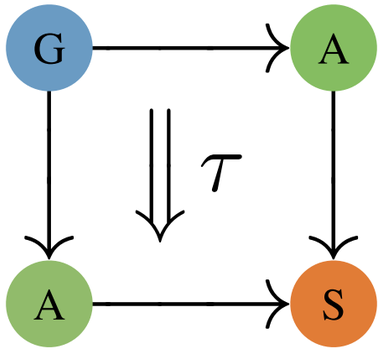
Building Modeling and Simulation Tools on Discrete Exterior Calculus Foundations

IMSI Workshop on DEC, Diff. Geo. and Applications
Chicago, Ill
09/5/2025



UF

James Fairbanks,
w/ Luke Morris,
George Rauta

The Team

TOPOS INSTITUTE

Dr. James Fairbanks

Luke Morris

George Rauta

Matt Cuffaro

Dr. Evan Patterson

Dr. Kevin Carlson

Andrew Baas

Owen Lynch

Georgia Tech

DECAPODES: A Framework for Directly Computable Physics

EC Theory and Operators

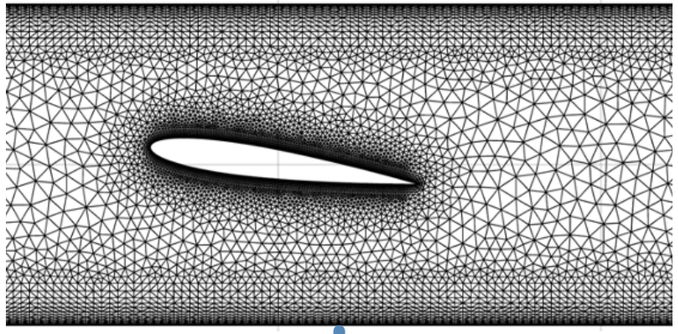
$\Omega_i, \Omega^{\star}_{(n-i)}$ Forms

δ, d, ∂ Topology

$\star, \wedge, \mathcal{L}(u)$ Geometry

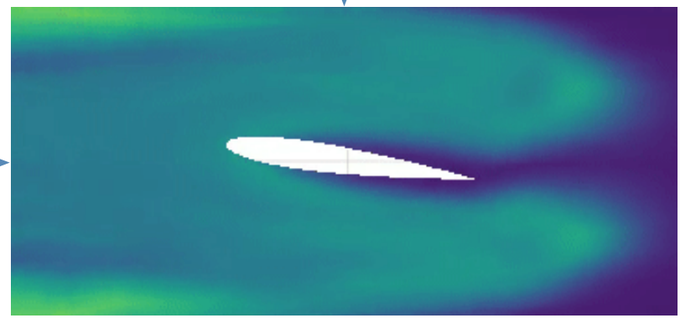
$\nabla, \nabla\cdot, \nabla^2$ Physics

Discrete Geometric Domain



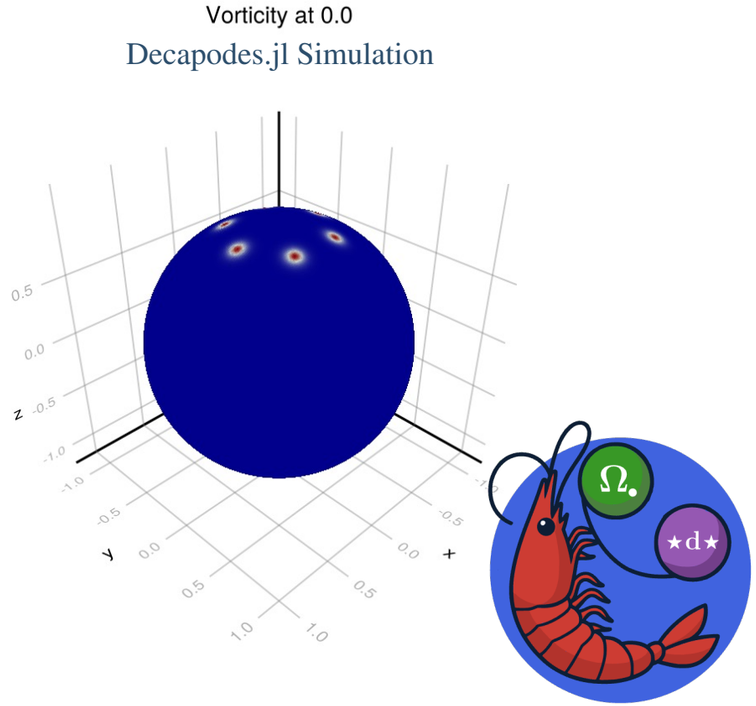
$$\frac{D\rho}{Dt} = -\rho\nabla\cdot\mathbf{u}$$
$$\rho\frac{D\mathbf{u}}{Dt} = -\nabla p + \mu\nabla^2\mathbf{u} + \frac{\mathbf{1}}{\mathbf{3}}\mu\nabla(\nabla\cdot\mathbf{u})$$
$$\rho\frac{De}{Dt} = -p\nabla\cdot\mathbf{u}$$

Physical Equations



Generated Simulation

```
eq11_inviscid_poisson = @decapode begin
  du::DualForm2
  u::DualForm1
  ψ::Form0

  ψ == Δ⁻¹(⋆(du))
  u == ⋆(d(ψ))

  ∂ₜ(du) ==  (-1) * ∘(♭♯, ⋆₁, d̃₁)(∧ᵈᵖ₁₀(u, ⋆(du)))
end
```

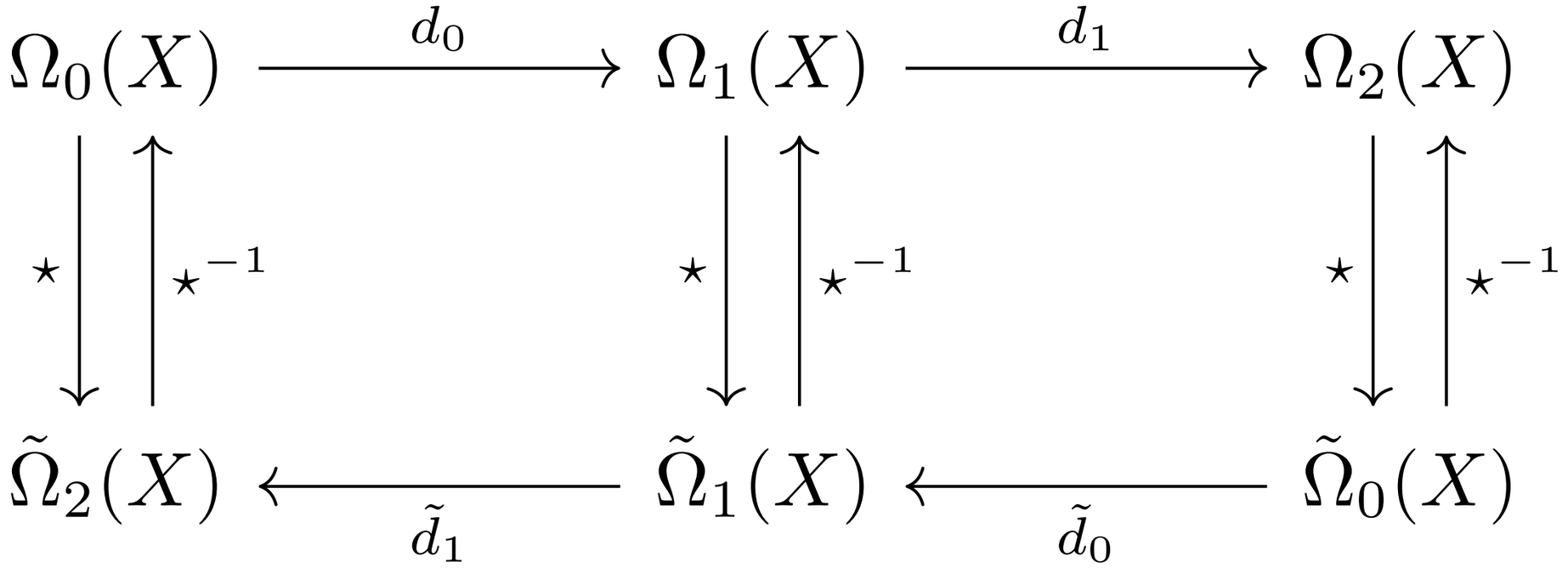


Functorial Semantics in Programming Languages

$$\text{Syntax} \xrightarrow{\text{Compile}} \text{Semantics}$$

- Syntax is combinatorial trees, graphs, hypergraphs
- Semantics is quantitative, functions, relations, shapes, distributions
- Functor from syntax to semantics recursively generates programs
- Structure of the categories/functor determines allowable language operations

DEC de Rahm Complex



- We use an explicit dual mesh
- Subdivision for both dual mesh construction and multigrid

Coverage of DEC

Simplex Operations
- {primal, dual} x {vertex,edge,triangle,tetrahedron} x{vertices, center},
- subdivide_duals!, subdivide x {Circumcenter, Barycenter, Incenter}
- Sign management!

Operators
- d, $\star$, $\star^{-1}$, δ, ∇^2, Δ, Δ^d, $\wedge$, interior_product, $\mathscr{L}$, $\mathscr{L}$_dd,
- $\flat$, $\sharp$, $\flat\sharp$, lie_derivative_flat

Operator Variations
- DiagonalHodge, GeometricHodge,
- PDSharp, PPSharp, AltPPSharp, DesbrunSharp, LLSDDSharp, DPPFlat, PPFlat,

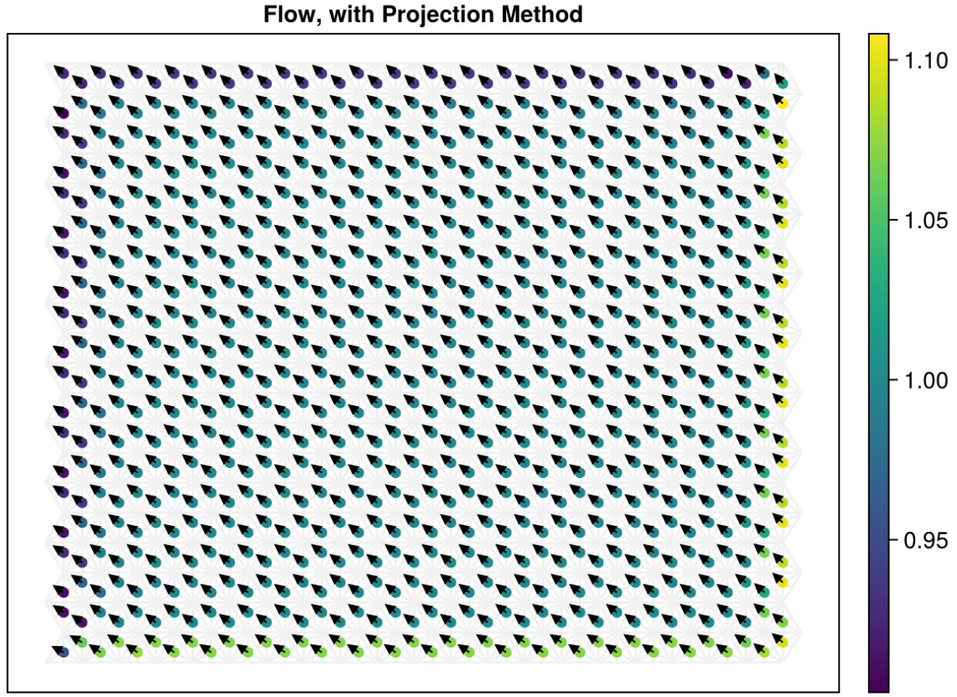
Interpolation and Averaging
- d0_p0_interpolation, p0_d0_interpolation p2_d2_interpolation, p3_d3_interpolation
- avg_{01}, avg_01, avg_{01}_mat, avg_01_mat,

Initialization:
- eval_constant_primal_form, eval_constant_dual_form

What is a sufficient collection of operators for "implementing DEC"?

CombinatorialSpaces.jl as a DEC Library

$$\frac{\partial \mathbf{u}^\flat}{\partial t} = -\mathcal{L}_u \mathbf{u}^\flat + \frac{1}{2}\mathbf{d}(\mathbf{u}^\flat(\mathbf{u})) - \frac{1}{\rho}\mathbf{d}p$$

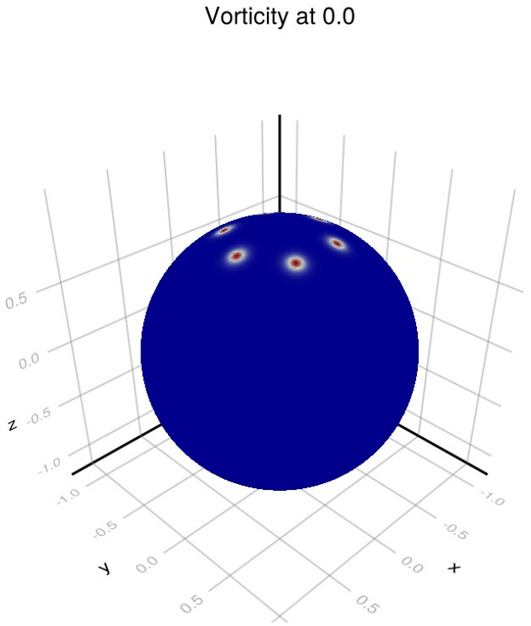


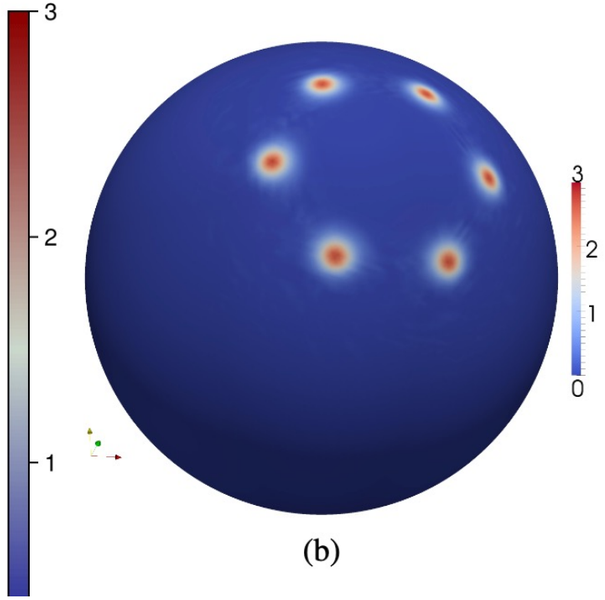
Simulation code uses operators to implement discretization

```
u = s1 * eval_constant_primal_form(sd, X♯)
du = copy(u)

dt = 1e-3
u_int = zeros(ne(sd))
p_next = zeros(ntriangles(sd))

function euler_equation_with_projection!(u)
u_int .= u .+ (- ℒ1(u,u) + 0.5*d0*ι1(u,u))*dt
p_next .= (solveΔ ∘ div)(u_int/dt)
u .= u_int - dt*(d0*p_next)
end

function eulers_method()
for _ in 0:dt:1
euler_equation_with_projection!(u)
end
u
end

eulers_method()

plot_dvf(sd, u, title="Flow, with Projection
Method")
```

Decapodes.jl is a Multiphysics Framework built on DEC

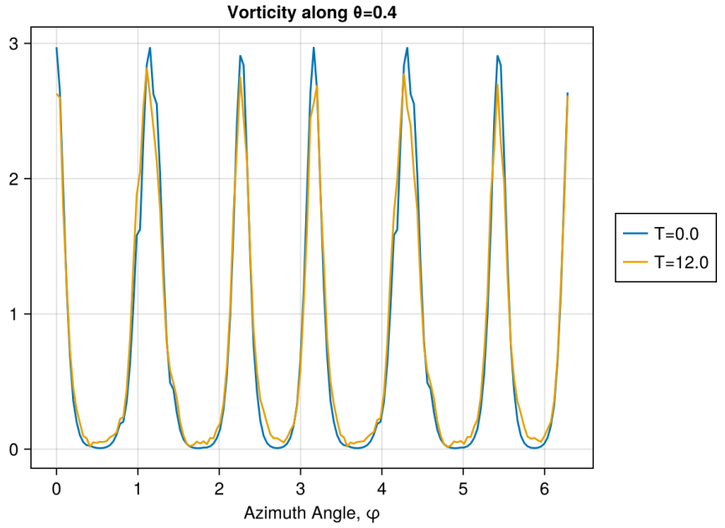
```
eq11_inviscid_poisson = @decapode begin

du::DualForm2
u::DualForm1
ψ::Form0

ψ == Δ⁻¹(⋆(du))
u == ⋆(d(ψ))

∂ₜ(du) == -1∘(♭♯, ⋆₁, d̃₁)(∧ᵈᵖ₁₀(u, ⋆(du)))
end
```

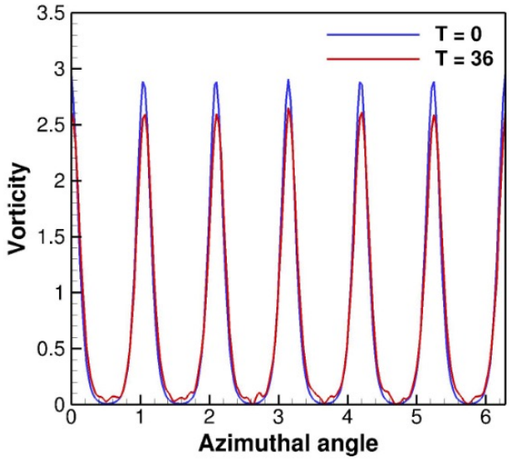




(b)

- Vorticity Formulation of Incomp. Navier Stokes
- Stable solves with nearly periodic solutions
- Replicates the analysis from: Mohamed, Hirani & Samtaney 2016

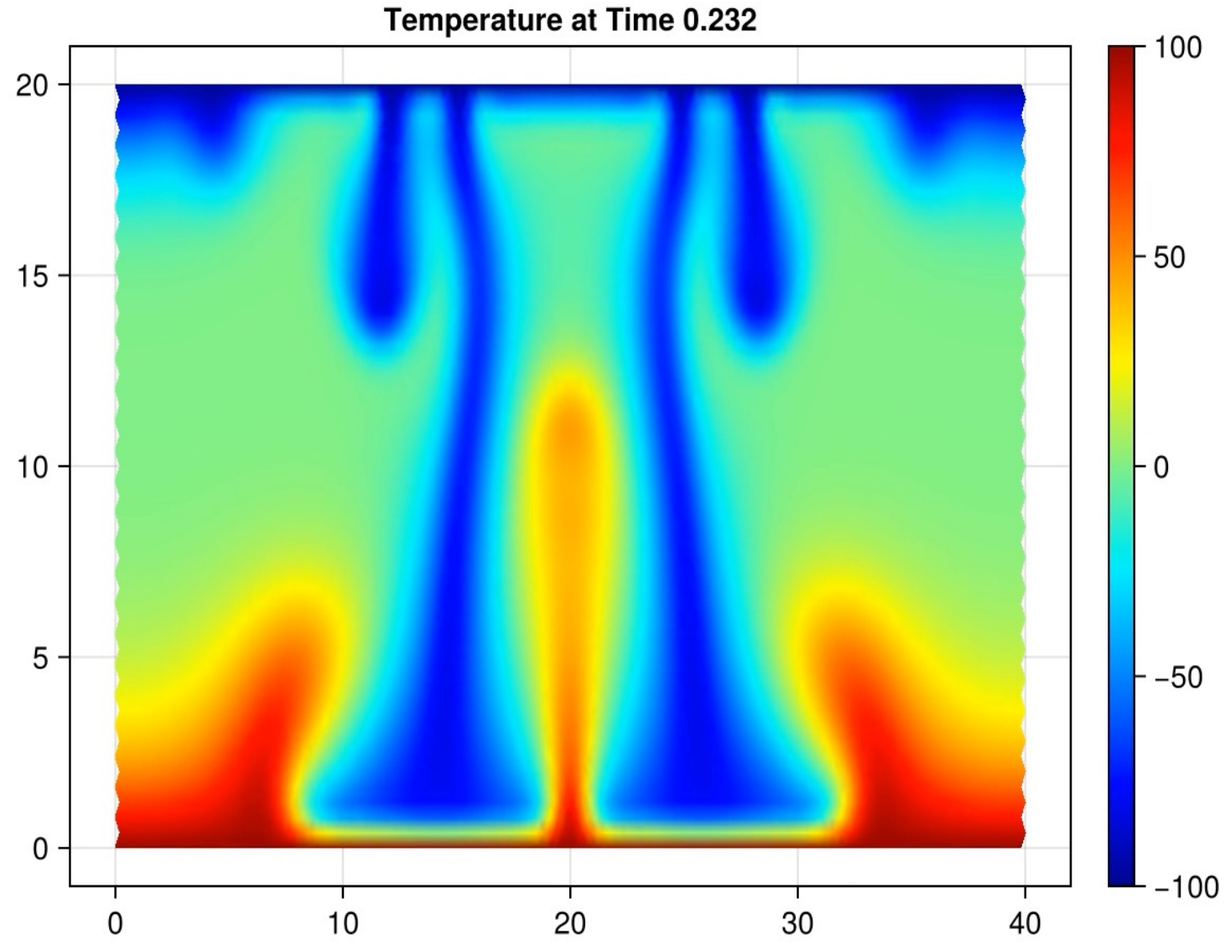




(d)

Porous Convection: Darcy + Boussinesq

```
Porous_Convection = @decapode begin
  (λ_ρ₀Cp, αρ₀, k_ηf, ϕ)::Constant
  (P, T, Adv, bound_T, bound_Ṫ)::Form0
  (g, qD)::Form1

  bound_T == adiabatic(T)

  # Darcy flux
  ρ == g ∧ (αρ₀ ∗ bound_T)
  P == Δ⁻¹(δ(ρ))
  qD == -(k_ηf ∗ (d(P) - ρ))

  Adv == ⋆(interpolate(∧ᵈᵖ₁₁(⋆(d(bound_T)), qD)))
  Ṫ == -1/ϕ ∗ Adv + λ_ρ₀Cp ∗ Δ(bound_T)

  bound_Ṫ == tb_bc(Ṫ)

  ∂ₜ(T) == bound_Ṫ
end
```



- Entire Specification of physics above
- Divergence-free condition on pressure field enforced by pressure projection method
- Top/bottom temperature conditions and adiabatic side conditions

Weakly Compressible Perturbed NS w/ Energy Equation

$$\frac{\partial U}{\partial t} = -U \wedge \delta u - \mathcal{L}_u U + \frac{1}{2}\rho||u||^2 - dP' + \rho' g + \mu \Delta u \tag{1}$$

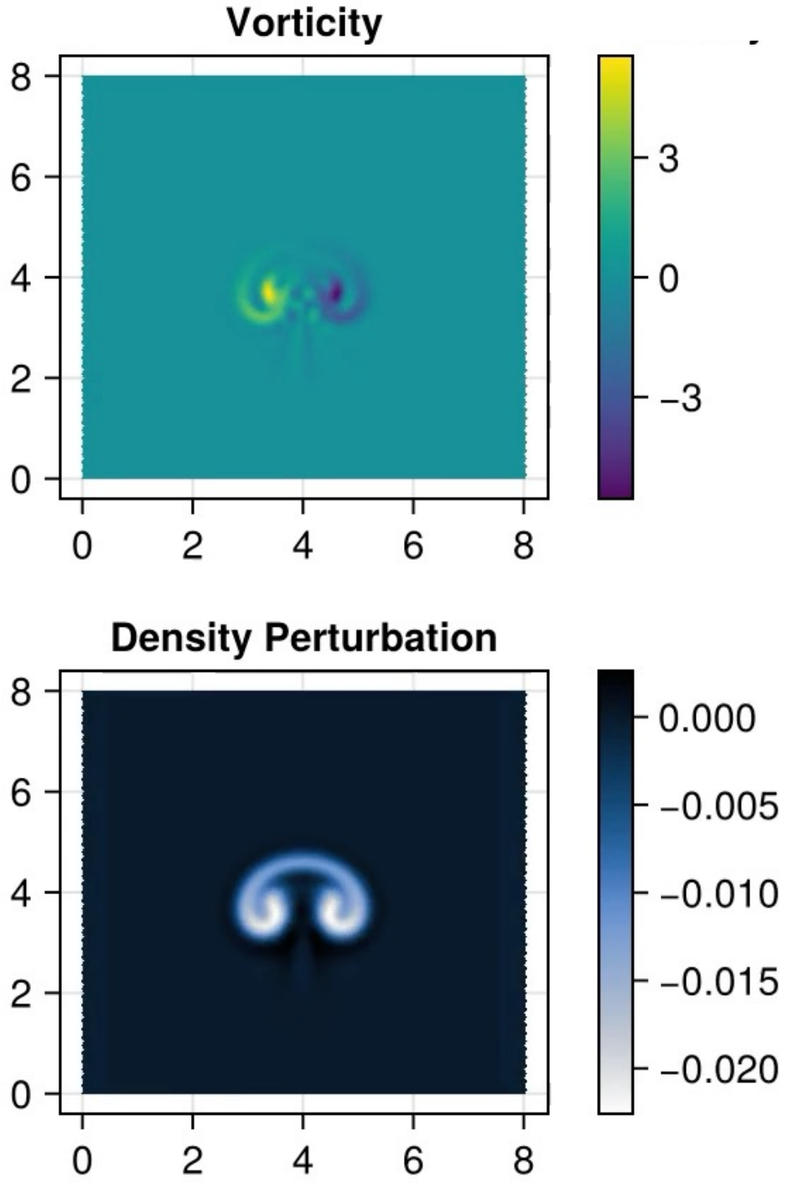
$$\frac{\partial \rho'}{\partial t} = -\delta U \tag{2}$$

$$\frac{\partial \Theta'}{\partial t} = -\Theta \wedge \delta u - \mathcal{L}_u \Theta + \kappa \Delta \theta \tag{3}$$

$$P' \approx \frac{1}{1 - R/C_p} \Theta_H^{\frac{1}{1-R/C_p}-1} \Theta' (R P_0^{-R/C_p})^{\frac{1}{1-R/C_p}} \tag{4}$$

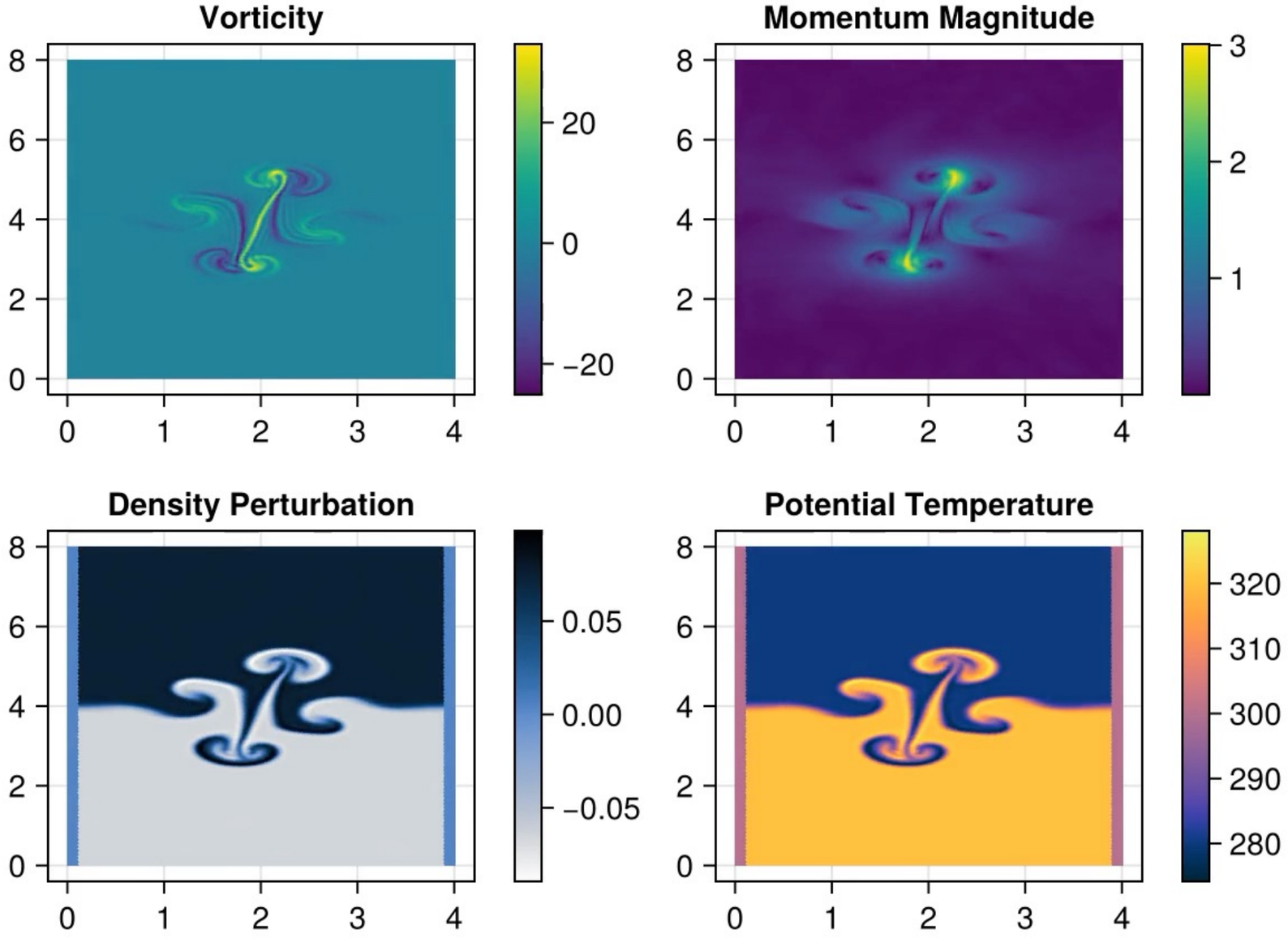
- To form our equations, we assume that:
  - The fluid equation of state is the ideal gas law.
  - The flow is low Mach-number, i.e., weakly compressible.
  - The isentropic heat variable is given by potential temperature, which experiences perturbations of no more than 10% of hydrostatic.
- Our simulated fluid will be dry air.

Rising Thermal Bubble



- Periodic boundary conditions on left and right.
- Gravity is turned on.
- Initial perturbation from stable fluid hydrostatic balance potential temperature ~20K Gaussian bubble.
- Relatively large random perturbations in other variables
- We see the bubble rise due to its lower density.
- Lower density of center causes it to rise quickest.
- Causes void in center which fills with fluid and causes counterrotating vortices.

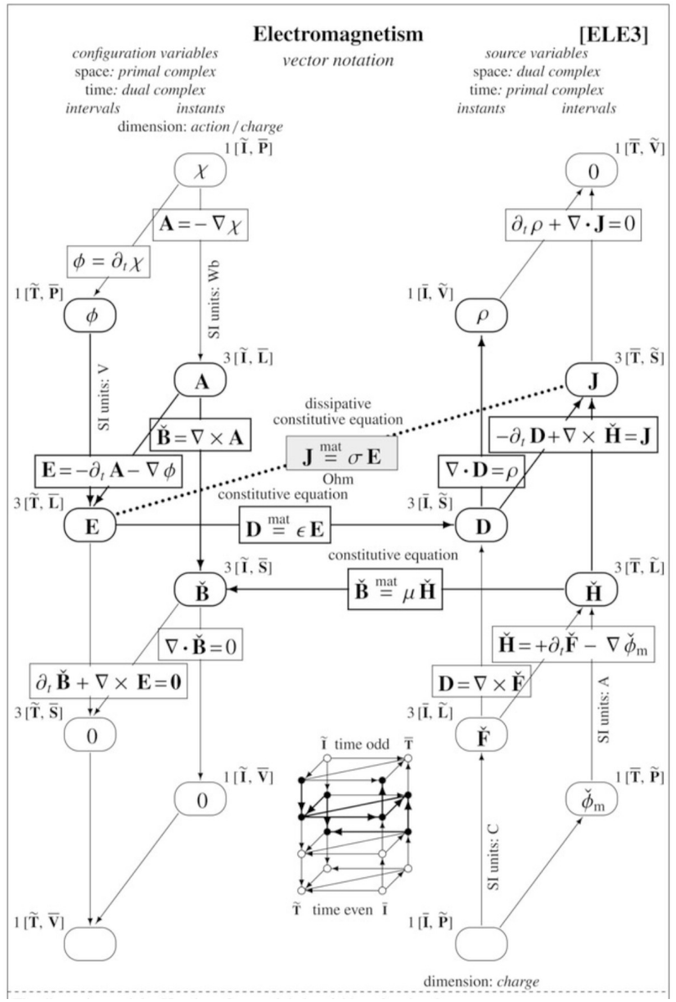
Rayleigh-Taylor Instability (RTI)



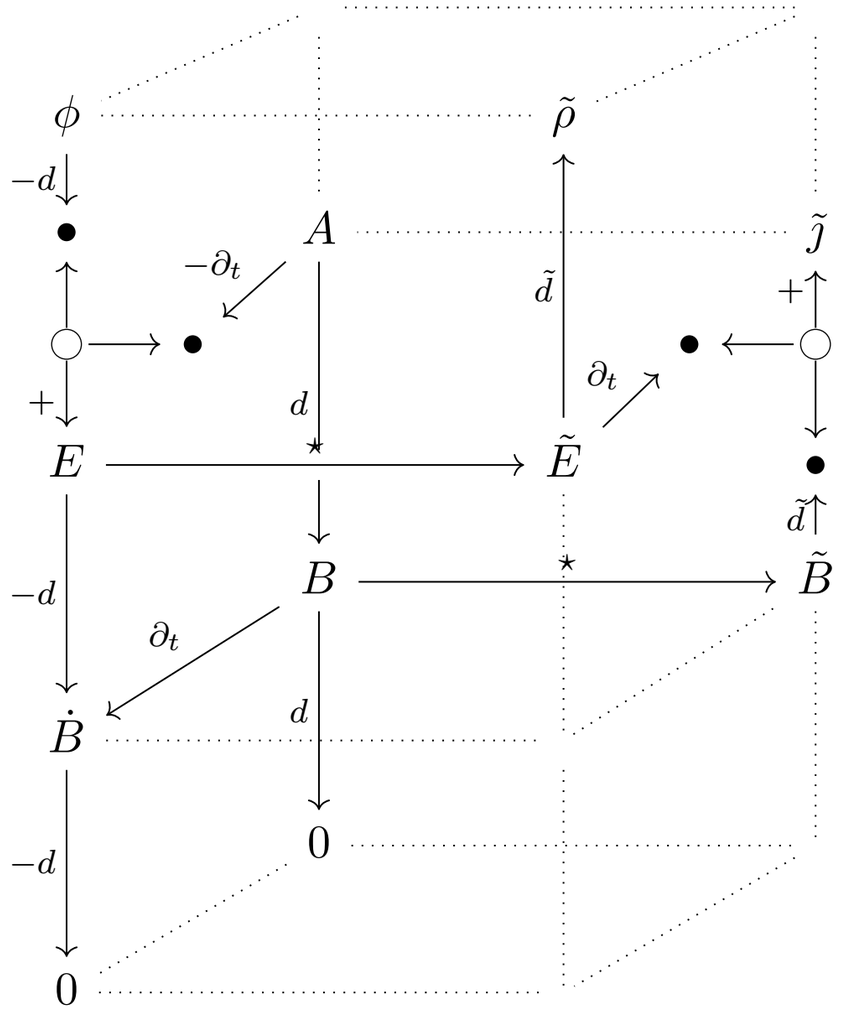
- Periodic boundary conditions on left and right
- Gravity is turned on
- Initial perturbation from stable fluid hydrostatic balance is from momentum.
- Better treatment of initial conditions and harder driver could lead to Kelvin-Hemholtz instabilities.

Motivation: Tonti Diagrams

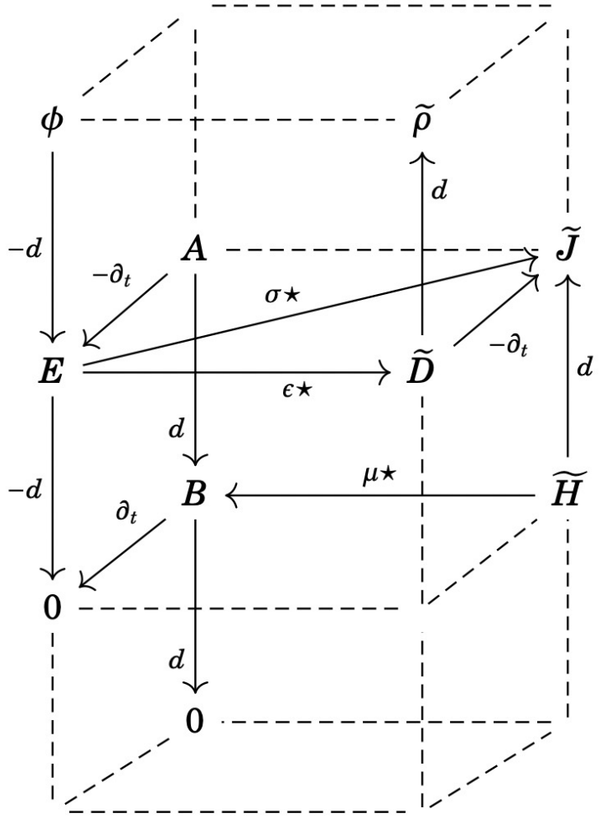
Tonti Diagram for E&M



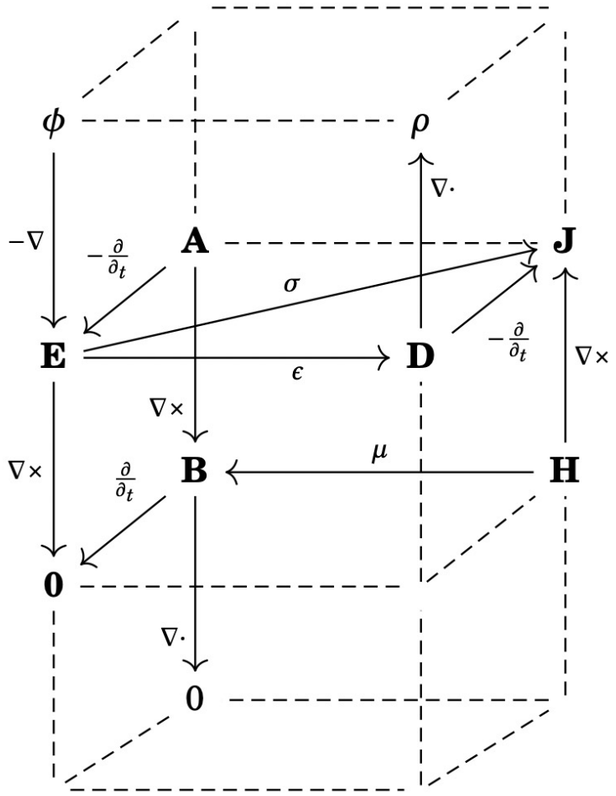
DECAPODE for Electricity and Magnetism



Tonti Diagrams Problems



(a) Maxwell's house in exterior calculus



(b) Maxwell's house in vector calculus

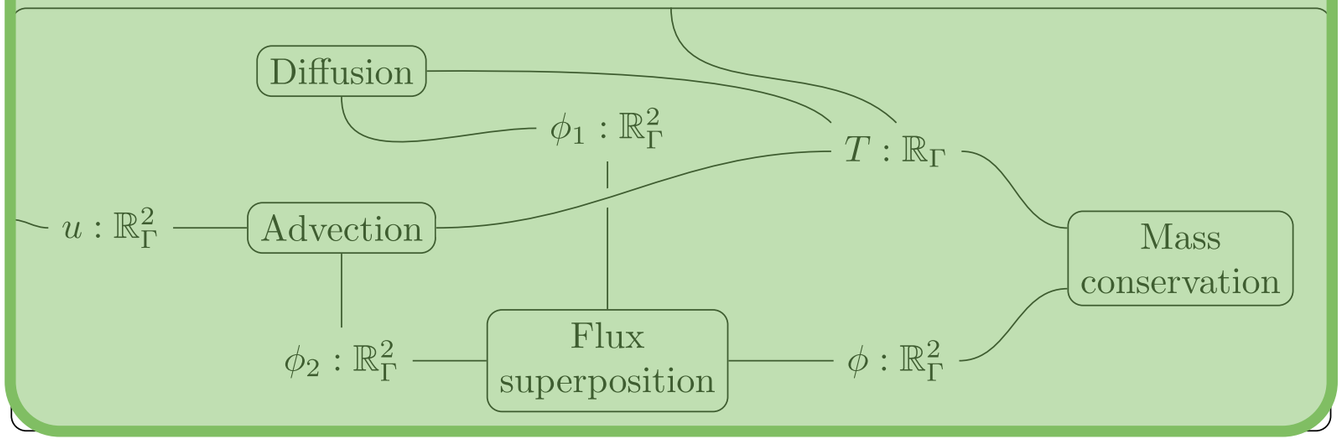
Figure 1.1: Traditional drawing of "Maxwell's house," depicting Maxwell's equations for electromagnetism in matter. The formulation in exterior calculus (a) is adapted from Bossavit [**bossavit1998d**] and the formulation in vector calculus (b) from Cortes Garcia et al [**garcia2018**]. Further variations on Maxwell's house in the literature include [**tonti2013**], [**deschamps1981**], and [**lacomba1991**].

Problems to solve:

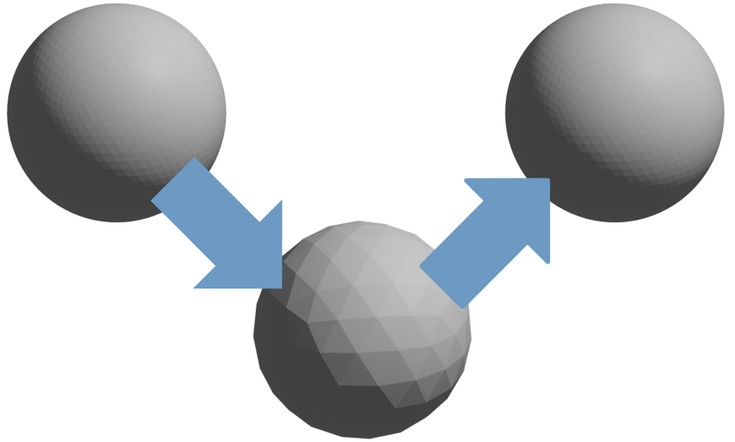
1. Saturation
2. Discrete vs Smooth framework
3. Privileged role of addition
4. Multivariate operations, e.g. wedge prod. & Lie derivative
5. Compositionality
6. Expressivity: must express diffusion, waves, EM, N-S
7. Boundary conditions

M3P: Multidomain, Multiscale, Multiphysics

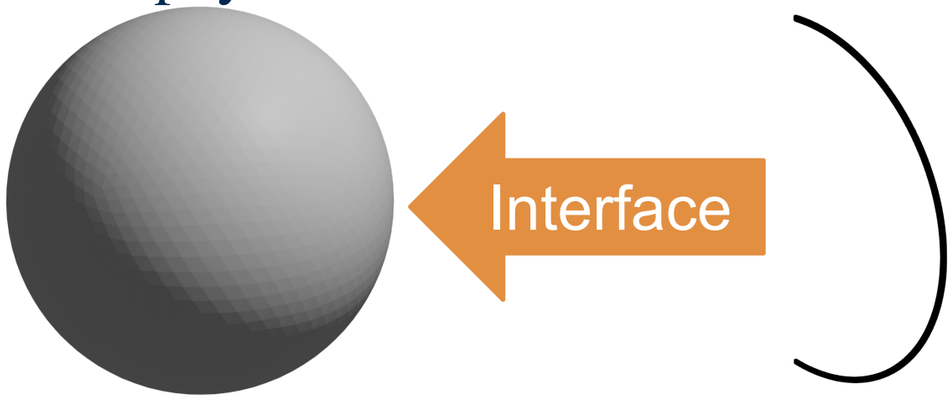
Multiphysics: composing physical laws



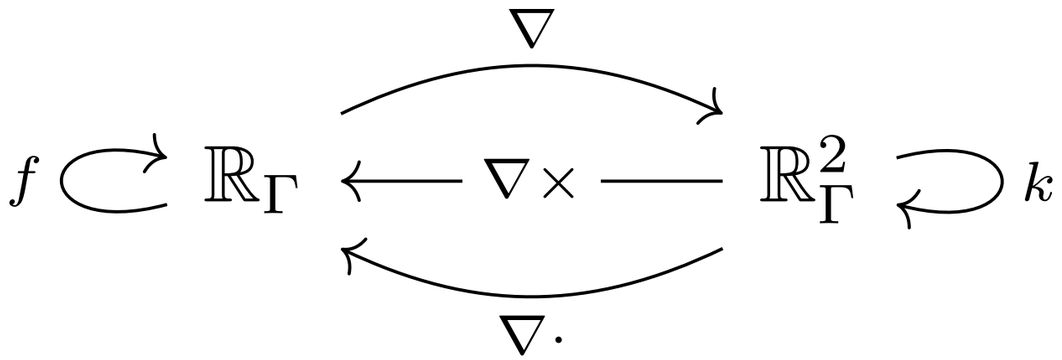
Multiscale: need multiple grid with different resolutions



Multidomain: complex scenarios need different physics on different domains



Vector Calculus Diagrams



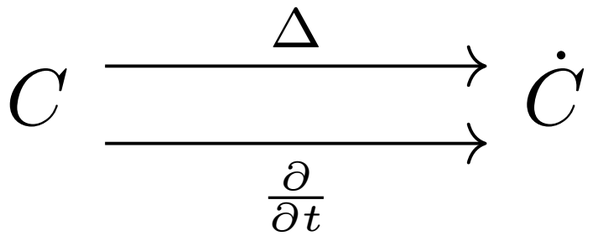
- Scalar and vector fields are the objects
- Differential operators Div, Grad, Curl are arrows
- Allow functions (f/k) on scalar/vector fields too

A diagrammatic view of differential equations in physics

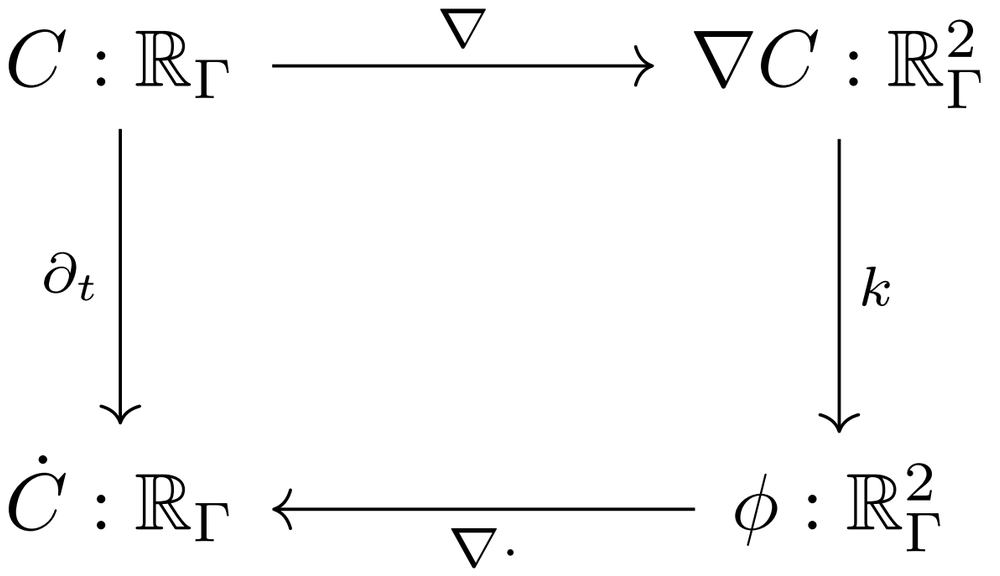
Evan Patterson[1,*], Andrew Baas[2], Timothy Hosgood[1] and James Fairbanks[3]

[1] Topos Institute, Berkeley, CA, USA
[2] Georgia Tech Research Institute, Atlanta, GA, USA
[3] University of Florida, Gainesville, FL, USA

Diffusion: Laplacian Formulation



Diffusion: Fick's Law Formulation



Specifying Diagrammatic Equations in Sheaves

- $\mathfrak{X}_M \in \mathrm{Sh}_{\mathbb{R}}(M)$: smooth vector fields on M
- $\mathfrak{X}_{t,M} \in \mathrm{Sh}_{\mathbb{R}}(M \times [0,\infty))$: smooth *time-dependent* vector fields on M
- $\Omega^k_M \in \mathrm{Sh}_{\mathbb{R}}(M)$, for $0 \leq k \leq m$: differential k-forms on M
- $\Omega^k_{t,M} \in \mathrm{Sh}_{\mathbb{R}}(M \times [0,\infty))$, for $0 \leq k \leq m$: *time-dependent* differential k-forms on M (i.e., the k-forms on $M \times [0,\infty)$ not involving the 1-form dt)
- $\widetilde{\Omega}^k_M \in \mathrm{Sh}_{\mathbb{R}}(M)$ and $\widetilde{\Omega}^k_{t,M} \in \mathrm{Sh}_{\mathbb{R}}(M \times [0,\infty))$, for $0 \leq k \leq m$: *twisted* (time-independent and

1. Fix a category C, and a Space M
2. Choose F, a sheaf on C
3. Take the Category of Elements Construction of F
4. Draw a diagram in C

$$C : Cat, M : Space$$

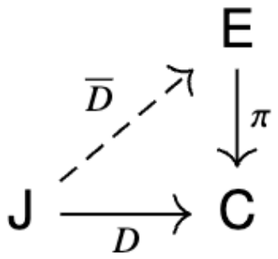
$$F : C \to Sh(M)$$

$$E \xrightarrow{\pi} C$$

$$J \xrightarrow{D} C$$

Solving Equations is Lifting

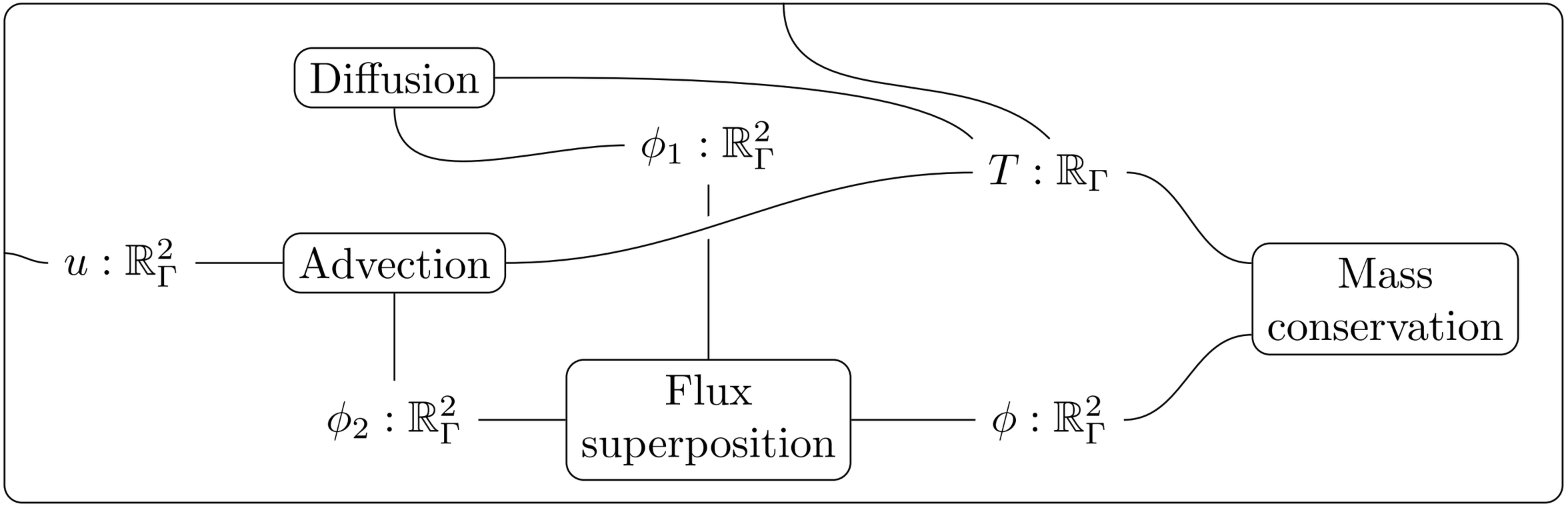
Definition 2.3 (Lifting problem)**.** A *lift* of a diagram $D : \mathsf{J} \to \mathsf{C}$ through a functor $\pi : \mathsf{E} \to \mathsf{C}$ is a diagram $\overline{D} : \mathsf{J} \to \mathsf{E}$ such that $\pi \circ \overline{D} = D$. Given a diagram D in C and a functor $\pi : \mathsf{E} \to \mathsf{C}$, the *lifting problem* is to find a lift of D through π.



$$E : \Omega^1_t \xrightarrow{-d} \dot{B} : \Omega^2_t$$
$$\uparrow \partial_t$$
$$B : \Omega^2_t \xrightarrow{d} \Omega^3_t \longleftarrow 0$$

- Functorial Semantics: Send each equation to the set of lifts
- Each lift is a specific solution
- Why not take limits to talk about solution spaces?
- Solutions spaces aren't differential forms, they are sets of differential forms.

Building Multiphysics with Wiring Diagrams



- Junctions are Variable : Domain
- Boxes are subsystem (component physics)
- Ports are exposed variables in the subsystems
- Wires connect ports to junctions with matching domains
- Outer Ports allow hierarchy by exposing variables of the composite system to next level of hierarchy

Physical Principles in Vector Calculus

Fick's Law of Diffusion

$$C : \mathbb{R}_\Gamma \xrightarrow{k\nabla} \phi : \mathbb{R}^2_\Gamma$$

Scalar Transport by Advection

$$\begin{array}{ccc} V : \mathbb{R}^2_\Gamma & & C : \mathbb{R}_\Gamma \\ \uparrow{\scriptstyle \pi_2} & \nearrow{\scriptstyle \pi_1} & \\ (C \otimes V) : \mathbb{R}_\Gamma \otimes \mathbb{R}^2_\Gamma & \xrightarrow[\wedge]{} & \phi : \mathbb{R}^2_\Gamma \end{array}$$

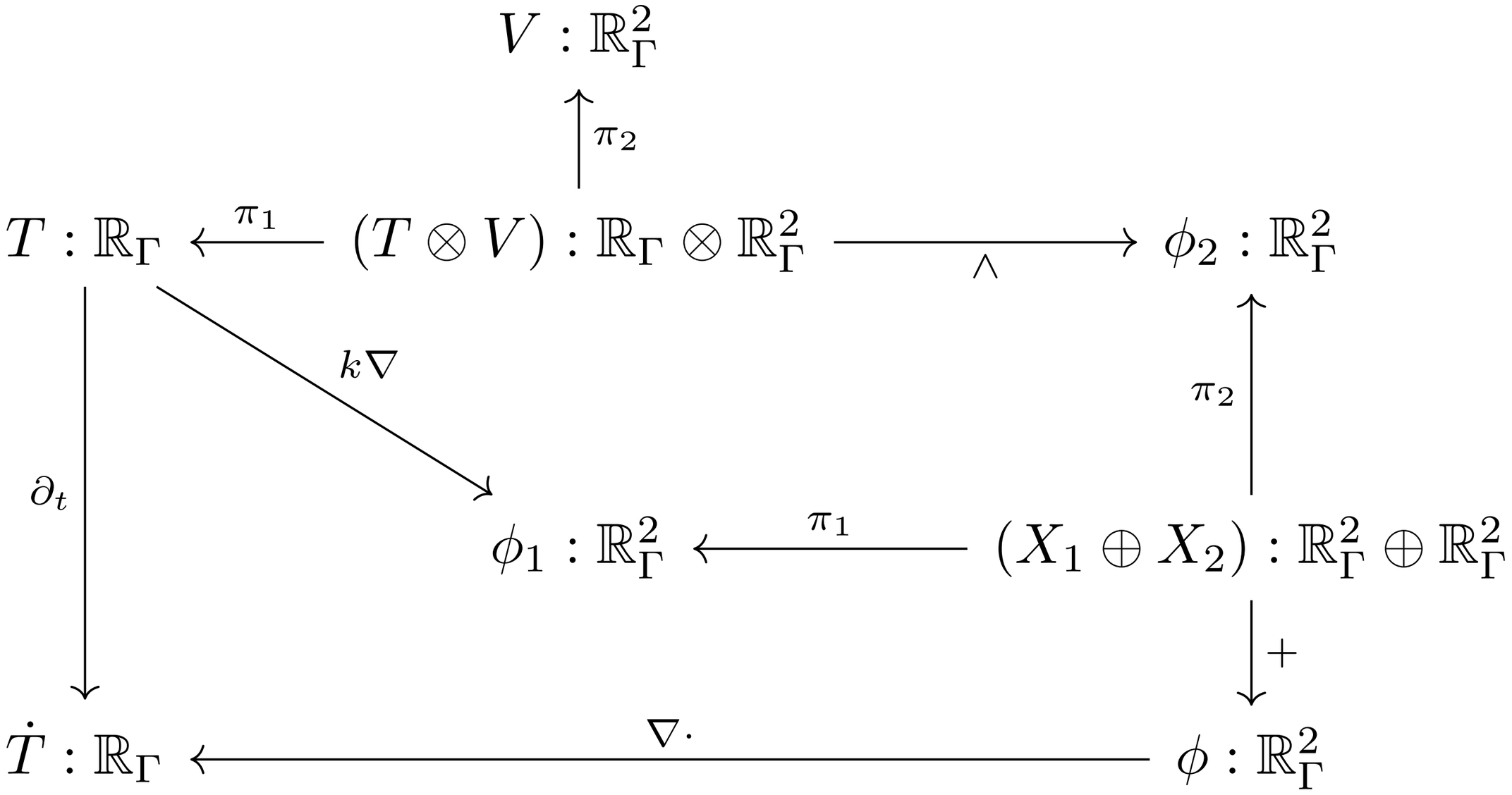
Conservation of Mass

$$\begin{array}{ccc} X : \mathbb{R}_\Gamma & & \\ \downarrow{\scriptstyle \partial_t} & & \\ \dot{X} : \mathbb{R}_\Gamma & \xleftarrow{\nabla\cdot} & \phi : \mathbb{R}^2_\Gamma \end{array}$$

Superposition of Flux

$$\begin{array}{ccc} X : \mathbb{R}^2_\Gamma & & X_2 : \mathbb{R}^2_\Gamma \\ \uparrow{\scriptstyle \pi_1} & \nearrow{\scriptstyle \pi_2} & \\ (X_1 \oplus X_2) : \mathbb{R}^2_\Gamma \oplus \mathbb{R}^2_\Gamma & \xrightarrow[+]{} & X : \mathbb{R}^2_\Gamma \end{array}$$

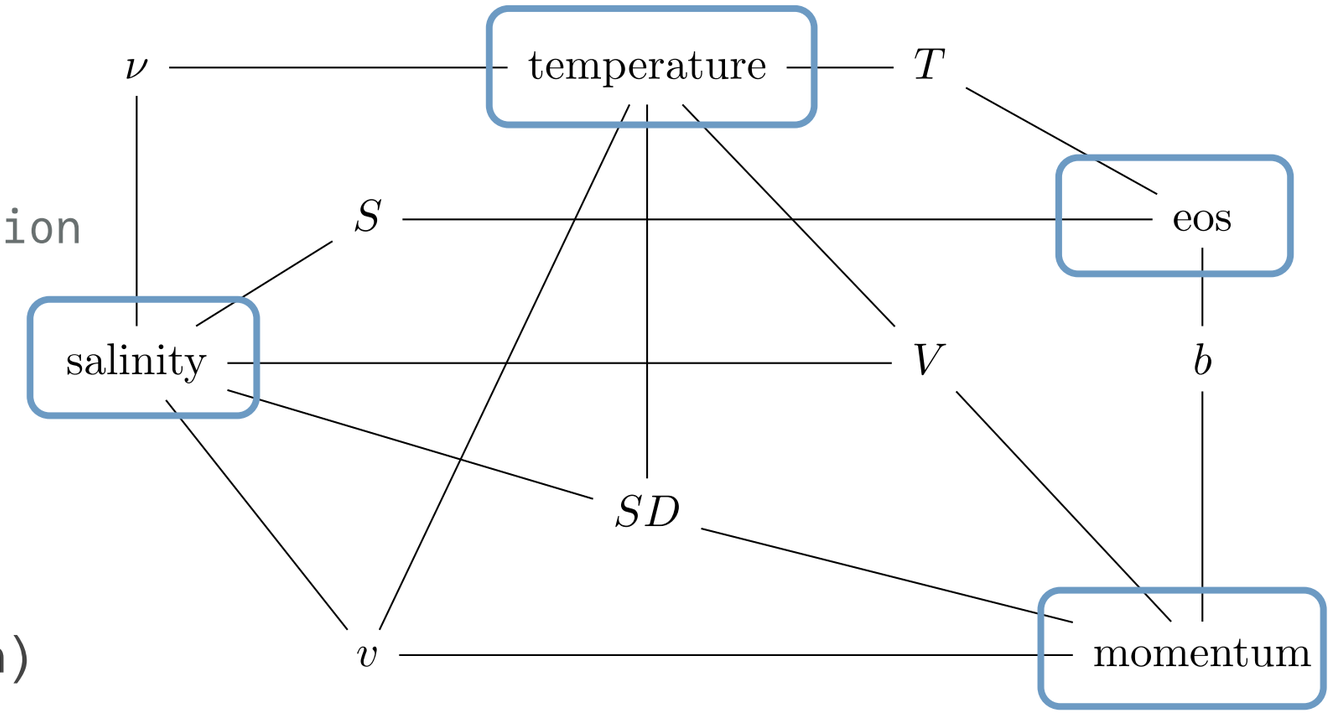
Advection Diffusion Multiphysics Model



Realistic Example taken from Oceananigans.jl

This is the Clima modeling effort at Caltech to build next generation climate models from scratch using modern approaches and tools

```
nonhydrostatic_composition = @relation () begin
  # "The turbulence closure selected by the
  user determines the form of stress
  divergence"
  momentum(V, v, b, SD)
  # "Both T and S obey the tracer conservation
  equation”
  temperature(V, v, T, SD, nu)
  salinity(V, v, S, SD, nu)
  # "Buoyancy is determined from a linear
  equation of state"
  eos(b, T, S)
end
draw_composition(nonhydrostatic_composition)
```



Substitution of Subsystems

```
momentum = @decapode
  (v,V)::DualForm1
  f::Form0
  (uˢ, ∂tuˢ)::DualForm1
  (p,b)::DualForm0
  (ĝ, Fᵥ, StrDiv) ::DualForm1
```

```
∂t(v) == -ℒ1(v,v)
         + 0.5*d(l11(v,v))
         - d(l11(v,V))
         + l12(v,d(V))
         + l12(V,d(v))
         - (f - ∘(d,⋆)(uˢ))∧ᵖᵈ01 v
         - d(p) + b ∧ᵈᵈ01 ĝ - StrDiv
         + ∂tuˢ + Fᵥ
```

```
tracer_conservation = @decapode
  (c,C,F)::DualForm0
  FlxDiv::DualForm0
  (v,V)::DualForm1
```

```
∂t(c) == -1*l11(v,d(c))
         - l11(V,d(c))
         - l11(v,d(C))
         - FlxDiv + F
```

Implementing DEC vs Simulation Ecosystem

CombinatorialSpaces.jl

```
u = s1 * eval_constant_primal_form(sd, X♯)
du = copy(u)

dt = 1e-3
u_int = zeros(ne(sd))
p_next = zeros(ntriangles(sd))

function euler_equation_with_projection!(u)
u_int .= u .+ (- ℒ1(u,u) + 0.5*d0*i1(u,u))*dt
p_next .= (solveΔ ∘ div)(u_int/dt)
u .= u_int - dt*(d0*p_next)
end

function eulers_method()
for _ in 0:dt:1
euler_equation_with_projection!(u)
end
u
end

eulers_method()

plot_dvf(sd, u, title="Flow, with Projection
Method")
```

is a DEC Library

Decapodes.jl

```
Porous_Convection = @decapode begin
  (λ_ρ₀Cp, αρ₀, k_ηf, ϕ)::Constant
  (P, T, Adv, bound_T, bound_Ṫ)::Form0
  (g, qD)::Form1

  bound_T == adiabatic(T)
  # Darcy flux
  ρ == g ∧ (αρ₀ * bound_T)
  P == Δ⁻¹(δ(ρ))
  qD == -k_ηf * (d(P) - ρ)


  Adv == ⋆(interpolate(∧ᵈᵖ₁₁(⋆(d(bound_T)),
  qD)))
  Ṫ == -1/ϕ * Adv + λ_ρ₀Cp * Δ(bound_T)


  bound_Ṫ == tb_bc(Ṫ)
  ∂ₜ(T) == bound_Ṫ
end

infer_types!(Porous_Convection)
resolve_overloads!(Porous_Convection)
```

is a DEC Simulator **Framework**

Type Checking for Geometric Corretness

```
Poise = @decapode begin
  P::Form0
  q::Form1
  (R, μ )::Constant

  # d ⋆ of q for the viscous effect
  dq == d(⋆q)
  # Gradient of P for the pressure driving force
  ∇P == d(P)

  # Definition of the time derivative of q
  ∂ₜ(q) == q̇

  # The core equation
  q̇ == μ ∗ ∂q(Δq) + ∇P + R ∗ dq
end

Poise = expand_operators(Poise)
infer_types!(Poise)
```

- This code will fail
- d(⋆q) is a dual 0-form
- ∇P is a 1-form
- ∇P + R ∗ dq is a ?-form

Type Checking for Geometric Corretness

```
Poise = @decapode begin
  P::Form0
  q::Form1
  (R, μ )::Constant

  # Laplacian of q for the viscous effect
  Δq == Δ(q)
  # Gradient of P for the pressure driving force
  ∇P == d(P)

  # Definition of the time derivative of q
  ∂ₜ(q) == q̇

  # The core equation
  q̇ == μ ⋆ ∂q(Δq) + ∇P + R ⋆ Δq
end

Poise = expand_operators(Poise)
infer_types!(Poise)
```

- This code will succeed
- $\Delta(q)$ is a 1-form
- ∇P is a 1-form
- $\nabla P + R \star dq$ is a 1-form

Algorithmic Rewriting to Improve Stability

- Decapodes doesn't know any physics or fancy CFD tricks
- Just Vanilla DEC + General Purpose Compute Graphs
- Representation is flexible enough to capture equational rewrites

Naïve Formulation

```
eq11_incorrect = @decapode begin
du::DualForm2
u::DualForm1

u == d₁⁻¹(du)
∂ₜ(du) == -1 ∘(♭♯, ⋆₁, d̃₁)(∧ᵈᵖ₁₀(u, ⋆(du)))
end
```

UNSTABLE

Pressure Projection Formulation

```
eq11_inviscid_poisson = @decapode begin
du::DualForm2
u::DualForm1
ψ::Form0

ψ == Δ⁻¹(⋆(du))
u == ⋆(d(ψ))

∂ₜ(du) == -1∘(♭♯, ⋆₁, d̃₁)(∧ᵈᵖ₁₀(u, ⋆(du)))
end
```

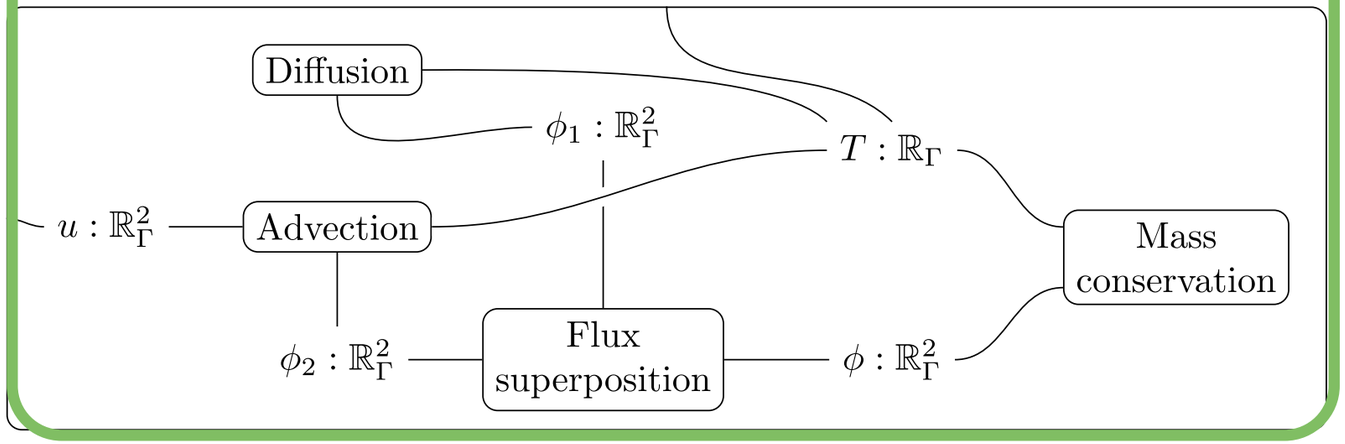
STABLE

Explicit Representation of the Model Allows

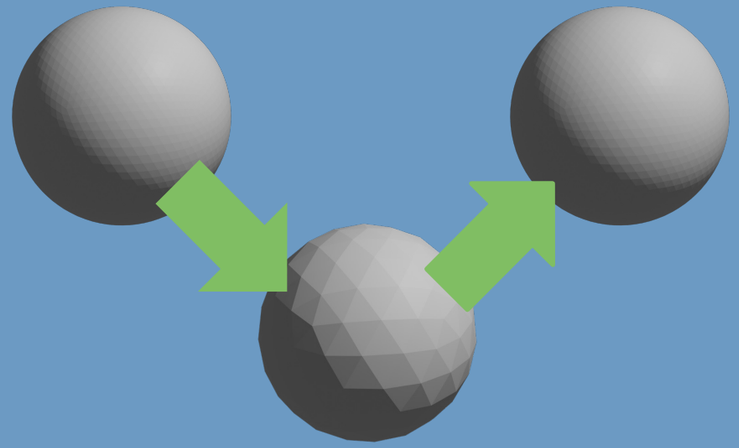
- Computer Algebra
  - Memory optimization
  - Canceling terms
  - Precomputing composite operators
- Type checking of formulas
  - Catching bugs
  - Calculating memory requirements
- Graph algorithms
  - Dependency checking with graph traversal
  - Parallel scheduling within a pode with DAG
- Compositionality
  - Assembling multiphysics models with multiple stakeholders
  - Ensuring compatibility between submodels
  - Enforcing abstraction boundaries

M3P: Multidomain, Multiscale, Multiphysics

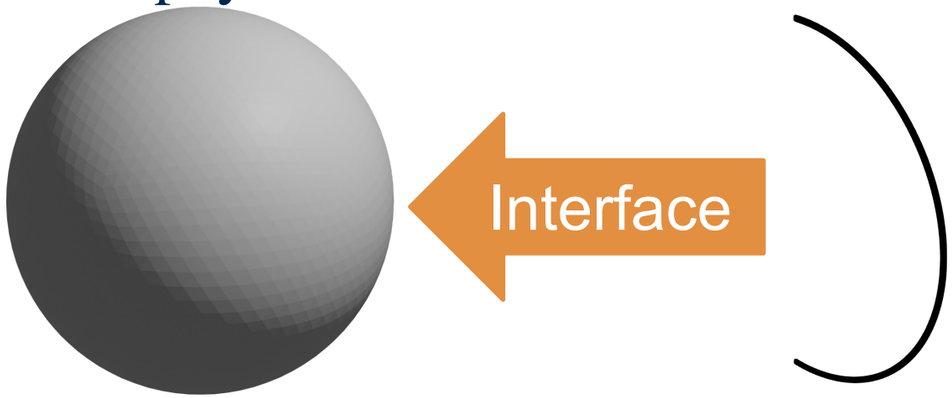
Multiphysics: composing physical laws



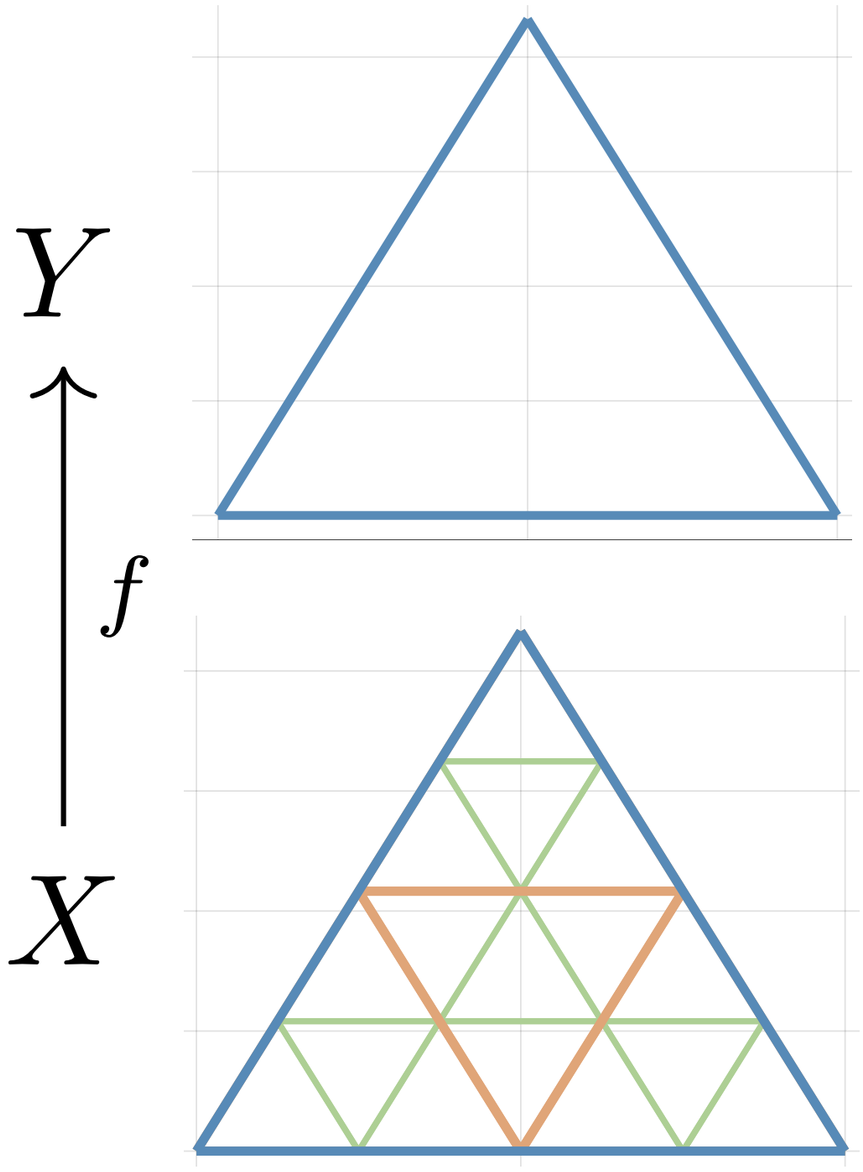
Multiscale: need multiple grid with different resolutions



Multidomain: complex scenarios need different physics on different domains



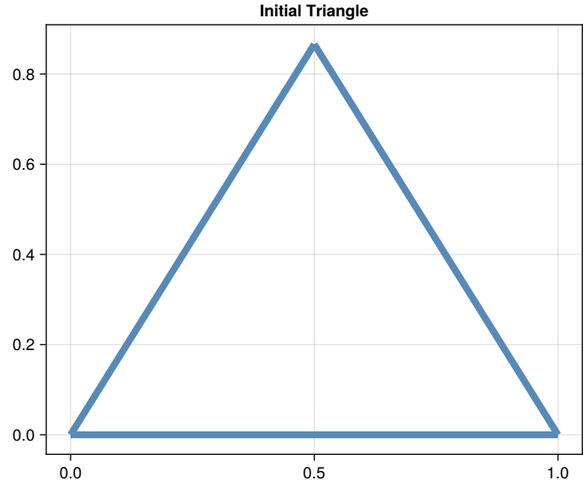
Functoriality of De Rahm Complex

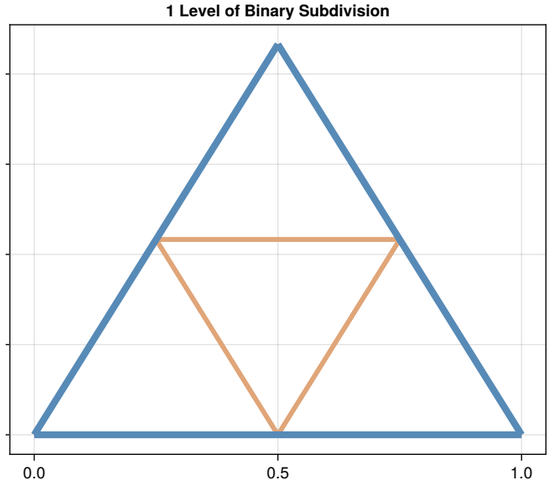


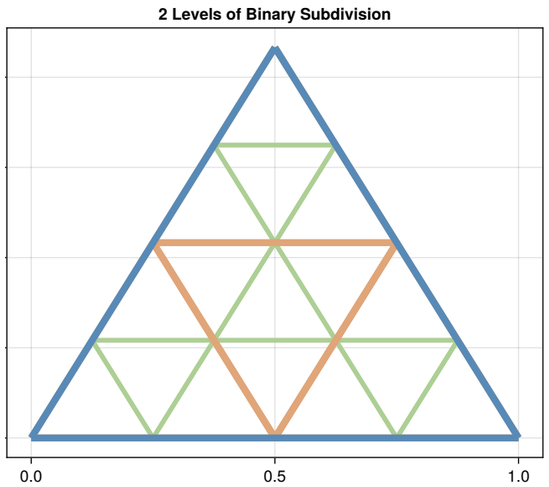
$$\begin{array}{ccccc} \Omega_0(X) & \xrightarrow{d} & \Omega_1(X) & \xrightarrow{d} & \Omega_2(X) \\ \downarrow f_\star \uparrow f^\star & & \downarrow f_\star \uparrow f^\star & & \downarrow f_\star \uparrow f^\star \\ \Omega_0(Y) & \xrightarrow{d} & \Omega_1(Y) & \xrightarrow{d} & \Omega_2(Y) \end{array}$$

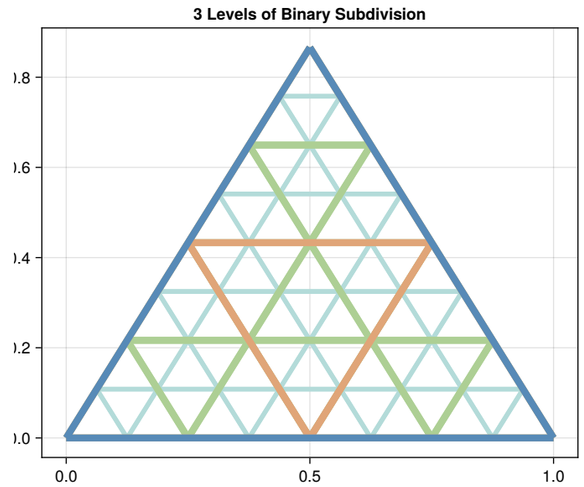
- Continuous maps between spaces induce pushforwards and pullbacks of forms
- Discretized: Geometric Morphisms of Simplicial Sets induce prolongation and relaxation linear maps
- Multigrid!

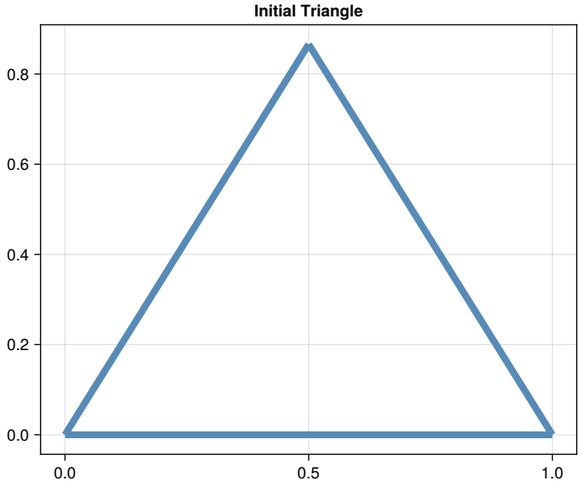
Geometric Multigrid Using Generic Subdivision

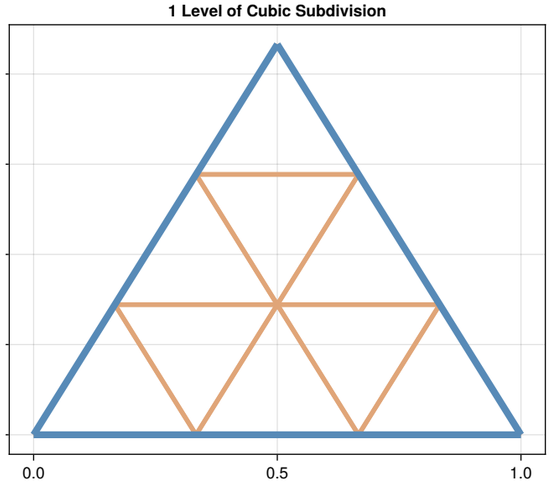


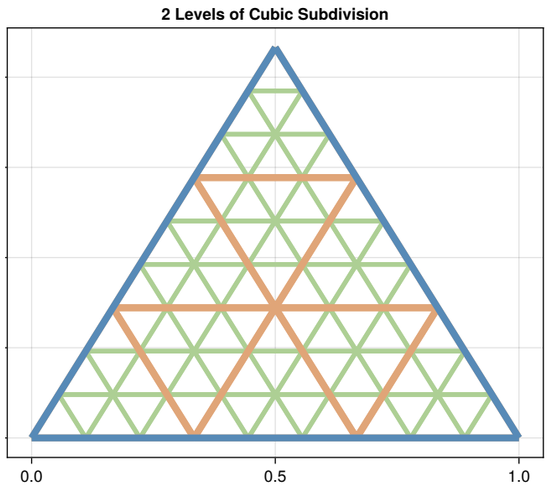


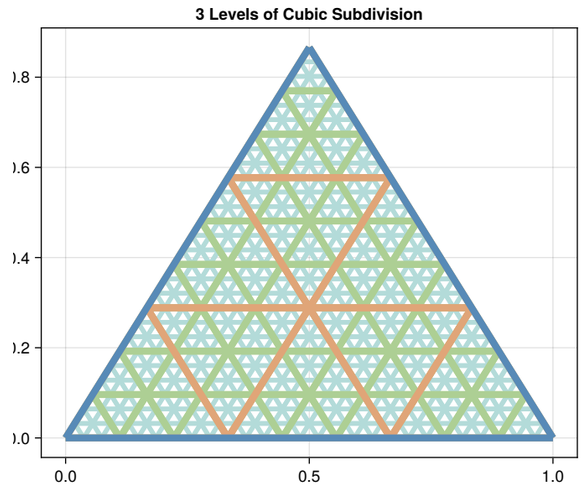






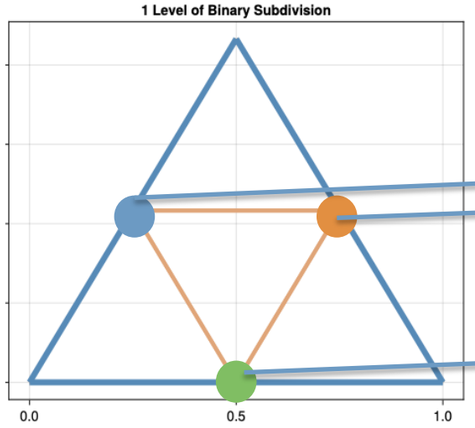




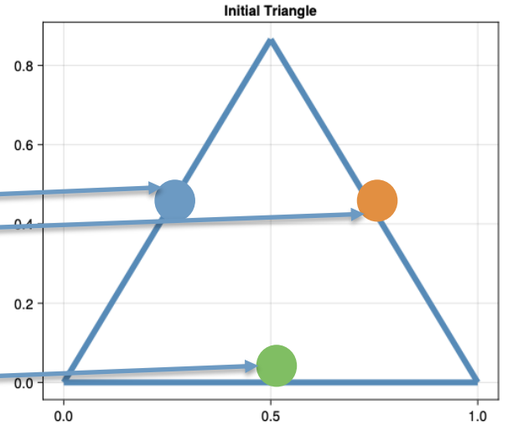


Functorial interpolation example

Matrix Encoding of Barycentric Coordinates

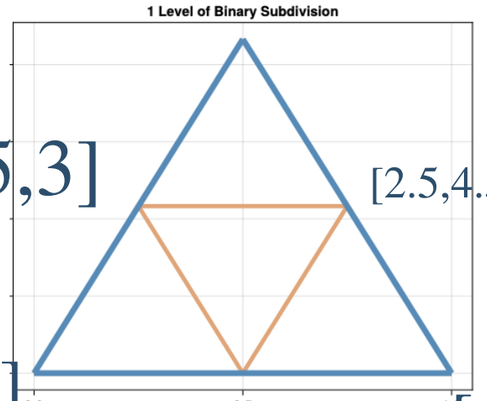


$$\begin{bmatrix} 1 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 1 & 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 1 & 0 & 1/2 & 1/2 \end{bmatrix}$$



Matrix Encoding of induced map on de Rham complex

[7,2]



[5,3]

[2.5,4.5]

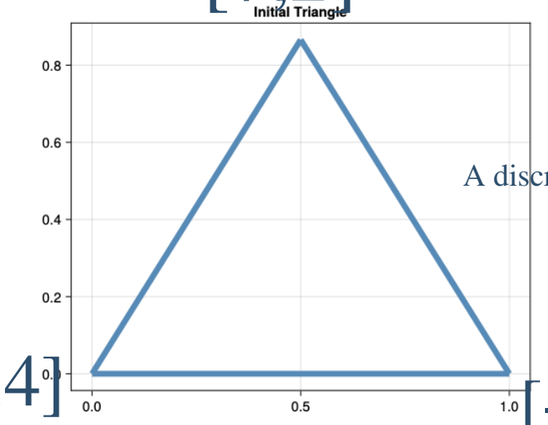
[3,4]

[0.5,5.5]

[-2,7]

$$\begin{bmatrix} 1 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 1 & 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 1 & 0 & 1/2 & 1/2 \end{bmatrix}$$

[7,2]



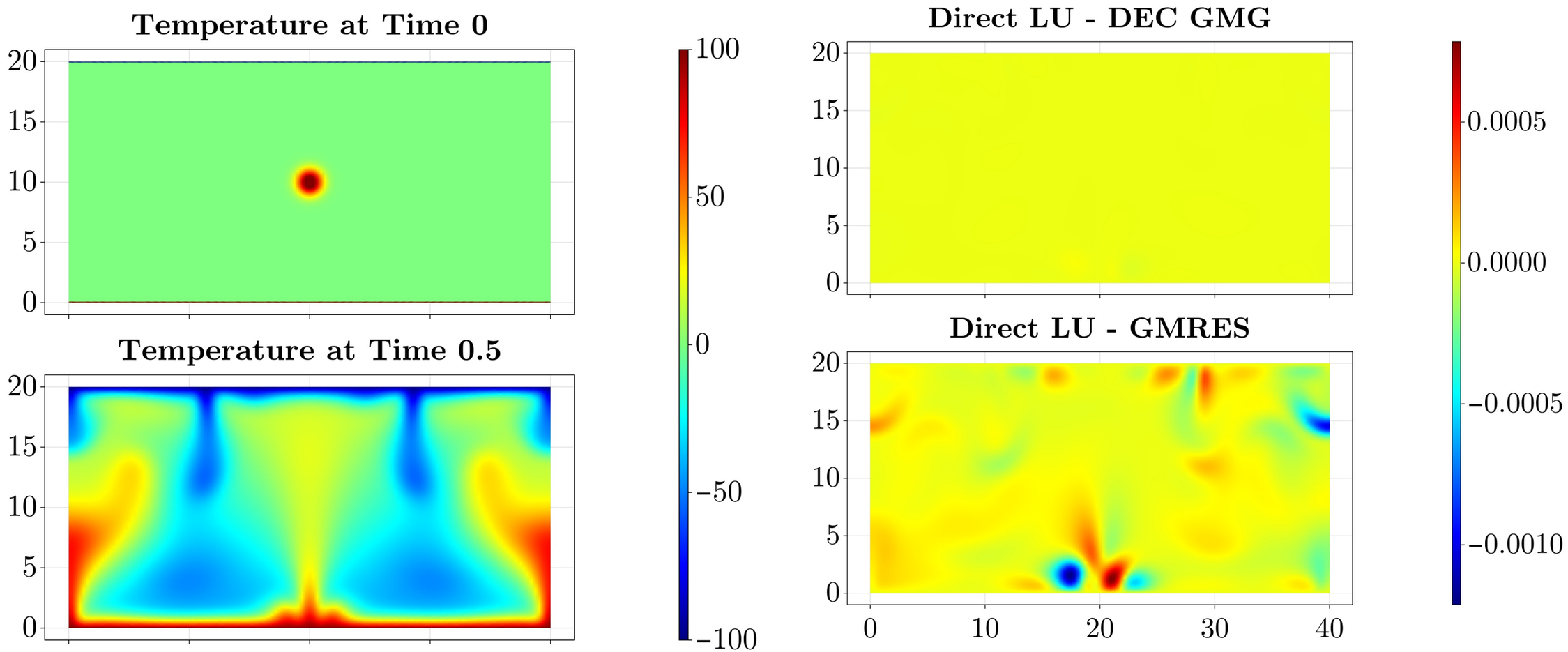
A discretized vector field

[3,4]

[-2,7]

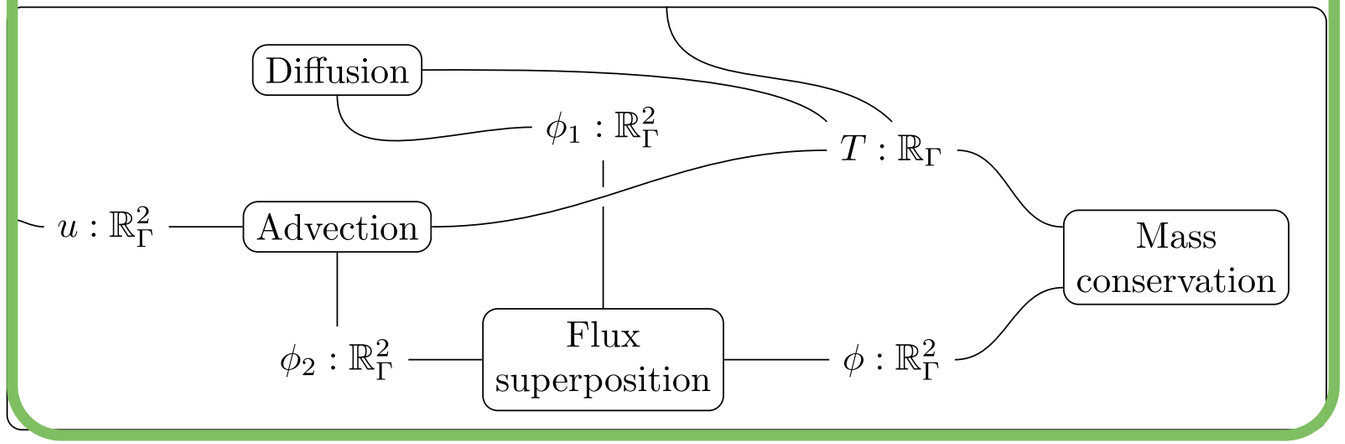
Multigrid Performance

Using GMG provides smoother error than GMRES

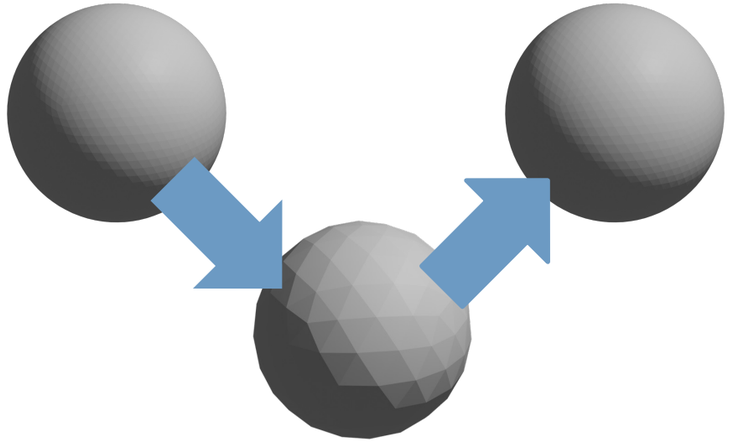


M3P: Multidomain, Multiscale, Multiphysics

Multiphysics: composing physical laws



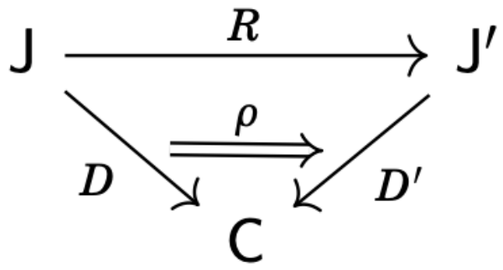
Multiscale: need multiple grid with different resolutions



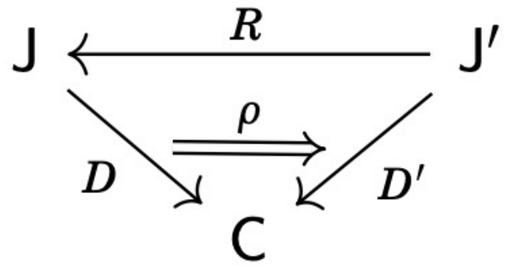
Multidomain: complex scenarios need different physics on different domains

Morphisms of Diagrams

Forward Direction



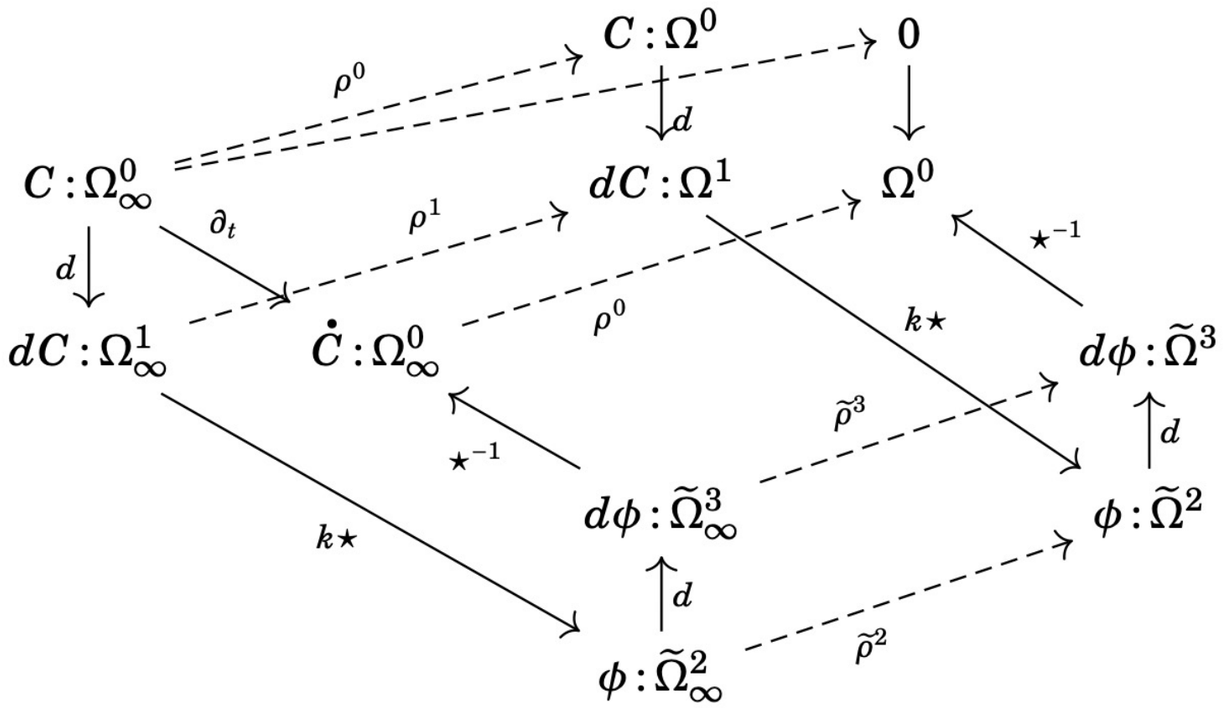
Reverse Direction



- Maps between diagrams encode relationships between physical theories
- Functor between the shapes and then a natural transformation between the data.
- R specifies relationships between syntactic variables and operators
- rho specifies the relationships between the numerical data
- When do two equations specify *the same physics*?

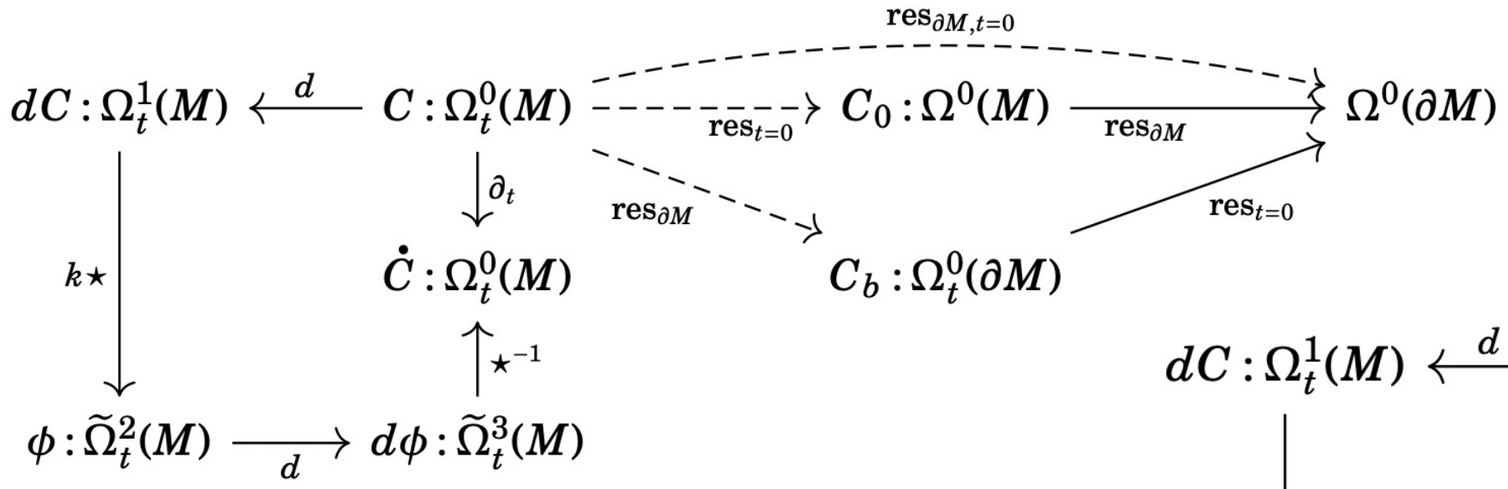
Steady States of a System Expressed Diagrammatically

A morphism (R, ρ) from the first diagram to the second can be depicted as



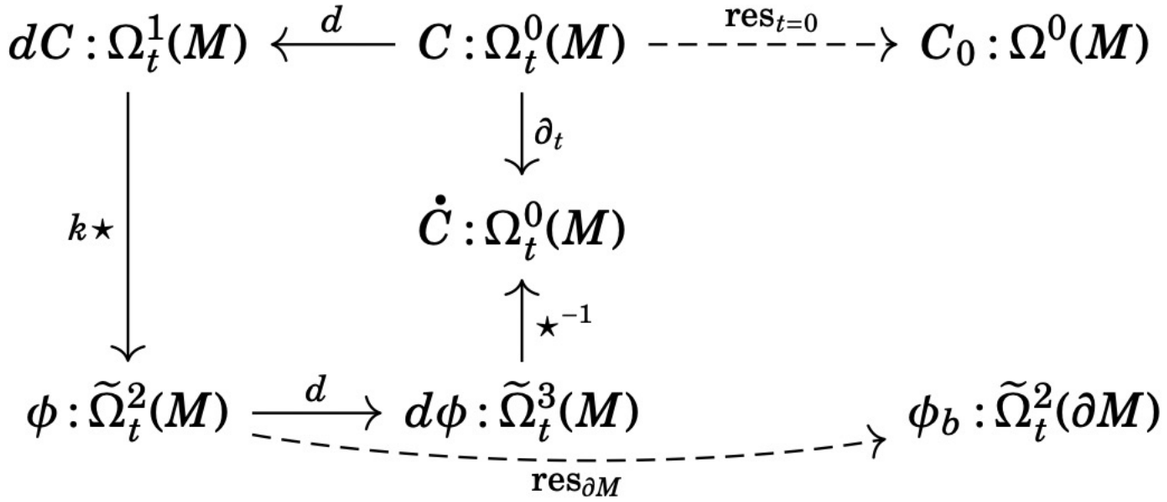
- Maps between diagrams encode relationships between physical theories
- ie. the steady state heat equation is the limit of the dynamic heat eqn as time goes to infinity

Boundary Conditions



Dirichlet

Neumann



- Morphisms between manifolds (like the boundary operator) induce maps between forms.
- Can enforce boundary and initial conditions in the same framework

GPU Implementation

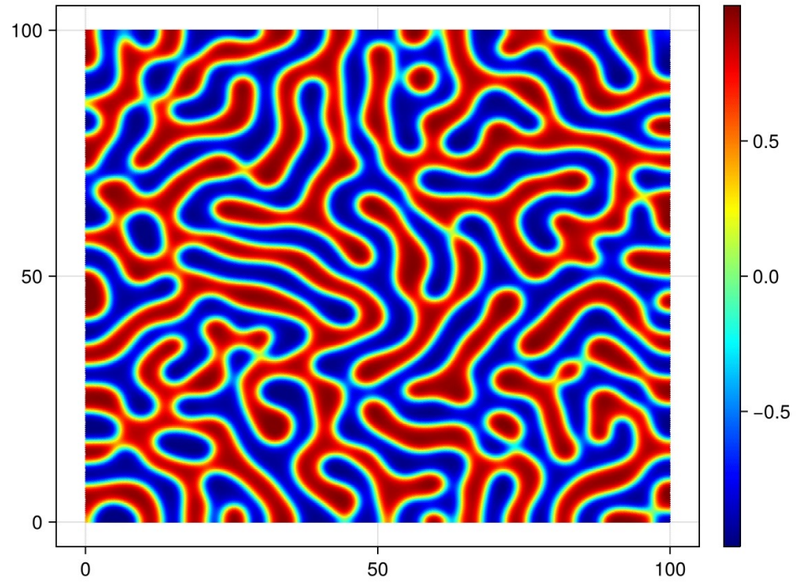
- Most DEC operators are sparse matrices. (d, δ, Δ, etc.)
- Hodge star (★) is diagonal
- Wedge products (∧) are rank 2 tensors, implemented as custom kernels done with KernelAbstractions.jl.
- All other operators, like the Lie Derivative are combinations of these basic operators

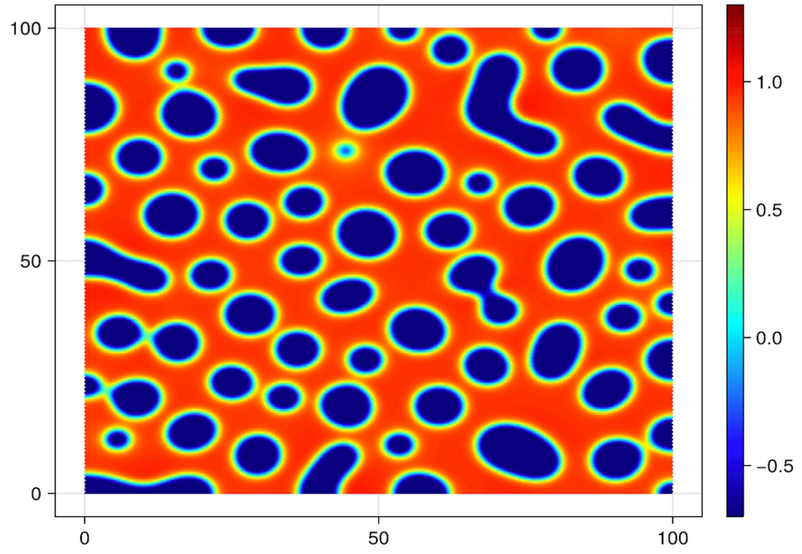
```
@kernel function wedge_kernel_11!(res,
@Const(α), @Const(β), @Const(e), @Const(c))

 i = @index(Global)
 e0, e1, e2 = e[Int32(1), i], e[Int32(2), i],
e[Int32(3), i]

 c1, c2, c3 = c[Int32(1), i], c[Int32(2), i],
c[Int32(3), i]

 ae0, ae1, ae2 = α[e0], α[e1], α[e2]
 be0, be1, be2 = β[e0], β[e1], β[e2]

 @inbounds res[i] = (c1 * (ae2 * be1 -  ae1 *
be2) + c2 * (ae2 * be0 - ae0 *    be2) + c3 *
(ae1 * be0 - ae0 * be1))
end
```

GPU Implementation

```
CahnHilliard = @decapode begin
  C::Form0
  (D, γ)::Constant
  ∂ₜ(C) ==
    D ∗ Δ(C.^3 - C - γ ∗ Δ(C))
end
```

Mesh Size (Vertices)	CPU Time (s)	GPU Time (s)	Speedup (CPU/GPU)
58,081 (1)	2.772	0.497	~5.58X
90,601 (0.8)	8.463	1.074	~7.88X
160,801 (0.6)	48.313	2.943	~16.42X
231,361 (0.5)	139.489	6.122	~22.78X
361,201 (0.4)	532.123	15.266	~34.86X

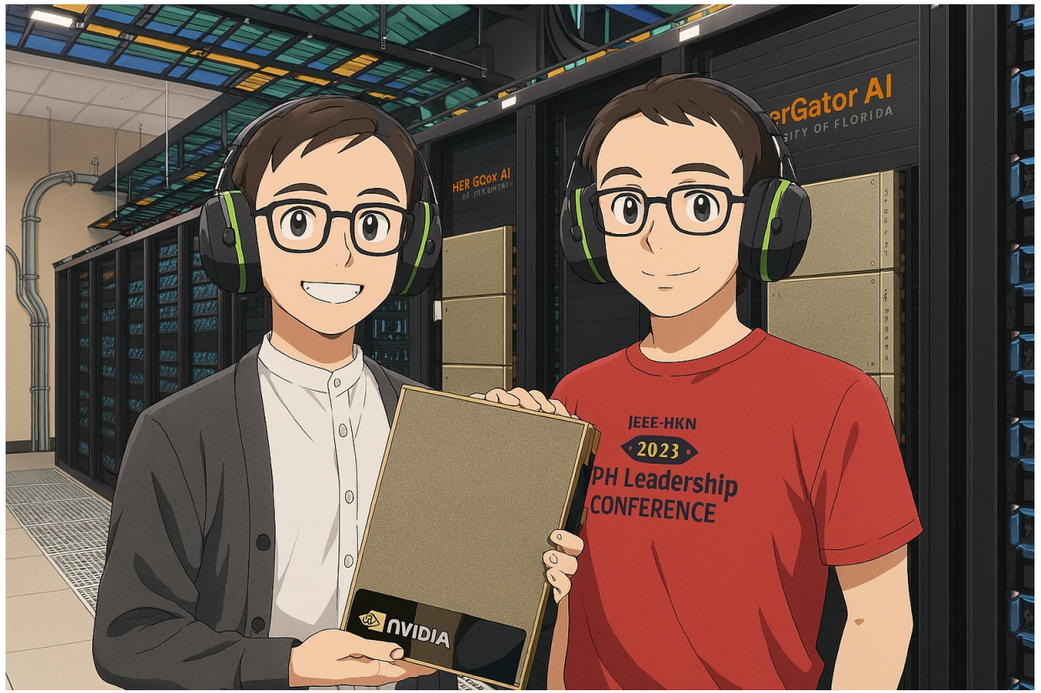




32 CPU Threads AMD Processor vs A100 GPU

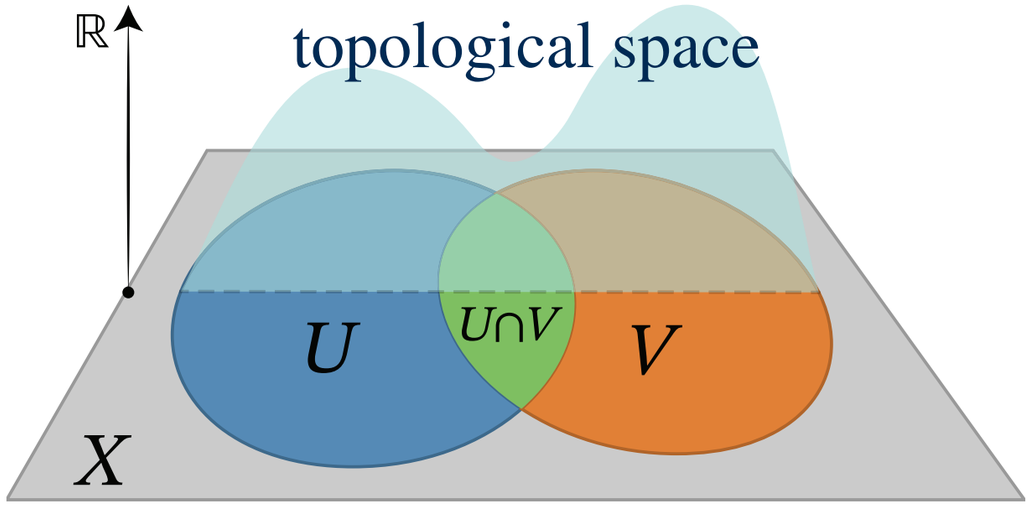
Conclusions

- Decapodes is an advanced multiphysics simulation platform for DEC
- Supports features that you expect, Multigrid, GPU acceleration
- Validated against other DEC results showing similar accuracy.
- Ready to support other groups in DEC research
- Future Directions: Distributed Memory Parallelism, FEEC, Specialized numerics for CFD/EM

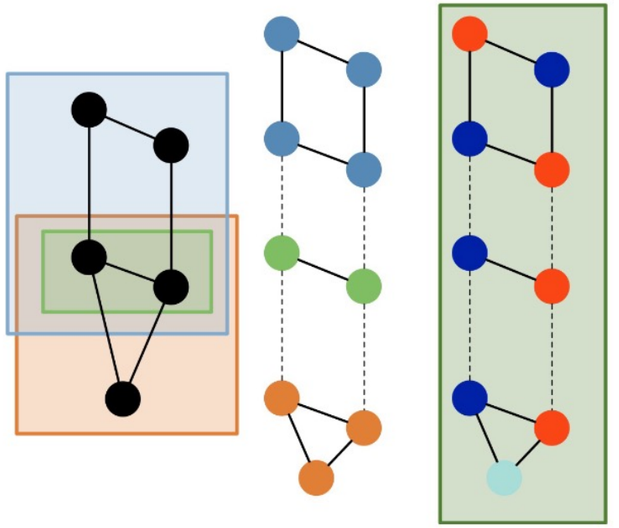


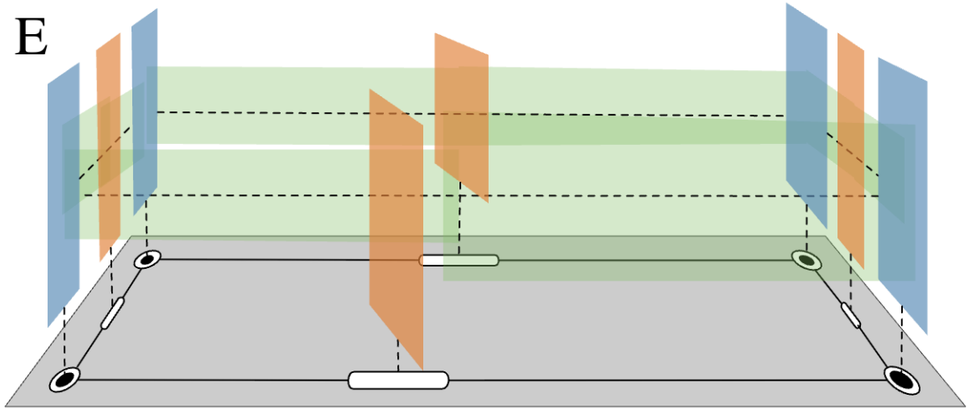
Sheaf theory studies local to global phenomena

The sheaf of continuous functions on a topological space



The sheaf of 3-colorings on a graph





A cellular sheaf of vector spaces over a graph

The sheaf of solutions to a PDE