In [10]:
```
from sherpa.astro.ui import * #import sherpa function
from ciao_contrib.runtool import * #import CIAO tools
import matplotlib.pyplot as plt
```

In [4]:
```
#we can define the source and bkg files to be used by specextract:
specextract.infile="../acisf09514_repro_evt2.fits[sky=region(srcA.reg)]"
specextract.bkgfile="../acisf09514_repro_evt2.fits[sky=region(bkg.reg)]"
# define the directory to load the outputs
specextract.outroot="srcA/srcA"
#run the specextract function
specextract()
```

Out[4]:
```
Running specextract
Version: 18 January 2023

Aspect solution file ../pcadf09514_000N001_asol1.fits found.

Bad pixel file ../acisf09514_repro_bpix1.fits found.

Mask file ../acisf09514_000N003_msk1.fits found.

Extracting bkg spectra

Extracting src spectra

Using mkacisrmf...

Using mkacisrmf...

Creating bkg RMF

Creating bkg ARF

Creating src RMF

Creating src ARF
```

In [5]:
```
#we can define the source and bkg files to be used by specextract:
specextract.infile="../acisf09514_repro_evt2.fits[sky=region(srcB.reg)]"
specextract.bkgfile="../acisf09514_repro_evt2.fits[sky=region(bkg.reg)]"
# define the directory to load the outputs
specextract.outroot="srcB/srcB"
#run the specextract function
specextract()
```

Out[5]:
```
Running specextract
Version: 18 January 2023

Aspect solution file ../pcadf09514_000N001_asol1.fits found.

Bad pixel file ../acisf09514_repro_bpix1.fits found.

Mask file ../acisf09514_000N003_msk1.fits found.

Extracting src spectra

Extracting bkg spectra

Using mkacisrmf...

Using mkacisrmf...

Creating src RMF

Creating src ARF

Creating bkg ARF

Creating bkg RMF
```

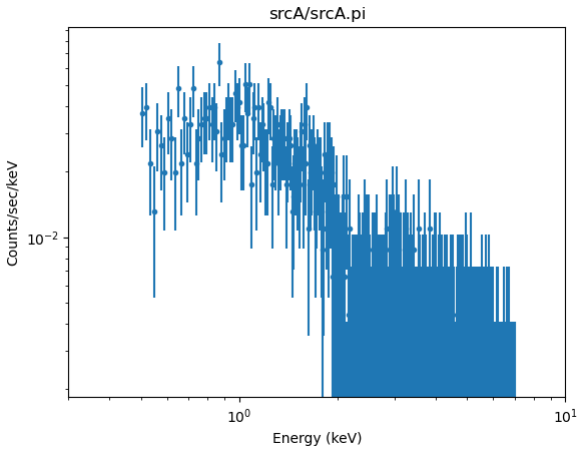
In [7]:
```
load_pha(1,"srcA/srcA.pi")
load_pha(2,'srcB/srcB.pi')
```

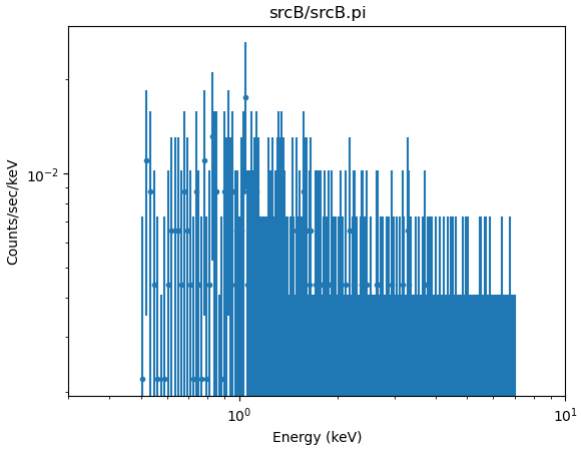
```
read ARF file srcA/srcA.arf
read RMF file srcA/srcA.rmf
read ARF (background) file srcA/srcA_bkg.arf
read RMF (background) file srcA/srcA_bkg.rmf
read background file srcA/srcA_bkg.pi
read ARF file srcB/srcB.arf
read RMF file srcB/srcB.rmf
read ARF (background) file srcB/srcB_bkg.arf
read RMF (background) file srcB/srcB_bkg.rmf
read background file srcB/srcB_bkg.pi
```

In [8]:
```
#we constrain our energy range from 0.5-7 keV
notice(0.5,7)
```

```
dataset 1: 0.0073:14.9504 -> 0.4964:7.008 Energy (keV)
dataset 2: 0.0073:14.9504 -> 0.4964:7.008 Energy (keV)
```

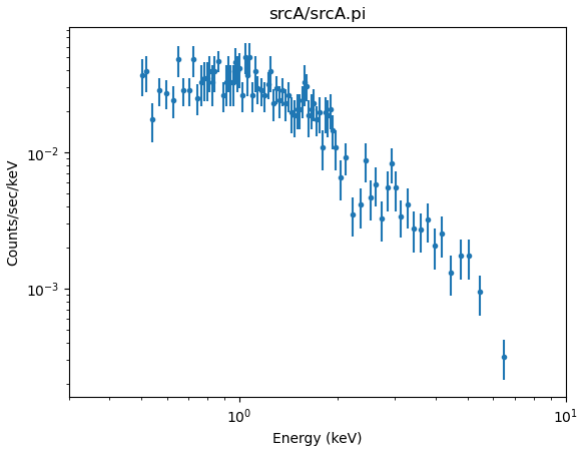
In [12]:
```
#We plot the raw data of dataset 1
plot_data(1,xlog=True,ylog=True)
plt.xlim(0.3,10)
plt.show()

#We plot the raw data of dataset 2
plot_data(2,xlog=True,ylog=True)
plt.xlim(0.3,10)
plt.show()
```

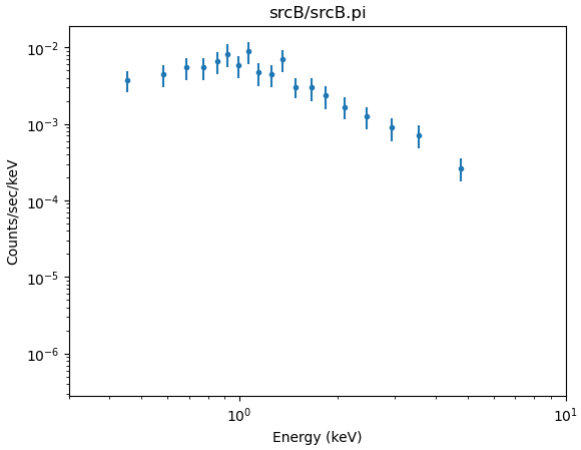




In [13]:
```
#we group the data by 50 counts per group (default is 15)
group_counts(1,15)
group_counts(2,15)


#we subtract the background associated with each dataset
subtract(1)
subtract(2)
```

In [14]:
```
#we now plot the regrouped, background subtracted spectrum of daset 1
plot_data(1,xlog=True,ylog=True)
plt.xlim(0.3,10)
plt.show()

#we do the same for dataset 2
plot_data(2,xlog=True,ylog=True)
plt.xlim(0.3,10)
plt.show()
```





we can now define our models. We will use

1 - `xszpowerlw` : redshifted power law photon spectrum. The default parameters are PhoIndex, redshift, and norm.

2 - `xszphabs` : photoelectric absorption of lens. Paramaters are nH and redshift.

3 - `xsphabs` : photoelectric absorption from line of sight of observer. Parameters are nH.

models from https://cxc.cfa.harvard.edu/sherpa/models/ (https://cxc.cfa.harvard.edu/sherpa/models/)

structure for defining model is set_source(index,model name.variable *model 2.variable 2* etc).

By passing the same variable name to multiple sources, we are linking the model (they will be the same value for all).

In [15]:
```
set_source(1,xszpowerlw.p1*xszphabs.labs1*xsphabs.gabs)
set_source(2,xszpowerlw.p2*xszphabs.labs2*gabs)
```

we can now redefine some of the parameters to their known/observed value. The redshift of the source and the lens is found in the CASTLES survey (https://lweb.cfa.harvard.edu/castles/).

The HI column density is found using HEASARC (https://heasarc.gsfc.nasa.gov/cgi-bin/Tools/w3nh/w3nh.pl).

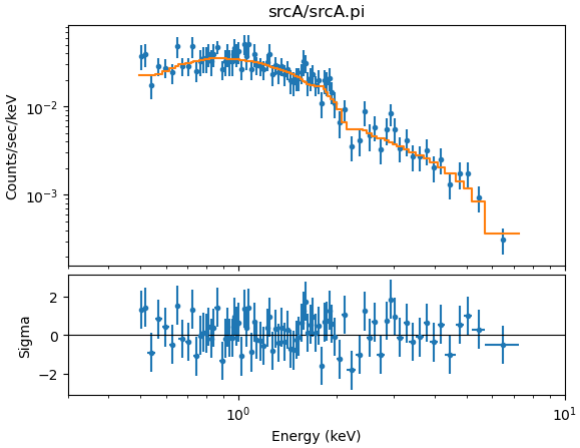
In [16]:
```
#we fix the redshift of the source since it is known.
#we link it for all models since it is same z for all.
p1.redshift = 1.37
link(p2.redshift,p1.redshift)

#similarly, the redshift of the lens is known.
#we fix it and link it to all the models.
labs1.redshift = 0.48
link(labs2.redshift,labs1.redshift)

#We fix the galactic nH value.
#Again, it will be the same for all models.
# 5.25 x 10^20 so
gabs.nh = 0.0525 #unit: 10^22 atoms / cm^2
freeze(gabs.nH)

#the photon index is expected to be the same for all sources:
link(p2.PhoIndex, p1.PhoIndex)

#we can look at our models:
show_model()
```

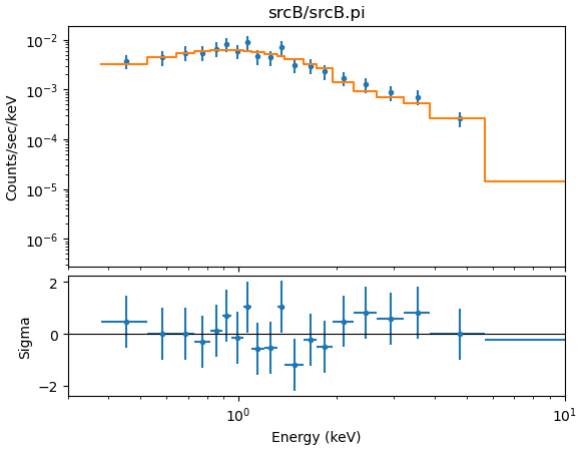
```
Model: 1
apply_rmf(apply_arf((31188.838169386 * ((xszpowerlw.p1 * xszphabs.labs1) * xsphabs.gabs))))
   Param        Type          Value          Min          Max      Units
   -----        ----          -----          ---          ---      -----
   p1.PhoIndex  thawed            1           -3           10
   p1.redshift  frozen         1.37       -0.999           10
   p1.norm      thawed            1            0        1e+24
   labs1.nH     thawed            1            0        1e+06 10^22 atoms / cm^2
   labs1.redshift frozen         0.48       -0.999           10
   gabs.nH      frozen       0.0525            0        1e+06 10^22 atoms / cm^2

Model: 2
apply_rmf(apply_arf((31188.838169386 * ((xszpowerlw.p2 * xszphabs.labs2) * xsphabs.gabs))))
   Param        Type          Value          Min          Max      Units
   -----        ----          -----          ---          ---      -----
   p2.PhoIndex  linked            1        expr: p1.PhoIndex
   p2.redshift  linked         1.37        expr: p1.redshift
   p2.norm      thawed            1            0        1e+24
   labs2.nH     thawed            1            0        1e+06 10^22 atoms / cm^2
   labs2.redshift linked         0.48     expr: labs1.redshift
   gabs.nH      frozen       0.0525            0        1e+06 10^22 atoms / cm^2
```

In [17]:
```
#once our parameters are adjusted, we can simultaneously fit all 4 models.
#we fit three times so the fit converges.
fit(1,2)
fit(1,2)
print('----------')
fit(1,2)
```

```
Datasets              = 1, 2
Method                = levmar
Statistic             = chi2gehrels
Initial fit statistic = 4.15646e+11
Final fit statistic   = 66.0503 at function evaluation 322
Data points           = 104
Degrees of freedom    = 99
Probability [Q-value] = 0.995587
Reduced statistic     = 0.667174
Change in statistic   = 4.15646e+11
   p1.PhoIndex    1.90249      +/- 0.0860248
   p1.norm        0.000407501  +/- 5.61505e-05
   labs1.nH       0.00202197   +/- 0.0545696
   p2.norm        7.71039e-05  +/- 1.13953e-05
   labs2.nH       0.0076188    +/- 0.0710819
Datasets              = 1, 2
Method                = levmar
Statistic             = chi2gehrels
Initial fit statistic = 66.0503
Final fit statistic   = 65.5828 at function evaluation 291
Data points           = 104
Degrees of freedom    = 99
Probability [Q-value] = 0.996122
Reduced statistic     = 0.662453
Change in statistic   = 0.467438
   p1.PhoIndex    1.90204      +/- 0.0884685
   p1.norm        0.000404742  +/- 5.96491e-05
   labs1.nH       0            +/- 0.0541931
   p2.norm        7.30239e-05  +/- 1.20793e-05
   labs2.nH       0            +/- 0.0698041
WARNING: parameter value labs1.nH is at its minimum boundary 0.0
WARNING: parameter value labs2.nH is at its minimum boundary 0.0
----------
Datasets              = 1, 2
Method                = levmar
Statistic             = chi2gehrels
Initial fit statistic = 65.5828
Final fit statistic   = 65.5665 at function evaluation 278
Data points           = 104
Degrees of freedom    = 99
Probability [Q-value] = 0.996139
Reduced statistic     = 0.662288
Change in statistic   = 0.0162759
   p1.PhoIndex    1.90221      +/- 0.0883648
   p1.norm        0.00040539   +/- 5.96707e-05
   labs1.nH       0            +/- 0.0541192
   p2.norm        7.30838e-05  +/- 1.20771e-05
   labs2.nH       0            +/- 0.0697446
WARNING: parameter value labs1.nH is at its minimum boundary 0.0
WARNING: parameter value labs2.nH is at its minimum boundary 0.0
```

In [18]:
```
#we can plot our fit for dataset 1 and include residuals
set_xlog()
set_ylog()
plot_fit_delchi(1)
plt.xlim(0.3,10)
plt.show()

#we can plot our fit for dataset 1 and include residuals
set_xlog()
set_ylog()
plot_fit_delchi(2)
plt.xlim(0.3,10)
plt.show()
```





In [19]:
```
#we estimate the parameter confidence intervals
conf(1,2)
```

```
labs1.nH lower bound:   -----
labs2.nH lower bound:   -----
p1.norm lower bound:    -2.00181e-05
labs2.nH upper bound:   0.049471
p1.norm upper bound:    4.90178e-05
labs1.nH upper bound:   0.0301181
p1.PhoIndex lower bound:        -0.0379315
p1.PhoIndex upper bound:        0.0766384
p2.norm lower bound:    -5.74022e-06
p2.norm upper bound:    1.08565e-05
Datasets              = 1, 2
Confidence Method     = confidence
Iterative Fit Method  = None
Fitting Method        = levmar
Statistic             = chi2gehrels
confidence 1-sigma (68.2689%) bounds:
   Param            Best-Fit  Lower Bound  Upper Bound
   -----            --------  -----------  -----------
   p1.PhoIndex       1.90221   -0.0379315    0.0766384
   p1.norm        0.00040539 -2.00181e-05  4.90178e-05
   labs1.nH                0        -----    0.0301181
   p2.norm       7.30838e-05 -5.74022e-06  1.08565e-05
   labs2.nH                0        -----     0.049471
```

In [ ]: