Tarea_3_Geodinamica

September 18, 2016

```
In [2]: import numpy as np
        import matplotlib.pyplot as plt
        %matplotlib inline
```

Problem 5-5 from Turcotte and Schubert: Determine the values of the acceleration of gravity at the equator and the poles using GRS 80 and the quadratic approximation given in Equation (5–72).

Equation 5-72:

$g_0 = \frac{GM}{a^2}\left(1+\frac{3}{2}J_2\cos^2\phi\right) + a\omega^2(\sin^2\phi - \cos^2\phi)$.

GRS80 equation from T&S:

$g0 = 9.7803267715(1 + 0.0052790414\sin^2\phi + 0.0000232718\sin^4\phi + 0.0000001262\sin^6\phi + 0.0000000007\sin^8\phi)$

From Wikipedia's GRS80 page:

Defining geometrical constants :

- Semi-major axis = Equatorial Radius =

$ a=6378137m$

Defining physical constants:

- Geocentric gravitational constant, including mass of the atmosphere

$GM = 3986005 \times 10^8 m^3/s^2$

- Dynamical form factor

$J_2 = 108263 \times 10^{-8}$

- Angular velocity of rotation

$\omega = 7292115 \times 10^{-11} rad/s$

```
In [3]: def GRS_g0(phi):
            #returns gravity acceleration according to GRS80 standard
            # in m/s^2
            return 9.7803267715*(1+0.0052790414*(np.sin(phi))**2+\
                                0.0000232718*(np.sin(phi))**4+\
                                0.0000001262*(np.sin(phi))**6+\
                                0.0000000007*(np.sin(phi))**8)


        a=6378137. #m
        GM=3986005e8 #m^3/s^2
        J2=108263e-8 #dimensionless
        omega=7292115e-11 #rad/s
        def g0(phi):
            #returns gravity acceleration according to GRS80 constants and T&S' equation 5-72
```

```
    # in m/s^2
    global a
    global GM
    global J2
    global omega
    return (GM/(a**2))*(1+1.5*J2*(np.cos(phi))**2)\
                +(a*omega**2)*((np.sin(phi))**2-(np.cos(phi))**2)
```

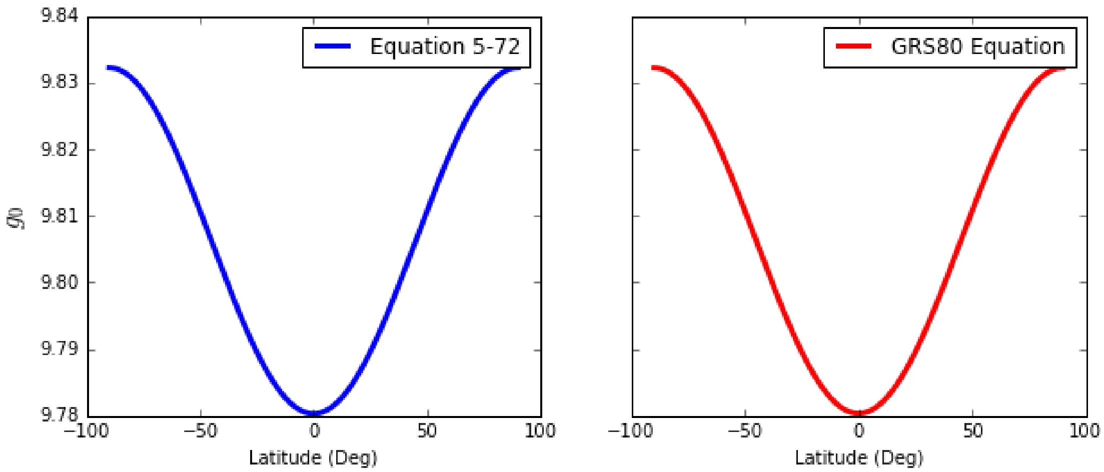
In [17]:
```
#usingEquation=[northPole, southPole, equator]
usingGRS=np.array([GRS_g0(np.pi/2), GRS_g0(-np.pi/2), GRS_g0(0.)])
usingEq572=np.array([g0(np.pi/2), g0(-np.pi/2), g0(0.)])

print 'The value for gravity at the north and south poles \n \
from GRS80 equation are respectively %(north).6f \n and %(south)\
.6f, while for the equator it is %(equator).6f \n '\
%{'north':usingGRS[0],'south':usingGRS[1], 'equator':usingGRS[2]}

print 'The value for gravity at the north and south poles \n \
from equation 5-72 are respectively %(north).6f \n and %(south)\
.6f, while for the equator it is %(equator).6f \n '\
%{'north':usingEq572[0],'south':usingEq572[1], 'equator':usingEq572[2]}

errPerc=abs(100.*(usingEq572-usingGRS)/usingEq572)
print 'The percentage difference between methods is \n of the order of 4e-4%.',errPerc
```

```
The value for gravity at the north and south poles
 from GRS80 equation are respectively 9.832186
 and 9.832186, while for the equator it is 9.780327

The value for gravity at the north and south poles
 from equation 5-72 are respectively 9.832203
 and 9.832203, while for the equator it is 9.780283

The percentage difference between methods is
 of the order of 4e-4%. [ 0.00016525  0.00016525  0.0004467 ]
```

It is possible to see that there are slight differences between methods, but these are truly negligible as the percentual differences between values are of the order of $10^{-4}\%$.

In [5]:
```
latitude=np.linspace(-np.pi/2, np.pi/2, 1000)
gravities, ax=plt.subplots(1, 2, sharey=True, figsize=(10,4))

ax[0].plot(np.rad2deg(latitude), g0(latitude), label='Equation 5-72', lw=3)
ax[1].plot(np.rad2deg(latitude), GRS_g0(latitude), label='GRS80 Equation', c='r', lw=3)
for i in range(len(ax)):
    ax[i].set_xlabel('Latitude (Deg)')
    ax[i].legend(loc=0)
ax[0].set_ylabel('$g_0$', fontsize=15)
```

Out[5]: `<matplotlib.text.Text at 0x6fd23c8>`


Equation 5-72
GRS80 Equation
g_0
9.84
9.83
9.82
9.81
9.80
9.79
9.78
−100
−50
0
50
100
Latitude (Deg)
Latitude (Deg)

```
In [ ]:
```