```
   Help
#include "integral.h"
#include "moments.h"
#include <stddef.h>
#include <stdlib.h>

#define FUNC(x) ((*func)(x))
#define FUNK(x) (2.0*(x)*(*func2)(aa+(x)*(x)))
#define FUNKY(x) ((*func1)(1.0/(x))/((x)*(x)))
/* pour le changement de variable qui ramene [a,b] a [0,1]*
    /
#define FUNCG(x) ((*funcg)(a + (b-a)*(x)))

#define FUNCG_VECT(x,n,fx) ((*funcg_vect)( ((a + (b-a)*(x))
    ) , n , fx) )


#define NR_END 1
#define FREE_ARG char*


static int ngauss=-1;
static double *xi,*wi;


/* static double midpnt(double (*func)(double), double a,
    double b, int n)
 * {
 *   double x,tnm,sum,del,ddel;
 *   double s;
 *   int it,j;
 *
 *   if(n==1){
 *     s=(b-a)*FUNC(0.5*(a+b));
 *     return s;
 *   } else {
 *     for(it=1,j=1;j<n-1;j++) it*=3;
 *     tnm=it;
 *     del=(b-a)/(3.0*tnm);
 *     ddel=del+del;
 *     x=a+0.5*del;
*     sum=0.0;
*     for(j=1;j<=it;j++) {
*       sum+=FUNC(x);
*       x+=ddel;
*       sum+=FUNC(x);
*       x+=del;
*
*     }
*     s=( midpnt(func,a,b,n-1) + (b-a)*sum/tnm )/3.0;
*     return s;
*   }
* }
*
* static double midpntbis(double (*func)(double), double
   a, double b, int n){
*   double x,tnm,sum,del,ddel;
*   static double s;
*   int it,j;
*
*   if(n==1){
*     s=(b-a)*FUNC(0.5*(a+b));
*     return s;
*
*   } else {
*     for(it=1,j=1;j<n-1;j++) it*=3;
*     tnm=it;
*     del=(b-a)/(3.0*tnm);
*     ddel=del+del;
*     x=a+0.5*del;
*     sum=0.0;
*     for(j=1;j<=it;j++){
*       sum+=FUNC(x);
*       x+=ddel;
*       sum+=FUNC(x);
*       x+=del;
*     }
*     s=( midpntbis(func,a,b,n-1) + (b-a)*sum/tnm )/3.0;
*     return s;
*   }
* }
*
```

```
* static double midsql(double (*func2)(double), double aa,
    double bb, int n){
*   double x,tnm,sum,del,ddel,b,a;
*   static double s;
*   int it,j;
*
*   b=sqrt(bb-aa);
*   a=0.0;
*   if(n==1) {
*     s=(b-a)*FUNK(0.5*(a+b));
*     return s;
*   } else {
*     for(it=1,j=1;j<n-1;j++) it*=3;
*     tnm=it;
*     del=(b-a)/(3.0*tnm);
*     ddel=del+del;
*     x=a+0.5*del;
*     sum=0.0;
*     for(j=1;j<=it;j++){
*       sum+=FUNK(x);
*       x+=ddel;
*       sum+=FUNK(x);
*       x+=del;
*     }
*     s=( midsql(func2,aa,bb,n-1) + (b-a)*sum/tnm )/3.0;
*     return s;
*   }
* }
*
*
* static double midsqlbis(double (*func2)(double), double
   aa, double bb, int n){
*   double x,tnm,sum,del,ddel,b,a;
*   static double s;
*   int it,j;
*
*   b=sqrt(bb-aa);
*   a=0.0;
*   if(n==1) {
*     s=(b-a)*FUNK(0.5*(a+b));
*     return s;
*   } else {
*     for(it=1,j=1;j<n-1;j++) it*=3;
*     tnm=it;
*     del=(b-a)/(3.0*tnm);
*     ddel=del+del;
*     x=a+0.5*del;
*     sum=0.0;
*     for(j=1;j<=it;j++) {
*       sum+=FUNK(x);
*       x+=ddel;
*       sum+=FUNK(x);
*       x+=del;
*     }
*     s=( midsqlbis(func2,aa,bb,n-1) + (b-a)*sum/tnm )/3.0
   ;
*     return s;
*   }
* }
*
* static double midinf(double (*func1)(double), double aa,
    double bb, int n){
*   double x,tnm,sum,del,ddel,b,a;
*   static double s;
*   int it,j;
*
*   b=1.0/aa;
*   a=1.0/bb;
*   if(n==1){
*     s=(b-a)*FUNKY(0.5*(a+b));
*     return s;
*   } else {
*     for(it=1,j=1;j<n-1;j++) it*=3;
*     tnm=it;
*     del=(b-a)/(3.0*tnm);
*     ddel=del+del;
*     x=a+0.5*del;
*     sum=0.0;
*     for(j=1;j<=it;j++){
*       sum+=FUNKY(x);
*       x+=ddel;
*       sum+=FUNKY(x);
```

```
 *       x+=del;
 *     }
 *     s=( midinf(func1,aa,bb,n-1) + (b-a)*sum/tnm )/3.0;
 *     return s;
 *   }
 * } */


/*----------------------------------------------------------
    -----------*/
double integrale_gauss(double (*funcg)(double), double a,
    double b){
  int i;
  double sum=0.;

  if(ngauss<0) {
    printf("Erreur : vous devez initialiser les points de
    les poids de Gauss.{n");
    exit(-1);
  }

  for(i=1;i<=ngauss;i++)
    {
      sum+=wi[i] * FUNCG(xi[i]);
    }
  /*pour le changement de variable qui ramene [a,b] a [0,1]
    */
  sum*=(b-a);
  return sum;
}

/*----------------------------------------------------------
    -----------*/
void integrale_gauss_vect(void (*funcg_vect)(double,int,
    double *), double a, double b, int dimx, double *sum)
{
  int i,n;
  double *fx;
  double x;

  for(n=0;n<dimx;n++)  sum[n]=0.;
  fx= malloc(sizeof(double)*dimx);

  if(ngauss<0) {
    printf("Erreur : vous devez initialiser les points de
    les poids de Gauss.{n");
    exit(-1);
  }
  for(i=1;i<=ngauss;i++)
    {
      x = xi[i];
      FUNCG_VECT(x,dimx,fx);
      for(n=0;n<dimx;n++)
  sum[n] = sum[n] + wi[i] * fx[n];
    }
  /* pour le changement de variable qui ramene [a,b] a [0,1
    ]*/
  for(n=0;n<dimx;n++)
    sum[n]=sum[n]*(b-a);

  free(fx);

}
/*-----------------------------------------------------------
    -----------*/

void init_gauss(int nbpts)
{
  ngauss = nbpts;
  xi = malloc(sizeof(double)*(ngauss+1));
  wi = malloc(sizeof(double)*(ngauss+1));
  gauleg(0.,1.,xi,wi,ngauss);
}
/*-----------------------------------------------------------
    -----------*/
void free_gauss()
{

  free(xi);
  free(wi);
}
```

References