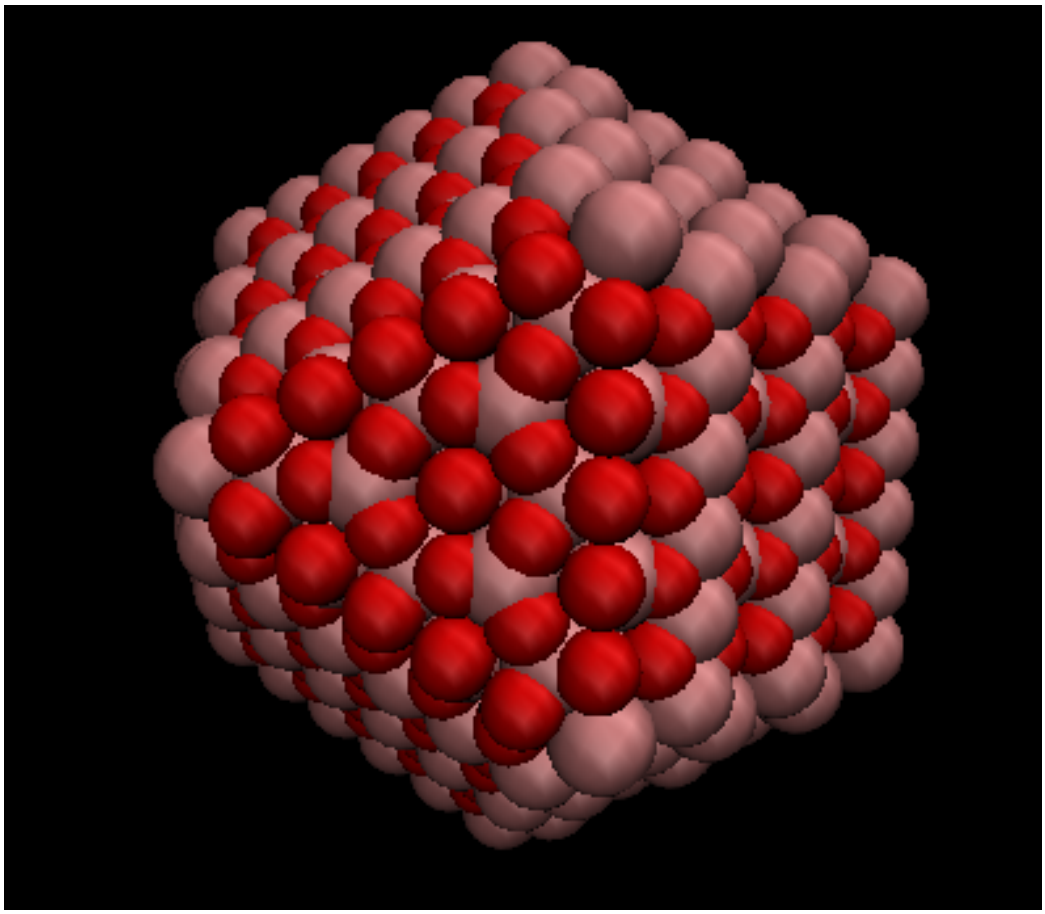
NANO-Crystal : A Web Server for the creation of Nanoparticles for modeling and simulation



Konstantina Karathanou & Zoe Cournia

Biomedical Research Foundation
Academy of Athens

http://83.212.102.179/NanoCrystal/Nano_Spherical/

Contents

1 Introduction

Nanoparticles (NPs) as drug delivery systems have shown significant promise in cancer treatment, where they are used to improve the biodistribution of cancer drugs. Thus, nanoparticles need to be designed with optimal size and surface characteristics in order to decrease side effects and drug toxicity while maximizing treatment impact. Computational approaches assist researchers in this design by modeling nanoparticles to systematize how MNP attributes affect their interaction with cell components as well as with their drug loading. A limiting factor in such modeling studies by the wider scientific community is the absence of a tool that constructs the morphology of nanoparticles. NANO-Crystal is a web-based tool, which constructs spherical nanoparticles of a given radius defined by the user. Apart from the creation of the spherical nanoparticles, we have already developed code that constructs crystal nanoparticles, which is going to be released as an update in NANO-Crystal within 2017. This computational toolbox computes the macroscopic morphology of any periodic crystal by forming different shapes based on Miller indices and is able to make a link between macroscopic morphology and atomistic structure for a periodic crystal, which is a valuable tool for scientists.

2 Theoretical Background

2.1 Methodology

Like a circle, which geometrically is a two-dimensional object, a sphere is defined mathematically as the set of points that are all at the same distance r from a given point, but in three-dimensional space.
First of all, the volume of the nanoparticle and the smaller sphere is computed. In three dimensions, the volume inside a sphere is derived to be:

$$V = \frac{4}{3}\pi r^3 \tag{3.1}$$

According to the Kepler conjecture [93], which is a mathematical conjecture about sphere packing in three-dimensional Euclidean space, no arrangement of equally sized spheres filling space has a greater average density than that of the cubic close packing (face-centered cubic) and hexagonal close packing arrangements. The density of these arrangements is around 74.05%, and that percentage is used to our study for the computing of the number of smaller spheres packed at the surface of the bigger nanosphere.

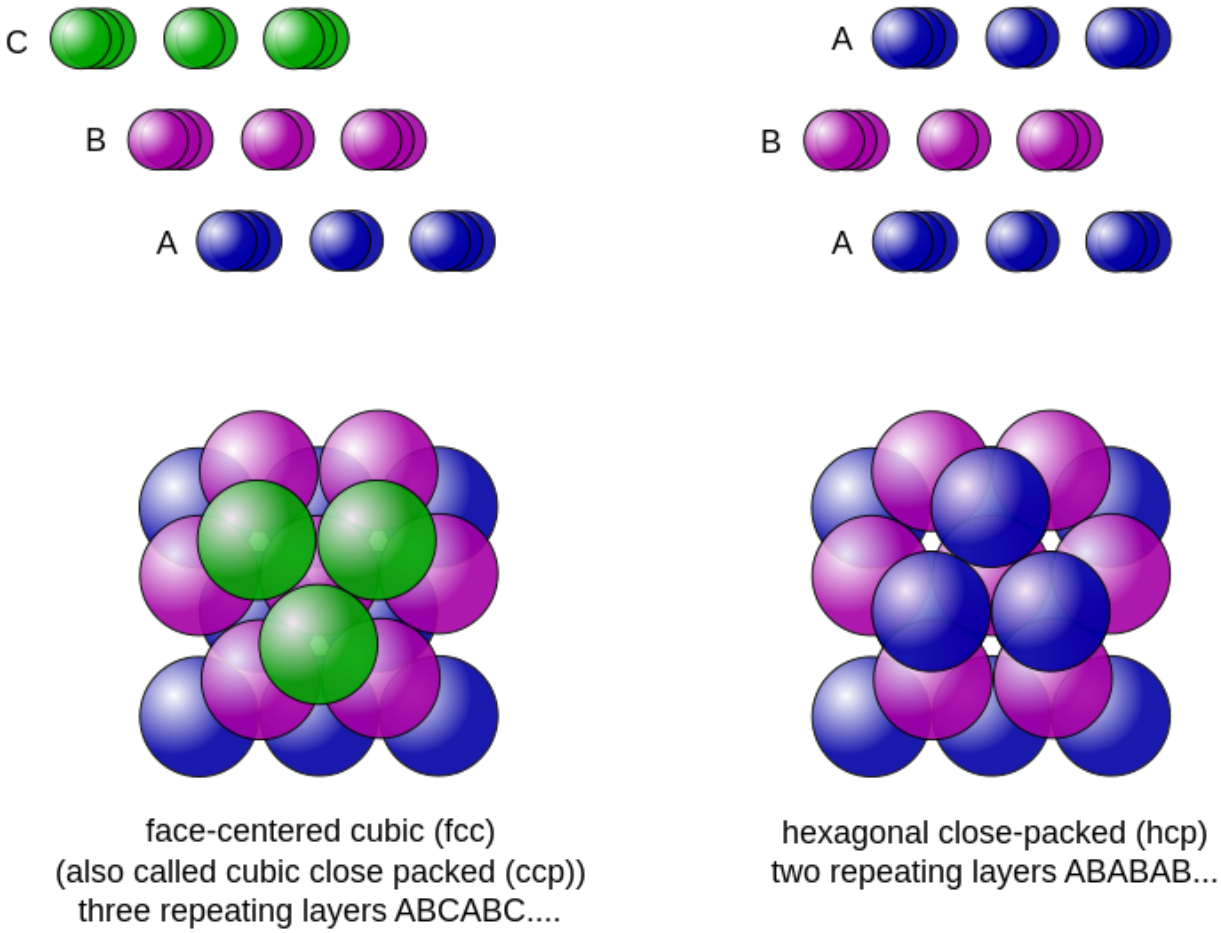


Figure 1: Diagrams of cubic close packing (left) and hexagonal close packing (right). [141]

In geometry, a sphere packing is an arrangement of non-overlapping spheres within a containing space. The spheres considered are usually all of identical size, and the space is usually three-dimensional Euclidean space. However, sphere packing problems can be generalized to consider unequal spheres, n-dimensional Euclidean space (where the problem becomes circle packing in two dimensions, or hypersphere packing in higher dimensions) or to non-Euclidean spaces such as hyperbolic space.

A typical sphere packing problem is to find an arrangement in which the spheres fill as large a proportion of the space as possible. The proportion of space filled by the spheres is called the density of the arrangement. As the local density of a packing in an infinite space can vary depending on the volume over which it is measured, the problem is usually to maximize the average or asymptotic density, measured over a large enough volume.

For equal spheres in three dimensions the densest packing uses approximately 74% of the volume. A random packing of equal spheres generally has a density around 64%.

2.2 Used Functions

Function circle

That function computes the Cartesian coordinates of the spheres that fit at the circumference of the bigger sphere.

```
function function_circle($radius_nanoparticle,$radius_small_spheres,$z)
{
    global $number_of_small_spheres;
    $length=2*pi()*$radius_nanoparticle;
    global $number_one;
    $number_one=round($length/(2*$radius_small_spheres));
    $angle=0.00;
    $angle_increment=(2*pi())/$number_one;
    global $number_two;

    $coordinates=array();
    $coordinates_inverse=array();
    $coordinates_x=array();
    $coordinates_y=array();
    $coordinates_z=array();
    $coordinates_second_layer=array();

for ($i=1; $i<=$number_one; $i++)
{

    $number_two=$number_two+1;
    $x[$number_two]=$radius_nanoparticle*cos($angle);
    $y[$number_two]=$radius_nanoparticle*sin($angle);
    $angle=$angle+$angle_increment;
    $coordinates_x[$i]=number_format((float)$x[$number_two],8,'.','');
    $coordinates_y[$i]=number_format((float)$y[$number_two],8,'.','');
```

Figure 1: Function circle of 'Spherical Coordinates Tool'.

Given the radius of the nanoparticle, the circle circumference is computed (= pi* diameter = 2*pi*radius). Dividing the above with the smaller spheres diameter, the number of spheres that fit at the sphere circumference is computed. We also define an angle increment and polar coordinates are computed and converted to Cartesian coordinates (Equations in Figure 28(i)).

Function sphere

That function computes the Cartesian coordinates of the spheres that fit at the surface of the bigger sphere.

```
function function_sphere($radius_small_spheres, $radius_nanoparticle)
{
    global $number_one;
    global $number_two;
    $number_two=round($number_two);
    $number_three=round($number_one/(4.0));
    $angle = 0.0;
    $angle_increment = (0.5*pi())/$number_three;

  for ($i=1; $i<=$number_three+1; $i++)
  {

   $new_radius = $radius_nanoparticle*cos($angle);
   $new_z = $radius_nanoparticle*sin($angle);
   function_circle($new_radius, $radius_small_spheres,$new_z);
   $angle = $angle +  $angle_increment;

}

}
```

Figure 2: Function sphere of 'Spherical Coordinates tool'.

Similarly, a new angle increment is defined and the function circle is called with parameters such that the spherical coordinates can be computed (Figure 28(ii)) and converted again to Cartesian coordinates in each spherical quadrant.

Polar coordinates:

$$x = r\cos\theta,\ y = r\sin\theta$$

Spherical coordinates:

$$X(u,v) = R * \cos(u) * \cos(v)$$
$$Y(u,v) = R * \sin(u) * \cos(v)$$
$$Z(u,v) = R * \sin(v)$$

$-\pi \le u \le \pi$

$(-\pi/2) \le v \le (\pi/2)$

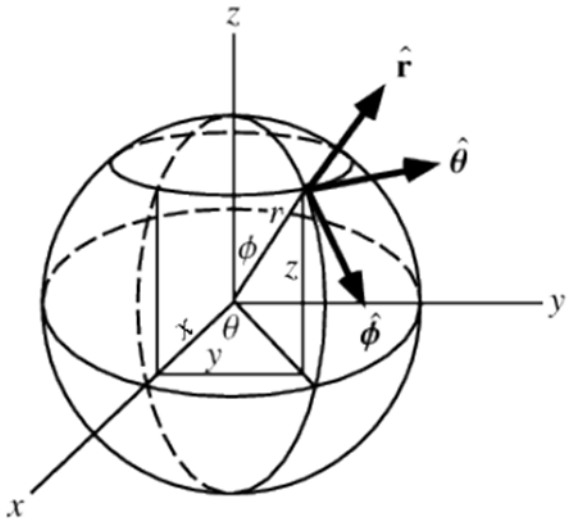


Figure 3: Equations of i) Polar coordinates and ii) Spherical coordinates.

3 Description of the Program

3.1 Construction of spherical nanoparticles (Spherical Coordinates tool)

The "Spherical Coordinates tool", which is a web-based tool, is implemented for the construction of spherical nanoparticles of a given radius. More specifically, our goal is to find the number and the Cartesian coordinates of smaller spheres that fit on the surface of the nanoparticle and visualize the output morphology.

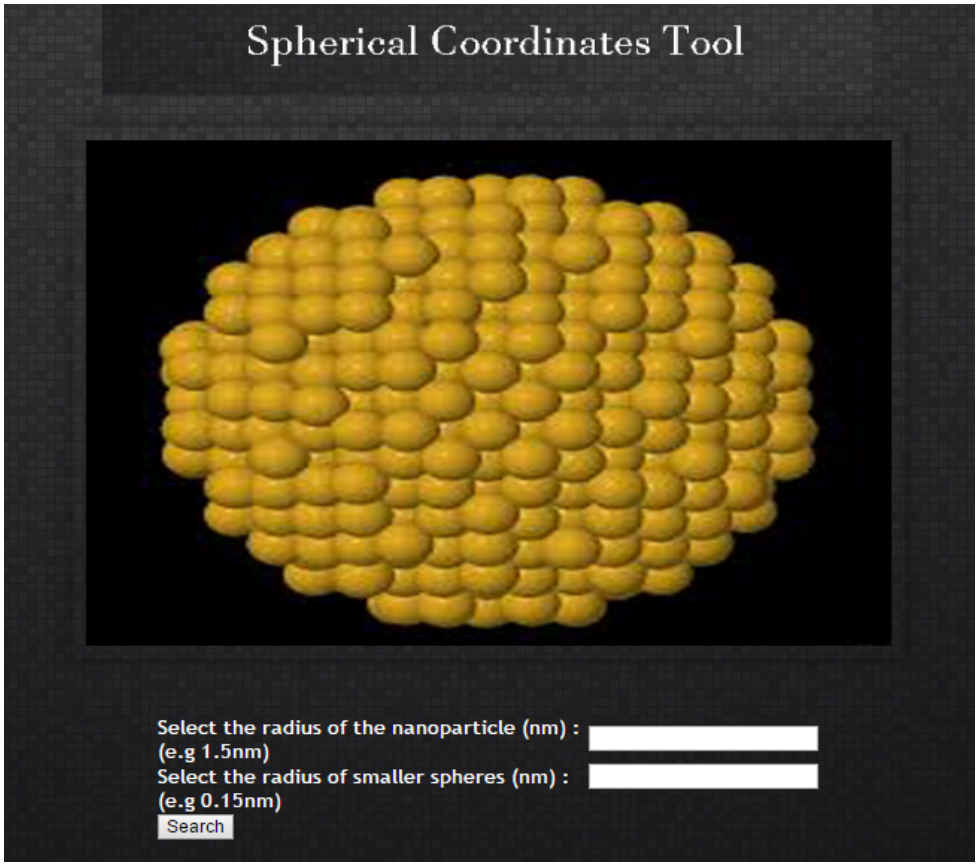


Figure 5: Home page of 'Spherical Coordinates Tool'.

3.2 Program Use

The home page menu (Figure 5), allows two selections for the user:

i) the radius of the nanosphere (nm), and

ii) the radius of smaller spheres (nm), that will cover the surface of the nanoparticle.

The program computes the number of smaller spheres that fit on the bigger surface and the user can download their Cartesian coordinates (output format .xyz).

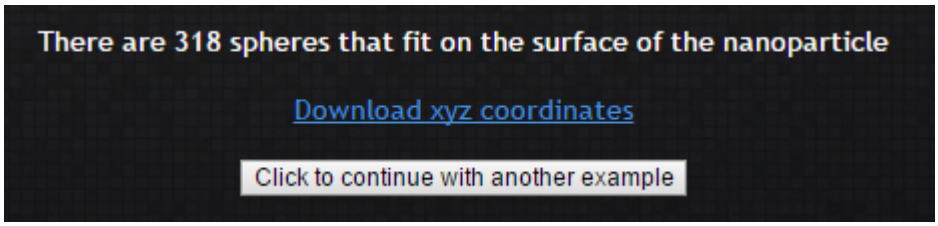


Figure 6: Home page output for 1.5nm and 0.15nm radius of the nanoparticle and the smaller spheres.

3.3 Technologies used

The program code is implemented using PHP server-side scripting language, which is embedded into the HTML and CSS code. JQuery, a cross-platform JavaScript library, is also used. For local host of the webpage tool, the Wamp server is used.

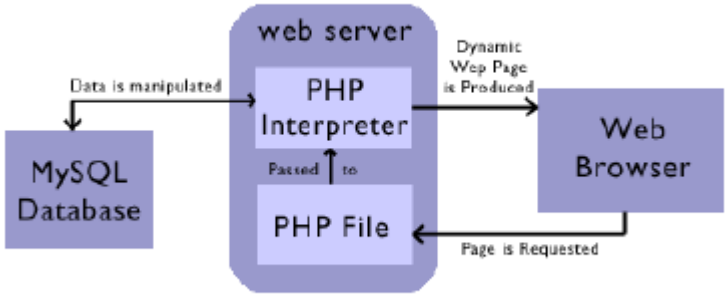


Figure 4: Technologies used for 'Spherical Coordinates tool'.

3.4 Example Results

Radius of the nanoparticle: 1.5nm
Radius of smaller spheres: 0.15nm

- There are 317 spheres that fit on the surface of the nanoparticle

Output visualization (.xyz file format):

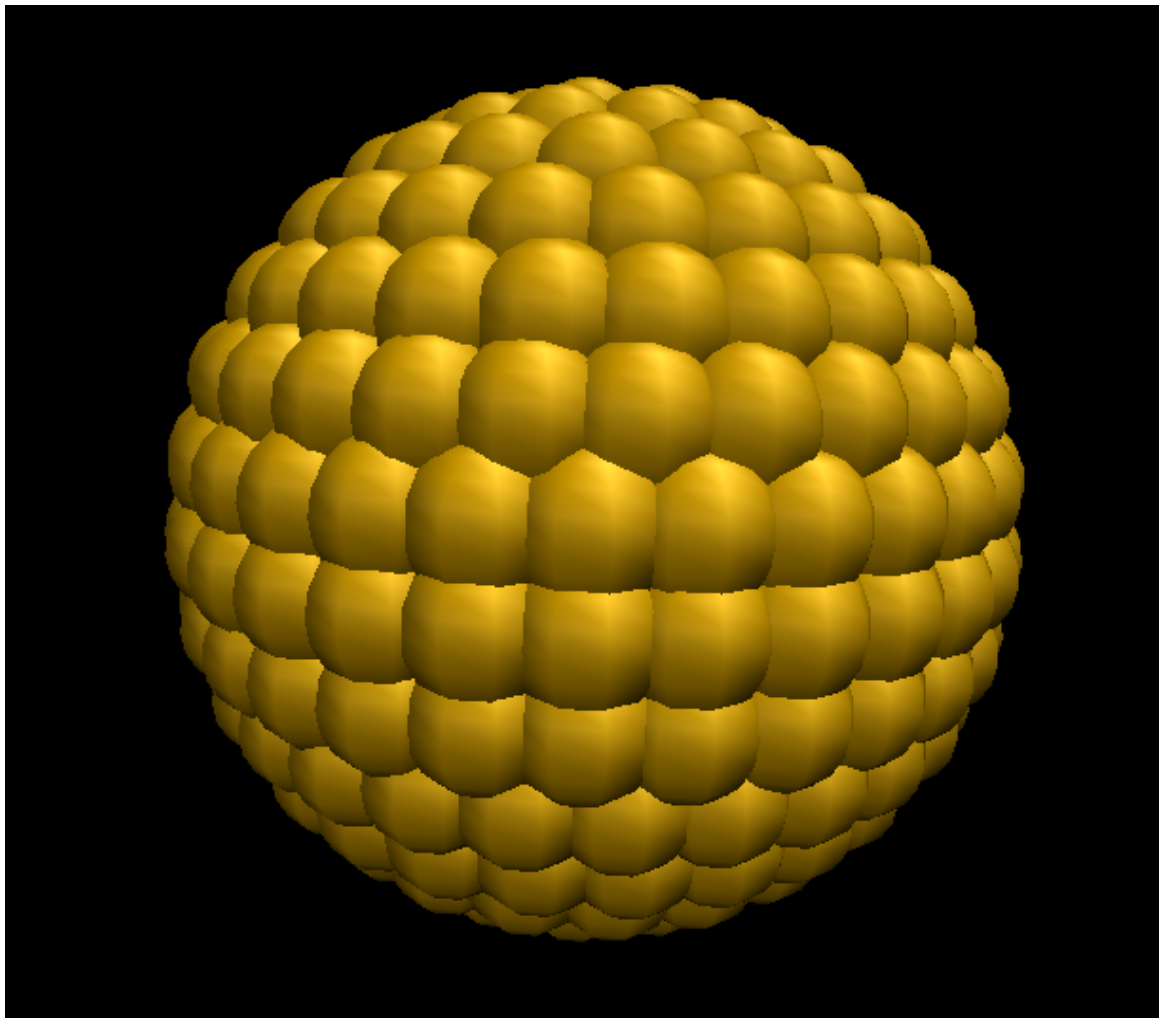


Figure 5: Output visualization of 'Spherical Coordinates tool', 1.5nm nanoparticle-0.15nm smaller spheres.

```
317

C    1.50000000  0.00000000  0.00000000
C    1.46929491  0.30194778  0.00000000
C    1.37843672  0.59153378  0.00000000
C    1.23114516  0.85690232  0.00000000
C    1.03345038  1.08718918  0.00000000
C    0.79344602  1.27296639  0.00000000
C    0.52095788  1.40662820  0.00000000
C    0.22714167  1.48270249  0.00000000
C   -0.07597375 1.49807476  0.00000000
C   -0.37597880 1.45211568  0.00000000
C   -0.66059123 1.34670681  0.00000000
C   -0.91815897 1.18616361  0.00000000
C   -1.13813718 0.97705872  0.00000000
C   -1.31151992 0.72795294  0.00000000
C   -1.43120888 0.44904468  0.00000000
C   -1.49230399 0.15175248  0.00000000
C   -1.49230399 -0.15175248 0.00000000
C   -1.43120888 -0.44904468 0.00000000
C   -1.31151992 -0.72795294 0.00000000
C   -1.13813718 -0.97705872 0.00000000
C   -0.91815897 -1.18616361 0.00000000
C   -0.66059123 -1.34670681 0.00000000
C   -0.37597880 -1.45211568 0.00000000
```

Figure 6: Sample of the xyz output file of 'Spherical Coordinates tool'.

Note that the equation $x^2+y^2+z^2=r^2$ is satisfied by the Cartesian coordinates of the smaller spheres that are packed at the surface of the bigger sphere.

Radius of the nanoparticle: 3nm
Radius of smaller spheres: 1.5nm

- There are 14 spheres that fit on the surface of the nanoparticle

Output visualization (.xyz file format):

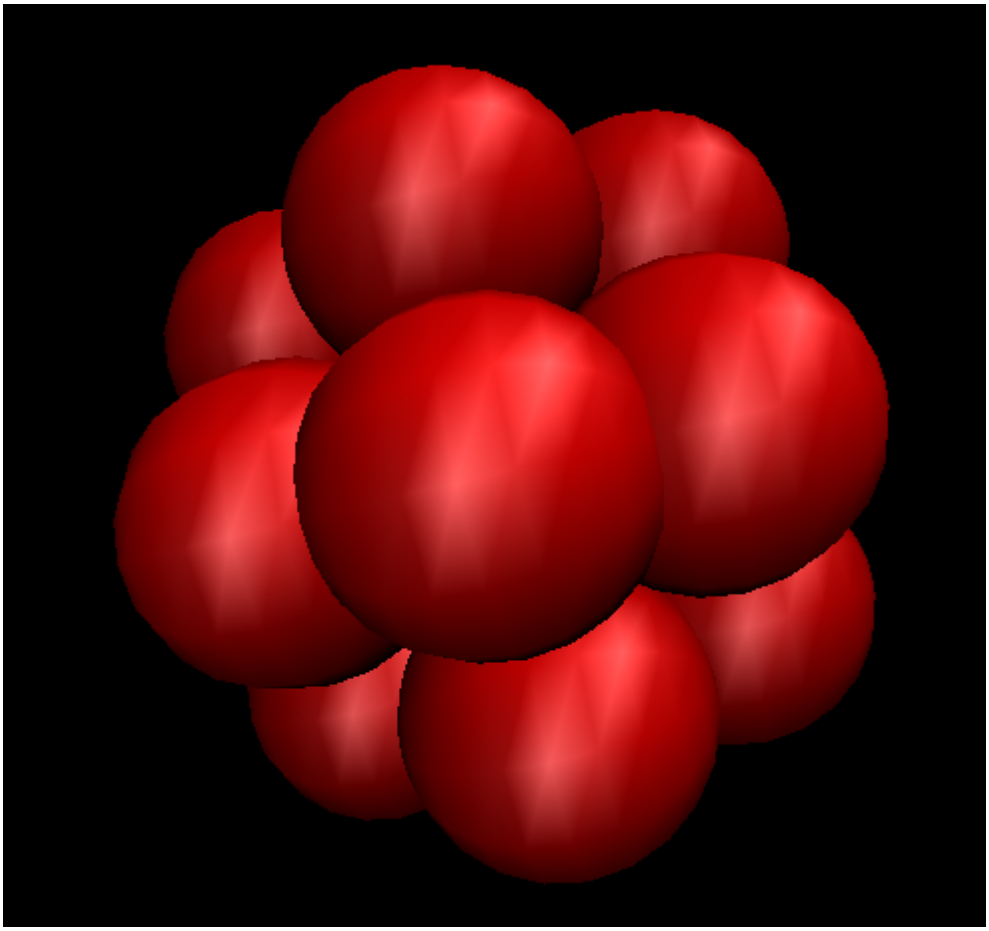


Figure9: Output visualization of 'Spherical Coordinates tool', 3nm nanoparticle-1.5nm smaller spheres.