Symbolic computation and finite element methods

Veronika Pillwein*

RISC, Johannes Kepler University, Linz, Austria,
veronika.pillwein@risc.jku.at,
WWW home page: https://www.risc.jku.at/home/vpillwei

Abstract. During the past few decades there have been many examples where computer algebra methods have been applied successfully in the analysis and construction of numerical schemes, including the computation of approximate solutions to partial differential equations. The methods range from Gröbner basis computations and Cylindrical Algebraic Decomposition to algorithms for symbolic summation and integration. The latter have been used to derive recurrence relations for efficient evaluation of high order finite element basis functions. In this paper we review some of these recent developments.

Keywords: computer algebra, symbolic summation, cylindrical algebraic decomposition, finite element methods

1 Overview

Many problems in science and engineering are described by partial differential equations on non-trivial domains which – except in special cases – can not be solved analytically. Numerical methods such as finite difference methods (FDM) or finite element methods (FEM) are used to solve these equations. In the past decades, FEM have become the most popular tools for obtaining solutions of partial differential equations on complicated domains [9],[12, 13]. The main advantage of finite element methods is their general applicability to a huge class of problems, including linear and nonlinear differential equations, coupled systems, varying material coefficients and boundary conditions.

Symbolic computation has in the last decades gained importance concerning ease of use and range of applicability in part due to the availability of bigger and faster computers and the development of efficient algorithms. There are hybrid symbolic-numeric algorithms for computing validated results, but there are also examples of collaborations where symbolic methods have been used to analyze numerical schemes, to develop new ones, and to simplify or speed up existing ones. We begin by giving a brief overview on some of these cooperations.

In many numerical methods, large scale linear systems of equations need to be solved and one popular technique for speeding up the iterative solution process is

* Supported by the Austrian FWF grants F50-07 and W1214.

using preconditioning. For this preconditioning matrices that are spectrally close to the given matrix need to be constructed. Langer, Schicho, and Reitzinger [24] carried this out by symbolically solving the optimization problem arising in this construction. For this type of problems Cylindrical Algebraic Decomposition (CAD) as well as Gröbner bases are applicable (in theory), but practically the issue of computational complexity has to be circumvented.

Levandovskyy and Martin [25] applied CAD to derive von Neumann stability of a given difference scheme. They also presented symbolic approaches to derive a finite difference scheme for a single PDE, where they focused on linear PDEs with constant coefficients as well as the computation of dispersion relations of a linear PDE in a symbolic, algorithmic manner. These tools were also implemented in the computer algebra system Singular.

CAD has been applied earlier by Hong, Liska, and Steinberg [18] in the analysis of (systems of) ordinary and partial differential-difference equations, where the necessary conditions for stability, asymptotic stability and well-posedness of the given systems were transformed into statements on polynomial inequalities using Fourier or Laplace transforms.

Very recently, Cluzeau, Dolean, Nataf and Quadrat [14, 15] have used algebraic and symbolic techniques such as Smith normal forms and Gröbner basis techniques to develop new Schwarz-like algorithms and preconditioners for linear systems of partial differential equations.

High order FEM are often preferrable in the numerical solution of PDEs because of their fast convergence, but they require high computational effort. Hence, every simplification is welcome. In [2], two types of high order basis functions with good numerical properties were constructed. By means of symbolic summation algorithms recurrence relations for the fast evaluation of these basis functions were derived entirely automatically.

Koutschan, Lehrenfeld and Schöberl [23] were concerned with an efficient implementation of a numerical discretization of the time-domain Maxwell's equations. As part of this algorithm the tensor product structure of the basis functions and mixed difference-differential relations satisfied by the underlying orthogonal polynomials were exploited. Below we comment on this type of relations for the basis functions discussed next.

For the iterative solution of linear systems it is convenient, if the matrices are sparse. For different types of partial differential equations and the method of high order finite elements on simplices (i.e., triangles and tetrahedra), families of basis functions have been proposed [4] that yield sparse system matrices. The proof of sparsity was carried out using symbolic computation. We review some of these results and give some recurrences that can be used in fast computation of the matrix entries.

As another specific example we present symbolic local Fourier analysis (sLFA) below. Local Fourier analysis (LFA) has been introduced by Brandt [10] to analyze multigrid methods. Multigrid methods are common iterative solvers for the large systems arising in FEM or FDM. LFA gives quantitative statements on the methods under investigation, i.e., it leads to the determination of sharp convergence rates. Typically only convergence is proven for these methods and the convergence rate is estimated using brute force numerical interpolation. In [27] we introduced sLFA to derive a closed form upper bound for the convergence rate of a model problem with a particular solver.

2 Brief Introduction to FEM

Before we give two specific examples of a symbolic-numeric cooperation we briefly introduce the basic concept of FEM. Finite element methods are based on the variational formulation of partial differential equations. Let us consider as an example the following simple problem: given f, find u (both from appropriate function spaces) s.t.

$$u(x) - \varDelta u(x) = f(x),$$

in a given domain $\varOmega \subseteq \mathbb{R}^d$, $d = 1, 2, 3$, where $\varDelta$ denotes the standard Laplace operator $\varDelta u(x) = \sum_{j=1}^{d} \frac{\partial u}{\partial x_j}$. The PDE above is multiplied on both sides by a smooth function v that vanishes at the boundary of $\varOmega$ and then integrated over the given domain. By partial integration (exploiting the compact support of v), we obtain the variational formulation of the given PDE

$$\int_{\varOmega} u(x)v(x)\,dx + \int_{\varOmega} \nabla u(x) \cdot \nabla v(x)\,dx = \int_{\varOmega} f(x)v(x), \tag{2.1}$$

where ∇ denotes the gradient operator. In this variational problem the task is: given f, find u (in a less restrictive function space) s.t. (2.1) is satisfied for all test functions v, i.e., all smooth functions vanishing on the boundary of the given domain.

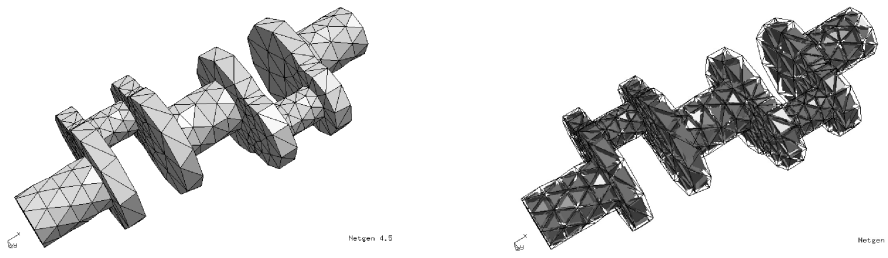

Fig. 1. FE mesh for a crankshaft, left: surface mesh; right: interior tetrahedral elements

For numerically determining an approximate solution to this problem, the domain of interest is subdivided into simple geometrical objects such as triangles, quadrilaterals, tetrahedra, or hexahedra. The approximate solution is expanded in a (finite) basis of local functions $\{\phi_i\}_{i=1}^{N}$, each supported on a finite number of elements in the subdivision. Continuing with our example (2.1), we replace the solution $u(x)$ by the approximate solution $u_h(x) = \sum_{i=1}^N u_i\phi_i(x)$ and use $\phi_j(x)$ as test functions. This yields the system of equations

$$\sum_{i=1}^N u_i \underbrace{\int_\Omega (\phi_i(x)\phi_j(x) + \nabla\phi_i(x)\cdot\nabla\phi_j(x))\, dx}_{=:A_{i,j}} = \underbrace{\int_\Omega f(x)\phi_j(x)}_{=:b_j}$$

to be solved for the unknown coefficients u_i. If we define the matrix $A = (A_{i,j})_{i,j=1}^N$ and the vectors $b = (b_j)_{j=1}^N$, $u = (u_i)_{i=1}^N$, then the above can be written as the linear system $Au = b$ to be solved for u.

In general, the FE discretization of a PDE yields a (usually large) system of linear equations that is commonly solved using iterative methods. There are three main strategies to improve the accuracy of the approximate solution.

(i) The classical approach is to use on each element basis functions of a fixed low polynomial degree, say $p = 1, 2$, and to increase the number of elements in the subdivision. This strategy of local or global refinement of the mesh is called the h-version of the finite element method, where h refers to the diameter of the elements in the subdivision. With this approach, the approximation error decays algebraically (i.e., with polynomial rate) in the number of unknowns.

(ii) An alternative strategy is to keep the mesh fixed and to locally increase the polynomial degree p of the basis functions. This method is called the p-version of the finite element method [28, 29] and in the case of a smooth solution leads to exponential convergence with respect to the number of unknowns. But in practical problems the solutions usually are not smooth. In this case the convergence rate of the p-method again degenerates to an algebraic one.

(iii) Exponential convergence can be regained by combining both strategies in the hp-version of the FEM [16],[19],[28]. On parts of the domain where the sought solution is smooth, few coarse elements with basis functions of high polynomial degrees are used, whereas in the presence of singularities (caused, e.g., by re-entrant corners) the polynomial degree is kept low and the mesh is refined locally towards the singularity. The p- and the hp-method are also referred to as high(er) order finite element methods.

FE basis functions are usually defined on some reference element and then transformed to the actual element in the mesh. In the case of one dimension the elements in the subdivision are just intervals and as reference element we choose $\hat{I} = [-1, 1]$.

We give an example for a hierarchic high order finite element basis, i.e., if the polynomial degree is increased, the set of basis functions is incremented keeping the previously basis functions, in contrast to nodal basis functions such as Lagrange polynomials where every time the whole set of basis functions needs to be replaced. The lowest order basis functions are the hat functions $\hat\phi_0(x) = \frac{1-x}{2}, \hat\phi_1(x) = \frac{1+x}{2}$ that are also referred to as vertex basis functions.

They are the piecewise linear functions that are 1 on the defining vertex and vanish in all other vertices. The vertex basis functions are supported on the two neighbouring elements sharing the defining vertex. For the higher order basis functions of degree $i \geq 2$ a common choice for a conforming basis for our model problem are integrated Legendre polynomials. Different PDEs require different continuity of the given basis and for the present problem the basis functions are requried to be continuous across element interfaces. This property is guaranteed by construction. Let $P_n(x)$ denote the nth Legendre polynomial [1]. Then, for $n \geq 1$ the nth integrated Legendre polynomial $\hat{p}_n^0(x)$ is defined as

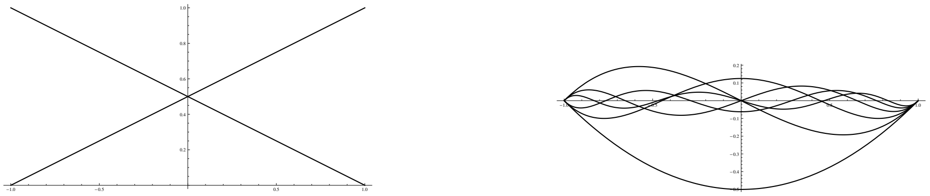

Fig. 2. FE basis functions in 1D, lowest order and higher order for degrees $i = 2, \ldots, 7$

$$\hat{p}_n^0(x) := \int_{-1}^{x} P_{n-1}(s)\, ds.$$

Note that for $n \geq 2$, $\hat{p}_n^0(-1) = 0$ (for obvious reasons) and that $\hat{p}_n^0(1) = 0$ (by the Legendre $L^2(-1,1)$-orthogonality). The basis functions $\hat{\phi}_i(x) = \hat{p}_i^0(x)$ for $i \geq 2$ are supported on a single element and are also referred to as cell based basis functions.

In the definition of the hierarchic basis in two and three dimensions besides vertex and cell based basis functions in addtion edge and face based basis functions are constructed. Edge based basis functions are nonzero on the defining edge and vanish on all others and are supported on only those elements sharing the defining edge. Face based basis functions are nonzero on the defining face and vanish on all other faces and are supported on the two elements sharing the defining face. All basis functions are collected in the vector of basis functions $\phi = (\phi_1, \ldots, \phi_N)$. Then the system matrix A is already sparse because only some of the basis functions forming its entries share common support.

One reason, why integrate Legendre polynomials are used as basis functions is that they are orthogonal w.r.t. the inner product $\langle f, g \rangle = \int_{-1}^{1} f'(x) g'(x)\, dx$ and the L^2 inner product $\int_{-1}^{1} \hat{p}_i^0(x) \hat{p}_j^0(x)\, dx$ vanishes if $|i - j| \neq 0, 2$. Hence, the one-dimensional basis introduced above is constructed to yield an even sparser system matrix. For quadrilateral or hexahedral elements the basis functions are typically defined as tensor products of 1D basis functions and thus many of the sparsity properties can be inherited.

But for approximating complicated domains often triangular or tetrahedral elements are preferred. These elements are usually considered as collapsed quadrilateral or hexahedron. The construction of basis functions that yield sparse system matrices for these elements and that satisfy continuity requirements imposed by the underlying PDE (i.e., the underlying function spaces) is not obvious.

3 Sparsity Optimized Basis Functions

Beuchler and Schöberl [8] introduced triangular basis functions that yield sparse system matrices of the type (2.1). These basis functions are defined using integrated Jacobi polynomials [1]. Let $P_n^{(\alpha,\beta)}(x)$ denote the nth Jacobi polynomial. Then, we define the nth integrated Jacobi polynomial as

$$\hat{p}_n^{\alpha}(x) := \int_{-1}^{x} P_{n-1}^{(\alpha,0)}(s)\, ds.$$

Integrated Legendre polynomials are special cases of these polynomials as $P_n(x) = P_n^{(0,0)}(x)$. The sparsity of the system matrix has been proven by explicitly computing the matrix entries by exploiting relations between Jacobi polynomials of different parameters. It has been shown that the rows of the inner block of the system matrix have a constant number of nonzero matrix entries independent of the maximal polynomial degree.

An extension of this construction to tetrahedral elements was presented in [3]. Again the proof of sparsity of the system matrix was accomplished by explicitly computing the matrix entries. In three dimensions this was no longer feasible to be carried out by hand. Hence the computations were handed over to a computer program that evaluated the integrals with a rewriting procedure. In this process relations between Jacobi polynomials of different parameters were again exploited. Let us note that these relations can be discovered and proven using automatic tools for symbolic summation. In our work, we applied the packages HolonomicFunctions [22] and SumCracker [20], both implemented in Mathematica.

Subsequently, the construction of basis functions was extended to cover further function spaces (i.e., other types of PDEs) [6, 7]. In all these cases the proof of sparsity was carried out using symbolic computation. The nonzero matrix entries of the different system matrices are rational functions in the polynomial degrees of the basis functions. The sparsity already eases handling the system matrices. Furthermore in [5] an algorithm was presented that computes the matrix entries in optimal complexity based on sum factorization and exploiting the nonzero pattern of the system matrix. In the remainder of this section we want to point out how the techniques of [23] can also be applied to the sparsity optimized basis functions and how recurrences can be used to quickly set up the system matrices.

As a specific example, we consider the following cell based basis functions defined on the triangular reference triangle $\hat{T}$ with vertices $(-1,-1), (1,-1), (0,1)$

$$\hat{\phi}_{i,j}(x,y) = \hat{p}_i^0\left(\frac{2x}{1-y}\right)\left(\frac{1-y}{2}\right)^i \hat{p}_j^{2i}(y),$$

that are optimal for the example PDE stated in the previous section. In [23] cell based basis functions that are orthogonal on the reference triangle w.r.t. the L^2 inner product were used. In the algorithm to evaluate the gradients of the basis functions, relations between shifts of Jacobi polynomials and their derivatives (that are Jacobi polynomials with different parameters) were used. Furthermore, these relations are required to have coefficients that are independent of the variables x and y as they are to be turned into relations between integrals over products of these basis functions. Such relations can be computed automatically using Koutschan's package HolonomicFunctions. To facilitate computations, we use a different representation of the integrated Jacobi polynomials that relates them to classical Jacobi polynomials,

$$\hat{p}_n^\alpha(x) = \frac{1+x}{n} P_{n-1}^{(\alpha-1,1)}(x), \quad \hat{p}_m^0(x) = \frac{x^2-1}{2(m-1)} P_{m-2}^{(1,1)}(x),$$

valid for $n \geq 1$ and $m \geq 2$. Then, in order to obtain the corresponding mixed relations for $\hat{\phi}_{i,j}(x,y)$ as defined above, one proceeds as follows:

In[1]:= $\mathbf{ann = Annihilator[\hat{\phi}_{i,j}[x,y], \{S[i], S[j], Der[x], Der[y]\}]}$

Here S$[n]$ (and below S_n) denotes the forward shift in n and Der$[x]$ (and below D_x) the derivative w.r.t. x. This command quickly delivers the annihilating ideal for the given input by performing the necessary closure properties. To determine a relation of the type we need in this annihilating ideal we use the command "FindRelation":

In[2]:= $\mathbf{FindRelation[ann, Eliminate \to \{x,y\}, Pattern \to \{\_,\_,0 \mid 1, 0\}]}$

Out[2]= $\{(2i+j+3)(2i+j+4)(2i+2j+3)S_i^2 S_j^2 D_x - (j+3)(j+4)(2i+2j+3)S_j^4 D_x - 4(j+1)(i+j+2)(2i+j+3)S_i^2 S_j\ D_x + 4(j+3)(i+j+2)(2i+j+1)S_j^3 D_x + j(j+1)(2i+2j+5)S_i^2 D_x - 2(2i+1)(i+j+2)(2i+2j+3)(2i+2j+5)S_i S_j^2 - (2i+j)(2i+j+1)(2i+2j+5)\ S_j^2 D_x\}$

In[3]:= $\mathbf{rel = FindRelation[ann, Eliminate \to \{x,y\}, Pattern \to \{\_,\_,0,0 \mid 1\}];}$

In[4]:= $\mathbf{Support[rel]}$

Out[4]= $\{\{S_i^2 S_j^4 D_y, S_j^6 D_y, S_i^2 S_j^3 D_y, S_j^5 D_y, S_i^2 S_j^3, S_i^2 S_j^2 D_y,\ S_j^5, S_j^4 D_y, S_i^2 S_j^2, S_i^2 S_j D_y, S_j^4, S_j^3 D_y, S_i^2 S_j, S_i^2 D_y, S_j^3,\ S_j^2 D_y\}\}$

The option "Eliminate" specifies which variables are not to occur in the coefficients and the option "Pattern" defines the admissible exponents for the operators, i.e., in the first call above any exponent is allowed for the shift operators, whereas D_x may occur with power at most one and D_y must not be in the result altogether. For the second example we only display the support of the resulting relation for sake of space. For this example, the computation times are still neglible. The same procedure can be carried out for the tetrahedral basis functions; however the comuputational effort increases significantly.

Another way to speed up the computations is to use recurrence relations for the nonzero parts. We continue with the example from the previous section and consider the two integrals constituting the system matrix A separately. If

$$M_{ij;kl} = \int\int_{\hat{T}} \hat{\phi}_{i,j}(x,y)\hat{\phi}_{k,l}(x,y)\, d(x,y),$$

then the matrix built from these entries is nonzero only if $|i-k| \in \{0,2\}$ and $|i-k+j-l| \leq 4$. In terms of the offset we have for $f(d,i,j,l) = M_{i,j;i+d,l}$ for $d = 0,2$ that

$$\begin{aligned} f(d,i,j,l) = {} & \frac{2i+j+l-8+d}{2i+j+l+2+d} f(d,i,j-1,l-1) \\ & + \frac{j-l-5-d}{2i+j+l+2+d} f(d,i,j-1,l) \\ & - \frac{j-l+5-d}{2i+j+l+2+d} f(d,i,j,l-1), \end{aligned}$$

with different initial values. This recurrence was determined using multivariate guessing with Kauers' package Guess [21]. Since all matrix entries have been computed explicitly, the correctness of the recurrence can easily be verified.

For the second integral defining the system matrix A, the following integrals need to be computed:

$$\begin{aligned} K^{xx}_{ij;kl} &= \int\int_{\hat{T}} \frac{d}{dx}\hat{\phi}_{i,j}(x,y)\frac{d}{dx}\hat{\phi}_{k,l}(x,y) d(x,y), \\ K^{yy}_{ij;kl} &= \int\int_{\hat{T}} \frac{d}{dy}\hat{\phi}_{i,j}(x,y)\frac{d}{dy}\hat{\phi}_{k,l}(x,y) d(x,y). \end{aligned}$$

The sparsity result for these matrices yields that $K^{xx}_{ij;kl}$ is nonzero only if $k = i$ and $|j-l| \leq 2$, and, $K^{yy}_{ij;kl}$ is nonzero only if $|i-k| \in \{0,2\}$ and $|i-k+j-l| \leq 2$ (i.e., in both cases we have nonzero only if $|i-k+j-l| \leq 2$). With $g(d,i,j,l) = K^{\xi\zeta}_{i,j;i+d,l}$ either of the matrix entries above in terms of the corresponding offsets satisfies the recurrence

$$\begin{aligned} g(d,i,j,l) = {} & \frac{2i+j+l-6+d}{2i+j+l+d} g(d,i,j-1,l-1) + \frac{j-l-3-d}{2i+j+l+d} g(d,i,j-1,l) \\ & - \frac{j-l+3-d}{2i+j+l+d} g(d,i,j,l-1), \end{aligned}$$

with different initial values. Let us also note that the integrals over the mixed products $\frac{d}{dx}\hat{\phi}_{i,j}(x,y)\frac{d}{dy}\hat{\phi}_{k,l}(x,y)$ (that enter in a more general PDE allowing for a coefficient matrix) satisfy the same recurrence.

4 Symbolic Local Fourier Analysis

The typically large linear systems that arise in FEM are usually solved approximately using some iterative scheme. The multigrid method is such an iterative solver and it operates on (at least) two grids in the finite element discretization. It has two main features: smoothing on the finer grid and error correction on the coarser grid. Intuitively speaking, the smoother is applied to dampen out the oscillatory part of the error. After applying a few such steps, the smooth part of the defect is dominant and the coarse-grid correction takes care of the low-frequency modes of the overall error.

In multigrid theory commonly convergence is proven, but neither sharp nor realistic bounds for convergence rates are given [17]. Local Fourier analysis [10] is a technique to analyze multigrid methods (and also various other numerical methods) that gives quantitative estimates, i.e., it leads to the determination of sharp convergence rates. Although it can be justified rigorously only in special cases such as rectangular domains with uniform grids and periodic boundary conditions, still results obtained with local Fourier analysis can be carried over rigorously to more general classes of problems [11].

In order to obtain a bound for the convergence rate, the supremum of some rational function needs to be computed. Usually this is merely resolved by numerical interpolation. With algorithms carrying out quantifier elimination such as CAD it is possible to determine an exact bound. This combination of classical local Fourier analysis and symbolic computation we refer to as symbolic local Fourier analysis (sLFA) [26, 27]. The proposed approach is certainly applicable to different kinds of problems and different types of solvers. Next, we give an overview of the main results of [27].

The particular model problem that we consider is a PDE-constrained optimization problem which has to be discretized so that numerical methods can be applied. It is an optimal control problem of tracking type: given a desired state y_D and a regularization (or cost) parameter $\alpha > 0$, find a state y and a control u s.t. they minimize

$$J(y,u) := \frac{1}{2}\int_{\Omega} (y(x) - y_D(x))^2 \, dx + \frac{\alpha}{2}\int_{\Omega} u^2(x)\, dx,$$

subject to the elliptic boundary value problem

$$-\Delta y = u \quad \text{in}\Omega \quad \text{and} \quad y = 0 \quad \text{on } \partial\Omega,$$

where $\Omega \subseteq \mathbb{R}^2$ is a given domain with sufficiently smooth boundary. For the discretization we use FEM with rectangular elements and bilinear basis functions (i.e., low order FEM). As a solution method we apply a two-grid algorithm. The steps in the iteration consist of some ν_1 (pre-)smoothings steps, then the coarse grid correction, followed by some ν_2 (post-)smoothing steps. The iteration matrix for this can be written as

$$TG_k^{k-1} := S_k^{\nu_2} \underbrace{\left(I - P_{k-1}^k A_{k-1}^{-1} P_k^{k-1} A_k\right)}_{CG_k^{k-1} :=} S_k^{\nu_1}, \tag{4.2}$$

where A_k and A_{k-1} denote the finite element system matrix on the finer and coarser level, respectively, P_{k-1}^k and $P_k^{k-1} = \left(P_{k-1}^k\right)^T$ are the intergrid transfer operators prolongation and restriction, I denotes the identity matrix, and S_k is the matrix of the smoothing step

$$S_k = I - \tau \hat{A_k}^{-1} A_k$$

with preconditioner $\hat{A_k}^{-1}$. The matrix CG_k^{k-1} above is the iteration matrix of the coarse grid correction, i.e., at the coarse grid an exact solve is performed. Summarizing, if $\bar{x}_k$ denotes the exact solution on the fine grid, and $x_k^{(n)}, x_k^{(n+1)}$ the nth and $n+1$st iterate, then

$$x_k^{(n+1)} - \bar{x}_k = TG_k^{k-1} \left(x_k^{(n)} - \bar{x}_k \right).$$

The convergence rate of the two-grid method can be bounded from above by the matrix norm of the iteration matrix, i.e.,

$$q \leq q_{TG} = \|TG_k^{k-1}\|.$$

This estimate is sharp if we consider the supremum over all possible starting values or, equivalently, all possible right-hand sides. If $q_{TG} < 1$ then the method converges for all starting values. If furthermore q_{TG} is independent of the mesh size and regularization parameter (or any other parameter) then the convergence is robust and optimal, i.e., the number of iterations does not depend on the parameters. In local Fourier analysis the iteration matrix is replaced by its symbol, where for our purpose it is sufficient to know that it is a matrix of fixed finite dimension. For the model problem at hand with the given discretization it is an 8×8 matrix. Instead of bounding the norm of the iteration matrix TG_k^{k-1} itself, the norm of the symbol is bounded, which can be expressed as the spectral radius of a certain matrix. For the full definitions of these matrices we refer (for sake of space) to [27] and its accompanying notebooks[1]. Up to this point we followed the steps of classical local Fourier analysis and ususally the bound for the spectral radius is approximated using numerical interpolation. After some suitable substitutions the entries of the symbol are rational functions and the supremum of the spectral radius can be computed explictely (at least in theory) using CAD.

As it turned out, only for the one-dimensional case, the spectral radius of the 4×4 two-grid iteration matrix could be estimated fully in an all-at-once approach. In two dimensions, the size of the entries of the involved matrices became prohibitively large. Hence, we proceed in two steps and treat the smoothing rate and the coarse grid correction separately and obtain this way an upper bound for the convergence rate. This task is computationally rather ambitious. Let $\mathcal{N}$ denote the matrix under discussion. It is a symmetric matrix that depends on four parameters $(c_1, c_2, \eta, q) \in [0, 1]^2 \times (0, \infty)^2$ (the mesh parameter h and the

[1] available for download at http://www.risc.jku.at/people/vpillwei/sLFA/

regularization parameter α are hidden by substitution in the parameter η). After pulling out the common denominator

$$256\left(16c_2^4c_1^4\eta+16c_2^2c_1^4\eta+4c_1^4\eta+16c_2^4c_1^2\eta+16c_2^2c_1^2\eta+4c_1^2\eta+4c_2^4\eta+4c_2^2\eta\right.$$
$$\left.+144c_2^4c_1^4-72c_2^2c_1^4+9c_1^4-72c_2^4c_1^2-126c_2^2c_1^2+36c_1^2+9c_2^4+36c_2^2+\eta+36\right),$$

the matrix has polynomial entries of degrees up to 6 in c_1, c_2 and degrees up to 4 in q. Already the symbolic computation of the eigenvalues is not possible using the built-in functions of a computer algebra system. Since the matrix depends polynomially on q, it is possible to determine the eigenvalues using exact interpolation in q. This way we find that the eigenvalues are given by

$$0,\ q^4,\ \left(e(q)+\sqrt{d(q)}\right)\quad\text{and}\quad\left(e(q)-\sqrt{d(q)}\right)$$

with multiplicity two each, where e and d are rational functions in the unknowns c_1, c_2 and η that are too large to be displayed here. With the eigenvalues at hand it is readily verified that the largest eigenvalue in absolute value is $e(q)+\sqrt{d(q)}$. It remains to determine a bound for this eigenvalue uniformly in the parameters c_1, c_2, and η depending on the remaining variable q (the smoothing rate).

In theory, this bound can be determined entirely automatically using CAD. In practice, however, these computations become too complicated to be carried out in reasonable time. By considering the boundary of the parametric domain for (c_1, c_2, η) the following simple guess for this bound could be found:

$$q_{GUESS}(q) := \begin{cases} \frac{q^2+3}{4}, & 0 < q \le Q, \\ q\sqrt{q^2+1}, & Q < q < 1. \end{cases}$$

The proof that this guess actually gives the true bound can be carried out with the aid of CAD, if the calculations are broken down to smaller pieces. This bound of the convergence rate is not sharp – in fact numerical experiments indicate much faster convergence. This is because the analysis had to be carried out in two steps. Nonetheless, the result gives quantitative statements on both, the choice of the parameter τ used in the smoothing iteration, and the number of smoothing steps ν that have to be applied. Such results cannot be obtained using a classical analysis based on smoothing and approximation property, even though the choice of the parameters is a key issue in the implementation of a numerical method.

It would be desirable to carry out sLFA all at once on the full iteration matrix. Even though this is currently out of scope of our computational capabilities, the two-step analysis carried out above hints on a heuristic procedure to determine the convergence rate of the full method depending on the damping parameter τ of the smoothing iteration.

Let $\sigma(c_1, c_2, \eta, \tau)$ be the spectral radius of the symbol (an 8×8 matrix) of the two-grid iteration matrix TG_k^{k-1} as defined in (4.2). Then in the all-at-once approach we need to bound $\sigma(c_1, c_2, \eta, \tau)$ uniformly in $(c_1, c_2, \eta) \in [0,1]^2 \times (0, \infty)$, i.e., we need to compute

$$q_{TG}^2(\tau) = \sup_{(c_1,c_2)\in[0,1]^2}\ \sup_{\eta>0} \sigma(c_1, c_2, \eta, \tau).$$

As the supremum is the least upper bound, $q^2_{TG}(\tau)$ is the smallest $\lambda(\tau)$ satisfying

$$\forall\,(c_1,c_2)\in[0,1]^2\,\forall\,\eta>0\colon\sigma(c_1,c_2,\eta,\tau)<\lambda.$$

Computing $\lambda(\tau)$ could be done by quantifier elimination using CAD entirely automatically, provided that the spectral radius is given (and an algebraic function). The symbol of the iteration matrix is non-symmetric with only few zero entries and again everything can be brought to a common denominator

$$\begin{aligned}&16384(\eta+36)^2\left(16c_2^4c_1^4\eta+16c_2^2c_1^4\eta+4c_1^4\eta+16c_2^4c_1^2\eta+16c_2^2c_1^2\eta\right.\\&+4c_1^2\eta+4c_2^4\eta+4c_2^2\eta+144c_2^4c_1^4-72c_2^2c_1^4+9c_1^4-72c_2^4c_1^2-126c_2^2c_1^2\\&\left.+36c_1^2+9c_2^4+36c_2^2+\eta+36\right).\end{aligned}$$

The matrix entries are now polynomials of degrees either 9 or 10 in c_1, c_2, degrees either 2 or 3 in η, and degree 4 in τ. Plugging in specific values quickly shows that the general formula for the eigenvalues does not have a simple closed form – other than as the roots of the characteristic polynomial – as the symbol of the coarse grid correction above.

However, both the upper bound computed in the two step analysis as well as the sharp bound in an all-at-once approach for the one-dimensional case were obtained by considering limiting cases at the boundary of the parameter domain [27]. Hence, as heuristics, we propose to consider limiting cases of the symbol TG, compute the eigenvalues there using a computer algebra system, and determine their supremum using CAD. This procedure is backed up by the rigorous analysis carried out in the other cases.

First, if we consider the corner $(c_1,c_2)=(1,1)$, the symbol degenerates to the singular diagonal matrix $\operatorname{diag}(0,0,\xi_0,\xi_0,\xi_0,\xi_0,\xi_1,\xi_1)$, where

$$\begin{aligned}\xi_0&=\frac{\left(9\eta\tau^2-24\eta\tau+16\eta+1296\tau^2-1728\tau+576\right)^2}{256(\eta+36)^2},\\\xi_1&=\frac{\left(\eta\tau^2-8\eta\tau+16\eta+576\tau^2-1152\tau+576\right)^2}{256(\eta+36)^2},\end{aligned}$$

are the nonzero eigenvalues of the matrix. The upper bound for these eigenvalues can be determined easily by considering the limits $\eta\to0,\infty$.

Secondly, we consider the values $(c_2,\eta)=(1,0)$, for which the symbol is a singular diagonal matrix with the eigenvalues

$$\begin{aligned}&0,\,\tfrac{1}{256}\left(c_1\tau-5\tau+4\right)^4,\,\tfrac{1}{256}\left(c_1\tau+5\tau-4\right)^4,\\&\tfrac{1}{256}\left(9c_1^2\tau^2+9\tau^2-24\tau+16\right)\left(9c_1^4\tau^2+54c_1^2\tau^2-72c_1^2\tau+16c_1^2+9\tau^2-24\tau+16\right).\end{aligned}$$

The supremum of these eigenvalues for all $c_1\in[0,1]$ can then be computed exactly using CAD. Restricting the matrix to values of the remaining parts of the parametric domain does not yield any further information. So it turns out that the spectral radius can be bounded by taking the supremum of these two special cases. This way we obtain the following guess for the supremum of the norm:

$$q_{TG}(\tau) = \max\left\{\frac{1}{16}(\tau-4)^2, \frac{1}{4}|3\tau-2|\sqrt{9\tau^2-12\tau+8}\right\}.$$

This guess was, as in the rigorous cases, obtained rather quickly. Compared to the large entries in the symbol, the final bound is again rather simple, but not quite of the form that would easily be used as an ansatz for numerical interpolation. Furthermore, since the symbolic bound was found at a point where a singularity occurs, these values cannot be recovered by brute force interpolation. This also becomes apparent when comparing the guessed bound to values obtained using a gridding approach for τ close to zero or τ much larger than one, see Figure 3. We propose to use this heuristic approach also for other PDEs in order to obtain a more precise insight on the dependence of the parameters.

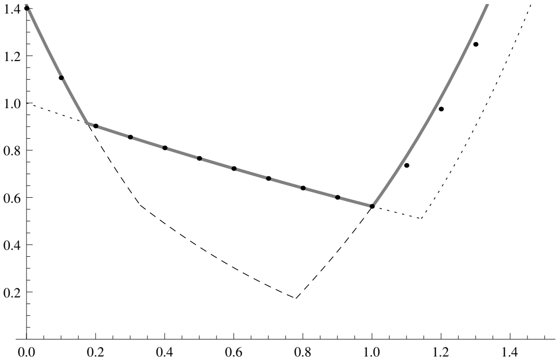

Fig. 3. Guessed bound (gray line) vs brute force interpolation Maxima (black points)

References

1. Andrews, G.E., Askey, R., Roy, R.: Special Functions. Encyclopedia of Mathematics and its Applications 71. Cambridge University Press, 2000.
2. Becirovic, A., Paule, P., Pillwein, V., Riese, A., Schneider, C., Schöberl, J.: Hypergeometric Summation Algorithms for High Order Finite Elements. Computing, 78(3):235–249, 2006.
3. Beuchler, S., Pillwein, V.: Sparse shape functions for tetrahedral p-FEM using integrated Jacobi polynomials. Computing, 80(4):345–375, 2007.
4. Beuchler, S., Pillwein, V., Schöberl, J., Zaglmayr, S.: Sparsity optimized high order finite element functions on simplices. In Ulrich Langer and Peter Paule, editors, Numerical and Symbolic Scientific Computing: Progress and Prospects, Texts and Monographs in Symbolic Computation, pages 21–44. Springer, Wien, 2012.
5. Beuchler, S., Pillwein, V., Zaglmay, S.: Fast summation techniques for sparse shape functions in tetrahedral hp-FEM. In Domain Decomposition Methods in Science and Engineering XX, pages 537–544, 2012.

6. Beuchler, S., Pillwein, V., Zaglmayr, S.: Sparsity optimized high order finite element functions for H(div) on simplices. Numerische Mathematik, 122(2):197–225, 2012.
7. Beuchler, S., Pillwein, V., Zaglmayr, S.: Sparsity optimized high order finite element functions for H(curl) on tetrahedra. Advances in Applied Mathematics, 50:749–769, 2013.
8. Beuchler, S., Schöberl, J.: New shape functions for triangular p-fem using integrated Jacobi polynomials. Numer. Math., 103:339–366, 2006.
9. Braess, D.: Finite Elements: Theory, Fast Solvers and Applications in Solid Mechanics. Cambridge University Press, Cambridge, 2007.
10. Brandt, A.: Multi-level adaptive solutions to boundary-value problems. Math. Comp., 31:333 – 390, 1977.
11. Brandt, A.: Rigorous Quantitative Analysis of Multigrid, I: Constant Coefficients Two-Level Cycle with L_2-Norm. SIAM J. on Numerical Analysis, 31(6):1695 – 1730, 1994.
12. Brenner, S.C., Scott, L.R.: The Mathematical Theory of Finite Element Methods, volume 15 of Texts in Applied Mathematics. Springer New York, 2nd edition, 2002.
13. Ciarlet, P.G.: The Finite Element Method for Elliptic Problems. North Holland, Amsterdam, 1978.
14. Cluzeau, T., Dolean, V., Nataf, F., Quadrat, A.: Symbolic preconditioning techniques for linear systems of partial differential equations. accepted for publ. in Proceedings of 20th International Conference on Domain Decomposition Methods, 2012.
15. Cluzeau, T., Dolean, V., Nataf, F., Quadrat, A.: Symbolic methods for developing new domain decomposition algorithms. Rapport de recherche RR-7953, INRIA, May 2012.
16. Demkowicz, L.: Computing with *hp* Finite Elements I. One- and Two-Dimensional Elliptic and Maxwell Problems. CRC Press, Taylor and Francis, 2006.
17. Hackbusch, W.: Multi-Grid Methods and Applications. Springer, Berlin, 1985.
18. Hong, H., Liska, R., Steinberg, S.: H. Hong, R. Liska, and S. Steinberg. Testing stability by quantifier elimination. J. Symbolic Comput., 24(2):161–187, 1997. Applications of quantifier elimination (Albuquerque, NM, 1995).
19. Karniadakis, G.E., Sherwin, S.J.: Spectral/hp Element Methods for CFD. Numerical Mathematics and Scientific Computation. Oxford University Press, New York, Oxford, 1999.
20. Kauers, M.: SumCracker – A Package for Manipulating Symbolic Sums and Related Objects. J. Symbolic Comput., 41(9):1039–1057, 2006.
21. Kauers, M.: Guessing Handbook. RISC Report Series 09-07, Research Institute for Symbolic Computation (RISC), Johannes Kepler University Linz, Schloss Hagenberg, 4232 Hagenberg, Austria, 2009.
22. Koutschan, C.: HolonomicFunctions (User's Guide). Technical Report 10-01, RISC Report Series, University of Linz, Austria, January 2010.
23. Koutschan, C., Lehrenfeld, C., Schöberl, J.: Computer Algebra meets Finite Elements: an Efficient Implementation for Maxwell's Equations. In Numerical and Symbolic Scientific Computing: Progress and Prospects, volume 1 of Texts and Monographs in Symbolic Computation, pages 105–121. Springer, Wien, 2012.
24. Langer, U., Reitzinger, S., Schicho, J.: Symbolic methods for the element preconditioning technique. In U. Langer and F. Winkler, editors, Proc. SNSC Hagenberg 2001, pages 293–308. Springer, 2003.
25. Levandovskyy, V., Martin, B.: A Symbolic Approach to Generation and Analysis of Finite Difference Schemes of Partial Differential Equations. In Ulrich Langer and Peter Paule, editors, Numerical and Symbolic Scientific Computing: Progress and

Prospects, Texts & Monographs in Symbolic Computation, pages 123–156. Springer, Wien, 2011.

26. Pillwein, V., Takacs, S.: Smoothing analysis of an all-at-once multigrid approach for optimal control problems using symbolic computation. In U. Langer and P. Paule, editors,Numerical and Symbolic Scientific Computing: Progress and Prospects. Springer, Wien, 2011.
27. Pillwein, V., Takacs, S.: A local Fourier convergence analysis of a multigrid method using symbolic computation. J. Symbolic Comput., 63:1–20, 2014.
28. Schwab, Ch.: *p*- and *hp*-Finite Element Methods: Theory and Applications in Solid and Fluid Mechanics. Numerical Mathematics and Scientific Computation. Oxford University Press, Oxford, 1998.
29. Szabó, B.A., Babuška, I.: Finite Element Analysis. John Wiley & Sons, 1991.