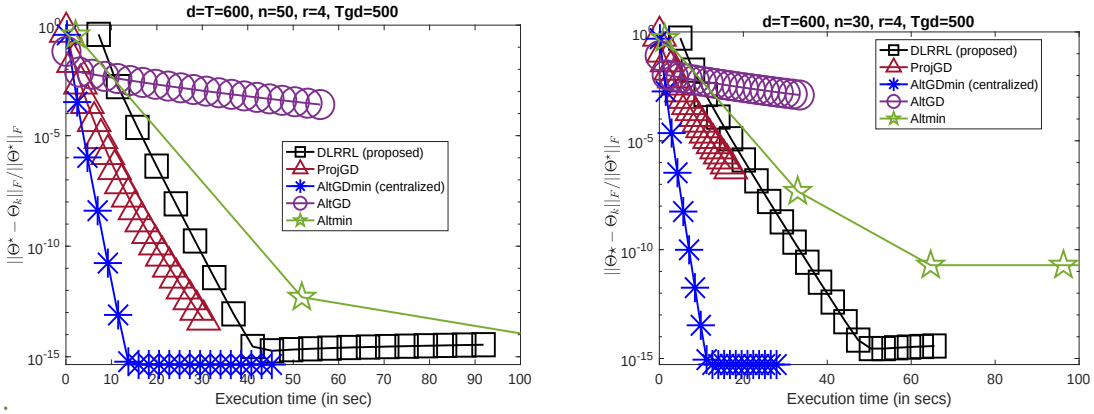
B. Additional Simulation Results

Comparison with centralized algorithms: We compared the performance of the proposed DLRRL algorithm with that of three other existing algorithms that are widely studied in the LR literature, AltGDmin (Nayer & Vaswani, 2023), Altmin (Netrapalli et al., 2013), AltGD (Yi et al., 2016b), and ProjGD (Jain & Netrapalli, 2015). The plots are presented on Figure 3. It is worth noting that in this comparison, DLRRL is the only decentralized algorithm, while the rest fall under the category of centralized algorithms. We set $d = T = 600$, $r = 4$, network connectivity probability $p = 0.5$, and number of agents is set as $L = 20$. We chose different values of n in our experiments to evaluate the performance of the different algorithms based on the extend of compression the data undergoes. In Figures 3a and 3b, we varied the value of n as $50, 30$, respectively. From Figures 3a and 3b we see that as the rate of compression increases (i.e., n/d decreases), the ProjGD, AltGD algorithms no longer converge. The AltGDmin and DLRRL still converge for these highly compressed cases, thereby validating the sample efficiency of these approaches over the other three. Additionally, we notice from the plots that the Altmin algorithm is computationally intensive. Another observation is that the error decay is not monotonic in the figures. This is because we use the same value of consensus iterations in our experiments. Although the consensus error will be below ϵ_{con}, the exact error value in the different GD iterations, will be different. During certain iterations of GD, while computing the gradient through average consensus, the consensus error may be lower than in previous steps.



(a) Error-X vs. Execution time: $n = 50$

(b) Error-X vs. Execution time: $n = 30$

Figure 3. Error versus execution time plot with time in seconds. In both figures $d = T = 600$, $r = 4$, $L = 20$, and the network was generated as an ER graph with $p = 0.5$ probability of nodes being connected. We compared the performance of our proposed algorithm (DLRRL) with the existing centralized counterparts from (Nayer & Vaswani, 2023) and with the LR matrix recovery algorithms, Altmin (Netrapalli et al., 2013), ProjGD (Jain & Netrapalli, 2015), and AltGD (Yi et al., 2016a) algorithms. The network was generated as an ER graph with $p = 0.5$ probability. In Figures 3a, 3b, we present the plots for error vs. execution time for $n = 50, 30$, respectively. For improved visualization, we have set the x-limit to 100 seconds. This choice aligns with the runtime of all algorithms except Altmin, which exceeds 1000 seconds.