* Polak_Ribière:

$$\beta^{(k)} = \frac{g^{T(k)}\left(g^{(k)} - g^{(k-1)}\right)}{g^{T(k-1)} g^{(k-1)}}$$

Convergence for PR method can be guaranteed if we modify

$$\beta \to \max(\beta, 0)$$

automatic reset

Generic line Search Method:

1. Pick an initial iterate x^0 by educated guess, $k = 0$
2. Until Convergence (x^k)

   i) Calculate a search direction d^k from x^k

this direction is the descent direction:

$$[g^k]^T d^k < 0 \quad \text{if } g^k \neq 0$$

small enough step away from x^k in the direction of d^k the objective function will be reduced

(ii) Calculate a suitable step length $\alpha^k > 0$ such that

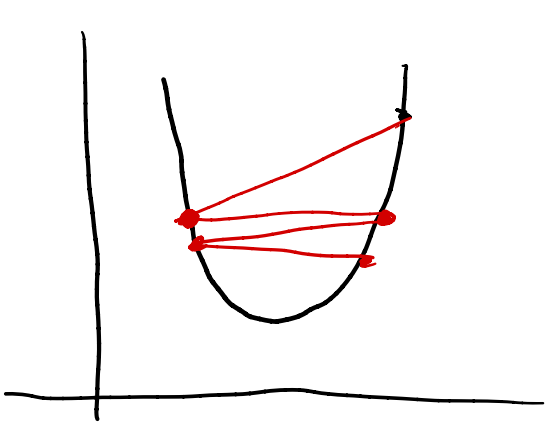
$$f(x^k + \alpha^k d^k) < f^k$$

computation of α^k is called line search. ⟹ usually an inner iterative loop

(iii) set $x^{k+1} = x^k + \alpha^k d^k$

⟹ How to compute α^k ?

Challenge:



Step size too long!

Step size too short!

Exact line Search:

α^k is picked as a solution to

ELS $\quad \underset{\alpha}{\text{minimize}} \quad f(x^k + \alpha d^k)$

s.t. $\quad \alpha \geq 0$ ↑ Invariant Opt.

$\Longrightarrow$ not cost effective

Inexact line Search Methods:

- formulate a criterion that ensure steps are not too long or too short
- Pick a good initial step size
- Construct sequence of updates satisfying the first criterion

(1) Backtracking line Search:

1. Given $\alpha_{init} > 0$ (eg. $\alpha_{init} = 1$) let $\alpha^{(0)} = \alpha_{init}$ and $l = 0$
2. Until $f(x^k + \alpha^{(l)} d^k) < f^k$
   i) set $\alpha^{(l+1)} = \tau \alpha^{(l)}$, where $\tau \in (0,1)$ is fixed, $\boxed{\tau = 1/2}$
   ii) increment l by 1

3. set $\alpha^k = \alpha^{(l)}$

This method prevents the step from getting too small, does not prevent steps that are too long relative to the decrease in f

to prevent long steps we require the Armijo Condition:

$$f(x^k + \alpha^k d^k) \leq f(x^k) + \alpha^k \beta \left[g^k\right]^T d^k$$

fixed value

($\beta = 0.1$ or even $\beta = 0.0001$)

We require that the achieved reduction be at least a fixed fraction (β) of the reduction promised by g^k

Backtracking - Armijo line Search:

1. Given $\alpha_{init} > 0$ (eg. $\alpha_{init} = 1$)
   let $\alpha^{(0)} = \alpha_{init}$ and $l = 0$
2. Until $f(x^k + \alpha^{(l)} d^k) \leq f(x^k) + \alpha^l \beta \cdot [g^k]^T d^k$
   i) set $\alpha^{(l+1)} = \tau \alpha^{(l)}$
   ii) increment l by 1
3. Set $\alpha^k = \alpha^{(l)}$

Method of Steepest descent:

$$d^k = -g^k$$

$$f^{k+1} = f(x^k - g^k)$$

$\longrightarrow$ not optimal !!

Second Order Methods:

We use second order derivative (Hessian) to inform the optimisation algorithm.

Newton's Method:

- first order: How far in the descent direction?
- Second order: Quadratic approximation of the obj. function

for univariate optimisation, the Quadratic approximation about $x^{(k)}$ comes from 2nd order Taylor Series expansion:

$$q(x) = f(x^k) + (x - x^k) f'(x^k) + \frac{(x - x^k)^2}{2} f''(x^k)$$

Setting the derivative to zero:

$$\frac{dq(x)}{dx} = f'(x^k) + (x - x^k) f''(x^k) = 0$$

$$x^{k+1} = x^k - \frac{f'(x^k)}{f''(x^k)}$$

Note: Involves dividing by f''

$\Rightarrow$ if $f'' = 0$ or close to it

* Oscillations
* Overshoots

$\longrightarrow$ bowl-like region (convex) close to local min

$\Rightarrow$ Quadratic convergence

"for multivariate optimisation"

$$q(x) = f(x^k) + (g^k)^T (x - x^k) + \frac{1}{2}(x - x^k)^T H^k (x - x^k)$$

(g^k: gradient; H^k: Hessian)

We evaluate the gradient & set it to zero:

$$\nabla q(x^k) = g^k + H^k (x - x^k)$$

next iterate:

$$x^{k+1} = x^k - (H^{(k)})^{-1} g^k$$

→ if f is quadratic and its Hessian is positive definite, then the update converges to global min in **one step**

$f(x_1, x_2)$

$$H_{(f)} = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \frac{\partial^2 f}{\partial x_2 \partial x_1} \\ \frac{\partial^2 f}{\partial x_1 \partial x_2} & \frac{\partial^2 f}{\partial x_2 \partial x_2} \end{pmatrix}$$

Ex: with $x^{(1)} = [9, 8]$, we will use Newton's method to minimize Booth's function:

$$f(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$$

form gradient:

$$g^{(1)} = \nabla f(x^{(1)}) = [10x_1 + 8x_2 - 34, \; 8x_1 + 10x_2 - 38]$$

form Hessian:

$$H^{(1)} =$$

the first iteration of Newton's method:

$$x^{(2)} = x^{(1)} - \left(H^{(1)}\right)^{-1} g^{(1)}$$

$$= \begin{bmatrix} 9 \\ 8 \end{bmatrix} - \begin{bmatrix} 10 & 8 \\ 8 & 10 \end{bmatrix}^{-1} \begin{bmatrix} 120 \\ 114 \end{bmatrix} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

Optimum.

Gradient at $x^{(2)}$ is zero!

- Hessian is positive definite everywhere ⟶ global optimum

Newton's method can also be used to supply a descent dir to line search.

$$d^{k} = -\left(H^{(k)}\right)^{-1} g^{(k)}$$

Secant Method:

Newton's method requires first & second order derivative (f', f'')

$|g, |+|$

in many cases f' ✓

f'' ✗

Secant Method applies Newton's method using estimates of the second derivative → f' ✓

Second derivative

$$\approx f''(x^{(k)}) \approx \frac{f'(x^{(k)}) - f'(x^{(k-1)})}{x^{k} - x^{(k-1)}}$$

app

$$x^{k+1} \leftarrow x^{k} - \frac{x^{(k)} - x^{(k-1)}}{f'(x^{(k)}) - f'(x^{(k-1)})} f'(x^{(k)})$$

→ needs more iterations than Newton's method.

Quasi-Newton Method:

Secant Method approximates f''

quasi-Newton approximates Hessian

$$x^{k+1} = x^k - \alpha^{(k)} Q^{(k)} g^k$$

$\alpha^{(k)}$: Scalar step size

$Q^{(k)}$: App. inverse Hessian

Usually $Q^{(1)} = I$

→ update Q by learning info from the system.

define:

$$\gamma^{(k+1)} = g^{(k+1)} - g^{(k)}$$

$$\delta^{(k+1)} = x^{(k+1)} - x^{(k)}$$

→ The Davidson-Hecher-Powell (DFP) method

$$Q^{k+1} = \left[Q - \frac{Q\delta\delta^T Q}{\gamma^T Q \gamma} + \frac{\delta\delta^T}{\delta^T\gamma} \right]^{(k)}$$

Q in DFP has three properties:

1. Q remains symmetric and Positive definite
2. if $f(x) = 1/2\, x^T A x + b^T x + c$ then $Q = A^{-1}$, then DFP has the same convergence

Properties of conjugate gradient method.

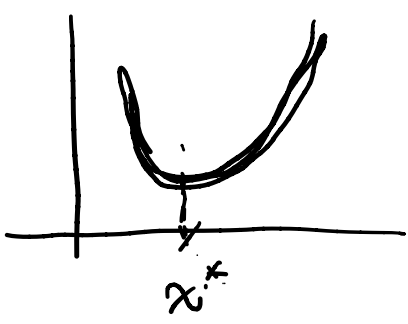
3. For High-dimensions, storing and updating Q can be significant compared to CG

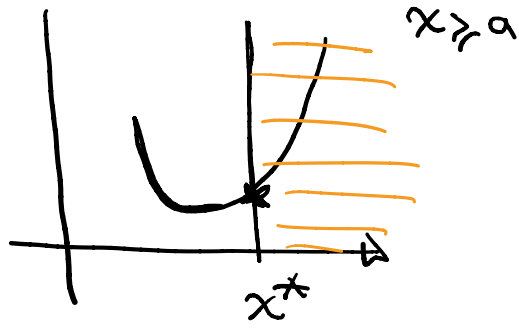
→ Brayden-Hecher-Goldfarb-Shanno (BFGS) methods

$$Q = Q - \left(\frac{\delta\delta^T Q + Q\gamma\delta^T}{\delta^T\delta}\right) + \left(1 + \frac{\gamma^T Q\gamma}{\delta^T\gamma}\right) \times \left(\frac{\delta\delta^T}{\delta^T\gamma}\right)$$

Constraint Optimisation:

→ transform Constraint Optimisation into unconstraint opt.





Constraint types:

* equality constraints, $h(x) = 0$
* inequality constraints, $g(x) \leq b$

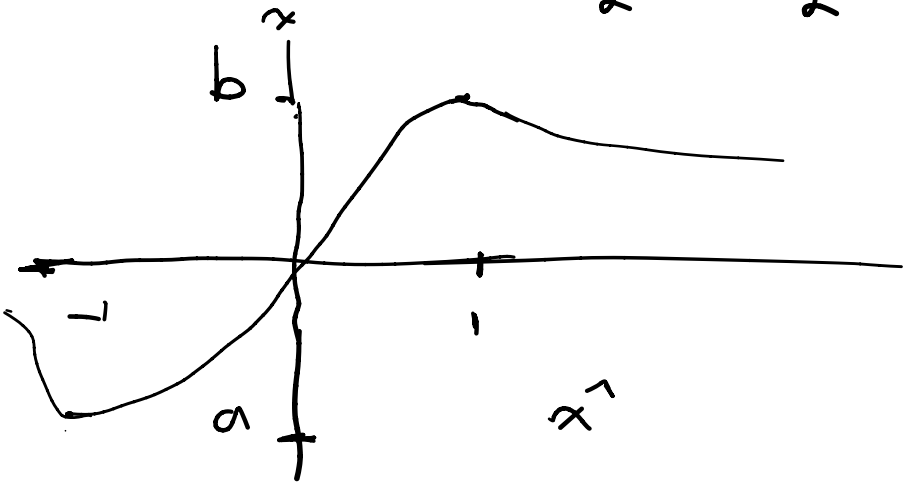
→ transformation to remove constraints:

e.g. bound constraints $a \leq x \leq b$

remove by passing x through

a transformation

$$x = t_{a,b}(\hat{x}) = \frac{b+a}{2} + \frac{b-a}{2}\left(\frac{2\hat{x}}{1+\hat{x}^2}\right)$$



- Lagrange Multipliers.

is used to optimise a function subject to equality constraint.

$$\begin{cases} \underset{x}{\text{minimize}} & f(x) \\ \text{subject to} & h(x) = 0 \end{cases}$$

This method is used to find where a Contourline of f is aligned with h.

$\Rightarrow$ grad $\perp$ contourlines

$\rightarrow$ ∇f, ∇h $\Rightarrow$ align.

$\Rightarrow$ $\nabla f_{(x)} = \lambda \nabla h(x)$

↑ Lagrange multiplier.

Ex: minimize $-\exp\left(-(x_1 x_2 - 3/2)^2 - (x_2 - 3/2)^2\right)$

subject to $x_1 - x_2^2 = 0$

$\rightarrow$ $x_1 = x_2^2$, substitute in the eqn.

$fun = -\exp\left(-(x_2^3 - 3/2)^2 - (x_2 - 3/2)^2\right)$

$\Rightarrow$ derivation

$$\partial f / \partial x_2 = 6 \exp\left(-(x_2 - 3/2)^3 - (x_2 - 3/2)^2\right) = 0$$

$$\left(x_2^5 - 3/2\, x_2^2 + 1/3\, x_2 - 1/2\right)$$

$\rightarrow$ $x_2 = 1.16\bar{6} \rightarrow x^* = [1.35\bar{8}, 1.16\bar{6}]$

contour of "f" align with "h"

if x^* optimises f along h

$\Rightarrow$ directional derivative @ x^* along h must be zero

$\Downarrow$

tangent to the contour lines.

formulate the Lagrangian which is a function of the design variables (x) & Lagrange multipliers (λ)

$$\mathcal{L}(x, \lambda) = f(x) - \lambda h(x)$$

↑ adjoint variable

Solving $\nabla \mathcal{L}(x, \lambda) = 0$ adjoint eqn.

$$\Rightarrow \quad \nabla_x \mathcal{L} = 0 \Rightarrow \nabla f = \lambda \nabla h$$

$$\nabla_\lambda \mathcal{L} = 0 \Rightarrow h(x) = 0$$

E.x Same Problem as before.

$$\underset{(x_1, x_2, \lambda)}{\mathcal{L}}: f(x) - \lambda h(x)$$

$\nabla . \mathcal{L}$:

$$\frac{\partial \mathcal{L}}{\partial x_1} = 2x_2 f(x)\left(3/2 - x_1 x_2\right) - \lambda = 0$$

$$\frac{\partial \mathcal{L}}{\partial x_2} = 2\lambda x_2 + f(x)\left(-2x_1(x_1 x_2^{3/2}) - 2(x_2^{3/2})\right) = 0$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = x_2^2 - x_1 = 0$$

$$\Longrightarrow \begin{cases} x_1 = 1.358 \\ x_2 = 1.165 \\ \lambda = 1.170 \end{cases}$$

The method of Lagrange multipliers can be extended to multiple constraints:

minimize $f(x)$

s.t. $h_1(x) = 0$ (1)

$h_2(x) = 0$

↓

minimize $f(x)$ (2)

$\sim$

s.t. $h_1^2(x) + h_2^2(x) = h_{comb} = 0$; $c > 0$

$\Rightarrow$ The soln (1) & (2) the same

$$\nabla f - \lambda \nabla h_{comb} = 0$$

$$\nabla f - \lambda (h_1 \nabla h_1 + h_2 \nabla h_2) = 0$$

for multiple constraints:

$$L(x, \lambda) = f(x) - \sum_{i=1}^{l} \lambda_i h_i(x) = f(x) - \lambda^T h(x)$$

Inequality Constraints:

$$\min_x \quad f(x)$$

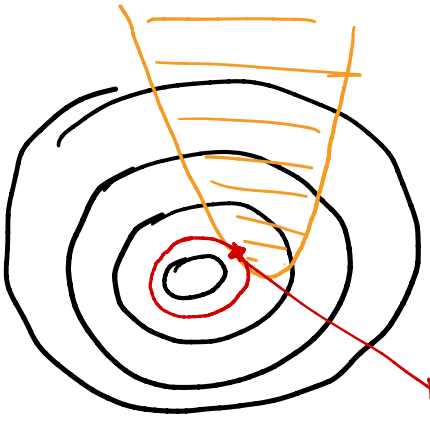
$$\text{s.t} \quad g(x) \leq 0$$

if the soln. lies at the constraint boundary, the Lagrange condition holds

$$\nabla f - \mu \nabla g = 0$$

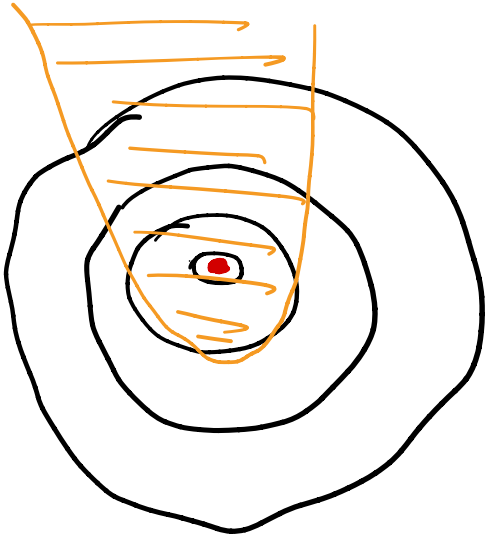
for some constant μ

$\Rightarrow$ active constraints



x^* at boundary of $f(x)$ $g(x)=c$

$\Rightarrow$ active constraint



$\rightarrow$ inactive constraint.

minimize $\bar{J}(\phi, q)$

- Drag
- Heat release
- Flame speed
- mixing

s.t $F(\phi, q) = 0$

$NS = 0$

Fluid motion