Università della Svizzera italiana

Institute of Computational Science ICS

Numerical Computing **2020**

Student: Claudio Maggioni Discussed with: –

Solution for Project 1

Due date: Thursday, 8 October 2020, 12:00 AM

Numerical Computing 2020 — 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.

The purpose of this assignment[1] is to learn the importance of numerical linear algebra algorithms to solve fundamental linear algebra problems that occur in search engines.

1. Page-Rank Algorithm

1.1. Theory [20 points]

1.1.1. What assumptions should be made to guarantee convergence of the power method?

The first assumption to make is that the biggest eigenvalue in terms of absolute values should (let's name it λ_1) be strictly greater than all other eigenvectors, so:

$$|\lambda_1| < |\Lambda_i| \forall i \in \{2..n\}$$

Also, the eigenvector *guess* from which the power iteration starts must have a component in the direction of x_i, the eigenvector for the eigenvalue λ_1 from before.

[1] This document is originally based on a SIAM book chapter from *Numerical Computing with Matlab* from Clever B. Moler.

1.1.2. What is a shift and invert approach?

The shift and invert approach is a variant of the power method that may significantly increase the rate of convergence where some application of the vanilla method require large numbers of iterations. This improvement is achieved by taking the input matrix A and deriving a matrix B defined as:

$$B = (A - \alpha I)^{-1}$$

where α is an arbitrary constant that must be chosen wisely in order to increase the rate of convergence. Since the eigenvalues u_i of B can be derived from the eigenvalues λ_i of A, namely:

$$u_i = \frac{1}{\lambda_i - \alpha}$$

the rate of convergence of the power method on B is:

$$\left|\frac{u_2}{u_1}\right| = \left|\frac{\frac{1}{\lambda_2-\alpha}}{\frac{1}{\lambda_1-\alpha}}\right| = \left|\frac{\lambda_1 - \alpha}{\lambda_2 - \alpha}\right|$$

By choosing α close to λ_1, the convergence is sped up. To further increase the rate of convergence (up to a cubic rate), a new α, and thus a new B, may be chosen for every iteration.

1.1.3. What is the difference in cost of a single iteration of the power method, compared to the inverse iteration?

Inverse iteration is generally more expensive than a regular application of the power method, due to the overhead caused by the intermediate matrix B. One must either recompute B every time α changes, which is rather expensive due to the inverse operation in the definition of B, or one must solve the matrix equation $(A - \alpha I)v_k = v_{k-1}$ in every iteration.

1.1.4. What is a Rayleigh quotient and how can it be used for eigenvalue computations?

The Railegh quotient is an effective way to either compute the corresponding eigenvalue of an eigenvector or the corresponding eigenvalue approximation of an eigenvector approximation. I.e., if x is an eigenvector, then:

$$\lambda = \mu(x) = \frac{x^T A x}{x^T x}$$

is the corresponding eigenvalue, while if x is an eigenvector approximation, for example found through some iterations of the power method, then λ is the closest possible approximation to the corresponding eigenvalue in a least-square sense.

1.2. Other webgraphs [10 points]

1.3. Connectivity matrix and subcliques [10 points]

1.4. Connectivity matrix and disjoint subgraphs [10 points]

1.5. PageRanks by solving a sparse linear system [50 points]