Test Case One

Scalar Laplace Equation

$$-\Delta\phi = f$$

TU Delft Faculty of Aerospace Engineering
Department of Aerodynamics

Supervisor

Prof. Marc Gerritsma
MSc. Yi Zhang

February 5, 2018

Contents

1 Test Case Setup

This part will set up a mimetic spectral element solver for a scalar Laplace equation. The theoretical description will not be elaborated here but the implementation detail will be recorded to serve the major purpose of this report.

1.1 Scalar Laplace Equation (Poisson Equation)

The Scalar Laplace equation is given in 1.1:

$$-\Delta\phi(x,y) = f, \quad (x,y) \in \Omega \tag{1.1a}$$

$$BC : \phi(x,y) = \phi_{tr}, \quad (x,y) \in \partial\Omega. \tag{1.1b}$$

To apply mimetic spectral element method to discretise this equation, equation is re-written in the following system of partial differential equations. The new system has equation 1.2a and 1.2c, two purely topological equations, and one extra constitutive equation 1.2b that connects an 1-form to an $(n-1)$-form.

$$\boldsymbol{d}\phi^{(0)} = \boldsymbol{u}^{(1)}, \tag{1.2a}$$

$$\star\boldsymbol{u}^{(1)} = \tilde{\boldsymbol{q}}^{(n-1)}, \tag{1.2b}$$

$$\boldsymbol{d}\tilde{\boldsymbol{q}}^{(n-1)} = f^{(n)}. \tag{1.2c}$$

The equation system can be deliberately used for discretisation.

1.2 Discretisation of Poisson Equation

This part will discretise the equation system 1.2 one by one, and then construct the discretised linear equation system.

Before writing all the equations out with variables, we place a Tonti Graph at the top first. Then referring to the graph, we can make certain rules for the notions of basis and variables. A Tonti diagram is given in figure 1. We let variables with a hat $\hat{}$ denote their own basis function, and if we remove the notions' subscription in the Tonti diagram, they will represent the degrees of freedom on cochain.

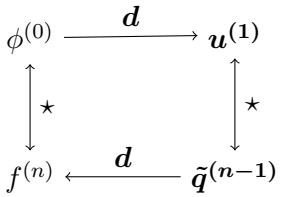

Figure 1: Tonti Diagram

1.2.1 Gradient Equation: $\boldsymbol{d\phi^{(0)} = u^{(1)}}$

On the left-hand-side of the gradient equation, $\phi^{(0)}$ is a scalar function defined on calculation domain Ω. It admits a projection Π_h that projects itself to a finite dimensional space $Q = span\{h_i(x) \otimes h_j(y)\},\ i, j = 0, 1, 2, ..., n$:

$$\Pi_h(\phi^{(0)}) = \phi_{(i,j)} \cdot \left(h_i^{(0)}(x) \otimes h_j^{(0)}(y)\right) = \phi_{k(i,j)} \cdot \hat{\phi}_{k(i,j)}^{(0)} \ . \tag{1.3}$$

After projection, function ϕ is expanded by the nodal basis functions $\hat{\phi}_k^{(0)}$, where $k(i, j)$ is a compression mapping that maps a 2D location array into 1D. This compression mapping affects the shape and sparsity pattern of the coefficient matrix of the final equation system.

Similarly, on the right-hand-side, the vector $\boldsymbol{u^{(1)}}$ admits another expansion given by:

$$\Pi_h(\boldsymbol{u^{(1)}}) = u_{(i,j)} \cdot \left(\boldsymbol{e_i^{(1)}}(x) \otimes h_j^{(0)}(y)\right) + v_{(j,i)} \cdot \left(h_i^{(0)}(x) \otimes \boldsymbol{e_j^{(1)}}(y)\right) \tag{1.4}$$

$$= u_{K(i,j)} \cdot \hat{u}_{K(i,j)}^{(1)} \ , \tag{1.5}$$

where $K(i, j)$ is the compression mapping that controls the sequence of the degrees of freedom. According to the construction of the edge function, $d(\alpha^{(0)}) = \sum_{i=0}^{N}(\alpha_i \cdot h_i(x)) = \sum_{i=1}^{N}(\alpha_i - \alpha_{i-1}) \cdot \boldsymbol{e_i^{(1)}}$. Then the gradient equation will be:

$$\left[\boldsymbol{E_{K\times k}^{(1,0)}}\right] \cdot \phi_{k(i,j)} = \boldsymbol{u}_{K(i,j)} \ . \tag{1.6}$$

1.2.2 Divergence Equation: $\boldsymbol{d\tilde{q}^{(n-1)} = -f^{(n)}}$

Similar to the previous gradient equation, we write the discretised system directly:

$$\left[\boldsymbol{\tilde{E}_{k\times K}^{(n,n-1)}}\right] \cdot \tilde{\boldsymbol{q}}_{K(i,j)} = f_{k(i,j)} \ . \tag{1.7}$$

In this relation, both k and K will take the same compression numbering scheme as the previous gradient equation.

1.2.3 Constitutive Relation: $\boldsymbol{\star u^{(1)} = \tilde{q}^{(n-1)}}$

We use the $\boldsymbol{L^2}$ inner product of flux vector $\boldsymbol{\tilde{q}^{(n-1)}}$ and its basis function sets to measure the local metrics:

$$\left\langle \boldsymbol{\tilde{q}^{(n-1)}},\ \boldsymbol{\hat{q}^{(n-1)}} \right\rangle_{\boldsymbol{L^2},\Omega} = \int_\Omega \left(\star \boldsymbol{\tilde{q}^{(n-1)}} \wedge \boldsymbol{\hat{q}^{(n-1)}}\right) \tag{1.8a}$$

$$= \int_\Omega \star\star \left(\boldsymbol{u^{(1)}} \wedge \boldsymbol{\hat{q}^{(n-1)}}\right) \tag{1.8b}$$

$$= (-1)^{n(n-k)} \cdot \int_\Omega \left(\boldsymbol{u^{(1)}} \wedge \boldsymbol{\hat{q}^{(n-1)}}\right) \tag{1.8c}$$

$$= \int_\Omega \left(\boldsymbol{d}\phi^{(0)} \wedge \boldsymbol{\hat{q}^{(n-1)}}\right) \tag{1.8d}$$

$$= \int_\Omega \boldsymbol{d}\left(\phi^{(0)} \cdot \boldsymbol{\hat{q}^{(n-1)}}\right) - \int_\Omega \phi^{(0)} \boldsymbol{d\hat{q}^{(n-1)}}. \tag{1.8e}$$

The left-hand-side gives a discrete system:

$$\left\langle \tilde{\boldsymbol{q}}^{(n-1)},\ \hat{\boldsymbol{q}}^{(n-1)} \right\rangle_{\boldsymbol{L^2},\Omega} = \left[\boldsymbol{M}^{(n-1)}_{\boldsymbol{K\times K}}\right] \cdot \boldsymbol{q_{K(i,j)}}, \tag{1.9}$$

and the right-hand-side will be:

$$\int_\Omega \boldsymbol{d}\left(\phi^{(0)} \cdot \hat{\boldsymbol{q}}^{(n-1)}\right) - \int_\Omega \phi^{(0)} \boldsymbol{d}\hat{\boldsymbol{q}}^{(n-1)} = \left[\boldsymbol{B}^{(n-1,0)}\right] \cdot \phi_{tr} - \left[\boldsymbol{E}^{(n,n-1)^T}_{\boldsymbol{k\times K}} \boldsymbol{W}^{(n,0)}_{\boldsymbol{k\times k}}\right] \cdot \phi_{k(i,j)}. \tag{1.10}$$

Thus the discretised system will be:

$$\left[\boldsymbol{M}^{(n-1)}_{\boldsymbol{K\times K}}\right] \cdot \boldsymbol{q_{K(i,j)}} + \left[\boldsymbol{E}^{(n,n-1)^T}_{\boldsymbol{k\times K}} \boldsymbol{W}^{(n,0)}_{\boldsymbol{k\times k}}\right] \cdot \phi_{k(i,j)} = \left[\boldsymbol{B}^{(n-1,0)}\right] \cdot \phi_{tr} \tag{1.11}$$

1.2.4 Final Discrete Equation System

The final discretised linear equation system of a single spectral element is:

$$\begin{bmatrix} \boldsymbol{M}^{(n-1)}_{\boldsymbol{K\times K}} & \boldsymbol{E}^{(n,n-1)^T}_{\boldsymbol{k\times K}} \boldsymbol{W}^{(n,0)}_{\boldsymbol{k\times k}} \\ \boldsymbol{W}^{(n,0)^T}_{\boldsymbol{k\times k}} \boldsymbol{E}^{(n,n-1)}_{\boldsymbol{k\times K}} & \boldsymbol{O} \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{q}} \\ \phi \end{bmatrix} = \begin{bmatrix} \boldsymbol{B}^{(n-1,0)} \phi_{tr} \\ \boldsymbol{W}^{(n,0)^T}_{\boldsymbol{k\times k}} f_{k(i,j)} \end{bmatrix}. \tag{1.12}$$

This is the starting point of the first preconditioning test case.

2 Analysis of the Final System

To analyse the condition of a saddle point linear equation system, we establish a common routine for such tasks. First of all, one will have to inspect the condition number of the mass matrix $\boldsymbol{M}$, incidence matrices $\boldsymbol{E}$, Schur complement of the mass matrix in the discrete system $\boldsymbol{S}$, and the wedge product matrix $\boldsymbol{W}$. In particular, one should find how the condition numbers scale with the problem size. The first step is done and the results are shown in table 1 and figure 2 below. In results, $\boldsymbol{LHS}$ is the full system's coefficient matrix, and $\boldsymbol{map}$ is defined by $\boldsymbol{map} = \boldsymbol{E}^{(n,n-1)}_{\boldsymbol{k\times K}} \cdot \boldsymbol{E}^{(n,n-1)^T}_{\boldsymbol{k\times K}}$.

Table 1: Condition Number of Matrices

p-order	5	9	13	25
$\boldsymbol{M^{(n-1)}}$	33.35	88.39	170.53	578.33
$\boldsymbol{S}$	29.90	106.39	248.40	1301.06
$\boldsymbol{LHS}$	22.60	24.20	48.82	251.93
$\boldsymbol{map}$	13.93	39.86	78.77	273.31
$\boldsymbol{W^{(0,n)}}$	1.82	2.06	2.17	2.30

The results show the same situation recorded in other similar literatures. The Schur complement matrix $\boldsymbol{S}$ has the worst condition, and $\kappa(\boldsymbol{M})$ also grows fast with the problem size. This is a "signature" symptom of many traditional PDE induced saddle point systems, where methods like Uzawa algorithm and Bramble-Pasciak CG become costly.

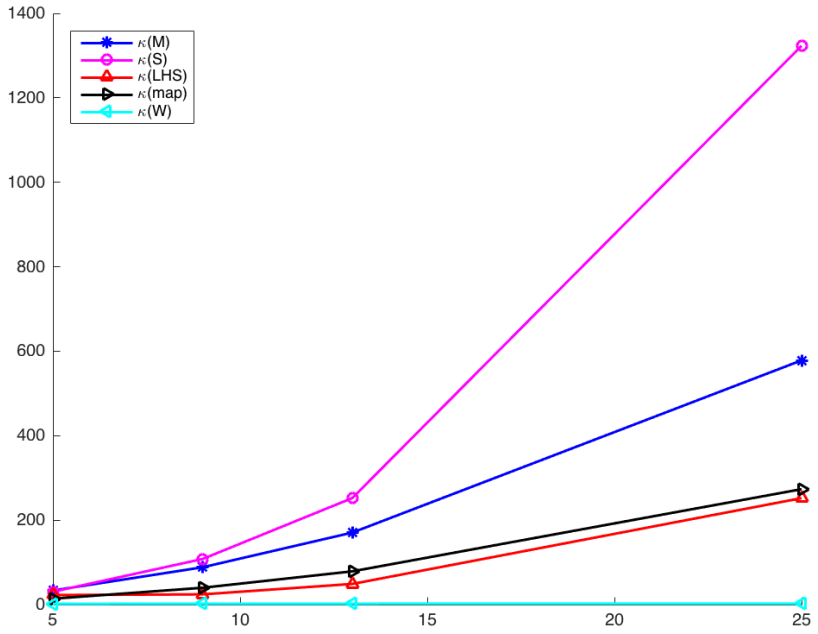

Figure 2: Condition Number of the System

In the result, we could see that the wedge product matrix do not vary with the problem size, an intuitive explanation is that the wedge product in this case is a metric-free operation, thus unlike the mass matrix, it will not be affected by the mesh density. Another interesting observation is that the condition number of the full system and the condition number of the so called $\boldsymbol{map}$ matrix change synchronously. From the minimisation problem perspective, we are solving a scalar Laplace equation and $\boldsymbol{q}$ is an intermediate function that minimises the energy, thus the condition of the full system is determined by the Laplacian part. To further illustrate this idea, it's found that the condition of a properly scaled block diagonal matrix $\mathcal{A}$ in equation 2.1 is bounded only by the $(2,2)$ block, which means that the "Laplacian" accounts for worsening the condition of the left-hand-side matrix.

$$\kappa(\mathcal{A}) = \min_{\lambda>0} \left\{ \kappa \begin{pmatrix} \boldsymbol{\lambda M} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{W^T E M^{-1} E^T W} \end{pmatrix} \right\} = \kappa \left(\boldsymbol{W^T E M^{-1} E^T W} \right) \tag{2.1}$$

This observation rings a bell of why constraint preconditioners are applied. About this phenomena, I give a short explanation of why I choose to inspect such a $\boldsymbol{map}$ matrix. This $\boldsymbol{map}$ matrix is exactly the finite difference 5-point Laplace operator. In the mimetic discretisation framework, this is equivalent to a finite difference discrete scalar Laplacian that applies on a uniform mesh. The coming work would have to work with the $\boldsymbol{map}$ matrix instead of the incidence matrix to reduce the Schur complement condition number.

The first inspection tells us about the possibilities of the coming work. We may try the traditional block diagonal preconditioning techniques first. After that, we could also try constraint preconditioners. To achieve these goals, we first need a powerful mass matrix approximation. In addition, traditional approaches also requires an algorithm to calculate preconditioner for the Schur complement. The coming subsections will prepare these basic

elements first.

2.1 Mass Matrix Approximation

The mass matrix of a $(n-1)$-form is obtained by calculating the inner product of the set of $(n-1)$-form basis functions. Each mass matrix element $m_{(K_1,K_2)}$ is calculated by:

$$m_{(K_1,K_2)} = \left\langle \hat{\boldsymbol{q}}^{\boldsymbol{n-1}}_{\boldsymbol{K_1(p,q)}} \ , \ \hat{\boldsymbol{q}}^{\boldsymbol{n-1}}_{\boldsymbol{K_2(m,n)}} \right\rangle_{\boldsymbol{L^2,\ \Omega}} . \tag{2.2}$$

Since $\hat{\boldsymbol{q}}^{\boldsymbol{n-1}}_{\boldsymbol{K}}$ is a compressed array with both $\boldsymbol{u}$ and $\boldsymbol{v}$ basis, if they are stored in the 1D array as two separated subarrays like $[\boldsymbol{u_{basis}}..., \boldsymbol{v_{basis}}...]$, the mass matrix will be a 2-by-2 block diagonal matrix. Further more, since the $\boldsymbol{u_{basis}}$ and $\boldsymbol{v_{basis}}$ inherited the symmetry from the co-ordinate system in a quadrilateral mesh, the mass matrix will have a repeating diagonal block if the compression mapping K is "symmetrical" applied on both vector basis. This idea is shown in figure 3.

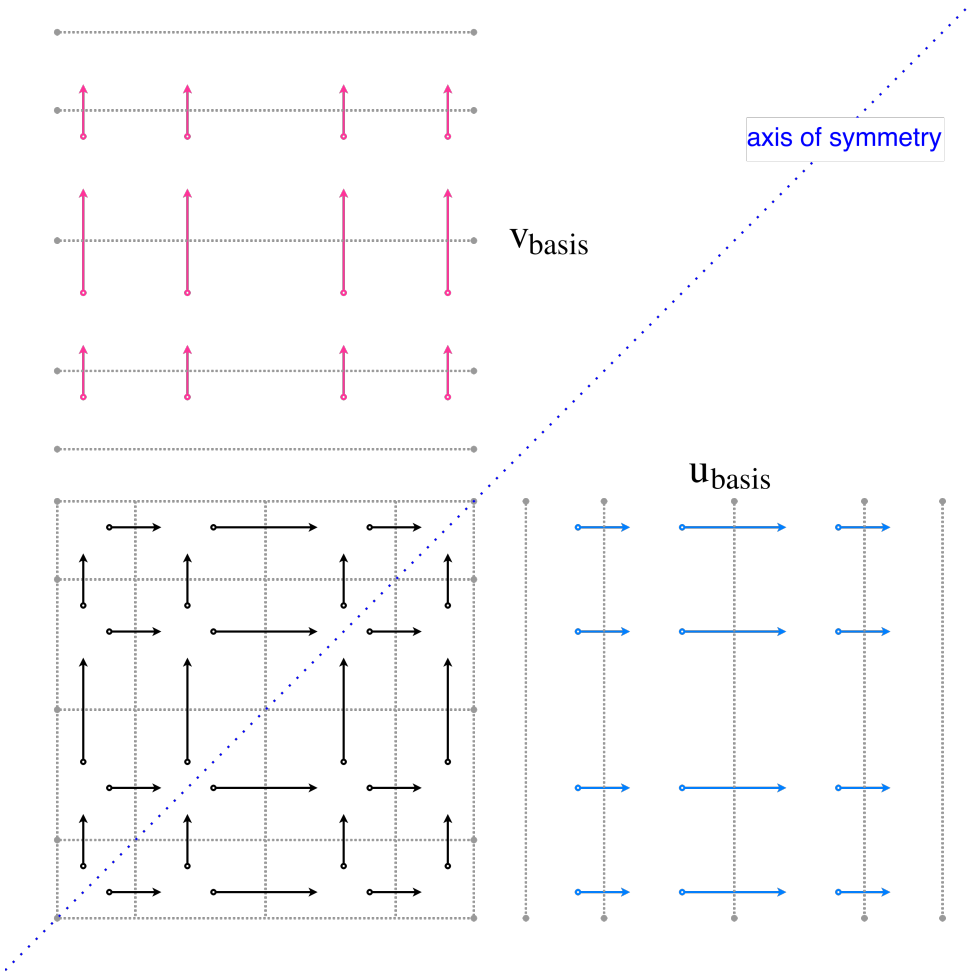

Figure 3: Symmetry of Vectors

Based on the previous analysis, I computed a mass matrix with polynomial expansion degree $p = 5$, using a standard GLL mesh. The mass matrix surface plot and sparsity

pattern are shown in figure 9 and 10.

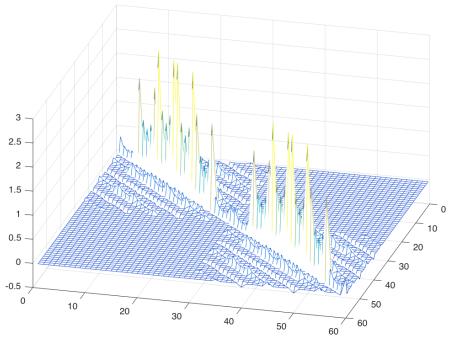

Figure 4: $\boldsymbol{M^{(n-1)}}$ with $p = 5$

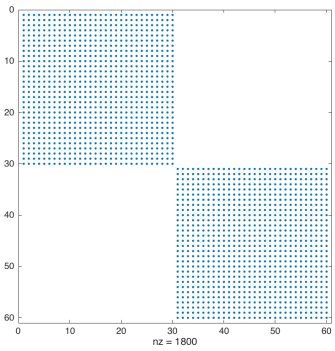

Figure 5: Sparsity Pattern of $\boldsymbol{M^{(n-1)}}$ with $p = 5$

From the mass matrix surface plot above, we further find out that the mass matrix is a diagonally dominant matrix. This is a good news for finding preconditioners for the mass matrix. Later on, a few ideas will be tried to obtain a cheap fast preconditioner.

2.1.1 Jacobi Preconditioning

Technically speaking, Jacobi preconditioning also known as diagonal scaling, is originally a preconditioning technique used for Krylov subspace iterative solvers. Nowadays being widely used as a smoother for multigrid solvers etc.. In this case, since the mass matrix is indeed a diagonally dominant matrix, Jacobi preconditioning seems to be a simple and good preconditioner.

The following figure 6 plots the condition number change before and after Jacobi preconditioning.

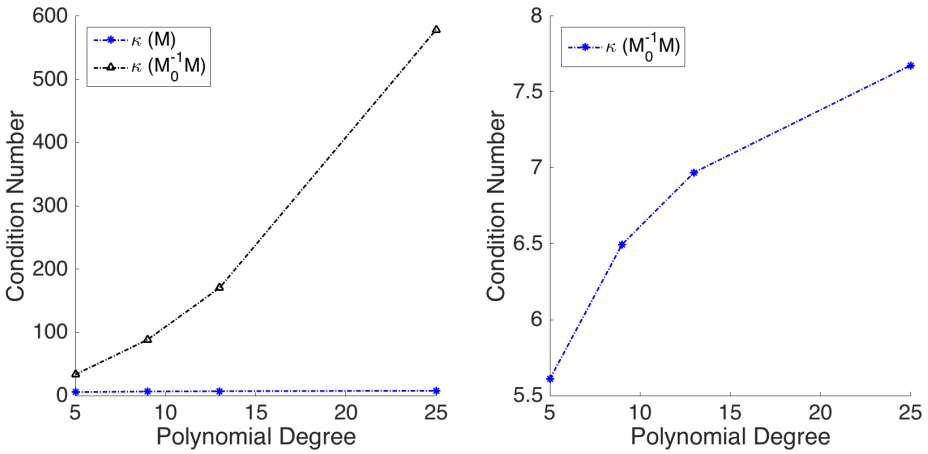

Figure 6: Condition Number of Mass Matrix Before and After Diagonal Scaling

We observed that the diagonal scaling suppressed the pitching-up trend of the mass matrix condition number, and significantly reduced the absolute value of the condition number. This tells us that the diagonal dominance of the mass matrix is increasing. This is an important information that leads to further improvements in making a good preconditioner.

If we inspect the reason of the diagonal dominance, we can see that both mass matrices of 1D basis are increasingly diagonally dominant as the expansion order increases. Because the higher the order of expansion basis goes, the closer they are to be exactly orthogonal. The next preconditioner will make use of the orthogonality of the basis.

2.1.2 Orthogonal Basis Preconditioner

In this section, we will make use of the unique structure of the mass matrix to construct a new preconditioner. If we take an element $m_{(K_1,K_2)}$ from the $(2,2)$ block of the mass matrix $\boldsymbol{M_v^{(n-1)}}$, we can make the following decomposition:

$$m_{(K_1,K_2)} = \iint_\Omega h_p(x)e_q(y) \cdot h_m(x)e_n(y)dxdy \tag{2.3a}$$

$$= \int_{x_0}^{x_{N+1}} h_p(x)h_m(x)dx \cdot \int_{y_0}^{y_{N+1}} e_q(y)e_n(y)dy \tag{2.3b}$$

$$= \lambda_{(p,m)} \cdot \mu_{(q,n)}, \tag{2.3c}$$

where $\lambda_{(p,m)}$ and $\mu_{(q,n)}$ are elements from mass matrix $\boldsymbol{M_\lambda^{(0)}}$ produced by 1D nodal basis and mass matrix $\boldsymbol{M_\mu^{(1)}}$ produced by 1D edge basis functions. Thus the mass matrix of $\boldsymbol{\hat{q}^{(n-1)}}$ admits this tensor product factorisation:

$$\boldsymbol{M_v^{(n-1)}} = \boldsymbol{M_\lambda^{(0)}} \otimes \boldsymbol{M_\mu^{(1)}}. \tag{2.4}$$

The surface plot below will demonstrate this tensor product in a graphical sense. More importantly, the mass matrix eigenvalue spectra would follow the same tensor product structure! This helps us design and analyse preconditioners. We will have a little discussion on this matrix tensor product properties in the next small section.

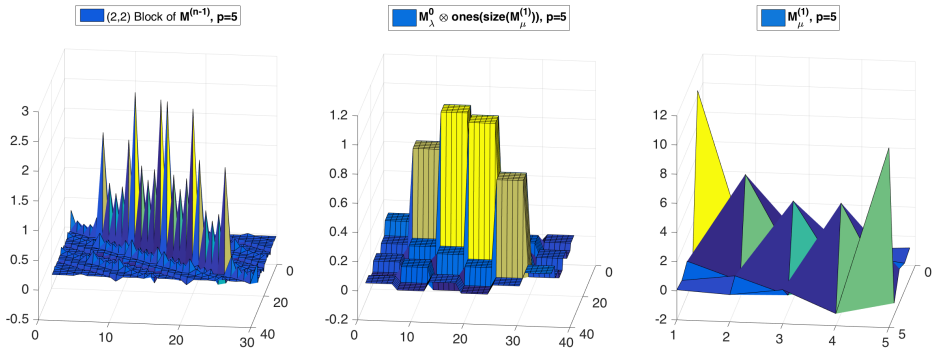

Figure 7: Tensor Product Decomposition of Mass Matrix

It is obvious from the surface plot above that the left mass matrix is composed by the middle one for the global shape, and the right one for the local shape scaling. According to the tensor product decomposition, we can see how the numbering scheme (compression map) affects the final shape of the mass matrix.

Now, we use a trick to produce a preconditioner. While evaluating the integrations in mass matrix $\boldsymbol{M}_{\boldsymbol{\lambda}}^{(\mathbf{0})}$, we use Gauss quadrature with nodal basis functions. If we specify the quadrature basis function set as exactly the same set of reduction basis functions, we would be able to obtain an orthogonal co-chain expansion vector space. In other words, $\int_{x_0}^{x_{N+1}} h_p(x)h_m(x)dx = \lambda_{(p,m)} \cdot \delta_{pm}$, δ is the Kronecker delta. This trick made $\boldsymbol{M}_{\boldsymbol{\lambda}}^{(\mathbf{0})}$ a diagonal matrix if the basis function follows the numbering scheme as in figure 8. Thus the final mass matrix $\boldsymbol{M}_{\mathbf{0}}^{(\boldsymbol{n-1})}$ will have a very fine block diagonal structure.

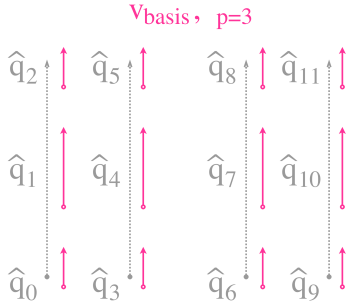

Figure 8: Numbering Scheme of Basis Functions

The block diagonal mass matrix will be shown below:

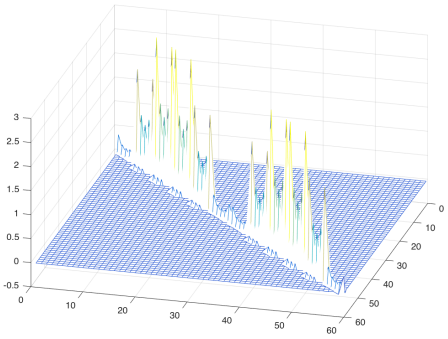

Figure 9: $\boldsymbol{M}_{\mathbf{0}}^{(\boldsymbol{n-1})}$ with $p = 5$

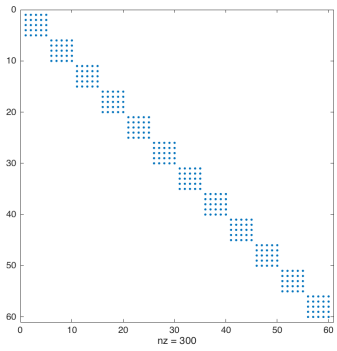

Figure 10: Sparsity Pattern of $\boldsymbol{M}_{\mathbf{0}}^{(\boldsymbol{n-1})}$ with $p = 5$

Within each block, the shape would be exactly the same as the mass matrix $\boldsymbol{M}_{\boldsymbol{\mu}}^{(\mathbf{1})}$. The new preconditioner is easily and fast invertible, and also easy in storage and manipulation, because $\boldsymbol{M}_{\boldsymbol{v}}^{(\boldsymbol{n-1})^{-1}} = \boldsymbol{M}_{\boldsymbol{\lambda}}^{(\mathbf{0})^{-1}} \otimes \boldsymbol{M}_{\boldsymbol{\mu}}^{(\mathbf{1})^{-1}}$. In this case $\boldsymbol{M}_{\boldsymbol{\lambda}}^{(\mathbf{0})^{-1}}$ is a diagonal matrix, thus the work load of taking the inverse only scales with the mass matrix size $\boldsymbol{M}_{\boldsymbol{\mu}}^{(\mathbf{1})^{-1}}$.

Now, we will show the performance of this preconditioner in figure 11.

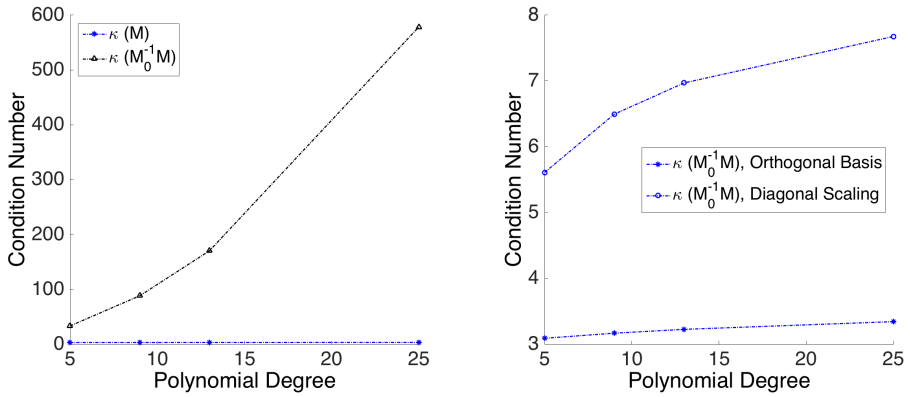

Figure 11: Performance of Orthogonal Basis Preconditioner

From the figure above, we surely see that the performance of the Orthogonal Basis preconditioner is much more stable and produce a preconditioned mass matrix whose conditioner number is nearly independent from the problem size.

2.2 Discussion on Matrix Kronecker Product

This is an additional discussion on the properties of the matrix Kronecker product. They could reveal the block structure of matrices and could be very helpful in analysing the spectral property, sparsity pattern, and matrix ordering (numbering), etc., especially for PDE induced linear equation systems.

We introduce two matrices: $\boldsymbol{A_{pm\times qn}} = \boldsymbol{A_{1p\times q}} \otimes \boldsymbol{A_{2m\times n}}$ and $\boldsymbol{B_{qn\times ro}} = \boldsymbol{B_{1q\times r}} \otimes \boldsymbol{B_{2n\times o}}$. We will use them in the coming discussion.

2.2.1 Permutation of Kronecker Product

Tensor product is not permutable. But there exists a pair of unique permutation matrices $(\boldsymbol{P_{pm\times pm}}, \boldsymbol{Q_{qn\times qn}})$ that helps the permutation:

$$\boldsymbol{P_{pm\times pm}} \left(\boldsymbol{A_{1p\times q}} \otimes \boldsymbol{A_{2m\times n}}\right) \boldsymbol{Q_{qn\times qn}} = \boldsymbol{A_{2m\times n}} \otimes \boldsymbol{A_{1p\times q}}. \tag{2.5}$$

Moreover, when $p = q$ and $m = n$, $\boldsymbol{P} = \boldsymbol{Q^T}$.

Using this property, we understand how the numbering scheme change the ordering of the mass matrix:

$$\boldsymbol{P}\left(\boldsymbol{M_\lambda^{(0)}} \otimes \boldsymbol{M_\mu^{(1)}}\right)\boldsymbol{P^T} = \boldsymbol{M_\mu^{(1)}} \otimes \boldsymbol{M_\lambda^{(0)}}. \tag{2.6}$$

This tells us that the sparsity pattern of mass matrix is affected by the numbering scheme while the spectrum is not.

2.2.2 Mixed Product Property

The Kronecker product could be mixed with matrix product:

$$\boldsymbol{A_{pm\times qn}}\boldsymbol{B_{qn\times ro}} = \left(\boldsymbol{A_{1p\times q}}\boldsymbol{B_{1q\times r}}\right) \otimes \left(\boldsymbol{A_{2m\times n}}\boldsymbol{B_{2n\times o}}\right). \tag{2.7}$$

Thus, if the mass matrices admits diagonalisations:

$$\boldsymbol{M}_{\boldsymbol{v}}^{(\boldsymbol{n-1})} = \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^{\boldsymbol{T}}, \tag{2.8a}$$

$$\boldsymbol{M}_{\boldsymbol{\lambda}}^{(\boldsymbol{0})} = \boldsymbol{T}_{\boldsymbol{\lambda}}\boldsymbol{\Lambda}_{\boldsymbol{\lambda}}\boldsymbol{T}_{\boldsymbol{\lambda}}^{\boldsymbol{T}}, \tag{2.8b}$$

$$\boldsymbol{M}_{\boldsymbol{\mu}}^{(\boldsymbol{1})} = \boldsymbol{T}_{\boldsymbol{\mu}}\boldsymbol{\Lambda}_{\boldsymbol{\mu}}\boldsymbol{T}_{\boldsymbol{\mu}}^{\boldsymbol{T}}, \tag{2.8c}$$

then we have:

$$\begin{aligned} \boldsymbol{M}_{\boldsymbol{v}}^{(\boldsymbol{n-1})} &= \boldsymbol{M}_{\boldsymbol{\lambda}}^{(\boldsymbol{0})} \otimes \boldsymbol{M}_{\boldsymbol{\mu}}^{(\boldsymbol{1})} \\ &= \left(\boldsymbol{T}_{\boldsymbol{\lambda}} \otimes \boldsymbol{T}_{\boldsymbol{\mu}}\right)\left(\boldsymbol{\Lambda}_{\boldsymbol{\lambda}} \otimes \boldsymbol{\Lambda}_{\boldsymbol{\mu}}\right)\left(\boldsymbol{T}_{\boldsymbol{\lambda}}^{\boldsymbol{T}} \otimes \boldsymbol{T}_{\boldsymbol{\mu}}^{\boldsymbol{T}}\right) \\ &= \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^{\boldsymbol{T}}. \end{aligned} \tag{2.9}$$

This tells us that the spectrum of the big mass matrix is spanned by small mass matrices. What's more, many symmetric iterative solvers require symmetric preconditioners which need to compute the Cholesky factorisation of the mass matrix. With this Kronecker product, the calculation load no longer scales with the 2D problem size, but with two 1D problem size. For direct solvers, the preconditioner is the inverse of mass matrix, which could also be calculated by inverting the small mass matrices.

About this matrix tensor product, we give one more example of how it helps us understand the block structure of the coefficient matrices. Take the Schur complement of the mass matrix:

$$\boldsymbol{S} = \boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{E}\boldsymbol{M}^{-\boldsymbol{1}}\boldsymbol{E}^{\boldsymbol{T}}\boldsymbol{W} = \boldsymbol{W}^{\boldsymbol{T}}\boldsymbol{G}\boldsymbol{W}. \tag{2.10}$$

The incidence matrix has a block structure:

$$\boldsymbol{E} = \begin{bmatrix} \boldsymbol{D}_{\boldsymbol{u}} & \boldsymbol{D}_{\boldsymbol{v}} \end{bmatrix}, \tag{2.11}$$

corresponding to the block diagonal structure of the mass matrix:

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{M}_{\boldsymbol{u}} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{M}_{\boldsymbol{v}} \end{bmatrix}. \tag{2.12}$$

Thus we have:

$$\boldsymbol{G} = \boldsymbol{E}\boldsymbol{M}^{-\boldsymbol{1}}\boldsymbol{E}^{\boldsymbol{T}} = \boldsymbol{D}_{\boldsymbol{u}}{\boldsymbol{M}_{\boldsymbol{u}}}^{-\boldsymbol{1}}{\boldsymbol{D}_{\boldsymbol{u}}}^{\boldsymbol{T}} + \boldsymbol{D}_{\boldsymbol{v}}{\boldsymbol{M}_{\boldsymbol{v}}}^{-\boldsymbol{1}}{\boldsymbol{D}_{\boldsymbol{v}}}^{\boldsymbol{T}}. \tag{2.13}$$

Further more, we have:

$$\begin{aligned} \boldsymbol{D}_{\boldsymbol{v}}{\boldsymbol{M}_{\boldsymbol{v}}}^{-\boldsymbol{1}}{\boldsymbol{D}_{\boldsymbol{v}}}^{\boldsymbol{T}} &= \left(\boldsymbol{D} \otimes \boldsymbol{I}\right)\left(\boldsymbol{M}_{\boldsymbol{\lambda}}^{(\boldsymbol{0})} \otimes \boldsymbol{M}_{\boldsymbol{\mu}}^{(\boldsymbol{1})}\right)\left(\boldsymbol{D}^{\boldsymbol{T}} \otimes \boldsymbol{I}^{\boldsymbol{T}}\right) \\ &= \left(\boldsymbol{D}\boldsymbol{M}_{\boldsymbol{\lambda}}^{(\boldsymbol{0})}\boldsymbol{D}^{\boldsymbol{T}}\right) \otimes \boldsymbol{M}_{\boldsymbol{\mu}}^{(\boldsymbol{1})} \end{aligned} \tag{2.14}$$

where D is the 1D mesh incidence matrix (discrete gradient operator) and I is the identity matrix. It's easy to see how the factor matrices contribute to the distorted spectrum.

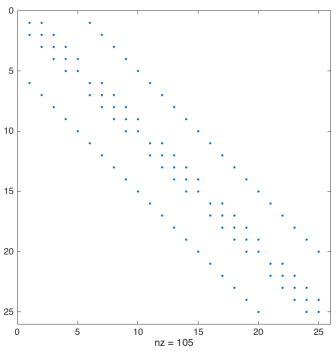

Figure 12: Finite Difference Laplacian Sparsity Pattern

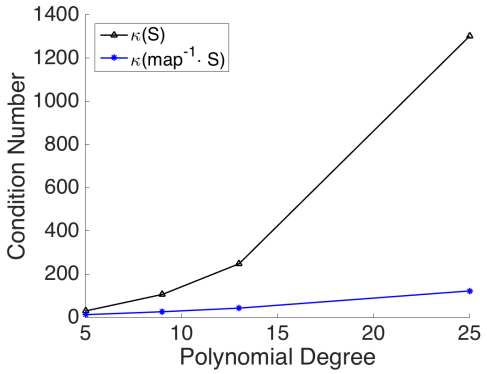

Figure 13: Condition Number Before and After Preconditioning

2.3 Schur Complement Approximation

The Schur complement of the mass matrix is the most difficult part of the work, where many solution method such as the two stage Uzawa Algorithm and its variations all break down. The Schur complement matrix usually involves mapping that changes vector space dimension and has a distorted spectrum. Uzawa algorithm is introduced while solving nonlinear programming problems, and one can see that the resulted Schur complement matrix has indeed a "nonlinearly" growing condition number. Thus preconditioning is crucial for such schemes to succeed.

From the previous inspection in figure 2, we noticed that the wedge product matrix $\boldsymbol{W}$ barely contribute to the growth of $\kappa(\boldsymbol{S})$. Thus we ignore the wedge product matrix while constructing the preconditioner. This leads to a preconditioner $\boldsymbol{G_0} = \boldsymbol{E}\boldsymbol{M_0^{-1}}\boldsymbol{E^T}$ that approximates the Schur complement matrix.

If we continue the analysis in equation 2.14, we notice that with symmetric numbering scheme, $\boldsymbol{M_u} = \boldsymbol{M_v}$, $\boldsymbol{D_u}$ and $\boldsymbol{D_v}$ are connected by row permutation $\boldsymbol{P}$:

$$\boldsymbol{P}\boldsymbol{D_u} = \boldsymbol{D_v}. \tag{2.15}$$

Thus,

$$\boldsymbol{G} = \boldsymbol{P}(\boldsymbol{D_v}\boldsymbol{M_v}^{-1}\boldsymbol{D_v}^{\boldsymbol{T}})\boldsymbol{P^T} + \boldsymbol{D_v}\boldsymbol{M_v}^{-1}\boldsymbol{D_v}^{\boldsymbol{T}}. \tag{2.16}$$

Further more, due to the symmetry of the co-ordinate system, it has to be the case that: $\boldsymbol{P} = \boldsymbol{P^T}$.

2.3.1 First Attempt

The first attempt of finding a preconditioner for $\boldsymbol{S}$ is done by following a conventional simplification in many literatures, which is to replace mass matrix with an identity matrix. We get $\boldsymbol{G_0} = \boldsymbol{map} = \boldsymbol{E} \cdot \boldsymbol{E^T}$, the finite difference Laplacian of a uniform mesh. This first attempt reveals the sparsity pattern of the preconditioner and shows how effective this preconditioner configuration could be. The sparsity pattern of $\boldsymbol{G_0}$ is given in figure 12. We know that $\boldsymbol{G_0} = \boldsymbol{P}(\boldsymbol{D_v}\boldsymbol{D_v}^{\boldsymbol{T}})\boldsymbol{P^T} + \boldsymbol{D_v}\boldsymbol{D_v}^{\boldsymbol{T}}$, and the mapping $f(\cdot) = \boldsymbol{P}[\cdot]\boldsymbol{P^T}$ won't change the diagonal dominance of any diagonal dominant matrix . Thus we hope that

$\boldsymbol{D_v M^{-1} D_v{}^T}$ in the preconditioner is diagonal dominant and has a very thin diagonal bandwidth. If we apply the mass matrix approximates developed in the previous section to obtain approximate of $\boldsymbol{G}$, the best sparsity we can get will be no better as in figure 12. Now we want to know if $\boldsymbol{G_0}$ could really improve the Schur complement matrices condition, thus we present the outcome of the first attempt in figure 13.

2.3.2 Schur Complement Approximate

In this section, we will use mass matrix approximates developed before to construct preconditioners for the Schur complement. First we will use the Jacobi preconditioner to compute $\boldsymbol{G_1} = \boldsymbol{E M_{Jacob}{}^{-1} E^T}$. Since $\boldsymbol{M_0}$ in this case is a diagonal matrix, the sparsity is at the best condition. Then, we compute $\boldsymbol{G_2}$ with orthogonal basis preconditioner M_{orth} and make comparisons between the preconditioners we have had by now. The results are shown in the following figure 14.

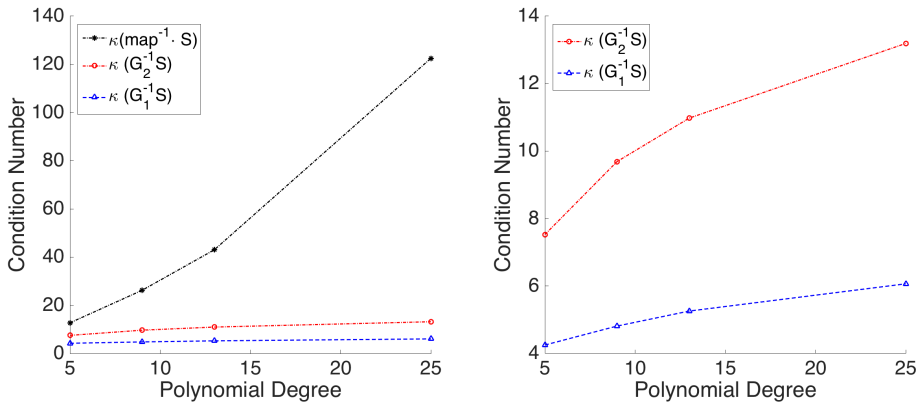

Figure 14: Condition Number After Preconditioning

Surprisingly, the diagonal scaling is the best preconditioner for the Schur complement. This is probably because we dropped the wedge product matrix while constructing the preconditioner.