This in place to keep some study notes, and other items to remember while taking this hard course.

1 Some HOWTO questions

1.1 How to show that sum of two convex functions is also convex function?

Let $G(u) = g(u) + f(u)$ where we know g, f are two convex functions. We need to show $G\left(u^{\lambda}\right)$ is also convex. Then, pick point $u^{\lambda} \in U$ therefore

But the set U is convex (it must be, these are convex functions, so their domain is convex by definition). Pick a point $u^{\lambda} = (1-\lambda)u^1 + \lambda u^2$ where $\lambda \in [0,1]$ and $u^1, u^2 \in U$. Hence we can write

$$\begin{aligned} G\left(u^{\lambda}\right) &= g\left(u^{\lambda}\right) + f\left(u^{\lambda}\right) \\ G\left((1-\lambda)u^1 + \lambda u^2\right) &= g\left((1-\lambda)u^1 + \lambda u^2\right) + f\left((1-\lambda)u^1 + \lambda u^2\right) \end{aligned}$$

But $g\left((1-\lambda)u^1 + \lambda u^2\right) \leq (1-\lambda)g\left(u^1\right) + \lambda g\left(u^2\right)$ and the same for f. Then the above reduces to

$$\begin{aligned} G\left((1-\lambda)u^1 + \lambda u^2\right) &\leq (1-\lambda)g\left(u^1\right) + \lambda g\left(u^2\right) + (1-\lambda)f\left(u^1\right) + fg\left(u^2\right) \\ &= (1-\lambda)\left(g\left(u^1\right) + f\left(u^1\right)\right) + \lambda\left(g\left(u^2\right) + f\left(u^2\right)\right) \end{aligned}$$

But $G(u) = g(u) + f(u)$, then the RHS above becomes

$$G\left((1-\lambda)u^1 + \lambda u^2\right) \leq (1-\lambda)G\left(u^1\right) + \lambda G\left(u^2\right)$$

Therefore G is a convex function.

1.2 What is convex Hull?

Smallest set that contains all the sets inside it, such that it is also convex. (put a closed convex "container" around everything)

1.3 Is convex full same as Polytope?

No. Polytope is region which has straight edges (flat sides) and also be convex. But `http://mathworld.wolfram.com/Polytope.html` says "The word polytope is used to mean a number of related, but slightly different mathematical objects. A convex polytope may be defined as the convex hull of a finite set of points" And `https://en.wikipedia.org/wiki/Polytope` says "In elementary geometry, a polytope is a geometric object with flat sides, and may exist in any general number of dimensions n as an n-dimensional polytope or n-polytope"

1.4 What is difference between polytope and polyhedron?

`https://en.wikipedia.org/wiki/Polytope` says "In elementary geometry, a polyhedron (plural polyhedra or polyhedrons) is a solid in three dimensions with flat polygonal faces, straight edges and sharp corners or vertices."

Polyhedron can be convex or not. But polyhedron can be open? While polytope not. Need to check.

2 Some things to remember

1. Principle of optimality, by Bellman: "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optinal policy with regard to the state resulting from the first decision" Proof is by contradiction. See page 54, optimal contrl theory by Donald kirk. Simplist proof I've seen. An important case is when the performance index J is quadratic as with LQR. We only looked at case where is no coupling term in the LQR in this course. $J = \min_u \sum_{k=0}^{\infty} x^T Q x + u^T R u$. This is solved for steady state by solving Riccati equation. For discrete case, use Matlab dare() function. See Introduction to Dynamic Programming: International Series in Modern Applied Mathematics and Computer Science, Volume 1 (Pergamon International Library ... Technology, Engineering & Social Studies)

2. Remember difference between state variables, and descision variable. There can be more than one state variable in the problem, but the number of decisions to make at each state is different. see problem 1, HW 7 for example. The fire stations allocation problem. In that problem, we had one state variable, which is the number of stations available. There more state variables there are, the harder it will be to solve by hand.

3. $$\lim_{\lambda \to 0} \frac{J(\mathbf{u} + \lambda \mathbf{d}) - J(\mathbf{u})}{\lambda} = [\nabla J(\mathbf{u})]^T \cdot \mathbf{d}$$

   Remember, $\nabla J(\mathbf{u})$ is column vector. $\nabla J(\mathbf{u}) = \begin{bmatrix} \frac{\partial J(u)}{\partial u_1} \\ \vdots \\ \frac{\partial J(u)}{\partial u_n} \end{bmatrix}$. This vector is the direction along which function $J(\mathbf{u})$ will increase the most, among all other directions, at the point it is being evaluated at.

4. For polytope, this is useful trick.

   $$\begin{aligned} \mathbf{u} &= \sum_{i=1}^{m} \lambda_i v^i \\ \|\mathbf{u}\| &= \left\| \sum_{i=1}^{m} \lambda_i v^i \right\| \\ &\leq \sum_{i=1}^{m} \left\| \lambda_i v^i \right\| \end{aligned}$$

   The last step was done using triangle inequality.

5. Definition of continuity: If $u^k \to u^*$ then $J(u^k) \to J(u^*)$. We write $\lim_{k\to\infty} J(u^k) = J(u^*)$. This is for all u^k sequences. See real analysis handout. If $u^k \to u^*$ then this is the same as $\lim_{k\to\infty} \|u^k - u^*\| = 0$

6. closed sets is one which include all its limits points. (includes it boundaries). Use $[0,1]$ for closed. Use $(0,1)$ for open set. A set can be both open and closed at same time (isn't math fun?, wish life was this flexible).

7. Intersection of closed sets is also closed set. If sets are convex, then the intersection is convex. But the union of convex sets is not convex. (example, union of 2 circles).

8. B-W, tells us that a sequence u^k that do not converge, as long as it is in a compact set, it will contain at least one subsequence in it, $u^{k,i}$ which does converge to u^*. So in a compact set, we can always find at least one subsequence that converges to u^* even inside non-converging sequences.

9. If a set is not compact, then not all is lost. Assume set is closed but unbounded. Hence not compact. What we do, it set some R large enough, and consider the set of all elements $\|u\| \leq R$. Then the new set is compact.

10. $J(u) = au^2 + bu + c$ is coercive for $a > 0$. Note, the function $J(u)$ to be coercive, has to blow up in all directions. For example, e^u is not coercive. If A is positive definite matrix and $\mathbf{b} \in \Re^n$ and $c \in \Re$, then $J(u) = \mathbf{u}^T A\mathbf{u} + \mathbf{b}^T\mathbf{u} + c$ is coercive function. To establish this, convert to scalar. Use $\lambda_{\min} \|\mathbf{u}\|^2 \leq \mathbf{u}^T A\mathbf{u}$ and use $\mathbf{b}^T\mathbf{u} \leq \|\mathbf{b}\| \|\mathbf{u}\|$, then $J(u) \leq \lambda_{\min} \|\mathbf{u}\|^2 + \|\mathbf{b}\| \|\mathbf{u}\| + c$. Since P.D. matrix, then $\lambda_{\min} > 0$. Hence this is the same as $J(u) = au^2 + bu + c$ for $a > 0$. So coercive.

11. If in $J(u) = \mathbf{u}^T A\mathbf{u} + \mathbf{b}^T\mathbf{u} + c$ the matrix A is not symmetric., write as $J(u) = \frac{1}{2}\mathbf{u}^T \left(A^T + A\right)\mathbf{u} + \mathbf{b}^T\mathbf{u} + c$. Now it expressions becomes symmetric.

12. $\sum_{i,j} x_i x_j = \left(\sum_i x_i\right)^2$

13. If $\bar{\alpha} = \frac{1}{n}\sum_i \alpha_i$ then $\sum_i (\alpha_i - \bar{\alpha})^2 \geq 0$. Used to show Hessian is P.D. for given $J(u)$. See HW 2, last problem.

14. $x^T A x = \sum_{ij} A_{ij} x_i x_j$

15. To find a basic solution x_B which is not feasible, just find basic solution with at least one entry negative. Since this violates the constraints (we also use $x \geq 0$) for feasibility, then x_B solves $Ax = b$ but not feasible. i.e. $\begin{bmatrix} I & B \end{bmatrix} \begin{bmatrix} x_B \\ 0 \end{bmatrix} = b$ with some elements in x_B negative. For basic solution to also be feasible, all its entries have to be positive. (verify).

16. Solution to $Ax = b$ are all of the form $x_p + x_h$ where x_h is solution to $Ax = 0$ and x_p is a particular solution to $Ax = b$. This is similar to when we talk about solution to ODE. We look for homogeneous solution to the ODE (when the RHS is zero) and add to it a particular solution to original ODE with the rhs not zero, and add them to obtain the general solution.

17. difference between Newton method and conjugate gradient, is that CG works well from a distance, since it does not need the Hessian. CG will converge in N steps if $J(u)$ was quadratic function. Newton will converge in one step for quadratic, but it works well only if close to the optimal since it uses the Hessian as per above.

18. CG has superlinear convergence. This does not apply to steepest descent.

19. difference betwen steepest descent and conjugate direction is this: In SD, we use $\nabla J(u^k)$ as the direction to move at each step. i.e we use

$$u^{k+1} = u^k - h\frac{\nabla J(u^k)}{\|\nabla J(u^k)\|}$$

Where h above is either fixed step or optimal. But In CD we use v^k which is the mutual conjugate vector to all previous v^i. See my table of summary for this below as they can get confusing to know the difference.

20. To use Dynamic programming, the problem should have optimal substructure, and also should have an overlaping sub-problems. Sometimes hard to see or check for this.

21. Steepest descent with optimal step size has quadratic convergence property.

22. For symmetric Q, then $\frac{\partial(x^T Q x)}{\partial x} = 2Qx$

3 Example using conjugate directions

This example is solved in number of ways. Given quadratic function $J(x_1, x_2) = \frac{1}{2}x^T A x + b x^T$ where $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$. To find x^*, which minimizes $J(x)$. Let $A = \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix}$ and $b = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$

3.1 First method, Direct calculus

$$\begin{aligned} \nabla J(x) &= 0 \\ Ax + b &= \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\ \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix} &= \begin{bmatrix} 0 \\ 0 \end{bmatrix} \\ \begin{bmatrix} 4x_1 + 2x_2 - 1 \\ 2x_1 + 2x_2 + 1 \end{bmatrix} &= \begin{bmatrix} 0 \\ 0 \end{bmatrix} \end{aligned}$$

Solving gives

$$x^* = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 1 \\ -\frac{3}{2} \end{bmatrix}$$

3.2 Second method, basic Conjugate direction

Since A is of size $n = 2$, then this will converge in 2 steps using conjugate directions. let $x^0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$. Let first direction be

$$v^0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Then

$$h_0 = \frac{-\left(v^0\right)^T \nabla J\left(x^0\right)}{\left(v^0\right)^T A v^0} = \frac{-\left(v^0\right)^T \left(Ax^0 + b\right)}{\left(v^0\right)^T A v^0} = \frac{-\begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix}}{\begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}} = \frac{1}{4}$$

Hence

$$\begin{aligned} x^1 &= x^0 + h_0 v^0 \\ &= \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \frac{1}{4} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{1}{4} \\ 0 \end{bmatrix} \end{aligned}$$

Second step. We need to find v^1. Using conjugate mutual property of A, we solve for v^1 using

$$\begin{aligned} \left(v^0\right)^T A v^1 &= 0 \\ \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} &= 0 \\ 4v_1 + 2v_2 &= 0 \end{aligned}$$

Let $v_1 = 1$ then $v_2 = -2$ and hence

$$v^1 = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$$

Now we find the next optimal step

$$h_1 = \frac{-\left(v^1\right)^T \nabla J\left(x^1\right)}{\left(v^1\right)^T A v^1} = \frac{-\left(v^1\right)^T \left(Ax^1 + b\right)}{\left(v^1\right)^T A v^1} = \frac{-\begin{bmatrix}1 & -2\end{bmatrix}\left(\begin{bmatrix}4 & 2\\2 & 2\end{bmatrix}\begin{bmatrix}\frac{1}{4}\\0\end{bmatrix} + \begin{bmatrix}-1\\1\end{bmatrix}\right)}{\begin{bmatrix}1 & -2\end{bmatrix}\begin{bmatrix}4 & 2\\2 & 2\end{bmatrix}\begin{bmatrix}1\\-2\end{bmatrix}} = \frac{3}{4}$$

Hence

$$\begin{aligned} x^2 &= x^1 + h_1 v^1 \\ &= \begin{bmatrix}\frac{1}{4}\\0\end{bmatrix} + \frac{3}{4}\begin{bmatrix}1\\-2\end{bmatrix} \\ &= \begin{bmatrix}1\\-\frac{3}{2}\end{bmatrix} \end{aligned}$$

Which is x^* that we found in first method. Using $n = 2$ steps as expected. In implementation, we will have to check we converged by looking at $\nabla J\left(x^2\right)$ which will be

$$\begin{aligned} \nabla J\left(x^2\right) &= Ax^2 + b \\ &= \begin{bmatrix}4 & 2\\2 & 2\end{bmatrix}\begin{bmatrix}1\\-\frac{3}{2}\end{bmatrix} + \begin{bmatrix}-1\\1\end{bmatrix} \\ &= \begin{bmatrix}0\\0\end{bmatrix} \end{aligned}$$

As expected.

3.3 Third method. Conjugate gradient

The difference here is that we find v^i on the fly after each step. Unlike the conjugate direction method, where v^i are all pre-computed. Let $v^0 = \nabla\left(J\left(x^0\right)\right) = \begin{bmatrix}-1\\1\end{bmatrix}$. In this method, we always pick $v^0 = \nabla\left(J\left(x^0\right)\right)$, where x^0 is the starting guess vector. First step

$$h_0 = \frac{-\left(v^0\right)^T \nabla J\left(x^0\right)}{\left(v^0\right)^T A v^0} = \frac{-\left(v^0\right)^T \left(Ax^0 + b\right)}{\left(v^0\right)^T A v^0} = \frac{-\begin{bmatrix}-1 & 1\end{bmatrix}\begin{bmatrix}-1\\1\end{bmatrix}}{\begin{bmatrix}-1 & 1\end{bmatrix}\begin{bmatrix}4 & 2\\2 & 2\end{bmatrix}\begin{bmatrix}-1\\1\end{bmatrix}} = \frac{-2}{2} = -1$$

Hence

$$\begin{aligned} x^1 &= x^0 + h_0 v^0 \\ &= \begin{bmatrix}0\\0\end{bmatrix} - 1\begin{bmatrix}-1\\1\end{bmatrix} = \begin{bmatrix}1\\-1\end{bmatrix} \end{aligned}$$

Now we find the mutual conjugate v^1 as follows

$$\beta_0 = \frac{\left(\nabla J\left(x^1\right)\right)^T A v^0}{\left(v^0\right)^T A v^0} = \frac{\left(Ax^1 + b\right)\left(Av^0\right)}{\left(v^0\right)^T A v^0} = \frac{\left(\begin{bmatrix}4 & 2\\2 & 2\end{bmatrix}\begin{bmatrix}1\\-1\end{bmatrix} + \begin{bmatrix}-1\\1\end{bmatrix}\right)^T \left(\begin{bmatrix}4 & 2\\2 & 2\end{bmatrix}\begin{bmatrix}-1\\1\end{bmatrix}\right)}{\begin{bmatrix}-1 & 1\end{bmatrix}\begin{bmatrix}4 & 2\\2 & 2\end{bmatrix}\begin{bmatrix}-1\\1\end{bmatrix}}$$

$$= \frac{\begin{bmatrix}1\\1\end{bmatrix}^T \begin{bmatrix}-2\\0\end{bmatrix}}{2} = \frac{-2}{2} = -1$$

Hence

$$\begin{aligned} v^1 &= -\nabla J\left(x^1\right) + \beta_0 v^0 \\ &= -\left(\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix}\right) - (1) \begin{bmatrix} -1 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} 0 \\ -2 \end{bmatrix} \end{aligned}$$

Now that we found v^1, we repeat the process.

$$h_1 = \frac{-\left(v^1\right)^T \nabla J\left(x^1\right)}{\left(v^1\right)^T A v^1} = \frac{-\left(v^1\right)^T \left(Ax^1 + b\right)}{\left(v^1\right)^T A v^1} = \frac{-\begin{bmatrix} 0 & -2 \end{bmatrix} \left(\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix}\right)}{\begin{bmatrix} 0 & -2 \end{bmatrix} \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ -2 \end{bmatrix}} = \frac{2}{8} = \frac{1}{4}$$

Hence

$$\begin{aligned} x^2 &= x^1 + h_1 v^1 \\ &= \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \left(\frac{1}{4}\right) \begin{bmatrix} 0 \\ -2 \end{bmatrix} \\ &= \begin{bmatrix} 1 \\ -\frac{3}{2} \end{bmatrix} \end{aligned}$$

Which is the same as with conjugate direction method. Converged in 2 steps also.

3.4 Fourth method. Conjugate gradient using Fletcher-Reeves

In this method

$$\beta_k = \frac{\nabla J\left(u^{k+1}\right)^T \nabla J\left(u^{k+1}\right)}{\nabla J\left(u^k\right)^T \nabla J\left(u^k\right)} = \frac{\left\|\nabla J\left(u^{k+1}\right)\right\|^2}{\left\|\nabla J\left(u^k\right)\right\|^2}$$

We also start here with $v^0 = \nabla J\left(u^0\right) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$ in this example.

$$h_0 = \frac{-\left(v^0\right)^T \nabla J\left(x^0\right)}{\left(v^0\right)^T A v^0} = \frac{-\left(v^0\right)^T \left(Ax^0 + b\right)}{\left(v^0\right)^T A v^0} = \frac{-\begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix}}{\begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix}} = \frac{-2}{2} = -1$$

Hence

$$\begin{aligned} x^1 &= x^0 + h_0 v^0 \\ &= \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 1 \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \end{aligned}$$

Now find the mutual conjugate v^1 as follows, using Fletcher-Reeves formula

$$\beta_0 = \frac{\left\|\nabla J\left(u^1\right)\right\|^2}{\left\|\nabla J\left(u^0\right)\right\|^2} = \frac{\left\|Ax^1 + b\right\|^2}{\left\|Ax^0 + b\right\|^2} = \frac{\left\|\left(\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix}\right)\right\|^2}{\left\|\left(\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix}\right)\right\|^2} = \frac{\left\|\begin{bmatrix} 1 \\ 1 \end{bmatrix}\right\|^2}{\left\|\begin{bmatrix} -1 \\ 1 \end{bmatrix}\right\|^2} = \frac{\left(\sqrt{2}\right)^2}{\left(\sqrt{2}\right)^2} = 1$$

$$
\begin{aligned}
v^1 &= -\nabla J\left(x^1\right) + \beta_0 v^0 \\
&= -\left(\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix}\begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix}\right) + (1)\begin{bmatrix} -1 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} -2 \\ 0 \end{bmatrix}
\end{aligned}
$$

Now that we found v^1, we repeat the process.

$$
h_1 = \frac{-\left(v^1\right)^T \nabla J\left(x^1\right)}{\left(v^1\right)^T A v^1} = \frac{-\left(v^1\right)^T \left(Ax^1 + b\right)}{\left(v^1\right)^T A v^1} = \frac{-\begin{bmatrix} -2 & 0 \end{bmatrix}\left(\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix}\begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix}\right)}{\begin{bmatrix} -2 & 0 \end{bmatrix}\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix}\begin{bmatrix} 0 \\ -2 \end{bmatrix}} = \frac{2}{8} = \frac{1}{4}
$$

Hence

$$
\begin{aligned}
x^2 &= x^1 + h_1 v^1 \\
&= \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \left(\frac{1}{4}\right)\begin{bmatrix} 0 \\ -2 \end{bmatrix} \\
&= \begin{bmatrix} 1 \\ -\frac{3}{2} \end{bmatrix}
\end{aligned}
$$

Which is the same as with conjugate direction method. It converges in 2 steps also.

3.5 Fifth method. Conjugate gradient using Polak-Ribiere

In this method

$$
\beta_k = \frac{\nabla J\left(u^{k+1}\right)^T \left(\nabla J\left(u^{k+1}\right) - \nabla J\left(u^k\right)\right)}{\nabla J\left(u^k\right)^T \nabla J\left(u^k\right)}
$$

We also start here with $v^0 = \nabla J\left(u^0\right) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$ in this example.

$$
h_0 = \frac{-\left(v^0\right)^T \nabla J\left(x^0\right)}{\left(v^0\right)^T A v^0} = \frac{-\left(v^0\right)^T \left(Ax^0 + b\right)}{\left(v^0\right)^T A v^0} = \frac{-\begin{bmatrix} -1 & 1 \end{bmatrix}\begin{bmatrix} -1 \\ 1 \end{bmatrix}}{\begin{bmatrix} -1 & 1 \end{bmatrix}\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix}\begin{bmatrix} -1 \\ 1 \end{bmatrix}} = \frac{-2}{2} = -1
$$

Hence

$$
\begin{aligned}
x^1 &= x^0 + h_0 v^0 \\
&= \begin{bmatrix} 0 \\ 0 \end{bmatrix} - 1\begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}
\end{aligned}
$$

Now we find the mutual conjugate v^1 direction as follows, using Polak-Ribiere formula

$$
\beta_0 = \frac{\nabla J\left(u^1\right)^T \left(\nabla J\left(u^1\right) - \nabla J\left(u^0\right)\right)}{\nabla J\left(u^0\right)^T \nabla J\left(u^0\right)}
$$

But

$$\nabla J\left(u^1\right) = \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$
$$\nabla J\left(u^0\right) = \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

Hence

$$\beta_0 = \frac{\begin{bmatrix} 1 & 1 \end{bmatrix} \left(\begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} -1 \\ 1 \end{bmatrix} \right)}{\begin{bmatrix} -1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 1 \end{bmatrix}} = \frac{2}{2} = 1$$

Hence

$$\begin{aligned} v^1 &= -\nabla J\left(x^1\right) + \beta_0 v^0 \\ &= -\left(\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix} \right) + (1) \begin{bmatrix} -1 \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} -2 \\ 0 \end{bmatrix} \end{aligned}$$

Now that we found v^1, we repeat the process.

$$h_1 = \frac{-\left(v^1\right)^T \nabla J\left(x^1\right)}{\left(v^1\right)^T A v^1} = \frac{-\left(v^1\right)^T \left(A x^1 + b\right)}{\left(v^1\right)^T A v^1} = \frac{-\begin{bmatrix} -2 & 0 \end{bmatrix} \left(\begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} -1 \\ 1 \end{bmatrix} \right)}{\begin{bmatrix} -2 & 0 \end{bmatrix} \begin{bmatrix} 4 & 2 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ -2 \end{bmatrix}} = \frac{2}{8} = \frac{1}{4}$$

Hence

$$\begin{aligned} x^2 &= x^1 + h_1 v^1 \\ &= \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \left(\frac{1}{4}\right) \begin{bmatrix} 0 \\ -2 \end{bmatrix} \\ &= \begin{bmatrix} 1 \\ -\frac{3}{2} \end{bmatrix} \end{aligned}$$

Which is the same as with conjugate direction method. Converges in 2 steps also as expected

4 collection of defintions

Basic solution for LP This is solution $\vec{x}$ which has non zero entries that correspond to linearly independent column in A. Where the constraints are $Ax = b$.

feasible solution for LP This is solution $\vec{x}$ which is in the feasible region. The region that satisfy the constraints. Feasible solution do not have to be basic.

basic and feasible solution for LP This is solution $\vec{x}$ which is both feasible and basic. Once we get to one of these, then simplex algorithm will jump from one basic feasible to the next, while reducing the $J(u)$ objective function until optimal is found.

Basic but not feasible solution is there one? Need example.

Newton Raphon method Iteration is

$$u^{k+1} = u^k - \frac{\nabla J(u^k)}{\nabla^2 J(u^k)}$$

where $\nabla^2 J(u^k)$ is the hessian. This is a A matrix in the quadratic expression

$$J(u) = \frac{1}{2} u^T A u + b^T u + c$$

Ofcourse we can't divide by matrix, this is the inverse of the Hessian. So the above is

$$u^{k+1} = u^k - \left[\nabla^2 J(u^k)\right]^{-1} \nabla J(u^k)$$

See handout Newton for example with $J(u)$ given and how to use this method to iterate to u^*. If $J(u)$ was quadratic, this will converge in one step.

Quadratic expression An expression is quadratic if it can be written as

$$\sum_i \sum_j a_{ij} u_i u_j + \sum_i b_i u_i + c$$

For example, $x_1^2 + 9x_1x_2 + 14x_2^2$ becomes

$$\begin{aligned} x_1^2 + 9x_1x_2 + 14x_2^2 &= a_{11}x_1x_1 + a_{21}x_2x_1 + a_{12}x_1x_2 + a_{22}x_2x_2 + b_1x_1 + b_2x_2 + c \\ &= a_{11}x_1^2 + a_{21}x_2x_1 + a_{12}x_1x_2 + a_{22}x_2^2 + b_1x_1 + b_2x_2 + c \end{aligned}$$

comparing both sides, we see that by setting $a_{11} = 1, a_{21} = \frac{9}{2}, a_{21} = \frac{9}{2}, a_{22} = 14$ and by setting $b_1 = 0, b_2 = 0$ and $c = 0$ we can write it in that form. Hence it is quadratic and

$$A = \begin{pmatrix} 1 & \frac{9}{2} \\ \frac{9}{2} & 14 \end{pmatrix}, b = \begin{pmatrix} 0 & 0 \end{pmatrix}$$

Therefore

$$\begin{aligned} x_1^2 + 9x_1x_2 + 14x_2^2 &= x^T A x + b^T x + c \\ &= \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} 1 & \frac{9}{2} \\ \frac{9}{2} & 14 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + 0 \end{aligned}$$

Since we are able to write $x_1^2 + 9x_1x_2 + 14x_2^2 = x^T A x + b^T x + c$ it is quadratic. Notice that the A matrix is always symmetric.

superlinear convergence A sequence $\{u^k\}$ in $\Re^n$ is said to converge superlinearly to u^* if the following holds. Given any $\theta \in (0, 1]$ then

$$\lim_{k \to \infty} \frac{\|u^k - u^*\|}{\theta^k} \to 0$$

Example is $u^k = e^{-k^2}$ Since $u^* = 0$ then $\frac{e^{-k^2}}{\theta^k} \to 0$ no matter what $\theta \in (0, 1]$ is. Remember, it has to go to zero for any θ

Quadratic convergence theorem Given quadratic

$$J(u) = \frac{1}{2} u^T A u + b^T u + c$$

And given N set of mutually conjugate vectors (with respect to A) $\{v^0, v^2, \dots, v^{N-1}\}$ then the conjugate direction algorithm converges to the optimal $u^* = -A^{-1}b$ in N steps of less. Proof in lecture 3/1/2016 (long)

A-conjugate vectors There are mutually conjugate vectors with respect to A. The directions $\{v^0, v^1, \ldots, v^{N-1}\}$ are said to be mutually conjugate with respect to A if

$$(v^i)^T A v^j = 0$$

For all $i \neq j$

5 Summary of iterative search algorithms

5.1 steepest descent

5.1.1 steepest descent, any objective function $J(x)$

The input is $x(0)$ the initial starting point and $J(x)$ itself.

1.	init	$x^0 = x(0), k = 0$
2.	g^k	$= \nabla J(x^k)$
3.	α_k	$= \min_\alpha J\left(x^k - \alpha \frac{g^k}{\|g^k\|}\right)$ (line search)
4.	x^{k+1}	$= x^k - \alpha_k \frac{g^k}{\|g^k\|}$
5.	k	$= k + 1$
6.	goto 2	

5.1.2 steepest descent, Quadratic objective function $J(x)$

If the objective function $J(x)$ is quadratic $J(x) = x^T A x - b^T x + c$ then there is no need to do the line search.

The input is $x(0)$ the initial starting point and A, b. The algorithm becomes

1.	Init	$x^0 = x(0), k = 0$
2.	g^k	$= \nabla J(x^k) = Ax^k - b$
3.	α_k	$= \frac{\left[g^k\right]^T g^k}{\left[g^k\right]^T A\, g^k}$
4.	x^{k+1}	$= x^k - \alpha_k g^k$
5.	k	$= k + 1$
6.	goto 2	

5.2 conjugate direction, Quadratic function $J(x)$

For quadratic $J(x) = x^T A x - b^T x + c$ the conjugate direction algorithm is as follows.

Input $x(0)$ starting point, and A, b and set of n mutually conjugate vectors $\{v^0, v^1, \ldots, v^{n-1}\}$ with respect to A, where n is the size of A. In otherwords, $(v^i)^T A v^j = 0$ for $i \neq j$.

These v^i vectors have to be generated before starting the algorithm. With the conjugate gradient (below), these A-conjugate vectors are generated on the fly inside the algorithm as it iterates. This is the main difference between conjugate direction and conjugate gradient.

1.	init	$u^0 = x(0)$, $k = 0$
2.	g^k	$= \nabla J(x^k) = Ax^k - b$
3.	α_k	$= \frac{[g^k]^T v^k}{[g^k]^T A\, v^k}$
4.	x^{k+1}	$= x^k - \alpha_k v^k$
5.	k	$= k + 1$
6.	goto 2	

We see the difference between the above and the steepest descent before it, is in line 3,4. Where now v^k replaces g^k in two places.

5.3 conjugate gradient, Quadratic function $J(x)$

Conjugate direction required finding set of v vectors before starting the algorithm. This algorithm generates these vectors as it runs.

Input $x(0)$ starting point, and A, b.

1.	Init	$u^0 = x(0)$, $k = 0$, $g^0 = \nabla J(x^0) = Ax^0 - b$, $v^0 = -g^0$
2.	α_k	$= \frac{[g^k]^T v^k}{[g^k]^T A\, v^k}$
4.	x^{k+1}	$= x^k + \alpha_k v^k$
5.	g^{k+1}	$= \nabla J(x^{k+1}) = Ax^{k+1} - b$
6.	β	$= \frac{[g^{k+1}]^T A v^k}{[v^k]^T A v^k}$
7.	v^{k+1}	$= -g^{k+1} + \beta v^k$
8.	k	$= k + 1$
9.	goto 2	

5.4 conjugate gradient, None quadratic function $J(x)$, Hestenses-Stiefel

If we do not have quadratic function, then we can not use A, b to generate β. The above algorithm becomes using Hestenses-Stiefel

Input $x(0)$ starting point.

1.	Init	$u^0 = x(0)$, $k = 0$, $g^0 = \nabla J(x^0)$, $v^0 = -g^0$
2.	α_k	$= \min_\alpha J\left(x^k + \alpha v^k\right)$ (line search)
3.	x^{k+1}	$= x^k + \alpha_k v^k$
4.	g^{k+1}	$= \nabla J(x^{k+1})$
5.	β	$= \frac{[g^{k+1}]^T [g^{k+1} - g^k]}{[v^k]^T [g^{k+1} - g^k]}$
6.	v^{k+1}	$= -g^{k+1} + \beta v^k$
7.	k	$= k + 1$
8.	goto 2	

5.5 conjugate gradient, None quadratic function $J(x)$, Polak-Ribiere

If we do not have quadratic function, then we can not use A, b to generate β. The conjugate gradient algorithm becomes using Polak-Ribiere as follows

Input $x(0)$ starting point.

1.	Init	$u^0 = x(0)$, $k = 0$, $g^0 = \nabla J(x^0)$, $v^0 = -g^0$
2.	α_k	$= \min_\alpha J\left(x^k + \alpha v^k\right)$ (line search)
3.	x^{k+1}	$= x^k + \alpha_k v^k$
4.	g^{k+1}	$= \nabla J(x^{k+1})$
5.	β	$= \frac{\left[g^{k+1}\right]^T\left[g^{k+1}-g^k\right]}{\left[g^k\right]^T g^k}$
6.	v^{k+1}	$= -g^{k+1} + \beta v^k$
7.	k	$= k + 1$
8.	goto 2	

5.6 conjugate gradient, None quadratic function $J(x)$, Fletcher-Reeves

If we do not have quadratic function, then we can not use A, b to generate β. The conjugate gradient algorithm becomes using Fletcher-Reeves as follows

Input $x(0)$ starting point.

1.	Init	$u^0 = x(0)$, $k = 0$, $g^0 = \nabla J(x^0)$, $v^0 = -g^0$
2.	α_k	$= \min_\alpha J\left(x^k + \alpha v^k\right)$ (line search)
3.	x^{k+1}	$= x^k + \alpha_k v^k$
4.	g^{k+1}	$= \nabla J(x^{k+1})$
5.	β	$= \frac{\left[g^{k+1}\right]^T g^{k+1}}{\left[g^k\right]^T g^k}$
6.	v^{k+1}	$= -g^{k+1} + \beta v^k$
7.	k	$= k + 1$
8.	goto 2	