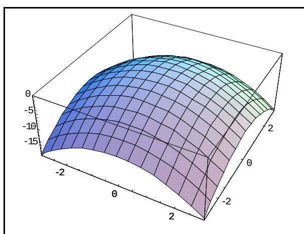
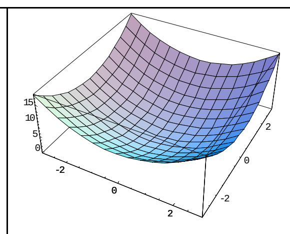
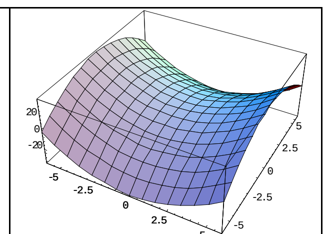
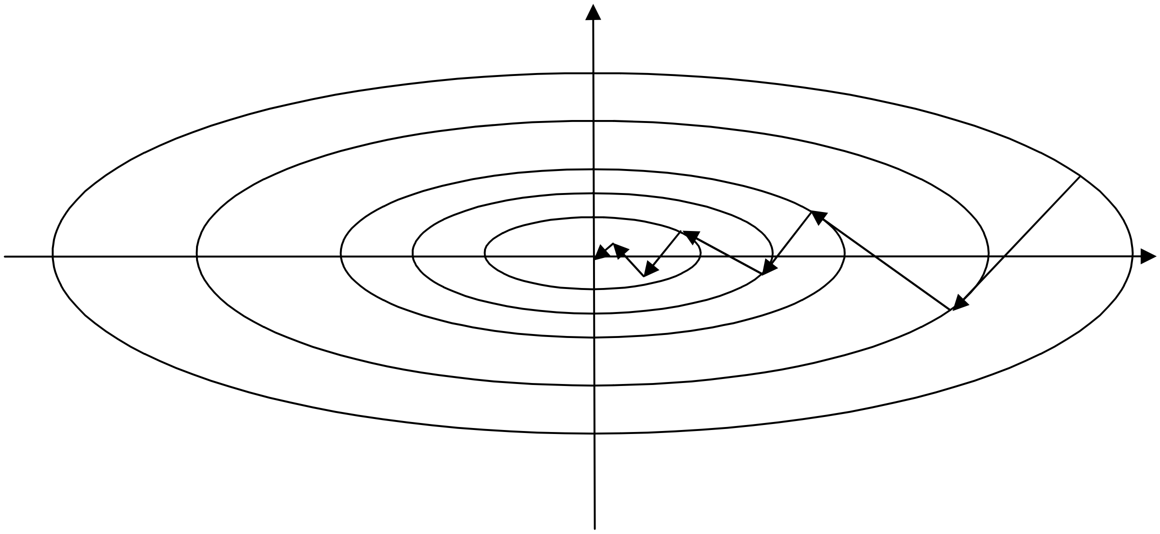
CS205 – Class 8

Covered In Class: 1, 3, 4, 5, 6
Reading: Heath Chapter 6

1. **Optimization** – given an **objective function** f, find relative maxima or minima. Note that since max f = $\min -f$ it is enough to only consider minima.
    a. We'll start with scalar functions f of one variable for now.
    b. **unconstained** – any $x \in \mathbf{R}^{\mathbf{r}}$ is acceptable
    c. **constrained** – minimize f on a subset $S \neq \mathbf{R}$
    d. usually find *local minima*, since global minima are hard to find
        i. one option is to find many local minima and compare them to find a global minimum
    e. **Not equivalent** to solving for *f(x)* = 0. There might exist no such x or the minimum may be attained somewhere *f(x)* < 0.
    f. *poorly conditioned* since $f'(x)=0$ at a minimum, i.e. locally flat (similar to a multiple root) – error tolerance should be more like $\sqrt{\varepsilon}$ as opposed to ε
    g. given a critical point where $f'(x)=0$, we can use the sign of the second derivative to determine whether we have a local minimum, a local maximum, or an inflection point
        i. if $f''(x)>0$, concave up, minimum
        ii. if $f''(x)<0$, concave down, maximum
        iii. otherwise when the second derivative vanishes, we have an inflection point, i.e. neither a minimum nor a maximum

		
Local maxima	Local minima	Saddle

    h. **unimodal** – $[a, x^*]$ is monotonically decreasing and $[x^*, b]$ is monotonically increasing x^* is the minimum – most schemes need a unimodal interval in order to converge
2. **golden section search** - Using the magic number $\tau = (\sqrt{5}-1)/2$ satisfying $\tau^2 = 1-\tau$
    a. starting with an interval $[a,b]$, find $x_1 = a + (1-\tau)(b-a)$ and $x_2 = a + \tau(b-a)$
    b. if $f(x_1) > f(x_2)$, discard $[a, x_1]$ and set $a = x_1$, $x_1 = x_2$ and $x_2 = a + \tau(b-a)$ (using the updated value of a) noting that $x_1 = a + (1-\tau)(b-a)$ (again, for the updated value of a) is still true (this is the reason for the magic number)
    c. otherwise discard $[x_2, b]$ and set $b = x_2$, $x_2 = x_1$ and $x_1 = a + (1-\tau)(b-a)$ (using the updated value of a) noting that $x_2 = a + \tau(b-a)$ (again, for the updated value of a) is still true (this is the reason for the magic number)
    d. stop when $b - a < \varepsilon$



e. linearly convergent with $C \approx .618$

3. **Newton's Method** $x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$ since we're looking for $f'(x) = 0$ instead of $f(x) = 0$
    a. in general f' is hard to find, here f'' is needed as well which is even worse
        i. secant type methods can be used to replace the second derivative with first derivatives
        ii. one could also replace all the first derivative with function evaluations
    b. **mixed methods** can be used, for example combining Newton iteration with golden section search.
4. In multiple spatial dimensions, we need to find a vector function $\vec{x}$ that minimizes $y = f(\vec{x})$
    a. Instead of solving the scalar equation $f'(x) = 0$ to find potential solutions, we need to solve the system of equations $\nabla f(\vec{x}) = 0$. That is $\partial f / \partial x_1(\vec{x}) = 0$, … , $\partial f / \partial x_i(\vec{x}) = 0$, … , $\partial f / \partial x_n(\vec{x}) = 0$ where there is one equation for each x_i
    b. once a potential solution is found we need to classify it as a maximum, minimum, or neither
        i. in 1D we looked at $f''(x) = 0$
        ii. in multiD, we look at the **Hessian matrix**, H(x), of 2nd partial derivatives where $H_{ij} = \partial^2 f / \partial x_i \partial x_j$
        iii. If *f* has continuous second partial derivatives, the order of differentiation does not matter, i.e. $H_{ij} = \partial^2 f / \partial x_i \partial x_j = \partial^2 f / \partial x_j \partial x_i = H_{ji}$, and H is symmetric
        iv. At a critical point where the system corresponding to $\nabla f(\vec{x}) = 0$, we have:
            1. if H is positive definite, then x is a local minimum of f
            2. if H is negative definite, then x is a local maximum of f
            3. otherwise H is indefinite and x is a saddle point
        v. In 1D H=[f''(x)], and this is positive or negative definite if f''(x) is positive or negative, respectively. Also, when f''(x)=0, it is indefinite. (Aside: a major theme of the course is that intuition is built through examining the scalar case)
        vi. There are many ways to see if a symmetric matrix is positive definite, e.g. compute the eigenvalues and see if they are all positive, although this is one of the more expensive ways
        vii. Think of this as walking down hill. But of course, if you never go uphill you can fall into local minima.
5. **steepest descent method** - look in the $-\nabla f$ direction, i.e. the direction where f is decreasing fastest
    a. look for a minimum on the parametric line $\vec{x}(t) = \vec{x}_k - \nabla f(\vec{x}_k)t$
    b. that is, find the min $f(\vec{x}(t))$ = min $f(\vec{x}_k - \nabla f(\vec{x}_k)t)$ = min g(t) where g is a one dimensional function of the one dimensional parameter t
        i. use a 1D solver

c. given t^* that minimizes g(t), set $\vec{x}_{k+1} = \vec{x}_k - \nabla f(\vec{x}_k) t^*$ and continue iterating

d. stop when $\|\nabla f\|$ is small

6. Conjugate Gradient Method – brief introduction
    a. steepest descent can converge slow due to repeated searching in the same direction, i.e. overlapping components in the search direction
    b. **conjugate gradient method** – avoids repeated searches in the same direction making it faster than steepest descent
    c. steepest descent in vector notation $\vec{x}_{k+1} = \vec{x}_k + \vec{s}_k \alpha$ where the search direction is $\vec{s}_k = -\nabla f(\vec{x}_k)$ and α comes from the 1D minimization of $F(\alpha) = f(\vec{x}(\alpha)) = f(\vec{x}_k + \vec{s}_k \alpha)$
    d. use $\vec{s}_o = -\nabla f(\vec{x}_o)$ so the first step is the same for both methods
    e. then $\vec{s}_{k+1} = -\nabla f(\vec{x}_{k+1}) + \dfrac{\nabla f(\vec{x}_{k+1})^T \nabla f(\vec{x}_{k+1})}{\nabla f(\vec{x}_k)^T \nabla f(\vec{x}_k)} \vec{s}_k$
    f. theoretically, the exact solution is obtained after at most n (# of dimensions) iterations for quadratic functions
    g. because of numerical errors, every n iterations, start over with $\vec{s}_k = -\nabla f(\vec{x}_k)$
    h. The conjugate gradient method covers more than just optimization, e.g. we'll use it later as an iterative solver to aid in solving PDE's