Consider the unconstrained optimization problem:

\[
\text{minimize } f(x)
\]

where \( f : \mathbb{R}^n \to \mathbb{R} \) is twice continuously differentiable.

The second order necessary conditions of a minimizer are:

\[
g(x) = 0, \quad H(x) \succeq 0
\]

where \( g(x) \) and \( H(x) \) are the gradient and Hessian, respectively.

The second order sufficient conditions of a minimizer are:

\[
g(x) = 0, \quad H(x) > 0.
\]

If you don’t know the difference between these, take a moment to think about the question: How can a piece of software with access to the gradient and Hessian of a function guarantee to the user that it’s at a minimizer?

In this class, we’ll study two types of optimization algorithms:

- line search methods
- trust region methods

Both start from a given point \( x^{(0)} \) and are iterative in nature. That is, they try to find a point \( x^{(k+1)} \) “nearby” \( x^{(k)} \) such that \( f(x^{(k+1)}) < f(x^{(k)}) \). Because writing

\[
f(x^{(k+1)}), g(x^{(k)}), g(x^{(k+1)}), \text{etc.}
\]

quickly becomes tiring, we use the following shorthand at a point \( x^{(k)} \):

\[
\begin{align*}
x &= x^{(k)} \\
x^* &= x^{(k+1)} \\
g &= g(x^{(k)}) \\
g^* &= g(x^{(k+1)}) \\
H &= H(x^{(k)}) \\
f_k &= f(x^{(k)}) \\
f^* &= f(x^{(k+1)}) \\
f_{k+1} &= f(x^{(k+1)})
\end{align*}
\]

**Line search** At a point \( x \), a line search method finds a direction \( p \) that ought to improve the value of the objective function \( f \), it then considers the “line” of points:

\[
x^* = x + \alpha p.
\]

The key question with a line search method is how to pick \( p \) and \( \alpha \).

**Trust region** At a point \( x \), a trust region method fits a quadratic model around \( x \) and then minimizes a quadratic model exactly without moving too far:

\[
f(x + p) \approx f(x) + p^T g + \frac{1}{2} p^T H p
\]

with \( |p| \) not to big.

The key question with a trust region method is how to pick the model and maximum distance \( |p| \).

Try thinking about the difference this way: in a line search method, you first pick a direction, and then determine how far to go. In a trust region method, you first pick how far you are willing to go, and then pick the best direction given that distance constraint.

\[\text{FIGURE 1} \quad \text{A line search method starts at a point and picks a direction} \ p. \ \text{It then chooses a step length} \ \alpha \ \text{to determine how far to go along that direction.}\]

\[\text{FIGURE 2} \quad \text{A trust region method picks a distance} \ (\text{the two dashed red circles}). \ \text{It then tries to find the best point within the circle.} \ \text{Hopefully, once it gets close to a minimizer, it’ll pick it out directly (e.g. the larger circle).}\]