STATISTICAL CHALLENGES IN MODERN ASTRONOMY VI

Approximate Bayesian computation: an application to weak-lensing peak counts

Chieh-An Lin & Martin Kilbinger

SAp, CEA Saclay

Carnegie Mellon University, Pittsburgh — June 6th, 2016





2 Approximate Bayesian computation



Chieh-An Lin (CEA Saclay)

Weak gravitational lensing in 3 minutes



Source: ALMA

Weak gravitational lensing in 3 minutes



Chieh-An Lin (CEA Saclay)

Weak gravitational lensing in 3 minutes



Weak gravitational lensing in 3 minutes



 κ map



Weak-lensing peak counts

 κ map and peaks





- · Local maxima of the projected mass
- Probe the mass function
- Non-Gaussian information

Dealing with selection function

Early studies Count only the true clusters with high S/N

Recent studies

Include the selection effect into the model

- Analytical formalism
- N-body simulations
- Fast stochastic model (this work)

A new model to predict weak-lensing peak counts



Public code in C: Camelus@GitHub

See also Lin & Kilbinger (2015a)

Chieh-An Lin (CEA Saclay)

Validation



Approximate Bayesian computation



Approximate Bayesian computation

Requirements:

- Stochastic model $P(\cdot | \boldsymbol{\pi})$
- Distance |x x'|
- Tolerance level ϵ



Approximate Bayesian computation

Accept-reject process:

- Draw a π from the prior $\mathcal{P}(\,\cdot\,)$
- Draw a $m{x}$ from the model $P(\,\cdot\,|m{\pi})$
- Accept $\boldsymbol{\pi}$ if $|\boldsymbol{x} \boldsymbol{x}^{\mathrm{obs}}| \leq \epsilon$
- Reject otherwise
- \Rightarrow One-sample test



Chieh-An Lin (CEA Saclay)

Approximate Bayesian computation



Distribution of accepted π = prior × green areas

- \approx prior $\times 2\epsilon \times$ likelihood
- \propto posterior

Combined with population Monte Carlo

Population Monte Carlo (PMC)

- Iterative solution for ϵ
- Set $\epsilon = +\infty$ for the first iteration
- Do ABC
- Update ϵ and the prior based on results from the previous iteration
- Repeat until satifying a stop criterion

See also Lin & Kilbinger (2015b)

Comparison with the likelihood



Contours and dots: ABC Colored regions: likelihood

Gain of time cost for ABC: \approx 2 orders of magnitude

Lin & Kilbinger (2015b)

Data from three surveys







Survey	Field size	Number of	Effective density
	[deg ²]	galaxies	$[deg^{-2}]$
CFHTLenS	126	6.1 M	10.74
KiDS DR1/2	75	2.4 M	5.33
DES SV	138	3.3 M	6.65

Model settings

• Compensated filter (suggested by Lin et al. 2016)

linc.tw

- Adaptive choice for pixel sizes and filtering scales
- No intrinsic alignment and baryons (not yet!)

ABC settings

- Data vector (summary statistic): peak counts with S/N > 2.5 of all scales
- Distance: a χ^2 -like normalized sum

Preliminary result



Lin & Kilbinger in prep.

Preliminary result



Width: $\Delta \Sigma_8 = 0.13$ Area: FoM = 5.2

Preliminary result



Lin & Kilbinger in prep.

Ongoing improvements and perspectives

- S/N bin choice: less bias, more accuracy and precision
- Halo correlation
- Tomography
- Intrinsic alignment
- Baryonic effects

- A new model to predict WLPC
- Likelihood-free parameter inference: ABC
- Constraints with CFHTLenS, KiDS, DES





Collaborators: Martin Kilbinger Austin Peel (poster talk on Wednesday) Sandrine Pires

References: [1410.6955] [1506.01076] [1603.06773] Camelus@GitHub http://linc.tw

Backup slides

Fast

Only few seconds for creating a 25-deg 2 field, without MPI or GPU programming

Flexible

Straightforward to include real-world effects (photo-z errors, masks, intrinsic alignment, baryonic effects, etc.)

Full PDF information

Allow more flexible constraint methods (varying covariances, copula, *p*-value, approximate Bayesian computation, etc.)

Map examples



Chieh-An Lin (CEA Saclay)

Cosmology-dependent covariance



True likelihood



Degeneracy with w_0^{de}



ABC: an application to weak-lensing peak counts

Technical detail

- Mass function from Jenkins et al. (2001)
- *M*-*c* relation from Takada & Jain (2002)
- Source redshift fitted from surveys
- Random source position, not catalogue
- Used raw galaxy densities, not effective
- Derive σ_{ϵ} from the emperical total variance
- Pixel size: $n_{\rm gal} \theta_{\rm pix}^2 \ge 7$
- Kaiser-Squires inversion
- Filtering with the "starlet" function
- Scale = 2, 4, 8 pixels
- Locally determined noise
- Dimension of *x* = 75 (= 36 + 15 + 24)

Field examples



ABC: an application to weak-lensing peak counts



Lin & Kilbinger in prep.

Constrain concentration paramters



ABC: an application to weak-lensing peak counts

Definition of Σ_8



 $\begin{array}{ll} \mbox{Definition 1} & \Sigma_8 = \sigma_8 (\Omega_{\rm m}/0.27)^{\alpha} \\ \\ \Sigma_8 = 0.825 & \Delta \Sigma_8 = 0.16 \\ \end{array} \qquad \qquad \alpha = 0.48 \end{tabular}$

Definition 2
$$\Sigma_8 = \left(\frac{\Omega_m + \beta}{1 - \alpha}\right)^{1 - \alpha} \left(\frac{\sigma_8}{\alpha}\right)^{\alpha}$$

 $\Sigma_8 = 1.935 \qquad \Delta \Sigma_8 = 0.13 \qquad \alpha = 0.38 \qquad \beta = 0.82$

PMC ABC algorithm

set
$$t = 0$$

for $i = 1$ to Q do
generate $\theta_i^{(0)}$ from $\mathcal{P}(\cdot)$ and x from $P\left(\cdot | \theta_i^{(0)}\right)$
set $\delta_i^{(0)} = D\left(x, x^{\text{obs}}\right)$ and $w_i^{(0)} = 1/Q$

end for

set $\epsilon^{(1)} = \text{median}\left(\delta^{(0)}_i\right)$ and $C^{(0)} = \text{cov}\left(\pi^{(0)}_i, w^{(0)}_i\right)$

while success rate $\geq r_{\rm stop}\;{\rm do}$

$$t \leftarrow t + 1$$

for i = 1 to Q do

repeat

generate
$$j$$
 from $\{1, \ldots, Q\}$ with weights $\left\{w_1^{(t-1)}, \ldots, w_Q^{(t-1)}\right\}$
generate $\pi_i^{(t)}$ from $\mathcal{N}\left(\pi_j^{(t-1)}, \mathbf{C}^{(t-1)}\right)$ and x from $P\left(\cdot | \pi_i^{(t)}\right)$
set $\delta_i^{(t)} = D\left(x, x^{\text{obs}}\right)$
until $\delta_i^{(t)} \leq \epsilon^{(t)}$
set $w_i^{(t)} \propto \mathcal{P}\left(\pi_i^{(t)}\right) / \sum_{j=1}^Q w_j^{(t-1)} K\left(\pi_i^{(t)} - \pi_j^{(t-1)}, \mathbf{C}^{(t-1)}\right)$

end for

set
$$\epsilon^{(t+1)} = \mathrm{median}\left(\delta_i^{(t)}\right)$$
 and $C^{(t)} = \mathrm{cov}\left(\pi_i^{(t)}, w_i^{(t)}\right)$

end while

Peaks v.s. power spectrum



Fast weak-lensing peak counts modelling in C with PMC ABC



Camelus@GitHub