EXPLORATORY PROJECT

Development of a Program for slope stability analysis by using BISHOP's method

Under the guidance of

Dr. Ashok Jaiswal

Associate Professor

Dept. of Mining Engineering

IIT(BHU), Varanasi

Submitted by :-

Rajat Agrawal

19155072

Dept. of Mining Engineering

OBJECTIVE

We write a code to analyze the stability of slope by use of the Bishop’s method by using a computer to make the process feasible,fast & efficient. This method can calculate the most accurate answer of stability of slope upto two decimal points.

The Bishop's method is slightly different from the ordinary method of slices in that normal interaction forces between adjacent slices are assumed to be collinear and the resultant interslice shear force is zero. The approach was proposed by Alan W. Bishop of Imperial College. The constraint introduced by the normal forces between slices makes the problem statically indeterminate. As a result, iterative methods have to be used to solve for the factor of safety.

METHODOLOGY

With this method, the analysis is carried out in terms of stresses instead of forces which were used with the Ordinary Method of Slices. This method is very similar to the ordinary method of slices. The major difference between the Bishop Method and the Ordinary Method of Slices is that resolution of forces takes place.

We use the iterative method for the calculation of the Factor safety(FS). We assume a value of FS and Iterate through all the values of FS if the value is SAME as the assumed one then we get the value of FACTOR SAFETY.

We have assumed a failure surface and some slices on this surface to calculate the stability of the slope along this failure surface. Also the interslice forces(T(i+1)-T(i)) is ignored due to the very less effect of it on the FS in Bishop's method

THE CODE is written in C++ language.
We have used C++ as the language for the development of our code as C++ is very fast and executes our operations efficiently.This was perfect for Our requirements as main objective is to write a code to execute Fast. Also it has many in built Functions to use like min(),max(),__gcd(),etc. And many data types like Double , int and float which were required for Our program.
We have used Sublime text editor to write our code as its user interface is perfect and gives the user a lot of customizations to help him write code.
GNU GCC compiler is used to compile our code.

Procedure

The procedure contains the following few steps:

1. **INPUT -** We give the input to the program sequentially.
   Cohesion , no. of slices(n),slice width,phi,Density ;
   Next 3 lines contain n elements of input
   First n are angles made by each slice with horizontal,
   Next n are Heights of slices assumed,
   Next are the pore pressures at the base of each slice.

2. After the input is taken then the weight of each slice is calculated and stored in an array.

3. Now some variables and counters are declared and they are initialized accordingly .

4. Formula of FS - for calculating the value of the right hand side of the FS we first calculate our Numerator with the assumed FS.
   And then the denominator and Calculate the FS.
   For this we also have to calculate the M_alpha factor.

5. **ITERATION METHOD** - Now the interesting part is the Calculation of FS so here the method of iteration is used. We

assume A FS and check if it satisfies aur equation. If yes then we return the answer else we increase the value of our FS and do this until we get approximately equal values.

6. Now after getting the right value of FACTOR SAFETY(FS) we output the value of FS to the user.

$$\mathbf{FS} = \frac{\sum\left(C \cdot b\sec\alpha + (W - u \cdot b\sec\alpha)\tan\varphi \right) 1/M_\alpha}{\sum W\sin\alpha}$$

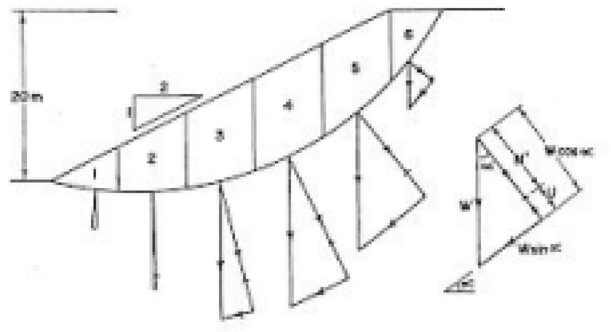
Also,

$$M_\alpha = \cos\alpha + \sin\alpha \cdot \tan\varphi / F$$

Where, c=cohesion W=weight of slice

Alpha= angle of slice b=width of slice

Φ = angle of internal friction u=pore pressure at base of each slice



THE CODE

```
#include <bits/stdc++.h>
using namespace std;
#define int long long


int bishop()
{
        int n;
        double c, b, phi, rho;
        cin >> c >> n >> b >> phi >> rho;
//cohesion ,no. of slices,width of slice,i angle of internal
//friction,density
        double a[n], w[n], h[n], u[n], m[n];
//alpha[i],weight[i],height[i],pore pressure[i],m_alpha[i]..
        //cin the input;
        for (int i = 0; i < n; i++)
```

```
	{
		cin >> a[i];
	}

	for (int i = 0; i < n; i++)
	{
		cin >> h[i];
	}
	for (int i = 0; i < n; i++)
	{
		cin >> u[i];
	}

	//calculate all w[i}

	for (int i = 0; i < n; i++)
	{
		w[i] = rho * h[i] * b;
	}
double fs, numerator = 0, denominator = 0, ans = INT_MAX;

	for (double j = 0.01; j < 10; j+=0.01)
	{
		for (int i = 0; i < n; i++)
		{
```

```
                        m[i] = cos(a[i]) + ((tan(phi) * sin(a[i])) / j);

            numerator+=(c*b/cos(a[i])+(w[i]-u[i]*b/cos(a[i])) * tan(phi)) /
                                                                        m[i];
                        denominator += w[i] * sin(a[i]);
                }


                fs = numerator / denominator;
                if (abs(fs - j) / fs <= 0.01)
                {
                        ans = fs;
                        break;
                }
        }

        cout << ans;
        return 0;
}
signed main()

{
        ios_base::sync_with_stdio(0);
        cin.tie(0);
        cout.tie(0);
        bishop();
        return 0;
}
```

Understanding the code:

We took all the input and calculated the weight of each slice Now we went into the actual calculations by assuming an FS value j and iterating over the formula by putting the desired values and the assumed FS to check which FS is correct.when the values are the same we exit the loop.
We assumed the j=0.01 as our initial FS and continued increasing it till the FS are equal.

SCOPE OF IMPROVEMENT

The above code analyses the stability of slope close to accurate but there can be some changes which can make the solution more accurate, We can assume more failure surfaces along the slope to get the minimum factor safety. We can consider more slices and consider the inter slice forces for more accurate answers.

So, the above program is good but can still be developed more and can be optimized for better results.