<u>PROBELM 2 FINAL EXAM:</u>
<u>SAGAR DAM; DNAP</u>

PYTHON FUNCTIONS:

##

(a) Computing Fourier transform of a sample

for finding FFT: np.fft.fft()
finding the interval in frequency space: np.fft.fftfreq()

example: Let we need FT of g(x)

```
sp = np.fft.fft(g)
freq = np.fft.fftfreq(x.size)*2*np.pi/h
sp*=h*np.exp(-complex(0,1)*freq*(a))/(np.sqrt(2*np.pi))
```

(b) Obtaining the QR decomposition of a matrix

numpy.linalg.qr; scipy.linalg.qr()

example: Let we need QR decomposition of matrix A

```
B=np.linalg.qr(A)    # this will give Q at B[1] and R at B[2]
```

(c) Obtaining a million random numbers from a lognormal PDF

numpy.random.Generator.lognormal()

example: Let we need 1000000 random numbers from lognormal distribution with mean=mu and sigma=sgm

```
s = numpy.random.Generator.lognormal(mu, sgm, 1000)
```

(d) Solving an ODE initial value problem using an 8th-order Runge-Kutta Method

scipy.integrate.ode()

example:

```
r = ode(f, jac).set_integrator('zvode', method='rk8')
r.set_initial_value(y0, t0).set_f_params(2.0).set_jac_params(2.0)
```

(e) Obtaining the singular value decomposition of a matrix

numpy.linalg.svd(A)
this will give singular value decomposition of matrix A.

(f) Sampling a 548-dimensional PDF

(g) Solving an initial value problem for an ODE using adaptive step-size control:

```
scipy.integrate.LSODA(fun, t0, y0, t_bound, first_step=None, min_step=0.0,
max_step=inf, rtol=0.001, atol=1e-06, jac=None, lband=None,
```

uband=None, vectorized=False)

or
scipy.integrate.solve_ivp(func,[t_min,t_max],y_initial,method='LSODA')

Example: Let's solve y'(t)=t with this.

```
import scipy.integrate
import numpy as np

def func(t,y):
return t

y0=np.array([1])
t_min=1
t_max=10
N_max=100
t_min2=np.linspace(t_min,t_max,N_max)
first_step=0.01
solution=scipy.integrate.solve_ivp(func,[t_min, t_max],y0,method='LSODA', atol=1e-4, rtol=1e-6)
```

(h) Integrating a 9-dimensional function using a Monte Carlo method

mcint.integrate(integrand, sampler(), measure, n)
This is not a numpy/scipy function. It is from mcint package.

Example: Let's calculate volume of 9 dimensional unit hypersphere with MC integral

```
import mcint
import random
import math

def integrand(x):
return (x[0]**2 + x[1]**2 + x[2]**2 +...+ x[8]**2)

def sampler():
while True:
x0 = random.random()
x1 = random.random()
x2 = random.random()
.
.
.
x8 = random.random()
if x0**2+x1**2+x2**2+...+x8**2 <= 1:
    yield (x0,x1,...,x8)

result, error = mcint.integrate(integrand, sampler(), measure=(np.pi)**4/np.factorial(4), n=1000000)
```

(i) Solving a boundary value problem for 3 coupled ODEs

scipy.integrate.odeint()

example: Let's integrate: theta'(t)=w, w'(t)= -w/Q+sin(theta)+d*cos(W*t)

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint

def f(y, t, params):
theta, omega = y
Q, d, Omega = params
derivs = [omega,-omega/Q + np.sin(theta) + d*np.cos(Omega*t)]
return derivs
# Parameters
Q = 2.0          # quality factor (inverse damping)
d = 1.5          # forcing amplitude
Omega = 0.65     # drive frequency
# Initial values
theta0 = 0.0     # initial angular displacement
omega0 = 0.0     # initial angular velocity
# Bundle parameters for ODE solver
params = [Q, d, Omega]
# Bundle initial conditions for ODE solver
y0 = [theta0, omega0]
# Make time array for solution
tStop = 200.
tInc = 0.05
t = np.arange(0., tStop, tInc)
# Call the ODE solver
psoln = odeint(f, y0, t, args=(params,))
```

(j) Computing the eigenvalues and eigenvectors of a 10 × 10 complex matrix:

numpy.linalg.eig(), scipy.linalg.eig()

C FUNCTIONS:
##

(a) Computing Fourier transform of a sample

```
fftw_plan fftw_plan_r2r_1d(int n, double *in, double *out,fftw_r2r_kind kind, unsigned flags);
fftw_plan fftw_plan_r2r_2d(int n0, int n1, double *in, double *out,fftw_r2r_kind kind0, fftw_r2r_kind kind1,unsigned flags);
fftw_plan fftw_plan_r2r_3d(int n0, int n1, int n2,double *in, double *out,fftw_r2r_kind kind0,fftw_r2r_kind kind1,fftw_r2r_kind kind2,   unsigned flags);
fftw_plan fftw_plan_r2r(int rank, const int *n, double *in, double *out,const fftw_r2r_kind *kind, unsigned flags);
fftw_plan fftw_plan_many_dft(int rank, const int *n, int howmany,fftw_complex *in, const int *inembed,int istride, int idist,fftw_complex *out, const int *onembed,int ostride, int odist,int sign, unsigned flags);
```

Example:

```
fftw_complex in[N],
out[N];fftw_plan plan;
plan = fftw_plan_dft_1d(N,in,out,FFTW_FORWARD,FFTW_ESTIMATE);fftw_execute(plan);
fftw_destroy_plan(plan);
```

(b) Obtaining the QR decomposition of a matrix:

```
x[, qr$pivot] == Q %*% R.
```

Example:

```
x <- matrix(runif(10), 5, 2);
q <- qr(x);
is.qr(x)   #  FALSE;
is.qr(q)   #  TRUE;
x <- runif(10);
y <- rnorm(10);
qr(lm( y~x , qr = TRUE) ) # OK;
qr(lm( y~x , qr = FALSE) ) ;
```

(c) Obtaining a million random numbers from a lognormal PDF (1)

```
log_normal_truncated_ab();
```

Example:

```
for ( i = 0; i < SAMPLE_NUM; i++ )
 {
   x[i] = log_normal_truncated_ab_sample ( mu, sigma, a, b, &seed );
 }
```

(d) Solving an ODE initial value problem using an 8th-order Runge-Kutta Method

(e) Obtaining the singular value decomposition of a matrix (GSL functions)

subroutine sgejsv (JOBA, JOBU, JOBV, JOBR, JOBT, JOBP, M, N, A, LDA, SVA, U, LDU, V, LDV, WORK, LWORK, IWORK, INFO)
SGEJSV More...

subroutine sgesdd (JOBZ, M, N, A, LDA, S, U, LDU, VT, LDVT, WORK, LWORK, IWORK, INFO)
SGESDD More...

subroutine sgesvd (JOBU, JOBVT, M, N, A, LDA, S, U, LDU, VT, LDVT, WORK, LWORK, INFO)
SGESVD computes the singular value decomposition (SVD) for GE matrices More...

subroutine sgesvdq (JOBA, JOBP, JOBR, JOBU, JOBV, M, N, A, LDA, S, U, LDU, V, LDV, NUMRANK, IWORK, LIWORK, WORK, LWORK, RWORK, LRWORK, INFO)
SGESVDQ computes the singular value decomposition (SVD) with a QR-Preconditioned QR SVD Method for GE matrices More...

subroutine sgesvdx (JOBU, JOBVT, RANGE, M, N, A, LDA, VL, VU, IL, IU, NS, S, U, LDU, VT, LDVT, WORK, LWORK, IWORK, INFO)
SGESVDX computes the singular value decomposition (SVD) for GE matrices More...

subroutine sggsvd3 (JOBU, JOBV, JOBQ, M, N, P, K, L, A, LDA, B, LDB, ALPHA, BETA, U, LDU, V, LDV, Q, LDQ, WORK, LWORK, IWORK, INFO)
SGGSVD3 computes the singular value decomposition (SVD) for OTHER matrices More...

(f) Sampling a 548-dimensional PDF

(g) Solving an initial value problem for an ODE using adaptive step-size control

(h) Integrating a 9-dimensional function using a Monte Carlo method

gsl_monte_function();

Example: Although this example is not in 9d

```
 struct my_f_params { double a; double b; double c; };
double
my_f (double x[], size_t dim, void * p) {
struct my_f_params * fp = (struct my_f_params *)p;

 if (dim != 2)
  {
   fprintf (stderr, "error: dim != 2");
abort ();
}

return  fp->a * x[0] * x[0]
 + fp->b * x[0] * x[1]
  + fp->c * x[1] * x[1];
```

```
}
gsl_monte_function F;
struct my_f_params params = { 3.0, 2.0, 1.0 };
F.f = &my_f;
F.dim = 2;
F.params = &params;
```

(i) Solving a boundary value problem for 3 coupled ODEs (1)
gsl_odeiv2_system

(j) Computing the eigenvalues and eigenvectors of a 10 × 10 complex matrix
I am not sure about this answer. It was given that these are for complex 16 matrix. Now all real matrices are some reduced complex matrices. But this will work for a large no of cases...

subroutine zgegs (JOBVSL, JOBVSR, N, A, LDA, B, LDB, ALPHA, BETA, VSL, LDVSL, VSR, LDVSR, WORK, LWORK, RWORK, INFO)
ZGEEVX computes the eigenvalues and, optionally, the left and/or right eigenvectors for GE matrices More...

subroutine zgegv (JOBVL, JOBVR, N, A, LDA, B, LDB, ALPHA, BETA, VL, LDVL, VR, LDVR, WORK, LWORK, RWORK, INFO)
ZGEEVX computes the eigenvalues and, optionally, the left and/or right eigenvectors for GE matrices More...

subroutine zgees (JOBVS, SORT, SELECT, N, A, LDA, SDIM, W, VS, LDVS, WORK, LWORK, RWORK, BWORK, INFO)
ZGEES computes the eigenvalues, the Schur form, and, optionally, the matrix of Schur vectors for GE matrices More...

subroutine zgeesx (JOBVS, SORT, SELECT, SENSE, N, A, LDA, SDIM, W, VS, LDVS, RCONDE, RCONDV, WORK, LWORK, RWORK, BWORK, INFO)
ZGEESX computes the eigenvalues, the Schur form, and, optionally, the matrix of Schur vectors for GE matrices More...

subroutine zgeev (JOBVL, JOBVR, N, A, LDA, W, VL, LDVL, VR, LDVR, WORK, LWORK, RWORK, INFO)
ZGEEV computes the eigenvalues and, optionally, the left and/or right eigenvectors for GE matrices More...

subroutine zgeevx (BALANC, JOBVL, JOBVR, SENSE, N, A, LDA, W, VL, LDVL, VR, LDVR, ILO, IHI, SCALE, ABNRM, RCONDE, RCONDV, WORK, LWORK, RWORK, INFO)
ZGEEVX computes the eigenvalues and, optionally, the left and/or right eigenvectors for GE matrices More...

subroutine zgges (JOBVSL, JOBVSR, SORT, SELCTG, N, A, LDA, B, LDB, SDIM, ALPHA, BETA, VSL, LDVSL, VSR, LDVSR, WORK, LWORK, RWORK, BWORK, INFO)
ZGGES computes the eigenvalues, the Schur form, and, optionally, the matrix of Schur vectors for GE matrices More...

subroutine zgges3 (JOBVSL, JOBVSR, SORT, SELCTG, N, A, LDA, B, LDB, SDIM, ALPHA, BETA, VSL, LDVSL, VSR, LDVSR, WORK, LWORK, RWORK, BWORK, INFO)

ZGGES3 computes the eigenvalues, the Schur form, and, optionally, the matrix of Schur vectors for GE matrices (blocked algorithm) More...

subroutine zggesx (JOBVSL, JOBVSR, SORT, SELCTG, SENSE, N, A, LDA, B, LDB, SDIM, ALPHA, BETA, VSL, LDVSL, VSR, LDVSR, RCONDE, RCONDV, WORK, LWORK, RWORK, IWORK, LIWORK, BWORK, INFO)

ZGGESX computes the eigenvalues, the Schur form, and, optionally, the matrix of Schur vectors for GE matrices More...

subroutine zggev (JOBVL, JOBVR, N, A, LDA, B, LDB, ALPHA, BETA, VL, LDVL, VR, LDVR, WORK, LWORK, RWORK, INFO)

ZGGEV computes the eigenvalues and, optionally, the left and/or right eigenvectors for GE matrices More...

subroutine zggev3 (JOBVL, JOBVR, N, A, LDA, B, LDB, ALPHA, BETA, VL, LDVL, VR, LDVR, WORK, LWORK, RWORK, INFO)

ZGGEV3 computes the eigenvalues and, optionally, the left and/or right eigenvectors for GE matrices (blocked algorithm) More...

subroutine zggevx (BALANC, JOBVL, JOBVR, SENSE, N, A, LDA, B, LDB, ALPHA, BETA, VL, LDVL, VR, LDVR, ILO, IHI, LSCALE, RSCALE, ABNRM, BBNRM, RCONDE, RCONDV, WORK, LWORK, RWORK, IWORK, BWORK, INFO)

ZGGEVX computes the eigenvalues and, optionally, the left and/or right eigenvectors for GE matrices More...