

[Help](#)

```
#include <math.h>
#include "moments.h"
#include "mathtools.h"
#include "complex.h"

/* (C) Copr. 1986-92 Numerical Recipes Software
   A2.>$Y0%9j. */
double gammln(double xx)
{
    double x,y,tmp,ser;
    static double cof[6]={76.18009172947146,-86.505
        32032941677,
        24.01409824083091,-1.2317395724
        50155,
        0.1208650973866179e-2,-0.539523
        9384953e-5};
    int j;

    y=x-xx;
    tmp=x+5.5;
    tmp -= (x+0.5)*log(tmp);
    ser=1.000000000190015;
    for (j=0;j<=5;j++) ser += cof[j]/++y;

    return -tmp+log(2.5066282746310005*ser/x);
}

void gauleg(double x1, double x2, double x[],
    double w[], int n)
{
    int m,j,i;
    double z1,z,xm,xl,pp,p3,p2,p1;

    m=(n+1)/2;
    xm=0.5*(x2+x1);
    xl=0.5*(x2-x1);
    for (i=1;i<=m;i++) {
        z=cos(3.141592654*(i-0.25)/(n+0.5));
        do {
            p1=1.0;
```

```

    p2=0.0;
    for (j=1;j<=n;j++) {
        p3=p2;
        p2=p1;
        p1=((2.0*j-1.0)*z*p2-(j-1.0)*p3)/j;
    }
    pp=n*(z*p1-p2)/(z*z-1.0);
    z1=z;
    z=z1-p1/pp;
} while (fabs(z-z1) > EPS);
x[i]=xm-x1*z;
x[n+1-i]=xm+x1*z;
w[i]=2.0*x1/((1.0-z*z)*pp*pp);
w[n+1-i]=w[i];
}
}

double gammadensity(double x, double a, double b)
{
    return exp(-a*log(b) +(a-1)*log(x)-x/b- gammln(a));
}

double factrl(int n)
{
    static int ntop=4;
    static double a[33]={1.0,1.0,2.0,6.0,24.0};
    int j;

    if (n > 32) return exp(gammln((double)n+1.0));
    while (ntop<n) {
        j=ntop++;
        a[ntop]=a[j]*ntop;
    }

    return a[n];
}

double Moments(int n,double r,double sigma,
double t)
{

```

```

double beta, lambda, v, sum, term, sigma2_t;
int i, j;

sigma2_t = SQR(sigma) * t;

if((n==1) && (r==0.))
return 1.;
if((n==2) && (r==0.))
return (2*(exp(sigma2_t)-1-sigma2_t)/pow(sigma, 4.0));
if((n==3) && (r==0.))
return Moments(3, 0.000001, sigma, t);
if((n==4) && (r==0.))
return Moments(4, 0.000001, sigma, t);

v = (r - sigma * sigma / 2) / sigma;
lambda = sigma;
sum = 0.0;

for(j=0; j<=n; j++)
{
term = 1.0;
i = 0;
for(i=0; i<=n; i++)
{
beta = v / lambda;
if((i!=j)) term = term / ((beta+j)*(beta+j) - (beta+i)*(beta+i));
}
term = term * pow(2.0, (double)n);
sum = sum + term * exp((lambda * lambda * j * j / 2 + lambda * j * v) * t);
}

return sum * factrl(n) / pow(lambda, 2.0 * (double)n)
;

}

double logdens(double x, double m, double sg)
{

```

```

double num, den;

num = exp(-(log(x) - m) * (log(x) - m) / (2.0 *
    sg * sg));
den = x * sqrt(2.0 * sg * sg * PI);

return num / den;
}

double Der1Logdens(double x, double m, double sg)
{
    double num, den;

    num = exp(-(log(x) - m) * (log(x) - m) / (2 *
        sg * sg)) * (log(x) - m + sg * sg);

    den = sqrt(2.0 * PI) * (sg * sg * sg) * x*x;

    return -num / den;
}

double Der2Logdens(double x, double m, double sg)
{
    double num, den;

    num = exp(-(log(x) - m) * (log(x) - m) / (2 *
        sg * sg)) * (m * m - sg * sg - 3 * m * sg * sg + 2 * sg
        * sg * sg * sg + (-2 * m + 3 * sg * sg) * log(x) +
        log(x) * log(x));
    den = sqrt(2.0 * PI) * pow(sg, 5.0) * x * x * x;

    return num / den;
}

double Der3Logdens(double x, double m, double sg)
{
    double num, den, fact;

    num = exp(-(log(x) - m) * (log(x) - m) / (2. *

```

```

    sg *sg)) ;
fact = ((((-m*m*m + 3.*m*sg*sg + 6.* m*m*sg*sg
- 6.* sg*sg*sg*sg - 11.* m*sg*sg*sg*sg + 6.* sg
*sg*sg*sg*sg*sg +
    ((3.*m*m - 3.*sg*sg - 12.* m*sg*sg + 1
    1.*sg*sg*sg*sg))* log(x) - 3.*((m - 2.0*sg*sg))*
    log(x)*log(x) + log(x)*log(x)*log(x)))));
den = sqrt(2.0 * PI) * pow(sg, 7.0) * pow(x,4.)
;

return -num *fact/ den;
}

double Der4Logdens(double x, double m, double sg)
{
    double num, den,fact;

    num = exp(-(log(x) - m) * (log(x) - m) / (2 *
    sg *sg));
    fact= pow(m,4.) - 10*pow(m,3)*pow(sg,2.) + po
    w(m,2.)*pow(sg,2.)*(-6 + 35*pow(sg,2.)) +
    pow(sg,4.)*(3 - 35*pow(sg,2.) + 24*pow(sg,4.))
    + m*(30*pow(sg,4.) - 50*pow(sg,6.)) -
    2*(2*pow(m,3.) - 15*pow(m,2)*pow(sg,2) + 5*pow
    (sg,4.)*(3 - 5*pow(sg,2)) + m*pow(sg,2)*(-6 + 3
    5*pow(sg,2))*log(x) +
    (6*pow(m,2) - 6*pow(sg,2.) - 30*m*pow(sg,2.) +
    35*pow(sg,4.))*pow(log(x),2) + (-4*m + 10*pow(
    sg,2.))*pow(log(x),3.) + pow(log(x),4.));
    den = sqrt(2.0 * PI) * pow(sg, 9.0) * pow(x,5.)
    ;

    return num *fact/ den;
}

double momlog(int n, double mean, double var)
{
    return exp(mean * n + var * n * n / 2.0);
}

/*Densit normale e sue derivate per l'uso nella

```

```

    serie di Edgeworth*/

double Normdens(double x, double m, double sg)
{
    double num, den;

    num = exp(-(x - m) * (x - m) / (2.0 * sg * sg)
    );
    den = sqrt(2.0 * sg * sg * PI);

    return num / den;
}

double Der1Normdens(double x, double m, double sg
)
{
    double num, den;

    num = exp(-(x - m) * (x - m) / (2 * sg * sg)) *
        (x - m);

    den = sqrt(2.0 * PI) * (sg * sg * sg) ;

    return -num / den;
}

double Der2Normdens(double x, double m, double sg
)
{
    double num, den;

    num = exp(-(x - m) * (x - m) / (2 * sg * sg)) *
        (-x*x+2*x*m - m*m+sg*sg);
    den = sqrt(2.0 * PI) * pow(sg, 5.0) ;

    return -num / den;
}

double Der3Normdens(double x, double m, double sg

```

```

    )
{
    double num, den, fact;

    num = exp(-(x - m) * (x - m) / (2. * sg * sg))
    ;
    fact = (x-m)*(-x*x+2*x*m-m*m+3*sg*sg);
    den = sqrt(2.0 * PI) * pow(sg, 7.0);

    return num * fact / den;
}

double Der4Normdens(double x, double m, double sg
)
{
    double num, den, fact;

    double x2= x*x;
    double x3= x2*x;
    double x4= x3*x;
    double m2= m*m;
    double m3= m2*m;
    double m4= m3*m;

    double sg2=sg*sg;
    double sg4=sg2*sg2;
    num = exp(-(x - m) * (x - m) / (2 * sg * sg));
    fact= (x4-4*x3*m+m4-6*m2*sg2+3*sg4+6*x2*(m2-sg
        2)-4*x*(m3-3*m*sg2));
    den = sqrt(2.0 * PI) * pow(sg, 9.0);

    return num * fact / den;
}

```

## References