Extending the Deep Conjugate Direction Method to the BiCG Algorithm

Arihant Rastogi Hrishiraj Mitra Prasoon Dev

Phase Two Evaluation, April 2025

Project Overview

- **Objective:** Extend the Deep Conjugate Direction Method (DeepCD) to the Bi-Conjugate Gradient (BiCG) algorithm for solving nonsymmetric linear systems.
- **Reference Implementation:** Based on the repository ayano721/2023_DCDM and the paper by Kaneda et al. [1].

Challenges in Running DeepCD Code

- **High RAM Requirements:** Attempted to run the original DeepCD code but encountered excessive RAM usage.
- **GPU Limitations:** Even with reduced batch sizes, the GPU requirements remained prohibitive.

Strategies Employed

- **Reduced Matrix Dimensions:** Trained on 64-dimensional matrices to lower computational demands.
- **Fixed-Point Precision:** Experimented with lower precision to decrease memory usage.
- **Batch Size Reduction:** Attempted smaller batch sizes, but GPU constraints persisted.

Bi-Conjugate Gradient (BiCG) Algorithm

Overview:

- BiCG is an iterative method for solving nonsymmetric linear systems.
- It generates two sequences of vectors, one for the original system and one for the transpose system.

Pseudocode:

1. Initialize x_0, compute $r_0 = b - Ax_0$, choose $\hat{r}_0$ such that $(\hat{r}_0, r_0) \neq 0$.
2. Set $p_0 = r_0$, $\hat{p}_0 = \hat{r}_0$.
3. For $k = 0, 1, 2, \ldots$ until convergence:
   3.1 $\alpha_k = \frac{(r_k, \hat{r}_k)}{(Ap_k, \hat{p}_k)}$.
   3.2 $x_{k+1} = x_k + \alpha_k p_k$.
   3.3 $r_{k+1} = r_k - \alpha_k A p_k$.
   3.4 $\hat{r}_{k+1} = \hat{r}_k - \alpha_k A^T \hat{p}_k$.
   3.5 If r_{k+1} is sufficiently small, then exit.
   3.6 Use neural network F_θ to predict new search directions:
      - $p_{k+1}, \hat{p}_{k+1} = F_\theta(r_{k+1}, \hat{r}_{k+1}, p_k, \hat{p}_k)$.

Proposed Framework for Deep BiCG

Conceptual Modifications:

- Integrate neural network-based direction predictions within BiCG iterations.
- Adapt DeepCD's learned updates to BiCG's bidirectional structure.

Algorithmic Adjustments:

- Train a neural network to predict optimal search directions using historical residuals and directions from both the original and transpose systems.
- Ensure bi-orthogonality of the generated directions.

Compute Resource Planning:

- Outline resource requirements to ensure readiness when adequate computational power becomes available.
- Plan for distributed computing or cloud-based solutions to handle increased computational demands.

Next Steps

- **Resource Acquisition:** Seek access to higher-capacity GPUs and increased RAM.
- **Collaborative Development:** Work with team members to implement and test the BiCG extension.
- **Evaluation and Benchmarking:** Compare performance against existing methods on benchmark problems.

References

Ayano Kaneda, Osman Akar, Jingyu Chen, Victoria Alicia Trevino Kala, David Hyde, and Joseph Teran.
A deep conjugate direction method for iteratively solving linear systems.
In *Proceedings of the 40th International Conference on Machine Learning*, pages 15720–15736, 2023.