1. One other variation of Gradient Descent, and why it is used?

Other variants of gradient descent include stochastic gradient descent (SGD) and small batch gradient descent. In SGD, instead of computing the gradient for the entire dataset, the gradient is only computed for a randomly selected subset of the training set (or individual data points) in each iteration. This makes the computation faster as it avoids the need to compute the gradient for the entire dataset, which can be computationally costly for large datasets. Small batch gradient descent, on the other hand, is somewhere between batch gradient descent and SGD, as it computes the gradients of a small number of random data points. This can lead to better convergence rates than random gradient descent, while still being computationally efficient.

Both SGD and small batch gradient descent are useful when dealing with large data sets or when the goal is to achieve faster convergence. However, they introduce additional noise into the gradient estimation, which can lead to less accurate parameter estimates.

2. Why convergence to a Loss Function minimum may be difficult to obtain in practice, and how this might be overcome

Convergence to a loss function minimum may be difficult to obtain in practice due to various reasons such as noisy or inconsistent data, overfitting, poor initialisation of model parameters etc.

The solution to this problem can be the use of regularisation techniques such as L1 or L2 regularisation, which can help prevent over-fitting and improve the generalisation performance of the model. Regularisation penalises large weights and encourages the model to select only the most relevant features for the prediction task.

Another approach is to use more sophisticated optimisation algorithms such as Adam, Adagrad or RMSprop. These optimisation algorithms adaptively adjust the learning rate according to the gradient of the loss function, which can help improve the convergence rate and avoid getting stuck in local minima.