Solution to running JLA

Benjamin Audren

École Polytechnique Fédérale de Lausanne

08/10/2014

Parameter file

```
data.experiments=['JLA']

# Cosmological parameters list
data.parameters['Omega_cdm'] = [0.2562,   None, None, 0.008,   1, 'cosmo']

# Nuisance
data.parameters['alpha']    = [0.15 ,   None, None, 0.001, 1, 'nuisance']
data.parameters['beta']     = [3.559,   None, None, 0.02,   1, 'nuisance']
data.parameters['M']        = [-19.02, None, None, 0.004,   1, 'nuisance']
data.parameters['Delta_M'] = [-0.10,   None, None, 0.004,   1, 'nuisance']

# Derived parameter list
data.parameters['Omega_m'] = [0,     -1, -1, 0,1,   'derived']


data.cosmo_arguments['Omega_b'] = 0.05
data.cosmo_arguments['h'] = 0.70
data.cosmo_arguments['T_cmb'] = 2.726
data.cosmo_arguments['N_eff'] = 3.046
data.cosmo_arguments['N_ncdm'] = 0
#------ Mcmc parameters -----
# Number of steps taken, by default (overwritten by the -N command)
data.N=10
# Number of accepted steps before writing to file the chain. Larger means less
# access to disc, but this is not so much time consuming.
data.write_step=5
```

Running a first test run

```
mpirun -np 4 python montepython/MontePython.py run -o chains/jla -p jla.param \
       -N 1000
```

Analyzing these four chains

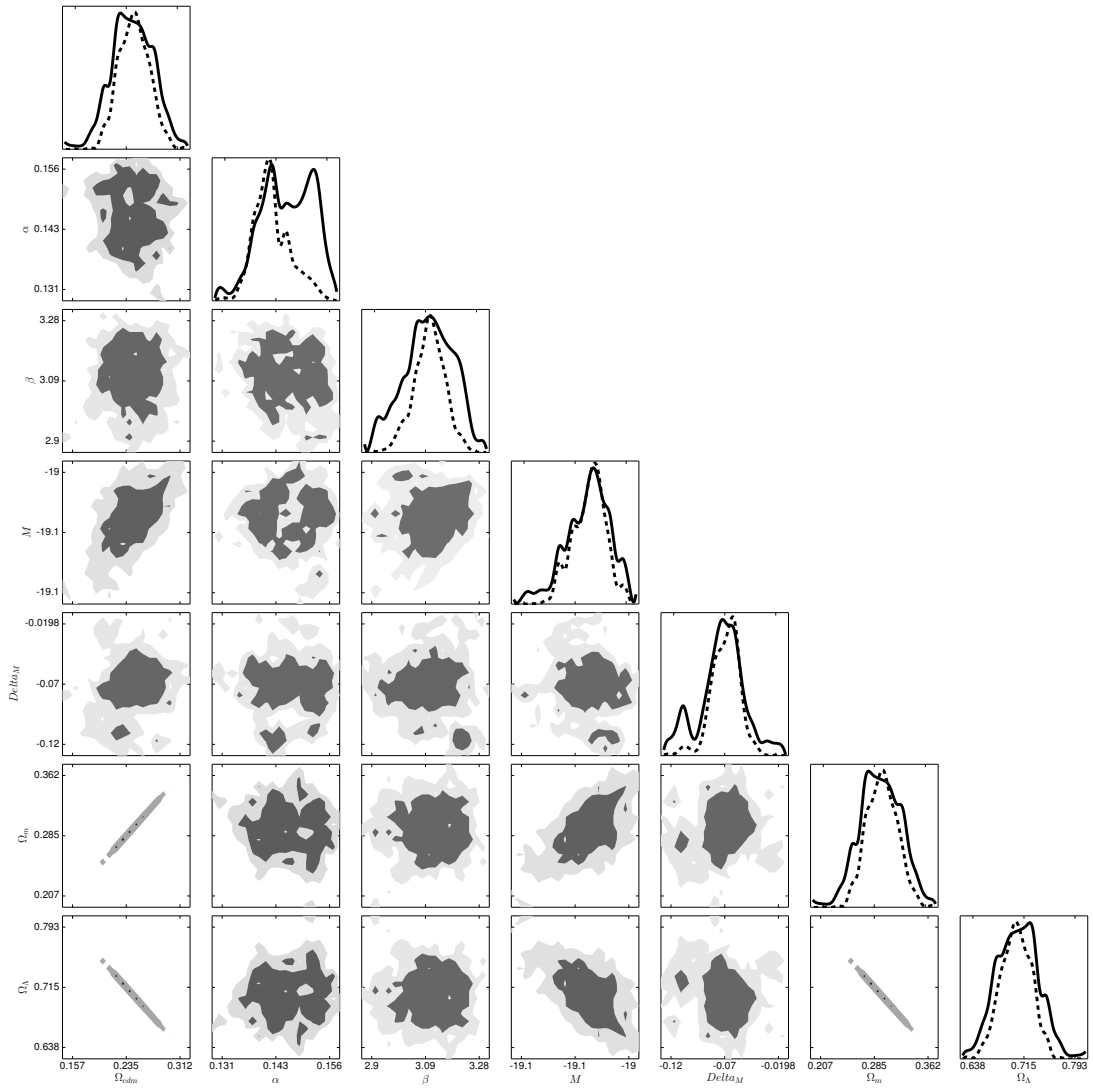
```
python montepython/MontePython info chains/jla
```

Output of analyse

```
Running Monte Python v2.1.0

---> Finding global maximum of likelihood
---> Removing burn-in
---> Scanning file chains/jla_full/2014-10-07_100000__3.txt : Removed 152 points of burn-in
                                   2014-10-07_100000__4.txt : Removed 79  points of burn-in
                                   2014-10-07_100000__1.txt : Removed 50  points of burn-in
                                   2014-10-07_100000__2.txt : Removed 61  points of burn-in
---> Computing mean values
---> Computing variance
---> Computing convergence criterium (Gelman-Rubin)
 -> R is 0.180485        for  Omega_cdm
         0.127232        for  alpha
         0.042833        for  beta
         0.081718        for  M
         0.452856        for  Delta_M
         0.180535        for  Omega_m
         0.180485        for  Omega_Lambda
---> Computing covariance matrix
-------------------------------------------------------
 -> Computing histograms for  Omega_cdm
 /!\ could not derive minimum credible intervals for this multimodal posterior
 -> Computing histograms for  alpha
 -> Computing histograms for  beta
 -> Computing histograms for  M
 -> Computing histograms for  Delta_M
 -> Computing histograms for  Omega_m
 -> Computing histograms for  Omega_Lambda
-------------------------------------------------------
---> Saving figures to .pdf files
---> Writing .info and .tex files
```

Triangle plot

in plots subfolder



Using a better starting point and proposal density

To keep them !

```
cp chains/jla/jla.covmat chains
cp chains/jla/jla.bestfit chains

mpirun -np 8 python montepython/MontePython.py run -o chains/jla -p jla.param \
       -N 20000 -c chains/jla.covmat -b chains/jla.bestfit
```

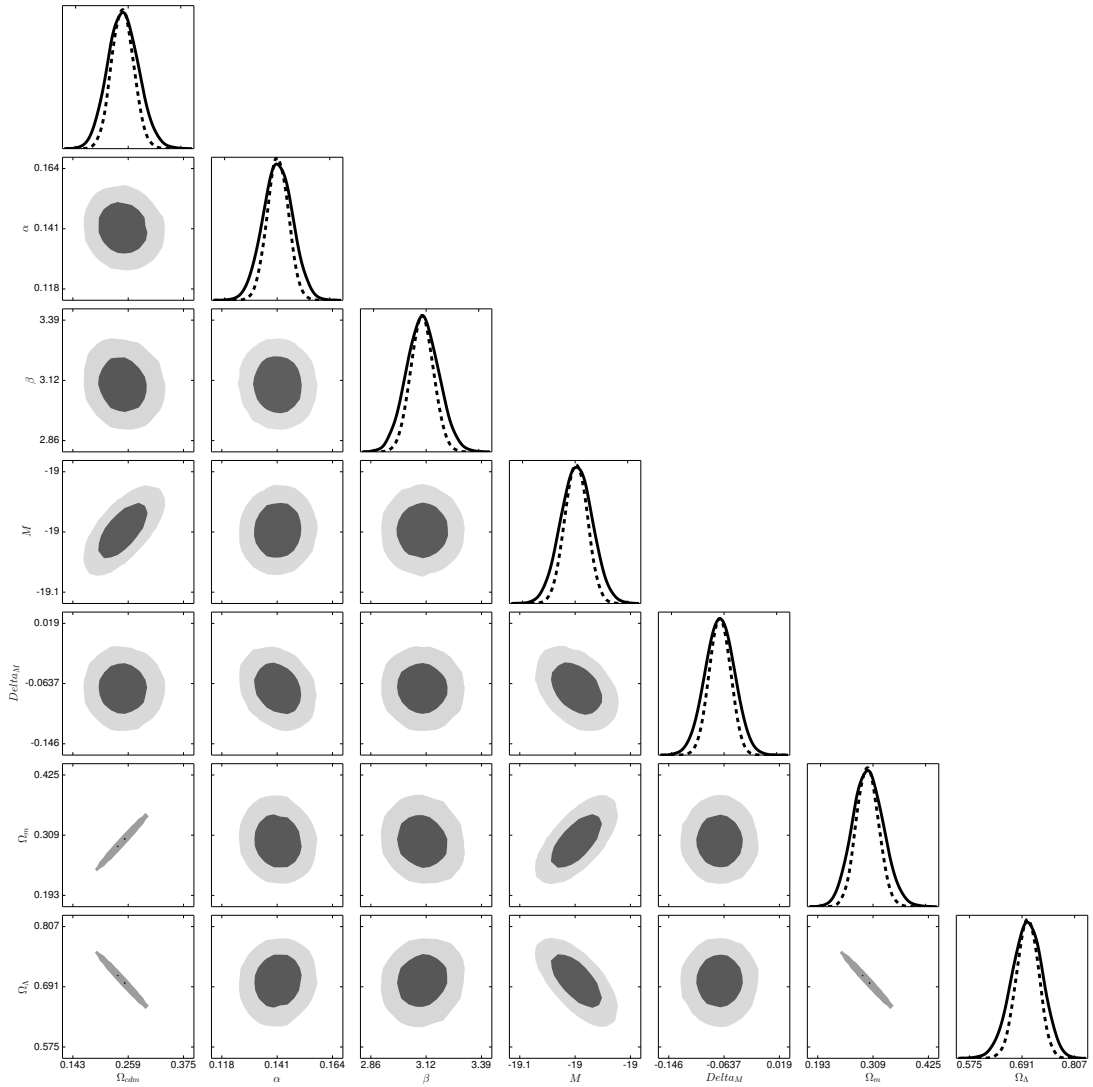
Analyse

```
python montepython/MontePython info chains/jla/*20000*
```

Output of analyse

```
Running Monte Python v2.1.0

---> Finding global maximum of likelihood
---> Removing burn-in
---> Scanning file chains/jla_full/2014-10-07_200000__9.txt   : Removed 0   points of burn-in
...
                                    2014-10-07_200000__6.txt   : Removed 0   points of burn-in
---> Computing mean values
---> Computing variance
---> Computing convergence criterium (Gelman-Rubin)
 -> R is 0.000911        for  Omega_cdm
         0.000629        for  alpha
         0.000457        for  beta
         0.000575        for  M
         0.000679        for  Delta_M
         0.000911        for  Omega_m
         0.000911        for  Omega_Lambda
---> Computing covariance matrix
-----------------------------------------------------
 -> Computing histograms for  Omega_cdm
 -> Computing histograms for  alpha
 -> Computing histograms for  beta
 -> Computing histograms for  M
 -> Computing histograms for  Delta_M
 -> Computing histograms for  Omega_m
 -> Computing histograms for  Omega_Lambda
-----------------------------------------------------
---> Saving figures to .pdf files
---> Writing .info and .tex files
```

Good plot


0.164
0.141
0.118
α
3.39
3.12
2.86
β
-19
-19
-19.1
M
0.019
-0.0637
-0.146
$Delta_M$
0.425
0.309
0.193
Ω_m
0.807
0.691
0.575
Ω_Λ
0.143 0.259 0.375
Ω_{cdm}
0.118 0.141 0.164
α
2.86 3.12 3.39
β
-19.1 -19 -19
M
-0.146 -0.0637 0.019
$Delta_M$
0.193 0.309 0.425
Ω_m
0.575 0.691 0.807
Ω_Λ

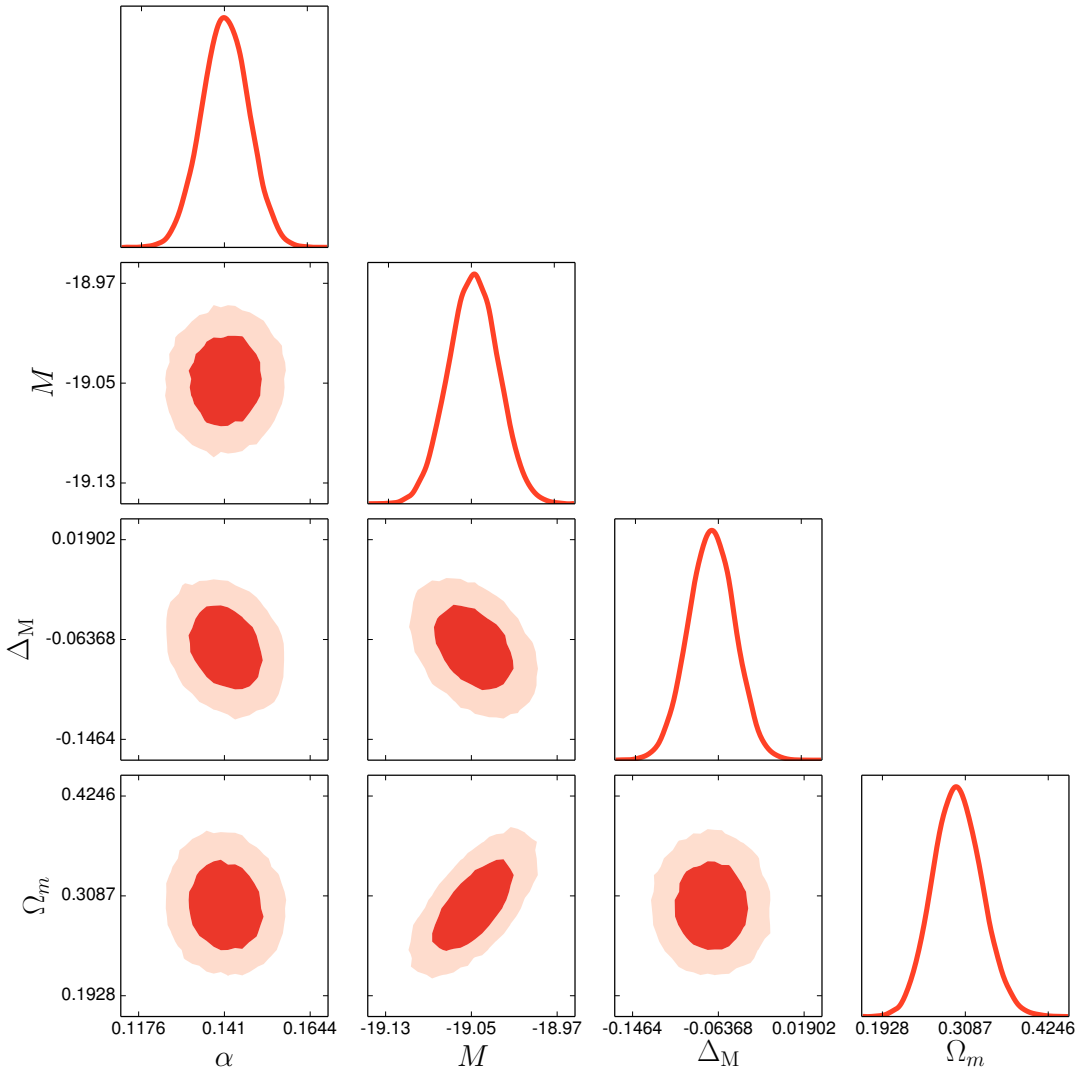
Plotting Refinement

```
import matplotlib.pyplot as plt

info.to_change = {'Delta_M': r'$\Delta_{\rm M}$', }

info.to_plot     = [r'$\Delta_{\rm M}$', 'alpha',
                    'M', 'Omega_m']

info.mean_likelihood = False

info.fontsize = 24
info.ticksize = 12
info.ticknumber = 3
info.decimal = 4

info.bins = 30

info.cm = [(0.99843, 0.25392, 0.14765, 1.),]
info.cmaps = [plt.cm.Reds, ]
```

Better plot

```
python montepython/MontePython info chains/jla/*20000* --extra jla.plot
```

Better plot


-18.97
-19.05
-19.13
M
0.01902
-0.06368
-0.1464
$\Delta_{\rm M}$
0.4246
0.3087
0.1928
Ω_m
0.1176
0.141
0.1644
α
-19.13
-19.05
-18.97
M
-0.1464
-0.06368
0.01902
$\Delta_{\rm M}$
0.1928
0.3087
0.4246
Ω_m