Additively Damped AFAC Variants for High Order Discretisations

Charles Murray, Tobias Weinzierl
c.d.murray@durham.ac.uk, tobias.weinzierl@durham.ac.uk

Durham University

Introduction

We improve upon existing methods to solve the variable coefficient poisson $\nabla\epsilon \cdot \nabla u$
An elliptic equation that appears in a lot of diffusion dominated setups
The material parameter ϵ can be discontinuous which is detrimental to rate of convergence
Complicated ϵ creates commensurately expensive construction of equation system

Additive Multigrid

Additive Multigrid is a variant of multigrid that removes the sequential definition of smoothing on different levels—that is coarse grid level smoothing no longer requires fine grid levels to be smoothed before hand.
This increases potential for parallelism but also can reduce stability of the solver.
Smooth (reduce the error) on multiple levels with no synchronisation between levels

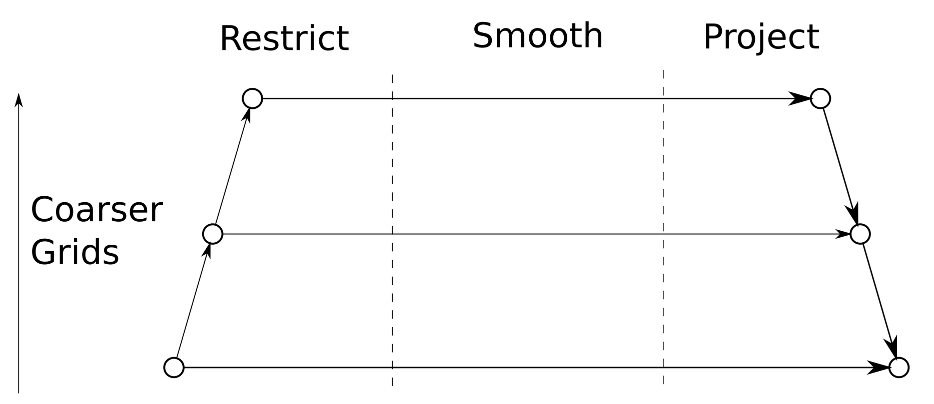

Figure 1: V-cycle for Additive Multigrid

Auxilliary grids

Geometric definition of the multigrid grid hierarchy, we use space trees
Grids that are locally cartesian but embed finer grids within a coarse grid for arbitrary meshes
Such grids mean neighbours can't necessarily be accessed, inability to guarantee or interact with neighbours forces specific sparsity patterns—FE stencils have local support

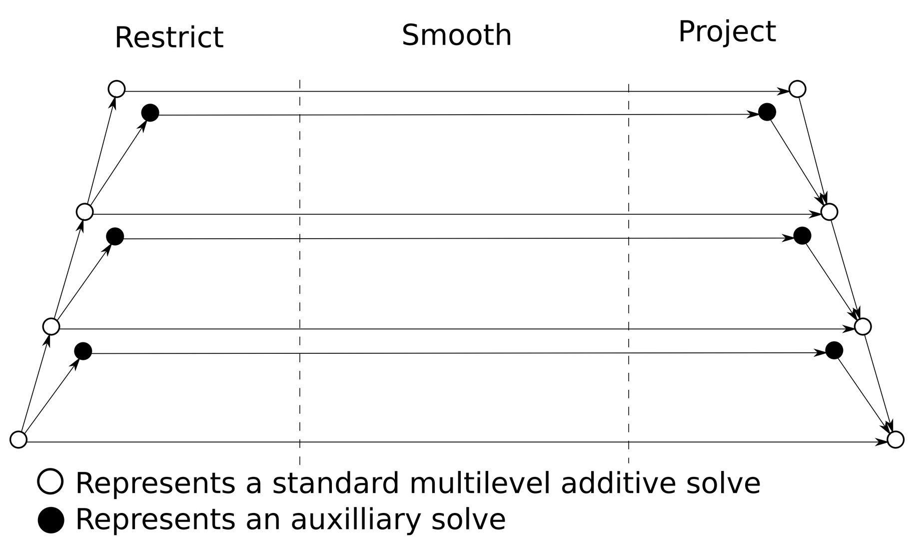

Figure 2: Modified V-cycle for our adAFAC-Jac method

Our inspiration is the AFAC/AFACx family of methds: a method on geometric grids that introduces "auxilliary" grids
Auxilliary grids used to compute damping parameters for the original grids so that all grid levels can be computed in parallel
However all have drawbacks—each damping parameter is result of multilevel solve or requires an sequential smoothing step.
We want to remove sequential component between smooths to increase parallelism, i.e. construct damping parameter in parallel with all other smoothing steps

High level view

We are motivated by the observation that a two level $V(1,0)$ Multigrid v-cycle can be written

$$\begin{aligned} u_{\ell_{max},mult} \leftarrow\; & PA^{-1}_{\ell_{max}-1}R(b_{\ell_{max}} - A_{\ell_{max}}\left[u_{\ell_{max}} + \omega_{\ell_{max}}M^{-1}_{\ell_{max}}(b_{\ell_{max}} - A_{\ell_{max}}u_{\ell_{max}})\right]) \\ & + \left[u_{\ell_{max}} + \omega_{\ell_{max}}M^{-1}_{\ell_{max}}(b_{\ell_{max}} - A_{\ell_{max}}u_{\ell_{max}})\right] \end{aligned}$$

A two level can Additive Multigrid v-cycle can be written

$$u_{\ell_{max},add} \leftarrow PA^{-1}_{\ell_{max}-1}R(b_{\ell_{max}} - A_{\ell_{max}}u^{(n)}_{\ell_{max}}) + \left[u_{\ell_{max}} + \omega_{\ell_{max}}M^{-1}_{\ell_{max}}(b_{\ell_{max}} - A_{\ell_{max}}u_{\ell_{max}})\right].$$

The difference between these is the term

$$-PA^{-1}_{\ell_{max}-1}RA_{\ell_{max}}\omega_{\ell_{max}}M^{-1}_{\ell_{max}}(b_{\ell_{max}} - A_{\ell_{max}}u_{\ell_{max}})$$

We formulate this as an additional additive and capture the additional smoothing step within the restriction operator, i.e. we used a smoothed restriction operator $\tilde{R} = RA_\ell M^{-1}_\ell$ when constructing the auxilliary grid error equation

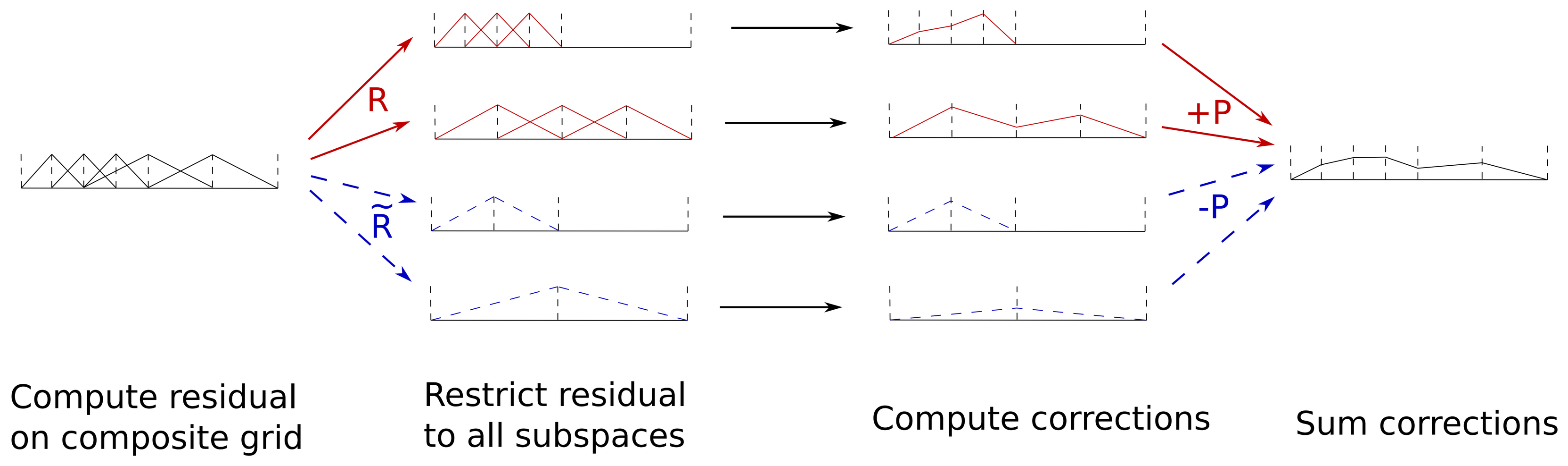

Figure 3: Data flow adaFAC-Jac as it restricts to different basis spaces to smooth and produce corrections

Our method is given the title adaFAC-Jac - an *a*dditively *d*amped AFAC with an additional *Jac*obi smoothing step

Algorithm 1 Blueprint of one sweep of the our adAFAC-Jac without AMR. R^i or P^i denote the recursive application of the restriction or prolongation, respectively. $\widetilde{R}^i$ applies R $(i-1)$ times, followed by an application of one smoothed operator. b_f is the right hand side of the fine grid equation.

```
function ADAFAC-JAC
    r_{ℓmax} ← b_f − A_{ℓmax} u_{ℓmax}
    for all ℓmin ≤ ℓ ≤ ℓmax do
        b_ℓ ← R^{ℓmax−ℓ} r_{ℓmax}                    ▷ Restrict fine grid residual to grid level ℓ
    end for
    for all ℓmin ≤ ℓ < ℓmax do
        b̃_ℓ ← R̃^{ℓmax−ℓ} r_{ℓmax}                    ▷ Additional restriction residual into additional grid space
    end for
    for all ℓmin < ℓ ≤ ℓmax do
        c_ℓ ← 0; c̃_ℓ ← 0                              ▷ Initial "guess" for correction and damping parameter
        JACOBI(A_ℓ c_ℓ = b_ℓ, ω)                       ▷ Iterate of correction equation stored in c_ℓ
        JACOBI(A_{ℓ−1} c̃_{ℓ−1} = b̃_{ℓ−1}, ω̃)          ▷ Iterate of damping equation (one level coarser)
    end for
    c_{ℓmin} ← 0
    JACOBI(A_{ℓmin} c_{ℓmin} = b_{ℓmin}, ω)            ▷ Coarsest grid update (no additional damping performed)
    u_{ℓmax} ← u_{ℓmax} + c_{ℓmin} + Σ_{ℓ=ℓmin−1}^{ℓmax} P^{ℓmax−ℓ} c_ℓ − P^{ℓmax−(ℓ−1)} c̃_{ℓ−1}
end function
```

Results 1

We verify consistency of our method with a known baseline: the poisson equation as grid levels increase with our methodology
Poisson equation sees increase in number of iterations before it reaches convergence—additional levels no longer see convergence
Our method does not suffer from this flaw

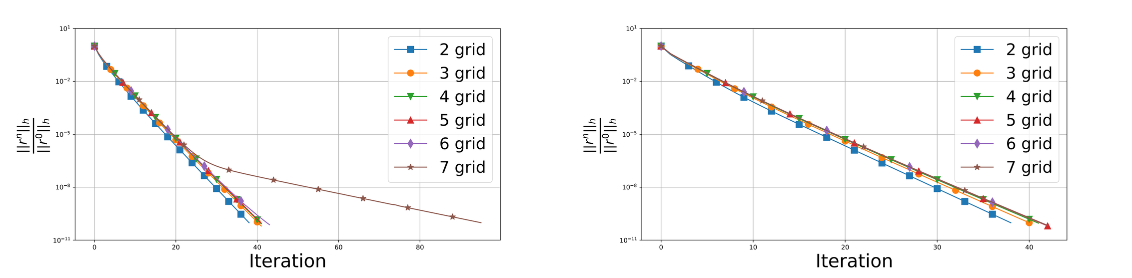

Figure 4: Additive Multigrid (left); adaFAC-Jac (right)

Results 2

Test on more challenging setup and with AMR
Comparison of processed DoFs (to show relative performance independent on specific mesh)
Many setups don't converge - ours do and see minimal deterioration in convergence rate as jump size increases and number of levels increases