Project report – Calcul scientifique

Gautier Rancoule, Ewen Le Bihan

ENSEEIHT, département Sciences du Numérique

1 Limitations of the power method

We test with matrices of the following shapes

1 $\begin{pmatrix} 1 & & & & (0) \\ & 2 & & & \\ & & 3 & & \\ & & & \ddots & \\ (0) & & & & n \end{pmatrix}$

2 diag(`random(1e-10, 1)`)

3 $\text{diag}\left((10^5)^{-\frac{i-1}{n-1}}\right)_{i\in[\![1,n]\!]}$

4 $\text{diag}\left(1-(1-10^{-2})\frac{i-1}{n-1}\right)_{i\in[\![1,n]\!]}$

Type / Alg	1	2	3	4
`eig` (10)	20 ms	0 ms	10 ms	10 ms
`power` (11)	1.77 s	40 ms	60 ms	1.81 s
`power` (12)	0.9 s	60 ms	60 ms	0.93 s

Table 1: Computation time comparisons

1.1 Main computing time drawback of the improved deflation method

`power_v12` is slower than `power_v11` on matrices of type **2** (diagonal matrices of random floating point values close to zero). It *is* twice as fast on matrices of type **1** or **4**.

2 Extending the power method to compute dominant eigenspace vectors

2.1 subspace$_iter_v0: abasicmethodtocomputeadominanteigenspace$

Without orthonormalisation to force the vectors to evolve to different values during the iteration, there is a large chance that all output eigenvectors will be copies of one of the eigenvectors.

2.1.1 Rayleigh quotient

The size of H is one order of magnitude less than A. Therefore, computing the spectral decompositon of H is much less intensive than computing that of A's.

2.2 v1: improved version making use of Raleigh-Ritz projection

2.2.1 Convergence analysis step

Table 2: Steps of Algorithm 4

Step	Code
Generate an initial set of m orthogonal vectors	48-49
$k = 0$	38
PercentReached = 0	40
$k = k + 1$ 54	
Compute Y such that $Y = A \cdot V$	56
Vorthonormalization of the columns of Y	58
Rayleigh-Ritz project applied on matrix A and V	61
Convergence analysis	64-112

3 v2 and v3: towards an efficient solver

3.1 Block approach

Question 8 Let n be the size of A (such that $A \in_{n,n} ()$). Computing A^2 takes $n^2(2n-1)$ flops.
Therefore, the computation is

$$pn^2(2n-1) \quad \text{flops}$$

We have $V \in_{n,m} ()$, therefore, computing $A^p \cdot V$ takes

$$pn^2(2n-1) + nm(2n-1) = (2n-1)n(pn+m) \quad \text{flops}$$

By computing $A \cdot V$ first, then pre-multiplying the resulting *vector* by A $p-1$ times, most of the matrix products will be with a vector instead of between two matrices:
The cost is then

$$nm(2n-1) + (p-1)nm(2n-1) = pm(2n-1) \quad \text{flops}$$

Table 3: Performance for a range of p values

p	iterations
1	86
10	9
20	5
30	3
50	2
100	2
200	*does not converge*

Question 10 By increasing the value of p, we do less orthonormalizations, and converge quicker to the result.

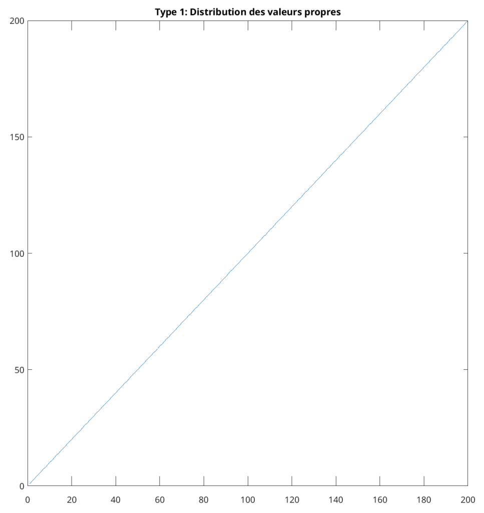
But, similarly to the *learning rate* hyperparameter in neural networks, if we try to converge too quickly, finding a local minimum will never happen and we'll oscillate around, because the steps taken are too big.

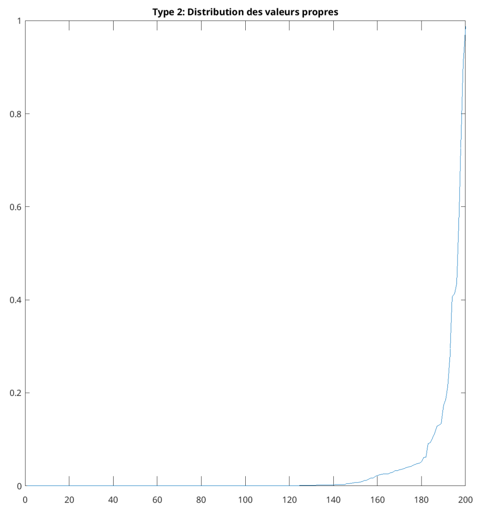
Question 11 We notice that some vectors converge faster to their eigenvector than others, specifically those that have a larger norm and those that are closer to the final guess. That faster convergence will allow for a greater eigenpair quality as the last steps of the convergence will be dedicated

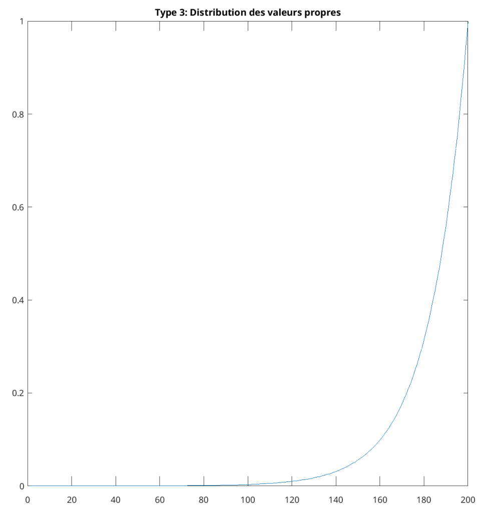
Question 12 Between `subspace_iter_v1.m` and `subspace_iter_v3.m`, the precision of `subspace_iter_v3.m` will be better because the stopping criterion of `subspace_iter_v3.m` is the same as that of `subspace_iter_v2.m` which was better than that of `subspace_iter_v1.m`. Nevertheless the precision of `subspace_iter_v3.m` is likely to be less good than that of `subspace_iter_v2.m` because vectors are freezer, but by being freezer it will be more towards a better guess

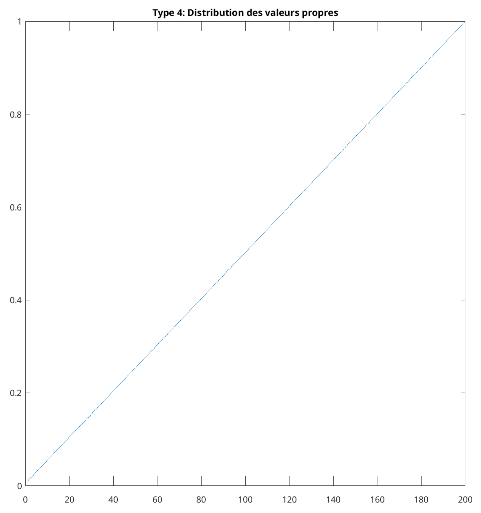
4 Numerical Experiments

4.1 Question 14









4.2 Question 15

Type / Alg	Temps	Nb itérations	Qualité couples propres	Qualité valeurs propres
`subspace iteration v0`	51 s	2689	[3.940e+04 , 1.031e+20]	[9.950e-01 , 1.006e+00]
`subspace iteration v1`	0.47 s	263	[5.752e-14 , 8.534e-08]	[1.670e-14 , 1.408e-13]
`subspace iteration v2`	0.47 s	86	[4.571e-10 , 8.079e-08]	[3.571e-15 , 4.211e-14]
`subspace iteration v3`	0.73 s	86	[1.646e-08 , 8.272e-08]	[1.646e-08 , 8.272e-08]

Table 4: Computation time comparisons