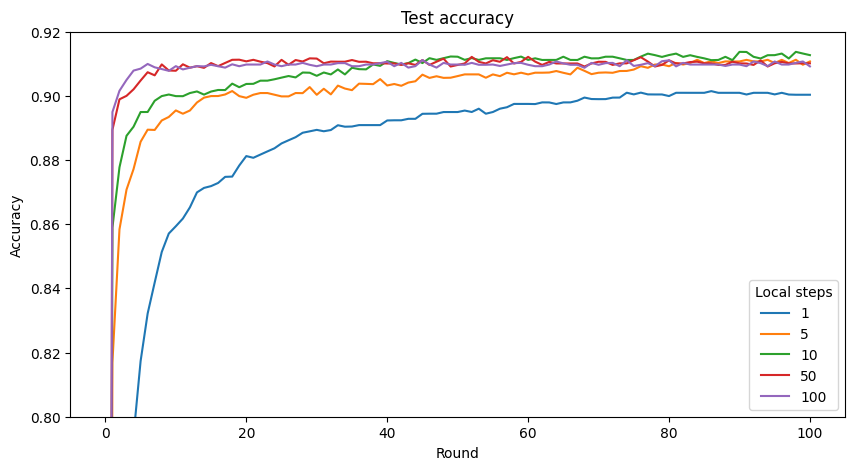
**Recap of Exercise 3**



The global model converges faster with high numbers of local steps, in particular, 50 and 100, and reaches the good accuracy.

5-10 local steps require more time to converge but the global can achieve the good accuracy as 50-100 steps after 100 rounds. The model achieves the highest accuracy with 10 local steps.

1 local step is very slow to converge, the model reaches the worst accuracy.

I was expecting the highest performance at 10 number of local steps as the trade-off between number of local steps and global model performance.

**Data:**

Explanation of `iid\_divide` function (data/utils.py):

* Divide the list `l` into `g` groups with little different in size if the number of samples in `l` cannot be divided by `g`.
* #samples of big group = #samples of small group + 1

In TP1, the generated data is iid. Steps:

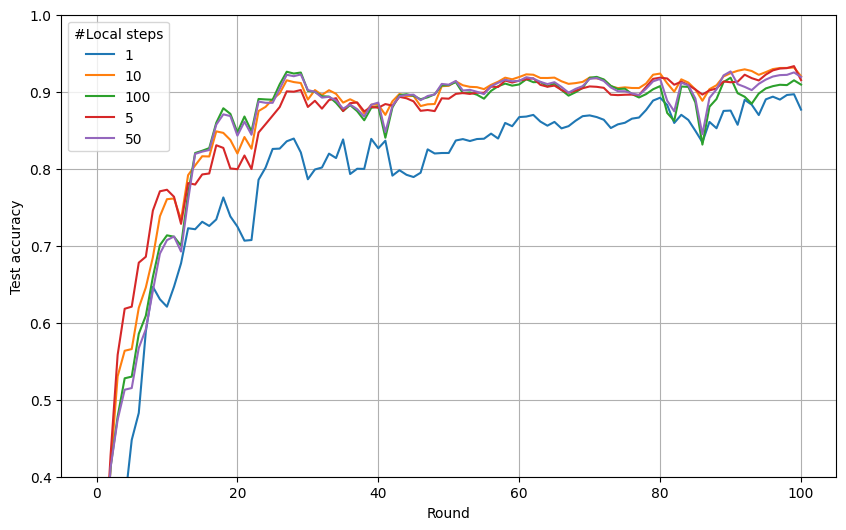
1. In `iid\_split` function, we take `n\_samples` by a fraction of the dataset. Then, use uniform random to choose `n\_samples` indices from 1 to `len(dataset)` with no replacement.
2. Shuffle the chosen indices and use `iid\_divide` to divide into `n\_clients` groups.

For non-iid split (in TP2), steps:

1. In `pathlogical\_non\_iid\_split` function, we sort the samples in the dataset by their classes, then, divide them into shards. Every shard has almost the same of number of samples after using `iid\_divide` function on shards.
2. Shuffle the shards and assign them to clients. This ensures that each client has a limited number of classes.

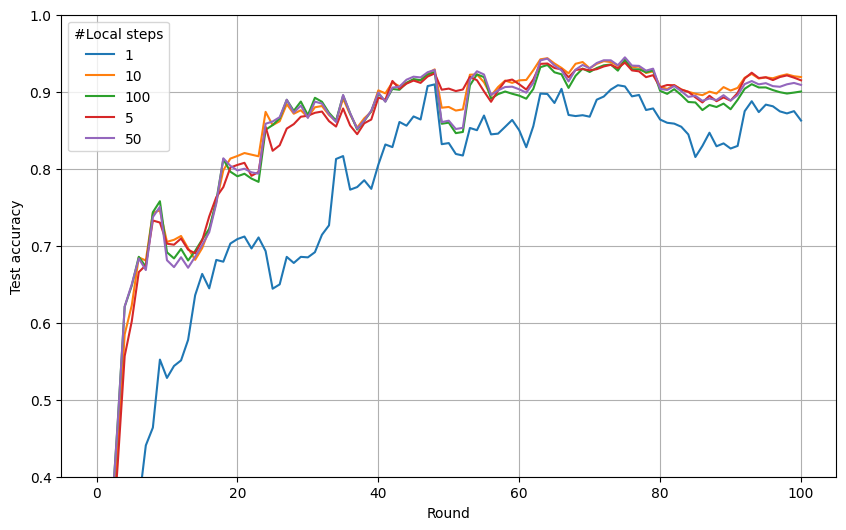
Lastly, each client has its own dataset (iid or non-iid), saved in .npy files based on the sample indices.

**Exercise 5.1:**



*Figure: Test accuracy of global model with different number of local steps after 100 rounds*

**Exercise 5.2:**



*Figure: Test accuracy of global model with different number of local steps after 100 rounds*