— XXVII. Országos Tudományos Diákköri Konferencia —

SZALAY ZSÓFIA

Genetikus algoritmus alkalmazása NMR spektrumok paramétereinek becslésére

Témavezető: Dr. Rohonczy János egyetemi docens

Készült az Eötvös Loránd Tudományegyetem
Általános és Szervetlen Kémia Tanszékén

— Budapest, 2005. —

Bevezetés

Az NMR spektroszkópia napjaink egyik legelterjedtebb szerkezetvizsgálati módszere, amely segítségével az anyagok szerkezete mind oldat-, mind szilárd fázisban meghatározható. Ez utóbbi esetben a különböző orientációjú mikrokristályok a külső sztatikus mágneses térrel bezárt szögük függvényében más-más frekvencián adnak jelet, melynek eredménye széles porspektrum lesz. A nagyon erős jelkiszélesedés miatt a spektrum felvétele is hosszabb időt vesz igénybe, és a spektrumok értelmezése is nehezebbé válik: újabb, tenzoriális paraméterek jelennek meg. A porminták gyors forgatása megfelelő tengely körül (MAS, *Magic Angle Spinning*) az irányfüggő kölcsönhatások egy részét átlagolja és ezáltal a jeleket refókuszálja. Ilyen kísérleti körülmények mellett kapott spektrumokból is rendkívül összetett és számolásigényes a tenzoriális mennyiségek meghatározása. Ehhez egyrészt szükséges egy kvantummechanikai modellezésen alapuló szimulációs program, amely adott paraméterekből kiszámolja a spektrumot, másrészt egy olyan algoritmus, amely a szimuláció paramétereit optimálja. Egyszerűbb esetekben meg tudjuk becsülni a várt értékeket, így ebből a pontból indítva a keresést, a paraméterek finomíthatók. Ilyenkor determinisztikus szélsőérték kereső eljárások használhatók a paraméterek optimálására. Ezen módszerek azonban kifejezetten lokális szélsőértéket keresnek, tehát sokat nem tudnak változtatni a megadott paramétereken (mindig csak jobb értékek felé lépnek el, ezért a paramétertér bizonyos részeit eleve kizárja). Mivel nincs olyan determinisztikus algoritmus, ami képes globális szélsőérték-keresésre, ezért célszerű véletlen választáson alapuló módszert alkalmazni, ami nemcsak pontosítani képes az általunk megadott értékeket, hanem azokat megkeresni is. Ilyen számítógépes módszer a *genetikus algoritmus*, amely a *darwini* evolúció és a *mendeli* genetika elvén működő optimalizálási eljárás.

Jelen dolgozat témája egy ilyen genetikus algoritmust felhasználó program, amit Java nyelven készítettem és a Bruker Biospin GmbH. „TopSpin” nevű programcsomag szilárd NMR spektrumokat szimuláló moduljába integráltam. A dolgozat bemutatja a program működési elvét, valamint néhány példán keresztül azt, hogy hogyan alkalmazható ez az algoritmus szilárd NMR spektrumok paramétereinek meghatározására olyan esetekben, amikor más módszerek nem, vagy csak bizonytalanul adnak megoldást.

A szilárd NMR spektrumok jellemző paraméterei

Az 1D NMR spektroszkópia kvantummechanikai leírását a 70-es, 80-as évekre kidolgozták, azonban a tenzoriális mennyiségek helyett, többnyire a kiátlagolódó kölcsönhatásoknak megfelelően skalár mennyiségekkel számoltak. Ez oldatmintáknál a molekulák szabad forgása miatt tökéletesen megfelel, azonban szilárd fázisban kénytelenek vagyunk a bonyolultabb matematikai apparátust igénylő tenzorokkal számolni.

Általában a Hamilton-operátor, amely leírja a magok kölcsönhatását egymással és külső mágneses térrel, felbontható a különböző kölcsönhatások operátorainak összegére: [1, 2]

$$H = H_Z + H_D + H_{CS} + H_{SC} + H_Q,$$

amelyek sorrendben a Zeeman-kölcsönhatás, a dipoláris csatolás, a kémiai árnyékolási, a spin-spin csatolási és a kvadrupol kölcsönhatás operátorai. Ezek a tenzorok általában irányfüggők, folyadékfázisban azonban a molekulák szabad mozgása miatt kiátlagolódnak. Szilárd fázisban viszont tenzorként kell kezelnünk őket.

A kölcsönhatások közül a Zeeman-effektus több nagyságrenddel nagyobb, így a többitől függetlenül kezelhető:

$$H_Z = -\gamma\hbar B_0 I_z.$$

A kémiai árnyékolási kölcsönhatás a legfontosabb a kémiai szerkezetvizsgálat szempontjából:

$$H_{CS} = \gamma_I \hbar \cdot I \cdot \sigma \cdot B_0,$$

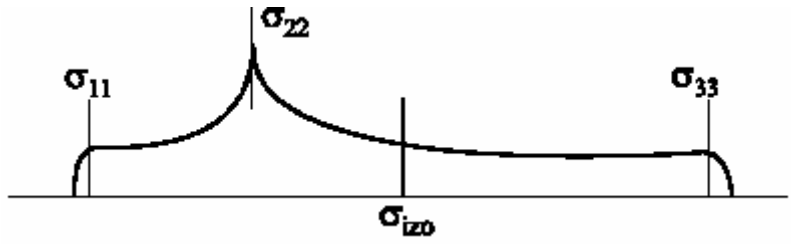
ahol a σ tenzor a kémiai árnyékolási tenzor. Ez a tag természetesen folyadékfázisban sem esik ki, ott a tenzor diagonális komponenseinek átlagával helyettesíthető a molekulák szabad forgása miatt. Szilárd fázisban a magok nem foroghatnak szabadon, ezért σ-t továbbra is tenzorként kell kezelni [3]. A Descartes-féle koordinátarendszerben felírt mátrix diagonalizálható, így kapjuk meg a krisztallit főtengelyeinek irányait, illetve az ezekre jellemző árnyékolási tényezőt:

$$\sigma = \begin{pmatrix} \sigma_{11} & 0 & 0 \\ 0 & \sigma_{22} & 0 \\ 0 & 0 & \sigma_{33} \end{pmatrix}.$$

Egykristályoknál a periodikusan ismétlődő magok főtengelyei párhuzamosak egymással, ezért ugyanolyan eltolódásértéknél fognak jelet adni. A tenzor főtengelyei és a külső mágneses tér által bezárt szög függvényében ez az érték más és más. Amennyiben valamelyik főtengely párhuzamos a B_0 térrel, akkor a neki megfelelő diagonális komponens árnyékolását fogjuk mérni. Az árnyékolási tenzor iránya és komponenseinek nagysága a mag körüli

elektronsűrűség függvénye. Ezen adatok kvantumkémiai számításokkal is megkaphatók. A számított és kísérletileg meghatározott tenzorkomponensek összehasonlítása közvetlen mércéje lehet a kvantumkémiai számítások helyességének.

Az árnyékolási tenzor főkomponenseinek meghatározásához azonban nem kell feltétlenül egykristályt mérni. Ha a minta por, a különböző krisztallitok orientációja eltér egymástól, ezért más és más lesz a kémiai eltolódásuk. Ezek eredőjeként jellegzetes alakú, széles jelet kapunk, amelyből az árnyékolási tenzor diagonálisának elemei megállapíthatók (1. ábra).



[1]

1. ábra Az árnyékolási tenzor főkomponensei egy jellegzetes porspektrumban

Ezek az értékek esetenként nehezen meghatározhatók, ezért bevezettek különböző paramétereket, amelyek egyrészt egyértelmű kapcsolatban vannak a σ_{11}, σ_{22}, σ_{33} értékekkel, másrészt a spektrumokból könnyebben megállapíthatók. Többféle ilyen rendszert definiáltak [4, 5, 6], ezek közül a program Haeberlen-féle [4] paramétereket használja, amelynek elemei:

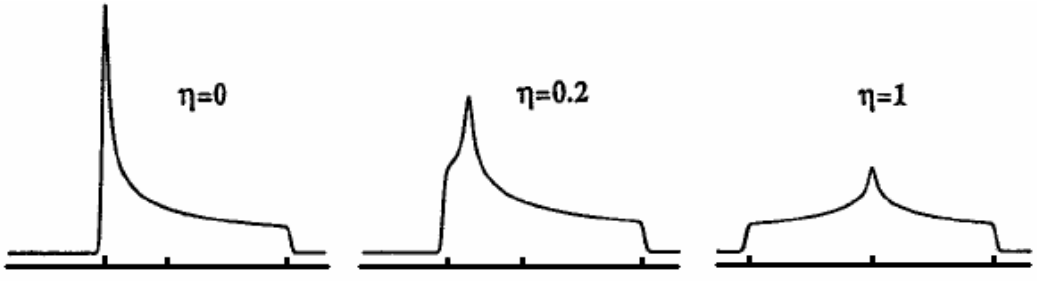
- izotróp kémiai eltolódás: $\sigma_{iso} = \frac{\sigma_{11} + \sigma_{22} + \sigma_{33}}{3}$.

  Ez az érték nyilvánvalóan megegyezik a folyadékfázisban kapott eltolódással (ha az oldószer nem befolyásolja a magspin által érzett elektromágneses teret).

- kémiai árnyékolási anizotrópia: $\delta = \sigma_{33} - \frac{\sigma_{11} + \sigma_{22}}{2}$.

  Ez a paraméter a porspektrum szélességét jellemzi. Előjele pozitív és negatív is lehet, attól függően, hogy σ_{22} kisebb vagy nagyobb, mint az izotrópátlag.

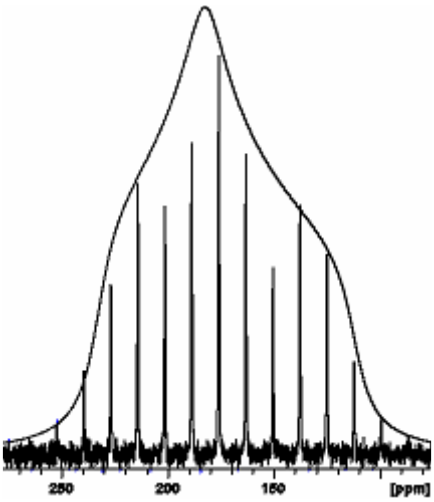
- árnyékolás aszimmetriáját leíró tényező: $\eta = \frac{\sigma_{11} - \sigma_{22}}{\sigma_{33} - \sigma_{iso}}$.



[3]

2. ábra Jelalakok különböző aszimmetria esetén

A porspektrumok hátránya, hogy a jel/zaj viszony jelentősen romlik mind az egykristály, mind az oldatban felvett jelekhez képest (ugyanaz a jel alatti terület jóval szélesebb tartományon oszlik szét.), ami a spektrum felvétel idejét megnyújtja, és az elemzést is nehezíti. A spektrum minősége jelentősen javítható a minta gyors forgatásával a mágneses tér irányával nem párhuzamos tengely körül. Ekkor a jelek kiszélesedéséért felelős anizotróp kölcsönhatások $(3\cos^2\varphi - 1)/2$ tényezővel szorzódnak (φ a forgási tengely és a mágneses tér által bezárt szög) és ezáltal a jelek keskenyebbé válnak. Abban a speciális esetben, amikor $\cos^2\varphi = 1/3$, azaz $\cos\varphi = 54{,}7°$ („magic angle") az anizotróp hatások kiátlagolódnak és a folyadékfázisban tapasztalthoz hasonló jeleket kapunk. Ha a forgatás frekvenciája kisebb, mint a jel anizotrópiája, a forgatás frekvenciájának egész számú többszöröseinél forgási oldalsávok jelennek meg. Ezek a jelek a porspektrum burkolója alá esnek (3. ábra), így ezekből is ugyanazok a paraméterek meghatározhatók, mint az álló mintából [1].



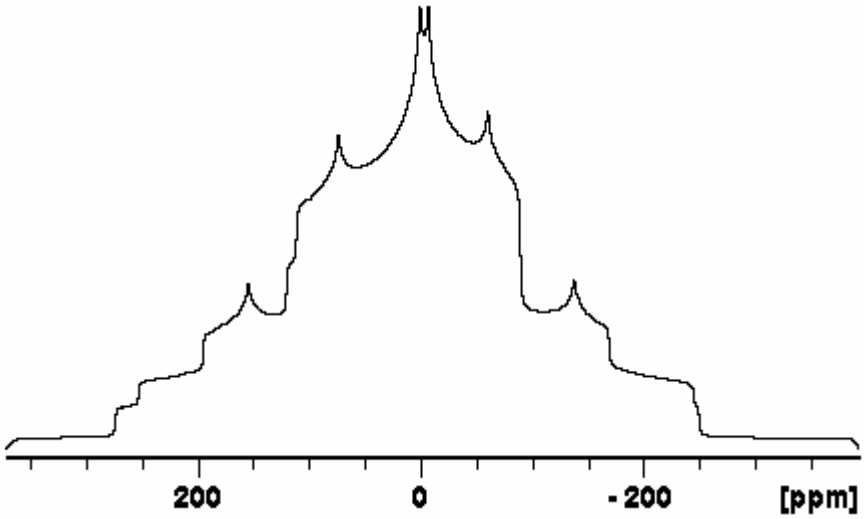
3. ábra Porspektrum (szimulált) és MAS spektrum (kísérleti) 950 Hz. A forgási oldalsávok a porspektrum alá esnek.

A spin-spin kölcsönhatás nagyságrendekkel gyengébb a többinél, így szilárd fázisban a széles jelek miatt ritkán észlelhető. Folyadékfázisú mintáknál azonban nem nulla átlagot ad, ezért oldatban a kémiai eltolódás mellett ez az egyik legfontosabb paraméter.

A dipoláris kölcsönhatás a Zeeman-effektusnál nagyságrendekkel kisebb, azonban a másik háromnál továbbra is nagyobb lehet, habár értéke a távolság növekedésével rohamosan csökken:

$$H_D = \gamma_I \gamma_S \frac{\hbar^2}{r_{IS}^3} \cdot I \cdot D \cdot S \,,$$

ahol I és S az egymással csatoló spineket jelölik, γ_I és γ_S a magok giromágneses tényezője, r_{IS} a két mag távolsága. A D dipoláris csatolási tenzor szimmetrikus és spurja 0, ezért folyadék fázisban (illetve forgatott mintára) nem jelenik meg a spektrumban. Álló mintára is csak a közeli spinek között hat, de így is elég összetetté tudja tenni a spektrumot, mint például a 4. ábrán látható esetben, ahol egy ^{31}P maggal csatolt dipolárisan egy 5/2 spinű ^{17}O, ennek eredménye a ^{31}P spektrum jelentős kiszélesedése.



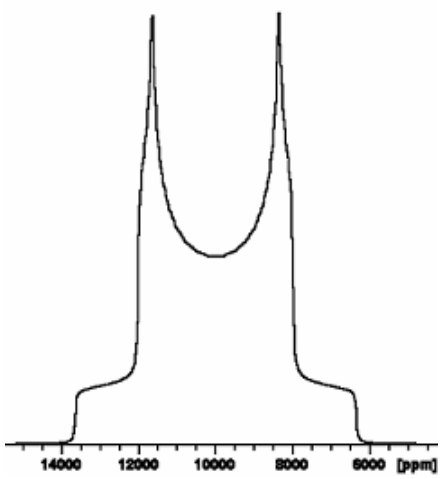
4. ábra Egy 5/2 spinű ^{17}O maggal dipolárisan csatolt P mag szimulált ^{31}P NMR spektruma

A kvadrupol tag I > 1/2 spinű részecskéknél megjelenő kölcsönhatás:

$$H_Q = \frac{eQ}{2I \cdot (2I-1) \cdot \hbar} \cdot I \cdot V \cdot I \ ,$$

ahol a V tenzor az elektromos tér gradiense, Q a mag kvadrupol momentuma. Mivel $SpurV = 0$, folyadékfázisú NMR-ben ezzel sem kell foglalkozni. Szilárd fázisú mintáknál V a többi tenzorhoz hasonlóan diagonalizálható, és a sajátértékekből a CSA paraméterekhez hasonló új, szemléletes jelentéssel bíró, változókat lehet definiálni. Feltételezve, hogy a tenzor diagonálisának elemei V_{11}, V_{22}, és V_{33}, a paraméterek:

- kvadrupol csatolási állandó: $C_Q = \frac{eQ \cdot V_{33}}{\hbar}$
- aszimmetria paraméter: $\eta_Q = \frac{V_{11} - V_{22}}{V_{33}}$



5. ábra I = 1 mag spektruma $\eta_Q = 0{,}1$, $C_Q = 1500$ kHz

Az 5. ábrán egy tipikus 1-es spinű (tehát kvadrupol) mag szilárd fázisú spektruma látható, 0,1-es aszimmetriával és 1500 kHz-es csatolási állandóval.

A kvadrupol kölcsönhatás alapvetően különbözik az összes többi anizotróp kölcsönhatástól abban, hogy felbontható első- és másodrendű tagok összegére. Ennek az a jelentősége, hogy míg az elsőrendű tagok forgatással kiátlagolhatók, a másodrendűek nem, és ezért a kvadrupol csatolások a MAS spektrumokban is megjelennek, és nagyon széles jeleket okoznak. [7]

Genetikus algoritmusok alapelvei

Az evolúció elméletét Darwin dolgozta ki az 1800-as évek közepén [8]. Ennek egyik alapelve, hogy a fajok és egyedek folytonos harcban állnak az életben maradásért: a gyengébbek elbuknak, az erősebbek tovább élnek, idővel szaporodnak. Ilyen módon a legéletrevalóbb egyedeknek lesznek utódaik, akik szüleik tulajdonságait részben öröklik. Pár évvel később a *mendeli* genetika az öröklődés további mélységeit is feltárta: a tulajdonságokat génekhez kötötte. Ma már tudjuk, hogy a kromoszómák határozzák meg az egyedek tulajdonságait, és ezek osztódási mechanizmusa is ismert.

A genetikus algoritmusok – mint a nevük is mutatja – ezt a biológiai modellt utánozzák. A korai genetikus algoritmusok célja nem szélsőértékek keresése volt, hanem csak azon kísérleteztek, hogy a természetes evolúciót utánozzák számítógépes rendszerekben [9]. Az áttörést Holland munkája hozta [10], akinek elméletében már kitisztultan jelenik meg az adaptáció fogalma. Az egyedeket – a biológiához hasonlóan – kromoszómák kódolják (eltérés azonban, hogy minden egyednek csak egyetlen kromoszómája van), amelyek gyakorlatilag gének (többnyire bitek) sorozatát jelentik. A populációkból valamilyen szelekciós elv alapján kiválaszt néhányat, amelyeket szülőknek nevez. Ezek utódokat hozhatnak létre, és utódaik az ő helyükbe lépnek. A kiválasztási szabályt úgy kell megalkotni, hogy mindig a legéletrevalóbb, legjobb egyedek szaporodhassanak nagyobb valószínűséggel, így az ő tulajdonságaik (génjeik, paramétereik) nagyobb eséllyel élnek tovább, és idővel a kizárólag rossz egyedekben előforduló gének eltűnnek a populációból (természetesen a kevésbé alkalmas egyedek is szaporodhatnak, csak kisebb valószínűséggel). A szaporodás mechanizmusa hasonló, mint a természetben: a kromoszómák kereszteződnek és az új kromoszómában a két szülő génjei keverednek. Mivel azonban egyszálú kromoszómák vannak, nem a két szál fűződik össze, hanem a szálon belül az egyes gének. Ez kicsit nagyobb variációs lehetőséget jelent, mint a biológiai crossover. További lehetőséget biztosít a változatosságra a természetben „evolúciós hibaként” emlegetett mutáció. Ez részben módosíthatja a kialakult kromoszómákat, véletlenszerűen megváltoztatva néhány gént.

Az ilyen elvek alapján felépített rendszerekkel modellezni lehet evolúciós, vagy ehhez hasonló, véletlen választáson alapuló rendszereket. A másik, széles körben elterjedt alkalmazási terület a szélsőérték-problémák megoldása, illetve különböző rendszerek optimális paramétereinek meghatározása (a kettő tulajdonképpen ugyanaz). Ennek alapja az, hogy az evolúció során feltételezhetően egyre jobb és jobb egyedek születnek, idővel már

csak az optimum környékén, és olyan közel lesznek egymáshoz, hogy már megkülönböztethetetlenek lesznek [11, 12].

Genetikus algoritmusokat számtalan területen alkalmaznak különböző problémák megoldására, többek között a kémia számos területén [13] is. Ezek közül néhány általános és kémiai példa:

- genetikus (vagy evolúciós) programozás [14]
- struktúrák optimalizálása [12, 15, 16, 17], játékelméleti kérdések megoldása [11]
- térképfeliratok optimális elhelyezése, képfeldolgozás [11]
- függvények abszolút szélsőértékének meghatározása [18], görbeillesztés [19]
- műszerek beállításainak optimálása [20, 21, 22]
- makromolekulák, fehérjék elméleti szerkezetének meghatározása [23, 24, 25, 26, 27]
- molekulák szerkezetének meghatározása spektroszkópiai adatokból [28, 29, 30]
- spektrumok [31, 32], kromatogramok [34], diffraktogramok [33, 35] kiértékelése

A genetikus algoritmusok alkalmazásának szélsőérték-keresési feladatok megoldásában számos előnye van. A legfontosabb, hogy globális szélsőértéket keres és csak a végleges megoldás lesz stabil, azaz nem áll meg lokális szélsőértéknél (feltéve, hogy elég hosszú ideig fut). Másrészt csak minimális feltételeket szab az optimalizálandó rendszernek: a lehetséges megoldások véges számsorral (bináris, egész vagy valós) kifejezhetők legyenek (azaz létre lehessen hozni kromoszómákat), és bármely két egyedről el lehessen dönteni, hogy melyik a jobb. Mint minden modellezési feladatnál, itt is alapvető feltétel, hogy legyen optimum (ha lehet, egyetlen). Függvények minimumának kereséséhez, más algoritmusoktól eltérően, nem követeli meg a függvény folytonosságát, deriválhatóságát és a futtatáskor nem szükséges kiindulási paramétereket megadni. Hátránya viszont, hogy számolásigényes, és csak a pontos szélsőérték közelébe jut el, nem feltétlenül találja el azt (ez a numerikus közelítésekre általában igaz, különösen a véletlen választást használókra). Éppen ezért (főleg az előbbi miatt) nem szokás egyszerű optimálási feladatok megoldására használni, inkább olyanokra, ahol más módszer már nem segít, mint a szilárd NMR spektrumok paramétereinek meghatározása, habár ilyen alkalmazásra az irodalomban nem találtam hivatkozást.

Genetikus változók és műveletek

Általában egy genetikus algoritmust implementáló program futásához az alapvető „változókat” (kromoszóma, jósági függvény) és műveleteket (kiválasztás, keresztezés, mutáció) kell definiálni, valamint a megfelelő paramétereket (pl. a populáció mérete) jól beállítani. A következőkben ezeket fogom bemutatni.

Jósági függvények (fit)

Minden paraméteroptimálási feladat első lépéseinek egyike a paraméterek jóságának jellemzése. Ebben a programban kísérleti spektrumra illesztünk egy, a kvantummechanikai modell alapján számolt spektrumot, ami számos paraméter függvénye. Egy meghatározott paraméterkészletből egy külön programszál számolja ki az „elméleti” spektrumot, aminek a lehető legjobban kell illeszkednie a mérthez. A spektrumok hasonlóságának mérőszáma a *fit* vagy más néven a jósági függvény, amelynek abszolút minimumát keresi az algoritmus. Ez a program háromféle *fit*et használ: a szórásnak, az átfedésnek és a maximumnak nevezett függvényeket.

A szórás függvény általánosan elfogadott jellemzése két spektrum hasonlóságának. Matematikai értelemben ez a két spektrum, mint függvény különbségnégyzetének integrálja, azaz:

$$S = \sum_{\nu} (y_m(\nu) - y_c(\nu))^2 ,$$

ahol y_m a mért, y_c pedig a számolt spektrum értéke az aktuális ν értékre. Az összehasonlíthatóság érdekében a mért spektrumok legmagasabb csúcsa egységnyi. Előnye, hogy a keresés végén a zajt kiátlagolja, így ez adja a legpontosabb közelítést zajos spektrumokra. Hátránya, hogy amikor még nem a megoldás közelében vagyunk, a keskeny jelek nagy eltérését is kiátlagolja, ezért nem képes megtalálni a helyes eltolódásértékeket illetve forgási oldalsávokat.

Az átfedés valamivel ritkábban használt célfüggvény, eredetileg a két spektrum által közösen lefedett terület mérőszáma, elosztva a két spektrum teljes területével. Képlete:

$$O_0 = \frac{\left(\sum_{\nu} |y_k(\nu)| \right)^2}{\sum_{\nu} |y_c(\nu)| \cdot \sum_{\nu} |y_m(\nu)|} ,$$

ahol y_m a mért, y_c a számolt spektrum értéke az aktuális ν értékre, y_k pedig a közös függvény: azonos előjel esetén a kisebb abszolútértékű, ellenkező előjel esetén 0. A normálásra azért van

szükség, hogy a mért spektrumot szinte teljesen lefedő spektrumok közül a legjobbat (legkisebb területűt) ki lehessen választani. Ez a függvény teljes egyezés esetén 1, minden egyéb esetben 0 és 1 közötti valós számot adna. Az, hogy az optimális egyed *fit*je nem nulla, később részletezendő okok miatt rontja a keresés hatásfokát, ezért érdemes úgy módosítani, hogy 0 legyen. A negatív *fit* értékek elkerülése érdekében mindenhol a minimumot célszerű keresni, ezért az itt használt átfedési függvény egy kicsit eltér az eredetitől, de a spektrumok rangsora nem változik:

$$O = \left(\frac{\sum_{\nu} |y_c(\nu)|}{\sum_{\nu} |y_k(\nu)|} - 1 \right) \left(\frac{\sum_{\nu} |y_m(\nu)|}{\sum_{\nu} |y_k(\nu)|} - 1 \right)$$

Ennek már 0 a minimuma, és tetszőleges pozitív értéket felvehet. Ez a függvény a nagyon eltérő jeleket gyorsan kiszűri, tehát az eltolódásértékeket (és a forgási oldalsávokat) gyorsan megtalálja, viszont a többi paramétert már nehezen illeszti, mivel az egymást majdnem teljesen lefedő spektrumok közül a kisebb területűt, tehát esetleg a rosszabbat választja ki. Ezért a minimum közelében a szórás jobban használható, viszont induláskor az átfedés ad értelmes értékeket.

A keresés során egy evolúciós ciklusban az előző kettőből mindig csak az egyik jósági függvényt használom, attól függően, hogy milyen jól közelítem a spektrumot. A váltást egy harmadik *fit* értéke szerint szabályozom, ez a maximum függvény, a két spektrum különbségének abszolútértékének maximuma. Ez nem alkalmazható a keresési folyamatban, mivel nem elég érzékenyen különbözteti meg a jobb-rosszabb megoldásokat (éppen ezért nem szokás illeszkedés jellemzésére használni), és ezzel megakaszthatja, rossz irányba terelheti a keresést. Két spektrum egyezését viszont az emberi szem is többnyire ilyen módszerrel állapítja meg, ezért leállási feltételt vagy azt, hogy melyik *fit* szerint keressen (mennyire van közel a megoldáshoz), ezzel lehet jól meghatározni.

A keresésben a kétféle *fit* használatának további előnye, hogy mivel lokális minimumaik többnyire máshol vannak, ezeken a helyeken egymás ellen dolgoznak, és így gyorsabbá tehetik a programot. Éppen ezért a mind a keresés elején, mind a végén 5-10%-ban a másik *fit* (azaz az elején a szórás, a végén az átfedés) alapján is futhat evolúciós ciklus.

Kromoszóma, gének

Az egyedek jelen esetben a spektrumok, amelyeket kromoszómák reprezentálnak. (Az elnevezés félreérthető lehet, hiszen itt a kromoszóma az egyedet teljesen leírja, azaz a biológiában a teljes génállománynak felelne meg.) Ez a gének egyszálú sorozata, azaz egy

számokból álló tömb. Ebből egy külön programszál számolja a spektrumot a megadott modell alapján, tehát ez nem része az algoritmusnak. Ennek előnye, hogy az eljárás általánosan használható bármilyen jellegű spektrum paramétereinek becslésére, ha megvan a szimuláló program.

A genetikus algoritmusok többségétől eltérően a gének jelen esetben nem binárisak vagy egészek, hanem valós számok: minden gén egy-egy paraméter értéke. A gének sorrendje előre meghatározott, azaz mindegyik kromoszómában ugyanabban a sorrendben következnek egymás után. Ennek két szerepe is van: egyrészt a keresztezés csak így működhet hatékonyan (különben teljesen különböző nagyságrendű számokat keverne össze), másrészt így lehet a kromoszómákhoz hozzárendelni az egyedeket, azaz a génekkel, mint adott paraméterekkel kiszámolható spektrumot.

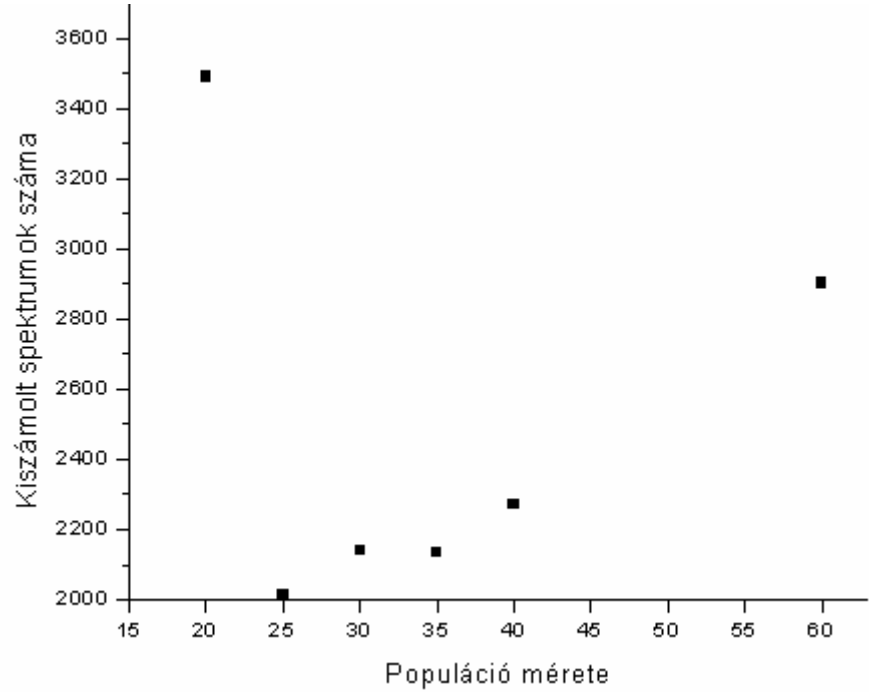
A hatékony működés további feltétele, hogy – lehetőleg – csak egyetlen optimális egyed legyen. Ugyanis ha több ilyen van, akkor a populáció egy idő után kettéosztódik (nem minden esetben, de elég gyakran) és a legjobb egyed váltakozva kerül ki a különböző alpopulációkból. Ez önmagában még nem lenne baj, csakhogy ezek az alpopulációk tulajdonképpen egy populációhoz tartoznak, ezért lehetnek közös utódaik. Ha két különböző alpopulációban nagyon különböző kromoszómájú, de egyforma egyedek vannak, akkor az utódlásnál – ahol csak a kromoszóma számít – mindkettőnél lényegesen rosszabb egyed jöhet létre, mivel génjeik összekeverednek. Nagyon sok „azonosan mély gödör” esetén ez félreviszi az evolúciót, mivel a szülőknél életképtelenebb egyedek jönnek létre (esetleg kizárólag) és a program nehezen (esetleg soha) nem konvergál. A másik probléma, hogy a populáció feleslegesen feldarabolódik, ezért nagyobb populációra lenne még akkor is szükség, ha nem keverednének egymással.

Ez a probléma egy jelből álló spektrum esetén nem merül föl, hiszen elméletileg csak egy jó megoldás létezik. Több jel esetén azonban ezek bármely permutációja jó, ami viszonylag kevés jel esetén már sok optimumot okoz (4 jel esetén 24, öt jel esetén már 120 egyformán jó megoldás létezik, ami figyelembe véve, hogy a populáció mérete 20 és 60 között van, sok). Ennek elkerülésére a magok mindig az eltolódásértékek szerint sorba rendezve kerülnek a kromoszómába (és ez is marad a sorrendjük), így már nem felcserélhetők. Ez már meghatározza a többi paraméter sorsát is, hiszen a spektrumot számoló eljárás „tudja”, hogy melyik paraméter melyik maghoz tartozik.

A magok sorba rendezésének hátránya, hogy nem lehet (nehéz) két magot felcserélni (abban a sorrendben maradnak, ahogy az első kromoszómában voltak). Ez akkor lehet probléma, ha két

jel nagyon közel van egymáshoz, és kezdetben nem tudjuk megállapítani, hogy melyiknek nagyobb az eltolódása.

A genetikus algoritmus lényege a többpontos keresés, azaz egyszerre több kromoszóma él, amelyek populációba tömörülnek. Ennek mérete a program futási idejére döntő hatással lehet. Kis populációnál egy ciklusban viszonylag keveset kell számolni, viszont kevés a kereszteződési lehetőség is, ezért a program csak sok generáció után találja meg az optimumot, vagy egy ahhoz közeli tartományt. Túl nagy populációméretnél viszont rengeteg kromoszóma *fit*jét kiszámolja, és ezek egy részét soha nem fogja felhasználni, mert rosszak, vagy mások kiszorítják. Ilyenkor már nem gyorsul annyit a keresés (ciklusszámban), mint amennyivel tovább tart egy ciklus. A 6. ábrán látható a kiszámolt *fit* értékek átlagos száma (azaz az átlagos ciklusszám és az egy ciklusban kiszámolt *fit*ek számának szorzata) és a populációméret közötti kapcsolat. (Az eredményt 1000 db, tetszőleges kezdőpontból indított keresés alapján számoltam. Az 500 ciklusnál hosszabb kereséseket leállítottam volna, de ilyen – ezeknél a paramétereknél – nem volt.).



6. ábra A populációméret hatása az átlagosan kiszámolt egyedek számára

Mint ezekből – és más hasonló tesztekből – látható, erre az algoritmusra az optimális érték 25 és 35 között van, jelenleg a program 30 tagú populációval számol.

Kiválasztás (selection)

A kiválasztás célja a szülők párosítása. Ennek első lépése meghatározni, hogy a kromoszómáknak milyen valószínűséggel lehet utódja. Ez történhet például egyszerű kizárással (a nagyon rosszaknak nem, a többieknek lehet utódja) vagy *fit*tel arányosan [36]. A programban a kettő keverékét alkalmazom: először is minden kromoszómának (illetve a hozzájuk rendelhető egyednek) kiszámolom a *fit*jét, majd ezeket átlagolom. Az átlagnál nagyobb értékeket adó (azaz rosszabb) egyedeket kizárom, a jobb értéket adó kromoszómáknak pedig annyi példánya kerül a szülőpopulációba, ahányad része a *fit*je az átlagnak (lefelé kerekítve). A kizárásnak köszönhetően az elején a teljesen rossz egyedek kiesnek, és a további keresésben elég kicsi a valószínűsége, hogy újra megjelennek. A *fit*tel arányos kiválasztás pedig biztosítja a nagyságrendekkel jobb egyedek érvényesülését. Például a 7. ábrán látható egy 20-as populáció szülőpopulációjának felépítése a keresés kezdetén és végéhez közeledve. Látható, hogy ahogy a populáció fejlődik, egyre jobban hasonlítanak egymásra az egyedek, ennek következtében körülbelül a populáció fele lesz jobb az átlagnál, és gyakorlatilag az egyszerű kizárás érvényesül (ilyen kis méreteknél a két- háromszoros valószínűség nem sokat számít). Ezzel szemben kezdetben a kiugróan jó kromoszóma gyakorlatilag uralja a szülőpopulációt és ezzel inkább a fitarányos kiválasztás szabályai érvényesek.

kromo	fit (o)	db	fit (i)	db
0	6,06E+01	1	2,12E-03	3
1	1,22E+01	6	7,85E-03	0
2	1,12E+03	0	1,44E-02	0
3	5,66E+00	14	4,16E-03	1
4	3,06E+00	27	9,76E-03	0
5	5,27E+01	1	3,83E-03	2
6	5,80E+00	14	1,31E-02	0
7	2,73E+00	30	3,56E-03	2
8	1,27E+01	6	4,37E-03	1
9	3,71E+00	22	9,05E-03	0
10	1,07E+02	0	1,37E-02	0
11	4,03E+00	20	1,71E-02	0
12	1,52E+02	0	4,70E-03	1
13	4,66E-01	178	7,07E-03	1
14	3,05E+01	2	1,37E-02	0
15	9,74E+00	8	2,85E-03	2
16	6,34E+01	1	5,81E-03	1
17	3,85E+00	21	7,91E-03	0
18	5,28E+00	15	7,33E-03	1
19	1,02E+01	8	1,72E-03	4
sum	8,32E+01	374	7,70E-03	19

7. ábra A fit értékek és a szülőpopuláció felépítése egy 20 tagú populációban a keresés kezdetén és végén.

Az utódképzés valószínűségében a *fit*ek aránya ilyen kiválasztásnál sokat számít. Ezt a jósági függvények megfelelő átalakításával lehet befolyásolni (az átalakítás a sorrendet nyilván nem befolyásolhatja, hiszen az egy új függvény lenne). Például, ha két egyed *fit*jének aránya eredetileg 2, és ehelyett a négyzete a jósági függvény, a *fit*ek aránya 4 lesz, ami gyorsabb konvergenciához vezet. Ez a konvergencia természetesen bármelyik lokális minimum közelében is gyorsabb lesz, tehát túl nagy kitevő esetén azokhoz is gyorsabban konvergál és ezért a program nehezebben zárja ki azt a lehetőséget. A megfelelő kitevő meghatározása még

nem történt meg ehhez a programhoz, jelenleg az előbbiekben megadott függvényeket használja.

A konvergenciát gyorsító másik módszer, mint már említettem, ha az abszolút minimum 0, és ennek érdekében a várt minimum értékét érdemes kivonni a *fit*ből. Ekkor ugyanis a keresés végén pl. két kromoszóma *fit*je 1,001 és 1,0001, ezek aránya 1 (tehát kb. egyforma valószínűséggel lesznek szülők) lenne; így viszont 0,001 és 0,0001 a *fit* értéke, és arányuk 10 lesz (feltéve, hogy 1 lenne az optimum). Ez a lépés a jó kromoszómák közti különbségeket jobban kiemeli, míg a nagyon rossz egyedeknél gyakorlatilag nem számít (pl. 10 és 11 esetén az arány körülbelül 1, a módosított *fit*ek pedig 9 és 10, az arány még mindig nagyjából 1). Az, hogy minimumot keres, annyiban egyszerűsíti a helyzetet, hogy pozitív számokkal kell számolni (maximum is lehetne a 0, ekkor minden negatív lenne).

A szülőpopulációból szülőpárokat az utódláshoz véletlenszerűen választódnak ki. Itt már minden egyednek ugyanakkora esélye van, de mivel a jobbak több példányban vannak jelen, az ő génjeik nagyobb valószínűséggel jutnak tovább (rulett-módszer) [8]. Ha egy kromoszóma egyszer már kereszteződött, akkor sem esik ki, hanem további egyedek szülője lehet. Tehát minden új kromoszóma létrejöttekor ugyanazok a valószínűségi viszonyok. (Másik lehetőség az lehetne, hogy a kiválasztott szülők kiesnek a populációból, azaz már nem lehetnek újra szülők.). A szülők kiválasztását és az utódképzést addig folytatja, amíg az új populáció mérete megegyezik a régiével, így a populáció mérete állandó.

Keresztezés (crossover)

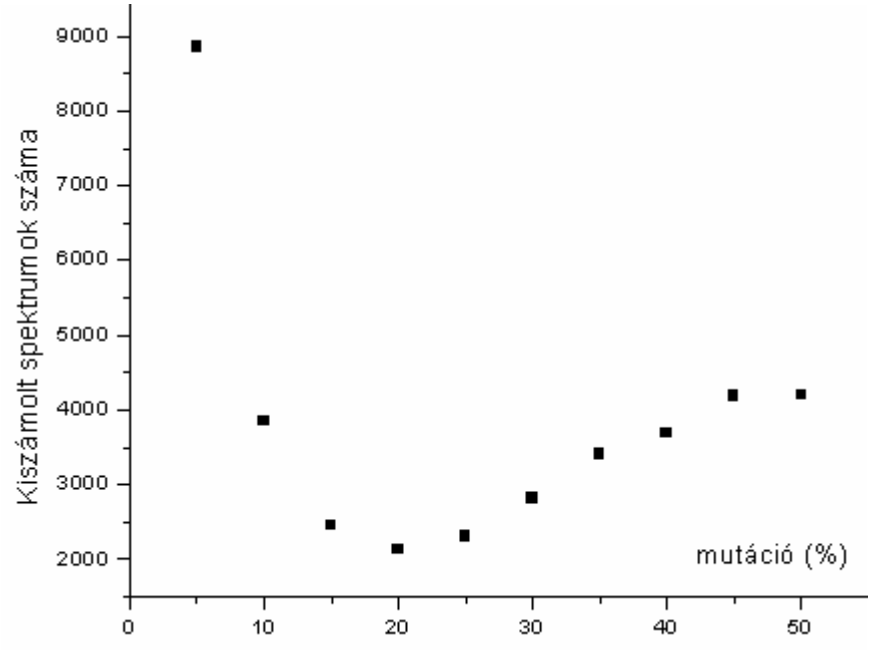
Bináris kódolású genetikus algoritmusoknál a keresztezés nagyon hasonlít a biológiai szaporodáshoz: a két kiválasztott kromoszóma egy véletlenszerűen kiválasztott helyen befűződik és az ezután következő részek kicserélődnek. A befűződések számának, vagy a befűződés valószínűségének megadásával lehet szabályozni a keveredést [8]. Ez a módszer alkalmazható valós gének esetén is, csak nagy hátránya, hogy ritkán változnak meg a gének lehetséges *allél*jai. Azaz, ha például kezdeti populációban ötvenféle lehetséges érték volt az első mag jelintenzitására, akkor ez az ötven érték fog variálódni (amelyek nagy része valószínűleg teljesen rossz), és így leszűkül a keresés. A másik – a programban is alkalmazott – lehetőség, hogy a keresztezés géneken belül történik. Ekkor nem fenyeget az előbb említett veszély, hiszen folyamatosan változnak a lehetséges paraméterek és a jobb értékek környékéről többféle lesz. Ebben a programban bármely két gén utódjának értéke a két szülő allél-értéke közötti véletlenszerűen kiválasztott szám (ha a két érték egyenlő, akkor

nyilván az utód is ugyanennyi lesz). A kromoszómák tehát génenként sorban kereszteződnek. Mivel a gének sorrendje minden kromoszómában ugyanaz, ezért automatikusan az azonos típusú gének fognak kereszteződni. Ennek a módszernek fontos következménye, hogy a középső értékeket nagyobb valószínűséggel találja meg, mint az értelmezési tartomány szélein lévőket (mivel a keresési tér egyre szűkül, és így a széle kiesik, ha nincs olyan értéket hordozó kromoszóma). Másik, elsőre félrevezető következménye az, hogy a gének értékei mindig megváltoznak, akkor is, ha jó volt, és akkor is, ha rossz. Ez viszont már magában a genetikus algoritmusban benne rejlő lehetőség, mivel az is előfordulhat, hogy egy kromoszómának az egyik génje tökéletes, de a többi miatt az egyed kiesik. Ez egyszerűen csak azt bizonyítja, hogy a géneknek önmagukban nincs értelmük (tehát nem lehetnek jók vagy rosszak), csak a teljes kromoszómának. Másrészt az igazán értékes kromoszómák génjei több példányban is öröklődnek, tehát nem veszhetnek el.

Mutáció (mutation)

A keresztezés, algoritmusából következően, mindig szűkíti a keresési teret. Elvileg a fejlődés irányába változtatja a kromoszómákat, és ezzel biztosítja a keresés determinisztikus voltát. Előfordulhat azonban, hogy átlépi az optimumot, és ezzel az kikerül a keresési körből. A program futásának végén pedig minden kromoszóma egyforma lesz (de nem biztos, hogy az optimális), hiszen nem szűkíthetem a végtelenségig. A megfelelő változatosság biztosításához van szükség a mutációra. A művelet algoritmusa nagyon hasonlít a keresztezésre: egy gén mutáció után az értelmezési tartományán belüli tetszőleges szám lesz (azaz nem csak a két szülőgén értéke között lehet). Ehhez szükséges minden génre az értelmezési tartomány megadása. A mutáció aránya egyike a legfontosabb futási paramétereknek. Túl nagy mutáció esetén véletlen kereséssé fajul a program, és ettől nagyon bizonytalan lesz a futási idő. Túl kicsi mutáció pedig beszűkíti a keresést és olyan, mintha nem is lenne. Bináris kódolású genetikus algoritmusokra maximum 0,5 – 1%-os mutációt (ott ez bitcserét jelent) javasolnak [8], itt azonban ennél jóval nagyobbra van szükség, mivel egyetlen valós szám többjegyű bináris kódnak felel meg.

A 8. ábrán látható, hogy hogyan függ az átlagosan kiszámolt *fit* értékek száma a mutációtól 30 fős populáció esetén. (A teszt a populációméret meghatározásánál elvégzettel azonos.) Amint látható, ennél az algoritmusnál az optimális mutáció 20% (15 – 25%).



8. ábra A mutáció valószínűségének hatása az átlagosan kiszámolt egyedek számára, 30-as populációméretnél

Mohóság

A természetes evolúciónak (és a korai genetikus algoritmusoknak) nem az optimum megkeresése, hanem életképes (jó) egyedeket kódoló kromoszómák megtalálása volt a célja, és alapvetően ezt lehet a genetikus algoritmusokkal megvalósítani. Ha a legjobb egyedeket keressük, akkor mindenképpen meg kell keresni az aktuális generáció legjobbját, hiszen ez a potenciális megoldás. Ezen kívül érdemes tárolni vagy az új populációba átmásolni (elitism [8]) az előforduló legjobb egyedet is, mivel a generációk változásával eltűnnek az addigi megoldások, és lehet, hogy egy későbbi körben rosszabb kromoszóma lesz a legjobb. Így minden generációnak lesz egy megoldási javaslata, de ez nem feltétlenül élő tagja a populációnak. A keresés tovább gyorsítható, ha ez a legjobb tag az evolúció során nem csak egy az egyben, hanem részletekben – génenként - is javítható. Ekkor az aktuális megoldás a következő lépésekben módosítható:

1. Ha még nincs globálisan legjobb egyed, akkor az adott generáció aktuálisan legjobb egyede (ALE) válik a globálisan legjobb egyeddé (GLE).
2. Amennyiben az aktuális generáció ALE-jének célfüggvénye kisebb, mint a GLE-é, akkor ez az egyed lesz a GLE.
3. Ha az előző pontok szerint nem keletkezett új GLE, akkor egy, az ALE és GLE egyedek génjein párhuzamosan végigfutó ciklus következik. E szerint az i-dik ciklusban egy új egyed jön létre GLE-ből úgy, hogy a GLE i-edik génje helyett az

ALE megfelelő génjét veszi át, majd kiszámolja ezen egyed célfüggvényét. Ha ez az érték kisebb, mint a GLE célfüggvénye, akkor ez az egyed válik GLE-vé. (a megoldás-kromoszóma „mohó algoritmussal” gyűjti a géneket).

Ezzel a módszerrel a 1000 kísérletből egyetlen keresés sem tartott 500 ciklusnál tovább (az átlag 53,5 ciklus), míg a „mohóság” nélkül ugyanilyen körülmények között futtatott keresések közül 912 nem fejeződött be 500 ciklus alatt.

Nyilvánvaló, hogy ilyen változtatások után a legjobb egyed már nem lehet hatással az evolúcióra, mivel akkor maga után húzhatja a populációt. Ennek köszönhető, hogy az előbbi kísérlethez hasonlóban az átlagos ciklusszám 64-re nőtt a már említett 53,5-ről.

Az összes kromoszómával a mohóságot nincs értelme végigjátszani, mert ez a ciklus idejét sokszorosára nyújtaná.

Kiindulási paraméterek

A genetikus algoritmus viszonylag nagy szabadságot hagy a felhasználónak. Nem igényel kiindulási értékeket, csak azt, hogy milyen paramétereket kell illeszteni, és ezek milyen határok között mozoghatnak. Ez utóbbiak viszont mindenképpen kellenek, különben a mutáció határait nem ismerjük. A kiindulási populációt a program véletlenszerűen generálja, ez az első lépés a teljes téren futó kereséshez. Ennek ellenére meg lehet adni kiindulási paramétereket, ezek egy kromoszómaként bekerülnek a populációba. Ha jó, esetleg ez a legjobb (a véletlenszerűen generáltak közül elég valószínű, hogy az általunk megadott lesz az), akkor meggyorsíthatja a keresést (a legjobb gyorsabban konvergál, a populációt viszont csak kis mértékben befolyásolja). A keresést el nem ronthatja, hiszen véletlenül is bekerülhetne egy ugyanilyen a populációba.

Az algoritmus további előnye, hogy a túlparaméterezés nem rontja el. A felesleges paraméterek futási hibát nem okoznak, csak a mohóság eljárásban többet kell számolni, és emiatt a program lassul (de egy-két ilyen paraméter esetén még nem jelentősen). Ez az előny hátrányként is jelentkezhet, hiszen nem derül ki a végeredményből, hogy a kapott érték hamis. Ennek elkerülésére a közeljövőben kiegészítjük egy eljárással, amely kiszűri az esetleges hatástalan paramétereket.

A genetikus illesztés a TopSpin programban

A TopSpin programcsomag a Bruker Biospin GmbH cég új NMR spektrométervezérlő és adatfeldolgozó programja, amely az alapvető funkciókon kívül új spektrumfeldolgozó modulokat is tartalmaz. Ezek közül a szilárd NMR spektrumok szimulációját végző fit1d modul fejlesztése témavezetőm közreműködésével folyik. E modulba szervesen integrálódik az általam írt genetikus algoritmus alapú paraméterillesztő eljárás.

A programcsomag egyik előnye a teljes egészében Java nyelven írt felhasználói felület, ami maximális hordozhatóságot biztosít különböző számítógéptípusokra és operációs rendszerekre. Az objektumorientált Java programozási nyelvben az objektumosztályok tulajdonságai és eljárásai írják le az algoritmusokat.

Az általam írt genetikus algoritmus négy osztályra épül: a `ChromoPop`, a `Chromosome`, a `GenSpectrum` és a `GenInterval` osztályokra. Ebből az első kettő a valódi genetikus objektum, utóbbiak csak egyszerűbbé teszik a (kísérleti és számolt) spektrumok és az intervallumok (többnyire értelmezési tartományok) kezelését, de magának a keresésnek nem részei. A `GenSpectrum` végzi két spektrum (általában egy számolt és a kísérleti) összehasonlítását, azaz az egyedek céfüggvényének (*fit*jének) meghatározását a `differenceInt`, az `overlap` és a `differenceMax` metódusokkal a már korábban tárgyalt háromféle jósági függvénynek megfelelően.

A `Chromosome` osztály egyedei, a kromoszómák, jelentik a keresés alapját. Ezeknek változói a gének (`dns` tömb elemei) és ezek értelmezési tartományai (`dnsInterval`), valamint a sorba rendezés miatt külön tömbben tárolt kémiai eltolódások (`shifts`). Mivel a kromoszómák *fit*jére életük során többször is szükség van, és az egyedek számolása az időigényes művelet, ezt az értéket (`fit`), valamint típusát (`fitMode`, értéke lehet i, o vagy m) is célszerű tárolni. Ezen kívül a következetes grafikus megjelenítés miatt szükség van még a magok eredeti sorrendjét tároló `shiftsOrder` tömbre, amelynek – hasonlóan a `dnsInterval` tömbhöz – minden kromoszómára azonosnak kell lenni, ezért osztályváltozók.

A kromoszómák közti műveletek két típusra oszthatók: a `fit` értékét számoló és a genetikus műveleteket végző eljárásokra. Ez előbbiek (a különböző argumentumú `fit` és `fitXXX` eljárások) gyakorlatilag a fit1d modul részét képező `GenQ` osztály segítségével kiszámolják a kromoszómához tartozó spektrumot (azaz az egyedet), és az így kapott `GenSpectrum`ok

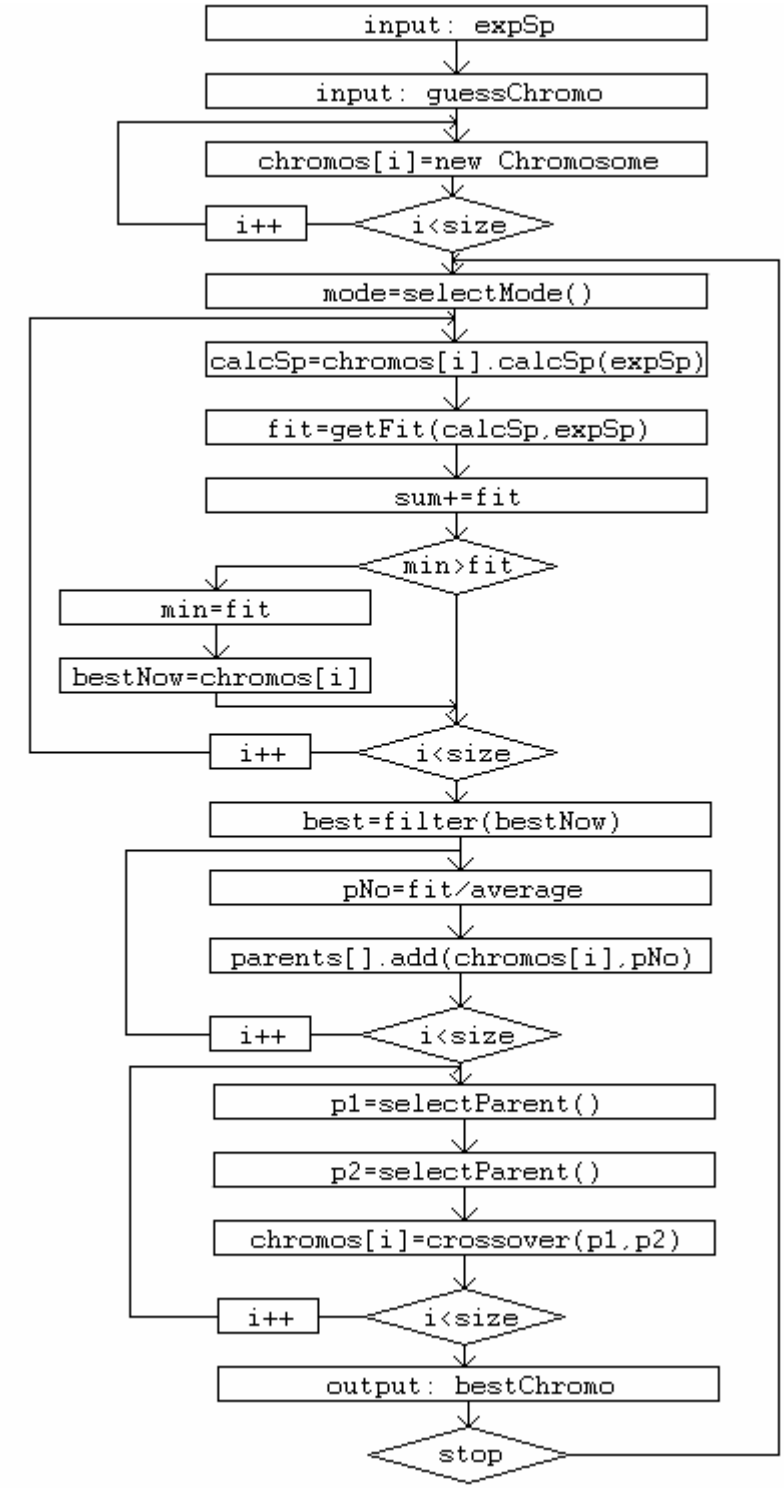
megfelelő eljárással számolt eltérése adja a `fit`et. Az alapvető kromoszómaszintű genetikus műveleteket (keresztezés és mutáció) a `child` eljárás hajtja végre, génekre lebontva. Az újonnan bevezetett „mohóságot” a `filter` eljárás végzi (elvileg két tetszőleges kromoszómára).

A `ChromoPop` osztály változói az adott populációt alkotó kromoszómák (`chromos`), valamint a félig-meddig efölött álló megoldás (`bestChromo`). Ez az osztály felelős az evolúcióért, így osztályváltozói a fejlődést szabályozzák. Mivel a genetikus résznek ez a legfelső szintje, ez tarja a kapcsolatot a TopSpin programmal a `ParamDialog` osztály egy egyedén keresztül.

A genetikus keresés a `ParamDialog` osztály `estimate` metódusának egyik ágában indul. Az előkészítés során a kísérleti spektrumot a `getSpectrum` metódus `GenSpectrum`má konvertálja (`expSp`), majd a kiindulási értékeket `Chromosome`-má alakítja a `ChromoPop` egyik konstruktora. Ugyanitt jön létre a random kiindulási populáció (`chromoPop`). Ezután indul az evolúciós ciklus: a `ParamDialog` meghívja a `chromoPop` `evolution` eljárását, amelyből minden ciklusban visszakéri a (megfelelően visszaalakított) `bestChromo`t, ami alapján a program az adatokat a képernyőn frissíti.

A 9. ábrán a program kromoszómaszintre lebontott folyamatábrája látható. Az első rész a kiindulási populáció létrehozása. Ezt követi az evolúciós ciklus, amelyet egy logikai változó állíthat le (stop). A ciklus részei az összes kromoszómán végigfutó ciklusok: a *fit*ek kiszámítása, a szülőpopuláció létrehozása és az új populáció születése. A legelső lépés a ciklus módjának meghatározása: ez dönti el, hogy a ciklus során melyik jósági függvényt használja a *fit*ek kiszámításához. A folyamatábrán selectMode-dal jelölt eljárás véletlenszerűen választ a két- vagy háromféle lehetőségből, amelyek valószínűségének aránya a bestChromo Max típusú *fit*jének (ez nagyobb vagy kisebb, mint a ChromoPop.CHANGE_SYS), valamint a ChromoPop osztály NOT_INT és NOT_OVLP változóinak függvénye. Ezek után az összes *fit*et számoló eljárás az így kapott módban fog számolni. Az első ciklusban minden egyes kromoszómának kiszámolja a *fit*jét, illetve ezzel párhuzamosan megkeresi a legjobban illeszkedő egyed kromoszómáját (`bestNow`), valamint kiszámolja a *fit*ek átlagát. A második ciklusban, az előzőekben meghatározott átlag segítségével, a fent leírt elvek alapján létrehozza a szülőpopulációt (`parents` tömb). A többször jelenlévő kromoszómákat természetesen nem tárolja sokszorosan, hanem csak egyszer, és külön tömbben (`parentsNo`) számon tartja, hogy elvileg mennyi van belőle (a

folyamatábrán parents.add, ahol pNo adja meg, hogy hányszor van jelen). Ezután a harmadik ciklusban kiválasztja a két szülőt és létrehozza az utódot, amíg fel nem tölti a populációt.



9. ábra Genetikus paraméterbecslés a TopSpinben

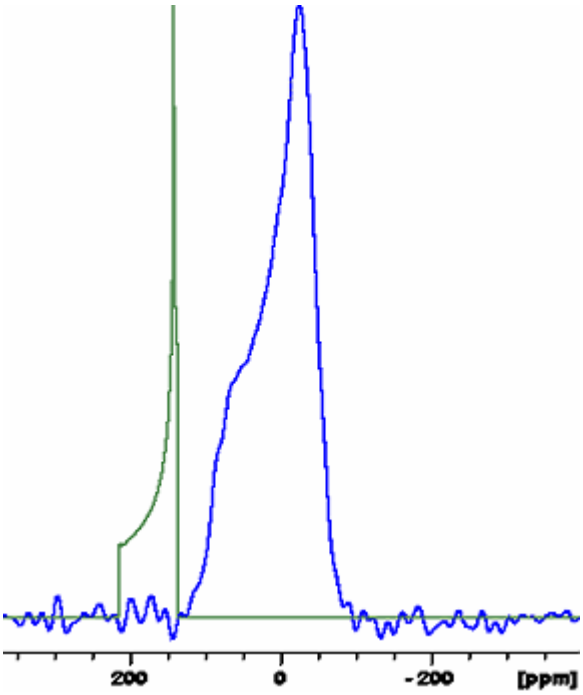
A genetikus illesztés bemutatása néhány példán keresztül

A következőkben a genetikus kereső algoritmus hatékonyságát és hibáit néhány példa mutatja be. Ezek célja kizárólag az algoritmus tesztelése, ezért a körülmények esetenként értelmetlenek az adott spektrum esetében (azaz: sokkal jobb kezdőfeltételekkel is lehetne keresést indítani).

A TopSpin programban a genetikuson kívül egy simplex kereső algoritmus van, így ezzel lehet összehasonlítani a működését. A további tervek egyike kombinálni a genetikus és a simplex keresést, és ezáltal mindkettőnél gyorsabb algoritmust létrehozni. Ez egyelőre még csak manuálisan lehetséges, de hatása így is jól látható. Az általában leghatékonyabbnak tartott Marquardt-féle paraméterbecslés az ilyen típusú feladatokra a tapasztalatok szerint nem működik megfelelően (csak nagyon pontos kiindulási értékekről), ezért ezekkel nem is hasonlítottam össze.

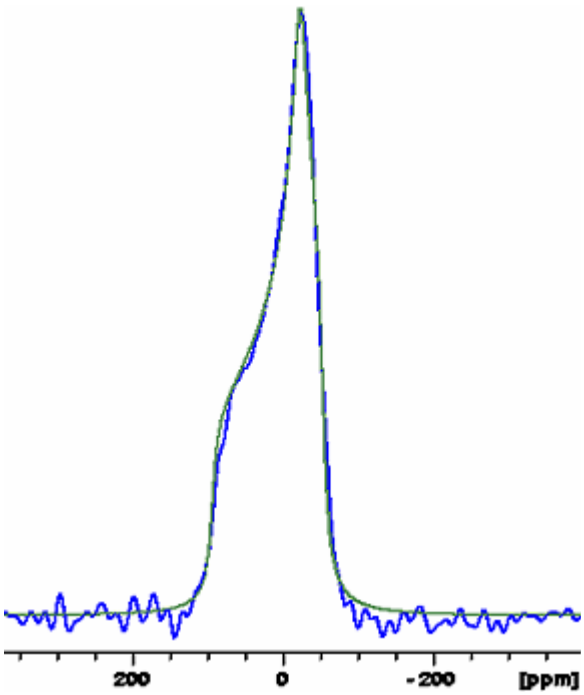
Dinátrium-fenil-foszfát ^{31}P-spektruma

Egyetlen ^{31}P magot tartalmazó dinátrium-fenil-foszfát ^{31}P NMR porspektruma mintaforgatás nélkül. (Eredeti Bruker spektrum, Field = 9,4 Tesla, SF = 161,98 MHz, SW = 125 kHz, SI = 2K)



10. ábra A mért spektrum (kék) és a kiindulási értékekből számolt spektrum (zöld)

→



11. ábra A mért (kék) és a becsült paraméterekből számolt spektrum (zöld) 50 ciklus után

A genetikus algoritmussal végzett illesztésnél kiindulási adatok megadása nem szükséges, azaz tetszőleges pontból indítva elvégezhető a keresés, például a 10. ábrán látható, nyilvánvalóan rossz értékekről is. A program genetikus algoritmussal 50 ciklus után a 11. ábrán látható eredményre jutott, ezután viszont a paraméterek értéke (és az illeszkedés) nem sokat változott. Az 50 ciklus után simplex módszerrel finomítva 208 ciklus alatt vált stacionáriussá a keresés, de az illeszkedésben ez szemmel látható változást ez nem jelentett, az eltérés csak paraméterek értékében mutatkozott meg. Az eredeti pontból rögtön simplex módszerrel indított keresés, nem adott értelmes végeredményt 500 ciklus alatt, és semmi remény nem volt arra, hogy valaha is adjon (a jel gyakorlatilag eltűnt az alapvonalban).

ciklusszám	0	50 genetikus	50 g. + 208 simplex	500 simplex
δ_{iso}/ppm	166,03	7,531	5,656	248,698
Int	185,9	187,7	187,4	1,8
σ/ppm	77,17	146,8	142,86	84,39
η	0,1	0,34	0,327	0,082
lb/Hz	0,0	2014,74	2285,21	0,0

1. táblázat Na_2PhPO_4 ^{31}P-NMR spektrumára illesztett paraméterek

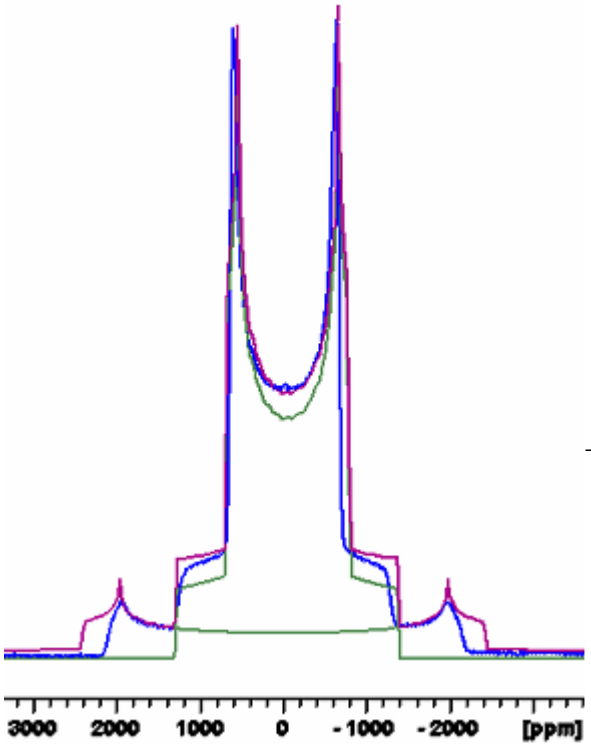
Megállapítható, hogy a genetikus algoritmus a paramétereket jól becsüli, míg a simplex ezeket a becsült értékeket hatékonyan finomítja.

Deuterált Plexi ^{2}H spektruma

Kétféle ^{2}H magot tartalmazó deuterált plexi ^{2}H NMR porspektruma mintaforgatás nélkül. (Eredeti Bruker spektrum, Field = 4,7 Tesla, SF = 30,72 MHz, SW = 2500 kHz, SI = 4K.)

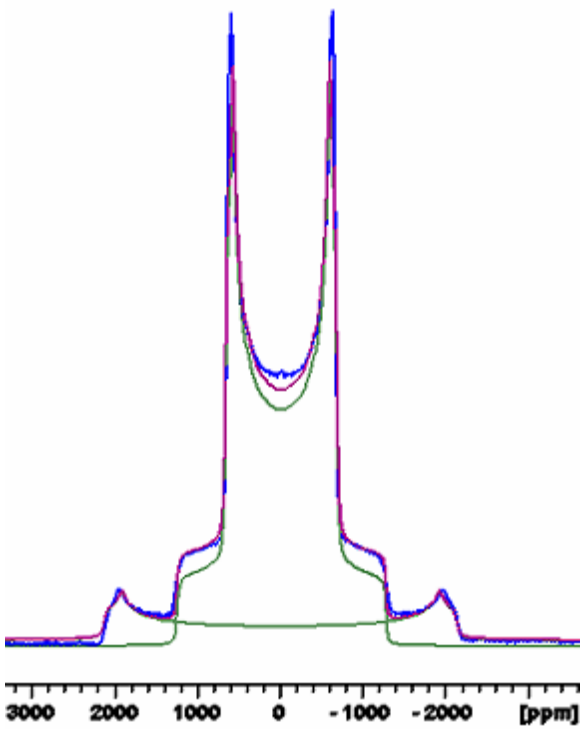
ciklusszám	0	150 genetikus	500 simplex
$\delta_{iso}(1)$/ppm	-36,146	-1,723	-36,158
int(1)	16,8	15,1	15,4
$c_Q(1)$/kHz	55	52	52
$\eta_Q(1)$	0,1	0,0661	0,09
lb(1)/Hz	0	648,33	383,34
$\delta_{iso}(2)$/ppm	0,006	0,06	-7,484
int(2)	2,1	1,5	1,8
$c_Q(2)$/kHz	180	166	164
$\eta_Q(2)$	0,1	0,053	0,042
lb(2)/Hz	0	1662,10	101,64

2. táblázat Deuterált plexi 2H spektrumára illesztett paraméterek



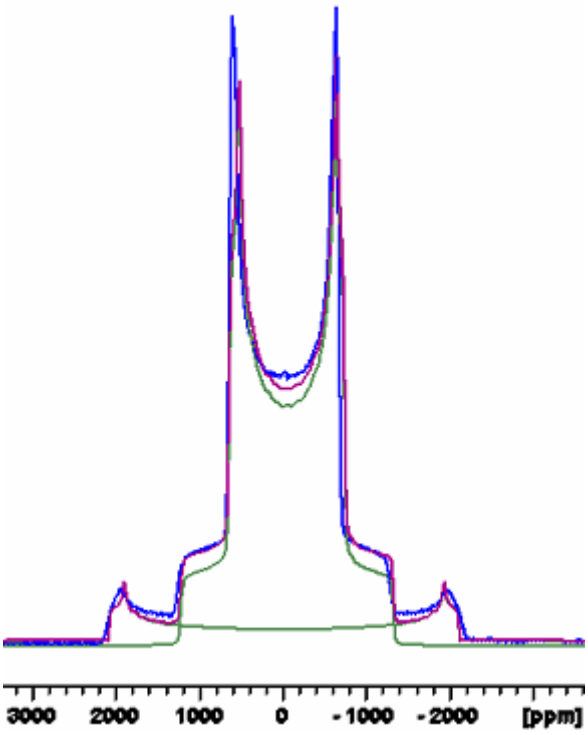
12. ábra A mért (kék), és a kiindulási paraméterekből számolt spektrum (zöld: a magokra külön, lila: az összegük)

→



13. ábra A mért (kék) és a 150 genetikus ciklus utáni értékekből számolt spektrum (zöld: a magokra külön, lila: az összegük)

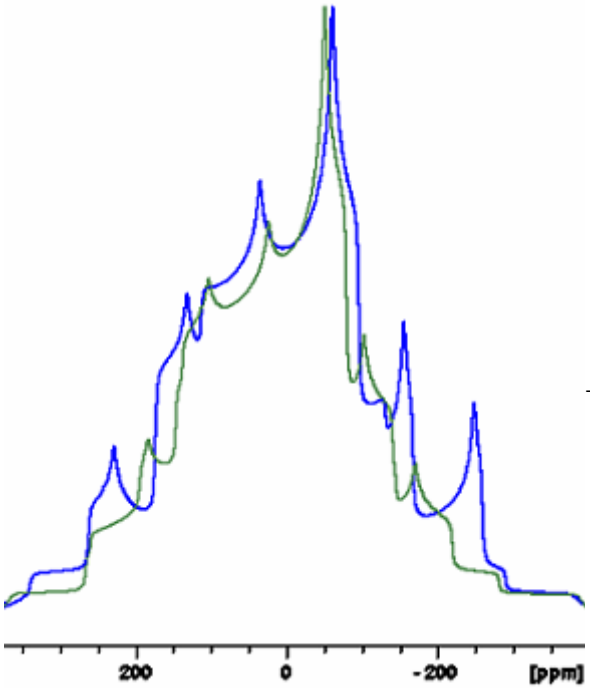
Ennél a spektrumnál már értelmes kiindulási paraméterektől indult a keresés (12. ábra). A genetikus algoritmus 100 ciklus után viszonylag jól eltalálta a megfelelő értékeket, ezután csak lassan változtatott a paramétereken, ezért 150 ciklusnál nem volt értelme tovább futtatni (13. ábra). A simplex közelítés valamivel rosszabb eredményt hozott 500 ciklus után, a 14. ábrán az ekkor beállt állapot látható. A két eredmény összevetéséből látható, hogy itt ugyan csak két paramétert nem becsültünk meg (a két félértékszélesség), de még így is gyorsabban megtalálja a genetikus algoritmus. A genetikus után indított simplex iteráció a paraméterek pontosítását már 200 ciklusban elvégzi.



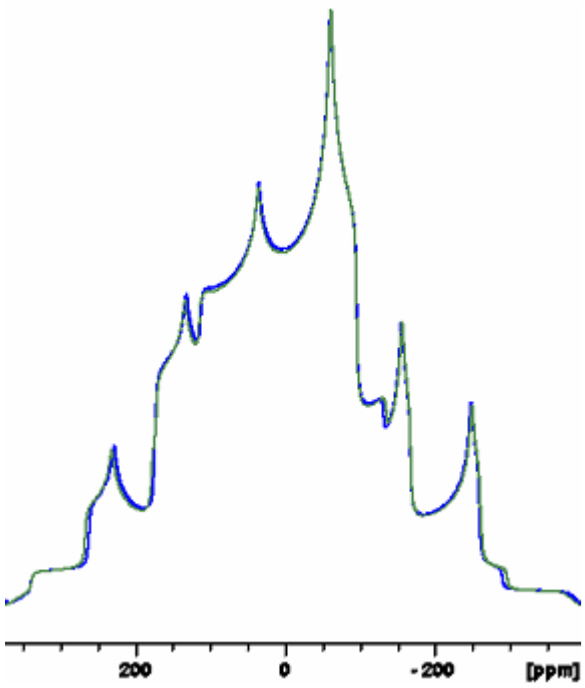
14. ábra A mért és az 500 simplex ciklus után kapott adatokból számolt spektrum

Szimulált spektrum dipoláris csatolással

Ez a spektrum egy szimulált spektrum, amelynek paraméterei a következők: detektált mag: ^{31}P, amelynek paraméterei: δ_{iso} = 13,736 ppm, int = 176,7, σ = 142,86 ppm, η = 0,487, lb = 515,21 Hz. A rendszer két maggal csatol dipolárisan: egy ^{17}O (5/2 spinű, paraméterei: D = 15700 Hz, a_2 = 29°, a_3 = 71°) és egy ^{1}H (1/2 spinű, paraméterei: D = 13000 Hz, a_2 = 58°, a_3 = 30°). A cél az volt, hogy minél bonyolultabb spektrumot kelljen genetikus algoritmussal közelíteni. Ebben az esetben, ha akarunk, sem tudunk olyan becslést megadni, mint az előző példákban tudtunk volna, mert a paraméterek kis mértékű változtatására is jelentősen megváltozik a spektrum, és ráadásul, ha csak a két dipoláris csatolás összesen hat paraméterét változtatjuk, akkor sem lehet egyesével beállítani őket. Ez tipikusan az az eset, amikor genetikus algoritmust érdemes használni.



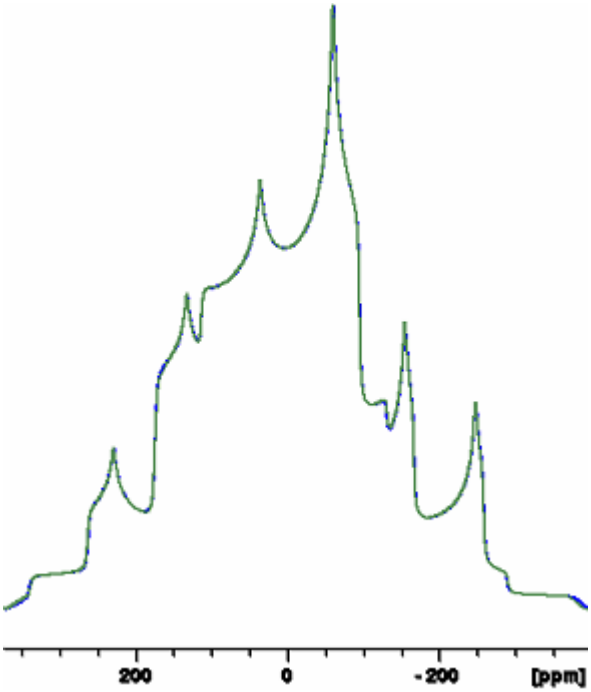
15. ábra Az eredeti (kék) és a kiindulási – becsült – adatokból számolt spektrum (zöld)



16. ábra Az eredeti (kék) és a 40 genetikus ciklus után kapott spektrum (zöld)

A simplex módszerrel a 15. ábrán látható állapotból indított keresés 500 ciklus alatt adott gyakorlatilag tökéletesen illeszkedő eredményt, a genetikus algoritmus 40 ciklus alatt elérte az általános zajszintnek megfelelő korlátot (16. ábra), ezután indítva a simplex keresést 100 ciklusban megkapjuk a hasonlóan illeszkedő megoldást, mint az előbb 500 ciklus alatt (17. ábra). Látható azonban, hogy utóbbi esetben jóval nagyobb tartományt járt be a keresés nagyjából azonos idő alatt. Ezek az adatok nem egyeznek meg a kezdetiekkel, de vizuálisan teljesen jó eredményt kaptunk (17. ábra) és „ismeretlen” spektrum esetén nyilván ezeket is

elfogadnánk helyes végeredménynek. Az eltérések oka nem az, hogy a spektrum ezektől a paraméterektől nem függ, hanem feltételezhetően nem egyértelműek ezek az értékek (azaz több abszolút minimum is van a keresési térben).



17. ábra Az eredeti (kék) és a simplex módszerrel továbbfinomított spektrum (zöld).

ciklusszám	szimulált	0	40 genetikus	40 g. + 100 simplex	500 simplex
D(1)/Hz	15700	13000	15871	15704	15695
$a_2(1)$/°	29	110	129	129	109
$a_3(1)$/°	71	0	32	32	5
D(2)/Hz	13000	13000	10771	10678	13789
$a_2(2)$/°	58	76	79	78	53
$a_3(2)$/°	30	15	37	37	23

3. táblázat A szimulált spektrumra illesztett paraméterek

Összefoglalás

A Bruker TopSpin programjának szilárd NMR spektrumok szimulációjára szolgáló moduljához olyan Java nyelvű programot készítettem, amely genetikus algoritmussal becsüli a szilárdfázisú NMR spektrumok paramétereit. Ez az algoritmus elvileg bármilyen bonyolult spektrum modellezésére használható. Számolásigénye miatt azonban elsősorban olyan összetett spektrumok esetén célszerű használni, amikor más módszerek (pl. gradiens módszer) nem bizonyulnak hatékonynak. A tesztek alapján elmondható, hogy az eljárás megbízható más szélsőérték kereső algoritmusokkal kombinálva is: amikor már láthatóan az abszolút szélsőérték közelében van, el lehet indítani egy simplex iterációt. E keresőmódszerek hatékony kombinálása a további fejlesztések egyik lehetséges iránya. További cél kiterjeszteni a működését más típusú spektrumokra (dinamikus NMR), illetve megvalósítani több spektrum közös paramétereinek együttes becslését.

Köszönetnyilvánítás

Köszönetet mondok témavezetőmnek, Rohonczy Jánosnak, a rengeteg hasznos tanácsért és az algoritmus TopSpinbe történő beépítésében nyújtott segítségért.

Irodalomjegyzék

1. Szalontai G.: NMR vizsgálatok szilárd fázisban (sparc4.mars.vein.hu/nmr/letoltes.html)
2. Mehring: High Resolution NMR Spectroscopy in Solids, *Springer-Verlag, Berlin*, 1983
3. Rohr-Spiess: Multidimensional Solid-State NMR and Polymers, *Academic Press, London,* 1994
4. Haeberlen, High Resolution NMR in Solids, *Suppl. I. Academic Press, New York*, 1976
5. J. Mason: Conventions for the reporting of nuclear magnetic shielding (or shift) tensors suggested by participants in the NATO ARW on NMR shielding constants, *Solid State Nuc. Magn. Res.*, **2**, 285 (1993)
6. R.K. Harris: Conventions for tensor quantities used in NMR, NQR and ESR, *Solid State Nuc. Magn. Res.*, **10**, 177 (1998)
7. Rob Schurko: Solid State NMR (http://mutuslab.cs.uwindsor.ca/schurko/ssnmr)
8. C. Darwin: On the Origin of Species, *John Murray, London*, 1859, (C. Darwin: A fajok eredete, *Magyar Helikon, Budapest*, 1973)
9. Barricelli, N.A.: Numerical testing of evolution theories, *Acta Biotheoretica*, **16**, 94 és122 (1962)
10. J. H. Holland. Adaptation in Natural and Artificial Systems. *The University of Michigan Press, Michigan*, 1975
11. Álmos A., Győri S., Horváth G., Várkonyiné Kóczy A.: Genetikus algoritmusok, *Typotex* 2002
12. Marek Obitko: Genetic Algoritms, 1998, http://cs.felk.cvut.cz/~xobitko/ga/
13. R. Leardi: Genetic algorithms in chemometrics and chemistry: a review, *J. Chemometrics* **15**, 559 (2001)
14. J. R. Koza. Genetic Programming. *The MIT Press, Cambridge, Massachusetts*, 1992 és J. R. Koza, F. H. Bennett III, D. Andre, M. A. Keane, Genetic Programming III: Darwinian Invention and Problem Solving, *Morgan Kaufmann, San Francisco, California*, 1999
15. Hiroaki Sengoku: A fast TSP solver using a genetic algorithm, http://www-cse.uta.edu/~cook/ai1/lectures/applets/gatsp/TSP.html, 1996
16. Robert Thomson: A Genetic Algorithm Demo, http://oldeee.see.ed.ac.uk/~rjt/ga.html

17. Jean-Philippe Rennard: Introduction to Genetic Algorithms, http://www.rennard.org/alife/english/gavintrgb.html

18. A. Dolan: Genetic Algorithms Toolkit, http://www.aridolan.com/ga/gaa/gaa.html

19. A. P. De Weijer, C. B. Lucasius, L. M. C. Buydens, G. Kateman, H. M. Heuvel, H. Mannee: Curve fitting using natural computation, *Anal. Chem.*, **66**, 23 (1994)

20. M.J. McShane, B.D. Cameron, G.L. Cote, C.H. Spiegelman: Improving complex near-IR calibrations using a new wavelength selection algorithm, *Appl. Spectrosc.*, **53**, 1575 (1999)

21. Andris, P.; Frollo, I.: Optimization of NMR coils by genetic algorithms *Measurement Science Review*, **2**, 2 (2002)

22. D. Jouan-Rimbaud, D. L. Massart, R. Leardi, O. De Noord: Genetic algorithms as a tool for wavelength selection in multivariate calibration, *Anal. Chem.* **67**, 4295 (1995)

23. R. Unger and J. Moult: A genetic algorithm for 3D protein folding simulations, *Journal of Molecular Biology*, **231**, 75, (1993).

24. S. Sun: Reduced representation model of protein structure prediction: Statistical potential and genetic algorithms, *Protein Science*, **2**, 762, (1993)

25. S. Schulze-Kremer: Genetic algorithms and protein folding, http://www.techfak.uni-bielefeld.de/bcd/Curric/ProtEn/proten.html

26. R. S. Judson: Teaching polymers to fold, *The Journal of Physical Chemistry*, **96**(25), 10102, (1992)

27. C. Roberts, R. L. Johnston: Investigation of the structures of MgO clusters using a genetic algorithm, *Phys. Chem. Chem. Phys.* **3**, 5024 (2001)

28. Meiler J, Will M.: Genius: a genetic algorithm for automated structure elucidation from 13C NMR spectra, *J Am Chem. Soc.* **124**(9) 1868 (2002)

29. D. D. Stranz, LeRoy B. Martin: Derivation of Peptide Sequence from Mass Spectral Data using the Genetic Algorithm, http://www.abrf.org/JBT/Articles/JBT0004/JBT0004.html

30. T. Blenkers, P. Zinn: Application of Genetic Algorithms to Structure Elucidation of Halogenated Alkenes Considering the Corresponding ^{13}C NMR Spectra, *Croatica Chimica Acta* **77**, 213 (2004)

31. W. L. Meerts, M. Schmitt, G. C. Groenenboom: New applications of the genetic algorithm for the interpretation of high-resolution spectra, *Can. J. Chem. / Rev. Can. Chim.* **82**(6), 804 (2004)

32. Fang L. Junde W.: Using Genetic Algorithm to Identify Completely Unknown System in FTIR Spectra Analysis, *Journal of Environmental Science and Health*, Part A, **39**(6) 1525 (2004)

33. Habershon, S., Harris, K.D.M., Johnston, R.L., Turner, G.W. Johnston, J.M.: Gaining Insights into the Evolutionary Behaviour in Genetic Algorithm Calculations, with Applications in Structure Solution from Powder Diffraction Data, *Chem. Phys. Lett.*, **353**, 185 (2002)

34. W. Cai, F. Yu, X. Shao, Z. Pan: Resolution of Overlapping Chromatographic Peaks Using a Genetic Algorithm, *Anal. Chem.* 2000, **33**(2) 373

35. B. Mukherjee: ANDI-03: a genetic algorithm tool for the analysis of activation detector data to unfold high-energy neutron spectra, *Radiation Protection Dosimetry* **110** (1-4) 249-254 (2004)

36. A. Schatten: Genetic Algorithms Short Tutorial, http://www.schatten.info/info/ga/genetic.html