Università
della
Svizzera
italiana

**Institute of
Computing
CI**

High-Performance Computing Lab **Institute of Computing**

Student: Alessio Barletta

Solution for Project 6

HPC Lab — Submission Instructions
(Please, notice that following instructions are mandatory: submissions that don't comply with, won't be considered)

- Assignments must be submitted to iCorsi (i.e. in electronic format).
- Provide both executable package and sources (e.g. C/C++ files, Matlab). If you are using libraries, please add them in the file. Sources must be organized in directories called:
  Project_number_lastname_firstname
  and the file must be called:
  project_number_lastname_firstname.zip
  project_number_lastname_firstname.pdf
- The TAs will grade your project by reviewing your project write-up, and looking at the implementation you attempted, and benchmarking your code's performance.
- You are allowed to discuss all questions with anyone you like; however: (i) your submission must list anyone you discussed problems with and (ii) you must write up your submission independently.

1. Task: Install METIS 5.0.2, and the corresponding Matlab mex interface

The output from the installation demo is shown below:

```
>> A = blkdiag(ones(5),ones(5));
>> A(1,10) = 1; A(10,1) = 1; A(5,6) = 1; A(6,5) = 1;
>> [p1,p2] = bisection_metis(sparse(A),0,0)

p1 =

     6     7     8     9    10


p2 =

     1     2     3     4     5

>>
```

2. Task: Construct adjacency matrices from connectivity data [10 points]

In this tasks we are asked to compute the adjacency matrix from the connectivity data of a mesh. We have as input two .csv files, containing respectively the nodes coordinates and the data to construct the adjacency matrix.

The oputput of this function is the graph data saved in a .mat file saved under *Datasets/Countries_mat/*.

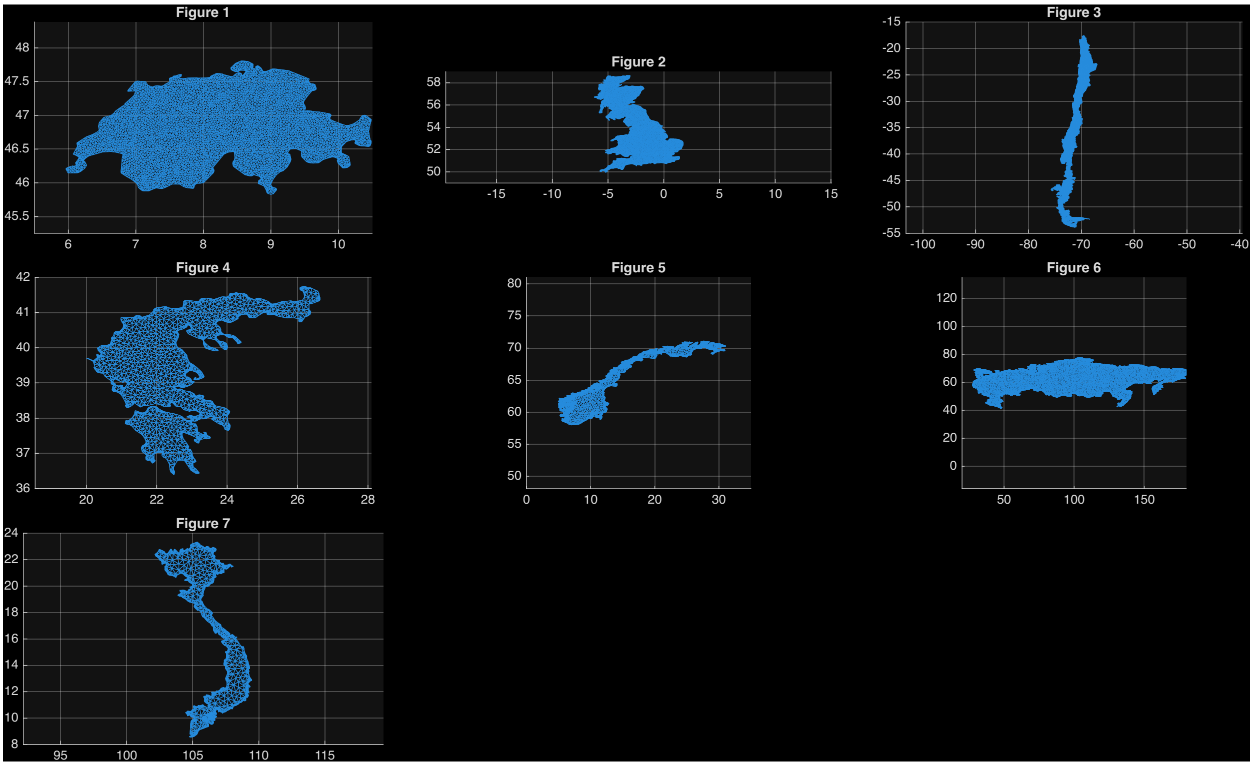

Figure 1: Visualization of the proposed countries graph `gplotg.m`.

3. Task: Implement various graph partitioning algorithms [25 points]

In this task we are asked to implement two graph partitioning algorithms: Spectral and Inertial partitioning.

- **Spectral partitioning:** we compute the Fiedler vector (the one associated with the second smallest eigenvalue) of the graph Laplacian matrix L and we use its sign to partition the graph into two subgraphs. I also applied a spectral shift of $\sigma = -10^{-6}$ to resolve the singularity of the Laplacian matrix at zero, to ensure numerical stability and avoid singularity issues while calculating the smallest eigenvalues.

- **Inertial partitioning:** here we compute the center of mass of all the points in the graph and then divide them using a line passing through the center of mass such that the distance from each point to the center of mass is minimized. To do this, I followed the method provided in the project description. One important note is that I had to rotate the vector that I passed to the `partition.m` function beacuse this expects a vector with the direction normal to the partitioning plane.

Results for the bisection of various meshes are reported in Table 1.

Table 1: Bisection results

Mesh	Coordinate	Metis 5.0.2	Spectral	Inertial
mesh1e1	18	17	18	19
mesh2e1	37	37	35	37
mesh3e1	19	19	24	19
mesh3em5	19	19	24	19
airfoil1	94	77	132	124
netz4504_dual	25	23	23	39
stufe	16	16	16	40
3elt	172	124	117	182
barth4	206	97	127	253
ukerbe1	32	27	32	88
crack	353	201	233	323

4. Task: Recursively bisecting meshes [15 points]

Here we make use of the function `rec_bisection.m` to recursively partition the graph in power-of-two number of partitions. What we had to do is to implement the function `bench_rec_bisection.m` to make calls to `rec_bisection.m` using various partitioning methods.

In the table below we report the edge-cut results for recursive bi-partitioning into 16 partitions.

Table 2: Edge-cut results for recursive bi-partitioning (p=16).

Mesh	Coordinate	Metis 5.0.2	Spectral	Inertial
airfoil1	819	563	631	903
netz4504_dual	198	161	183	202
stufe	227	194	246	267
3elt	1168	651	752	1342
barth4	1306	689	835	1348
ukerbe1	374	224	878	470
crack	1860	1290	1419	1618

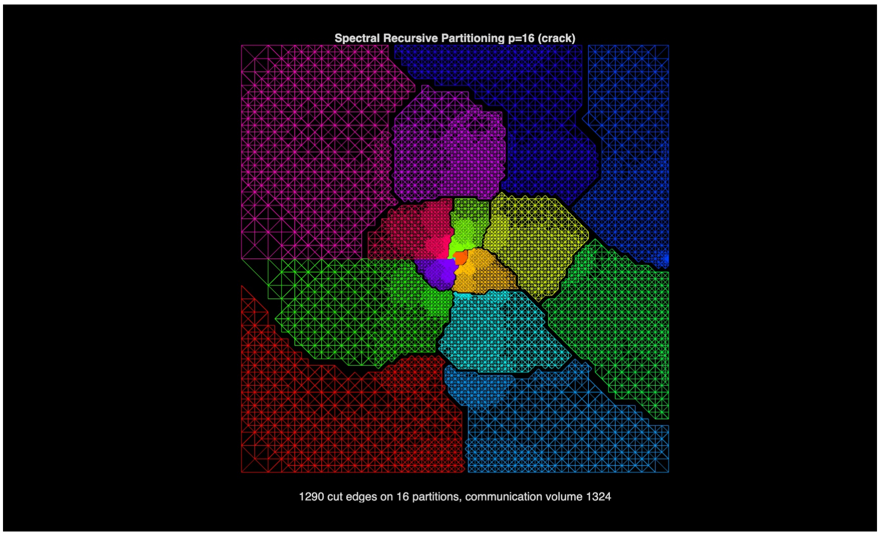

Figure 2: Visualization of the proposed countries graph `gplotg.m`.

5. Task: Comparing recursive bisection to direct k-way partitioning [10 points]

Here we use metismex to perform direct multiway partitioning and compare the results with those obtained from recursive bisection.

We had to finish the code provided in `bench_metis.m` to perform the comparison, doing the command `help metismex` we could understand how to call metismex to make either recursive bisection or direct multiway partitioning.

In the results we can observe how the k-way partitioning always produces slightly less cut edges than the recursive bisection, giving better results. This is expected since the recursive method is probably faster but lack some global information that the k-way partitioning can take into account. As a result we can expect better quality partitions from the k-way method, as observed.

In the following table we can observe the difference between the two methods for 16 and 32 partitions, and below that we can see the visualizations of the partitions obtained with both methods for Luxembourg, US roads and Russia graphs.

Table 3: Comparing the number of cut edges for recursive bisection and direct multiway partitioning in Metis 5.0.2.

Partitions	Luxemburg	usroads-48	Greece	Switzerland	Vietnam	Russia	Norway
16 (Rec.)	197	607	297	730	284	616	245
16 (K-way)	170	579	278	673	255	551	245
32 (Rec.)	322	988	509	1089	470	1006	445
32 (K-way)	279	961	471	1042	439	933	411

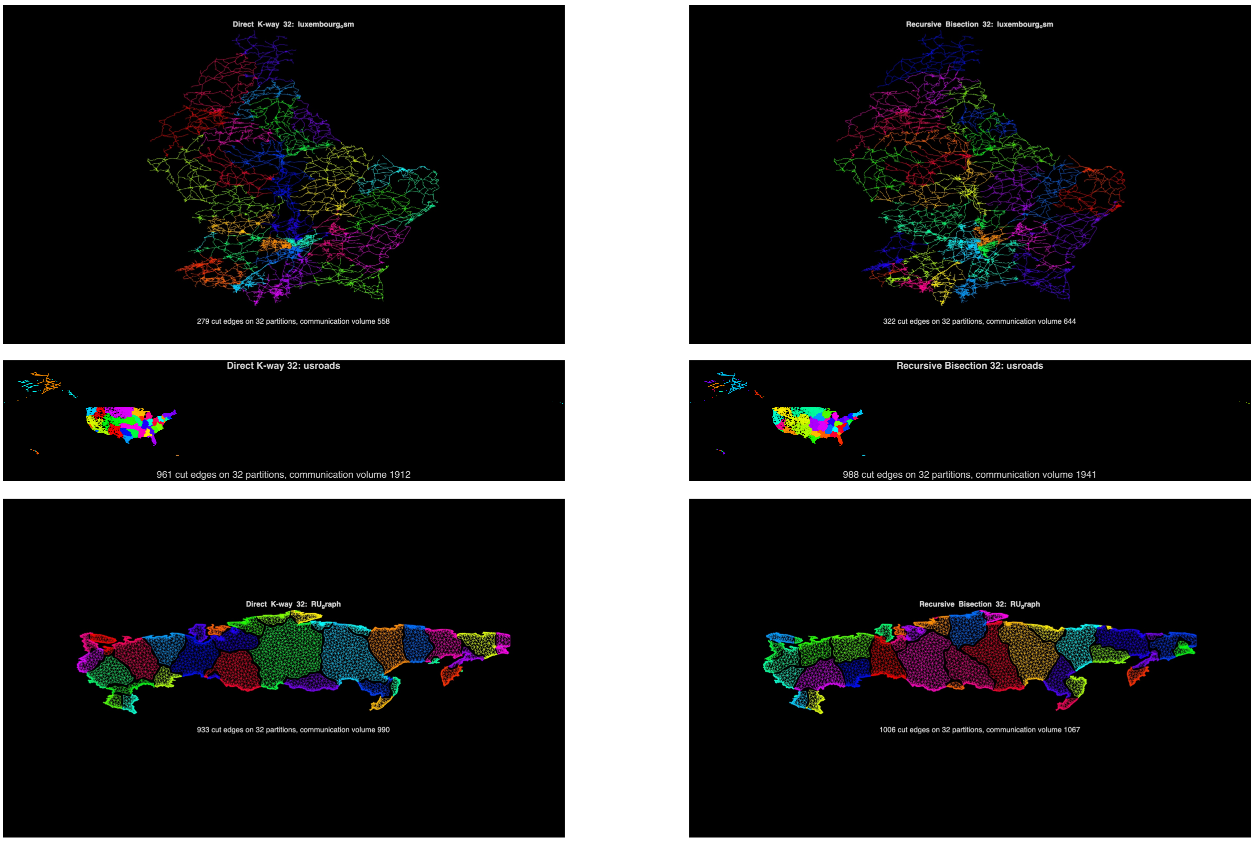

Figure 3: Visualization of partitioning results: Luxembourg (top), US (middle), and Russia (bottom). Left: Direct Multiway; Right: Recursive Bisection.

6. Task: Utilizing graph eigenvectors [25 points]

6.1. Plot the entries of the eigenvectors associated with the first and second smallest eigenvalues of the graph Laplacian matrix L for the graph "airfoil1." Comment on the visual result. Is this behavior expected?

Yes, the uppermost graph represented above corresponds to the eigenvector associated with the smallest eigenvalue of the Laplacian matrix, which is always zero. This eigenvector is constant across all nodes, which is why we see a flat line in the plot. Moreover, also the value of each entry is 0.015 is expected since the eigenvector is normalized, and with 4253 nodes, each entry should be approximately $\frac{1}{\sqrt{4253}} \approx 0.0153$.

The second graph represents the eigenvector associated with the second smallest eigenvalue, known as the Fiedler vector. We know that this vector is orthogonal to the first eigenvector, which implies that all its entry must sum to 0 (because L is symmetric and the entries of the first eigenvector are constant). For this reason we expect to see both positive and negative values in the plot, which is indeed the case.

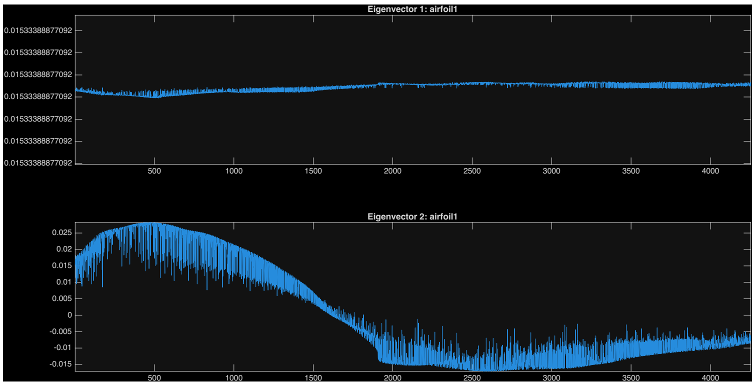

Figure 4: Visualization of Eigenvalues distribution.

The following figures show the plots for some meshes comparing them in spatial coordinates (together with the 3D projection of the associated fielder vector) and the spectral coordinates' bisection (given by the second and third-smallest eigenvectors):

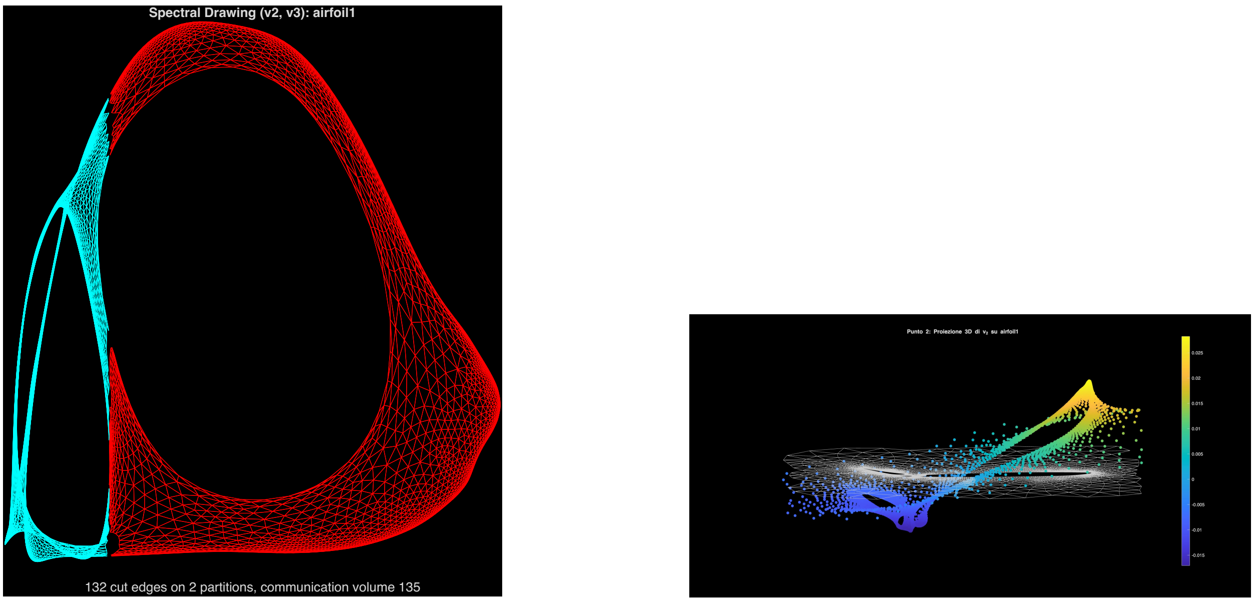

Figure 5: Airfoil: Spectral partitioning (left) and 3D Projection (right).

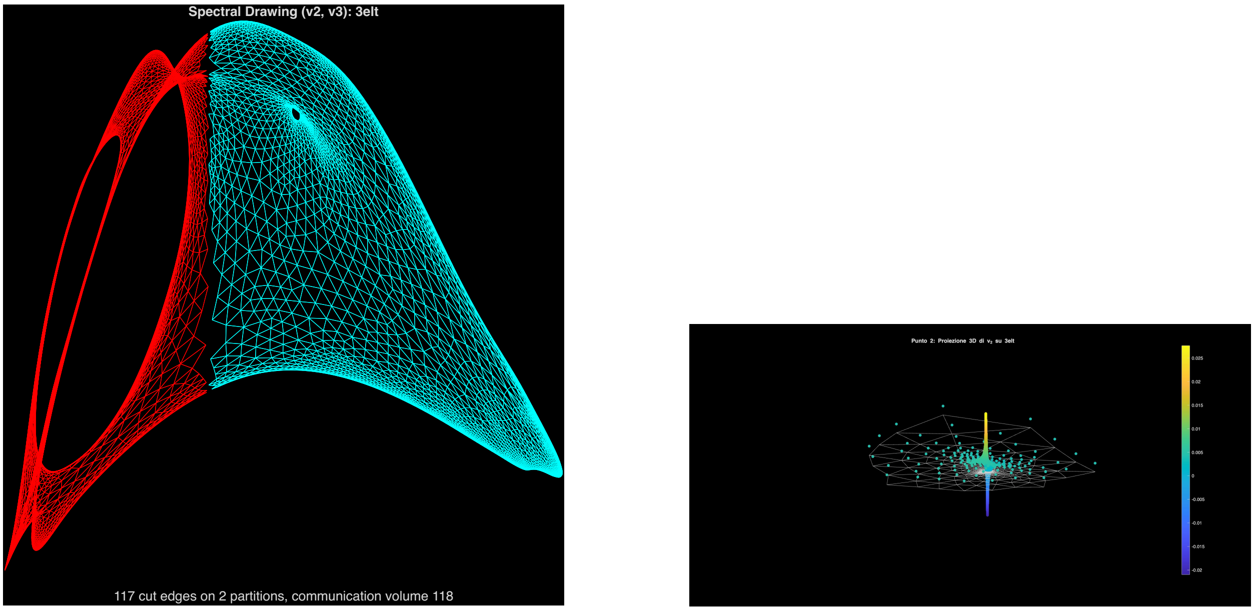

Figure 6: 3elt: Spectral partitioning (left) and 3D Projection (right).

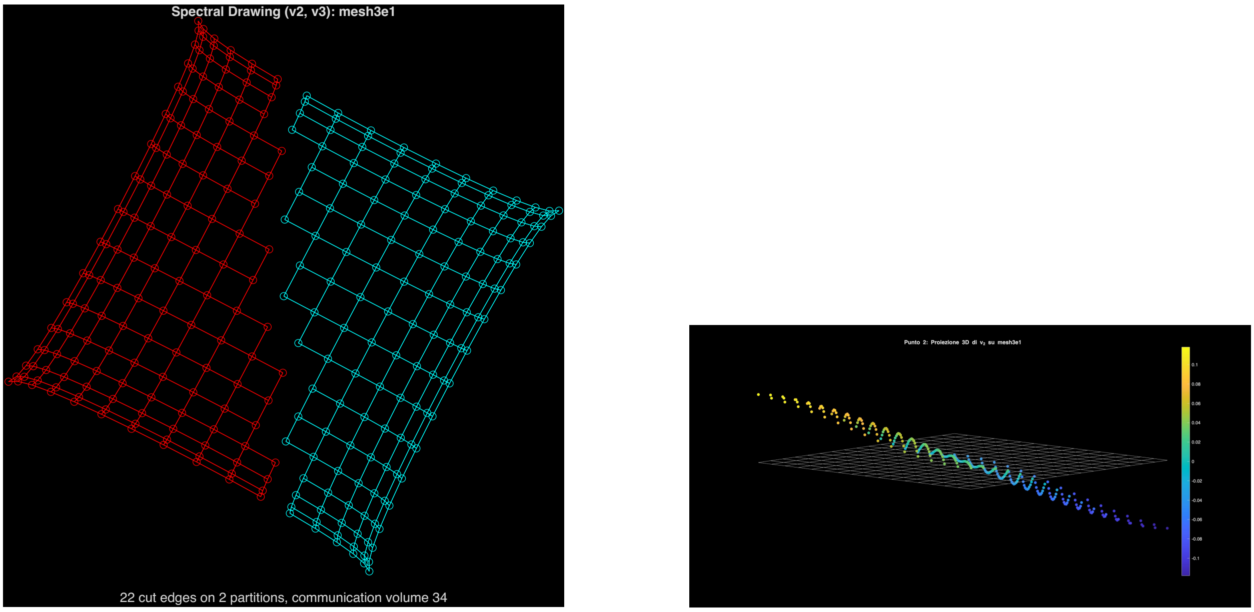

Figure 7: Mesh3: Spectral partitioning (left) and 3D Projection (right).

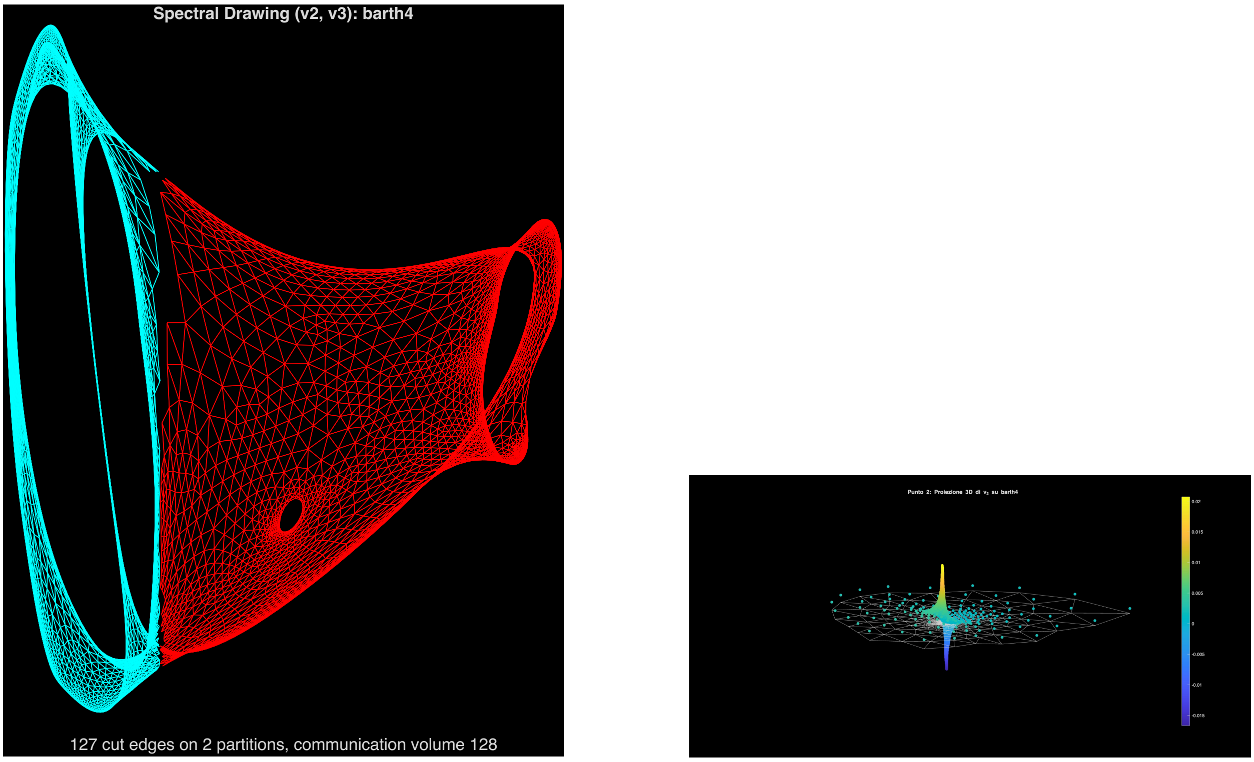

Figure 8: Barth4: Spectral partitioning (left) and 3D Projection (right).

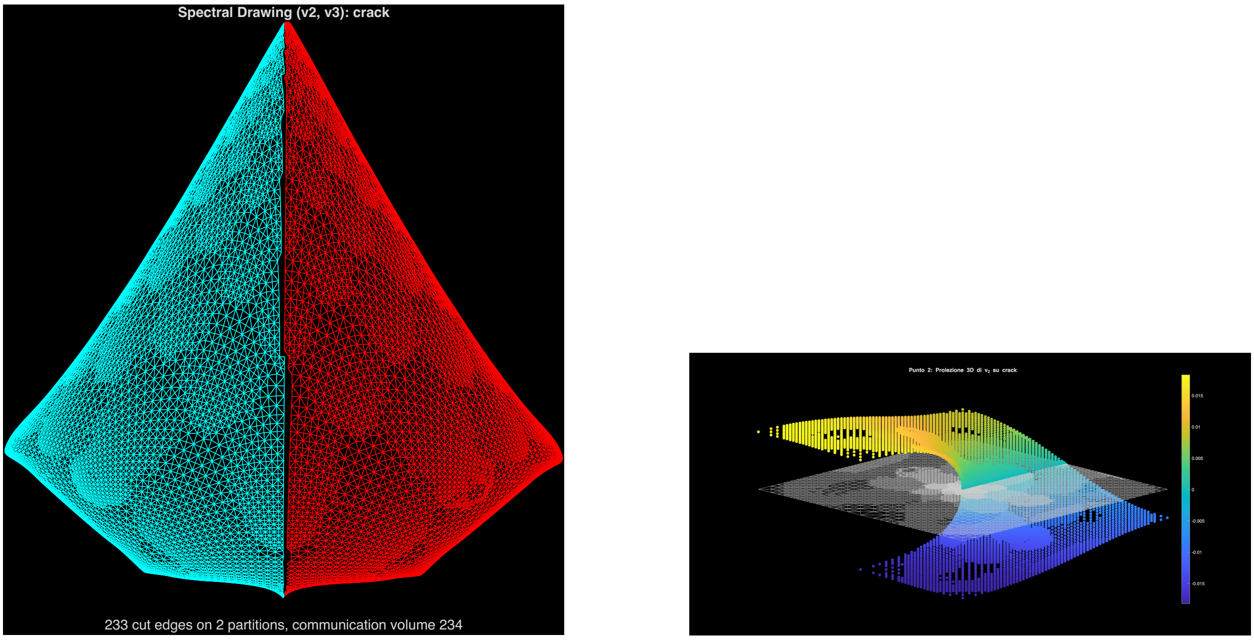

Figure 9: Crack: Spectral partitioning (left) and 3D Projection (right).

7. Task: Quality of the Report [15 Points]

Additional notes and submission details

Submit the source code files (together with your used `Makefile`) in an archive file (tar, zip, etc.), and summarize your results and the observations for all exercises by writing an extended Latex report. Use the Latex template from the webpage and upload the Latex summary as a PDF to iCorsi.

- Your submission should be a gzipped tar archive, formatted like project_number_lastname_firstname.zip or project_number_lastname_firstname.tgz. It should contain:
  - all the source codes of your MATLAB solutions;
  - your write-up with your name project_number_lastname_firstname.pdf.
- Submit your .zip/.tgz through Icorsi.