Cosmology Population Monte Carlo

CosmoPMC v1.3 User's manual

Martin Kilbinger
Karim Benabed
Olivier Cappé
Jean Coupon
Jean-François Cardoso
Gersende Fort
Henry J. McCracken
Simon Prunet
Christian P. Robert
Darren Wraith

Contents

1 What is CosmoPMC?

name

CosmoPMC (Cosmology Population Monte Carlo) is a Bayesian sampling method to explore the likelihood of various cosmological probes. The sampling engine is implemented with the package pmclib. It is called Population Monte Carlo (PMC), which is a novel technique to sample from the posterior (**?**). PMC is an adaptive importance sampling method which iteratively improves the proposal to approximate the posterior. This code has been introduced, tested and applied to various cosmology data sets in **?**. Results on the Bayesian evidence using PMC are discussed in **?**.

1.1 Importance sampling

One of the main goals in Bayesian inference is to obtain integrals of the form

$$\pi(f) = \int f(x)\pi(x)\mathrm{d}x \tag{1}$$

over the posterior distribution π which depends on the p-dimensional parameter x, where f is an arbitrary function with finite expectation under π. Of interest are for example the parameter mean ($f = \mathrm{id}$) or confidence regions S with $f = \mathbf{1}_S$ being the indicator function of S. The Bayesian evidence E, used in model comparison techniques, is obtained by setting $f = 1$, but instead of π using the unnormalised posterior $\pi' = L \cdot P$ in (**??**), with L being the likelihood and P the prior.

The evaluation of (**??**) is challenging because the posterior is in general not available analytically, and the parameter space can be high-dimensional. Monte-Carlo methods to approximate the above integrals consist in providing a sample $\{x_n\}_{n=1\ldots N}$ under π, and approximating (**??**) by the estimator

$$\hat{\pi}(f) = \frac{1}{N}\sum_{n=1}^{N} f(x_n). \tag{2}$$

Markov Chain Monte Carlo (MCMC) produces a Markov chain of points for which π is the limiting distribution. The popular and widely-used package `cosmomc` (`http://cosmologist.info/cosmomc`; **?**) implements MCMC exploration of the cosmological parameter space.

Importance sampling on the other hand uses the identity

$$\pi(f) = \int f(x)\pi(x)\mathrm{d}x = \int f(x)\frac{\pi(x)}{q(x)}q(x)\,\mathrm{d}x \tag{3}$$

where q is any probability density function with support including the support of π. A sample

$\{x_n\}$ under q is then used to obtain the estimator

$$\hat{\pi}(f) = \frac{1}{N}\sum_{n=1}^{N} f(x_n)\, w_n; \qquad w_n = \frac{\pi(x_n)}{q(x_n)}. \tag{4}$$

The function q is called the *proposal* or *importance function*, the quantities w_n are the *importance weights*. Population Monte Carlo (PMC) produces a sequence q^t of importance functions ($t = 1 \ldots T$) to approximate the posterior π. Details of this algorithm are discussed in **?**.

The package CosmoPMC provides a C-code for sampling and exploring the cosmological parameter space using Population Monte Carlo. The code uses MPI to parallelize the calculation of the likelihood function. There is very little overhead and on a massive cluster the reduction in wall-clock time can be enormous. Included in the package are post-processing, plotting and various other analysis scripts and programs. It also provides a Markov Chain Monte-Carlo sampler.

1.2 This manual

This manual describes the code CosmoPMC, and can be obtained from `www.cosmopmc.info`. CosmoPMC is the cosmology interface to the Population Monte Carlo (PMC) engine pmclib. Documentation on the PMC library can be found at the same url. The cosmology module of CosmoPMC can be used as stand-alone program, it has the name nicaea (`http://www2.iap.fr/users/kilbinge/nicaea`).

Warning: Use undocumented features of the code at your own risk!

2 Installing CosmoPMC

2.1 Software requirements

CosmoPMC has been developed on GNU/Linux and Darwin/FreeBSD systems and should run on those architectures. Required are:

- C-compiler (e.g. `gcc`, `icc`)
- pmclib (Sect. **??**)
- GSL (`http://www.gnu.org/software/gsl`), version 1.15 or higher
- FFTW (`http://www.fftw.org`)
- Message Parsing Interface (MPI) (`http://www-unix.mcs.anl.gov/mpi`) for parallel calculations

Optional:

- csh, for post-processing, auxiliary scripts; recommended
- perl (http://www.perl.org), for post-processing, auxiliary scripts; recommended
- yorick (http://yorick.sourceforge.net), post-processing, mainly plotting
- python (http://www.python.org), for running the configuration script
- R (http://www.r-project.org), post-processing

To produce 1D and 2D marginal posterior plots with scripts that come with CosmoPMC, either yorick or R are required.

Necessary for CMB anisotropies support:

- Fortran compiler (e.g. ifort)
- Intel Math Kernel libraries (http://software.intel.com/en-us/intel-mkl)
- CAMB (http://camb.info,http://cosmologist.info/cosmomc)
- WMAP data and likelihood code (http://lambda.gsfc.nasa.gov)

2.2 Download and install pmclib

The package pmclib can be downloaded from the CosmoPMC site http://www.cosmopmc.info.

After downloading, unpack the gzipped tar archive

```
> tar xzf pmclib_x.y.tar.gz
```

This creates the pmclib root directory pmclib_x.y. pmclib uses waf[1] instead of configure/make to compile and build the software. Change to that directory and type

```
> ./waf --local configure
```

See ./waf --help for options. The packages lua, hdf5 and lapack are optionally linked with pmclib but are not necessary to run CosmoPMC. Corresponding warnings of missing files can be ignored. Instead of a local installation (indicated by --local), a install prefix can be specified with --prefix=PREFIX (default /usr/local).

2.3 Patch pmclib

For CosmoPMC v ≥ 1.2 and pmclib v1.x, a patch of the latter is necessary. From http://www.cosmopmc.info , download patch_pmclib_1.x_1.2.tar.gz and follow the instructions in the readme file readme_patch_pmclib_1.x_1.2.txt.

[1] http://code.google.com/p/waf

2.4 Download and install CosmoPMC

The newest version of CosmoPMC can be downloaded from the site `http://www.cosmopmc.info`.

First, unpack the gzipped tar archive

```
> tar xzf CosmoPMC_v1.3.tar.gz
```

This creates the the CosmoPMC root directory `CosmoPMC_v1.3`. Change to that directory and run

```
> [python] ./configure.py
```

This (poor man's) configure script copies the file `Makefile.no_host` to `Makefile.host` and sets host-specific variables and flags as given by the command-line arguments. For a complete list, see '`configure.py --help`'.

Alternatively, you can copy by hand the file `Makefile.no_host` to `Makefile.host` and edit it. If the flags in this file are not sufficient to successfully compile the code, you can add more flags by rerunning `configure.py`, or by manually editing `Makefile.main`. Note that a flag in `Makefile.main` is overwritten if the same flag is present in `Makefile.host`.

To compile the code, run

```
> make; make clean
```

On success, symbolic links to the binary executables (in `./exec`) will be set in `./bin`.

It is convenient to define the environment variable `COSMOPMC` and to set it to the main CosmoPMC directory. For example, in the C-shell:

```
> setenv COSMOPMC /path/to/CosmoPMC_v1.3
```

This command can be placed into the startup file (e.g. `~/.cshrc` for the C-shell). One can also add `$COSMOPMC/bin` to the `PATH` environment variable.

3 Running CosmoPMC

3.1 Quick reference guide

Examples

To get familiar with CosmoPMC, use the examples which are contained in the package. Simply change to one of the subdirectories in `$COSMOPMC/Demo/MC_Demo` and proceed on to the point **Run** below.

User-defined runs

To run different likelihood combinations, or your own data, the following two steps are necessary to set up a CosmoPMC run.

1. **Data and parameter files**

   Create new directory with `newdir_pmc.sh`. When asked, enter the likelihood/data type. More than one type can be chosen by adding the corresponding (bit-coded) type id's. Symbolic links to corresponding files in `$COSMOPMC/data` are set, and parameter files from `$COSMOPMC/par_files` are copied to the new directory on request.

   If necessary, copy different or additional data and/or parameter files to the present directory.

2. **Configuration file**

   Create the PMC configuration file `config_pmc`. Examples for existing data modules can be found in `$COSMOPMC/Demo/MC_Demo`, see also Sect. **??** for details.

   In some cases, information about the galaxy redshift distribution(s) have to be provided, and the corresponding files copied (see `$COSMOPMC/Demo` for example files '`nofz*`').

Run

Type

```
> $COSMOPMC/bin/cosmo_pmc.pl -n NCPU
```

to run CosmoPMC on `NCPU` CPUs. See '`cosmo_pmc.pl -h`' for more options. Depending on the type of initial proposal (Sect. **??**), a maximum-search is started followed by a Fisher matrix calculation. After that, PMC is started. Fig. **??** shows a flow chart of the script's actions.

Diagnostics

Check the files `perplexity` and `enc`. If the perplexity reaches values of 0.8 or larger, and if the effective number of components (ENC) is not smaller than 1.5, the posterior has very likely been explored sufficiently. Those and other files are updated during run-time and can be monitored while PMC is running. See Sect. **??** for more details.

Results

The text file `iter_{niter-1}/mean` contains mean and confidence levels. The file `iter_{niter-1}/all_contour2d.pdf` shows the 1d- and 2d-marginals. Plots can be redone

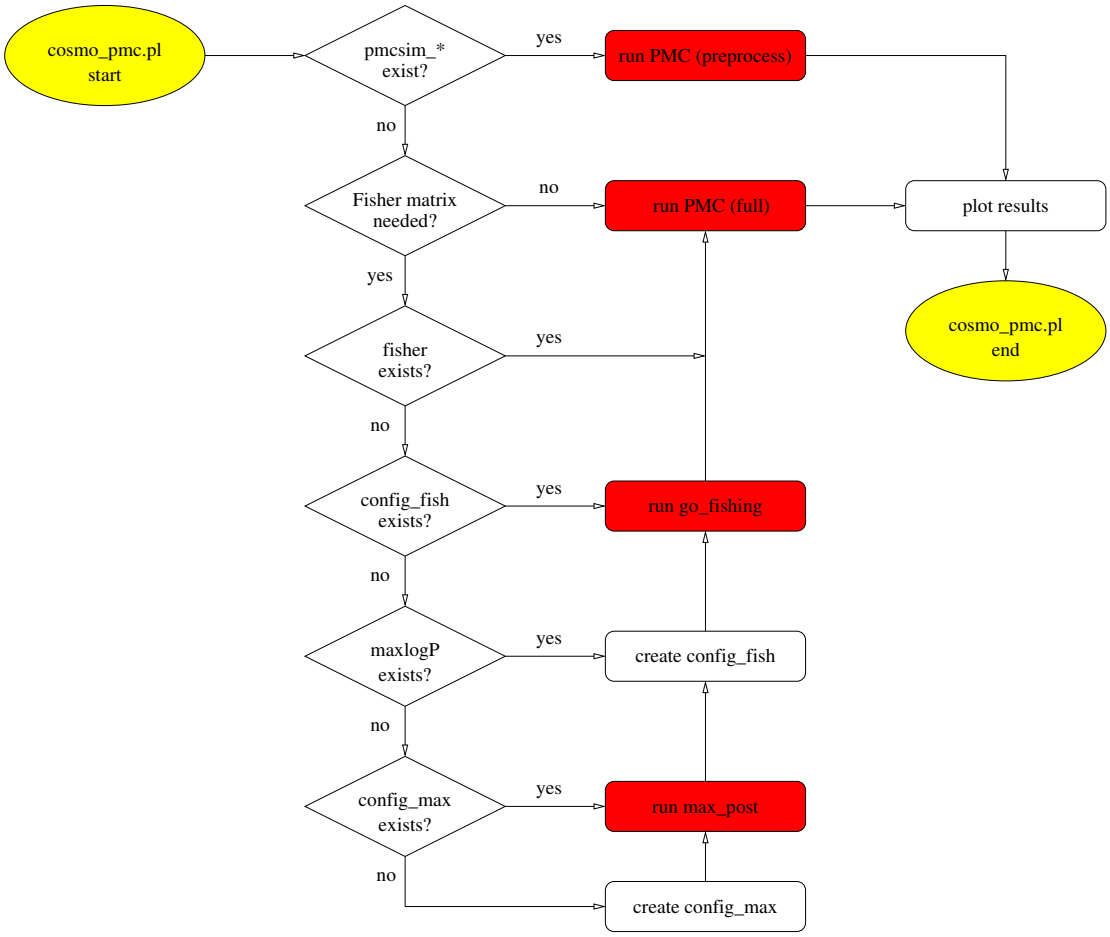

Figure 1: Flow chart for `cosmo_pmc.pl`.

or refined, or created from other than the last iteration with `plot_contour2d.pl`. Note that in the default setting, the posterior plots are not smoothed. See Sect. **??** for more details, and for information on the alternative script `plot_confidence.R`.

3.2 CosmoPMC in detail

This section describes in more detail how PMC is run, and which decisions the user has to make before starting and after stopping a PMC run.

Initial proposal The choice of the initial proposal, used during the first PMC iteration, is of great importance for a successful PMC run. The following options are implemented, determined by the key '`sinitial`' in the configuration file (see Sect. **??**):

1. **`sinitial = fisher_rshift`** The Fisher matrix is used as the covariance of a multi-variate Gaussian/Student-t distribution g. A mixture-model is constructed by creating D copies of g. Each copy is displaced from the ML point by a random uniform shift, and its variance is stretched by random uniform factor.
2. **`sinitial = fisher_eigen`** A mixture-model is constructed in a similar way as the first case, with the difference that the shift from the ML point is now performed along the major axes of the Fisher ellipsoid. Note that if the Fisher matrix is diagonal, the shift of each component only concerns one parameter.
3. **`sinitial = file`** The initial proposal is read from a file (of `mix_mvdens` format), e.g. from a previous PMC run.
4. **`sinitial = random_pos`** Mixture-model components with random variance (up to half the box size) and random positions. This case should only be used if the posterior is suspected to be multi-modal, or the calculation of the Fisher matrix fails.

In many cases, a mixture of multi-variate Gaussians as the proposal is the best choice. For that, set the degrees-of-freedom (ν) parameter `df` to -1. For a posterior with heavy tails, a Student-t distribution might be more suited. The degrees of freedom ν can be chosen freely; $\nu = 3$ is a common choice. For $\nu \to \infty$, a Gaussian distribution is reached asymptotically.

If the Fisher matrix has to be calculated for the initial proposal, the script `cosmo_pmc.pl` calls `max_post` and `go_fishing` to estimate the maximum-likelihood point and the Hessian at that point, respectively. The script `config_pmc_to_max_and_fish.pl` can be used to create the corresponding configuration files from the PMC config file for manual calls of `max_post` and `go_fishing`.

Updating the proposal The PMC algorithm automatically updates the proposal after each iteration, no user interference is necessary.

The method to update the proposal is a variant of the Expectation-Maximization algorithm (EM, **?**). It leads to an increase of the perplexity and an increase of ESS. Detailed descriptions of this algorithm in the case of multi-variate Gaussian and Student-t distributions can be found in **?** and **?**.

Dead components A component can 'die' during the updating if the number of points sampled from that component is less than `MINCOUNT = 20`, or its weight is smaller than the inverse total number of sample points $1/N$. There are two possibilities to proceed. First, the component is 'buried', its weight set to zero so that no points are sampled from it in subsequent iterations.

Alternatively, the component can be revived. In this case, it is placed near the component ϕ_{d_0} which has maximum weight, and it is given the same covariance as ϕ_{d_0}.

The first case is the standard method used in **?**. The second method tries to cure cases where the majority of components die. This can happen if they start too far off from the high-density posterior region. Often, only one component remains to the end, not capable of sampling the posterior reliably.

Both options can be chosen using the config file (Sect. **??**) key `sdead_comp = {bury|revive}`.

Errors If an error occurs during the calculation of the likelihood, the error is intercepted and the likelihood is set to zero. Thus, the parameter vector for which the error occurs is attributed a zero importance weight and does not contribute to the final sample. An error message is printed to `stderr` (unless CosmoPMC is run with the option `-q`) and PMC continues with the next point.

An error can be due to cosmological reasons, e.g. a redshift is probed which is larger than the maximum redshift in a loitering Universe. Further, a parameter could be outside the range of a fitting formulae, e.g. a very small scalar spectral index in the dark matter transfer function.

Usually, the errors printed to `stderr` during PMC sampling can be ignored.

Random numbers The GSL random number generator is used to generate random variables. It is initialised with a seed reading the current time, to produce different (pseudo-) random numbers at each call. The seed is written to the log file. Using the option '`-s SEED`', a user-specified seed can be defined. This is helpful if a run is to be repeated with identical results.

3.3 Output files

Each iteration i produces a number of output files which are stored in subdirectories `iter_i` of the CosmoPMC starting directory. Files which are not specific to a single iteration are placed in the starting directory.

3.3.1 Diagnostics

Unlike in MCMC, with adaptive importance sampling one does not have to worry about convergence. In principle, the updating process can be stopped at any time. There are however diagnostics to indicate the quality and effectiveness of the sampling.

Perplexity and effective sample size `perplexity`

The perplexity p is defined in eq. (18) of **?**. The range of p is $[0; 1]$, and will approach unity if the proposal and posterior distribution are close together, as measured by the Kullback-Leibler

divergence. The initial perplexity is typically very low (< 0.1) and should increase from iteration to iteration. Final values of 0.99 and larger are not uncommon, but also for p of about 0.6-0.8 very accurate results can be obtained. If p is smaller than say 0.1, the PMC sample is most likely not representative of the posterior. Intermediate values for p are not straight-forward to interpret.

Closely related to the perplexity is the effective sample size ESS, which lies in the range $[1; N]$. It is interpreted as the number of sample point with zero weight (**?**). A large perplexity is usually accompanied by a high ESS. For a successful PMC run, ESS is much higher than the acceptance rate of a Monte Carlo Markov chain, which is typically between 0.15 and 0.25.

The file `perplexity` contains the iteration i, perplexity p, ESS for that iteration, and the total ESS. This file is updated after each iteration and can therefore be used to monitor a PMC run.

If there are points with very large weights, they can dominate the other points whose normalised weights will be small. Even a few sample points might dominate the sum over weights and result in a low perplexity. The perplexity is the most sensitive quantity to those high-weight points, much more than e.g. the mean, the confidence intervals or the evidence.

Effective number of proposal components `enc`

The proposal q^t provides useful information about the performance of a PMC run. For example, the effective number of components, defined in complete analogy to ESS,

$$\mathrm{ENC} = \left(\sum_{d=1}^{D} \left\{ \alpha_d^t \right\}^2 \right)^{-1}, \tag{5}$$

is an indication of components with non-zero weight. If ENC is close to unity, the number of remaining components to sample the posterior is likely to be too small to provide a representative sample. For a badly chosen initial proposal, this usually happens already at the first few iterations. By monitoring the file `enc` which is updated each iteration, an unsuccessful PMC run can be aborted.

The effective number of components can also be determined from any proposal file (`mix_mvdens` format) with the script `neff_proposal.pl`.

An additional diagnostic is the evolution of the proposal components with iteration. This illustrates whether the components spread out nicely across the high-posterior region and reach a more or less stationary behaviour, or whether they stay too concentrated at one point. The scripts `proposal_mean.pl` (`proposal_var.pl`) read in the proposal information q^t and plot the means (variances) as function of iteration t.

3.3.2 Results

PMC samples `iter_i/pmcsim`

This file contains the sample points. The first column is the (unnormalised) importance weight (log), the second column denotes the component number from which the corresponding point was sampled. Note that the n_{clip} points with highest weights are not considered in subsequent calculations (of moments, perplexity, evidence etc.). The next p columns are the p-dimensional parameter vector. Optionally, n_{ded} numbers of deduced parameters follow.

Proposals `iter_i/proposal`

The proposal used for the importance sampling in iteration i is in `mix_mvdens` format (Sect. **??**). The final proposal, updated from the sample of the last iteration, is `proposal_fin`.

Mean and confidence intervals `iter_i/mean`

This file contains mean and one-dimensional, left- and right-sided confidence levels (c.l.). A c.l. of $p\%$ is calculated by integrating the one dimensional normalised marginal posterior starting from the mean in positive or negative direction, until a density of $p\%/2$ is reached. PMC outputs c.l.'s for $p = 68.27\%, 95.45\%$ and 99.73%. With the program `cl_one_sided`, one-sided c.l.'s can be obtained.

For post-processing, the program `meanvar_sample` outputs the same information (mean or median, and c.l.) from an existing PMC sample, including possible deduced parameters.

Resampled PMC simulations `iter_{niter-1}/sample`

If `cosmo_pmc.pl` has been run with the option '`-p R`', the directory of the final iteration contains the file of parameter vectors `sample`, which is resampled from the PMC simulation `pmcsim`, taking into account the importance weights. The resampled points all have unit weight. Resampling is a post-processing step, it is performed by calling the `R` script `sample_from_pmcsimu.R` from `cosmo_pmc.pl`; this can also be done manually with any `pmscim` simulation.

Histograms `iter_i/chi_j, iter_i/chi_j_k`

One- and two-dimensional histograms are written at each iteration to the text files `chi_j` and `chi_j_k`, respectively, where j and k, $j < k$, are parameter indices. Those histograms can be used to create 1d- and 2d-marginals, using the script `plot_contour2d.pl`. The bin number is set by the config entry `nbinhist`.

In post-processing, use `histograms_sample` to produce histograms from a PMC sample. This can be useful if deduced parameters have been added to the sample.

Covariance `iter_i/covar*.fin`

The parameter covariance and inverse covariance are printed to the files `covar.fin` and, respectively, `covarinv.fin`. The addition "`+ded`" in the file name indicates the inclusion of deduced parameters. The covariance matrices are in "mvdens"-format (see Sect. **??**).

Evidence `evidence`

This file contains the Bayesian evidence as a function of iteration. Before the first iteration, the Laplace approximation using the Fisher matrix is printed to `evidence_fisher` if the file `fisher` exists. At each iteration i, `iter_i/evidence_covarinv` contains the Laplace approximation of the evidence from the inverse covariance matrix of the sample `iter_i/pmcsim`.

3.3.3 Deduced parameters

Deduced parameters can be part of a PMC simulation. These parameters are not sampling parameters, but they are deduced from the main parameters. For example, if Ω_m and Ω_Λ are sampling parameters of a non-flat model, the curvature $\Omega_K = \Omega_\mathrm{m} + \Omega_\Lambda$ can be a deduced parameter.

In most cases, deduced parameters are ignored while running CosmoPMC. They are usually added to the PMC simulation after the sampling, for example using the script `add_ded.pl`. In the case of galaxy clustering, `add_deduced_halomodel` adds deduced parameters which depend on the sampling parameters but also on the underlying cosmology and halo model. For weak lensing, a COSEBIs orthogonal data vector (**?**) can be added with `add_deduced_cosebis`.

A PMC simulation with deduced parameters added can be used as input to `histograms_sample`, to create the histogram files, now including the deduced parameters. These can then in turn be read by and `plot_contour2d.pl` to produce 1d- and 2d-marginals, including the deduced parameters. Alternatively, the PMC simulation with added parameters can be resampled using `sample_from_pmcsimu.R`, from which plots can be created by `plot_confidence.R`.

3.3.4 Other files

Maximum-posterior parameter `max_logP`

`max_post` stores its estimate of the maximum posterior in this file.

Fisher matrix `fisher`

The final result of `go_fishing`, the Fisher matrix in mvdens (Sect. **??**) format.

Log files `log_max_post, log_fish, log_pmc`

`max_post`, `go_fishing` and `cosmo_pmc` each produce their corresponding log file.

4 Cosmology

The cosmology part of CosmoPMC is essentially the same as the stand-alone package NICAEA[2]. This excludes the external program camb and the WMAP likelihood library, which are called by CosmoPMC for CMB anisotropies. Further, CosmoPMC contains a wrapper layer to communicate between the PMC sampling and the cosmology modules.

4.1 Basic calculations

A number of routines to calculate cosmological quantities are included in the code. These are

- Background cosmology: Hubble parameter, distances, geometry
- Linear perturbations: growth factor, transfer function, cluster mass function, linear 3D power spectra
- Non-linear evolution: fitting formulae for non-linear power spectra (**??**), emulators (**???**), halo model
- Galaxy clustering: HOD model
- Cosmic shear: convergence power spectrum, second-order correlation functions and derived second-order quantities, third-order aperture mass moment
- CMB anisotropies via camb.

4.1.1 Density parameters

Both the density parameters ($\Omega_\mathrm{X} = \rho_\mathrm{X}/\rho_\mathrm{c}$) and the physical density parameters ($\omega_\mathrm{x} = \Omega_\mathrm{x} h^2$) are valid input parameters for sampling with PMC. Internally, the code uses non-physical density parameters (Ω_X). All following rules hold equivalently for both classes of parameters. Note that physical and non-physical density parameters can not be mixed, e.g. Ω_c and ω_K on input causes the program to abort.

The parameter for massive neutrinos, $\Omega_{\nu,\mathrm{mass}}$, is not contained in the matter density $\Omega_\mathrm{m} = \Omega_\mathrm{c} + \Omega_\mathrm{b}$.

A parameter which is missing from the input list is assigned the default value, found in the corresponding cosmology parameter file (cosmo.par), unless there is an inconsistency with other input parameters. E.g., if Ω_de and Ω_K are input parameters, Ω_m is assigned the value $\Omega_\mathrm{m} = 1 - \Omega_\mathrm{de} - \Omega_K - \Omega_{\nu,\mathrm{mass}}$, to keep the curvature consistent with Ω_K.

A flat Universe is assumed, unless (a) both Ω_m and Ω_de, or (b) Ω_K are given as input parameter.

[2] http://www2.iap.fr/users/kilbinge/nicaea

Table 1: Extrapolation of the power spectra

snonlinear	$k_{\max}$	n_{ext}
linear	$333.6\,h$ Mpc^{-1}	$n_{\text{s}} - 4$
pd96	$333.6\,h$ Mpc^{-1}	-2.5
smith03, smith03_de, smith03_revised	$333.6\,h$ Mpc^{-1}	Eq. (61), **?**
coyote10	2.416 Mpc^{-1}	no extrapolation
coyote13	8.569 Mpc^{-1}	no extrapolation

4.1.2 Matter power spectrum

Usually, models of the non-linear power spectrum have a limited validity range in k and/or redshift. For small k, each model falls back to the linear power spectrum, which goes as $P_\delta(k) \propto k^{n_{\text{s}}}$. For large k, the extrapolation as a power law $P_\delta(k) \propto n_{\text{ext}}$ is indicated in Table **??**.

See for more details on the models.

The Coyote emulator In the coyote10 and coyote13 cases, the power spectrum is zero for $k > k_{\max}$. The same is true for redshifts larger than the maximum of $z_{\max} = 1\,(4)$ for coyote10 (coyote13). See **?** for an alternative approach.

For coyote10, the Hubble constant h can not be treated as a free parameter. For a given cosmology, it has to be fixed to match the CMB first-peak constraint $\ell_{\text{A}} = \pi d_{\text{ls}}/r_{\text{s}} = 302.4$, where d_{ls} is the distance to last scattering, and r_{s} is the sound horizon. This can be done with the function set_H0_Coyote, see Demo/lensingdemo.c for an example. When doing sampling with non-physical density parameters, h has to be set at each sample point. Alternatively, the physical density parameters can be sampled, where h is set internally to match the CMB peak.

The updated version coyote13 also emulates variations of the Hubble constant, and the above described restrictions do not apply.

4.1.3 Likelihood

Each cosmological probe has its own log-likelihood function. The log-likelihood function is called from a wrapping routine, which is the interface to the PMC sampler. In general, within this function the model vector is computed using the corresponding cosmology routine. The exception are the WMAP-modules where the C_ℓ's are calculated using camb and handed over to the log-likelihood function as input.

4.2 Cosmic shear

CosmoPMC implements second- and third-order weak lensing observables.

4.2.1 Second-order

The basic second-order quantities in real space for weak gravitational lensing are the two-point correlation functions $\xi_\pm$ (2PCF) (e.g **?**),

$$\xi_\pm(\theta) = \frac{1}{2\pi}\int_0^\infty \mathrm{d}\ell\, \ell P_\kappa(\ell) \mathrm{J}_{0,4}(\ell\theta). \tag{6}$$

Data corresponding to both functions (`slensdata=xipm`) as well as only one of them (`xip`, `xim`) can be used. The E-mode correlation function ξ_+^E (**?**) is possible on input (`slensdata=xiE`). Its model is equal to ξ_+.

The aperture-mass dispersion (**?**)

$$\langle M_\mathrm{ap}^2\rangle(\theta) = \frac{1}{2\pi}\int_0^\infty \mathrm{d}\ell\, \ell P_\kappa(\ell)\hat{U}^2(\theta\ell). \tag{7}$$

The function $\hat{U}(\theta\ell)$ is the Fourier-transform of a filter function $U_\theta(\vartheta) = u(\vartheta/\theta)/\theta^2$, of which two forms are implemented (**??**),

$$\text{polynomial (}\texttt{map2poly}\text{):}\quad u(x) = \frac{9}{\pi}(1-x^2)\left(\frac{1}{3}-x^2\right)\mathrm{H}(1-x); \tag{8}$$

$$\text{Gaussian (}\texttt{map2gauss}\text{):}\quad u(x) = \frac{1}{2\pi}\left(1-\frac{x^2}{2}\right)\mathrm{e}^{-\frac{x^2}{2}}. \tag{9}$$

The top-hat shear dispersion (**?**)

$$\langle|\gamma|^2\rangle_\mathrm{E,B}(\theta) = \frac{1}{2\pi}\int_0^\infty \mathrm{d}\ell\, \ell\, P_\kappa(\ell)\, \frac{4\mathrm{J}_1(\ell\theta)}{(\ell\theta)^2} \tag{10}$$

is used with `slensdata = gsqr`.

Pure E-/B-mode separating functions (**?**) are chosen with `slensdata = decomp_eb`. For the lack of analytical expressions for filter functions to obtain these real-space statistics from the convergence power spectrum, they are calculated by integrating over the 2PCF. The integral is performed over the finite angular interval $[\vartheta_\mathrm{min};\vartheta_\mathrm{max}]$. The prediction for the E-mode is

$$E = \frac{1}{2}\int_{\vartheta_\mathrm{min}}^{\vartheta_\mathrm{max}} \mathrm{d}\vartheta\, \vartheta\, \left[T_+(\vartheta)\,\xi_+(\vartheta) \pm T_-(\vartheta)\,\xi_-(\vartheta)\right]. \tag{11}$$

Two variants of filter functions are implemented: The 'optimized' E-/B-mode function **?** for which the real-space filter functions are Chebyshev polynomials of the second kind,

$$T_+(\vartheta) = \tilde{T}_+\left(x = \frac{2\vartheta - \vartheta_{\rm max} - \vartheta_{\rm min}}{\vartheta_{\rm max} - \vartheta_{\rm min}}\right) = \sum_{n=0}^{N-1} a_n U_n(x); \quad U_n(x) = \frac{\sin[(n+1)\arccos x]}{\sin(\arccos x)}. \tag{12}$$

The coefficients a_n have been optimized with respect to signal-to-noise and the $\Omega_{\rm m}$-σ_8 Fisher matrix. The function E is defined as a function of the lower angular limit $\vartheta_{\rm min}$. The ratio η of lower to upper limit, $\eta = \vartheta_{\rm min}/\vartheta_{\rm max}$ is fixed.

The second variant are the so-called COSEBIs (Complete Orthogonal Sets of E-/B-mode Integrals; **?**). We implement their 'logarithmic' filter functions,

$$T^{\rm log}_{+,n}(\vartheta) = t^{\rm log}_{+,n}\left[z = \ln\left(\frac{\vartheta}{\vartheta_{\rm min}}\right)\right] = N_n \sum_{j=0}^{n+1} c_{nj} z^j = N_n \prod_{j=1}^{n+1}(z - r_{nj}). \tag{13}$$

The coefficients c_{nj} are fixed by integral conditions that assure the E-/B-mode decomposition of the 2PCF on a finite angular integral. They are given by a linear system of equations, which is given in **?**. To solve this system, a very high numerical accuracy is needed. The MATHEMATICA notebook file `$COSMOPMC/par_files/COSEBIs/cosebi.nb`, adapted from **?**, can be run to obtain the coefficients for a given $\vartheta_{\rm min}$ and $\vartheta_{\rm max}$. An output text file is created with the zeros r_{ni} and amplitudes N_n. The file name is `cosebi_tplog_rN_[Nmax]_[thmin]_[thmax]`, where `Nmax` is the number of COSEBI modes, `thmin` and `thmax` are the minimum and maximum angular scale $\vartheta_{\rm min}$ and $\vartheta_{\rm max}$, respectively. For a given $\vartheta_{\rm min}$ and $\vartheta_{\rm max}$, specified with the config entries `th_min` and `th_max`, CosmoPMC reads the corresponding text file from a directory that is specified by `path`. A sample of files with various scales are provided in `$COSMOPMC/par_files/COSEBIs`.

The COSEBIs are discrete numbers, they are specified by an integer mode number n.

In both cases of pure E-/B-mode separating statistics, the function T_- is calculated from T_+ according to **?**.

The additional flag `decomp_eb_filter` decides between different filter functions:

`decomp_eb_filter`	Reference	Filter function typ	η
`FK10_SN`	**?**	optimized Signal-to-noise	1/50
`FK10_FoM_eta10`	**?**	optimized Fisher matrix	1/10
`FK10_FoM_eta50`	**?**	optimized Fisher matrix	1/50
`COSEBIs_log`	**?**	logarithmic	

Further, the convergence power spectrum P_κ with covariance matrix can be used with the flag `slensdata = pkappa`.

Intrinsic alignment contributions can be added with `sia = HS04`. This model used the linear model from **?**, but with the non-linear power spectrum as input (**?**), see **?** for a recent application of this model to data.

4.2.2 Third-order

We implement the aperture-mass moment (**???**) with the Gaussian filter (eq. **??**). In terms of the bispectrum B_κ, the third-order aperture-mass moments are given as,

$$\left\langle M_{\rm ap}^3 \right\rangle(\theta_1,\theta_2,\theta_3) \equiv \left\langle M_{\rm ap}(\theta_1) M_{\rm ap}(\theta_2) M_{\rm ap}(\theta_3) \right\rangle$$
$$= \int \frac{{\rm d}^2\ell_1}{(2\pi)^2} \int \frac{{\rm d}^2\ell_2}{(2\pi)^2}\, B_\kappa(\boldsymbol{\ell}_1,\boldsymbol{\ell}_2) \sum_{(i,j,k)\in S_3} \hat{U}(\theta_i|\boldsymbol{\ell}_1|)\, \hat{U}(\theta_j|\boldsymbol{\ell}_2|)\, \hat{U}(\theta_k|\boldsymbol{\ell}_1+\boldsymbol{\ell}_2|), \quad (14)$$

where S_3 is the symmetric permutation group of (123). One of the four integrals in (**??**) is performed analytically using the angular dependence of the bispectrum due to the statistical isotropy of the convergence field. The result is given in **?**.

There are two cases that can be chosen:

- `slensdata = map3gauss`

  The 'generalised' third moment $\langle M_{\rm ap}^3\rangle(\theta_1,\theta_2,\theta_3)$ with three filter scales.

- `slensdata = map3gauss_diag`

  The 'diagonal' third moment $\langle M_{\rm ap}^3\rangle(\theta) = \langle M_{\rm ap}^3\rangle(\theta,\theta,\theta)$ using a single aperture filter scale.

The former contains more information, since it probes the bispectrum on a wider range of triangles in Fourier space. The advantage of the latter choice is that the computing time is shorter. For N angular scales N entries of the third aperture-mass moment vector have to be calculated. For the former, this vector contains $N(N+1)(N+2)/6$ entries.

Models of intrinsic galaxy alignment and source-lens clustering can be added for the diagonal aperture-mass third moment. Intrinsic alignments contain three terms, *GGI*, *GII*, and *III*. *III* does not play a large role for moderately wide redshift bins, and is not included here. *GGI* and *GII* are modeled with an exponential function in angular separation, following **?**, with `sia = S08`. Source-lens clustering is calculated using perturbation theory, and a linear galaxy bias model (`sslc = slc_FK13`; **?**).

4.2.3 Second- plus third-order

A joint data vector of second- and third-order observables can be used in CosmoPMC. The covariance is interpreted as a joint block matrix, with the second-order and third-order auto-covariances on the diagonal, and the cross-correlation on the off-diagonal blocks. The possible scenarios are:

- `slensdata = map2gauss_map3gauss`

  Gaussian aperture-mass dispersion and generalised third moment.

- `slensdata = map2gauss_map3gauss_diag`

  Gaussian aperture-mass dispersion and diagonal third moment.

- `slensdata = decomp_eb_map3gauss`

  Log-COSEBIs and generalised aperture-mass third moment. The flag `decomp_eb_filter` has to be set to `COSEBIs_log`.

- `slensdata = decomp_eb_map3gauss_diag`

  Log-COSEBIs and diagonal aperture-mass third moment. The flag `decomp_eb_filter` has to be set to `COSEBIs_log`.

The first two cases use the same filter for second- and third-order, and provide therefore a consistent measure for both orders. The last two cases use the optimal E-/B-mode function known for second order.

4.2.4 Covariance

The covariance matrix is read from a file, and the inverse is calculated in CosmoPMC. The matrix has to be positive definite. An Anderson-Hartlap debiasing factor is multiplied to the inverse (**??**), which is specified with the config entry `corr_invcov`.This can also be used to rescale the covariance, e.g. to take into account a different survey area. Set this value to unity if no correction is desired.

The covariance is either taken to be constant and not dependent on cosmology. In that case, set `scov_scaling` to `cov_const`. Or the approximated schemes from **?** are adopted, see **?** for the implementation. In that scheme (`scov_scaling = cov_ESH09`, the shot-noise term D is constant, the mixed term M is modulated with Ω_m and σ_8 using fitting formluae, and the cosmic-variance term V is proportional to the square of the shear correlation function. This scheme is available for `slensdata = xipm`. The three covariance terms have to be read individually. The entry `covname`, which for `scov_scaling = cov_const` corresponds to the total covariance matrix, now specified the file name of cosmic-variance term, `covname_M` the name of the mixed term, and `covname_D` the name of the shot-noise term. The varying covariance might be not positive definite for some parameter combinations. In that case, the Cholesky decomposition fails and an error is created, and the corresponding likelihood valus is returned as zero.

In the cases `slensdata = xipm`, and for a combined second- and third-order data vector (Sect. **??**), the covariance is the combined covariance of the two data vectors, including cross-correlation terms. See Sect. **??** for details on the file format.

Table 2: Parameter limits where the reduced-shear corrections are valid (from **?**).

α	Parameter	lower	upper
1	Ω_m	0.22	0.35
2	Ω_de	0.33	1.03
3	w	-1.6	-0.6
4	Ω_b	0.005	0.085
5	h	0.61	1.11
6	σ_8	0.65	0.93
7	n_s	0.86	1.16

4.2.5 Reduced shear

The fact that not the shear γ but the reduced shear $g = \gamma/(1-\kappa)$ is observable leads to corrections to the shear power spectrum of a few percent, mainly on small scales. These corrections are either ignored, or modelled to first order according to **?**. This is controlled in the lensing parameter file (`cosmo_lens.par`). The parameter range where the reduced-shear corrections are valid are indicated in Table **??**.

4.2.6 Angular scales

The flag `sformat` describes the mapping of angular scales (given in the data file) and 'effective' scales, where the model predictions of the shear functions are evaluated:

1. `sformat = angle_center`: The effective scale is the same as given in the data file, $\theta_\mathrm{eff} = \theta$.
2. `sformat = angle_mean`: The model is averaged over a range of scales $[\theta_0, \theta_1]$ given in the data file.
3. `sformat = angle_wlinear`: The model is the weighted average over a range of scales $[\theta_0, \theta_1]$, where the weight is $w = \theta/\mathrm{arcmin}$.
4. `sformat = angle_wquadr`: The model is the weighted average over a range of scales $[\theta_0, \theta_1]$, where the weight is $w = a_1(\theta/\mathrm{arcmin}) + a_2(\theta/\mathrm{arcmin})^2$.

The first mode (`angle_center`) should be used for aperture-mass, shear rms and 'ring' statistics, since those quantities are not binned, but instead are integrals up to some angular scale θ. For the correlation functions, in particular for wide angular bins, one of the last three modes is preferred. The quadratic weighting (`angle_wquadr`) corresponds to a weighting of the correlation function

by the number of pairs[3]. This mode was used in the COSMOS analysis (**?**).

4.3 SNIa

The standard distance modulus (`schi2mode = chi2_simple`) for a supernova with index i is

$$\mu_{B,i} = m^*_{B,i} - \bar{M} + \alpha(s_i - 1) - \beta c_i. \tag{15}$$

where the quantities measured from the light-curve fit are the rest-frame B-band magnitude $m^*_{B,i}$, the shape or stretch parameter s_i, and the color c_i. The universal absolute SNIa magnitude is $\bar{M}$, the linear response parameters to stretch and color are α and β, respectively. The χ^2-function is

$$\chi^2_{\rm sn}(\boldsymbol{p}) = \sum_i \frac{\left[\mu_{B,i}(\boldsymbol{p}) - 5\log_{10}\left(\frac{d_{\rm L}(z_i,\boldsymbol{p})}{10\,{\rm pc}}\right)\right]^2}{\sigma^2(\mu_{B,i}) + \sigma^2_{{\rm pv},i} + \sigma^2_{\rm int}}, \tag{16}$$

where $d_{\rm L}$ is the luminosity distance and z_i the redshift of object i. The contributions to the total error for object i are: (1) The light-curve parameter variance $\sigma^2(\mu_{B,i}) = \boldsymbol{\theta}_2^{\rm t} W_2 \boldsymbol{\theta}_2$ with the parameter vector $\boldsymbol{\theta}_2 = (1, \alpha, \beta)$ and the covariance W_2 of the data vector $(m^*_{B,i}, s_i, c_i)$. (2) The peculiar velocity uncertainty $\sigma_{{\rm pv},i} = 5/\ln 10 \cdot v_{\rm p}/(c\, z_i)$. (3) The intrinsic absolute magnitude scatter $\sigma_{\rm int}$.

The Hubble parameter is absorbed into the absolute magnitude which we define as $M = \bar{M} - 5\log_{10} h_{70}$.

The form of this log-likelihood function has been used in **?**.

The following variations of the distance modulus and log-likelihood are implemented:

- `schi2mode = chi2_Theta1`: The χ^2 is extended to include photometric zero-point uncertainties, see **?**.
- `schi2mode = chi2_Theta2_denom_fixed`: The parameters α and β in the denominator of (**??**) are fixed and kept constant during the Monte-Carlo sampling.
- `schi2mode = chi2_no_sc`: The stretch and color parameters are ignored, the distance modulus is $\mu_{B,i} = m^*_{B,i} - \bar{M}$.
- `schi2mode = chi2_betaz`: Instead of a single parameter, the color response is redshift-dependent, $\beta \to \beta + \beta_z z_i$.
- `chi2_dust`: Intergalactic dust absorption is taken into account in the distance modulus, see **?**.

[3] P. Simon, private communication

The covariance matrix W_2 of the data vector $(m^*_{B,i}, s_i, c_i)$ depends on the parameters α and β. In a Bayesian framework, this leads to an additional term $\frac{1}{2} \log \det W_2$ in the log-likelihood function. Taking into account this parameter-dependent term leads however to a biased maximum-likelihood estimator, in particular for α and β[4]. Therefore, it is recommended to not include this term. Use the flag `add_logdetCov = 0/1` in the configuration file to disable/enable this term.

4.4 CMB anisotropies

The full CMB anisotropies are handled externally: The C_ℓ's are calculated by calling `camb`[5] (**?**), the WMAP likelihood function (3rd-, 5th- and 7th-year) is computed using the WMAP public code[6] (**?**). The maximum ℓ up to which the C_ℓ's are calculated and used in the likelihood can be determined in the configuration file. An $\ell_{\rm max} = 2000$ is recommended for high precision calculations.

The power spectrum from the Sunyaev-Zel'dovich (SZ) effect can be added to the C_ℓ's, multiplied with an amplitude A as free parameter. The predicted SZ power spectrum is taken from **?**. This model has been used in the 3-, 5- and 7-year analyses of the WMAP data (**?**).

Alternatively, the WMAP distance priors (**?**) can be employed.

4.5 Galaxy clustering

4.5.1 Halomodel and HOD

The theoretical model of galaxy clustering is the one used in **?**; see this paper for details of the model and further references.

As the basis to describe galaxy clustering, we implement the halo-model as reviewed in (**?**), which accounts for the clustering of dark-matter halos. On top of that, a halo occupation distribution (HOD) function (**???**) is the prescription of how galaxies populate those halos. This function is the number of galaxies N in a halo of mass M. With the flag `hod = berwein02_excl`, this number is expressed as the sum of central ($N_{\rm c}$) plus satellite ($N_{\rm s}$) galaxies,

$$N(M) = N_{\rm c}(M) \times [1 + N_{\rm s}(M)] \ , \tag{17}$$

[4] J. Guy, private communication
[5] `http://camb.info`
[6] `http://lambda.gsfc.nasa.gov`

with

$$n_{\rm c}(M) = \frac{1}{2}\eta\left[1 + {\rm erf}\left(\frac{\log_{10} M - \log_{10} M_{\rm min}}{\sigma_{\log M}}\right)\right]; \tag{18}$$

$$n_{\rm s}(M) = \begin{cases} \left(\frac{M-M_0}{M_1}\right)^{\alpha}; & \text{if } M > M_0 \\ 0 & \text{else} \end{cases} . \tag{19}$$

The parameter $\eta \in [0; 1]$ allows to specify the fraction of central galaxies for a given sample. The extremes values are $\eta = 0/1$ (no/all halos have a central galaxy of a given type). We further compute the galaxy two-point correlation function $\xi(r)$ and its angular projection $w(\theta)$ using the redshift distribution provided by the user, as well as the galaxy number density (for a full description of the model see **?**). To prevent haloes from overlapping, we implement the halo exclusion formalism as described in **?**.

For the halo bias, three options are available:

- `shalo_bias = bias_sc`

  Bias expansion from the spherical collapse model, see e.g. eq. (68) from **?**.

- `shalo_bias = bias_tinker05`

  Bias calibrated with numerical simulations, **?** eq. (A1).

- `shalo_bias = bias_tinker10`

  Updated bias fitting formua from **?**, eq. (6) and Table 2.

The mass function describes the number of halos for a given mass and redshift. It is defined as

$$\frac{{\rm d}n}{{\rm d}\ln M} = \frac{\overline{\rho}_0}{M}\frac{\nu f(\nu)}{\nu}\frac{{\rm d}\nu}{{\rm d}\ln M}, \tag{20}$$

where $\nu(M, z) = \delta_{\rm c}(z)/[D_+(z)\sigma(M)]$ is a measure of the overdensity with $\sigma(M)$ being the rms matter fluctuation in a top-hat window containing the mass M. $\overline{\rho}_0 = \Omega_{\rm m}\rho_{\rm c,0}$ is the mean density of matter at the present day.

The following mass functions are implemented, via the flag `smassfct`:

- From the spherical/elliptical collapse model:

$$\nu f(\nu) = A\sqrt{\frac{2}{\pi a\nu^2}}\left[1 + (a\nu^2)^{-p}\right]\exp\left(-\frac{a\nu^2}{2}\right), \tag{21}$$

  - `ps`: $p = 0, q = 1$ (**?**)
  - `st`: $p = 0.3, q = 0.75$ (**?**)
  - `st2`: $p = 0.3, q = 0.707$ (**?**)

- From numerical simulations:

$$\nu f(\nu) = f(\sigma) = 0.315 \exp\left[-|\ln(\sigma^{-1} + 0.61|^{3.8}\right] \tag{22}$$

  - `j01`: (**?**)

The dark-matter halos have the density profile

$$\rho(r) = \rho_s \left[(r/r_s)^{\alpha}(1 + r/r_s)^{3-\alpha}\right]^{-1} . \tag{23}$$

For slopes unequal to the **?** value of $\alpha = 1$, closed expressions for the Fourier transform of ρ do not exist, and the code will be slower.

The concentration parameter is given by

$$c(M, z) = \frac{c_0}{1 + z}\left[\frac{M}{M_\star}\right]^{-\beta} , \tag{24}$$

following **?**. The parameters c_0 and β can be chosen freely in the halomodel parameter file `halomodel.par`.

The log-likelihood function is the sum of the contribution from the angular correlation function and the galaxy number density $n_{\rm gal}$:

$$\chi^2 = \sum_{i,j}\left[w^{\rm obs}(\theta_i) - w^{\rm model}(\theta_i)\right]\left(C^{-1}\right)_{ij}\left[w^{\rm obs}(\theta_j) - w^{\rm model}(\theta_j)\right] + \frac{\left[n_{\rm gal}^{\rm obs} - n_{\rm gal}^{\rm model}\right]^2}{\sigma^2_{n_{\rm gal}}} , \tag{25}$$

where $n_{\rm gal}^{\rm model}$ is estimated at the mean redshift of the sample.

The number of galaxies (second term in eq. **??**) can be included in the following way, with the config flag `sngal_fit_type`:

- `ngal_lin_fit`: linear (standard; according to the above equation)
- `ngal_log_fit`: logarithmical
- `ngal_no_fit`: no inclusion, second term is omitted
- `ngal_lin_fit_only`: exclusive, first term is omitted

4.5.2 Deduced parameters

The following deduced parameters can be computed:

- Mean galaxy bias
$$b_g(z) = \int \mathrm{d}M\, b_h(M,z)\, n(M,z) \frac{N(M)}{n_{gal}(z)}\,, \tag{26}$$
where b_h is the halo bias, and
$$n_{gal}(z) = \int N(M)\, n(M,z)\, \mathrm{d}M \tag{27}$$
is the total number of galaxies.
- Mean halo mass
$$\langle M_{halo}\rangle(z) = \int \mathrm{d}M\, M\, n(M,z) \frac{N(M)}{n_{gal}(z)}. \tag{28}$$
- Mean number of galaxies **do**
- Galaxy density
- Fraction of satellite galaxies
$$f_s(z) = 1 - f_c(z); \quad f_c(z) = \int \mathrm{d}M\, n(M,z) \frac{N_c(M)}{n_{gal}(z)}. \tag{29}$$

Use the program `add_deduced_halomodel` to add those deduced parameters to a PMC sample. See the example config file `config_pmc_ded` in `Demo/MC_Demo/HOD/CFHTLS-T06`.

4.5.3 Clustering data

The angular two-point correlation function $w(\theta)$ is implemented, with the flag `shalodata = woftheta`. The measured (input) data w_{mes} is corrected for the integral constraint, via
$$w(\theta) = w_{mes}(\theta) + w_C, \tag{30}$$
assuming that the measured correlation function can be fit by a power law
$$w_{mes}(\theta) \approx A_w \left(\theta^{-\delta} - C\right). \tag{31}$$

The program `haloplot` outputs the correlation functions $w(\theta)$ and $\xi(r)$, the HOD function $N(M)$, and deduced parameters for given HOD input parameters.

4.5.4 Comoving volume

The comoving volume is needed to calculate the comoving number density of galaxies, following from the halomodel and the HOD parameters. There are two possibilities to calculate the comoving volume V_C. First, if z_{min} and z_{max} are larger than zero in the HOD parameter file `halomodel.par` (see Table **??**), V_C is computed between those two redshifts. Second, if both numbers are < 0, V_C is weighted by the redshift distribution $n(z)$, see e.g. eq. (28) in **?**. In this weighting, the maximum value of $n(z)$ is set to unity.

Table 3: Redshift distribution types

nofz	Description	$n(z) \propto \ldots$	parameter list
hist	Histogram	$\sum_{i=0}^{n-1} N_i \cdot \mathbb{1}_{[z_i;z_{i+1}]}$	(see text)
single	Single redshift	$\delta_{\mathrm{D}}(z - z_0)$	z_0, z_0
ludo	Fitting function	$(z/z_0)^{\alpha} \exp\left[-(z/z_0)^{\beta}\right]$	$z_{\mathrm{min}}, z_{\mathrm{max}}, \alpha, \beta, z_0$
jonben		$z^a / \left(z^b + c\right)$	$z_{\mathrm{min}}, z_{\mathrm{max}}, a, b, c$
ymmk		$\left(z^a + z^{ab}\right) / \left(z^b + c\right)$	$z_{\mathrm{min}}, z_{\mathrm{max}}, a, b, c$

4.6 BAO

BAO constraints are implemented with two distance measures:

- `smethod = distance_A`

  The distance parameter A is defined in **?** as

  $$A(z) = \frac{D_{\mathrm{V}}(z)}{c/H_0} \frac{\sqrt{\Omega_{\mathrm{m}}}}{z} \tag{32}$$

  where

  $$D_{\mathrm{V}}(z) = \left[f_K^2[w(z)] \frac{cz}{H(z)} \right]^{1/3} \tag{33}$$

  is the spherically averaged distance to redshift z.

- `smethod = distance_d_z`

  The distance parameter d is the ratio of sound horizon r_{s} at drag epoch z_{d} to spherically averaged distance (e.g. **?**),

  $$d(z) = \frac{r_{\mathrm{s}}(z_{\mathrm{d}})}{D_V(z)}. \tag{34}$$

  We use the fitting formula for the drag redshift z_{d} from **?** and calculate the sound horizon as the distance a sound wave can travel prior to z_{d} by numerical integration.

4.7 Redshift distribution

Some of the cosmology modules require a redshift distribution, for example lensing and HOD. Table **??** lists the implemented redshift distributions $n(z)$, via the flag `nofz`.

Each redshift bin can have a different type. The syntax for a redshift bin file is described in Appendix **??**.

All redshift distributions are internally normalised as

$$\int_{z_{\min}}^{z_{\max}} \mathrm{d}z\, n(z) = 1. \tag{35}$$

4.8 CMB and the power spectrum normalisation parameter

The power spectrum normalisation parameter taken as input for CAMB is $\Delta^2_{\mathcal{R}}$, which is the amplitude of curvature perturbations at the pivot scale $k_0 = 0.002\,\mathrm{Mpc}^{-1}$. For lower-redshift probes such as lensing or HOD, the normalisation is described by σ_8, the rms fluctuation of matter in spheres of 8 Mpc/h. To combine those probes in a PMC run, $\Delta^2_{\mathcal{R}}$ has to be an input parameter, and σ_8 a deduced parameter. CMB has to come first in the list of data sets so that CAMB can calculate σ_8, which in turn is handed over to the lensing likelihood.

4.9 Parameter files

Tables **??** - **??** list the contents of the parameter files for basic cosmology, lensing, SNIa and HOD. Proto-types can be found in `$COSMOPMC/par_files`. These files specify the default values of parameters and flags. These default values are over-written if any of those parameter is used for Monte-Carlo sampling.

5 The configuration file

The programs `max_post`, `go_fishing`, `cosmo_pmc`, and `cosmo_mcmc` read a configuration file on startup. Each configuration file consist of two parts:

The first, basic part is common to all four config file types (Table **??**). It consists of (1) the parameter section, (2) the data section and (3) the prior section. The data-specific entries in the data section are listed in Table **??**.

The second part is type-specific. See Table **??** for the PMC part, and Table **??** for the MCMC part. Example files can be found in subdirectories of `$COSMOPMC/Demo/MC_DEMO`.

To create a config file of type `max_post` or `go_fishing` from a PMC config file, the script `config_pmc_to_max_and_fish.pl` can be used.

Some flags are handled internally as integers (enumerations), but identified and set in the config file with strings. The corresponding key word carries the same name as the internal variable, preceded with an 's', e.g. the integer/string pair `lensdata`/`slensdata`.

The prior file, indicated if desired with the flag `sprior`, is a file in `mvdens` format. It specifies a Gaussian prior with mean and covariance as given in the file. Note that the covariance and not the inverse covariance is expected in the file.

Table 4: Basic cosmology parameter file (cosmo.par)

Omega_m	$\Omega_{\rm m}$	Matter density, cold dark matter + baryons
Omega_de	$\Omega_{\rm de}$	Dark-energy density (if $w = -1$, corresponds to Ω_Λ
w0_de	w_0	Dark-energy equation-of-state parameter (constant term)
w1_de	w_1	Dark-energy equation-of-state parameter (linear term, see sde_param)
h_100	h	Dimensionless Hubble parameter
Omega_b	$\Omega_{\rm b}$	Baryon density
Omega_nu_mass	$\Omega_{\nu,\rm mass}$	Massive-neutrino density (so far only for CMB)
N_eff_nu_mass	$N_{\rm eff,\nu,mass}$	Effective number of massive neutrinos (so far only for CMB)
normalization	σ_8	Power-spectrum normalisation at small scales (for normmode==0, see below)
n_spec	$n_{\rm s}$	Scalar power-spectrum index
snonlinear		Power spectrum prescription
	linear	Linear power spectrum
	pd96	**?**
	smith03	**?** halofit model
	smith03_de	**?** + dark-energy correction from icosmo.org
	smith03_revised	**?**, revised halofit model
	coyote10	Coyote emulator version 1, **?**, **?**, **?**
	coyote13	Coyote emulator version 2, **?**
stransfer		Transfer function
	bbks	**?**
	eisenhu	**?** 'shape fit'
	eisenhu_osc	**?** with BAO wiggles
sgrowth		Linear growth factor
	heath	**?** fitting formula
	growth_de	Numerical integration of differential equation for δ (recommended)
sde_param		Dark-energy parameterisation
	jassal	$w(a) = w_0 + w_1 a(1-a)$
	linder	$w(a) = w_0 + w_1(1-a)$
normmode		Normalization mode. 0: normalization=σ_8
a_min	$a_{\rm min}$	Minimum scale factor

Table 11: Data-specific entries in the configuration file's data section

Weak gravitational lensing	`Lensing`	
`slensdata`	string	Data type, one of `xipm`, `xip`, `xim`, `map2poly`, `map2gauss`, `gsqr`, `decomp_eb`, `pkappa`, `map3gauss`, `map3gauss_diag`, `map2gauss_map3gauss`, `map2gauss_map3gauss_diag`, `decomp_eb_map3gauss`, `decomp_eb_map3gauss_diag`
`sdecomp_eb_filter`[a]	string	One of `FK10_SN`, `FK10_FoM_eta10`, `FK10_FoM_eta50`, `COSEBIs_log`
`th_min`[b]	double	Minimum angular scale
`th_max`[b]	double	Maximum angular scale
`path`[b]	double	Path to COSEBIs files
`sformat`	string	Data format of angular scales, one of `angle_center`, `angle_mean`, `angle_wlinear`, `angle_wquadr`
`a1`[c]	double	Linear weight
`a2`[c]	double	Quadratic weight, $w =$ `a1` $\cdot\, \theta/\mathrm{arcmin} +$ `a2` $\cdot\, (\theta/\mathrm{arcmin})^2$
`datname`	string	Data file name
`datname2`[d]	string	Second data file name
`scov_scaling`	string	One of `cov_const`, `cov_ESH09`
`covname`	string	Covariance file name
`covname_M`[e]	string	Covariance mixed term file name
`covname_D`[e]	string	Covariance shot-noise term file name
`corr_invcov`	double	Correction factor for inverse covariance ML estimate, see **?**
`model_file`	string	Parameter file name, e.g. `cosmo_lens`
`sspecial`	string	Additional prior, one of `none` (recommended), `unity`, `de_conservative`

Supernovae type Ia	SNIa	
datname	string	Data file name
datformat	string	Data format, SNLS_firstyear
schi2mode	string	χ^2 and distance modulus estimator type (one of chi2_simple, chi2_Theta2_denom_fixed, chi2_betaz, chi2_dust, chi2_residual)
Theta2_denom[a]	2 doubles	Fixed α, β in χ^2-denominator
zAV_name[b]	string	File with $A_V(z)$ table
datname_beta_d[b]	string	Prior file (mvdens format) on β_d ("-" if none)
add_logdetCov	integer	1 if 0.5 log det Cov is to be added to log-likelihood, 0 if not (recommended; see Sect. **??**)
model_file	string	Parameter file name, e.g. cosmo_SN
sspecial	string	Additional prior, one of none (recommended), unity, de_conservative

Table 11: Data-specific entries in the configuration file's data section (continued).

CMB anisotropies	CMB	
scamb_path	string	/path/to/scamb
data_path	string	/path/to/wmap-data. This path should contain the directory data with subdirectories healpix_data, highl, lowlP, lowlP
Cl_SZ_file	string	File with SZ correction angular power spectrum ("-" if none)
lmax	integer	Maximum ℓ for angular power spectrum
accurate	0 | 1	Accurate reionisation and polarisation calculations in camb
model_file	string	Parameter file name, e.g. cosmoDP.par
sspecial	string	Additional prior, one of none (recommended), unity, de_conservative

WMAP distance priors	CMBDistPrior	
datname	string	Data (ML point and inverse covariance) file
model_file	string	Parameter file name, e.g. cosmo_lens.par
sspecial	string	Additional prior, one of none (recommended), unity, de_conservative

Galaxy clustering (HOD)		GalCorr
shalodata	string	Data type, woftheta
shalomode	string	χ^2 type, one of galcorr_var, galcorr_cov, galcorr_log
datname	string	Data (+variance) file name
covname[a]	string	Covariance file name
corr_invcov[a]	double	Correction factor for inverse covariance ML estimate, see **?**
sintconst_type	string	Integral constaint type, one of constant, random_file
delta[b]	double	Power-law slope δ, for integral constaint
intconst	double	Integral constaint C
area	double	Area [deg^2]
sngal_fit_type	string	Likelihood type, inclusion of galaxy number. One of ngal_lin_fit, ngal_log_fit, ngal_no_fit, ngal_lin_fit_only
ngal[c]	double	Number of observed galaxies
ngalerr[c]	double	Error on the number of observed galaxies
model_file	string	Parameter file name
sspecial	string	Not used for HOD, set to none

Table 11: Data-specific entries in the configuration file's data section (continued).

Baryonic acoustic oscillations		BAO
smethod	string	BAO method, one of distance_A, distance_d_z
datname	string	Data + covariance file name (mvdens format)
model_file	string	Parameter file name, e.g. cosmoDP.par
sspecial	string	Additional prior, one of none (recommended), unity, de_conservative

Table **??** contains a list of input parameters, which can be given as strings to the spar key in the config file.

Table 5: Weak lensing parameter file (cosmo_lens.par)

cosmo_file		Basic cosmology file name (cosmo.par)
nofz_file		Redshift distribution master file
redshift module[a]		(see Table **??**)
stomo		Tomography correlations
	tomo_all	All correlations
	tomo_auto_only	Only auto-correlations (*ii*)
	tomo_cross_only	Only cross-correlations ($i \neq j$)
sreduced		Reduced-shear treatment
	none	No correction
	K10	Fitting-formulae from **?**
q_mag_size[b]	q	Magnification-bias coefficient, $q = 2(\alpha + \beta - 1)$ (see **?**, eq. 16)
sia		Intrinsic alignment model
	none	No intrinsic alignment correction
	HS04	**?** linear model, but using the non-linear power spectrum à la **?**
sia_terms[c]		Intrinsic aligmnent terms to be added
	GI_II	*GI* and *II*
	GI	*GI*
	II	*II*

Table 6: SNIa parameter file (cosmo_SN.par)

cosmo_file		Basic cosmology file name (cosmo.par)
Theta2	$-M\ \alpha\ -\beta\ \beta_z$	Distance modulus parameters

Table 7: HOD parameter file (halomodel.par)

cosmo_file		Basic cosmology file name (cosmo.par)
nofz_file		Redshift distribution master file
redshift module[a]		(see Table **??**)
alpha_NFW	α	Halo density profile slope ($\alpha = 1$ for NFW)
c0	c_0	Concentration parameter at $z = 0$
beta_NFW	β	Concentration parameter slope of mass dependence
smassfct		Halo mass function type
	ps	(**?**), $p = 0, q = 1$
	st	(**?**), $p = 0.3, q = 0.75$
	st2	(**?**), $p = 0.3, q = 0.707$
	j01	(**?**)
shalo_bias	Halo bias type	
	halo_bias_sc	From spherical collapse, see **?**
	halo_bias_tinker05	**?**
	halo_bias_tinker10	**?**
pi_max	$\pi_{\max}$	Maximum π, for integration of $w_p(r_p, \pi)$ (for future release)
shod		HOD type
	berwein02_hexcl	**?** with halo exclusion
Mstellar_min	$M_{*,\min}$	Minimum stellar mass (for future release)
Mstellar_max	$M_{*,\max}$	Maximum stellar mass (for future release)
M_min	$M_{\min}$	Minimal mass for central galaxies $[h^{-1}M_\odot]$
M1	M_1	Scale mass for satellites $[h^{-1}M_\odot]$
M0	M_0	Minimum mass for satellites $[h^{-1}M_\odot]$
sigma_log_M	$\sigma_{\log M}$	Logarithmic dispersion for central galaxies
alpha	α	Slope for satellite mass dependence
eta	η	Fraction of halos with central galaxy of given type

Table 8: Redshift module file (nofz.par)

Nzbin	N_z	Number of redshift bins
snzmode	nz_read_from_files	File mode
nzfile	f_1, f_2, ..., f_{Nzbin}	File names. See Appendix **??** for the file syntax.

Table 9: Basic, common part of the configuration file

`version`	double	Config file version. Upwards compatibility (config file `version` > CosmoPMC version) cannot be guaranteed. Downwards compatibility (config file version < CosmoPMC `version`) is most likely ensured.
		Parameter section
`npar`	integer	Number of parameters
`n_ded`	integer	Number of deduced parameters. The deduced parameters are not sampled but deduced from the other parameters and written to the output files as well
`spar`	string	Parameterisation type, necessary for the wrapping into the individual posterior parameters and for plotting, see Table **??** for possible parameters
`min`	`npar`+`n_ded` doubles	Parameter minima
`max`	`npar`+`n_ded` doubles	Parameter maxima
		Data section
`ndata`	integer	Number of data sets
`sdata`	string	Data set 1
		⋮
`sdata`	string	Data set `ndata`
		Prior section
`sprior`	string	Prior file name ("-" for no prior)
[`nprior`	integer	If `sprior` ≠ "-": Number of parameters to which prior applies]
[`indprior`	`npar` × {0, 1}	If `sprior` ≠ "-": Indicator flags for prior parameters]

Table 10: PMC part of the configuration file

`nsample`	integer	Sample size per iteration
`niter`	integer	Number of iterations
`fsfinal`	integer	Sample size of final iteration is `fsfinal` × `nsample`
`niter`	integer	Number of iterations (importance runs)
`nclipw`	integer	The `nclipw` points with the largest weights are discarded
		Proposal section
`df`	double	Degrees of freedom (`df`=-1 is Gaussian, `df`=3 is 'typical' Student-t)
`ncomp`	integer	Number of components
`sdead_comp`	string	One of 'bury', 'revive'
`sinitial`	string	Proposal type (one of `fisher_rshift`, `fisher_eigen`, `file`, `random_position`)
`fshift`[a]	double	Random shift from ML point ~ $U(-r, r)$; r = `fshift/(max-min)`
`fvar`[a]	double	Random multiplier of Fisher matrix
`prop_ini_name`[b]	string	File name of initial proposal
`fmin`[c]	double	Components have variance ~ $U($`fmin`$, ($`max` − `min`$)/2)$
		Histogram section
`nbinhist`	integer	Number of density histogram bins

Table 12: Input parameters

Name	Symbol	Description
Basic cosmology	(some of them given in `cosmo.par`)	
`Omega_m`	Ω_m	Matter density, cold dark matter + baryons
`omega_m`	ω_m	
`Omega_b`	Ω_b	Baryon density
`omega_b`	ω_b	
`100_omega_b`	$100 \times \omega_\mathrm{b}$	
`Omega_de`	Ω_de	Dark-energy density (if $w = -1$, corresponds to Ω_Λ
`omega_de`	ω_de	
`Omega_nu_mass`	$\Omega_{\nu,\mathrm{mass}}$	Massive-neutrino density (so far only for CMB)
`omega_nu_mass`	$\omega_{\nu,\mathrm{mass}}$	
`Omega_c`	Ω_c	Cold dark matter
`omega_c`	ω_c	
`Omega_K`	Ω_K	Curvature density parameter
`omega_K`	ω_K	
`w0_de`	w_0	Dark-energy equation-of-state parameter (constant term)
`w1_de`	w_1	Dark-energy equation-of-state parameter (linear term, see `sde_param`)
`h_100`	h	Dimensionless Hubble parameter
`N_eff_nu_mass`	$N_{\mathrm{eff},\nu,\mathrm{mass}}$	Effective number of massive neutrinos (so far only for CMB)
`N_eff_nu0`	$N_{\mathrm{eff},\nu,0}$	Effective number of massless neutrinos (default = 3.04, so far only for CMB)
`sigma_8`	σ_8	Power-spectrum normalisation at small scales
`Delta_2_R`	$\Delta^2_\mathcal{R}$	Power-spectrum normalization at large scales (CMB)
`n_spec`	n_s	Scalar power-spectrum index
`alpha_s`	α_s	Running spectral index (so far only for CMB)
`n_t`	n_t	Tensor power-spectrum index
`r`	r	Tensor to scalar ratio
`ln_r`	$\ln r$	
`tau`	τ	Optical depth for reionisation
`A_SZ`	A_SZ	SZ-power spectrum amplitude

Lensing-specific (some of them given in cosmo_lens.par)		
A_ia	$A_{\rm ia}$	Intrinsic alignment (IA) power-spectrum amplitude (**?**)
A_GGI	$A_{\rm GGI}$	IA *GGI* amplitude for third-order (**?**, eq. (42))
theta_GGI	$\theta_{\rm GGI}$	IA *GGI* scale for third-order
A_GII	$A_{\rm GII}$	IA *GII* amplitude for third-order
theta_GII	$\theta_{\rm GII}$	IA *GII* scale for third-order
b_slc	b_0	Source-lens clustering galaxy bias (**?**, eq. (41))
gamma_slc	γ	Source-lens clustering galaxy bias index

Table 12: Input parameters (continued)

SNIa-specific (some of them given in cosmo_SN.par)		
M	$M - \log_{10} h_{70}$	Universal SNIa magnitude
alpha	α	Linear response factor to stretch
beta	β	Linear response factor to color
beta_z	β_z	Redshift-dependent linear response to color
beta_d	$\beta_{\rm d}$	Linear response to the color component due to intergalactic dust

Galaxy-clustering-specific (some of them given in halomodel.par)		
M_min	$M_{\rm min}$	Minimum halo mass for central galaxies $[M_\odot h^{-1}]$
log10_M_min	$\log_{10}[M_{\rm min}/(M_\odot h^{-1})]$	
M_1	M_1	Scale mass for satellite galaxies $[M_\odot h^{-1}]$
log10_M_1	$\log_{10}[M_1/(M_\odot h^{-1})]$	
M_0	M_0	Minimum halo mass for satellite galaxies $[M_\odot h^{-1}]$
log10_M_0	$\log_{10}[M_0/(M_\odot h^{-1})]$	
sigma_log_M	$\sigma_{\log M}$	Dispersion for central galaxies
alpha_halo	$\alpha_{\rm h}$	Slope of satellite occupation distribution
eta	η	Fraction of halos with central galaxy for given type
Mhalo_av*	$\langle M_{\rm h}\rangle$	Average halo mass $[M_\odot h^{-1}]$
log10_M_halo_av*	$log_{10}\langle M_{\rm h}/(M_\odot h^{-1})\rangle$	
b_halo_av*	$\langle b_{\rm h}\rangle$	Average halo bias
N_gal_av*	$\langle N_{\rm g}\rangle$	Average galaxy number per halo
fr_sat*	$f_{\rm s}$	Fraction of satellite galaxies to total
ngal_den*	$n_{\rm g}$	Comoving galaxy number density $[{\rm Mpc}^{-3}h^3]$
log10ngal_den*	$\log_{10} n_{\rm g}$	

6 Post-processing and auxiliary programs

All scripts described in this section are located in `$COSMOPMC/bin`.

6.1 Plotting and nice printing

6.1.1 Posterior marginal plots

Marginals in 1d and 2d can be plotted in two ways, using (1) `plot_contour2d.pl` or (2) `plot_confidence.R`. The first is a `perl` script calling `yorick` for plotting, the second is an R script. The second option produces nicer plots in general, in particular, smoothing workes better without producing over-smoothed contours. Further, filled contours with more than one data set are only possible with the `R` option, `yorick` can only combine several plots with empty contours. The computation time of the `R` script is however much longer.

1. `plot_contour2d.pl` creates 1d and 2d marginals of the posterior, from the histogram files `chi2_j` and `chi2_j_k`.

   To smooth 1d and 2d posteriors with a Gaussian, use `plot_contour2d.pl -n -g FACTOR`. The width of the Gaussian is equal to the box size divided by `FACTOR`. It is recommended to test the smoothing width `FACTOR` by setting it to a negative number which causes both smoothed and unsmoothed curves being plotted. This can reveal cases of over-smoothing. If contours have very different width in different dimension, the addition option `-C` uses the PMC sample covariance (from the file `covar+ded.fin`) as the covariance for the Gaussian. For the final plot, replace `-FACTOR` with `FACTOR` to remove the unsmoothed curves. Remove the option `-n` to add color shades to the 2d contours.

   The file `log_plot` contains the last plot command with all options. This can be used to reproduce and modify a plot which has been generated automatically by other scripts, e.g. `cosmo_pmc.pl`.

2. `plot_confidence.R` creates 1d and 2d marginals of the posterior, from the re-sample file `sample`, which has unit weights for all sample points. Re-sampling can be done with `sample_from_pmcsimu.R` see Sect. **??**.

   Smoothing is done with a kernel density estimation using the `R` function `kde2d`. The kernel width can be set with the option `-g`. The number of grid points, relevant both for smoothing and filled contours, is set with `-N`. Use both `-i` and `-j` options to only plot the 2D marginals of parameters *i*and*j* to save computation time.

6.2 Mean and confidence intervals

From a "mean" output file, containing parameter means and confidence levels, one can create a ps/pdf file using the command mean2eps.pl.

This is equivalent to the following steps (see also essential_cosmo_pmc_run.pl):

- meanvar2tab.pl creates a table with parameter names and values formatted in TEX-format.
- tab2tex.pl wraps a LATEX table header around the table.
- txt2tex.pl wraps a LATEX header around the file.
- lde.sh creates an eps file.

Example:

```
> meanvar2tab.pl -s 1 -p 2 -e iter_9/mean > mean.tab
> tab2tex.pl -s 1.25 mean.tab > mean_in.tex
> txt2tex.pl mean_in.tex > mean.tex
```

6.3 Importance sampling

A PMC simulation file (pmcsim) from an earlier PMC run, corresponding to a sample from posterior p_1, can be used to do importance sampling with another posterior p_2. For that, simply replace the data section of the earlier config file with the corresponding data section of posterior p_2. The command importance_sample creates a new PMC simulation which corresponds to a sample under the posterior product $p_1 \cdot p_2$.

6.4 Bayesian evidence, Bayes' factor

evidence.pl calculates and prints the evidence from a PMC simulation file. The same information is printed to the file evidence during a PMC run.

bayes_factor.pl prints Bayes' factor between two PMC runs together with the Jeffrey scale.

evidence_list.pl prints a list of evidences for a number of PMC runs.

6.5 Reparameterisation

remap.sh swaps and removes parameters from a MCMC or PMC run. The histogram files, mean and covariances are remapped. This is useful if different runs are to be reduced to a common parameter set for comparison or joint plotting. The removal of parameters is equivalent to marginalisation over the corresponding parameter subspace.

For example, suppose there is a SNIa run in directory Sn, and a lensing run in Lensing. SNIa has the following parameters:

```
Omegam Omegade w0de M alpha beta
```

Lensing has the parameters:

```
Omegam sigma8 w0de Omegade h100
```

In Sn, create the file remap.dat with the line

```
0 1 2
```

In Lensing, create the file remap.dat with the line

```
0 3 2
```

In both directories run the command

```
> remap.sh -i iter_<niter-1>
```

which creates sub-directories remap containing symbolic links and/or copies of histogram files to/from iter_{niter-1}, mean, covariance files and updated configuration files.

To create joint marginal plots, simply run e.g.

```
> plot_contours2d.pl -c /path/to/WMAP/remap/config_pmc -n
/path/to/Sn/remap /path/to/Lensing/remap
```

New parameters, sampled from a flat or Gaussian distribution, can be added using add_par_from_prior.pl.

6.6 Analysis

6.6.1 mvdens/mix_mvdens format utilities

See Sect. **??** for a description of the mvdens and mix_mvdens formats.

fisher_to_meanvar.pl reads a mvdens file, inverts the covariance matrix and prints the mean and variance.

corr_coeff.sh reads a mvdens or block matrix file and prints the correlation matrix of the covariance.

diag_mvdens.pl replaces the covariance by its diagonal.

add_par_to_mvdens.pl adds parameters to a mvdens file. Useful, if CosmoPMC is run with additional parameters, and the initial proposal is chosen from a previous run with the reduced parameter set.

6.6.2 PMC simulation/MCM chain utilities

`sample2fixpar.pl` reads a sample file and fixes a parameter by cutting off all points outside a given (narrow) range.

`sample_from_pmcsimu.R` reads a PMC sample file with arbitrary (importance) weights, and creates a re-sample file with unit weights. That means that sample points will occur multiple times. This is required for `plot_confidence.R`.

`pmcsim_cat.pl` concatenates PMC simulation files. Use this to add samples from different iterations with high perplexity, to create a large sample file with reduced sample noise.

6.6.3 PMC proposal and diagnostics

`neff_proposal.pl` calculates the effective number of components (eq. **??**). It is the same quantity which is printed to the file `enc`.

`proposal_mean.pl` (`proposal_var.pl`) creates plots of the proposal component's means (variances) as function of the iteration.

`add_pmc_proposal` adds the proposal density to a PMC sample file. Together with the importance weight, this can be used, for example, to recover the posterior density (eq. **??**.

7 Using and modifying the code

7.1 Modifying the existing code

Note: Code to be used with MPI should not contain global variables and static variables.

7.2 Creating a new module

In this section, the steps required to add a new cosmology module to CosmoPMC are described.

1. Create the directory `newmodule` and create (or copy) files with the necessary code to deal with the data and likelihood. Include files (`*.h`) should be in `newmodule/include`, source files (`*.c`) in `newmodule/src`. Edit the (or create a new) Makefile (in `newmodule`) and add the rules `libnewmodule.so`, `libnewmodule.dylib` and `libnewmodule.a` as well as the rule `clean`.
2. In `wrappers/include/types.h`:

   Define a new data type by extending the enumeration `data_t`. Add the corresponding string (for identification of the module in the configuration file) in the macro `sdata_t(i)`, and increase `Ndata_t` by one.

3. In wrappers/include/all_wrappers.h:

   Add the line

   `#include "newmodule.h"`

4. In tools/include/par.h:

   If necessary, add new parameter types (`p_newparameter`) to enumeration `par_t`, add the corresponding identifier strings to the macro `spar_t`, and increase `Npar_t` by one.

   Optional: Add the parameter name and syntax for different programs (e.g. gnuplot, yorick, TEX) to bin/spar.txt.

5. In wrappers/src/wrappers.c:

   Add the corresponding case to the 'switch' instruction in the function `init_func_t`. This function sets the data type.

6. Create the files wrappers/include/newmodule.h and wrappers/src/newmodule.c. (Those files need to have different names than the files in newmodule/{src,include}.) Write the following functions:

   a) `init_function_newmodule`

   b) `read_from_config_newmodule`

   c) `init_newmodule`

   d) `likeli_newmodule` (returning $\log L$)

   e) `special_newmodule` (optional)

   f) `print_newmodule` (optional)

   To see what these functions are supposed to do, have a look at already existing modules, e.g. in bao.c.

7. In Makefile.main:

   a) In the section "Additional directories", define the path to the new module's directory as

      `NEWMODULE = $(COSMOPMC)/newmodule`

   b) In the section "Libraries", define the library of the new module as

      `LIBNEWMODULE = libnewmodule.$(EXT)`

   c) In the section "Combined cosmo include and linker flags", add the following flags:
      `-I$(NEWMODULE)/include` to the variable `IINCDIRS`
      `-L$(NEWMODULE)` to `LLIBDIRS`
      `-lnewmodule` to `LLIBS`.

8. In exec/Makefile:

Define the new rule:

```
$(LIBNEWMODULE):
        cd $(NEWMODULE) && $(MAKE) $@
```

(The second line has to start with a <TAB> and *not* with spaces.)

9. Optional: Extend newdir_pmc.sh.

7.3 Error passing system

Most of the situations where an error occurs are intercepted by the program. In such a case, a variable `*err` of type `error*` is set via the macros

```
*err = addError(error_type, "message", *err, __LINE__);
```

or

```
*err = addErrorVA(error_type, "formatted message", *err,
                  __LINE__, VA_LIST);
```

printing the current line and function in the code, a message and the error type (negative integer). With

```
testErrorRet(test, error_type, "message", *err, __LINE__,
             return_value);
```

or

```
testErrorRetVA(test, error_type, "formatted message", *err,
               __LINE__, return_value, VA_LIST);
```

a conditional error is produced if the (Boolean) expression `test` is true. The error is transported up the stack to the calling function with the macro

```
forwardError(*err, __LINE__, return_value);
```

Omit `return_value` in case of a void function. This can be used as diagnostics even for errors deep in the hierarchy of functions.

During the calculation of the importance weights, any error is intercepted and the corresponding point does not contribute to the final sample. See Sect. **??** for more details. Therefore, in the routines which calculate the importance weights, the following is used:

```
forwardErrorNoReturn(*err, __LINE__, return_value);
ParameterErrorVerb(*err, param, quiet, ndim);
```

In case of an error, the first line forwards the error but does not return from the current routine. The second line prints the `ndim`-dimensional parameter `param` to stderr (if `quiet!=1`) and purges the error.

To exit on an error, use

```
quitOnError(*err, __LINE__, FILE)
```

This is usually done only from the main program.

More macros and functions regarding error communication and handling can be found in the files `errorlist.h, errorlist.c` which are part of PMCLIB.

Acknowledgements

CosmoPMC was developed with support of the CNRS ANR ECOSSTAT, contract number ANR-05-BLAN-0283- 04 ANR ECOSSTAT.

We thank P. Astier, F. Beaujean, J. Guy, L. Fu, A. Halkola, J. Hartlap, B. Joachimi, J. Lui, K. Markovič, P. Schneider, F. Simpson, R. E. Smith, M. Takada and I. Tereno for discussions and insights which helped to develop the cosmology code.

We thank L. Fu for helping with and testing the lensing E-/B-mode decomposition and the third-order lensing code.

We acknowledge R. .E. Smith and J. A. Peacock for making public their code `halofit`[7], which we implemented into CosmoPMC.

The people from the Coyote project[8] are thanked for making their code public. An adapted version of their emulator is part of this code.

The following people are thanked for providing data or simulations:

M. Kowalski for the Union data (**?**), J. Hartlap and S. Hilbert for the Lensing covariance, computed using ray-tracing through the Millennium Simulation (**?**).

[7] `http://www.roe.ac.uk/~jap/haloes`
[8] `http://www.hep.anl.gov/cosmology/CosmicEmu/emu.html`

PMC references

Introductory papers on PMC

?

?

Comparison of sampling methods including PMC

?

?

Optimisation of the Population Monte Carlo algorithm: Application to constraining isocurvature models with cosm

Main papers on CosmoPMC

?

?

A File formats

A.1 Data files

A.1.1 Lensing

For all `lensdata_t` types, the data format is the same. Each line contains the data for a given angular scale and (arbitrary many) redshift bin pair combinations.

The angular scales are defined as follows. For `lensformat = angle_center`, the fist column contains the angular bin center in arc minutes. For the cases `lensformat = angle_mean`, `angle_wlinear` and `angle_wquadr`, first two columns specify the lower and upper end of the angular bin.

Following the angular information are the data. For N_z redshift bins, $N_z(N_z+1)/2$ columns specify all pair combinations $(ij)_{i\leq j}$ in lexical order, that is $(11)(12)(13)\ldots(1N_z)(22)(23)\ldots(N_zN_z)$.

Note that for `lensdata = xipm` the first N_θ lines of the data file contain ξ_+ for N_θ angular scales, the last N_θ lines contain ξ_-, where the angular scales (first or first two columns) are identical in both halfs.

In the cast of a combined second- and third-order data vector (Sect. **??**), two separate data files are indicated in the config file. This is because for the generalised third-order aperture moments three angular scales have to be specified, and thus the number of columns is different compared to second order. Further, the data vectors can have a different number of scales, since there is no reason that second- and third-order moments be measured at the same scales.

The covariance matrix for `lensdata = xipm` is a single file in block format: It consists of N lines and N columns, where $N = N_\mathrm{s}N_z(N_z+1)/2$ is the length of the data. Usually, N_s is the number of measured angular scales, N_θ, unless there is more than one data point per scale (e.g. for `lensdata = xipm`, $N_\mathrm{s} = 2N_\theta$).

For joint second- plus third-order, the covariance matrix is also in block format, given in a single file. The upper-left block is the second-order covariance, and the lower-right block matrix is the third-order covariance. The number of entries has to be consistent with the two input data files, taking into account the number of angular and redshift bins, and whether the third-order vector is the diagonal or general case.

In both of the above cases of a combined data vector, the lower left and upper right covariance matrix blocks are the cross-correlation.

A matrix element C_{ij} equals $\langle d_i d_j\rangle - \langle d_i\rangle\langle d_j\rangle$, where d_i is the i^{th} data point. In the counting over angular scale and redshift, the former varies faster than the latter[9]. For example, with two redshift bins and three angular scales, the element C_{77} is the data variance for the redshift pair (11) and angular scale θ_1 (starting counting at zero). Or, in other words, the covariance matrix

[9] This was wrongly stated here until version 1.01.

consists of $N_z(N_z+1)/2$ block sub-matrices, each of size $N_s \times N_s$. Each sub-matrix corresponds to one redshift bin combination. It is therefore easy to exclude some redshift bins, by (1) setting the diagonal of a sub-matrix to a very high value, and (2) setting the off-diagonal to zero. (The option to let CosmoPMC do this internally does no longer exists for versions ≥ 1.3.)

A.1.2 SNIa

The SNIa data file in `SN_SALT` format starts with the following two lines:

```
@INTRINSIC_DISPERSION double
@PECULIAR_VELOCITY double
```

The peculiar velocity value is in units of km/s. This is followed by a list of supernovae, one object on each line as follows:

$$\text{name} \;\; z \;\; m \;\; s \;\; c \;\; < m^2 > \;\; < s^2 > \;\; < c^2 > \;\; < ms > \;\; < mc > \;\; < sc >$$

A.1.3 BAO

The BAO distance measures are modeled as Gaussian variables, the data files are in `mvdens` format (see Sect. **??**). In the same file, following the `mvdens` data, there is a list of redshifts, corresponding to where the distances are measured.

A.1.4 CMB

The CMB data for WMAP are the ones released by the WMAP team. They are not included in CosmoPMC and can be obtained e.g. from the LAMBDA site[10].

The SZ correction power spectrum file has two columns in each row containing ℓ and C_ℓ, respectively. The first line has to start with $\ell = 2$.

The CMB distance priors (**?**) are given in `mvdens` format.

A.1.5 Redshift distribution

The first line of a file describing a the redshift distribution for a redshift bin contains the type, see Sect. **??**,

```
# nofz
```

[10] `http://lambda.gsfc.nasa.gov`

This is followed by the list of parameter values, in the order given in Table **??**. Each parameter value has to be in a new line. exeption of the histogram, `nofz = hist`. In that case, the parameter lines are as follows:

$$\begin{array}{cc} z_0 & N_0 \\ z_1 & N_1 \\ \dots & \\ z_{n-1} & N_{n-1} \\ z_n & 0 \end{array}$$

N_i is the number of galaxies in the bin $[z_i; z_{i+1}]$. The last line denotes the upper limit of the last histogram bin $z_n = z_{\max}$, followed by a zero. For `nofz = single`, the file has to contain two identical lines with the value of z_0 in each line.

A.2 Output file names

The default names of all output files are defined in `stdnames.h`. Edit this file and to `make clean; make` to set user-defined file names. Note however that some of the pre-processing scripts expect the default names.

A.3 Multi-variate Gaussian/Student-t (`mvdens`), mixture models (`mix_mvdens`)

The `mvdens` file format is as follows. The first (header) line contains four integers:

$$p \quad \nu \quad B \quad c.$$

Here, p is the number of dimensions, ν the degrees of freedom. For a multi-variate Gaussian, choose $\nu = -1$, and $\nu > 0$ for Student-t. B indicates the number of secondary diagonal of the covariance matrix which are updated during the PMC iterations. For most purposes, B can be set equal to p, which corresponds to the whole matrix being updated. Finally, c is 1 if the matrix is Cholesky-decomposed and 0 otherwise.

This is followed by p doubles indicating the mean, followed by p lines with p doubles each, giving the (symmetric) covariance matrix.

Here is an example of a 5-dimensional multi-variate Gaussian (not Cholesky-decomposed):

```
5 -1 5 0
0.38559 -1.5238 19.338 1.3692 -2.4358
0.0053677 -0.025608 0.00066748 -0.0011893 0.00087517
-0.025608 0.16837 -0.0079163 0.0027364 -0.0035709
0.00066748 -0.0079163 0.0011077 0.0010986 -0.00067815
-0.0011893 0.0027364 0.0010986 0.016716 0.0026266
```

```
0.00087517 -0.0035709 -0.00067815 0.0026266 0.014881
```

The `mix_mvdens` format has two doubles as the header:

$$D \quad p$$

where D is the number of components of the mixture and `ndim` the dimension. This is followed by D blocks specifying the weights w_d (doubles) and data m_d (in `mvdens` format) of the D multi-variate densities of the mixtures.

$$\begin{aligned} &w_1 \\ &m_1 \\ &w_2 \\ &m_2 \\ &\dots \\ &w_D \\ &m_D. \end{aligned}$$

The weights should be normalised, $\sum_{d=1}^{D} w_d = 1$.

In many cases, an `mvdens` file indicates a parameter covariance matrix, for example to be used as Gaussian prior using the config file flag `sprior`. In some cases, the inverse covariance matrix is expected, as in the case of the Fisher matrix.

B Syntax of all commands

All following scripts are located in `$COSMOPMC/bin`. All programs (executables) are located in `$COSMOPMC/exec` and linked from `$COSMOPMC/bin` after running `make` in `$COSMOPMC`.

- `add_comment_to_ps.pl`

```
Usage:add_comment_to_ps.pl [OPTIONS] FILE
Adds username, hostname and current directory as
 comment to a ps file.
OPTIONS:
  FILE        Input ps file
  -l LANG     Language LANG:
               'g': gnuplot (default)
               'y': yorick
  -o OUT      Output file OUT (default stdout)
  -h          This message
```

- add_ded.pl ??

```
Usage: add_ded.pl [OPTIONS] SAMPLE SPAR [PARAMS]
OPTIONS:
   SAMPLE      PMC simulation file, e.g. 'pmcsim'
   SPAR        Deduced parameter name, one in
                [Sigma|q_acc|Omega_K|Omega_de|Omega_m|Omega_b|Omega_c|omega_b|omega_c|wa]
   PARAMS      Further options:
                SPAR             PARAMS
                Sigma            alpha Omega_m_fid
                wa               Na (default 5)
   -c CONF     Config file CONF (default 'config_pmc')
   -h          This message
```

- add_deduced_cosebis ??

```
sh: ../bin/add_deduced_cosebis: No such file or directory
```

- add_deduced_halomodel ??, ??

```
sh: ../bin/add_deduced_halomodel: No such file or directory
```

- add_par_from_prior.pl ??

```
Usage: add_par_from_prior.pl [OPTIONS] sample
Adds a new random parameter to a PMC sample file, drawn under a distribution
OPTIONS:
  -o OUT      Output sample file OUT (default: '<sample>.out'
  -p DIST     Prior distribution, DIST one of 'Flat' (default), 'Gauss'
  -P ARG      Prior arguments (white-spaced list if more than one). For DIST =
               Flat:  ARG = 'min max' (defaut '-1 1')
               Gauss: ARG = 'mean sigma'
  -C COL      Column COL of new parameter (default: last)
  -s STR      Name string STR of new parameter
  -h          This message
```

- add_par_to_mvdens.pl ??

```
sh: ../bin/add_par_to_mvdens.pl: Permission denied
```

- add_pmc_proposal ??

```
sh: ../bin/add_pmc_proposal: No such file or directory
```

- all_vs_all.pl

```
Usage all_vs_all.pl [OPTIONS]
Creates a tex file for a triangle plot with all combinations BASE_i_j.ENDING
OPTIONS:
  -b BASE          Basename of files (required)
  -e ENDING        Default is 'eps'
  -l 1DBASE        If 1d-plots are present
  -s START         Start index (default: 0)
  -t TITLE         Print title string (default: none)
  -d               Include diagonal (i=j)
  -B BOUNDINGBOX   'b1 b2 b3 b4'
  -s STR           Add 'STR' to includegraphics options
  -o FNAME         Output filename (default: stdout)
```

- allps2tex.pl

```
Usage: allps2tex.pl [OPTIONS] [DIR1 [DIR2 [...]]]
Writes a latex file to stdout with all [e]ps[i] files except 'all.ps' in given directories.'
OPTIONS:
  -t TITLE     Title string (default none)
  -w WIDTH     Width of individual postscript files in cm (default 8)
  -f [eps|pdf] Format of input files (default 'ps', includes eps, ps, epsi)
  -r ANGLE     Rotate figues by ANGLE degrees (default 0)
  -B BBOX      'b1 b2 b3 b4'
  -s FSIZE     Font size [pt], default FSIZE=11
  -n           Write file name in tex file after including figure
  -o FNAME     Output filename (default: stdout)
  -H FNAME     Include header from file FNAME
  -h           This message
  DIR1 ...     Directory list (default '.')
```

- ascii2sample.pl

```
defined(@array) is deprecated at ../bin/ascii2sample.pl line 44.
(Maybe you should just omit the defined()?)
ascii2sample.pl [OPTIONS] FILE
Transforms an ascii file, e.g. a MCM chain, into a PMC-compatible sample file
OPTIONS:
  FILE           Input file
  -w COL         Use column COL as log(weight) (default: zero log(weight), or COL=0 if '-p' is given)
  -t WTYPE       Weight type WTYPE = LOG (logarithmic; default), or LIN (linear)
  -c 'C0 C1 ...' Use columns C0, C1, ... as parameters (default: all input columns
                  that not specified as weight columns)
  -s NAMES       String of parameter names to be written in header
  -p             File is in PMC sample format, columns: log(weight) comp param_0 param_1 ...
  -h             This message
The specification of a weight column implies option '-c'
```

- bayes_factor.pl ??

```
Usage: bayes_factor.pl [OPTIONS] DIR1 DIR2
Calculates the Bayes factor between models. The corresponding
 evidence files (from PMC) have to be in the directories DIR1 and DIR2
OPTIONS:
  -i 'ITER1 [ITER2]'  Use iteration ITER1 for DIR1 and ITER2 for DIR2
                       (default: all iterations)
  -f 'EVI1 [EVI2]'    Use files DIR1/EVI1 and DIR2/EVI2 (default: 'evidence')
  -s                  Short output, last iteration only
  -l                  Laplace approx. from Fisher matrix (denoted with iter=-1)
  -h                  This message
```

- cl_one_sided ??

```
sh: ../bin/cl_one_sided: No such file or directory
```

- config_pmc_to_max_and_fish.pl ??, ??

```
Usage: config_pmc_to_max_and_fish.pl [OPTIONS]
OPTIONS:
  -M           Create config file for maximum search (max_post)
```

```
  -F            Create config file for Fisher matrix (go_fishing)
  -c CONFIG     Input PMC config file CONFIG (default: 'config_pmc')
  -r            Random starting point (for maximum search)
  -f FID        Fiducial starting point FID. FID is a white-space
                 separated list in quotes, e.g. '0.25 0.75'
  -p FILE       Fidcucial parameter from FILE (e.g. 'maxlogP')
  -t TOLERANCE  Tolerance for maximum-search (default: 0.01)
  -d            Calculate only diagonal of Fisher matrix (go_fishing)
  -h            This message
One of '-M' or '-F' is obligatory
The default starting point for maximum search is (max-min)/2
For Fisher matrix ('-F'), a fiducial parameter has to be indicated with '-f FID'
 or '-p FILE'
```

- corr_coeff.sh ??

```
Usage: corr_coeff filename [mvdens|block]
```

- cosmo_mcmc

```
sh: ../bin/cosmo_mcmc: No such file or directory
```

- cosmo_pmc ??

```
sh: ../bin/cosmo_pmc: No such file or directory
```

- cosmo_pmc.pl ??, ??, ??

```
Usage: cosmo_pmc.pl [OPTIONS]
OPTIONS:
   -n NCPU              Run PMC in parallel on NPCU cpus using 'mpirun' (default: 1)
   -c CONFIG            Configuration file for PMC (default: config_pmc)
   -f FID               Fiducial starting point FID. FID is a white-space
                         separated list in quotes, e.g. '0.25 0.75'
   -r                   Random starting point for maximum search
                         (default: (max-min)/2)
   -m [c|a]             Maximum-search method: 'c' (cg), 'a' (amoeba)
   -d                   Calculate only diagonal of Fisher matrix
   -D                   Do not force Fisher matrix F to be positiv. If F is negative,
                         script exits with an error
   -a                   Adaptive numerical differentiation for Fisher matrix
   -s SEED              Use SEED for random number generator. If SEED=-1 (default)
                         the current time is used as seed.
   -S [M|F]             Stops after maximum search ('M') or Fisher matrix ('F')
   -A [y|n]             Default answer to all questions on stdin
   -P PATH              Use PATH as CosmoPMC directory (default: environment
                         variable $COSMOPMC)
   -e                   Create 'essential' plots
   -p PRO               Plotting of marginalized posterior (1d and 2d):
                         PRO = 'y' (yorick; default), 'R' (R), 'n' (none),
                         'o' (only). Letters can be combined, e.g. 'yRo'.
                         Combinations of letters are possible, e.g. 'yR' or 'oy'
   -M MULT              Output sample MULT times input (default 1).
                         Valid if plotting script is 'R'
   -O OPT               Pass options OPT to 'plot_contour2d.pl'
```

```
   -q                  Quiet mode
   -h                  This message
```

- diag_mvdens.pl ??

```
Usage: diag_mvdens.pl IN
    Prints the mvdens file 'IN' with the covariance replaced by its diagonal.
```

- essential_cosmo_pmc_run.pl ??

```
Usage: essential_cosmo_pmc.pl [OPTIONS]
OPTIONS:
    -c CONFIG      Uses config file CONFIG (default: 'config_pmc')
    -P PATH        Use PATH as CosmoPMC directory (default: environment
                    variable $COSMOPMC)
    -k             Keep temporary files
    -v             Verbose
    -h             This message
```

- evidence.pl ??

```
Usage: evidence.pl [OPTIONS] SAMPLE
OPTIONS:
   -h         This message
SAMPLE        PMC sample file
```

- evidence_list.pl ??

```
Usage: evidence_list.pl [OPTIONS] DIR1 [DIR2 [...]]
OPTIONS:
  -r N            Subtract log(E) from DIRN (default: no subtraction)
                  For N=-1 subtract log(E_min)
  -k KEY          Use KEY (string list) instead of
                  directory names (default)
  -s SEP          Use SEP as input separator for KEY list
  -S SEP          Use SEP as output separator
                  (default for both: white-space)
  -n              Write number of model parameters
  -L              Use Laplace approximation (reading file 'evidence_fisher')
  -h              This message
```

- fisher_to_meanvar.pl ??

```
fisher_to_meanvar.pl [OPTIONS] file
OPTIONS:
    -n             No inverse
    -m             Marginal errors (don't invert matrix)
    -x             mixmvdens format (default: mvdens format)
    -k             Keep temporary file 'fishtmp.i'
    -h             This message
Options '-m' and '-n' exclude each other
```

- get_spar.pl

```
Usage: get_spar.pl [OPTIONS] LANG [PAR1 [PAR2 [...]]]
OPTIONS:
   -c CONFIG          Configuration file ONFIG (default 'config_pmc')
   -i INDEX           Returns only par[INDEX]
```

```
    -P PATH           Use PATH as CosmoPMC directory (default: environment
                       variable $COSMOPMC)
    -p                Print 'p<i> for unknown parameters instead of input string
    LANG              One of 'yorick', 'gnuplot', 'TeX', 'R'.
                       More languages can be defined in spar.txt
    PAR1 ...          Prameter strings
```

- go_fishing ??, ??, ??

```
sh: ../bin/go_fishing: No such file or directory
```

- haloplot ??

```
sh: ../bin/haloplot: No such file or directory
```

- histogram.pl

```
Usage: histogram.pl OPTIONS FILE [COL]
OPTIONS:
  FILE             File with list of numbers
  COL              Column number or name (default 0)
  -c               Force COL to be name (useful if column name is alphanumeric)
  -b binwidth      Width of bins
  -N Nbin          Number of bins
  -n               Normalized histogram
  -l               Left corner rather than bin center in output
  -L               Logarithmic bins
  -m min, -M max   Minimum and maximum (by default determined from data)
  -w COL           Weights in column COL
  -o OUT           Output file (default stdout)
  -H               No header is printed
  -f               Output in 'nicaea' histogram format. Sets options '-l -H'
  -q               Quiet mode
  -h               This message
  Either the binwidth or number of bins have to be given
```

- histograms_sample ??, ??

```
sh: ../bin/histograms_sample: No such file or directory
```

- importance_sample ??

```
sh: ../bin/importance_sample: No such file or directory
```

- max_post ??, ??, ??

```
sh: ../bin/max_post: No such file or directory
```

- mean2eps.pl ??

```
Usage: mean2eps.pl [OPTIONS] MEAN
OPTIONS:
    MEAN           File containing mean and confidence levels (output of
                    'cosmo_pmc' or 'histograms_sample'
    -c CONFIG      Uses config file CONFIG (default: 'config_pmc')
    -P PATH        Use PATH as CosmoPMC directory (default: environment
                    variable $COSMOPMC)
    -o BASE        Outname BASE (default: <MEAN>)
    -v             Verbose
```

```
    -h            This message
```

- meanvar2tab.pl ??

```
defined(@array) is deprecated at ../bin/meanvar2tab.pl line 72.
(Maybe you should just omit the defined()?)
Usage: meanvar2tab.pl [OPTIONS] file [file2 [...]]

Options:
  -s {123}     68% (1), 95% (2) or 99.7% (3) errors (default = 1)
  -p PREC      Output with PREC digits ('%PREC' format string)
  -e           Error(s) written to PREC significant digits (use -p PREC)
  -c CONFIG    Uses config file CONFIG (default: 'config_pmc')
  -t TITLE     Title (table heading) TITLE is string list with entries according
               to the number of input files
  -S SEP       Use SEP as input separator for TITLE list (default: white space)
  -P PATH      Use PATH as CosmoPMC directory (default: environment
                variable $COSMOPMC)
  -h           This message
```

- meanvar_sample ??

```
sh: ../bin/meanvar_sample: No such file or directory
```

- neff_proposal.pl ??, ??

```
Usage: neff_proposal.pl PROP
   Calculates the effective number of components for the mix_mvdens file 'PROP'
```

- newdir_pmc.sh ??, ??

```
Usage: newdir_pmc.sh [DIR]
Directory DIR (default: read on input) is created.
Links are set to data files in \$COSMOPMC/data.
Parameter files are copied on request from \$COSMOPMC/par_files.
```

- plot_confidence.R ??, ??, ??, ??

```
sh: ../bin/plot_confidence.R: /opt/local/bin/Rscript: bad interpreter: No such file or directory
```

- plot_contour2d.pl ??, ??, ??, ??

```
Usage: plot_contour2d.pl [OPTIONS] [DIR1 [DIR2 [...]]]

OPTIONS:
  -i NITER       Number of iterations (needed if do_proposal=2)
  -c CONFIG_FILE Configuration file (default: in order config_mcmc, config_pmc)
  -t TITLE       Title string for each panel (default empty)
  -T TITLE       Title string for all_contour2d.{eps|pdf} (default empty)
  -n             No shade
  -w WIDTH       Line width WIDTH (default 4)
  -1 OPT         Add 1d posterior plots. OPT can contain the following letters:
                  m    Plot line at mean position
                  123  Plot line at 68%,95%,99.7 density
                  t    Write mean and 68% confidence intervals as text
                        (use with 'm' and '1'
                  n    None of the above
  -S             All contours with solid lines
```

```
  -s N           Outermost level is N sigma
  -r             Aspect ratio=1, changes plot limits such that dx=dy
  -g FACTOR      Gaussian smoothing of 2d-histograms with variance
                  box-width/|FACTOR|. If FACTOR is negativ, plots
                  unsmoothed histogram in addition (use with '-n').
                  Note: For multiple contours, use a list of values "g1 g2 ..."
  -G FACTOR      Gaussian smoothing of 1d-histograms (default: 2d factor)
  -C             Use covariance (file covar.fin) for Gaussian smoothing
  -N NORM        Normalisation of 1d posterior
                  'm'  Maximum = 1 (default)
                  'i'  Integral over posterior = 1
  -F NUM         Color scheme, NUM=0,1,2
  -k             Add key to plots
  -K "KEY1 [KEY2 [...]]" Key strings (default: directory names)
  -y FS          Font size FS (default 24)
  -o FORMAT      Output file format, FORMAT=eps|pdf (default: eps)
  -b             Writes the chi2 files in block format
  -m PAR         Plots a mark at position PAR (e.g. best-fit). PAR is white-space
                  separated list (use quotes or '\ ', e.g. '0.3 0.8')
  -P PATH        Use PATH as CosmoPMC root directory (default: environment
                  variable $COSMOPMC)
  -q             Run quietly, no verbose
  -h             This message
  DIR1 ...       List of directories containing histogram files (chi2_*_*)
                  Default: DIR1 = '.'
```

- pmcsim_cat.pl **??**

```
Usage: pmcsim_cat.pl [OPTIONS]
Concatenates PMC simulations (files 'pmcsim').
OPTIONS:
  -d DEPTH   Descend to DEPTH subdirectories. Default value is 1. No descend is 0.
  -i IN      Input file IN (default: 'pmcsim')
  -o OUT     Output file OUT (default: '<pmcsim>.cat')
  -q         Quiet mode
  -h         This message
```

- proposal_mean.pl **??**, **??**

```
Usage: proposal_mean.pl [OPTIONS]
OPTIONS:
  -d DIR         Directory DIR containing the sub-directories 'iter_*'
                  with the proposal files (default '.')
  -c CONFIG      Configuration file CONFIG (default 'DIR/config_pmc')
  -n             No plotting, only creates '.gnu' file
  -i             x- and y-axes inverted
  -I             x- and y-labels on top/right
  -P PATH        Use PATH as CosmoPMC root directory (default: environment
                  variable $COSMOPMC)
  -h             This message
```

- proposal_var.pl **??**, **??**

```
Usage: proposal_var.pl [OPTIONS]
```

```
OPTIONS:
  -d DIR         Directory DIR containing the sub-directories 'iter_*'
                  with the proposal files (default '.')
  -c CONFIG      Configuration file CONFIG (default 'DIR/config_pmc')
  -P PATH        Use PATH as CosmoPMC root directory (default: environment
                  variable $COSMOPMC)
  -h             This message
```

- remap.sh ??

```
Usage: remap.sh [OPTIONS]
OPTIONS:
   -c CONFIG            Input PMC configuration file (default './config_pmc')
   -i INPUT             Input directory INPUT (default '.')
   -s PMCSIM            Sample/PMC simulation file PMCSIM
   -o OUTPUT            Output directory OUTPUT (default './remap')
   -r REMAP             Remap file REMAP (default './remap.dat')
   -n NPAR              Number of parameters NPAR (default: read from remap file)
   -d N_DED             Number of deduced parameters N_DED (default: 0)
   -h                   This message
```

- sample2fixpar.pl ??

```
Usage: sample2fixpar.pl SAMPLE_IN COL MIN MAX
    SAMPLE_IN          Input sample (PMC simulation or MCM chain)
    COL                Column number of fixed parameter
                        (Note that par #i is in column i+2)
    MIN, MAX           Minimum and maximum values for fixed parameter
```

- tab2tex.pl ??

```
Smartmatch is experimental at ../bin/tab2tex.pl line 106.
Usage: tab2tex.pl [OPTIONS] file

OPTIONS:
   -a           Produce tex array, not tex table
   -b           Bare output, no table/array header
   -s STRETCH   Set arraystretch to STRETCH
   -S SIZE      LaTeX font size, e.g. 'small'
   -m           Add '$' around entries (tex inline math mode)
   -l MODE      Print vertical lines between rows according to MODE;
                 a    all lines (default)
                 n    no lines
                 h    header lines
   -L MODE      Print horizontal lines between columns according to MODE:
                 a    all lines (default)
                 n    no lines
   -t TITLE     Title string TITLE
   -r CHR       Replace character CHR with space in output (unless in math mode)
   -h           This message
```

- test_suite_cosmo_pmc.pl

```
Usage: test_suite_cosmo_pmc.pl [OPTIONS]
OPTIONS:
```

```
-r              Do PMC test runs
-R              Only do PMC test runs
-n NCPU         Run PMC in parallel on NCPU cpus (default: 1)
-c              Include CMB tests
-P PATH         Use PATH as CosmoPMC root directory (default: environment
                 variable $COSMOPMC)
-s              Short, without time-taking PMC runs (e.g. Lensing/COSMOS-S10)
-k              Keep temporary files
-x              Clean previous run and exit
-v              Verbose
-h              This message
```

- xi2xi.pl

```
sh: ../bin/xi2xi.pl: No such file or directory
```

C MCMC

We provide a Metropolis-Hastings Monte-Carlo Markov Chain sampler, which is included in the CosmoPMC package. This MCMC implementation has been used in **?** in the comparison with PMC. In the following, we briefly describe our MCMC program.

C.1 MCMC configuration file

C.2 Proposal and starting point

The proposal for the Metropolis-Hastings algorithm is a multi-variate Gaussian distribution. After choosing an initial proposal, a new proposal can optionally be re-calculated after a number of `ncov` (accepted) steps. The covariance of this new proposal is the chain covariance from steps up to this point. This proposal is then updated after each `ncov` accepted steps using all previous accepted points.

There are several options for the initial proposal:

1. **`sinitial = diag`** A diagonal covariance with width being a fraction of the box size.
2. **`sinitial = Fisher`** The Hessian at a given point in parameter space. If this point is the maximum-likelihood point, the Hessian corresponds to the Fisher matrix.
3. **`sinitial = Fisher_inv`** The inverse Hessian/Fisher matrix, e.g. the covariance from a previous chain. This can be useful for ill-conditioned matrices which are difficult to invert numerically.
4. **`sinitial = previous`** A proposal read from a file, e.g. from a previous MCMC run.

The starting point is either chosen randomly or specified in the config file. The second case might be convenient if the prior volume is very large and a very long burn-in phase is to be

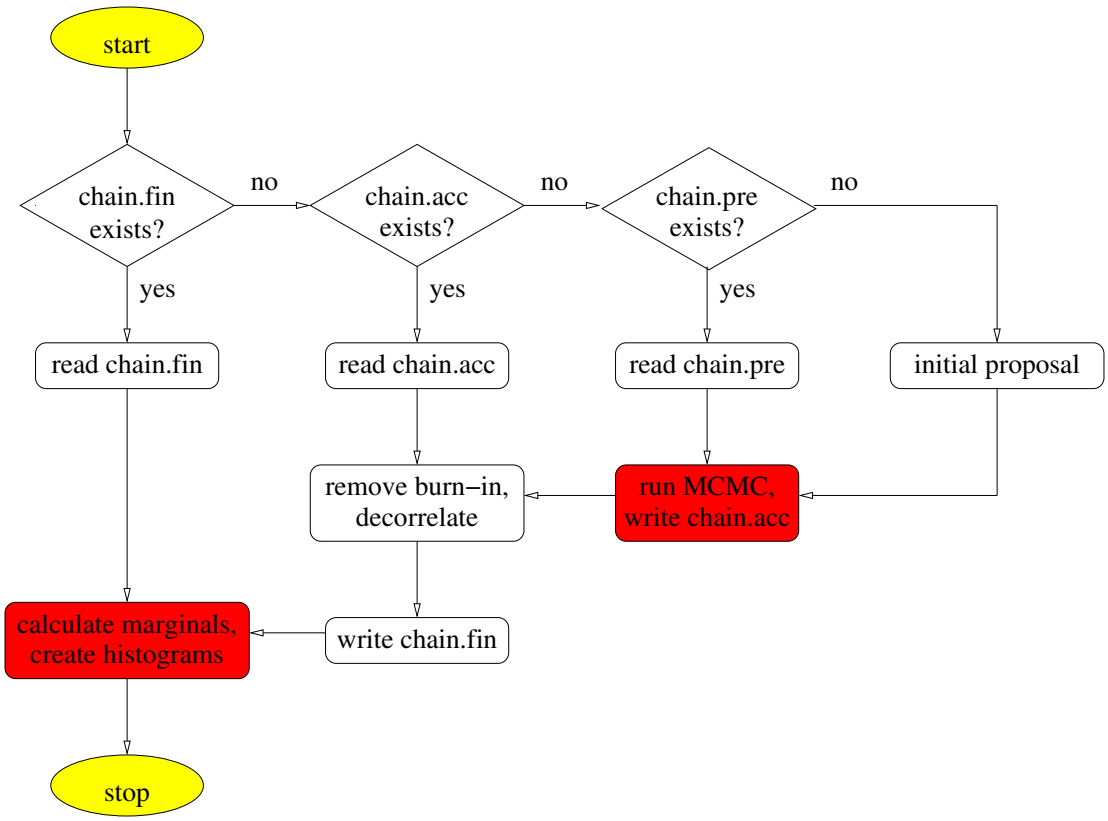

Figure 2: Flow chart of the MCMC implementation.

avoided. For example, the ML-point or best-fit value from a previous experiment can be chosen **?**.

C.3 Output files

The MCMC output files have the same format as their PMC counterparts (see Sect. **??**).

A complete run of `cosmo_mcmc` produces three files containing the points of the Markov chain:

1. `chain.all` containing all, accepted and rejected, sample points. This is the only chain file will not be read or used in subsequent calls of `cosmo_mcmc`.
2. `chain.acc` containing the accepted points.
3. `chain.fin` containing the accepted points after removal of the burn-in phase and after decorrelating (thinning-out) the chain. The results produced by `cosmo_mcmc` (mean, errors, histograms, covariance) are based on this file.

Table 13: MCMC section of the configuration file

nchain	integer	Chain Length
ncov	integer	Interval between updates of the proposal covariance
fburnin	double	Burn-in phase are the first ncov×ncor points
ndecorr	double	De-correlation (thinning-out): one in ndec points is kept in the final chain
fudge	double	Proposal covariance is multiplied by fudge^2/n_par
sinitial	string	Initial proposal type, one of Fisher_inv, Fisher, Fisher, previous, Hessian, Hessian_diag, diag.
boxdiv[a]	double	Diagonal of proposal covariance is (max-min)/boxdiv
sstart	string	Starting point type, one of ran, fid, min, max, nul
fid[b]	npar doubles	Starting parameter
	Histogram section	
nbinhist	integer	Number of density histogram bins

[a] only if sinitial = diag
[b] only if sstart = fid

The chains are ASCII-files, in the same format as the PMC sample files. All weights are 1, and the second column contains the log-likelihood (only in chain.all.

The parameter mean and confidence intervals are printed to the file mean. The names of files containing the histograms and parameter covariances are the same as for PMC.

C.4 Diagnostics

In general it is not straight-forward to diagnose an MCM chain. There exists tests but no formal proofs for convergence (e.g. Gellman-Rubin), which in addition require very long or multiple chains. We have not implemented such tests in the code. However, there are a few (rather hand-waving) diagnostic tools to check the reliability of an MCMC run.

Firstly, the acceptance rate η should be in the range between 15% and 25%. A larger η most probably corresponds to a chain which stayed mainly in the high-density region and strongly under-sampled the lower-density posterior regions. In that case the error bars will be underestimated. A very small η means probably an under-sampling of the posterior since only few points are accepted. However, this need not cause a bias for the parameters and errors if the chain has been run long enough.

C.5 Resuming an interrupted run

Sometimes a MCMC run is interrupted before finishing, or one wishes a previous run to be extended, for example because its convergence is doubted. The MCMC program allows to read in and extend a previous chain. To that end, rename the file `chain.acc` into `chain.pre`. The proposal for the resumed run can but need not be calculated from the previous chain (to be controlled in the config file, see Sect.**??**). In the config file, the number of desired sample points has to be larger than the previous chain.