IAM Data Science Pre-Workshop
Linear Algebra lecture #2

Many optimization algorithms have linear system solves as a key ingredient.

(1) Optimization Basics.

Many continuous optimization problems can be put into the following form:

(A) optimize (max or min) $f(\underline{x})$ $\underline{x} \in \mathbb{R}^n$

↑ given scalar function.

That is, find $\underline{x}^*$ so that

$f(\underline{x}^*) < f(\underline{x})$ for all $\underline{x} \neq \underline{x}^*$

(at least locally).

(B) possibly subject to $\underline{g}(\underline{x}) = \underline{0}$

$\underline{g} \in \mathbb{R}^m$ $(m < n)$. (equality constraints).

(C) and also possibly subject to

$\underline{h}(\underline{x}) \geq_c \underline{0}$, $\underline{h} \in \mathbb{R}^l$. (inequality constraints)

↑ component-wise

As a reminder, you have seen all of this in Math 200:

★

for A look for critical points $\underline{x}^*$, $\nabla f(\underline{x}^*) = \underline{0}$.
If the eigenvalues of the Hessian of f at $\underline{x}^*$ are all positive, then $\underline{x}^*$ is a local minimum.

for B Lagrange multipliers.

for C the inequality constraints specify a domain, and you have to check all the boundary points - with Lagrange multipliers if necessary.

We'll forget B & C for the rest of this lecture - no constraints!

Notes: + In applications, you may not need to find the minimum - if you can decrease f by just a little bit, maybe that is enough to make millions of dollars.

+ If f, g and h are affine, it is a linear programming problem.

+ If f is quadratic and g & h are affine, it is a quadratic programming problem.

+ If f is a specific type of quadratic with no constraints, A, $\underline{b}$ given

$$f(\underline{x}) = \|A\underline{x} - \underline{b}\|^2$$

A $m \times n$, $m > n$, A full rank (n), then it is a linear least squares problem.

② Linear Least Squares.

Find $\underline{x}$ that minimizes $\|A\underline{x}-\underline{b}\|^2$

Notation $(A\underline{x})_i = \sum_{j=1}^{n} a_{ij} x_j$.

$$f(\underline{x}) = \|A\underline{x} - \underline{b}\|^2 = \sum_{i=1}^{m}\left[\left(\sum_{j=1}^{n} a_{ij} x_j\right) - b_i\right]^2$$

To find the minimum,

$$\frac{\partial f}{\partial x_l} = 0 \qquad l = 1, \dots n.$$

only nonzero when $j=l$

$$\Downarrow$$

$$2 \sum_{i=1}^{m}\left[\sum_{j=1}^{n} a_{ij} x_j - b_i\right] \underbrace{\sum_{j=1}^{n} a_{ij} \frac{\partial x_j}{\partial x_l}}_{a_{il}}$$

$$\underbrace{\sum_{i=1}^{m} a_{il} \underbrace{\sum_{j=1}^{n} a_{ij} x_j}_{(A\underline{x})_i}}_{(A^T A \underline{x})_l} - \underbrace{\sum_{i=1}^{m} a_{il} b_i}_{(A^T \underline{b})_l} = 0.$$

A^T: $n \times m$, $\underline{b}$: $m \times 1$

A^T: $n\times m$, A: $m\times n$, $\underline{x}$: $n \times 1$

when considering matrix equations, it is good to check sizes.

so $A^T A \underline{x} = A^T \underline{b}$. (1)

<u>Note</u>: If $m>n$, A is $m\times n$, A has rank n, the $A^T A$ is $n\times n$ and has rank n, so (1) is

solvable.

+ often the least squares problem is not solved in the form (1), but rather using the Singular Value decomposition (which we will discuss next lecture). This process is better conditioned for iterative approximation.

+ $\mathbb{A}^T\mathbb{A}$ is symmetric and positive definite (all positive eigenvalues – next lecture). There are good solver options for this class of matrix (like CG).

3 Multiple linear regression.

Note: some confusing change of notation coming up.

Suppose that a process can be described by

not the F from before.

$$F(\underline{x}) = a_1 x_1 + \cdots a_n x_n = \underline{a} \cdot \underline{x}$$

$\in \mathbb{R}^n$

Here, the a's are the unknowns.

m measurements are taken at conditions $\underline{x}_i$

$$f(\underline{x}_i) = \underline{a} \cdot \underline{x}_i = \sum_{j=1}^{n} x_{ij} a_j = (\mathbb{A}\underline{x})_i \approx y_i$$

row vector.

Pick the coefficients $\underline{a}$ so that the error to the measurements is minimized

That is, minimize

$$\| \mathbb{X} \underline{a} - \underline{y} \|^2$$

can be written $\| a^T \mathbb{X}^T - y^T \|^2$

It is a least squares problem!

$$\mathbb{X}^T \mathbb{X} \underline{a} = \mathbb{X}^T \underline{y}.$$

Ex linear regression.

$$f(x) = ax + b.$$

a & b to be determined

m data $\quad y_i \approx a x_i + b.$

given.

"train" the model on the data. Useful if the form of the model is appropriate.

$$\mathbb{X} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \\ x_m & 1 \end{bmatrix} \qquad \underline{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}.$$

$$\mathbb{X}^T \mathbb{X} = \begin{bmatrix} \sum x_i^2 & \sum x_i \\ \sum x_i & m \end{bmatrix}. \qquad \mathbb{X}^T \underline{y} = \begin{bmatrix} \sum x_i y_i \\ \sum y_i \end{bmatrix}.$$

→ computational examples.

4 Vector Newton's Method

Let's go back to ⊛ $\nabla f(\underline{x}^*) = \underline{0}$.

In general, this is n nonlinear equations in n unknowns

Let us consider the more general case $\underline{g}(\underline{x}^*) = \underline{0}$

aside: Scalar Newton's method on the board.

The Jacobian matrix of derivatives is

$$\mathbb{J}; \quad d_{ij} = \frac{\partial g_i}{\partial x_j}.$$

Theory: If $\mathbb{J}$ has full rank at zero of $\underline{g}$, the zero is locally unique.

at an arbitrary point $\underline{x}$, we can do a linear approximation:

$$g_i(\underline{x}^*) \approx g_i(\underline{x}) + \sum_{j=1}^{n} \frac{\partial g_i}{\partial x_j}(x_j^* - x_j).$$

in vector notation

$$\underline{g}(\underline{x}^*) \approx \underline{g}(\underline{x}) + \mathbb{J}(\underline{x})(\underline{x}^* - \underline{x}).$$

So if $\underline{x}$ were given and $\underline{g}(\underline{x})$ were not zero, we could use

$$\underline{g}(\underline{x}) + \mathbb{J}(\underline{x}^* - \underline{x}) \approx \underline{0}$$

and solve for $\underline{x}^*$ to approximate the root.

$$\underline{x}^* = \underline{x} - \mathbb{J}^{-1}(\underline{x})\, \underline{g}(\underline{x}).$$

Remember, we never find $\mathbb{J}^{-1}$ and multiply by it. This is notation for $\underline{\delta}$ that solves

$$\mathbb{J}\,\underline{\delta} = \underline{g}(\underline{x}).$$

Ex Nonlinear boundary value problem.

Consider a FD approx of

$$-u'' + u - u^3 = f(x).$$

$$-\mathbb{D}_2 \underline{U} + \underline{U} - \underline{U}^{\wedge 3} = \underline{F}.$$

$\uparrow$ pointwise cube.

So $\mathcal{G}(\underline{U}) = -\mathbb{D}_2 \underline{U} + \underline{U} + \underline{U}^{\wedge 3} - \underline{F}$.

Jacobian matrix

$$\mathbb{J} = -\mathbb{D}_2 + \mathbb{I} + 3\,\mathrm{diag}(U_j^2)$$

→ computational demo.

aside: continuation on the board.

If $\underline{g}(\underline{x}) = \nabla f$, then the Jacobian is the Hessian of f

$$\mathbb{J}_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j}.$$

In this case, $\mathbb{J}$ is symmetric, and will be positive definite near a local minimum.

Ex Rosenbrock function

$$f(\underline{x}) = 100 \sum_{j=1}^{n-1} (x_{j+1} - x_j^2)^2 + \sum_{j=1}^{n-1} (x_j - 1)^2$$

$\uparrow$ $\in \mathbb{R}^n$

→ computational examples.

(5) Steepest Descent.

What if the problem is too big to solve

$$\mathbb{H}\, \underline{\delta} = \underline{\nabla f} \,?$$

Well, ∇f is the direction of maximum increase of $\underline{f}$, so going in the direction $-\nabla f$ will reduce $\underline{f}$.

Iterate $\{\underline{x}^n\}$.

$$\underline{x}^n = \underline{x}^{n-1} - \varepsilon\, \underline{\nabla f}$$

↑

constant (small) or do a line search, reject if f increases.

Final <u>note</u>: Finding a <u>global</u> minimum may be impossible.

Suggested Problems:

#1 Solve the linear programming problem below (find appropriate software).

maximize $x_1 + 2x_2 + 3x_3 + 4x_4 + 5$

subject to the constraints

$$4x_1 + 3x_2 + 2x_3 + x_4 \leq 10$$
$$x_1 - x_3 + 2x_4 = 2$$
$$x_1 + x_2 + x_3 + x_4 \geq 1.$$

and $x_1 \geq 0$, $x_3 \geq 0$, $x_4 \geq 0$.

#2. Solve the following linear programming problem with N^2 variables g_{ij}.

minimize $\sum_{i=1}^{N} \sum_{j=1}^{N} g_{ij} (i-j)^2$

subject to $\sum_{i=1}^{N} g_{ij} = 1$ for all j

$\sum_{j=1}^{N} g_{ij} = 1$ for all i.

$0 \leq g_{ij} \leq \gamma/N$. for all i,j. ($\gamma > 1$ is a parameter).

Notes: + as $N \to \infty$ this approximates a continuum optimal transport problem.

+ You should see interesting behaviour around $\gamma = 2$.

+ as $\gamma \to \infty$ nonzero g values should cluster around the diagonal.

#3. Export the python exponential data points to R and do the regression in that platform.

#4. Investigate ways to limit the influence of the outlier in the linear regression. There are statistical methods to identify outliers. You can use l_1 error estimation. You could also try the following:

$$\text{minimize} \sum_{i=1}^{m} r(mx_i + b - y_i)$$

where $r(z) = \dfrac{z^2}{1+z^2/\sigma^2}$.

as $\sigma \to \infty$ this tends to the least squares problem. You could take $\sigma \approx$ noise level to do the "right" thing. If you do the nonlinear solve yourself, you may need to do continuation in σ from σ large (use least squares to start).

#5 apply gradient descent to the Rosenbrock function, using different constant ε or "strategies".