# Local DP

**Algorithm 2** Local Differential Privacy in Federated Learning

**procedure** MAIN ▷ Executed at the server side

    Initialize: model $\theta_0$

    **for** each round $r = 1, 2, \ldots$ **do**

        $K_r \leftarrow$ randomly select $K$ participants

        **for** each participant $k \in K_r$ **do**

            $\theta_r^k \leftarrow$ DP-SGD ▷ This is done in parallel

        $\theta_r \leftarrow \Sigma_{i=1}^{K_r} \frac{n^k}{n} \theta_r^k$ ▷ $n^k$ is the size of dataset at participant $k$

**function** DP-SGD(Clipping norm $C$, dataset $D$,sampling probability $p$ noise magnitude $\sigma$, learning rate $\eta$, Iterations $E$, loss function $L(\theta(x), y)$)

    Initialize $\theta_0$

    **for** each local epoch $i$ from 1 to E **do**

        **for** $(x, y) \in$ random *batch* from dataset $D$ with probability $p$ **do**

            $g_i = \nabla_\theta L(\theta_i; (x, y))$

            $Temp = \frac{1}{pD} \Sigma_{i \in batch} g_i min(1, \frac{C}{\|g_i\|_2}) + N(0, \sigma^2 I)$

            $\theta_{i+1} = \theta_i - \eta(Temp)$

    return $\theta_E$

# Central DP

**Algorithm 1** Central Differential Privacy in Federated Learning

**procedure** MAIN ▷ Executed at the server side
  Initialize: model $\theta_0$, Moment_Accountant($\epsilon$, N) ▷ N is number of all participants
  **for** each round $r = 1, 2, \ldots$ **do**
    $C_r \leftarrow$ randomly select participants with probability q
    $p_r \leftarrow$Moment_Accountant.get_privacy_spent() ▷ It returns the spect privacy budget for current round
    **if** $p_r > T$ **then** ▷ If spent privacy budget is greater than threshold, return current model
      return $\theta_r$
    **for** each participant $k \in C_r$ **do**
      $\Delta_k^{r+1} \leftarrow$PARTICIPANT_UPDATE($k, \theta_r$) ▷ This is done in parallel
    $S \leftarrow bound$
    $z \leftarrow noisescale$
    $\sigma \leftarrow zS/q$
    $\theta_{r+1} \leftarrow \theta_r + \Sigma_{i=1}^{C_r} \Delta_i^{r+1} / C_r + N(0, I\sigma^2)$
    Moment_Accountant.accumulate_spent_privacy($z$)
**function** PARTICIPANT_UPDATE($k, \theta_r$)
  $\theta \leftarrow \theta_r$
  **for** each local epoch $i$ from 1 to E **do**
    **for** batch $b \in B$ **do**
      $\theta \leftarrow \theta - \eta \nabla L(w; b)$
      $\Delta \leftarrow \theta - \theta_r$
      $\theta \leftarrow \theta_0 + \Delta \min(1, \frac{S}{\|\Delta\|_2})$
  return $\theta - \theta_r$ ▷ This one is already clipped