Lecture 15

Wednesday, February 12, 2020

We have learned how to use bisection method and Newton's method to solve approximately the equation $f(x) = 0$. Now we want to solve a system of equations, the simplest of which is a system of two equations and two unknowns of the form

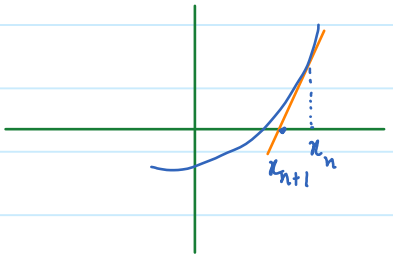
$$\begin{cases} f(x,y) = 0, \\ g(x,y) = 0. \end{cases}$$

In other words, we want to search on the plane for a point (x,y) that satisfies the system. The idea of bisection method is very elegant 1D. Roughly speaking, if a chicken can only run back and forth on a straight line, we can easily catch it by narrowing the interval where it is allowed to run.

In 2D or higher dimensions, it is hard to catch the chicken because there are more directions and spaces for it to escape. In the same way, it is difficult to "trap" the solution of a system of equation. In 1976, Harvey and Stenger found an analog of bisection method in 2D. Their approach relies on an advanced mathematical tool called topological degree (introduced by Brouwer, 1911).

[The link to their paper is here: https://pdfs.semantics...]

Newton's method, on the other hand, remains relatively simple on higher dimension. Recall that the key idea of Newton's method is linearization.



We start at x_0, the first guess for root. We "correct" this guess by x_1. Then correct the guess by x_2, and so on.

At step n, we have x_n. How to correct x_n? (i.e. how to obtain x_{n+1}?)

We approximate the graph of f by the tangent line at x_n. Instead of searching for the intersection of the graph and the x-axis, we search for the intersection of the tangent line and the x-axis. This intersection is called x_{n+1}.

$$f(x) \approx \underbrace{f(x_n) + f'(x_n)(x - x_n)}_{\text{linearization of } f \text{ at } x_n}$$

x_{n+1} is the root of the equation:

$$f(x_n) + f'(x_n)(x - x_n) = 0.$$

This equation is easy to solve because it is a linear equation. We get

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

Let us apply the same idea for the system of equation

$$\begin{cases} f(x,y) = 0, \\ g(x,y) = 0. \end{cases}$$

Put $X = \begin{bmatrix} x \\ y \end{bmatrix}$, $F(X) = \begin{bmatrix} f(X) \\ g(X) \end{bmatrix} = \begin{bmatrix} f(x,y) \\ g(x,y) \end{bmatrix}$.

We want to solve for vector $X \in \mathbb{R}^2$ such that $F(X) = 0$.

The key idea is to linearize F. Note that F is a function from $\mathbb{R}^2$ (or a subset of $\mathbb{R}^2$) to $\mathbb{R}^2$. Taylor approximation can be defined for multivariable functions (this topic is usually omitted from Calculus syllabus) The first Taylor approximation of F at $X_n = \begin{bmatrix} x_n \\ y_n \end{bmatrix}$ is

$$\underbrace{F(X)}_{2\times 1} \approx \underbrace{F(X_n)}_{2\times 1} + \underbrace{DF(X_n)}_{2\times 2} \underbrace{F(X_n)}_{2\times 1}$$

matrix multiplication

Here $DF(X_n)$ is the derivative matrix (or Jacobian matrix) of F at X_n

$$DF(X_n) = \begin{bmatrix} \frac{\partial f}{\partial x}(X_n) & \frac{\partial f}{\partial y}(X_n) \\ \frac{\partial g}{\partial x}(X_n) & \frac{\partial g}{\partial y}(X_n) \end{bmatrix}$$

← The first row is the gradient of f.

← The second row is the gradient of g.

Instead of solving for X such that $F(X)=0$, we solve for X such that $F(X_n) + DF(X_n)(X-X_n) = 0$. The second equation is easier to solve because it is a matrix equation.

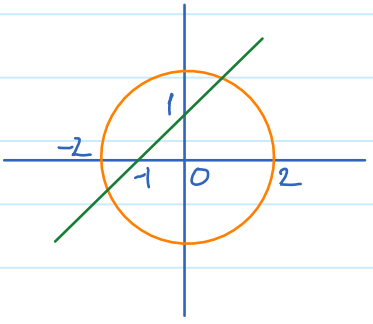
$$DF(X_n)(X-X_n) = -F(X_n).$$

The root is called X_{n+1}:

$$X_{n+1} = X_n - [DF(X_n)]^{-1} F(X_n).$$

This is the iteration formula of the Newton's method. It tells us how to "correct" X_n to get a better approximation for the root. The formula is applicable to any dimensions, not just two dimensions. In other words, one can use Newton method to solve a system of k equations and k unknowns.

Ex



Let us consider the circle $x^2+y^2=4$ and the line $y=x+1$. They have two intersection points as we can see on the picture. How do we solve approximately for these intersection points? We need to solve the system

$$\begin{cases} x^2+y^2 = 4, \\ y = x+1. \end{cases}$$

We then bring the system into a form where the RHS are 0.

$$\begin{cases} x^2+y^2-4 = 0, \\ y-x-1 = 0. \end{cases}$$

But $f(x,y) = x^2+y^2-4$, $g(x,y) = y-x-1$.

The system now becomes $\begin{cases} f(x,y)=0, \\ g(x,y)=0. \end{cases}$

But $X = \begin{bmatrix} x \\ y \end{bmatrix}$ and $F(X) = \begin{bmatrix} f(X) \\ g(X) \end{bmatrix} = \begin{bmatrix} x^2+y^2-4 \\ y-x-1 \end{bmatrix}$

Then the iteration formula of the Newton's method is

$$X_{n+1} = X_n - [DF(X_n)]^{-1} F(X_n). \qquad (*)$$

We need to compute the derivative matrix and its inverse.

$$\frac{\partial f}{\partial x}(x,y) = 2x \;, \qquad \frac{\partial f}{\partial y}(x,y) = 2y,$$

$$\frac{\partial g}{\partial x}(x,y) = -1 \;, \qquad \frac{\partial g}{\partial y}(x,y) = 1$$

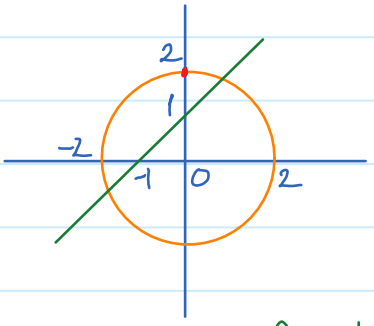
Thus, $DF(x,y) = \begin{bmatrix} 2x & 2y \\ -1 & 1 \end{bmatrix}$.

The inverse of this 2x2 matrix is $[DF(x,y)]^{-1} = \frac{1}{2x+2y} \begin{bmatrix} 1 & -2y \\ 1 & 2x \end{bmatrix}$.

Now we can rewrite (*) as

$$\begin{bmatrix} x_{n+1} \\ y_{n+1} \end{bmatrix} = \begin{bmatrix} x_n \\ y_n \end{bmatrix} - \frac{1}{2x_n+2y_n} \begin{bmatrix} 1 & -2y_n \\ 1 & 2x_n \end{bmatrix} \begin{bmatrix} x_n^2+y_n^2-4 \\ y_n-x_n-1 \end{bmatrix}$$

This is the iteration formula to find (x_{n+1}, y_{n+1}) from (x_n, y_n). One can pick the initial point (x_0, y_0) close to the true solution. For example, one can pick $(x_0, y_0) = (0, 2)$.



We can ask Matlab to do the iterations for us by writing (roughly):

x=0 ; y=2 ; n=10
(initial point; number of iterations)

```
for k=1:n
    a= [x;y]- ... [1 -2y; 1 2x]* [x^2+y^2-4 ; y-x-1]
    x = a(1);   (or a(1,1))   } updating
    y = a(2);   (or a(2,1))   } x and y
end
```