Brigham Young University

BYU ScholarsArchive

2010-07-06

In Search of Empty Places: Voids in the Distribution of Galaxies

Brian K. Bucklein
Brigham Young University - Provo

Follow this and additional works at: https://scholarsarchive.byu.edu/etd

Part of the Astrophysics and Astronomy Commons, and the Physics Commons

www.manaraa.com

In Search of Empty Places

Voids in the Distribution of Galaxies

Brian K. Bucklein

A dissertation submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

J. Ward Moody, Chair
Eric G. Hintz
Benjamin J. Taylor
Denise Stephens
Grant W. Hart

Department of Physics and Astronomy

Brigham Young University

August 2010

ABSTRACT

In Search of Empty Places

Voids in the Distribution of Galaxies

Brian K. Bucklein

Department of Physics and Astronomy

Doctor of Philosophy

We investigate several techniques to identify voids in the galaxy distribution of matter in the universe. We utilize galaxy number counts as a function of apparent magnitude and Wolf plots to search a two- or three-dimensional data set in a pencil-beam fashion to locate voids within the field of view. The technique is able to distinguish between voids that represent simply a decrease in density as well as those that show a build up of galaxies on the front or back side of the void. This method turns out to be primarily useable only at relatively short range (out to about 200 Mpc). Beyond this distance, the characteristics indicating a void become increasingly difficult to separate from the statistical background noise. We apply the technique to a very simplified model as well as to the Millennium Run dark matter simulation. We then compare results with those obtained on the Sloan Digital Sky Survey.

We also created the Watershed Void Examiner (WaVE) which treats densities in a fashion similar to elevation on a topographical map, and then we allow the "terrain" to flood. The flooded low-lying regions are identified as voids, which are allowed to grow and merge as the level of flooding becomes higher (the overdensity threshold increases). Void statistics can be calculated for each void. We also determine that within the Millennium Run semi-analytic galaxy catalog, the walls that separate the voids are permeable at a scale of 4 Mpc. For each resolution that we tested, there existed a characteristic density at which the walls could be penetrated, allowing a single void to grow to dominate the volume. With WaVE, we are able to get comparable results to those previously published, but often with fewer choices of parameters that could bias the results. We are also able to determine the the density at which the number of voids peaks for different resolutions as well as the expected number of void galaxies. The number of void galaxies is amazingly consistent at an overdensity of -0.600 at all resolutions, indicating that this could be a good choice for comparing models.

Keywords: large-scale structure of universe, dark matter, methods: n-body simulations, surveys

ACKNOWLEDGMENTS

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Most people that set out to write a dissertation first decide on *something* to research; in fact, it is very rare to intentionally choose to research *nothing.* However, it turns out that empty regions in the galaxy distribution of the universe can shed light on the mechanisms involved in the clustering of galaxies as well as on the role of environment in the evolution of galaxies. Large scale structure is created after the big bang and so is a tracer of conditions that existed in the early universe. The distribution of voids and superclusters can shed light on the dark corners of the universe's past.

In order to understand voids and their effects, there are several questions that need to be answered. First, what do we mean by a void? Second, how do we identify voids? And finally, are voids empty, and do we want them to be? We consider each question in turn.

1.1 What is a Void?

Webster's New World Dictionary of American English (1988) defines a void as "total emptiness; an empty space or vacuum" or a "total absence of something normally expected." While these definitions are technically correct, they are not the meanings used by most

astronomers.

When astronomers speak of voids, we most often want to allow them to contain a few galaxies. But how many is a few? How do we quantify what is acceptable? Most often, this is done by defining a parameter called the *overdensity*, δ, as follows:

$$\delta = \frac{\rho}{\bar{\rho}} - 1 \tag{1.1}$$

where ρ is the density of the region in question and $\bar{\rho}$ is the density of the overall sample volume. Negative values of the overdensity correspond to regions with below-average density, with a value of -1.00 indicating a totally empty region. However, there is no general agreement as to what value of the overdensity should be defined as the upper limit for a void.

Colberg et al. (2008) set out to compare 13 different void-finder algorithms currently in use in the literature. They asked each set of authors to run their algorithm (using their preferred parameters) on a small portion of the Millennium Run simulation (Springel et al. 2005, 2006) that had been populated with the semi-analytic galaxy model of Croton et al. (2006). The Millennium Run modeled the evolution of dark matter over the majority of the history of the universe. Once that simulation was complete, the galaxies were added using a bias function that distributed galaxies based on the size of the dark matter haloes. What Colberg et al. found was that the results of these void finders differed greatly in what constituted a void, in how many void galaxies were identified, and even in which medium the underdensity occurred. Seven of the void finders utilize the dark matter distribution instead of the galaxy distribution. This has a drawback of not being a directly-observable quantity, so these void finders cannot be used on observational datasets. We present an extract of the results from this project, called the Aspen-Amsterdam void finder comparison project, in Table 1.1 as well as a visual representation of the largest void identified by each finder in Figure 1.1.
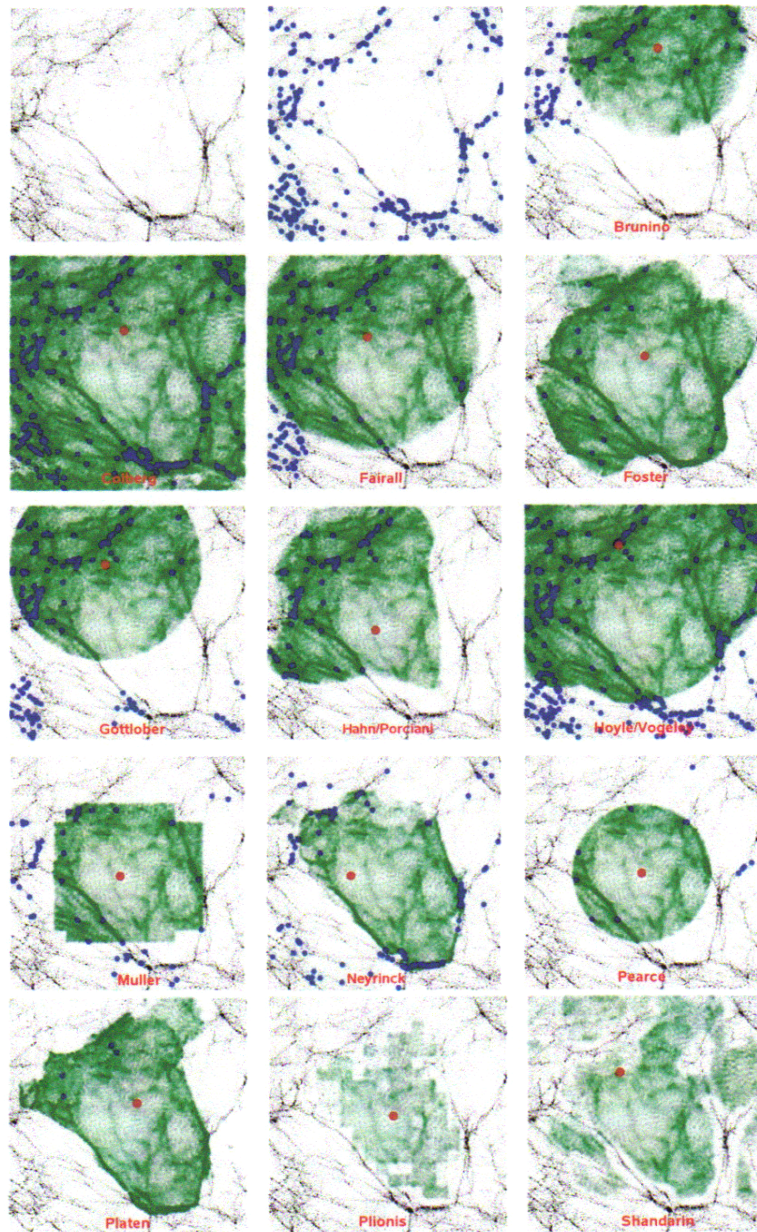
Figure 1.1 Graphical results from the Aspen-Amsterdam void finder comparison project of Colberg et al. (2008). The first panel shows the distribution of dark matter particles in the region from the Millennium Run simulation of Springel et al. (2005). The second panel shows the location of the galaxies from the semi-analytic galaxy map of Croton et al. (2006). The remaining panels show the largest void identified by each void finder, labeled by author names. The red dot indicates the center of the largest void. The green region is the extent of the largest void. The blue dots are the galaxies identified as void galaxies by the finder, although they are not limited to only being in the largest void.

Table 1.1 Results of the Aspen-Amsterdam void finder comparison project, from Colberg et al. (2008). For each void finder, we give the medium in which the void finder searches for voids (G is galaxies, DM is dark matter), the total number of voids identified, N_V, the volume filling fraction, FF_v, the number of void galaxies identified, N_g, and the average overdensity in the voids, δ_g.

Author(s)	Medium	N_V	FF_v	N_g	δ_g
Brunino	G	3	0.37	754	-0.71
Colberg	DM	21	0.92	2258	-0.65
Fairall	G	18	0.59	1376	-0.67
Foster/Nelson	G	3	0.41	114	-0.96
Gottlöber	G	9	0.35	733	-0.70
Hahn/Porciani	DM	14	0.29	248	-0.92
Hoyle/Vogeley	G	4	0.84	2166	-0.56
Müller	G	24	0.58	1469	-0.65
Neyrinck	DM	29	0.32	834	-0.63
Pearce	DM	5	0.15	51	-0.95
Platen/Weygaert	DM	167	1.00	18	-1.00
Plionis/Basilakos	DM	15	0.13	0	-1.00
Shandarin/Feldman	DM	19	0.23	0	-1.00

Because our methods are designed to detect voids in the galaxy distribution, we will focus on the characteristics of the galaxy-based finders. As can be seen, there are wide discrepancies in voids that are found. The number of voids ranged from a low of 3 to a high of 24. Notice the vast differences between the contents of the voids: Foster/Nelson found 3 voids with 114 void galaxies in them compared to Hoyle/Vogeley who found 4 voids with 2,166 void galaxies in them. The percentage of the total volume identified as part of a void ranged from 35% to 84%. Perhaps the biggest indicator of how important it is to have a

consistent definition of a void is shown by the number of void galaxies, which ranges from 114 to 2166, a factor of 19 more galaxies! Even the average overdensity of void regions shows significant variations, from -0.56 to -0.96.

It became increasingly clear to us that part of the research question we would have to address would be to determine just how sensitive our void finder would be to arbitrary choices of parameters. Is there a compelling reason to prefer one choice of parameters over another? How significant are the differences?

1.2 How do we Identify Voids?

Our first attempt to identify voids was by using galaxy number counts. By utilizing plots of galaxy counts versus apparent magnitude, we hoped to be able to calibrate the deviations we saw in a three-dimensional data set so we could search for voids in a larger, deeper two-dimensional data set.

We utilized this technique on a simple three-dimensional model of the universe and it showed promising results. We were able to see the expected deviations out to approximately 200 Mpc. We then applied the technique to the Millennium Run simulation. The technique showed mixed results. On some voids, the technique worked very well. However, on other voids, some of the deviations turned out to be so small as to be unable to be separated out from the statistical "background noise." This is most commonly due to the distance to the center of the void. This limits the usefulness of the method to nearby space. Also, some voids failed to have a sufficient number of galaxies on the front side, causing the signal to never reach our threshold level.

Chapter 2 provides more detail on the method and our results. It also includes some suggestions for ways to improve the technique and areas for which it may turn out to be useful.

Our second attempt at locating voids was very different. Many void finding algorithms have been developed over the years. Many of them are based on the work of El-Ad and Piran (1997). Their method was a so-called "empty sphere" technique which sought to identify empty voids while ignoring small gaps in the walls surrounding them. We wanted to use some of their ideas, but with a very different twist. We designed a watershed void finder algorithm which allows us to examine voids with any threshold for the overdensity. It also tracks how voids grow and merge as the overdensity threshold is increased. We can use it to identify void galaxies in the sample as well.

We applied this algorithm on the Millennium Run simulation with very good results. It was able to probe the permeability of the walls between voids as well as providing distributions of voids sizes at various underdensities. Once again, we discovered that the technique, although it only has a single input parameter, is sensitive to the resolution selected. See Chapter 3 for the details of our technique.

1.3 Are Voids Empty (and do we want them to be)?

The fundamental problem with looking at empty voids is that, by definition, there is nothing to see. Unfortunately, the only way we can gather information is if there *is* something to see. If there isn't something there to emit photons, we cannot receive any information about the area. Therefore, it is often desirable to modify our definition of a void to allow it to contain a small number of galaxies.

Void galaxies often have different properties than galaxies found in the surrounding clusters, the so-called "wall galaxies." The differences in color, size, luminosity, etc. have been thoroughly documented in the literature (see, for example, Grogin and Geller 1999; Popescu et al. 1999; Lindner et al. 1999; Hoyle and Vogeley 2004; Rojas et al. 2004). The void galaxies are found to be bluer, have a higher proportion of spiral galaxies, and have higher star

formation rates. This is the direction we are pursuing in our future work. We are compiling a list of void galaxies in order to investigate how the average properties of these galaxies change as the parameters of the void finding algorithm are adjusted.

There is no generally accepted threshold for how many galaxies are allowed to be inside a void. Some void finders are "purists" in that they do not allow any galaxies. Others will allow any number of galaxies up to the average density in the region. In Chapter 3, we avoid picking a threshold, instead choosing to show how things change as we allow the density threshold to vary.

Chapter 2

Galaxy Number Counts

Galaxy number count (GNC) analysis is an accessible tool widely used in many applications including studies of galaxy luminosity evolution (Metcalfe et al. 2006; Bershady et al. 1998), mapping galactic extinction (Fukugita et al. 2004; Yasuda et al. 2007), mapping the extent of external galaxies (e.g. Ellis and Bland-Hawthorn 2007), delineating large-scale structure (Frith et al. 2006; Dolch and Ferguson 2005; Fukugita et al. 2004), discovering galaxy star-formation evolution (Kong et al. 2006), mapping galaxy x-ray evolution (Georgakakis et al. 2006), determining the AGN fraction (Treister et al. 2006), and better understanding galaxy formation processes (Berrier et al. 2006; López-Corredoira and Betancort-Rijo 2004).

We investigate the suitability of using GNC analysis to analyze nearby voids. As a diagnostic, we utilize plots of $\log(n < m)$ vs. m, which we call Wolf plots after M. Wolf, who used them in stellar extinction studies.

Before using Wolf plots to analyze voids, it is instructive to consider what the slope of a Wolf plot is expected to be in a uniform distribution of galaxies without the presence of voids. We begin with the standard distance modulus equation given by

$$m - M = 5\log(d) - 5 \tag{2.1}$$

where m is the apparent magnitude, M is the absolute magnitude, and d is the distance to

8

the galaxy in parsecs.

If we consider two galaxies of identical absolute magnitude, M, but located at two different distances from the observer, d_1 and d_2 (given in pc), then we can write an expression for the apparent magnitude of each galaxy as

$$m_j = M + 5\log(d_j) - 5 \tag{2.2}$$

where $j = 1$ or 2. Taking the difference in the apparent magnitudes gives us

$$m_2 - m_1 = (M + 5\log(d_2) - 5) - (M + 5\log(d_1) - 5) \tag{2.3}$$

$$m_2 - m_1 = 5\log(d_2) - 5\log(d_1) \tag{2.4}$$

$$m_2 - m_1 = 5\log\left(\frac{d_2}{d_1}\right) \tag{2.5}$$

When we talk of the slope of the graph at some given apparent magnitude, m_k, we use a modified traditional three-point slope formula, where the slope at the middle point is determined by the values one point ahead and one point behind it. The modification involves taking the log of the values first. This provides the following formula

$$slope(m_k) = \frac{\log(n_{k+1}) - \log(n_{k-1}))}{m_{k+1} - m_{k-1}} \tag{2.6}$$

where k is the bin corresponding to a given apparent magnitude and n_k is the number of galaxies brighter than m_k.

If we let s represent the slope, this gives

$$s = \frac{\log(n_2) - \log(n_1)}{m_2 - m_1} \tag{2.7}$$

$$s = \frac{\log\left(\frac{n_2}{n_1}\right)}{m_2 - m_1} \tag{2.8}$$

The number of galaxies in a given volume is based on the density, ϕ, and the volume, V. The volume located within a given distance d is just the volume of a sphere with radius d times the fraction of the sky observed, f. Because we are only dealing with galaxies of

a single absolute magnitude, the density will be constant out to the edge of the surveyed volume. For each of the two distances, then, the number of galaxies brighter than a given magnitude in that volume will be

$$n_j = V_j \phi = \frac{4}{3} \pi d_j^3 f \phi \tag{2.9}$$

If we combine these results by substituting Equations 2.5 and 2.9 into Equation 2.8, we obtain

$$s = \frac{\log \left(\frac{\frac{4}{3} \pi d_2^3 f \phi}{\frac{4}{3} \pi d_1^3 f \phi} \right)}{5 \log \left(\frac{d_2}{d_1} \right)} \tag{2.10}$$

$$s = \frac{\log \left(\frac{d_2^3}{d_1^3} \right)}{5 \log \left(\frac{d_2}{d_1} \right)} \tag{2.11}$$

$$s = \frac{\log \left(\left(\frac{d_2}{d_1} \right)^3 \right)}{5 \log \left(\frac{d_2}{d_1} \right)} \tag{2.12}$$

$$s = \frac{3 \log \left(\frac{d_2}{d_1} \right)}{5 \log \left(\frac{d_2}{d_1} \right)} \tag{2.13}$$

$$s = \frac{3}{5} = 0.6 \tag{2.14}$$

This was derived under the assumption that the galaxies had the same absolute magnitude. In reality, where there are a wide range of absolute magnitudes, we use a binning technique and expect to obtain the same slope of 0.6 for each narrow bin. As long as the bins are sufficiently narrow, the assumption that the density is the same for galaxies of similar absolute magnitude is still valid. Adding together two lines with equal slope gives a resulting line with the same slope. Therefore, when we add all of our bins together, we would anticipate a graph with a slope of 0.6.

A slope that is less than expected would then indicate that there was an underdensity in the expected number of galaxies, while a slope that is larger than predicted would represent

an overdensity of galaxies. Our hope was to use this as a method of finding voids. We had hoped to calibrate the distance of the void using a three-dimensional galaxy survey (such as the spectroscopic data of the Sloan Digital Sky Survey). This would then allow us to utilize the technique on a two-dimensional data set (such as the photometric data of the SDSS).

2.1   Application to a Simple Model

When modeling a system that can become quite complex, it is useful to analyze the effects of each piece in isolation as much as is practical. This allows us to more clearly define how the characteristics of void distance, void size, and edge density will impact our diagnostic. Therefore, we start our analysis with a simple model that allows us to control these parameters.

The galaxy distribution was modeled by Schechter (1976), who found that the number of galaxies in a given volume can be parametrized as follows:

$$\phi(L)dL = \phi^*(L/L^*)^\alpha \exp(-L/L^*)d(L/L^*) \tag{2.15}$$

where ϕ^*, L^*, and α are the parameters chosen to best fit the distribution of galaxies. However, for our purposes, we can re-write Equation 2.15 in terms of magnitudes:

$$\phi(M)dM = \frac{2}{5}\phi^*(\ln 10)\left[10^{\frac{2}{5}(M^*-M)}\right]^{\alpha+1}\exp\left[-10^{\frac{2}{5}(M^*-M)}\right]dM \tag{2.16}$$

where M is the absolute magnitude, and $\phi(M)dM$ is the number of galaxies with magnitude between M and $M + dM$ per Mpc^3. This leaves three parameters, ϕ^*, M^*, and α, to use to fit the observational data. In order to set the overall number density, ϕ^* is adjusted. M^* controls the turn-over point, effectively limiting the bright end of the distribution. Finally, α sets the slope of the faint end. These effects can be distinguished in Figure 2.1.

To obtain a straightforward random distribution of galaxies out of which we could carve voids, we created a Monte-Carlo simulation of a magnitude-limited survey based on the

ϕ(M) in galaxies/Mpc^3
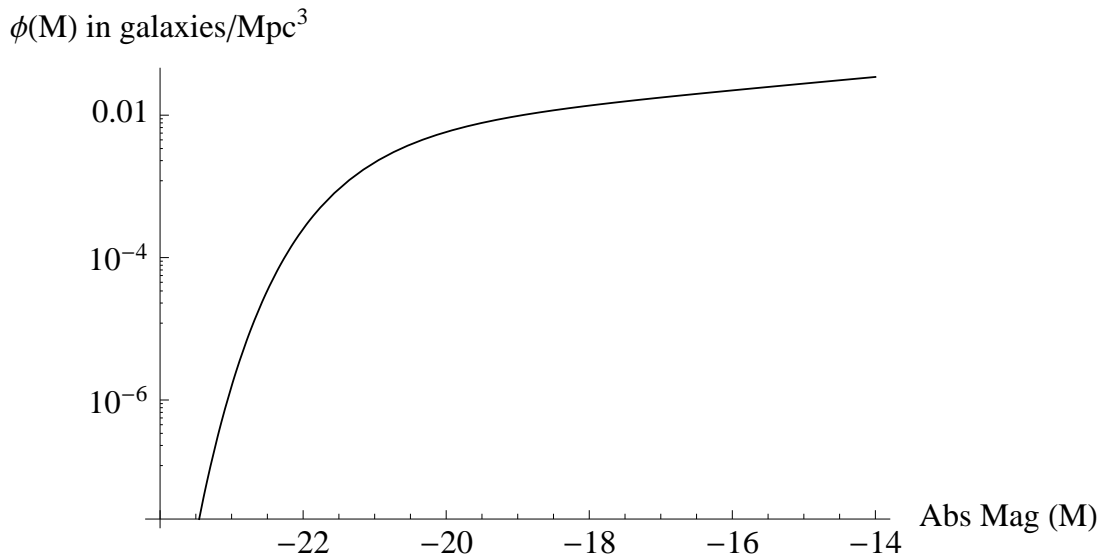
Figure 2.1 Schechter function using the values for ϕ^*, M^*, and α of Montero-Dorta and Prada (2009). ϕ^* adjusts the overall density, M^* controls where the function turns downward at the bright end, and α controls the slope of the faint end.

r-band Schechter function parameters derived from the SDSS DR6 by Montero-Dorta and Prada (2009). The best-fit values they found are $\phi^* = 0.90 \times 10^{-2}h^3$ Mpc^{-3}, $M^* = -20.73 + 5\log(h)$, and $\alpha = -1.23$, all with $h = 1$.

We set a range of absolute magnitudes for the galaxies of $-24.0 < M_r < -14.0$, and let the Schechter function determine how many of each magnitude will appear in our sample. These limits were chosen based on the following reasoning:

1. The Schechter function guarantees that no galaxies with $M_r \lesssim -23.5$ will appear in the sample

2. The faint limit is chosen to be low enough to only exclude galaxies on the near side of the void, so it should not affect the changes introduced by the presence of the void.

As we step through the range of absolute magnitudes, we calculate the maximum distance at which a galaxy of that absolute magnitude would have an apparent magnitude brighter than the completeness limit of a comparison survey. This is done by solving Equation 2.1

for d at the faint magnitude limit.

$$d_{max} = 10^{0.2(m_{lim}-M)+1} \tag{2.17}$$

We used the level for m_{lim} set by the SDSS DR 7, which achieved 90% completeness for photometric data down to $m_r = 22.2$. The distances are randomly assigned to the galaxies with volume weighting within the limit to ensure that their spatial distribution is uniform. This ensures that they would be visible in the field of view given the limiting magnitude. The apparent magnitude was then calculated from the generated distance and the absolute magnitude based on Equation 2.1.

Since the purpose of the toy model is to quantify the effect of voids in the near universe, no higher-order corrections for extinction, K-corrections, or spatial curvature were applied. The apparent magnitude limits at which these effects become important were pointed out by Liske et al. (2003). K-corrections become important at $m_B = 19$, while evolutionary corrections do not become important until $m_B = 22$.

The distribution of galaxies for a sample volume that is 5° in right ascension (arbitrarily chosen to be from 215° to 220°) and 1° in declination (chosen to be from 34.5° to 35.5°) with a limiting magnitude of $m_r = 22.2$ is shown in Figure 2.2. To illustrate the characteristic behavior of Wolf plots, we generated a plot for this distribution, which is shown in Figure 2.3. In these and all following plots, we have arbitrarily increased the value of ϕ^* by a factor of five to better visually illustrate the differences in the various models.

Our model shows good agreement with the predicted slope. The bright end is dominated by small-number statistics which causes the slope to rise rapidly and overshoot the expected slope. In order to minimize the effects of the "noise," we used a least-squares fit of a high-order Legendre polynomial. This serves to smooth out the rapid fluctuations that could mask the front edge of a void. It does, however, produce some oscillation at the faint end when it attempts to model a straight line, so we will ignore the faint end as well. The primary area
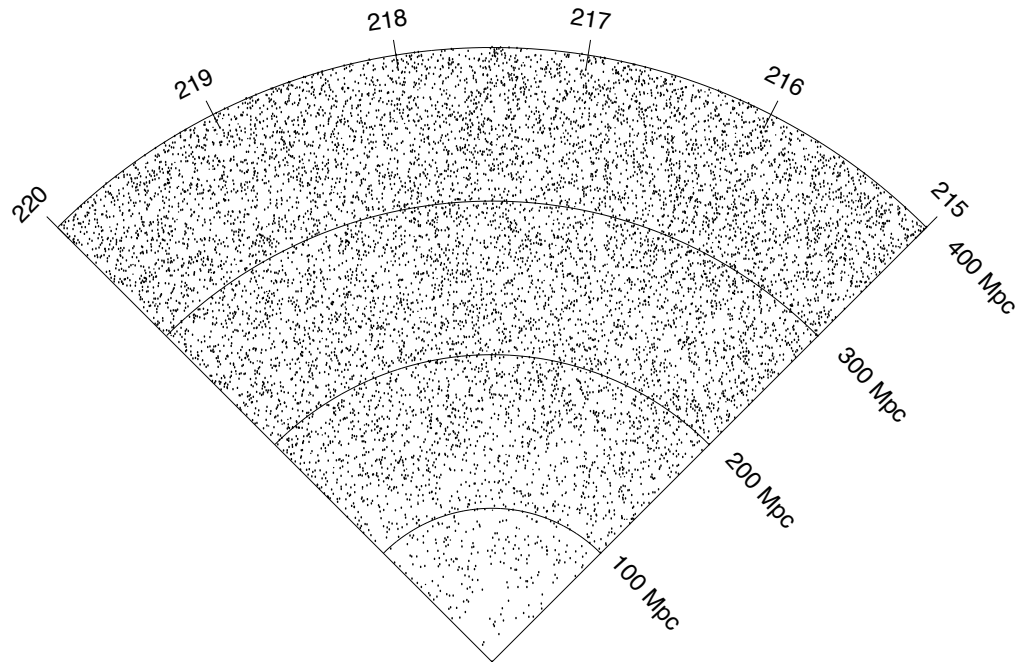
Figure 2.2 A sample uniform galaxy distribution from our simple model. The region is 5° wide in right ascension and 1° thick in declination.

of interest lies in the range of $13.5 \lesssim m \lesssim 18.5$. For all Wolf plots in this study, we have taken the average number of galaxies in each magnitude bin from 10 randomly-generated runs of our models to reduce the error.

2.1.1 Cut-Out Voids

The introduction of voids raises a pertinent question: how does one carve a void in the galaxy distribution? While there are certainly any number of answers to that question, we will look at three cases with our model. For all voids, we wished to avoid "edge effects," where the line-of-sight includes the wall of the void along the radial direction. When this happens, the galaxies in the walls can act to reduce the amplitude of the variations away from the slope of 0.6, potentially masking small voids. Therefore, our voids were designed to span the entire
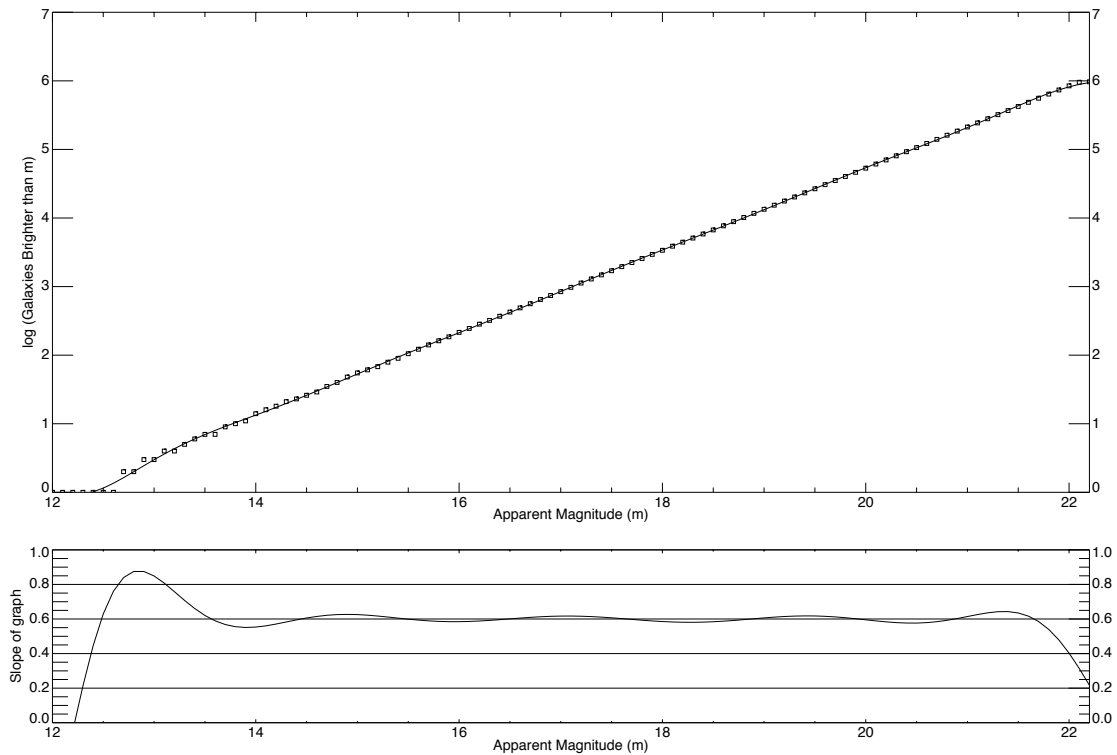
Figure 2.3 The top box shows a Wolf plot for a uniform distribution of galaxies. This is produced by taking the average number of galaxies brighter than the magnitude of the bin from 10 randomly generated models. To smooth out the effects of small-number statistics, we fit a Legendre polynomial to the data. The bottom portion shows the slope of the fit-line using Equation 2.6.

field of view in right ascension as a best-case scenario for using this technique. We expect to see these edge effects when looking at more realistic simulations, such as the Millennium Run (see Section 2.2).

The simplest type of void we consider is what we call a "cut-out" void. We specify the distance to the center of the void and the width along the radial direction. We then begin the same process to locate galaxies as was used in the uniform distribution described above. If the galaxy would have been placed within the boundaries of the void, the galaxy is simply discarded. Although this causes the value of ϕ^* of our sample to be lower than that of Montero-Dorta and Prada (2009), this was deemed a useful first step that would be

corrected later (see Section 2.1.2).

We illustrate a cut-out void centered at 250 Mpc with a width of 100 Mpc in Figure 2.4. The Wolf plot for this void is shown in Figure 2.5. Ignoring the bright end "noise" in the slope, we begin our analysis at $m = 14.0$. At this point, the slope has reached our benchmark of 0.6. This is approximately where our line-of-sight reaches the front edge of the void. The lack of galaxies causes the density to decrease as indicated by the drop in the slope from $14.0 < m < 16.5$. The slope then plays "catch-up" by overshooting 0.6 before settling back down at the faint end (ignoring the oscillation from our curve fit).

An interesting note is that the cross-over point in this model occurs at about $m = 16.50$. If we calculate the apparent magnitude of an M^* galaxy located at the same distance as the center of the void (250 Mpc in this example), we get $m = 16.30$. This is a very good agreement, with only a 1.23% error. We will revisit this in the next section.
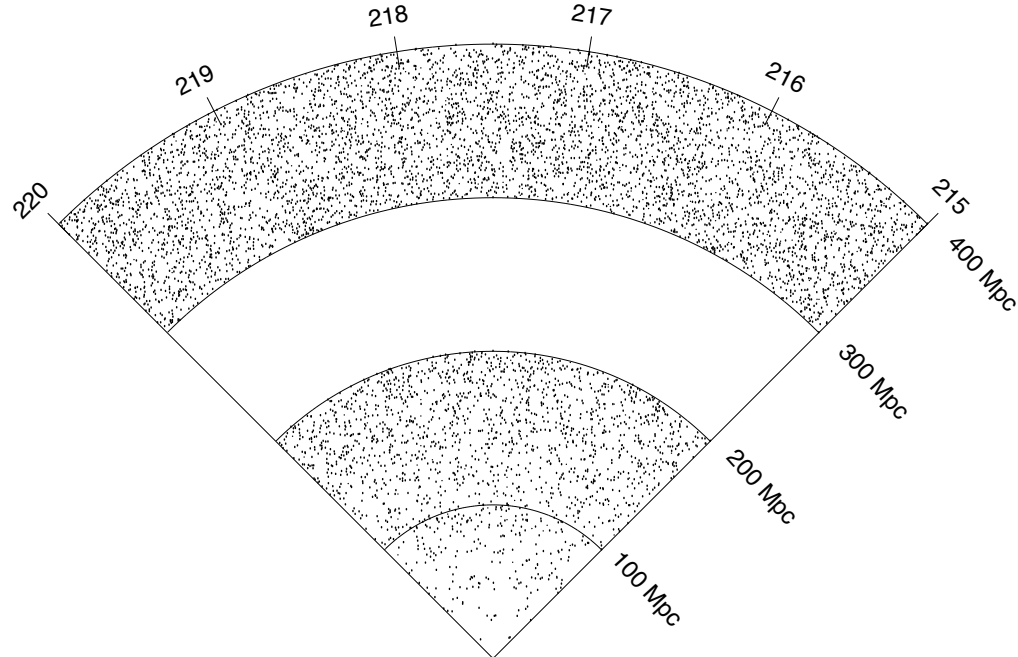
Figure 2.4 A uniform galaxy distribution with a cut-out void. The void center is located at 250 Mpc and has a radial width of 100 Mpc.
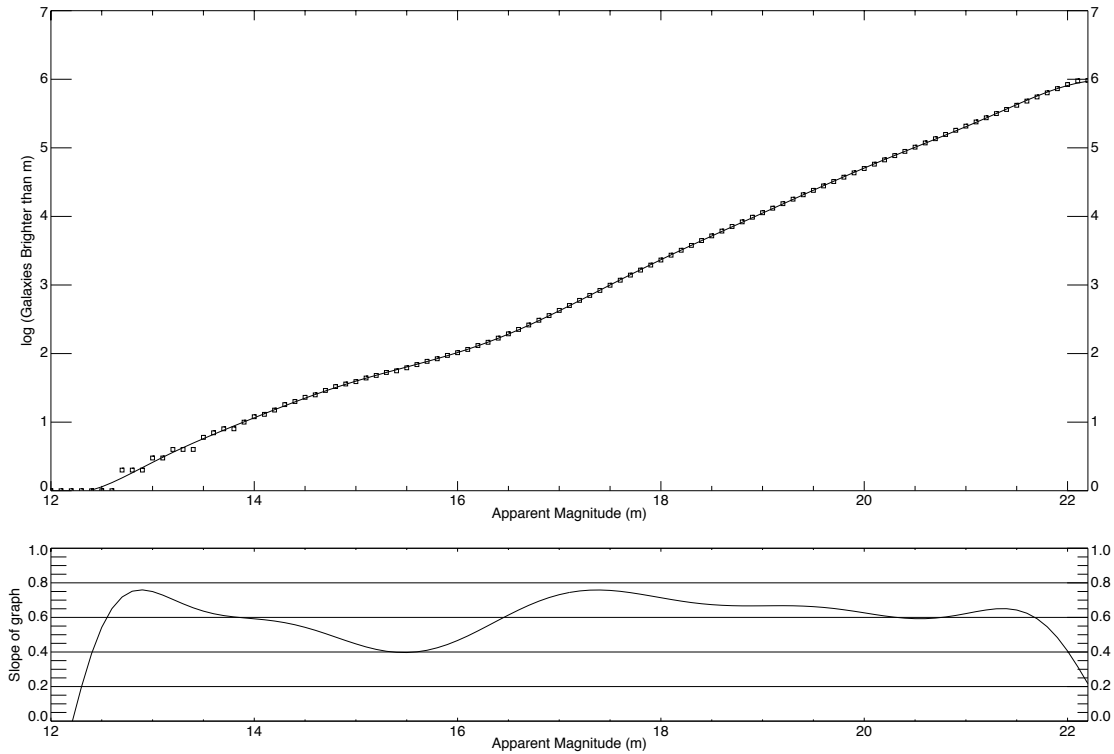
Figure 2.5 Wolf plot through a region containing a cut-out void. The void center is located at 250 Mpc and has a radial width of 100 Mpc.

2.1.2   Scooped-Out Voids

The next step up in realism was to change the way we create the voids so as to preserve the luminosity function, specifically ϕ^*. We call this method creating "scooped-out" voids. As with cut-out voids, we specify a void center and width along the radial direction. When we are assigning the d value to the galaxies in our model, if the calculated distance falls within the boundaries of the void(i.e. $d_{center} - \frac{1}{2}width \leq d \leq d_{center} + \frac{1}{2}width$), we randomly choose another distance (still using volume weighting) until the galaxy is not in the void. This preserves the luminosity function, produces a completely empty void with sharply defined edges, and has no clustering on the front or back edge of the void. Figure 2.6 shows the galaxy distribution with a scooped-out void located at $d_{center} = 250$ Mpc with $width = 100$ Mpc.

The plot does not look substantially different than Figure 2.4; the only difference is that there are more randomly-distributed galaxies on the front and back sides of the void.
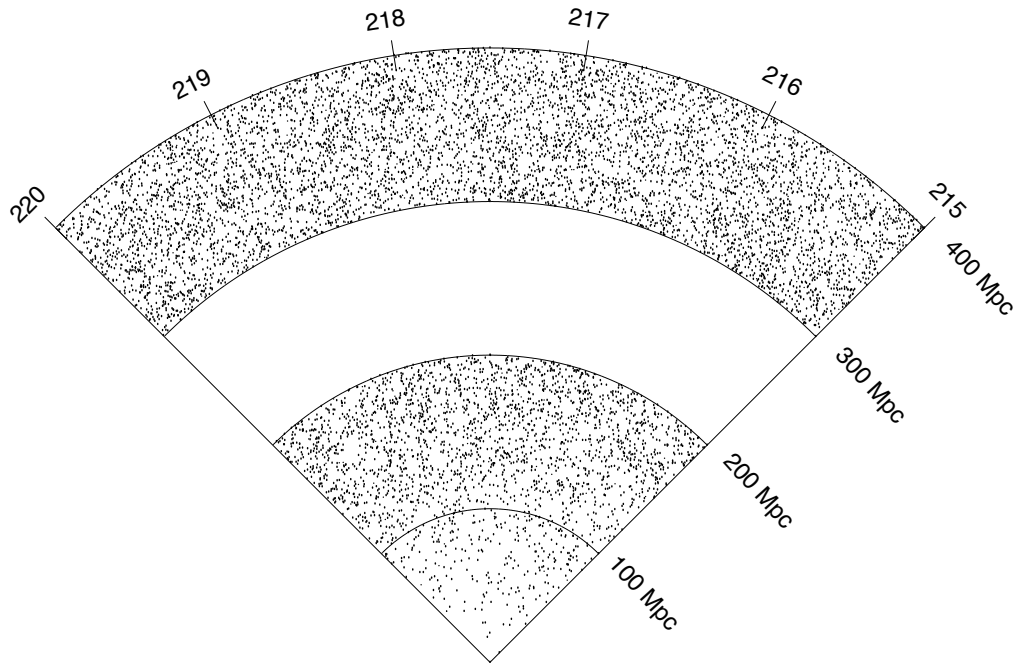
Figure 2.6 A uniform galaxy distribution with a scooped-out void. The void center is located at 250 Mpc and has a radial width of 100 Mpc.

The Wolf plot for this same void is shown in Figure 2.7. When the edge of the void is encountered, the slope decreases, indicating an under-density in the population of galaxies. The slope then crosses back above 0.6 before returning to the expected slope at the end. This pattern of decreasing slope followed by an increasing slope is characteristic of all scooped-out voids regardless of size or distance, although if the void is small or the center is far away, the deviation may be too small to differentiate from the statistical "noise." See Section 2.1.4 for more information. The cross-over point occurs at $m = 16.45$, which is also very close to the value for an M^* galaxy located at the same distance as the center of the void (0.31% error). This result, combined with that from Section 2.1.1, gave us great hope of being able to utilize the location of the crossover point to calibrate the distance to the center of the

void. In the next section, however, we show that there are limitations to how reliable this
method is.
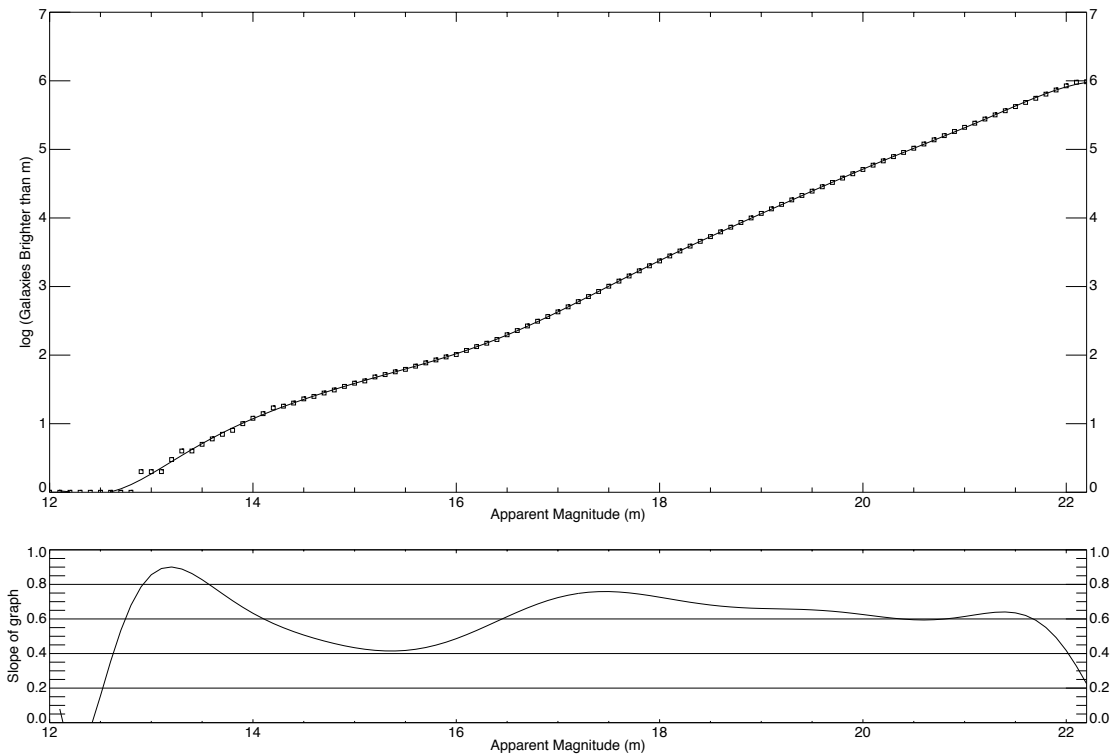
Figure 2.7 Wolf plot through a scooped-out void. The void center is located at 250
Mpc and has a radial width of 100 Mpc.

2.1.3 Built-Up Voids

The other void profile that we used is a "built-up" void. Again we start with a random
distribution of galaxies. We specify the center and width of the void as before, but this time
we also specify an edge width, which controls the size of the clustering region on the front
and back of the void. When the d values are assigned to each galaxy, they are checked to
see if they are located within the void, using the same criteria as the scooped-out void. If
they are, the galaxy is "moved" out of the void and into the wall (as defined by the edge
width), with galaxies located in the front half of the void being moved to the near wall

Density
(arbitrary units)
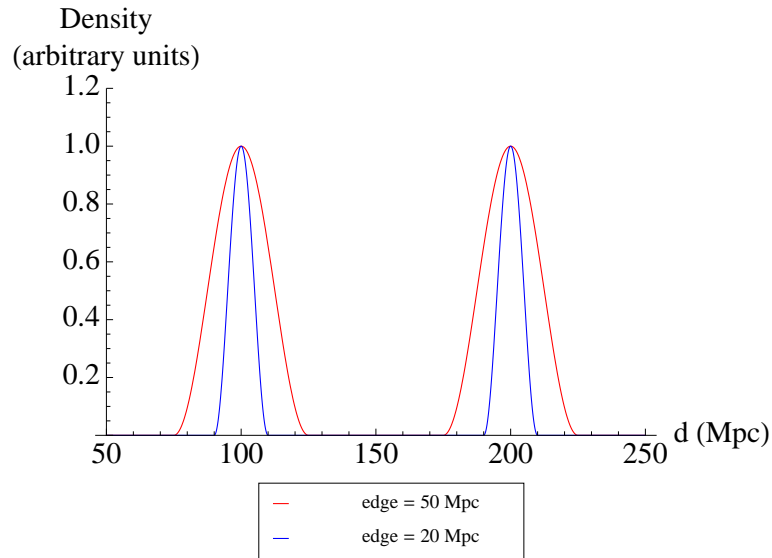
Figure 2.8 Density distribution in the edge of a built-up void illustrating two different edge widths.

and those in the back half of the void are moved to the far wall. The location within the edge is determined by a probability function of the form $\cos^2(x)$ such that the center of the distribution is at the edge of the void and also such that the distribution falls to zero by the time $x = void\_edge \pm \frac{1}{2}edge\_width$, as shown in Figure 2.8 for two different edge widths. The graph shows the probability that a galaxy, located initially within the void, will be moved to that new distance.

This produces a void that can have galaxies within it (depending on the edge width specified), but the galaxies are less likely to be found farther from the void edge. The edges are similar to the clustering seen in the walls of voids in observational surveys. The smaller the edge value, the more concentrated the galaxies will be. We tested this model with edge values ranging from 0 up to the void width (which could have galaxies in the center). To illustrate the model, Figures 2.9 and 2.10 show the galaxy distribution for a void with center of $d = 250$ Mpc, a width of 100 Mpc, and an edge width of 20 Mpc and 50 Mpc respectively. The corresponding Wolf plots are shown in Figures 2.11 and 2.12.
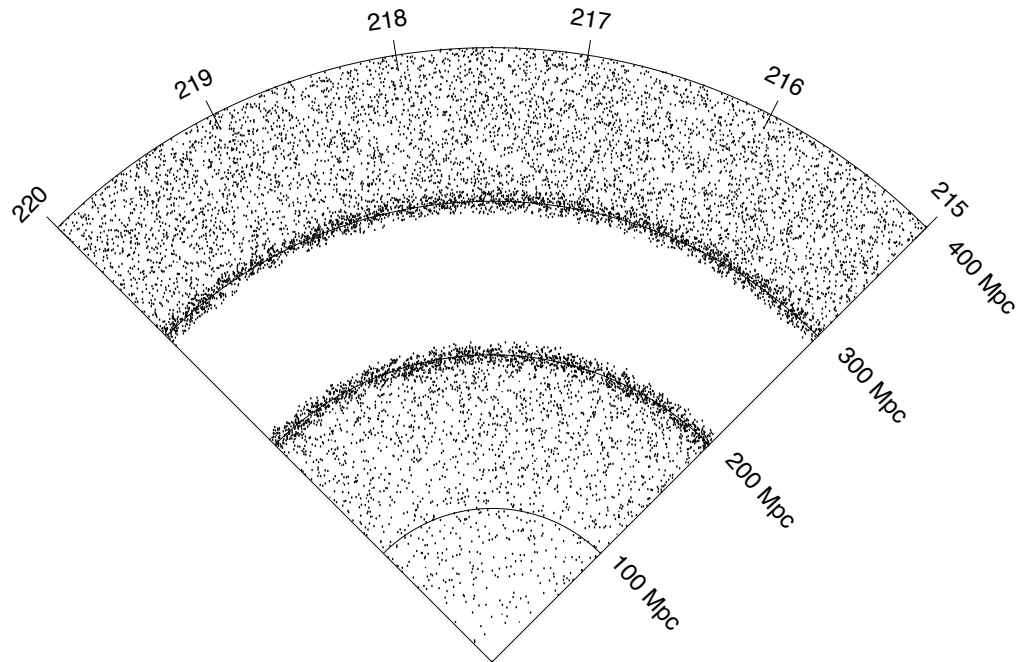
Figure 2.9 A uniform galaxy distribution with a built-up void. The void center is located at 250 Mpc, has a radial width of 100 Mpc, and an edge width of 20 Mpc.

Notice that the behavior of the slope is very different from that of the cut-out void. When the build up on the front edge of the void is encountered, the slope increases above 0.6. The slope then decreases to below the normal value, and eventually returns to 0.6. This is characteristic behavior for all built-up voids, regardless of width or distance. However, without the fit line, this effect is harder to separate from the statistical noise on the bright end. Also notice that the cross-over point is much brighter than in the previous two scenarios (they occur at $m = 15.00$ and $m = 14.80$ for an edge width of 20 Mpc and 50 Mpc respectively). In nature, there are instances where voids appear more like a scooped out void (e.g. Boötes) and instances where they appear either in front of or behind density enhancements (e.g. Coma). In using Wolf plots, which have only magnitudes but no distance information, we would not know *a priori* which kind of void we might be encountering. This makes us very cautious about using the cross-over point as a tool to locate the center of a void. On
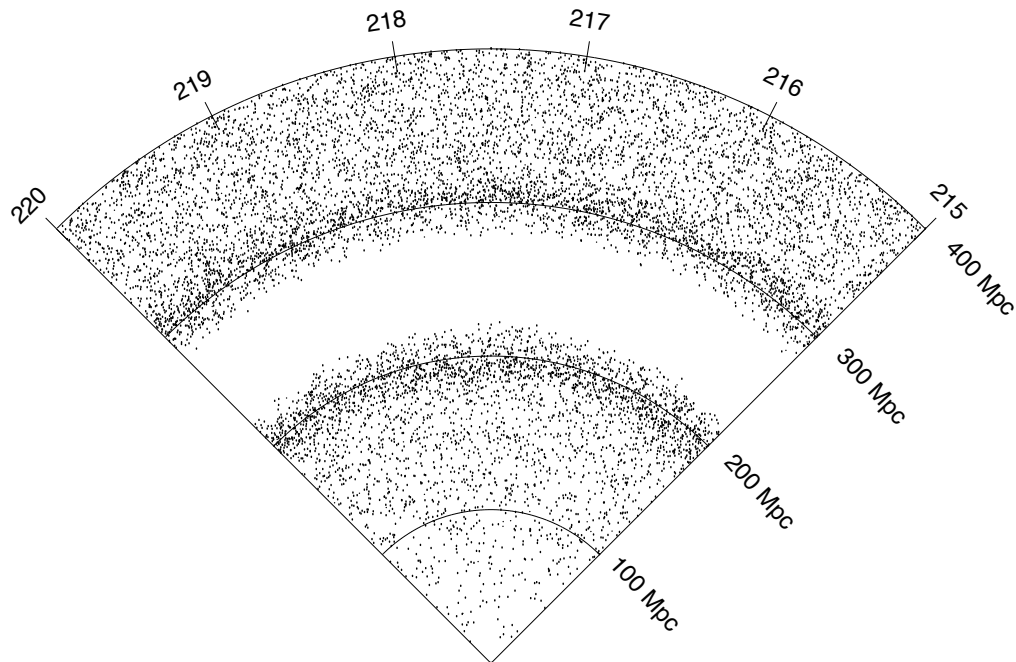
Figure 2.10 A uniform galaxy distribution with a built-up void. The void center is located at 250 Mpc, has a radial width of 100 Mpc, and an edge width of 50 Mpc.

the other hand, if we do encounter a slope fluctuation like that of Figure 2.7, it is a good indicator of a void, and should encourage follow-up investigation, preferably spectroscopic observations to pin down the location of the void.

2.1.4 Resolution

One limitation that became readily apparent is that the size of a void that was clearly discernible was strongly influenced by the distance to the void. The farther away the void is located, the smaller the amplitude of the deviation away from a slope of 0.6 that is seen. To illustrate this, we present a grid of plots of the slope from individual Wolf plots in Figure 2.13. The size of the void increases as you progress to the right from 50 Mpc to 150 Mpc in 25 Mpc increments, while the distance to the center of the void increases as you go down from 100 Mpc to 300 Mpc in 50 Mpc increments. All of the voids in this figure were scooped-out
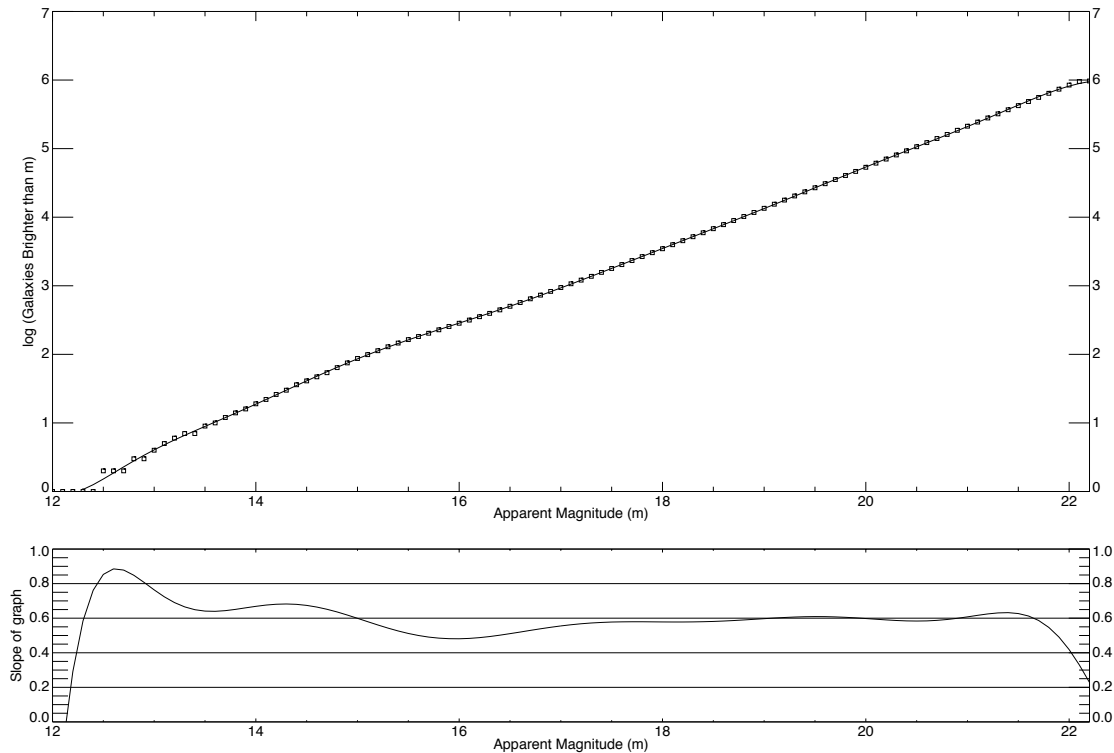
Figure 2.11 Wolf plot through a built-up void. The void center is located at 250 Mpc, has a radial width of 100 Mpc, and an edge width of 20 Mpc.

voids. In the description that follows, I will use a shorthand notation to specify the graph I am referring to: I will specify it as cXXXwYYY, where XXX will specify the distance to the center of the void and YYY will indicate the width of the void.

The first effect to notice is clearly seen by looking across the first row. This is the effect of the size of the void. In graph c100w025, and in all of the graphs, it is important to ignore the slope at the bright end of the graph; this is due to the small-number statistics. By magnitude 13, the plot has come to a slope of 0.6. However, the presence of the 25 Mpc wide void immediately causes the slope to drop lower. The slope then rises above 0.6 before coming back down to hover around it. As one moves across the row, an interesting development occurs. As the void gets wider, it limits the number of galaxies that can be found on the front side of the void. Because of this, there aren't enough galaxies in order
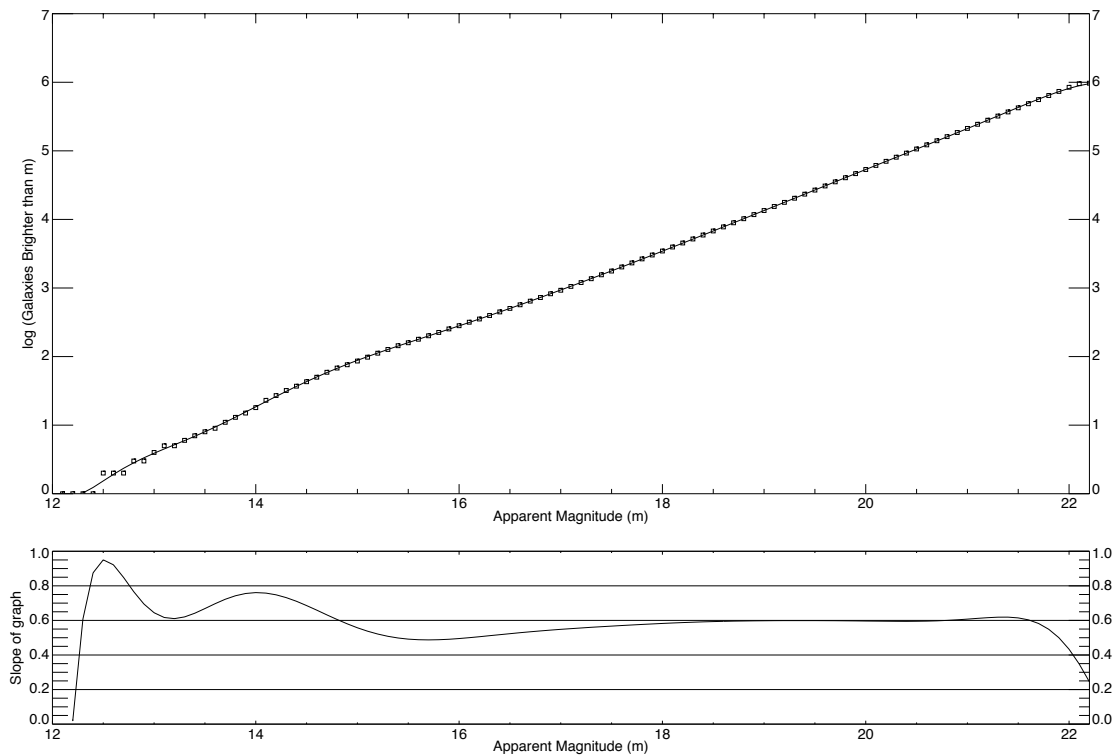
Figure 2.12 Wolf plot through a built-up void. The void center is located at 250 Mpc, has a radial width of 100 Mpc, and an edge width of 50 Mpc.

to even establish the baseline 0.6 slope. Therefore, it would be very difficult to utilize this technique to find voids located closer than about 150 Mpc away.

Looking at the second row, you can see a predictable effect. A small void, as seen in c150w025, produces a small-amplitude deviation about 0.6. At the other extreme, such as c150w150, shows a very large discrepancy in the slope.

You can examine the effect that the distance to the void center has by looking down a single column. A convenient reference point to use is to look at the point that the slope graph crosses over from being less than 0.6 to being greater than 0.6. We will call that the slope cross-over point. Because of the previously mentioned problems with the c100 graphs, we will look at the graphs from c150w150 to c300w150. In c150w150, the slope cross-over point occurs at $m \sim 14.75$. In c200w150, the cross-over point has shifted to $m \sim 15.50$, and
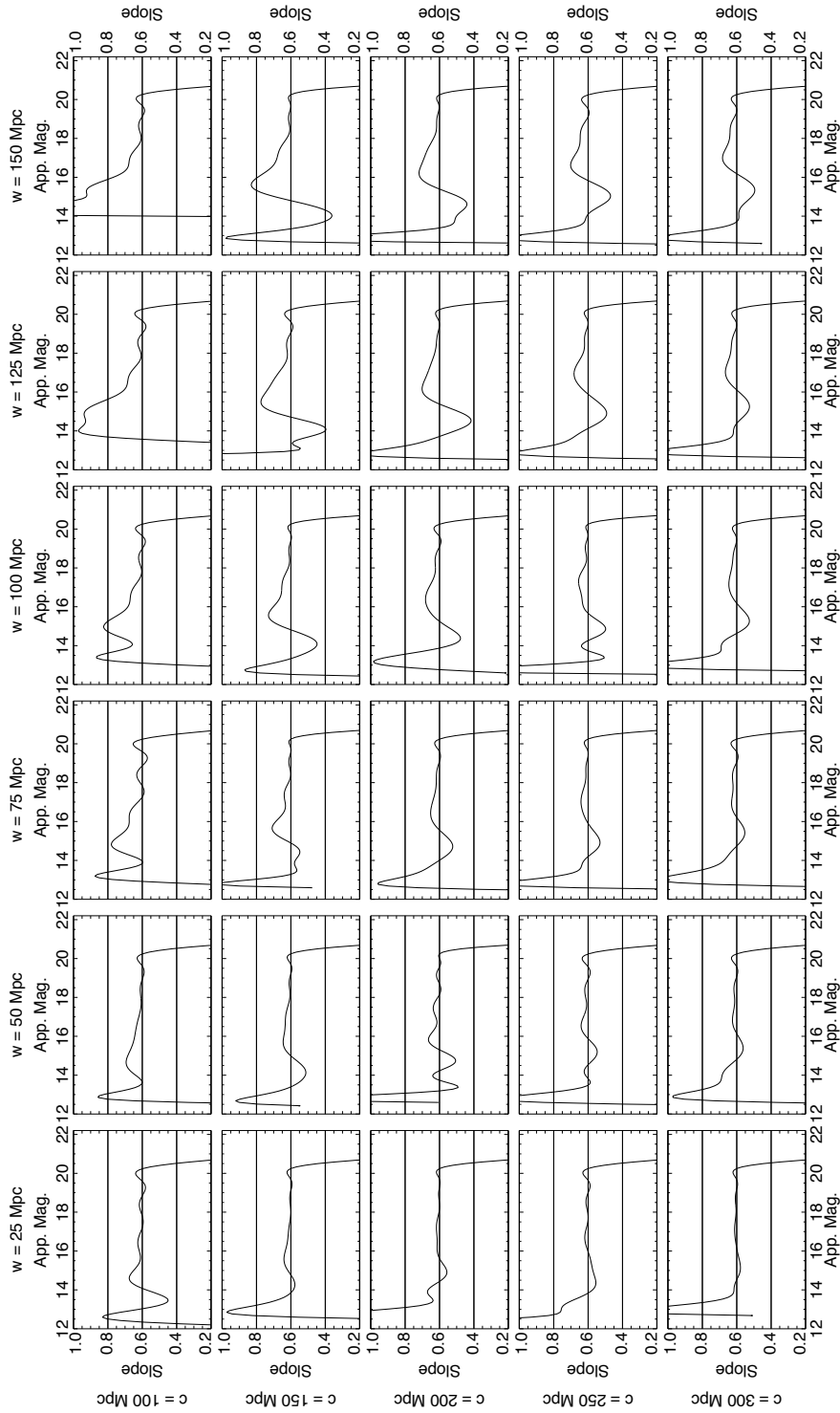
Figure 2.13 Grid of slope graphs showing Wolf plot resolution. The distance to the center of the void increases as you move down each row, and the width of the void increases as you move right across a row. All models were created using a scooped-out void.

by the time you get to c300w150, the cross-over happens at $m \sim 16.25$. Not surprisingly, this indicates that the farther away the void is located, the fainter the magnitude at which the cross-over occurs. See Table 2.1 for a summary.

Table 2.1 Observed versus calculated crossover values in Wolf plots. Distance is measured to the center of the void, the observed crossover is measured from Figure 2.13, the calculated crossover is determined using Equation 2.1, and the percent difference is the difference between the observed and calculated values.

Distance (Mpc)	Observed Crossover (mag)	Calculated Crossover (mag)	% Difference
150	14.75	15.15	2.64%
200	15.50	15.78	1.77%
300	16.25	16.66	2.46%

There is another crucial effect that the distance to the void center creates. This is the size (amplitude) of the deviation away from a slope of 0.6. In c150w150, the peak-to-peak amplitude is about 0.5. However, by c300w150, the amplitude is only about 0.2. The practical implication of this is that the farther away the void is located, the harder it becomes to differentiate a small void from the fluctuations in our fit curve. Using the slope of the actual data is worse because of the rapid fluctuations in the slope. Therefore, the farther out we probe with this technique, the more blunt the probe becomes.

2.2 Application to the Millennium Run

In order to determine whether or not our technique was viable on an archival observational data set, such as the SDSS, we decided to first apply it to a much more detailed galaxy model. We chose the Millennium Run simulation of Springel et al. (2005) because it represented the largest dark-matter simulation to date. The Millennium Run was based on the 2-degree Field

Galaxy Redshift Survey (2dFGRS) described in Colless et al. (2001), Folkes et al. (1999), and Cole et al. (2005), and the first-year Wilkinson Microwave Anisotropy Probe (WMAP) data in Bennett et al. (2003) and Spergel et al. (2003).

Springel et al. used the temperature variations from the WMAP data to infer primordial density fluctuations. They traced more than 10^{10} dark matter particles, each with a mass of $8.6 \times 10^8 h^{-1}$ M$_\odot$ in a periodic cubical region $500 h^{-1}$ Mpc on a side. They began the simulation when the universe was about 12 million years old and allowed the particles to interact gravitationally until the present day. A slice through the dark matter distribution at $z = 0$ is shown in Figure 2.14.

125 Mpc/h
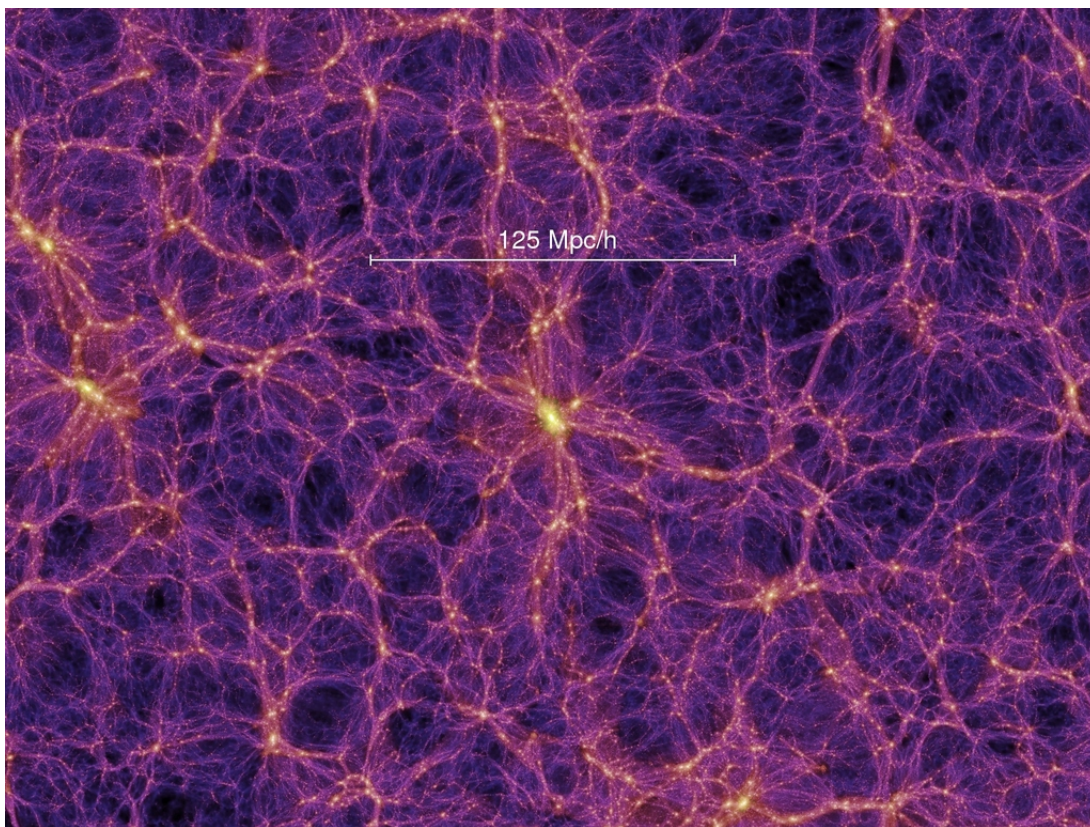
Figure 2.14 A slice through the dark matter distribution of the Millennium Run simulation of Springel et al. (2005). Image courtesy of the Virgo Consortium.

Once the dark matter simulation was completed, it was time for them to populate the resulting dark matter haloes with galaxies. The process is outlined in Springel et al. (2005)

and Croton et al. (2006). This resulted in ~ 9.5 million galaxies with $M_r \leq -16$. The corresponding slice through the galaxy distribution can be seen in Figure 2.15.
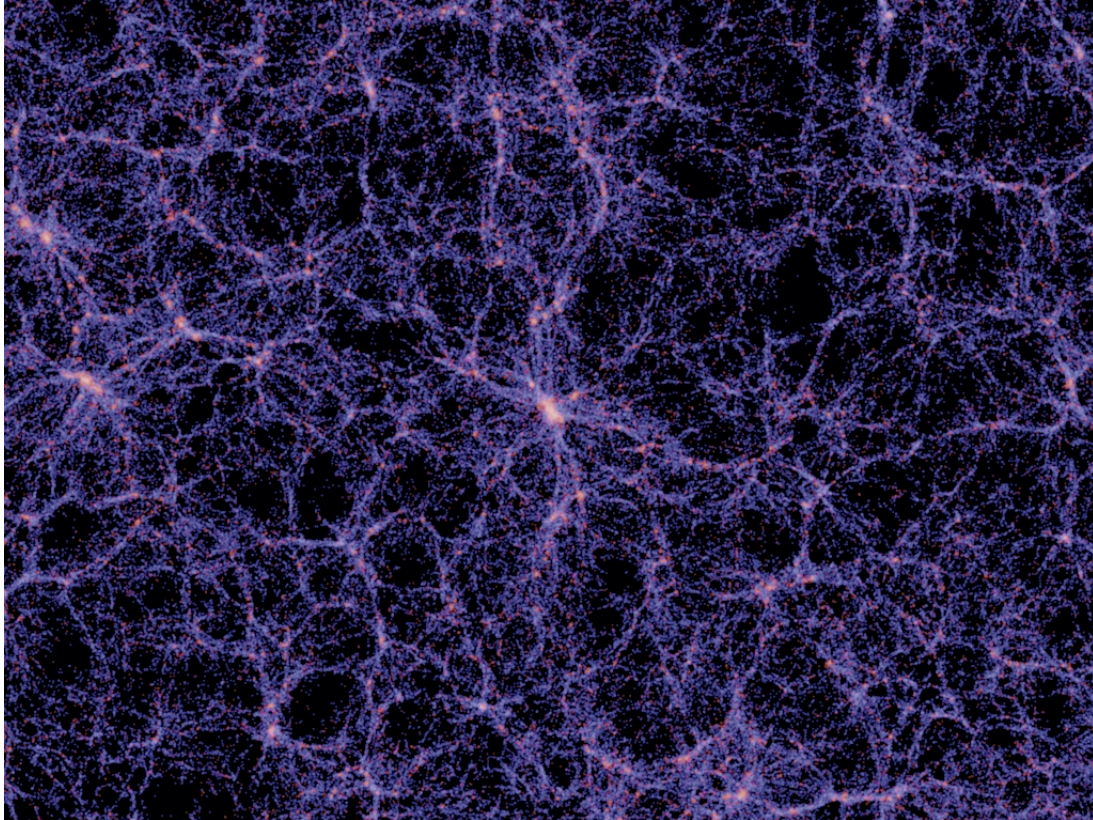
Figure 2.15 A slice through the galaxy distribution of the Millennium Run simulation of Croton et al. (2006). Image courtesy of the Virgo Consortium.

I wrote a program that would allow us to look through the Millennium Run galaxies as slices through the volume (see Figures 2.16, 2.22, and 2.25 for examples). We created slices that were 10 Mpc thick in the z-coordinate, with each slice incrementing z by 5 Mpc. This allowed the slices to overlap so we could get a three-dimensional feel for the voids. We could then visually search for promising voids. We wanted underdense regions that were located within 250 Mpc of the origin, based on the discussion of resolution in Section 2.1.4. The program would then provide the appropriate right ascension and declination to generate a wedge plot centered through the void. The voids that were chosen are summarized in Table 2.2. We will look at each of these voids individually to learn what we can about this

technique. We will refer to the voids by the slice in which the center of the void is located.

Table 2.2 Voids that were visually selected from the Millennium Run simulation of Springel et al. (2005) for analysis with galaxy number counts. The name used to reference the void comes from the slice through the data set on which it is found, X, Y, Z are the rectangular coordinates of the center of the void in Mpc, D is the distance to the center of the void in Mpc, RA is the right ascension to the center of the void, and Dec is the declination to the center of the void.

Name	X	Y	Z	D	RA	Dec
Z011	115	50	60	139.0	23.50	25.57
Z032	150	80	165	236.9	28.07	44.14
Z037	40	40	190	198.2	45.00	73.42

2.2.1 Void Z011

We begin our analysis with Z011, which is shown in Figure 2.16. The red ellipse indicates the void. The corresponding wedge plot is shown in Figure 2.17. Note that in the wedge plot, some of the structure is artificially spread out in right ascension due to the scale factor used to increase the size of the wedge so that the individual galaxies would be more clearly visible; this will not impact the results in any way. It also bears pointing out that the wedge plot avoids the wall on the front side of the void that is shown in Figure 2.16 because the line of sight is from the origin (0, 0, 0), which means that we are looking from below the void, passing through it, and continuing on above the void.

When looking at the Wolf plot for Z011 (Figure 2.18), the first thing that should jump out at you is that the galaxy counts don't begin until magnitude 14, almost two magnitudes fainter than our model scooped-out void (Figure 2.7). Looking again at the wedge plot in Figure 2.17, it is obvious that there are very few galaxies on the front side of the void, which would have helped populate the bright end of the Wolf plot. However, the pattern that we observed for our scooped-out voids (slope below 0.6 then switching to be above 0.6) is still
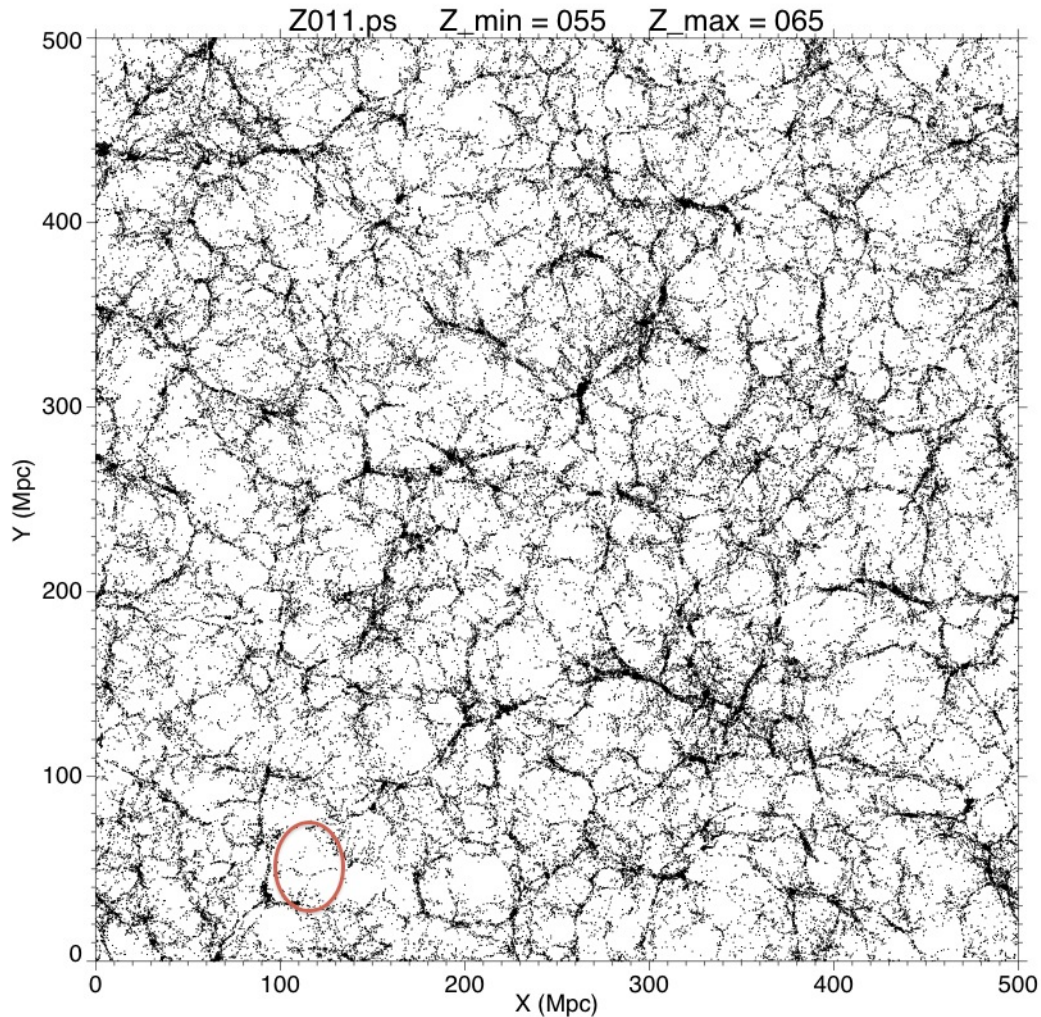
Figure 2.16 A 10 Mpc slice through the Millennium Run data set showing the location of void Z011.

clearly visible. One interesting phenomenon that may not be immediately obvious is the location of the slope cross-over point. The cross-over occurs at about $m = 15.80$, which is almost a full magnitude fainter than would be expected for a void centered at 139 Mpc. We explored two possible explanations for the discrepancy.

The first possible explanation for the discrepancy in the cross-over point that we explored was the Schechter function parameters for the Millennium Run simulation. Since the galaxies were not generated using a Schechter function, it was reasonable to wonder if differences
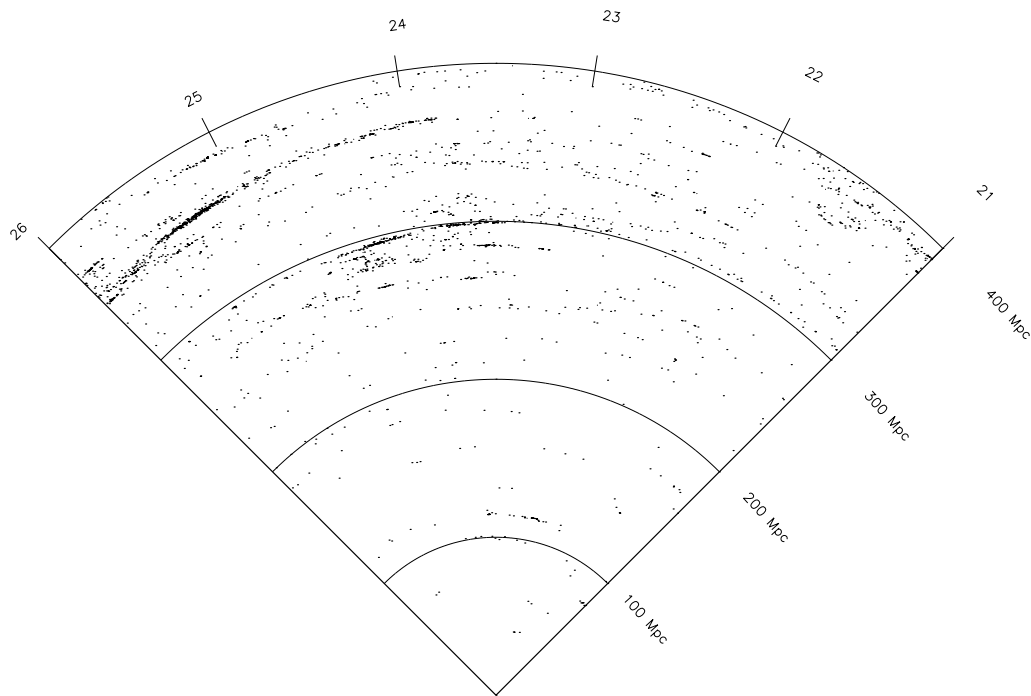
Figure 2.17 Wedge plot from the origin through void Z011.

in models could lead to divergent values that might explain these results. We created a histogram of all galaxies in the simulation binned in 0.1-magnitude bins. We then plotted the data and fit a Schechter function to the data using a Bayesian fit (see Figure 2.19). The values obtained for the parameters are as follows: $\phi^* = 0.9h^3$ Mpc^{-3}, $M^* = -20.90 + 5\log(h)$, and $\alpha = -1.115$, again with $h = 1$. With the exception of ϕ^*, these results are not significantly different from the values of Montero-Dorta and Prada (2009), and are insufficient to explain the discrepancy.

The other possible scenario we investigated was the possibility that there could exist a larger void farther along our line of sight that could have led to the unexpected faintness of the cross-over. Visually inspecting the slices through the Millennium Run did not reveal a larger void. However, closer inspection of the wedge plot (Figure 2.17) provides some insight. Although the void we chose is centered at a distance of 139 Mpc, it is clear from the wedge plot that the underdensity actually extends much farther than that, perhaps out to as much
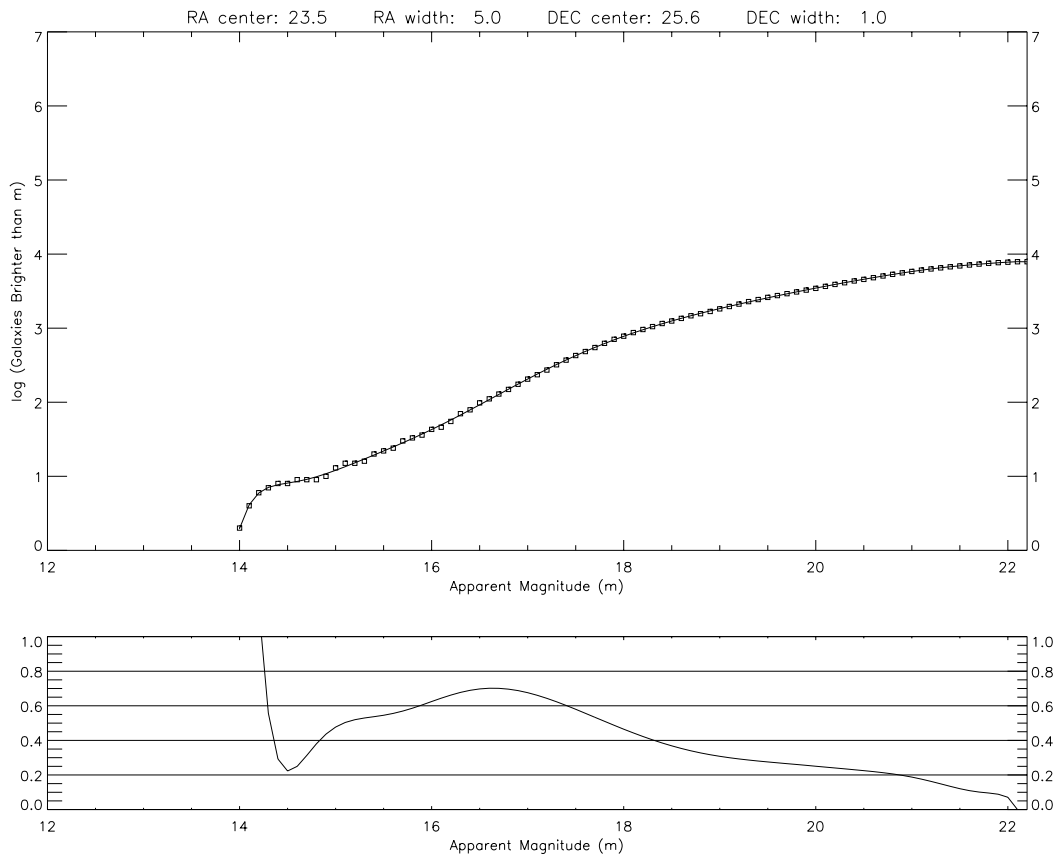
Figure 2.18 Wolf plot through void Z011.

as 270 Mpc. This would give a void center closer to 200 Mpc, which would correspond quite well to the cross-over predicted distance of 218 Mpc. The reason the void appears larger on the wedge plot than in the map is most likely due to the narrowness of the line-of-sight. The field of view seems to have found a small corridor of decreased density, even though a visual inspection of the surrounding galaxy maps does not reveal a prominent void. Therefore, it is always a good idea to use a wedge plot of the galaxy locations, if possible, in order to check on the distribution and ensure the line-of-sight does not find a similarly empty corridor.

To test this idea, we used the same path through the void, but increased the size of the line-of-sight to 10° in right ascension and 5° in declination. The wedge plot for this line-of-sight is shown in Figure 2.20. Notice now that the region from 200 to 300 Mpc has become
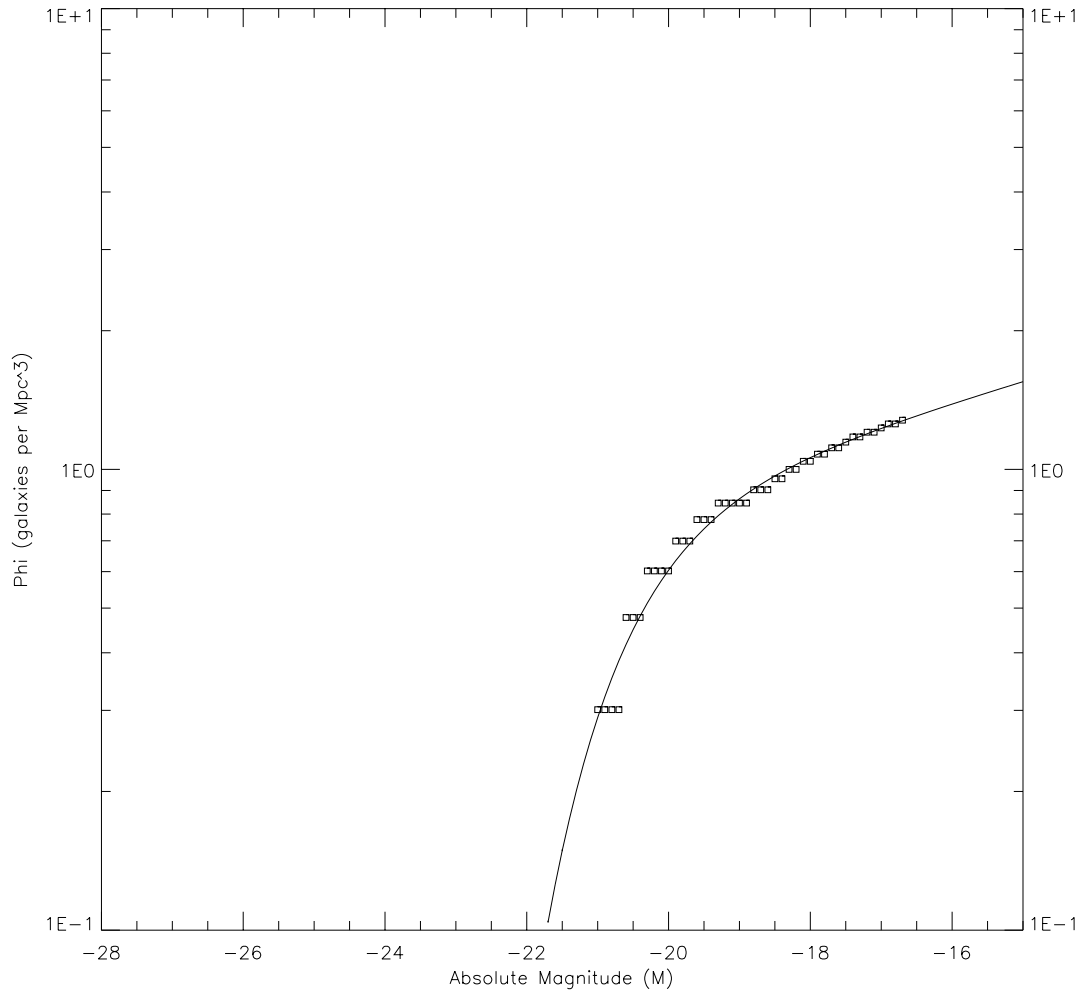
Figure 2.19 The binned galaxy counts for the Millennium Run simulation (squares) along with the best-fit Schechter function. The parameters for the function are $\phi^* = 0.9h^3$ Mpc^{-3}, $M^* = -20.90 + 5\log(h)$, and $\alpha = -1.115$, with $h = 1$.

much more populated, which is what we expected since we didn't see any noticeable voids at that distance in the maps. The area from 100 to 200 Mpc still looks underdense.
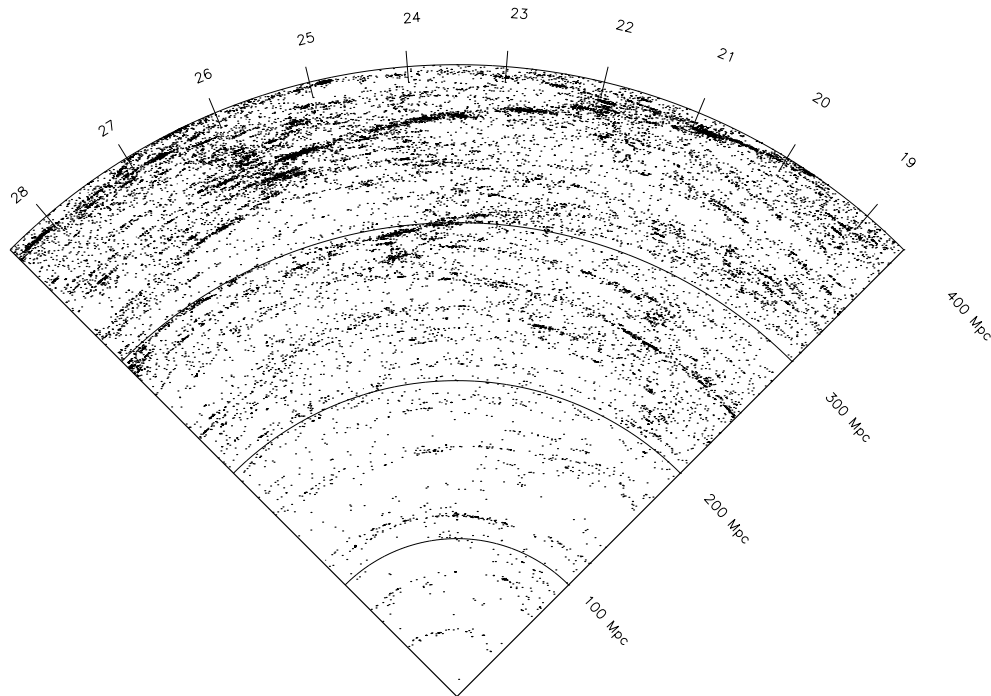
Figure 2.20 A larger line-of-sight through void Z011, corresponding to 10° in right ascension and 5° in declination.

The Wolf plot for this expanded region (Figure 2.21) has an interesting plateau from $m = 14.3$ to $m = 15.3$ at a slope of 0.6 in between the variations expected for a void. Because of this pause, it is difficult to infer a single cross-over point, making a precise distance calculation impractical. However, we can say that the distance should be confined to the range of 110 to 174 Mpc, which in the average give 142 Mpc (only about a 2% error from the expected value of 139 Mpc).

2.2.2 Void Z032

The next void we looked at was located in slice Z032. This void is shown in Figure 2.22. It is located almost 100 Mpc farther away than void Z011, which puts it at about the limit of

RA center: 23.5   RA width: 10.0   DEC center: 25.6   DEC width: 5.0
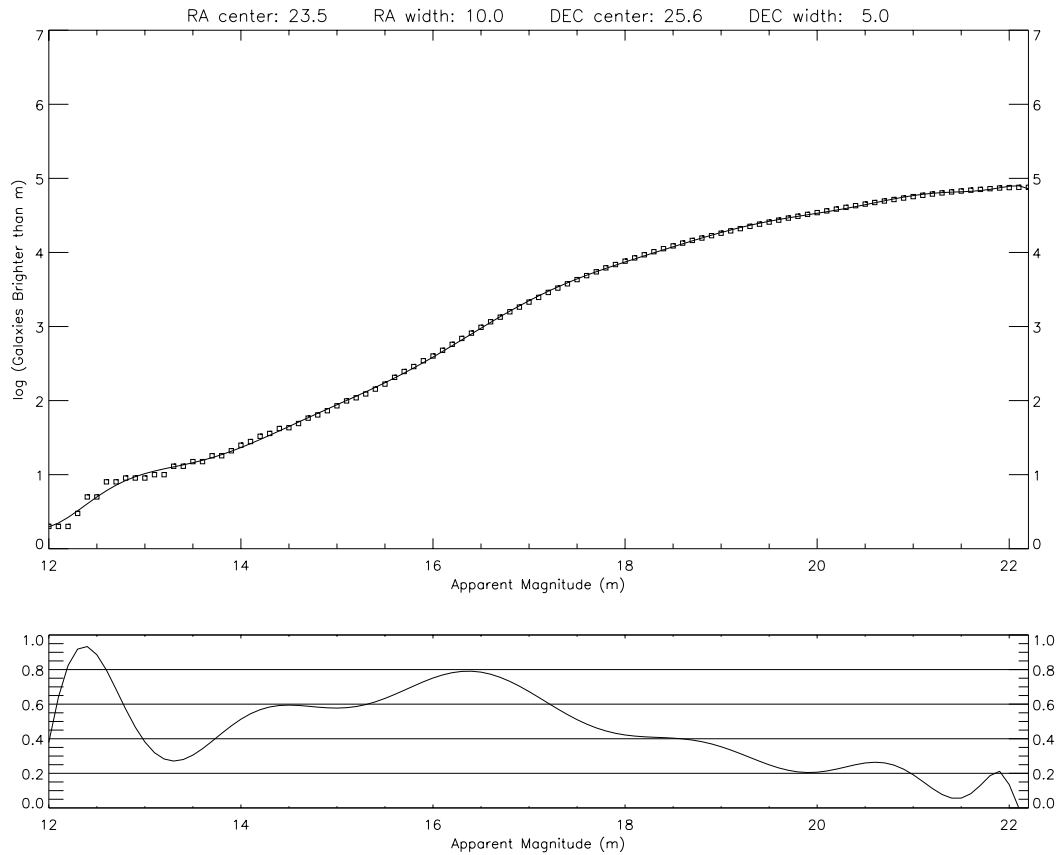
Figure 2.21 Wolf plot for the expanded region through void Z011.

where we hoped to be able to detect reasonably sized voids.

Looking at the wedge plot for this line-of-sight (Figure 2.23) reveals that we may have a problem. It appears that the whole region from the origin out to about 230 Mpc is underdense. Figure 2.24 confirms our suspicion as the slope never reaches 0.6. Our polynomial fit to the data is also not very good. There are a very small number of galaxies at the bright end of the graph and then the data seems to oscillate a bit from about $m = 14.5$ to $m = 18.0$ as the line-of-sight crosses small regions with and without galaxies.

We attempted to use a larger field of view to see if that would help populate the front end, similar to what we did in Section 2.2.1, but it didn't help. It appears that regardless of how big the angle is (within reason), there just aren't that many galaxies located in such
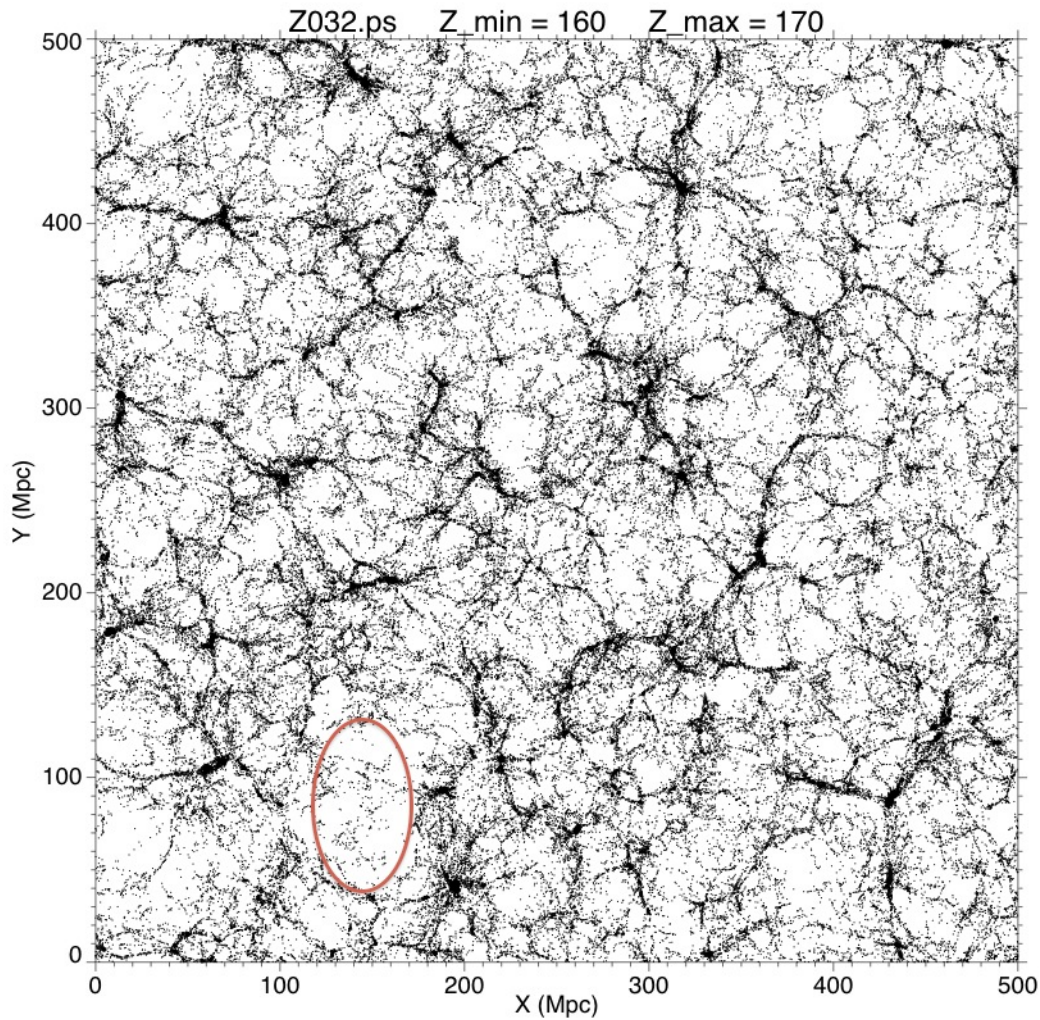
Figure 2.22 A slice through the Millennium Run simulation showing the location of void Z032.

a small volume on the front edge. The wolf plot for this larger region still never reached a slope of 0.6, indicating that in this situation, our technique would have failed to isolate the void.

2.2.3   Void Z037

The last void we considered is shown in Figure 2.25 and is referred to as void Z037. This void has a center located midway between the distances of the previous two voids. The
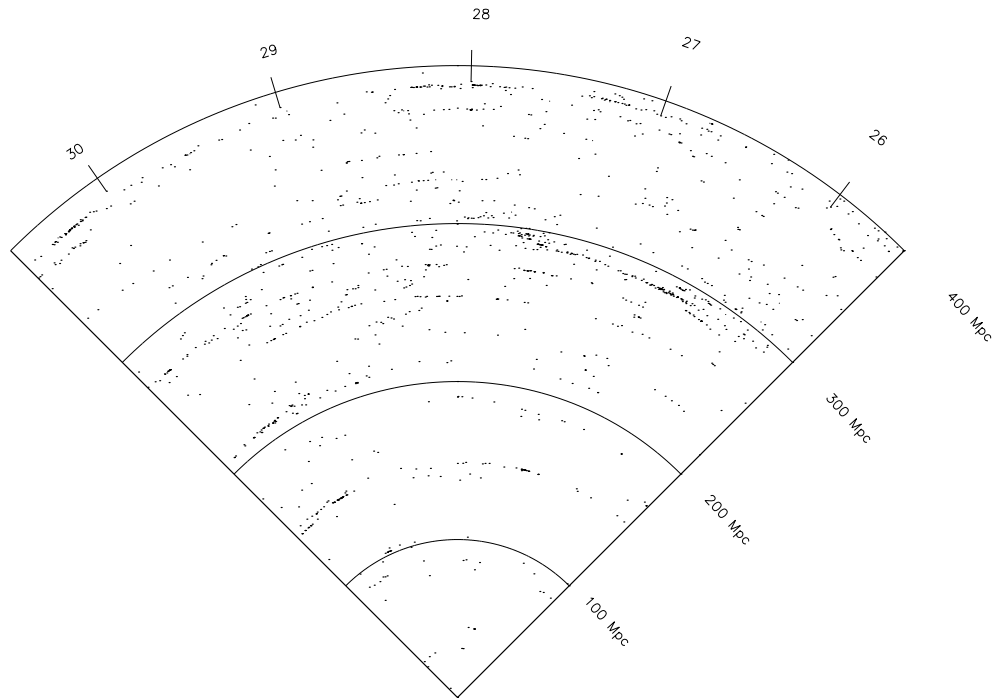
Figure 2.23 Wedge plot for the line-of-sight through void Z032.

wedge plot (Figure 2.26) shows that there are very few galaxies on the front side of the void, raising the same concerns as Section 2.2.2. However, the back side of void Z037 is very well delineated, and the front side does seem to have a fairly distinct edge, making this perhaps the closest match yet to our simple scooped-out void model of Section 2.1.2.

The Wolf plot for this line-of-sight is truly unexpected. It is shown in Figure 2.27. The first thing to notice is that there are only three galaxies by a magnitude of about 15.5, giving this region the least populated bright end. While the shape of the slope is very classical for a scooped-out void, the amplitude is huge. It falls and rises by more than ± 0.4, which is larger than void c200w150 in Figure 2.13. Since the void is centered at about 198 Mpc, this would have to correspond to a void of at least 200 Mpc in diameter.
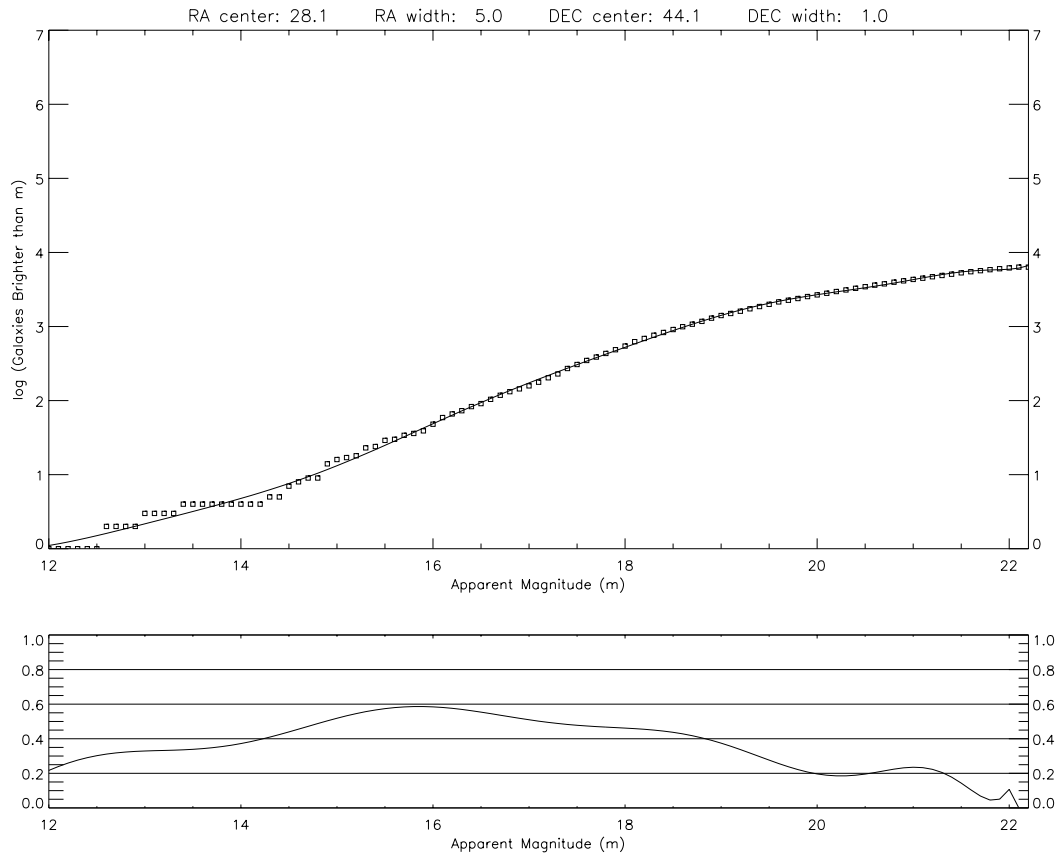
Figure 2.24 Wolf plot through void Z032.

2.3   Application to the Sloan Digital Sky Survey

We applied our technique to the Sloan Digital Sky Survey (SDSS) in order to search for voids. There were two approaches that we could have chosen with this data set. The first option was to only use the spectroscopic data which had been obtained, giving us a three-dimensional data set to work with. This would only give us a small fraction of the galaxies in the sample region. The second option was to use the complete photometric data set. This would give us a much larger number of galaxies, including going to a faint magnitude completeness limit of $m_r = 22.2$ (compared to the spectroscopic data which has a magnitude completeness limit of $m_r = 17.7$), but we lose the three-dimensional nature of the data. In the end, we opted for the latter, since the Wolf plots are not dependent of having the redshift
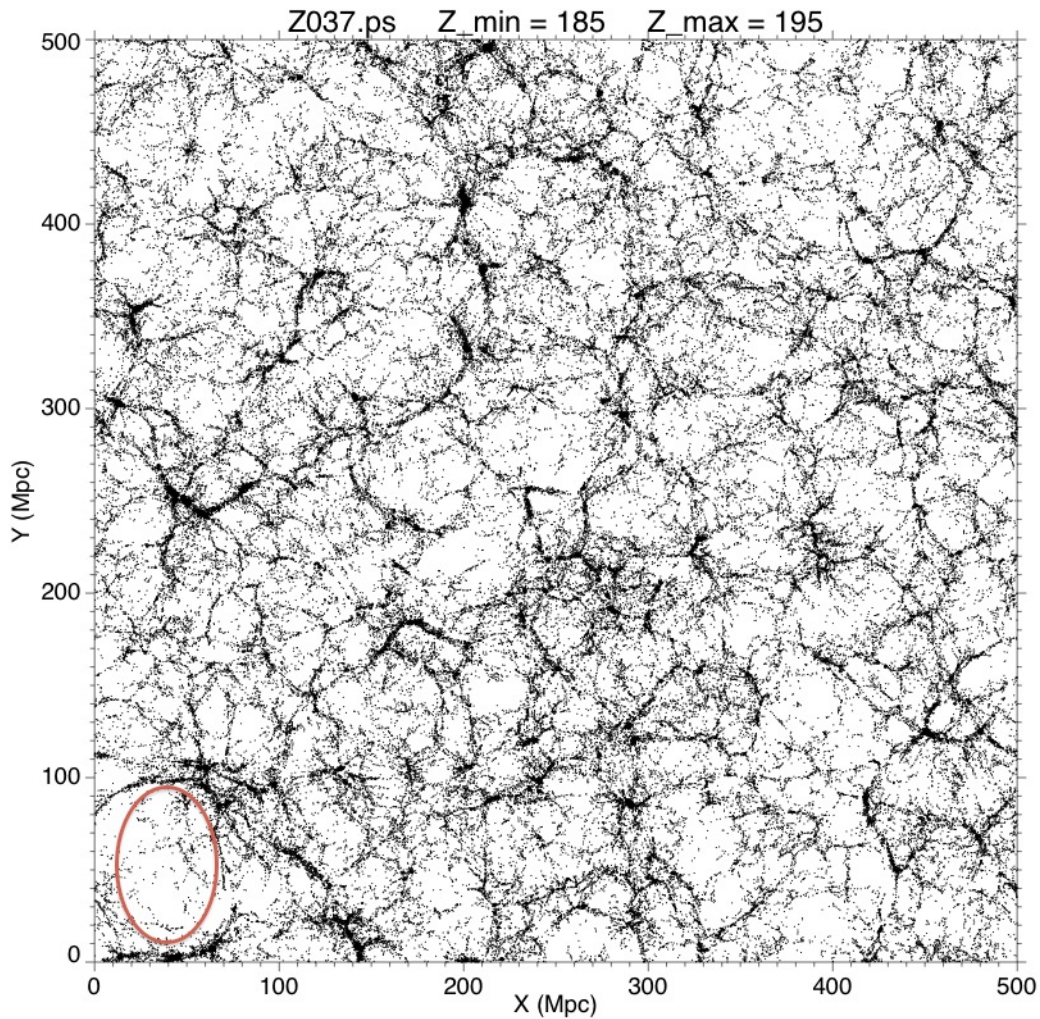
Figure 2.25 A slice through the Millennium Run simulation showing the location of void Z037.

information.

We ran the Wolf plots for various lines-of-sight through the data and found enormous plateaus. The plateaus were so large that we could not duplicate them with our simple galaxy model. Two examples are shown in Figures 2.28 and 2.29. As you can see, the plateaus stretch for over five magnitudes, which would indicate a void of approximately 125 Mpc in diameter. The void would begin a mere 15 Mpc from us and extend out to about 140 Mpc. However, in our models, it is impossible to generate enough bright galaxies to
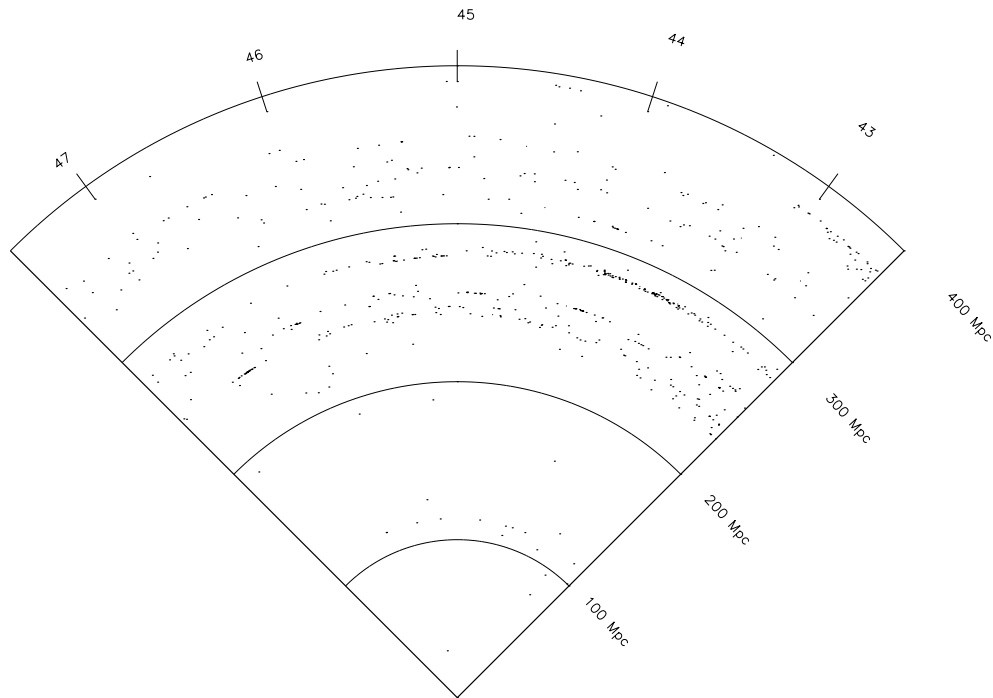
Figure 2.26 Wedge plot along the line-of-sight through void Z037.

sufficiently anchor the front edge of the void in such a small volume. This gave us the clue
we needed to understand this behavior.

In trying to make sense of the discrepancy between the Wolf plots of the SDSS data
and those of our model, we looked at the list of objects identified as galaxies within the
region of interest in the SDSS. Since this is a two-dimensional, photometric data set, we
do not have the redshifts of the individual objects, but we took the right ascension and
declination of the 2,000 brightest objects and ran it through the SIMBAD database. Only
200 of the objects (10%) came back as galaxies or unknown objects; the other 90% came back
as stars. It appears that the SDSS pipeline does not sufficiently account for the possibility of
saturated bright stars appearing to be an extended object like a galaxy. Note that the SDSS
spectroscopic data set does not suffer from this problem as the spectra of stars and galaxies
are easily differentiated. However, the trade-off is that with the spectroscopic data, you lose
over 90% of the total number of objects, not just the stars. This makes it very difficult to
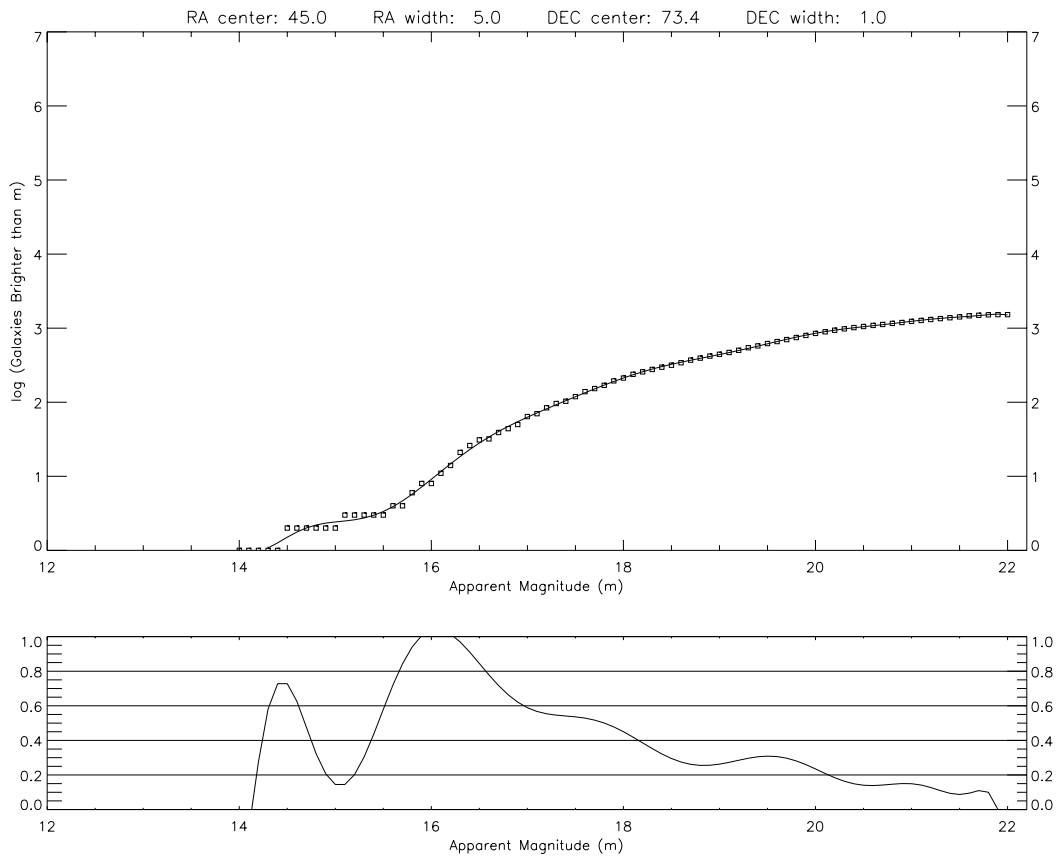
Figure 2.27 Wolf plot through void Z037.

find any void signals at all.

This contamination of the data set completely masks any voids that may be present along the selected line of sight. However, it does appear that the technique is working. It is finding a void, but the void it is finding is a dearth of stars as the line of sight leaves the galaxy.

In order to combat this problem, we turned to a different source of data from the SDSS. The NASA/IPAC Extragalactic Database (NED) is attempting to collect, correlate, and disseminate data on as many extragalactic sources as they can. As part of their efforts, they imported the data from the SDSS and cross-checked it against existing galaxies in their database.

In conversations with the director of the project, Joe Mazzarella, he informed us of
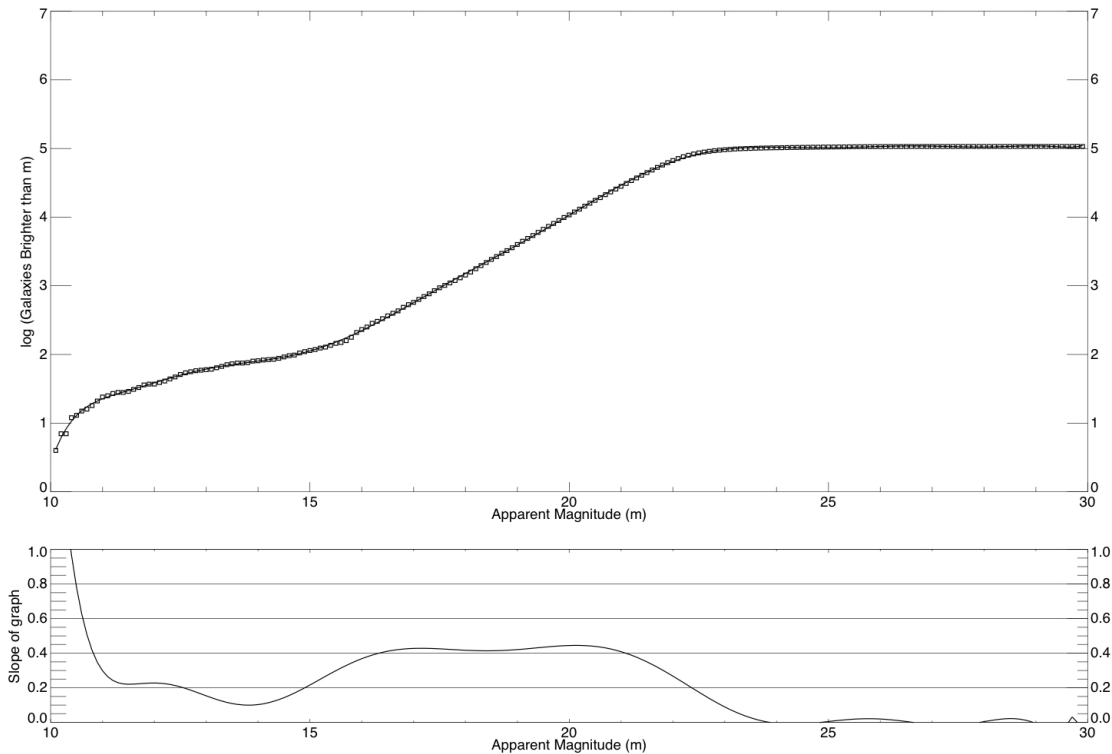
Figure 2.28 Wolf plot of SDSS data centered on coordinates RA: 155°, Dec: 3.5°. The large plateau turns out to be due to contamination of the galaxy data set by saturated bright stars.

another problem with the SDSS that could also have played a role. When bright galaxies are run through the SDSS data pipeline, the software looks for regions of peak brightness. For a distant object, that is not really a problem since the nucleus is the brightest region. However, in extended nearby bright galaxies, it is possible that there could be multiple regions of peak brightness near the center. The pipeline interprets this as being the result of multiple non-resolved galaxies and forcibly splits them apart, in essence shredding a galaxy. The NED team was quite successful at restoring these objects to their former and proper whole status.

With the help of Joe Mazzarella and Rick Ebert at NED, we were able to download a large segment of SDSS data with a line of sight centered on the Boötes void (central coordinates: 217.0° right ascension and 43.0° declination). We chose a region that was 10° wide in right
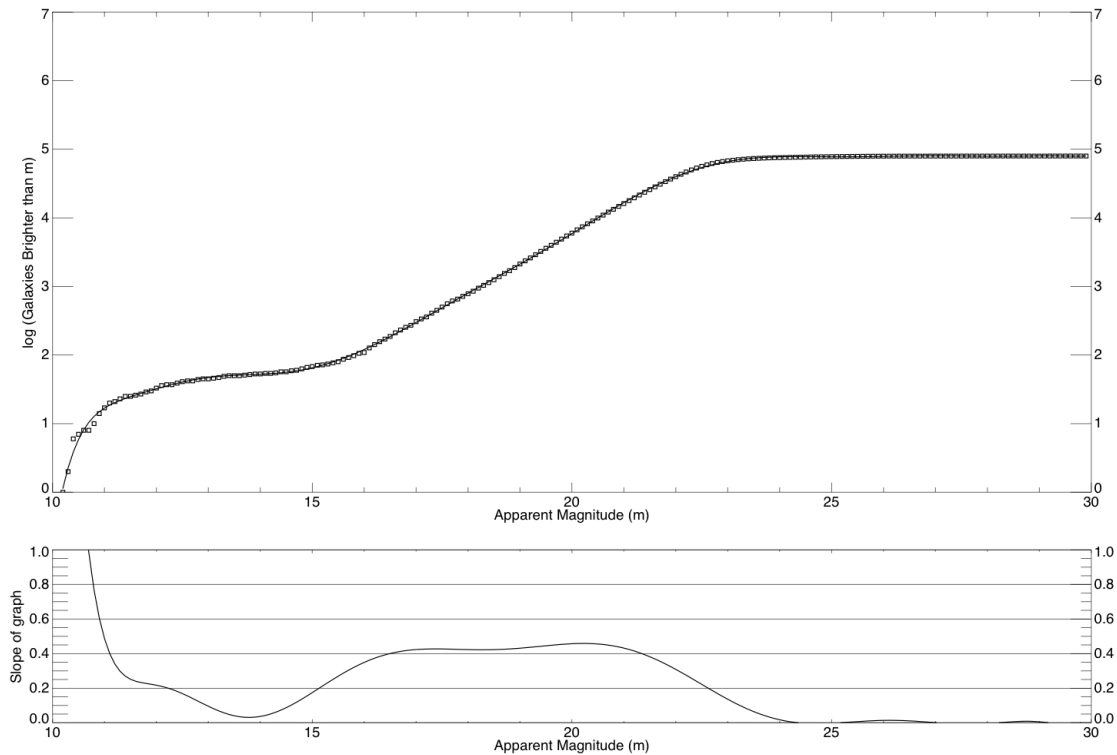
Figure 2.29 Wolf plot of SDSS data centered on coordinates RA: 217°, Dec: 43°. The large plateau turns out to be due to contamination of the galaxy data set by saturated bright stars.

ascension and 5° in declination. The Wolf plot for this data set is shown in Figure 2.30.

The Boötes void is the largest void that has been measured to date. It is not very surprising that the slope drops to about 0.4 and just stays there. The number of galaxies tapers off before the back side of the void is reached.

2.4   Conclusions

Our analysis of Wolf plots and galaxy number counts show that this technique could potentially be useful for probing the structure of the nearby universe. The methodology requires neither redshift information nor multi-filter photometry. This combination of features makes it a very useful tool in conjunction with large observational datasets. However, care must
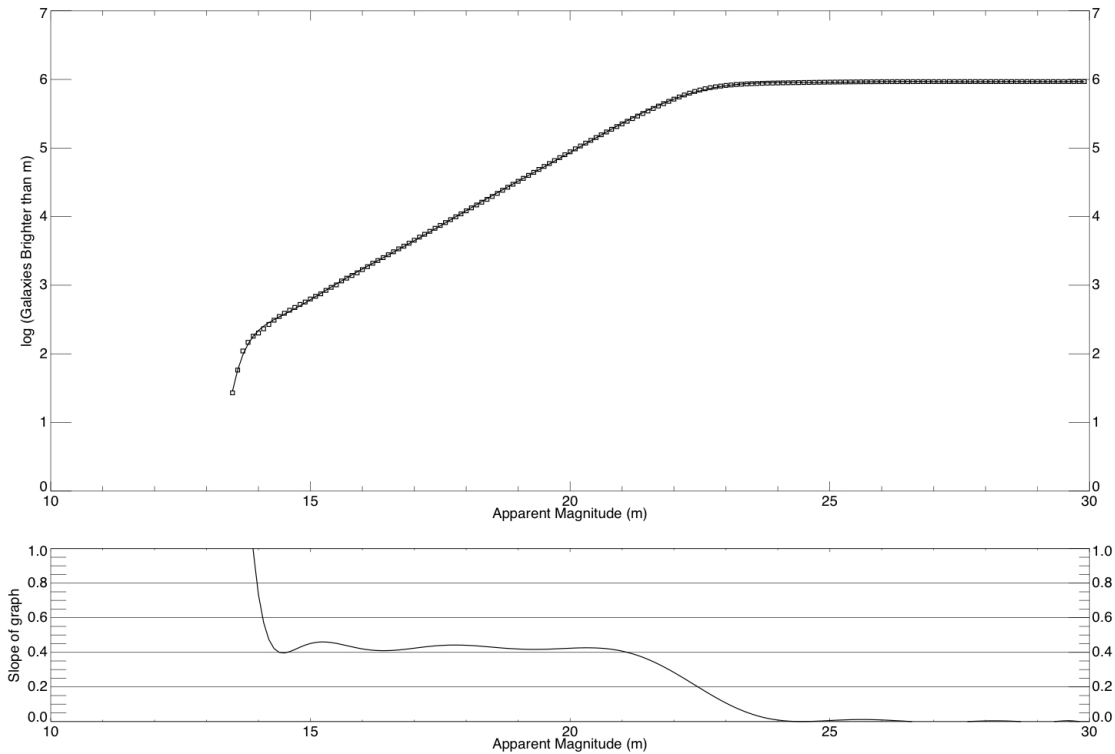
Figure 2.30 Wolf plot of SDSS data obtained through NED centered on coordinates RA: 217°, Dec: 43°.

be taken to ensure that the datasets are not contaminated by mis-identified bright stars.

One interesting thing that we did conclude from this research was that the pendulum has swung too far in the large-scale galaxy survey methodology. A few decades ago, the trend was to do very narrow and very deep "pencil-beam" surveys (e.g. Kirshner et al. 1983). These projects would utilize very long exposure times in order to detect galaxies that appear very faint due to the great distance. The trade-off is that they cover only a tiny fraction of the sky (in the case of Kirshner et al., only $40° \times 40°$). This makes them unusable for detecting large-scale structure in the universe unless that structure just happens to lie in the direction of the chosen line of sight.

Recently, however, surveys like the Harvard Centre for Astrophysics (CfA) survey (Huchra et al. 1988) and the SDSS attempted to cover much larger swaths of the sky, but with

a trade-off in the depth of coverage. This allows large filamentary galaxy superclusters, such as the CfA Great Wall and the Sloan Great Wall, to be detected, but only at the expense of sacrificing coverage of faint galaxies. The larger area coverage typically means shorter exposure times in each field of view. The SDSS, for example, is only able to reliably detect galaxies with an absolute magnitude brighter than $M_r = -15.90$ at a redshift of only $z = 0.10$. This completely excludes dwarf galaxies, which would be expected to form a large fraction of void galaxies.

What would be ideal for large-scale structure studies is a survey that took the middle road between the two extremes. It would be very useful to have a moderate region of the sky, perhaps on the order of $100°$ by $40°$, that had photometry extending as faint as say $m = 26$. This would allow fairly sizable structures to be observed out to a much larger distance.

As far as the usefulness of Wolf plots for pinpointing the center of a void without having redshift information for the individual galaxies, the complications induced by clustering on the front and/or back of the void makes it problematic. The slope of the Wolf plot can give you clues about the structure, however. If the slope drops below 0.6 and then crosses back above it, this would tend to indicate the presence of a scooped-out type void and a distance calculation may yield good results. On the other hand, if the slope first rises above 0.6 and then crosses below it, this indicates more a built-up type void and may require spectroscopic observations to pin down the location of the void in redshift space.

Some possibilities occur to us that could make the technique more useful. It would be interesting to use a subsample of galaxies that had a smaller distribution of absolute magnitudes (e.g. giant elliptical galaxies only). While difficult to get enough galaxies, this could be a useful probe and should be examined, if only because a data set with a uniform value of M would delineate voids perfectly.

Chapter 3

A Watershed Void Finder

In the previous chapter, we explored finding voids using only galaxy counts and magnitudes, which is a difficult task. With distance information, the job becomes more straightforward, but still has problems.

We next decided to use a completely different approach to locating voids, based upon a watershed method. Watershed transforms have been used for many years in image segmentation and edge finding algorithms (Beucher and Lantuejouil 1979; Beucher and Meyer 1993), although the first source in which I encountered it was in Kling (2000). Variations on this technique were applied by Platen et al. (2007) to find voids, but they first used a Delaunay Tessellation Field Estimator to convert the discrete data into a continuous density field. Since galaxies are not continuous, we felt that the "clumpiness" should be preserved in model data so as to allow more direct comparison to observational data. In addition, they do not allow voids to merge; they extend arbitrarily high "dams" above each ridgeline in order to keep each void separate and distinct.

A watershed technique for finding voids has several advantages and disadvantages. The primary advantage is that it requires very few input parameters. As discussed in Section 1.2, the choice of input parameters can alter the results significantly. With our watershed tech-

nique, the only parameter that needs to be specified in advance is the resolution of the grid. Once that is specified, the program produces results for all possible values of the overdensity that can be considered voids. However, to quote results, a second parameter (the overdensity threshold for what is considered a void) must be specified.

Another advantage of this technique over galaxy number counts is that it does not require you to choose a line-of-sight. As we saw in Section 2.2.1, sometimes specifying how large an area you want to consider can have a drastic effect on the identified void. The watershed method analyzes an entire volume of space, finding all voids in the region.

The third reason for using a watershed technique is for the insight it can provide into how voids "grow" as the overdensity threshold is raised. This allows us to look at how voids merge and can give us characteristic sizes of voids for different thresholds. It can also allow us to probe how porous are the walls separating the voids.

The first disadvantage is that the watershed technique must be supplied with the three-dimensional coordinates of each galaxy. In a simulation like the Millennium Run, this is not a problem since each galaxy is precisely located within the volume. However, applying it to observational data necessitates gathering spectra for every galaxy in the region to obtain redshift values. Any technique (other than GNC) will have the same requirement, however.

Another drawback of this particular technique is related to the choice of cell size. Small cell sizes, while better able to fill the void region, are also able to penetrate the walls, leading to a single large void that dominates the region. In addition, the void finder will identify any cell below the current overdensity threshold as a void. However, depending on the resolution you choose, this can lead to hundreds and thousands of isolated cells, only about 2-5 Mpc in radius, being identified as separate voids. This can generate a lot of spurious results depending on the size of the volume.

A related problem comes in how the finder treats the edge of the volume. Since the edge is treated identically to the interior, it is possible to get many small voids located along the

sides of the volume. If the data had extended further, perhaps they would have grown larger or merged, but this does complicate the calculation of typical void sizes.

3.1   The Watershed Void Explorer

Watershed techniques have their origin in geophysics. The idea derives its name from the concept of rainwater runoff. In a heavy downpour, the water will run downhill and collect in basins at the lowest point of local elevation. I specify local elevation because the rainwater would not necessarily be able to reach the level of the global minimum unless there were a sufficient volume of water to overflow the walls of the local basin. These walls define the edges of each segment.

In an astrophysical context, we use the local overdensity (see Equation 1.1) as a proxy for elevation. The lowest values of the overdensity (-1.00) correspond to regions of zero elevation, while high values would correspond to the peaks of mountains. Given that we add in an extra spatial dimension in our analysis, the concept of raining down the water will not work. Instead, we assume that water is flooding up from the "bottom" of each basin. We simply increment the overdensity threshold which indicates the elevations that are "under water."

The void finding program I wrote is called the **Wa**tershed **V**oid **E**xplorer (WaVE). This program performs a watershed analysis of a region of space and outputs information relating to voids and void galaxies. The program was written in the IDL programming language developed by ITT Visual Information Solutions. The code for WaVE is reproduced in Appendix A.

Before beginning program execution, the user must specify the resolution to use. We have used values ranging from 2 Mpc to 12 Mpc. The program takes the volume and divides it up into cells of the specified size. If the size of the region is not evenly divisible by the cell

size, the last few Mpc of the volume will not be used. In the case of the Millennium Run, the volume is cubical which makes the division simple; in the SDSS, the region is a wedge in right ascension, declination, and redshift, so it requires a different approach outlined in Section 3.3.

The overall density of the region, $\bar{\rho}$, is computed by taking the number of galaxies in the sample and dividing it by the volume. The density of each cell, ρ, is then computed in the same fashion. The overdensity for each cell is calculated according to Equation 1.1 and then the cells are sorted in order of increasing density.

WaVE begins with an overdensity of -1.000, which corresponds to the cells that have no galaxies in them. Cells with an overdensity less than the current threshold are flagged as void cells. They are then examined individually to check their status. We identify the immediate neighbors of the void cell as those that are ± 1 in x, y, and z. Each neighbor is checked to see if it is already a part of a void. There are four possible scenarios at this point:

1. If no neighbors are already assigned to a particular void, the cell is the beginning of a new void. The next void number in sequence is assigned to the cell.

2. If only a single neighbor has been previously assigned to a void, the new cell becomes part of the same void. The void size is incremented.

3. If multiple neighbors all belong to the same void, the new cell is added to the same void. The void size is increased by the size of the cell.

4. If multiple neighbors belong to different voids, the cell is arbitrarily assigned to the lowest numbered void it borders. However, because the addition of the new cell brings multiple voids into contact, a merger event also occurs. The largest void "consumes" the smaller voids.

We then increment the overdensity in steps of 0.001 until we find new cells which are

"flooded" and become part of a void. This causes additional void mergers as well as new voids to be formed. The process repeats until it reaches the threshold at -0.100, at which we stop considering the region a part of a void since that is only 10% below the average density of the volume.

At each overdensity, we output information about the voids. We calculate a "center of masslessness," which is akin to the center of mass for a void. In this calculation, the lack of mass (characterized as the absolute value of the overdensity) acts as a weighting factor such that the totally empty cells are weighted more heavily than cells that contain some galaxies. We also calculate an equivalent radius for each void, which is the radius of a spherical void with the same total volume. In addition, we calculate the average overdensity of each void and the total fraction of the volume contained in all the voids. The program also outputs the information about any galaxies located within each void. Although we calculate these statistics for each void, the focus in this work is on the overall nature of voids at each resolution and overdensity.

3.2 Application to the Millennium Run

The Millennium Run simulation volume is a cube 500 Mpc on a side. The nearly 9.5 million galaxies located in the volume were supplied with x, y, and z coordinates, along with absolute magnitudes in the SDSS *ugriz* filter set. The overall galaxy number density of the simulation, $\bar{\rho}$, is approximately 0.076 galaxies/Mpc^3.

While we ran the simulation with resolutions ranging from 2 Mpc to 12 Mpc, many of the results were similar. Therefore, we have decided to focus on the results obtained from four illustrative resolutions: 4, 5, 8, and 10 Mpc. It should be noted that not all of these cell sizes divide the volume evenly. Our first reaction was that this shouldn't make a significant difference; we would just start at coordinate (0,0,0) and lay out cells of the appropriate size

until another whole cell would not fit within the volume, thus trimming off the far side in each direction.

There was a question raised that relates to the problem of the volume not being evenly divisible by the cell size. The question was, "Does the size of the voids and the paths through the walls of the voids depend on the choice of the origin we picked to start dividing the region into cells?" The probability that this would have an effect seemed small, but we couldn't rule out this possibility entirely, so we had to test it.

To check whether or not these changes would have an effect on the results, we ran a test utilizing the 4 Mpc resolution. We first ran the void finder starting from (0,0,0) as a control to compare the other results to, but to standardize the volume, we trimmed off the last 4 Mpc in each direction. We then ran models with the origin of the cells at the following coordinates: (2,0,0), (4,0,0), (0,2,0), (2,2,0), (4,2,0), (0,4,0), (2,4,0), (4,4,0), (0,0,2), (2,0,2), (4,0,2), (0,2,2), (2,2,2), (4,2,2), (0,4,2), (2,4,2), (4,4,2), (0,0,4), (2,0,4), (4,0,4), (0,2,4), (2,2,4), (4,2,4), (0,4,4), (2,4,4), and (4,4,4), trimming off the excess volume from the edges.

To compare with our control run, we looked at two measures of comparison. The first was the volume filling fraction of all voids. This was calculated as the fraction of the volume (the portion located within the grid of cells) identified as part of any void. The second measure examined was the fraction of the void volume located within the largest void. After running all of the models, we calculated the mean for each statistic along with the standard deviation. The results are shown in Table 3.1.

As can be seen from these results, the differences between the mean values and the control run values are insignificant. This gives us a high level of confidence in excluding further consideration of these effects.

One thing that became obvious to us is that there may need to be another parameter specified for the minimum size of a void. The majority of voids were a single isolated cell,

Table 3.1 WaVE results when the origin was shifted. FF_v is the volume filling fraction and F_{vv} is the fraction of the void volume in the largest void.

Overdensity	Mean FF_v	Standard Deviation	Mean F_{vv}	Standard Deviation	Control FF_v	Control F_{vv}
-1.000	0.353	0.0001	0.858	0.0019	0.353	0.857
-0.793	0.502	0.0002	0.978	0.0005	0.502	0.978
-0.587	0.602	0.0002	0.993	0.0003	0.602	0.993
-0.381	0.675	0.0001	0.997	0.0002	0.675	0.997
-0.175	0.731	0.0001	0.999	0.0002	0.731	0.999

which would likely be considered too small to represent a true void. However, for the purposes of this research, since we are not looking at the characteristics of each void, no minimum size is applied.

We first go through the four different cell sizes and note the patterns that are observed. Then, in Section 3.2.5, we compare the results between the different cell sizes and analyze the importance of the choices of parameters.

3.2.1   4 Mpc Resolution

Our first run through the data was done with a resolution of 4 Mpc. This provides for 125 cells along each direction, for a total of just under 2 million cells. Within our definition of the threshold for voids, there are only five values of the overdensity that occur. These results are shown in Table 3.2.

There are some interesting things to draw from these results. The first thing to notice is what happens at an overdensity of -1.000. This represents cells that are completely empty. The volume filling fraction indicates that 35.3% of the volume is completely devoid

of galaxies. In addition, 85.7% of the void volume is located within a single void. The remaining 14.3% of the void volume is split among the remaining 36,307 voids, meaning that the vast majority of them are quite small, often only a single isolated cell. In fact, the next largest void only represents about 0.06% of the total void volume. A histogram showing the distribution of void sizes at this density threshold is shown in Figure 3.1a. It is clear from the histogram that there are more than 25,000 individual "voids" that are only single isolated cells. These voids would probably not be considered voids by most standards, but an arbitrary decision about the minimum proper size of a void is not the focus of this work.

At an overdensity of -0.793, nearly half of the entire volume is contained in the largest void. Looking at the histograms (Figure 3.1a – d) of the distribution of the void sizes, a trend can be identified. Excluding the largest void, the remainder of the void distribution shifts to smaller percentages of the void volume and the numbers are decreased as the largest void "consumes" the smaller voids. This pattern will be repeated at all resolutions, especially at higher density thresholds.

Table 3.2 The results from WaVE for a resolution of 4 Mpc. The overdensity is defined in Equation 1.1, the volume filling fraction is the fraction of the total volume classified as part of any void, the percentage of void volume is the portion of the total void volume that is part of the largest single void, the number of voids is self-explanatory, the number of void galaxies is the total of all galaxies located in the cells identified as part of any void, and the histogram figure is the subfigure within Figure 3.1 of the histograms of void sizes at that overdensity.

Overdensity	Vol. Fill. Frac.	Percent. Void Vol.	Num Voids	Num. Void Gals.	Fig. 3.1
-1.000	0.353	0.857	36,308	0	(a)
-0.793	0.502	0.978	13,425	291,295	(b)
-0.587	0.602	0.993	5,872	682,547	(c)
-0.381	0.675	0.997	2,697	1,111,490	—
-0.175	0.731	0.999	1,335	1,545,246	(d)
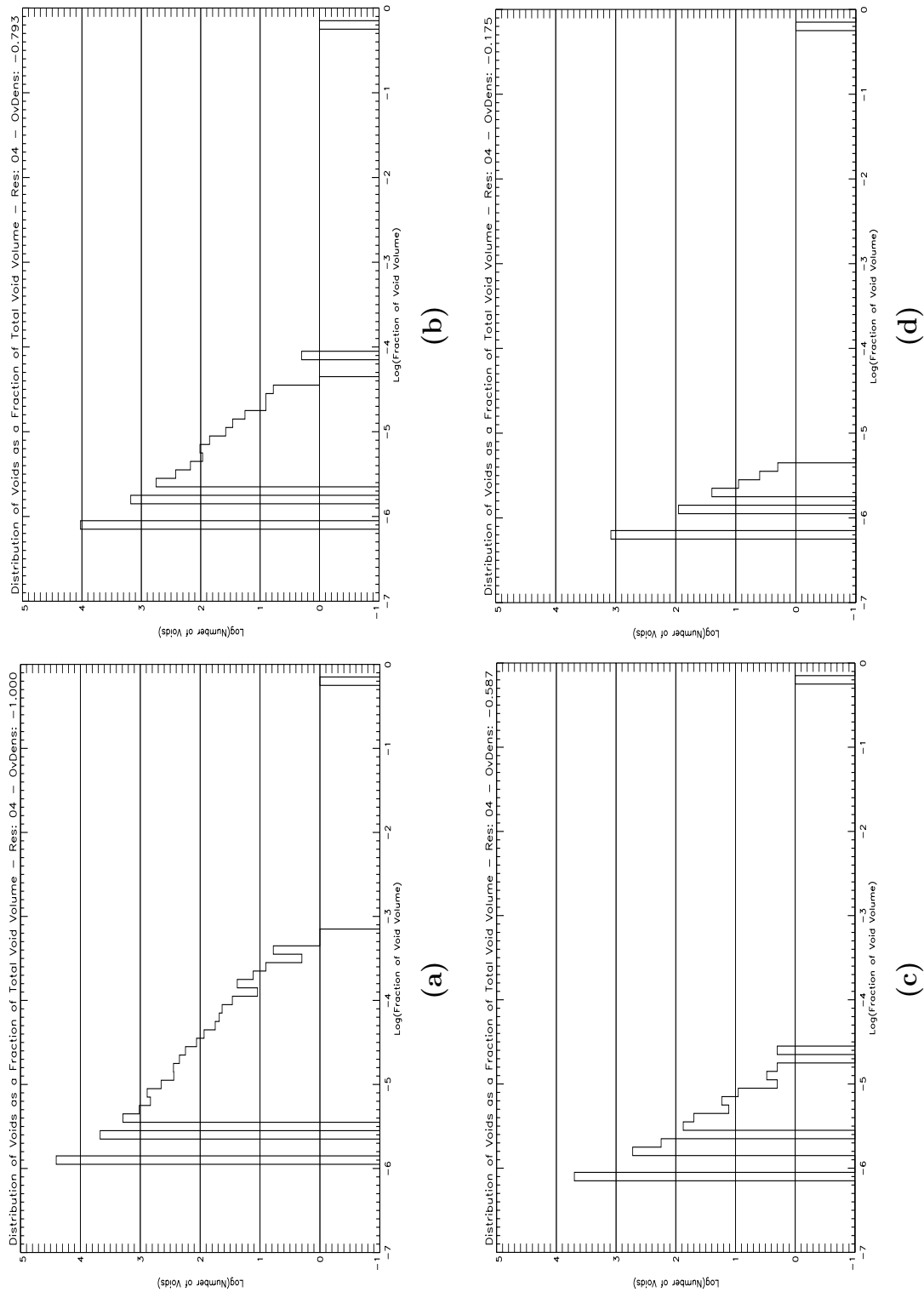
Figure 3.1 Histograms of the distribution of void sizes with 4 Mpc resolution: (a) is at an overdensity of −1.000; (b) is at an overdensity of −0.793; (c) is at an overdensity of −0.587; and (d) is at an overdensity of −0.175. Note the single large void dominates at all overdensities and that the distribution of void sizes narrows as the overdensities increase. Also notice that the majority of the "voids" in the distribution are actually single isolated cells at all overdensities.

3.2.2   5 Mpc Resolution

Dividing the region up into 5 Mpc cells provides 100 cells on each side of the volume. This gives a total of 1,000,000 cells. Table 3.3 summarizes the results of the run. Note that although there are nine density steps within our density threshold, we only include histograms for four of them as a representative sample.

Once again, there are a number of interesting things that can be seen in the results. The first thing that stands out is the enormous difference at an overdensity of -1.000 between the 4 Mpc and 5 Mpc resolutions. In the 4 Mpc case, the volume filling fraction was 0.353, while at 5 Mpc resolution it was only 0.199. Perhaps even more significant is the fraction of the void volume in the largest void: 0.857 versus 0.010 for 4 and 5 Mpc respectively. This enormous change leads to only one conclusion – the walls of the voids are permeable at the 4 Mpc resolution (allowing the vast majority of voids to merge), but not at the 5 Mpc cell size. However, if we are willing to accept at least one galaxy per cell (an overdensity of -0.894), the fraction in the largest void jumps all the way up to 0.742, indicating that the walls become very porous at that level.

Also of interest in Figure 3.2b, we notice that in addition to the largest void, which dominates the volume, there are a few other voids that have separated themselves from the pack in terms of size. However, by the histogram in Figure 3.2c, those larger voids have been consumed by the largest one. It is also interesting to note that in Figure 3.2d there are a number of void sizes that are not found within the distribution.

3.2.3   8 Mpc Resolution

The results for the 8 Mpc resolution are shown in Table 3.4. Because the 500 Mpc cube is not evenly divisible by an 8 Mpc cell size, we trim the farthest 4 Mpc off of each side. This produces 230,328 cells with a total volume of about 97.6% of the original. Once again, we
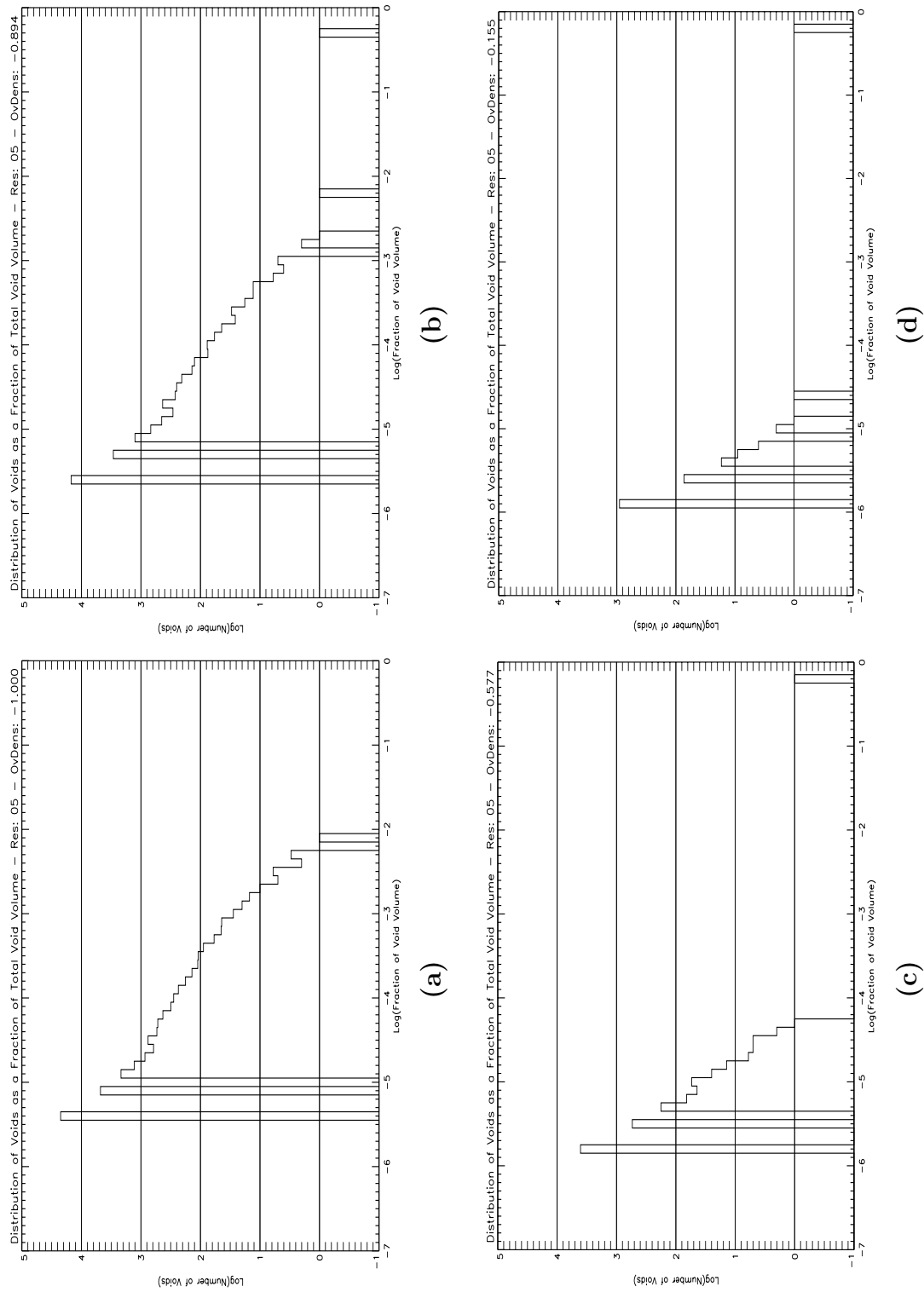
Figure 3.2 Histograms of the distribution of void sizes with 5 Mpc resolution: (a) is at an overdensity of −1.000; (b) is at an overdensity of −0.894; (c) is at an overdensity of −0.577; and (d) is at an overdensity of −0.155. The largest void doesn't come to dominate the void volume until an overdensity of −0.894. There are a number of voids that are standing out from the rest in (b), but they have been consumed in (c).

Table 3.3 The results from WaVE for a resolution of 5 Mpc. The overdensity is defined in Equation 1.1, the volume filling fraction is the fraction of the total volume classified as part of any void, the percentage of void volume is the portion of the total void volume that is part of the largest single void, the number of voids is self-explanatory, the number of void galaxies is the total of all galaxies located in the cells identified as part of any void, and the histogram figure is the subfigure within Figure 3.2 of the histograms of void sizes at that overdensity and resolution.

Overdensity	Vol. Fill. Frac.	Percent. Void Vol.	Num Voids	Num. Void Gals.	Fig. 3.2
−1.000	0.199	0.010	35,992	0	(a)
−0.894	0.314	0.742	22,233	115,337	(b)
−0.788	0.404	0.929	13,293	294,139	—
−0.683	0.476	0.971	8,128	512,557	—
−0.577	0.537	0.987	4,985	756,969	(c)
−0.472	0.589	0.993	3,136	1,014,654	—
−0.366	0.634	0.996	2,084	1,281,828	—
−0.261	0.671	0.997	1,438	1,545,735	—
−0.155	0.704	0.998	1,009	1,809,039	(d)

choose to only show a sample of four histograms.

It is clear from the data that there are very few cells that are totally empty at this resolution (the volume filling fraction at an overdensity of -1.000 is only 0.021). Also, as can be seen in Figure 3.3a, there really are no dominant voids at this level; the largest void is only 0.3% of the total void volume. In fact, at this density threshold, the spread in void sizes is actually quite small.

However, by the time we reach an overdensity of -0.767 (Figure 3.3b), the spread is significantly wider, and we see the beginnings of three significant voids, although they are not substantially larger than the rest of the pack. Notice also that at this density level, the volume filling fraction is still lower than it was for the 5 Mpc resolution at an overdensity of -0.894.

By an overdensity of -0.716, one void has clearly established itself as dominant, occupying 53.6% of the void volume (roughly 15% of the total volume, and representing a 670% increase in void volume occupied). The next largest void only contributes about 1% of the void volume. This can be seen in Figure 3.3c.

The number of voids also peaks at an overdensity of -0.871 at 9,125. By the last density step (Figure 3.3d), there are only 407 voids remaining and about 350 of those are single isolated cells. The largest void occupies nearly 64% of the total volume.

Table 3.4 The results from WaVE for a resolution of 8 Mpc. The overdensity is defined in Equation 1.1, the volume filling fraction is the fraction of the total volume classified as part of any void, the percentage of void volume is the portion of the total void volume that is part of the largest single void, the number of voids is self-explanatory, the number of void galaxies is the total of all galaxies located in the cells identified as part of any void, and the histogram figure is the subfigure within Figure 3.3 of the histograms of void sizes at that overdensity and resolution.

Overdensity	Vol. Fill. Frac.	Percent. Void Vol.	Num Voids	Num. Void Gals.	Fig. 3.3
−1.000	0.021	0.004	3,655	0	(a)
−0.974	0.043	0.006	5,948	5,426	—
−0.948	0.066	0.005	7,558	16,850	—
−0.922	0.090	0.005	8,491	33,917	—
−0.896	0.115	0.006	9,004	58,457	—
−0.871	0.139	0.008	9,125	87,872	—
−0.845	0.163	0.011	8,952	123,440	—
−0.819	0.188	0.015	8,612	164,838	—
−0.793	0.211	0.033	8,017	211,422	—
−0.767	0.234	0.044	7,409	260,301	(b)
−0.742	0.256	0.080	6,680	315,911	—
−0.716	0.279	0.536	6,012	377,511	(c)
−0.690	0.301	0.703	5,385	440,031	—
−0.664	0.322	0.793	4,785	507,228	—
−0.638	0.343	0.856	4,195	578,628	—
−0.613	0.363	0.892	3,702	651,153	—
−0.587	0.383	0.924	3,244	729,473	—
−0.561	0.401	0.941	2,849	807,197	—
−0.535	0.420	0.954	2,525	887,513	—

Continued on next page...

Table 3.4 *continued*

Overdensity	Vol. Fill. Frac.	Percent. Void Vol.	Num Voids	Num. Void Gals.	Fig. 3.3
−0.510	0.438	0.963	2,218	971,094	—
−0.484	0.455	0.970	1,948	1,055,714	—
−0.458	0.472	0.975	1,745	1,141,604	—
−0.432	0.488	0.980	1,548	1,227,492	—
−0.406	0.503	0.983	1,372	1,314,501	—
−0.381	0.518	0.986	1,181	1,402,197	—
−0.355	0.532	0.988	1,065	1,489,222	—
−0.329	0.547	0.990	934	1,580,092	—
−0.303	0.560	0.992	844	1,668,301	—
−0.277	0.573	0.993	758	1,756,669	—
−0.252	0.585	0.993	696	1,843,031	—
−0.226	0.597	0.995	629	1,928,891	—
−0.200	0.609	0.995	568	2,017,396	—
−0.174	0.620	0.996	505	2,107,124	—
−0.149	0.631	0.996	465	2,194,013	—
−0.123	0.641	0.997	407	2,276,905	(d)

3.2.4   10 Mpc Resolution

Table 3.5 shows the results for the run with a 10 Mpc cell size. The volume is divided up into 125,000 cells, with 50 on each side of the region. Notice that because of the larger cell volume, the size of the density steps has decreased dramatically, leading to a much larger data set. However, we still choose to only show a representative sample of the histograms generated.

As was seen in Section 3.2.3, there are very few empty voids in the volume at this resolution, with the majority of them being isolated cells (Figure 3.4a). In fact, the number of voids peaks at 4,872 at an overdensity of -0.815.

The largest void does not come to dominate the volume until a density threshold of -0.643. From -0.656 to -0.643, the fraction of the void volume in the largest void jumped from 14.4% up to 46.0% (corresponding to 3.9% and 12.9% of the total volume respectively). After the largest void emerges, it continues to consume the smaller voids, causing the number of voids to decline and the spread of void sizes to decrease.
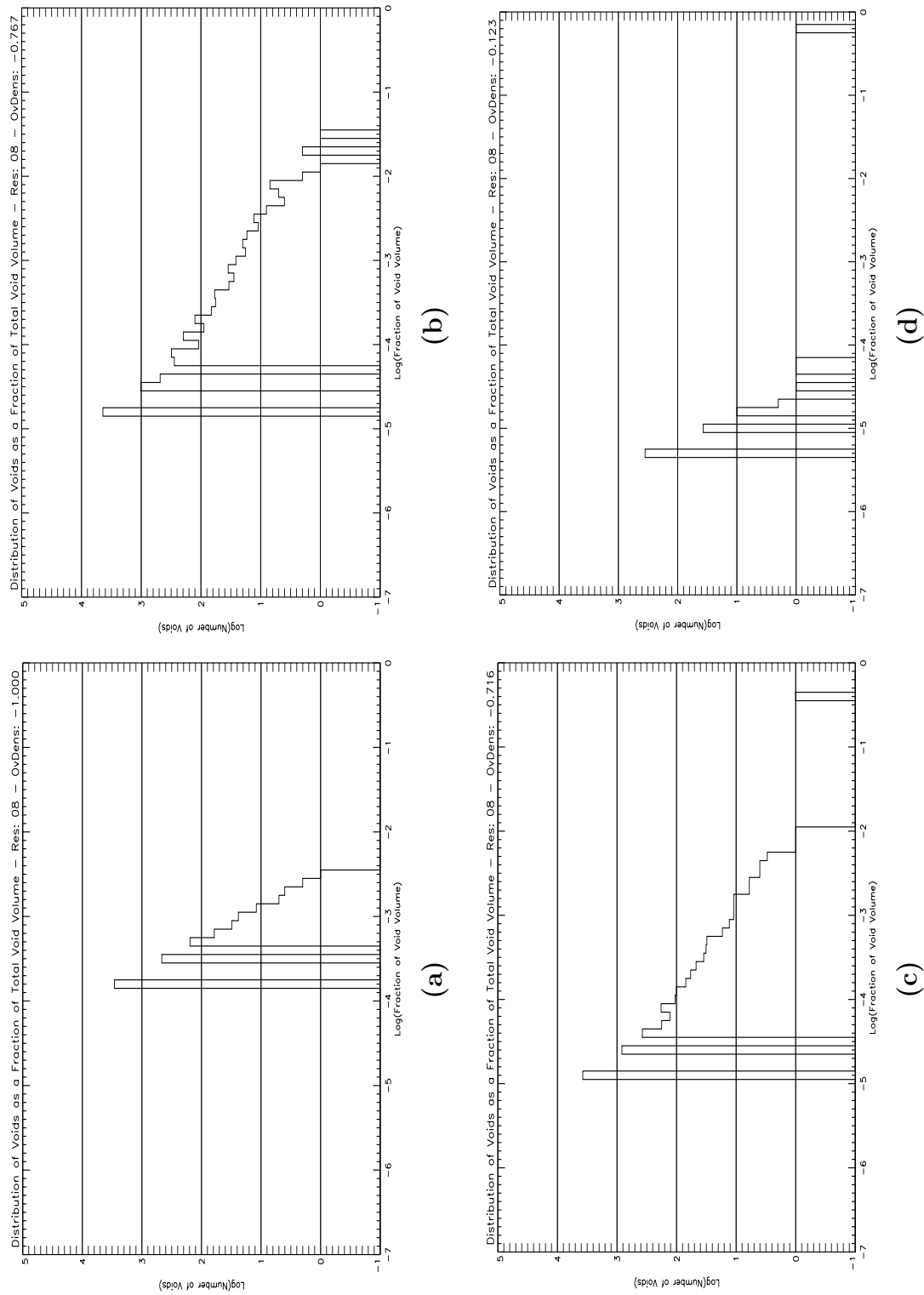
Figure 3.3 Histograms of the distribution of void sizes with 8 Mpc resolution: (a) is at an overdensity of −1.000; (b) is at an overdensity of −0.767; (c) is at an overdensity of −0.716; and (d) is at an overdensity of −0.123. The distribution is the widest in (b)–(c). There are three voids that separate from the pack at an overdensity of −0.767, but only one dominates by −0.716.

Table 3.5 The results from WaVE for a resolution of 10 Mpc. The overdensity is defined in Equation 1.1, the volume filling fraction is the fraction of the total volume classified as part of any void, the percentage of void volume is the portion of the total void volume that is part of the largest single void, the number of voids is self-explanatory, the number of void galaxies is the total of all galaxies located in the cells identified as part of any void, and the histogram figure is the subfigure within Figure 3.4 of the histograms of void sizes at that overdensity and resolution.

Overdensity	Vol. Fill. Frac.	Percent. Void Vol.	Num Voids	Num. Void Gals.	Fig. 3.4
−1.000	0.003	0.010	368	0	(a)
−0.986	0.008	0.007	818	586	—
−0.973	0.014	0.008	1,305	2,034	—
−0.960	0.020	0.006	1,797	4,491	—
−0.947	0.028	0.005	2,267	8,119	—
−0.934	0.036	0.005	2,707	13,234	—
−0.920	0.044	0.006	3,119	19,618	—
−0.907	0.054	0.006	3,491	27,885	—
−0.894	0.063	0.006	3,815	37,557	—
−0.881	0.074	0.006	4,094	49,347	—
−0.868	0.085	0.007	4,390	62,787	—
−0.854	0.096	0.007	4,552	77,868	—
−0.841	0.107	0.007	4,677	94,368	—
−0.828	0.118	0.007	4,769	112,555	—
−0.815	0.129	0.007	4,872	132,561	—
−0.802	0.141	0.010	4,840	153,891	(b)
−0.788	0.152	0.013	4,844	176,259	—
−0.775	0.164	0.015	4,799	201,538	—
−0.762	0.175	0.017	4,710	227,638	—

Continued on next page...

Table 3.5 *continued*

Overdensity	Vol. Fill. Frac.	Percent. Void Vol.	Num Voids	Num. Void Gals.	Fig. 3.4
−0.749	0.188	0.018	4,598	256,803	—
−0.736	0.199	0.019	4,489	285,103	—
−0.722	0.211	0.018	4,378	316,183	—
−0.709	0.222	0.023	4,199	348,105	—
−0.696	0.234	0.026	4,054	381,478	—
−0.683	0.246	0.058	3,860	417,934	—
−0.670	0.258	0.132	3,661	453,809	—
−0.656	0.269	0.144	3,479	491,639	—
−0.643	0.280	0.460	3,295	529,169	(c)
−0.630	0.292	0.579	3,107	569,573	—
−0.617	0.303	0.642	2,908	610,376	—
−0.604	0.315	0.734	2,756	655,136	—
−0.590	0.326	0.765	2,618	696,180	—
−0.577	0.337	0.802	2,495	741,492	—
−0.564	0.348	0.834	2,342	785,943	—
−0.551	0.358	0.857	2,237	828,477	—
−0.538	0.368	0.876	2,101	873,907	—
−0.525	0.379	0.895	1,978	920,779	—
−0.511	0.389	0.903	1,867	967,695	—
−0.498	0.399	0.912	1,767	1,014,739	—
−0.485	0.409	0.921	1,664	1,063,372	—
−0.472	0.419	0.931	1,565	1,114,252	—

Continued on next page...

Table 3.5 *continued*

Overdensity	Vol. Fill. Frac.	Percent. Void Vol.	Num Voids	Num. Void Gals.	Fig. 3.4
−0.459	0.428	0.938	1,473	1,163,739	—
−0.445	0.438	0.949	1,370	1,212,123	—
−0.432	0.447	0.956	1,270	1,264,110	—
−0.419	0.457	0.964	1,150	1,315,722	—
−0.406	0.466	0.969	1,056	1,367,382	—
−0.393	0.475	0.972	995	1,420,006	—
−0.379	0.484	0.975	918	1,473,586	—
−0.366	0.493	0.977	858	1,525,282	—
−0.353	0.502	0.980	807	1,578,937	—
−0.340	0.510	0.982	756	1,631,437	—
−0.327	0.518	0.983	721	1,683,406	—
−0.313	0.526	0.985	682	1,735,458	—
−0.300	0.534	0.987	625	1,790,101	—
−0.287	0.542	0.987	601	1,842,967	—
−0.274	0.550	0.989	562	1,896,592	—
−0.261	0.557	0.990	525	1,946,040	—
−0.247	0.565	0.990	492	2,000,760	—
−0.234	0.572	0.991	458	2,052,612	—
−0.221	0.579	0.992	422	2,104,532	—
−0.208	0.586	0.993	399	2,156,192	—
−0.195	0.592	0.993	381	2,206,639	—
−0.181	0.599	0.994	364	2,255,805	—

Continued on next page...

Table 3.5 *continued*

Overdensity	Vol. Fill. Frac.	Percent. Void Vol.	Num Voids	Num. Void Gals.	Fig. 3.4
-0.168	0.606	0.994	354	2,311,875	—
-0.155	0.613	0.994	328	2,365,955	—
-0.142	0.619	0.995	304	2,417,500	—
-0.129	0.625	0.995	292	2,470,696	—
-0.116	0.632	0.996	280	2,522,822	—
-0.102	0.638	0.996	265	2,576,746	(d)

3.2.5 Comparisons

This section presents comparative graphs for various statistics from the different runs. By plotting the same statistic from each resolution, interesting trends can be seen. We first consider how the volume filling fraction varies with density, as shown in Figure 3.5. The behavior is exactly what we expected – the smaller the volume of the cells, the larger the percentage of the volume that is identified as part of a void for any given overdensity. This makes sense since the smaller cells are able to fit within the crevices of the higher-density structures. At the upper end of the density range, the various models appear to be converging towards a volume filling fraction of $0.700 - 0.7500$.

We next turn our attention the fraction of the void volume contained in the largest void at each density step, which is shown in Figure 3.6. This graph provides insight into the permeability of the walls separating the voids. Notice that at an overdensity of -1.000, for all resolutions except 4 Mpc, the largest void in each of the samples had $\sim 1\%$ of the total void volume. However, the 4 Mpc resolution cells are able to penetrate the walls with impunity, leading the largest void to consume more than 85% of the total void volume. It is also interesting to note that for each resolution, there is a given threshold at which the walls become porous; this is characterized by a large jump in the fraction of the void volume in the largest void. For 5 Mpc, this occurs at an overdensity of -0.894, representing a single galaxy within a cell. The 8 Mpc cells reach the ability to penetrate the walls much later, at a density of -0.716 (11 galaxies in a cell). At 10 Mpc, the cells are so large that you must allow up to 27 galaxies to be in a cell before you can cross through most walls.

The graph of the number of voids at each density level (Figure 3.7) shows a stark contrast between the smaller resolutions and the larger ones. At 4 and 5 Mpc, the number of voids is very similar. They both begin with approximately 35,000 voids at an overdensity of -1.000, and drop to about 500 at the upper density threshold. By contrast, the 8 and 10 Mpc
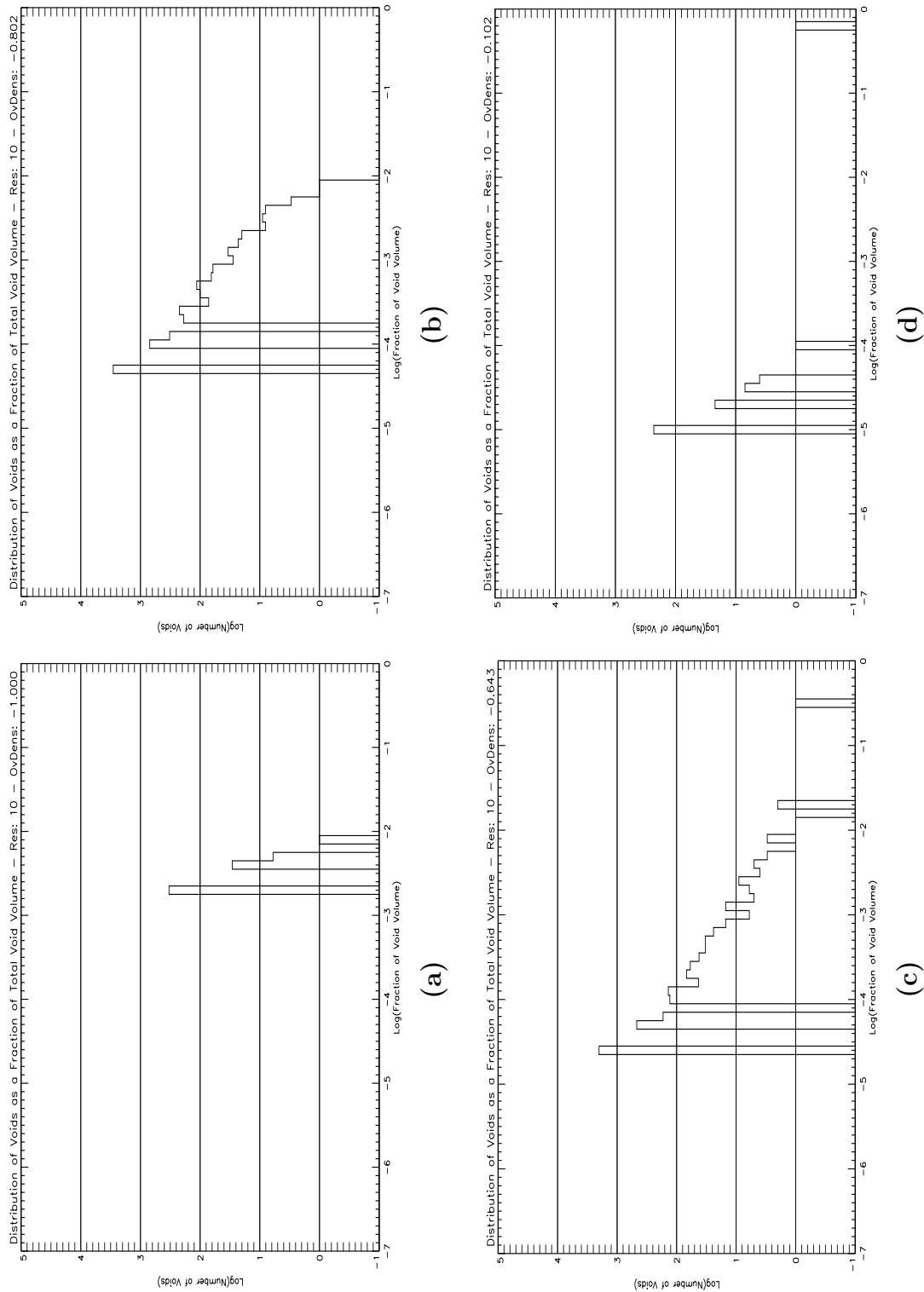
Figure 3.4 Histograms of the distribution of void sizes with 10 Mpc resolution: (a) is at an overdensity of -1.000; (b) is at an overdensity of -0.802; (c) is at an overdensity of -0.643; and (d) is at an overdensity of -0.102. Note the sparseness of the distribution in (a) and (d), and that there is no dominant void until an overdensity of -0.643.
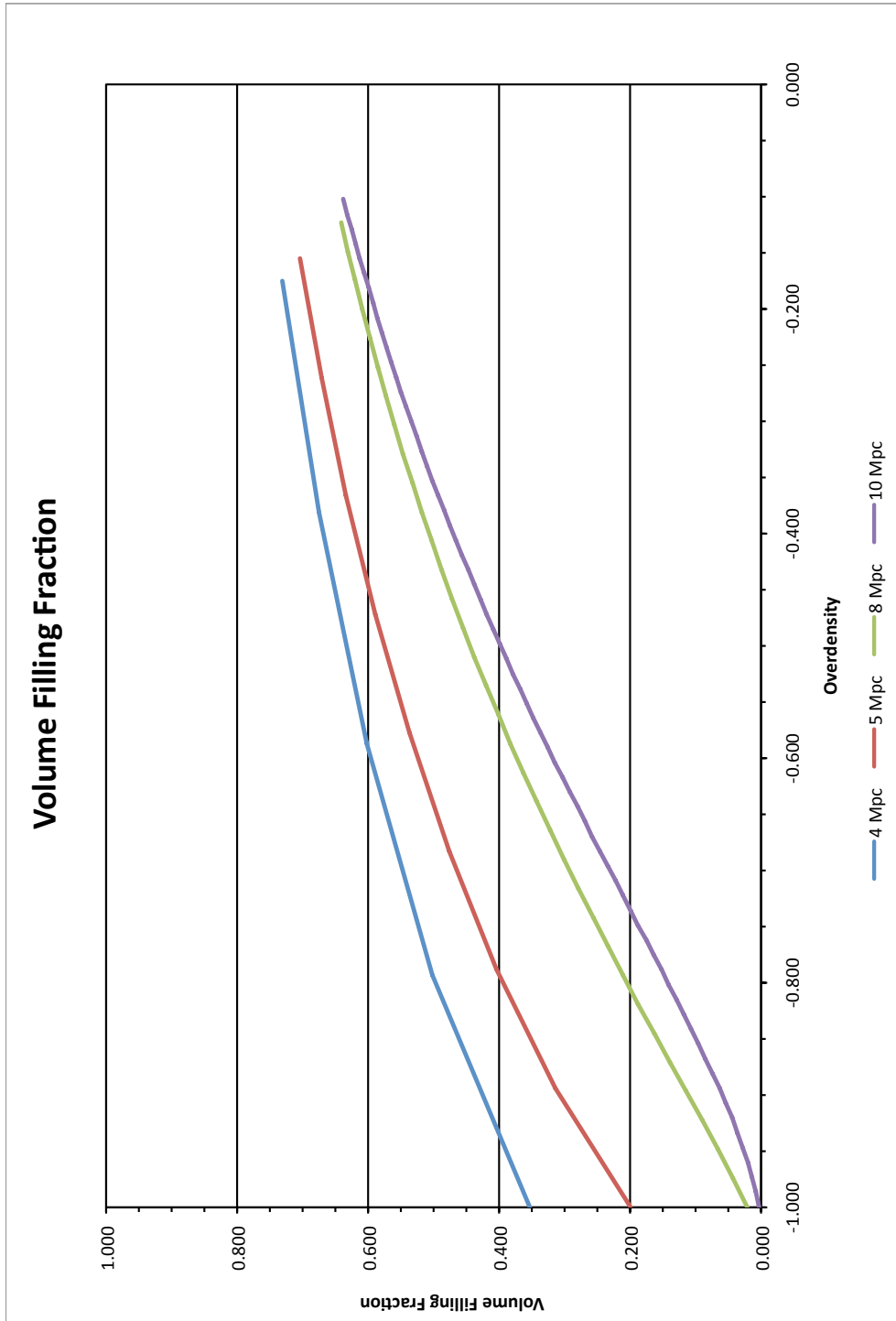
Figure 3.5 Comparison of the volume filling fraction between resolutions.
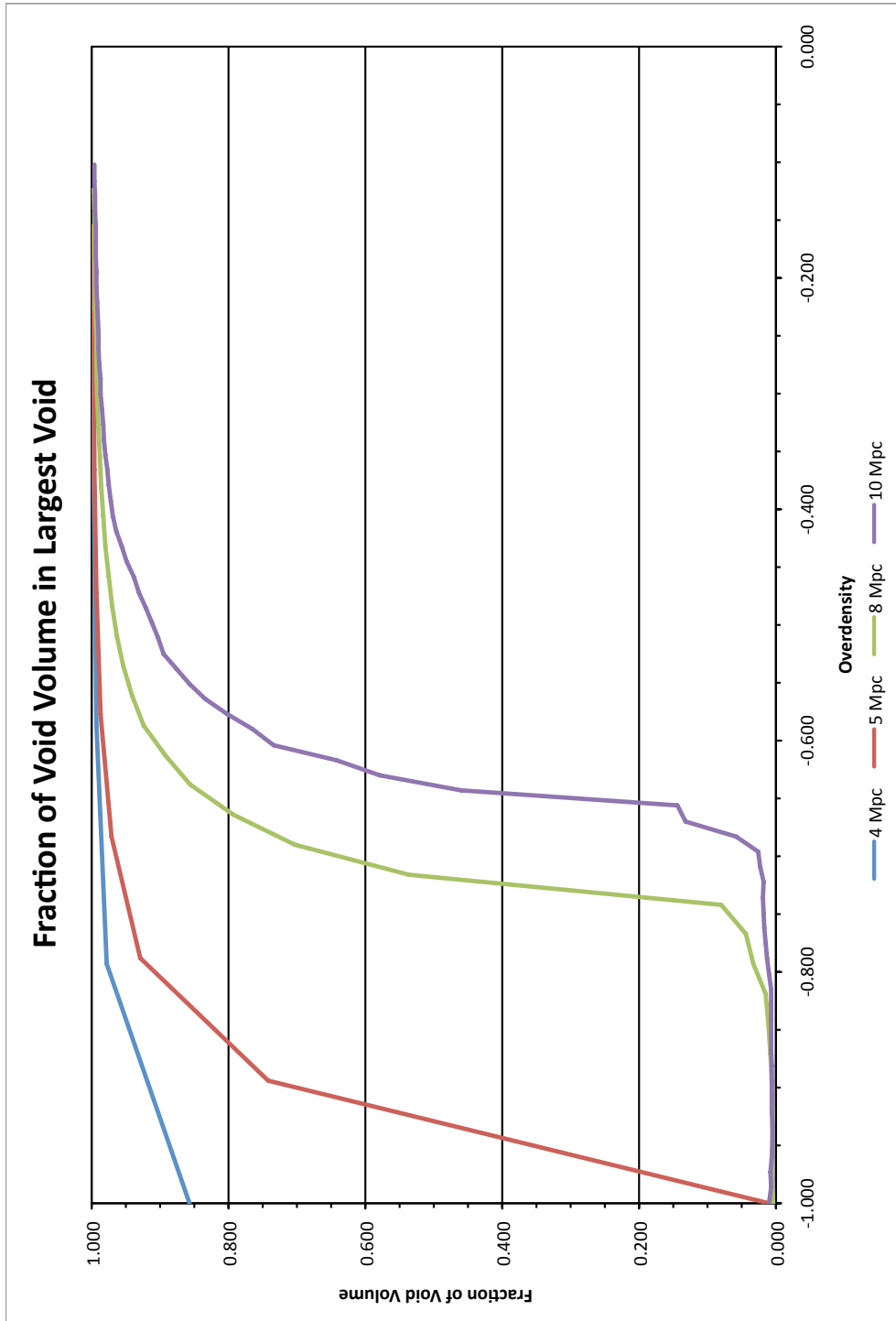
Figure 3.6 Comparison of the fraction of the void volume in the largest void between resolutions.

resolutions do not reach their peak number of voids until later, peaking at around 9,000 voids at a density of -0.871 for the 8 Mpc cells and at about 5,000 voids at -0.815 for 10 Mpc. The number of voids then tapers off in a similar fashion for both large-cell models.

Figure 3.8 was, at first glance, a bit surprising, but on further reflection made perfect sense. The high density end was exactly what we expected – the larger cells were allowed to contain more galaxies at a given density threshold, so they contained the most void galaxies. However, at an overdensity of about -0.600, all of the lines converge and cross over, completely reversing the order. On further reflection, it became clear that this is due to the larger numbers of voids produced by the smaller cell sizes. It just takes the larger resolutions a bit longer to catch up to the smaller ones.

As a further comparison, we ran our WaVE technique on the same region of the Millennium Run as used in the Aspen-Amsterdam void finder comparison. Due to the convergence of the number of void galaxies shown in Figure 3.8, we will set our overdensity threshold at -0.600 and quote results for the closest density to that threshold without exceeding it. The results from the different runs are shown in Table 3.6.

Table 3.6 Application of the WaVE technique to the same region as the Aspen-Amsterdam void finder comparison project (Colberg et al. 2008). The resolution is the cell size, the overdensity is defined in Equation 1.1, the number of voids is the number that have a center that is located more than 10 Mpc from the edge of the volume, the volume filling fraction is the fraction of the total volume classified as part of any void, the number of void galaxies is the total of all galaxies located in the cells identified as part of a void, and the average overdensity within the voids.

Resolution (Mpc)	Overdensity	N_V	FF_v	N_g	δ_g
4	-0.40	1	0.69	1175	-0.85
5	-0.54	1	0.60	944	-0.86
6	-0.56	1	0.53	947	-0.84

These results are significantly different from those found in the Aspen-Amsterdam project.
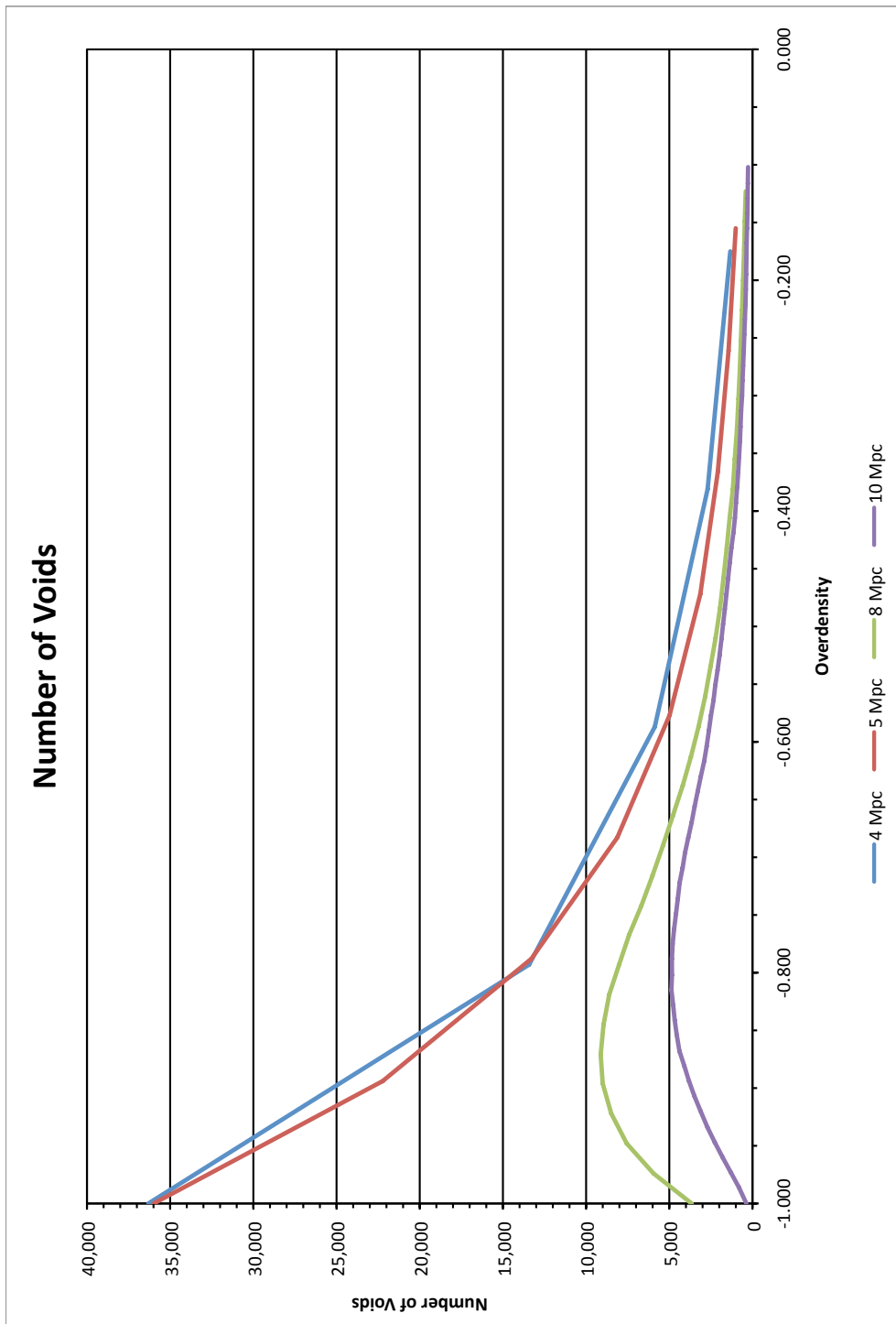
Figure 3.7 Comparison of the number of voids between resolutions.
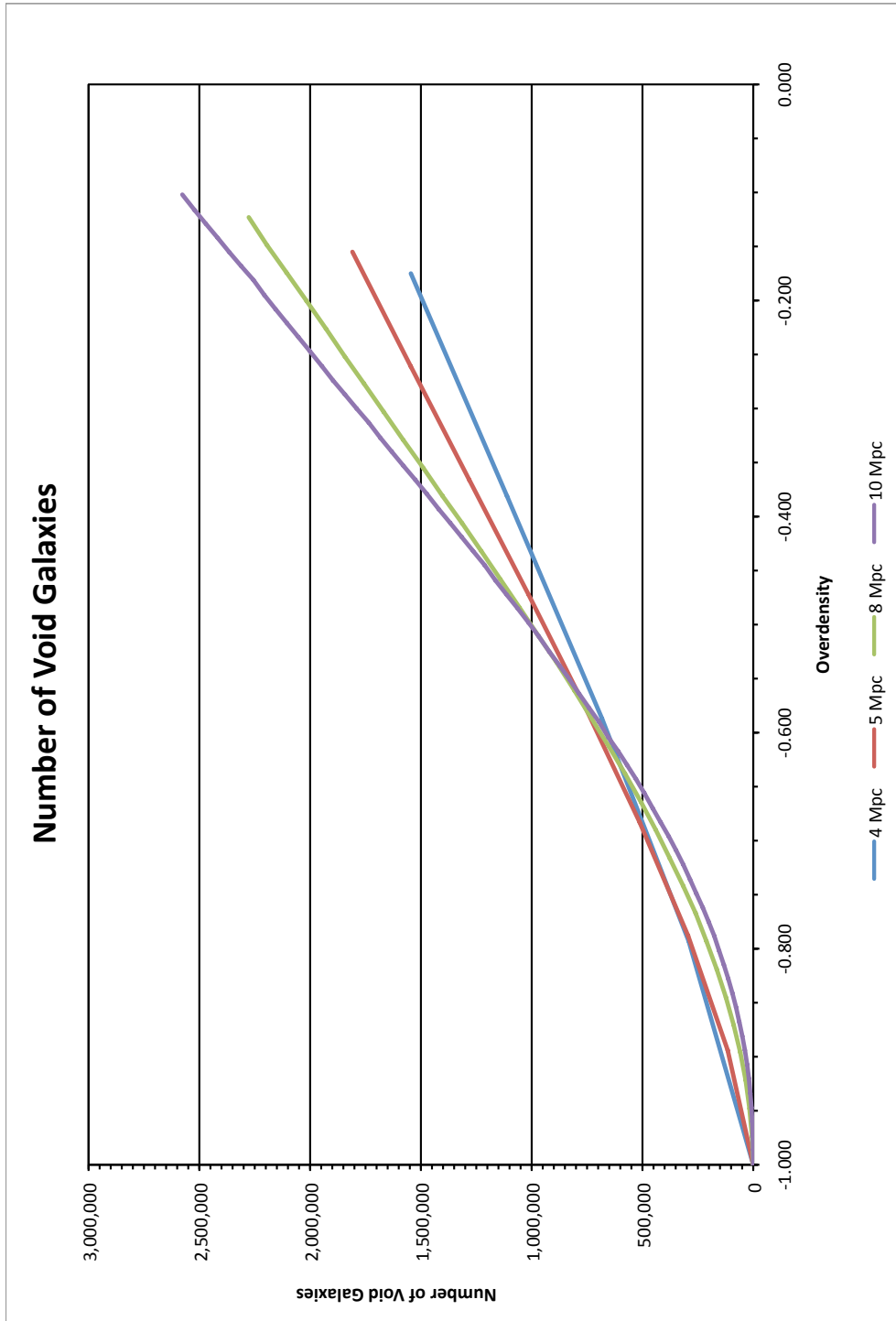
Figure 3.8 Comparison of the number of void galaxies between resolutions.

Because of the small volume used in the comparison, we felt constrained to utilize smaller cell sizes. However, as has already been pointed out, small cells allow the voids to penetrate the walls and connect. Many, if not all, of the models compared in the project apply some form of edge condition which, once established, preserves the identity of the void, preventing it from merging with other voids. Because we do not apply similar criteria, all of the voids in the region have become connected into a single large void "network."

The discrepancies in the number of void galaxies and the average overdensity have a straightforward explanation. When we apply the same boundaries specified by the authors to the Millennium Run, we found 500,000 fewer galaxies in the region than Colberg et al. found. When I discussed this situation with Darren Croton, the designer of the galaxy catalog, he informed me that the catalog that is publicly available has a faint magnitude limit of $M_r = -16$, while the Aspen-Amsterdam project used a data set that went as faint as $M_r = -14$. Croton emphasized to me that the galaxy model is unreliable at these faint levels, as many galaxies that should be populated in the region are missed due to the resolution of the dark matter haloes. This would tend to provide an increased number of faint void galaxies since the low-mass haloes would most often be located within voids.

While it would be nice to be able to say with certainty that we had identified every void galaxy in the region, we are hampered in that effort by the lack of a clear definition of a void, and hence what constitutes a void galaxy. Several void finders required the voids to be spherical (e.g. those of Brunino and Gottlöber), while others (such as Foster and Hoyle/Vogeley) start with spheres and then merge them with smaller spheres down to a specified minimum size. Other models utilize much more complex criteria to define a void. This leads to the widely different appearance of the largest void identified by each void finder (see Figure 1.1). With these different definitions for a void, the authors obtained very divergent answers for the number of void galaxies.

The average overdensity of the voids is a little more tricky to compare, again due to

the definition of a void used by each author. Some models are more rigorous in excluding galaxies than others. If you take the mean value of the average void overdensities found by the authors that used the galaxy distribution to find voids, you get -0.71. This is larger than the values we found, which is again attributable to the presence of fainter galaxies in their data set. Also, the average overdensity in voids would change is we selected a different value for the overdensity threshold in our data.

3.3 Application to the Sloan Digital Sky Survey

There are two main challenges that we faced when attempting to apply WaVE to the SDSS data. The first, as was alluded to earlier, is that the technique requires a three-dimensional data set, which means limiting the sample to those objects which had spectroscopic data available. To illustrate just how limiting that is, it is instructive to consider how far away an M^* galaxy could be and still appear bright enough to make it within the survey completeness limits. We calculate the maximum distance ($d_{max,ph}$) using the photometric limit of $m_{lim,ph} = 22.2$ and calculate the maximum distance($d_{max,sp}$) using the spectroscopic limit of $m_{lim,sp} = 17.77$. From Equation 2.17, we find:

$$d_{max,ph} = 10^{0.2(m_{lim,ph}-M^*)+1} = 10^{0.2(22.2-(-20.73))+1} = 10^{9.586} = 3,855 \text{Mpc} \qquad (3.1)$$

$$d_{max,sp} = 10^{0.2(m_{lim,sp}-M^*)+1} = 10^{0.2(17.77-(-20.73))+1} = 10^{8.700} = 501 \text{Mpc} \qquad (3.2)$$

This is a significant reduction, allowing the galaxy in the spectroscopic sample to be seen at only 13% of the distance of the photometric limit. We will return to this problem later in the section after we describe our methodology.

The second problem that we encounter has to do with the geometry of the region. Because we take a range in right ascension and a range in declination, we end up with a three-dimensional wedge-shaped volume. However, we still want to use cubical cells. Therefore,

we identify a block large enough to totally encompass the selected wedge.

We then divide that block into cells of the designated size. Because the density of a cell would be affected by the exclusion of even a single galaxy, we require that the only cells that are considered in our model be completely contained within the galaxy sample limits. We check this by converting the coordinates of all eight corners of each cell from rectangular coordinates to polar coordinates and check to ensure they are all within the range of the sample.

The survey region that we chose was $120.0 \leq RA \leq 240.0$, $10.0 \leq Dec \leq 50.0$, and $0.0132 \leq z \leq 0.0500$. The block that we designed to encase the volume was divided into 26,232,850 cells 10 Mpc on a side. Of those cells, only about 16% (4,155,120) were completely within the boundaries of the region. The SDSS spectroscopic data only found 46,981 galaxies within the parameters specified. This is only one galaxy per 100 cells!

When we ran the WaVE program on the volume, it identified 4,142,788 cells (99.7%) as completely empty. This was divided among 1,497 separate voids. Obviously, there were no void galaxies detected since even a single galaxy in a cell would give it an overdensity of nearly 100. While disappointing, this result further highlights the need for a more complete, large-scale nearby survey.

3.4   Conclusions

In spite of the limitations discussed at the beginning of this chapter, the watershed method of locating voids shows a lot of promise. On a data set like the Millennium Run, it is able to find voids and merge connected voids into a single larger void. However, because it requires three dimensional coordinates for all objects, it can only be used with spectroscopic data sets. Because spectroscopy requires a much higher threshold of incoming photons than photometry, the usable data sets will necessarily be much smaller than photometric surveys

of equal sky coverage and will only include the brighter galaxies. For example, in the SDSS, the r-band completeness limit for photometry is $m_{r,ph} = 22.20$ while for spectroscopy it is only $m_{r,sp} = 17.77$. To further illustrate these effects, we queried the SDSS database in a $40° \times 40°$ patch of sky to obtain all objects flagged as galaxies. The spectroscopic survey returned 136,445 galaxies, while the photometric dataset returned 49,750,649 "galaxies" in the same region (subject to the misidentified bright stars as discussed in Section 2.3).

When using a much more sparse data set, like the SDSS spectroscopic data, the WaVE program quickly identifies a large fraction of the total volume as being a void. This not due to a real lack of galaxies as much as it is due to the inability of the survey instrument to obtain spectra of the much more abundant faint galaxies. The back edge of the survey will always be less populated than the front edge because as distance increases, the luminosity a galaxy must have in order to still be visible in the survey also increases. In practical terms, this means that at each given distance, the survey is really sampling a different population of galaxies.

However, on model data or on a volume limited observational data set, it can generate a lot of information about individual voids and their merger history at different density thresholds. It can calculate a "center of masslessness" for each void (equivalent to a center of mass but with higher weight given to emptier regions), determine how oblong the void might be, and an equivalent radius for a spherical void of the same volume. It can also produce a list of which galaxies are located within voids, which is useful for comparing the properties of void galaxies to those of non-void galaxies.

Overall, we find that our technique is very good at characterizing the overall volume contained in voids. It is also good at identifying void galaxies and will be put to use in this regard in future work. However, it would not necessarily be useful if someone wanted to produce a catalog of distinct voids. Because we do not artificially constrain voids to be of a certain size or geometry, we often end up with voids that have a number of filamentary

tendrils connecting to other voids. The voids thus identified would not appear to the eye of an observer to be necessarily connected.

Chapter 4

Conclusion

Galaxy number count analysis, through the use of Wolf plots, shows some promise at being able to map out voids from a two- or three-dimensional data set at close distances. If we had access to a more focused survey with a fainter magnitude limit, this could be a very good probe of structure. However, it has limitations that must be kept in mind. First, unless the void is huge (> 100 Mpc in diameter), the technique is only really effective out to a distance of about 200 Mpc. Second, it is a little like shining a flashlight in a dark room – it only illuminates the area where it is pointed, making it impractical to try to identify every void in a large survey. Third, it is sensitive to the clustering of galaxies along the line of sight. And fourth, we have not considered the case where there may be multiple voids along a single line of sight, which could cause the front-edge underdensity of the second void to erase the back-edge overdensity of the first void.

The watershed technique also turned out to be much more successful on model data than on observational data. This, too, could be improved with a more appropriate galaxy survey. However, it allows us to calculate a lot of very useful information about individual voids and to gather aggregate statistics about all voids in the region. One of the most promising things WaVE does is to identify void galaxies. In Section 1.1, we raised the questions about

how strongly the results of the void finders depend on the choices of parameters and is there a compelling reason to choose a particular set of parameters. We can fairly convincingly state that the choice of initial parameters can have a dramatic difference in the results. For example, if you choose a 10 Mpc cell size for WaVE, you would determine that about 0.3% of the volume was completely empty, while choosing 4 Mpc would lead to the conclusion that 35.3% of the region was devoid of galaxies.

There are several avenues for future work using these tools. The primary need is for a new galaxy survey as outlined above. This work may provide the motivation for a telescope time allocation committee to provide a sizable block of time to carry it out. We have produced histograms of the size distributions of voids in the Millennium Run. Once a suitable survey was conducted, it would be instructive to generate similar graphs for the observed data to get a better sense of how representative of reality the simulation is. We would also like to look at the properties of void galaxies in the Millennium Run and compare the results with previously published data on the properties of void galaxies. This would allow us to determine another aspect of how well the simulation models reality.

vanced Study, the Japan Participation Group, Johns Hopkins University, the Joint Institute for Nuclear Astrophysics, the Kavli Institute for Particle Astrophysics and Cosmology, the Korean Scientist Group, the Chinese Academy of Sciences (LAMOST), Los Alamos National Laboratory, the Max-Planck-Institute for Astronomy (MPIA), the Max-Planck-Institute for Astrophysics (MPA), New Mexico State University, Ohio State University, University of Pittsburgh, University of Portsmouth, Princeton University, the United States Naval Observatory, and the University of Washington.

This research has made use of the NASA/IPAC Extragalactic Database (NED) which is operated by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

The Millennium Run simulation used in this paper was carried out by the Virgo Supercomputing Consortium at the Computing Centre of the Max-Planck Society in Garching. The semi-analytic galaxy catalogue is publicly available at http://www.mpa-garching.mpg.de/galform/agnpaper.

Appendix A

WaVE Code

```
FUNCTION BSORT, ARRAY, ASORT, INFO=INFO, REVERSE = REV
;+
; NAME:
;        BSORT
; PURPOSE:
;        Function to sort data into ascending order, like a simple bubble sort.
; EXPLANATION:
;        Original subscript order is maintained when values are equal (FIFO).
;        (This differs from the IDL SORT routine alone, which may rearrange
;        order for equal values)
;
; CALLING SEQUENCE:
;        result = bsort( array, [ asort, /INFO, /REVERSE ] )
;
; INPUT:
;        Array - array to be sorted
;
; OUTPUT:
;        result - sort subscripts are returned as function value
;
; OPTIONAL OUTPUT:
;        Asort - sorted array
;
; OPTIONAL KEYWORD INPUTS:
;        /REVERSE - if this keyword is set, and non-zero, then data is sorted
```

```
;                     in descending order instead of ascending order.
;        /INFO = optional keyword to cause brief message about # equal values.
;
; HISTORY
;        written by F. Varosi Oct.90:
;        uses WHERE to find equal clumps, instead of looping with IF ( EQ ).
;        compatible with string arrays, test for degenerate array
;        20-MAY-1991     JKF/ACC via T AKE- return indexes if the array to
;                        be sorted has all equal values.
;        Aug - 91  Added  REVERSE keyword   W. Landsman
;        Always return type LONG    W. Landsman        August 1994
;        Converted to IDL V5.0   W. Landsman   September 1997
;-
        N = N_elements( Array )
        if N lt 1 then begin
                print , 'Input to BSORT must be an array '
                return , [0L]
           endif


        if N lt 2 then begin
            asort = array          ;MDM added 24-Sep-91
            return ,[0L]      ;Only 1 element
        end
;
; sort array (in descending order if REVERSE keyword specified )
;
        subs = sort( Array )
        if keyword_set( REV ) then subs = rotate(subs,5)
        Asort = Array[subs]
;
; now sort subscripts into ascending order
; when more than one Asort has same value
;
            weq = where( (shift( Asort, -1 ) eq Asort) , Neq )

        if keyword_set( info ) then $
                message, strtrim( Neq, 2 ) + " equal values Located",/CON,/INF

        if (Neq EQ n) then return ,lindgen(n) ;Array is degenerate equal values
```

```
        if  (Neq GT 0) then begin

                if  (Neq GT 1) then begin                    ;find clumps of equality

                        wclump = where( (shift( weq, −1 ) − weq) GT 1, Nclump )
                        Nclump = Nclump + 1

                   endif else Nclump = 1

                if  (Nclump LE 1) then begin
                        Clump_Beg = 0
                        Clump_End = Neq−1
                   endif else begin
                        Clump_Beg = [0, wclump+1]
                        Clump_End = [wclump, Neq−1]
                   endelse

                weq_Beg = weq[ Clump_Beg ]                   ;subscript ranges
                weq_End = weq[ Clump_End ] + 1               ; of Asort equalities.

                if  keyword_set( info ) then message, strtrim( Nclump, 2 ) + $
                                " clumps of equal values Located",/CON,/INF

                for  ic = 0L, Nclump−1 do begin              ;sort each clump.

                        subic = subs[ weq_Beg[ic] : weq_End[ic] ]
                        subs[ weq_Beg[ic] ] = subic[ sort( subic ) ]
                   endfor

                if  N_params() GE 2 then Asort = Array[subs]        ;resort array.
        endif

return, subs
END


PRO WAVE_PSON, FILENAME=FILENAME, PAPER=PAPER, MARGIN=MARGIN, PAGE_SIZE=PAGE_SIZE, INCHES=
        INCHES, ASPECT=ASPECT, LANDSCAPE=LANDSCAPE, QUIET=QUIET
```

```
;-- Check arguments.
if (n_elements(filename) eq 0) then filename='idl.ps'
if (n_elements(paper) eq 0) then paper='LETTER'
if (n_elements(margin) eq 0) then begin
    margin = 2.54 / 2
endif else begin
    if keyword_set(inches) then margin = margin * 2.54
endelse


;-- Check if Printer mode is active.
if !d.name eq 'PS' then begin
    message, 'PS output is already active', /continue
    return
endif


;-- Get ratio of character width/height to screen width/height.
xratio = float(!d.x_ch_size) / float(!d.x_vsize)
yratio = float(!d.y_ch_size) / float(!d.y_vsize)


;-- Save current device information in common block.
common pson_information, inform

inform = {device:!d.name, window:!d.window, font:!p.font, filename:filename, xratio:xratio,
    yratio:yratio}


;-- Get size of page (centimeters).
widths = [[8.5, 8.5, 11.0, 7.25]*2.54, 21.0, 29.7]
heights = [[11.0, 14.0, 17.0, 10.50]*2.54, 29.7, 42.0]
names = ['LETTER', 'LEGAL', 'TABLOID', 'EXECUTIVE', 'A4', 'A3']
index = where(strupcase(paper) eq names, count)
if (count ne 1) then begin
    message, 'PAPER selection not supported', /continue
    return
endif
page_width = widths[index[0]]
page_height = heights[index[0]]


;-- If page size was supplied, use it.
```

```
if (n_elements(page_size) eq 2) then begin
    page_width = page_size[0]
    page_height = page_size[1]
    if keyword_set(inches) then begin
        page_width = page_width * 2.54
        page_height = page_height * 2.54
    endif
endif


;- Compute aspect ratio of page when margins are subtracted.
page_aspect = float(page_height - 2.0 * margin) / float(page_width - 2.0 * margin)


;- Get aspect ratio of current graphics window
if (!d.window ge 0) then begin
    win_aspect = float(!d.y_vsize) / float(!d.x_vsize)
endif else begin
    win_aspect = 512.0 / 640.0
endelse


;- If aspect ratio was supplied, use it.
if (n_elements(aspect) eq 1) then win_aspect = float(aspect)


;- Compute size of drawable area.
case keyword_set(landscape) of
    0 : begin
        if (win_aspect ge page_aspect) then begin
            ysize = page_height - 2.0 * margin
            xsize = ysize / win_aspect
        endif else begin
            xsize = page_width - 2.0 * margin
            ysize = xsize * win_aspect
        endelse
    end


    1 : begin
        if (win_aspect ge (1.0 / page_aspect)) then begin
            ysize = page_width - 2.0 * margin
            xsize = ysize / win_aspect
        endif else begin
```

```
                xsize = page_height - 2.0 * margin
                ysize = xsize * win_aspect
          endelse
     end
endcase


;- Compute offset of drawable area from page edges.
if (keyword_set(landscape) eq 0) then begin
     xoffset = (page_width - xsize) * 0.5
     yoffset = (page_height - ysize) * 0.5
endif else begin
     xoffset = (page_width - ysize) * 0.5
     yoffset = (page_height - xsize) * 0.5 + xsize
endelse


;- Switch to PS device.
set_plot , 'PS'
device , landscape=keyword_set(landscape), scale_factor=1.0
device , xsize=xsize, ysize=ysize, xoffset=xoffset, yoffset=yoffset
device , filename=filename, bits_per_pixel=8


;- Set character size.
xcharsize = round(inform.xratio * !d.x_vsize)
ycharsize = round(inform.yratio * !d.y_vsize)
device , set_character_size=[xcharsize, ycharsize]


;- Report to user.
if (keyword_set(quiet) eq 0) then print, filename, format='("Started PS output to ", a)'


END


PRO WAVE_PSOFF, QUIET=QUIET

;- Check that PS output is active.
if (!d.name ne 'PS') then begin
     message , 'PS output not active: nothing done', /continue
     return
endif
```

```
;- Get entry device information from common block.
common pson_information , inform
if (n_elements(inform) eq 0) then begin
    message , 'WAVE_PSON was not called prior to WAVE_PSOFF:   nothing done', /continue
    return
endif


;- Close PS device.
device , /close_file


;- Switch to graphics entry device.
set_plot , inform.device


;- Restore window and font.
if (inform.window ge 0) then wset , inform.window
!p.font = inform.font


;- Report to user.
if (keyword_set(quiet) eq 0) then print , 'Finished PS output'


END



FUNCTION CELL_SORT, DENS=DENS


print , 'Sorting by density ... '
num = ulong(n_elements(dens))
sort_index = ulonarr(num)
sort_index = bsort(dens)
print , 'Finished sorting!'
return , sort_index


END



PRO WaVE, MAX_DENS=MAX_DENS


if (n_elements(max_dens) eq 0) then max_dens = 0.90
```

```
;- Load cell info.
print, 'Reading cells...'

cellfilename = dialog_pickfile(filter='MR_cells.bin', path='/media/usbdisk/MRData/')
header = {cell_size:0, num_cells_dim:0UL, total_cells:0UL}
openr, 1, cellfilename

readu, 1, header

cells      = { Cell_ID         : ulonarr(header.total_cells),   $
               Cell_X          : ulonarr(header.total_cells),   $
               Cell_Y          : ulonarr(header.total_cells),   $
               Cell_Z          : ulonarr(header.total_cells),   $
               Galaxies        : ulonarr(header.total_cells),   $
               Void_ID         : ulonarr(header.total_cells),   $
               Flow_Into       : ulonarr(header.total_cells),   $
               Mass            : dblarr(header.total_cells),    $
               Dens_Ratio      : dblarr(header.total_cells),    $
               Mass_Dens_Ratio : dblarr(header.total_cells)     $
             }

voids      = { Void_ID         : ulonarr(header.total_cells), $
               Origin          : ulonarr(header.total_cells), $
               Num_Cells       : ulonarr(header.total_cells)  $
             }

mergers    = { Void_1_ID       : ulonarr(header.total_cells), $
               Void_1_Size     : ulonarr(header.total_cells), $
               Void_2_ID       : ulonarr(header.total_cells), $
               Void_2_Size     : ulonarr(header.total_cells), $
               Merge_Dens      : fltarr(header.total_cells)   $
             }

nsort_cells = { Cell_ID        : ulonarr(header.total_cells), $
                Dens_Ratio     : dblarr(header.total_cells)   $
              }

readu, 1, cells
```

```
print, 'Done!'

;- Prep statistics file.
statsfilenm = 'MR_void_stats.txt'
statsfilename = filepath(statsfilenm, root_dir='/media/usbdisk/MRData/VoidData'+string(
    header.cell_size, format='(I2.2)')+'/')
openw, slun, statsfilename, /get_lun
close, 1

;- Set up counters, arrays, and variables.

num_voids = 1UL          ;- Void ID number.
num_mergers = 1UL        ;- Merger number.
curr_dens = 0.0          ;- Current density.
ddens = 0.001            ;- Amount to increment the density each step.
ctr = 0UL               ;- Counter to step through cells in each batch.

        sort_index = ulonarr(header.total_cells)

        ;- Sort the cells by number density.
        sort_index = cell_sort(dens=cells.Dens_Ratio)
        nsort_cells.Cell_ID = cells.Cell_ID[sort_index]
        nsort_cells.Dens_Ratio = cells.Dens_Ratio[sort_index]

        ;- Start with num_dens of 0 (actually less than 1E-06).
        batch_index = where(nsort_cells.Dens_Ratio lt 1E-06)
        print, 'Density = ', float(0), format='(A, F5.3)'

        if (batch_index[0] ne -1) then begin            ;- Test to make sure there
            are cells of density 0.
          num_cells_step = ulong(n_elements(batch_index))
        endif else begin
          num_cells_step = 0
        endelse

        print, 'Cells in this density threshold: ', num_cells_step, format='(A, I7)'
        for ctr=0UL, num_cells_step-1UL do begin        ;- Go through the empty
            cells.
```

```
if ((ctr mod 1000) eq 0) then print, 'Cell # ', ctr, format='(A, I7)'
temp_id = nsort_cells.Cell_ID[ctr]                    ;- The ID of the empty cell.
num_neighbors = 6
if ((cells.Cell_X[temp_id] eq 0) or (cells.Cell_X[temp_id] eq header.num_cells_dim
    -1UL)) then num_neighbors = num_neighbors - 1
if ((cells.Cell_Y[temp_id] eq 0) or (cells.Cell_Y[temp_id] eq header.num_cells_dim
    -1UL)) then num_neighbors = num_neighbors - 1
if ((cells.Cell_Z[temp_id] eq 0) or (cells.Cell_Z[temp_id] eq header.num_cells_dim
    -1UL)) then num_neighbors = num_neighbors - 1


neighbors = { Cell_ID         : ulonarr(num_neighbors+1),    $
              Void_ID         : ulonarr(num_neighbors+1),    $
              Flow_Into       : ulonarr(num_neighbors+1),    $
              Dens_Ratio      : dblarr(num_neighbors+1),     $
              Mass_Dens_Ratio : dblarr(num_neighbors+1)      $
            }

neighbors.Cell_ID[0] = cells.Cell_ID[temp_id]
neighbors.Void_ID[0] = cells.Void_ID[temp_id]
neighbors.Flow_Into[0] = cells.Flow_Into[temp_id]
neighbors.Dens_Ratio[0] = cells.Dens_Ratio[temp_id]
neighbors.Mass_Dens_Ratio[0] = cells.Mass_Dens_Ratio[temp_id]

i = 1
if (cells.Cell_X[temp_id] ne 0) then begin
  neighbors.Cell_ID[i] = cells.Cell_ID[temp_id-1UL]
  neighbors.Void_ID[i] = cells.Void_ID[temp_id-1UL]
  neighbors.Flow_Into[i] = cells.Flow_Into[temp_id-1UL]
  neighbors.Dens_Ratio[i] = cells.Dens_Ratio[temp_id-1UL]
  neighbors.Mass_Dens_Ratio[i] = cells.Mass_Dens_Ratio[temp_id-1UL]
  i = i + 1
endif
if (cells.Cell_X[temp_id] ne header.num_cells_dim-1UL) then begin
  neighbors.Cell_ID[i] = cells.Cell_ID[temp_id+1UL]
  neighbors.Void_ID[i] = cells.Void_ID[temp_id+1UL]
  neighbors.Flow_Into[i] = cells.Flow_Into[temp_id+1UL]
  neighbors.Dens_Ratio[i] = cells.Dens_Ratio[temp_id+1UL]
  neighbors.Mass_Dens_Ratio[i] = cells.Mass_Dens_Ratio[temp_id+1UL]
  i = i + 1
```

```
    endif
    if ( cells . Cell_Y [ temp_id ] ne 0) then begin
      neighbors . Cell_ID [ i ] = cells . Cell_ID [ temp_id − header . num_cells_dim ]
      neighbors . Void_ID [ i ] = cells . Void_ID [ temp_id − header . num_cells_dim ]
      neighbors . Flow_Into [ i ] = cells . Flow_Into [ temp_id − header . num_cells_dim ]
      neighbors . Dens_Ratio [ i ] = cells . Dens_Ratio [ temp_id − header . num_cells_dim ]
      neighbors . Mass_Dens_Ratio [ i ] = cells . Mass_Dens_Ratio [ temp_id − header .
          num_cells_dim ]
      i = i + 1
    endif
    if ( cells . Cell_Y [ temp_id ] ne header . num_cells_dim −1UL) then begin
      neighbors . Cell_ID [ i ] = cells . Cell_ID [ temp_id + header . num_cells_dim ]
      neighbors . Void_ID [ i ] = cells . Void_ID [ temp_id + header . num_cells_dim ]
      neighbors . Flow_Into [ i ] = cells . Flow_Into [ temp_id + header . num_cells_dim ]
      neighbors . Dens_Ratio [ i ] = cells . Dens_Ratio [ temp_id + header . num_cells_dim ]
      neighbors . Mass_Dens_Ratio [ i ] = cells . Mass_Dens_Ratio [ temp_id + header .
          num_cells_dim ]
      i = i + 1
    endif
    if ( cells . Cell_Z [ temp_id ] ne 0) then begin
      neighbors . Cell_ID [ i ] = cells . Cell_ID [ temp_id − header . num_cells_dim ^2]
      neighbors . Void_ID [ i ] = cells . Void_ID [ temp_id − header . num_cells_dim ^2]
      neighbors . Flow_Into [ i ] = cells . Flow_Into [ temp_id − header . num_cells_dim ^2]
      neighbors . Dens_Ratio [ i ] = cells . Dens_Ratio [ temp_id − header . num_cells_dim ^2]
      neighbors . Mass_Dens_Ratio [ i ] = cells . Mass_Dens_Ratio [ temp_id − header .
          num_cells_dim ^2]
      i = i + 1
    endif
    if ( cells . Cell_Z [ temp_id ] ne header . num_cells_dim −1UL) then begin
      neighbors . Cell_ID [ i ] = cells . Cell_ID [ temp_id + header . num_cells_dim ^2]
      neighbors . Void_ID [ i ] = cells . Void_ID [ temp_id + header . num_cells_dim ^2]
      neighbors . Flow_Into [ i ] = cells . Flow_Into [ temp_id + header . num_cells_dim ^2]
      neighbors . Dens_Ratio [ i ] = cells . Dens_Ratio [ temp_id + header . num_cells_dim ^2]
      neighbors . Mass_Dens_Ratio [ i ] = cells . Mass_Dens_Ratio [ temp_id + header .
          num_cells_dim ^2]
      i = i + 1
    endif

    cells . Flow_Into [ temp_id ] = cells . Cell_ID [ temp_id ]
```

```
if (min(neighbors.Void_ID) eq 999999999UL) then begin

  ;- If there are no neighbors already assigned to a void, then create a new void.
  cells.Void_ID[temp_id] = num_voids
  voids.Void_ID[num_voids-1UL] = num_voids
  voids.Origin[num_voids-1UL] = cells.Cell_ID[temp_id]
  voids.Num_Cells[num_voids-1UL] = 1UL
  num_voids = num_voids + 1UL

endif else begin

  ;- If a neighbor is already assigned to a void, determine which void to join.
  void_neighbors = where(neighbors.Void_ID ne 999999999UL)
  num_void_neigh = n_elements(void_neighbors)

  temp_void_neigh = { Cell_ID         : ulonarr(num_void_neigh),   $
                      Void_ID         : ulonarr(num_void_neigh),   $
                      Flow_Into       : ulonarr(num_void_neigh),   $
                      Dens_Ratio      : dblarr(num_void_neigh),    $
                      Mass_Dens_Ratio : dblarr(num_void_neigh)     $
                    }

  temp_void_neigh.Cell_ID = neighbors.Cell_ID[void_neighbors]
  temp_void_neigh.Void_ID = neighbors.Void_ID[void_neighbors]
  temp_void_neigh.Flow_Into = neighbors.Flow_Into[void_neighbors]
  temp_void_neigh.Dens_Ratio = neighbors.Dens_Ratio[void_neighbors]
  temp_void_neigh.Mass_Dens_Ratio = neighbors.Mass_Dens_Ratio[void_neighbors]

  sort_void_idx = uniq(temp_void_neigh.Void_ID, sort(temp_void_neigh.Void_ID))
  num_uniq_voids = n_elements(sort_void_idx)

  cells.Void_ID[temp_id] = temp_void_neigh.Void_ID[sort_void_idx[0]]
  voids.Num_Cells[cells.Void_ID[temp_id]-1UL] = voids.Num_Cells[cells.Void_ID[
      temp_id]-1UL] + 1UL

  if (num_uniq_voids ne 1) then begin

    void_one = temp_void_neigh.Void_ID[sort_void_idx[0]]
```

```
for j=1, num_uniq_voids −1 do begin

  void_two = temp_void_neigh . Void_ID [ sort_void_idx [ j ]]
  ;− Merge the smaller void into the larger one, defaulting to the lower # in
      a tie.

  if ( voids . Num_Cells [ void_one −1UL] lt voids . Num_Cells [ void_two −1UL]) then
    begin

    mergers . Void_1_ID [ num_mergers −1UL] = void_two
    mergers . Void_1_Size [ num_mergers −1UL] = voids . Num_Cells [ void_two −1UL]
    mergers . Void_2_ID [ num_mergers −1UL] = void_one
    mergers . Void_2_Size [ num_mergers −1UL] = voids . Num_Cells [ void_one −1UL]
    mergers . Merge_Dens [ num_mergers −1UL] = 0.0
    num_mergers = num_mergers + 1UL
    merge_index = ulong ( where ( cells . Void_ID eq void_one ))
    cells . Void_ID [ merge_index ] = void_two
    voids . Num_Cells [ void_two −1UL] = voids . Num_Cells [ void_two −1UL] + voids .
        Num_Cells [ void_one −1UL]
    voids . Num_Cells [ void_one −1UL] = 0UL
    void_one = void_two

  endif else begin

    mergers . Void_1_ID [ num_mergers −1UL] = void_one
    mergers . Void_1_Size [ num_mergers −1UL] = voids . Num_Cells [ void_one −1UL]
    mergers . Void_2_ID [ num_mergers −1UL] = void_two
    mergers . Void_2_Size [ num_mergers −1UL] = voids . Num_Cells [ void_two −1UL]
    mergers . Merge_Dens [ num_mergers −1UL] = 0.0
    num_mergers = num_mergers + 1UL
    merge_index = ulong ( where ( cells . Void_ID eq void_two ))
    cells . Void_ID [ merge_index ] = void_one
    voids . Num_Cells [ void_one −1UL] = voids . Num_Cells [ void_one −1UL] + voids .
        Num_Cells [ void_two −1UL]
    voids . Num_Cells [ void_two −1UL] = 0UL

  endelse
```

```
        endfor

      endif

    endelse

  endfor

  voids_dens = where(voids.Num_Cells ne 0UL)
  num_voids_dens = ulong(n_elements(voids_dens))
  if (num_voids_dens gt 100000UL) then begin
    print, 'Number of voids exceeds maximum!!!'
    stop
  endif
  print, 'Number of voids at this density: ', num_voids_dens, format='(A, I7)'
  print, 'Calculating statistics for each void...'

  cell_adj = header.cell_size / 2.0

  void_stats = { Void_ID          : ulonarr(num_voids_dens), $
                 Origin           : ulonarr(num_voids_dens), $
                 Num_Cells        : ulonarr(num_voids_dens), $
                 Num_Gals         : ulonarr(num_voids_dens)  $
               }

  for v=0UL, num_voids_dens-1UL do begin
    idx = voids_dens[v]
    current_void_ID =  voids.Void_ID[idx]
    print, '   Void '+string(v, format='(I7)')+':   Void_ID = '+string(current_void_ID,
        format='(I7.7)')
    void_stats.Void_ID[v] = current_void_ID
    void_stats.Origin[v] = voids.Origin[idx]
    void_stats.Num_Cells[v] = voids.Num_Cells[idx]
    void_cells = where(cells.Void_ID eq current_void_ID)
    void_stats.Num_Gals[v] = total(cells.Galaxies[void_cells])
  endfor

  printf, slun, 'Resolution: ', header.cell_size, ' Mpc', format='(A, I2, A, //)'
  printf, slun, 'Overdensity: ', curr_dens - 1.0, format='(A, F6.3, /)'
```

```
x = void_stats.Num_Cells * header.cell_size^3
tempindex = where(void_stats.Num_Cells ne max(void_stats.Num_Cells))
y = void_stats.Num_Cells[tempindex] * header.cell_size^3
average = mean(float(x))
saverage = mean(float(y))
std = stddev(float(y))
voidgal = total(void_stats.Num_Gals)
bindata = alog10(x / float(total(x)))
histdata = float(histogram(bindata, min = -7.0, max = 0.0, nbins = 70))
tempindex2 = where(histdata le 1.0E-6)
histdata[tempindex2] = 0.1                           ;- Fudge so we can plot
    the log of the data
histdata = alog10(histdata)
histbins = float(indgen(70)) * (7.0 / 70.0) - 7.0


printf, slun, 'Total Volume in Voids: ', total(x), ' Mpc^3', format='(A, E9.3, A)'
printf, slun, 'Volume Filling Fraction: ', total(x) / 500.0^3, format='(A, F5.3)'
printf, slun, 'Volume of Largest Void: ', max(x), ' Mpc', format='(A, E9.3, A)'
printf, slun, 'Fraction of Void Volume in Largest Void: ', max(x) / total(x), format
    ='(A, F5.3)'
printf, slun, 'Number of Voids: ', ulong(n_elements(x)), format='(A, I5)'
printf, slun, 'Average Volume of All Voids: ', average, ' Mpc^3', format='(A, E9.3,
    A)'
printf, slun, 'Average Volume of Voids (excluding largest): ', saverage, ' Mpc^3',
    format='(A, E9.3, A)'
printf, slun, 'Standard Deviation of Voids (excluding largest): ', std, format='(A,
    E9.3)'
printf, slun, 'Number of Void Galaxies: ', voidgal, format='(A, I7, /)'


fln = 'Hist'+string(header.cell_size, format='(I2.2)')+string(round((curr_dens-1.0)
    *1000), format='(I5.4)')
wave_pson, filename=fln+'.ps'


histtitle = 'Distribution of Voids as a Fraction of Total Void Volume - Res: '+
    string(header.cell_size, format='(I2.2)')+' - OvDens: '+string(curr_dens-1.0,
    format='(F6.3)')
plot, histbins, histdata, /noerase, xstyle=1, ystyle=1, psym=10, xrange=[-7.0, 0.0],
    yrange=[-1, 5], yticklen=1.0, xtitle='Log(Fraction of Void Volume)', ytitle='
```

```
          Log(Number of Voids)', title=histtitle, color='000000'XL


wave_psoff


;- Step through the density range.


old_dens = curr_dens
for curr_dens=ddens, max_dens, ddens do begin
  batch_index = where((nsort_cells.Dens_Ratio gt old_dens) and (nsort_cells.
      Dens_Ratio le curr_dens))
  print, 'Density = ', curr_dens, format='(A, F5.3)'
  if (batch_index[0] ne -1) then begin                    ;- Test to make sure there
      are cells in the range.
    num_cells_step = ulong(n_elements(batch_index))
    low_idx = ulong(min(batch_index))
  endif else begin
    print, 'No cells in this density range.'
    continue
  endelse


  print, 'Cells in this density threshold: ', num_cells_step, format='(A, I7)'
  for ctr=0UL, num_cells_step-1UL do begin                    ;- Go through the new
      void cells.
    if ((ctr mod 1000) eq 0) then print, 'Cell # ', ctr, format='(A, I7)'
    temp_id = nsort_cells.Cell_ID[low_idx+ctr]              ;- The ID of the void cell
        .
    num_neighbors = 6
    if ((cells.Cell_X[temp_id] eq 0) or (cells.Cell_X[temp_id] eq header.
        num_cells_dim-1UL)) then num_neighbors = num_neighbors - 1
    if ((cells.Cell_Y[temp_id] eq 0) or (cells.Cell_Y[temp_id] eq header.
        num_cells_dim-1UL)) then num_neighbors = num_neighbors - 1
    if ((cells.Cell_Z[temp_id] eq 0) or (cells.Cell_Z[temp_id] eq header.
        num_cells_dim-1UL)) then num_neighbors = num_neighbors - 1


    neighbors = { Cell_ID          : ulonarr(num_neighbors+1),    $
                  Void_ID          : ulonarr(num_neighbors+1),    $
                  Flow_Into        : ulonarr(num_neighbors+1),    $
                  Dens_Ratio       : dblarr(num_neighbors+1),     $
                  Mass_Dens_Ratio  : dblarr(num_neighbors+1)      $
```

```
                      }

neighbors.Cell_ID[0] = cells.Cell_ID[temp_id]
neighbors.Void_ID[0] = cells.Void_ID[temp_id]
neighbors.Flow_Into[0] = cells.Flow_Into[temp_id]
neighbors.Dens_Ratio[0] = cells.Dens_Ratio[temp_id]
neighbors.Mass_Dens_Ratio[0] = cells.Mass_Dens_Ratio[temp_id]


i = 1
if (cells.Cell_X[temp_id] ne 0) then begin
  neighbors.Cell_ID[i] = cells.Cell_ID[temp_id-1UL]
  neighbors.Void_ID[i] = cells.Void_ID[temp_id-1UL]
  neighbors.Flow_Into[i] = cells.Flow_Into[temp_id-1UL]
  neighbors.Dens_Ratio[i] = cells.Dens_Ratio[temp_id-1UL]
  neighbors.Mass_Dens_Ratio[i] = cells.Mass_Dens_Ratio[temp_id-1UL]
  i = i + 1
endif
if (cells.Cell_X[temp_id] ne header.num_cells_dim-1UL) then begin
  neighbors.Cell_ID[i] = cells.Cell_ID[temp_id+1UL]
  neighbors.Void_ID[i] = cells.Void_ID[temp_id+1UL]
  neighbors.Flow_Into[i] = cells.Flow_Into[temp_id+1UL]
  neighbors.Dens_Ratio[i] = cells.Dens_Ratio[temp_id+1UL]
  neighbors.Mass_Dens_Ratio[i] = cells.Mass_Dens_Ratio[temp_id+1UL]
  i = i + 1
endif
if (cells.Cell_Y[temp_id] ne 0) then begin
  neighbors.Cell_ID[i] = cells.Cell_ID[temp_id - header.num_cells_dim]
  neighbors.Void_ID[i] = cells.Void_ID[temp_id - header.num_cells_dim]
  neighbors.Flow_Into[i] = cells.Flow_Into[temp_id - header.num_cells_dim]
  neighbors.Dens_Ratio[i] = cells.Dens_Ratio[temp_id - header.num_cells_dim]
  neighbors.Mass_Dens_Ratio[i] = cells.Mass_Dens_Ratio[temp_id - header.
      num_cells_dim]
  i = i + 1
endif
if (cells.Cell_Y[temp_id] ne header.num_cells_dim-1UL) then begin
  neighbors.Cell_ID[i] = cells.Cell_ID[temp_id + header.num_cells_dim]
  neighbors.Void_ID[i] = cells.Void_ID[temp_id + header.num_cells_dim]
  neighbors.Flow_Into[i] = cells.Flow_Into[temp_id + header.num_cells_dim]
  neighbors.Dens_Ratio[i] = cells.Dens_Ratio[temp_id + header.num_cells_dim]
```

```
neighbors.Mass_Dens_Ratio[i] = cells.Mass_Dens_Ratio[temp_id + header.
    num_cells_dim]
  i = i + 1
endif
if (cells.Cell_Z[temp_id] ne 0) then begin
  neighbors.Cell_ID[i] = cells.Cell_ID[temp_id − header.num_cells_dim^2]
  neighbors.Void_ID[i] = cells.Void_ID[temp_id − header.num_cells_dim^2]
  neighbors.Flow_Into[i] = cells.Flow_Into[temp_id − header.num_cells_dim^2]
  neighbors.Dens_Ratio[i] = cells.Dens_Ratio[temp_id − header.num_cells_dim^2]
  neighbors.Mass_Dens_Ratio[i] = cells.Mass_Dens_Ratio[temp_id − header.
      num_cells_dim^2]
  i = i + 1
endif
if (cells.Cell_Z[temp_id] ne header.num_cells_dim−1UL) then begin
  neighbors.Cell_ID[i] = cells.Cell_ID[temp_id + header.num_cells_dim^2]
  neighbors.Void_ID[i] = cells.Void_ID[temp_id + header.num_cells_dim^2]
  neighbors.Flow_Into[i] = cells.Flow_Into[temp_id + header.num_cells_dim^2]
  neighbors.Dens_Ratio[i] = cells.Dens_Ratio[temp_id + header.num_cells_dim^2]
  neighbors.Mass_Dens_Ratio[i] = cells.Mass_Dens_Ratio[temp_id + header.
      num_cells_dim^2]
  i = i + 1
endif


if (min(neighbors.Void_ID) eq 999999999UL) then begin

  if (cells.Dens_Ratio[temp_id] le min(neighbors.Dens_Ratio)) then begin

    ;− This is if no neighbors are already assigned to a void and the new cell
        has the lowest density.
    cells.Flow_Into[temp_id] = cells.Cell_ID[temp_id]
    cells.Void_ID[temp_id] = num_voids
    voids.Void_ID[num_voids−1UL] = num_voids
    voids.Origin[num_voids−1UL] = cells.Cell_ID[temp_id]
    voids.Num_Cells[num_voids−1UL] = 1UL
    num_voids = num_voids + 1UL


  endif else begin

    ;− This is if no neighbors are already assigned to a void but the new cell
```

```
          is not the lowest density.
    cells.Flow_Into[temp_id] = cells.Cell_ID[temp_id]
    cells.Void_ID[temp_id] = num_voids
    voids.Void_ID[num_voids-1UL] = num_voids
    voids.Origin[num_voids-1UL] = cells.Cell_ID[temp_id]
    voids.Num_Cells[num_voids-1UL] = 1UL
    num_voids = num_voids + 1UL


  endelse

endif else begin

  ;- This is if a neighbor is already assigned to a void.
  void_neighbors = where(neighbors.Void_ID ne 999999999UL)
  num_void_neigh = n_elements(void_neighbors)

  temp_void_neigh = {  Cell_ID         : ulonarr(num_void_neigh),   $
                       Void_ID         : ulonarr(num_void_neigh),   $
                       Flow_Into       : ulonarr(num_void_neigh),   $
                       Dens_Ratio      : dblarr(num_void_neigh),    $
                       Mass_Dens_Ratio : dblarr(num_void_neigh)     $
                    }

  temp_void_neigh.Cell_ID = neighbors.Cell_ID[void_neighbors]
  temp_void_neigh.Void_ID = neighbors.Void_ID[void_neighbors]
  temp_void_neigh.Flow_Into = neighbors.Flow_Into[void_neighbors]
  temp_void_neigh.Dens_Ratio = neighbors.Dens_Ratio[void_neighbors]
  temp_void_neigh.Mass_Dens_Ratio = neighbors.Mass_Dens_Ratio[void_neighbors]

  sort_void_idx = uniq(temp_void_neigh.Void_ID, sort(temp_void_neigh.Void_ID))
  num_uniq_voids = n_elements(sort_void_idx)
  cells.Flow_Into[temp_id] = temp_void_neigh.Cell_ID[0]
  cells.Void_ID[temp_id] = temp_void_neigh.Void_ID[0]
  voids.Num_Cells[temp_void_neigh.Void_ID[0]-1UL] = voids.Num_Cells[
      temp_void_neigh.Void_ID[0]-1UL] + 1UL

  if (num_uniq_voids ne 1) then begin

    void_one = temp_void_neigh.Void_ID[sort_void_idx[0]]
```

```
for j=1, num_uniq_voids−1 do begin

  void_two = temp_void_neigh.Void_ID[sort_void_idx[j]]
  ;− Merge the smaller void into the larger one, defaulting to the lower #
      in a tie.

  if (voids.Num_Cells[void_one−1UL] lt voids.Num_Cells[void_two−1UL]) then
      begin

    mergers.Void_1_ID[num_mergers−1UL] = void_two
    mergers.Void_1_Size[num_mergers−1UL] = voids.Num_Cells[void_two−1UL]
    mergers.Void_2_ID[num_mergers−1UL] = void_one
    mergers.Void_2_Size[num_mergers−1UL] = voids.Num_Cells[void_one−1UL]
    mergers.Merge_Dens[num_mergers−1UL] = curr_dens
    num_mergers = num_mergers + 1UL
    merge_index = ulong(where(cells.Void_ID eq void_one))
    cells.Void_ID[merge_index] = void_two
    voids.Num_Cells[void_two−1UL] = voids.Num_Cells[void_two−1UL] + voids.
        Num_Cells[void_one−1UL]
    voids.Num_Cells[void_one−1UL] = 0UL
    void_one = void_two

endif else begin

    mergers.Void_1_ID[num_mergers−1UL] = void_one
    mergers.Void_1_Size[num_mergers−1UL] = voids.Num_Cells[void_one−1UL]
    mergers.Void_2_ID[num_mergers−1UL] = void_two
    mergers.Void_2_Size[num_mergers−1UL] = voids.Num_Cells[void_two−1UL]
    mergers.Merge_Dens[num_mergers−1UL] = curr_dens
    num_mergers = num_mergers + 1UL
    merge_index = ulong(where(cells.Void_ID eq void_two))
    cells.Void_ID[merge_index] = void_one
    voids.Num_Cells[void_one−1UL] = voids.Num_Cells[void_one−1UL] + voids.
        Num_Cells[void_two−1UL]
    voids.Num_Cells[void_two−1UL] = 0UL

endelse
```

```
          endfor

        endif

      endelse

    endfor

voids_dens = where(voids.Num_Cells ne 0UL)
num_voids_dens = ulong(n_elements(voids_dens))
print, 'Number of voids at this density: ', num_voids_dens, format='(A, I7)'
print, 'Calculating statistics for each void...'

cell_adj = header.cell_size / 2.0

void_stats = { Void_ID          : ulonarr(num_voids_dens), $
               Origin           : ulonarr(num_voids_dens), $
               Num_Cells        : ulonarr(num_voids_dens), $
               Num_Gals         : ulonarr(num_voids_dens)  $
             }

for v=0UL, num_voids_dens-1UL do begin
  idx = voids_dens[v]
  current_void_ID =  voids.Void_ID[idx]
  print, '   Void '+string(v, format='(I7)')+':  Void_ID = '+string(current_void_ID,
       format='(I7.7)')
  void_stats.Void_ID[v] = current_void_ID
  void_stats.Origin[v] = voids.Origin[idx]
  void_stats.Num_Cells[v] = voids.Num_Cells[idx]
  void_cells = where(cells.Void_ID eq current_void_ID)
  void_stats.Num_Gals[v] = total(cells.Galaxies[void_cells])
endfor

printf, slun, 'Overdensity: ', curr_dens - 1.0, format='(A, F6.3, /)'

x = void_stats.Num_Cells * header.cell_size^3
tempindex = where(void_stats.Num_Cells ne max(void_stats.Num_Cells))
y = void_stats.Num_Cells[tempindex] * header.cell_size^3
average = mean(float(x))
```

```
saverage = mean(float(y))
std = stddev(float(y))
voidgal = total(void_stats.Num_Gals)
bindata = alog10(x / float(total(x)))
histdata = float(histogram(bindata, min = -7.0, max = 0.0, nbins = 70))
tempindex2 = where(histdata le 1.0E-6)
histdata[tempindex2] = 0.1                          ;- Fudge so we can plot
    the log of the data
histdata = alog10(histdata)
histbins = float(indgen(70)) * (7.0 / 70.0) - 7.0


printf, slun, 'Total Volume in Voids: ', total(x), ' Mpc^3', format='(A, E9.3, A)'
printf, slun, 'Volume Filling Fraction: ', total(x) / 500.0^3, format='(A, F5.3)'
printf, slun, 'Volume of Largest Void: ', max(x), ' Mpc', format='(A, E9.3, A)'
printf, slun, 'Fraction of Void Volume in Largest Void: ', max(x) / total(x), format
    ='(A, F5.3)'
printf, slun, 'Number of Voids: ', ulong(n_elements(x)), format='(A, I5)'
printf, slun, 'Average Volume of All Voids: ', average, ' Mpc^3', format='(A, E9.3,
    A)'
printf, slun, 'Average Volume of Voids (excluding largest): ', saverage, ' Mpc^3',
    format='(A, E9.3, A)'
printf, slun, 'Standard Deviation of Voids (excluding largest): ', std, format='(A,
    E9.3)'
printf, slun, 'Number of Void Galaxies: ', voidgal, format='(A, I7, /)'

fln = 'Hist'+string(header.cell_size, format='(I2.2)')+string(round((curr_dens-1.0)
    *1000), format='(I5.4)')
wave_pson, filename=fln+'.ps'


histtitle = 'Distribution of Voids as a Fraction of Total Void Volume - Res: '+
    string(header.cell_size, format='(I2.2)')+' - OvDens: '+string(curr_dens-1.0,
    format='(F6.3)')
plot, histbins, histdata, /noerase, xstyle=1, ystyle=1, psym=10, xrange=[-7.0, 0.0],
     yrange=[-1, 5], yticklen=1.0, xtitle='Log(Fraction of Void Volume)', ytitle='
    Log(Number of Voids)', title=histtitle, color='000000'XL


wave_psoff


old_dens = curr_dens
```

```
        endfor

;- Write the void file.
print, 'Writing density output file ...'

writefilename =  filepath('MR_density.bin', root_dir='/media/usbdisk/MRData/VoidData'+string
    (header.cell_size, format='(I2.2)')+'/')

dens_map = { Cell_ID    : ulonarr(header.total_cells), $
             Dens_Ratio : dblarr(header.total_cells)   $
           }
dens_map.Cell_ID = cells.Cell_ID
dens_map.Dens_Ratio = cells.Dens_Ratio

openw, 1, writefilename
writeu, 1, header
writeu, 1, dens_map
close, 1

print, 'Done!'

;- Write the merger file.
print, 'Writing merger output file ...'

writefilename =  filepath('MR_mergers.txt', root_dir='/media/usbdisk/MRData/VoidData'+string
    (header.cell_size, format='(I2.2)')+'/')
openw, mlun, writefilename, /get_lun

for k=0UL, num_mergers-2UL do begin
  if (mergers.Merge_Dens[k] gt 0.05) then printf, mlun, mergers.Void_1_ID[k], ' ', mergers.
      Void_1_Size[k], ' ', mergers.Void_2_ID[k], ' ', mergers.Void_2_Size[k], ' ', mergers.
      Merge_Dens[k], format='(I7.7, A, I7.7, A, I7.7, A, I7.7, A, F4.2)'
endfor

free_lun, mlun

print, 'Done!'
free_lun, slun
```

```
print , 'Run complete!'

END
```

Bibliography

Bennett, C. L., et al., 2003, ApJS, 148, 1

Berrier, J. C., et al., 2006, ApJ, 652, 56

Bershady, M. A., Lowenthal, J. D., Koo, D. C., 1998, ApJ, 505, 50

Beucher, S., Lantuejouil, C., 1979, Proceedings International Workshop on Image Processing, CCETT/IRISA, Rennes, France

Beucher, S., Meyer, F., *Mathematical Morphology in Image Processing* (Marcel Dekker Inc., New York, 1993), 433

Colberg, J. M., et al., 2008, MNRAS, 387, 933

Cole, S., et al., 2005, MNRAS, 362, 505

Colless, M., et al., 2001, MNRAS, 328, 1039

Croton, D. J., et al., 2006, MNRAS, 365, 11

Dolch, T., Ferguson, H. C., 2005, Bull. AAS, 37, 1193

El-Ad, H., Piran, T., 1997, ApJ, 491, 421

Ellis, S. C., Bland-Hawthorn, J., 2007, MNRAS, 377, 815

Folkes, S., et al., 1999, MNRAS, 308, 459

Frith, W. J., Metcalfe, N., Shanks, T., 2006, MNRAS, 371, 1601

Fukugita, M., et al., 2004, AJ, 127, 3155

Georgakakis, A., et al., 2006, ApJ, 641, 101

Grogin, N. A., Geller, M. J., 1999, AJ, 118, 2561

Hoyle, F., Vogeley, M. S., 2004, ApJ, 607, 751

Huchra, J. P., et al., in *Large Scale Structures of the Universe*, volume 130 (IAU, 1988), 105

Kirshner, R. P., et al., 1983, AJ, 88, 1285

Kling, R., *Application Development with IDL* (Ronn Kling Consulting, Warrenton, VA, 2000)

Kong, X., et al., 2006, ApJ, 638, 72

Lindner, U., et al., in K. Sato (ed.), *Cosmological Parameters and the Evolution of the Universe*, volume 183 (IAU, 1999), 256

Liske, J., et al., 2003, MNRAS, 344, 307

López-Corredoira, M., Betancort-Rijo, J. E., 2004, A&A, 416, 1

Metcalfe, N., et al., 2006, MNRAS, 370, 1257

Montero-Dorta, A. D., Prada, F., 2009, MNRAS, 399, 1106

Neufeldt, V. (ed.), *Webster's New World Dictionary of American English* (Simon & Schuster, Cleveland & New York, 1988), third college edition

Platen, E., van de Weygaert, R., Jones, B. J. T., 2007, MNRAS, 380, 551

Popescu, C. C., Hopp, U., Rosa, M. R., 1999, A&A, 350, 414

Rojas, R. R., et al., 2004, ApJ, 617, 50

Schechter, P., 1976, ApJ, 203, 297

Spergel, D. N., et al., 2003, ApJS, 148, 175

Springel, V., Frenck, C. S., White, S. D. M., 2006, Nature, 440, 1137

Springel, V., et al., 2005, Nature, 435, 629

Treister, E., et al., 2006, ApJ, 640, 603

Yasuda, N., Fukugita, M., Schneider, D. P., 2007, AJ, 134, 698