Discrete Geometries of Mathematics and Physics

Mimetic spectral element method[1]

Assignment #1

From 1D to higher dimensions

Yi Zhang (张仪)
: www.mathischeap.com
@: zhangyi_aero@hotmail.com
: https://github.com/mathischeap

Note that we will restrict ourselves to no more than 2 dimensions in this series of assignments.

1 Function spaces

In $\mathbb{R}^2$, consider a bounded domain Ω that is contractible (its topology is same to a point, not to, for example, a doughnut) and has a smooth enough boundary $\partial\Omega$ (means its smoothness is enough to carry on all derivatives in this series of assignments).

The space of square integrable functions in Ω is the Sobolev space $L^2(\Omega)$,

$$L^2(\Omega) = \{\varphi \,|\langle\varphi, \varphi\rangle_\Omega < +\infty\}\,.$$

We use $\langle\cdot,\cdot\rangle_\Omega$ to denote a inner product. Namely, for any scalars a, b in Ω and any vectors $\boldsymbol{c}, \boldsymbol{d}$ in Ω, we have

$$\langle a, b\rangle_\Omega = \int_\Omega ab \,\mathrm{d}\Omega, \quad \langle \boldsymbol{c}, \boldsymbol{d}\rangle_\Omega = \int_\Omega \boldsymbol{c}\cdot\boldsymbol{d} \,\mathrm{d}\Omega.$$

Some subspaces of $L^2(\Omega)$ are

$$\begin{aligned}
H(\mathrm{curl};\Omega) &:= \left\{\omega \,\middle|\, \omega \in L^2(\Omega), \nabla\times\omega \in \left[L^2(\Omega)\right]^2\right\},\\
\boldsymbol{H}(\mathrm{div};\Omega) &:= \left\{\boldsymbol{u} \,\middle|\, \boldsymbol{u} \in \left[L^2(\Omega)\right]^2, \nabla\cdot\boldsymbol{u} \in L^2(\Omega)\right\},\\
H^1(\Omega) &:= \left\{\phi \,\middle|\, \phi \in L^2(\Omega), \nabla\phi \in \left[L^2(\Omega)\right]^2\right\},\\
\boldsymbol{H}(\mathrm{rot};\Omega) &:= \left\{\boldsymbol{\sigma} \,\middle|\, \boldsymbol{\sigma} \in \left[L^2(\Omega)\right]^2, \nabla\times\boldsymbol{\sigma} \in L^2(\Omega)\right\}.
\end{aligned}$$

[1] https://mathischeap.com/contents/teaching/advanced_numerical_methods/mimetic_spectral_element_method/main

They form two exact Hilbert complexes,

$$0 \longrightarrow H(\mathrm{curl};\Omega) \xrightarrow{\nabla\times} \boldsymbol{H}(\mathrm{div};\Omega) \xrightarrow{\nabla\cdot} L^2(\Omega) \longrightarrow 0,$$

$$0 \longrightarrow H^1(\Omega) \xrightarrow{\nabla} \boldsymbol{H}(\mathrm{rot};\Omega) \xrightarrow{\nabla\times} L^2(\Omega) \longrightarrow 0.$$

This means, for example, for any element ω in $H(\mathrm{curl};\Omega)$, $\nabla\times\omega = \begin{bmatrix}\dfrac{\partial\omega}{\partial y} & -\dfrac{\partial\omega}{\partial x}\end{bmatrix}^{\mathsf{T}}$ must be in $\boldsymbol{H}(\mathrm{div};\Omega)$. Similarly, you can understand other connections in these complexes. Note that at the continuous level, $H(\mathrm{curl};\Omega)$ and $H^1(\Omega)$ are the same space, and the operator $\nabla\times$ on $\omega \in H(\mathrm{curl};\Omega)$ is also called the perpendicular gradient. These spaces are all at the continuous level. Namely, they are infinite-dimensional. This is saying, for anyone of them, we cannot find a set of finite functions that forms a base for it. While the domain Ω is a sub-domain of two-dimensional space $\mathbb{R}^2$. You should think carefully about the difference.

Apparently, we cannot solve something of infinite dimensions with our computers. This is why numerical methods need *discretization*. Different methods have their own ways for discretization. And the *mimetic spectral element method* decides to approximate these Sobolev spaces with the finite-dimensional (or discrete) *mimetic spectral element spaces* which are denoted by $C(\Omega)$, $\boldsymbol{D}(\Omega)$, $G(\Omega)$, $\boldsymbol{R}(\Omega)$, and $S(\Omega)$. They are subspaces of the continuous spaces, i.e.

$$C(\Omega) \subset H(\mathrm{curl};\Omega), \quad \boldsymbol{D}(\Omega) \subset \boldsymbol{H}(\mathrm{div};\Omega), \quad G(\Omega) \subset H^1(\Omega), \quad \boldsymbol{R}(\Omega) \subset \boldsymbol{H}(\mathrm{rot};\Omega), \quad S(\Omega) \subset L^2(\Omega),$$

and also forms exact Hilbert complexes,

$$0 \longrightarrow C(\Omega) \xrightarrow{\nabla\times} \boldsymbol{D}(\Omega) \xrightarrow{\nabla\cdot} S(\Omega) \longrightarrow 0, \tag{1a}$$

$$0 \longrightarrow G(\Omega) \xrightarrow{\nabla} \boldsymbol{R}(\Omega) \xrightarrow{\nabla\times} S(\Omega) \longrightarrow 0. \tag{1b}$$

In this assignment, we will try to construct these discrete mimetic spectral element spaces.

2 Mimetic spectral element spaces

As we just introduced, a mimetic spectral element space is a discrete or finite-dimensional space. Recall what we have learned in linear algebra. If we can find a base for a mimetic spectral element space, we can express any element of it using the base and a set of expansion coefficients. Just like the space L or E in Assignment #0. Thus, the key of this assignment is constructing bases for the mimetic spectral element spaces.

In $\mathbb{R}^2$, we equip it with an orthogonal coordinate system (ξ,η) and consider a square domain $\Omega_{\mathrm{r}} := [-1,1]^2$. This domain is called the reference domain (or the reference element). We see that, along each axis, Ω_{r} ranges from -1 to 1. So, we can apply the partition in Assignment #0 (see (1) there) to both axes,

$$-1 = \xi_0 < \xi_1 < \xi_2 < \cdots < \xi_N = 1, \tag{2}$$

$$-1 = \eta_0 < \eta_1 < \eta_2 < \cdots < \eta_N = 1, \tag{3}$$

where $\xi_i = \eta_i$, $i \in \{0,1,\cdots,N\}$. Note that we do not have to use the same partition along both axes. However, to present the idea, using the same partition is a good choice. And also, it is usually the case in most scenarios.

With partitions (2) and (3), we can construct Lagrange polynomials and edges polynomials along ξ-axis and η-axis,

$$\begin{aligned} &l^i(\xi), \quad i \in \{0,1,\cdots,N\}, \\ &e^i(\xi), \quad i \in \{1,2,\cdots,N\}, \\ &l^j(\eta), \quad j \in \{0,1,\cdots,N\}, \end{aligned}$$

$$e^j(\eta), \quad j \in \{1, 2, \cdots, N\}.$$

These are one-dimensional polynomials. Using them, we construct two-dimensional polynomials as

$$\begin{aligned}
\mathtt{ll}^{ij}(\xi, \eta) &:= l^i(\xi) l^j(\eta), \quad i, j \in \{0, 1, \cdots, N\}, \\
\mathtt{le}^{ij}(\xi, \eta) &:= l^i(\xi) e^j(\eta), \quad i \in \{0, 1, \cdots, N\}, j \in \{1, 2, \cdots, N\}, \\
\mathtt{el}^{ij}(\xi, \eta) &:= e^i(\xi) l^j(\eta), \quad i \in \{1, 2, \cdots, N\}, j \in \{0, 1, \cdots, N\}, \\
\mathtt{ee}^{ij}(\xi, \eta) &:= e^i(\xi) e^j(\eta), \quad i, j \in \{1, 2, \cdots, N\}.
\end{aligned}$$

From the one-dimensional Kronecker delta properties in Assignment#0, we can easily derive two-dimensional Kronecker delta properties,

$$\mathtt{ll}^{ij}(\xi_m, \eta_n) = l^i(\xi_m) l^j(\eta_n) = \delta^{i,j}_{m,n} \tag{5a}$$

$$\mathtt{le}^{ij}(\xi_m, \eta_n) = l^i(\xi_m) \int_{\eta_{n-1}}^{\eta_n} e^j(\eta) \mathrm{d}\eta = \delta^{i,j}_{m,n}, \tag{5b}$$

$$\mathtt{el}^{ij}(\xi_m, \eta_n) = l^j(\eta_n) \int_{\xi_{m-1}}^{\xi_m} e^i(\xi) \mathrm{d}\xi = \delta^{i,j}_{m,n}, \tag{5c}$$

$$\mathtt{ee}^{ij}(\xi_m, \eta_n) = \int_{\xi_{m-1}}^{\xi_m} e^i(\xi) \mathrm{d}\xi \int_{\eta_{n-1}}^{\eta_n} e^j(\eta) \mathrm{d}\eta = \delta^{i,j}_{m,n}, \tag{5d}$$

where $\delta^{i,j}_{m,n} = \begin{cases} 1, & \text{if } i = m, j = n \\ 0, & \text{else} \end{cases}$. Then we can use them as basis functions of mimetic spectral element spaces in Ω_{r}. In details, we have

$$C(\Omega_{\mathrm{r}}) = G(\Omega_{\mathrm{r}}) := \mathrm{span}\left(\left\{ \mathtt{ll}^{ij}(\xi, \eta) \middle| \, i, j \in \{0, 1, \cdots, N\} \right\}\right), \tag{6a}$$

$$\boldsymbol{D}(\Omega_{\mathrm{r}}) := \begin{bmatrix} \mathrm{span}\left(\left\{ \mathtt{le}^{ij}(\xi, \eta) \middle| \, i \in \{0, 1, \cdots, N\}, j \in \{1, 2, \cdots, N\} \right\}\right) \\ \mathrm{span}\left(\left\{ \mathtt{el}^{ij}(\xi, \eta) \middle| \, i \in \{1, 2, \cdots, N\}, j \in \{0, 1, \cdots, N\} \right\}\right) \end{bmatrix}, \tag{6b}$$

$$\boldsymbol{R}(\Omega_{\mathrm{r}}) := \begin{bmatrix} \mathrm{span}\left(\left\{ \mathtt{el}^{ij}(\xi, \eta) \middle| \, i \in \{1, 2, \cdots, N\}, j \in \{0, 1, \cdots, N\} \right\}\right) \\ \mathrm{span}\left(\left\{ \mathtt{le}^{ij}(\xi, \eta) \middle| \, i \in \{0, 1, \cdots, N\}, j \in \{1, 2, \cdots, N\} \right\}\right) \end{bmatrix}, \tag{6c}$$

$$S(\Omega_{\mathrm{r}}) := \mathrm{span}\left(\left\{ \mathtt{ee}^{ij}(\xi, \eta) \middle| \, i, j \in \{1, 2, \cdots, N\} \right\}\right). \tag{6d}$$

Now, elements in these spaces can be expressed as

$$\omega_h = \sum_{i=0}^{N} \sum_{j=0}^{N} \mathtt{w}_{ij} \mathtt{ll}^{ij}(\xi, \eta), \qquad \omega_h \in C(\Omega_{\mathrm{r}}) \text{ or } G(\Omega_{\mathrm{r}}), \tag{7a}$$

$$\boldsymbol{u}_h = \begin{bmatrix} u_h \\ v_h \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{N} \sum_{j=1}^{N} \mathtt{u}_{ij} \mathtt{le}^{ij}(\xi, \eta) \\ \sum_{i=1}^{N} \sum_{j=0}^{N} \mathtt{v}_{ij} \mathtt{el}^{ij}(\xi, \eta) \end{bmatrix}, \qquad \boldsymbol{u}_h \in \boldsymbol{D}(\Omega_{\mathrm{r}}), \tag{7b}$$

$$\boldsymbol{\sigma}_h = \begin{bmatrix} \sigma_h \\ \tau_h \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{N} \sum_{j=0}^{N} \mathtt{s}_{ij} \mathtt{el}^{ij}(\xi, \eta) \\ \sum_{i=0}^{N} \sum_{j=1}^{N} \mathtt{t}_{ij} \mathtt{le}^{ij}(\xi, \eta) \end{bmatrix}, \qquad \boldsymbol{\sigma}_h \in \boldsymbol{R}(\Omega_{\mathrm{r}}), \tag{7c}$$

$$\phi_h = \sum_{i=1}^{N} \sum_{j=1}^{N} \mathtt{f}_{ij} \mathtt{ee}^{ij}(\xi, \eta), \qquad \phi_h \in S(\Omega_{\mathrm{r}}). \tag{7d}$$

Since the basis functions are two-dimensional polynomials, these elements are also polynomials. And because all basis functions are known, each set of expansion coefficient exclusively refers to one polynomial. For example, a particular set of expansion coefficients $\{\mathtt{w}_{ij}\}$[2] one-to-one refers to an polynomial ω_h. A particular set of expansion coefficients $\{\mathtt{u}_{ij}\} \cup \{\mathtt{v}_{ij}\}$ one-to-one refers to an polynomial $\boldsymbol{u}_h$. This is why these expansion coefficients are also called *degrees of freedom (DoF's)*.

[2] A shortcut of $\{\mathtt{w}_{ij} | \, i, j \in \{0, 1, \cdots, N\}\}$ to simplify the notation. Same convention is used for other sets.

In fact, (7) reveals the *reconstructions* for mimetic spectral element spaces in Ω_{r}.

The next important question is how to do *reductions* for these spaces. For $\omega \in H(\mathrm{curl}; \Omega_{\mathrm{r}})$ or $H^1(\Omega_{\mathrm{r}})$, to reduce ω to ω_h, we just need to evaluate ω at the partition nodes to compute w_{ij}, i.e.,

$$\mathsf{w}_{ij} = \omega\left(\xi_i, \eta_j\right). \tag{8}$$

For $\boldsymbol{u} = \begin{bmatrix} u \\ v \end{bmatrix} \in H(\mathrm{div}; \Omega_{\mathrm{r}})$, the expansion coefficients of $\boldsymbol{u}_h \in \boldsymbol{D}(\Omega_{\mathrm{r}})$ are computed through

$$\mathsf{u}_{ij} = \int_{\eta_{j-1}}^{\eta_j} u(\xi_i, \eta) \mathrm{d}\eta, \tag{9a}$$

$$\mathsf{v}_{ij} = \int_{\xi_{i-1}}^{\xi_i} v(\xi, \eta_j) \mathrm{d}\xi. \tag{9b}$$

For $\boldsymbol{\sigma} = \begin{bmatrix} \sigma \\ \tau \end{bmatrix} \in H(\mathrm{rot}; \Omega_{\mathrm{r}})$, the expansion coefficients of $\boldsymbol{\sigma}_h \in \boldsymbol{R}(\Omega_{\mathrm{r}})$ are computed through

$$\mathsf{s}_{ij} = \int_{\xi_{i-1}}^{\xi_i} \sigma(\xi, \eta_j) \mathrm{d}\xi, \tag{10a}$$

$$\mathsf{t}_{ij} = \int_{\eta_{j-1}}^{\eta_j} \tau(\xi_i, \eta) \mathrm{d}\eta. \tag{10b}$$

And for $\phi \in S(\Omega_{\mathrm{r}})$, the expansion coefficients of $\phi_h \in S(\Omega_{\mathrm{r}})$ are

$$\mathsf{f}_{ij} = \int_{\xi_{i-1}}^{\xi_i} \int_{\eta_{j-1}}^{\eta_j} \phi(\xi, \eta) \mathrm{d}\eta \mathrm{d}\xi. \tag{11}$$

Clear, the idea of these reductions lies behind the Kronecker delta properties (5).

Assignment 1.1.0: Projection for $\omega \in H(\mathrm{curl}; \Omega_{\mathrm{r}})$ or $H^1(\Omega_{\mathrm{r}})$

You need to program two functions, the first one does the reduction and the second one does the reconstruction.

```
1  def CG_space_reduction(nodes, func):
2      """Reduce the function "func" to mimetic spectral element space C or G defined
       over "nodes".
3
4      Parameters
5      ----------
6      nodes : np.ndarray
7          The partition of the interval I. It should be a 1d array of shape (m, ).
           For example, nodes = [-1, -0.8, -0.3, 0.3, 0.8, 1].
8      func :
9          The function to be reduced to the space.
10
11     Returns
12     -------
13     expansion_coefficients:
14         A 2d array of shape (m+1, m+1) that contains the expansion coefficients.
15
16     """
17
18 def CG_space_reconstruction(nodes, expansion_coefficients, xi, eta):
19     """Reconstruct the polynomial in the mimetic spectral element space C or G
```

```
    over meshgrid of "xi" and "eta".

    Parameters
    ----------
    nodes : np.ndarray
        The partition of the interval I. It should be a 1d array of shape (m, ).
        For example, nodes = [-1, -0.8, -0.3, 0.3, 0.8, 1].
    expansion_coefficients : np.ndarray
        A 2d array of shape (m+1, m+1) that contains the expansion coefficients.
    xi : np.ndarray
        The coordinates we evaluate the polynomial along the first axis. It should
        be a 1d array of shape (i, ). For example, xi = np.linspace(-1, 1, 100).
        The polynomial will be evaluated on "np.meshgrid(xi, eta, indexing='ij')".
    eta : np.ndarray
        The coordinates we evaluate the polynomial along the second axis. It
        should be a 1d array of shape (j, ). For example, eta = np.linspace(-1, 1,
        100). The polynomial will be evaluated on "np.meshgrid(xi, eta,
        indexing='ij')".

    Returns
    -------
    reconstructed_values : np.ndarray
      A 2d array of shape (i, j) that represents the reconstructed values at
      "np.meshgrid(xi, eta, indexing='ij')".

    """
```

Assignment 1.1.1: Projection for $\boldsymbol{u} \in H(\mathrm{div}; \Omega_r)$

You need to program two functions, the first one does the reduction and the second one does the reconstruction.

```
def D_space_reduction(nodes, func):
    """Reduce the function "func" to mimetic spectral element space D defined over
    "nodes".

    Parameters
    ----------
    nodes : np.ndarray
        The partition of the interval I. It should be a 1d array of shape (m, ).
        For example, nodes = [-1, -0.8, -0.3, 0.3, 0.8, 1].
    func :
        The function to be reduced to the space.

    Returns
    -------
    exp_coef_u:
        A 2d array of shape (m+1, m) that contains the expansion coefficients for
        the first component.
    exp_coef_v:
        A 2d array of shape (m, m+1) that contains the expansion coefficients for
        the second component.

    """

def D_space_reconstruction(nodes, exp_coef_u, exp_coef_v, xi, eta):
```

```
    """Reconstruct the polynomial in the mimetic spectral element space C or G
    over meshgrid of "xi" and "eta".

    Parameters
    ----------
    nodes : np.ndarray
        The partition of the interval I. It should be a 1d array of shape (m, ).
        For example, nodes = [-1, -0.8, -0.3, 0.3, 0.8, 1].
    exp_coef_u : np.ndarray
        A 2d array of shape (m+1, m) that contains the expansion coefficients of
        the first component.
    exp_coef_v : np.ndarray
        A 2d array of shape (m, m+1) that contains the expansion coefficients of
        the second component.
    xi : np.ndarray
        The coordinates we evaluate the polynomial along the first axis. It should
        be a 1d array of shape (i, ). For example, xi = np.linspace(-1, 1, 100).
        The polynomial will be evaluated on "np.meshgrid(xi, eta, indexing='ij')".
    eta : np.ndarray
        The coordinates we evaluate the polynomial along the second axis. It
        should be a 1d array of shape (j, ). For example, eta = np.linspace(-1, 1,
        100). The polynomial will be evaluated on "np.meshgrid(xi, eta,
        indexing='ij')".

    Returns
    -------
    reconstructed_values_u : np.ndarray
        A 2d array of shape (i, j) that represents the reconstructed values of the
        first component at "np.meshgrid(xi, eta, indexing='ij')".
    reconstructed_values_v : np.ndarray
        A 2d array of shape (i, j) that represents the reconstructed values of the
        second component at "np.meshgrid(xi, eta, indexing='ij')".

    """
```

Assignment 1.1.2: Projection for $\boldsymbol{\sigma} \in H(\mathrm{rot}; \Omega_r)$

You need to program two functions, the first one does the reduction and the second one does the reconstruction.

```
def R_space_reduction(nodes, func):
    """Reduce the function "func" to mimetic spectral element space R defined over
    "nodes".

    Parameters
    ----------
    nodes : np.ndarray
    The partition of the interval I. It should be a 1d array of shape (m, ). For
    example, nodes = [-1, -0.8, -0.3, 0.3, 0.8, 1].
    func :
    The function to be reduced to the space.

    Returns
    -------
    exp_coef_s:
        A 2d array of shape (m, m+1) that contains the expansion coefficients for
```

```
        the first component.
    exp_coef_t:
        A 2d array of shape (m+1, m) that contains the expansion coefficients for
        the second component.

    """

def R_space_reconstruction(nodes, exp_coef_s, exp_coef_t, xi, eta):
    """Reconstruct the polynomial in the mimetic spectral element space C or G
    over meshgrid of "xi" and "eta".

    Parameters
    ----------
    nodes : np.ndarray
        The partition of the interval I. It should be a 1d array of shape (m, ).
        For example, nodes = [-1, -0.8, -0.3, 0.3, 0.8, 1].
    exp_coef_s : np.ndarray
        A 2d array of shape (m, m+1) that contains the expansion coefficients of
        the first component.
    exp_coef_t : np.ndarray
        A 2d array of shape (m+1, m) that contains the expansion coefficients of
        the second component.
    xi : np.ndarray
        The coordinates we evaluate the polynomial along the first axis. It should
        be a 1d array of shape (i, ). For example, xi = np.linspace(-1, 1, 100).
        The polynomial will be evaluated on "np.meshgrid(xi, eta, indexing='ij')".
    eta : np.ndarray
        The coordinates we evaluate the polynomial along the second axis. It
        should be a 1d array of shape (j, ). For example, eta = np.linspace(-1, 1,
        100). The polynomial will be evaluated on "np.meshgrid(xi, eta,
        indexing='ij')".

    Returns
    -------
    reconstructed_values_s : np.ndarray
        A 2d array of shape (i, j) that represents the reconstructed values of the
        first component at "np.meshgrid(xi, eta, indexing='ij')".
    reconstructed_values_t : np.ndarray
        A 2d array of shape (i, j) that represents the reconstructed values of the
        second component at "np.meshgrid(xi, eta, indexing='ij')".

    """
```

Assignment 1.1.3: Projection for $\phi \in S(\Omega_r)$

You need to program two functions, the first one does the reduction and the second one does the reconstruction.

```
def S_space_reduction(nodes, func):
    """Reduce the function "func" to mimetic spectral element space S defined over
    "nodes".

    Parameters
    ----------
    nodes : np.ndarray
        The partition of the interval I. It should be a 1d array of shape (m, ).
```

```
        For example, nodes = [-1, -0.8, -0.3, 0.3, 0.8, 1].
    func :
        The function to be reduced to the space.

    Returns
    -------
    expansion_coefficients:
        A 2d array of shape (m, m) that contains the expansion coefficients.

    """

def S_space_reconstruction(nodes, expansion_coefficients, xi, eta):
    """Reconstruct the polynomial in the mimetic spectral element space C or G
    over meshgrid of "xi" and "eta".

    Parameters
    ----------
    nodes : np.ndarray
        The partition of the interval I. It should be a 1d array of shape (m, ).
        For example, nodes = [-1, -0.8, -0.3, 0.3, 0.8, 1].
    expansion_coefficients : np.ndarray
        A 2d array of shape (m, m) that contains the expansion coefficients.
    xi : np.ndarray
        The coordinates we evaluate the polynomial along the first axis. It should
        be a 1d array of shape (i, ). For example, xi = np.linspace(-1, 1, 100).
        The polynomial will be evaluated on "np.meshgrid(xi, eta, indexing='ij')".
    eta : np.ndarray
        The coordinates we evaluate the polynomial along the second axis. It
        should be a 1d array of shape (j, ). For example, eta = np.linspace(-1, 1,
        100). The polynomial will be evaluated on "np.meshgrid(xi, eta,
        indexing='ij')".

    Returns
    -------
    reconstructed_values : np.ndarray
        A 2d array of shape (i, j) that represents the reconstructed values at
        "np.meshgrid(xi, eta, indexing='ij')".

    """
```

3 Incidence matrices

So far, we have constructed mimetic spectral element spaces $C(\Omega)$, $G(\Omega)$, $\boldsymbol{D}(\Omega)$, $\boldsymbol{R}(\Omega)$, and $S(\Omega)$, and know how to project functions into them. However, the key structure between them, i.e. the Hilbert complexes

$$0 \longrightarrow C(\Omega) \xrightarrow{\nabla\times} \boldsymbol{D}(\Omega) \xrightarrow{\nabla\cdot} S(\Omega) \longrightarrow 0,$$
$$0 \longrightarrow G(\Omega) \xrightarrow{\nabla} \boldsymbol{R}(\Omega) \xrightarrow{\nabla\times} S(\Omega) \longrightarrow 0.$$

is not addressed yet. Recall that, in Assignment #0, the connection between the Lagrange space and the edge space can be interpolated as the incidence matrix. Naturally, we look forward to a similar sturcture in two-dimensions.

We repeat that polynomials in the mimetic spectral element spaces have general formats as

$$\omega_h = \sum_{i=0}^{N}\sum_{j=0}^{N} \mathtt{w}_{ij}\mathtt{ll}^{ij}(\xi,\eta), \qquad \omega_h \in C(\Omega_\mathrm{r}) \text{ or } G(\Omega_\mathrm{r}),$$

$$\boldsymbol{u}_h = \begin{bmatrix} u_h \\ v_h \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{N}\sum_{j=1}^{N} \mathtt{u}_{ij}\mathtt{le}^{ij}(\xi,\eta) \\ \sum_{i=1}^{N}\sum_{j=0}^{N} \mathtt{v}_{ij}\mathtt{el}^{ij}(\xi,\eta) \end{bmatrix}, \qquad \boldsymbol{u}_h \in \boldsymbol{D}(\Omega_\mathrm{r}),$$

$$\boldsymbol{\sigma}_h = \begin{bmatrix} \sigma_h \\ \tau_h \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{N}\sum_{j=0}^{N} \mathtt{s}_{ij}\mathtt{el}^{ij}(\xi,\eta) \\ \sum_{i=0}^{N}\sum_{j=1}^{N} \mathtt{t}_{ij}\mathtt{le}^{ij}(\xi,\eta) \end{bmatrix}, \qquad \boldsymbol{\sigma}_h \in \boldsymbol{R}(\Omega_\mathrm{r}),$$

$$\phi_h = \sum_{i=1}^{N}\sum_{j=1}^{N} \mathtt{f}_{ij}\mathtt{ee}^{ij}(\xi,\eta), \qquad \phi_h \in S(\Omega_\mathrm{r}).$$

In a geometric angle of view, the expansion coefficients represent degrees of freedom defined on points, edges, and faces, see Fig. 1.

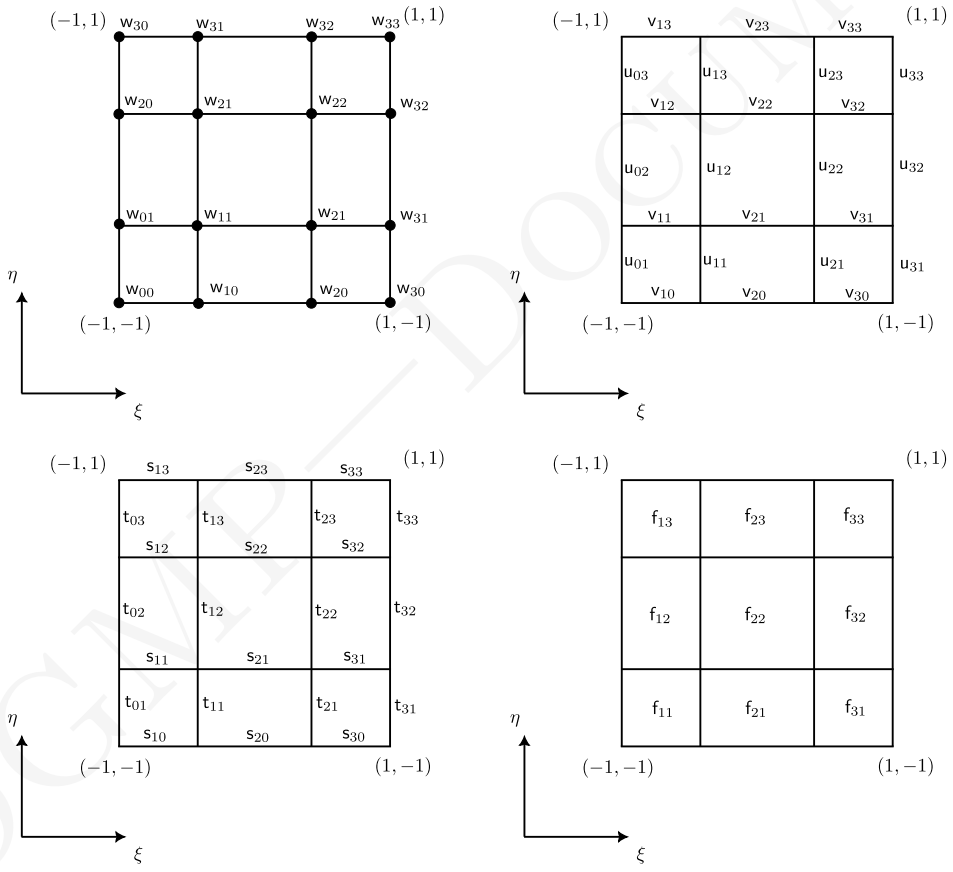

Fig. 1: Distributions of degrees of freedom in a geometric anlge of view.

Assume that these polynomials are projections of ω, $\boldsymbol{u}$, $\boldsymbol{\sigma}$, and ϕ in corresponding spaces,

$$\omega_h = \pi(\omega), \quad \boldsymbol{u}_h = \pi(\boldsymbol{u}), \quad \boldsymbol{\sigma}_h = \pi(\boldsymbol{\sigma}), \quad \phi_h = \pi(\phi).$$

From the properties of Lagrange polynomials and edge polynomials, we can easily get the colusions in following subsections.

3.1 Incidence matrix $\mathbb{E}_C$

If $\boldsymbol{u} = \nabla \times \omega$, we must have $\boldsymbol{u}_h = \nabla \times \omega_h$ and

$$\vec{\boldsymbol{u}} = \mathbb{E}_C \vec{\omega}. \tag{14}$$

The vectors are

$$\vec{\omega} = \begin{bmatrix} \mathsf{w}_1 \\ \mathsf{w}_2 \\ \vdots \\ \mathsf{w}_{(N+1)^2} \end{bmatrix}$$

where $\mathsf{w}_{j\times(N+1)+i+1} = \mathsf{w}_{ij}, i, j \in \{0, 1, \cdots, N\}$, and

$$\vec{\boldsymbol{u}} = \begin{bmatrix} \mathsf{u}_1 \\ \mathsf{u}_2 \\ \vdots \\ \mathsf{u}_{(N+1)\times N} \\ \mathsf{v}_1 \\ \mathsf{v}_2 \\ \vdots \\ \mathsf{v}_{N\times(N+1)} \end{bmatrix}$$

where $\mathsf{u}_{(j-1)\times(N+1)+i+1} = \mathsf{u}_{ij}, i \in \{0, 1, \cdots, N\}, j \in \{1, 2, \cdots, N\}$ and $\mathsf{v}_{j\times N+i} = \mathsf{v}_{ij}, i \in \{1, 2, \cdots, N\}, j \in \{0, 1, \cdots, N\}$. And the matrix $\mathbb{E}_C$ is the incidence matrix who has $2N \times (N+1)$ rows and $(N+1)^2$ columns. It is a sparse matrix and the non-zero entries of it are

$$\begin{aligned}
&\mathbb{E}_C|_{(j-1)\times(N+1)+i+1,(j-1)\times(N+1)+i+1} = -1, \quad i \in \{0, 1, \cdots, N\}, j \in \{1, 2, \cdots, N\}, \\
&\mathbb{E}_C|_{(j-1)\times(N+1)+i+1,j\times(N+1)+i+1} = 1, \quad i \in \{0, 1, \cdots, N\}, j \in \{1, 2, \cdots, N\}, \\
&\mathbb{E}_C|_{N\times(N+1)+j\times N+i,j\times(N+1)+i} = 1, \quad i \in \{1, 2, \cdots, N\}, j \in \{0, 1, \cdots, N\}, \\
&\mathbb{E}_C|_{N\times(N+1)+j\times N+i,j\times(N+1)+i+1} = -1, \quad i \in \{1, 2, \cdots, N\}, j \in \{0, 1, \cdots, N\}.
\end{aligned}$$

See Fig. 2 for an illustration of the geometric representation for the incidence matrix $\mathbb{E}_C$.

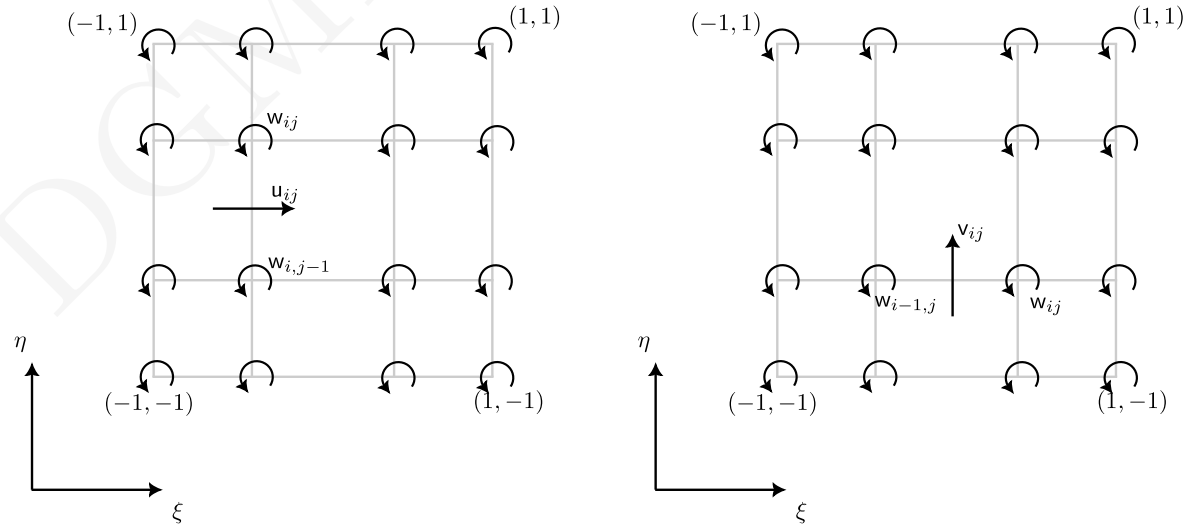

Fig. 2: An illustration of the geometric representation for the incidence matrix $\mathbb{E}_C$.

3.2 Incidence matrix $\mathbb{E}_G$

If $\boldsymbol{\sigma} = \nabla\omega$, we must have $\boldsymbol{\sigma}_h = \nabla\omega_h$ and

$$\vec{\boldsymbol{\sigma}} = \mathbb{E}_G\vec{\omega}. \tag{15}$$

The vector $\vec{\boldsymbol{\sigma}}$ is

$$\vec{\boldsymbol{\sigma}} = \begin{bmatrix} \mathsf{s}_1 \\ \mathsf{s}_2 \\ \vdots \\ \mathsf{s}_{(N+1)\times N} \\ \mathsf{t}_1 \\ \mathsf{t}_2 \\ \vdots \\ \mathsf{t}_{N\times(N+1)} \end{bmatrix}$$

where $\mathsf{s}_{j\times N+i} = \mathsf{s}_{ij}, i \in \{1,2,\cdots,N\}, j \in \{0,1,\cdots,N\}$ and $\mathsf{t}_{(j-1)\times(N+1)+i+1} = \mathsf{t}_{ij}, i \in \{0,1,\cdots,N\}, j \in \{1,2,\cdots,N\}$. And the matrix $\mathbb{E}_G$ is the incidence matrix who has $2N\times(N+1)$ rows and $(N+1)^2$ columns. It is a sparse matrix and the non-zero entries of it are

$$\begin{aligned}
&\mathbb{E}_G|_{j\times N+i,j\times(N+1)+i} = -1, \quad i \in \{1,2,\cdots,N\}, j \in \{0,1,\cdots,N\},\\
&\mathbb{E}_G|_{j\times N+i,j\times(N+1)+i+1} = 1, \quad i \in \{1,2,\cdots,N\}, j \in \{0,1,\cdots,N\},\\
&\mathbb{E}_G|_{N\times(N+1)+(j-1)\times(N+1)+i+1,(j-1)\times(N+1)+i+1} = -1, \quad i \in \{0,1,\cdots,N\}, j \in \{1,2,\cdots,N\},\\
&\mathbb{E}_G|_{N\times(N+1)+(j-1)\times(N+1)+i+1,j\times(N+1)+i+1} = 1, \quad i \in \{0,1,\cdots,N\}, j \in \{1,2,\cdots,N\}.
\end{aligned}$$

See Fig. 3 for an illustration of the geometric representation for the incidence matrix $\mathbb{E}_G$.

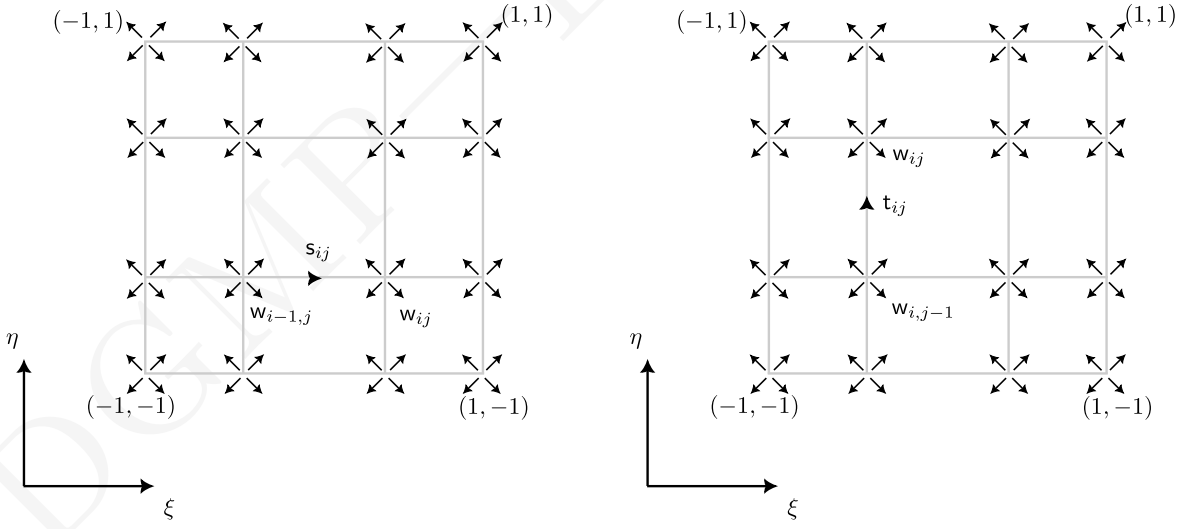

Fig. 3: An illustration of the geometric representation for the incidence matrix $\mathbb{E}_G$.

3.3 Incidence matrix $\mathbb{E}_D$

If $\phi = \nabla\cdot\boldsymbol{u}$, we must have $\phi_h = \nabla\cdot\boldsymbol{u}_h$ and

$$\vec{\phi} = \mathbb{E}_D\vec{\boldsymbol{u}}. \tag{16}$$

The vector $\vec{\phi}$ is

$$\vec{\phi} = \begin{bmatrix} \mathsf{f}_1 \\ \mathsf{f}_2 \\ \vdots \\ \mathsf{f}_{N^2} \end{bmatrix}$$

where $\mathsf{f}_{(j-1)\times N+i} = \mathsf{w}_{ij}, i, j \in \{1, 2, \cdots, N\}$. The matrix $\mathbb{E}_D$ is the incidence matrix who has N^2 rows and $2N \times (N+1)$ columns. It is a sparse matrix and the non-zero entries of it are

$$\begin{aligned}
&\mathbb{E}_D|_{(j-1)\times N+i,(j-1)\times(N+1)+i} = -1, \quad i, j \in \{1, 2, \cdots, N\}, \\
&\mathbb{E}_D|_{(j-1)\times N+i,(j-1)\times(N+1)+i+1} = 1, \quad i, j \in \{1, 2, \cdots, N\}, \\
&\mathbb{E}_D|_{(j-1)\times N+i,N\times(N+1)+(j-1)\times N+i} = -1, \quad i, j \in \{1, 2, \cdots, N\}, \\
&\mathbb{E}_D|_{(j-1)\times N+i,N\times(N+1)+j\times N+i} = 1, \quad i, j \in \{1, 2, \cdots, N\}.
\end{aligned}$$

See Fig. 4 for an illustration of the geometric representation for the incidence matrix $\mathbb{E}_D$.

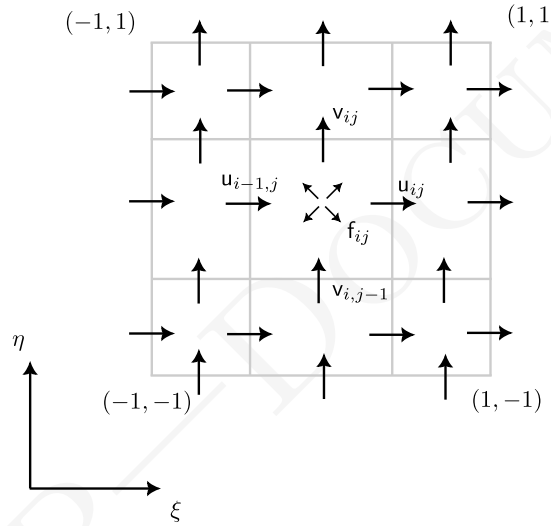

Fig. 4: An illustration of the geometric representation for the incidence matrix $\mathbb{E}_D$.

3.4 Incidence matrix $\mathbb{E}_R$

If $\phi = \nabla \times \boldsymbol{\sigma}$, we must have $\phi_h = \nabla \times \boldsymbol{\sigma}_h$ and

$$\vec{\phi} = \mathbb{E}_R \vec{\boldsymbol{\sigma}}. \tag{17}$$

The matrix $\mathbb{E}_R$ is the incidence matrix who has N^2 rows and $2N \times (N+1)$ columns. It is a sparse matrix and the non-zero entries of it are

$$\begin{aligned}
&\mathbb{E}_R|_{(j-1)\times N+i,N\times(N+1)+(j-1)\times(N+1)+i} = -1, \quad i, j \in \{1, 2, \cdots, N\}, \\
&\mathbb{E}_R|_{(j-1)\times N+i,N\times(N+1)+(j-1)\times(N+1)+i+1} = 1, \quad i, j \in \{1, 2, \cdots, N\}. \\
&\mathbb{E}_R|_{(j-1)\times N+i,(j-1)\times N+i} = 1, \quad i, j \in \{1, 2, \cdots, N\}, \\
&\mathbb{E}_R|_{(j-1)\times N+i,j\times N+i} = -1, \quad i, j \in \{1, 2, \cdots, N\}.
\end{aligned}$$

See Fig. 5 for an illustration of the geometric representation for the incidence matrix $\mathbb{E}_R$.

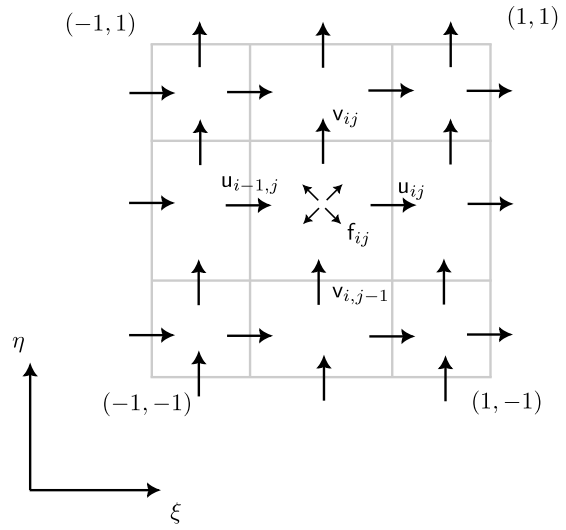

Fig. 5: An illustration of the geometric representation for the incidence matrix $\mathbb{E}_R$.

Assignment 1.2.0: Incidence matrix $\mathbb{E}_C$

You need to program a function to compute the incidence matrix $\mathbb{E}_C$. Then you can play with the incidence matrix to verify (14).

Assignment 1.2.1: Incidence matrix $\mathbb{E}_G$

You need to program a function to compute the incidence matrix $\mathbb{E}_G$. Then you can play with the incidence matrix to verify (15).

Assignment 1.2.2: Incidence matrix $\mathbb{E}_D$

You need to program a function to compute the incidence matrix $\mathbb{E}_D$. Then you can play with the incidence matrix to verify (16).

Assignment 1.2.3: Incidence matrix $\mathbb{E}_R$

You need to program a function to compute the incidence matrix $\mathbb{E}_R$. Then you can play with the incidence matrix to verify (17).

4 Three dimensions

In this series of assignments, we will not go to three dimensions. But the approach is similar. Read [1, Chapter 2] for the details.

References

[1] Y. Zhang, Mimetic Spectral Element Method and Extensions toward Higher Computational Efficiency (2022).