Introduction To DM Data Concepts

Simon Krughoff – Bootcamp SLAC November 2018

Butler

Key concepts

- Input/Output abstraction layer
- Data are retrieved from a “data repository” using
  - Dataset name: “raw”
  - Data id: {“visit”:1234, “detector”:0}

```
> from lsst.daf.persistence import Butler
> dataId = {‘visit’:1234, ‘detector’:0}
> butler = Butler(‘/path/to/data/repository’)
> raw = butler.get(‘raw’, **dataId)
> butler.put(‘src’, some_catalog, **dataId)
```

Exposure

Key concepts

- Holder for image information and metadata
- Contains (partial list):
  - Key/Value pair metadata
  - Models for PSF and background
  - A masked image
  - Astrometric and photometric solutions

```
> calexp = butler.get('calexp', **dataId)
> md = calexp.getMetadata()
> psf = calexp.getPsf()
> wcs = calexp.getWcs()
> masked_image = calexp.getMaskedImage()
> calib = calexp.getCalib()
```

MaskedImage

Key concepts

- Pixel data information
  - Image – Flux values in a pixel grid in counts
  - Mask – Bit packed mask values per pixel
  - Variance – The per pixel variance as computed by ISR (covariance not included at the moment)

```
> im = masked_image.getImage()
> mask = masked_image.getMask()
> var = masked_image.getVariance()
> im_arr, mask_arr, var_arr = masked_image.getArrays()
```

Image

Key concepts

- Single plane pixel level information
  - Values in counts
  - Stores XY0 offset from the parent pixel coordinate system

```
> im = masked_image.getImage()
> value = im[10,40]
> im[45:50,15:26] = 15
> size = im.getDimensions()
> offset = im.getXY0()
> arr = im.getArray()
```

Bounding Boxes

Key concepts

- Rectangular regions in integer or floating point space
- When in integer coordinates, can be used to subset images

```
> from lsst.geom import Point2I, Extent2I, Box2I
> point = Point2I(10, 10)
> ext = Extent2I(31, 54)
> box = Box2I(point, ext)
> sub_im = masked_image[box] # use .clone() to copy instead of a view
```

Catalogs

Key concepts

- Row based record storage
- Fixed schema per run
- Generic access through schema keys
- Some special values: e.g. ra, dec have getters
- Support casting as astropy.tables

```
> src = butler.get(‘src’, **dataId) # catalogs are iterable
> src.schema
> src.get(‘column_name’)
> coords = src.getCoord()
> table = src.asAstropy()
```

Image math

Key concepts

- Math done on images should respect the variance and mask planes
    - Pixels with certain mask bits set will be left out of calculations
    - Variance will be updated appropriately under image arithmetic
- Statistics can also be calculated on images
- It is also possible to do math on views of images represented as numpy.arrays, but not these will not take into account masks or variances.

```
> import lsst.afw.math as afwMath
> import numpy
> mi1 = masked_image.clone()
> mi2 = masked_image.clone()
> mi1 += mi2
> mi1 -= mi2
> mi1 /= mi2
> mi1 *= mi2
# note mi1 = mi1 + mi2 is not implemented
# to avoid making copies without user's
# knowledge
> stats = afwMath.makeStatistics(mi1, afwMath.MEAN | afwMath.MEDIAN)
> mean = stats.getValue(afwMath.MEAN)
> arr = masked_image.getImage().getArray()
> mean = numpy.mean(arr)
```

Tasks

Key concepts

- A task is algorithmic code with associated configuration
- Configuration can include sub-tasks: e.g. what algorithm to use for PSF determination
- Command line tasks have argument parsing as well as configuration.
- Note that the task system is under revision and will be replaced in the next year.

```
$> runIsr.py repository_path –id visit=1234 –rerun \
> my_special_rerun --configfile config.py
# Now in python
> from lsst.ip.isr import IsrTask
> config = IsrTask.ConfigClass()
> help(config)
> config.doDark = False
> task = IsrTask(config=config)
> result = task.run(**kwargs) # these are specific to the task
```