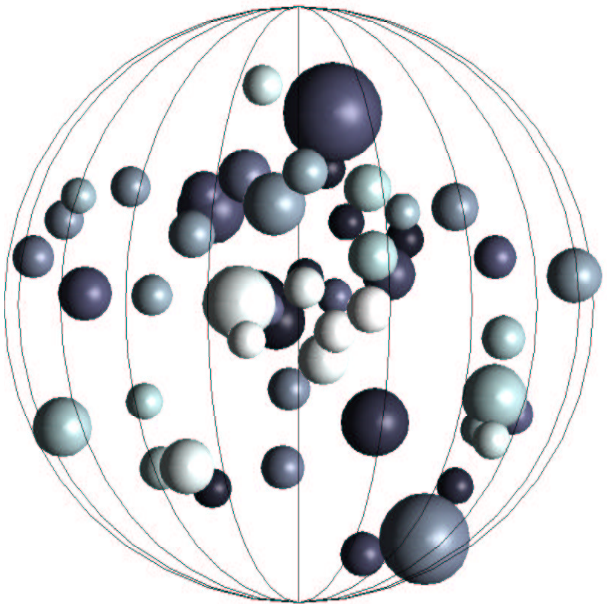
Examensarbete i Astronomi / Rymd- och Plasmafysik 20p
Januari 2005
Nr: 9

UPPSALA
UNIVERSITET

# The mass distribution along the line of sight to light sources at cosmological distances

Teresa Riehm



## Abstract

It is conventional to adopt the Press-Gunn approximation when carrying out microlensing investigations. This means that the lensing objects are assumed to be uniformly and randomly distributed throughout the universe with a comoving density. In order to test the applicability of this assumption, a code has been developed which is able to model more realistic mass distributions along the line of sight to a pointlike light source at arbitrary redshift. Therein the effects of the mass distribution of dark halos, the dark halo density profiles, halo substructure, halo-to-halo clustering and the fraction of mass clustered in halos are all taken into account. The obtained average matter density distributions showed substantial deviations from the predictions by the Press-Gunn approximation. While the existence of halo substructure is negligible, halo-halo clustering has a significant impact along the line of sight to light sources at more modest distance ($z < 1$). The fraction of mass clustered in halos turned out to be an important variable at any redshift, indicating a mass resolution problem. Considering variations to such an extend, the quality of the Press-Gunn approximation has to be assessed as poor and the implementation of more realistic matter distribution models into microlensing studies is advised.

# Contents

# Chapter 1

# Introduction

Our present comprehension of the dynamics of the universe on large scales suggests that most of the material in the universe is not luminous. Its gravitational impact on luminous matter is however observable. There is diverse observational evidence that this dark matter is distributed throughout galaxies and reaches out beyond the luminous material in so called "halos".

The probably most established evidence for the existence of dark matter is the observed flatness of the rotation curves of spiral galaxies. While the light in these systems falls off exponentially with distance from the galaxy's centre, its rotational velocity is constant out to the limits of detection. But if light would mark mass, the rotational velocity should fall off with radius as $\propto \sqrt{1/r}$. This indicates that there has to be much more mass existing in these systems than can be accounted for by luminous matter.

Furthermore it has been shown in numerical studies as well as in N-body simulations that the flattened disks of "cold" spiral galaxies supported mainly by rotation are subject to large-scale (barlike) instabilities. As our own galaxy belongs to this kind of systems and doesn't seem to exhibit such an instability, there has to be some stabilising process. Models adding a spherical halo component of substantial mass to the disk have been considerably successful in producing the required stability (Ostriker and Peebles, 1973).

Dark halos enclosing galaxies have also been used to explain the rotational velocities of binary galaxies (Ostriker et al., 1974), warped disks (Tubbs and Sanders, 1979), H I flaring (Sanders, 1986) and the confinement of radio jets and lobes (Valtonen, 1980). As there are also other theories that can explain these observations, none of them can however be assumed to be thorough evidence for the existence of dark halos.

Since the simple concept of galaxies forming inside dark halos can explain many different types of observations though, their existence is today generally accepted.

Gravitational microlensing is currently the main technique for detecting and constraining compact objects in the subsolar mass range. It is the result of bending of light rays from cosmologically distant sources by gravitational fields along the line of sight and therefore sensitive to both luminous and dark matter. These compact objects are likely to be associated with the aforementioned dark halos.

Microlensing can plainly be detected through flux variability. The idea that most quasar variability originates from microlensing due to such compact objects has amongst others primarily been championed by Hawkins (e.g. Hawkins (1996), Hawkins and Taylor (1997), Hawkins (2001)). Other characteristic signatures for microlensing are uncorrelated flux variations in the different images of multiply-images quasars (Schild, 1996), differential magnification of emission regions having different scales (Dalcanton et al., 1994), apparently repeated gamma ray bursts

(Marani et al., 1999) and a broadening of the observed absolute brightness distribution of standard candles (Metcalf and Silk (1999), Wang (1999)).

Virtually all papers on microlensing by cosmologically distributed compact objects assume them to be uniformly and randomly distributed along the line of sight with a constant comoving density. This assumption is the so-called Press-Gunn approximation (Press and Gunn, 1973).

If this assumption does not correspond to the prevailing conditions, the results derived from many previous studies of microlensing at cosmological distances may be seriously flawed (Wyithe and Turner, 2002). These include not only constraints on the mass function, velocity dispersion and cosmological density of the microlenses but also predictions on the magnification bias and thereby potentially more fundamental cosmological quantities (Seljak and Holz, 1999).

The purpose of my work is to test the applicability of the Press-Gunn approximation by constructing a code that simulates the mass distribution along the line of sight to a light source at cosmological distance using Monte-Carlo methods.

Over the last few decades, the cold dark matter (CDM) model has been studied in detail. Its success in reproducing observational results, both on galactic and on cluster scales, has turned it into the standard paradigm for the formation of structure in the universe. Current cosmological data favours a $\Lambda$CDM cosmology with a present matter density $\Omega_M \sim 0.3$ and the density for the cosmological constant $\Lambda \sim 0.7$ (in units of the critical density). This agrees with a generic prediction from inflation, namely $\Omega_{tot} = \Omega_M + \Lambda = 1$, a "flat" universe.

In conformance with recent results from the Sloan Digital Sky Survey (SDSS) in combination with WMAP (Tegmark et al., 2004), the preset cosmological parameters used throughout this paper are $\Omega_M = 0.3$, with a baryonic density of $\Omega_b = 0.04$ and the CDM density $\Omega_{CDM} = 0.26$, $\Lambda = 0.7$ and the Hubble constant $H_0 = 70$ $km\ Mpc^{-1}\ s^{-1}$.

If desired, the cosmological parameters may be modified. The applied formulae hold for cosmologies in which the universe is composed of photons, baryons (and their accompanying electrons), massless neutrinos, cold dark matter and a potential cosmological constant. The calculations are applicable to Einstein-de Sitter ($\Omega_M = 1, \Lambda = 0$), open ($\Omega_M \leq 1, \Lambda = 0$) and flat ($\Omega_M + \Lambda = 1$) universes (due to limitations of the density profile formalism, compare Navarro et al. (1997)).

This paper is organised as follows. In Chapter 2 the applied formulae are given. Chapter 3 describes the numerical method used for the simulations. In Chapter 4 I discuss the results obtained by the output of my code. Conclusions are drawn in Chapter 5. The code itself and an example of adequate input are given in Appendix A and Appendix B respectively.

# Chapter 2

# Theoretical framework

As discussed in the introduction, current models of large scale structure predict dark matter to be located in a hierarchy of dark matter halos. Assuming that the compact objects giving rise to microlensing are composed of dark matter, it is essential to have profound knowledge of the properties of dark halos to be able to draw conclusions on the distribution of microlenses. Hence I begin by explaining the code's underlying formulae describing the physical characteristics of the dark matter halos.

## 2.1 Mass distribution of dark matter halos

It proves to be advantageous to use the quantity ln$\sigma^{-1}$(M,z) instead of M as the mass variable. $\sigma^2$(M,z) is the variance of the density field at redshift z in linear perturbation theory, smoothed with a real-space window function.

Following Mitchell et al. (2004) and Eisenstein and Hu (1999), this smoothed variance is given in terms of the power spectrum P(k) of the linear density field extrapolated to redshift zero by

$$\sigma^2(M,z) = \frac{b^2(z)}{2\pi^2} \int_0^\infty k^2 P(k) |\widetilde{W}_R(k;M)|^2 dk \tag{2.1}$$

The normalised linear growth factor is calculated from b(z)=D(z)/D(0), where the growth function is given by

$$D(z) = \left(\frac{1+z_{eq}}{1+z}\right) \frac{5\Omega(z)}{2} \left\{ \Omega(z)^{4/7} - \Omega_\Lambda(z) + \left[1 + \frac{\Omega(z)}{2}\right]\left[1 + \frac{\Omega_\Lambda(z)}{70}\right] \right\}^{-1}, \tag{2.2}$$

where

$$\Omega(z) = \Omega_M (1+z)^3 g^{-2}(z) \tag{2.3}$$

and

$$\Omega_\Lambda(z) = \Lambda g^{-2}(z), \tag{2.4}$$

with

$$g^2(z) = \Omega_M (1+z)^3 + (1 - \Omega_M - \Lambda)(1+z)^2 + \Lambda \tag{2.5}$$

and $z_{eq}$ as defined below (2.12).

$\widetilde{W}_R$(k;M) is the Fourier transform of a real-space window function $W_R$(r). In this project I used a spherical top hat of radius $R(M_R)$,

$$W_R(r) \propto \begin{cases} 1, & if\ r \leq R \\ 0, & otherwise, \end{cases} \tag{2.6}$$

$$\widetilde{W}_R(k) = \frac{3}{(kR)^3}(\sin kR - kR\cos kR), \tag{2.7}$$

$$M_R = \frac{4\pi}{3}\rho R^3. \tag{2.8}$$

The power spectrum of the density fluctuations is constructed from the transfer function in the usual way:

$$P(k) = 2\pi^2\delta_H^2(\frac{ck}{H_0})^{3+n}k^nT^2(k), \tag{2.9}$$

where $\delta_H$ is the amplitude of perturbations on the horizon scale today, given by

$$\delta_H = 1.95 \times 10^{-5}\Omega_M^{-0.35-0.19\ln\Omega_M-0.17\tilde{n}}e^{-\tilde{n}-0.14\tilde{n}^2} \qquad (\Lambda = 0) \tag{2.10}$$

$$\delta_H = 1.94 \times 10^{-5}\Omega_M^{-0.785-0.05\ln\Omega_M}e^{-0.95\tilde{n}-0.169\tilde{n}^2} \qquad (\Lambda = 1-\Omega_M) \tag{2.11}$$

and $\tilde{n} = n - 1$, where n is the initial power spectrum index, which is equal to 1 for a scale-invariant spectrum.

I restrict my calculations to a general CDM plus baryon universe to be able to take advantage of a commonly used fitting formula for the matter transfer function $T^2$(k) which is valid for both large and small scales (Eisenstein and Hu, 1998).

The physics governing the evolution of density perturbations in such a cosmology features three unique length scales:

- The horizon size at matter-radiation equality

  The transition from a radiation- to a matter-dominated universe occurred roughly at
  $$z_{eq} = 2.50 \times 10^4\Omega_M h^2\Theta_{2.7}^{-4}, \tag{2.12}$$
  where $h \equiv H_0/(100\ km\ s^{-1}\ Mpc^{-1})$, the Hubble parameter, and $\Theta_{2.7} = T_{CMB}/2.7$, with the cosmic microwave background temperature $T_{CMB}$.

  This leads to a particle horizon scale at the equality epoch of
  $$k_{eq} \equiv 7.46 \times 10^{-2}\Omega_M h^2\Theta_{2.7}^{-2}Mpc^{-1}, \tag{2.13}$$
  which sets the dynamics of the expansion and perturbation growth.

- The sound horizon at the time of recombination

  Defining the drag epoch $z_d$ as the time at which the baryons are released from the Compton drag of the photons and using numerical recombination results gives
  $$z_d = 1291\frac{(\Omega_M h^2)^{0.251}}{1+0.659(\Omega_M h^2)^{0.828}}[1 + c_1(\Omega_b h^2)^{c_2}], \tag{2.14}$$
  $$c_1 = 0.313(\Omega_M h^2)^{-0.419}[1 + 0.607(\Omega_M h^2)^{0.674}, \tag{2.15}$$
  $$c_2 = 0.238(\Omega_M h^2)^{0.223}. \tag{2.16}$$

  Prior to $z_d$, small-scale perturbations in the photon-baryon fluid propagate as acoustic waves with the sound speed $c_s$ = c/[3(1+R)]$^{1/2}$, where R is the ratio of the baryon to photon momentum density,
  $$R \equiv 3\rho_b/4\rho_\gamma = 31.5\Omega_b h^2\Theta_{2.7}^{-4}(z/10^3)^{-1}. \tag{2.17}$$

Hence, if defining the sound horizon at the drag epoch as the comoving distance a wave can travel prior to redshift $z_d$, the above implicates

$$s = \int_0^{t(z_d)} c_s(1+z)dt = \frac{2}{3k_eq}\sqrt{\frac{6}{R_eq}} ln \frac{\sqrt{1+R_d}+\sqrt{R_d+R_eq}}{1+\sqrt{R_eq}}, \qquad (2.18)$$

where $R_d \equiv R(z_d)$ and $R_{eq} \equiv R(z_{eq})$. In the presence of baryons, the growth of CDM perturbation is suppressed on scales below the sound horizon.

- The Silk damping length at recombination

  On small scales, the coupling between the baryons and the photons is not perfect, such that the two species are able to diffuse past one another. This so called Silk damping scale is well fitted by

$$k_{silk} = 1.6(\Omega_b h^2)^{0.52}(\Omega_M h^2)^{0.73}[1 + (10.4\Omega_M h^2)^{-0.95}]Mpc^{-1}. \qquad (2.19)$$

Thus the transfer function in a CDM-baryon universe should be divided into baryon- and cold dark matter contributions,

$$T(k) = \frac{\Omega_b}{\Omega_M}T_b(k) + \frac{\Omega_c}{\Omega_M}T_c(k), \qquad (2.20)$$

as the two species were dynamically independent before the drag epoch and afterwards their fluctuations are weighted by the fractional density they contribute. It should however be noted that $T_b$ and $T_c$ are no true transfer functions themselves, as they do not reflect the density perturbations of the respective species today.

To introduce the suppression of the growth of CDM perturbations by baryons as mentioned above, two solutions near the sound horizon are interpolated, resulting in

$$T_c(k) = f\tilde{T_0}(k, 1, \beta_c) + (1-f)\tilde{T_0}(k, \alpha_c, \beta_c) \qquad (2.21)$$

where

$$f = \frac{1}{1 + (ks/5.4)^4}, \qquad (2.22)$$

and

$$T_b(k) = \left[\frac{\tilde{T_0}(k,1,1)}{1+(ks/5.2)^2} + \frac{\alpha_b}{1+(\beta_b/ks)^3}e^{-(k/k_{Silk})^{1.4}}\right] j_0(k\tilde{s}) \qquad (2.23)$$

with $j_0(x) \equiv$ (sin x)/x, the spherical Bessel function,

$$\tilde{s}(k) = \frac{s}{[1 + (\beta_{node}/ks)^3]^{1/3}}, \qquad (2.24)$$

the effective sound horizon, and the node shift parameter

$$\beta_{node} = 8.41(\Omega_0 h^2)^{0.435}. \qquad (2.25)$$

The pressureless transfer function $\tilde{T_0}$ is given in terms of

$$\tilde{T_0}(k, \alpha_c, \beta_c) = \frac{ln(e + 1.8\beta_c q)}{ln(e + 1.8\beta_c q) + Cq^2}, \qquad (2.26)$$

where

$$C = \frac{14.2}{\alpha_c} + \frac{386}{1 + 69.9q^{1.08}}, \qquad (2.27)$$

with the variable

$$q = \left(\frac{k}{Mpc^{-1}}\right)\Theta_{2.7}^2(\Omega_M h^2)^{-1} = \frac{k}{13.41k_{eq}} \qquad (2.28)$$

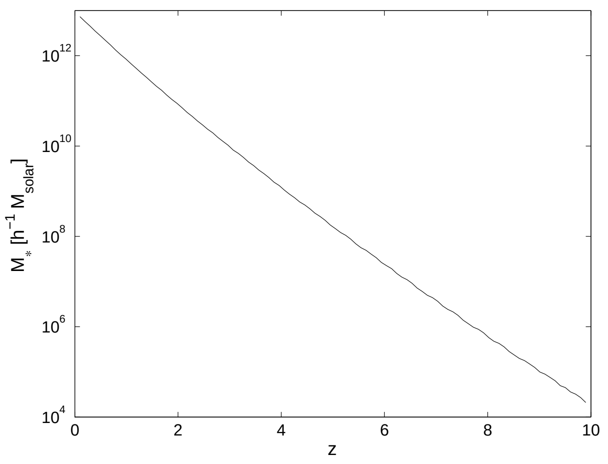


Figure 2.1: The characteristic mass scale $M_*(z)$, given by $\sigma(M, z) = \delta_c(z)$, using the preset parameters.

and the CDM suppression

$$\alpha_c = a_1^{-\Omega_b/\Omega_M} a_2^{-(\Omega_b/\Omega_M)^3}, \tag{2.29}$$

with

$$a_1 = (46.9\Omega_M h^2)^{0.670}[1 + (32.1\Omega_M h^2)^{-0.532}] \tag{2.30}$$

and

$$a_2 = (12.0\Omega_M h^2)^{0.424}[1 + (45.0\Omega_M h^2)^{-0.582}], \tag{2.31}$$

the shift in CDM log, $\beta_c$, as

$$\beta_c^{-1} = 1 + b_1[(\Omega_c/\Omega_M)^{b_2} - 1], \tag{2.32}$$

with

$$b_1 = 0.944[1 + (458\Omega_M h^2)^{-0.708}]^{-1} \tag{2.33}$$

and

$$b_2 = (0.395\Omega_M h^2)^{-0.0266}, \tag{2.34}$$

the baryon suppression

$$\alpha_b = 2.07k_{eq}s(1 + R_d)^{-3/4}G\left(\frac{1 + z_{eq}}{1 + z_d}\right), \tag{2.35}$$

where

$$G(y) = y\left[-6\sqrt{1+y} + (2+3y)ln\left(\frac{\sqrt{1+y}+1}{\sqrt{1+y}-1}\right)\right] \tag{2.36}$$

and

$$\beta_b = 0.5 + \frac{\Omega_b}{\Omega_M} + \left(3 - 2\frac{\Omega_b}{\Omega_M}\right)\sqrt{(17.2\Omega_M h^2)^2 + 1}. \tag{2.37}$$

An elaborate instruction on how to calculate the power spectra for a wider range of cosmological parameters and an other popular choice for the window function are given in the paper by Eisenstein and Hu (1999).

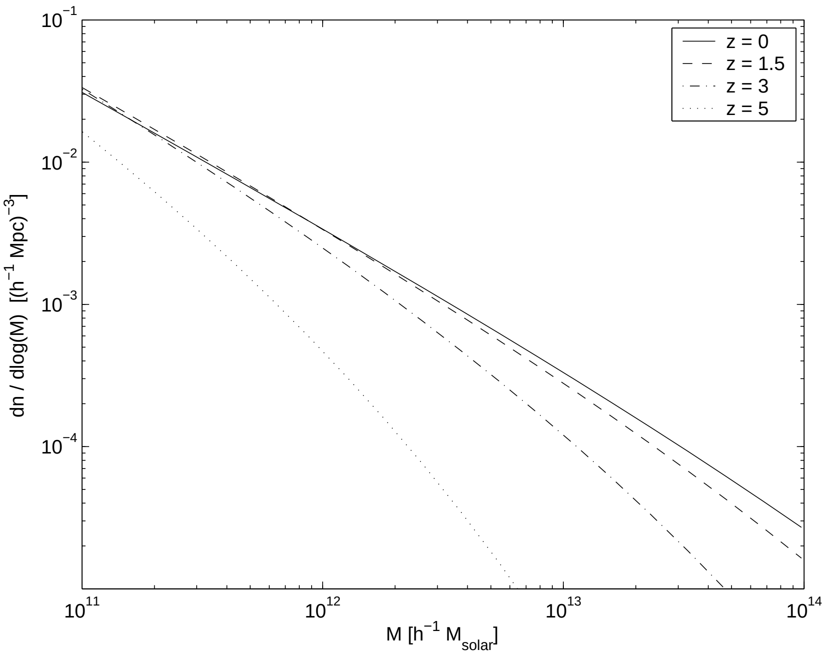


Figure 2.2: Comoving mass function for varying redshift using the $\sigma^2$-approximation (2.38), the Seth & Tormen form (2.41) and the preset parameters.

In order to ensure the efficiency of my method of computation I adopt an estimate of the variance $\sigma^2$, admissible under the assumption of a simple power law spectrum, P(k)$\propto k^{n_{eff}}T^2(k)$ (Yano et al., 1996):

$$\sigma^2(M,z) = \delta_c^2(z)(\frac{M}{M_*(z)})^{-(n_{eff}+3)/3}, \tag{2.38}$$

where $n_{eff}$ is the effective slope of the power spectrum. According to Reed et al. (2003), $n_{eff}$ is set to -2.15 which represents an applicable approximation over the required ranges in halo mass and redhift.

$M_*$(z) is the characteristic mass scale such that $\sigma(M_*(z)) = \delta_c(z)$, with

$$\delta_c(z) = \begin{cases} 0.15(12\pi)^{\frac{2}{3}}[\Omega_M(z)]^{0.0185} & \text{if } \Omega_M < 1 \text{ and } \Lambda = 0, \\ 0.15(12\pi)^{\frac{2}{3}}[\Omega_M(z)]^{0.0055} & \text{if } \Omega_M + \Lambda = 1, \end{cases} \tag{2.39}$$

the critical threshold of the initial overdensity for spherical collapse at z. To determine the characteristic mass scale, $M_*$(z), I use the accurate machinery as described above.

Using these quantities the halo mass function f($\sigma$,z) is defined as

$$f(\sigma,z) \equiv \frac{M}{\overline{\rho}(z)}\frac{dn(M,z)}{dln\sigma^{-1}}, \tag{2.40}$$

where $\overline{\rho}(z)$ is the mean density of the universe at redshift z and n(M,z) is the abundance of halos with mass less than M at that time (Jenkins et al., 2001).

As indicated by N-body simulations of structure formation in various cold dark matter models, the mass function is well fitted by a modification of the Press-Schechter function introduced by Sheth and Tormen (1999):

$$f(\sigma; S-T) = A\sqrt{\frac{2a}{\pi}}\Big[1+(\frac{\sigma^2}{a\delta_c^2})^p\Big]\frac{\delta_c}{\sigma}\exp\Big[\frac{-a\delta_c^2}{2\sigma^2}\Big], \tag{2.41}$$

with the fitting parameters a=0.707, p=0.3 and A(p)=0.3222.

Equation (2.41) shows the convenience of using ln$\sigma^{-1}$ as the mass variable. The mass function has no explicit dependency on redshift or power spectrum, but describes structure in all gaussian hierarchical clustering models at all times in any cosmology, provided that abundances are plotted in the f - ln($\delta_c/\sigma$) plane.

Combining (2.41) with (2.40) then gives the mass function in the form used in my simulations:

$$\frac{dn(M,z)}{dlogM} = \frac{\overline{\rho}(z)}{M}\frac{dln\sigma^{-1}}{dlogM}A\sqrt{\frac{2a}{\pi}}\left[1+(\frac{\sigma^2}{a\delta_c^2})^p\right]\frac{\delta_c}{\sigma}\exp\left[\frac{-a\delta_c^2}{2\sigma^2}\right]. \tag{2.42}$$

The performance of the mass function using the above mentioned approximations can be assessed as good when comparing to corresponding N-body data (e.g. Reed et al. (2003), Kravtsov et al. (2004)).

## 2.2 Halo density profile

As generally accepted, I assume that the halo density distribution follows the equilibrium density profiles of dark matter halos in a hierarchically clustering universe obtained in high-resolution N-body simulations (Navarro et al. (1997), hereafter NFW). It can be expressed as a simple analytical function, namely

$$\frac{\rho(r)}{\rho_{crit}} = \frac{\rho_c}{(r/r_s)(1+r/r_s)^2} \tag{2.43}$$

where $r_s$ is a scale radius, $\rho_c$ is a characteristic (dimensionless) density and $\rho_{crit} = 3H^2/8\pi G$ is the critical density for closure.

This can be transposed into the following expression:

$$\rho(r) = \frac{3H_0^2}{8\pi G}(1+z)^3\frac{\Omega_M}{\Omega(z)}\frac{\rho_c}{cx(1+cx)^2} \tag{2.44}$$

where x = r/$r_{200}$, with the halo virial radius defined as

$$r_{200} = 1.63\times10^{-2}\left(\frac{M}{h^{-1}M_\odot}\right)^{1/3}\left(\frac{\Omega_M}{\Omega(z)}\right)^{-1/3}(1+z)^{-1}h^{-1}kpc, \tag{2.45}$$

the radius which spans a sphere so that the mean enclosed density is 200 times the critical value, $\rho_{crit}(z)$, and $\Omega(z)$ as given by equation (2.3).

$c = r_{200}/r_s$ is the concentration parameter given by

$$c = \frac{124}{1+z}\left(\frac{M}{h^{-1}M_\odot}\right)^{-0.084} \tag{2.46}$$

(Oguri and Lee, 2004) and linked to the characteristic density, $\rho_c$, by

$$\rho_c = \frac{200}{3}\frac{c^3}{ln(1+c)-c/(1+c)} \tag{2.47}$$

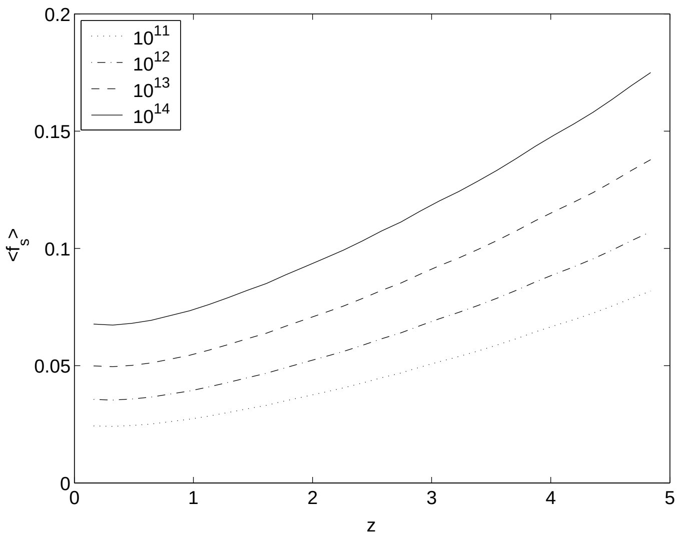


Figure 2.3: Redshift dependence of the average subhalo mass fraction, $< f_s >$, for parent halos of 4 different masses. Each curve is labelled by M(z) (in $h^{-1}M_\odot$).

## 2.3 Substructure

The formation of structure in our universe processed, according to the standard CDM scenario, hierarchically. Small-scale fluctuations collapsed first into virialised objects whose inner regions were, due to their high compactness, often able to survive accretion onto a larger system and thus to become self-bound subhalos of their host. The evolution of this substructure is governed by the forces that try to dissolve it: tidal forces, impulsive collisions and dynamical friction. Thereby the subhalo will experience mass loss while the parent mass, M, increases on account of onward merging and accretion.

Assuming that the average distribution of subhalo orbits is independent of parent halo mass, an average mass loss rate can be considered which only depends on redshift and the momentary mass ratio of subhalo to parent halo, $\Psi$ = m/M.

By matching the subhalo mass function (SHMF) of galaxy- to cluster-sized dark matter halos achieved from high-resolution numerical simulations, an average SHMF with a Schechter function of the form

$$\frac{dn}{dln\Psi} = \frac{\gamma}{\beta\Gamma(1-\alpha)}\left(\frac{\Psi}{\beta}\right)^{-\alpha} exp\left(-\frac{\Psi}{\beta}\right) \tag{2.48}$$

is obtained, where

$$\gamma = \frac{f_s}{P(1-\alpha, 1/\beta) - P(1-\alpha, 10^{-4}/\beta)}, \tag{2.49}$$

the normalised total subhalo mass fraction, with the subhalo mass fraction, $f_s$, fitted by

$$log[< f_s >] = \sqrt{0.4(log[M/M_*]+5)} - 2.74 \tag{2.50}$$

with P(a,x), the incomplete Gamma function, and $M_*$, the critical mass scale as defined above.

The power-law slope $\alpha$ scales nearly linearly with log(M/$M_*$), which is best fit by

$$\alpha = 0.996 - 0.028log(M/M_*) \tag{2.51}$$

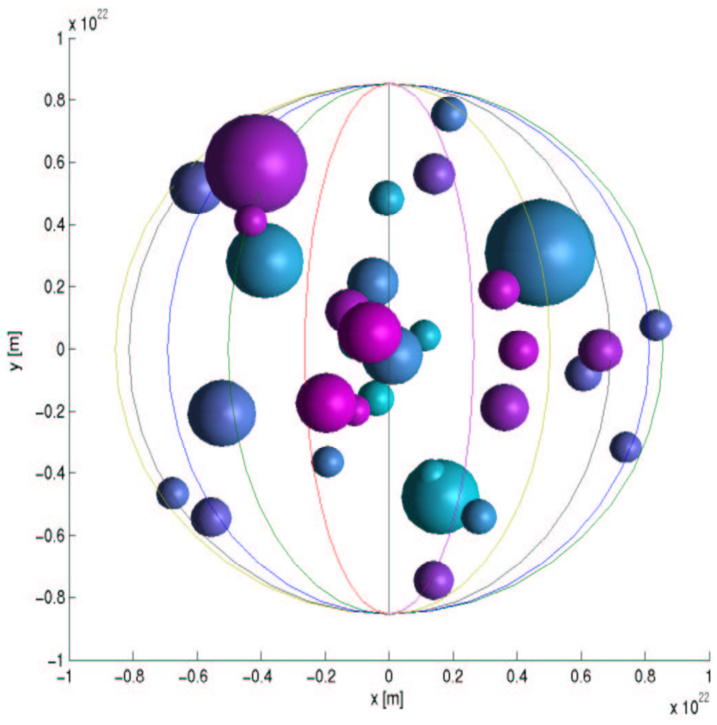


Figure 2.4: Example of subhalo distribution inside a parent halo of mass $M = 2.5 \times 10^{12} M_\odot$ at redshift 0. Purple subhalos are located in the foreground while turquoise ones are farther.

while the parameter $\beta$ has a best fit of $\beta = 0.13$ (van den Bosch et al., 2004).

So both the slope and the normalisation of the subhalo mass function depend solely on the ratio of parent halo mass, M, and characteristic mass, $M_*$. This merely reflects a halo formation time dependence. Parent halos that form earlier have a lower subhalo mass fraction, because there is relatively more time for subhalo mass loss to operate.

When analysing the results of several high resolution halo simulations, the evidence is that the distribution of the subhalos within their parent halo generally speaking does neither depend on the subhalo nor the parent halo mass. It appears however that as the density profile of the parent halo becomes more concentrated, so does that of the subhalo population, too. This effect is yet much weaker for the subhalos than for mass as a whole.

Following Gao et al. (2003), the radial distribution of subhalos within their parent halos is well fit by

$$\frac{n(<x)}{N} = \frac{(1+ac)x^\beta}{1+acx^\alpha}, \tag{2.52}$$

where x is the distance to the host centre in units of $r_{200}$ (2.45), n(x) is the number of subhalos within x, N is the total number of subhalos inside $r_{200}$, a = 0.244, $\alpha = 2$, $\beta = 2.75$ and c = $r_s/r_{200}$ is the concentration of the host halo (2.46).

Since the above mentioned form of the SHMF is valid for parent halo masses between $\sim 10^{10}$ to $10^{15} h^{-1} M_\odot$ and a subhalo to parent halo mass ratio, $\Psi$, which obeys -4 $< log\Psi < 0$ (van den Bosch et al., 2004), the default subhalo mass boundaries are set to $5 \times 10^8 M_\odot$ respective $10^{11} M_\odot$ and only one level of substructure is allowed, meaning that there is no generation of substructure within subhalos.

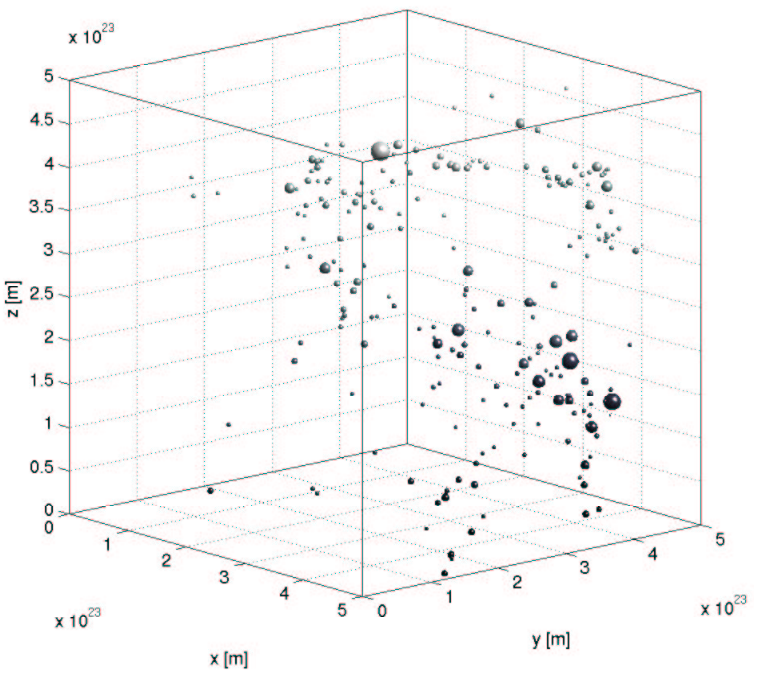


Figure 2.5: Graphic display of a cut-out of the GIF halo catalogue for z = 1.05. The spatial extend of each halo is assessed by its respective virial radius (compare equation (2.45)).

## 2.4 N-body simulations

As became apparent above, cosmological N-body simulations play a pivotal role in the study of cosmic structure formation.

In this methodology, initial conditions are set at an early epoch by using linear theory to calculate the statistical properties of fluctuations. Such a computation requires some specific mechanism for generating primordial structure, together with assumptions about the nature of the dominant dark matter component and the global cosmological parameters. N-body simulations are then used to follow the later evolution of the dark matter into the nonlinear regime where it can be compared with the large-scale structure in galaxy surveys.

Additionally to generating halos according to the halo mass function as derived above (2.42), my code offers the option to directly use N-body data which is taken from the GIF N-body simulations of the $\Lambda$CDM model, coupled to a semi-analytic galaxy formation model (Kauffmann et al., 1999a).

The code used for the GIF simulations is called Hydra. It is a parallel adaptive particle-particle particle-mesh ($AP^3M$) code (for details see Couchman et al. (1995), Thomas et al. (1996)). The data are publicly available at www.mpa-garching.mpg.de/GIF/.

I used the halo catalogues, which are given at various redshifts in a comoving cosmological simulation box of size 141 $h^{-1}$Mpc, to extract information about the mass and the respective coordinates of halos. The cosmological parameters of the GIF $\Lambda$CDM model are given by $\Omega_M = 0.3$, $\Lambda = 0.7$, $h = 0.7$ and $\sigma_8 = 0.9$, corresponding to the default parameters used in the program.

A specification on how these data are employed can be found in section 3.1.2.

## 2.5 Smooth matter

As the code only accepts a finite mass range for generation of haloes, the question arises which fraction of the total matter density ($\Omega(z)\rho_{crit}(z)$) actually is concentrated in haloes within these limits.

The program accepts in fact several options which can be selected in the input file (see appendix B). These are:

**All mass is clustered in halos.** Any simulation volume comprises halos of masses within the set interval which, when summed up, correspond to the critical mass expected for the selected cosmological parameters and time. The mass distribution of the halos matches the one derived in section 2.1.

**The fraction follows the mass function.** Halos are generated according to the Press-Schechter mass function (2.42). All halos of masses outside the mass range contribute to a background density (see curve "Press-Schechter" in figure 2.6).

**The fraction follows the GIF data.** The halo generation is carried out pertinent to the the mass distribution within the preset mass range which corresponds to the span of halo masses in the N-body simulations of the GIF project. The fraction $\rho_{halos}/\rho_{matter}(z)$ obeys the performance of the same quantity calculated from the GIF data (see curve "GIF data" in figure 2.6). The discrepancy between the total mass and the critical mass of the simulation volume is smoothly distributed as a background density, calculated by a 5th degree polynomial fit which is of good quality up to a redshift of 10.

The disagreement in the outcomes of the GIF data method and the Press-Schechter mass function approach is striking. This problem is however recognised by the scientists involved in the GIF project (Kauffmann et al., 1999a) as well as by Sheth & Tormen (Sheth et al., 2001). For halo masses between $10^{11}$ and $10^{14}M_{\odot}$

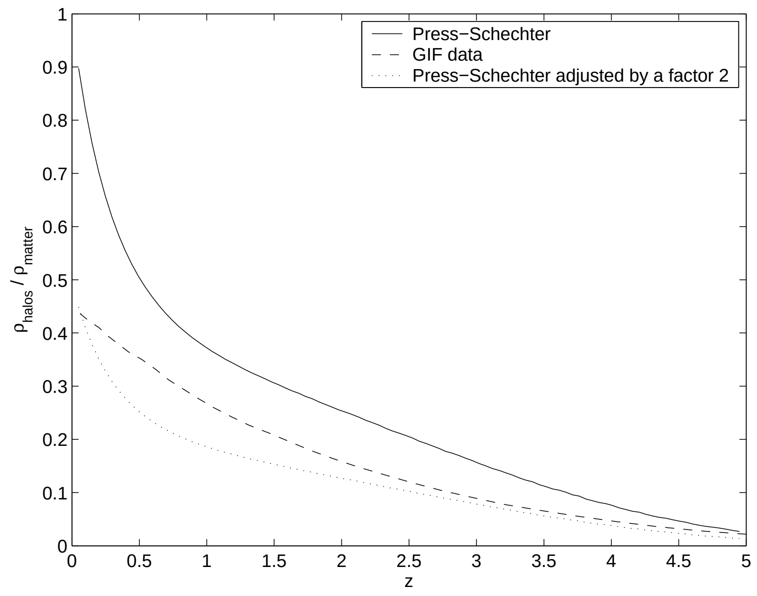


Figure 2.6: Evolution of $\rho_{halos}/\rho_{matter}$(z) using different manners of determination as discussed in the text and the preset parameters (mass range $10^{11} - 10^{14}M_{\odot}$).

(the default mass range), the Press-Schechter theory predicts roughly twice as many halos at redshift 0 as are actually found in the GIF simulations for both the $\tau$CDM and the here used $\Lambda$CDM cosmologies. As indicated by the pertained persons, most of the missing mass in the N-body simulations is in the form of "unresolved" material - single particles or groups with less than 10 members.

Figure 2.6 shows also the fourth option:

**The fraction follows the mass function adjusted by a factor 2** This approach proceeds identically to the original Press-Schechter method but simply modifies the halo number density with the above mentioned factor 2. Although this entails an enhancement in the consistency of GIF data and Press-Schechter theory for certain redshifts, it is commended to use the Press-Schechter machinery or, for a direct comparison with the implementation using N-body data, to use the $\rho_{halos}/\rho_{matter}(z)$ fraction which is following the GIF data.

# Chapter 3

# Numerical method

The basic concept of my approach consists in generating a cylindric volume between the observer at redshift zero and a light source at redshift z and subsequently determining the mass distribution along the line of sight, which is represented by a small cylinder aligned to the axis of the larger simulation cylinder (see figure 3.1). Thereto the code divides the simulation volume into a number of subvolumes and determines the ratio of actual mass density to expected mass density, $\frac{\rho(n)}{\rho_{exp}(z_n)}$, for each subcylinder.

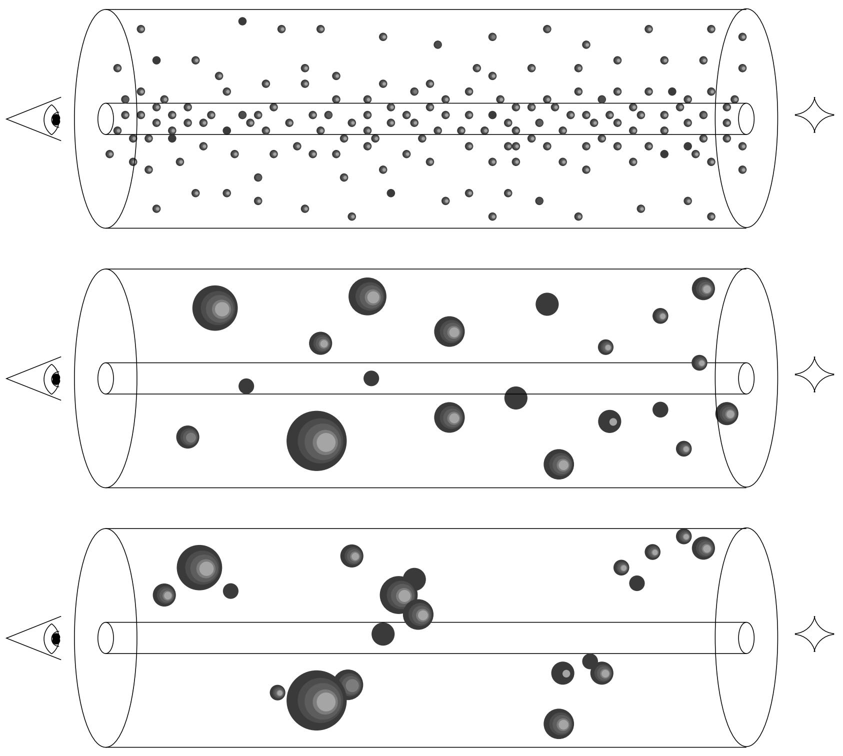


Figure 3.1: Schematic diagram of the basic idea of the method. The observer is symbolised leftmost, the light source on the right. Top panel: Press-Gunn approximation. The compact objects are uniformly and randomly distributed. Middle panel: The compact objects are associated with dark halos which are randomly distributed. Bottom panel: The compact objects are associated with dark halos which cluster.

## 3.1 Large / small cylinder

When deciding which value to assign to the radius of the large cylinder, the size of the simulation objects has to be considered. The simulation cylinder radius has to be larger than the virial radius of a maximum mass halo to avoid errors due to clustering effects around the line of sight. To ensure the codes efficiency, the radius shouldn't be chosen too large either, as the size of the simulation volume plays a major role when it comes to computing time.

For the preset mass range as well as for the GIF N-body data the maximum halo mass is $10^{14} M_\odot$ which corresponds to a virial radius ($r_{200}$, see equation (2.45)) of nearly $3 \times 10^{22} m$ at zero redshift. The preselected simulation cylinder radius is therefore set to $5 \times 10^{22} m$. It can however be modified if a different mass range is desired.

The size of the small cylinder, representing the line of sight, is preset to $10^{17}$m. Factors that have to be taken in consideration when selecting a value for this quantity are discussed in section 3.2.

The simulation volume then gets replenished with halos according to the chosen mechanism. For this the code offers several options as stated in the following.

### 3.1.1 Halo generation applying the Press-Schechter mass function

The first step is to divide the simulation cylinder into subvolumes by interposing 30 lens planes evenly spread in redshift. This is done in accordance with the parameter setting in the microlensing code by Zackrisson and Bergvall (2003) to be able to employ the output of my program in their code. The number of lens planes can however be modified if desired as it is a global parameter in the code.

Following Surpi et al., the length of the $n$th subcylinder is given by the corresponding light travel distance, calculated by

$$d_n = \frac{c}{H_0} \int_{z_{n-1}}^{z_n} \frac{1}{(1+z)g(z)} dz \tag{3.1}$$

where g(z) is given by equation (2.5), c denotes the speed of light and $z_0 = 0$.

Each subvolume becomes assigned to the lens plane in which it ends, which means that it adopts the values of the cosmological parameters valid for its lens plane ($\Omega(z_n)$, $\Omega_\Lambda(z_n)$ and $\rho_{exp}(z_n) = \Omega(z_n)\rho_{crit}(z_n) = \Omega_M \rho_{crit}(1+z_n)^3$).

Utilising these, the program determines the respective critical mass scale $M_*(z_n)$ defined in section 2.1. This quantity is needed to get an approximation for the variance of the density field, $\sigma^2(z_n)$, as given in equation (2.38), which is crucial for the halo mass function (2.42).

Depending on which option for the evaluation of the mass fraction clustered in haloes is selected (see section 2.5), the code is now able to assess the total mass in haloes - within the selected mass range - expected for the given subvolume.

In the next step, haloes are generated according to the mass function (2.42) until their accumulated mass comes up to the expected total mass as estimated above $\pm 5 * M_{min}$, the minimum halo mass. This assignation provides for a nearly even spread of accumulated masses around the expected total ones when using the default parameters and thereby avoids a systematic over- or underfilling.

The discrepancy between the actual density and the expected matter density for the specified redshift, $\rho_{exp}(z_n)$, is stored as background density of the subvolume.

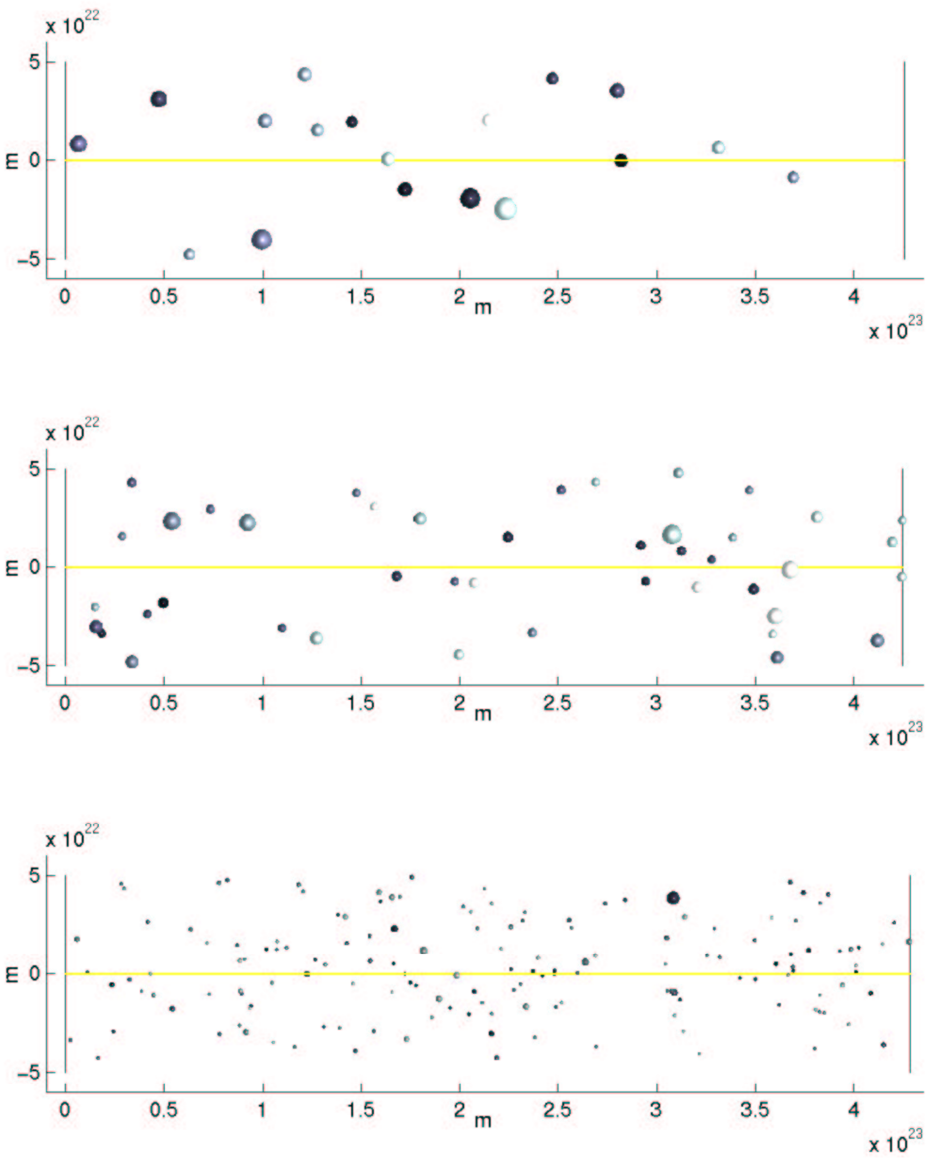


Figure 3.2: Example of halo distribution inside a cylindric subvolume for 3 different redshifts (at the top: z = 0, in the middle: z = 1, bottom: z = 4). The simulations are carried out using the preset parameters, a halo mass fraction following the Press-Schechter mass function (see section 2.5) and without subhalo generation. The line of sight ($\equiv$ the cylinder axis) is illustrated by the yellow straight line and the darker a halo the farther it is situated.

For each halo a pair of cylindric coordinates based on the subvolume gets randomly generated and assigned.

A graphic display of the outcome of such a subvolume routine with varying redshift is shown in figure 3.2.

The evolution with redshift is striking, particularly when considering that the simulations are based on a halo mass fraction following the mass function (see section 2.5). This means that the visualised haloes in the upper panel comprise roughly 90 percent of the total mass present in the given volume while the haloes shown in the lower panel account for only around 10 percent (compare figure 2.6 ).

Thus in the lower panel there is additionally nearly 9 times as much mass distributed in a smooth background density as there is clustered in haloes.

Consequently the output of this process consists in 30 subvolumes, each filled with haloes according to the mass function for the given redshifts, their coordinates based on the subcylinder they belong to and potentially a background density.

Stringing these together in the correct order then produces the large simulation cylinder.

### 3.1.2 Assembling the cylindric volume using N-body data

Alternatively to halo generation according to the Press-Schechter mass function, the program provides the possibility to employ N-body data for the build-up of the simulation volume.

The data implemented in my code are a selective version of the GIF project halo catalogues that can be found at www.mpa-garching.mpg.de/GIF/.

It is composed of one file per available redshift, containing information about mass and respective coordinates for each halo comprised in the comoving simulation box at that time.

The basic idea of this routine is to construct the large simulation cylinder by cutting subcylinders out of the N-body simulation boxes and stringing them together to assemble the simulation cylinder (see figure 3.5).

The first step in this machinery is to determine which N-body files are needed. The program requires the data for all available redshifts ($z_{nb}$) up to at least the light source redshift to be able to carry out the ensuing computations.

Thereafter the corresponding distances are calculated, analogue to equation (3.1). These light travel distances equal the lengths of the respective subcylinder that will be cut out of the N-body simulation boxes. The subcylinder lengths using all available redshifts and the preset parameters are plotted in figure 3.3.

As the N-body data is given for a comoving box of size 141 $h^{-1}$ Mpc, all coordinates have to be corrected before use in the code. This conversion is done according to following formula:

$$[x, y, z] = h^{-1}Mpc(1 + z_{nb})^{-1}[x, y, z]_{GIFdata} \tag{3.2}$$

The converted simulation box sizes are shown in figure 3.4.

It is apparent that the simulation box size doesn't fall below the preset simulation cylinder diameter of $10^{23}$m for any given redshift $z_{nb}$.

In case it is desired to modify the simulation cylinder radius, it has to be kept in mind that the resulting diameter has to be not less than the box size of the highest required redshift $z_{nb}$.

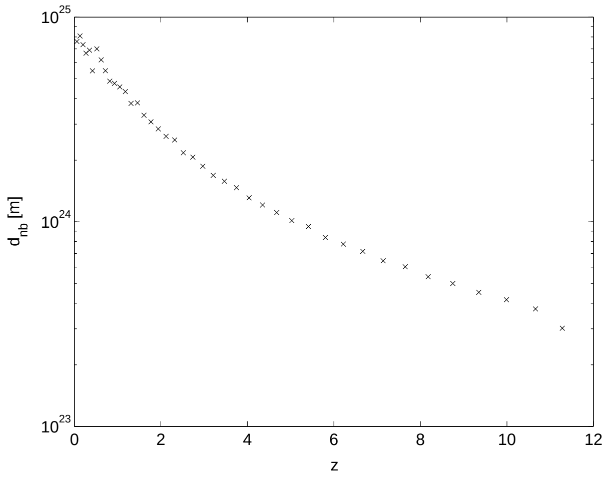


Figure 3.3: Length of the $nb$th subcylinder which is assigned to redshift $z_{nb}$. The calculations are carried out according to equation (3.1) using the preset cosmological parameters.

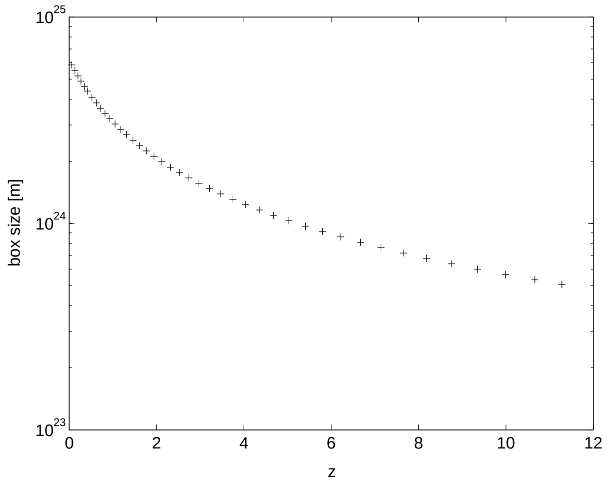


Figure 3.4: Converted size of the nbth N-body simulation box which is assigned to redshift $z_{nb}$. The calculations are carried out according to equation (3.2) using the preset cosmological parameters.

When comparing figure 3.3 with figure 3.4 it becomes however clear that the box size can be less than the needed subcylinder length for certain redshifts. For the preset parameters this is the case for all redshifts up to $z_{nb=29} = 4.68$.

Thus when cutting the subcylinders out of their respective N-body simulation cube, it has to be checked if they fit entirely. Otherwise the subcylinder is assembled by repeating the cutting routine and stringing the parts together until the required length is obtained.

The cutting routine works identical for all redshifts and their associated simulation boxes.

First a random point in the (x,y,0)-plane is picked which complies with the requirement that it has a minimum distance equal to the simulation cylinder radius to the nearest edges of the cube. This point, $(x_s, y_s, 0)$, is then adopted as start of the simulation cylinder axis. Its endpoint is consequently $(x_s, y_s, d_{nb})$, if $d_{nb} \leq$ box size$_{nb}$. Otherwise the endpoint is set to $(x_s, y_s, z_{max})$ and a dummy, $d^*_{nb}$, is created and set to $d_{nb} - z_{max}$ (figure 3.5 a).

Subsequently all halos within the simulation cylinder are selected and their coordinates are converted into polar coordinates with respect to the cylinder.

In the case that $d_{nb}$ exceeds the size of the respective simulation box, the routine is carried out once more, this time using $d^*_{nb}$ as cylinder length. When converting the halo coordinates, the z-coordinate becomes this time however $z_{halo} + z_{max}$. This simply implicates that the in this manner produced subcylinder gets merged with the simulation subcylinder from the first step (figure 3.5 b).

After executing this routine for all given $z_{nb}$, the simulation volume is finally build up by adding the subvolumes in the right order (figure 3.5 c).

In accordance with the method based on the Press-Schechter mass function the large simulation cylinder is subsequently apportioned by interposing a number

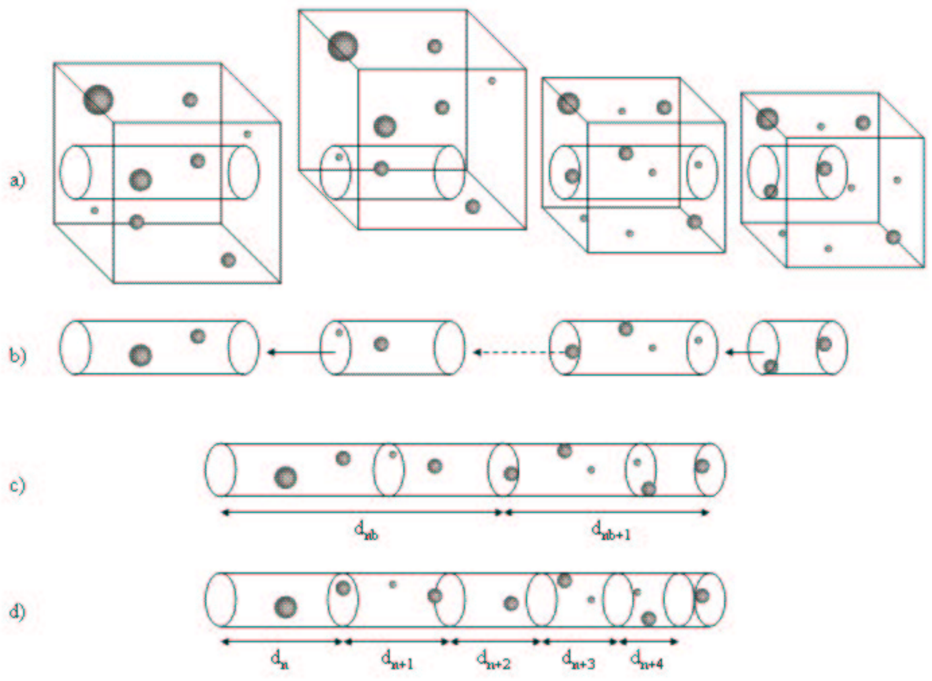


Figure 3.5: Schematic diagram of the basic concept of simulation volume assembling using the N-body data. See text for details.

of lens planes (preset to 30), evenly spread in the redshift range up to the light source redshift (compare section 3.1.1). Thereby subvolumes with lengths given by equation (3.1) are generated and the halo coordinates are transferred to their respective subcylinder (figure 3.5 d).

In a final step a background density is assigned to each subvolume, based on the GIF data mass fraction fit (see section 2.5).

Thus both methods yield $n$ subvolumes of length $d_n$, each filled with halos of certain masses, assigned to coordinates with respect to the subvolume, and an adequate background density.

Subsequently the process steps are the same regardless of which machinery was chosen to produce the simulation volume.

If subhalo generation was selected, the code determines the mass fraction in substructure for each halo according to equation (2.50). The expected mass in subhalos is then stored and the parent halo mass gets reduced by the same amount.

Next the cut-off radii of all haloes, which define their spatial boundaries, are computed. For all points, which lie outside a sphere of cut-off radius around the halo coordinates, the density contribution of this halo is set to zero.

Following common practice, I adopt $r_{200}$ as cut-off radius (see equation (2.45)).

If no substructure production is demanded, the simulation volume contains now all information necessary to accomplish the determination of the mass distribution along the line of sight.

Otherwise the program checks which halos are in direct contact with the line of sight. This means it selects those halos, whose coordinates have a maximum distance, equal to their cut-off radius and the small cylinder radius combined, to the simulation cylinder axis.

To save computing time, substructure generation takes only place for these halos.

The subhalo production proceeds likewise for all relevant halos.

Subhalos are generated according to the average subhalo mass function (see equation 2.48) until the cumulative mass in substructure is equal the above calculated expected mass in subhalos $\pm 5 * M_{min}^{sub}$, the minimum substructure mass. This is done in accordance with the method used when filling up the subvolumes with the Monte-Carlo routine (compare with section 3.1.1) to avoid a constant under-run or excess of the expected total mass in substructure.

Subsequently, every subhalo gets assigned to spherical coordinates following the distribution given by equation (2.52). These coordinates are then transformed into cartesian coordinates, with the parent halo coordinates as origin, the y-axis parallel and the z-axis perpendicular to the simulation cylinder axis (see figure 3.6).

To be able to evaluate the contribution of a subhalo to the density inside the small integration volume, its distance to the simulation cylinder axis, $r_{sub}$, and its height inside its subvolume, $h_{sub}$, are required. These are calculated with help of basic calculus as

$$r_{sub} = \sqrt{z_{sub}^2 + r_{parent}^2 - 2 * z_{sub} * r_{parent} + x_{sub}^2} \tag{3.3}$$

and

$$h_{sub} = h_{parent} + y_{sub}, \tag{3.4}$$

where $x_{sub}$, $y_{sub}$ and $z_{sub}$ are the cartesian coordinates of the subhalo as described above, while $r_{parent}$ and $h_{parent}$ are polar coordinates of the parent halo with respect

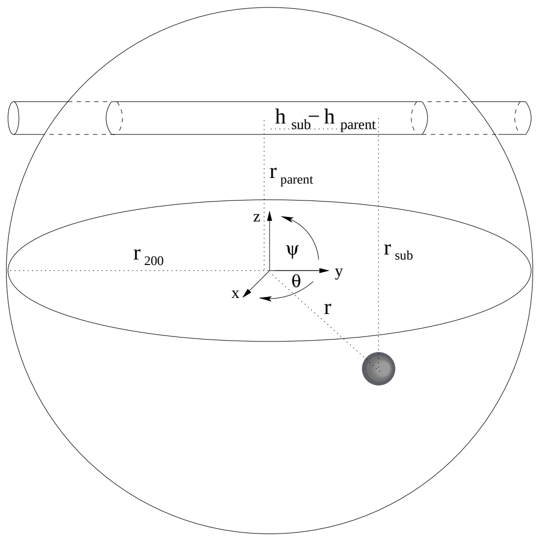


Figure 3.6: Schematic diagram of the different coordinate systems assigned to the subhaloes inside their parent haloes, as explained in the text.

to its subvolume.

In a last step the cut-off radii for all subhalos are computed, again adopting $r_{200}$ and using equation (2.45).

After having accomplished this procedure for all halos of interest, the simulation volume contains information, in form of mass, position and cut-off radius, about all halos and, if needed, their respective subhalos for all subvolumes.

## 3.2 Numerical halo integration

After the assembling of the large simulation cylinder volume (compare section 3.1), the program is now able to determine the mass distribution along the line of sight by computing the mass density for each small subcylinder within its respective simulation subvolume.

That mass density emerges from the halo contribution and potentially a subhalo component.

Allthough there are two different functions implemented in the code, the machineries to determine the density in halos and in subhalos are the same. The distinction is simply due to a discrepancy in storage format.

In case of substructure the program processes halos and subhalos separately and superposes the corresponding densities afterwards.

The technique applied in the code works in an analogous manner for all objects in all subvolumes.

In a first step it is checked if the current (sub)halo actually encounters the small cylinder. To be assessed relevant, an object needs to satisfy two conditions: Its coordinates are not allowed to correspond to a distance to the simulation cylinder axis larger than their cut-off radius and the small cylinder radius combined and it mustn't cut across the lens plane confining its subvolume towards lower redshift. The latter has the purpose of avoiding a constant excess by taking haloes, that don't lie completely inside their respective subvolume, into account.

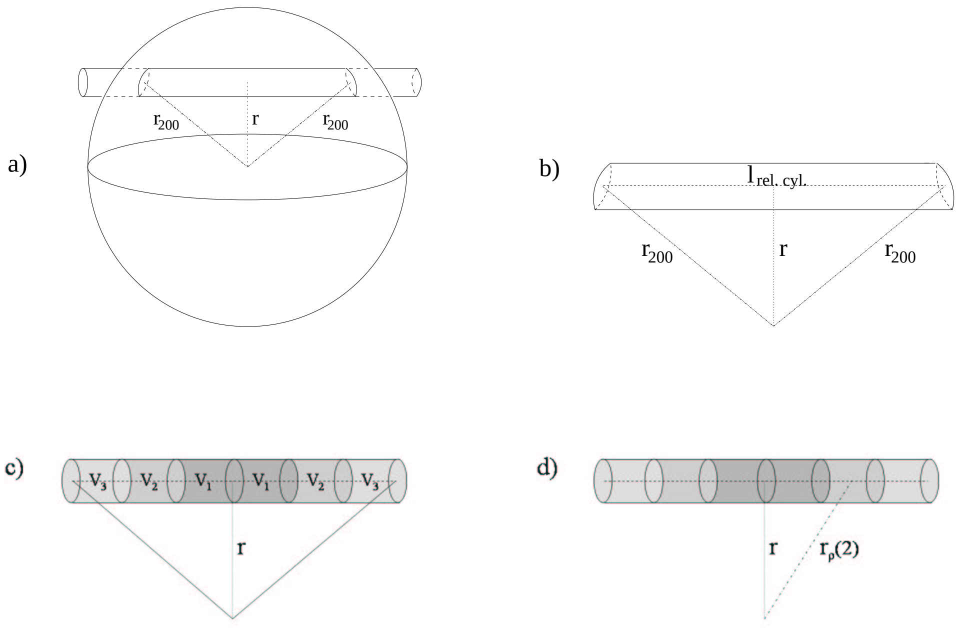


Figure 3.7: Schematic diagram of the basic concept of simulation volume assembling using the N-body data.

If this is the case, the code evaluates the (sub)halo's share of mass that lies within the small cylinder (figure 3.7 a).

Thereto it subdivides that part of the small cylinder, that lies within the object, into small subvolumes. The number of small subvolumes ($n_s$) is a hardcoded parameter in the code and set to 100. It can however easily be changed if desired. In doing so, it has to be kept in mind that by lowering this number, the length of each subvolume increases, whereby the accuracy of the method is lowered. As the savings of computing time are minor, it is advised against choosing this quantity too low. Furthermore it has to meet the condition of being an even number, due to a property of the coding as discussed below.

The second factor that has an effect on the performance of the routine is the integration cylinder radius itself. It has to be small in proportion to the size of the object, which is specified by its cut-off radius. For the preset parameters, the lowest possible mass is that of a minimal subhalo, $5 \times 10^8 M_\odot$, which results in a least cut-off radius of around $5.1 \times 10^{20}$m at zero redshift and $6.9 \times 10^{19}$m at redshift 10 (compare equation 2.45). As mentioned in section 3.1, the small cylinder radius is preset to $10^{17}$m. This ensures a sufficient precision over a wide range of redshifts.

The total length of the relevant small cylinder part results from Pythagoras' theorem (figure 3.7 b) as

$$l_{rel.cyl.} = 2 \times \sqrt{r_{200}^2 - r^2} \tag{3.5}$$

Thanks to the spherical symmetry of the NFW density profile (see section 2.2), there are throughout two small subvolumes which are identical, if the number of small subvolumes is chosen even (figure 3.7 c). This fact allows to cut down on the calculational complexity. Only the subvolumes on one side of the symmetry axis are evaluated regarding their mass content. The total mass share of the object within the small cylinder is then given as twice the accumulated mass of these subvolumes.

The mass evaluation proceeds likewise for all integration subvolumes.

First of all, the code determines the density radius, $r_\rho$, which assigns the appropriate density to the subvolume. For this purpose the centre of the subvolume is opted. The density radius of subvolume number n follows also as (figure 3.7 d)

$$r_\rho(n) = \sqrt{r^2 + (\frac{2n-1}{2(n_s)} l_{rel.cyl.})^2} \tag{3.6}$$

where n $\in [1, \frac{n_s}{2}]$.

Subsequently, the density, $\rho(r_\rho)$, is computed according to equation (2.44). The respective mass is simply this density times the subvolume, which is determined by its radius ($\equiv$ integration cylinder radius) and its lenght ($= \frac{l_{rel.cyl.}}{n_s}$).

Thus the program is able to assess, separately for each large simulation subcylinder, all mass within the respective integration cylinder due to (sub)halos.

If necessary, these mass densities get amended by the appropriate background densities (compare section 2.5).

In a last step the code calculates the ratio of actual mass density to expected mass density, $\frac{\rho(n)}{\Omega(z_n)\rho_{crit}(z_n)}$, for each subcylinder and outputs it into a file.

# Chapter 4

# Results

As mentioned above, the program outputs information on the matter density ratio, $\frac{\rho(n)}{\Omega(z_n)\rho_{crit}(z_n)}$ (the actual over the expected matter density), for each lens plane along the line of sight.

The average matter density ratio, $< \frac{\rho}{\rho_{exp}} >$, for a line of sight is weighted by the expected masses of the individual subvolumes

$$< \frac{\rho}{\rho_{exp}} >= \frac{\sum_{n=1}^{n_l} M_{exp}(n) \frac{\rho(n)}{\Omega(z_n)\rho_{crit}(z_n)}}{\sum_{n=1}^{n_l} M_{exp}(n)}, \tag{4.1}$$

where $M_{exp}(n)$ is given by

$$M_{exp}(n) = 2\pi(radius)^2 d_n \Omega(z_n)\rho_{crit}(z_n) \tag{4.2}$$

and $n_l$ is the number of lens planes inserted in the simulation volume.

According to the Press & Gunn approximation, which is conventionally adopted when calculating the probability of microlensing for a cosmologically distant source, the lensing objects are uniformly and randomly distributed in the intervening space with a constant comoving density. This is equipollent to $< \frac{\rho}{\rho_{exp}} > \equiv 1$ for all lines of sight.

Thus when plotting the cumulative fraction of the lines of sight with an average matter density ratio up to a certain value (cumulative distribution function, hereafter CDF), the Press & Gunn assumption predicts a step function at 1.

Using the output of my program, I tested this supposition for varying configurations. If not stated otherwise, the data for each setting consists of 1000 simulations, carried out applying the preset parameters.

## 4.1 Dependency on redshift

Intuitively the spread of the density distribution should be larger for low-redshift light sources, as the scatter in the number of dark halos along the line of sight is expected to be far in excess of the one for a light source at high cosmological distance.

This property can in fact be seen in all simulations for consistent settings (compare figure 4.1).

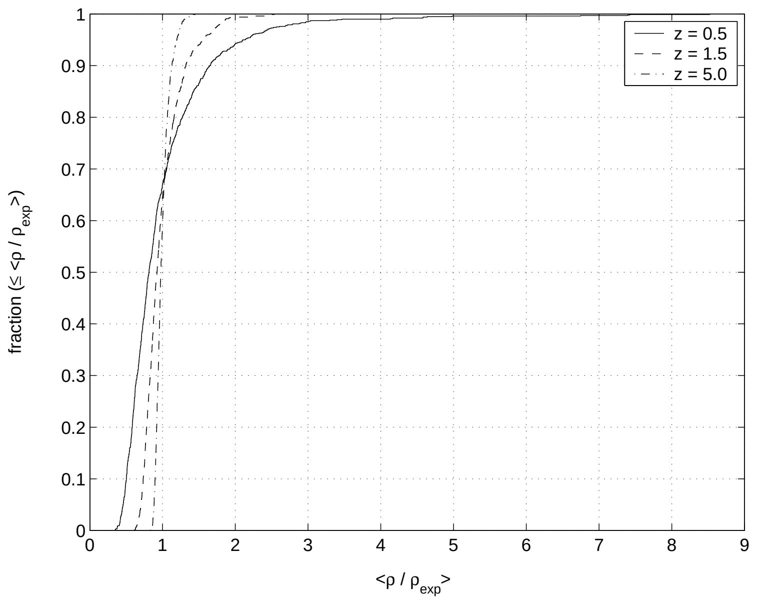


Figure 4.1: CDF of the average matter density ratio, $<\frac{\rho}{\rho_{exp}}>$, along the line of sight to a light source at z = 0.5, z = 1.5 and z = 5.0 using the Press-Schechter mass function for halo production (no substructure generation).

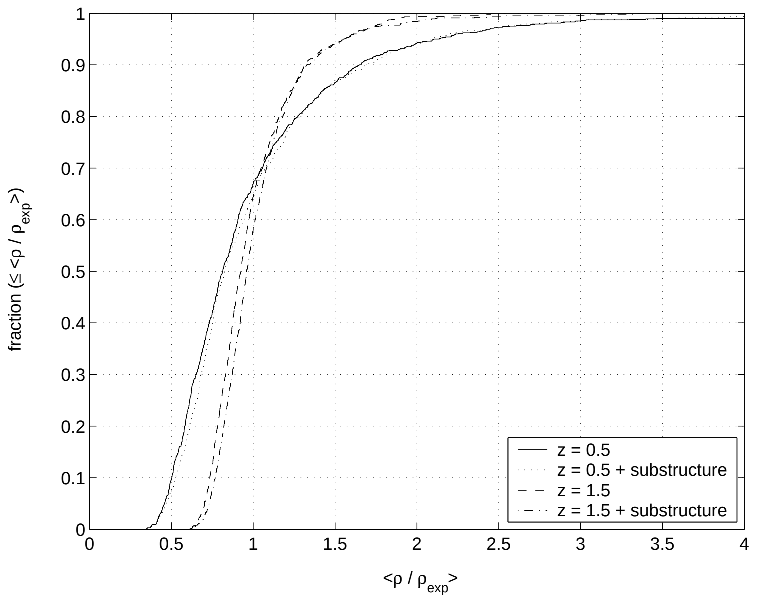


Figure 4.2: Zoom in on the CDF of the average matter density ratio, $<\frac{\rho}{\rho_{exp}}>$, along the line of sight to a light source at z = 0.5 and at z = 1.5 using the Press-Schechter mass function for halo production with and without generation of substructure.

## 4.2 Dependency on substructure

The effect of substructure on the spread of the average matter density plays a minor role compared to the dependency on redshift (see figure 4.2). The existence of subhalos seems to smooth the matter density slightly, in agreement with naive expectations, since it causes a partial redistribution of the parent halo mass which for a NFW profile is mostly concentrated in the centre (see section 2.2 and figure 2.4).

For increasing redshifts this effect seems to become somewhat more important. This is in accordance with the fact that the average subhalo mass fraction for a constant parent mass is rising with redshift (see figure 2.3). For high redshifts ($z \sim 5$) however, there can't be attached any more importance to this as the absolute effect of mass clustering in subhalos becomes negligible compared to the smoothing effect of increased redshift.

## 4.3 Dependency on clustering

To analyse the impact clustering has on the spread in average matter density, I compare simulations using the N-body data directly with simulations using the Press-Schechter mass function while following the GIF data mass fraction (see section 2.5) for concordant parameter settings.

Assuming that the N-body mass function and the Press-Schechter mass function are nearly alike, the only difference between the two scenarios is that the haloes from the GIF data are spatially clustered while the haloes generated by using the Press-Schechter mass function are randomly distributed.

As shown in figure 4.3 the provision for halo-halo clustering doesn't play an important role for the average matter density distribution. It causes a minor amplification of the dispersion observed for randomly distributed haloes at the respective redshift. The importance of this effect decreases with increasing redshift.

## 4.4 Dependency on mass fraction

The question which fraction of the total expected mass along the line of sight actually is clustered in halos within the given mass range has already been discussed in section 2.5.

As there are papers in which computations are carried out assuming all matter in the universe to be clustered in galaxy-sized objects, it is interesting to look into the outcomes of this scenario as well.

There is yet another reason why these data are significant. When restricting the halo mass range and following the mass fraction predicted by the mass function (Press-Schechter or GIF N-body), all mass in haloes outside this mass range becomes smoothly distributed as a background density. Thus for the preset parameters at redshifts $\geq 4$ less than 10 percent of the total mass gets clustered in halos (compare figure 2.6). This effect produces an increased density equalisation with redshift.

To abandon the concept of having a background density and thereby forcing all mass to cluster within the mass range causes however an opposite impact on the average matter density distribution. Analysing the findings of simulations without background density can therefore help to assess an upper limit on the density dispersion with redshift.

Figure 4.4 illustrates the important role the mass fraction variable plays. As anticipated, the spread intensifies for a larger mass fraction clustered in halos at fixed redshift.

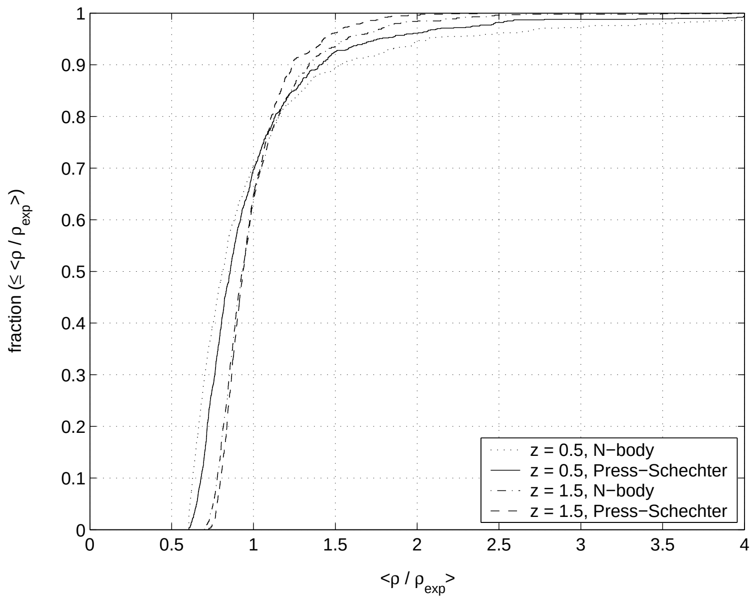


Figure 4.3: Zoom in on the CDF of the average matter density ratio, $< \frac{\rho}{\rho_{exp}} >$, along the line of sight to a light source at z $= 0.5$ and at z $= 1.5$ using N-body assembling and simulations applying the Press-Schechter mass function with a mass fraction following the GIF data (no generation of substructure).

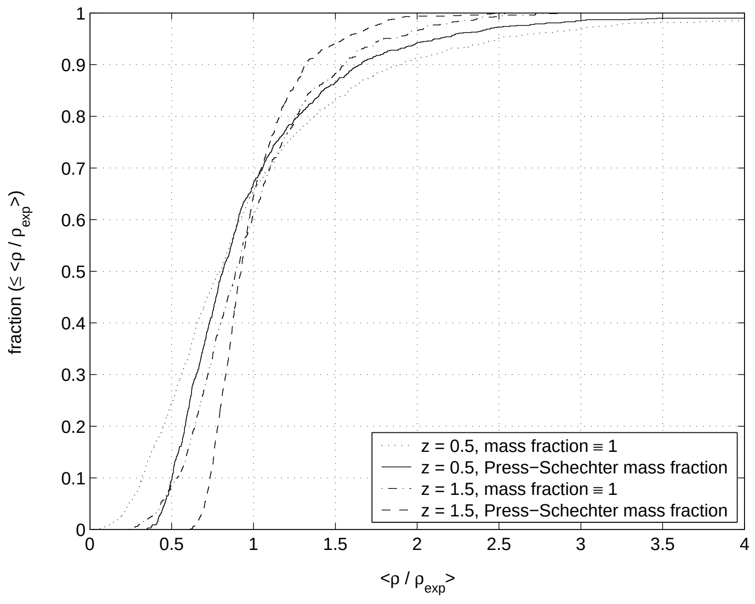


Figure 4.4: Zoom in on the CDF of the average matter density ratio, $< \frac{\rho}{\rho_{exp}} >$, along the line of sight to a light source at z $= 0.5$ and at z $= 1.5$ using simulations applying the Press-Schechter mass function and its mass fraction and simulations applying the Press-Schechter mass function with all matter clustered in halos within the preset mass range (no generation of substructure).

| Scenario | z = 0.5 | | z = 1.5 | |
|---|---|---|---|---|
| | median | mean | median | mean |
| Press-Schechter (Press-Schechter mass fraction) | 0.8135 | 1.0014 | 0.9241 | 0.9912 |
| Press-Schechter + substructure (Press-Schechter mass fraction) | 0.8250 | 1.0088 | 0.9597 | 1.0297 |
| Press-Schechter (N-body mass fraction) | 0.8577 | 0.9989 | 0.9349 | 0.9944 |
| N-body | 0.8106 | 1.0322 | 0.9321 | 1.0170 |
| Press-Schechter (mass fraction $\equiv 1$) | 0.7925 | 1.0031 | 0.8903 | 0.9772 |

Table 4.1: The median and mean values of the average matter density, $< \rho/\rho_{exp} >$, for different scenarios at redshifts z = 0.5 and z = 1.5. Each set of data consists of 1000 simulations using the preset parameters.

Table 4.1 shows an overview of the median and mean values of the average matter density, $< \rho/\rho_{exp} >$, computed for the configurations used to create figures 4.1 - 4.4.

As expected for a large sample, the mean value is very close to unity for all scenarios. The median value however exhibits a negative-only variation. This property is already noticeable in figures 4.1 - 4.4. At least about 60 percent of the simulations for any chosen scenario produce an average matter density lower than the expected one.

Just like the spread in the average matter density distribution, its median value displays a strong dependency on redshift.

The existence of substructure produces a slight approach of the median to the expected value of unity. This effect is however negligible when comparing with the variation due to redshift.

In contrast to the performance of the spread in matter density distribution due to clustering, the dependency of the median on clustering is only and highly distinct for light sources at low redshift. For z = 1.5, the median values are already nearly identical.

The magnitude of the mass fraction affects the median value in the same manner as it affects the distribution spread (see section 4.4). The importance of this effect is however smaller.

# Chapter 5

# Conclusions

Using Monte-Carlo techniques, I have developed a code that is able to simulate diverse models of universes filled with large-scale structure and with the help of those to determine the integrated matter distribution along the line of sight to a pointlike light source at arbitrary cosmological distance.

I compared the results from my code to the predictions of the Press-Gunn approximation which is conventionally adopted when carrying out microlensing calculations. According to the Press-Gunn approximation lensing objects are uniformly and randomly distributed throughout the universe.

As anticipated, the discrepancy between the assumption and the simulations is especially at low-redshift distinct.

The distribution of the average matter density evaluated by the code shows a broad spread as well as a negative-only variation of its expected median.

While the effects of halo substructure can overall be neglected at any redshift, halo-halo clustering shows a substantial impact on the matter distribution along the line of sight to sources at more modest distance ($z < 1$).

Another factor causing deviations is the magnitude of the mass fraction clustered in halos. This property suggests a mass resolution problem. The spread in the average matter density distribution is heavily dependent on the extend of the mass range within which halos are allowed to cluster. As the used Press-Schechter parent halo mass function by Sheth & Tormen has a lower resolution limit of $\sim 10^{11}M_{\odot}$ (Reed et al., 2003), a more powerful mass function formalism is needed in order to assess data unaffected of the smoothing feature of a background density.

Since the results indicate a severe deviation from the Press-Gunn approximation even when allowing the existence of a balancing background density, more realistic matter distributions along the line of sight have to be adopted in microlensing studies and their impact on current constraints on the mass function, velocity dispersion and cosmological density of the microlenses has to be evaluated.

To address this potentially serious problem, the output of the code described in this paper will be implemented into the microlensing code developed by Zackrisson and Bergvall (2003).

In addition, the strong redshift dependency of the spread in the mean density distribution along a line of sight to light sources at cosmological distances opens the exciting possibility to test the notion of microlensing-induced long-term variability in quasars by checking quasar samples on decreasing variability dispersion with increasing redshift .

# Acknowledgements

I would like to thank my supervisor Erik Zackrisson for his help and encouragement throughout the whole project.

In addition, I am thankful for the support consistently provided by Nils Bergvall and Kjell Olofsson, Uppsala University. I am also grateful for the hospitality of the Department of Astronomy and Space Physics at the Uppsala University.

# Bibliography

Couchman, H. M. P., Thomas, P. A., and Pearce, F. R.: 1995, *ApJ* **452**, 797

Dalcanton, J. J., Canizares, C. R., Granados, A., Steidel, C. C., and Stocke, J. T.: 1994, *ApJ* **424**, 550

Diaferio, A., Kauffmann, G., Colberg, J. M., and White, S. D. M.: 1999, *MNRAS* **307**, 537

Eisenstein, D. J. and Hu, W.: 1998, *ApJ* **496**, 605

Eisenstein, D. J. and Hu, W.: 1999, *ApJ* **511**, 5

Gao, L., White, S. D. M., Jenkins, A., Stoehr, F., and Springel, V.: 2003, *Microlensing of Gamma Ray Bursts by Stars and MACHOs*, astro-ph/0311569

Hawkins, M. R. S.: 1996, in *IAU Symp. 173: Astrophysical Applications of Gravitational Lensing*, pp 291–+

Hawkins, M. R. S.: 2001, *ApJ* **553**, L97

Hawkins, M. R. S. and Taylor, A. N.: 1997, *ApJ* **482**, L5+

Jenkins, A., Frenk, C. S., White, S. D. M., Colberg, J. M., Cole, S., Evrard, A. E., Couchman, H. M. P., and Yoshida, N.: 2001, *MNRAS* **321**, 372

Kauffmann, G., Colberg, J. M., Diaferio, A., and White, S. D. M.: 1999a, *MNRAS* **303**, 188

Kauffmann, G., Colberg, J. M., Diaferio, A., and White, S. D. M.: 1999b, *MNRAS* **307**, 529

Kravtsov, A. V., Berlind, A. A., Wechsler, R. H., Klypin, A. A., Gottlöber, S., Allgood, B., and Primack, J. R.: 2004, *ApJ* **609**, 35

Marani, G. F., Nemiroff, R. J., Norris, J. P., Hurley, K., and Bonnell, J. T.: 1999, *ApJ* **512**, L13

Metcalf, R. B. and Silk, J.: 1999, *ApJ* **519**, L1

Mitchell, J. L., Keeton, C. R., Frieman, J. A., and Sheth, R. K.: 2004, *Robust cosmological constraints from gravitational lens statistics*, astro-ph/0401138

Navarro, J. F., Frenk, C. S., and White, S. D. M.: 1997, *ApJ* **490**, 493

Oguri, M. and Lee, J.: 2004, *MNRAS* **355**, 120

Ostriker, J. P. and Peebles, P. J. E.: 1973, *ApJ* **186**, 467

Ostriker, J. P., Peebles, P. J. E., and Yahil, A.: 1974, *ApJ* **193**, L1

Press, W. H. and Gunn, J. E.: 1973, *ApJ* **185**, 397

Reed, D., Gardner, J., Quinn, T., Stadel, J., Fardal, M., Lake, G., and Governato, F.: 2003, *MNRAS* **346**, 565

Sanders, R. H.: 1986, *MNRAS* **223**, 539

Schild, R. E.: 1996, *ApJ* **464**, 125

Seljak, U. and Holz, D. E.: 1999, *A&A* **351**, L10

Sheth, R. K., Mo, H. J., and Tormen, G.: 2001, *MNRAS* **323**, 1

Sheth, R. K. and Tormen, G.: 1999, *MNRAS* **308**, 119

Surpi, G., Refsdal, S., and Helbig, P., *Quasar variability due to large-source microlensing by a cosmologically distributed lens population*, unpublished

Tegmark, M., Strauss, M. A., and Blanton, M. R.: 2004, *Phys. Rev. D* **69(10)**, 103501

Thomas, P. A., Pearce, F. H., Couchman, H. M. P., Colberg, J. M., White, S. D. M., Jenkins, A. R., and Frenk, C. S.: 1996, in *ASP Conf. Ser. 94: Mapping, Measuring, and Modelling the Universe*, pp 101–+

Tubbs, A. D. and Sanders, R. H.: 1979, *ApJ* **230**, 736

Valtonen, M. J.: 1980, *ApJ* **236**, 750

van den Bosch, F. C., Tormen, T., and Giocoli, C.: 2004, *The mass function and average mass-loss rate of dark matter subhaloes*, astro-ph/0409201

Wang, Y.: 1999, *ApJ* **525**, 651

Wyithe, J. S. B. and Turner, E. L.: 2002, *ApJ* **567**, 18

Yano, T., Nagashima, M., and Gouda, N.: 1996, in *Cosmological Constant and the Evolution of the Universe*, pp 313–+

Zackrisson, E. and Bergvall, N.: 2003, *A&A* **399**, 23

# Appendix A

# The code

```
function mat_dist;
% generates a simulation volume and calculates the matter
% density along the line of sight to a light source at z_QSO


% **************************** %
%      hardcoded parameters    %
% **************************** %

c                    = 3.e8;        % speed of light [m/s]
G                    = 6.67259e-11; % grav. constant [m^3/kg/s^2]
Pc                   = 3.0857e16;   % Pc in m
MPc                  = 3.0857e22;   % MPc in m
M_sun                = 1.9899e30;   % solar mass [kg]
theta27              = 1.0104;      % T(CMB)=theta27*2.7K
nr_of_lens_planes    = 30;          % number of lens planes
radius               = 5e22;        % simulation cylinder radius
radius_small         = 1e17;        % radius of the small cylinder
bin_steps            = 80;          % nr of steps in massinterval
nr_of_small_volumes  = 100;         % nr of small volumes in a (sub)halo
                                    % when integrating
ipsi                 = 1;           % initial power spectrum index
p                    = 0.3;         % halo mass function fitting parameter
A                    = 0.3222;      % halo mass function fitting parameter
q                    = 0.707;       % halo mass function fitting parameter
n_eff                = -2.15;       % effective slope of the CDM power spectrum
beta                 = 0.13;        % subhalo mass function fitting parameter
rep                  = 1000;        % number of simulations
show_plot            = 0;           % boolean
                                    % 1 => subvolume plot
plot_vol             = 1;           % index of subvolume cylinder to plot
show_sub_plot        = 0;           % boolean
                                    % 1 => halo & subhalo plot


% **************************** %
%          variables           %
% **************************** %

% input parameters
% outfile            : file which will contain the output data
% z_QSO              : redshift of the light source
% lambda             : cosmological constant today
% omega_b              : omega in baryons today
% omega_c              : omega in CDM today
% hubble               : Hubble constant [km/MPc/s]
% M_min                : minimum halo mass [M_sun]
% M_max                : maximum halo mass [M_sun]
% n_body               : boolean
%                        0 => create halos applying Press-Schechter mass funct.
%                        1 => use GIF-project halo catalogues
% mass_in_halos      : 0 => all mass clustered in halos
%                        1 => halos outside the mass interval contribute to a
%                        background density (following the mass function)
%                        2 => halos outside the mass interval contribute to a
%                        background density (following the mass function adjusted
%                        by a factor of 2)
%                        3 => halos outside the mass interval contribute to a
%                        background density (following the GIF data fit)
% substruc              : boolean
%                        0 => no substructure in halos
%                        1 => create subhalos
% M_sub_min            : minimum subhalo mass [M_sun]
% M_sub_max            : maximum subhalo mass [M_sun]

% main program
% M_int                : logarithmically even spread masses within given mass
%                        range (length(M_int)=bin_steps) [kg]
% z_nb(nb)             : redshift of the nbth n-body simulation volume
% z(n)                 : redshift of the nth lens plane
% dc(n)                : length of the nth subvolume cylinder [m]
% rho_critical         : critical density of the universe
% rho_mass(n)          : expected density for the nth subvolume [kg/m^3]
% v(n)                 : subvolume belonging to the nth lens plane [m^3]
% M_c(n)               : characteristic mass scale used in the halo mass
%                        function for the nth subvolume [kg]
% delta_c_z(n)         : delta_c(z), critical value of the initial overdensity
%                        which is required for collapse
% mass(n)              : expected mass in halos in the nth subvolume [kg]
% bg_density(n)        : background density in the nth subvolume [kg/m^3]
% fmax(n)              : maximum of the halo mass function at z(n)
% mass_sub(n)          : mass in halos in the nth subvolume [M_sun]
% mass_ran(n,l)        : mass of the lth halo in the nth subvolume [M_sun]
% h(n,l)               : height of the lth halo in the nth subvolume [m]
% r(n,l)               : radius of the lth halo in the nth subvolume [m]
% phi(n,l)             : angle of the lth halo in nth the subvolume [rad]
% dc_nb(nb)            : length of the nbth subvolume cylinder [m]
% density_nb(nb)       : background density in the nbth subvolume [kg/m^3]
% mass_nb(nb,l)        : mass of the lth halo in the nbth subvolume [M_sun]
% zeta(nb,l)           : height of the lth halo in the nbth subvolume [m]
% rho(nb,l)            : radius of the lth halo in the nbth subvolume [m]
% phi_nb(nb,l)         : angle of the lth halo in the nbth subvolume [rad]
% total_length_nb(nb) : total lenght of the sim. cylinder until z_nb(nb-1) [m]
% total_length(n)      : total length of the sim. cylinder until z(n-1) [m]
% M2M_c_s(n,l)         : mass ratio of the lth halo in the nth subvolume to
%                          the characteristic non-linear mass
% mass_substr(n,l)     : mass in substructure of lth halo in nth subvolume
% cut_off(n,l)         : cut-off radius of the lth halo in the nth subvolume
% sub_halo(n,l)        : boolean
%                          0 => halo not relevant for substructure generation
%                          1 => halo contains relevant substructure
% mass_ran_sub(n,m,1) : mass of the (1-1)th subhalo in the mth relevant halo
%                       in the nth subvolume
% mass_sub_struc      : mass in subhalos
% conc_par            : concentration parameter
% x_sub(n,m,s)        : x-coordinate of the (s-1)th subhalo in themth
%                       relevant halo in the nth subvolume
% y_sub(n,m,s)        : y-coordinate of the (s-1)th subhalo in the mth
%                       relevant halo in the nth subvolume
% z_sub(n,m,s)        : z-coordinate of the (s-1)th subhalo in the mth
%                       relevant halo in the nth subvolume
% r_sub(n,m,s)        : radius of the (s-1)th subhalo in mth relevant halo
%                       in nth subvolume (polar coordinates)
% cut_off_sub(n,m,s)  : cut-off radius of the (s-1)th subhalo in the mth
%                       relevant halo in the nth subvolume
% mass_small_cyl_in_sub(n) : mass in substructure inside a small cylinder
%                            through the nth subvolume
% mass_small_cyl(n)        : mass in halos inside a small cylinder
%                            through the nth subvolume
% int_cyl_vol(n)           : volume of a small cylinder through the nth
%                            subvolume
% density_small_cyl(n)     : density of a small cylinder through the nth
%                            subvolume
% dens_quot(n)             : density quotient of a small cylinder through
%                            the nth subvolume

% sigma_d2            : square variance of the density field at epoch z in
%                       linear perturbation theory



% ************************** %
%        main program        %
% ************************** %


% get input parameters from user
 % .............................
[outfile,z_QSO,lambda,omega_b,omega_c,hubble,M_min,M_max,n_body,...
        mass_in_halos,substruc,M_sub_min,M_sub_max] = input_parameters;

omega_m = omega_b+omega_c;

% convert masses to kg
 % ...................
M_min = M_min*M_sun;
M_max = M_max*M_sun;
M_sub_min = M_sub_min*M_sun;
M_sub_max = M_sub_max*M_sun;

M_int = logspace(log10(M_min),log10(M_max),bin_steps);

z_nb = [0.06 0.13 0.20 0.27 0.35 0.42 0.52 0.62 0.72 0.82 0.93 1.05 1.18 ...
        1.31 1.46 1.61 1.77 1.94 2.12 2.32 2.52 2.74 2.97 3.21 3.47 3.75 ...
        4.04 4.35 4.68 5.03 5.41 5.80 6.22 6.67 7.14 7.65 8.18 8.75 9.35 ...
        9.99 10.66 11.28];

z = zeros(1,nr_of_lens_planes);
dc = zeros(1,nr_of_lens_planes);
```

```
rho_mass = zeros(1,nr_of_lens_planes);
v = zeros(1,nr_of_lens_planes);

% calculate the redshifts of the lens planes
% .....................................
for n = 1:nr_of_lens_planes
    z(n) = n * (z_QSO / (nr_of_lens_planes + 1));
end

% calculate relevant cosmological distances
% .....................................
dc = calc_distances(z,omega_m,lambda,c,hubble,MPc,nr_of_lens_planes);

% calculate the density given by the used cosmology for each lens plane
% ..............................................................
rho_critical = (3*(1000*hubble/MPc)^2)/(8*pi*G);
omega_m_z = omega_m.*(1+z).^3./(lambda+(1-lambda-omega_m).*(1+z).^2+...
    omega_m.*(1+z).^3);
lambda_z = lambda./(lambda+(1-lambda-omega_m).*(1+z).^2+omega_m.*(1+z).^3);
rho_critical_z = rho_critical.*(lambda+(1-lambda-omega_m).*(1+z).^2+...
    omega_m.*(1+z).^3);
rho_mass = omega_m_z.*rho_critical_z;

% generate subvolumes
% ..................
v = generate_subvolume(dc,radius);

% calculate the critical value of the initial overdensity which is required
% for spherical collapse at z
for n = 1:nr_of_lens_planes
    delta_c_z(n) = 0.15*(12*pi)^(2/3)*(omega_m_z(n))^0.0055;
end

z_eq = 2.50*10^4*omega_m*(hubble/100)^2*(theta27)^(-4);
% redshift of matter-radiation equality

k_eq = 7.46*10^(-2)*omega_m*(hubble/100)^2*(theta27)^(-2)/MPc;

b1 = 0.313*(omega_m*(hubble/100)^2)^(-0.419)*(1+(0.607*(omega_m*(hubble/...
    100)^2)^0.674));
b2 = 0.238*(omega_m*(hubble/100)^2)^0.223;
z_d = 1291*((omega_m*(hubble/100)^2)^0.251)/(1+0.659*(omega_m*(hubble/...
    100)^2)^0.828)*(1+b1*(omega_b*(hubble/100)^2)^b2);
% redshift at which the baryons are released from the compton drag of the
%photons

R_eq = 31.5*omega_b*(hubble/100)^2*theta27^(-4)*(z_eq/10^3)^(-1);

R_d = 31.5*omega_b*(hubble/100)^2*theta27^(-4)*(z_d/10^3)^(-1);

s = 2/(3*k_eq)*sqrt(6/R_eq)*log((sqrt(1+R_d)+sqrt(R_d+R_eq))/(1+sqrt(R_eq)));
% sound horizon at drag epoch

k_silk = 1.6*(omega_b*(hubble/100)^2)^0.52*(omega_m*(hubble/100)^2)^0.73*...
    (1+(10.4*omega_m*(hubble/100)^2)^(-0.95))/MPc;
% silk damping scale

a1 = (46.9*omega_m*(hubble/100)^2)^0.67*(1+(32.1*omega_m*(hubble/100)^2)...
    ^(-0.532));
a2 = (12.0*omega_m*(hubble/100)^2)^0.424*(1+(45.0*omega_m*(hubble/100)^2)...
    ^(-0.582));
alpha_c = a1^(-omega_b/omega_m)*a2^(-omega_b/omega_m)^3;
% CDM suppression

b1 = 0.944*(1+(458*omega_m*(hubble/100)^2)^(-0.708))^(-1);
b2 = (0.395*omega_m*(hubble/100)^2)^(-0.0266);
beta_c = (1+b1*((omega_c/omega_m)^b2-1))^(-1);
% shift in CDM log

y_d = (1+z_eq)/(1+z_d);

Gfunc = y_d*(-6*sqrt(1+y_d)+(2+3*y_d)*log((sqrt(1+y_d)+1)/(sqrt(1+y_d)-1)));

alpha_b = 2.07*k_eq*s*(1+R_d)^(-3/4)*Gfunc;
% baryon suppression

beta_b = 0.5+omega_b/omega_m+(3-2*omega_b/omega_m)*sqrt((17.2*omega_m*...
    (hubble/100)^2)^2+1);
% baryon envelope shift

beta_node = 8.41*(omega_m*(hubble/100)^2)^0.435;
% node shift parameter

for n = 1:nr_of_lens_planes
    D_1(n) = (1+z_eq)*(1+z(n))^(-1)*5*omega_m_z(n)/2*(omega_m_z(n)^(4/7)-...
        lambda_z(n)+(1+omega_m_z(n)/2)*(1+lambda_z(n)/70))^(-1);
end
D_10 = (1+z_eq)*5*omega_m/2*(omega_m^(4/7)-lambda+(1+omega_m/2)*(1+lambda/...
    70))^(-1);
% growth function

m = ipsi-1;
if lambda == 0
    delta_h = 1.95*10^(-5)*omega_m^(-0.35-0.19*log(omega_m)-0.17*m)*exp(-m-...
        0.14*m^2);
else
    delta_h = 1.94*10^(-5)*omega_m^(-0.785-0.05*log(omega_m))*exp(-0.95*m-...
        0.169*m^2);
end
% amplitude of perturbations on the horizon scale today

% calculate the characteristic mass scale for each lens plane
% ......................................................
for n = 1:nr_of_lens_planes
    got_M_c = 0;
    M(n) = 5e42;
    while got_M_c == 0
        sigma_d2 = quadl('sigma_integral',0,1e-9,[],[],M(n),omega_m,...
            omega_c,omega_b,hubble,c,ipsi,rho_critical_z(n),D_1(n),D_10,...
            z_eq,k_eq,k_silk,s,alpha_c,beta_c,alpha_b,beta_b,beta_node,...
            delta_h);
        if (sqrt(sigma_d2)/delta_c_z(n)) < 0.93
            M(n) = 0.2*M(n);
        elseif (sqrt(sigma_d2)/delta_c_z(n)) > 1.07
            M(n) = 1.8*M(n);
        elseif (sqrt(sigma_d2)/delta_c_z(n)) < 0.98
```

```
            M(n) = 0.5*M(n);
        elseif (sqrt(sigma_d2)/delta_c_z(n)) > 1.02
            M(n) = 1.5*M(n);
        elseif (sqrt(sigma_d2)/delta_c_z(n)) < 0.996
            M(n) = 0.95*M(n);
        elseif (sqrt(sigma_d2)/delta_c_z(n)) > 1.004
            M(n) = 1.05*M(n);
        elseif (sqrt(sigma_d2)/delta_c_z(n)) < 0.998
            M(n) = 0.99*M(n);
        elseif (sqrt(sigma_d2)/delta_c_z(n)) > 1.002
            M(n) = 1.01*M(n);
        else
            got_M_c = 1;
        end
    end
    M_c(n) = M(n);
end

for i = 1:rep

    i

    rand('state',sum(100*clock));

    mass = zeros(1,nr_of_lens_planes);
    bg_density = zeros(1,nr_of_lens_planes);
    fmax = zeros(1,nr_of_lens_planes);
    mass_sub = zeros(1,nr_of_lens_planes);
    mass_small_cyl_in_sub = zeros(1,nr_of_lens_planes);
    mass_small_cyl = zeros(1,nr_of_lens_planes);
    int_cyl_vol = zeros(1,nr_of_lens_planes);
    density_small_cyl = zeros(1,nr_of_lens_planes);
    dens_quot= zeros(1,nr_of_lens_planes);
    mass_ran = zeros(nr_of_lens_planes,3000);
    h = zeros(nr_of_lens_planes,3000);
    r = zeros(nr_of_lens_planes,3000);
    phi = zeros(nr_of_lens_planes,3000);
    cut_off = zeros(nr_of_lens_planes,3000);
    sub_halo = zeros(nr_of_lens_planes,3000);

    if n_body == 0
        % halo production via Press-Schechter mass function chosen

        for n = 1:nr_of_lens_planes

            % calculate mass in dark halos in each subvolume
            % ..........................................
            if mass_in_halos == 1
                mass(n) = mass_in_interval(bin_steps,M_c(n),v(n),n,M_int,...
                    rho_critical_z(n),M_min,M_max,c,p,A,q,delta_c_z(n),...
                    n_eff,MPc);
                if mass(n) > (rho_mass(n)*v(n))
                    mass(n) = rho_mass(n)*v(n)/M_sun;
                    bg_density(n) = 0;
                else
                    bg_density(n) = (rho_mass(n)*v(n)-mass(n))/v(n);
                    mass(n) = mass(n)/M_sun;
                end
        elseif mass_in_halos == 2
            mass(n) = mass_in_interval_corr(bin_steps,M_c(n),v(n),n,...
                M_int,rho_critical_z(n),M_min,M_max,c,p,A,q,...
                delta_c_z(n),n_eff,MPc);
            bg_density(n) = (rho_mass(n)*v(n)-mass(n))/v(n);
            mass(n) = mass(n)/M_sun;
        elseif mass_in_halos == 3
            mass(n) = mass_in_interval_gif(z(n),rho_mass(n),v(n));
            bg_density(n) = (rho_mass(n)*v(n)-mass(n))/v(n);
            mass(n) = mass(n)/M_sun;
        else
            mass(n) = rho_mass(n)*v(n)/M_sun;
        end

        % calculate maximum of the mass distribution
        % ........................................
        fmax(n) = getmax_mass_dist(M_c(n),n,M_int,rho_critical_z(n),...
            M_min,M_max,c,p,A,q,delta_c_z(n),n_eff,MPc);

        % set initial dark halo mass and number in subvolume zero ...
        % ......................................................
        mass_sub(n) = 0;
        l = 0;

        % ... and fill it up
        % .................
        while mass_sub(n) < (mass(n)-5*M_min/M_sun)
            l = l + 1;
            mass_ran(n,l) = halomass(M_sun,M_c(n),fmax,n,M_int,...
                rho_critical_z(n),M_min,M_max,c,p,A,q,delta_c_z(n),...
                n_eff,MPc);
            mass_sub(n) = mass_sub(n) + mass_ran(n,l);
            if mass_sub(n) > (mass(n)+5*M_min/M_sun)
                mass_sub(n) = mass_sub(n) - mass_ran(n,l);
                l = l - 1;
            else

                % generate coordinates for each dark halo
                % ....................................
                [h(n,l),r(n,l),phi(n,l)]  = generate_coordinates_rand...
                    (dc,n,radius);
            end
        end
    end

else
    % use the n-body simulation data

    % calculate index of the maximum relevant n-body redshift
    % ....................................................
    found_z = 0;
    max_z = 0;
    while found_z == 0
        max_z = max_z + 1;
        if z_nb(max_z) >= z(nr_of_lens_planes)
            found_z = 1;
        end
    end
```

```
% calculate subvolume lengths coresponding to the relevant n_body
% redshifts
% .........................................................
dc_nb = calc_distances(z_nb,omega_m,lambda,c,hubble,MPc,max_z);

mass_nb = zeros(max_z,10000);
phi_nb = zeros(max_z,10000);
rho = zeros(max_z,10000);
zeta = zeros(max_z,10000);
mass = zeros(1,max_z);
max_length = zeros(1,max_z);
density_nb = zeros(1,max_z);
X = zeros(1,max_z);
Y = zeros(1,max_z);
Z_l = zeros(1,max_z);
Z_u = zeros(1,max_z);

dc_nb_dum = dc_nb;

% cut subvolumes out of the n-body simulation cubes
% ..............................................
for nb = 1:max_z
    C = zeros(89000,4);
    mass_prov = zeros(10000,4);
    indat = ['nb',num2str(z_nb(nb),'%4.2f'),'.txt.'];
    C = load(indat);
    mass(nb) = 0;
    for i = 1:size(C,1)
        if C(i,1) ~= 0
            mass(nb) = mass(nb)+(M_sun*10^C(i,1));
        end
    end

    % converte the coordinates into not comoving and [m]
    % ..............................................
    C(:,2:4) = C(:,2:4).*MPc./(hubble/100*(1+z_nb(nb)));
    max_length(nb) = 141*MPc/(hubble/100*(1+z_nb(nb)));

    density_nb(nb) = mass(nb)/(max_length(nb))^3;

    if dc_nb_dum(nb) <= max_length(nb)
        % simulation subcylinder fits into the respective nb-cube

        rand_num = rand(1);
        X(nb) = rand_num*(max_length(nb)-2*radius)+radius;
        rand_num = rand(1);
        Y(nb) = rand_num*(max_length(nb)-2*radius)+radius;
        Z_l(nb) = 0;
        Z_u(nb) = Z_l(nb) + dc_nb_dum(nb);
        k = 1;
        for m = 1:size(C,1)
            if C(m,2) > (X(nb)-radius)
                if C(m,2) <= (X(nb)+radius)
                    if C(m,3) > (Y(nb)-radius)
                        if C(m,3) <= (Y(nb)+radius)
                            if C(m,4) <= Z_u(nb)
                                mass_prov(k,1:4) = C(m,1:4);
                        k = k+1;
                    end
                end
            end
        end
    end
end
mass_prov(:,2) = mass_prov(:,2)-X(nb);
mass_prov(:,3) = mass_prov(:,3)-Y(nb);
[mass_prov(:,2),mass_prov(:,3),mass_prov(:,4)] = cart2pol...
    (mass_prov(:,2),mass_prov(:,3),mass_prov(:,4));
l = 1;
i = 1;
while mass_prov(i,1) ~= 0
    if mass_prov(i,3) > radius
        mass_prov(i,1:4) = [0 0 0 0];
    else
        mass_nb(nb,l) = 10^mass_prov(i,1);
        phi_nb(nb,l) = mass_prov(i,2);
        rho(nb,l) = mass_prov(i,3);
        zeta(nb,l) = mass_prov(i,4);
        l = l+1;
    end
    i = i+1;
end
    else
% simulation subcylinder longer than its nb-cube

dc_nb_dum(nb) = dc_nb_dum(nb)-max_length(nb);
rand_num = rand(1);
X(nb) = rand_num*(max_length(nb)-2*radius)+radius;
rand_num = rand(1);
Y(nb) = rand_num*(max_length(nb)-2*radius)+radius;
Z_l(nb) = 0;
Z_u(nb) = max_length(nb);
k = 1;
for m = 1:size(C,1)
    if C(m,2) > (X(nb)-radius)
        if C(m,2) <= (X(nb)+radius)
            if C(m,3) > (Y(nb)-radius)
                if C(m,3) <= (Y(nb)+radius)
                    if C(m,4) <= Z_u(nb)
                        mass_prov(k,1:4) = C(m,1:4);
                        k = k+1;
                    end
                end
            end
        end
    end
end
mass_prov(:,2) = mass_prov(:,2)-X(nb);
mass_prov(:,3) = mass_prov(:,3)-Y(nb);
l = k;
if dc_nb_dum(nb) <= max_length(nb)
    % simulation subcylinder fits into the respective nb-cube

    rand_num = rand(1);
    X(nb) = rand_num*(max_length(nb)-2*radius)+radius;
```

```
         rand_num = rand(1);
         Y(nb) = rand_num*(max_length(nb)-2*radius)+radius;
         Z_l(nb) = 0;
         Z_u(nb) = dc_nb_dum(nb);
         for m = 1:size(C,1)
             if C(m,2) > (X(nb)-radius)
                 if C(m,2) <= (X(nb)+radius)
                     if C(m,3) > (Y(nb)-radius)
                         if C(m,3) <= (Y(nb)+radius)
                             if C(m,4) <= Z_u(nb)
                                 mass_prov(l,1:3) = C(m,1:3);
                                 mass_prov(l,4) = C(m,4)+...
                                     max_length(nb);
                                 l = l+1;
                             end
                         end
                     end
                 end
             end
         end
         mass_prov(k:(l-1),2) = mass_prov(k:(l-1),2)-X(nb);
         mass_prov(k:(l-1),3) = mass_prov(k:(l-1),3)-Y(nb);
     else
         % simulation subcylinder longer than its nb-cube

         dc_nb_dum(nb) = dc_nb_dum(nb)-max_length(nb);
         rand_num = rand(1);
         X(nb) = rand_num*(max_length(nb)-2*radius)+radius;
         rand_num = rand(1);
         Y(nb) = rand_num*(max_length(nb)-2*radius)+radius;
         Z_l(nb) = 0;
         Z_u(nb) = max_length(nb);
         for m = 1:size(C,1)
             if C(m,2) > (X(nb)-radius)
                 if C(m,2) <= (X(nb)+radius)
                     if C(m,3) > (Y(nb)-radius)
                         if C(m,3) <= (Y(nb)+radius)
                             if C(m,4) <= Z_u(nb)
                                 mass_prov(l,1:3) = C(m,1:3);
                                 mass_prov(l,4) = C(m,4)+...
                                     max_length(nb);
                                 l = l+1;
                             end
                         end
                     end
                 end
             end
         end
         mass_prov(k:(l-1),2) = mass_prov(k:(l-1),2)-X(nb);
         mass_prov(k:(l-1),3) = mass_prov(k:(l-1),3)-Y(nb);
         o = l;
         if dc_nb_dum(nb) <= max_length(nb)
             % simulation subcylinder fits into its nb-cube

             rand_num = rand(1);
             X(nb) = rand_num*(max_length(nb)-2*radius)+radius;
             rand_num = rand(1);
             Y(nb) = rand_num*(max_length(nb)-2*radius)+radius;
```

```
                Z_l(nb) = 0;
                Z_u(nb) = dc_nb_dum(nb);
                for m = 1:size(C,1)
                    if C(m,2) > (X(nb)-radius)
                        if C(m,2) <= (X(nb)+radius)
                            if C(m,3) > (Y(nb)-radius)
                                if C(m,3) <= (Y(nb)+radius)
                                    if C(m,4) <= Z_u(nb)
                                        mass_prov(o,1:3) = C(m,1:3);
                                        mass_prov(o,4) = C(m,4)+2*...
                                            max_length(nb);
                                        o = o+1;
                                    end
                                end
                            end
                        end
                    end
                end
                mass_prov(1:(o-1),2) = mass_prov(1:(o-1),2)-X(nb);
                mass_prov(1:(o-1),3) = mass_prov(1:(o-1),3)-Y(nb);
            end
        end

        % converte the cartesian halo coordinates into polar
        % coordinates respective to the simulation cylinder axis
        [mass_prov(:,2),mass_prov(:,3),mass_prov(:,4)] = cart2pol...
            (mass_prov(:,2),mass_prov(:,3),mass_prov(:,4));
        l = 1;
        i = 1;
        while mass_prov(i,1) ~= 0
            if mass_prov(i,3) > radius
                mass_prov(i,1:4) = [0 0 0 0];
            else
                mass_nb(nb,l) = 10^mass_prov(i,1);
                phi_nb(nb,l) = mass_prov(i,2);
                rho(nb,l) = mass_prov(i,3);
                zeta(nb,l) = mass_prov(i,4);
                l = l+1;
            end
            i = i+1;
        end
    end
    clear C mass_prov;
    pack;
end

% calculate the total length of the simulation cylinder until z(nb-1)
% .............................................................
total_length_nb(1) = 0;
for nb = 2:max_z
    total_length_nb(nb) = total_length_nb(nb-1)+dc_nb(nb-1);
end

for nb = 2:max_z
    zeta(nb,:) = zeta(nb,:)+total_length_nb(nb);
end

% calculate the total length of the simulation cylinder until z(n-1)
% ...................................................................
total_length(1) = 0;
for n = 2:nr_of_lens_planes
    total_length(n) = total_length(n-1)+dc(n-1);
end

% assign the halos to their proper nth subvolume
% ..........................................
nb = 1;
for n = 1:nr_of_lens_planes
    l = 1;
    if z(n) < z_nb(nb)
        i = 1;
        while mass_nb(nb,i) ~= 0
            if zeta(nb,i) <= (total_length(n)+dc(n))
                if zeta(nb,i) > total_length(n)
                    mass_ran(n,l) = mass_nb(nb,i);
                    phi(n,l) = phi_nb(nb,i);
                    r(n,l) = rho(nb,i);
                    h(n,l) = zeta(nb,i)-total_length(n);
                    l = l+1;
                end
            end
            i = i+1;
        end
    else
        for i = 1:size(mass_nb,2)
            if mass_nb(nb,i) ~= 0
                if zeta(nb,i) <= total_length_nb(nb+1)
                    if zeta(nb,i) > total_length(n)
                        mass_ran(n,l) = mass_nb(nb,i);
                        phi(n,l) = phi_nb(nb,i);
                        r(n,l) = rho(nb,i);
                        h(n,l) = zeta(nb,i)-total_length(n);
                        l = l+1;
                    end
                end
            end
        end
        nb = nb+1;
        for i = 1:size(mass_nb,2)
            if mass_nb(nb,i) ~= 0
                if zeta(nb,i) <= (total_length(n)+dc(n))
                    if zeta(nb,i) > total_length_nb(nb)
                        mass_ran(n,l) = mass_nb(nb,i);
                        phi(n,l) = phi_nb(nb,i);
                        r(n,l) = rho(nb,i);
                        h(n,l) = zeta(nb,i)-total_length(n);
                        l = l+1;
                    end
                end
            end
        end
    end
end

% determine the proper background density for the nth subvolume
% applying the GIF N-body mass fraction fit
```

```
        % ............................................................
        for n = 1:nr_of_lens_planes
            bg_density(n) = (1-(-1.408e-5*z(n)^5+0.00046552*z(n)^4-0.0065069...
                *z(n)^3+0.050636*z(n)^2-0.22559*z(n)+0.45099))*rho_mass(n);
        end
end

 if substruc ~= 0
        % substructure generation chosen

        M2M_c_s = zeros(nr_of_lens_planes,3000);
        subfrac = zeros(nr_of_lens_planes,3000);
        mass_substr = zeros(nr_of_lens_planes,3000);
        for n = 1:nr_of_lens_planes
            for l = 1:size(mass_ran,2)
                if mass_ran(n,l) ~= 0
                    M2M_c_s(n,l) = mass_ran(n,l)*M_sun/M_c(n);
                    % calculate mass fraction in substructure
                    % ......................................
                    subfrac(n,l) = 10^(sqrt(0.4*(log10(M2M_c_s(n,l))+5))-2.74);
                    % get the halo masses in substructure
                    % .................................
                    mass_substr(n,l) = subfrac(n,l)*mass_ran(n,l);
                    % reduce the halo masses by their fraction in substructure
                    % ......................................................
                    mass_ran(n,l) = (1-subfrac(n,l))*mass_ran(n,l);
                else
                    subfrac(n,l) = 0;
                end
            end
        end
 end

 % calculate the cut_off radii for all halos
 % .......................................
 cut_off = calc_cut_off_all(mass_ran,z,omega_m,hubble,Pc,...
        nr_of_lens_planes,omega_m_z);

 if substruc ~= 0
        % substructure generation chosen

        % check which halos lie inside a small integration cylinder through
        % the subvolumes
        % ............................................................
        sub_halo = check_sub_halo(cut_off,r,h,radius_small,nr_of_lens_planes);

        mass_ran_sub = zeros(nr_of_lens_planes,10,4000);
        x_sub = zeros(nr_of_lens_planes,10,4000);
        y_sub = zeros(nr_of_lens_planes,10,4000);
        z_sub = zeros(nr_of_lens_planes,10,4000);
        r_sub = zeros(nr_of_lens_planes,10,4000);
        h_sub = zeros(nr_of_lens_planes,10,4000);
        cut_off_sub = zeros(nr_of_lens_planes,10,4000);

        % generate subhalos in the halos that lie inside the small
        % integration cylinder
        % ......................................................
        for n = 1:nr_of_lens_planes
```

```
            m = 0;
            for l = 1:size(cut_off,2)
                if sub_halo(n,l) == 1
                    m = m + 1;
                    mass_ran_sub(n,m,1) = l;
                    x_sub(n,m,1) = l;
                    y_sub(n,m,1) = l;
                    z_sub(n,m,1) = l;
                    r_sub(n,m,1) = l;
                    h_sub(n,m,1) = l;
                    s = 1;
                    mass_sub_struc = 0;
                    conc_par = 124/(1+z(n))*(mass_ran(n,l)*hubble/100)...
                        ^(-0.084);
                    while mass_sub_struc < (mass_substr(n,l)-5*M_sub_min/M_sun)
                        s = s + 1;
                        mass_ran_sub(n,m,s) = subhalomass(M_sun,M_sub_min,...
                            M_sub_max,hubble,M2M_c_s(n,l),subfrac(n,l),beta,...
                            mass_substr(n,l),mass_ran(n,l));
                        mass_sub_struc = mass_sub_struc + mass_ran_sub(n,m,s);
                        if mass_sub_struc > (mass_substr(n,l)+5*M_sub_min/...
                                M_sun)
                            mass_sub_struc = mass_sub_struc-mass_ran_sub(n,m,s);
                            s = s - 1;
                        else
                            % generate coordinates for each subhalo
                            % ....................................
                            [x_sub(n,m,s),y_sub(n,m,s),z_sub(n,m,s),r_sub...
                                    (n,m,s),h_sub(n,m,s)] = ...
                                generate_sub_coordinates_rand(conc_par,...
                                cut_off,r,n,l,h);
                        end
                    end
                end
            end
        end
        % calculate the cut-off radii for all subhalos
        % .........................................
        cut_off_sub = calc_cut_off_sub_all(mass_ran_sub,z,omega_m,hubble,Pc,...
            nr_of_lens_planes,omega_m_z);
        % calculate the mass in substructure inside the small cylinder through
        % the subvolumes
        % ..............................................................
        mass_small_cyl_in_sub = calc_mass_small_cyl_in_sub(radius_small,...
            cut_off_sub,r_sub,h_sub,mass_ran_sub,hubble,G,M_sun,z,omega_m,...
            dc,Pc,MPc,nr_of_lens_planes,nr_of_small_volumes,omega_m_z);
    end

    % calculate the mass in halos inside the small cylinder through the
    % subvolumes
    % ..............................................................
    mass_small_cyl = calc_mass_small_cyl(radius_small,cut_off,r,h,mass_ran,...
        hubble,G,M_sun,z,omega_m,dc,Pc,MPc,nr_of_lens_planes,...
        nr_of_small_volumes,omega_m_z);

    % calculate the volume of the same small cylinder through the subvolumes
    % ..................................................................
    int_cyl_vol = dc.*pi.*(radius_small^2);
```

```
% calculate the density of the small cylinder through the subvolumes
% ..............................................................
if mass_in_halos == 0
    % all mass clustered in halos within the mass range

    if substruc == 0
        % no substructure has been generated

        density_small_cyl = mass_small_cyl./int_cyl_vol;
    else
        density_small_cyl = (mass_small_cyl+mass_small_cyl_in_sub)./...
            int_cyl_vol;
    end
else
    if substruc == 0
        % no substructure has been generated

        density_small_cyl = mass_small_cyl./int_cyl_vol + bg_density;
    else
        density_small_cyl = (mass_small_cyl+mass_small_cyl_in_sub)./...
            int_cyl_vol + bg_density;
    end
end

% compare to the critical density in matter for each subvolume
% ........................................................
dens_quot = density_small_cyl./rho_mass;

% output dens_quot into outfile
% ...........................
fid = fopen(outfile,'a');
count = fprintf(fid,'%10.4f',dens_quot);
count = fprintf(fid,'\n',' ');
fclose(fid);


if show_plot == 1

    % plot the spatial halo distribution for the subvolume with index anf
    % ..............................................................
    X = zeros(1,size(cut_off,2));
    Y = zeros(1,size(cut_off,2));
    Z = zeros(1,size(cut_off,2));
    xx = zeros(1,size(cut_off,2));
    yy = zeros(1,size(cut_off,2));
    zz = zeros(1,size(cut_off,2));
    clf
    k = anf;
    [X,Y,Z] = pol2cart(phi(k,:),r(k,:),h(k,:));
    i = 1;
    eoX = 0;
    while eoX == 0
        if X(1,i) == 0
            X = X(1,1:(i-1));
            Y = Y(1,1:(i-1));
            Z = Z(1,1:(i-1));
            eoX = 1;
        else
            i = i+1;
        end
    end
    colordef white;
    colormap(bone);
    hold on
    for i = 1:size(X,2)
        if cut_off(k,i) ~= 0
            [xx,yy,zz] = sphere;
            xx = cut_off(k,i)*xx+X(1,i);
            yy = cut_off(k,i)*yy+Y(1,i);
            zz = cut_off(k,i)*zz+Z(1,i);
            s1=surf(zz, yy, xx);
            set(s1, 'FaceColor', 'flat', 'FaceLighting', 'phong', ...
                'DiffuseStrength', 0.1, 'SpecularStrength', 0.18, ...
                'AmbientStrength', 0.7);
            shading flat
            light;
        end
    end
    t = 0:0.2:2*pi+0.2;
    null = linspace(0,0,33);
    plot3(null,radius.*cos(t),radius.*sin(t),'k');
    plot3(null+dc(k),radius.*cos(t),radius.*sin(t),'k');
    [x,y,z] = cylinder(radius_small,20);
    z = dc(k).*z;
    plot3(null,radius_small.*cos(t),radius_small.*sin(t),'k');
    plot3(null+dc(k),radius_small.*cos(t),radius_small.*sin(t),'k');
    plot3(z,y,x,'y');
    hold off
    set(gca,'PlotBoxAspectRatio',[1 0.27 1])
end

if show_sub_plot == 1

    % plot the first halo and its subhalos that penetrates the line of
    % sight
    % ..............................................................
    clf
    if substruc == 1
        found_sub = 0;
        n = 0;
        while found_sub == 0 & n ~= nr_of_lens_planes
            n = n+1;
            for l = 1:size(sub_halo,2)
                if sub_halo(n,l) == 1
                    found_sub = 1;
                    break
                end
            end
        end
        if found_sub == 1
            colordef white;
            colormap(cool);
            hold on
            [XX,YY,ZZ] = sphere;
            XX = cut_off(n,l)*XX;
```

```
                YY = cut_off(n,l)*YY;
                ZZ = cut_off(n,l)*ZZ;
                plot3(YY,ZZ,XX);
                for s = 2:size(mass_ran_sub,3)
                    if mass_ran_sub(n,1,s) ~= 0
                        s
                        [xx,yy,zz] = sphere;
                        xx = cut_off_sub(n,1,s)*xx+x_sub(n,1,s);
                        yy = cut_off_sub(n,1,s)*yy+y_sub(n,1,s);
                        zz = cut_off_sub(n,1,s)*zz+z_sub(n,1,s);
                        s2=surf(xx, yy, zz);
                        set(s2, 'FaceColor', 'flat', 'FaceLighting', ...
                            'phong', 'DiffuseStrength', 0.03, ...
                            'SpecularStrength', 0.01, 'AmbientStrength', 0.3);
                        shading flat
                        light;
                        set(gcf, 'color', [1 1 1]);
                    end
                end
                hold off;
                mass_ran(n,l)
                n
            else
                disp('no halo hit this time...');
            end
        end
    end


    clear mass bg_density fmax dc_nb mass_nb phi_nb rho zeta max_z C ...
    mass_prov dc_nb_dum max_length density_nb X Y Z_l Z_u total_length_nb ...
    total_length bg_d1 bg_d2 cut M2M_c_s subfrac mass_sub mass_substr ...
    mass_ran h r phi cut_off sub_halo mass_ran_sub x_sub y_sub z_sub r_sub ...
    h_sub conc_par cut_off_sub mass_sub_struc mass_small_cyl_in_sub ...
    mass_small_cyl int_cyl_vol density_small_cyl dens_quot;
    pack;

end


% ************************* %
%       subfunctions        %
% ************************* %


function [outfile,z_QSO,lambda,omega_b,omega_c,hubble,M_min,M_max,n_body,...
        mass_in_halos,substruc,M_sub_min,M_sub_max] = input_parameters;
% get input parameters manual/from file
input_mode = input('choose input mode: \n1 = manual input \n2 = read input ...
    parameters from file \n');
if input_mode==1
    % manual input selected

    outfile = input('enter name of output file:                 ','s');
    z_QSO = input('enter source redshift:                   ');
    lambda = input('enter value of cosmological constant:     ');
    omega_b = input('enter value for omega_b:                  ');
    omega_c = input('enter value for omega_c:                  ');
```

```
        hubble = input('enter value of hubble contant (km/MPc/s):  ');
        M_min = input('enter minimum halo mass [M_sun]:            ');
        M_max = input('enter maximum halo mass [M_sun]:            ');
        disp(' - halo generation -');
        n_body = input(' 0 = apply Press-Schechter mass function\n 1 = use '...
            'n-body simulation data \nchoice: ');
        if n_body == 0
            disp(' - mass distribution -');
            mass_in_halos = input(' 0 = halos of masses outside the given '...
                'boundaries contribute to a background density \n 1 = all mass '...
                'is clustered in halos of sizes within the given boundaries '...
                '\nchoice: ');
        else
            mass_in_halos = 0;
        end
        disp(' - substructure -');
        substruc = input(' 0 = no substructure in halos \n 1 = generate '...
            'subhalos \nchoice: ');
        if substruc == 0
            M_sub_min = 0;
            M_sub_max = 0;
        else
            M_sub_min = input('enter minimum substructure mass [M_sun]:   ');
            M_sub_max = input('enter maximum substructure mass [M_sun]:   ');
        end
    elseif input_mode==2
        % input via file selected
        parameterfile = input('enter name of file containing input parameters: '...
            ,'s');
        fid = fopen(parameterfile);
        if fid==-1
            error('can`t open the file');
        end
        outfile = input('enter name of output file:                       ','s');
        B = fscanf(fid,'%g',12);
        z_QSO = B(1);
        lambda = B(2);
        omega_b = B(3);
        omega_c = B(4);
        hubble = B(5);
        M_min = B(6);
        M_max = B(7);
        n_body = B(8);
        mass_in_halos = B(9);
        substruc = B(10);
        M_sub_min = B(11);
        M_sub_max = B(12);
        disp('following parameters used:');
        z_QSO
        lambda
        omega_b
        omega_c
        hubble
        M_min
        M_max
        n_body
        if n_body == 0
            mass_in_halos
    else
        mass_in_halos = 1;
    end
    substruc
    if substruc ~= 0
        M_sub_min
        M_sub_max
    end
else
    disp('mistyped..?');
end

function [dc] = calc_distances(z,omega_m,lambda,c,hubble,MPc,nolp);
% calculate needed distances for each lense plane to create subvolumes [m]
for n = 1:nolp
    if n==1
        dc(1,n) = quad('light_travel',0,z(n),[],[],omega_m,lambda,c,hubble,MPc);
    else
        dc(1,n) = quad('light_travel',z(n-1),z(n),[],[],omega_m,lambda,c,...
            hubble,MPc);
    end
end

function [v] = generate_subvolume(dc,radius);
% generate subvolume for each lens plane
v = pi*radius^2.*dc;

function [mass] = mass_in_interval(bin_steps,M_c,v,n,M_int,rho_critical,...
    M_min,M_max,c,p,A,q,delta_c_z,ipsi,MPc);
% evaluates the mass of halos clustered within the given mass range
% following the Press-Schechter mass function
mass = 0;
for i = 1:(bin_steps-1)
    M(i) = M_int(i)+10^(0.5*(log10(M_int(i+1))-log10(M_int(i))));
    M_u = M_int(i+1);
    M_l = M_int(i);
    sigma_d2 = delta_c_z^2*(M(i)/M_c)^(-(3+ipsi)/3);
    sigma_d2_u = delta_c_z^2*(M_u/M_c)^(-(3+ipsi)/3);
    sigma_d2_l = delta_c_z^2*(M_l/M_c)^(-(3+ipsi)/3);
    df = (log(sqrt(1/sigma_d2_u))-log(sqrt(1/sigma_d2_l)))/(log10(M_u)-log10...
        (M_l));
    % Press-Schechter:
    f(i) = rho_critical/M(i)*df*A*(1+(q*delta_c_z^2/sigma_d2)^(-p))*sqrt...
        (2*q*delta_c_z^2/(pi*sigma_d2))*exp(-q*delta_c_z^2/(sigma_d2*2));
    mass = mass + (f(i)*(log10(M_int(i+1))-log10(M_int(i)))*v*M(i));
end

function [mass] = mass_in_interval_corr(bin_steps,M_c,v,n,M_int,...
    rho_critical,M_min,M_max,c,p,A,q,delta_c_z,ipsi,MPc);
% evaluates the mass of halos clustered within the given mass range
% following the Press-Schechter mass function adjusted by a factor 2
mass = 0;
for i = 1:(bin_steps-1)
    M(i) = M_int(i)+10^(0.5*(log10(M_int(i+1))-log10(M_int(i))));
    M_u = M_int(i+1);
    M_l = M_int(i);
    sigma_d2 = delta_c_z^2*(M(i)/M_c)^(-(3+ipsi)/3);
    sigma_d2_u = delta_c_z^2*(M_u/M_c)^(-(3+ipsi)/3);
    sigma_d2_l = delta_c_z^2*(M_l/M_c)^(-(3+ipsi)/3);
    df = (log(sqrt(1/sigma_d2_u))-log(sqrt(1/sigma_d2_l)))/(log10(M_u)-...
        log10(M_l));
    % Press-Schechter adjusted by a factor 2:
    f(i) = rho_critical/M(i)*df*A*(1+(q*delta_c_z^2/sigma_d2)^(-p))*sqrt...
        (q*delta_c_z^2/(2*pi*sigma_d2))*exp(-q*delta_c_z^2/(sigma_d2*2));
    mass = mass + (f(i)*(log10(M_int(i+1))-log10(M_int(i)))*v*M(i));
end

function[mass] = mass_in_interval_gif(z,rho_mass,v);
% evaluates the mass of halos cluster within the iven mass range following
% the GIF N-body fit
frac = -1.408e-5*z^5+0.00046552*z^4-0.0065069*z^3+0.050636*z^2-0.22559*z+0.45099;
mass = frac*rho_mass*v;

function [h,r,phi] = generate_coordinates_rand(dc,n,radius);
% generate random polar coordinates for the lth dark halo in the nth subvolume
rand_num = rand(1);
h = rand_num * dc(n);
r = 0;
while r == 0
    x = rand(1);
    y = rand(1);
    rad = sqrt(x^2+y^2);
    if rad <= 1
        r = rad * radius;
    end
end
rand_num = rand(1);
phi = rand_num*2*pi;

function [fmax] = getmax_mass_dist(M_c,n,M_int,rho_critical_z,M_min,M_max,c,...
    p,A,q,delta_c_z,n_eff,MPc);
% evaluates the maximum value of the Press-Schechter mass function
M = M_min + 0.5*(M_int(2)-M_int(1));
M_u = M_int(2);
M_l = M_int(1);
sigma_d2 = delta_c_z^2*(M/M_c)^(-(3+n_eff)/3);
sigma_d2_u = delta_c_z^2*(M_u/M_c)^(-(3+n_eff)/3);
sigma_d2_l = delta_c_z^2*(M_l/M_c)^(-(3+n_eff)/3);
df = (log(sqrt(1/sigma_d2_u))-log(sqrt(1/sigma_d2_l)))/(log10(M_u)-log10(M_l));
% Press-Schechter mass function
fmax = MPc^3*rho_critical_z/M*df*A*(1+(q*delta_c_z^2/sigma_d2)^(-p))*sqrt...
    (2*q*delta_c_z^2/(pi*sigma_d2))*exp(-q*delta_c_z^2/(sigma_d2*2));

function [mass_ran] = halomass(M_sun,M_c,fmax,n,M_int,rho_critical_z,M_min,...
    M_max,c,p,A,q,delta_c_z,n_eff,MPc);
% return random halo mass drawn from the Press-Schechter halo mass function
mass_ran = 0;
while mass_ran == 0
    rand_num = rand(1);
    M = 10^(log10(M_min) + rand_num*(log10(M_max)-log10(M_min)));
    i = 1;
    while M >= M_int(i)
        i = i + 1;
    end
    M_u = M_int(i);
    M_l = M_int(i-1);
    sigma_d2 = delta_c_z^2*(M/M_c)^(-(3+n_eff)/3);
    sigma_d2_u = delta_c_z^2*(M_u/M_c)^(-(3+n_eff)/3);
    sigma_d2_l = delta_c_z^2*(M_l/M_c)^(-(3+n_eff)/3);
    df = (log(sqrt(1/sigma_d2_u))-log(sqrt(1/sigma_d2_l)))/(log10(M_u)-log10...
        (M_l));
    % Press-Schechter mass function:
    f = MPc^3*rho_critical_z/M*df*A*(1+(q*delta_c_z^2/sigma_d2)^(-p))*sqrt...
        (2*q*delta_c_z^2/(pi*sigma_d2))*exp(-q*delta_c_z^2/(sigma_d2*2));
    rand_num = rand(1);
    if rand_num <= f/fmax(n)
        mass_ran = M/M_sun;
    end
end

function [cut_off] = calc_cut_off_all(mass_ran,z,omega_m,hubble,Pc,nolp,...
    omega_m_z);
% calculate the cut_off radii for all halos
for n = 1:nolp
    if n == 1
        cut_off(n,:) = calc_cut_off(mass_ran,z,n,omega_m,hubble,Pc,...
            omega_m_z(n));
    else
        cut_off_dum = calc_cut_off(mass_ran,z,n,omega_m,hubble,Pc,omega_m_z(n));
        if size(cut_off_dum,2) > size(cut_off,2)
            cut_off = [cut_off zeros(n-1,size(cut_off_dum,2)-size(cut_off,2))];
        end
        cut_off(n,:) = cut_off_dum;
    end
end

function [sub_halo] = check_sub_halo(cut_off,r,h,radius_small,nolp);
% check which halos penetrate the small integration cylinder
for n = 1:nolp
    for l = 1:size(cut_off,2)
        if cut_off(n,l) == 0
            % space in cut_off matrix not occupied by a halo
            sub_halo(n,l) = 0;
        else
            if (r(n,l)-cut_off(n,l)) <= radius_small
                % halo reaches into the small integration cylinder
                if (h(n,l)-cut_off(n,l)) > 0
                    % taking out the halos that don't lie totally in the
                    % subvolume - only on one side
                    sub_halo(n,l) = 1;
                else
                    sub_halo(n,l) = 0;
                end
            else
                sub_halo(n,l) = 0;
            end
        end
    end
end

function [fsubmax] = getmax_sub_mass_dist(M_sun,M_sub_min,hubble);
% evaluate the maximum value of the subhale mass function
fsubmax = 10^3.2*(M_sub_min*hubble/(100*M_sun))^(-1.9)*hubble/(100*M_sun);
```

```
function [mass_ran_sub] = subhalomass(M_sun,M_sub_min,M_sub_max,hubble,...
    M2M_c_s,subfrac,beta,mass_substr,mass_ran);
% return random subhalo mass drawn from the given subhalo mass distribution
% function
mass_ran_sub = 0;
alpha = 0.966-0.028*log10(M2M_c_s);
normaliz = real(subfrac/(gammainc(1-alpha,1/beta)-gammainc(1-alpha,1e-4/beta)));
fsubmax = normaliz/(beta*gamma(1-alpha))*(M_sub_min/(beta*(mass_substr+...
    mass_ran)*M_sun))^(-alpha)*exp(-M_sub_min/(beta*(mass_substr+mass_ran)*...
    M_sun));
while mass_ran_sub == 0
    rand_num = rand(1);
    M_sub = 10^(log10(M_sub_min) + rand_num*(log10(M_sub_max)-log10...
        (M_sub_min)));
    f = normaliz/(beta*gamma(1-alpha))*(M_sub/(beta*(mass_substr+mass_ran)*...
        M_sun))^(-alpha)*exp(-M_sub/(beta*(mass_substr+mass_ran)*M_sun));
    rand_num = rand(1);
    if rand_num <= f/fsubmax
        mass_ran_sub = M_sub/M_sun;
    end
end

function [x_sub,y_sub,z_sub,r_sub,h_sub]= generate_sub_coordinates_rand...
    (conc_par,cut_off,r,n,l,h);
% generate coordinates for the mth subhalo inside the lth halo inside the
% nth subvolume
got_r = 0;
while got_r == 0
    rand_num_r = rand(1);
    f = (1+0.244*conc_par)*(rand_num_r^2.75)/(1+0.244*conc_par*rand_num_r^2);
    rand_num_f = rand(1);
    if rand_num_f <= f
        r_polar = rand_num_r*cut_off(n,l);   % r_polar in units of cut_off(n,l)
        got_r = 1;
    end
end
rand_num = rand(1);
theta = rand_num * 2 * pi;
rand_num = rand(1);
phi = rand_num * pi - pi/2;
[x_sub,y_sub,z_sub] = sph2cart(theta,phi,r_polar);
r_sub = sqrt(z_sub^2+r(n,l)^2-2*z_sub*r(n,l)+x_sub^2);
h_sub = h(n,l) + y_sub;

function [cut_off_sub] = calc_cut_off_sub_all(mass_ran_sub,z,omega_m,hubble,...
    Pc,nolp,omega_m_z);
% calculate the cut_off radii for all subhalos
for n = 1:nolp
   for m = 1:size(mass_ran_sub,2)
       cut_off_sub(n,m,:) = calc_cut_off_sub(mass_ran_sub,z,n,m,omega_m,...
           hubble,Pc,omega_m_z(n));
   end
end

function [mass_small_cyl] = calc_mass_small_cyl(radius_small,cut_off,r,h,...
    mass_ran,hubble,G,M_sun,z,omega_m,dc,Pc,MPc,nolp,nosv,omega_m_z);
% calculate the masses inside a small cylinder through the subvolumes [kg]
mass_small_cyl = zeros(1,nolp);
```

```
for n = 1:nolp
    for l = 1:size(cut_off,2)
        if cut_off(n,l) == 0
            % space in cut_off matrix not occupied by a halo
        else
            if (r(n,l)-cut_off(n,l)) <= radius_small
                % halo reaches into the small cylinder
                if (h(n,l)-cut_off(n,l)) > 0
                    % taking out the halos which don't lie totally inside the
                    % subvolume - only on one side
                    rel_cyl_length = 2*sqrt((cut_off(n,l))^2-(r(n,l))^2);
                    small_volume = rel_cyl_length/nosv*pi*(radius_small^2);
                    conc_par = 124/(1+z(n))*(mass_ran(n,l)*hubble/100)^(-0.084);
                    dens_c =  real(200/3*conc_par^3/(log(1+conc_par)-...
                        conc_par/(1+conc_par)));
                    for m = 1:(nosv/2)
                        % deviding the halo-cylinder into subvolumes
                        b = (2*m-1)/(2*nosv)*rel_cyl_length;
                        r_rho = sqrt((r(n,l))^2+b^2);
                        if r_rho <= cut_off(n,l)
                            % checking that the subvolume lies inside the
                            % cut_off radius
                            x = r_rho/cut_off(n,l);
                            rho = (3*(1000*hubble/MPc)^2)/(8*pi*G)*(1+z(n))...
                                ^3*omega_m/omega_m_z(n)*dens_c/(conc_par*x*...
                                (1+conc_par*x)^2);
                            mass_small_volume = rho*small_volume;
                            mass_small_cyl(n) = mass_small_cyl(n) + (2*...
                                mass_small_volume);
                        end
                    end
                end
            end
        end
    end
end

function [mass_small_cyl_in_sub] = calc_mass_small_cyl_in_sub(radius_small,...
    cut_off_sub,r_sub,h_sub,mass_ran_sub,hubble,G,M_sun,z,omega_m,dc,Pc,...
    MPc,nolp,nosv,omega_m_z);
% calculate the masses inside a small cylinder through the subvolumes [kg]
mass_small_cyl_in_sub = zeros(1,nolp);
for n = 1:nolp
    for m = 1:size(cut_off_sub,2)
        for s = 2:size(cut_off_sub,3)
            if cut_off_sub(n,m,s) ~= 0
                if (r_sub(n,m,s)-cut_off_sub(n,m,s)) <= radius_small
                    if (h_sub(n,m,s)-cut_off_sub(n,m,s)) > 0
                        rel_cyl_length = 2*sqrt((cut_off_sub(n,m,s))^2-...
                            (r_sub(n,m,s))^2);
                        small_volume = rel_cyl_length/nosv*pi*(radius_small^2);
                        conc_par = 124/(1+z(n))*(mass_ran_sub(n,m,s)*hubble/...
                            100)^(-0.084);
                        dens_c =  real(200/3*conc_par^3/(log(1+conc_par)-...
                            conc_par/(1+conc_par)));
                        for o = 1:(nosv/2)
                            % deviding the halo-cylinder into subvolumes
                            b = (2*o-1)/(2*nosv)*rel_cyl_length;
```

```
                                    r_rho = sqrt((r_sub(n,m,s))^2+b^2);
                                    if r_rho <= cut_off_sub(n,m,s)
                                        % checking that the subvolume lies inside the
                                        % cut_off radius
                                        x = r_rho/cut_off_sub(n,m,s);
                                        rho = (3*(1000*hubble/MPc)^2)/(8*pi*G)*(1+...
                                            z(n))^3*omega_m/omega_m_z(n)*dens_c/...
                                            (conc_par*x*(1+conc_par*x)^2);
                                        mass_small_volume = rho*small_volume;
                                        mass_small_cyl_in_sub(n) = ...
                                            mass_small_cyl_in_sub(n) + (2*...
                                            mass_small_volume);
                                    end
                                end
                            end
                        end
                    end
                end
            end
        end

function [cut_off] = calc_cut_off(mass_ran,z,n,omega_m,hubble,Pc,omega_m_z);
% calculate the cut-off-radius of all generated halos in the nth subvolume
% (r200)
cut_off = 1.63e-2.*(mass_ran(n,:).*hubble./100).^(1/3).*(omega_m./omega_m_z)...
    ^(-1/3)*(1+z(n))^(-1)*(hubble/100)^(-1)*1000*Pc;

function [cut_off_sub] = calc_cut_off_sub(mass_ran_sub,z,n,m,omega_m,hubble,...
    Pc,omega_m_z);
% calculate the cut-off-radius of all generated halos in the nth subvolume
%(r200)
cut_off_sub = 1.63e-2.*(mass_ran_sub(n,m,:).*hubble./100).^(1/3).*(omega_m./...
    omega_m_z)^(-1/3)*(1+z(n))^(-1)*(hubble/100)^(-1)*1000*Pc;
cut_off_sub(1) = mass_ran_sub(n,m,1);


function y = light_travel(x,omega_m,lambda,c,hubble,MPc);
% integrant of Surpi's formula to evaluate the distance for a volume between
% z(n-1) and z(n)
y = c/(1000*hubble)*MPc*((1+x).*sqrt(omega_m.*(1+x).^3-(omega_m+lambda-1).*...
    (1+x).^2+lambda)).^(-1);


function y = sigma_integral(x,M,omega_m,omega_c,omega_b,hubble,c,ipsi,...
    rho_critical,D_1,D_10,z_eq,k_eq,k_silk,s,alpha_c,beta_c,alpha_b,beta_b,...
    beta_node,delta_h);
% calculates the integral for sigma_d2
MPc                 = 3.0857e22;            % MPc in m
theta27             = 1.0104;               % theta27 = T(CMB)/2.7K

h = hubble/100;

R = ((M*3)/(4*pi*rho_critical))^(1/3);
W = 3./(x.*R).^3.*(sin(x.*R)-x.*R.*cos(x.*R));
% fourier transform of the spherical top hat function of radius R(M)

q = x./(13.41*k_eq);
% k scaled with k_eq
```

```
f = 1./(1+(x.*s/5.4).^4);

C_k_1_b = 14.2+386./(1+69.9.*q.^1.08);
T_k_1_b = log(exp(1)+1.8*beta_c.*q)./(log(exp(1)+1.8*beta_c.*q)+C_k_1_b.*q.^2);
% pressureless transfer function for (k,1,beta_c)

C_k_a_b = 14.2/alpha_c+386./(1+69.9.*q.^1.08);
T_k_a_b = log(exp(1)+1.8*beta_c.*q)./(log(exp(1)+1.8*beta_c.*q)+C_k_a_b.*q.^2);
% pressureless transfer function for (k,alpha_c,beta_c)

T_c = f.*T_k_1_b+(1-f).*T_k_a_b;
%CDM sector transfer function

s_t = s./(1+(beta_node./(x.*s)).^3).^(1/3);
% effective sound horizon

C_k_1_1 = 14.2+386./(1+69.9.*q.^1.08);
T_k_1_1 = log(exp(1)+1.8.*q)./(log(exp(1)+1.8.*q)+C_k_1_1.*q.^2);
% pressureless transfer function for (k,1,1)

T_b = (T_k_1_1./(1+(x.*s./5.2).^2)+alpha_b./(1+(beta_b./(x.*s)).^3).*...
    exp(-(x./k_silk).^1.4)).*sin(x.*s_t)./(x.*s_t);
% baryon sector transfer function

T = omega_b/omega_m.*T_b+omega_c/omega_m.*T_c;
% transfer function

delta_k2 = delta_h^2.*(c.*x.*MPc/(1000*hubble)).^(3+ipsi).*T.^2.*D_1.^2./...
    D_10.^2;
% present-day normalization of the power spectrum

y = 1./x.*delta_k2.*(W.^2);
```

# Appendix B

# List of input parameters

Example of adequate input involving the default parameters (input via file is read into a 12-elemented row vector organised as following):

```
0.5    % z_QSO                 : redshift of the light source

0.7    % lambda                : cosmological constant today

0.04   % omega_b               : cosmological baryon density today

0.26   % omega_c               : cosmological CDM density today

70     % hubble [km/MPc/s]     : Hubble constant

1e11   % M_min [M_sun]         : minimum halo mass

1e14   % M_max [M_sun]         : maximum halo mass

0      % n-body                  0 => Press-Schechter mass function
       %                         1 => N-body

1      % mass_in_halos         : 0 => all mass clustered
       %                         1 => Press-Schechter mass fraction
       %                         2 => Press-Schechter mass fraction / 2
       %                         3 => GIF mass fraction fit

0      % substruc              : 0 => no substructure
       %                         1 => substructure

5e8    % M_sub_min [M_sun]     : minimum subhalo mass

1e11   % M_sub_max [M_sun]     : maximum subhalo mass
```