Pretty plots using matplotlib

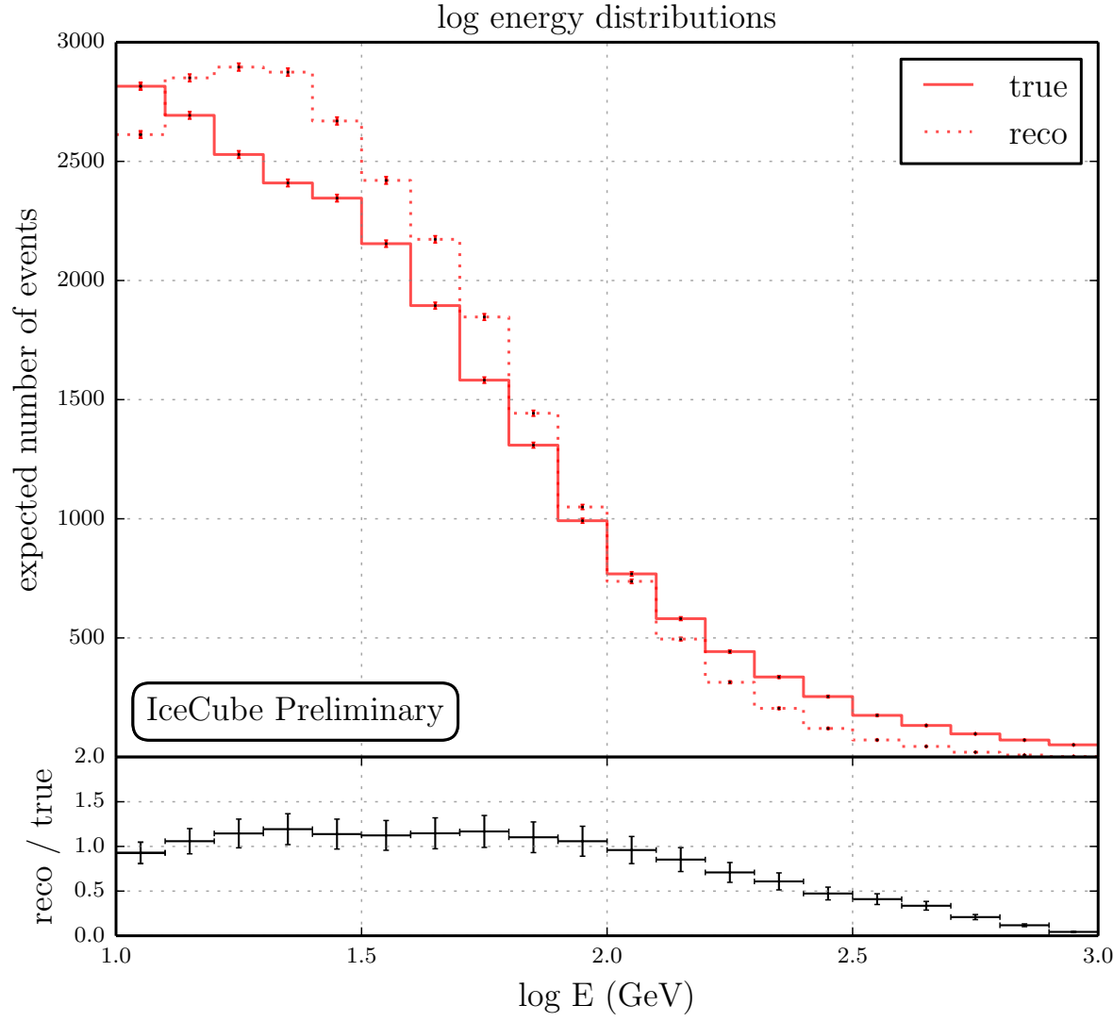
Elim Cheung

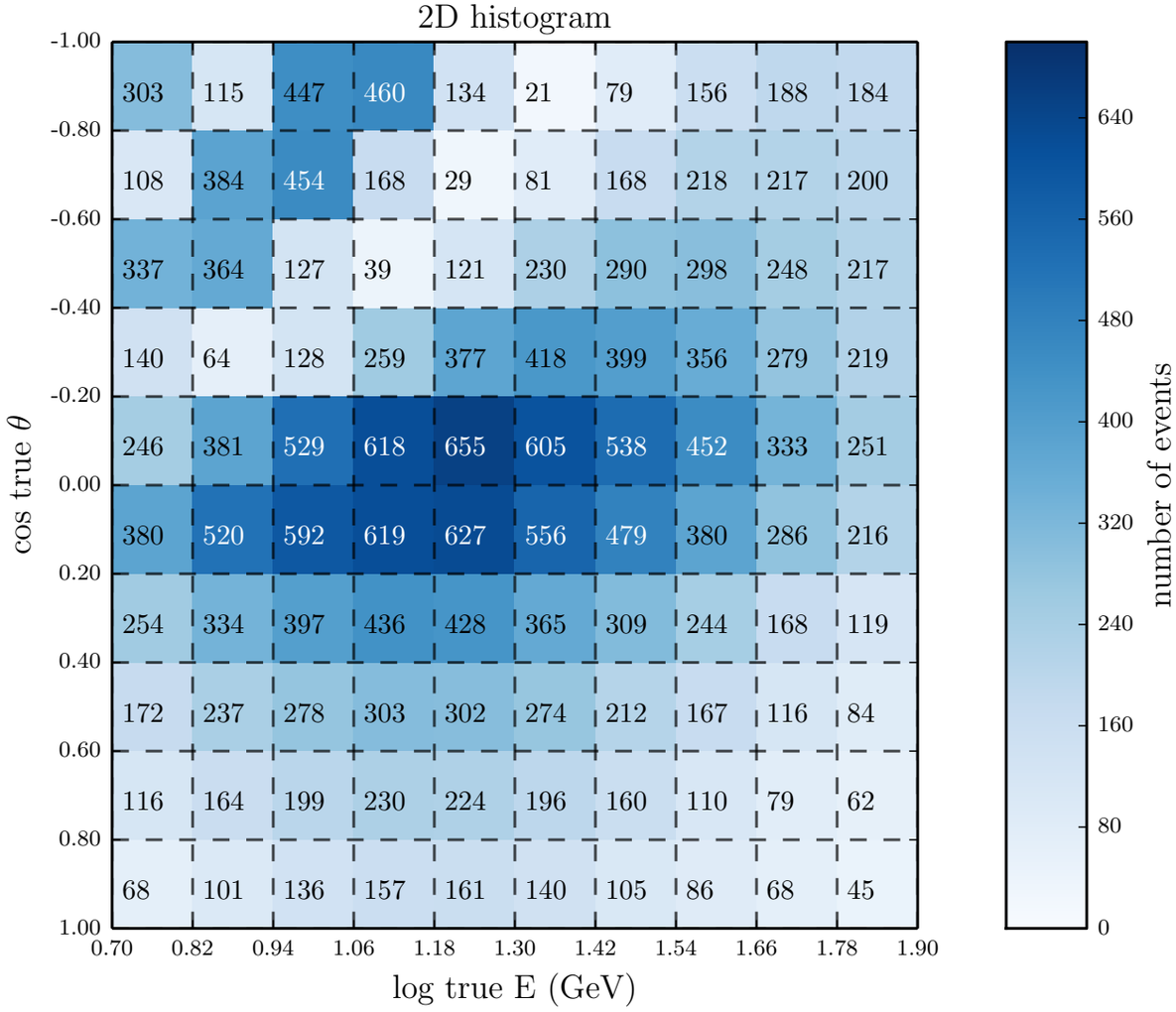
06/11/16

|

Summary: your goal before lunch

ν_μ histograms (expected numbers in a year)



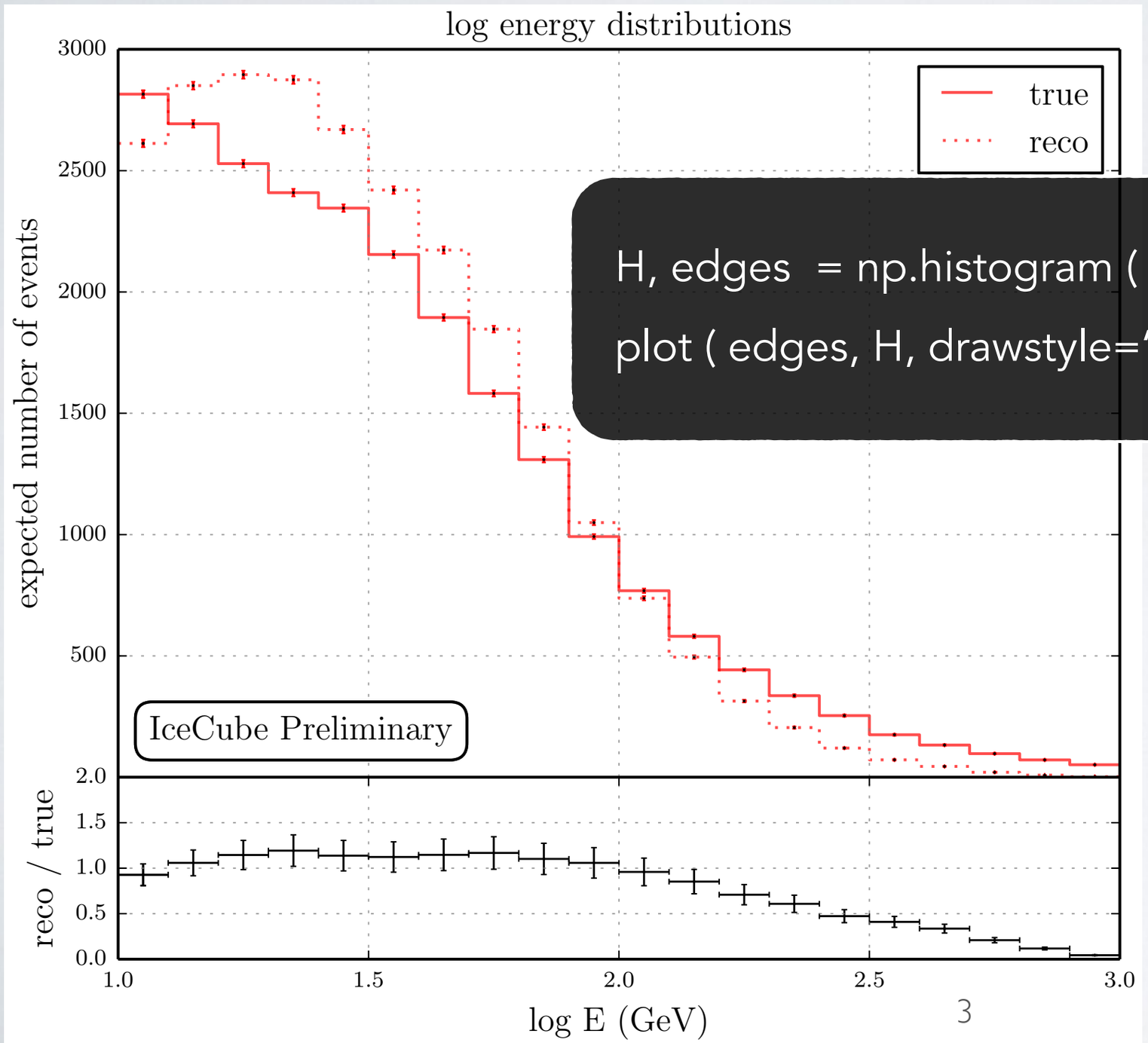


to see details of this plot: http://umdgrb.umd.edu/~elims/plots/bootcamp/histograms.pdf

Histograms & Weights

A histogram = event counts per bin

A bin = a small range of a variable (energy/zenith/etc)



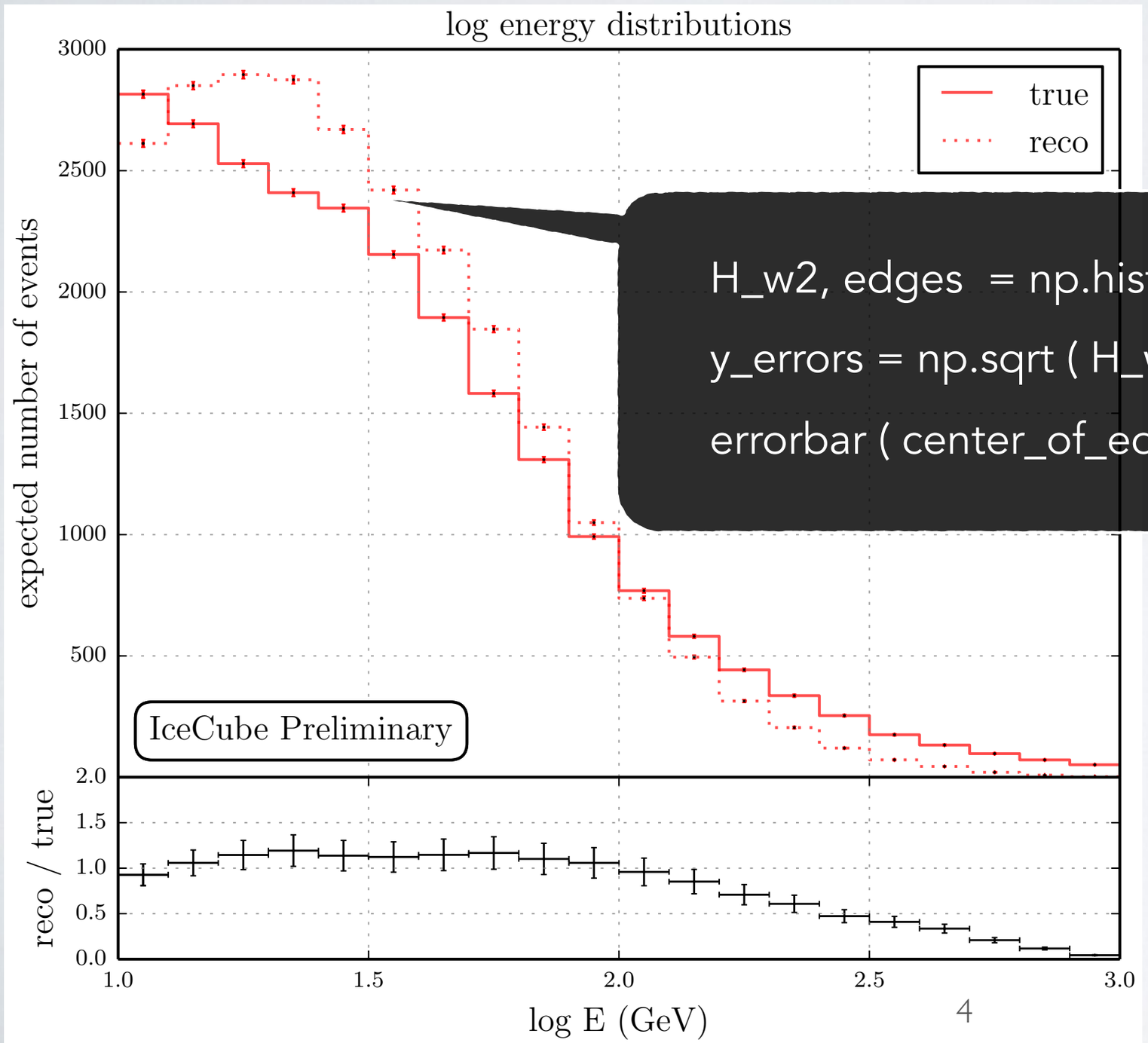

```
H, edges  = np.histogram ( data, bins=nbins, weights=weights )
plot ( edges, H, drawstyle='steps-pre', label='label', color='red' )
```

for more info about plt.plot: here

Errorbars for a histogram

By Poisson statistics, given N events in a bin,

Error on counts in this bin = $\sqrt{N}$



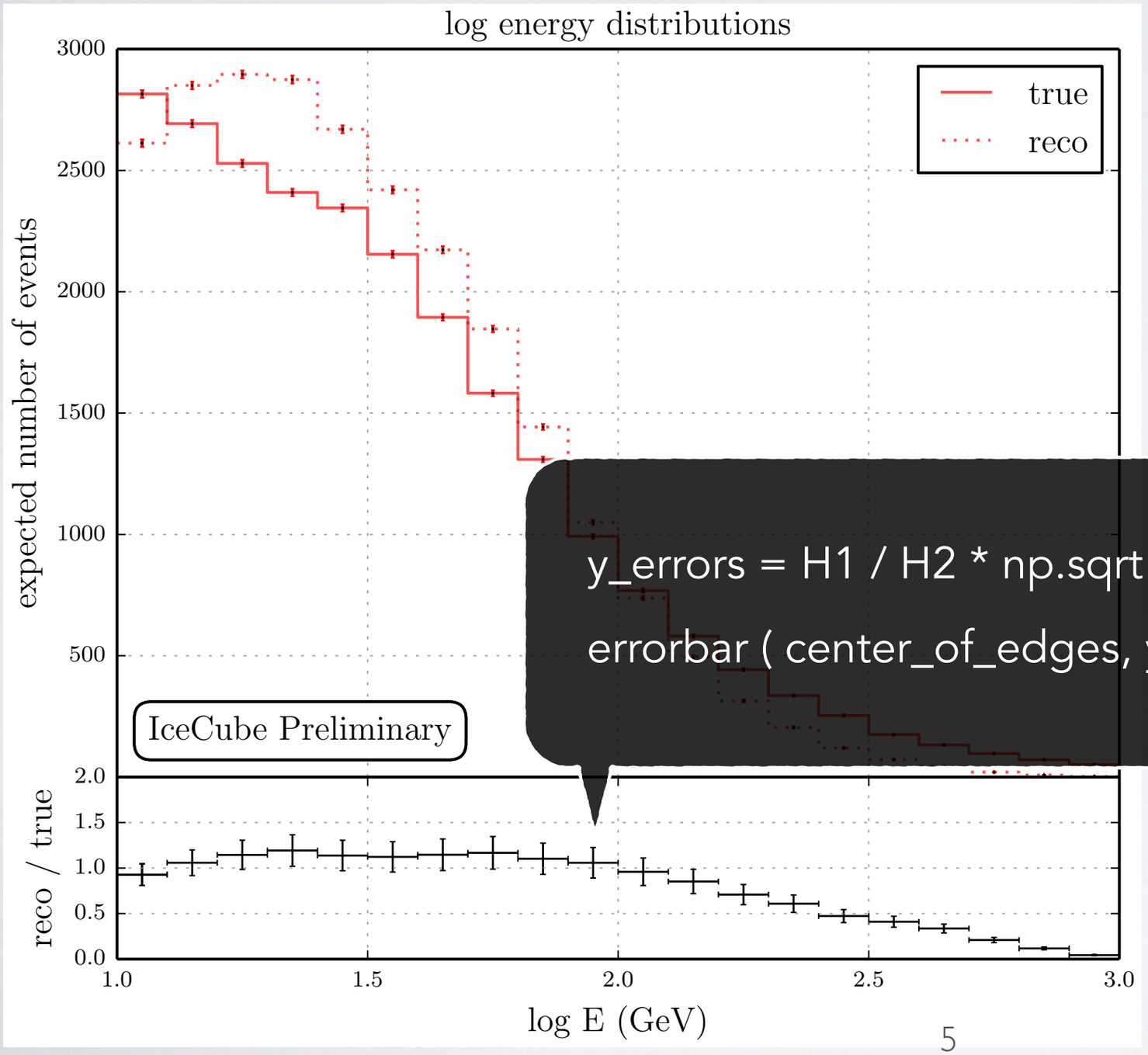

```
H_w2, edges  = np.histogram ( data, weights=weights**2 )
y_errors = np.sqrt ( H_w2 )
errorbar ( center_of_edges, yerr=y_errors )
```

for more info about plt.errorbar: here

Errorbars for ratio of two histograms

Ratio = H_1/H_2

Error on ratio in this bin = $\frac{H_1}{H_2} * \sqrt{\left(\frac{H_{1_{w2}}}{H_1}\right)^2 + \left(\frac{H_{2_{w2}}}{H_2}\right)^2}$

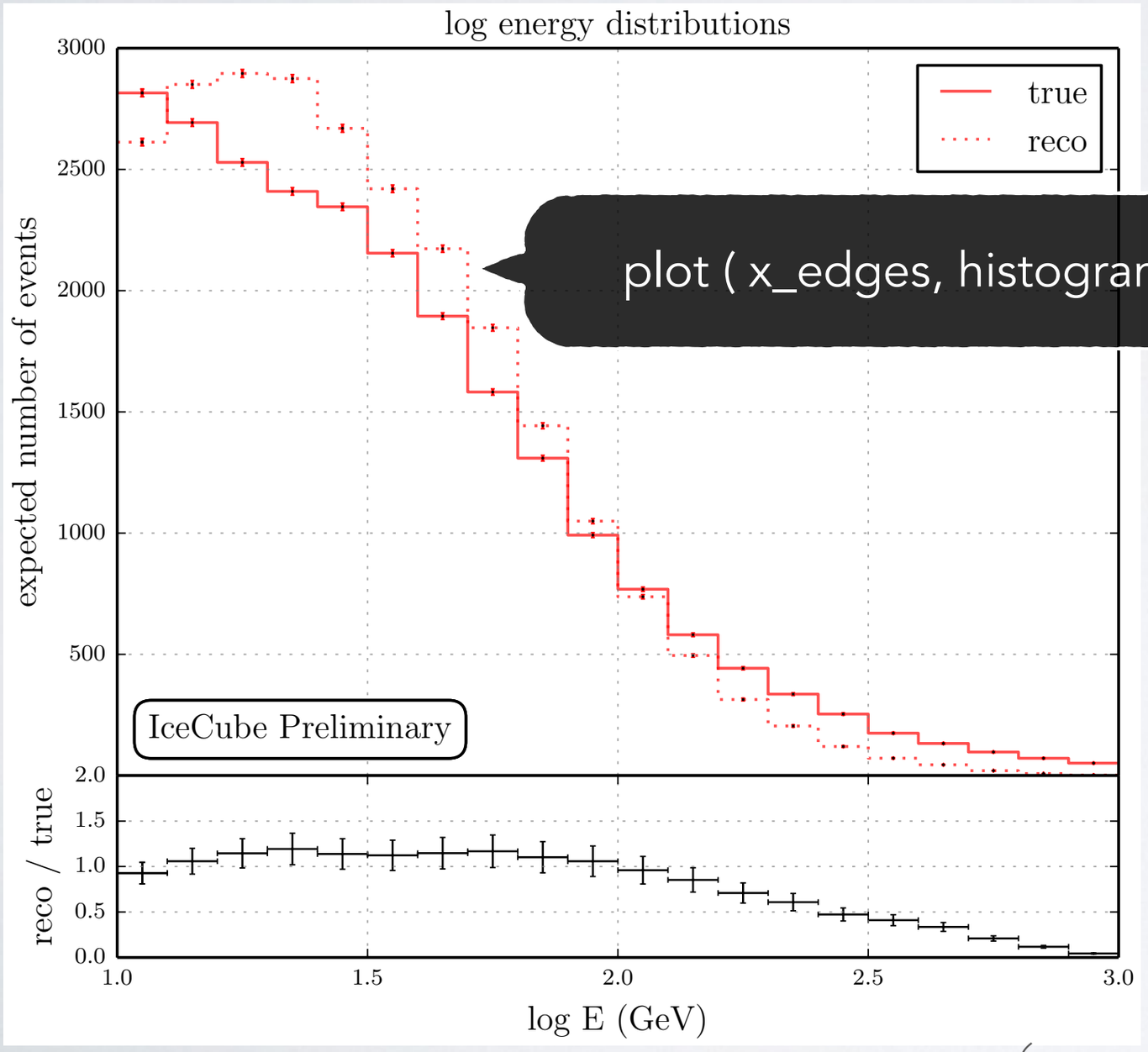



```
y_errors = H1 / H2 * np.sqrt ( (H1_w2/H1)**2 + (H2_w2/H2)**2 )

errorbar ( center_of_edges, yerr=y_errors )
```

Color Scheme: color blind and black_and_white_print proof

- Different line styles (solid, dashed, dotted, etc …)
- Palettable package (**palettable.colorbrew.sequential.whatever_color_map**)



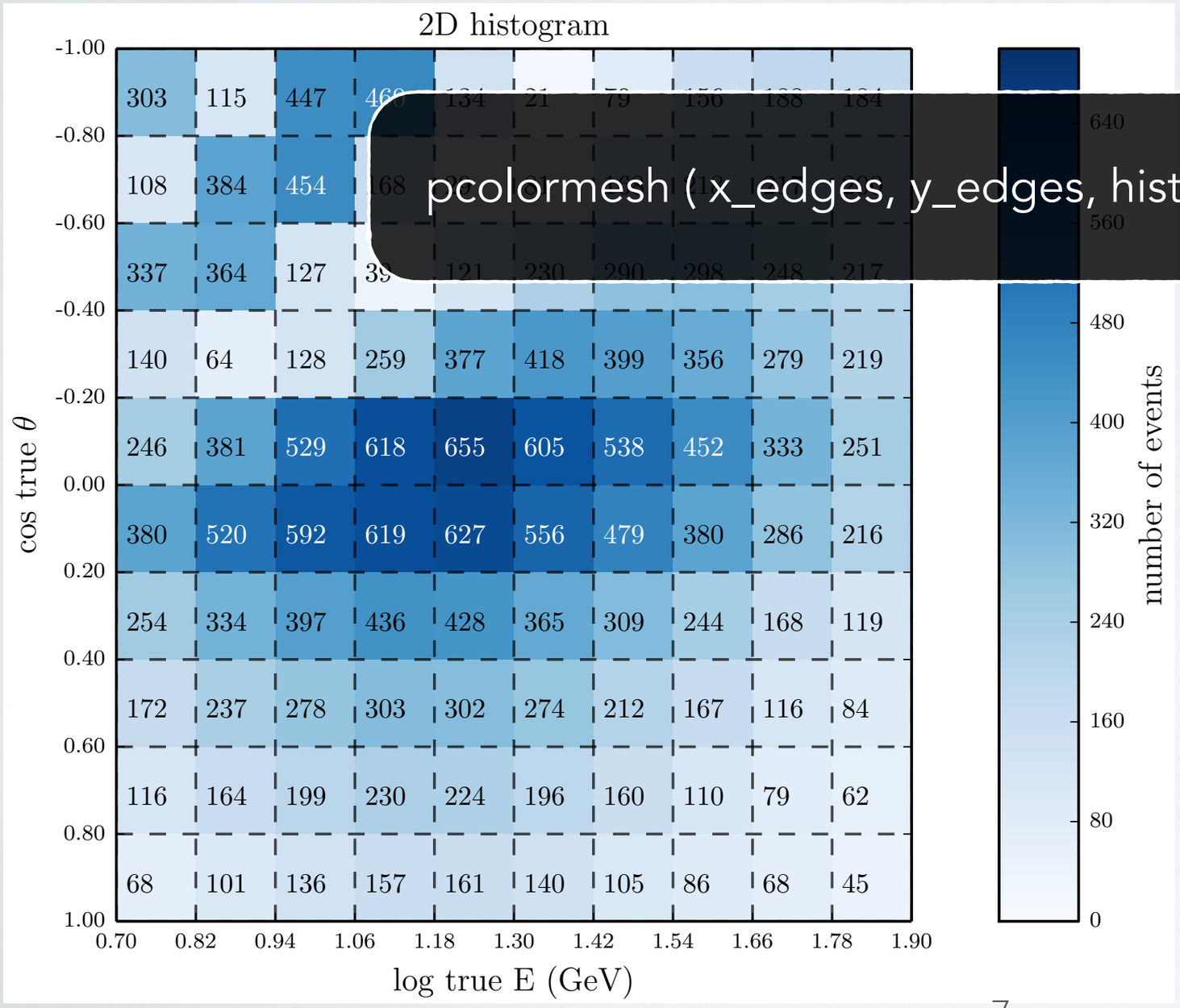
plot (x_edges, histogram, linestyle='..', linewidth=2, alpha=1)

Linestyle	ls argument
solid	'-'
dashed	'- -'
dotted	'..'
dashdot	'-.'

for more info about palettable: here

Color Scheme: color blind and black_and_white_print proof

- Gradient / sequential color map (e.g. plt.get_cmap ('Blues'))
- Palettable package (palettable.colorbrew.sequential.whatever_color_map)



pcolormesh (x_edges, y_edges, histogram, cmap=plt.get_cmap('Blues'))

colors	color argument
blue	'Blues'
orange to red	'OrRd'
yellow to green to blue	'YlGnBu'

see cmap reference: here

Fonts with Latex

- Customize plot style for all plots

  matplotlib.rcParams [key] = value

  matplotlib.rc (name, **style_arguments)

Examples:

- to set all lines to be red

  matplotlib.rcParams ['lines.color'] = 'red'

- to customize plot style for all line objects

  matplotlib.rc ('lines', linewidth=2 , color='red')

- to restore default values

  matplotlib.rcdefaults ()

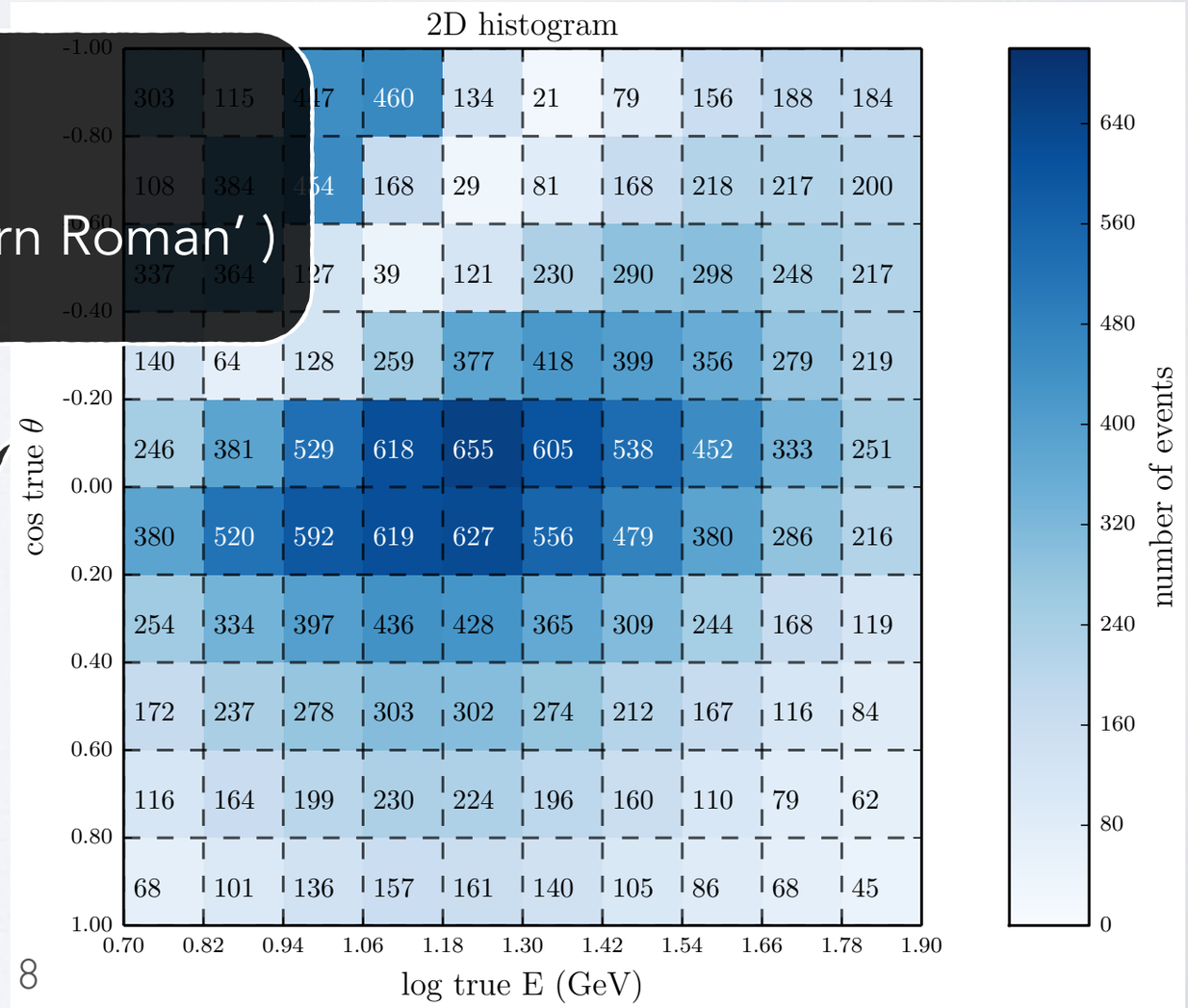
matplotlib.rc ('font', family='serif')

matplotlib.rc ('font', serif='Computer Modern Roman')

matplotlib.rc ('text', usetex=True)

** syntax: r'cos true θ'

see font manager reference: here



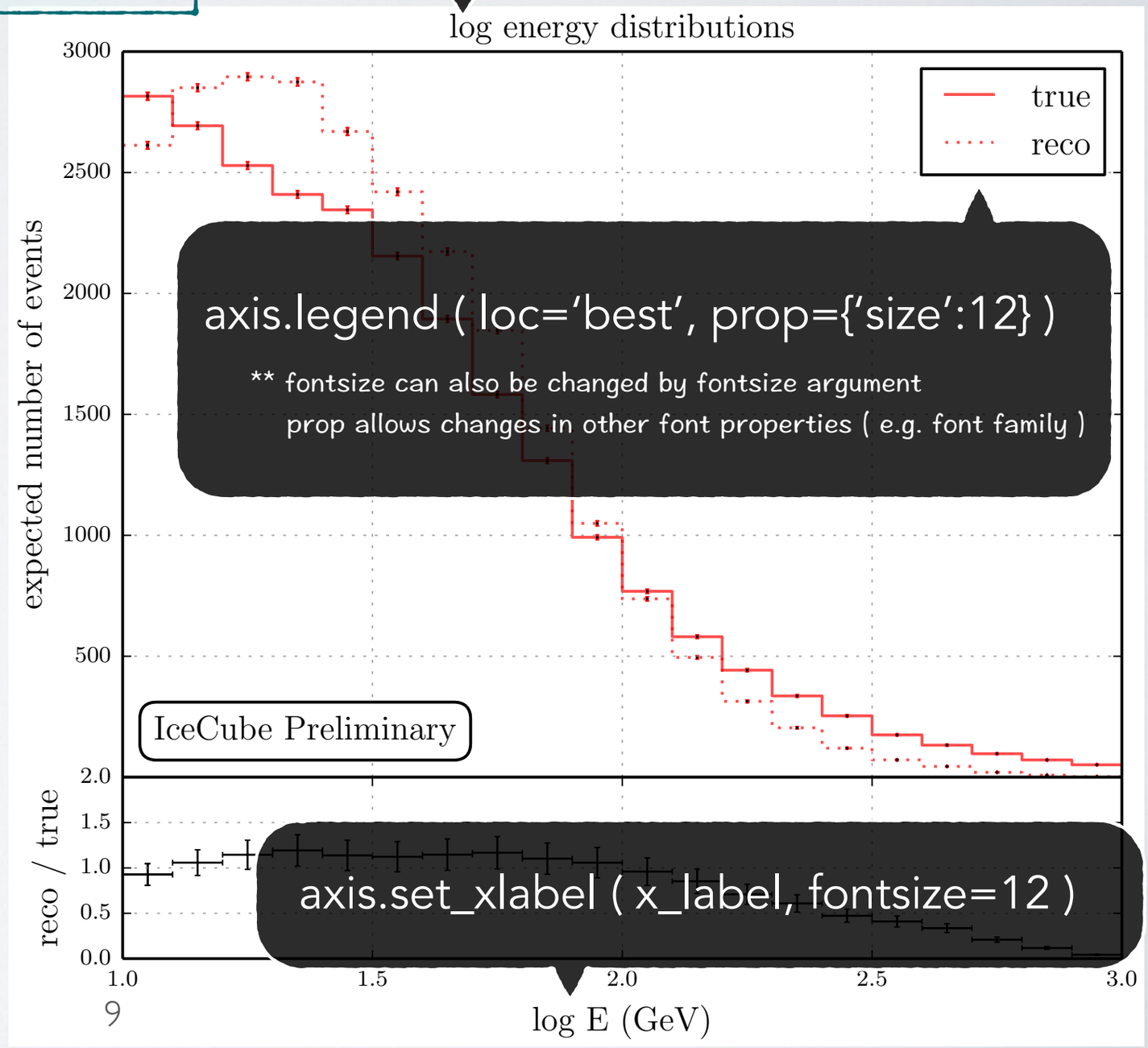
Labels, Legend, Title

- Labels with units
- Legend with fontsize as big as labels
- Title

location	loc argument
best	0 or ‘best’
upper right	1 or ‘upper right’
lower left	3 or ‘lower left
center	10 or ‘center’

see legend loc and other reference: here

axis.set_title (title, fontsize=12)



axis.legend (loc=‘best’, prop={‘size’:12})

** fontsize can also be changed by fontsize argument
prop allows changes in other font properties (e.g. font family)

axis.set_xlabel (x_label, fontsize=12)

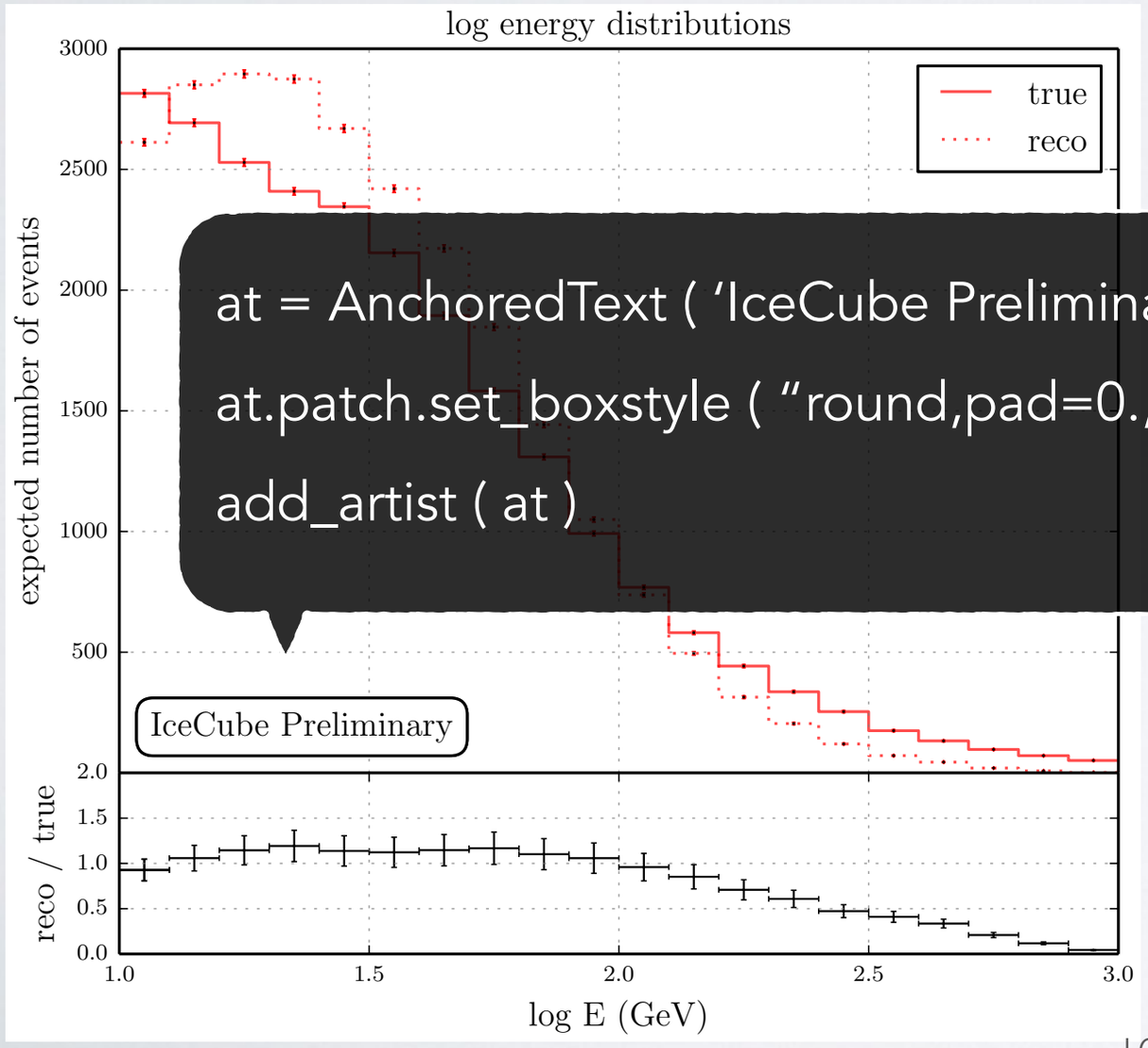
Text, Anchored, Annotate

- Text

  text (x, y, ‘some string’)

- AnchoredText - include text and drawings from mpl_toolkits

  from mpl_toolkits.axes_grid.anchored_artists import AnchoredText




```
at = AnchoredText ( ‘IceCube Preliminary’, prop=dict(size=12), frameon=True, loc=3 )
at.patch.set_boxstyle ( “round,pad=0., rounding_size=0.5” )
add_artist ( at )
```

for more info about plt.text () : here

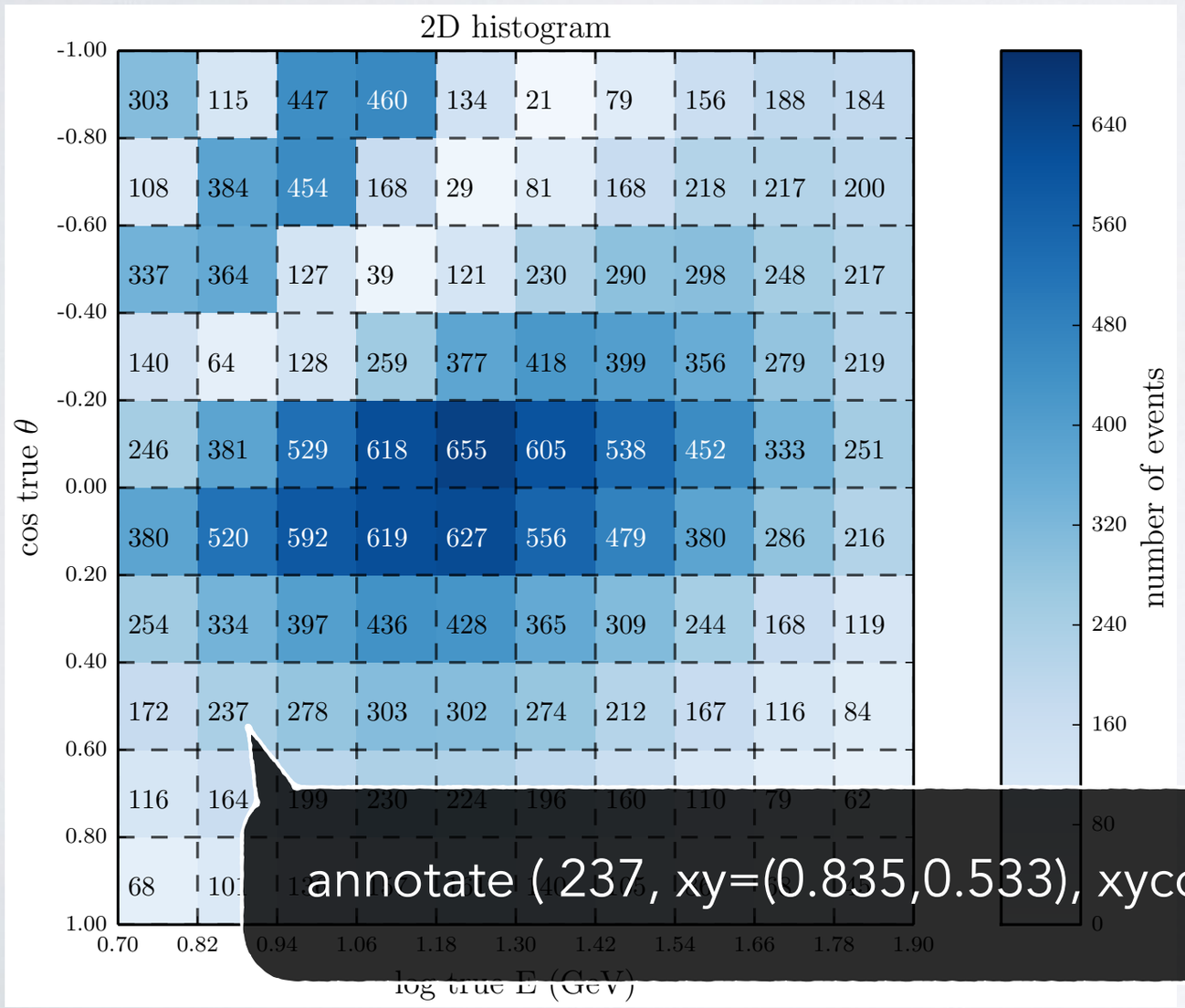
Text, Anchored, Annotate

- Text - data coordinates by defalt

  text (x, y, ‘some string’)

- Annotate

  annotate (‘some string’, xy=(x,y), xycoords=‘data’)



xycoords	explanation
data	based on x, y limits
axes	(0,0) = bottom left of axes (1, 1) = upper right of axes
figure	(0,0) = bottom left of figure (1, 1) = upper right of figure
display	(0,0) = bottom left of display (1, 1) = upper right of display

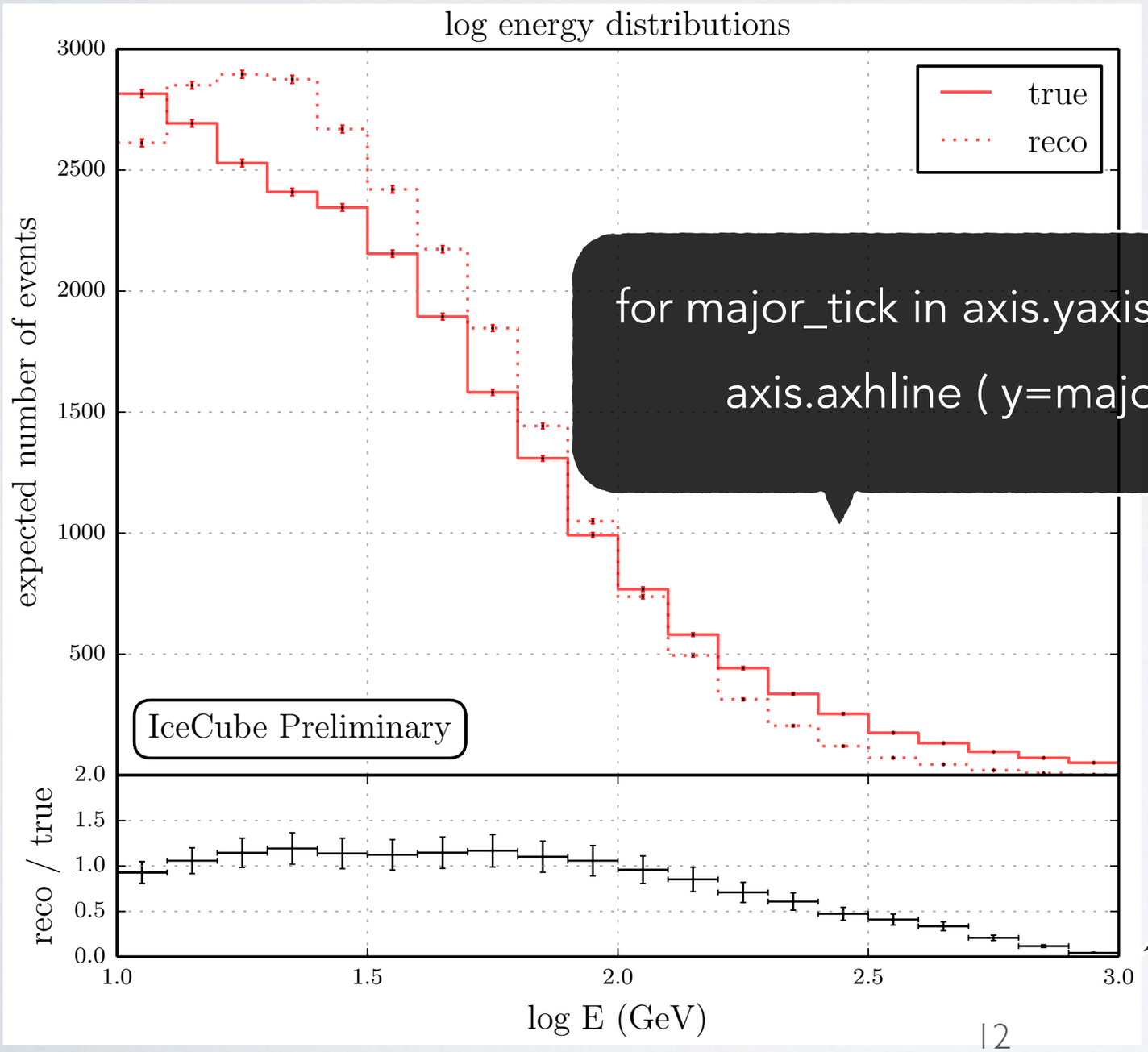
for more info about plt.annotate () : here

annotate (237, xy=(0.835,0.533), xycoords=‘data’, color=‘black’, fontsize=10)

Axis ticks

- Limits
- Gridline: a line at every major ticks

```
axhline ( y=y_value ); axvline ( x=x_value )
```




```
for major_tick in axis.yaxis.get_majorticklocs ():
    axis.axhline ( y=major_tick, ls='..', color='grey', alpha=0.7 )
```

```
axis.set_xlim ( xmin, xmax )
```

Axis ticks

- Customize tick parameters

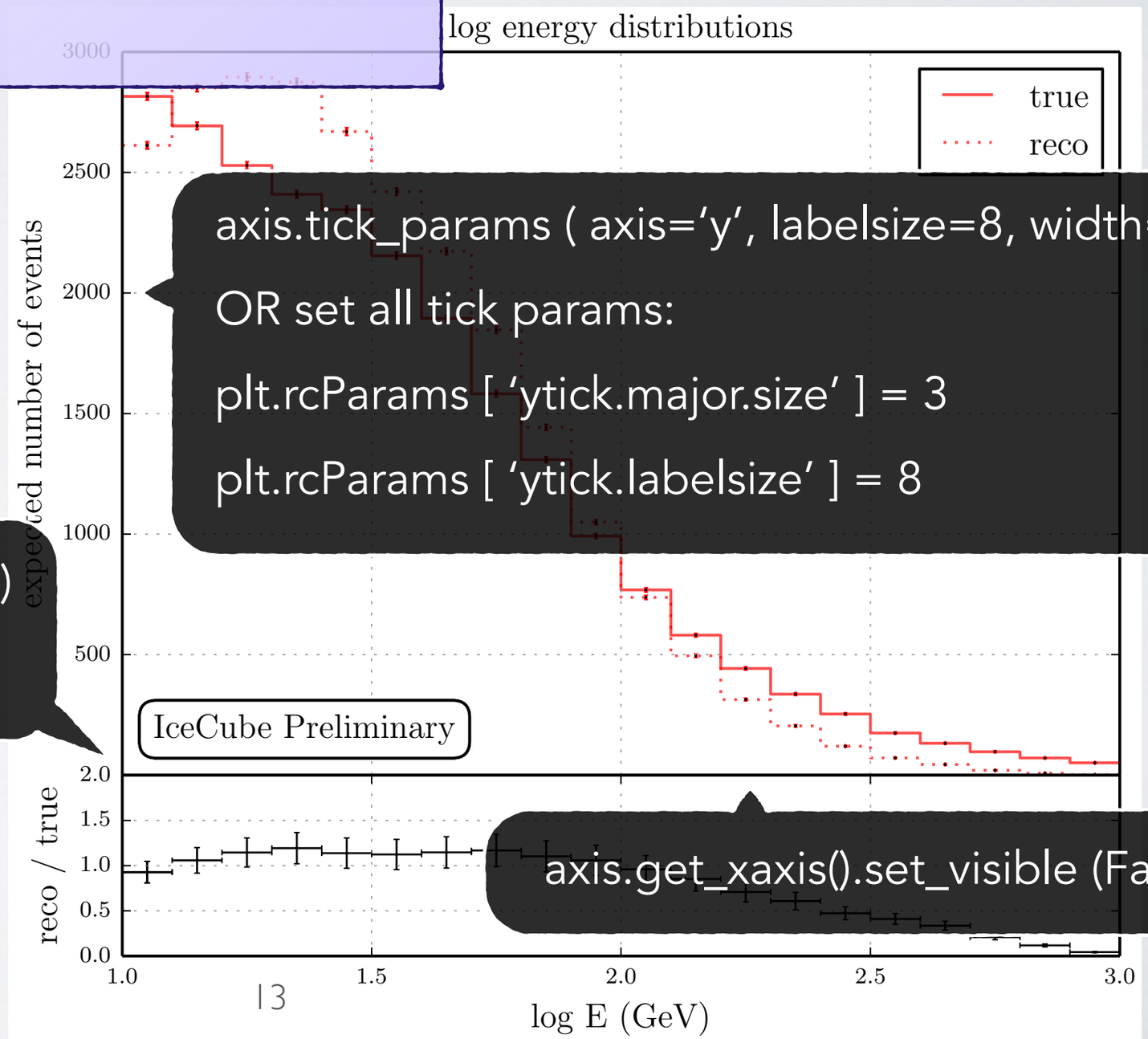
  axis.tick_params (axis='y', labelsize=12, width=5)

- Customize tick visibility

  axis.get_xaxis().set_visible (False)

  axis.get_xaxis().set_ticks ([ ])

for more info about tick_params () : here



axis.tick_params (axis='y', labelsize=8, width=3)

OR set all tick params:

plt.rcParams ['ytick.major.size'] = 3

plt.rcParams ['ytick.labelsize'] = 8

yticks = axis.yaxis.get_major_ticks()

yticks[0].set_visible (False)

axis.get_xaxis().set_visible (False)

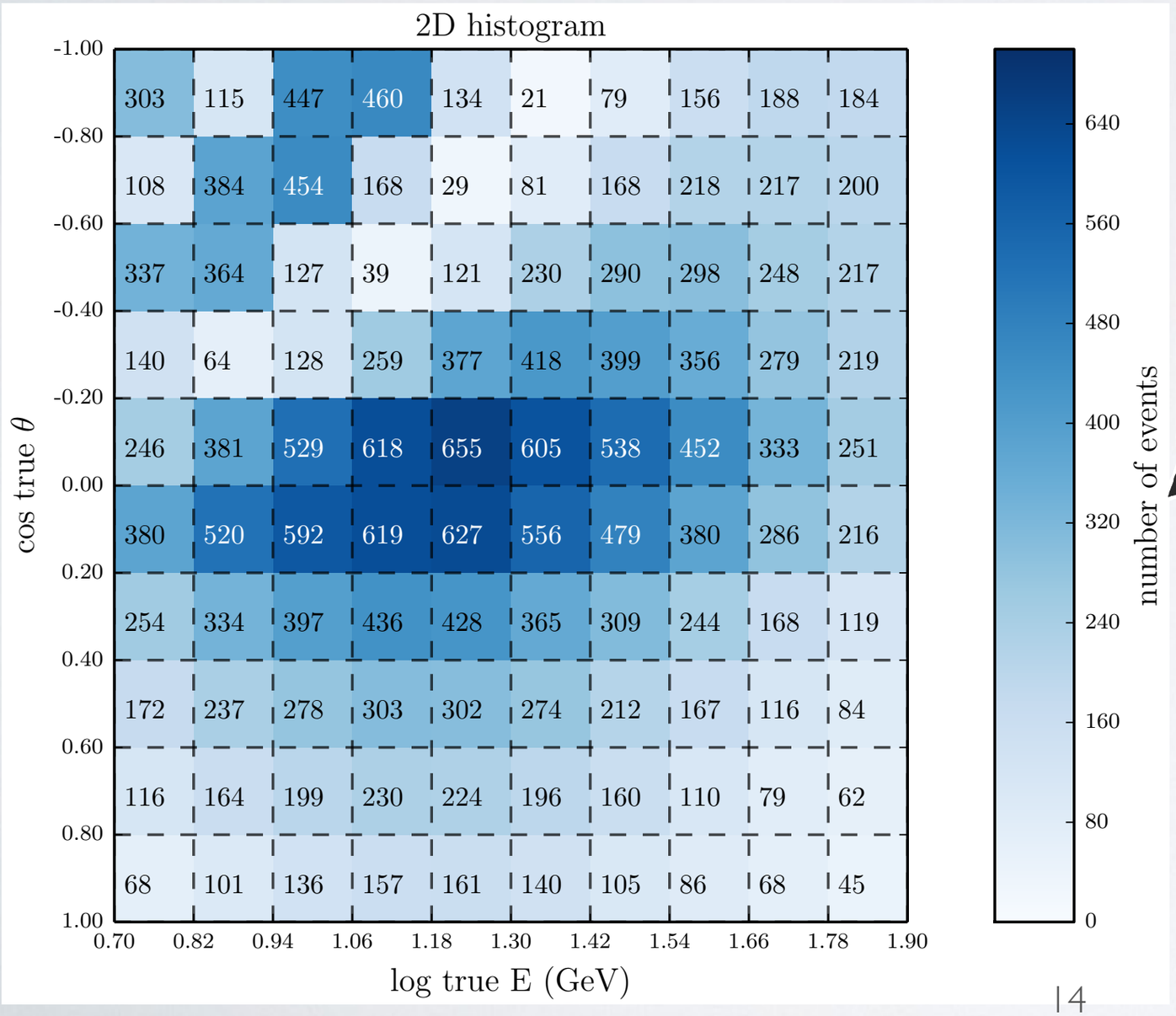
Color bars

- Color bar limits - defined when generating 2D histogram

  pcolormesh (x_edges, y_edges, histogram, vmin=0, vmax=700)

- Customize tick parameters

  axis.tick_params ( )




```
cb = plt.colorbar ( cax=axis )
cb.ax.tick_params ( labelsize=8 )
cb.set_label ( 'label_string', fontsize=12 )
```

Sub plots

- Customize sub plots

  import matplotlib.gridspec as gridspec

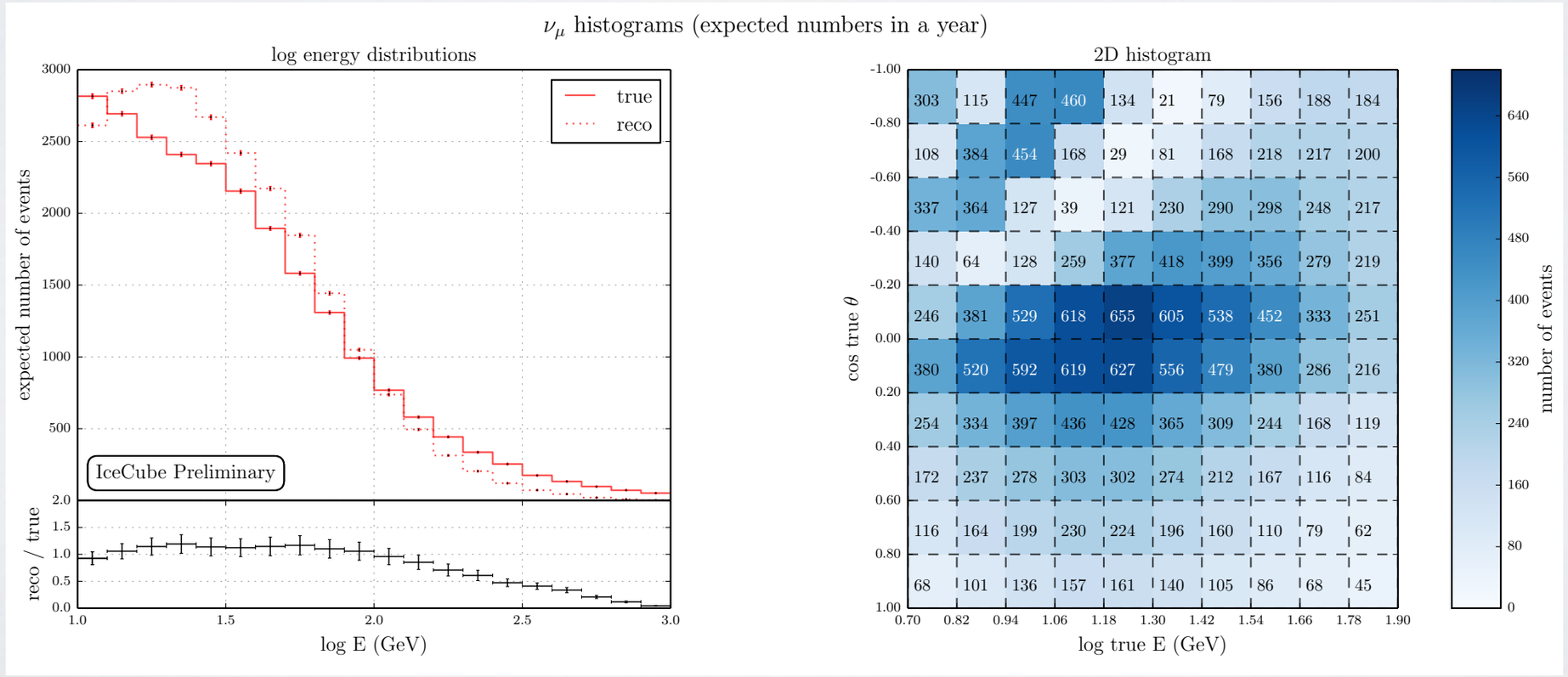
  gs = gridspec.GridSpec (n_rows, m_columns)

- Customize sub plots under a subplot

  gs0 = gridspec.GridSpecFromSubplotSpec (n_rows, m_columns, subplot_spec=gs[index])

- Define height/width spacing/ratio between plots

  gs.update (hspace=0.4, wspace=0.4, height_ratios=[4,1], width_ratios=[10,1])



Sub plots

for more info about matplotlib.gridspec () : here

- Customize sub plots

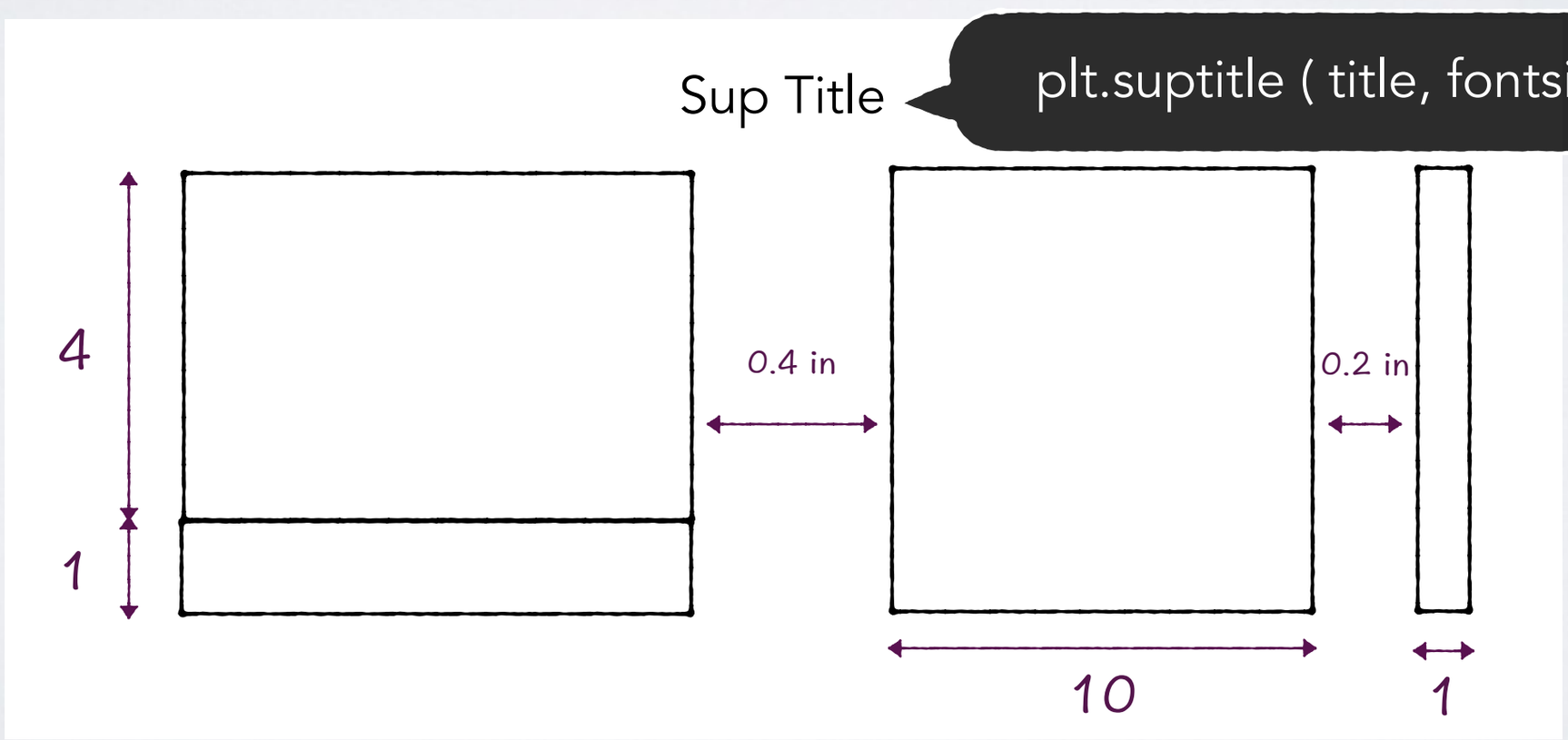
```
import matplotlib.gridspec as gridspec
gs = gridspec.GridSpec ( n_rows, m_columns )
```

- Customize sub plots under a subplot

```
gs0 = gridspec.GridSpecFromSubplotSpec ( n_rows, m_columns, subplot_spec=gs[index] )
```

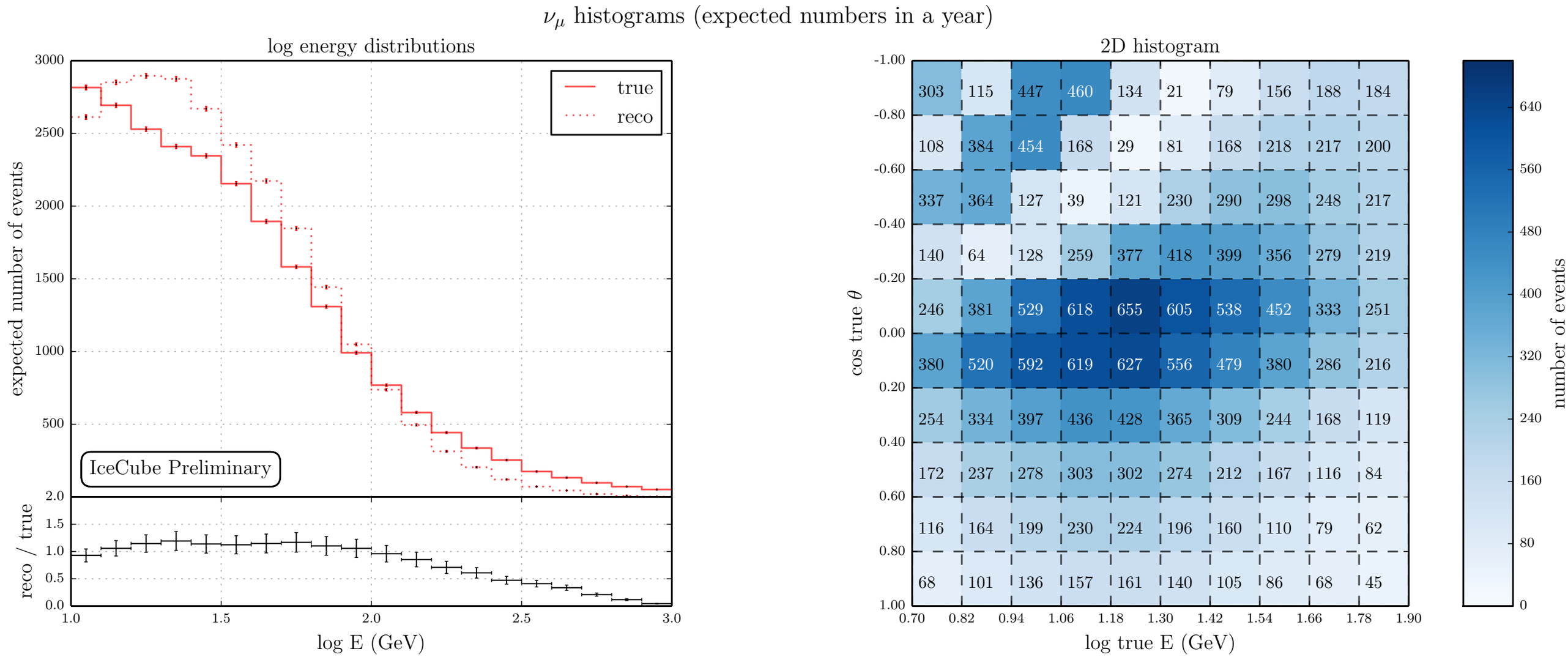
- Define height/width spacing/ratio between plots

```
gs.update ( hspace=0.4, wspace=0.4, height_ratios=[4,1], width_ratios=[10,1] )
```



Summary: your goal before lunch

plt.savefig (" /your/path/plot.pdf")



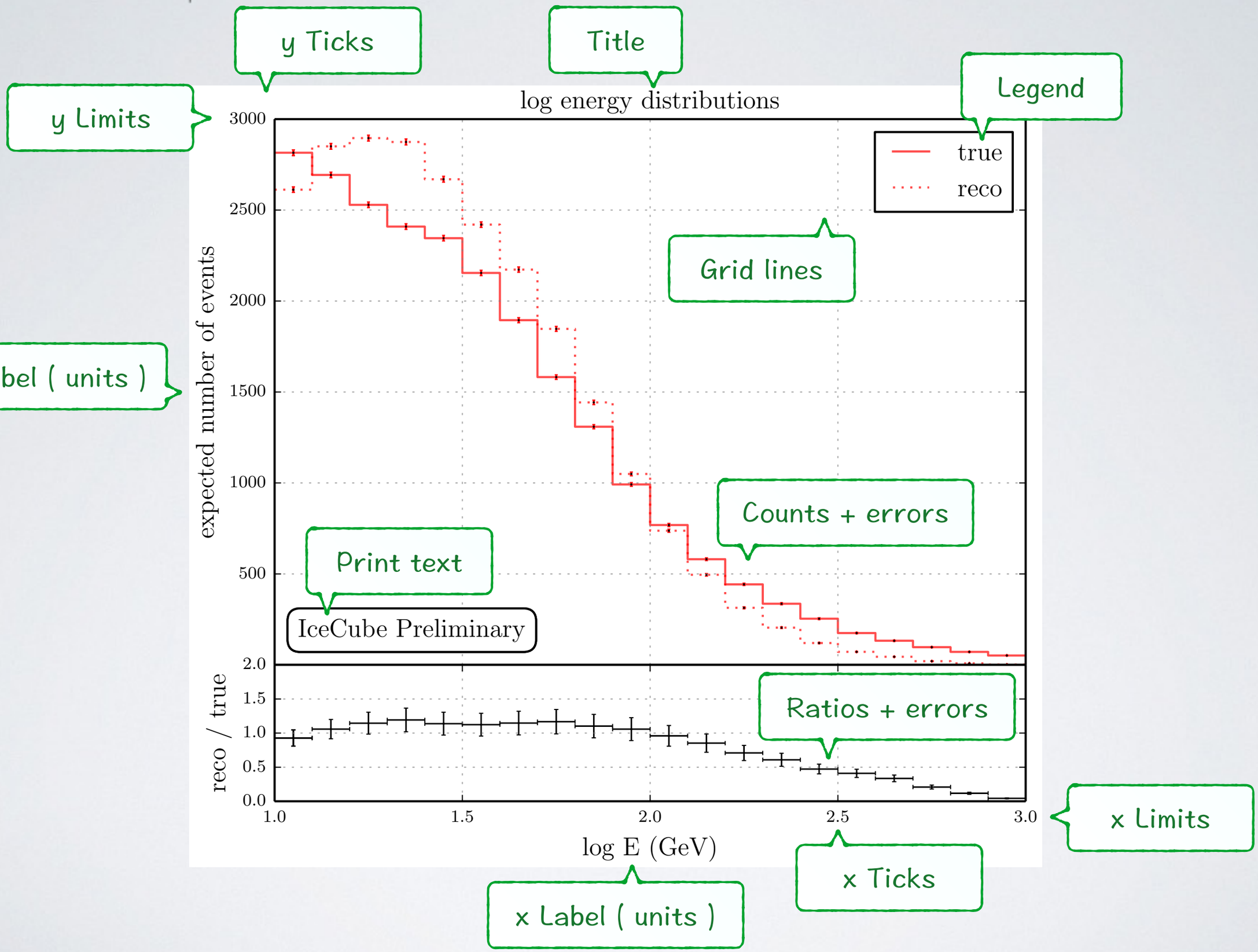
to see details of this plot: http://umdgrb.umd.edu/~elims/plots/bootcamp/histograms.pdf

template to start with: http://umdgrb.umd.edu/~elims/plots/bootcamp/plot_distributions_template.py

materials: here

** You don't have to follow my template.
Do whatever you want to your easiest understanding :)

1D histogram



2D histogram