An Application of the Non-conforming Crouzeix-Raviart Finite Element Method for Efficient Space Charge Calculations

C. Bahls, U. van Rienen
University of Rostock
Current and future accelerator design requires efficient 3D space charge calculations. These computations should be as efficient as possible.

One possible approach is Particle-in-Cell (PIC), especially the Particle-Mesh method which calculates the potential in the rest-frame of the bunch.

This computation usually is done by solving Poisson’s equation

\[-\Delta u(x) = f(x), \quad \forall x \in \Omega.\]

subject to some boundary conditions:

\[u(x) = g_D(x), \quad \forall x \in \partial \Omega_D,\]
\[\nabla u(x) \cdot n(x) = g_N(x), \quad \forall x \in \partial \Omega_N.\]
Current and future accelerator design requires efficient 3D space charge calculations. These computations should be as efficient as possible.

One possible approach is Particle-in-Cell (PIC), especially the Particle-Mesh method which calculates the potential in the rest-frame of the bunch.

This computation usually is done by solving Poisson’s equation

$$-\Delta u(x) = f(x), \quad \forall x \in \Omega.$$ 

subject to some boundary conditions:

$$u(x) = g_D(x), \quad \forall x \in \partial \Omega_D,$$

$$\nabla u(x) \cdot n(x) = g_N(x), \quad \forall x \in \partial \Omega_N.$$
• Current and future accelerator design requires efficient 3D space charge calculations. These computations should be as efficient as possible.

• One possible approach is Particle-in-Cell (PIC), especially the Particle-Mesh method which calculates the potential in the rest-frame of the bunch.

This computation usually is done by solving Poisson’s equation:

\[-\Delta u(x) = f(x), \quad \forall x \in \Omega.\]

subject to some boundary conditions:

\[u(x) = g_D(x), \quad \forall x \in \partial \Omega_D,\]
\[\nabla u(x) \cdot n(x) = g_N(x), \quad \forall x \in \partial \Omega_N.\]
Current and future accelerator design requires efficient 3D space charge calculations. These computations should be as efficient as possible.

One possible approach is Particle-in-Cell (PIC), especially the Particle-Mesh method which calculates the potential in the rest-frame of the bunch.

This computation usually is done by solving Poisson’s equation

\[-\Delta u(x) = f(x), \quad \forall x \in \Omega.\]

subject to some boundary conditions:

\[u(x) = g_D(x), \quad \forall x \in \partial \Omega_D,\]

\[\nabla u(x) \cdot n(x) = g_N(x), \quad \forall x \in \partial \Omega_N.\]
Current and future accelerator design requires efficient 3D space charge calculations. These computations should be as efficient as possible.

One possible approach is Particle-in-Cell (PIC), especially the Particle-Mesh method which calculates the potential in the rest-frame of the bunch.

This computation usually is done by solving Poisson’s equation

\[-\Delta u(x) = f(x), \quad \forall x \in \Omega.\]

subject to some boundary conditions:

\[u(x) = g_D(x), \quad \forall x \in \partial \Omega_D,\]
\[\nabla u(x) \cdot n(x) = g_N(x), \quad \forall x \in \partial \Omega_N.\]
Current and future accelerator design requires efficient 3D space charge calculations. These computations should be as efficient as possible.

One possible approach is Particle-in-Cell (PIC), especially the Particle-Mesh method which calculates the potential in the rest-frame of the bunch.

This computation usually is done by solving Poisson’s equation

\[-\Delta u(x) = f(x), \quad \forall x \in \Omega.\]

subject to some boundary conditions:

\[u(x) = g_D(x), \quad \forall x \in \partial\Omega_D,\]
\[\nabla u(x) \cdot n(x) = g_N(x), \quad \forall x \in \partial\Omega_N.\]
- We are aiming at computing the self-field of the bunch.
- So we are estimating a solution to Gauss’ Law:

\[
\text{div } D = \rho,
\]

where \( D \) denotes the dielectric flux and \( \rho \) the charge density.
- There are infinitely many solutions to that equation. (a very large subspace of all vectorial functions)
- One can add any divergence-free (curl) field to a solution without changing the divergence of the field (the divergence of a curl is zero).
• We are aiming at computing the self-field of the bunch.
• So we are estimating a solution to Gauss’ Law:

\[ \text{div } \mathbf{D} = \rho, \]

where \( \mathbf{D} \) denotes the dielectric flux and \( \rho \) the charge density.

• There are infinitely many solutions to that equation. (a very large subspace of all vectorial functions)
• One can add any divergence-free (curl) field to a solution without changing the divergence of the field (the divergence of a curl is zero).
We are aiming at computing the self-field of the bunch.

So we are estimating a solution to Gauss’ Law:

$$\text{div } D = \rho,$$

where $D$ denotes the dielectric flux and $\rho$ the charge density

There are infinitely many solutions to that equation. (a very large subspace of all vectorial functions)

One can add any divergence-free (curl) field to a solution without changing the divergence of the field (the divergence of a curl is zero).
• We are aiming at computing the self-field of the bunch.
• So we are estimating a solution to Gauss’ Law:
\[ \text{div } D = \rho, \]
where $D$ denotes the dielectric flux and $\rho$ the charge density.

• There are infinitely many solutions to that equation. (a very large subspace of all vectorial functions)
• One can add any divergence-free (curl) field to a solution without changing the divergence of the field (the divergence of a curl is zero).
We are aiming at computing the self-field of the bunch.

So we are estimating a solution to Gauss’ Law:

$$\text{div } D = \rho,$$

where $D$ denotes the dielectric flux and $\rho$ the charge density

There are infinitely many solutions to that equation. (a very large subspace of all vectorial functions)

One can add any divergence-free (curl) field to a solution without changing the divergence of the field (the divergence of a curl is zero).
We are only interested in curl-free solutions of Gauss’ law

For this we will use fields which are gradients of a scalar function.

\[ \Psi = -\nabla u. \]

Then our equations become

\[
\begin{align*}
\nabla u(x) + \Psi(x) &= 0 \\
\operatorname{div} \varepsilon(x) \Psi(x) &= \rho(x),
\end{align*}
\]

Which usually get shortened to:

\[ -\operatorname{div} \varepsilon(x) \nabla u(x) = \rho(x). \]

Or if \( \varepsilon(x) \) is isotropic or constant:

\[ -\Delta u(x) = \varepsilon^{-1} \rho(x). \]
• We are only interested in curl-free solutions of Gauss’ law
• For this we will use fields which are gradients of a scalar function.
\[ \Psi = -\nabla u. \]

Then our equations become
\[
\begin{align*}
\nabla u(x) + \Psi(x) &= 0 \\
\text{div} \, \varepsilon(x) \Psi(x) &= \rho(x),
\end{align*}
\]

Which usually get shortened to:
\[
-\text{div} \, \varepsilon(x) \nabla u(x) = \rho(x).
\]

Or if \( \varepsilon(x) \) is isotropic or constant: 
\[
-\Delta u(x) = \varepsilon^{-1} \rho(x).
\]
We are only interested in curl-free solutions of Gauss’ law.

For this we will use fields which are gradients of a scalar function.

\[ \Psi = -\nabla u. \]

Then our equations become

\[
\nabla u(x) + \Psi(x) = 0 \\
\n\text{div} \ \varepsilon(x)\Psi(x) = \rho(x), 
\]

Which usually get shortened to:

\[-\text{div} \ \varepsilon(x) \ \nabla u(x) = \rho(x).\]

Or if \( \varepsilon(x) \) is isotropic or constant: \(-\Delta u(x) = \varepsilon^{-1} \rho(x).\)
We are only interested in curl-free solutions of Gauss’ law

For this we will use fields which are gradients of a scalar function.

\[ \Psi = -\nabla u. \]

Then our equations become

\[
\begin{align*}
\nabla u(x) + \Psi(x) &= 0 \\
\text{div } \varepsilon(x) \Psi(x) &= \rho(x),
\end{align*}
\]

Which usually get shortened to:

\[-\text{div } \varepsilon(x) \nabla u(x) = \rho(x).\]

Or if \(\varepsilon(x)\) is isotropic or constant:

\[-\Delta u(x) = \varepsilon^{-1} \rho(x).\]
• We are only interested in curl-free solutions of Gauss’ law

• For this we will use fields which are gradients of a scalar function.

\[ \Psi = -\nabla u. \]

• Then our equations become

\[
\begin{align*}
\nabla u(x) + \Psi(x) & = 0 \\
\operatorname{div} \varepsilon(x) \Psi(x) & = \rho(x),
\end{align*}
\]

• Which usually get shortened to:

\[ -\operatorname{div} \varepsilon(x) \nabla u(x) = \rho(x). \]

• Or if \( \varepsilon(x) \) is isotropic or constant: 

\[ -\Delta u(x) = \varepsilon^{-1} \rho(x). \]
• Our currently used numerical scheme (solving \(-\Delta u(x) = \rho(x)/\varepsilon_0\)) using a Finite Difference approximation of the Laplace operator \(\Delta\) is suboptimal for estimating the electric field.

• We are loosing one order of convergence \((O(h^2) \rightarrow O(h^1))\) by having to compute the gradient from the potential.

• The discretized solution \(u_h\) on an equidistant structured mesh approximates the solution \(u\) with an order of \(O(h^2)\):

\[ u_h(x) = u(x) + O(h^2). \]

• The gradient \(\Psi\) will then be approximated with an order of \(O(h^1)\):

\[ \Psi_h(x) = \Psi(x) + O(h^1). \]
Our currently used numerical scheme (solving $-\Delta u(x) = \rho(x)/\varepsilon_0$) using a Finite Difference approximation of the Laplace operator $\Delta$ is suboptimal for estimating the electric field.

We are loosing one order of convergence ($O(h^2) \rightarrow O(h^1)$) by having to compute the gradient from the potential.

The discretized solution $u_h$ on an equidistant structured mesh approximates the solution $u$ with an order of $O(h^2)$:

$$u_h(x) = u(x) + O(h^2).$$

The gradient $\psi$ will than be approximated with an order of $O(h^1)$:

$$\psi_h(x) = \psi(x) + O(h^1).$$
• Our currently used numerical scheme (solving $-\Delta u(x) = \rho(x)/\varepsilon_0$) using a Finite Difference approximation of the Laplace operator $\Delta$ is suboptimal for estimating the electric field.

• We are losing one order of convergence ($O(h^2) \to O(h^1)$) by having to compute the gradient from the potential.

• The discretized solution $u_h$ on an equidistant structured mesh approximates the solution $u$ with an order of $O(h^2)$:

$$u_h(x) = u(x) + O(h^2).$$

• The gradient $\Psi$ will then be approximated with an order of $O(h^1)$:

$$\Psi_h(x) = \Psi(x) + O(h^1).$$
• Our currently used numerical scheme (solving \(-\Delta u(x) = \rho(x)/\varepsilon_0\)) using a Finite Difference approximation of the Laplace operator \(\Delta\) is suboptimal for estimating the electric field.

• We are loosing one order of convergence \((O(h^2) \to O(h^1))\) by having to compute the gradient from the potential.

• The discretized solution \(u_h\) on an equidistant structured mesh approximates the solution \(u\) with an order of \(O(h^2)\):

\[
u_h(x) = u(x) + O(h^2).
\]

• The gradient \(\Psi\) will than be approximated with an order of \(O(h^1)\):

\[
\Psi_h(x) = \Psi(x) + O(h^1).
\]
So let's ask: What do we actually need for our computations?
⇒ The electric field $E$, as it accelerates the charged particles.

What has the Poisson-Equation originally been derived from?
⇒ Gauss' law $\text{div} \, D = \rho$, plus some Gauging

In our Problem setting the potential seems somewhat arbitrary - it could be calculated as an integrated field strength from the boundary of the domain.

So instead we want to discretize and solve for the vector field directly.

The Discretization used has to be curl-free and should somehow allow for a sane definition of the Divergence of the field (e.g. be conformal)
So let's ask: What do we actually need for our computations?

⇒ The electric field $E$, as it accelerates the charged particles.

What has the Poisson-Equation originally been derived from?

⇒ Gauss' law $\text{div} \, D = \rho$, plus some Gauging

In our problem setting the potential seems somewhat arbitrary - it could be calculated as an integrated field strength from the boundary of the domain.

So instead we want to discretize and solve for the vector field directly.

The discretization used has to be curl-free and should somehow allow for a sane definition of the divergence of the field (e.g. be conformal).
So let's ask: What do we actually need for our computations?
⇒ The electric field $E$, as it accelerates the charged particles.

What has the Poisson-Equation originally been derived from?
⇒ Gauss' law $\text{div} \, D = \rho$, plus some Gauging

In our Problem setting the potential seems somewhat arbitrary - it could be calculated as an integrated field strength from the boundary of the domain.

So instead we want to discretize and solve for the vector field directly.

The Discretization used has to be curl-free and should somehow allow for a sane definition of the Divergence of the field (e.g. be conformal).
• So let's ask: What do we actually need for our computations?
  ⇒ The electric field $\mathbf{E}$, as it accelerates the charged particles.

• What has the Poisson-Equation originally been derived from?
  ⇒ Gauss' law $\text{div } \mathbf{D} = \rho$, plus some Gauging

• In our Problem setting the potential seems somewhat arbitrary - it could be calculated as an integrated field strength from the boundary of the domain.

• So instead we want to discretize and solve for the vector field directly.

• The Discretization used has to be curl-free and should somehow allow for a sane definition of the Divergence of the field (e.g. be conformal).
So let's ask: What do we actually need for our computations?
⇒ The electric field \( E \), as it accelerates the charged particles.

What has the Poisson-Equation originally been derived from?
⇒ Gauss' law \( \text{div } D = \rho \), plus some Gauging.

In our Problem setting the potential seems somewhat arbitrary - it could be calculated as an integrated field strength from the boundary of the domain.

So instead we want to discretize and solve for the vector field directly.

The Discretization used has to be curl-free and should somehow allow for a sane definition of the Divergence of the field (e.g. be conformal).
So let's ask: What do we actually need for our computations?
⇒ The electric field $E$, as it accelerates the charged particles.

What has the Poisson-Equation originally been derived from?
⇒ Gauss’ law $\text{div} \ D = \rho$, plus some Gauging

In our Problem setting the potential seems somewhat arbitrary - it could be calculated as an integrated field strength from the boundary of the domain.

So instead we want to discretize and solve for the vector field directly.

The Discretization used has to be curl-free and should somehow allow for a sane definition of the Divergence of the field (e.g. be conformal)
So let's ask: What do we actually need for our computations?  
⇒ The electric field $E$, as it accelerates the charged particles.

What has the Poisson-Equation originally been derived from?  
⇒ Gauss’ law $\text{div } D = \rho$, plus some Gauging

In our Problem setting the potential seems somewhat arbitrary - it could be calculated as an integrated field strength from the boundary of the domain.

So instead we want to discretize and solve for the vector field directly.

The Discretization used has to be curl-free and should somehow allow for a sane definition of the Divergence of the field (e.g. be conformal).
So let's ask: What do we actually need for our computations?
⇒ The electric field $E$, as it accelerates the charged particles.

What has the Poisson-Equation originally been derived from?
⇒ Gauss' law $\text{div } D = \rho$, plus some Gauging

In our Problem setting the potential seems somewhat arbitrary - it could be calculated as an integrated field strength from the boundary of the domain.

So instead we want to discretize and solve for the vector field directly.

The Discretization used has to be curl-free and should somehow allow for a sane definition of the Divergence of the field (e.g. be conformal).
So let's ask: What do we actually need for our computations?
⇒ The electric field $E$, as it accelerates the charged particles.

What has the Poisson-Equation originally been derived from?
⇒ Gauss' law $\text{div} \ D = \rho$, plus some Gauging

In our Problem setting the potential seems somewhat arbitrary - it could be calculated as an integrated field strength from the boundary of the domain.

So instead we want to discretize and solve for the vector field directly

The Discretization used has to be curl-free and should somehow allow for a sane definition of the Divergence of the field (e.g. be conformal)
So let's ask: What do we actually need for our computations?
⇒ The electric field \( \mathbf{E} \), as it accelerates the charged particles.

What has the Poisson-Equation originally been derived from?
⇒ Gauss' law \( \text{div} \mathbf{D} = \rho \), plus some Gauging

In our Problem setting the potential seems somewhat arbitrary - it could be calculated as an integrated field strength from the boundary of the domain.

So instead we want to discretize and solve for the vector field directly.

The Discretization used has to be curl-free and should somehow allow for a sane definition of the Divergence of the field (e.g. be conformal)
One suitable ansatz space is $RT_0$ the Raviart-Thomas space of lowest order, whose vector functions have following element-wise linear expression:

$$\Psi_h(x) = a_k + b_k x,$$

(where $x$ is in the element $T_k$ of the triangulation $T$ of the domain $\Omega$)

For the discretisation to be conformal the normal Components of the Field have to be continuous at every inner Interface (Edges in 2D, Faces in 3D).

So $RT_0$ usually is represented by an Edge/Face-based discretization (as shown in the next few frames) using following local representation:

$$\psi_{E_j}(x) = \sigma_j \frac{|E_j|}{2|T_k|} (x - P_j).$$

$|T_k|$ is the area/volume of $T_k$, $|E_j|$ is the length/area of the edge/face $E_j$, $\sigma_j$ indicates the orientation of $E_j$ and $P_j$ is the Vertex opposite to $E_j$. 
One suitable ansatz space is $\mathbf{RT}_0$, the Raviart-Thomas space of lowest order, whose vector functions have following element-wise linear expression:

$$\psi_h(x) = a_k + b_k x,$$

(where $x$ is in the element $T_k$ of the triangulation $T$ of the domain $\Omega$)

For the discretisation to be conformal the normal Components of the Field have to be continuous at every inner Interface (Edges in 2D, Faces in 3D).

So $\mathbf{RT}_0$ usually is represented by an Edge/Face-based discretization (as shown in the next few frames) using following local representation:

$$\psi_{E_j}(x) = \sigma_j \frac{|E_j|}{2|T_k|} (x - P_j).$$

$|T_k|$ is the area/volume of $T_k$, $|E_j|$ is the length/area of the edge/face $E_j$, $\sigma_j$ indicates the orientation of $E_j$ and $P_j$ is the Vertex opposite to $E_j$. 
One suitable ansatz space is $\mathbf{RT}_0$ the Raviart-Thomas space of lowest order, whose vector functions have following element-wise linear expression:

$$\Psi_h(x) = a_k + b_k x,$$

(\text{where } x \text{ is in the element } T_k \text{ of the triangulation } \mathbf{T} \text{ of the domain } \Omega)

For the discretisation to be conformal the normal Components of the Field have to be continuous at every inner Interface (Edges in 2D, Faces in 3D).

So $\mathbf{RT}_0$ usually is represented by an Edge/Face-based discretization (as shown in the next few frames) using following local representation:

$$\psi_{E_j}(x) = \sigma_j \frac{|E_j|}{2 |T_k|} (x - P_j).$$

$|T_k|$ is the area/volume of $T_k$, $|E_j|$ is the length/area of the edge/face $E_j$, $\sigma_j$ indicates the orientation of $E_j$ and $P_j$ is the Vertex opposite to $E_j$. 
One suitable ansatz space is \( \textbf{RT}_0 \) the Raviart-Thomas space of lowest order, whose vector functions have following element-wise linear expression:

\[
\Psi_h(x) = a_k + b_k x,
\]

(where \( x \) is in the element \( T_k \) of the triangulation \( T \) of the domain \( \Omega \))

For the discretisation to be conformal the normal Components of the Field have to be continuous at every inner Interface (Edges in 2D, Faces in 3D).

So \( \textbf{RT}_0 \) usually is represented by an Edge/Face-based discretization (as shown in the next few frames) using following local representation:

\[
\psi_{E_j}(x) = \sigma_j \frac{|E_j|}{2|T_k|} (x - P_j).
\]

\( |T_k| \) is the area/volume of \( T_k \), \( |E_j| \) is the length/area of the edge/face \( E_j \), \( \sigma_j \) indicates the orientation of \( E_j \) and \( P_j \) is the Vertex opposite to \( E_j \).
One suitable ansatz space is $\text{RT}_0$, the Raviart-Thomas space of lowest order, whose vector functions have following element-wise linear expression:

$$\Psi_h(x) = a_k + b_k x,$$

(Where $x$ is in the element $T_k$ of the triangulation $T$ of the domain $\Omega$)

For the discretisation to be conformal the normal Components of the Field have to be continuous at every inner Interface (Edges in 2D, Faces in 3D).

So $\text{RT}_0$ usually is represented by an Edge/Face-based discretization (as shown in the next few frames) using following local representation:

$$\psi_{E_j}(x) = \sigma_j \frac{|E_j|}{2|T_k|} (x - P_j).$$

$|T_k|$ is the area/volume of $T_k$, $|E_j|$ is the length/area of the edge/face $E_j$, $\sigma_j$ indicates the orientation of $E_j$ and $P_j$ is the Vertex opposite to $E_j$. 
One suitable ansatz space is $\mathbf{RT}_0$ the Raviart-Thomas space of lowest order, whose vector functions have following element-wise linear expression:

$$\Psi_h(x) = a_k + b_k x,$$

(where $x$ is in the element $T_k$ of the triangulation $T$ of the domain $\Omega$)

For the discretisation to be conformal the normal Components of the Field have to be continuous at every inner Interface (Edges in 2D, Faces in 3D).

So $\mathbf{RT}_0$ usually is represented by an Edge/Face-based discretization (as shown in the next few frames) using following local representation:

$$\psi_{E_j}(x) = \sigma_j \frac{|E_j|}{2|T_k|}(x - P_j).$$

$|T_k|$ is the area/volume of $T_k$, $|E_j|$ is the length/area of the edge/face $E_j$, $\sigma_j$ indicates the orientation of $E_j$ and $P_j$ is the Vertex opposite to $E_j$. 
Raviart-Thomas Ansatz-Space – Basis-Element $\psi_{E_1}$
Raviart-Thomas Ansatz-Space – Basis-Element $\psi_{E_3}$
We are now using the canonical Galerkin approach for Mixed Finite Elements to compute approximate solutions for the field $\Psi_h$ and the potential $u_h$:

$$
\int_\Omega \tau \cdot \Psi_h + \int_\Omega \tau \cdot \varepsilon \text{grad} \ u_h = 0 \quad \forall \tau \in RT_0
$$

$$
\int_\Omega \nu \ \text{div} \ \Psi_h = \int_\Omega \nu \ f \quad \forall \nu \in P_{1}^{-1}
$$

To later remove the flux variable from the system, we will relax the continuity requirement on the Ansatz-space and use the flux $\tilde{\Psi}_h$ from the space $RT_0^{-1}$ of discontinuous linear vector functions.
We are now using the canonical Galerkin approach for Mixed Finite Elements to compute approximate solutions for the field $\Psi_h$ and the potential $u_h$:

$$\int_\Omega \tau \cdot \Psi_h + \int_\Omega \tau \cdot \text{grad} \, u_h = 0 \quad \forall \tau \in \mathbf{RT}_0$$

$$\int_\Omega \nu \, \text{div} \, \Psi_h = \int_\Omega \nu \, f \quad \forall \nu \in \mathbf{P}_1^{-1}$$

To later remove the flux variable from the system, we will relax the continuity requirement on the Ansatz-space and use the flux $\tilde{\Psi}_h$ from the space $\mathbf{RT}_0^{-1}$ of discontinuous linear vector functions.
We are now using the canonical Galerkin approach for Mixed Finite Elements to compute approximate solutions for the field $\Psi_h$ and the potential $u_h$:

\[
\int_{\Omega} \tau \cdot \Psi_h + \int_{\Omega} \tau \varepsilon \text{grad} u_h = 0 \quad \forall \tau \in RT_0
\]

\[
\int_{\Omega} v \text{div} \Psi_h = \int_{\Omega} v f \quad \forall v \in P^{-1}_1
\]

To later remove the flux variable from the system, we will relax the continuity requirement on the Ansatz-space and use the flux $\tilde{\Psi}_h$ from the space $RT_0^{-1}$ of discontinuous linear vector functions.
We are now using the canonical Galerkin approach for Mixed Finite Elements to compute approximate solutions for the field $\Psi_h$ and the potential $u_h$:

$$
\int_{\Omega} \tau \cdot \Psi_h + \int_{\Omega} \tau \cdot \nabla u_h = 0 \quad \forall \tau \in RT_0
$$

$$
\int_{\Omega} v \, \text{div} \, \Psi_h = \int_{\Omega} v \, f \quad \forall v \in P_{1}^{-1}
$$

To later remove the flux variable from the system, we will relax the continuity requirement on the Ansatz-space and use the flux $\tilde{\Psi}_h$ from the space $RT_0^{-1}$ of discontinuous linear vector functions.
We then enforce the continuity of the normal component of the flux on faces by the use of Lagrange multipliers $\lambda_h \in M_1^{-1}$, leading to the system:

\[
\begin{align*}
\int_{\Omega} \tilde{\tau} \cdot \tilde{\Psi}_h + \int_{\Omega} \tilde{\tau} \varepsilon \text{grad } u_h + \int_{\delta\Omega} \lambda_h \mathbf{n}_T \cdot \tilde{\tau} &= 0 \quad \forall \tilde{\tau} \in RT_0^{-1} \\
\int_{\Omega} v \text{ div } \tilde{\Psi}_h &= \int_{\Omega} v f \quad \forall v \in P_1^{-1} \\
\int_{\delta\Omega} \mu \mathbf{n}_T \cdot \tilde{\Psi}_h &= 0 \quad \forall \mu \in M_1^{-1}
\end{align*}
\]

leading to following linear system of equations:

\[
\begin{pmatrix}
A & B & C \\
B^T & & \\
C^T & & \\
\end{pmatrix}
\begin{pmatrix}
\tilde{\Psi}_h \\
u_h \\
\lambda_h
\end{pmatrix}
= 
\begin{pmatrix}
0 \\
f_h \\
0
\end{pmatrix}
\]
We then enforce the continuity of the normal component of the flux on faces by the use of Lagrange multipliers $\lambda_h \in M_1^{-1}$, leading to the system:

\[
\begin{align*}
\int_{\Omega} \tilde{\tau} \cdot \tilde{\Psi}_h + \int_{\Omega} \tilde{\tau} \in \text{grad } u_h + \int_{\delta\Omega} \lambda_h \mathbf{n}_T \cdot \tilde{\tau} &= 0 \quad \forall \tilde{\tau} \in RT_0^{-1} \\
\int_{\Omega} \nu \text{ div } \tilde{\Psi}_h &= \int_{\Omega} \nu f \quad \forall \nu \in P_1^{-1} \\
\int_{\delta\Omega} \mu \mathbf{n}_T \cdot \tilde{\Psi}_h &= 0 \quad \forall \mu \in M_1^{-1}
\end{align*}
\]

leading to following linear system of equations:

\[
\begin{pmatrix}
A & B & C \\
B^T & C^T
\end{pmatrix}
\begin{pmatrix}
\tilde{\Psi}_h \\
u_h \\
\lambda_h
\end{pmatrix}
= 
\begin{pmatrix}
0 \\
f_h \\
0
\end{pmatrix}
\]
The Submatrix $A$ is block-diagonal, so it is easily element-wise invertable, so the flux $\tilde{\Psi}_h$ can be computed by:

$$\tilde{\Psi}_h = -A^{-1}(B u_h + C \lambda_h)$$

leading to following linear system of equations:

$$
\begin{pmatrix}
B^T A^{-1} B & B^T A^{-1} C \\
C^T A^{-1} B & C^T A^{-1} C
\end{pmatrix}
\begin{pmatrix}
u_h \\
\lambda_h
\end{pmatrix}
=
\begin{pmatrix}
-f_h \\
0
\end{pmatrix}
$$

This method is called static condensation.

Eliminating the $u_h$ using a Schur complement we would arrive at a variant of the Crouzeix-Raviart Finite Element Method $\Rightarrow$ which can also be derived directly.
The Submatrix $A$ is block-diagonal, so it is easily element-wise invertable, so the flux $\tilde{\Psi}_h$ can be computed by:

$$
\tilde{\Psi}_h = -A^{-1}(B \ u_h + C \lambda_h)
$$

leading to following linear system of equations:

$$
\begin{pmatrix}
B^T A^{-1} B & B^T A^{-1} C \\
C^T A^{-1} B & C^T A^{-1} C
\end{pmatrix}
\begin{pmatrix}
u_h \\
\lambda_h
\end{pmatrix} =
\begin{pmatrix}
-f_h \\
0
\end{pmatrix}
$$

This method is called static condensation.

Eliminating the $u_h$ using a Schur complement we would arrive at a variant of the Crouzeix-Raviart Finite Element Method which can also be derived directly.
The Submatrix $A$ is block-diagonal, so it is easily element-wise invertable, so the flux $\tilde{\Psi}_h$ can be computed by:

$$\tilde{\Psi}_h = -A^{-1}(Bu_h + C\lambda_h)$$

leading to following linear system of equations:

$$\begin{pmatrix}
B^T A^{-1} B & B^T A^{-1} C \\
C^T A^{-1} B & C^T A^{-1} C
\end{pmatrix}
\begin{pmatrix}
u_h \\
\lambda_h
\end{pmatrix} = \begin{pmatrix}
-f_h \\
0
\end{pmatrix}$$

This method is called static condensation.

Eliminating the $u_h$ using a Schur complement we would arrive at a variant of the Crouzeix-Raviart Finite Element Method $\Rightarrow$ which can also be derived directly.
The Submatrix $A$ is block-diagonal, so it is easily element-wise invertable, so the flux $\tilde{\Psi}_h$ can be computed by:

$$
\tilde{\Psi}_h = -A^{-1}(B u_h + C \lambda_h)
$$

leading to following linear system of equations:

$$
\begin{pmatrix}
B^T A^{-1} B & B^T A^{-1} C \\
C^T A^{-1} B & C^T A^{-1} C
\end{pmatrix}
\begin{pmatrix}
u_h \\
\lambda_h
\end{pmatrix}
= 
\begin{pmatrix}
-f_h \\
0
\end{pmatrix}
$$

This method is called static condensation.

Eliminating the $u_h$ using a Schur complement we would arrive at a variant of the Crouzeix-Raviart Finite Element Method which can also be derived directly.
The Submatrix $A$ is block-diagonal, so it is easily element-wise invertable, so the flux $\tilde{\Psi}_h$ can be computed by:

$$\tilde{\Psi}_h = -A^{-1}(B u_h + C \lambda_h)$$

leading to following linear system of equations:

$$\begin{pmatrix} B^T A^{-1} B & B^T A^{-1} C \\ C^T A^{-1} B & C^T A^{-1} C \end{pmatrix} \begin{pmatrix} u_h \\ \lambda_h \end{pmatrix} = \begin{pmatrix} -f_h \\ 0 \end{pmatrix}$$

This method is called static condensation.

Eliminating the $u_h$ using a Schur complement we would arrive at a variant of the Crouzeix-Raviart Finite Element Method which can also be derived directly.
The Submatrix $A$ is block-diagonal, so it is easily element-wise invertable, so the flux $\tilde{\Psi}_h$ can be computed by:

$$\tilde{\Psi}_h = -A^{-1}(B u_h + C \lambda_h)$$

leading to following linear system of equations:

$$
\begin{pmatrix}
B^T A^{-1} B & B^T A^{-1} C \\
C^T A^{-1} B & C^T A^{-1} C
\end{pmatrix}
\begin{pmatrix}
u_h \\
\lambda_h
\end{pmatrix}
=
\begin{pmatrix}
-f_h \\
0
\end{pmatrix}
$$

This method is called static condensation.

Eliminating the $u_h$ using a Schur complement we would arrive at a variant of the Crouzeix-Raviart Finite Element Method $\Rightarrow$ which can also be derived directly.
The Submatrix $A$ is block-diagonal, so it is easily element-wise invertable, so the flux $\tilde{\Psi}_h$ can be computed by:

$$\tilde{\Psi}_h = -A^{-1}(B u_h + C \lambda_h)$$

leading to following linear system of equations:

$$
\begin{pmatrix}
B^T A^{-1} B & B^T A^{-1} C \\
C^T A^{-1} B & C^T A^{-1} C
\end{pmatrix}
\begin{pmatrix}
u_h \\
\lambda_h
\end{pmatrix}
= 
\begin{pmatrix}
f_h \\
0
\end{pmatrix}
$$

This method is called static condensation.

Eliminating the $u_h$ using a Schur complement we would arrive at a variant of the Crouzeix-Raviart Finite Element Method which can also be derived directly.
In “An inexpensive Method for the Evaluation of the Solution of the lowest order Raviart-Thomas Mixed Method” Marini suggested to use the $P_1$ nonconforming finite element spaces $P_{NC}^1$.

This $P_{NC}^1$ are also called Crouzeix-Raviart or loof finite element spaces.

These finite Elements have their degrees of freedom allocated to the barycenters of their Edges/Faces, rather than their Vertices.

The function space $P_{NC}^1$ actually contains the space $P_1$ of linear functions.

So $P_{NC}^1$ can at least represent the continuous solutions from nodal $P_1$-FEM.

As the Ansatz space is larger there is room for additional contraints, e.g. for choosing a solution with a higher order approximation of the field.
• In “An inexpensive Method for the Evaluation of the Solution of the lowest order Raviart-Thomas Mixed Method” Marini suggested to use the $P_1$ nonconforming finite element spaces $P_{1NC}$.

• This $P_{1NC}$ are also called Crouzeix-Raviart or loof finite element spaces.

• These finite Elements have their degrees of freedom allocated to the barycenters of their Edges/Faces, rather than their Vertices.

• The function space $P_{1NC}$ actually contains the space $P_1$ of linear functions.

• So $P_{1NC}$ can at least represent the continuous solutions from nodal $P_1$-FEM.

• As the Ansatz space is larger there is room for additional contraints, e.g. for choosing a solution with a higher order approximation of the field.
• In “An inexpensive Method for the Evaluation of the Solution of the lowest order Raviart-Thomas Mixed Method” Marini suggested to use the $P_1^{\text{NC}}$ nonconforming finite element spaces $P_1^{\text{NC}}$.

• This $P_1^{\text{NC}}$ are also called Crouzeix-Raviart or loof finite element spaces.

• These finite Elements have their degrees of freedom allocated to the barycenters of their Edges/Faces, rather than their Vertices.

• The function space $P_1^{\text{NC}}$ actually contains the space $P_1$ of linear functions.

• So $P_1^{\text{NC}}$ can at least represent the continuous solutions from nodal $P_1$-FEM.

• As the Ansatz space is larger there is room for additional contraints, e.g. for choosing a solution with a higher order approximation of the field.
In “An inexpensive Method for the Evaluation of the Solution of the lowest order Raviart-Thomas Mixed Method” Marini suggested to use the $P_1^{\text{NC}}$ nonconforming finite element spaces.

This $P_1^{\text{NC}}$ are also called Crouzeix-Raviart or loof finite element spaces.

These finite Elements have their degrees of freedom allocated to the barycenters of their Edges/Faces, rather than their Vertices.

The function space $P_1^{\text{NC}}$ actually contains the space $P_1$ of linear functions.

So $P_1^{\text{NC}}$ can at least represent the continuous solutions from nodal $P_1$-FEM.

As the Ansatz space is larger there is room for additional constraints, e.g. for choosing a solution with a higher order approximation of the field.
• In “An inexpensive Method for the Evaluation of the Solution of the lowest order Raviart-Thomas Mixed Method” Marini suggested to use the $P_1$ nonconforming finite element spaces $P_{1}^{\text{NC}}$.

• This $P_{1}^{\text{NC}}$ are also called Crouzeix-Raviart or loof finite element spaces.

• These finite Elements have their degrees of freedom allocated to the barycenters of their Edges/Faces, rather than their Vertices.

• The function space $P_{1}^{\text{NC}}$ actually contains the space $P_1$ of linear functions.

• So $P_{1}^{\text{NC}}$ can at least represent the continuous solutions from nodal $P_1$-FEM.

• As the Ansatz space is larger there is room for additional contraints, e.g. for choosing a solution with a higher order approximation of the field.
In “An inexpensive Method for the Evaluation of the Solution of the lowest order Raviart-Thomas Mixed Method” Marini suggested to use the $P_1$ nonconforming finite element spaces $P_{NC}^1$.

This $P_{NC}^1$ are also called Crouzeix-Raviart or loof finite element spaces.

These finite Elements have their degrees of freedom allocated to the barycenters of their Edges/Faces, rather than their Vertices.

The function space $P_{NC}^1$ actually contains the space $P_1$ of linear functions.

So $P_{NC}^1$ can at least represent the continuous solutions from nodal $P_1$-FEM.

As the Ansatz space is larger there is room for additional contraints, e.g. for choosing a solution with a higher order approximation of the field.
• In “An inexpensive Method for the Evaluation of the Solution of the lowest order Raviart-Thomas Mixed Method” Marini suggested to use the $P_1$ nonconforming finite element spaces $P_{1NC}$.

• This $P_{1NC}$ are also called Crouzeix-Raviart or loof finite element spaces.

• These finite Elements have their degrees of freedom allocated to the barycenters of their Edges/Faces, rather than their Vertices.

• The function space $P_{1NC}$ actually contains the space $P_1$ of linear functions.

• So $P_{1NC}$ can at least represent the continuous solutions from nodal $P_1$-FEM.

• As the Ansatz space is larger there is room for additional constraints, e.g. for choosing a solution with a higher order approximation of the field.
Figure: The ansatz-functions are only continuous at the midpoints of interfaces
The space $P_1$ of piecewise linear and continuous functions is contained in $P_1^{NC}$.

**Figure:** The space $P_1$ of piecewise linear and continuous functions is contained in $P_1^{NC}$. 
Figure: The space $P_{1}^{NC}$ also contains discontinuous functions.
Another (more straightforward) way to arrive at the Crouzeix-Raviart FEM is to apply the Galerkin approach to the nonconforming Ansatz space $P_{1}^{NC}$ directly:

$$\sum_{T_k \in T} \int_{T_k} \varepsilon \ \text{grad} \ u_h \cdot \text{grad} \ v = \int_{\Omega} f_h v \quad \forall v \in P_{1}^{NC}.$$

With decreasing mesh-size the num. solution $u_h$ converges to $u$ with $O(h^2)$.

More interestingly using a special post-processing we can recover a flux $\Psi_h$ of second order accuracy $O(h^2)$ using:

$$\Psi_h(x) = \varepsilon \ \text{grad} \ u_h - \frac{f_{T_k}}{n} (x - x_{T_k}), \quad x \in T_k, \ x_{T_k} \text{ barycenter of } T_k.$$

The normal component of $\Psi_h$ is continuous at inter-element interfaces.
Another (more straightforward) way to arrive at the Crouzeix-Raviart FEM is to apply the Galerkin approach to the nonconforming Ansatz space $P_{1}^{NC}$ directly:

$$\sum_{T_{k} \in T} \int_{T_{k}} \varepsilon \text{grad } u_{h} \cdot \text{grad } v = \int_{\Omega} f_{h} v \quad \forall v \in P_{1}^{NC}. $$

With decreasing mesh-size the num. solution $u_{h}$ converges to $u$ with $O(h^2)$.

More interestingly using a special post-processing we can recover a flux $\Psi_{h}$ of second order accuracy $O(h^2)$ using:

$$\Psi_{h}(x) = \varepsilon \text{grad } u_{h} - \frac{f_{T_{k}}}{n}(x - x_{T_{k}}), \quad x \in T_{k}, \ x_{T_{k}} \text{ barycenter of } T_{k}. $$

The normal component of $\Psi_{h}$ is continuous at inter-element interfaces.
Another (more straightforward) way to arrive at the Crouzeix-Raviart FEM is to apply the Galerkin approach to the nonconforming Ansatz space $P_{NC}^1$ directly:

$$\sum_{T_k \in T} \int_{T_k} \varepsilon \text{grad } u_h \cdot \text{grad } v = \int_{\Omega} f_h v \quad \forall v \in P_{NC}^1.$$

With decreasing mesh-size the num. solution $u_h$ converges to $u$ with $O(h^2)$.

More interestingly using a special post-processing we can recover a flux $\Psi_h$ of second order accuracy $O(h^2)$ using:

$$\Psi_h(x) = \varepsilon \text{grad } u_h - \frac{f_{T_k}}{n}(x - x_{T_k}), \quad x \in T_k, \ x_{T_k} \text{ barycenter of } T_k.$$

The normal component of $\Psi_h$ is continuous at inter-element interfaces.
Another (more straightforward) way to arrive at the Crouzeix-Raviart FEM is to apply the Galerkin approach to the nonconforming Ansatz space \( P_{1}^{\text{NC}} \) directly:

\[
\sum_{T_k \in T} \int_{T_k} \varepsilon \, \text{grad } u_h \cdot \text{grad } v = \int_{\Omega} f_h v \quad \forall v \in P_{1}^{\text{NC}}.
\]

With decreasing mesh-size the num. solution \( u_h \) converges to \( u \) with \( O(h^2) \).

More interestingly using a special post-processing we can recover a flux \( \Psi_h \) of second order accuracy \( O(h^2) \) using:

\[
\Psi_h(x) = \varepsilon \, \text{grad } u_h - \frac{f_{T_k}}{n}(x - x_{T_k}), \quad x \in T_k, \ x_{T_k} \text{ barycenter of } T_k.
\]

The normal component of \( \Psi_h \) is continuous at inter-element interfaces.
Figure: The Displacement is only continuous at the Midpoints of the Edges
The Crouzeix-Raviart Finite Element Method – Numerical Displacement $u_h$

**Figure:** The Displacement is only continuous at the Midpoints of the Edges
The Crouzeix-Raviart Finite Element Method – Numerical Flux $\Psi_h$

Figure: Plot of vector field at element interfaces and barycenters
The Crouzeix-Raviart Finite Element Method – Numerical Flux $\Psi_h$

Figure: Plot of vector field at element interfaces and barycenters
For convergence studies we implemented the Crouzeix-Raviart Finite Element Method in MATLAB [6] for two and three dimensional simplical meshes.

For benchmarking the robustness and the efficiency of this approach we used:

\[-\Delta u(x) = 3\pi^2 \sin(\pi x_1) \sin(\pi x_2) \sin(\pi x_3) \text{ in 3D,}\]
\[-\Delta u(x) = 2\pi^2 \sin(\pi x_1) \sin(\pi x_2) \text{ in 2D.}\]

The analytic solutions for the scalar potential \( u \) is given by:

\[ u(x) = \sin(\pi x_1) \sin(\pi x_2) \sin(\pi x_3) \text{ in 3D} \]
\[ u(x) = \sin(\pi x_1) \sin(\pi x_2) \text{ in 2D respectively.} \]
For convergence studies we implemented the Crouzeix-Raviart Finite Element Method in MATLAB [6] for two and three dimensional simplical meshes.

For benchmarking the robustness and the efficiency of this approach we used:

\[-\Delta u(x) = 3\pi^2 \sin(\pi x_1) \sin(\pi x_2) \sin(\pi x_3) \text{ in 3D,}\]
\[-\Delta u(x) = 2\pi^2 \sin(\pi x_1) \sin(\pi x_2) \text{ in 2D.}\]

The analytic solutions for the scalar potential \(u\) is given by:

\[u(x) = \sin(\pi x_1) \sin(\pi x_2) \sin(\pi x_3) \text{ in 3D}\]
\[u(x) = \sin(\pi x_1) \sin(\pi x_2) \text{ in 2D respectively.}\]
For convergence studies we implemented the Crouzeix-Raviart Finite Element Method in MATLAB [6] for two and three dimensional simplical meshes.

For benchmarking the robustness and the efficiency of this approach we used:

\[-\Delta u(x) = 3\pi^2 \sin(\pi x_1) \sin(\pi x_2) \sin(\pi x_3) \text{ in 3D,}\]
\[-\Delta u(x) = 2\pi^2 \sin(\pi x_1) \sin(\pi x_2) \text{ in 2D.}\]

The analytic solutions for the scalar potential $u$ is given by:

\[u(x) = \sin(\pi x_1) \sin(\pi x_2) \sin(\pi x_3) \text{ in 3D}\]
\[u(x) = \sin(\pi x_1) \sin(\pi x_2) \text{ in 2D respectively.}\]
Results – Benchmark

- We benchmarked the convergence of the numerical solution for successively refined meshes (halving the element-diameter in each step).
- The convergence of the potential shows the expected behaviour (order $O(h^2)$ implying a reduction of the error by a factor of 4 with every step).
- The error in the approximated field distribution seems to be dominated by the error at the boundary (especially at the corners of the domain) first – only approaching order $O(h^2)$ on highly refined grids.
- We benchmarked the convergence of the numerical solution for successively refined meshes (halving the element-diameter in each step).

- The convergence of the potential shows the expected behaviour (order $O(h^2)$ implying a reduction of the error by a factor of 4 with every step).

- The error in the approximated field distribution seems to be dominated by the error at the boundary (especially at the corners of the domain) first — only approaching order $O(h^2)$ on highly refined grids.
We benchmarked the convergence of the numerical solution for successively refined meshes (halving the element-diameter in each step). The convergence of the potential shows the expected behaviour ($O(h^2)$) implying a reduction of the error by a factor of 4 with every step. The error in the approximated field distribution seems to be dominated by the error at the boundary (especially at the corners of the domain) first – only approaching order $O(h^2)$ on highly refined grids.
• We benchmarked the convergence of the numerical solution for successively refined meshes (halving the element-diameter in each step).

• The convergence of the potential shows the expected behaviour (order $O(h^2)$ implying a reduction of the error by a factor of 4 with every step).

• The error in the approximated field distribution seems to be dominated by the error at the boundary (especially at the corners of the domain) first – only approaching order $O(h^2)$ on highly refined grids.
• We benchmarked the convergence of the numerical solution for successively refined meshes (halving the element-diameter in each step).

• The convergence of the potential shows the expected behaviour (order $O(h^2)$ implying a reduction of the error by a factor of 4 with every step).

• The error in the approximated field distribution seems to be dominated by the error at the boundary (especially at the corners of the domain) first – only approaching order $O(h^2)$ on highly refined grids.
We benchmarked the convergence of the numerical solution for successively refined meshes (halving the element-diameter in each step).

The convergence of the potential shows the expected behaviour (order $O(h^2)$ implying a reduction of the error by a factor of 4 with every step).

The error in the approximated field distribution seems to be dominated by the error at the boundary (especially at the corners of the domain) first – only approaching order $O(h^2)$ on highly refined grids.
Table: Convergence of successive refinements of the square $[0, 1] \times [0, 1]$; $\| e_{u_h} \|_2$ and $\| e_{u_h} \|_{\infty}$ are the $L_2$ and the maximum error of the potential $u_h$, while $\| e_{\Psi_h} \|_{\infty}$ indicates the maximum error of the approximated gradient $\Psi_h$ at interface midpoints.
<table>
<thead>
<tr>
<th>$K$</th>
<th>t(sec)</th>
<th>$| e_{uh} |_2$</th>
<th>$| e_{uh} |_\infty$</th>
<th>$| e_{\Psi h} |_\infty$</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0.002</td>
<td>1.42e+0</td>
<td>9.74e-1</td>
<td>1.44e+1</td>
</tr>
<tr>
<td>72</td>
<td>0.002</td>
<td>7.14e-1</td>
<td>9.20e-1</td>
<td>1.32e+1</td>
</tr>
<tr>
<td>672</td>
<td>0.006</td>
<td>1.49e-1</td>
<td>3.20e-1</td>
<td>6.87e+0</td>
</tr>
<tr>
<td>5760</td>
<td>0.044</td>
<td>4.02e-2</td>
<td>1.13e-1</td>
<td>2.93e+0</td>
</tr>
<tr>
<td>47616</td>
<td>0.597</td>
<td>1.07e-2</td>
<td>3.21e-2</td>
<td>8.63e-1</td>
</tr>
<tr>
<td>387072</td>
<td>9.373</td>
<td>2.74e-3</td>
<td>8.58e-3</td>
<td>2.23e-1</td>
</tr>
</tbody>
</table>

**Table:** Convergence of successive refinements of the cube $[0, 1] \times [0, 1] \times [0, 1]$; $\| e_{uh} \|_2$ and $\| e_{uh} \|_\infty$ are the $L_2$ and the maximum error of the potential $u_h$, while $\| e_{\Psi h} \|_\infty$ indicates the maximum error of the approximated gradient $\Psi_h$ at interface midpoints.
Figure: Plot of vector field at element interfaces and barycenters
Figure: Convergence is not optimal at the corners of the domain
• Though the computation time could be reduced by a factor of 2 - using SSOR as a preconditioner - for our current applications and the accuracy sought the computational cost seems to be prohibitive.

• Besides using an more efficient preconditioner we want to explore the use of geometric multigrid for Crouzeix Raviart finite elements.

• Using the right prolongation and restriction operators is not completely straight-forward for the finite element spaces involved as the successively refined function spaces are not nested (see Figure 9).

• Exploring the approaches by Kraus, Margenov and Synka [4] and geometric multigrid as described by Braess, Dryja and Hackbusch [3] seems to be most promising at the moment.
• Though the computation time could be reduced by a factor of 2 - using SSOR as a preconditioner - for our current applications and the accuracy sought the computational cost seems to be prohibitive.

• Besides using an more efficient preconditioner we want to explore the use of geometric multigrid for Crouzeix Raviart finite elements.

• Using the right prolongation and restriction operators is not completely straight-forward for the finite element spaces involved as the successively refined function spaces are not nested (see Figure 9).

• Exploring the approaches by Kraus, Margenov and Synka [4] and geometric multigrid as described by Braess, Dryja and Hackbusch [3] seems to be most promising at the moment.
Conclusions

• Though the computation time could be reduced by a factor of 2 - using SSOR as a preconditioner - for our current applications and the accuracy sought the computational cost seems to be prohibitive.

• Besides using an more efficient preconditioner we want to explore the use of geometric multigrid for Crouzeix Raviart finite elements.

• Using the right prolongation and restriction operators is not completely straight-forward for the finite element spaces involved as the successively refined function spaces are not nested (see Figure 9).

• Exploring the approaches by Kraus, Margenov and Synka [4] and geometric multigrid as described by Braess, Dryja and Hackbusch [3] seems to be most promising at the moment.
• Though the computation time could be reduced by a factor of 2 - using SSOR as a preconditioner - for our current applications and the accuracy sought the computational cost seems to be prohibitive.

• Besides using an more efficient preconditioner we want to explore the use of geometric multigrid for Crouzeix Raviart finite elements.

• Using the right prolongation and restriction operators is not completely straight-forward for the finite element spaces involved as the successively refined function spaces are not nested (see Figure 9).

• Exploring the approaches by Kraus, Margenov and Synka [4] and geometric multigrid as described by Braess, Dryja and Hackbusch [3] seems to be most promising at the moment.
• Though the computation time could be reduced by a factor of 2 - using SSOR as a preconditioner - for our current applications and the accuracy sought the computational cost seems to be prohibitive.

• Besides using an more efficient preconditioner we want to explore the use of geometric multigrid for Crouzeix Raviart finite elements.

• Using the right prolongation and restriction operators is not completely straight-forward for the finite element spaces involved as the successively refined function spaces are not nested (see Figure 9).

• Exploring the approaches by Kraus, Margenov and Synka [4] and geometric multigrid as described by Braess, Dryja and Hackbusch [3] seems to be most promising at the moment.
Questions and Answers

[Intentionally left empty]
Figure: Continuous coarse-grid function is contained in fine grid
Figure: Discontinuous coarse-grid function is not contained in fine grid
L. Marini.

An inexpensive method for the evaluation of the solution of the lowest order Raviart–Thomas mixed method.


D. N. Arnold and F. Brezzi.

Mixed and nonconforming finite element methods: implementation, postprocessing and error estimates.

D. Braess, M. Dryja, and W. Hackbusch.

A multigrid method for nonconforming FE-discretisations with application to non-matching grids.


J. Kraus, S. Margenov, and J. Synka.

On the multilevel preconditioning of Crouzeix–Raviart elliptic problems.

G. Pöplau and U. van Rienen.

An efficient 3D space charge routine with self-adaptive discretization.


**MATLAB.**

*Version 7.10.0 (R2010a).*

The MathWorks Inc.