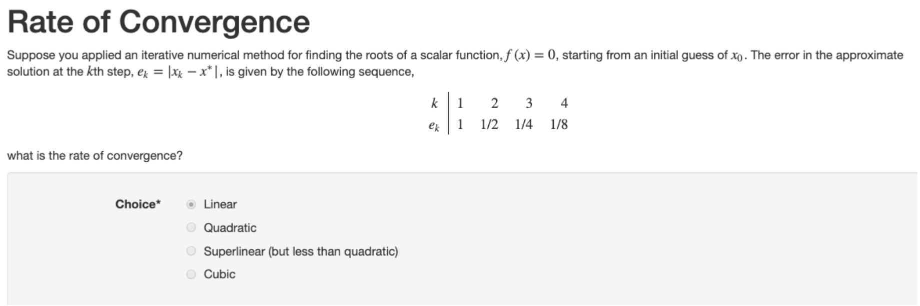
1 Lecture 16

Newton

1.1 Quiz

Problem 1.



Rate of Convergence

Suppose you applied an iterative numerical method for finding the roots of a scalar function, $f(x) = 0$, starting from an initial guess of x_0. The error in the approximate solution at the kth step, $e_k = |x_k - x^*|$, is given by the following sequence,

k	1	2	3	4
e_k	1	1/2	1/4	1/8

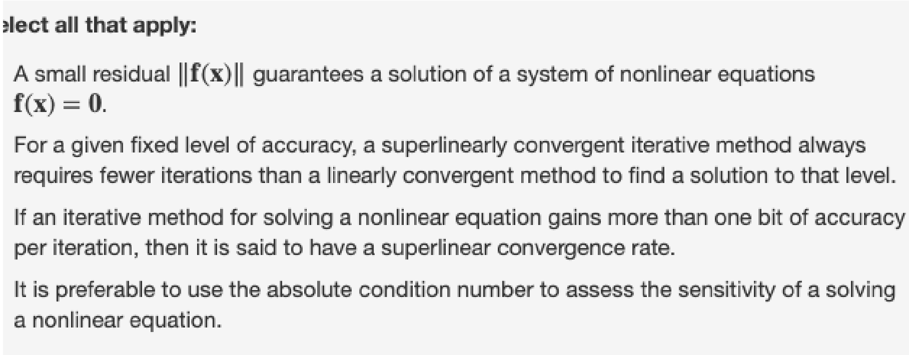
what is the rate of convergence?

Choice*
- ● Linear
- ○ Quadratic
- ○ Superlinear (but less than quadratic)
- ○ Cubic

Solution 1.1. The error is increasing by a constant multiplicative factor on every iteration; hence the error must be linear.

Problem 2.

Which of the following is true:



elect all that apply:

A small residual $\|\mathbf{f}(\mathbf{x})\|$ guarantees a solution of a system of nonlinear equations $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.

For a given fixed level of accuracy, a superlinearly convergent iterative method always requires fewer iterations than a linearly convergent method to find a solution to that level.

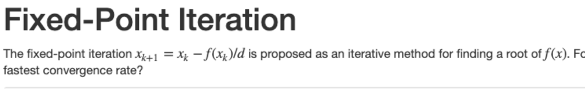
If an iterative method for solving a nonlinear equation gains more than one bit of accuracy per iteration, then it is said to have a superlinear convergence rate.

It is preferable to use the absolute condition number to assess the sensitivity of a solving a nonlinear equation.

Solution 1.2. A small residual may just indicate that the values of $\|f\|$ are generally quite small. The second: if the convergence constant is above 1, then in fact, the method will not converge – so it really depends on the convergence constant (if the convergence constant is less than 1 and we start off with the same initial error, the n the answer is yes). Superlinear methods need to have an increasing number of bits that they gain. In general, if the initial error is e and the the convergence is rate r with some constant less than 1 as C then the error at stage m can be seen just by repeatedly substituting Ce^r into the cheapo version of this equation and observing how the resulting value changes.

Problem 3.

Which value of d allows us to achieve optimal convergence.

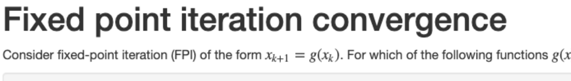


Fixed-Point Iteration

The fixed-point iteration $x_{k+1} = x_k - f(x_k)/d$ is proposed as an iterative method for finding a root of $f(x)$. Fo
fastest convergence rate?

Solution 1.3. Note that this question is a bit misleading in that we can only determine if a variant of a fixed point function is better than another variant on the basis of its behavior near a fixed point. Away from the fixed point, all bets are off. Thus, we really should be comparing, for each value of d, what the consequent derivative of the fixed point function $g(x) = x - f(x)/d$ is at the fixed point of the function f. If, however, we assume that the derivative of g as 2 sufficiently approximates the derivative of g at $\sqrt{5}$ which is the fixed point that g intends to navigate to, then we find that $g' = 1 - 4/d$ at $x = 2$ making $d = 4$ the best choice.

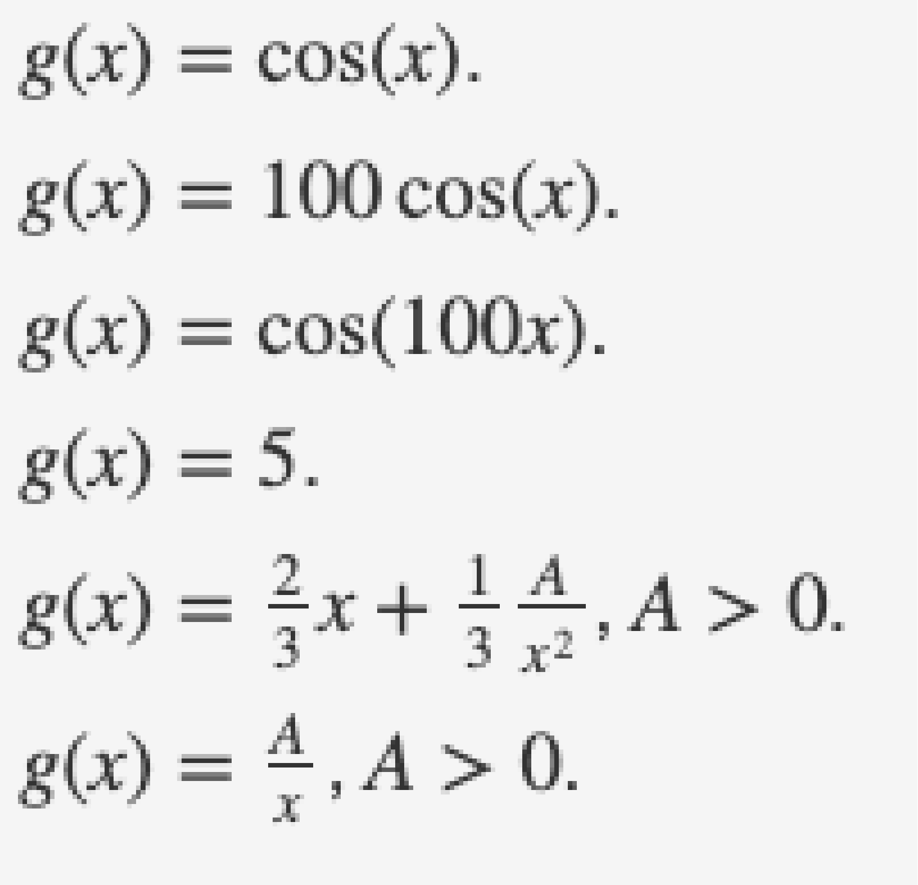
Problem 4.



Fixed point iteration convergence

Consider fixed-point iteration (FPI) of the form $x_{k+1} = g(x_k)$. For which of the following functions $g(x$

Which value of d allows us to achieve optimal convergence.



$g(x) = \cos(x).$

$g(x) = 100\cos(x).$

$g(x) = \cos(100x).$

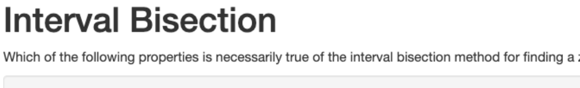
$g(x) = 5.$

$g(x) = \frac{2}{3}x + \frac{1}{3}\frac{A}{x^2}, A > 0.$

$g(x) = \frac{A}{x}, A > 0.$

Solution 1.4. The strategy is to either realize that some functions will always have a derivative at any point less than 1 – meaning that if the function does have a fixed point, then it will converge to the fixed point sufficiently closed to it. These functions are $g(x) = \cos(x)$ and $g(x) = 5$. Otherwise, for the remaining functions solve for $g(x^*) = x^*$ and see if $g'(x^*) < 1$. The only function that satisfies this for all values of A is $g(x) = \frac{2}{3}x + \frac{1}{3}\frac{A}{x^2}$; the other $g(x, A)$ function does not; finally, I am unsure how we invalidate the functions $g(x) = 100\cos(x)$ and $g(x) = \cos(100x)$.

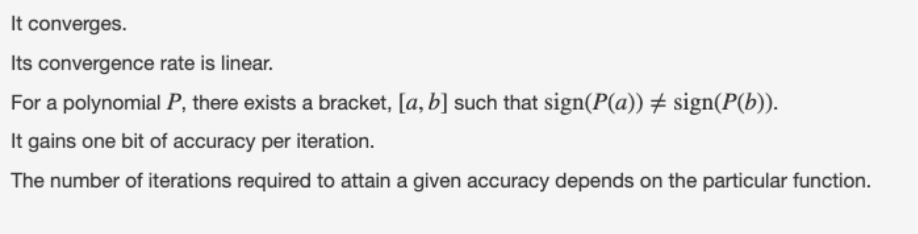
Problem 5.



Interval Bisection

Which of the following properties is necessarily true of the interval bisection method for finding a

Which value of d allows us to achieve optimal convergence.



It converges.

Its convergence rate is linear.

For a polynomial P, there exists a bracket, $[a, b]$ such that $\text{sign}(P(a)) \neq \text{sign}(P(b))$.

It gains one bit of accuracy per iteration.

The number of iterations required to attain a given accuracy depends on the particular function.

Solution 1.5. Note that if we have a starting interval $[a, b]$ and we wish to reduce the error to tol, then we need to solve for the k such that

$$\frac{(b-a)}{2^k} = tol$$

which turns out to be $\log(\frac{b-a}{tol})$. This is true for any function.

Note that given an interval where we've identified a root to fall into, this method always converges; its convergence is linear, because we halve the interval each time; the third remark is non-sequitur and useless; the fourth remark is true because linearly convergent methods gain a constant number of bits per iteration and here in particular, since the error goes down by a half each iteration, we gain one bit in accuracy (assuming that our interval $b - a$ is intially less than 1).

Definition 1.6. We want to find the h such that $f(x+h) = f(x) + hf'(x) = 0$. This sets $h = \frac{-f(x)}{f'(x)}$, which implies that given an initial x_k,

$$x_{k+1} = x_k + h = x_p + \frac{-f(x_p)}{f'(x_p)}$$

Set $g(x)$ to be

$$x + \frac{-f(x)}{f'(x)}$$

Observe that if $f'(x) = 0$ (ie if f has a double root at a fixed point x^*, then $g(x^*)$ is indeterminate and Newton fails.

Observe that if Newton's method hits a point y such that $f(y) = 0$, then y is fixed point, since all successive iterates will still remain at y.

Also observe that $g'(x) = \frac{f(x)f''(x)}{f'(x)^2}$ so that if $f(x) = 0$ but $f'(x) \neq 0$ (ie x is a simple root), then fixed point iteration converges quadratically if we are close enough to the root – then newton's method converges quadratically (assuming that there is no double root). Even in the event that a double root exists, it can be shown that the convergence is then linear with constant $C = 1 - 1/m$ where m is the multiplicit of the root.

Remark 1.7. There are two downsides to using Newton's method:

- We need the derivative.

- If we want other roots, we need to hope that some choice of another initial vector will converge to root different than the one we first found.

Remark 1.8. It is claimed that even if newton is not quadratically convergent, then it is at least linearly convergent since the derivative of the fixed point function $g'(x) = \frac{f(x)f''(x)}{f'(x)^2}$ is such that $f(x)f''(x) \to 0$ faster than $f'(x) \to 0$. Why this is true is unresolved. Indeed, Heath on page 251 explains that Newton's method is convergent with linear rate $1 - 1/m$ where m is the multiplicity of the root for $m \geq 2$. It is claimed that $f(x) \to 0$. Newton breaks down in the event that the derivative function does not approximate a function very well or if the derivative function change sign often, taking us "left" and "right" of the roots repeatedly.

Definition 1.9. The secant method works by replacing $f'(x)$ in Newton's method with $s = f(x_k) - f(x_{k-1})$. We can also use a quadratic interpolant. What does this mean? Recall that we approximated $f(x+h)$ by $f(x) + hf'(x)$. We could have also approximated $f(x+h)$ by $f(x) + hf'(x) + h^2 f''(x)/2$. Then we need to approximate $f'(x)$ and $f''(x)$ using three points (can you guess what we might do?). Alternatively, we find an inverse quadratic interpolant that takes three points $(x, f(x)), (y, f(y)), (z, f(z))$ and then finds an inverse interpolant q such that $q(f(x_i)) = x_i$. Having found this interpolant, we then let the next point be $q(0)$, since we expect that $f(q(0)) = 0$.

Remark 1.10. Both the use of newton's method through a quadratic interpolant and inverse quadratic iteration converge with superlinear rate $r \approx 1.81$.

Remark 1.11. Safeguarded methods combine several numerical methods together; for example:

Suppose that we apply newton's method within a bracket (that is, we are given an input that falls in some set S). If it happens that $x_{k+1} \notin S$, then we apply bisection and return the midpoint of the half that bisection concludes that we should use. If it does converge, then use Newton's iterate as the solution.

2 Lecture 17

Remark 2.1. Newton's method performs poorly if there are multiple roots.

Definition 2.2. In n dimensions, fixed point iteration generalizes as follows:

- Suppose that $g : \mathbb{R}^n \to \mathbb{R}^n$. Then if $p(J(g(x^*))) < 1$, g converges to its fixed point x^* linearly. Note that p is the spectral radius.
- If $J(g(x^*)) = 0$, then g converges to x^* quadratically.

Remark 2.3. Bisection is hard to generalize for even if we could recursively find good bounds for a function f_1 where $f = [f_1 \dots f_n]$, bounds particular to f_1 are likely unrelated to the bounds needed for f_2.

Definition 2.4. We can approximate $f : \mathbb{R}^n \to \mathbb{R}^n$ as

$$f(x+h) = f(x) + J(f(x))h + O(h^2)$$

solving for h we find

$$h = -J(f(x))^{-1}f(x)$$

meaning that the new iteration scheme is:

$$x_{k+1} = x_k - J(f(x))^{-1}f(x) \qquad ()$$

Remark 2.5. This has a couple of downsides; namely, computing the Jacobian may be expensive; moreover, computing the inverse may be expensive; finally this is only locally convergent (as before).

Remark 2.6. UNRESOLVED: when is this method quadratically convergent?

Remark 2.7. The secant method is hard to generalize, because it is not clear how we can obtain an approximation to the Jacobian $J(f(x))$ using only $f(x_1), f(x_2), x_1, x_2$ although obtaining an approximation to $f'(x)$ given $f(x_i)$ and x_i is certainly plausible. Broyden's method starts with a guess for $J(f(x))$, and then with some update akin to (α)it then finds the next Jacobian.

2.1 Optimization

Lemma 2.8. If a function is continuous on a closed and bounded set $S \subseteq \mathbb{R}^n$, then it attains its minimum and maximum. See `https://en.wikipedia.org/wiki/Extreme_value_theorem#Proof_of_the_extreme_value_theorem` for the proof.

Definition 2.9. A function is coercive if $\lim_{\|x\|\to\infty} f(x) = \infty$. A coercive function attains a global minimum, although the minimum may not be unique. Think of a cubic function with two dips.

Definition 2.10. A set S is convex if for $x, y \in S$ it holds that $x(\alpha)+y(1-\alpha) \in S$ for all $\alpha \in [0, 1]$. A function f is convex if it holds that $f(x\alpha + y(1 - \alpha)) \leq \alpha f(x) + (1 - \alpha)f(y)$.

Proposition 2.11. If a function f is continuous and convex, then it attains a minimum; if it is instead strictly convex and continuous, then its minimum is unique.

Lemma 2.12. The following are sufficient and necessary conditions for minimality in 1 dimension and in $\mathbb{R}^n$.

- $\mathbb{R}^1$:
  - $f'(x) = 0$ is a necessary condition.
  - In conjunciton with $f''(x) > 0$, the foregoing is a sufficient condition.
  - $\nabla f(x) = 0$ is a necessary condition.
  - In conjunction with $H(f(x)) > 0$ – ie the Hessian being positive definite – we get a sufficient condition.

Definition 2.13. A matrix admits a Cholesky Decomposition, a factorization of the form LL^* where L is lower triangular, iff it is positive definite hermitian. In particular, if A is positive definite, then the factorization is unique; but if A is merely positive semidefinite, then the factorization is not unique; if A is not event positive semidefinite, then the cholesky decomposition will have a negative entry somewhere along the diagonal (this is not true otherwise).

Remark 2.14. We can find the minimum of a function f by solving for $\nabla f = 0$ using any of the methods that we've developed for multidimensional root finding. To then assert that the found root actually corresponds to a minimum, we need to check whether the Hessian of the matrix is positive definite. Note that since H is symmetric, if we attempt to decompose H into a Cholesky Decomposition and find that it has a negative entry on one of its diagonals, then the matrix is in fact not positive definite.

Proposition 2.15. The error in a solution to an optimization problem is at best 10^{-8}.

Proof. Suppose that x^* is a true minimizer but that we estimate $\hat{x}$ to be the minimizer instead. Let $\hat{x} = x^* + h$. Then observe that

$$f(x^*+h) \approx f(x^*)+f'(x^*)h+f''(x^*)h^2/2 = f(x^*)+f''(x^*)h^2/2 \implies h^2 = \frac{2(f(x^*+h)-f(x^*))}{f''(x^*)}$$

What this implies is that if we take $\hat{x}$ to be a solution and we have $|f(\hat{x} - f(x^*)| < \epsilon$, then our error in $\hat{x}$, which is h, is at best $\sqrt{\frac{2\epsilon}{f''(x^*)}}$. This is about 10^{-8}, implying 8 accurate digits, if $\epsilon = \epsilon_{mach}$. We upper bounded the error in x^*, which suggests that we should revise our original statement to be that we can at least get a tolerance of 10^{-8}. The fact that during the course of a problem, we can conceivably find $\hat{x}$ such that $|\hat{x} - x^*| \approx \epsilon_{mach}$ and declare this value our minimum makes this a best bound however. □

Remark 2.16. The follwoing facts about convexity should be known:

- If a convex function has a local minimum, then the local minimum is, in fact, a global minimum.
- If a strictly convex function has a local minimum, then the local minimum is a unique global minimum.
- If a continuous function is restricted to a closed and bounded set $S \subseteq \mathbb{R}^n$, then $f(S) \subseteq \mathbb{R}$ is compact and, hence, obtains a minimum, so that there is a minimum $x \in S$ of f.
- More generally, if f is a continuous function restricted to a closed but unbounded set and f is coercive, then f attains a minimum. This can be proven. Note that a convex function on a convex set is necessarily continuous at interior points of its domain. This has never been proven.

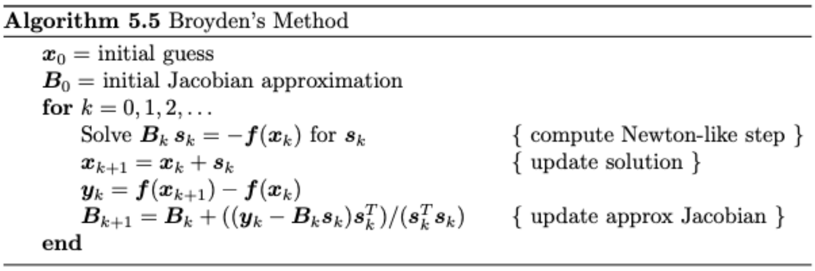
3 Lecture 18

3.1 Quiz

Problem 1.

Comparison of Newton and Broyden Methods Which of the following is NOT an advantage of Broyden's method over Newton's method for solving a system of nonlinear equations? Choice* Matrix factorization can be updated each iteration rather than having to be recomputed No derivatives required Less work per iteration Fewer iterations required for given accuracy

Solution 3.1. Matrix factorization has to be updated in Newton's method everytime – meaning that we have to compute the Jacobian every time. The Jacobian method also requires a Jacobian which is not required by Broyden's method. The update in Broyden's method also resembles a Sherman-Morrison update, making it less costly per iteration to compute the next Jacobian (approximation to the Jacobian) than in Newton's method (this is true in spite of the fact that Broyden's method also computes a difference of the form $f(x_{k+1}) - f(x_k)$.



Algorithm 5.5 Broyden's Method

```
$\boldsymbol{x}_0$ = initial guess
$\boldsymbol{B}_0$ = initial Jacobian approximation
for $k = 0, 1, 2, \ldots$
    Solve $\boldsymbol{B}_k\,\boldsymbol{s}_k = -\boldsymbol{f}(\boldsymbol{x}_k)$ for $\boldsymbol{s}_k$        { compute Newton-like step }
    $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{s}_k$                                  { update solution }
    $\boldsymbol{y}_k = \boldsymbol{f}(\boldsymbol{x}_{k+1}) - \boldsymbol{f}(\boldsymbol{x}_k)$
    $\boldsymbol{B}_{k+1} = \boldsymbol{B}_k + ((\boldsymbol{y}_k - \boldsymbol{B}_k\boldsymbol{s}_k)\boldsymbol{s}_k^T)/(\boldsymbol{s}_k^T\boldsymbol{s}_k)$   { update approx Jacobian }
end
```

Problem 2.

Easy to tell what a convex set looks like.

Problem 3.

Have to review convexity.

Problem 4.

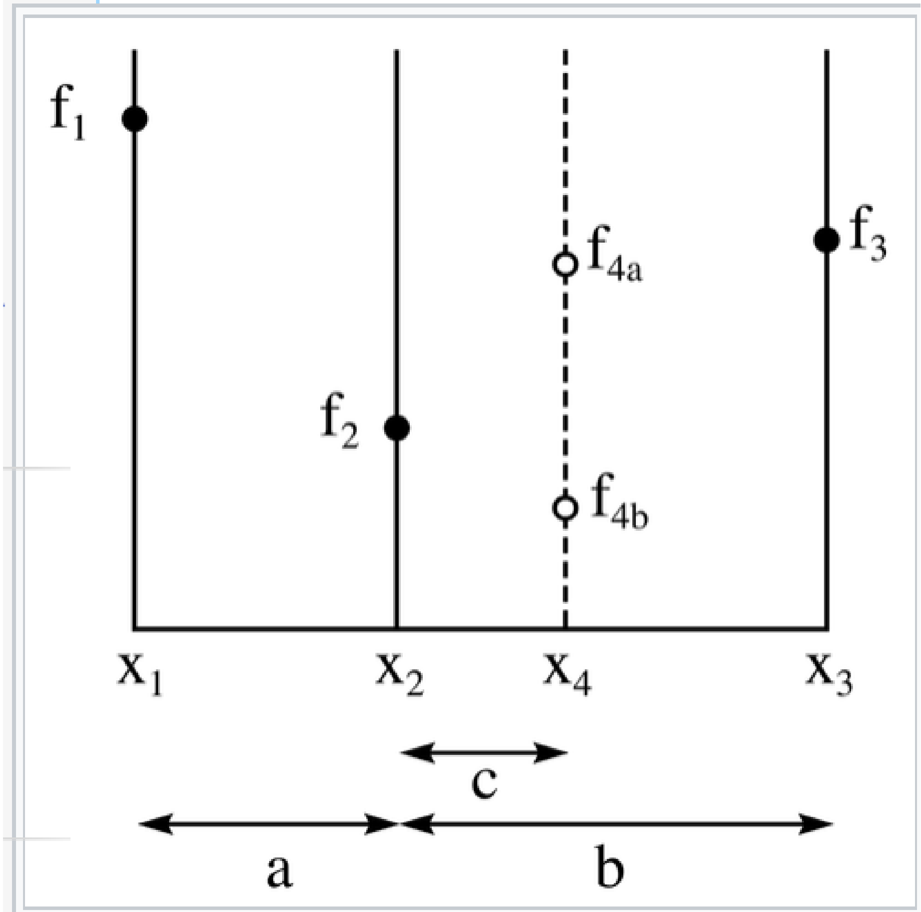
Simply remember that the Hessian is the transpose of the Jacobian of the gradient. ALso remember that if a hessian is neither positive semidefinite nor negative semidefinite at a point where the gradient is zero, then the critical point is a saddle point.

Problem 5.

Optimization Accuracy Suppose that f can be evaluated to within a relative error of 10−10. If x∗ is an extreme point of f, with |f(x∗)|=1, then about how many decimal digits of x∗ can reliably be computed as a solution to an optimization problem?

Solution 3.2. Apply the lemma. from the previous lecture.

Definition 3.3.



In golden section search, given an interval $[x_1, x_3]$ (or in our case the interval is often denoted $[a, b]$) we choose points inside the interval x_2, x_4 (in our case, we define these points to be $a + (b - a)(1 - \tau)$ and $a + (b - a)\tau$ such that

$$\frac{|x_3 - x_2|}{|x_2 - x_1|} = \frac{|x_2 - x_1|}{|x_4 - x_2|}$$

In the diagram above, we see that at any point in the initiation of a golden search we have three points that are known, and depending on the function evaluation of the fourth point, we will choose the subinterval $[x_1, x_4]$ or $[x_2, x_3]$.Thus the stipulation that $c/a = a/b$ represents our desire that when we recurse into a new subinterval that the three given points have been some subset of the givens and function evaluations in the interval "one frame up." Note that this inequality assumes that we progress into the left subinterval; the analysis for the right subinterval is the same, however, since $a + c = b$ and $b - c = a$ (that is $|x_3 - x_4| = |x_2 - x_1|$

It can be shown that solving this proportion requires that τ be the golden ratio.

Definition 3.4. Newton's method for optimization is to use newton's method for root finding in order to solve $f'(x) = 0$. We explore the 1-D case, because this easily generalizes to the n-D case:

$$f(x+h) = hf'(x) + h^2 f''(x)/2 + O(h^3)$$

Differentiate this with respect to h

:

$$f'(x+h) = f'(x) + hf''(x) + O(h^3)$$

Set this to 0 and solve for h

$$h = \frac{-f'(x)}{f''(x)}$$

So the update becomes

$$x \leftarrow x - \frac{f'(x)}{f''(x)}$$

In n-D, this will generalize to:

$$\mathbb{x} \leftarrow \mathbb{x} - (H_f(x))^{-1} \nabla f(x) \tag{β}$$

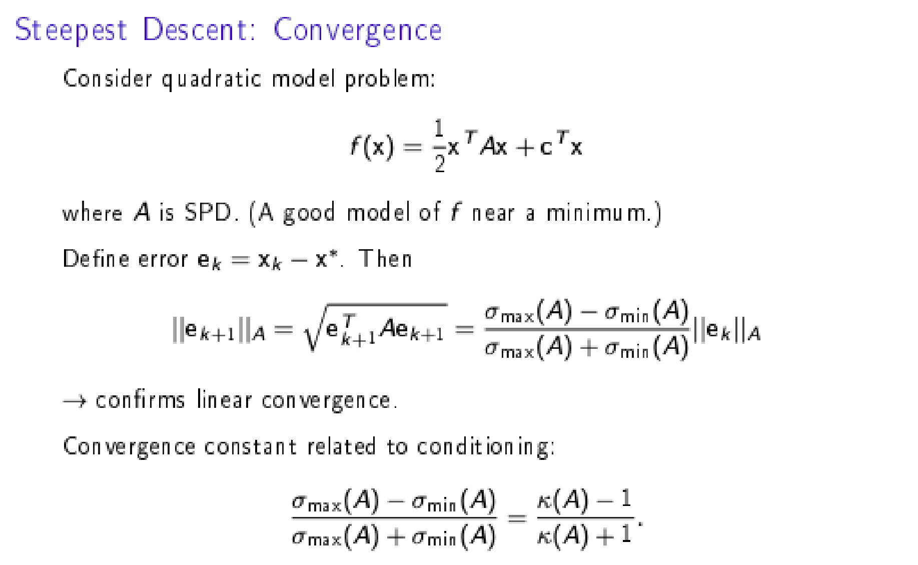
This quadratic approximation of f via a taylor series expansion is only accurate within a region. If we define what that region us as a ball centered at $x$with radius r and we see that $r < \left|(H_f(x))^{-1}\nabla f(x)\right|$ then we can bind the decrement to be precisely $r * (H_f(x))^{-1}\nabla f(x) / \left|(H_f(x))^{-1}\nabla f(x)\right|$.

For reasons that are not clear, this approach (called using a trust region), may also change the direction of the search. Note that we can also employ a so called line search to find the scalar factor of $(H_f(x))^{-1}\nabla f(x) / \left|(H_f(x))^{-1}\nabla f(x)\right|$ that best minimizes (β).

Definition 3.5. Gradient descent repeatedly runs the following function

$$\mathbb{x}_{\mathbb{k}+\mathbb{1}} \leftarrow \mathbb{x}_{\mathbb{k}} - \alpha_k \nabla f(x_k)$$

where α_k is chosen using some other method. For example. One may try to optimize the function $g(\alpha_k) = \mathbb{x}_{\mathbb{k}} - \alpha_k \nabla f(x_k)$.



Steepest Descent: Convergence

Consider quadratic model problem:

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T A\mathbf{x} + \mathbf{c}^T\mathbf{x}$$

where A is SPD. (A good model of f near a minimum.)

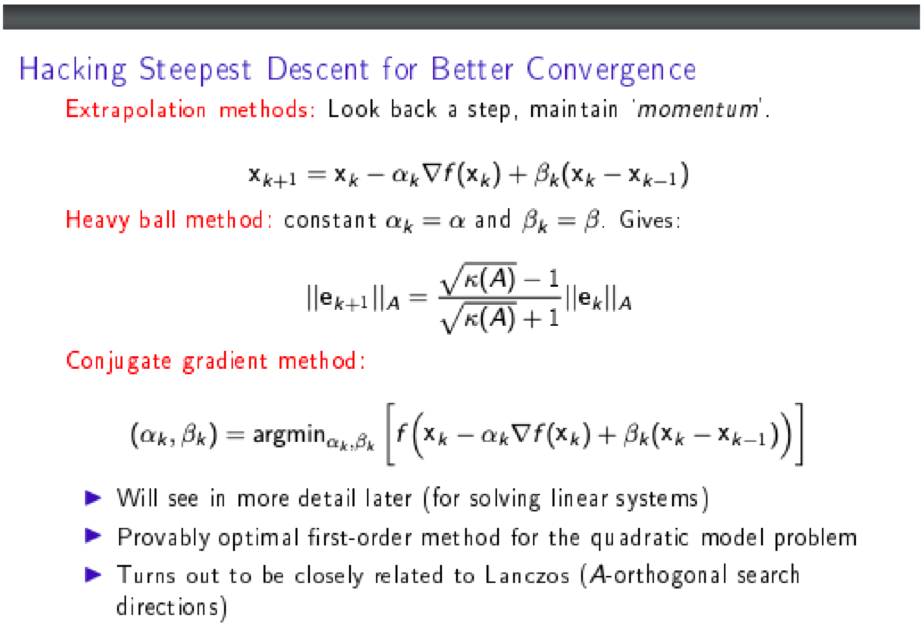
Define error $\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}^*$. Then

$$||\mathbf{e}_{k+1}||_A = \sqrt{\mathbf{e}_{k+1}^T A\mathbf{e}_{k+1}} = \frac{\sigma_{\max}(A) - \sigma_{\min}(A)}{\sigma_{\max}(A) + \sigma_{\min}(A)}||\mathbf{e}_k||_A$$

→ confirms linear convergence.

Convergence constant related to conditioning:

$$\frac{\sigma_{\max}(A) - \sigma_{\min}(A)}{\sigma_{\max}(A) + \sigma_{\min}(A)} = \frac{\kappa(A) - 1}{\kappa(A) + 1}.$$



Hacking Steepest Descent for Better Convergence

Extrapolation methods: Look back a step, maintain *'momentum'*.

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \nabla f(\mathbf{x}_k) + \beta_k(\mathbf{x}_k - \mathbf{x}_{k-1})$$

Heavy ball method: constant $\alpha_k = \alpha$ and $\beta_k = \beta$. Gives:

$$||\mathbf{e}_{k+1}||_A = \frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}||\mathbf{e}_k||_A$$

Conjugate gradient method:

$$(\alpha_k, \beta_k) = \operatorname{argmin}_{\alpha_k, \beta_k}\left[f\left(\mathbf{x}_k - \alpha_k \nabla f(\mathbf{x}_k) + \beta_k(\mathbf{x}_k - \mathbf{x}_{k-1})\right)\right]$$

- Will see in more detail later (for solving linear systems)
- Provably optimal first-order method for the quadratic model problem
- Turns out to be closely related to Lanczos (A-orthogonal search directions)

4 Lecture 19

4.1 Quiz

Problem 1.

Unimodal Function Suppose the real-valued function f(x) is unimodal on the interval [a,b]. Let x1 and x2 be two points in the interval, with a<x1<x2<b. If f(x1)=1.232 and f(x2)=3.576, then which of the following statements is valid on the subinterval [a,b].

Solution 4.1. Recall that unimodality tells us that if we have two points x and y such that $a \leq x < y \leq b$ and $f(x) < f(y)$ then the minimum will be found going “down the slope”, meaning that it will be contained in the interval $[a, y]$. Then tells us that the solution is that the minimum must lie in the interval $[a, x_2]$. There is an option that attempts to dupe us: The subinterval of the minimum of f is indeterminate without knowing the values of $f(a)$ and $f(b)$. This is not true, however – if we are granted unimodality on $[a, b]$ even without knowing the values of a and b we can be assured of the statements in this solution.

Problem 2.

Golden Section Search Suppose we decided to use golden section search for finding the minimum of a unimodal function, but, in the inner loop of the algorithm, instead of picking evaluation points x1,x2 using the formulas

x1=a+(1−)(b−a) x2=a+(b−a) at each iteration (where a and b are the endpoints of the interval, and =12(5√−1)), we chose

x1=a+13(b−a) x2=a+23(b−a). The remainder of the algorithm is the same. This modified strategy is called ternary search.

What is a legitimate disadvantage of using ternary search compared to golden section search?

Solution 4.2. These were the choices:

On average per iteration, ternary search necessarily requires evaluating the function at more points. There are no substantial disadvantages compared to golden section search. The convergence rate of ternary search may be sublinear, while that of golden section search is always linear. Ternary search may fail to converge for some unimodal functions for which golden section search converges.

If we use ternary search, then we don’t preserve the proportion that we listed above, requiring us to have on more function evaluation. Note that with ternary search, the interval length shortens by 2/3 every time. Note that because unimodality is always preserved and the interval decreases, ternary search must converge. Thus, the answer is the first choice.

Problem 3.

Robust Newton Methods For unconstrained optimization of a function f:n→, Newton’s method is often unreliable when started far from the solution. How do line searches or trust regions improve its reliability?

Select all that apply: A trust region modifies the length of the Newton step. A line search modifies the length of the Newton step. A line search modifies the direction of the Newton step. A trust region modifies the direction of the Newton step.

Solution 4.3. See the logic above. The following are true: A trust region modifies the length of the Newton step. A line search modifies the length of the Newton step. A trust region modifies the direction of the Newton step.

Problem 4.

Convergence Rates Recall the definition of convergence rates

limk→∞‖ek+1‖‖ek‖r=C We are trying to minimize

$f(x) = \frac{1}{2}x^T \begin{bmatrix} 4 & 0 \\ 0 & 3 \end{bmatrix} x$ If we are using steepest descent to minimize f, what is r and C? Give 4 significant figures.

Solution 4.4. See the page on convergence rates in the your notes from the previous lecture.

Problem 5.

Newton's Method Consider minimization of the function $\phi(x, y) = (x - 2)^2 + (y - 1)^2$ Starting with an initial guess [x,y]0:=[0,0], what are the values of x and y after one round of Newton's Method

Solution 4.5. Compute the gradient; compute the hessian $A = H_f(0, 0)$

Then solve for

$$A(s_k) = -\nabla f(0, 0)$$

Definition 4.6. Newton's method in N dimensions does the following:

$$f(x + h) = f(x) + \nabla f(x)h + \frac{1}{2}h^T H_f(x)h$$

Now differentiate with respect to h and solve for h

$$0 = \nabla f(x) + H_f(x)h$$

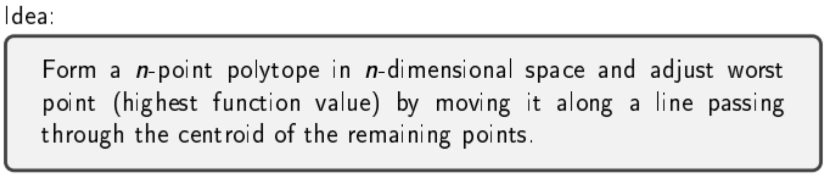
Then set $x_k \leftarrow x_k + h$

Remark 4.7. This method has a few problems:

- It depends on the validity of a Taylor expansion and, hence, is only locally convergent.
- It requires that we compute second derivatives.
- UNRESOLVED (works poorly when H_f is nearly indefinite).

Definition 4.8.

Nelder-Mead Method



Idea:

Form a n-point polytope in n-dimensional space and adjust worst point (highest function value) by moving it along a line passing through the centroid of the remaining points.

Definition 4.9. Let $y - a(x) = r(x)$ and suppose that we wish to minimize

$$\phi(x) = \frac{1}{2} r(x)^T r(x) = \frac{1}{2} \sum_{i=1}^{n} r_i(x)^2$$

$$\frac{\partial}{\partial x_i} \phi(x) = \sum_{j=1}^{n} \frac{\partial r_j(x)}{\partial x_i} (r_j(x)) \implies \nabla \phi(x) = J^T(x) r(x)$$

This then gives:

$$\frac{\partial^2}{\partial x_k \partial x_i} \phi(x) = \sum_{j=1}^{n} \frac{\partial^2 r_j(x)}{\partial x_k \partial x_i} r_j(x) + \frac{\partial r_j(x)}{\partial x_i} \frac{\partial r_j(x)}{\partial x_k} \implies H_\phi(x) = J^T J + \sum_{j=1}^{n} r_j H_{r_j}(x)$$

We assume that the terms $r_j H_{r_j}(x) \approx 0$ so that $H_\phi(x) = J^T J$. Then the step size in newton's method, $-H_\phi(x)^{-1} \nabla \phi(x)$ becomes $-(J^T J)^{-1} J^T r(x)$.

That is, Gauss Newton computes the step size by computing s where $J(x)s \approx r(x)$

5 Lecture 20

5.1 Quiz

Problem 1.

We use the approximation

$$x_k \leftarrow x_{k-1} + \alpha (H_f(x))^{-1} (-\nabla_f(x))$$

This works out to be

$$x_k \leftarrow [2, 1]^T + 1 \begin{bmatrix} 1 & -0.5 \\ 0 & 1 \end{bmatrix} (- \begin{bmatrix} \cos(x_1) \\ -\sin(x_2) \end{bmatrix})$$

Let us approximate $\cos(x_1) = \cos(2)$ by -0.41614 and let us approximate $-\sin(x_2) = -\sin(1)$ by -0.84147. This gives us

$$[2, 1]^T + \begin{bmatrix} 1 & -0.5 \\ 0 & 1 \end{bmatrix}^{-1} (- \begin{bmatrix} -0.41614 \\ -0.84147 \end{bmatrix})$$

See the code below

Problem 2.

Secant Updating Methods For unconstrained minimization of a function f:n→, why is it not a good idea to find a critical point by using Broyden's method to solve the nonlinear system f(x)=0? Select all that apply: It would require a matrix factorization at each iteration. It would not preserve symmetry of the approximate Hessian matrix. It would require evaluation of the Hessian matrix. It would not converge superlinearly.

Solution 5.1. First recall what Broyden's method is. Brodyen's method computes the solution to $\nabla_f(x) = 0$ using a series of approximations. We start with an initial approximation of the Jacobian B_k. Then we compute a step size that will update the current value of x_{k+1}. This gives us an updated value of the diference $f(x_{k+1}) - f(x_k) = y$. We then use y and x_{k+1}, x_k to update B_k with some update resembling a Sherman Morrison update. Note taht this update is not guaranteed to preserve symmetry of the Hessian matrix. Realize that here B_k corresponds to the Hessian. Secant methods have been shown to converge superlinearly, so the convergence would still be superlinear; at no point do we ever explicitly evalute the object that B mocks so there is no evaluation of a Hessian or anything of that sort.

Problem 3.

Gauss-Newton vs. Newton Which of the following statements about Gauss-Newton in comparison with 'regular' Newton applied to the 2-norm of the residual norm

$$\phi(x) := \frac{1}{2}\|f(x) - y\|_2^2$$

Gauss-Newton requires fewer known derivatives.
The approximation to H used by Gauss-Newton may be inaccurate if the residual is large.
Gauss-Newton has a lower cost per iteration.
The two are equivalent.
Gauss-Newton uses a more accurate approximation to f in computing the next iterate.
Both converge globally

Solution 5.2. Recall that Gauss newton approximates the Hessian as J^TJ and ignores the terms $r_i H_{r_i}(x)$;if the residual terms actually happen to be big, then our approximation is off. Note that since Gauss Newton does not use the Hessian explicitly, the costs of finding derviatives that are part of the Hessian are non-existent in Gauss-Newton (and so Guass newton has a lower costs per iteration).

Problem 5.

Constrained Optimization Consider the constrained optimization problem minf(x),g(x)=0. At a minimal point, f(x) must...

Solution 5.3.

$$\begin{aligned} f(x) - \lambda^T g(x) = 0 \\ \implies J(f(x)) - J(\lambda^T g(x)) = 0 \\ = J(f(x)) - \lambda^T J(g(x)) = 0 \\ \implies \nabla_f(x) = J(g(x))^T \lambda \end{aligned}$$

hence only the first choice is correct.

Definition 5.4. A constrained optimization problem is of the form:

$$\min f(x) \text{ such that } g(x) = 0$$

here $f : \mathbb{R}^n \to \mathbb{R}^1$ and $g : \mathbb{R}^n \to \mathbb{R}^m$.

Remark 5.5. Let the feasible set (ie those x such that $g(x) = 0$) be S. Given a point $x \in S$, we say that a direction s is feasible if for some $\alpha \in [0, r)$ where $r > 0$ that $x + \alpha * s \in S$ as well. A first order condition is that at the boundary of the feasible set, if s is a feasible direction then,

$$\nabla f(x^*)^T s \geq 0$$

or else, we would proceed in the direction of s and hence minimize f.

If we are at an interior point, then both s and $-s$ are feasible directions in which csae the optimality condition implies that $\nabla f(x^*) = 0$, which coincides with the first order optimality condition for unconstrained optimization. Indeed, constrained optimization is only interesting at the boundary of the feasible set, because constrained optmization problems behave like unconstrained optmization problems at the interior of the feasible set.

Remark 5.6. A second order optimality condition is that for any point x with a feasible direction s it holds that

$$s^T H_f(x^*) s \geq 0$$

It is easy to see why this is a necessary condition for the unconstrained case, for an approximation to $f(x^* + s)$ is given by

$$\nabla f(x^*) \cdot s + s^T H_f(x^*) s$$

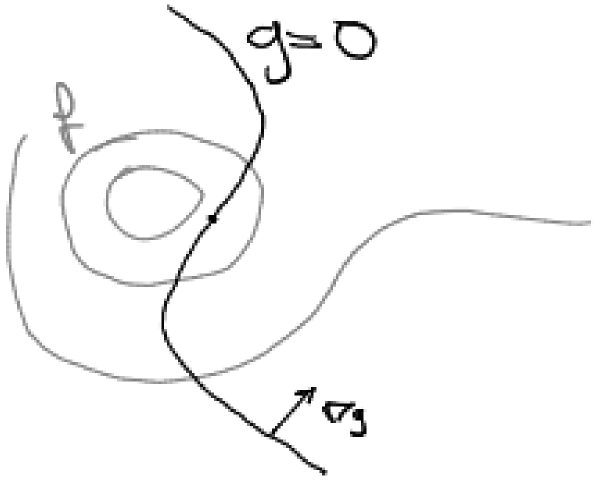
so that if $s^T H_f(x^*) s < 0$, then $f(x^* + s) < f(x^*)$, contradicting that $f(x^*)$ is optimal. It is not clear, however, why this is a necessary condition for a boundary point, for it may just take place that in the approximation above of $f(x + a)$, if $\nabla f(x^*) \cdot s > \left| s^T H_f(x^*) s \right|$, then $f(x^* + s) > f(x^*)$.

Proposition 5.7. At an optimal point x^*, it holds that

$$-\nabla f(x^*) = J_g^T \lambda^*$$

for some $\lambda^* \in \mathbb{R}^m$.

Proof. This is easiest to imagine:



Suppose that the dotted point is x^*. Now if we try to proceed in the direction of greatest descent, which is $-\nabla f(x^*)$, then we find that we must move in the direction of greatest ascent for g. That is, any further attempt to minimize f necessarily forces us to violate our constraint on g and, hence, fall off of the level set $g = 0$.

It is known that $\nabla f(x)^T$ is the direction of greatest ascent (recall that there is some calculus proof that explains that not only does $\nabla f(x) \cdot s$ where s is some n vector give us the instantaneous increase in the direction s but that were s in the direction of $\nabla f(x)$ itself, we would increase most rapidly (indeed use the cosine identity involving the dot product). One can similarly generalize (though, I am not sure how) and conclude that $(\nabla g)^T = J_g^T$ gives us the direction of greatest ascent in g.

We hit J_g^T by some m vector to give us an n vector. □

Definition 5.8. The lagrange function is defined to be

$$L(x, \lambda) f(x) + \lambda^T g(x)$$

since then $\nabla L = 0 \implies \nabla f(x) + J_g(x)^T \lambda = 0$ and $g(x) = 0$. That is

$$\nabla L = \begin{bmatrix} \nabla_x L(x, \lambda) \\ \nabla_\lambda L(x, \lambda) \end{bmatrix} = \begin{bmatrix} \nabla f(x) + \lambda^T g(x) \\ g(x) \end{bmatrix}$$

which are necessary conditions that we previously discussed.

Remark 5.9. If we wish to compute the hessian of this function, then simply do the following, take the jacobian with respect to x of the first row of ∇L and with respect to λ of first row; the resulting entries become the first row of the Hessian; likewise, obtain the second row of the Hessian. This gives us

$$H_L(x,\lambda) = \begin{bmatrix} B(x,\lambda) & J_g^T x \\ J_g(x) & 0 \end{bmatrix}$$

where $B(x,\lambda) = H_f(x) + \sum_{i=1}^m \lambda_i H_{g_i}(x)$.

Note that since a matrix is positive definite if and only if its eigenvalues are all positive, and the product of the diagonals of a matrix is the product of its eigenvalues, H_L is not positive definite (since it has 0 as an eigenvalue). This matrix is, however, symmetric (check this yourself).

Remark 5.10. As a consequence, Heath asserts that any critical point of the Lagrangian is necessarily a saddle point. That is, it must hold that at x^*, there exists s_1 such that $s_1^T H_f(x^*) s_1 > 0$ and s_2 such that $s_2^T H_f(x^*) s_2 < 0$. UNRESOLVED (Why is this true?) At present, the only lead I have is that if, suppose, that x^* corresponded to a minimum, it would have to hold that for all feasible directions s, there is some $\alpha \in [0,1]$ such that $f(x^* + \alpha s) \geq f(x)$.

This woudl imply using the Taylor remainder theorem that

$$f(x^* + s) = f(x^*) + \underbrace{\nabla f(x^*) \cdot s}_{0} + s^T H_f(x^* + \alpha' s) s \geq 0 \implies s^T H_f(x^* + \alpha' s) s \geq 0$$

where $\alpha' \in [0,1]$

It is not clear, however, that

$$s^T H_f(x^* + \alpha' s) s \geq 0 \implies s^T H_f(x^*) s > 0$$

for all s

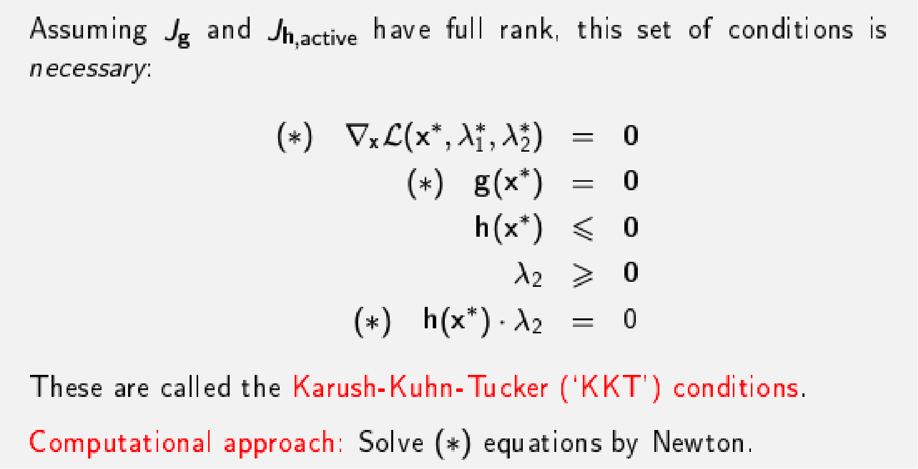
Remark 5.11. UNRESOLVED: if f is convex, then any critical point must be a global minimum (a statement made in Heath). This implies that convex functions have either minimums or maximums but not both?

Definition 5.12. When we introduce inequality constrains, ie functions $h_i : \mathbb{R}^n \to \mathbb{R}^m$ such that $h_i(x) \leq 0$, then we redefine the Lagragian to be

$$f(x) + \lambda_1^T(g(x)) + \lambda_2^T(h(x))$$

Remark 5.13. We must have $\lambda_{2,i} h_i(x) = 0$ for all i that correspond to indices of the inequality functions h_i. It is not known why this must hold.

We say that h_i is active at x^* if $h_i(x^*) = 0$. If h_i is active, then it can happen that in some direction s, for some small α we have $h_i(x^* + \alpha s) > 0$. This cannot happen, by contrast (assuming that h_i is sufficiently smooth, if $h_i(x^*) < 0$.



Assuming $J_{\mathbf{g}}$ and $J_{\mathbf{h},\text{active}}$ have full rank, this set of conditions is *necessary*:

$$\begin{array}{rcl} (*)\quad \nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x}^*,\lambda_1^*,\lambda_2^*) & = & \mathbf{0} \\ (*)\quad \mathbf{g}(\mathbf{x}^*) & = & \mathbf{0} \\ \mathbf{h}(\mathbf{x}^*) & \leqslant & \mathbf{0} \\ \lambda_2 & \geqslant & \mathbf{0} \\ (*)\quad \mathbf{h}(\mathbf{x}^*)\cdot\lambda_2 & = & 0 \end{array}$$

These are called the Karush-Kuhn-Tucker ('KKT') conditions.

Computational approach: Solve $(*)$ equations by Newton.

Assemble the problem's inequalities as well as the complentarity condition for h_i and the fact that $\lambda_2 \geq 0$ to obtain the KKT conditions listed above.