Newtons Method Step

Denoting $f'(x)$ as the first derivative of continuous function $f(x)$, solving the problem of finding the extrema (i.e. roots) of $f(x)$ is equivalent to solving the nonlinear equation

$$f'(x) = 0$$

If we have a current guess of x_n, we can approximate $f'(x)$ near x_n using a first order Taylor expansion

$$f'(x) \approx f'(x_n) + f''(x_n)(x - x_n)$$

This linear approximation is just the tangent line to $f'(x)$ at x_n. Using this representation, instead of solving for $f'(x) = 0$ directly, instead we solve for the root of the tangent line

$$0 = f'(x_n) + f''(x_n)(x_{n+1} - x_n)$$

which, rearranging, gives the Newton iteration formula

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

Newtons Method Algorithm

Starting at some initial guess x_0, calculate $F(x_0)$ and $H(x_0)$, then use these to determine the step

$$\delta x_1 = -\frac{1}{H(x_0)} F(x_0)$$

Adding this step to the x_0, the next test point is found

$$x_1 = x_0 + \delta x_1$$

This process is repeated iteratively

$$x_{k+1} = x_k + \delta x_{k+1}$$

until $F(x_k)$ becomes sufficiently small or the step δx_{k+1} is sufficiently small.

$$|F(x_k)| < e_F \quad \text{or} \quad |\delta x_{k+1}| < e_x$$

This represents convergence upon a local minima (or maxima) or that the algorithm is stuck in a saddle point.

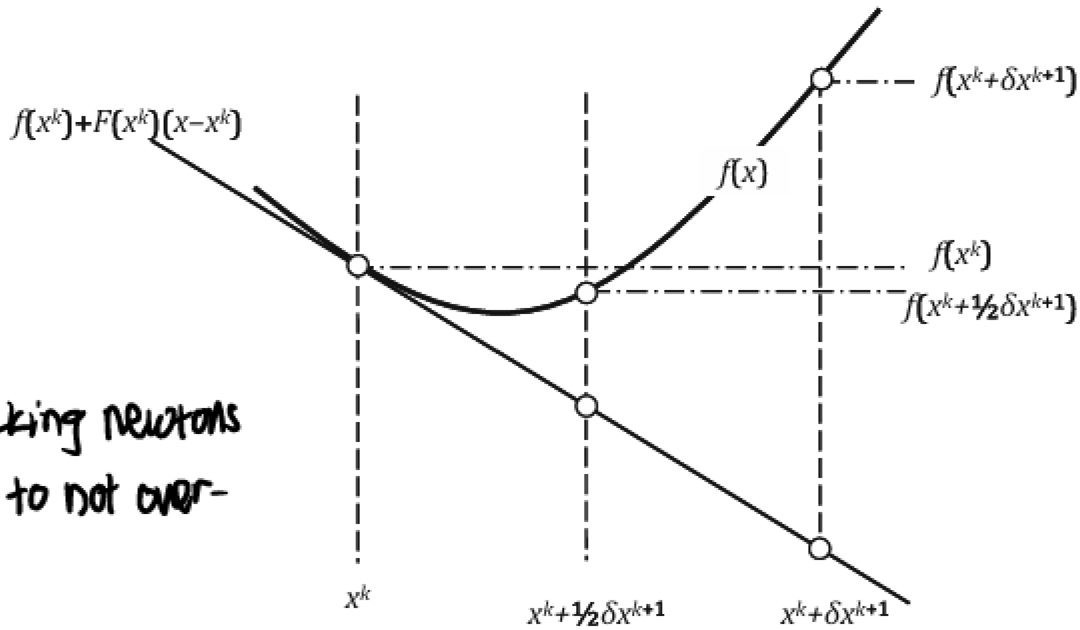
Backtracking

Becuase the basic Newton's method is not very successful in practical applications (unless we have a priori information of our solution location), the algorithm is altered to be more robust. Instead of unconditionally accepting the full Newton's step, given we have no guaruntee that $f(x_{k+1}) < f(x_k)$, we iteratively backtrack the full step size searching until $f(x_{k+1}) < f(x_k)$ is satisfied. One implementation of this methodology is as follows

$$x_{k+1} = x_k + \left(\frac{1}{2}\right)^n \delta x_{k+1}$$

Starting at $n = 0$, we iteratively check if f has been decreased. If not, n is increased and the process is repeated a finite number of times to avoid termination.

Other techniques for controlling Newton's step size include scaling by a constant or performing an exact line search. These two extremes trade off number of iterations for complexity in terms of computational cost.



backtracking newtons method to not over-step

Fig. 1.16 Naïve backtracking algorithm (note that taking a full Newton step results in an increased value of $f(x)$, while one backtracking step results in a decreased value)

$$x^{k+1} = x^k + \left(\frac{1}{2}\right)^n \delta x^{k+1}, \; n \text{ is smallest } n : f\left(x^{k+1}\right) < f\left(x^k\right) \text{ is satisfied} \qquad (1.91)$$