ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Alba Regia Műszaki Kar
Természettudományi és Szoftvertechnológiai Intézet

ZÁRÓDOLGOZAT

MŰHOLDAS ADATOK FELDOLGOZÁSA NYÍLT FORRÁSKÓDÚ ESZKÖZÖKKEL

OE-AMK

2023

Hallgató neve: Katona Zsolt

Hallgató törzskönyvi száma: T000355/FI12904/A-G

ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Óbudai Egyetem
Alba Regia Műszaki Kar
Geoinformatikai Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve: Katona Zsolt Sándor
Szakdolgozat száma: [redacted]
Törzskönyvi száma: T000355/FI12904/A-G **Neptun kódja:** [redacted]

Szak: geoinformatikai szakmérnök
Specializáció:

A dolgozat címe: Műholdas adatok feldolgozása nyílt forráskódú eszközökkel
A dolgozat címe angolul: Satellite and terrestial sensor data processing using open source tools
A feladat részletezése:
Fejlesszen egy megadott terület Sentinel adatainak automatizált letöltésére képes programot. A program legyen alkalmas jól automatizálható feldolgozási műveletek (pl. megadott terület kivágása, igény esetén a vetületi rendszer megváltoztatásával; NDVI számítás) automatizált elvégzésére. A munkához használjon nyílt forráskódú eszközöket.

Intézményi konzulens neve: Dr. Nagy Gábor József

A kiadott téma elévülési határideje:2026. 02. 10.
Beadási határidő: 2023. 12. 15.

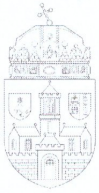
A szakdolgozat: Nem titkos.

Kiadva: Székesfehérvár, 2023. 12. 12. P.H. .....
Intézetigazgató

A dolgozatot beadásra alkalmasnak találom: 2023. 12. 15.

.....
belső konzulens

.....
külső konzulens



ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Alba Regia Műszaki Kar
Természettudományi és Szoftvertechnológiai Intézet

HALLGATÓI NYILATKOZAT

Alulírott hallgató kijelentem, hogy a záródolgozat saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült záródolgozatban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja, a titkosításra vonatkozó esetleges megkötések mellett.

Kelt: LÉTAVÉRTES, 2023. 12. 14

.....

hallgató aláírása

Tartalomjegyzék

1. BEVEZETÉS

Az információ korát éljük. Mind a forrását és mennyiségét tekintve rendkívül mennyiségben, hiszen a nyomtatott formák, rádió, TV mellett életünk részévé váltak az internetről származó adatok is. Ezek a csoportok természetesen tovább bővülnek aszerint, hogy milyen tudományágat művelünk, vagy milyen az érdeklődési körünk. A műholdas távérzékelés ebben a információs dömpingben is kiemelkedő szerepű, még ha ez a mindennapokban nem feltétlenül nyilvánvaló a nagyközönség számára, de mégis meteorológia műholdak képeit (vagy az azokból származtatott adatokat) nézzük, amikor az időjárás-előrejelzésre vagyunk kíváncsiak, amikor valamilyen természeti katasztrófa hatását akarjuk megnézni - vulkánkitörések, áradások, a sor folytatható - vagy gazdálkodóként a terményeink egészségi állapotára vagyunk kíváncsiak egy tábla belsejében, ahová csak károkozással tudnánk bejutni, esetleg a gleccserek állapotát szeretnénk mérni. Ez a lista természetesen csak néhány kiragadott példát tartalmaz, hiszen a műholdas felvételek felhasználása rendkívül széleskörű. Miért ilyen fontos a műholdas távérzékelés? Nem kell a megfigyelni kívánt hely, vagy esemény közelében lennünk, mégis rengeteg információt szerezhetünk róla, akár idősorosan is. Az viszont teljesen nyilvánvaló - ahogy a dolgozatomban a későbbiekben rámutatok - hogy a rendelekzésre álló adatok mennyisége rendkívüli, emiatt szükségszerű ezek szűrése és céljainknak megfelelő feldolgozása ahhoz, hogy segítséget jelenthessenek. Mint ahogy az adatok mennyisége is, a feldolgozási lehetőségek is széleskörűek, ezért egy kiválasztott példán keresztül fogom szemléltetni a célzott feldolgozás hasznosságát. A dolgozat adta keretek nem teszik lehetővé a téma teljes mélységben történő feldolgozását, inkább csak egy bepillantást próbálok nyújtani abba, hogy milyen lehetőségek állhatanak rendelkezésünkre.

2. A TÉMA VÁLASZTÁSÁNAK INDOKLÁSA

A cél az, hogy egy, a felhasználó által megadott területre és időpontra a Sentinel-2 adatokat letöltsük, kivágjuk és kiszámoljuk rá az NDVI értékét.

Miért kell letölteni?

Nem minden projekt kíván meg térképi szerver / infrastruktúra fenntartást, kisebb területeket érintő, vagy térben nagyon szétszórt megfigyelések esetén, vagy pilot projektekben hasznos lehet egy kis erőforrásigényű szkript, amely segítségével egy adott, egyszerűbb feladat gyorsan megoldható. Ehhez mintapéldának az NDVI kiszámítását választottam.

Miért kell kivágni?

Az előzőeken kívül azért, mert az ESA Sentinel-2 műholdrendszer naponta kb. 1,7 terabyte (TB) nyers adatot generál [10]. Ez a mennyiség az összes kép és kiegészítő metainformáció kombinációja, amelyeket a műholdak az összes érzékelőjük által rögzítenek.

Belátható, hogy rendkívüli mennyiségű adatról van szó, így az adat natív felbontásán nagy területre nagyon nagy számítási kapacitást igényelne a feldolgozása. 1 km^2-re körülbelül 1-2GB adat jut, így Magyarország teljes területére 90-180TB adat jönne létre(természetesen nem azonos időben). Ez az adatmennyiség nehezen kezelhető, személyi számítógépeken, lokális elemzésre nem használható.

Miért kell egy szkript?

Több válasz is lehetséges:

- Csak egy egyszerű feladat elvégzésére van szükség, amihez nem kell szakember.
- Nem szükséges nagy méretű teljes térinformatikai szoftverek telepítése.
- Az online megoldásokkal szemben a szkript eredménye internet nélkül is elérhető.(még ha a letöltéshez szükség is van rá)
- Könnyen bővíthető a változó igényeknek megfelelően.

3. A VÁLASZTOTT TERÜLET/PROBLÉMA BEMUTATÁSA.

A megoldanadó probléma sokrétű, több különálló tudományágat érint, emaitt - röviden - bemutatom a kapcsolódó területeket, a következő sorrendben:

1. Távérzékelés - ennek alapjai, formái

2. Műholdas távérzékelés - amely a távérzékelési technikák egy karakterisztikus formája

3. Sentinel-2 műholdak áttekintése

4. A mérhető növényállapot indexekről

5. Az NDVI áttekintése

6. A probléma megfogalmazása

3.1 Távérzékelés

A távérzékelés során a megfigyelésünk tárgyáról úgy gyűjtünk információt, hogy azzal fizikai kapcsolatba nem kerülünk.[1]

Ebből következtethető, hogy a megfigyelésünk tárgyának vagy magától kell kibocsájtania valamilyen érzékelhető sugrázást, és/vagy vissza kell vernie valamilyen sugárzást ahhoz hogy róla adatot gyűjthessünk.

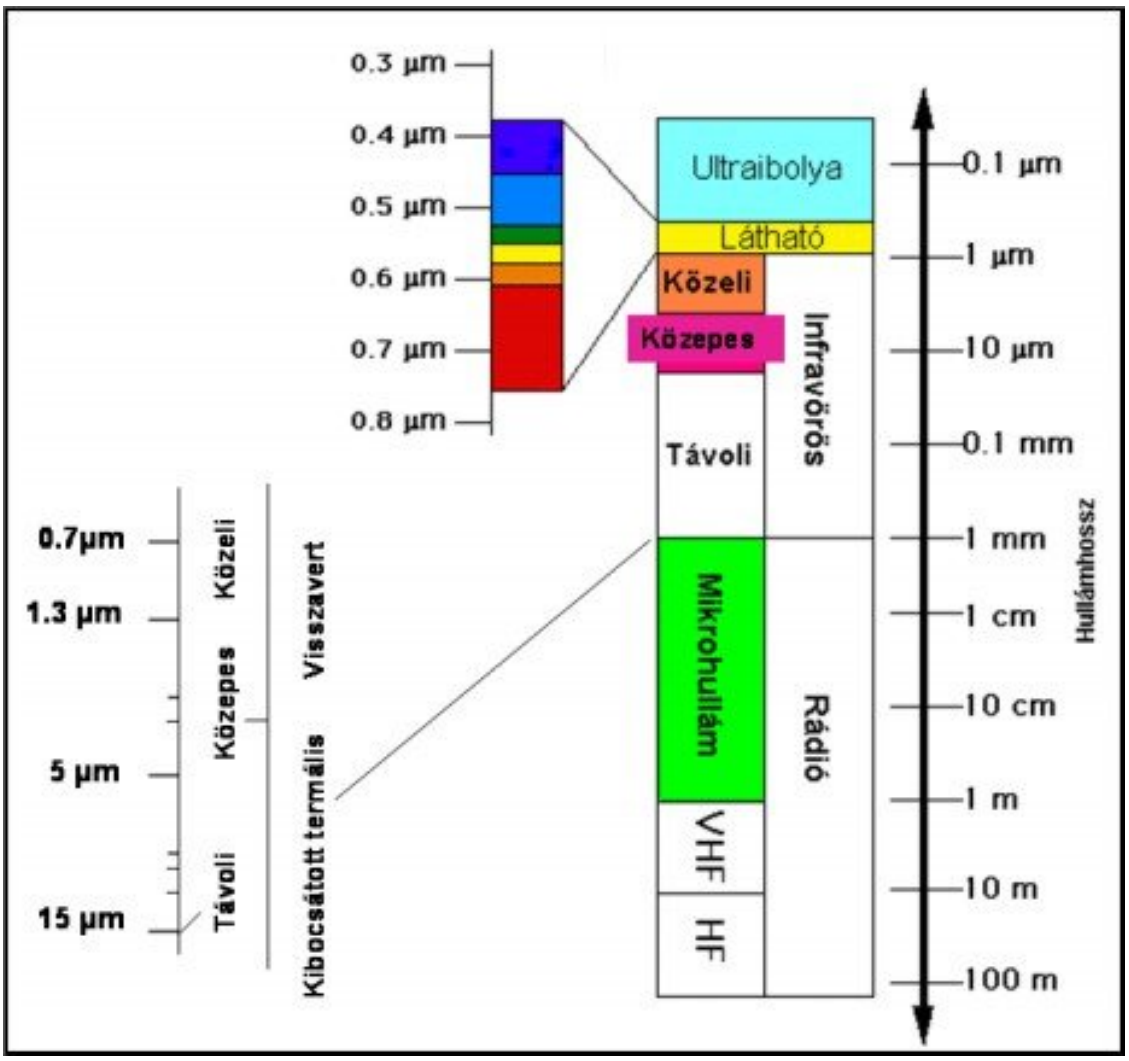
“A távérzékelés fogalmába nem csak az adatgyűjtés, hanem a kapott adatok feldolgozása, értékelése is beletartozik (felvétel készítése, elemzése, interpretálása)” [2][3]

3.1.1 A távérzékelés fizikai alapjai röviden

A távérzékelésbe használt elektromágneses sugárási formák különbözőek és a felhasználási lehetőségeik is eltérőek, emiatt célszerű ezek tulajdonságait áttekinteni.

3.1.2 Az elektromágneses spektrum

Az elektromágneses spektrumból mi csak a látható tartományt - 0,38μm- 0,72μm közötti sávját érzékeljük a szemünkkel.[3] A következő ábrán (1.ábra) láthatjuk, hogy ez a tar-



1.ábra - Az elektromágneses spektum[4]

tomány csak egy nagyon kis szelete a teljes színképnek:

A bármilyen forrásból származó elektromágneses sugárzás energiájának tulajdonságai a hullámfügvénnyel írhatók le [5]:

$$c = \lambda f$$

Ahol a c a fény sebessége, a λ a hullámhossz, a f pedig a frekvencia.

Az egyenletből látható, hogy a hullámhossz csökkenésével (a látható tartománytól az ultraibolya irányába indulva) a sugárzás frekvenciája nő. Az elektromágneses spektrumon ebben az irányban tovább haladva találhatjuk meg a röntgen, γ és kozmikus sugárzásokat.[1]

Ha a hullámhossz nő, a látható tartománytól nézve átlépünk az infravörös tartományba, amely tovább tagolható közeli, közép- és termális részekre.[5]

"A távérzékelésben az elektromágneses hullámokat leggyakrabban a hullámhosszal és az elektromágneses spektrumon belül elfoglalt helyükkel jellemezzük." (Verőné Wojtaszek 2010:6)[5]

"A földi távérzékelési célokra a legfontosabb spektrális régiók a 0,4-0,14μm közötti, az optikai tartományba eső és a 2mm-0,8m közötti, a mikrohullámú tartományba eső részek." (R. P. Gupta, 2018)[6]

Tekintsük át a spektrális régiókat röviden: - ultaribolya - röntgen - γ - 0,4μm-nél rövidebb hullámhosszakon - látható taromány: 0,4μm - 0,7μm -között a kéktől a vörösig terjedő tartomány - Infravörös tartomány: - közeli infra: 0,7μm - 1,3μm közötti tartomány - közepes, vagy rövid hullámú infra: 1,3μm - 3μm - távoli, vagy termális infra: 3μm - 1 mm - Mikrohullámú tartomány: 1mm - 1m hullámhoszak között. - Rádióhullámok: 1m - és a feletti hullámhosszú tartomány.

(Gupta 2018) a látható és infravörös tartományt nevezi *optikai tartomány*-nak. [6][1] Az optikai taromány jellegzetessége, hogy az ebben a tartományban lévő sugárzásokat lehet optikai módszerekkel fókuszálni.

3.1.3 Passzív és aktív távérzékelés

Az adatok gyűjtése az érzékelt sugárzás forrását tekintve két csoportba osztható: - Passzív rendszerek - ez esetben a detektált sugárzás vagy a Nap visszavert sugárzásának egy része, vagy az anyag kémiai-fizikai tulajdonságaiból származó kibocsájtott(emittált) sug-

árzás. - Aktív rendszerek - amikor a detektor mellett sugárforrást is hordoz az (űr)eszköz, például ilyenek a radar és lidar rendszerek.[1]

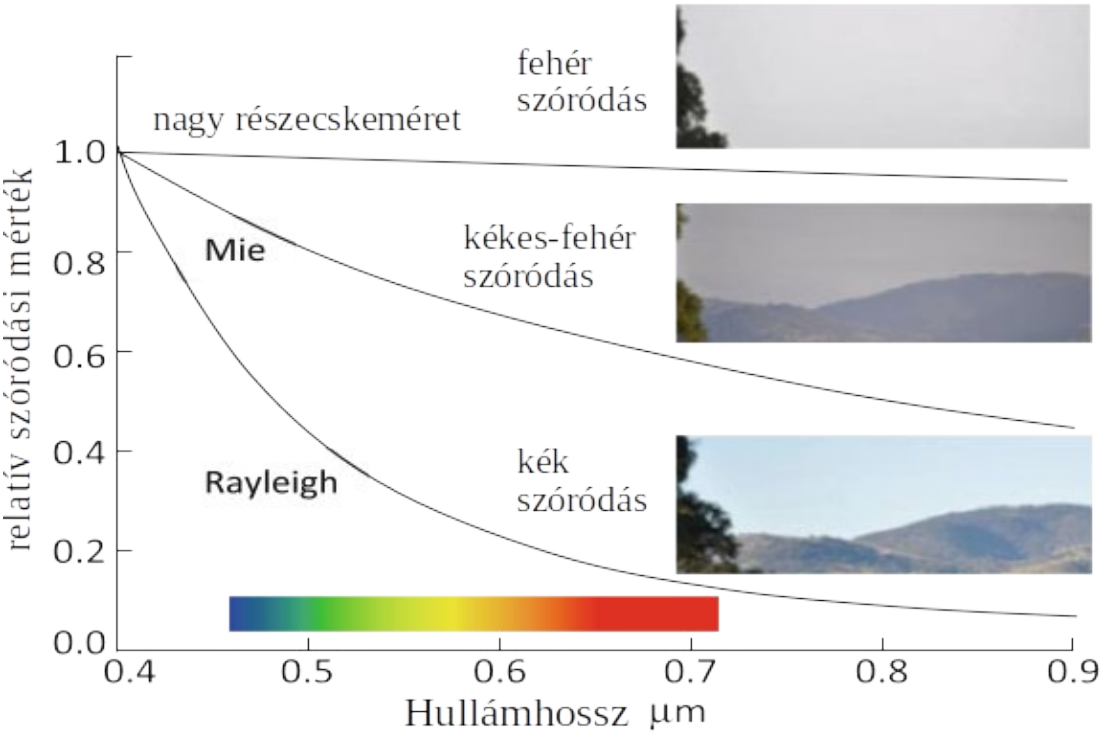
3.1.4 Befolyásoló tényezők

Mind a passzív érzékelők esetében a Nap, az aktívak esetében pedig a használt sugárforrás által előállított sugárzásnak kétszer át kell haladni a légkörön, amely hatással lesz rájuk a hullámhossz és a légkör összetétele függvényében. Hasonlóképpen a fizikai felszín is befolyással van a visszavert sugárzás tualjdonságaira, csakúgy, mint az összetétele, ezért a következő pontokban áttekintjünk ezeknek a hatásoknak a okait.

Légkör

A légkörön áthaladva a sugrázás elnyelődhet és szóródhat, főként annak a függvényében, hogy milyen a sugárzás hullámhosszának és a kontaktusba kerülő részecske méretének az aránya. Három fő hatást sorolhatunk fel: - a részecske átmérője jóval nagyobb, mint a beeső sugárzás hullámhossza: ez a Raylieght szóródás, jellemzője, hogy a hullámhossz csökkenésével párhuzamosan nő. Hatására csökken az élesség, romlik a kontraszt. Kiküszöbölése a magasabb hullámhosszú sugárzás kiszűrésére alkalmas szűrők használatával lehetséges.[5] - A második a Mie szóródás, amikor a szóródást okozó részecske mérete a hullámhossz 1/10-e és 10 szerese közé esik. Ebben a méretben találhatóak a füst, pára részecskéi. Ez a fajta szóródás már nem függ annyira a hullámhossztól, csupán fordítottan arányos.
- A részecskeméret növekedésével a hullámhossz - függőség megszűnik.[7]

A szóródások mértékét, hatásait mutatja be a 2. ábra:



2. ábra Szóródások mértéke és hatásai, forrás: (Richards 2013:53)[7] alapján

A légkör nem csak szórja a rajta áthaladó elektromágneses sugárzást, hanem el is különböző állapotának függvényében részben el is nyeli, ezt hívjuk abszorpciónak.[5]

Ez az abszorpció bizonyos hullámsáv-tartományokat teljesen el tud nyelni, így ezekben a hullámsávokban a légkör átlátszatlan.

“Azokat a tartományokat, melyekben az atmoszféra teljesen vagy részlegesen átengedi az elektromágneses energiát légköri ablakoknak nevezzük.” (Verőné Wojtaszek 2010:10)[5]

Ezen ablakoknak a jelentősége az, hogy műholdas távérzékelést csak ezeken a légköri ablakokon keresztül tudunk végezni.

Domborzat

A Föld felszínére lejutó elektromágneses sugárzás egy része elnyelődik, egy része tovább halad a közegben, a maradék része pedig visszaverődik.[5] Az elnyelést befolyásolja a felszín összetétele, a borítottsága, a simasága és a lejtése is.[1]

$$R_\lambda = \frac{E_v(\lambda)}{E_b(\lambda)} * 100$$

Felszín összetétele és borítottsága

A felszín összetétele és borítottsága a műholdas távérzékelés egyik legfontosabb befolyásoló tényezője.[8] A felszín különböző összetevői eltérőképpen nyelik el a különböző hullámhosszúságú sugárzást. Például a növényzet a látható fényben erősen visszaverődik, míg a talaj a közeli infravörös tartományban erősen elnyeli a sugárzást.

A felszín borítottsága szintén fontos tényező a műholdas távérzékelés szempontjából. A különböző típusú borítások eltérőképpen nyelik el a különböző hullámhosszúságú sugárzást. Például a vízfelszín a látható fényben erősen visszaverődik, míg a felhők a közeli infravörös tartományban erősen elnyeli a sugárzást. [3]

3.2 Műholdas távérzékelés

A földmegfigyelés felől közelítve, a megfigylés tárgyától távolodva a következő "magasságú" csoportokat küllöníthetjük el:

- felszíni távérzékelés - toronyból, daruról, állványról, olyan helyeken, ahol nincs lehetőség légi, vagy műholdas távérzékelésre - barlangokban, belső terekben, illetve ide soroljuk a mikroszkópos megfigyeléseket is

- légi távérzékelés - valamilyen légi járműről történő felvételezést hívjuk légi távérzékelésnek 300m - 10km felszín feletti magasságból

- műholdas távérzékelés – Föld körüli pályáról[9]

3.3 Sentinel-2

A Sentinel-2A-t 2015. június 23.-án bocsájtották fel, a testvérholdját, a Sentinel-2B-t 2017. március 7.-én, egy Vega rakétával, a Francia Guayana-beli Kourou-ból. 786km magas napszinkron pályán keringenek, visszatérési idejük 5 nap az egyenlítőnél, azaz 5 naponta tudnak egy adott területről felvételeket szolgáltatni. Alkalmazási területei: mezőgazdasági monitoring, földhasználati és erdészeti változás vizsgálat,biofizikai jellemzők térképezése: levélzet víztartalmának mérése, levélzet felületének mérése; parti- és szárazföldi vizek monitorozása; veszély- és katasztrófa térképezés.[Sentinel1]

Az eszköz, amellyel a fenti feladatokat végzi egy multispektrális kamera - MSI - , amely 13 csatornával rendelkezik [10] :

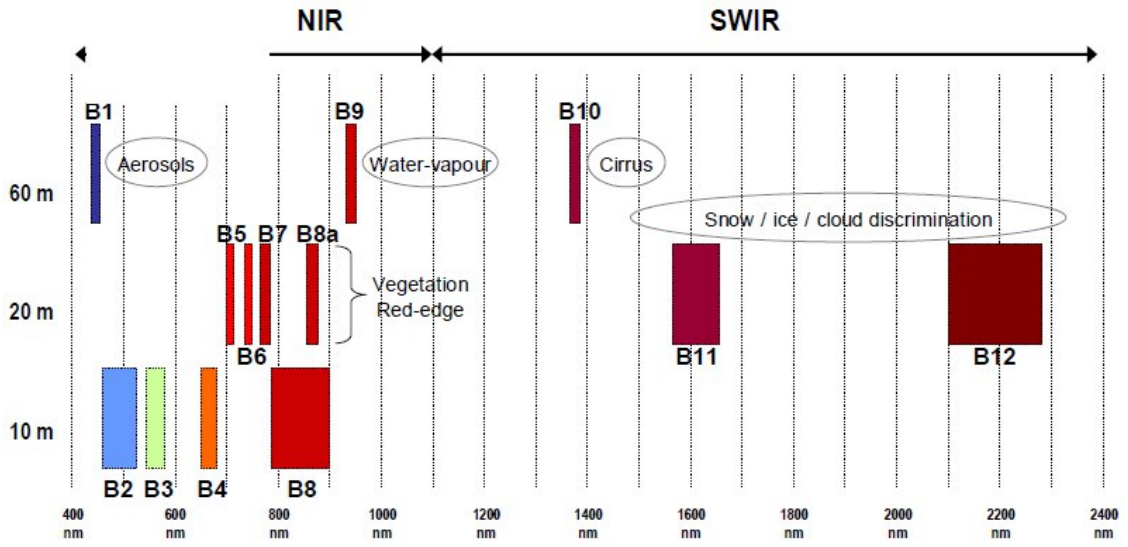
A Sentinel-2 szenzorok spektrális csatornái

Sentinel-2 csatornák	Sentinel-2A		Sentinel-2B		
	Központi hullámhossz (nm)	Sávszélesség (nm)	Központi hullámhossz (nm)	Sávszélesség (nm)	Térbeli felbontás (m)
Band 1 – Tengerparti aeroszol	442.7	21	442.2	21	60
Band 2 – Kék	492.4	66	492.1	66	10
Band 3 – Zöld	559.8	36	559.0	36	10
Band 4 – Vörös	664.6	31	664.9	31	10
Band 5 – Vegetáció “vörös él”	704.1	15	703.8	16	20
Band 6 – Vegetáció	740.5	15	739.1	15	20

A Sentinel-2 szenzorok spektrális csatornái					
"vörös él"					
Band 7 – Vegetáció "vörös él"	782.8	20	779.7	20	20
Band 8 – NIR	832.8	106	832.9	106	10
Band 8A – Keskenysávú NIR	864.7	21	864.0	22	20
Band 9 – Vízpára	945.1	20	943.2	21	60
Band 10 – SWIR – Cirrus	1373.5	31	1376.9	30	60
Band 11 – SWIR	1613.7	91	1610.4	94	20
Band 12 – SWIR	2202.4	175	2185.7	185	20

1. táblázat - A Sentinel-2 műhold csatornakiosztása és jellemzői

A 3. ábrán a sávok spektrális térben való elhelyezkedését és kiterjedését láthatjuk [11]:



3.ábra - Az MSI spektrális sávjai

3.4 Növényállapot indexek

A növényállapot indexek olyan mérőszámok, amelyek segítségével kvantitatív információkat nyerhetünk a növényzet egészségi állapotáról és stresszszintjéről. Ezek az indexek gyakran távérzékelési adatokon alapulnak, és lehetővé teszik a növények fiziológiai és fenológiai jellemzőinek monitorozását. Az növényállapot indexek segítségével információkat kaphatunk a fotoszintézis aktivitásáról, a vízhiányról, a tápanyagellátottságról és más fontos paraméterekről.[12]

3.5 NDVI

Az NDVI (Normalized Difference Vegetation Index) a távérzékelés során alkalmazott mérőszám, amely segítségével a növényzet zöldességét és egészségi állapotát lehet meghatározni. Az NDVI értéke a következő képlettel számítható: NDVI = (NIR - Vörös) / (NIR + Vörös), ahol NIR a közeli infravörös tartományban mért sugárzás intenzitása, míg Vörös a vörös tartományban mért sugárzás intenzitása. Az NDVI skála -1 és 1 között helyezkedik el, ahol a pozitív értékek magas növényzetet, egészséget és zöldességet jelentenek, míg a negatív értékek alacsony növényzetet vagy stresszt mutatnak. Az NDVI alkalmazható a mezőgazdaságban, erdőgazdálkodásban és környezeti monitoringban. [12][13]

4. LEHETSÉGES MEGKÖZELÍTÉSI MÓDOK ÉS MEGOLDÁSOK ÁTTEKINTÉSE ÉS ELEMZÉSE

4.1 Open Source, azaz nyílt forráskód

A nyílt forráskód olyan szoftverek és projektek, amelyek forráskódjukat nyilvánosan hozzáférhetővé teszik és engedélyezik annak módosítását, terjesztését és felhasználását. Ez lehetővé teszi az együttműködést, az innovációt és a szabad hozzáférést a kódkészlettel kapcsolatosan. A nyílt forráskód népszerűvé vált az informatikai iparban és számos projektben, mint például a Linux operációs rendszer és a Mozilla Firefox böngésző.[14]

4.2 Python

A Python egy magas szintű programozási nyelv, amelyet könnyen olvasható és érthető szintaxisa jellemzi. Kiválóan alkalmas általános célú programozásra, adatelemzésre, webfejlesztésre és gépi tanulásra is.[15]

4.2.1 Python modulok

A Python modulok kódrészletek gyűjteményei, amelyek kész funkciókat, osztályokat és változókat tartalmaznak, amelyek segítségével kibővíthetjük a Python nyelv alapfunkcióit. Ezek a modulok egyszerű importálással használhatók a Python programokban, és lehetővé teszik az új funkcionalitás hozzáadását és a kód újrafelhasználását.[15]

Sentinel API A Sentinel API egy olyan programozható felület, amely lehetővé teszi a hozzáférést az Európai Űrügynökség (ESA) Sentinel földmegfigyelő műholdak által gyűjtött adatokhoz. Az API lehetővé teszi a felhasználók számára, hogy különböző földi megfigyelési adatokat, például képeket és sávspektrumokat kérjenek le, valamint különböző lekérdezéseket végezzenek az adatokra, például időszakokra, földrajzi helyekre vagy érzékelési sávokra vonatkozóan. [16]

A műholdadatok feldolgozását segítő pyhon modulok

Sentinelsat: A Sentinelsat egy Python modul, amely lehetővé teszi a Sentinel adatok keresését és letöltését a Copernicus Open Access Hub (SciHub) szolgáltatás egyes nemzeti másolatairól (az eredeti oldal 2023 Októberében megszűnt). A dolgozat elkészítéséhez a Sentinel National Mirror Austria osztrák mirror oldalt használtam. Segítségével lekérdezhetők az elérhető műholdfelvételek és metaadatok, valamint letölthetők a kiválasztott adatok.[17]

PyCWT: A PyCWT (Python Continuous Wavelet Transform) egy Python modul, amely a folytonos hullámmegfejtést és az összehasonlító hullámmegfejtést (continuous wavelet transform, CWT) valósítja meg. Segítségével lehetőség van a Sentinel adatok időbeli

frekvenciájának elemzésére, például az időbeni változások vagy periodikus minták azonosítására.[18]

Snappy: A Snappy egy Python könyvtár, amely az ESA SNAP (Sentinel Application Platform) eszközkészletének Python API-jára épül. Segítségével lehetőség van a Sentinel adatok olvasására, feldolgozására és manipulálására, például felvételek kombinálására, képek kivágására vagy radiometrikus korrekciókra. [19]

4.3 A megoldási módszer kiválasztása, a választás indoklása

A szkript célja hogy egyszerű, paranacssorból használható legyen, emiatt az ehhez legmegfelelőbbnek ítélt modulokat választottam ki. A szükséges Python modulokat a conda csomagkezelőn keresztül telepítettem. A conda *miniconda* változatát használtam, hogy csak a szükséges csomagok kerüljenek telepítésre. A fejlesztés és tesztelés egésze Linux operációs rendszeren történt (Ububntu 22.04 alapú, Pop_OS 22.04).

5. A TERVEZÉS SORÁN VÉGZETT MUNKAFÁZISOK ÉS A TAPASZTALATOK LEÍRÁSA

5.1 A szkript folyamata

A szkript futását a következő lépésekkel terveztem

- A feldolgozáshoz szükséges modulok importálása
- Köszönti a felhasználót és megkéri a CopernicusHUB felhasználói nevét és jelszavát, amit a műholdfelvételek letöltéséhez használ
- Kilistázza a futtatási könyvtárban lévő GeoJSON állományokat és megkéri a felhasználót, hogy válasszon.
- Megkéri a felhasználót, hogy adja meg a keresési feltételeket: kezdő időpont, végidőpont és maximális felhőborítottság
- Lekérdezi a CopernicusHUB-ról a keresési feltételeknek megfelelő és a megadott GeoJSON-nal átfedő felvételeket

- Sorbarendezi fedettségi százalék és dátum szerint a találati listát és kiválasztja a legkisebb felhőborítottságú és legfrissebb felvételt
- Letölti a kiválasztott felvételt és kitömöríti a B04-es és B08-as rétegeket, amelyek az NDVI számításához szükségesek
- Elvégzi az NDVI érték kiszámítását a kitömörített rétegek felhasználásával, kivágja a megadott GeoJSON befoglalójára és elmenti GeoTif formátumban

5.2 A megvalósítás leírása

A korábban felsorolt modulokat a következő célra használtam a szkriptben:

sys: A Python rendszermodul, amely hozzáférést biztosít a Python környezethez és a rendszerhez.

glob: A glob modul a fájlrendszeren keresi a fájlokat és könyvtárakat.

os: Az os modul a fájlrendszerhez és a rendszerhez való hozzáférést biztosít.

zipfile: A zipfile modul a ZIP fájlok kezelését biztosítja.

getpass: A getpass modul titkosított bevitelt kér a felhasználótól.

fiona: A fiona modul a földrajzi adatformátumok olvasását és írását biztosítja.

geopandas: A geopandas modul a földrajzi adatokat pandas adatkeretek formájában reprezentálja.

numpy: A numpy modul a numerikus számításokhoz szükséges eszközöket biztosítja.

rasterio: A rasterio modul a raster adatok olvasását és írását biztosítja.

gdal/osgeo: A gdal/osgeo modul a térinformatikai adatok olvasását és írását biztosítja.

sentinelsat: A sentinelsat modul a Sentinel-2 műholdképekhez való hozzáférést biztosítja.

shapely: A shapely modul a földrajzi alakzatokat reprezentálja.

5.2.1 A szükséges modulok importálása

```
import sys
import glob
import os
import zipfile
import getpass
import fiona
import geopandas as gpd
import numpy as np
import rasterio
from osgeo import ogr
from rasterio.mask import mask
from sentinelsat import SentinelAPI
from shapely.geometry import shape, box
```

5.2.2 A felhasználói adatok bekérése

Ebben a részben használtam fel a **getpass** modult, hogy elkerüljem a jelszavak kiírását a parancssorba. Ezután egy **try...except** eljárásban ellenőrzöm, hogy tud-e kapcsolódni a megadott adatokkal a szkript; ha nem, hibát ad vissza és kilép.

```
print("Üdvözli a Sentinel2 NDVI számító szkript!")
print("")
print("Kérem adja meg a CopernicusHUB azonosítóját és jelszavát:")
user = input("Felhasználói név: ")
passw = getpass.getpass(prompt='Password:', stream=None)
try:
  api = SentinelAPI(user, passw, 'https://scihub.copernicus.eu/dhus')
except:
  print("Hiba a kapcsolódáskor.")
  sys.exit("Hibás API lekérés.")
```

5.2.3 GeoJSON állományok listázása

Ebben a részben a szkript forráskönyvtárában lévő geojson formátumú állományokat listázza ki a `glob` modul használatával; ellenőrzi, hogy vannak-e egyáltalán, ha nincsenek visszajelzi a felhasználónak. Ezután történik a felhasználni kíván geojson állomány kiválasztása: a szkript egy számozott listát készít az `enumerate` függényt alkalmazásával, ezt kiírja a képernyőre. Ezután a felhasználó kiválasztja a használni kívánt geojson állományt, a sorszámát beírva; ha nem a listában szereplő sorszámok közül választ, hibaüzenet kap.

```
geojson_files = glob.glob("*.geojson")
    if not geojson_files:
        sys.exit("Nincs geojson file!")
 print("Most a keresési feltételeket adja meg: ")
 for i, file in enumerate(geojson_files):
    print(f"{i+1}. {file}")

 selection = int(input("Válasszon egy sorszámot a fenti GeoJSON fájlok közül: "))
if 1 <= selection <= len(geojson_files):
    geojson_file = geojson_files[selection - 1]
else:
    sys.exit("Érvénytelen sorszám.")
```

5.2.4 Keresési feltételek megadása

Ebben a szakaszban folytatódik a keresési feltételek bekérése a felhasználótól, a kezdő- és végdátumok és a elfogadható maximális felhőborítottság megadásával. Az elfogadható formátumokat a promptban jeleníti meg a szkript.

```
date_from = input("Keresési kezdődátum (yyyymmdd): ")
```

```
date_to = input("Keresési végdátum (yyyymmdd): ")

max_cloud_cover = int(input("Maximális felhőfedettség (0-100): "))
```

5.2.5 Lekérdezés a Sentinel National Mirror Austria szerverről

Ez a kezdő szakasza a Sentinel National Mirror Austria-ról történő lekérdezéseknek. - A felhasználó által megadott dátumok egy date_range nevű változóban kerülnek tárolásra. - A platform_name változóban a kívánt műlodas platform neve van megadva.

- A szkript ezután előkészíti a kiválasztott geojson állományt lekérdezéshez a fiona modul használatával:
  - végigmegy a geojson-ben található geometriai elemeken
  - átalakítja wkt formátumúvá
  - kinyeri a geometriai elem befoglalóját
  - megadja az elem sarok koordinátáit
  - utána wkt-vá alakítja
- Ezután a geojson állományt egy külön változóban - gdf - megnyitja a később letöltésre kerülő sávok kivágásához a geopandas modul segítségével
- Ezt követően kerül sor a Copernicus HUB-ról a lekérdezés indítására, az előkészített adatokkal egy try...except struktúrában. Ha a lekérdezés sikertelen vagy ha nem talál a keresési feltételeknek megfelelő terméket, visszajelzi a felhasználónak.
- Ha talál megfelelő adatot akkor kiszámolja, hogy teljes átfedésben van-e a geojson geometriájával.
  - Az intersect függvénnyel megnézi, hogy valóban ven-e átfedés; ha van, kiszámítja az átfedett területet
  - kiszámolja a geojson poligon területét
  - összehasonlítja az átfedő terület méretét a geojson területével;

– ha a két szám egyezik, a talált terméket a filtered_products dictionary típusú változóhoz csatolja

```
date_range = (date_from, date_to) platform_name = ‘Sentinel-2’
with fiona.open(geojson_file) as src:
  for feature in src: geometry = shape(feature[‘geometry’])
    geometry_wkt = ogr.CreateGeometryFromWkt(str(geometry))
    envelope = geometry.envelope
    xmin, ymin, xmax, ymax = envelope.bounds
    envelope_wkt = ogr.CreateGeometryFromWkt(str(envelope))

gdf = gpd.read_file(geojson_file)

try:
  products = api.query(envelope, date=date_range, platformname=platform_name, pro-
ducttype=‘S2MSI2A’)
except Exception:
  sys.exit(“Sikertelen API lekérés!”)
  if len(products) == 0: sys.exit(“Nem található temék a megadott keresési feltétalakkal!”)

filtered_products = {}
for product_id in products:
  product_info = api.get_product_odata(product_id)
  product_footprint = ogr.CreateGeometryFromWkt(str(product_info[‘footprint’]))
  if product_footprint.Intersects(envelope_wkt):
    intersection = envelope_wkt.Intersection(product_footprint)
    intersection_area = intersection.GetArea()
    polygon_area = envelope_wkt.GetArea()
    coverage_percentage = (intersection_area / polygon_area) * 100

  if coverage_percentage == 100:
    filtered_products[product_id] = products[product_id]
```

5.2.6 Találati lista rendezése és letöltendő termék kiválasztása

Ebben a lépésben a szkript sorbarendezi fedettségi százalék és dátum szerint a találati listát.

- `geodataframe`-be konvertálja a találati listát
- sorbarendezi a felhőfedettségi százalé és a felvételi készítésének dátuma szerint növekvő sorrendbe egy új változóban (`products_df_sorted`)
- A `products_df_sorted` -et szűri a `head` függvénnyel, annak az első elemére, így kapjuk *egy* eredményterméket
- kinyeri az eredményterméket tartlamazó listából a termék azonosítóját és szöveges formátumúvá alakítja
- kinyeri a `baseline` értékét a tulajdonságok közül - erre azért van szükség a későbbiekben, mert a 4-es baseline-ú felvételek pixelértékeit új feldolgozási eljárással készültek, emiatt az értékük módosult.[20]

```
products_df = api.to_geodataframe(filtered_products)
products_df_sorted = products_df.sort_values(['cloudcoverpercentage', 'ingestiondate'], ascending=[True, True])
products_df_sorted = products_df_sorted.head(1)

prod_id = str(products_df_sorted['uuid'].iloc[0])

baseline = float(products_df_sorted['processingbaseline'].iloc[0])

selected_columns = ['title', 'processingbaseline', 'cloudcoverpercentage']
products_df_info = products_df_sorted[selected_columns]
product_name = products_df_sorted['title'].iloc[0]
print(products_df_info.to_string(index=False))
```

5.2.7 Kiválasztott felvétel letöltése

A letöltés előtt egy ./imgtmp alkönyvtárat hoz létre a szkript, majd ebbe letölti a kiválasztott felvételt. A letöltött zip kiterjesztésű állományok közül a legutoljára letöltöttet - max(..., key=os.path.getctime) - kiírja a képernyőre. Ezután a letöltött zip állományból szelektíven kicsomagolja a _B04_10m.jp2 és a _B08_10m.jp2 -re végződő állományokat a zip állomány nevét viselő alkönyvtárba. A kicsomagolás sikerességéről értesíti a felhasználót.

```
os.makedirs(directory_path, exist_ok=True)
search_pattern = os.path.join(directory_path, '*.zip')
if zip_files := glob.glob(search_pattern):
  latest_zip_file = max(zip_files, key=os.path.getctime)  # A legfrissebb .zip állomány kiválasztása
  print("Legújabb .zip állomány:", latest_zip_file)
else:
  print("Nincs .zip állomány a könyvtárban.")

with zipfile.ZipFile(latest_zip_file, 'r') as zip_ref:
  for file_name in zip_ref.namelist():
    if file_name.endswith('_B04_10m.jp2') or file_name.endswith('_B08_10m.jp2'):
      dest = os.path.join(directory_path, os.path.basename(file_name))
      zip_ref.getinfo(file_name).filename = dest
      zip_ref.extract(file_name)
       print("A sávok sikeresen kibontva a megadott célmappába.")
```

5.2.8 Kivágás, NDVI kiszámítása a megadott területre, GeoTiff mentése

A szkript ezen része először megkeresi a kicsomagolt állományokat a letöltési mappában. Megnézi, hogy az állományok nevében benne van-e a _clip szöveg - ami arra utalna, hogy már megtörtént a jp2 állományok kivágása. Ezután megnyitja az állományokat a rasterio modul segítségével. A megnyitott objektumból kiolvasott koordinátarendszert használja a “gdf” nevű GeoDataFrame objektum (amit korábban hoztunk létre a geojson megnyitásakor) koordinátáinak átalakítására, és létrehoz egy box objektumot, amely a geojson

objektum teljes területét körülvevő négyzet koordinátáit tartalmazza. Ezután ezt a geometriá használja a rasterio.mask.maks függvényben, mint a kivágás körvonalát. Itt még nem történik meg a kivágás. Előbb kimásolja a forrásállomány metaadatait a meta.copy() függvénnyel ezután frissíti ezeket az előbbiekben létrhozott paraméterekkel. Ezt követően megadja a kimeneti állomány nevét és a kiterjesztését tif-re állítja, majd megadja a kimeneti könyvtárat. Ezután történik a kivágott állomány kiírása a háttértárra, majd a csatornaszámuknak megfelelő nevű változókba betöltésre kerülnek. Ez követően kerül az NDVI kiszámításra, a következő módon. - A sávokat beolvassuk a red_band (b4) és nir_band (b8) változókba és folattá konvertáljuk az értékekiket. - Ezután vizsgáljuk, hogy melyik feldolgozási baseline-ba tartoznak; ha 4-nél kissebbe, akkor az értékeikből 1000-et kivonunk - Ezt követően lekezeljük az esetleges érvénytelen értékeket illetve a nullával való osztást a numpy modul segítségével és elvégezzük az ndvi értékek kiszámítását. - Végül az egyik sáv metaaddatait felhasználva, a kimeneti könyvtárba a letöltött felvétele neve + `_NDVI.tif` utótaggal, elmentjük az állományt.

```
jp2_files = glob.glob(os.path.join(directory_path, '*.jp2'))

for jp2_file in jp2_files:
  if '_clip' not in jp2_file:
    with rasterio.open(jp2_file) as src:
      raster_crs = src.crs
      gdf_utm = gdf.to_crs(raster_crs)
      bbox_utm = gdf_utm.total_bounds
      bbox_utm_geometry = box(*bbox_utm)

      out_image, out_transform = rasterio.mask.mask(src, [bbox_utm_geometry], crop=True)
      out_meta = src.meta.copy()
      out_meta.update({
        "driver": "GTiff",
        "height": out_image.shape[1],
        "width": out_image.shape[2],
```

```
                "transform": out_transform
    })

    clipped_file = os.path.basename(jp2_file).replace('.jp2', '_clip.tif')
    output_path = os.path.join(directory_path, clipped_file)

    with rasterio.open(output_path, "w", **out_meta) as dst:
      dst.write(out_image)
      os.rename(output_path, output_path.replace('.jp2', '.tif'))

    if 'B04' in jp2_file:
      b4 = rasterio.open(output_path)
    else:
      b8 = rasterio.open(output_path)

 # NDVI számítás
red_band = b4.read()
nir_band = b8.read()
red_band = red_band.astype(float)
nir_band = nir_band.astype(float)

if not float(baseline) < 4:
    red_band = red_band - 1000
    nir_band = nir_band - 1000

with np.errstate(divide='ignore', invalid='ignore'):
    ndvi = np.where((nir_band + red_band) == 0.0, np.nan, (nir_band - red_band) /
(nir_band + red_band))

meta = b4.meta
meta.update(driver='GTiff')
meta.update(dtype=rasterio.float32)
```

```
ndvi_output_path = os.path.join(directory_path, f"{product_name}_NDVI.tif")

with rasterio.open(ndvi_output_path, "w", **meta) as ndvi_dst:
  ndvi_dst.write(ndvi.astype(rasterio.float32))
```

5.3 A tapasztalatok és továbbfejlesztési lehetőségek

A következő fejlesztési lehetőségek merülnek fel:

- A geometriák kezelését valószínűleg lehetne egyszerűsíteni.

- Sokkal több ellenőrzést végrehajtani(try...except) és több segítséget adni a felhasználónak, ha valami probléma lép fel. Például ha túlságosan sok töréspontot tartalmazó, vagy túl nagy területet lefedő poligont adunk meg, olyan hibával lép ki, ami nem utal egyértelműen arra, hogy emiatt szakadt meg a program futása.

- Előzetesen is lehetne a szolgáltatás állapotát ellenőrizni, esetleg több mirrort megadni, és azokat rangsorolni latency szerint.

- A külön változó egy platform használata esetén nem szükséges, de ha ki szeretnénk bővíteni a feldolgozni kívánt műholdas adatok listáját, itt tárolhatnánk a kiválasztott platform(műhold) nevét.

- A szript szerkezetét is lehetne ésszerűsíteni.

- A nagy méretű eredeti Sentinel felvételeket a feldolgozás után lehetne törölni.

- Több index számítását is bele lehetne venni(például a https://www.indexdatabase.de/ alapján), műholdfüggő módon.

- Egyszerű webes térképi felületet készíteni az interaktívabb használat kedvéért, Leaflet/Openlayers használatával.

- Teljesen automatizált(interakció nélküli) feldolgozás specializált szolgáltatáshoz, sqlite, vagy postgres alapú adattárolással.

- Nem csak befoglalóra, hanem poligonra vágni.

- Egy egyszerű docker image-et készíteni a futtatáshoz.

6. RÖVID TARTALMI ÖSSZEFOGLALÓ

A dolgozatban a nagy mennyiségű rendelkezésre álló műholdas adatok célorientált feldolgozása kerül bemutatásra, az NDVI számítás példáján keresztül. Az alapvető kérdés az, hogy hogyan tudunk származtatott adatot egyszerű eszközökkel, könnyen ismételhető módon létrehozni a nyers adatokból amely már speciális kérdésekre tud választ adni. Ennek érdekében olyan Python szkipt fejlesztése kerül bemutatásra, amely a felhasználó által megadott paraméterek - terület (poligon, geojson formátumban), kezdeti- és végdátumok és maximális felhőborítottság százalékos megadása után egy Sentinel mirror oldalon elvégzi a lekérdezést, ha talál, letölti a legfrissebb és legkisebb felhőborítottságú felvételt, majd kivágja a megadott területre, kiszámítja az NDVI értéket és tif formátumban elmenti. A könnyebb választhatóság kedvéért a geojson állományokat a szkript futtatásának könyvtárából kilistázza. A szkript bekéri a felhasználói nevet és jelszót a kiszolgálóhoz, de a jelszót rejtve kezeli; a megadott geojson állományból kinyeri annak befoglalóját, és annak koordinátáit a letöltött adathoz vetületéhez igazítja. Megvizsgálja, hogy a kiválasztott területünk valóban tartalmaz-e adatot és csak akkor tölti le, ha az átfedés 100%-os. Ezután letölti, kicsomagolja az NDVI számításhoz szükséges két csatornát és majd ezek adatait annak fényében módosítja, hogy milyen feldolgozáson mentek át, majd ezekből kiszámítja az NDVI értékét és tif formátumban lementi a munkakönyvtárba. A szkript alpul szolgálhat egy webes alkalmazáshoz, ahol a műhold, az index, a terület, a kezdő- és végdátumok, illetve a felhőborítottság kiválasztása egy böngészős felületen törénhetne, az eredméyn állomány pedig a webes térképen jelenne meg. Vagy egy automatizált szkriptnek, amely idősoros NDVI felvételeket készít változásvizsgálathoz egy adott területre.

Idegen nyelvű tartalmi összefoglaló (angolul)

The paper presents the targeted processing of large amounts of available satellite data, using the calculation of the NDVI as an example. The basic question is how to create derived data from raw data using simple tools and in a repeatable manner, which can already answer specific questions. To this end, the development of a Python script is presented, which, after the user has specified the parameters - area (polygon, in GeoJSON format), start and end dates, and maximum cloud cover percentage, performs a query on a Sentinel mirror site. If it finds a match, it downloads the latest and least cloud-covered image, then cuts it to the specified area, calculates the NDVI value, and saves it in TIFF format. For easier selection, it lists the GeoJSON files from the script's execution directory. The script requests the user name and password for the server, but handles the password in a hidden manner; it extracts the bounding box from the specified GeoJSON file, and aligns its coordinates to the projection of the downloaded data. It checks whether the selected area actually contains data and only downloads it if the overlap is 100%. Then it downloads, unpacks the two channels required for NDVI calculation, and then modifies their data based on the processing they have undergone. It then calculates the NDVI value from these and saves it in TIFF format to the working directory. The script can be used as the basis for a web application, where the selection of the satellite, index, area, start and end dates, and cloud cover could be done on a web interface, and the resulting file would be displayed on the web map. Or for an automated script that creates time series NDVI images for change detection for a given area.

7. IRODALOMJEGYZÉK

[1]D. József Lóki, Távérzékelés: földrajz tanárszakos és szakgeográfus hallgatók számára. Debrecen: Kossuth Egyetemi Kiadó, 1996.

[2]Berke, J., Kozma-Bognár, V., Távérzékelés alapjai, köt. Digitális képfeldolgozás és alkalmazásai (DIGKEP v7.0). Elektronikus és nyomtatott tankönyv. Keszthely: Kvark Számítástechnikai Bt., 2010a.

[3]C. P. Lo és A. K. W. Yeung, Concepts and techniques of geographic information systems, 2nd ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2007.

[4]F. Sárközi, „Térinformatika”, Térinformatika elmélet oktató anyag. Elérés: 2021. április 27. [Online]. Elérhető: http://www.agt.bme.hu/tutor_h/terinfor/tbev.htm

[5]Verőné Wojtaszek M., Fotointerpretáció és távérzékelés 1., A távérzékelés fizikai alapjai|Digitális Tankönyvtár. Nyugat-magyarországi Egyetem, 2010. Elérés: 2021. március 30. [Online]. Elérhető: https://regi.tankonyvtar.hu/hu/tartalom/tamop425/0027_FOI1/adatok.html

[6]R. P. Gupta, Remote Sensing Geology, 3rd ed. 2018. Berlin, Heidelberg: Springer Berlin Heidelberg : Imprint: Springer, 2018. doi: 10.1007/978-3-662-55876-8.

[7]J. A. Richards, Remote sensing digital image analysis: an introduction, Fifth edition. Berlin: Springer, 2013.

[8]T. Lillesand, R. W. Kiefer, és J. Chipman, Remote Sensing and Image Interpretation. Wiley, 2015. [Online]. Elérhető: https://books.google.hu/books?id=AFHDCAAAQBAJ

[9]„Fotogrammetria 1., A távérzékelés fogalma, a fotogrammetria és a távérzékelés kapcsolata|Digitális Tankönyvtár”. Elérés: 2021. április 27. [Online]. Elérhető: https://regi.tankonyvtar.hu/hu/tartalom/tamop425/0027_FOT1/ch01s02.html

[10]„Facts and figures”. Elérés: 2020. november 14. [Online]. Elérhető: https://www.esa.int/Applications/Observing_the_Earth/Copernicus/Sentinel-2/Facts_and_figures

[11]„Sentinel-2_ESA_Bulletin161.pdf”.

[12]S. Huang, L. Tang, J. P. Hupy, Y. Wang, és G. Shao, „A commentary review on the use of normalized difference vegetation index (NDVI) in the era of popular remote sensing”, J. For. Res., köt. 32, sz. 1, o. 1–6, febr. 2021, doi: 10.1007/s11676-020-01155-1.

[13]A. Bannari, D. Morin, F. Bonn, és A. R. Huete, „A review of vegetation indices”, Remote Sensing Reviews, köt. 13, sz. 1–2, o. 95–120, 1995, doi: 10.1080/02757259509532298.

[14]N. Vainio, „Free Software Philosophy and Open Source”, Handbook of Research on Open Source …, jan. 2007, Elérés: 2023. december 12. [Online]. Elérhető: https://www.academia.edu/918555/Free_Software_Philosophy_and_Open_Source

[15]„The Python Tutorial”, Python documentation. Elérés: 2023. december 12. [Online]. Elérhető: https://docs.python.org/3/tutorial/index.html

[16]„Open Access Hub”. Elérés: 2023. december 12. [Online]. Elérhető: https://sci-hub.copernicus.eu/userguide/BatchScripting

[17]„Sentinelsat — Sentinelsat 1.2.1 documentation”. Elérés: 2023. december 10. [Online]. Elérhető: https://sentinelsat.readthedocs.io/en/latest/

[18]„PyCWT: spectral analysis using wavelets in Python — PyCWT 0.3.0a22 documentation”. Elérés: 2023. december 12. [Online]. Elérhető: https://pycwt.readthedocs.io/en/latest/

[19]„SNAP – STEP”. Elérés: 2023. december 12. [Online]. Elérhető: https://step.esa.int/main/toolboxes/snap/

[20]„S2 Products - SentiWiki - Confluence”. Elérés: 2023. december 10. [Online]. Elérhető: https://omcs.atlassian.net/wiki/spaces/CSWK/pages/407406098/S2+Products