file: sdo-manual.txt = simple manual for my SDO data processing
init: Dec 8 2013 Rob Rutten Sac Peak
last: Apr 18 2021 Rob Rutten Deil
func: Jan 17 2021 "all systems go" except gong_sdo.pro
site: ~/rr/idl/rjrlib/00-README, parallel versions .txt, .html, .pdf
      accessible at RR page "Recipes for IDL"

   https://robrutten.nl/www/Recipes_IDL.html

note: if you don't know what SDO means this is not for you
      # or ; precedes a comment
blog: Apropos.jot in sdolib
hist: version history at file bottom below (since Dec 2020)
step: 1 = find your solar location
      2 = get SDO data, cross-align all AIAs to HMI
      3 = co-align SDO data and your STX data (STX = "Solar Telescope X")
      4 = browse co-aligned data

This manual

Content: this is a brief manual in telegram style on how to use my pipeline to obtain and cross-align SDO cutout image sequences and co-align these to other data ("STX" = solar telescope X, sofar SST and DST. I use "cross-" for aligning different SDOs and "co-" for SDO-STX alignment.)

Motivation: using cutouts avoids downloading full-disk AIA and HMI images and so improves the download and processing speeds if the desired solar field of view is small (groundbased solar telescopes, Sunrise, IRIS, etc.). The required disk space is correspondingly minimal. The required computer memory is also small because all programs use assoc into data files (which may be any size).

However, the pipeline may also be used to obtain full-disk long-duration SDO sequences, slower than simply downloading SDO full disk images but offering precise cross-alignment and differential derotation across the disk.

Websites:
  my website:
   http://robrutten.nl/www

  my IDL libs in one download (refresh to date of func line above)
   http://robrutten.nl/www/rridl/rjrlib.zip

  JSOC for getting SDO data (called within these programs):
   http://jsoc.stanford.edu/ajax/exportdata.html

  daily SDO movies:
   http://sdo.gsfc.nasa.gov/assets/img/dailymov

AIA cutouts (slower than JSOC; for older dates no HMI; not used by me);
http://www.lmsal.com/get_aia_data

SolarSoft:
http://www.lmsal.com/solarsoft

Fanning's coyotelib:
http://www.idlcoyote.com/documents/programs.php

My IDL programs:
- are split over topical subdirectories under rjrlib
- are densily commented (both meanings apply)
- use fits files and accept older SST "La Palma" and DOT files
- use assoc to permit data files exceeding the computer memory
- can often be run for multiple similar data files with doallfiles.pro
- most have a usage example as main at file bottom (my one-click runtest from emacs per IDLWAVE; do not try .r program.pro or first switch the directory specification at the start of the main part to your own data)
- usually answer a no-parameter call by showing the start-off comment block in a gedit session using my sp.pro (try IDL> sp,sp; if gedit lacks then install it, or change sp.pro to another editor, or make sp.pro a null program).

I run ancient IDL 6.4 (2007, my forced retirement) under Ubuntu but only seldomly run into SSW problems from it being too old. These I solve with overrides in fixsswlib.

Some MacOS IDL problems are solved.

My linux setup:
My IDLWAVE .emacs setup is shown on my website under "Recipes for linux". Options for producing movies (e.g., calling sdo_maketargetmovie.pro) require linux/unix commands "convert" (imagemagick), "ffmpeg", and more. Don't call these options if your installation does not have those. The programs also call "fpack" and "funpack" for data compression but work also when these are not present.

Examples:
some given in the appendices of LAR 1
https://robertten.app/www/rweb/rjr-pubs/2020LingAstRep...1R.pdf

Flow:
Step 0: install my IDL programs;
Step 1: define your desired SDO cutout target;
Step 2: order and get SDO sequences, cross-align into "fitscube" files;
Step 3: co-align these SDO fitscubes with your STX data;
Step 4: inspect/browse results.

```
Advice on choices
-----------------
  Forward and/or rotated and/or reverse and/or full output mode(s)?
      = sdo2stx, stx2rotsdo, stx2sdo, stx2fullsdo
    Forward: the SDO sequences are cropped, rotated, interpolated
             (spatially and temporally) onto the STX sequences; the
             latter are not modified except for optional pixel
             asymmetry correction and resampling to AIA pixels when smaller.
    Rotated: put STX on cropped SDO that is first rotated to STX
             orientation.  STX resolution if better, regularized STX timing.
    Reverse: rotate STX onto cropped SDO, STX resolution if better, SDO timing
    Full:    put STX on the non-cropped full-duration SDO cutouts
             at SDO resolution and timing.
    The last two have solar (X,Y) orientation unless option rotate_up is
    set: then the direction to disk center is down (nearest limb up,
    vertical = radial structures upright: "up=up").
    The co-alignment itself is done forward in order to maximize the STX
    field when it is much rotated (as for SST/CRISP) and to not
    re-interpolate finer-scale STX data.

    I prefer rotated = stx2rotsdo for SST articles because the SST
    figures then do not show ugly empty de-rotation corners from
    rotating back to the (X,Y) orientation of SDO over the large and
    time-varying angle that the SST suffers from its alt-az mount.
    Then only small left-over triangles remain from the temporal
    modulation around the mid-sequence value.  In this and the other
    stx2*sdo modes the tracking follows the standard SSW differential
    solar rotation law, not the drifts and bad-seeing jumps affecting
    feature tracking with ground-based telescopes (or the pointing
    wobble of IRIS).  Only the small (0.3 arcsec) 24-hour sub-SDO
    pointing ellipse remains.  Such more stable tracking is desired
    for temporal Fourier and scatter analyses of given features or
    areas that are stationary in differentially-rotating location.

    I use the "reverse" modes stx2sdo and stx2fullsdo for analysis
    studying what SDO showed wider-field and longer-duration.  I then
    use showex in two parallel sessions to compare hi-res STX
    phenomena in stx2sdo to the low-res SDO environment in
    stx2fullsdo.  The two time sliders are then the same (both SDO
    timing).  These outputs also suit when up=up orientation is
    desired.

  Interpolate (splinip=1) or pixelate (splinip=0)?  In both sdo2stx
    and stx2sdo the resulting SDO images are on the SXT-pixel scale
    when finer.  The alignments use spline interpolation for sub-pixel
    pattern matching.  This is usually the best choice for visual
    feature matching per inspection (e.g., with showex or crispex).
    However, for publication figures one may favor original pixelation
```

for the SDO channels to demonstrate their awfully large 0.6 arcsec size. At large STX rotation the SDO pixels in sdo2stx then become ragged over the pixel ratio. For SST/CRISP (ratio 10 or larger) this is not disturbing, but for IRIS (ratio about 4) the raggedness gets ugly. Nearest-neighbor pixelation is also better if an image pair to be interpolated has large small-feature shifts that double in appearance with splinip.

Use applypx2asym or not? With this applied the Metcalf iteration in findalignimages.pro solves simultaneously for image rotation, 2D shift, and 2D scale difference. Without it the scale fit is only 1D. I found that not setting applypx2asym is better for SST/CRISP whereas setting it is needed for DST/IBIS (its pixels have over 1% anisotropy). If the Metcalf iteration diverges the first remedy is to set or unset this option.

Use trimbox or not? Set trimboxstx to cut off blank edges and corners for the STX area with which the co-alignment is done (the result is always full-field). It is als needed to get better greyscaling in the showex browser and for honest scatter diagrams. For SST/CRISP this setting is not needed since the pipeline (since 2017) removes blank border strips and regular derotation triangles (but detecting the latter may be unreliable). For DST/IBIS it serves to select a box within the old-style circular field delimiter.

Muck with my image mucking? Extensive image mucking based on my best knowledge of how solar scenes form and can be matched best for different diagnostic pairs occurs both in the internal SDO AIA-to-HMI cross-alignments and in the SDO-STX co-alignment. This mucking only affects the trimbox-selected subfield. If you run into bad alignment it may help to play with the various muck parameters. Example: AIA-1700 matches very well to magnetograms after undoing polarity and removing umbrae and acoustics - but even then only at the center of the disk because off-center the mechanism by which magnetic elements become bright points differs per diagnostic: not simply "height of formation" (as many believe) but more intricate differences in how fluxtube evacuation (not heating!) affects diagnostics. Example: AIA-1600 and AIA-1700 differentially shift much towards the limb; do not blame that to cross-alignment mismatch! (Explanation in appendix A of LAR-1.)

Apply sdo_intscale.pro or not? Usage is default and adapts AIA and HMI-continuum intensities to better accommmodate the dynamic range in 8-bit-only displays and the resolution in the 16-bit integer cubes. For AIA-094 it takes the square root, for other EUVs the log. The HMI magnetograms are always changed into units of 0.1 Gauss to enable integer output. It also compensates for variations in exposure duration and for the gradual sensitivity

```
    loss of most AIA channels since launch, particulalry severe for
    304.  See sdo_intscale.pro for details.  If you instead prefer the
    original data values (excepting magnetograms) then set
    /no_intscale in sdo_getdata (or the underlying programs when you
    use these separately).  You can also change its application or its
    clipmin and clipmax settings subsequently by running
    sdo_maketargetcubes.pro anew avoiding lengthy data ordering,
    download, aia_prep.

  Apply diffdesolrot to target or not?  Normally JSOC-im_patch-tracked
    target cutouts track solar rotation by shifting to the rot_xy
    result for target field center.  For large target fields and long
    duration option /diffdesolrot in sdo_getdata etc instead applies
    differential derotation across the field to make features appear
    stationary everywhere, out to near the limb.  When co-aligning
    diffdesolrotated SDO with correlation-tracked STX sequences the
    latter must then also be diffdesolrotated including prior
    de-derotation with the dedesolrot option of diffdesolrotate.pro
    (see example using doallfiles.pro in its comment block with
    sp,diffdesolrotate).  The SDO cross-alignment center fields are
    fairly large but do not need differential corrrection because they
    are compared only locally at small time lags.


Step 0: install IDL programs
----------------------------

  Install or refresh my "rjrlib" dir by pulling over the zip file at
   http://robrutten.nl/www/rridl/rjrlib.zip
  unzip, put rjrlib in your IDL search path.
  Refresh rjrlib whenever the func line at the top of this manual
  specifies a newer modification date.
  If you work in the ITA/RoCS cluster you may simply add
  ~rutten/rr/idl/rjrlib to your IDL search path (always up to date).

  You also need the SolarSoft (SSW) lib, and for some programs
  Fanning's coyotelib (websites above).

  In order to IDL-load coyote originals before SSW-changed coyote
  copies, and also to override-load any same-name fix-SSW program I
  sometimes need, I have in my idlstartup.pro:
    cd,'~/xxxx/idl/coyote',current=thisdir
        addtopath
        cd,thisdir
        cd,'~/xxxx/idl/rjrlib/fixsswlib',current=thisdir
        addtopath
        cd,thisdir
        cd,current=workdir
```

```
      addtopath,workdir

  You must be registered at JSOC (name and email address):
   http://jsoc.stanford.edu/ajax/register_email.html

  Step 1 programs sst_findlocation.pro, stx_findlocation.pro and
  sdo_featurelocator.pro get images per ssw_jsoc_time2data.pro
  which may want your email address.  Specify that (once for all) in
  $SSW/site/setup/IDL_STARTUP as in:
  set_logenv,'ssw_email_address','R.J.Rutten@uu.nl'


Step 1: define your desired SDO cutout sequences
------------------------------------------------

  Define (X,Y) of desired cutout center precisely for the start time
  of the desired SDO data sequences (solar rotation moves your target
  up to 10 arcsec/hr across the central meridian).

  For SST data use:
    IDL> sst_findlocation,sstfile,obsdate,ssttimesav,$
           x_sst,y_sst,angle_sst,px_sst [, options]
  which precisely locates your SST scene in a near-simultaneous SDO
  image, starting by downloading SST turret pointing info (or with
  better values if you supply these).

  For data from other telescopes ("STX") use the more general
    IDL> stx_findlocation,stx_im,datetime,$
           x_stx,y_stx,angle_stx,px_stx [, options]
  which either needs good starting guesses for (X,Y), angle, px
  or very long runs trying a range of angles assuming datetime and px OK.

  If you know only a rough position or have only an image with a
  sizable recognizable feature (say a sunspot or filament or plage
  patch, for example in some publication that you want to SDO-check or
  on your live telescope display, than find its precise coordinates
  with:

  IDL> sdo_featurelocator,specifier,wav [,rotate=rotate,fixfov=fixfov]
       e.g.: sdo_featurelocator,'2014.01.01_12:00','magnetogram'
       NB: these quotes must be straight-up single quotes
     # Shows an SDO image in the requested SDO diagnostic (unless
       there was an SDO eclipse), either an archived full-disk image
       (more than a week ago) or the very latest (at lower resolution)
       to help target a telescope.  Draw out the recognized feature
       and click on the desired cutout center location to print its
       solar coordinates [(X,Y), (latitude,longitude), Carrington
       values, mu].  It also works on target/level2 cutout images from
```

```
      the pipeline below.  For targeting use keyword options 'rotate'
      and 'fixfov' to get direct similarity to your STX display.
      Regretfully no HMI 'continuum' for specifier='latest' (while
      AIA '4500' is near-useless).

  IDL> gong_featurelocator,specifier [,rotate=rotate,fixfov=fixfov]
       e.g.:  gong_featurelocator,'2014.01.01_12:00'
     # Shows full-disk GONG Halpha image (if available for the
       specified minute, or the latest one) for eyeball comparison
       with STX Halpha.  Better first sum the latter over 0.4 Angstrom
       and smear it over 1-2 arcsec to make it as bad as the GONG one.
       Zoom in on the recognized Halpha feature and click on the
       desired cutout center location to print its solar coordinates.


Step 2: order, get, process SDO cutout sequences
------------------------------------------------

  IDL> sdo_getdata_rr,datetimestart,duration,xsol,ysol,[options]
    e.g.: sdo_getdata_rr,'2015.09.27_09:00',50,766,-218,$
             name='xxx',email='yyy@zzz'
          (the _rr denotes that my JSOC identity is default)
    NB: these quotes must be straight-up single quotes

    Parameters: date_time string, duration in minutes, solar (X,Y).
    Keyword options include: target field size, notrack, diffdesolrot,
    no_intscale, wavelength selectors.  See the initial comment block
    in the program (as for all my programs) or open that with IDL>
    sp,sdo_getdata_rr. I use calls as this one for SDO-SST co-alignment
    to get 130x130 arcsec SDO fields around the smaller SST field.

    A long-duration full-disk example (containing Mercury):
      sdo_getdata_rr,'2019.11.11_11:00',660,0,0,xsize=2200,ysize=2200,$
        centercadence='5m',targetcadence='5m',/notrackcenter,$
        /diffdesolrot, name='xxx',email='yyy@zzz'

    This is a robust black box!  It is a wrapper first calling:

  sdo_orderjsoc.pro

    # emulates JSOC exportdata webform filling within IDL.  You will
      get 8 JSOC emails that you can ignore unless the topic line says
      "FAILED" (may be SDO eclipse?).  You get no emails when
      duplicating a recent request: JSOC recognizes such and
      returns the corresponding existing identifier.  If the ordering
      is automatically time-segmented to accommodate long durations
      (generating more files then the JSOC limits) you get nsegments*8
      emails.  I made Gmail regard JSOC mails as spam so that I don't
```

```
    get them in my inbox (but can still check them in the spam box).
    Since February 2020 JSOC forbids running multiple calls in
    parallel; this program solved that by internally waiting on yet
    busy orders, but JSOC waits eternally at multiple-call conflicts
    so do not start another sdo_getdata call before you see:
    " ===== sdo_getdata starts getting JSOC target data".

sdo_getjsoc.pro
  # downloads the requested tarfiles from JSOC within IDL,
    with waits when JSOC hasn't yet completed an order.

These 2 JSOC programs are called twice, first to get large-field
low-cadence disk-center cutouts which serve to measure offsets and
drifts between all SDO diagnostics, and then for your target area to
which these offsets and drifts are subsequently applied.

I wrote analogous programs sdo_orderssw.pro and sdo_getssw.pro that
do not use JSOC but ssw_cutout_service.pro, but these are 2-3x
slower, less robust, and didn't solve a persistent aia_prep MP table
mistreat for cutouts.

sdo_getdata_rr.pro then calls:

sdo_alignaia_rr.pro
  # another wrapper program to cross-align SDOs.  One option (passed by
    sdo_getdata_rr.pro) is /aligncenter to additionally produce
    ./center/cubesxal containing cross-aligned disk-center SDO fitscubes
    for checking (with showex) the internal SDO cross-alignment quality.
  # In that case it displays ./driftscenter/check_xal_xx-yy.ps which
    should show excursions about 0.5 px or less (xx, yy are no steps
    in the chain but end products of two chains, so that this plot
    shows summed errors along these chains).
  # This program can also be run on its own if you already have
    level1 (then set /sdoprep) or level2 (set no /sdoprep) image files.
    Set /sdopreptarget if dir center was done, dir target not yet.
  # Set option /no_intscale if you want no intensity rescaling
It calls successively:

sdo_getdriftsplines.pro
  # determines SDO offsets and drifts; results in subdir driftscenter
    also containing driftplots for each cross-alignment pair.  These show
    +-95% confidence bars per time sample (statistics from the tile
    subdivision of the center field), specify their mean, and specify
    standard deviations from the spline solutions shown as solid curves.
   # uses:
      # sdo_muckimagepair.pro: make two SDO images look alike
      # findimshift_tiled: find offsets by tiling center-field image pair
      # sdo_writepairspline.pro: find tiled offsets throughout sequence
```

```
      # sdo_writeallpairsplines.pro: do this for chains of pair combinations
  # includes a call of next program without /dedrift cross-alignment

sdo_allimages2fitscubes.pro
  # processes SDO image sequences to produce synchronized fitscubes
  # files in center/cubesxal and target/cubes are cross-aligned
  This is another wrapper calling:

  sdo_prep.pro
    # aia_preps all level1 to level2 using a sliding reference
      to undo solar-rotation 1-px sawtooths left by JSOC
      NB: my level2 is NOT SDO "level 2" but SDO "level 1.5" because
          I don't want periods (nor spaces) in directory names
    # gets rotation and scale from MPO but not shifts
    # takes ages (2s or more per image for my laptop)
    # spews tons of ignorable output (16 lines/image) to the IDL window

  sdo_makereffitscube.pro
    # produces reference cube, default 171 (don't use 1700 which has gaps)
    # determines X shifts to undo leftover whole-pixel rotation sawtooth
    # finds crop size to strip blank edges left by the rotation correction

  sdo_makeslavefitscube.pro
    # generates fitscube interpolated to the cadence of the reference cube
    # evaluates and applies cross-alignment "sdoshifts" using:
      # sdo_get_onesplineshift.pro: get pair spline offset per timestep
      # sdo_getsumsplineshift.pro: do for all, current sequence order:
          HMI-mag>1700; 1600>1700; 304>mag; other EUVs to 304
    # shifts each level2 image using:
      # sdo_shiftjsoc.pro:
       - undoes JSOC tracking sawtooth from whole-pixel cutting
       - de-derotates differentially if diffdesolrot is set
       - shifts over the border crop done in sdo_makerreffitscube.pro
       - shifts over the current sdoshift for this wavelength
    # crops to reference cube size
    # rescales intensities using sdo_intscale (unless /no_intscale set)

  sdo_allfitscubes2mpegs.pro
    # optional: produce target/cubes mpegs (needs convert, ffmpeg)

  sdo_maketargetmovie.pro
    # optional: produce 4-panel movie of target (needs convert, ffmpeg)

The individual programs can be run on single channels, eg for other
intensity clipping values for AIA wavelengths (see sdo_intscale.pro
header).  For such rescaling you can redo all target cubes with:
  sdo_maketargetcubes.pro
    # remake all target SDO cubes from ./target/level2 image files and the
```

```
      ./driftscenter spline fits, specifying clipping values or noscaleaia
      and movie making options.
    # Example call:
      sdo_maketargetcubes,'target','driftscenter',clipmax=3,/mpegs
or redo only a single cube with:
  sdo_makeslavefitscube.pro
    # example call:
      sdo_makeslavefitscube,'target','94',dedrift=1,/no_intscale
      works also for the reference wavelength (default '171')

Warning: the SDO cross-alignment operates along sucessive pairs:
  1700 > hmimag, 304 > mag, 171 > 304 and further EUV pairs > 304,
  so that you must always include hmimag and 304 in your ordering;
  sdo_getdata_rr.pro warns if you don't.

Speed: for SST observations I usually (default) request 150x150
  arcsec cutouts of all AIA channels plus HMI continuum and
  magnetograms, each at fastest cadence.  JSOC used to take only a
  few minutes but many more on waits since in early 2020 it forbade
  parallel firing.  Then data download needs minutes only (500 Mbps
  at Lingezicht) and then my laptop (2 GHz quadcore) needs time for
  endless sdo_prep, often roughly half the ordered duration.

Hint: always keep ./target/level2 files and ./driftscenter data to
  be able to regenerate [say other intscale] cubes.
  I usually delete the complete /center directory and the /target/level1
  directory to clean disk space.


Step 3: co-align SDO target data to STX data or reverse
-------------------------------------------------------

  IDL> sdo_stx_align,<parameters>,<keywords>

  This SDO <> STX (vice-versa) program does not handle time-dependent
  image rotation and scale changes.  It so suits SST data after
  derotation and non-rotating DST and IRIS data.  Program
  coalignfitscubes.pro Metcalves every timestep and may be useful for
  sequences needing that.

  Nearly a black box, but its usage and fine-tuning (mucking) remain tricky:
  - produce an "stxalignfile" (for SST usually the standard
    Halpha wide-band cubefile (e.g., wb.6563.hh:mm:ss.corrected.fits)
    or a fitscube for the redmost Halpha wavelength made with crisp2fits.pro)
  - select which SDO target file to use for comparison = "sdoalignfile"
    (for the above Halphas hmicont.fits; aia1700.fits suits  a selected
     red CaII 8542 wavelength but only at disk center)
  - select a sharp moment or rely on the max_rms self-finding
```

```
- perhaps adapt the various image muck parameters for better match
- at problems specify a trimbox subfield to select better-similarity region
- choose underway whether to add cross-alignment correction
  AIA 304 > Halpha_lc (works only when there share filaments in the field)
- reversely, choose to undo 304 > 1700 anchoring of all EUVs (not default)
- adapt splinestx and splineha for spline fit curve smoothness
- choose between pixelated (splinip=0) or interpolated (splinip=1) SDO
I usually run initially with only do_findalign=1, no outputs yet;
when the co-alignment data are in order then I rerun inserting the
best-match parameter values and setting do_findalign=0
and then switching on my choice of the outputs (or all):
- do_fulldisk = get full-disk SDO images with STX field outlined
- do_sdo2stx = put cropped rotated SDO on your STX sequences
- do_stx2rotsdo = put STX on co-cropped STX-rotated SDO
- do_stx2sdo = put STX on co-cropped SDO (SDO or up=up orientation)
- do_stx2fullsdo = put STX within non-cropped full-duration SDO (idem)
See example(s) at the end of sdo_stx_align.pro.

This program is again a wrapper calling other programs:

  findalignimages.pro
    # find co-align parameters from a pair of images (shift,
      rotate, scale) iterating Tom Metcalf's slow but precise
      auto_align_images.pro.  Specify which pair or have it
      selected per highest STX rms (best seeing).

  sdo_stx_findalignfiles.pro
    # uses the one-pair results above as basis to find
      time-dependent shifts between SDO and STX assuming there is no
      temporal variation in their scale and angle differences.  The
      cross-aligned SDO cutouts follow standard solar differential
      rotation whereas e.g., the SST correlation-tracking follows
      some surface feature.  The measured drifts in x and y are
      splined to avoid bad-seeing moments at the STX.  There is also
      a tiny (0.05 arcsec) wobble left from the 1-px sawtooth in
      JSOC's solar-rotation tracking.  This wobble is also splined
      (a faster one around the drift spline).  The drift spline is
      used to correct the STX position, the wobble spline to correct
      the SDO position.  Another small ingredient is the orbital
      sub-SDO offset ("sub" = solar surface intersection point along
      line telescope - sun center) which tracks a small ellipse in
      24 hours with max half axis below 0.3 arcsec.  This diurnal
      pointing variation is ignored, hence retained in stx2sdo and
      stx2rotsdo results.  It uses:

    sdo_shiftaligncubefiles.pro
      # find image-by-image time-dependent shifts and spline approximations
```

```
      reformcubefile.pro
        # general program to re-size, scale, rotate, diffdesolrotate, muck,
          sample a fitscube or "La Palma file" image sequence.
          For most actions it employs:

        reformimage.pro
        # lots of image transformations including border and edge removal.

      doallfiles.pro
        # run a program such as reformcubefile for e.g., all SDO files.

        and it may optionally call:

      sdo_fitscubes2crispex.pro
        # combine co-aligned SDO files into one multi-wavelength crispex file.

      sdo_stx_fulldisk.pro
        # get full-disk SDO image with STX field outlined

  After this cubefile production you can still rescale, shift, rotate,
  unpixelate, etc., one cube with reformcubefile.pro, or all of them
  by using doallfiles.pro calling reformcubefile.pro.

  If you obtain such SDO-aligned data from multiple STX instruments
  you may combine them into common temporal and spatial overlap files
  with a sdo_stx_combine.pro.  I probably will add more such.

  If your STX has varying image scale and angle you may use
  coalignsfitscubes.pro which Metcalves every time step and hence is
  very slow.  It should be expanded with spline fits to its results.


Step 4: inspect and browse the results
--------------------------------------

  Advised browsers:

    crispex.pro
      # by Gregal Vissers, in SSW, for CRISP and IRIS files with
        spectral dimension (and possibly Stokes) folded with time into
        the 3rd assoc dimension.  A very comprehensive and versatile
        browser - but it needs a large screen beyond my laptop.

      examples SDO + SST inspection:
        - set makecrispexsdocube to use sdo_fitscubes2crispex.pro to merge
          all SDO files in into a single crispex-format multi-wav file,
          then:
          IDL> crispex,'sdo2sst/sdo_crispex_for_px-11.bcube',$
```

```
          single_cube=11,$
          refcube=['crispex/crispex.6563.08:15:35.time_corrected.icube',$
         'crispex/crispex.6563.08:15:35.time_corrected.sp.icube']
```

- use sdo_sst_writeblinkfile.pro to merge SDO plus SST files into one (large) file in crispex format setting the (-slow-) spfile option, then:
  IDL> crispex,imfile,spfile

showex.pro (calls movex.pro)

```
# my image/movie browser to show and blink image or image
  sequences given as variables or files.  All must have
  identical dimensions and be cospatial and synchronous.  This
  browser is less comprehensive then crispex.pro (no slicers or
  measurement tools), but it fits small screens as my laptop.
  The underlying engine is movex.pro built on ximovie.pro by
  Wikstol and Hansteen for Solar-B.  It assocs into any number
  of files.  When showex.pro is used on variables (one or more
  2D image or 3D sequence arrays in memory), it rewrites these
  first as /tmp files and then starts movex.pro on those.  It
  does the same with .jpg, .png, .mp4, .jpg, .jpeg, .mov, .ps files.
  At present movex.pro accepts fits files (including from SDO,
  IRIS, DST/IBIS), DOT cube files, and SST "La Palma" format
  image and crispex-format spectral-image sequence files.
  Movex.pro permits easy blinking between slider-set wavelengths
  and also at slider-set time delay.  It maintains ximovie's
  clever new-instance option for subfield zoom-in.  It has
  options /allsdo, /allmwf, /allfits to load all files in given
  directories.  It can plot live spectral profiles, timelines,
  and power spectra for the pixel under the cursor.  It can show
  and blink Dopplergrams with optional blue-red color coding.
  It can show live correlative scatter plots per blink pair,
  then shows where the pixel under the cursor sits in the
  scatter plot, and it can color those image pixels that lie
  within specified ranges in the scatter plot.  It is my
  workhorse inspection tool, even for single images for easy
  zoom-in and measurement.
```

showex usage examples:

- one 2D array image variable: use to zoom in, print pixel coordinates and brightness values, obtain ps or png output
  IDL> showex,image

- two image variables: blink, get scatter plot, locate funny pixels
  IDL> showex,image1,image2,/scatterplot

```
   - four cube variables: play, (time-delay) blink, scatterplot, etc.
       IDL> showex,cube1,cube2,cube3,cube4,/scatterplot,/plotpower

   - one cubefile: time-delay blinking and scatterplot for evolution
       IDL> showex,'cube.fits',/plotscatter

   - two SDO files: play, (time-delay) blink, scatterplot, etc.
       IDL> showex,'sdo2sst/aia171_for_px.fits',$
                   'sdo2sst/aia304_for_px.fits

   - all SDO files in a directory to play, blink, scatterplot any pair:
       IDL> showex,/allsdo,sdodirs='mysdodir'

   - single SST/CRISP file (specification nt_mw mandatory):
       IDL> showex,'crispex.6563.09:48:31.time_corrected.aligned.icube',$
              nt_mw=35

   - all co-aligned SDO files and two specified SST/CRISP files:
       IDL> showex,'sst/crispex.8542.09:48:31.stokesI.icube',$
              'sst/crispex.6563.09:48:31.time_corrected.aligned.icube',$
               sstspecsavs=['sst/spectfile.6563.idlsave',$
                            'sst/spectfile.8542.idlsave'],$
               /allsdo,/plotscatter

   - all co-aligned SDO, IBIS and other DST (white-light) sequences
       IDL> showex,/allsdo,sdodirs='ibis2sdo',/allmwf,mwfdirs='ibis2sdo',$
              /allfits,fitsdirs='ibis2sdo',/addlabels,trimbox=trimboxstx

    I also run showex from the command line with a shell script piping
    its call to IDL:
      #!/bin/csh
      if  ($#argv == 1) then
        echo "showex,'$1'" | idl
      endif
      if  ($#argv == 2) then
        echo "showex,'$1','$2'" | idl
      endif
      ## etcetera for more
      end

   - example: pdf image pages extracted with pdftk from a pdf file
       > showex p5.pdf p6.pdf p9.pdf


xslice.pro
  # by Alfred de Wijn, in SSW, cube slicer for multiple
    image-sequence cubes in memory.  Displays (x,y) images with
```

```
        simultaneous (x,t) and (y,t) time slices, with co-moving
        cursor crosses for precise co-localization.

       example:
         IDL> xslice,gb,root=root,mag=2,ytslice=0
         IDL> xslice,ca,root=root


================================================================================
        Done!  The rest of this manual offers loose titbits
================================================================================


miscellaneous image or image sequence processing programs in my libraries
--------------------------------------------------------------------------------
  sdo_diskfigs.pro
    # for any SDO-era minute collect full-disk SDO images, optionally
      also GONG Halpha, make image figures and tilechart cross-alignment
      figures
  diffdesolrotate.pro
    # undo differential solar rotation across a large field to make
      scenes appear unstretched
  deforepix.pro
    # destretch foreshortened limbward pixels to rectangular "view from above"
  sdo_viewlevel2.pro
    # showex SDO level2 files for given wavelength, plot cumulative shifts
  sdo_printtranges.pro
    # print t_obs in first and last SDO level1 or level2 fits files
  sdo_checkcadence.pro
    # check cadence regularity of JSOC cutouts with request segmentation
  sdo_stx_combine_v1.pro
    # combine data sets SDO, DST/IRIS, DST/MXIS
  sdo_stx_combine_v2.pro
    # combine data sets SDO, SST/CRISP, SST/CHROMIS
  sdo_muckimagepair.pro
    # muck images at two SDO wavelengths to make them look alike
  findimshift_rr.pro
    # my reworking of a Pit Suetterlin Fourier cross-correlation program
  findimshift_tiled.pro
    # tesselate images into tiles, cross-correlate all to find shift
  findalignimages.pro
    # find shift, scale, rotate to align two images; slow but precise
  selfalignfitscube.pro
    # straighten a fitscube (shift only of full Metcalf per image)
  coalignfitscubes.pro
    # coalign two fitscubes (shift only or full Metcalf per image pair)
  reformimage.pro
    # change many properties: orientation, size, shift, intensity scaling, ...
```

```
reformcubefile.pro
  # same for image seqence, plus sequence timing, scaling, ...
shiftfitscube.pro
  # apply given shifts only
rotatefitscube.pro & cube_rotate.pro
  # rotate a (fitsfile) cube, including time-dependent rotation
solrotcomp.pro
  # compute (X,Y) shifts following differential rotation for given time span
concatfitscubes.pro
  # concatenate successive fitscube files
makefitsflowcubes
  # determine flow, divergence, vorticity fields by correlation tracking
makefitsflowarrowcube
  # make cube showing little arrows outlining flow patterns
addfitscube2crispex
  # add another fits cube to a multi-wavelength crispex file
fitscube2mpeg.pro
  # make presentation movie with clock, datetime
make4panelmovie.pro
  # make 4-panel presentation movie with clock, datetime, sundisk, location
sdo_maketargetmovie.pro
  # make 4-panel movie: hmimag, aia1600, aia304, aia171
sdo_ebfafmovie.pro
  # make 4-panel movie for EB/FAF distinction (first run sdo_addebfafs.pro)
movex_loadfiles.pro
  # multi-file assoccer for movex.pro.  Extendable to other formats
movex_scatcont.pro
  # make scatcont diagrams integrating any duration for any movex pair
crisp2fits.pro
  # make a fits file for one wavelength in a CRISP La Palma file
lp2fits.pro
  # make a fits file from a La Palma SST cubefile
lp_im2spfile
  # reorder a La Palma crispex-format file into an .sp. file for crispex
readfitscube
  # read fitscube file with x,y,t range options
readcrispfile.pro
  # get part of a CRISP file into a variable
readsstfile.pro
  # get part of a La Palma SST cubefile into a variable
mwseq2special.pro
  # get special spectral profile measures with prof2special.pro
sdo_makenosyncfitscubes
  # make same-size SDO fitscubes without interpolation to the refwav timing
gong_sdo.pro
  # select best GONG/Halpha images and (try to) coalign to AIA 304
alma_gongsdo,pro
  # put ALMA on SDO using SDO-aligned GONG alpha as intermediary
```

```
SDO cube loaders for cut-and-paste in IDL session (small files only)
-----------------------------------------------------------------------
  a94=readfits('cubes/aia94.fits',head94)     ; 2 peaks log(T) = 6, 7
  a131=readfits('cubes/aia131.fits',head131)  ; 2 peaks 5.9, 7.1
  a193=readfits('cubes/aia193.fits',head193)  ; wide 5-8, peak 6.2
  a211=readfits('cubes/aia211.fits',head221)  ; wide tail hump 5.5, peak 6.2
  a171=readfits('cubes/aia171.fits',head171)  ; 1 peak 6.0
  a335=readfits('cubes/aia335.fits',head335)  ; wide box 5 - 6.8
  a304=readfits('cubes/aia304.fits',head304)  ; peak 4.9  all EUV cadence 12s
  a1600=readfits('cubes/aia1600.fits',head1600)  ; as 1700 + C IV cadence 24s
  a1700=readfits('cubes/aia1700.fits',head1700)  ; plm 350 km up  cadence 24s
  cont=readfits('cubes/hmicont.fits',headcont) ; original HMI cadence 45s
  mag=readfits('cubes/hmimag.fits',headmag)    ; units 0.1 Gauss
  dop=readfits('cubes/hmidop.fits',headdop)    ; units 0.1 m/s, black=blue

example rescaled SDO cube comparison
------------------------------------
showex,cont,histo_opt_rr(a1700,0.005),mag>(-7000)<7000


Version history (used to be reversedly on top, now here in time order)
===============
December 2013: start.

April 2016: Two way SDO-SST version.  Optionally produces a
  CRISPEX-readable [x,y,wav*time] cube to let CRISPEX loose on all SDO
  cutouts together with SST CRISP data (example under step 4 below).

April 2017: step 2.  Improved AIA/EUV cross-alignments.  They now
  avoid unsolved errors for cutout treatment in SSW's aia_prep.pro by
  using "best" current scale and rotate values from the JSOC/MPO
  master tables while setting /use_hdr_pnt in aia_prep.pro for the
  shifts aren't used anynow.  They then find and apply shift
  corrections including time-dependent drifts from pair-wise
  cross-correlations of synchronous low-cadence large disk-center
  cutouts using elaborate tricks including image mucking to better
  alikeness and image tiling into 30x30 arcsec subfields.  The
  resulting EUV cross-alignments usually reach about 0.1 arcsec
  precision.  The hardest AIA alignment step is 304 > 1700 which
  necessarily serves as anchor to put all EUVs on UV and HMI.  The
  pipeline then applies these cross-alignment shifts at the faster
  desired cadence to the desired target field of view, which is
  usually small and away from disk center (where cross-correlations do
  not work for features at different heights).  Altogether step 2 is
  now a robust black box: a single IDL command including data ordering
  and getting (sdo_getdata_rr.pro).
```

June 2017: step 3 automated for SST; also, it now works both ways. It includes optional correction of all AIA EUV channels by co-aligning AIA-304 to SST-Halpha (but this works only for scenes with much activity) and reversely optional undoing of the anchoring of all EUV channels to the hard-to-measure shifts 304 > 1700. Both options should be decided by inspection of the corresponding spline-fit co-alignment graphs. This is now also a single command (sdo_sst_align.pro) and largely a black box. Step 1: improved to high precision (sst_findlocation.pro). Step 4: versatile multi-sequence player/blinker movex.pro.

Autumn 2017: step 4 = file browser movex.pro extended with more features and wrapper showex.pro accepting variables (image or cube arrays in memory) as well as files.

December 2017: step 3 revised. Pipeline sdo_sst_align.pro and underlying now accept a non-fits CRISP prefilter .wb. file as sstfile and auto-detect and ignore blank borders of SST images, including time-varying triangular derotation corner edges.

January 2018: step 3 generalized and tested on DST/IBIS data in addition to SST/CRISP. It required restoring treatment of pixel anisotropy. The pipeline is now called sdo_stx_align.pro.

March 2018: Also co-align IRIS/SLI data with SDO together with SST or DST data and then browse them all together with showex.pro.

July 2018: sdo_getdata_rr.pro automatically segments JSOC ordering to enable long-duration high-cadence studies circumventing JSOC number-of-files-per-request limits.

December 2018: step 3 = sdo_stx_align.pro extended with output choices:
- do_fulldisk: collect one set of full-disk SDO images, outline the STX field, optionally rotate to up=up;
- do_stx2rotsdo: put STX sequences on STX-cropped and STX-rotated SDO (STX orientation, SDO tracking, regularized STX timing);
- do_stx2sdo: put STX on STX-cropped SDO at SDO timing;
- do_stx2fullsdo: put STX at SDO resolution within full SDO cutouts.

The last three are now the advised output modes because they stably follow standard solar rotation. Do_stx2rotsdo minimizes de-rotation triangles; the other two either have solar (X,Y) orientation or the direction to disk center downwards ("up=up").
Outputs do_fulldisk and do_stx2fullsdo show surrounding SDO context.

June 2019: sdo_featurelocator.pro and gong_featurelocator.pro extended with option 'latest' for current images (use in targeting).

January 2020: SDO-STX co-alignment now splits measured time-dependent offsets into STX guiding trends and the small (well subarcsec) SDO wobble left from JSOC 1-px cutout solar-rotation quantization, correcting each separately in stx2rotsdo output mode.

February 2020: JSOC now forbids synchronous rapid-fire ordering; I added waiting per order on still active previous ones. SDO cross-alignments checked on 2019-11-11 Mercury transit: good.

March 2020: hands-on demo practical of this software at
https://webspace.science.uu.nl/~rutte101/rrweb/sdo-demo
providing convenient material to try this pipeline.

April 2020: JSOC option notrack can give bad target/level1 shifts for 304 and 94, default refwav changed to 171.

May 2020: dir fixsswlib for overriding SSW problems. Tuned timings. Switched EUV anchor from 304>1600 to 304>1700.

July 2020: EUV anchor now 304>mag. Added formation-height differences in SDO cross-alignment. Added diffdesolrot option to differentially undo solar rotation across large fields up to full-disk, suited also for correlation-tracked imagery. Added deforepix.pro to unshrink foreshortened limbward pixels. Added sdo_diskfigs.pro collecting full-disk SDO images, GONG Halpha too, and producing image and tilechart offset figures.

August 2020: 2020arXiv200900376R = LAR-1 = Lingezicht Astrophysics Reports 1 shows tilecharts and driftscenter plots.

September 2020: SDO problems. Major (for you?): aia_prep.pro aborts for some users but not for me nor Greg Slater. Minor (for me): JSOC required registration renewal; aia_prep.pro gives many harmless error messages ("v9 versus v8 file missing"). (Later gone)

November 2020: option addfires: adds 304x131 "fire" fitscubes in the wake of Solar Orbiter "campfires" following LAR-1 (see its fig 69).

December 2020: sdo_addebfafs.pro with choice ratio or multiply 1600 and 1700, and 4panel wrap movie in sdo_ebfafmovie.pro.

January 2021:
- gong_sdo.pro: selects the best of a sequence of GONG Halpha images and coaligns these precisely with SDO/AIA 304 following success in LAR-1. Tricky because the GONG image quality varies from bad to lousy. Quiet-Sun trimbox selection avoiding activity and filaments is crucial. I wrote selfalignfitscube.pro and

coalignfitscubes.pro for it, both are tricky and slow. I am still improving this, remaining offsets occur that I have trouble with.
- alma_gongsdo.pro: puts an ALMA quiet-Sun image sequence precisely on GONG Halpha and if that was aligned to SDO then on to SDO.