1 Compare Richardson and Jacobi

1. Implement a function `iter_solve()` which takes as arguments a matrix $A \in \mathbb{R}^{n \times n}$, a vector $b \in \mathbb{R}^n$ and a parameter θ, and returns an approximate solution of the problem $Ax = b$ after performing $m \in \mathbb{N}$ steps of the *Richardson*-iteration.

2. Add the *Jacobi*-iteration by adding an additional input `method` to your function so that the user can choose between the solvers.

3. Test your two solvers for some invertible matrix $A \in \mathbb{R}^{3 \times 3}$, some $b \in \mathbb{R}^3$ and $m = 50$. In both cases, plot the distance $\|x^k - x^*\|$ to the solution x^* (of `numpy.linalg.solve()`) for each iterate $k = 1, \ldots, m$.

Hint: Of course, it can happen that the algorithm does not converge. Use small values for θ in (i) and matrices with large values on the diagonal (compared to its other entries) in (ii). This will assure that $\rho(I - NA) < 1$.

Solution: