Darcy+: Finite Element Solvers for Flow and Transport in Porous Media Implemented in C++

James "James" Liu
Graham Harper
Department of Mathematics
Colorado State University
Fort Collins, CO 80523-1874, USA
{liu,harper}@math.colostate.edu

May 18, 2018

Contents

Abstract

`Darcy+` is a C++ code package for finite element solvers for flow and transport problems in porous media. The package concentrates on 3-dim problems, although it can solve 2-dim problems also. The "workhorse" in this code package is the family of the novel weak Galerkin finite element methods. Applications focus on flow and transport in biological porous media, especially drug delivery.

Keywords: anisotropy, Darcy flow, heterogeneity, porous media, weak Galerkin

Chapter 1

Flow and Transport in Porous Media

See [?, 4, ?, ?, 6, 7, 8]. See also [16, 14, 10].

1.1 The Darcy Equation

The Darcy's law is a fundamental equation for modeling flow in porous media. It is usually further coupled with transport equations. Two examples amongst the vast applications in this regard are oil recovery in petroleum reservoirs [?, 7, 19] and drug delivery to tumors.

We consider 2-dim elliptic boundary value problems (Darcy problems) formulated as

$$\begin{cases} \nabla \cdot (-\mathbf{K}\nabla p) \equiv \nabla \cdot \mathbf{u} = f, \quad \mathbf{x} \in \Omega, \\ p = p_D, \ \ \mathbf{x} \in \Gamma^D, \qquad \mathbf{u} \cdot \mathbf{n} = u_N, \ \ \mathbf{x} \in \Gamma^N, \end{cases} \tag{1.1}$$

where $\Omega \subset \mathbb{R}^d (d = 2, 3)$ is a bounded polygonal/polyhedral domain, p the primal unknown (pressure), $\mathbf{K}$ a permeability tensor that is uniformly symmetric positive-definite, f a source term, p_D, u_N are respectively Dirichlet and Neumann boundary data, $\mathbf{n}$ the unit outward normal vector on $\partial\Omega$, which has a nonoverlapping decomposition $\Gamma^D \cup \Gamma^N$.

Define a subspace and manifold for scalar-valued functions as follows

$$H_{D,0}(\Omega) = \{p \in H^1(\Omega) : p|_{\Gamma^D} = 0\}, \qquad H_{D,p_D}(\Omega) = \{p \in H^1(\Omega) : p|_{\Gamma^D} = p_D\}.$$

The variational formulation for the primal variable pressure reads as: Seek $p \in H^1_{D,p_D}(\Omega)$ such that

$$\int_\Omega \mathbf{K}\nabla p \cdot \nabla q = \int_\Omega fq - \int_{\Gamma_N} u_N q \qquad \forall q \in H^1_{D,0}(\Omega). \tag{1.2}$$

The Darcy equation can also be rewritten as a system of two first-order equations by considering the primal variable (pressure) and flux (velocity $\mathbf{u} = -\mathbf{K}\nabla p$) as follows

$$\mathbf{K}^{-1}\mathbf{u} + \nabla p = \mathbf{0}, \qquad \nabla \cdot \mathbf{u} = f. \tag{1.3}$$

Define a subspace and manifold for vector-valued functions as follows

$$H_{N,0}(\mathrm{div}, \Omega) = \{\mathbf{v} \in L^2(\Omega)^2 : \mathrm{div}\mathbf{v} \in L^2(\Omega), \mathbf{v}|_{\Gamma^N} = \mathbf{0}\},$$
$$H_{N,u_N}(\mathrm{div}, \Omega) = \{\mathbf{v} \in L^2(\Omega)^2 : \mathrm{div}\mathbf{v} \in L^2(\Omega), \mathbf{v}|_{\Gamma^N} \cdot \mathbf{n} = u_N\}.$$

The mixed variational formulation is then: Seek $\mathbf{u} \in H_{N,u_N}(\mathrm{div}, \Omega)$ and $p \in L^2(\Omega)$ such that the following hold

$$\begin{cases} \displaystyle\int_\Omega (\mathbf{K}^{-1}\mathbf{u}) \cdot \mathbf{v} - \int_\Omega p(\nabla \cdot \mathbf{v}) = -\int_{\Gamma_D} p_D \mathbf{v} \cdot \mathbf{n} & \forall \mathbf{v} \in H_{N,0}(\mathrm{div}, \Omega), \\ \displaystyle -\int_\Omega (\nabla \cdot \mathbf{u}) q = -\int_\Omega f q & \forall q \in L^2(\Omega). \end{cases} \tag{1.4}$$

1.2 Transport Equations

1.3 Steady-state Equations

1.4 Two-phase Flow

In this section, we focus on the flow and transport in a domain Ω with heterogeneous permeability, governed by an immiscible two-phase system with a wetting phase and a nonwetting phase (denoted by w and o respectively), for example, water and oil. For simplicity of presentation, capillary pressure and gravity are not included in the model. The Darcy's law combined with a statement of conservation of mass are expressed as

$$\nabla \cdot \mathbf{u} = q, \quad \text{where } \mathbf{u} = -\lambda(S)k(\mathbf{x})\nabla p, \tag{1.5}$$

and

$$\frac{\partial S}{\partial t} + \nabla \cdot (f(S)\mathbf{u}) = q_w, \tag{1.6}$$

where $\mathbf{u}$ is the Darcy velocity, S is the saturation of the wetting phase, and k is the permeability coefficient. The total mobility $\lambda(S)$ and the flux function $f(S)$ are respectively given by

$$\lambda(S) = \frac{k_{rw}(S)}{\mu_w} + \frac{k_{ro}(S)}{\mu_o}, \quad f(S) = \frac{k_{rw}(S)/\mu_w}{\lambda(S)}, \tag{1.7}$$

where $k_{r\alpha}(\alpha = w, o)$ is the relative permeability of the phase α.

Chapter 2

Poroelasticity

2.1 Linear Elasticity

A common form of the linear elasticity is

$$\begin{cases} -\nabla \cdot \sigma(\mathbf{u}) = \mathbf{f}, \\ \mathbf{u}|_{\Gamma_D} = \mathbf{u}_D, \quad \mathbf{x} \in \Gamma_D, \\ \sigma \mathbf{n} = \mathbf{t}_N, \quad \mathbf{x} \in \Gamma_N, \end{cases} \tag{2.1}$$

where $\mathbf{u}$ is the unknown displacement (deformation) vector, the strain and stress are respectively defined as

$$\varepsilon(\mathbf{u}) = \frac{1}{2}\left(\nabla \mathbf{u} + (\nabla \mathbf{u})^T\right),$$
$$\sigma(\mathbf{u}) = 2\mu\, \varepsilon(\mathbf{u}) + \lambda(\nabla \cdot \mathbf{u})\mathbf{I} = 2\mu\, \varepsilon(\mathbf{u}) + \lambda\, \mathrm{tr}(\varepsilon(\mathbf{u})).$$

Furthermore, $\mathbf{f}$ is a body force. Dirichlet condition or displacement boundary condition; Neumann condition or traction boundary condition;

Note that the above equation (2.1) can be rewritten as

$$-\mu \Delta \mathbf{u} - (\mu + \lambda)\nabla(\nabla \cdot \mathbf{u}) = \mathbf{f}. \tag{2.2}$$

2.2 Linear Poroelasticity

Poroelasticity couples elasticity and flow in porous media.

Let $\mathbf{u}$ be the solid displacement /deformation, p be the solid pressure

$$\varepsilon(\mathbf{u}) = \frac{1}{2}\Big(\nabla \mathbf{u} + (\nabla \mathbf{u})^T\Big)$$

be the ... and the total stress tensor

$$\sigma = 2\mu\varepsilon(\mathbf{u}) + \lambda(\nabla \cdot \mathbf{u})\mathbf{I} - \alpha p \mathbf{I}$$

Two equations

$$\begin{cases} -\nabla \cdot \left(2\mu\varepsilon(\mathbf{u}) + \lambda(\nabla \cdot \mathbf{u})\mathbf{I} - \alpha p\mathbf{I}\right) = \mathbf{f} \\ \partial_t \left(c_0 p + \alpha \nabla \cdot \mathbf{u}\right) + \nabla \cdot \left(-\dfrac{\mathbf{K}}{\mu_f}\nabla p\right) = s \end{cases} \tag{2.3}$$

It is interesting to note that [13] for the two terms for the fluid content, $c_0 p$ characterizes the amount of fluid that can be squeezed into a fixed volume?!, $\alpha \nabla \cdot \mathbf{u}$ characterizes the amount of fluid that can be squeezed out of where?!.

2.3 Poroelasticity in Life Sciences

See [12, 18, 20].

Cell Cytoplasm Modeled as a Poroelastic Material. In [12], experimental evidence is provided to validate a cellular rheology model that treats the cytoplasm of living cells as a poroelastic material. This biphasic material consists of an elastic solid network (cytoskeleton, organelles, macromolecules) bathed in an interstitial fluid (cytosol). The cellular deformation is limited by the rate at which intracellular water can redistribute with the cytoplasm.

A Poroelastic Model for Hydrogel Scaffold in Tissue Engineering. In [20], a poroelastic model for a highly porous hydrogel subject to cyclic strain is validated by matching the predicted bead penetration into the hydrogel with experimental observations. The results provide insight into nutrient transport within a cyclically strained hydrogel, which could lead to improved designs of engineered tissues.

A Poroelastic Model for Brain Tissue in Convection-enhanced Drug Delivery. In [18],

2.4 Some Benchmarks for Poroelasticity

3-dim Unconfined Compression. See [**?**, 3, 5].

Chapter 3

FEM Preliminaries

3.1 Some Nice Properties of Tetrahedra

Lemma. Let T be a tetrahedron and $\lambda_i (i = 1, 2, 3, 4)$ be the barycentric coordinates. Let $\alpha, \beta, \gamma, \delta$ be nonnegative integers. Then

$$\int_T \lambda_1^\alpha \lambda_2^\beta \lambda_3^\gamma \lambda_3^\delta dT = \frac{|T|\ \alpha!\ \beta!\ \gamma!\ \delta!\ 3!}{(\alpha + \beta + \gamma + \delta + 3)!}. \tag{3.1}$$

Lemma. Let T be a tetrahedron and $\lambda_i (i = 1, 2, 3, 4)$ be the Lagrangian basis functions. Then the Gram matrix is

$$\frac{|T|}{20} \begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}.$$

Let (x_c, y_c, z_c) be the center of a tetrahedron T. Let $X = x - x_c, Y = y - y_c, Z = z - z_c$ be the normalized coordinates. Then

$$\int_T \lambda_1 X = \int_T \lambda_1 x - x_c \frac{|T|}{4} = \frac{|T|}{80}(3x_1 - x_2 - x_3 - x_4).$$

Similarly,

$$\int_T \lambda_1 Y = \tfrac{|T|}{80}(3y_1 - y_2 - y_3 - y_4),$$

$$\int_T \lambda_1 Z = \tfrac{|T|}{80}(3z_1 - z_2 - z_3 - z_4).$$

Furthermore,

$$\int_T \lambda_2 X = \tfrac{|T|}{80}(-x_1 + 3x_2 - x_3 - x_4),$$

$$\int_T \lambda_2 X = \tfrac{|T|}{80}(-y_1 + 3y_2 - y_3 - y_4),$$

$$\dots$$

$$\int_T \lambda_4 Z = \tfrac{|T|}{80}(-z_1 - z_2 - z_3 + 3z_4).$$

If we introduce two matrices,

$$C = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \end{bmatrix}, \quad D = \begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ -1 & -1 & -1 & 3 \end{bmatrix},$$

and set B as another 3×4 matrix

$$B = \begin{bmatrix} \int_T X\lambda_j \\ \int_T Y\lambda_j \\ \int_T Z\lambda_j \end{bmatrix}_{j=1,2,3,4},$$

then it is clear that

$$B = C\ D.$$

3.2 FEMs

For convenience of presentation, we concentrate on the three-dimensional model of the Darcy equation, which is usually formulated as

$$\begin{cases} \nabla \cdot (-\mathbf{K}\nabla p) \equiv \nabla \cdot \mathbf{u} = f, \quad \mathbf{x} \in \Omega, \\ p = p_D, \quad \mathbf{x} \in \Gamma^D, \\ \mathbf{u} \cdot \mathbf{n} = u_N, \quad \mathbf{x} \in \Gamma^N, \end{cases} \tag{3.2}$$

where $\Omega \subset \mathbb{R}^d (d = 2, 3)$ is a bounded polygonal/polyhedral domain that can be equipped with ... tetrahedral or hexahedral meshes, p the unknown pressure, $\mathbf{u}$ the Darcy velocity, $\mathbf{K}$ a permeability tensor that is uniformly symmetric positive-definite, f a source term, p_D, u_N are respectively Dirichlet and Neumann boundary data, $\mathbf{n}$ the unit outward normal vector on $\partial\Omega$, which has a nonoverlapping decomposition $\Gamma^D \cup \Gamma^N$. When $\Gamma^D \neq \emptyset$, the problem has a unique solution.

In many applications, the permeability matrix is a 3-by-3 full tensor. For some medical phenomena, cylindrical coordinates are more suitable [17]. The diffusion/permeability is anisotropic in the radial, angular, and vertical directions. In the cylindrical coordinates, one has a diagonal matrix

$$\mathbf{D} = \begin{bmatrix} D_r & & \\ & D_\theta & \\ & & D_z \end{bmatrix}.$$

But transformation into the Cartesian coordinates gives a full 3-by-3 permeability tensor.

The weak Galerkin finite element methods introduced in [22] adopt a completely different approach. In [7], it is demonstrated through two-dimensional examples that WGFEMs can handle well the anisotropy and heretogeneity in Darcy flow problems and WGFEMs are viable alternatives of the classical MFEMs. WGFEMs can be extended three-dimensional Darcy problems on brick and tetrahedral meshes with little technical treatment. However, extension of WGFEMs to general hexahedral meshes is nontrivial.

Mixed finite element methods with multipoint flux have been developed for hexahedral meshes [10].

Numerical schemes on two-dimensional quadrilateral meshes have been developed and analyzed for elliptic and parabolic problems in [**?**].

Properties of hexahedron and quadratures on it have been investigated in [23, 24].

Existing work [16], [**?**], [14].

3.3 Meshes

3.3.1 Brick Meshes

3.3.2 Tetrahedral Meshes

3.3.3 Hexahedral Meshes

For many applications, hexahedral meshes are desired and are preferred over the tetrahedral meshes. Hexahedral meshes with good quality can be generated using `CUBIT/Trelis`.

Characterization of Hexahedra For the unit cube $\hat{E} = [0,1]^3$, we label its vertices using binary numbers: 000 for the origin $(0,0,0)$, 100 for the vertex $(1,0,0)$, and so on so forth, instead of using $0, 1, 2, \ldots, 7$. In general we use ijk for the vertex (i,j,k) where $i,j,k \in \{0,1\}$. These should be a bit convenient as we shall see bitwise operations and triple-loops for tensor products.

Let E be a generic hexahedron with vertices labeled as P_{ijk}. as shown in Figure **??**. Here ??? characterizes the deviation of face from being a parallelogram. Similarly for all other five faces.

Let $\phi_0(s) = 1-s, \phi_1(s) = s$ for $s \in [0,1]$. The trilinear Lagrangian basis functions associated with the vertex (i,j,k) of the unit cube is $\phi_i(\hat{x})\phi_j(\hat{y})\phi_k(\hat{z})$, where $0 \leq \hat{x}, \hat{y}, \hat{z} \leq 1$. For any $(\hat{x}, \hat{y}, \hat{z})$, one can use the 8-tuple

$$(\phi_0(\hat{x})\phi_0(\hat{y})\phi_0(\hat{z}), \phi_1(\hat{x})\phi_0(\hat{y})\phi_0(\hat{z}), \cdots, \phi_1(\hat{x})\phi_1(\hat{y})\phi_1(\hat{z}),)$$

as its generalized barycentric coordinates.

Consider a general hexahedron E with vertices $\mathbf{P}_{ijk}, i, j, k \in \{0, 1\}$. The trilinear transformation from the unit cube $\hat{E}$ to the hexahedron E is given as

$$\mathbf{p} = \mathbf{p}_0 + \mathbf{v}_a\hat{x} + \mathbf{v}_b\hat{y} + \mathbf{v}_c\hat{z} + \mathbf{v}_d\hat{y}\hat{z} + \mathbf{v}_e\hat{z}\hat{x} + \mathbf{v}_f\hat{x}\hat{y} + \mathbf{v}_g\hat{x}\hat{y}\hat{z}.$$

where

$$\begin{aligned}
\mathbf{v}_a &= \mathbf{p}_{100} - \mathbf{p}_{000}, \\
\mathbf{v}_b &= \mathbf{p}_{010} - \mathbf{p}_{000}, \\
\mathbf{v}_c &= \mathbf{p}_{001} - \mathbf{p}_{000}, \\
\mathbf{v}_d &= (\mathbf{p}_{011} - \mathbf{p}_{000}) - (\mathbf{v}_b + \mathbf{v}_c), \\
\mathbf{v}_e &= (\mathbf{p}_{101} - \mathbf{p}_{000}) - (\mathbf{v}_c + \mathbf{v}_a), \\
\mathbf{v}_f &= (\mathbf{p}_{110} - \mathbf{p}_{000}) - (\mathbf{v}_a + \mathbf{v}_b), \\
\mathbf{v}_g &= (\mathbf{p}_{111} - \mathbf{p}_{000}) - ((\mathbf{v}_a + \mathbf{v}_b + \mathbf{v}_c) + (\mathbf{v}_d + \mathbf{v}_e + \mathbf{v}_f)).
\end{aligned}$$

Our notations are similar to those in [15] but emphasize the cyclic symmetry.

Clearly $\mathbf{v}_f$ characterizes deviation of the bottom face from being a parallelogram, whereas $\mathbf{v}_g$ characterizes deviation of the hexahedron from being a parallelepiped.

The tangential vectors or covariant vectors [15] are respectively

$$\begin{aligned}
\partial_{\hat{x}}\mathbf{p} &= \mathbf{a} + \mathbf{f}\hat{y} + \mathbf{e}\hat{z} + \mathbf{g}\hat{y}\hat{z}, \\
\partial_{\hat{y}}\mathbf{p} &= \mathbf{b} + \mathbf{d}\hat{z} + \mathbf{f}\hat{x} + \mathbf{g}\hat{z}\hat{x}, \ . \\
\partial_{\hat{z}}\mathbf{p} &= \mathbf{c} + \mathbf{e}\hat{x} + \mathbf{d}\hat{y} + \mathbf{g}\hat{x}\hat{y}.
\end{aligned}$$

It is interesting to note that $\partial_{\hat{x}}\mathbf{p} =: \mathbf{X}(\hat{y}, \hat{z})$ does not depend on $\hat{x}$. We use these three tangential vectors to form the Jacobian matrix of the trilinear mapping

$$\mathbf{J} = [\partial_{\hat{x}}\mathbf{p}, \partial_{\hat{y}}\mathbf{p}, \partial_{\hat{z}}\mathbf{p}].$$

It is clear that the mixed mixed product of these covariant vectors gives the determinant of the Jacobian matrix

$$\det(\mathbf{J}(\hat{x}, \hat{y}, \hat{z})) = (\mathbf{X}(\hat{y}, \hat{z}) \times \mathbf{Y}(\hat{z}, \hat{z})) \cdot \mathbf{Z}(\hat{x}, \hat{y}).$$

h^2-parallelepiped Assumptions

3.3.4 Other Types of Meshes

3.4 Approximation Spaces on Hexahedra

The K-transformation induced by the permeability matrix $\mathbf{K}$ maps the $RT_{[0]}$ basis into the P_1^3 space

The 6 basis functions $\mathbf{w}_i, 1 \leq i \leq 6$ are

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} X \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ Y \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ Z \end{bmatrix},$$

The 12 basis functions $\mathbf{v}_i, 1 \leq i \leq 12$ are

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} X \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} Y \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} Z \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ X \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ Y \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ Z \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ X \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ Y \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ Z \end{bmatrix},$$

The 12×6 matrix is

$$\begin{bmatrix} K_{11} & K_{12} & K_{13} & & & \\ K_{21} & K_{22} & K_{23} & & & \\ K_{31} & K_{32} & K_{33} & & & \\ & & & 1 & 0 & 0 \\ & & & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \\ & & & 0 & 1 & 0 \\ & & & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \\ & & & 0 & 0 & 1 \end{bmatrix}$$

3.5 Quadratures

3.6 Linear Solvers

Chapter 4

FEMs for the Darcy Equation

Hexhedral Mesh or Tetrahedral Mesh?

Chapter 5

FEMs for Transport Equations

Chapter 6

FEMs for Linear Elasticity and Poroelasticity

6.1 FEMs for Linear Elasticity

6.1.1 Continuous Galerkin FEMs for Linear Elasticity

CG P_1^3 works well for linear elasticity if the locking issue needs to be addressed.

6.1.2 Weak Galerkin FEMs for Linear Elasticity

See [**?**].

WGFEM:

Bilinear form

$$\mathcal{A}(\mathbf{u}_h, \mathbf{v}) = 2\mu(\nabla_{w,n}\mathbf{u}_h, \nabla_{w,n}\mathbf{v}) + \lambda(\nabla_{w,n}\cdot\mathbf{u}_h, \nabla_{w,n}\cdot\mathbf{v}) \tag{6.1}$$

Stabilizer

$$\mathcal{S}(\mathbf{u}_h, \mathbf{v}) = \sum_{E\in\mathcal{E}_h}(Q_h^\partial\mathbf{u}_h^\circ - \mathbf{u}_h^\partial, Q_h^\partial\mathbf{v}^\circ - \mathbf{v}^\partial), \tag{6.2}$$

that is consider the difference between the boundary piece and the (projection of) of the interior piece.

Now we establish the concepts and computation procedures for discrete weak divergences and discrete weak gradients of vector-valued discrete (polynomial) weak functions via integration by parts.

We shall $\Phi = \{\Phi^\circ, \Phi^\partial\}$ to denote a discrete (polynomial) weak function.

Its discrete weak divergence $\nabla_{w,n}\cdot\Phi$ is a scalar-valued function on E such that

$$\int_E(\nabla_{w,n}\cdot\Phi)\, w = \int_{E^\partial}\Phi^\partial\cdot(w\ \mathbf{n}) - \int_{E^\circ}\Phi^\circ\cdot(\nabla w), \quad \forall w\in P^n(E). \tag{6.3}$$

Here w is a scalar-valued degree n polynomial, and ∇w is taken in the usual sense and hence a vector-valued polynomial.

Its discrete weak gradient $\nabla_{w,n}\Phi$ is a matrix-valued polynomial function on E such that

$$\int_E (\nabla_{w,n}\Phi) : W = \int_{E^\partial} \Phi^\partial \cdot (W\ \mathbf{n}) - \int_{E^\circ} \Phi^\circ \cdot (\nabla \cdot W), \quad \forall W \in P^n(E)^{d\times d}. \tag{6.4}$$

Here W is a matrix-valued degree n polynomial, and the divergence $\nabla \cdot W$ is taken in the classical sense and hence a vector-valued polynomial. On the left-hand side of this equation, we have colon product of two matrices.

Let E be a triangle, quadrilateral, or polygon in 2-dim. Consider WG

$$(P_1(E^\circ)^2, P_1(E^\partial)^2; P_0^{2\times 2} + P_0)$$

finite element, that is, 2-dim vector-valued polynomial in element interior and 2-dim vector-valued polynomial on element boundaries, but its discrete weak gradient is specified as a 2×2 matrix with constant entries, its discrete weak divergence is just a constant. Notice that in this case, in Equation (?), w is actually taken as constant 1, so ∇w is the zero vector, and hence we have

$$\int_E (\nabla_{w,0} \cdot \Phi)\ w = \int_{E^\partial} \Phi^\partial \cdot (w\ \mathbf{n}) \tag{6.5}$$

Similarly, in equation (?), W will be taken as one of these four matrices:

$$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}, \quad \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix},$$

so their divergence will be the usual zero vector in $\mathbb{R}^2$. Hence we have

$$\int_E (\nabla_{w,0}\Phi) : W = \int_{E^\partial} \Phi^\partial \cdot (W\mathbf{n}). \tag{6.6}$$

As before, we take these functions as a basis for $P_1(E^\circ)^2$:

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \begin{bmatrix} X \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ X \end{bmatrix}, \quad \begin{bmatrix} Y \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ Y \end{bmatrix}, \tag{6.7}$$

where $X = x - x_c, Y = y - y_c$ are normalized coordinates, (x, y) are natural coordinates, (x_c, y_c) is the element center. For $P_1(e)^2$ on any edge of E, we use these four functions as its basis:

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \begin{bmatrix} Z \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ Z \end{bmatrix}. \tag{6.8}$$

By the definition, their discrete weak gradients are ... their discrete weak divergences are ...

6.2 FEMs for Linear Poroelasticity

See [**?**, 3].

Here are some fundamental issues regarding FEMs for (linear) poroelasticity.

- Time-marching should be the approach;
  – Investigate implicit Euler first;
  – Investigate Crank-Nicholson.

- Both operator-splitting (OS) and monolithic system (MS) are to be investigated;
  – OS allows utilization of existing elasticity and Darcy solvers;
  – MS involves coupling of different types of large sparse matrices.

- For simple operator splitting at each time step:
  – Solving displacement and then pressure;
  – Solving pressure and then displacement;
  Both will be investigated.

- For Strang-type operator splitting at each time step:
  – Displacement -¿ pressure -¿ displacement -¿ pressure;
  – Pressure -¿ displacement -¿ pressure -¿ displacement;
  Both will be investigated.

- For solving solid displacement (linear elasticity equation):
  CG, NC (non-conforming), WG each should be allowed;
  Need to design lowest-order WGFEMs on tetrahedral, brick, hexaherdral meshes.
  (lower than those in Wang2Zhang-JCAM-2016, should be possible)

- For solving fluid pressure (Darcy equation):
  WG on tetrahedral, brick, hexaherdral meshes all will be utilized.

6.3 A Marching-OS FE Solver for Poroelasticity

This is based on implicit Euler for time-marching and operator-splitting (OS) in space.

The 1st equation is rewritten as (simplified to) an elasticity equation

$$-\mu\Delta\mathbf{u} - (\mu+\lambda)\nabla(\nabla\cdot\mathbf{u}) + \alpha\nabla p = \mathbf{f}, \tag{6.9}$$

and hence can be solved as a linear elasticity problem

$$\mu\left(\nabla\mathbf{u}_h^{(n+1)}, \nabla\mathbf{v}\right) + (\mu+\lambda)\left(\nabla\cdot\mathbf{u}_h^{(n+1)}, \nabla\cdot\mathbf{v}\right) = \left(\mathbf{f}^{(n)}, \mathbf{v}\right) - \alpha\left(\nabla p_h^{(n)}, \mathbf{v}\right). \tag{6.10}$$

JKL Remark: For ease of presentation, we assume the fluid viscosity constant μ_f has been absorbed into the permeability tensor $\mathbf{K}$.

Applying the implicit Euler time discretization, the 2nd equation

$$\partial_t(c_0 p + \alpha\nabla\cdot\mathbf{u}) + \nabla\cdot(-\mathbf{K}\nabla p) = s \tag{6.11}$$

is discretized and simplified as an equation for fluid pressure:

$$c_0\left(\frac{p_h^{(n+1)} - p_h^{(n)}}{\Delta t}, q\right) + \alpha\left(\frac{\nabla\cdot\mathbf{u}_h^{(n+1)} - \nabla\cdot\mathbf{u}_h^{(n)}}{\Delta t}, q\right) + \left(\mathbf{K}\nabla p_h^{(n+1)}, \nabla q\right) = (s, q). \tag{6.12}$$

This is further reorganized as

$$\begin{aligned} &c_0\left(p_h^{(n+1)}, q\right) + \Delta t\left(\mathbf{K}\nabla p_h^{(n+1)}, \nabla q\right) \\ &\quad = c_0\left(p_h^{(n)}, q\right) + \Delta t\left(s^{(n)}, q\right) - \alpha\left((\nabla\cdot\mathbf{u}_h^{(n+1)}, q) - (\nabla\cdot\mathbf{u}_h^{(n)}, q)\right). \end{aligned} \tag{6.13}$$

Subproject 1:
– Use CG P_1^3 for $\mathbf{u}_h^{(n+1)}$ on a tetrahedral mesh;
– Use WG (P_0, P_0, RT_0) to solve $p_h^{(n+1)}$ on a tetrahedral mesh;
– Need compute interaction between CP_1^3 and RT_0 for $\left(\nabla p_h^{(n)}, \mathbf{v}\right)$.

Subproject 2:
– Use CG Q_1^3 for $\mathbf{u}_h^{(n+1)}$ on a brick mesh;
– Use WG $(Q_0, Q_0, RT_{[0]})$ to solve $p_h^{(n+1)}$ on a brick mesh;
– Need compute interaction between CQ_1^3 and $RT_{[0]}$ for $\left(\nabla p_h^{(n)}, \mathbf{v}\right)$.

JKL Remark: A critical issue: The approximation subspaces for $\mathbf{u}, p$ should match. In details: the discrete weak gradient of $p^{(n)}$ should match $\mathbf{v}$; the discrete weak divergence of $\mathbf{u}^{(n+1)}, \mathbf{u}^{(n)}$ should match q. Lifting technique?

Note that the above operator-splitting iterative scheme is actually a Gauss-Seidel type iterative algorithm. Let x_1, x_2 be the vectors obtained from discretizing the unknown functions with appropriate finite elements and $A_{11}, A_{12}, A_{21}, A_{22}$ be the matrices obtained from discretizing the corresponding bilinear forms (with appropriate finite elements). We have a linear system

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{6.14}$$

. Here A_{12}, A_{21} reflect the interaction of the two finite element subspaces. Actually A_{11}, A_{22} are two SPD matrices. Knowledge about the eigenvalues of A_{11}, A_{22} and properties of A_{12}, A_{21} helps determine convergence of the operator-splitting algorithm.

The operator-splitting approach has some obvious advantages

- The sizes of the two separated linear systems being solved are significantly smaller than that of the single linear system. It is well known that solving linear systems is the major part of computational costs of FEMs.
- Virtually all FE solvers for elasticity and Darcy flow can be incorporated in this framework.

Chapter 7

Testcases

7.1 Testcases for 3-dim Darcy Equation

7.1.1 Simple Testcases for 3-dim Darcy Equation

Example 1. Consider $\Omega = [0,1]^3$ (the unit cube), $\mathbf{K} = \mathbb{I}_3$ (the order-3 identity matrix), a known exact solution $p(x,y,z) = \sin(\pi x)\sin(\pi y)\sin(\pi z)$, and hence $f(x,y,z) = 3\pi^2 \sin(\pi x)\sin(\pi y)\sin(\pi z)$, accordingly a homogeneous Dirichlet boundary condition on the whole boundary $\partial\Omega$. This is probably the simplest example. But unlike its 2-dim counterpart, the numerical (or exact) pressure cannot be easily visualized, e.g., by `VisIt`, since the pressure value on the domain boundary is simply zero. Slicing is needed (or utilized) to visualize the interior pressure value.

Example 2. Consider $\Omega = [0,1]^3$ (the unit cube), $\mathbf{K} = \mathbb{I}_3$, a known exact solution $p(x,y,z) = \cos(\pi x)\cos(\pi y)\cos(\pi z)$, and hence $f(x,y,z) = 3\pi^2 \cos(\pi x)\cos(\pi y)\cos(\pi z)$. Now one has a nonzero Dirichlet boundary condition. But the numerical pressure can be easily visualized, e.g., by `VisIt`.

Example 4. `JL Remark: To be done by GH and JKL.`
Consider an example similar to the above example but the domain Ω is the unit cube minus a brick. More precisely,

$$\Omega = [0,1]^3 \setminus \Omega_I, \qquad \Omega_I = \left\{ (x,y,z) : \frac{1}{4} < x, y < \frac{3}{4}, 0 < z < 1 \right\}.$$

Example 4. `JL Remark: To be done by GH and JKL.`
Consider an example similar to the above example but the domain Ω is the unit cube minus a cylinder that has the z-axis as its axis, passes through the origin and has a radius $1/4$. The permeability is still $\mathbf{K} = \mathbb{I}_3$, a known exact solution is set as $p(x,y,z) =$

$\cos(\pi x)\cos(\pi y)\cos(\pi z)$, and hence $f(x,y,z) = 3\pi^2\cos(\pi x)\cos(\pi y)\cos(\pi z)$. The Dirichlet boundary value on the exterior flat faces of the unit cube and on the interior circular face of the cylinder are determined by the exact solution.

The focus of this testcase will be on trying tetrahedral and hexahedral meshes.

(i) Use `TetGen` to generate a sequence of tetrahedral meshes, then test `CGP1, WG(P0,P0,RT0)Tetra` on these meshes.

(ii) Use `Trelis/CUBIT` to generate a sequence of hexahedral meshes, then test `WG(P0,P0,RT[0])Hexa` on these meshes.

(iii) Use `Trelis/CUBIT` **THex** algorithm to generate a sequence of *nested* tetrahedral and hexahedral meshes, then compare the results on the tetrahedral and hexahedral meshes.

A sample mesh would be look like what is shown in Figure **??**.

Example 4. `JL Remark: To be done by GH and JKL.`
Further variation from the above example with a varying permeability.

7.1.2 More Testcases for 3-dim Darcy/Poisson

Example 1. See [10] (p.166). Consider $\Omega = [0,1]^3$ (the unit cube), a full permeability tensor

$$\mathbf{K} = \begin{bmatrix} a & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix},$$

where the parameter a could take a value of 1, 10, 100, 1000. A known exact solution is specified as

$$p(x,y,z) = x^2(1-x)^2y^2(1-y)^2z^2(1-z)^2.$$

Accordingly

$$f(x,y,z) = ...$$

Example 2. See [10] (p.167). Consider $\Omega = [0,1]^3$ (the unit cube) and a varying full permeability tensor

$$\mathbf{K} = \begin{bmatrix} y^2+2 & \cos(xy) & \sin(xz) \\ \cos(xy) & (x+3)^2 & \cos(xz) \\ \sin(xz) & \cos(yz) & (x+1)^2+z^2 \end{bmatrix}.$$

A known exact solution is specified as

$$p(x,y,z) = x^2(1-x)^2y^2(1-y)^2z^2(1-z)^2.$$

Accordingly

$$f(x,y,z) = ...$$

7.1.3 Benchmark Listed at Univ-MRS

These five problems are listed at
`https://www.latp.univ-mrs.fr/latp_numerique/?q=node/163`
Unless specified, the domain is the unit cube.

Test 1. Consider $\Omega = [0,1]^3$ (the unit cube) and a constant full permeability tensor

$$\mathbf{K} = \begin{bmatrix} 1 & 0.5 & 0 \\ 0.5 & 1 & 0.5 \\ 0 & 0.5 & 1 \end{bmatrix}.$$

A known exact solution is specified as

$$p(x,y,z) = 1 + \sin(\pi x)\sin(\pi(y+\frac{1}{2}))\sin(\pi(z+\frac{1}{3})).$$

Accordingly

$$f(x,y,z) = ...$$

Test 2. Consider $\Omega = [0,1]^3$ (the unit cube) and a varying full permeability tensor (anisotropic and heterogenous)

$$\mathbf{K} = \begin{bmatrix} y^2+z^2+1 & -xy & -xz \\ -xy & x^2+z^2+1 & -yz \\ -xz & -yz & x^2+y^2+1 \end{bmatrix}.$$

A known exact solution is specified as

$$p(x,y,z) = x^3y^2z + x\sin(2\pi xz)\sin(2\pi xy)\sin(2\pi z).$$

Accordingly

$$f(x,y,z) = ...$$

Test 3. Consider $\Omega = [0,1]^3$ (the unit cube), a constant full permeability tensor

$$\mathbf{K} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 10^3 \end{bmatrix},$$

and a known exact solution

$$p(x,y,z) = ...$$

Test 4.

Test 5. Consider $\Omega = [0,1]^3$ (the unit cube), a piecewise constant diagonal permeability tensor

$$\mathbf{K} = \begin{bmatrix} K_{11}^{(i)} & 0 & 0 \\ 0 & K_{22}^{(i)} & 0 \\ 0 & 0 & K_{33}^{(i)} \end{bmatrix} \quad \text{on } \Omega_i \ (i = 1,2,3,4),$$

where the four subdomains $\Omega_i (i = 1,2,3,4)$ of Ω are defined as follows

$$\begin{aligned} \Omega_1 &= \{(x,y,z) : 0 <= x <= 1, y <= 0.5, z <= 0.5\}, \\ \Omega_2 &= \{(x,y,z) : 0 <= x <= 1, y > 0.5, z <= 0.5\}, \\ \Omega_3 &= \{(x,y,z) : 0 <= x <= 1, y > 0.5, z > 0.5\}, \\ \Omega_4 &= \{(x,y,z) : 0 <= x <= 1, y <= 0.5, z > 0.5\}. \end{aligned}$$

by bisection in the y,z-directions. So Ω_1 is opposite to Ω_3, whereas Ω_2 is opposite to Ω_4. The permeability tensor components are specified in the following table with assignments

$$a_1 = 0.1, \quad a_2 = 10, \quad a_3 = 100, \quad a_4 = 0.01.$$

It is observed that the values of K_{22} and K_{33} are swapped in the opposite subdomains.

i	1	2	3	4
K_{11}^i	1	1	1	1
K_{22}^i	a_2	a_1	a_4	a_3
K_{33}^i	a_4	a_3	a_2	a_1

The exact solution on these four subdomains are prescribed as

$$p(x,y,z) = a_i \sin(2\pi x)\sin(2\pi y)\sin(2\pi z), \quad (x,y,z) \in \Omega_i \ (i = 1,2,3,4).$$

Accordingly,

$$f(x,y,z) = 4\pi^2 A_i \sin(2\pi x)\sin(2\pi y)\sin(2\pi z), \quad (x,y,z) \in \Omega_i \ (i = 1,2,3,4),$$

where

$$\begin{aligned} A_1 &= a_1(1 + a_2 + a_4), \quad A_2 = a_2(1 + a_1 + a_3), \\ A_3 &= a_3(1 + a_2 + a_4), \quad A_4 = a_4(1 + a_1 + a_3). \end{aligned}$$

7.2 Testcases for 3-dim Transport Equations

7.2.1 Testcases for Transient Convection-Diffusion Equations

7.2.2 Testcases for Steady Convection-Diffusion Equations

7.3 Testcases for 3-dim Linear Elasticity

A Simple Example on the Unit Cube. This example is derived from [3] Section 6.2. Here $\Omega = (0,1)^3$. An analytical solution for displacement is specified as

$$\mathbf{u}(x,y,z) = \frac{-1}{6\pi} \begin{bmatrix} \cos(2\pi x)\sin(2\pi y)\sin(2\pi z) \\ \sin(2\pi x)\cos(2\pi y)\sin(2\pi z) \\ \sin(2\pi x)\sin(2\pi y)\cos(2\pi z) \end{bmatrix}.$$

Accordingly

$$\nabla \cdot \mathbf{u} = \sin(2\pi x)\sin(2\pi y)\sin(2\pi z),$$

and

$$\nabla(\nabla \cdot \mathbf{u}) = \Delta \mathbf{u} = -12\pi^2 \mathbf{u},$$

and hence

$$\mathbf{f} = -(2\mu + \lambda)(2\pi) \begin{bmatrix} \cos(2\pi x)\sin(2\pi y)\sin(2\pi z) \\ \sin(2\pi x)\cos(2\pi y)\sin(2\pi z) \\ \sin(2\pi x)\sin(2\pi y)\cos(2\pi z) \end{bmatrix}.$$

Iron Piece of Hardware. See [1], pp.257–258,261–262. For this example, the domain models an iron piece of hardware. There are three (3) holes centered respectively at $(17, -1.5, 21)$, $(48, -1.5, 21)$, and $(20, -77, 11.5)$. Homogeneous Direct boundary conditions are posed around the inner edges by the first two holes. For the third hole, a nonzero Dirichlet boundary condition is posed by the inner edge that describes a lift in the z-direction by 0.1 unit that is, $u_D = (0, 0, 0.1)$. The remaining boundary is equipped with a traction-free boundary condition, that is homogeneous Neumann boundary condition.

7.4 Testcases for 3-dim Linear Poroelasticity

Testcase 1. In this example, in the domain $\Omega = (0,1)^3$, two-layered porous medium saturated with incompressible fluid. The interface is at 0.5 (`JL Remark: The original paper has` $? = 0.5$.) The time interval is $[0,1]$. For the lower layer, $\lambda_1 = 10^4$, $\mu_1 = 10^4$, $\kappa_1 = 10^{-1}$. For the upper layer, $\lambda_2 = 1$, $\mu_2 = 1$, $\kappa_2 = 10^{-4}$. These parameters indicate that the upper layer is softer, but less permeable than the lower one.

A local load is applied on the top surface in a rectangular domain $R = [a, b] \times [c, d] = [0.15, 0.25]^2$. The top ($z = 1$) and bottom ($z = 0$) surfaces are free to drain, the lateral walls are rigid and impermeable. After the load is applied, the porous medium deforms and fluid flows through the layered medium.

Here are the boundary conditions: On the bottom surface: $S^n = 0$, $p = 0$. On the top surface: $S^n = S_{local}$ if $(x, y) \in R$, $S^n = 0$ otherwise. Here S^n denotes the normal component of the stress tensor. S_{local} is the local load. On the side surfaces ($x = 0$ or $x = 1$ or $y = 0$ or $y = 1$), the displacement $\mathbf{u} = \mathbf{0}$ and the fluid velocity normal component $v^n = 0$.

Testcase 2: 3-dim Unconfined Compression. See [**?**, 3, 5]. This testcase examines a cylinder-shaped sepcimen that undergoes unconfined compression. The specimen/tissue is subject to a prescribed displacement in the axial direction but expand freely in the radial direction, due to the interstitial fluid flow through the radial boundary.

A closed-form analytical solution for the displacement at the outer radius a is given in an infinite series

$$\frac{u(a,t)}{a}\frac{1}{\epsilon_0} = \nu + (1-\nu)(1-2\nu)\sum_{n=1}^{\infty}\frac{e^{-\alpha_n^2\tilde{t}}}{\alpha_n^2(1-\nu)^2-(1-\nu)}, \tag{7.1}$$

where $\tilde{t} = \frac{KM}{a^2}t$ and α_n are the roots of the characteristic equation

$$J_1(x) - \frac{1-\nu}{1-2\nu}xJ_0(x) = 0.$$

Here $J_0(x)$, $J_1(x)$ are Bessel functions. Moreover, ϵ_0 is the magnitude of applied strain, $K =$ is the permeability constant, $M = \lambda + 2\mu$ is the P-wave modulus of the elastic solid skeleton .

Chapter 8

Design and Implementation of Darcy+

8.1 Implementation of FEMs

8.1.1 Basis Functions on a Line Segment/Beam

Line segments (beams, intervals) are encountered in all FEM applications.
Basis functions based on baby coordinates are good for certain cases.
Normalized basis functions should be good also (at least from some case, e.g., the Gram matrix and hence its inverse will be diagonal in some cases.
But how to normalize? Let P_0, P_1 are the beginning and ending vertices (in 1d, 2d, 3d). Let P_c be the center or midpoint and P be an arbitrary point on this line segment. It seems there are 3 or more choices.

- Choice 1: Let
$$u = \langle P_cP, P_0P_1\rangle / \|P_0P_1\|^2.$$
Then $u \in [-\frac{1}{2}, \frac{1}{2}]$. Then for the $\mathcal{P}_1$ space on this line segment, $1, u$ as an orthogonal basis, the Gram matrix and its inverse are respectively
$$\begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{12} \end{bmatrix}, \qquad \begin{bmatrix} 1 & 0 \\ 0 & 12 \end{bmatrix}.$$
We could call $\mathbf{v} = \overline{P_0P_1}/\|\overline{P_0P_1}\|^2$ the (a) direction(al) vector for the line segment.
-
-

8.1.2 Bases for RT_0 on Triangles

Normalized Basis

The natural basis is

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \begin{bmatrix} x \\ y \end{bmatrix},$$

The normalized basis is

$$\mathbf{w}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{w}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \quad \mathbf{w}_3 = \begin{bmatrix} X \\ Y \end{bmatrix},$$

where $X = x - x_c, Y = y - y_c$ and (x_c, y_c) is the element center.

Intrinsic Basis

Let $P_i (i = 1, 2, 3)$ be the vertices, e_i be the opposite edges, $|e_i|$ be the length of the i-th edge, $|E|$ the element area, $\mathbf{n}_i$ the unit outward normal vector on the i-th edge. We define

$$\phi_i = \frac{|e_i|}{2|E|}(P - P_i). \tag{8.1}$$

Then one can verify that

- $\phi_i \cdot \mathbf{n}_j = \delta_{ij}$
- $\nabla \cdot \phi_i = |e_i|/|E|$.

Conversion between the Normalized and Intrinsic Bases

From the normalized basis to the intrinsic basis: Note that

$$\begin{aligned}
\phi_1 = \frac{|e_1|}{2|E|}(P - P_1) &= -\frac{|e_1|}{2|E|}\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \frac{|e_1|}{2|E|}\begin{bmatrix} x \\ y \end{bmatrix} \\
&= -\frac{|e_1|}{2|E|}\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \frac{|e_1|}{2|E|}\begin{bmatrix} X + x_c \\ Y + y_c \end{bmatrix} \\
&= \frac{|e_1|}{2|E|}\begin{bmatrix} x_c - x_1 \\ y_c - y_1 \end{bmatrix} + \frac{|e_1|}{2|E|}\begin{bmatrix} X \\ Y \end{bmatrix}
\end{aligned}$$

Thus we have

$$\phi_1 = \frac{|e_1|}{2|E|}(x_c - x_1)\ \mathbf{w}_1 + \frac{|e_1|}{2|E|}(y_c - y_1)\ \mathbf{w}_2 + \frac{|e_1|}{2|E|}\ \mathbf{w}_3,$$

where $\mathbf{w}_i (i = 1, 2, 3)$ is the normalized basis. Similar formulas hold for ϕ_2, ϕ_3.

Using matrix notation, we have

$$\begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix} = \frac{1}{2|E|} \begin{bmatrix} |e_1|(x_c - x_1) & |e_1|(y_c - y_1) & |e_1| \\ |e_2|(x_c - x_2) & |e_2|(y_c - y_2) & |e_2| \\ |e_3|(x_c - x_3) & |e_3|(y_c - y_3) & |e_3| \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \mathbf{w}_3 \end{bmatrix}$$

Chapter 9

Use of Darcy+

9.1 Running a C++ Project with Darcy+

9.2 Mesh Generation

Here are some common ways for generating a 3-dim (brick, tetrahedral, hexahedral, etc.) mesh.

(1) Writing a simple C++ function to generate a tetrahedral or brick mesh. A hexahedral mesh can be generated by perturbing the nodes of a brick mesh.

(2) Using `TetGen` to generate tetrahedral meshes. Use a `.poly` file when needed. The mesh files can be obtained via FFT (flat file transer) or incorporating the header file and binary library provided by `TetGen`.

(3) Using `Gmsh` to generate tetrahedral or hexahedral meshes. The mesh files can be obtained via FFT (flat file transer). `Gmsh` file format `.msh` could be ASCII (plain text) or binary.

(4) Using `Trelis/CUBIT` to generate tetrahedral or hexahedral meshes. The `THex` algorithm provided by `Trelis/CUBIT` can refine a tetrahedral mesh into a hexahedral mesh. `Trelis/CUBIT` provides several file formats mesh files, but it is hard to tell which one is really good.

`TetGen` and `Gmsh` are free of charges, but `Trelis` is not and a license needs to be purchased.

Chapter 10

LinLite

10.1 Block Diagonal Schur Complement Solver: `BDSchur`

Schur complement is useful in certain special cases. Here we illustrate this procedure. Suppose a large linear system is organized as

$$\begin{bmatrix} A_{00} & A_{01} \\ A_{10} & A_{11} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \end{bmatrix}. \tag{10.1}$$

This is encountered often in FEM programming. Just imagine 0 corresponds to something related element interior, 1 corresponds to something about element interfaces. Suppose we want to eliminate the unknowns represented by vector x_0. The first equation is

$$A_{00}x_0 + A_{01}x_1 = b_0.$$

Assuming A_{00} is invertible, we obtain

$$x_0 = A_{00}^{-1}\left(b_0 - A_{01}x_1\right). \tag{10.2}$$

This implies that x_0 is available if x_1 is known. On the other hand, we could plug the above equation into the original 2nd equation to get

$$\left(A_{11} - \left(A_{10}A_{00}^{-1}\right)A_{01}\right)x_1 = b_1 - \left(A_{10}A_{00}^{-1}\right)b_0. \tag{10.3}$$

The above linear system is about just x_1 and surely has a smaller size than the original linear system. We solve it to get x_1 and then use Equation (?) to get x_0.

Here $\widehat{A_{11}} := A_{11} - A_{10}A_{00}^{-1}A_{01}$ is named as the Schur complement matrix of the original coefficient matrix.

In this elimination/solving process, two quantities appear more than once and hence new notations are introduced:

$$\widehat{A_{00}} := A_{00}^{-1}, \qquad \widehat{A_{10}} := A_{10}A_{00}^{-1}. \tag{10.4}$$

Note that for real applications, e.g., finite element schemes, x_1 represents the unknowns on (2-dim) edges or (3-dim) faces and a part of it is actually known, e.g., through Dirichlet boundary condition. But we assume modification of the linear system is done after assembly but before Schur complement.

For the weak Galerkin finite element methods, it is known that A_{00} is a block diagonal matrix, A_{01}, A_{10}, A_{11} are sparse block matrices, b_0, b_1 are known vectors, and all these data items have compatible sizes.

In summary, we have the following procedure for the Block Diagonal Schur complement solver.

- Step 1: Finding $\widehat{A_{00}} = A_{00}^{-1}$;
  (`BlockDiagMatrix` inverse)
- Step 2.1: Computing $\widehat{A_{10}} := A_{10}\widehat{A_{00}}$;
  (`SparseBlockMatrix` and `BlockDiagMatrix` multiplication)
- Step 2.2: Computing $\widehat{A_{10}}\, A_{01}$;
  (`SparseBlockMatrix` multiplication (compatible patterns))
- Step 2.3: Computing $\widehat{A_{11}} := A_{11} - \widehat{A_{10}} A_{01}$ to form the Schur complement matrix;
  (`SparseBlockMatrix` subtraction (the same pattern))
- Step 3.1: Computing $\widehat{b_1} := b_1 - \widehat{A_{10}}\, b_0$;
  (`SparseBlockMatrix` and `Vector` multiplication; `Vector` subtraction)
- Step 3.2: Reducing the Schur complete system by using vector b_2 in the "gentle" approach; See [9] (Liu,Sadre,Wang, *Proc.Comput.Sci.*(2016) p.1308)
- Step 3.3: Solving the reduced Schur system by using CG, GMRES, etc.
- Step 4: Computing $x_0 = \widehat{A_{00}}(b_0 - A_{01}x_1)$
  (`SparseBlockMatrix` and `Vector` multiplication;
  `Vector` subtraction;
  `BlockDiagMatrix` and `Vector` multiplication)

It is obvious that certain parts of this procedure are parallelizable.

Chapter 11

Interface to Other Packages

Mesh generation

✓	`PolyMesher`	Matlab code (thru Flat File Transfer (FFT))
✓	`TetGen`	A tetrahedral mesh generator (FFT, Linking)
...	`CUBIT/Trelis`	A hexahedral mesh generator

Linear Solvers: `PETSc`

Visualization

✓	`Silo`	A mesh & field I/O library, scientific database
...	`VisIt`	An interactive visualization tool (interactive computing!)

? `FreeFEM++`: Use its script language

11.1 Interface to PETSc

11.2 Interface to TetGen

11.3 Interface to Trelis/CUBIT

11.4 Interface to Silo

11.5 Interface to VisIt

Chapter 12

Extra Stuff To Be Reorganized

JKL Remark: Most parts are to be finished by GH.

12.1 More About Meshes

12.1.1 Predefining a Domain for TetGen

Mesh generation
There are many methods available to load a piecewise linear complex (PLC) into `TetGen`'s C++ interface as the domain for the Darcy flow problem. The easiest way is to define a `.poly` file beforehand and load it into the input `tetgenio` class with the `load_poly(char *)` command, where the character array represents the file name.

The format of a `.poly` file is relatively simple. It can be opened and modified in any text editor. If the first character in a line is `#`, the line is commented out. The basic layout of the file can be found on `wias-berlin.de/software/tetgen/fformats.poly.html` but the main focus here is defining more complicated objects with `.poly` files. For example, to define the unit cube with a square cut through perpendicular to the z axis, the first line to define the top facet with the square hole in it should be `2 1 1`. This means that the facet is made up of 2 regions with 1 hole and it has a boundary marker of 1. Since the facet is made up of 2 regions with a hole, the following lines should describe the shape it takes. `4 5 6 7 8` means that the first region is a quadrilateral with 4 nodes, which are nodes 5, 6, 7, and 8. Then `4 13 14 15 16` means the second region is a quadrilateral with 4 nodes, which are nodes 13, 14, 15, and 16. These two quadrilaterals should be square forming the unit cube, then the square forming the hole cut through the center. The last line to add to this facet is the hole, and it is already known that the second region defines the hole. Thus `1 0.5 0.5 1` means that hole 1 is located at coordinates $(0.5, 0.5, 1)$, which is the center of the second region. This tells TetGen to remove the second region from the first, which leaves a facet with a square hole in the center, as desired.

12.1.2 Implementation of THex Algorithm

The **THex** algorithm divides each tetrahedron into four (4) hexahedra and hence refines a tetrahedral mesh into a hexahedral mesh. In this process, we see proliferation of nodes, faces, and elements. Besides the original nodes of the tetrahedral mesh, we create

- A new node at the centroid of each (original) tetrahedron,
- A new node at the centroid of each (original) triangular face,
- A new node at the midpoint of each (original) edge.

Similarly,

- Each original triangular face is divided into three (3) **flat** quadrilateral faces,
- Six (6) new quadrilateral faces are created inside each original tetrahedron.

JKL Remark: Conjecture: These six (6) new interior faces are flat also.

See Figure ? for an illustration: A tetrahedron is partitioned into 4 hexahedra.

For a given tetrahedral mesh, we use respectively `NumNdsT`, `NumEgsT`, `NumFcsT`, `NumEmsT`, to denote the numbers of nodes, edges, triangular faces, and tetrahedra. Similarly, we use `NumNdsH`, `NumFcsH`, `NumEmsH`, to denote the numbers of nodes, quadrilateral faces, and hexahedra of the resulting hexahedral mesh. Then it is interesting to see that

```
NumNdsH = NumNdsT + NumEgsT + NumFcsT + NumEmsT;
NumFcsH = 3*NumFcsT + 6*NumEmsT;
NumEmsH = 4*NumEmsT;
```

It is not difficult to handle just one tetrahedron. However, it is nontrivial to handle an entire tetrahedral mesh. The main difficulty lies in coordinating the orientations of faces and hexahedra. We implement the THex algorithm at three different levels:

- `THexEasy`:
- `THexMed`:
- `THexFull`:

Suppose the four vertices of a given tetrahedron are labelled locally as $0, 1, 2, 3$ with 0 **being the top (zenith) and** 123 **as the base**.

JKL Remark: To ensure THex algorithm works correctly, we need to ensure a tetrahedron has the correct orientation by checking its volume being positive and swapping two base vertices if needed.

The four triangular faces of a tetrahedra are

$$\begin{aligned} &\#0 : (0, 1, 2) \\ &\#1 : (0, 2, 3) \\ &\#2 : (0, 3, 1) \\ &\#3 : (3, 2, 1). \end{aligned}$$

Note that the orientation of the base 123 (actually now oriented as 321) is somehow different than the other three faces. This assures that the normal vector points outwards as we traverse following the given order of nodes (vertices).

The edges (with connecting nodes) are labelled as

$$\#0 : (0, 1), \quad \#1 : (0, 2), \quad \#2 : (0, 3), \quad \#3 : (3, 2), \quad \#4 : (2, 1), \quad \#5 : (1, 3).$$

Accordingly, the midpoints on these edges are labelled as $4, 5, 6, 7, 8, 9$. The centroids of the four faces are labelled as $10, 11, 12, 13$. Finally, the centroid of the tetrahedron is labelled as 14. So the partition scheme uses totally 15 nodes for four hexahedra, whereas the original tetrahedron has four (4) nodes.

As shown in Figure ?, the labels of the four new hexahedra has nodes (on bottom faces and top faces, oriented counterclockwise)

$$\begin{aligned} &\#0 : \ (0, 4, 12, 6, \ \ 5, 10, 14, 11) \\ &\#1 : \ (1, 8, 13, 9, \ \ 4, 10, 14, 12) \\ &\#2 : \ (2, 7, 13, 8, \ \ 5, 11, 14, 10) \\ &\#3 : \ (3, 6, 12, 9, \ \ 7, 11, 14, 13) \end{aligned}$$

It is nice to see that each hexahedron involves

- one original node;
- three (adjacent) edge midpoints;
- three face centroids;
- one (the) element centroid.

Note that each of the original four (4) triangular faces of the tetrahedron is partitioned into three (3) quadrilaterals. There are totally twelve (12) such quadrilaterals and they are all **flat**. These 12 flat quadrilateral faces can be expressed as

```
#0: (0,4,10,5),     #1: (1,8,10,4),     #2: (2,5,10,8),
#3: (0,5,11,6),     #4: (2,7,11,5),     #5: (3,6,11,7),
#6: (0,6,12,4),     #7: (3,9,12,6),     #8: (1,4,12,9),
#9: (3,7,13,9),    #10: (2,8,13,7),    #11: (1,9,13,8).
```

In this partition process, six (6) new quadrilateral faces are created inside the tetrahedron, as shown below

```
#12: (4,10,14,12),   #13: (5,11,14,10),    #14: (6,12,14,11),
#15: (7,13,14,11),   #16: (8,13,14,10),    #17: (9,13,14,12).
```

Each of them appears twice (is shared by two hexahedra). So there are totally (4*3+6*2) 24 faces. Of course, this is correct $4 * 6 = 24$.

In summary, totally four (4) hexahedra are created and hence $4 * 6 = 24$ faces are involved. Each of the six (6) interior quadrilateral faces is clearly used twice, and each of the 12 flat quadrilateral faces (on the original triangular faces) is used only once.

12.2 H(div)- and WG Finite Elements

Inspired by [2], we construct H(div)-type $ABF_{[0]}$ elements for bricks and hexahedra as a remedy for the deficiency of $RT_{[0]}$ elements in approximating divergence. In particular, dim$ABF_{[0]} = 9$ for bricks or hexahedra. One could choose a set of normalized basis functions as follows

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} X \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} X^2 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ Y \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ Y^2 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ Z \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ Z^2 \end{bmatrix},$$

where $X = x - x_c, Y = y - y_c, Z = z - z_c$ and (x_c, y_c, z_c) is the element center.

The Gram matrix of the above basis is ...

`JKL Remark`: We might need rearrange the above basis functions in another order to show its hierarchy with $RT_{[0]}$ normalized basis.

It is clear that the divergence of these basis functions offers variation in X, Y, Z in addition to constants. and better approximation capacity than the $RT_{[0]}$ elements.

`JKL Reamrk: The following is up to investigation. We don't know yet what would be a good combination.`

We utilize $ABF_{[0]}$ elements to construct WG $(P_1, P_1, ABF_{[0]})$ elements on hexahedra, which include bricks as a special case.

12.3 A Usual Procedure for WG in C++

The **AMS Procedure** for WG (in C/C++)

- **A**ssembly for element interior unknowns and face unknowns separately;
- **M**odifying the partitioned linear system;
- **S**olving the modified linear system based on **S**chur complement.

12.3.1 Assembly

Assembling element interior unknowns and face unknowns separately:

- 4 matrices:
  A_{00}:
  A_{01}:
  A_{10}:
  A_{11}:
  Compatible sparsity patterns

- 3 vectors:
  `GlbRhsE`:
  `GlbRhsC`: The Neumann boundary conditions contribute to this part;
  `EssVec`: Same size as `GlbRhsC`; Contributed from Dirichlet boundary conditions.
  Note: Faces have 3 subcategories: Interior, Neumann, Dirichlet.

12.3.2 Modifying

Modifying the partitioned linear system as illustrated below. For ease of presentation, let

- x_1, x_2 represent interior unknowns;
- x_3, x_4, x_5 represent face unknowns;
- $x_4 = c$ represents a Dirichlet type (essential) boundary condition;

and the partitioned linear system takes the following form

$$\begin{bmatrix} a_{11} & 0 & a_{13} & a_{14} & a_{15} \\ 0 & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix}.$$

By enforcing the essential condition, we obtain

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & 0 & a_{15} \\ a_{21} & a_{22} & a_{23} & 0 & a_{25} \\ a_{31} & a_{32} & a_{33} & 0 & a_{35} \\ 0 & 0 & 0 & 1 & 0 \\ a_{51} & a_{52} & a_{53} & 0 & a_{55} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} b_1 - a_{14}c \\ b_2 - a_{24}c \\ b_3 - a_{34}c \\ c \\ b_5 - a_{54}c \end{bmatrix}.$$

Assuming the assembly is completed with 4 matrices and 3 vectors

$$\begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \qquad \begin{bmatrix} b_3 \\ b_4 \\ b_5 \end{bmatrix}, \qquad \begin{bmatrix} 0 \\ c \\ 0 \end{bmatrix}.$$

The modification proceeds in three steps.

Step 1. Modifying RHS upper part by matrix-vector multiplication and vector subtraction

$$\begin{bmatrix} b_1 \\ b_2 \end{bmatrix} - \begin{bmatrix} a_{13} & a_{14} & a_{15} \\ a_{23} & a_{24} & a_{25} \end{bmatrix} \begin{bmatrix} 0 \\ c \\ 0 \end{bmatrix} \Longrightarrow \begin{bmatrix} b_1 - a_{14}c \\ b_2 - a_{24}c \end{bmatrix}.$$

Step 2. Modifying RHS lower part in two sub-steps: Perform matrix-vector multiplication and vector subtraction; Then replace the middle component by c.

$$\begin{bmatrix} b_3 \\ b_4 \\ b_5 \end{bmatrix} - \begin{bmatrix} a_{33} & a_{34} & a_{35} \\ a_{43} & a_{44} & a_{45} \\ a_{53} & a_{54} & a_{55} \end{bmatrix} \begin{bmatrix} 0 \\ c \\ 0 \end{bmatrix} \Longrightarrow \begin{bmatrix} b_3 - a_{34}c \\ b_4 - a_{44}c \\ b_5 - a_{54}c \end{bmatrix} \rightarrow \begin{bmatrix} b_3 - a_{34}c \\ c \\ b_5 - a_{54}c \end{bmatrix}.$$

Step 3. Modifying the 3 matrices as follows

$$\begin{bmatrix} a_{13} & a_{14} & a_{15} \\ a_{23} & a_{24} & a_{25} \end{bmatrix} \rightarrow \begin{bmatrix} a_{13} & 0 & a_{15} \\ a_{23} & 0 & a_{25} \end{bmatrix}, \begin{bmatrix} a_{31} & a_{32} \\ a_{41} & a_{42} \\ a_{51} & a_{52} \end{bmatrix} \rightarrow \begin{bmatrix} a_{31} & a_{32} \\ 0 & 0 \\ a_{51} & a_{52} \end{bmatrix},$$

$$\begin{bmatrix} a_{33} & a_{34} & a_{35} \\ a_{43} & a_{44} & a_{45} \\ a_{53} & a_{54} & a_{55} \end{bmatrix} \rightarrow \begin{bmatrix} a_{33} & 0 & a_{35} \\ 0 & 1 & 0 \\ a_{53} & 0 & a_{55} \end{bmatrix}.$$

12.3.3 Solving

Solving the modified linear system based on **Schur complement**. There are two paths for handling Schur complement.

- Forming the Schur complement explicitly:

  $$S = A_{11} - A_{10}A_{00}^{-1}A_{01}.$$

  This is usually expensive and hence less used.

- Just implement the action of Schur complement on a vector, as frequently seen in an iterative solver:

  $$\texttt{Sv} = \texttt{A11} * \texttt{v} - \texttt{A10} * (\texttt{A00inv} * (\texttt{A01} * \texttt{v}));$$

  This is all matrix-vector multiplication.

Numerical Experiments on Type II meshes: Trapezoidal but asymptotically on the xy-plane; Uniform in the z-direction. Test on the unit cube $\Omega = [0,1]^3$, permeability $\mathbf{K} = \mathbf{I}_3$, known pressure solution $p(x,y,z) = \cos(\pi x)\cos(\pi y)\cos(\pi z)$.

```
m=3;  nz=8;
#elements= 512  #faces= 1728
DOFs= 2240
itr=145
NumerPresEm max = 0.934151
ProjPresEm  max = 0.934487
Max. err. pressure = 0.00513034
Max. flux discrepancy= 1.54737e-015
L2ErrPres= 0.0732277  L2ErrVel= 0.343678  L2ErrFlux= 0.0707968
Time taken = 4.61 seconds

m=4;  nz=16;
#elements= 4096  #faces= 13056
DOFs= 17152
itr=322
NumerPresEm max = 0.983861
ProjPresEm  max = 0.983839
Max. err. pressure = 0.00162264
Max. flux discrepancy= 1.84835e-015
L2ErrPres= 0.0369048  L2ErrVel= 0.171415  L2ErrFlux= 0.0353306
Time taken = 38.62 seconds

m=5;  nz=32;
#elements= 32768  #faces= 101376
DOFs= 134144
itr=654
NumerPresEm max = 0.996018
ProjPresEm  max = 0.99601
Max. err. pressure = 0.000453773
Max. flux discrepancy= 1.2837e-015
L2ErrPres= 0.0184889  L2ErrVel= 0.0856066  L2ErrFlux= 0.0176481
Time taken = 346.623 seconds

m=6;  nz=64;
#elements= 262144  #faces= 798720
DOFs= 1060864
itr=1311
NumerPresEm max = 0.999011
ProjPresEm  max = 0.99901
Max. err. pressure = 0.00013029
Max. flux discrepancy= 1.07441e-015
L2ErrPres= 0.00924903  L2ErrVel= 0.0427858  L2ErrFlux= 0.00882106
Time taken = 3481.54 seconds
```

12.4 Lowest-order WGFEMs for Linear Elasticity on Hexahedral Meshes

WG$(Q_0^3, Q_0^3; RT_{[0]}^3, Q_0)$ can be developed for 3-dim linear elasticity problems on hexahedral meshes.

We first examine the matrix-version Raviart-Thomas space $RT_{[0]}^3$ on a hexahedron E. As usual, we denote $X = x - x_c, Y = y - y_c, Z = z - z_c$, assuming (x_c, y_c, z_c) is the element center. We use

$$\mathbf{w}_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \mathbf{w}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \mathbf{w}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \mathbf{w}_4 = \begin{bmatrix} X \\ 0 \\ 0 \end{bmatrix}, \mathbf{w}_5 = \begin{bmatrix} 0 \\ Y \\ 0 \end{bmatrix}, \mathbf{w}_6 = \begin{bmatrix} 0 \\ 0 \\ Z \end{bmatrix}$$

as a basis for vector-version $RT_{[0]}(E)$, which has dimension 6. Then $RT_{[0]}^3$ consists of matrix functions, in which each row vector is in $RT_{[0]}$. So there are 18 basis functions for such a basis of $RT_{[0]}^3$, whose Gram matrix is an 18×18 SPD matrix. Shown below are just 4 of these 18 basis functions.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \cdots, \begin{bmatrix} X & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \cdots \begin{bmatrix} 0 & 0 & 0 \\ X & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \cdots, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & Z \end{bmatrix}.$$

Within the WG framework, we use constant vectors (Q_0^3) in element interiors and constant vectors (Q_0^3) on faces to approximate displacement. Locally for each hexahedron, we have 21 such basis functions. Their discrete weak gradients, as 3×3 matrix, are specified in $RT_{[0]}^3$.

`JL20170519: TO BE FINISHED/CHECKED BY Graham`

Example 1. This is a simple example on the unit cube $\Omega = (0,1)^3$. By default, the Lame constants are $\lambda = \mu = 1$. But λ value can be changed to test locking-free property. We have a known analytical expression for the displacement

$$\mathbf{u} = \frac{1}{3\pi} \begin{bmatrix} \sin(\pi x)\cos(\pi y)\cos(\pi z) \\ \cos(\pi x)\sin(\pi y)\cos(\pi z) \\ \cos(\pi x)\cos(\pi y)\sin(\pi z) \end{bmatrix}.$$

Accordingly, we have the dilation (divergence of displacement) expressed as

$$\nabla \cdot \mathbf{u} = \cos(\pi x)\cos(\pi y)\cos(\pi z).$$

The right-hand side is

$$\mathbf{f} = \pi(\lambda + 2\mu) \begin{bmatrix} \sin(\pi x)\cos(\pi y)\cos(\pi z) \\ \cos(\pi x)\sin(\pi y)\cos(\pi z) \\ \cos(\pi x)\cos(\pi y)\sin(\pi z) \end{bmatrix}.$$

The stress is

$$\sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix}.$$

where

$$\begin{cases} 3\sigma_{xx} = (3\lambda + 2\mu)\cos(\pi x)\cos(\pi y)\cos(\pi z), \\ 3\sigma_{xy} = -2\mu \sin(\pi x)\sin(\pi y)\cos(\pi z), \\ 3\sigma_{xz} = -2\mu \sin(\pi x)\cos(\pi y)\sin(\pi z), \\ 3\sigma_{yx} = -2\mu \sin(\pi x)\sin(\pi y)\cos(\pi z), \\ 3\sigma_{yy} = (3\lambda + 2\mu)\cos(\pi x)\cos(\pi y)\cos(\pi z), \\ 3\sigma_{yz} = -2\mu \cos(\pi x)\sin(\pi y)\sin(\pi z), \\ 3\sigma_{zx} = -2\mu \sin(\pi x)\cos(\pi y)\sin(\pi z), \\ 3\sigma_{zy} = -2\mu \cos(\pi x)\sin(\pi y)\sin(\pi z), \\ 3\sigma_{zz} = (3\lambda + 2\mu)\cos(\pi x)\cos(\pi y)\cos(\pi z). \end{cases}$$

Example 2. This is example is derived from [3] Section 6.2. Here we consider just an elasticity problem: $\Omega = (0,1)^3$, $\lambda = 1$, $\mu = 1$. An analytical solution for displacement is specified as

$$\mathbf{u} = -\frac{1}{6\pi} \begin{bmatrix} \cos(2\pi x)\sin(2\pi y)\sin(2\pi z) \\ \sin(2\pi x)\cos(2\pi y)\sin(2\pi z) \\ \sin(2\pi x)\sin(2\pi y)\cos(2\pi z) \end{bmatrix}.$$

Accordingly, we have

$$\nabla \cdot \mathbf{u} = \sin(2\pi x)\sin(2\pi y)\sin(2\pi z),$$

and

$$\mathbf{f} = -(2\mu + \lambda) \begin{bmatrix} \cos(2\pi x)\sin(2\pi y)\sin(2\pi z) \\ \sin(2\pi x)\cos(2\pi y)\sin(2\pi z) \\ \sin(2\pi x)\sin(2\pi y)\cos(2\pi z) \end{bmatrix}.$$

The expression for stress σ is a bit more complicated and hence omitted.

Applying WG$(Q_0^3, Q_0^3; RT_{[0]}^3, Q_0)$ to this problem on a sequence of brick meshes, we observe first order convergence in the L_2-norms of the errors in displacement, stress, and dilation. In Table 12.4, for $h = \frac{1}{64}$, we have more than 4 millions unknowns, and the code runs for about 1 hour on a MacBookPro.

JL Remarks: Further investigation on

(i) Proof of first order convergence in displacement, stress, and dilation. `Talk to Dr.Zheng PNNL.`

Table 12.1: Example: Numerical results of WG$(Q_0^3, Q_0^3; RT_{[0]}^3, Q_0)$ on **brick** meshes

$1/h$	L2ErrDspl	L2ErrStrs	L2ErrDiv	#Iterations	Runtime
4	2.268E-2	6.813E-1	2.457E-1	27	0.3s
8	1.239E-2	3.515E-1	1.350E-1	113	2.3s
16	6.333E-3	1.761E-1	6.894E-2	264	23s
32	3.183E-3	8.811E-2	3.465E-2	528	268s
64	1.594E-3	4.405E-2	1.734E-2	987	3460s
Conv.rate	1st order	1st order	1st order		

Table 12.2: Example: Numerical results of WG$(Q_0^3, Q_0^3; RT_{[0]}^3, Q_0)$ on asymptotically parallelopiped **hexahedral** meshes ($\delta = 1$ for perturbation from brick meshes)

$1/h$	L2ErrDspl	L2ErrStrs	L2ErrDiv	#Iterations	Runtime
4	2.275E-2	6.862E-1	2.466E-1	113	0.4s
8	1.261E-2	3.714E-1	1.377E-1	310	3.0s
16	6.517E-3	1.900E-1	7.103E-2	714	38s
32	3.287E-3	9.567E-2	3.579E-2	1413	534s
64					
Conv.rate					

(ii) Do we get weak continuity at face midpoint as for quadrilateral meshes? Any proof?

(iii) Effects of nonflat faces. Nonflat faces are really annoying. It affects computational cost and approximation accuracy.

(iv) We are using only Q_0^3 on faces. Three other basis functions for P_{rm} are not used. What are we missing?

(v) More testing examples: See [11] for examples from mechanical engineering. Any examples from tissue engineering? See [3] Section 6.4 Example for the elasticity part. See also [18].

12.5 WG$(P_1^3, P_{rm}; P_0^{3\times 3}, P_0)$ for Linear Elasticity on Hexahedral Meshes

A family of WGFEMs for linear elasticity was developed in [21] that can be applied polygonal and polyhedral meshes, but stabilization is needed.

In this section, we discuss WG$(P_1^3, P_{rm}; P_0^{3\times 3}, P_0)$ for elasticity in hexahedral meshes.

For $P_0^{3\times 3}$, it has dimension 9 and an obvious basis as follows

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \cdots, \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \cdots, \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

whose Gram matrix is simply an order 9 scalar matrix, the scalar is the hexahedral volume.

Later on, for computing numerical strain, we need the Gram matrix of the averaged basis functions. For convenience, we listed below.

$$\frac{|E|}{2}\left[\begin{array}{ccc|ccc|ccc} 2 & & & & & & & & \\ & 1 & & 1 & & & & & \\ & & 1 & & & & 1 & & \\ \hline & 1 & & 1 & & & & & \\ & & & & 2 & & & & \\ & & & & & 1 & & 1 & \\ \hline & & 1 & & & & 1 & & \\ & & & & & 1 & & 1 & \\ & & & & & & & & 2 \end{array}\right].$$

Note that on each face, the space of rigid motions has dimension 6. For consistency and convenience, we consider the following normalized basis

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ \tilde{z} \\ -\tilde{y} \end{bmatrix}, \begin{bmatrix} -\tilde{z} \\ 0 \\ \tilde{x} \end{bmatrix}, \begin{bmatrix} \tilde{y} \\ -\tilde{x} \\ 0 \end{bmatrix}, \tag{12.1}$$

where (x_m, y_m, z_m) is the face midpoint and $\tilde{x} = x - x_m, \tilde{y} = y - y_m, \tilde{z} = z - z_m$. The Gram matrix is a 6×6 SPD and its entries can be computed using (Gaussian) quadratures. The Gram matrix will be used for implementing the projection used in the stabilization term.

Note that for the element interior, P_1^3 has dimension 12. However, the discrete weak gradient and discrete weak divergence of these 12 basis (vector-valued) functions are simply 3×3 zero matrix and scalar.

So this WGFEM relies on the basis functions defined on the faces. For discrete weak gradients, the definition is still

$$\int_E \nabla_{w,d}\phi : W = \int_{E^\partial} \phi^\partial \cdot (W\mathbf{n}) - \int_{E^\circ} \phi^\circ \cdot (\nabla \cdot W). \tag{12.2}$$

On each face, for any of the 6 basis functions in P_{rm}, the first term on the above right-hand side produces nonzero terms, which are shown in the right-hand side of a 9×9 SPD linear system. Specifically, the six RHS are as follows

$$\begin{bmatrix} \int_f n_1 \\ \int_f n_2 \\ \int_f n_3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ \int_f n_1 \\ \int_f n_2 \\ \int_f n_3 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \int_f n_1 \\ \int_f n_2 \\ \int_f n_3 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ \int_f n_1\tilde{z} \\ \int_f n_2\tilde{z} \\ \int_f n_3\tilde{z} \\ -\int_f n_1\tilde{y} \\ -\int_f n_2\tilde{y} \\ -\int_f n_3\tilde{y} \end{bmatrix}, \begin{bmatrix} -\int_f n_1\tilde{z} \\ -\int_f n_2\tilde{z} \\ -\int_f n_3\tilde{z} \\ 0 \\ 0 \\ 0 \\ \int_f n_1\tilde{x} \\ \int_f n_2\tilde{x} \\ \int_f n_3\tilde{x} \end{bmatrix}, \begin{bmatrix} \int_f n_1\tilde{y} \\ \int_f n_2\tilde{y} \\ \int_f n_3\tilde{y} \\ -\int_f n_1\tilde{x} \\ -\int_f n_2\tilde{x} \\ -\int_f n_3\tilde{x} \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

where $\mathbf{n} = (n_1, n_2, n_3)$ is the (varying) outward unit normal vector on a particular face.

Similarly, for a particular face, the discrete weak divergence of the 6 basis functions in P_{rm} are respectively

$$\frac{\int_f n_1}{|E|}, \ \frac{\int_f n_2}{|E|}, \ \frac{\int_f n_3}{|E|}, \ \frac{\int_f n_2\tilde{z} - n_3\tilde{y}}{|E|}, \ \frac{\int_f n_3\tilde{x} - n_1\tilde{z}}{|E|}, \ \frac{\int_f n_1\tilde{y} - n_2\tilde{x}}{|E|}. \tag{12.3}$$

12.6 Lowest-order WGFEM for Elasticity on Tetrahedral Meshes

WG $(P_0^3, P_0^3; RT_0^3, P_0)$ for linear elasticity on a tetrahedral mesh.

Let Te be a tetrahedron. It is known that $\dim(RT_0^3) = 12$. Let (x_c, y_c, z_c) be the element center and $X = x - x_c, Y = y - y_c, Z = z - z_c$ be the normalized coordinates.

For convenience, we denote

$$W_1=\begin{bmatrix}1&0&0\\0&0&0\\0&0&0\end{bmatrix},W_2=\begin{bmatrix}0&1&0\\0&0&0\\0&0&0\end{bmatrix},W_3=\begin{bmatrix}0&0&1\\0&0&0\\0&0&0\end{bmatrix},W_4=\begin{bmatrix}X&Y&Z\\0&0&0\\0&0&0\end{bmatrix},$$
$$W_5=\begin{bmatrix}0&0&0\\1&0&0\\0&0&0\end{bmatrix},W_6=\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix},W_7=\begin{bmatrix}0&0&0\\0&0&1\\0&0&0\end{bmatrix},W_8=\begin{bmatrix}0&0&0\\X&Y&Z\\0&0&0\end{bmatrix},$$
$$W_9=\begin{bmatrix}0&0&0\\0&0&0\\1&0&0\end{bmatrix},W_{10}=\begin{bmatrix}0&0&0\\0&0&0\\0&1&0\end{bmatrix},W_{11}=\begin{bmatrix}0&0&0\\0&0&0\\0&0&1\end{bmatrix},W_{12}=\begin{bmatrix}0&0&0\\0&0&0\\X&Y&Z\end{bmatrix}, \tag{12.4}$$

Then we have $\overline{W}_j=\frac{1}{2}(W_j+W_j^T)$ expressed as

$$\overline{W}_1=\begin{bmatrix}1&0&0\\0&0&0\\0&0&0\end{bmatrix},\overline{W}_2=\begin{bmatrix}0&\frac{1}{2}&0\\\frac{1}{2}&0&0\\0&0&0\end{bmatrix},\overline{W}_3=\begin{bmatrix}0&0&\frac{1}{2}\\0&0&0\\\frac{1}{2}&0&0\end{bmatrix},\overline{W}_4=\begin{bmatrix}X&\frac{Y}{2}&\frac{Z}{2}\\\frac{Y}{2}&0&0\\\frac{Z}{2}&0&0\end{bmatrix},$$
$$\overline{W}_5=\begin{bmatrix}0&\frac{1}{2}&0\\\frac{1}{2}&0&0\\0&0&0\end{bmatrix},\overline{W}_6=\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix},\overline{W}_7=\begin{bmatrix}0&0&0\\0&0&\frac{1}{2}\\0&\frac{1}{2}&0\end{bmatrix},\overline{W}_8=\begin{bmatrix}0&\frac{X}{2}&0\\\frac{X}{2}&Y&\frac{Z}{2}\\0&\frac{Z}{2}&0\end{bmatrix},$$
$$\overline{W}_9=\begin{bmatrix}0&0&\frac{1}{2}\\0&0&0\\\frac{1}{2}&0&0\end{bmatrix},\overline{W}_{10}=\begin{bmatrix}0&0&0\\0&0&\frac{1}{2}\\0&\frac{1}{2}&0\end{bmatrix},\overline{W}_{11}=\begin{bmatrix}0&0&0\\0&0&0\\0&0&1\end{bmatrix},\overline{W}_{12}=\begin{bmatrix}0&0&\frac{X}{2}\\0&0&\frac{Y}{2}\\\frac{X}{2}&\frac{Y}{2}&Z\end{bmatrix}, \tag{12.5}$$

For a WG (P_0^3,P_0^3) element, there are 15 basis functions: 3 for interior, and 3 for each of the four faces. Through integration by parts, their discrete weak gradients are specified in RT_0^3, and their discrete weak divergence are specified in P_0. To be more specific, if $\phi_i(1\le i\le 15)$ is such a WG basis function, then

$$\nabla_{w,d}\phi_i=\sum_{j=1}^{12}c_{ij}W_j,\qquad \nabla_{w,d}\cdot\phi_i=d_i.$$

Accordingly

$$\varepsilon_{w,d}(\phi_i)=2\mu\sum_{j=1}^{1}2c_{ij}\overline{W}_j,$$

where $\overline{W}_j=\frac{1}{2}(W_j+W_j^T)$. Finally we have stress

$$\sigma(\phi_i)=2\mu\sum_{j=1}^{12}c_{ij}\overline{W}_j+\lambda d_i\mathbb{I}_3.$$

Actually, we will rewrite the identity matrix as

$$\mathbb{I}_3=W_1+W_6+W_{11}.$$

Based on these efforts, we have

$$\begin{cases} \sigma_{xx} = (2\mu c_{i,1} + \lambda d_i) + 2\mu c_{i,4} X, \\ \sigma_{yy} = (2\mu c_{i,6} + \lambda d_i) + 2\mu c_{i,8} Y, \\ \sigma_{zz} = (2\mu c_{i,11} + \lambda d_i) + 2\mu c_{i,12} Z, \\ \sigma_{xy} = ... \\ \sigma_{xz} = ... \\ \sigma_{yz} = ... \end{cases} \tag{12.6}$$

It is clear that the normal stress σ_{xx} is a linear function of X, σ_{yy} is a linear function of Y, σ_{zz} is a linear function of Z, whereas the shear stress σ_{xy} is a linear function of X, Y and similarly for σ_{xz}, σ_{yz}.

Next we present numerical experiments.

Example 1: Domain $\Omega = (0,1)^3$ (unit cube), Elasticity parameter $\lambda = \mu = 1$. Consider uniform tetrahedral meshes: n partitions each in the x, y, z-directions. Each brick block is further divided into 6 tetrahedra.

The single-matrix approach is used. A 5-point Gaussian quadrature is applied on tetrahedra, whereas a 3-point Gaussian quadrature is applied on triangles. A conjugate gradient solver for sparse block SPD linear systems is employed with both tolerance and threshold set as 10^{-24}.

Shown in Table ? are the numerical results: L^2-norms of the errors in displacementment, stress, and dilation. First order convergence rates can be clearly observed.

Table 12.3: Example 1: WG $(P_0^3, P_0^3; RT_0^3, P_0)$ for elasticity on tetrahedral meshes

n	DOFs	#itrs	L2ErrDspl	L2ErrStrs	L2ErrDiv	Runtime
4	3744	328	1.885E-2	4.927E-1	1.163E-1	
8	28800	619	9.214E-3	2.796E-1	5.389E-2	
16	225792	1190	4.540E-3	1.453E-2	2.544E-2	136s
32	1787904	2306	2.259E-3	7.345E-2	1.245E-2	2583s
			1st order	1st order	1st order	

Bibliography

[1] J. Alberty, C. Carstensen, S. A. Funken, and R. Klose. Matlab implementation of the finite element method in elasticity. *Computing*, 69:239–263, 2002.

[2] D.N. Arnold, D. Boffi, and R.S. Falk. Quadrilateral h(div) finite elements. *SIAM J. Numer. Anal.*, 42:2429–2451, 2005.

[3] Lorenz Berger, Rafel Bordas, David Kay, and Simon Tavener. Stabilized lowest-order finite element approximation for linear three-field poroelasticity. *SIAM J. Sci. Comput.*, 37:A2222–A2245, 2015.

[4] L. Bush and V. Ginting. On the application of the continuous galerkin finite element method for conservation problems. *SIAM J. Sci. Comput.*, 35:A2953–A2975, 2013.

[5] Stephen C. Cowin and Stephen B. Doty. *Tissue Mechanics*. Springer, 2007.

[6] V. Ginting, Guang Lin, and Jiangguo Liu. On application of the weak galerkin finite element method to a two-phase model for subsurface flow. *J. Sci. Comput.*, 66:225–239, 2016.

[7] G. Lin, J. Liu, L. Mu, and X. Ye. Weak galerkin finite element methdos for darcy flow: Anistropy and heterogeneity. *J. Comput. Phys.*, 276:422–437, 2014.

[8] Guang Lin, Jiangguo Liu, and Farrah Sadre-Marandi. A comparative study on the weak galerkin, discontinuous galerkin, and mixed finite element methods. *J. Comput. Appl. Math.*, 273:346–362, 2015.

[9] Jiangguo Liu, Farrah Sadre-Marandi, and Zhuoran Wang. Darcylite: A matlab toolbox for darcy flow computation. *Procedia Computer Science*, 80:In press, 2016.

[10] M.F.Wheeler, Guangri Xue, and I.Yotov. A multipoint flux mixed finite element method on distorted quadrilaterals and hexahedra. *Numer. Math.*, 121:165–204, 2012.

[11] Dubravka Mijuca. On hexahedral finite element hc8/27 in elasticity. *Comput. Mech.*, 33:466–480, 2004.

[12] E. Moeendarbary, L. Valon, M. Fritzsche, A. R. Harris, D. A. Moulding, A. J. Thrasher, E. Stride, L. Mahadevan, and G. T. Charras. The cytoplasm of living cells behaves as a poroelastic material. *Nature Materials*, 12:253–261, 2013.

[13] Phillip J. Phillips. *Finite element methods in linear poroelasticity: Theoretical and computational results*. PhD thesis, University of Texas at Austin, 2005.

[14] R.Ingram, M.F.Wheeler, and I.Yotov. A multipoint flux mixed finite element method on hexahedra. *SIAM J. Numer. Anal.*, 48:1281–1312, 2010.

[15] R.L.Naff, T.F.Russell, and J.D.Wilson. Shape functions for velocity interpolation in general hexahedral cells. *Comput. Geosci.*, 6:285–314, 2002.

[16] R.S.Falk, P.Gatto, and P.Monk. Hexahedral $h(div)$ and $h(curl)$ finite elements. *M2AN*, 2011.

[17] S.Hossain, S.Hossainy, Y.Bazilevs, V.Calo, and T.Hughes. Mathematical modeling of coupled drug and drug-encapsulated nanoparticle transport in patient-specific coronary artery walls. *Comput. Mech.*, 49:213–242, 2012.

[18] K. Støverud, M. Darcis, R. Helmig, and S. Hassanizadeh. Modeling concentration distribution and deformation during convection-enhanced drug delivery into brain tissue. *Transp. Porous Med.*, 92:119–143, 2012.

[19] Shuyu Sun and Jiangguo Liu. A locally conservative finite element method based on piecewise constant enrichment of the continuous galerkin method. *SIAM J. Sci. Comput.*, 31:2528–2548, 2009.

[20] B. L. Vaughan, P. A. Galie, J. P. Stegemann, and J. B. Grotberg. A poroelastic model describing nutrient transport and cell stresses within a cyclically strained collagen hydrogel. *Biophys. J.*, 105:2188–2198, 2013.

[21] Chunmei Wang, Junping Wang, Ruishu Wang, and Ran Zhang. A locking-free weak galerkin finite element method for elasticity problems in the primal formulation. *J. Comput. Appl. Math.*, 307:346–366, 2016.

[22] Junping Wang and Xiu Ye. A weak galerkin finite element method for second order elliptic problems. *J. Comput. Appl. Math.*, 241:103–115, 2013.

[23] Shangyou Zhang. On the nested refinement of quadrilateral and hexahedral finite elements and the affine approximation. *Numer. Math.*, 98:559–579, 2004.

[24] Shangyou Zhang. Numerical integration with taylor truncations for the quadrilateral and hexahedral finite elements. *J. Comput. Appl. Math.*, 205:325–342, 2007.