AMG for Higher-Order Discretizations of Second-Order Elliptic PDEs

J. Ruge
S. McCormick
T. Manteuffel
J. Nolting

Why do I care?

Answer: FOSPACK

- First-Order System Least Squares (FOSLS). McCormick & Manteuffel.

- Finite element discretization on unstructured triangular/quadrilateral meshes.

- AMG Solution Method.

Ruge, McCormick, Brandt, & Stüben.

- Grid Generation (flexible, but basic).
- Equation interpreter (“user-friendly”).
- Allow for nonlinear problems.
- FMG with local/global refinement.

Goal: Introduce h-p local/global refinement.

Higher accuracy/computational cost.
Must solve if linear systems efficiently.

This talk does not cover:

- Accuracy of p-refinement methods.
- h-p refinement strategies.
- Cost effectiveness of p-refinement.
- FOSLS theory, applications, etc.

This talk covers:

- FOSLS basics (Discretization).
- AMG basics.
- Higher-Order finite elements.
- Results & conclusions (so far…)
- Future work.

FOSLS (First-Order System Least Squares)

Convert problem to first-order system $L\underline{u} = \underline{f}$:

$$L = \begin{bmatrix} L_1 \\ L_2 \\ \vdots \\ L_m \end{bmatrix} \quad \underline{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \quad \underline{f} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_m \end{bmatrix}$$

where L_i is a first-order operator operating on $\underline{u}$.

$$\left.\begin{aligned} L_1\underline{u} &= f^1 \\ L_2\underline{u} &= f^2 \\ &\vdots \\ L_m\underline{u} &= f^m \end{aligned}\right\} \text{ on } \Omega \quad (+ \text{ bc's}).$$

Form least-squares functional:

$$F(\underline{u};\underline{f}) = \sum_k \int_\Omega \left| f^k - L_k\underline{u} \right|^2 d\Omega \quad (+ \text{ bc terms})$$

The FOSLS Problem: minimize $F(\underline{u};\underline{f})$.

Properties of FOSLS formulation*:

- $F(\underline{u};\underline{f}) = F(\underline{e};\underline{0})$ is a natural error measure.
- Natural criterion for local refinement.
- F is H^1 equivalent (i.e., $F(\underline{e};0) \approx \sum_k \left\| e^k \right\|_1^2$).
- "Separates" components in a nice way.

Roughly, minimizing $F(\underline{u};\underline{f})$ is equivalent to solving $L^*L\underline{u} = L^*\underline{f}$.

$$\begin{bmatrix} L_{11} & L_{12} & .. & L_{1n} \\ L_{21} & L_{22} & .. & L_{2n} \\ : & : & :: & : \\ L_{n1} & L_{n2} & .. & L_{nn} \end{bmatrix} \begin{bmatrix} u^1 \\ u^2 \\ : \\ u^n \end{bmatrix} = \begin{bmatrix} g^1 \\ g^2 \\ : \\ g^n \end{bmatrix}$$

- Diagonal blocks are elliptic (Laplacian-like).
- Off-diagonal blocks are "small".
- Ideal for solution by Multigrid-like methods.
- Can use MG as block-diagonal preconditioner.

The Discretization (2D):

- Triangular/quadrilateral mesh T:

$$\Omega = \bigcup_{\alpha \in T} \alpha .$$

- Nodes $i = 1, 2, ..., N$.

- Space S of piecewise linear/bilinear functions. Nodal basis functions:

$$\phi_i = \begin{cases} 1 & \text{at node } i, \\ 0 & \text{at nodes } j \neq i. \end{cases}$$

Let $\underline{\phi}_i^j = (0,\ldots,\phi_i,\ldots,0)^t$ (ϕ_i in the j'th position).

A function $\underline{u}$ in the finite element space S^n is:

$$\underline{u} = \sum_{i=1}^{N} \sum_{j=1}^{n} u_i^j \underline{\phi}_i^j$$

Discretization continued...

$$F(\underline{u};\underline{f}) = \sum_k \sum_\alpha \int_\alpha (f^k - L_k \sum_i \sum_j u_i^j \phi_i^j)^2 d\Omega$$

$F(\underline{u};\underline{f}) \geq 0$ and quadratic in u_i^j, so to minimize, for I=1,...,N, J=1,...,m set

$$0 = \frac{\partial F}{\partial u_I^J}$$

$$= 2\sum_k \sum_\alpha \int_\alpha (L_k \phi_I^J)(L_k \sum_i \sum_j u_i^j \phi - f^k) d\Omega$$

or

$$\sum_i \sum_j \left[\sum_k \sum_\alpha \int_\alpha (L_k \phi_I^J)(L_k \phi_i^j) d\Omega \right] u_i^j$$

$$= \sum_k \sum_\alpha \int_\alpha (L_k \phi_I^J) f^k d\Omega$$

This gives the discrete matrix equation for u_I^J.
We need to be able to compute the integrals.

Computing the quantities (for fixed α)

$$\int_\alpha L_k(\phi_I^J)\,L_k(\phi_i^j)\,d\Omega \quad \text{and} \quad \int_\alpha L_k(\phi_I^J)\,f^k\,d\Omega$$

Integration by numerical quadrature. For $\alpha \in T$:

(*) $$\int_\alpha \beta(x,y)\,d\Omega \approx \sum_l w_l \beta(x_l, y_l)$$

Compute values for numerical differentiation:

(**)
$$\frac{\partial \beta}{\partial x}(x_l, y_l) \approx \sum_j c_{lj}^x \beta(x_j, y_j)$$
$$\frac{\partial \beta}{\partial y}(x_l, y_l) \approx \sum_j c_{lj}^y \beta(x_j, y_j)$$

To compute integrals:

- Obtain $\phi_I^J(x_l, y_l)$, $\phi_i^j(x_l, y_l)$, and $f^k(x_l, y_l)$.
- Using the instruction lists, compute
  $L_k\phi_I^J(x_l, y_l)$ and $L_k\phi_i^j(x_l, y_l)$, l = 1,...,n_q.
  - Apply dx and dy using (**).
  - Otherwise, apply operators pointwise.
- Apply (*) to products to obtain integrals.

Algebraic Multigrid.

- Uses multigrid principles to solve problem.
- Requires only the matrix.
- Automatically determines MG components.

Off-the-shelf AMG codes work well on:

- Elliptic problems.
- Unstructured meshes.
- Complicated domains.
- Varying or discontinuous coefficients.
- 2 and 3D problems.

AMG for a scalar problem $A^1 u^1 = b^1$ on Ω^1:

- (Setup) For k=1, 2, ...
  Choose $\Omega^{k+1} \subset \Omega^k$.
  Define interpolation P_{k+1}^k.
  Define restriction $P_k^{k+1} = (P_{k+1}^k)^t$.
  Define coarse grid matrix $A^{k+1} = P_k^{k+1} A^k P_{k+1}^k$.
- (Solve) Perform standard MG cycling.

AMG for Systems.

$$\begin{bmatrix} A_{11} & A_{12} & .. & A_{1n} \\ A_{21} & A_{22} & .. & A_{2n} \\ \vdots & \vdots & :: & \vdots \\ A_{n1} & A_{n2} & .. & A_{nn} \end{bmatrix} \begin{bmatrix} u^1 \\ u^2 \\ \vdots \\ u^n \end{bmatrix} = \begin{bmatrix} b^1 \\ b^2 \\ \vdots \\ b^n \end{bmatrix}$$

Could apply as a block-diagonal preconditioner:

$$A_{ii} u^i = b^i - \sum_{j \neq i} A_{ij} u^j$$

- Find coarse grids $\Omega_c^{(i)}$, interpolation $P^{(i)}$, etc.
- Solve each problem with 1 or more cycles.

Alternative: $P^{global} = \begin{bmatrix} P^{(1)} & 0 & .. & 0 \\ 0 & P^{(2)} & .. & 0 \\ \vdots & \vdots & :: & \vdots \\ 0 & 0 & .. & P^{(n)} \end{bmatrix}$

Then define restriction, CG matrix as usual.
Accounts for coupling on all levels. (faster)

AMG interpolation for scalar problems:

Assumes relaxation gives small residuals $Ae \approx 0$

Partition the mesh into C and F.

At an F-point i

$$a_{ii}e_i = -\sum_{k \in C_i} a_{ik}e_k - \sum_{j \in F} a_{ij}e_j$$

Make the approximation

$$e_j \approx \sum_{k \in C_i} a_{jk}e_k \Big/ \sum_{k \in C} a_{jk}$$

Combine to get interpolation to point i.

Note: This assumes that constants functions (in the coefficient space) are algebraically smooth. This is generally true for PDEs discretized with a nodal basis.

Higher-order elements (p-refinement):

Introduce higher-order polynomials over each element. (Still C_0 across element boundaries).

- Start with unstructured mesh.
- Isoparametric elements.
- Nodes correspond to uniform refinement of a canonical triangular or quadrilateral element.

To allow for “easy” adaptive p-refinement:

Start with *k* additional points/side, then refine further by a factor of 2.
(*k* is global & fixed for the run.)

k=1, *p* = 1, 2, 4, 8, …
k=2, *p* = 1, 3, 6, 12, …

Helps in varying *p* across elements.

Choice of basis:

1. Standard nodal basis (obvious choice)

2. Hierarchical basis:

In 1D, introduce basis functions at new nodes:

Level 0 functions – linears.
Level 1 functions – k+1 order polynomials.
Level 2 functions – 2(k+1) order polynomials.
.
.

Note: Level m functions ($m \geq 1$) are a subset of the nodal basis for order $2^{m-1}(k+1)$ polynomials.

Each level gives a “correction” to previous one.

On quadrilaterals, use tensor products.

Advantages: Simpler coding, local p-refinement, FMG implementation, AMG coarsening,…

AMG Expectations:

1. Off-the-shelf AMG would degrade quickly with *p* for the nodal basis discretizations.

2. Off-the-shelf AMG would converge very poorly for the hierarchical basis (violates assumption used in interpolation).

3. *p*-coarsening using defect correction plus smoothing would work well.

4. AMG could easily be modified to deal with the hierarchical basis functions. (Coarsen through p-levels with trivial intergrid transfer operators.)

5. Could use structure of p-nodes (although underlying mesh is unstructured).

6. I don’t expect uniform convergence in *p*.

Off-The-Shelf AMG Results:

Test problem: Poisson equation on [0,1]x[0,1].

1. Nodal basis functions. (1,1) V-cycle.

p	Mesh size	entries per row	c_A	AMG conv. ρ
1	128^2	8.9	1.46	0.10
2	128^2	15.9	1.53	0.16
3	64^2	25.5	2.30	0.22
4	64^2	35.8	2.96	0.34
5	48^2	48.7	2.94	0.82
6	32^2	63.4	2.60	0.95
7	32^2	80.4	2.92	0.97

2. Hierarchical basis functions. (1,1) V-cycle.

p	Mesh size	entries per row	c_A	AMG conv. ρ
2	64^2	8.9	2.74	0.98
3	64^2	15.9	6.72	0.96

AMG for Hierarchical Basis Functions:

Method: coarsen by p-refinement levels to $p = 1$, then apply standard AMG.

For fixed k and m p-refinement levels…

AMG level	p-level	p
1	m	$2^{m-1}(k+1)$
2	m-1	$2^{m-2}(k+1)$
.		
m+1	0	1

Standard AMG is applied to the level m+1 matrix.

Note: Coarsening from level 1 to m+1 is “free.”

B_0 = The set of linear basis functions, and
B_n = The set of hierarchical functions introduced on *p*-level *n*, for *n*=1,…,*m*.
A_{ij} = Submatrix connecting B_i and B_j.

Interpolation is simple embedding:

$$P_n^{n+1} = \begin{bmatrix} I_0 & 0 & .. & 0 \\ 0 & I_1 & .. & 0 \\ : & : & :: & : \\ 0 & 0 & .. & I_n \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The level *n* matrix (for *n* =0,…,*m*) is:

$$\begin{bmatrix} A_{00} & A_{01} & .. & A_{0n} \\ A_{10} & A_{11} & .. & A_{1n} \\ : & : & :: & : \\ A_{n0} & A_{n1} & .. & A_{nn} \end{bmatrix}$$

Thus level 1,…,*m*+1 AMG matrices are nested.

Results for (1,1) V-cycle.

k,m	*p*	c_A	ρ
1,1	2	1.16	0.67
2,1	3	1.05	0.89
3,1	4	1.02	0.97
1,2	4	1.13	0.79
2,2	6	1.10	0.90

What happened?

Results for (5,5) V-cycle.

k,m	*p*	ρ	$\rho^{1/5}$
1,1	2	0.15	0.68
2,1	3	0.55	0.89
3,1	4	0.85	0.97
1,2	4	0.31	0.79
2,2	6	0.69	0.93

Smoothing is too slow.
Too much “distance” between grids?

Options…

1. Get a better smoother.
2. Make coarse grid closer to fine grid (c_A low).

Try better smoother…

Hierarchical coarsening, element smoothing (simultaneously relax all nodes in element):

k,m	*p*	ρ
1,1	2	0.28
2,1	3	0.26
3,1	4	0.57
1,2	4	0.53
2,2	6	0.72

Some improvement.
Work/point in element relaxation is large.

Try better coarse grid…

Defect correction + smoothing.
Keep same nodes, reduce order.
Use nodal basis in these tests.
Gauss-Seidel relaxation.
(1,1) V-cycle

p	$p_{coarser}$	c_A	ρ
2	1	1.75	0.29
3	1	1.48	0.15
4	1	1.34	0.55
4	2,1	1.78	0.40
6	3,1	1.59	1.83

Some good results for small p.
More complicated to implement.
Non-variational coarsening. Can diverge.
More smoothing may help.

Additional relaxation sweeps on p levels.

p	$p_{coarser}$	sweeps	ρ
2	1	2,2	0.10
3	1	2,2	0.09
4	1	2,2	0.12
4	2,1	3,3	0.11
6	3,1	5,5	0.47

Good results.
Same drawbacks as above.

Conclusions:

Hierarchical approach not very promising.
OTS AMG easiest & most efficient for small *p*.
Good AMG convergence is possible.
Results preliminary. Further study needed.

Further work:

Improve AMG for direct application.

Use a modified smoothness assumption for interpolation using level numbers of nodes.

Need better approaches where p–refinement is used locally. (*p* tau extrapolation?)

Large *p* may only be needed locally. Can afford to do more & better smoothing.