Programming Concepts in Scientific Programming EPFL, Master class

Prof. Guillaume Anciaux

TAs : A. Nielsen & L. Pegolotti

Students: Bonifazi Alberto & D'Urso Salvatore

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Eigenvalue problems

Introduction
The final aim of our program is to implement numerical methods for the calculation of the eigenvalues and eigenvectors of a matrix. In particular, the following methods have been applied:

- *Power method;*

- *Inverse Power method;*

- *Power method with Shift;*

- *Inverse Power method with Shift;*

Compiling and Installation
The whole project has been developed using CLion. For this reason, we recommend running it on this convenient idle. It's also possible to compile it with cmake, following these instructions.
mkdir build
cd build
cmake
make ..
make
./Eigenvalue
To clone the repository, you can run from the terminal *git clone ssh://git@c4science.ch/source/eigpro.git*

Program Usage
The functioning of the program is very simple. It has a menu with different options. It's possible to load your own matrix and perform the different operations. If you are interested in finding all eigenvalues of a matrix, it is possible to proceed as follow: first call the power and inverse method to find the highest and lowest eigenvalue, then call the shift inverse function setting the shift between these two eigenvalues until you find all the remaining ones. In order to get an idea of the potential of the program it is possible to run some tests (select 3), and in automatic the program runs 2 different tests on 7 different matrices. The output of the operations is shown on the screen.

Load matrix
The matrixes are imported as text file and it must have the following formatting:

- The first row indicates the size of the square matrix (Ex: 3 indicates a 3 x 3 matrix);

- The following lines represent the content of the matrix, with the numbers separated by a space, and a return to go to the next row;

- The final line: if it is 0, it indicates that we want to perform the numerical calculation with the *Power method* and/or *Inverse Power method*; if it is different from 0, it indicates that we want to perform the numerical calculation with the *Power method with Shift* and/or *Inverse Power method with Shift*. In this case, the final value represents the *Shift amount*.

Features of the program

- The *Main* specifies the tolerance of the computations and reads the input files;

- The input file is saved inside the *Matrix class*, where the values are assigned to the size of the matrix, to the coefficients of the matrix and to the Shift Amount;

- The *printMatrix* function prints the content of the square matrix on the screen;

- The *powerMethod* function runs the eigenvalue computation based on the *Power method* (if the Shift Amount is 0) or on the *Power method with Shift*;

- The *inversePowerMethod* function runs the eigenvalue computation based on *the Inverse Power method* (if the Shift Amount is 0) or on the *Inverse Power method with Shift*;

- The *LUPDecompose* function computes the PA = LU decomposition;

- The *LUPSolve* function resolves the equation A * x = b;

- The *shift* function calculates the shift matrix A – shift_amount * I;

- The *l2_norm* function computes the Euclidean norm of a vector;

- The function *linf_norm* computes the infinity norm of a vector.

Output

The output of the program consists of:

- The matrix read from file.txt;

- The Eigenvalue calculated with the *Power method* (with *Shift* if the shift amount is not 0);

- The Eigenvector correlated;

- The verification with the values of the eigenvalues identified with *WolframAlpha*;

- The Eigenvalue calculated with the Inverse *Power method* (with *Shift* if the shift amount is not 0);

- The Eigenvector correlated;

- The verification with the values of the eigenvalues identified with *WolframAlpha*;

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Test List

$input\ 1 \begin{bmatrix} 0 & 11 & -5 \\ -2 & 17 & -7 \\ -4 & 26 & -10 \end{bmatrix}$ expected eigenvalue: $\lambda_1 = 4$ $\lambda_2 = 1$

$Shift\ Amount = 0$

$input\ 2 \begin{bmatrix} 0 & -1 & 1 \\ 7 & 5.5 & -7 \\ 5 & 2.5 & -4 \end{bmatrix}$ expected eigenvalue: $\lambda_1 = 2$ $\lambda_2 = 1$

$Shift\ Amount = 0$

$input\ 3 \begin{bmatrix} 0 & -1 & 1 \\ 7 & 5.5 & -7 \\ 5 & 2.5 & -4 \end{bmatrix}$ expected eigenvalue: $\lambda_1 = 2$ $\lambda_2 = -1.5$

$Shift\ Amount = -1$

$input\ 4 \begin{bmatrix} 1 & 2 & 3 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{bmatrix}$ expected eigenvalue: $\lambda_1 = 2$ $\lambda_2 = 1$

$Shift\ Amount = 0$

$input\ 5 \begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 0 \\ 3 & 2 & 1 \end{bmatrix}$ expected eigenvalue: $\lambda_1 = 2$ $\lambda_2 = 1$

$Shift\ Amount = 0$

$input\ 6 \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 2 \end{bmatrix}$ expected eigenvalue: $\lambda_1 = 3.8$ $\lambda_2 = 0.2$

$Shift\ Amount = 0$

$input\ 7 \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 2 \end{bmatrix}$ with the eigenvalue: $\lambda_1 = 3.8$ $\lambda_2 = 0.75$

$Shift\ Amount = 0.8$

Future Development

In the input data we have always assumed a correct formatting of the file, which is not necessarily true. For this reason, a control on the input data is hypothesised. It would be advisable to check:

- The size of the matrix;

- If the matrix is square.

Furthermore, in order to apply the different iterative methods some assumption have been made with respect to the matrix A. In fact, it is assumed that it has a diagonal Jordan canonical form, such as P^(−1)AP = D = diag[λ1, ..., λn]. In addition, it is assumed there is a single dominant eigenvalue, in the form |λ1| > |λ2|≥|λ3|≥···≥|λn| ≥ 0. For this reason in all other cases the program might converge to the wrong values or it might not converge at all.

Moreover, in the case of the inverse shift method the shift needs to be different from the eigenvalues of A, otherwise (A - shift_amount * I) becomes a singular matrix and it is not possible to compute the inverse. It should be implemented in future releases a way to check during the execution of the program that these conditions are still satisfied and in the case they are not there should be a message to inform the user.

Other improvements are:

Stopping the algorithms in the case they are not converging and inform the user, a reasonable number of iterations might be 100000. This might happen in the case the conditions above are not satisfied or the shift is taken to be exactly in the middle of two eigenvalues. In this case it keeps oscillating between 2 numbers.

Try multiple initial values in the iterative procedure and pick the one that converges better, due to the numerical nature of these methods sometimes the algorithm doesn't converge because of a bad choice of initial condition. Probably three different initial conditions would already solve the problem.

Finally, for completeness also the QR method should be added in a future release.