# Q1

**(a)**

**i. Two assumptions of convolutional networks**

1) Locality, which means that the values of two inputs are more correlated if the two inputs are close to each other, whereas the values of two inputs are uncorrelated if they are far away. In other words, locality means patterns appear locally in the domain of the input signals.

2) Stationarity, which means that a pattern can appear multiple times and it can appear at any location in the input domain.

**ii. Advantages when the assumptions are true**

1) Locality leads to sparsity of the weight matrix because we only need to check local regions for a feature/pattern instead of having to check the entire domain of the signal.

2) Stability leads to parameter sharing since we are checking for the same pattern.

**iii. Example of a data set where the assumptions would not hold**

The "Car Evaluation Data Set" from UCI Machine Learning Repository. Each input sample is a car. The features/attributes are the descriptions of this particular car, such as buying price, number of doors, and person capacity.

Actually, I think the assumptions do not hold for most (if not all) input samples (as opposed to input signals). I can't think of any data sets with input samples where the locality and stationary assumptions would hold.

**(b)**

$$\frac{\partial z[k]}{\partial x[i]} = y[(i+k) \bmod n]$$

$$\frac{\partial z[k]}{\partial y[i]} = x[(i-k) \bmod n]$$

**(c)**

**i. Dimensionality of the output space**

$$f(\cdot) \in R^{1000} \qquad g(\cdot) \in R^{10\times 100}$$

**ii. Number of trainable parameters**

$f(\cdot)$ has $1000 \times 100 = 10^5$ parameters. $g(\cdot)$ has $3 \times 10 = 30$ parameters.

**iii. Computational complexity of forward pass (number of operations)**

Ignoring the point wise non-linearity.

The linear part of $f(\cdot)$ has $(100 + 99) \times 1000 = 199000$ operations, because it has 1000 dot products and each dot product has 100 multiplications and 99 summations.

The linear part of $g(\cdot)$ has $(3 + 2) \times 100 \times 10 = 5000$ operations, because it has 10 filters, each filter is applied 100 times, and each time contains 3 multiplications and 2 summations since the kernel size is 3.