

Numerical Differentiation

The formulae for approximating the first, second, and third derivatives of $f'(x)$ are

$$f'(x) \approx \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h}$$

$$f''(x) \approx \frac{-f(x+2h) + 16f(x+h) - 30f(x) + 16f(x-h) + f(x-2h)}{12h^2}$$

$$f'''(x) \approx \frac{f(x+2h) - 2f(x+h) + 2f(x-h) - f(x-2h)}{2h^3}$$

where h is chosen to be very small.

The C++ implementation for each of these is given below.

```
//Approximation of the first derivative of a function at a certain
//point using a 5-point stencil

#include<iostream>
#include<math.h>
#include <assert.h>

double (*func)(double); //points to differentiable function

using namespace std;

double firstDerivative(double f, double x0, double h)
{ //need to specify: f=function, x0=value)

    double answer;

    answer = (-func(x0+2*h)+8*func(x0+h)-8*func(x0-h)+func(x0-2*h))/(12*h);

    return answer;
}
```

```
//Approximation of the second derivative of a function at a certain
```

```

//point using a 5-point stencil

#include<iostream>
#include<math.h>
#include <assert.h>

double (*func)(double); //points to differentiable function

using namespace std;

double secondDerivative(double f, double x0, double h)
{//need to specify: f=function, x0=value)

    double answer;

    answer = (-func(x0+2*h)+16*func(x0+h)-30*func(x0)+16*func(x0-h)-func(x0-2*h))/(12*h*h);

    return answer;
}

```

```

//Approximation of the third derivative of a function at a certain
//point using a 5-point stencil

#include<iostream>
#include<math.h>
#include <assert.h>

double (*func)(double); //points to differentiable function

using namespace std;

double thirdDerivative(double f, double x0, double h)
{//need to specify: f=function, x0=value)

    double answer;

    answer = (func(x0+2*h)-2*func(x0+h)+2*func(x0-h)-func(x0-2*h))/(2*h*h*h);

    return answer;
}

```

The partial derivatives of a function of several variables can be found using this method by making only small modifications to the above codes. Just as analytically solving for the partial derivative of a function of more than one variable requires keep all other variables constant, so too does numerically solving for the partial derivative. For example, suppose we have a function $f(x, y)$.

$$\frac{\partial f}{\partial x} \approx \frac{-f(x+2h, y) + 8f(x+h, y) - 8f(x-h, y) + f(x-2h, y)}{12h}$$

The code for the approximation of $\frac{\partial f}{\partial x}$ is given below.

```
//Approximation of the first partial derivative of a function at a certain
//point with respect to x using a 5-point stencil

#include<iostream>
#include<math.h>
#include <assert.h>

double (*func)(double, double); //points to differentiable function

using namespace std;

double partialX(double f, double x0, double y0, double h)
{//need to specify: f=function, x0= x value, y0= y value)

    double answer;

    answer = (-func(x0+2*h, y0)+8*func(x0+h, y0)-8*func(x0-h, y0)+func(x0-2*h, y0))/(12*h);

    return answer;
}
```

This can easily be extended to finding $\frac{\partial^2 f}{\partial x^2}$ or $\frac{\partial f}{\partial y}$ or $\frac{\partial^2 f}{\partial x \partial y}$ or any other mixed partial derivative.