Preprint typeset using LaTeX style emulateapj v. 05/12/14

TRANSFER FUNCTION README

Charles E. Romero[1,2,3],
(Dated: April 19, 2016)

Keywords:

1. OVERVIEW

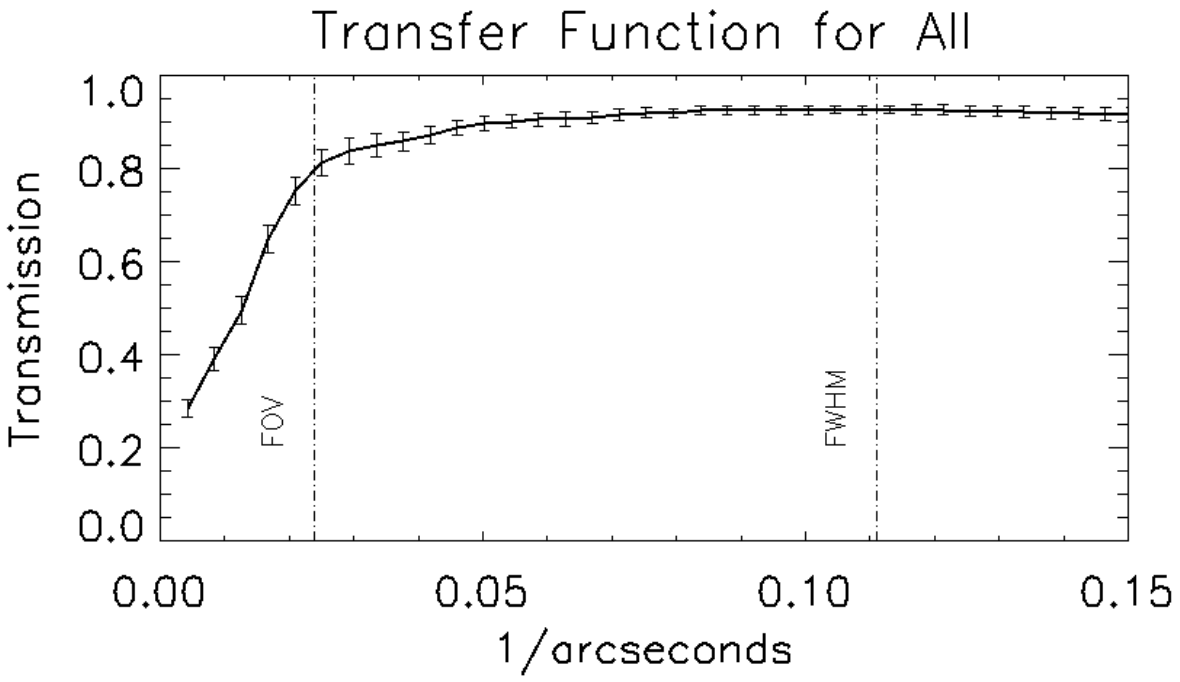

Figure 1. The transfer function for the fourteen clusters analysed by Romero2016. The error bars are from the RMS of individual cluster transfer functions. The largest intercluster standard deviation is 0.03 and occurs at $k \sim 0.016$. That is, at most a 3% scatter in transmission. While this is true, I'll point out that the scatter can be $4-6$ times the reported error in the mean.

The transfer function for each cluster is calculated by simulating observations of 25 fake skies (the fake skies contain white noise). Here, simulated observations make use of the "long QV" structures. One-dimensional power spectra are computer for the input fake skies and output (observed) fake skies. The transfer function is taken as the square root of the ratio of average input power spectra to average output power spectra.

Over the 14 clusters, the transfer functions are recorded, along with associated standard deviations for each wavenumber bin. A transfer function over all fourteen clusters is then calculated as the weighted average of the individual cluster transfer functions. This is shown in a Figure.

2. UPDATE

The original transfer functions apply to white noise across an entire sky. However, that transfer function is not so accurate for a cluster model. We have recalculated the transfer functions by filtering as strictly A10 profile. That is, we filter a gNFW profile with parameters $[\alpha, \beta, \gamma, C_{500}, P_0] = [1.05, 5.49, 0.31, 1.18, 8.42]$ through our standard pipeline and calculate the transfer function for each cluster in this way; we call this our A10 transfer function. For all scales larger than the FWHM (for all wavenumbers smaller than ~ 0.11 inverse arcseconds, we take the lower of the original transfer function and the A10 transfer function; we call the resulting transfer function our cluster transfer function.

[1] IRAM
[2] NRAO
[3] UVA

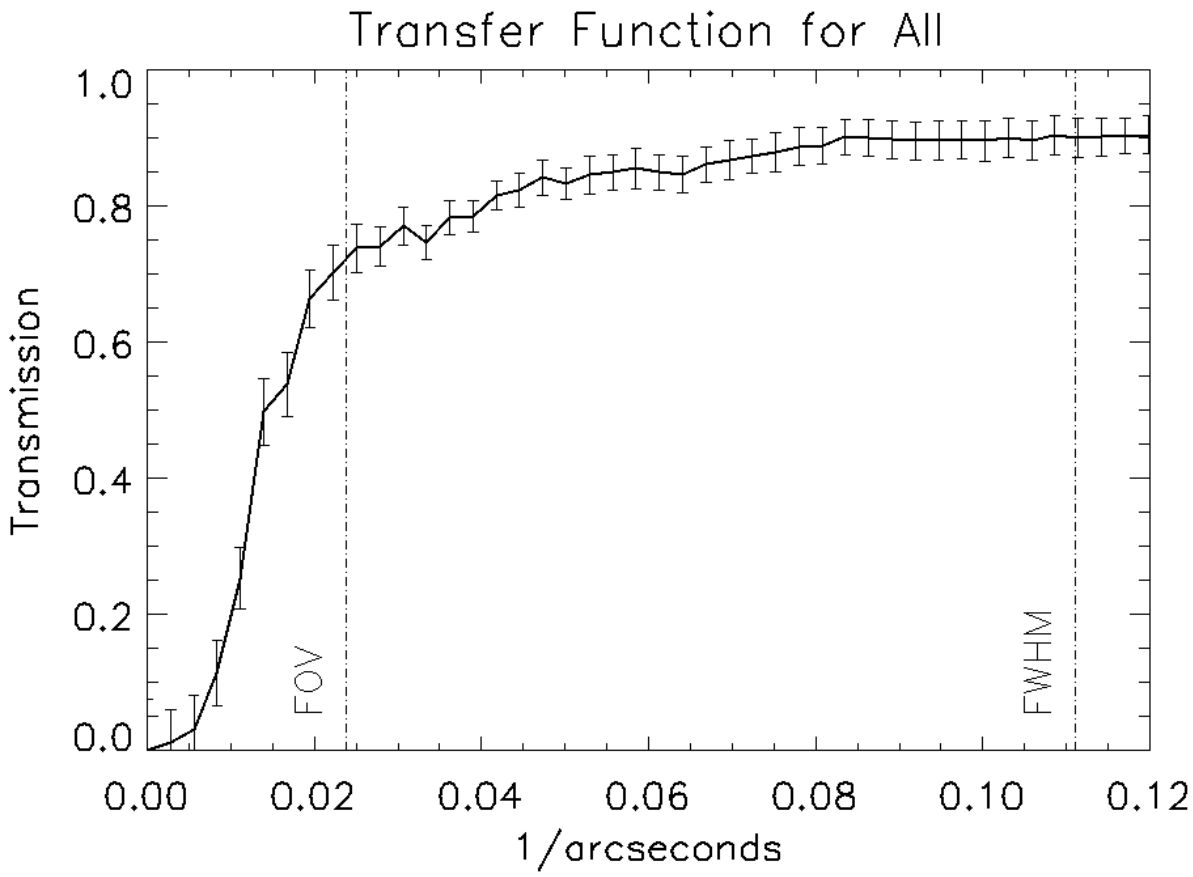

Figure 2. The cluster transfer function for the fourteen clusters analysed by Romero2016. The error bars are from the RMS of individual cluster transfer functions. This transfer function better reproduces filtered cluster models than the white noise transfer function.

We have compared the results of these transfer functions to the standard pipeline and find agreement, principally of the peak amplitude, within 10%. To further quantify the how well the transfer function reproduces the true filtering, we fit our transfer-function-filtered cluster models to our data and compare to the original results. In general, we find good agreement for the cluster models. However, for cluster models that have a large scale radius, and thus exhibit nearly a single power law within the source size ($180''$) will not be reproduced well. Suffice it to say that for clusters with $0.5 < C_{500} < 3.3$ are well reproduced with fitted amplitudes that are within 10% of the original. Furthermore, the profile shape parameters (C_{500}, P_0, and γ) are within $\sim 10\%$ agreement as well.

3. IMPLEMENTATION

In calculating the MUSTANG transfer function(s), we have adopted a source size of $120''$. This means that we select the central 4 arcminutes (length and height of a square) in our maps. We apply a Fourier transform to our input and output fake skies over this region, and calculate the transfer function as the square root of the ratio of average input power spectra to average output power spectra.

Thus, in order to apply the provided transfer functions, you should assume the same (or smaller) source size. In principle the only variables you need to supply are MAP, PIXELSIZE, and FILE. MAP is a 2-D array (image) of the model you wish to filter. The image should be smoothed by the MUSTANG beam, which is well approximated[4] by a single 2-D Gaussian with full width, half maximum (FWHM) of $9''$. PIXELSIZE is the (1D) size, assuming the size is the same in each direction. For MUSTANG, all of our maps have $1''$pixels. We recommend using the same pixelization for the application of the transfer function. Finally, FILE will be the string which identifies the (ASCII) text file corresponding to the transfer function (i.e. of a cluster, or the average over all clusters) which you wish to apply.

```
;;; Define variables
pixelsize=1.0                ; Standard size is 1 arcsecond for MUSTANG maps.
                             ; Define this as whatever
```

[4] If you are truly inclined, you may use a double Gaussian: the primary Gaussian with FWHM of $8.7''$ and normalization of 0.94, and the secondary Gaussian with FWHM of $28.4''$ and normalization of 0.06 (see **?**).

```
                                ; your input MAP's pixel size is.
source_size=180.0               ; 120 arcseconds
map_size=size(map,/dim)         ; MAP is the 2D input array (model to be filtered)
nxy=floor(map_size/2)
;set nl to well under the max size of the map
nl=floor(min([(source_size/pixelsize),(min(nxy)-4)]))
kxx=((findgen(2*nl+1,2*nl+1) mod (2*nl+1))-nl)/(2*nl+1)/pixelsize
k=(sqrt(kxx^2+transpose(kxx)^2))
index=floor(k*(2*nl+1)*pixelsize+0.5)
```

Here, we list the code for reading in the transfer function separately. We strongly encourage you to check that your variable K is the same as the variable K_ARRAY (in the context of the code provided here). Assuming that they are, you can proceed to applying the transfer function.

```
;;; Read in the appropriate transfer function (in FILE)
readcol,file,k_array,transfer_function,format='(F,F)'

;;; Apply transfer function
myfftmap=(fft(map[nxy[0]-nl:nxy[0]+nl,nxy[1]-nl:nxy[1]+nl],/center))
for i=0,nl do begin
  loc=where(index eq i)
  myfftmap[loc]=myfftmap[loc]*transfer_function[i]
endfor
mynewmap=map
mynewmap[nxy[0]-nl:nxy[0]+nl,nxy[1]-nl:nxy[1]+nl]=$
  real_part(fft(myfftmap,/inverse,/center))
```