A CELESTIAL SPHERE MODEL IN GEOGEBRA

Mónica López García[1] and Santiago Cerisola[2]

[1]Elisa Soriano Fischer Secondary School
[2]Department of Mathematics, Pradolongo School

Abstract

The article presents a celestial sphere model made in GeoGebra. The authors describe the mathematical tools necessary to model celestial objects using spherical coordinates and the construction of stars, circles, and meridians. They incorporate a brief description of the construction of constellations using GeoGebra through JavaScript. In addition, the article describes the modeling of the ecliptic and the positions of the Sun and the Earth to understand their apparent movements and the concept of the Zodiac.

Keywords: GeoGebra, science, celestial sphere, rotation motion, constellations

1 INTRODUCTION

Many high school education curricula incorporate the study of the Solar System and the planets as well as the main stars and constellations. This is a transversal topic for many disciplines and can be the motivation source for students of different subjects: Geography students in the study of maps, Classical Studies students due to the connection between mythology and constellations, Physics students in the study of traditional gravitation theory and the Kepler laws, Mathematics students to understand the three-dimensional representation of points either with rectangular or spherical coordinates and Arts and Crafts students, in the construction of small models.

In collaboration with other educational institutions, our teaching groups complement formal education with workshops that, having Astronomy as the linking subject among disciplines, emphasize the importance of understanding the concepts from different points of view. These workshops are mainly intended for students under 16 interested in STEM subjects (Science, Technology, Engineering, and Mathematics). Still, they also complement part of the curriculum of the aforementioned social subjects. The workshops introduce the student to the study of Astronomy through activities that explain the apparent movement of the stars, the movement of the Sun, prominent constellations, the concept of the Zodiac, management of the planisphere, movement of the Moon, equation of time and construction of solar clocks.

In the study of constellations, we often use maps that identify their shapes and locate them to other constellations. In our workshops, we provide the students with blank maps that describe parts of the sky, and they identify the stars that conform to each constellation. These maps, known as planispheres, are usually the result of a stereographic or orthogonal projection of a fixed-radius sphere

in which the stars and other objects in the sky are arranged. This sphere is a classical model for representing the stars that have been used since ancient times by Hipparchus, Aristarchus, Ptolemy and others (Martín, 1990), and it is known in positional astronomy as the celestial sphere (Karttunen et al., 2007). However, we have noticed that the single use of these maps is not enough in learning the position of the stars as objects of the celestial sphere. A 3D model could significantly improve the spatial vision of the student and complement the learning we carry out with flat maps of many other features of the sky such as the size (*angular area*) of each constellation, opposite constellations, and the elliptic curve (*track of the apparent position of the Sun in the sky over a year*).

With this objective, the article summarizes the modeling of a celestial sphere in GeoGebra (Chan, 2013; Hohenwarter, 2004) that we have developed to work on these contents with high school students. The celestial sphere model is an ideal representation of the stars that considers them all to be confined at a fixed distance from the Earth and allows the basic modeling of the apparent daily and annual motion of the stars, planets, and the Sun. The celestial sphere takes a point close to the Polaris star as a north pole (North Celestial Pole) and its antipodal point as a south pole (South Celestial Pole). It considers an imaginary axis that passes through these two poles, which is an extension of the Earth's rotation axis, and assumes that the sphere presents a clockwise rotation movement about this axis opposite to the direction of the Earth. This assumption paves the way to model the apparent motion of the stars. The planet Earth is considered at the center of the sphere, although, due to astronomical distances, the Sun can also be considered the center of the sphere. This is the convention that we adopt in the article for the later purpose of understanding the concept of the Zodiac.

The model we present has been developed in GeoGebra and has the following characteristics:

- incorporates the representation of the 88 constellations of the modern almagest;
- incorporates the position of the ecliptic;
- considers the Sun at the center of the sphere and positions the planet Earth accordingly to model the apparent position of the Sun;
- incorporates the rotation motion of the celestial sphere and the orbital motion of the Earth and the apparent position of the Sun in the ecliptic; and
- incorporates the identification of the zodiac constellations.

However, it assumes the next simplifications that we consider necessary for the student to whom the dynamic construction is directed.

- Models the orbital motion of the Earth as a uniform circular motion; thus the orbit of the Earth is a circle nor an ellipse as stated by the traditional gravitation theory.
- The model does not incorporate the nutation motion nor the precession motion of the Earth, and thus it does not represent the precession of the equinoxes.

The following few sections describe our model, with each section focusing on a different issue. The first section describes the mathematics we use to model the various objects of the sphere. The second section describes how to build the sphere and the programming of constellations using the GeoGebra API in JavaScript. In the third section, we explain how to introduce the functionality in the application

by assigning actions to the already created buttons. As can be seen, the former sections are oriented to those teachers, and GeoGebra users who would like to make a similar application. The last section explains the application from a didactic point of view, emphasizing aspects that might be outlined when using the application with students. For example, the form we see in the celestial sphere is the mirror image of the one that an observer can see in the night sky. We also suggest other didactic uses of the application: (i) the identification of the position of the Sun in the sphere over a year and (ii) the identification of the Zodiac constellations. We finally conclude our work and suggest future development ideas.

2 SPHERE MODELING

2.1 Sphere, poles and meridians

We model the celestial sphere as a sphere with radius r centered at the origin $(0,0,0)$, by $x^2+y^2+z^2 = r^2$ and identify the celestial North and South poles as vectors $\mathbf{p}_n = (0,0,r)^T$ and $\mathbf{p}_s = (0,0,-r)^T$, respectively. In the celestial equator (*intersection of the celestial sphere and the equatorial plane*) we identify 24 hour points $\mathbf{p}_k$ that correspond to the hours of the meridians (also as vectors)

$$\mathbf{p}_k = (r\cos(15k), r\sin(15k), 0)^T \quad \text{for} \quad 0 \leq k \leq 23.$$

With these points we model the celestial meridians with the GeoGebra command `circle` through three given points.

$$\text{Circle}(\mathbf{p}_n, \mathbf{p}_s, \mathbf{p}_k) \ \text{ for } \ 0 \leq k \leq 23.$$

We simulate the motion of the celestial sphere considering a matrix that describes a rotation of angle a ($0 \leq a \leq 360°$) about the z-axis and applying this transformation to the points $\mathbf{p}_k$. We declare this parameter a as a slider and redefine the hour points this way, obtaining a meridian circle that incorporates this transformation. The result is the rotation of the celestial sphere when the slider is animated. We must take into account that the angle is measured in positive sense, and thus the motion of the celestial sphere may be done in contrary sense, a feature that can be done in the properties of the object.

The matrix that describes a rotation $T : \mathbb{R}^3 \to \mathbb{R}^3$ about the z-axis of angle a has three columns corresponding to the images of the standard basis vectors, $\mathbf{e}_1 = (1,0,0)^T, \mathbf{e}_2 = (0,1,0)^T$, and $\mathbf{e}_3 = (0,0,1)^T$.

$$T\mathbf{e}_1 = \begin{pmatrix} \cos(a) \\ \sin(a) \\ 0 \end{pmatrix}, \qquad T\mathbf{e}_2 = \begin{pmatrix} -\sin(a) \\ \cos(a) \\ 0 \end{pmatrix}, \qquad \text{and} \qquad T\mathbf{e}_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

Thus, the matrix A of the linear transformation T that describes a rotation of angle a around the vertical axis is

$$A = \begin{pmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The point $\mathbf{p}_k$ rotates about the z-axis at the given a angle. The coordinates of this point are

$$Ap_k = \begin{pmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r\cos(15k) \\ r\sin(15k) \\ 0 \end{pmatrix} =$$

$$= \begin{pmatrix} r\cos(a)\cos(15k) - r\sin(a)\sin(15k) \\ r\sin(a)\cos(15k) + r\cos(a)\sin(15k) \\ 0 \end{pmatrix} = \begin{pmatrix} r\cos(15k+a) \\ r\sin(15k+a) \\ 0 \end{pmatrix}$$

2.2 Stars and constellations

Each star is identified by its *right ascension* α (*angular separation of a star from the vernal equinox, a specific point of the celestial equator*) and *declination* δ (*angular separation of a star from the equatorial plane* (Karttunen et al., 2007)), which determine the spherical coordinates of the star. With this information we create a star-point object whose rectangular coordinates are

$$\mathbf{s} = \begin{pmatrix} r\cos(\delta)\cos(\alpha) \\ r\cos(\delta)\sin(\alpha) \\ r\sin(\delta) \end{pmatrix}$$

Like the hour points, the star points must rotate on the sphere. Thus, the coordinates that must be considered in the modeling are the result of applying the linear transformation T to the previous vector, obtaining

$$A\mathbf{s} = \begin{pmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r\cos(\delta)\cos(\alpha) \\ r\cos(\delta)\sin(\alpha) \\ r\sin(\delta) \end{pmatrix} = \begin{pmatrix} r\cos(a)\cos(\delta)\cos(\alpha) - r\sin(a)\cos(\delta)\sin(\alpha) \\ r\sin(a)\cos(\delta)\cos(\alpha) + r\cos(a)\cos(\delta)\sin(\alpha) \\ r\sin(\delta) \end{pmatrix}$$

After creating the stars, we build each constellation as polygons through its stars.

2.3 Ecliptic, Sun and Earth

We model the ecliptic (*apparent orbit of the Sun in the sky*) like a circle. To do this, we identify three of its points and build the circle they determine. For two of these points, we consider the Aries point and the Libra point, coinciding with the spring and autumn equinoxes and defined as the intersection of the ecliptic with the celestial equator. Given the modeling we have chosen for our development, the Aries point corresponds to the point p_0 and the Libra point to the point p_{12} in the celestial equator. For the initial instant of the animation of the rotation motion ($a = 0$), these points correspond to $(r, 0, 0)^T$ and $(-r, 0, 0)^T$. We choose the summer solstice in the northern hemisphere for the third point of the construction. This point is defined as the highest one in the ecliptic, and its coordinates, at the initial time of the animation, are those of the image of point $(0, r, 0)^T$ under a rotation of ϵ (*obliquity of the ecliptic*) around the x-axis.

Arguing as in previous paragraphs, we get the fundamental matrix B that defines this rotation. In addition, we notice that the vernal and autumn equinoxes are fixed points for this transformation.

$$B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\epsilon) & -\sin(\epsilon) \\ 0 & \sin(\epsilon) & \cos(\epsilon) \end{pmatrix}$$

Thus, the coordinates of the three points we need for the construction of the ecliptic are those of the application of the composition of the rotation about the x-axis (B) and the rotation about the vertical axis (A) to points $(r,0,0)^T$, $(0,r,0)^T$ and $(-r,0,0)^T$. We get three points in the ecliptic, which move accordingly to the rest of the stars.

$$\mathbf{E}_1 = AB\begin{pmatrix} r \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\epsilon) & -\sin(\epsilon) \\ 0 & \sin(\epsilon) & \cos(\epsilon) \end{pmatrix}\begin{pmatrix} r \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} r\cos(a) \\ r\sin(a) \\ 0 \end{pmatrix}$$

$$\mathbf{E}_2 = AB\begin{pmatrix} 0 \\ r \\ 0 \end{pmatrix} = \begin{pmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\epsilon) & -\sin(\epsilon) \\ 0 & \sin(\epsilon) & \cos(\epsilon) \end{pmatrix}\begin{pmatrix} 0 \\ r \\ 0 \end{pmatrix} =$$

$$= \begin{pmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ r\cos(\epsilon) \\ r\sin(\epsilon) \end{pmatrix} = \begin{pmatrix} -r\sin(a)\cos(\epsilon) \\ r\cos(a)\cos(\epsilon) \\ r\sin(\epsilon) \end{pmatrix}$$

$$\mathbf{E}_3 = AB\begin{pmatrix} -r \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\epsilon) & -\sin(\epsilon) \\ 0 & \sin(\epsilon) & \cos(\epsilon) \end{pmatrix}\begin{pmatrix} -r \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -r\cos(a) \\ -r\sin(a) \\ 0 \end{pmatrix}$$

Once we have identified these points, we build the ecliptic as the circle they determine.

$$Ecliptic = Circle(\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3)$$

To model the apparent Sun (*position of the Sun in the sphere as seen from Earth*), we consider a point in the celestial equator depending on a parameter t ($0 \leq t \leq 360°$) and locate it in the ecliptic applying the rotation B. Then, we rotate it by using the rotation A. This parameter t is modeled with positive velocity to maintain the coherence of the annual motion of the Sun on the celestial sphere. We thus obtain the following.

$$\text{Sun}_A = AB\begin{pmatrix} r\cos(t) \\ r\sin(t) \\ 0 \end{pmatrix} = \begin{pmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\epsilon) & -\sin(\epsilon) \\ 0 & \sin(\epsilon) & \cos(\epsilon) \end{pmatrix}\begin{pmatrix} r\cos(t) \\ r\sin(t) \\ 0 \end{pmatrix} =$$

$$= \begin{pmatrix} \cos(a) & -\sin(a) & 0 \\ \sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} r\cos(t) \\ r\cos(\epsilon)\sin(t) \\ r\sin(\epsilon)\sin(t) \end{pmatrix} = \begin{pmatrix} r\cos(a)\cos(t) - r\sin(a)\cos(\epsilon)\sin(t) \\ r\sin(a)\cos(t) + r\cos(a)\cos(\epsilon)\sin(t) \\ r\sin(\epsilon)\sin(t) \end{pmatrix}$$

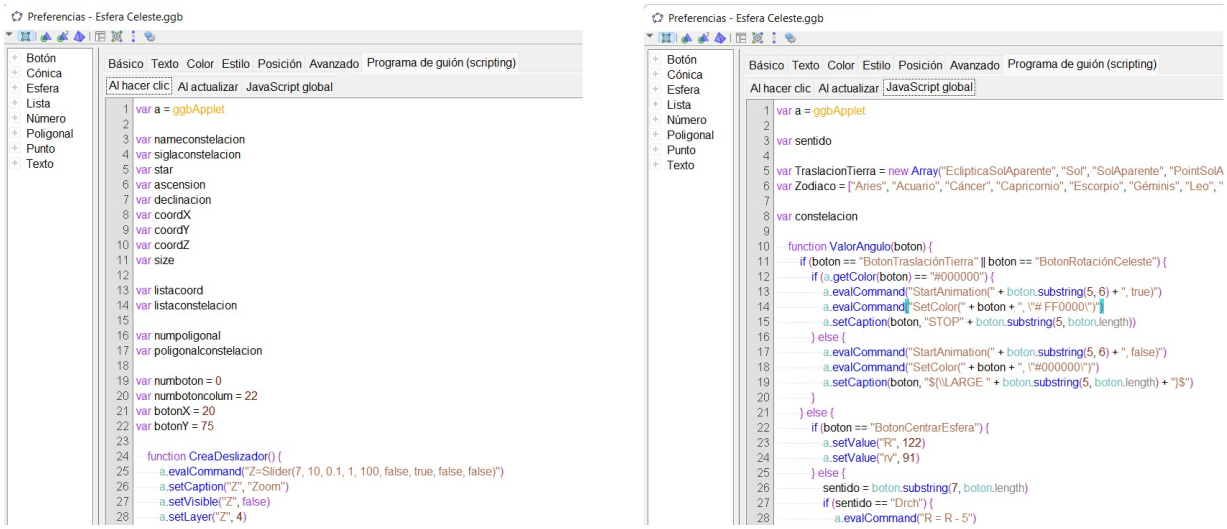
In a similar way we model the orbital motion of Earth, shifting the slider $180°$ and downsizing the radius r to r' for the simulation. We thus obtain.

$$\text{Earth} = \begin{pmatrix} r'\cos(a)\cos(t+180°) - r'\sin(a)\cos(\epsilon)\sin(t+180°) \\ r'\sin(a)\cos(t+180°) + r'\cos(a)\cos(\epsilon)\sin(t+180°) \\ r'\sin(\epsilon)\sin(t+180°) \end{pmatrix}$$

3 CONSTRUCTION OF THE CELESTIAL SPHERE IN GEOGEBRA

Building a celestial sphere in GeoGebra requires a tremendous amount of information due to the coordinates of all the stars that we have included. The identification of the stars with the constellations they belong, the geometric polygonal that defines its classical figure, and the identification of the zodiac constellations. Additionally, the didactic use we have oriented the application required the programming of buttons for simulating the orbital motions of the Earth and simulating the apparent location of the Sun in the sphere. Likewise, we have included buttons to identify the constellations making visible the polygonal of its figure, and other buttons to zoom in, zoom out and rotate the sphere so that it can be analyzed from different positions of an external observer.

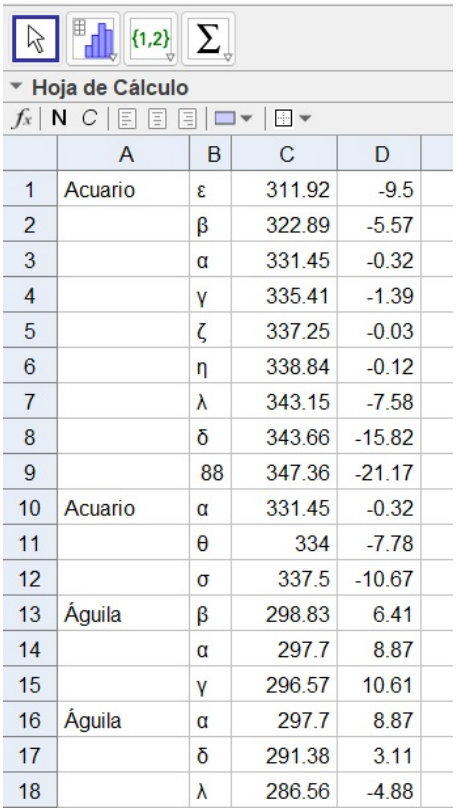
With this purpose, we have compiled the requested information in a Geogebra spreadsheet and written two codes that allow, on the one hand, to automate the construction of all necessary GeoGebra objects and, on the other, provide the functionality to the buttons that the code built.



(a) Creation code **(b)** Functionality code

Figure 1. Main codes

In the spreadsheet, we have organized the information on constellations, assigning each row of the sheet to a star and including the name of the constellation it belongs to in the first column. In the second column, we list the star's name, and in the third and fourth columns, the right ascension and declination of the star. The order of the stars in the spreadsheet has been chosen so that the reading of the information by the creation code permits the creation of the constellation figure. Figure 2 presents the use of the information in the spreadsheet.



	A	B	C	D
1	Acuario	ε	311.92	-9.5
2		β	322.89	-5.57
3		α	331.45	-0.32
4		γ	335.41	-1.39
5		ζ	337.25	-0.03
6		η	338.84	-0.12
7		λ	343.15	-7.58
8		δ	343.66	-15.82
9		88	347.36	-21.17
10	Acuario	α	331.45	-0.32
11		θ	334	-7.78
12		σ	337.5	-10.67
13	Águila	β	298.83	6.41
14		α	297.7	8.87
15		γ	296.57	10.61
16	Águila	α	297.7	8.87
17		δ	291.38	3.11
18		λ	286.56	-4.88

Figure 2. Data in the spreadsheet

We incorporate all the creation instructions in a code that we write in the tab *Onclik* of a button `CreateCelestialSphere`. In this code, we define, using the GeoGebra API, the functions that create the application. In particular, we use scripts of the GBBScript language using the `evalCommand` of the GeoGebra API. Thus we have:

- Function for the construction of sliders.
- Function for the construction of buttons to simulate the rotation motion, the orbital motion, the identification of the ecliptic, and the constellations of the zodiac, zoom in and zoom out.
- Function for the construction of buttons of the rotation console of the celestial sphere.
- Function for the construction of buttons for the identification of each constellation.
- Function for the construction of the sphere the stars lay on.
- Function for the construction of the north and south poles.
- Function for constructing the ecliptic, the apparent Sun, and the Earth.
- Function for the construction of the constellations.

The pseudocode for the construction is shown in the following lines. Notice the way to access the GeoGebra API by defining the variable `a` and the reading of the information from the spreadsheet using a **for** that goes through its rows as long as there is information in the cells of the sheet.

```
var a = gbb.Applet;
...
CreateSliders();
CreateButtons();
CreateRotationConsole();
CreateSphere();
CreateCelestialPoles();
CreateSunEarth();
CreateConstellationButton();
for(var i = 1; a.getValueString("`C"'+ i ) != "`"'; i++){
   CreateConstellation(i);
}
```

The function `CreateSliders()` incorporates GeoGebra commands that are executed with the instruction `a.evalCommand()`. Thus, the creation of a slider may be done using the next instruction.

```
function CreateSliders(){
   ...
   a.evalCommand("t=Slider(0,360,0.1,0.25,100,false,true,false,false)");
   ...
}
```

The functions `CreateButtons()`, `CreateConstellationButton()`, and `CreateRotationConsole()` build the buttons that provide functionality to the application. These functions create the buttons, generate the labels that appear on them and place them at the chosen coordinates as shown below.

The function `CreateSphere()` creates a sphere object and specifies its properties. Necessary properties such as color and opacity by the properties of the rest of the objects so these can be visible.

```
function CreateSphere(){
   ...
   a.evalCommand("Esfera = Sphere((0,0,0),Z)");
   a.evalCommand("SetColor = Esfera,\'#003399\")");
   ...
}
```

Functions `CreateCelestialPoles()` and `CreateSunEarth()` incorporate the mathematics that we described in Section 2, used for the creation of the poles, the ecliptic, the apparent Sun, and the Earth. Additionally, the functions create the necessary objects for the animation of the annual motion of the Sun on the ecliptic.

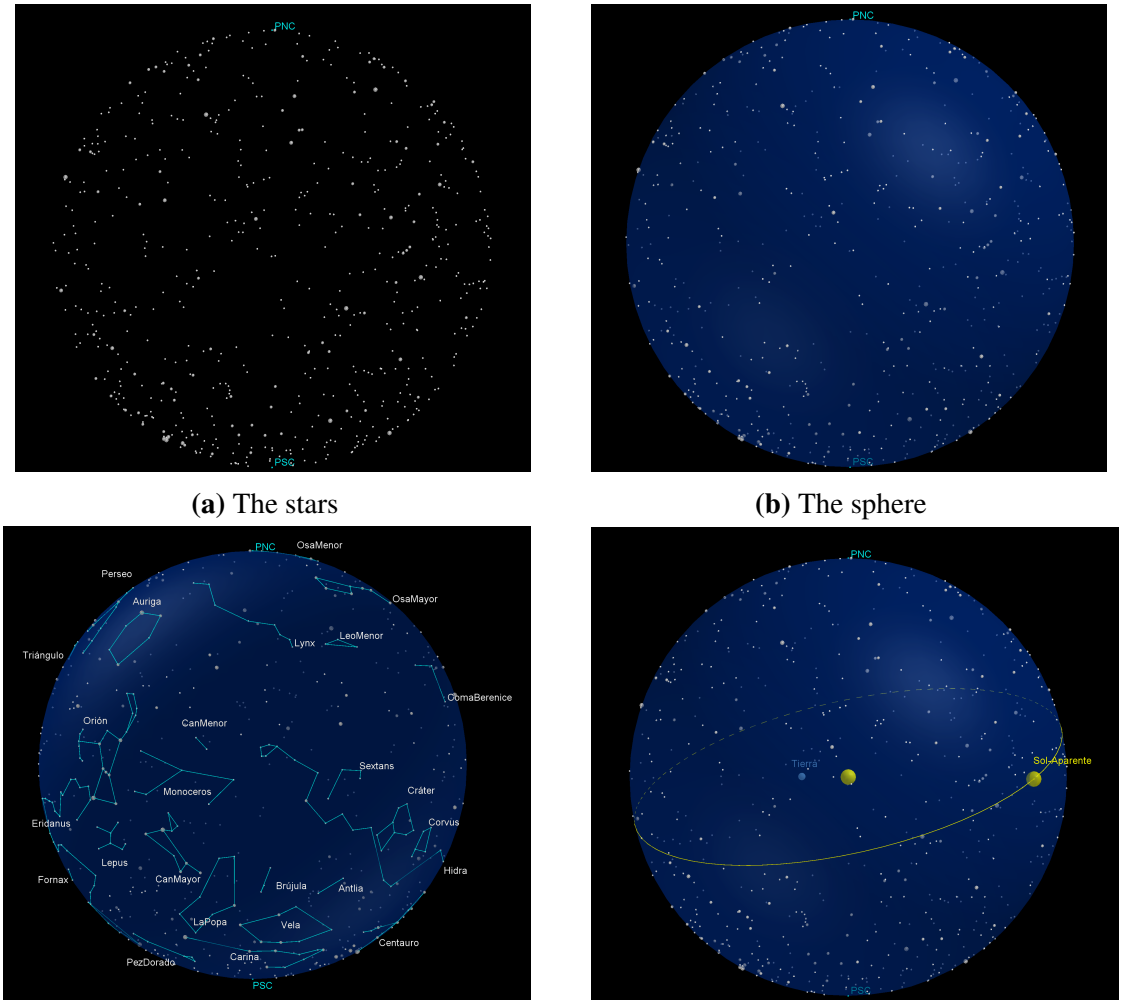
```
function CreateSunEarth(){
   ...
   a.evalCommand("E_1 = ...    ");
   a.evalCommand("E_2 = ...    ");
   a.evalCommand("E_3 = ...    ");
   a.evalCommand("Ecliptica = Circle(E_1,E_2,E_3)");
```

```
    a.setLineThickness("Ecliptica"=2);
    ...
    a.evalCommand("Sun = Sphere((0,0,0),0.5)");
    a.evalCommand("SetColor(Sol, \"#FFFF00\")");
    ...
    a.evalCommand("Tierra=(
       rcos(a)cos(t) + rsen(a)cos(e)sen(t),
       rsen(a)cos(t) + rcos(a)cos(e)sen(t),
       rsen(e)sen(t))";
    )
    ...
    a.evalCommand("SolAparente=Sphere(
       rcos(a)cos(t + 180) + rsen(a)cos(e)sen(t + 180),
       rsen(a)cos(t + 180) + rcos(a)cos(e)sen(t + 180),
       rsen(e)sen(t + 180),0.5)";
    )
    a.setLabelVisible("SolAparente", false);
    ...
}
```

Finally, the code that creates all these objects goes through the rows of the spreadsheet, invoking the function `CreateConstellation()` in each of them. This function creates a list with the coordinates of all the points required to create every polygonal representing each constellation's figure.

```
function CreateConstellation(i){
  ...
  nameconstelacion = a.getValueString("A" + i);
  ascension = a.getValue("C" + i);
  declinacion = a.getValue("D" + i);
  numpoligonal = a.getValue("E" + i);
  coordX = "R*cos("+ascension+" a)*cos(declinacion)";
  coordY = "R*sen("+ascension+" a)*cos(declinacion)";
  coordZ = "R*sen(declinacion)";
  ...
  listacoord = listacoord + "," + "("+coordX "," + coordY "," + coordZ + ")";
  a.evalCommand(listaconstelacion + "{" + listacoord + "}");
  a.evalCommand(poligonalconstelacion + "=" + "Polyline(" + listaconstelacion +
     ")");
  a.evalCommand("SetColor(" + poligonalconstelacion + ", \"#00FFFF\")");
  ...
}
```

The execution of the aforementioned code builds the celestial sphere, the stars of our catalog, the constellations, the Sun, the Earth and the ecliptic as shown in Figure 3.



(a) The stars **(b)** The sphere

(c) The constellations **(d)** The ecliptic

Figure 3. Objects in the celestial sphere

4 Functionality in the celestial sphere

We denote this concept's actions that the already created buttons may execute. The code is hosted in the global JavaScript tab of the button to create the celestial sphere. It incorporates the definition of all the actions executed when clicking on any application button. Later, in the function `ggbOnInit()` we associate each button's action by the `registerObjectClickListener` function of the GeoGebra API. The code we include in this tab is executed any time we open the `.gbb` file, so the way of using GeoGebra to provide the functionality to the application consists in executing the code that creates the objects, saves the construction, closes the file, and opens it again so that the function `ggbOnInit()` could assign actions to the already created buttons. Some part of this initial main code is shown below.

```
function ggbOnInit() {
   a.enableShiftDragZoom(false);
   a.registerObjectClickListener("BotonZodiaco", "ShowHideObjects");
   ...
   a.registerObjectClickListener("BotonZoomLess", "Zoom");
   a.registerObjectClickListener("BotonCentrarEsfera", "CeterRotation");
```

```
    ...
    a.registerObjectClickListener("BotonRotacionCeleste", "Animation");
    a.registerObjectClickListener("BotonAcuario", "ShowHideObjects");
    ...
}
```

As seen, `registerObjectClickListener` associates each button with a function. So it is necessary to define those actions that are executed when the already created button is clicked. These actions are defined by processes whose code may appear before the function `ggbOnInit()` of the button that creates the sphere. For our application, we have defined actions that show and hide objects, animate the orbital motion of Earth and Sun and zoom in and zoom out on the celestial sphere. We next show part of those lines of code that build the functions whose actions are those of showing and hiding objects of the celestial sphere.

```
function ShowHideConstellation(boton,constelacion) {
    ...
    if(a.getVisible(constelacion) == true){
        a.setVisible(constelacion,false);
        a.evalCommand("SetBackgroundColor(boton,\#"0099CC\")"');
    }
    else{
        a.setVisible(constelacion,true);
        a.evalCommand("SetBackgroundColor(boton,\#"669900\")"');
    }
    ...
}
```

The function `Animation()` initializes the sliders that simulates the orbital motion of Earth around the Sun and its apparent position in the sphere. Its main sentences are shown below.

```
function Animation(boton) {
    ...
    if(boton == "botonRotacion"){
        if(a.getColor(boton) == "#000000"){
            a.evalCommand("StartAnimation(a,true)");
            a.evalCommand("SetBackgroundColor(boton,\#"FF0000\")");
            a.setCaption(boton, "${\\LARGE Stop Rotacion}$");
        }
        else{
            a.evalCommand("StartAnimation(a,false)");
            a.evalCommand("SetBackgroundColor(boton,\#"000000\")");
            a.setCaption(boton, "${\\LARGE Rotacion}$");
        }
        ...
    }
...
}
```

The function `Zoom()` permits increase and reduce the size of the sphere within a some limits we have considered appropriate for the use of the application by a student. All the objects in the sphere depend on the radius parameter r that we can modify in the action assign to this button.

```
function Zoom(boton) {
   if(boton == "botonZoomMore"){
      a.evalCommand("R=Min(15,R+1)");
   }
   else{
      a.evalCommand("R=Max(12.5,R-1)");
   }
}
```

5 Description of the application and didactic use

The described developments allowed us to build an application of the celestial sphere that allows a student to identify its different objects. It incorporates buttons for animating, showing, and hiding, zooming in and out on the sphere, and centering it to an initial position of study. The button panels are incorporated in the two-dimensional graphic views of GeoGebra, and the celestial sphere is shown in the central part of the application in the three-dimensional view. The appearance of the application is the one shown in Figure 4. The interested reader can test the performance of the application at `https://www.educa2.madrid.org/web/mlopezgarcia/esfera-celeste`.

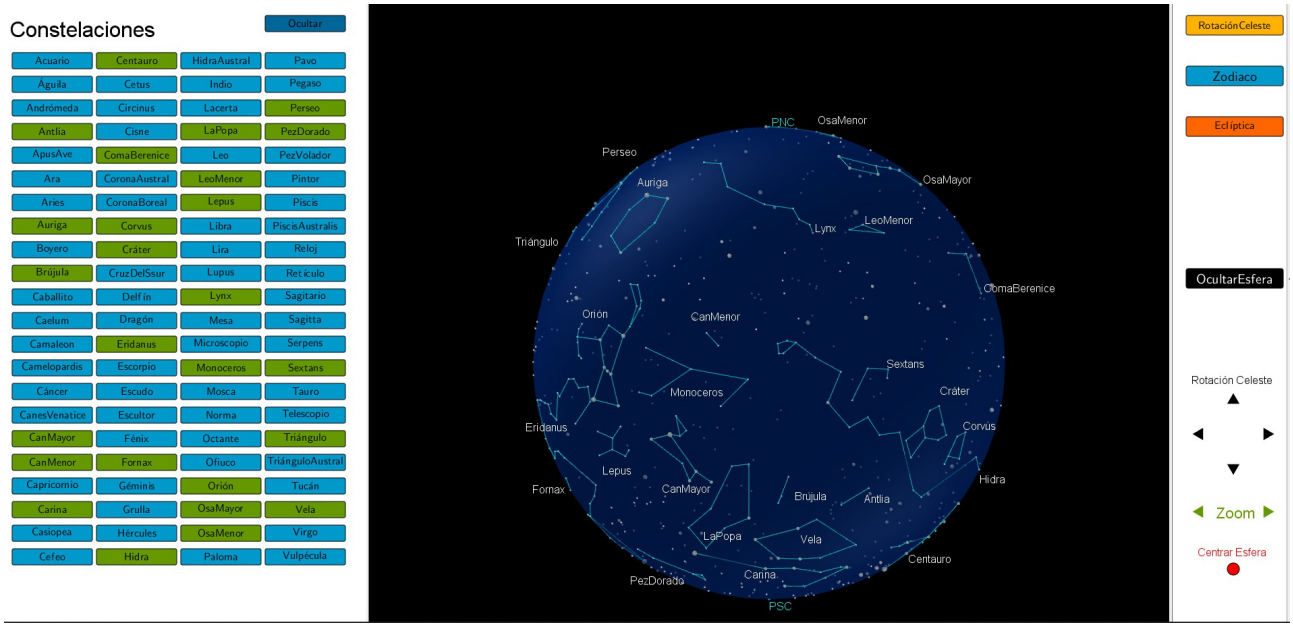


Figure 4. Screenshot of the application Celestial Sphere

The buttons of the panel execute the following actions:

- Hide: hides all the visible constellations.
- Constellation buttons: buttons with the names of the 88 constellations of the modern almagest; they show/hide the corresponding constellation. The button appears green if the constellation is visible and blue if not.
- Celestial Rotation: initiates/stops the animation of the rotation of the celestial sphere.

- Zodiac: shows/hides the constellations of the Zodiac. If any other constellation is visible the action hides that non-zodiac constellation.
- Ecliptic: shows/hides the ecliptic and the button of the motion of the Earth.
- Earth motion: initiates/stops the motion of the Earth.
- Hide Sphere: hides/shows the surface of the sphere.
- Directional buttons: four buttons for the manual rotation of the sphere.
- Zoom: two buttons for zooming the sphere in the 3D view (More and Less).
- Center sphere: returns the celestial sphere to its initial position.

In using the Celestial Sphere application, we suggest highlighting some aspects for a didactic use of the model. Special emphasis must be made on that the students understand that the celestial sphere presents the stars as they would be seen by an observer outside the universe, contrary to the vision that an observer on the Earth has of them. For this reason, the images observed on the sphere are the symmetrical ones that can be seen from the Earth or on a map.

Another essential aspect that is didactically shown with the sphere is the rotation motion of Earth. This movement is explained after the rotation motion of the celestial sphere, which experiments with an inverse rotation motion to that of the Earth. This apparent motion of the sphere is simulated in the application. It allows the students to understand the rotation motion of stars around the Polaris star and the apparent motion of the stars of the equatorial belt from east to west. The movement results in the stars rising from the left side and setting on the right side of an observer located in the northern hemisphere (and looking southwards). Similarly, and complementing the construction with the horizon plane of an observer located at a given latitude, we may appreciate how the visible part of the ecliptic changes depending on the observer's position. For an observer sited on the northern hemisphere, the visible part of the ecliptic is more prominent when the Sun is in the summer solstice and smaller when the Sun is in the winter solstice. The situation is the opposite for an observer in the southern hemisphere.

The apparent position of the Sun is also an important concept that the Celestial Sphere allows showing. The simulation of the orbital motion of the Earth around the Sun incorporated in our application allows us to understand the position of the apparent Sun as one of the extreme points of a segment that, passing through the Sun, has another extreme on the Earth. The simulation of this orbital motion shows how the position of the Sun moves towards greater right ascensions as the year progresses. In particular, it clarifies the definition of the vernal equinox as the point at which the path of the apparent Sun crosses the celestial equator and passes from the southern celestial hemisphere to the northern celestial hemisphere. Similarly, it makes it easy to understand the concept of autumnal equinox when the Sun crosses the celestial equator half a cycle later. Consequently, it can also be observed that the Sun remains above the equator for half a year, while for the other half-year, it is below it. Due to the didactic interest that arises, we have prominently represented the constellations of the Zodiac, i.e., those through which the apparent Sun passes in its annual cycle. The Figures 5 and 6 show this representation.

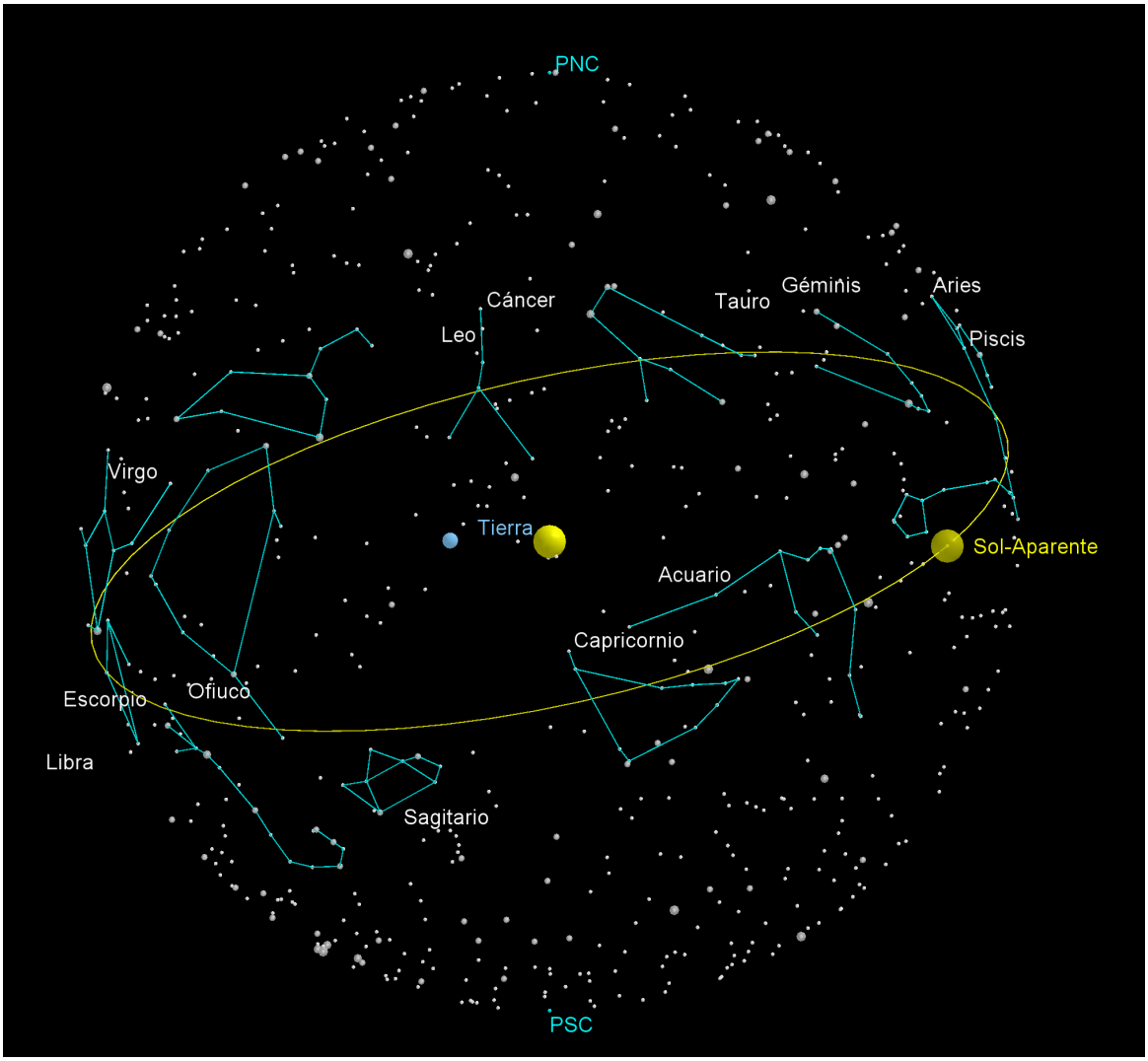


Figure 5. Zodiac Constellations

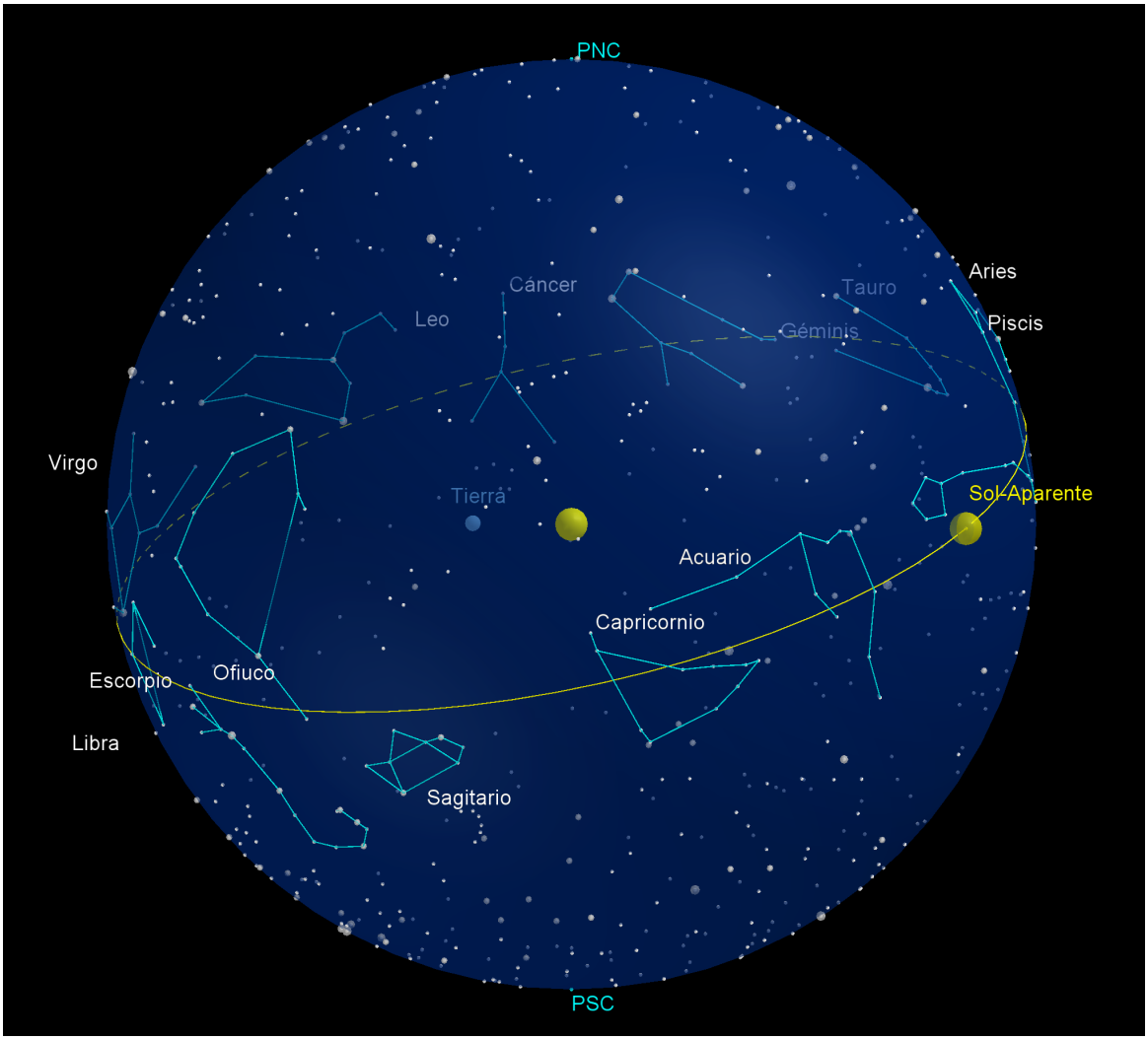


Figure 6. Zodiac Constellations on the Celestial Sphere

6 Conclusion

In the article, we have described the construction of a celestial sphere in GeoGebra and the simulation of its motions using GeoGebra sliders. The intent behind development of the tool was to provide better support for young astronomy students. Our simulation complements traditional materials in learning the positions of the main stars of the sky while providing students with a more hands-on experience through GeoGebra. The work presented here is a starting point for our development team. We are incorporating improvements to simulate the role of Earthly observers and create dynamical planispheres projecting the sphere over different planes. Due to the complexity of such calculations, the work is ideally suited for JavaScript using the GeoGebra API. In the article, we have presented some ideas for doing this based on our experience building with GeoGebra.

References

Chan, Y.-C. (2013). Geogebra as a tool to explore, conjecture, verify, justify, and prove: The case of a circle. *North American GeoGebra Journal*, 2(1).

Hohenwarter, M. (2004). Geogebra software.

Karttunen, H., Kröger, P., Oja, H., Poutanen, M., and Donner, K. J. (2007). *Fundamental astronomy*. Springer.

Martín, A. (1990). *Astronomía*. Springer.

Mónica López García is teacher of Mathematics at Elisa Soriano Fischer Secondary School at Comunidad de Madrid, Spain, and a member of Madrid Kepler Group. Ms. García's research interests include the development of educational tools for the didactics in mathematics and astronomy.

Santiago Cerisola, is Head of the Department of Mathematics at Pradolongo School and part-time professor at Carlos III University in Madrid, Spain. He is also an affiliated researcher at Comillas University in Madrid and a member of the Madrid Kepler Group. His research interests include applied optimization methods and the development of educational tools for didactics in mathematics.

ACKNOWLEDGMENTS

The authors wish to thank Eduardo Martín, from Aula de Astronomía de Fuenlabrada, Madrid, Spain, for his careful editing suggestions. In addition, they would like to thank the NAGJ editors for their support and assistance.