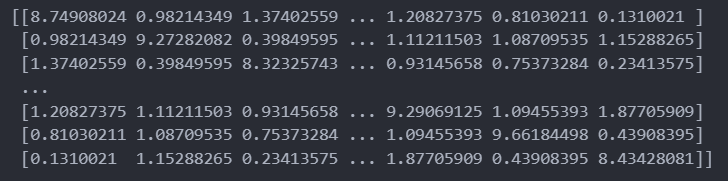
MINRES to calculate Vi

# A tale of two matrices

Here, I describe the matrices used (100x100) matrices

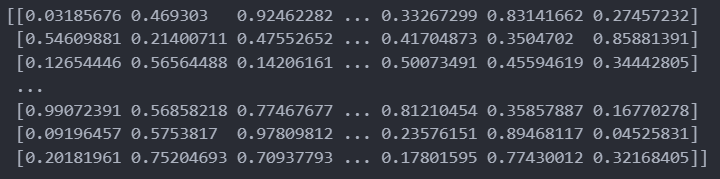
**C = B + eta\*del\_B**

The matrix B is -



* Symmetric.
* Each non-diagonal element comes in [0,2]
* The diagonal elements are greater because we added a constant times Identity matrix to an arbitrary symmetric matrix to generate B, with the constant chose so as to ensure that B has positive eigenvalues

The matrix del\_B is –



* Random
* All entries are in the range [0,1]

Now, we investigate the behaviour of our procedure for various eta values.

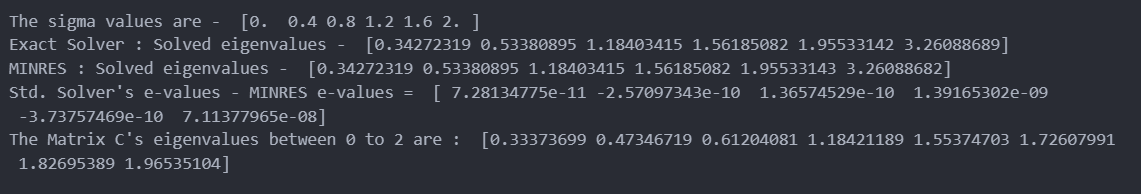
# Eta = 0 : C = B

There are 8 eigenvalues in the range [0,2] – which is the window chosen for filter diagonalization in this entire report - as seen below (Printing the first ten eigenvalues of C in ascending order)



## Sigma = 6

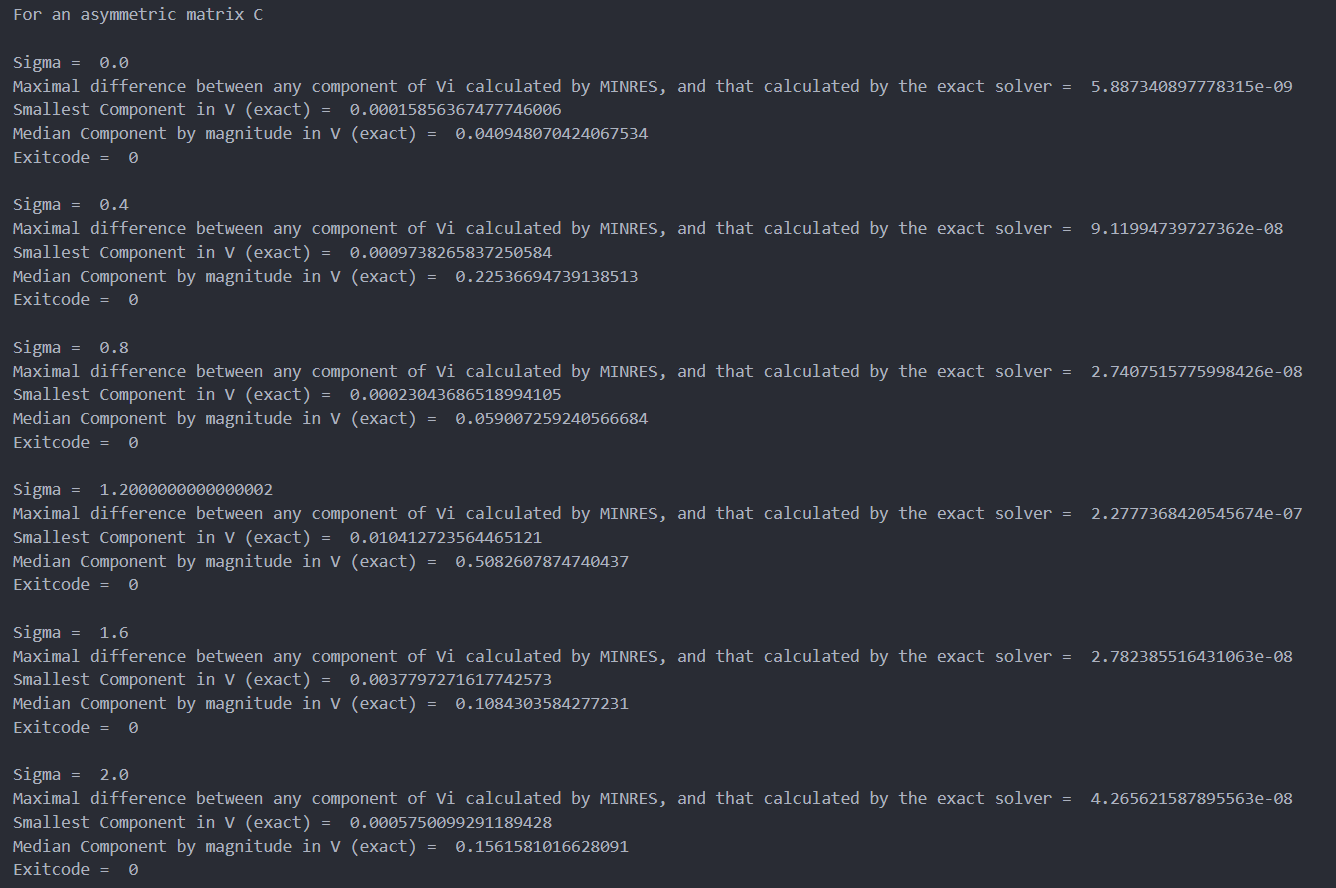
### Eigenvalue Results



The “exact solver” refers to the eigenvalues obtained by using the python package solver to calculate Vi, and correspondingly so for “MINRES”.

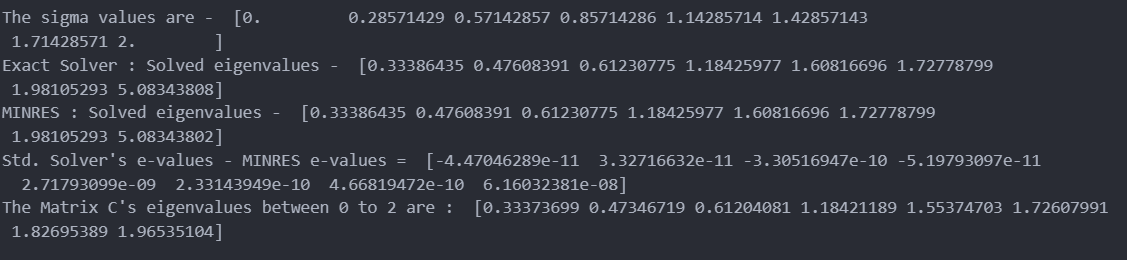
The Matrix C’s eigenvalues, as printed on the last line, was obtained by the package directly computing the eigenvalues of the 100x100 matrix C.

### Vi vector calculation results

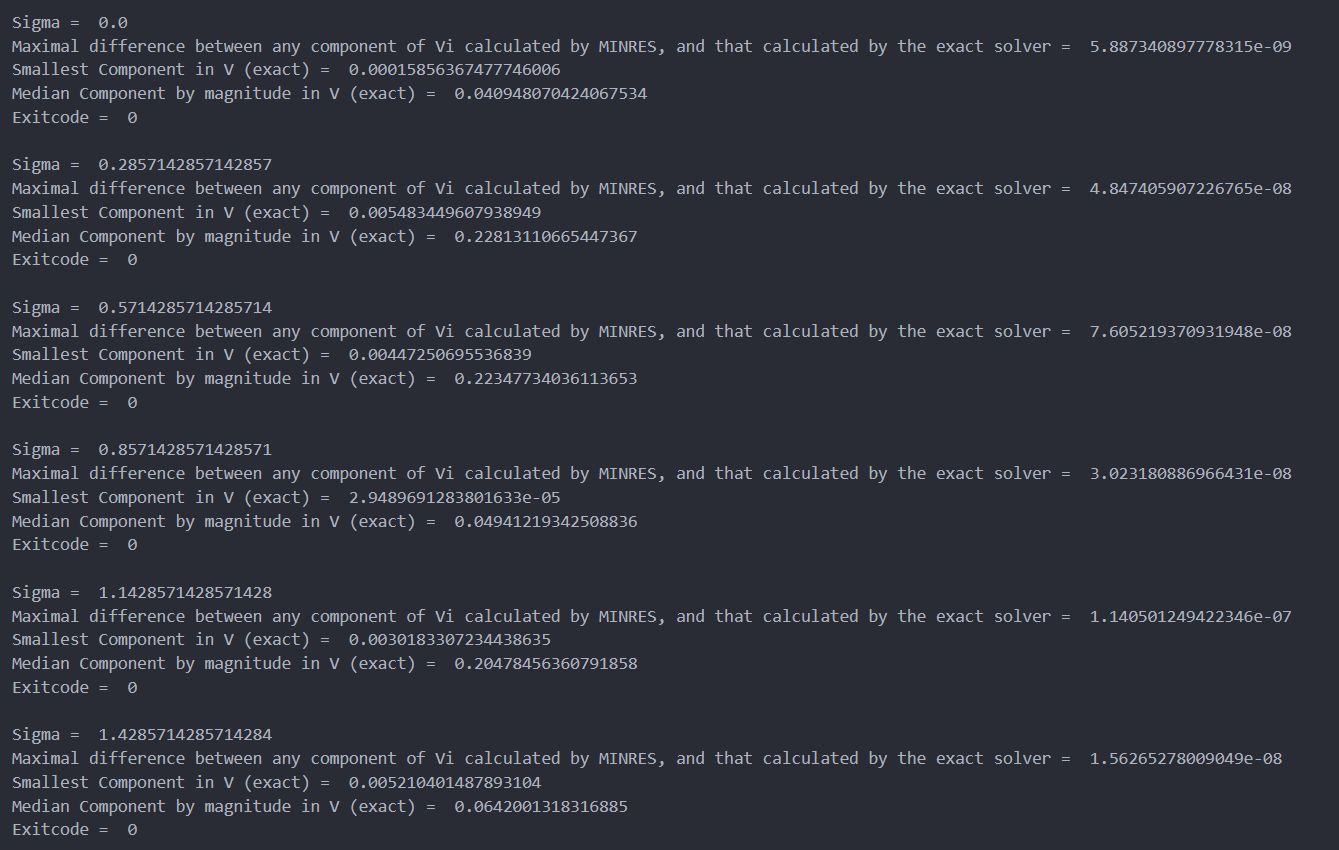


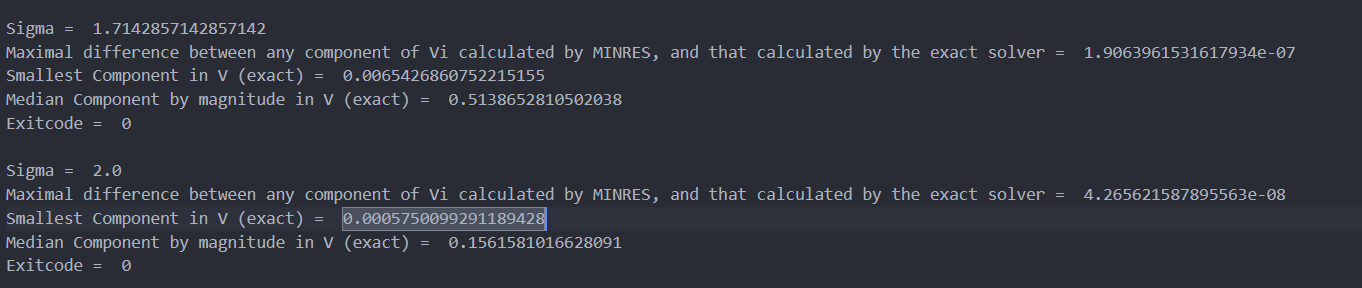
## Sigma = 8

### Eigenvalue Results



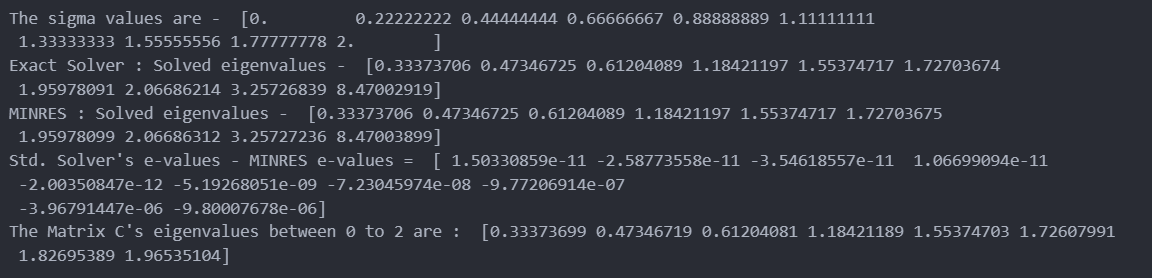
### Vi vector calculation results



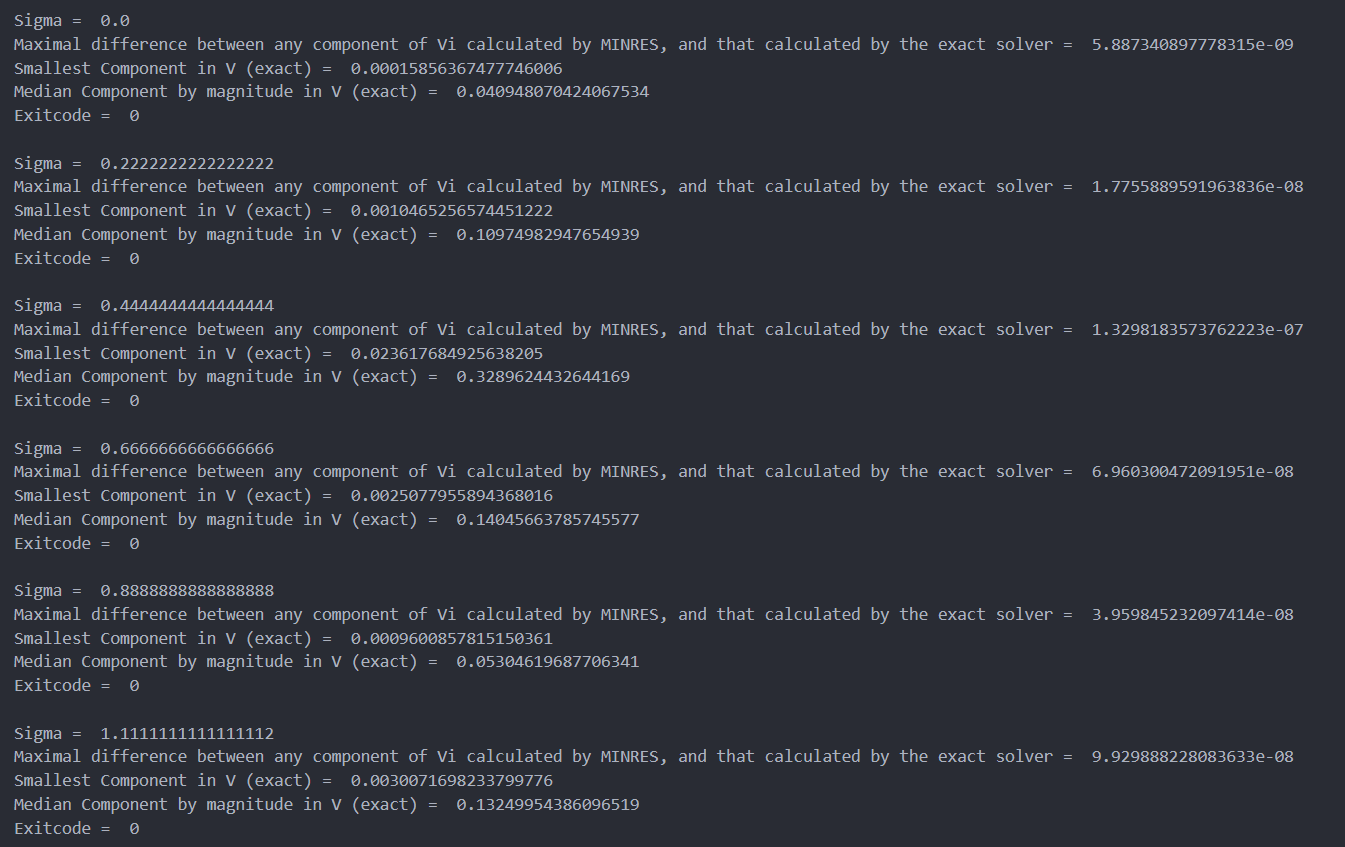


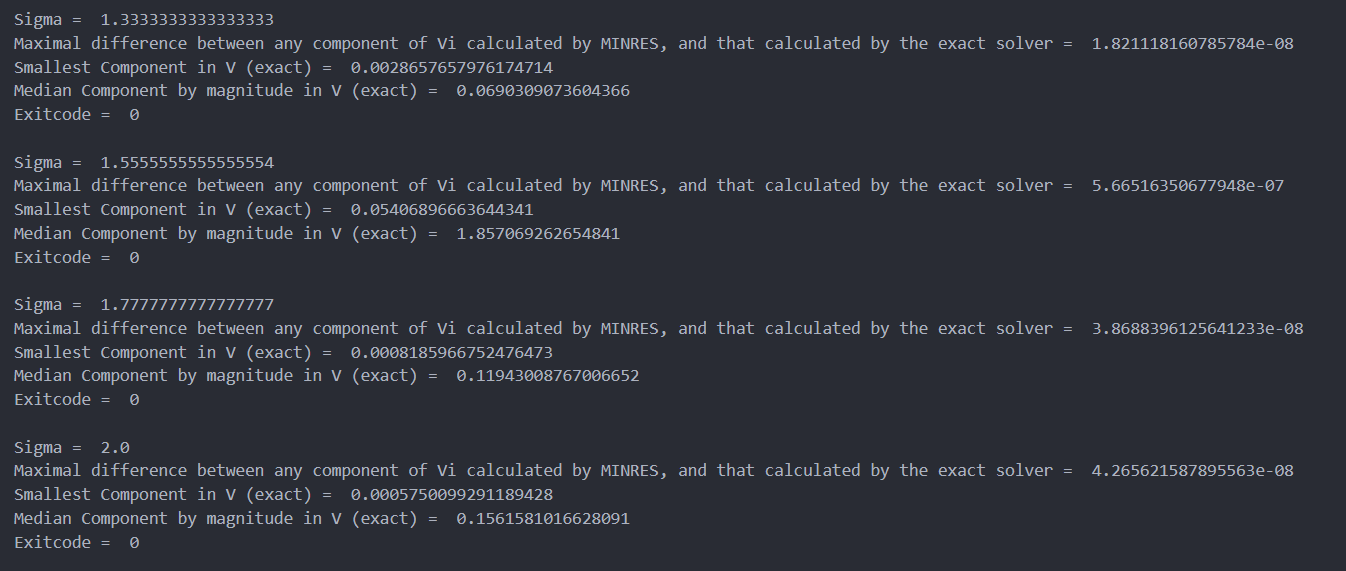
## Sigma = 10

### Eigenvector Results



### Vi vector calculation Results





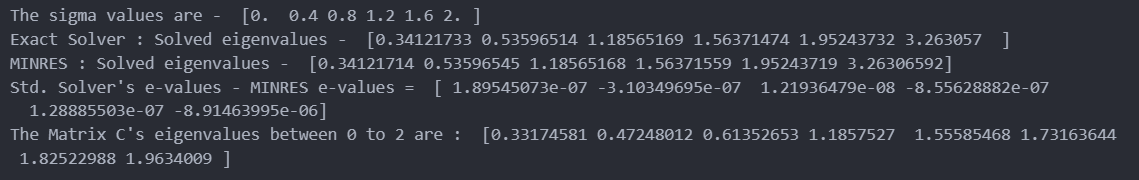
# Eta = 0.01; C = B + 0.01\*del\_B

Here too, the first 8 eigenvalues fall in the window between [0,2]. Printed below are the first 0 eigenvalues of this new C

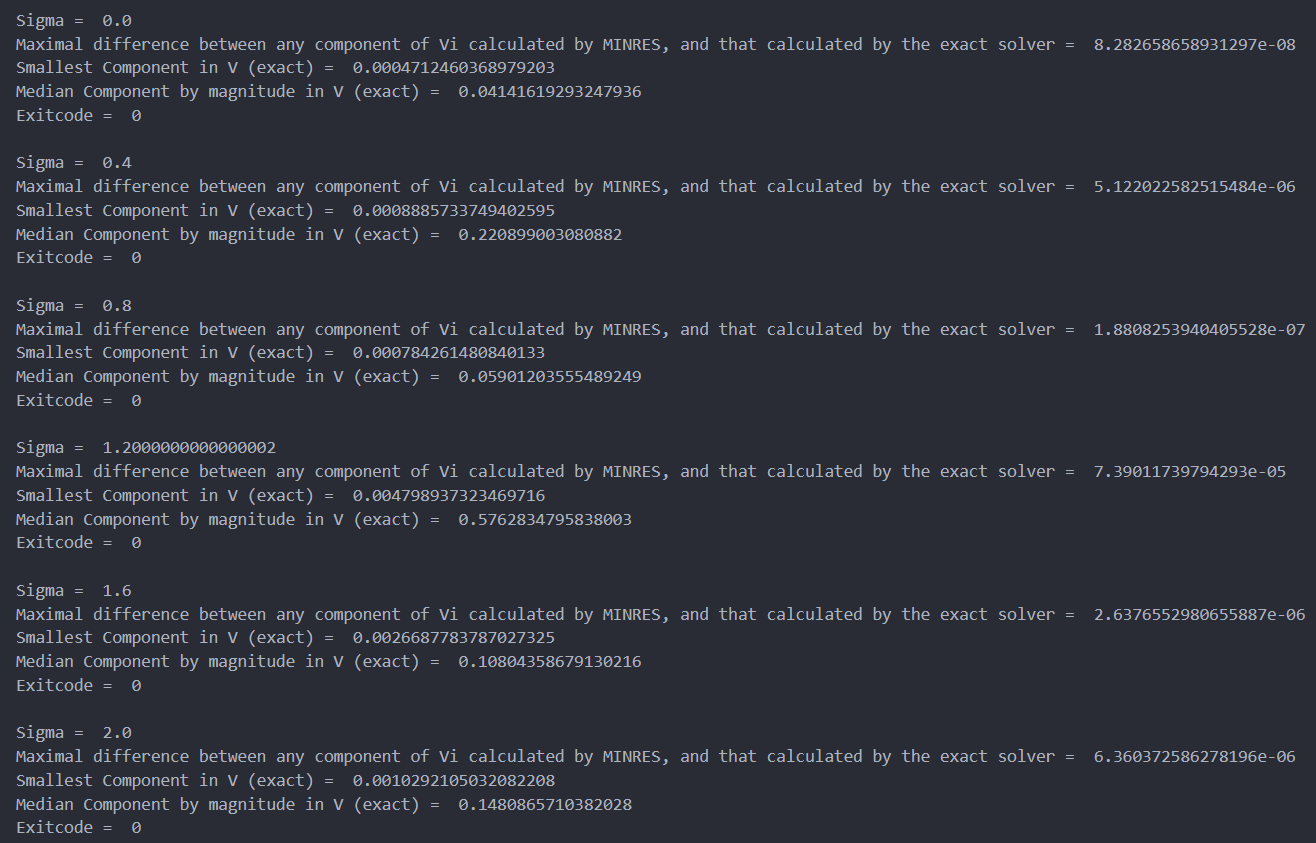


## Sigma = 6

### Eigenvalue Results

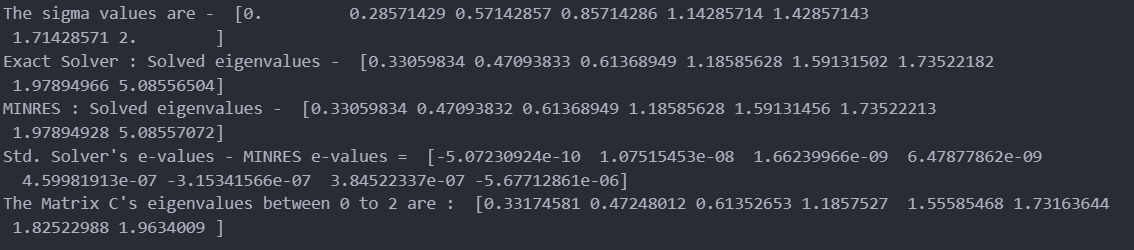


### Vi vector calculation results

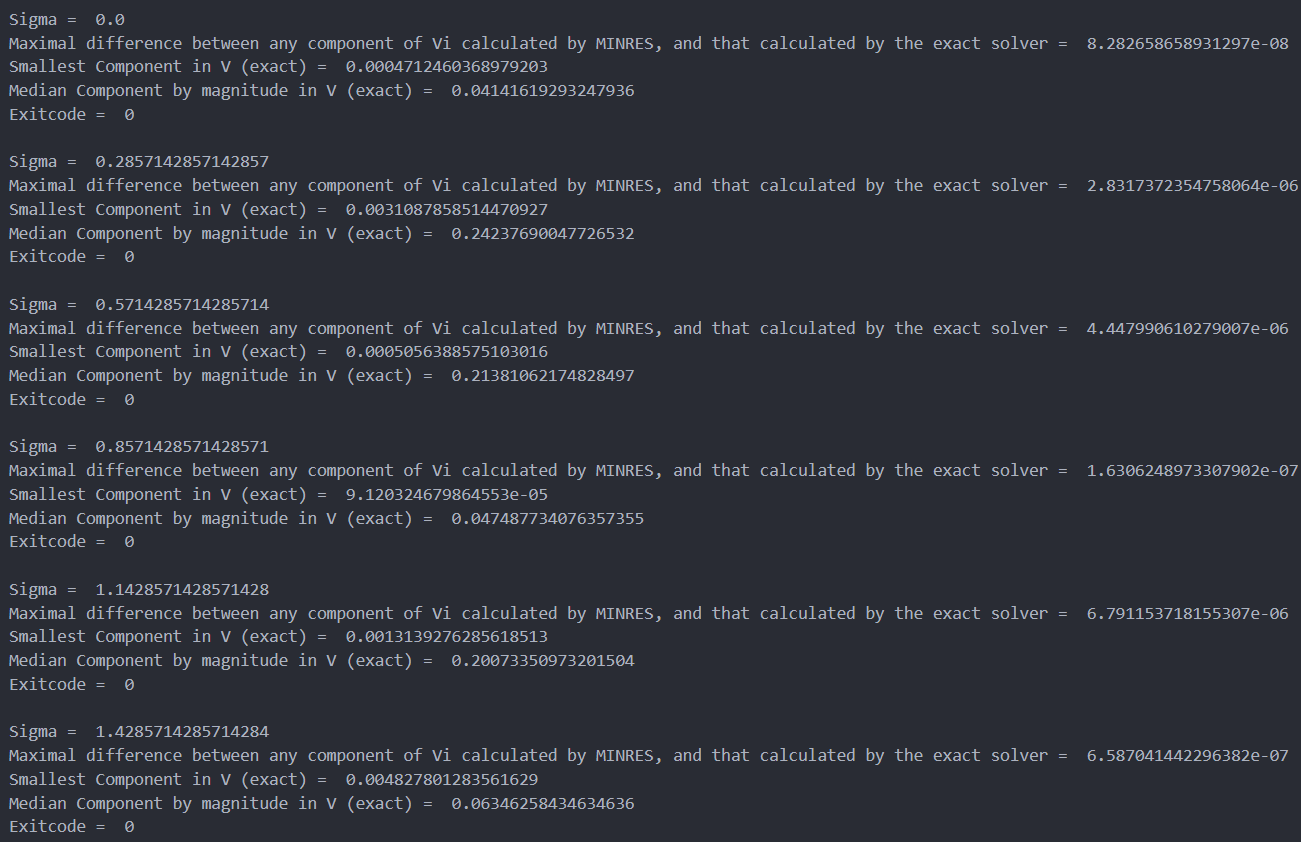


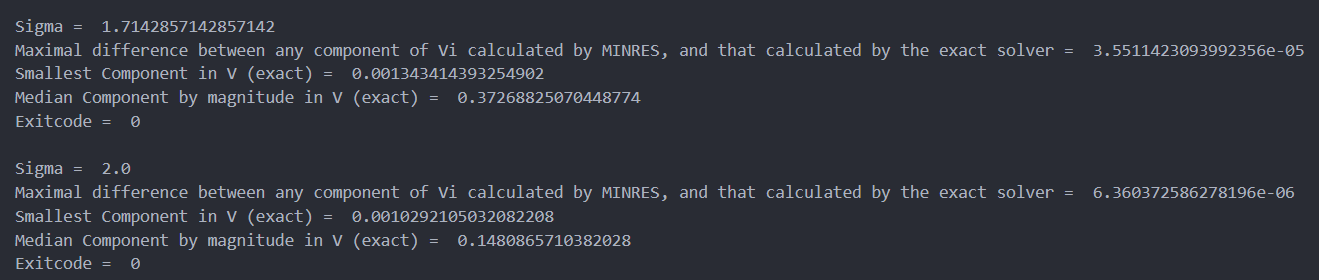
## Sigma = 8

### Eigenvalue Results



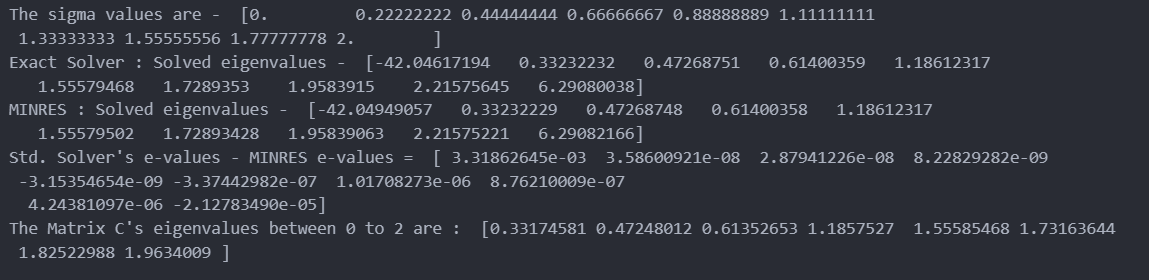
### Vi vector calculation results



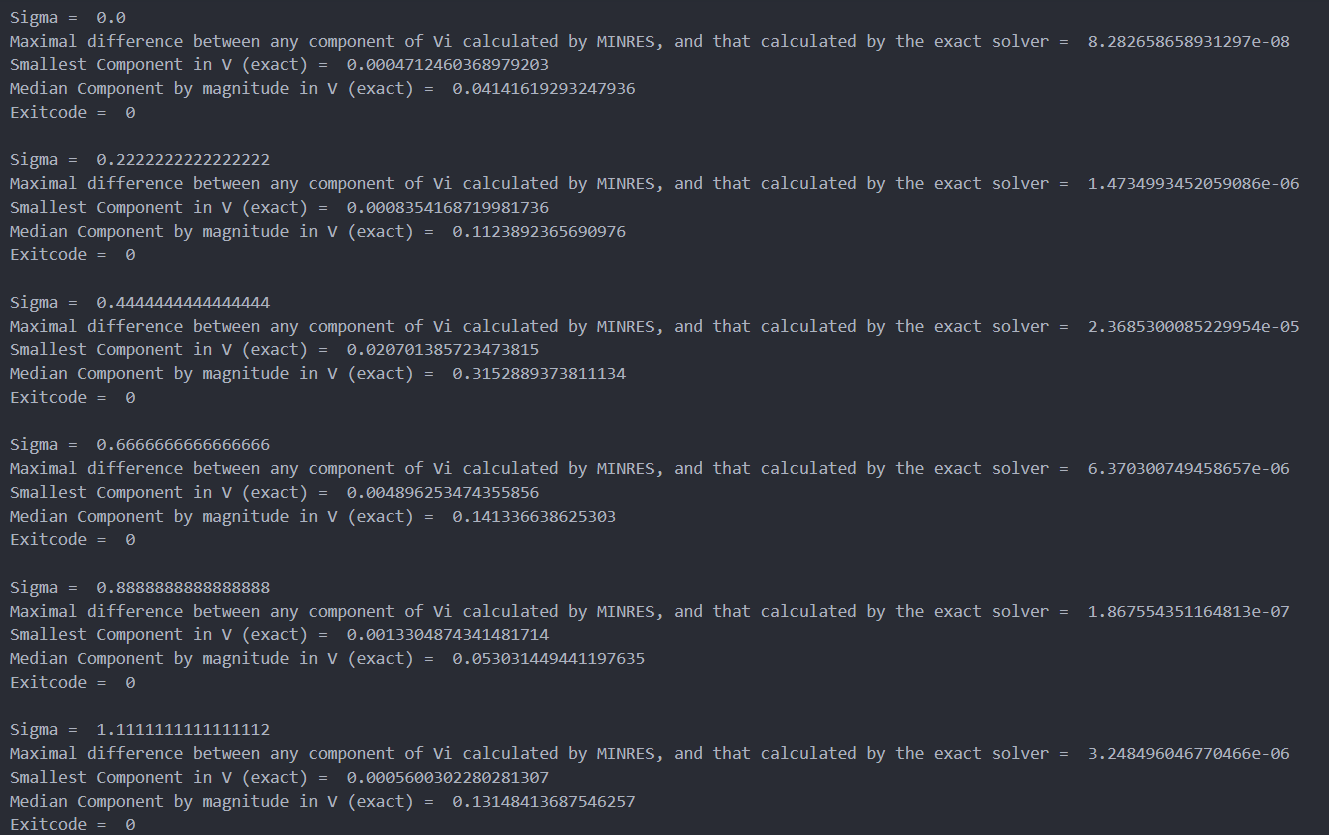


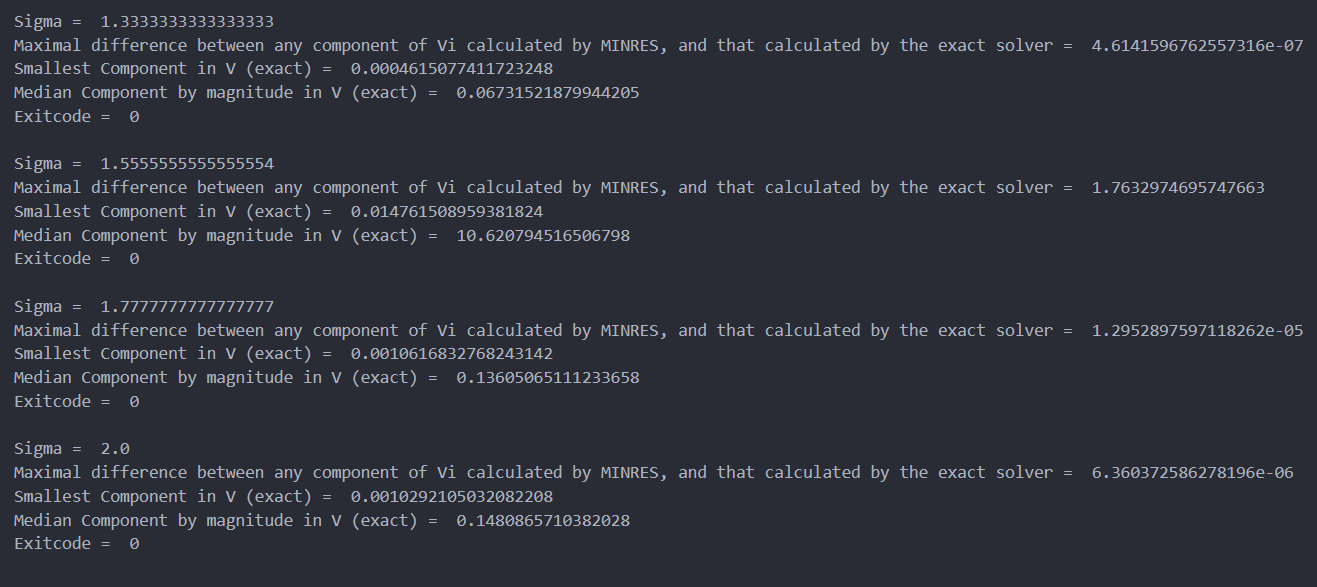
## Sigma = 10

### Eigenvector Results

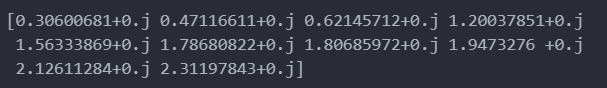


### Vi vector calculation Results



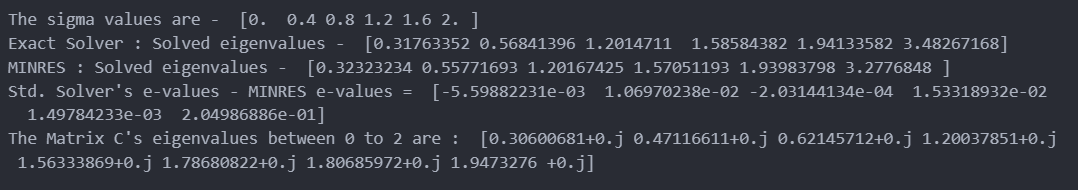


# Eta = 0.1; C = B + 0.1\*del\_B

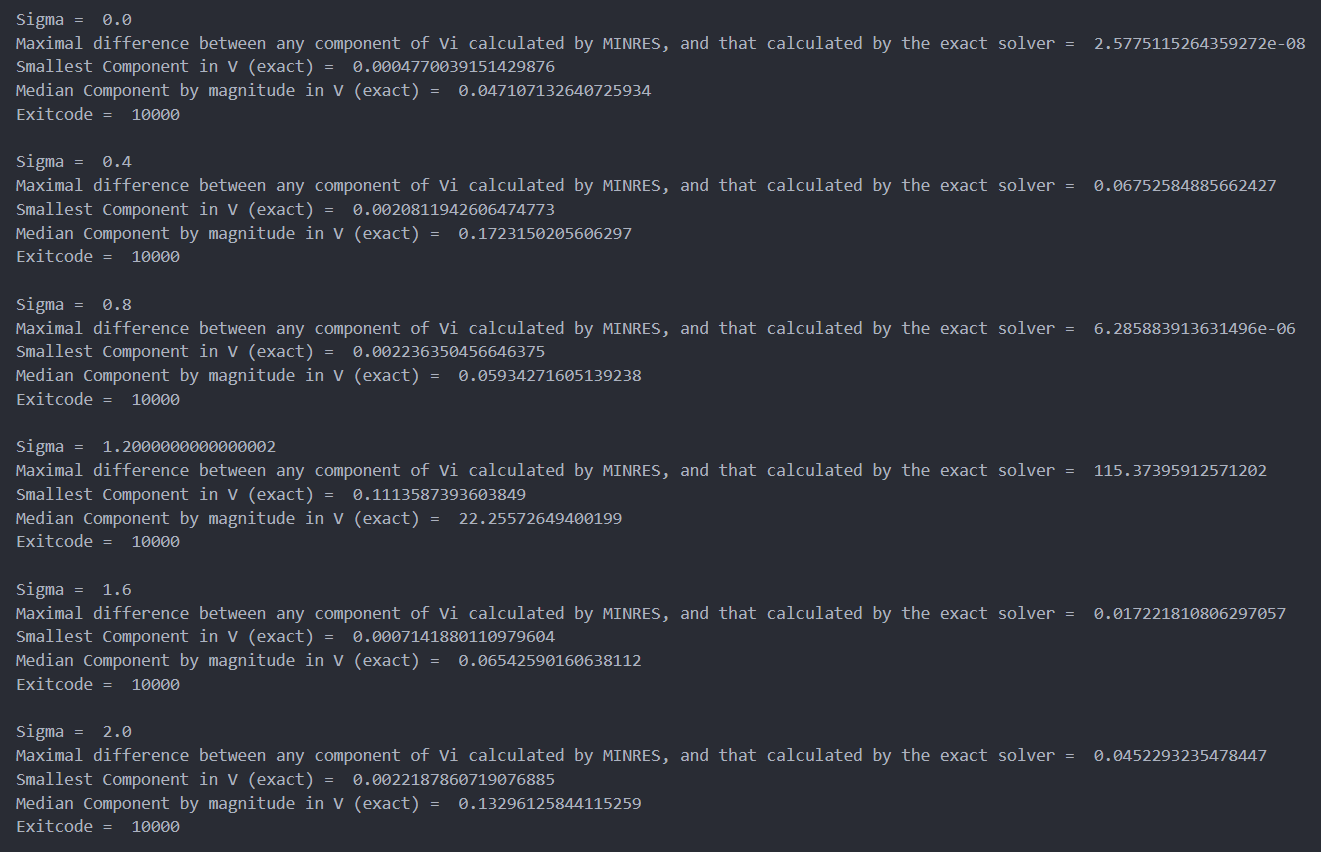
Here as well, our window from [0,2] contains the first 8 eigenvalues. The first ten of C = B + 0.1\*del\_B are printed below - 

## Sigma = 6 values

### Eigenvalue Results

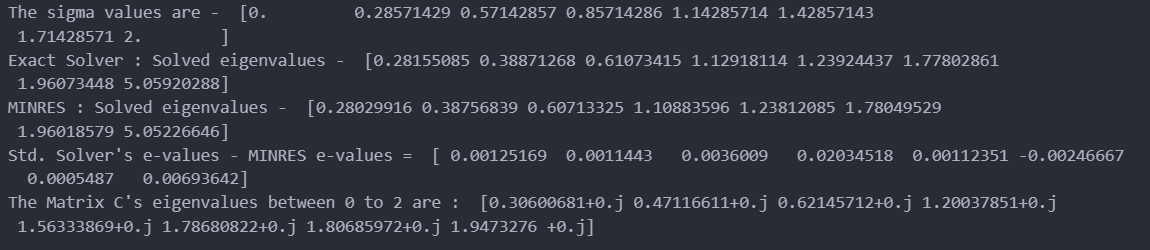


### Vi vector calculation results

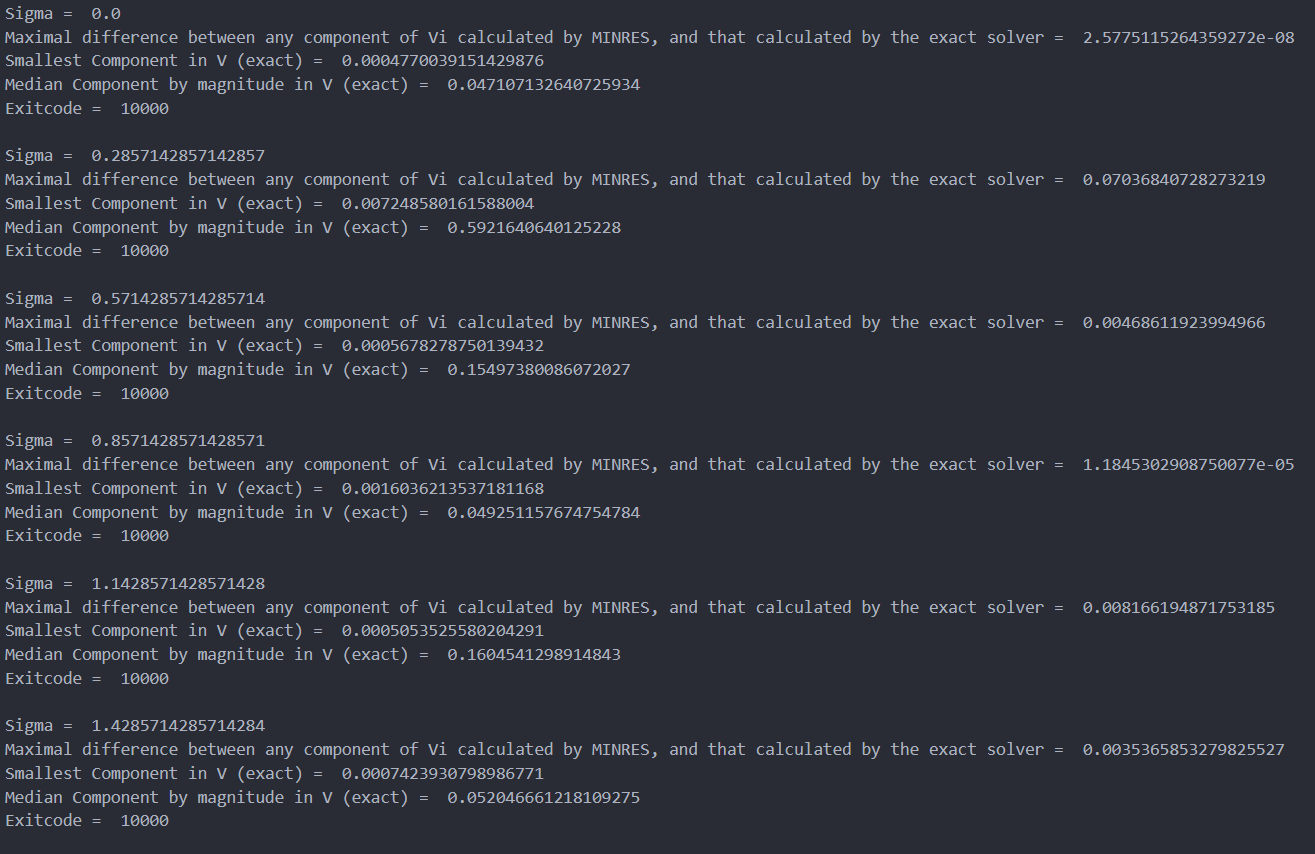


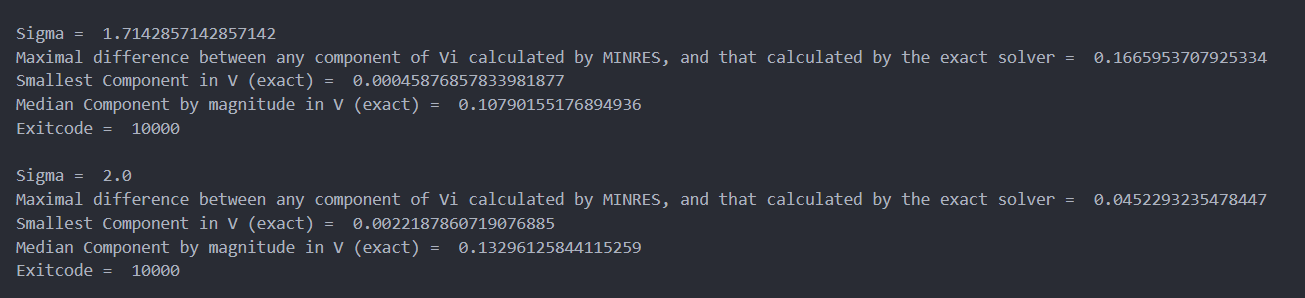
## Sigma = 8 values

### Eigenvalue Results



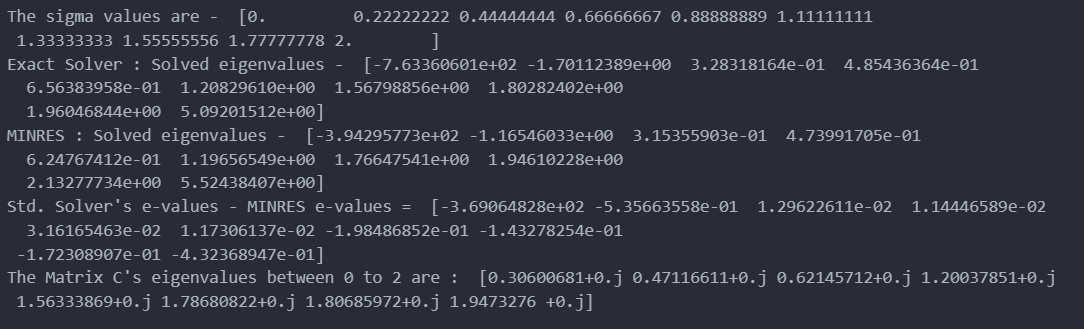
### Vi vector calculation results



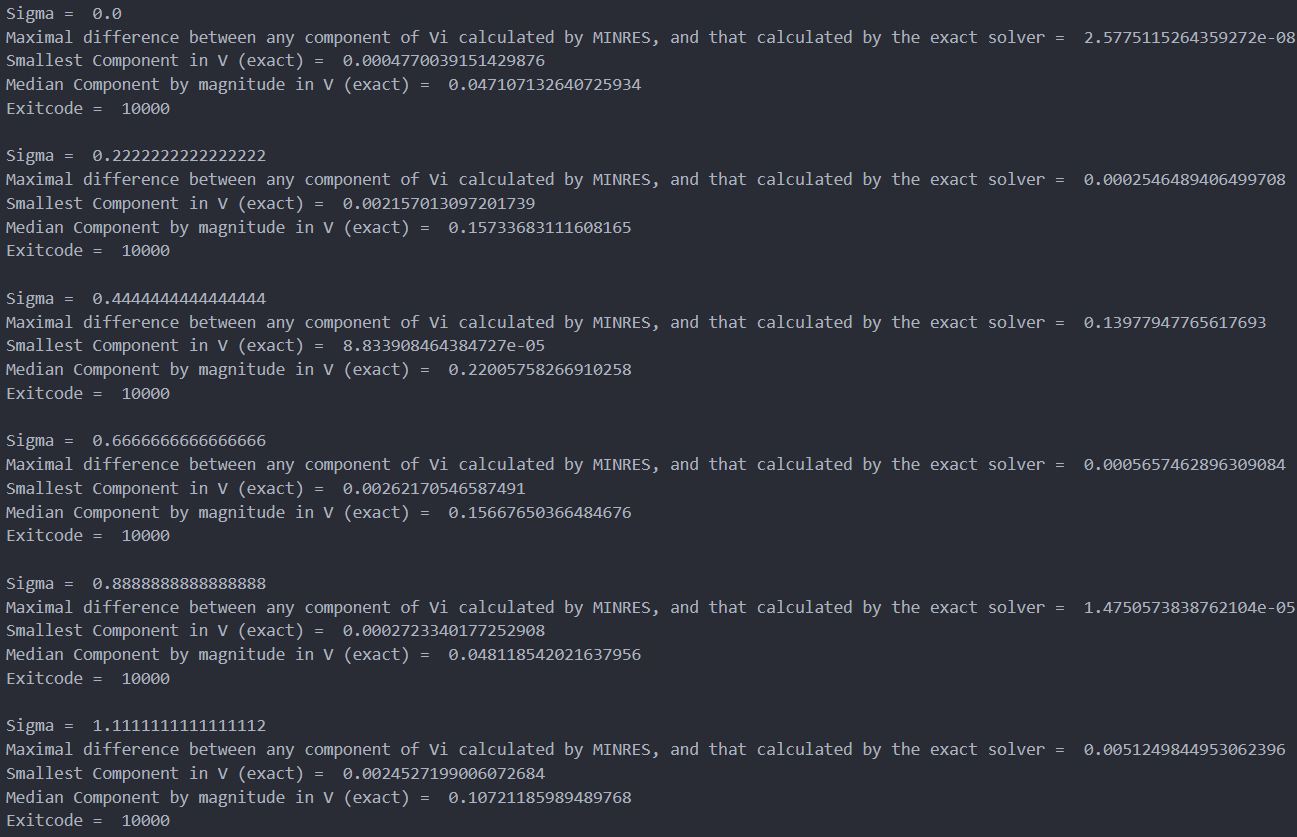
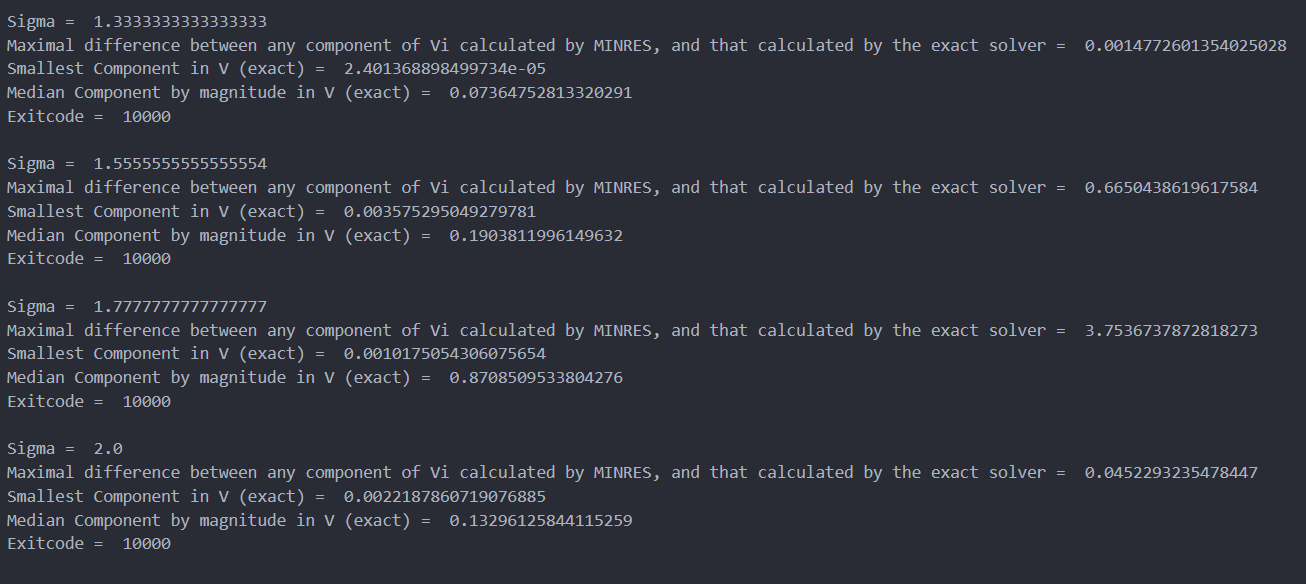


## Sigma = 10 values

### Eigenvector Results



### Vi vector calculation Results

# Conclusions

The eigenvalue computed by the final sigma value in the window (here, 2) seems to yield an eigenvalue completely different from what is near the window. This appears to be true for all 3 sigma cases, across the three eta values. It is interesting to note that for 10 sigma values, the eigenvalues corresponding to the 9th sigma value (in the window between 0 to 2) approximates the first eigenvalue outside the window, whereas the first two sigma values seem to give junk eigenvalues (negative). I think there is some more insight to be gleaned from comparing the eigenvalue data. MINRES doesn’t seem to have any major issues. The error codes mean thus –

* 10000 – after 10000 iterations, the convergence to a tolerance 1e-9 was not achieved. However, the MINRES solution was still very close to the standard solver solution, so I opted to allow this. Bear in mind that the tolerance I am using is 1000 times stricter than the typical tolerance used (1e-6)
* 0 – Successful exit.