Week 2: Friday, Sep 6

Order notation

We'll use order notation in multiple ways this semester, so we briefly review it here. This should be familiar to many of you.

- We say $f(n) = O(g(n))$ (read "$f(n)$ is big-O of $g(n)$") if there is some C and N such that

  $$|f(n)| \leq Cg(n), \quad \forall n \geq N.$$

  For example, multiplying two $n \times n$ matrices costs $2n^3 = O(n^3)$ flops.

- We say $f(n) = \Theta(g(n))$ if $f(n) = O(g(n))$ and $g(n) = O(f(n))$. Informally, we sometimes write $f(n) = O(g(n))$ and mean $\Theta(g(n))$. For example, the cost of multiplying a matrix by a vector is $2n^2 = O(n^2)$ flops; while it would also be technically correct to say that it's $O(n^3)$ flops, you'd get strange looks for saying it at a cocktail party[1].

- We say $f(n) = o(g(n))$ if for every $C > 0$, there is an N such that $f(n) \leq Cg(n)$ for all $n \geq N$. This is almost the same as saying

  $$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0,$$

  except that we might[2] want to let $g(n)$ to be zero infinitely often.

- We say $f(\epsilon) = O(g(\epsilon))$ if there is some C and δ such that

  $$|f(\epsilon)| \leq Cg(\epsilon), \quad \forall |\epsilon| \leq \delta.$$

  For example, if f is a function with two continuous derivatives in a neighborhood of zero, then Taylor's theorem gives

  $$f(\epsilon) = f(0) + f'(0)\epsilon + O(\epsilon^2).$$

[1] You may object that you'd get strange looks for using order notation in any form in a cocktail party conversation, and with some justification. So perhaps let's add the qualifying phrase "at a technical conference" to the end of the sentence. Alternately, you may want to avoid inviting me to cocktail parties – I won't take offense.

[2] Very rarely.

> We rely on context to distinguish between using big-O notation to describe asymptotics as $n \to \infty$ or asymptotics as $\epsilon \to 0$. We can similarly use little-o or big-Theta notation to describe asymptotic behavior of functions of ϵ as $\epsilon \to 0$

In many cases in this class, we work with problems that have more than one size parameter; for example, in a factorization of an $m \times n$ matrix, both m and n are size parameters. In some cases, these parameters are constrained with respect to each other; for example, if I talk about a "tall skinny" matrix, I am assuming that $n \leq m$. In other cases, there is no such constraint. This point seems worth mentioning explicitly because on the second problem of your homework, there are two independent size parameters (n and k), and I've asked for a complexity of $O(n)$ – try to get something that really does take $O(n)$ time, and not $O(nk)$ or $O(n+k)$!

The power of log-log plots

Often, an algorithm with a size parameter n will have running time $\Theta(n^p)$, or an approximation depending on some parameter h will have error $\Theta(h^p)$. If you've determined that this is the case with pen and paper, and you want a sanity check, the simplest approach I know is a log-log plot. For example, if $f(n) = \Theta(n^p)$, then for large enough n we should have $f(n) \approx Cn^p$, and so

$$\log(f(n)) \approx p \log n + C.$$

If we make a log-log plot of n versus $f(n)$, we should see a straight line with slope p, and with the vertical offset depending on C.

Calculus with matrices

Calculus with vector-valued and matrix-valued functions is quite useful in perturbation theory. The basic rules are the same as those you learned in Calculus I, save only that matrix multiplication is not generally commutative. So if $A : \mathbb{R} \to \mathbb{R}^{m \times n}$ and $B : \mathbb{R} \to \mathbb{R}^{n \times p}$ are differentiable matrix-valued

functions, we have

$$\begin{aligned}(AB)' &= A'B + AB' \\ (A^k)' &= \sum_{j=1}^{k} A^{j-1} A' A^{k-j}, \quad k \in \mathbb{Z}^+ \\ (A^{-1})' &= -A^{-1}A'A^{-1}.\end{aligned}$$

I sometimes ask for the proofs of questions like this on homework assignments. Usually, these proofs can be done either in terms of matrices or in terms of matrix elements. As a toy example, to show that $C' = A'B + AB'$ for $C = AB$, I could either argue

$$\left(\sum_k a_{ik} b_{kj}\right)' = \sum_k \left(a'_{ik} b_{kj} + a_{ik} b'_{kj}\right) = \left(\sum_k a'_{ik} b_{kj}\right) + \left(\sum_k a_{ik} b'_{kj}\right)$$

or write $A(t) = A_0 + \epsilon A'_0 + O(\epsilon)^2$ and $B = B_0 + \epsilon B'_0 + O(\epsilon)^2$, and then observe

$$\begin{aligned}(AB)' &= \left(A_0 + \epsilon A'_0 + O(\epsilon^2)\right)\left(B_0 + \epsilon B'_0 + O(\epsilon^2)\right) \\ &= A_0 B_0 + \epsilon(A'_0 B_0 + A_0 B'_0) + O(\epsilon^2) \\ &= C_0 + \epsilon C'_0 + O(\epsilon^2)\end{aligned}$$

I typically prefer the latter approach. To see why, consider the formula for $(A^{-1})'$. Note that $(AA^{-1}) = I$, so $(AA^{-1})' = 0$; implicit differentiation gives

$$(AA^{-1})' = A'A^{-1} + A(A^{-1})' = 0,$$

and multiplication from the left by A^{-1} completes the proof.

Variational notation

I will frequently write matrix calculus in terms of variational notation. Here δA should be read to mean a first-order change to a matrix A; for example,

$$\delta(A^{-1}) = -A^{-1}(\delta A)A^{-1}$$

If you like, this is really a shorthand for writing directional derivatives of matrix expressions, i.e.

$$\delta(A^{-1}) \equiv \left.\frac{d}{dt}\right|_{t=0} (A + t\delta A)^{-1}.$$

Sometimes, it makes sense to restrict our attention to a set of *admissible* variations. For example, suppose at a point x we wanted to look at all the directions that are consistent (at first order) with the constraint $\|x\|_2^2 = 1$. Note that $\|x\|^2 = x^T x$ is differentiable, and

$$\delta(x^T x - 1) = (\delta x)^T x + x^T(\delta x) = 2x^T(\delta x);$$

i.e., the directions consistent with the constraint satisfy $x^T \delta x = 0$.

Lagrange multipliers

While we're discussing calculus, let's briefly recall the method of Lagrange multipliers for constrained optimization problems. If $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^m$ are sufficiently smooth functions, then any local minimizer of f subject to the constraint $g(x) = 0$ must be a critical point of

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

For x_* to be a critical point of L means that

$$0 = \delta L = (f'(x_*) + \lambda^T g'(x_*))\delta x + \delta\lambda^T g(x_*)$$

for any variations δx and $\delta\lambda$. Note that $f'(x_*)$ here means a *row* vector; this is needed for the expression $f'(x_*)\delta x$ to make dimensional sense. This gives the equations

$$\begin{aligned} f'(x_*) + \lambda^T g'(x_*) &= 0 \\ g(x_*) &= 0, \end{aligned}$$

The directions consistent with the constraint $g(x) = 0$ at x_* satisfy $g'(x_*)\delta x = 0$, so the first equation says that

$$f'(x_*)\delta x = -\lambda^T g'(x_*)\delta x = 0$$

in any direction consistent with the constraint, i.e. there is no (constrained) first-order descent direction at x_*. The second stationary equation just says that the constraint is satisfied at x_*.

The operator two-norm

Last time, we gave some background on vector norms, consistent matrix norms, and operator (or induced) norms. We gave formulas for the operator 1-norm and operator ∞-norm for matrices; they are respectively the maximum absolute column and row sums. I then promised we would describe the operator 2-norm and the singular value decomposition in the next lecture; let me make good on that promise now!

Suppose $A \in \mathbb{R}^{m \times n}$. By the definition of the operator norm,

$$\|A\|_2 \equiv \max_{\|v\|_2=1} \|Av\|_2,$$

or, equivalently,

$$\|A\|_2^2 = \max_{\|v\|_2^2=1} \|Av\|_2^2.$$

This is a constrained optimization problem, so we apply the method of Lagrange multipliers. Define the augmented Lagrangian L

$$L(v, \lambda) = \|Av\|^2 - \lambda(\|v\|^2 - 1);$$

then note that L is differentiable, and

$$\delta L = 2(v^T A^T A - \lambda v^T)\delta v - \delta\lambda(\|v\|^2 - 1),$$

and so the critical point equations are

$$\begin{aligned} A^T A v_* &= \lambda v_* \\ \|v_*\|^2 &= 1. \end{aligned}$$

This critical point equation is a real symmetric eigenvalue problem!

We know from past linear algebra classes (or from reading the book) that a real symmetric $n \times n$ matrix has are n real eigenvalues (counting multiplicity) and a system of n orthonormal eigenvectors. If we pre-multiply the first equation by v_*^T, note that we have

$$\lambda = v_*^T A^T A v_* = \|Av_*\|_2^2.$$

Thus, the largest eigenvalue $\lambda_{\max} = \sigma_{\max}^2$ of $A^T A$ is equal to $\|A\|_2^2$. We say $\sigma_{\max}$ is the largest *singular value* of A, and v_* is a corresponding *singular vector*. Note that $u_* = Av_*/\sigma_{\max}$ is also a unit-length vector (why?).

The SVD

More generally, we write the eigenvalues of A in descending order as $\sigma_1^2 \geq \sigma_2^2 \geq \ldots \geq \sigma_n^2$, and accumulate the corresponding eigenvectors $v_1, \ldots, v_n$ into a matrix V. Then

$$AV = U\Sigma, \quad \Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$$

where the columns of U are unit length; one can also show that they are orthogonal. Since V is an orthogonal matrix, we have that $V^{-1} = V^T$, and so we have the *singular value decomposition* (SVD)

$$A = U\Sigma V^T.$$

If $m > n$, we have actually defined the "economy" SVD; in the full SVD, the matrix U is square and Σ is a rectangular diagonal matrix.

The SVD has several nice properties; if $A = U\Sigma V^T$, then

1. $\|A\|_2 = \sigma_{\max}(A)$
2. $\|A\|_F^2 = \sum_{i=1}^n \sigma_i(A)^2$
3. If A is square and invertible, then $A^{-1} = V\Sigma^{-1}U^T$
4. Thus $\|A^{-1}\|_2 = \sigma_{\min}(A)^{-1}$ if A invertible
5. The rank of A is the number of nonzero singular values
6. The best rank-k approximation to A (two-norm or Frobenius) is $\sum_{i=1}^k \sigma_i u_i v_i^T$

See Section 2.4 in the book. We will be returning to the properties of the SVD periodically throughout the course.