Exercise 5
Eigenproblem & Random Matrix Theory

Michele Guadagnini - ID 1230663

November 10, 2020

Abstract

The aim of this exercise is to study the distribution of normalized spacings of eigenvalues for different types of random matrices. The two types considered in this work are: *Hermitian* matrix and *Diagonal* matrix.

1 Theory

A complex square matrix is *Hermitian* if it is equal to its own conjugate-transpose matrix. It means that the element $a_{i,j}$ is the complex conjugate of $a_{j,i}$, so the main diagonal contains only real values, and it has real eigenvalues. The normalized spacings s_i between eigenvalues sorted in increasing order are defined as:

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

where $\overline{\Delta\lambda}$ represents the average of $\Delta\lambda_i$.

The normalized spacings of an random Hermitian matrix are expected to have a distribution that can be modeled with four parameters as:

$$P(s) = as^{\alpha}\exp(-bs^{\beta}) \tag{2}$$

2 Code Development

2.1 Design and Implementation

The code implementation started from building up the module *NormalizedSpacings* that contains all the subroutines needed for the calculation of eigenvalues and spacings. It is implemented in the file *Ex5-Guadagnini-NormSpacings-CODE.f90*. In particular the computations has been divided in the following sub-tasks:

- *RandInitHermitianMat*: it initializes the hermitian matrix as a double complex upper triangular matrix with real diagonal, since lower triangular elements can be calculated from the upper ones. Real and Imaginary part are set to be in $[10, -10]$ range.
- *HermEigenvalues*: it computes the eigenvalues of the hermitian matrix by using the *ZHEEV* subroutine from *LAPACK* library.

- *NormSpacings*: it computes the spacings from an array of eigenvalues, that must be previously ordered.
- *ComputePDF*: it computes the probability distribution from an array of double precision numbers. It receives as input also the number of bins, the extremes of the histogram and two arrays where to save the results, one for the middle values of the bins and one for the computed probabilities.
- *PrintColumnsOnFile*: it prints in two columns the results, in the first the bin centers, in the second the corresponding probabilities.

All these subroutines are then used in the main program in the file *Ex5-Guadagnini-CODE.f90*.

The first part of the program reads the size of the matrices and the number of bins from the command line arguments. It also reads the debug flag if present.

Once set the matrix size and the number of bins the program continues with the initialization of the diagonal matrix. The diagonal elements are sampled with uniform distribution between 10 and -10 and sorted in increasing order by using the subroutine *DLASRT* from *LAPACK*. Then, the array of eigenvalues is passed to *NormSpacings* subroutine and finally the subroutine *ComputePDF* is called with the computed spacings, the number of bins, the appropriate extremes for the histogram and the arrays to store the results.

The last part of the program is about the hermitian matrix spacings. The structure of this part is very similar to the previous one, except for the matrix initialization and diagonalization. These tasks are done by the dedicated subroutines *RandInitHermitianMat* and *HermEigenvalues*.

It is useful to mention also that the debugger module has been included and used in this program as in the previous exercises.

Once produced the data the implementation proceeded by writing down the *gnuplot* script to do plots and fits. The script is structured to receive as input the name of the file containing the data and the maximum x and y values to set the proper range of the plot. The main steps of the script are the definition of the fit model according to Eq. 2, the plot of distribution and fitted model and the residuals plot. They are reported in the following listing:

```
## Defining model and fitting
PP(x) = a*(x**alpha)*exp(-b*(x**beta))
fit PP(x) datafile via a, alpha, b, beta
```

```
## Save the plot
set term pdf color enhanced size 7,5
set output sprintf("%s%s%s","Fit_",MatType,".pdf")
plot datafile w boxes lc "red" title MatType, PP(x) w l lc "blue" lw 1.2
    title sprintf("%s%s", MatType, " fit")

## Save Residuals plot
unset yrange
set xlabel "Spacing s"; set ylabel sprintf("%s%s", "Fit Residuals of ",
    MatType)
set term pdf color enhanced size 7,5
set output sprintf("%s%s%s","Res_",MatType,".pdf")
plot datafile using 1:($2-PP($1)) w lp pt 13 ps 0.7 lc "red" t "Residuals"
```

Listing 1: More relevant parts of the *gnuplot* script.

2.2 Debug and Test

The program has been firstly tested with the default matrix size that is set to be 200. With this size, and also greater ones, the program worked fine (the greatest size that has been tested to work is 5000), even with all the optimization flags up to *-O3*. With a larger matrix (10000) the program returned only some non-zero values for the probabilities of the spacings distribution. It happened with all the optimization flags but only for the diagonal matrix spacings when it was computed after the hermitian matrix spacings. The solution was to reverse the order of computation, calculating firstly the diagonal matrix spacings and then the hermitian ones, suggesting the problem to be related to memory allocation or management. This way the program works fine also for very large matrices with up to *-O3* flag, while with *-Ofast* option the program returns bad results (some NaN values).

In the end the program has been compiled and executed for a matrix of size 10000 and a number of bins of 150:

```
gfortran *CODE.f90 -o RandMatTheory.x -lblas -llapack -O3
./RandMatTheory.x 10000 150
```

3 Results

The obtained distributions has been fitted according to the model in Eq. 2. The resulting parameters for both diagonal and hermitian matrices are reported in Table 1. It can be

	Diagonal spacings	**Hermitian spacings**
a	1.7 ± 0.1	17 ± 7
α	0.027 ± 0.033	2.6 ± 0.2
b	1.0 ± 0.1	3.0 ± 0.4
β	1.03 ± 0.07	1.3 ± 0.1

Table 1: Results of the spacings distributions fits.

easily noticed that the obtained parameters are very different for the two fits. In particular the parameter a is much bigger for the hermitian spacings fit, though it has a very large error (42%), and the parameter α is 2.6 in the hermitian spacings fit, while it is very close to 0 for the diagonal fit (actually, considering the error, its value is surely compatible with zero). This suggests that the diagonal spacings distribution has a shape much more similar to a simpler exponential decay, as can be seen in Figure 1. The residuals plots in Figure 1 and 2 together with the *fit.log* file content confirms the goodness of the fits.

4 Self-evaluation

Things learned while completing this assignment are to use *LAPACK* library in a program and in particula the subroutine *ZHEEV* to diagonalize a hermitian matrix.

The first thing to be done next is to complete the optional assignments (local average spacings and the average <r> computation). It would have been useful to separate spacings computation and histogram building in different programs. This would provide more flexibility. It would have been better also to do the computation multiple times with smaller matrices in order to increase available data while keeping the same computation time and also to be able to estimate a statistical error over distribution points.

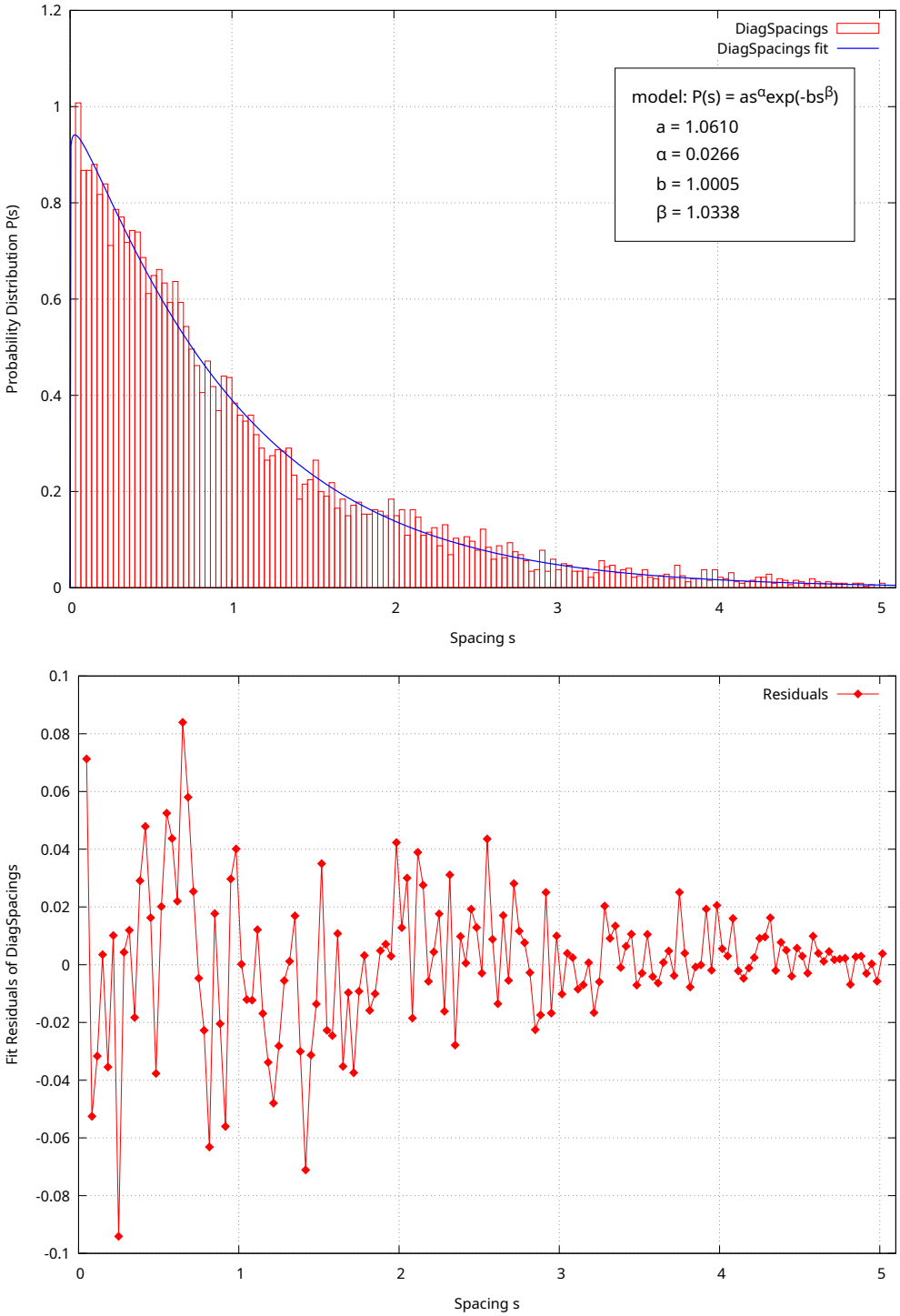

Figure 1: Fit of the distribution of diagonal matrix spacings and related residuals plot.

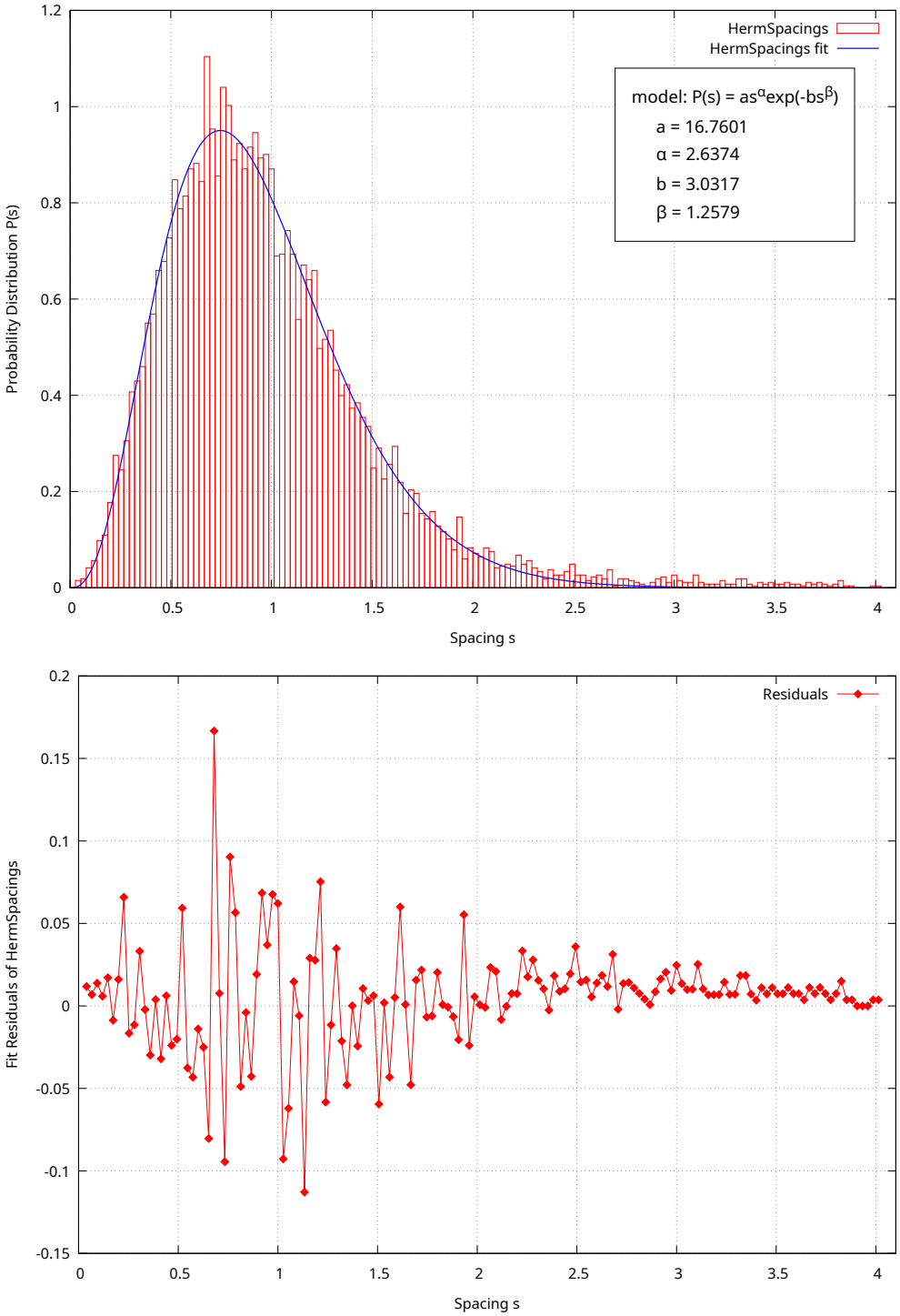

Figure 2: Fit of the distribution of hermitian matrix spacings and related residuals plot.