1. Gradients and Hessians

The gradient of a scalar-valued function \( g: \mathbb{R}^n \rightarrow \mathbb{R} \), is the column vector of length \( n \), denoted as \( \nabla g \), containing the derivatives of components of \( g \) with respect to the variables:

\[
(\nabla g(\vec{x}))_i = \frac{\partial g}{\partial x_i}(\vec{x}), \quad i = 1, \ldots, n.
\] (1)

The Hessian of a scalar-valued function \( g: \mathbb{R}^n \rightarrow \mathbb{R} \), is the \( n \times n \) matrix, denoted as \( \nabla^2 g \), containing the second derivatives of components of \( g \) with respect to the variables:

\[
(\nabla^2 g(\vec{x}))_{ij} = \frac{\partial^2 g}{\partial x_i \partial x_j}(\vec{x}), \quad i = 1, \ldots, n, \quad j = 1, \ldots, n.
\] (2)

For the remainder of the class, we will repeatedly have to take gradients and Hessians of functions we are trying to optimize. This exercise serves as a warm up for future problems. Compute the gradients and Hessians for the following functions:

(a) Compute the gradient and Hessian (with respect to \( \vec{x} \)) for \( g(\vec{x}) = \vec{y}^T A \vec{x} \).

**Solution:** Let \( A = [\vec{a}_1 \quad \vec{a}_2 \quad \ldots \quad \vec{a}_n] \) where \( a_i \) is the \( i \)-th column of \( A \). then

\[
g(\vec{x}) = \vec{y}^T A \vec{x} = \vec{y}^T \begin{bmatrix} \vec{a}_1 & \vec{a}_2 & \ldots & \vec{a}_n \end{bmatrix} \vec{x} = \vec{y}^T (\vec{a}_1 x_1 + \vec{a}_2 x_2 + \ldots + \vec{a}_n x_n) = \sum_{i=1}^{n} (\vec{y}^T \vec{a}_i) x_i.
\] (6)

Thus

\[
\frac{\partial g}{\partial x_j}(\vec{x}) = \vec{y}^T \vec{a}_j = \vec{a}_j^T \vec{y},
\] (7)

and the gradient \( \nabla g(\vec{x}) = A^T \vec{y} \). Since the gradient does not depend on \( \vec{x} \), we then have the Hessian \( \nabla^2 g(\vec{x}) = 0 \).

(b) Compute the gradient and Hessian of \( h(\vec{x}) = \sum_{i=1}^{n} (x_i \log(x_i) - x_i) \) for \( \vec{x} \in \mathbb{R}^n_{++} \) and establish that the Hessian is positive semi-definite (as we will see soon in lecture, this establishes that \( h \) is a convex function).

**NOTE:** In fact, the Hessian is positive definite.

**Solution:** We have

\[
\frac{\partial h(\vec{x})}{\partial x_i} = \log(x_i)
\]

\[
\frac{\partial^2 h(\vec{x})}{\partial x_i^2} = 1/x_i
\]

\[
\frac{\partial^2 h(\vec{x})}{\partial x_i \partial x_j} = 0, \quad \text{for } i \neq j.
\]

Hence the \( i \)-th entry of \( \nabla h(\vec{x}) \) is \( \log(x_i) \) and the Hessian \( \nabla^2 h(\vec{x}) \) is a diagonal matrix with the \( (i, i) \)-th entry is \( 1/x_i \). As \( x_i \) are positive, so is \( 1/x_i \) and so the diagonal matrix has only positive entries, and hence has positive eigenvalues.
(c) Compute the gradient and Hessian of \( g(\vec{x}) = e^{\vec{a}^\top \vec{x} + b} \) for \( \vec{a}, \vec{x} \in \mathbb{R}^n, b \in \mathbb{R} \) and establish that the Hessian is positive semi-definite.

**Solution:** We can either compute the gradient and Hessian directly or we can use the properties of gradient and Hessians under composition with linear functions.

We will first see the former.

\[
\frac{\partial g(\vec{x})}{\partial x_i} = e^{\vec{a}^\top \vec{x} + b} a_i \\
\frac{\partial^2 g(\vec{x})}{\partial x_i^2} = e^{\vec{a}^\top \vec{x} + b} a_i^2 \\
\frac{\partial^2 g(\vec{x})}{\partial x_i \partial x_j} = e^{\vec{a}^\top \vec{x} + b} a_i a_j
\]

Writing these in matrix form, we get,

\[
\nabla g(\vec{x}) = e^{\vec{a}^\top \vec{x} + b} \vec{a} \\
\nabla^2 g(\vec{x}) = e^{\vec{a}^\top \vec{x} + b} \vec{a} \vec{a}^\top
\]

The Hessian is clearly a rank one positive semi-definite matrix.

To see the second way, we notice that considering \( e(x) = e^x \) for a scalar \( x \), the derivative and second derivative of \( e(x) \) is just \( e^x \). Since the linear transform we are taking is \( \vec{a}^\top \vec{x} + b \), we get the same result.

### 2. Jacobians

The **Jacobian** of a vector-valued function \( \vec{g}: \mathbb{R}^n \rightarrow \mathbb{R}^m \) is the \( m \times n \) matrix, denoted as \( D\vec{g} \), containing the derivatives of the components of \( \vec{g} \) with respect to the variables:

\[
(D\vec{g})_{ij} = \frac{\partial g_i}{\partial x_j}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n. \tag{8}
\]

Compute the Jacobian of \( \vec{g}: \mathbb{R}^n \rightarrow \mathbb{R}^n \), where

\[
g \left( \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \right) = \frac{1}{2} \begin{bmatrix} x_1^2 \\ \vdots \\ x_n^2 \end{bmatrix}. \tag{9}
\]

**Solution:** Notice that

\[
g_i(\vec{x}) = \frac{1}{2} x_i^2, \quad \text{so} \quad \frac{\partial g_i}{\partial x_j}(\vec{x}) = \begin{cases} x_i & i = j \\ 0 & i \neq j \end{cases}. \tag{10}
\]

Thus \( D\vec{g}(\vec{x}) = \begin{bmatrix} x_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & x_n \end{bmatrix} = \text{diag}(\vec{x}) \) where \( \text{diag}(\vec{x}) \in \mathbb{R}^{n \times n} \) is the diagonal matrix whose diagonal entries are the entries of \( \vec{x} \).
3. Gradient of the Cross Entropy Loss

Consider the data \((\vec{x}_i, y_i)\) for \(i = 1, \ldots, n\) where \(\vec{x} \in \mathbb{R}^d\) and \(y_i \in \{0, 1\}\). Consider the parameter vector \(\vec{w} \in \mathbb{R}^d\). For each \(i \in \{1, \ldots, n\}\), define the logistic function \(p_i : \mathbb{R}^d \mapsto \mathbb{R}\) given as

\[
p_i(\vec{w}) = \frac{1}{1 + e^{-\vec{w}^\top \vec{x}_i}}.
\]  

(a) Find the gradient of the function \(p_i(\vec{w})\).

**Solution:** The gradient is

\[
\nabla p_i(\vec{w}) = \begin{bmatrix}
\frac{\partial p_i}{\partial w_1}(\vec{w}) \\
\vdots \\
\frac{\partial p_i}{\partial w_d}(\vec{w})
\end{bmatrix}
\]

Here

\[
\frac{\partial p_i}{\partial w_j}(\vec{w}) = \frac{(\vec{x}_i)_j e^{-\vec{w}^\top \vec{x}_i}}{(1 + e^{-\vec{w}^\top \vec{x}_i})^2}.
\]

Thus

\[
\nabla p_i(\vec{w}) = \vec{x}_i \cdot \frac{e^{-\vec{w}^\top \vec{x}_i}}{(1 + e^{-\vec{w}^\top \vec{x}_i})^2}
\]

(b) For \(i \in \{1, \ldots, n\}\), the cross entropy of \(p \in [0, 1]\) against \(y_i\) is defined as

\[
H_i(p) \equiv -y_i \log(p) - (1 - y_i) \log(1 - p).
\]  

Find the gradient of the function \(\ell_i(\vec{w}) = H_i(p_i(\vec{w}))\) with respect to \(\vec{w}\).

**Solution:** The gradient is

\[
\nabla_{\vec{w}} \ell_i(\vec{w}) = \begin{bmatrix}
\frac{\partial \ell_i}{\partial w_1}(\vec{w}) \\
\vdots \\
\frac{\partial \ell_i}{\partial w_d}(\vec{w})
\end{bmatrix}
\]

We can use the chain rule to find each component:

\[
\frac{\partial \ell_i}{\partial w_j}(\vec{w}) = -\left[ \frac{\partial H_i}{\partial p}(p_i(\vec{w})) \right] \left[ \frac{\partial p_i}{\partial w_j}(\vec{w}) \right]
\]

\[
= -\left[ y_i - \frac{1 - y_i}{p_i(\vec{w})} \right] \frac{(\vec{x}_i)_j e^{-\vec{w}^\top \vec{x}_i}}{(1 + e^{-\vec{w}^\top \vec{x}_i})^2}
\]

\[
= -\left[ \frac{y_i}{1/(1 + e^{-\vec{w}^\top \vec{x}_i})} - \frac{1 - y_i}{e^{-\vec{w}^\top \vec{x}_i}/(1 + e^{-\vec{w}^\top \vec{x}_i})} \right] \frac{(\vec{x}_i)_j e^{-\vec{w}^\top \vec{x}_i}}{(1 + e^{-\vec{w}^\top \vec{x}_i})^2}
\]

\[
= -\left[ y_i(1 + e^{-\vec{w}^\top \vec{x}_i}) - \frac{(1 - y_i)(1 + e^{-\vec{w}^\top \vec{x}_i})}{e^{-\vec{w}^\top \vec{x}_i}} \right] \frac{(\vec{x}_i)_j e^{-\vec{w}^\top \vec{x}_i}}{(1 + e^{-\vec{w}^\top \vec{x}_i})^2}
\]

\[
= -(\vec{x}_i)_j \left[ y_i e^{-\vec{w}^\top \vec{x}_i} - (1 - y_i) \frac{1}{1 + e^{-\vec{w}^\top \vec{x}_i}} \right]
\]

\[
= -(\vec{x}_i)_j \left[ y_i(1 - p_i(\vec{w})) - (1 - y_i)p_i(\vec{w}) \right]
\]

\[
= -(\vec{x}_i)_j \left[ y_i - p_i(\vec{w}) \right]
\]

\[
= (\vec{x}_i)_j \left[ p_i(\vec{w}) - y_i \right].
\]
Thus
\[ \nabla \ell_i(\vec{w}) = \vec{x}_i \left[ p_i(\vec{w}) - y_i \right]. \quad (25) \]

(c) Define the cross-entropy loss function as the sum of the cross entropy functions over the entire data set:
\[ \ell(\vec{w}) = \sum_{i=1}^{n} \ell_i(\vec{w}). \quad (26) \]

Find the gradient of the function \( \ell(\vec{w}) \).

**Solution:** Using linearity of the derivatives,
\[ \nabla \ell(\vec{w}) = \sum_{i=1}^{n} \nabla \ell_i(\vec{w}) \]
\[ = \sum_{i=1}^{n} \vec{x}_i \cdot (p_i(\vec{w}) - y_i) \]
\[ = X^\top (\vec{p}(\vec{w}) - \vec{y}). \quad (29) \]

Here
\[ X = \begin{bmatrix} \vec{x}_1^\top \\ \vdots \\ \vec{x}_n^\top \end{bmatrix}, \quad \vec{p}(\vec{w}) = \begin{bmatrix} p_1(\vec{w}) \\ \vdots \\ p_n(\vec{w}) \end{bmatrix}, \quad \vec{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \]

Notice that this is the same type of gradient as least squares! All it requires is replacing our linear predictors \( X \vec{w} \) with our logistic predictors \( \vec{p}(\vec{w}) \).