Confidence Maps - False Discovery Rate Control of cryo-EM maps

Maximilian Beckers and Carsten Sachse
EMBL Heidelberg/Forschungszentrum Juelich

2019

1 Introduction

Confidence Maps are complementary maps generated from cryo-EM maps by means of statistical hypothesis testing and subsequent FDR control. They allow thresholding of EM maps based on the expected amount of background noise visible at the respective threshold and thus allow rigorous error assessment of visible features in the density. Additional post-processing like local filtering or local amplitude scaling (LocScale) can be incorporated in the framework in order to increase the statistical power.

2 Getting Started

2.1 Prerequisites

The software is written in Python and only dependent on NumPy and Matplotlib libraries. Both Python2.7 and Python3 are supported.

2.2 Installing

The folder that contains the whole software has only to be copied to your computer and can then be run from the folder with your Python installation with

```
python FDRcontrol.py -em yourMap.mrc
```

2.2.1 Installation using Git

Alternatively, you can just clone the repository to your local machine with:

```
git clone https://git.embl.de/mbeckers/FDRthresholding.git
```

2.3 Input and Output

2.3.1 Input

Confidence Maps can in general be generated from any unmasked EM map, as long as background noise can be estimated. It will work on sharpened/unsharpened and/or filtered/unfiltered maps. The most informative case might be on sharpened and filtered maps. Locally filtered and/or locally sharpened maps can not be used as input, as local noise levels have to be taken into account. How to incorporate local resolution information, local filtering and LocScale is explained in the next section.

2.3.2 Output

The Confidence Map will have the name your_input_map_confidenceMap.mrc. It contains values between 0 and 1 and can be visualized in any dedicated visualization software (e.g. Chimera, Coot, ...). E.g., thresholding a FDR-controlled Confidence Map at 0.99 means a false discovery rate of 1 %, i.e. that less than 1 % of all pixels seen at this threshold are expected to be background noise.

3 How to generate a Confidence Map

In order to do the tutorial, please download all maps from

`https://oc.embl.de/index.php/s/5GTptSDNFskvYuF`

3.1 Confidence Map generation without additional information

The simplest, and probably most important case, is the generation of a Confidence Map from a simple cryoEM density without incorporation of local resolution or atomic model information.

```
python FDRcontrol.py -em yourMap.mrc
```

The output will be the corresponding Confidence Map and a diagnostic image, that shows three slices through the map together with the regions used for noise estimation. In order to get good estimates of the background noise distribution, you should make sure that the region contains just noise and no signal of the particle.

Size of the noise estimation region can be adjusted with -w sizeOfRegion. If the default regions for noise estimation fall on top of the molecule, you can specify the center of region of your choice with -noiseBox x y z. For example:

```
python FDRcontrol.py -em yourMap.mrc -w 20 -noiseBox 50 50 120
```

Boxes for noise estimation should be made as big possible in order to get precise and reliable background noise estimates.

3.1.1 Demonstration with TRPV1 EMD5778

We will now demonstrate the procedure using the 3.4 Angstrom structure of TRPV1 (EMD5778, Liao et al. 2013).
If you didn't download the map so far, then download it from

```
https://oc.embl.de/index.php/s/5GTptSDNFskvYuF
```

We are already ready to generate the Confidence Map by

```
python FDRcontrol.py -em /path/to/TRPV1_sharpened_-100_3.4A.map
```

If we have a look in the diagnostic image, we see that we can increase the box sizes for background noise estimation, which should make the estimates more precise. Let's choose 50 and generate a new Confidence Map with

```
python FDRcontrol.py -em /path/to/TRPV1_sharpened_-100_3.4A.map -w 50
```

Don't forget to set the right path! The expected run time on normal desktop computer should be around 1-2 minutes for this example. We can have look at the Confidence Map with Chimera:

```
chimera TRPV1_sharpened_-100_3.4A_confidenceMap.mrc
```

3.2 Incorporation of local resolution information

A corresponding local resolution map can be supplied to the program by -locResMap yourLocalResolutionMap.mrc. The ouput will be a locally filtered map together with the diagnostic image and the Confidence Map. Adjustment of the noise estimation region can be done accordingly. Example usage:

```
python FDRcontrol.py -em yourMap.mrc -locResMap yourLocalResolutionMap.mrc
```

3.2.1 Turorial with beta-Galactosidase EMD2984

As above, maps are available from the same link as for the example given above.
Let's generate a first Confidence Map by

```
python FDRcontrol.py -em /path/to/EMD-2984.map -locResMap /path/to/betaGal_resmap.mrc
```

This calculation might take a few minutes, (ca 10 - 20 minutes, depending on the size of your map). So now lets have a look in diag_image.pdf.

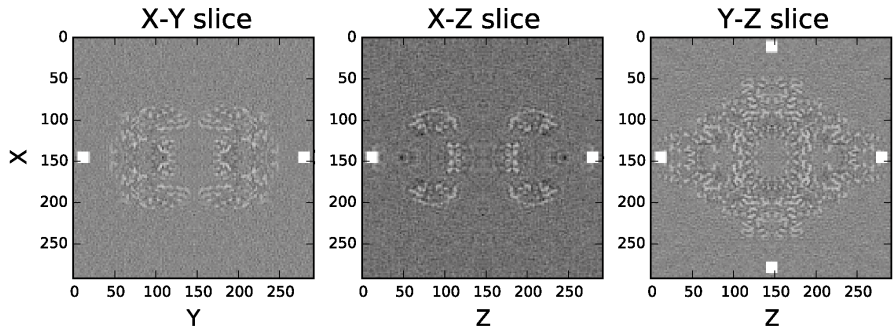

Figure 1: diag_image.pdf. The third image on the right clearly shows that noise boxes fall into the molecule.

We see that the noise estimation regions clearly falls into molecule density, so we have to set a new noise estimation region. From the coordinate systems given in diag_image.pdf, we can see that the box center x = 50, y=150, z=150 and a box size of 50 should give a reasonable region. So we run it again:

```
python FDRcontrol.py -em /path/to/EMD-2984.map
-locResMap /path/to/betaGal_resmap.mrc -noiseBox 50 150 150 -w 50
```

Again we check diag_image.pdf. It shows that the noise region was set to a reasonable region and 50 is quite a large sidelength of the box, which allows precise background noise estimation.

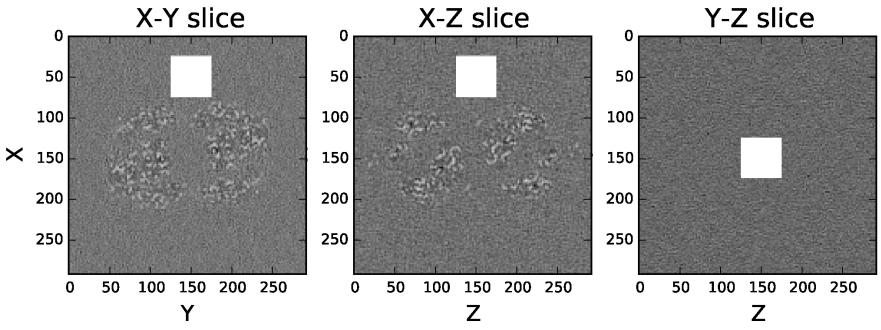

Figure 2: diag_image.pdf with new box coordinates. Box has reasonable size and is outside of the molecular density.

Again, we can have look at the Confidence Map with Chimera:

```
chimera EMD2984_confidenceMap.mrc
```

The additonal map generated is the locally filtered version and will have the name `EMD2984_locFilt.mrc`.

3.3 Incorporation of local amplitude scaling

LocScale inside FDRcontrol.py can be used by supplying a model map with -mm youModelMap.mrc, where the model map is generated in the LocScale workflow. The window size of the rolling window for local sharpening can be specified with -w_locscale windowSize. Size of the box for background noise estimation with -w boxSize and its location again with -noiseBox x y z. To speed up everything, a stepsize can be specified with -stepSize, which skips the specified amount of voxels between each scaling. This leads to massive speedups and no need for parallelisation, default is a stepsize of 5. The ouput will be the locally scaled map together with the diagnostic image and the Confidence Map. For usage of parallelisation, you will need mpi4py and run the application with mpirun with the -mpi flag.

Example usage:

```
python FDRcontrol.py -em yourMap.mrc -mm yourModelMap.mrc
```

3.3.1 Turorial with beta-Galactosidase EMD2984

Using the same parameters for boxsize and box location of the noise window as before, we can run LocScale together with the generation of a Confidence Map by

```
python FDRcontrol.py -em /path/to/betaGal.mrc -noiseBox 50 150 150
-w_locscale 15 -mm /path/to/pdb5a1a_rsref_NCS_4locscale.mrc
```

Here we used a window size of 15 pixels for the LocScale sliding window; the window size for background noise estimation will have the same size. Please note that instead of the sharpened map EMD2984.map we input here the unsharpened and unfiltered map betaGal.mrc. We can accelerate the procedure even more by choosing a bigger stepsize, e.g. 7, by

```
python FDRcontrol.py -em /path/to/betaGal.mrc -noiseBox 50 150 150
-w_locscale 15 -stepSize 7 -mm /path/to/pdb5a1a_rsref_NCS_4locscale.mrc
```

The stepsize cannot be bigger than the window size.

3.4 Working with non-normal background noise

The program gives you some diagnostics about the deviation from a normal distribution. For example, something like

```
Checking the normal distribution assumption ...
Maximum Distance Dn between ECDF and CDF: Dn=0.0067647795847524117,
in Tail:  0.0011355796488379255.  Sample size used:  62500
Anderson-Darling test summary:  0.883163821648978.  p-Value:  0.023902441731938867.
Sample size used:  1000
```

The first quantities are the maximum difference that occur between the cumulative distribution function of the estimated normal distirbution with the empirical cumulative distribution function. If the difference in the tail area is bigger than 0.01, the program gives you a warning like

```
WARNING: Deviation in the tail areas between the normal distribution
and the empircal CDF is higher than 1%.  If boxes for background noise
estimation are set properly, please consider using the flag -ecdf to
use the empirical CDF instead of the normal distribution.
```

The second diagnostic, the Anderson-Darling test, is a statistical hypothesis test that tests the null hypothesis of a normal distribution of the respective sample. Based on the p-value it allows rejection of normality. However, this kind of tests tend to detect trivial deviations from normality, i.e. small deviations that will not have any effect on the final inference. This is why we use the difference between CDF and ECDF as our favourite criterion.
If you want to switch from the assumption of a normal distribution to a nonparametric procedure, you can calculate p-values based on the empirical cumulative distribution function (ECDF). You can set the respective option with `-ecdf`. For example with TRPV1 the command will look like

```
python FDRcontrol.py -em /path/to/TRPV1_sharpened_-100_3.4A.map -w 50
-ecdf
```

The disadvantage of using the ECDF is a longer runtime. You should also make sure to have a decent box size that allows accurate estimation of the tail probabilites (i.e. around than 20x20x20). The file ECDF_vs_CDF.pdf shows you a comparison of the ECDF versus the CDF of the normal distribution.

4 Instructions for use and important tips

The programs require an unmasked map as input. Masking will make the noise estimation impossible.

It is critical that the regions used for noise estimation do not fall into the particle of interest. While in single particle analysis (SPA), the data generating process makes choice of the regions straightforward, sub-tomogram-averaged structures usually require specification of this region from the user.
The region can be identified by means of the slice-views in the diagnostic image of a first run with default parameters, and then adjust center and width of the noise region for a subsequent run. The axes in the diagnostic image are labelled with the voxel indices.
Always have a look in the diagnostic image. After a first run you should use the information from the diagnostic image to increase the box size for noise estimation.
If your map contains large background areas, i.e. small particle compared to the box size, you should increase the sizes of the noise estimation regions in order to get more accurate estimates of the background noise distribution. **The box size can be supplied with -w yourBoxSize.**

Recommended workflow:
1. Run the FDR control on your map
2. Have a look in the diagnostic image and check the noise estimation. Make sure the marked regions do not fall into your molecule and identify reasonable size possible for correct background noise estimaion and/or identify new box coordinates.
3. Rerun the FDR control with the optimised noise estimation regions. Make the regions as big as possible. Size can be set with -w and the box, if it has to be adjusted, with -noiseBox xCoord yCoord zCoord

Be aware, inclusion of either local resolution and/or atomic model information depend on the correctness of this prior information. Make sure orientations of model maps and/or local resolution maps with respect to the input EM-map are correct.

As Confidence Maps are almost binary and have huge contrast, they might look quite sharp and edged. For visualization the representation can be easily improved, for example in Chimera by turning on surface smoothing in the Volume Viewer in Features `-->` Surface and Mesh options.

5 FAQ

1. My Confidence Map doesn't show expected high resolution details
Confidence maps are not invariant to B-factor sharpening and filtering. Oversharpening will lead to increased background noise levels; Confidence Maps take care of that and oversharpened regions will be simply calssified as noise. Undersharpening will be visible in confidence maps though the absence of high resolution details. So you can simply try to increase the sharpening and generate upadted Confidence Maps, which will help avoiding typical oversharpening issues (e.g. interpretation of noise as signal).

2. My confidence map shows unspecific, noise-like signal at low FDR
First of all make sure, your background noise estimation did work properly by checking the diagnostic image. If your noise estimation region is to small or when you estimate noise from masked regions, your background noise estimates might be inaccurate. However, highly flexible or completely disordered regions can appear noise-like in Confidence Maps; especially when using masked refinement this effect might occur. These densities are indeed significant and truly there, however, care has to be taken not to interpret them as ions or water. Incorporation of local resolution information with -locResMap yourLocalResolutionMap.mrc might help in this case.

3. Parts of the molecule seem to be missing in the Confidence Map
If parts of your molecule (loops, ligands, etc.) are missing in the Confidence Map at low FDR, these might be due to oversharpening. An easy solution that almost always helps in this situation is incorporation of local resolution information with -locResMap yourLocalResolutionMap.mrc.

4. Incorporation of local resolution information seem to introduce artifacts in the confidence map
If the local resolution map contains low resolution artifacts (e.g. when estimated from masked half maps), these might introduce smeared out densities. In this case it can help to cut the lowest resolutions in your local resolution map to the lowest expected value (e.g. 10 Angstroms).

6 List of options

6.1 Required parameters

- -em EM map Example: -em yourEMmap.mrc

6.2 Optional parameters

- -p pixel size Example: -p 1.06
- -locResMap local resolution map for local filtering Example: -locResMap yourLocalResolutions.mrc
- -mm model map for LocScale Example: -mm yourModelMap.mrc
- -w window size for background noise estimation Example: -w 50
- -noiseBox box coordinates for background noise estimation Example: -noiseBox 50 150 150
- -method: method for error control, 'BH' for Benjamini-Hochberg FDR-control, 'BY' for Benjamini-Yekutieli FDR-control (default) or 'Holm' for Holm FWER-control
- -w_locscale window size for LocScale Example: -w_locscale 15
- -stepSize step size for LocScale Example: -stepSize 5 (default)
- -ecdf use ECDF instead of Gaussian CDF for p-value calculation Example: -ecdf
- -halfmap2 If you want to give two halfmaps instead of one averaged map, give here the second halfmaps and the first halfmap with -em Example: -halfmap2 half2_unfil.mrc
- -o filename for output Example: -o out.mrc
- -testProc use right-, left- or two-sided testing i.e. 'rightSided', 'leftSided'or 'twoSided' Example: -testProc 'rightSided'
- -fdr threshold confidence map at the desired FDR Example: -fdr 0.01
- -mpi MPI parallelisation for LocScale Example: -mpi
- -lowPassFilter resolution in Angstrom to low-pass filter the map before multiple testing Example: -lowPassFilter 5