Explaining the code:

This project implements spherical harmonics. Summing up spherical harmonics is like the fourier series, but 3D and for surfaces described in spherical coordinates. So we can approximate surfaces using spherical harmonics.

This is the formula used to approximate a function

$$g(\theta, \phi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} A_{lm} r^l P_l^m(cos(\theta)) e^{im\phi}$$

(Walter Strauss - Page 277)

To understand this formula, let's start with $P_l^m(cos(\theta))$. P_l^m is the associated legendre function. The formula for it is given in the picture below.

$$P_l^m(s) = \frac{(-1)^m}{2^l l!}(1-s^2)^{m/2} \frac{d^{l+m}}{ds^{l+m}}[(s^2-1)^l]. \qquad (17)$$

The function (17) is called the *associated Legendre function*. Notice that it is merely a polynomial multiplied by a power of $\sqrt{1-s^2}$. Also notice that it is finite at $s = \pm 1$.

(Walter Strauss - 273)

This is the function that returns the Legendre functions evaluated at s

```
% -- function that returns Legendre function evaluated at s--
function P = legendreFunction(l, m, s)
    abs_m = abs(m);
    p0 = [1 0 -1]; %this vector represents s^2 + 0s - 1

    p_to_l = 1; % initializes to a constant polynomial

    for i = 1:l
        p_to_l = conv(p_to_l, p0); %this is just polynomial multiplication, but faster
    end

    p_der = p_to_l;
    %taking (l+m) derivatives
    for i = 1:(l + abs_m)
        p_der = polyder(p_der);
    end

    P = ((-1)^abs_m) / (2^l * factorial(l)).*(1 - s.^2).^(abs_m/2).* polyval(p_der, s);
end
```

$$Y_l^m(\theta, \phi) = P_l^{|m|}(\cos\theta)\, e^{im\phi}$$

(Walter Strauss - 275) This is a spherical harmonic. It is a complex valued function and the set of spherical harmonics is a complete set.

This is the function that returns the spherical harmonics evaluated at theta and phi

```
% -- functions that return spherical harmonics evaluated at theta and phi--
function Y = spherical_harmonic(l, m, theta, phi) %takes in inputs l, m, theta, and phi
    % l,m Legendre function evalulated at cos(theta)
    Plm = legendreFunction(l, m, cos(theta));
    Y = Plm.*exp(1i * m * phi);
end
```

The coefficients A_lm are given this double integral inner product:

$$A_{lm} = \frac{2l+1}{4\pi} \frac{(l-m)!}{(l+m)!} \int_0^{2\pi} \int_0^{\pi} g(\theta, \phi) \overline{Y_l^m(\theta, \phi)} \sin(\theta)\, d\theta\, d\phi$$

The constant on the right of the integral is the norm of the spherical harmonics.

This part of the program computes all the coefficients and stores into an array/vector.

```
L = 20; %if you want less terms, change this variable. L = 20 could take a long time.
A_lm = zeros(L + 1, 2*L + 1);
for l = 0:L

    for m = -l:l
        Y = spherical_harmonic(l, m, THETA, PHI);
        Norm = (4*pi / (2*l + 1)) * (factorial(l + abs(m)) / factorial(l - abs(m)));
        A_lm(l+1, m + L + 1) = (1/Norm) * trapz(phi, trapz(theta, g .* conj(Y) .* sin(THETA), 1));
    end
end
```

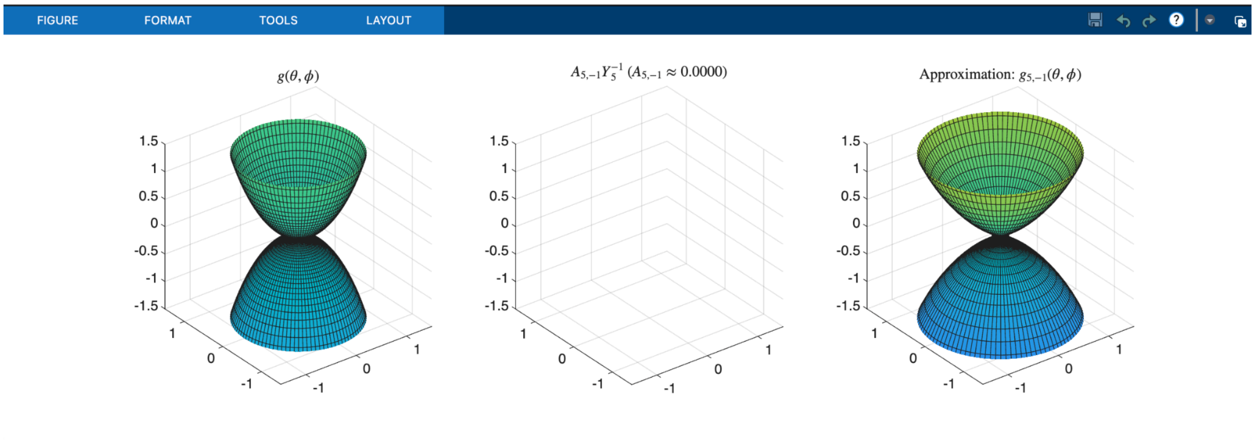
This part of the code is used to plot the partial sum of the series above.

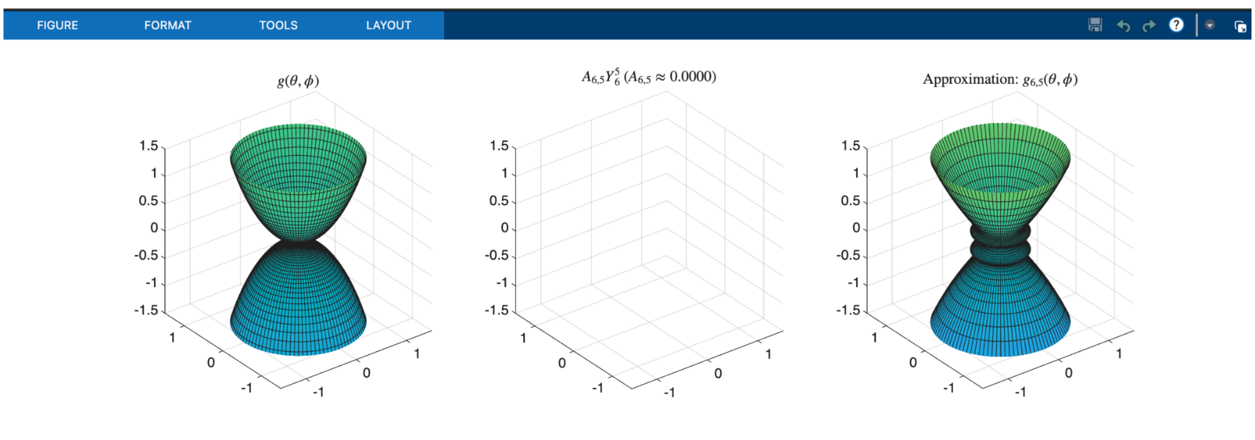
```
g_lm = zeros(size(THETA));
fig = figure('Color', 'White', 'Position', [50, 50, 1200, 400]);
for l = 0:L
    for m = -l:l
        cf = A_lm(l+1, m + L + 1);
        y = spherical_harmonic(l, m, THETA, PHI);

        g_lm = g_lm + (cf * y);

        clf;

        subplot(1,3,1);
        my_plot(THETA, PHI, g);
        title('$g(\theta, \phi)$', "Interpreter", "latex");

        subplot(1,3,2);
        my_plot(THETA, PHI, real(cf * y));
        title(sprintf('$A_{%d,%d} Y_{%d}^{%d}$ ($A_{%d,%d} \\approx %.4f$)',l,m,l,m,l,m,real(cf)),'Interpreter','latex');

        subplot(1,3,3);
        my_plot(THETA, PHI, real(g_lm)); %takes the real part of g_lm.
        title(sprintf('Approximation: $g_{%d,%d}(\\theta, \\phi)$', l, m), 'Interpreter', 'latex');

        drawnow;
        pause(1); %changing how fast you want to iterate over the plots.
    end
end

function my_plot(THETA, PHI, g)
    % spherical to rectangular coordinates
    r = abs(g);
    X = r .* sin(THETA) .* cos(PHI);
    Y = r .* sin(THETA) .* sin(PHI);
    Z = r .* cos(THETA);

    surf(X, Y, Z, g);
    axis equal;
    grid on;

    axis([-1.5 1.5 -1.5 1.5 -1.5 1.5]); %you can change this if you want to see g over a larger domain
end
```

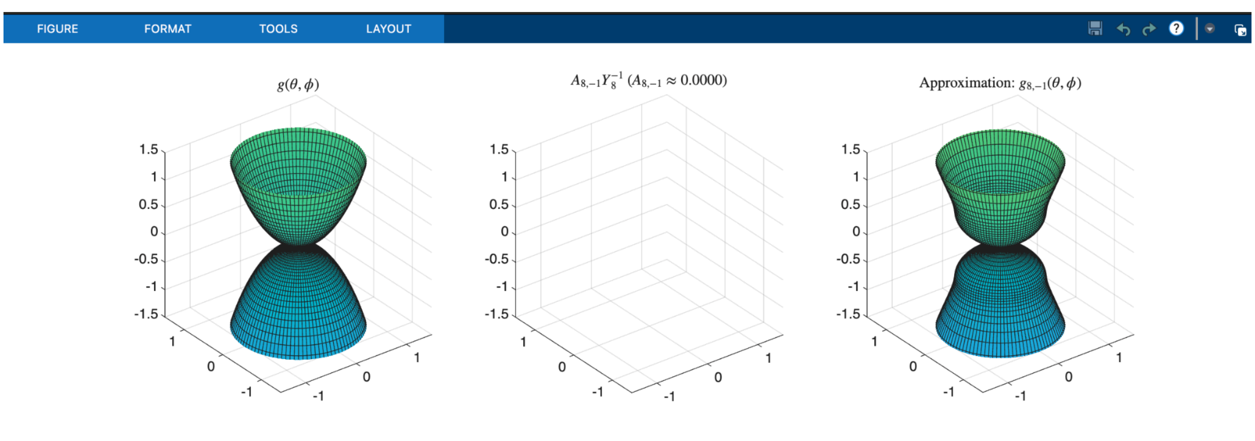
I graphed the real component of cf and g_lm, because our g is real valued (so the complex parts should be 0 or very very low).

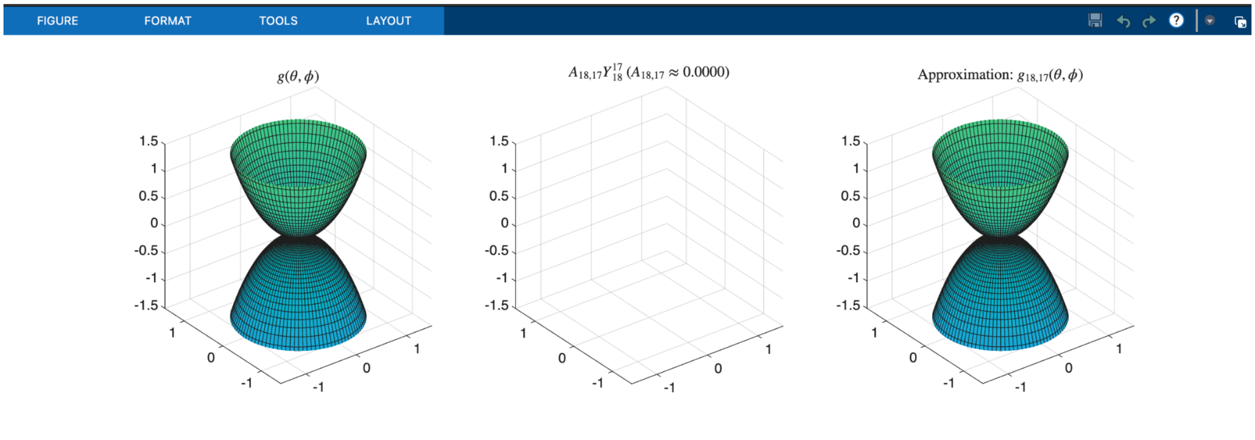
<u>Running the program:</u>

with g = cos(THETA) ./ (sin(THETA).^2 + 0.1);



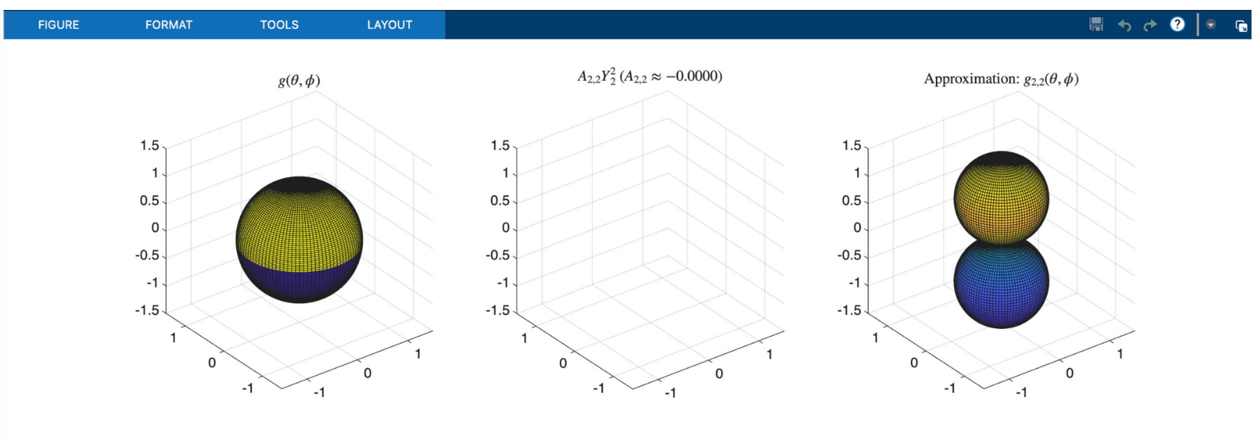


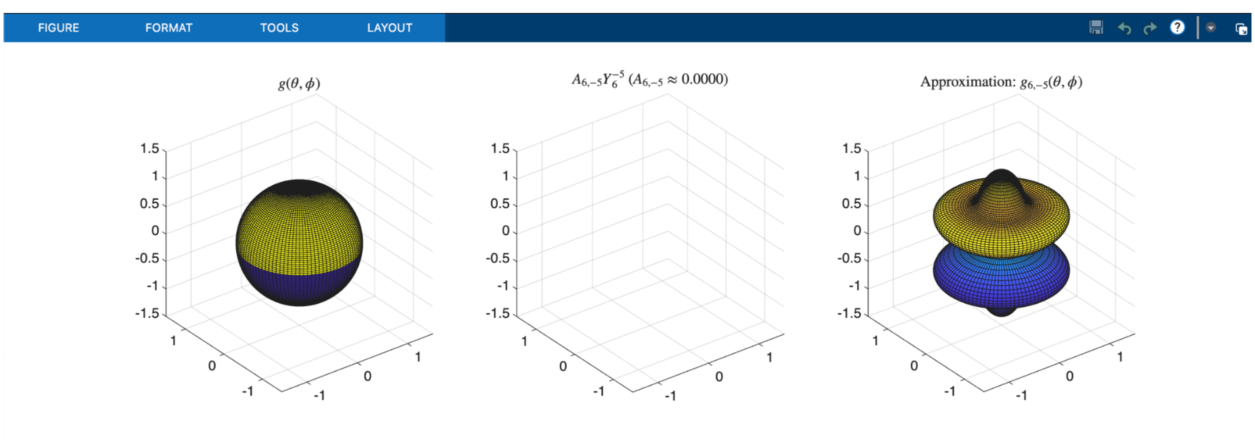


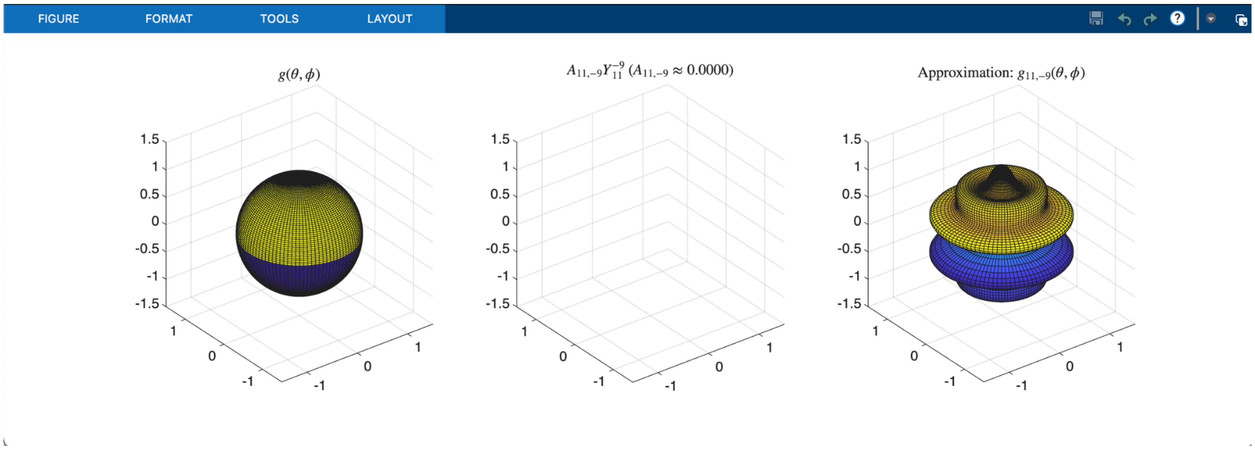


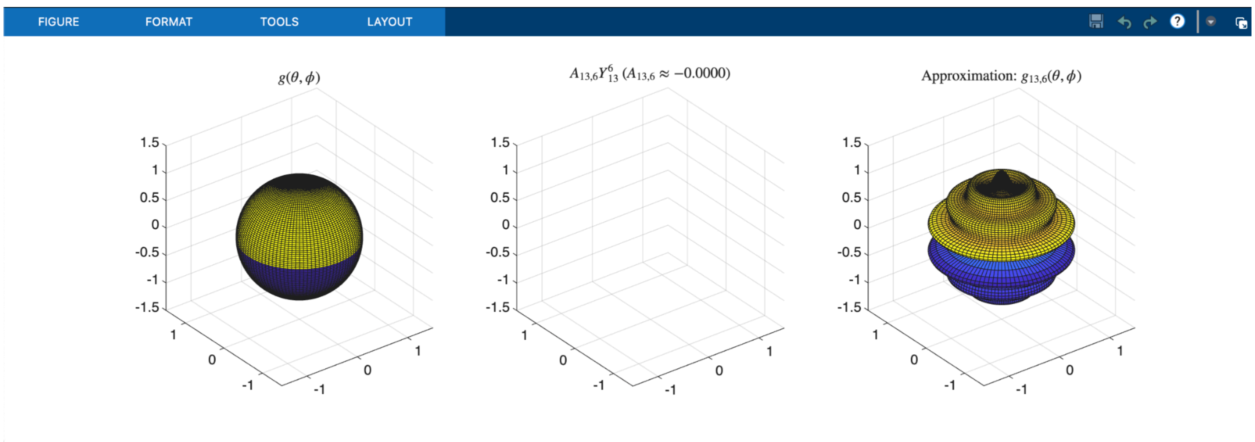
If you look (not in these images, but when running the program because a lot of the coefficients are 0), as l and m get larger, the coefficients shrink. (Bessel's inequality)

with g = double(THETA < pi/2) - double(THETA > pi/2);
I think this is the Gibbs phenomenon with spherical harmonics..

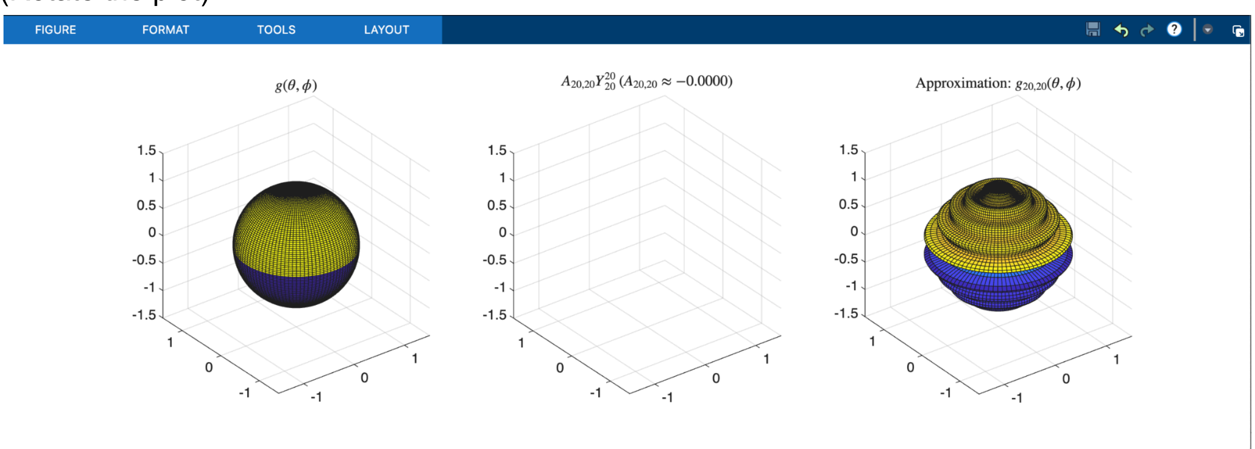








(Rotate the plot)



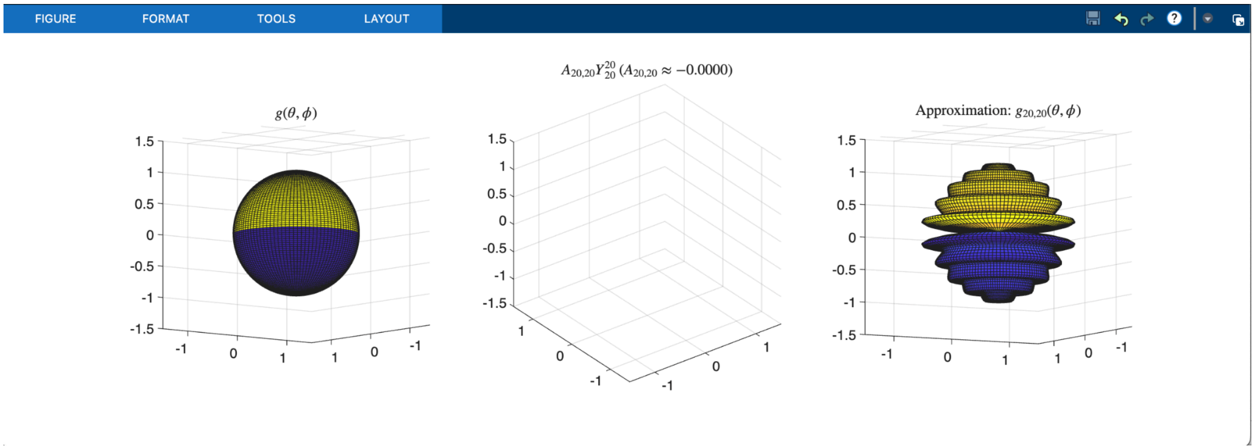


FIGURE
FORMAT
TOOLS
LAYOUT
$g(\theta, \phi)$
$A_{20,20}Y_{20}^{20}$ ($A_{20,20} \approx -0.0000$)
Approximation: $g_{20,20}(\theta, \phi)$

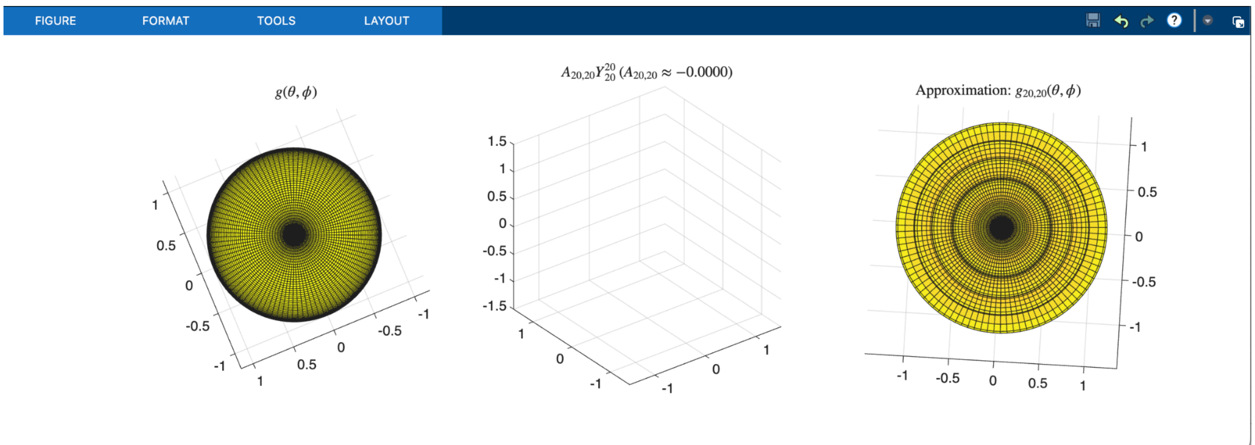


FIGURE
FORMAT
TOOLS
LAYOUT
$g(\theta, \phi)$
$A_{20,20}Y_{20}^{20}$ ($A_{20,20} \approx -0.0000$)
Approximation: $g_{20,20}(\theta, \phi)$