorok) szimulációja [6]; genetikában a DNS jellegzetes mintázatainak keresésében [4].

1.3. Problémakörök

A felsorolt problémák, amelyeknél segítséget nyújthat a POD, különböző kategóriákba sorolhatóak.

Elsősorban statisztikában merül fel az a gond manapság, hogy egy nagyon sok bejegyzést tartalmazó adathalmaz áll rendelkezésre, amelynek minden bejegyzése sokdimenziós, emiatt az

adatok által hordozott információ nehezen hasznosítható. Ez az információkinyerés annak ellenére is problémás, hogy a sok rendelkezésre álló adat összesen sok információt hordoz magában. Ekkor a POD (PCA) segítségével kinyerhetők az adathalmazt jól leíró, jellegzetes mintázatok, amelyek a hasznos információ nagy részét magukban hordozzák.

A szimulációk esetében más a helyzet. Itt sokszor az jelenti a gondot, hogy a szimulált modell nagyon részletes, például nagyon sok (könnyen 10^3 – 10^6 nagyságrendű) végeselemet tartalmaz, emiatt a szimuláció futtatása sok számítógép erőforrást igényel. Egyrészt a megoldás memóriában való tárolása is problémás lehet, másrészt a végeselemek módszerénél maradva nagyon nagy egyenletrendszert kell megoldani, ami sok processzoridőt igényel, ezért sokáig tart a szimuláció. Ekkor a POD segítségével a ténylegesen megoldandó egyenletrendszer és a megoldás méretét is nagyságrendekkel csökkenteni lehet. Ez úgy történhet, hogy valamilyen előzetes szimuláció eredményét felhasználva az adott modellhez tartozó jellegzetes részmegoldások lineáris kombinációjaként igyekszünk előállítani egy kisebb szabadsági fokú, közelítő megoldást, aminek sokkal kisebb az erőforrásigénye.

1.4. Alkalmazás az elektromágneses térszámításban

Az EM térszámításban többféle kontextusban is hasznos lehet a POD a futási idő vagy a memóriafelhasználás jelentős csökkentésére. Az egyik megközelítésben egy időtartománybeli végeselem modellben zajló tranziens folyamat lefolyására kaphatunk számítás szempontjából olcsó, közelítő megoldást. Ehhez először a teljes kérdéses időintervallum első töredék részére egy teljes értékű szimulációt futtatunk, amely viszonylag sok számítást igényel. Ennek a rövid részmegoldásnak az eredményei szolgálnak a POD bemenetéüli. A POD ezek alapján meghatározza a rendszer dinamikájában megjelenő struktúrákat, majd csak a lényeges összetevőkre szorítkozva egy lecsökkentett szabadsági fokú rendszert szimulálunk tovább a hátralévő időben, ami már időlépésenként sokkal kevesebb számítást igényel.

A fent vázolt, tranziens szimulációban történő alkalmazáson kívül más módokon is hasznosítható lehet a POD a térszámítási problémákban, de a dolgozatomban elsősorban ezzel a megközelítéssel foglalkozom. Az alkalmazási lehetőségeket az korlátozza elsősorban, hogy a megoldásnak újrafelhasználhatónak kell lennie egy szimuláció vagy részszimuláció után egy másik futás során. Például egy végeselem szimulációnál ha a térbeli struktúrát megváltoztatjuk, akkor ez majdnem szükségszerűen azzal jár, hogy megváltozik a végeselem-háló, ami miatt már nem igazán lehet újrafelhasználni a korábbi hálóra kapott megoldást. A végeselem szimulációknál a háló mozgásának, változásának lekezelése a Moving Mesh problémakörébe tartozik, a dolgozatomban ezzel nem foglalkozom.

Egy lényeges potenciális felhasználási terület lehet a széles frekvenciasávon történő frekvenciatartománybeli szimuláció. Ilyen problémáknál a végcél elsősorban valamilyen analitikus módszerekkel kezelhetetlen elrendezés frekvenciafüggő átviteli karakterisztikájának meghatározása egy relatíve széles frekvenciasávban. A relatíve széles frekvenciasáv alatt nem feltétlenül a nagy relatív sávszélességet értem, hanem azt, hogy a sávon belül sok diszkrét frekvencián kellene szimulációt végezni, majd ezeknek az eredményeiből kellene interpolálni az átviteli karakterisztikát. Ennek a sok pontban történő szimulációnak a kiváltására már eddig is előszeretettel alkalmaztak időtartománybeli tranziens szimulációt, aminek az eredményét Fourier-transzformálva meg lehet kapni a rendszer átviteli karakterisztikáját. Ebben az ekvivalens időtartománybeli szimulációban lehetne alkalmazni a dolgozatban demonstrált POD módszert. Ez a felhasználási mód felveti azt a fontos kérdést, hogy a redukált rendű szimuláció hogyan hat a frekvenciatartománybeli végeredményre, mennyire képes azt meghamisítani.

Egy másik potenciális felhasználási terület lehet az anyagparaméterekre való érzékenység vizsgálata. Rádiófrekvenciás alkalmazásoknál gyakran előkerülő probléma, hogy a használt

szubsztrát dielektromos állandója gyártóról gyártóra változik, esetleg egy adott gyártónál is inkonzisztens és ez a legyártott áramkör teljesítőképességét nagyban ronthatja. Ha egy számításigény szempontjából olcsó becslést lehet adni arra, hogy az adott eszköz mennyire érzékeny a szubsztrát dielektromos állandójára, az segítene az anyagparaméterekre érzéketlenebb eszközök tervezésében. Azért lehet az anyagparaméterek megváltozását könnyen vizsgálni a POD segítségével, mert a változtatás nem jár egy végeselem modell esetén a háló megváltoztatásával és feltehetőleg az új anyagparaméterek melletti megoldás jellegre hasonló, mint az eredeti, így a bázisvektorok újrafelhasználhatóak.

2. A POD-ról részletesebben

A bevezetésben említettem, hogy az adathalmaz energiájának vagy információtartalmának szempontjából optimális a Ψ bázis, de ez pontosításra szorul. A POD által megadott, rendezett Ψ_n bázisvektoroknak az a tulajdonsága, hogy ha csak az első r bázisvektor által kifeszített lineáris altérre vetítjük az adathalmazt, akkor a vetítés utáni adathalmaznak a lehető legkisebb az összes négyzetes hibája az eredeti adatokhoz képest. Ez bármilyen r-re igaz. Más szóval a Ψ bázis bármilyen r rendű Ψ' csonkítása least-squares értelemben optimális. [8]

2.1. A felbontás egyenletei

A redukálandó adathalmaz méreteitől függően különböző számítási módok optimálisak a POD redukált bázisának előállításához. Az eredeti adathalmazt egy mátrixba rendezzük úgy, hogy az egyes mintákhoz artozó adatok vektorai ($\mathbf{x}_i \in \mathbb{C}^n,\ i \in \{1,\ \ldots,\ k\}$) a mátrix oszlopvektorai legyenek. Az így kapott mátrix ($\mathbf{S}$) a snapshotmátrix vagy mintamátrix:

$$\mathbf{S} = [\mathbf{x}_1\ \mathbf{x}_2\ \mathbf{x}_3\ \ \ldots\ \ \mathbf{x}_k] \tag{1}$$

Az adott alkalmazástól függ, hogy $\mathbf{S}$-nek melyik mérete olyan nagy, hogy az gondot jelentsen. Az 1.4. részben vázolt nagy szabadsági fokú szimulációknál az okozza a gondot, hogy az egy időlépéshez tartozó $\mathbf{x}_i$ dimenziószáma (vagyis n) nagy, könnyen milliós nagyságrendű, miközben egy jó redukált bázis előállításához, $k \ll n$ db. $\mathbf{x}_i$ minta elég a dinamikus rendszerből. Ebben az esetben jól használható az $\mathbf{S}$ szinguláris érték szerinti felbontása (Singular Value Decomposition, SVD):

$$\mathbf{S}\ = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{H}} \tag{2}$$

2.2. A felbontás értelmezése

Itt érdemes megállni és értelmezni a felbontásban szereplő mátrixokat és azok jelentését a modellezett rendszerrel kapcsolatban. A 2. egyenletben $\mathbf{U} \in \mathbb{C}^{n\times n}$ lesz a teljes Ψ bázis vektorait ($\mathbf{u}_i,\ i \in \{1,\ \ldots,\ k\}$), mint oszlopvektorokat tartalmazó unitér mátrix. Az $\mathbf{U}$ mátrixnak csak az első k db. oszlopa hordoz információt az $\mathbf{S}$ mintamátrixszal kapcsolatban, így csak ezek az oszlopok tartoznak a Ψ bázisba. Az utolsó $n-k$ db. oszlop csak ahhoz kell, hogy $\mathbf{U}$ unitér mátrix legyen.

A $\mathbf{\Sigma} \in \mathbb{R}^{n\times k}$ egy diagonálmátrix, aminek az i-eik diagonáleleme, σ_i az $\mathbf{u}_i$-hez tartozó együtthható, ami azt fejezi ki, hogy az adott bázisvektor mennyire fontos, mennyire járul hozzá általában a rendszer állapotához, más szóval a rendszer energiájának vagy információtartalmának mekkora része írható le az adott bázisvektorral. Mivel $\mathbf{\Sigma}$ általános esetben nem négyzetes, de diagonális, ezért lesz egy olyan részmátrixa, ami csupa 0 értékekkel van feltöltve. Jelen esetben, vagyis ha a $n > k$, akkor az alsó $n-k$ db. sora csupa 0. A $\mathbf{\Sigma}$ olyan felépítésű az SVD miatt, hogy

a diagonálelemek csökkenő sorrendben szerepelnek az átlóban bal fentről jobbra lefelé haladva. A $\mathbf{U\Sigma} \in \mathbb{C}^{n\times k}$ szorzat tehát már az eredeti adatokhoz skálázott bázisvektorok mátrixának tekinthető, amelyek az oszlopaiban fontosság szerint csökkenő sorrendben szerepelnek. Ez a $\mathbf{U\Sigma}$ szorzat is olyan, hogy már csak $\mathbf{U}$ első k db. oszlopa szerepel benne σ_i-vel skálázva.

Tömören összefoglalva a $\mathbf{V}^{\mathrm{H}}$ mátrix a skálázott bázisvektorokhoz tartozó együtthatók mátrixa. A $\mathbf{V}^{\mathrm{H}}$ mátrix az i-edik $\mathbf{v}_i$ oszlopában tartalmazza az $\mathbf{x}_i$ mintavektor előállításához szükséges együtthatókat, amelyek a $\mathbf{U\Sigma}$ szorzat oszlopait súlyozzák:

$$\mathbf{x}_i = \mathbf{U\Sigma v}_i \tag{3}$$

Más szemszögből megközelítve a $\mathbf{V}^{\mathrm{H}}$ értelmezését, $v_{j,i}$, tehát $\mathbf{V}^{\mathrm{H}}$ j-edik sorának i-edik eleme ($\mathbf{v}_i$ j-edik eleme) jelenti $\mathbf{u}_j\sigma_j$ hozzájárulását vagy súlyát $\mathbf{x}_i$-hez.

Az ellenkező esetben, amikor $k \gg n$, már számításigény szempontjából nem optimális $\mathbf{S}$ teljes SVD-jének direkt kiszámolása, mert ez az immár $k \times k$ méretű $\mathbf{V}^{\mathrm{H}}$ mátrix kiszámolásával jár, ami a redukált modell létrehozása szempontjából lényegtelen. Ebben az esetben érdemesebb lenne kiszámítani az adathalmaz kovarianciamátrixát, a $\mathbf{C} \in \mathbb{C}^{n\times n}$ mátrixot. Azért jutunk ezzel előrébb, mert a 4. egyenletben látható módon az $\mathbf{U}$ mátrixot megkaphatjuk a $\mathbf{C}$ spektrálfelbontásából is. Ezzel a módosított eljárással Method of Snapshots néven találkozhatunk az irodalomban [3]. Mivel $\mathbf{C}$ szimmetrikus, ezért pozitív definit, emiatt mindig létezik spektrálfelbontása.

$$\begin{aligned} \mathbf{C} &= \mathbf{SS}^{\mathrm{H}} \\ &= \mathbf{U\Sigma V}^{\mathrm{H}}\mathbf{V\Sigma U}^{\mathrm{H}} \\ &= \mathbf{U\Sigma}^2\mathbf{U}^{\mathrm{H}} \end{aligned} \tag{4}$$

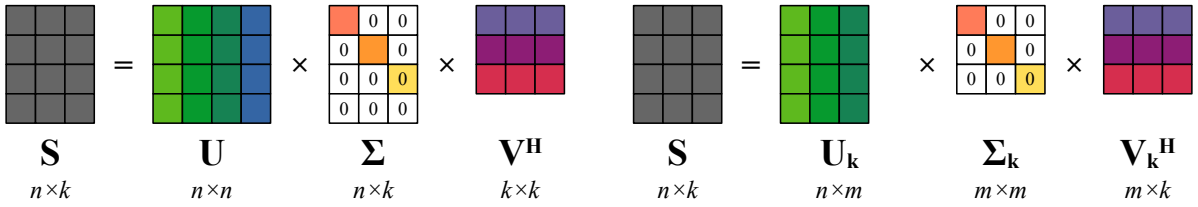
Itt $\mathbf{S}$ a snapshot-mátrix, $\mathbf{U}$ oszlopai az új bázisvektorok, $\mathbf{\Sigma}$ tartalmazza a bázisvektorok információtartalmát jellemző szinguláris értékeket a főátlójában, $\mathbf{V}^{\mathrm{H}}$ sorai az egyes bázisvektorok időfüggő együtthatói, $\mathbf{C}$ pedig az $\mathbf{x}_i$ mintavektorokból álló adathalmaz kovarianciamátrixa.

2.3. Az SVD takarékos változatai

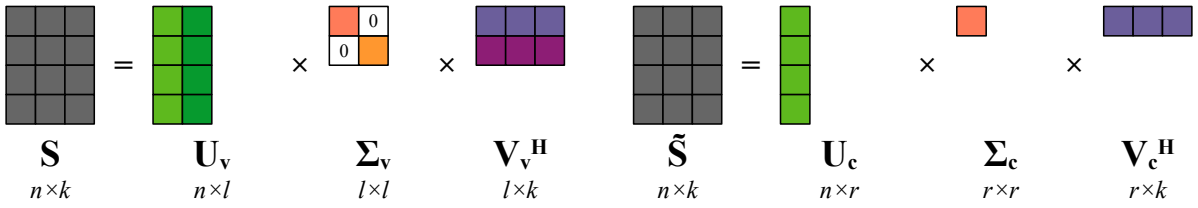
Az SVD algoritmusnak többféle változata ismert, amelyek mind műveletszám, mind memóriahasználat terén takarékosabbak a teljes SVD-nél. Ahogy már fent is említettem, a redukált bázis meghatározása szempontjából a $\mathbf{V}^{\mathrm{H}}$ mátrix kiszámolása érdektelen. Ezen kívül, ha előre adott egy határ, hogy legalább mekkora szinguláris értékhez tartozó bázisvektorokat tartsunk meg, akkor ennél a határnál kisebb szinguláris értékekhez tartozó oszlopok $\mathbf{U}$-ban, illetve sorok $\mathbf{V}^{\mathrm{H}}$-ban érdektelenek. Ez a határ a szinguláris értékekre vonatkozóan lehet abszolút vagy relatív is, célszerűen a legnagyobb szinguláris értékhez képest. A számítógépek véges pontosságú számábrázolása miatt az is előfordulhat, hogy egy felbontani kívánt mátrix rangja lényegesen kisebb, mint a numerikus ábrázolási hibákkal terhelt megfelelőjéé. Más szóval a numerikus rangja eltér a valódi rangtól. Ez indokolttá teszi egy hibahatár bevezetését, aminél kisebb számokat már csak numerikus hibának tekintünk és emiatt pontosan 0-val helyettesítjük őket.

Az SVD algoritmusnak többféle implementációja elérhető, némelyiknek pedig megadhatóak további paraméterek, amelyekkel például a fent vázolt, szinguláris értékekre vagy ábrázolási pontosságra vonatkozó határokat lehet megadni, továbbá egy fix redukált rangot is. Elvileg az is megadható, hogy az $\mathbf{U}$ és $\mathbf{V}^{\mathrm{H}}$ mátrixok közül melyikre van szükségünk, eszerint a nem szükségesnek megadott mátrix nem számolódik ki teljesen, ami egyes esetekben nagy csökkenéssel járhat erőforrásigény tekintetében [5].

A takarékos SVD változatok az 1. ábrán láthatóak olyan esetben, amikor $k < n$. Ezen belül a teljes SVD az, aminél $\mathbf{\Sigma}$ ugyanolyan méretű, mint a felbontott $\mathbf{S}$ mátrix, ezzel a változattal írtam fel előzőleg a mintamátrix felbontását (1a. ábra).



(a) Teljes SVD. (b) Kompakt SVD.



(c) Vékony SVD. (d) Csonkított SVD.

1. ábra. Az SVD különböző változatai [1].

A kompakt SVD a következőkben különbözik a teljes SVD-től. Éljünk a következő elnevezéssel: $m = \min(n, k)$. Az $\mathbf{U}$ mátrixból kihagyjuk azokat az oszlopokat, amelyekhez nem tartozik sem szinguláris érték, sem $\mathbf{V}^{\mathrm{H}}$-beli oszlop, és hasonlóan $\mathbf{V}^{\mathrm{H}}$ soraival is, de az $\mathbf{U}$ és $\mathbf{V}^{\mathrm{H}}$ mátrixok közül egyszerre csak az egyiknek a megcsonkítására lehet szükség n és k viszonyától függően. Ezen kívül a $\mathbf{\Sigma}$ mátrixnak is kihagyjuk a csupa 0 részét, amely az utolsó $n - m$ sora vagy az utolsó $k-m$ oszlopa, így egy $m \times m$ méretű diagonálmátrix lesz belőle, aminek minden diagonáleleme egy szinguláris érték. Ekkor a kompakt SVD-ben $\mathbf{U_k} \in \mathbb{C}^{n\times m}$, $\mathbf{\Sigma_k} \in \mathbb{C}^{m\times m}$, $\mathbf{V_k^{\mathrm{H}}} \in \mathbb{C}^{m\times k}$. A kompakt SVD a teljes változathoz képest nem jár információveszteséggel $\mathbf{S}$-re nézve. Az SVD-nek ez a változata használható MATLAB-ban, ha megadjuk az `svd()` függvénynek a `'econ'` opciót.

A vékony SVD egy következő redukcióval jön létre a kompakt változatból. A kompakt SVD esetén még a szinguláris értékek között lehetnek 0 értékűek, amelyek a $\mathbf{\Sigma_v}$ főátlójának utolsó elemei. Tegyük fel, hogy l db. nemnulla szinguláris érték van. A 0 szinguláris értékekhez tartozó $\mathbf{U_v}$-beli oszlopok és $\mathbf{V_v^{\mathrm{H}}}$-beli sorok szintén nem hordoznak hasznos információt, így ezektől is meg lehet szabadulni, valamint $\mathbf{\Sigma_v}$-ből is elég csak azt az $l \times l$-es bal felső részmátrixot megtartani, aminek a főátlójában a nemnulla szinguláris értékek vannak. Ekkor $\mathbf{U_v} \in \mathbb{C}^{n\times l}$, $\mathbf{\Sigma_v} \in \mathbb{C}^{l\times l}$, $\mathbf{V_v^{\mathrm{H}}} \in \mathbb{C}^{l\times k}$. Mivel itt már azt vizsgáljuk, hogy a szinguláris értékek 0 értékűek-e, felmerül az árbázolási hiba kérdése, ami miatt tévesen sokkal több szinguláris értéket is pozitívnak vélhet az algoritmus, mint amennyi valójában jelentős méretű. Erre a problémára nyújt megoldást az ábrázolási hibahatár megadása. Még ez a változat sem jár információvesztéssel, de az ábrázolási hibahatár használata miatt más numerikus pontatlanságból adódó zaj fogja terhelni a vékony SVD felbontásból visszaállított $\mathbf{S}$ mátrixot, mint az előző két esetben.

Végül pedig a csonkított változat olyan, hogy csak a legnagyobb r db. szinguláris érték, valamint az ezekhez tartozó $\mathbf{U}$-beli oszlopok és $\mathbf{V}^{\mathrm{H}}$-beli sorok számolódnak ki. Az r szám egy előre megadott konstans. Ekkor $\mathbf{U_c} \in \mathbb{C}^{n\times r}$, $\mathbf{\Sigma_c} \in \mathbb{C}^{r\times r}$, $\mathbf{V_c^{\mathrm{H}}} \in \mathbb{C}^{r\times k}$. Ebben az esetben már az $\mathbf{S}$ mátrixnak csak egy kisebb rangú reprezentációja, $\tilde{\mathbf{S}}$ állítható elő a $\mathbf{U_c} \times \mathbf{\Sigma_c} \times \mathbf{V_c^{\mathrm{H}}}$ szorzatként.

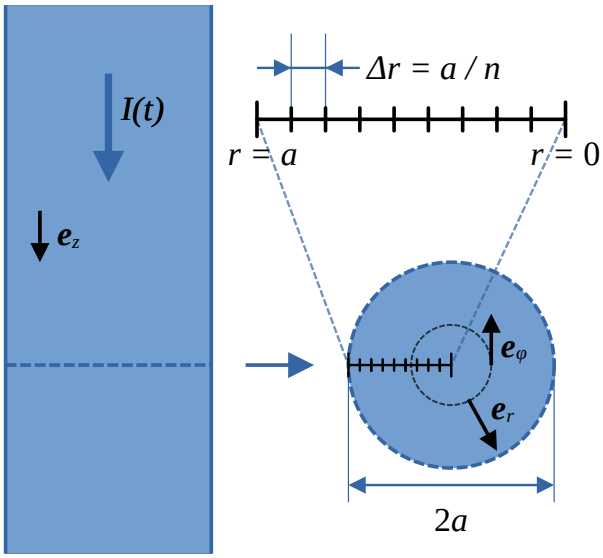
3. Példa szimuláció

A POD működésének demonstrálására a konzulensem, Dr. Bilicz Sándor segítségével írtam egy MATLAB szkriptet. A szkript egy egydimenziós véges differencia közelítést használó mo-

dell tranziens viselkedését szimulálja, pontosabban az elektromos és mágneses tér alakulását. A megoldást többféleképpen számítja ki, teljes és redukált bázisú változatban is, hogy ezek összehasonlíthatóak legyenek.

3.1. A modell

A szimulációval egy kör keresztmetszetű, a sugarú, végtelen hosszú, véges vezetőképességű vezetőben létrejövő mágneses és elektromos teret vizsgáltam. A 2. ábrán látható módon áll össze a modell.



2. ábra. A szimulált modell.

A vezető rúd keresztmetszete a szaggatottal jelölt kör. A vezető hengernek σ fajlagos vezetése és $\mu = \mu_0$ permeabilitása van. A használt henger-koordinátarendszer bázisvektorai $\mathbf{e}_z, \mathbf{e}_\varphi$ és $\mathbf{e}_r$. A vezető henger szimmetriatengelye z-irányú és szintén z-irányú benne a felületi áramsűrűség vektora is, ez az áramsűrűség a vezető hossza mentén nem változik. A modell általában z-irányú eltolásra és a szimmetriatengely körüli (φ irányú) forgatásra invariáns, így csak az egyes mennyiségek r sugártól való függését kell vizsgálni, emiatt egydimenziós a modell.

A vezető keresztmetszetére elő van írva az összesített áram időfüggvénye, $I(t)$, de a skin-hatás miatt a vezető belsejében nem egyenletes az áramsűrűség, hanem a sugár mentén haladva változik. Ugyanígy a vezető belsejében létrejövő H_φ és E_z is függ a sugártól. Ezek a mennyiségek továbbá időfüggőek is, de a vezető henger önindukciója miatt nem pontosan a gerjesztéssel együtt változnak. Ezeknek a mennyiségeknek az időbeli alakulása a kérdés, ezeket határozza meg a szimuláció egy adott gerjesztés mellett. A kísérleteim során az össz-áram időfüggvénye egy fél szinuszhullám, egyébként nulla.

A modell például egy villámhárító földelő vezetőjében létrejövő elektromágneses tér alakulásának a vizsgálatára használható, amikor a villám meghatározza a villámhárítón folyó áramot.

3.2. Teljes rendű megoldás

A félév során készített szimuláció a következőképpen épül fel.

A rendszer 0 kezdeti állapotból indul, tehát kezdetben a hengerben nem folyik áram és emiatt 0 benne a mágneses és elektromos térerősség is. Az eredő $I(t)$ olyan formában van előírva, hogy a vezető felületére adott a mágneses térerősség időfüggvénye, vagyis $H_\varphi(r = a, t)$.

Kétféle időlépéses módon készül megoldás a problémára, az egyik egy egyszerű előrelépő Euler sémát használ, a másik pedig a MATLAB beépített `ode45()` függvényét, ami Runge-Kutta sémára épül és adaptív az időlépése. Mindkét megoldástípushoz szükség van arra, hogy a rendszer adott pillanatbeli állapotából, amit $H_\varphi(r,t)$ teljesen leír, ki lehessen számítani H_φ idő szerinti parciális deriváltját. Ennek a parciális deriváltnak a segítségével aztán lehet előre léptetni az időlépéses sémákat, amiből egy Δt idővel későbbi $H_\varphi(r,t+\Delta t)$ állapot adódik.

$$H_\varphi(r,t+\Delta t) \approx \frac{\partial H_\varphi(r,t)}{\partial t}\cdot\Delta t \tag{5}$$

Az előrelépő Euler séma időléptetése az 5. egyenletben látható. Ez az időlépéses séma könnyen használhatatlan lesz, ha rosszul választjuk meg a térbeli diszkretizáló távolságot, (Δr), valamint az időlépés hosszát (Δt). A szimuláció előtt a szkript konvergenciavizsgálatot végez, amihez a következő képletet használja:

$$F = \frac{\alpha\Delta t}{\Delta r^2}, \quad \alpha = \frac{1}{\mu\sigma} \tag{6}$$

A megoldás csak akkor konvergens, vagyis használható, ha $F < 0.5$.

A Maxwell-egyenleteknek a modellben használt alakja a 7. egyenletrendszerben látható. Magneto-kvázistacionárius közelítést használok, valamint a σ és μ anyagjellemzők egy-egy időben állandó skalárral leírhatóak.

$$\begin{aligned}
\operatorname{rot}\mathbf{H} &= \mathbf{J}\\
\operatorname{rot}\mathbf{E} &= -\frac{\partial\mathbf{B}}{\partial t}\\
\operatorname{div}\mathbf{B} &= 0\\
\operatorname{div}\mathbf{E} &= 0\\
\mathbf{B} &= \mu\mathbf{H}\\
\mathbf{J} &= \sigma\mathbf{E} + \mathbf{J}_i
\end{aligned} \tag{7}$$

A $\mathbf{J}_i$ gerjesztő áramsűrűség csak z irányú, így az Ampère-törvény miatt a rá merőleges $\mathbf{H}$ mágneses térerősségnek csak φ-irányú komponense lesz. A $\mathbf{H}$ változása pedig a Faraday-törvény miatt szintén csak z irányú, $\sigma\mathbf{E}$ vezetési áramot hozhat létre, tehát az eredő $\mathbf{J}$ áramnak mindig csak z-irányú komponense lesz, a $\mathbf{H}$ térerősségnek pedig csak φ-irányú komponense. Így elég csupán ezekkel a komponensekkel foglalkozni, amelyeket mint skalármennyiségekként lehet meghatározni.

$$\begin{aligned}
\mathbf{H} &= H_\varphi\mathbf{e}_\varphi\\
\mathbf{J} &= J_z\mathbf{e}_z\\
\mathbf{E} &= E_z\mathbf{e}_z
\end{aligned} \tag{8}$$

A 7. egyenletrendszer kifejezéseit átrendezve a 9. egyenletben látható összefüggéshez jutunk. Ennek egy véges differenciálokkal kifejezett alakját használva áll elő a szimuláció során az idő szerinti derivált közelítő értéke.

$$\frac{\partial\mathbf{H}}{\partial t} = -\frac{1}{\sigma\mu}\operatorname{rot}\operatorname{rot}\mathbf{H} \tag{9}$$

Mivel a hengerre az összáram időfüggvénye adott, így ezzel együtt a felszíni mágneses térerősség is adott. Ezen kívül a forgási szimmetria miatt az $r = 0$ pontban is ismerten $\mathbf{H}_\varphi = 0$ a gerjesztéstől függetlenül. Ha n db. diszkretizáló ponttal dolgozunk, akkor tehát két pontra már adott a megoldás $\mathbf{H}_\varphi$-re vonatkozóan, így $n-2$ pontra kell azt kiszámítani. A 10. egyenlet szerint véges differencia közelítéssel kiszámítható egy adott időlépésbeli mágneses térerősség

eloszlásából annak az idő szerinti deriváltja, ami felhasználható az időléptetéshez mindkétféle időlépéses sémában:

$$\frac{\Delta \mathbf{H}}{\Delta t} \approx -\frac{1}{\sigma\mu}\mathbf{R}\mathbf{H} \tag{10}$$

ahol $\mathbf{R}$ a rot rot operátor véges differencia közelítésének mátrixa. Így az időléptetéshez nincs szükség az $\mathbf{E}$ vagy $\mathbf{J}$ terek kiszámolására, ezek meghatározása csak egy utófeldolgozási lépést jelent.

3.3. Redukált rendű megoldás

A teljes rendű megoldás időben a gerjesztő impulzus hosszának kétszereséig terjed a kísérleteimben (T a teljes időtartam hossza). Ennyi idő elteltével már a tranziens válasz lényegi része lejátszódik a tapasztalataim szerint. Ennek az időtartamnak az első kis részére, $c \cdot T$ időtartamra keletkező megoldás szolgál a POD bemeneti mintamátrixaként, amiben egy-egy oszlop, vagyis minta, egy adott t-hez tartozó időlépésre érvényes megoldás H_φ-re, amit $\mathbf{H}_\varphi(t)$-vel jelölök. A kísérleteim során $c = 0.07$, 0.15 és 0.3 értékeket próbáltam ki.

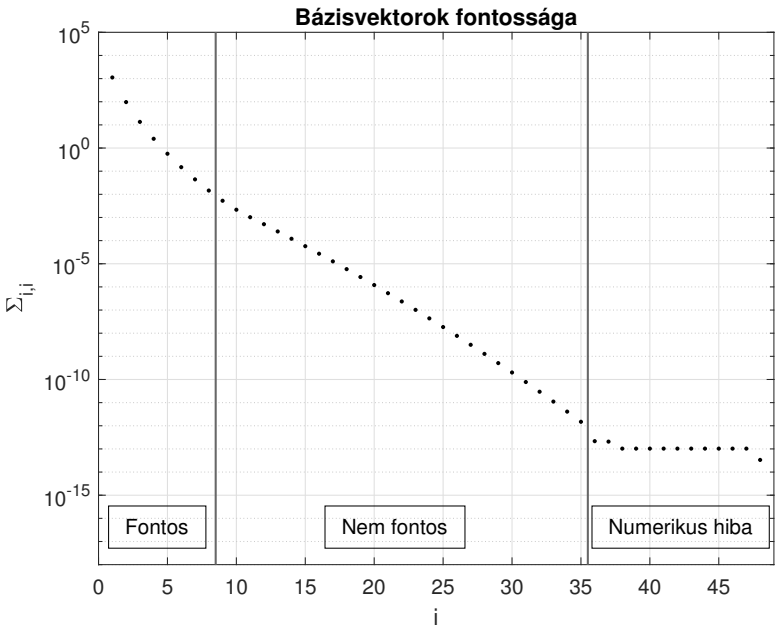
$$\mathbf{S} = \begin{bmatrix} \mathbf{H}_\varphi(0 \cdot \Delta t) & \mathbf{H}_\varphi(1 \cdot \Delta t) & \mathbf{H}_\varphi(2 \cdot \Delta t) & \dots & \mathbf{H}_\varphi(k \cdot \Delta t) \end{bmatrix} \tag{11}$$

Megjegyzem, hogy valószínűleg nem mindig célszerű a mintavételezett időtartam összes időlépését felhasználni $\mathbf{S}$ megkonstruálásához. Ha jól meg tudjuk választani a mintákat, akkor valószínűleg elég a redukált bázis számosságánál csak legfeljebb 1–2 nagyságrenddel nagyobb számú minta a rendszerből, attól függetlenül, hogy hány időlépést tartalmazó időintervallumot mintavételezünk. Ennek a csökkentett mintavételezésnek a hatását a félév során már nem volt időm megvizsgálni.

A Ψ bázist a MATLAB `svd()` függvényével állítom elő. Ez a függvény mint egy melléktermékként az $\mathbf{S}$ mátrix teljes SVD felbontását kiszámítja, így a szinguláris értékek mátrixát, Σ-t is. A szinguláris értékek egy futtatásból a 3. ábrán láthatóak. Az ábrán elég szembetűnő, hogy a szomszédos szinguláris értékek között elég nagy különbség van, főként az első néhány szinguláris értéknél. Ez a redukált rendű szimuláció szempontjából egy nagyon jó hír, hiszen ekkor ha csak az első néhány bázisvektort használjuk is, a megoldás jól fogja közelíteni a teljes rendűt. A 3. ábrán bejelölt határvonalak közül a „numerikus hibá"-t elválasztó vonal elég egyértelmű helyen van, ott van egy törés az értékek sorozatában. Ezzel szemben a „fontos" és „nem fontos" részek közötti vonal pontos helye már nem magától értetődő, attól függ, hogy mekkora hibát engedhetünk meg a szimuláció során.

A numerikus hibák által dominált szinguláris értékek egymástól csak nagyon kevéssé térnek el, a többség majdnem pontosan 1×10^{-13} értékű. A legnagyobb szinguláris érték pedig 1×10^3 körüli. Ezek hányadosa tehát körülbelül 1×10^{16}. A számok MATLAB-ban a sztenderd double formátumban tárolódnak el, amiknek az értékes helyiértékeket tároló része, vagyis a mantissza effektíve 53 bites. Az 53 biten ábrázolható legnagyobb szám $(2^{53} - 1)$ körülbelül 9.0072×10^{15}, ami jól illeszkedik a tapasztalt relatív pontossághoz. Gyakorlatilag amikor két double-ként tárolt számot adna össze a számítógép, amelyek hányadosa eléri a 2^{53}-t, akkor az exponensek egyeztetéséhez a kisebb szám mantisszáját legalább 53 helyiértékkel kell léptetelni, de ezáltal az összes értékes bit kiléptetődik és 0-kkal töltődik fel a helyük. Emiatt az összeadás eredménye pontosan a nagyobb szám lesz, mintha hozzá sem adtuk volna a kisebbet, így a kisebb szám effektíve 0. Az SVD kiszámolása közben ilyen double összeadások miatt lehetetlenül el a legnagyobbhoz képest nagyon kis szinguláris értékek és az ezekhez tartozó szinguláris vektorok kiszámolása.

Az SVD kimenetei között van az $\mathbf{U}$ mátrix is, ami az új bázisvektorokat ($\mathbf{u}_i$) tartalmazza fontosság szerint csökkenő sorrendben az oszlopaiban. Ezek közül a legfontosabb r db-ot meg-

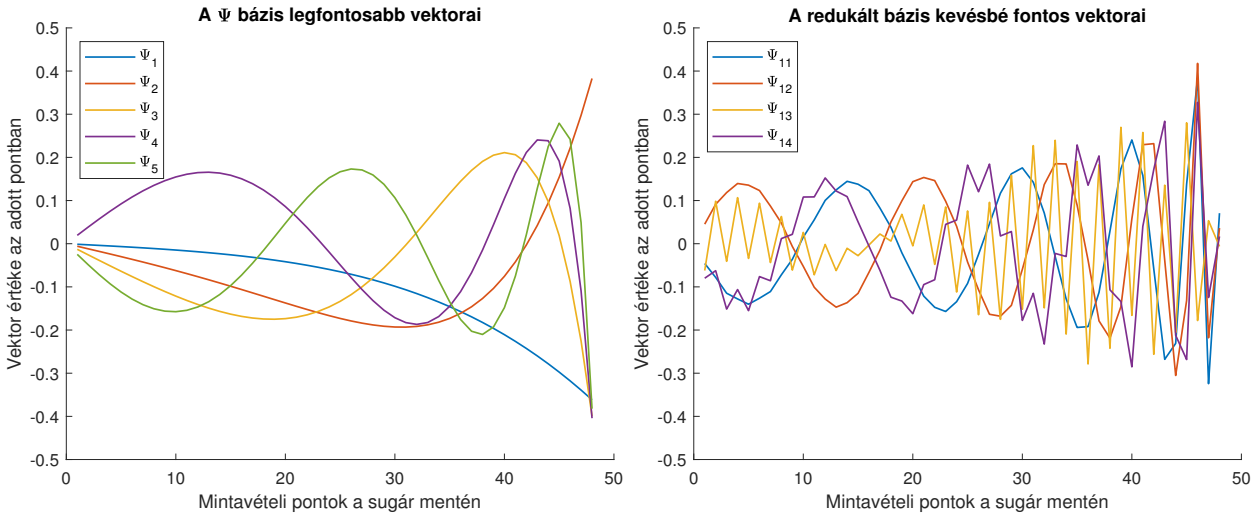


3. ábra. A szinguláris értékek az előrelépő Euler megoldásból.

tartva és azonos sorrendben egy új mátrixba rendezve kapjuk a redukált bázis $\hat{\mathbf{U}}$ mátrixát, ami az új, redukált bázisból a standard bázisba való áttérés mátrixa.

$$\hat{\mathbf{U}} = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_3 & \dots & \mathbf{u}_r \end{bmatrix} \tag{12}$$

Érdemes szemügyre venni magukat a bázisvektorokat is, amelyek közül néhány a 4. ábrán látható. Fontos megjegyezni, hogy az egyes bázisvektorok valódi térerősség-eloszlásokat jelentenek, amelyeknek valamilyen lineáris kombinációjával írjuk le a rendszer aktuális állapotát.



(a) Az öt leglényegesebb vektor.

(b) Négy lényegtelen vektor.

4. ábra. A Ψ bázis néhány vektora.

Mivel $\mathbf{U} \in \mathbb{C}^{n \times n}$ unitér, ezért:

$$\mathbf{U}\mathbf{U}^{\mathrm{H}} = \mathbf{U}^{\mathrm{H}}\mathbf{U} = \mathbf{I}_{n \times n} \tag{13}$$

De a csonkítás miatt az $\hat{\mathbf{U}} \in \mathbb{C}^{n\times r}$ mátrixra már csak a következő igaz:

$$\hat{\mathbf{U}}^{\mathrm{H}}\hat{\mathbf{U}} = \mathbf{I}_{r\times r} \tag{14}$$

Az $\hat{\mathbf{U}}$ mátrixnak egy további fontos tulajdonsága, hogy ennek a mátrixnak a segítségével lehet egy standard bázisban felírt $\mathbf{x} \in \mathbb{C}^n$ vektort a redukált Ψ' bázisba vetíteni, valamint a fordított irányú bázistranszformációt is el lehet végezni $\hat{\mathbf{U}}^{\mathrm{H}}$ segítségével:

$$\begin{aligned} \hat{\mathbf{U}}^{\mathrm{H}}\mathbf{x} &= \mathbf{y} \\ \hat{\mathbf{U}}\mathbf{y} &= \mathbf{x}' \approx \mathbf{x} \end{aligned} \tag{15}$$

ahol $\mathbf{y} \in \mathbb{C}^r$ a Ψ' bázisbeli reprezentációja $\mathbf{x}$-nek. Az iylen irányú éttérés információvesztéssel jár, de a POD tulajdonságai miatt az euklideszi távolság $\mathbf{x}$ és $\mathbf{x}'$ között kicsi olyan $\mathbf{x}$-ekre, amelyek előfordulhatnak a rendszer állapotai között.

Az időléptetéshez még szükséges az $\mathbf{R}$ mátrixnak a vetítése a Ψ' bázisba, vagyis $\mathbf{R}_\Psi$ kiszámolása. Ezt a jól ismert bázistranszformációs képlet alapján lehet megtenni:

$$\hat{\mathbf{U}}^{\mathrm{H}}\mathbf{R}\hat{\mathbf{U}} = \mathbf{R}_\Psi \tag{16}$$

Az eddigiek felhasználásával pedig kiszámíthatjuk az $\mathbf{R}\mathbf{x}'$ egy közelítését a Ψ' bázisban anélkül, hogy az egyes időlépések közben bármikor is vissza kellene térni a standard bázisba:

$$\mathbf{R}_\Psi\mathbf{y} = \hat{\mathbf{U}}^{\mathrm{H}}\mathbf{R}\hat{\mathbf{U}}\mathbf{y} = \hat{\mathbf{U}}^{\mathrm{H}}\mathbf{R}\mathbf{x}' \tag{17}$$

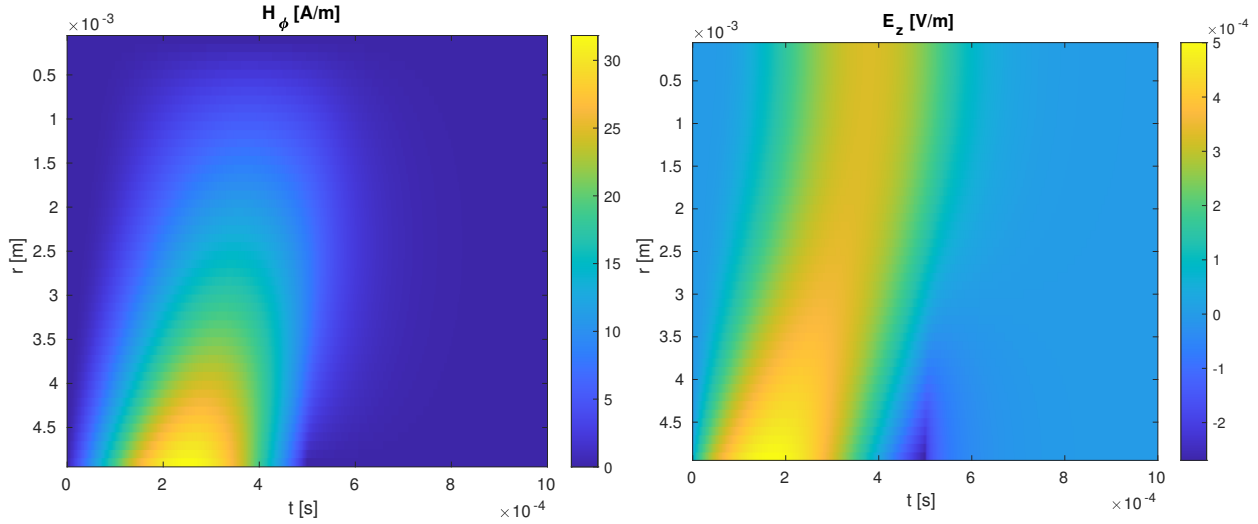
A redukált bázisban való számolás összefoglalva a következőképpen zajlik. A mintavételezett időintervallum alapján megkonstruálom a Ψ' bázist. Ebbe a bázisba vetítem a mintavételezés végén a rendszer aktuális állapotát, valamint a rot rot operátor mátrixát. Ezek után már csak a redukált bázisban számolok tovább. A szimuláció végén a megoldást visszavetítem a standard bázisba, de ezt a vetítést nem feltétlenül kell megtenni minden időlépés minden térbeli pontjára, mind térben, mind időben lehet szelektálni.

4. Eredmények

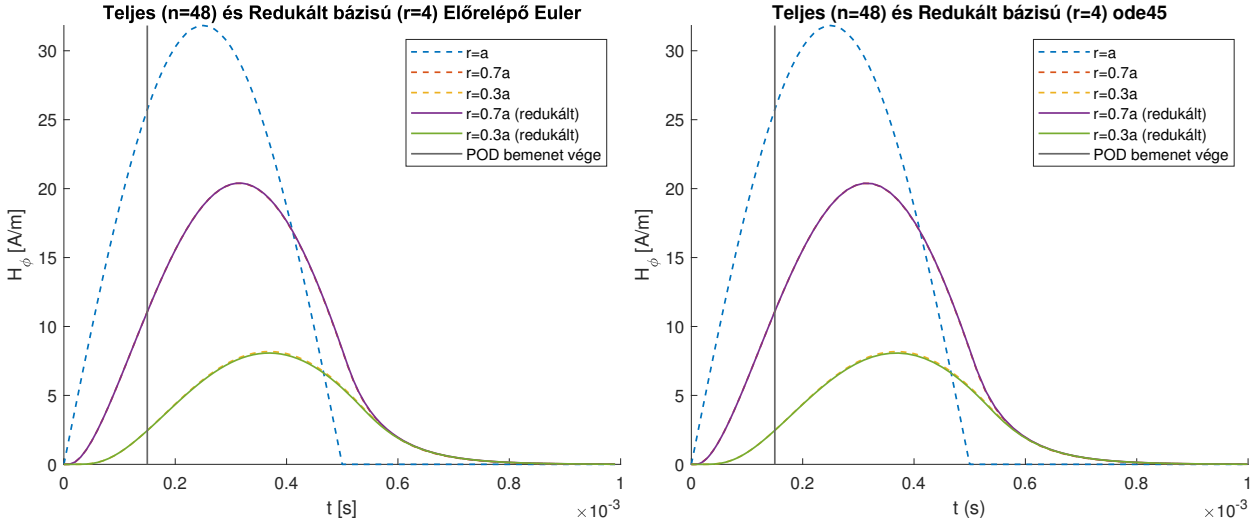
Olyan alkalmazások esetén, amikor a tranziens szimulációnál lényeges erőforráscsökkenéssel jár a POD alkalmazása, nem érdemes a teljes rendű szimulációt végig futtatni, csak a POD bemenetének használt időtartam végéig, legfeljebb kicsivel tovább az esetleges nyilvánvaló hibák felismerése céljából. A vizsgálódásomhoz a teljes rendű szimulációt is végig futtattam, hogy össze lehessen hasonlítani a redukált megoldással. A teljes rendű Euler-séma megoldásának vízesésdiagramjai az 5. ábrán láthatóak.

Külön megoldást készítettem azonos paraméterekkel, mind az egyszerű előrelépő Euler, mind a Runge-Kutta sémák használatával. Ezeknek az eredményei a 6. ábrán láthatóak. Az ábrák alapján mindkét séma hasonlóan jó megoldást ad, így többnyire bármelyik használható. A továbbiakban elsősorban az Euler-sémán mutatom be a felmerülő jelenségeket.

Amint az ábrán is látszik, azonos paraméterek mellett mindkét időlépéses séma esetén hasonlóan kis hibával működik a redukált rendű közelítés. Jobb kvantitatív képet kapunk a redukált bázisú szimuláció pontosságáról, ha a teljes rendű szimuláció összetartozó pontjához viszonyított relatív hibát, valamint az abszolút hibát vizsgáljuk. Ezek az eredmények a 7. ábrán láthatóak. A redukált rendű szimulációnak több paraméterét lehet szabadon megadni, ezek között valamilyen kompromisszumot kell találni. A kompromisszummal a cél, hogy még elfogadható hiba mellett, minél kisebb legyen az erőforrásigénye a redukált rendű közelítésnek.



5. ábra. Teljes rendű megoldások az előrelépő Euler sémával.



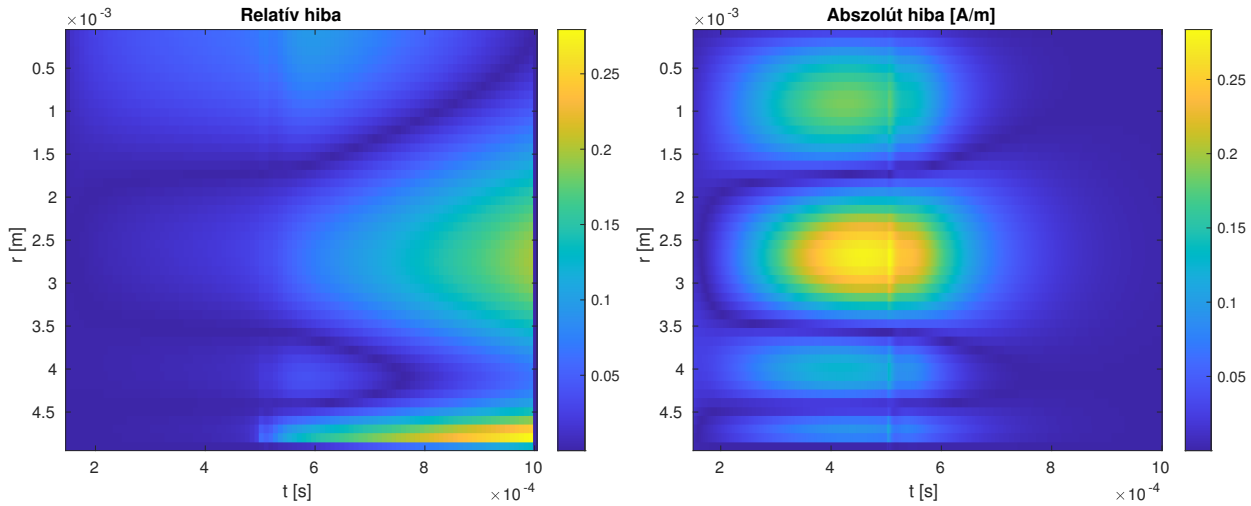
(a) Előrelépő Euler

(b) Runge-Kutta

6. ábra. Megoldások kétféle sémával, teljes és redukált bázisokban ($r = 4$ és $c = 0.15$).

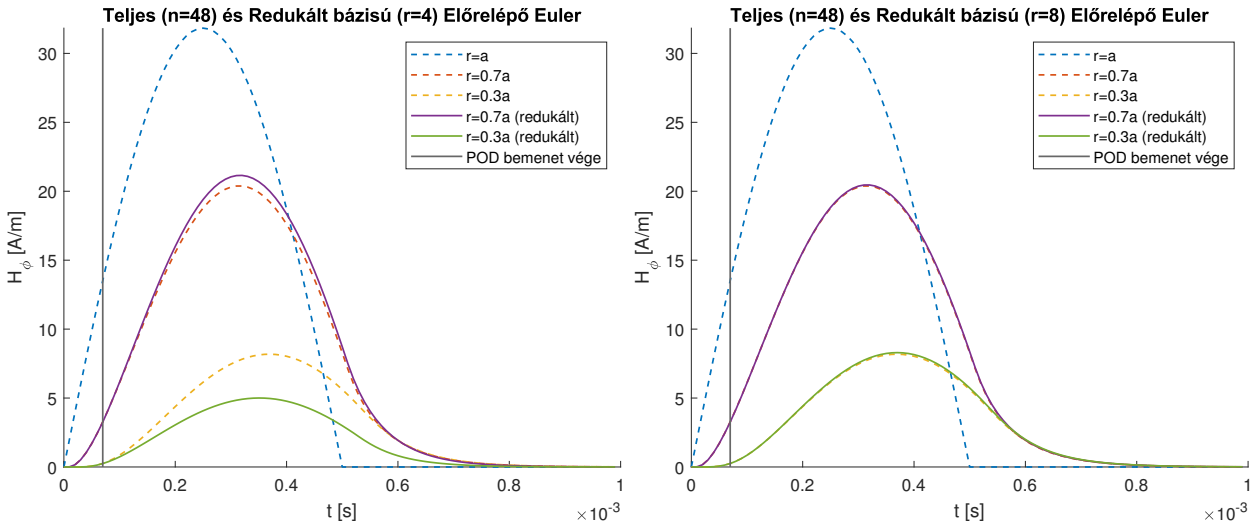
A megválasztható paraméterek a redukált bázis rendje, r, valamint a szimuláció teljes rendű részének hossza az teljes szimulált időtartamhoz képest, c. Tapasztalataim szerint a rögzített $c = 0.15$ választás mellett $r = 4$ az alsó határ, aminél még ránézésre használható közelítő eredmény adódik. Ha ugyanilyen $r = 4$ mellett $c = 0.07$-ot választottam, jelentősen megnőtt a közelítő megoldás hibája. Ez a megnövekedett hibájú megoldás a 8a. ábrán látható. Ebben a helyzetben az egyik megoldás, hogy a lecsökkentett c mellé megnöveljük r-t, hogy a kevés és kevésbé reprezentatív mintahalmaz alapján is olyan bázist tudjon produkálni a POD, amiben jól leírható a rendszer. Enek a megoldásnak a hatása, pontosabban az $r = 8$ választás hatása, a 8b. ábrán látható.

Nagy hiba esetén a másik irányú korrekcióval is próbálkozhatunk, név szerint c növelésével. Ennek a bemutatásához az eredeti $c = 0.15$, $r = 4$ paraméterektől az $r = 2$ választással tértem el, ami már egy extrém kicsi szám erre a paraméterre, így nem várhatunk sok jót ettől a közelítő megoldástól. Az ehhez tartozó eredmények a 9a. ábrán láthatóak. Ezek után a $c = 0.3$ választással éltem, ennek az eredménye a 9b. ábrán látható. Megfigyelhető, hogy a teljes rendű megoldáshoz viszonyított hibával jelentős javulást nem értem el. A $c = 0.3$ már azt jelenti, hogy



(a) Hiba az összetartozó ponthoz képest. (b) Abszolút hiba.

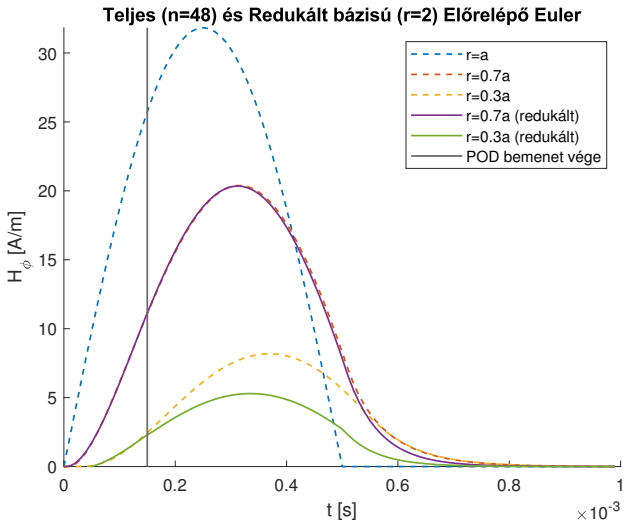
7. ábra. Az Euler séma abszolút és relatív hibája ($r = 4$ és $c = 0.15$).



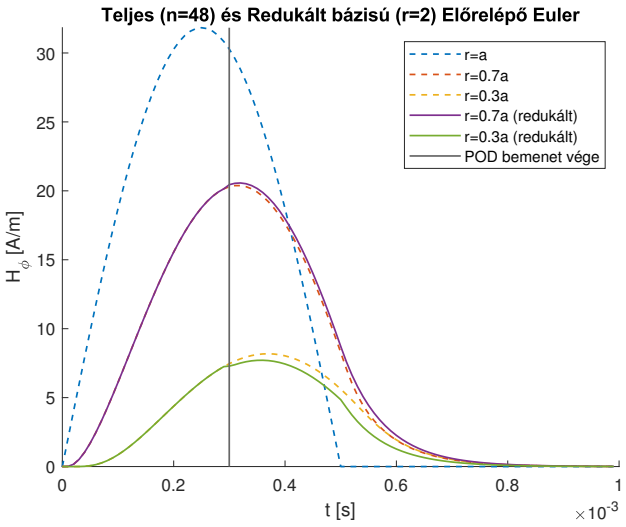
(a) Probléma (b) Megoldás ($r = 4 \rightarrow 8$)

8. ábra. Túl rövid a teljes rendű szimuláció időtartama (c).

a teljes időtartamnak egy számottevő részét a teljes rendű szimulációval számolom ki, ezek után már megkérdőjelezhető a POD használata a fennmaradó irőre. A redukált bázis kiszámolásának, valamint az ebben kiszámolt megoldás visszaalakításának többletmunkájától eltekintve még szélsőséges esetben is csak körülbelül 0.3-szorosára csökkenthető az erőforrásigény az egész időtartamra végzett teljes rendű számításhoz képest. Ezek alapján ki lehet jelenteni, hogy a vizsgált rendszer elfogadható modellezésére nem elegendő 2 db. bázisvektor. A bázisvektorok számára vonatkozó határ mellett feltétlenül létezik a minták számára, valamint azok reprezentativitására vonatkozó határ is. Ez pedig azt korlátozza, hogy a teljes időtartamnak mennyire kis részére lehet korlátozni a mintavételezést. Természetesen ez a két határ különböző modelleknél más és más.



(a) Probléma



(b) Részleges megoldás ($c = 0.15 \rightarrow 0.3$)

9. ábra. Túl kicsi a bázisvektorok száma (r).

5. Összegzés

A félévben megismerkedtem a Proper Orthogonal Decomposition módszerrel és sikeresen alkalmaztam egy MATLAB-ban megírt FDTD alapú tranziens szimuláción. Megvizsgáltam a módszer néhány alapvető korlátját és jellegzetességét.

A későbbiekben a POD potenciális alkalmazási területeit tervezem részletesebben felkutatni és az ígéretesebb alkalmazásoknál megvizsgálni, hogy van-e haszna a POD-nak az adott problémában. Ezek után pedig egy kiválasztott bonyolultabb problémára tervezem alkalmazni a módszert és elemezni a hatásait. Egy további kérdés lehet a redukált bázis használatából adódó hibák nagyságának és jellegének becslése a redukált rendű szimuláció futtatása előtt. Egy másik fennmaradó kérdés pedig az, hogy a mintamátrixba kerülő mintavektorokat hogyan érdemes kiválasztani a rendelkezésre álló adatok közül, és érdemes-e egyáltalán megszűrni az összes rendelkezésre álló adatot.

Köszönetnyilvánítás

Köszönettel tartozom a konzulensemnek, Dr. Bilicz Sándornak a segítségért, ötletekért és iránymutatásért, amivel a munkám során támogatott.

Hivatkozások

[1] Wikipedia: Singular value decomposition. Elsősorban a képek miatt hivatkozom ezt a Wikipedia oldalt. URL: https://en.wikipedia.org/wiki/Singular_value_decomposition.

[2] Karthik Venkatraman Aadithya, Eric R. Keiter, and Ting Mei. The Karhunen Loéve Expansion. 6 2018. DOI: 10.2172/1761975.

[3] Anindya Chatterjee. An introduction to the proper orthogonal decomposition. 2000.

[4] Davide Chicco and Marco Masseroli. Software suite for gene and protein annotation prediction and similarity search. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 12(4):837–843, 2015. DOI: 10.1109/TCBB.2014.2382127.

[5] Golub G. H. and Reinsch C. Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14(5):403–420, 1970. DOI: 10.1007/BF02163027.

[6] Thomas Henneron and Stéphane Clénet. Model order reduction of non-linear magnetostatic problems based on pod and dei methods. *IEEE Transactions on Magnetics*, 50(2):33–36, 2014. DOI: 10.1109/TMAG.2013.2283141.

[7] Philip Holmes, John L. Lumley, Gahl Berkooz, and Clarence W. Rowley. *Galerkin projection*, page 106–129. Cambridge Monographs on Mechanics. Cambridge University Press, 2 edition, 2012. DOI: 10.1017/CBO9780511919701.006.

[8] Ian T. Jolliffe and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016. DOI: 10.1098/rsta.2015.0202.

[9] R. Narasimha. Kosambi and proper orthogonal decomposition. *Resonance*, 16:574–581, 2011. DOI: 10.1007/s12045-011-0062-8.

[10] Lawrence Sirovich. Turbulence and the dynamics of coherent structures. I-III. *Quarterly of Applied Mathematics*, 45:561–590, 1987.