From PIPSA to MultiPIPSA

Stefan Richter

HITS gGmbH

Feb. 28th, 2019

What is PIPSA

- Protein Interaction Property Similarity Analysis
- Method developed by Blomberg N, Gabdoulline RR, Nilges M, and Wade 1999.
- Compares interaction fields (electrostatic potential) on superimposed structures and calculates heatmap or cluster tree
- Initially available as commandline tool, since 2007 also as webserver: http://pipsa.h-its.org
- Now presented here with extensions as a python module and as you see later by Lukas used in a jupyter notebook.

What is PIPSA

- Protein Interaction Property Similarity Analysis
- Method developed by Blomberg N, Gabdoulline RR, Nilges M, and Wade 1999.
- Compares interaction fields (electrostatic potential) on superimposed structures and calculates heatmap or cluster tree
- Initially available as commandline tool, since 2007 also as webserver: http://pipsa.h-its.org
- Now presented here with extensions as a python module and as you see later by Lukas used in a jupyter notebook.

What is PIPSA

- Protein Interaction Property Similarity Analysis
- Method developed by Blomberg N, Gabdoulline RR, Nilges M, and Wade 1999.
- Compares interaction fields (electrostatic potential) on superimposed structures and calculates heatmap or cluster tree
- Initially available as commandline tool,
  since 2007 also as webserver: http://pipsa.h-its.org
- Now presented here with extensions as a python module and as you see later by Lukas used in a jupyter notebook.

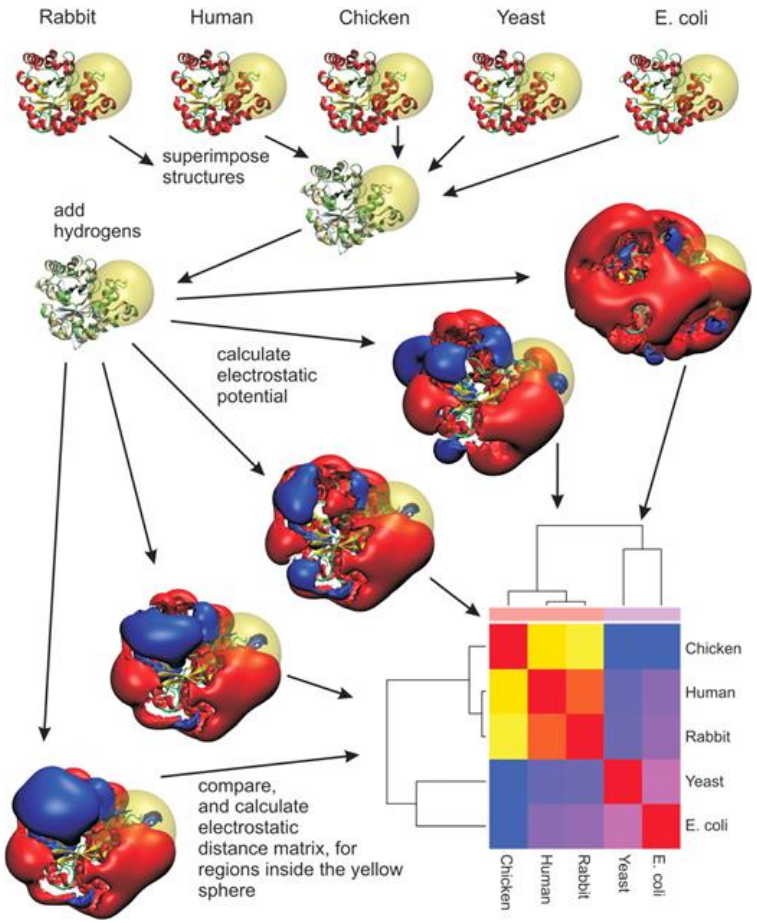
What is PIPSA

- Protein Interaction Property Similarity Analysis
- Method developed by Blomberg N, Gabdoulline RR, Nilges M, and Wade 1999.
- Compares interaction fields (electrostatic potential) on superimposed structures and calculates heatmap or cluster tree
- Initially available as commandline tool,
  since 2007 also as webserver: http://pipsa.h-its.org
- Now presented here with extensions as a python module and as you see later by Lukas used in a jupyter notebook.

What is PIPSA

- Protein Interaction Property Similarity Analysis
- Method developed by Blomberg N, Gabdoulline RR, Nilges M, and Wade 1999.
- Compares interaction fields (electrostatic potential) on superimposed structures and calculates heatmap or cluster tree
- Initially available as commandline tool,
  since 2007 also as webserver: http://pipsa.h-its.org
- Now presented here with extensions as a python module and as you see later by Lukas used in a jupyter notebook.

PIPSA example



Stefan Richter, Anne Wenzel, Matthias Stein, Razif R. Gabdoulline, Rebecca C. Wade; Nucleic Acids Research, 2008

Prerequisites

- Python 2 or 3 is possible, ideally use anaconda (see next slide).
- Download multipipsa distribution, unpack
- Installing multipipsa

```
conda activate mypipsa
python setup.py install
```

Prerequisites

- Python 2 or 3 is possible, ideally use anaconda (see next slide).
- Download multipipsa distribution, unpack
- Installing multipipsa

```
conda activate mypipsa
python setup.py install
```

Prerequisites

- Python 2 or 3 is possible, ideally use anaconda (see next slide).
- Download multipipsa distribution, unpack
- Installing multipipsa

```
conda activate mypipsa
python setup.py install
```

Use of Anaconda

Useful commands

```
   # Setting addition sources for packages in conda
conda config --add channels bioconda
   # Create an environment that uses python 3.4 as default
conda create -n myenv python=3.4
   # Install the scipy package with a give version number
conda install -n myenv scipy=0.15.0
   # make all packages/interpreters installed in myenv available
conda activate myenv
   # Create a new environment based on an existing environment
conda create --name myclone --clone myenv
   # Write out the information about a specific environment to a file
conda list --explicit > spec-file.txt
   # Create an environemnt based on a file
conda create --name myenv --file spec-file.txt
   # Install into the current environement all packages from a file
conda install --name myenv --file spec-file.txt
   # List available environments from a conda installation
conda info --envs
   # Get rid of the conda environment
conda deactivate
```

Example from `multipipsa/testMultipipsa.py`

Make sure, your structures are

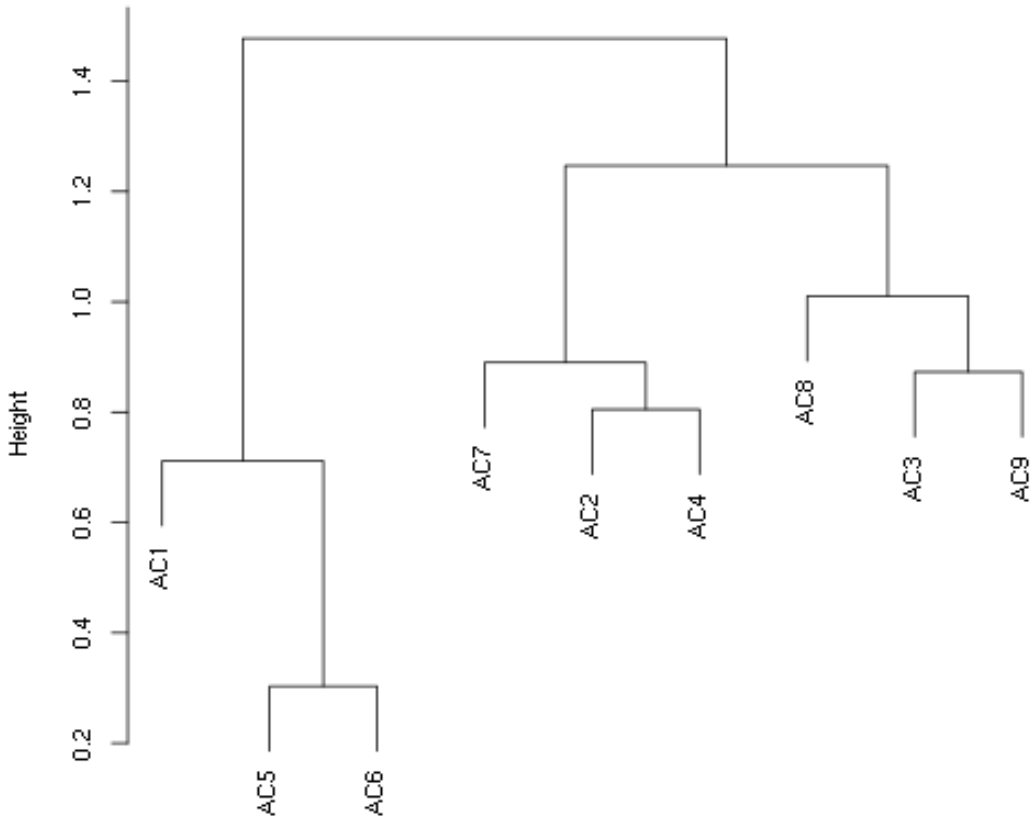
Running APBS

```
1 ingrp  = ["AC1", "AC5", "AC6"]
2 outgrp = ["AC2", "AC3", "AC4", "AC7", "AC8", "AC9"]
3 apbs = ApbsRun(
4      dataDir=os.getcwd()+"/example/pdbs",
5      temp=298.15,ios=0.100,structures=ingrp+outgrp)
6 apbs.runPdb2Pqr()
7 apbs.runApbs()
```

Standard PIPSA

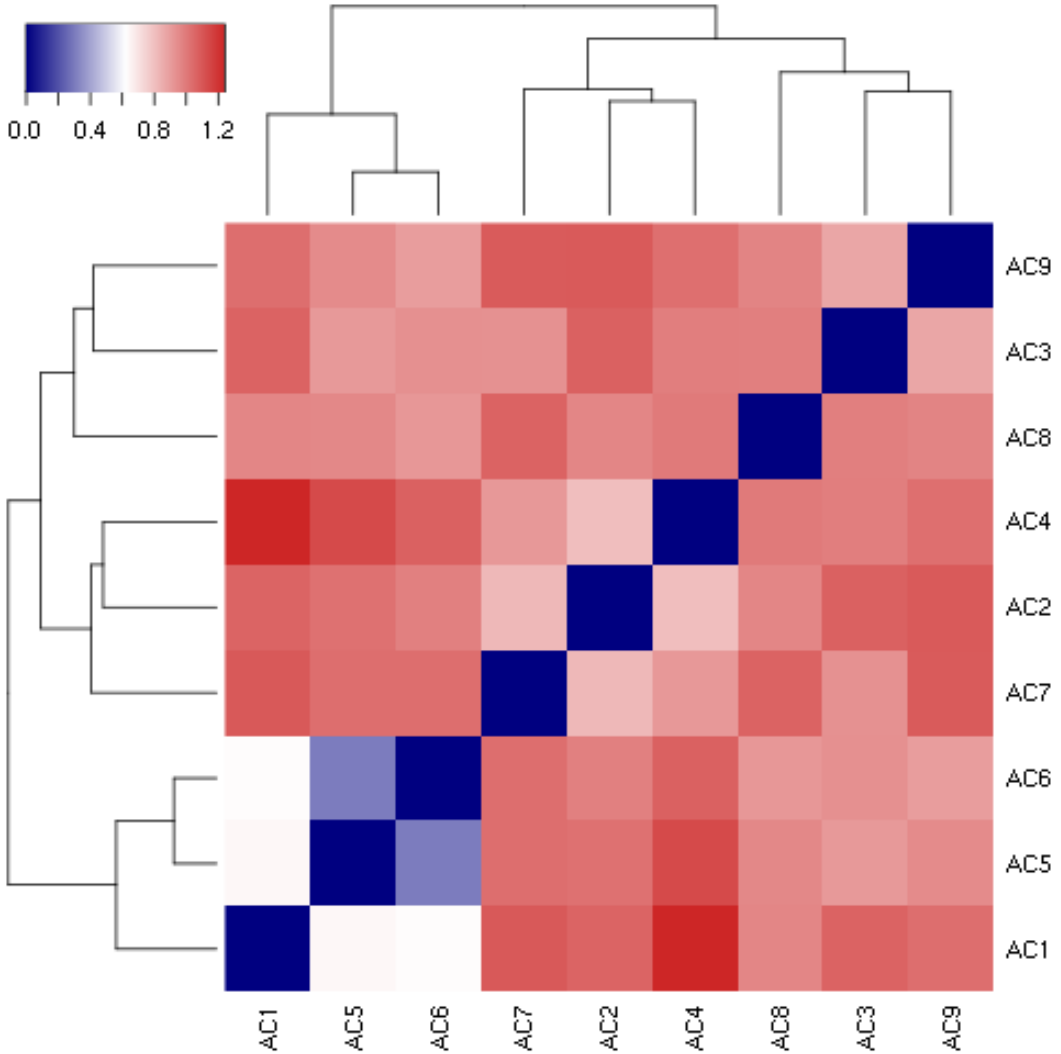
```
pr.runPipsa(ingrp+outgrp)
#########################################################
cl0 = ClusterPipsa(structures=ingrp+outgrp,pipsaRoot="../../pipsa",
          dataDir=os.getcwd()+"/example/pdbs",
          distanceType=DistanceType.PIPSA,
          graphicsFileRoot="clust0")
pr.runClusterPipsa(ingrp+outgrp,
          points=[],cluster=cl0)
#########################################################
cl1 = ClusterPipsa(structures=ingrp+outgrp,pipsaRoot="../../pipsa",
          dataDir=os.getcwd()+"/example/pdbs",
          distanceType=DistanceType.PIPSA,
          graphicsFileRoot="clust1")
pr.runClusterPipsa(ingrp+outgrp,
          points=pr.getStructure().getCAAtoms(),cluster=cl1)
#########################################################
cl2 = ClusterPipsa(structures=ingrp+outgrp,pipsaRoot="../../pipsa",
                   dataDir=os.getcwd()+"/example/pdbs",
                   distanceType=DistanceType.PIPSA,
                   graphicsFileRoot="clust2")
pr.runClusterPipsa(ingrp+outgrp,
          points=pr.getStructure().getCAAtoms()[10],cluster=cl2)
```

Standard PIPSA – Results



structure(c(0.641872261435248, 0.627694193059009, 1.06301458127346,
1.06677082824757, 1.24177292610203, 1.09361784915939, 0.965401470891774,

Standard PIPSA – Results



Example from `multipipsa/testMultipipsa.py`

Calculate Binding Scores

Reproduces the Fig. 5 of the binding scores of the reference in the footnote below.

```
1 pr = PipsaRun(pipsaRoot="../../pipsa",
2      dataDir=os.getcwd()+"/example/pdbs",
3      pointsTemplate="AC5")
4 pr.runBindingScorePipsa(ingrp, outgrp)
5 pr.savePDBResult(filename="SimHodgkin.pdb",
6      scoreType=ScoreType.MS,
7      similarityType=SimilarityType.HODGKIN)
```

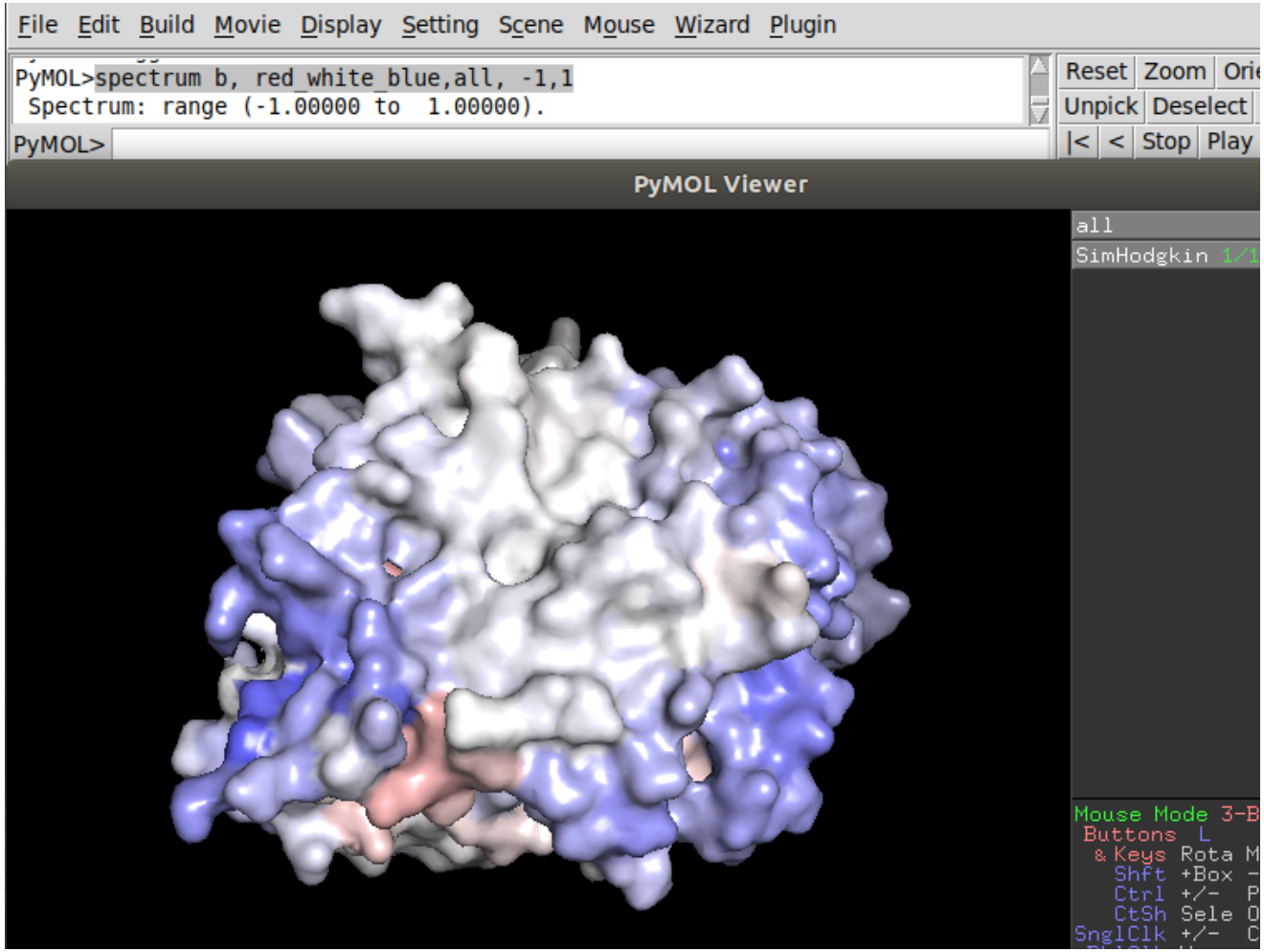
Tong, Wade and Bruce, Comparative electrostatic analysis of adenylyl cyclase for isoform dependent regulation properties., Proteins 2016; 84:1844-1858; doi: 10.1002/prot.25167.

Score is saved in B-Factor column

Excerpt from SimHodgkin.pdb file of previous example

```
1  ATOM   2364  CZ  PHE B 300      38.312  -9.485  36.510  0.00 0.20           C
2  ATOM   2365  N   GLN B 301      44.946  -8.363  34.106  0.00 0.38           N
3  ATOM   2366  CA  GLN B 301      46.308  -8.562  33.712  0.00 0.38           C
4  ATOM   2367  C   GLN B 301      46.943  -9.327  34.831  0.00 0.38           C
5  ATOM   2368  O   GLN B 301      46.523  -9.218  35.981  0.00 0.38           O
6  ATOM   2369  CB  GLN B 301      47.086  -7.243  33.560  0.00 0.38           C
7  ATOM   2370  CG  GLN B 301      46.532  -6.335  32.460  0.00 0.38           C
8  ATOM   2371  CD  GLN B 301      47.373  -5.066  32.412  0.00 0.38           C
9  ATOM   2372  NE2 GLN B 301      47.920  -4.745  31.209  0.00 0.38           N
10 ATOM   2373  OE1 GLN B 301      47.539  -4.378  33.417  0.00 0.38           O
11 ATOM   2374  N   MET B 302      47.972 -10.137  34.521  0.00 0.44           N
12 ATOM   2375  CA  MET B 302      48.602 -10.927  35.540  0.00 0.44           C
13 ATOM   2376  C   MET B 302      49.886 -10.255  35.926  0.00 0.44           C
```

Visualization in pymol



Select surface representation in pymol and enter the command

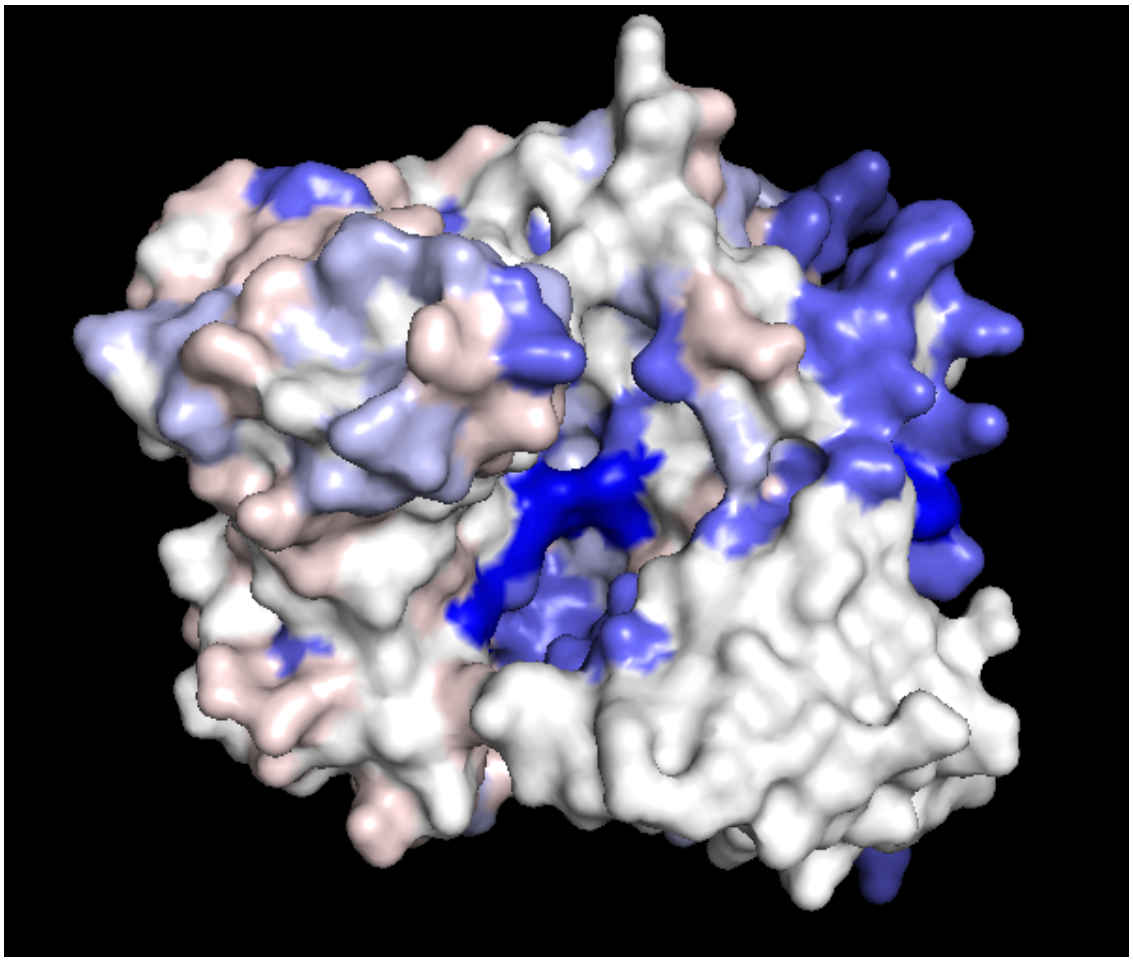
`spectrum b, red_white_blue, all, -1,1`

Standard PIPSA – Compare Result to given cluster

Standard PIPSA

```
1 referenceCluster = np.array([1,1,1,2,2,2,2,2,2])
2 cl = ClusterPipsa(structures=ingrp+outgrp,pipsaRoot="../../pipsa",
3                    dataDir=os.getcwd()+"/example/pdbs",
4                    referenceCluster=referenceCluster,
5                    distanceType=DistanceType.PIPSA,
6                    graphicsFileRoot="clust")
7 pr.runClusterPipsa(ingrp+outgrp,cluster=cl)
8 pr.savePDBResult(filename="ClusterCorrectedRand.pdb",scoreType=
    ScoreType.CORRECTED_RAND,
9                  clusterObject=cl)
```

Standard PIPSA – Areas with very similar clusters as the reference cluster



Standard PIPSA – Areas with high or low information content in the clustering