Use of XSPEC for XRISM

Keith Arnaud

xspec12@athena.gsfc.nasa.gov
https://www.facebook.com/groups/320119452570

High Energy Astrophysics Science Archive Research Center

University of Maryland College Park,
NASA's Goddard Space Flight Center,
and CRESST

Title: "Exploring the Universe with xspec and XRISM"

I. Introduction

Brief overview of X-ray spectroscopy and its importance in understanding the Universe

Introduction to xspec and XRISM

II. X-ray Spectroscopy

Explanation of X-ray spectroscopy and how it works

Importance of X-ray spectroscopy in studying the properties of astrophysical objects

Applications of X-ray spectroscopy in astrophysics research

III. xspec

Introduction to xspec and its history

Overview of xspec's features and capabilities

How xspec is used in astrophysics research

Examples of xspec analysis in recent astrophysics research

IV. XRISM

Introduction to XRISM and its mission

Overview of XRISM's capabilities and instrumentation

Explanation of how XRISM will use X-ray spectroscopy to study the Universe

Potential discoveries and contributions of XRISM to astrophysics research

V. Future of X-ray Spectroscopy

Discussion of future advancements in X-ray spectroscopy and their potential impact on astrophysics research

Importance of continued development and utilization of X-ray spectroscopy in studying the Universe

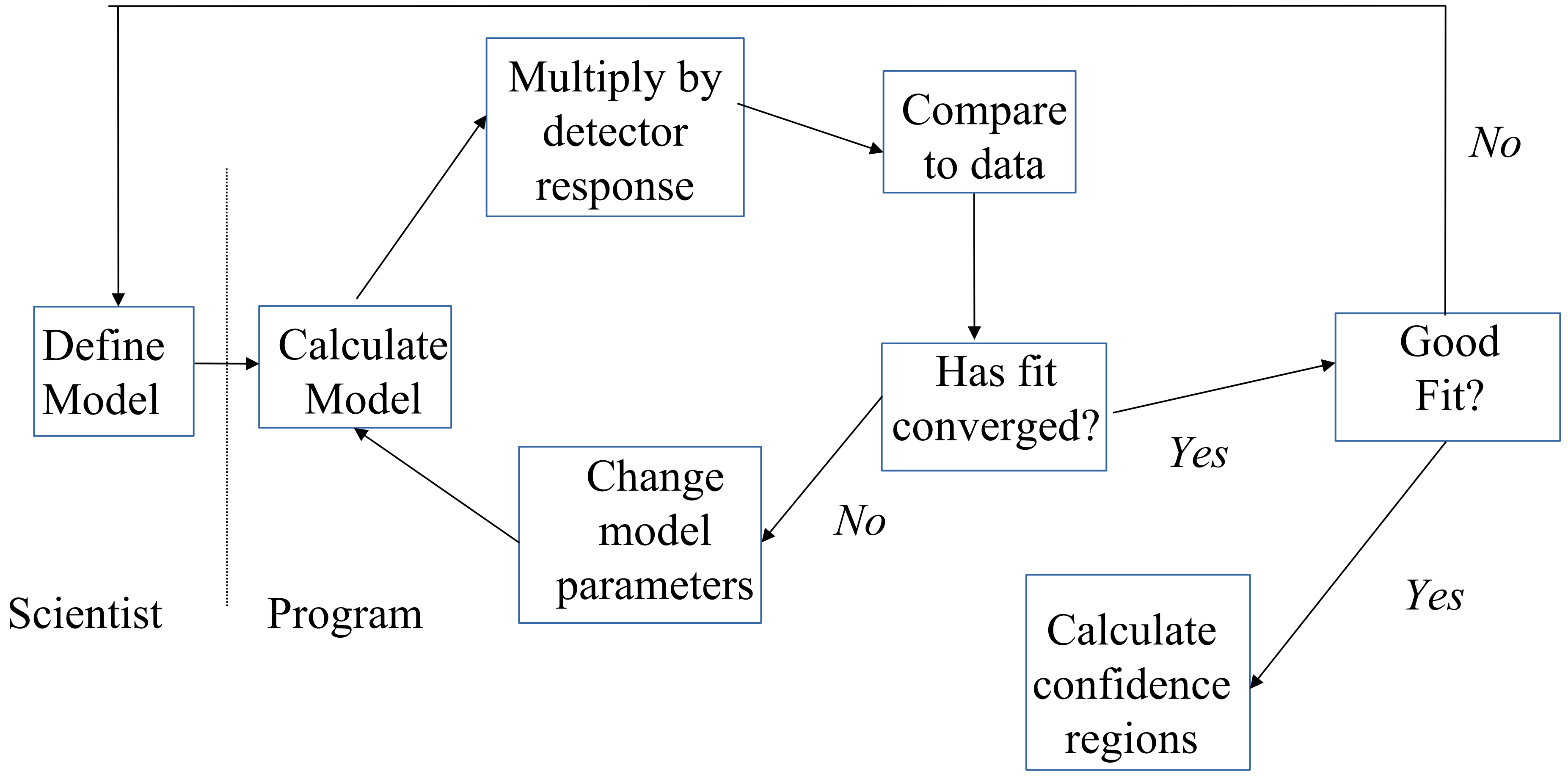
VI. Conclusion

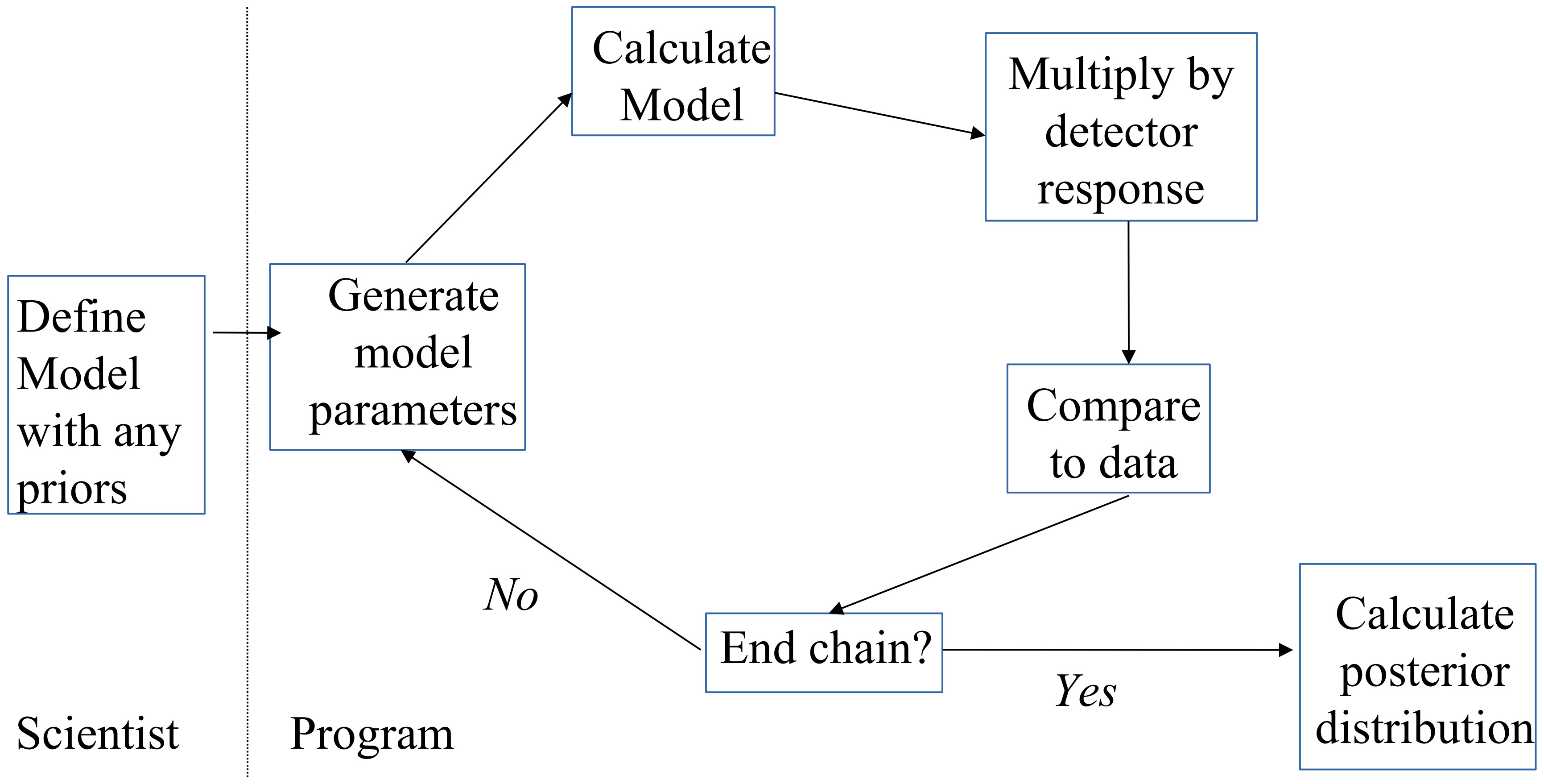
Recap of the importance of X-ray spectroscopy and the role of xspec and XRISM in astrophysics research
Call to action for continued support and funding for X-ray spectroscopy research and technology development.

General comments

The Resolve response matrix

Simultaneous fitting of Resolve and Xtend


Define Model
Calculate Model
Multiply by detector response
Compare to data
Has fit converged?
Good Fit?
Change model parameters
Calculate confidence regions
No
Yes
No
Yes
Scientist
Program


Define Model with any priors
Generate model parameters
Calculate Model
Multiply by detector response
Compare to data
End chain?
No
Yes
Calculate posterior distribution
Scientist
Program

XSPEC v12.13 and PyXspec v2.1.1 were released with HEAsoft 6.31.1 in 12/22.

We try to do one new XSPEC release per year.

Patches are available at https://heasarc.gsfc.nasa.gov/docs/xanadu/xspec/bugs.html

Additional models sent to us are made available and if requested are added as standard models in the next release. See https://heasarc.gsfc.nasa.gov/docs/xanadu/xspec/newmodels.html

Note that HEAsoft now includes gsl so any models can use these routines.

Please do not use Numerical Recipes routines in models since these are copyrighted and are not supposed to be distributed.

If you are not already a member I recommend joining the xspec Facebook group which now has ~1,570 members (https://www.facebook.com/groups/320119452570/).

The standard XSPEC interface is written in Tcl and is difficult to use for complex scripts. PyXspec is a Python package which provides a way of running XSPEC from the Python interface. Only one XSPEC process can be run from a Python session.

Internal XSPEC data can then be used in other Python packages. For instance plotting can be done using matplotlib.

https://heasarc.gsfc.nasa.gov/docs/xanadu/xspec/python/html

https://heasarc.gsfc.nasa.gov/docs/xanadu/xspec/morepython.html

https://github.com/HEASARC/PyXspec-Jupyter-notebooks

HEASP and HEASPTOOLS

heasp is a C++ library of routines to manipulate spectra, responses, table models.

A python module called heasp provides an interface to these C++ routines.

(https://heasarc.gsfc.nasa.gov/docs/software/lheasoft/headas/heasp/heasp_guide.html)

heasptools are a set of ftools built on top of heasp and will replace many old tools in caltools and heasarc.

For instance, ftrbnpha replaces rbnpha. ftgrouppha is a replacement for grppha with many more grouping options including optimal binning.

(https://heasarc.gsfc.nasa.gov/lheasoft/ftools/headas/heasptools.html)

```
from heasp import *
import xspec as xsp
import numpy as np

responseName = “input.rsp”
inputRSP = rmf(responseName)

# energyBins needs to be the standard internal xspec energy array
numEnergies = inputRSP.NumberEnergyBins() + 1
energies = np.empty((numEnergies))
energies[0] = inputRSP.getLowEnergyElement(0)
for j in range(1,numEnergies): energies[j] = inputRSP.getHighEnergyElement(j-1)

# set up parameters as a list
params = [ 1.7 ]
# and output flux as a list
fluxlist = []
xsp.callModelFunction(“powerlaw”,energies.tolist(), params, fluxlist)
flux = np.array(fluxlist)
phaValues = inputRSP.multiplyByModel(flux)

channel = np.arange(phaValues.size).astype(np.int32)
exposure = 10000.0
phaValues *= exposure
```

```
phaOut = pha()
phaOut.setFirstChannel(0)
phaOut.setPha(phaValues)
phaOut.setChannel(channel)
phaOut.setExposure(exposure)
phaOut.setDetChans(phaValues.size)
phaOut.setPoisserr(True)
phaOut.setDatatype(“COUNT”)
phaOut.setSpectrumType(“TOTAL”)
phaOut.setResponseFile(responseName)
phaOut.setTelescope(inputRSP.getTelescope())
phaOut.setInstrument(inputRSP.getInstrument())
phaOut.setFilter(inputRSP.getFilter())

status = phaOut.write(“output.pha”)
```

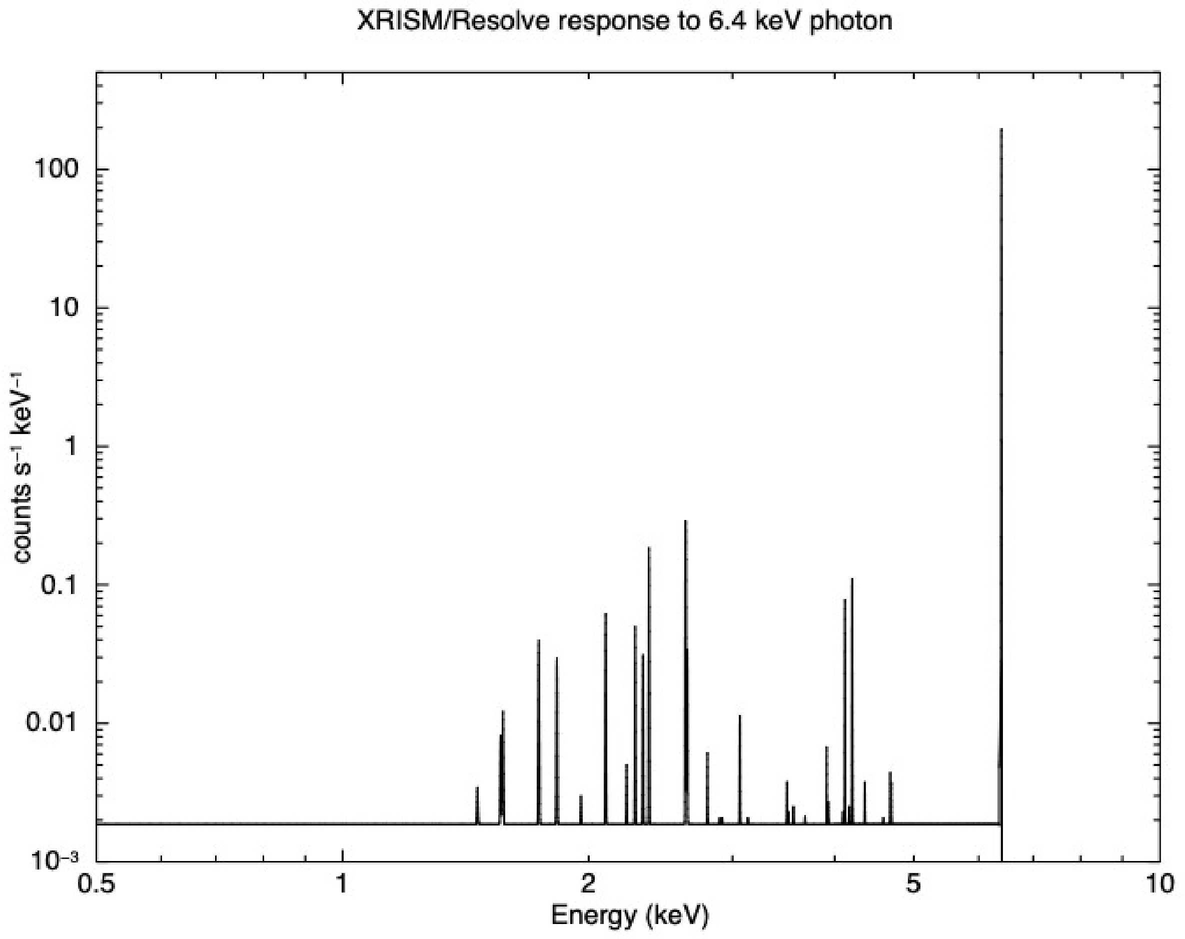
XSPEC speed

XSPEC speed is dominated by two steps: calculating the model and multiplying the model by the response matrix.

Model calculation is usually $O(N_E)$ or $O(N_E \log N_E)$ where N_E is the number of energy bins.

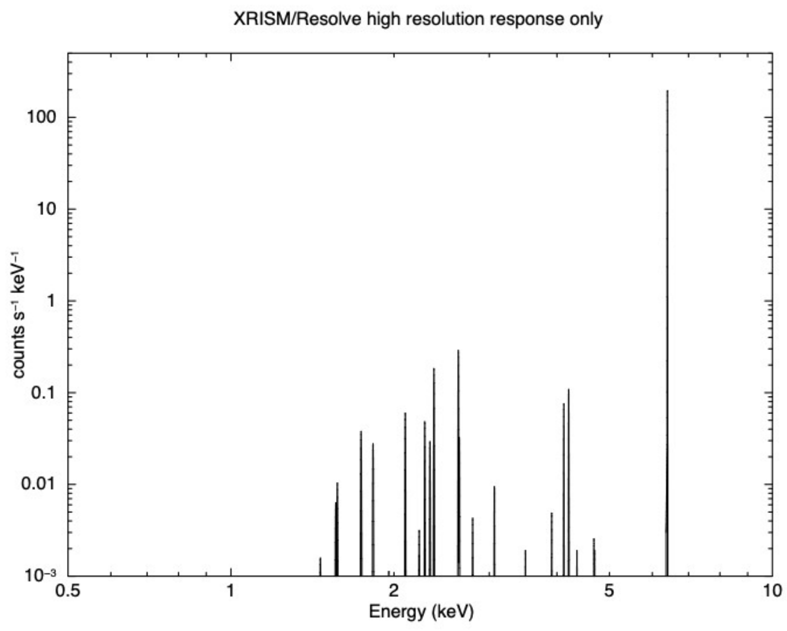
Response matrix calculation is $O(N_R)$ where N_R is the number of non-zero elements in the matrix.

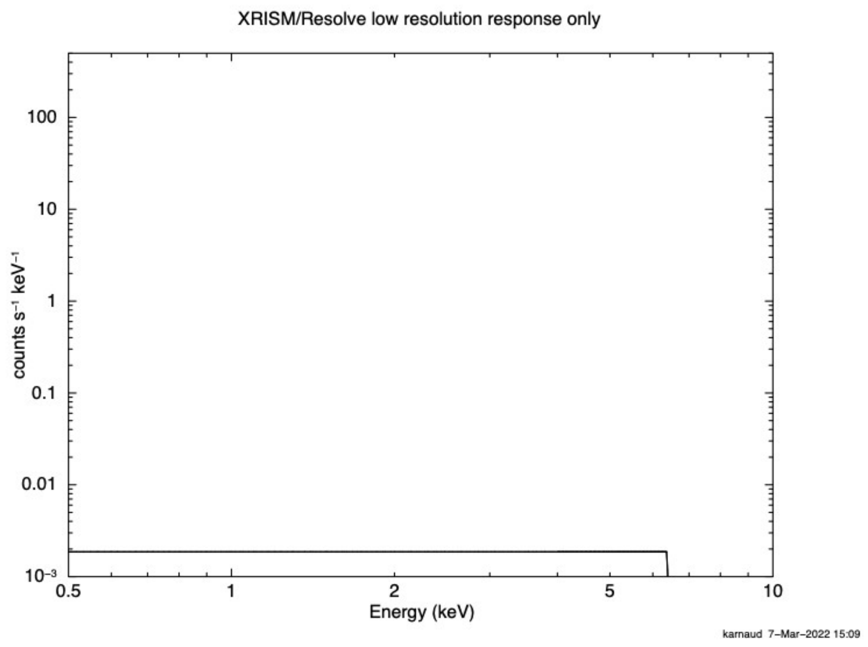
XRISM/Resolve has 60,000 energy bins and 60,000 channels. Since the response matrix is triangular it is >7 Gb in size.



Calculate model on 60,000 energy bins

Bin up to 3,750 energy bins





Combining high res and low res data: A Toy Case

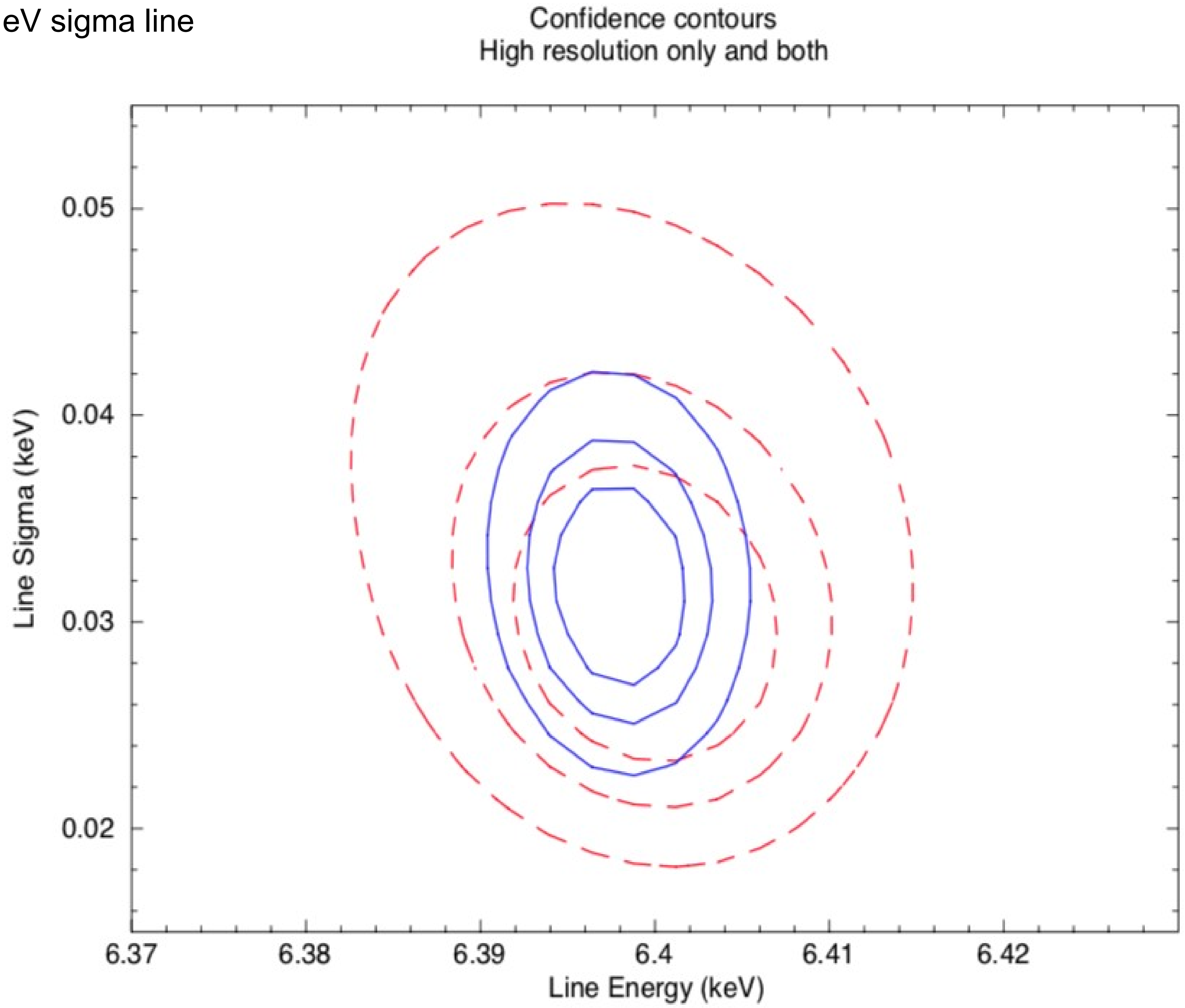
High resolution detector with FWHM 5 eV

Low resolution detector with FWHM 100 eV and 10x the effective area of high resolution detector

Single gaussian line and power-law background.

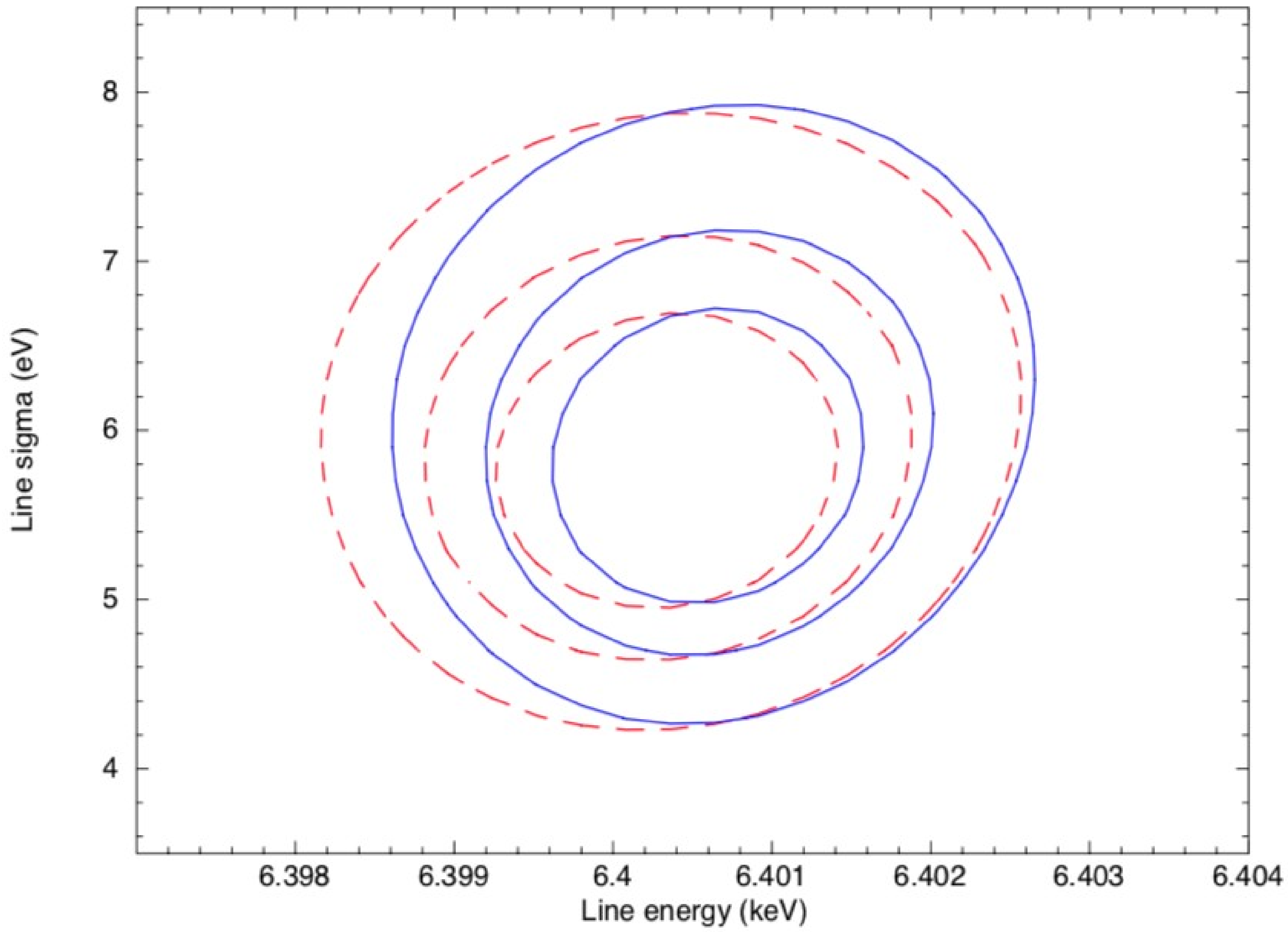
Look at cases of 30 eV sigma line and 5 eV sigma line.

30 eV sigma line

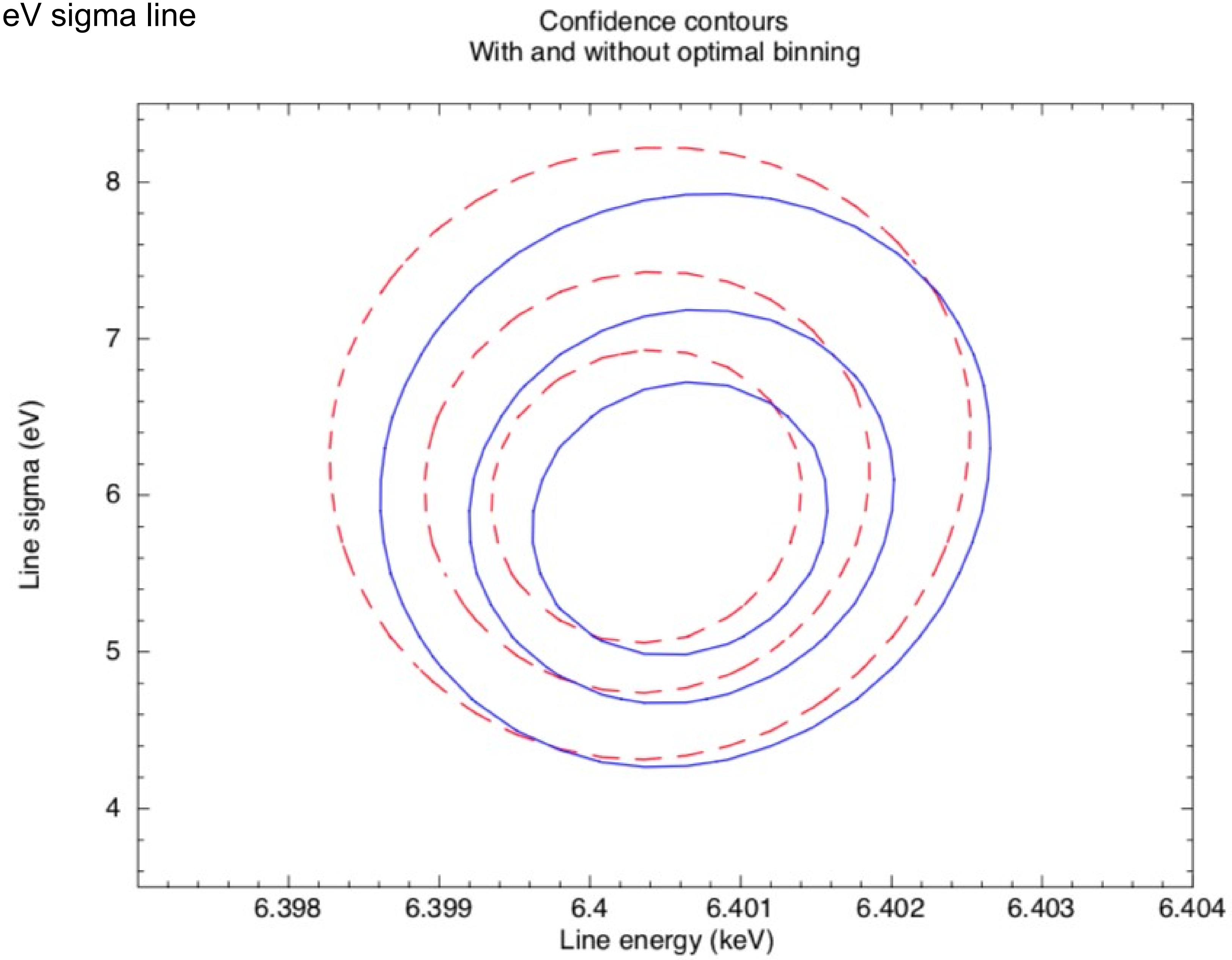


5 eV sigma line

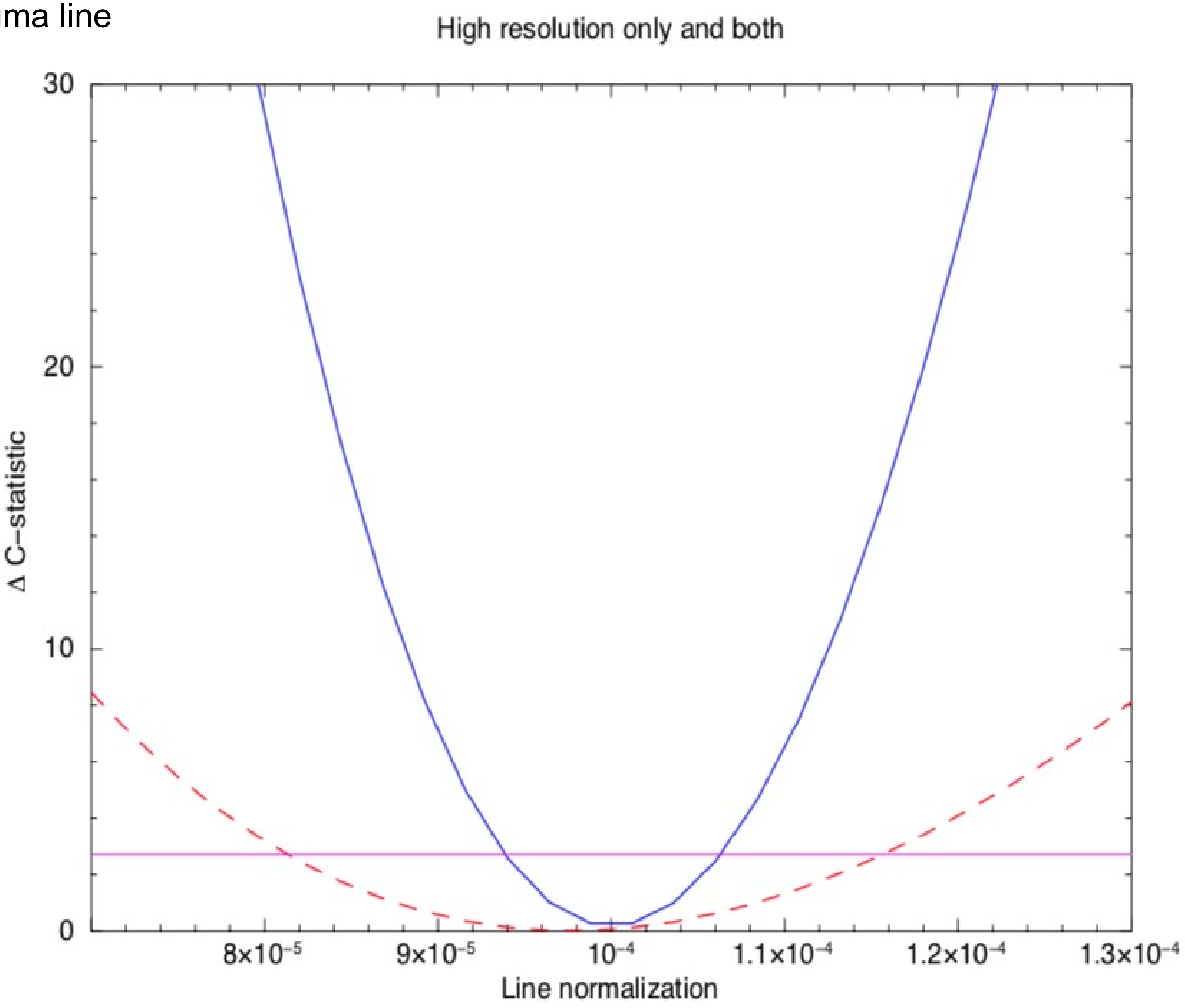
Confidence contours
High resolution only and both



5 eV sigma line



5 eV sigma line



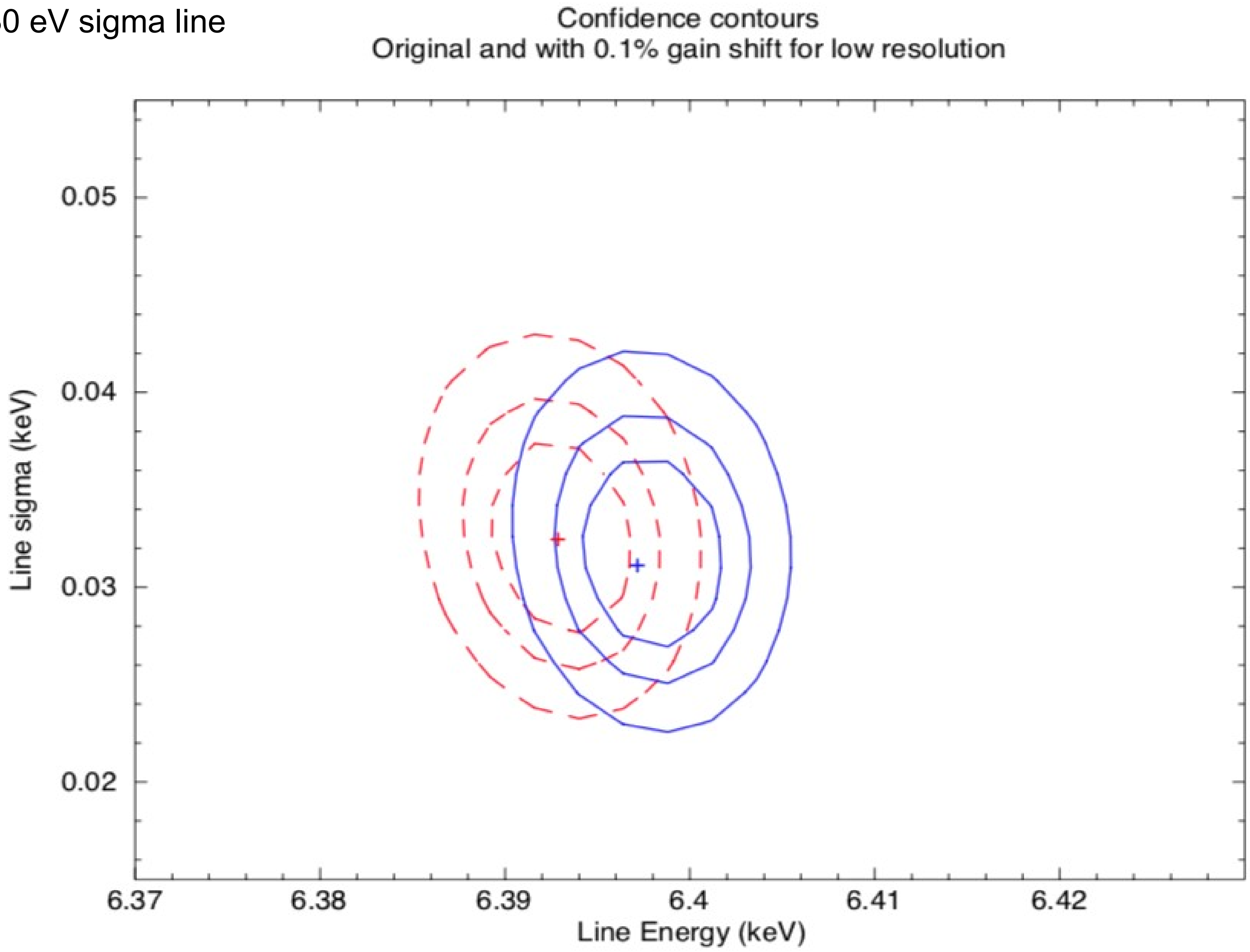
Combining high res and low res data: What could possibly go wrong?

If the model is correct the greater number of counts in the low res data could drive the fit to a local minimum which doesn’t work for the high res data. - Probably best to fit the high res data first.

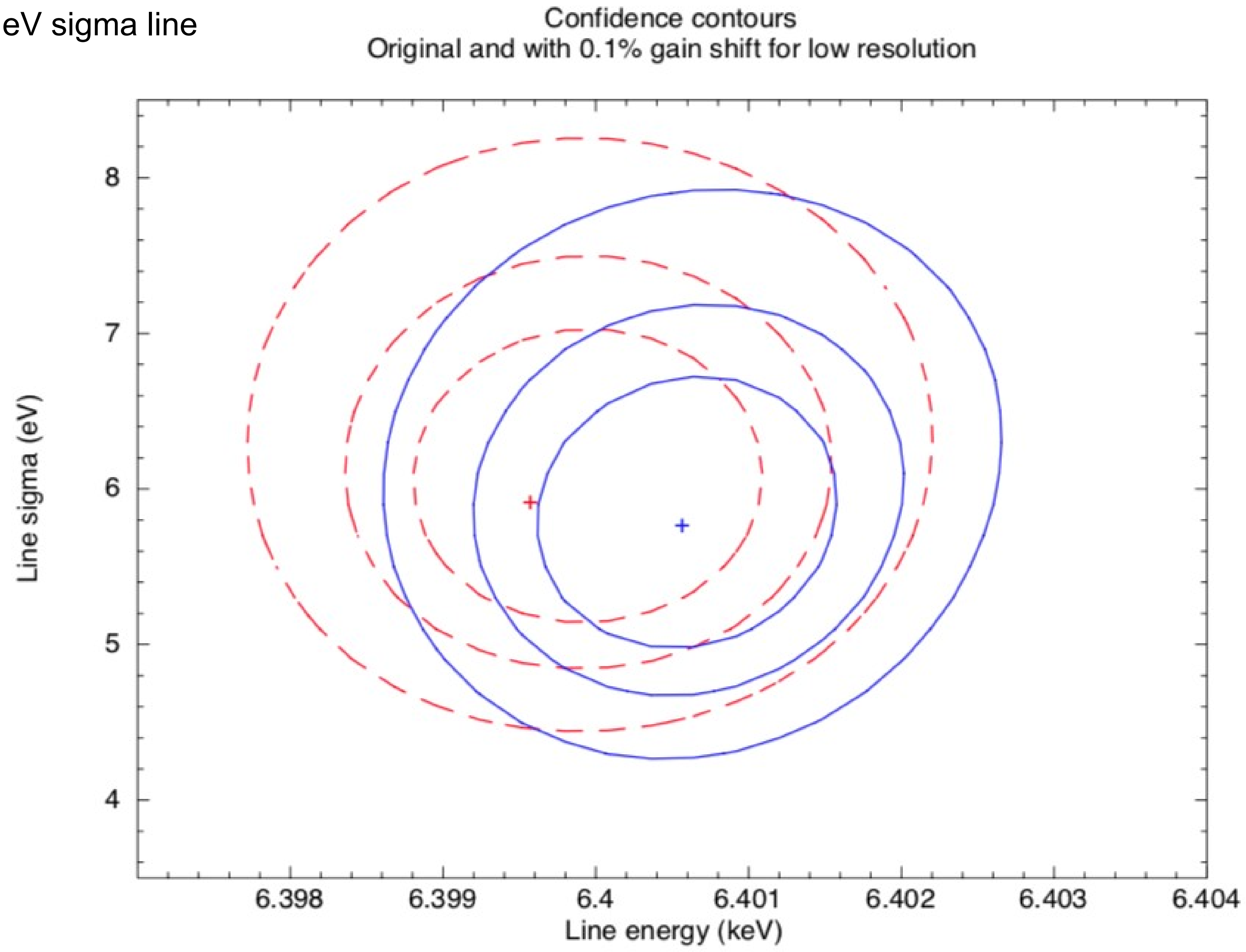
If the model is incorrect then the fit may end up in a global minimum which doesn’t work for the high res data. Find a better model.

If the low res data has calibration problems then the fit may not work for the high res data and the parameters may be wrong.

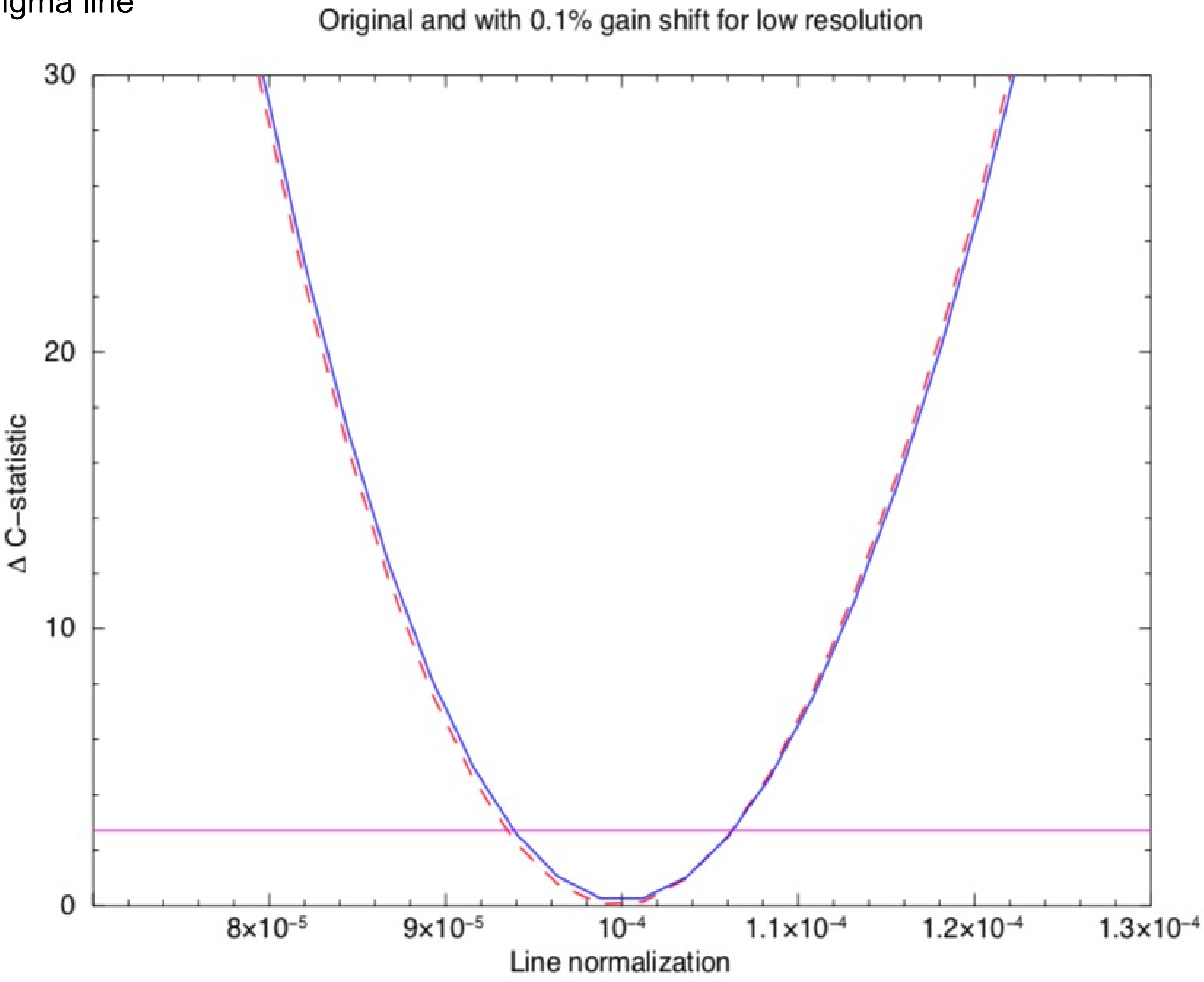
30 eV sigma line



5 eV sigma line



5 eV sigma line


Original and with 0.1% gain shift for low resolution
Δ C–statistic
8×10⁻⁵
9×10⁻⁵
10⁻⁴
1.1×10⁻⁴
1.2×10⁻⁴
1.3×10⁻⁴
Line normalization