Lightcurves for spherical outflow

We use this notebook to generate plots for the spherical outflow analysis. This notebook looks at lightcurves at different resolutions.

First we import necessary libraries

In [1]:
```
%matplotlib notebook
import processmcrat as pm
import astropy.units as unit
from astropy import constants as const
import matplotlib.pyplot as plt
import matplotlib as mpl
import numpy as np
```

We lead the MCRaT output files, and set our mock observations to be $\theta_{\rm obs} = 1^\circ$, $\Delta\theta = 4^\circ$, $r_{\rm obs} = 10^{14}$ cm and framerate = 5 fps. The spectral fit energy range is $10^{-2} - 4 \times 10^{4}$ keV. We set the lightcurve dictionaries to have a 0.2 s spacing in time.

In [2]:
```
mcrat_sim5_5=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/spatial-res-levs/fi
mcrat_sim5_5.load_frame(2638, read_stokes=False)
observation5_5=pm.MockObservation(1, 4, 1e14, 5, mcratsimload_obj=mcrat_sim5_5)
observation5_5.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict5_5=observation5_5.lightcurve(observation5_5.detected_photons.detectio
                                             observation5_5.detected_photons.detection_time.ma

mcrat_sim5_4=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/spatial-res-levs/fi
mcrat_sim5_4.load_frame(2638, read_stokes=False)
observation5_4=pm.MockObservation(1, 4, 1e14, 5, mcratsimload_obj=mcrat_sim5_4)
observation5_4.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict5_4=observation5_4.lightcurve(observation5_4.detected_photons.detectio
                                            observation5_4.detected_photons.detection_time.ma

mcrat_sim5_3=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/spatial-res-levs/fi
mcrat_sim5_3.load_frame(2638, read_stokes=False)
observation5_3=pm.MockObservation(1, 4, 1e14, 5, mcratsimload_obj=mcrat_sim5_3)
observation5_3.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict5_3=observation5_3.lightcurve(observation5_3.detected_photons.detectio
                                            observation5_3.detected_photons.detection_time.ma

mcrat_sim5_2=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/spatial-res-levs/fi
mcrat_sim5_2.load_frame(2638, read_stokes=False)
observation5_2=pm.MockObservation(1, 4, 1e14, 5, mcratsimload_obj=mcrat_sim5_2)
observation5_2.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict5_2=observation5_2.lightcurve(observation5_2.detected_photons.detectio
                                            observation5_2.detected_photons.detection_time.ma

mcrat_sim5_1=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/spatial-res-levs/fi
mcrat_sim5_1.load_frame(2638, read_stokes=False)
observation5_1=pm.MockObservation(1, 4, 1e14, 5, mcratsimload_obj=mcrat_sim5_1)
observation5_1.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict5_1=observation5_1.lightcurve(observation5_1.detected_photons.detectio
                                            observation5_1.detected_photons.detection_time.ma

mcrat_sim4_5=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/temporal-res-levs/f
mcrat_sim4_5.load_frame(1319, read_stokes=False)
observation4_5=pm.MockObservation(1, 4, 1e14, 2.5, mcratsimload_obj=mcrat_sim4_5)
observation4_5.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict4_5=observation4_5.lightcurve(observation4_5.detected_photons.detectio
                                            observation4_5.detected_photons.detection_time.ma


mcrat_sim4_4=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/mixed-res-levs/final
mcrat_sim4_4.load_frame(1319, read_stokes=False)
observation4_4=pm.MockObservation(1, 4, 1e14, 2.5, mcratsimload_obj=mcrat_sim4_4)
observation4_4.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict4_4=observation4_4.lightcurve(observation4_4.detected_photons.detectio
                                            observation4_4.detected_photons.detection_time.ma

mcrat_sim3_5=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/temporal-res-levs/f
mcrat_sim3_5.load_frame(659, read_stokes=False)
observation3_5=pm.MockObservation(1, 4, 1e14, 1.25, mcratsimload_obj=mcrat_sim3_5)
observation3_5.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict3_5=observation3_5.lightcurve(observation3_5.detected_photons.detectio
                                            observation3_5.detected_photons.detection_time.ma

mcrat_sim3_3=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/mixed-res-levs/final
mcrat_sim3_3.load_frame(659, read_stokes=False)
observation3_3=pm.MockObservation(1, 4, 1e14, 1.25, mcratsimload_obj=mcrat_sim3_3)
observation3_3.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict3_3=observation3_3.lightcurve(observation3_3.detected_photons.detectio
                                            observation3_3.detected_photons.detection_time.ma


mcrat_sim2_5=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/temporal-res-levs/f
mcrat_sim2_5.load_frame(329, read_stokes=False)
observation2_5=pm.MockObservation(1, 4, 1e14, 0.625, mcratsimload_obj=mcrat_sim2_5)
observation2_5.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict2_5=observation2_5.lightcurve(observation2_5.detected_photons.detectio
                                            observation2_5.detected_photons.detection_time.ma

mcrat_sim2_2=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/mixed-res-levs/final
mcrat_sim2_2.load_frame(329, read_stokes=False)
observation2_2=pm.MockObservation(1, 4, 1e14, 0.625, mcratsimload_obj=mcrat_sim2_2)
observation2_2.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict2_2=observation2_2.lightcurve(observation2_2.detected_photons.detectio
                                            observation2_2.detected_photons.detection_time.ma

mcrat_sim1_5=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/temporal-res-levs/f
mcrat_sim1_5.load_frame(164, read_stokes=False)
observation1_5=pm.MockObservation(1, 4, 1e14, 0.3125, mcratsimload_obj=mcrat_sim1_5)
observation1_5.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict1_5=observation1_5.lightcurve(observation1_5.detected_photons.detectio
                                            observation1_5.detected_photons.detection_time.ma

mcrat_sim1_1=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/mixed-res-levs/final
mcrat_sim1_1.load_frame(164, read_stokes=False)
observation1_1=pm.MockObservation(1, 4, 1e14, 0.3125, mcratsimload_obj=mcrat_sim1_1)
observation1_1.set_spectral_fit_parameters(spectral_fit_energy_range=[0.01, 40000.0])
lightcurve_dict1_1=observation1_1.lightcurve(observation1_1.detected_photons.detectio
                                            observation1_1.detected_photons.detection_time.ma
```

We normalize all lightcurves to start at $t=0$ when the first photon is detected.

In [3]:
```
init_time5_1=observation5_1.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict5_1['times'])):
    lightcurve_dict5_1['times'][i]-=init_time5_1

init_time5_2=observation5_2.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict5_2['times'])):
    lightcurve_dict5_2['times'][i]-=init_time5_2

init_time5_3=observation5_3.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict5_3['times'])):
    lightcurve_dict5_3['times'][i]-=init_time5_3

init_time5_4=observation5_4.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict5_4['times'])):
    lightcurve_dict5_4['times'][i]-=init_time5_4

init_time5_5=observation5_5.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict5_5['times'])):
    lightcurve_dict5_5['times'][i]-=init_time5_5
```

In [4]:
```
init_time4_5=observation4_5.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict4_5['times'])):
    lightcurve_dict4_5['times'][i]-=init_time4_5

init_time3_5=observation3_5.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict3_5['times'])):
    lightcurve_dict3_5['times'][i]-=init_time3_5

init_time2_5=observation2_5.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict2_5['times'])):
    lightcurve_dict2_5['times'][i]-=init_time2_5

init_time1_5=observation1_5.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict1_5['times'])):
    lightcurve_dict1_5['times'][i]-=init_time1_5
```

In [5]:
```
init_time4_4=observation4_4.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict4_4['times'])):
    lightcurve_dict4_4['times'][i]-=init_time4_4

init_time3_3=observation3_3.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict3_3['times'])):
    lightcurve_dict3_3['times'][i]-=init_time3_3

init_time2_2=observation2_2.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict2_2['times'])):
    lightcurve_dict2_2['times'][i]-=init_time2_2

init_time1_1=observation1_1.detected_photons.detection_time.min()*unit.s

for i in range(len(lightcurve_dict1_1['times'])):
    lightcurve_dict1_1['times'][i]-=init_time1_1
```

We now calculate the center of the time bins.

In [6]:
```
difference5_1=np.zeros(lightcurve_dict5_1['times'].size)
difference5_1[:lightcurve_dict5_1['times'].size - 1] = np.diff(lightcurve_dict5_1['ti
difference5_1[-1] = np.diff(lightcurve_dict5_1['times']).min().value
t_cen5_1 = (lightcurve_dict5_1['times'].value + (lightcurve_dict5_1['times'].value +
x_err5_1=difference5_1/2

difference5_2=np.zeros(lightcurve_dict5_2['times'].size)
difference5_2[:lightcurve_dict5_2['times'].size - 1] = np.diff(lightcurve_dict5_2['ti
difference5_2[-1] = np.diff(lightcurve_dict5_2['times']).min().value
t_cen5_2 = (lightcurve_dict5_2['times'].value + (lightcurve_dict5_2['times'].value +
x_err5_2=difference5_2/2

difference5_3=np.zeros(lightcurve_dict5_3['times'].size)
difference5_3[:lightcurve_dict5_3['times'].size - 1] = np.diff(lightcurve_dict5_3['ti
difference5_3[-1] = np.diff(lightcurve_dict5_3['times']).min().value
t_cen5_3 = (lightcurve_dict5_3['times'].value + (lightcurve_dict5_3['times'].value +
x_err5_3=difference5_3/2

difference5_4=np.zeros(lightcurve_dict5_4['times'].size)
difference5_4[:lightcurve_dict5_4['times'].size - 1] = np.diff(lightcurve_dict5_4['ti
difference5_4[-1] = np.diff(lightcurve_dict5_4['times']).min().value
t_cen5_4 = (lightcurve_dict5_4['times'].value + (lightcurve_dict5_4['times'].value +
x_err5_4=difference5_4/2

difference5_5=np.zeros(lightcurve_dict5_5['times'].size)
difference5_5[:lightcurve_dict5_5['times'].size - 1] = np.diff(lightcurve_dict5_5['ti
difference5_5[-1] = np.diff(lightcurve_dict5_5['times']).min().value
t_cen5_5 = (lightcurve_dict5_5['times'].value + (lightcurve_dict5_5['times'].value +
x_err5_5=difference5_5/2
```

In [7]:
```
difference4_4=np.zeros(lightcurve_dict4_4['times'].size)
difference4_4[:lightcurve_dict4_4['times'].size - 1] = np.diff(lightcurve_dict4_4['ti
difference4_4[-1] = np.diff(lightcurve_dict4_4['times']).min().value
t_cen4_4 = (lightcurve_dict4_4['times'].value + (lightcurve_dict4_4['times'].value +
x_err4_4=difference4_4/2

difference3_3=np.zeros(lightcurve_dict3_3['times'].size)
difference3_3[:lightcurve_dict3_3['times'].size - 1] = np.diff(lightcurve_dict3_3['ti
difference3_3[-1] = np.diff(lightcurve_dict3_3['times']).min().value
t_cen3_3 = (lightcurve_dict3_3['times'].value + (lightcurve_dict3_3['times'].value +
x_err3_3=difference3_3/2

difference2_2=np.zeros(lightcurve_dict2_2['times'].size)
difference2_2[:lightcurve_dict2_2['times'].size - 1] = np.diff(lightcurve_dict2_2['ti
difference2_2[-1] = np.diff(lightcurve_dict2_2['times']).min().value
t_cen2_2 = (lightcurve_dict2_2['times'].value + (lightcurve_dict2_2['times'].value +
x_err2_2=difference2_2/2

difference1_1=np.zeros(lightcurve_dict1_1['times'].size)
difference1_1[:lightcurve_dict1_1['times'].size - 1] = np.diff(lightcurve_dict1_1['ti
difference1_1[-1] = np.diff(lightcurve_dict1_1['times']).min().value
t_cen1_1 = (lightcurve_dict1_1['times'].value + (lightcurve_dict1_1['times'].value +
x_err1_1=difference1_1/2
```

In [8]:
```
difference1_5=np.zeros(lightcurve_dict1_5['times'].size)
difference1_5[:lightcurve_dict1_5['times'].size - 1] = np.diff(lightcurve_dict1_5['ti
difference1_5[-1] = np.diff(lightcurve_dict1_5['times']).min().value
t_cen1_5 = (lightcurve_dict1_5['times'].value + (lightcurve_dict1_5['times'].value +
x_err1_5=difference1_5/2

difference2_5=np.zeros(lightcurve_dict2_5['times'].size)
difference2_5[:lightcurve_dict2_5['times'].size - 1] = np.diff(lightcurve_dict2_5['ti
difference2_5[-1] = np.diff(lightcurve_dict2_5['times']).min().value
t_cen2_5 = (lightcurve_dict2_5['times'].value + (lightcurve_dict2_5['times'].value +
x_err2_5=difference2_5/2

difference3_5=np.zeros(lightcurve_dict3_5['times'].size)
difference3_5[:lightcurve_dict3_5['times'].size - 1] = np.diff(lightcurve_dict3_5['ti
difference3_5[-1] = np.diff(lightcurve_dict3_5['times']).min().value
t_cen3_5 = (lightcurve_dict3_5['times'].value + (lightcurve_dict3_5['times'].value +
x_err3_5=difference3_5/2

difference4_5=np.zeros(lightcurve_dict4_5['times'].size)
difference4_5[:lightcurve_dict4_5['times'].size - 1] = np.diff(lightcurve_dict4_5['ti
difference4_5[-1] = np.diff(lightcurve_dict4_5['times']).min().value
t_cen4_5 = (lightcurve_dict4_5['times'].value + (lightcurve_dict4_5['times'].value +
x_err4_5=difference4_5/2
```

We now sort them on the type of resolution that is changed.

In [9]:
```
levs=["Spatial Level 1","Spatial Level 2","Spatial Level 3","Spatial Level 4","Spatia

fps=["0.3125 fps","0.625 fps","1.25 fps","2.5 fps","5 fps"]

mix=["Level 1, 0.3125 fps","Level 2, 0.625 fps","Level 3, 1.25 fps","Level 4, 2.5 fps

colors=['b','r','g','y','k']

lightcurve_dict_spatial=[lightcurve_dict5_5, lightcurve_dict5_4,
                         lightcurve_dict5_3, lightcurve_dict5_2, lightcurve_dict5_1]

lightcurve_dict_temporal=[lightcurve_dict5_5, lightcurve_dict4_5,
                          lightcurve_dict3_5, lightcurve_dict2_5, lightcurve_dict1_5]

lightcurve_dict_mixed=[lightcurve_dict5_5, lightcurve_dict4_4,
                       lightcurve_dict3_3, lightcurve_dict2_2, lightcurve_dict1_1]
```

We now plot the lightcurves

In [10]:
```
plt.rcParams.update({'font.size': 20})
    #see how many panels we need for the plot and how many light curves the user want
num_panels=3

formatter = mpl.ticker.ScalarFormatter(useMathText=True)
formatter.set_scientific(True)
formatter.set_powerlimits((0, 1))


f, axarr = plt.subplots(3, sharex=True)

    #decide which panels will plot what based on input
lc_panel = axarr

f.set_figwidth(12)
f.set_figheight(17)

formatter = mpl.ticker.ScalarFormatter(useMathText=True)
formatter.set_scientific(True)
formatter.set_powerlimits((0, 1))

axarr[0].yaxis.set_major_formatter(formatter)

axarr[1].yaxis.set_major_formatter(formatter)

axarr[2].yaxis.set_major_formatter(formatter)



#lc_panel.plot(main_lightcurve_dict['times'],main_lightcurve_dict['lightcurve'],ds='s
        #if 'ct' in spectrum_dict['lightcurve'].unit.to_string():
lc_panel[0].set_ylabel(r'L$_\mathrm{iso}$ ('+
  lightcurve_dict5_5['lightcurve'].unit.to_string('latex_inline')+')')

lc_panel[1].set_ylabel(r'L$_\mathrm{iso}$ ('+
  lightcurve_dict5_5['lightcurve'].unit.to_string('latex_inline')+')')

lc_panel[2].set_ylabel(r'L$_\mathrm{iso}$ ('+
  lightcurve_dict5_5['lightcurve'].unit.to_string('latex_inline')+')')


lc_panel[2].set_xlabel('Time since first photon detection (s)')


lc_panel[0].plot(lightcurve_dict5_5['times'],lightcurve_dict5_5['lightcurve'],
  ds='steps-post', color='k',lw=1.5, label=levs[4] ,zorder=5)

lc_panel[0].errorbar(t_cen5_5,lightcurve_dict5_5['lightcurve'],
  yerr = lightcurve_dict5_5['lightcurve_errors'], ls='none', color='k',zorder=5)

lc_panel[0].plot(lightcurve_dict5_4['times'],lightcurve_dict5_4['lightcurve'],
  ds='steps-post', color='r',lw=1.5, label=levs[3],zorder=4)

lc_panel[0].errorbar(t_cen5_4,lightcurve_dict5_4['lightcurve'],
  yerr = lightcurve_dict5_4['lightcurve_errors'], ls='none', color='r',zorder=4)

lc_panel[0].plot(lightcurve_dict5_3['times'],lightcurve_dict5_3['lightcurve'],
  ds='steps-post', color='blue',lw=1.5, label=levs[2],zorder=3)

lc_panel[0].errorbar(t_cen5_3,lightcurve_dict5_3['lightcurve'],
  yerr = lightcurve_dict5_3['lightcurve_errors'], ls='none', color='blue',zorder=3)

lc_panel[0].plot(lightcurve_dict5_2['times'],lightcurve_dict5_2['lightcurve'],
  ds='steps-post', color='cyan',lw=1.5, label=levs[1],zorder=2)

lc_panel[0].errorbar(t_cen5_2,lightcurve_dict5_2['lightcurve'],
  yerr = lightcurve_dict5_2['lightcurve_errors'], ls='none', color='cyan',zorder=2)

lc_panel[0].plot(lightcurve_dict5_1['times'],lightcurve_dict5_1['lightcurve'],
  ds='steps-post', color='green',lw=1.5, label=levs[0],zorder=1)

lc_panel[0].errorbar(t_cen5_1,lightcurve_dict5_1['lightcurve'],
  yerr = lightcurve_dict5_1['lightcurve_errors'], ls='none', color='green',zorder=1)

lc_panel[1].plot(lightcurve_dict5_5['times'],lightcurve_dict5_5['lightcurve'],
  ds='steps-post', color='k',lw=1.5, label=fps[4],zorder=5)

lc_panel[1].errorbar(t_cen5_5,lightcurve_dict5_5['lightcurve'],
  yerr = lightcurve_dict5_5['lightcurve_errors'], ls='none', color='k',zorder=5)

lc_panel[1].plot(lightcurve_dict4_5['times'],lightcurve_dict4_5['lightcurve'],
  ds='steps-post', color='r',lw=1.5, label=fps[3],zorder=4)

lc_panel[1].errorbar(t_cen4_5,lightcurve_dict4_5['lightcurve'],
  yerr = lightcurve_dict4_5['lightcurve_errors'], ls='none', color='r',zorder=4)

lc_panel[1].plot(lightcurve_dict3_5['times'],lightcurve_dict3_5['lightcurve'],
  ds='steps-post', color='blue',lw=1.5, label=fps[2],zorder=3)

lc_panel[1].errorbar(t_cen3_5,lightcurve_dict3_5['lightcurve'],
  yerr = lightcurve_dict3_5['lightcurve_errors'], ls='none', color='blue',zorder=3)

lc_panel[1].plot(lightcurve_dict2_5['times'],lightcurve_dict2_5['lightcurve'],
  ds='steps-post', color='cyan',lw=1.5, label=fps[1],zorder=2)

lc_panel[1].errorbar(t_cen2_5,lightcurve_dict2_5['lightcurve'],
  yerr = lightcurve_dict2_5['lightcurve_errors'], ls='none', color='cyan',zorder=2)

lc_panel[1].plot(lightcurve_dict1_5['times'],lightcurve_dict1_5['lightcurve'],
  ds='steps-post', color='green',lw=1.5, label=fps[0],zorder=1)

lc_panel[1].errorbar(t_cen1_5,lightcurve_dict1_5['lightcurve'],
  yerr = lightcurve_dict1_5['lightcurve_errors'], ls='none', color='green',zorder=1)

lc_panel[2].plot(lightcurve_dict5_5['times'],lightcurve_dict5_5['lightcurve'],
  ds='steps-post', color='k',lw=1.5, label=mix[4] ,zorder=5)

lc_panel[2].errorbar(t_cen5_5,lightcurve_dict5_5['lightcurve'],
  yerr = lightcurve_dict5_5['lightcurve_errors'], ls='none', color='k',zorder=5)

lc_panel[2].plot(lightcurve_dict4_4['times'],lightcurve_dict4_4['lightcurve'],
  ds='steps-post', color='r',lw=1.5, label=mix[3],zorder=4)

lc_panel[2].errorbar(t_cen4_4,lightcurve_dict4_4['lightcurve'],
  yerr = lightcurve_dict4_4['lightcurve_errors'], ls='none', color='r',zorder=4)

lc_panel[2].plot(lightcurve_dict3_3['times'],lightcurve_dict3_3['lightcurve'],
  ds='steps-post', color='blue',lw=1.5, label=mix[2],zorder=3)

lc_panel[2].errorbar(t_cen3_3,lightcurve_dict3_3['lightcurve'],
  yerr = lightcurve_dict3_3['lightcurve_errors'], ls='none', color='blue',zorder=3)

lc_panel[2].plot(lightcurve_dict2_2['times'],lightcurve_dict2_2['lightcurve'],
  ds='steps-post', color='cyan',lw=1.5, label=mix[1],zorder=2)

lc_panel[2].errorbar(t_cen2_2,lightcurve_dict2_2['lightcurve'],
  yerr = lightcurve_dict2_2['lightcurve_errors'], ls='none', color='cyan',zorder=2)

lc_panel[2].plot(lightcurve_dict1_1['times'],lightcurve_dict1_1['lightcurve'],
  ds='steps-post', color='green',lw=1.5, label=mix[0],zorder=1)

lc_panel[2].errorbar(t_cen1_1,lightcurve_dict1_1['lightcurve'],
  yerr = lightcurve_dict1_1['lightcurve_errors'], ls='none', color='green',zorder=1)

lc_panel[0].set_ylim(top = 15e51)

lc_panel[1].set_ylim(top = 2.7e51)

lc_panel[2].set_ylim(top = 7.3e51)

lc_panel[2].legend(loc='upper center', ncol = 2)

lc_panel[0].legend(loc='upper center', ncol = 2)

lc_panel[1].legend(loc='upper center' , ncol = 3 , bbox_to_anchor=(0.5, 1.03))

lc_panel[0].annotate('(a)',xy=(0.02, 0.9), xycoords="axes fraction")

lc_panel[1].annotate('(b)',xy=(0.02, 0.9), xycoords="axes fraction")

lc_panel[2].annotate('(c)',xy=(0.02, 0.9), xycoords="axes fraction")

#plt.title('''Lightcurves for mixed refinement levels
#spherical outflow, final frame''')

#plt.yscale('log')
#plt.legend()

#plt.savefig('lightcurves_spherical.pdf', dpi = 600)

plt.show()
```