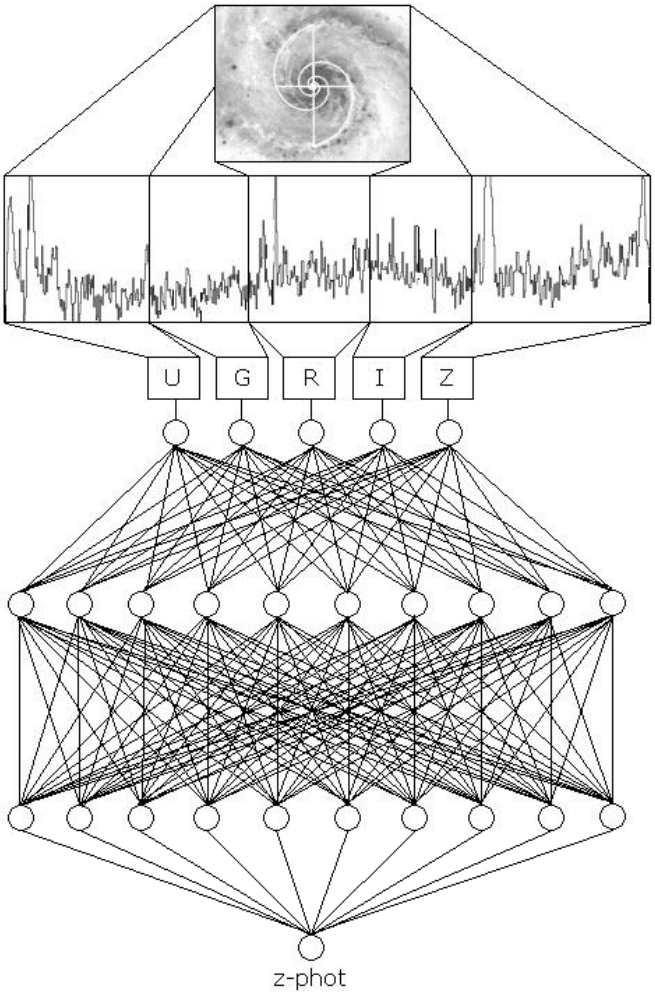
# Estimating photometric redshifts using Artificial Neural Networks.




*Author:*
Erling Johnsen

*Supervisors:*
Jens Hjorth
Kristian Pedersen

Dark Cosmology Centre, Niels Bohr Institute
Faculty of Science, University of Copenhagen

July 3, 2008

# Contents

# Chapter 1

# Introduction

Determination of the spectroscopical redshift of distant galaxies, by means of direct measurements of the shifts of known spectral lines, with respect to the well known wavelenghts in an Earth laboratory, associated with a specific nuclear transition between states in a certain atomic configuration, is by far the most precise method of distance estimations to distant galaxies. However, when looking at an object, too distant, and too faint to ensure high enough signal-to-noise ratios, some sort of indirect method, using photometry as a starting point, has to be used in order to determine the model dependent distance to the object. Throughout the years researchers have used different kinds of methods, competing to obtain better and better results with increasingly lower uncertainty than their predecessors, using technolocical improvements to their advantage.
One of these methods, demonstrated by Lehav e.a. in 2001 [1], uses the now easier accesible computational power to train Artificial Neural Networks (ANN), in recognising patterns between results obtained from less faint objects, where both spectroscopy and photometry are applicable, and afterwards applying these recognized patterns to the fainter objects, where only photometry is possible.
The starting point of my thesis is the publication by Lehav e.a. and the open source code (once?) available for download.[1] I'll demonstrate how the program package works, both in theory and in practice with the ready-to-use test samples, and follow up with results from newly extracted and filtered data from the SDSS database. Later I'll add my own improved code and demonstrate how this highly improves the results, as well as the chance of creating a tool for future use in different parts of astronomy.

[1]The package used to be available for download at http://www.ast.cam.ac.uk/~aac, but haven't been the last couple of times, I've checked.

# Chapter 2

# Astrometry

Estimation of distances to celestrial bodies, along with determination of position and movement of such, is part of the science discipline *astrometry*, reaching back to the greek philosopher Hippharcus. Around 190 B.C. he discovered the Earth's precession, and in doing so, also invented the apparent brightness scale that is the predessesor of the (slightly contraintuitive) one still in use. Although the determination process itself can be cumbersome and far from trivial, the discipline often receives less attention than the fields, where the results are later used as very important inputs. However, motivation for trying to reach further and further with increasingly lower uncertainty is highly justified, since the results are used widely in many different scientific fields. From fine tuning of the parameters of Big Bang theory, and the following nuclesynthesis throughout the evolution of the Universe and it's inhabitants, via morphological evolution of galaxies, to a whole lot of other fields as large scale dynamics (galaxies and clusters of such), theory for exotic objects like black holes, white dwarves, (super)novae, gamma ray bursts and the distribution and nature of dark matter.

## 2.1 The astronomical distance ladder.

The astronomical distance ladder is the common term for the different methods used for determination of distances to astronomical objects. The term *ladder* is used, because each method reaches out into space one step further, some of which have certain overlays, making intercalibration possible. [4]

In Table 2.1 on page 6 a selection of different measuring techniques are listed along with the typical objects the methods are applied to, and the distances they are approximately applicable for. However, it is not my intension to dig further into the different steps of this ladder, but it serves as a

| Method | Typical object | Distance |
|---|---|---|
| Radar | Inner planets / Asteroids | Solar system |
| Direct measurement of orbits | Planets / Comets | Solar system |
| The parallax method | Alpha Centauri | 100pc |
| Main sequence fitting | Open clusters of stars | 100kpc |
| Bright objects | Cepheids / Novae | 5Mpc |
| Individual galaxies in clusters | Virgo Cluster | 10 Mpc |
| The Tully-Fisher relation | Large Spirals | 10 Mpc |
| Supernovae fitting | Type Ia supernovae | $10^9$pc |
| Redshifts and Hubble's Law | Distant galaxies | ? |

Table 2.1: methods

means of comprehending the domain of this thesis, should anyone not being too familiar with the terms, find an interest in reading it.

## 2.2 Spectroscopy

By splitting up the incoming light from a distant object, suchs as a galaxy, with a prism or a slit, into it's different wavelengths, the object's spectrum will appear, showing distinct differences in intensity at each wavelenght when measured. Some of the most striking features in the spectrum can be recognized as appearing at almost, but not quite, the wavelenghts of well known spectral lines from certain transitions of states in a specific atom, ion or molecule, with respect to well controlled experiments in Earth laboratories. Figure 2.1 on page 7 is an example of how to obtain the spectroscopical redshift. The lines marked with arrows are the hydrogen alpha, beta and gamma lines respectively, which are associated with electron transitions from principal quantum number levels 3, 4 and 5 to level 2 above the ground state 1. Normally these lines would be found at 6563Å, 4861Åand 4341Å, but instead they are localized at much higher wavelenghts. In other types of galaxies, lines due to transitions in other atoms might be easier recognized.
As closer examination shows, all of these lines are shifted proportionally towards the red, less energetic, end of the spectrum by an amount given by z:

$$z = (\lambda - \lambda_0)/\lambda_0, \tag{2.1}$$

where $\lambda$ is a recognizable spectral line shifted from it's resting wavelenght given by $\lambda_0$.
Furthermore, research early in the 20th century showed the now well known

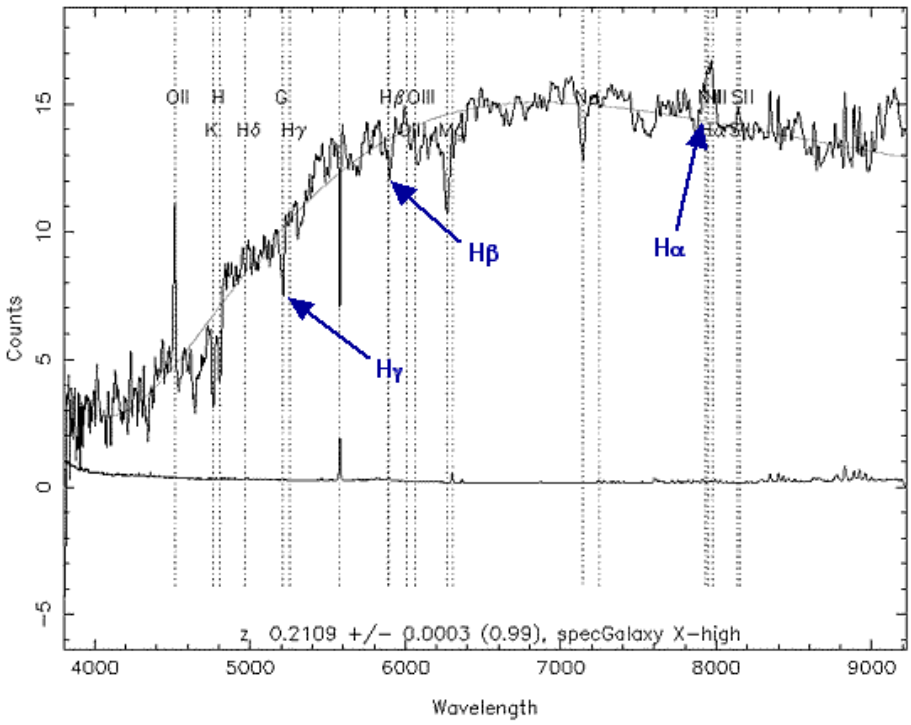



Figure 2.1: Example of spectroscopical redshift determination

fact that the greater distance between any two galaxies, the greater their relative speed of separation from each other will be, and subsequently the greater the shifts of the easily identifiable spectral lines towards the red (less energetic) end, are. This empirically observed proportionality, nowadays known as Hubble's law, was established and formulated by Edwin Hubble together with Milton L. Humason In 1929. It is consistent with the solutions of Einstein's equations of general relativity for a homogeneous, isotropic expanding space, and can simply be writen as:

$$z = (H_0/c)d, \tag{2.2}$$

where d is the distance to the galaxy, $H_0$ is the so called Hubble 'constant' which has dimensions of velocity divided by distance, the index, '0', refers to the value H has now, and c is the speed of light. If the redshift is interpreted as a measure of recession speed, v, between the objects, and $v << c$ so that

$$z \approx v/c, \tag{2.3}$$

then the recessional velocity will be given by:

$$v = H_0 d. \tag{2.4}$$

A rather precise calculation of the proportionality constant, using the satellite Wilkinson Microwave Anisotropy Probe (WMAP) that produced the first full-sky map of the microwave background with a resolution under one degree, began in 2003, yielding a value of $71 \pm 4$ (km/s)/Mpc [1], which expresses that for each mega parsec [2] away a galaxy is from us, the space between us expands with an additional 71 $km/s$. The fact that all the recognizable lines are shifted with an amount, only governed by the distance to the object is the interesting part, since determination of the shift of the spectroscopical lines then becomes a direct distance measurement - the further the lines are shifted, the more the space between the object and the observer has expanded since the light was emitted, and thus the further away the object is.
When applicable, spectroscopy is by far the most precise method for distance estimation to extragalactic sources, but in the end it only returns a relative order of distances to different objects that later on has to be connected to absolute values through fine tuning of a cosmological model.[4] Obviously, the success of spectroscopy rests on whether or not it is possible to split the spectrum into lines, and still gain a high enough signal-to-noise ratio to precisely determine the center of easily identifiable lines known from controlled experiments in laboratories on Earth. Since the flux density of a distant object wears off as the square of the distance to the object [3], the success is ultimately a matter of distance and absolute brightness of the object as well as technical development.

## 2.3 Photometry - indirect methods

When spectroscopy fails, photometry usually prevails.
Photometry also involves a lens splitting up the incoming light, but now the light is passed through different filters, allowing only photons with wavelenghts within a certain bandwidth to pass through, whereby you can effectively measure the brightness contribution from each of the intervals at the same time. This brightness distribution, of course, is a very coarse approximation to the actual *spectral energy distribution* (SED), but it has both a higher natural boundry for application than spectroscopy, and can be done without time consuming integrations, since photometry integrates the incoming intensity of light from a broad interval of wavelenghts, whereas spectroscopy handles much smaller resolutions, such that photometry can be thought of as a very coarse, low-resolution, spectroscopy.
So when the direct spectroscopic method shows too low signal-to-noise ratios

[1] $http://map.gsfc.nasa.gov/universe/uni_expansion.html$
[2] $1parsec = 3,27ly = 3,86 * 10^{16}m$

(or speed is a more important aspect than precision), more imaginative and indirect methods has to be used, taking photometry as it's starting point. These methods are with one common term called methods of *photometric redshift determination* and as we'll see in the following sections, ANN is just one method among many. Naturally what is gained on wider application, is instantly lost on resolution - where spectroscopy resolves single lines resulting from atomic transitions, photometry integrates the incoming light from a whole bandwith of these, thus incorporation higher unavoidable uncertainties.

### 2.3.1 Direct shift measurement

The first who succeded in developing a method for calculating the redshift photometrically was W.A.Baum[8], who in 1962 demonstrated it by a series of 9 bandpass filters. The method was practically a matter of plotting the averaged SED of a couple of galaxies in the Virgo cluster together with the averaged SED of members of another cluster, Abell 0801, and then measuring the displacement on a logarithmic scale. The resulting value of z=0,19 seems incredibly close to the spectroscopical one of z=0,192 and the project was therefore extended to other clusters out to a spectroscopical value of z=0.46, which made a fairly accurate estimation (at the time) of the cosmological density parameter, $\Omega_0$ possible. However, the method relied heavily on a rather large spectral disconitinuity feature at around 4000Å, and could thus only be applied to elliptical galaxies containing an old stellar population, where this feature is particularly significant [7]

### 2.3.2 Template fitting.

Not two galaxies are alike, but nevertheless it is possible to divide the myriad of galaxies into a number of subsets on basis of different objective empirical parameters, first of all starting with the Hubble fork diagram as in figure 2.2 on page 10. The template fitting method starts with the user setting up a grid of standard galaxies at different z-values. At each grid point a set of flux values are given, representing the center of a bandwith similar to the photometric bandwiths the galaxies, which unknown z-values are sought, later on has to be estimated on basis of. The flux values are determined from SEDs that are either based on empirical data or synthesised (i.e. computer simulated) spectra. The template fitting method then becomes a matter of breaking down the vastness of different types of galaxies into smaller and smaller subsets, calculating their different flux values for different values of z. Finally, when looking at a test galaxy, it is a matter of localising the

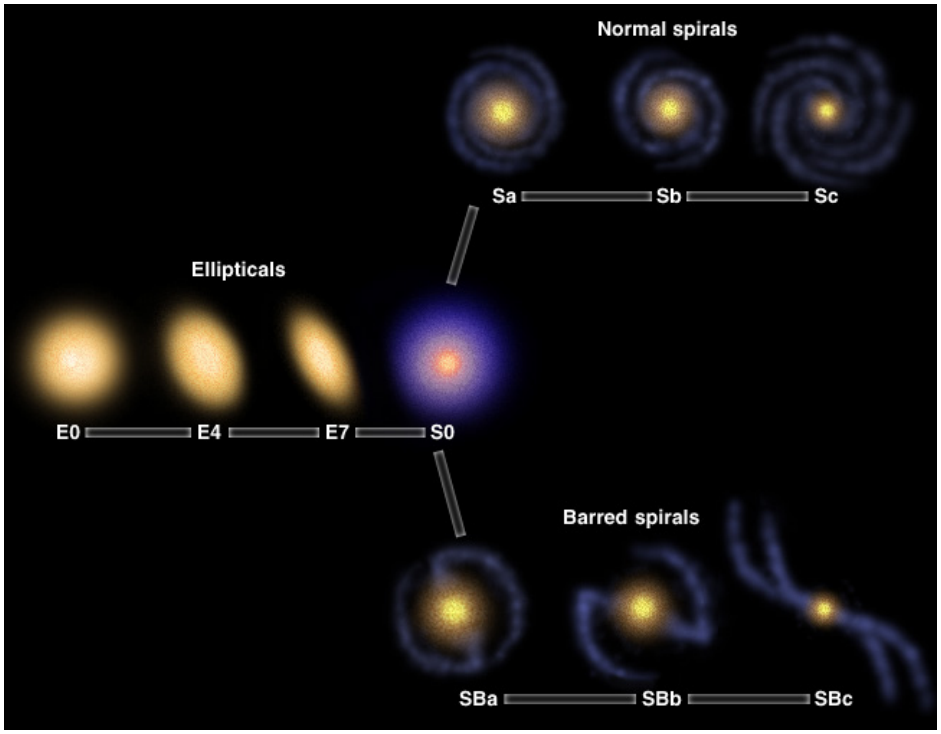



Figure 2.2: Classical Hubble fork diagram.

grid point, closest resembling the object in focus, and thus adopting the z-value directly from the grid - or estimating it as an interpolating between a couple of them as some sort of a weighted average between points, where a chi-square function is sought lowest as given by something like:

$$\chi^2 = \sum_{i=1}^{N_{filters}} \left[ \frac{F_{obs,i} - bF_{temp,i}(z)}{\sigma_i} \right]^2, \tag{2.5}$$

where $F_{obs,i}$ and $F_{temp,i}$ are the fluxes in the N filters of a test galaxy and a grid point respectively, and $\sigma$ is the variance.
Obviously the big advantage of the method is speed, when first the grid has been set up - a process however that can be very cumbersome. However extending results beyond the upper boundaries of the grid is either not possible or not advisable. The method was originally tested by Loh & Spillar (1986), where they gained a standard deviation of 0,12 by testing on 34 galaxies of known redshifts all part of the same cluster. They too extended their research to galaxies of unknown redshift (1000 in total) in an attempt to calculate the value of $\Omega_0$.

### 2.3.3 Color-color diagrams

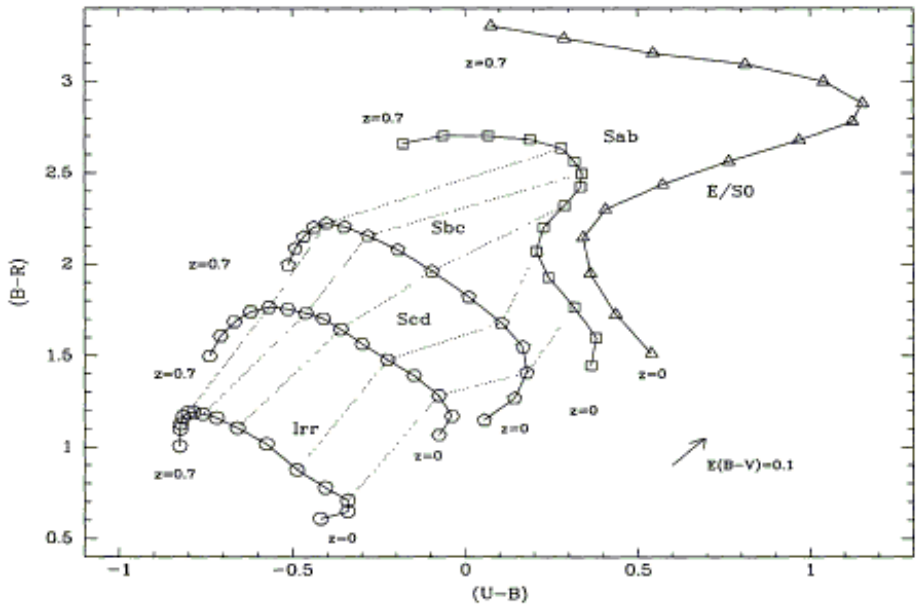



Figure 2.3: Color evolutionary tracks in U-B vs. B-R for the empirically derived template galaxy spectral types. The tracks assume no luminosity evolution with redshift. Each point on the tracks represents a stepwise increase in z of 0.05. The dotted lines show the iso-z contours for z = 0.1, 0.2, 0.3, 0.4 and 0.5.

Color-color diagrams are somewhat a form of template fitting, best described by looking at figure 2.3 on page 11. It shows a plot made on basis of empirically derived template galaxy spectral types (as opposed to templates calibrated on basis of theoretical SEDs (e.g. Bruzual 1983)). Each point corresponds to the color (total difference in flux between two neighbouring filters) of a Hubble galaxy type at a certain redshift z. All galaxy types are plotted at different values of z, ranging form 0.0 to 0.7, and illustrative iso-z lines are added to the late type galaxies. B.D. Koo (1985) then testet the validity of the method on 100 galaxies with known spectroscopical redshifts, by addapting the redshift of the point in the template system, closest to the point representing a test galaxy. As with all template fitting techniques, the quality relies heavily on the resolution of the templates - small deviations from the main classifications of the Hubble types, could result in rather large errors.

### 2.3.4 Linear regression

Linear regression is the direct predessesor of this thesis' main subject, artificial neural networks, as the later is basically an extended version of the former. Both methods takes off with an assumption that the redshift, z, can be writen as a linear combination of some sort of preselected functions that somehow include the photometric magnitudes (or colors for that matter) as inputs, and then the physically obscure coefficients has to be found by numerical, iteration constrained to a simple setup. In the case of linear regression, it can either be given by a linear function:

$$z = c_0 + \sum_{i=1,..,N} c_i m_i \tag{2.6}$$

or an extended quadratic:

$$z = c_0 + \sum_{i=1,..,N} c_i m_i + \sum_{\substack{j=i,..,N \\ i=1,..,N}} c_{ij} m_i m_j, \tag{2.7}$$

where, the c's are the sought coefficients, and the m's are the magnitude (or color) values of each filter. Applied to a training sample of a couple of hundred galaxies or more at the same time, the task becomes a matter of seeking the values of the c's returning the minimum total deviation for the whole sample. Having found this then allows for using the same parametrization for a similar object that was not used in determining it in the first place. As we will see in the following, good results rely on high resemblence between members of training -and test samples. By training on a variety of galaxy types, you ensure a wider usability, but add unnessesary info to the values of the constants. When blessed with large amounts of training data, it seems logical to include a selection mechanism, thus training on different galaxies depending on what sort of galaxy you intend to apply the parametrization to. In fact that's exactly what Y.H. Zhao e.a. did as in [6] (2007), where they used different resulting parametrizations on different types of morphological galaxies based on the SDSS galaxy spectral type spec *eClass* as additional input to the model colors.

### 2.3.5 Artificial Neural Networks

Artificial Neural Networks (ANN) bear their name, because one of the motivations for using them, is to obtain a better understanding of how the neurons of the human brains function, acquire knowledge and remember this for future use. In essence ANN recognizes patterns between data, in much

the same way as the brain does, without leaving much insight as to how this is actually done. However, with well sorted data, these recognized patterns are highly efficient in working were no other method is applicable, mainly because it is not taken into considerations, what the nature of the data is.
What the ANN basically does (in this context at least), is to recognize a pattern in the inputs from a less precise, but more applicable method, and the output(s) from a more precise method with applicative limitations from the same object. It all sounds rather far fetched, but broken down, it's *just* a more complex way of doing what was outlined in 2.3.4 - in fact equation 2.6 on page 12 can be thought of as the smallest possible artificial neural network with a X:1 architecture (see more below).
In the case of determination of photometric redshifts, results from spectroscopy and model color magnitudes from photometry for the same rather large number of objects, are fed into the training program. Afterwards the resulting pattern, the trained network, can be used to run similar objects through, where spectroscopy is technically impossible. This nifty little trick is what this thesis is all about, and the next chapter, will dig into, how the theory is worked out...

# Chapter 3

# General theory of ANN

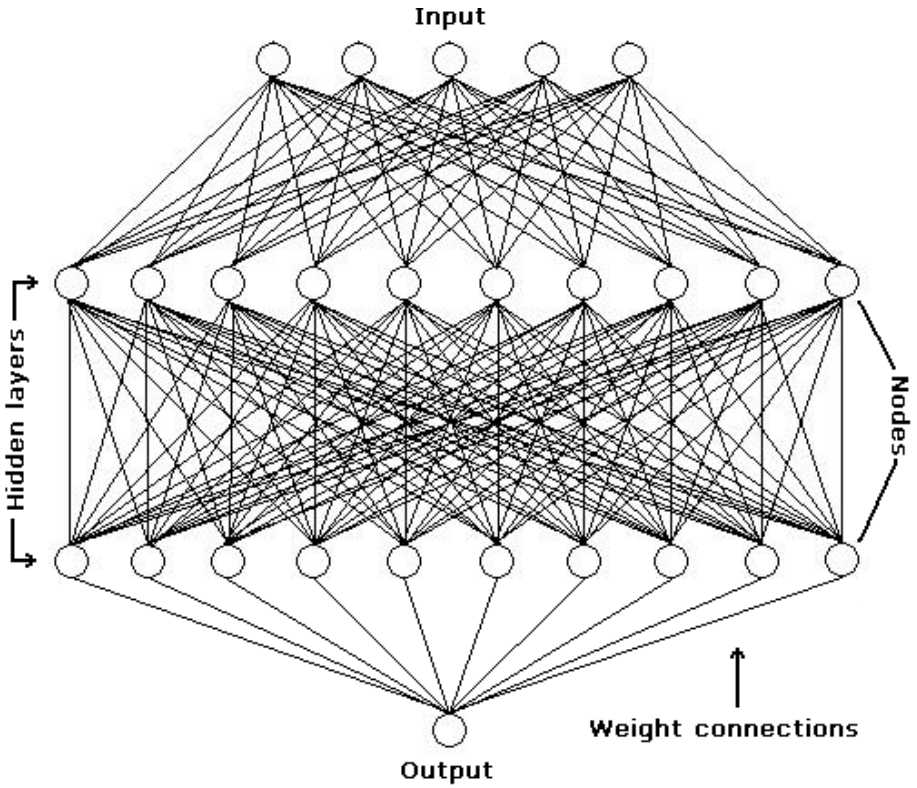



Figure 3.1: Artificial Neural Network structure. In the topmost layer are the input nodes, connected through weights to nodes in two hidden layers, which themselves are connected through other weights to the output node

Artificial Neural Networks come in many variations with the supervised, feed forward, multi layer perceptron as the simplest and wideliest used. They

are build up as in figure 3.1 on page 14 of so-called *nodes* arranged in *layers* in a certain preselected *architecture* that descibes how many nodes are placed in each layer. Each node is connected through *weights* with any other node in the layers above and below. The first layer is the *input layer*, the last one is the *output layer* and the ones in between are said to be *hidden layers*. The ANN in figure 3.1 has a 5:10:10:1 architecture with 5 inputs, 1 output, and 20 other nodes divided into 2 hidden layers with 10 in each. The resulting values, the *activities* of the nodes in the output layer, is what we seek, whereas the values of the intermediate layers are of no immediate interest, since no physical meaning can be extracted from other nodes than the inputs (obviously) and the output(s).

The *activations* of the nodes in layer i above j are given by a predefined activation function, $g_i(u_i)$, which is selected corresponding to the problem at hand. As simple a function as a linear $g(u) = u$ can be chosen, but usually the logistic sigmoid is preferred, since it resembles the activation responses of neurons in the human brain. The logistic sigmoid is used in all of the following and is given by [9]:

$$g(u) = \frac{1}{1 + e^{-u}}, (g \in (0, 1)) \tag{3.1}$$

The activity, $u_j$, of any node, j, is then given as a summation of products between the activations, $g_i(u_i)$, of all the nodes, i, in the layer above, and the strenght of the weights connected to the node, according to:

$$u_j = \sum_i^n g_i(u_i) w_{ij}, \tag{3.2}$$

where $w_{ij}$ is the weight of the connection between node i and j. Now, the output of the network, $y = y(\mathbf{x}; \mathbf{w})$, is a nonlinear function of the input vector, $\mathbf{x}$, parametrized by the weight vector $\mathbf{w}$. The result is therefore:

$$y(\mathbf{x}; \mathbf{w}) = \frac{1}{1 + e^{-\mathbf{w} \cdot \mathbf{x}}} \tag{3.3}$$

The central idea of a supervised network is then this: Given examples of a relationship between an input vector $\mathbf{x}$ and a target t (either scalar or vector), we hope to make the network recognize a pattern between $\mathbf{x}$ and t. A successfully trained network will, for any given $\mathbf{x}$, give an output, y (same dimensions as t), that is close (in some sense) to the target value t. So "training" the network involves searching the weight space for a value $\mathbf{w}$ that produces a function that fits the provided data well.

## 3.1 Estimating photometric redshifts with ANN

Initially the strenght of the weights are randomized and the data are divided into 3 samples named respectively *training*, *validation* and *testing* samples. Common for both the training and validation samples are that the resulting outputs are actually known from other experiments (in this case from spectroscopy). The testing sample is not used in the actual training proces. By carefully calculating the activity of each node on basis of each line of input (photometrical magnitudes) from both the training -and the validation sample in a logical *Feed Forward* progress, the output(s) will result (the photometric redshifts in this context). The *supervised* part of the algorith is now that each line of data in the training sample are associated with the actual result (with much less uncertainty at least) known from spectroscopy. When one cycle has ended, the standard deviation of all the residuals between the photometric outputs and the spectroscopical results are calculated both on the training -and the validation samples. By making a small change to the weights and running through the networks again, the gradient and second order derivatives that form the basis of the so-called Hessian matrix around the point in the weight space, can be calculated on basis of the change of the outputs of the training sample, such that the changes applied to the weights before the next run are optimum.
If one wants to minimize a function, $f(x)$, it's shape around a point, $a$, is given by Taylor's expansion theorem, saying that [11] and [12]:

$$f(x_0 + \Delta x) \approx f(x_0) + \nabla f(x_0)^T \Delta x + \frac{1}{2} \Delta x^T B \Delta x, \tag{3.4}$$

where $\Delta x$ is the incremental distance from the point $a$, $\nabla f$ is the gradient, $T$ the transposition, and $B$ the square Hessian matrix composed of the second-order partial derivatives of the function $f$. Now, the Taylor series of the gradient itself as given by:

$$\nabla f(x_0 + \Delta x) = \nabla f(x_0) + B \nabla x \tag{3.5}$$

is called the secant equation, and as usual when seeking an extremum, setting the gradient equal to zero, does the trick, such that:

$$\nabla f(x_0 + \Delta x_0) = 0, \tag{3.6}$$

results in what we're looking for:

$$\Delta x_0 = -B^{-1} \nabla f(x_0) \tag{3.7}$$

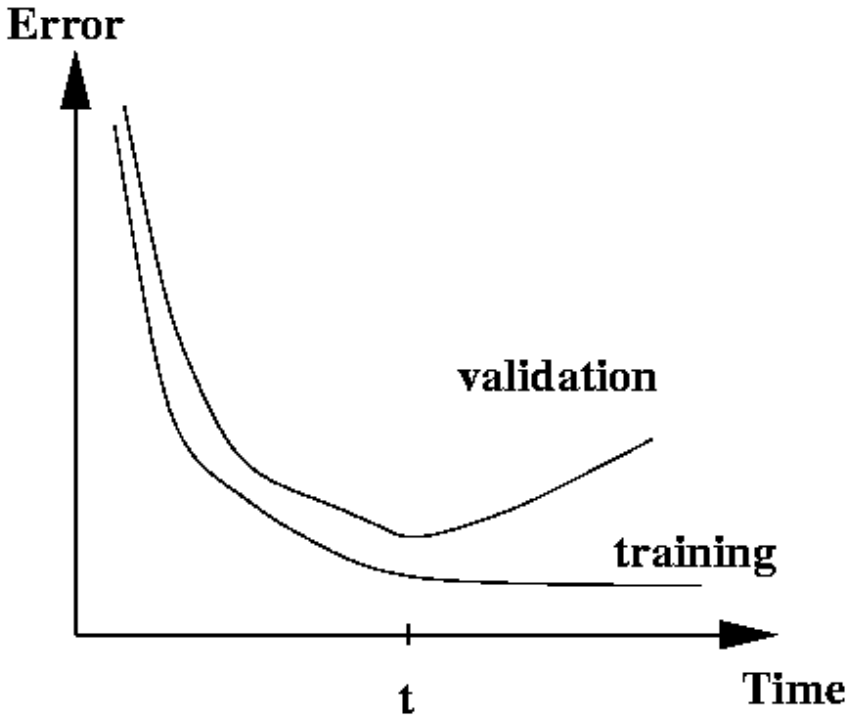



Figure 3.2: To avoid over fitting, the weight vector is selected when rms was minimum calculated on the validation set and not the training set.

- namely that $\Delta x_0$ is the value, we need to change the x with before next step.

When convergence is reached and the algorithm stops, a weight vector, **w**, is selected. To avoid over-fitting the training sample (see below), by having too many free parameters included in the parametrization given by a too lavish network architecture, **w** is not necessarily selected from the last step that made the algorithm converge. Instead, after each step, a root-mean-square value is calculated on basis of all of the residuals from the validation sample, and eventually, **w** is selected from the run, t, when this rms was minimum as in figure 3.2 on page 17.

## 3.2 Comittees

Exactly where in weight space the algorithm converges to, is a matter of three things:

---

1. The topology of the weight space.
2. Starting point i.e. initial randomization of weights.
3. The algorithm and it's ability to avoid local minima.

---

If pt. 3 is very high, i.e. the ability of the algorithm to avoid local minima is, it should be good at converging to the global minimum, which make the first two points less important, although they still govern the quality the algorith should have. By running the same proces again with differently initialized random weights, you would end at the global minimum yet again. // If the algorith is not good enough with respect to the topology, you could by chance localize the global minimum after the first run. But generaly you won't be certain that the downslope iteration has avoided getting stuck in a local minimum, which will return unnessesary erroneous results. Now, obviously the result of getting stuck in a local minimum, could happen again, if you tried from a new starting point, and maybe you wouldn't even be able to differentiate significantly between results coming from localization of global minimum (should this actually be found) from local minima. The way around this problem is to incorporate all of these possible misshaps into a so-called *committee* of networks. Using the average of the results as the estimator instead of either a random one, or the seemingly best of them, not only gives a better result on average, than you would expect from any single training, but it also opens up the possibility of calculating the *network variance* on basis of the estimations.

## 3.3 Errors.

The quality of the estimations of the outputs, when working on the test sample, are bounded by three things, which are further described below:

---

1. The uncertainties by which the photometric model magnitudes are measured.
2. Quality of training. Network variance.
3. How well the test sample resembles the training sample.

---

1: Uncertainties in the inputs, $\sigma_j$ , propagate via the so-called chain rule through the layers of the neural networks to the resulting estimated output values, giving uncertainties of size:

$$\sigma_i^2 = \sum_{j=1}^{N} \left(\frac{\delta y}{\delta x_j}\right)^2 \sigma_j^2, \tag{3.8}$$

where the sum is over the N bypass filters (and other inputs if chosen). Obviously it is necessary to select differentiable activation functions to ensure the practical matter of actually performing the calculations of the uncertainties of the outputs as part of the algorithm. The logistic sigmoid as writen in 3.3 on page 15, is easily differentiable:

$$\begin{aligned} \frac{\mathrm{d}g(u)}{\mathrm{d}u} &= \frac{\mathrm{d}}{\mathrm{d}u}\left(1+e^{-u}\right)^{-1} && (3.9)\\ &= (-1)(1+e^{-u})^{-2}(-e^{-u}) && (3.10)\\ &= \frac{1}{1+e^{-u}}\frac{e^{-u}}{1+e^{-u}} && (3.11) \end{aligned}$$

Since:

$$\begin{aligned} 1-g(u) &= 1-\frac{1}{1+e^{-u}} && (3.12)\\ &= \frac{1+e^{-u}}{1+e^{-u}} - \frac{1}{1+e^{-u}} && (3.13)\\ &= \frac{e^{-u}}{1+e^{-u}} && (3.14)\\ & && (3.15) \end{aligned}$$

We finally get:

$$\frac{\mathrm{d}g(u)}{\mathrm{d}u} = g(u)(1 - g(u)) \tag{3.16}$$

2: How well the selected (trained) parametrization actually predicts the results, when working on data that was not used in the training process, is also a matter of how good the algorithm is to avoid getting stuck in a local minimum that has little or nothing to do with the global one. This has both to do with the nature of the algorithm itself, but also of the topology of the weight space. The greater complexity, the worse results if the algorithm isn't stabil enough in the sense of avoiding local minima.
A way to get a feeling for the stability of the algorithm, is by running through the same estimations a couple of times to compare the outputs, where the only difference in the calculations, is another initial randomization of the weights. Chosing the mean of the outputs is on average a better choice than you could expect from a single run, and by calculating the average of the deviations of each of the estimations from the mean, you get the additional benefit of an estimate of the *network variance*, $\sigma_n$:

$$\sigma_n = \frac{1}{N}\sum_{i=1}^{N} \|y_i - \overline{y}\| \tag{3.17}$$

where the N is now the number of runs with different initial randomization of the weights, returning different estimations of output y.
3: This point has to do with the nature of the data. *Interpolating* between points of data in parameter space is a matter of technique, but *extrapolation* outside of this, demands thoughtful caretaking of method and study of the different consequences this step might invoke. The reason to use ANN in fields as astronomy in the first place, is to estimate parameters that is otherwise either too time consuming to measure, or impossible either because of a natural obscurity inherited from the underlying physics, or because of (short term) technical difficulties in getting past a such. When working at the boundaries of what is technically possible (at least for the time being), the training process will always in some sense become a matter of extrapolating beyond the boundaries of parameter space given by the data in the training sample. There is no other way to get around this than either to analyze the possible effects of taking this step, hope for the best, or something in between.
An obvious problem in this context is that as a consequence of the consistent theories of Big Bang and the following expansion of the Universe, the redshift is interpreted as a direct measure of the amount of expansion, the space between an object and the observer has undergone since the light

was emitted. When extrapolating outside parameter space, estimating photometric redshifts for fainter and fainter galaxies then eventually becomes a matter of looking further and further backwards in time to epochs, where the Universe and it's inhabitants morphologically looked different from now. Some care then has to be taken solely on this basis, but it is not my intention to make calculations in this thesis outside parameter space.
The total error of a single estimation of photometric redshift then becomes a combination of the photometric noise and the network variance as above, added in quadrature as:

$$\sigma_p = \sqrt{\sigma_i^2 + \sigma_n^2} \tag{3.18}$$

# Chapter 4

# Testing the existing code

As a starting point, the existing code has to be tested, and along with the code, some example files were conveniently provided, containing data from SDSS early data release. Specifically 12.000 objects in no particular order, had been divided into three files containing 5000 galaxies for training, 1.000 for validation, and 6.000 for testing. The original program package provided by Lahav e.a. consists of 3 main programs and a couple of modules:

---

1. annz_net.f90, a Fortran90 main program creating the network structure after inputs.

2. annz_train.c, a C++ main program that trains an ANN with the structure chosen in annz_net on basis of all data in 2 separate files - a training file and a validation file.

3. annz_test.f90, a Fortran90 main program used for running the contents of a third file - the test data - through the ANN trained by annz_train. It results in an output file containing both spectroscopical (if available) redshifts, the estimated photometric redshifts and errors for each galaxy in the test sample.

4. file_fns.f90, a Fortran90 external module used by annz_test.f90 to read the number of elements contained in each of the input files.

5. annz_ann.f90, a Fortran90 external module used by annz_test.f90 to perform the actual calculations on each test galaxy on basis of the trained net(s).

---

## 4.1 Choosing network architecture

After compilation, using the provided Makefile, the network structure has to be decided. My choice was the obvious 5:10:10:1, since this is what Lahav e.a. used, even though it looks like a rather arbitrary choice probably in the 'it just looks good'-category. By using the same architecture, the results of my improved code will be easier to compare with those of Lahav e.a. However, generally you could run into problems of over-fitting your data with a linear combination having too many parameters (besides the obvious problem of spending way too much valuable computer time, working with oversized and thus slower networks). Over-fitting is not a problem in the training process itself, but might be when you later on apply the result of the training proces to testing data. When allowing to many degrees of freedom to the parametrization, you allow the network not only to fit to the data, but also to the noise inherited from the color magnitude errors. By doing so the strength of the trained network as a means of predicting the result from a test object with respect to the true value diminshes, or even becomes untrustworthy. Figure 4.1 on

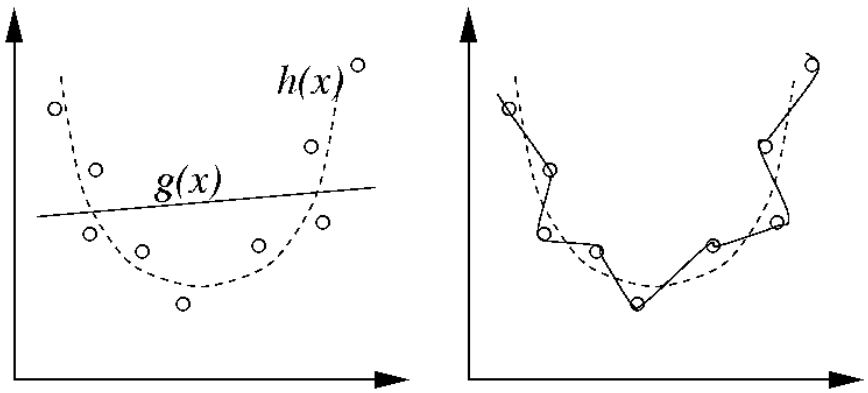



Figure 4.1: An example of overfitting data.

page 23 is an illustration of the general problem [10] of fitting functions to a data set (the dots in the figures), without prior knowledge of the underlying mechanisms that governs them. The function, $g$, is clearly lacking the ability to predict the resulting value at a certain input by any standards. It is indeed the best fit to the data with the degrees of freedom allowed, but as can clearly be seen, a straight line with it's build-in limitations is a very bad choice. The second degree function, $h$, is much more capable of the job.

In fact, the data might as well be governed by a second degree function itself, which true nature is obscured, because of uncertainties inherited from the measuring technique. In fact, $h$ might just be *the* function that governs the data, but at least it's a very good approximation.
The sharply drawn function in the right panel obviously fits the data even better than the function, $h$. However *if* the data were in fact governed by some second order functionality rule, overlapped with noise, as the dotted line illustrates, this function will then be unnessesarily bad at estimating a value for a certain input, only because of the lavish number of parameters, or degrees of freedom. The function now not only fits the data, but also the errors - errors, though, that are supposedly random and uninteresting in this respect.
There are a couple of methods used to get around this problem of selecting the best suited size of network. Lahav e.a. uses the *early stop* method as illustrated in figure 3.2 on page 17, but a more stringent way to minimize the problem would be to analyze what the optimum architecture should be. The process either involves starting with an obviously too small architecture, having too few hidden parameters and sequentially add on more and more nodes, until the overall performance, measured by some sort of root-mean-square value of mean deviation, is no longer improved; or by the reverse method, starting with a way too large architecture, and sequentially removing nodes until the overall performance is lowered. However this process is left as future work and the architecture chosen by Lahav e.a. is used in all of the following, when not otherwise specified.
To make the confusion complete, as of data release six, the SDSS provides photometric z-values based on ANN on basis of a 4:15:15:15:1 network. Interestingly the results are very much the same, as I'll discuss later on, without I've been able to find any documentation for their seemingly lavish choice of architecture.

## 4.2 Testing on example data.

Testing the existing code is then pretty straightforward, using the provided example data, and the ANNz User Guide [2] as follows.
The command:

```
annz_net
```

runs the network structuring program, asks for proper inputs and eventually prepares the actual training of the networks for initialization via inputs:

```
annz_train 'archfile' 'trainfile' 'validfile'
```

does the job and choosing first a *random* number as seed for the random generator, and then '0.0001' as a reasonable weight decay for the quasi-Newton converging algorithm, and 2000 as maximum number of iteration steps, completes the initialization. 2000 steps has shown to be sufficient to ensure the program doesn't stop until converged, but it's possible to save the weights up till after each step if wanted, as can be seen in figure 4.2 on page 35. I think it's an excelent illustration of how the algorithm works it's way through the steps towards convergence. From upper left to lower right even 4 frames is enough to get a feel for the dynamics going on until the system settles. The run has been stopped after 10, 25, 50 and 100 steps, and then the rest is a matter of computer time, before convergence is (hopefully) reached and the final weights can be writen to a file.
These are the last 3 lines of output from each of the three runs. It is worth noting that the number of iterations, before convergence is reached, is very random, depending on the initial randomization of the weights and the topology of the weight space around the starting point.

```
converged
1196 iterations, 2723 epochs
Training set: Error 2.61423  Total 3.02022  RMS (valid) 0.02604

converged
740 iterations, 1709 epochs
Training set: Error 2.61899  Total 3.00625  RMS (valid) 0.03997

converged
1839 iterations, 4099 epochs
Training set: Error 2.56002  Total 2.97803  RMS (valid) 0.03648
```

The comittee consisting of the N resulting weight files is then used as inputs in the testing program, together with the testing file with the command

```
annz_test 'testfile' 'testresultfile' 'wtsfile1' ['wtsfile2' ...'wtsfileN']
```

finally returning the output:

```
 ==================================
  ANNz: Photo-z determination
 ==================================
 Using a committee of 3 networks.
 # of inputs  :  5
 # of outputs :  1
 Sample size : 6000
 Found spectroscopic redshifts.
 Calculating photometric redshifts...
 ================================================
 Output 1: Mean:  0.00006  rms:  0.02588
 Photometric redshifts saved to ../EXAMPLE/sdss.ugriz.testresult
 ================================================
```

The program finishes off as seen above, by calculating the mean -and root-mean-square deviation, rms, between the results from the network and the spectroscopical redshifts of the members in the test sample. The mean is given by the average residual between the spectroscopical and photometric redshifts as:

$$Mean = \frac{1}{N}\sum_{i=1}^{N}(z_{spec,i} - z_{phot,i}), \tag{4.1}$$

where the sum is over the N galaxies used in each of the methods, and $z_{spec,i}$ and $z_{phot,i}$ are the spectroscopical and photometric redshifts for the i'th galaxy respectively. A positive mean, means that on average, the estimated values of the photometric redshifts are too high with respect to the measured spectroscopical ones.
The rms values is what ultimately expresses the goodness of the method in a comparable way to other methods working in the same field (on the same data if one should be strict):

$$rms = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(z_{spec,i} - z_{phot,i})^2}, \tag{4.2}$$

In Appendix A.1, I've added an IDL program, I've made, named plot_zvz.pro that handles the results and allows you to create certain illustrative plots as figure 4.2 on page 36. The upper panel is an illustration of the result of the test run on the provided example data. All the points

represent a galaxy, whose redshift has been calculated both on basis of spectroscopy (x-axis) and with ANN on basis of photometry (y-axis). On the plot is also shown the angular bisector that is the optimum line, where all data points should lie, if the photometric redshift estimations were in absolute accordance with the spectroscopical ones - something we already know from the output of the program, they're not. Output from the IDL program also reveals the exact same values of mean and rms as above. In the following, whenever a plot is presented, where it's not emphasized how it's done, It'll have been made with the abovementhioned program or one somewhat similar to. So the program 'works' with the provided data. Neural networks are trained and the outputs are quantitatively as expected. Figure 4.2 on page 37 is a demonstration of what was briefly noted in chapter 3.3, *Errors*, that the average estimation of an output based on different trained ANNs, a so-called *committee*, is generally a better estimator than any single one of them. However in this occation, the effect is not that obvious. In fact the first set of weights actually returns a slighty lower rms (approximately 0,5%)than the three of them together. The two others return rms values around 4% higher than the combined weights. What this tells us though is that the algorith is quite stable at grinding towards similar results. And besides that, using a committee does give you an estimate of the network variance that is used to calculate the total error as given by equation 3.18 on page 21.

## 4.3 Errors

The errors connected to the estimated photometric redshifts consists of two parts, the noise inherited from the color magnitude errors, and the network variance, calculated as a mean deviation of the different photometric estimations that results from different initial randomization of the weights. The two parts are added in quadrature as in equation 3.18 on page 21 and are automatically writen to each line of the output file.
The calculated errors are shown for a subsample of the above testing set in the lower panel of figure 4.2 on page 36, where every 25th data point were selected to improve the general view.
Next step is to test on controlled data extracted directly from a database...

## 4.4 Gathering data from SDSS.

The Sloan Digital Sky Survey [1] is a still undergoing project aiming to gather optical images of about one forth of the sky with the 120 megapixel camera mounted on the dedicated 2,5m telescope, and along the way, furthermore gather spectroscopical measurements of the observed objects. Accessing the huge database is overwhelming at first, but soon becomes rather easy. Some care has to be taken at all times though, in order to extract exactly the right data.

[1] located at www.sdss.org

After a lot of initial testing, I finally ended with extracting the following data for each galaxy from the spectroscopy query form part of the data access [2], thereby ensuring that all objects had already undergone spectroscopy:

---

1. Limit number of output rows (0 for unlimited) to: '0'
2. Output Format: 'CSV'
   Parameters to return:
3. Spectroscopy: bestObjID, z, zConf, zErr, zStatus, zWarning [3]
4. Imaging: model_mags, model_magerrs
5. Position Constraints: None
6. Spectroscopy Constraints: Galaxy
7. Imaging Constraints [4]:
   - Model magnitude $0 < u < 20$
   - Model magnitude $u > 20$
8. Obj type: Extended Sources (e.g., Galaxies)

---

What results is 2 files containing model colors with corresponding uncertainties and spectroscopical values for in total 667609 different galaxies. This raw catalogue is then our starting point.

[2] http://cas.sdss.org/astrodr6/en/tools/search/SQS.asp

[3] The bestObjID was included to ensure later identification of the object

[4] This part had to be split in 2 since the database only allows you to directly extract 500.000 lines of data at a time. An arbitrary cut was made at magnitude 20 in the u-band, where a second border was made in the bright part below 20 that it also at least has to have a magnitude of 0 to avoid extracting very bad data points with -9999 as entries in some or all of the magnitudes. Afterwards the data were united in one file.

## 4.5 Filtering files for easy use in existing code.

The extracted data from the SDSS database come in an unusable format with respect to direct use in the open source code provided by Lahav e.a. A Fortran90 coded program, 'filter.f90' was created in order to prepare these data for the task at hand, and along the way some details were added, giving the user control of which types of data is to be included in the outputfiles, and how big each of those should be relative to each other. Most of the downloaded data have undergone spectroscopy, but not all of this was done with the same precise methods, which is marked with the above mentioned zStatus spec data value, ranging from 0 to 12 according to the method used. I decided to filter out all the galaxies not having the zStatus spec equal to 4 [5], and finally I chose to make the training file 10 times larger as both the validation -and the testfile. The 667609 abovementioned lines of data are then cut down to 409308 all marked with the zStatus=4 spec (61,31%). These are then subsequently divided into the 3 needed files at random.
By taking this step, it is ensured that the training is only performed on the best suited galaxies, where the spectroscopical measurements have the lowest possible associated uncertainties. However, what is lost in this way, is instead possible more interesting data from fainter objects further away, with slightly higher uncertainties, bases on another type of measurements. These data might especially be more interesting when working with *real* data without the spectroscopical redshifts as guideline, since they asumably are closer on average to each other in time, thus asumably resembling each other closer morphologically. I'll make another notice about this in the later chapter concerning 'Future work', but for now, I'll settle for the original choice, I made.
The Fortran90 code is shown in Appendix A.2 and includes comments and a short intro for the user.

[5] Redshift determined from x-corr with high confidence

The run results in a total of 5 output files, where the numbers in parenthesis is the number of lines of data each representing one galaxy:

- Training file (368377)
- Validation file (20465)
- Testing file (20465)
- Statistics file
- Numbering file for easy reference to SDSS (20465)

The *Statistics file* as shown in Appendix B.1 contains information on how many galaxies in the raw catalogue were marked by each of the different zStatus specs, and the *Numbering file* contains information on which galaxy in the SDSS database actually is connected to each line of data in the test sample. This part was done for easy reference if it one day should be needed to dig further into why a particular galaxy is extraordinarily difficult to train an usable neural network for.

## 4.6 Recreating the results

By following the same steps with the filtered self-extracted data as above in chapter 4 on page 22 with the provided test data, the following output resulted:

```
annz_test sdss6.test sdss6.ANNz.result sdss6_1.wts sdss6_2.wts sdss6_3.wts sdss6_4.wts
=====================================
  ANNz: Photo-z determination
 ==================================
 Using a committee of 5 networks.
 # of inputs  :  5
 # of outputs :  1
 Sample size : 20465
 Found spectroscopic redshifts.
 Calculating photometric redshifts...
 ================================================
 Output 1: Mean:  0.00033  rms:  0.04016
 Photometric redshifts saved to sdss6.ANNz.result
 ================================================
```

Note that I decided to go for 5 nets both here and later on when using the improved code.
The rms is significantly higher (approximately 25%) than the same proces run on last years updated SDSS5 catalogue containing a total of 557299 galaxies, 339729 of those marked with the zStatus=4 spec (60,96%), divided into the three files as 283109, 28310 and 28310 - the rms then was 0.03120 with a mean of 0.00012 calculated on basis of a comittee of three members. Why this is so, is rather unclear. It is also unclear why both these rms values are significantly higher than the one Lahav e.a. originally published as 0,0236.
As can be seen in 4.6 on page 39 that illustrates the errors relative to the magnitudes in each filter, sorted on basis of the values in filter r, the quantitative feel of the data are alike on the example data (leftmost frames) and the SDSS6 data (rightmost frames) respectively. However on closer examination these mean errors relative to the magnitudes, given by:

$$Mean = \frac{1}{N}\sum_{i=1}^{N}\frac{\sigma_i}{M_i}, \tag{4.3}$$

where the sum is over the i galaxies with magnitude M and associated error $\sigma$, are indeed greater in the SDSS6 data than in the example data that

according to Lahav e.a. originated form the early data release of SDSS dating back to june 2001. In fact the mean relative error is increased by 38,61%, 23,05%, 7,32%, 4,50% and 2,75% in the five filters u, g, r, i and z from the example data to the SDSS6, which might account for the increased rms.
As noted in the chapter about choice of network architecture, as of data release six of june 2007, the SDSS provides photometric redshifts estimated on basis of ANN with a lavish 4:15:15:15:1 architecture [6] The main result is 4.6 on page 40 [7] that are based on data reaching magnitudes in the r filter below $r > 20$ as mine do. A rms value of 0.045 is presented. A point has to be emphasized here though. What is most interesting in the following is not the resulting rms values themselves, but the relative improvement, when using an improved method on the exact same samples of data. Neither the example data or the data that has been run through the 4:15:15:15:1 ANN are the same, which makes direct comparison extremely difficult.

[6] In fact SDSS also provides photo-z values estimated with a template fitting technique, but that is a totally different story.

[7] Taken from http://yummy.uchicago.edu/SDSS/

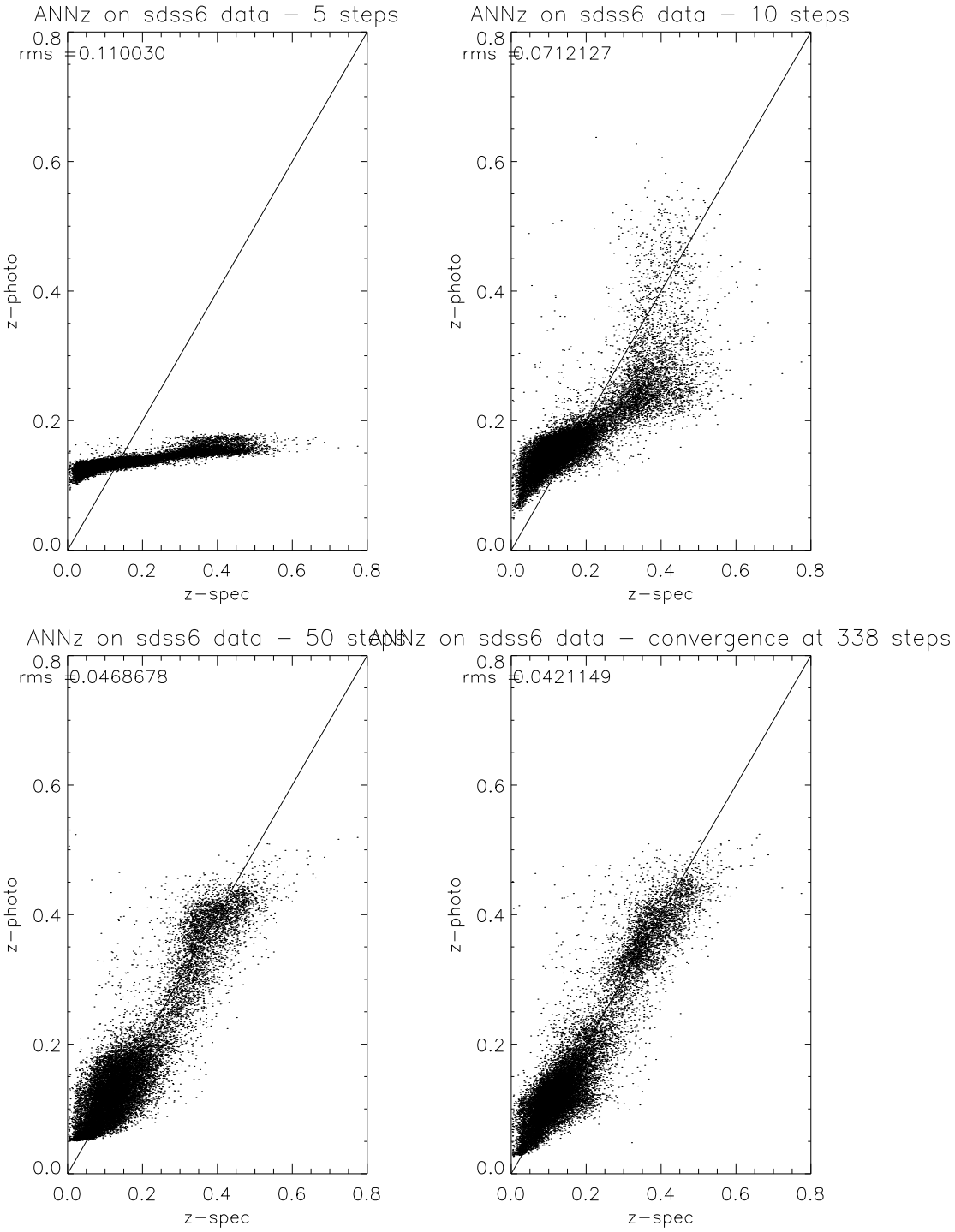



Figure 4.2: Photometric vs. spectroscopic redshift for provided data example. Illustration of progress towards convergence

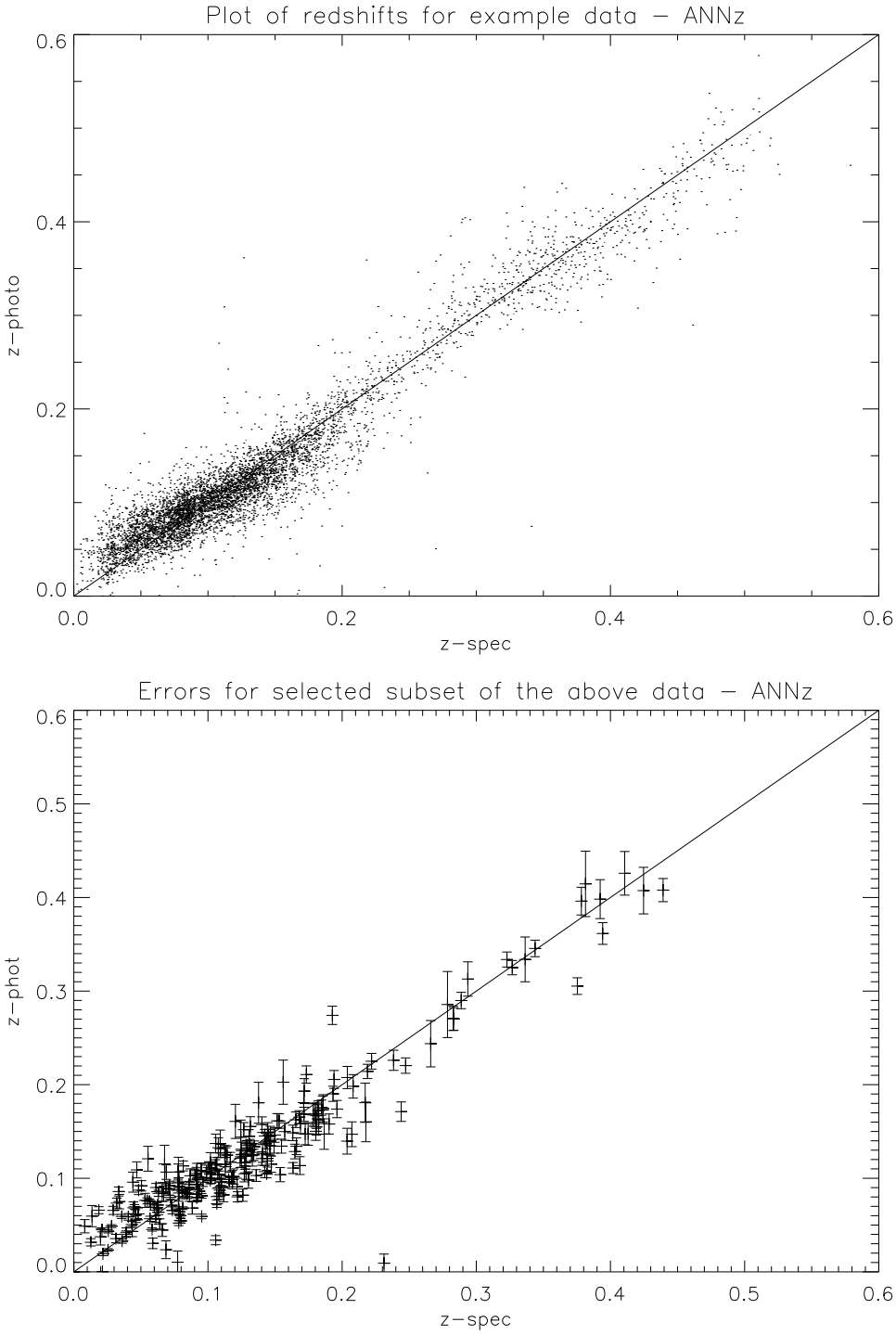



Figure 4.3: Photometric vs. spectroscopic redshift for provided data example. Upper panel: Normal plot. Lower panel: Error plot for selected subset.

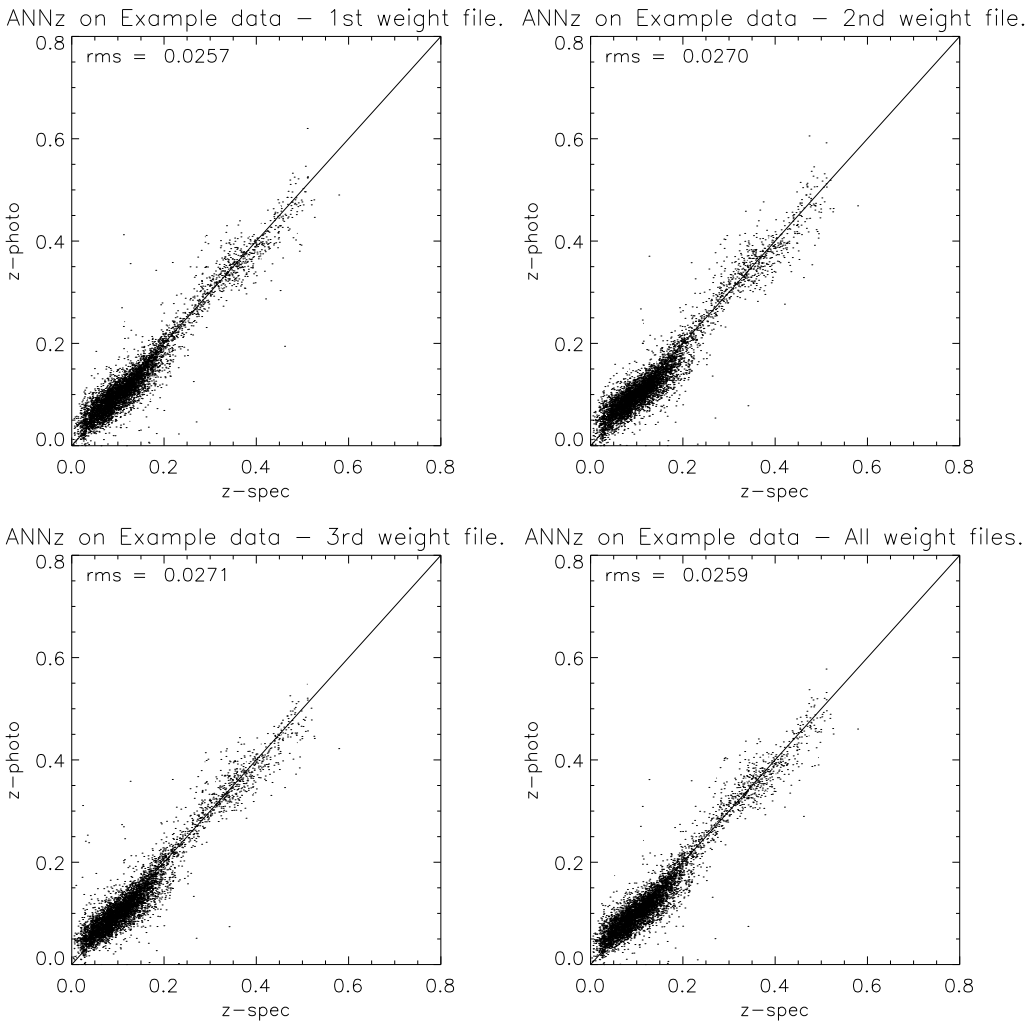



Figure 4.4: Committee effect demonstrated on example test data. Lower right frame is the results based on a combination of the other three frames.

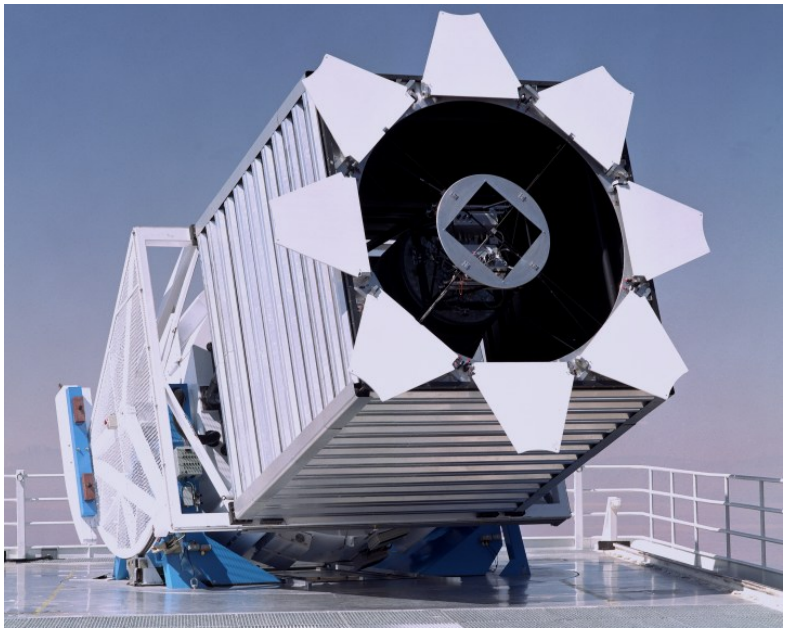


Figure 4.5: The SDSS 2.5-meter telescope located in Apache Point, New Mexico, USA.

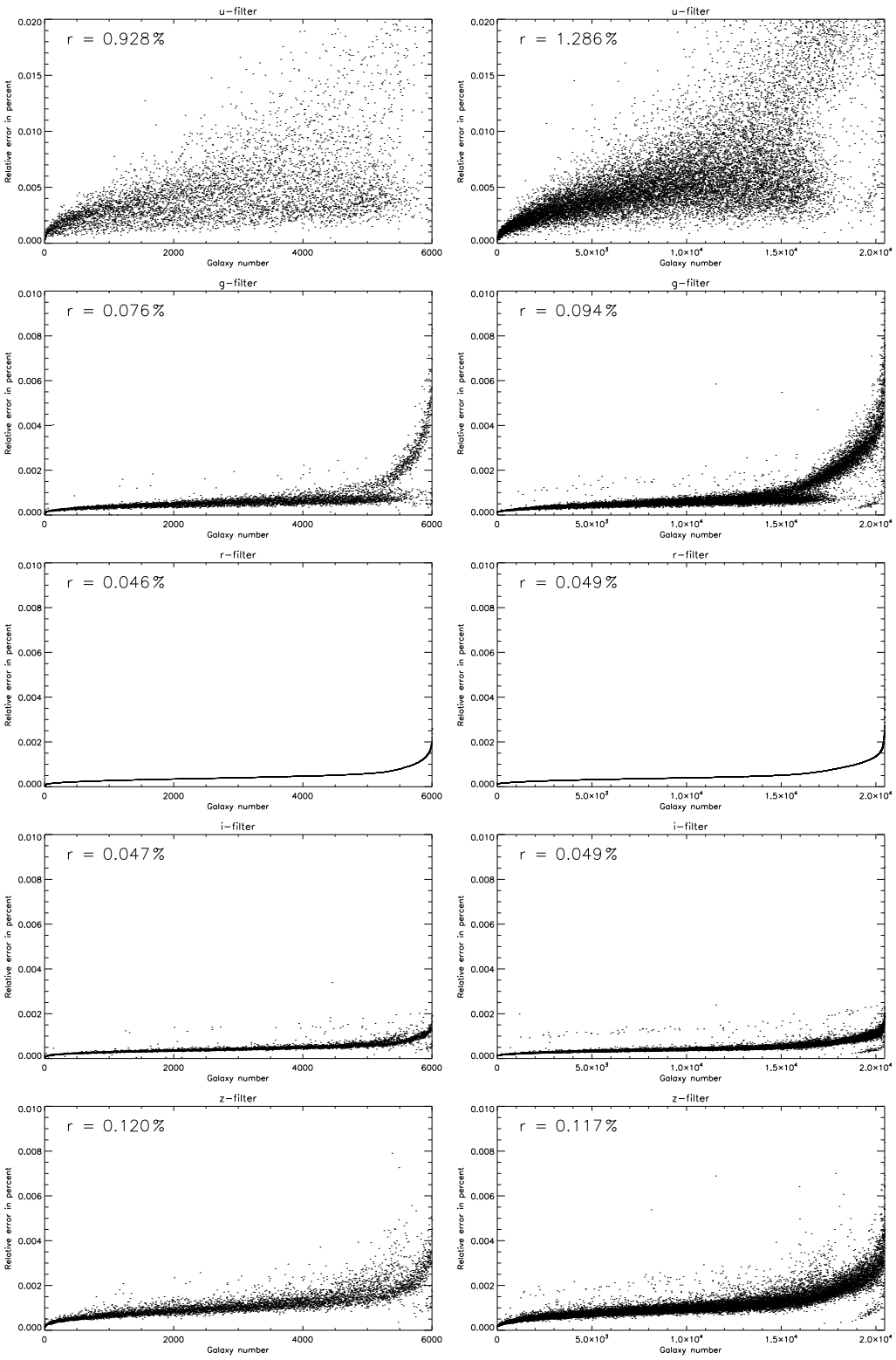



Figure 4.6: Errors relative to model magnitude values given in percent for each of the five filters u, g, r, i and z. Left column: Example data. Right column: SDSS6 data. Both data sets are sorted after the relativ error in the r-filter.

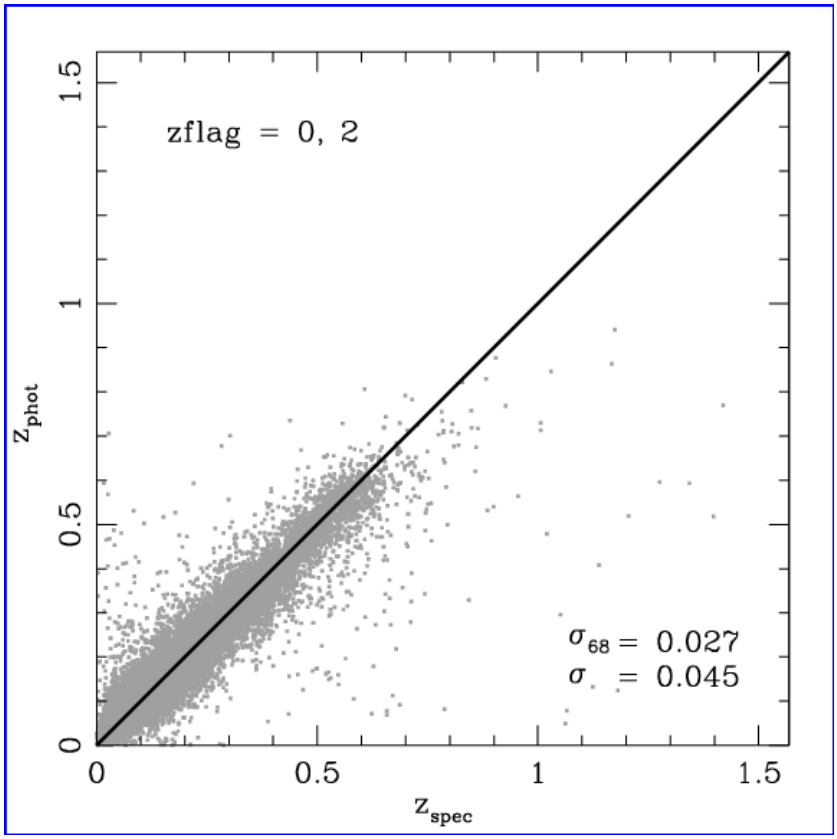



Figure 4.7: Photometric redshifts presented by SDSS on basis of a ANN with 4:15:15:15:1 architecture.

# Chapter 5

# Modification of the code - nANNz

As noted in the introductory history section, another well known method of determining photometric redshift is the now slightly outdated method involving template fitting, where you adopt the redshift of the galaxy showing most similarities with the one you want to determine the redshift for - similarities basically being galaxy type and colorspace distances given by the chi-squared method in equation 2.5 on page 10.
My thought was then to combine this rather outdated approach with the ANN code, by automating a selection process, selecting a subsample from the training samle, closest resembling the galaxy at hand, thereby training a new set of artificial neural networks for each galaxy in the testing sample, creating a set of weights that are unique to the test galaxy in focus, thus hopefully effectively leaving out unnessesary information in the training process concerning other types of galaxies.
The point is that when you train a couple of networks on all the possible different types of galaxies, ranging form the early type elliptical ones to the late type spiral shaped as seen in figure 2.2 on page 10, together with the rather diffuse looking, and unclassifiable irregulars, you'll find that it *IS* in fact possible to make the training process converge to a solution that solves the problem at hand. However, what you get is a set of values for your weights that are the basis of the unphysical parametrization, we initially arbitrarily set up. By incorporating a wide variety of galaxies in the training proces, we can safely apply the solution to another wide variety of test galaxies, but the results on overall should be worse, than if we started with a different set of values of the weights for at least each of the different types of galaxies.
The selection mechanism, I've made, lets you choose how many galaxies

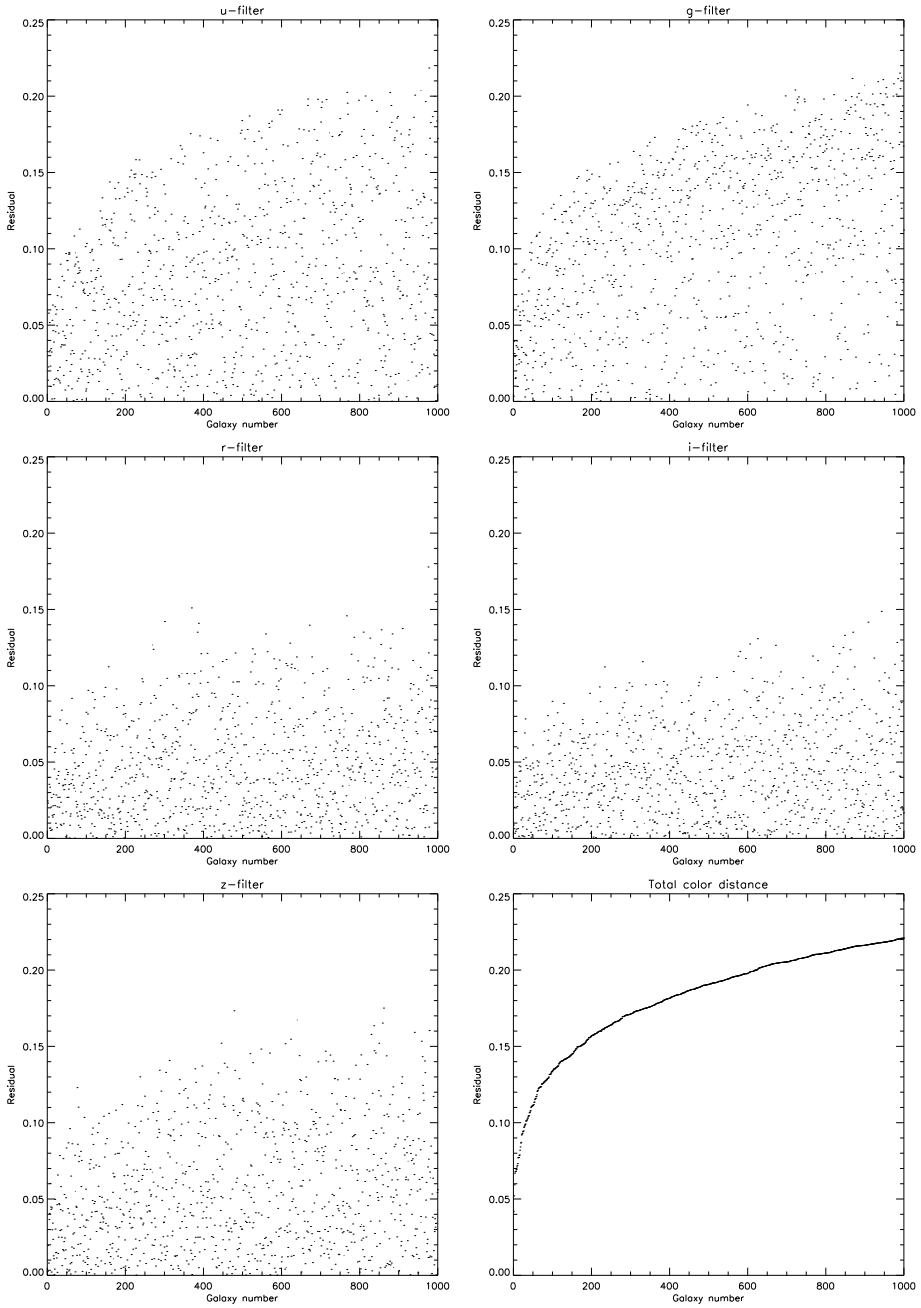



Figure 5.1: Example of how the selection mechanism prior to training works. Color space distances from selected members of training sample to Gal # 18585 in each filter and total.

should be used for training of each galaxy in the testing sample. As of yet, there is no way to let the program select the optimum number for each galaxy but obviously this number should depend on how special the test galaxy at hand is, compared to the different members of the training sample - there is no reason to include training on 1000 galaxies if only the 50 closest to it in color-space objectively resembles it morphologically.

## 5.1 nearby Artificial Neural Network z-phot.

The resulting modified code consists of a new Fortran90 main program, nANNz.f90, that calls each of the original three (only slightly modified) main programs as external routines, and along the way, a selection mechanism is applied to each test galaxy, and valuable data are writen to different files for later use and/or examination. The selection mechanism is limited to color-space distances only, so that the G galaxies in the training sample that has minimal values of $D_{lm}$ as given by:

$$D_{lm} = \sqrt{\sum_{i=1}^{5} \left( M_{l,i} - M_{m,i} \right)^2} \tag{5.1}$$

are selected for training, where M is the magnitude of filter i for respectively training (l) -or testing (m) galaxy. The rest of the galaxies in the training sample are left out of the current training, but will be included in following calculations if they are close enough in color-space to another of the galaxies in the testing sample. For the sake of future improvement all of the color distance data related to each galaxy in the test sample are writen to a file, and the distributions of the residues for each filter as well as the total, for each galaxy can then be printed both to screen and PostScript format. Figure 5 on page 42 is an illustration of how the selection is done. I've singled out the galaxy in the test sample that returns the best (i.e. smallest) residual and used that as example, but it could have been anyone. All compilation is done with the command 'Make' refering to the updated Makefile as seen in appendix A.3. After due editing of the fortran90 file 'nANNz.f90' and run of the network structuring program in point 1 below, all calculations will proceed with no further ado than the command 'main.x'.

The structure of the nANNz-code can be listed as:

---

1. An initial choice of network architecture is made by running the original annz_net.f90 program.
2. Choice of how many differently initialized networks, N, should be trained for each testing galaxy [1]
3. The program nANNz.f90 is run.
4. One line of data at a time is read from the test file containing photometric values with corresponding uncertainties, and if possible also the spectroscopical redshift.
5. The selection mechanism is applied to the training sample, selecting those G galaxies closest to the current test galaxy in color-space. [2]
6. The relevant data for the selected galaxies are writen to a temporary file.
7. The annz_train.c program is called, training N different networks on basis of the same list of galaxies, but with different initially randomized weights. Each of the networks are writen to temporary files. [3]
8. The annz_test.f90 program is called which calculates the final photometric redshift for the current test galaxy and appends various results to the output files.

---

[1] Less than 3 is not wise, since this makes calculation of the internal uncertainty for each calculated distance impossible. 9 however is the maximum choice with the code as it stands.

[2] G is set prior to run and has been chosen as G=1000, but this is just another choice in the 'it just looks good'-department and logically an optimum number should exist for each galaxy type on basis of overall resemblence to other members of the training sample.

[3] Thanks to Christian Vinter for help with making the coupling between the Fortran -and C+ code parts.

The resulting output files are:

1. Results file
2. Color space distance file
3. Network variance file

The results file contain the resulting outputs from the run along with the spectroscopic redshift if available, and the combined error. The color space distance file contain information of which members of the training sample were used for training the networks for each test galaxy, and the associated color space distances. The network variance file contain the different outputs from each run that is the basis for the mean used as the final photometric redshift estimator.

# Chapter 6

# Results.

Figure 6 on page 47 is an illustration of the main result of this thesis, as it clearly shows the improvement, the modified code, nANNz, returns. Common for each of the two graphs are that each point represents the z-value for a galaxy in the test sample obtained both by spectroscopy (x-axis) and ANN on basis of photometry (y-axis). The anglular bisector illustrates the virtual line that all the points would lie on if the ANN based on the photometric inputs returned the actual results given by spectroscopy. The same galaxies have been used in both calculations and the only difference is whether the original code provided by Lahav e.a. has been used as in the topmost graph (ANNz), or if the modified code has, as in the bottommost graph (nAANz).
Two important aspects are worth emphasizing right here: The first is the obvious feature that the points overall are closer to the line using the modified code as can be seen with the naked eye. The average improvement measured as a lowering of averaged residues between the two z-values are as high as 39,60% given by:

$$P = \frac{rms_{ANNz} - rms_{nANNz}}{rms_{ANNz}} * 100\%, \tag{6.1}$$

where the rms-values are calculated on basis of the results of the two different methods given by equation 4.2 on page 27.
The other important aspect is that there is a significant decrease in 'drifters' in the results from nANNz - points that are either largely over -or undershot, and thus 'very far' from the optimum line. However there are still way too many points being flawed in this way, as to fully trust the output based on photometry, where no spectroscopical redshift can be used as validation. The prospects of eliminating these further, is discussed in the later chapter named 'Future work'.

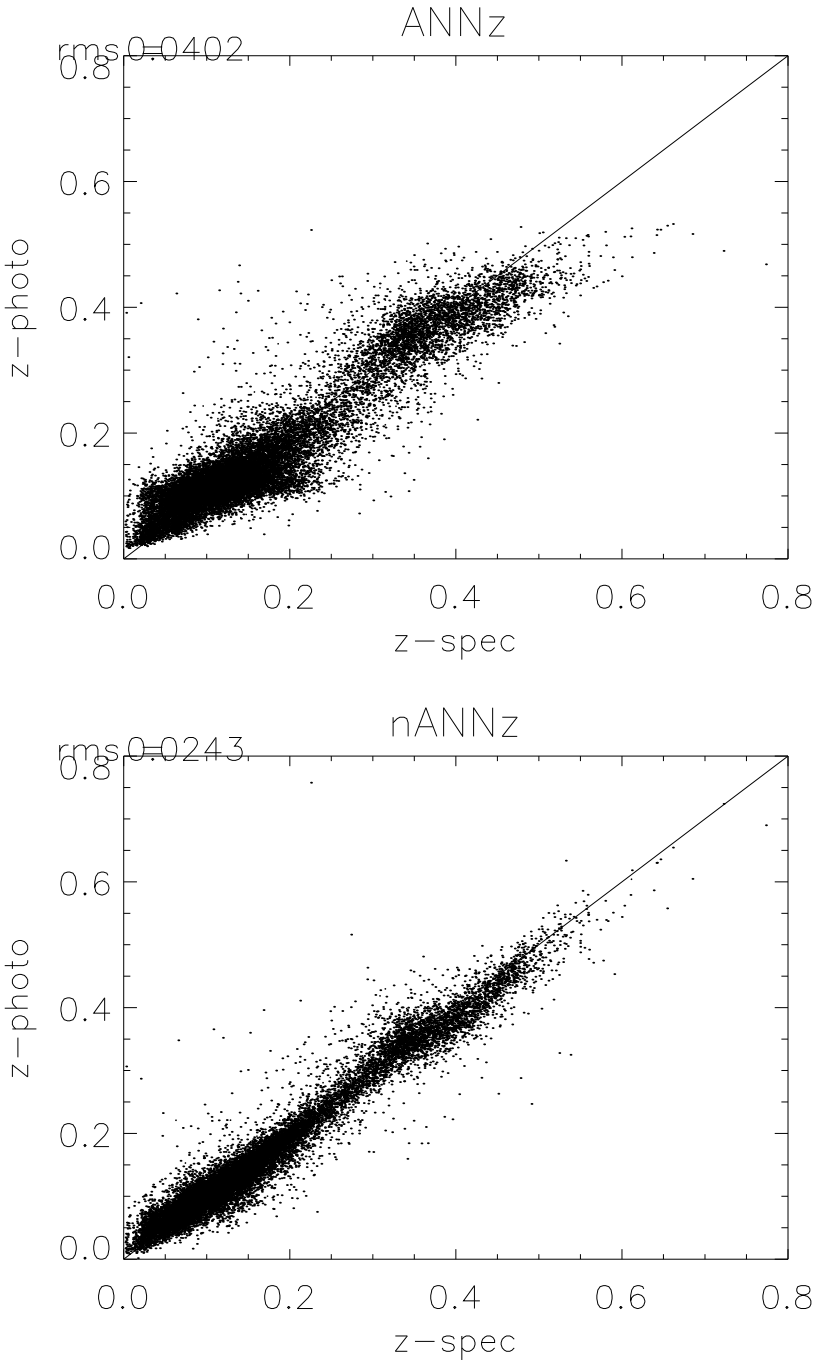



Figure 6.1: Photometric vs. spectroscopic redshift for SDSS6 data. Upper panel: ANNz. Lower panel: nANNz

Thirdly, a couple of noticable unfortunate features in the ANNz-results are almost entirely eliminated in the nANNz-results. In the ranges from $z = 0 - 0.1$ and $z = 0.3 - 0.4$ redshifts are generally overestimated, whereas they are underestimated in the ranges from approximately $z = 0.1 - 0.25$ and above $z = 0.4$. From $z = 0.5$ there's even a very unfortunate dropoff in the ANNz-results that is almost, but not entirely eliminated in the nANNz-results.

## 6.1 Re-accessing SDSS database with bestObjID.

By means of the numbering file and the bestObjID spec, it is easy to re-access the SDSS database to check selected objects to get a feel for why certain types seem to gain especially bad estimation of their redshifts. The two extreme galaxies which photometric redshifts are respectively the best and worst estimated with respect to their spectroscopic values, can be singled out of the sample to serve as a good exercise in working with the SDSS database.
As it happens, they're located at lines number 14335 (minimum residual) and 18585 (maximum residual). The color space distributions of the 1000 galaxies that are selected for training in each of the two cases can be compared by looking at Figure 5 on page 42 and Figure 6.1 on page 52, where it is already seen that it's not particularly surprising if redshift estimation goes much better for galaxy number 14335 than 18585 as the average color-space distance from the test object to the 1000 selected galaxies are close to 0.15 in the first example, and just below 3 in the last. In other words the 1000 selected galaxies in the last example doesn't really resemble the test galaxy - and the 1000 selected are those *closest* to it in color-space out of all of the 368377 that are part of the training file! In the test numbering file it can now be found that these numbers corresponds to the SDSS objects with ID numbers given by: 587736585505931698 and 588017109679865928 respectively, and the original images together with the properties of the objects can then be found at:

`http://cas.sdss.org/dr5/en/tools/explore/obj.asp?id=587736585505931698`

`http://cas.sdss.org/dr5/en/tools/explore/obj.asp?id=588017109679865928`

The upper parts of figure 6.1 on page 50 and Figure 6.1 on page 51 are screendumps of the webpages containing the relevant data of the objects. The actual recorded spectrums can be seen in greater detail by clicking at it below the listed data. An easy check to see if we have indeed located the right object in the database, is to check the spectroscopical redshift value, z, found next to the corresponding error, zErr. It is seen to be 0.199 for the first galaxy, which corresponds to the value of line 14335 above. I've marked the data, we already know about the galaxy and also the link in the margin that brings you to a page listing the photo-z data that SDSS provides as of data release six. The topmost of the PhotoZ data is in the lower parts of the figures. For the first galaxy, it is seen that the photo-z

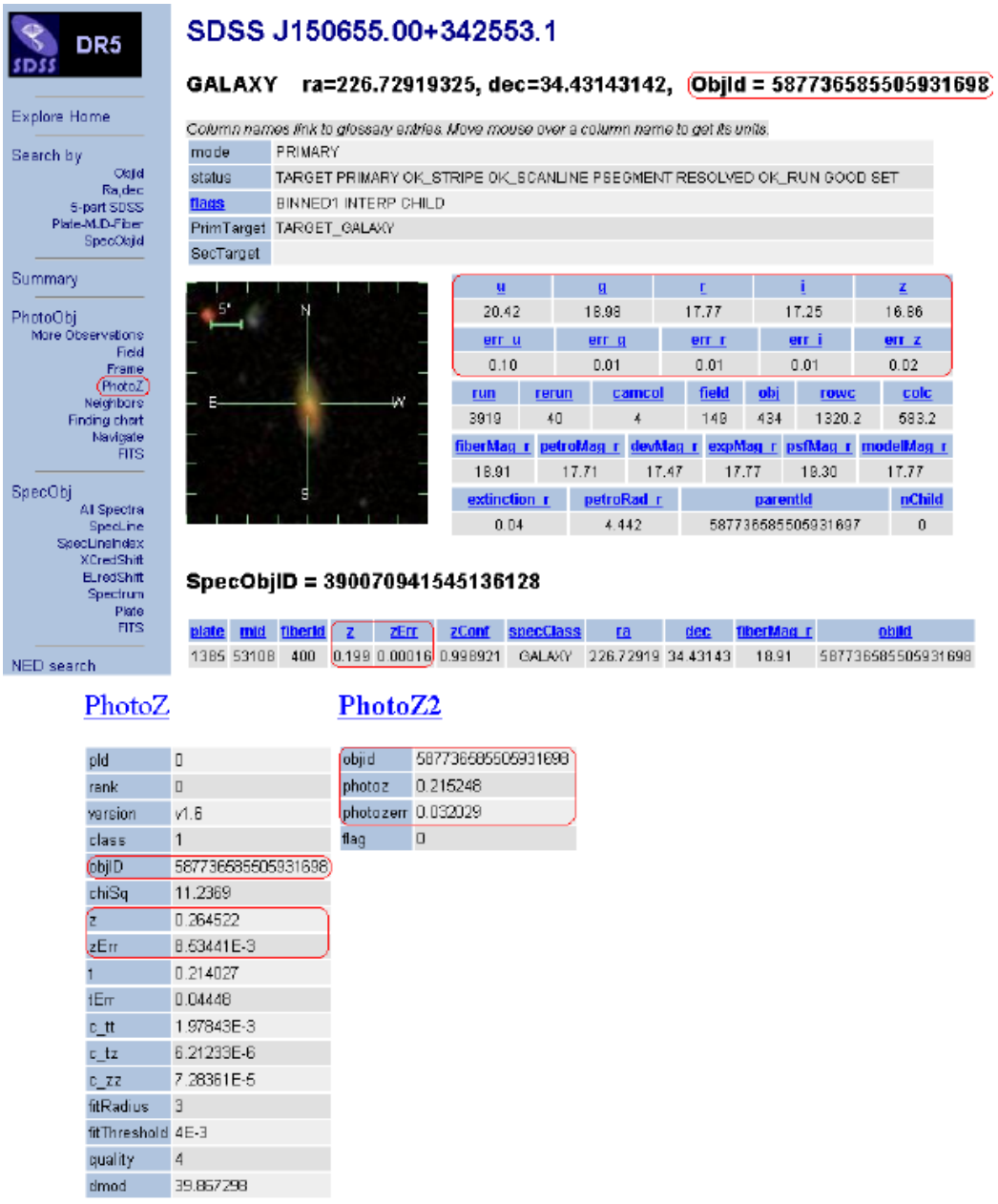
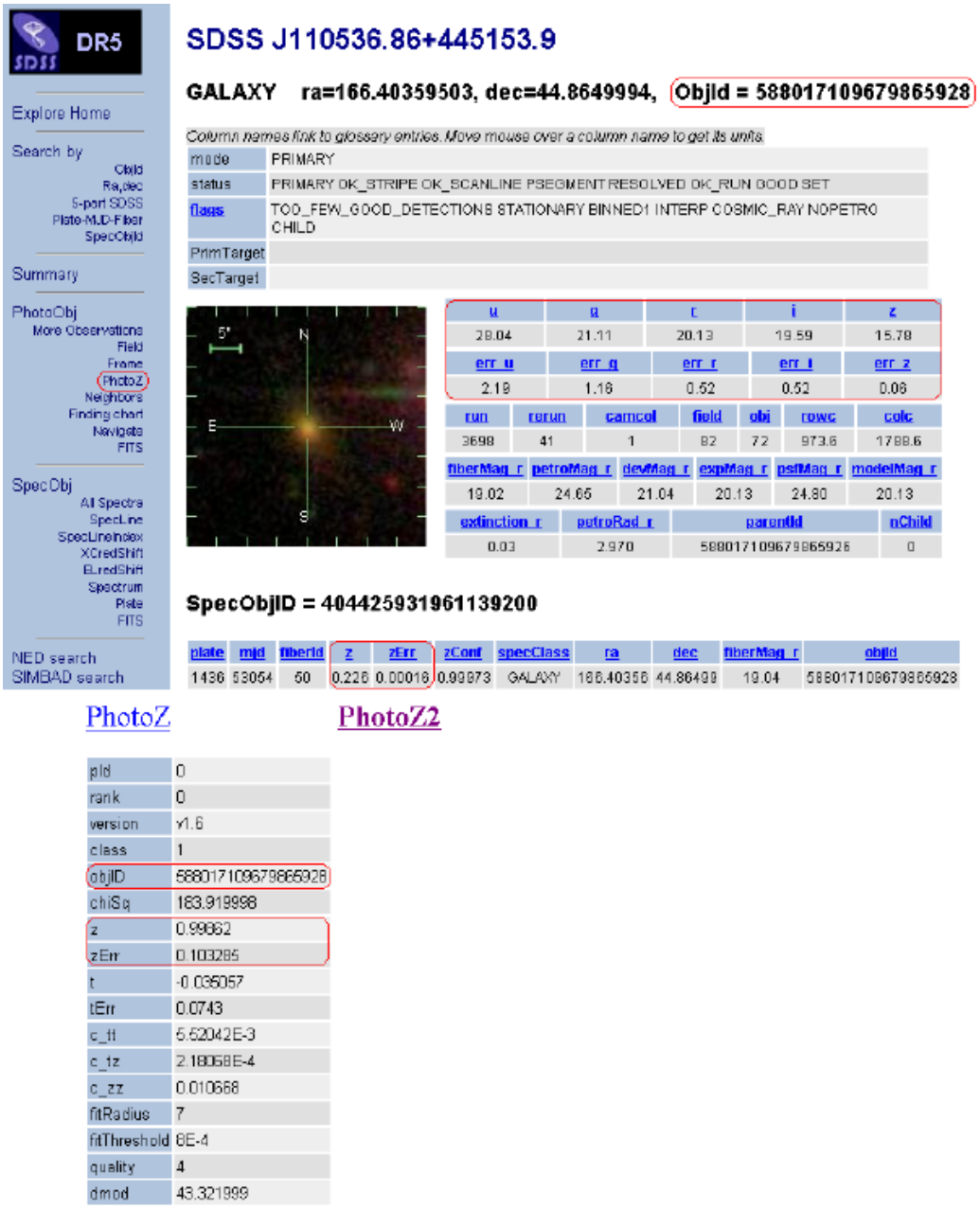



Figure 6.2: Example of Re-accessing SDSS database with bestObjID. - selected galaxy # 14335 with lowest residual. Below is the PhotoZ data from SDSS.



DR5

Explore Home

Search by
Objid
Ra,dec
5-part SDSS
Plate-MJD-Fiber
SpecObjId

Summary

PhotoObj
More Observations
Field
Frame
PhotoZ
Neighbors
Finding chart
Navigate
FITS

SpecObj
All Spectra
SpecLine
SpecLineIndex
XCredShift
ELredShift
Spectrum
Plate
FITS

NED search
SIMBAD search

**SDSS J110536.86+445153.9**

**GALAXY ra=166.40359503, dec=44.8649994, ObjId = 588017109679865928**

*Column names link to glossary entries. Move mouse over a column name to get its units.*

| | |
|---|---|
| mode | PRIMARY |
| status | PRIMARY OK_STRIPE OK_SCANLINE PSEGMENT RESOLVED OK_RUN GOOD SET |
| flags | TOO_FEW_GOOD_DETECTIONS STATIONARY BINNED1 INTERP COSMIC_RAY NOPETRO CHILD |
| PrimTarget | |
| SecTarget | |




| u | g | r | i | z |
|---|---|---|---|---|
| 28.04 | 21.11 | 20.13 | 19.59 | 15.78 |
| **err_u** | **err_g** | **err_r** | **err_i** | **err_z** |
| 2.19 | 1.16 | 0.52 | 0.52 | 0.06 |

| run | rerun | camcol | field | obj | rowc | colc |
|---|---|---|---|---|---|---|
| 3698 | 41 | 1 | 82 | 72 | 973.6 | 1788.6 |

| fiberMag_r | petroMag_r | devMag_r | expMag_r | psfMag_r | modelMag_r |
|---|---|---|---|---|---|
| 19.02 | 24.65 | 21.04 | 20.13 | 24.80 | 20.13 |

| extinction_r | petroRad_r | parentId | nChild |
|---|---|---|---|
| 0.03 | 2.970 | 588017109679865926 | 0 |

**SpecObjID = 404425931961139200**

| plate | mjd | fiberId | z | zErr | zConf | specClass | ra | dec | fiberMag_r | objId |
|---|---|---|---|---|---|---|---|---|---|---|
| 1436 | 53054 | 50 | 0.226 | 0.00016 | 0.99873 | GALAXY | 166.40356 | 44.86499 | 19.04 | 588017109679865928 |

PhotoZ PhotoZ2

| | |
|---|---|
| pid | 0 |
| rank | 0 |
| version | v1.6 |
| class | 1 |
| objID | 588017109679865928 |
| chiSq | 183.919998 |
| z | 0.99982 |
| zErr | 0.103285 |
| t | -0.035057 |
| tErr | 0.0743 |
| c_ti | 5.52042E-3 |
| c_tz | 2.18058E-4 |
| c_zz | 0.010668 |
| fitRadius | 7 |
| fitThreshold | 8E-4 |
| quality | 4 |
| dmod | 43.321999 |

Figure 6.3: Example of Re-accessing SDSS database with bestObjID. - selected galaxy # 18585 with highest residual. Below is the PhotoZ data from SDSS.

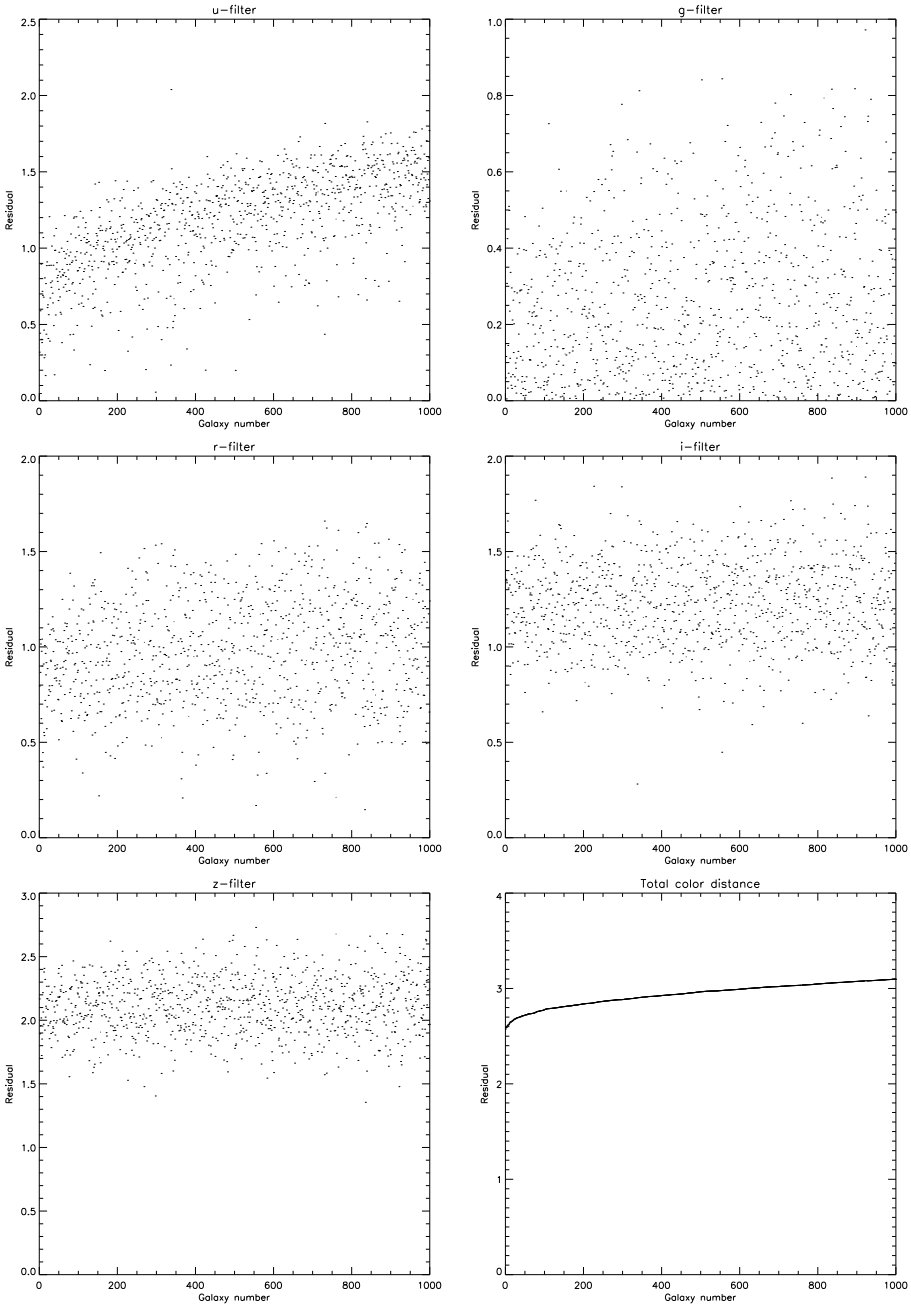



Figure 6.4: Color space distances from selected members of training sample to Gal # 18585

values estimated on basis of a template fitting technique (0.264522) and ANN (0.215248) respectively, show some resemblance to the nANNz estimated value (0.199211) and that they're all pretty close to the spectroscopical value of 0.199212. For the second galaxy, it is seen that the photo-z value estimated on basis of a template fitting technique (0.99862) show some resemblance to the nANNz estimated value (0.757629) and that they're both pretty far from the spectroscopical value of 0.225593. No photoZ on basis of ANN is listed in the SDSS for that particular galaxy.

## 6.2 Additional calculations.

Figure 6.2 on page 54 shows the calculated error bars for a random subset of the full test sample. Nothing much to say about that really, as it should be self-explanatory. Even though the error bars on average are smaller in the lower frame, they're more often striked through by the angular bisector, which tells that the spectroscopical z-value is estimated within the uncertainty inherited from the model magnitudes (and network varianse too for that matter).
Since the number of galaxies selected on basis of color-space distances in each run, was initially arbitrarily selected as 1000, primarily due to the overall wastness of data, a swift test was made to see whether this had been chosen unnecessarily high, making the computer runs lenghtier than necessary. Figure 6.2 on page 55 is made on the same basis as the lower frame of figure 6 on page 47, but instead of selecting 1000 galaxies for each training, 50 in the upper frame and 100 in the lower was chosen.
The rms for the three runs were thus $0,0243$, $0,263$ and $0,0260$ as seen, delivering the conclusion that out of the three arbitrarily chosen numbers, 1000 worked out the best, followed by 100. It's not easy, if not impossible, to conclude anything more on that matter. And probably the answer is much more complicated than just working our way into the best suitable number, as there's no a logic in this should exist. In fact the best suitable number of galaxies selected for training might very well be an individual parameter depending on the abundance of the type of galaxy for instance. This is not something I'll look into further though, since my objective has been fullfilled; proving that the modified code indeed does a significantly better job.

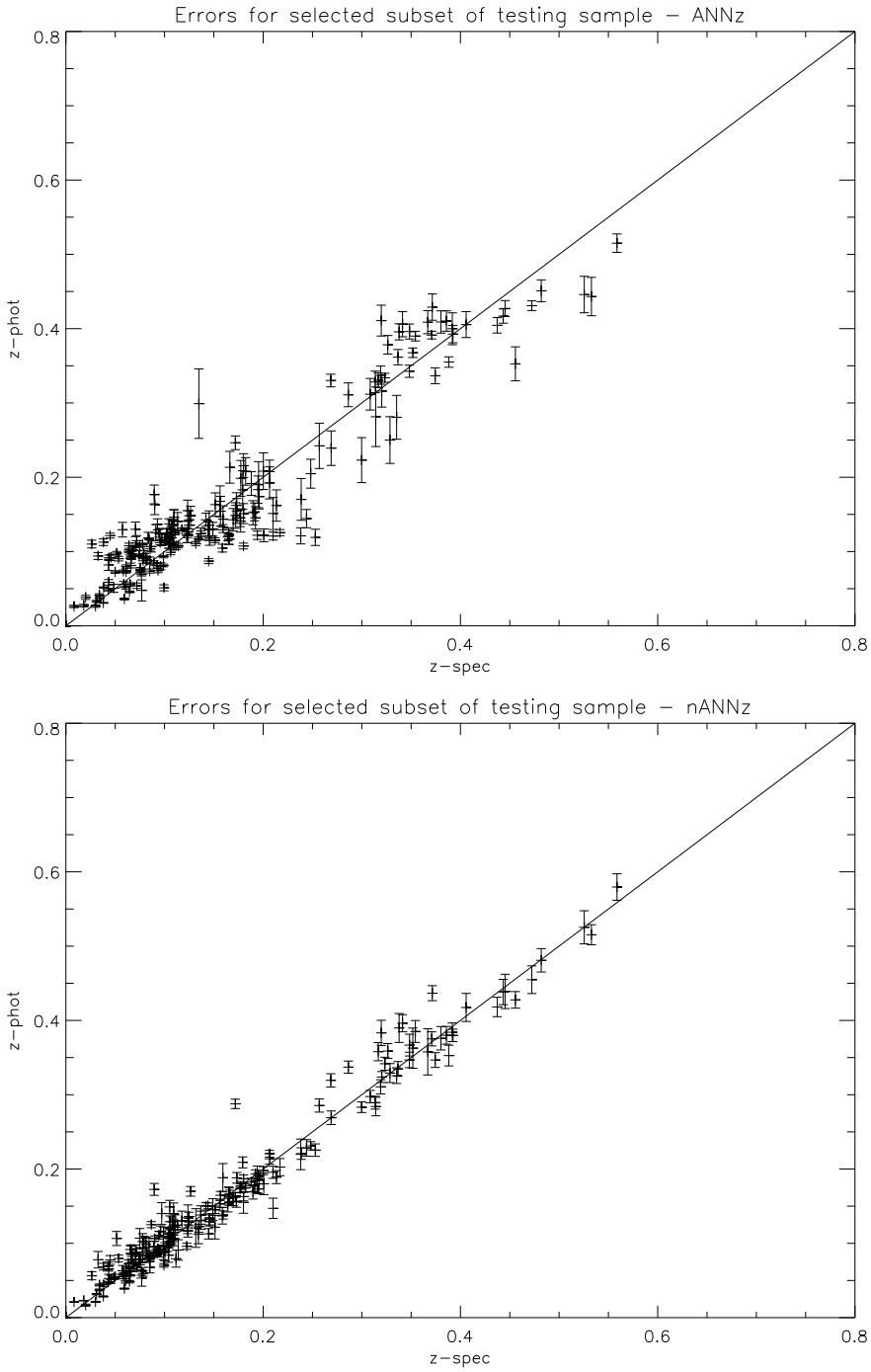



Figure 6.5: Photometric vs. spectroscopic redshift errors for subset of SDSS6 test sample. Upper panel: ANNz. Lower panel: nANNz

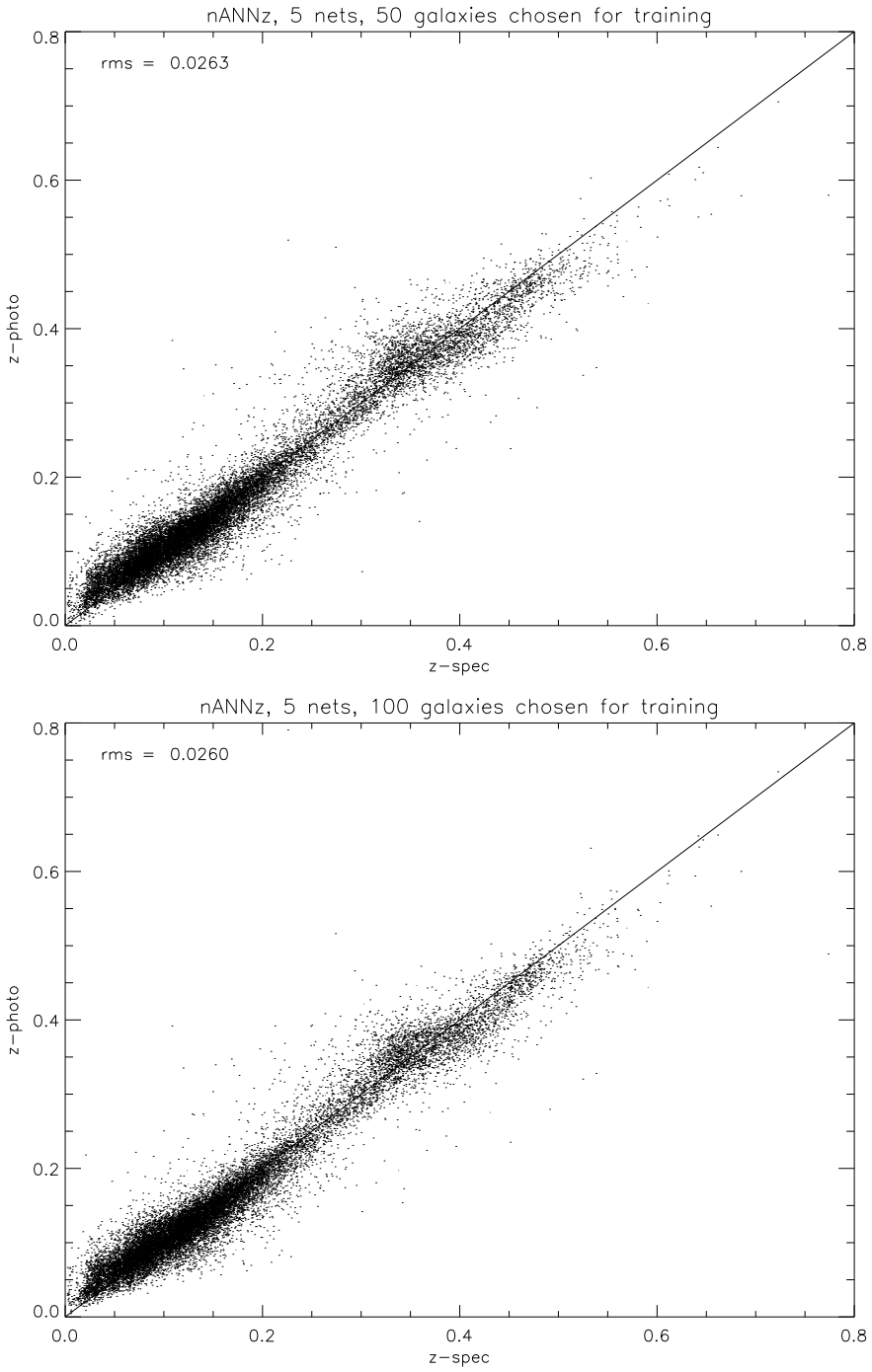



Figure 6.6: Photometric vs. spectroscopic redshift for SDSS6 test sample. Upper panel: nANNz. with 5 nets, where the 50 closest galaxies in color-space were selected for each training. Lower panel: nANNz with 5 nets, where the 100 closest galaxies in color-space were selected for each training.

# Chapter 7

# Future work.

While working on this thesis, a couple of improvements have come up that might give promise of even smaller uncertainties in future use of ANN as a means of estimating photometric redshifts, and thus the possibility of getting an actual valuable and trustworthy tool in research fields where for instance speed is an essence.

---

1. Optimal selection of architecture.
2. Automatic selection of optimum #Gal selected for training.
3. Inclusion of ALL zSpecs in training -and validation samples.
4. Weighted training.
5. Extraction of eClass spec from SDSS.
6. Redshift effect.

---

1: To avoid over fitting and unnecessary use of computer-time as noted in a previous chapter. It can either be done by removing nodes one at a time until huge differences in the results are seen, or by adding until the opposite happens. The result should be a better overall performance in estimations.
2: This is probably the most important part. As noted earlier there's no need whatsoever to include galaxies in the temporary training sample that does not objectively resemble the test galaxy at hand. If so, unnecessary *noise* (in lack of a better term) is added to the resulting parametrization representing the galaxy in focus. Check the figures 5 on page 42 and 6.1 on

page 52 again. Maybe some kind of cut off could be made, when the total color-space distance (lower right frames) has deviated some fractional amount from the closest galaxies in this parameter space. Maybe color-space selection isn't even the best way to do it, or maybe it should work side by side with other parameters for optimum selection. One thing at least is certain, the number 1000 I initially chose and stuck with, is completely arbitrary, and thus logically not the best choice. Nor is there a single number that will be best choice for all types of galaxies. Galaxy types that are numerous of in catalogue will be better off trained on a lot of almost similar ones, than weird looking irregulars that probably will find more unusuable information added to the training process than needed if a too high number of galaxies are selected into the temporary training sample. 3: In the chapter concerning the filtering of the SDSS data, I noted that I'd chosen to only include the galaxies with the zSpec set equal to 4. It ensured, I only worked with the galaxies that had undergone the most precise method of spectroscopical redshift determination. However logically this is done for the relatively brighter and closer galaxies in the total sample, whereby I effectively left out some of the probably more interesting galaxies. Since I chose to train on 1000 selected galaxies afterwards, it might just have been a good move in this case, but generally at least some of the other zSpec flagged galaxies should be included.
4: By adding identical lines of data to the temporary training file, the copied line will get more weight in the process, since the algorith will just see it as though a lot of galaxies had this specific look. By adding a number of copies of each line depending on how close it resembles the test galaxy in focus, could thereby enhance the value of the information coming from the galaxies of closest resemblence. When computer time gets even easier available, you could even train on a temporary file including all the traing sample galaxies in an amount depending on their resemblence. Until then, some golden middle course should probably be used - to say the least... 5: The spectroscopical eClass value included in the SDSS database is a number ranging from about -0.35 to 0.5 for early- to late-type galaxies calculated on basis of different spectroscopical values. It could be a very nice test to see how good the nANNz is at estimating this parameter as a second output node. Also this parameter could be included as a 6th input node in future works. 6: When estimating distances to distant galaxies, by training on closer ones that seemingly looks similar, you risk working on a worse subset than optimum simply because you only compare the apparent flux values. But according to the inverse square law, a galaxy, $b$, at twice the distance of another galaxy, $a$ that's four times as bright, will be seen as though they're equally bright and thus close in color-space. On top of this, the spectrum is

more shifted for the galaxy furthest away, which further implies that if you had two identical galaxies at different redshifts, they would probably not be close enough in color-space to get selected for each other's training. This is simply not feasible and should somehow be avoided although it appears to include some prior knowledge to the absolute brightness of the objects, which is sort of what we're looking for in the first place.

# Chapter 8

# Outro and conclusion

## 8.1 Outro

Artificial Neural Networks caught my eye in popular science articles many years ago simply because of the bombastic statements that the possibility for getting better understanding of the human brains was suddenly possible. Frightening scenarioes were presented for the future and is alive and kicking this day forth. When I was first presented with the possibility of working with ANN in my master thesis by Jens, I had already read more than enough on the topic, on a still not so detailed level though, to know that it primarily was a mathematical tool, a numerical method involving (second order) derivatives as a means of closing in on a global minimum in some kind of abstract parameter space. Obviously I grabed the offer almost instantly.
I could probably have gotten easier and faster(!) to the goal - handing in my thesis - a lot earlier, had I chosen a not so programming technically tough topic, with lots of obvious (in retrospect at least) time consuming pits, where the actual data runs on top of it, also have taken many months in total, before I finally chose (was advised) to be satisfied - a couple of the things mentioned in the above chapter 'Future work' would be really nice to go through, but not for me, at least not now.
However from the day I realized that I could actually produce quality results that might even be significant in some form, either as they stand or with someone's following work, it has really been a thrill to work with the topic. I find that the method itself is extremely useful, and I actually find the main result of the thesis, figure 6 on page 47 beautiful in itself. I was prepared to *just* recreate the results of Lahav e.a. and bring the same method to other types of data sets, but in the end, to my undivided

satisfaction, I ended with modifying their code on basis of an intuitive feeling and some logical thinking, which turned out to work better than I could have hoped for. Needless to say, I'm quite satisfied with the outcome. The method of ANN could maybe even get a second boost, who knows. The actual beauty of the method is that it is absolutely indifferent, whether it's working on the distance to galaxies as a function of apparent flux densities, or something way more down to earth that the human mind realizes there's a connection between, but is just overwhelmed by the complexity of.

## 8.2 Conclusion.

When working with distant galaxies you'd like to be able to measure their spectrum for many reasons. One of them is because Hubble's law, the direct proportionality between the amount that known spectral lines are all shifted towards longer wavelenghts, as a function of how much the space between it and the observer has expanded since emission, makes this redshift a direct distance measurement. When spectroscopy for some reason isn't applicable, other indirect methods with photometry as it's basis, can do the job, with somewhat higher uncertainties though. One of these methods is the use of Artificial Neural Networks (ANN) which Lahav e.a. in 2001 showed was highly competitive compared to other methods working in the same field. In this thesis I've started out with walking in their footsteps, showing how the software works and testing it on ready-to-use example data. Later on, I've worked on extracting up-to-date data from the huge SDSS database and prepared them with a filtering program for easy use in the existing code. My main project was to incorporate a selection mechanism, I found could logically do an even better job than Lahav e.a.'s *Artificial Neural Network z-phot* (ANNz) could. What they did was to train one set of neural network weights on all of the galaxies they had available, thus gaining a parametrization that was later valid for each galaxy outside the training sample. The point was that by training on galaxies where the spectroscopical redshift was already known, one could know in advance what kind of error to expect when later using the parametrization on *real* data, where it was not. The idea behind my improved *nearby Artificial Neural Network z-phot* (nANNz) code is that in the *Galactic Zoo* illustrated by the classic Hubble Fork diagram as in figure 2.2 on page 10, it's illogical to think that *one* parametrization should be enough to cover the whole bunch. And furthermore the problem is that if, say, you trained on a set of large spiral galaxies, but incidentally added a fuzzy irregular, the latter would get it's saying in the final parametrization, which in the end would make the model less valid at predicting photometric redshifts for large spirals. What the nANNz code basically does differently then, is to look at each galaxy in the test sample individually, and then select those in the training sample only that resembles it the most. The selection is done only by color-space distances, but could easily incorporate other parameters too. Figure 6 on page 47 is the main result of this thesis, which clearly illustrates the improvement from the ANNz-code (upper frame) to the nANNz-code (lower frame). Both frames have been processed with the very same data, and each dot represents a galaxy which redshift has been plotted with the spectroscopical one extracted directly from the SDSS

database on the horizontal axis, and the photometric redshift estimated with either ANNz or nANNz on the vertical axis. Since it's originally the same parameter that's illustrated on both axes, all dots should lie on the angular bisector if the photometric redshifts could be estimated with as less uncertainty as the spectroscopical ones. However uncertainties inherited from the photometrically measured magnitudes, and the network variance makes this impossible.
What is significant, is that on average, the dots are closer to the angular bisector by some 39.60% in the nANNz estimations. Furthermore the amount of *drifters*, i.e. points that are very far from the line, are fewer, which makes promises for the future with respect to some day presenting a trustworthy tool for easy and fast distance estimations to faint and distant objects. A code similar to the ANNz is incorporated in the SDSS database as a ready to use parameter associated to a whole lot of the galaxies that can be found there. Maybe nANNz could indeed do a better job as my work more than suggests?

# Chapter 9

# Dansk resume - danish summary

Med spektroskopi kan man opløse og identificere de enkelte spektrallinjer i en galakses spektrum som værende stråling udsendt fra overgange mellem specifikke atomare konfigurationer som de ses i kontrollerede laboratorieforsøg p Jorden. Tillige vil man opdage, at samtlige genkendelige linjer i galaksens spektrum er ligeligt forskudt mod den røde og mindre energirige ende af spektret med en størrelse som udelukkende afhænger proportionalt med afstanden til galaksen som formuleret af Edwin Hubble og Milton L. Humason In 1929, hvad der idag kendes som "Hubbles lov". Når en galakse er såfjern og svag, at spektroskopi er teknisk umuligt, måman bruge andre mere indirekte metoder med udgangspunkt i langt mindre præcise fotometriske målinger for at bestemme rødforskydningen, som sammenholdt med en model for universets ekspansion, dermed bliver et egentlig afstandsmål. I 2004 demonstrerede Lahav e.a. effektiviteten af kunstige (artificial) neurale netvrk (ANN) til hjælp mht. at gennemskue de store mængder rådata, hvor resultater fra relativt nærme galakser, hvor spektroskopi er mulig, sammenholdes med fotometriske målinger påsvagere objekter, hvor en præcis bestemmelse af afstanden kan sikre bedre resultater i fintuningen af andre modeller af kosmologisk betydning.
Figure 6 påside 47 er en illustration af dette speciales hovedresultat, idet jeg satte mig for at modificere Lahav e.a.'s oprindelige kode ved at tilføje en selektionsmekanisme med henblik påsærskilt udvælgelse af de bedst egnede galakser i hver enkelt delresultat. I den oprindelige metode (ANNz) trænedes t sæt neurale netværk påbaggrund af samtlige galakser i et træningssample som resulterede i n linearkombination som beskrev forbindelsen mellem en inputvektor, $\mathbf{x}$ og outputet $\mathbf{y}$ via de i det trænede netværk optimerede vægte $\mathbf{x}$. I den modificerede version (nANNz) tages

der derimod udgangspunkt i hver enkelt ny galakse, man er interesseret i at bestemme afstanden til med sigte på, at man med tiden kan opstille et troværdigt værktøj, såman rent faktisk kan benytte det ogsåtil galakser, hvor man ikke har "facit" ud fra de spektroskopiske målinger som målestok. Fælles for de 2 grafer er, at de samme data ligger til grund for samtlige punkter - eneste forskel er altsåden indbyggede selektionsmekanisme. Hver prik i graferne repræsenterer en galakse hvortil afstanden er bestemt ud fra såvel spektroskopi (x-aksen) og neurale netværk med udgangspunkt i fotometri (y-aksen). Den øverste graf er påbaggrund af ANNz og den nederste nANNz. Forskellen i kvaliteten af beregningerne er ret tydelig selv med det blotte øje, idet punkterne beregnet med den modificerede kode i gennemsnit ligger 39,60% tættere påvinkelhalveringslinjen, som er den virtuelle linje punkterne ville ligge på, hvis de neurale netværk kunne trænes optimalt. Endvidere ses en markant nedgang i antallet af punkter, hvor de neurale netværk enten over -eller undershooter resultatet - en yderst vigtig parameter, hvis man påsigt skal gøre sig forhåbninger om at opbygge et egentligt værktøj, hvis resultater, man kan have tillid til.

# Chapter 10

# Acknowledgements

I'd like to thank all of my friends and family who's made the long awaiting of the hand-in bearable. Thanks to those of you who supported me, pushed me, and even just left me to myself, when that was needed.
Thanks also goes to my two supervisors, Jens Hjorth and Kristian Pedersen for keeping their faith in me, and advising me in a comforting and professional way, whenever I found it necessary/unavoidable to stick my head into your offices.
A special dedication goes to my girlfriend, Annette, who has shown incomprehensible amounts of patience even at times, where special emotional pressure was put on us at the arrival of our twin daughters Astrid & Elise a couple of weeks ago.

# Appendix A

# Sourcecodes - in order of appearance

## A.1 plot_zvz.pro

```
PRO plot_zvz2

aN    =2  ;Choose from list of input files
bN    =5  ;0 if one file only
zvz   =0  ;1 for plot of z vs. z , otherwise 0
err   =1  ;1 for plot of errors, otherwise 0
EselA =100 ;Fraction of data selected for plot of errors in file a
EselB =100 ;Fraction of data selected for plot of errors in file b
acol  =4
bcol  =4
;------------------------List of usable input files----------------------|
; 1: '../EXAMPLE/sdss.ugriz.testresult'
; 2: '../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.ANNz.result'
; 3: '../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.25_5_result'
; 4: '../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.100_5_result'
; 5: '../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.1000_5_result'
;------------------------------------------------------------------------|

; Selecting file a
IF (aN EQ 1) THEN BEGIN
  afil ='../EXAMPLE/sdss.ugriz.testresult'
  acol =3 ;#columns in afil
ENDIF
```

```
IF (aN EQ 2) THEN BEGIN
  afil ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.ANNz.result'
  acol =3 ;#columns in afil
ENDIF
IF (aN EQ 3) THEN afil    ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.25_5_resu
IF (aN EQ 4) THEN afil    ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.100_5_res
IF (aN EQ 5) THEN afil    ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.1000_5_re

;Selecting file b
IF (bN EQ 1) THEN BEGIN
  bfil ='../EXAMPLE/sdss.ugriz.testresult'
  bcol =3 ;#columns in afil
ENDIF
IF (bN EQ 2) THEN BEGIN
  bfil ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.ANNz.result'
  bcol =3 ;#columns in afil
ENDIF
IF (bN EQ 3) THEN bfil    ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.25_5_resu
IF (bN EQ 4) THEN bfil    ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.100_5_res
IF (bN EQ 5) THEN bfil    ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.1000_5_re

;Creating output file according to selected parameters
IF (bN EQ 0) THEN BEGIN
 IF ((zvz EQ 1) AND (err EQ 0)) THEN BEGIN
  IF (aN EQ 1) THEN output ='../REPORT/figs/example_result.ps'
  IF (aN EQ 2) THEN output ='../REPORT/figs/sdss6.ANNz.result.ps'
  IF (aN EQ 3) THEN output ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.25_5_res
  IF (aN EQ 4) THEN output ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.100_5_re
  IF (aN EQ 5) THEN output ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.1000_5_
 ENDIF
 IF ((zvz EQ 0) AND (err EQ 1)) THEN BEGIN
  IF (aN EQ 1) THEN output ='../REPORT/figs/example_error.ps'
  IF (aN EQ 2) THEN output ='../REPORT/figs/sdss6.ANNz.error.ps'
  IF (aN EQ 3) THEN output ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.25_5_er
  IF (aN EQ 4) THEN output ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.100_5_e
  IF (aN EQ 5) THEN output ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.1000_5_
 ENDIF
 IF ((zvz EQ 1) AND (err EQ 1)) THEN BEGIN
  IF (aN EQ 1) THEN output ='../REPORT/figs/example_both.ps'
  IF (aN EQ 2) THEN output ='../REPORT/figs/sdss6.ANNz.both.ps'
  IF (aN EQ 3) THEN output ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.25_5_bo
```

```
  IF (aN EQ 4) THEN output ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.1
  IF (aN EQ 5) THEN output ='../../../../d1/joher/ANNz/nANNz/SDSS6/sdss6.nANNz.1
 ENDIF
ENDIF
IF (bN NE 0) THEN BEGIN ;Add special cases if needed
 IF ((zvz EQ 1) AND (err EQ 0)) THEN BEGIN
  IF ((aN EQ 1) and (bN EQ 2)) THEN output ='../REPORT/figs/ANNresults.ps'
  IF ((aN EQ 2) and (bN EQ 5)) THEN output ='../REPORT/figs/modelresults.ps'
  IF ((aN EQ 3) and (bN EQ 4)) THEN output ='../REPORT/figs/numbergaleff.ps'
  IF ((aN EQ 4) and (bN EQ 5)) THEN output ='../REPORT/figs/nANNz100_1000_5resul
 ENDIF
 IF ((zvz EQ 0) AND (err EQ 1)) THEN BEGIN
  IF ((aN EQ 2) and (bN EQ 5)) THEN output ='../REPORT/figs/modelerrors.ps'
  IF ((aN EQ 4) and (bN EQ 5)) THEN output ='../REPORT/figs/nANNz100_1000_5erro
 ENDIF
 IF ((zvz EQ 1) AND (err EQ 1)) THEN BEGIN
  IF ((aN EQ 2) and (bN EQ 5)) THEN output ='../REPORT/figs/modelboths.ps'
 ENDIF
ENDIF

; Reading files
  PRINT,'reading file a.'
  atemp       =READ_ASCII(afil, DATA_START=0, MISSING_VALUE=9999.000, count=anta
  PRINT,'File a was read:  ', afil
  afil        =atemp.field1
  GalA        =n_elements(afil)/acol
  number_a    =dblarr(GalA)
  spec_a      =dblarr(GalA)
  photo_a     =dblarr(GalA)
  error_a     =dblarr(GalA)
  spec_a_sel  =dblarr(GalA)
  photo_a_sel =dblarr(GalA)
  error_a_sel =dblarr(GalA)
  IF (acol EQ 4) THEN BEGIN
    number_a  =afil(0,*)
  ENDIF
  spec_a      =afil(acol-3,*)
  photo_a     =afil(acol-2,*)
  error_a     =afil(acol-1,*)
  stat_a      =spec_a-photo_a
sumcheck=0
```

```
MinDev_a = number_a(WHERE(ABS(stat_a) EQ MIN(ABS(stat_a))))
MaxDev_a = number_a(WHERE(ABS(stat_a) EQ MAX(ABS(stat_a))))
PRINT, 'Min_res a:  ',MIN(ABS(stat_a)), MinDev_a, ABS(stat_a(MinDev_a-1)), spec_a(MinD
PRINT, 'Max_res a:  ',MAX(ABS(stat_a)), MaxDev_a, ABS(stat_a(MaxDev_a-1)), spec_a(MaxD
;FOR i=0,N_ELEMENTS(GalA)-1 DO BEGIN
; PRINT, i, stat_a(i), stat_a(i)^2
; sumcheck=sumcheck+stat_a(i)^2
;  PRINT, i, sumcheck
;ENDFOR
;PRINT, sumcheck
  sum_a1      =MOMENT(stat_a)
  sum_a2      =TOTAL((stat_a)^2)
  rms_a       =SQRT(sum_a2/GalA)
  PRINT,'#Gal in file a:   ', GalA
  PRINT,'Max z-spec a:     ', MAX(spec_a)
  PRINT,'Mean a:           ', sum_a1(0)
  PRINT,'RMS a:            ', rms_a
IF (bN NE 0) THEN BEGIN
  PRINT,'reading file b.'
  btemp       =READ_ASCII(bfil, DATA_START=0, MISSING_VALUE=9999.000, count=antal_b)
  PRINT,'File b was read:  ', bfil
  bfil        =btemp.field1
  GalB        =n_elements(bfil)/bcol
  number_b    =dblarr(GalB)
  spec_b      =dblarr(GalB)
  photo_b     =dblarr(GalB)
  error_b     =dblarr(GalB)
  spec_b_sel  =dblarr(GalB)
  photo_b_sel =dblarr(GalB)
  error_b_sel =dblarr(GalB)
  IF (bcol EQ 4) THEN BEGIN
    number_b  =bfil(0,*)
  ENDIF
  spec_b      =bfil(bcol-3,*)
  photo_b     =bfil(bcol-2,*)
  error_b     =bfil(bcol-1,*)
  stat_b      =spec_b-photo_b
  sum_b1      =MOMENT(stat_b)
  sum_b2      =SUM((stat_b)^2)
  rms_b       =SQRT(sum_b2/GalB)
  PRINT,'#Gal in file b:   ', N_Elements(stat_b)
```

```
  PRINT,'Max z-spec b:     ', number_b(WHERE(spec_b EQ MAX(spec_b))), MAX(spec_b
  PRINT,'Mean a:           ', sum_a1(0)
  PRINT,'RMS b:            ', rms_b
ENDIF
IF (bN NE 0) THEN BEGIN
; PRINT, 'Improvement of difference in percent:', (mom_a2(0)-mom_b2(0))/(mom_a2(
 PRINT, 'Improvement of difference in percent:', (rms_a-rms_b)/rms_a*100
ENDIF

;Creating array for plotting of angular bisector
tendens  =dblarr(11)
FOR j=0,N_ELEMENTS(tendens)-1 DO BEGIN
 tendens(j)=j/10.
ENDFOR

;Creating error arrays
IF (err EQ 1) THEN BEGIN
 sela = 0
 PRINT, 'Creating errorarray_a'
 FOR i=0,N_ELEMENTS(spec_a)-1 DO BEGIN
  IF ((i MOD EselA) EQ 0) THEN BEGIN
   spec_a_sel(sela) =spec_a(i)
   photo_a_sel(sela)=photo_a(i)
   error_a_sel(sela)=error_a(i)
   sela=sela+1
  ENDIF
 ENDFOR
 PRINT, 'Galaxies chosen for errorprint a', sela
 IF (bN NE 0) THEN BEGIN
  selb = 0
  PRINT, 'Creating errorarray_b'
  FOR i=0,N_ELEMENTS(spec_b)-1 DO BEGIN
   IF ((i MOD EselB) EQ 0) THEN BEGIN
    spec_b_sel(selb) =spec_b(i)
    photo_b_sel(selb)=photo_b(i)
    error_b_sel(selb)=error_b(i)
    selb=selb+1
   ENDIF
  ENDFOR
  PRINT, 'Galaxies chosen for errorprint b', selb
 ENDIF
```

```
ENDIF

SET_PLOT, 'x'
IF (bN EQ 0) THEN BEGIN
 IF ((zvz EQ 1) AND (err EQ 1)) THEN BEGIN
   !p.multi=[0,1,2]
   plot, spec_a, photo_a, psym=3, XRANGE=[0.0,0.8], YRANGE=[0.0,0.8] , TITLE='Plot of
   oplot, tendens, tendens
   XYOUTS, 0.120, 0.9, 'rms = ', /NORMAL
   XYOUTS, 0.145, 0.9, rms_a, /NORMAL
   ploterror, spec_a_sel, photo_a_sel, error_a_sel, PSYM=1, XRANGE=[0.0,0.8], YRANGE=[
   oplot, tendens, tendens
   SET_PLOT, 'ps' ; create .ps-file:
   DEVICE, filename=output, yoffset=2.5, ysize=25
   plot, spec_a, photo_a, PSYM=3, XRANGE=[0.0,0.8], YRANGE=[0.0,0.8] , TITLE='Plot of
   oplot, tendens, tendens
   XYOUTS, 0.120, 0.9, 'rms = ', /NORMAL
   XYOUTS, 0.145, 0.9, rms_a, /NORMAL
   ploterror, spec_a_sel, photo_a_sel, error_a_sel, PSYM=1, XRANGE=[0.0,0.8], YRANGE=[
   oplot, tendens, tendens
 ENDIF ELSE BEGIN
  !p.multi=[0,1,1]
  IF (zvz EQ 1) THEN BEGIN
   plot, spec_a, photo_a, psym=3, XRANGE=[0,0.8], YRANGE=[0,0.8] , xtitle='z-spec', yt
   oplot, tendens, tendens
   XYOUTS, 0.120, 0.9, 'rms = ', /NORMAL
   XYOUTS, 0.145, 0.9, rms_a, /NORMAL
   SET_PLOT, 'ps' ; De samme plots laegges i filer til udskrift:
   DEVICE, filename=output
   plot, spec_a, photo_a, PSYM=3, XRANGE=[0,0.8], YRANGE=[0,0.8]
   oplot, tendens, tendens
   XYOUTS, 0.120, 0.9, 'rms = ', /NORMAL
   XYOUTS, 0.145, 0.9, rms_a, /NORMAL
  ENDIF ELSE BEGIN
   ploterror, spec_a_sel, photo_a_sel, error_a_sel, PSYM=1, XRANGE=[0.0,0.8], YRANGE=[
   oplot, tendens, tendens
   SET_PLOT, 'ps' ; De samme plots laegges i filer til udskrift:
   DEVICE, filename=output
   ploterror, spec_a_sel, photo_a_sel, error_a_sel, PSYM=1, XRANGE=[0.0,0.8], YRANGE=[
   oplot, tendens, tendens
  ENDELSE
```

```
 ENDELSE
ENDIF

IF (bN NE 0) THEN BEGIN
 IF ((zvz EQ 1) AND (err EQ 1)) THEN BEGIN
   !p.multi=[0,2,2]
   plot, spec_a, photo_a, psym=3, XRANGE=[0,0.8], YRANGE=[0,0.8] , xtitle='z-spe
   oplot, tendens, tendens
   XYOUTS, 0.060, 0.95, 'rms = ', /NORMAL
   XYOUTS, 0.075, 0.95, rms_a, /NORMAL
   plot, spec_b, photo_b, psym=3, XRANGE=[0,0.8], YRANGE=[0,0.8] , xtitle='z-spe
   oplot, tendens, tendens
   XYOUTS, 0.560, 0.95, 'rms = ', /NORMAL
   XYOUTS, 0.575, 0.95, rms_b, /NORMAL
   ploterror, spec_a_sel, photo_a_sel, error_a_sel, PSYM=1, XRANGE=[0.0,0.8], YR
   oplot, tendens, tendens
   ploterror, spec_b_sel, photo_b_sel, error_b_sel, PSYM=1, XRANGE=[0.0,0.8], YR
   oplot, tendens, tendens
   SET_PLOT, 'ps' ; Dump to .ps
   DEVICE, filename=output, xsize=20, ysize=30
  !p.multi=[0,2,2]
   plot, spec_a, photo_a, psym=3, XRANGE=[0,0.8], YRANGE=[0,0.8] , xtitle='z-spe
   oplot, tendens, tendens
   XYOUTS, 0.060, 0.95, 'rms = ', /NORMAL
   XYOUTS, 0.075, 0.95, rms_a, /NORMAL
   plot, spec_b, photo_b, psym=3, XRANGE=[0,0.8], YRANGE=[0,0.8] , xtitle='z-spe
   oplot, tendens, tendens
   XYOUTS, 0.560, 0.95, 'rms = ', /NORMAL
   XYOUTS, 0.575, 0.95, rms_b, /NORMAL
   ploterror, spec_a_sel, photo_a_sel, error_a_sel, PSYM=1, XRANGE=[0.0,0.8], YR
   oplot, tendens, tendens
   ploterror, spec_b_sel, photo_b_sel, error_b_sel, PSYM=1, XRANGE=[0.0,0.8], YR
   oplot, tendens, tendens
 ENDIF ELSE BEGIN
  !p.multi=[0,1,2]
  IF (zvz EQ 1) THEN BEGIN
   plot, spec_a, photo_a, psym=3, XRANGE=[0,0.8], YRANGE=[0,0.8] , TITLE='ANNz',
   oplot, tendens, tendens
   XYOUTS, 0.150, 0.95, 'rms = ', /NORMAL
   XYOUTS, 0.225, 0.95, Number_formatter(rms_a, Decimals=4),  /NORMAL
   plot, spec_b, photo_b, psym=3, XRANGE=[0,0.8], YRANGE=[0,0.8] , TITLE='nANNz
```

```
   oplot, tendens, tendens
   XYOUTS, 0.150, 0.45, 'rms = ', /NORMAL
   XYOUTS, 0.225, 0.45, Number_formatter(rms_b, Decimals=4),  /NORMAL
   SET_PLOT, 'ps' ; Create .ps-file
   DEVICE, filename=output, xsize=10, ysize=15
   plot, spec_a, photo_a, PSYM=3, XRANGE=[0,0.8], YRANGE=[0,0.8], TITLE='ANNz', xtitle=
   oplot, tendens, tendens
   XYOUTS, 0.150, 0.95, 'rms = ', /NORMAL
   XYOUTS, 0.225, 0.95, Number_formatter(rms_a, Decimals=4),  /NORMAL
   plot, spec_b, photo_b, psym=3, XRANGE=[0,0.8], YRANGE=[0,0.8], TITLE='nANNz',xtitle=
   oplot, tendens, tendens
   XYOUTS, 0.150, 0.45, 'rms = ', /NORMAL
   XYOUTS, 0.225, 0.45, Number_formatter(rms_b, Decimals=4),  /NORMAL
  ENDIF ELSE BEGIN
   !p.multi=[0,1,2]
   ploterror, spec_a_sel, photo_a_sel, error_a_sel, PSYM=1, XRANGE=[0.0,0.8], YRANGE=[
   oplot, tendens, tendens
   ploterror, spec_b_sel, photo_b_sel, error_b_sel, PSYM=1, XRANGE=[0.0,0.8], YRANGE=[
   oplot, tendens, tendens
   SET_PLOT, 'ps' ; De samme plots laegges i filer til udskrift:
   DEVICE, filename=output, xsize=20, ysize=30
   ploterror, spec_a_sel, photo_a_sel, error_a_sel, PSYM=1, XRANGE=[0.0,0.8], YRANGE=[
   oplot, tendens, tendens
   ploterror, spec_b_sel, photo_b_sel, error_b_sel, PSYM=1, XRANGE=[0.0,0.8], YRANGE=[
   oplot, tendens, tendens
  ENDELSE
 ENDELSE
ENDIF
PRINT, 'An output file was created:  ', output
DEVICE, /CLOSE
SET_PLOT, 'x'

END
```

## A.2 filter.f90

```
!*********************************************************************
! PROGRAM FOR FILTERING DATA FROM SDSS DATABASE FOR EASY USE I ANN-CODE *
!                          PROVIDED BY LAHAV E.A.                      *
!                THIS PROGRAM IS MADE BY ERLING JOHNSEN.               *
!                                                                      *
!                Compile as 'f90 filter.f90 file_fns.f90'              *
!                Run as a.out getarg1-5 (See lines 21-25)              *
!*********************************************************************

PROGRAM filter
USE file_fns
IMPLICIT NONE
!Variable:
INTEGER :: i, j, k, n_params, n_gals, c, sum, B(13), spec(13), spec_value, stat
REAL*16, ALLOCATABLE :: A(:,:)
CHARACTER (len=100) :: header, input, test_number
CHARACTER (len=100) :: output_train, output_valid, output_test, output_stat

PRINT *, 'Opening and creating files'

CALL GETARG(1, input)
CALL GETARG(2, output_train)
CALL GETARG(3, output_valid)
CALL GETARG(4, output_test)
CALL GETARG(5, output_stat)
CALL GETARG(6, test_number)
PRINT *, 'Input:   ', input
PRINT *, 'Train:   ', output_train
PRINT *, 'Valid:   ', output_valid
PRINT *, 'Test :   ', output_test
PRINT *, 'Stat :   ', output_stat
OPEN(1, file = input, status = 'old', action = 'read')
OPEN(2, file = output_train, status = 'Replace', action = 'write')
OPEN(3, file = output_valid, status = 'Replace', action = 'write')
OPEN(4, file = output_test,  status = 'Replace', action = 'write')
OPEN(5, file = output_stat,  status = 'Replace', action = 'write')
OPEN(6, file = test_number,  status = 'Replace', action = 'write')

!*********************************************************************
```

```
! specZstatus galaxies to include (values from 0-12 i place 1-13!       *
!*************************************************************************
spec     = (/0,0,0,1,0,0,0,0,0,0,0,0,0/) !
n_params = 16
n_gals   = no_of_rows(input)-1

PRINT *, '# of galaxies:     ', n_gals
PRINT *, '# of parameters:   ', n_params
ALLOCATE (A(n_params,n_gals))
CLOSE(1)
OPEN(1, file = input, status = 'old', action = 'read')
READ(1,*) header
!j=0
!DO j=0,12
!  B(j)=0
!ENDDO
k=0
DO i=1,N_gals                          !REMEMBER A(column,row) in fortran
  READ(1,*) A(:,i)
  IF (MOD(i,10000) .EQ. 0) THEN
    PRINT *, i!, INT(A(1,i))
  ENDIF
!*************************************************************************
! Filtering out galaxies without the right specZstatus and negative z   *
!                Good ones are distributed into 3 files                 *
!*************************************************************************
  spec_value=A(5,i)
  IF ((spec(spec_value) .eq. 1) .AND. (A(2,i) .GT. 0)) THEN
    j=j+1
    IF (MOD(j,10) .NE. 0) THEN
      stat(1)=stat(1)+1
      WRITE(2,*) A(7:16,i), A(2,i)
    ENDIF
    IF (MOD(j,20) .EQ. 0) THEN
      stat(2)=stat(2)+1
      WRITE(3,*) A(7:16,i), A(2,i)
    ENDIF
    IF (MOD(j+10,20) .EQ. 0) THEN
      k=k+1
      stat(3)=stat(3)+1
      WRITE(4,*) A(7:16,i), A(2,i)
```

```
      WRITE(6,FMT='(I6,I7,f22.0,f12.8)') k, j, A(1,i), A(2,i)
    ENDIF
  ENDIF
!*********************************************************************
!           Updating counter array for specZstatus statistics        *
!*********************************************************************
  c=A(5,i)+1
  B(c)=B(c)+1
enddo
!*********************************************************************
!             Creating and printing specZstatus statistics           *
!*********************************************************************

CALL DATATYPE(B, sum)
PRINT *, N_gals, sum, N_gals-sum
PRINT *, B
PRINT *, '#gal in .train-file:', stat(1)
PRINT *, '#gal in .valid-file:', stat(2)
PRINT *, '#gal in .test-file :', stat(3)
PRINT *, 'Total #gal in files:', stat(1)+stat(2)+stat(3)
WRITE (5,*) N_gals, sum, N_gals-sum
WRITE (5,*) B
WRITE (5,*) '#gal in .train-file:', stat(1)
WRITE (5,*) '#gal in .valid-file:', stat(2)
WRITE (5,*) '#gal in .test-file :', stat(3)
WRITE (5,*) 'Total #gal in files:', stat(1)+stat(2)+stat(3)

CLOSE(1)
CLOSE(2)
CLOSE(3)

END PROGRAM

!*********************************************************************
!      Subroutine to create and print the specZstatus statistics     *
!*********************************************************************

SUBROUTINE DATATYPE (B,sum)

Integer :: B(13), j, sum
```

```
sum=0

DO j=1,13
  IF (j .eq. 1) THEN
    PRINT *, 'Redshift not yet measured:'
    WRITE (5,*) 'Redshift not yet measured:'
  ENDIF
  IF (j .eq. 2) THEN
    PRINT *, 'Redshift measurement failed:'
    WRITE (5,*) 'Redshift measurement failed:'
  ENDIF
  IF (j .eq. 3) THEN
    PRINT *, 'Redshift cross-correlation and emz both high confidence,'
    PRINT *, 'but incosistent:'
    WRITE (5,*) 'Redshift cross-correlation and emz both high confidence,'
    WRITE (5,*) 'but incosistent:'
  ENDIF
  IF (j .eq. 4) THEN
    PRINT *, 'Redshift determined from cross-correlation and emz are'
    PRINT *, 'consistent:'
    WRITE (5,*) 'Redshift determined from cross-correlation and emz are'
    WRITE (5,*) 'consistent:'
  ENDIF
  IF (j .eq. 5) THEN
    PRINT *, 'Redshift determined from x-corr with high confidence:'
    WRITE (5,*) 'Redshift determined from x-corr with high confidence:'
  ENDIF
  IF (j .eq. 6) THEN
    PRINT *, 'Redshift determined from cross-correlation with low '
    PRINT *, 'confidence:'
    WRITE (5,*) 'Redshift determined from cross-correlation with low '
    WRITE (5,*) 'confidence:'
  ENDIF
  IF (j .eq. 7) THEN
    PRINT *, 'Redshift from emz plus consistent xcorr redshift '
    PRINT *, 'measurement:'
    WRITE (5,*) 'Redshift from emz plus consistent xcorr redshift '
    WRITE (5,*) 'measurement:'
  ENDIF
  IF (j .eq. 8) THEN
    PRINT *, 'Redshift determined from em/lines with high confidence:'
```

```
    WRITE (5,*) 'Redshift determined from em/lines with high confidence:'
  ENDIF
  IF (j .eq. 9) THEN
    PRINT *, 'Redshift determined from em/lines with low confidence:'
    WRITE (5,*) 'Redshift determined from em/lines with low confidence:'
  ENDIF
  IF (j .eq. 10) THEN
    PRINT *, 'Redshift determined "by hand" with high confidence:'
    WRITE (5,*) 'Redshift determined "by hand" with high confidence:'
  ENDIF
  IF (j .eq. 11) THEN
    PRINT *, 'Redshift determined "by hand" with low confidence:'
    WRITE (5,*) 'Redshift determined "by hand" with low confidence:'
  ENDIF
  IF (j .eq. 12) THEN
    PRINT *, 'x/corr redshift determined when EW(4000- break)>0.95:'
    WRITE (5,*) 'x/corr redshift determined when EW(4000- break)>0.95:'
  ENDIF
  IF (j .eq. 13) THEN
    PRINT *, 'Redshift determined from average of CaII fits:'
    WRITE (5,*) 'Redshift determined from average of CaII fits:'
  ENDIF
  PRINT *, j-1, B(j)
  WRITE (5,*) j-1, B(j)
  sum = sum + B(j)
ENDDO

END SUBROUTINE DATATYPE
```

## A.3 Makefile

```
CC = gcc
FC = f90
F77= f77
FLAGS = -O3

main.x:    annz_train file_fns.o Near_ANNz.o conversion.o annz_ann.o annz_test_CV.o
$(FC) $(FLAGS) -o main.x file_fns.o Near_ANNz.o conversion.o annz_ann.o annz_test_CV.o

annz_train:      annz_vmmin.o annz_ann.o annz_util.o annz_train.o
$(FC) $(FLAGS) -o annz_train annz_vmmin.o annz_ann.o annz_util.o annz_train.o -lm

#Convention:
#Destination: Afhngighed
# compiler -c -o destination afhaengighed

%.o: %.f90
${FC} $(FLAGS) -c -o $@ $<

%.o: %.c
$(CC) $(FLAGS) -c -o $@ $<

%.o: %.f
$(F77) $(FLAGS) -c -o $@ $<

clean:
rm *.mod *.o
```

## A.4 nANNz.f90

```
PROGRAM Near_ANNz

USE file_fns

IMPLICIT NONE
!Variable:
INTEGER :: i, j, k, l, m, o, p, N_params, N_gals, N_train, N_column, c, sum, B(
INTEGER :: spec(13), spec_value, stat(3), check,seed, N_test, N_wts, Max_ite, St
INTEGER, ALLOCATABLE :: Best2(:)
REAL*8, ALLOCATABLE  :: A(:,:), T(:), Best1(:), Dist(:,:)
REAL*8 :: Decay
CHARACTER (len=100) :: header, traindata, testdata, output_train, output_valid,
CHARACTER (len=100) :: argumentfil, testdata_temp, test_result, arch_file, Distf
CHARACTER (len=15), ALLOCATABLE :: wts_file(:)
REAL*8  :: snit_zerr, snit_Uerr, near, near1, near2, near3, near4, near5

traindata     ="SDSS6/sdss6.trainvalid"
testdata      ="SDSS6/sdss6.test"
arch_file     ="arch.5:10:10:1.net"
output_train  ="train_temp.dat"
output_valid  ="valid_temp.dat"
testdata_temp ="testdata_temp.dat"
argumentfil   ="argumentfil.txt"
test_result   ="SDSS6/sdss6.nANNz.max18585.result"
Distfile      ="SDSS6/sdss6_max18585.Distfile"
Photofile     ="SDSS6/sdss6.nANNz.max18585.photos"

Start    = 1                   !Start at galaxy number start in testfile
N_train  = 1000                  !#selected galaxies for training for each test
N_wts    = 5                     !# weightfiles in commitee - MAX 9!
n_params = 11                    !Do not change!
Max_ite  = 10000                 !Just any high value above 2000 will do
seed     = 12974                 !Should probably be changed more elegantly
Decay    = 0.0001
Distchk  = 1                     !Set=1 to create Distfile
ALLOCATE (wts_file(N_wts))
                                 !Open files to count data lines
PRINT *, 'Counting lines in training data file:    ', traindata
OPEN(1, file = traindata, status = 'old', action = 'read')
```

```
N_gals   = no_of_rows(traindata) !# galaxies in train sample
CLOSE(1)
PRINT *, 'Counting lines in test data file:    ', testdata
OPEN(22, file = testdata, status = 'old', action = 'read')
N_test   = no_of_rows(testdata)  !# galaxies in test  sample
CLOSE(22)
PRINT *, '# of galaxies in training file:     ', n_gals

ALLOCATE (A(n_params,n_gals))
ALLOCATE (T(n_params))
ALLOCATE (Best1(N_train))
ALLOCATE (Best2(N_train))
ALLOCATE (Dist(8,n_gals))

OPEN(22, file = testdata, status = 'old', action = 'read')
IF (Start .GT. 1) THEN          !-----------------------------|
  DO o=1,Start-1                !                             |
    READ(22,*) T(:)             !  Go to right line of input  |
  ENDDO                         !                             |
ENDIF                           !-----------------------------|

DO k=Start,N_test               !1 loop for each galaxy for z-phot
  PRINT *, "Starting determination of z-phot for Gal#",k
  PRINT *, N_test-k,"left after this."
  READ(22,*) T(:)               !Read right line of test galaxy data from file
  PRINT *, k, T(:)
  OPEN(7, file = testdata_temp, status = 'old', action = 'write')
    WRITE(7,*) T(:)
  CLOSE(7)
  IF (Distchk .EQ. 1) THEN
    OPEN(8, file = Distfile    , status = 'Old', action = 'write', Position = 'Append'
      WRITE(8,*) k, N_train, T(:)
    CLOSE(8)
  ENDIF
  DO p=1,N_wts                     !***************CHANGE SEED!!!!!
    seed=seed+1
    wts_file(p)='wts_file'//ACHAR(48+p)
    PRINT *, wts_file(p)
    IF (p .EQ. 1) THEN             !Selection is only done once for each galaxy
      OPEN(1, file = traindata, status = 'old', action = 'read')
      DO j=1,N_train               !Reset array
```

```
        Best1(j)=1000000.          !1000 is arbitrarily chosen to be big enough
        Best2(j)=0.
      ENDDO
      DO i=1,N_gals                !REMEMBER A(column,row) in fortran
          READ(1,*) A(:,i)
          near1=(T(1)-A(1,i))
          near2=(T(2)-A(2,i))
          near3=(T(3)-A(3,i))
          near4=(T(4)-A(4,i))
          near5=(T(5)-A(5,i))
          Dist(1,i)=ABS(near1)
          Dist(2,i)=ABS(near2)
          Dist(3,i)=ABS(near3)
          Dist(4,i)=ABS(near4)
          Dist(5,i)=ABS(near5)
          near=SQRT((near1)**2+(near2)**2+(near3)**2+(near4)**2+(near5)**2)
          Dist(6,i)=near
          Dist(7,i)=1
          Dist(8,i)=i
        IF (near .LT. Best1(N_train)) THEN !Selects nearest galaxies in colorspa
          Dist(7,i)=2              !2 is a temporary chosing of the gal. at hand
          DO l=1,N_train
            IF (near .LT. Best1(l)) THEN
!              PRINT *, l, near, Best1(l)
              DO m=N_train,l+1,-1
                Best1(m)=Best1(m-1)
                Best2(m)=Best2(m-1)
              ENDDO
              Best1(l)=near
              Best2(l)=i
!              PRINT *, 'now exiting'
              EXIT
            ENDIF
          ENDDO
        ENDIF
      ENDDO
    CLOSE(1)
    ENDIF
    OPEN(3, file = output_train, status = 'Replace', action = 'write')
    OPEN(4, file = output_valid, status = 'Replace', action = 'write')
    OPEN(6, file = argumentfil , status = 'Replace', action = 'write')
```

```
    IF (Distchk .EQ. 1) THEN
      OPEN(8, file = Distfile    , status = 'Old', action = 'write', Position = 'Appen
    ENDIF
    DO l=1,N_train
      check=Best2(l)
      Dist(7,check)=3          !Parameter changed to 3 for all galaxies selected
      IF (MOD(l,5) .EQ. 0) THEN!Galaxies distributed to files for train and valid
        WRITE(4,*) A(:,Best2(l))
      ELSE
        WRITE(3,*) A(:,Best2(l))
      ENDIF
      IF ((p .EQ. 1) .AND. (Distchk .EQ. 1)) THEN
        WRITE(8, FMT='(5f7.3,12f12.6,f8.0)') A(:,Best2(l)), Dist(1:6,Best2(l)), Dist(8
      ENDIF
    ENDDO

    CLOSE(3)
    CLOSE(4)
    IF (Distchk .EQ. 1) THEN
      CLOSE(8)
    ENDIF
    WRITE(6,*) seed
    WRITE(6,*) Decay
    WRITE(6,*) Max_ite
    WRITE(6,*) "y"
    WRITE(6,*) wts_file(p)
    WRITE(6,*) 0
    WRITE(6,*)
    CLOSE(6)
    CALL called_train(seed)
  ENDDO
  CALL annz_test(testdata_temp, test_result, wts_file , N_wts, arch_file, k, photofile
  PRINT *, "Calculation done - next galaxy starting!"
ENDDO
CLOSE(22)
END PROGRAM
```

# Appendix B

# Miscellaneous outputs

## B.1 Filter.f90 statistics

```
Redshift not yet measured:
0 0
Redshift measurement failed:
1 51
Redshift cross-correlation and emz both high confidence,
but incosistent:
2 28700
Redshift determined from cross-correlation and emz are
consistent:
3 198658
Redshift determined from x-corr with high confidence:
4 409308
Redshift determined from cross-correlation with low
confidence:
5 4316
Redshift from emz plus consistent xcorr redshift
measurement:
6 25010
Redshift determined from em/lines with high confidence:
7 173
Redshift determined from em/lines with low confidence:
8 151
Redshift determined "by hand" with high confidence:
9 777
Redshift determined "by hand" with low confidence:
```

```
10 459
x/corr redshift determined when EW(4000- break)>0.95:
11 0
Redshift determined from average of CaII fits:
12 5
667608 667608 0
0 51 28700 198658 409308 4316 25010 173 151 777 459 0 5
#gal in .train-file: 368377
#gal in .valid-file: 20465
#gal in .test-file : 20465
Total #gal in files: 409307
```

# Bibliography


[1] Adrian A. Collister, Ofer Lahav: *astro-ph/0311058* ANNz: Estimating photometric redshifts using artificial neural networks.

[2] Adrian Collister: ANNz User Guide.

[3] H. Karttunen, P.Krger, H.Oja, M. Poutanen, K.J.Donner: Fundamental Astronomy (Springer-verlag, 3rd ed. 1996).

[4] Bradley W.Carroll, Dale A. Ostlie: An introduction to Modern Astrophysics (Addison-Westley publishing company, Inc. 1996).

[5] D.Emerson: Interpreting Astronomical Spectra (John Wiley and Sons 1996).

[6] D.Wang, Y.X.Zhang, C.Liu and Y.H.Zhao: Kernel regression for determining photometric redshifts form Sloan broad-band photometry.

[7] J.Gorgas, N.Cardiel ans S.Pedraz: Towards an understanding of the $\lambda$ 4000 break behaviour in old stellar population. Astrophysics and space science 263: 167-170, 1999.

[8] Baum, W. A. Problems of Extra-Galactic Research, Proceedings from IAU Symposium no. 15. Edited by George Cunliffe McVittie. International Astronomical Union Symposium no. 15, Macmillan Press, New York, p.390

[9] Glen Cowan: Statistical Data Analysis (Oxford University Press / Clarendon Press, 1998)

[10] Genevieve Orr e.a.: Lecture notes for the course: 'CS-449: Neural Networks', Willamette University, 1999. http://www.willamette.edu/ gorr/classes/cs449/intro.html.

[11] Erwin Kreyszig: 'Advanced Engineering Mathematics', 8th edition, John Wiley & Sons, inc. 1999.

[12] William H. Press e.a: 'Numerical recipes in Fortran77', Cambridge University Press.