The Imcat Reference Manual

Nick Kaiser

April 13, 2003
## Contents

1. `imcat(1)`  
2. `cattools(1)`  
   2.1 `catcats(1)`  
   2.2 `catstats(1)`  
   2.3 `diff2Dpolymodel(1)`  
   2.4 `fit2Dpolymodel(1)`  
   2.5 `gen2Dpolymodel(1)`  
   2.6 `gen2Dpolymodelimage(1)`  
   2.7 `gridavg(1)`  
   2.8 `int2Dpolymodel(1)`  
   2.9 `maskcat(1)`  
   2.10 `pair(1)`  
   2.11 `pastecats(1)`  
   2.12 `rebin(1)`  
   2.13 `cattools lc(1)`  
      2.13.1 `lc(1)`  
      2.13.2 `makegridcat(1)`  
      2.13.3 `makerandcat(1)`  
   2.14 `cattools lmodel(1)`  
      2.14.1 `difflmodel(1)`  
      2.14.2 `fitlmodel(1)`  
      2.14.3 `generatelmodel(1)`  
      2.14.4 `generatelmodelimage(1)`  
   2.15 `cattools mergecats(1)`  
      2.15.1 `cleancat(1)`  
      2.15.2 `findtrailobjects(1)`  
      2.15.3 `getclosepairs(1)`  
      2.15.4 `getisolatedobjects(1)`  
      2.15.5 `mergecats(1)`  
      2.15.6 `nuketrailobjects(1)`  
      2.15.7 `unmergecat(1)`  
3. `dev(1)`  
   3.1 `findcluster(1)`  
   3.2 `makelist(1)`  
   3.3 `matchpairs(1)`  
   3.4 `matchtriplets(1)`  
   3.5 `mergelists(1)`  
   3.6 `pairs(1)`  
   3.7 `transformlist(1)`  
   3.8 `triplets(1)`  
   3.9 `imcattools astrometry(1)`  
      3.9.1 `convertcoords(1)`
3.9.2 decimaltodms(1) ........................................ 64
3.9.3 decimaltohms(1) ...................................... 65
3.9.4 dmstodecimal(1) ...................................... 66
3.9.5 getUSNOACatfromESOhtml(1) ......................... 67
3.9.6 getwcsinfo(1) ........................................ 68
3.9.7 getworldcoords(1) ................................... 69
3.9.8 getxfromradec(1) .................................... 70
3.9.9 hmsstodecimal(1) .................................... 71
3.9.10 readusnoacat(1) .................................... 72
3.9.11 setwcsinfo(1) ....................................... 74

3.10 imcattools_cosmicrays(1) ............................ 75
3.10.1 gethotpix(1) ........................................ 76
3.10.2 removecrs(1) ........................................ 77

3.11 imcattools_cosmology(1) .............................. 78
3.11.1 frw(1) ................................................ 79
3.11.2 nfw(1) ............................................... 80

3.12 imcattools_geomview(1) ............................... 82
3.12.1 fits3Dviewer(1) ..................................... 83
3.12.2 geomviewsetup(1) ................................... 84

3.13 imcattools_mosaics(1) ................................. 85
3.13.1 combinestacks(1) .................................. 87
3.13.2 fitgeometry(1) ...................................... 88
3.13.3 fitgeometry2(1) .................................... 90
3.13.4 fitgeometry3(1) .................................... 93
3.13.5 fitmagshifts(1) .................................... 94
3.13.6 fitmagshiftsplus(1) ................................ 95
3.13.7 fitpolymodelmagshift(1) ............................ 96
3.13.8 fitshifts(1) .......................................... 97
3.13.9 getfieldgeomfromheaders(1) ....................... 98
3.13.10 makemosaiccats(1) ................................ 99
3.13.11 makemosaicstack(1) ................................ 100
3.13.12 makemosaicstacks(1) ............................... 101
3.13.13 makestack(1) ....................................... 102
3.13.14 makewarpedalbum(1) ............................... 103
3.13.15 mergemosaiccats1(1) .............................. 104
3.13.16 mergemosaiccats2(1) .............................. 105
3.13.17 mosaicfit(1) ........................................ 106
3.13.18 printimages(1) .................................... 108
3.13.19 transformmosaiccat(1) ............................ 109

3.14 imcattools_objectfinding(1) ......................... 110
3.14.1 chunkyfp(1) ......................................... 111
3.14.2 findpeaks(1) ........................................ 112
3.14.3 getbadpix(1) ........................................ 114
3.14.4 hcat2cat(1) .......................................... 116
3.14.5 hfindpeaks(1) ...................................... 117
3.14.6 makechart(1) ........................................ 119
3.14.7  makestamps(1) ........................................ 120
3.14.8  minisky(1) ............................................ 122
3.14.9  surfdens(1) ........................................... 123

3.15  imcattools_optics(1) ...................................... 125
   3.15.1  make2Dotf(1) ......................................... 126
   3.15.2  makemotf(1) ......................................... 127
   3.15.3  makevonkarmanS(1) .................................... 129
   3.15.4  otftopsf(1) .......................................... 130
   3.15.5  phasetopsf(1) ....................................... 131
   3.15.6  stackpsfs(1) ......................................... 132
   3.15.7  trackpeak(1) ......................................... 133

3.16  imcattools_orbits(1) ..................................... 134
   3.16.1  airmass(1) .......................................... 136
   3.16.2  convertorbits(1) .................................... 137
   3.16.3  evolveorbits(1) ..................................... 138
   3.16.4  fitorbit(1) .......................................... 139
   3.16.5  getdet(1) ........................................... 141
   3.16.6  getmoid(1) .......................................... 142
   3.16.7  laplace3(1) ......................................... 143
   3.16.8  makeobs_circ(1) ..................................... 144
   3.16.9  makeobs_inertial(1) ................................. 146
   3.16.10  orbanim(1) ........................................ 147
   3.16.11  r2n(1) ............................................. 148

3.17  imcattools_pgplot(1) .................................... 149
   3.17.1  contour(1) .......................................... 150
   3.17.2  epsfcompose(1) ..................................... 152
   3.17.3  plotcat(1) .......................................... 154

3.18  imcattools_photometry(1) ................................ 159
   3.18.1  Getshapes(1) ........................................ 161
   3.18.2  apphot(1) ......................................... 162
   3.18.3  deproject(1) ....................................... 165
   3.18.4  getfitsval(1) ....................................... 166
   3.18.5  getshapes(1) ....................................... 168
   3.18.6  getshapes2(1) ..................................... 170
   3.18.7  getshapes3(1) ..................................... 172
   3.18.8  getsky(1) .......................................... 174
   3.18.9  makedensity(1) ..................................... 176
   3.18.10  makegausskernel(1) ............................... 178
   3.18.11  makekernel(1) ..................................... 179
   3.18.12  makepeff(1) ....................................... 180
   3.18.13  makepeff2(1) ..................................... 182
   3.18.14  modelpsf(1) ....................................... 184
   3.18.15  profile(1) ......................................... 186
   3.18.16  imcattools_photometry_fitting(1) .................. 187
   3.18.17  betafit(1) ......................................... 188
   3.18.18  fit1object(1) ...................................... 190
4  intools(1)  
4.1  acf(1)  . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 247
4.2  album(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 249
4.3  asciitofits(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 250
4.4  boxavg(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 251
4.5  bscale(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 252
4.6  changexsex(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 253
4.7  colorbox(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 254
4.8  colorize(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 255
4.9  combineimages(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 256
4.10 combinestamps(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 259
4.11 complex(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 260
4.12 convolve(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 261
4.13 cycleimage(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 262
4.14 fft(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 263
4.15 fillcores(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 265
4.16 fitstoascii(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 266
4.17 fitstocat(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 267
4.18 fitstomovie(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 268
4.19 flatten(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 270
4.20 flip(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 271
4.21 getplane(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 272
4.22 getplanes(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 273
4.23 grad(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 274
4.24 growmagic(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 275
4.25 helicalscan(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 276
4.26 ic_stream(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 277
4.27 imhead(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 278
4.28 invgrad(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 279
4.29 kappa2stuff(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 280
4.30 lensmap(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 281
4.31 linesmode(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 282
4.32 logscaleimage(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 283
4.33 make_image(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 284
4.34 makecolourimage(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 285
4.35 makesubimage(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 286
4.36 mapbynumericdef(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 287
4.37 mask(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 289
4.38 maskfits(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 290
4.39 overscancorr(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 291
4.40 pixdist(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 292
4.41 projectfits(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 293
4.42 scrunch(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 294
4.43 simulate(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 295
4.44 sliceimage(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 296
4.45 smooth(1) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 297
4.46 spinflip(1) .......................................................... 298
4.47 stackplanes(1) ....................................................... 299
4.48 stats(1) ............................................................. 300
4.49 statstocat(1) ....................................................... 301
4.50 transformimage(1) ................................................ 302
4.51 translate_fft(1) .................................................... 304
4.52 transposebits(1) .................................................... 305
4.53 unpackextensions(1) ............................................. 306
4.54 unscrunch(1) ....................................................... 307
4.55 warpimage(1) ....................................................... 308
4.56 imtools_fitstops(1) ............................................. 310
   4.56.1 fitstops(1) ................................................... 311
4.57 imtools_ic(1) ...................................................... 313
   4.57.1 ic(1) .......................................................... 314
4.58 imtools_iis(1) ..................................................... 317
   4.58.1 iis(1) .......................................................... 318
4.59 imtools_xfv(1) ..................................................... 319
   4.59.1 xfv(1) .......................................................... 320
4.60 imtools_xim(1) ..................................................... 321
   4.60.1 xim(1) .......................................................... 322

5 nk(1) 323
5.1 clickstocat(1) ...................................................... 325
5.2 doto(1) ............................................................. 326
5.3 makecatmanpage(1) .............................................. 327
5.4 makegroffmanpage(1) ......................................... 328
5.5 makehtmlmanpage(1) ......................................... 329
5.6 makeimcatman(1) ............................................... 330
5.7 makenmanpages(1) ............................................. 331
5.8 maketexmanpage(1) ............................................ 332
5.9 xevtocat(1) ....................................................... 333
5.10 nk_tablestuff(1) ............................................... 335
   5.10.1 mova(1) ....................................................... 336
   5.10.2 tablearith(1) ............................................... 337
   5.10.3 tablecumulate(1) ......................................... 338
   5.10.4 tablefilter(1) ............................................... 339
   5.10.5 tableop(1) ................................................... 340
   5.10.6 tablesort(1) ............................................... 341
   5.10.7 tablesum(1) .................................................. 342
1  imcat(1)

NAME

imcat - image and catalogue manipulation software

DESCRIPTION

The imcat tools are arranged into sections. Use 'man sectionname' to list the tools available in a section.

SECTIONS

"cattools"
"cattools_lc"
"cattools_lmodel"
"cattools_mergcats"
"dev"
"imcattools.astrometry"
"imcattools.cosmicrays"
"imcattools.cosmology"
"imcattools.geomview"
"imcattools.mosaics"
"imcattools.objectfinding"
"imcattools.optics"
"imcattools.orbits"
"imcattools.pgplot"
"imcattools.photometry"
"imcattools.photometry_fitting"
"imcattools.psfcorrection"
INFORMATION

The 'imcat' software was developed initially to do faint galaxy photometry for weak lensing studies, and provides a fairly complete set of tools for this kind of work. Unlike most packages for doing data analysis, the tools here are provided as a set of standalone unix commands which you can invoke from the shell, via shell scripts or from perl scripts as you think appropriate. You don't have to learn a new shell syntax - just use whatever you are most comfortable with - and you can just use whatever bits you want. The imcat libraries are not shared, so you can just download a single executable if you like. The standard unix and pgplot libraries are shared, and you may need to explicitly locate them via the LD_LIBRARY_PATH env variable. All of the commands have a "man-page". The html versions here tend to be out of date, but the most definitive version for any command can always be obtained by issuing the command with the '-u' (for usage) flag. Of course, there is no guarantee that these are perfectly accurate - as a last resort simply peruse the c-code.
The tools are arranged in a tree of directories. One main branch is the 'imtools'. These deal only with fits files. The most important imtool is the 'image calculator' 'ic' which allows one to do rather general operations on fits images. A second branch is the 'catools' which operate only on catalogues. The key catool is 'lc'; this effectively defines the format of imcat catalogues, and allows one to do very general operations on and filtering of such catalogues. A third branch is the 'imcattools'. These tend to be much more specialised than the catools and imcattools and are focussed on faint galaxy photometry. They also tend to be more volatile than the other sections as they are under active development.

AUTHOR

Nick Kaiser - kaiser@hawaii.edu
2 cattools(1)

NAME

cattools - cattools tools section

DESCRIPTION

This section contains some tools for doing basic operations on 'lc' format catalogues.

COMMANDS

"catcats " shell script to concatenate catalogues
"catstats " calculate statistics for catalogue object values
"diff2Dpolymodel " differentiate a 2-dimensional spatial polynomial model
"fit2Dpolymodel " fit a 2-dimensional spatial polynomial model to a cat
"gen2Dpolymodel " generate 2D spatial polynomial model
"gen2Dpolymodelimage " generate 2D spatial polynomial model as fits image
"gridavg " average objects binned onto a spatial grid
"int2Dpolymodel " integrate a 2-dimensional spatial polynomial model
"maskcat " remove objects falling in a list of rectangles
"pair " generate a catalogue containing pairs of objects from input catalogue
"pastecats " combine catalogues
"rebin " rebin x-y table to logarithmically spaced intervals in x

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
2.1 catcats(1)

NAME

catcats - shell script to concatenate catalogues

SYNOPSIS

catcats cat1 cat2.....

DESCRIPTION

'catcats' reads a set of catalogues, writes the first to stdout and then follows this with the objects from the subsequent files but without the header. This is very rudimentary and no check is made that the catalogues contain the same items even. The catalogues are output in the same form (binary or text) as input, and must all have the same form.

AUTHOR

Nick Kaiser - kaiser@hawaii.edu
NAME

catstats - calculate statistics for catalogue object values

SYNOPSIS

catstats [options....]

"-s " only output count, min, max, mean, sigma1

"-v statistic " only output 'statistic'

DESCRIPTION

'catstats' reads a catalogue from stdin and creates a new catalogue whose first object item is a text entry 'statistic' which takes values 'count', 'min', 'max', 'mean'.... and whose subsequent object items have the same names as the numerical items in the input catalogue and which contain the appropriate statistic.

Catstats will always output the basic statistics:

"count " number of objects

"min " minimum value

"max " maximum value

"mean " \( <f> = \frac{\text{sum } f}{\text{count}} \) \( \sigma_1 \# \sqrt{<f^2> - <f>^2} \)

and by default will also calculate

"mode " 'robust' mode estimator

"median " median

"1quart " upper quartile

"uquart " lower quartile

"sigma2 " 'robust' sigma estimator
provided there are at least 8 objects in the catalogue.

The statistics 'mode' and 'sigma2' are designed to be insensitive to outliers.

The 'mode' is estimated by first making a crude estimate of sigma as 
(uquart - lquart) / 1.34; smoothing the histogram of values with a
gaussian of width sigma and returning the location of the peak.

'sigma2' is estimated from the width of the region around the mode
containing 25 percent of the values (and assuming) a gaussian distribution.
Thus, sigma2 is effectively measured from the curvature of the distribution
around the mode and will, for example, overestimate the real sigma
for a very boxy distribution.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
2.3 diff2Dpolymodel(1)

NAME

diff2Dpolymodel - differentiate a 2-dimensional spatial polynomial model

SYNOPSIS

diff2Dpolymodel

DESCRIPTION

'diff2Dpolymodel' reads from stdin a 2D polynomial model file with parameters for a single variable and sends to stdout a 2D poly model file for the two spatial derivatives of the variable.

This program requires that for each l, the mode values are supplied in the standard order: i.e. 0,1, ..... l.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
2.4 fit2Dpolymodel(1)

NAME

    fit2Dpolymodel - fit a 2-dimensional spatial polynomial
    model to a cat

SYNOPSIS

    fit2Dpolymodel xname lmin lmax var [options...]

    "-o x0 y0 " spatial origin for mode functions

DESCRIPTION

'fit2Dpolymodel' reads from stdin a catalogue which must contain at
least a 2-vector 'xname[2]' and a numerical variable 'var' (which may
be a scalar, vector or tensor of arbitrary rank) and determines the
coefficients of a 2-D spatial poly model:

    var_model = sum_l sum_m a_lm f_lm(x)

where the f_lm's are 2-D polynomials:

    f_lm(x) = x[0]^ (l - m) x[1]^ m

with lmin <= l <= lmax and 0 <= m <= l.

The model coefficients are output as an lc-format catalogue containing
the indices l, m, followed by the mode coefficients which have the
same names as the 'var' variable.

AUTHOR

    Nick Kaiser --- kaiser@ifa.hawaii.edu
2.5  gen2Dpolymodel(1)

NAME

gen2Dpolymodel - generate 2D spatial polynomial model

SYNOPSIS

gen2Dpolymodel parfile [options...]

"-x xname " specify spatial coordinate name

"-s suffix " suffix for model values ('mod')

DESCRIPTION

'gen2Dpolymodel' reads a catalogue from stdin, and a parameter file generated by 'fit2Dpolymodel' from 'parfile' and sends to stdout a catalogue containing all the items from the stdin cat plus an extra item consisting of the fitted model values: By default the name of the spatial variable is taken from the header of the 'parfile', but you can override this with the -x option. By default the name of the new item is inherited from the names of the model coefficient item in the 'parfile' with the addition of the suffix 'mod', but you can change this with the -s option.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
2.6  gen2Dpolymodelimage(1)

NAME

gen2Dpolymodelimage - generate 2D spatial polynomial model
as fits image

SYNOPSIS

gen2Dpolymodelimage parfile N1 N2

DESCRIPTION

'gen2Dpolymodelimage' generates a N1 x N2 float format fits image whose
value is given by f = sum_l sum_m a_{lm} f_{lm}(x) where the mode coefficients
are given in a parameters file (as generated by fit2Dpolymodel perhaps)
'parfile'.

Currently only works for fit to a scalar function.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
2.7  gridavg(1)

NAME

gridavg - average objects binned onto a spatial grid

SYNOPSIS

gridavg d [options....]

"-x xname " name for spatial coord ('x')
"-r x1 x2 y1 y2 " range of spatial coords
"-m " take median

DESCRIPTION

'gridavg' first reads a catalogue of objects from stdin and then assigns them to cells in a grid with spacing d. We then average the object values for each grid cell and output the resulting catalogue to stdout. The output catalogue contains a leading column 'ncell' containing the count of objects on the cell and following columns contain average values for the numeric items in the input catalogue.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
2.8 int2Dpolymodel(1)

NAME

int2Dpolymodel - integrate a 2-dimensional spatial polynomial model

SYNOPSIS

int2Dpolymodel

DESCRIPTION

'int2Dpolymodel' reads from stdin a 2D polynomial model file with parameters for a 2-vector variable (which is supposed to represent the spatial derivative of some scalar function 'f' [thought the actual name is arbitrary]) and sends to stdout a 2D poly model file for the mode coefficients of f. More explicitly, given a set of pairs of mode amplitudes $a^0_{l,m}$ $a^1_{l,m}$ then we output a single set of mode amplitudes: $a_{l,m} = ((l-m) a^0_{l-1,m} + m a^1_{l-1,m-1}) / (m^2 + (l-m)^2)$ which is the optimal combination assuming statistically independent, but equally noisy, coefficients for $a^0$, $a^1$.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
2.9  maskcat(1)

NAME

    maskcat - remove objects falling in a list of rectangles

SYNOPSIS

    maskcat [options...] maskfile

      "-x xname"  name for spatial coordinate ('x')
      "-i"       inverse selection mode

DESCRIPTION

"maskcat" reads a catalogue from stdin, removes any objects which fall within any of a set of rectangles specified in the file 'maskfile', and writes the result to stdout.

The mask file must be a lc-format catalogue containing (at least) a pair of position vectors $x1[2]$, $x2[2]$ which are diagonally opposite corners of the rectangle. Objects lying on the boundary will be rejected.

With the -i option, we output only those objects which fall within the union of the rectangles specified in maskfile.

AUTHOR

    Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

pair - generate a catalogue containing pairs of objects from input catalogue

SYNOPSIS

pair [options...]

"-d " only output N (N - 1) / 2 distinct pairs

DESCRIPTION

'pair' reads a catalogue from stdin and writes to stdout a catalogue which has object items with the same names as in the input cat but where each object is a 2-vector formed from a pair of input objects. By default all N (N - 1) pairs are output.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
2.11 pastecats(1)

NAME

pastecats - combine catalogues

SYNOPSIS

pastecats a.cat b.cat ....

DESCRIPTION

‘pastecats’ reads a set of catalogues and combines them line by line into a single catalogue. It reads the object contents information from each of the input headers in turn, and installs each item into the output catalogue’s list of object items. If no object item names are repeated then the result is essentially the same as using the unix ‘paste’ facility. If an item name appears more than once then the output catalogue will contain a single object item of that name, whose value comes from the last input object item of that name (though the position of the object in the list of items is determined by the first occurrence). The input catalogues will normally have the same length and be in some ordered correspondence with each other. Pastecats will return with exit value -1 if it appears that the catalogues do not have identical numbers of objects.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
2.12 rebin(1)

NAME

rebin - rebin x-y table to logarithmically spaced intervals in x

SYNOPSIS

rebin xname yname xmin xmax dlnx

DESCRIPTION

Rebin reads a from stdin a lc-format catalogue containing x-y pairs, with x values in increasing order. It then outputs to stdout a table of x and linearly interpolated y values where the x values are now uniformly distributed between xmin and xmax with log spacing dlnx.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

catools_lc - catools_lc tools section

DESCRIPTION

'lc' is a general purpose catalogue filter. It uses a reverse polish notation syntax to allow you to generate complicated operations on the command line. It is similar in some ways to 'awk', but much more efficient as it allows binary format catalogues. lc can be very useful is invoked as a filter to pipe catalogue data into and out of c-progs or perl-scripts.

COMMANDS

"lc " list and process catalogues of objects

"makegridcat " generate catalogue containing a grid of points

"makerandcat " generate catalogue containing random values

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

lc - list and process catalogues of objects

SYNOPSIS

lc [options...] [name1 [name2 ...]]

DESCRIPTION

Lc lists and processes catalogues of objects. A catalogue consists of a header, containing various numerical or textual header 'items' (along with their names and type information); a history of operations which have been applied to the catalogue, and other comments; and then a list of contents (the names and types of object items) followed by the objects themselves.

Catalogues can exist on disk as text files, for ease of transportation and reading by other applications such as plotting packages, or in binary form (but with a text format header) for efficiency; lc automatically recognizes the input catalogue format and by default outputs a text-format catalogue. Override this with the -o or -O options. As of 3/9/99 the catalogue I/O library function writes catalogues with a header item 'byteorder' which can be BIG_ENDIAN or LITTLE_ENDIAN according to the byte order of the machine, and will automatically convert such catalogues on input (catalogues are always output in the native byte order). There is an option -z which will swap bytes in a non-native byte order binary catalogue generated with earlier versions the the cat I/O routines.

The basic items that may appear in a catalogue (either in the header or in the objects) are arrays of numbers and text strings of arbitrary dimensionality. Numbers are all stored internally in double precision floating point format (but are printed by lc in '%g' format so integers retain their appearance).

Lc can be used to generate an output catalogue of objects containing a subset of the input catalogue object items by supplying the names of the items on the command line. It can also create new items which are fairly general functions of the input items by supplying an name argument in the form 'newvalname = rpnexpr' where the rhs is a reverse
polish notation expression for the new value 'newvalname' in terms of input object values, header values or constants. So, to create a new variable c = a + b from existing scalar variables a, b for example, supply the argument 'c = %a %b +'.

Other facilities include: selecting only header or object data for output; output conditional on truth of an (rpn) expression; sorting of output; addition, deletion and editing of header values; trimming the comments list; adding a date stamp to the header; adding a 'count' value, and converting a table of numbers to lc's catalogue format.

Lc's table filtering facilities are similar in some ways to those of 'awk'. One major difference is the use of meaningful names rather than numbers for column entries; this should make shell or perl scripts which invoke 'lc' somewhat clearer, and protect against errors if one makes changes in the format of catalogues by adding or changing the order of entries (which will have no effect if lc is used) or if a needed object item is missing (in which case lc will bail out with an error message). Other differences are the support for vectors, matrices and the ability to use header values in mathematical operations.

Lc reads from stdin and writes to stdout.

The following options apply to lc:

"-u " Print this 'man page'. Must be first option if used.
"-b " Output object data in binary form.
"-B " Output object data in same form (binary or text) as the input catalogue.
"-h " Output only the input catalogue header.
"-x " Do not include history string in comments.
"-c " Return number of objects which would otherwise have been output.
"-o " Output only table of object values, no header at all.
"-O " Output only table of object values, with a single header line showing names of the entities.
"-p name " Print the value of header item 'name' and exit.
"-P name " Print the value of header item 'name' and carry on.
"-i rpnexpr " Print output object conditional on 'rpnexpr',
which is a function of variables in the input catalogue.

"-s val " Sort the output in ascending order of val, which
may be the name of an input object item or a rpnexpr derived
from input object or header item values or constants.

"-S val " Sort in descending order.

"-H 'sym = expr' " Create a new header item named 'sym'
from the rpn-expression 'expr' which can refer to input catalogue
header values.

"-r name " Remove output object value 'name' (see +add
below).

"-R name " Remove header value 'name'.

-a string # Add 'string' as a comment.

"-d " Add a date stamp to header.

"-L n " Retain only last n comments.

"-F n " Retain only first n comments.

"-q seed " Seed for random number generator.

"-z " swap bytes on input

"-C " Convert a table of numbers into lc's format. Must
be first option if used.

With -C option, the following options may be used. Options -n -t -N
and -T should be supplied in appropriate order to define the format
of the input line. Corresponding items are created in the output catalogue.

"-n name " Create a single number object item 'name'.

"-t name " Create a single text string object item.

"-N string " Create an n-dimensional numeric array object
value num[i][j]... where i = 0..d0-1, j = 0..d1-1 etc. The
string contains the size and name info in the format 'n d0
d1... name'.

"-T string " Same as -N but creates a text string array
object item.
"-Q c " Treat lines starting with character 'c' (pound sign by default) as comments and copy to comments list of the output catalogue header.

"-I " Ignore lines beginning with pound sign (or character set with -C).

There are some special names for (collections of) variables:

"+all " Copy all of input object values over to output.

"+cname " An output object value containing the count of output objects with specified name.

"+Cname " Counter of input values.

NOTES

Lc first reads the input catalog header (or with -C creates one according to the information provided with -n -t -N -T options). It then creates a copy (the output catalogue header) containing all the header info, but with no object values. It then reads the command line arguments in turn, modifying the header values and creating the output object values accordingly. If, after parsing all the arguments, the output object is empty then output object items will be created for each input object item (i.e. it is assumed that if you don't explicitly specify which object items you want then you want everything rather than empty lines).

Objects are then read from the input stream one at a time, all the rpn expressions (including a conditional expression if specified with the -i option of present) are calculated and the resulting output object is sent to the output function either to be printed directly to stdout, counted (with -c option) or (with -s or -S) sorted and then printed.

Rpn expressions and assignments thereof are given as quoted strings of tokens separated by white space. Header and object values refer to values in the input catalogue. Object values are referred to using prefix '% ' and header values with prefix '^ '. Vectors and multi-dimensional arrays can be dereferenced with the c-like f[a][b] notation (though see 'deref' below). Dereferencing is not allowed on the the lhs of an assignment.

The rpn expression parser understands a lot of functions. These include all the standard C math library functions, along with '+', '-', '*', '/'; (and 'mult' is provided as a synonym for '*' to avoid problems
with the shell expanding '*' if you invoke 'lc' from a script). There are the logical comparison functions '>', '>=', '<', '<=', '==', '!=' and the negation operator '!' as well as 'and' and 'or'. There is a function 'if' (a.k.a. '?') which mimics the C language '(c ? t : f)' which returns 't' or 'f' respectively depending on the truth or falseness of the condition 'c'. The rpn syntax for this expression is 't f c ?' in which '?' pops the condition 'c' followed by 'f' and then 't' and pushes 't' or 'f' as appropriate. The condition 'c' will of course most likely be a compound logical expression. There is also a function 'enter' which duplicates the top value of the stack. There is a dereferencing operator 'deref' (e.g. 'f 2 deref' to get f[2]), but you will most likely use the 'f[2]' notation described above. Numerical constants are recognized as such automatically, string constants must be enclosed in curly braces: '{...}'. Strings should not enclose spaces.

There is a 'vector' function for constructing a vector of objects (as in 'menu = spam eggs 2 vector' to make a 2-vector with elements menu[0] = spam; menu[1] = eggs. Matrices can be constructed as vectors of vectors. There is a dot product operator 'dot' which can multiply two vectors, a vector by a matrix or multiply two matrices. There is a 'lintrans' function whose syntax is '%x phi_00 phi_01 phi_10 phi_11 lintrans' which pushes the 2-vector x = phi_i0 x_0 + phi_i1 x_1. There are vector subtraction and addition functions 'vsub' and 'vadd' to subtract or add two vectors of the same size, and 'vscale' and 'vshift' which take a vector as first argument, a scalar as second. 'vscale' multiplies each component of the vector by the scalar and 'vshift' adds the scalar to each component. There are analogous functions msub, madd, mscale which subtract, add or scale matrices. The function 'inverse' computes the inverse of its single argument which must be a square matrix of dimension 2 or 3.

There is a random number generator rand which calls drand48() and generates a number between 0.0 and 1.0, and a unit variance, zero mean gaussian random number 'grand'.

There is a string comparison operator 'eq', thanks to David Donovan.

EXAMPLES

Let's assume you have a non-lc format data file with contents:

```
# my unformatted data file
# X Y temp index file

0.123 0.345 451.2 27 spam
```

```
then feeding this to the stdin of the command

```
lc -C -N '1 2 r' -n temp -n index -t file -H 'title = foo' -H 'e = 2.1718'
```

would generate the lc-format catalogue:

```
# text format catalogue file --- do not edit manually, use 'lc'
# header:
# text 1 1 title foo
# number 1 1 e 2.1718
# comment: # my unformatted data file
# comment: # X Y temp index file
# comment: history: lc -C -N '1 2 r' -n temp -n index -t file -H 'title = foo' -H 'e = 2.1718'
# contents: 4
# number 1 2 r
# number 1 1 temp
# number 1 1 index
# text 1 1 file
# r[0] r[1] temp index file
0.123 0.345 451.2 27 spam
0.754 0.923 451.7 15 eggs
0.034 0.252 451.0 12 fish
0.967 0.204 452.9 30 spam
```

where we have specified that we want to read a 2-vector, 2 numbers
and a text string into each object (with the -N, -n and -t options)
and we have added a header string and numerical header value for good
measure. Note that the comment lines and also the lc-invocation have
been installed as comments (use -I and -x options to switch off these
features. In what follows it is assumed that this formatted catalogue
is the input to lc.

To generate a new catalogue containing only the 'r', 'temp' and 'file'
object values, you could use

lc r temp file

or

lc +all -r index

which first adds all of the input fields to the output catalogue and
then removes 'index'.

To print the subset of entries with 'index' > 20 say, do

lc -i '%index 20 >'

which illustrates how we refer to object values in rpn-expressions.
Similarly, to select only items with |r| > 0.3 say, you could do

lc -i '%r %r dot sqrt 0.3 >'

To select objects with 'file' equal to 'spam':

lc -i '%file spam eq'

To append an additional column 'Z' containing 1st component of the
r[] vector times the numerical header value 'e', prepend a column with
a counter 'ID', and sort the output values according to |r| do

lc +cID +all 'Z = %r[0] ^ e *' -s '%r %r dot'

Which illustrates how to refer to a header value in an rpn expression.
Note that object values in the sort-expression refer to objects in
the output catalogue rather than the input catalogue as is the case
for all other rpn-expressions.

BUGS

Indices in rpn expressions such as '%F[index]' should really be constants
or header values. If you try to use an object value or function derived
therefrom as an index then the index will evaluate to zero when the
rpn-string is parsed (as no input object data has been read at that
time), and you will subsequently just get the zeroth component of the
vector regardless of the current value of the index.
The rpn-expression format is somewhat clumsy. Someone should write a front end to lc which converts C-like expressions to rpn-expressions....

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

makegridcat - generate catalogue containing a grid of points

SYNOPSIS

makegridcat [-u | -b ] nx [ny [nz ....]]

DESCRIPTION

Makegridcat generates a lc-format catalogue containing a unit spaced grid of points for testing purposes.

If invoked with a single argument nx, makegridcat generates a catalogue containing nx objects consisting of a numerical scalar x with values 0,1...(nx-1). If invoked with two arguments nx, ny, it generates a catalogue containing nx * ny objects consisting of a 2-vector x[] where x[0] runs from 0 through nx-1 and x[1] runs from 0 through ny-1. If invoked with three arguments nx, ny, nz, it generates a catalogue containing nx * ny * nz objects consisting of a 3-vector x[] where x[0] runs from 0 through nx-1 and x[1] runs from 0 through ny-1 and x[2] runs from 0 through nz-1, and so on.

Default is text format output. Use -b option to generate a binary format catalogue.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

makerandcat - generate catalogue containing random values

SYNOPSIS

makerandcat n [-seed seed] [-dim dim] [-b]

DESCRIPTION

Makerandcat generates a lc-format catalogue containing random values for testing purposes. By default the catalogue contains n objects consisting of a 2-vector x[2], with values x[0], x[1] uniformly distributed on the interval 0,1 and is in text format.

Options:

"-seed seed " supply an integer seed for random number generator (1)

"-dim dim " specify alternative dimension for x-vector (2)

"-b " generate a binary format catalogue.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
2.14  cattools_lmodel(1)

NAME

cattools_lmodel - cattools_lmodel tools section

DESCRIPTION

This section contains some tools for fitting data to models which are superposition of mode functions:

\[ a(x) = \sum_m a_m f_m(x) \]

where \( x \) is some position vector (of arbitrary length), and \( a \) is a tensor of arbitrary rank. Possible examples are polynomials, fourier modes, Zernike polynomials etc.. These 'lmodels' are output as lc-format catalogues with the following required header items:

- string model_type e.g. polynomial, zernike
- string aname name of dependent variable
- string xname name of independent variable
- int xdim length of independent variable

and the following optional header items

- int nmodes
- double *xorigin

and for polynomial models

- int lmin
- int lmax

and for Zernike models

- int nmin
- int nmax

and for Fourier models
int kmin
int kmax
double lbox

Supported models are

model_type = polynomial

The mode functions are labelled by a set of indices $p[]$ with same length as $x[]$, the functions are $f_p = x_0^{p_0} x_1^{p_1} \ldots = \text{product } x_i^{p_i}$ and the order $l = \sum p_i$ lies in the inclusive interval $l_{\min}-l_{\max}$.

An alternative parameterisation of the indices is in terms of the order array $1[i] = l - \sum_{i=0}^{i-1} p[i]$, in terms of which the $p$-indices are $p[i] = 1[i] - 1[i+1]$.

model_type = zernike

The Zernike polynomial mode functions are only defined in two dimensions and are given by the functions $U_{n,m}$ as defined in eq 10 of sec 9.2.1 of Born and Wolf.

model_type = fourier

The modes are indexed by the vector $k[]$ (which has the same length as $x[]$) and the auxiliary index $i$, and the modes are

$f_k^j(x)$

where

$f_k^0(x) = \cos(2 \pi k \cdot x / L)$

and

$f_k^1(x) = \sin(2 \pi k \cdot x / L)$

Only one half of $k$-space needs to be occupied. The standard set of modes are defined to be those for which the first non-zero component of $k[]$ is non-negative. Additionally, the mode $k=0$, $i=1$ is not used.

COMMANDS

"difflmodel " differentiate linear mode superposition model
"fitlmodel " fit for linear superposition of mode function
"generatelmodes " generate lmodel for catalog of points
"generatelmmodelimage " generate realisation of a lmodel as a FITS image

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

difflmodel - differentiate linear mode superposition model

SYNOPSIS

difflmodel [-i]

DESCRIPTION

'difflmodel' reads from stdin a catalogue containing the definition of a linear mode function superposition model or 'lmodel' and sends to stdout a lmodel giving the coefficients of the derivative of the input model function.

For example, given a model of a rank-2 matrix valued function \( f_{ij}(x) \) on an n-dimensional space which we model as:

\[
F_{ij}(x) = \sum_m a_{mij} f_m(x)
\]

the result is a set of mode coefficients \( a'_{mijlm} \) such that

\[
F'_{ijl}(x) = \sum_m a'_{mijl} f_m(x) = \frac{d F_{ij}}{d x_l}
\]

With -i option we perform the inverse operation - i.e. we compute the integral of the input model.

It only works for polynomial or Fourier models.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

fitlmodel - fit for linear superposition of mode function

SYNOPSIS

fitlmodel xname aname modeldefinition [options...]

DESCRIPTION

'fitlmodel' reads from stdin a catalogue containing at least a 'postion vector' x called xname and some other variable a called aname, which may be a scalar, vector or matrix of arbitrary rank, and fits for a model of a(x) as a linear superposition of a set of mode functions:

\[ a(x) = \sum_{M} a_M f_M(x) \]

where the mode function coefficients \( a_M \) have the same dimensionality as \( a \).

The 'modeldefinition' is a combination of arguments which can be one of:

- `p lmin lmax` Fit a polynomial. The mode functions are labelled by a set of indices \( p[] \) with same length as \( x[] \), the functions are

\[ f_p = x_0^{p_0} x_1^{p_1} \ldots = \text{product } x_i^{p_i} \]

and the order \( l = \sum p_i \) lies in the inclusive interval \( l_{\text{min}}-l_{\text{max}} \).

An alternative parameterisation of the indices is in terms of the order array \( l[i] = l - \sum_{i=0}^{i-1} p[i] \), in terms of which the p-indices are \( p[i] = l[i] - l[i+1] \).

- `z nmin nmax` Fit for Zernike polynomials of order \( n_{\text{min}} \) through \( n_{\text{max}} \) as defined in Born and Wolf.

- `f kmin kmax lbox` Fit a sum of Fourier modes labelled by compound index \( m = k[] \), \( i \)

\[ f_m = \cos(2 \pi k.x / lbox) \quad i = 0 \]
\[ f_m = \sin(2 \pi k.x / lbox) \quad i = 1 \]
where the modulus of the integerised wave number $k$ lies in the range $k_{\text{min}} - k_{\text{max}}$ inclusive.

- $F$ $k_{\text{max}}$ lbox As for -f option save that all modes in hypercube with edges $\pm k_{\text{max}}$.

Other options:

"-c " generate covariance matrix 'covar''

**AUTHOR**

Nick Kaiser --- kaiser@ifa.hawaii.edu
2.14.3 generatelmodel(1)

NAME

generatelmodel - generate lmodel for catalog of points

SYNOPSIS

generatelmodel [-n aname] lmodel

DESCRIPTION

'generatelmodel' reads from stdin a catalogue containing at least a
position vector and a linear mode function superposition model from
'lmodel' and computes the model function for each point in the catalog.

The name and dimension of the position vector must match 'xname' and
'xdim' in the lmodel header.

By default the realised values of the functions will be named as the
'aname' header item in the lmodel header, and will therefore overwrite
any pre-existing object item of that name but you can supply an alternative
with the -n option.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
2.14.4  generatelmodelimage(1)

NAME

generatelmodelimage - generate realisation of a lmodel as a FITS image

SYNOPSIS

generatelmodelimage x1 x2 nx y1 y2 ny ....

DESCRIPTION

'generatelmodelimage' reads from stdin a 'lmodel' and computes a FITS image containing the realisation of the model function on a grid of points spanning the rectangle bounded by x = x1, x2; y = y1, y2; etc and with nx samples in x etc. There must be one triplet of arguments for each dimension of x. For example, for a 2-dimensional x, the command

generatelmodelimage 0 1 512 0 1 512

will generate an image of the model on the unit square with 512 pixels in each dimension.

If the dependent variable a is a scalar then the dimensionality of the image is the same as that of the independent variable x. For a 3-vector x[] for instance

\[ f[iz][iy][ix] = \sum_m a_m f_m(x) \text{ with } x[0] = x1 + ix \times (x2 - x1) / nx \]
\[ x[1] = y1 + iy \times (y2 - y1) / ny \text{ etc.} \]

If a is a matrix a[j0][j1]...[jn] then the image dimensionality is the sum of the rank of a and the length of x, and the pixel values are, for 2-vector x[] say

\[ f[j0][j1]...[jn][iy][ix] = \sum_m a_m[j0][j1]...[jn] f_m(x) \]

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
2.15 cattools_mergecats(1)

NAME

cattools_mergecats - cattools_mergecats tools section

DESCRIPTION

This section contains 'mergecats' and some related tools. 'mergecats' is used for linking together objects in different catalogues on the basis of spatial coincidence. See mergecats man page for more details.

COMMANDS

"cleancat " remove objects with brighter close neighbours
"findtrailobjects " find objects in very narrow satellite trails etc
"getclosepairs " make catalogue of close pairs from 2 input cats
"getisolatedobjects " find isolated objects
"mergecats " merge catalogues of objects by position
"nuketrailobjects " mask satellite trails etc
"unmergecat " extract one member from a merged catalogue

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

cleancat - remove objects with brighter close neighbours

SYNOPSIS

cleancat [options...] dmax

"-u " print this man page
"-x xname " specify name for coords (x)
"-m magname " specify name for magnitude (mag)

DESCRIPTION

'cleancat' reads a catalogue from stdin and writes to stdout a catalogue containing only those objects from the input catalogue which have no brighter neighbours within distance dmax.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

findtrailobjects - find objects in very narrow satellite trails etc

SYNOPSIS

findtrailobjects [options....] dmax

"-u " print this man page
"-x xname " specify name for coords (x)
"-n nphi " number of bins for histogram (512)

DESCRIPTION

'findtrailobjects' reads a catalogue from stdin which must contain a 2 vector 'x' (substitute some other name with -x option). For each object it computes a histogram of angles to neighbours within distance dmax, where angle lies in domain 0 - PI. It outputs a lc binary format catalogue containing

"x " the position

"nmax " the count in the highest histogram bin

"phi " the angle of the highest histogram bin With judicious choice of parameters and filtering, this will detect satellite trails, diffraction spikes around stars etc.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

getchosepairs - make catalogue of close pairs from 2 input cats

SYNOPSIS

getclosepairs [options....] dmax cat1 cat2

"-u " print this man page

"-x xname " specify name for coords (x)

DESCRIPTION

'getclosepairs' reads two catalogues cat1, cat2 and writes to stdout a catalogue containing pairs with |separation| < dmax Output cat has object items with the same names as in the input cats (which must have identical object items) but each object is a 2-vector formed from a pair of input objects. It works by installing the objects from the first cat into a grid of linked lists, and it then reads the objects from the second cat and outputs a pair-object for each pair with separation < dmax.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
2.15.4 getisolatedobjects(1)

NAME

getisolatedobjects - find isolated objects

SYNOPSIS

getisolatedobjects [options...] dmax referencecat

"-u " print this man page
"-x xname " specify name for coords (x)
"-e " don't exclude zero separation objects

DESCRIPTION

'getisolatedobjects' reads a source catalogue from stdin, a reference catalogue from 'referencecat' and writes to stdout a catalogue containing objects from the source cat which have no reference neighbour with |separation| < d. It works by installing the objects from the reference cat into a grid of linked lists, and it then processes the objects from the source cat sequentially. The -e option is useful if one wants to find the objects in a cat which have no close neighbours in the same catalogue. To do this simply use the source cat as reference cat.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
2.15.5  mergecats(1)

NAME

mergecats - merge catalogues of objects by position

SYNOPSIS

mergecats [options...] d a.cat b.cat ....

DESCRIPTION

'mergecats' reads N catalogues of objects and outputs a single merged
catalogue of objects whose positions match to within tolerance d.

We first read all the catalogues and then for each object in turn construct
an N-tuple consisting of it and any neighbours which meet the positional
tolerance criterion. We then rank the N-tuples in order of quality
of match (an N-tuple with all slots filled ranks higher than one with
one empty slot etc., otherwise rank is the sum of the N (N - 1) / 2
separations). We then output the N-tuples (as objects with same named
items as the input catalogue but where each item is a N-vector of the
input values) in order of decreasing rank, but only using the objects
which were not contained in a previously output N-tuple.

By default, mergecats will only output complete ntuplets (i.e those
with detections in all input catalogues).

The idea here is that if one has three input catalogues containing
positionally coincidental objects B,V,I say, plus an extra nearby neighbour
N detected in B only, then the algorithm will construct four triplets
BVI, VIB, IBV and NVI, it will then output whichever of the first 3
triples is tightest and then output an extra object N-- with two empty
slots.

For efficiency we read the objects from each catalogue into a checkerboard
grid of null terminated linked lists of objects. Options are:

-x xname Supply name for the 2-vector spatial coord ('x')

-n nmin Output only objects with >= nmin detections. With this it
may be useful to use -m option:
-N nmax Output only objects with <= nmax detections. With this it may be useful to use -m option:

-m Prepend the output object items with a mask which is a binary representation of the detections. E.g mask = '10010' indicates a detection in the zeroth and third catalogues of a five catalogue merge. Leading zeros are not printed.

-M mask Output only objects which match the specified mask.

-s Prepend the output cat with a column containing the 'size' of the object (sum of the N (N - 1) / 2 separations.

-d Prepend the output cat with a column containing the number of detections.

-e Exclude zero separation ntuplets

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

nuketrailobjects - mask satellite trails etc

SYNOPSIS

nuketrailobjects [options...] srcfits d w

"-u " print this man page

DESCRIPTION

'nuketrailobjects' reads a catalogue from stdin which must contain a 2 vector 'x' and a scalar phi (possibly generated by 'findtrailobjects') and then sets to MAGIC any pixels in srcimage lying in a strip of width (2 w + 1) straddling the line of half-length d passing through x with orientation phi. Resulting FITS image is sent to stdout.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
2.15.7  unmergecat(1)

NAME

unmergecat - extract one member from a merged catalogue

SYNOPSIS

unmergecat N

DESCRIPTION

'unmergecat' reads a merged catalogue created by 'mergecats'. Ignoring any object items which are not vectors of things (e.g. the 'detection mask' or number of detections column) it creates a new catalogue in which each object item is the (N-1)th element of the item of the same name in the input catalogue.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3 dev(1)

NAME

dev - dev tools section

DESCRIPTION

This section contains some tools which are, or were, under development.

COMMANDS

"findcluster "
"makelist "
"matchpairs "
"matchtriplets "
"mergelists "
"pairs "
"transformlist "
"triplets "

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.1 findcluster(1)

usage: findcluster ncol sigma_1...sigma_ncol [options...] searches for a cluster of particles in multi-dimensional space.

"-b dbox " half-box side d_i = dbox * sigma_i (2.0)

"-w " weight the points by f

"-p " print the points in the box You supply ncol, the number of columns and a smoothing scale sigma_i for each dimension. For each point we calculate f = sum over neighbours of exp(-0.5 sum_i dx_i^2 / sigma_i^2) and find the hottest particle. Finally we average coordinate summing over particles in the box of half-side d_i = dbox * sigma_i. Use -w option to weight points by f. Use -p option to print the points in the box (with f-values in last column).
3.2 makelist(1)

usage: makelist nobj [xmin xmax ymin ymax [seed]] lay down random objects within a rectangle.
3.3 matchpairs(1)

usage: matchpairs a.prs b.prs [dprec [mprec]] reads a pair of pair-files generated by "pairs" and finds pairs of pairs whose separations (dr) match to precision dprec (1.0) and whose mags and mag differences match to precision mprec (0.2). We output dphi = phi_a - phi_b, dx, dy where latter are translation in b-coord system.
3.4 matchtriplets(1)

usage: matchtriplets.c a.trp b.trp [mmax smax] reads a pair of lists of triangles generated by "triplets" and outputs matching pairs. Pairs match if their m=M/L and s=S/L values agree to within precision mmax, smax (0.01, 0.01). Outputs R, phi, dx, dy where R = L_b/L_a is ratio of long side lengths phi = phi_b - phi_a is rotation angle in range (-PI,PI) and (dx,dy) is the translation in b-coords
3.5 **mergelists(1)**

usage: `mergelists list1 list2 [options...]` reads two lists containing (x1, y2) and (x2, y2) particle coords. Construct a N x N checkerboard containing null terminated linked lists of particles from list1 and then use this to find nearest neighbour of each object in list 2 with separation < d. Used for frame registration. Issues a warning if > 1 neighbour is found. Input files must have x,y in 1st two columns. Outputs line1 line2 r_{12} where r_{12} = separation and line1 = line read from 1st file. Options are:

`"-d d "` maximum separation (3)
3.6 pairs(1)

usage: pairs reads a list of x,y,m(agnitude) and outputs list of pairs of points. Outputs x1, y1, m1, dr, phi, dm Where x1, y1, m1 are pos, mag of brightest object and dr, phi, dm are \(|r2-r1|\), atan2(y2-y1, x2-x1), m2-m1.
3.7 transformlist(1)

usage: transformlist a phi dx dy applies transformation
x0 = a (x cos phi - y sin phi) + dx
y0 = a (x sin phi + y cos phi) + dy
3.8 triplets(1)

usage: triplets [dmax] reads a list of coords and finds triangles with max side dmax orders vertices so long, medium and short sides are $L = AB$; $M = BC$; $S = CA$ and prints out $L$, $M$, $S$, $\phi$, $x_A$, $y_A$ where $\phi = \text{atan2}(y_B - y_A, x_B - x_A)$
3.9  imcattools_astrometry(1)

NAME

   imcattools_astrometry - imcattoolsAstrometry tools section

DESCRIPTION

The tools in this section are to help link imcat generated catalogues to high precision astrometry databases.

COMMANDS

   "convertcoords " transform between different sky coordinate systems
   "decimaltodms " convert an angle from decimal degrees to d:m:s format
   "decimaltohms " convert an angle from decimal degrees to h:m:s format
   "dmstodecimal " convert an angle from d:m:s to decimal degrees format
   "getUSNOAcatfromESOhtml " extract lc cat from ESO USNOA web page
   "getwcsinfo " report on WCS fits header info
   "getworldcoords " add world coordinates to a catalogue
   "getxsfromradec " convert from ra, dec to stereographic x-coord
   "hmstodecimal " convert an angle from h:m:s to decimal degrees format
   "readusnoacat " extract catalogue from USNO-A database
   "setwcsinfo " insert world coordinate system info into fits header

AUTHOR

   Nick Kaiser -- kaiser@hawaii.edu
3.9.1 convertcoords(1)

NAME

convertcoords - transform between different sky coordinate systems

SYNOPSIS

convertcoords convtype angle1 angle2

DESCRIPTION

convertcoords invokes Doug Mink’s utilities for conversion between
different sky coordinate systems.

Angles can be expressed in either decimal or [hd]:m:s format.

The string 'convtype' specifies the type of conversion and can be one of

"fk524 " Convert J2000(FK5) to B1950(FK4) coordinates
"fk425 " Convert B1950(FK4) to J2000(FK5) coordinates
"fk42gal " Convert B1950(FK4) to galactic coordinates
"fk52gal " Convert J2000(FK5) to galactic coordinates
"gal2fk4 " Convert galactic coordinates to B1950(FK4)
"gal2fk5 " Convert galactic coordinates to J2000(FK5)

Output angles are expressed in decimal degrees.

AUTHOR

Nick Kaiser kaiser@hawaii.edu
NAME
decimaltodms - convert an angle from decimal degrees to d:m:s format

SYNOPSIS
decimaltodms angle

DESCRIPTION
decimaltodms takes a decimal format angle (in degrees) as argument and generates a string corresponding to that argument in d:m:s format

AUTHOR
Nick Kaiser kaiser@hawaii.edu
3.9.3  decimaltohms(1)

NAME

decimaltohms - convert an angle from decimal degrees to h:m:s format

SYNOPSIS

decimaltohms angle

DESCRIPTION

decimaltohms takes a decimal format angle as argument and generates a string corresponding to that argument in h:m:s format

AUTHOR

Nick Kaiser kaiser@hawaii.edu
3.9.4 dmstodecimal(1)

NAME

dmstodecimal - convert an angle from d:m:s to decimal degrees format

SYNOPSIS

dmstodecimal angle

DESCRIPTION

dmstodecimal takes a string as argument; decipers it using "%d:%d:%lf" format specification to d, m, s; checks that m, s lie in range 0-60 and writes the angle \( \theta = (\text{sign}(d)) \times (\text{fabs}(d) + m / 60.0 + s / 3600.0) \) to standard output.

AUTHOR

Nick Kaiser kaiser@hawaii.edu
3.9.5 getUSNOAcatfromESOhtml(1)

NAME

getUSNOAcatfromESOhtml - extract lc cat from ESO USNOA web page

SYNOPSIS

getUSNOAcatfromESOhtml

DESCRIPTION

getUSNOAcatfromESOhtml reads a html format catalogue obtained from http://archive.eso.org/skycat/servers/usnoa (using the 'short' format with decimal RA, DEC) from stdin and generates a lc format catalogue containing RA, DEC, rmag, bmag, ID which is sent to stdout.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

getwcsinfo - report on WCS fits header info

SYNOPSIS

getwcsinfo

DESCRIPTION

getwcsinfo reads a fits header from stdin, runs Doug Mink’s wcsinit() and reports what, if any, world coordinate system information is present.

AUTHOR

Nick Kaiser kaiser@hawaii.edu
3.9.7 getworldcoords(1)

NAME

getworldcoords - add world coordinates to a catalogue

SYNOPSIS

getworldcoords fitsfile

DESCRIPTION

getworldcoords reads a catalogue from stdin and computes from the pixel coordinates x[] the celestial coordinates RA and DEC according to the transformation model encoded in 'fitsfile' (which need only be a header). RA and DEC are expressed in decimal degrees.

Uses Doug Mink’s wcs utilities code.

AUTHOR

Nick Kaiser kaiser@hawaii.edu
3.9.8 getxsfromradec(1)

NAME

getxsfromradec - convert from ra, dec to stereographic x-coord

SYNOPSIS

getxsfromradec RA0 DEC0 [THETA]

DESCRIPTION

getxsfromradec reads from stdin a catalogue containing celestial coordinates (and perhaps other stuff) and computes the stereographic projection xs with tangent point The orientation is such that xs[0] increases with decreasing RA, and x[1] increases with DEC. The resulting vector is expressed in degrees.

Tangent point args may be decimal (in degrees) or colon separated triplets interpreted as h:m:s for ra and d:m:s for dec.

If the third optional argument THETA is given then the stereographic coords will be rotated anticlockwise through THETA degrees. THETA must be given in decimal notation.

AUTHOR

Nick Kaiser kaiser@hawaii.edu
3.9.9  hms2decimal(1)

NAME

hms2decimal - convert an angle from h:m:s to decimal degrees format

SYNOPSIS

hms2decimal angle

DESCRIPTION

hms2decimal takes a string as argument; deciphers it using "%d:%d:%lf" format specification to h, m, s; checks that m, s lie in range 0-60 and writes the angle \( \theta = (\text{sign}(h)) \times (15 \times \text{fabs}(h) + m / 60.0 + s / 3600.0) \) to standard output.

AUTHOR

Nick Kaiser kaiser@hawaii.edu
NAME

readusnoacat - extract catalogue from USNO-A database

SYNOPSIS

readusnoacat ra dl dec ddec

DESCRIPTION

Readusnoacat extracts a lc-format catalogue from the US Naval Observatory all-sky astrometric catalogue.

All objects with ra lying in the range ra +- (dl / cos(dec)) and dec in the range dec +- ddec (epoch 2000) are extracted. The parameterisation of the range in longitude dl is chosen so that in the small angle approximation one obtains a box of height ddec and width dl both in degrees.

Angle arguments may be given in decimal notation, in which case they are interpreted as degrees, or as colon separated triplets, in which case they are interpreted as h:m:s (for ra, dra) and d:m:s (dec, ddec)

Readusnoacat expects to find an environment variable USNOADIR telling it the directory containing the source catalogue files.

The output catalogue contains the following entries:

"x[2] " sterographic sky coords
"RA " right ascension [deg]
"DEC " declination [deg]
"rmag " red magnitude
"bmag " blue magnitude
"qflag " indicates mag error if set
"gflag " indicates a correlated GSC entry
"fieldno " source image number See the USNO-A readme files for more information on the meaning of the flags.
$x^{[2]}$ is a stereographic projection (in units of degrees).

AUTHOR

Nick Kaiser kaiser@hawaii.edu
NAME

setwcsinfo - insert world coordinate system info into fits header

SYNOPSIS

setwcsinfo CTYPE1 CDELT1 CRVAL1 CRPIX1 CTYPE2 CDELT2 CRVAL2 CRPIX2 CROTA2 [-c | -p] [-e EPOCH EQUINOX]

DESCRIPTION

setwcsinfo reads a fits image from stdin, removes any existing WCS info; inserts new ones; and writes out the data.

Rotation angle CROTA2 must be given in degrees.

With -c option we encode the rotation and scaling in the CDn.m matrix. However, this practice is deprecated.

With -p option we encode the rotation angle in the PCnnmmmm matrix.

Refer to Greisen and Calabretta, 96 for description of FITS WCS convention.

AUTHOR

Nick Kaiser kaiser@hawaii.edu
3.10  imcattools_cosmicrays(1)

NAME

    imcattools_cosmicrays - imcattools_cosmicrays tools section

DESCRIPTION

The tools in this section are for cosmic ray detection and removal.

COMMANDS

"gethotpix " get candidate cosmic ray pixels

"removecrs " replace cosmic rays with MAGIC

AUTHOR

    Nick Kaiser -- kaiser@hawaii.edu
NAME

gethotpix - get candidate cosmic ray pixels

SYNOPSIS

gethotpix fmin rmax

DESCRIPTION

'gethotpix' reads a 2-dimensional FITS image from stdin and sends to stdout a catalogue containing the locations of pixels with f > fmin and an immediate N,S,E or W neighbour with r = f.neighbour / f < rmax.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
3.10.2 removecrs(1)

NAME

removecrs - replace cosmic rays with MAGIC

SYNOPSIS

removecrs [option...]

"-c nuc " threshold for centre pixel (default = 20)
"-s nus " threshold for surrounding pixels (default = 5)
"-n nn " required number of neighbours (default = 4)

DESCRIPTION

'removecrs' replaces cosmic rays with MAGIC.

A pixel is replaced provided \( f > \text{mode} + \text{nuc} \times \sigma \) and at least \( nn \) immediate neighbours have \( f < \text{mode} + \text{nus} \times \sigma \). Reads from stdin and writes to stdout

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
3.11  imcattools.cosmology(1)

NAME

    imcattools.cosmology - imcattools.cosmology tools section

DESCRIPTION

The utilities in this section allow one to calculate physical quantities in FRW cosmologies (frw) and also to generate Navarro-Frenck-White profiles.

COMMANDS

    "frw " compute physical quantities in FRW cosmology

    "nfw " compute Navarro-Frenk-White profile

AUTHOR

    Nick Kaiser -- kaiser@hawaii.edu
3.11.1  frw(1)

NAME

frw - compute physical quantities in FRW cosmology

SYNOPSIS

frw [options....]

where options are:

"-m Omega_matter " present Omega in matter (0.99)
"-l Omega_lambda " present Omega in lambda (0.0)
"-Z Z0 " starting redshift (1.e5)
"-s Zs " max source redshift (9.0)
"-d dlna " logarithmic step size (0.01)
"-u " print this message

DESCRIPTION

'frw' numerically integrates Freidmann equation to obtain physical
quantities:  z = redshift y = a / a0 = 1 / (1 + z) h = H / H0 = sqrt(Omegam / y^ 3 + Omegal + (1 - Omega0) / y^ 2) eta = conformal time zeta = eta0 - eta = conformal lookback time d = solution of ddotdot + 2 h
ddot + (3/2) Omegam d / y^ 3 = 0 dphi = d / y = potential growth factor;

r = radial comoving distance D = comoving angular diameter distance

a = scale factor where Omega0 = Omega_matter + Omega_lambda and We start
integrating at very high Z0 in order to get into pure growing mode,
but only output for Z < Zs.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3.11.2 nfw(1)

NAME

nfw - compute Navarro-Frenk-White profile

SYNOPSIS

nfw [-u] [-v v200] [-x x1 x2 nx] [-z z0] [-c c] [-l]

DESCRIPTION

nfw computes nfw density profile. It outputs a table (in lc format if IMCAT is installed) containing:

"x " dimensionless radius = r/r200
"v " circular velocity
"r " physical radius in cm
"dr " physical delta-radius in cm
"m " mass in M\_solar
"rho " physical density in g/cm\^3
"sigma " physical projected density in g/cm\^2
"theta " angle in arcseconds
"kappa " dimensionless surface density

Options are:

"-u " print this message
"-v v200 " circular velocity at r_200 in km/s (200)
"-x x1 x2 " min and max dimensionless radii (0.01, 100)
"-z z0 " lens redshift (0.2)
"-c c " compactness parameter (10.0)
"-l " use pure lambda cosmology (otherwise use EdeS)

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.12  imcattools_geomview(1)

NAME

imcattools_geomview - imcattools_geomview tools section

DESCRIPTION

The tools in this section can be used to display 3-D FITS files using
geomview's 3-D surface rendering.

These tools will only work if the geomview distribution is installed
along with the xforms library. The location of the geomview library
directory must be supplied as an argument to 'makemake' in order for
these tools to be implemented.

COMMANDS

"fits3Dviewer " use geomview to display 3D FITS files

"geomviewsetup " create links for geomview fits3Dviewer

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.12.1 fits3Dviewer(1)

NAME

fits3Dviewer - use geomview to display 3D FITS files

SYNOPSIS

fits3Dviewer

DESCRIPTION

fits3Dviewer is a geomview module. To use it, move to a working directory, and run the script geomviewsetup. Then start geomview. 'FITS 3D viewer' should appear in the 'External Modules' menu. Clicking on this should bring up a window 'fits3Dviewer' into which you can enter an arbitrary command that, when executed, will generate a 3-D FITS file. The planes of that image will be rendered by geomview as a series of surface plots. This is a useful way to make animations. One can use the controls in the various windows to fine-tune the appearance of the surface and the z-scale. Then, pressing the 'record' followed by the 'play' button will result in a series of files /tmp/fits3Dviewer.nnnn.ppm being created. These can then be converted into e.g. animated GIF format using the gimp.

For example ic -c 1024 1024 'grand 3 *' | smooth -p -1 | smooth -f 0 0 2 | helicalscan 64 2 22.5

SEE ALSO

geomviewsetup

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.12.2 geomviewsetup(1)

NAME

geomviewsetup - create links for geomview fits3Dviewer

SYNOPSIS

geomviewsetup

DESCRIPTION

geomviewsetup creates symbolic links to /home/surf/kaiser/research/imcat/src/imcattools/geom and /home/surf/kaiser/research/imcat/bin/Linux/fits3Dviewer.

SEE ALSO

fits3Dviewer

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.13 imcattools_mosaics(1)

NAME

imcattools_mosaics - imcattools_mosaics tools section

DESCRIPTION

The tools here are to help you register images taken with a multi-CCD 'mosaic' camera.

There are several approaches that I have developed, each of which has advantages and disadvantages.

The tools are mostly scripts; tend to be changed quite often, but can be used as a starting point to generate a customised procedure.

See <a href="fitgeom.html">notes on various fitgeometry tools</a>

These are mostly out-dated.

COMMANDS

"combinestacks " combine a stack of images
"fitgeometry " fit layout of a set of images
"fitgeometry2 " fit layout of a set of images
"fitgeometry3 " fit layout of a set of images
"fitmagshifts " fit for extinction/gain coefficients for a set of exposures
"fitmagshiftsplus " fit for extinction/gain coefficients for a set of exposures
"fitpolymodelmagshift " fit for extinction/gain coefficients for a set of exposures
"fitshifts " solve for residual shifts of chips between exposures
"getfieldgeomfromheaders " determine field layout from RA, DEC in headers
"makemosaiccats" combine cats from a mosaic camera
"makemosaicstack" tool for registering mosaic CCD data
"makemosaicstacks" tool for registering mosaic CCD images
"makestack" generate a stack of image planes
"makewarpedalbum" creates an album of doubly scrunched images

"mergemosaiccats1" perl script to generate a merged catalogue of mosaiccats

"mergemosaiccats2" perl script to generate refined merged mosaiccats

"mosaicfit" fit for transformation coefficients for a set of exposures

"printimages" send many files to printer with pauses

"transformatmosaiccat" apply transformations to a mosaic-cat

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.13.1 combinestacks(1)

NAME

combinestacks - combine a stack of images

SYNOPSIS

combinestacks ix1 ix2 iy1 iy2 srcdir dstdir

DESCRIPTION

run 'combineimages' on a set of stacks of images loop over ix = ix1 ...
... ix2; iy = iy1 ... iy Expects to find source images in srcdir/ix_iy/*.fits
average images placed in dstdir/ix_iy_avg.fits error images placed in
dstdir/ix_iy_sig.fits
NAME

fitgeometry - fit layout of a set of images

SYNOPSIS

fitgeometry

"-c nimages " number of images (8)
"-l lmax " max order for distortion polynomials (1)
"-d parfilebase " basename for transformation parameter files
"-w weightfac " relative weight for pairs involving reference cat (1)

DESCRIPTION

'fitgeometry' reads from stdin the result of merging a set of overlapping catalogues, which must contain at least entries for a pair of spatial coordinate vectors 'x[2][2]' and image numbers c[2]. It then fits a model in which the coordinates of the object on the zeroth image (the 'reference image') are related to those on the c'th image by \( x_0 = x_c + \sum_m a_{cm} f_m(x_c) \) where mode function \( f_m \) are polynomials up to order \( l_{max} \) in \( x \).

The solution is obtained by minimising squared residuals in \( x_0 \) space.

'fitgeometry' can be used in various ways. One application is to generate accurately registered (but generally somewhat distorted) images from a set of dithered images from a mosaic camera. To do this one must first generate a set of overlapping catalogues, one for each chip, by 'growing' the coordinate system for some reference exposure. This is done by finding a low order polynomial transformation which maps successive exposures onto the reference exposure. Once this is done the overlapping catalogues can be merged in pairs and the concatenation of all the merges fed to 'fitgeometry' which will then find a solution for the layout of the chips on some idealised 'detector plane' (whose
coordinates coincide with pixel coordinates on one of the chips ---
the zeroth chip --- which may be chosen arbitrarily). This does not
take out telescope distortion, and (especially with small offsets between
the dithered exposures) the process is liable to introduce additional
distortion.

A second application is to register a set of CCD images to some external
'reference' catalogue --- such as a catalogue derived from the digital
sky survey or from the USNOA catalogue. In this case it may be useful
to use the '-w' option with a small argument to downweight the contribution
to the 'chi-squared' from pairs with one element in the reference catalogue
to reflect their relatively poor precision.

By default, the transformation parameters \( a_{lm} \) are written to stdout
as a concatenation of '.par' format files (which is not particularly
useful), but with the -d option you can specify a basename and fitgeometry
will create a set of files 'parfilebase'.par for \( c = 0, nimages - 1 \), where the first of these contains mode coefficients which are all
zero. You will likely want to make 'parfilebase' a directory, in which
case make sure you terminate it with '/' and remember to 'mkdir' it
first.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

fitgeometry2 - fit layout of a set of images

SYNOPSIS

fitgeometry2 np [options...]

"-l lmax " maximum order for polynomial distortion model
(1)

"-o outputdir " directory for the output files (must exist)

DESCRIPTION

fitgeometry2 reads from stdin the result of merging (using 'mergecats') a set of np 'planes' of catalogues and solves for the location in some 'reference frame coordinates' of objects on the catalogue and also a set of parameters describing the distorted mapping between these reference frame coordinates and pixel coordinates on the images from which the catalogues were derived.

Fitgeometry2 was written to solve the following problem: We have a set of 'data' images from a mosiac camera (which have some uncertain layout of the chips on the detector frame - which need not be static - and also suffer from telescope field distortion and possibly atmospheric refraction). From each these images one can extract typically 100 stars whose positions have a precision of a small fraction < 1/10 of a pixel. These allow one to determine a set of polynomial mappings (with one image taken to define the reference coordinate system) which map these images onto one another to a very high precision. However, this procedure does not remove telescope or atmospheric field distortion and tends to be unstable to introducing further artificial field distortion. To avoid this we incorporate in the fitting a catalogue which derives from the digital sky survey image for example, and for which a good 'plate solution' already exists. Providing this catalogue as the 'reference catalogue' solves our problem and provides one with a mapping from data pixel coordinates to the 'world coordinate system'. A complication of this procedure is that the reference catalogue positions tend to be relatively imprecise, and it is necessary to incorporate this information
in the fitting (as weight factors), so the input catalogues most contain both the measured position and an estimate of the precision.

The catalogues to be merged must contain at least the following items:

"x[2] " spatial coordinate
"i " unique image ID (i = 0 for reference image)
"I " I = 1 -- used internally
"s " sigma^2 = position measurement variance

but will usually contain additional information such as an approximate 'sky coordinate' used by mergecats to link the objects. The result of such a merging is a set of unique particles and their measured positions in each and all of the images in which they appear.

The 'planes' which get merged could be just the set of all images. However, in the case of a mosaic camera an object can only be detected on at most one chip per exposure, so the merged catalogue will be very sparse. It is more efficient to group sets of such 'known to be mutually non-overlapping' catalogues into planes before merging. Since one cannot then infer the 'image number' of a catalogue from its 'plane number' we include the image identifier 'i' in the catalogue.

Fitgeometry2 assumes that coordinates x_{pi} of the p'th of np stars on the i'th image is related to the coordinates r_p in the reference frame by:

r_p = x_{pi} + \sum_m a_{im} f_m(x_{pi}) + e_{pi}

whereas for the reference image (i = 0)

r_p = x_{p0} + e_{p0}

Here the f_m(x) are a set of nm polynomial mode functions and a_{im} denotes the amplitudes of these functions in the distortion of the i'th image. The e_{pi} represent the uncertainty in the position. It finds the set of parameters a_{il} and positions r_p which minimise the 'chi-squared' function:

\chi^2 = \sum_p \left[ \sum_i \left( x_{pi} - \sum_l a_{il} f_l(x_{pi}) - r_p \right)^2 / s_{pi}^2 + (x_{p0} - r_p)^2 / s_{p0} \right]

Since this is quadratic in the a_{il}, r_p the result is a set of nm * (nimages - 1) + np linear equations for each of the 2 spatial coordinate components. The summation over 'i' here does not include the reference image.
catalogue (i=0). For small distortions this is equivalent to the maximum
likelihood solution if we assume that the position errors are gaussian
distributed.

The result is: 1) A set of parameter files ouputdir/i.par, where i
is the image number. By default ouputdir = "geofit2dir", but you can
change this with the -o flag. 2) A set of catalogues ouputdir/i.cat
which contains the x, i, I, s values for the points detected on the
i'th image, as well as the following items:

"rref " the reference catalogue solution (rref = r_p)
"p " the particle number
"ndet " the number of images in which particle p was detected
"refdet " refdet = 1 if particle was detected in the reference
cat
"r " obtained by applying the approximate polynomial model

3) A reference catalogue ouputdir/ref.cat containing the solutions
r_p.

See also <a href="fitgeom.html">notes on various fitgeometry tools</a>

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
3.13.4 fitgeometry3(1)

NAME

fitgeometry3 - fit layout of a set of images

SYNOPSIS

fitgeometry3 [options...]

"-l lmax " maximum order for polynomial distortion model
(1)

"-o outputdir " directory for the output files (must exist)

"-v " 'verbose' mode

DESCRIPTION

fitgeometry3 is very similar to fitgeometry2. The difference is that
fitgeometry3 reads a catalogue (usually a concatenation of catalogues
from a number of images) containing, in addition to the pixel position
'x', image number 'i' and position measurement variance 's', a particle
identifier 'p' which is an integer.

Output files are put in 'geofit3files/' by default, but you can change
this with the -o option.

With -v option, fitgeometry3 will report how many objects read, size
of system of equations etc to stderr.

See <a href="fitgeom.html">notes on various fitgeometry tools</a>

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

fitmagshifts - fit for extinction/gain coefficients for
a set of exposures

SYNOPSIS

fitmagshifts nc ne [options...]

DESCRIPTION

'fitmagshifts' reads a catalogue containing (at least) pairs of magnitudes
stars observed on a mosaic of nc chips with ne exposures, and solves
for any gain variations between chips and any differential extinction
between exposures.

More explicitly, we model the magnitude of a star as measured in the
c'th chip and e'th exposure as: \( m_{ce} = m + m_c + M_e \) where \( m \) is the
true magnitude, and we solve for the gain variations \( m_c \) and the extinctions
\( M_e \) by least squares minimisation (these being measured relative to
the 0th chip and 0th exposure respectively).

We output the coefficients as a pair of lc-format catalogues. By default
these are concatenated to stdout, but can be sent to named files by
using the \(-c\) and \(-e\) options. Options are \(-c\) chipcoefftfile \(-e\) expcoefftfile

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

fitmagshifts - fit for extinction/gain coefficients for a set of exposures

SYNOPSIS

fitmagshifts nc

DESCRIPTION

'fitmagshifts' reads a catalogue containing (at least) pairs of magnitudes mag[2], chip numbers c[2], and positions x[2][2] for a set of reference stars observed on a mosaic of nc chips and solves for any gain variations between chips and gradients thereof.

More explicitly, we model the magnitude of a star as measured at position x, y on the c’th chip as: m_ce = m + m_c + x * m_cx + y * m_cy where m is the true magnitude, and we solve for the coefficients. We set M_0 and m_0 to be zero.

We output the coefficients as a lc-format catalog.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

fitpolymodelmagshift - fit for extinction/gain coefficients
for a set of exposures

SYNOPSIS

fitpolymodelmagshift ne fitorder opname [-outputarray]
   [-noextinction]

DESCRIPTION

'fitpolymodelmagshift' reads a catalogue containing (at least) pairs
of magnitudes mag[2]; detector coords xdet[2][2] and exposure numbers
e[2] for a set of reference stars observed in ne exposures.

It solves for differential extinction between exposures and for a spatial
polynomial magnitude shift.

More explicitly, we model the magnitude of the i’th star as measured
at position xdet and e’th exposure as: m_ei = m + m_e + sum_j m_j f_j(xdet)
where m is the true magnitude and the functions f_j are polynomials
(no DC term).

We solve for the coefficients by least squares. minimisation (these
being measured relative to the 0th chip and 0th exposure respectively).

We output the m_e coefficients as a lc-format catalogue opname.cat.
The coefficients m_j are output as a l-model par file opname.par.

With -outputarray option we output the A-matrix etc to stdout.

With -noextinction option we don’t solve for the extinction terms.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
3.13.8 fitshifts(1)

NAME

fitshifts - solve for residual shifts of chips between exposures

SYNOPSIS

fitshifts nchip nexp srcmergecat

DESCRIPTION

'fitshifts' reads from file srcmergecat (as created by 'mergemosaiccats1' and possibly subsequently filtered to remove bad pairs) and for each chip, finds transformations to map coords in the e’th exposure onto the 0th. 'nchip' is the number of chips in the array 'nexp' is the number of exposures.

AUTHOR

Nick Kaiser -- kaiser@cita.utoronto.ca
3.13.9 getfieldgeomfromheaders(1)

NAME

getfieldgeomfromheaders - determine field layout from RA, DEC in headers

SYNOPSIS

getfieldgeomfromheaders [options]

"-s srcdir " source image directory (images/chip0)
"-S imsuff " suffix for images (.fits)
"-d dstdir " destination directory (fieldgeomdir)
"-r dtheta " add dtheta (deg) to rotangle (0.0)

DESCRIPTION

getfieldgeomfromheaders extracts the RA, DEC, ROTANGLE fields from fits headers for a set of ne images in and srcdir/chip0.imsuff and then for each chip and each exposure computes for a set of points around the chip boundary the pixel coordinates xp, the idealised detector plane coords xd (according to to the nominal chip geometry layout) and approximate sky coordinates xs. The result is a set of catalogues dstdir/c?e?.cat which can be used e.g. to find the region of a digital sky survey image underlying a given chip in a given exposure for initial registration.

The ROTANGLE measures the angle of N relative to the slow axis of the lower bank of chips in the CFH mosaic (increasing in a clockwise sense.

Requires chips.db; fields.db and nominal.db.

WARNING

There are known problems with the encoders on CFH causing these header values to occasionally be in error.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

makemosaiccats - combine cats from a mosaic camera

SYNOPSIS

makemosaiccats [options...]

"-s suf " suffix (stars)
"-d dbdir " data-base dir ( /8k)
"-c catdir " base directory for cats (.)
"-m moscatdir " directory for mosaiccats (./mosaiccats)

DESCRIPTION

'makemosaiccats' reads catalogues with suffix 'suf' which assumed to be contained in catdir/chip? and creates 'mosaiccat's containing position, chip-number, exposure-number and magnitude using information on chip names, geometry and exposure names contained in associative arrays in directory 'dbdir'

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
makemosaicstack(1)

NAME

makemosaicstack - tool for registering mosaic CCD data

SYNOPSIS

makemosaicstack X Y dX dY scalefac destdir imdir [mapmode]

DESCRIPTION

warp images to create a stack for sub-image of size dX, dY at X, Y
It expects to find images in imdir/chip? It is a driver for 'mosaicmap'.
scalefac gives the linear size of the target image pixels relative
to those of the source image. Note that X, Y, dX, dY are given in
units of target image pixel size The optional argument 'mapmode' controls
the type of mapping (see manpage for mosaicmap). Default is mapmode
= 1 for linear interpolation.
NAME

makemosaicstacks - tool for registering mosaic CCD images

SYNOPSIS

makemosaicstacks ix1 ix2 iy1 iy2 dX dY destdir imdir

DESCRIPTION

generate a grid of stacked images of size dX, dY. loop over ix = ix1 ... ix2; iy = iy1 ... iy origin is at (ix - 5) * dX, (iy - 5) * dY stacked images placed in destdir/ix_iy/ Expects to find source images in imdir/chip?
NAME

makestack - generate a stack of image planes

SYNOPSIS

makestack x0 y0 Nx Ny stackdir warpimdir [filter]

DESCRIPTION

'makestack' generates a stack of images of size Nx, Ny with spatial origin x0, y0 and with names 'stackdir'/‘fieldname’.fits where the fieldnames are defined in 'fields.db' (or in fields_filter.db if filter is specified) and the images are in warpimdir/chip?/etc.. You need to define the environment variable TMPDIR where makestack stores temporary files.
3.13.14  makewarpedalbum(1)

NAME

makewarpedalbum - creates an album of doubly scrunched images

SYNOPSIS

makewarpedalbum ix1 ix2 iy1 iy2 expname warpdir dest-dir name

DESCRIPTION

creates an album of doubly scrunched images from warpdir/ix_iy/expname.fits
where ix = ix1 ... ix2 etc Result is placed in destdir/name
NAME

mergemosaiccats1 - perl script to generate a merged catalogue of mosaiccats

SYNOPSIS

mergemosaiccats1 [options...]

DESCRIPTION

mergemosaiccats1 makes a first attempt at merging the mosaic-cats created by 'makemosacicats'. For each pair of mosaic cats we run 'registercats' to get approximate transformations which we apply to the 1st of each pair and then run 'mergecats' with a fairly generous linking length. The results of all these merges is concatenated to the file mergemosaiccats1.out. Options are:

"-d d " linking length (30)

"-D dbdir " directory containing mosaic database ( /Sk)

"-c catdir " directory containing the mosaic cats (.mosaiccats)

"-t " just calculate transformation parameters

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

mergemosaiccats2 - perl script to generate refined merged mosaiccats

SYNOPSIS

mergemosaiccats2 [options...]

DESCRIPTION

mergemosacicats2 makes a second attempt at merging the mosaic-cats created by 'makemosaiccats' using the parameters in the file 'mosaicfit.par' (produced from 'mergemosacicats1.out' using 'mosaicfit'). The resulting merged file contains extra objects values 'r' and 'magc'

which are (pairs of) sky coords and corrected magnitudes respectively,

and which can be used to filter out bad pairs before feeding to

'mosaicfit' to get a refined solution.

Options are: -d d # linking length (5) -D dbdir # directory containing mosaic database (.) -c catdir # directory containing the mosaic cats (./mosaiccats)

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3.13.17  mosaicfit(1)

NAME

mosaicfit - fit for transformation coefficients for a set of exposures

SYNOPSIS

mosaicfit [options...]

DESCRIPTION

'mosaicfit' reads a catalogue containing the result of merging all pairs of 'mosaiccat's (as created by 'mergemosaiccats1' or 'mergemosaiccats2') and which must contain entries for the spatial coords 'x', chip-number 'chip' and exp number 'exp' and magnitude 'mag' from stdin, and fits a linear model in which x,y are related to "detector coords" xd,yd by

xd = x + phi_c * y + dx_c
yd = y - phi_c * x + dy_c

so chips are rotated thru phi_c and displaced by dx_c, dy_c relative to coordinate frame define by chip-0 and where sky coords X_e,Y_e are related to detector coords by

X_e = (1 + alpha * (xd * xd + yd * yd)) xd
Y_e = (1 + alpha * (xd * xd + yd * yd)) yd

and where sky coords in frame defined by exposure-0 are

X = X_e + Phi00_e * X_e + Phi01_e * Y_e + dX_e
Y = Y_e + Phi10_e * X_e + Phi11_e * Y_e + dY_e

which allows for pointing shifts and rotations as well as any scale change or differential refraction. The model is linearised - so only valid for small alpha, phi_c, dx_c, dy_c, Phi_e (dX_e, dY_e can be large
though) Solves by minimizing squared residuals. We also read magnitudes, which we model as:

\[ m_{ce} = m + m_c + M_e \]

where \( m \) is the true magnitude and \( m_c \) and \( M_e \) are magnitude offsets for chip and exposure (relative to chip-0, exp-0). Outputs coefficients in tabular form: alpha

\[
\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\Phi_00 & m & \Phi_01 & m & \Phi_10 & m & \Phi_11 & m \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\phi_n & dx_n & dy_n & m_n & \\
\end{array}
\]

OPTIONS

Options are

"-c Nc " number of chips (7)
"-e Ne " number of exposures (11)
"-n " don’t compute magnitude shifts

See also <a href="./mosaicfitting.ps"> mosaicfitting.ps </a>

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

printimages - send many files to printer with pauses

SYNOPSIS

printimages fits1....

DESCRIPTION

perl script to print a set of images
NAME

transformmosaiccat - apply transformations to a mosaic-cat

SYNOPSIS

transformmosaiccat

DESCRIPTION

transformmosaiccat: apply transformations defined by mosaicfit.par to a mosaiccat generates extra object values \( r \) = transformed coordinate and \( \text{magc} \) = corrected magnitude reads transformation coefficients from 'mosaicfit.par'
NAME

imcattools_objectfinding - imcattools_objectfinding tools section

DESCRIPTION

This section contains tools for finding objects and making charts of them. It also contains 'minisky' which is useful for flattening images. 'surfdens' is used to generate a fits image consisting of 'points of light' from a catalogue.

COMMANDS

"chunkyfp " perform peak finding by dividing image into chunks

"findpeaks " simple object finder

"getbadpix " reject discrepant pixels

"hcat2cat " process hfindpeaks output

"hfindpeaks " hierarchical object finder

"makechart " overlay FITS image with object markers

"makestamps " extract 'postage stamp images'

"minisky " make a smooth local sky flat from cat of minima

"surfdens " generate a fits image from a catalogue

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

chunkyfp - perform peak finding by dividing image into chunks

SYNOPSIS

chunkyfp [options...] fitsfile

where options are

"-s " run silently
"-f nfft " size of FFT (512)
"-m " find minima
"-n rf " use findpeaks -r rf rather than hfindpeaks
"-N nu " minimum nu (4)
"-b nx nx nbox " use nx x ny chunks of size nbox (4 4 400)
"-v saopipe " pipe charts of chunks in iisformat to 'saopipe'
"-S sigma mode " supply mode and sigma by hand

DESCRIPTION

'chunkyfp' divides image up into overlapping chunks and runs hfindpeaks (by default) and then combines the resulting catalogues. The catalogue name is derived from the fitsfile, but with suffix '.cat' (or '.min' in case of -m option). Temporary files will be kept in env variable TEMPDIR so define this environment variable sensibly.

The -v option allows you to monitor the progress using saoimage.

Thanks to Andreas Jaunsen for correcting a bug.

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.14.2 findpeaks(1)

NAME

findpeaks - simple object finder

SYNOPSIS

findpeaks fitsimage [option...] catalogue

"-r sigma " gaussian filter radius (default = 2)
"-n nu " threshold (default = 3)
"-e " find all extrema
"-m " find all minima
"-s sigma mode " sky statistics
"-d " add 2nd derivative information

DESCRIPTION

"findpeaks" fft gaussian filters image and finds peaks above a threshold nu. Creates a catalogue with limited information. With -e or -m option, nu parameter ignored. The position 'x' is measured relative to the bottom left corner of the bottom left pixel (so e.g. a single 'hot' pixel at (ix,iy) = (23,67), would generate an object with x = (23.5, 67.5)

Use -r option to control the smoothing radius. With negative sigma we don’t smooth at all.

'findpeaks outputs

"x[2] " peak position
"fs " smoothed image value at the peak
"nu " significance value
"maximum " true if extremum is a maximum
and with '-d' option it also outputs

"ddfs[2][2] " discretized 2nd derivative at peak

"detddf " determinant

'findpeaks' uses iostream library.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3.14.3 getbadpix(1)

NAME

getbadpix - reject discrepant pixels

SYNOPSIS

getbadpix nu referenceimage [option...]

"-s sigma " rms noise value
"-l " output pixel values as well as positions
"-f d rg " allow for difference in psf shape

DESCRIPTION

By default "getbadpix" reads a fits image from stdin, extracts the value of the image header item with keyword 'SIGMA', and sends to stdout an 'lc' format catalogue containing the a list of pixels for which the pixel value differs from the reference image pixel value by more than nu * SIGMA.

By default, the catalogue contains simply the pixel position 'x[2]', but with the -l option it will also contain the input and reference image pixel values.

If the '-s' option is given the rms noise is read from the following command line argument rather than from the header.

The -f option is provided to allow for the fact that for bright objects such as stars the difference between the source and reference images may greatly exceed the statistical nu * sigma limit, so instead we reject pixels if |f - fref| exceeds the greater of:

nu sigma 2 d fref d fref rg 2 grad(fref) 2 / (fref 2 + sigma 2)

the last two expressions being the difference in the flux of an approximately gaussian star of radius r in the centre and on the edge where d is the assumed fractional change in the gaussian scale length: d = delta rg / rg = d ln(rg).
NAME

hcat2cat - process hfindpeaks output

SYNOPSIS

hcat2cat [options...] hcatfile > catfile

"-a " fire on all local maxima

DESCRIPTION

"hcat2cat" reads a set of peak trajectories in "hcat" format from stdin and applies an algorithm to pick out particular points (e.g. points of max significance). Default is to pick only the most significant local maxima along a peak trajectory, but -a option finds all local maxima. Standard format catalogue goes to stdout THIS IS NOW CALLED AUTOMATICALLY BY hfindpeaks

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

hfindpeaks - hierarchical object finder

SYNOPSIS

hfindpeaks fitsfile [option...] \( \sigma \) catalogue

" -r rf1 rf2 " range of filter radii (0.5 20.0)
"-d dlnrf " step in log(rf) (0.2)
"-l flink " linking parameter(1.0)
"-n nu " significance threshold (4.0)
"-s sigma mode " sky statistics
"-a noiseacf " supply noise autocorrelation function
"-N N1 N2 " set working image size

DESCRIPTION

"hfindpeaks" --- a hierarchical object finder fft gaussian filters
an image with sequence of progressively larger smoothing radius mexican
hat filters and computes the significance \( \nu(x; rf) \) (defined to be
the smoothed field at \( x \) divided by the rms noise fluctuation for smoothing
radius \( rf \)). It then finds peaks of the \( \nu \) field links these together
to construct peak trajectories \( x_{pk}(rf) \). We define an object to be
the point of highest significance along such a trajectory. \( rf1, rf2 \)
are min and max filter radii and we filter with logarithmic steps in
\( rf \) defined by \( dlnrf \). Peaks at adjacent smoothing levels are connected
if their separation is less than \( flink \ast rf \). Use \( -s \) option to supply
sky mode, sigma rather than have them calculated from the image. The
catalogue is in 'lc' format. The position 'x' is measured relative
to the bottom left corner of the bottom left pixel (so e.g. a single
'hot' pixel at (ix,iy) = (23,67), would generate an object with \( x =\)
(23.5, 67.5) The rms noise at smoothing scale \( rf \) is computed assuming
that the noise is incoherent. If the noise is correlated (from resampling say) you can supply a noise psf with the -p option.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

makechart - overlay FITS image with object markers

SYNOPSIS

makechart [option...] ¡ catfile ¥ fitsout

"-r rname a " draw or zap disk  
"-c " draw 16-32 pixel collar  
"-s sf " scrunch factor (1)  
"-f image " fitsfile  
"-m mask " maskfile  
"-z " zap circles or ellipses around object  
"-e ename efac " draw an ellipse  
"-v circval " paint circles etc this value (MAGIC)

DESCRIPTION

"makechart" draws boxes or crosshairs round objects. Normally uses the
image found in the cat header. -f option forces it to use fitsfile
With -m option it draws rectangles from maskfile which must be an 'lc'
format catalogue containing entries for a pair of position vectors x1[2],
x2[2] for bottom-left and top-right corners respectively. Use '-r
rname a' to draw circle of radius a times the value of the object item
'rname', so use e.g. '-r rh 3' for circles 3 times the half-light
radius. Similarly, with -e option it will draw an ellipse with ellipticity
efac times the object ellipticity and with sqrt(a b) equal to the radius
as calculated above.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3.14.7 makestamps(1)

NAME

makestamps - extract 'postage stamp images'

SYNOPSIS

makestamps [option...]

"-b b " box side (32)
"-n " normalise images
"-M mag0 " normalisation magnitude
"-f fitsfile " specify source fits
"-x xname " name for position 2-vector (x)
"-m magname " name for magnitude (mag)
"-N normvalname " normalise to unit 'normvalname'
"-c " generate catalog with x0[2], x[2], f, i values

DESCRIPTION

"makestamps" creates a set of tiny images of patches of sky around objects in the catalogue. It reads a catalogue containing positions and optionally magnitudes of nobj objects from stdin and writes to stdout a 3-D fits image of dimensions b x b x nobj.

It uses the images named in the catalogue header by default. Use -f option to override this.

With -n option the surface brightness will be scaled by a factor $10^{-0.4 \times (mag - mag0)}$ where $mag0$ is the magnitude of the first object unless you supply a value by hand with -M option.

Use -N option to normalise by dividing by 'normval': e.g. do makestamps -N flux .... to normalise to unit flux.
Use -c option to generate lc-catalog format output with pixel values f, object number i, object coords x0[], and pixel center coords x[] with respect to the object coords.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

minisky - make a smooth local sky flat from cat of minima

SYNOPSIS

minisky [option...] catalogue .fits

"-s n " work with scrunch^ n-ed image (2)

"-r rf " gaussian filter radius in real pixels (64)

DESCRIPTION

"minisky" reads minima from a catalogue on stdin It then creates two coarsely sampled images (chunky pixel side = 2^n real pixels); f1 is number of minima falling in pixel and f2 is sum of their fs values. These are smoothed with gaussian radius rf and then fsky = f2 / f1. This will find low-frequency sky variations (as well as any v low frequency and low surface brightness real objects. Output image needs unscrunching to full resolution

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

surfdens - generate a fits image from a catalogue

SYNOPSIS

surfds [option...]

"-x xname " name for position vector ('x')
"-r x1 x2 y1 y2 " range of coordinates
"-n N1 N2 " image size in pixels
"-w weight " weight points by this value
"-s " generate 16 bit image
"-d " print input lc filter string an quit
"-m " set points to MAGIC value
"-f fitsfile " supply source image

DESCRIPTION

"surfdens" reads a catalogue from stdin and calculates a surface density map by binning counts onto an image. By default it looks for 2-vector entry 'x', gets the image size from the catalogue header, sets the range of coordinates to be 0-N1, 0-N2, and uses unit weight per object. Specify name for weight with -w option. Fits image is output to stdout.

With the -m option, occupied pixels are set to MAGIC value and -w option, if present, is ignored.

With the -f option we initialise the image to 'fitsfile' and the -n and -s options, if present, are ignored (the output image inheriting the pixel type of the source image.

NOTES
If you want to use an rpn expression in the input lc filter you will need to double quote it. E.g.: surfdens -w "'w = %x[0]'" to get a weight proportional to x-position. To use a pair of scalar catalogue object values do e.g.: surfdens -x 'fs nu'.

See also 'makedensity' which can handle arbitrary dimension FITS file and position variables.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3.15  imcattools_optics(1)

NAME

imcattools_optics - imcattools_optics tools section

DESCRIPTION

The tools in this section can be used to generate optical transfer functions, point spread functions etc.

COMMANDS

"make2Dotf " make 2-D fast guiding otf with offset guide star
"makeotf " make fast guiding otf
"makevonkarmanS " von Karman structure function
"otftopsf " compute PSF from makeotf output
"phasetopsf " computes PSF from phase screen FITS image
"stackpsfs " read a fits stream of PSFs and average after recentering
"trackpeak " read stream of PSF images and track cell averaged peak

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.15.1 make2Dotf(1)

NAME

make2Dotf - make 2-D fast guiding otf with offset guide star

SYNOPSIS

make2Dotf [-N N] [-R r_outer] [-r r0] [-D D] [-o opfile]
[-z zmax] [-i imname]

DESCRIPTION
'make2Dotf' computes the OTF gk(z) for perfect fast guiding.
Options are

"-N N " image size in pixels (512)
"-r r0 " Fried length in m (0.2)
"-D D " telescope diameter (1.0)
"-z zmax " upper limit for integerized z (N/2)
"-d dz " step in integer z (2)
"-p nphi " number of rays in azimuthal angle (5)
"-g rg " distance to guide star (1.0)

Output is a fits image of the OTF.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
3.15.2 makeotf(1)

NAME

makeotf - make fast guiding otf

SYNOPSIS

makeotf [-N N] [-R r_outer] [-r r0] [-D D] [-o opfile] [-z zmax] [-i imname]

DESCRIPTION

'makeotf' computes the OTF g\(k(z)\) for perfect fast guiding.

Options are

"-N N " image size in pixels (256)

"-R r_outer " outer scale in m (infinite)

"-r r0 " Fried length in m (0.4)

"-D D " telescope diameter (1.6)

"-e e " obscuration = D_2 / D (0.0)

"-o opfile " output file

"-z zmax " upper limit for integerized z (N/2)

"-d dz " step in integer z (1)

"-i imname " output image 'imname' and exit

"-g xg yg " location of the guide star

Output is a table (in lc or human readable format) containing:

"z " argument of transfer function (in m)

"S(z) " atmospheric phase structure function

"gknatl " atmospheric OTF = exp(-S/2)
"gkdiff " diffraction limited OTF

"gktlilt " fast guiding OTF

"gkfried " Fried approximation except with -i option in which case it will output FITS image 'imname' which can be one of T, S, Sg, TT, TTxS, WxS, TxT

Currently, -g option only works with pure Kolmogorov (infinite outer scale) spectrum.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
3.15.3 makevonkarmanS(1)

NAME

makevonkarmanS - von Karman structure function

SYNOPSIS

makevonkarmanS R r0 dz Nz [-N N] [-d dy] [-y ystar]

DESCRIPTION

'makevonkarmanS' computes the function

\[ S(z) = r0^{(5/3)} z^{(-5/3)} \sum dy \frac{y}{(y^2 + (2 \pi z / R)^2)^{11/6}} \left(1 - J_0(y) \exp\left(-\frac{y}{ystar}\right)^2\right) \]

for \( z = iz \cdot dz \) and 0 <= iz < Nz.

Options are:

"-N Ny " number of steps in y (10000)
"-d dy " step size in y (0.01)
"-y ystar " integration parameter (50)

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

otftopsf - compute PSF from makeotf output

SYNOPSIS

otftopsf dtheta thetamax [-l lambda]

DESCRIPTION

'otftopsf' reads a file containing z gk(z) from stdin and computes

\[ g(\theta) = \frac{2 \pi \int dz \, z \, J_0 \left( \frac{2 \pi \theta z}{\lambda} \right) g_k(z)}{\lambda^2} \]

for \( \theta = 0 - \text{thetamax} \) with increment \( d\theta \). The quantity output is \( g(\theta) \times (1 \text{ radian} / 1 \text{ arcsec})^2 \). Command line args are given in units of arcseconds.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

phasetopsf - computes PSF from phase screen FITS image

SYNOPSIS

phasetopsf [-N Nout] [-D D] [-p pupil]

Options are:

"-N Nout " output only central Nout x Nout subimage (N)
"-D D " mirror diameter in pixels (N)
"-p pupilim " supply pupil image.

DESCRIPTION

phasetopsf reads a stream of N x N phase screen samples from stdin and sends to stdout a stream of psfs.

With -p option we ignore any D value supplied with the -D option and we read the pupil from 'pupilim'. If this is 2N x 2N then it is treated as a real amplitude transmission function $0 <= T <= 1$. If it is 2 x 2N x 2N the first plane is considered to be the real amplitude transmission function and the second plane is the phase in radians.

AUTHOR

   Nick Kaiser --- kaiser@hawaii.edu
NAME

stackpsfs - read a fits stream of PSFs and average after recentering

SYNOPSIS

stackpsfs [-n nplanes]

DESCRIPTION

stackpsfs reads a fits stream of PSFs and averages after recentering. It generates a 4 x N2 x N1 image consisting of 4 image planes:

"0 " straight average
"1 " center on centroid
"2 " center on g^2 weighted centroid
"3 " center on peak

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.15.7 trackpeak(1)

NAME

trackpeak - read stream of PSF images and track cell averaged peak

SYNOPSIS

trackpeak [-x nx]

DESCRIPTION

'trackpeak' reads a 3D FITS image consisting of a stream of PSF images, average them with a cell of size nx * nx * nt

Options

"-x nx " cell size in angle (6)

"-t nt " cell size on time (3)

"-n nphotons " number of photons (100)

"-r readnoise " read noise in electrons (2.0)

"-T " only output peak position

"-o opfname " save accumulated psfs as opfname.fits

"-O opfname " ditto, but don’t output the video stream

"-N nframes " only read the first nframes frames

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
3.16  imcattools_orbits(1)

NAME

    imcattools_orbits - imcattools_orbits tools section

DESCRIPTION

The tools in this section implement procedures useful for celestial mechanics. They allow one to convert orbits between Kepler elements and Cartesian phase space (convertorbits). One can compute approximate orbits from 3 observations (laplace3) and make a least squares solution (fitorbit). There are some utilities for generating observations (makeobs_circ, makeobs_inertial).

One can evolve these forward (evolveorbits), compute observable properties (orbs2obs). There is a utility (airmass) for computing the airmass of the observations. One can track the detection status of objects (getdet). Use getmoid to compute the MOID for an orbit. Use orbanim to generate a 3-D FITS image movie of the trajectories. Use r2n to find perpendicular components of a vector relative to a reference direction.

Orbit conversion routines courtesy of Jim Heasley.

COMMANDS

    "airmass " compute minimum air-mass for observation of a target

    "convertorbits " convert between Kepler elements and phase-space coordinates

    "evolveorbits " evolve orbits by two-body or time-centered leapfrog

    "fitorbit " find least squares orbit solution

    "getdet " keeps track of the detections of a set of objects

    "getmoid " compute the MOID from Kepler orbit elements

    "laplace3 " compute an approximate orbit from 3 observations

    "makeobs_circ " make observations for a circular orbit
"makeobs_inertial " make a set of observations for an inertial observatory

"orbanim " generate an animation of orbit trajectories

"r2n " get perpendicular components of a vector

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.16.1 airmass(1)

NAME

airmass - compute minimum air-mass for observation of a target

SYNOPSIS

airmass lon lat min_zd_solar min_zd_lunar dphase nsteps

DESCRIPTION

airmass computes, as a function of time, the minimum angle from zenith and airmass for a target at helio-ecliptic coords lon lat when the solar angle from zenith is at least min_zd_solar degrees and the lunar angle from the zenith is at least min_zd_lunar degrees.

The phase increment dphase is given in degrees relative to winter.

We output the time (relative to midnight in hours) and the and minimum zenith distance and air-mass of the target.

If nsteps is negative, then we output the results for a single time = -nsteps * dphase.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.16.2 convertorbits(1)

NAME

convertorbits - convert between Kepler elements and phase-space coordinates

SYNOPSIS

convertorbits k2c | c2k [-extravs vardefs]

DESCRIPTION

convertorbits converts between Keplerian elements a, e, i, omega, Omega, M and cartesian r[3], v[3].

With -extravs option we carry defined variables along. For example, use -extrvars myscalar:1:myvector:3 to carry along myscalar and myvector[3]

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.16.3 evolveorbits(1)

NAME

    evolveorbits - evolve orbits by two-body or time-centered leapfrog

SYNOPSIS


DESCRIPTION

evolveorbits reads an lc catalog containing a set of positions $r[3]$ and velocities $v[3]$ from stdin, evolves the positions and velocities through nsteps steps of length and writes the results to stdout. By default, it does this by converting from phase-space to Kepler orbital elements $(a, e, i, \omega, \Omega, M)$ and incrementing $M$ and then performing the inverse transformation.

With -writekepler option, we also output the orbital elements.

With -extravars option we carry defined variables along. For example, use -extravars myscalar:1:myvector:3 to carry along myscalar and myvector[3]

With option -tcl we do the evolution using time-centered leapfrog with nsubsteps steps of length $dt / nsubsteps$

Times are given in units of earths dynamical time (approx 58 days). Distances are in AU.

SEE ALSO

maketestpparfile.pl makeobs_circ laplace3 evolve

AUTHOR

    Nick Kaiser --- kaiser@hawaii.edu
NAME

fitorbit - find least squares orbit solution

SYNOPSIS

fitorbit initsolcat np [-tcl dtmax]

DESCRIPTION

fitorbit finds a minimum chi-squared solution for a set of observations given an initial solution.

fitorbit first reads an lc format catalog initsolcat containing the a preliminary solution ra0[3], va0[3], as well as the position re0[3] and velocity ve0[3] of the Earth at t0 the initial time t0 (though these just get passed through. It also contains the number of observations nt.

It then reads a set of nt observations re[3], rho[3], nobs[3], sigma and t from obscat, and finds a minimum chi^ 2 by propagating from the t0 solution and computing the residuals. By default it uses Kepler elements but with -tcl flag will use time-centered leapfrog with maximum timestep dtmax. Minimum is found using Powell's method.

Here re[] is the geocenter, rho[] is the position of the observatory relative to the geocenter and nobs is a unit vector in the direction ra[] - re[].

fitorbit outputs a series of np+2 sets of phase-space coordinates r[], v[] where the first set is the observer, the second is the asteroid solution and subsequent np sets of coordinates are solutions obtained by randomly perturbing the observations with Gaussian distributed errors with rms value sigma. Also output is the minimum value of chi-squared for the solution, and a particle number p (-1 for the observer, 0 for the actual solution and 1 ... np for the virtual asteroids.

SEE ALSO

makeobs_inertial makeobs_circ laplace3 tcl_evolvet t2n tcl_evolvetN orbs2obs
AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

getdet - keeps track of the detections of a set of objects

SYNOPSIS

getdet np

DESCRIPTION

getdet reads from stdin an lc catalog containing variables 'pnum' (particle number) and 'fnum' (frame number) and outputs a sequence of fnum’s and the count 'ndet' of distinct particles that have appeared in the input stream.

It also outputs 'risksum' - the sum of the risk values for the detected particles.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.16.6 getmoid(1)

NAME

getmoid - compute the MOID from Kepler orbit elements

SYNOPSIS

getmoid [-extravars vardefs]

DESCRIPTION

getmoid reads a catalog containing Keplerian elements a, e, i, omega, 
Omega, M and cartesian r[3], v[3].

It then varies M to find the moid and outputs the elements as well 
as the velocity of the particles at MOID.

With -extravars option we carry defined variables along. For example, 
use -extrvars myscalar:1:myvector:3 to carry along myscalar and myvector[3]

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

laplace3 - compute an approximate orbit from 3 observations

SYNOPSIS

laplace3

DESCRIPTION

laplace3 reads an lc format catalog (probably generated by makeobs) from stdin. This catalog must contain 3 lines, each containing at least: t time sigma astrometry precision in radians re[] earth's position ve[] earth's velocity rho[] position of observatory vs earth ra[] asteroid's position va[] asteroid's velocity

It then solves for the phase space coordinates of the asteroid at t=0. Currently it does this graphically.

Times are given in units of earths dynamical time (approx 57.21 days in this model).

Distances are expressed in AU.

SEE ALSO

maketestpscoords.pl makeobs

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.16.8 makeobs_circ(1)

NAME

makeobs_circ - make observations for a circular orbit

SYNOPSIS

makeobs_circ parfile

DESCRIPTION

makeobs_circ generates a catalog containing triplets of observations consisting of: t the time t re[] the position of the earth ve[] velocity of the earth ra[] position of the asteroid va[] velocity of the asteroid va rho[] the position of the observer relative to re[] n[] the direction vector

It generates these according to parameters in 'parfile', another lc cat. These are: pe the phase of the earths orbit (degrees, winter = 0.0) t0 the local time of the central observation (hours, midnight = 0.0) dt the interval between the observations (hours) ra the radius of the asteroid's orbit (AU) ia the inclination (deg) la the longitude of the ascending node (deg) pa the phase of the asteroid in its orbit (deg) sigma the uncertainty astrometry (arcsec) nreal the number of realizations

Both the earth and asteroid are in circular orbits.

Both observatory latitude and tilt of earth's axis are hard-wired to 20 deg.

We use an idealised model where the year has precisely 360 days

Output times are given in units of earths dynamical time (approx 58 days).

Distances are in AU.

SEE ALSO

makeobs_inertial laplace3
AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.16.9  makeobs_inertial(1)

NAME

makeobs_inertial - make a set of observations for an inertial observatory

SYNOPSIS

makeobs_inertial dt sigma

DESCRIPTION

makeobs_inertial reads an lc format catalog from stdin containing at least re[3], ve[3], ra[3], va[3], these being the position and velocity of the earth and the asteroid at t=0. It then generates a catalog containing direction of the asteroid at times t = -dt, 0, +dt and those times. The catalog also contains, the position of the earth re[] and the observatory rho[] wrt the earth at those times.

Times are given in units of earths dynamical time (approx 58 days). Distances are in AU.

SEE ALSO

maketestpscoords.pl laplace3

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

orbanim - generate an animation of orbit trajectories

SYNOPSIS

orbanim np nframes Nx Ny [-r rname] [-d decay_factor] [-b bgfits]

DESCRIPTION

orbanim reads from stdin a catalog containing a set of nframes sets of np particles with coordinates x[2]. It generates a 3D fits image f[nframes][N][N] showing the positions of the particles in the frames.

The -r option is used to modulate the brightness of points. If r is less than 1, a single pixel is painted with the value r and if r is greater than one, a disk of radius r is painted.

With -d option, displayed intensity decays exponentially, with successive frames reduced by a factor decay_factor.

With -b option we use the image 'bgfits' as a background image.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

r2n - get perpendicular components of a vector

SYNOPSIS

r2n rname x y z

DESCRIPTION

r2n reads an lc catalog containing positions rname[3] from stdin, evolves the positions and outputs a two-vector n[2] which are the components of the unit vector |rname| perpendicular to the direction (x, y, z).

The reference vector (x, y, z) need not be normalized.

SEE ALSO

maketestpparfile.pl makeobs_circ laplace3 tcl_evolve

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.17  imcattools_pgplot(1)

NAME

imcattools_pgplot - imcattools_pgplot tools section

DESCRIPTION

'plotcat' is useful for plotting positions, shapes etc of objects in catalogues using pgplot. It is also useful for editing catalogues (with -S, -R, -M options). 'contour' reads a fits file and generates a contour plot. 'profile' actually involves supermongo, but use it with -n option if you don’t have this.

COMMANDS

"contour " create contour plot from fits image
"epsfcompose " generate composite eps files
"plotcat " make 2-d scatter plot from a catalogue

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.17.1 contour(1)

NAME

contour - create contour plot from fits image

SYNOPSIS

contour [option...] fi tsin

"-n nc " number of contour intervals (10)
"-f fmin fmax " range of contour heights
"-d device " pgplot style device ('/xserve')
"-t title " text for title ('contour plot')
"-l xlab ylab " labels for axes ('x' 'y')
"-w lwidth " line width (1)
"-H charheight " character height (1)
"-X X1 X2 Y1 Y2 " range for tick-labels
"-j " switch off justification
"-a axis " pgenv 'axis' value (0)
"-g " add gray-scale background image
"-c colmap " add color-mapped background image
"-L x y text " add a label
"-W " add a wedge for image range
"-C " output mouse-clicks
DESCRIPTION

"contour" produces a contour plot from a fits image using pgplot routines.

If fmin, fmax values are not specified these are calculated from the input image.

It then draws (nc + 1) contours at levels f - fmin + i * df, with df = (fmax - fmin) / nc. By default, it produces output in an X-window on the screen, but use -d option to specify alternative.

Use -c to display a colorised image - colmap can be 0,1,2.

Use -L option to add labels at arbitrary positions.

Use -W option to add a wedge showing the range of image values.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

epsfcompose - generate composite eps files

SYNOPSIS

epsfcompose [-hp] [-m mx my] [-a] epsf1 x1 y1 s1 r1
[epsf2 x2 y2 s2 r2 ....]

DESCRIPTION

'epsfcompose' reads 1 or more eps files epsf1, epsf2,... and generates
a composite in which the origin of epsf_i is shifted by (x_i,y_i) and
each sub-image is scaled so its size is s_i times its original size
and is rotated anticlockwise by an angle r_i degrees.

You can use this to overlay eps files (or rotate/scale individual eps
files) or to generate a composite eps file containing sub-plots on
a grid say.

Note that the spatial origin about which the image will scale, rotate
is where the lower-left corner of the page would be if the eps file
were printed by itself. This most likely does not coincide with the
origin of physical or user coordinates. This can be confusing. Note
also that it is possible to generate eps images where the lower-left
corner has one or both coordinates negative, so the image would necessarily
be cropped if printed. With the -m optional of the images will be
subject to an additional shift of coordinates so that the final lower-left
corner lies at (mx,my).

Note that ghostview displays the whole interior of the bounding box
of an image, and gives cursor location readout.

'epsfcompose' first looks for the bounding boxes of the input epsf
files and computes outputs a header for an epsf file whose bounding
box just contains the shifted, scaled and rotated corners of the input
bounding boxes (with extra shift to get lower-left point wrt the lower
left page corner with -m option). For each input image it saves the
state; does minimalistic set up of fg/bg and default line type, outputs
postscript commands to implement the shift, scaling and rotation followed
by the non-comment lines of the input eps file bracketed by BeginDocument, EndDocument comments.

With the -a option, the x,y origin shifts are multiplied by the width and height of the scaled bounding box of the first eps file to facilitate laying out sets of images on a grid (you can set the spacing to be convenient multiples of the 1st plot size).

Use -hp option if printing on the big HP printer to include the 'setpagesize' directive.

Note that options must be given in order shown in synopsis.

AUTHOR

Nick Kaiser - kaiser@hawaii.edu
NAME

plotcat - make 2-d scatter plot from a catalogue

SYNOPSIS

plotcat [coords] [other options....]

"-t " read table of numbers
"-s sym " set symbol from object value 'sym'
" -c shade " set fill shade from object value 'shade'
"-r rname " use scalar 'rname' for radius
"-e ename " use 2-vector 'ename' for ellipticity
"-V vname " use 2-vector 'vname' for a line-vector
"-d device " pgplot style device ('/xserve')
"-T title " text for title
"-l xlab ylab " labels for axes
"-w lwidth " line width (1)
"-H charheight " character height (1)
"-x x1 x2 y1 y2 " left, right, bottom and top coords
"-X X1 X2 Y1 Y2 " limits for tickmarks if different from above
"-j just " pgenv 'justification' value (1)
"-a axis " pgenv 'axis' value (0)
"-o " draw filled circles, ellipses with outline
"-f fillshade " fill circles, ellipses with fillshade
"-v rfrac " set default radius = rfrac * |x2 - x1| (0.01)
"-y defsym " set default symbol
"-R " edit cat: reject mode
"-S " edit cat: select mode
"-M " edit cat: generate mask file
"-F fitsf cmap " background image
"-m maskfile " plot boxes from a mask file
"-C nc " plot contours over background image
"-L x y text " add a label
"-W " draw wedge (with -F option)
"-h " plot histogram
"-E " plot vertical error bars
"-Q linestyle " connect the points

DESCRIPTION

"plotcat" reads a catalogue (or table of numbers with -t option) from stdin and produces a scatter plot of markers, circles, ellipses etc using pgplot routines. By default, 'plotcat' expects to find a 2-vector 'x' in the catalogue which it uses as the spatial coords, but you can supply alternatives with the 'coords' option. The range of spatial coords can be specified with -x option, otherwise they are calculated from input data. By default, it produces output in an X-window on the screen, but use -d option to specify alternative.

The variables to be used for the spatial position, symbol, colour, circle size, ellipse parameters etc, are specified as the 'coords' argument(s) and following the -r, -s... options. Each of these must be a legal 'lc' variable name which will cause 'lc' to generate an appropriate number of numeric variables. So, to plot radius 'rh' vs magnitude 'mag' from catalogue 'my.cat' say, do

plotcat rh mag < my.cat

or to use the ellipticity values 'e[0]', 'e[1]' do

plotcat e < my.cat
or something more fancy like

```
plotcat 'x = %x 100 200 2 vector vsub' < my.cat
```

to shift the coords by (dy,dy) = (100,200).

Use the -c option to set the greyness of the symbol. If the shade value is negative we plot the point in the color with index = (1 - (int) shade).

Warning: each variable must have a distinct name. Thus, if you have a catalogue containing a single position vector x[], and you want say to plot an ellipse with e = x at each point then you cannot do

```
plotcat -e x < my.cat
```

instead, you should generate a new variable explicitly, as e.g.

```
plotcat -e 'e = %x'
```

For ellipses, the position angle of the ellipse is

```
phi = atan2(e1, e0) / 2
```

and the major and minor axes are

```
a = r (1 + e)
b = r (1 - e)
```

where e = sqrt(e0\^2 + e1\^2), so the 'ellipticity' is really the 'distortion' since e = (1 - b/a) / (1 + b/a).

The -V option allows you to draw a line vector and the default symbol at each point. This cannot be used with -e option.

The default symbol is a small circle (symbol # 22) if npts < 1000, otherwise the default is the smallest dot. Some other useful symbols are

```
"48 + i " the integer i
"65 - 90 " A-Z
"97 - 122 " a-z
```

The -R and -S options allow you to select a subset of points from a catalogue (which will be sent to stdout) using the mouse to create a set of rectangles. Click left button to start a 'rubber band' rectangle and click it again to enter it into rectangle list (use either centre
or right button to ignore this rectangle). Clicking right button within
a rectangle will remove it from the list and click right button outside
of all rectangles to perform the edit and exit. The -R and -S options
do not function properly if you construct the plotted variables as
rpn expressions on the command line. Instead, you should use "1c"
to generate the variables and then pipe the result to plotcat. With
-M option a mask file containing coordinates of lower left (x1,y1)
and upper right (x2,y2) corners of the rectangles is sent to stdout.

Use -F option to supply a fits file for a background image. Use cmap
= -1 for a gray scale or 0, 1, or 2 for a variety of pretty color maps.
Use -C to plot nc contours on top.

Use the -a option to control the drawing of axes and tickmarks etc.
Generally speaking this is passed to pgbox as the AXIS variable. Some
options are.

"-a -2 " no box or tickmarks
"-a -1 " box with no ticks or numbers
"-a 0 " draw box with annotated ticks
"-a 1 " as 0, but draw x=0, y=0 axes too
"-a 2 " as 1, but draw drid too
"-a 10 " x-axis is logarithmic
"-a 20 " y-axis is logarithic
"-a 30 " both axes are logarithmic
"-a 40 " annotate with h/d:m:s with x,y in seconds
"-a 42 " as 40, but with grid

Use -h option to plot a histogram. Cannot be used with e.g. symbol,
errobars, ellipses etc.

Use -E erry to plot vertical errorbars. The two-vector erry must contain
the y-coordinates of the top and bottom of the error bar (in no particular
order).

Use '-Q linestyle' to connect the points with line:

"1 " solid
"2 " dash
"3 " dot-dash
"4 " dotted
"5 " dot-dot-dot-dash

Plotcat creates some temporary files, which it will remove on graceful termination. By default these go in the directory "P_tmpdir" defined in stdio.h, but you can override this by defining an environment variable TMPDIR.

_AUTHOR_

Nick Kaiser --- kaiser@hawaii.edu
3.18  imcattools.photometry(1)

NAME

imcattools.photometry - imcattools.photometry tools section

DESCRIPTION

The tools in this section are provided to add detailed shape, size magnitude information to catalogues of objects detected with the object detection tools 'findpeaks' and 'hfindpeaks'.

COMMANDS

"Getshapes" calculate ellipticities etc. for catalogue of objects

"apphot" perform aperture photometry

"deproject" deproject an image of an assumed spherically symmetric object

"getfitsval" add values from a fits file to a catalogue

"getshapes" calculate ellipticities etc. for catalogue of objects

"getshapes2" calculate ellipticities etc. for catalogue of objects

"getshapes3" calculate ellipticities etc. for catalogue of objects

"getsky" determine model for local sky background around objects

"makedensity" bin catalogue into FITS image

"makegausskernel" calculate the kernel for gaussian psf

"makekernel" calculate the kernel for the polarizability

"makepeff" compute effective polarisability

"makepeff2" compute effective polarisability
"modelpsf " generate a 2D polynomial model for psf
"profile " compute azimuthal average of a FITS file

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.18.1 Getshapes(1)

NAME

gETCHAPES - calculate ellipticities etc. for catalogue of objects

SYNOPSIS

getchapes [options...]

"-r rname " name for window radius (’rg’)
"-m rmult " multiplier: r_window = rmult * rname (1.0)
"-f fitsfile " specify fits file explicitly
"-i " ignore the sky background information.
"-Z r mul " ignore pixels within mul * r of other objects
"-R " always output object

DESCRIPTION

"getshapes" calculates second moments of sky brightness (and related quantities) for objects detected by (h)findpeaks, though possibly after having been processed by getsky and/or apphot. It uses a gaussian window of size determined by flags -r, -m. Use ’-r unity’ for r_window = rmult. It adds the following items to the catalogue:

"qll " trace of flux normalised quadrupole moment matrix
"q[2] " q[a] = M[a][l][m] q[l][m]
"R[2][2] " response to psf anisotropy
"P[2][2] " response to (post seeing) shear

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

apphot - perform aperture photometry

SYNOPSIS

apphot [options...]

"-m mul " r_aperture = mul * r (3.0)
"-r rname " name for aperture radius ('rg')
"-R rap " fixed aperture
"-f fitsfile " specify fits file explicitly
"-i " ignore local sky parameters
"-z zeropoint " zeropoint for magnitude scale
"-Z r mul " ignore pixels within mul * r of other objects
"-M rapmax " maximum aperture for photometry (50)

DESCRIPTION

By default, 'apphot' measures flux and mag for aperture of radius mul * rg as determined by hfindpeaks, though you can use the -r (and optionally the -m option) to choose an alternative, or use the -R option to specify a fixed aperture.

It adds to the catalogue objects the following items

"flux " flux (sum of pixel values) within the aperture
"mag " magnitude = zeropoint - 2.5 * log_10(flux)
"rh " radius within which 1/2 of flux is found
"rp " Petrosian radius
"rql" radius within which 1/4 of flux is found
"rqu" radius within which 3/4 of flux is found
"nbad" number of magic pixels within the aperture
"fmax" highest pixel value within aperture.

The 'Petrosian' radius 'rp' is defined to be the first maximum in the cumulative enclosed flux divided by the radius. In order to avoid assigning an unreasonably small petrosian radii to small objects where the centroid happens to lie very close to the centre of a pixel we 'soften' the radius: \( r \rightarrow \sqrt{r^2 + 0.3^2} \)

Apphot takes all of the pixels whose centres fall within a distance less than \( r_{\text{aperture}} \), sorts them by (softened) distance, and computes the half-light radius, and also the 'quartile' radii rql, rqu which contain 25 and 75 percent of the light respectively. It also outputs a count of 'bad pixels' nbad, which are pixels whose centres fall within the aperture, but are either MAGIC or lie off the image. To compute the magnitude it looks for zeropoint in the input catalogue header unless you override this with -z option.

It looks for fits filename in the header variable 'fits_name' unless you specify alternative with -f option. The argument here can be 'somecommand |' to generate the image on the fly.

It will use the local sky background parameters fb0 and dfb if present, unless you give the -i option. With the '-Z' option, we ignore pixels around other objects if distance is \( \leq r \times \text{mul} \).

Objects with negative fluxes are assigned magnitude -100.0.

In order to get good 'total' magnitudes, it is necessary that you use a good radius (big enough to get nearly all the light, but small enough to avoid counting neighbour object light). This is a tricky problem in general. I have found that 'rg' computed by 'hfindpeaks' provides a good choice (that's why I adopted it as the default), but if you have a catalogue of objects with no decent size parameter a reasonable alternative is to run apphot to compute the petrosian radius using a fixed aperture of say 10 pixels, and then use a suitable multiple (say 2.0) of rp as the aperture for a second pass.

**AUTHOR**

Nick Kaiser --- kaiser@ifa.hawaii.edu
3.18.3 deproject(1)

NAME

deproject - deproject an image of an assumed spherically symmetric object

SYNOPSIS

deproject rmin rmax nbins [options....]

"-c xc yc " spatial origin (N1/2 + 0.5, N2/2 + 0.5)

"-s d " spacing of points for volume integration (0.1)

DESCRIPTION

"deproject" computes deprojection of an assumed spherically symmetric structure from a FITS image in the style of Fabian et al.

The model is \( f_{2D} = \int dz f_{3D}(\sqrt{r^2 + z^2}) \).

Input image is read from stdin through profile (which generates an lc-format version of the azimuthal sum of the image \( F(r) \) in nbins log-spaced shells ranging from rmin to rmax.

We then work inward from outermost shell, computing first the volumes \( V[ir] = 2\pi \int dz \int drp \) where integrals are over volumes such that \( r = \sqrt{rp^2 + z^2} \) falls in some given bin \( r \), and where we use spacing of \( d \) times \( r \) to perform the integrals.

We then compute: \( f_{3D[i]} = (F[i] - \sum_{j>i} V[j] f[j]) / V[i] \)

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.18.4 getfitsval(1)

NAME

getfitsval - add values from a fits file to a catalogue

SYNOPSIS

getfitsval fitsfile valname [options...]

"-x xname " spatial coordinate ('x')

"-m magicval " magic value (-32768)

DESCRIPTION

'getfitsval' reads a catalogue from stdin, which must contain at least some coordinate 'x' and adds an entry named 'valname' with value derived from the FITS file 'fitsfile' and writes result to stdout.

The dimensionality of the new entry depends on the dimensionality of the x-coordinate and the image f. If these match then the new entry is a scalar; for a two dimensional coordinate and a two dimensional image for example, the output value is v = f[iy][ix] where iy = (int) floor(x[1]) ix = (int) floor(x[0]) If the FITS image is of higher dimension, then the new value will be a vector or matrix. For example, with a 2-vector x and 3D image f[iz][iy][ix] with dimensions Nz, Ny, Nx, the output value v will be a vector of size v[Nz], with values v[iz] = f[iz][iy][ix] Similarly, if f has five dimensions say, and x is a 3-vector, then v is a matrix of dimensions v[N5][N4].

By default, getfitsval looks for a spatial coordinate named 'x' but you can substitute another name with the '-x' option.

Points which lie in pixels with MAGIC value or which fall outside the image are assigned the value 'magicval'.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.18.5 getshapes(1)

NAME

getshapes - calculate ellipticities etc. for catalogue of objects

SYNOPSIS

getshapes [options...]

"-r rname " name for window radius ('rg')
"-m rmult " multiplier: r_window = rmult * rname (1.0)
"-f fitsfile " specify fits file explicitly
"-i " ignore the sky background information.
"-Z r mul " ignore pixels within mul * r of other objects
"-R " always output object

DESCRIPTION

"getshapes" calculates second moments of sky brightness (and related quantities) for objects detected by (h)findpeaks, though possibly after having been processed by getsky and/or apphot. It uses a gaussian window of size determined by flags -r, -m. Use '-r unity' for r_window = rmult. It adds the following items to the catalogue:

"e[2] " ellipticity or polarisation
"psm[2][2] " 'smear polarizability tensor'
"psh[2][2] " 'shear polarizability tensor'
"d[2] " centroid Sometimes trace of quadrupole moment tensor is negative. The default behaviour is not to output such objects, but with -R option (for 'Rambo mode') we output an object with zero polarization and polarizability.
AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3.18.6 getshapes2(1)

NAME

getshapes2 - calculate ellipticities etc. for catalogue of objects

SYNOPSIS

getshapes2 psfimage rf [options...]

"-f fitsfile " specify fits file explicitly
"-i " ignore the sky background information.
"-Z r mul " ignore pixels within mul * r of other objects

DESCRIPTION

"getshapes2" is a catalogue filter which calculates polarisation and polarisability for objects.

It requires that the input catalogue contain at least position vector 'x[2]', and a (aperture) flux 'flux'.

It uses 'makekernel' to compute the kernels W[i] and K[i][j] from the psf supplied in the fits image 'psfimage' and then computes F = sum f[y][x] w[y][x] q[l] = sum f[y][x] W[l][y][x] / F P[l][m] = sum f[y][x] K[l][m][y][x] / F R[l][m] = sum f[y][x] R[m][y][x] / F for l = 0,1,2 and m = 1,2

If sky background values (from getsky) are present they will be used (unless you give -i flag).

Use -Z option to zap circles around neighbouring objects.

By default the source image name is taken from the catalogue header item 'fits.name', but you can specify alternative explicitly with -f option.

AUTHOR

170
3.18.7 getshapes3(1)

NAME

getshapes3 - calculate ellipticities etc. for catalogue of objects

SYNOPSIS

getshapes3 f_image fs_image psf_image rf [options...]

"-i " ignore the sky background information.

DESCRIPTION

"getshapes3" is a catalogue filter which calculates polarisation and polarisability for objects.

It requires that the input catalogue contain at least position vector 'x[2]', and a (aperture) flux 'flux'.

It uses 'makekernel' to compute the kernels W[i], K[i][j], ZZ[i][j] from the psf supplied in the fits image 'psf_image' and then computes

\[
F = \sum f[y][x] \cdot w[y][x]
\]
\[
q_0 = \sum f[y][x] \cdot W[0][y][x] / F
\]
\[
q[l] = \sum f[y][x] \cdot W[l+1][y][x] / F
\]
\[
P_0[m] = \sum f[y][x] \cdot K[0][1+m][y][x] / F
\]
\[
P[l][m] = \sum f[y][x] \cdot K[l+1][1+m][y][x] / F
\]
\[
Z[l][m] = \sum f[y][x] \cdot ZZ[1+l][1+m][y][x] / F
\]
\[
R[m] = \sum f[y][x] \cdot R[m][y][x] / F
\]
\[
Psh[l][m] = 2 \cdot q_0 \cdot \delta[l][m] - Z[l][m] / 2 \cdot rf^2 + 2 \cdot q[l] \cdot q[m] / rf^2
\]
\[
Psh[l][m] = (rf^2 - 2 \cdot q_0) \cdot \delta[l][m] + Z[l][m] / 4 \cdot rf^2 - q[l] \cdot q[m] / rf^2
\]
for \( l,m = 0,1 \)

If sky background values (from getsky) are present they will be used (unless you give -i flag).

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

getsky - determine model for local sky background around objects

SYNOPSIS

getsky [options...]

"-m mode " mode for background sky calculation (1)
"-a a1 a2 " inner and outer radii (default 16 32)
"-f fitsfile " specify fits file explicitly
"-Z r mul " ignore pixels within mul * r of other objects

DESCRIPTION

"getsky" determines the local sky background for objects detected by (h)findpeaks. We determine a mean + gradient model using modal sky value for each of four quadrants of an annulus around the object. The inner and outer annulus radii are a1, and a2 times:

"1 pixel " for mode = 1
"rg " for mode = 2
"rh " for mode = 3
"rp " for mode = 4

The latter two require that we have already run analyse once. Getsky adds entries fb0, dfb[2] to the catalogue and sets the header value 'has_sky'. By default we use the image named in the catalogue, but you can specify an alternative (if you want to use one where the objects have been 'zapped' with makechart for instance). We require at least 16 good pixels in a viable quadrant. Our strategy is:

"0 good quadrants: " mean and gradient = 0.0
"1,2 good quadrants: " only calculate mean, gradient = 0

"3 good quadrants: " determine mean and gradient

"4 good quadrants: " remove most extreme quadrant.

With the '-Z' option, we ignore pixels around other objects if distance is <= r * mul.

**AUTHOR**

Nick Kaiser --- kaiser@cita.utoronto.ca
3.18.9 makedensity(1)

NAME

makedensity - bin catalogue into FITS image

SYNOPSIS

makedensity r x1 x2 nx ..... [-v val] [-c]

DESCRIPTION

'makedensity' reads a catalogue from stdin, which must contain at least some N >=1 dimensional vector and sums the counts of objects (or with the -v option sums some specified object value) into bins in a floating point FITS image which is sent to stdout.

The first argument is the name of the coordinate vector. This is then followed by N triplets giving, starting with the fastest coordinate dimension, the range of dimension to be mapped and the corresponding number of pixels in the output image.

Use the -v option to sum some numerical object value named 'val'. If val is a scalar then the output image will have the same dimensionality as the coordinate vector, but if val is a vector or matrix then the output image will be of higher dimensionality and contain a set of images containing the the sums of the various components of val. For example, if the input catalogue contains a three dimensional coordinate r[3] = x,y,z, and a M1 x M2 matrix valued quantity m[M2][M1], then the result is a 5-dimensional image f[N5][N4][N3][N2][N1] with N5 = M2, N4 = M1 (and N3, N2, N1 given in the command line arguments).

With the -c option we assign the 'charge' (v or unity) to four neighbouring pixels. This is done in such a way that if dx = (x2 - x1) / nx etc then a point with x = ix + dx / dx is assigned entirely to the pixel with index ix. The model here is that the zeroth pixel extends from x = 0 to x = dx, etc.

The coordinate ranges are stored as FITS header records named x0min, x0max, x1min, x1max, ....
AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.18.10 makegausskernel(1)

NAME

makegausskernel - calculate the kernel for gaussian psf

SYNOPSIS

makegausskernel alpha beta sigma rf [N1 N2]

DESCRIPTION

'makegausskernel' computes kernel $K_{\alpha \beta}$ for the polarisation for gaussian psf. rf is the scale length for the gaussian weight function. sigma is the scale length for the psf. By default $N1 - N2 = 256$. 
NAME

makekernel - calculate the kernel for the polarizability

SYNOPSIS

makekernel [-w rf | -W alpha rf | -K alpha beta rf | -Z alpha beta rf | -R alpha rf]

DESCRIPTION

'makekernel' reads a psf image from stdin and computes various kernels for the polarisation

Options are:

"-u " print this message

"-w rf " compute w

"-W alpha rf " compute W_alpha

"-K alpha beta rf " compute K_alpha beta

"-Z alpha beta rf " compute Z_alpha beta

"-R alpha rf " compute R_alpha

rf is the scale length for the gaussian weight function.

With -w, -W, -Z options, only the size of the psf image is used.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

makepeff - compute effective polarisability

SYNOPSIS

makepeff [options...]

DESCRIPTION

'makepeff' first reads from stdin a catalogue which must contain at least the following entries:

"F " windowed flux
"q0 " size
"q[2] " polarisation
"R[2] " flux response
"P0[2] " size response

"P[2][2] " polarisation response as created by 'getshapes2'. It bins appropriate combinations of these in a cubical array in F, p0, q space and computes P_effective.

OPTIONS

Options are

"-u " print this message

"-F logF1 logF2 nF " range of log_10 F and number of bins defaults to 2 3.6 8

"-q q1 q2 nq " range of |q| and number of bins defaults to 0.0 0.5 32
"-Q q01 q02 nq0 " range of q0 and number of bins defaults to 2.5 3.5 32

OUTPUT
Output is a 3-plane FITS file with planes containing

"plane 0 n " number of objects in cell = sum 1

"plane 1 nP " sum P_eff

"plane 2 nq " sum sqrt(q.q)

"plane 3 nPbar " sum P

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

makepeff2 - compute effective polarisability

SYNOPSIS

makepeff2 [options...]

DESCRIPTION

'makepeff2' first reads from stdin a catalogue which must contain at least the following entries:

"F " windowed flux
"q0 " size
"q[2] " polarisation
"R[2] " flux response
"P0[2] " size response
"P[2][2] " polarisation response as created by 'getshapes2'. It bins appropriate combinations of these in a cubical array in F, p0, q space and computes P_effective.

OPTIONS

Options are

"-u " print this message

"-F logF1 logF2 nF " range of log_{10} F and number of bins # defaults to 2 3.6 8

"-q q1 q2 nq " range of \(|q|\) and number of bins # defaults to 0.0 0.5 32
"-Q q01 q02 nq0 " range of q0 and number of bins # defaults to 2.5 3.5 32

OUTPUT

Output is a multi-plane FITS file with planes containing

-plane 0 n " number of objects in cell = sum 1
-plan 1-2 nR " sum R[0], R[1]
-plan 3-4 nP0 " sum P0[0], P0[1]
-plan 5-8 nP " sum P[0][0], P[0][1], P[1][0], P[1][1]

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

`modelpsf` - generate a 2D polynomial model for psf

SYNOPSIS

`modelpsf starcat fitsimage [options...]`

"-n N " postage stamp image size (32)
"-l lmax " max order for fit (1)
"-g " generate psf model
"-s " read stamps image
"-w f_bg " use background in weighting

DESCRIPTION

`modelpsf` first reads a catalogue of stars from `starcat` and fits for the following model for the psf:

\[ \text{fpsf}(x; x_{\text{obj}}) = \sum_l \text{fpsf}_l(x) f_l(x_{\text{obj}}) \]

where \(f_l(x_{\text{obj}})\) are polynomial mode functions and the \(\text{fpsf}_l(x)\) are the image valued 'mode amplitudes'.

The stars need to be normalised in flux, so it is necessary that the input catalogues contain an entry 'flux' in addition to the position vector 'x[2]'.

By default it generates and reads a postage stamp album image of these stars from `fitsimage`. We then do a least squares fit to obtain a final image model psf amplitudes \(\text{fpsf}_l(x)\).

With `-s` option `fitsimage` is supplied as a ready made stamp album.

By default, objects receive weight proportional to their flux, which is optimum in the limit of negligible sky background. Use `-w` option...
to supply a background value (input image is assumed to have had background subtracted however), and then weight objects in proportion to $\text{flux}^2 / (f + f_{bg})$.

With the -g option, the fits image is instead interpreted as the $n\text{modes} \times N \times N$ model psf image and we generate an $n\text{stars} \times N \times N$ image containing the synthesised models. In this mode the -n flag is ignored and the input catalogue need contain only a position vector $x[2]$.

**AUTHOR**

Nick Kaiser --- kaiser@hawaii.edu
NAME

profile - compute azimuthal average of a FITS file

SYNOPSIS

profile [option....] ¡ fitsin

"-c xc yc " spatial origin (N1/2 + 0.5, N2/2 + 0.5)
"-n " does nothing
"-r rmax " maximum radius (N1 / 2)
"-d dr " linear steps - size dr (1.0)
"-l r1 r2 nbins " logarithmic steps

DESCRIPTION

"profile" computes the azimuthally averaged profile of an image. The output is an lc format catalogue containing i r f fsum npix fmin fmax where i = 0,1,2...., r = i * dr, npix is the number of pixels for which the distance R from the centre of the pixel to the point (xc,yc) lies in the range r <= R < r + 1, fsum is the summed image values over those pixels and f = fsum / npix.

By default the spatial origin is taken to be the centre of the pixel (N1/2, N2/2) and the maximum radius is half the width of the image: rmax = N1 / 2

Use the -l option to do logarithmically spaced bins.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

imcattools.photometry_fitting - imcattools.photometry_fitting

tools section

DESCRIPTION

'fitobject' allows you to fit a model for a single object to a small
fits file...

COMMANDS

"betafit " fit image to beta model
"fitobject " fit FITS file to generalised Gaussian model
"gaussfit " fit image to gaussian ellipsoid object model
"testscript "

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

betafit - fit image to beta model

SYNOPSIS

betafit [options....]

"-u " print this message
"-f " output the model as fits image
"-N N " number of micropixels per pixel (1.e4)
"-v " verbose mode
"-r rc " initial core radius (1.0)
"-b beta " assumed beta

DESCRIPTION

"betafit" reads a fits image of counts n[][] from stdin and fits this to a simple beta model for mean counts with central value f0; core radius rc; index beta centered on x0, y0:

\[ n_{\text{model}} = n_0 \times (1 + \frac{r^2}{rc^2})^{-\beta} \]

where \( n_0 = f_0 \times f \) and \( r^2 = (x - x_0)^2 + (y - y_0)^2 \)

by minimising sum_pixels \( N \times f - \text{sum_pixels} (n[iy][ix] \times \log(f)) \)

Where \( N \) is the number of micropixels per real pixel.

output is

\( N \times f_0, x_0, y_0, rc, \beta \)

We start with \( x_0, y_0 \) given by mean of \( x, y \) for input counts and core radius unity by default and with initial \( f = \text{sum n} / (N \times rc^2) \)
BUGS

Couldn’t get it to work stably with beta as free parameter.

AUTHOR

Nick Kaiser: kaiser@ifa.hawaii.edu
NAME

fit1object - fit FITS file to generalised Gaussian model

SYNOPSIS

fit1object [options....]

"-u " print this message

DESCRIPTION

"fit1object" reads a fits image f from stdin and fits this to a simple model object of the form

\[ f_{model} = f_0 \exp\left(-0.5 \times [q_{ij} (r-d)_i (r-d)_j]^\alpha \right) \]

by minimising

\[ \text{sum}_r \text{fabs}(f - f_{model})^\beta \]

By default, \( \alpha = 1 \) and \( \beta = 2 \), so the program is fitting a 2-dimensional gaussian by least squares. The output is in the form: \( f_0, d_x, d_y, q_{xx}, q_{xy}, q_{yy} \) the matrix \( q_{ij} \) being taken to be symmetric. You can modify the behaviour with the following parameters:

"-a alpha " set slope of exponential arg

"-b beta " type of fit

"-i " output inverse of \( q_{ij} \)

"-A " output a, b, phi (TBI)

"-f " output the model as fits image The position is measured relative to the bottom left corner of the bottom left pixel (so e.g. a single 'hot' pixel at \((ix, iy) = (23, 67)\), would generate an object with \( x = (23.5, 67.5) \)

AUTHOR

190
NAME

gaussfit - fit image to gaussian ellipsoid object model

SYNOPSIS

gaussfit [options....]

"-u " print this message
"-f " output the model as fits image
"-n n " fit n gaussians

DESCRIPTION

"gaussfit" reads a fits image f from stdin and fits to this a simple gaussian ellipsoid model with central value f0; semi-axes a,b; position angle phi; at x0, y0:

\[
f(x,y) = f0 \times \exp\left(-\frac{(X/a)^2 + (Y/b)^2}{2}\right)
\]

where

\[
X = (x - x0) \cos(\phi) + (y - y0) \sin(\phi) \quad \text{and} \quad Y = (y - y0) \cos(\phi) - (x - x0) \sin(\phi)
\]

Output is: f0 x0 y0 a b phi

AUTHOR

Nick Kaiser: kaiser@ifa.hawaii.edu
3.18.20 testscript(1)
NAME

imcattools_psfcorrection - imcattools_psfcorrection tools
section

DESCRIPTION

These tools allow you to measure, model and correct for anisotropy of the psf. They implement the analysis of KSB.

COMMANDS

"ecorrect " correct ellipticities for psf anisotropy
"ecorrect2 " psf correction using a smoothed e/n image
"efit " generate model for stellar ellipticities

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

ecorrect - correct ellipticities for psf anisotropy

SYNOPSIS

ecorrect [option...]

"-f efitdata " file for stellar e / psm model parameters
(efit.out)

DESCRIPTION

"ecorrect" reads a catalogue from stdin and corrects ellipticities
according to model for psf polarization p = e / psm as determined by
efit. Correction applied is e -= psm * p.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

eorrect2 - psf correction using a smoothed e/n image

SYNOPSIS

eorrect2 [options...] starcat

where options are

"-o x0 y0" origin in x-coord units (0 0)
"-d dbox" side of box in x-coord units (2048)
"-n N" size of smoothed images (128)
"-r rf" gaussian smoothing scale in pixels (10)

DESCRIPTION

'ecorrect2' is a perl script which invokes 'surfdens' to construct an image which is a smoothed image of p(r), the number weighted shear field (from the 'starcat') divided by the smoothed star number density times the mean 'psm'. It then invokes 'getfitsval' to import these values into the catalogue and subtracts from the e-vector a correction delta_e = psm * p(r). This procedure can accommodate more complex psf patterns than 'ecorrect' (which is limited to low-order polynomials) but watch for problems if smoothing scale is too small (it should encompass a reasonable number of objects) or if there are strong gradients of the psf over a smoothing scale. You should choose the size of the box to have a generous overlap around the bounding box of the catalogue objects.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

efit - generate model for stellar ellipticities

SYNOPSIS

efit [option...]

"-m emax " reject stars with |e| > emax (0.4)
"-x numax " don't use stars with |eres| > numax * sigma (3.0)
"-n N " image size (2048)
"-o order " order of Taylor series (1)
"-e ename " name for ellipticity (e)
"-i niter " number of times to iterate (2)

DESCRIPTION

"efit" reads a catalogue of stars from stdin and writes a set of coefficients for Taylor series expansion of the e / psm values to stdout. We first reject stars with |e| > emax, make the fit, and calculate the rms deviation from the fit. We then reject outliers with |eres| > numax * sigma and then we refit to get refined coefficients. Use '-i' option to iterate rejection-fitting cycle. Maximum order is 6.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3.20  imcattools_registration(1)

NAME

imcattools_registration - imcattools_registration tools
section

DESCRIPTION

The tools in this section are to help you register images taken with a 'single chip' CCD camera.

COMMANDS

"acfregister " determine approximate transformation coefficients
"fit2cats " fit for transformation/distortion for a pair
of cats
"fitdistortion " fit for field distortion
"fitstack " fit for transformation coefficients for a set
of exposures
"getstacktrans " calculate linear transformations and
"lintransfit " determine accurate transformation coefficients
"merge2cats " perl script to merge a pair of catalogues
"mergestack " perl script to merge a stack of catalogues
"registercats " perl script to do 2-step registration on
pair of cats
"registercats2 " another perl script to do registration
on pair of cats
"scalerottrans " scale rotate and translate spatial coords
in a cat
"warp" apply spatial transformation to a catalogue
"warpstack " perl script to merge a stack of images (or
catalogues)

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

acfregister - determine approximate transformation coefficients

SYNOPSIS

acfregister [options...] a.cat b.cat

"-x xname " name for spatial coords ('x')
"-i imsize " internal image size (512)
"-p phi1 phi2 " range for phi wrapping (0, PI)
"-a a1 a2 " die if solution has scale factor outside this range
"-d dlnr " range of (natural) log(r) for wrapping (1.0)
"-v " pipe images to 'iis -p /dev/saopipe' for display

DESCRIPTION

'acfregister' reads 2 catalogues and determines scale, rotation and translation that maps (x,a,y,a) => (x,b,y,b)

\[ x_b = a (x_a \cos \phi - y_a \sin \phi) + x_0 \]
\[ y_b = a (x_a \sin \phi + y_a \cos \phi) + y_0 \]

It works by cross-correlating images of wrapped lnr, phi values for pairs of objects and locating peak to determine a, phi. We then rotate and scale x_b y_b, generate images of x,y positions and cross correlate these to get x0, y0.

Outputs x0, y0, a, phi to stdout.

Use the -v option to get some visual feedback. You must first have saoimage running and listening to the FIFO /dev/saopipe so make this pipe if necessary (using mknod) and then run the image display using
'saoimage -idev /dev/saopipe'. Acfregister will then pipe the various images it generates internally to the viewer.

Acfregister will not work properly if the scale difference is very large ($|\log(a)| > dlnr/2$) as the wrapping will cause it to return $\log(a)$ in range $\pm dlnr/2$. You can increase dlnr, but at the cost of reduced precision in $\log(a)$. It is probably better to rescale one of the input cats to get roughly similar coordinate scales.

**AUTHOR**

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

fit2cats - fit for transformation/distortion for a pair of cats

SYNOPSIS

fit2cats

"-l lmax " max order for distortion polynomials (1)
"-o x0 y0 " origin for spatial coordinates (0,0)

DESCRIPTION

'fit2cats' reads from stdin the result from 'merge2cats' of merging a pair of cats, and which must contain entries for a pair of spatial coordinate vectors 'x[2][2]'. It then fits a model in which \( r = x[1] \) is related to \( (x,y) = x[0] r = x[0] \) + \( \sum_{l=0}^{l\text{max}} \sum_{m=0}^{l\text{max}} a_{lm} x^l y^m \) With the default \( l\text{max} = 1 \), this is just a straight linear transformation, but by going to higher order one can include distortions of the telescope optics. With -o option, these mode functions become polynomials in position relative to specified spatial origin. Transformation parameters \( a_{lm} \) are written to stdout in form to be read by warpimage or warpcat.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

fitdistortion - fit for field distortion

SYNOPSIS

fitdistortion ne [options...]

"-l lmax " max order for field distortion polynomials

"-d parfilebase " basename for transformation param files

"-o x0 y0 " spatial origin for mode functions

DESCRIPTION

'fitdistortion' reads from stdin a set of pairs of coords x[2][2] and exposure numbers e[2] (as produced by merging a set of catalogues pair by pair, and with 0 <= e < ne) and fits a model in which sky coord 'r' is related to chip coordinate 'x' by

\[ r = x + \sum_M a_M f_M(x) + \sum_N a_{eN} f_N(x) \]

where the f_M's are polynomials of order l = 2 to l_max and describe the field distortion and the f_N's are polynomials of order 0, 1 (i.e. linear transformations) which describe the telescope pointing as well as possible scale changes, or shear introduced by atmospheric refraction etc. With the -o option these become polynomials position relative to given spatial origin. We let exposure e = 0 define the orientation and scale of the image (so a_{0N} = 0).

The model is only approximate; really we should have:

\[ r = x + \sum_M a_M f_M(x) + \sum_N a_{eN} f_N(x + \sum_M a_M f_M(x)) \]

which is equivalent to the simpler form above in the limit of small mode coefficient amplitudes (i.e. small rotations between fields etc. By default, the transformation parameters a_{lm} are written to stdout as a concatenation of '.par' format files (which is not particularly useful), but with the -d option you can specify a basename (which might be a preexisting directory) and fitgeometry will create a set of files
'parfilebase' e'.par with e = 0... ne-1 containing the coefficients a_{eN} describing the pointings (a 'null' parameter file for the e = 0 exposure is provided for convenience) and parfilebase'dist.par containing the distortion parameters a_M.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
3.20.4 fitstack(1)

NAME

fitstack - fit for transformation coefficients for a set of exposures

SYNOPSIS

fitstack nexp [options....]

"-l lmin lmax " min and max order for distortion polynomials
"-i niter " number of iterations (1)
"-c outcat " output catalogue containing magc, r-values
"-o x0 y0 " origin for spatial coordinates (0, 0)
"-d distparfilename " file for distortion model parameters
"-t transparfilename " file for linear transformation params

DESCRIPTION

'fitstack' reads from stdin a catalogue containing the result of merging all pairs of cats for a stack of 'nexp' images (as created by 'mergestack') and which must contain entries for spatial coords 'x[2][2]', magnitude 'mag[2]' and exposure number 'exp[2]'. It then fits a model in which sky coords (in frame defined by exposure-0) are \( r = r_e + d\phi_e r_e + d_e \) where the 2x2 matrix \( d\phi \) allows for rotations between exposures and possibly atmospheric refraction, and we set \( d\phi = d = 0 \) for the 0th exposure. It will also optionally then fit for distortion of telescope using a model in which sky coords \( r \) are related to detector coords \( x \) by \( r_e = x_e + \sum a_m f_m(x_e) \) where \( m \) labels the modes, and where each mode coefficient \( a_m \) is a 2-vector and the modes are polynomials of order 'lmin' through 'lmax'. For \( lmin = 2, lmax = 4 \) say, the modes are: \( x^2, xy, y^2, x^3, x^2 y, x y^2, y^3, x^4, x^3 y, x^2 y^2, x y^3, y^4 \). With -o option, these become polynomials in position relative to specified spatial origin. We also read magnitudes, which we model as: \( m_e = m + M_e \) where \( m \) is the true magnitude and \( M_e \) is the magnitude offset the e'th exposure (relative to exp-0).
Use the '-c' option to generate a merged catalogue which contains, in addition to the source catalogue values, the sky coordinates 'r' and also corrected magnitudes 'magc', which can then be filtered to remove bad pairs and then fed back to 'fitstack' to improve the solution. See also fitstack.tex.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

getstacktrans - calculate linear transformations and magnitude offsets for a stack of catalogues

SYNOPSIS

getstacktrans [options...] cat_a, cat_b ....

-d d # merging length (1) -v # verbose mode -r ref_cat # reference catalogue for spatial coords

"-R residfile " output residuals to 'residfile'

"-m " don’t calculate magnitude offsets

"-o opts " options for registercats2

DESCRIPTION

By default 'getstacktrans' calculates the linear transformation coefficients between the first catalogue cat_a and the following catalogues cat_b, cat_c ... by first running 'registercats2'; applying transformation; merging pair of cats; un-applying the transformation and then feeding the resulting list of pairs of coords to 'lintransfit'. It also calculates the magnitude offset dm, and outputs a table containing the names of the input catalogues, transformation coefficients phi_ij, d_i and magnitude shifts dm for each of the cats listed (so the entries for the first cat are just phi_ij = delta_ij, d_i = 0, dm = 0) but if you supply another catalogue with the -r option the spatial transformations will be calculated with respect to this catalogue (but the magnitude shifts will still be calculated relative to 'cat_a'. You might use the r-option if you have already stacked the images for one passband and you wish to transform a subsequent set of images in another passband into the same frame, but with their own magnitude shifts. Also contained in the catalogue are a column containing the number of objects whose transformed positions match those in the reference catalogue and a column containing the names of the catalogues.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3.20.6 lintransfit(1)

NAME

lintransfit - determine accurate transformation coefficients

SYNOPSIS

lintransfit [options...]

DESCRIPTION

'lintransfit' reads a merged catalogue (created by 'mergecats' and containing a pair of vector coordinates in x[2][2] as x[0] = (x,y), x[1] = (x',y') from stdin and fits a linear transformation model: x' = x0 + Pxx x + Pxy y y' = y0 + Pyx x + Pyy y by least squares. Outputs dx, dy, Pxx, Pxy, Pyx, Pyy to stdout by default. Options are: -r Output catalogue containing x[2][2], dx[2] where dx = x' - x'_model.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

merge2cats - perl script to merge a pair of catalogues

SYNOPSIS

merge2cats cat0 cat1 [options...]

DESCRIPTION

Merge2cats makes a first attempt at merging a pair of cats which should each contain position vector 'x'. For each pair of cats we run 'registercats' to get approximate linear transformations which we apply to the 1st of each pair and then run 'mergecats' with specified linking length d. Options are:

"-d d " linking length (10)

"-i N " size of image used by acfregister (128)

"-x xname " name for spatial coordinate ('x')

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

mergestack - perl script to merge a stack of catalogues

SYNOPSIS

mergestack [options...]

DESCRIPTION

Mergestack makes a first attempt at merging a stack of cats which should each contain position vector 'x', and magnitude 'mag'. For each pair of cats we run 'registercats' to get approximate transformations which we apply to the 1st of each pair and then run 'mergecats' with specified linking length d. The results of all these merges is written to the file mergestack1.out by default. It reads a database file (default: exp.db) containing the names of the catalogues (without extensions) one per line. The result can be fed to 'fitstack' which will make a first solution for pointing and telescope distortion parameters. Options are:

"-d d " linking length (10)
"-c catdir " directory containing the mosaic cats (.images)
"-t " just calculate transformation parameters
"-s suffix " suffix for cats (’.cat’)
"-i N " size of image used by acfregister (128)
"-e expnames " specify alternative to ’exp.db’
"-o outfile " specify alternative to ’mergestack1.out’

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

register - perl script to do 2-step registration on pair of cats

SYNOPSIS

register a.cat b.cat [options....]

DESCRIPTION

first we run acfregister to get crude transformation coefficients then we approximately transform one list, merge the lists, transform back, and run lintransfit to get accurate coefts. Options are:

"-i imsize " imsize parameter for acfregister (256)

"-v " output acfregister images to iis

"-t tol " tolerance for mergecats (30)

"-V " verbose mode

"-x xname " name for spatial coordinate vector

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

registercats2 - another perl script to do registration on pair of cats

SYNOPSIS

registercats2 a.cat b.cat [options....]

"-d dmin " min separation to select isolated stars (100)
"-D dX " linking parameter for pairs in X-space (0.03)
"-N N0 " image size for X-space peak finding (256)
"-g DX " smoothing scale for X-space peak finding (0.001)
"-v " verbose mode

DESCRIPTION

registercats2 determines the scale a, rotation phi, and translation x0 y0 that maps (x_a,y_a) => (x_b,y_b): 

x_b = a (x_a cos phi - y_a sin phi) + x0
y_b = a (x_a sin phi + y_a cos phi) + y0

Like 'acfregister' it looks for a cross-correlation peak for pairs in log-separation -- orientation angle space but should be more efficient/accurate. However, it requires change in scale factor and rotation should be small. We first run 'mergecats' on each of the input cats (which should contain enough moderately bright stars to ensure good match) to excise stars with close neighbours (<100 pixels by default). We then run 'pair' on each of the isolated star cats, and for each pair calculate the vector 'X[0] = log |dx|, X[1] = phi' where dx is the pair separation and phi = atan2(dx[1], dx[2]) is the orientation of the pair. We then run mergecats to link pairs of pairs whose X-values match to within some tolerance (dX = 0.03 by default, but increase this if you anticipate e.g. relative rotations > this). We then generate a fits image of side N0 pixels (N0 = 256 by default) which is the density of points in the dX plane, and run 'findpeaks' with a gaussian smoothing scale DX (DX= 0.001 by default) and locate the highest peak. This provides us with the scale factor and rotation angle. We then select pairs which we apply to the spatial coord of the 1st cat => x_a' and then look for a peak in the x_a' - x_b in a similar manner... Outputs x0, y0, a, phi to stdout.
NAME

scalerottrans - scale rotate and translate spatial coords in a cat

SYNOPSIS

scalerottrans dx dy a phi [options....]

DESCRIPTION

applies transformation

\[ x_0 = a (x \cos \phi - y \sin \phi) + dx \]
\[ y_0 = a (x \sin \phi + y \cos \phi) + dy \]

where options are

"-x xname " name for the spatial coord vector

"-i " apply inverse transformation

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

warpcat - apply spatial transformation to a catalogue

SYNOPSIS

warpcat [options...] distparfile

-p psi_xx psi_xy psi_yx psi_yy -d d_x d_y

DESCRIPTION

"warpcat" reads a catalogue from stdin and applies a spatial transformation to position vector x[2] according to the parameters in 'distparfile'. It sends to stdout a catalogue with extra vector r = x + sum m a_m f_m(x) where mode functions are polynomials in x[0], x[1]. Use -p and -d options to apply a further linear transformation: r_i => phi_ij r_j + d_j

AUTHOR

Nick Kaiser: kaiser@ifa.hawaii.edu
NAME

warpstack - perl script to merge a stack of images (or catalogues)

SYNOPSIS

warpstack distparfile transparfile [options...]

DESCRIPTION

Warpstack reads a distortion parameter and linear transformation parameter files created by fitstack and applies appropriate warping to a set of images (or catalogues). Options are:

"-c datadir " directory for source and target files (images)

"-s srcsuffix " suffix for source files ('.sub')

"-d dstsuffix " suffix for target files ('.warp')

"-e expnames " exposure names file (exp.names)

"-C " warp catalogues

"-D " just print the commands for a 'dry run'

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3.21  imcattools_rshloop(1)

NAME

imcattools_rshloop - imcattools_rshloop tools section

DESCRIPTION

The tools in this section can be used to run imcat processes in parallel on a collection of nodes.

COMMANDS

"rshloop " execute shell commands in parallel.

"runcom " run a command for rshloop

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.21.1 rshloop(1)

NAME

rshloop - execute shell commands in parallel.

SYNOPSIS

ni commandformatstring

DESCRIPTION

'rshloop' executes shell commands in parallel across a network. It is currently functionally equivalent, and has identical arguments, to 'pvmloop'.

'rshloop' first reads a table of slaves from 'rshloopslaves.lst' (or from 'slavelist' with the -s option). The first line of this table should contain ns, the number of slaves to be used, followed by ns lines containing two strings: nodetag and nodename. For example:

3 01 node01 02 node02 03 node03

You may specify any node, including the local node, multiple times. This will result in multiple processes running on each physical node.

'rshloop' then creates a set of temporary FIFO pipes /tmp/pid.s.fifo, one per slave.

'rshloop' then initiates a sequence of nb batches of nc <= ns commands until all of the ni commands have been exhausted. If the node is the local node, the form of the command actually executed is

runcom command s | /tmp/pid.s.fifo &

where 'command' is generated from the 'commandformatstring' as described below and s is the the slave number.

The command 'runcom' executes the command given as its first argument and saves the stdout and stderr, if not already redirected elsewhere to temporary files. Once the command has terminated, it sends a message
to its stdout containing any stderr and stdout from the command and then cleans up after itself.

If the node is not the local node, the command actually executed is

rsh remotenode 'runcom command s' | /tmp/pid.s.fifo &

Once a batch of commands has been initiated, rshloop opens all of the FIFOs for reading (otherwise the commands will block). It then loops over the nc processes and collects the output.

The commandformatstring has a syntax similar to a printf format string. On each iteration it is processed and each occurrence of %i is replaced by the iteration number i = 0...ni-1, %I is replaced by a fixed length textual representation of i %n is replaced by the node number, and %N is replaced by the nodetag, and %% is replaced by %. The result may be a compound command of subprocesses linked by the pipe symbol ’|’, and the output of the command may be redirected into a disk file on the slave (in which case the master process will receive, and generate, no standard output).

EXAMPLES

These examples assume the rshloopslaves.lst file above, and that each slave has a disk named /dnn, where nn is the nodetag.

To check on the status of the slaves:  rshloop 3 w

To clear a scratch directory on each slave:  rshloop 3 'rm /d%N/tmp/*'

To generate a set of 1000 Monte Carlo simulations with some command 'monty' which takes as an argument a seed (given here by the iteration number): rshloop 1000 "monty -seed %i > /d%N/tmp/monty%I.dat" This would cause the following commands to be executed: monty -seed 0 > /d01/tmp/monty000.dat (on node01) monty -seed 1 > /d02/tmp/monty001.dat (on node02) monty -seed 2 > /d03/tmp/monty002.dat (on node03) monty -seed 3 > /d01/tmp/monty003.dat (on node01) .... monty -seed 998 > /d01/tmp/monty998.dat (on node03) monty -seed 999 > /d01/tmp/monty999.dat (on node01)

OPTIONS

With -u flag we output this man page and exit. With -d flag we just output the series of commands that would otherwise be executed and exit. This is highly recommended with commands that delete files etc. Use -v flag to invoke verbose mode, or -q to run quietly. Use -f ifmt to specify a format string for the iteration number, otherwise we use
'%.nd' specification where n is just large enough to hold the number ni-1. With '-i inticom' option the shell command string 'initcom' is executed on the master node using system(initcom) before any other output is collected. This allows you to generate a header, to which the output of the slave processes can be prepended. For example, rshloop -q -i 'lc -C -n x < /dev/null' 10 'makerandcat 1000 -seed 2 -dim 1 | lc -o' generates a catalogue containing 10 x 1000 random numbers. The effect of the 'lc -o' call is to chop the headers off the slave output. Similarly you can use 'imhead -g ....' to generate a fits header and pipe the output of the slaves through 'imhead -d'.

BUGS
'rshloop' does not die if the remote shell commands fail.
'rshloop' does not do any load balancing.

FILES
rshloopslaves.lst

SEE ALSO
pvmloop, pvmserver, topvm.

AUTHOR
Nick Kaiser --- kaiser@hawaii.edu
NAME

runcom - run a command for rshloop

SYNOPSIS

runcom comfile [-v]

DESCRIPTION

runcom reads a command from comfile and then executes it, capturing the stderr and stdout in temporary files /tmp/runcom.PID.stderr and /tmp/runcom.PID.stdout. It then sends those to stdout.

SEE ALSO

rshloop

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
NAME

imcattools_shearanal - imcattools_shearanal tools section

DESCRIPTION

The tools in this section are for measuring and analysing weak lensing shear.

COMMANDS

"apmass " calculate mkII style aperture mass profile
"etprofile " calculates tangential alignment profile
"makeshearimage " generate a shear image from catalogue
"maketestshear " make a test shear field
"massmap " direct summation massmap
"massmap_ft " ks93 reconstruction using FFT
"massmap_mp " maximum probability reconstruction
"sheartogradkappa " calculate gradkappa image from shear image

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.22.1 apmass(1)

NAME

apmass - calculate mkII style aperture mass profile

SYNOPSIS

apmass [options]

"-c xc yc " centre of coords (1024,1024)
"-n NX NY " box dimensions (2048, 2048)
"-e eps " softening parameter (0.01)
"-l lossfac " signal loss factor (1.0)

DESCRIPTION

Aperture mass statistic: reads cat file; outputs info analogous to etprofile but for rectangular aperture of linear size alpha times box size using mkII kernel. Remember the centre of coords is the point at which lines from corners of box through corners of aperture meet --- NOT the centroid of the aperture. Calculates the mean kappa in aperture rel to mean kappa in surrounding strip which depends only on shear estimates outside the aperture. The mean surface density nbar is also calculated using only galaxies outside aperture.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

etprofile - calculates tangential alignment profile

SYNOPSIS

etprofile [option...] ¡ catfile ® asciifile

"-o io jo " origin about which we do profile (2048, 2048)
"-d dlnr " log bin size 0.25
"-r rmin rmax " min and max radii (200, 2000)
"-l lossfactor " multiply e by 1/ lossfactor
"-e ename " name for 2-vector ellipticity (e)
"-x xname " name for 2-vector spatial coordinate (x)

DESCRIPTION

"etprofile" calculates tangential alignment profile from a catalogue. Also calculates kappaabar = 2 int d ln r eT with geometrical boost factor 1 / ( 1 - r^-2 / rmax^-2). Error bars calculated using orthogonal shear component.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

makeshearimage - generate a shear image from catalogue

SYNOPSIS

makeshearimage [options...]

"-N N1 N2 " size of image (128,128)
"-x xname " name for spatial coordinate ('x')
"-X dx " pixel size in x-coords (16.0)
"-o x0 y0 " origin in x-space (0.0, 0.0)
"-r x1 x2 y1 y2 " box for calculating nbar
"-e ename " name for polarisation ('e')
"-p psh " polarisability (1.0)
"-n nbar " density of points in x-space

DESCRIPTION

'makeshearimage' reads a catalogue which must contain a spatial 2-vector coordinate ('x' by default) and a 2-vector polarisation ('e' by default), and creates a N1 by (2 N2) fits image which is a simple binned average of the input shear values. The shear is defined as $\gamma_i = \epsilon_i / psh$, and $\gamma_0$ and $\gamma_1$ are stored in the first and last N2 lines of the output image respectively. By default the density of points is estimated as the number of objects divided by the area of the image (in x-coord units) but you can supply an alternative x-rectangle with the -r option.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
maketestshear(1)

NAME

maketestshear - make a test shear field

SYNOPSIS

maketestshear [options...]

"-n N " image size (256)
"-o i0 j0 " lens position (128,128)
"-c rc " core radius (50)
"-k " generate kappa
"-g " generate gradkappa

DESCRIPTION

'maketestshear' generates a float format fits image of shear for kappa
= exp(-r^2 / 2 r_c^2) model

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

massmap - direct summation massmap

SYNOPSIS

massmap [option...]

"-g ng " output grid size (64)
"-s rs " gaussian smoothing radius in output grid size units (2.0)
"-n N " input image size (2048)
"-o xo yo " origin of cat in pixels (0,0)
"-R R " radius for determining n_bar (N/4)
"-e pol " output smoothed (n * e[pol] / nbar) map
"-d " output D.s
"-c " output D x s
"-l lossfactor " divide final mass-map by lossfactor (1.0)
"-E ename " name for 2-vector ellipticity (e)
"-x xname " name for 2-vector spatial coordinate (x)

DESCRIPTION

"massmap" reads e[2], x[2] from a catalogue and calculates foreground surface mass density field from background galaxy ellipticities a la KS. Calculates mean galaxy number density in disk radius R around field centre Outputs Sigma / Sigma_crit unless -e option set in which case it outputs a gaussian smoothed map of (n * e[pol] / nbar) for pol = 0 or 1 Use lossfactor (< 1.0) to correct for seeing Uses W(theta) = (0.1 y^4 / (1 + 0.1 y^4)) / theta^2 (where y = theta / sigma) as truly excellent approximation to bessel function window for gaussian T(k)

AUTHOR

226
NAME

massmap_ft - ks93 reconstruction using FFT

SYNOPSIS

massmap_ft [options...]

"-m mode " mode (0)

"-r R " pad by factor R (2)

DESCRIPTION

'massmap_ft' reads a shear image in the format produced by 'makeshearimage' from stdin and reconstructs density field by fourier-space version of KS93. The input data are zero-padded onto an internal image which is by default twice the size of the original. You can use a slightly different estimator with -m option: mode = 0 for KS93 chi estimator mode = 1 for zeta estimator.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

massmap_mp - maximum probability reconstruction

SYNOPSIS

massmap_mp [option...] <input.cat> <output.image>

"-g ng " output grid size (64)
"-n N " input image size (2048)
"-k kmax " maximum k-value (5)
"-a alpha " power parameter (0.01)
"-e ename " name for 2-vector ellipticity (e)
"-x xname " name for 2-vector spatial coordinate (x)

DESCRIPTION

"massmap_mp" reads x[2], e[2] from a catalogue and calculates foreground surface mass density field from background galaxy ellipticities a la KS. Uses maximum probability technique diagonal stripe controlled by alpha. Use alpha << 1 for weak regularization.

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

sheartogradkappa - calculate gradkappa image from shear image

SYNOPSIS

sheartogradkappa

DESCRIPTION

'sheartogradkappa' reads shear from a fits file on stdin and calculates grad - kappa which is sent to stdout

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
3.23  imcattools_simulation(1)

NAME

imcattools_simulation - imcattools_simulation tools section

DESCRIPTION

The tools in this section are generate mock catalogues, images.

COMMANDS

"circimfromcat " generate a circularly symmetric image
"makecosmocat " generate catalogue of mock galaxies
"makemockimage " generate mock deep CCD image

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

circimfromcat - generate a circularly symmetric image

SYNOPSIS

circimfromcat [options]

DESCRIPTION

'circimfromcat' reads an lc-format catalogue from standard input containing at least a single scalar (default name F) which is interpreted as a set of uniformly spaced samples of a function F(r), and generates a square circularly symmetric 2-D image \( f(x,y) = F(r = \sqrt{x^2 + y^2}) \). Options are:

"-d dr " spacing of input samples in pixels (1.0)

"-n N " size of output image (512)

"-N nrmax " input array size (10000)

"-F Fname " name for the input value

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

makecosmocat - generate catalogue of mock galaxies

SYNOPSIS

makecosmocat [-Magmin Magmin] [-Magmax Magmax] [-nM nM]


DESCRIPTION

Makecosmicat generates a lc-format catalogue containing mock galaxies. Default parameters are:

"Magmin " min abs magnitude (-23)
"Magmax " max abs magnitude (-10)
"nM " number of bins in magnitude (2048)
"Mstar " LF knee (-20)
"phistar " LF normalisation (0.03)
"alpha " LF faint end slope (1)
"zmin " min redshift (0.03)
"zmax " max redshift (10)
"zstar " limiting z for L* gal (3)
"omega " solid angle (5e-06)
"seed " random number seed (1)
OUTPUT

Output variables are:

"r " comoving distance [Mpc]
"z " redshift
"M " absolute magnitude
"m " apparent magnitude
"theta " angular size
"csb " central surface brightness

where theta and csb are normalised to value for L_* galaxy at z = z_* and apparent magnitude scale is such that L_* gal at z_* has m = 25.

AUTHOR

Nick Kaiser --- kaiser@ifa.hawaii.edu
NAME

makemockimage - generate mock deep CCD image

SYNOPSIS

makemockimage [option...]

"-n N1 N2 " size of image (1024, 1024)
"-x xname " get position from 2 vector input variable 'xname'
"-phi phiname " get position angle from input variable 'phiname'
"-mu muname " get position angle from input variable 'muname'
"-seed seed " seed for random numbers (1)

DESCRIPTION

"makemockimage" reads a catalogue of containing size 'theta' and central surface brightness 'csb' (perhaps generated by 'makecosmocat') information and generates a fits image containing exponential disks of random orientation.

By default, makemockimage will generate uniform random position x, position angle phi and orientation mu (the latter being the cosine of the angle between the disk polar axis and the line of sight) but you can use -x, -phi and -mu options to read these from the input catalogue instead.

For dust free disk galaxies one would expect the surface brightness to vary with orientation as 1 / mu. However, images made in this way look quite unrealistic, with way too many bright edge on things, so in the default mode we take the 'optically thick' approximation and don't scale the csb.

AUTHOR
3.24  imcattools_tesselation(1)

NAME

    imcattools_tesselation - imcattools_tesselation tools section

DESCRIPTION

The tools in this section can be used to perform Delauney tesselation to interpolate unevenly sampled data onto an image for example.

Uses Jonathan Richard Shewchuk’s triangle.[ch]

COMMANDS

"triangulatecat " Delauney tesselation of a catalogue

AUTHOR

    Nick Kaiser -- kaiser@hawaii.edu
3.24.1 triangulatecat(1)

NAME

triangulatecat - Delauney tesselation of a catalogue

SYNOPSIS

triangulatecat srcat [-c | -f x1 x2 Nx y1 y2 Ny] [-m | -M]

DESCRIPTION

Triangulatecat performs Delauney tesselation using Shewchuk’s code. Input catalog should contain a 2-vector x[2] and scalar or 1-D vector f[N].

Default is to output the triangles x[3][2], f[3][N]. With -c option we output an average cat x[2] = xbar[2], f[N] = fbar[N]. Ditto with -c -m options. With -c -M options we output a median cat x[2] = xbar[2], f[N] = fmedian[N]. With -f ... option we output a Nx by Ny FITS image consisting of triangular linear ramp segments, or, with -m or -M options the triangles are uniform and painted with the mean or median f[] value respectively.

Uses iostream utilities so use '-' for standard input, 'somecommand |' to read from a process etc.

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.25  imcattools_waves(1)

NAME

imcattools_waves - imcattools_waves tools section

DESCRIPTION

The tools in this section allow one to numerically evolve dispersive waves (including the free scalar field) and also to evolve scalar fields (including self interactions etc).

COMMANDS

"edw " evolve dispersive waves
"evolvescalar " evolve a 2-D scalar field
"generate_dw " generate waves for edw

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
3.25.1 edw(1)

NAME

edw - evolve dispersive waves

SYNOPSIS

edw dt [-ocean | -debroglie] [-verbose]

DESCRIPTION

Edw reads a 3D FITS image f[2][Ny][Nx] from standard input, the two planes of which are the initial field f = f[0] and the initial velocity fdot = f[1].

It computes the transforms fk of f and fdotk of fdot and forms the positive frequency component: fp(k) = (fk + fdotk / i omega) It then computes and sends to stdout a stream of images containing the the inverse transform of fp(k) * exp(i omega(k) * t). for t = integer multiples of dt.

By default, the dispersion relation is omega = k. With option -ocean we use omega = sqrt(k).

With -verbose option we tell stderr what we are doing.

SEE ALSO

generate_dw, evolvescalarfield, xfv

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
evolvescalar(1)

NAME

evolvescalar - evolve a 2-D scalar field

SYNOPSIS

evolvescalar [-nsteps nsteps (10)] [-nframes nframes (99999)]

[-kstar kstar (0.3)] [-autoscale] [-dt dt (0.2)] [-lambda lambda (0.0)]
[-gamma gamma (0.0)] [-u]

DESCRIPTION

Evolvescalar reads from stdin a 3-D FITS file \( f[2][Ny][Nx] \) consisting
of 2 planes containing the initial field \( d[Ny][Nx] \) and the initial
field velocity \( v[Ny][Nx] \). It then evolves the coupled equations
\[
\frac{dv}{dt} = \text{laplacian}(d) - kstar^2 d - 4 \lambda d^3 - \gamma v \\
\frac{dd}{dt} = v
\]
With \( \gamma = 0 \) these are equivalent to the Klein-Gordon equation with mass
kstar and a \( \lambda \phi^4 \) self-interaction.

If kstar is negative, it evolves\( \frac{dv}{dt} = \text{laplacian}(d) + kstar^2 d \\
- 4 \lambda d^3 - \gamma v \frac{dd}{dt} = v \)
which has a negative mass term and gives a 'w'-shaped potential.

The laplacian function is computed as
\[
\text{laplacian} = d[y][x-1] + d[y][x+1] + d[y-1][x] + d[y+1][x] - 4 * d[y][x].
\]

The evolution scheme is
\[
v += 0.5 * \frac{dv}{dt} * dt \\
d += \frac{dd}{dt} * dt \\
v += 0.5 * \frac{dv}{dt} * dt
\]
It outputs a 3-D FITS file \( f[nframes][Ny][Nx] \) containing the evolved
field.

OPTIONS

"-nsteps nsteps " number of steps between output frames
"-nframes nframes " total number of output frames

"-kstar kstar " Compton wave-number

"-autoscale " scale each output frame to 0-256

"-dt dt " time-step

"-lambda lambda " interaction strength

"-gamma gamma " damping rate

"-u " print man-page

SEE ALSO
edw, generate_dw

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
3.25.3  generate_dw(1)

NAME

generate Dw - generate waves for edw

SYNOPSIS

generate Dw N [-gaussian alpha rf] [-packet theta sigma k] [-droplet sigma] [-dr ocean | debroglie] [-u]

DESCRIPTION

Generate Dw generates N x N initial data for a wave field suitable for evolution with edw.

By default it generates Gaussian noise with flat spectrum (alpha = 0) and 1-pixel smoothing length; i.e. equivalent to 'generate Dw N -gaussian 0.0 1.0'.

With '-packet theta sigma k' it generates a gaussian profile wave packet with scale length sigma and wave vector k propagating in direction theta (in degrees). Use the -dr flag to set the dispersion relation.

SEE ALSO

edw, xfv

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
4 imtools(1)

NAME

imtools - imtools tools section

DESCRIPTION

These tools deal only with FITS format images. See <a href="../byteorder.html">Byte-Ordering, FITS and Imcat</a> regarding imcat’s handling of FITS images.

COMMANDS

"acf " calculates 2-D autocorrelation function of a fits image

"album " paste fits images into a larger image

"asciitofits " converts an ascii image to fits format

"boxavg " boxcar smooth FITS image of arbitrary dimensionality

"bscale " set or check BSCALE, BZERO header values in FITS file

"changesex " switch byte order of a FITS file

"colorbox " generate example color image

"colorize " convert gray fits file to rgb colormapped version

"combineimages " create mean, median, avsigclip of a stack of images

"combinestamps " take mean, median etc of a stack of images

"complex " perform operations on complex images

"convolve " convolve one fits file with another

"cycleimage " translate a FITS image

"fft " take the fast fourier transform of a fits image

"fillcores " patch up holes in stellar cores
"fitstoascii " convert a fits image to ascii format
"fitstocat " make lc format list of fits image values
"fitstomovie " convert a 3 dimensional FITS file to a movie file
"flatten " convert 3-D fits to 2D
"flip " reflect or rotate a fits image
"getplane " extracts an image plane from multi-dimensional fits image
"getplanes " extracts a set of planes from multi-dimensional fits image
"grad " take the gradient of a fits image
"growsmagic " expand MAGIC regions of an image
"helicalsacan " generate a helical scan of a 2D FITS image
"ic_stream " apply simple arithmetic to a stream of images
"imhead " extract fits header
"invgrad " take the inverse gradient of a FITS image
"kappa2stuff " generate shear etc from surface density
"lensmap " map a source image to a target image
"linesmode " calculate mode of rows/cols of a fits file
"logscaleimage " scale image for printing
"make_image " generates a mock CCD image
"makecolourimage " combines 2 or 3 images into a color fits image
"makesubimage " extracts a subimage from a fits file
"mapbynumericdef " map image using numerical deflection func
"mask " set abnormal pixels in sky-flat to MAGIC
"maskfits " set pixels in rectangles to MAGIC
"overscancorr " dark subtraction using overscan region
"pixdist " compute histogram of pixel values in a FITS image
"projectfits " average over rows, cols etc of a FITS file
"scrunch " demagnify image by factor 2
"simulate " generate mock fits image
"sliceimage " cut a 2-D FITS image into a grid of sub-sections
"smooth " spatially filter a fits image
"spinflip " apply 90 degree rotations or parity flips to FITS image
"stackplanes " concatenate a set of fits files to multidimensional image
"stats " calculate simple statistics for an image
"statstocat " convert stats output to lc cat format
"transformimage " apply spatial linear transformation to a FITS image
"translate_fft " translate an image using 'sinc-interpolation'
"transposebits " swap order of bits and numbers in a FITS file
"unpackextensions " extract extensions from multi-part FITS file
"unscrunch " expand a FITS image by a factor 2
"warpimage " apply spatial transformation to a FITS image

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
4.1 acf(1)

NAME

acf - calculates 2-D autocorrelation function of a fits image

SYNOPSIS

acf [options...]

DESCRIPTION

'acf' reads a fits file from stdin and writes the autocorrelation function to stdout. If the input image is \( \text{fin}(r) \) then we compute \( \text{fout}(r) = \sum_r \text{fin}(r') \text{fin}(r' + r) / (N1 * N2) \) and resulting image is wrapped so that zero lag is at pixel \((N1/2, N2/2)\).

Options are:

"-p " calculate power spectrum instead

"-c f1 f2 " cross correlate named fits files

"-P pixtype " output pixtype (FLOAT_PIXTYPE)

"-n " no MAGIC substitution

"-m magicval " replacement for MAGIC pixels (0)

Power is defined so that white noise with variance \( \sigma_2 \) will produce \( P = \sigma_2 \). Power is translated so that zero frequency lies at \( N1/2, N2/2 \).

With ‘-c fits0 fits1’ we compute

\[
c(r) = \sum_r' f0(r') f1(r' + r) / (N1 * N2)
\]

to compute instead

\[
c(r) = \sum_r' f0(r') f1(r - r') / (N1 * N2)
\]
you should first rotate the first image by 180 degrees.

Supply file name '-' to read from stdin.

Default output image format is BITPIX = -32 (i.e. 4-byte float).

AUTHOR

    Nick Kaiser: kaiser@cita.utoronto.ca
4.2 album(1)

NAME

album - paste fits images into a larger image

SYNOPSIS

album [options... ] nx ny fits1 fits2 ...

album -f offsetsfile [options... ] N1 N2 fits1 fits2 ...

"-d dx dy " grid spacing

"-b " draw a line around each chip

"-f file " file for offsets

"-M " initialise to MAGIC.

DESCRIPTION

"album" combines a set of images into an album. With -f option the layout of the image is determined from the'offsetsfile', and the size of the output image is specified after the options. The format of this file should be a single comment line followed by x,y pairs to define location of bottom left corners of the images to be pasted. Without the -f option, the grid spacing will be equal to the dimensions of the first image (unless you override this with -d option) and the images will be placed on a nx by ny grid.

The output image is initialised to zero, images are painted on sequentially, erasing any previously painted values, except that MAGIC values are not painted. Use the -M option to initialise to MAGIC instead.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.3 asciitofits(1)

NAME

asciitofits - converts an ascii image to fits format

SYNOPSIS

asciitofits [option...]

DESCRIPTION

'asciitofits' reads an ascii format image from stdin and writes a fits image to stdout

"-f " create float format fits file

"-i " create 32 bit int format fits file

"-p " data in 2-hex digits a la postscript

ascii format is # N2 N1 then N2 x N1 pixvals

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.4  boxavg(1)

NAME

boxavg - boxcar smooth FITS image of arbitrary dimensionality

SYNOPSIS

boxavg [-u] [-m] [-d d0 d1...]

DESCRIPTION

"boxavg" reads a FITS image of arbitrary dimensionality from stdin and applies a simple boxcar smoothing. By default the width of the boxcar is 3 in each dimension, but you can specify an array d[] with the -d option such that the width in the i’th direction is 1 + 2 * d[i].

Options:

"-u " print this message

"-d d0 d1... " specify (width-1)/2 of box car (1,1,...)

"-m " take median

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
4.5 bscale(1)

NAME

bscale - set or check BSCALE, BZERO header values in FITS file

SYNOPSIS

bscale [fmin fmax | -g | -u ]

DESCRIPTION

"bscale fmin fmax" reads an image (which must be an integer format
BITPIX = 8, 16 or 32) from stdin and modifies or generates BZERO, BSCALE
values such that the max and min representable values are fmin, fmax.
This has the effect of rescaling the image values.

Use "bscale -g" to get the current fmin, fmax values.

Use -u option to print this message.

Beware that the minimum representable values for BITPIX 16, 32 and
the maximum representable values for BITPIX 8 are 'magic' values used
by imcat to flag bad or missing data.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

changesex - switch byte order of a FITS file

SYNOPSIS

changesex [-u] [-n]

DESCRIPTION

The filter "changesex" swaps the byte order in a fits image.

Imcat commands can work with either big- or little-endian images. Since 2/99 imcat written images contain a header value with keyword BYTEORDR whose value can be either BIG_ENDIAN or LITTLE_ENDIAN, and the value of which is set by the presence or absence of the environment variable IMCATSWAPFITBYTES. If IMCATSWAPFITBYTES is set then the image is stored in non-native byte order.

Non-imcat images are assumed to be BIG_ENDIAN, while pre 2/99 imcat images are assumed to be in the native byte order. If you have set IMCATSWAPFITBYTES then you will need to filter old style images through 'changesex' to bring them into non-native order.

Use 'changesex -n' to force images with BYTEORDR=LITTLE_ENDIAN to BIG_ENDIAN format for compatibility with non-imcat software.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.7  colorbox(1)

NAME

colorbox - generate example color image

SYNOPSIS

colourbox N

DESCRIPTION

"colourbox" generates a N x N 2-colour image which spans the spectrum

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
NAME

colorize - convert gray fits file to rgb colormapped version

SYNOPSIS

colorize colormapindex [options...]

DESCRIPTION

If "colorize" reads a 2-dimensional fits file f[N2][N1] from standard input and writes a 3-dimensional rgb version f[3][N2][N1] with f[0][][] = r[], f[1][][] = g[], f[2][][] = b[].

If it reads a 3 dimensional image f[N3][N2][N1] it generates a four dimensional version f[N3][3][N2][N1] -- i.e. a stream of 3-D rgb images.

The rgb colors are computed by linear interpolation on a color ramp.

Options are:

"-f fmin fmax " limits for input image (0, 255)

"-a " use autoscaling

"-p bitpix " output pixtype (8)

There are currently three colormaps available (0,1,2).

BUGS

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
4.9 combineimages(1)

NAME

combineimages - create mean, median, avsigclip of a stack of images

SYNOPSIS

combineimages [options...] infile1 infile2....

"-a clip " reject pixels with |f - f_median| > clip * sigma
"-r r " output the r'th image
"-s " get sigmas from fits headers
"-e " get sigmas from exposure map images
"-f f1 f2 " allow all pixels with f1 < (f - f_median) / |fmedian| < f2
"-v " verbose mode.
"-o name " output average (.fits) and exposure (.exp) images
"-O name " output average (.fits), exposure (.exp) and bad planes (.bp) images
"-F " force 4 byte float format output image
"-I bscale bzero " force 2-byte output with b-scaling
"-m bscale " bscale factor for sigma image (0.0001)

DESCRIPTION

"combineimages" reads a collection of fits images (which must have identical sizes) and writes an average image to stdout. Default behaviour is to calculate the median.

Use -a option for various types of averaging. With -a option we do avsigclip style rejection, rejecting pixels with f - f_median > clip
* sigma, and take the weighted average of what’s left. With negative clip we take straight mean.

With -s option we read sigma’s from ’SIGMA’ value in fits header. With -e option we get sigma’s from exposure map images containing 1/sigma^2. These images must have the same names as the source images, but with suffix ’.exp’.

With (-s or -e) and -a options we reject pixels which differ from the median by more than clip * sigma and also lie outside the range f1 < (f - fmedian) / |fmedian| < f2. The extra tolerance for large |fmedian| values is to allow for variations in the shapes of stars due to seeing variations. Sensible values for a range of 3:5 in seeing (eg 0.6' - 1.0'') are f1 = -0.5, f2 = 1.0.

With (-s or -e) but without the -a option we take the weighted mean with no rejection.

If sigma values are supplied (with -s or -e options) then the pixels which survive selection are averaged with weight proportional to 1 / sigma^2.

With ’-o foo’ option (and -s or -e) we output two images:

"foo.fits " the average image
"foo.exp " the exposure map = sum 1/sigma^2

With ’-O foo’ option (and -s or -e) we output three images:

"foo.fits " the average image
"foo.exp " the exposure map = sum 1/sigma^2
"foo.bp " bad-planes image: n’th bit set if n’th image rejected

With -r option we output the pixel value of a certain rank. With rank = n and n > 0 we output the n’th lowest image, unless there are fewer than n non-MAGIC values in which case we output the highest pixel. With rank = -n and n > 0 we output the n’th highest image, unless there are fewer than n non-MAGIC values in which case we output the lowest pixel.

With -v option we print the SIGMA values to stderr. Use -F option to generate float format image -- otherwise we inherit the value from the first input image.

AUTHOR

257
Nick Kaiser: kaiser@cita.utoronto.ca
4.10 combinestamps(1)

NAME

combinestamps - take mean, median etc of a stack of images

SYNOPSIS

  combinestamps [options...]

DESCRIPTION

'combinestamps' reads a set of Nim images in the form of a Nim x N2 x N1 fits image (as created by 'makestamps' for instance) and performs some kind of average. By default it takes the median. Options are:

  "-m " take mean instead

  "-a clip " take average clipping at +- clip * sigma

Input images must have identical sizes. Combinestamps is functionally very similar to 'combineimages', but is unsuitable for large images as it reads all of the images into memory. However, it is therefore capable for reading very large *numbers* of images unlike 'combinimages' which reads images line by line and must therefore have a file open for each input image.

AUTHOR

  Nick Kaiser: kaiser@hawaii.edu
NAME

complex - perform operations on complex images

SYNOPSIS

complex op [files...]

DESCRIPTION

Complex works with 2-D complex images stored in f[2][n2][n1] FITS format.
Examples

complex multiply f1 f2
complex conjugate f
complex ab2rphi f
complex rphi2ab f

where f, f1, f2 are filenames or ’-‘ for standard input.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

convolve - convolve one fits file with another

SYNOPSIS

convolve [options...] target.fits psf.fits

"-f " psf.fits contains fft
"-n " normalise psf
"-m mval " substitute value for magic target pixels (0.0)
"-d " don't fix magic pixels
"-o " psf origin at 0,0.

DESCRIPTION

'Convolve' convolves target file 'target.fits' with a point spread function 'psf.fits'. Unlike 'acf -c', the two input files need not have the same dimensions (and will typically be used with a relatively small psf image) but should have the same pixel scale. If the two images have the same size then 'acf -c' is more efficient. By default 'convolve' expects the psf to be a M1 * M2 real-space image with spatial origin at M1 / 2, M2 / 2; use '-o' option if the origin is 0, 0. Use -f option to interpret 'psf.fits' as a fourier transform of the psf (in format generated by 'fft'). The output image has the same dimensions and type as the input image. WARNING - THIS DOES NOT REALLY WORK PROPERLY - TO BE FIXED.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.13 cycleimage(1)

NAME

    cycleimage - translate a FITS image

SYNOPSIS

    cycleimage dx dy

DESCRIPTION

"cycleimage" cycles an image moving each point a distance dx, dy (in pixels).

AUTHOR

    Nick Kaiser: kaiser@cita.utoronto.ca
4.14  fft(1)

NAME

    fft - take the fast fourier transform of a fits image

SYNOPSIS

    fft [options...]

DESCRIPTION

'fft' creates a three dimensional image with dimensions N1 = Nx, N2 = Ny, N3 = 2 whose zeroth and first planes contain the real/imaginary parts of the discrete fft of the input image. Zero spatial frequency lives at Nx/2, Ny/2. Options are:

"-s " output 16 bit image
"-i " output 32 bit int format image
"-n " no MAGIC substitution
"-m magicval " replacement for MAGIC pixels (0)
"-c " don’t cycle input image.
"-I " perform inverse transform.
"-C " complex fft

Default output image format is BITPIX = -32 (i.e. 4-byte real).

With the numerical-recipes fft package this only works for image dimensions 2^N.

This may be useful for ‘psf-surgery’: You could 'fft' a composite psf image; modify the fft e.g. by dividing it into some desired circular psf, and then apply the fft as a filter to your original image using 'smooth -F' and --- hey presto! --- your psf is now circular.

By default, and when doing a forward transform the input image will be cycled by N1/2, N2/2, but you can override this with -c option.
There is redundancy in the fft image created from real input since 
\( f(-k) = f^\ast(-k) \).

Use -C option to take transform of a complex function.

AUTHOR

Nick Kaiser:  kaiser@cita.utoronto.ca
4.15 fillcores(1)

NAME

fillcores - patch up holes in stellar cores

SYNOPSIS

fillcores threshold replaceval [corevalue]

DESCRIPTION

fillcores is a crude kludge to patch up holes in stellar cores as sometimes get produced by clumsy image arithmetic. By default, any pixel with MAGIC value which has a neighbour with value exceeding 'threshold' will be replaced by 'replaceval'. If the optional third argument is given then any pixel with value 'corevalue' will be replaced by 'replaceval'.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

fitstoascii - convert a fits image to ascii format

SYNOPSIS

fitstoascii [option...]

"-f fmt " use this format string (%13.8g)
"-l " 1 col list output format
"-t " x,y,pixval triplets
"-g " generate gnuplot format list

DESCRIPTION

cconverts a fits image to ascii format format must be valid for float (i.e. f,e or g)

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.17 fitstocat(1)

NAME

fitstocat - make lc format list of fits image values

SYNOPSIS

fitstocat [-u] [-b]

DESCRIPTION

'fitstocat' reads a FITS image from stdin and sends a list of pixel values to stdout as a lc format catalogue containing pixel indices x[] and values f.

Options:

"-u " print this message

"-b " generate a binary format catalogue

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
4.18 fitstomovie(1)

NAME

    fitstomovie - convert a 3 dimensional FITS file to a movie file

SYNOPSIS

    fitstomovie [-c cmap] [-z zoomfac] [-g | -G dstgif | -M dstmpg] [-x]

DESCRIPTION

'fitstomovie' reads a 3 dimensional FITS image from stdin and generates a set of individual frame images in tmp/nnnnn.sfx using fitstopnm with nnnnn = 00000, 00001, 00002....

By default sfx = ppm or pgm depending on color/grayscale image type. With -g option we output a set of sfx = gif files and with -G dstgif we combine them with multigif to generate the multiframe output file dstgif and then deleting the temporary frames. Similarly, with -M option we generate a set of .ppm files and then combine them with 'mpeg_encode tmp/mpeg.param' where tmp/mpeg.param is generated by fitstomovie and then deleted. The -M and -G options require that you have installed mpeg_encode and/or multigif as appropriate.

With -x option we don’t clear up any temporary files we create.

Images should be scaled to min = 0, max = 255.

fitstomovie will first 'rm -f tmp/?????.sfx' so use with care.

Options are:

    "-c cmap " pipe fits output through colorize cmap -f 0 255

    "-z zoomfac " paint pixels of size 2^ zoomfac (0)

    "-u " print this message
AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

flatten - convert 3-D fits to 2-D

SYNOPSIS

flatten [options...]

DESCRIPTION

"flatten" reads a 3-dimensional fits file from standard input and writes a 2-dimensional version to stdout. By default the output image is simply the concatenation of the input image planes.

Options are:

"-c ncols " write images in rows ncols wide (1)

BUGS

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
4.20  flip(1)

NAME

    flip - reflect or rotate a fits image

SYNOPSIS

    flip [h|v]

DESCRIPTION

By default "flip" rotates an image by 180 degrees to produce an image of the same dimensions. With optional first argument 'h' or 'v' it will reflect the image about the horizontal or vertical axis respectively.

AUTHOR

    Nick Kaiser:  kaiser@hawaii.edu
NAME

getplane - extracts an image plane from multi-dimensional fits image

SYNOPSIS

getplane p

DESCRIPTION

"getplane" reads from stdin a N-dimensional fits image f_in[Nn][Nn-1]....[N2][N1] and writes to stdout a N-1 dimensional image f_out[Nn-1]....[N2][N1] with values f_out[xn-1]...[x2][x1] = f_in[p][xn-1]...[x2][x1]

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

getplanes - extracts a set of planes from multi-dimensional fits image

SYNOPSIS

getplanes i1 i2...

DESCRIPTION

"getplanes" reads from stdin a N-dimensional fits image f_in[Nm]....[N2][N1] and writes to stdout a N dimensional image with planes f_out[i1]....[N2][N1] f_out[i2]....[N2][N1] etc.

AUTHOR

Nick Kaiser:  kaiser@hawaii.edu
4.23  grad(1)

NAME

ggrad - take the gradient of a fits image

SYNOPSIS

grad [-p]

DESCRIPTION

'grad' reads a N1 by N2 2-D fits file f[][] from stdin and calculates
discrete second derivatives. The output is sent to stdout in the form
of a N1 by N2 by 2 image where the zeroth plane is df[0][y][x] = f[y][x+1]
- f[y][x] and the second plane is df[1][y][x] = f[y+1][x] - f[y][x]
By default the rightmost column of the first half and the last row
are zero. Use -p option to use periodic bc's.

AUTHOR

Nick Kaiser:  kaiser@cita.utoronto.ca
4.24  growmagic(1)

NAME

growmagic - expand MAGIC regions of an image

SYNOPSIS

growmagic [-n n]

DESCRIPTION

"growmagic" reads a source image from stdin. It makes a copy and then sets to MAGIC any pixel for which any of its eight neighbours have the MAGIC value.

With '-n n' option it performs this operation n times.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

helicalscan - generate a helical scan of a 2D FITS image

SYNOPSIS

helicalscan M d phi [-N nframes] [-o xo yo]

DESCRIPTION

'helicalscan' reads a 2 dimensional FITS image fin[y][x] from stdin and writes to stdout a 3 dimensional FITS image fout[i][y][x] consisting of a set of M x M subimages with origin placed at a sequence of positions on the input image:

x0 = (xo + i * d * cos(\phi)) \mod N y0 = (yo + i * d * sin(\phi)) \mod N

The angle \phi at which the subimage moves across the source image is given in degrees.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
4.26   ic_stream(1)

NAME

   ic_stream - apply simple arithmetic to a stream of images

SYNOPSIS

   ic_stream [plus | minus | times | divide] operandfits

DESCRIPTION

'ic_stream plus myfits' reads a 2D image myfits and then reads planes from a higher dimensional image, adds the value from myfits and outputs the result.

AUTHOR

   Nick Kaiser:  kaiser@hawaii.edu
4.27 imhead(1)

NAME

imhead - extract fits header

SYNOPSIS

imhead [options...]

"-b " output full binary header
"-D " discard header
"-d " discard header and tail
"-v name " print value of header item 'name'
"-t name " print value of header item 'name'
"-g pixtype NAXIS NAXIS1 ... " generate fits header.

DESCRIPTION

"imhead" prints fits image header info reads/writes from/to stdin/stdout. By default it formats the header values, but with -b option it will output full header with padding exactly as it appears - this is useful if you have some data in raw bytes format and want to construct a legal fits header: use 'ic' to generate a suitable image and then pipe this into 'imhead -b' to make the header. Use -D, -d options to extract raw image data. With -v we print the value for the last header item whose name matches 'name' exactly. Only the first word of the value will be printed. With -t option we print everything following the '='.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
NAME

invgrad - take the inverse gradient of a FITS image

SYNOPSIS

invgrad

DESCRIPTION

'invgrad' applies inverse gradient operator to a gradient image using the FFT. It reads a N1 by N2 by 2 image from stdin (in format as generated by 'grad') and writes N1 by N2 image to stdout. It works by first fixing the end column of df/dx and the last row of df/dy to enforce periodic boundary conditions and we then apply the inverse discrete difference operator in k-space to generate two inverse gradient images (one from df/dx, one from df/dy) and we then combine these with weights \( w_x = k_x^{-2} / k^{-2} \) \( w_y = k_y^{-2} / k^{-2} \) and perform the inverse fourier transform.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.29  kappa2stuff(1)

NAME

kappa2stuff - generate shear etc from surface density

SYNOPSIS

kappa2stuff [options...]

"-f Nx Ny " set size of fft box

DESCRIPTION

"kappa2stuff" reads a fits file containing a smooth kappa image (perhaps generated by "lightmap ... ... | smooth ...") and generates a grid of distortion, shear, deflection etc values. By default it does an FFT in a box twice size of original image to remove periodic BC effects. Use -f option to set size of FFT (must be bigger than actual image) The output is in 'lc' catalogue format with items:

"x[2] " position
"kappa " surface density as input
"def[2] " deflection
"lambda[2] " eigenvalues of distortion tensor
"e[2] " distortion
"R " axis ratio
"phi " position angle
"parity " parity
"amp " magnification

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.30  lensmap(1)

NAME

lensmap - map a source image to a target image

SYNOPSIS

lensmap [option...]

"-d datafile " file for cluster parameters (cluster.dat)
"-o xo, yo " offset for the origin of the image (0,0)
"-m mode " mapping mode (1)

DESCRIPTION

"lensmap" maps a source image to a target image using multiple isothermal sphere lens model lens properties specified in datafile in form

erEinstein  rCore  xc  yc
all in pixel units

Use -m option to specify mode, where these are (in order of expense)

"mode = 0: " nearest pixel
"mode = 1: " linear interpolation
"mode = 2: " sum over triangles

AUTHOR

Nick Kaiser:  kaiser@cita.utoronto.ca
4.31 linesmode(1)

NAME

linesmode - calculate mode of rows/cols of a fits file

SYNOPSIS

linesmode [options...]

"-c " work in columns mode
"-f fitsf " fix fits file 'fitsf'

DESCRIPTION

By default, "linesmode" reads a N1 * N2 fits file from stdin, calculates the median for each row (might be useful to zap detected objects with 'makechart' first) and writes the result to stdout as a 1-D fits file. With -c option, we calculate the median for each column instead. With -f option we read the line modes (or column modes with -c) mode file from stdin, and subtract mode from the named fits file, writing the result to standard out.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
NAME

logscaleimage - scale image for printing

SYNOPSIS

logscaleimage [options...] sigma

where options are

"-b b " b-parameter (1.0)
"-a a " a-parameter (2.0)
"-f f0 " scale parameter (30)

DESCRIPTION

'logscaleimag' reads a fits image f from stdin and calculates f0 * 
(b + a * log(1 + f / (a * sigma))) which is linear in f for small values 
(|f| < a few sigma) but compresses higher values. Useful for printing 
images.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.33 make_image(1)

NAME

make_image - generates a mock CCD image

SYNOPSIS

make_image ncolours N1 N2 [option...]

"-o n r f0 " number, size and central SB of gaussian objects
"-e n r f0 " number, size and central SB of exponential objects
"-s sigma " rms sky fluctuation
"-b col " a bad column
"-r seed " for ran num generator
"-f " generate float format image
"-i " generate 32 bit int format image

DESCRIPTION

"make_image" generates a mock CCD image containing families of gaussian profile objects + noise + bad column defects. The first argument must be the number of colours. The second argument must be the size of the image on a side. Random number seed should precede -o or -s args Cosmic rays are indicated by a negative gaussian scale length.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
NAME

makecolourimage - combines 2 or 3 images into a color fits image

SYNOPSIS

makecolourimage [options....]

"-o offsetsfile " file containing positional offsets
"-f fits1 colorname1 " name of 1st image and its colour
"-f fits2 colorname2 " name of 2nd image and its colour....

DESCRIPTION

"makecolourimage" combines 2 or 3 images into a single color fits file using offsets in offsets.out if supplied. Names are arbitrary. Files should be supplied in increasing order of bandpass frequency.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.35  makesubimage(1)

NAME

makesubimage - extracts a subimage from a fits file

SYNOPSIS

makesubimage x y dx dy [options...]

"-o " set unmapped pixels to zero

DESCRIPTION

'makesubimage' extracts a dx x dy subimage with origin at (x,y) from the input image. By default, if the subimage extends beyond the source image then extra pixels are set to MAGIC. Use -o option to force these to zero. Reads from stdin, writes to stdout.

AUTHOR

Nick Kaiser:  kaiser@cita.utoronto.ca
4.36 mapbynumericdef(1)

NAME

mapbynumericdef - map image using numerical deflection
func

SYNOPSIS

mapbynumericdef [options...] sourcefits deflectionfits

"-i L1 L2 " perform inverse mapping
"-u " print this message
"-m mapmode " type of mapping
"-M " initialise image to magic
"-s sfac " use scrunched deflection image
"-b bgfits " background fits image to paint onto

DESCRIPTION

Mapbynumericdef maps a source image to a target image using a numerically
defined deflection function. It reads a N1 x N2 image fs = 'sourcefits'
and a M1 x M2 x 2 image d = 'deflectionfits' whose 0th ans 1st planes
are x and y components of a deflection function. By default it generates
a M1 * M2 image ft(r) = fs(r + d(r)), i.e. the deflection is given
as a function of target image coordinates.

If either of the source images is given as 'stdin' then the image will
be read from standard input.

With -i option it performs 'inverse' mapping --- i.e. deflection is
given as a function of source image --- so the resulting image satisfies
ft(x + d(x)) = fs(x) * M, where M is the magnification. With this
option, the deflection function image size should be matched to the
source image so M1 = N1 and M2 = N2, and the output image size is L1
x L2. Physically, inverse mapping projects a source image through
a lens.

Use -m flag to specify mapping mode:

287
"mode = 0: " nearest pixel

"mode = 1: " linear interpolation

"mode = 2: " sum over triangles the default being linear interpolation. With inverse mapping this flag is ignored and triangle mapping is used (which can be very slow).

The -s option allows you to read a miniature deflection image which has been scrunched by a factor $2^\text{sfac}$. This is currently not implemented for inverse mapping.

The -b option allows you to supply a background image to be painted onto. The size of this image will override either the dimensions determined from the deflection image size for forward mapping or the L1, L2 values set with '-i' option.

AUTHOR

Nick Kaiser: kaiser@ifa.hawaii.edu
4.37  mask(1)

NAME

mask - set abnormal pixels in sky-flat to MAGIC

SYNOPSIS

mask [options] fits_in fits_out

"-b inner outer " box for local median (3 5)
"-f dfcrit " critical (f - f_median) value (100)

DESCRIPTION

"mask" reads fits_in (typically a sky_flat) calculates a local median and sets to MAGIC any pixels with abs(f - f_local_median) > dfcrit Finally any neighbours of MAGIC pixels are also set to MAGIC. The MAGIC property will be inherited when the result is used to sky flatten an image

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
NAME

maskfits - set pixels in rectangles to MAGIC

SYNOPSIS

maskfits yyy.msk

DESCRIPTION

Filter which sets to magic any pixels falling within rectangles defined in yyy.msk. The latter must be an lc-format catalogue containing at least a pair of position vectors x1, x2 for diagonally opposite corners of the rectangle. Mask files may be generated by 'plotcat'. In general, these coords are floating point numbers, but get integerised on input using 'floor'.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.39 overscancorr(1)

NAME

overscancorr - dark subtraction using overscan region

SYNOPSIS

overscancorr [options....]

"-x xmin xmax " overscan region in x (2052,2086)
"-y ymin ymax " overscan region in y (0,2047)

DESCRIPTION

"overscancorr" corrects for the bias dc level. Reads the overscan region and fits a linear ramp model along the long (y) axis. The dc and gradient is subtracted from the image.

AUTHOR

Gordon Squires
NAME

pixdist - compute histogram of pixel values in a FITS image

SYNOPSIS

pixdist [options...]

"-d fmin fmax " histogram range
"-n nbins " number of bins (40)

DESCRIPTION

"pixdist" generates a histogram of pixel values from a fits image. Use -d option to specify range of pixvalues (default = fmin fmax) Use -n option to specify number of bins. Output is a lc-format catalogue with object items f, n giving the bin center and count and with header items overcount, undercount giving the count of pixels outside the histogram range.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
NAME

projectfits - average over rows, cols etc of a FITS file

SYNOPSIS

projectfits [-u] [-d M] d

DESCRIPTION

"projectfits" reads a FITS image of arbitrary dimensionality D from stdin and outputs a D-1 dimensional image to stdout which contains the average along the d'th dimension, where d=0 is the fastest direction (row average), d=1 is the next fastest direction (column average) etc.

Options:

"-u " print this message

"-d M " deproject

With the -d option we stretch out a N-dimensional image along the d'th direction to make a N+1 dimensional image. For example, with a 3-D input image fin[Nz][Ny][Nx], the result of projectfits -d 10 2 is to make a 4-dimensional image fout[Nz][10][Ny][Nx] with fout[z][t][y][x] = fin[z][y][x].

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

    scrunch - demagnify image by factor 2

SYNOPSIS

    scrunch [-m | -c]

    "-m " take median (mean is default)
    "-c " conservative mean

DESCRIPTION

Scrunches a fits image down by a factor 2. Works in stream processing mode. With -c option output is MAGIC if any of source pixels are MAGIC.

AUTHOR

    Nick Kaiser: kaiser@cita.utoronto.ca
NAME

simulate - generate mock fits image

SYNOPSIS

simulate [option...]

"-n N1 N2 " size of image (256, 256)

"-S As gs " amp and slope of stellar lum fun (As = 3e-4; gs = -1.5)

"-G Ag gg " amp and slope of galaxy lum fun (Ag = 1e-2; gg = -2)

"-R rbar lnrsig " av and SD log-normal distn of galaxy sizes (rbar = 2.0; lnrsig = 0.3)

"-s sigma " rms sky fluctuation (0)

"-i seed " for ran num generator (1)

"-l lmin " min luminosity (50)

"-e e " global ellipticity (0.0)

"-m mumin " min cos theta for galaxy inclination (0.5)

"-c catfile " output object positions to an lc-format catalogue

DESCRIPTION

"simulate" generates a mock fits image Galaxies and stars have power law lum funs Galaxies have log normal distn of sizes.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
NAME
slicimage - cut a 2-D FITS image into a grid of sub-sections

SYNOPSIS
slicimage [-u] [-i] nx ny

DESCRIPTION
In its default mode, slicimage reads a 2D image fin[N2][N1], whose
fast dimension N1 must be a multiple of nx and whose slow index N2
must be a multiple of ny, from stdin and sends to stdout a 4-D image
fout[ny][nx][N2/ny][N1/nx] such that
fout[y][x][Y][X] = fin[y * N2 / ny + Y][x + N1 / nx + X]

This slices a single image into a grid of contiguous patches.
With -i option it performs the inverse operation.
With -u option it sends this man page to stderr and exits with abnormal
status.

SEE ALSO
flatten(1), getplane(1), getplanes(1), stackplanes(1).

AUTHOR
Nick Kaiser: kaiser@hawaii.edu
NAME

smooth - spatially filter a fits image

SYNOPSIS

smooth [option...]

DESCRIPTION

'smooth' reads a fits image from standard input and writes a smoothed version to standard output.

Various types of smoothing are provided:

"-k m rf " smooth with m x m gaussian kernel
"-b m " m x m box filter
"-t " tukey-style running median
"-f a s1 s2 " fft filter \((1+k^2 s1^2)^{-a/2} \exp(-0.5 k^2 s2^2)\)
"-g a b phi " gaussian: major/minor = a/b, pos angle phi [deg]
"-K r " fft kolmogorov turb: \(\exp(-0.5 (k r)^{(5/3)})\)
"-p alpha " power law: transfer function = k^alpha
"-e r gamma " generalized exponential: \(\exp(-(k r)^gamma)\)

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
4.46  spinflip(1)

NAME

spinflip - apply 90 degree rotations or parity flips to FITS image

SYNOPSIS

spinflip mii mij mji mjj

DESCRIPTION

"spinflip" reflects images or rotates them through multiples of 90 degrees according to
\[ di' = Mii \cdot di + Mij \cdot dj \]
\[ dj' = Mji \cdot di + Mjj \cdot dj \]
with matrix elements \( M = -1, 0, 1 \) and with the constraint
\[ Mii^2 + Mij^2 = Mji^2 + Mjj^2 = 1 \]

AUTHOR

Nick Kaiser:  kaiser@cita.utoronto.ca
NAME

stackplanes - concatenate a set of fits files to multidimensional image

SYNOPSIS

stackplanes fits1 fits2...

stackplanes nplanes

DESCRIPTION

In the first invocation form (with 2 or more arguments) "stackplanes" reads a set of N-dimensional images from the files given as arguments and sends to stdout a single (N+1)-dimensional fits images whose planes are the input images.

In the second invocation form, "stackplanes" reads a single N-dimensional fits image, whose slowest dimension must be an integral multiple of nplanes, from stdin and generates a N+1 dimensional image whose planes are simply concatenated in the input image.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

stats - calculate simple statistics for an image

SYNOPSIS

stats [option...]

"-m n " ignore outer n pixel margin (def = 0)
"-v stat " just output value for 'stat', which can be one
of: N1 N2 pixtype min max mean mode median lquart uquart
sigma

DESCRIPTION

"stats" reads a 2-D fits file from stdin and writes the descriptive
statistics listed above to stdout. Mode, sigma are computed as described
in the 'catstats' man page.

If stats is given an image of dimensionality 3 then it will generate
a lc-format output giving the stats for the NAXIS3 planes (each of
size NAXIS2 x NAXIS1) and with an index 'i = 0 ... NAXIS3 - 1' giving
the plane number. With an image of dimensionality 4 it generates statistics
for the NAXIS4 x NAXIS3 planes, and the index i becomes a 2-vector
with i[0] = 0...NAXIS3-1, i[1] = 0...NAXIS4-1, and similarly for higher
dimensions.

SEE ALSO

catstats(1)

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
NAME

statstocat - convert stats output to lc cat format

SYNOPSIS

statstocat

DESCRIPTION

statstocat reads single image stats output from stdin and writes a lc cat format file to stdout

AUTHOR

Nick Kaiser:  kaiser@cita.utoronto.ca
NAME

transformimage - apply spatial linear transformation to a FITS image

SYNOPSIS

transformimage [options...]

"-p psi_xx psi_xy psi_yx psi_yy " distortion matrix (1,0,0,1)
"-t t_x t_y " translation vector (0,0)
"-n N1 N2 " size of output image
"-m mode " mapping mode (1)
"-c " keep centre fixed
"-i " do inverse transformation
"-f fitsfile " source for target image

DESCRIPTION

"transformimage" applies a general linear transformation to a source image $f_s(x)$ to make a target image $f(r) = f_s(x(r))$ where the mapping is $x_i(r) = \psi_{ij} r_j + t_i$. By default output image = input image size. Use -m option to specify mode, where these are (in order of expense)

"mode = 0: " nearest pixel
"mode = 1: " linear interpolation
"mode = 2: " sum over triangles With -c option we calculate $t_x, t_y$ so that the centre pixel is mapped to centre of output pixel. By default, target image is initialised to zero but use -f option to read in an image on which we paint the mapped pixels.
AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
4.51 translate_fft(1)

NAME

translate_fft - translate an image using 'sinc-interpolation'

SYNOPSIS

translate_fft dy dx

DESCRIPTION

'translate_fft' uses the FFT to translate an image.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
NAME

transposebits - swap order of bits and numbers in a FITS file

SYNOPSIS

transposebits [-i]

DESCRIPTION

"transposebits" reads a FITS file a line at a time and transposes the order of bits and numbers such that if the input bit array is b[x][i], where 0 <= x < N1, and 0 <= i < |BITPIX| then the output line is the bit array b[i][x]. Thus the N1 lowest order bits will be output first, followed by the N1 second lowest and so on. This may aid compression. Use -i option to perform the inverse transpose.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
4.53  unpackextensions(1)

NAME

    unpackextensions  - extract extensions from multi-part FITS file

SYNOPSIS

    unpackextensions extname fmt

DESCRIPTION

"unpackextensions" is used to separate a multi-component FITS file
where a number of separate images (of arbitrary dimensionality) have
been packed as extensions.

We first read the primary header, check that it has EXTEND = T and
get the number of extensions from the NEXTEND header item value. Then,
for each extension, we read the header and image data and write them
to a file with name 'filename' given by sprintf(filename, fmt, EXTNAME)
where 'fmt' is a format string and EXTNAME is the integerized value
of the header item named 'extname'.

For example, to unpack a CFHT archive format CFH12K file 504338o.fits
to 504338o00.fits 504338o01.fits ... 504338o11.fits you can do

    unpackextensions IMAGEID 504338o04.fits < 504338o.fits

since IMAGEID (or CHIPID) contains the chip number as an integer.

AUTHOR

    Nick Kaiser:  kaiser@hawaii.edu
NAME

unscrunch - expand a FITS image by a factor 2

SYNOPSIS

unscrunch

DESCRIPTION

expands a fits image by a factor 2.

AUTHOR

Nick Kaiser: kaiser@cita.utoronto.ca
NAME

warpimage - apply spatial transformation to a FITS image

SYNOPSIS

warpimage [options...] distparfile

"-n N1 N2 " size of output image
"-g sf N1 N2 " generate distortion image
"-m " initialise image to magic
"-M mode " mapping mode (1)
"-q qparfile " recircularise

DESCRIPTION

"warpimage" reads a fits file from stdin and applies a spatial transformation according to the parameters in 'distparfile' such that: \( r = x + \sum_{m} a_{m} f_{m}(x) \) where mode functions are polynomials in \( x[0], x[1] \). Use -p and -d options to apply a further linear transformation: \( r_{,i} \to r_{,i} + \phi_{ij} r_{,j} + d_{,j} \) By default output image = input image size. Output image is initialised to zero, unless you use -m option.

Use -M option to specify mode, where these are (in order of expense)

"mode = 0: " nearest pixel
"mode = 1: " linear interpolation
"mode = 2: " sum over triangles

With -g option (and s = 0) we generate a N1 by (2 * N2) image whose lower and upper halves contain the x, y components of the distortion \( d = r - x \). Set 'sf' to be some small integer to create a deflection image which has been demagnified by a factor \( 2^{-sf} \), but N1, N2 are
given in source image pixel scale, so ‘-g 3 1024 1024’ will generate a 128 x 256 pixel image.

With -q option we read a parameter file for a model of the field q = e / P_sm and then make appropriate additional deflections to recircularise the psf.

AUTHOR

Nick Kaiser: kaiser@ifa.hawaii.edu
NAME

imtools_fitstops - imtools_fitstops tools section

DESCRIPTION

fitstops is a tool for making encapsulated postscript greyscale, contour plots

COMMANDS

"fitstops " convert image from fits to postscript

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

fitstops - convert image from fits to postscript

SYNOPSIS

    fitstops fitsfile [option...]

    

    "-f fmin fmax " max and min f values

    

    "-n " don’t print header info

    "-b " draw a box around image

    "-c comment " use (quoted if multiword) comment as label

    

    "-p pw ph mx my " set page width, height, xmargin, ymargin

    (612, 792, 50, 50)

    "-P " include setpagedevice command

    "-g " pipe output through gs to compress

    "-C nc lw " draw nc contours of width lw

    "-s " draw 3D hidden line surface plot (alt = az = 30.0;

    zfacs = 3.0; zoffs = -0.2)

    

    "-S alt az zfac zoff " draw 3D hidden line surface plot

DESCRIPTION

"fitstops" reads fitsfile and, by default, sends postscript gray scale
or color image to stdout

The range of values may be specified with the -f option, otherwise
the range is 0 (=white) to 255 (=black).

Supports 1 and 3-color images.

If fitsfile = ’-’, then we read from stdin.
Default page size info is 612x792 = (8.5x11)in with 50 pt margin so the actual inked area is 512x692. Use -p option to change this.

The -P option is provided to include a 'setpagedevice' command giving the physical total page size. This is used for big prints on the designjet, but seems to be problematic with latex epsf handling. Do not use this with -g option.

With the -g option we pipe the output through gs -q -sDEVICE=pswrite -sOutputFile=- -dNOPAUSE - which will result in a much smaller output file. Do not use this with -P option.

Use the -s, -S options to generate a 3D hidden line surface plot using Tonry's mongo routine. The simple -s option uses default parameters which work reasonably well for a positive peak of height unity near the origin. Use the -S option to fiddle with the parameters alt, az, zfac, zoff where: alt, az = altitude azimuth viewing angle in degrees. zfac = scaling of z-axis (roughly 3.0 / data max) zoff = offset of z-origin.

AUTHOR

Nick Kaiser: kaiser@hawaii.edu
NAME

imtools_ic - imtools_ic tools section

DESCRIPTION

'ic' is the 'Image Calculator' which allows rather general operations on fits images (though restricted to 1-point operations where the output image at a given position is some function only of one or more input pixel values at the same position). It uses a simple reverse-polish notation to specify the operation you wish to perform.

COMMANDS

"ic " image calculator

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

ic - image calculator

SYNOPSIS

ic [options...] rpnexpr fitsfile...

where options are:

"-u " print usage

"-c N1 N2 " create an image of this size

"-p pixtype " specify output pixtype

"-s seed " seed rand num generator

"-m magicsub " substitute magic value

"-h name val " add header comment 'name = val'

"-b BSCALE BZERO " add scaling header information

DESCRIPTION

"ic" does arithmetic on one or more images according to the reverse-polish
notation expression 'rpnexpr'. Images are referred to in rpnexpr as
'\%1', '\%2'.... and must all have the same size. If the fitsfilename
is given as '-' then that file will be read from stdin. If fitsfilename
is given as 'command |' then we read from a pipe executing that command.

'ic' operates on images a line at a time, and so can be used on very
large images. A reference to an input image such as '\%1' causes a
line of that image to be pushed onto a stack. Single argument math
functions operate on the line at the top of the stack and multi-argument
functions pop lines as necessary and then push the resultant line.

If any of the input images is flagged as MAGIC (as defined 'fits.h')
then the result will be MAGIC also (though with '-m' option 'ic' will
output 'magicsub' in place of the usual SHRT_MIN)
Fits header comments are inherited from the first image — other comments are discarded. You may add extra comments with the -h option; see below.

The functions supported include all of the standard C math library (including Bessel functions $j_0(x)$, $j_1(x)$, $j_n(n,x)$, $y_0(x)$, $y_1(x)$, $y_n(n,x)$) plus the operators '$+', '$-', '$\ast$', '$/$, and 'mult' is provided as a synonym for '$\ast$' to avoid potential problems if you invoke ic from a script. There are the logical operations '>', '<', '>=', '<=', '!=', '==' and the negation operator '!'. There is a random number generator 'rand' which generates a uniform random number on the range 0.0-1.0 and 'grand' which generates a zero-mean, unit variance normal variate. There are two functions 'xp', 'yp' to get the horizontal and vertical pixel positions respectively, and two further functions 'x', 'y' which return the position in units of the length of the side of the image. There is a constant MAGIC (defined in magic.h — and currently set to -32768) which is a flag for bad data. There is a function 'if' (a.k.a. '?') which mimics the C language '(c ? t : f)' which returns 't' or 'f' respectively depending on the truth or falseness of the condition 'c'. The rpn syntax for this expression is 't f c ?' in which '?' pops the condition 'c' followed by 'f' and then 't' and pushes 't' or 'f' as appropriate. The condition 'c' will of course most likely be a compound logical expression. There are functions 'max' and 'min' which pop two values and pushes the maximum or minimum respectively.

There is also a function 'enter' which duplicates the top value of the stack, and a function 'swap' which unsurprisingly swaps the top two values on the stack.

Use -c option (with no input images) to generate an image from scratch.

Use -p option to specify output pixtype which can be one of

- 8 " 1-byte unsigned char
- 16 " 2-byte signed integer
- 32 " 4-byte signed int
- =32 " 4-byte floating point
- =64 " 8-byte floating point

Otherwise the output will have same pixtype as that of the first input image, or, with -c option, will have pixtype =32.

Use the -b option to apply scaling of pixel values on output and record the BSCALE, BZERO values in the header. Otherwise the BSCALE, BZERO values (if any) are inherited from the (first) input image. The definition
of BSCALE and BZERO is such that the internal values are computed from
disk values as:

\[ f_{\text{internal}} = BZERO + BSCALE \times f_{\text{disk}} \]

Output goes to stdout.

The '-h' option can be used to add new header values or to modify existing
ones. If an existing name is provided then the existing value will
be overwritten with the new value. Otherwise, or if the name is 'HISTORY'
or 'COMMENT', the new header line will be appended. The -h option
can be given repeatedly to insert a number of comments.

EXAMPLES

Subtract b.fits from a.fits:

ic '%1 %2 -' a.fits b.fits

Take sqrt of image to be read from stdin; output as float:

ic -p -32 '%1 sqrt' -

Create a 512 x 512 image with a linear horizontal ramp and multiply
by 10:

ic -c 512 512 'x 10 *'

Replace all pixels in a.fits with value < 10 with MAGIC:

ic '%1 MAGIC %1 10 > ?' a.fits

Filter to clip image at fmin = 0.25 fmax = 0.75:

ic '%1 0.25 max 0.75 min' -

AUTHOR

Nick Kaiser - kaiser@hawaii.edu
NAME

imtools_iis - imtools_iis tools section

DESCRIPTION

'iis' provides a way to pipe image data directly to the 'saoimage' display (thus avoiding having to click buttons and type filenames within the saoimage window). A significant limitation is that it does not currently work for images larger than 2048 x 2048.

COMMANDS

"iis " pipe fits image data into saopipe

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

  iis - pipe fits image data into saopipe

SYNOPSIS

  iis [options...]

where options are:

  "-u " print this message
  "-f fmin fmax " limits for pixel values
  "-p pipename " supply FIFO pipe name

DESCRIPTION

"iis" reads a fits image from stdin, prepends an iis header and writes the output to a FIFO pipe /dev/imt1o so that saoimage can display it. If the standard FIFO does not exist (it typically won’t unless you have IRAF installed) then create one with e.g. ‘mknod /dev/saopipe p’ and then fire up saoimage with ’saoimage -idev /dev/saopipe k’ Beware: some old versions of saoimage crash when you try to supply 'idev' argument. Many thanks to Karl Glazebrook for refinements to original code.

AUTHOR

  Nick Kaiser (kaiser@cita.utoronto.ca)
4.59  imtools_xfv(1)

NAME

    imtools_xfv - imtools_xfv tools section

DESCRIPTION

'xfv' is an X-based Fits image Viewer I developed. It reads a stream of 2D fits images and displays them on the screen as a movie. There are two versions. The one built by default uses the Imlib libraries. You can obtain these from <a href="http://www.rasterman.com"> Rasterman’s Zone of Cruft </a>.

COMMANDS

"xfv " FITS stream viewer

AUTHOR

    Nick Kaiser -- kaiser@hawaii.edu
NAME

xfv - FITS stream viewer

SYNOPSIS

xfv [-options]

DESCRIPTION

xfv is an X-based FITS Viewer for viewing 3D FITS images as a movie.

Options include:  -min fmin minimum f-value (0.0) -max fmax maximum f-value (255) -b blocksize paint pixels this size (1) -c cmapindex colormap (-1)

AUTHOR

Nick Kaiser --- kaiser@hawaii.edu
4.60  imtools_xim(1)

NAME

    imtools_xim - imtools_xim tools section

DESCRIPTION

'xim' is a colour image viewer I developed. It is not very fast (compared to e.g. saoimage), but it does allow one to read 2- or 3-color images and to modify the colormap dynamically. I have found it useful in adjusting the color mapping for making color postscript images to send to a printer. A major limitation is that it only reads 16 bit images, and, if these are color, then the format is as created by makecolorimage.

COMMANDS

    "xim " colour image display

AUTHOR

    Nick Kaiser -- kaiser@hawaii.edu
NAME

xim - colour image display

SYNOPSIS

xim [-options]

DESCRIPTION

xim is an X-based image displayer. It can display monochrome or colour fits images created with makecolourimage but these must be in 16 bit format. Options include: -min fmin minimum f-value (0) -max fmax maximum f-value (256) -goodsize good window size (512)

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

nk - nk tools section

DESCRIPTION

This section contains a number of simple utilities.

Some scripts with no man page:

"bundle " - simple 'shar' type archiver to bundle text files
"getbundle " - extract text files from a bundle
"echoeval " - 'echo' a command line and then 'eval' it --- useful to make scripts generate a log
"fixpgplotps " - fix bounding box in pgplot postscript files
"notabs " - replace tabs with spaces
"overwrite " - copy standard input to output after EOF
"pscat " - concatenate two or more postscript files
"remote " - rsh a command in current directory on remote machine
"replace " - replace str1 in files with str2, in place
"reverse " - list file in reverse order
"rotateps " - rotate a postscript file
"translateps " - translate a postscript file
"stree " - list directories...
"swap " - swap two files
"tfilter " - awk script to filter table of numbers
"tfilter " - awk script to generate histogram from table
"Top " - mimics 'top'
"tproc " - awk script to process table ignoring comments
"travesty " - my favourite perl script
"truepath " - Report true full pathname of file(s)
"tsort " - sort a table of numbers ignoring comments
"thist " - generate histogram

COMMANDS

"clickstocat " convert contour -C output to scaled cat
"doto " perlscript to automate overwriting of processed file
"makecatmanpage " perl script to generate a man page
"makegroffmanpage " perl script to generate a groff man page from cat version
"makehtmlmanpage " perl script to generate a html man page
"makeimcatman " perl script to generate top-level man page and the incat users manual
"makemanpages " perl script to generate man pages in various formats
"maketexmanpage " perl script to generate a tex man page
"xevtocat " capture mouse click coordinates

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
5.1 clickstocat(1)

NAME

clickstocat - convert contour -C output to scaled cat

SYNOPSIS

clickstocat x1 y1 x2 y2

DESCRIPTION

clickstocat reads output generated with contour -C and scales the coordinates.

You can use this to generate fairly accurate data from e.g. postscript files using the mouse. It assumes that the first 2 clicks correspond to (x1, y1), (x2, y2) and then scales the rest appropriately.

EXAMPLE

To rip data points from a postscript file my.ps:

cat my.ps | gs -q -sDEVICE=pbmraw -sOutputFile=- -r50x50 - | pnmflip -tb < tmp.pbm | pnmtofits | contour -g -n 0 -C > tmp.out

Then click on points in the PGLOT window. First two points should correspond to (x1, y1), (x2, y2) and subsequent points should be the data points.

Then run

clickstocat x1 y1 x2 y2 < tmp.out

to generate the scaled cat.

AUTHOR

Nick Kaiser - kaiser@hawaii.edu
5.2 doto(1)

NAME

doto - perlscript to automate overwriting of processed file

SYNOPSIS

doto file command

DESCRIPTION

doto executes the shell commands

command < file > doto.tmp

mv doto.tmp file
5.3 makecatmanpage(1)

NAME

makecatmanpage - perl script to generate a man page

SYNOPSIS

makecatmanpage commandname | libname headers

DESCRIPTION

With a single argument we invoke the command ‘commandname’ with ‘-u’ option to generate the cat-style man page.

With multiple arguments we output a man page for a library ‘libname’
5.4 makegroffmanpage(1)

NAME

makegroffmanpage - perl script to generate a groff man page from cat version

SYNOPSIS

makegroffmanpage commandname sectionname

DESCRIPTION

makegroffmanpage reads a cat-style man page from stdin and sends a formatted version to stdout.
5.5 makehtmlmanpage(1)

NAME

makehtmlmanpage - perl script to generate a html man page

SYNOPSIS

makehtmlmanpage

DESCRIPTION

'makehtmlmanpage' reads a groff-style man page from stdin and wraps this in html code to generate a web-page.
5.6 makeimcatman(1)

NAME

makeimcatman - perl script to generate top-level man page and the imcat users manual

SYNOPSIS

makeimcatman [-b | -t] docdir section ....

DESCRIPTION

We first sort the command line arguments. We then generate a top level index page.

With -b option this is copied to imcatman.groff followed by the section index pages, followed by the actual command the man pages.

With -t option we make a tex format reference manual imcatref.tex.
5.7 makemanpages(1)

NAME

makemanpages - perl script to generate man pages in various formats

SYNOPSIS

makemanpages sectionname mandir docdir tools... scripts...

DESCRIPTION

Generate man pages in various formats.
5.8  maketexmanpage(1)

NAME

maketexmanpage - perl script to generate a tex man page

SYNOPSIS

maketexmanpage [n] [-u]

DESCRIPTION

'maketexmanpage' reads a groff-style man page from stdin and converts to latex.
Optional integer argument 'n' shifts sub-section level up by n.
Use -u option to print this man page.
5.9  xevtocat(1)

NAME

xevtocat - capture mouse click coordinates

SYNOPSIS

xevtocat x1 y1 x2 y2

DESCRIPTION

xevtocat reads xev output and generates a lc-format catalogue containing
coordinates of mouse-button clicks

You can use this to generate fairly accurate data from postscript files
using the mouse.

EXAMPLE

To digitize data from 'foo.ps', first run

gv foo.ps &
to bring up a gv window. Then run

xwininfo

click once in the gv window and copy the id to the clipboard. Then run

xev -id id > xev.out

go back into the xv window, and middle button click on a series of
points, the first two of which should be conveniently chosen reference
points. Add a few more points at the end, as some may get lost, and
then control-C in the window where xev was launched to kill it. Finally,
run

xevtocat x1 y1 x2 y2

supplying the user-coordinates of the two reference points.
AUTHOR

Nick Kaiser - kaiser@hawaii.edu
5.10  nk_tablestuff(1)

NAME

nk_tablestuff - nk_tablestuff tools section

DESCRIPTION

This section contains some commands that perform operations on simple ascii format tabular data. They all ignore lines beginning with #, and so can read text-format lc output.

COMMANDS

"mova " calculate moving average of tabulated data
"tablearith " do simple math on a table of numbers
"tablecumulate " cumulate a column of numbers in a table
"tablefilter " filter a table of numbers
"tableop " apply operation to table of numbers
"tablesort " sort a table of numbers
"tablesum " sum a table of numbers

AUTHOR

Nick Kaiser -- kaiser@hawaii.edu
NAME

mova - calculate moving average of tabulated data

SYNOPSIS

mova xmin xmax nbins binwidth

DESCRIPTION

mova calculates average of y-values (column 2) binned according to x-value (column 1) with nbins bins of width binwidth between xmin and xmax. It outputs a table with \( <y>, x, n \) for non empty bins.

AUTHOR

Nick Kaiser - kaiser@hawaii.edu
5.10.2  tablearith(1)

NAME

tablearith - do simple math on a table of numbers

SYNOPSIS

  tablearith col op operand

DESCRIPTION

tablearith read lines of a table containing lines X_1 X_2 X_3 ..... from stdin lines beginning with "#" and empty lines are ignored changes the col'th column entry op can be x / + or - e.g. tablearith 2 / 3.0 will divide 2nd column by 3

AUTHOR

  Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

tablecumulate - cumulate a column of numbers in a table

SYNOPSIS

    tablecumulate col

DESCRIPTION

tablecumulate read lines of a table containing lines X_1 X_2 X_3 ..... from stdin lines beginning with "#" and empty lines are ignored cumulates the col'th column entry

AUTHOR

    Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

tablefilter - filter a table of numbers

SYNOPSIS

    tablefilter col minval maxval

DESCRIPTION

tablefilter read lines from a table from stdin lines beginning with
"#" and empty lines are passed as comments as are lines in which value
of the col'th entry has (x >= minval && x <= maxval). Entries spearated
by tab(s) or space(s). Ist column is col = 1 max line length 4096
characters

AUTHOR

    Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

tableop - apply operation to table of numbers

SYNOPSIS

    tableop col op

DESCRIPTION

tableop read lines of a table containing lines \( x_1 \ x_2 \ x_3 \ldots \) from stdin lines beginning with "#" and empty lines are ignored changes the col’th column entry op can be one of exp, ln, dex, log e.g. tableop 2 ln will replace 2nd column its natural log

AUTHOR

    Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

tablesort - sort a table of numbers

SYNOPSIS

tablesort col

DESCRIPTION

tablesort read lines of a table from stdin lines beginning with "#"
and empty lines are ignored lines sorted in ascending order of colth
column values. negative column number for descending order

AUTHOR

Nick Kaiser --- kaiser@cita.utoronto.ca
NAME

tablesum - sum a table of numbers

SYNOPSIS

  tablesum c1 p1 [c2 p2 ....]

DESCRIPTION

tablesum read lines of a table containing lines \( X_1 \) \( X_2 \) \( X_3 \) ..... from stdin lines beginning with "#" and empty lines are ignored returns sum \( X_{c1} \) \( p1 \) if additional c,p pairs are given the it returns sum of \( (X_{c1} \) \( p1) \) \( (X_{c2} \) \( p2) \) *..... max line length 4096 characters, max number of cols = 32.

AUTHOR

  Nick Kaiser --- kaiser@cita.utoronto.ca