Solving 2D Laplace equation using Jacobi Iteration

Sergio Oyaga - Carlos Mougan
1515122 - 1510206

In this assignment we wrote a C code to be able to solve de 2-D Laplace equations of dimension M and N. For more details, see the attached C code. We have also answered the following questions.

a) Modify the code so that parameters n and m are provided at runtime in the command line, and the memory space for the data used in the simulation is dynamically allocated.

In order to provide the parameters n and m we have use the scanf function. For the number of iterationes we have give the compiler the option that in case that there are no arguments given, that takes iter_max=200 as default case.

To make the memory dynamically allocated for the data used in the simulation we have used 'malloc'

```
int iter_max=(argv[1]!=0)?( atoi(argv[1])):(int)200;
  printf("Indroduce the n and m dimensions: ");
  scanf("%d",&n);
  scanf("%d",&m);
  A=malloc(n * sizeof(double));
  Anew=malloc(n * sizeof(double));
  for (i=0;i<n;++i)
  {
      A[i] = malloc(m * sizeof(double));
      Anew[i] = malloc(m * sizeof(double));
  }
```

In order to see the full code, the C code is given as an attachment file.

b) Modify the code so that it is functionally equivalent but it is executed faster. This is called program optimization.

For the optimization part we have done the following

- Initialization of the matrix at the same time.
- Use multiplication operations instead of divisions due that they are faster to process.
- Iterate first through columns due to that its closer in memory and it takes less time to process.
- To optimize the rewriting of the two matrix, we have used 'double buffer' which enables to iterate through one matrix in odd numbers and through the other matrix in even iterations.