10.34, Numerical Methods Applied to Chemical Engineering
Professor William H. Green
Lecture #16: Unconstrained Optimization.

Unconstrained Optimization

$\min_{\underline{x}} f(\underline{x})$

$\bullet$ $\underline{x}^{guess}$, $\underline{x}^{[k]}$ $\qquad$ $\bullet$ $\underline{x}^{[k+1]}$

Require: $f(\underline{x}^{[k+1]}) < f(\underline{x}^{[k]})$

Which direction to move?

Move Downhill → "Steepest Descent"

- very robust but poor convergence at the end

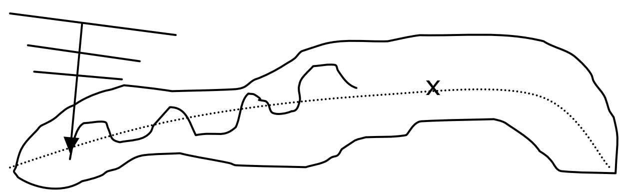

Figure 1. Diagram of steepest descent approach to global minimum.

Unless you start on the center line, you will zigzag inefficiently

- going down contour lines is easy with this method

$$\underline{f}_{approx}(\underline{x}) = \underline{f}(\underline{x}^{[k]}) + \underline{\nabla f}|_{\underline{x}^{[k]}} \cdot (\underline{x}-\underline{x}^{[k]}) + \tfrac{1}{2}(\underline{x}-\underline{x}^{[k]})^T \underline{\underline{B}} \cdot (\underline{x} - \underline{x}^{[k]})$$

$$(\underline{x} - \underline{x}^{[k]}) = \underline{p} = -\underline{\nabla f}(||\underline{\nabla f}||^2 / \underline{\nabla f}^T \underline{\underline{B}} \underline{\nabla f}) \qquad \{\text{Cauchy}\}$$

↙ $\underline{p}^{cauchy}$

$\underline{\underline{B}}$ must be positive definite and not singular.

$$\underline{p}^{steepest\ descent} = \begin{cases} (-\underline{\nabla f}/||\underline{f}||)\Delta \leftarrow \text{max step size allowed} \\ \underline{p}^{cauchy} \end{cases} \quad \text{take min } ||\underline{p}||$$

Look at Figures 5.5 and 5.6 in BEERS

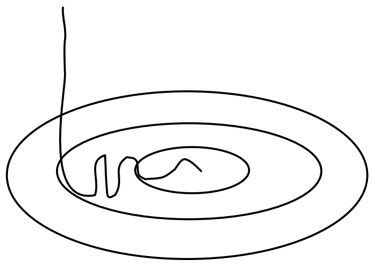

Figure 2. An example of poor scaling.

If you rescale into circles (2^{nd} derivatives similar), good scaling

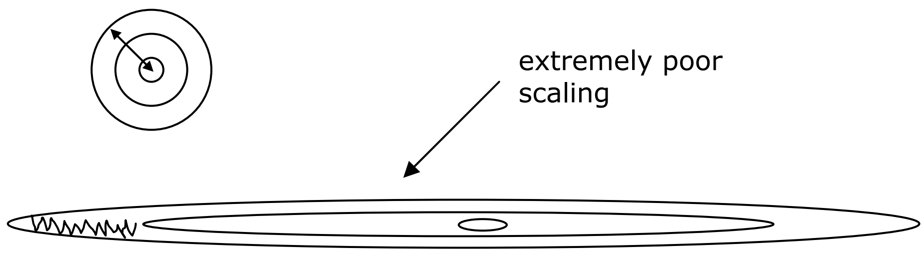

Figure 3. SCALING IS KEY.

Newton Step

If $\underline{f}_{approx}$ is correct, (guess → $\underline{p}$)

$O = \underline{\nabla} \underline{f}_{approx} = \underline{\nabla} \underline{f}_{true} + \underline{\underline{B}} \cdot \underline{p}$

$\underline{\underline{B}} \cdot \underline{p}^{newton} = \underline{\nabla} \underline{f}_{true}|_{\underline{x}^{[k]}}$ If $\underline{\underline{B}}$ is accurate and initial guess is close, converges quickly;

Similar to Newton's: ➔ $\{\underline{\underline{J}}\Delta x = \underline{F}\}$ otherwise, you may step too far

Dogleg or Trust Region Method

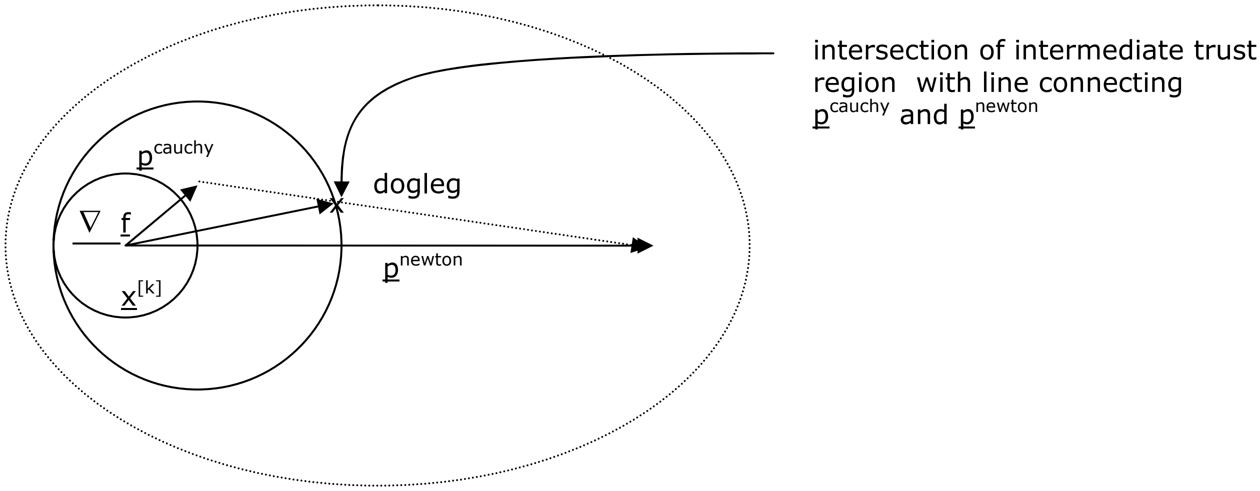

Figure 4. Diagram of dogleg method.

Broyden-Fletcher-Goldfarb-Shanno Algorithm (BFGS)

1. have $\underline{x}^{[k]} \rightarrow \underline{f}(\underline{x}^{[k]}) \rightarrow \underline{\nabla f}(\underline{x}^{[k]}) \rightarrow \underline{\underline{B}}^{[k]} \rightarrow \rightarrow \underline{p}^{old}$

$\underline{x}^{[k+1]} = \underline{x}^{[k]} + \underline{p}$

2. Compute $\underline{f}(\underline{x}^{[k+1]})$, $\underline{\nabla f}(\underline{x}^{[k+1]})$

3.

$$\underline{\underline{B}}^{[k+1]} = \underline{\underline{B}}^{[k]} + \frac{(\underline{\nabla f}(\underline{x}^{[k+1]})-\underline{\nabla f}(\underline{x}^{[k]}))(\underline{\nabla f}(\underline{x}^{[k+1]})-\underline{\nabla f}(\underline{x}^{[k]}))^T}{(\underline{\nabla f}(\underline{x}^{[k+1]})-\underline{\nabla f}(\underline{x}^{[k]}))^T\underline{p}^{old}} - \frac{(\underline{\underline{B}}^{[k]}\underline{p}^{old})(\underline{\underline{B}}^{[k]}\underline{p}^{old})^T}{(\underline{p}^{old})^T(\underline{\underline{B}}^{[k]}\underline{p}^{old})}$$

Most programs use this!

Always get symmetric matrix but sometimes eigenvalues are negative

Use $\underline{\underline{B}}_{new} = \underline{\underline{B}}^{[k+1]} + \tau\underline{\underline{I}}$ to guarantee positive eigenvalues.

In quantum mechanics, use estimates of stretching frequencies, but rest of bond angles are are set to be identity matrix.

If number of variables ($N_{variables}$) large, the total numbe of entries in $\underline{\underline{B}}$ ($N^2_{variables}$) will get too large. Can try sparse matrix storage methods.

Conjugate Gradient Method

Work like steepest descent but avoid zigzagging by forcing NEW direction to be orthogonal to OLD direction.

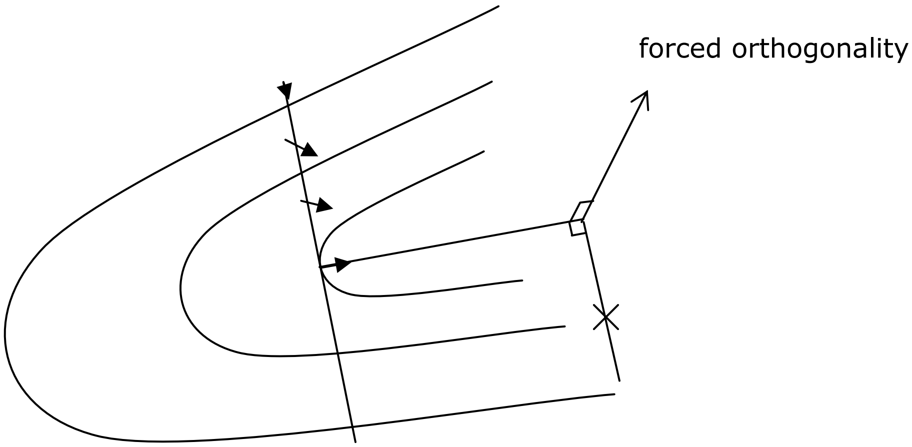

Figure 5. Conjugate gradient method.

Polak-Ribiere Formula for Step direction

(direction only) $$\underline{p}^{[k+1]} = -\underline{\nabla f}(\underline{x}^{[k+1]}) + \left[\frac{\underline{\nabla f}(\underline{x}^{[k+1]})\cdot(\underline{\nabla f}(\underline{x}^{[k+1]})-\underline{\nabla f}(\underline{x}^{[k]}))}{||(\underline{\nabla f}(\underline{x}^{[k]})||^2} \right] \underline{p}^{old,[k]}$$

↑ new

For quadratic, it takes n steps to find the minimum (n = dimension) no matter what the dimension. Use this method for LARGE SYSTEMS. The minima found are local.

* no matrices (doesn't require a lot of memory)

For 2D quadratic, gives you exact minimum direction

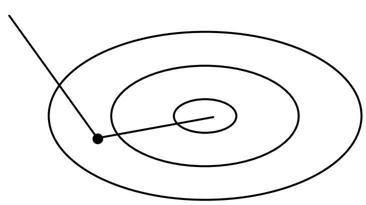

Figure 6. Diagram of search for global minimum.

must do "strong search" at each step to find absolute minimum

$\underline{\underline{B}}\cdot\underline{p}^{new} = -\underline{\nabla f}$

$\underline{f}(\underline{x}) = c + \frac{1}{2}\underline{x}^T\underline{\underline{A}}\underline{x} - \underline{b}\cdot\underline{x}$

$\underline{\nabla f} = \underline{\underline{A}}\cdot\underline{x} - \underline{b}$

$\underline{\nabla f} = 0 \rightarrow \underline{\underline{A}}\cdot\underline{x} = \underline{b}$

The Conjugate Gradient Method can solve linear systems.

$f = \frac{1}{2}\underline{x}^T\underline{\underline{B}}\underline{x} + \underline{\nabla f}\cdot\underline{x}$ ($\underline{x} = \underline{p}$) use conjugate gradient to find $\underline{p}$

$\underline{\nabla f} = \underline{\underline{B}}^{[k+1]}\ \underline{P} + \underline{\nabla f}(\underline{x}^{[k+1]})$ great for sparse matrices