Release Notes for 11.0 (Summer 2015)

LSST Stack Summer 2015 Release

The Summer 2015 release of the LSST stack is internally numbered "11.0" (git tag) or v11_0 (EUPS distrib).

These release notes document notable changes since 10.1, which was the Winter 2015 release.

Major Functionality and Interface Changes

Improved semantics for loading `Exposures` and `MaskedImages` from arbitrary FITS files

The `Exposure` and `MaskedImage` represent image data with associated mask and variance information. When serialized to FITS, these are stored as three consecutive extensions in the FITS files. It is possible to load `Exposures` and `MaskedImages` from multi-extension FITS files which were not generated by LSST, but, due to the limitations of the FITS data model, it is not possible to ensure that the creator of the file adhered to the LSST convention: while an image object may be successfully instantiated, its contents may not be logically consistent.

We now go to greater lengths to check that the information in the file is consistent with the LSST standard, warning the user – and in some cases refusing to proceed – if it does not. (DM-2599)

Improved support for non-standard FITS headers

The LSST stack is now capable of loading FITS files which contain non-standard headers of the form `PVi_nn` (`i=1..x`, `nn=5..16`), as written by SCAMP, and `EQUINOX` headers with a "`J`" prefix, as written by SkyMapper. (DM-2883, DM-2924, DM-3196)

It is now possible to perform instrument signal removal on an `Exposure` which has no `Detector`

`FakeAmp`, a `Detector`-like object object which supports returning gain and saturation level, was added to make it possible to run `updateVariance` and `saturationDetection` if required. (DM-2890)

`PVi_j` header cards are correctly saved to FITS files

This makes it possible to round-trip e.g. TPV headers. (DM-2926)

Changes to compound fields and delimiters in Catalog Schemas.

In the older ("version 0") approach to table schemas, we had several compound field types (`Point`, `Moments`, `Covariance`, `Coord`) which behaved differently from other field types - the square bracket `[]` operators could not be used to access them, and they could not be accessed as columns (though their scalar subfields – e.g. "x" and "y" for `Point` – could be). In version 0, we used periods to separate both words and namespace elements in field names, but converted periods to underscores and back when writing to FITS. These schemas were mostly produced by the old measurement framework in `meas_algorithms`' `SourceMeasurementTask`, which was removed in the 10.1 release..

In the new ("version 1") approach, compound objects are simply stored in catalogs as their constituent scalars, with helper classes called `FunctorKeys` provided to pack and unpack them from `Records` (the `FunctorKeys` that replace the old compound fields are all in `afw/table/aggregates.h`). Unlike the original compound fields, there's no limit to how many types of `FunctorKey` we can have, or what package they can live in, making the system much more extensible. By making the constituent scalar objects what the `Schema` object knows about, it will be much easier to map a `Schema` to other table representations that don't know about LSST classes (e.g. SQL or Pandas). Most `FunctorKeys` can be used anywhere a regular `Key` can be used. Also, in version 1, we use underscores as namespace separators, and CamelCase to separate words, eliminating some ambiguity between word and namespace boundaries. The new measurement framework in `meas_base`'s `SingleFrameMeasurementTask` and `ForcedMeasurementTask` uses version 1 tables exclusively.

In previous releases of the pipeline, version 0 schemas were deprecated but still supported. They have now been removed, but old catalogs saved as version 0 will still be readable - they will be converted to version 1 on read, with period delimiters converted to underscores, and all compound fields unpacked into scalar fields that can be used with a corresponding `FunctorKey`. This procedure obviously does not preserve field names, but all slot definitions will be preserved, so code that only relies on slot or minimal schema accessors (`getCoord()`, `getCentroid(`, `getPsfFlux()`, etc.) should not need to be modified. (DM-1766)

Allow for use of Approximate (Chebyshev) model in background estimation

In previous releases, the only method for background estimation was to use an interpolation scheme (constant, linear, or various splines). These schemes tend to lead to over-subtraction of the background near bright objects. The Approximate (Chebyshev) approach to background estimation greatly improves the background subtraction around bright objects. The relevant code to use this latter approach (including persistence and backwards compatibility issues) is now in place.

While the intention is to eventually set the Approximate background subtraction scheme as the default, there is some clean-up and restructuring that needs to be done before resetting the defaults (which may also require adjusting some defaults in the calibrate stage to be more appropriate for the approximation, as opposed to interpolation, scheme). Therefore, the default setting has not been changed (i.e. the default is still to use an interpolation scheme for background estimation). The Chebychev approximation can be selected for background estimation through configuration parameters in the obs_CAMERA packages, i.e. useApprox=True and, optionally, approxOrderX (approximation order in X for background Chebyshev), approxOrderX (approximation order in Y for background Chebyshev: currently approxOrderY must be equal to approxOrderX), weighting (if True, use inverse variance weighting in calculation). (DM-2778)

Multi-band processing for coadds

See the description of the multi-band coadd processing work performed in S15 for details. In short, four new command-line Tasks have been added for consistent multi-band coadd processing:

DetectCoaddSourcesTask
Detect sources (generate Footprints for parent sources) and model background for a single band.
MergeDetectionsTask
Merge Footprints and Peaks from all detection images into a single, consistent set of Footprints and Peaks.
MeasureMergedCoaddSourcesTask
Deblend and measure on per-band coadds, starting from consistent Footprints and Peaks for parent objects.
MergeMeasurementsTask
Combine separate measurements from different bands into a catalog suitable for driving forced photometry. Essentially, it must have a centroid, shape, and CModel fit for all objects, even for objects that were not detected on the canonical band. Will assume that all input catalogs already have consistent object lists.

(DM-1945, DM-3139)

Enable use of deblended HeavyFootprints in coadd forced photometry

Given the new multi-band processing for coadds (above), we now have a reference catalog that is consistent across all bands. This catalog allows the use of the source's HeavyFootprints to replace neighbors with noise in forced photometry, thus providing deblended forced photometry and consistent deblending across all bands. This provides much better colors for blended objects as well as measurements for drop-out objects that do not get detected in the canonical band. This functionality has been enabled for forced coadd photometry.

See the description of the multi-band coadd processing work performed in S15 for further motivation of this change. (DM-1954)

Limited the fractional number of masked pixels per source

CModel has difficulties modelling backgrounds in vignetted regions of the focal plane, leading to a performance bottleneck. To mitigate the issue, if the fractional number of masked pixels in a particular source exceeds a given threshold, that source will be skipped. (DM-2914)

Peak culling around large objects

An excess of "junk" peaks may be observed around large objects. Given the new multi-band processing architecture (above), these must be consistently removed across bands. We therefore provide a method to consistently "cull" this peaks at an earlier stage, immediately after merging and sorting in `MergeDetectionsTask`. (DM-2914)

Parent Footprints are the union of their children

Parent `Footprints` are now trimmed so that they are strictly the union of their children: any pixels which are not assigned to a child are removed. This mitigates an issue whereby stray flux from the parent was not correctly assigned to the children. Note that this has the consequence that parent `Footprints` are not necessarily contiguous. (DM-2914)

Large Footprints may be skipped on initial processing

For practical processing purposes (specifically total processing time and memory limits due to current hardware limitations), we have the option to skip over objects with large `Footprints` during large-scale processing, with the intention to return to these objects to "reprocess" them using different hardware in future. The ability to properly record which objects have been skipped and require further processing has been implemented along with optimizations to the deblender configuration for the maximum number of `Peaks` per `Footprint`, and the size and area of `Footprint`s. (DM-2914)

Command line tasks for measurement transformation

The measurement transformation framework provides a generic mechanism for transforming the outputs of measurement plugins in raw units, such as pixel positions or flux, to calibrated, physical units, such as celestial coordinates or magnitudes. Appropriate transformations are defined on a per-measurement-plugin basis, and may make use of the calibration information and WCS stored with the data.

This system is designed such that the transformation of a given catalog is performed by a command line task. Different catalog types (such as `src`, `forced_src`, etc) make use of separate command line tasks. In this release, we provide a variety of tasks to handle different source types.

- Documentation for generic transforms
- Documentation for `SrcTransformTask`
- Documentation for `ForcedSrcTransformTask`
- Documentation for `CoaddSrcTransformTask`

(DM-2191, DM-3473, DM-3483)

Add `NO_DATA` mask plane

Previously, we have used the `EDGE` mask plane to indicate *both* pixels which are off-the-edge of the detector, and hence have no data available, and pixels near the edge which cannot therefore be properly searched for sources. Here, we introduce the `NO_DATA` plane to refer to the former case and now use `EDGE` strictly for the latter. (DM-3136)

Add slot for flux used in photometric calibration

We define a new slot, `CalibFlux`, on `SourceRecord`s. This slot is used to record the flux used for photometric calibration, rather than hard-coding the name of a particular algorithm in the `PhotoCal` task. This slot defaults to a 12 pixel circular aperture flux, the previous default in `PhotoCal`. (DM-3106, DM-3108)

Table field prefix for aperture flux measurements changed

Our aperture flux measurement algorithms take a list of radii, in pixels, which define the radii over which measurements should be made. Previously, the names of the table fields produced by the algorithm were defined purely based on the position of the radius in that list (thus, the first radius listed would produce a flux field named `PluginName_0_flux`). This has been changed so that the fields are now named after the radius, regardless of its position in the list. Thus, a 12.5 pixel aperture will reult in a field named `PluginName_12_5_flux`, regardless of its position in the list. (DM-3108)

Faster astrometry reference catalog loading

The reference catalog loading was optimised by caching HEALpix identifiers for the catalog files. This has been observed to speed up loading times from 144 sec to 12 sec.

The cache is saved as `andCache.fits` in the astrometry catalog directory. The use of the cache can be disabled through the `andConfig.py` file (or the `AstrometryNetDataConfig`) by setting `allowCache` to `False`. To prepare a cache, `setup astrometry_net_data` and use the `generateANetCache.py` script that now comes in `meas_astrom`. (DM-3142)

Bad pixels tracked when coadding images

When co-adding images, we now keep track of what fraction of the input data for a given pixel was masked. If the total masked data exceeds some user-configurable threshold, the mask is propagated to the coadd. (DM-3137)

Polygon masking in coadded PSFs

Polygonal masks are used to define the usable area of the focal plane; they can be used to, for example, exclude vignetted areas from coaddition. We now take account of these masks to determine which PSF images to included when building co-added PSFs. (DM-3243, DM-3528)

Scale counts to reflect CCD-specific zero-points when warping to create coadd inputs

(DM-2980)

Solving astrometry with distortions

The default astrometry matcher (`matchOptimisticB`) can now match stars against a reference catalog when the stars are distorted (e.g., at the outskirts of a wide field imager) if there is an estimate of the distortion available. (DM-3492)

Rejection iterations in astrometry fitting

Astrometric fitting (`FitTanSipWcsTask`) now includes support for iterative fitting with rejection. (DM-3492)

Inclusion of external package PSF Extractor as option for PSF determination

PSFEx is currently the state of the art external package for PSF determination, used in projects such as DES. LSST wrappers were created such that PSFEx could be used as a plugin in place of the built in PSF determiner. Tests with Hyper Supreme Camera data have shown that PSFEx out performs the built in PSF determiner. (DM-2961)

Improvements to CModel magnitude measurement

This release includes many miscellaneous improvements and fixes resulting from testing on HSC data, including:

- parameter tuning for computational performance improvement
- correction to uncertainty estimation to account for extrapolation beyond the fit region
- much more robust flagging of failure modes

Interface changes to forced measurement

The order of arguments to the forced measurement task was reversed, so that it takes a source catalog followed by an `Exposure`. This brings it into line with the single frame measurement interface. (DM-3459)

N-way spatial matching

A simple utility class for performing spatial matches between multiple catalogs with identical has been added as `lsst.afw.table.multiMatch.MultiMatch`. This is intended as a stop-gap measure until more flexible and efficient functionality becomes available, but is already usable. (DM-3490)

Display CCD data as laid out in the focal plane

It is now possible to use `lsst.afw.cameraGeom.utils` to display CCD data laid out in the focal plane. An example of how this functionality works in practice is available as an IPython notebook. (DM-2347)

Bug Fixes

The following fixes resolve problems visible to end users.

Doxygen documentation now correctly includes LaTeX formatting

Correctly referring to MathJax means that LaTeX markup in documentation is nicely formatted. (DM-2545)

Performance regression in `Footprint` dilation resolved

The previous release included improved algorithms for dilating `Footprints`. Unfortunately, in some circumstances (notably when dealing with particularly large `Footprints`) this code could actually perform more slowly than the previous implementation. This could have significant performance implications for many image processing operations. This regression has now been rectified, and the new dilation operations are significantly faster than the old ones in all circumstances tested. (DM-2787)

Footprint fixes

The following updates/fixes to Footprint handling have been made:

- The default 32-bit heap space used to store FITS variable-length arrays isn't large enough to store some of our extremely large HeavyFootprints. This persistence issue has been fixed the by switching to 64-bit heap descriptors, which is now supported by FITS.
- Footprint::transform is now properly copying peaks over to the new footprint.
- Footprint::clipTo is now properly removing those peaks lying outside the desired region.
- Several parts of the pipeline assume peaks are sorted from most positive to most negative. We now ensure the cross-band merge code maintains this ordering as much as possible (even though the sorting may not be consistent across different bands).
- The merging of a parent and its children's Footprints was failing in cases where one or more child Footprints were themselves noncontiguous. This has been fixed by adapting the mergeFootprints code in afw such that it combines all the Footprints in the FootprintSet it uses in its implementation (instead of requiring that the FootprintSet have only one Footprint).

(DM-2606)

Fixed error in memory access in interpolation

An off-by-one error resulted in an attempt to read beyond the allocated memory. (DM-3112)

Fixed truncated write of certain WCS information to FITS

(DM-2931)

Use the correct weighting in photometric calibration

Previously, we were incorrectly weighting by errors, rather than inverse errors. (DM-2423)

Remove non-positive variance pixels in coadd creation

When interpolating variance maps we can produce negative values. These then cause failures when we try to take the square root. Ultimately, the means of creating variance maps needs to be fixed (which is DM-3201); as a temporary workaround, we replace negative variance values with infinity. (DM-2980)

Task defaults are set correctly for difference imaging

The `DipoleMeasurementConfig.setDefaults` method incorrectly contained a `return` that was executed before the defaults were actually applied. This has been corrected, and a number of tests updated to rely on those defaults. (DM-3159)

Build and code improvements

These improvements should not usually be visible to end users. They may be important for developers, however.

Backend-agnostic interface to displays

The image display code no longer makes the assumption that display is carried out using ds9. Rather, an API is available which is independent of the the particular image viewer is in use. A backwards compatibility layer ensures that display through ds9 is still supported, while other backends will be added in future.

(RFC-42, DM-2709, DM-2849, DM-2940, DM-3203, DM-3468)

Measurement framework compiler warnings resolved

The measurement framework was refactored to avoid a series of warnings produced by the clang compiler. (DM-2131)

Unsanctioned access to the display by tests suppressed

Some unit tests were attempting to write to a display, even when no display was available. On some systems, this directly caused test failures; on others, it could obscure the true cause of failures when a test did fail. (DM-2492, DM-2494)

Unused & obsolete code has been removed from the `datarel` package

This package is effectively obsolete, but is still used in documentation generation which makes removing it entirely complex. For now, therefore, it has simply been trimmed of all unused functionality; it may be removed entirely following . (DM-2949)

Reduced verbosity of astrometry.net solver

A correction to the way that astrometry.net logging was propagated to the LSST logging system, together with reducing the priority of some messages, leads to a substantial reduction in chatter from astrometry. (DM-3141)

Ensure that slots are present before initializing algorithms that depend upon them

When initializing an algorithm that refers to a particular slot, we resolve the target of the slot and refer to that instead. That means that if the slot definition is changed after measurement has been performed, we are still pointing to the correct information. However, if the algorithm is initialized before the slot it depends on, this resolution could not take place and "circular" aliases could result. We now explicitly check for and throw an error in this case. (DM-3400)

Visualizations for astrometry.net solver

It is now possible to display the source positions, distorted source positions and reference positions to assist with debugging. (DM-3209)

Subaru support reinstated

The `obs_subaru` package, which provides packages and tasks specific to the Subaru telescope, has been brought up to date with recent changes to the LSST stack and improvements made during Hyper Suprime Cam development. (DM-1794, DM-3403)

Refactor & document coadd construction

A number of minor changes and documentation improvements were made to the `CoaddBase`, `AssembleCoadd`, `CoaddInputRecorder` and `MakeCoaddTempExp` tasks. These brought the structure of the code better into line with the state-of-the-art development on Hyper Suprime Cam. (DM-2980)

Properly handle masking NaN or saturated values in overscans

Resolved an issue where, in certain circumstances, flags in the mask plane for saturated and nan values in overscans were being improperly propagated to all amplifiers in an image. These flags are now applied to the amplifier where the bad values are seen. (DM-2923)

Deblender optimization

Several performance optimizations to the (C++) algorithms used in the deblender have been implemented, in particular those which identify objects with significant amounts of their flux attributed to edge pixels. In addition, memory usage was reduced by removing unused mask planes left over from debugging, not storing masks for deblending templates, and by clipping template images when their associated `Footprints` are clipped. (DM-2914)

Release notes for Qserv and Webserv

These release notes focus on the Science Pipelines part of the LSST Stack, but there are other important components developed by Data Management teams.

- Summer 2015 release notes for Qserv
- Summer 2015 release notes for Webserv
- Summer 2015 release notes for SUI (Science User Interface)

Known Issues

An up-to-date list of known issues for 11.0 affecting users is available.

Measurements & Characterisation

A summary of Stack characterisation measurements is available.