Question 1

奇异值分解的三条应用：

1. 可用于量子动力学分析，通过奇异值分解，降低Hilbert空间的维数，提取主导自由度，对理解系统的演化有很大帮助。例如[1]，在二维哈伯德模型的研究中，Singular Value Decomposition被用来分析强相互作用下的量子态。在本目录/article中也有pdf。
2. 可以用于处理量子信息，Carlos Bravo-Prieto等人在文章[2]中提出了一种量子电路的概念，称为“量子奇异值分解器”(QSVD)。其可以产生二分纯态的奇异值分解。利用其和CNOT构成的电路可以充当原始状态信息的编码器，并存储在其中一方上，将思路反过来就可以用于创建具有精确纠缠结构的随机状态。同时这种电路在量子通信中也有重要应用。在本目录/article中也有pdf。
3. 在机器学习和深度学习领域，SVD也具有重要作用；例如，在数据压缩领域，可以通过奇异值分解删除矩阵中的冗余，提供一种更紧凑的数据表示形式，著名的JPEG、MP3算法等都使用了奇异值分解。
4. 当然，奇异值分解作为最重要的数值算法之一，其应用十分广泛，比较详细的介绍可见Wikipedia[3]。在本目录/article中也有pdf。

Question 2

编写矩阵计算任意矩阵的奇异值分解，矩阵的对角化可用cpplapack中的函数heev(...)或者你熟悉的函数。

1. **注意**：heev(...)得到的特征向量是按行排列的，需要转置之后才能当列向量使用。
2. 检验你的程序正确：(a).矩阵分解成立。(b).U,V幺正。(c).奇异值为正。

定义：设矩阵$A \in \mathbb{C}^{m\times n}$，$A^\dagger A$的特征值为$\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_r > \lambda_{r+1} = ... = \lambda_n = 0$，称$\sigma_i = \sqrt{\lambda_i}(i = 1, ..., r)$为矩阵的奇异值。

SVD分解定理：设矩阵$A \in \mathbb{C}^{m\times n}$，则存在幺正矩阵$U \in \mathbb{C}^{m\times m}, V \in \mathbb{C}^{n\times n}$，使

$$A = U\begin{bmatrix}\mathbf{\Sigma} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}V^{\dagger}，对角矩阵\mathbf{\Sigma} = \begin{bmatrix}\sigma_1 & 0 & \dots & 0\\ 0 & \sigma_2 & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \dots & \sigma_r\end{bmatrix}。$$

证明课上讲的十分清楚，需要注意的是对引理进行一些说明：

引理：

设矩阵$A \in \mathbb{C}^{m\times n}$，则(a).$A^{\dagger}A$与$AA^{\dagger}$的特征值均为非负实数；(b).$A^{\dagger}A$与$AA^{\dagger}$的非零特征值相同，并且具有相同的代数重数。

说明：(a).利用其形式可证明其是半正定矩阵。(b).厄米矩阵可以幺正对角化，因此其特征空间的维数和等于矩阵的维数。

证明过程：

奇异值分解(SVD)(4.5)

定理： 设矩阵 $A \in \mathbb{C}^{m\times n}$, 则

（1）$A^{\dagger}A$与 $AA^{\dagger}$的特征值均为非负实数；

（2）$A^{\dagger}A$与 $AA^{\dagger}$的非零特征值相同，并且有相同的代数重数.

证：（1）省略

（2）设$A^{\dagger}A$的特征值为$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_r > \lambda_{r+1} = \cdots = \lambda_n = 0$. 设$A^{\dagger}A$非零特征值$\lambda_i (i \leq r)$对应的特征向量为$\boldsymbol{x}_i$,

$$A^{\dagger}A\boldsymbol{x}_i = \lambda_i \boldsymbol{x}_i, \qquad i = 1,2,\cdots,r.$$

有 $(AA^{\dagger})A\boldsymbol{x}_i = \lambda_i A\boldsymbol{x}_i$, $i = 1,2,\cdots,r.$

注意到 $A\boldsymbol{x}_i \neq \mathbf{0}$ (?)，所以 λ_i 也是 $AA^{\dagger}$ 的非零特征值. 对$A^{\dagger}A$重复同样的过程即证明（2）中第一部分. 设 $\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_p$ 为 $A^{\dagger}A$ 对应于$\lambda \neq 0$ 的线性无关的特征向量，这 $A\boldsymbol{y}_k (k = 1, \dots, p)$是$AA^{\dagger}$的特征向量. 下面证明$A\boldsymbol{y}_1, A\boldsymbol{y}_2, \dots, A\boldsymbol{y}_p$线性无关.

设 $k_1 A\boldsymbol{y}_1 + \cdots + k_p A\boldsymbol{y}_p = 0 => k_1 A^{\dagger}A\boldsymbol{y}_1 + \cdots + k_p A^{\dagger}A\boldsymbol{y}_p = 0$, $k_1\lambda\boldsymbol{y}_1 + \cdots + k_p\lambda\boldsymbol{y}_p = 0$,

$k_1\boldsymbol{y}_1 + \cdots + k_p\boldsymbol{y}_p = 0 =>$ 所以 $k_i = 0\ (i = 1, \cdots, p)$. (以下省略)

奇异值分解(SVD)(4.5)

定义：设矩阵 $A \in \mathbb{C}^{m\times n}$, 则存在幺正矩阵 $U \in \mathbb{C}^{m\times m}, V \in \mathbb{C}^{n\times n}$，使

$$A = U\begin{bmatrix}\boldsymbol{\Sigma} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}V^\dagger,\ \text{对角矩阵}\ \Sigma = \begin{bmatrix}\sigma_1 & 0 & \cdots & 0\\ 0 & \sigma_2 & \cdots & 0\\ \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \cdots & \sigma_r\end{bmatrix}$$

证：设厄密矩阵 $A^\dagger A$ 的单位正交特征向量为$\boldsymbol{v}_i, i = 1,2,\cdots,n$. 非零奇异值的数为 r.

那么 $\{A\boldsymbol{v}_1, A\boldsymbol{v}_2, \cdots, A\boldsymbol{v}_r\}$ 是 Col$A$ 的一组正交基.

很容易证明$A\boldsymbol{v}_i \cdot A\boldsymbol{v}_j = \lambda_i\delta_{ij}$且对于所有$i > r$, 有$A\boldsymbol{v}_i = 0$.

* Col$A$中的任意向量$\boldsymbol{y}$可以写成$\boldsymbol{y} = A\boldsymbol{x}$的形式，$\boldsymbol{x}$是$\mathbb{C}^n$中的向量，

故$\boldsymbol{x} = c_1\boldsymbol{v}_1 + \cdots + c_n\boldsymbol{v}_n$. 所以

$$\begin{aligned}\boldsymbol{y} &= c_1A\boldsymbol{v}_1 + \cdots + c_rA\boldsymbol{v}_r + c_{r+1}A\boldsymbol{v}_{r+1} + \cdots + c_nA\boldsymbol{v}_n\\ &= c_1A\boldsymbol{v}_1 + \cdots + c_rA\boldsymbol{v}_r + \quad 0 \quad + \cdots + \quad 0\end{aligned}$$

奇异值分解(SVD)(4.5)

定义$\boldsymbol{u}_i = A\boldsymbol{v}_i/\sigma_i, 1 \le i \le r$. 注意$\boldsymbol{u}_i \in \mathbb{C}^m$.

将$\{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_r\}$扩充为$\mathbb{C}^m$中的单位正交基$\{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_r, \boldsymbol{u}_{r+1}, \cdots, \boldsymbol{u}_m\}$

并取$U = \{\boldsymbol{u}_1, \cdots, \boldsymbol{u}_m\}, V = \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n\}$,

$$U^\dagger AV = \begin{bmatrix}\boldsymbol{u}_1^\dagger\\ \vdots\\ \boldsymbol{u}_r^\dagger\\ \vdots\\ \boldsymbol{u}_m^\dagger\end{bmatrix}AV = \begin{bmatrix}\boldsymbol{u}_1^\dagger\\ \vdots\\ \boldsymbol{u}_r^\dagger\\ \vdots\\ \boldsymbol{u}_m^\dagger\end{bmatrix}[A\boldsymbol{v}_1, \cdots, A\boldsymbol{v}_n] = \begin{bmatrix}\boldsymbol{\Sigma} & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix} = \Omega.$$

$$A = U\begin{bmatrix}\boldsymbol{\Sigma} & 0\\ 0 & 0\end{bmatrix}V^\dagger = U\Omega V^\dagger$$

奇异值分解(SVD)(4.5)

$$A = U\begin{bmatrix}\boldsymbol{\Sigma} & 0\\ 0 & 0\end{bmatrix}V^{\dagger}$$

记 U 的前 r 列为 U', V 的前 r 列为V', 则有

$$A = U'\Sigma V'^{\dagger}$$

程序编写：

同样，按找以前的套路，直接编写函数，在文件中调用即可。

整个程序的编写可以分为一下几块：

1. 求得奇异值及$A^{\dagger}A$的特征向量，也就是矩阵V.
   i. 我将使用LAPACK_zheev来进行对角化，其函数原型及简介如下：

◆ zheev()

subroutine zheev (	character	*jobz*,
	character	*uplo*,
	integer	*n*,
	complex*16, dimension(lda, *)	*a*,
	integer	*lda*,
	double precision, dimension(*)	*w*,
	complex*16, dimension(*)	*work*,
	integer	*lwork*,
	double precision, dimension(*)	*rwork*,
	integer	*info*
)		

ZHEEV computes the eigenvalues and, optionally, the left and/or right eigenvectors for HE matrices

Purpose:

```
ZHEEV computes all eigenvalues and, optionally, eigenvectors of a
   complex Hermitian matrix A.
```

Parameters

[in]	JOBZ	JOBZ is CHARACTER*1 = 'N': Compute eigenvalues only; = 'V': Compute eigenvalues and eigenvectors.
[in]	UPLO	UPLO is CHARACTER*1 = 'U': Upper triangle of A is stored; = 'L': Lower triangle of A is stored.
[in]	N	N is INTEGER The order of the matrix A. N >= 0.
[in,out]	A	A is COMPLEX*16 array, dimension (LDA, N) On entry, the Hermitian matrix A. If UPLO = 'U', the leading N-by-N upper triangular part of A contains the upper triangular part of the matrix A. If UPLO = 'L', the leading N-by-N lower triangular part of A contains the lower triangular part of the matrix A. On exit, if JOBZ = 'V', then if INFO = 0, A contains the orthonormal eigenvectors of the matrix A. If JOBZ = 'N', then on exit the lower triangle (if UPLO='L') or the upper triangle (if UPLO='U') of A, including the diagonal, is destroyed.

[in]	LDA	LDA is INTEGER The leading dimension of the array A. LDA >= max(1,N).
[out]	W	W is DOUBLE PRECISION array, dimension (N) If INFO = 0, the eigenvalues in ascending order.
[out]	WORK	WORK is COMPLEX*16 array, dimension (MAX(1,LWORK)) On exit, if INFO = 0, WORK(1) returns the optimal LWORK.
[in]	LWORK	LWORK is INTEGER The length of the array WORK. LWORK >= max(1,2*N-1). For optimal efficiency, LWORK >= (NB+1)*N, where NB is the blocksize for ZHETRD returned by ILAENV. If LWORK = -1, then a workspace query is assumed; the routine only calculates the optimal size of the WORK array, returns this value as the first entry of the WORK array, and no error message related to LWORK is issued by XERBLA.
[out]	RWORK	RWORK is DOUBLE PRECISION array, dimension (max(1, 3*N-2))
[out]	INFO	INFO is INTEGER = 0: successful exit < 0: if INFO = -i, the i-th argument had an illegal value > 0: if INFO = i, the algorithm failed to converge; i off-diagonal elements of an intermediate tridiagonal form did not converge to zero.

ii. 由于zheev会将得到的特征向量存入传入的矩阵中，因此我们可以将$A^\dagger A$的值赋给$V^\dagger$，这样我们让特征向量以行排列，再取复共轭就是最后的结果。（只要将zheev的第一个参数写为LAPACK_COL_MAJOR即可）

iii. 注意各个矩阵的维数一定要对应好，不然会出现奇奇怪怪的错误，我由于Vdag没有定义好，就导致了不管怎么算都只有一个非零特征值的情况🥹。

iv. 还有一个问题是zheev给出的特征值是按从小到大排列的，我们当然可以按照这个顺序来向下进行，只是矩阵Sigma对角元的上半部分是0。但是为了和定义相一致，我们还是选择将矩阵的列交换一下顺序。

2. 下面需要做的就是将得到的Vdag列交换并给出矩阵Sigma。还有就是奇异值的个数。

    i. 奇异值的个数r比较好整，Vdag也好交换，两个循环的事。

3. 最后我们需要给出U.

    i. 首先是U的前r列，直接带入公式即可。

    ii. 后面的通过施密特正交化的方式可以给出。需要注意的是，用以填充的列要与前面的列线性无关。比较好取的是单位向量。这里要判断线性无关比较困难，但选择合适的单位向量并不是难事，可以先随便默认，报错了再调整（概率很小）。

    iii. 需要注意的是，LAPACK_zheev计算的不是特征向量，而是特征向量的

复共轭，否则结果是不对的，至于这一结论，可以在程序"SVD/eigenvector.c"中看到。

iv. 在写程序的过程中，LAPAVK_zheev有个bug,那就是当参数传入的是**LAPACK_COL_MAJOR**时，虽然特征向量是按行排列的，但是变成了原来的复共轭，从下面的程序可以看出：

程序完成后，带入了三个进行运算：

Linear_Algebra > SVD > C eigenvector.c > main(void)

```c
#include <stdio.h>
#include <stdlib.h>
#include <complex.h>
#include <math.h>
#include <lapacke.h>
#include "include/my_SVD.h"

int main(void)
{
    double *E;
    E = (double*)malloc(2*sizeof(double));
    double complex *A;
    A = (double complex*)malloc(4*sizeof(double complex));
    A[0]=0;A[1]=1*I;A[2]=-1*I;A[3]=0;
    m_cprint(A,2,2);
    int info = LAPACKE_zheev(LAPACK_ROW_MAJOR, 'V', 'U', 2, A, 2, E);
    if (info != 0) {
        fprintf(stderr, "LAPACKE_zheev returned info=%d\n", info);
        free(E);
        printf("error!");
    }
    m_print(E,1,2);
    m_cprint(A,2,2);

    A[0]=0;A[1]=1*I;A[2]=-1*I;A[3]=0;
    m_cprint(A,2,2);
    info = LAPACKE_zheev(LAPACK_COL_MAJOR, 'V', 'U', 2, A, 2, E);
    if (info != 0) {
        fprintf(stderr, "LAPACKE_zheev returned info=%d\n", info);
        free(E);
        printf("error!");
    }
    m_print(E,1,2);
    m_cprint(A,2,2);
    free(E);
    free(A);
    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

```
0.0000+i0.0000  0.0000+i1.0000
-0.0000+i-1.0000  0.0000+i0.0000

-1.0000  1.0000

-0.0000+i0.7071  0.0000+i0.7071
-0.7071+i0.0000  0.7071+i0.0000

0.0000+i0.0000  0.0000+i1.0000
-0.0000+i-1.0000  0.0000+i0.0000

-1.0000  1.0000

0.0000+i0.7071  0.7071+i0.0000
0.0000+i-0.7071  0.7071+i0.0000

[1] + Done                    "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<
heaven@heaven:~/Desktop/doc/Linear_Algebra$
```

Result

程序完成后，带入了三个进行运算：

1. 课件上的例题，结果稍有不同，但都是对的，也就是说U、v对应向量之间可以同时

取相反数。

Terminal Help

README.md U | my_SVD.c 1, U | my_SVD.h U

Linear_Algebra > SVD > my_SVD.c > ...

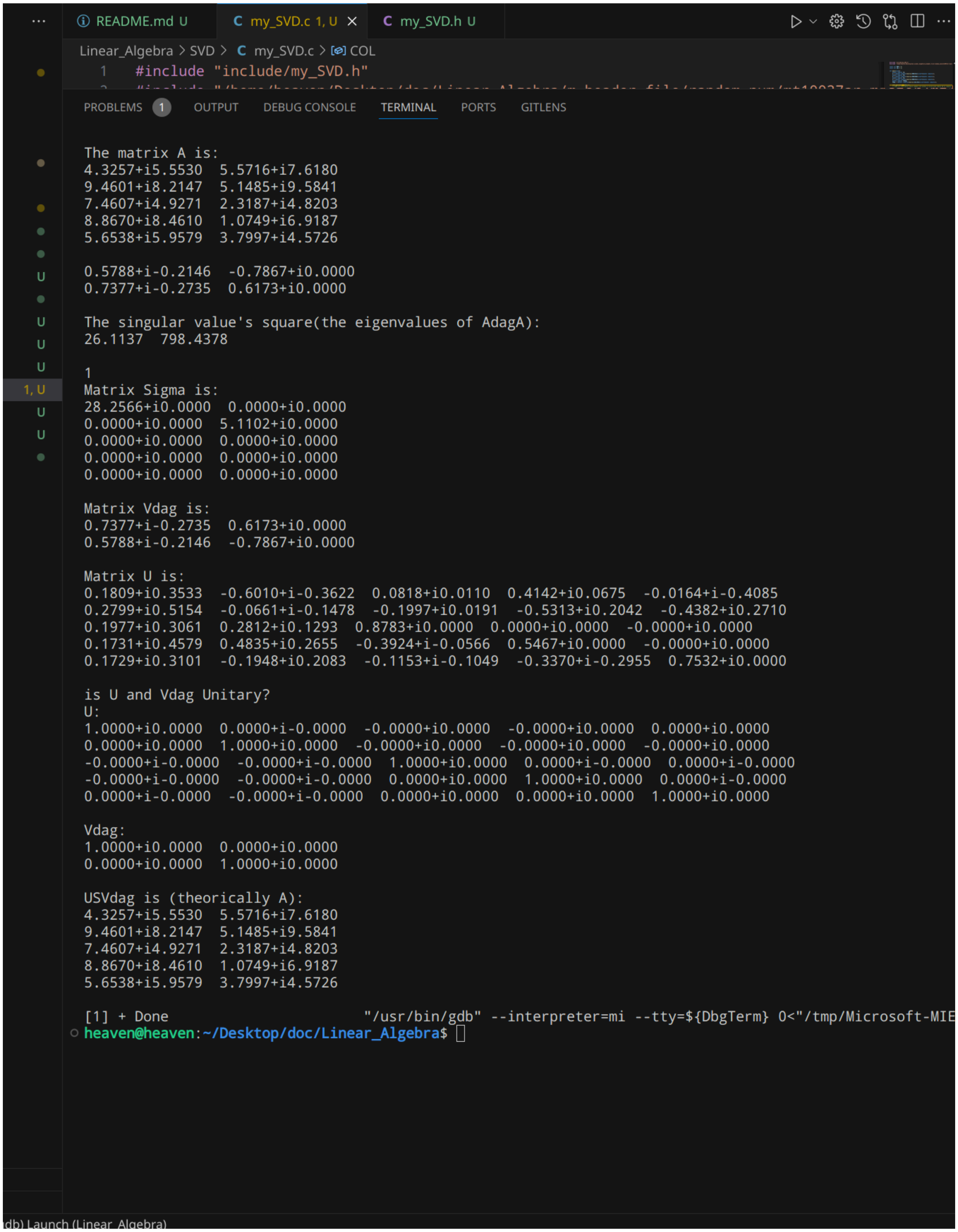
```c
#include "include/my_SVD.h"
#include "/home/heaven/Desktop/doc/Linear_Algebra/m_header_file/random_n
const int ROW = 2;
```

PROBLEMS 1 | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS | GITLENS

```
The matrix A is:
4.0000+i0.0000  11.0000+i0.0000  14.0000+i0.0000
8.0000+i0.0000  7.0000+i0.0000  -2.0000+i0.0000

0.6667+i0.0000  -0.6667+i0.0000  0.3333+i0.0000
0.6667+i0.0000  0.3333+i0.0000  -0.6667+i0.0000
0.3333+i0.0000  0.6667+i0.0000  0.6667+i0.0000

The singular value's square(the eigenvalues of AdagA):
-0.0000  90.0000  360.0000

1
Matrix Sigma is:
18.9737+i0.0000  0.0000+i0.0000  0.0000+i0.0000
0.0000+i0.0000  9.4868+i0.0000  0.0000+i0.0000

Matrix Vdag is:
0.3333+i0.0000  0.6667+i0.0000  0.6667+i0.0000
0.6667+i0.0000  0.3333+i0.0000  -0.6667+i0.0000
0.6667+i0.0000  -0.6667+i0.0000  0.3333+i0.0000

Matrix U is:
0.9487+i0.0000  -0.3162+i0.0000
0.3162+i0.0000  0.9487+i0.0000

is U and Vdag Unitary?
U:
1.0000+i0.0000  -0.0000+i0.0000
-0.0000+i0.0000  1.0000+i0.0000

Vdag:
1.0000+i0.0000  0.0000+i0.0000  0.0000+i0.0000
0.0000+i0.0000  1.0000+i0.0000  0.0000+i0.0000
0.0000+i0.0000  0.0000+i0.0000  1.0000+i0.0000

USVdag is (theorically A):
4.0000+i0.0000  11.0000+i0.0000  14.0000+i0.0000
8.0000+i0.0000  7.0000+i0.0000  -2.0000+i0.0000

[1] + Done                    "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm
heaven@heaven:~/Desktop/doc/Linear_Algebra$
```

gdb) Launch (Linear_Algebra)

2. 取随机数，m=5,n=2



3. 取随机数，m=2,n=5

un Terminal Help

README.md U | my_SVD.c 1, U | my_SVD.h U

Linear_Algebra > SVD > my_SVD.c > main(void)

```
1   #include "include/my_SVD.h"
```

PROBLEMS 1 | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS | GITLENS

```
The matrix A is:
1.1155+i6.6184  0.0778+i4.2371  1.7220+i1.0791  1.8346+i8.0075  5.7054+i9.8633
8.6708+i6.8950  7.9419+i1.4942  1.9241+i5.9661  0.6674+i3.4469  1.5206+i4.0843

0.0916+i-0.1169  -0.2300+i0.2968  0.2590+i0.3071  -0.5810+i-0.3601  0.4564+i0.0000
-0.1228+i0.4050  -0.0206+i0.0646  0.7290+i-0.2697  0.1859+i-0.3662  -0.2084+i0.0000
-0.4364+i0.3225  0.3167+i-0.6194  -0.0589+i-0.0075  -0.2531+i-0.0827  0.3836+i0.0000
0.3770+i-0.0222  0.3706+i-0.2992  0.1274+i0.3888  -0.3530+i-0.1114  -0.5711+i0.0000
0.6014+i0.0026  0.3640+i-0.1054  0.2462+i0.0544  0.3804+i0.1071  0.5243+i0.0000

The singular value's square(the eigenvalues of AdagA):
-0.0000  -0.0000  0.0000  97.9421  425.1665

1
Matrix Sigma is:
20.6196+i0.0000  0.0000+i0.0000  0.0000+i0.0000  0.0000+i0.0000  0.0000+i0.0000
0.0000+i0.0000  9.8966+i0.0000  0.0000+i0.0000  0.0000+i0.0000  0.0000+i0.0000

Matrix Vdag is:
0.6014+i0.0026  0.3640+i-0.1054  0.2462+i0.0544  0.3804+i0.1071  0.5243+i0.0000
0.3770+i-0.0222  0.3706+i-0.2992  0.1274+i0.3888  -0.3530+i-0.1114  -0.5711+i0.0000
-0.4364+i0.3225  0.3167+i-0.6194  -0.0589+i-0.0075  -0.2531+i-0.0827  0.3836+i0.0000
-0.1228+i0.4050  -0.0206+i0.0646  0.7290+i-0.2697  0.1859+i-0.3662  -0.2084+i0.0000
0.0916+i-0.1169  -0.2300+i0.2968  0.2590+i0.3071  -0.5810+i-0.3601  0.4564+i0.0000

Matrix U is:
0.2570+i0.6654  -0.5178+i-0.4722
0.4940+i0.4971  0.6758+i0.2283

is U and Vdag Unitary?
U:
1.0000+i0.0000  -0.0000+i-0.0000
-0.0000+i0.0000  1.0000+i0.0000

Vdag:
1.0000+i0.0000  0.0000+i-0.0000  -0.0000+i0.0000  0.0000+i-0.0000  0.0000+i0.0000
0.0000+i0.0000  1.0000+i0.0000  0.0000+i-0.0000  0.0000+i0.0000  0.0000+i-0.0000
-0.0000+i-0.0000  0.0000+i0.0000  1.0000+i0.0000  0.0000+i0.0000  -0.0000+i0.0000
0.0000+i0.0000  0.0000+i-0.0000  0.0000+i-0.0000  1.0000+i0.0000  0.0000+i-0.0000
0.0000+i-0.0000  0.0000+i0.0000  -0.0000+i-0.0000  0.0000+i0.0000  1.0000+i0.0000

USVdag is (theorically A):
1.1155+i6.6184  0.0778+i4.2371  1.7220+i1.0791  1.8346+i8.0075  5.7054+i9.8633
8.6708+i6.8950  7.9419+i1.4942  1.9241+i5.9661  0.6674+i3.4469  1.5206+i4.0843

[1] + Done                    "/usr/bin/gdb" --interpreter=mi --tty=${DbgTerm} 0<"/tmp/Mic
heaven@heaven:~/Desktop/doc/Linear_Algebra$
```

(gdb) Launch (Linear_Algebra)

1. J. Tokimoto, S. Ohmura, A. Takahashi, K. Iwano, H. Okamoto, "New approach to extract important degrees of freedom in quantum dynamics using singular value decomposition: Application to linear optical spectrum in two-dimensional Mott insulators," *arXiv:2312.06300*, 2023. arXiv: cond-mat.str-el. ↩
2. Carlos Bravo-Prieto, Diego García-Martín, José I. Latorre, "Quantum singular value decomposer," *Phys. Rev. A*, vol. 101, issue 6, pp. 062310, Jun 2020, doi: 10.1103/PhysRevA.101.062310. ↩
3. Wikipedia contributors. (2024, May 2). Singular value decomposition. In Wikipedia, The Free Encyclopedia. Retrieved 05:46, May 5, 2024, from Wikipeda_Singular_value_decomposition ↩