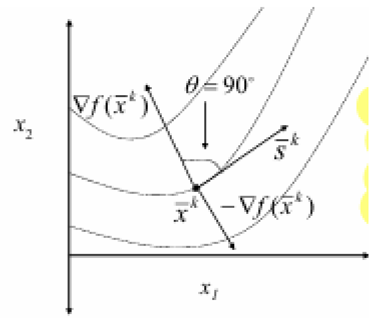
Search Directions Intuition

When iteratively searching for local minima/maxima, we know that we must move in a direction in which $f(x_{k+1}) < f(x_k)$. It is non-trivial, however, which specific direction will satisfy this equation. They key here comes in proving that any direction $\mathbf{s}$ which satisfies $\nabla f(\mathbf{x})^T\mathbf{s} < 0$ will lead to an improvement in $\mathbf{x}$.

Expanding the innerproduct $\nabla f(\mathbf{x})^T\mathbf{s}$ into its geometric representation gives

$$\nabla f(\mathbf{x})^T\mathbf{s} = |\nabla f(\mathbf{x})||\mathbf{s}|\cos(\theta)$$

If $\theta > 90°$, then $\cos(\theta) > 0$ and thus $\nabla f(\mathbf{x})^T\mathbf{s} > 0$. On the other hand, if $\theta < 90°$, then $\cos(\theta) < 0$ and thus $\nabla f(\mathbf{x})^T\mathbf{s} < 0$. The following visual helps to describe this point



From the visual above and examining the inner product, we can see that the search direction that minimizes $\nabla f(\mathbf{x})^T\mathbf{s}$ is that in which $\mathbf{s} = -\nabla f(\mathbf{x})^T$ and $\theta = 180°$.

Steepest Descent Algorithm

Using this compute step size in an interative fashion builds the foundation of the steepest descent algorithm.

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \nabla f(\mathbf{x_k})$$

The step-rate, α_k, is used to control the step size performed at each iteration. This value can be set to scalar (which may decrease robustness or convergence speed) or an exact line search can be performed similar to that in *newtonsMethodDerivation.mlx*.