Геодезические системы пространственных координат

Обсудить в форуме Комментариев — 18

Эта страница опубликована в основном списке статей сайта
по адресу http://gis-lab.info/qa/geodesic-coords.html

Рассматриваются преобразования между пространственными координатными системами. Приводится пример программной реализации на языке Питон.

Содержание

Земной эллипсоид

Земным эллипсоидом называется эллипсоид вращения, поверхность которого по форме и размерам довольно близка к поверхности геоида.

Поверхность эллипсоида образуется вращением эллипса вокруг его малой оси, которая также является осью вращения эллипсоида.

Эллипс обычно определяется размером его большой полуоси *a* и сжатием *f*. Реже вместо сжатия задаётся размер малой полуоси *b*:

$$f = \frac{a-b}{a} \qquad b = a(1-f)$$

В теории и практике вычислений широко используются такие параметры, как полярный радиус кривизны поверхности *c*, первый эксцентриситет *e* и второй эксцентриситет *e'*:

$$c = \frac{a}{1-f} \qquad e^2 = f(2-f) \qquad e'^2 = \frac{e^2}{1-e^2}$$

Пример функции Питона, вычисляющей по *a* и *f* параметры *b*, *c*, *e* и *e'*:

```
def initSpher(a, f):
    b = a * (1. - f)
    c = a / (1. - f)
```

```
    e2 = f * (2. - f)
    e12 = e2 / (1. - e2)
    return (b, c, e2, e12)
```

Системы координат

Рассмотрим следующие системы координат.

1. Геоцентрические декартовы прямоугольные координаты:
    - начало координат находится в центре эллипсоида,
    - ось *z* расположена вдоль оси вращения эллипсоида и направлена в северный полюс,
    - ось *x* лежит в пересечении экватора и начального меридиана,
    - ось *y* лежит в пересечении экватора и меридиана с долготой $L = 90°$.
2. Система геодезических координат: геодезическая широта B — угол между нормалью к поверхности эллипсоида и плоскостью экватора, геодезическая долгота L — угол между плоскостями данного и начального меридианов, геодезическая высота H — кратчайшее расстояние до поверхности эллипсоида.
3. Топоцентрические декартовы прямоугольные координаты:
    - начало координат находится в некоторой точке Q_0 (B_0, L_0, H_0) над эллипсоидом,
    - ось *z* расположена вдоль нормали к поверхности эллипсоида и направлена вверх,
    - ось *x* расположена в плоскости меридиана и направлена на север,
    - ось *y* перпендикулярна к осям *x* и *z* и направлена на восток.

Помимо широкого использования в геодезических целях, каждая из представленных координатных систем находит важное применение в прикладных областях.

Геодезические координаты со времён седой древности используются в навигации и картографии. В картографии они являются основой построения проекций.

Геоцентрическая система координат необходима для вычисления спутниковых орбит и решения других орбитальных задач.

Проекции, используемые картографами различных стран, основаны на различных геодезических датумах, т.е. созданы на различных эллипсоидах с разными размерами, положением центров и ориентацией осей в пространстве. Самый простой и точный способ пересчёта координат, заданных в разных датумах, зиждется на преобразованиях между геодезическими и геоцентрическими системами. В общем случае схема пересчёта координат между двумя проекциями выполняется в пять этапов:

1. координаты первой проекции — в геодезические координаты на первом эллипсоиде,
2. геодезические координаты — в геоцентрические координаты первого датума,
3. геоцентрические координаты первого датума — в геоцентрические координаты второго датума,
4. геоцентрические координаты — в геодезические координаты на втором эллипсоиде,
5. геодезические координаты — в координаты второй проекции.

Топоцентрическая система координат — естественная система для работы различных наземных объектов: ракетных стартовых комплексов, станций слежения за спутниками, станций ПВО и других измерительных комплексов. Естественно, собираемая информация в каждом случае преобразуется в общую систему координат, связанную с Землёй — геодезическую систему координат.

Преобразования координат

Переход от геодезических координат к геоцентрическим

Это преобразование выполняется по следующим формулам:

$$x = (N + H)\cos B \cos L$$
$$y = (N + H)\cos B \sin L$$
$$z = (N + H - e^2 N)\sin B$$

Здесь *N* — так называемый радиус кривизны первого вертикала:

$$N = \frac{a}{\sqrt{1 - e^2 \sin^2 B}} = \frac{c}{\sqrt{1 + e'^2 \cos^2 B}}$$

Реализация на Питоне:

```
def fromLatLong(lat, lon, h, a, f):
    b, c, e2, e12 = initSpher(a, f)
    cos_lat = math.cos(lat)
    n = c / math.sqrt(1. + e12 * cos_lat ** 2)
    p = (n + h) * cos_lat
    x = p * math.cos(lon)
    y = p * math.sin(lon)
    z = (n + h - e2 * n) * math.sin(lat)
    return (x, y, z)
```

Переход от геоцентрических координат к геодезическим

Проще всего вычисляется долгота:

$$\operatorname{tg} L = \frac{y}{x}$$

Сложнее с определением широты и высоты. Существует множество способов решения этой задачи. Воспользуемся итеративным методом Боуринга.

В начале находится предварительная оценка широты *B*:

$$\operatorname{tg} B = \frac{z}{r}\left(1 + e'^2 \frac{b}{\rho}\right)$$

Здесь *ρ* — геоцентрический радиус-вектор, *r* — расстояние от оси вращения эллипсоида:

$$\rho = \sqrt{x^2 + y^2 + z^2} \qquad r = \sqrt{x^2 + y^2}$$

Затем вычисляется параметр *ϑ* (приведённая широта) и получается уточнённое значение широты:

$$\operatorname{tg} \theta = (1 - f)\operatorname{tg} B$$

$$\operatorname{tg} B = \frac{z + e'^2 b \sin^3 \theta}{p - e^2 a \cos^3 \theta}$$

Действия по последним двум формулам предполагается повторять до сходимости к требуемой точности. Как правило, бывает достаточно одной итерации. В примере реализации метода Боуринга, приведённом ниже, запрограммировано две итерации.

В конце определяется высота:

$$H = \frac{r}{\cos B} - N = \frac{z}{\sin B} - N(1 - e^2)$$

```
def toLatLong(x, y, z, a, f):
    b, c, e2, e12 = initSpher(a, f)
    p = math.hypot(x, y)
    if p == 0.:
        lat = math.copysign(math.pi / 2., z)
        lon = 0.
        h = math.fabs(z) - b
    else:
        t = z / p * (1. + e12 * b / math.hypot(p, z))
```

```
        for i in range(2):
            t = t * (1. - f)
            lat = math.atan(t)
            cos_lat = math.cos(lat)
            sin_lat = math.sin(lat)
            t = (z + e12 * b * sin_lat ** 3) / (p - e2 * a * cos_lat ** 3)
        lon = math.atan2(y, x)
        lat = math.atan(t)
        cos_lat = math.cos(lat)
        n = c / math.sqrt(1. + e12 * cos_lat ** 2)
        if math.fabs(t) <= 1.:
            h = p / cos_lat - n
        else:
            h = z / math.sin(lat) - n * (1. - e2)
    return (lat, lon, h)
```

Переход от геоцентрических координат к топоцентрическим

Постановка задачи: начало топоцентрической системы координат задано точкой Q_0 (B_0, L_0, H_0); по геоцентрическим координатам точки Q (x, y, z) вычислить её топоцентрические координаты.

Конформное преобразование между двумя декартовыми прямоугольными системами координат всегда может быть представлено последовательностью сдвигов и вращений координатной системы. Данное преобразование можно реализовать по следующему алгоритму:

- сместить начало координат вдоль оси z на величину $e^2 N_0 \sin B_0$ до вершины конуса, образованного нормалями, лежащими на параллели с широтой B_0,
- повернуть систему координат вокруг оси z на угол L_0, чтобы ось x оказалась в плоскости меридиана точки Q_0,
- повернуть систему координат вокруг оси y на угол $90° - B_0$, чтобы ось z совпала с нормалью к поверхности эллипсоида в точке Q_0,
- сместить начало координат вдоль оси z на величину $N_0 + H_0$ в точку Q_0,
- изменить знак x на противоположный.

Реализация алгоритма:

```
def toTopo(lat0, lon0, h0, x, y, z, a, f):
    b, c, e2, e12 = initSpher(a, f)
    sin_lat = math.sin(lat0)
    n = a / math.sqrt(1. - e2 * sin_lat ** 2)
    z = z + e2 * n * sin_lat
    x, y = rotate(x, y, lon0)
    z, x = rotate(z, x, math.pi / 2. - lat0)
    z = z - (n + h0)
    x = -x
    return (x, y, z)
```

Функция **toTopo()** содержит обращения к функции вращения **rotate()**:

```
def rotate(x, y, a):
    c, s = math.cos(a), math.sin(a)
    return (x * c + y * s, -x * s + y * c)
```

Переход от топоцентрических координат к геоцентрическим

Постановка задачи: начало топоцентрической системы координат задано точкой Q_0 (B_0, L_0, H_0); по топоцентрическим координатам точки Q (x, y, z) вычислить её геоцентрические координаты.

Алгоритм решения получается обращением алгоритма обратной задачи:

- изменить знак x на противоположный,
- сместить начало координат вдоль оси z на величину $N_0 + H_0$ в точку пересечения с осью вращения эллипсоида,

- повернуть систему координат вокруг оси *y* на угол $B_0 - 90°$, чтобы ось *z* совпала с осью вращения эллипсоида,
- повернуть систему координат вокруг оси *z* на угол $-L_0$, чтобы ось *x* оказалась в плоскости начального меридиана,
- сместить начало координат вдоль оси *z* на величину $e^2 N_0 \sin B_0$ в центр эллипсоида.

Реализация алгоритма:

```
def fromTopo(lat0, lon0, h0, x, y, z, a, f):
    b, c, e2, e12 = initSpher(a, f)
    sin_lat = math.sin(lat0)
    n = a / math.sqrt(1. - e2 * sin_lat ** 2)
    x = -x
    z = z + (n + h0)
    z, x = rotate(z, x, lat0 - math.pi / 2.)
    x, y = rotate(x, y, -lon0)
    z = z - e2 * n * sin_lat
    return (x, y, z)
```

Переход от геодезических координат к топоцентрическим. Обратная пространственная задача

Постановка задачи: начало топоцентрической системы координат задано точкой Q_0 (B_0, L_0, H_0); по геодезическим координатам точки *Q* (*B*, *L*, *H*) вычислить её топоцентрические координаты *x*, *y*, *z*.

Задача решается последовательным применением готовых алгоритмов:

- по геодезическим координатам точки *B*, *L*, *H* вычислить её геоцентрические координаты *x*, *y*, *z*,
- по геоцентрическим координатам точки вычислить её топоцентрические координаты *x*, *y*, *z*.

Реализация алгоритма:

```
def inverse3d(lat0, lon0, h0, lat, lon, h, a, f):
    x, y, z = fromLatLong(lat, lon, h, a, f)
    return toTopo(lat0, lon0, h0, x, y, z, a, f)
```

Рассмотренная задача является разновидностью обратной геодезической задачи в пространстве. Вместо декартовых прямоугольных топоцентрических координат может требоваться вычисление каких-то других связанных с ними величин, например, полярных координат «дальность-азимут-зенитное расстояние», варианты могут быть разные. Однако в большинстве случаев сначала находятся топоцентрические *x*, *y*, *z*, по которым и выводятся искомые значения.

Переход от топоцентрических координат к геодезическим. Прямая пространственная задача

Постановка задачи: начало топоцентрической системы координат задано точкой Q_0 (B_0, L_0, H_0); по топоцентрическим координатам точки *Q* (*x*, *y*, *z*) вычислить её геодезические координаты *B*, *L*, *H*.

Задача решается через вычисление геоцентрических координат:

- по тороцентрическим координатам точки *x*, *y*, *z* вычислить её геоцентрические координаты,
- по геоцентрическим координатам точки *x*, *y*, *z* вычислить её геодезические координаты *B*, *L*, *H*.

Реализация алгоритма:

```
def forward3d(lat0, lon0, h0, x, y, z, a, f):
    x, y, z = fromTopo(lat0, lon0, h0, x, y, z, a, f)
    return toLatLong(x, y, z, a, f)
```

Эта задача является разновидностью прямой геодезической задачи в пространстве. Вместо декартовых прямоугольных топоцентрических координат могут задаваться какие-то другие связанные с ними величины,

например, полярные координаты «дальность-азимут-зенитное расстояние», варианты могут быть разные. Однако в большинстве случаев сначала находятся топоцентрические *x*, *y*, *z*, по которым и решается задача.

Пример программной реализации

Коды вышеприведённых функций находятся в архиве Spheroid.zip в файле **spheroid.py**. Напишем программы, которые используют их для преобразования координат.

Пересчёт топоцентрических координат в геодезические

В этом примере программы явно задаются параметры эллипсоида *a*, *f* и геодезические координаты начала топоцентрической системы B_0, L_0, H_0. Координаты точек *x*, *y*, *z* читаются из файла данных и пересчитанные значения *B*, *L*, *H* выводятся в консоль.

```
from sys import argv
import math
import spheroid

script, fn = argv

a, f = 6378137., 1./298.257223563 # WGS 84

lat0, lon0, hgt0 = math.radians(65.), math.radians(45.), 500.

fp = open(fn, 'r')
for line in fp:
    x, y, z = map(float, line.split(" "))
    lat, lon, hgt = spheroid.forward3d(lat0, lon0, hgt0, x, y, z, a, f)
    print "%.8f %.8f %.3f" % (math.degrees(lat), math.degrees(lon), hgt)
fp.close()
```

Этот скрипт находится в архиве Spheroid.zip в файле **forwrd3d.py**.

Файл данных должен содержать в каждой строке координаты одной точки *x*, *y*, *z*, разделённые пробелом. Создадим файл данных **fwd3d.dat**:

```
-40000 30000 0
```

Выполним скрипт в командной строке:

```
$ python forwrd3d.py fwd3d.dat
```

Координаты на выходе:

```
64.63992461 45.62743323 695.578
```

Запишем полученные координаты в файл результатов **inv3d.dat**:

```
$ python forwrd3d.py fwd3d.dat > inv3d.dat
```

Пересчёт геодезических координат в топоцентрические

В этом примере программы явно задаются параметры эллипсоида *a*, *f* и геодезические координаты начала топоцентрической системы B_0, L_0, H_0. Координаты точек *B*, *L*, *H* читаются из файла данных и пересчитанные значения *x*, *y*, *z* выводятся в консоль.

```
from sys import argv
import math
import spheroid

script, fn = argv

a, f = 6378137., 1./298.257223563 # WGS 84
```

```
lat0, lon0, hgt0 = math.radians(65.), math.radians(45.), 500.

fp = open(fn, 'r')
for line in fp:
    lat, lon, hgt = map(float, line.split(" "))
    lat = math.radians(lat)
    lon = math.radians(lon)
    x, y, z = spheroid.inverse3d(lat0, lon0, hgt0, lat, lon, hgt, a, f)
    print "%.3f %.3f %.3f" % (x, y, z)
fp.close()
```

Этот скрипт находится в архиве Spheroid.zip в файле **invers3d.py**.

Файл данных должен содержать в каждой строке координаты одной точки *B*, *L*, *H*, разделённые пробелом. Используем в качестве файла данных созданный выше **inv3d.dat**:

```
64.63992461 45.62743323 695.578
```

Выполним скрипт в командной строке:

```
$ python invers3d.py inv3d.dat
```

Координаты на выходе:

```
-40000.000 30000.000 0.000
```

Ссылки

- Robert Burtch, A Comparison of Methods Used in Rectangular to Geodetic Coordinate Transformations, 2006

Обсудить в форуме Комментариев — 18

Последнее обновление: 2015-02-25 17:49

Дата создания: 23.03.2014
Автор(ы): ErnieBoyd