```
//  file: harmonic_oscillator.cpp
//
//  Functions to generate normalized harmonic oscillator wave functions
//
//  Programmer:  Dick Furnstahl  furnstahl.1@osu.edu
//
//  Revision history:
//      01/24/04  original version, translated from harmonic_oscillator.c
//
//  Notes:
//   * The potential is V(r) = (1/2)m \omega^2 r^2.
//   * The oscillator parameter b is related to m and omega by
//      \hbar\omega = \hbar^2/(m b^2).
//   * We use units with \hbar = 1.
//   * b sets the length scale; so q \equiv r/b is the dimensionless
//      coordinate.
//   * The oscillator state is specified by the radial quantum number
//      n and the angular momentum quantum number l.
//   * Conventions NOT the same as Fetter and Walecka section 57.
//       * Laguerre polynomials not normalized with cube
//   * Normalization is: \int_0^\infty dr [u_{nl}(r)]^2 = 1
//   * Uses gsl library; compile and link with:
//         g++ -c harmonic_oscillator.c
//         g++ -o ... harmonic_oscillator.o -lgsl -lgslcblas -lm
//
//  To do:
//   * should check that n and l are non-negative
//
//*****************************************************************************

// include files
#include <cmath>

#include <gsl/gsl_math.h>
#include <gsl/gsl_sf_gamma.h>
#include <gsl/gsl_sf_laguerre.h>
#include <gsl/gsl_errno.h>

// function prototypes
double ho_radial (int n, int l, double b, double r);
double norm (int n, int l, double b);
double ho_eigenvalue (int n, int l, double b, double m);

//*****************************************************************************
//
//     Calculate the eigenvalue of a harmonic oscillator radial function
//
//  * \epsilon_{nl} = \hbar\omega ( 2(n-1) + l + 3/2 )
//  * \hbar=1 here
//  * \hbar\omega \equiv \hbar^2/(m b^2)
//
//*****************************************************************************
double
ho_eigenvalue (int n, int l, double b, double m)
{
  return ((2. * ((double) n - 1.) + (double) l + 3. / 2.) / (m * b * b));
}

//*****************************************************************************
//
//     Calculate a normalized harmonic oscillator radial function
//
//  * evaluate at position r
//  * u_{nl}(q) = N_{nl} q^{l+1} e^{-q^2/2} L^{l+1/2}_{n-1}(q^2)
//
//*****************************************************************************
double
ho_radial (int n, int l, double b, double r)
{
```



```
  double q = r / b;
  double qsq = q * q;
  double a = (double) l + 1. / 2.;

  return
    norm (n, l, b) * gsl_pow_int (q, (l + 1)) * exp (-qsq / 2.)
    * gsl_sf_laguerre_n ((n - 1), a, qsq);
}

//*****************************************************************************
//
//     Normalization factor for a harmonic oscillator radial function
//
//  * N_{nl} = 2(n-1)!/[b \Gamma(n+l+1/2)]
//  * verified by checking against Mathematica for different n,l,b
//
//*****************************************************************************
double
norm (int n, int l, double b)
{
  double arg = (double) n + (double) l + 1. / 2.;

  return sqrt (2. * gsl_sf_fact ((unsigned) (n - 1)) /
               (b * gsl_sf_gamma (arg)));
}
```