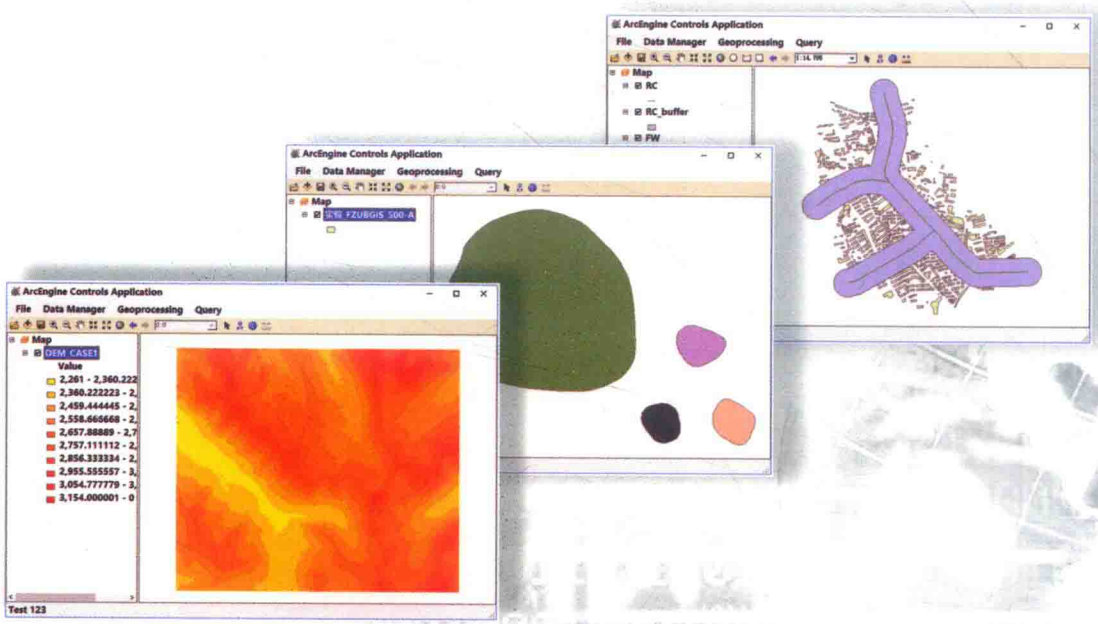
地理信息系统应用与开发丛书

地理信息系统开发与编程实验教程

李进强　编著

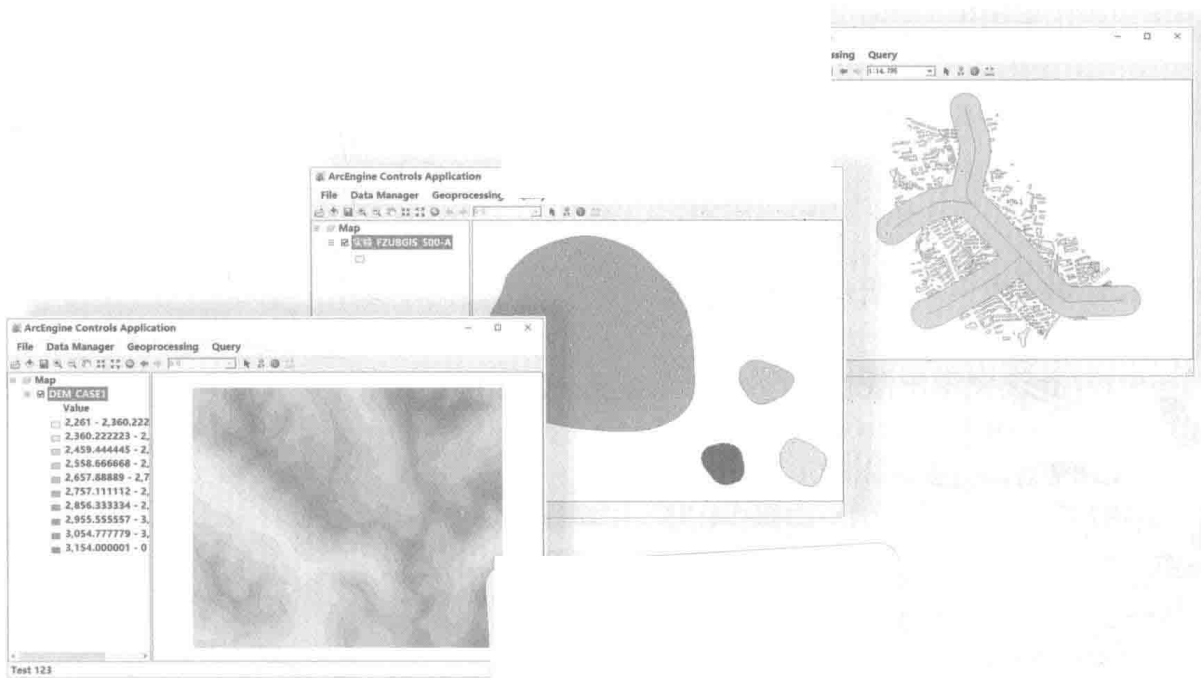


WUHAN UNIVERSITY PRESS
武汉大学出版社

地理信息系统应用与开发丛书

地理信息系统开发与编程实验教程

李进强　编著



WUHAN UNIVERSITY PRESS
武汉大学出版社

图书在版编目(CIP)数据

地理信息系统开发与编程实验教程/李进强编著. —武汉:武汉大学出版社,2018.11

地理信息系统应用与开发丛书

ISBN 978-7-307-16244-0

Ⅰ.地… Ⅱ.李… Ⅲ.地理信息系统—教材 Ⅳ.P208

中国版本图书馆 CIP 数据核字(2018)第 239193 号

责任编辑:鲍 玲 责任校对:李孟潇 装帧设计:韩闻锦

出版发行:**武汉大学出版社** (430072 武昌 珞珈山)

(电子邮件:cbs22@whu.edu.cn 网址:www.wdp.com.cn)

印刷:湖北金海印务有限公司

开本:787×1092 1/16 印张:14.75 字数:347 千字

版次:2018 年 11 月第 1 版 2018 年 11 月第 1 次印刷

ISBN 978-7-307-16244-0 定价:36.00 元

前　言

本书以 ArcGIS Engine10.5+Visual Studio 2015 组件式开发为主线，通过 21 个典型 GIS 开发实验案例，介绍了地理信息系统开发框架设计，空间信息可视化，空间查询统计与分析，空间数据管理等方面初步开发技术与方法。

各部分按照以下顺序展开：

目的与要求：阐述通过本实验达到的效果；

实验原理：对每部分所涉及的 ArcGIS Engine 接口、实现类，以及对应的属性和方法进行了详细讲解；

内容与步骤：主要介绍实验步骤和参考代码，对技术难点内容作出恰当解释。

编译与测试：配套实验数据可供读者测试所编程序的正确性。

思考与练习：提出了若干同类思考问题和扩展练习题，供进一步提升学习使用。

为简化学习难度，作者在编写时尽量简化界面元素，并尽量减少多技术关联(如读者需要了解实用化的开发技术，请参考作者编著的《基于 ArcGIS Engine 地理信息系统开发技术与实践》一书)，使得每个实验相互独立又有联系，帮助读者掌握面向对象程序设计的基本知识。

本书可作为地理科学、资源环境科学等相关专业及开展“地理信息系统设计与编程”课程教学的实验教材，也可供其他相关人员自学参考。

由于作者水平有限，再加上编写时间仓促，书中错漏之处在所难免，敬请读者批评指正。作者邮箱：1361639771@qq.com。

目　录

实验一　投影计算程序设计(C#增强)

一、目的与要求

通过投影正反算的编程，加深对面向对象程序设计的理解，巩固复习有关 C 语言的基础知识，同时提高学生运用计算机技术解决实际问题的能力。

本次实验的主要任务是：每人独立完成 Gauss、Lambert 投影正反算算法设计、计算编程、程序调试等，并通过实例验证程序的正确性。

实验过程中，同学们应当复习《地图投影学》或《控制测量学》有关内容。

二、实验原理

1. Gauss 投影计算公式

(1)基本公式

Gauss 投影正算公式(精度精确至 0.001m，角度 B，L 以弧度为单位)：

$$\left.\begin{aligned} x &= X + \frac{N}{2}\sin B\cos BL^2 + \frac{N}{24}\sin B\cos^3 B(5 - t^2 + 9\eta^2 + 4\eta^4)L^4 + \\ &\quad \frac{N}{720}\sin B\cos^5 B(61 - 58t^2 + t^4)L^6 \\ y &= N\cos BL + \frac{N}{6}\cos^3 B(1 - t^2 + \eta^2)L^3 + \\ &\quad \frac{N}{120}\cos^5 B(5 - 18t^2 + t^4 + 14\eta^2 - 58\eta^2 t^2)L^5 \end{aligned}\right\}$$

Gauss 投影反算公式：

$$\left.\begin{aligned} B &= B_f - \frac{t_f}{2M_f N_f}y^2 + \frac{t_f}{24M_f N_f^3}(5 + 3t_f^3 + \eta_f^2 - 9\eta_f^2 t_f^2)y^4 - \\ &\quad \frac{t_f}{720M_f N_f^5}(61 + 90t_f^2 + 45t_f^4)y^6 \\ L &= \frac{1}{N_f\cos B_f}y - \frac{1}{6N_f^3\cos B_f}(1 + 2t_f^2 + \eta_f^2)y^3 + \\ &\quad \frac{1}{120N_f^5\cos B_f}(5 + 28t_f^2 + 24t_f^4 + 6\eta_f^2 + 8\eta_f^2 t_f^2)y^5 \end{aligned}\right\}$$

子午线收敛角计算公式：

$$\gamma = \sin B \cdot L + \frac{1}{3}\sin B \cos^2 B \cdot L^3(1 + 3\eta^2 + 2\eta^4) + \frac{1}{15}\sin B \cos^4 B \cdot L^5(2 - t^2) + \cdots$$

(2)子午线长度计算公式

$$X = a_0 B - \frac{a_2}{2}\sin 2B + \frac{a_4}{4}\sin 4B - \frac{a_6}{6}\sin 6B + \frac{a_8}{8}\sin 8B$$

$$\left.\begin{aligned} a_0 &= m_0 + \frac{m_2}{2} + \frac{3}{8}m_4 + \frac{5}{16}m_6 + \frac{35}{128}m_8 + \cdots \\ a_2 &= \frac{m_2}{2} + \frac{m_4}{2} + \frac{15}{32}m_6 + \frac{7}{16}m_8 \\ a_4 &= \frac{m_4}{8} + \frac{3}{16}m_6 + \frac{7}{32}m_8 \\ a_6 &= \frac{m_6}{32} + \frac{m_8}{16} \\ a_8 &= \frac{m_8}{128} \end{aligned}\right\}$$

(3)子午圈卯酉圈曲率半径计算公式

$$N = \frac{a}{W}, \quad M = \frac{c}{V^3}$$

$$\left.\begin{aligned} W &= \sqrt{1 - e^2 \sin^2 B} \\ V &= \sqrt{1 + e'^2 \cos^2 B} \end{aligned}\right\}$$

(4)椭球参数计算公式

$$c = \frac{a^2}{b}, \quad t = \tan B, \quad \eta^2 = e'^2 \cos^2 B$$

$$e = \frac{\sqrt{a^2 - b^2}}{a}, \quad e' = \frac{\sqrt{a^2 - b^2}}{b}$$

$$m_0 = a(1 - e^2), \quad m_2 = \frac{3}{2}e^2 m_0, \quad m_4 = \frac{5}{4}e^2 m_2,$$

$$m_6 = \frac{7}{6}e^2 m_4, \quad m_8 = \frac{9}{8}e^2 m_6$$

2. Lambert 投影计算公式

(1)正算公式

$$\begin{cases} X = r_0 - r\cos(\theta) \\ Y = r\sin(\theta) \end{cases}$$

说明：r 为纬线投影半径(r_0为原点纬线投影半径)，θ 为计算点的子午线与中央子午线投影后的夹角。

$$r = aFt^n$$

$$r_0 = aFt_0^n$$

$$\theta = n(L - L_0)$$

其中，n，F 值由下式计算：

$$n = \frac{\ln(m_{B_1}/m_{B_2})}{\ln(t_{B_1}/t_{B_2})}$$

$$F = m_{B_1}/(nt_{B_1}^n)$$

其中，m_{B_1}，m_{B_2}分别为标准维度 B_1，B_2处的 m 值。

其中 t_{B_1}，t_{B_2}，t_0分别为标准维度 B_1，B_2和原点维度 B_0处的 t 值。

计算公式如下：

$$m = \frac{\cos(B)}{\sqrt{1 - e^2\sin^2(B)}}$$

$$t = \tan\left(\frac{\pi}{4} - \frac{B}{2}\right) \Big/ \left(\frac{1 - e\sin(B)}{1 + e\sin(B)}\right)^{\frac{e}{2}}$$

(2)反算公式

$$B = \frac{\pi}{2} - 2\arctan\left[t'\left(\frac{1 - e\sin(B)}{1 + e\sin(B)}\right)^{\frac{e}{2}}\right]$$

$$L = \frac{\theta'}{n} + L_0$$

$$t' = \left(\frac{r'}{aF}\right)^{\frac{1}{n}}$$

$$\theta' = \arctan\frac{Y}{r_0 - X}$$

$$r' = \mathrm{Sign}(n) \cdot \sqrt{Y^2 - (r_0 - X)^2}$$

说明：

公式中 n，r_0与坐标正算方法相同。

r'，θ'分别表示由直角坐标计算得出的纬线投影半径和经线与中央子午线的投影角。

Sign(n)为 n 的符号函数，结果为+1 或-1。

三、实验环境与数据

①编程环境：VS 2015 C#。

②实验数据：

Lambert 投影要求实现以下几组坐标之间的互相转换，详见表 1-1。

表 1-1　　利用 Lambert 投影实现的几组坐标之间的转换

椭球	B	L	L_0	X	Y
克拉索夫斯基	43°18′17.6562	87°57′11.0581	105°	5294592.403	−1321241.192
克拉索夫斯基	35°34′11.5623	106°23′17.1521	105°	4319291.848	123137.909

其中：原点纬度：0°；第一标准纬度：30°；第二标准纬度：62°。

Gauss 投影要求实现以下几组坐标之间的互相转换，见表 1-2。

表 1-2　　利用 Gauss 投影实现的几组坐标之间的转换

椭球	B	L	L_0	X	Y
克拉索夫斯基	51°38′43.9023″	126°02′13.1360″	123°	5728374.726	210198.193
克拉索夫斯基	29°34′16.5112″	106°25′14.8663″	105°	3273488.972	137682.377
1975 年国际椭球	51°38′43.9023″	126°02′13.1360″	123°	5728276.609	210194.803
1975 年国际椭球	29°34′16.5112″	106°25′14.8663″	105°	3273431.384	137680.138

1975 国际椭球体参数：

$a=6378140.0$；

$b=6356755.2881575287$。

克拉索夫斯基椭球体参数：

$a=6378245.0$；

$b=6356863.0187730473$。

四、实验内容与步骤

1. 接口/基类/辅助结构定义

(1) GeorefEllipsoid 基类定义

定义参考椭球体基类 GeorefEllipsoid，提供获取参考椭球体长半轴、短半轴、偏心率等基本计算功能，设计代码如下：

```
public class GeorefEllipsoid
{
    ///<summary>
    ///椭球体长半轴(单位:米)
    ///</summary>
    private double _a;
    ///<summary>
```

```
///椭球体短半轴(单位:米)
///</summary>
private double _b;
///<summary>
///构造函数
///</summary>
///<param name="A"></param>
///<param name="B"></param>
public GeorefEllipsoid(double A, double B)
{
    _a = A;
    _b = B;
}
///<summary>
///椭球体长半轴
///</summary>
public double a
{
    get { return _a; }
}
///<summary>
///椭球体短半轴
///</summary>
public double b
{
    get { return _b; }
}
///<summary>
///偏心率(a-b)/a
///</summary>
public double f
{
    get { return (_a - _b) / _a; }
}
///<summary>
///第一偏心率
///</summary>
public double e1
{
```

```
            get
            {
                return Math.Sqrt(a * a - b * b) /a;
            }
        }
        /// <summary>
        /// 第二偏心率
        /// </summary>
        public double e2
        {
            get
            {
                return Math.Sqrt(a * a - b * b) /b;
            }
        }
    }
```

(2) IProjection 接口定义

定义两个投影计算接口函数：地理坐标投影到直角坐标，直角坐标反算到地理坐标：

```
interface IProjection
{
    Vector2d BLtoVector2d( double B, double L);
    Geographics XYtoGeographics(double X, double Y);
}
```

(3) Vector2d、Geographics 辅助结构

Vector2d、Geographics 结构分别表示直角坐标和地理坐标：

```
public struct Vector2d
{
    public double X;
    public double Y;
    public Vector2d(double x, double y)
    {
        X = x;
        Y = y;
    }
}

public struct Geographics
{
    public double B;
```

```
    public double L;

    public Geographics(double lat, double lng)
    {
        B = lat;
        L = lng;
    }
}
```

(4) Function 静态类实现

定义静态类 Function，提供“拟十进制”角度和“十进制”角度的转换：

```
public static class Function
{
    public static double DegToDms(double deg)
    {
        double d, m, s;
        s = (deg > 0) ? deg : -deg;
        d = Math.Floor(s);
        s = (s - d) * 60;
        m = Math.Floor(s);
        s = (s - m) * 60;

        s = d + m /100 + s /10000;
        return (deg > 0) ? s : -s;
    }
    //dms to deg
    public static double DmsToDeg(double dms)
    {
        double d, m, s;
        s = (dms > 0) ? dms : -dms;
        d = Math.Floor(s);
        s = (s - d) * 100;
        m = Math.Floor(s);
        s = (s - m) * 100;

        s = (d + (m /60) + s /3600);
        return (dms > 0) ? s : -s;
    }
}
```

2. 创建 Gauss 投影计算窗体

使用 VS 2015 模板创建 Windows 应用程序，项目命名为“Projection”，创建窗体 GaussProjection。

①在 GaussProjection 窗体设计界面上，添加 5 个 TextBox、2 个 Button 控件，详见表 1-3。

表 1-3　　**GaussProjection 窗体控件命名及 Name 属性**

控件	Name 属性	含义
TextBox	txtLongitude	经度
TextBox	txtLatitude	纬度
TextBox	txtX	Gauss 坐标 X
TextBox	txtY	Gauss 坐标 Y
TextBox	txtCentreLng	中央子午线
Button	btnComputation	正算
Button	btnInverseComputation	反算

效果如图 1-1 所示。

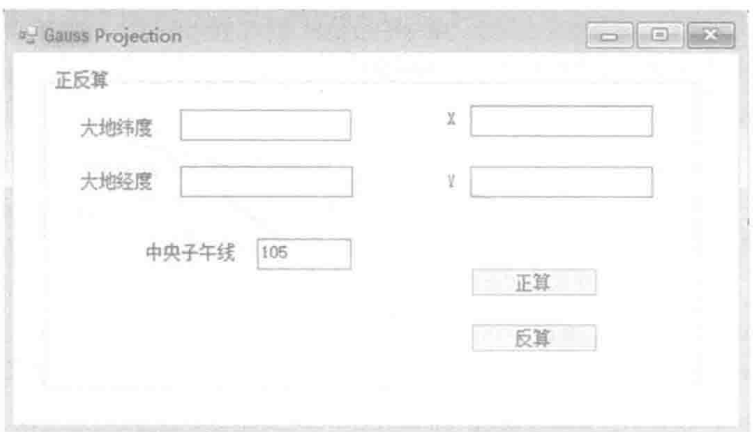


图 1-1　GaussProjection 用户界面

②添加两个常量成员(椭球的长轴，短轴)：

```
const double _a = 6378140.0;
const double _b = 6356755.2881575287。
```

③为两个 Button 控件的 Click 事件添加响应函数。

代码如下：

```
public partial classGaussFrm : Form
{
```

```
        //1975 国际椭球体参数
        const double _a = 6378140.0;
        const double _b = 6356755.2881575287;

        public GaussFrm()
        {
            InitializeComponent();
        }

        //正算响应函数
        private void btnComputation_Click(object sender, EventArgs e)
        {
            //从界面获取经度、纬度、中央子午线,同时转换为十进制角度
            double degL = Function.DmsToDeg(double.Parse(txtLongitude.
Text));
            double degB = Function.DmsToDeg(double.Parse(txtLatitude.
Text));
            int iCentreL = int.Parse(txtCentreLng.Text);

            //计算经差,并转为弧度
            double dL = (degL - iCentreL) * Math.PI /180;
            double dB = degB * Math.PI /180;

            //初始化 GaussProjection 对象,调用 BLtoVector2d 函数完成投影正算
            IProjection gp = new GaussProjection(_a, _b);
            Vector2d vet = gp.BLtoVector2d(dB, dL);

            //Y 附加常数 500 公里,取整至 0.001
            double X = Math.Round(vet.X, 3);
            double Y = Math.Round(vet.Y + 500000.0, 3);

            //更新 X、Y 显示控件
            txtX.Text = X.ToString();
            txtY.Text = Y.ToString();
        }

        //反算响应函数
         private void btnInverseComputation_Click(object sender,
```

```
EventArgs e)
        {

            //从界面获取 X、Y、中央子午线
            double dX = double.Parse(txtX.Text);
            double dY = double.Parse(txtY.Text) - 500000;
            int  iCentreL = int.Parse(txtCentreLng.Text);

            //初始化 GaussProjection 对象,调用 XYtoGeographics 函数完成投影反算
            IProjection gp = new GaussProjection(_a, _b);
            Geographics pos = gp.XYtoGeographics(dX, dY);

            //弧度制经纬度转换为拟十进制,由经差计算经度
             double degL = Function.DegToDms(pos.L * 180 / Math.PI +
iCentreL);
            double degB = Function.DegToDms(pos.B * 180 /Math.PI);

            //取整至 0.00000001
            degB = Math.Round(degB, 8);
            degL = Math.Round(degL, 8);

            //更新 L,B 显示控件
            txtLongitude.Text = degL.ToString();
          txtLatitude.Text = degB.ToString();
        }
    }
```

3. GaussProjection 功能类实现

窗体类用到的功能类是 GaussProjection，该类继承 GeorefEllipsoid，IProjection 实现，代码如下：

```
    public class GaussProjection: GeorefEllipsoid, IProjection
    {
        public GaussProjection(double a, double b)
              :base(a, b)
        {
        }

        ///正算
```

```
        public Vector2d BLtoVector2d( double dB, double dL )
        {
            //计算辅助函数
            double cosb, sinb, W, V, N, t, g, p;
            cosb = Math.Cos(dB);
            sinb = Math.Sin(dB);
            W = Math.Sqrt(1 - e1 * e1 * sinb * sinb);
            V = Math.Sqrt(1 + e2 * e2 * cosb * cosb);
            N = a /W;
            t = Math.Tan(dB);
            g = e2 * e2 * cosb * cosb;
            p = 1;

            //计算子午线弧长
            double[] A = getMeridianDisParameter();
            double dX = A[0] * dB - (A[2] /2) * Math.Sin(2 * dB) + (A[4]
/4) * Math.Sin(4 * dB) - (A[6] /6) * Math.Sin(6 * dB) + (A[8] /8) *
Math.Sin(8 * dB);

            //计算 X , Y , R
            double l1_ = dL /p;
            double l2_ = l1_ * l1_;
            double l3_ = l2_ * l1_;
            double l4_ = l2_ * l2_;
            double l5_ = l4_ * l1_;
            double l6_ = l4_ * l2_;
            double cosb3w = cosb * cosb * cosb;
            double cosb5w = cosb3w * cosb * cosb;
            double t2 = t * t;
            double t4 = t2 * t2;
            double g2 = g * g;
            double g4 = g2 * g2;

            dX = dX + (N /2) * sinb * cosb * l2_ + (N /24) * sinb *
cosb3w * (5 - t2 + 9 * g2 + 4 * g4) * l4_ + (N /720) * sinb * cosb5w *
(61 - 58 * t2 + t4) * l6_;

            double dY = N * cosb * l1_ + (N /6) * cosb3w * (1 - t2 + g2)
```

```
* l3_ + (N /120) * cosb5w * (5 - 18 * t2 + t4 + 14 * g2 - 58 * g2 * t2)
* l5_;

            return new Vector2d(dX, dY);
        }

        ///反算
        public Geographics XYtoGeographics(double dX, double dY)
        {
         //计算底点纬度
            double bf = get_Bf(dX);

            //计算辅助函数
            double tf = Math.Tan(bf);
            double cosbf = Math.Cos(bf);
            double sinbf = Math.Sin(bf);

            double Wf = Math.Sqrt(1 - e1 * e1 * sinbf * sinbf);
            double Vf = Math.Sqrt(1 + e2 * e2 * cosbf * cosbf);
            double Nf = a /Wf;
            double Mf = Nf /(Vf * Vf);
            double gf = e2 * e2 * cosbf * cosbf;

            //计算 L ,B
            double y2 = dY * dY;
            double y3 = dY * y2;
            double y4 = y2 * y2;
            double y5 = dY * y4;

            double tf2 = tf * tf;
            double tf3 = tf * tf2;
            double tf4 = tf2 * tf2;
            double Nf3 = Nf * Nf * Nf;
            double Nf5 = Nf * Nf * Nf3;
            double gf2 = gf * gf;

            double dB = bf - tf * y2 /(2 * Mf * Nf) + tf * (5 + 3 * tf2 +
gf2 - 9 * gf2 * tf2) * y4 /(24 * Mf * Nf3);
```