

# Problems with CLAPACK SVD routines on OS X

---

*Source:* <http://sci.tech-archive.net/Archive/sci.math.num-analysis/2007-08/msg00047.html>

---

- *From:* "Robert V." <[robert.voyer@xxxxxxxxxx](mailto:robert.voyer@xxxxxxxxxx)>
  - *Date:* Tue, 07 Aug 2007 20:42:55 -0000
- 

Hello,

I apologize if these questions have been addressed before, but I am new to both C++ and LAPACK.

I am trying to write a program to calculate the SVD of a term-document matrix. Since I have no experience with Fortran, I thought it best to do this in C/C++. I am doing my development on an Intel Mac running OS X 10.4.10.

I am able to get results using Lapack's dgesvd routine, but only when allocating my array statically. The following is based on some sample code from the netlib site and it works:

```
####  
#include <vecLib/vecLib.h>  
#include "f2c.h"  
#include <iostream>  
  
#define SIZE 4  
  
using std::cout;  
using std::endl;  
  
int main() {  
    char JOBU;  
    char JOBVT;  
  
    int i;  
  
    integer M = SIZE;  
    integer N = SIZE;  
  
    integer LDA = M;  
    integer LDU = M;  
    integer LDVT = N;  
    integer LWORK;  
    integer INFO;
```

## Problems with CLAPACK SVD routines on OS X

```
double a[SIZE*SIZE] = { 16.0, 5.0, 9.0 , 4.0, 2.0, 11.0, 7.0 , 14.0,
3.0, 10.0, 6.0, 15.0, 13.0, 8.0, 12.0, 1.0};
double s[SIZE];
double wk[201];
double uu[SIZE*SIZE];
double vt[SIZE*SIZE];

JOBU = 'A';

JOBVT = 'A';

LWORK = 201;

/* Subroutine int dgesvd_(char *jobu, char *jobvt, integer *m,
integer *n,
double real *a, integer *lda, double real *s, double real *u,
integer *
ldu, double real *vt, integer *ldvt, double real *work, integer
*lwork,
integer *info)
*/

dgesvd_( &JOBU, &JOBVT, &M, &N, a, &LDA, s, uu,
&LDU, vt, &LDVT, wk, &LWORK, &INFO);

cout << "\nINFO=" << INFO;

for ( i= 0; i< SIZE; i++ ) {
cout << "\ns[ " << i << " ] = " << s[i];
}

cout << endl;

return 0;
}
####
```

The following program is based on the sample above, but it differs in that I have to allocate my arrays dynamically. Unfortunately, I can't seem to get it to run without seg faulting during the call to dgesvd:

```
####
#include <vecLib/vecLib.h>
#include <iostream>
#include "f2c.h"
#include <fstream>
#include <string>
```

## Problems with CLAPACK SVD routines on OS X

```
#include <sstream>

using std::cout;
using std::endl;
using std::ifstream;
using std::istringstream;
using std::cerr;
using std::string;
using std::getline;
using std::ios;

#define SIZE 4

int main(int argc, char **argv) {

    if (argc < 2) {
        cout << "Error. No input file" << endl;
        exit(-1);
    }

    int number = 0;
    ifstream infile(argv[1]);

    cout << "Opening " << argv[1] << endl;
    if (infile.fail()) {
        cerr << "unable to open file " << argv[1] << " for reading" <<
            endl;
        exit(1);
    }

    // initialize matrix dimension variables
    integer M = 0;
    integer N = 0;

    // pass through infile to determine dimensions
    infile.clear();
    infile.seekg(0, ios::beg);
    string line = "";
    while (getline(infile, line)) {
        N = 0;
        istringstream s(line);
        while(s >> number) {
            N++;
        }
        M++;
    }

    char JOBU;
    char JOBVT;
    char JOBZ;
```

## Problems with CLAPACK SVD routines on OS X

```
integer LDA = M;
integer LDU = M;
integer LDVT = N;
integer INFO = 0;
integer *IWORK = new integer[8*M];

double *s = new double[M];
double *wk = new double[3*M*N];
double *uu = new double[M*N];
double *vt = new double[M*N];

// reset file handle
infile.clear();
infile.seekg(0, ios::beg);

cout << "Initializing a " << M << " x " << N << " matrix." << endl;
double *a = new double[M*N];

cout << "Finished initializing..." << endl;

int row = 0;
int column = 0;

int index = 0;

while (getline(infile, line)) {

    column = 0;
    istringstream s(line);

    while(s >> number && (index < (M*N)) && (row < M) && (column < N))
    {
        if (number > 0) {
            //cout << "A[" << row << "][" << column << "] = " << number <<
            endl;
            a[index] = number;
        } else {
            a[index] = 0;
        }
        index++;
        column++;
    }
    row++;
}
infile.close();

JOBU = 'A';
JOBVT = 'A';
JOBZ = 'A';

cout << "Initializing LWORK..." << endl;
```

## Problems with CLAPACK SVD routines on OS X

```
integer LWORK = 3*M*N;

/* Subroutine int dgesvd_(char *jobu, char *jobvt, integer *m,
integer *n,
double real *a, integer *lda, double real *s, double real *u,
integer *
ldu, double real *vt, integer *ldvt, double real *work, integer
*lwork,
integer *info)
*/

cout << "Calculating SVD..." << endl;

dgesvd_( &JOBU, &JOBVT, &M, &N, a, &LDA, s, uu, &LDU, vt, &LDVT, wk,
&LWORK, &INFO);

// delete all of our dynamically allocated objects
delete IWORK;
delete s;
delete wk;
delete uu;
delete vt;

//cout << "\n INFO=" << INFO;

//for ( i= 0; i< SIZE; i++ ) {
//cout << "\ns[ " << i << " ] = " << s[i];
//}

//cout << endl;

return 0;
}
####
```

```
g++ svd_test.cpp -I/usr/local/include -framework vecLib -o svd
../svd small.matrix
Opening small.matrix
Initializing a 5 x 15513 matrix
Finished initializing...
Initializing LWORK...
Calculating SVD...
Segmentation fault
```

I suspect that my problem stems from my confusion about row-major vs column-major matrices, but this is a shot in the dark. Any insight would be greatly appreciated.

Thanks!  
Robert

