Gradient descent methods

Source: Ahmadi, ORF 363 slides.

Last time we learned how to optimize scalar functions

$$\min_x f(x) \qquad f : \mathbb{R} \to \mathbb{R}$$

In this lecture, we are going to solve **unconstrained convex** problem

$$\min_x f(x) \qquad f : \mathbb{R}^n \to \mathbb{R}$$

What do we know about convex functions?

- A stationary point (a point where the gradient vanishes) is a local minimum, which is automatically a global minimum

- How can we find a stationary point?

What do we know about convex functions?

- A stationary point (a point where the gradient vanishes) is a local minimum, which is automatically a global minimum
- How can we find a stationary point?
- We now begin to see some algorithms for this purpose, starting with gradient descent algorithms
- These are iterative algorithms: start at a point, jump to a new point that has a lower objective value and continue.

General form of the iterations

$$x_{k+1} = x_k + \alpha_k d_k$$

- $k = 1, 2, 3, \ldots$
- $x_k \in \mathbb{R}^n$: current point
- $x_{k+1} \in \mathbb{R}^n$: next point

General form of the iterations

$$x_{k+1} = x_k + \alpha_k d_k$$

- $k = 1, 2, 3, \ldots$
- $x_k \in \mathbb{R}^n$: current point
- $x_{k+1} \in \mathbb{R}^n$: next point
- $d_k \in \mathbb{R}^n$: direction to move along
- $\alpha_k \in \mathbb{R}$: step size

Goal: choose d_k and α_k so that $f(x_{k+1}) < f(x_k)$

Gradient methods

$$x_{k+1} = x_k + \alpha_k d_k$$

- The direction d_k to move along at step k is chosen based on ”information” from $\nabla f(x_k)$
- Why is $\nabla f(x_k)$ a natural vector to look at? Lemmas 1 and 2 below provide two reasons

Why $\nabla f(x)$?

Lemma 1. Consider yourself sitting at a point $x \in \mathbb{R}^n$ and looking at the value of the function f in all directions around you. The direction with the maximum rate of decrease is along $-\nabla f(x)$

Why $\nabla f(x)$?

Lemma 2. Consider a point $x \in \mathbb{R}^n$. For a direction d that satisfies

$$\nabla f(x)^T d < 0,$$

there exists a small $\alpha > 0$ such that $f(x + \alpha d) < f(x)$. In particular, we can choose $d = -\nabla f(x)$

Why $\nabla f(x)$?

Lemma 2. Consider a point $x \in \mathbb{R}^n$. For a direction d that satisfies

$$\nabla f(x)^T d < 0,$$

there exists a small $\alpha > 0$ such that $f(x + \alpha d) < f(x)$. In particular, we can choose $d = -\nabla f(x)$

Remark. The condition $\nabla f(x)^T d < 0$ means that the vectors $f(x)$ and d make an angle of more than 90 degrees, since $\nabla f(x)^T d = \|f(x)\|\|d\| \cos(\theta)$

General form of gradient descent

Lemma 3. Consider any positive definite matrix B. For any point with $\nabla f(x) \neq 0$, the direction $-B\nabla f(x)$ is a descent direction

General form of gradient descent

Lemma 3. Consider any positive definite matrix B. For any point with $\nabla f(x) \neq 0$, the direction $-B\nabla f(x)$ is a descent direction

This suggests that a general form of our descent algorithms:

$$x_{k+1} = x_k - \alpha_k B_k \nabla f(x_k) \quad B_k \succ 0$$

Common choices of descent direction

$$x_{k+1} = x_k - \alpha_k B_k \nabla f(x_k) \quad B_k \succ 0$$

- **Steepest descent**: $B_k = I$ for all k
  - Simplest descent direction but not always the fastest

Common choices of descent direction

$$x_{k+1} = x_k - \alpha_k B_k \nabla f(x_k) \quad B_k \succ 0$$

- **Steepest descent**: $B_k = I$ for all k
  - Simplest descent direction but not always the fastest
- **Newton Direction**: $B_k = (\nabla^2 f(x_k))^{-1}$ (assuming Hessian positive definite)
  - More expensive, but can have much faster convergence

Common choices of descent direction

$$x_{k+1} = x_k - \alpha_k B_k \nabla f(x_k) \quad B_k \succ 0$$

- **Diagonally Scaled Steepest Descent**: $B_k = \text{diag}(d_{1,k}, d_{2,k}, \ldots, d_{n,k})$ where all $d_{i,k} > 0$
  - For example, can take $d_{i,k} = \left(\frac{\partial^2 f(x_k)}{\partial x_i^2} \right)$ i.e., diagonally approximate Newton.

Common choices of descent direction

$$x_{k+1} = x_k - \alpha_k B_k \nabla f(x_k) \quad B_k \succ 0$$

- **Diagonally Scaled Steepest Descent**: $B_k = \text{diag}(d_{1,k}, d_{2,k}, \ldots, d_{n,k})$ where all $d_{i,k} > 0$
  - For example, can take $d_{i,k} = \left(\frac{\partial^2 f(x_k)}{\partial x_i^2}\right)$ i.e., diagonally approximate Newton.

- **Modified Newton Direction**: $B_k = (\nabla^2 f(x_0))^{-1}$
  - Compute Newton direction only at the beginning, or once every M steps

Common choices of the step size α_k

Back to the general form of our iterative algorithm

$$x_{k+1} = x_k + \alpha_k d_k$$

- **Constant step size**: $\alpha_k = s$ for all k ($s > 0$)
  - Simplest rule to implement, but may not converge if too large; may be too slow if too small

Common choices of the step size α_k

Back to the general form of our iterative algorithm

$$x_{k+1} = x_k + \alpha_k d_k$$

- **Constant step size**: $\alpha_k = s$ for all k ($s > 0$)
  - Simplest rule to implement, but may not converge if too large; may be too slow if too small

- **Diminishing step size**: $\alpha_k \to 0, \sum_{k=1}^{\infty} \frac{1}{\alpha_k} = \infty$ (e.g. $\alpha_k = \frac{1}{k}$)
  - Descent not guaranteed at each step; only later when becomes small
  - $\sum_{k=1}^{\infty} \frac{1}{\alpha_k}$ imposed to guarantee progress does not become too slow

Common choices of the step size α_k

$$x_{k+1} = x_k + \alpha_k d_k$$

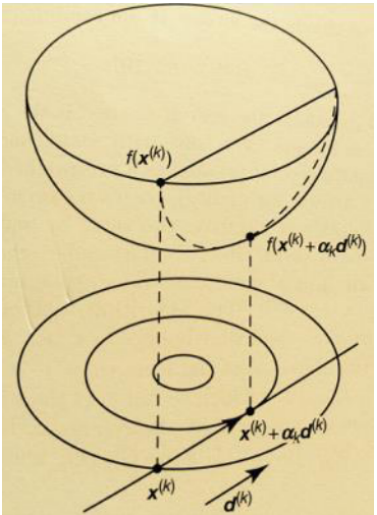
- **Exact line search**: $\alpha_k = \operatorname{argmin}_\alpha f(x_k + \alpha d_k)$
  - A minimization problem itself, but an easier one (one dimensional)
  - If f is convex, the minimization problem is also convex (why?)
  - Can use methods that we learned in the previous lecture

Common choices of the step size α_k

$$x_{k+1} = x_k + \alpha_k d_k$$

- **Exact line search**: $\alpha_k = \operatorname{argmin}_\alpha f(x_k + \alpha d_k)$
  - A minimization problem itself, but an easier one (one dimensional)
  - If f is convex, the minimization problem is also convex (why?)
  - Can use methods that we learned in the previous lecture
- **Limited exact line search**: $\alpha_k = \operatorname{argmin}_{\alpha \in [0,s]} f(x_k + \alpha d_k)$
  - Same as above, but tries not to step to far

Illustration of exact line search



Stopping criteria

- Once we have a rule for choosing the search direction and the step size, we are good to go for running the algorithm.
- Typically the initial point is picked randomly, or if we have a guess for the location of local minima, we pick close to them
- But when to stop the algorithm?

Stopping criteria

Some common choices ($\epsilon > 0$ is a small prescribed threshold):

- $\|\nabla f(x_{k+1})\| < \epsilon$
    - This means that we have found a point that is close to a stationary point ($\nabla f(x) = 0$)

Stopping criteria

Some common choices ($\epsilon > 0$ is a small prescribed threshold):

- $\|\nabla f(x_{k+1})\| < \epsilon$
    - This means that we have found a point that is close to a stationary point ($\nabla f(x) = 0$)
- $|f(x_{k+1}) - f(x_k)| < \epsilon$
    - Improvement in function has become small

Stopping criteria

Some common choices ($\epsilon > 0$ is a small prescribed threshold):

- $\|\nabla f(x_{k+1})\| < \epsilon$
  - This means that we have found a point that is close to a stationary point ($\nabla f(x) = 0$)
- $|f(x_{k+1}) - f(x_k)| < \epsilon$
  - Improvement in function has become small
- $\|x_{k+1} - x_k\| < \epsilon$
  - Movement between iterations has become small

Stopping criteria

Some common choices ($\epsilon > 0$ is a small prescribed threshold):

- $\dfrac{|f(x_{k+1}) - f(x_k)|}{\max\{1, |f(x_k)|\}} < \epsilon$
  - A ”relative” measure – removes dependence on the scale of f
  - The $\max$ is taken to avoid dividing by small numbers

Stopping criteria

Some common choices ($\epsilon > 0$ is a small prescribed threshold):

- $\dfrac{|f(x_{k+1}) - f(x_k)|}{\max\{1, |f(x_k)|\}} < \epsilon$
  - A "relative" measure – removes dependence on the scale of f
  - The $\max$ is taken to avoid dividing by small numbers

- $\dfrac{\|x_{k+1} - x_k\|}{\max\{1, \|x_k\|\}} < \epsilon$
  - Same as above - removes dependence on the scale of x_k

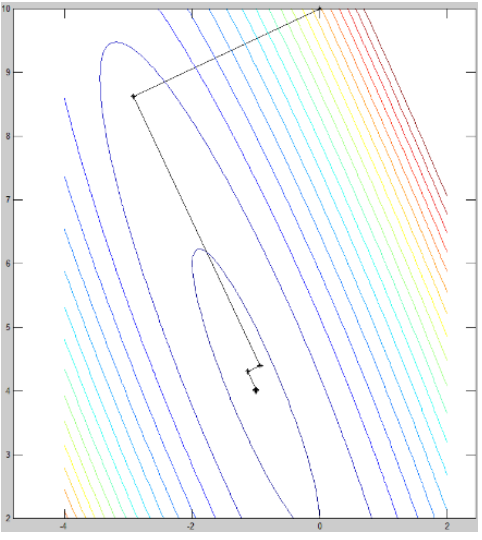
Example

$$\min_x f(x) = 5x_1^2 + x_2^2 + 4x_1x_2 - 6x_1 - 4x_2 + 15$$

This is a convex function – Any stationary point must be the unique global minimizer

Let's try the steepest descent method:
$d_k = -\nabla f(x_k)$
α_k: get from exact line search

$x_0 = (0, 10)^T$
Stopping criterion: $\|\nabla f(x)\| < 10^{-6}$



K	x_k		$\nabla f(x_k)$		$\|\nabla f(x_k)\|$	α_k	$f(x_k)$
1.000000000000000	0	10.000000000000000	34.000000000000000	16.000000000000000	37.576888456111871	0.085971748660497	75.000000000000000
2.000000000000000	-2.923039454456893	8.624452021432051	-0.732586458840725	1.556746225036530	1.720506242023632	2.715384615384604	14.303945445689237
3.000000000000000	-0.933785454681702	4.397287271909798	2.251294540822171	1.059432725092788	2.488116719234297	0.085971748660497	10.284983790761091
4.000000000000000	-1.127333183106014	4.306205987945416	-0.048507879278480	0.103079243666774	0.113922585532891	2.715384615384754	10.018870721228331
5.000000000000000	-0.995615633988288	4.026306196070238	0.149068444398068	0.070149856187323	0.164749517262910	0.085971748660497	10.001249473246101
6.000000000000000	-1.008431308823290	4.020275290265531	-0.003211927170778	0.006825345237901	0.007543329090456	2.715384615384132	10.000082733302879
7.000000000000000	-0.999709691198026	4.001741852811849	0.009870499267134	0.004644940831593	0.010908814376984	0.085971748660486	10.000005478148033
8.000000000000000	-1.000558275280174	4.001342519126133	-0.000212676297206	0.000451937131571	0.000499478105912	2.715384615432194	10.000000362733084
9.000000000000000	-0.999980777334673	4.000115335991923	0.000655570620962	0.000307562645155	0.000722322183848	0.085971748660553	10.000000024018206
10.000000000000000	-1.000036965943830	4.000088894293497	-0.000014082264316	0.000029924811672	0.000033077215672	2.715384615356879	10.000000001590355
11.000000000000000	-0.999998727179956	4.000007636920265	0.000043275881499	0.000020365120705	0.000047828234975	0.085971748657113	10.000000000105302
12.000000000000000	-1.000002447683163	4.000005886095226	-0.000000932450728	0.000001981457800	0.000002189894831	-0.692156863782812	10.000000000006972
13.000000000000000	-1.000003093085335	4.000007257574842	-0.000001900553979	0.000002142808345	0.000002864215954	0.644738427673033	10.000000000010715
14.000000000000000	-1.000001867725151	4.000005876023959	0.000004826844329	0.000004281147315	0.000006451871706	0.095703090047471	10.000000000008070
15.000000000000000	-1.000002329669068	4.000005466304932	-0.000001431470952	0.000001613933592	0.000002157287817	0.644738430768048	10.000000000006079
16.000000000000000	-1.000001406744733	4.000004425739920	0.000003635512357	0.000003224500911	0.000004859460487	0.095703089988433	10.000000000004576
17.000000000000000	-1.000001754674499	4.000004117145219	-0.000000107816411	0.000001215592444	0.000001624839327	0.644738428822084	10.000000000003446
18.000000000000000	-1.000001059540664	4.000003333406057	0.000002738217589	0.000002428649458	0.000003660078381	0.095703090024235	10.000000000002597
19.000000000000000	-1.000001321596548	4.000003100976799	-0.000000812058286	0.000000915567405	0.000001223806493	-0.251801315258460	10.000000000001959
20.000000000000000	-1.000001526073893	4.000003331517876	-0.000001934667424	0.000000558740181	0.000002013734995	0.140893244247662	10.000000000002405
21.000000000000000	-1.000001253492323	4.000003252795159	0.000000476257405	0.000001491621026	0.000001565807907	0.241956665294677	10.000000000002125
22.000000000000000	-1.000001368725976	4.000002891887509	-0.000002119709727	0.000000308871113	0.000002142094930	0.115598723360049	10.000000000001897
23.000000000000000	-1.000001123690238	4.000002856182403	0.000000187827231	0.000001217603853	0.000001232005768	0.511898030938667	10.000000000001631
24.000000000000000	-1.000001219838628	4.000002232893388	-0.000003266812726	-0.000000413567736	0.000003292886828	0.092258931075849	10.000000000001528
25.000000000000000	-1.000000918445978	4.000002271048705	-0.000000100264955	0.000000868313499	0.000000874083174	0.000000000100000	10.000000000001036

Convergence

We say that the algorithm converges if x_k approaches a single point x^*

Convergence

We say that the algorithm converges if x_k approaches a single point x^*

Theorem. Consider the sequence generated by any descent algorithm with $d_k = -B_k \nabla f(x_k)$ such that eigenvalues of B_k are larger than some m for all k and the step size is chosen according to the exact line search, or the limited exact line search. Then, the algorithm converges to a stationary point

Rates of convergence

- Once we know an iterative algorithm converges, the next question is how fast?
- For example, if $|f(x_k) - f(x^*)| \approx \frac{1}{\log(\log(k))}$ then sure, this difference will eventually go to zero, but it will take years to go even below 0.1,

Rates of convergence

Definition. Let $\{x_k\}$ converge to x^*. We say the convergence is of **order** $p(\geq 1)$ and with **factor** $\gamma(> 0)$, if for large enough k,

$$\|x_{k+1} - x^*\| \leq \gamma \|x_k - x^*\|^p$$

Rates of convergence

Definition. Let $\{x_k\}$ converge to x^*. We say the convergence is of **order** $p(\geq 1)$ and with **factor** $\gamma(> 0)$, if for large enough k,

$$\|x_{k+1} - x^*\| \leq \gamma \|x_k - x^*\|^p$$

- Larger $p \Rightarrow$ faster convergence

Rates of convergence

Definition. Let $\{x_k\}$ converge to x^*. We say the convergence is of **order** $p(\geq 1)$ and with **factor** $\gamma(> 0)$, if for large enough k,

$$\|x_{k+1} - x^*\| \leq \gamma \|x_k - x^*\|^p$$

- Larger $p \Rightarrow$ faster convergence
- For the same p, smaller $\gamma \Rightarrow$ faster convergence

Rates of convergence

Definition. Let $\{x_k\}$ converge to x^*. We say the convergence is of **order** $p(\geq 1)$ and with **factor** $\gamma(> 0)$, if for large enough k,

$$\|x_{k+1} - x^*\| \leq \gamma \|x_k - x^*\|^p$$

- Larger $p \Rightarrow$ faster convergence
- For the same p, smaller $\gamma \Rightarrow$ faster convergence
- If $\{x_k\}$ converges with order p, it also converges with any order $p' \leq p$

Rates of convergence

Definition. Let $\{x_k\}$ converge to x^*. We say the convergence is of **order** $p(\geq 1)$ and with **factor** $\gamma(> 0)$, if for large enough k,

$$\|x_{k+1} - x^*\| \leq \gamma \|x_k - x^*\|^p$$

- Larger $p \Rightarrow$ faster convergence
- For the same p, smaller $\gamma \Rightarrow$ faster convergence
- If $\{x_k\}$ converges with order p, it also converges with any order $p' \leq p$
- If $\{x_k\}$ converges with factor γ, it also converges with any factor $\gamma' \geq \gamma$

Rates of convergence

Definition. Let $\{x_k\}$ converge to x^*. We say the convergence is of **order** $p(\geq 1)$ and with **factor** $\gamma(> 0)$, if for large enough k,

$$\|x_{k+1} - x^*\| \leq \gamma \|x_k - x^*\|^p$$

- Larger $p \Rightarrow$ faster convergence
- For the same p, smaller $\gamma \Rightarrow$ faster convergence
- If $\{x_k\}$ converges with order p, it also converges with any order $p' \leq p$
- If $\{x_k\}$ converges with factor γ, it also converges with any factor $\gamma' \geq \gamma$
- So we typically look for the largest p and the smallest γ for which the inequality holds

Rates of convergence

Definition. Let $\{x_k\}$ converge to x^*. We say the convergence is of **order** $p(\geq 1)$ and with **factor** $\gamma(> 0)$, if for large enough k,

$$\|x_{k+1} - x^*\| \leq \gamma \|x_k - x^*\|^p$$

Some more terminology:

- **Linear convergence**: $p = 1$ and $\gamma < 1$
- **Sublinear convergence**: $p = 1$ and $\gamma = 1$
- **Superlinear convergence**: $p > 1$
- **Quadratic convergence**: $p = 2$

Rates of convergence

Linear convergence: $p = 1$ and $\gamma < 1$

Why is it called linear convergence?

For k large enough, we have

$$\|x_{k+\ell} - x^*\| \leq \gamma \|x_{k+\ell-1} - x^*\| \leq \gamma^2 \|x_{k+\ell-2} - x^*\| \leq \dots$$

Rates of convergence

Linear convergence: $p = 1$ and $\gamma < 1$

Why is it called linear convergence?

For k large enough, we have

$$\|x_{k+\ell} - x^*\| \leq \gamma \|x_{k+\ell-1} - x^*\| \leq \gamma^2 \|x_{k+\ell-2} - x^*\| \leq \ldots \leq \gamma^\ell \|x_k - x^*\|$$

Rates of convergence

Linear convergence: $p = 1$ and $\gamma < 1$

Why is it called linear convergence?

For k large enough, we have

$$\|x_{k+\ell} - x^*\| \leq \gamma \|x_{k+\ell-1} - x^*\| \leq \gamma^2 \|x_{k+\ell-2} - x^*\| \leq \ldots \leq \gamma^\ell \|x_k - x^*\|$$

Taking $\log$ on both sides,

$$\log\|x_{k+\ell} - x^*\| \leq \log\left[\gamma^\ell \|x_k - x^*\|\right] = \ell \log \gamma + \log\|x_k - x^*\|$$

$-\log\|x_{k+\ell} - x^*\|$, which measures the number of correct significant digits in $x_{k+\ell}$, grows linearly with ℓ

Examples

- $\|x_k - x^*\| \approx a^k, 0 < a < 1$ linear convergence

Examples

- $\|x_k - x^*\| \approx a^k, 0 < a < 1$ linear convergence
- $\|x_k - x^*\| \approx a^{2^k}$ quadratic convergence
- Quadratic convergence is super fast! Number of correct significant digit doubles in each iteration

Examples

- $\|x_k - x^*\| \approx a^k, 0 < a < 1$ linear convergence
- $\|x_k - x^*\| \approx a^{2^k}$ quadratic convergence
- Quadratic convergence is super fast! Number of correct significant digit doubles in each iteration
- $\|x_k - x^*\| \approx \frac{1}{k}$ sublinear convergence

Convergence rate of steepest descent for quadratic functions

Theorem. Consider a quadratic function

$$f(x) = \frac{1}{2}x^T Q x + b^T x + c \qquad Q \succ 0$$

Let m and M be the smallest and largest eigenvalue of Q. Then the sequence $\{f(x_k)\}$ generated by the steepest descent algorithm with exact line search converges to the unique global minimum of f, where the convergence is **linear** ($p = 1$), and with factor $\gamma = \left(\frac{M-m}{M+m}\right)^2$

Remarks

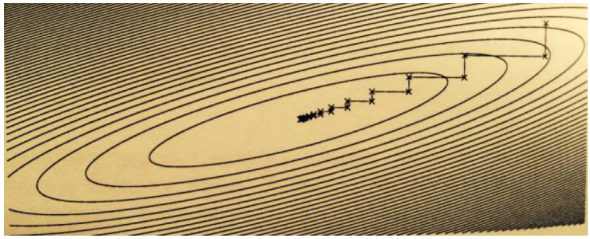
- $\kappa = \frac{M}{m}$ is called **condition number** of the matrix Q
  - Appears often in numeric analysis
- $\left(\frac{M-m}{M+m}\right)^2 = \left(\frac{\kappa-1}{\kappa+1}\right)^2$
- We want κ close to 1 for fast convergence

Remarks

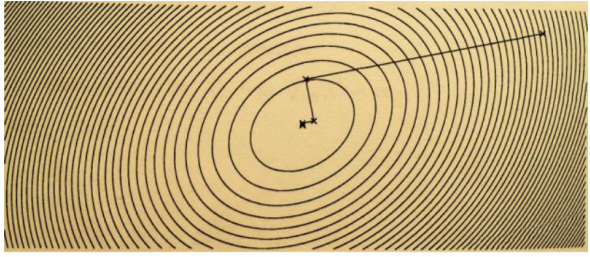
- $\kappa = \frac{M}{m}$ is called **condition number** of the matrix Q
  - Appears often in numeric analysis
- $\left(\frac{M-m}{M+m}\right)^2 = \left(\frac{\kappa-1}{\kappa+1}\right)^2$
- We want κ close to 1 for fast convergence

κ	$\left(\frac{M-m}{M+m}\right)^2$	k; $\|x_k - x^*\| < 0.1$
1.1	0.002	1
3	0.25	2
10	0.67	6
100	0.96	58
200	0.98	116
400	0.99	231

Large κ



Small κ



Beyond quadratic case

What if the function f we are minimizing is not quadratic?

- Let x^* be the optimal solution
- Using the Taylor series, f can be approximated by a quadratic:

$$\frac{1}{2}x^T\nabla^2 f(x^*)x \quad \text{(plus linear and constant terms)}$$

- Hence, the condition number κ of the Hessian $\nabla^2 f(x^*)$ dictates convergence rate