Parallel time integration of hyperbolic PDEs

Oliver Krzysik

March 27, 2018

Chapter 1

Linear advection with first-order methods

1.1 Overview

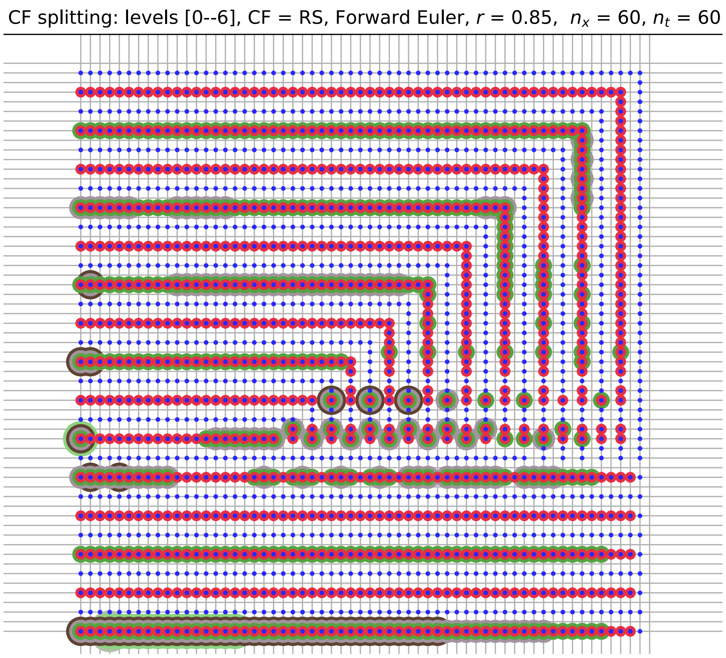
- Test problem is the following linear advection problem:

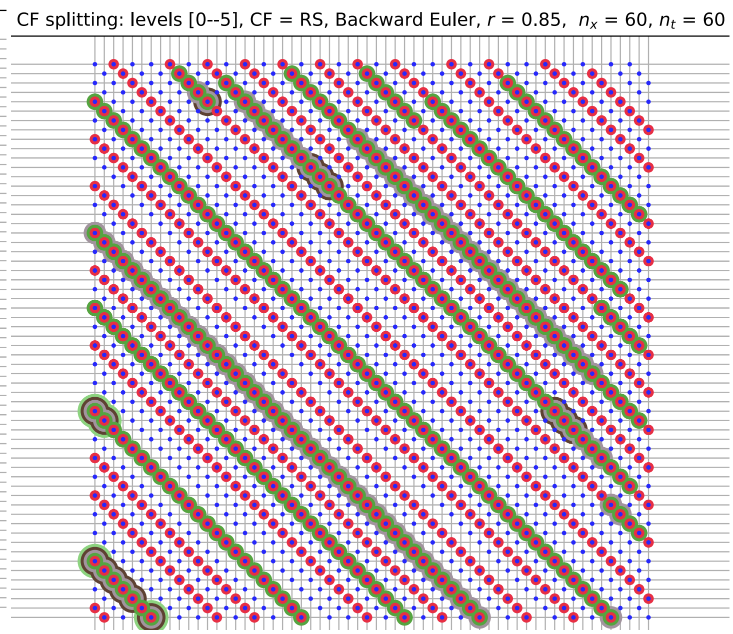
$$u_t + u_x = 0; \quad (x,t) \in (0,2) \times (0,T); \tag{1.1}$$
$$u(x,0) = \exp(-4x^2); \tag{1.2}$$
$$u(0,t) = 1, \tag{1.3}$$

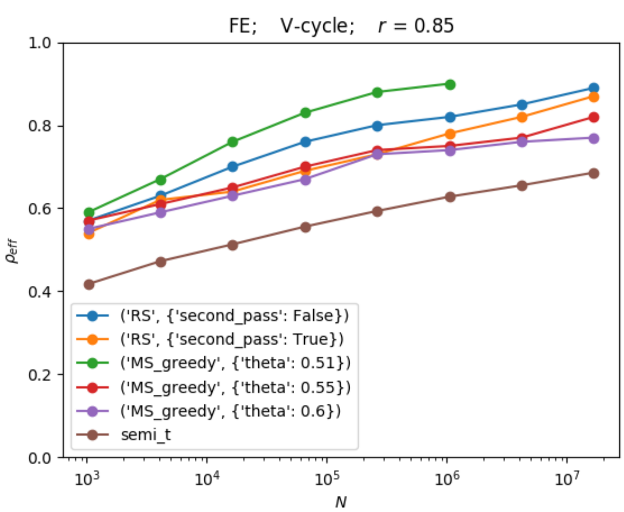
- First-order upwind discretization in space
- Forward Euler (FE): $u_j^{n+1} - u_j^n = r(u_j^n - u_{j-1}^n)$
- Backward Euler (BE): $u_j^{n+1} - u_j^n = r(u_j^{n+1} - u_{j-1}^{n+1})$
- Form space-time linear systems and solve them with AIR. Both systems are lower triangular.
- The total number of unknowns is $N := n_x n_t$.

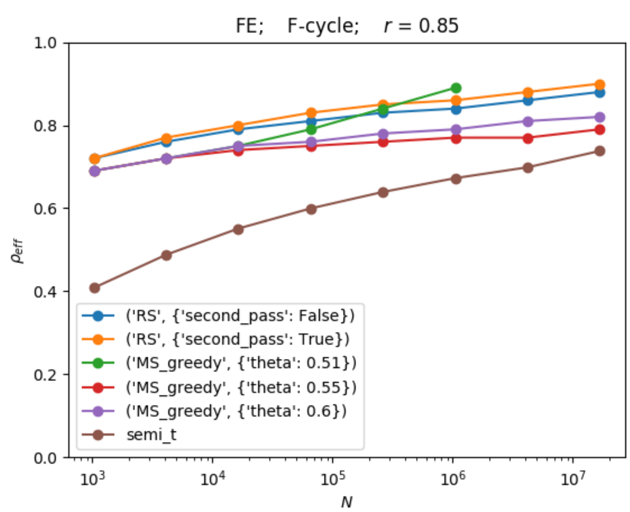
1.2 Results

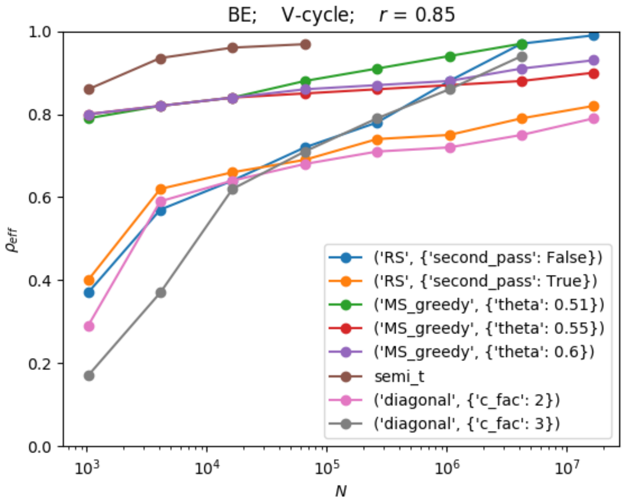
- Get quite different convergence behaviour for FE and BE. Performs worse for BE.
- It looks like RS coarsening attempts to semi-coarsen in time for FE discretization and coarsen along grid diagonals for BE discretization (see figs below). Have grid complexities of a little less than 2 in most cases.
- Find that if these two geometric coarsening approaches are explicitly enforced (with no drop tolerances used anywhere) get an exact reduction method.
- In fact, RS coarsening within AIR leads to exact restriction on finest level. Only able to happen due to discretizations having such small stencils.
- To compare effectiveness of different coarsening strategies look at effective convergence factor: $\rho_{\text{eff}} = \rho^{1/\text{CC}}$ which measures the residual reduction per WU (ρ is the convergence factor, CC is the cycle complexity).

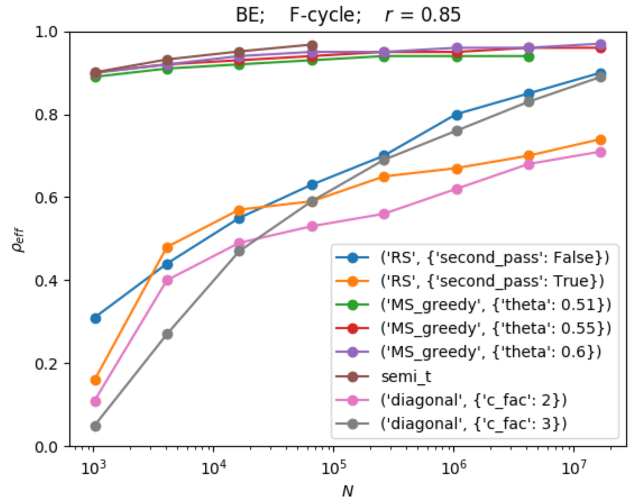












Chapter 2

General first-order PDEs with high-order methods

2.1 Overview

2.1.1 Model problem

- The scalar 1D hyperbolic problem:

$$u_t + f(u)_x = 0; \quad u(x, 0) = u^0(x). \tag{2.1}$$

- Subject to periodic boundary conditions (easiest to consider for high-order FV schemes).
- Think Burgers' and linear advection equations.

2.1.2 Discretization

- Use high-order (standard) WENO (weighted essentially non-oscillatory) FV discretization in space with global Lax–Friedrichs numerical flux function. Typically I use 5th order.
- In time, consider explicit RK method. Typically I use a 4th order one. Can also use multistep methods (but I haven't considered yet).
- Relatively small amount of literature on using WENO with implicit time stepping. But may be something to consider if want steady-state solutions.

2.1.3 Some info about WENO

- WENO is an interpolation procedure that's unrelated to solving PDEs. The idea is (at least in the FV case) to reconstruct point values of a function given its cell-averaged values. And to do this in an essentially non-oscillatory manner.
- If the function is smooth in an interpolation stencil, then can just use standard/linear interpolation (reconstructed values are linear combinations of the cell averages). However, if function to be reconstructed is not smooth in stencil, then standard interpolation will yield an oscillatory (and likely wildly inaccurate) reconstruction.
- WENO performs nonlinear reconstruction of the function. It's nonlinear since the interpolation weights depend on the smoothness of the function to be reconstructed. If solution is smooth in stencil, then WENO weights attempt to mimic the linear weights. If solution is not smooth in stencil, WENO weights become small there.
- To solve PDEs, we use the WENO procedure to reconstructed point values which are then used in a numerical flux function.
- WENO discretization is nonlinear even when applied to a linear PDE!

2.2 The discrete problem

- Applying method of lines (discretizing in space) gives coupled system of ODEs:

$$\frac{\mathrm{d}\bar{u}}{\mathrm{dt}} = L(\bar{u}); \quad \bar{u}(0) = \bar{u}^0, \tag{2.2}$$

  where L is a WENO approximation of $-f(u)_x$. Here, $\bar{u} = (\bar{u}_1, \ldots, \bar{u}_{n_x})^T$ are the cell-averaged values of u.

- Explicit RK applied to ODEs gives rise to the system of equations:

$$\bar{u}^{n+1} = \Phi(\bar{u}^n), \quad n = 0, \ldots, n_t - 1 \tag{2.3}$$

  with $\Phi : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ encoding the RK method. Letting $\phi_i : \mathbb{R}^{n_x} \to \mathbb{R}$ be the ith component of Φ, at time level n have

$$\bar{u}_i^{n+1} = \phi_i(\bar{u}^n), \quad i = 1, \ldots, n_x. \tag{2.4}$$

- Letting $v = (\bar{u}^1, \ldots, \bar{u}^{n_t})^T \in \mathbb{R}^{n_x n_t}$, the discrete space-time problem may be written

$$F(v) := A(v) - b = 0, \tag{2.5}$$

  with nonlinear operator $A : \mathbb{R}^{n_x n_t} \to \mathbb{R}^{n_x n_t}$ and RHS vector $b = (\Phi(\bar{u}^0), 0, \ldots, 0) \in \mathbb{R}^{n_x n_t}$.

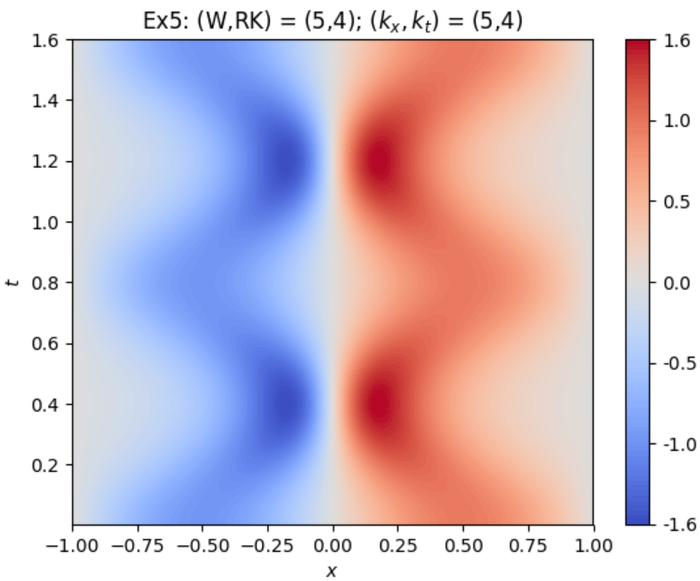
- The (i, n)th component of the nonlinear system of equations (2.5) is $F(v)_i^n = \bar{u}_i^n - \phi_i(\bar{u}^{n-1}) = 0$

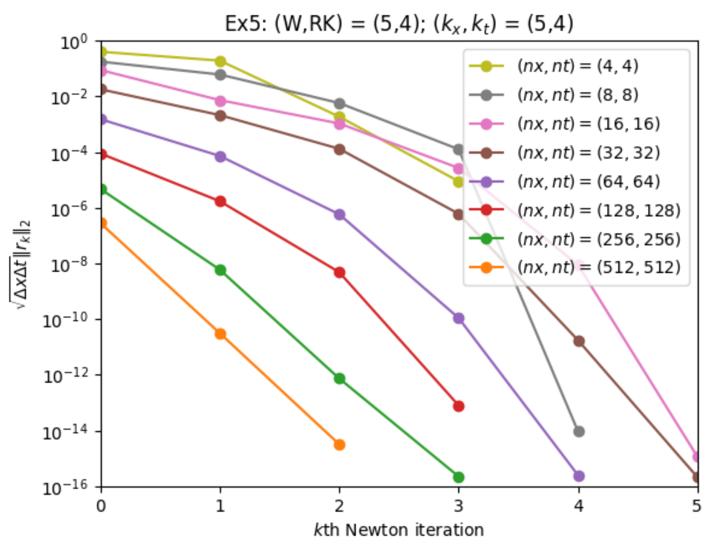
2.3 Numerical solution of discrete problem

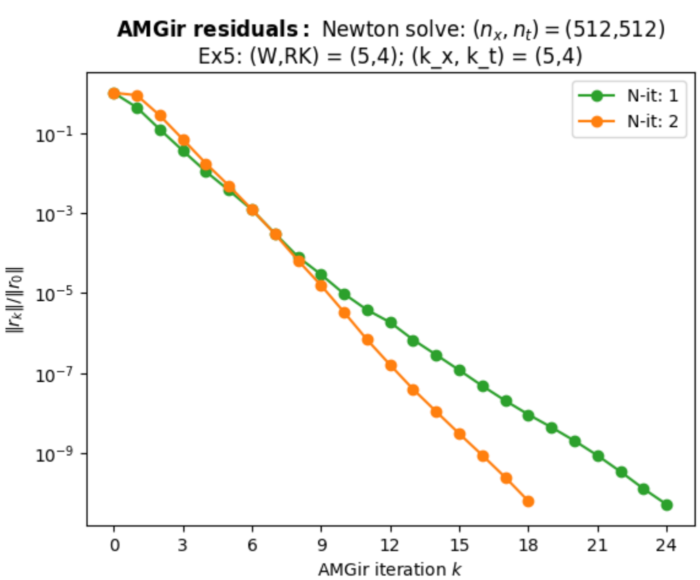
- The approach explored here is to use Newton-multigrid to solve (2.5) with AIR as the multigrid solver in each Newton iteration.
- All components of the resulting algorithm are parallelizable: the construction of the Jacobian, the residual computation, and the linear solve.
- Note that the Jacobian is block bidiagonal, with identity blocks along diagonal.
- Using a nested iteration strategy to generate starting guess for Newton method (solve the same problem discretized on coarser space and interpolate it to the desired space as the initial guess).

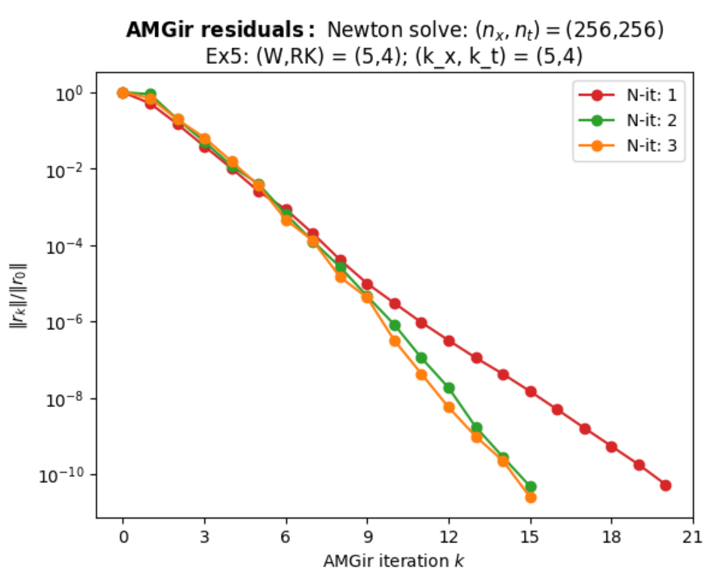
2.3.1 Linear advection example

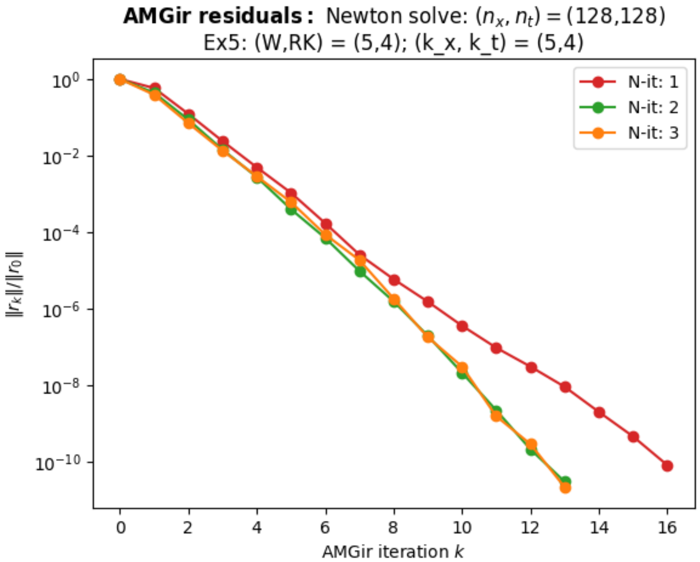
- The PDE (2.1) has flux $f(u, x, t) = -\sin(2.5\pi t)\sin(\pi x)u$ (linear PDE)
- As space-time solution is smooth, WENO nonlinearity shouldn't really be too strong, and it should vanish as $\Delta x \to 0$.
- Newton converges with few iterations, requiring only 2 on the finest grid.
- AMGir convergence factors are reasonable for all cases; slight degradation for increasing problem size. It seems like quadrupling the problem size requires 3 extra AMGir iterations to achieve same residual tolerance.
- The linear system in the first Newton iteration always seems more difficult to solve.

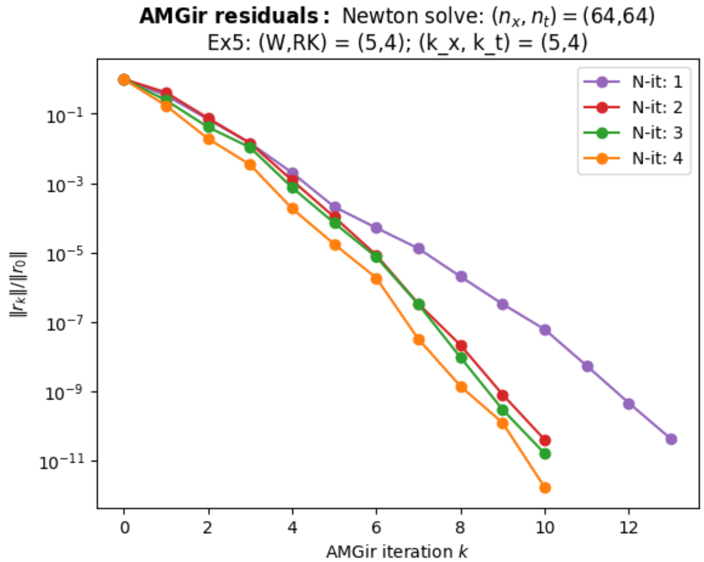






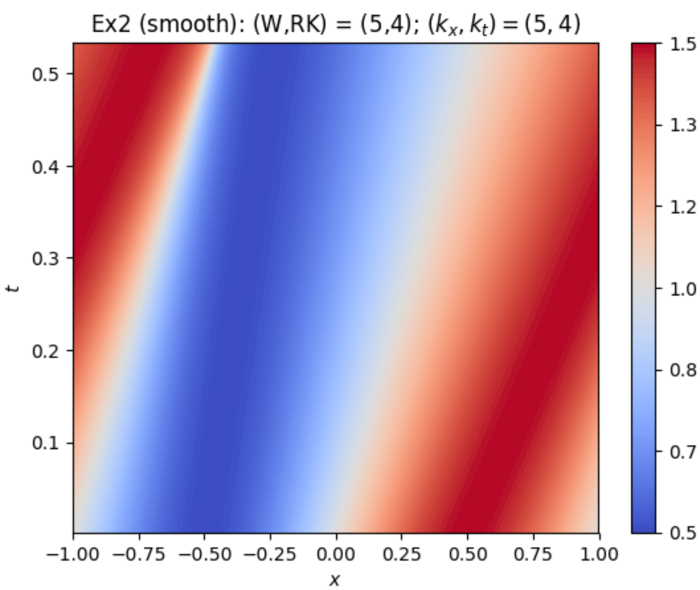


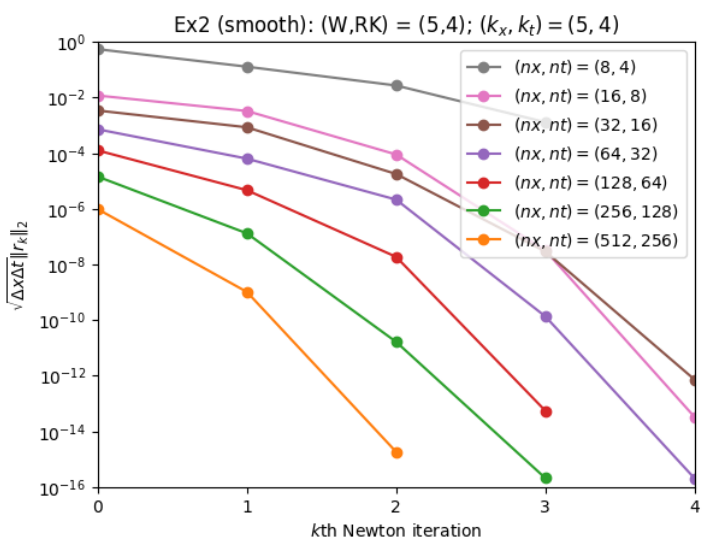


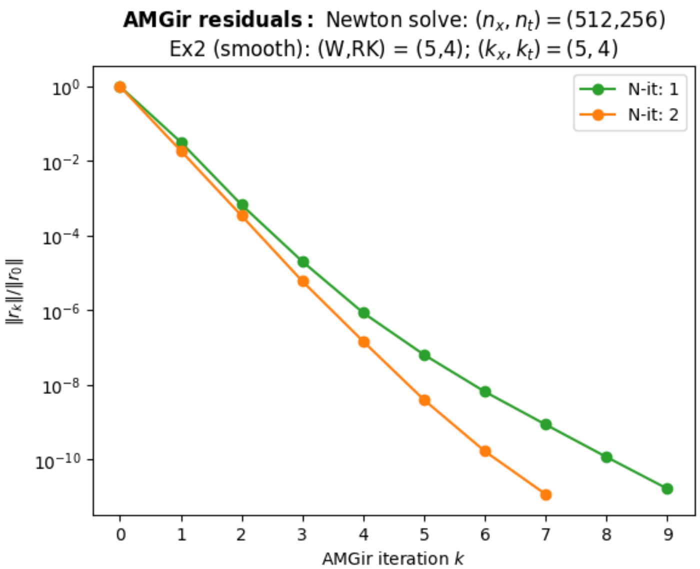


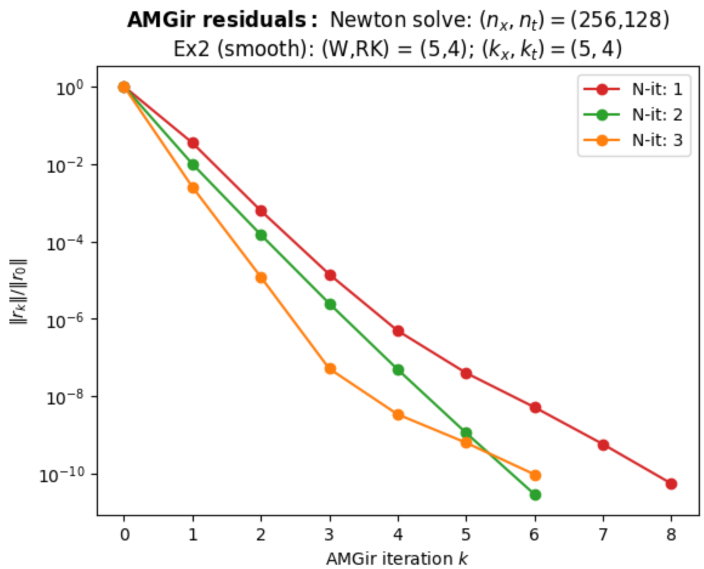
2.3.2 Smooth solution of Burgers'

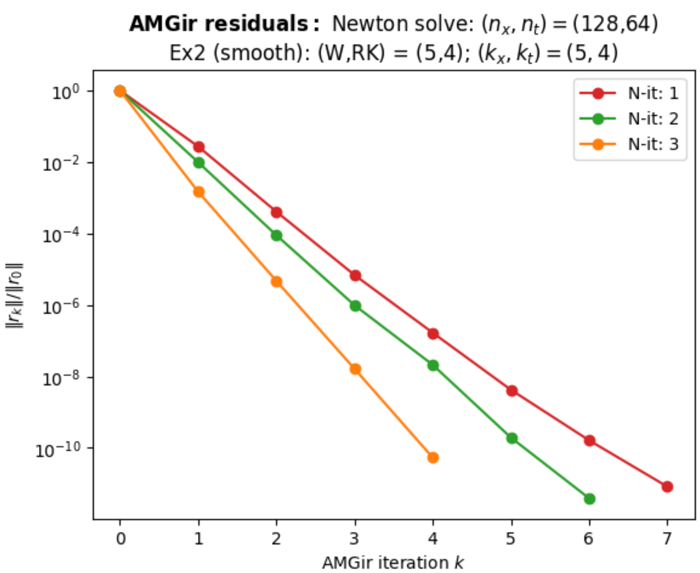
- The PDE (2.1) has flux $f(u) = u^2/2$ (nonlinear PDE).
- Basically all of the observations from last case hold here too, except fewer AIR iterations required to solve the linear systems (note that linear systems here have half as many DOFs as in previous case).

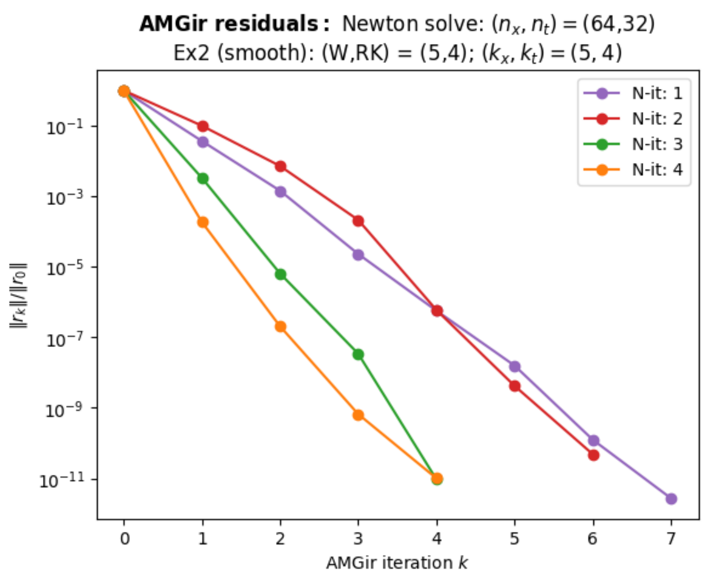




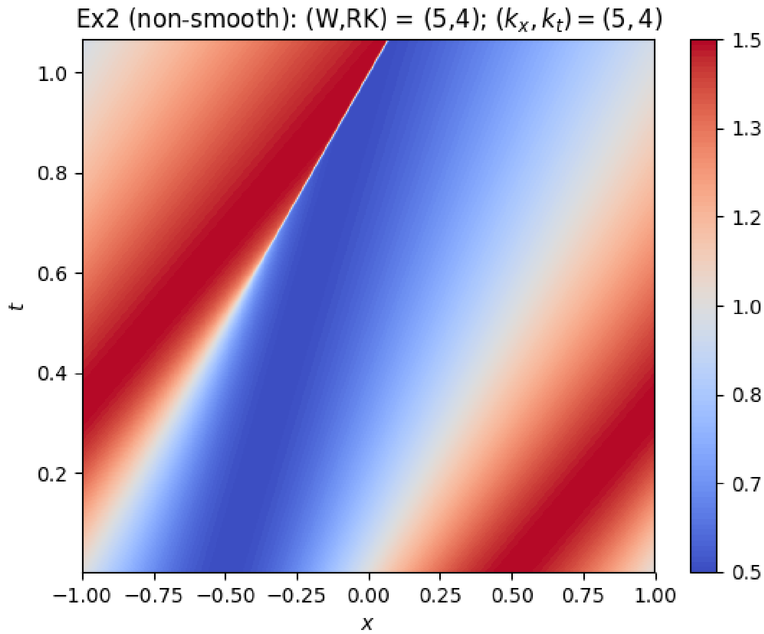

AMGir residuals: Newton solve: $(n_x, n_t) = (512,256)$
Ex2 (smooth): (W,RK) = (5,4); $(k_x, k_t) = (5, 4)$
N-it: 1
N-it: 2
$\|r_k\|/\|r_0\|$
AMGir iteration k


AMGir residuals: Newton solve: $(n_x, n_t) = (256,128)$
Ex2 (smooth): (W,RK) = (5,4); $(k_x, k_t) = (5, 4)$
N-it: 1
N-it: 2
N-it: 3
$\|r_k\|/\|r_0\|$
AMGir iteration k

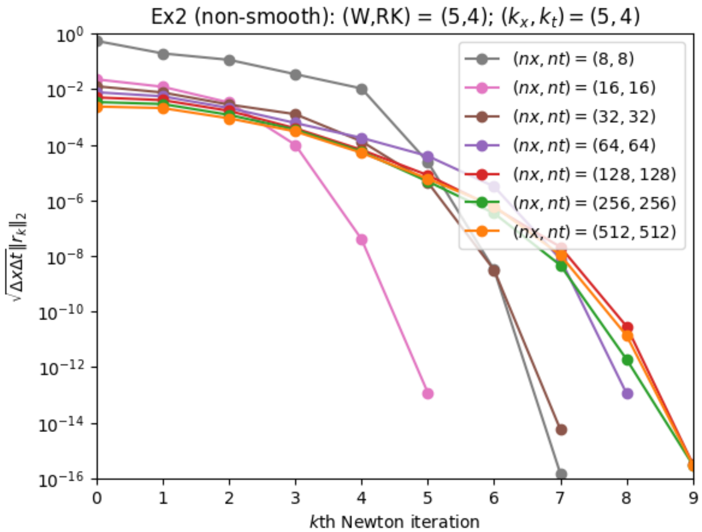

AMGir residuals: Newton solve: $(n_x, n_t) = (128,64)$
Ex2 (smooth): (W,RK) = (5,4); $(k_x, k_t) = (5, 4)$
N-it: 1
N-it: 2
N-it: 3
$\|r_k\|/\|r_0\|$
AMGir iteration k

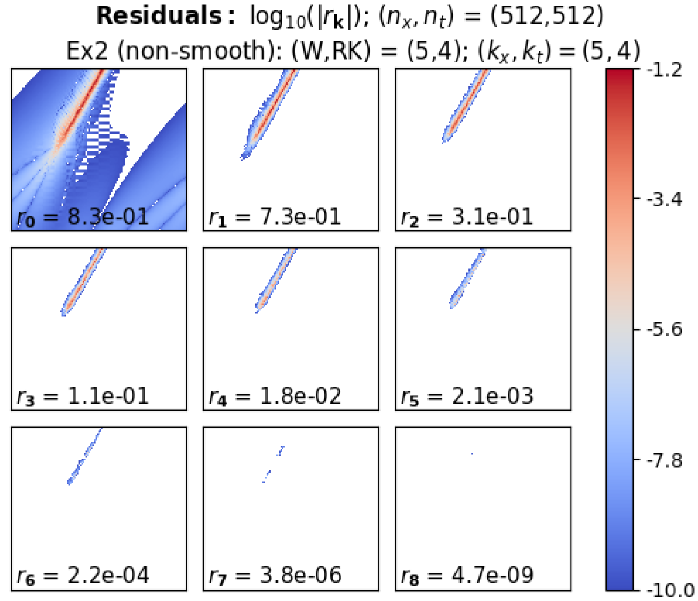

AMGir residuals: Newton solve: $(n_x, n_t) = (64,32)$
Ex2 (smooth): (W,RK) = (5,4); $(k_x, k_t) = (5, 4)$
N-it: 1
N-it: 2
N-it: 3
N-it: 4
$\|r_k\|/\|r_0\|$
AMGir iteration k

2.3.3 Discontinuous solution of Burgers'

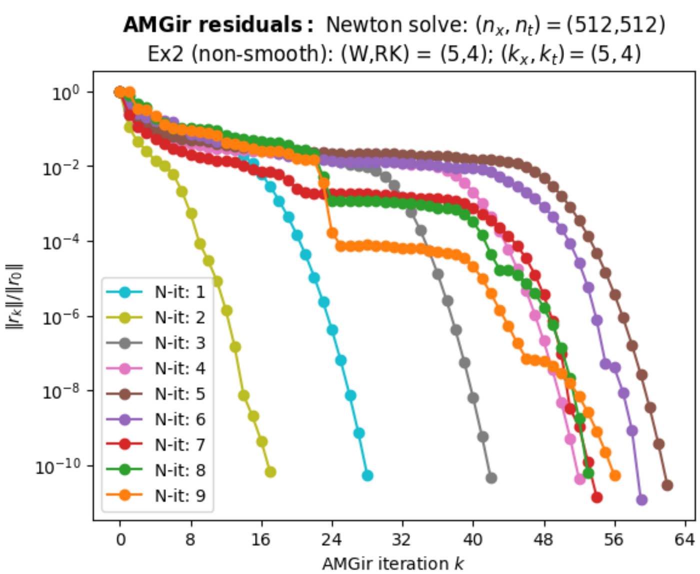


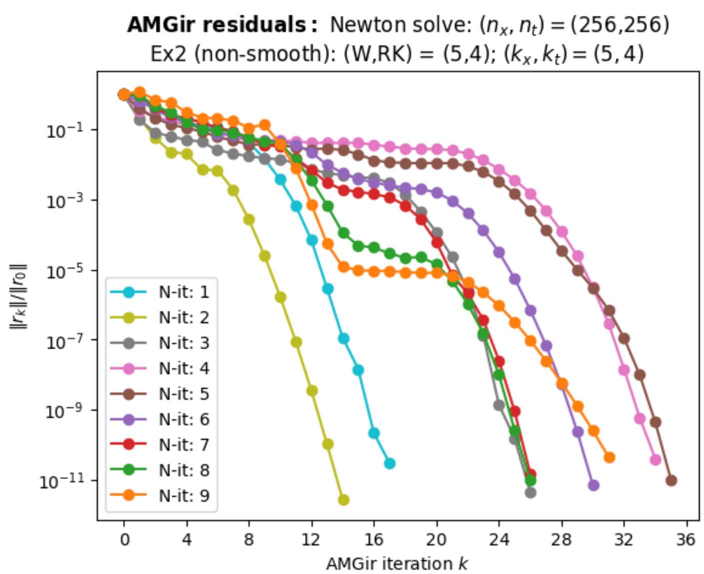
- The PDE (2.1) has flux $f(u) = u^2/2$ (nonlinear PDE).
- A shock forms in the domain and then propagates to the right.
- Get significant degradation in Newton convergence. Iteration counts no longer decrease with problem size. But they do seem to be stagnating for 3 finest grids.
- Newton convergence stalls around the shock. Note that initial guess is very accurate everywhere except in this region (discretization is diffusive around discontinuity, so when grid is refined the dissipation is less and the shock steepens).
- In one Newton step nonlinear residual is less than $1\mathrm{e}{-10}$ everywhere except in shocked region.
- To improve Newton convergence have tried line searching to find optimal step length. Also tried to accelerate it by forming system of equations as an unconstrained optimisation problem and applying N-GMRES to this. Neither of these helped!

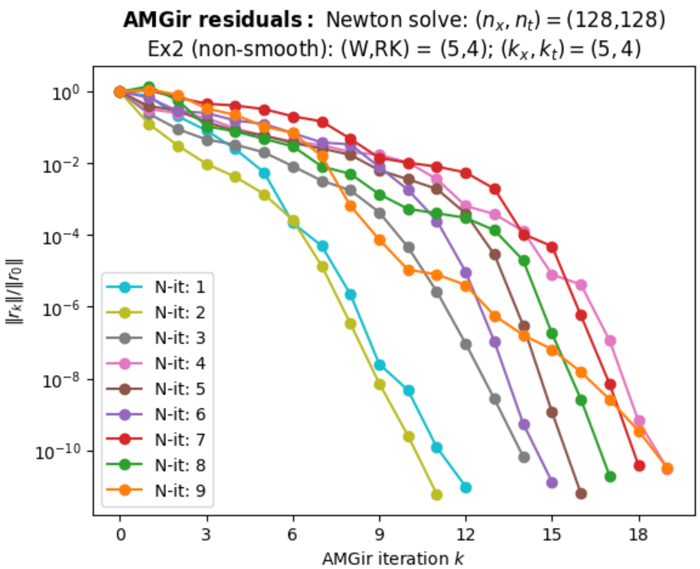


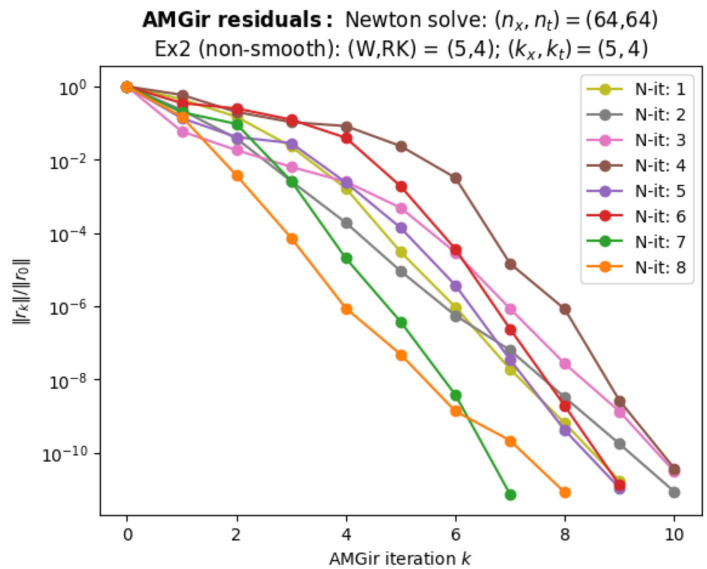


- Also get very poor AIR convergence on resulting linear systems.
- Haven't really looked into poor convergence of AIR, since it likely stems from poorly scaled Jacobians, so more important to try and fix that first!
- AIR convergence degrades significantly as moving to finer meshes. Solves arising from the first couple of Newton iterations always seem reasonable(ish).






AMGir residuals: Newton solve: $(n_x, n_t) = (128,128)$
Ex2 (non-smooth): (W,RK) = (5,4); $(k_x, k_t) = (5,4)$
$\|r_k\|/\|r_0\|$
AMGir iteration k
N-it: 1
N-it: 2
N-it: 3
N-it: 4
N-it: 5
N-it: 6
N-it: 7
N-it: 8
N-it: 9


AMGir residuals: Newton solve: $(n_x, n_t) = (64,64)$
Ex2 (non-smooth): (W,RK) = (5,4); $(k_x, k_t) = (5,4)$
$\|r_k\|/\|r_0\|$
AMGir iteration k
N-it: 1
N-it: 2
N-it: 3
N-it: 4
N-it: 5
N-it: 6
N-it: 7
N-it: 8