Cosmology Project #1
Isaac Malsky
Proving that the Universe is Isotropic on large scales

Introduction

A central tenent of cosmology is the idea that the Universe is homogeneous and isotropic on large scales. Taken together, these two assumptions make up the Cosmological principal. Homogeneity implies that the properties of the Universe, such as density, do not vary with position. Isotropy implies that the properties of the Universe do not vary based on direction. Neither homogeneity or isotropy hold on small scales. For example, the average density is far greater towards the center of the Milky Way Galaxy than far outside of our Galaxy.

Any test of isotropy or homogeneity is also limited by the fact that we can only make observations from a single location in the Universe. On the scales of interest, Earth is nearly a single point in the Universe, and all observations can only be taken with respect to our position as a single observer. Therefore, I make the simplifying assumption of the Copernican Principle: that our location in the Universe is not special. While it is theoretically possible that the Universe could be isotropic and spherically symmetric with Earth at it's center, we proceed from the simpler assumption that Earth has no privileged spatial position in the Universe. One important note is that any Universe that is isotropic about 2 distinct points is also homogeneous.

Many observations have provided strong evidence that the Universe is homogeneous and isotropic. For example, the discovery of the cosmic microwave background (Penzias et al 1965) showed a nearly isotropic background of radiation for the Universe. These findings were confirmed through decades of observations. Larger, more technically advanced surveys, such as the COsmic Background Explorer (Cobe), and Planck refined and confirmed these blackbody measurements (DJ et al 1996, Aghanim et al 2018). Quantifying the scale of anisotropies is an important step in understanding the properties of the early Universe.

In this project I sample quasar number densities for constant z shifts in order to quantify the anisotropies. I show that the distribution of quasars in the Universe becomes isotropic for redshifts larger than approximately z=0.50.

Data

In this project, I used the quasar dataset from the 16th SDSS release (Blanton et al, 2009, Ahumada et al 2019, Lyke et al, 2020). The catalogue contains 750,414 quarsars, including subsets from previous SDSS surveys. This is the most comprehensive quasar catalog at present. The eBoss Survey (Dawson et al 2016) covered an area of 7,500 square degrees, and observed over 500,000 quasars with redshifts between 0.8 and 2.2. This survey was tested with the DR16Q, which contained over 480,000 quasars with redshifts between 0.8 and 2.2.

Methods

In order to show the isotropy of the Universe, I compare the number density of quasars in different sections of the sky.

First, I qualitatively show the distribution of quasars in the survey, to show both the initial object density as well as the area on the sky that was observed during the survey. This is necessary in order to choose the solid angle that I explore on the sky, as it is critically important that the analysis below is performed on data that was observed in a standarized way. Any error introduced in the underlying data, such as longer observation periods for different sections of the sky, would chance anisotropy quantification.

In addition to these quantitative projections I also show the distribution of the sample in terms of redshift, signal to noise, absolute magnitude, apparent magnitude, and several other characterisitcs. The majority of quasars in the survey have redshifts between 0 and 4, and absolute I band magnitudes brighter than -20.

To calculate the absolute magnitude of a quasar, it will be necessary to use the luminosity distance. Luckily however, the absolute magnitude is calculated in the survey with the following cosmology: H_0 = 67.6 km/s Mpc, Ω_m = 0.31, Ω_Λ = 0.69, and Ω_R = 9.11x10^{-5}. Additionally, these values are calculated with K corrections from Richards (2006). To maintain consistency, I adopt these comsological parameters when calculating luminosity distance for all of my figures also.

Second, I filter the dataset to only include objects with signal to noise above 3, and an I band absolute magnitude below -20. I also selected the DR7Q results and the DR12Q results from within the catalog. The full dataset includes a composite of quasars, and combining these results would incorrectly show that the Universe is anisotropic, based solely on incorrectly characterizing the areas that are covered by more than one survey. For this work, I only use the SDSS7 survey. With all filtering in place, the SDSS7 survey has 84148 objects while the SDSS12 survey has 108398 objects.

Third, I divided the dataset into groups within a certain redshift. Each volume included quasars from 0 to some maximum set z. For each redshift, this created conical volumes, for which I could count the number of quasars. Then, in each subgroup, I chose a range of right ascension and declination that they survey covered. Within this range of RA and DEC, I randomly sampled 1 degree^2 patches of the sky and counted the number of objects within this patch. I analyzed the resulting distribution, and also calculated the expected number of objects per degree squared (total objects / total number of square degrees). I repeated this procedure for mulitple z shifts, in order to show how the anisotropies of the quasar distribution changed with increasing redshift.

In calculating the solid angle subtended in the sample, I accounted for the spherical geometry of the solid angle according to

$$\Omega = \int sin(\theta) d\theta \int d\phi$$

Organizing the project this way allowed for significant simplifications. First, there was no need to adjust for cosmological volume effects as would be necessary if I was choosing volumes that were not radially symmetric. Showing isotropy only required showing that there was no directionality to the count rates, and choosing radially symmetric patches of the sky ensured that different solid angle choices would always contain the same volume, as long as I only compared for constant maximum z values.

Results and Code

Import Libraries

In [144]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.ticker as mtick
import matplotlib as mpl
from matplotlib import rcParams, rc
import matplotlib.colors as mcolors
from astropy.io.fits import getdata
from matplotlib.colors import ListedColormap
import astropy.coordinates as coord
import astropy.units as u

import numpy as np
import matplotlib.pyplot as plt
from astropy.io import ascii
from astropy.coordinates import SkyCoord
from astropy import units

import random

from astropy.cosmology import FlatLambdaCDM
import astropy.units as u

import seaborn as sns
import matplotlib.mlab as mlab


plt.style.use('./mplstyle.txt')
```

Functions used in calculating objects per dictionary

In [145]:

```
def get_total_objects(dictionary):
    """ Go through a dictionary of dataframes and count all the objects"""
    total_objects = 0
    for i in range(len(dictionary)):
        total_objects = total_objects + len(dictionary[i])
    return total_objects

def get_entropy(dictionary, total_objects):
    """Get the entropy of the item in the dictionary"""
    temp_val = 0
    for i in range(len(dictionary)):
        temp_val = temp_val + (len(dictionary[i]) * np.log10(len(dictionary[i
])))
    return np.log10(total_objects) - (temp_val)/total_objects
```

Read in the Data

In [146]:

```
data = getdata('/Users/imalsky/Desktop/DR16Q_v4.fits',1)
```

Get the shape of the dataset and the column names

In [147]:

```
print('galaxies in sample:', data.shape)
print('data fields:', data.dtype.names)
```

```
galaxies in sample: (750414,)
data fields: ('SDSS_NAME', 'RA', 'DEC', 'PLATE', 'MJD', 'FIBERID',
'AUTOCLASS_PQN', 'AUTOCLASS_DR14Q', 'IS_QSO_QN', 'Z_QN', 'RANDOM_SEL
ECT', 'Z_10K', 'Z_CONF_10K', 'PIPE_CORR_10K', 'IS_QSO_10K', 'THING_I
D', 'Z_VI', 'Z_CONF', 'CLASS_PERSON', 'Z_DR12Q', 'IS_QSO_DR12Q', 'Z_
DR7Q_SCH', 'IS_QSO_DR7Q', 'Z_DR6Q_HW', 'Z_DR7Q_HW', 'IS_QSO_FINAL',
'Z', 'SOURCE_Z', 'Z_PIPE', 'ZWARNING', 'OBJID', 'Z_PCA', 'ZWARN_PC
A', 'DELTACHI2_PCA', 'Z_HALPHA', 'ZWARN_HALPHA', 'DELTACHI2_HALPHA',
'Z_HBETA', 'ZWARN_HBETA', 'DELTACHI2_HBETA', 'Z_MGII', 'ZWARN_MGII',
'DELTACHI2_MGII', 'Z_CIII', 'ZWARN_CIII', 'DELTACHI2_CIII', 'Z_CIV',
'ZWARN_CIV', 'DELTACHI2_CIV', 'Z_LYA', 'ZWARN_LYA', 'DELTACHI2_LYA',
'Z_LYAWG', 'Z_DLA', 'NHI_DLA', 'CONF_DLA', 'BAL_PROB', 'BI_CIV', 'ER
R_BI_CIV', 'AI_CIV', 'ERR_AI_CIV', 'BI_SIIV', 'ERR_BI_SIIV', 'AI_SII
V', 'ERR_AI_SIIV', 'BOSS_TARGET1', 'EBOSS_TARGET0', 'EBOSS_TARGET1',
'EBOSS_TARGET2', 'ANCILLARY_TARGET1', 'ANCILLARY_TARGET2', 'NSPEC_SD
SS', 'NSPEC_BOSS', 'NSPEC', 'PLATE_DUPLICATE', 'MJD_DUPLICATE', 'FIB
ERID_DUPLICATE', 'SPECTRO_DUPLICATE', 'SKYVERSION', 'RUN_NUMBER', 'R
ERUN_NUMBER', 'CAMCOL_NUMBER', 'FIELD_NUMBER', 'ID_NUMBER', 'LAMBDA_
EFF', 'ZOFFSET', 'XFOCAL', 'YFOCAL', 'CHUNK', 'TILE', 'PLATESN2', 'P
SFFLUX', 'PSFFLUX_IVAR', 'PSFMAG', 'PSFMAGERR', 'EXTINCTION', 'M_I',
'SN_MEDIAN_ALL', 'GALEX_MATCHED', 'FUV', 'FUV_IVAR', 'NUV', 'NUV_IVA
R', 'UKIDSS_MATCHED', 'YFLUX', 'YFLUX_ERR', 'JFLUX', 'JFLUX_ERR', 'H
FLUX', 'HFLUX_ERR', 'KFLUX', 'KFLUX_ERR', 'W1_FLUX', 'W1_FLUX_IVAR',
'W1_MAG', 'W1_MAG_ERR', 'W1_CHI2', 'W1_FLUX_SNR', 'W1_SRC_FRAC', 'W1
_EXT_FLUX', 'W1_EXT_FRAC', 'W1_NPIX', 'W2_FLUX', 'W2_FLUX_IVAR', 'W2
_MAG', 'W2_MAG_ERR', 'W2_CHI2', 'W2_FLUX_SNR', 'W2_SRC_FRAC', 'W2_EX
T_FLUX', 'W2_EXT_FRAC', 'W2_NPIX', 'FIRST_MATCHED', 'FIRST_FLUX', 'F
IRST_SNR', 'SDSS2FIRST_SEP', 'JMAG', 'JMAG_ERR', 'JSNR', 'JRDFLAG',
'HMAG', 'HMAG_ERR', 'HSNR', 'HRDFLAG', 'KMAG', 'KMAG_ERR', 'KSNR',
'KRDFLAG', 'SDSS2MASS_SEP', '2RXS_ID', '2RXS_RA', '2RXS_DEC', '2RXS_
SRC_FLUX', '2RXS_SRC_FLUX_ERR', 'SDSS2ROSAT_SEP', 'XMM_SRC_ID', 'XMM
_RA', 'XMM_DEC', 'XMM_SOFT_FLUX', 'XMM_SOFT_FLUX_ERR', 'XMM_HARD_FLU
X', 'XMM_HARD_FLUX_ERR', 'XMM_TOTAL_FLUX', 'XMM_TOTAL_FLUX_ERR', 'XM
M_TOTAL_LUM', 'SDSS2XMM_SEP', 'GAIA_MATCHED', 'GAIA_DESIGNATION', 'G
AIA_RA', 'GAIA_DEC', 'GAIA_PARALLAX', 'GAIA_PARALLAX_ERR', 'GAIA_PM_
RA', 'GAIA_PM_RA_ERR', 'GAIA_PM_DEC', 'GAIA_PM_DEC_ERR', 'GAIA_G_MA
G', 'GAIA_G_FLUX_SNR', 'GAIA_BP_MAG', 'GAIA_BP_FLUX_SNR', 'GAIA_RP_M
AG', 'GAIA_RP_FLUX_SNR', 'SDSS2GAIA_SEP')
```

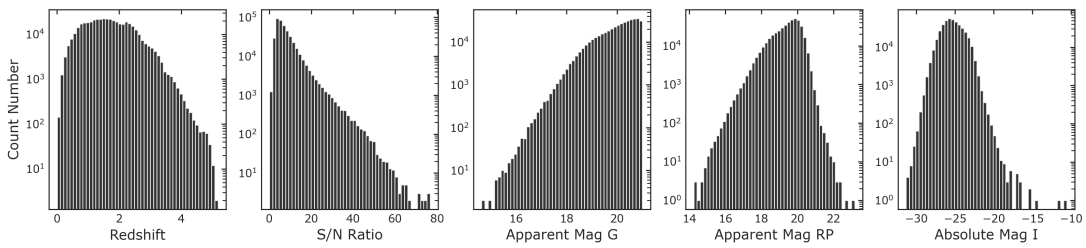
Create summary graphs of the data

In [148]:

```
fig, ax = plt.subplots(nrows=1, ncols=5, figsize=(20, 4), sharex=False, squeeze=
True)
plt.subplots_adjust(hspace=0.1, wspace=0.2)


cut = np.where((data['Z'] > 0.0) &
               (data['SN_MEDIAN_ALL'] > 0) &
               (data['GAIA_G_MAG'] > 0) &
               (data['GAIA_RP_MAG'] > 0) &
               (data['IS_QSO_FINAL'] == 1) &
               (data['M_I'] < 0))
histogram_data = data[cut]


n_bins = 50


# We can set the number of bins with the `bins` kwarg
ax[0].hist(histogram_data['Z'], bins=n_bins,  alpha=0.8, color='black')
ax[1].hist(histogram_data['SN_MEDIAN_ALL'], bins=n_bins, alpha=0.8, color='blac
k')
ax[2].hist(histogram_data['GAIA_G_MAG'], bins=n_bins, alpha=0.8, color='black')
ax[3].hist(histogram_data['GAIA_RP_MAG'], bins=n_bins, alpha=0.8, color='black')
ax[4].hist(histogram_data['M_I'], bins=n_bins, alpha=0.8, color='black')


ax[0].set_yscale('log')
ax[1].set_yscale('log')
ax[2].set_yscale('log')
ax[3].set_yscale('log')
ax[4].set_yscale('log')

ax[0].set_xlabel('Redshift')
ax[1].set_xlabel('S/N Ratio')
ax[2].set_xlabel('Apparent Mag G')
ax[3].set_xlabel('Apparent Mag RP')
ax[4].set_xlabel('Absolute Mag I')

ax[0].set_ylabel('Count Number')
```

Out[148]:

```
Text(0, 0.5, 'Count Number')
```

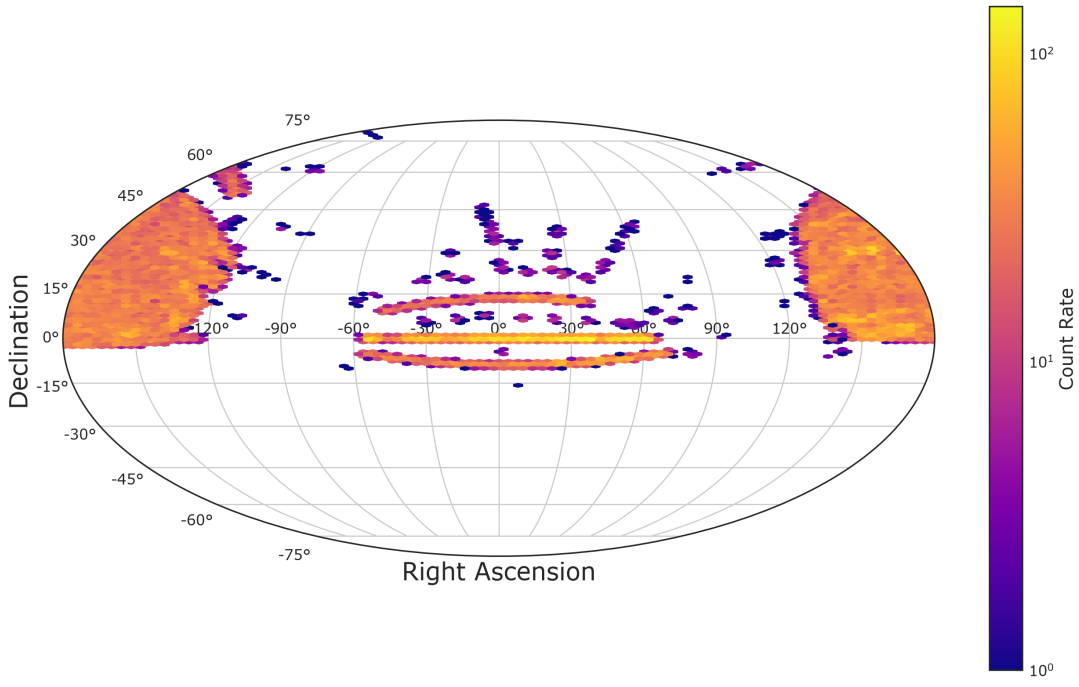


The dataset has redshifts mainly between 0 and 5, absolute I band magnitudes less predominantly less than -20, and signal to noise ratios with a peak below 20 and a long tail towards stronger signals. The number of quasars per redshift bin stays appoximately constant between 0 and 2.

Visualize the points in the DR7

In [149]:

```
cut = np.where((data['Z'] > 0.0) &
               (data['SN_MEDIAN_ALL'] > 3) &
               #(data['SN_MEDIAN_ALL'] < 60) &
             #  (data['GAIA_G_MAG'] > 16) &
             #  (data['GAIA_RP_MAG'] > 0) &
               (data['M_I'] < -20) &
               (data['IS_QSO_DR7Q'] == 1))
sky_data7 = data[cut]


coords = SkyCoord(ra=sky_data7['ra'], dec=sky_data7['dec'], unit='degree')
ra = coords.ra.wrap_at(180 * units.deg).radian
dec = coords.dec.radian

color_map = plt.cm.plasma
fig = plt.figure(figsize=(16, 10))
fig.add_subplot(111, projection='mollweide')

image = plt.hexbin(ra, dec, cmap=color_map, gridsize=100, mincnt=1, bins='log')

plt.xlabel('Right Ascension', size=20, fontstyle='normal')
plt.ylabel('Declination', size=20, fontstyle='normal')
plt.grid(True)
plt.colorbar(image, spacing='uniform', label='Count Rate')
plt.show()
```

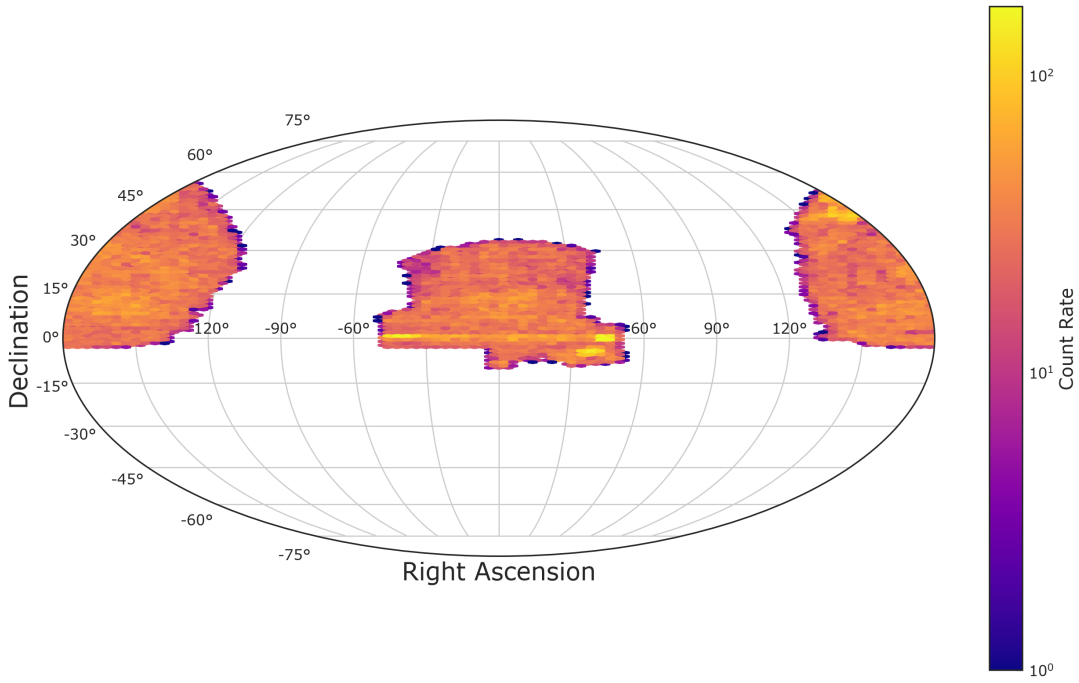


The DR7 subset of the survey is gragmented across the sky. The above results are distinctinly anisotropic. However, this is only a function of the limited observational area. The later results must be restricted to only the a portion of the sky that was observered in a standardized way.

Visualize the points in the DR12

In [150]:

```
cut = np.where((data['Z'] > 0.0) &
               (data['SN_MEDIAN_ALL'] > 3) &
              # (data['SN_MEDIAN_ALL'] < 60) &
            #   (data['GAIA_G_MAG'] > 16) &
             #  (data['GAIA_RP_MAG'] > 0) &
               (data['M_I'] < -20) &
               (data['IS_QSO_DR12Q'] == 1))
sky_data12 = data[cut]


coords = SkyCoord(ra=sky_data12['ra'], dec=sky_data12['dec'], unit='degree')
ra = coords.ra.wrap_at(180 * units.deg).radian
dec = coords.dec.radian

color_map = plt.cm.plasma
fig = plt.figure(figsize=(16, 10))
fig.add_subplot(111, projection='mollweide')

image = plt.hexbin(ra, dec, cmap=color_map, gridsize=100, mincnt=1, bins='log')

plt.xlabel('Right Ascension', size=20, fontstyle='normal')
plt.ylabel('Declination', size=20, fontstyle='normal')
plt.grid(True)
plt.colorbar(image, spacing='uniform', label='Count Rate')
plt.show()
```



In [151]:

```
print ("The SDSS7 survey has:", len(sky_data7), "objects")
print ("The SDSS12 survey has:", len(sky_data12), "objects")
```

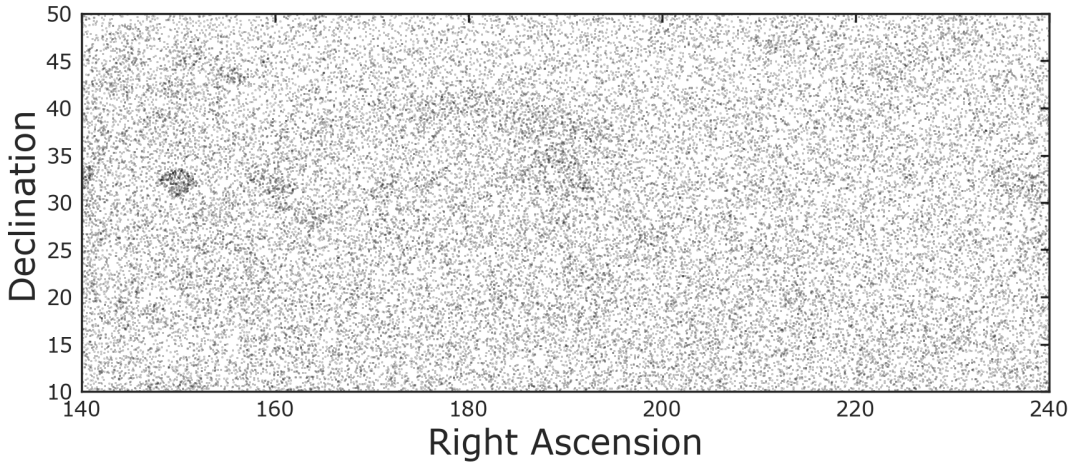
```
The SDSS7 survey has: 97588 objects
The SDSS12 survey has: 174809 objects
```

In [152]:

```
fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(10, 4), sharex=False, squeeze=
True)
plt.subplots_adjust(hspace=0.1, wspace=0.2)

ax.scatter(sky_data7['ra'], sky_data7['dec'], color='black', s=0.01)

ax.set_xlim([140, 240])
ax.set_ylim([10, 50])

ax.set_xlabel('Right Ascension', size=20, fontstyle='normal')
ax.set_ylabel('Declination', size=20, fontstyle='normal')
```

Out[152]:

```
Text(0, 0.5, 'Declination')
```



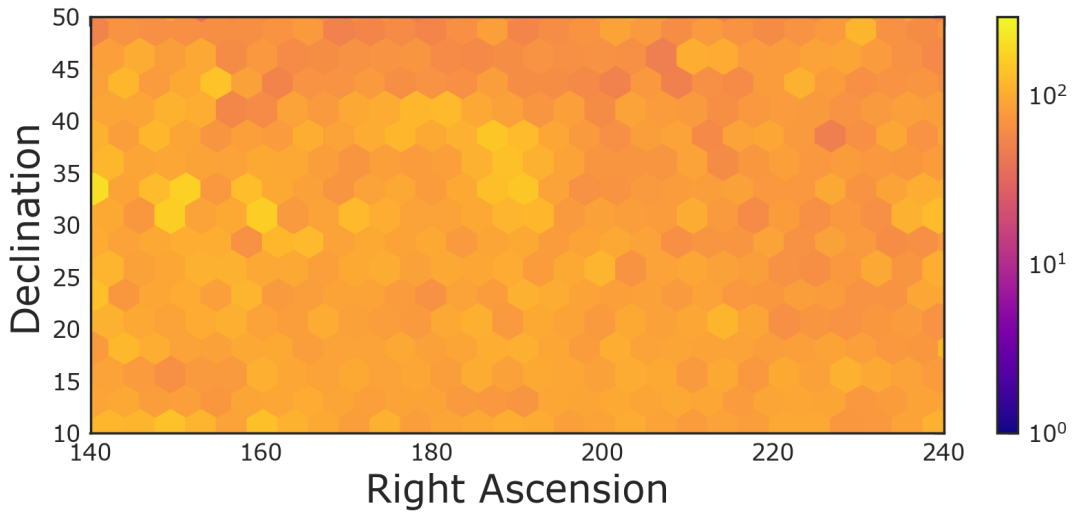
Unsorted look at count numbers for all z

In [153]:

```
fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(10, 4), sharex=False, squeeze=
True)
plt.subplots_adjust(hspace=0.1, wspace=0.2)

image = ax.hexbin(sky_data7['ra'], sky_data7['dec'], gridsize=(100,20), cmap='pl
asma',bins='log')
plt.colorbar(image, spacing='uniform')

ax.set_xlim([140, 240])
ax.set_ylim([10, 50])

ax.set_xlabel('Right Ascension', size=20, fontstyle='normal')
ax.set_ylabel('Declination', size=20, fontstyle='normal')
```

Out[153]:

```
Text(0, 0.5, 'Declination')
```



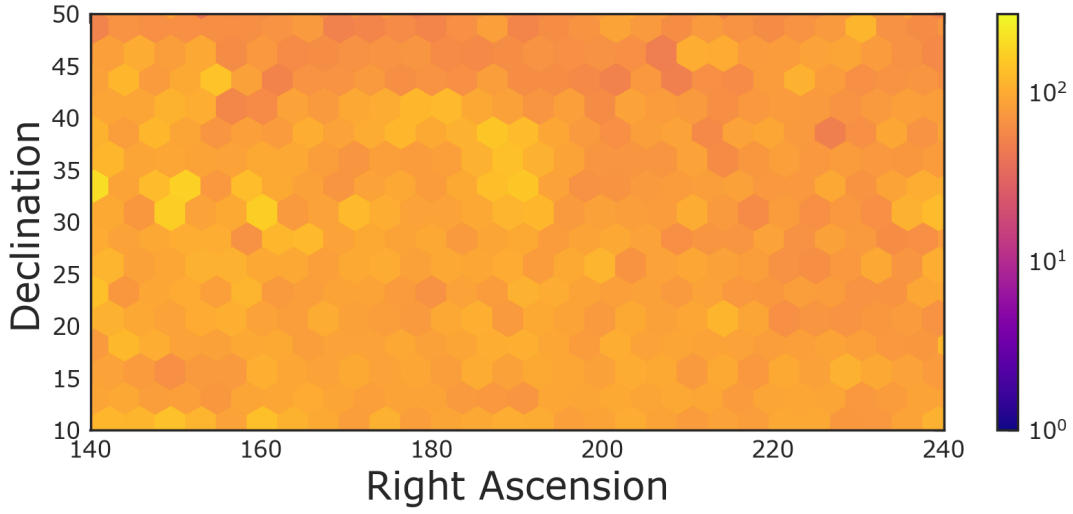
Unsorted look at count numbers for z < 2

In [154]:

```
fig, ax = plt.subplots(nrows=1, ncols=1, figsize=(10, 4), sharex=False, squeeze=
True)
plt.subplots_adjust(hspace=0.1, wspace=0.2)

cut = np.where((data['Z'] > 0.0) &
               (data['Z'] < 2.0) &
               (data['SN_MEDIAN_ALL'] > 3) &
               (data['M_I'] < -20) &
               (data['IS_QSO_DR7Q'] == 1))
sky_data7_again = data[cut]


image = ax.hexbin(sky_data7['ra'], sky_data7['dec'], gridsize=(100,20), cmap='pl
asma',bins='log')
plt.colorbar(image, spacing='uniform')

ax.set_xlim([140, 240])
ax.set_ylim([10, 50])

ax.set_xlabel('Right Ascension', size=20, fontstyle='normal')
ax.set_ylabel('Declination', size=20, fontstyle='normal')
```

Out[154]:

```
Text(0, 0.5, 'Declination')
```

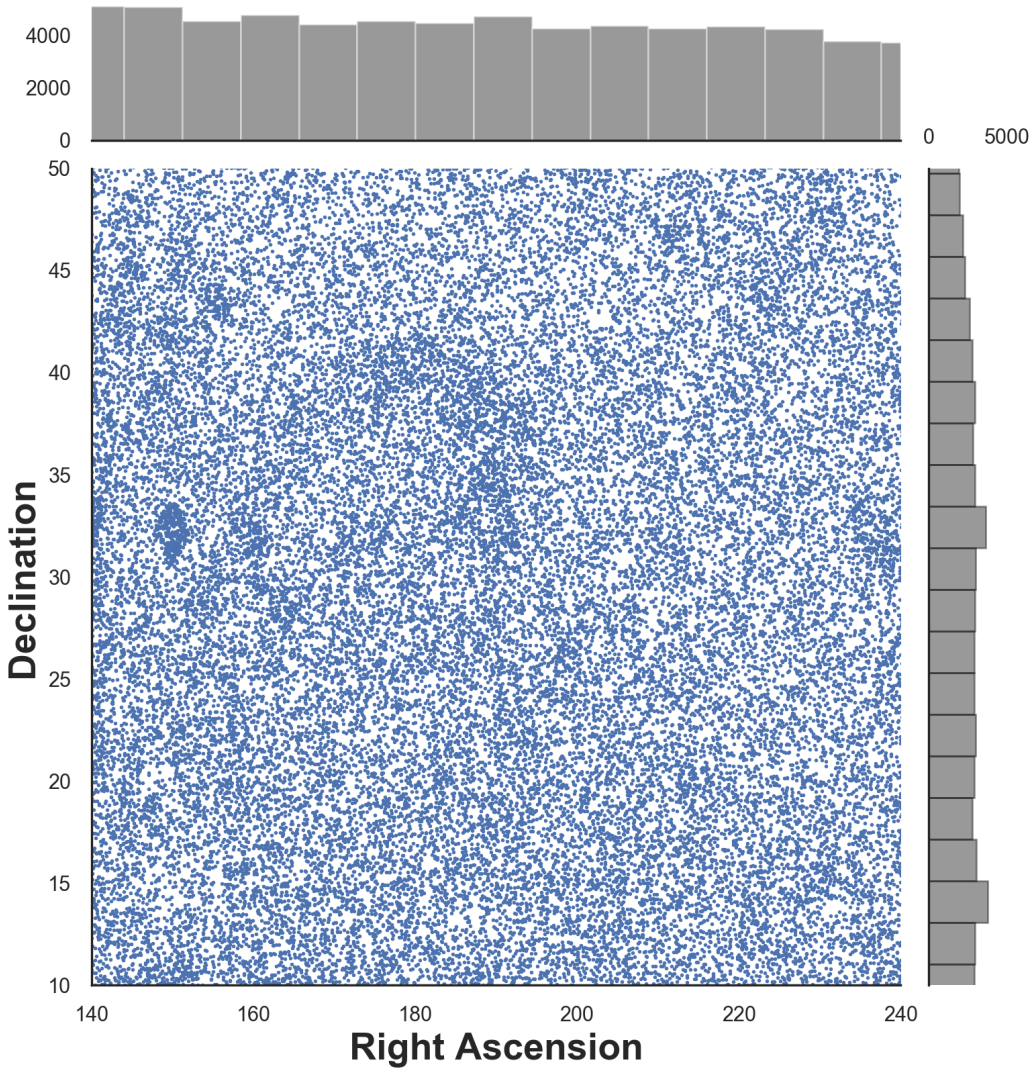


In [176]:

```
import seaborn as sns


#sns.set(style="white", color_codes=True)

plot = sns.jointplot(sky_data7['ra'], sky_data7['dec'], kind="scatter", s=1,
                  stat_func=None, marginal_kws={'color': 'black'}, height=8)

plt.setp(plot.ax_marg_y.patches, color="black")

plot.ax_marg_y.tick_params(labeltop=True)
plot.ax_marg_x.tick_params(labelleft=True)


plot.ax_marg_x.set_xlim(140, 240)
plot.ax_marg_y.set_ylim(10, 50)


plot.ax_joint.set_xlabel('Right Ascension', fontweight='bold', fontsize=20)
plot.ax_joint.set_ylabel('Declination', fontweight='bold', fontsize=20)
```

Out[176]:

```
Text(115.83333333333334, 0.5, 'Declination')
```



The graphs above give a qualitiative sense of the isotropy, but do not qualitatively show whether the Universe is isotropic, and also are limited in that they do not show how the anisotropies change with z.

I need to find the total solid angle subtended by the survey in the area of interest, and then normalize based on cos(θ)

$$\Omega = \int d\theta sin(\theta) \int d\phi$$

$$\Omega = \int d\theta_{40°}^{90°} sin(\theta) \int_{140°}^{240°} d\phi$$

$$\Omega = 1.03 \; radians^2$$

$$\Omega = 3,381 \; degrees^2$$

For different z values, I now sample 1 degree squared solid angles and create histrograms for the number of counts in each sample. Additionally, I compare these values to the expected count rate based on the total number of objects in the total observational area

Function to get the different data subsets

In [156]:

```
def get_cut_for_z(num_solid_angles, z, d_solid_angle):
    df = {}
    for j in range(num_solid_angles):
        ra  = random.randint(140, 240)
        dec = random.randint(10, 50)
        cut = np.where((data['Z'] > 0) &
                       (data['Z'] < z) &
                       (data['SN_MEDIAN_ALL'] > 3) &
                     #  (data['SN_MEDIAN_ALL'] < 60) &
                     #  (data['GAIA_G_MAG'] > 16) &
                    #    (data['GAIA_RP_MAG'] > 0) &
                       (data['M_I'] < -20) &
                       (data['IS_QSO_DR7Q'] == 1) &
                       (data['RA'] > ra - (d_solid_angle / np.cos(dec * np.pi /
180))) &
                       (data['RA'] < ra + (d_solid_angle / np.cos(dec * np.pi /
180))) &
                       (data['DEC'] > dec - d_solid_angle) &
                       (data['DEC'] < dec + d_solid_angle))
        df[j] = data[cut]

    cut = np.where((data['Z'] > 0) &
                   (data['Z'] < z) &
                   (data['SN_MEDIAN_ALL'] > 3) &
                #   (data['SN_MEDIAN_ALL'] < 60) &
                #   (data['GAIA_G_MAG'] > 16) &
                #   (data['GAIA_RP_MAG'] > 0) &
                   (data['M_I'] < -20) &
                   (data['IS_QSO_DR7Q'] == 1) &
                   (data['RA'] > 140) &
                   (data['RA'] < 240) &
                   (data['DEC'] > 10) &
                   (data['DEC'] < 50))


    total_counts = len(data[cut])
    total_degrees = 3381 # THIS WAS REALLY HARD
    expected_number = ((d_solid_angle*2)**2 / total_degrees) * total_counts

    N = get_total_objects(df)

    counts = []
    for key in df:
        counts.append(len(df[key]))

    return (expected_number, counts)
```

Z of 0.25

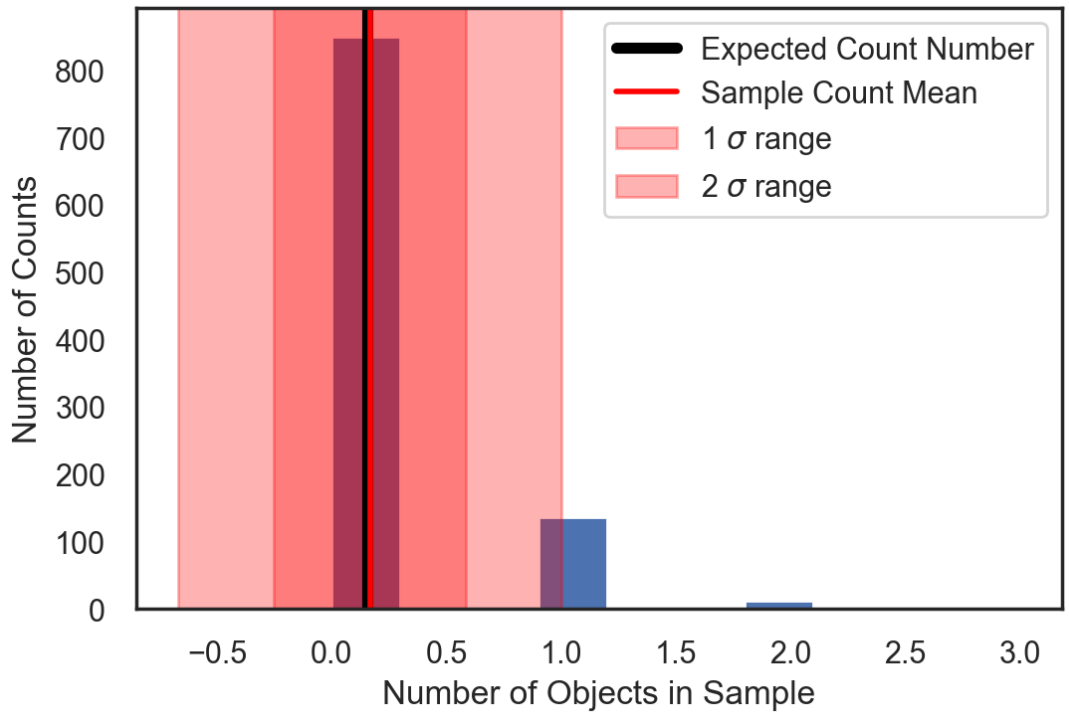
In [193]:

```
#expected_number1, counts1 = get_cut_for_z(1000, 0.25, 0.5)

plt.hist(counts1)
plt.axvline(x=expected_number1, linewidth=4, color='black', label=r'Expected Cou
nt Number')
plt.axvline(x=np.mean(counts1), linewidth=2, color='red', label=r'Sample Count M
ean')
plt.axvspan(np.mean(counts1) - np.std(counts1),
            np.mean(counts1) + np.std(counts1), alpha=0.3, color='red', label=r'1
$\sigma$ range')

plt.axvspan(np.mean(counts1) -  2*np.std(counts1),
            np.mean(counts1) + 2*np.std(counts1), alpha=0.3, color='red', label=r
'2 $\sigma$ range')

print (np.mean(counts) / expected_number)

plt.xlabel("Number of Objects in Sample")
plt.ylabel("Number of Counts")
plt.legend()
```

```
0.9834063875409097
```

Out[193]:

```
<matplotlib.legend.Legend at 0x126a3d978>
```



Z of 0.50

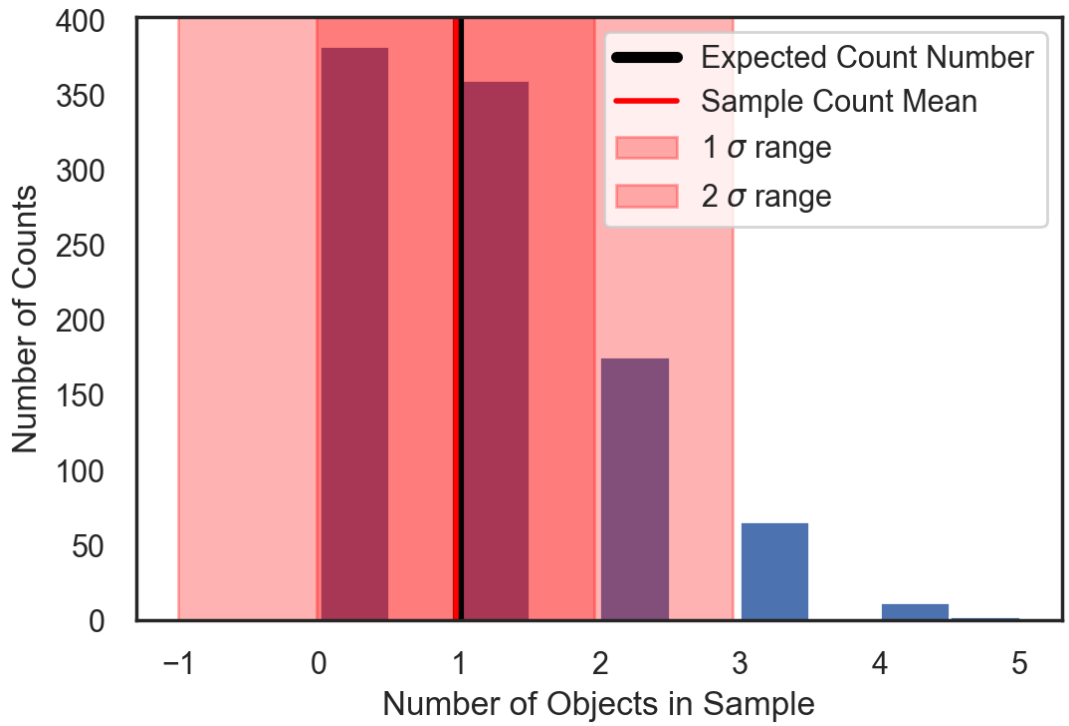
In [194]:

```
#expected_number2, counts2 = get_cut_for_z(1000, 0.5, 0.5)

plt.hist(counts2)
plt.axvline(x=expected_number2, linewidth=4, color='black', label=r'Expected Cou
nt Number')
plt.axvline(x=np.mean(counts2), linewidth=2, color='red', label=r'Sample Count M
ean')
plt.axvspan(np.mean(counts2) - np.std(counts2),
            np.mean(counts2) + np.std(counts2), alpha=0.3, color='red', label=r'1
$\sigma$ range')

plt.axvspan(np.mean(counts2) -  2*np.std(counts2),
            np.mean(counts2) + 2*np.std(counts2), alpha=0.3, color='red', label=r
'2 $\sigma$ range')

print (np.mean(counts2) / expected_number2)

plt.xlabel("Number of Objects in Sample")
plt.ylabel("Number of Counts")
plt.legend()
```

```
0.98582948756368
```

Out[194]:

```
<matplotlib.legend.Legend at 0x15cd44518>
```



Z of up to 0.75

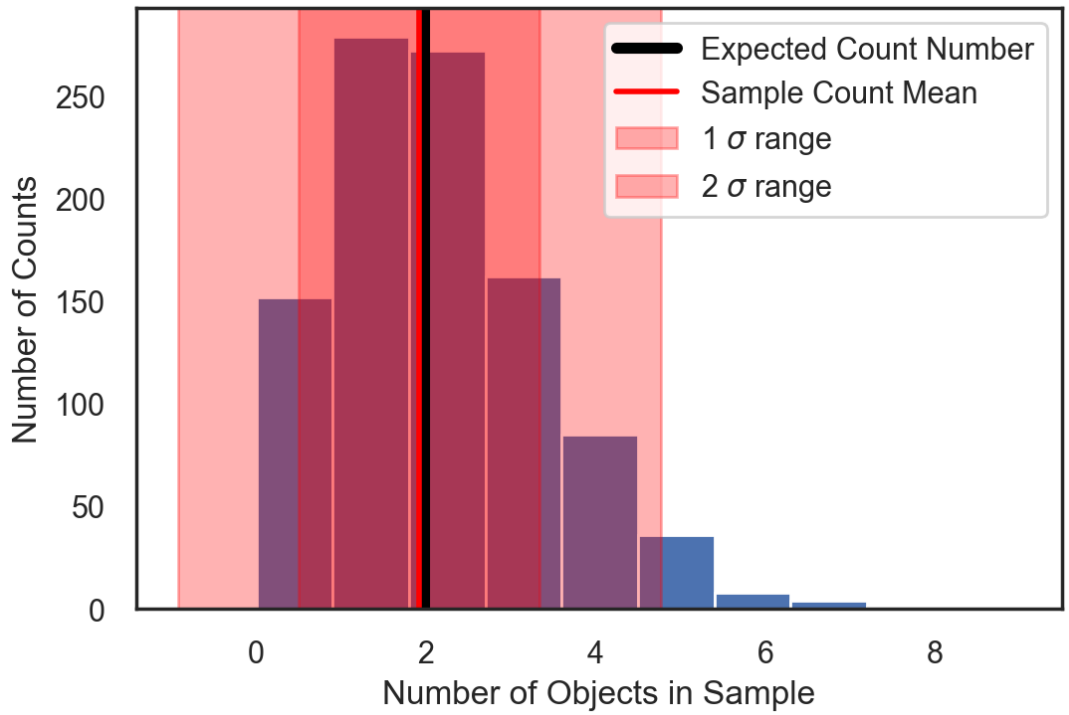
In [195]:

```
#expected_number3, counts3 = get_cut_for_z(1000, 0.75, 0.5)

plt.hist(counts3)
plt.axvline(x=expected_number3, linewidth=4, color='black', label=r'Expected Cou
nt Number')
plt.axvline(x=np.mean(counts3), linewidth=2, color='red', label=r'Sample Count M
ean')
plt.axvspan(np.mean(counts3) - np.std(counts3),
            np.mean(counts3) + np.std(counts3), alpha=0.3, color='red', label=r'1
$\sigma$ range')

plt.axvspan(np.mean(counts3) -  2*np.std(counts3),
            np.mean(counts3) + 2*np.std(counts3), alpha=0.3, color='red', label=r
'2 $\sigma$ range')

print (np.mean(counts3) / expected_number3)

plt.xlabel("Number of Objects in Sample")
plt.ylabel("Number of Counts")
plt.legend()
```

```
0.9717783759533424
```

Out[195]:

```
<matplotlib.legend.Legend at 0x15c897c50>
```



Z up to 1.00

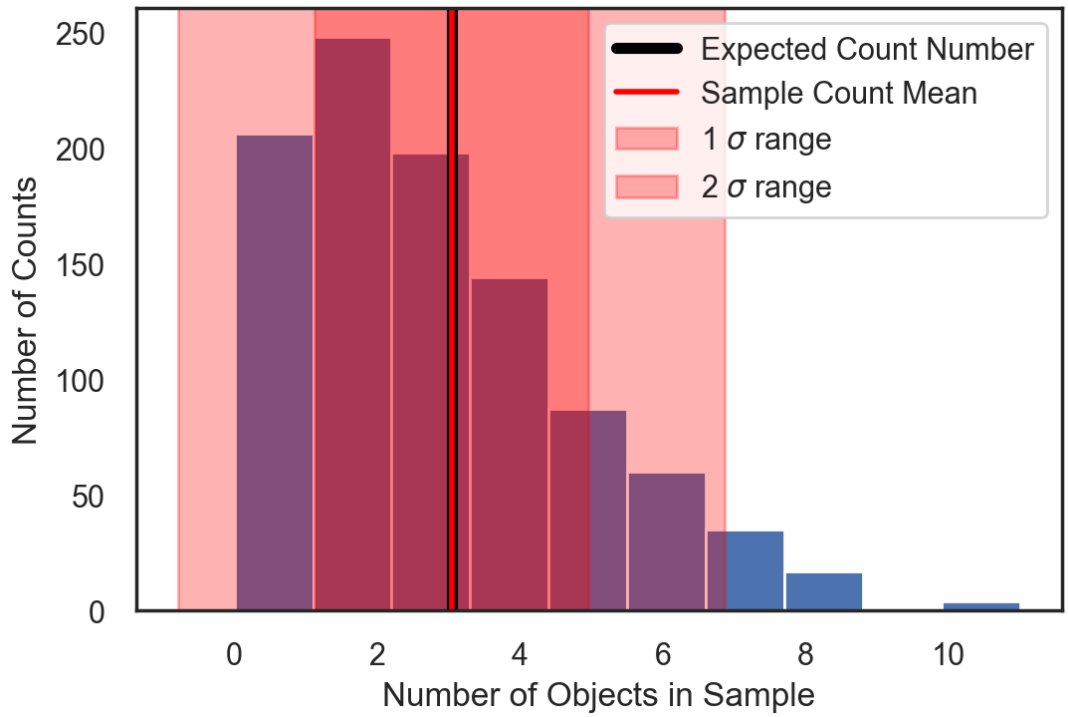
In [196]:

```
#expected_number4, counts4 = get_cut_for_z(1000, 1.0, 0.5)

plt.hist(counts4)
plt.axvline(x=expected_number4, linewidth=4, color='black', label=r'Expected Cou
nt Number')
plt.axvline(x=np.mean(counts4), linewidth=2, color='red', label=r'Sample Count M
ean')
plt.axvspan(np.mean(counts4) - np.std(counts4),
            np.mean(counts4) + np.std(counts4), alpha=0.3, color='red', label=r'1
$\sigma$ range')

plt.axvspan(np.mean(counts4) -  2*np.std(counts4),
            np.mean(counts4) + 2*np.std(counts4), alpha=0.3, color='red', label=r
'2 $\sigma$ range')

print (np.mean(counts4) / expected_number4)

plt.xlabel("Number of Objects in Sample")
plt.ylabel("Number of Counts")
plt.legend()
```

```
1.0002179836512264
```

Out[196]:

```
<matplotlib.legend.Legend at 0x15c9287b8>
```



Z up to 1.5

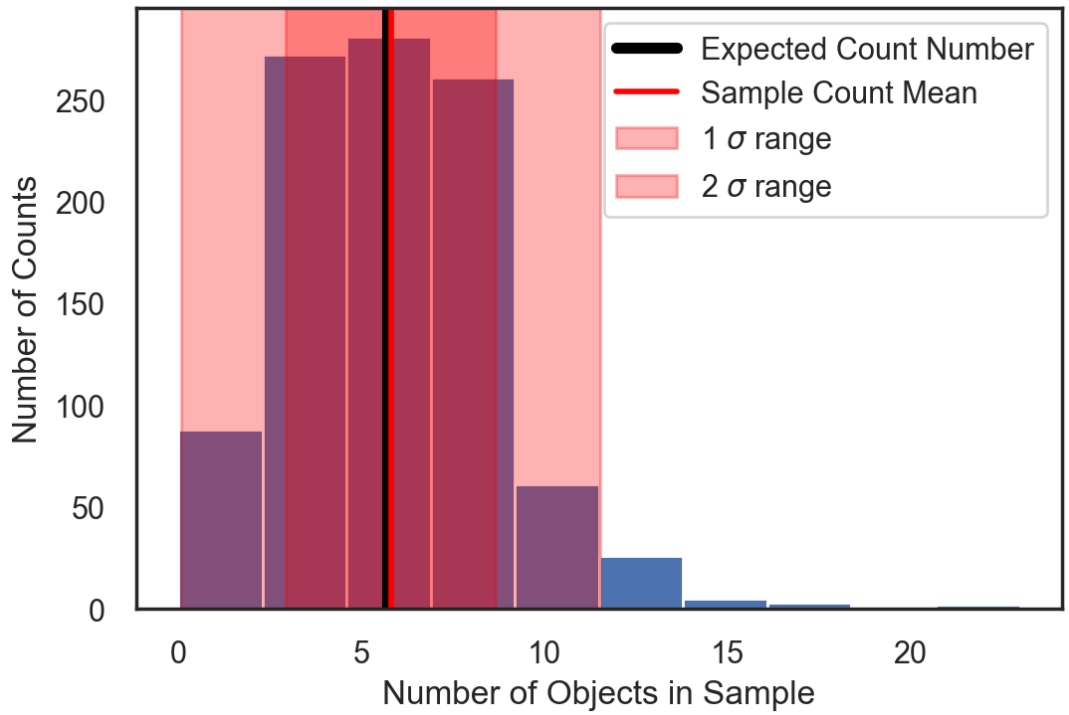
In [197]:

```
#expected_number5, counts5 = get_cut_for_z(1000, 1.5, 0.5)

plt.hist(counts5)
plt.axvline(x=expected_number5, linewidth=4, color='black', label=r'Expected Cou
nt Number')
plt.axvline(x=np.mean(counts5), linewidth=2, color='red', label=r'Sample Count M
ean')
plt.axvspan(np.mean(counts5) - np.std(counts5),
            np.mean(counts5) + np.std(counts5), alpha=0.3, color='red', label=r'1
$\sigma$ range')

plt.axvspan(np.mean(counts5) -  2*np.std(counts5),
            np.mean(counts5) + 2*np.std(counts5), alpha=0.3, color='red', label=r
'2 $\sigma$ range')

print (np.mean(counts5) / expected_number5)

plt.xlabel("Number of Objects in Sample")
plt.ylabel("Number of Counts")
plt.legend()
```

```
1.0175740251670033
```

Out[197]:

```
<matplotlib.legend.Legend at 0x11ab4cdd8>
```



Z up to 2.0

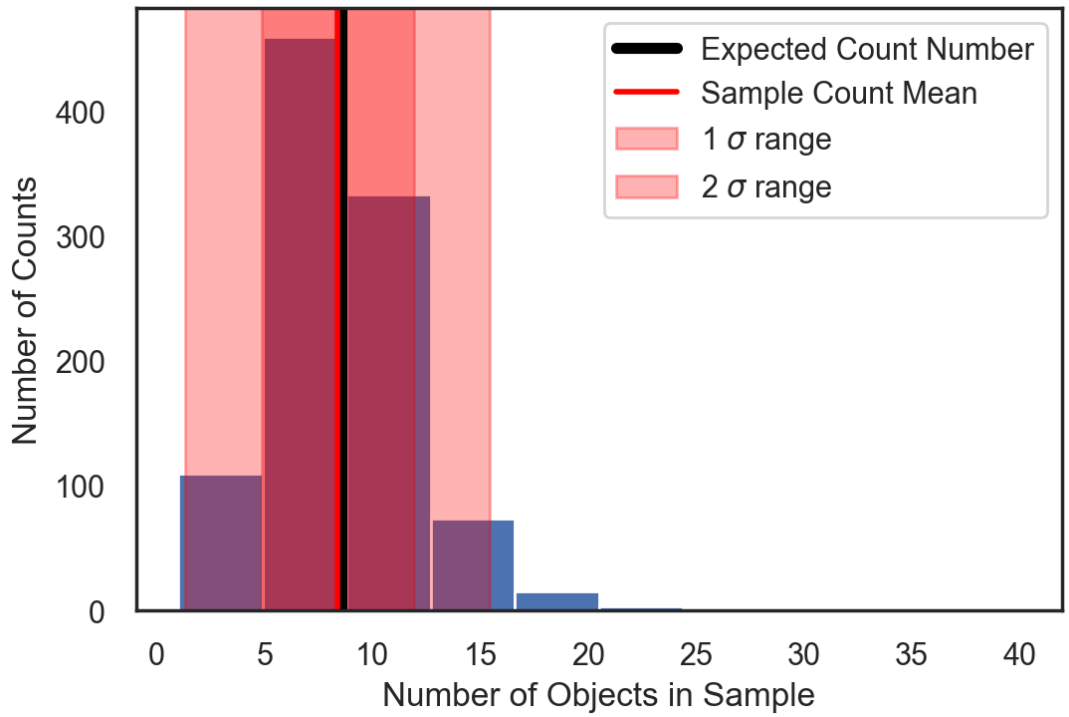
In [198]:

```
#expected_number6, counts6 = get_cut_for_z(1000, 2.0, 0.5)

plt.hist(counts6)
plt.axvline(x=expected_number6, linewidth=4, color='black', label=r'Expected Cou
nt Number')
plt.axvline(x=np.mean(counts6), linewidth=2, color='red', label=r'Sample Count M
ean')
plt.axvspan(np.mean(counts6) - np.std(counts6),
            np.mean(counts6) + np.std(counts6), alpha=0.3, color='red', label=r'1
$\sigma$ range')

plt.axvspan(np.mean(counts6) -  2*np.std(counts6),
            np.mean(counts6) + 2*np.std(counts6), alpha=0.3, color='red', label=r
'2 $\sigma$ range')

print (np.mean(counts6) / expected_number6)

plt.xlabel("Number of Objects in Sample")
plt.ylabel("Number of Counts")
plt.legend()
```

```
0.9817013418176789
```

Out[198]:

```
<matplotlib.legend.Legend at 0x12613c908>
```



Z up to 3

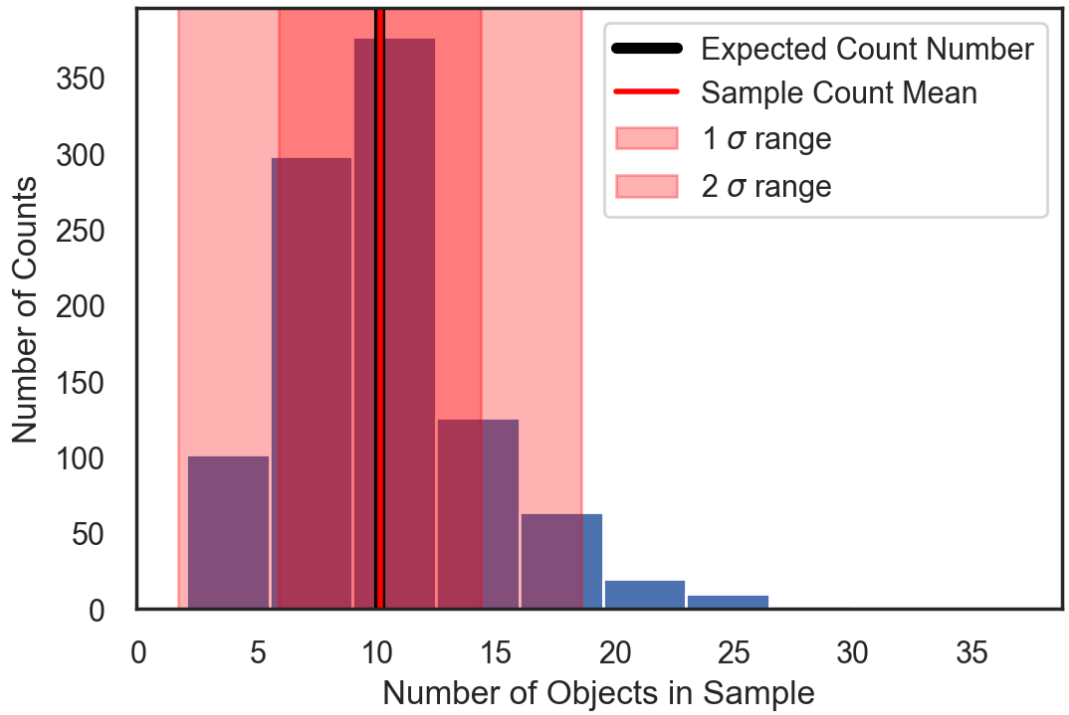
In [199]:

```
#expected_number7, counts7 = get_cut_for_z(1000, 3.0, 0.5)

plt.hist(counts7)
plt.axvline(x=expected_number7, linewidth=4, color='black', label=r'Expected Cou
nt Number')
plt.axvline(x=np.mean(counts7), linewidth=2, color='red', label=r'Sample Count M
ean')
plt.axvspan(np.mean(counts7) - np.std(counts7),
            np.mean(counts7) + np.std(counts7), alpha=0.3, color='red', label=r'1
$\sigma$ range')

plt.axvspan(np.mean(counts7) -  2*np.std(counts7),
            np.mean(counts7) + 2*np.std(counts7), alpha=0.3, color='red', label=r
'2 $\sigma$ range')

print (np.mean(counts7) / expected_number7)

plt.xlabel("Number of Objects in Sample")
plt.ylabel("Number of Counts")
plt.legend()
```

```
1.003092975629736
```

Out[199]:

```
<matplotlib.legend.Legend at 0x183a42208>
```



Z up to 4

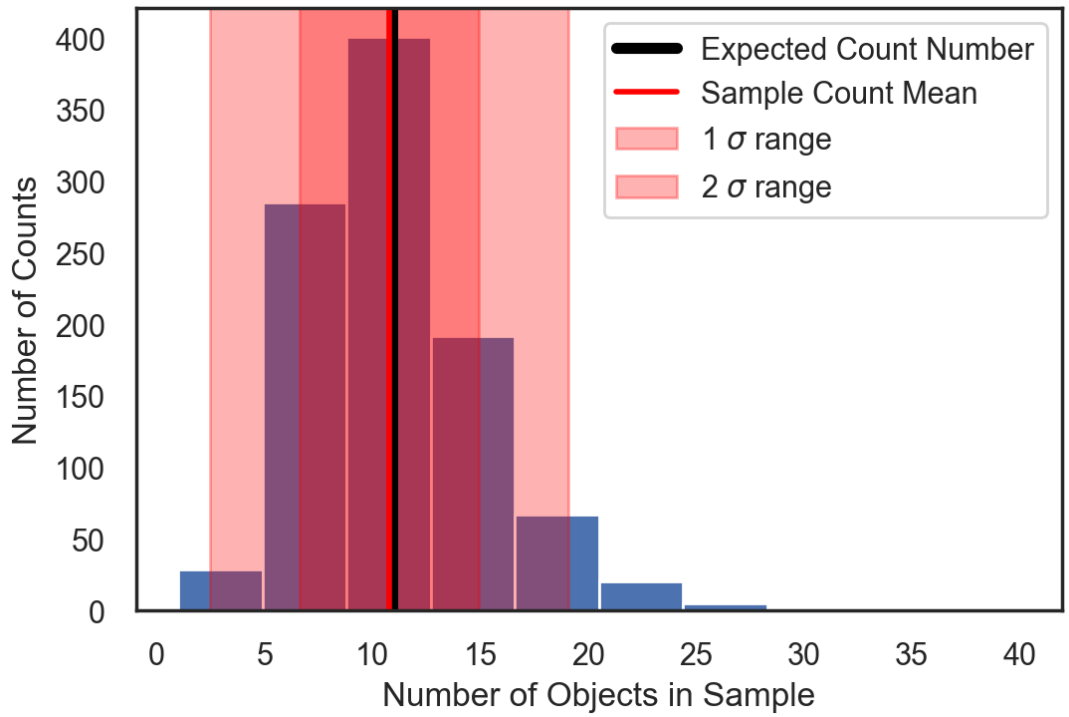
In [200]:

```
#expected_number8, counts8 = get_cut_for_z(1000, 4.0, 0.5)

plt.hist(counts8)
plt.axvline(x=expected_number8, linewidth=4, color='black', label=r'Expected Cou
nt Number')
plt.axvline(x=np.mean(counts8), linewidth=2, color='red', label=r'Sample Count M
ean')
plt.axvspan(np.mean(counts8) - np.std(counts8),
            np.mean(counts8) + np.std(counts8), alpha=0.3, color='red', label=r'1
$\sigma$ range')

plt.axvspan(np.mean(counts8) -  2*np.std(counts8),
            np.mean(counts8) + 2*np.std(counts8), alpha=0.3, color='red', label=r
'2 $\sigma$ range')

print (np.mean(counts8) / expected_number8)

plt.xlabel("Number of Objects in Sample")
plt.ylabel("Number of Counts")
plt.legend()
plt.legend()
```

```
0.9860104860998212
```

Out[200]:

```
<matplotlib.legend.Legend at 0x123f872e8>
```



Sampled vs Expected Count Rate as a Function of Luminosity Distance

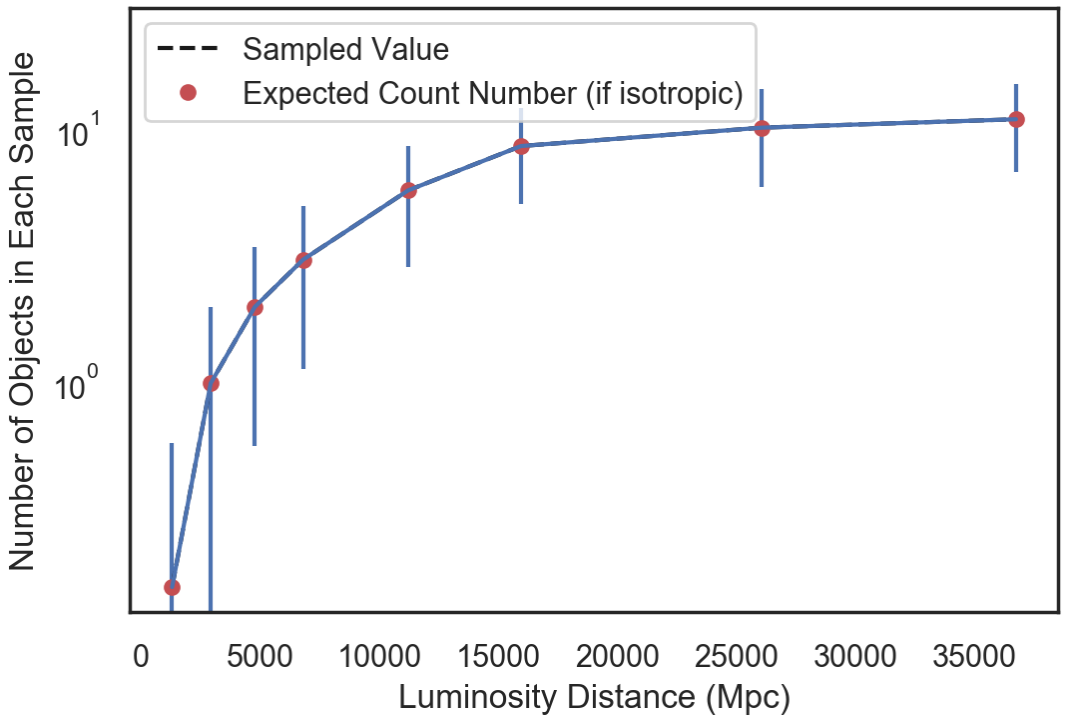
In [170]:

```
from astropy.visualization import quantity_support
quantity_support()

z_values             = [0.25, 0.50, 0.75, 1.00, 1.50, 2.00, 3.00, 4.00]
luminosity_distances = []

expected_numbers = [expected_number1,
                    expected_number2,
                    expected_number3,
                    expected_number4,
                    expected_number5,
                    expected_number6,
                    expected_number7,
                    expected_number8]


counted_numbers = [np.mean(counts1),
                   np.mean(counts2),
                   np.mean(counts3),
                   np.mean(counts4),
                   np.mean(counts5),
                   np.mean(counts6),
                   np.mean(counts7),
                   np.mean(counts8)]

errors = [np.std(counts1),
          np.std(counts2),
          np.std(counts3),
          np.std(counts4),
          np.std(counts5),
          np.std(counts6),
          np.std(counts7),
          np.std(counts8)]

cosmo = FlatLambdaCDM(H0=67.6 * u.km / u.s / u.Mpc, Tcmb0=2.725 * u.K, Om0=0.31)

for i in range(len(z_values)):
    luminosity_distances.append(cosmo.luminosity_distance(z_values[i]) / u.Mpc)

plt.errorbar(luminosity_distances, expected_numbers, yerr=errors)
plt.plot(luminosity_distances, expected_numbers, 'k--', label='Sampled Value')
plt.plot(luminosity_distances, expected_numbers, 'r.',markersize=10, label='Expe
cted Count Number (if isotropic)')
plt.ylabel(r'Number of Objects in Each Sample', size=12)
plt.xlabel(r'Luminosity Distance (Mpc)')
plt.legend(loc='upper left')
plt.yscale('log')
plt.ylim(0, 30)
```

```
/Users/imalsky/.conda/envs/py36/lib/python3.6/site-packages/ipykerne
l_launcher.py:47: UserWarning: Attempted to set non-positive bottom
ylim on a log-scaled axis.
Invalid limit will be ignored.
```

Out[170]:

```
(0.12278773392570999, 30)
```



Discussion and Results analysis

These tests show that on scales of thousands of Mpc, the universe is istropic. The solid angles of the samples were randomly chosen, but all were within 1 standard deviation of the expected value. If the Universe were no isotropic, some direction would have had lower or higher count rates, and this would have been reflected in the resulting distributions. However, I only showed isotropy in the small solid angle range covered by the survey.

For the smallest z shift the expected number of counts (assuming isotropy) were equal to the sampled count, as shown in the histogram, However, this is merely due to the majority of the the samples having no objects in them. The larger values of z show histrograms where the Universe appears far more isotropic. A perfectly isotropic Universe would have every single solid angle sample return the same number of quasars. However, due to the scarcity of quasars in the Universe and the finite number of objects in each sample, I would not expect perfect isotropy.

By z shifts of approximately 2, the sample looked Guassian, with an isotropic distribution of quasars. By this luminosity distance, there were a sufficient number of quasars in the sampled solid angle that the total area of the sky became statsistically isotropic.

My results of an isotropic Universe are in line with the expectation from Cosmology. However, I found that the Universe looks isotropic on very very large scales, on the order of thousands of Mpc. Other researches have found that the scale of isotropy is far small. For one example, see Sarkar et al (2018), who showd that on scales as small as several hundred Mpc only small statistical anisotropies remained.

Bibliography

Ahumada et al. 2019, The Sixteenth Data Release of the Sloan Digital Sky Surveys: (Ahumada et al. submitted to ApJS)

Aghanim et al 2019. "Planck 2018 results. I. Overview, and the cosmological legacy of Planck", Astronomy and Astrophysics, 641: A1

Blanton, M. R., & Moustakas, J. 2009, ARA&A, 47, 159,doi: 10.1146/annurev-astro-082708-101734

Dawson, K. S., Kneib, J.-P., Percival, W. J., et al. 2016,The Astronomical Journal, 151, 44,doi: 10.3847/0004-6256/151/2/44

DJ et al. Fixsen. The cosmic microwave background spectrum from the full cobe firas data set. The Astrophysical Journal, 473(2):576, 1996. arXiv:astro-ph/9605054v1

Lyke, B. W., Higley, A. N., McLane, J. N., et al. 2020, TheAstrophysical Journal Supplement Series, 250, 8, doi: 10.3847/1538-4365/aba623

Penzias, A. A., & Wilson, R. W. 1965, ApJ, 142, 419

Richards, G. T., Strauss, M. A., Fan, X., et al. 2006, TheAstronomical Journal, 131, 2766–2787,doi: 10.1086/503559

Sarkar, S., Pandey, B., & Khatri, R. 2018, Monthly Noticesof the Royal Astronomical Society, 483, 2453–2464,doi: 10.1093/mnras/sty3272

In [ ]: