Information Theory and Computation
Exercise 5

Vincenzo Maria Schimmenti - 1204565

November 12, 2019

Theory and requests

In this assignment we are going to study the behavior of the eigenvalues of random Hermitian matrices; denote $\{\lambda_i\}_{i=1}^N$ the ordered set of these eigenvalues (which, by the spectral theorem, are real). We are interested in studying the normalized spacing:

$$s_i = \frac{\lambda_{i+1} - \lambda_i}{\Delta\lambda} \tag{1}$$

where $\Delta\lambda$ is the average $\Delta\lambda_i$. This average difference can be either compted *globally*, considering all eigenvalues at once, or *locally*, by computing an average difference for each eigenvalue, considering only some neighbors eigenvalues. We know (from Wigner) that the normalized spacing follows the following distribution:

$$P(s) = 2\left(\frac{4s}{\pi}\right)^2 e^{-\frac{4s^2}{\pi}} \tag{2}$$

Actually this expression is exact for 2×2 hermitian matrices and a good approximation for the bigger cases. We are exploit to use this functional form and fit the data we will obtain using:

$$P(s) = as^\alpha e^{-bx^\beta} \tag{3}$$

The entries of the matrix are gaussian; the gaussian distributed numbers are obtained using the so called Box Muller distribution. Starting from a couple (U, V) of uniform random variables in $[0, 1]$ we have the the two variables Z_1 and Z_2 are distributed (and independent) as $\mathcal{N}(0, 1)$ if:

$$Z_1 = \sqrt{-2\ln U}\cos(2\pi V) \tag{4}$$

$$Z_2 = \sqrt{-2\ln U}\sin(2\pi V) \tag{5}$$

We are also required to the report the average of

$$r_i = \frac{\min(\Delta\lambda_i, \Delta\lambda_{i+1})}{\max(\Delta\lambda_i, \Delta\lambda_{i+1})} \tag{6}$$

Code Development

After we generated the random matrix H of size $N \times N$ (which is a parameter of the program), we use the Lapack subroutine *cheev* to diagonalize it:

```
call cheev('N','U',nn,matr,nn,eigvs,work,lwork,rwork,info)
```

The character N tells Lapack to returns only the eigenvalues, U tells to use the upper diagonal part of the matrix (since is hermitian we can choose); the eigenvalues are stored inside the vector *eigvs*. Actually before being able to properly diagonalize the matrix, one has to tweak an internal parameter of the subroutine by calling it with *lwork=-1*.

```
! optimal lwork
lwork=-1
allocate(work(1))
allocate(rwork(max(1, 3*nn-2)))
if(.not.allocated(eigvs))allocate(eigvs(nn))
call cheev('N','U',nn,matr,nn,eigvs,work,lwork,rwork,info)
lwork = int(real(work(1)))
deallocate(work)
deallocate(rwork)
! actual diag
allocate(work(max(1,lwork)))
allocate(rwork(max(1, 3*nn-2)))
call cheev('N','U',nn,matr,nn,eigvs,work,lwork,rwork,info)
```

If the diagonalization goes well (i.e. the parameter *info* is zero), we compute the difference between the eigenvalues and normalize according to the given prescription (global or local average, as above explained); during this step we also compute and save the values r_i's. Having the spacings, we order them (using a merge-sort) and we create an histogram from them, according to a fixed cutoff (i.e. biggest value of the spacing to be considered in the binning procedure) and to a given number of bins. The Fortran program stops here by saving the obtained histogram. This whole procedure is done multiple times in order to sample different matrices, then a python script handles the fitting procedure using the fucntion described in the previous section.

Results

We used 2000×2000 matrices with 150 sampled matrices; we also used 300 bins for the histogram with a cutoff equals to 3.5, which retains the 99% of the spacings. As mentioned earlier, we also computed the histogram both using the global and local mean and for diagonal and general hermitian matrices. In the following table the results of the fit and the average r_i are shown:

	Global	Local	Global Diag	Local Diag
a	11.90	3.77	1.23	1.12
α	2.48	2.02	≈ 0	≈ 0
b	2.67	1.45	1.11	1.11
β	1.36	1.86	1.17	1.05
$\langle r \rangle$	0.60	0.60	0.39	0.39

Here instead we show the density from the data, the fitted one and, in case of general matrices, the theoretical one.

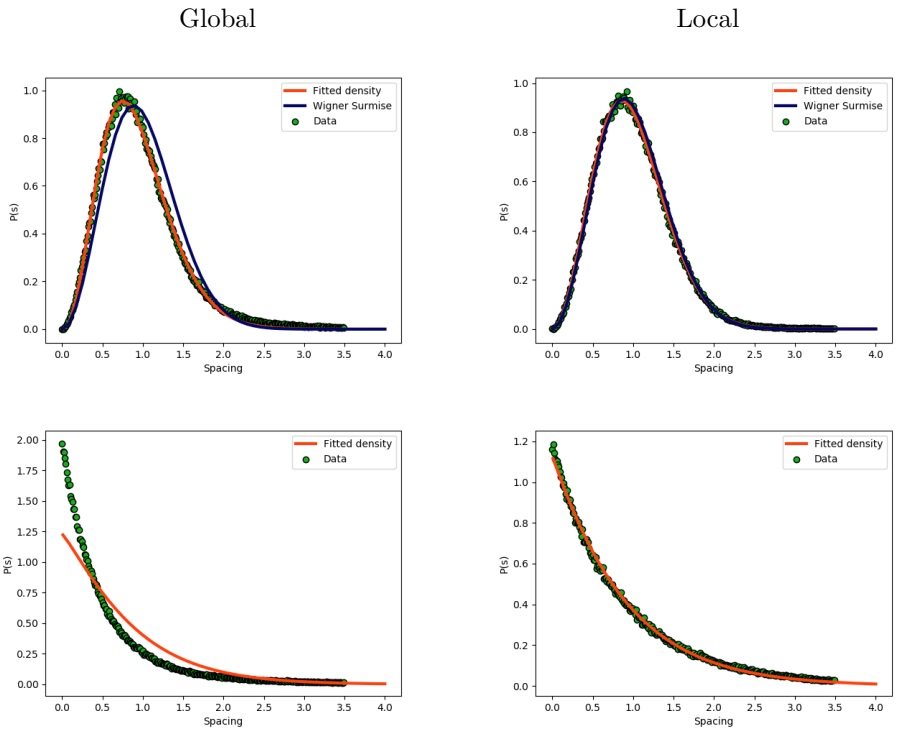

Figure 1: The first rows represent general hermitian matrices, the second the diagonal ones.

We notice how using local averages instead of the global ones the data follows more accurately the theoretical distribution (in the case of general matrices).

Info

If one wants to launch the program the following is the argument list:
./eigenproblem.out *size nSamples nBins mode avgMode*
where mode=0,1 stands for general or diagonal and avgMode=0,1 for global or local average of spacings. The same goes for the python script:
python eigenproblem.py *size nSamples nBins mode avgMode*