Quantum information and computing: Report 5

Nicola Lonigro
University of Padua
(Dated: November 9, 2020)

The statistical distribution of the distance between subsequent eigenvalues of random matrices has been studied using the LAPACK library. In particular the difference between local and global normalizations is studied.

I. THEORY

Consider the linear transformation of n-dimensional vectors defined by an n by n matrix A,

$$Av = w$$

where, for each row, $w_i = \sum_{j=1}^{n} A_{ij} v_j$. If it occurs that v and w are scalar multiples, that is if

$$Av = w = \lambda v$$

then v is an eigenvector of the linear transformation A and the scale factor λ is the eigenvalue corresponding to that eigenvector. The previous equation is the eigenvalue equation for the matrix A and it can be stated equivalently as

$$(A\mathbb{1})\, v = 0$$

where $\mathbb{1}$ is the n by n identity matrix and 0 is the zero vector.

II. CODE DEVELOPMENT

The previous code was modified to generate and compute the eigenvalues of random hermitian or diagonal matrices. The code is called with three required values: the name of the input file, the root of the output file and the normalization locality parameter. In addition to the "DEBUG" flag already implemented a new optional flag to trigger the use of diagonal matrices has been implemented.The default is to use hermitian martices but if the program is called with the flag "DIAGONAL" it will trigger the generation of diagonal matrices.

```
CALL GET_COMMAND_ARGUMENT(3,local_char)
READ(local_char,*)local_dist
CALL FATAL(  local_dist<= 0,      "main","
  Local distance must be positive integer " )

CALL GET_COMMAND_ARGUMENT(4,call_flags)
IF (call_flags == "DEBUG")     D_FLAG = .TRUE.
IF (call_flags == "DIAGONAL")  diag_switch = .
  TRUE.

CALL GET_COMMAND_ARGUMENT(5,call_flags)
IF (call_flags == "DEBUG")     D_FLAG = .TRUE.
IF (call_flags == "DIAGONAL")  diag_switch = .
  TRUE.
```

The matrix is then generated with random real numbers on the diagonal (in order to be hermitian) and complex values on the upper diagonal.

```
DO i = 1,dim_x_1
  matrix_1(i,i) = RAND()
END DO
IF(diag_switch .EQV. .FALSE.) THEN
  DO i = 2,dim_x_1
    DO j = 1,i-1
      matrix_1(i,j) = COMPLEX(RAND(),RAND())
    END DO
  END DO
END IF
CALL CHECKPOINT(D_FLAG,"After Initialization",
   dim_x_1,dim_y_1,matrix_1)
```

Subsequently the eigenvalues are computed using the CHEEV subroutine in LAPACK and the spacings and the ratios are computed

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

```
CALL CHEEV('N','L',dim_x_1,matrix_1,dim_x_1,
  egvs,WORK,LWORK,RWORK,INFO)
CALL FATAL(INFO /= 0, "main", "Eigenvalues not
   found")
DO i = 1,dim_x_1-2
  spacings(i) =   egvs(i+1) -  egvs(i+2)
END DO
DO i = 1,dim_x_1-3
  ratios(i) =   MIN(spacings(i),spacings(i+1))
  /MAX(spacings(i),spacings(i+1))
END DO
```

The spacings are then normalized to the average of $\frac{N}{local\_dist}$ where local_dist is an input parameter. Notice that if local_dist is 0 is computation is done without the for loop and using directly the SUM native function.

```
IF(local_dist /= 1) THEN
  DO i = 1,dim_x_1-2
    mean_spacing(i) = SUM(spacings(max(1,i-
  dim_x_1/local_dist):min(dim_x_1-1,i +
  dim_x_1/local_dist)))/ &
                  (min(dim_x_1-1,i+dim_x_1/
  local_dist) - max(1,i-dim_x_1/local_dist))
  END DO
  DO i = 1,dim_x_1-2
    spacings(i) = spacings(i)/mean_spacing(i)
  END DO
ELSE
  mean_spacing(1) = SUM(spacings)/(dim_x_1-1)
  spacings = spacings/mean_spacing(1)
END IF
```

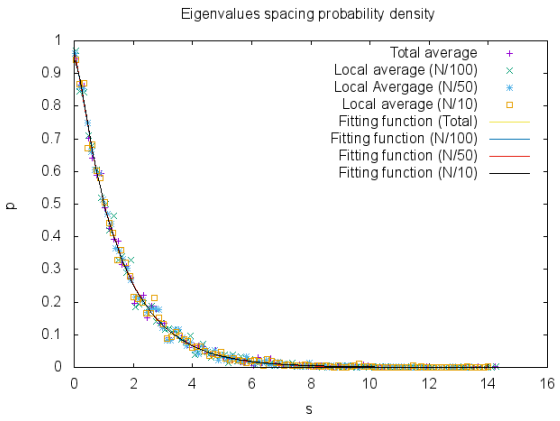

FIG. 1. Comparison of probability distribution for random diagonal matrices eigenvalues spacing

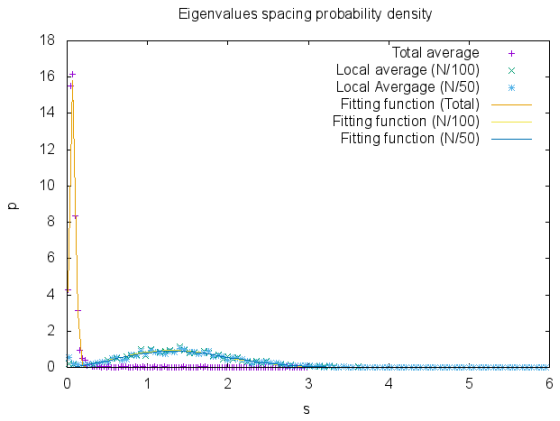

FIG. 2. Comparison of probability distribution for random hermitian matrices eigenvalues spacing

The mean on the ratios is written to a file of the type "root_rat.txt" while the eigenvalues are appended to a file of the form "root.txt".
The python script was update to work with the new program interface and it can now run the program an arbitrarily number of times with squared matrices of given dimensions. In particular it was run with values for the local_dist equal to 1, 100, 50 and 10.

```
for k in range(0,N_iter):
  f = open("N.txt", "w")
  f.write(str(int(N_min)))
  f.close()
  os.system("./Ex5.exe N.txt herm_0 1 ")
  os.system("./Ex5.exe N.txt herm_100 100 ")
  os.system("./Ex5.exe N.txt herm_50 50 ")
  os.system("./Ex5.exe N.txt herm_10 10 ")
```

The values from the different eigenvalues are then read from the correspondent txt file, binned and written to a temporary file of the form (x,y) for plotting and fitting.

```
lines=[]
text_file = open("herm_0.txt", "r")
for y in text_file.readlines():
    lines.append(float(y))
#print (lines)
print (len(lines))
text_file.close()
freq, bins = np.histogram(lines, 100, range
    =(0,4))

with open('data_0.temp', 'w') as f:
    for i in range(len(freq)):
        f.write("%s   " % (bins[i]+bins[i+1]/2))
        f.write("%s   \n" % (freq[i]/(sum(freq)
    *(-bins[i]+bins[i+1]))))
    f.close()
```

Finally the different files corresponding to the different values of local_distance are fitted and plotted together.

III. RESULTS

The code has been tested with diagonal matrices and the probability density in figure 1 has been obtained. Notice that the effect of local normalization is negligible.The parameters for the fits are given in Table I, where a fit of the type

$$f(x) = a * x^b \exp(-c * x^d)$$

is assumed.

local_distance	a	b	c	d
1	1.18 ± 0.09	0.06 ± 0.02	0.84 ± 0.08	0.90 ± 0.05
100	1.04 ± 0.08	0.014 ± 0.02	0.72 ± 0.08	0.98 ± 0.06
50	1.09 ± 0.06	0.03 ± 0.02	0.76 ± 0.06	0.95 ± 0.04
10	1.2 ± 0.1	0.057 ± 0.02	0.84 ± 0.08	0.90 ± 0.05

TABLE I. Fit parameters for normalization comparison in diagonal matrices

If hermitian matrices are used instead, the probability densities in Fig.2 are obtained and the parameters are given in II.

local_distance	a	b	c	d
1	890 ± 91	1.16 ± 0.03	212 ± 18	2.04 ± 0.05
100	1.2 ± 0.2	1.6 ± 0.2	0.4 ± 0.1	2.3 ± 0.3
50	1.1 ± 0.2	1.4 ± 0.2	0.3 ± 0.1	2.4 ± 0.4

TABLE II. Fit parameters for normalization comparison in hermitian matrices

Notice how in this case the local average changes strongly both the shape of the curve and the coefficients. Finally, the ratios average values $< r >$ for the hermitian matrices was computed and it is equal to

$$< r >= 2.13$$

IV. SELF-EVALUATION

The probability distribution of different kind of random matrices was studied. The libraries BLAS and LAPACK were introduced and the normalization of data histograms to obtain PDF's was utilized. The paramters for the hermitian matrices fit with local_distance seem large and may warrant further investigation as it may be a problem with the fit and not a physical property