

Luminosity for spherical outflow

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

First we import necessary libraries

```
%matplotlib notebook
```

```
import astropy.units as unit
from astropy import constants as const
import matplotlib.pyplot as plt
import matplotlib as mpl
import numpy as np
```

```
mcraat_sim5_5_load_frame(2638, read_stokes=False)
observation5_5=pm.MockObservation(1, 4, 1e14, 5, mcraatsimload_obj=mcraat_s
observation5_5.set_spectral_fit_parameters(spectral_fit_energy_range=[0.0,
```

```

mcrat_sim5_4=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/spatial-res-levs/fin
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])
spectrum_dict5_4=observation5_4.spectrum(observation5_4.detected_photons.detection_tir
    observation5_4.detected_photons.detection_time.ms)

mcrat_sim5_3=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/spatial-res-levs/fin
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])
spectrum_dict5_3=observation5_3.spectrum(observation5_3.detected_photons.detection_tir
    observation5_3.detected_photons.detection_time.ms)

mcrat_sim5_2=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/spatial-res-levs/fin
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])
spectrum_dict5_2=observation5_2.spectrum(observation5_2.detected_photons.detection_tir
    observation5_2.detected_photons.detection_time.ms)

mcrat_sim5_1=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/spatial-res-levs/fin
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])
spectrum_dict5_1=observation5_1.spectrum(observation5_1.detected_photons.detection_tir
    observation5_1.detected_photons.detection_time.ms)

mcrat_sim4_5=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/temporal-res-levs/fi
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])
spectrum_dict4_5=observation4_5.spectrum(observation4_5.detected_photons.detection_tir
    observation4_5.detected_photons.detection_time.ms)

mcrat_sim4_4=pm.McratSimLoad(
    "/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-photons/mixed-res-levs/fin
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])
spectrum_dict4_4=observation4_4.spectrum(observation4_4.detected_photons.detection_tir
    observation4_4.detected_photons.detection_time.ms)

mcrat_sim3_5=pm.McratSimLoad("/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-ph
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])
spectrum_dict3_5=observation3_5.spectrum(observation3_5.detected_photons.detection_tir
    observation3_5.detected_photons.detection_time.ms)

mcrat_sim3_3=pm.McratSimLoad("/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-ph
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])
spectrum_dict3_3=observation3_3.spectrum(observation3_3.detected_photons.detection_tir
    observation3_3.detected_photons.detection_time.ms)

mcrat_sim2_5=pm.McratSimLoad("/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-ph
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])
spectrum_dict2_5=observation2_5.spectrum(observation2_5.detected_photons.detection_tir
    observation2_5.detected_photons.detection_time.ms)

mcrat_sim2_2=pm.McratSimLoad("/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-ph
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])
spectrum_dict2_2=observation2_2.spectrum(observation2_2.detected_photons.detection_tir
    observation2_2.detected_photons.detection_time.ms)

mcrat_sim1_5=pm.McratSimLoad("/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-ph
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])
spectrum_dict1_5=observation1_5.spectrum(observation1_5.detected_photons.detection_tir
    observation1_5.detected_photons.detection_time.ms)

mcrat_sim1_1=pm.McratSimLoad("/MCRaT-resolution/CHOMBO/spherical-outflow/3000-8000-ph
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])
spectrum_dict1_1=observation1_1.spectrum(observation1_1.detected_photons.detection_tir
    observation1_1.detected_photons.detection_time.ms)

```

```
spectrum_dict5_3, spectrum
```

```
spectrum_dict3_3, spectrum_dict2_3, spectrum_dict1_3])

spectrum_dict_mixed=[spectrum_dict3_5, spectrum_dict4_4,
                      spectrum_dict3_3, spectrum_dict2_2, spectrum_dict1_1]

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

levs=[1,2,3,4,5]

mix = ['(1,0.3125)', '(2,0.625)', '(3,1.25)', '(4,2.5)', '(5,5)']
```

```
photon_num_min=10
```

```

lum = np.trapz(spectrum_dict['spectrum'] [np.where(spectrum_dict['ph_num']>photon_num_min)
spectrum_dict['energy_bin_center'] [np.where(spectrum_dict['ph_num']>photon_num_min)
return lum

lum_spatial = []

lum_temporal = []

lum_mixed = []

for i in range(1,6):
    lum_spatial.append(luminosity(spectrum_dict_spatial[-i]))

    lum_temporal.append(luminosity(spectrum_dict_temporal[-i]))

    lum_mixed.append(luminosity(spectrum_dict_mixed[-i]))

```

```

In [5]:
idx_spatial = []

idx_temporal = []

idx_mixed = []

for i in range(5):

```

```
idx_spatial.append(np.where(spectrum_dict_spatial[i]['ph_num']>photon_num_min)[0])

idx_temporal.append(np.where(spectrum_dict_temporal[i]['ph_num']>photon_num_min)[0])

idx_mixed.append(np.where(spectrum_dict_mixed[i]['ph_num']>photon_num_min)[0])
```

```
In [6]: lum_err_spatial=[]

lum_err_temporal=[]

lum_err_mixed=[]

for i in range(5):

    alpha_spatial = []

    alpha_temporal = []

    alpha_mixed = []
```

```

spec_err_spatial = spectrum_dict_spatial[i]['spectrum_errors'][idx_spatial[i]]
spec_err_mixed = spectrum_dict_mixed[i]['spectrum_errors'][idx_mixed[i]]

for j in range(len(spectrum_dict_spatial[i]['energy_bin_center'][idx_spatial[i]])):
    alpha_spatial.append((spectrum_dict_spatial[i]['energy_bin_center'][idx_spatial[i]][j])/2)
    spectrum_dict_spatial[i]['energy_bin_center'][idx_spatial[i]][j])/(2)

for j in range(len(spectrum_dict_temporal[i]['energy_bin_center'][idx_temporal[i]])):
    alpha_temporal.append((spectrum_dict_temporal[i]['energy_bin_center'][idx_temporal[i]][j])/2)
    spectrum_dict_temporal[i]['energy_bin_center'][idx_temporal[i]][j])/(2)

for j in range(len(spectrum_dict_mixed[i]['energy_bin_center'][idx_mixed[i]])-1):
    alpha_mixed.append((spectrum_dict_mixed[i]['energy_bin_center'][idx_mixed[i]][j])/2)
    spectrum_dict_mixed[i]['energy_bin_center'][idx_mixed[i]][j])/(2)

lum_err_n_spatial = []

lum_err_n_temporal = []

lum_err_n_mixed = []

for k in range(len(alpha_spatial)):
    lum_err_n_spatial.append(((alpha_spatial[k]**2)*(spec_err_spatial[k]**2+spec_err_mixed[k]**2)))

for k in range(len(alpha_temporal)):
    lum_err_n_temporal.append(((alpha_temporal[k]**2)*(spec_err_temporal[k]**2+spec_err_mixed[k]**2)))

for k in range(len(alpha_mixed)):
    lum_err_n_mixed.append(((alpha_mixed[k]**2)*(spec_err_mixed[k]**2+spec_err_mixed[k]**2)))

lum_n_spatial=0

lum_n_temporal=0

lum_n_mixed=0

for i in lum_err_n_spatial:
    lum_n_spatial+=i

for i in lum_err_n_temporal:
    lum_n_temporal+=i

for i in lum_err_n_mixed:
    lum_n_mixed+=i

lum_err_spatial.append(np.sqrt(lum_n_spatial))

lum_err_temporal.append(np.sqrt(lum_n_temporal))

lum_err_mixed.append(np.sqrt(lum_n_mixed))

lum_err_spatial.reverse()

lum_err_temporal.reverse()

lum_err_mixed.reverse()

In [7]:
print(lum_spatial)
print(lum_temporal)
print(lum_mixed)

print(lum_err_spatial)
print(lum_err_temporal)

```

```
[<Quantity 1.37797801e+51 erg / s>, <Quantity 6.27357143e+50 erg / s>, <Quantity 4.99259259e+50 erg / s>, <Quantity 3.33385028e+50 erg / s>, <Quantity 2.82892976e+50 erg / s>]  
[<Quantity 2.46218207e+50 erg / s>, <Quantity 2.68939577e+50 erg / s>, <Quantity 2.62617094e+50 erg / s>, <Quantity 2.71502512e+50 erg / s>, <Quantity 2.82892976e+50 erg / s>]
```

```

5179e+50 erg / s>, <Quantity 3.24587811e+50 erg / s>, <Quantity 2.82892976e+50 erg / s>]
[<Quantity 1.74679675e+48 erg / s>, <Quantity 7.34328396e+47 erg / s>, <Quantity 5.96610198e+47 erg / s>, <Quantity 4.54527034e+47 erg / s>, <Quantity 3.92742765e+47 erg / s>]
[<Quantity 1.1664939e+48 erg / s>, <Quantity 8.61404766e+47 erg / s>, <Quantity 6.49883383e+47 erg / s>, <Quantity 3.94120008e+47 erg / s>, <Quantity 3.92742765e+47 erg / s>]
[<Quantity 2.37380378e+48 erg / s>, <Quantity 2.04010158e+48 erg / s>, <Quantity 1.16537856e+48 erg / s>, <Quantity 5.08046569e+47 erg / s>, <Quantity 3.92742765e+47 erg / s>]

```

In [8]:

```

lumi_spatial = [ 1.37797801e+51 , 6.27357143e+50 ,
                 4.99259259e+50 , 3.33385028e+50 , 2.82892976e+50 ]

lumi_temporal = [ 2.46218207e+50 , 2.68939577e+50 ,
                  2.62617094e+50 , 2.71502512e+50 , 2.82892976e+50 ]

lumi_mixed = [ 5.61176557e+50 , 4.97049663e+50 ,
               3.85951769e+50 , 3.24587811e+50 , 2.82892976e+50 ]

```

```

                    5.96610198e+47 , 4.54527034e+47 , 3.92742765e+47 ]

lumi_err_temporal = [ 1.1664939e+48 , 8.61404766e+47 ,
                      6.49883383e+47 , 3.94120008e+47 , 3.92742765e+47 ]

lumi_err_mixed = [ 2.37380378e+48 , 2.04010158e+48 ,
                   1.16537856e+48 , 5.08046569e+47 , 3.92742765e+47 ]

```

```
label size = 20
```

```
f, axarr = plt.subplots(3, sharex=False, sharey=False)
axarr_spe_x = axarr

f.set_figwidth(12)
f.set_figheight(15)

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

axarr_spe_x[0].yaxis.set_major_formatter(formatter)

axarr_spe_x[1].yaxis.set_major_formatter(formatter)

axarr_spe_x[2].yaxis.set_major_formatter(formatter)

axarr_spe_x[0].scatter(levs, lumi_spatial)

axarr_spe_x[0].errorbar(levs, lumi_spatial, lumi_err_spatial,
                        fmt='none', barsabove=True, ecolor='blue')

axarr_spe_x[0].set_xticks(range(1, 6))

axarr_spe_x[0].set_ylabel(r'L (erg/s)')

axarr_spe_x[0].set_xlabel("Spatial Refinement Levels")

axarr_spe_x[1].scatter(fps, lumi_temporal)

axarr_spe_x[1].errorbar(fps, lumi_temporal, lumi_err_temporal,
                        fmt='none', barsabove=True, ecolor='blue')

axarr_spe_x[1].set_ylabel(r'L (erg/s)')

axarr_spe_x[1].set_xlabel("Temporal Refinement Levels (fps)")

axarr_spe_x[2].scatter(mix, lumi_mixed)

axarr_spe_x[2].errorbar(mix, lumi_mixed, lumi_err_mixed,
                        fmt='none', barsabove=True, ecolor='blue')

axarr_spe_x[2].set_ylabel(r'L (erg/s)')

axarr_spe_x[2].set_xlabel("Mixed Refinement Levels (Spatial, Temporal (fps))")

#plt.yscale('log')
axarr_spe_x[0].annotate(' (a)', xy=(0.9, 0.9), xycoords="axes fraction")

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

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

plt.tight_layout()
#plt.savefig('lum_spherical.pdf', dpi=600, bbox_inches='tight')
plt.show()
```

Spatial Refinement Levels	L (erg/s) × 10 ⁵⁰
1	~14.0
2	~6.0
3	~5.0
4	~3.0
5	~2.5

