Draft version October 18, 2018
Typeset using LaTeX **preprint** style in AASTeX62

Lab 1: Photon Counting & Detectors

Jessica Birky, Julian Beaz-Gonzalez, Russell Van-Linge

ABSTRACT

In this lab, we examine the statistics of photon counting, and the noise properties of a charged couple device (CCD) using 2048x2048 pixel optical images taken on the Nickel 1 meter direct imaging camera. In particular we characterize the effect of several types of noise: first the inherent noise of photon counting by comparing to the Poisson distribution, second the read noise, and lastly we comment the effect of different systematics such as dark current due to temperature variations or detector deffects. In total we collect XX frames integrated at varying exposure times ranging from 0 to 768 seconds. Comparing the slope between mean and variance of each bias-subtracted, combined frame of each exposure, we determine the gain of the detector to be XX. Normalizing our frames (dividing the counts by exposure time) we find that as we add more exposures, the mean of the mean (MOM) detector counts per second converges to XX ADU/s and the standard deviation of the mean (SDOM) decreases to XX ADU/s with more frames added.

1. INTRODUCTION

In order to understand the. Advancements in technology in the 20th century has now made it possible to observe electromagnetic radiation from the scales of $10^{-12} - 10^{3}$m (as short as gamma rays to as long as radio waves). Understanding how detectors work, the statistics of light collection, as well as the limitations of the instrument and sources of contaminations are a fundamental first steps before performing any scientific analysis of light with an instrument.

CCDs, the photoelectric effect

Types of noise we want to quantify

2. OBSERVATIONS

Nickel telescope, what are bias and flat frames

3. DATA REDUCTION & METHODS

3.1. *Theoretical Distribution of Light*

Before we can characterize our observations, it is important to first determine what we expect our data will look like. That is, we are measuring the number of photons hitting a detector over a fixed amount of time, what do we predict will be the number of photons that will hit each pixel in that tme interval? Let's assume that on average μ photons will hit a pixel over fixed time interval t. Now

Corresponding author: Jessica Birky (A13002163)
jbirky@ucsd.edu

Start File #	End File #	# Frames	Type	Exposure Time (sec)
401	404	3	Bias	0
419	422	3	Flat	3
423	425	3	Flat	6
426	428	3	Flat	12
429	431	3	Flat	24
432	434	3	Flat	48
435	437	3	Flat	96
438	439	3	Flat	192
440	–	1	Flat	384
441	–	1	Flat	768

Table 1. Observation log, recording the files associated with each exposure time. Files $405 - 418$ were removed due to inconsistent count numbers.

consider dividing that time interval into n increments: the probability that a n photons arrive to the detector is approximately Binomial. Now in the limit that $n \to \infty$, we arrive at the Poisson distribution:

$$P(x;\mu) = \frac{\mu^x}{x!}e^{-\mu} \tag{1}$$

which we interpret as the probability that x photons will arrive to a detector pixel in time interval t, given that μ photons arrive in that interval on average. The expected mean and variation for this distribution being $\langle x \rangle = \mu$ and $\langle \sigma^2 \rangle = \mu$. Under the conditions that the mean of the data is much smaller than the number of observations ($\mu << n$), and

In practice, the Poisson distribution is unstable to compute for large values of x because the term $x!$ becomes too large for the memory of the computer to handle. In order to try to reduce the memory load for large values of x, we approximate the $x!$ term using the Sterling approximation:

$$n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \tag{2}$$

the code for the Poisson distribution is implemented in lines $7 - 9$ of Appendix 8.2.

However when the conditions for the Poisson distribution are not met very well, the Gaussian distribution may instead be a good approximation:

$$P(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3}$$

where the input parameters are the mean and standard deviation of the data (μ and σ). The code for the Gaussian distribution is implemented in lines $11 - 13$ of Appendix 8.2.

3.2. *Data Manipulation*

Procedure to combine frames and subtract bias, point to code

3.3. *Types of Detector Noise*

How to calculate the readnoise, other noise, error propagation

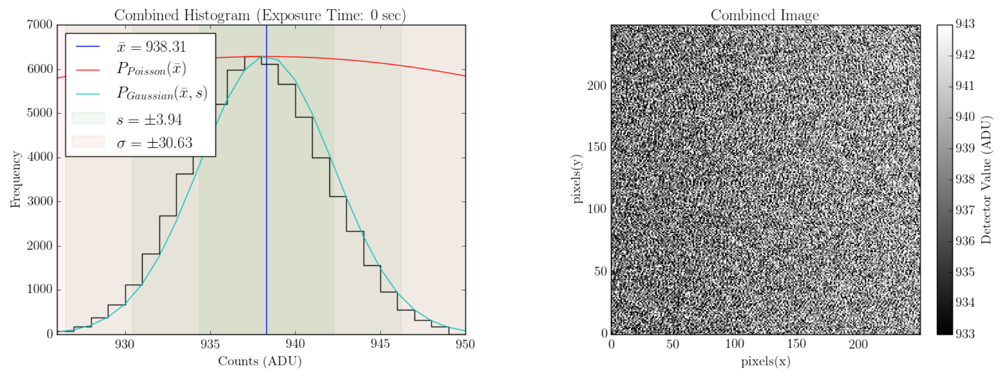

Figure 1. Combined bias frames.

4. DATA ANALYSIS & MODELING

4.1. *Bias Frames*

4.2. *Combined Images & Histograms*

The figures in 2 and 3 show the bias-subtracted, combined histograms and images for each of the eight different exposure times (3, 6, 12, 24, 48, 96, 192, 384, and 768 seconds). The left side of each panel shows the histogram of the number of detector counts in analog to digital units (ADU), normalized such that the sum of all of the bins is one (plotted in black). The solid blue vertical line marks the mean of the data, and the dashed blue line marks the median, indicating which distrubtions are skewed by outlying points (where the mean is far from the median): in the combined bias frame the mean is skewed higher than the center of the distribution, and for all of the other exposures the mean is skewed lower than the center. The two shades of green mark ± 1 and ± 2 standard deviations above and below the mean. For all distributions, the $\pm 1\sigma$ lines lie far outside of the bulk of the distribution around the median, which indicates again that there are many outlying the part of the distribution shown. Finally, the distribution in red shows the expected Poisson distribution, as a function of the mean of the data ($\bar{x}$), with a one standard deviation range also shaded in red (where the expected standard deviation for the Poisson is $\sigma = \sqrt{\bar{x}}$).

The right side of each panel shows the 2D images. In order to visualize the detector counts per pixel, we must map the number of counts to an RGB color value, which we use the matplolib 'gray' colormap for. To emphasize the features in the data, we choose the min an max values of the colormap to be the 10th and 90th percentile counts of the data.

4.3. *Mean of the Mean & Standard Deviation of the Mean*

Describe computation, show plots

4.4. *Quatifying Detector Noise*

Calculate readnoise n stuff

5. DISCUSSION

5.1. *Poisson & Gaussian Comparisons*

Looking at Figures 2 and 3, we find that the detector count distribution is close to a Poisson distribution for low exposure times, but for larger exposure times it increasingly diverges with a

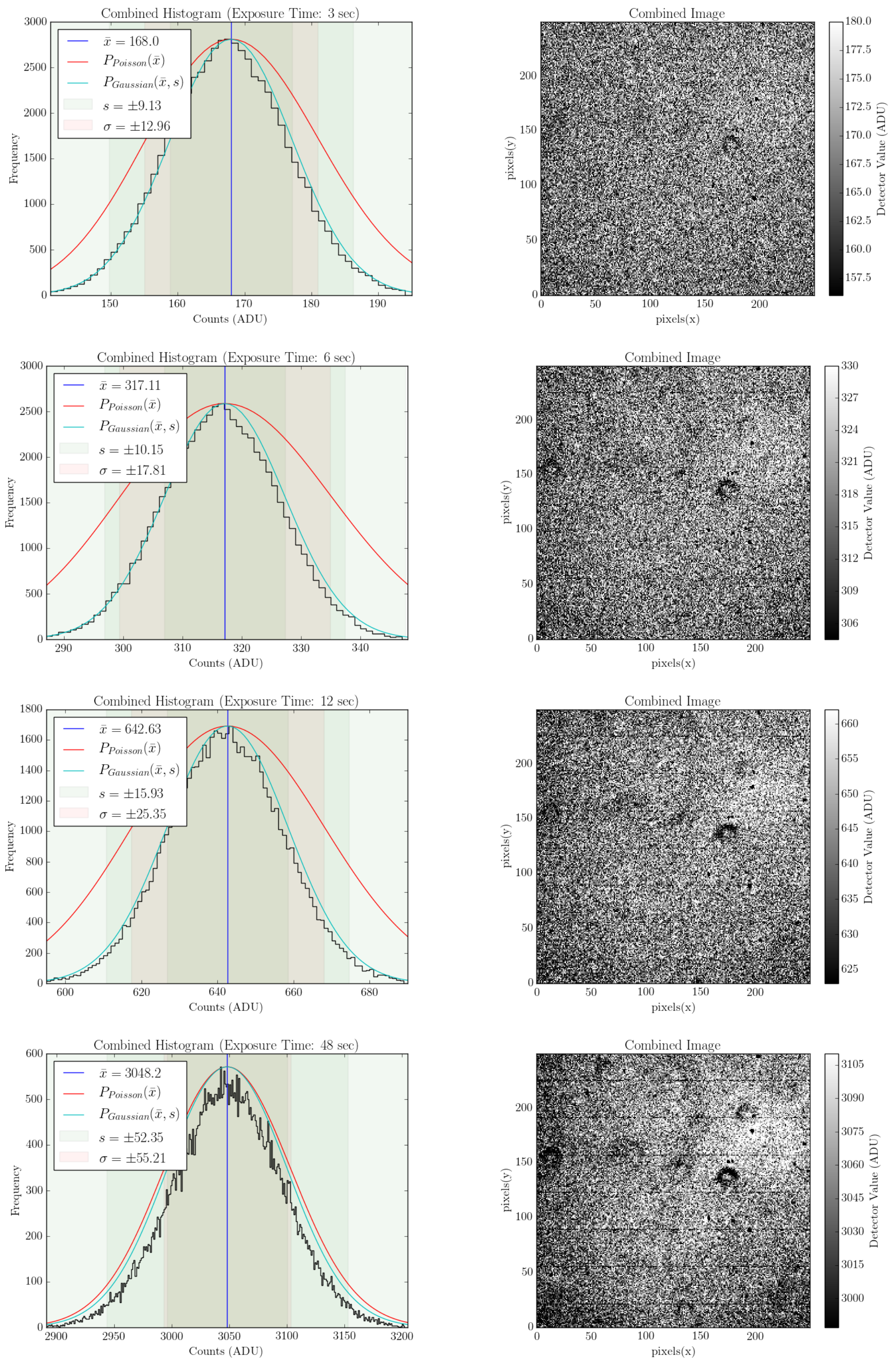

Figure 2. Combined, bias-subtracted, flat frame histograms and images for varying exposure times (0, 3, 6, and 12 sec).

much larger standard deviations than expected (as summarized by Figure 4). We also see that low

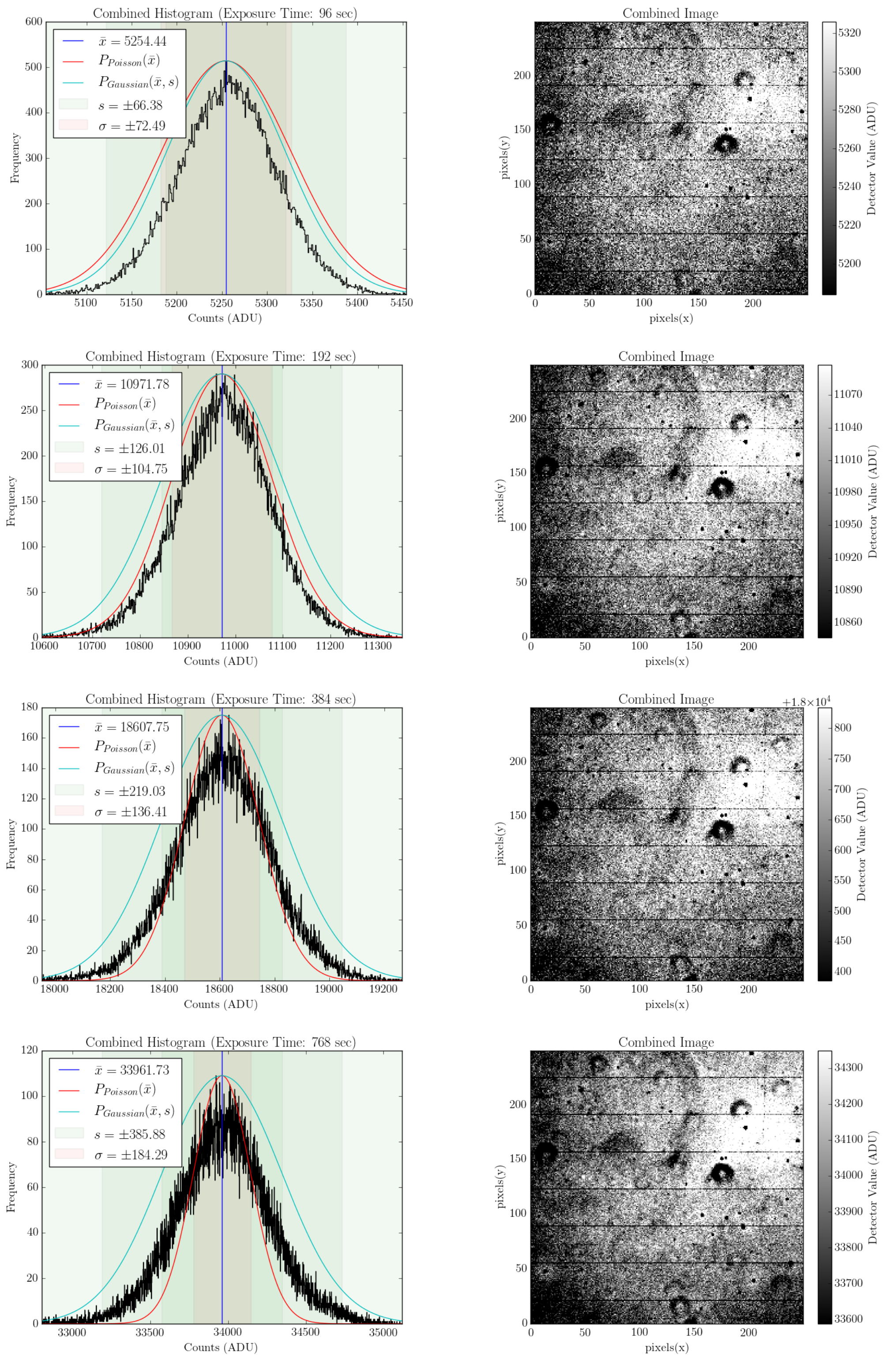

Figure 3. Combined, bias-subtracted, flat frame histograms and images for varying exposure times (96, 192, 384, and 768 sec).

exposure times (3, 6, and 12 sec) have a single peak in the distribution around the mean, however

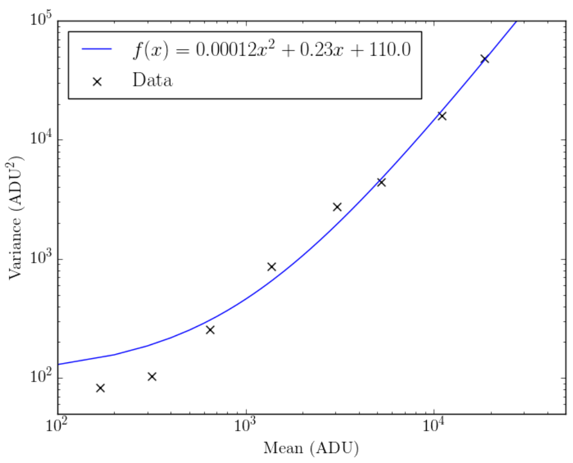

Figure 4. Mean vs. variance for all bias-subtracted exposures, fitted to a second-order polynomial.

at 48 sec the distribution becomes bimodal, and for high exposure times (96, 192, 384, and 768 sec) the distribution develops three separate peaks. Comparing these histograms to the corresponding images at the right, we can see the regions on the detector which correspond to the different peaks: the dark edges make up roughly the bottom third of the distribution, and the bright center making up the higher thrid of the distribution. If the distribution was Poissonian for each pixel, we would expect the signal to vary uniformly across the detector, but instead there is obvious contamination at different spatial regions.

5.2. *Sources of Noise & Systematic Effects*

Variations in the quantum efficiency of each pixel. Dust particles on the lens scatter and absorb light unevenly at different points on the detector.

Checked for systematics due to temperature variation that could maybe be responsible for dark current: found temperature to vary only within 4C (Figure 5). Checked for systematic differences in different datasets.

6. CONCLUSION

7. AUTHOR CONTRIBUTIONS

This project was done in collaboration with Julian Beas-Gonzalez and Russell Van-Linge (Group E), using data collected by Group B. Russell wrote code for fitting Poisson and Gaussian distribtions and computing MOM/SDOM, Julian wrote code for histograms and computing MOM/SDOM, and I wrote code for combining frames/bias subtracting images, plotting histograms/images and slicing images (except for a few snippets for making histograms and reading fits files from the class notes). We met several times outside of class to compare results, fix issues, and combine our code together, which can be found on github: https://github.com/jbirky/photonCounting.

8. APPENDIX

8.1. *Statistics*

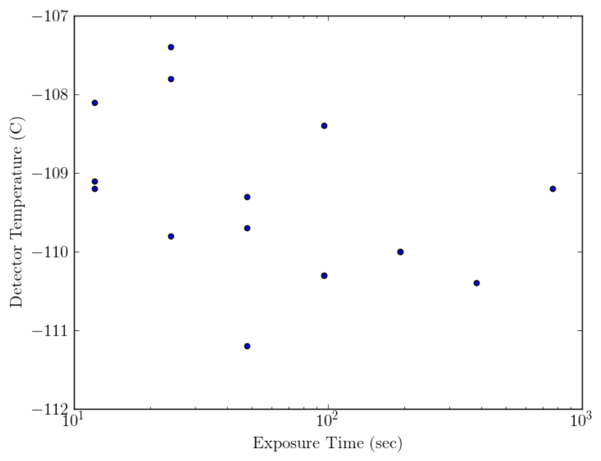

Figure 5. Temperature of the CCD detector for different exposure frames does not vary more than about 4C.

Mean and variance for a set of data points $x = x_1, ..., x_N$:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{4}$$

$$s^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2 \tag{5}$$

8.2. *Code*

Statistical Code

```python
def mean(array):
    return np.sum(array)/len(array)

def stdev(array):
    return np.sqrt(sum((array - mean(array))**2)/(len(array) - 1))

def poisson_approx(data, mean):
    pdist = np.array(np.exp((data*np.log(mean))-(data*np.log(data))+data-mean))
    return pdist

def gaussian(data,mean,sigma):
    gdist = np.array((1/(sigma*(2*math.pi)**(.5)))*np.exp(-.5*((data-mean)/sigma)**2))
    return gdist
```

Data Processing code

```python
def readData(folder):
    """
    Read all frames from a given directory into one matrix.
    input:  directory to folder containing frames
    output: array3d (dimension: xpixel x ypixel x # frames)
```

```
            array2D (dimension: 1D flattened img x # frames)
    """

    files = os.listdir(folder)

    array3D, array2D = [], []
    for ff in files:
        arr = fits.getdata(folder + ff)
        array3D.append(arr)
        array2D.append(arr.flatten())

    return np.array(array3D)

def combineFrame(data_array):
    """
    Combine image frames and return 1D array.
    input:  data array (dimension: # 1D detector counts x # frames)
    output: 1D combined array of detector counts (ADU)
    """

    return np.median(data_array, axis=0)
```

Histogram/Image code

```
def plotAll(array2D, **kwargs):

    arr = array2D.flatten()

    avg = mean(arr)
    std = stdev(arr)
    med = np.median(arr)
    Npix = len(arr)

    sigma = kwargs.get('sigma', 2)
    low = int(np.round((avg-sigma*std)))
    high = int(np.round((avg+sigma*std)))
    rng = kwargs.get('rng', [low, high])
    exp = kwargs.get('exp')
    if 'nbins' in kwargs:
        nbins = kwargs.get('nbins')
        bin_size = (rng[1]-rng[0])/nbins
    else:
        bin_size = kwargs.get('bin_size', 1)

    fig, (ax1, ax2) = plt.subplots(1,2, figsize=[18,6])

    # Histogram
    #===========
    hr = np.arange(rng[0], rng[1]+1, bin_size)
    hist = []
    for i in range(len(hr)):
        try:
            counts = len(np.where((arr >= hr[i]) & (arr < hr[i+1]))[0])
        except:
            counts = 0
        hist.append(counts)
    ax1.step(hr, hist, color='k')

    #mean and median lines
    ax1.axvline(avg, color='b', label=r'$\bar{x}=%s$'%(np.round(avg,2)))

    #sigma levels
    if kwargs.get('show_level', True) == True:
        for i in np.arange(1,sigma+1):
            if i == 1:
                ax1.axvspan(avg-i*std, avg+i*std, facecolor='g', alpha=0.05, \
                    label=r'$s=\pm␣%s$'%(np.round(std,2)))
            else:
                ax1.axvspan(avg-i*std, avg+i*std, facecolor='g', alpha=0.05)


    #poisson distribution
    xarray = np.arange(rng[0]-10, rng[1]+10, 1)
    pdist = poisson_approx(xarray, avg)
    pdist = max(hist)/max(pdist)*pdist
    ax1.plot(xarray, pdist, color='r', label=r'$P_{Poisson}(\bar{x})$')
    std_expected = math.sqrt(avg)
    ax1.axvspan(avg - std_expected, avg + std_expected, facecolor='r', alpha=0.05, \
                label=r'$\sigma=\pm␣%s$'%(np.round(std_expected,2)))

    #gaussian distribution
    gdist = gaussian(xarray, avg, std)
    gdist = max(hist)/max(gdist)*gdist
    ax1.plot(xarray, gdist, color='c', label=r'$P_{Gaussian}(\bar{x},␣s)$')

    ax1.legend(loc='upper␣left')
    ax1.set_xlabel('Counts␣(ADU)')
    ax1.set_ylabel('Frequency')

    if 'exp' in kwargs:
        ax1.set_title('Combined␣Histogram␣(Exposure␣Time:␣%s␣sec)'%(exp))
    ax1.set_xlim(rng)

    # Image
    #===========
    hrng = kwargs.get('hrng', [np.percentile(arr, 10), np.percentile(arr, 90)])
    pl = ax2.imshow(array2D, origin='lower', interpolation='nearest', \
         cmap='gray', vmin=hrng[0], vmax=hrng[1])
    fig.colorbar(pl, ax=ax2, fraction=0.046, pad=0.04).set_label('Detector␣Value␣(ADU)')

    ax2.set_xlabel('pixels(x)')
    ax2.set_ylabel('pixels(y)')
    ax2.set_title('Combined␣Image')

    if 'save_dir' in kwargs:
        save_dir = kwargs.get('save_dir')
        plt.savefig(save_dir + 'exposure%s.png'%(exp))
    plt.show()
```