Csillagászati Laboratórium II.

4. óra: Kozmikus sugarak és koordinátarendszerek

Az előző óra végén eljutottunk oda, hogy a képeinkre elvégeztük a flat-field korrekciót. A flatelt képeket vizuálisan ellenőriztük és konklúziókat vontunk le a flatelést illetően. A mai óra első feladata a képek kozmikus-sugár mentesítése lesz.

1. A kozmikus-sugár levonás - `cosmicray`

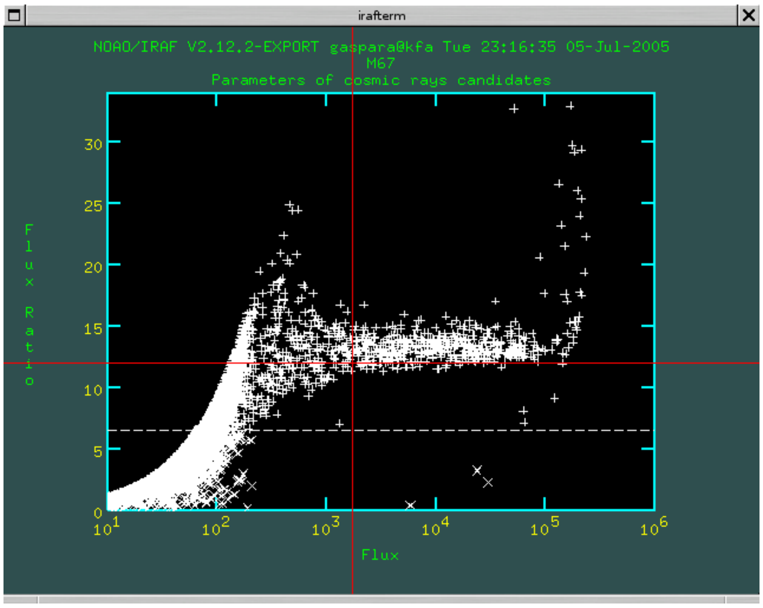
A képek készítése közben a CCD kamera pixeleit nem csak az objektumokból jövő fotonok bombázzák, hanem a kozmikus sugarak nagyenergiájú részecskéi is. Ezek a képeken egy δ-függvényként jelentkeznek, mivel csak egy pixelt érintenek. Ennek köszönhetően nem is annyira nehéz (bár nagyon időigényes) a képekről való eltávolításuk.

A sugarakat a `noao.imred.crutil.` csomagon belül található `cosmicrays` *TASK*-kal lehet eltávolítani. Ezt a *TASK*-ot mindíg interaktívan futtassuk, nagy figyelemmel. Nem szeretnénk ugyanis véletlenül csillagokat is eltávolítani a képekről. **Mivel ez a *TASK* nagyon időigényes, ezért csak az NGC 2126-os képekre (4 darab kép) végezzük el a korrekciót. Az M67-es képeknél amúgy is csak a pár standard csillag fényességértékére lesz szükségünk.** A *TASK* paraméter-listája így néz ki:

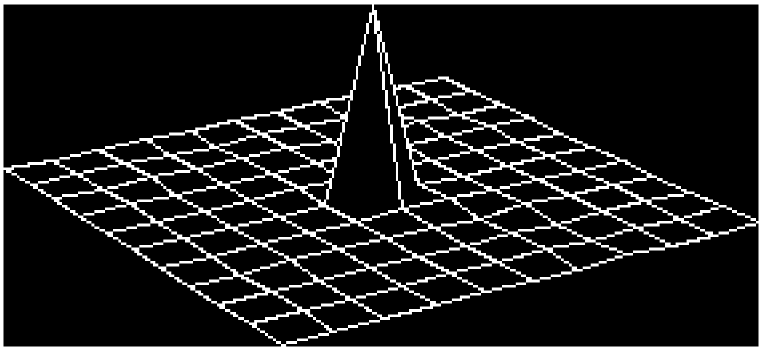
```
                         Image Reduction and Analysis Facility
PACKAGE = crutil
   TASK = cosmicrays

input   =          @flatelt.ls  List of images in which to detect cosmic rays
output  =           @cosmic.ls  List of cosmic ray replaced output images
(crmasks=                    ) List of bad pixel masks (optional)

(thresho=                 10.) Detection threshold above mean
(fluxrat=                  2.) Flux ratio threshold (in percent)
(npasses=                   5) Number of detection passes
(window =                   5) Size of detection window

(interac=                 yes) Examine parameters interactively?
(train  =                  no) Use training objects?
(objects=                    ) Cursor list of training objects
(savefil=                    ) File to save train objects
(plotfil=                    ) Plot file
(graphic=            stdgraph) Interactive graphics output device
(cursor =                    ) Graphics cursor input
answer  =                      Review parameters for a particular image?
(mode   =                  ql)
```

A bemenetnek meg kell adni azon képek listáját, amikre a kozmikus levonást el szeretnénk végezni. Emellett ugyanúgy, mint a flatkorrekció során, meg kell adni egy kimeneti képlistát is. Ügyeljünk rá, hogy ne ugyanaz legyen a képek neve a két listában, illetve, hogy adjunk meg kimeneti lisasort (melyben ugyanannyi név szerepel, mint a bemenetiben), különben felülírjuk az eredeti fájlokat.

A program futtatásakor egy `irafterm` ablak jelenik meg előttünk (1. ábra). Az azonosított objektumok tipusuktól függően általában jellegzetes helyen találhatóak a grafikonon. Értelmezzük egy kicsit az ábrát. Az alsó tengelyen az objektumok összfluxusa található, míg a függőleges tengelyen a fluxus-hányadosuk. A kozmikus sugaraknak a fluxusa nagy, hisz a CCD pixelét telítésbe viszi. Viszont a fluxus-hányadosa, azaz hogy mekkora különbségek vannak a fluxusok között az objektumban belül, az alacsony (a második legfényesebb pixelt hasonlítja össze a környező pixelekkel). Így a kozmikus sugarak leginkább a diagram alsó jobb oldalán találhatóak. Az összefüggő terület a diagram bal oldalán a képen látható kicsiny háttérfluktuációk. Azokat nem kell levonni. A nagyobb fluxussal és fluxus-hányadossal rendelkező területen a csillagok találhatóak.



1. ábra. Az irafterm ablaka



2. ábra. Egy tipikus kozmikus sugár

Itt nagyon óvatosan szabad csak törölni. Általában az ábra legtetején egy másik tömörülés is szokott lenni. Ezek a „negatív” beütések.

Nézegessük meg az egyes pontokhoz tartozó objektumok kontúrábráit. A kontúrokat az „`s`” gomb megnyomásával kérhetjük. Az ábrán beállíthatunk egy „`threshold`” szintet a „`t`” megnyomásával, mi alatt mindent töröl az *IRAF*. Ezen kívül a „`d`”-vel törölhetünk pontokat, illetve az „`u`”-val visszavonhatjuk a törlést. Kijelölhetünk egy négyzetes területet is, annak két sarkával. A sarokpontokban az „`e`”-t kell megnyomni. Az egyéb használható gombokat itt is a „`?`” megnyomásával kaphatjuk meg. Ha minden kozmikus sugarat töröltünk és beállítottuk a threshold szintet, akkor a „`q`”-val kilépünk az interaktív részből. A program ezután levonja a kozmikus sugarakat. **FONTOS! Az `irafterm` ablakot az *IRAF*-ból való kilépésig sohase zárjuk be, ha már egyszer használtuk!** Csináljuk meg a kozmikus sugár levonásokat az egyes objektum-képekre. A biztonság kedvéért ne írjuk felül a korábbi képeket, hanem hozzunk létre új képeket. Az új képek listáját awk scripttel vagy shell scripttel hozzuk létre.

Ha készen vagyunk a korrekcióval, akkor blinkeljük össze a levonás előtti képeket a levonás utániakkal. Az eredményt szövegesen értékeljük a jegyzőkönyvben.

2. A képek összecsúsztatása

A zsúfolt csillagmezők fotometriájánál gyakran alkamazott eljárás a képek összecsúsztatása. Ezzel biztosíthatjuk, hogy ugyanarra a pixelre ugyanaz a csillag kerül minden képnél. Emellett egyedi csillagok fotometriájánál is szokás a képeket összetolni. Ekkor a cél az, hogy az összehasonlító csillag is és a mérendő csillag ugyanazon a pontban legyen (itt az egyszerűsítés az, hogy az *IRAF*-nak a saját eltolás kereső algoritmusa automatikusan megkeresi az eltolás értékeket).

Köztudott, hogy a csillagászati színszűrők nem feltétlenül ugyanolyan vastagságúak és törésmutatójúak, éppen ezért a különböző szűrőkkel készült képek között nem csak `x` és `y` irányú eltolás lehetséges, hanem valamely 2D-s függvény szerinti transzformáció. Ez nehezítő tény, főleg mondjuk egy gömbhalmaz sűrűségű csillagászati objektum esetén.

Az eltolás illetve transzformációkat több úton meg lehet határozni az *IRAF* segítségével. Ha kevés csillag (pl.: csillag, comp, check) található az eredeti képeken és nagyjából ugyanoda esnek mindegyik képen, akkor meg lehet próbálni az *IRAF* beépített `imalign` *TASK*-ját futtatni az `images.immatch.` csomagon belül. Mivel esetünkben zsúfolt csillagmezőről (crowded field) van szó, ezért más, macerásabb módszerhez kell nyúlnunk.

Ha bízunk abban, hogy nem görbült a fókuszsíkja a műszerünknek, nem túl zsúfolt csillagmezőt vizsgálunk (azaz mondjuk nem gömbhalmazt) és csak egy szűrőt alkalmaztunk (ami igen ritka), akkor elég a képeket szimplán lineárisan egymásba csúsztatni. Ehhez logikus módszer az például, hogy megmérjük egy csillagnak a pozícióját képenként és meghatározzuk ezzel a szükséges pixeleltolásokat. Ez több ezer kép esetén viszonylag macerásabb munka. Ekkor szokták alkalmazni a keresztkorreláció maximumának módszerét.

> Két függvény keresztkorrelációja megadja, hogy mekkora a két függvény szorzatának értéke az egyik függvény eltolásának függvényében. Azaz, amikor a két függvény a lehető legjobban átfed, akkor van maximuma. Ezzel a módszerrel szokták például spektrumvonalak vizsgálata során megállapítani csillagok radiális sebességét. A keresztkorrelációs függvény alakja:
>
> $$C(t) = \int_{min}^{max} f(\tau) \cdot g(\tau + t)\ d\tau \tag{1}$$
>
> Ugyanezt a játékot el lehet játszani diszkrét értékű pixelekkel és 2D-ban. Ezzel a módszerrel könnyű nagy adatbázisokat kezelni, illetve az infravörös képfeldolgozásban bevett módszer a képek keresztkorrelációjának a megkeresése.

Mi egy harmadik (legbabrásabb, de legszebb eredményt adó) módszerrel fogjuk az M67-es méréseinket egymáshoz illeszteni. Minden egyes képen megmérjük több (legalább 20) csillag pozícióját. Fontos, hogy ezek a csillagok egyenletesen legyenek elosztva a látómezőben. Az órán az oktatótól erre a célra mindenki kap egy kinyomtatott fits képet, melyen bejelölheti a vizsgálatra használandó csillagokat. Fontos, hogy a csillagok magányosak legyenek és szép csillagprofillal rendelkezzenek. A kiválasztott csillagokat ezután (ugyanolyan sorrendben) minden képen megmérjük. Hogy ez egyszerű legyen, irjunk egy scriptet, mely egyesével berakja a képeket a `ds9`-be és elindítja az `imexamine` *TASK*-ot.

```
$ for i in *.fit;do echo "displ" $i "1";echo "imexam >" $i".dat";done > imex.cl
```

> A script magyarázata tömören: A `for`-ral egy ciklust indítunk mindent olyan fájra, melynek neve passzol a `*.fits` mintára. Ezeket a képneveket egy `i` változóval jelölt tömben tároljuk el. A `do`-val indítjuk és a `done`-nal zárjuk le a ciklust. Az `echo` (mint a visszhang) mindent visszamond, amit `""` közé zárunk. A `;`-vel pedig sort törünk. A `$i` helyére a változó nevét írja ki a script. A végén a `>` jellel irányítjuk át a szövegsort egy fájlba. Érdemes az irányítás nélkül kiprobálni a scriptet, hogy lássuk, hogyan is működik.

A script meggyorsítja a 32 soros *IRAF* scriptfájl írását. Akik akarják megírhatják kézzel is a fájlt. Ha valaki ezt az utat választja, akkor a scriptfájljának így kell kinéznie (ugyanezt kapjuk a scripttel is):

```
displ cft332.fit 1
imexam > cft332.fit.dat
displ cft333.fit 1
imexam > cft333.fit.dat
...
```

Az *IRAF* scripteket általában `cl` kiterjesztéssel szoktuk ellátni, hogy lássuk később esetleg, hogy ez egy *IRAF* script. Ezt a létrehozott scriptet most adjuk be az *IRAF*-nak (ne feledjük, „`cl>`"-lel a promptot jelölöm):

```
cl> cl < imex.cl
```

Ennek hatására beadja az első képet az egyes frame-be, és várja a `ds9` ablaknál a cursor parancsokat. Mint korábban láttuk, az `imexamine`-nal lehet egyszerű, gyors apertúrafotometriát végezni. E során a program kiszámolja a csillag közepének a koordinátáját. Hangsúlyozzuk, hogy ezzel komoly, tudományos fotometriát nem szabad végezni, ez csak egy első értékadáshoz megfelelő eredményt ad. Nézzük meg az `imexamine` paraméterfájlját, és hogy a `centering` opció be legyen kapcsolva, mielőtt futtatjuk a scriptet.

A script futtatása közben kiadja a képeket. Feladatunk a kiválasztott legalább 20 darab csillagot egyesével, sorrendben, képenként a „`,`" lenyomásával megmérni. Miután egy képpel végeztünk, a „`q`"-val léphetünk a következő képre.

Egy létrejött `dat` fájl valahogy így néz ki:

```
#   COL    LINE    RMAG     FLUX      SKY   N  RMOM ELLIP    PA     PEAK MFWHM
 646.06  472.10   13.01  62699.4   104.70  76  4.16 0.129  30.8  2276.72  5.06
 375.96  371.41   14.81  11875.3   101.47  79  4.01 0.091  43.8   422.09  5.06
 268.38  247.17   13.32  47154.5   102.02  79  3.95 0.103  36.8  1686.03  5.03
 ...
```

Most ezen koordináták segítségével fogjuk eltolni, illetve megfelelő formájúra transzformálni a képeket. Ki kell választani egy referenciaképet először. Ez általában az a kép, melynél a halmaz a legszebben a kép közepén található. Esetünkben nyugodtan lehet az első kép például.

A transzformációt az `images.immatch.` csomagon belül található `geomap` illetve `geotran` *TASK*-okkal fogjuk végrehajtani. A `geomap` csomag keresi meg a transzformációs mátrixot, míg a `geotran` hajtja végre a transzormációkat az egyes képekre. Nézzük meg a `geomap` help-jét, hogy milyen formájú bemeneti fájlokat vár.

```
input
    The list of text  files  containing  the  pixel  coordinates  of
    control  points  in  the reference and input images. The control
    points are listed one per line with xref, yref, xin, and yin  in
    columns 1 through 4 respectively.
```

Látható, hogy a képenkénti bemenő adatfájlban négy oszlopnak kell lennie, méghozzá úgy, hogy az első két oszlopban a referencia koordinátáknak, a második két oszlopban pedig az adott képen a csillagok koordinátái kell, hogy szerepeljenek.

Ezt a problémát egy nagyon egyszerű Linux scripttel gyorsan megoldhatjuk. Először is válogassuk ki a `dat` fájlokból az első két oszlopot (illetve a kezdő magyarázó sort szedjük ki). A `$i` a script második sorában a script része, nem promptjel.

```
$ for i in *.dat;do echo "grep -v \"#\" $i | awk '{print "'$'"1,"'$'"2}' >"
  $i"a";done
```

A script magyarázata: A `for`-ral itt is ciklust indítunk a „`*.dat`" mintájú fájlokra. A ciklus a `do` és a `done` közötti részre vonatkozik. Az `echo`-val most is visszhangozzuk a `""` között részeket. Mivel a `#` jel speciális script parancsjel is (ugyanúgy, mint a `$` jel), ezért kell, hogy jelezzük, hogy karakterként értelmezzük. Illetve ezt majd később a `grep`-nek is kell, ezért ez a sok `"'"` karakter. A „`|`" pipe jellel a kimenetet átirányítjuk majd egy `awk` parancsnak, s végül a kimenetet pedig fájlonként külön fájlba. Kicsikét bonyolult elsőre, de nagyobb adatbázisok esetén lehetetlen minden fájlt kézzel megszerkeszteni. Itt is ajánlatos először a képernyőre kiiratni a sorokat (nem pedig rögtön átirányítani a parancsértelmező `bash` shellbe ... azaz „`| bash`" nélkül futtatni).

Ha valaki nem akar scriptekkel bajlódni, akkor az a 16 fájlt kézzel is megszerkesztheti. Ezzel megvannak a csak koordinátákat tartalmazó „`data`" fájljaink. Érdemes egy `wc -l *.data` paranccsal leellenőrizni, hogy ugyanannyi sort tartalmaz-e minden fájl. Most mindegyik fájlba bele kell tennünk az első két oszlop helyére a referenciakoordinátákat. Ezt is egy scripttel oldjuk meg:

```
$ cp cft332.fits.data refcord
$ for i in *.data;do echo "paste refcord" $i "> tmp";echo "mv tmp" $i;done | bash
$ rm refcord
```

A script magarázata: Az első sorral szimplán csak lemásoljuk a referenciának használandó koordinátalistát a `refcord` nevű fájlba. A második sor az igazi script. Itt a ciklust minden `*.data` nevű fájlra futtatjuk. A `paste` szimpla *UNIX* parancs, mely fájlok adatoszlopait egymás mellé másolja. A kimenetet a képernyő helyett mindig egy `tmp` nevű fájlba irányítjuk, amivel aztán felülírjuk az eredetit (természetesem nem, írhatjuk felül a fájlt, miközben olvasunk is belőle, ezért kell ez a trükk). Az utolsó sorban kitöröljük a `refcord` fájlt, hisz többet nem lesz rá szükség.

Ezzel megvannak a `geomap`-nak beadandó koordinátafájljaink. Tegyük be ezeket a fájlokat egy listába:

```
$ ls *.data > data.ls
```

Ezt a listát egyenest lehet beadni a `geomap`-nak:

```
PACKAGE = immatch
   TASK = geomap

input   =             @data.ls  The input coordinate files
database=            data.base  The output database file
xmin    =                    1  Minimum x reference coordinate value
xmax    =                 1536  Maximum x reference coordinate value
ymin    =                    1  Minimum y reference coordinate value
ymax    =                 1024  Maximum y reference coordinate value
(transfo=                     ) The output transform records names
(results=                     ) The optional results summary files
(fitgeom=              general) Fitting geometry
(functio=           polynomial) Surface type
(xxorder=                    3) Order of x fit in x
(xyorder=                    3) Order of x fit in y
(xxterms=                 half) X fit cross terms type
(yxorder=                    3) Order of y fit in x
(yyorder=                    3) Order of y fit in y
(yxterms=                 half) Y fit cross terms type
(maxiter=                    2) Maximum number of rejection iterations
(reject =                   3.) Rejection limit in sigma units
(calctyp=                 real) Computation type
(verbose=                  yes) Print messages about progress of task ?
(interac=                   no) Fit transformation interactively ?
(graphic=             stdgraph) Default graphics device
(cursor =                     ) Graphics cursor
(mode   =                   ql)
```

Ha minden beállítást megfelelőnek találunk (érdemes olvasgatni a help fájlt), akkor futtassuk a programot („`:g`”). Látjuk, hogy az `xgterm` ablakba a program sok információt kiír (természetesen csak akkor, ha a `verbose` funkciót `yes`-re állítottuk). A létrejött `database` fájlt adjuk meg a `geotran` *TASK*-nak. A paraméterfájlja:

```
PACKAGE = immatch
   TASK = geotran

input   =           @kepek.ls  Input data
output  =         @eredmeny.ls  Output data
database=           data.base  Name of GEOMAP database file
transfor=            @data.ls  Names of coordinate transforms in database
(geometr=           geometric) Transformation type (linear,geometric)
(xin    =               INDEF) X origin of input frame in pixels
(yin    =               INDEF) Y origin of input frame in pixels
(xshift =               INDEF) X origin shift in pixels
(yshift =               INDEF) Y origin shift in pixels
(xout   =               INDEF) X origin of output frame in reference units
(yout   =               INDEF) Y origin of output frame in reference units
(xmag   =               INDEF) X scale of input picture in pixels
(ymag   =               INDEF) Y scale of input picture in pixels
(xrotati=               INDEF) X axis rotation in degrees
(yrotati=               INDEF) Y axis rotation in degrees
(xmin   =                   1) Minimum reference x value of output picture
(xmax   =                1536) Maximum reference x value of output picture
(ymin   =                   1) Minimum reference y value of output picture
(ymax   =                1024) Maximum reference y value of output picture
(xscale =                  1.) X scale of output picture in reference units
(yscale =                  1.) Y scale of output picture in reference units
(ncols  =                1536) Number of columns in the output picture
(nlines =                1024) Number of lines in the output picture
(xsample=                  1.) Coordinate surface sampling interval in x
(ysample=                  1.) Coordinate surface sampling interval in y
(interpo=              linear) Interpolant
(boundar=             nearest) Boundary (nearest,constant,reflect,wrap)
(constan=                  0.) Constant boundary extension
(fluxcon=                 yes) Preserve image flux?
(nxblock=                 512) X dimension of working block size in pixels
(nyblock=                 512) Y dimension of working block size in pixels
(verbose=                 yes) Print messages about the progress of the task
(mode   =                  ql)
```

Érdemes egy eredmény listafájlt készíteni és nem felülírni a képeket... ki tudja! Ha készek vagyunk a transzformációval, akkor érdemes a képeket összeblinkelni és ellenőrizni, hogy jók lettek-e a transzformációk.

Kérdések, feladatok:

1. *Végezzétek el a flatelt, NGC 2126-os képek kozmikus sugár mentesítését. Az eredményül kapott képeket blinkeljétek össze az eredetiekkel, és szövegben foglaljátok össze tapasztalatotokat.*

2. *Toljátok össze az M67-es képeket a* `geomap, geotran` *TASK-ok segítségével. Az eljárást, illetve a lényegét ismertessétek szóban. A képeket blinkeljétek össze, hogy lássátok, jó eredményt kaptatok-e.*

3. *Szükségesnek tartott képeket, ábrákat mentsetek le* `eps` *formátumba, hogy a dolgozatba beilleszthessétek őket.*

4. *A laborvezetőnek mutassátok meg a végeredményül kapott képeket, mindkét eljárás alatt. Azaz a kozmikus levonás eredményét, illetve az eltolások eredményét is.*