Задачи на сфере: угловая засечка

Обсудить в форуме Комментариев — 21

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

Угловая засечка — это нахождение положения точки по координатам двух исходных пунктов и значениям азимутов направлений с этих пунктов на определяемую точку.

Содержание

Общие положения

В качестве модели Земли принимается сфера с радиусом *R*, равным среднему радиусу земного эллипсоида. Аналогом прямой линии на плоскости является геодезическая линия на поверхности. На сфере геодезическая линия — дуга большого круга.

Введём следующие обозначения:

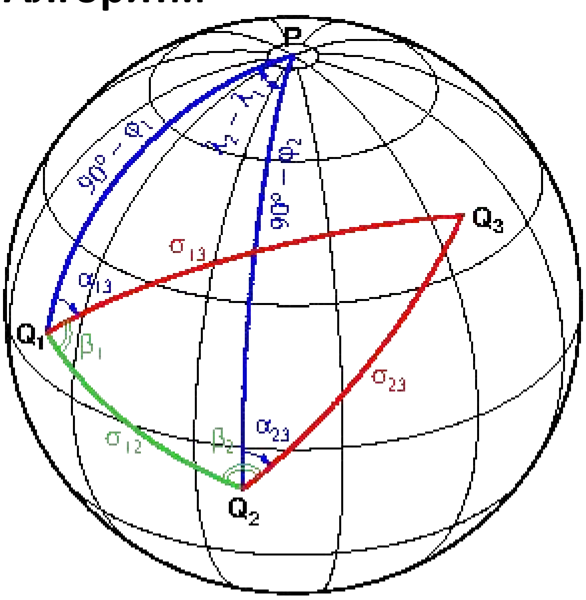
- φ — географическая широта,
- λ — географическая долгота,
- α — азимут дуги большого круга,
- σ — сферическое расстояние (длина дуги большого круга, выраженная в долях радиуса шара).

Линейное расстояние по дуге большого круга *s* связано со сферическим расстоянием σ формулой $s = R\,\sigma$.

Постановка задачи

Исходные данные координаты пунктов Q_1, Q_2 — φ_1, λ_1, φ_2, λ_2, начальные направления с пунктов Q_1, Q_2 на точку Q_3 — α_{13}, α_{23}. Определяемые величины координаты точки Q_3 — φ_3, λ_3.

Алгоритм



Угловая засечка

Решение любого вида засечек сводится к нахождению полярных координат искомой точки, т.е. начального направления и расстояния на неё с одного из исходных пунктов. На конечном этапе координаты находятся из решения прямой геодезической задачи. Поскольку в угловой засечке направления α_{13} и α_{23} уже заданы, остаётся определить расстояние σ_{13} или σ_{23}.

На рисунке синим цветом выделены заданные элементы сферических треугольников, красным цветом неизвестные, зелёным — вспомогательные элементы. Очевидно, в треугольнике $Q_1Q_2Q_3$ нет ни одного известного элемента. Однако из решения обратной геодезической задачи для пунктов Q_1, Q_2 могут быть получены расстояние σ_{12}, а также азимуты α_{12} и α_{21}, после чего углы β_1 и β_2 вычисляются как разности азимутов при соответствующих пунктах. Далее из решения треугольника $Q_1Q_2Q_3$ найдём сторону σ_{13}.

Последовательность действий:

1. решить обратную геодезическую задачу для Q_1, Q_2: по φ_1, λ_1, φ_2, λ_2 получить α_{12}, α_{21}, σ_{12};
2. вычислить углы β_1, β_2;
3. в треугольнике $Q_1Q_2Q_3$ по σ_{12}, β_1, β_2 вычислить σ_{13};
4. решить прямую геодезическую задачу для Q_1, Q_3: по φ_1, λ_1, α_{13}, σ_{13} вычислить φ_3, λ_3.

Действия по первому и последнему пунктам рассмотрены в статьях Задачи на сфере: обратная геодезическая задача и Задачи на сфере: прямая геодезическая задача.

Углы β_1, β_2 и длина σ_{13} вычисляются по формулам:

$$\beta_1 = \alpha_{12} - \alpha_{13}$$
$$\beta_2 = \alpha_{23} - \alpha_{21}$$

$$\sigma_{13} = \operatorname{arctg} \frac{\sin\beta_2 \sin\sigma_{12}}{\cos\beta_2 \sin\beta_1 + \sin\beta_2 \cos\beta_1 \cos\sigma_{12}}$$

Правда, до вычисления длины σ_{13} необходимо проанализировать полученные значения углов β_1 и β_2. Ниже в коде функции можно увидеть пример такого анализа:

- если линии Q_1Q_3 и Q_2Q_3 совпадают с Q_1Q_2, решение не определено, т.к. решением может быть любая точка геодезической линии Q_1Q_2;
- если одна из линий Q_1Q_3 и Q_2Q_3 совпадает с Q_1Q_2, а другая нет, решением является пункт, из которого выходит другая;
- если линии Q_1Q_3 и Q_2Q_3 уходят в разные полушария от Q_1Q_2, функция находит ближайшее к Q_1Q_2 «ложное пересечение» этих линий.

Здесь необходимо пояснить, что на сфере две несовпадающие геодезические линии всегда пересекаются в двух точках-антиподах. В традиционной постановке задачи направление на нужное пересечение задаётся явно. Если же прямое и обратное направления по условию равнозначны, возникает вопрос выбора одного из антиподов: φ_3, λ_3 или $\varphi_3{}' = -\varphi_3$, $\lambda_3{}' = \lambda_3 \pm 180°$.

Пример программной реализации

Пример функции SphereAngular на языке Си, реализующей вышеизоложенный алгоритм:

```
/*
 * Решение угловой засечки
 *
 * Аргументы исходные:
 *     pt1   - {широта, долгота} пункта Q1
 *     pt2   - {широта, долгота} пункта Q2
 *     azi13 - азимут направления Q1-Q3
 *     azi23 - азимут направления Q2-Q3
 *
```

```
 * Аргументы определяемые:
 *     pt3 - {широта, долгота} точки Q3
 */
int SphereAngular(double pt1[], double pt2[], double azi13, double azi23,
                  double pt3[])
{
  double azi12, dist12, azi21, dist13;
  double cos_beta1, sin_beta1, cos_beta2, sin_beta2, cos_dist12, sin_dist12;

  SphereInverse(pt2, pt1, &azi21, &dist12);
  SphereInverse(pt1, pt2, &azi12, &dist12);
  cos_beta1 = cos(azi13 - azi12);
  sin_beta1 = sin(azi13 - azi12);
  cos_beta2 = cos(azi21 - azi23);
  sin_beta2 = sin(azi21 - azi23);
  cos_dist12 = cos(dist12);
  sin_dist12 = sin(dist12);

  if (sin_beta1 == 0. && sin_beta2 == 0.)      // Решение - любая точка
    return -1;                                //    на большом круге Q1-Q2.
  else if (sin_beta1 == 0.) {
    pt3[0] = pt2[0];                          // Решение - точка Q2.
    pt3[1] = pt2[1];
    return 0;
  } else if (sin_beta2 == 0.) {                       // Решение - точка Q1.
    pt3[0] = pt1[0];
    pt3[1] = pt1[1];
    return 0;
  } else if (sin_beta1 * sin_beta2 < 0.) {     // Лучи Q1-Q3 и Q2-Q3 направлены
    if (fabs(sin_beta1) >= fabs(sin_beta2)) { //    в разные полусферы.
      cos_beta2 = -cos_beta2;                 // Выберем ближайшее решение:
      sin_beta2 = -sin_beta2;                 //    развернём луч Q2-Q3 на 180°;
    } else {                                  //      иначе
      cos_beta1 = -cos_beta1;                 //    развернём луч Q1-Q3 на 180°.
      sin_beta1 = -sin_beta1;
    }
  }
  dist13 = atan2(fabs(sin_beta2) * sin_dist12,
                 cos_beta2 * fabs(sin_beta1)
                 + fabs(sin_beta2) * cos_beta1 * cos_dist12);
  SphereDirect(pt1, azi13, dist13, pt3);

  return 0;
}
```

Этот код находится в архиве Sph.zip в файле **sph.c**. Кроме того, в файл **sph.h** включены следующие определения:

```
#define A_E 6371.0                              // радиус Земли в километрах
#define Degrees(x)  (x * 57.29577951308232)     // радианы -> градусы
#define Radians(x)  (x / 57.29577951308232)     // градусы -> радианы
```

Теперь напишем программу, которая обращается к функции SphereAngular для решения угловой засечки:

```
#include <stdio.h>
#include <stdlib.h>
#include "sph.h"

int main(int argc, char *argv[])
{
  char buf[1024];
  double pt1[2], pt2[2], pt3[2];
  double lat1, lon1, lat2, lon2, azi13, azi23;

  while (fgets(buf, 1024, stdin) != NULL) {
```

```
    sscanf(buf, "%lf %lf %lf %lf %lf %lf",
           &lat1, &lon1, &lat2, &lon2, &azi13, &azi23);
    pt1[0] = Radians(lat1);
    pt1[1] = Radians(lon1);
    pt2[0] = Radians(lat2);
    pt2[1] = Radians(lon2);
    if (SphereAngular(pt1, pt2, Radians(azi13), Radians(azi23), pt3))
      puts("\t"); /* Бесконечно много решений на большом круге Q1-Q2 */
    else
      printf("%f\t%f\n", Degrees(pt3[0]), Degrees(pt3[1]));
  }
  return 0;
}
```

В архиве Sph.zip этот код находится в файле **ang.c**. Создадим исполняемый модуль **ang** компилятором **gcc**:

```
$ gcc -o ang ang.c sph.c -lm
```

Впрочем, в архиве есть **Makefile**. Для MS Windows готовую программу **ang.exe** можно найти в архиве Sph-win32.zip.

Программа читает данные из стандартного ввода консоли и отправляет результаты на стандартный вывод. Для чтения и записи файлов используются символы перенаправления потока «>» и «<» соответственно. Из каждой строки ввода программа считывает координаты первого и второго пунктов φ_1, λ_1, φ_2, λ_2, начальные азимуты α_{13} и α_{23} в градусах; решает угловую засечку; выводит в строку вывода координаты третьей точки φ_3, λ_3 в градусах.

Создадим файл **ang.dat**, содержащий одну строку данных:

```
30 0 60 30 44.80406 110.389945
```

После запуска программы

```
$ ang < ang.dat
```

получим φ_3, λ_3:

```
52.000000 54.000000
```

В архиве Sph-py.zip находится код на языке Питон. Выполнение скрипта в командной консоли:

```
$ python ang.py ang.dat
```

Ссылки

- Вычисление расстояния и начального азимута между двумя точками на сфере
- Нахождение точки пересечения двух линий по углам и двум известным точкам (биангуляция)
- Задачи на сфере: обратная геодезическая задача
- Задачи на сфере: прямая геодезическая задача
- Задачи на сфере: линейная засечка
- Краткий справочник по сферической тригонометрии
- Earth radius
- Степанов Н. Н. Сферическая тригонометрия

Обсудить в форуме Комментариев — 21

Последнее обновление: 2014-06-21 11:41

Дата создания: 11.03.2014

Автор(ы): [ErnieBoyd]