



ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

TUDOMÁNYOS DIÁKKÖRI DOLGOZAT

A MAGYARORSZÁGI ROBINIA PSEUDOACACIA ERDŐK DETEKTÁLÁSA SENTINEL-2 MŰHOLDFELVÉTELEK ALKALMAZÁSÁVAL

Szerző:

Böröcz Balázs

Mérnök-informatikus MSc. szak, II. évf.

Konzulens:

Dr. habil. Molnár Gábor Péter

egyetemi docens

Székesfehérvár, 2024.



Tartalomjegyzék

Tartalmi kivonat	3
1. Bevezetés.....	4
2. Adatok	5
3. Módszertan	8
3.1 Megfelelő műholdfelvételek kiválasztása	8
3.2 Az elemzés kezdeti szakasza.....	9
3.3 Szórásképek.....	12
3.4 Mahalanobis távolság	17
3.4.1 Matematikai műveletek végzése képeken	20
3.4.2 A Mahalanobis távolság számításával kapott eredmények	23
4. Eredmények kiértékelése és Diszkusszió	26
5. Konklúzió	28
Irodalomjegyzék	29
Köszönetnyilvánítás	31



Tartalmi kivonat

A *Robinia Pseudoacacia* (magyar nevén fehér akác) egy rendkívül invazív faj, melynek fája értékes alapanyagként szolgál akár szerszámkészítéshez, akár magas kalória tartalma miatt jó tüzelőanyagként is felhasználható. Fontos szerepet tölt be továbbá a magyarországi méhészetekben, hiszen az elsőszámú méhlegelőként tartjuk számon Magyarországon.

Jelenleg Magyarország erdőterületeinek 25%-a *Robinia pseudoacacia*, és ez a szám folyamatosan növekszik, így környezetvédelmi szempontból létfontosságú a megfigyelése. A *Robinia pseudoacacia* megfigyeléséhez Észak Magyarország területén található, Szirák és Vanyarc települések közelében elhelyezkedő, *Robinia pseudoacacia* erdők virágzásáról készült felvételeket alkalmaztam. A műholdas felvételek alapján történő azonosításához a Mahalanobis-távolság számítását használtam. A Mahalanobis-távolság egy statisztikai módszer, amelyet több változó közötti távolság mérésére használnak, figyelembe véve a változók közötti korrelációt.

A feladatot a Google Earth Engine segítségével hajtottam végre, amely egy felhőalapú térinformatikai rendszer. Az eredmények segítségével új képek készültek, amelyek lehetővé teszik a *Robinia pseudoacacia* erdőterületek lehatárolását. A vizsgálat során összehasonlítottam a látható tartományban lévő sávokkal végzett elemzést a Sentinel-2 összes spektrális sávján végzett elemzéssel.



1. Bevezetés

TDK dolgozatom fókuszában a magyarországi *Robinia pseudoacacia* faj (magyar nevén fehér akác) területi kiterjedésének monitorozása áll, melyet Sentinel-2 műholdfelvételek felhasználásával kívánok véghez vinni. A *Robinia pseudoacacia* a pillangósvirágúak (*Fabaceae*) család *Robinia* nemzetségének tagja. Eredendően Észak-Amerika és Mexikó meleg, időszakosan száraz részeiről származik. Hosszúéletű (200-250 év), gyorsan nöövő fafaj, mely Magyarországon a 18. század elején jelent meg, tömeges telepítése pedig a 19. század második felében történt. Rendkívül invazív faj, mely jelenleg Magyarország erdőterületeinek közel 25%-át lefedi [1]. Fája értékes alapanyagként szolgál szerszámkészítéshez, valamint magas kalória tartalma miatt rendkívül jó tüzelőanyag. Külön szeretném kiemelni mint legfontosabb hazai mézelő növényt, melynek méze, az akácméz hungarikumként van számon tartva.

A dolgozat megírásához motivációmát elsősorban a szüleim családi vándorméhészete jelentette. A vándorméhészet szakkifejezést azokra a méhészetekre használjuk, melyek a méhlegelők virágzását követve vándorolnak az ország területén. Sajátosságai, hogy a vándorlásnak köszönhetően több mézfajtát képesek előállítani, valamint nagyobb termés hozamokat képesek produkálni. Az ágazat körülbelül 70%-át érinti a rendszeres vándoroltatás [2].

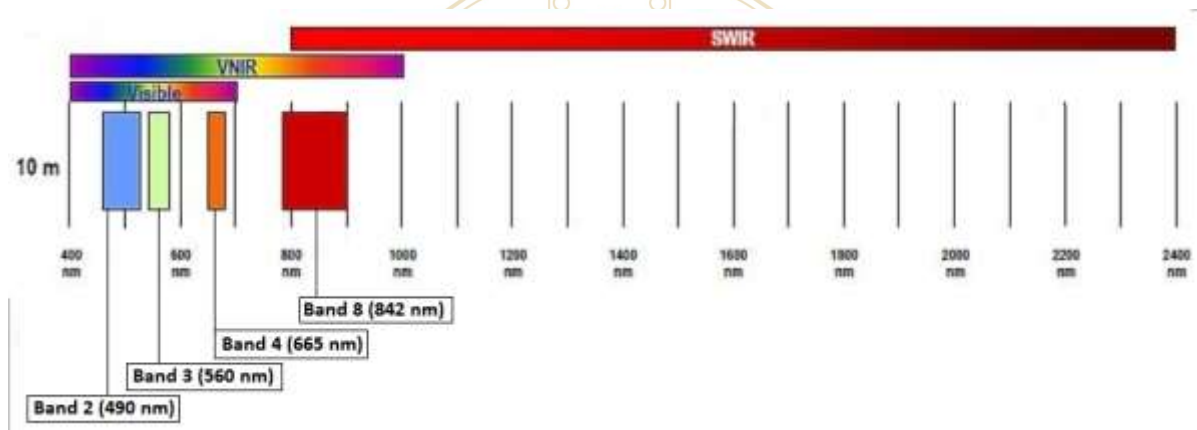
Magyarország legfontosabb méhlegelői virágzási sorrendben, a repce (*Brassica napus*), a **fehér akác** (*Robinia pseudoacacia*), selyemkóró (*Asclepias syriaca*, elsősorban az Alföldön jelentős), napraforgó (*Helianthus annuus*), valamint a napraforgóval közel egy időben virágzó aranyvessző (*Solidago canadensis*, elsősorban a Dunántúlon jelentős). Természetesen egyéb méhlegelők is előfordulnak hazánkban, ám azok méhészeti szempontból kevésbé jelentősek.

A *Robinia pseudoacacia* megfigyeléséhez felhasznált Sentinel-2 műholdfelvételek ingyenesen elérhetőek bárki számára, a felvételek feldolgozását azonban nem osztályozással kívántam elérni, hanem indexeléssel. Az indexelést a Mahalanobis távolság kiszámításával hajtottam végre.

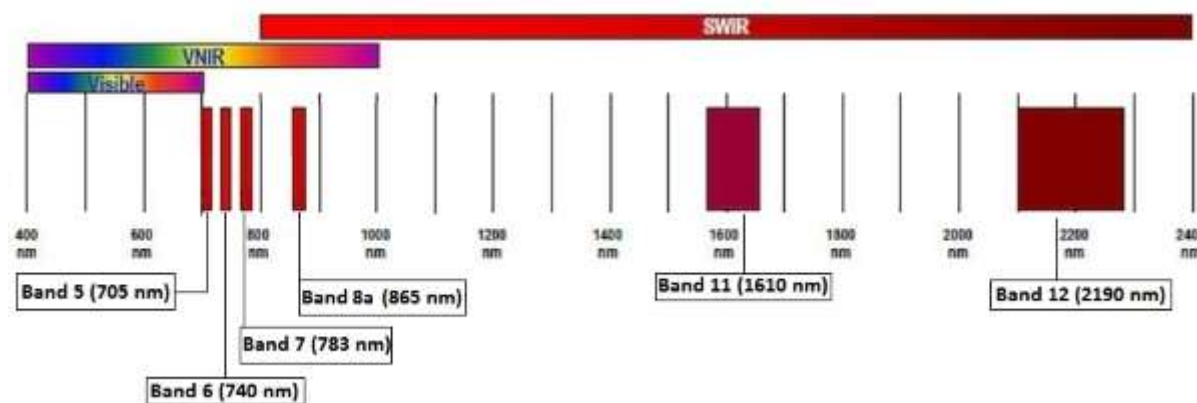
2. Adatok

A dolgozat elkészítéséhez Sentinel-2 műholdfelvételeket alkalmaztam. A Sentinel-2 műholdak a Sentinel műholdcsalád részét képezik, melyet az Európai Űrügynökség (European Space Agency, ESA) üzemeltet a Kopernikusz program keretein belül.

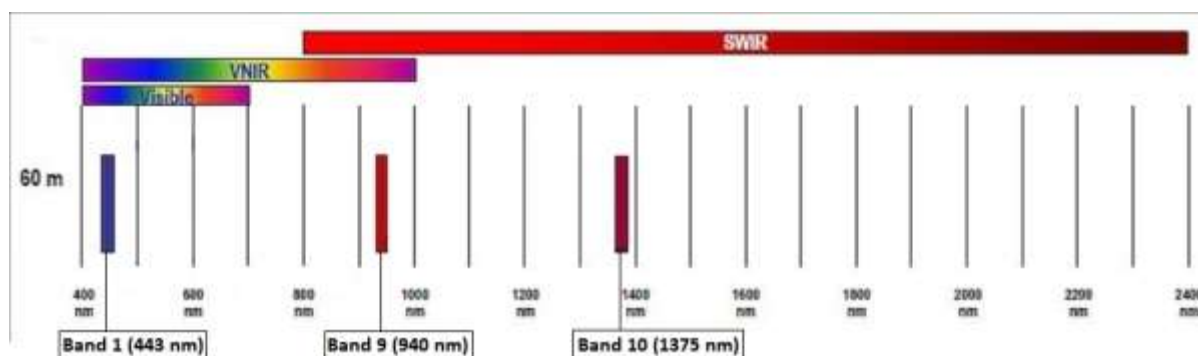
A műholdak multispektrális szenzorokkal vannak felszerelve, melyekkel 13 sávban készítenek felvételeket, a látható fénytartománytól a közepes infravörös tartományig. Térbeli felbontása sávonként eltérő, 4 sávban 10 m, 6 sávban 20 m és 3 sávban pedig 60 m felbontással rendelkezik. [3] [4]



1. ábra: Sentinel-2 10 méteres felbontású sávok



2. ábra: Sentinel-2 20 méteres felbontású sávok



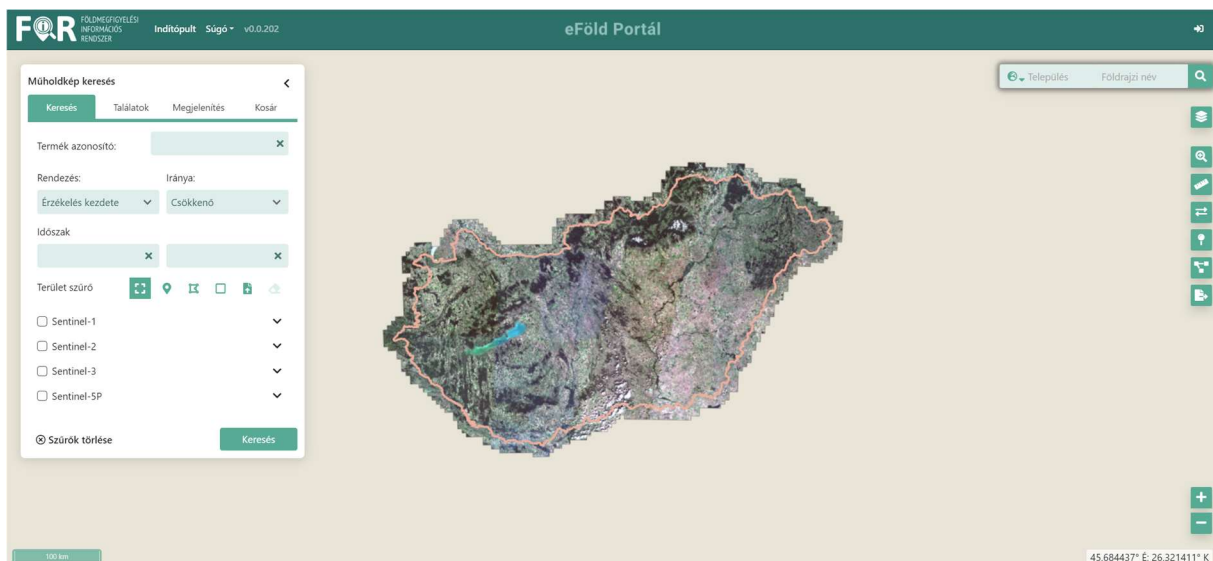
3. ábra: Sentinel-2 60 méteres felbontású sávok

A feladat elkészítéséhez elsősorban a látható fénytartományban készült sávokat használtam fel, azonban készítettem olyan elemzéseket is, mely során mindegyik sávban készült felvételt felhasználtam.

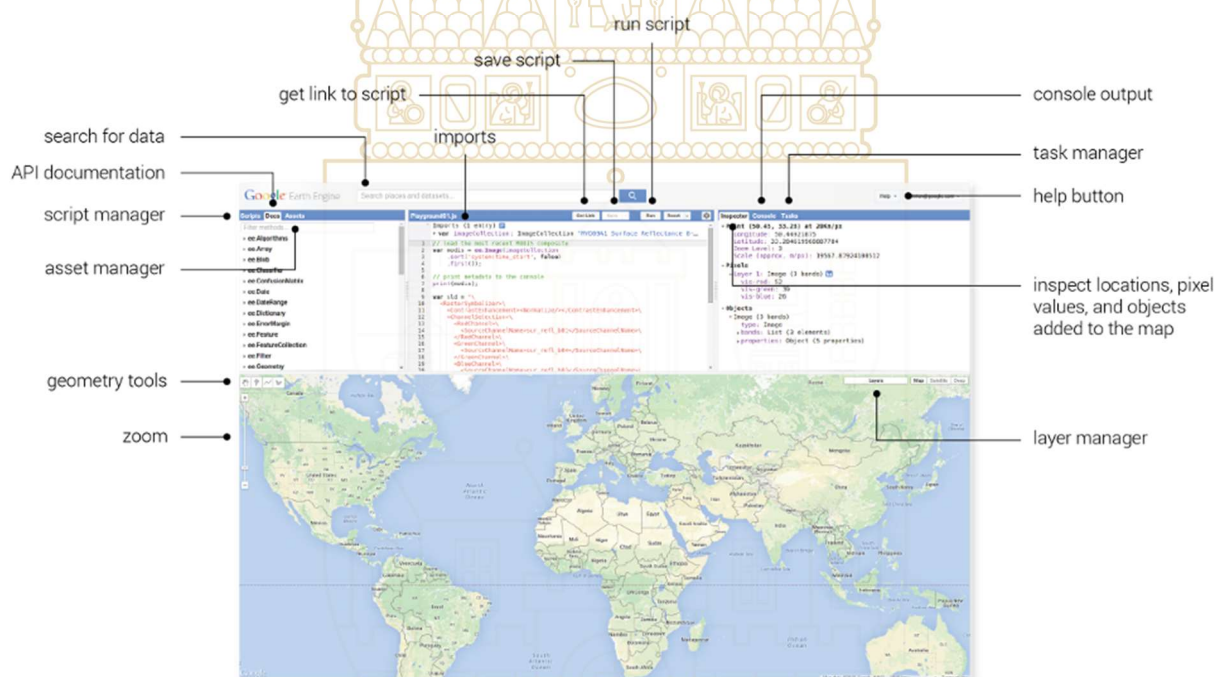
A Sentinel-2 két műholdból álló műhold párt alkot (Sentinel-2A és Sentinel-2B) melyek egymással 180° -os szöget zárnak be és napszinkron pályán keringenek. A műholdak visszatérési ideje 10 nap, mivel azonban a két műhold 180° -ot zár be egymással, ugyanarról a területről 5 naponta készítenek képet [3] [4].

A felvételek elérhetőek és ingyenesen letölthetőek többek között az ESA által üzemeltetett „Copernicus Open Access Hub” oldaláról [5], a Lechner Tudásközpont által üzemeltetett „Földmegfigyelési Információs Rendszer (FIR)” oldaláról [6] és számtalan egyéb oldalról, azonban célra vezetőbbnek tartottam, egy olyan webes szolgáltatást igénybe venni, ahonnan nem kellett letölteni a felvételeket és a különböző számítási műveletek Cloud alapon elvégezhetőek. Erre megoldást nyújtott az Alphabet cég által üzemeltetett Google Earth Engine felhő alapú BigData környezete, mely kutatási célokra ingyenesen elérhető [7].

A Google Earth Engine felhasználói felülete öt fő részre tagolódik. Baloldalon található scriptkönyvtár és dokumentációk. Középen található a script szerkesztő, melyben JavaScript nyelven tudunk programozni. Jobb oldalon található a konzol ablak, ahova az eredményeket tudjuk kiíratni. Alul a térkép látható, felül pedig egy kereső, mellyel helyekre, adatokra, és funkciókra is rá tud keresni a felhasználó. A rendszer elérhető külső Python-os környezetből is, mint például a QGIS. Munkám elkészítése során a saját környezetében használtam a rendszert.



4. ábra: Földmegfigyelési Információs Rendszer (FIR) kezelői felülete. A FIR felületén könnyedén letölthet bárki, olyan Sentinel műholdfelvételeket, melyek Magyarország területét érintik.



5. ábra: A Google Earth Engine felhasználói felülete

A vizsgálatokhoz Észak-Magyarországi régióban elhelyezkedő, elsősorban Szirák és Vanyarc települések környezetében megtalálható akácerdőkről készült felvételeket használtam fel. A választás azért került erre a területre, mert az itt található akácerdőket terepen is sikerült korábban beazonosítani.

3. Módszertan

3.1 Megfelelő műholdfelvételek kiválasztása

A feladat megkezdése előtt ki kellett választani a megfelelő felvételeket a vizsgált területről, azonban csak olyan felvételek jöhettek szóba, melyeken egyértelműen elkülöníthető a *Robinia pseudoacacia* az egyéb fafajoktól. Mivel a műholdfelvételek felbontása látható fény esetén is csak 10 m, így a faj felismerése nem feltétlen egyértelmű, ehhez kellett keresni egy olyan időszakot, amikor a növény egyedei egyértelműen megkülönböztethetőek. A választás a növény virágzására esett, mely mindig májusban történik és nagyobb akácerdőknél műholdfelvételen is kimutatható az erdők „kifehéredése”.

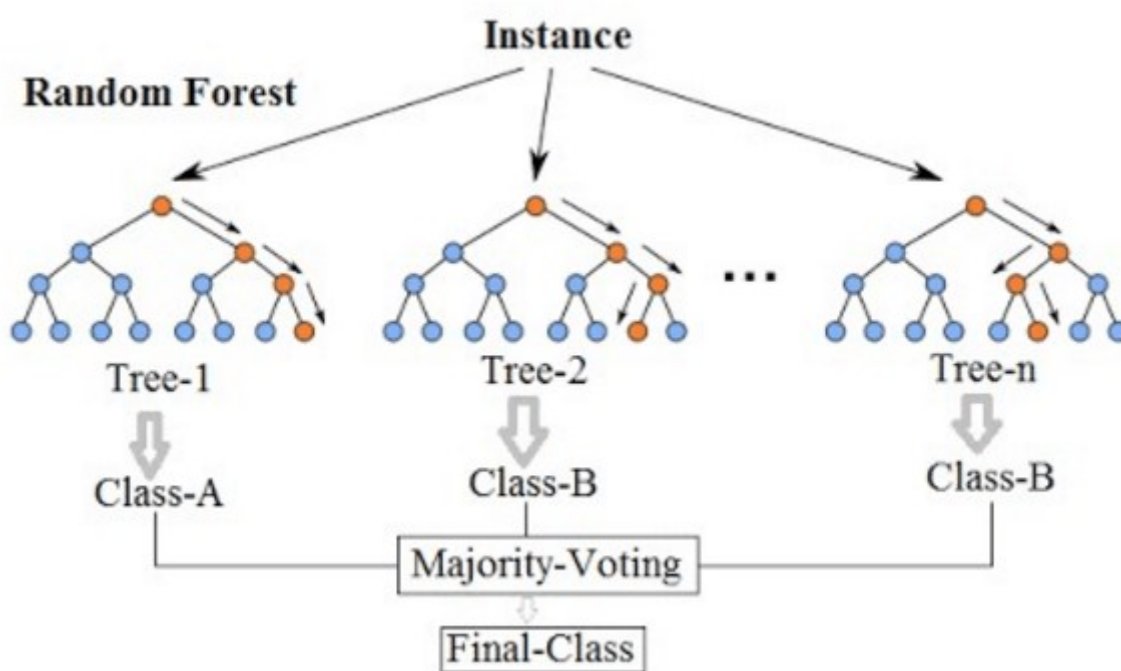
A kiválasztáshoz tisztában kellett lenni azzal, hogy a vizsgált területen pontosan mikor történt a *Robinia pseudoacacia* virágzása, melyhez segítségemre volt szüleim méhészete, mely évről évre az adott területen végzi a méhek legeltetését, melyről naplózást is készítenek. A feladatot nehezíti, hogy olyan felvételt kell kiválasztani, mely felhőmentes, továbbá azt is figyelembe kell venni, hogy a *Robinia pseudoacacia* 1 maximum 2 hétig virágzik évente, így kevés műholdfelvétel áll rendelkezésre. A választás, a 2020-as évi virágzásra esett, melyen sikerült teljesen felhőmentes felvételt találni a *Robinia pseudoacacia* virágzásáról.



6. ábra: *Robinia pseudoacacia* virágzása Szirák környékén 2020. május 22-én.

3.2 Az elemzés kezdeti szakasza

Az elemzés kezdeti szakaszában több elképzelés is született a lehetséges megoldásokra. A legelső elképzelés szerint a Random Forests Classification módszerrel szerettem volna véghez vinni a feladatot. A Random Forests, vagy magyar szakirodalomban véletlen erdő osztályozás, egy gépi tanuláson alapuló eljárás, lényegében döntési fák halmazaként írható le. A módszer alkalmazása során több döntési fa által leadott előrejelzéseket kombinálnak egy többségi szavazási séma segítségével. Mindegyik fa rögzített valószínűségi eloszlásból számított véletlen vektorok egy független halmazának értékei alapján jön létre [8].



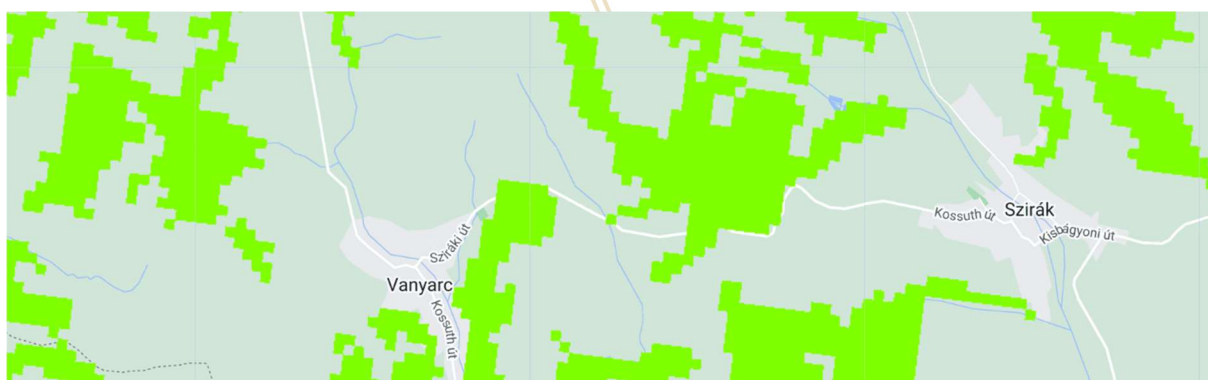
7. ábra: A Random Forest Classification működési elve

Végül a módszer hamar elvetésre került, mivel nem szerettem volna minden egyes objektumot egy-egy osztályba besorolni a műholdfelvételen, csak a *Robinia pseudoacacia* erdők leválogatása volt a cél. Így kilátásba került, valamilyen számszerűen kezelhető index megalkotása. Mivel egy műholdfelvétel nagy területet fed le, célszerűnek tűnt az adathalmaz szűkítése. Erre az elképzelésre megfelelőnek tűnt a CORINE felszínborítási adatbázisa [9]. A CORINE 1990 óta mutatja be, az Európai Unió területének felszínborítási változásait, legfrissebb adatai jelenleg a 2018-as felszínborítási térképen láthatóak. A felszínt 100x100 m-es parcellákra osztja fel, és mindegyik parcellát egy felszínborítási osztályba sorol be. Az adatbázis elérhető a Google Earth Engine szolgáltatásaiban is, így könnyedén hozzá tudtam rendelni az addigi munkámhoz.

A CORINE felszínborítási adatbázisában minden felszínborítási elemhez egy értéket rendelnek hozzá, mely a lombhullató erdők esetében a 311-es érték volt, mellyel az adatokat maszkolni lehet.

```
//Magyarországi lombhullató erdők leválogatása
var corine = ee.Image("COPERNICUS/CORINE/V20/100m/2018");
var forest = corine.updateMask(corine.eq(311));
```

8. ábra: A 2018-as CORINE felszínborítási adatok beimportálása a Google Earth Engine használatával. Az alsó sorban az adatbázis maszkolása látható, mellyel a lombhullató erdőkre végeztem a szűrést. A lombhullató erdők az adatbázisban a 311-es értéket veszik fel.



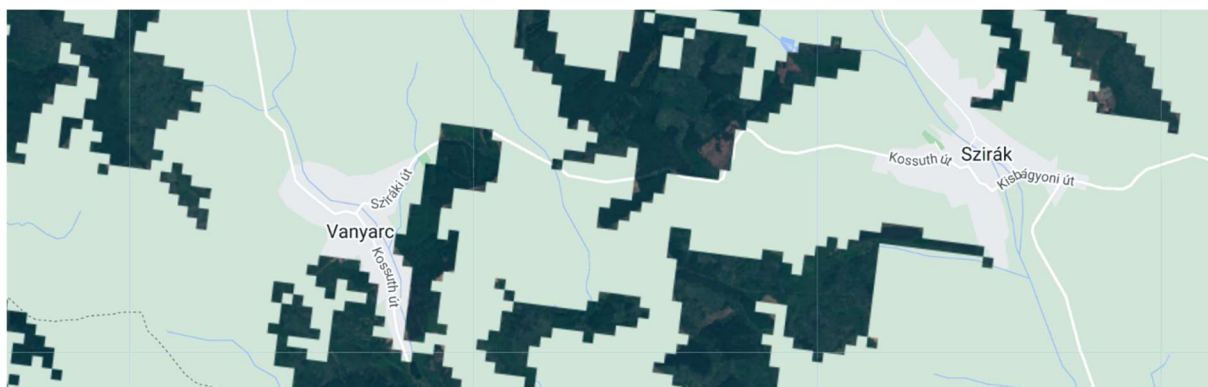
9. ábra: Maszkolás eredménye a Szirák és Vanyarc környezetében található lombhullató erdőkre.

Ezt követően a leszűrt lombhullató erdő felszínborítási adatait felhasználva kivágatot készíthetünk a kiválasztott Sentinel-2 műholdfelvételen. A Google Earth Engine egy raszteres állomány kivágását raszteres állománnyal azonban nem képes elvégezni, így először az erdőborítási adatokból vektoros adatot kellett készíteni.

```
// Erdőterületek vektorizálása
var forestVectors = forestAreas.reduceToVectors({
  geometry: StudAR,
  scale: 10,
  geometryType: 'polygon',
  eightConnected: false
});
```

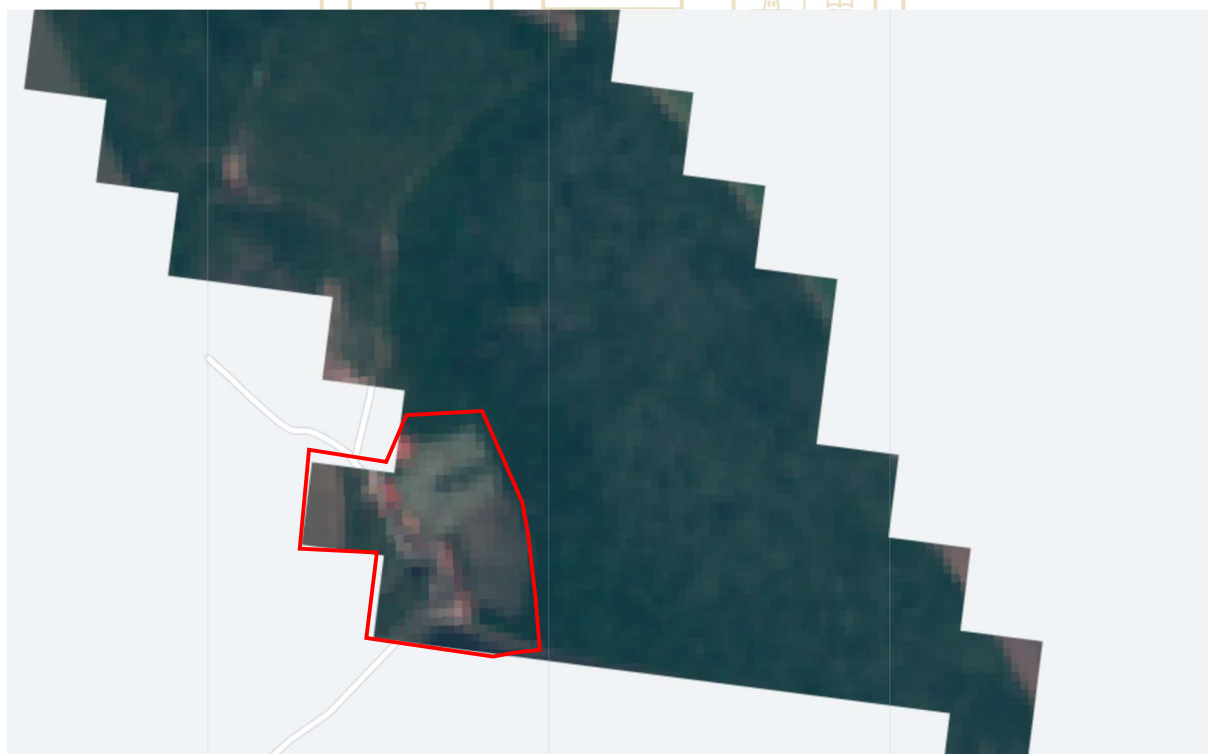
10. ábra: Leszűrt lombhullató erdők raszteres állományból vektoros állományba való konvertálása.

Az elkészült vektoros állományt felhasználva, már könnyedén elkészíthető a Sentinel-2 műholdfelvétel kivágata, melyen már csak a lombhullató erdők látszódnak.



11. ábra: A Sentinel-2 műholdfelvétel kivágatának elkészítése

Az elkészült kivágatokat azonban ha jobban megvizsgáljuk, felfedezhető rajtauk több hiba is, melyeket tovább tetéz, ha a kivágatot az eredeti képpel hasonlítjuk össze. A kivágaton számtalan helyen felfedezhető olyan parcella, melyet a CORINE lombhullató erdőnek sorol be, valójában pedig a területen szántó helyezkedik el. Az eredeti képpel összehasonlítva pedig fellelhetők olyan területek is, melyeken jól kivehető erdős területek helyezkednek el, azonban a felszínborítási adatbázis nem erdőterületként tünteti fel az érintett helyeket. Miután a hibákat felfedeztem, arra a következtetésre jutottam, hogy a CORINE adatbázisának felhasználása nélkül kell a feladatot megoldani.

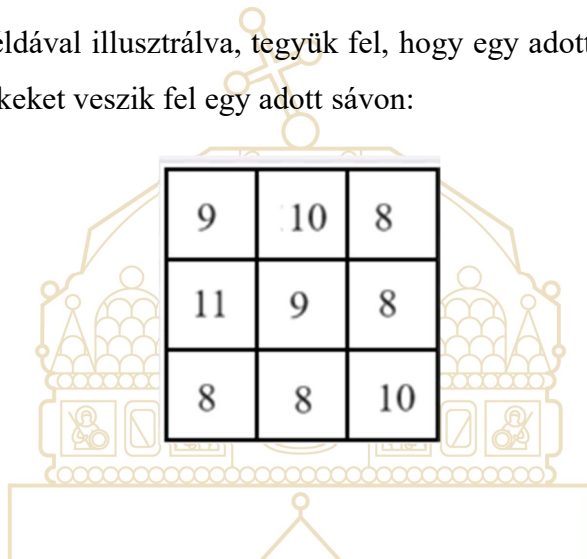


12. ábra: Példa a CORINE adatbázis hibás felszínborítási adataira. A kijelölt területeken Szirák szomszédságában fellelhető zártkerti rész, látható.

3.3 Szórásképek

Azt követően, hogy a kezdeti feltevések kizárásra kerültek, egyértelművé vált, hogy a vizsgálathoz érdemes lehet minden pixelt a szomszéd pixelekkel összehasonlítva vizsgálni. Erre megoldás a szórások előállítása sávonként. A szórás értékek előállításához 3x3 pixeles ablak került felhasználásra. Az ablakon belül található 9 pixelnek a kiválasztott sávokon kiszámolható a szórása majd a középső pixelen eltárolható az adott szórás értéke.

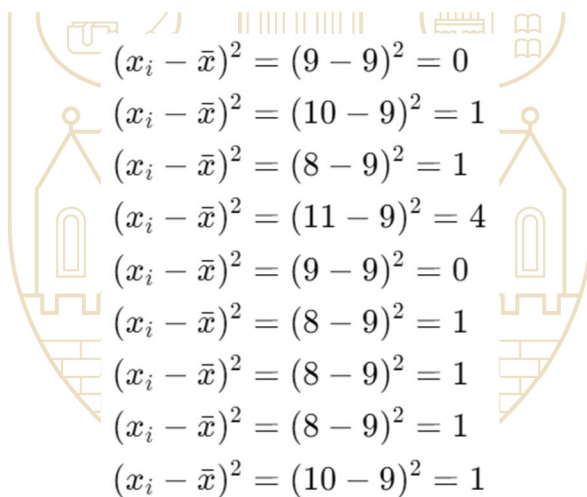
Egy egyszerű példával illusztrálva, tegyük fel, hogy egy adott ablakon belül a pixelek az alábbi intenzitás értékeket veszik fel egy adott sávon:



9	10	8
11	9	8
8	8	10

Számítás menete:

$$\bar{x} = \frac{9+10+8+11+9+8+8+8+10}{9} = \frac{81}{9} = 9$$



$$\begin{aligned} (x_i - \bar{x})^2 &= (9 - 9)^2 = 0 \\ (x_i - \bar{x})^2 &= (10 - 9)^2 = 1 \\ (x_i - \bar{x})^2 &= (8 - 9)^2 = 1 \\ (x_i - \bar{x})^2 &= (11 - 9)^2 = 4 \\ (x_i - \bar{x})^2 &= (9 - 9)^2 = 0 \\ (x_i - \bar{x})^2 &= (8 - 9)^2 = 1 \\ (x_i - \bar{x})^2 &= (8 - 9)^2 = 1 \\ (x_i - \bar{x})^2 &= (8 - 9)^2 = 1 \\ (x_i - \bar{x})^2 &= (10 - 9)^2 = 1 \end{aligned}$$

$$\sum_{i=1}^9 (x_i - \bar{x})^2 = 0 + 1 + 1 + 4 + 0 + 1 + 1 + 1 + 1 = 10$$

$$\sigma = \sqrt{\frac{10}{8}} \approx 1.05$$

Az így kapott szórás (σ) érték pedig a középső pixelen kerül eltárolásra. Ezt követően az ablak egy pixellel tovább lép, és a folyamatot addig ismétli, ameddig a felvétel minden pixelére el nem végezte a számítást.

A számítás megoldására a Google Earth Engine rendelkezik saját függvénnyel, így ennek leprogramozása nem okozott különösebb gondot.

```
var stdDevIm = image.reduceNeighborhood(ee.Reducer.stdDev(), ee.Kernel.square(1));
```

13. ábra: Szóráskép létrehozása

Ezt követően a vörös, zöld és kék sávok szórásértékeit visszahelyettesítettem egy a megjelenített kép RGB intenzitás értékeinek helyére és megjelenítettem a képet.

```
Map.addLayer(stdDevIm, {min: 0, max: 100, bands: ['B4_stdDev', 'B3_stdDev', 'B2_stdDev']}, 'StdDev');
```

14. ábra: Szóráskép megjelenítése a térképen



15. ábra: Az elkészült szóráskép. Kékkel bekarikázva a *Robinia pseudoacacia* erdők.

Az elkészült szórásképen már jobban beazonosítható az *Robinia pseudoacacia* erdő, azonban a kívánt végeredmény eléréséhez szükség van egy indexérték előállítására, mellyel a vizsgálni kívánt erdők könnyedén leszűrhetők.

Következő lépésként kivágatok készültek az akácerdők területeiről, majd az elkészült kivágatok exportálásra kerültek. Az exportálásra a Google Earth Engine használatával több lehetőség is van, azonban mindenképp szükséges a funkció leprogramozása. Ezt történhet úgy, hogy minden futtatáskor a rendszer elindítja az exportot, vagy leprogramozható egy gomb, mely a térképen megjelenik és lenyomásával triggerelődik az exportálás. Véleményem szerint érdemes az utóbbit választani.

```
// Exportgomb
var exportButton = ui.Button({
  label: 'Export',
  onClick: function() {
    Export.image.toDrive({
      image: akacDevClip,
      description: 'AkacDevClip_export',
      folder: 'GEE',
      scale: 10,
      fileFormat: 'GeoTIFF'
    });
  }
});
Map.add(exportButton);
```

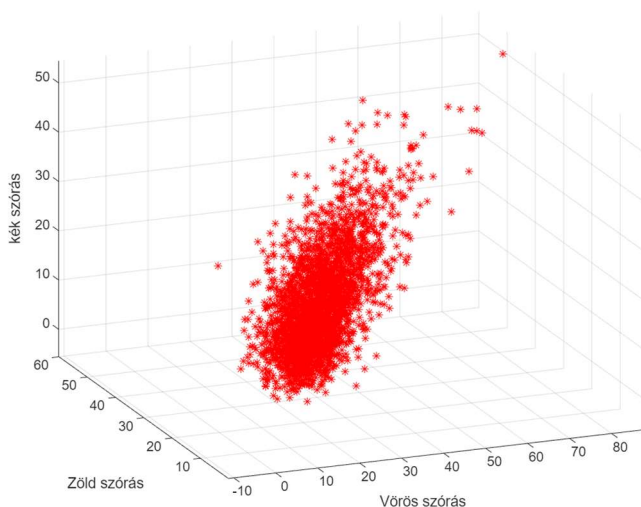
16. ábra: Exportgomb készítése és hozzárendelése a térképhez a Google Earth Engine felületén. Az exportálás GeoTIFF formátumban került végrehajtásra.

A Google Earth Engine az exportálást a Google Drive felületére végzi el, innen a felvétel letölthető. A letöltött állomány Matlab nevű szoftver segítségével került további elemzésre. GeoTIFF állomány Matlab-ba történő betöltéséhez szükség van a Matlab Mapping Toolbox nevű kiegészítőjére.

```
[A,R] = readgeoraster("AkacDevClip_export.tif");
```

17. ábra: GeoTIFF állomány Matlab szoftverbe való betöltése.

A kivágatok Matlab-ba történő behívásakor figyelembe kell venni, hogy a Matlab mátrixban tárolja el az adatokat, így ott ahol nincs pixel *NaN* (Not a Number) értéket fog behelyettesíteni, tehát ezen értékeket ki kell szűrni. Ezt követően a pontok halmazát egy 3 dimenziós derékszögű koordináta rendszerben lehet megjeleníteni.



18. ábra: A szórás kép pixelértékei derékszögű koordináta rendszerben megjelenítve.



Következő lépés a sávonkénti szórásértékek kovariancia mátrixának kiszámítása volt, mely az eredmények helyes kiértékelése érdekében a pixel intenzitásértékekkel is elvégzésre került. A kovariancia mátrix négyzetes (sorainak, és oszlopainak száma megegyezik) és szimmetrikus. A kovariancia mátrix a változók páronkénti együtt változás mértékét adja meg, főátlójában az egyes változók szórásnégyzetei szerepelnek [10].

Egy 3x3-as kovariancia mátrix számítása egyszerű példával levezetve:

Kiinduló adatok:

$$X = \begin{vmatrix} 2 & 4 & 6 \end{vmatrix} \quad Y = \begin{vmatrix} 3 & 6 & 9 \end{vmatrix} \quad Z = \begin{vmatrix} 1 & 2 & 3 \end{vmatrix}$$

1. lépés átlag számítás:

$$\bar{X} = \frac{2+4+6}{3} \quad \bar{Y} = \frac{3+6+9}{3} \quad \bar{Z} = \frac{1+2+3}{3}$$

2. lépés páronkénti kovariancia kiszámítása

$$Cov(X, X) = \frac{(2-4)^2 + (4-4)^2 + (6-4)^2}{3-1} = \frac{4+0+4}{2} = 4$$

$$Cov(X, Y) = \frac{(2-4)(3-6) + (4-4)(6-6) + (6-4)(9-6)}{3-1} = \frac{6+6}{2} = 6$$

$$Cov(X, Z) = \frac{(2-4)(1-2) + (4-4)(2-2) + (6-4)(3-2)}{3-1} = \frac{2+2}{2} = 2$$

$$Cov(Y, Y) = \frac{(3-6)^2 + (6-6)^2 + (9-6)^2}{3-1} = \frac{9+0+9}{2} = 9$$

$$Cov(Y, Z) = \frac{(3-6)(1-2) + (6-6)(2-2) + (9-6)(3-2)}{3-1} = \frac{3+3}{2} = 3$$

$$Cov(Z, Z) = \frac{(1-2)^2 + (2-2)^2 + (3-2)^2}{3-1} = \frac{1+0+1}{2} = 1$$

3. lépés mátrixos alakba rendezés

$$\bar{\bar{C}} = \begin{vmatrix} Cov(X, X) & Cov(X, Y) & Cov(X, Z) \\ Cov(X, Y) & Cov(Y, Y) & Cov(Y, Z) \\ Cov(X, Z) & Cov(Y, Z) & Cov(Z, Z) \end{vmatrix}$$

$$\overline{C} = \begin{bmatrix} 4 & 6 & 2 \\ 6 & 9 & 3 \\ 2 & 3 & 1 \end{bmatrix}$$

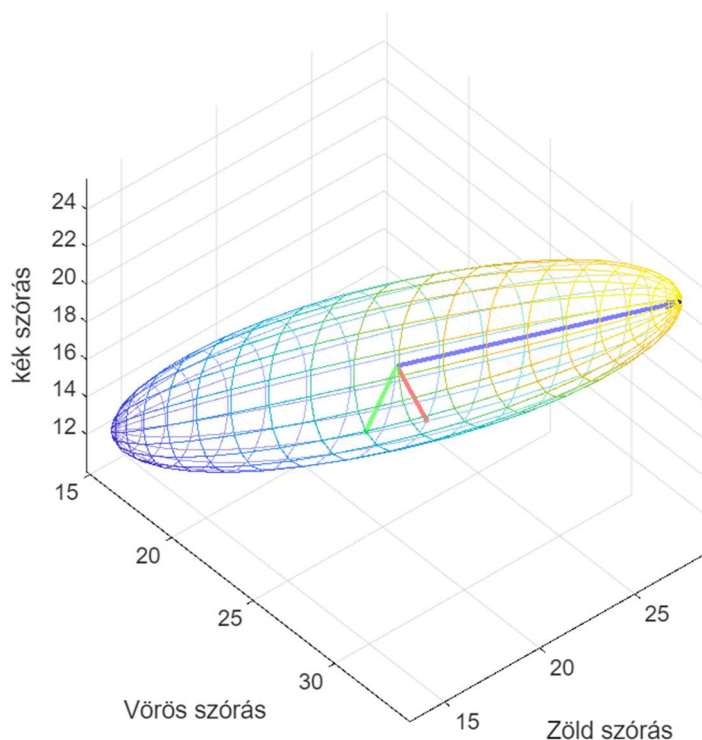
A kovariancia mátrix kiszámítására, a Matlab és a Google Earth Engine rendelkezik saját függvénnyel, így mind a két programmal a feladat könnyedén végrehajtható.

```
var cov = akacClip.toArray().reduceRegion({
  reducer: ee.Reducer.covariance(),
  scale: 10
});
```

```
X = [2 4 6]';
Y = [3 6 9]';
Z = [1 2 3]';
XYZ = [X, Y, Z];
C = cov(XYZ);
```

19. ábra: Kovariancia mátrix kiszámítása Google Earth Engine (bal oldalon), illetve Matlab (jobb oldalon) segítségével.

A kovariancia mátrix sajátértéke és sajátvektora meghatározható, melyekre a hibaellipszoid meghatározása miatt volt szükség [11].



20. ábra: A szóráskép hibaellipszoidjának kirajzoltatása Matlab segítségével.

A műholdfelvételen szereplő pontok indexelésére kézenfekvőnek tűnt a pixelek szórásértékeinek, valamint intenzitásértékeinek hibaellipszoidtól mért távolságát felhasználni. A hibaellipszoidtól mért távolság meghatározására a legmegfelelőbb megoldás a pontok Mahalanobis távolságának meghatározása volt.

3.4 Mahalanobis távolság

A Mahalanobis távolságot egy indiai matematikus, Prasanta Chandra Mahalanobis fedezte fel 1936-ban melyet eredetileg az indiai kasztok közötti távolság mérésére talált ki [12]. A Mahalanobis távolságot számtalan helyen felhasználják, a régészettől az orvosi diagnosztikán át, egészen a távérzékelésig. A Mahalanobis távolság számítása egy statisztikai módszer, melyet több változó közötti távolság mérésére használnak fel, figyelembe véve a változók közötti korrelációt. A Mahalanobis távolságot lehet úgyis értelmezni, hogy a vizsgált pontok hibaellipsziszét (ellipszoidját) visszanyomjuk egy körre (gömbbé), ezáltal az összes többi pont helyzete is megváltozik, és a távolság azt adja vissza, hogy hány kör (gömb) sugárnyi az adott pont távolsága a vizsgált pontok átlagától, mely a kör (gömb) középpontját jelenti.

A Mahalanobis távolság számítás során a *Robinia pseudoacacia* területek tanulóterületekként voltak felhasználva. A számítás során az eredeti kép sávonkénti intenzitás értékeiből kivonásra kerültek a tanulóterület sávonkénti intenzitás átlagai, melyek egy oszlopvektorban kerültek eltárolásra. Következő lépés a tanulóterületeken található pixel intenzitás értékekre történő kovariancia mátrix számítás volt.

A Mahalanobis távolság implementálásakor az oszlopvektor transzponáltjával beszoroztam a kovariancia mátrix inverzét bal oldalról, jobb oldalról pedig az oszlopvektorral szoroztam meg.

$$D = \sqrt{|X - \bar{X}|^T * \bar{C}^{-1} * |X - \bar{X}|}$$

21. ábra Mahalanobis távolság számítása

A számítás reprezentálható egy egyszerű példa segítségével. A műholdfelvételen egy P pont kiválasztása után, melynek intenzitás értékei látható fénytartományban R_P , G_P , B_P , az alábbi módon számítható ki a Mahalanobis távolság:

$$D^2 = \begin{bmatrix} R_P - \bar{R} & G_P - \bar{G} & B_P - \bar{B} \end{bmatrix} * \begin{bmatrix} C_{1,1} & C_{1,2} & C_{1,3} \\ C_{2,1} & C_{2,2} & C_{2,3} \\ C_{3,1} & C_{3,2} & C_{3,3} \end{bmatrix}^{-1} * \begin{bmatrix} R_P - \bar{R} \\ G_P - \bar{G} \\ B_P - \bar{B} \end{bmatrix}$$

22. ábra: Mahalanobis távolság négyzetének kiszámítása RGB képen, egy P pont és a tanulóterület között.

A számítás Matlab környezetben történő kiszámítása nem okoz különösebb problémát, hiszen könnyedén kezeli a mátrix műveleteket, azonban a Google Earth Engine-be való

implementálása összetettebb feladat. A probléma forrása, hogy a Google Earth Engine több objektum típust különböztet meg, melyekkel különböző műveletek hajthatók végre. A rendszer különbséget tesz *Image*, *ImageCollection*, *Feature*, *FeatureCollection*, *Geometry*, *Reducer* és *Array* között. A különböző objektum típusok között van ugyan valamekkora átjárás, ám ez nem korlátlan. A mátrix műveletek, mint például a transzponálás, mátrix inverz, és mátrix szorzás csak *Array* objektumokra hajtható végre, a feladat végrehajtásához azonban *Image* típusú objektumokat kell alkalmaznunk.

A kovariancia mátrix számítása során a Google Earth Engine, *Reducer* típusú objektumot ad vissza, mely a "toArray" paranccsal konvertálható *Array* típusú objektummá, mellyel már elvégezhető a mátrix inverz kiszámítása.

```
var cov = akacClip.toArray().reduceRegion({  
  reducer: ee.Reducer.covariance(),  
  scale: 10  
});  
  
var covA = cov.toArray();  
  
var invCov = covA.matrixInverse();
```

23. ábra Kovariancia mátrix kiszámítása a tanulóterületre, majd *Array* típusú objektummá konvertálása, mellyel a mátrix inverzének kiszámítása elvégezhető.

Mivel a Google Earth Engine környezetében *Image* típusú objektumok csak *Image* típusú objektumokkal tudnak műveleteket végrehajtani, így a mátrix elemeiből konstansképeket kellett létrehozni. Minden egyes konstans kép a mátrix egy-egy elemét tartalmazza.

```
var cons00 = ee.Image.constant(invCov.get([0,0]));  
var cons01 = ee.Image.constant(invCov.get([0,1]));  
var cons02 = ee.Image.constant(invCov.get([0,2]));  
var cons10 = ee.Image.constant(invCov.get([1,0]));  
var cons11 = ee.Image.constant(invCov.get([1,1]));  
var cons12 = ee.Image.constant(invCov.get([1,2]));  
var cons20 = ee.Image.constant(invCov.get([2,0]));  
var cons21 = ee.Image.constant(invCov.get([2,1]));  
var cons22 = ee.Image.constant(invCov.get([2,2]));
```

24. ábra: Az inverz kovariancia mátrix elemei konstans képekké konvertálása.

A tanulóterületek átlag intenzitás értékeinek a kiszámítása során szintén *Reducer* típusú objektumot adott vissza a függvény. Mivel azonban ebben az esetben sávonként voltak megnevezve az átlagok, így a *Reducer* objektumból kiemelve az értékeket létrehozható a

konstanskép a sávonkénti átlagokkal. A konstanskép sávjai ezután az eredeti kép sávjaihoz lettek hozzárendelve.

```
var mean = akacClip.reduceRegion({  
  reducer: ee.Reducer.mean(),  
  scale: 10  
});  
  
var meanValues = mean.values();  
  
var meanIm = ee.Image.constant(meanValues);  
var image_mean = image.addBands(meanIm);  
  
var array = ee.Array([1,9]);
```

25. ábra: Sávonkénti átlagok kiszámítása, majd az értékek segítségével konstans kép létrehozása. A konstanskép sávjai végül az eredeti képhez kerültek hozzárendelésre.

A Google Earth Engine használata során a legtöbb *Reducer* alkalmazása során a sávok indexelését nem írja felül a rendszer, azonban érdemes megjegyezni, hogy az átlag kiszámításakor igen. Ezt a felhasználónak úgy kell értenie, hogy amikor létrehoz egy képet, akkor beállít a felvételhez egy bizonyos sáv sorrendet, például Sentinel-2 sávokból valós színes felvétel készítése során B4 (vörös), B3 (zöld), és B2 (kék) kombinációt kell alkalmazni. Amikor egy *Reducer* futtatásra kerül, akkor a legtöbb esetben ez a sorrend nem változik, tehát B4, B3, B2 sorrend marad a sávok indexelésénél.

Az átlag számítása során azonban a *Reducer* az indexelést felülírja, és ABC sorrendbe helyezi el a sávokat, így az eredmény a valós színes felvétel esetén B2, B3, B4 kombináció lesz. Jelentősége csak akkor lesz a felcserélt sorrendnek, ha a felhasználó matematikai műveleteket hajt végre a sávokkal, és azt nem a sáv nevére, hanem a sáv indexére hivatkozva teszi meg.

Az újonnan létrehozott felvételen, melyet az átlagintenzitásokkal alkotott konstanskép és az eredeti kép összeillesztésével kap a felhasználó az alábbi sáv indexeléssel lesznek elérhetőek a különböző sávok valós színes felvétel esetén:

- b(0) index: B4 (vörös) sáv
- b(1) index: B3 (zöld) sáv
- b(2) index: B2 (kék) sáv
- b(3) index: B2_mean (a tanulóterület kék átlaga) sáv
- b(4) index: B3_mean (a tanulóterület zöld átlaga) sáv
- b(5) index: B4_mean (a tanulóterület vörös átlaga) sáv

3.4.1 Matematikai műveletek végzése képeken

A Google Earth Engine több módot is biztosít a felhasználó számára matematikai műveletek végzésére, mivel azonban Cloud alapú, BigData környezetről van szó a matematikai szimbólumok, *for* ciklusok, vagy *if* elágazások használata kerülendő, mivel nagymértékben lassítja a programok futását. Erre a program is felhívja a felhasználó figyelmét, helyettük külön függvények állnak rendelkezésre.

Néhány példa a matematikai műveletek Google Earth Engine környezetben képekre történő használatára:

- Összeadás: *kép1.add(kép2)*
- Kivonás: *kép1.subtract(kép2)*
- Szorzás: *kép1.multiply(kép2)*
- Osztás: *kép1.divide(kép2)*
- Gyökvonás: *kép.sqrt()*

Létezik egyéb mód is a Google Earth Engine környezetében matematikai műveletek végzésére, például az *expression* függvény, melynek segítségével a matematikai kifejezést tudjuk elkészíteni. Az *expression* használatával a felhasználó már használhat matematikai műveleti jeleket, ugyanis a függvény lefordítja a matematikai jeleket a rendszer számára könnyedén értelmezhető formára, anélkül, hogy a program futása lelassulna.

A sávok egymásból történő kivonása során az *expression* függvény került alkalmazásra az alábbi módon:

```
var expR = 'b(0)-b(5)';  
var XR = image_mean.expression(expR);  
var expG = 'b(1)-b(4)';  
var XG = image_mean.expression(expG);  
var expB = 'b(2)-b(3)';  
var XB = image_mean.expression(expB);
```

26. ábra: Az *expression* függvény alkalmazása

A 26. ábrán látható kódrészlet a felvétel minden pixelére lefut. A futás során a pixeleken eltárolt sávok indexeit felhasználva végzi el a program kivonásokat. A kivonás eredményeit tartalmazó *XR*, *XG* és *XB* változók *Image* objektumként kerülnek eltárolásra, így a későbbi műveletekhez felhasználhatóak.



A dolgozat elkészítése során több vizsgálat is készült, így volt olyan verzió, melyben a műholdfelvételen található összes sáv bekerült a számításba.

```
var expB1 = 'b(0)-b(13)';  
var XB1 = image_mean.expression(expB1);  
var expB2 = 'b(1)-b(17)';  
var XB2 = image_mean.expression(expB2);  
var expB3 = 'b(2)-b(18)';  
var XB3 = image_mean.expression(expB3);  
var expB4 = 'b(3)-b(19)';  
var XB4 = image_mean.expression(expB4);  
var expB5 = 'b(4)-b(20)';  
var XB5 = image_mean.expression(expB5);  
var expB6 = 'b(5)-b(21)';  
var XB6 = image_mean.expression(expB6);  
var expB7 = 'b(6)-b(22)';  
var XB7 = image_mean.expression(expB7);  
var expB8 = 'b(7)-b(23)';  
var XB8 = image_mean.expression(expB8);  
var expB8A = 'b(8)-b(24)';  
var XB8A = image_mean.expression(expB8A);  
var expB9 = 'b(9)-b(25)';  
var XB9 = image_mean.expression(expB9);  
var expB10 = 'b(10)-b(14)';  
var XB10 = image_mean.expression(expB10);  
var expB11 = 'b(11)-b(15)';  
var XB11 = image_mean.expression(expB11);  
var expB12 = 'b(12)-b(16)';  
var XB12 = image_mean.expression(expB12);
```

27. ábra: A műholdfelvétel pixelén tárolt intenzitás értékek és intenzitás átlagértékek közötti különbség számítása mind a 13 sávon. A kód elkészítése során nagy figyelmet kellett a sávok indexeire fordítani a mean függvény újraindexelése miatt.

Miután a sávok intenzitás értékei, és a tanulóterületek intenzitás értékei átlagának különbsége rendelkezésre állt, a Mahalanobis távolságba való behelyettesítés következett. Mivel a mátrix műveletek azonban a számítás kivitelezéséhez nem álltak rendelkezésre a számítást le kellett vezetni.

A levezetés reprezentálásához három sávos RGB felvétel kerül felhasználásra. A levezetésben az ΔR a vörös intenzitás érték és átlag különbséget jelöli, a ΔG a zöld sáv intenzitás érték és átlag különbségét jelöli, a ΔB pedig a kék sáv intenzitás érték és átlag különbségét jelöli. Az IC a kovarianciamátrix inverzét jelöli, az alsóindexben található számok közül az első a sort, a második az oszlopot jelöli.

$$D^2 = \begin{bmatrix} \Delta R & \Delta G & \Delta B \end{bmatrix} * \begin{bmatrix} IC_{1,1} & IC_{1,2} & IC_{1,3} \\ IC_{2,1} & IC_{2,2} & IC_{2,3} \\ IC_{3,1} & IC_{3,2} & IC_{3,3} \end{bmatrix} * \begin{bmatrix} \Delta R \\ \Delta G \\ \Delta B \end{bmatrix}$$

Az átláthatóság érdekében a levezetés három elemre (E1, E2, E3) tagolva történik.

$$E1 = (\Delta R * IC_{1,1} + \Delta G * IC_{2,1} + \Delta B * IC_{3,1}) * \Delta R$$

$$E2 = (\Delta R * IC_{1,2} + \Delta G * IC_{2,2} + \Delta B * IC_{3,2}) * \Delta G$$

$$E3 = (\Delta R * IC_{1,3} + \Delta G * IC_{2,3} + \Delta B * IC_{3,3}) * \Delta B$$

$$D = \sqrt{E1 + E2 + E3}$$

28. ábra: A Mahalanobis távolság elemekre történő lebontása. A Mahalanobis távolság jelölése D

A Google Earth Engine környezetébe így a számolás már könnyedén implementálható. A számítás elvégzését követően egy képet kapunk eredményként, melyen egy sáv található, mely a Mahalanobis távolság értékeit tartalmazza. Érdeemes a számítás után a sávot is átnevezni, mert a program a számításba bekerülő sávok közül indexelés szerinti első sáv nevével látja el a számított sávunkat.

```
var elem1 = (XR.multiply(cons00).add(XG.multiply(cons01)).add(XB.multiply(cons02))).multiply(XR);  
var elem2 = (XR.multiply(cons10).add(XG.multiply(cons11)).add(XB.multiply(cons12))).multiply(XG);  
var elem3 = (XR.multiply(cons20).add(XG.multiply(cons21)).add(XB.multiply(cons22))).multiply(XB);  
  
var elemek = elem1.add(elem2.add(elem3));  
  
var mahal = elemek.sqrt().rename(['Mahal']);
```

29. ábra: A Mahalanobis távolság levezetése Google Earth Engine használatával

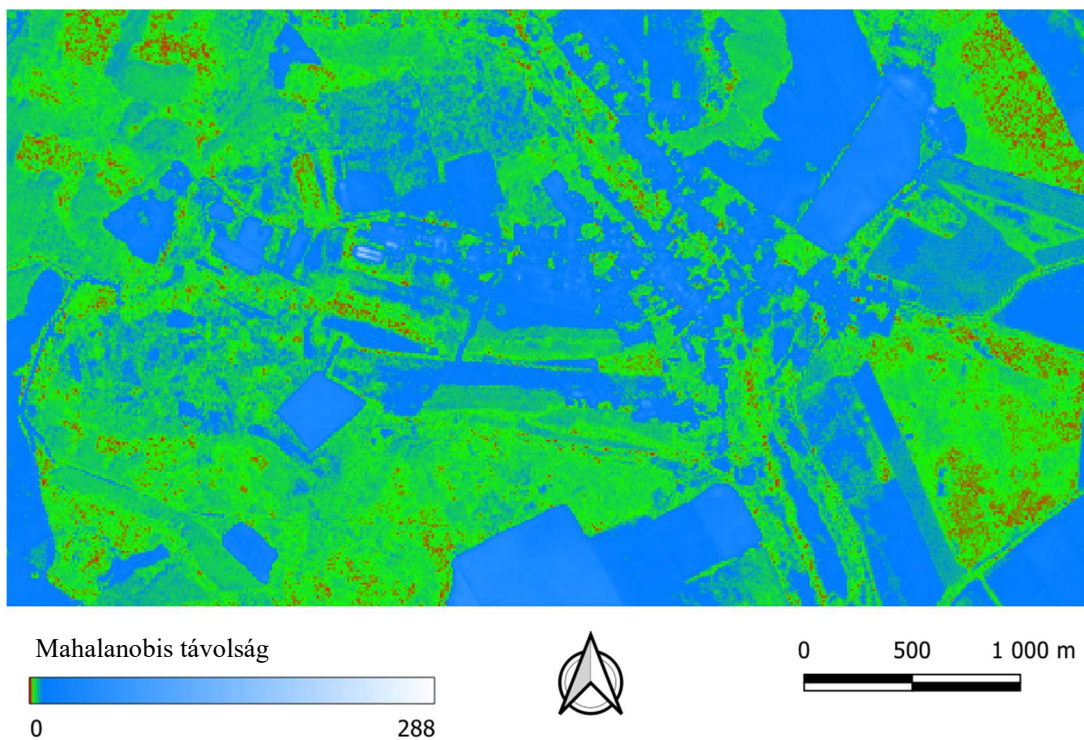


3.4.2 A Mahalanobis távolság számításával kapott eredmények

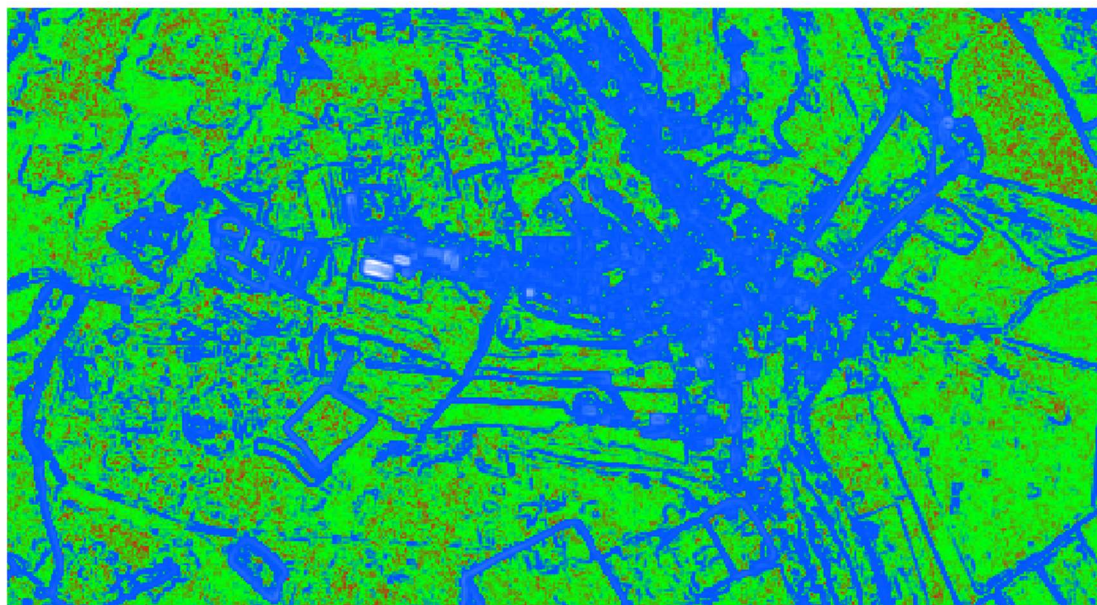
A vizsgálat folyamán 4 különböző verzió készült el a Mahalanobis távolság felhasználásával.

- Látható fénytartomány sávjaival készült elemzés
- Látható fénytartomány sávjaival készült szórás kép elemzése
- Mind a 13 sáv felhasználásával készült elemzés
- Mind a 13 sáv felhasználásával készült szóráskép elemzése

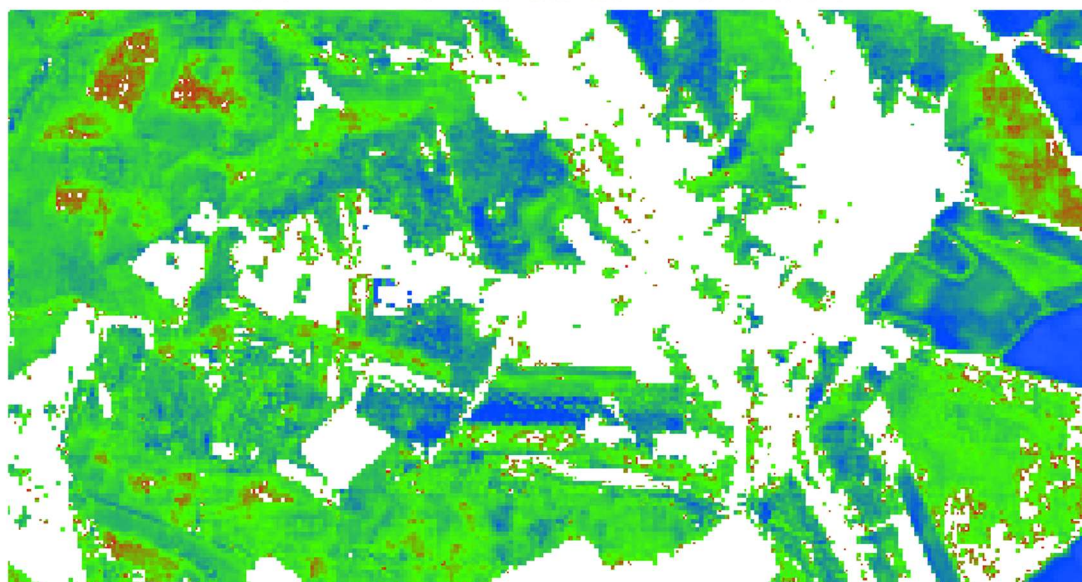
Mindegyik vizsgálat elkészülte után a tanulóterületeken egy *Reducer* megvizsgálta a tanulóterületek minimum, maximum, medián és átlag Mahalanobis távolságát. Az eredmények vizualizálásakor küszöbértékek kerültek elhelyezésre. A vizualizáláskor a vörös értékek a legvalószínűbb akácerdőket jelölik.



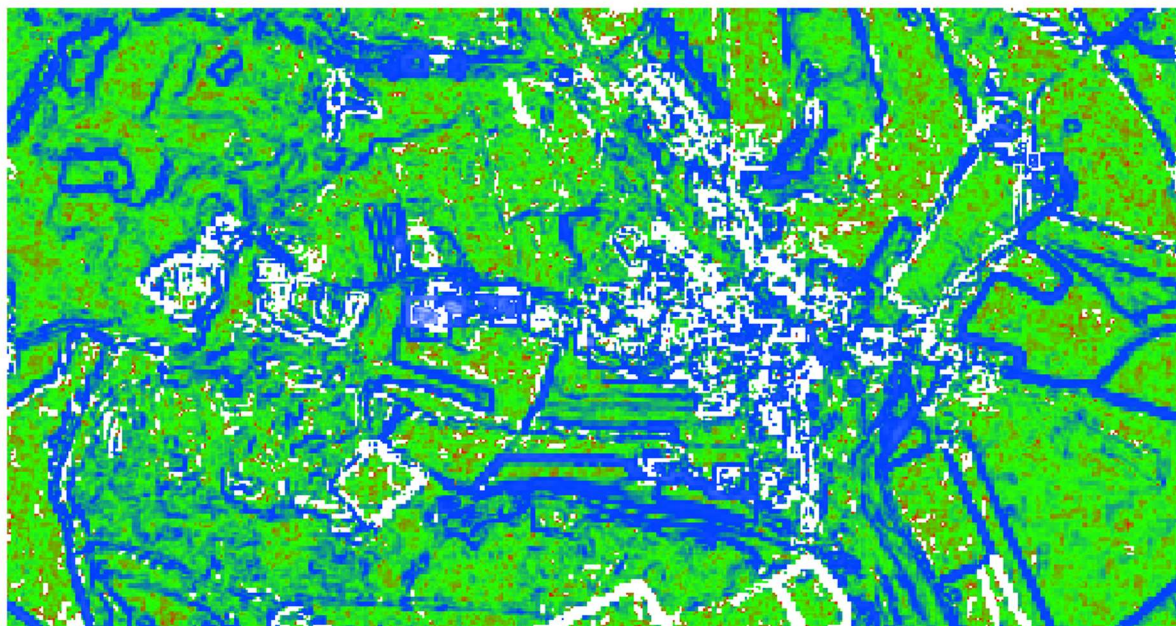
30. ábra: Látható fénytartományra készült Mahalanobis távolság számítása. A vörös foltok 0-2 szórás távolságra lévő értékek, melyek az akácerdőket reprezentálják.



31. ábra: Látható fénytartomány sávjaira készült szóráskép Mahalanobis távolsága. A vörös foltok 0-2 szórástávolság tartományát reprezentálják



32. ábra: 13 sávra számított Mahalanobis távolság. A vörös foltok 0-10 szórás távolságot jelölnek.



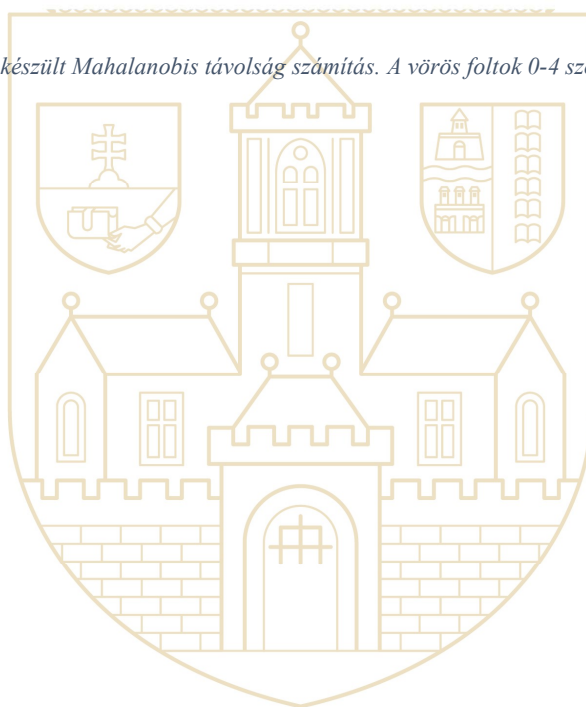
Mahalanobis távolság



0 500 1 000 m



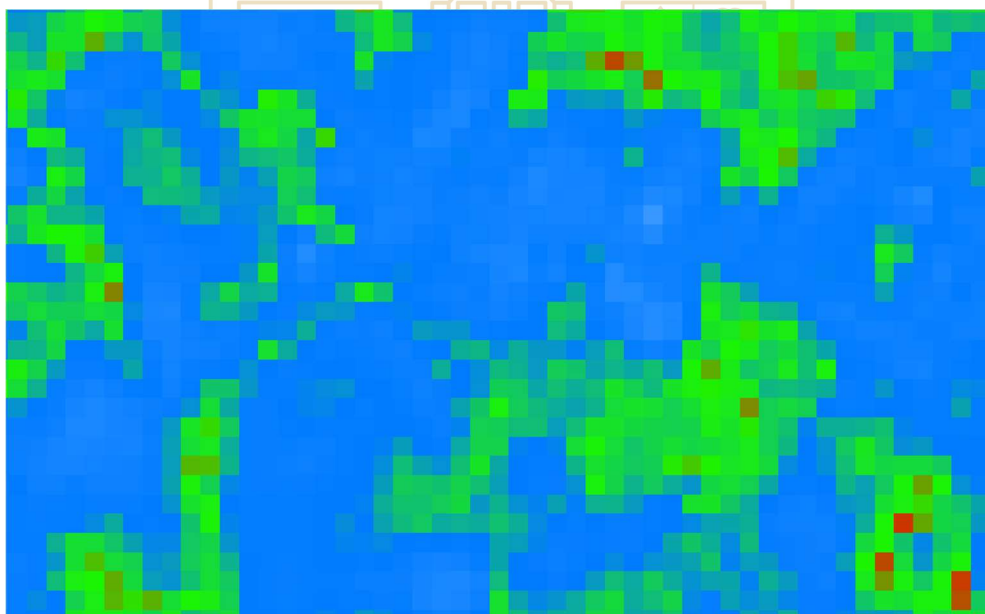
33. ábra: 13 sávós szórásképre készült Mahalanobis távolság számítás. A vörös foltok 0-4 szórás távolságot jelenítenek meg.



4. Eredmények kiértékelése és Diszkusszió

A kapott eredményeket figyelembe véve azt a következtetést lehet levonni, hogy a *Robinia pseudoacacia* műholdfelvételen történő detektálására megfelelő módszert biztosíthat a Mahalanobis távolság számítása. Érdekes megjegyezni, hogy bár az eredeti intenzitásértékekkel rendelkező képeknél kevésbé voltak beazonosíthatók a *Robinia pseudoacacia* erdők mint a szórásképek esetében, a Mahalanobis távolság számítását követően egyértelműbb és megbízhatóbb eredményeket produkáltak, szemben a szórásképek eredményeivel. Továbbá azt is érdemes megjegyezni, hogy több sáv használata sem eredményezett jobb eredményeket, mint a látható tartományokkal készült számítás esetében.

Fontos megjegyezni, hogy a módszer nem tökéletes. A legmegbízhatóbbnak mutakozó kiértékelés, mely során a látható fénytartomány eredeti intenzitás értékei kerültek felhasználásra, is rendelkezik hibás eredményekkel. Szórványosan, de előfordulnak a készült eredmények „hamis” akácerdő foltok, melyek valójában nem azok. A módszer továbbfejlesztésében érdemes lehet a Mahalanobis távolság számításának robosztus verzióját felhasználni, mellyel pontosabb eredmények érhetőek el [13].

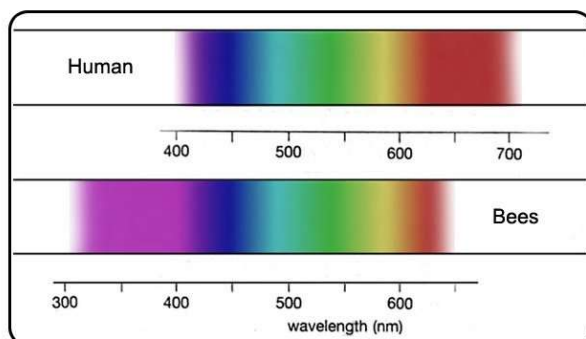


34. ábra: Az eredményen elszórtan fellelhető "hamis" akácerdő foltok.

A vizsgálat során az is megállapítható, hogy a *Robinia pseudoacacia* detektálására a legalkalmasabb időpont a növény virágzásának időpontja. Erre megállapításra jutottak egy Szlovéniai kutatásban is, melyben szintén a *Robinia pseudoacacia* távérzékelés detektálása volt

a cél, azonban ők a műholdfelvételek mellett légifelvételeket is alkalmaztak, valamint a módszertanuk is eltérő volt [14].

Érdekes gondolat lenne tovább, a méhek látásának reprodukálása, mellyel könnyebb lehet a méhlegelők detektálása. A méhek látható fénytartománya, 300 nm-es elektromágneses hullámhossztól 600 nm-es elektromágneses hullámhosszig terjed, szemben az ember látható fénytartományával, mely 400 nm-es elektromágneses hullámhossztól 700 nm-es elektromágneses hullámhosszig terjed. Sajnos azonban a Sentinel-2 műholdfelvételek látható fénytartománynál kisebb hullámhossztartományba eső sávokat nem tartalmaznak [15].



35. ábra: Az emberi, látható fénytartománya a méhek látható fénytartományával összehasonlítva





5. Konklúzió

A *Robinia pseudoacacia* Észak- és Közép Amerikában őshonos növényfaj, mely a 18. században került betelepítésre Magyarországon. Mindazonáltal, hogy jelentős gazdasági szerepeket tölt be, például méhlegelőként vagy tüzelőanyagként, rendkívül invazív. Területhódításának nyomon követése környezetvédelmi szempontból is kiemelhető, de a méhészeti ágazatban is hasznos információkat szolgáltatathat.

A faj műholdas földmegfigyelésen alapuló detektálása költséghatékony módszert jelenthet, azonban a feladat kivitelezése nem triviális, ugyanis a műholdfelvételeken nem egyértelműen beazonosítható. A leghatékonyabb detektálásra a növény virágzásakor van lehetőség, azonban a feladatot nehezíti, hogy nem áll rendelkezésünkre műholdfelvétel naponta, valamint a virágzás egy, maximum két hétig tart.

Módszertanilag a Mahalanobis távolság alkalmazása megfelelő lehet a feladat végrehajtására, de az eddigiekben kidolgozott formula még tökéletesítést igényel, mert csupán elszórtan, de tartalmaz „hamis” eredményeket.

A fejlesztésre megfelelő környezetet biztosít a Google Earth Engine fejlesztői környezete, mely rendelkezik különböző mátrix műveletekkel is, azonban a képekkel való végrehajtása nem minden esetben kivitelezhető, így a fejlesztőnek szükséges a mátrixműveleteket ismernie.

Érdekes gondolatokat vethet fel továbbá, hogy a rendszer hogyan átültethető, különböző méhlegelők vizsgálatára, valamint, hogyan lehetne gépi tanulás módszereit alkalmazva továbbfejleszteni és automatizálni.



Irodalomjegyzék

- [1] D. Bartha, Á. Csiszár, G. Zagyvai és V. Zsigmond, „Fehér akác (*Robinia pseudoacacia*),” in *Inváziós növényfajok Magyarországon*, Sopron, Nyugat-magyarországi Egyetem Kiadó, 2021, pp. 126-131.
- [2] E. Zilahy, „A méhészet, méztermelés helyzete és lehetőségei, különös tekintettel Észak-Magyarország megyéire,” Központi Statisztikai Hivatal, Budapest, 2012.
- [3] K. Ali és B. Johnson, „Land-Use and Land-Cover Classification in Semi-Arid Areas from Medium-Resolution Remote-Sensing Imagery: A Deep Learning Approach,” *Sensors*, 12 11 2022.
- [4] F. Spoto, O. Sy, P. Laberinti, P. Martimort, V. Fernandez, O. Colin, B. Hoersch és A. Meygret, „Overview Of Sentinel-2,” in *2012 IEEE International Geoscience and Remote Sensing Symposium*, 2012.
- [5] „Copernicus Open Access Hub,” European Space Agency, [Online]. Available: <https://scihub.copernicus.eu/>.
- [6] „Földmegfigyelési Információs Rendszer,” Lechner Tudásközpont, [Online]. Available: <https://fir.gov.hu/>.
- [7] N. Gorelick, M. Hancher, M. Dixon, S. Ilyushchenko, D. Thau és R. Moore, „Google Earth Engine: Planetary-scale geospatial analysis for everyone,” *Remote Sensing of Environment*, p. 18, 2017.
- [8] A. Gudmann és L. Mucsi, „Döntési fa és véletlen erdő osztályozási módszerekkel készített felszínborítási térképek pontosságának összehasonlító elemzése,” in *Debreceni Egyetem*, Debrecen, 2019.



- [9] G. Büttner, „CORINE Land Cover and Land Cover Change Products,” in *Land Use and Land Cover Mapping in Europe*, Dordrecht, Springer Netherlands, 2014, p. 55.
- [10] N. P. Dr. Szabó, „Bevezetés a geostatisztikába,” Miskolci Egyetem, Miskolc, 2012.
- [11] Á. Dr. Detrekői és G. Dr. Tóth, Kiegészítő számítások MSc, Budapest: Budapesti Műszaki és Gazdaságtudományi Egyetem, 2012.
- [12] G. J. McLachlan, „Mahalanobis Distance,” *Resonance*, pp. 20-26, 1999.
- [13] B. Magyar, A. Kenyeres, S. Tóth, I. Hajdú és R. Horváth, „Spatial outlier detection on discrete GNSS velocity fields using robust Mahalanobis-distance-based unsupervised classification,” *GPS Solutions*, 2022.
- [14] I. Somodi, A. Čarni, D. Ribeiro és T. Podobnikar, „Recognition of the invasive species *Robinia pseudacacia* from combined remote sensing and GIS sources,” *Biological Conservation*, p. 59, 2012.
- [15] S. Riddle, „How bees see and why it matters.,” *Bee Culture: The Magazine of American Bee Keeping*, 2016.



Köszönetnyilvánítás

TDK dolgozatom a Kulturális és Innovációs Minisztérium ÚNKP-23-2-II-OE-29 kódszámú Új Nemzeti Kiválóság Programjának a Nemzeti Kutatási, Fejlesztési és Innovációs Alapból finanszírozott szakmai támogatásával készült.

Szeretném elsősorban megköszönni konzulensem Dr. habil. Molnár Gábor Péter szakmai támogatását és tanácsadását, valamint, hogy mindig készséggel válaszolt felmerülő problémáimra. Szeretném továbbá megköszönni édesapámnak, Böröcz Istvánnak, a méhlegelőkről tartott eszmecseréit, valamint, hogy segített a műholdfelvételeken és terepen beazonosítani az akácerdők helyzeteit.

Köszönet illeti továbbá Gábor Dániel és Darázs Balázs gyerekkori cimboráimat amiért készséggel végighallgatták a kutatásomról szóló apróbb eszmecseréket. Legvégül nagy köszönet illeti menyasszonyomat, dr. Albert Annát, amiért minden este hajlandó volt a kutatásomról szóló beszámolóim meghallgatására.

