Università
della
Svizzera
italiana

**Institute of
Computational
Science
ICS**

Numerical Computing **2020**

Student: Claudio Maggioni Discussed with: –

Solution for Project 3

Due date: Wednesday, 4 November 2020, 11:55 PM

Please note all scripts used can be found under the folder `Project_3_Maggioni_Claudio/src`.

Note on collaboration

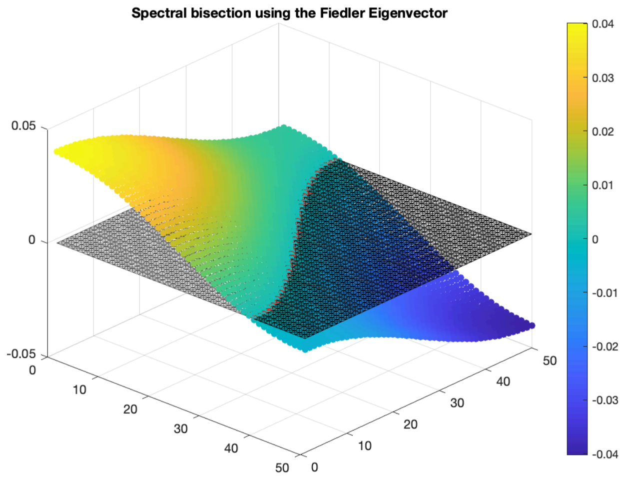
Please note that in the spirit of collaboration I shared with some of my classmates the helper function `drawgraph(...)` used in exercise 3 and included in the file `bisection_spectral.m`, and the functions `runtest(...)` and `recurse(...)` in file `bench_rec_bisection.m` for exercise 4. I am therefore claiming here I am the sole author of these functions.

Please also note this sharing was done with Edoardo Vecchi's supervision.

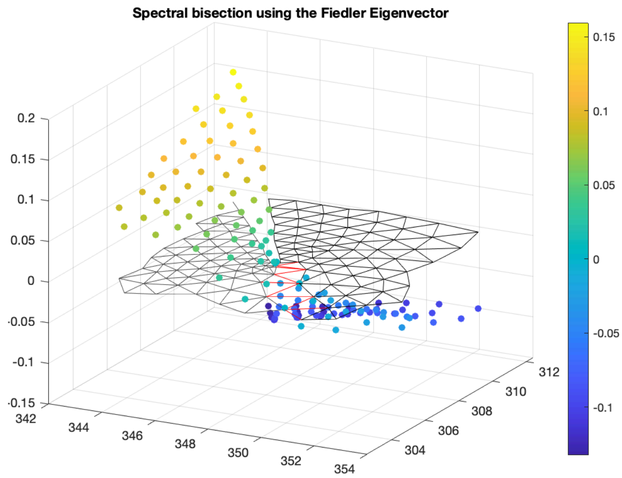
1. Install METIS 5.0.2, and the corresponding Matlab mex interface

2. Implement various graph partitioning algorithms *(60 Points)*

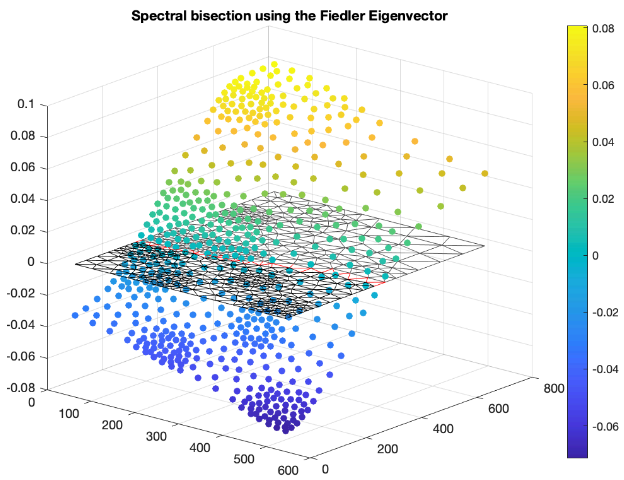
I summarize the various benchmark results in table 1. Please note that this table can be generated at will with the script `ex2_bisection_table.m`.



(a) Plot for *Grid9*



(b) Plot for *Small*



(c) Plot for *Eppstein*

Figure 1: Graph outputs for the 3 adjacency matrices.

3. Visualize the Fiedler eigenvector *(10 Points)*

In figure 1 there are graph outputs respectively from *Grid9*, *Small*, and *Eppstein*. Please note that these results can be reproduced using script `ex3_plots.m`.

Colors to represent the partitions and the eigenvector components are the same as the ones used in the the assignment's Figure 3. It is quite easy to evince from these plots that spectral partitioning indeed partitions the graph verticies based on the sign of their respective Fiedler eigenvector component: the only intersection between the Graph plane and the surface the eigenvector entries lay corresponds to the edgecut produced by spectral partitioning. This is due to the fact that in the figure the graph is positioned at $z = 0$.

4. Recursively bisecting meshes *(20 Points)*

I summarize my results in table 2. Additionally, the graph plots for a recursive partition in 8 and 16 parts of *Crack* are avaliable in figure 2.

The figure shows the partitions and edge cuts with colors randomly assigned by MATLAB.

Table 1: Bisection results

Mesh	Coordinate	Metis 5.0.2	Spectral	Inertial
grid5rect(10,100)	10	10	10	10
grid5rect(100,10)	10	10	10	10
grid5recRotate(100,10,-45)	18	10	10	10
gridt(40)	58	58	58	58
grid9(30)	88	92	102	88
Smallmesh	25	13	12	30
Tapir	55	34	18	49
Eppstein	42	48	42	45

Table 2: Edge-cut results for recursive bi-partitioning (data for $n = 8$ on the top and $n = 16$ on the bottom).

Case $n = 8$	Spectral	Metis 5.0.2	Coordinate	Inertial
airfoil1	327	320	516	578
3elt	372	395	733	880
barth4	505	405	875	888
mesh3e1	75	75	75	76
crack	804	784	1343	1061

Case $n = 16$	Spectral	Metis 5.0.2	Coordinate	Inertial
airfoil1	578	563	819	903
3elt	671	651	1168	1342
barth4	758	689	1306	1348
mesh3e1	124	117	122	116
crack	1303	1290	1860	1618

Again the data shown here can reproduced using a MATLAB script (`Bench_rec_bisection.m`). Please note that the given recursion helper function `rec_bisection` was not used. Instead, the function `runtest` was implemented to allow for partitioning graphs to be generated at will.

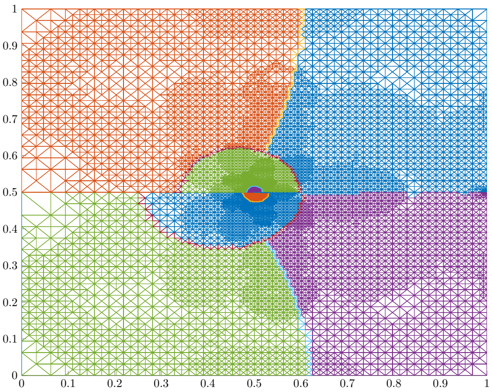
Please note that while computing spectral partitioning for *barth4* MATLAB emits a warning in the computation of the adjacency matrix eigenvalues stating that the matrix is ill conditioned. This warning was suppressed so as not to interfere with the table printout.

5. Compare recursive bisection to direct k-way partitioning *(10 Points)*

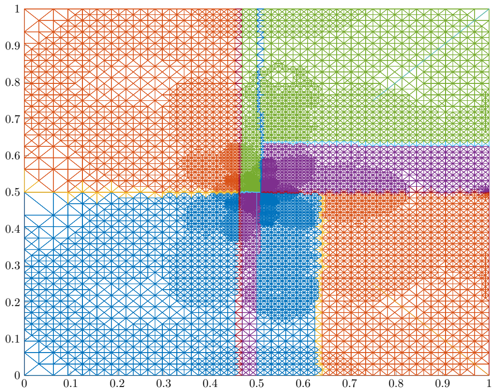
I summarize my results in table 3. Additionally, plots for the $n = 32$ cases can be found in figure 3. Please note that both the table and the plots can be reproduced using script `Bench_metis.m`.

The figure shows the resulting partitions with colors randomly assigned by MATLAB. All edge cuts are colored in black.

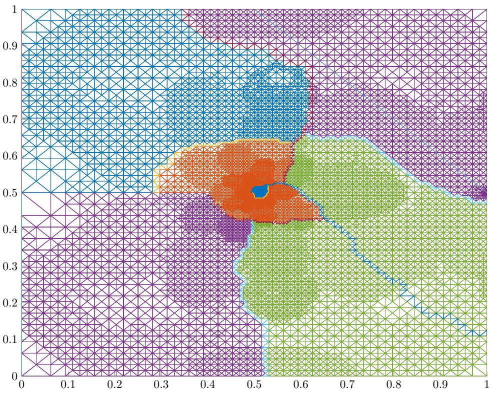
I had the suspicion that k-way partitioning would perform better in all cases, since in this the partitioning algorithm has a chance to see the entire problem and to perform decision that could affect all partitions. Instead, for the `Crack` mesh, recursive partitioning surprisedly performed better than k-way for both $n = 16$ and $n = 32$, producing slighly smaller edge cuts.



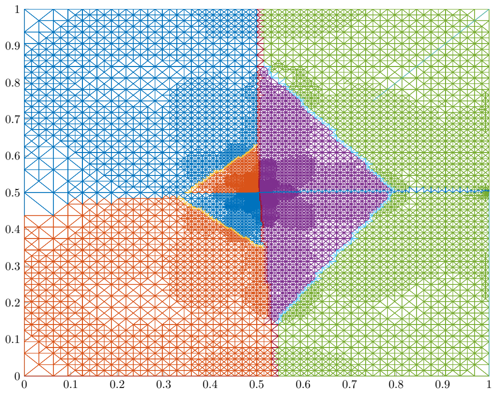
(a) Spectral bisection for $n = 8$



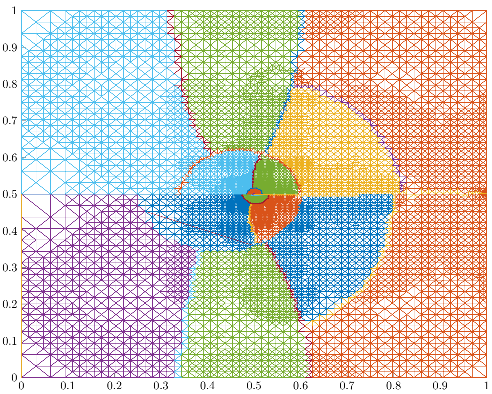
(b) Coordinate bisection for $n = 8$



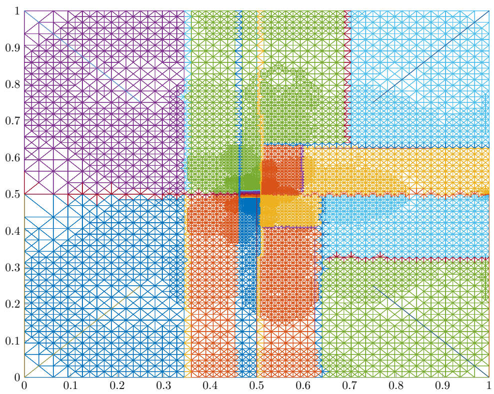
(c) METIS bisection for $n = 8$



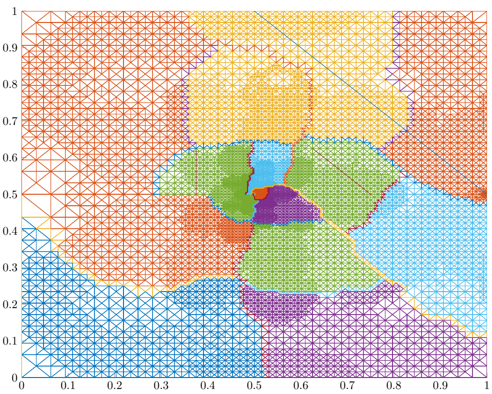
(d) Inertial bisection for $n = 8$



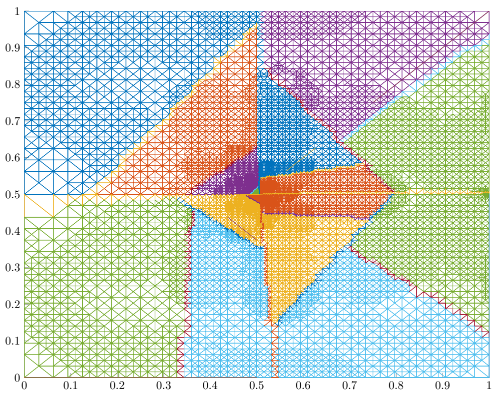
(e) Spectral bisection for $n = 16$



(f) Coordinate bisection for $n = 16$

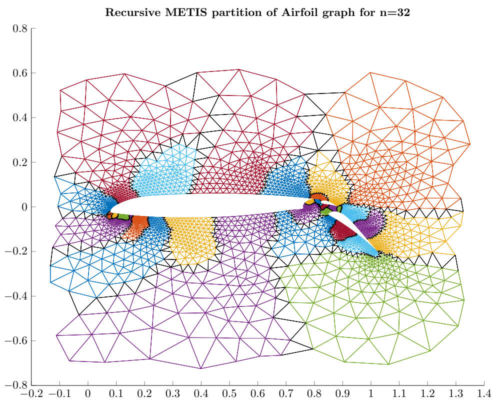


(g) METIS bisection for $n = 16$

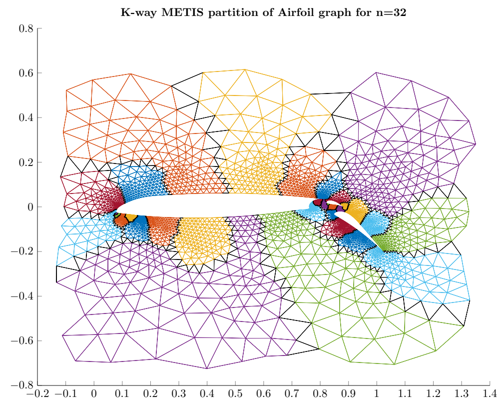


(h) Inertial bisection for $n = 16$

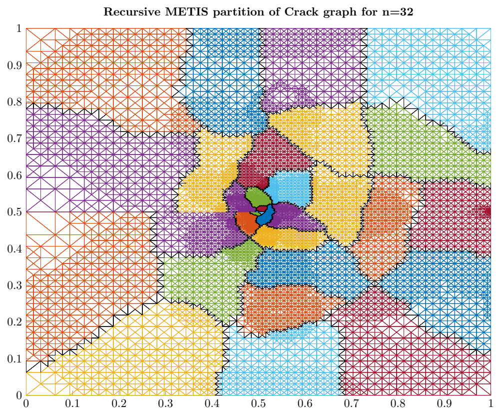
Figure 2: Graph outputs for *Crack* graph with $n = 8$ and $n = 16$



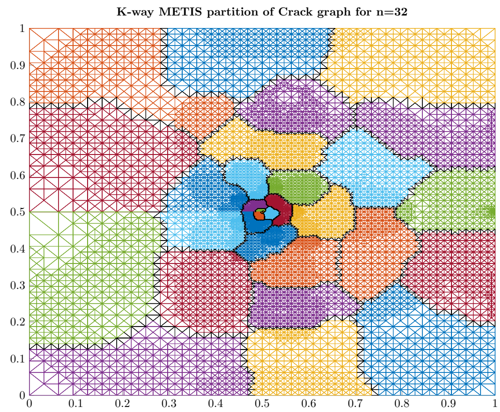
(a) *Airfoil* with recursive partitioning



(b) *Airfoil* with k-way partitioning



(c) *Crack* with recursive partitioning



(d) *Crack* with k-way partitioning

Figure 3: Graph outputs for *METIS* recursive and k-way partitioning with $n = 32$. Edge cuts are displayed in black

Table 3: Comparing the number of cut edges for recursive bisection and direct multiway partitioning in Metis 5.0.2. Results for recursive partitioning shown on top, k-way on the bottom table.

Recursive part.	crack	airfoil1
16	10055	580
32	10824	967

K-way part.	crack	airfoil1
16	10077	564
32	10943	947