UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Deep Reinforcement Learning for Adaptive Mesh Refinement

Corbin Foucart, Aaron Charous, Pierre F.J. Lermusiaux

John Sunny George

04/25/2023

Introduction

Adaptive Mesh Refinement (AMR)

Preferentially resolve regions containing important features during simulation.

- Advantages
  - Solves problems intractable on uniform mesh.
- Disadvantages
  - Heuristic
  - Relies on domain-specific knowledge and uses trial-and-error.
  - Best practices are not universally agreed upon.

Deep Reinforcement Learning Adaptive Mesh Refinement (DRL-AMR)

Formulate AMR as POMDP (local, sequential decision-making problem under incomplete information)

- No use of exact solution.
- No use of high fidelity ground truth.
- No use of precomputed training dataset.
- Local nature allows faster training on smaller problems.

Problem Formulation

Formulate AMR as POMDP

POMDP is characterized by

$(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \mathcal{O}, \mathcal{O}, \gamma)$

- $\mathcal{S}$ is a set of states
- $\mathcal{A}$ is a set of actions
- $\mathcal{T}$ (s, a, s') $\rightarrow$ [0, 1] state transition function
- $\mathcal{R}$ is the reward function
- $\mathcal{O}$ is a set of observations
- $\mathcal{O}$ (o|s, a) is a set of conditional observation probabilities
- γ is the discount factor.

Agent does not have access to the complete state $\mathcal{S}_t$, but an indirect observation.

During training,

- Agent observes a single element locally.
- Takes action on that cell alone.
- Receives a reward for the effect of its local action on the entire computational domain.

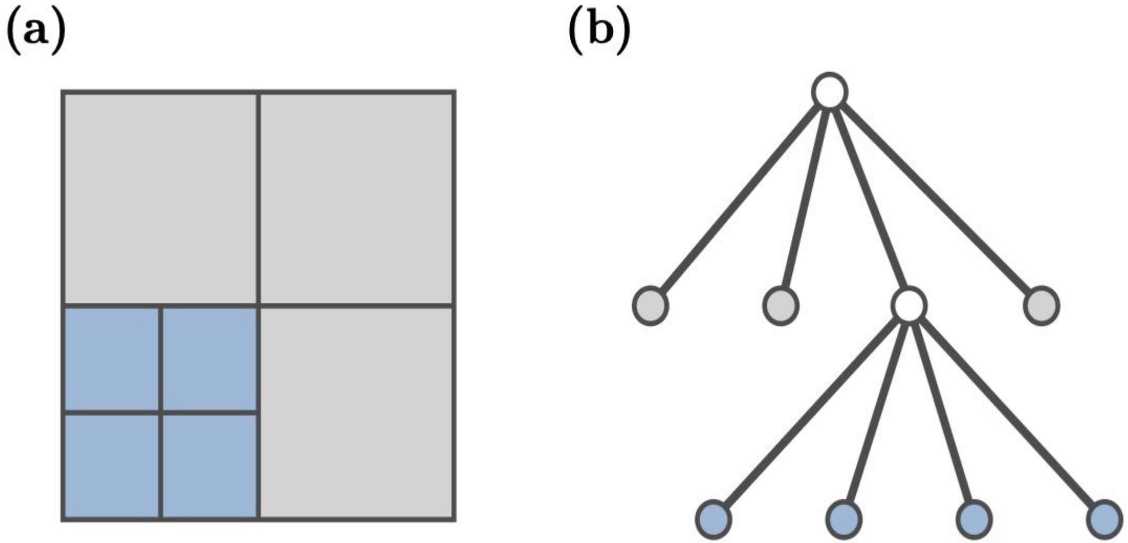
What are the actions, observations, and reward function?

Problem Formulation

Action Space

- Refine
- Coarsen
- Do nothing

Mesh is represented using a tree data structure.
Active elements are the leaves of the tree.
Coarsening is possible only if all the sibling nodes are active.
If coarsening is not possible, action defaults to "do nothing".



Quadtree data structure for 2D mesh.[1]

Problem Formulation

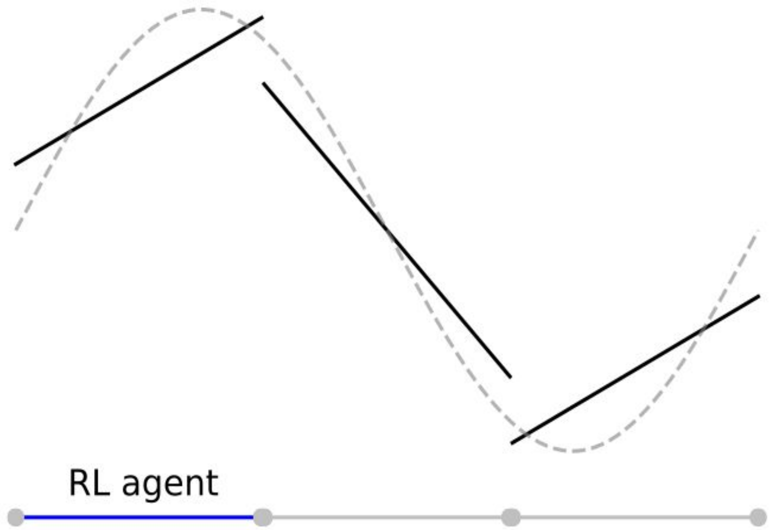
Observation Space

Solve the PDE using Discontinuous Galerkin Finite Element Method (DGFEM).

The observation space on an element of the computational mesh consists of:

- The jumps in the numerical solution integrated over the boundary of the cell, as well as for the cell neighbors. $\Xi_K = \int_{\partial K} [\![u_h]\!] \, d\partial K$
- The average integrated jump across all mesh elements. $\sum_K \Xi_K / N_K$
- The current usage of available computational resources, p.
- Any physically relevant features of the numerical solution or data local to the cell.

For choosing a cell, a uniform distribution is assumed over all cells in a mesh.



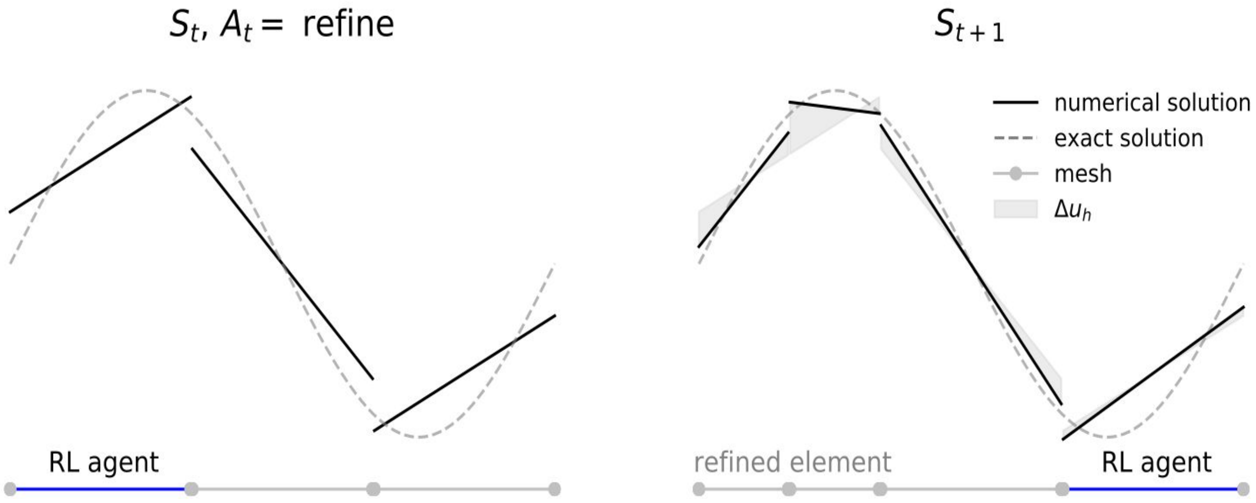
DGFEM solution using linear element[1]

Problem Formulation

Reward Function

- A positive reward when refining.
- A negative reward when coarsening.
- Zero reward for doing nothing.
- Large negative reward for exceeding computational resources.
- Scaling factor, γ_c ,determines the penalty on using computational resources.

$$\Delta u_h = \sum_{K \in \mathcal{T}_h} \int_K |u_h^{t+1} - u_h^t| \, dK$$



Agent refining an element.[1]

$$R_{\Delta C} = B\left(p^{t+1}\right) - B\left(p^t\right) \qquad B(p) = \sqrt{p}/(1-p).$$

$$R(s_t, a_t) = \begin{cases} +\left[\log(\Delta u_h + \epsilon_{\text{machine}}) - \log(\epsilon_{\text{machine}})\right] - \gamma_c R_{\Delta C}, & \text{if } a_t = \texttt{refine} \\ -\left[\log(\Delta u_h + \epsilon_{\text{machine}}) - \log(\epsilon_{\text{machine}})\right] - \gamma_c R_{\Delta C}, & \text{if } a_t = \texttt{coarsen} \\ 0, & \text{if } a_t = \texttt{do nothing} \end{cases}$$

Training

$$Q_t(s,a) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1} \mid S_t = s,\ A_t = a\right]$$

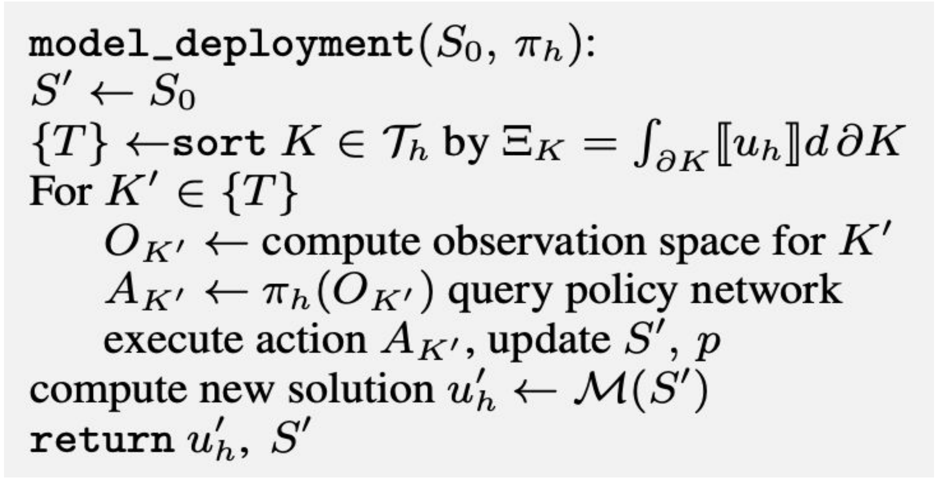
- Learn the Q-function using a Deep Q Network.
- 2 hidden layers, relu activation,100 - 200 iteration per episode, 2000 episodes.
- Early termination possible when agent chooses to do nothing consecutively.

```
training_step($A_t$, $O_t$):
$K_t \leftarrow$ cell corresponding to $O_t$
if $A_t$ = refine
    $S_{t+1} \leftarrow$ refine $K_t$
if $A_t$ = coarsen
    $S_{t+1} \leftarrow$ coarsen $K_t$
update $u_h^{t+1} = \mathcal{M}(S_{t+1})$,  $p_{t+1}$
compute $\Delta u_h$ (on finer mesh) via (2), $R_{\Delta C}$ via (3)
compute $R_{t+1}$ via (4)
sample cell $K_{t+1} \leftarrow B_t(O_t, S_{t+1})$
$O_{t+1} \leftarrow$ compute observation space for $K_{t+1}$
if $p_{t+1} > 1$
    $R_{t+1} \leftarrow$ large, negative reward
    done $\leftarrow$ true
if iteration $\geq$ episode iterations
    done $\leftarrow$ true
return $R_{t+1}$, done, $O_{t+1}$
```

Model Deployment

Given a trained policy, a mesh and a solution,

- Sort elements in the mesh based on jump in numerical solution integrated over boundary of the cell. Loop:
  - Compute observation space for each element.
  - Execute the action queried from the network on each element while updating the mesh and resource used. End loop.
- Compute new solution on the new mesh.
- Repeat the process if necessary.



```
model_deployment($S_0$, $\pi_h$):
$S' \leftarrow S_0$
$\{T\} \leftarrow$ sort $K \in \mathcal{T}_h$ by $\Xi_K = \int_{\partial K} [\![ u_h ]\!] d\,\partial K$
For $K' \in \{T\}$
    $O_{K'} \leftarrow$ compute observation space for $K'$
    $A_{K'} \leftarrow \pi_h(O_{K'})$ query policy network
    execute action $A_{K'}$, update $S'$, $p$
compute new solution $u_h' \leftarrow \mathcal{M}(S')$
return $u_h'$, $S'$
```

Experiments

Training

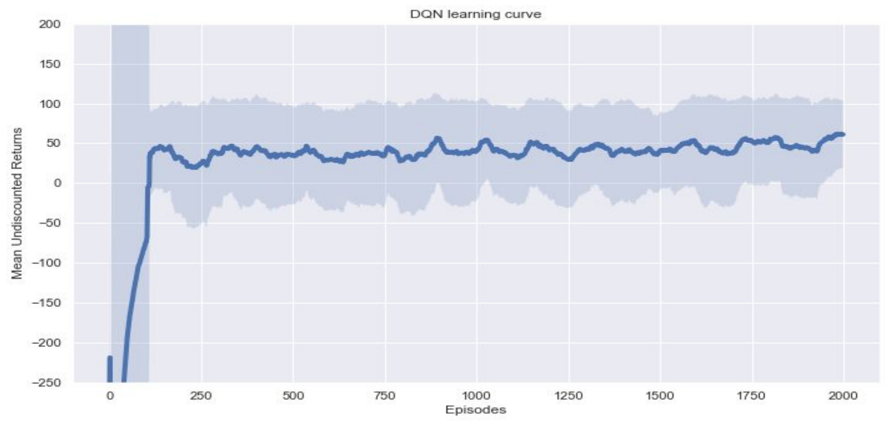
- 1D linear advection equation:

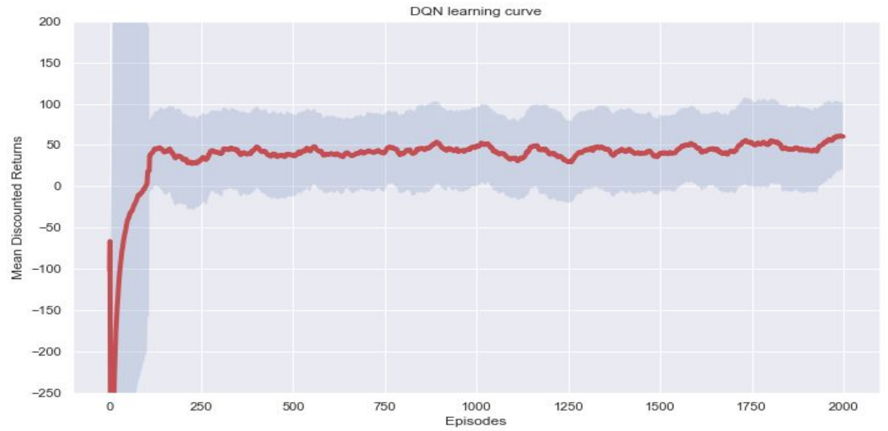
$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0$$

- Computational domain, x ∈ [0, 8]
- Periodic boundary condition.
- Initial condition:

$$u(x,0) = exp(\frac{-(x-\mu)^2}{2\sigma^2})$$

  Mean = 4, Standard deviation = 0.25
- Initial grid contains 4 elements.
- Solver : Discontinuous Galerkin Finite Element Method
- Single time step 1e-5.
- Cell budget: 25
- Scaling factor: 100





Results

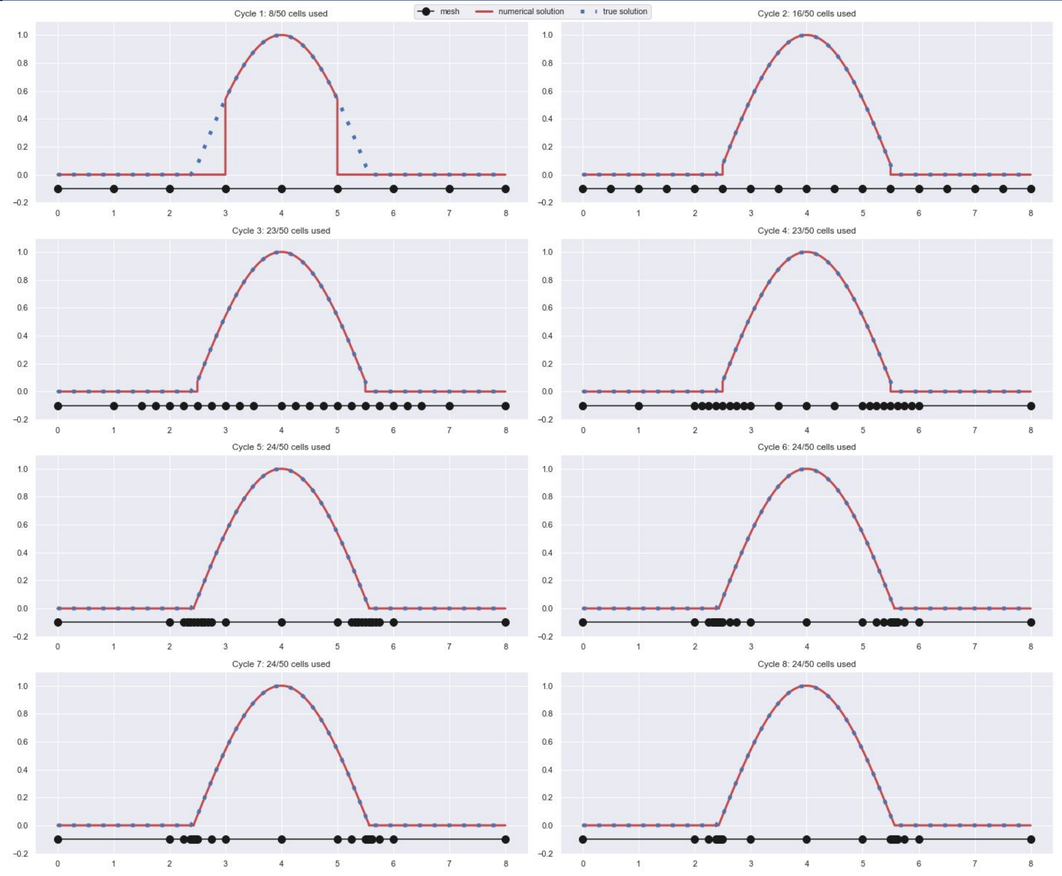
- 1D linear advection equation:

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0$$

- Computational domain, x ∈ [0, 8]
- Periodic boundary condition.
- Initial condition:

$$u(x,0) = \begin{cases} \sin(x - 4 + \pi/2) & 4 - \pi/2 \leq x \leq 4 + \pi/2 \\ 0 & else \end{cases}$$

- Initial grid contains 4 elements.
- Cell budget: 50
- 8 cycles of refinement.
- Initial refine cycles distributes elements.
- Later cycles changes the mesh topology.



Conclusions

Summary

- Formulate Adaptive Mesh Reinforcement (AMR) as POMDP.
- Train the RL agent using a Deep-Q Network to improve mesh topology to solve a PDE, numerically.
- Deploy the trained agent for sufficient cycles, on a test problem to generate the adapted mesh.

Questions?

References

[1] Foucart, C., Charous, A., & Lermusiaux, P.F. (2022). Deep Reinforcement Learning for Adaptive Mesh Refinement. *ArXiv, abs/2209.12351*.https://doi.org/10.48550/arXiv.2209.12351

The Grainger College of Engineering

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN