1 Resolució del problema

a Acotació de les normes de les matrius d'iteració

La norma de B_J es pot calcular explícitament:

$$\begin{pmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1n} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{d1} & x_{d2} & x_{d3} & \dots & x_{dn} \end{pmatrix}$$

$$D^{-1} = \begin{pmatrix} 3 & 0 & \dots & 0 \\ 0 & 4 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 4 \end{pmatrix}^{-1} = \begin{pmatrix} 1/3 & 0 & \dots & 0 \\ 0 & 1/4 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1/4 \end{pmatrix}$$

2 Estructura del programa

El programa està separat en tres fitxers:

`jacobi.c` conté el codi per resoldre el sistema mitjançant el mètode de Jacobi.

`gs.c` conté el codi per resoldre el sistema mitjançant el mètode de Gauss-Seidel.

`sor.c` conté el codi per resoldre el sistema mitjançant el mètode de SOR. A més a més, conté la funció que s'ha utilitzat per trobar el valor òptim de la constant de SOR ω.

Cada programa retorna per la sortida estàndard totes les components del vector x, en ordre, una per línia.

S'inclou un `Makefile`. Dins del mateix s'explica cada instrucció. En particular, la instrucció per defecte (`make`) genera els tres programes, els executa, i emmagatzema les sortides en fitxers de text per poder-les comparar.