AMCS 394E: Contemp. Topics in Computational Science.
Computing with the finite element method

David I. Ketcheson and Manuel Quezada de Luna

KAUST
King Abdullah University of
Science and Technology

From last week

From last week

* We need to compute quantitiies per DoF. For example

$$F_i = \int_\Omega f(x)\varphi_i dx, \quad M_{ij} = \int_\Omega \varphi_i \varphi_j dx,$$

where $i = 1, ..., \dim(V_h) =$ Total number of DoF.

From last week

* We need to compute quantitiies per DoF. For example

$$F_i = \int_\Omega f(x)\varphi_i dx, \quad M_{ij} = \int_\Omega \varphi_i \varphi_j dx,$$

  where $i = 1, ..., \dim(V_h) =$ Total number of DoF.
* These quantities per DoF are entries of vectors and matrices.

From last week

* We need to compute quantitiies per DoF. For example

$$F_i = \int_\Omega f(x)\varphi_i dx, \quad M_{ij} = \int_\Omega \varphi_i \varphi_j dx,$$

where $i = 1, ..., \dim(V_h) =$ Total number of DoF.
* These quantities per DoF are entries of vectors and matrices.
* We break up the integrals into integrals per element.

From last week

* We need to compute quantitiies per DoF. For example

$$F_i = \int_\Omega f(x)\varphi_i dx, \quad M_{ij} = \int_\Omega \varphi_i \varphi_j dx,$$

where $i = 1, ..., \dim(V_h) =$ Total number of DoF.

* These quantities per DoF are entries of vectors and matrices.
* We break up the integrals into integrals per element.
* In each element, we need to compute

$$F_i = \int_K f(x)\varphi_i dx, \quad M_{ij} = \int_K \varphi_i \varphi_j dx,$$

where $i = 1, ..., $ DoFs per element in K.

From last week

* We need to compute quantitiies per DoF. For example

$$F_i = \int_\Omega f(x)\varphi_i dx, \quad M_{ij} = \int_\Omega \varphi_i \varphi_j dx,$$

where $i = 1, ..., \dim(V_h) =$ Total number of DoF.

* These quantities per DoF are entries of vectors and matrices.
* We break up the integrals into integrals per element.
* In each element, we need to compute

$$F_i = \int_K f(x)\varphi_i dx, \quad M_{ij} = \int_K \varphi_i \varphi_j dx,$$

where $i = 1, ...,$ DoFs per element in K.

* We need to combine the element based operators.

From last week: The finite element (FE) loop

The kernel of most FE codes is the FE loop

From last week: The finite element (FE) loop

The kernel of most FE codes is the FE loop

* Loop on elements (since we broke the integrals into elements).

From last week: The finite element (FE) loop

The kernel of most FE codes is the FE loop

* Loop on elements (since we broke the integrals into elements).
* Compute element based integrals (vectors and matrices).

From last week: The finite element (FE) loop

The kernel of most FE codes is the FE loop

* Loop on elements (since we broke the integrals into elements).
* Compute element based integrals (vectors and matrices).
* Combine (or assemble) the element integrals to global integrals.

From last week: The finite element (FE) loop

The kernel of most FE codes is the FE loop

* Loop on elements (since we broke the integrals into elements).
* Compute element based integrals (vectors and matrices).
* Combine (or assemble) the element integrals to global integrals.

How to compute the element based integrals?

For each element, proceed as follows:

<u>From last week:</u> The finite element (FE) loop

The kernel of most FE codes is the FE loop

* Loop on elements (since we broke the integrals into elements).
* Compute element based integrals (vectors and matrices).
* Combine (or assemble) the element integrals to global integrals.

How to compute the element based integrals?

For each element, proceed as follows:

* Compute all quantities wrt the reference element.

<u>From last week:</u> The finite element (FE) loop

The kernel of most FE codes is the FE loop

* Loop on elements (since we broke the integrals into elements).
* Compute element based integrals (vectors and matrices).
* Combine (or assemble) the element integrals to global integrals.

How to compute the element based integrals?

For each element, proceed as follows:

* Compute all quantities wrt the reference element.
* Loop on quadrature points (since we use quadratures).

<u>From last week:</u> The finite element (FE) loop

The kernel of most FE codes is the FE loop

- Loop on elements (since we broke the integrals into elements).
- Compute element based integrals (vectors and matrices).
- Combine (or assemble) the element integrals to global integrals.

How to compute the element based integrals?

For each element, proceed as follows:

- Compute all quantities wrt the reference element.
- Loop on quadrature points (since we use quadratures).
- Loop on i-DoFs.

From last week: The finite element (FE) loop

The kernel of most FE codes is the FE loop

* Loop on elements (since we broke the integrals into elements).
* Compute element based integrals (vectors and matrices).
* Combine (or assemble) the element integrals to global integrals.

How to compute the element based integrals?

For each element, proceed as follows:

* Compute all quantities wrt the reference element.
* Loop on quadrature points (since we use quadratures).
* Loop on i-DoFs.
* Loop on j-DoFs and more indices if needed.

From last week: The finite element (FE) loop

The FE loop looks as follows:

From last week: The finite element (FE) loop

The FE loop looks as follows:

for $K = 1$ to N_{el} **do**

end for

From last week: The finite element (FE) loop

The FE loop looks as follows:

for $K = 1$ to N_{el} **do**
Compute quantities wrt reference element.

end for

From last week: The finite element (FE) loop

The FE loop looks as follows:

for $K = 1$ to N_{el} **do**
  Compute quantities wrt reference element.
  for $q = 1$ to N_q **do**

  end for
end for

From last week: The finite element (FE) loop

The FE loop looks as follows:

for $K = 1$ to N_{el} **do**
  Compute quantities wrt reference element.
  for $q = 1$ to N_q **do**
    for $i = 1$ to DoFs per element **do**

    end for

  end for
end for

From last week: The finite element (FE) loop

The FE loop looks as follows:

for $K = 1$ to N_{el} **do**
  Compute quantities wrt reference element.
  for $q = 1$ to N_q **do**
    for $i = 1$ to DoFs per element **do**
      Compute element based vectors.
    end for
  end for
end for

From last week: The finite element (FE) loop

The FE loop looks as follows:

for $K = 1$ to N_{el} **do**
  Compute quantities wrt reference element.
  for $q = 1$ to N_q **do**
    for $i = 1$ to DoFs per element **do**
      Compute element based vectors.
      for $j = 1$ to DoFs per element **do**

      end for
    end for

  end for
end for

From last week: The finite element (FE) loop

The FE loop looks as follows:

```
for K = 1 to N_el do
    Compute quantities wrt reference element.
    for q = 1 to N_q do
        for i = 1 to DoFs per element do
            Compute element based vectors.
            for j = 1 to DoFs per element do
                Compute element based matrices.
            end for
        end for

    end for
end for
```

From last week: The finite element (FE) loop

The FE loop looks as follows:

for $K = 1$ to N_{el} **do**
  Compute quantities wrt reference element.
  for $q = 1$ to N_q **do**
    for $i = 1$ to DoFs per element **do**
      Compute element based vectors.
      for $j = 1$ to DoFs per element **do**
        Compute element based matrices.
      end for
    end for
    Assemble from local to global operators.
  end for
end for

From last week: The finite element (FE) loop

The FE loop looks as follows:

```
for K = 1 to N_el do
    Compute quantities wrt reference element.
    for q = 1 to N_q do
        for i = 1 to DoFs per element do
            Compute element based vectors.
            for j = 1 to DoFs per element do
                Compute element based matrices.
            end for
        end for
        Assemble from local to global operators.
    end for
end for
```

<u>From last week:</u> The finite element (FE) loop

The FE loop looks as follows:

```
for K = 1 to N_el do
    Compute quantities wrt reference element.
    for q = 1 to N_q do
        for i = 1 to DoFs per element do
            Compute element based vectors.
            for j = 1 to DoFs per element do
                Compute element based matrices.
            end for
        end for
        Assemble from local to global operators.
    end for
end for
```

Basis functions for C^0 piecewise polynomials in 2D:

We only need to define the shape functions at the reference element.

Basis functions for C^0 piecewise polynomials in 2D:

We only need to define the shape functions at the reference element.

Quadrilateral elements

Basis functions for C^0 piecewise polynomials in 2D:

We only need to define the shape functions at the reference element.

Quadrilateral elements

* Shape functions are bilinear, biquadratic, etc.

Basis functions for C^0 piecewise polynomials in 2D:

We only need to define the shape functions at the reference element.

Quadrilateral elements

* Shape functions are bilinear, biquadratic, etc.
* How many DoFs do I need per element?

Basis functions for C^0 piecewise polynomials in 2D:

We only need to define the shape functions at the reference element.

Quadrilateral elements

* Shape functions are bilinear, biquadratic, etc.
* How many DoFs do I need per element?
* The shape functions can be obtained via tensor products.

Basis functions for C^0 piecewise polynomials in 2D:

We only need to define the shape functions at the reference element.

Quadrilateral elements

* Shape functions are bilinear, biquadratic, etc.
* How many DoFs do I need per element?
* The shape functions can be obtained via tensor products.
* Deal.ii only considers this type of elements.

Basis functions for C^0 piecewise polynomials in 2D:

We only need to define the shape functions at the reference element.

Quadrilateral elements

* Shape functions are bilinear, biquadratic, etc.
* How many DoFs do I need per element?
* The shape functions can be obtained via tensor products.
* Deal.ii only considers this type of elements.
* Other libraries (e.g. MFEM or Proteus can consider multiple type of elements).

Basis functions for C^0 piecewise polynomials in 2D:

We only need to define the shape functions at the reference element.

Quadrilateral elements

* Shape functions are bilinear, biquadratic, etc.
* How many DoFs do I need per element?
* The shape functions can be obtained via tensor products.
* Deal.ii only considers this type of elements.
* Other libraries (e.g. MFEM or Proteus can consider multiple type of elements).

Triangular elements

Basis functions for C^0 piecewise polynomials in 2D:

We only need to define the shape functions at the reference element.

Quadrilateral elements

* Shape functions are bilinear, biquadratic, etc.
* How many DoFs do I need per element?
* The shape functions can be obtained via tensor products.
* Deal.ii only considers this type of elements.
* Other libraries (e.g. MFEM or Proteus can consider multiple type of elements).

Triangular elements

* Shape functions are linear, quadratic, etc.

Basis functions for C^0 piecewise polynomials in 2D:

We only need to define the shape functions at the reference element.

Quadrilateral elements

* Shape functions are bilinear, biquadratic, etc.
* How many DoFs do I need per element?
* The shape functions can be obtained via tensor products.
* Deal.ii only considers this type of elements.
* Other libraries (e.g. MFEM or Proteus can consider multiple type of elements).

Triangular elements

* Shape functions are linear, quadratic, etc.
* How many DoFs do I need per element?

<u>From last week:</u> The finite element (FE) loop

The FE loop looks as follows:

for $K = 1$ to N_{el} **do**
  Compute quantities wrt reference element.
  for $q = 1$ to N_q **do**
    for $i = 1$ to DoFs per element **do**
      Compute element based vectors.
      for $j = 1$ to DoFs per element **do**
        Compute element based matrices.
      end for
    end for
    Assemble from local to global operators.
  end for
end for

Transformations from physical to reference elements

We integrate the element based vectors and matrices using the reference element. So we need the corresponding transformations.

Transformations from physical to reference elements

We integrate the element based vectors and matrices using the reference element. So we need the corresponding transformations.

Simplices in 2D

Let's consider the transformations for triangles.

Transformations from physical to reference elements

We integrate the element based vectors and matrices using the reference element. So we need the corresponding transformations.

Simplices in 2D

Let's consider the transformations for triangles.

Quads in 2D

The transformations for quadrilaterals are not affine transformations. We can still get them via a linear space in the reference element.

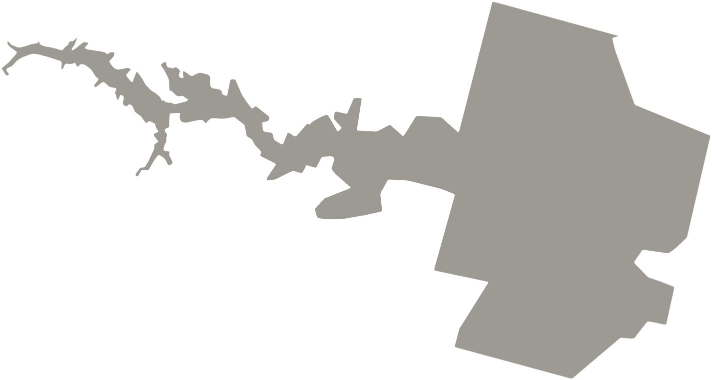
From last week: The finite element (FE) loop

The FE loop looks as follows:

for $K = 1$ to N_{el} **do**
  Compute quantities wrt reference element.
  for $q = 1$ to N_q **do**
    for $i = 1$ to DoFs per element **do**
      Compute element based vectors.
      for $j = 1$ to DoFs per element **do**
        Compute element based matrices.
      end for
    end for
    Assemble from local to global operators.
  end for
end for

Mesh and data structures

Consider a domain Ω



Mesh and data structures

Mesh and nodes

Mesh and data structures

Mesh and nodes

* The mesh is a discretization of Ω into elements.

Mesh and data structures

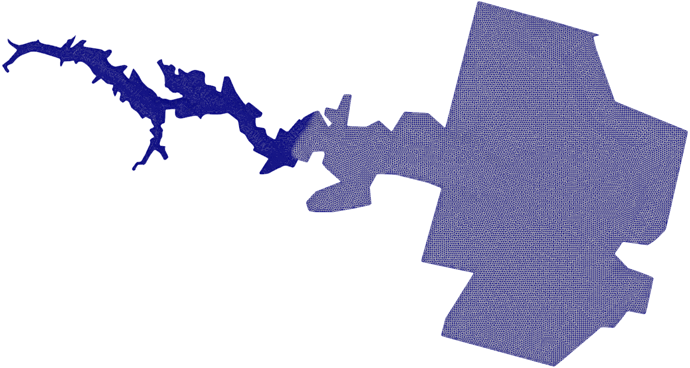
Mesh and nodes

* The mesh is a discretization of Ω into elements.
* The nodes are the corners of these elements.

Mesh and data structures

Mesh and nodes

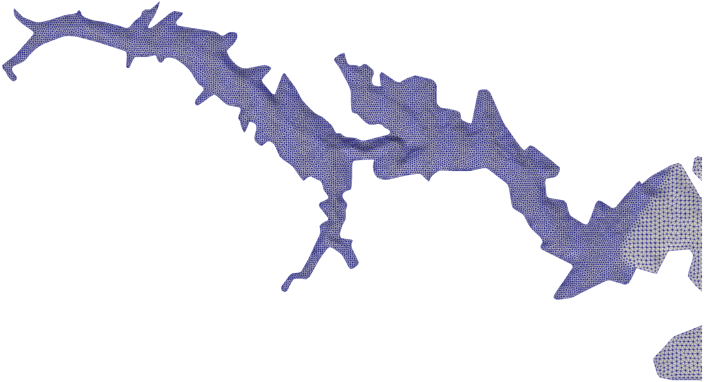
* The mesh is a discretization of Ω into elements.
* The nodes are the corners of these elements.



Mesh and data structures

Mesh and nodes

* The mesh is a discretization of Ω into elements.
* The nodes are the corners of these elements.



Mesh and data structures

Some definitions based on the geometry:

Mesh and data structures

Some definitions based on the geometry:

* Let N_{geo} be the number of nodes.

Mesh and data structures

Some definitions based on the geometry:

* Let N_{geo} be the number of nodes.
* Let N_{el} be the number of elements.

Mesh and data structures

Some definitions based on the geometry:

* Let N_{geo} be the number of nodes.
* Let N_{el} be the number of elements.
* Let N_{faces} be the number of faces in the boundary.

Mesh and data structures

Some definitions based on the geometry:

* Let N_{geo} be the number of nodes.
* Let N_{el} be the number of elements.
* Let N_{faces} be the number of faces in the boundary.
* N^e_{geo} be the number of nodes in element e.

Mesh and data structures

Some definitions based on the geometry:

* Let N_{geo} be the number of nodes.
* Let N_{el} be the number of elements.
* Let N_{faces} be the number of faces in the boundary.
* N^{e}_{geo} be the number of nodes in element e.
* N^{f}_{geo} be the number of nodes in face f.

Mesh and data structures

Some definitions based on the geometry:

* Let N_{geo} be the number of nodes.
* Let N_{el} be the number of elements.
* Let N_{faces} be the number of faces in the boundary.
* N^e_{geo} be the number of nodes in element e.
* N^f_{geo} be the number of nodes in face f.

Some definitions based on the space:

Mesh and data structures

Some definitions based on the geometry:

* Let N_{geo} be the number of nodes.
* Let N_{el} be the number of elements.
* Let N_{faces} be the number of faces in the boundary.
* N^e_{geo} be the number of nodes in element e.
* N^f_{geo} be the number of nodes in face f.

Some definitions based on the space:

* Let $N_h = \dim(V_h)$ be the total number of DoF.

Mesh and data structures

Some definitions based on the geometry:

* Let N_{geo} be the number of nodes.
* Let N_{el} be the number of elements.
* Let N_{faces} be the number of faces in the boundary.
* N_{geo}^{e} be the number of nodes in element e.
* N_{geo}^{f} be the number of nodes in face f.

Some definitions based on the space:

* Let $N_h = \dim(V_h)$ be the total number of DoF.
* If $p = 1$, $N_{\text{geo}} = N_h$.

Mesh and data structures

Some definitions based on the geometry:

* Let N_{geo} be the number of nodes.
* Let N_{el} be the number of elements.
* Let N_{faces} be the number of faces in the boundary.
* N^e_{geo} be the number of nodes in element e.
* N^f_{geo} be the number of nodes in face f.

Some definitions based on the space:

* Let $N_h = \dim(V_h)$ be the total number of DoF.
* If $p = 1$, $N_{\text{geo}} = N_h$.
* Let N^e_h be the number of DoF in element e.

Mesh and data structures

Some definitions based on the geometry:

* Let N_{geo} be the number of nodes.
* Let N_{el} be the number of elements.
* Let N_{faces} be the number of faces in the boundary.
* N^e_{geo} be the number of nodes in element e.
* N^f_{geo} be the number of nodes in face f.

Some definitions based on the space:

* Let $N_h = \dim(V_h)$ be the total number of DoF.
* If $p = 1$, $N_{\text{geo}} = N_h$.
* Let N^e_h be the number of DoF in element e.
* Let N^f_h be the number of DoF in face f.

Mesh and data structures

Connectivity matrix for elements

Mesh and data structures

Connectivity matrix for elements

* This is a matrix $C \in \mathbb{R}^{N_{\text{el}} \times N_{\text{geo}}^{e}}$.

Mesh and data structures

Connectivity matrix for elements

* This is a matrix $C \in \mathbb{R}^{N_{\text{el}} \times N^e_{\text{geo}}}$.
* The i-th row corresponds to the i-th element.

Mesh and data structures

Connectivity matrix for elements

* This is a matrix $C \in \mathbb{R}^{N_{\text{el}} \times N^e_{\text{geo}}}$.
* The i-th row corresponds to the i-th element.
* C_{ij} has the global index of the j-th local node in element i.

Mesh and data structures

Connectivity matrix for elements

* This is a matrix $C \in \mathbb{R}^{N_{\mathrm{el}} \times N_{\mathrm{geo}}^{e}}$.
* The i-th row corresponds to the i-th element.
* C_{ij} has the global index of the j-th local node in element i.
* Consider an example.

Mesh and data structures

Connectivity matrix for elements

* This is a matrix $C \in \mathbb{R}^{N_{\text{el}} \times N^e_{\text{geo}}}$.
* The i-th row corresponds to the i-th element.
* C_{ij} has the global index of the j-th local node in element i.
* Consider an example.

Connectivity matrix for faces

Mesh and data structures

Connectivity matrix for elements

* This is a matrix $C \in \mathbb{R}^{N_{\text{el}} \times N^e_{\text{geo}}}$.
* The i-th row corresponds to the i-th element.
* C_{ij} has the global index of the j-th local node in element i.
* Consider an example.

Connectivity matrix for faces

* This is a matrix $C^f \in \mathbb{R}^{N_{\text{faces}} \times N^f_{\text{geo}}}$.

Mesh and data structures

Connectivity matrix for elements

* This is a matrix $C \in \mathbb{R}^{N_{\text{el}} \times N^e_{\text{geo}}}$.
* The i-th row corresponds to the i-th element.
* C_{ij} has the global index of the j-th local node in element i.
* Consider an example.

Connectivity matrix for faces

* This is a matrix $C^f \in \mathbb{R}^{N_{\text{faces}} \times N^f_{\text{geo}}}$.
* The i-th row corresponds to the i-th face.

Mesh and data structures

Connectivity matrix for elements

* This is a matrix $C \in \mathbb{R}^{N_{\mathrm{el}} \times N^e_{\mathrm{geo}}}$.
* The i-th row corresponds to the i-th element.
* C_{ij} has the global index of the j-th local node in element i.
* Consider an example.

Connectivity matrix for faces

* This is a matrix $C^f \in \mathbb{R}^{N_{\mathrm{faces}} \times N^f_{\mathrm{geo}}}$.
* The i-th row corresponds to the i-th face.
* C^f_{ij} has the global index of the j-th local node in face i.

Mesh and data structures

Connectivity matrix for elements

* This is a matrix $C \in \mathbb{R}^{N_{\text{el}} \times N^{e}_{\text{geo}}}$.
* The i-th row corresponds to the i-th element.
* C_{ij} has the global index of the j-th local node in element i.
* Consider an example.

Connectivity matrix for faces

* This is a matrix $C^f \in \mathbb{R}^{N_{\text{faces}} \times N^{f}_{\text{geo}}}$.
* The i-th row corresponds to the i-th face.
* C^f_{ij} has the global index of the j-th local node in face i.
* Consider an example.

Local to global maps for elements

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\text{el}} \times N_h^e}$.

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\text{el}} \times N_h^e}$.
* If $p = 1$, then $C_h = C$.

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\text{el}} \times N_h^e}$.
* If $p = 1$, then $C_h = C$.
* The i-th row corresponds to the i-th element.

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\text{el}} \times N_h^e}$.
* If $p = 1$, then $C_h = C$.
* The i-th row corresponds to the i-th element.
* $C_{h,ij}$ has the global index of the j-th local DoF in element i.

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\text{el}} \times N_h^e}$.
* If $p = 1$, then $C_h = C$.
* The i-th row corresponds to the i-th element.
* $C_{h,ij}$ has the global index of the j-th local DoF in element i.
* Consider an example.

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\text{el}} \times N_h^e}$.
* If $p = 1$, then $C_h = C$.
* The i-th row corresponds to the i-th element.
* $C_{h,ij}$ has the global index of the j-th local DoF in element i.
* Consider an example.

Local to global maps for faces

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\text{el}} \times N_h^e}$.
* If $p = 1$, then $C_h = C$.
* The i-th row corresponds to the i-th element.
* $C_{h,ij}$ has the global index of the j-th local DoF in element i.
* Consider an example.

Local to global maps for faces

* This is a matrix (or a map) $C_h^f \in \mathbb{R}^{N_{\text{faces}} \times N_h^f}$.

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\mathrm{el}} \times N_h^e}$.
* If $p = 1$, then $C_h = C$.
* The i-th row corresponds to the i-th element.
* $C_{h,ij}$ has the global index of the j-th local DoF in element i.
* Consider an example.

Local to global maps for faces

* This is a matrix (or a map) $C_h^f \in \mathbb{R}^{N_{\mathrm{faces}} \times N_h^f}$.
* If $p = 1$, then $C_h^f = C^f$.

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\text{el}} \times N_h^e}$.
* If $p = 1$, then $C_h = C$.
* The i-th row corresponds to the i-th element.
* $C_{h,ij}$ has the global index of the j-th local DoF in element i.
* Consider an example.

Local to global maps for faces

* This is a matrix (or a map) $C_h^f \in \mathbb{R}^{N_{\text{faces}} \times N_h^f}$.
* If $p = 1$, then $C_h^f = C^f$.
* The i-th row corresponds to the i-th face.

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\text{el}} \times N_h^e}$.
* If $p = 1$, then $C_h = C$.
* The i-th row corresponds to the i-th element.
* $C_{h,ij}$ has the global index of the j-th local DoF in element i.
* Consider an example.

Local to global maps for faces

* This is a matrix (or a map) $C_h^f \in \mathbb{R}^{N_{\text{faces}} \times N_h^f}$.
* If $p = 1$, then $C_h^f = C^f$.
* The i-th row corresponds to the i-th face.
* $C_{h,ij}^f$ has the global index of the j-th local DoF in face i.

Mesh and data structures

Local to global maps for elements

* This is a matrix (or a map) $C_h \in \mathbb{R}^{N_{\text{el}} \times N_h^e}$.
* If $p = 1$, then $C_h = C$.
* The i-th row corresponds to the i-th element.
* $C_{h,ij}$ has the global index of the j-th local DoF in element i.
* Consider an example.

Local to global maps for faces

* This is a matrix (or a map) $C_h^f \in \mathbb{R}^{N_{\text{faces}} \times N_h^f}$.
* If $p = 1$, then $C_h^f = C^f$.
* The i-th row corresponds to the i-th face.
* $C_{h,ij}^f$ has the global index of the j-th local DoF in face i.
* Consider an example.

Mesh and data structures

How to use C or C_h to assemble the global operators?

Mesh and data structures

How to use C or C_h to assemble the global operators?

Within the loop on elements, proceed as follows:

Mesh and data structures

How to use C or C_h to assemble the global operators?

Within the loop on elements, proceed as follows:

* Compute the element based operators; e.g., F_i^e and M_{ij}^e.

Mesh and data structures

How to use C or C_h to assemble the global operators?

Within the loop on elements, proceed as follows:

* Compute the element based operators; e.g., F_i^e and M_{ij}^e.
* Loop on local DoF; i.e., for $i = 1$ to $i = N_h^e$.

Mesh and data structures

How to use C or C_h to assemble the global operators?

Within the loop on elements, proceed as follows:

* Compute the element based operators; e.g., F_i^e and M_{ij}^e.
* Loop on local DoF; i.e., for $i = 1$ to $i = N_h^e$.
* Assemble the global operators F and M as follows:

$$F_{i_g} = F_{i_g} + F_i^e,$$
$$M_{i_g,j_g} = M_{i_g,j_g} + M_{ij}^e,$$

where

$$i_g = C_h(e,i), \quad j_g = C_h(e,j),$$

are global indices.

The FE loop looks as follows:

for $K = 1$ to N_{el} **do**
  Compute quantities wrt reference element.
  for $q = 1$ to N_q **do**
    for $i = 1$ to DoFs per element **do**
      Compute element based vectors.
      for $j = 1$ to DoFs per element **do**
        Compute element based matrices.
      end for
    end for
    Assemble from local to global operators.
  end for
end for

Pseudocode to assemble from local to global operators

for $i = 1$ to N_h^e **do**
  $i_g = C_h(e, i)$
  $F_{i_g} = F_{i_g} + F_i^e$
  for $j = 1$ to N_h^e **do**
    $j_g = C_h(e, j)$
    $M_{i_g, j_g} = M_{i_g, j_g} + M_{ij}^e$
  end for
end for