NumCosmo: Numerical Cosmology Library

LIneA – Bootcamp

Sandro D. P. Vitenti[1], Mariana Penna-Lima[1] and Cyrille Doux[2]

[1] Universidade de Brasília (UnB)

[2]University of Pennsylvania

Observatório Nacional – Rio de Janeiro – September 04, 2019

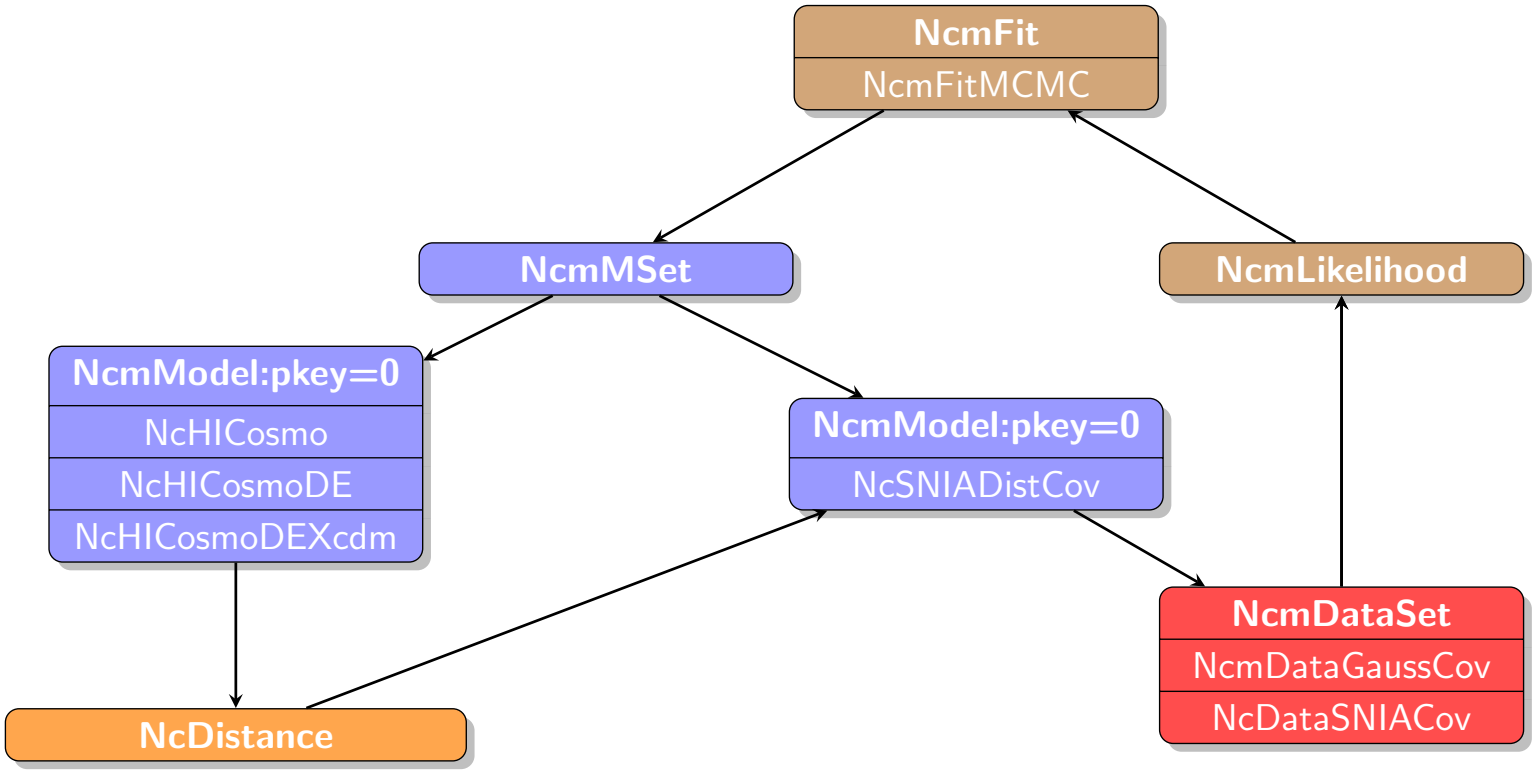
Computational design

- Free software C library (GPL license);
- Repository: https://github.com/NumCosmo/NumCosmo ;
- Continuous integration: https://travis-ci.org/NumCosmo/NumCosmo ;
- Main submodules:
  - NumCosmoMath: a set of computational tools to construct models and likelihoods, and perform statistical analyses;
  - NumCosmo: a set of cosmological (astrophysical) observables, likelihoods (including multiple probes) built on top of NumCosmoMath.
- GLib/GObject: object-oriented programming in C, reference based garbage collection, . . . ;
- GObjectIntrospection: automatic bindings for Python, Perl, Java, . . . it also allows extensions in other languages;
- Serialization framework: transform an object or a set of objects in a sequence of bytes that can be save (loaded) to (from) disk, transfered between processes, ...

NumCosmo - Basic concepts

- `Ncm` prefix – stands for `NumCosmoMath` which contains the general codes not specific to cosmology, e.g., `NcmVector`.
- `Nc` prefix – represents the cosmological and astrophysical codes written on top of Ncm.
- `NcmModel` – is a `GObject` and an abstract class which basically defines a set of parameters and a common interface through a set of virtual methods.
- `NcmMSet` – is a collection of models which can contain models of the same and/or different families and the free parameters specification.
- `NcmData` – is an abstract class which encapsulates functions such as resample, $-2 \ln L$, bootstrap resample, ...
- `NcmDataSet` – is a collection of `NcmData` (combined probes).
- `NcmLikelihood` – is a combination of a `NcmDataSet` and priors (`NcmPrior`).
- `NcmFit` – is the final object that utilizes a `NcmLikelihood` and a `NcmMSet` to perform different statistical analyses.

Example 1: Type Ia Supernova – Update all

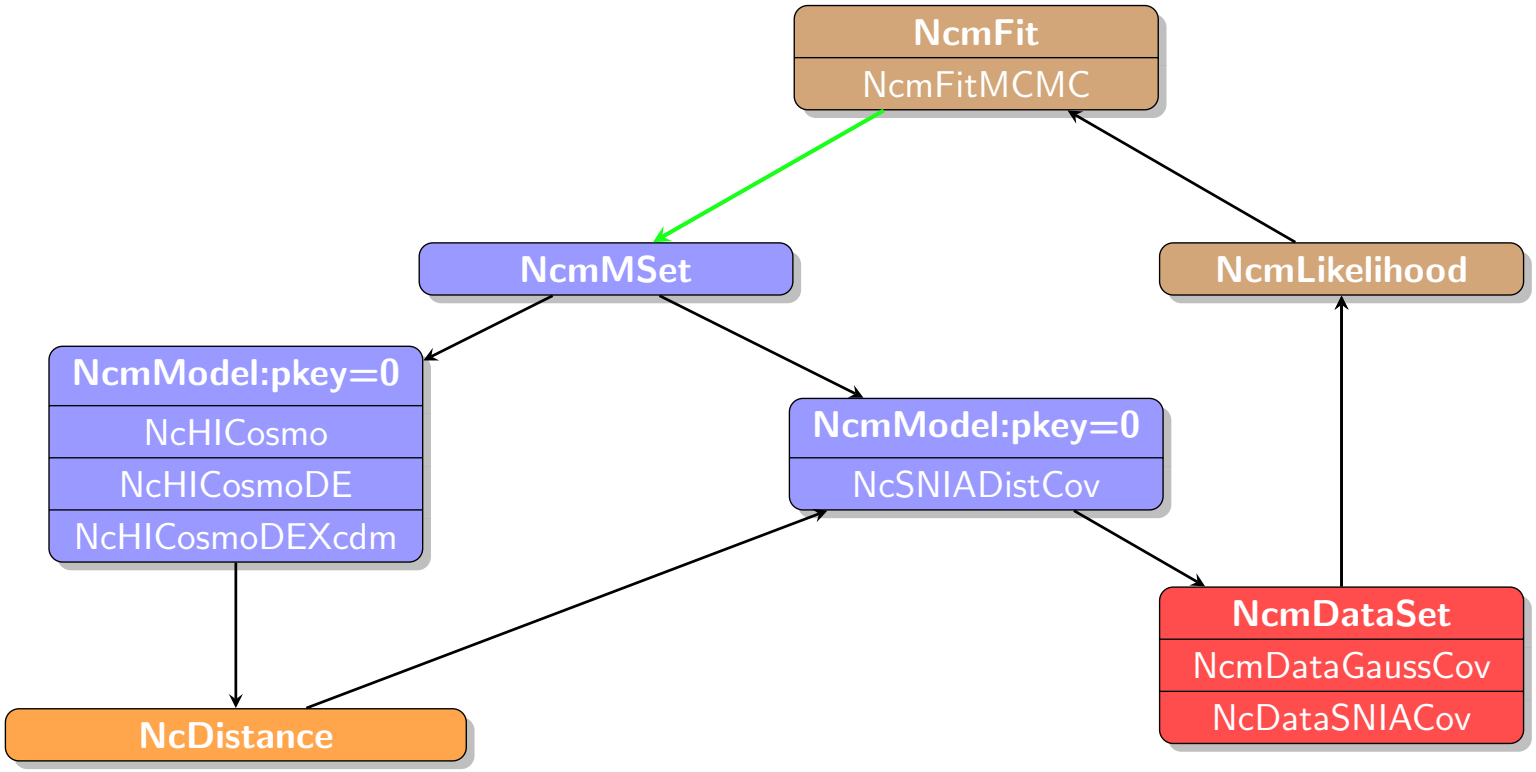


$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T \mathrm{C}_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\mathrm{hel}}, z_i^{\mathrm{cmb}})) + \alpha X_i - \beta \mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

α and β are related to the stretch-luminosity and colour-luminosity, respectively, and M_{h_i} are absolute magnitudes.

Example 1: Type Ia Supernova – Update all

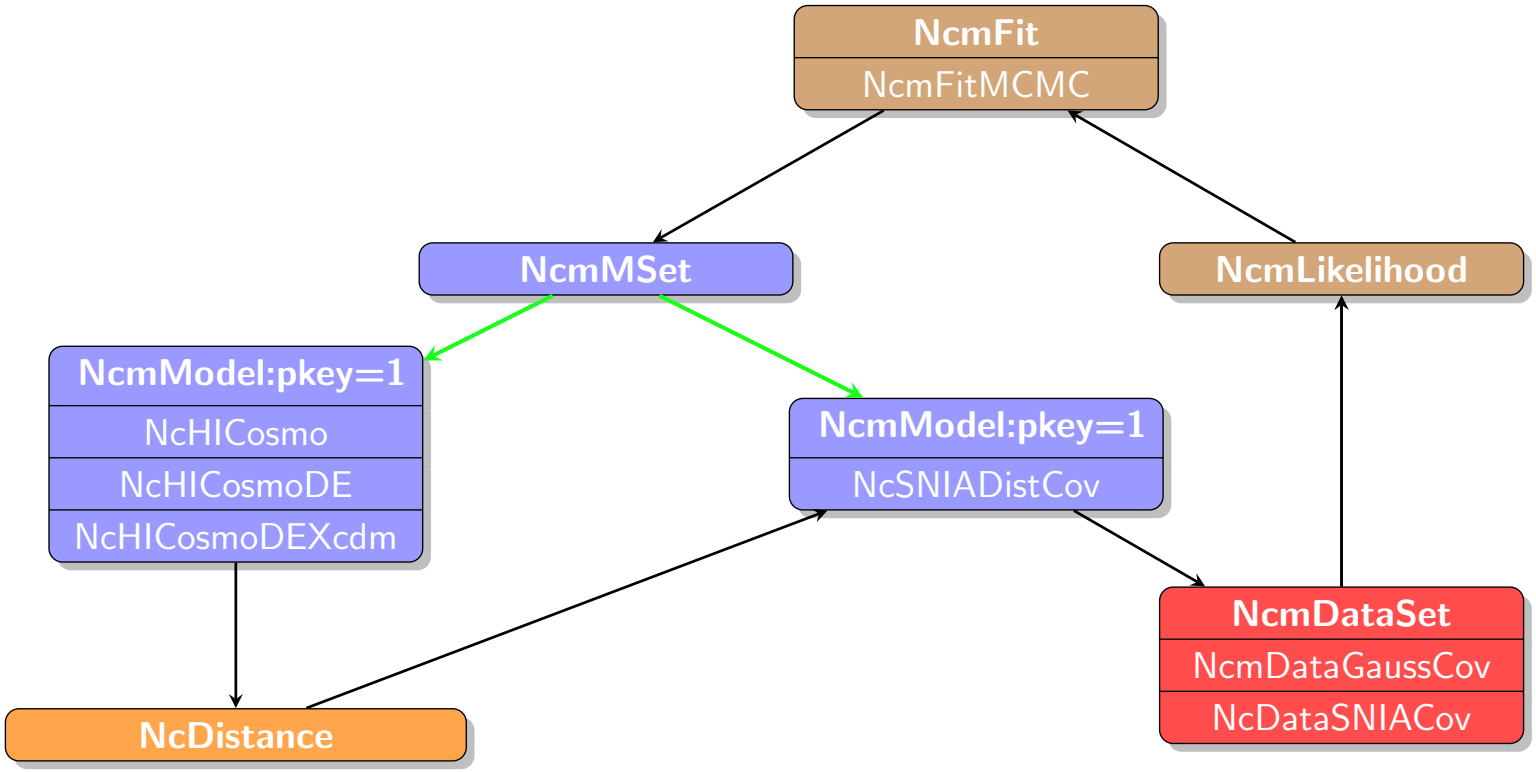


$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T \mathrm{C}_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\mathrm{hel}}, z_i^{\mathrm{cmb}})) + \alpha X_i - \beta \mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

α and β are related to the stretch-luminosity and colour-luminosity, respectively, and M_{h_i} are absolute magnitudes.

Example 1: Type Ia Supernova – Update all

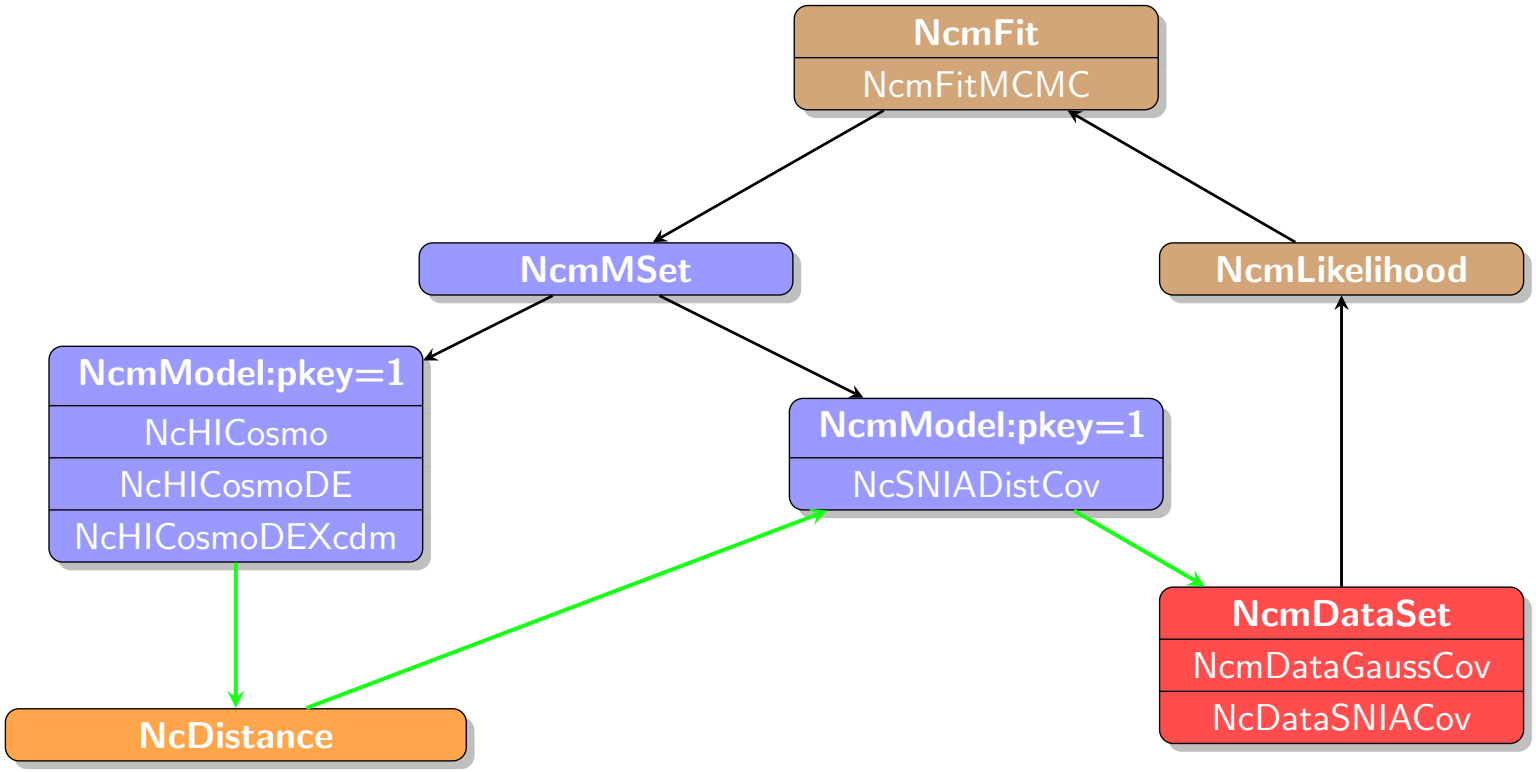


$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T \mathrm{C}_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\mathrm{hel}}, z_i^{\mathrm{cmb}})) + \alpha X_i - \beta \mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

α and β are related to the stretch-luminosity and colour-luminosity, respectively, and M_{h_i} are absolute magnitudes.

Example 1: Type Ia Supernova – Update all

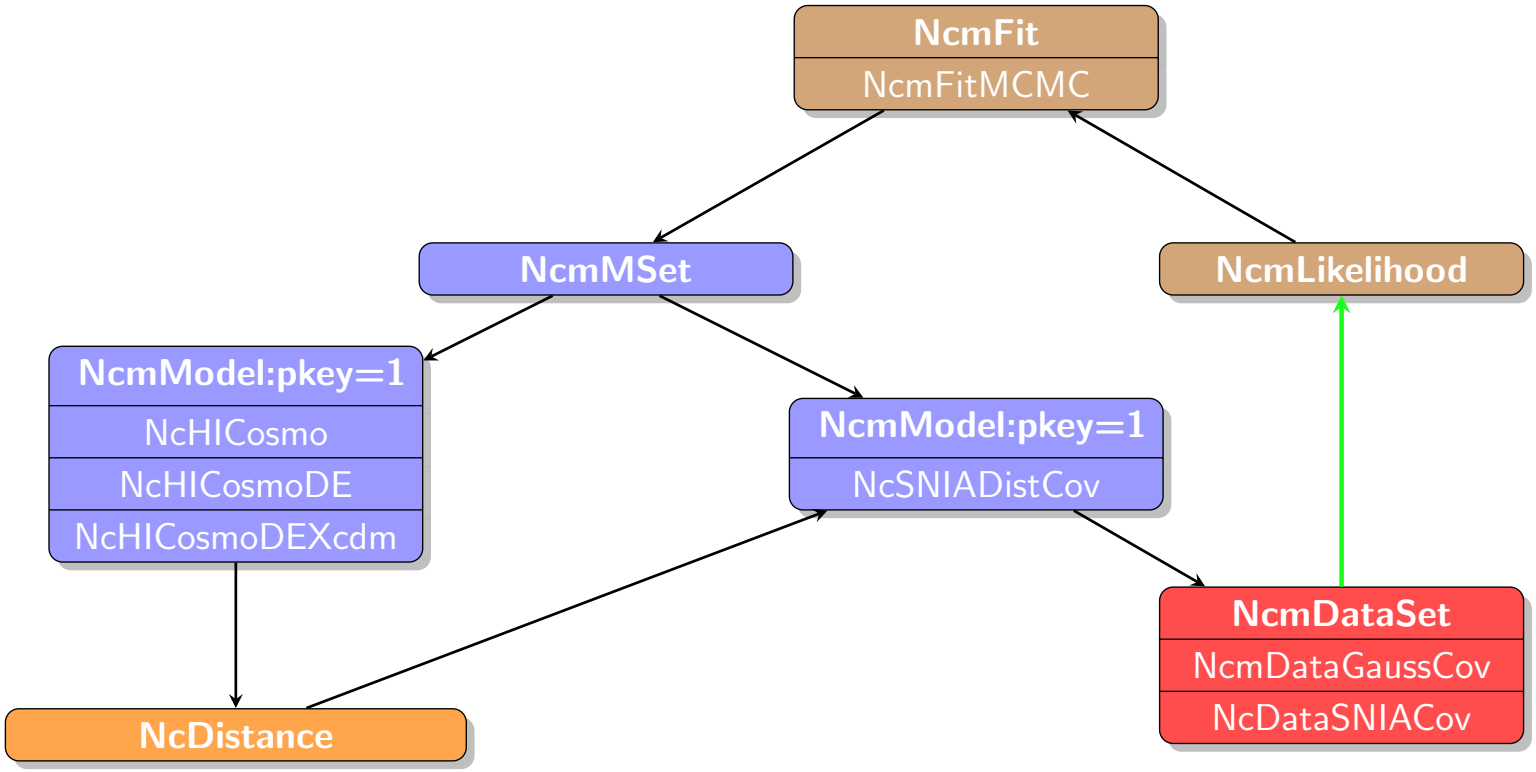


$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T \mathrm{C}_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\mathrm{hel}}, z_i^{\mathrm{cmb}})) + \alpha X_i - \beta \mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

α and β are related to the stretch-luminosity and colour-luminosity, respectively, and M_{h_i} are absolute magnitudes.

Example 1: Type Ia Supernova – Update all

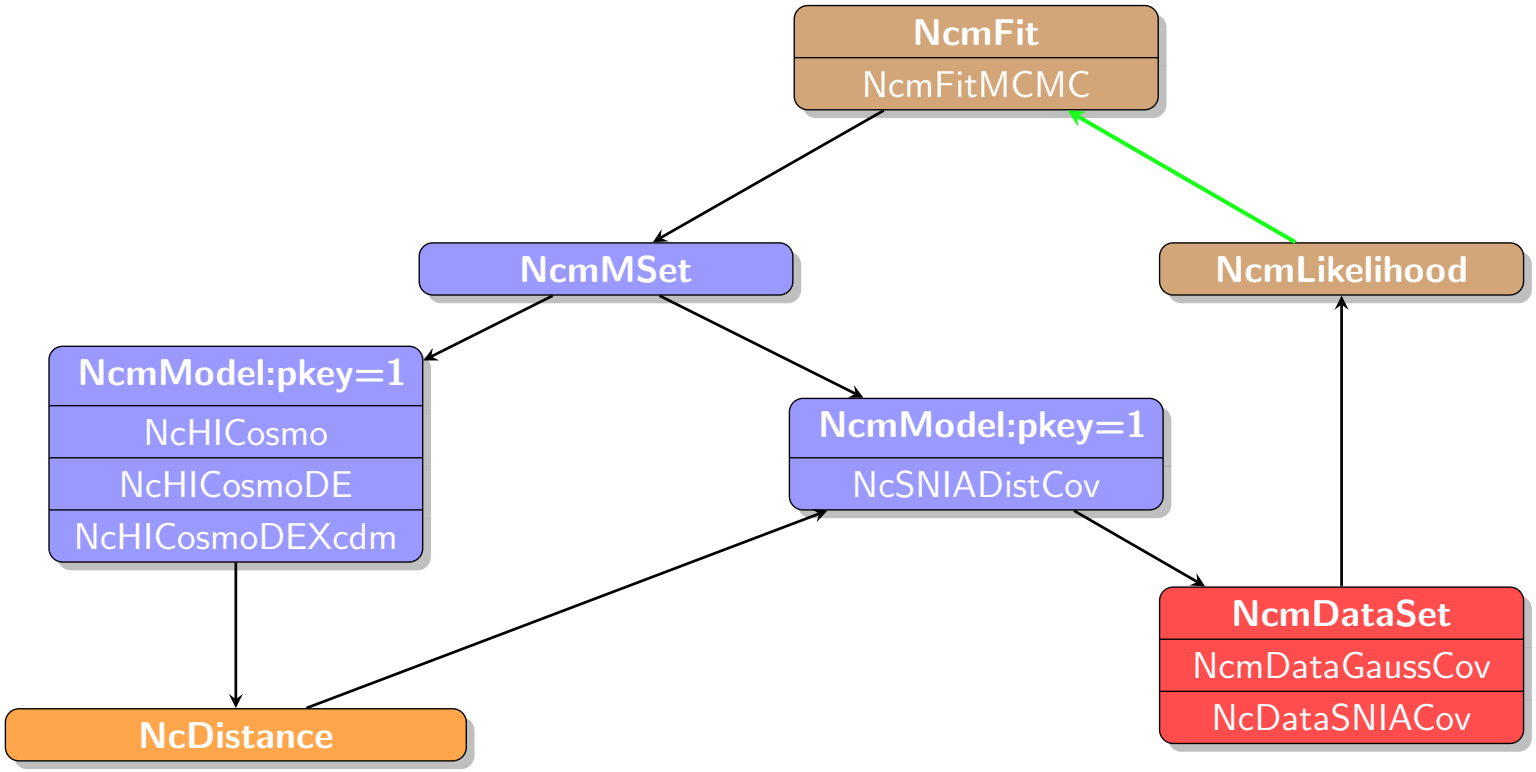


$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T C_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\rm hel}, z_i^{\rm cmb})) + \alpha X_i - \beta \mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

α and β are related to the stretch-luminosity and colour-luminosity, respectively, and M_{h_i} are absolute magnitudes.

Example 1: Type Ia Supernova – Update all

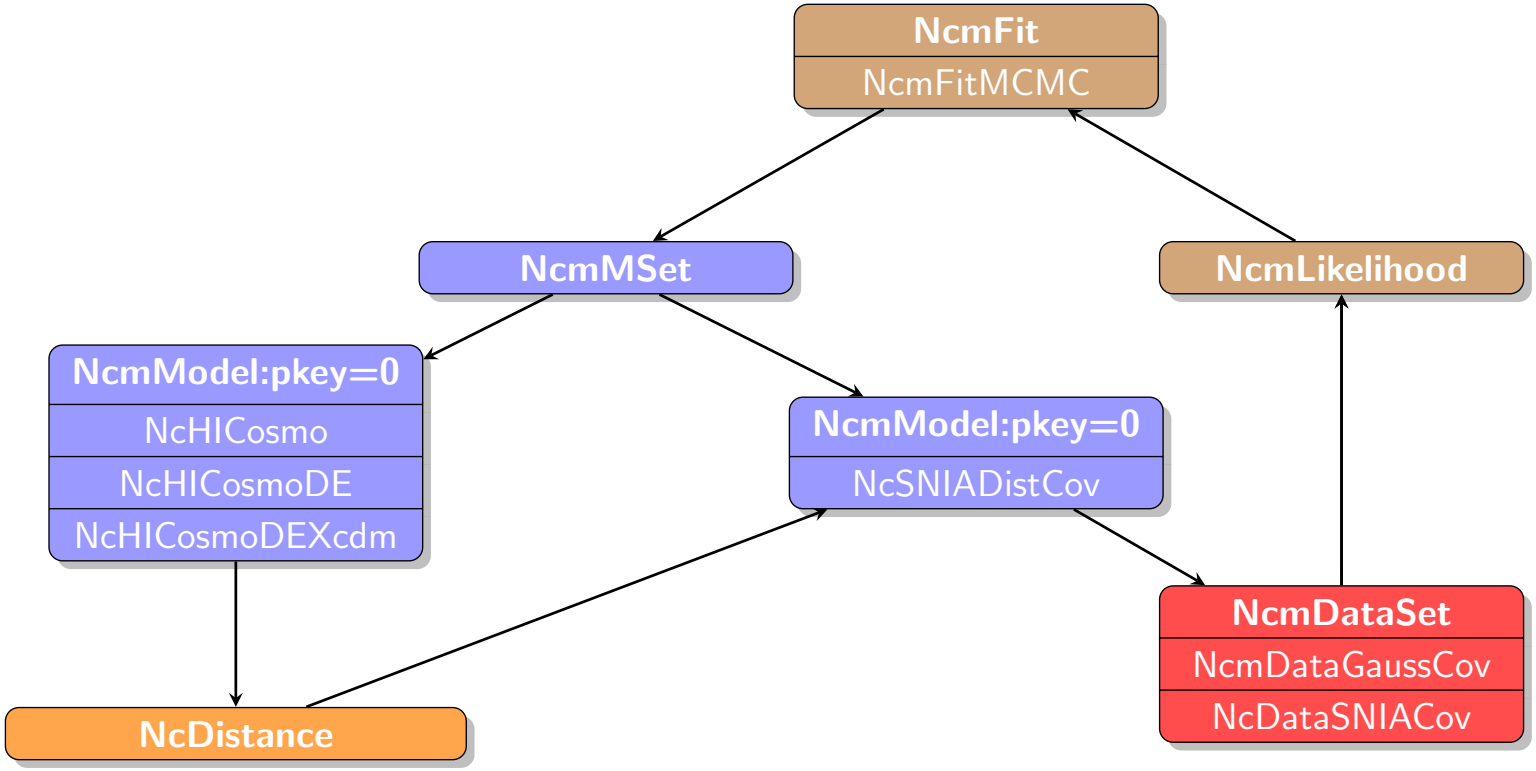


$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T C_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\rm hel}, z_i^{\rm cmb})) + \alpha X_i - \beta\mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

α and β are related to the stretch-luminosity and colour-luminosity, respectively, and M_{h_i} are absolute magnitudes.

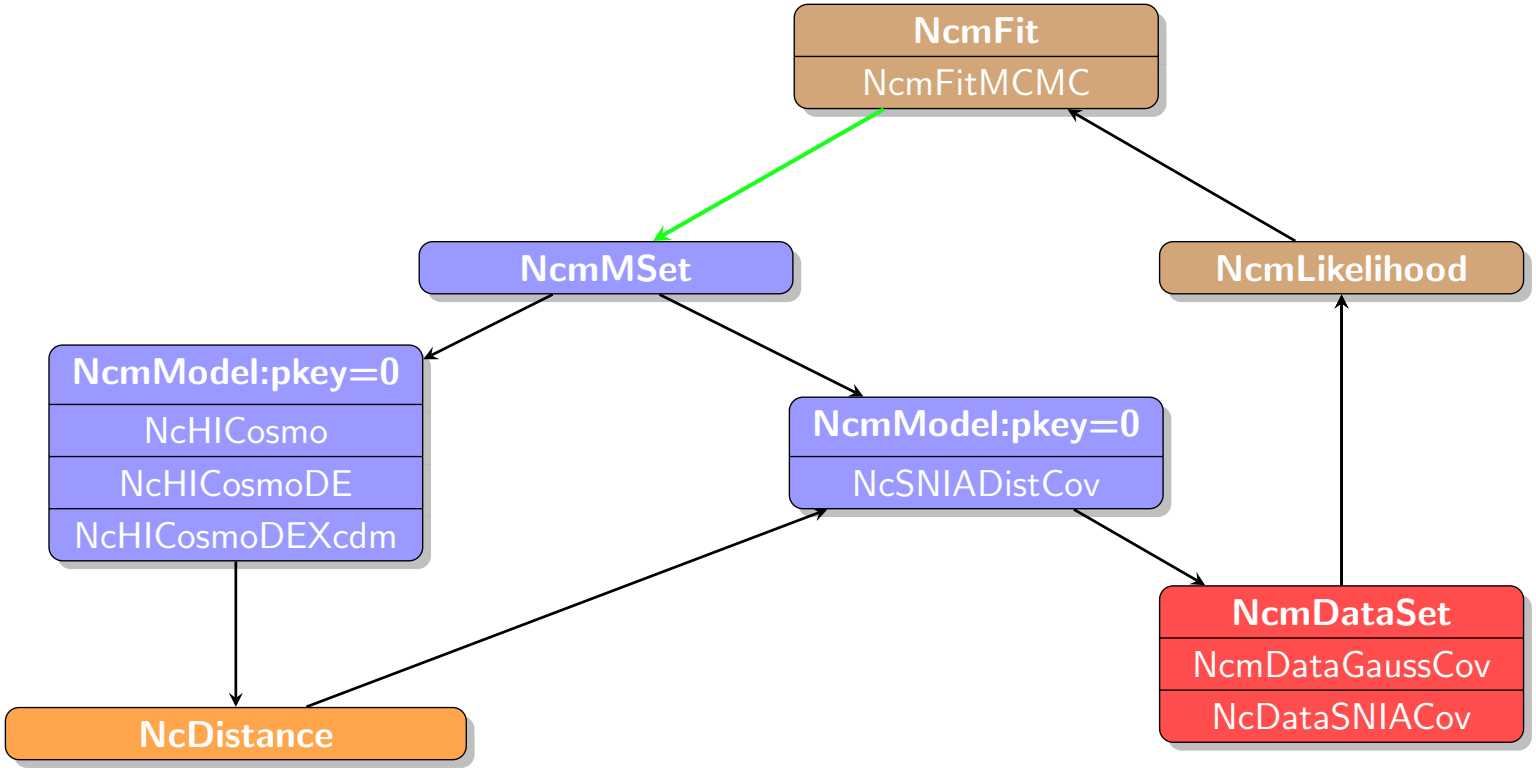
Example 2: Type Ia Supernova – Update cosmo only



$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T \mathsf{C}_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\rm hel}, z_i^{\rm cmb})) + \alpha X_i - \beta\mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

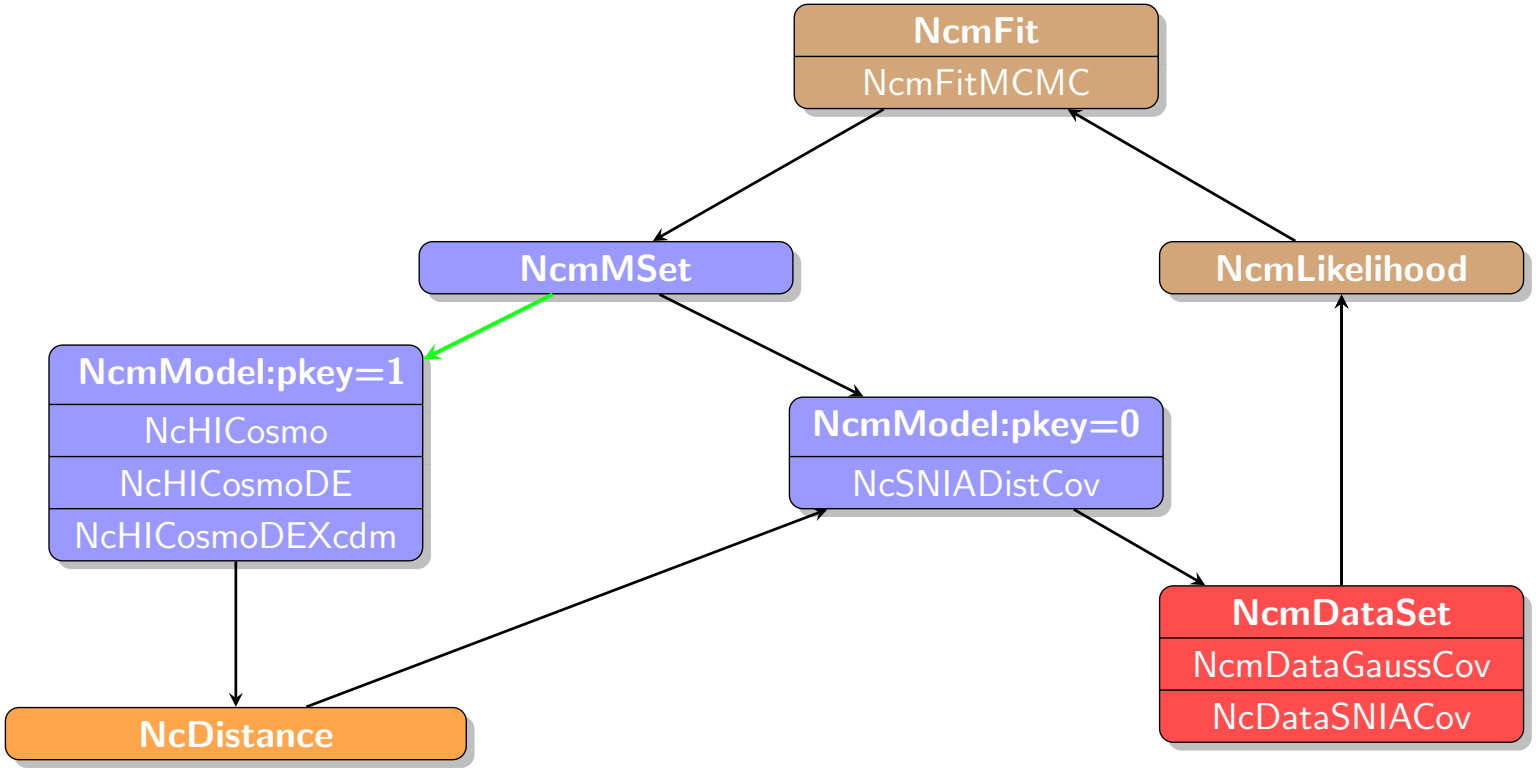
Example 2: Type Ia Supernova – Update cosmo only



$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T \mathrm{C}_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\mathrm{hel}}, z_i^{\mathrm{cmb}})) + \alpha X_i - \beta\mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

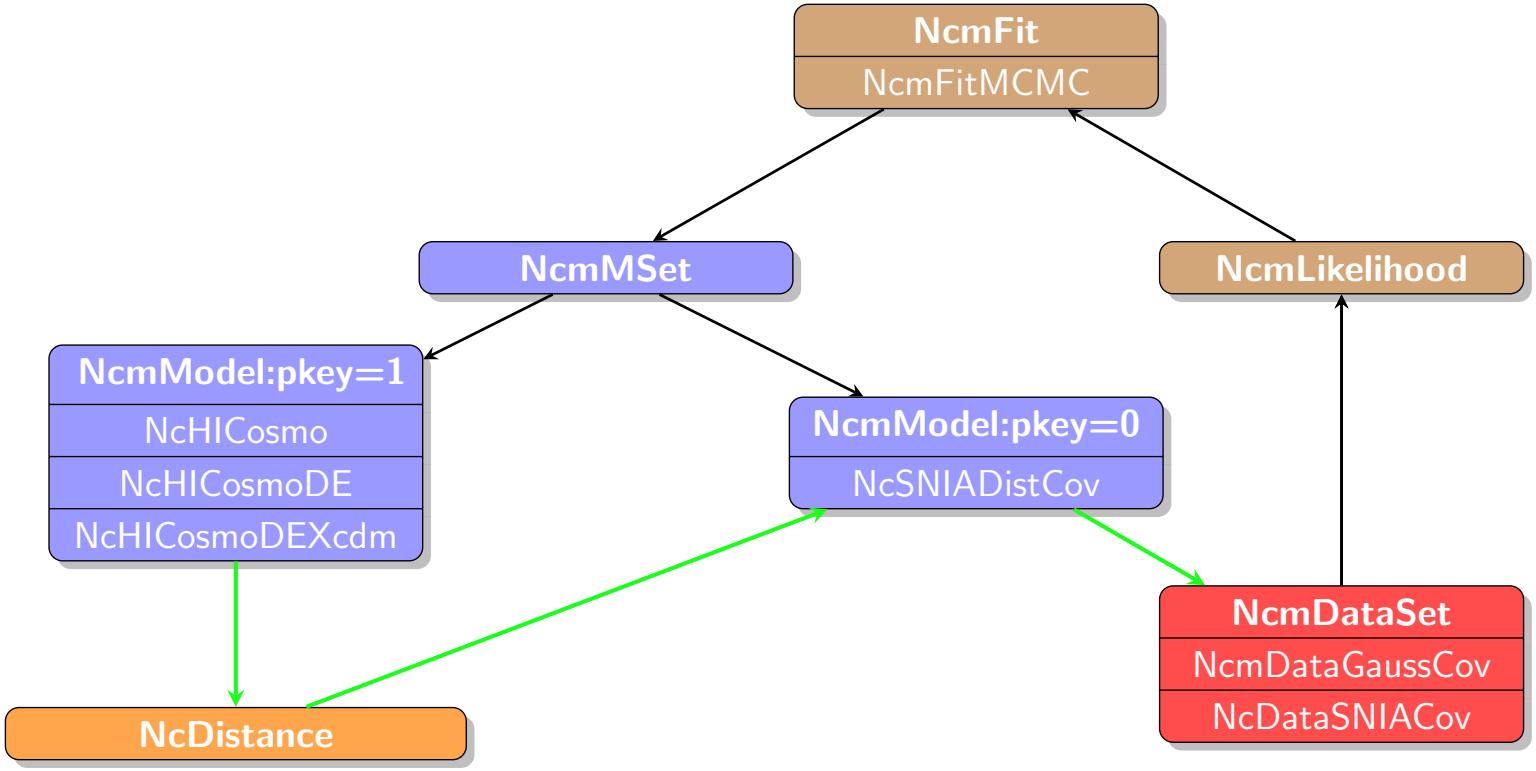
Example 2: Type Ia Supernova – Update cosmo only



$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T \mathrm{C}_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\mathrm{hel}}, z_i^{\mathrm{cmb}})) + \alpha X_i - \beta\mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

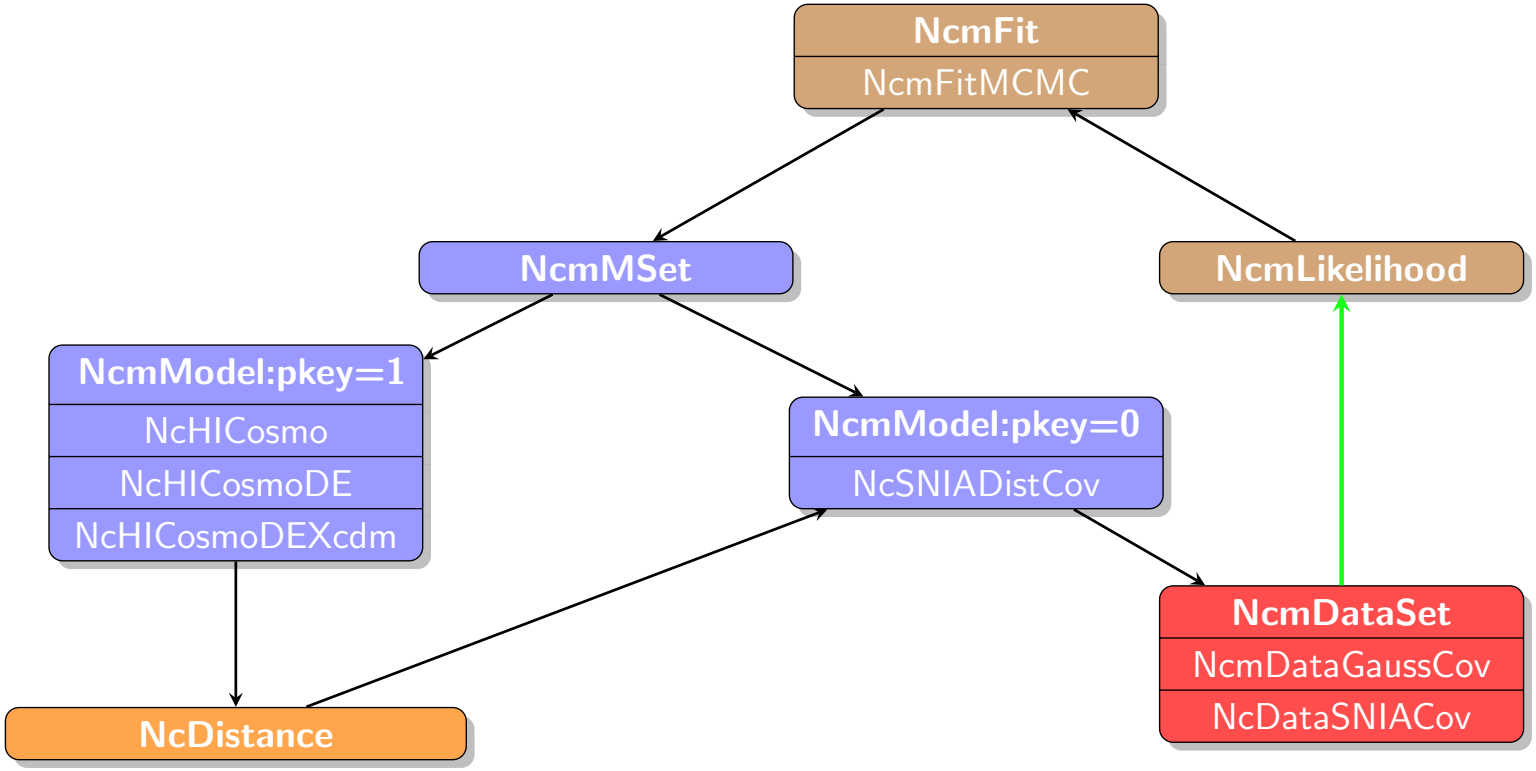
Example 2: Type Ia Supernova – Update cosmo only



$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T \mathsf{C}_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\rm hel}, z_i^{\rm cmb})) + \alpha X_i - \beta\mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

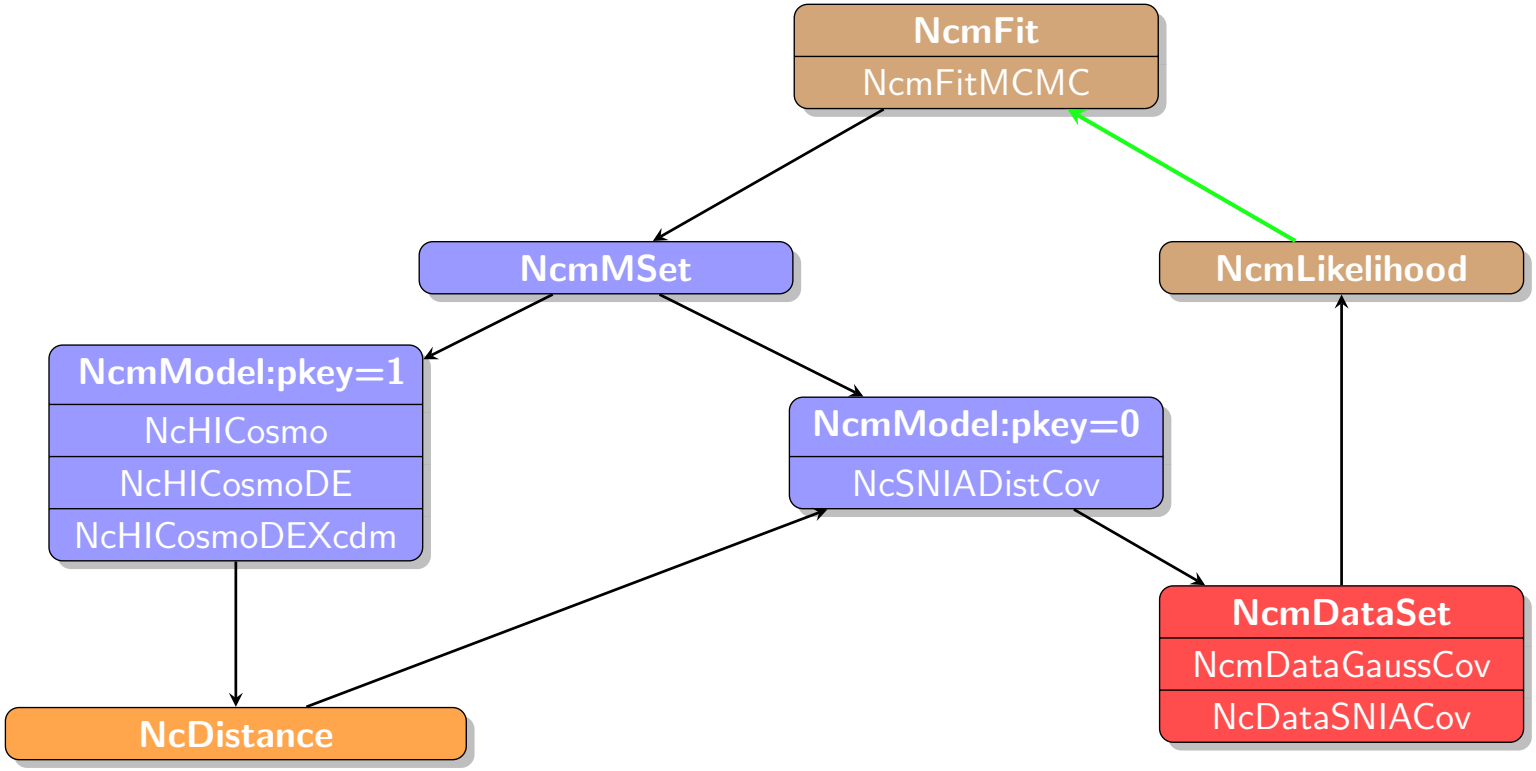
Example 2: Type Ia Supernova – Update cosmo only



$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T \mathsf{C}_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\rm hel}, z_i^{\rm cmb})) + \alpha X_i - \beta\mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

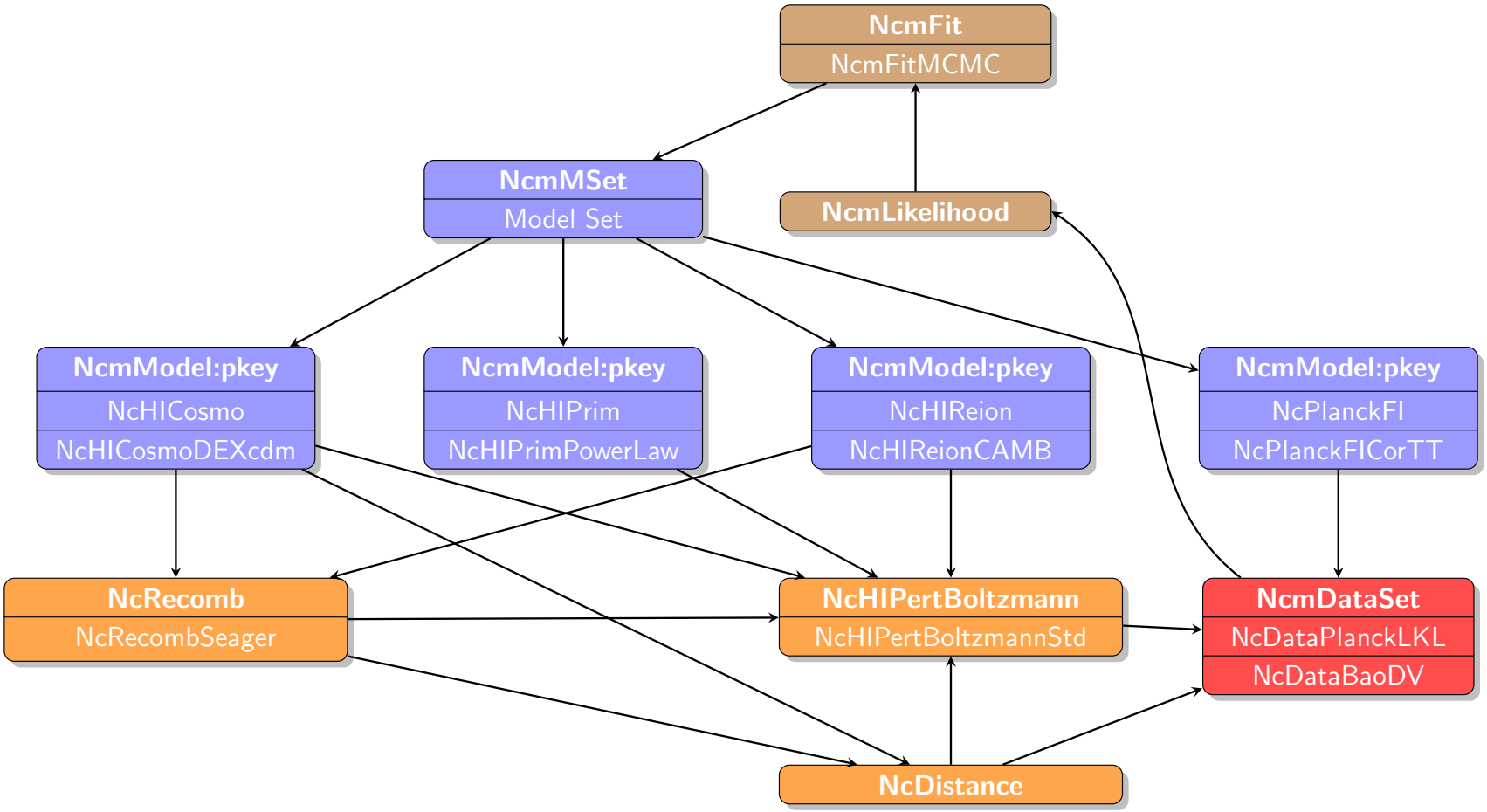
Example 2: Type Ia Supernova – Update cosmo only



$$-2\ln(L_{SNIa}) = \Delta\vec{m}^T \mathsf{C}_{SNIa}^{-1}(\alpha,\beta)\Delta\vec{m},$$

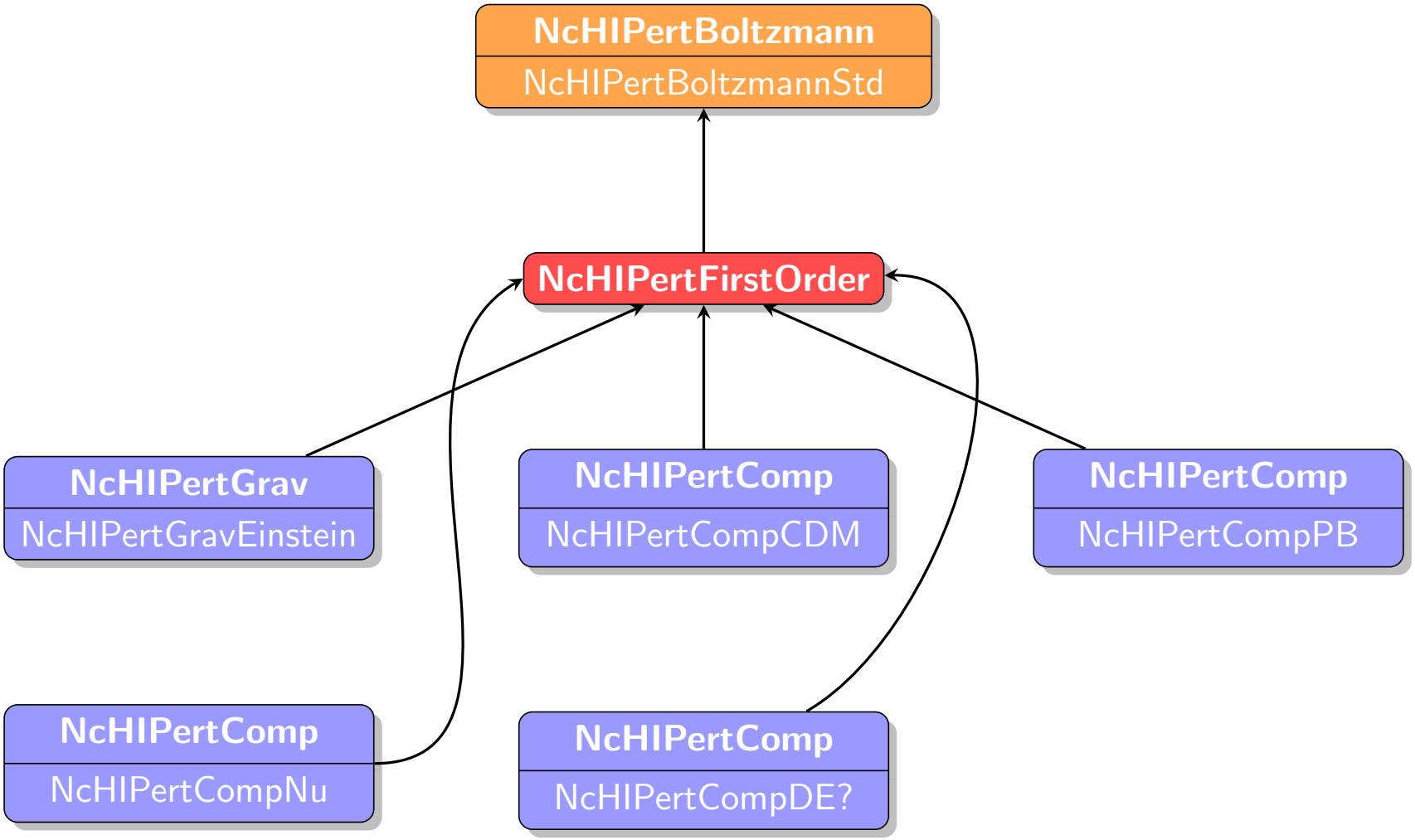
$$\Delta m_i = m_{Bi} - 5\log_{10}(\mathcal{D}_L(z_i^{\rm hel}, z_i^{\rm cmb})) + \alpha X_i - \beta\mathcal{C}_i - M_{h_i} + 5\log_{10}(c/H_0) - 25,$$

Example 3: CMB TT only + BAO



- NcHIPrimPowerLaw
- NcHIReionCAMB
- NcPlanckFICorTT
- NcDistance
- NcRecomb
- NcHIPertBoltzmann

A zoom at NcHIPertBoltzmannStd



Software Quality Assurance

- Unit testing: Test file for each object to assure the correct behavior of each part of NumCosmo.
- `NcmC` – consistent set of fundamental physical and mathematical constants: CODATA (2014), IAU (2015), NIST.
- `NcmSplineFunc` – Automatic determination of the spline knots, given a precision.
- `Autotools` – Widely used build-system, support several operational systems and architectures and compilers.
- Packages for different package managers, conda-forge, Ubuntu, OpenSUSE and others.
- Continuous integration support through travis-ci https://travis-ci.org/NumCosmo/NumCosmo ; each commit is compiled and tested automatically in both Linux and MacOS.
- Not reinventing the wheel – Uses well known and tested libraries as back-end:
  - GLib (object system/portability), Cuba (multidimensional integration), NLopt (non-linear optimization), GSL (several scientific algorithms), cfitsio (fits file manipulation), Sundials (ODE integrator), HDF5 and others.

CMB anisotropies status:

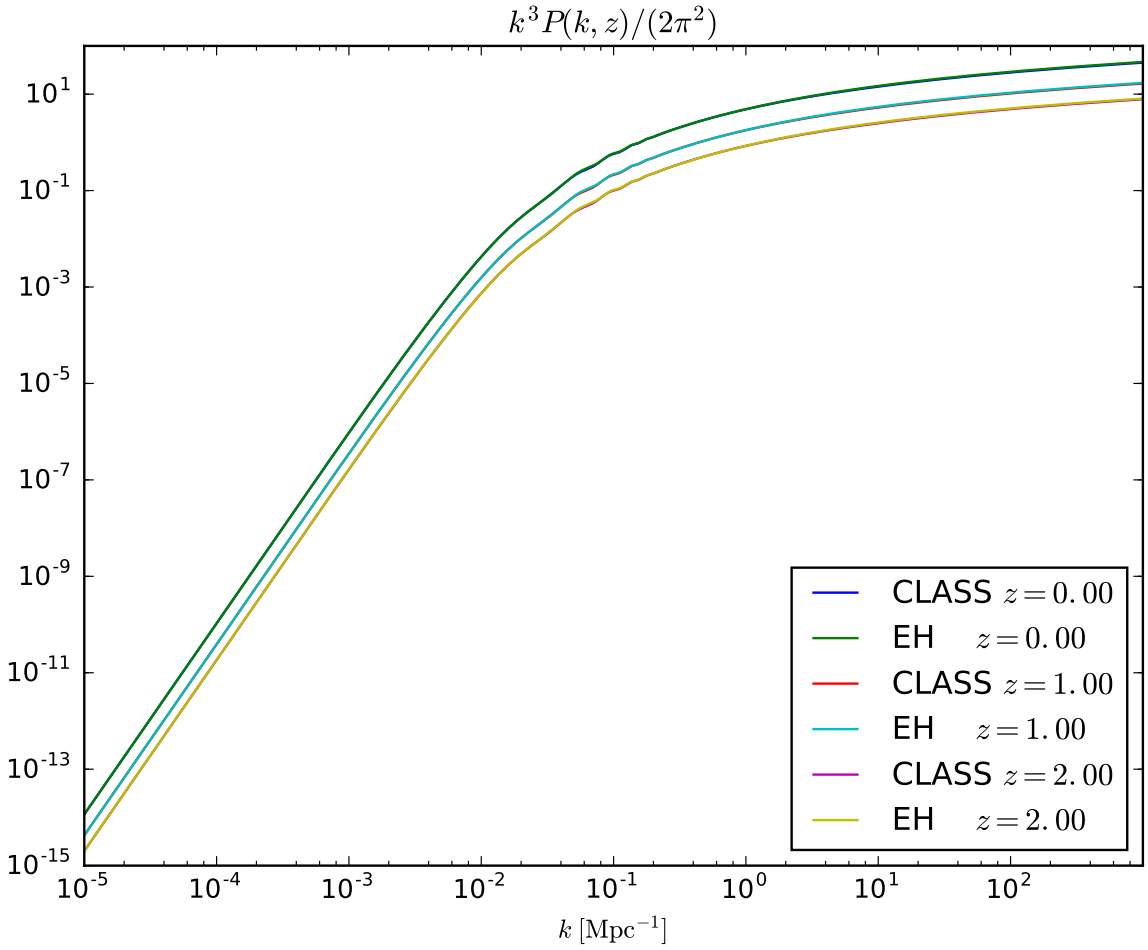
- CLASS and the Planck's likelihood (clik) are integrated in the library building system.
- Abstract Planck foreground and instrument models are defined by `NcPlanckFI`, `NcPlanckFICorTT` and `NcPlanckFICorTTTEEE`.
- CLASS interface is described in the CLASS Backend object `NcCBE`.
- Interface to C_l through `NcHIPertBoltzmann` and Planck's likelihood through `NcDataPlanckLKL`.
- Currently fully working implementation of `NcHIPertBoltzmann`: CLASS based `NcHIPertBoltzmannCBE`.
- Standard NumCosmo implementation `NcHIPertBoltzmannStd`, work in progress, support for seeds and GI implementation.

CMB anisotropies status:

- Alternative primordial power spectra interface:
  - `NcHIPrimAtan`: implements an inverse tangent modification of the power spectrum.
  - `NcHIPrimExpc`: implements exponential cutoff power spectrum.
  - `NcHIPrimBPL`: implements broken power law power spectrum.
- Thermodynamics through CLASS `NcRecombCBE`, NumCosmo implementation `NcRecombSeager` (provides a single pass integration for the whole system, same equations as RECFAST).
- Linear matter power spectrum `NcPowspecML`, interface to CLASS Matter Transfer Function `NcPowspecMLCBE`
- Matter power spectrum non-linear corrections `NcPowspecMNL` through Halofit `NcPowspecMNLHalofit` (Halo model in a future next release).

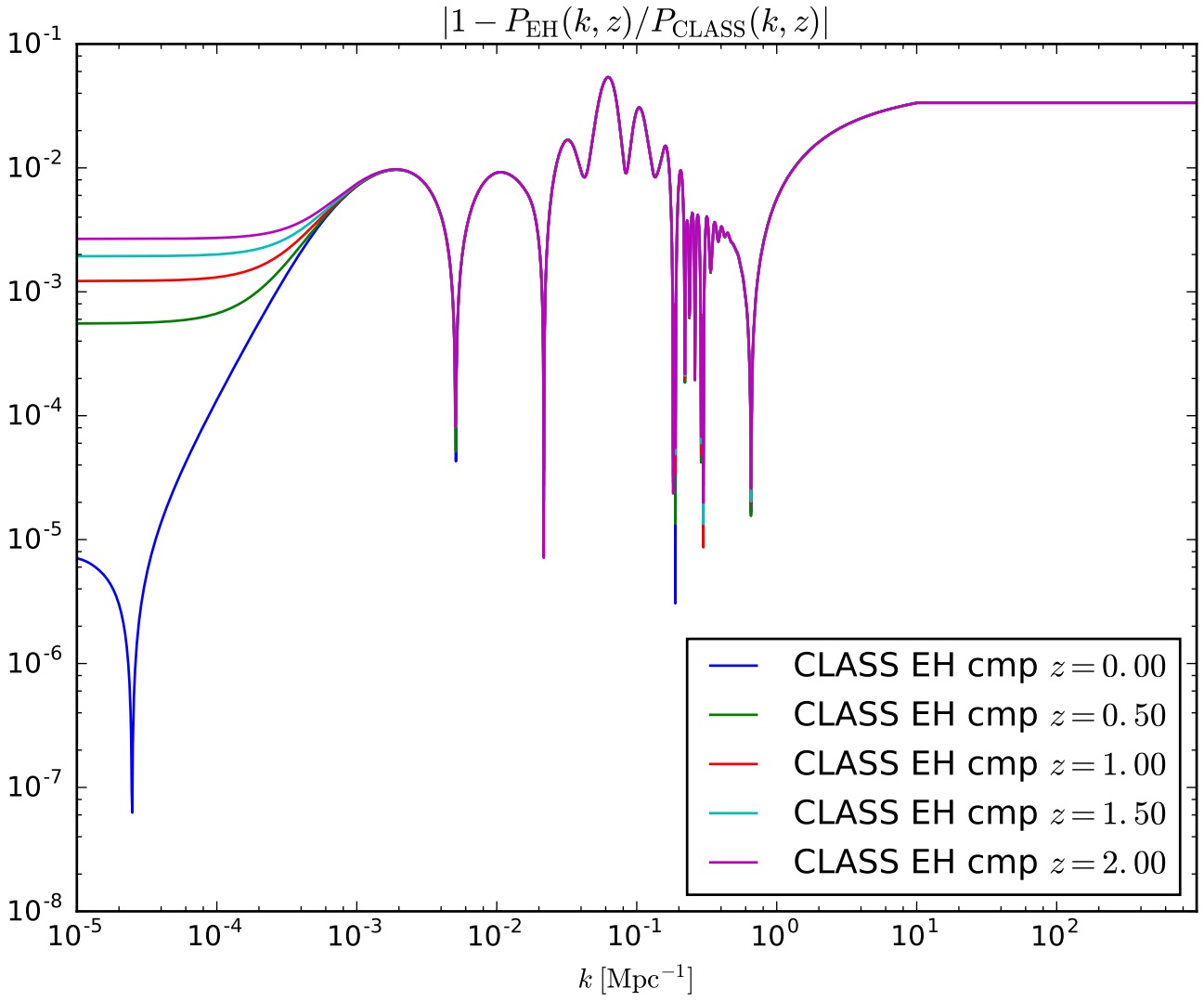
Matter power spectrum products

- `NcPowspecMLCBE`: CLASS linear matter power spectrum;
- `NcPowspecMLTransfer` + `NcTransferFuncEH` + `NcGrowthFunc`: Eisenstein and Hu (EH);



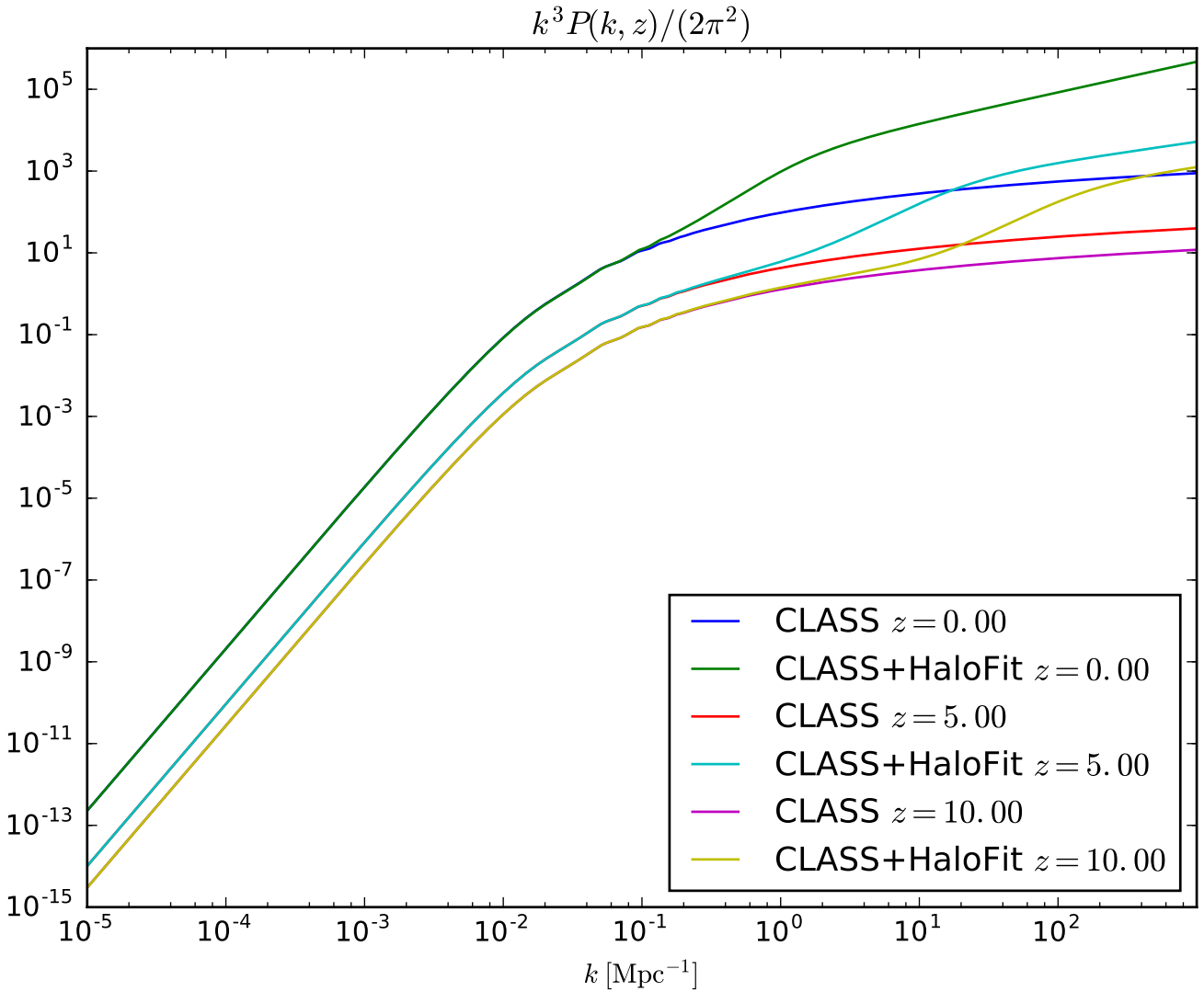
Matter power spectrum products

- CLASS vs EH linear matter power spectrum;



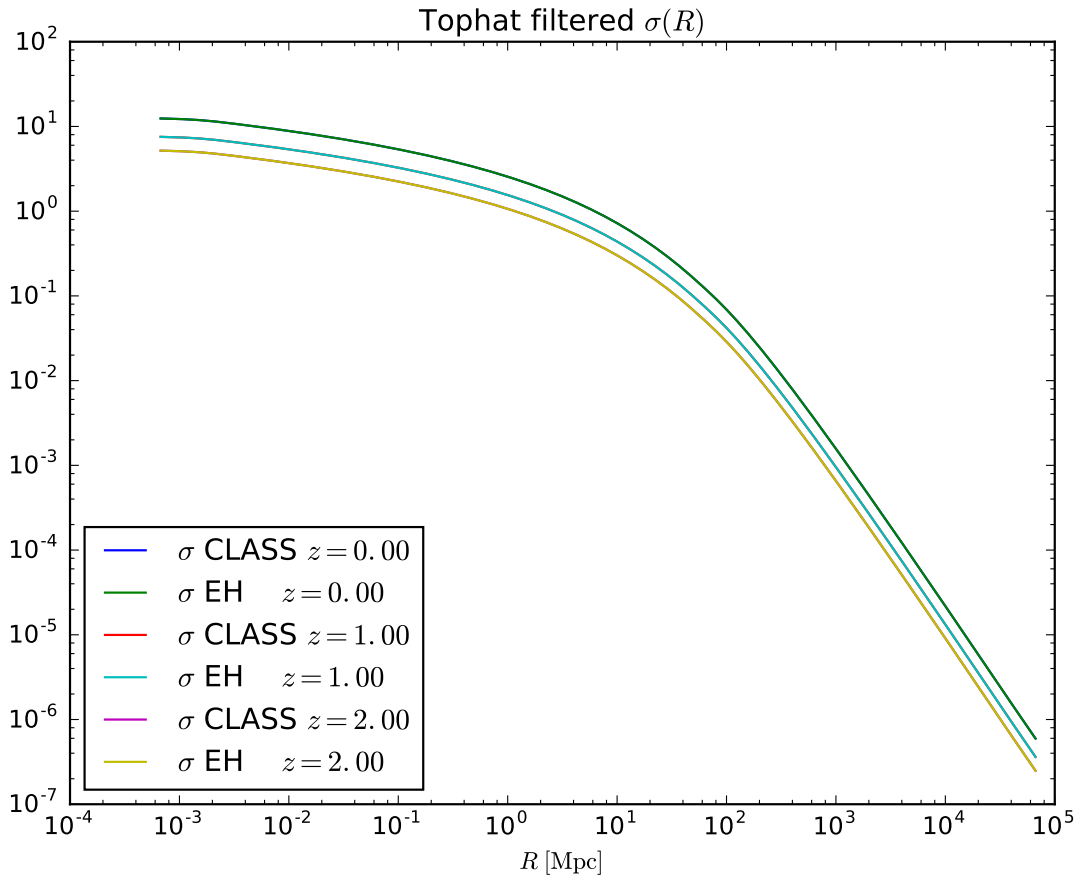
Matter power spectrum products

- CLASS + `NcPowspecMNLHaloFit`;



Matter power spectrum products

- `NcmPowspecFilter` + CLASS: $\sigma_8 = 0.846$;
- `NcmPowspecFilter` + EH: $\sigma_8 = 0.853$;
- `NcmPowspecFilter` uses `NcmFftlog` to compute $\int_0^\infty j_l(kR)^2 P(k,z) k^2 \mathrm{d}k$ efficiently.



Cosmology calculators:

Standard codes: CAMB/CLASS

NumCosmo

- `NcHICosmo`:
  - `NcHICosmoDE*` (several);
  - ⋆ `NcHICosmoQSpline` (kinematic);
  - ...
- `NcHIPrim` (primordial):
  - `NcHIPrimPowerLaw`;
  - ⋆ `NcHIPrimExpc` (exponential cut-off);
  - ...
- `NcHIReion` (reionization):
  - `NcHIReionCAMB`;
- `NcDistance`
- `NcRecomb` (recombination):
  - `NcRecombCBE` (CLASS backend);
  - ⋆ `NcRecombSeager` (recfast);

Cosmology calculators:

Standard codes: CAMB/CLASS

NumCosmo

- `NcHIPertBoltzmann`:
  - `NcHIPertBoltzmannCBE` (CLASS backend);
  - ⋆ `NcHIPertBoltzmannStd` (in progress);
- `NcPowspecML` (linear matter ps):
  - `NcPowspecMLCBE` (CLASS backend);
  - `NcPowspecMLTransfer`;
  - `NcPowspecMLPert`;
- `NcPowspecMNL` (non-linear power-spectrum):
  - `NcPowspecMNLHaloFit`;
  - `NcPowspecMNLHaloModel`;

Observational Probes:

Standard codes: CosmoMC/MontePython

NumCosmo

- NcDataSNIACov: NcmDataGaussCov;
- NcDataDistMu: NcmDataGaussDiag;
- NcDataClusterNCount: NcmData;
- NcDataClusterPoisson: NcmDataPoisson;
- ⋆ NcDataClusterPseudoCounts: NcmData;
- NcDataBao*: NcmDataGauss*;
- NcDataHubble: NcmDataGaussDiag;
- NcDataCMBDistPriors: NcmDataGauss;
- NcDataPlanckLKL: NcmData;
- ⋆ NcDataXcor: NcmDataGaussCov
  - ⋆ NcXcorLimberGal: NcXcorLimber
  - ⋆ NcXcorLimberLensing: NcXcorLimber

Data Analysis Tools:

Standard codes: CosmoMC/MontePython

NumCosmo

- Bayesian Analysis:
  - `NcmFitMCMC` – Markov Chain Monte Carlo (Metropolis-Hastings): `NcmMSetTransKern`.
  - `NcmFitESMCMC` – Ensemble sampler Markov Chain Monte Carlo: `NcmFitESMCMCWalker`.
    - `NcmLFitESMCMCWalkerStretch` – Stretch move as in emcee.
      - ⋆ `NcmLFitESMCMCWalkerAPS` – Approximate posterior sampling (work in progress).
  - ⋆ `NcmMSetCatalog` – (ES)MCMC output catalog, includes support for Bayesian evidence and posterior volume computation directly from the (ES)MCMC output.
  - ⋆ `NcmABC` – Approximate Bayesian Computation (ABC).
- Frequentist Analysis:
  - ⋆ `NcmLHRatio1d` – 1D Profile likelihood .
  - ⋆ `NcmLHRatio2d` – 2D Profile likelihood.

Data Analysis Tools:

Standard codes: CosmoMC/MontePython

NumCosmo

- ⋆ Best-fit finder: `NcmFitGSLLS`, `NcmFitGSLMM`, `NcmFitGSLMMS` and `NcmFitNLOpt`.
- ⋆ `NcmFit` – Observed Fisher Matrix [ncm_fit_obs_fisher] using `NcmDiff`.
- ⋆ `NcmFit` – Expected Fisher Matrix [ncm_fit_fisher] using `NcmDiff`, needs direct support in the `NcmData`, all `NcmDataGauss*` objects already includes support.
- ⋆ `NcmFitMC` – Monte Carlo analysis (resample/bootstrap from a probability density function).
- ⋆ `NcmFitMCBS` – Monte Carlo and bootstrap analysis.

ESMCMC example

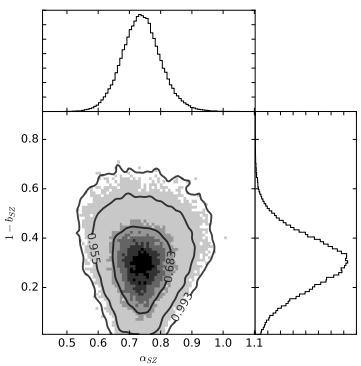
Snapshot of a ESMCMC run using Planck TT data and a modified primordial power spectrum.

- Object: `NcmFitESMCMC`,
- Walker: `NcmFitESMCMCWalkerStretch`,
- Output: a `NcmMSetCatalog` object serialized in a fits file.

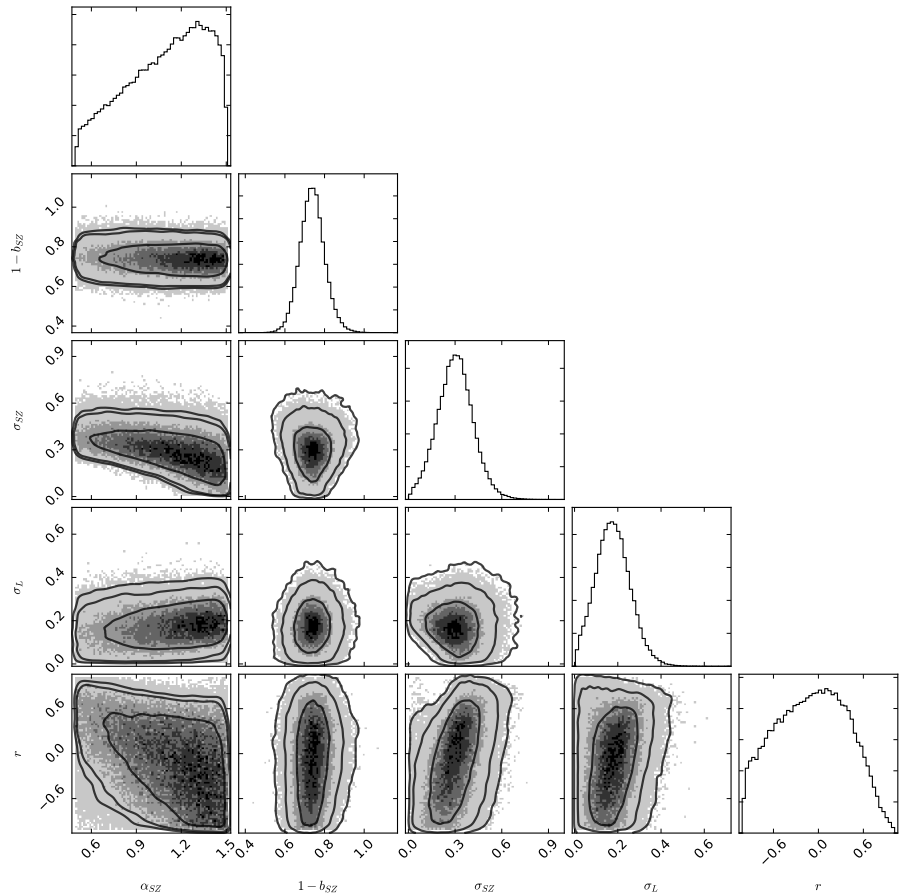
```
#  Elapsed time: 01 days, 21:57:54.8427910
#  degrees of freedom   [002523]
#  m2lnL     =      796.42768789029
#  Fit parameters:
#     69.8108499055435     0.115986617072077    0.0226852361053738   3.23602243188527
# NcmMSetCatalog: Current mean:   854.15       69.938       0.11588      0.022799
# NcmMSetCatalog: Current msd:    0.54997      0.0049737    1.2071e-05   1.8152e-06
# NcmMSetCatalog: Current sd:     511.73       0.90615      0.0020219    0.00036516
# NcmMSetCatalog: Current var:    2.6187e+05   0.82111      4.0882e-06   1.3334e-07
# NcmMSetCatalog: Current tau:    1            26.084       30.858       21.395
# NcmMSetCatalog: Current skfac:  1.0029       1.1474       1.1388       1.0652
# NcmMSetCatalog: Maximal  Shrink factor =     1.25590891624521
# NcmFitESMCMC:acceptance ratio 62.5521%, offboard ratio  0.0000%.
# Task:NcmFitESMCMC, completed: 865800 of 10371900, elapsed time: 1 day, 21:57:54.816
# Task:NcmFitESMCMC, mean time: 00:00:00.1911 +/- 00:00:00.0036
# Task:NcmFitESMCMC, time left: 21 days, 00:40:40.0570 +/- 09:22:42.2361
# Task:NcmFitESMCMC, estimated to end at: Tue May 17 2016, 07:18:32 +/- 09:22:42.2361
```

Data Analysis Tools:

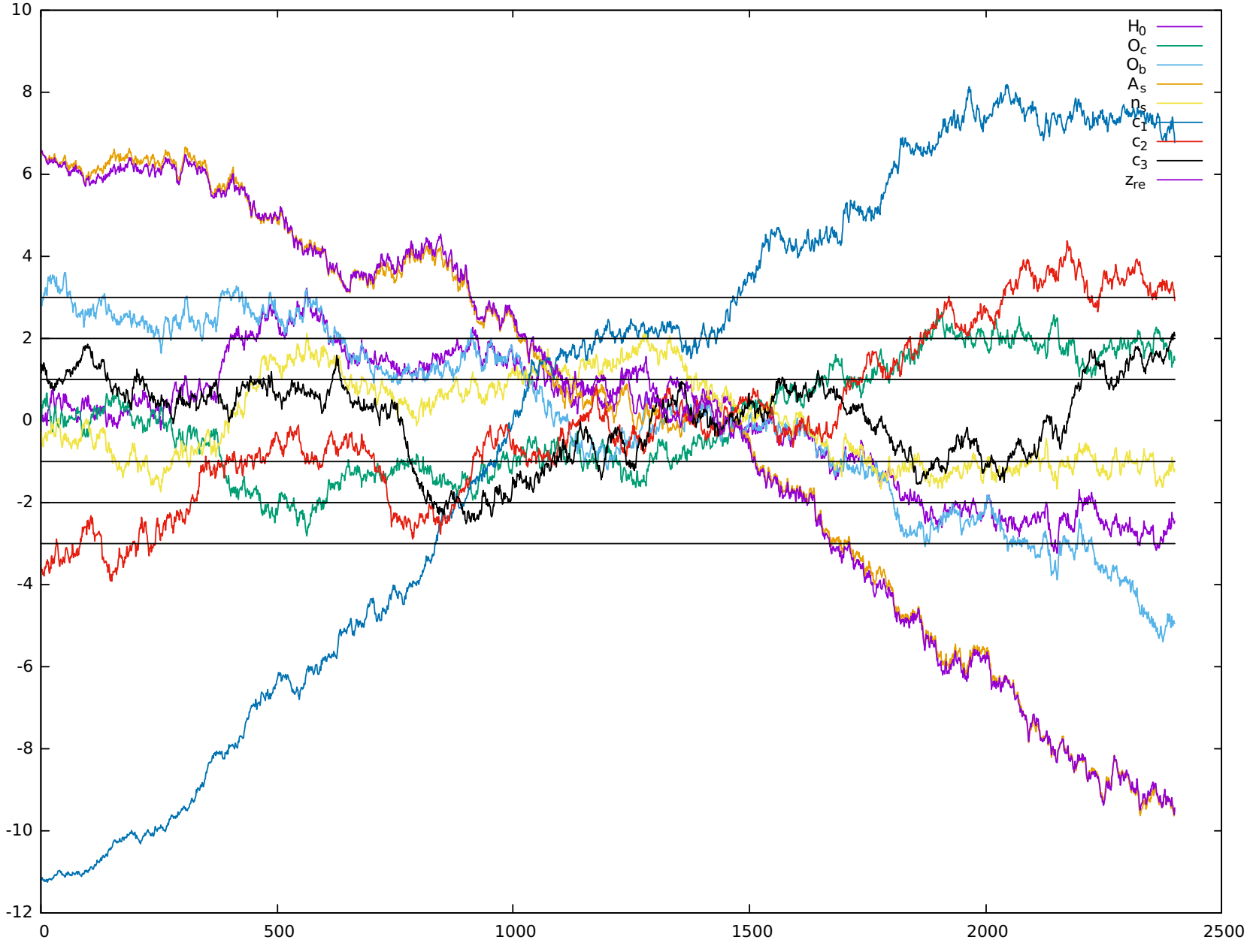
- mcat_analyze:
  - Compute $1-3\sigma$ confidence intervals for the best-fit, mean, mode or median of marginal distributions.
  - Compute covariance matrix.
- de_mc_plot.py: make plot of histograms of marginal parameter distributions and the bidimensional contours.



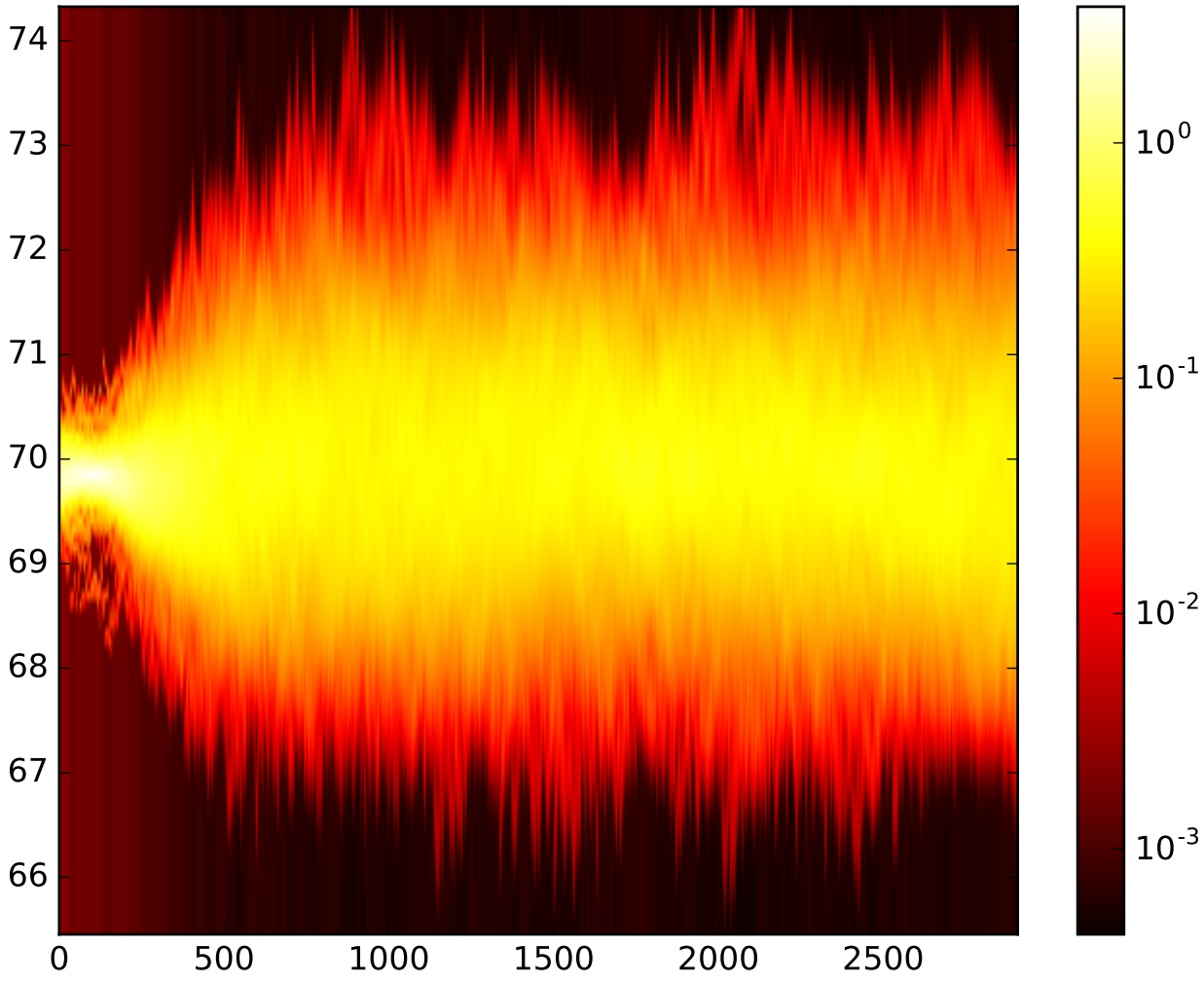
- de_mc_corner_plot.py: make corner plot.



Example: Chains evolution



Example: Parameter distribution evolution



Thank you!

Questions?