

Naprendszerbeli kis égitestek nyílt forráskódú termális modellje

Országos Tudományos Diákköri Konferencia Dolgozat

Kinyó András

Eötvös Loránd Tudományegyetem
Fizika BSc II. évfolyam



Témavezetők:

Kalup Csilla

tudományos segédmunkatárs
CSFK Konkoly Thege Miklós Csillagászati Intézet

Kiss Csaba

tudományos tanácsadó
CSFK Konkoly Thege Miklós Csillagászati Intézet

Eötvös Loránd Tudományegyetem Természettudományi Kar
Csillagászati Tanszék
Budapest
2023

Tartalomjegyzék

Kivonat	3
1. Bevezetés	4
2. NEATM gömbre	6
2.1. Geometriai elrendezés	6
2.2. Nyalábparaméter	6
2.3. A felszín paraméterezése	6
2.4. Szubszoláris hőmérséklet	7
2.5. Beesési szögek, felszíni hőmérséklet	7
3. NEATM ellipszoidra	9
3.1. A felszíni rács	9
3.2. Beesési szögek	10
3.3. Szubszoláris hőmérséklet	10
3.4. Felszíni hőmérséklet	10
3.5. Vetítés a fokális rácsra	10
4. NEATM poliéderre	13
4.1. A felszíni rács	13
4.2. Beesési szögek	14
4.3. Szubszoláris hőmérséklet	14
4.4. Felszíni hőmérséklet	14
4.5. Vetítés a fokális rácsra	15
4.6. A spektrális fluxussűrűség-eloszlás	15
4.7. Méretskálák, a poliéderrács konfigurálása	15
4.8. A (216) Kleopatra	16
5. Különböző alakmodellek összehasonlítása	18
5.1. Ellipszoid modell Descartes-i ráccsal	18
5.2. Ellipszoid modell HealPix ráccsal	19
5.3. Poliéder modell	21
6. Színkorrekció	24
7. A program felépítése	25
7.1. Segédfüggvények	25
7.2. Vektorok beolvasása és kezelése	25
7.3. A paraméterek beolvasása	26
7.4. A <code>neatm</code> osztály	28
8. Összefoglalás	30
9. Köszönetnyilvánítás	31
Hivatkozások	32

Kivonat

A csillagászat több területén is nagyon fontos szerepet játszanak a termális modellek. Naprendszerbeli objektumok esetén ezek segítségével határozhatjuk meg különböző égitestek méretét és albedóját, melyek a legalapvetőbb adatok közé sorolhatóak. A termális modellek egyik típusa az ún. NEATM (Near-Earth Asteroid Thermal Model), mely eredetileg földsúroló kisbolygók leírására találták ki, azonban számos más esetben is felhasználható közelítésnek bizonyult. Annak ellenére, hogy a naprendszerkutatás területén széles körben elterjedt módszerről van szó, nincs egy, a tudományos közösség számára elérhető ingyenes szoftver, amellyel lehet a NEATM modell szerint számolni.

Munkám során a célom egy ilyen, továbbfejlesztett NEATM program létrehozása volt *python* nyelven. A programot sikerült létrehoznom, mely `pyneatm` néven szabadon elérhető a GitHub nevű online szoftvermegosztó oldalon¹.

¹<https://github.com/kinyeaux/pyneatm>

1. Bevezetés

A termális kisbolygómodellek a naprendszerkutatás egyik alapvető jelentőségű sarokkövei. Látható és infravörös, valamint akár a szubmilliméteres tartományban végzett fotometriai méréseket megfelelő termális modellekkel kombinálva lehetőségünk van ugyanis meghatározni a kisbolygók méretét és albedóját. Az egyik legegyszerűbb ilyen modell a földsúroló kisbolygók példájára kifejlesztett ún. NEATM (Near-Earth Asteroid Thermal Model, [Delbó & Harris \(2002\)](#)), ilyen módszerrel határozták meg kisbolygók túlnyomó többségének méretét, pl. a WISE/NEOWISE űrprogramok mérései alapján. Az így kapott átmérő és albedó mérések kulcsfontosságúak ahhoz, hogy következtetni lehessen a kisbolygók egyéb fizikai tulajdonságaira, taxonómiájára és eredetére, valamint információkat kaphatunk a felszín egyéb jellemzőiről, pl. a hőtehetetlenségéről és kráterezettségéről is.

Egy konkrét felhasználási terület például a földsúroló kisbolygók vizsgálata. A kisbolygóövben keringő égitestek pályájának módosulása többek között a Jarkovszkij- és YORP-effektusoknak köszönhető. Mindkét hatás az égitest felszíni sugárzásával áll kapcsolatban, így azok albedójának, méretének, forgási sebességének, alakjának és hőtehetetlenségének ismerete elengedhetetlen, ha meg akarjuk érteni, pontosan milyen mechanizmusokkal kerülhetnek kisbolygók a Föld közelébe. Emellett persze sok más helyen is felhasználható a modell, fővi kisbolygókon, vagy bármely olyan populáció esetén, amikor bizonyos fizikai paraméterek ismeretének hiánya meggátolja komplexebb, ún. termofizikai modellek használatát. A CSFK Naprendszerkutató Csoportja által fejlesztett Infravörös Kisbolygó Adatbázis² ([Szakáts et al., 2020](#)) több százezer termális kisbolygó mérést tartalmaz, amely óriási, kiaknázatlan lehetőséget rejt termális modellek alkalmazására.

Az eredeti NEATM modell egy nagyon egyszerű elrendezéssel dolgozik. Az égitest felszínét gömb alakúnak tekinti, a spektrális emisszivitását minden hullámhosszon azonosan konstansnak. Ezen feltételezések nagymértékben befolyásolják a számolások eredményét, és a modellt sokféleképpen lehet fejleszteni. Munkám során több módon is próbáltam pontosítani az elérhető eredményeket. A kódban implementálva van egy függvény, amely a bemeneti paraméterként megadott spektrális emisszivitásból (itt már nem egy konstans, hanem egy hullámhosszfüggő mennyiség) számolja ki a felszíni hőmérsékletet és a spektrális fluxussűrűséget. Emellett (eddig) 28 különböző űrtávcsőkamerára tud a program színkorrekciót számolni, így helyesen adja vissza az adott műszer által érzékelt intenzitáseloszlást. További fejlesztés még, hogy gömb mellett háromtengelyű ellipszoid alakot is tud kezelni, kétféle különböző felszíni rács mellett, valamint úgynevezett wavefront (.obj) fájlokat is tud olvasni, mely lehetőséget ad arra, hogy amorf alakú égitestek felszínét poliéderként modellezve számoljunk. Ez különösen érdekes, hiszen manapság rengeteg, okkultációs mérésekből származtatott alakmodell elérhető ([Ortiz et al., 2017](#); [Dunham et al., 2019](#)), ezek termális modellezése alapul szolgálhat komplexebb, termofi-

²<http://ird.konkoly.hu>

zikai modellek létrehozásához, melyek figyelembe veszik a felszín egyes rétegei közti hőátadást, és más belső folyamatokat (Delbo et al., 2015; Müller et al., 2019).

Habár a fenti módszer sokrétűen használt a naprendszerkutatás során, jelenleg nem áll rendelkezésre szabadon elérhető, nyílt forráskódú, kisbolygók hősugárzását (termális emisszióját) modellezni képes program, ezért egy ilyen létrehozása hiánypótlónak bizonyult a naprendszerkutatás területén. A kód fejlesztését 2022. szeptemberében kezdtem el, a Csillagászati és Földtudományi Kutatóközpont Konkoly Thege Miklós Csillagászati Intézetének numerikus asztrofizikai demonstrátoraként. Egyik témavezetőm, Kiss Csaba már készített egy hasonló programot, IDL nyelven, így elsődleges célom ennek megértése volt, majd implementációja, fejlesztése pythonban, a `numpy` modul felhasználásával.

A kód használatát igyekeztem minél érthetőbbé és egyszerűbbé tenni. Az egész programot pythonban írtam, a különböző függvényeket pedig a rendszer paramétereivel együtt egy osztályba rendeztem, így a programot modulként importálva, pár soros szkriptekkel lehet használni. A könnyebb érthetőség érdekében a program minden végrehajtott lépésnél részletes parancssori üzenetekkel vezeti a felhasználót, valamint jelzi, hogy egy hibás deklaráció vagy egy kihagyott lépés pontosan melyik paranccsal orvosolható. A megannyi bemeneti paramétert (geometriai elrendezés, a felszín albedója, emisszivitása, az égitest alakja, stb) külön-külön is bármikor meg lehet változtatni a szkripten belül is, de egy adatfájlban is eltárolhatjuk őket, ami szintén segíti a használatot, a felhasználó számára rögtön világos, hogy milyen paramétereket, formailag hogyan kell átadni a modulnak.

Az egész kód, egy példa szkripttel és egy hozzá tartozó paraméterfájllal együtt megtalálható az [alábbi oldalon](#)³. A GitHub.com egy szoftvermegosztó platform, amely kiválóan alkalmas verziókezelésre. Ha feltöltött kódot továbbfejlesztjük, frissítjük, akkor ezt tételesen láthatóvá tehetjük az oldalon is, így mindenki számára követhetőek lesznek a változtatások, és mindig naprakészen tudja használni a programot. Dolgozatomban ezen `pyneatm` modul felépítését, működését, felhasználhatóságát, és hatékonyságát fogom tárgyalni.

³<https://github.com/kinyeaux/pyneatm>

2. NEATM gömbre

2.1. Geometriai elrendezés

A NEATM modellel egy földközeli aszteroida különböző paramétereit szeretnénk meghatározni, mint pl. az átmérő és az albedó. A program bemenő paramétereként ismerjük az adott méréshez tartozó geometriai konfiguráció jellemző szög- és távolságértékeit, az aszteroidához rögzített gömbi poláris koordináta-rendszerben. A teljes Nap-Föld-kisbolygó rendszer leírható 6 adattal: a Föld (továbbiakban: megfigyelő) azimuttszöge és deklinációja, a megfigyelő távolsága a kisbolygótól $(\theta_{obs}, \phi_{obs}, r_{obs})$, valamint a Nap ugyanezen adatai $(\theta_{\odot}, \phi_{\odot}, r_{\odot})$.

Emellett a kisbolygóra jellemző, eleinte ismeretlen adatokat paraméterként használjuk. Ilyen annak mérete (átmérője), spektrális emisszivitása, albedója, nyalábparamétere.

2.2. Nyalábparaméter

Az η nyalábparaméter ebben az egyszerű modellben, definíció szerint, a teljes termális fluxus többletének reciproka egy egyenletesen sugárzó gömbéhez képest nulla fázisszög esetén. A nyalábparaméter több effektust (pl. kráterezettség, forgás, hőtehetetlenség) ír le egyetlen, koncentrált paraméterben.

2.3. A felszín paraméterezése

Alapértelmezésben a kisbolygó felszínén felvesszünk egy Descartes-i rácsot, úgy, hogy a kisbolygó forgástengelye legyen a z tengely (ebben a koordináta-rendszerben adjuk meg a megfigyelő és a Nap helyzetét is). A kapott rács minden pontját egy deklináció (földrajzi szélesség, $\phi \in [0, 2\pi)$) és egy azimuttszög (földrajzi hosszúság, $\theta \in [0, \pi)$) jellemzi.

A polár- és Descartes-i koordináták közötti átváltás az alábbi egyenletek alapján történik:

$$x = R \cos \theta \cos \phi \quad (1)$$

$$y = R \sin \theta \cos \phi \quad (2)$$

$$z = R \sin \phi, \quad (3)$$

a felszínt pedig az

$$\frac{x^2}{R^2} + \frac{y^2}{R^2} + \frac{z^2}{R^2} = 1 \quad (4)$$

egyenlet írja le.

2.4. Szubszoláris hőmérséklet

Meghatározzuk az ún. szubszoláris hőmérsékletet, azon pont hőmérsékletét, amelyből nézve a Nap a zenitben látszik. Ezt az egyenletet a Stefan–Boltzmann-törvény átrendezésével kapjuk, feltételezve, hogy a felszín hőmérsékleti egyensúlyban van, és bolometrikus emisszivitása 1:

$$\sigma_{\text{SB}} T_{\text{ss}}^4 \eta \epsilon_{\text{bol}} = \frac{S_{\odot}(1 - A_{\text{Bond}})}{r_{\text{h}}^2} \quad (5)$$

ahol $S_{\odot}$ a napállandó, A_{Bond} a Bond albedó, r_{h} a test Naptól vett távolsága (csillagászati egységekben), ϵ_{bol} a bolometrikus emisszivitás, σ_{SB} a Stefan–Boltzmann állandó, és η a nyaláb-paraméter. Ebből a szubszoláris hőmérséklet:

$$T_{\text{ss}} = \left[\frac{S_{\odot}(1 - A_{\text{Bond}})}{r_{\text{h}}^2 \epsilon_{\text{bol}} \sigma_{\text{SB}} \eta} \right]^{1/4} \quad (6)$$

2.5. Beesési szögek, felszíni hőmérséklet

Minden rácsponthoz hozzárendelhetünk egy σ_{obs} és egy $\sigma_{\odot}$ szögértéket, mely megadja az adott ponthoz tartozó normális és a megfigyelő/Nap iránya által bezárt szöget. Ezután minden pont hőmérséklete megadható a szubszoláris hőmérséklet segítségével:

$$T_{\text{app}}(\theta, \phi) = T_{\text{ss}}(\theta, \phi) \cos^{1/4}(\sigma_{\odot}) \quad (7)$$

A test megvilágítatlan oldalán, azaz ahol $\sigma_{\odot} > \pi/2$, a felületi hőmérsékletet 2,7 K-re állítjuk, ez a kozmikus háttérsugárzáshoz tartozó hőmérséklet.

Fontos kiemelni, hogy az így kiszámolt hőmérsékletértékek nem a felszín valódi, hanem a „látszó” hőmérsékletét adják, abban az esetben, ha a felszín spektrális emisszivitása ϵ_{ν} hullámhossz-függő (erre utal a T_{app} , azaz *apparent temperature* elnevezés is).

Amennyiben ϵ_{ν} változik a hullámhossz függvényében, akkor ahhoz, hogy a valódi hőmérséklet-eloszlást meghatározzuk, először ki kell számolnunk a felszín bolometrikus emisszivitását, ismerve az $\epsilon(\nu)$ spektrális emisszivitást a frekvencia függvényében. Ehhez a spektrális emisszivitást súlyozzuk a Planck-függvénnyel (hiszen feltételezzük, hogy a spektrális emisszivitástól magától eltekintve a felszín feketetestként sugároz):

$$\epsilon_{\text{bol}}(T) = \frac{\int \epsilon_{\nu} \frac{2h\nu^3}{c^2} \left[\exp\left(\frac{h\nu}{kT}\right) - 1 \right]^{-1} d\nu}{\int \frac{2h\nu^3}{c^2} \left[\exp\left(\frac{h\nu}{kT}\right) - 1 \right]^{-1} d\nu} = \frac{\int \epsilon_{\nu} \nu^3 \left[\exp\left(\frac{h\nu}{kT}\right) - 1 \right]^{-1} d\nu}{\int \nu^3 \left[\exp\left(\frac{h\nu}{kT}\right) - 1 \right]^{-1} d\nu}. \quad (8)$$

Első közelítésben a spektrális emisszivitás hullámhosszfüggése a felszín jellemzője, és nem függ a hőmérséklettől, míg a bolometrikus emisszivitás igen (ld. a fenti integrált). Tudjuk, hogy egy

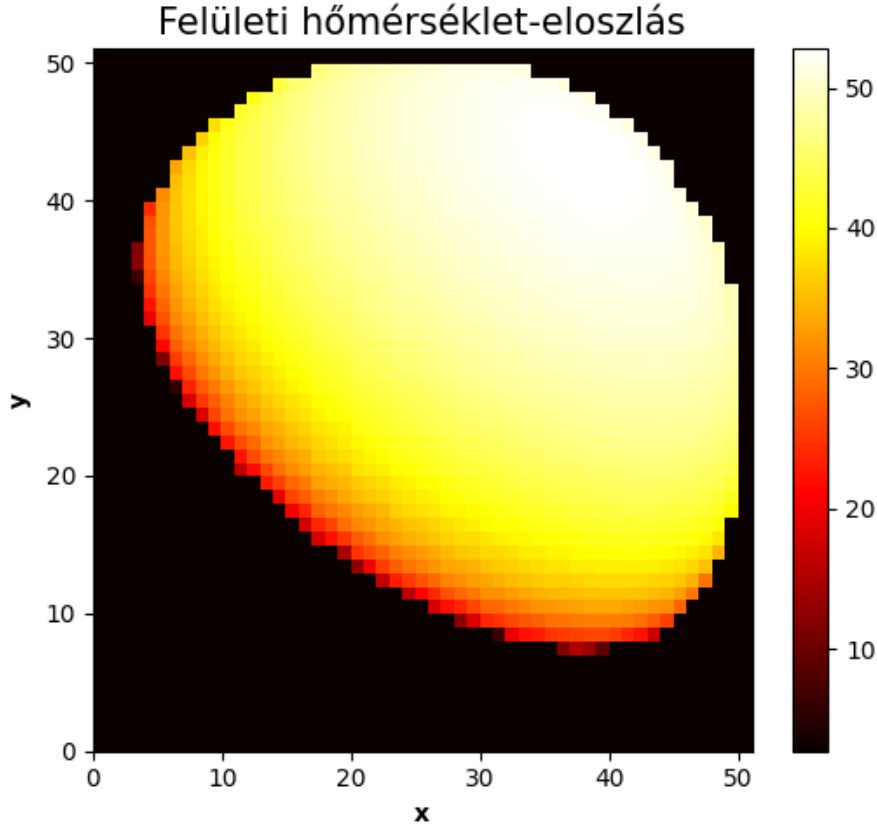
felületelem által kibocsátott hőmérsékleti sugárzás felületegységenkénti intenzitása a Stefan–Boltzmann-törvény alapján:

$$\sigma \epsilon_{\text{bol}} T_{\text{true}}^4 = \sigma T_{\text{app}}^4, \quad (9)$$

ahol T_{true} az adott felületelem valódi hőmérséklete. Ebből:

$$T_{\text{app}} = (\epsilon_{\text{bol}} T_{\text{true}}^4)^{1/4}. \quad (10)$$

Mivel ϵ_{bol} függ a hőmérséklettől, ezért a valódi hőmérséklet meghatározásához kiszámoljuk több különböző hőmérsékletre az $\epsilon_{\text{bol}}(T) \cdot T^4$ értéket, és interpolációval meghatározzuk, hogy mely T -re lesz egyenlő T_{app}^4 -el. Ezt a hőmérsékletet tekinthetjük az adott felületelem valódi hőmérsékletének. Példaként az 1. ábrán láthatjuk az így kapott hőmérséklet-eloszlást a $\theta_{\odot} = 55^\circ, \phi_{\odot} = 30^\circ, \theta_{\text{obs}} = 0^\circ, \phi_{\text{obs}} = 0^\circ$ esetben. Megjegyzendő, hogy a bolometrikus emisszivitás önkonzisztens kiszámítása nem része az általánosan használt NEATM modelleknek, az extra funkcióként került bele a kódomba.



1. ábra. Gömb alakú kisbolygó felületi hőmérséklet-eloszlása. A színskala értékei kelvinben értendők.

Miután meghatároztuk a kisbolygó hőmérséklet-eloszlását, a teljes fluxust egyszerűen meghatározhatjuk a Planck-törvény segítségével:

$$f_\nu = \int_0^\pi \int_0^{2\pi} \frac{2h}{c^2} \frac{\varepsilon_\nu \cdot \nu^3}{\exp\left(\frac{h\nu}{k_B \cdot T_{True}(\theta, \phi)}\right) - 1} \frac{R^2}{d^2} \cos(\theta) d\theta d\phi. \quad (11)$$

Ezek a tulajdonságok összevethetőek a megfigyelési eredményekkel, melyekből, a látható tartománybeli mérésekből ismert abszolút fényesség ismeretében egyszerre tudjuk meghatározni az égitest méretét és geometriai albedóját.

3. NEATM ellipszoidra

A standard NEATM modell gömbszerűnek tételezi fel az égitestet. A legnagyobb kis égitesteket kivéve az aszteroidák többségének alakja egyértelműen eltér egy gömbétől, ahogyan azt leginkább a kisbolygók fénygörbéiből ismerjük. Legegyszerűbb esetben egy ilyen égitest háromtengelyű ellipszoidként közelíthető. Az ilyen égitestek korrekt modellezéséhez a NEATM modellben is implementálni kell ezt az alakot, vagyis úgy módosítani a használt egyenleteket, hogy azok egy tetszőleges féltengelyarányú ellipszoid viselkedését is képesek legyenek leírni (MacLennan & Emery, 2018). Az elrendezés geometriája nagyon hasonló a fent tárgyalt esethez. Vegyük az r_0 skálahosszt, mely az ellipszoid nagytengelyének hossza km egységekben. A félnagytengelyek hosszának r_0 -val vett arányát jelöljük rendre $a \equiv 1 > b > c$ -vel.

3.1. A felszíni rács

A felszín egyenlete itt:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1. \quad (12)$$

Most is rögzítsünk a test szimmetria-centrumához egy polárkoordináta-rendszert, amelyben ismét megadhatjuk a Nap és a Föld helyzetét. A felületen vegyünk fel egy rácsot, az előzővel analóg módon, és tegyük fel, hogy a kisbolygó a c tengelye körül forog, $+\theta$ irányban. ($z \parallel c$) Mivel a felszín ez esetben nem gömb, ezért az egyes rácsponthoz tartozó felületnormálisok iránya eltér az adott pontba mutató helyvektor irányától. A felületnormálisok polár-koordinátáit az alábbi egyenletekkel határozhatjuk meg:

$$\tan\vartheta = \left(\frac{a}{b}\right) \tan\theta \quad (13)$$

$$\tan\varphi = \left(\frac{ab}{c}\right) \frac{\tan\phi}{\left(a^2 \sin^2 \theta + b^2 \cos^2 \theta\right)^{1/2}}, \quad (14)$$

ahol φ és ϑ az adott normálvektor gömbi polárkoordinátái.

3.2. Beesési szögek

Ismét kiszámítjuk a σ_{obs} és $\sigma_{\odot}$ értékeket is, melyekből azonban minden ponthoz 2 pár szögérték tartozik. Egyrészt az adott pontba mutató helyvektor és a Nap/megfigyelő irányvektora által, másrészt pedig a felületnormális és a Nap/megfigyelő irányvektora által bezárt szög. Ezeket a $\sigma_{\odot}$, σ_{obs} , $\sigma_{\odot}^{[n]}$ és $\sigma_{obs}^{[n]}$ szimbólumokkal jelöljük, ahol n a felületnormálisra utal:

$$\begin{cases} \cos \sigma_{\odot} = \sin \phi \sin \phi_{\odot} + \cos \phi \cos \phi_{\odot} \cos (\theta - \theta_{\odot}), \\ \cos \sigma_{obs} = \sin \phi \sin \phi_{obs} + \cos \phi \cos \phi_{obs} \cos (\theta - \theta_{obs}), \end{cases} \quad (15.a)$$

$$\begin{cases} \cos \sigma_{\odot}^{[n]} = \sin \phi^{[n]} \sin \phi_{\odot} + \cos \phi^{[n]} \cos \phi_{\odot} \cos (\theta^{[n]} - \theta_{\odot}), \\ \cos \sigma_{obs}^{[n]} = \sin \phi^{[n]} \sin \phi_{obs} + \cos \phi^{[n]} \cos \phi_{obs} \cos (\theta^{[n]} - \theta_{obs}). \end{cases} \quad (16.a)$$

$$(16.b)$$

3.3. Szubszoláris hőmérséklet

A szubszoláris hőmérsékletet az előzőekhez hasonlóan számíthatjuk:

$$T_{ss} = \left[\frac{S_{\odot}(1 - A_{Bond})}{r_h^2 \epsilon_{bol} \sigma_{SB} \eta} \right]^{1/4} \quad (17)$$

3.4. Felszíni hőmérséklet

Ebből kifejezhetjük egy tetszőleges rácspont hőmérsékletét is, azonban fontos különbség, hogy mivel a felületelem termális egyensúlyának felírásakor a beérkező sugárzási teljesítmény a Nap felületnormálissal bezárt szögétől függ, itt $\sigma_{\odot}$ helyett $\sigma_{\odot}^{[n]}$ -nel kell számolnunk, azaz:

$$T_{app}(\theta, \phi) = T_{ss} \cos^{1/4}(\sigma_{\odot}^{[n]}). \quad (18)$$

A megvilágítatlan oldal hőmérsékletét itt is 2,7 K-re állítjuk, azonban itt egy összetett feltétel alapján szűrjük a rácspontokat: $\sigma_{\odot}^{[n]} > \pi/2 \wedge \sigma_{\odot} > \pi/2$.

Ahogy azt a standard NEATM számolásnál csináltuk, itt is kiszámítjuk a bolometrikus emisszivitást, majd minden rácspontra a látszó hőmérsékletből a valódi hőmérsékletet, a fent leírt interpolációs módszerrel. (Ld. (8), (9), (10) egyenletek.)

3.5. Vetítés a fokális rácusra

Mivel egy ellipszoid teljes felszínére integrálni nagyon költséges számítási szempontból, más módszerhez kell folyamodnunk. Először meghatározzuk a polárkoordinátákból minden rácspont Descartes-i koordinátáját, az alábbi egyenleteket használva:

$$\begin{cases} x = a \cos \theta \cos \phi, \\ y = b \sin \theta \cos \phi, \\ z = c \sin \phi. \end{cases} \quad \begin{matrix} (19.a) \\ (19.b) \\ (19.c) \end{matrix}$$

Ezek után az eredeti $\{xyz\}$ koordináta-rendszer y tengelyét két forgatással (először z , majd y körül) a $[\theta_{obs}, \phi_{obs}]$ irányba állítjuk, hogy megkapjuk az észlelő által megfigyelt képet és a hozzá tartozó látszó hőmérséklet-eloszlást:

$$\begin{cases} x_1 = x \cos(\theta_{obs}) - y \sin(\theta_{obs}), \\ y_1 = x \sin(\theta_{obs}) + y \cos(\theta_{obs}), \\ z_1 = z, \end{cases} \quad \begin{matrix} (20.a) \\ (20.b) \\ (20.c) \end{matrix}$$

majd

$$\begin{cases} x_2 = x_1 \cos(\phi_{obs}) + z_1 \sin(\phi_{obs}), \\ y_2 = y_1, \\ z_2 = -x_1 \sin(\phi_{obs}) + z_1 \cos(\phi_{obs}). \end{cases} \quad \begin{matrix} (21.a) \\ (21.b) \\ (21.c) \end{matrix}$$

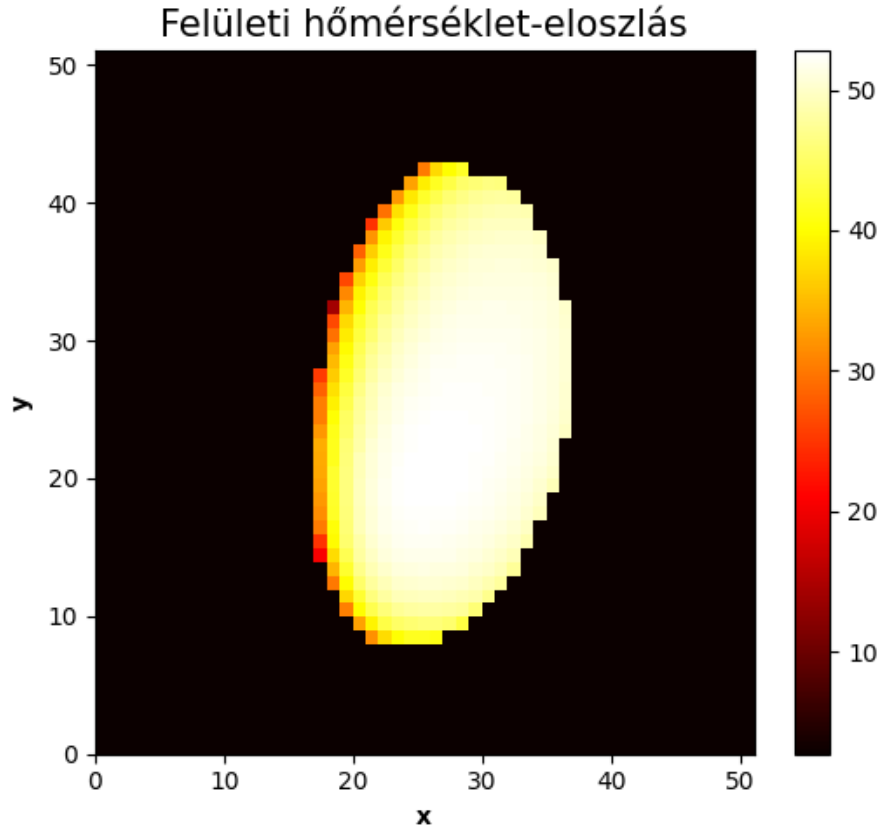
Az $x_2 z_2$ síkon definiálunk egy $x_p z_p$ rácsot (továbbiakban: fokális rács), és minden rácselemre kiszámoljuk a „domináns” T_2 hőmérsékletet (vesszük azon pontokhoz tartozó hőmérsékletek maximumát, melyek x_2 és z_2 koordinátája a $[x_p \pm dx, z_p \pm dz]$ téglalapba esik).

Így az f_ν spektrális fluxussűrűség meghatározható a fokális rácsra vett integrállal, mely véges sok rácspontra lévén az alábbi alakot ölti:

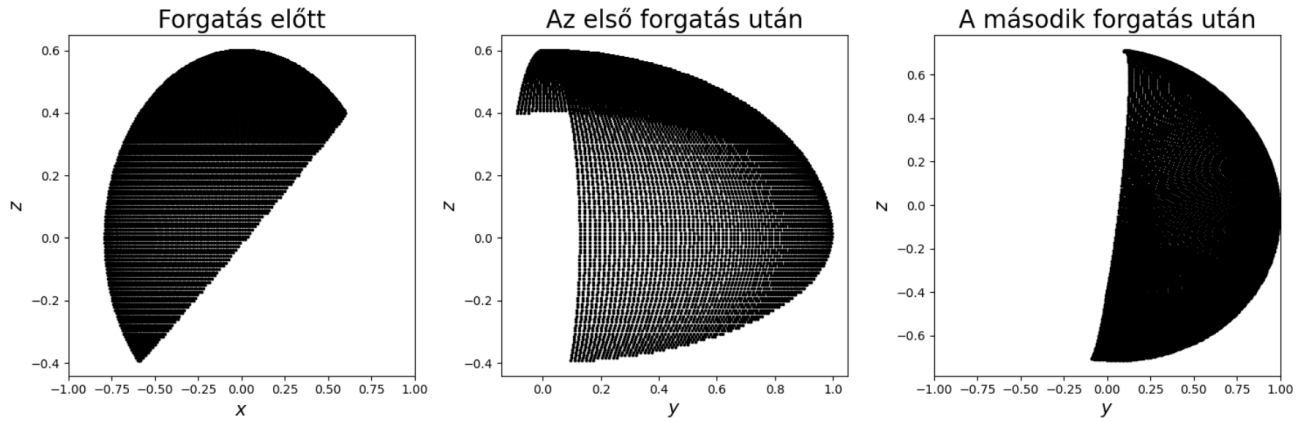
$$f_\nu(\lambda) = \frac{h\nu^3 \varepsilon(\lambda) r_0^2}{2c^2 \Delta^2} \sum_{ij} \left[\exp\left(\frac{h\nu}{k_b T_2^{ij}}\right) - 1 \right]^{-1} dx dz, \quad (22)$$

ahol h a Planck-állandó, k_b a Boltzmann-állandó, c a fénysebesség, ν a frekvencia (c/λ), dx és dz a rácselemek méretei.

A 2. ábrán láthatjuk egy $a = 1, b = 0.6, c = 0.4$ félnagy tengelyekkel rendelkező ellipszoid hőmérséklet-eloszlását a $\theta_\odot = 30^\circ, \phi_\odot = 60^\circ, \theta_{obs} = 25^\circ, \phi_{obs} = 15^\circ$ konfigurációban, míg a felületi rács vetületeit a fent használt koordináta-rendszerekben a 3. ábrán láthatjuk. Ezen a ponton természetesnek tűnik ellenőrizni, hogy a módszer konzisztens-e a standard számítással, melyet egyszerűen megtehetünk, ha az $a = b = c = 1$ választással futtatjuk le a kódot. (Az ellenőrzést elvégezve a két modell jó egyezést mutat, ld. a későbbi fejezetekben.)



2. ábra. Egy $a = 1, b = 0,6, c = 0,4$ félnagytengelyekkel rendelkező háromtengelyű ellipszoid hőmérséklet-eloszlása a $\theta_{\odot} = 30^{\circ}, \phi_{\odot} = 60^{\circ}, \theta_{obs} = 25^{\circ}, \phi_{obs} = 15^{\circ}$ konfigurációban. A színek a hőmérsékletet jelölik kelvinben.



3. ábra. A felületi rács vetületei az egyes forgatások elvégzése után.

4. NEATM poliéderre

Bizonyos esetekben ismerjük a vizsgált objektum konkrét alakját, ilyenkor érdemes figyelembe venni a felszínen levő krátereket, kitüremkedéseket. Ezt úgy tehetjük meg, hogy egy poliéderrel közelítjük a felszínt. Jelöljük ezen poliéder csúcsainak Descartes rendszerbeli helyvektorait $\mathbf{r}^{(i)}$ -vel.

4.1. A felszíni rács

A felszín adatait egy úgynevezett *wavefront* fájlban tároljuk, ebben az egyes pontok $r_k^{(i)}$ koordinátái, illetve az egyes lapokat alkotó csúcsok indexei, $\mathbf{F}^{(\alpha)} = (\alpha_1, \alpha_2, \alpha_3)$ szerepelnek. Ez azt jelenti, hogy az α -dik lapot az $(\mathbf{r}^{(\alpha_1)}, \mathbf{r}^{(\alpha_2)}, \mathbf{r}^{(\alpha_3)})$ pontok alkotják. A wavefront fájlokban a pontok indexei pozitív körüljárással szerepelnek az egyes lapok esetén. Emiatt a lapok súlypontjának ($\mathbf{f}^{(\alpha)}$, a továbbiakban: rácspont) és normálvektorának ($\mathbf{n}^{(\alpha)}$) kiszámításához a következő képleteket használhatjuk:

$$\mathbf{f}^{(\alpha)} = \frac{1}{3} \sum_{i=1}^3 \mathbf{r}^{(F_i^{(\alpha)})}, \quad (23)$$

$$f_k^{(\alpha)} = \frac{1}{3} \sum_{i=1}^3 r_k^{(F_i^{(\alpha)})}, \quad (24)$$

$$\mathbf{n}^{(\alpha)} = \frac{1}{2} (\mathbf{r}^{(F_2^{(\alpha)})} - \mathbf{r}^{(F_1^{(\alpha)})}) \times (\mathbf{r}^{(F_3^{(\alpha)})} - \mathbf{r}^{(F_1^{(\alpha)})}), \quad (25)$$

$$n_k^{(\alpha)} = \frac{1}{2} \sum_{i,j=1}^3 \epsilon_{kij} (r_i^{(F_2^{(\alpha)})} - r_i^{(F_1^{(\alpha)})}) (r_j^{(F_3^{(\alpha)})} - r_j^{(F_1^{(\alpha)})}). \quad (26)$$

Az egyes rácspontok és normálvektorok koordinátáit átválthatjuk gömbi polár-koordinátákba, az alábbi képletek szerint.

Rácspontokra:

$$\left\{ \begin{aligned} R^{(\alpha)} &= \sqrt{f_x^{(\alpha)2} + f_y^{(\alpha)2} + f_z^{(\alpha)2}}, & (27.a) \\ \theta^{(\alpha)} &= \arctan \left(\frac{f_y^{(\alpha)}}{f_x^{(\alpha)}} \right), & (27.b) \\ \phi^{(\alpha)} &= \arctan \left(\frac{f_z^{(\alpha)}}{\sqrt{f_x^{(\alpha)2} + f_y^{(\alpha)2}}} \right). & (27.c) \end{aligned} \right.$$

Felületnormálisokra:

$$\begin{cases} R^{[n](\alpha)} = \sqrt{n_x^{(\alpha)2} + n_y^{(\alpha)2} + n_z^{(\alpha)2}}, & (28.a) \\ \theta^{[n](\alpha)} = \arctan\left(\frac{n_y^{(\alpha)}}{n_x^{(\alpha)}}\right), & (28.b) \\ \phi^{[n](\alpha)} = \arctan\left(\frac{n_z^{(\alpha)}}{\sqrt{n_x^{(\alpha)2} + n_y^{(\alpha)2}}}\right) & (28.c) \end{cases}$$

4.2. Beesési szögek

Az előzőekhez hasonlóan minden rácspontra kiszámoljuk a $\sigma_{\odot}^{(\alpha)}$, $\sigma_{\text{obs}}^{(\alpha)}$, $\sigma_{\odot}^{[n](\alpha)}$ és $\sigma_{\text{obs}}^{[n](\alpha)}$ szögeket:

$$\begin{cases} \cos \sigma_{\odot}^{(\alpha)} = \sin \phi^{(\alpha)} \sin \phi_{\odot} + \cos \phi^{(\alpha)} \cos \phi_{\odot} \cos (\theta^{(\alpha)} - \theta_{\odot}), & (29.a) \\ \cos \sigma_{\text{obs}}^{(\alpha)} = \sin \phi^{(\alpha)} \sin \phi_{\text{obs}} + \cos \phi^{(\alpha)} \cos \phi_{\text{obs}} \cos (\theta^{(\alpha)} - \theta_{\text{obs}}), & (29.b) \\ \cos \sigma_{\odot}^{[n](\alpha)} = \sin \phi^{[n](\alpha)} \sin \phi_{\odot} + \cos \phi^{[n](\alpha)} \cos \phi_{\odot} \cos (\theta^{[n](\alpha)} - \theta_{\odot}), & (29.c) \\ \cos \sigma_{\text{obs}}^{[n](\alpha)} = \sin \phi^{[n](\alpha)} \sin \phi_{\text{obs}} + \cos \phi^{[n](\alpha)} \cos \phi_{\text{obs}} \cos (\theta^{[n](\alpha)} - \theta_{\text{obs}}). & (29.d) \end{cases}$$

4.3. Szubszoláris hőmérséklet

Habár ebben az esetben nem biztos, hogy van szubszoláris pont, vagy hogy pontosan egy van, mégis tudunk számolni egy szubszoláris hőmérséklet értéket. Ez egy olyan hipotetikus rácspont hőmérséklete lenne, amely normálvektora párhuzamos a Nap irányába mutató vektorral.

$$T_{\text{ss}} = \left[\frac{S_{\odot}(1 - A_{\text{Bond}})}{r_h^2 \epsilon_{\text{bol}} \sigma_{\text{SB}} \eta} \right]^{1/4}. \quad (30)$$

4.4. Felszíni hőmérséklet

Az ellipszoid modellel analóg módon itt is kiszámíthatjuk a felszín hőmérséklet-eloszlását. A látszó hőmérséklet:

$$T_{\text{app}}(\theta^{(\alpha)}, \phi^{(\alpha)}) = T_{\text{ss}} \cos^{1/4}(\sigma_{\odot}^{[n](\alpha)}), \quad (31)$$

$$T_{\text{app}}(\theta^{(\alpha)}, \phi^{(\alpha)}) = 2,7 \text{ K} \quad , \sigma_{\odot}^{[n](\alpha)} > \pi/2 \vee \sigma_{\odot}^{(\alpha)} > \pi/2. \quad (32)$$

A valódi hőmérséklet a (8), (9), (10) egyenletek alapján számolható.

4.5. Vetítés a fokális rácsra

Habár a spektrális fluxussűrűség kiszámításához itt nem az ellipszoidoknál bemutatott vetítéses módszert használjuk, mégis érdemes kiszámítani a vetített hőmérsékleteloszlást. A (21)-(29) egyenletek itt is működnek, a következő értékek mellett:

$$x^{(\alpha)} = f_x^{(\alpha)}, \quad y^{(\alpha)} = f_y^{(\alpha)}, \quad z^{(\alpha)} = f_z^{(\alpha)}. \quad (33)$$

4.6. A spektrális fluxussűrűség-eloszlás

A fluxussűrűség kiszámításához az alábbi integrált kell elvégeznünk:

$$f_\nu = \frac{1}{r_{\text{obs}}^2} \int_A B_\nu(T) \mathbf{n}_{\text{obs}} \cdot d\mathbf{A}, \quad (34)$$

ahol $\mathbf{n}_{\text{obs}}$ a megfigyelő irányába mutató egységvektor. A diszkrét rácsra (49) a következő alakot ölti:

$$f_\nu = \frac{1}{r_{\text{obs}}^2} \sum_{\alpha \in \Omega} B_\nu(T^{(\alpha)}) \mathbf{n}_{\text{obs}} \cdot \mathbf{n}^{(\alpha)}, \quad \Omega = \left\{ \alpha : \sigma_{\text{obs}}^{[n](\alpha)} \leq \frac{\pi}{2} \wedge \sigma_{\text{obs}}^{(\alpha)} \leq \frac{\pi}{2} \right\}, \quad (35)$$

itt $T^{(\alpha)}$ az α -adik pont hőmérséklete.

4.7. Méretskálák, a poliéderrács konfigurálása

Ahogy azt az ellipszoidrácsoknál már említettük, a hatékony számolás érdekében a felszíni rácsot normálni kell. Gömb esetén a sugár 1, míg ellipszoidnál a leghosszabb féltengely hossza 1: $1 = a > b > c$. Poliéder rácsok használatakor többféle méretezés is természetesnek tűnhet. Fontos, hogy a renormáláskor a pontok helyvektorait ugyanazzal a skalárral szorozzuk meg, hogy a méretarányok megmaradjanak. Mivel szeretnénk, hogy egy gömböt poliéderrel közelítve vissza- kapjuk a klasszikus számolás eredményét, a legnagyobb abszolút értékű helyvektor hosszával renormáljuk az egész rácsot. Könnyen belátható, hogy mivel a (27)-(30) egyenletek lineárisak, ezért ezt a renormálást elég a csúcsokon, $\mathbf{r}^{(i)}$ -n elvégezni.

Fontos észrevétel, hogy a fokális rács a $[-1, 1] \times [-1, 1]$ halmazon értelmezett, és a pontokat a megfigyelő irányába merőlegesen vetítjük. Így mivel az újonnan preparált ponthalmazban a legnagyobb abszolút értékű helyvektor hossza 1, ezért minden pont minden koordinátája a $[-1, 1]$ intervallumba esik, tehát az egész test belefér a képbe.

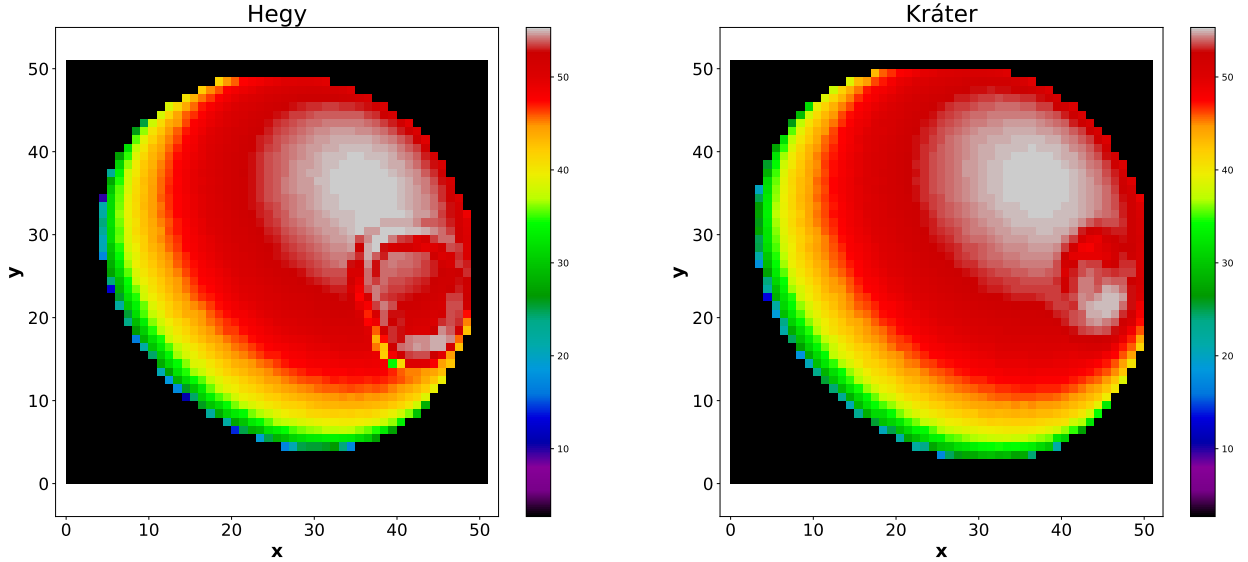
Látható, hogy a (45) és (46) egyenletekben Az integrál értékét is befolyásoltuk a normálással. Így az eredményeket vissza kell skáláznunk, az alábbiak szerint:

$$\mathbf{n}^{(\alpha)} = \frac{1}{2} r_0^2 (\hat{\mathbf{r}}^{(F_2^{(\alpha)})} - \hat{\mathbf{r}}^{(F_1^{(\alpha)})}) \times (\hat{\mathbf{r}}^{(F_3^{(\alpha)})} - \hat{\mathbf{r}}^{(F_1^{(\alpha)})}), \quad (36)$$

ahol $\hat{\mathbf{r}}^{(i)}$ az i -edik renormált helyvektor. Valójában $\mathbf{n}^{(\alpha)}$ helyett $\frac{\mathbf{n}^{(\alpha)}}{r_0^2} = \hat{\mathbf{n}}^{(\alpha)}$ -t számoljuk. Tehát a program által használt képlet:

$$f_\nu = \frac{r_0^2}{r_{\text{obs}}^2} \sum_{\alpha \in \Omega} B_\nu(T^{(\alpha)}) \mathbf{n}_{\text{obs}} \cdot \hat{\mathbf{n}}^{(\alpha)}, \quad \Omega = \left\{ \alpha : \sigma_{\text{obs}}^{[n](\alpha)} \leq \frac{\pi}{2} \wedge \sigma_{\text{obs}}^{(\alpha)} \leq \frac{\pi}{2} \right\}. \quad (37)$$

Mint fent említettem, a poliéder-modell úgynevezett wavefront fájlokat tud beolvasni. Ez a kiterjesztés a 3D modellezésben széleskörben elterjedt, talán az egyik leggyakrabban használt. Sokféle CAD (Computer Aided Design) programmal lehet ilyeneket létrehozni, én a Blender[©] ingyenes szoftvert használtam. Bemutatásként két alakmodellt gyártottam le, az elsőnél egy gömb alakú égitest felszínén egy hegyet, a másodiknál egy krátert hoztam létre, ezek láthatóak a 4. ábrán.

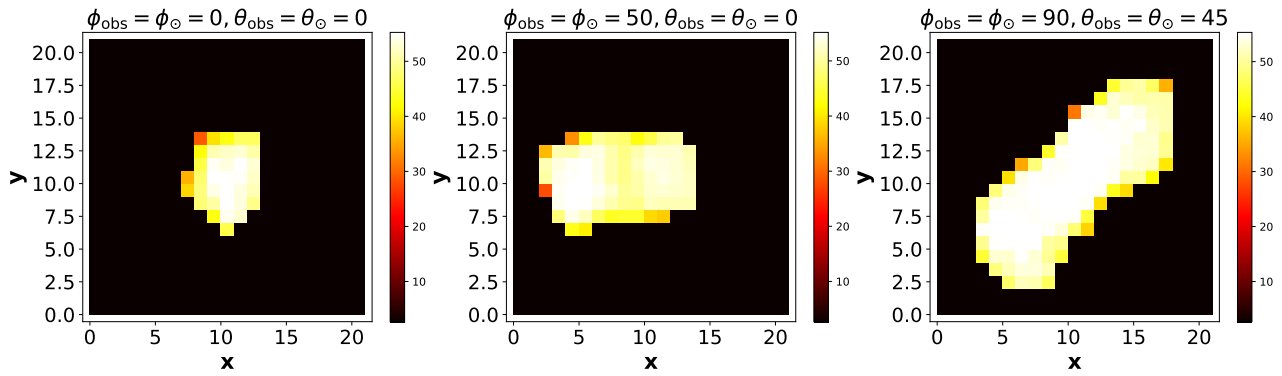


4. ábra. Két különböző poliéder-modell hőmérséklet-eloszlása. A színek itt is a hőmérsékletet jelölik kelvinben.

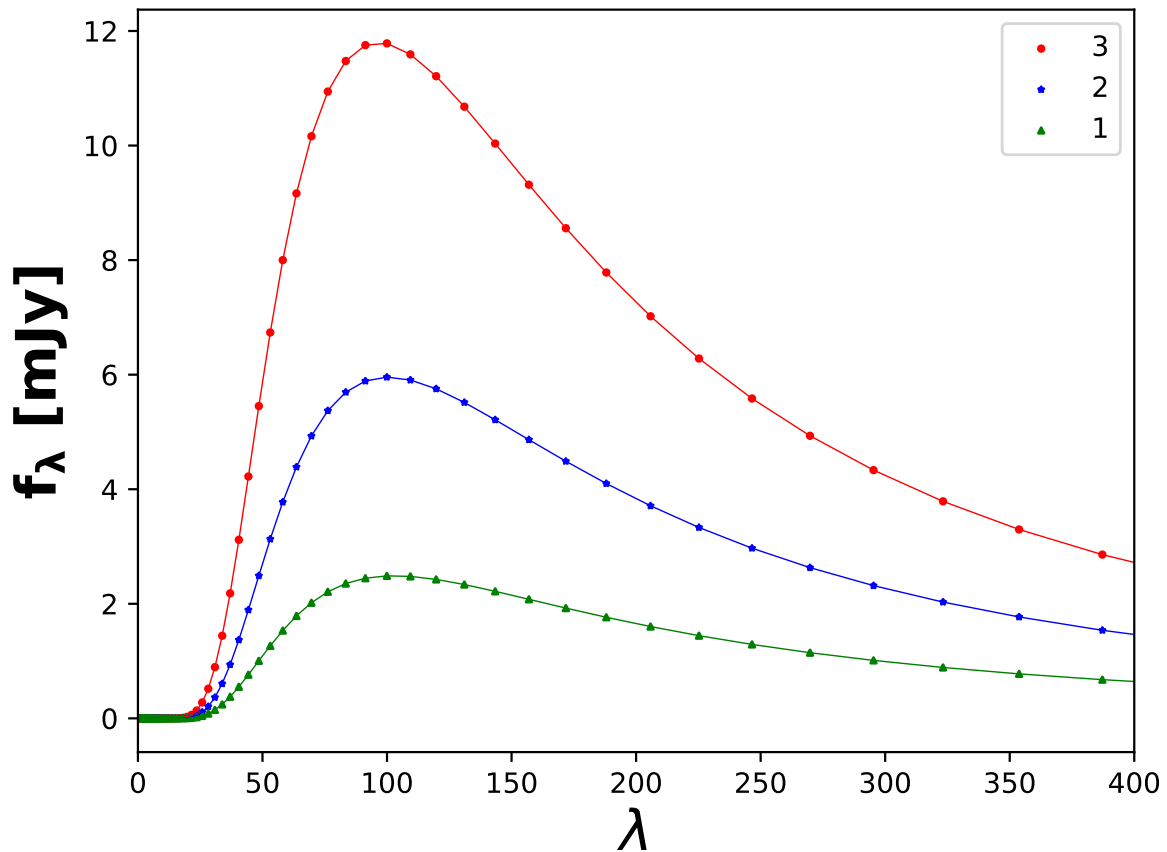
4.8. A (216) Kleopatra

Mint azt a bevezetőben is említettem, sok geometriai modell elérhető aszteroidákról, például a DAMIT⁴ adatbázisában. Ezek közül választottam egyet, a (216) Kleopatra kisbolygót (Shepard et al., 2018). Erre lefutattva a kódot az 5. és 6. ábrákon jól látszik, hogy különböző orientációkban más-más lesz az irradiancia, így ebben az esetben például nagyon pontatlan lenne a gömb közelítés.

⁴<https://astro.troja.mff.cuni.cz/projects/damit/>

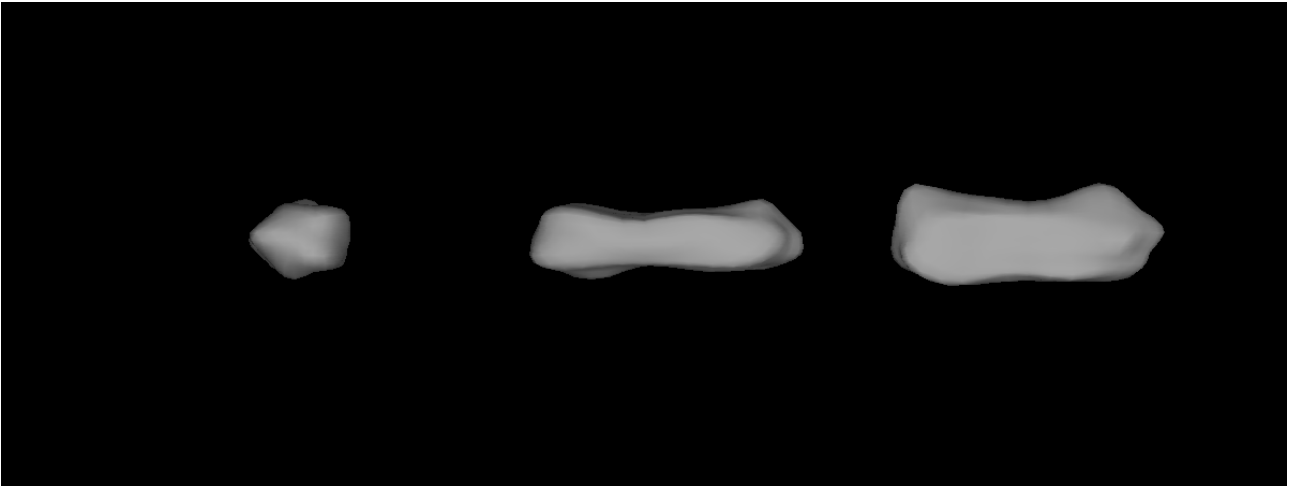


5. ábra. A (216) Kleopatra hőmérséklet-eloszlása a fókális rácson, különböző orientációkban. A színek itt is a hőmérsékletet jelölik kelvinben.



6. ábra. A (216) Kleopatra irradiancia görbéi az egyes esetekben.

A DAMIT adatbázisában megtalálható a 3D modell rajza is, ez a 7. ábrán látható.



7. ábra. A (216) Kleopatra alakmodellje (Shepard et al., 2018).

Ezekből látszik a poliédermodell széleskörű felhasználhatósága. A programhoz írtam egy rövid szkriptet is, amellyel a DAMIT adatbázisából letöltött fájlokat lehet `obj` formátumba konvertálni, így rögtön futtatható rajtuk a kód.

5. Különböző alakmodellek összehasonlítása

Az elkészült program három különböző alakmodellel, és ezen belül is különböző rácsokkal tud dolgozni. Az alábbiakban bemutatom a különböző modellek előnyeit és hátrányait, megvizsgálom, hogy bizonyos esetekben melyik ad pontosabb eredményt, milyen számítási idő mellett. Ez lehetőséget ad arra, hogy ha mérési adatok elemzésére használjuk a kódot, eldönthessük, milyen beállításokat érdemes választani.

5.1. Ellipszoid modell Descartes-i ráccsal

A standard NEATM modell gömb alakú égitesteket tud kezelni. Mivel ezekre tudunk analitikusan integrálni, ezért ezt vesszük etalonnak. A felszíni rács itt lehet Descartes-i vagy HealPix, mindkettő különböző felbontásokban. Ezzel szemben az ellipszoidok esetén csak úgy tudjuk meghatározni a fluxust, ha a felszíni rácsot levetítjük a fokális rácsra, ahogyan korábban tárgyaltuk. Itt rögtön két paraméterrel változtathatjuk az eredmény pontosságát. Egyrészt a felszíni rács (mely itt is ugyanolyan típusú lehet, mint gömb esetén) önmagában rendelkezik egy felbontással, amely befolyásolja az integrál eredményét. Másrészt a fokális rácsnak is van egy felbontása. Ezt a kettőt fontos megfelelően összehangolni. Ha a felszíni rács túl nagy felbontású a fokális rácséhoz képest, akkor feleslegesen számoltatjuk ki a pontosabb hőmérséklet-eloszlást a felszínen, hiszen az integrálás során átlagoljuk a fokális rácson egy pixelbe eső pontok hőmérsékletét. Ellenben ha túl kicsi a felszíni rács felbontása, akkor lehetnek olyan pixelek, melyekbe nem esik pont, így ott a hőmérséklet 0 lesz. Viszont ilyenkor az adott pixelhez tartozó térszögrészt is

kihagyjuk az integrálból, így összességében ugyanannyi hőmérséklet-járulék kevesebb térszögre jut, mely esetenként igen komoly hibát is tud okozni. Az ellipszoid-modell az $a = b = c = 1$ választás esetén visszaadja a gömb alakot, így könnyedén ellenőrizhetjük, hogy a számolásunk helytálló-e, illetve megvizsgálhatjuk, hogy hogyan kell hangolni a használt rácsoakat.

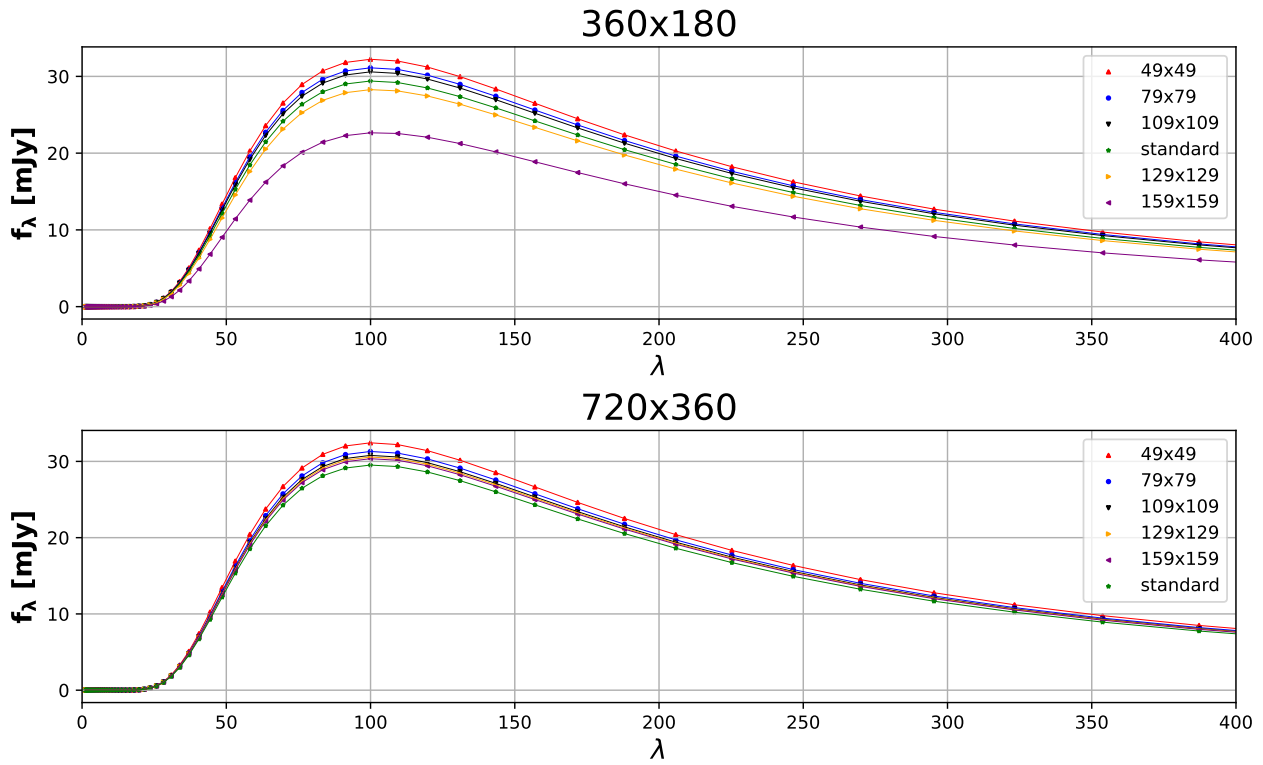
A számolás során a két legfontosabb adat a felszíni hőmérséklet és a spektrális fluxussűrűség-eloszlás. Megvizsgálhatjuk, hogyan függenek ezek a fokális rács felbontásától. A 8. ábrán látható egy 180x360-as felbontású (vagyis mindkét koordinátában fokenként felosztott) felszíni rácsra kiszámolt fluxussűrűség és különböző felbontású fokális rácsoak mellett vetítéssel számolt fluxussűrűség. Megfigyelhető, hogy a felbontást növelve egy ideig egyre pontosabb eredményeket kapunk, viszont tovább növelve a felbontást, elromlik a modell. Ennek oka a fentebb említett "lyukakból" ered. A fokális rács egy adott térszögnek felel meg. Ha túl nagy a felbontás, azaz az egy pixelhez tartozó térszög túl kicsi, akkor az integráláskor a lyukakat kihagyva az összfelület kisebb lesz, mint a valódi, ezért kapunk kisebb irradianciaértékeket.

Összevetve a 8. ábrát (360x180-as felbontás) a 9. ábrával, megbizonyosodhatunk, hogy a számolt fluxussűrűség valóban akkor kisebb, mint a standard modell eredménye, ha a hőmérséklet-eloszlásban lyukak jelennek meg.

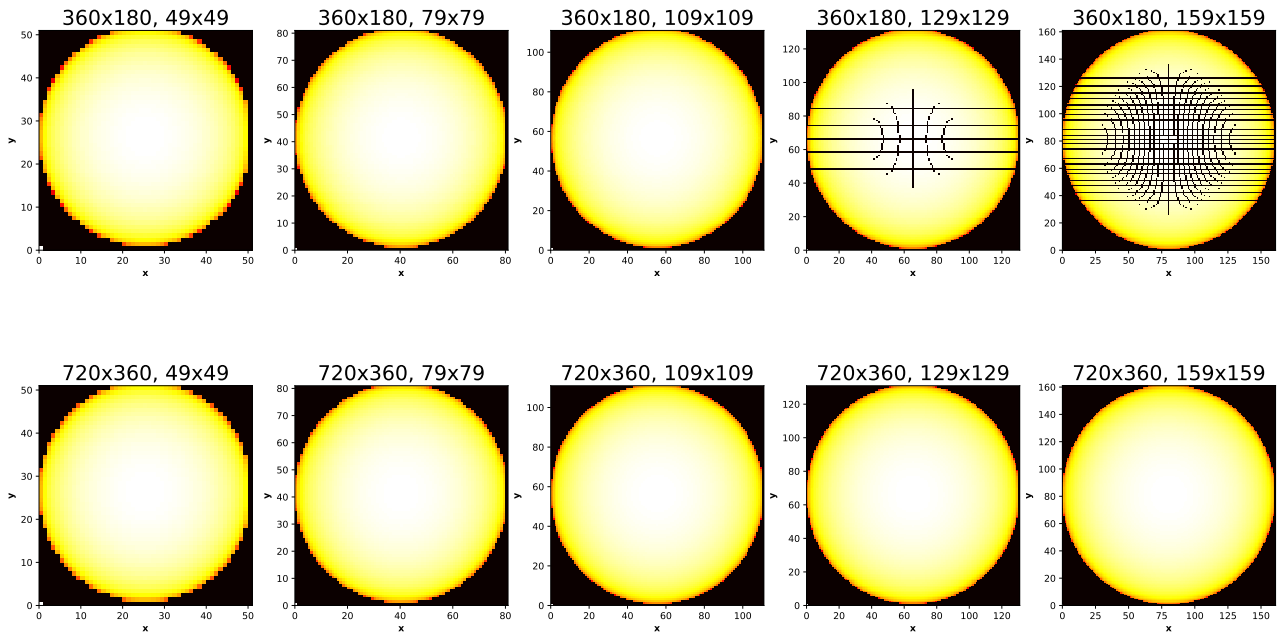
Mindezt összevetve azt állapíthatjuk meg, hogy Descartes-i rács használatakor a lehető legpontosabb eredmény érdekében a fokális rács felbontását addig kell növelni, míg a hőmérséklet-eloszlási képen az első üres pixel fel nem tűnik.

5.2. Ellipszoid modell HealPix rácscsal

Az előző részben Descartes-i rácsot használtunk, melynek előnye, hogy könnyű legyártani, és tájékozódni rajta. Azonban mivel a felosztásunk csak a polárkoordinátákban egyenletes, az egyes rácpontokhoz tartozó felületelemek területe erősen változik a hellyel (az egyenlítőhöz közelítve nagyobb, míg a pólusok felé haladva kisebb). Ez persze azt is jelenti, hogy a különböző pontok hőmérséklete különböző súllyal ad járulékot a fluxushoz, így ugyanakkora hiba az egyik vagy a másik pont hőmérsékletében különböző hibát okoz a végeredményben. Ennek kiküszöbölésére használhatjuk az ún. HealPix rácsot. (Hierarchical Equal Area isoLatitude Pixelation.) A rács úgy épül fel, hogy a gömbfelületet először 12 egyenlő részre bontjuk, majd minden iterációval a kapott felületelemeket további négy részre osztjuk. Ez látható a 12 ábrán. Könnyű kiszámolni, hogy a k -adik iteráció után $12 \cdot 4^{k-1}$ rácpontunk lesz. A program $k \in \{4, 5, 6\}$ esetekre számol, tehát az egyes esetekben a rácpontok száma rendre 768, 3072 és 12288. Megvizsgálhatjuk ebben az esetben is az egyes rácsoak pontosságát. Itt három különböző felszíni rács (H4, H5, H6) közül mindegyiket levetítjük különböző felbontású fokális rácsoakra, és kiszámoljuk a hőmérséklet-eloszlást, valamint a spektrális irradianciát. Összehasonlításakor etalonnak a 360x180-as felbontású Descartes-i rácsra vett integrált vesszük.



8. ábra. A spektrális fluxussűrűség-eloszlás 360x180-os felszíni rácson, különböző felbontású fókális rácson



9. ábra. A fókális rácra vetített hőmérséklet-eloszlás különböző felbontások mellett.

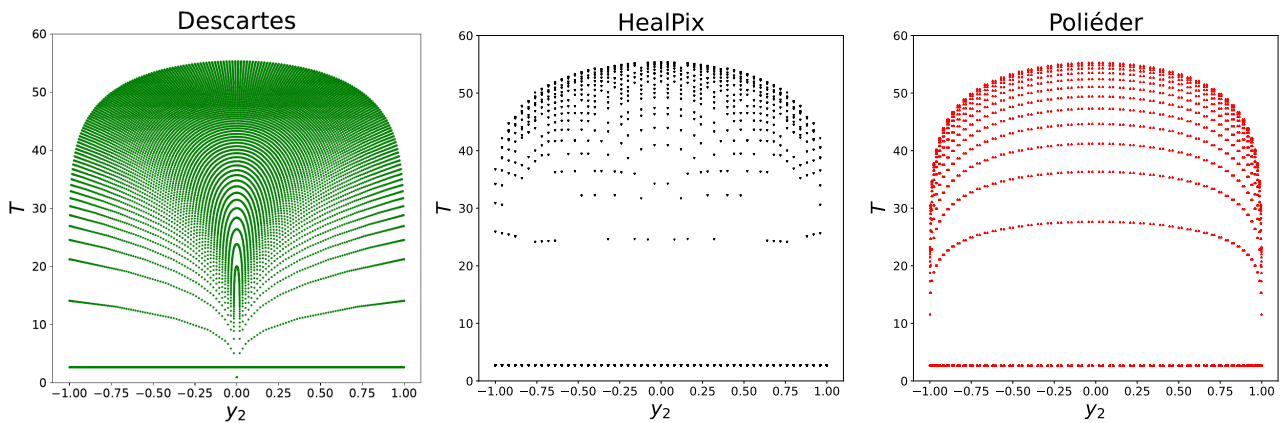
Ahogy az a 11 és a 13 ábrákon látható, itt is érvényes a megállapítás, miszerint amíg a fokális rács felbontása nem túl finom a felszíni rácshoz képest, a számolt irradiancia a valós értékhez konvergál, viszont amint túllépünk egy adott (a felszíni rács felbontásától függő) határt, a pontosság hirtelen elromlik.

5.3. Poliéder modell

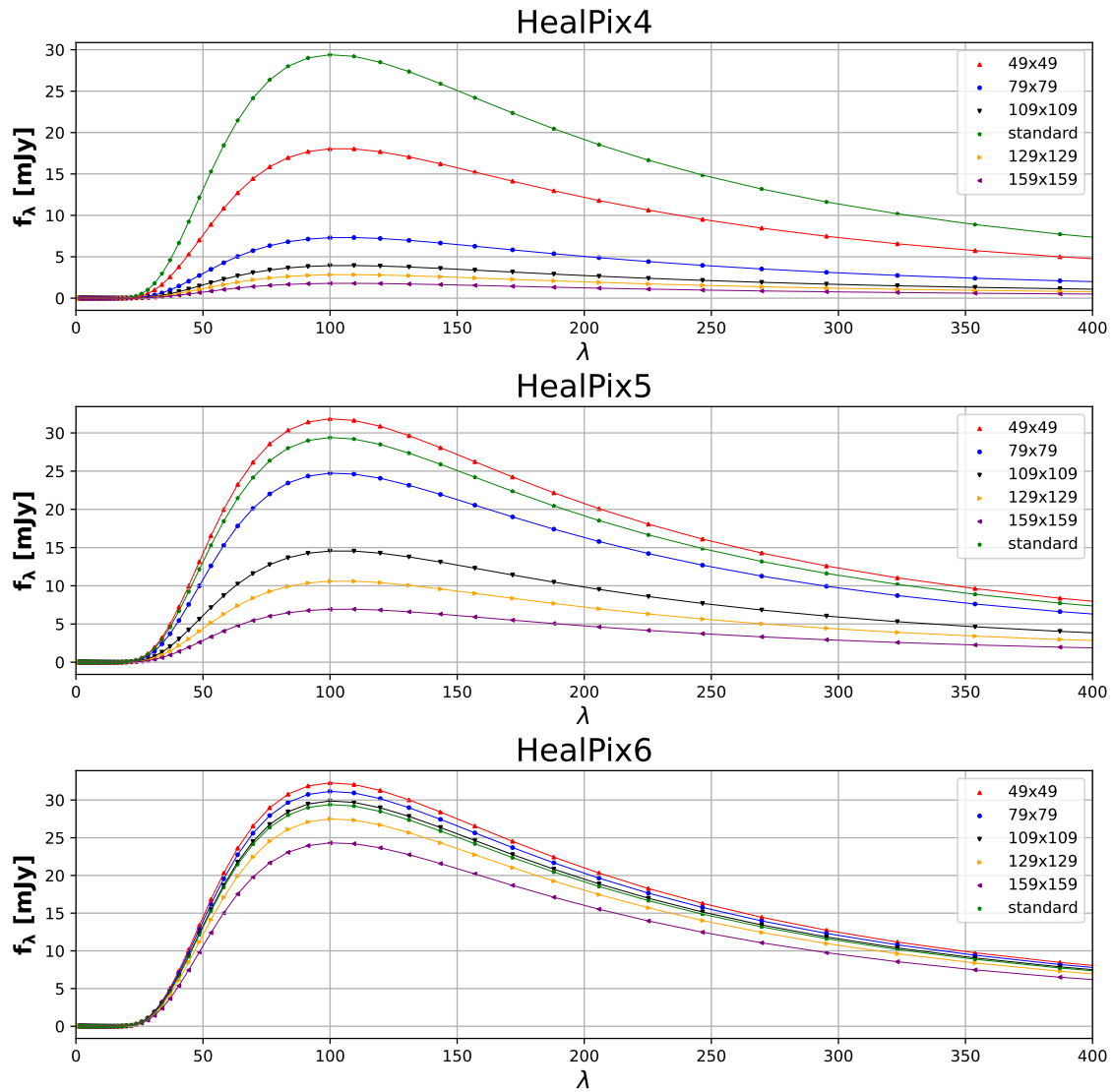
A poliéder modell esetében is ellenőrizhetjük a modellünk helyességét. Itt ugyan a fokális rács nem játszik szerepet az integrálásakor, a monokromatikus fluxussűrűséget kiszámíthatjuk több különböző lapszámú poliéder-közelítésre. Itt a felbontást egyetlen számmal jellemezzük, mégpedig a rácsot alkotó háromszöglapok számával. A 14 ábrán látható, hogy már kis felbontás mellett is nagy pontossággal visszkapjuk a felszíni rácstra integrált irradianciát, emiatt a poliéder rács nemcsak sokoldalúan használható, de gyors és alacsony számítási kapacitást igényel. Ennek az a magyarázata, hogy a descartes-i rácson a pólusok körül sokkal sűrűbben helyezkednek el a pontok, így ott feleslegesen sokat számolunk a poliéder- vagy a HealPix rácsokhoz képest.

Érdekes lehet még leellenőrizni, hogy egy háromtengelyű ellipszoidot poliéderrel közelítve vajon milyen eredményre jutunk, ezzel felmérhetnénk, hogy a HealPix rács egy ellipszoid felületén hatékonyabb-e, mint a Descartes-i.

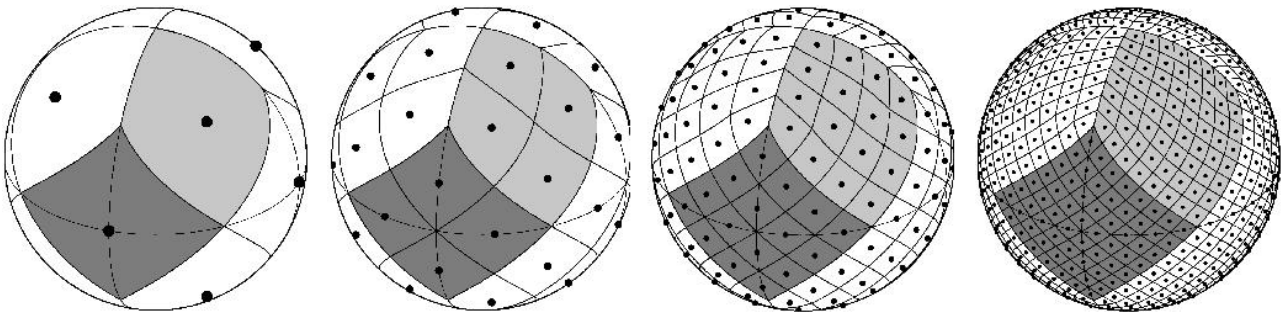
A 10. ábra mutatja, hogy a különböző alakmodellek hogyan közelítik a fokális rács hőmérséklet-eloszlását. Látható, hogy a pontok burkolója ugyanaz.



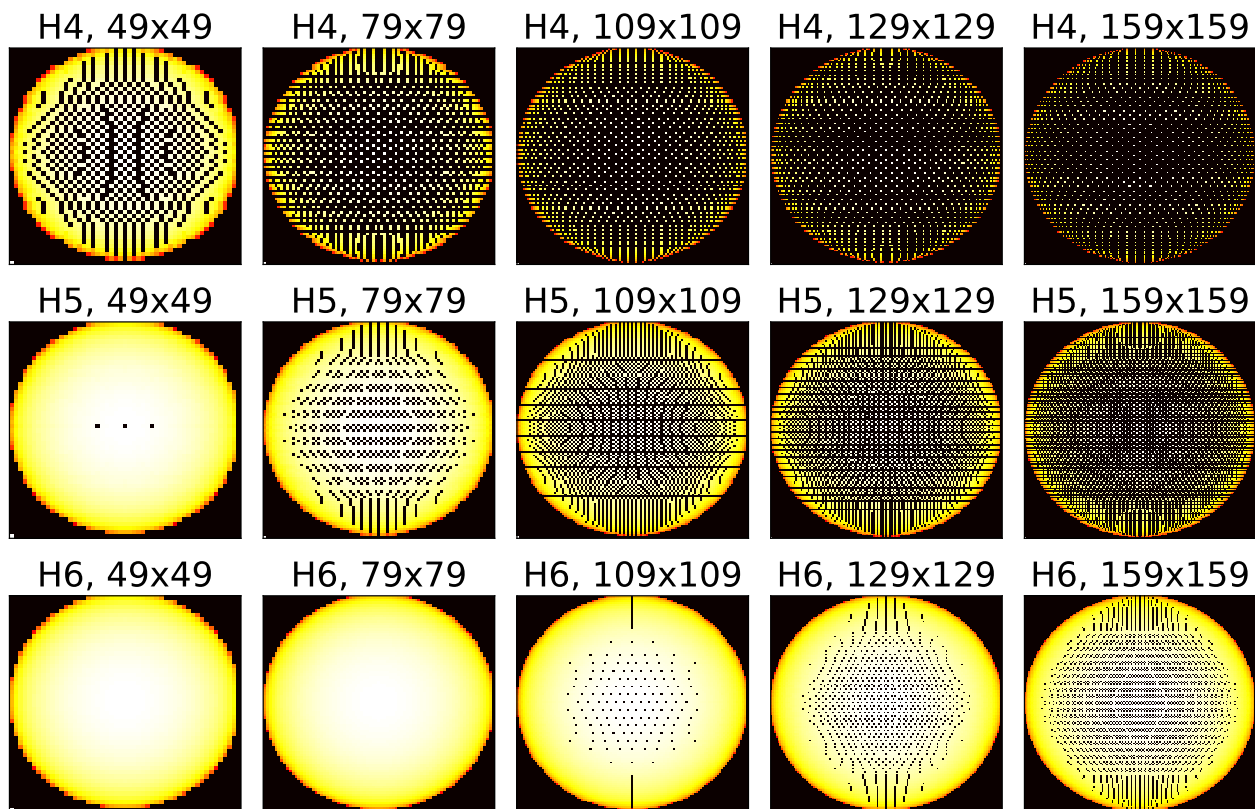
10. ábra. A fokális rács hőmérséklet-eloszlása a különböző modellek esetében.



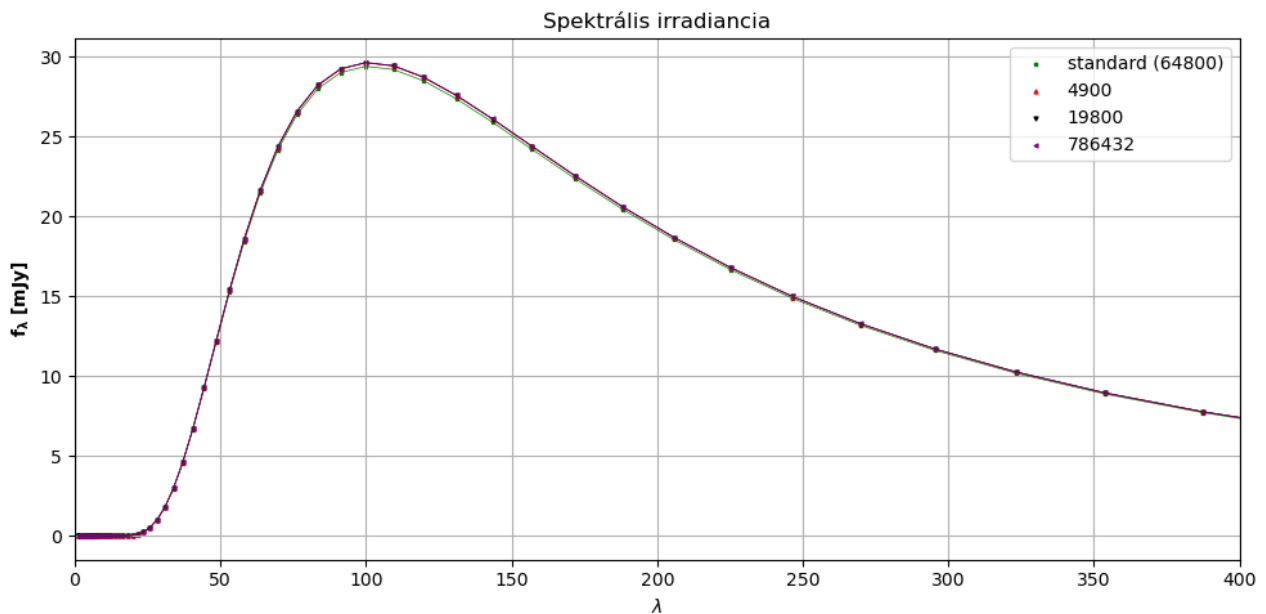
11. ábra. A rácsra integrált irradiancia különböző felbontások mellett.



12. ábra. A HealPix-rács szerkezete.



13. ábra. A hőmérséklet-eloszlás a fokális rácson különböző felbontásoknál.



14. ábra. Az irradiancia különböző felbontások mellett, poliéderrács esetén. Érdekes megfigyelni, hogy a standard descartes-i rács itt 64800 rácspontból áll, és a poliéderrács már 4900 rácspontnál is nagyon pontosan az eredetivel konzisztens eredmény adott.

6. Szíinkorrekcio

A fent leírt modellek mindegyike képes szíinkorrekcio számítására is, különböző űrtávcsövek esetén. A NEATM modell alapértelmezésben F_m monokromatikus fluxussűrűségeket szolgáltat egy adott modellre, a mérések ugyanakkor „sávbeli” F_i fluxussűrűségeket adnak. Mivel a fotometriai sáv kalibrálásához használt referencia spektrális energiasűrűség (pl. $\nu F_\nu = \text{konstans}$) nem egyezik meg a forrás valódi spektrális energiasűrűségével, a mért (sávbeli) fluxussűrűségeket szíinkorrigálni kell:

$$F_m = F_i / C_\lambda,$$

ahol C_λ a szíinkorrekcio tényező. Mivel C_λ spektrális energiaeloszlás függő, és a valódi spektrális energiaeloszlás meghatározásához C_λ -t ismerni kell, a spektrális energiaeloszlás, és ezzel párhuzamosan C_λ meghatározását iteratív módon szokták elvégezni. Ennek során egy kezdeti, tipikus hőmérsékletet tételezünk fel (pl. a szubszoláris hőmérséklet valamilyen átlagos albedót feltételezve), ennek alapján kiszámítjuk a szükséges C_λ szíinkorrekciokat a mérési sávokra, és ebből az F_m monokromatikus fluxussűrűségeket. Ezekhez az F_m -ekhez megkeressük a legjobban illeszkedő modellt, ami megadja a (monokromatikus) spektrális energiaeloszlás első iterációját. Ennek alapján új szíinkorrekciokat számolunk minden mérési sávra, amivel új monokromatikus fluxussűrűségeket számolunk, és új, legjobban illeszkedő modellt keresünk. Ezt addig ismételjük, amíg a két egymást követő lépésben kapott spektrális energiaeloszlások egy adott tolerancián belül meg nem egyeznek egymással (pl. 5%).

Az előbbi, iteratív módszer hátránya, hogy bár a felszín egyes pontjai kb. feketetesként sugároznak, azok integrált fluxussűrűsége ezen felületelemenkénti fluxussűrűségek összege lesz, ami nem egy jól meghatározott, szokásos spektrális energiaeloszlás, amire pl. az űreszközök infravörös műszereinek hullámhosszaira a szíinkorrekcio ki szokták számolni (pl. feketetest, vagy módosított feketetest). A szíinkorrekciokat minden egyes szűrőre, minden egyes iteratív lépésben a spektrális energiaeloszlás és a szűrő áteresztési függvényének konvolúciójával kellene kiszámolni, ami rendkívül nagy mértékben bonyolítaná az eljárást. Ehelyett közelítő (általában feketetest) szíinkorrekciokat szoktak alkalmazni, amik pl. a távoli infravörös tartomány szűrőire és detektoraira kb. $\leq 10\%$ -os hibát okoznak egy tipikus esetben a végső monokromatikus fluxussűrűségben; bizonyos speciális esetekben (pl. több, különböző albedójú felszín keveredése) azonban az ebből származó hiba jóval nagyobb is lehet.

Az általam alkalmazott szíinkorrekcio-számolás a modell keretein belül hibamentes, hiszen az adott felületelem hőmérsékletének megfelelő monokromatikus fluxussűrűségből számolja ki a mérhető, sávbeli fluxussűrűségeket a megfelelő egyedi szíinkorrekciokkal, majd az így kapott felület-integrált sávbeli fluxussűrűségek és a felület-integrált monokromatikus fluxussűrűségek aránya megadja az integrált fluxussűrűségekre vonatkozó szíinkorrekcio értékét.

7. A program felépítése

Miután a számolások elméleti hátterét tárgyaltuk, most nézzük meg a kód tényleges felépítését. A `python` nyelv `numpy` modulát használjuk. Létrehozunk egy `neatm` osztályt, ebben tároljuk az összes felhasznált információt és a számoláshoz szükséges függvényeket. Néhány univerzális konstans (pl. Napállandó, Boltzmann-állandó, etc.) a kód elején definiálunk, ezek nem részei az osztálynak.

7.1. Segédfüggvények

A `neatm` osztályban meghívunk néhány előre definiált külső függvényt, a jobb átláthatóság érdekében. Ezeket alább tételesen felsoroltam:

- `quadrant(angle)`: Bemeneti argumentuma egy radiánban mért szög, visszatérési értéke 1, ha $\pi/2 \leq \text{angle} \leq \pi$; -1, ha $-\pi \leq \text{angle} \leq -\pi/2$, és 0 egyébként. Tulajdonképpen azt határozza meg, hogy melyik síknegyedbe esik `angle`.
- `atanx(value, angle)`: Két bemeneti argumentuma van, melyekre teljesül a következő egyenlet: `value = tan(angle)`. Visszatérési értéke egy α szög, melyre $\tan \alpha = \text{value}$ és `quadrant( $\alpha$ ) = quadrant(angle)`. Erre azért van szükség, mert a `python` beépített `arctangens` függvénye csak a $[-\pi/2, \pi/2]$ tartományban ad visszatérési értéket.
- `B(nu, T)`: tulajdonképpen a Planck-függvény. `nu` egy `numpy array` típusú változó, így egyszerre több frekvenciaértékre ki tudjuk számolni a spektrális radianciát. Emellett a függvény elkerüli a túlsordulást is: azon `nu` értékekre, melyekre $\frac{h \cdot \text{nu}}{k_B T} \geq 100$, nem értékeli ki az exponenciális kifejezést, ilyenkor visszatérési értéke 0.
- `cc(T, band)`: A színkorrekciót kezelő függvény. A `band` változó egy string, mely megadja, milyen távcsőre számolunk színkorrekciót. A programba eddig beépített távcsövek (illetve kamerák): MIPS24, MIPS70, MIPS160, PACS70, PACS100, PACS160, SPIRE250, SPIRE350, SPIRE500, ALMA6, ALMA7, ALMA8. A függvényben el van tárolva minden távcsőhöz és kamerához egy előre kiszámolt színkorrekció-hőmérséklet függvény, ezt interpolálja, majd kiértékeli a `T` argumentumban.

7.2. Vektorok beolvasása és kezelése

A poliéder-modellben sokszor kell háromdimenziós vektorokkal számolni. Ehhez létrehoztam egy `vector3d` osztályt, melynek 3 változója van: `x`, `y`, `z` (a koordináták). Emellett van néhány belső függvénye: összeadás, kivonás, skalárral való szorzás, skalárszorzat, vektoriális szorzat és abszolút érték. Az utolsó kivételével mindet `operator overflow`-val vezettem be, a könnyebb átláthatóság érdekében. Ahhoz, hogy ilyen vektorokat le tudjunk gyártani wavefront fájlokból,

egy egyszerű beolvasófüggvényt, a `readobj(filename)`-t definiáltam. Ez a hagyományos `.obj` formátumú fájlokat tudja értelmezni, vagyis a "v" kezdetű sorokból beolvas 3 `float` típusú valós számot (az adott csúcs koordinátái), a "f" kezdetűekből pedig három `int` típusú egész számot (az adott lapot alkotó 3 pont indexe). (Az egyes sorokban a koordináták mellett még 2, számunkra irreleváns számot is tartalmaznak a fájlokat, ezeket egyszerűen figyelmen kívül hagyjuk).

A beolvasott koordináták és a lapokhoz tartalmazó indexek segítségével rögtön ki tudjuk számítani a fent tárgyalt felületvektorokat és lapközéppontokat. Ehhez először megkeressük a legnagyobb hosszúságú vektort, ennek abszolút értékével leosztjuk az összes koordinátát. (Még a lapközéppontok és a felületnormálisok kiszámítása előtt). Ezután a megfelelő koordináták átlagolásával megkeressük a ponthalmaz súlypontját, melyet koordinátánként kivonunk minden pontból. Így már a számunkra releváns, $\hat{\mathbf{f}}^{(\alpha)}$ és $\hat{\mathbf{n}}^{(\alpha)}$ vektorokat kapjuk meg.

Bizonyos alkalmazások során szokás a lapfelületvektorok helyett az egyes csúcsokban definiálni alternatív "normálisokat". Ezek elegendően sok pont esetén kiválóan helyettesítik a lapfelületvektorokat, így kisebb számítási kapacitást igényelnek ugyanakkora objektumok rendereléséhez. Megjegyzendő, hogy esetünkben ezek használata nem lehetséges, mivel nem csak a lapok iránya, de azok nagysága is fontos a számolások során, és sajnos ezek az alternatív normálisok mind egységnyi hosszúak.

A létrehozott vektorok mellett eltároljuk magukat a koordinátákat is, hogy ne kelljen külön kiszámolni őket újra a fokális vetítéskor.

7.3. A paraméterek beolvasása

A felhasznált paramétereket, mint fent említettem, egy szövegfájlban tároljuk. Ennek két előnye van: egyrészt a felhasználó megtudhatja, milyen adatokat milyen formátumban kell megadni bemeneti paraméterként, illetve egy adott konfigurációt könnyedén el tudunk menteni. Alább látható egy ilyen fájl szerkezete a 15. ábrán.

```
#name of the object:
name = HAUMEA

#surface grid type: (H4, H5, H6, cartesian or polyhedron)
healpixmap = polyhedron

#polyhedron grid file (.obj):
polyhedron source = sphere_with_hill.obj

#distance from Sun [AU]:
r_helio = 50.7

#distance from Earth [AU]:
r_geo = 50.0

#bolometric albedo
A_bol = 0.1

#beaming parameter:
eta = 1.0

pV = 0.51

hv = 0.0

q_phase = 0.5

eps = 0.9

#size of the major axis:
scale = 1008

#semi axes 1 >= a > b > c >= 0:
a = 1
b = 1
c = 1

#polar coordinates of the Sun and the Earth [deg]:
theta_sun = 130.0
phi_sun = 235.0
theta_obs = 180.0
phi_obs = 215.0

#spectral emissivity function:
#number of points in the interpolation:
nn_wvl = 81
#wavelengths and the corresponding emissivity values:
eps_spec_wvls0 = 1.0, 24.0, 70.0, 100.0, 160.0, 250.0, 350.0, 500.0, 1500.0
eps_spec_vals = 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9
```

15. ábra. Paraméterezés.

Ezeket a fájlokat a `getdatafromstr(string)` függvénnyel dolgozzuk fel. Semmi mást nem csinál, mint soronként beolvassa a fájlt, majd megnézi az első néhány karaktert, és ha megtalál egy kulcsszót, akkor az utána lévő adatot elmenti a megfelelő változóba.

7.4. A neatm osztály

A program használatakor létrehozunk egy `neatm` típusú objektumot, mely minden információt tartalmaz a vizsgált aszteroidáról. Az osztálynak sok változója van, melyeket olyan sorrendben hozunk létre, ahogyan az elméleti bevezetőben láttuk. Alább felsorolom az összes belső függvényt, illetve azokat a változókat, melyeket ezen függvények létrehoznak:

- `__init__(self)`: Ezzel a függvénnyel deklarálhatunk egy `neatm` objektumot. Az előző pontban tárgyalt fájlból beolvassa a megfelelő változókat a `getdatafromstr()` függvény segítségével, ezután egy rövid parancssori üzenettel jelzi a felhasználó felé, hogy sikeresen létrehozta a kért objektumot, például: `"> new neatm object created, name: HAUMEA"`.
- Ezután a program kiszámítja a szubszoláris hőmérsékletet és a Nap illetve a megfigyelő irányába mutató egységvektorok koordinátáit.
Az `nn_wvl` változó megadja, hogy hány pontban szeretnénk kiszámítani a spektrális radianciát. Az `eps_spec_wvls0` és az `eps_spec_vals` változókkal definiált spektrális emisszivitást interpolálja `nn_wvl`, logaritmikusan elhelyezett pontban az `eps_spec_wvls0` által definiált tartományon.
- `help(self)`: Egy rövid útmutató a kódhoz, tartalmazza a legtöbb függvény leírását, abban a sorrendben, ahogyan azokat meg kell hívni:

```
>> HAUMEA is a NEATM object that represents a Near-Earth asteroid.
Here are the different commands one can use in this model:
>>> HAUMEA.create_grid() : Creates a grid on the surface of the asteroid, which can
be either cartesian or HEALPix (4, 5 or 6). The healpixmod variable decides which grid
is to be used (default: cartesian).
>>> HAUMEA.set_healpixmod(hpm) : Takes a string (hpm) as an argument,
which can be 'H4', 'H5', 'H6' or 'cartesian'. Sets the healpixmod of the object.
>>> HAUMEA.plot_theta_phi() : Plots theta vs phi for each gridpoint.
>>> HAUMEA.spectral_emissivity(NN_WVL, EPS_SPEC_WVLS0, EPS_SPEC_VALS) : With EPS_SPE
C_VALS vs EPS_SPEC_WVLS0 numpy arrays, one can give the spectral emissivity function o
f the body. Then the program interpolates this function to NN_WVL logarithmically spac
ed points. EPS_SPEC_VALS and EPS_SPEC_WVLS0 must be the same size.
>>> HAUMEA.calculate_T() : Calculates the surface temperature of each gridpoint.
>>> HAUMEA.project_to_focal_grid(N) : Projects the surface grid to the focal plane.
With two rotations, the y axis of the coordinate system is pointing towards the obser
ver. Then creates a N×N grid on the x-z plane and projects each surface grid point to
it.
>>> HAUMEA.mono_flux_density() : Integrates the monochromatic flux density for each f
requency for the whole focal grid.
>>> HAUMEA.ib_flux_density() : Integrates the in-band flux density for each given ins
trument and corresponding center wavelengths for the whole focal grid.
HAUMEA.f_nu_ib stores the in-band flux densities with color correction, while HAUMEA.f
_nu_mc stores the monochromatic flux densities without color correction, for the same
wavelengths.
```

- `set_healpixmod(self, hpm)`: átállítja a felszíni rács típusát a `hpm` változó szerinti értékre ("cartesian", "H4", "H5", "H6" vagy polyhedron).

- `create_grid(self)`: Létrehozza a kívánt felszíni rácsot, majd kiszámolja a fent definiált $\sigma_{\odot}$ és σ_{obs} értékeket. Egy rövid parancssori üzenettel visszajelzi a felhasználónak az eredményt: "» Created "cartesian" surface grid for "HAUMEA" object.
Calculated the surface normal coordinates and the sigma angle values. "
- `spectral_emissivity(self, NN_WWL, EPS_SPEC_WVLS0, EPS_SPEC_VALS)`:
kiszámítja a spektrális emmisszivitást az `__init__()` függvénnyel azonos módon, de az argumentumként megadott tartományra. Itt is kapunk visszajelzést:
"» Refreshed the spectral emissivity function of the HAUMEA object,
interpolated for 81 logarithmically spaced frequencies and wavelengths.
Don't forget to recalculate the surface temperature."
- `calculate_T(self)`: Kiszámítja a felszíni hőmérséklet-eloszlást.
"» Calculated surface temperature for HAUMEA object."
- `project_to_focal_grid(self, N_PROJ_GRID=49)`: Levetíti az imént kiszámolt hőmérséklet-eloszlást egy `N_PROJ_GRID`×`N_PROJ_GRID`-es fokális rácusra.
"» Projected the surface grid to the focal plane (HAUMEA.T2 is the temperature distribution on the focal grid).
The focal grid has 49×49 gridpoints."
- `mono_flux_density(self)`: A fent interpolált hullámhossztartományra kiszámítja a spektrális irradianciát.
"» Calculated monochromatic flux density (HAUMEA.f_nu_t2)."
- `ib_flux_density(self, band_ib)`: A `band_ib` argumentumban (amely egy lista) tárolt távcsövekre számítja ki az irradianciaértékeket.
"» Calculated in-band flux density (HAUMEA.f_nu_ib) and the corresponding monochromatic flux density (HAUMEA.f_nu_mc) for the given instruments and center wavelengths."
- `plot_theta_phi(self)`, `plot_T(self)`, `plot_f_mono(self)`: Ábrázolja a felszíni rácsot, a hőmérséklet-eloszlást a fokális rácson illetve a spektrális irradianciát.

Annak érdekében, hogy a felsorolt függvényeket megfelelő sorrendben hívja meg a felhasználó, és ne hagyjon ki egy lépést sem a számolás során, a GitHub linken egy példaszkript is megtalálható, mely a 16. ábrán is látható.

```
#IMPORTING MODULES
from math import *
import numpy as np
from astropy.io import fits
import matplotlib.pyplot as plt
from scipy import interpolate, integrate
from neatm_ell_4gh_v8 import neatm
from tqdm import tqdm
import warnings
warnings.filterwarnings("ignore")

haumea = neatm("parameters.dat")

haumea.help()

haumea.create_grid()
haumea.calculate_T()
haumea.project_to_focal_grid(49)
haumea.mono_flux_density()
haumea.plot_T()
haumea.plot_f_mono()
```

16. ábra. Példaszkript.

8. Összefoglalás

Dolgozatomban bemutattam, hogyan épül fel és működik az a nyílt forráskódú python program, amelyet kutatásom alatt fejlesztettem. A `pyneatm` aszteroidák termális modelljeit tudja számolni. Három különböző alakmodellel dolgozik, valamint tartalmaz színtkorrekciót és kezelni tud nem konstans spektrális emisszivitást is. A különböző esetekben kiszámolt hőmérséklet- és irradianciaértékek azt mutatják, hogy az egyes alakmodellek egymással konzisztensek. Láthattuk, hogy a kód könnyen felhasználható és továbbfejleszthető, így nagyszerű alapja lehet további, komplexebb asztrofizikai modelleknek.

9. Köszönetnyilvánítás

Az, hogy a jelen dolgozat elkészülhetett, több ember alázatos munkájának köszönhető.

Szeretném megköszönni témavezetőimnek, Kalup Csillának és Kiss Csabának, azt a sok időt és energiát, amelyet saját kutatásuk mellett rám fordítottak. Elsőéves egyetemistaként kezdtem el foglalkozni a témával, és ők már az első perctől fogva egyenlő félként kezeltek, bátorító szavaik nagy motivációt jelentettek.

A 2022-2023-as tanév során 10 hónapig a Csillagászati és Földtudományi Kutatóközpont demonstrátori programjában vehettem részt, ennek keretei közt heti rendszerességgel tudtam személyesen találkozni a témavezetőimmal az intézmény falai között. Ez a munkakörnyezet alapvető jelentőséggel bírt a kutatásom előrehaladása során, és meghatározó szakmai élmény volt számomra. Köszönöm a CSFK mindazon munkatársainak, akik segítettek eligazodni a kutatói környezetben, és köszönöm a rengeteg szakmai tanácsot, amit adtak.

A témával az Új Nemzeti Kiválóság Program "Tehetséggel Fel!" Felsőoktatást Megkezdő Kutatói Ösztöndíj Pályázatát is sikerült elnyernem. (Pályázati azonosító: ÚNKP-22-6-I-ELTE-675). Köszönöm ezúton a Kulturális és Innovációs Minisztériumnak, valamint a Nemzeti Kutatási, Fejlesztési és Innovációs Hivatalnak.



Hivatkozások

Delbó, M., & Harris, A. W. 2002, *M&PS*, 37, 1929

Delbo, M., Mueller, M., Emery, J. P., Rozitis, B., & Capria, M. T. 2015, in *Asteroids IV*, 107–128

Dunham, E. T., Desch, S. J., & Probst, L. 2019, *The Astrophysical Journal*, 877, 41. <http://dx.doi.org/10.3847/1538-4357/ab13b3>

MacLennan, E. M., & Emery, J. P. 2018, *The Astronomical Journal*, 157, 2. <http://dx.doi.org/10.3847/1538-3881/aaed47>

Müller, T., Kiss, C., Alí-Lagoa, V., et al. 2019, *Icarus*, 334, 39–51. <http://dx.doi.org/10.1016/j.icarus.2018.11.011>

Ortiz, J. L., Santos-Sanz, P., Sicardy, B., et al. 2017, *Nature*, 550, 219–223. <http://dx.doi.org/10.1038/nature24051>

Shepard, M. K., Timerson, B., Scheeres, D. J., et al. 2018, *Icarus*, 311, 197

Szakáts, R., Müller, T., Alí-Lagoa, V., et al. 2020, *A&A*, 635, A54