Plotting results – python scripts

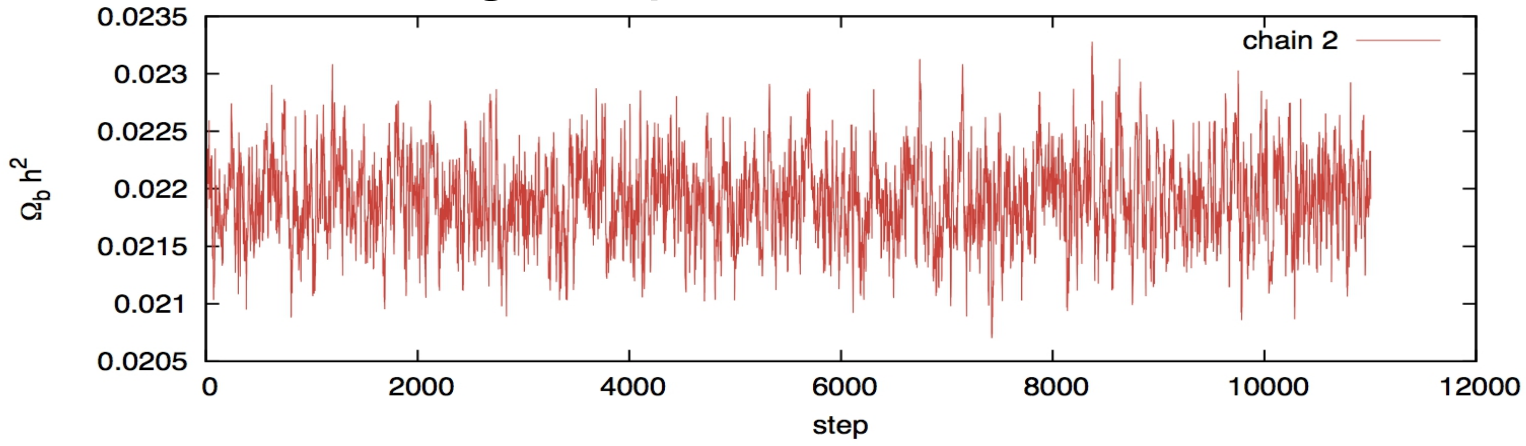
Claudia G. Scóccola

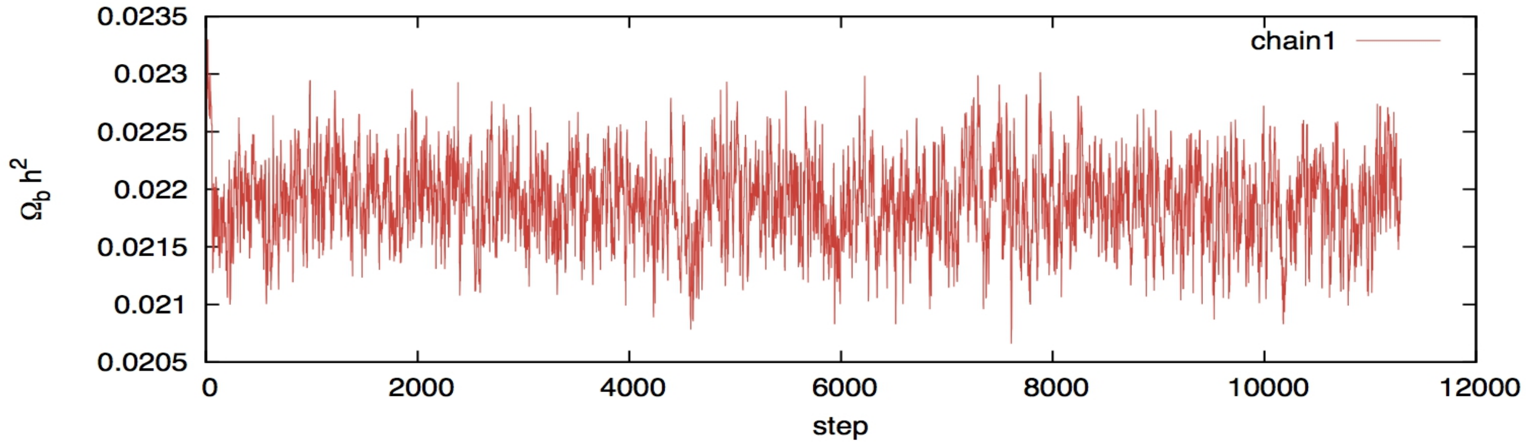
UAM/IFT

School on Cosmology Tools. 12-15 November 2013

Plotting results

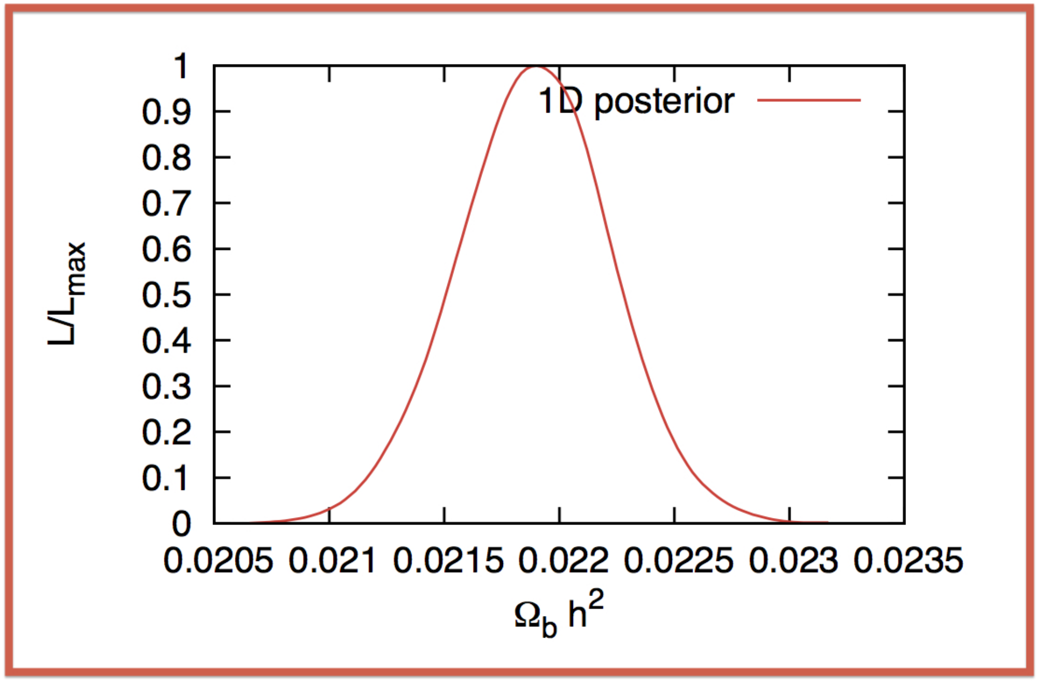
After Markov chains converge ==> The chains need to be analyzed in order to extract information, such as constraints on the cosmological parameters.

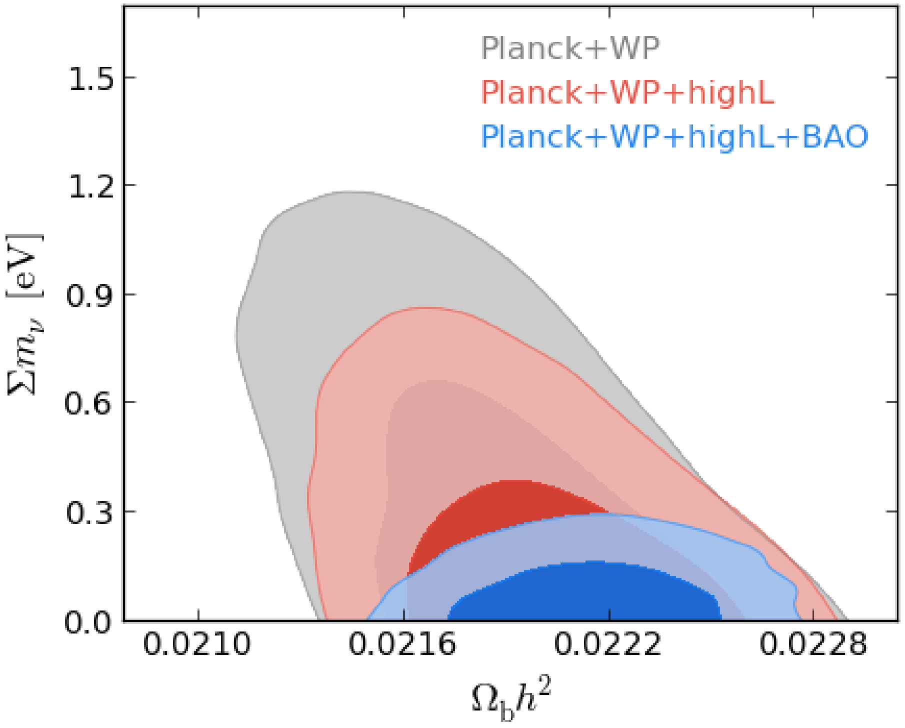




Plotting results

In general, one is interested in 1D and 2D (contours) posterior probability density, that represent the posterior probability density marginalized over all parameters except the one or two under study.





Plotting results

GetDist analyzes the chains. It produces several files with information about convergence, mean values and 1-sigma errors, Best Fit parameters...

It also produces the necessary files to make 1D and 2D plots (saved in plot_data/).

Python scripts

In the /python directory of the cosmomc distribution, there are several python scripts to plot results.

You can analyze the chains by yourself, and produce the 1D and 2D files to plot the results (with GNUplot, for example).

But, for now, we will see how to use the python scripts provided by cosmomc.

CosmoMC and GetDist with Planck Likelihood and Chains

http://cosmologist.info/cosmomc/readme_planck.html

To prepare a grid of models:

```
python python/makeGrid.py PLA my_settings
```

Then, you can run the chains. Even when you already have the chains, **makeGrid.py** is useful to inform the python scripts which is the available data.

(You need to edit settings_sample.py => my_settings.py)

PLA is the head directory where chains are.

To make plots from the grid you need to first run getdist over the grid:

```
python python/runGridGetdist.py PLA
--burn_removed
```

Plotting files in PLA/plot_data for the entire grid.

There are optional parameters to restrict to specific models. To see the full list of options, run:

python python/runGridGetdist.py

E.g.: if you are only interested in models with massive neutrinos using Planck data you could do

```
python python/runGridGetdist.py PLA
--paramtag base_mnu --data planck
```

You can use **python python/makePlots.py PLA output_dir** to make plots en masse (all 1D plots at once).

See **batch1/outputs/makeGridPlots** for examples.

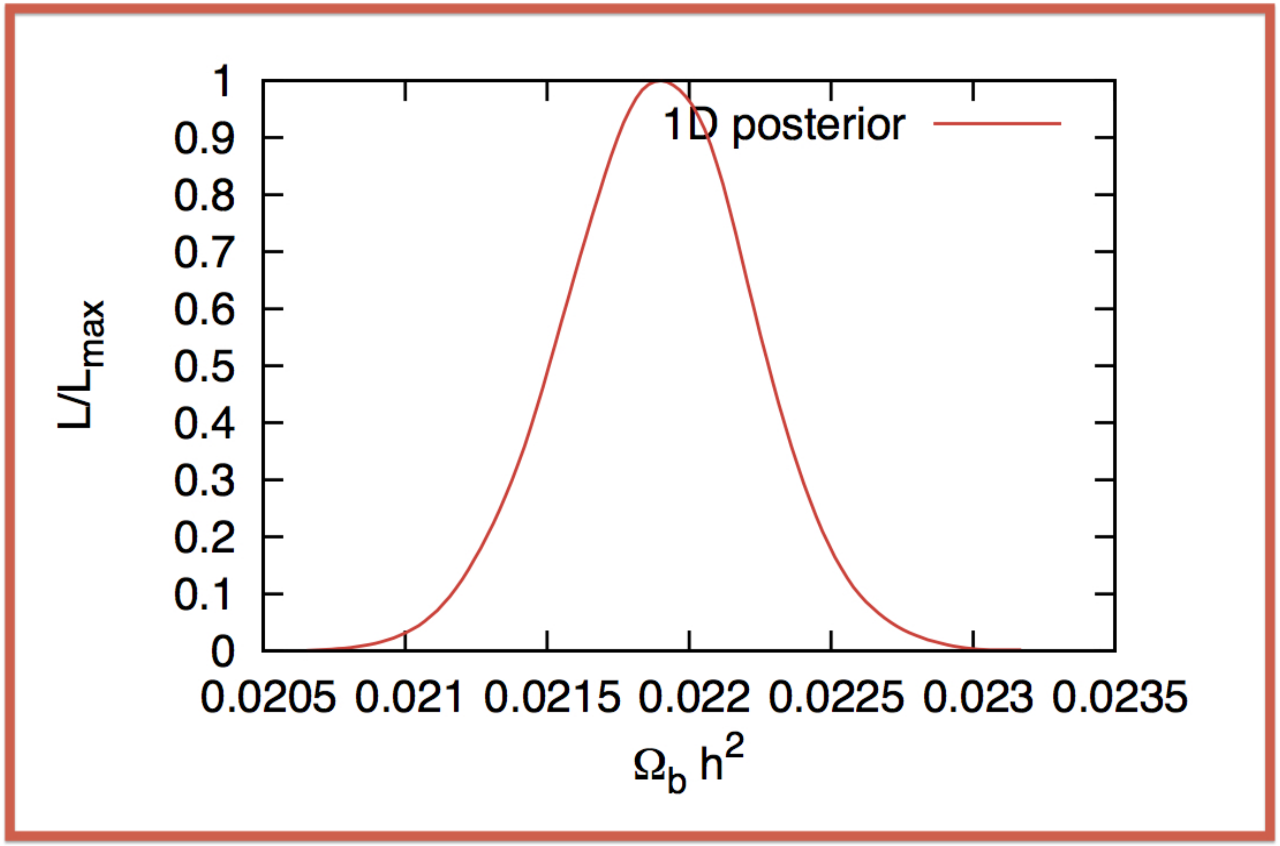
You can also write your own short python scripts to plot things of interest; many examples are in **batch1/outputs/** (note that these use planckStyle.py, which assumed your main grid directory is ./main not ./PLA, and needs output folders called outputs and plots).

You can also make scripts that do not use planck settings.

Example: Plotting 1D posterior with gnuplot

>> gnuplot plot_1D.gnu

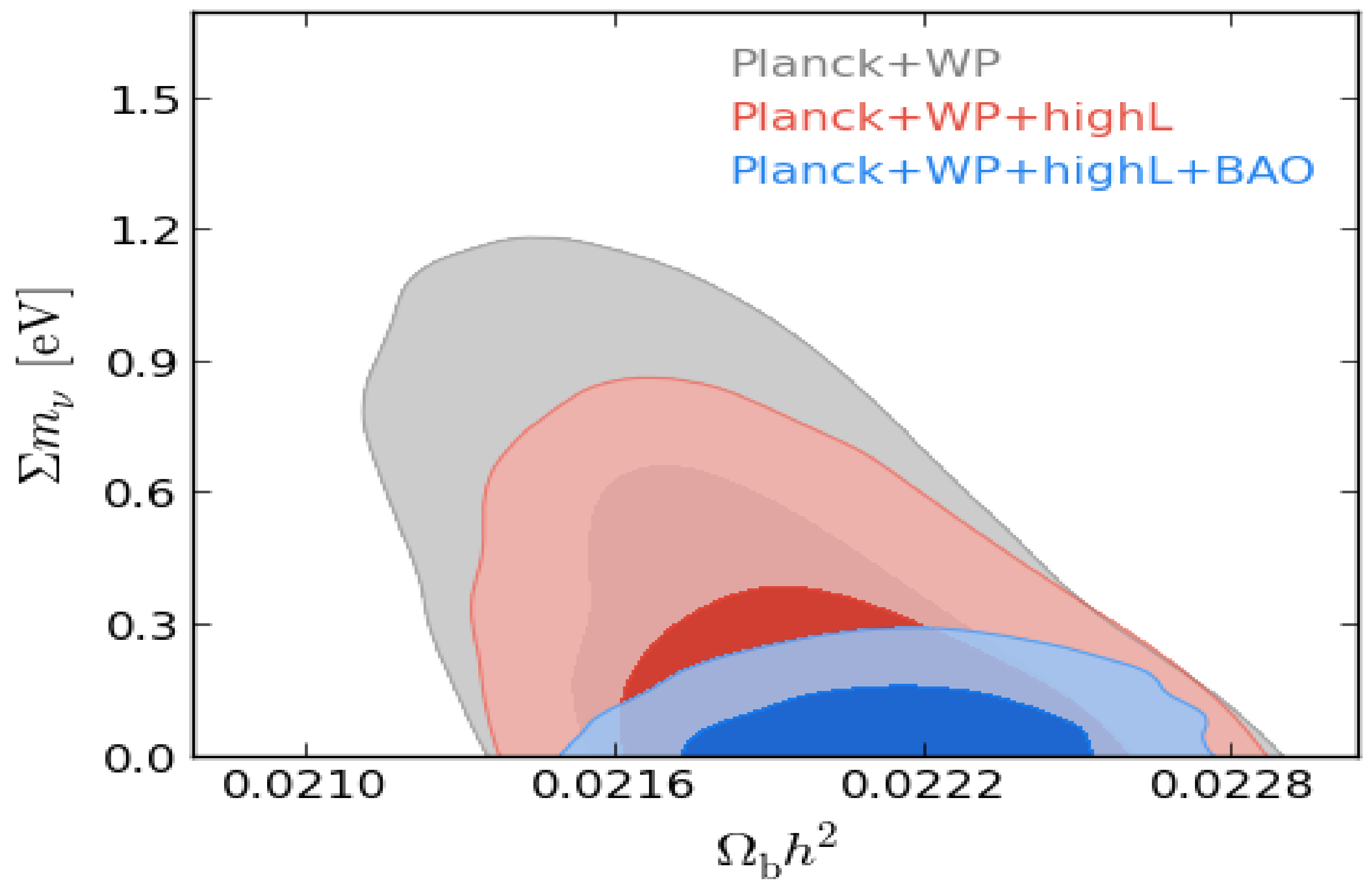
```
set term jpeg  enhanced

set output 'posterior_1D_omb.jpg'

set size 0.5,0.5

set xlabel '{/Symbol W}_b h^2'
set ylabel 'L/L_{max}'

file = '../base_mnu_planck_lowl_lowLike_p_omegabh2.dat'

pl file us 1:2 title '1D posterior' lw 2 w l
```


1D posterior
L/L$_{max}$
$\Omega_b h^2$
0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1
0.0205, 0.021, 0.0215, 0.022, 0.0225, 0.023, 0.0235

Example: Plotting 2D contours with python

>> python plot_2D_contour.py

```
import GetDistPlots
g=GetDistPlots.GetDistPlotter('main/plot_data')
g.settings.setWithSubplotSize(4)
g.make_figure(xstretch=1.3)

roots =
['base_mnu_planck_lowl_lowLike','base_mnu_planck_lowl_l
owLike_highL','base_mnu_planck_lowl_lowLike_highL_post
_BAO']
g.plot_2d(roots, param_pair=['omegabh2','mnu'], filled=True)
g.add_legend(['Planck+WP','Planck+WP+highL','Planck+WP
+highL+BAO'],legend_loc='upper right',colored_text=True);
g.export('omegabh2_mnu.png')
```



Planck+WP
Planck+WP+highL
Planck+WP+highL+BAO
Σm_ν [eV]
0.0
0.3
0.6
0.9
1.2
1.5
0.0210
0.0216
0.0222
0.0228
$\Omega_b h^2$