OPM
OPEN POROUS MEDIA

From block-shaped to flexible LGRs

Antonella Ritorto

May 27, 2025

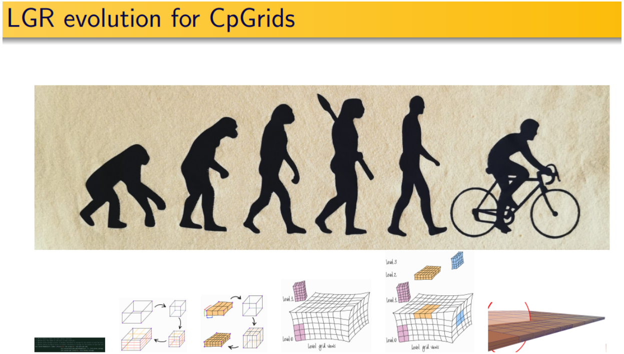
Local Grid Refinement (LGR)

Cooperation between

- **OPM-OP**
  - Blatt, Markus
  - Ritorto, Antonella
- **SINTEF**
  - Skaflestad, Bård
- **TNO**
  - Barros, Eduardo
  - Castiel, Artur
  - Khoshnevis, Negar

Flashback

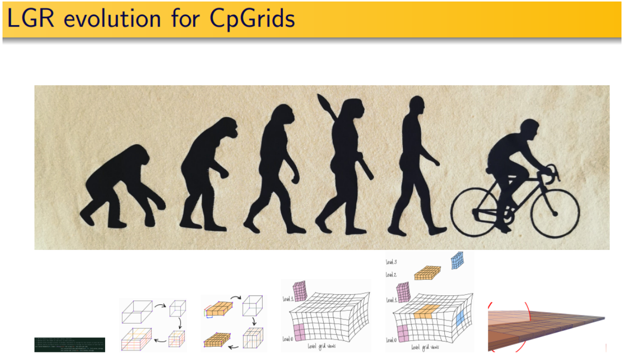
Last OPM Summit - April 2024



LGR evolution for CpGrids

- Parent cells sets
  - block-shaped

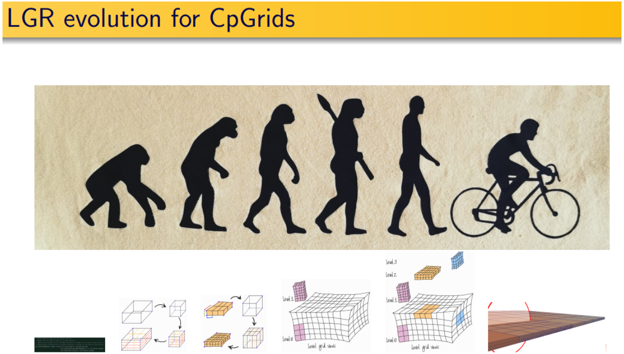
Last OPM Summit - April 2024



LGR evolution for CpGrids

- Parent cells sets
  - block-shaped
  - all active

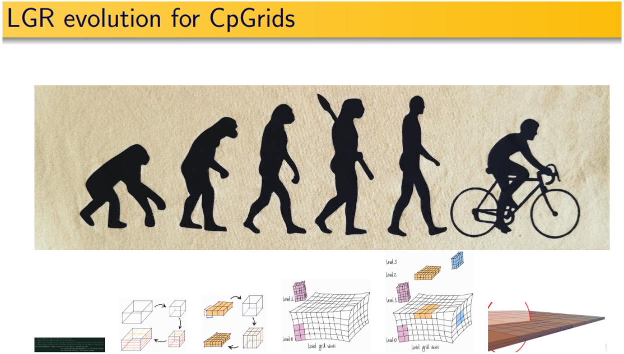
Last OPM Summit - April 2024



LGR evolution for CpGrids

- Parent cells sets
  - block-shaped
  - all active
  - fully disjoint LGRs

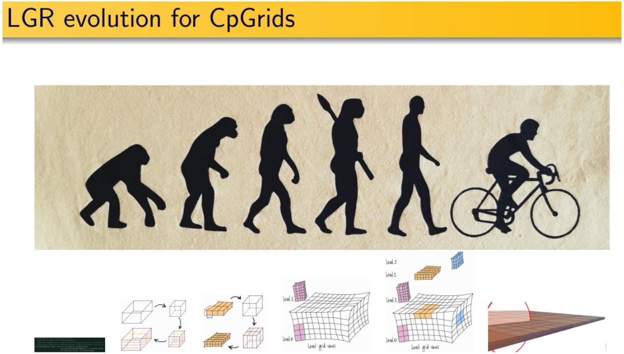
Last OPM Summit - April 2024



LGR evolution for CpGrids

- Parent cells sets
  - block-shaped
  - all active
  - fully disjoint LGRs
  - single-cell refinement for irregular CpGrid

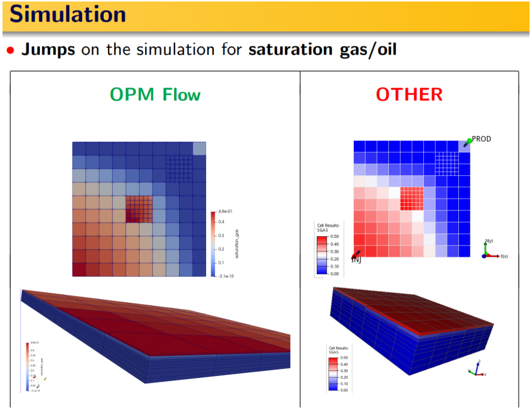
Last OPM Summit - April 2024



LGR evolution for CpGrids

- Parent cells sets
  - block-shaped
  - all active
  - fully disjoint LGRs
  - single-cell refinement for irregular CpGrid
- Simulation
  - serial

Last OPM Summit - April 2024



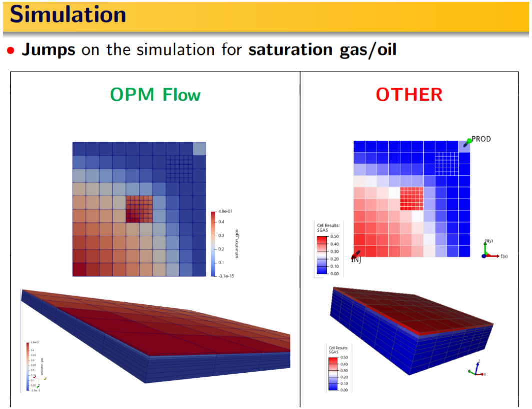
Simulation

- **Jumps** on the simulation for **saturation gas/oil**

- Parent cells sets
  - block-shaped
  - all active
  - fully disjoint LGRs
  - single-cell refinement for irregular CpGrid

- Simulation
  - serial
  - discontinuities in saturation oil/gas

Last OPM Summit - April 2024



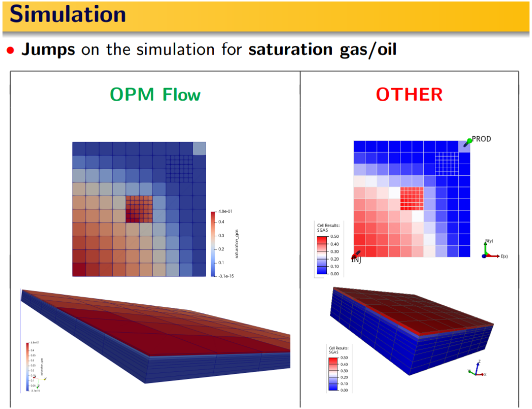
Simulation

- **Jumps** on the simulation for **saturation gas/oil**

- Parent cells sets
  - block-shaped
  - all active
  - fully disjoint LGRs
  - single-cell refinement for irregular CpGrid

- Simulation
  - serial
  - discontinuities in saturation oil/gas
  - no wells

Last OPM Summit - April 2024



Simulation

- **Jumps** on the simulation for **saturation gas/oil**

OPM Flow

OTHER

- Parent cells sets
  - block-shaped
  - all active
  - fully disjoint LGRs
  - single-cell refinement for irregular CpGrid

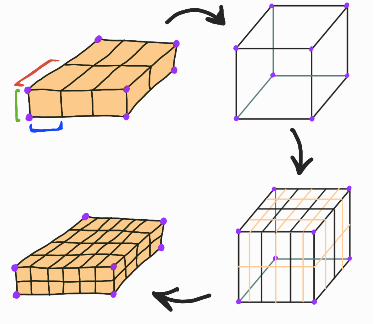
- Simulation
  - serial
  - discontinuities in saturation oil/gas
  - no wells
  - no output files

OPM
OPEN POROUS MEDIA

What has happened since then?

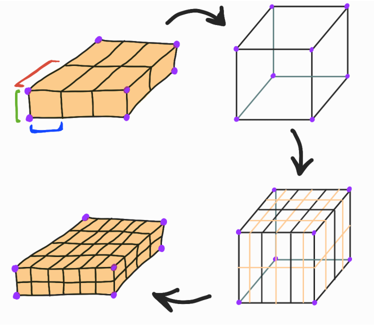
From block-shaped to arbitrary parent cells set

Before: fake-huge-cell-refinement

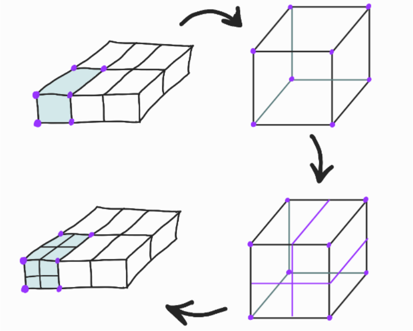


From block-shaped to arbitrary parent cells set

Before: fake-huge-cell-refinement

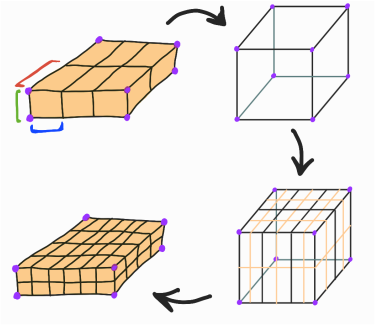


Now: single-cell-refinement

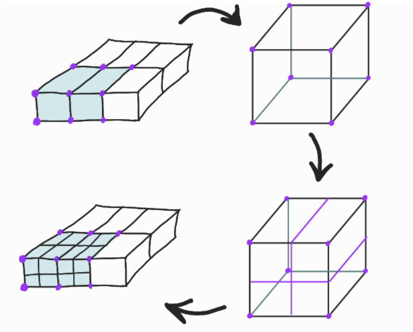


From block-shaped to arbitrary parent cells set

Before: fake-huge-cell-refinement

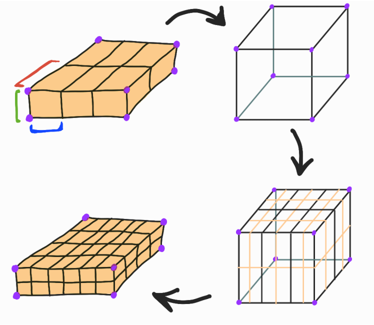


Now: single-cell-refinement

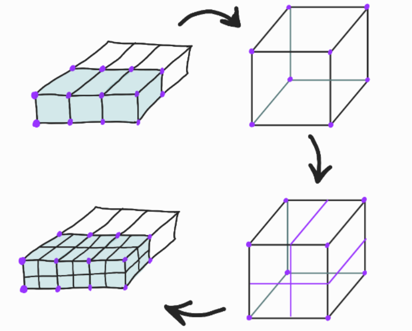


From block-shaped to arbitrary parent cells set

Before: fake-huge-cell-refinement

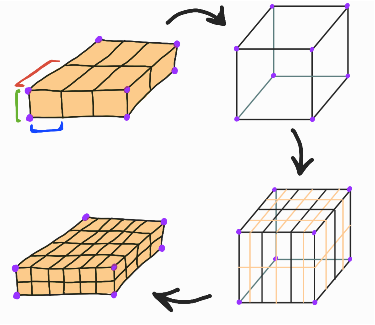


Now: single-cell-refinement

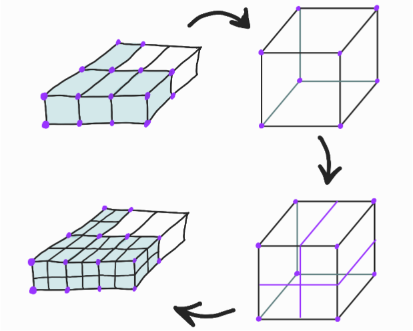


From block-shaped to arbitrary parent cells set

Before: fake-huge-cell-refinement

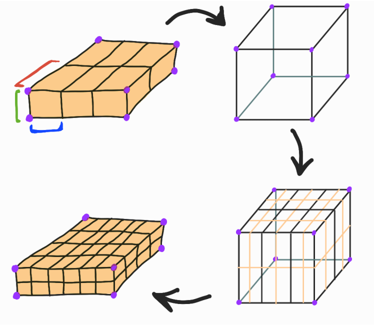


Now: single-cell-refinement

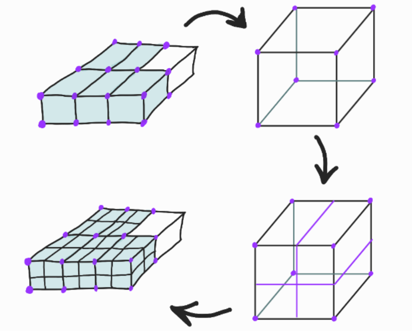


From block-shaped to arbitrary parent cells set

Before: fake-huge-cell-refinement

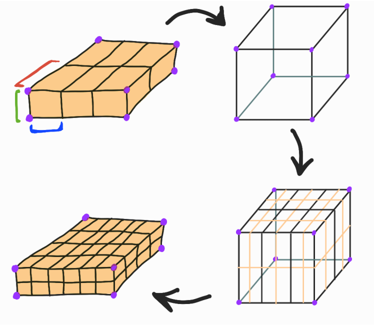


Now: single-cell-refinement

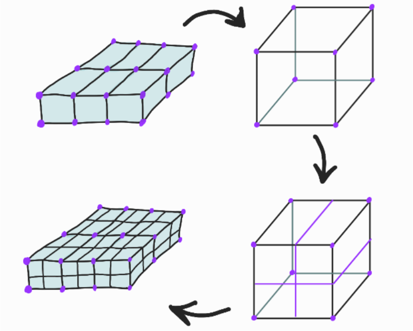


From block-shaped to arbitrary parent cells set

Before: fake-huge-cell-refinement



Now: single-cell-refinement

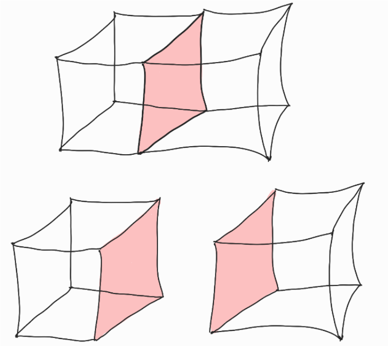


Main challenges

- **Store entities only once!**

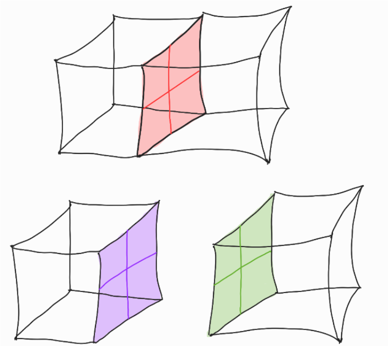
Main challenges

- Store entities only once!



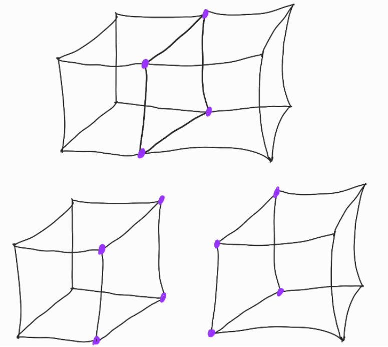
Main challenges

- Store entities only once!



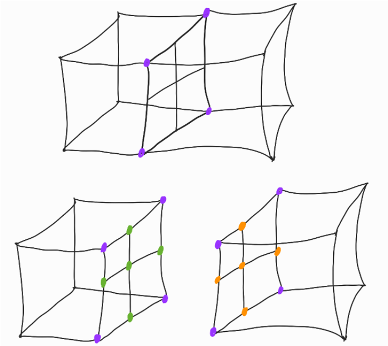
Main challenges

- **Store entities only once!**



Main challenges

- **Store entities only once!**



Main challenges

- **Store entities only once!**

A few methods to avoid repeating entities

- isRefinedCornerInInteriorLgr(…)

Main challenges

- **Store entities only once!**

A few methods to avoid repeating entities

- isRefinedCornerInInteriorLgr(…)
- isRefinedFaceInInteriorLgr(…)

Main challenges

- **Store entities only once!**

A few methods to avoid repeating entities

- isRefinedCornerInInteriorLgr(…)
- isRefinedFaceInInteriorLgr(…)
- isRefinedNewBornCornerOnLgrBoundary(…)

Main challenges

- **Store entities only once!**

A few methods to avoid repeating entities

- isRefinedCornerInInteriorLgr(…)
- isRefinedFaceInInteriorLgr(…)
- isRefinedNewBornCornerOnLgrBoundary(…)
- newRefinedCornerLiesOnEdge(…)

Main challenges

- **Store entities only once!**

A few methods to avoid repeating entities

- isRefinedCornerInInteriorLgr(…)
- isRefinedFaceInInteriorLgr(…)
- isRefinedNewBornCornerOnLgrBoundary(…)
- newRefinedCornerLiesOnEdge(…)
- isRefinedFaceOnLgrBoundary(…)

Main challenges

- **Store entities only once!**

A few methods to avoid repeating entities

- isRefinedCornerInInteriorLgr(…)
- isRefinedFaceInInteriorLgr(…)
- isRefinedNewBornCornerOnLgrBoundary(…)
- newRefinedCornerLiesOnEdge(…)
- isRefinedFaceOnLgrBoundary(…)
- replaceLgr1CornerIdxByLgr2CornerIdx(…)

Main challenges

- **Store entities only once!**

A few methods to avoid repeating entities

- isRefinedCornerInInteriorLgr(…)
- isRefinedFaceInInteriorLgr(…)
- isRefinedNewBornCornerOnLgrBoundary(…)
- newRefinedCornerLiesOnEdge(…)
- isRefinedFaceOnLgrBoundary(…)
- replaceLgr1CornerIdxByLgr2CornerIdx(…)
- replaceLgr1FaceIdxByLgr2FaceIdx(…)

Main challenges

- **Store entities only once!**

A few methods to avoid repeating entities

- isRefinedCornerInInteriorLgr(…)
- isRefinedFaceInInteriorLgr(…)
- isRefinedNewBornCornerOnLgrBoundary(…)
- newRefinedCornerLiesOnEdge(…)
- isRefinedFaceOnLgrBoundary(…)
- replaceLgr1CornerIdxByLgr2CornerIdx(…)
- replaceLgr1FaceIdxByLgr2FaceIdx(…)
- …

Main challenges

- **Store entities only once!**

A few methods to avoid repeating entities

- isRefinedCornerInInteriorLgr(…)
- isRefinedFaceInInteriorLgr(…)
- isRefinedNewBornCornerOnLgrBoundary(…)
- newRefinedCornerLiesOnEdge(…)
- isRefinedFaceOnLgrBoundary(…)
- replaceLgr1CornerIdxByLgr2CornerIdx(…)
- replaceLgr1FaceIdxByLgr2FaceIdx(…)
- …

Don't worry, there is always room from improvement :)

Main challenges

- Store entities only once!
- **Remove any assumption on the parent cells sets *shape***

Main challenges

- Store entities only once!
- **Remove any assumption on the parent cells sets *shape***
    - block-shaped

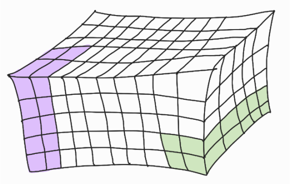
Main challenges

- Store entities only once!
- **Remove any assumption on the parent cells sets *shape***
  - block-shaped
  - fully disjoint

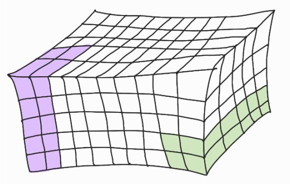
Main challenges

- Store entities only once!
- **Remove any assumption on the parent cells sets *shape***
  - block-shaped
  - fully disjoint

Before: addLgrsUpdateLeafView(...)



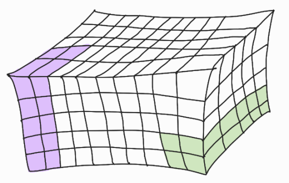
Now: addLgrsUpdateLeafView(...)



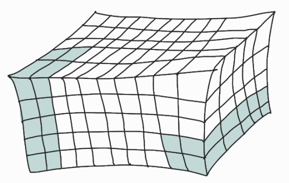
Main challenges

- Store entities only once!
- **Remove any assumption on the parent cells sets *shape***
    - block-shaped
    - fully disjoint

Before: addLgrsUpdateLeafView(...)



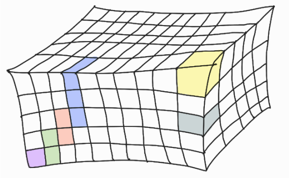
Now: adapt()



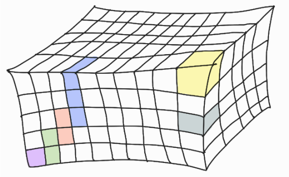
Main challenges

- Store entities only once!
- **Remove any assumption on the parent cells sets *shape***
  - block-shaped
  - fully disjoint

Before: addLgrsUpdateLeafView(...)



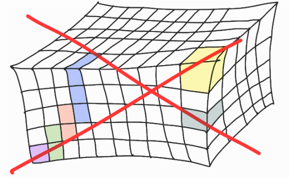
Now: addLgrsUpdateLeafView(...)



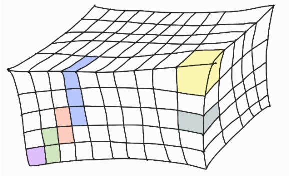
Main challenges

- Store entities only once!
- **Remove any assumption on the parent cells sets *shape***
  - block-shaped
  - fully disjoint

Before: addLgrsUpdateLeafView(...)



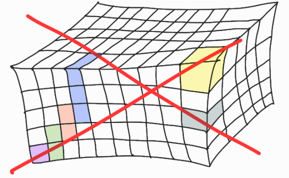
Now: addLgrsUpdateLeafView(...)



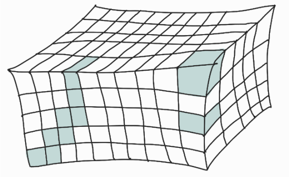
Main challenges

- Store entities only once!
- **Remove any assumption on the parent cells sets *shape***
  - block-shaped
  - fully disjoint

Before: addLgrsUpdateLeafView(...)

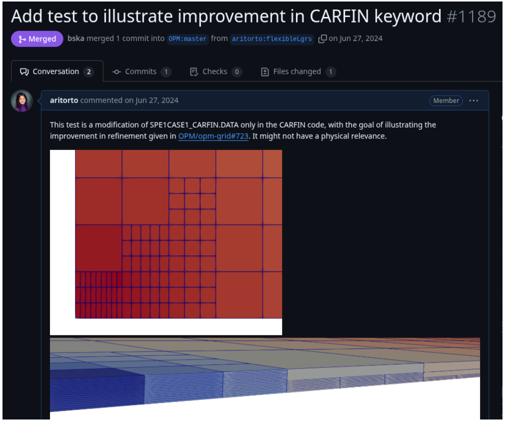


Now: adapt()



Main challenges

- Store entities only once!
- **Remove any assumption on the parent cells sets *shape***
  - block-shaped
  - fully disjoint



Add test to illustrate improvement in CARFIN keyword #1189

Merged bska merged 1 commit into `OPM:master` from `aritorto:flexibleLgrs` on Jun 27, 2024

aritorto commented on Jun 27, 2024

This test is a modification of SPE1CASE1_CARFIN.DATA only in the CARFIN code, with the goal of illustrating the improvement in refinement given in OPM/opm-grid#723. It might not have a physical relevance.

Main challenges

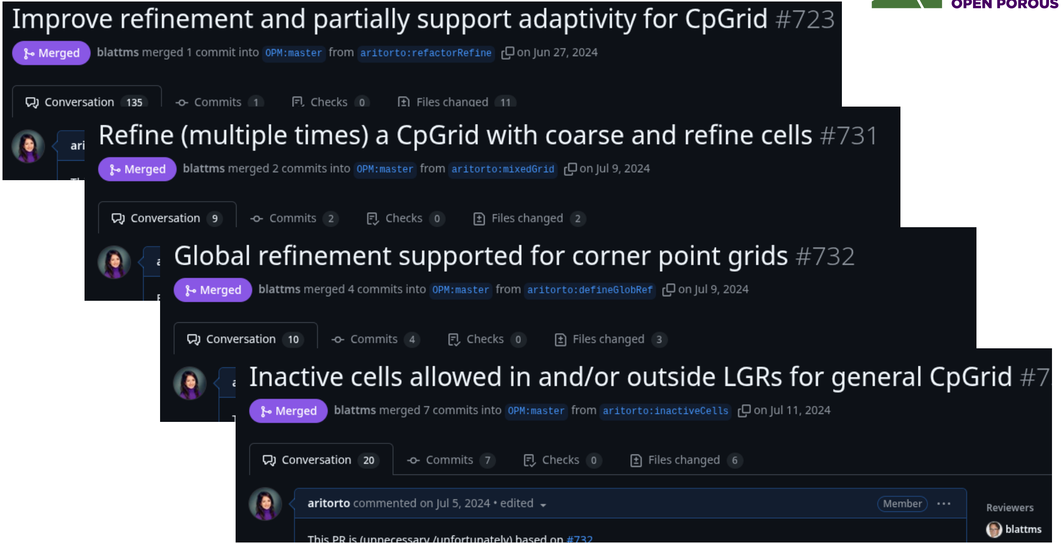
- Store entities only once!
- Remove any assumption on the parent cells sets *shape*
  - block-shaped
  - fully disjoint
- **Define topology aspects of the refined and leaf grid views**

Main challenges

- Store entities only once!
- Remove any assumption on the parent cells sets *shape*
  - block-shaped
  - fully disjoint
- Define topology aspects of the refined and leaf grid views
- ...

What a summer!

OPM
OPEN POROUS MEDIA



Improve refinement and partially support adaptivity for CpGrid #723

Merged blattms merged 1 commit into OPM:master from aritorto:refactorRefine on Jun 27, 2024

Refine (multiple times) a CpGrid with coarse and refine cells #731

Merged blattms merged 2 commits into OPM:master from aritorto:mixedGrid on Jul 9, 2024

Global refinement supported for corner point grids #732

Merged blattms merged 4 commits into OPM:master from aritorto:defineGlobRef on Jul 9, 2024

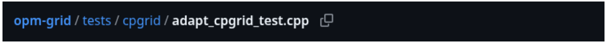
Inactive cells allowed in and/or outside LGRs for general CpGrid #7

Merged blattms merged 7 commits into OPM:master from aritorto:inactiveCells on Jul 11, 2024

aritorto commented on Jul 5, 2024 • edited Member

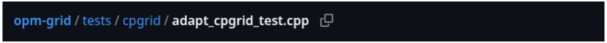
This PR is (unnecessary/unfortunately) based on #732

A few highlights



opm-grid / tests / cpgrid / adapt_cpgrid_test.cpp

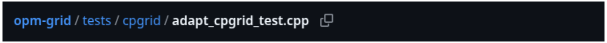
A few highlights



opm-grid / tests / cpgrid / adapt_cpgrid_test.cpp

- *more flexible* parent cell sets

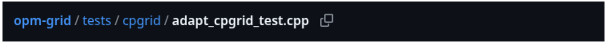
A few highlights



opm-grid / tests / cpgrid / adapt_cpgrid_test.cpp

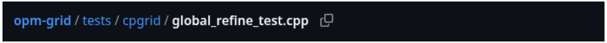
- *more flexible* parent cell sets
- call **adapt()** multiple times as long as marked elements are *far from LGR boundaries*

A few highlights



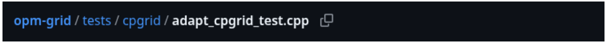
opm-grid / tests / cpgrid / adapt_cpgrid_test.cpp

- *more flexible* parent cell sets
- call **adapt()** multiple times as long as marked elements are *far from LGR boundaries*



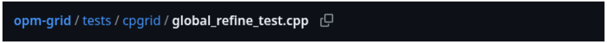
opm-grid / tests / cpgrid / global_refine_test.cpp

A few highlights



opm-grid / tests / cpgrid / adapt_cpgrid_test.cpp

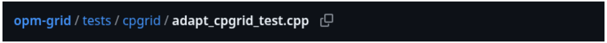
- *more flexible* parent cell sets
- call **adapt()** multiple times as long as marked elements are *far from LGR boundaries*



opm-grid / tests / cpgrid / global_refine_test.cpp

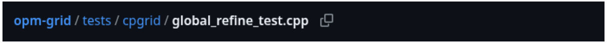
- call **globalRefine(n)** is equivalent to call **globalRefine(t_i)** with $\sum_{i\in I} t_i = n$

A few highlights



opm-grid / tests / cpgrid / adapt_cpgrid_test.cpp

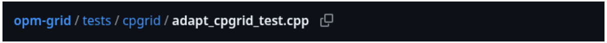
- *more flexible* parent cell sets
- call **adapt()** multiple times as long as marked elements are *far from LGR boundaries*



opm-grid / tests / cpgrid / global_refine_test.cpp

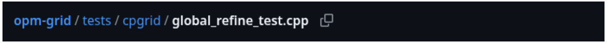
- call **globalRefine(n)** is equivalent to call **globalRefine(t_i)** with $\sum_{i\in I} t_i = n$
- call **adapt()** on a globally refined CpGrid

A few highlights



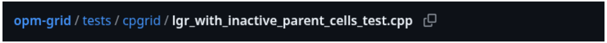
opm-grid / tests / cpgrid / adapt_cpgrid_test.cpp

- *more flexible* parent cell sets
- call **adapt()** multiple times as long as marked elements are *far from LGR boundaries*

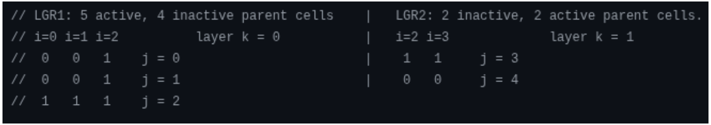


opm-grid / tests / cpgrid / global_refine_test.cpp

- call **globalRefine(n)** is equivalent to call **globalRefine(t_i)** with $\sum_{i\in I} t_i = n$
- call **adapt()** on a globally refined CpGrid

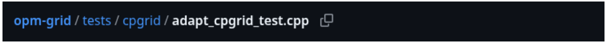


opm-grid / tests / cpgrid / lgr_with_inactive_parent_cells_test.cpp

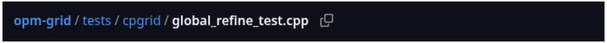


```
// LGR1: 5 active, 4 inactive parent cells    |  LGR2: 2 inactive, 2 active parent cells.
// i=0 i=1 i=2          layer k = 0           |  i=2 i=3             layer k = 1
//  0   0   1    j = 0                        |   1   1     j = 3
//  0   0   1    j = 1                        |   0   0     j = 4
//  1   1   1    j = 2
```

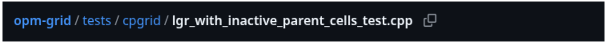
A few highlights

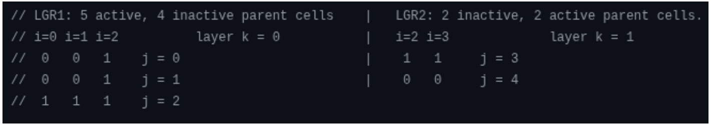

opm-grid / tests / cpgrid / adapt_cpgrid_test.cpp

- *more flexible* parent cell sets
- call **adapt()** multiple times as long as marked elements are *far from LGR boundaries*


opm-grid / tests / cpgrid / global_refine_test.cpp

- call **globalRefine(n)** is equivalent to call **globalRefine(t_i)** with $\sum_{i\in I} t_i = n$
- call **adapt()** on a globally refined CpGrid

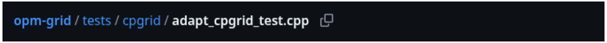

opm-grid / tests / cpgrid / lgr_with_inactive_parent_cells_test.cpp



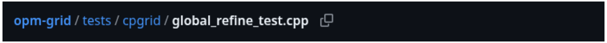
```
// LGR1: 5 active, 4 inactive parent cells   |   LGR2: 2 inactive, 2 active parent cells.
// i=0 i=1 i=2          layer k = 0           |   i=2 i=3             layer k = 1
//  0   0   1    j = 0                        |    1   1     j = 3
//  0   0   1    j = 1                        |    0   0     j = 4
//  1   1   1    j = 2
```

- serial

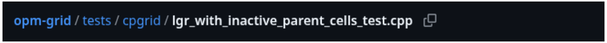
A few highlights

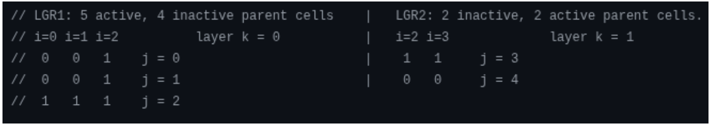

opm-grid / tests / cpgrid / adapt_cpgrid_test.cpp

- *more flexible* parent cell sets
- call **adapt()** multiple times as long as marked elements are *far from LGR boundaries*


opm-grid / tests / cpgrid / global_refine_test.cpp

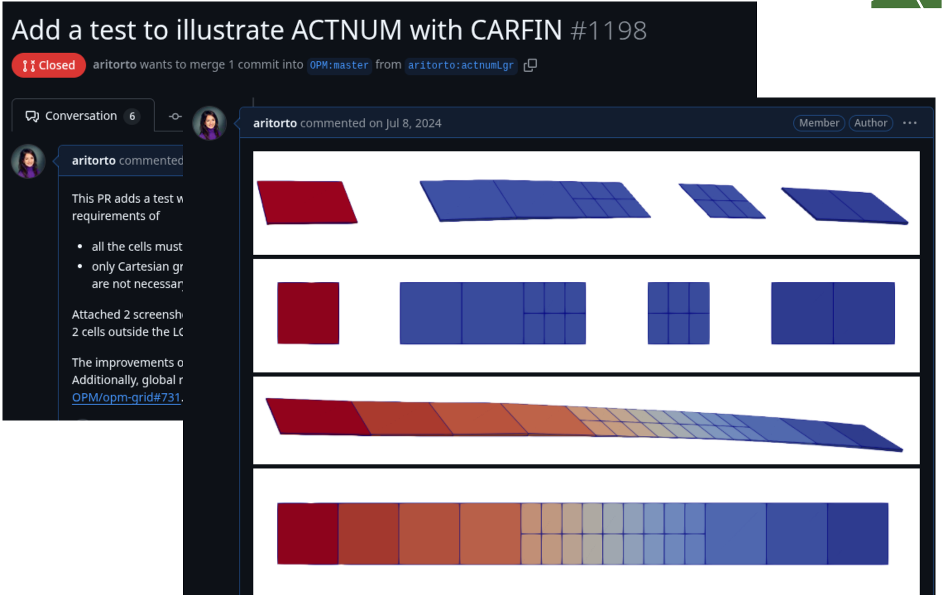
- call **globalRefine(n)** is equivalent to call **globalRefine(t_i)** with $\sum_{i\in I} t_i = n$
- call **adapt()** on a globally refined CpGrid


opm-grid / tests / cpgrid / lgr_with_inactive_parent_cells_test.cpp


```
// LGR1: 5 active, 4 inactive parent cells    |   LGR2: 2 inactive, 2 active parent cells.
// i=0 i=1 i=2          layer k = 0           |   i=2 i=3             layer k = 1
//  0   0   1    j = 0                        |    1   1     j = 3
//  0   0   1    j = 1                        |    0   0     j = 4
//  1   1   1    j = 2
```

- serial
- parallel 😱

From all-active to active/inactive parent cells



Add a test to illustrate ACTNUM with CARFIN #1198

Closed aritorto wants to merge 1 commit into OPM:master from aritorto:actnumLgr

aritorto commented

This PR adds a test w
requirements of

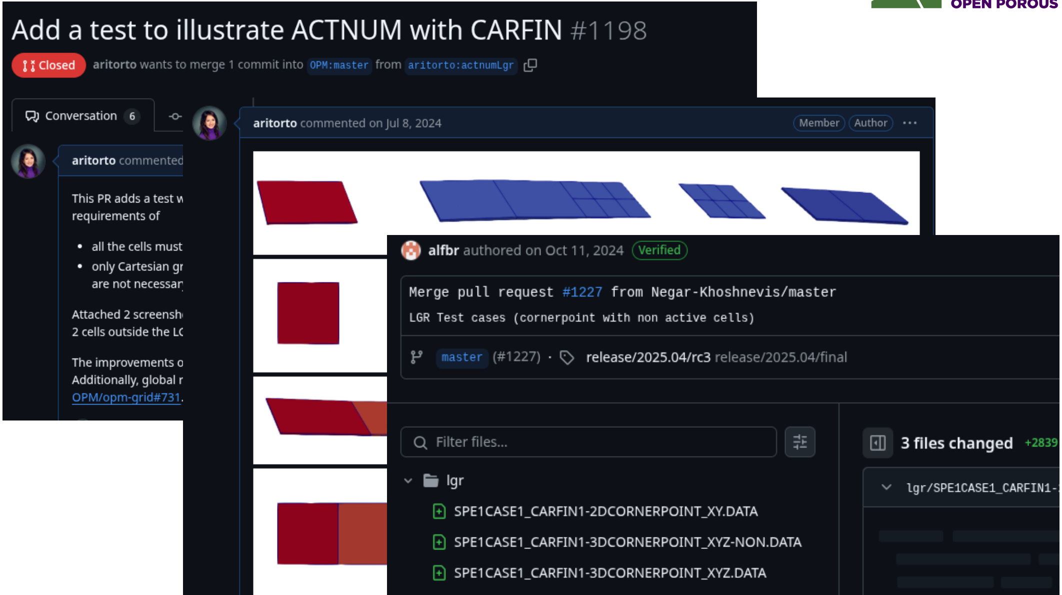
- all the cells must
- only Cartesian gr
are not necessar

Attached 2 screensh
2 cells outside the LG

The improvements o
Additionally, global r
OPM/opm-grid#731.

aritorto commented on Jul 8, 2024

From all-active to active/inactive parent cells



Add a test to illustrate ACTNUM with CARFIN #1198

Closed aritorto wants to merge 1 commit into `OPM:master` from `aritorto:actnumLgr`

aritorto commented

This PR adds a test w
requirements of

- all the cells must
- only Cartesian gr
  are not necessar

Attached 2 screensh
2 cells outside the LC

The improvements o
Additionally, global r
OPM/opm-grid#731.

aritorto commented on Jul 8, 2024 Member Author ···

alfbr authored on Oct 11, 2024 Verified

```
Merge pull request #1227 from Negar-Khoshnevis/master
LGR Test cases (cornerpoint with non active cells)
```

`master` (#1227) · release/2025.04/rc3 release/2025.04/final

- lgr
  - SPE1CASE1_CARFIN1-2DCORNERPOINT_XY.DATA
  - SPE1CASE1_CARFIN1-3DCORNERPOINT_XYZ-NON.DATA
  - SPE1CASE1_CARFIN1-3DCORNERPOINT_XYZ.DATA

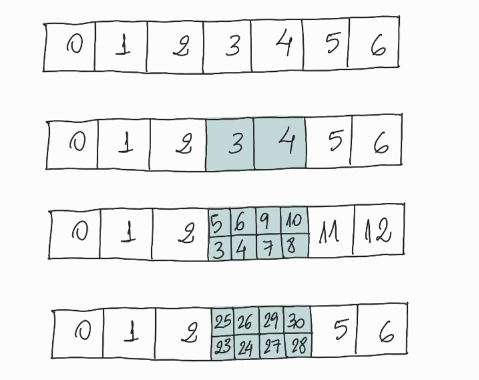
3 files changed +2839

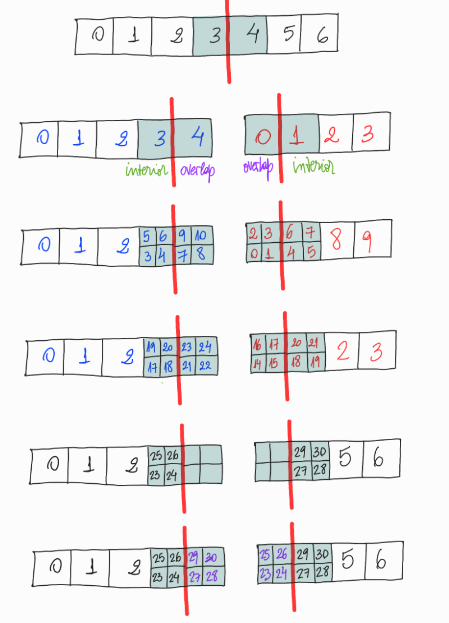
lgr/SPE1CASE1_CARFIN1-

Parallel 😱 simulation for CpGrid with LGRs!?

Parallel simulation, how far?

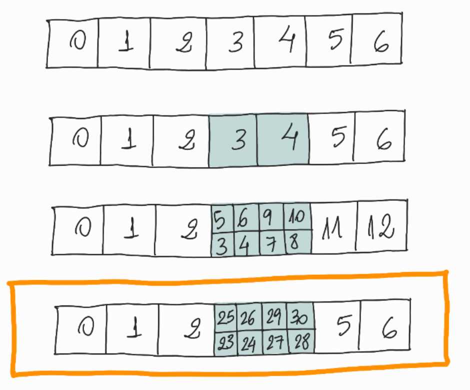
Local indices vs. **Ids**

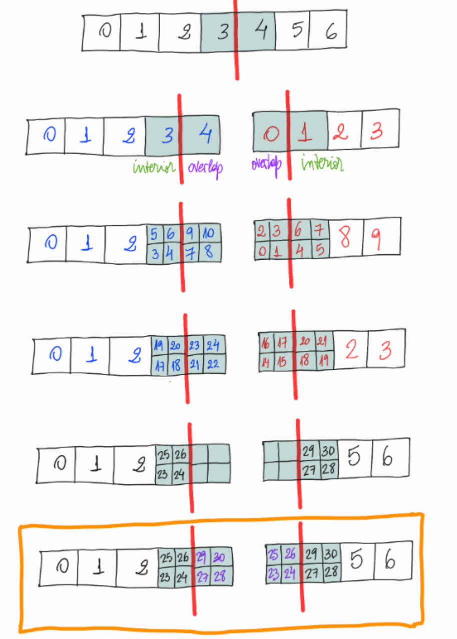




Parallel simulation, how far?

Local indices vs. **Ids**





Parallel simulation, how far?

Distribute level zero grid of a CpGrid with LGRs (PR845)

Partition method: **ZoltanWell** (see **Michal's talk!**)

Parallel simulation, how far?

Distribute level zero grid of a CpGrid with LGRs (PR845)

Partition method: **ZoltanWell** (see **Michal's talk!**)

Main challenges

- Improve **refine-code** to take into account a distributed grid (PR748, PR825)

Parallel simulation, how far?

Distribute level zero grid of a CpGrid with LGRs (PR845)

Partition method: **ZoltanWell** (see **Michal's talk!**)

Main challenges

- Improve **refine-code** to take into account a distributed grid (PR748, PR825)
- Modify **cell/face partition type** (PR782)

Parallel simulation, how far?

Distribute level zero grid of a CpGrid with LGRs (PR845)

Partition method: **ZoltanWell** (see **Michal's talk!**)

Main challenges

- Improve **refine-code** to take into account a distributed grid (PR748, PR825)
- Modify **cell/face partition type** (PR782)
- Define **ids** for cells and vertices, in serial, for CpGrid with LGRs (PR735, PR783, PR802)

Parallel simulation, how far?

Distribute level zero grid of a CpGrid with LGRs (PR845)

Partition method: **ZoltanWell** (see **Michal's talk!**)

Main challenges

- Improve **refine-code** to take into account a distributed grid (PR748, PR825)
- Modify **cell/face partition type** (PR782)
- Define **ids** for cells and vertices, in serial, for CpGrid with LGRs (PR735, PR783, PR802)
- Modify **loadBalance(...)** to distribute level zero grid (for CpGrid with/without LGRs in its global view) (PR783)

Parallel simulation, how far?

Distribute level zero grid of a CpGrid with LGRs (PR845)

Partition method: **ZoltanWell** (see **Michal's talk!**)

Main challenges

- Improve **refine-code** to take into account a distributed grid (PR748, PR825)
- Modify **cell/face partition type** (PR782)
- Define **ids** for cells and vertices, in serial, for CpGrid with LGRs (PR735, PR783, PR802)
- Modify **loadBalance(…)** to distribute level zero grid (for CpGrid with/without LGRs in its global view) (PR783)
- **Synchronize cell ids** (PR854)

Parallel simulation, how far?

Distribute level zero grid of a CpGrid with LGRs (PR845)

Partition method: **ZoltanWell** (see **Michal's talk!**)

Main challenges

- Improve **refine-code** to take into account a distributed grid (PR748, PR825)
- Modify **cell/face partition type** (PR782)
- Define **ids** for cells and vertices, in serial, for CpGrid with LGRs (PR735, PR783, PR802)
- Modify **loadBalance(…)** to distribute level zero grid (for CpGrid with/without LGRs in its global view) (PR783)
- **Synchronize cell ids** (PR854)
- …

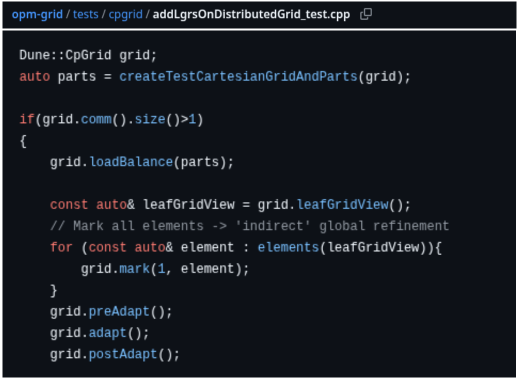
Parallel simulation, how far?

Distribute level zero grid of a CpGrid with LGRs (PR845)

Partition method: **ZoltanWell** (see **Michal's talk!**)

Main challenges

- Improve **refine-code** to take into account a distributed grid (PR748, PR825)
- Modify **cell/face partition type** (PR782)
- Define **ids** for cells and vertices, in serial, for CpGrid with LGRs (PR735, PR783, PR802)
- Modify **loadBalance(...)** to distribute level zero grid (for CpGrid with/without LGRs in its global view) (PR783)
- **Synchronize cell ids** (PR854)
- ...



opm-grid / tests / cpgrid / addLgrsOnDistributedGrid_test.cpp

```
Dune::CpGrid grid;
auto parts = createTestCartesianGridAndParts(grid);

if(grid.comm().size()>1)
{
    grid.loadBalance(parts);

    const auto& leafGridView = grid.leafGridView();
    // Mark all elements -> 'indirect' global refinement
    for (const auto& element : elements(leafGridView)){
        grid.mark(1, element);
    }
    grid.preAdapt();
    grid.adapt();
    grid.postAdapt();
```

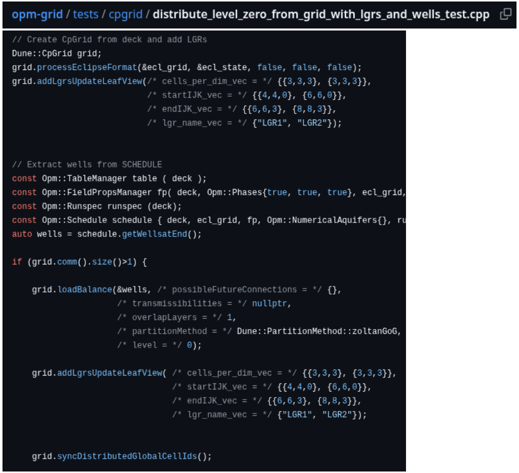
Parallel simulation, how far?

Distribute level zero grid of a CpGrid with LGRs (PR845)

Partition method: **ZoltanWell** (see Michal's talk!)

Main challenges

- Improve **refine-code** to take into account a distributed grid (PR748, PR825)
- Modify **cell/face partition type** (PR782)
- Define **ids** for cells and vertices, in serial, for CpGrid with LGRs (PR735, PR783, PR802)
- Modify **loadBalance(…)** to distribute level zero grid (for CpGrid with/without LGRs in its global view) (PR783)
- **Synchronize cell ids** (PR854)
- …



opm-grid / tests / cpgrid / distribute_level_zero_from_grid_with_lgrs_and_wells_test.cpp

```
// Create CpGrid from deck and add LGRs
Dune::CpGrid grid;
grid.processEclipseFormat(&ecl_grid, &ecl_state, false, false, false);
grid.addLgrsUpdateLeafView(/* cells_per_dim_vec = */ {{3,3,3}, {3,3,3}},
                           /* startIJK_vec = */ {{4,4,0}, {6,6,0}},
                           /* endIJK_vec = */ {{6,6,3}, {8,8,3}},
                           /* lgr_name_vec = */ {"LGR1", "LGR2"});


// Extract wells from SCHEDULE
const Opm::TableManager table ( deck );
const Opm::FieldPropsManager fp( deck, Opm::Phases{true, true, true}, ecl_grid,
const Opm::Runspec runspec (deck);
const Opm::Schedule schedule { deck, ecl_grid, fp, Opm::NumericalAquifers{}, ru
auto wells = schedule.getWellsatEnd();

if (grid.comm().size()>1) {

    grid.loadBalance(&wells, /* possibleFutureConnections = */ {},
                     /* transmissibilities = */ nullptr,
                     /* overlapLayers = */ 1,
                     /* partitionMethod = */ Dune::PartitionMethod::zoltanGoG,
                     /* level = */ 0);

    grid.addLgrsUpdateLeafView( /* cells_per_dim_vec = */ {{3,3,3}, {3,3,3}},
                                /* startIJK_vec = */ {{4,4,0}, {6,6,0}},
                                /* endIJK_vec = */ {{6,6,3}, {8,8,3}},
                                /* lgr_name_vec = */ {"LGR1", "LGR2"});


    grid.syncDistributedGlobalCellIds();
```

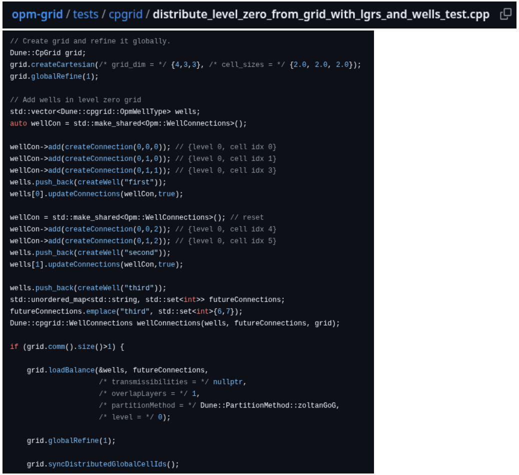
Parallel simulation, how far?

Distribute level zero grid of a CpGrid with LGRs (PR845)

Partition method: **ZoltanWell** (see **Michal's talk!**)

Main challenges

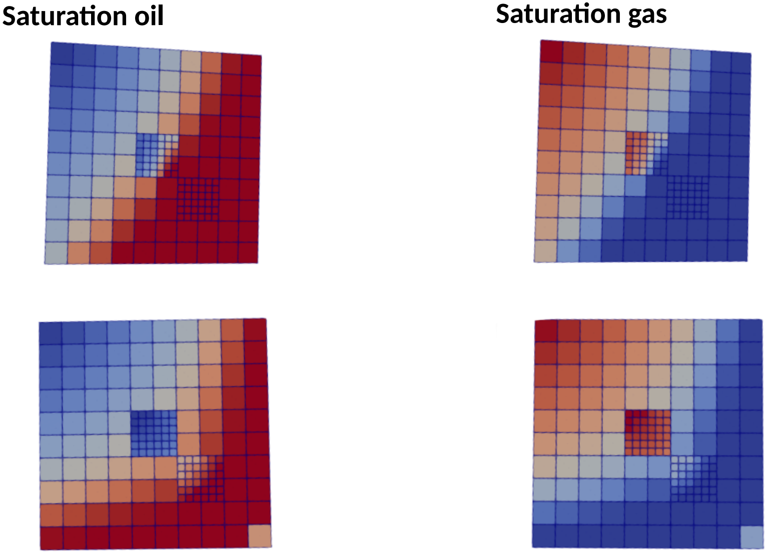
- Improve **refine-code** to take into account a distributed grid (PR748, PR825)
- Modify **cell/face partition type** (PR782)
- Define **ids** for cells and vertices, in serial, for CpGrid with LGRs (PR735, PR783, PR802)
- Modify **loadBalance(...)** to distribute level zero grid (for CpGrid with/without LGRs in its global view) (PR783)
- **Synchronize cell ids** (PR854)
- ...



opm-grid / tests / cpgrid / distribute_level_zero_from_grid_with_lgrs_and_wells_test.cpp

```
// Create grid and refine it globally.
Dune::CpGrid grid;
grid.createCartesian(/* grid_dim = */ {4,3,3}, /* cell_sizes = */ {2.0, 2.0, 2.0});
grid.globalRefine(1);

// Add wells in level zero grid
std::vector<Dune::cpgrid::OpmWellType> wells;
auto wellCon = std::make_shared<Opm::WellConnections>();

wellCon->add(createConnection(0,0,0)); // {level 0, cell idx 0}
wellCon->add(createConnection(0,1,0)); // {level 0, cell idx 1}
wellCon->add(createConnection(0,1,1)); // {level 0, cell idx 3}
wells.push_back(createWell("first"));
wells[0].updateConnections(wellCon,true);

wellCon = std::make_shared<Opm::WellConnections>(); // reset
wellCon->add(createConnection(0,0,2)); // {level 0, cell idx 4}
wellCon->add(createConnection(0,1,2)); // {level 0, cell idx 5}
wells.push_back(createWell("second"));
wells[1].updateConnections(wellCon,true);

wells.push_back(createWell("third"));
std::unordered_map<std::string, std::set<int>> futureConnections;
futureConnections.emplace("third", std::set<int>{6,7});
Dune::cpgrid::WellConnections wellConnections(wells, futureConnections, grid);

if (grid.comm().size()>1) {

    grid.loadBalance(&wells, futureConnections,
                     /* transmissibilities = */ nullptr,
                     /* overlapLayers = */ 1,
                     /* partitionMethod = */ Dune::PartitionMethod::zoltanGoG,
                     /* level = */ 0);

    grid.globalRefine(1);

    grid.syncDistributedGlobalCellIds();
```

Smoother discontinuities in saturation oil/gas (PR6235)

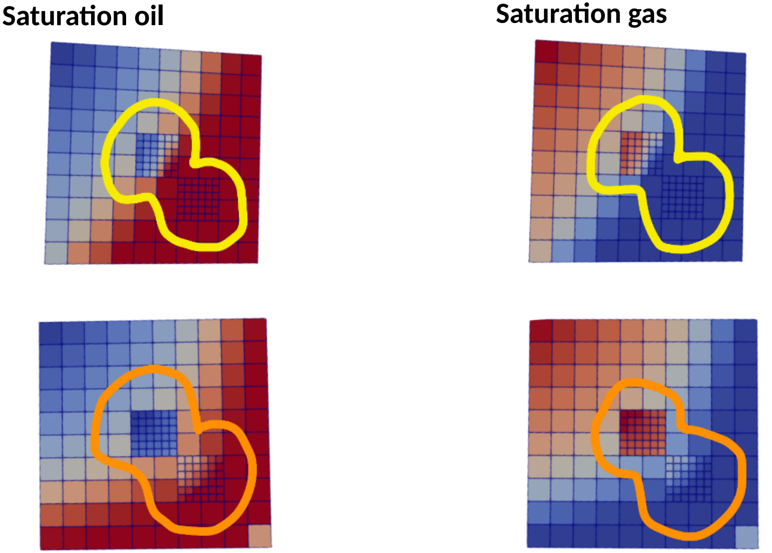


Saturation oil

Saturation gas

top: PR6235, **bottom**: before PR6235

Smoother discontinuities in saturation oil/gas (PR6235)



Saturation oil

Saturation gas

top: PR6235, **bottom**: before PR6235

Wells & output files from opm-grid

- **Wells** (PR862)

Wells & output files from opm-grid

- **Wells** (PR862)
  - small modification in **WellConnections** constructor
    Use **logicalCartesianSize()** and **globalCell()** from **level zero** if the CpGrid has LGRs.

Wells & output files from opm-grid

- **Wells** (PR862)
  - small modification in **WellConnections** constructor
    Use **logicalCartesianSize()** and **globalCell()** from **level zero** if the CpGrid has LGRs.

- To contribute in generating **output files** (*.EGRID, see **Artur's talk!**)

Wells & output files from opm-grid

- **Wells** (PR862)
  - small modification in **WellConnections** constructor
    Use **logicalCartesianSize()** and **globalCell()** from **level zero** if the CpGrid has LGRs.

- To contribute in generating **output files** (*.EGRID, see **Artur's talk!**)
  - **Local Cartesian Mappers** (PR766, PR828)

Wells & output files from opm-grid

- **Wells** (PR862)
  - small modification in **WellConnections** constructor
    Use **logicalCartesianSize()** and **globalCell()** from **level zero** if the CpGrid has LGRs.

- To contribute in generating **output files** (*.EGRID, see **Artur's talk!**)
  - **Local Cartesian Mappers** (PR766, PR828)
  - **COORD** (PR829, PR837) and **ZCORN** (PR831, PR836) for **LGRs**, (PR838)

Last OPM Summit - April 2024

- Parent cells sets
  - block-shaped
  - all active
  - fully disjoint LGRs
  - single-cell refinement for irregular CpGrid

- Simulation
  - serial
  - discontinuities in saturation oil/gas
  - no wells
  - no output files

What has happened since then?

- Parent cells sets
  - block-shaped
  - all active
  - fully disjoint LGRs
  - single-cell refinement for irregular CpGrid

- Simulation
  - serial
  - discontinuities in saturation oil/gas
  - no wells
  - no output files

What has happened since then?

- Parent cells sets
  - **arbitrary *shape***
  - all active
  - fully disjoint LGRs
  - single-cell refinement for irregular CpGrid

- Simulation
  - serial
  - discontinuities in saturation oil/gas
  - no wells
  - no output files

What has happened since then?

- Parent cells set
  - **arbitrary *shape***
  - active and **inactive parent cells**
  - fully disjoint LGRs
  - single-cell refinement for irregular CpGrid

- Simulation
  - serial
  - discontinuities in saturation oil/gas
  - no wells
  - no output files

What has happened since then?

- Parent cells set
  - **arbitrary *shape***
  - active and **inactive parent cells**
  - neighboring LGRs **sharing corners and faces**
  - single-cell refinement for irregular CpGrid

- Simulation
  - serial
  - discontinuities in saturation oil/gas
  - no wells
  - no output files

What has happened since then?

- Parent cells set
  - **arbitrary *shape***
  - active and **inactive parent cells**
  - neighboring LGRs **sharing corners and faces**
  - **refinement for irregular CpGrid** (cells with 6 faces/intersections)

- Simulation
  - serial
  - discontinuities in saturation oil/gas
  - no wells
  - no output files

What has happened since then?

- Parent cells set
  - **arbitrary *shape***
  - active and **inactive parent cells**
  - neighboring LGRs **sharing corners and faces**
  - **refinement for irregular CpGrid** (cells with 6 faces/intersections)

- Simulation
  - serial and **on-the-way-of-parallel** simulation (PR6147 **DRAFT**)
  - discontinuities in saturation oil/gas
  - no wells
  - no output files

What has happened since then?

- Parent cells set
  - **arbitrary *shape***
  - active and **inactive parent cells**
  - neighboring LGRs **sharing corners and faces**
  - **refinement for irregular CpGrid** (cells with 6 faces/intersections)

- Simulation
  - serial and **on-the-way-of-parallel** simulation (PR6147 **DRAFT**)
  - **smoother discontinuities** in saturation oil/gas
  - no wells
  - no output files

What has happened since then?

- parent cells set:
  - **arbitrary *shape***
  - active and **inactive parent cells**
  - neighboring LGRs **sharing corners and faces**
  - **refinement for irregular CpGrid** (cells with 6 faces/intersections)

- Simulation
  - serial and **on-the-way-of-parallel** simulation (PR6147 **DRAFT**)
  - **smoother discontinuities** in saturation oil/gas
  - **wells on-the-way** (much more in **Artur's talk!**)
  - **output files on-the-way** (much more in **Artur's talk!**)

What’s next?

What’s next/on-going?

- **Parallel simulation** for CpGrid with LGRs (PR6147 **DRAFT**)

What's next/on-going?

- **Parallel simulation** for CpGrid with LGRs (PR6147 **DRAFT**)
- Support **coarsening cells** by calling **adapt()**

What's next/on-going?

- **Parallel simulation** for CpGrid with LGRs (PR6147 **DRAFT**)
- Support **coarsening cells** by calling **adapt()**
- Incorporate/test **adapt()** in opm-simulators

What’s next/on-going?

- **Parallel simulation** for CpGrid with LGRs (PR6147 **DRAFT**)
- Support **coarsening cells** by calling **adapt()**
- Incorporate/test **adapt()** in opm-simulators
- **Uniqueness vertex ids** for CpGrid with LGRs (**open issue** 804)

What’s next/on-going?

- **Parallel simulation** for CpGrid with LGRs (PR6147 **DRAFT**)
- Support **coarsening cells** by calling **adapt()**
- Incorporate/test **adapt()** in opm-simulators
- **Uniqueness vertex ids** for CpGrid with LGRs (**open issue** 804)
- Refactor/improve **addLgrsUpdateLeafView(…)** to support **nested refinement** defined on input file (see **Artur’s talk!**)

What’s next/on-going?

- **Parallel simulation** for CpGrid with LGRs (PR6147 **DRAFT**)
- Support **coarsening cells** by calling **adapt()**
- Incorporate/test **adapt()** in opm-simulators
- **Uniqueness vertex ids** for CpGrid with LGRs (**open issue** 804)
- Refactor/improve **addLgrsUpdateLeafView(...)** to support **nested refinement** defined on input file (see **Artur’s talk!**)
- ...

What’s next/on-going?

- **Parallel simulation** for CpGrid with LGRs (PR6147 **DRAFT**)
- Support **coarsening cells** by calling **adapt()**
- Incorporate/test **adapt()** in opm-simulators
- **Uniqueness vertex ids** for CpGrid with LGRs (**open issue** 804)
- Refactor/improve **addLgrsUpdateLeafView(...)** to support **nested refinement** defined on input file (see **Artur’s talk!**)
- ...

Thank you for your attention!

To be continued…