FROSch Preconditioners for Land Ice Simulations of Greenland and Antarctica

Alexander Heinlein\textsuperscript{1}  Mauro Perego\textsuperscript{2}  Sivasankaran Rajamanickam\textsuperscript{2}  Ichitaro Yamazaki\textsuperscript{2}

Trilinos User-Developer Group Meeting 2021, 30 November–2 December, 2021

\textsuperscript{1}Delft University of Technology

\textsuperscript{2}Sandia National Laboratories
Land Ice Simulations of Greenland and Antarctica

Greenland and Antarctica ice sheets
- store most of the fresh water on earth and
- mass loss from these ice sheets
  significantly contributes to sea-level rise.

The simulation of **temperature and velocity**
of the ice sheets gives rise to **large highly
nonlinear systems of equations** with a **strong
coupling** of the variables.

The simulations are also characterized by:
- **The mesh structure:**
  - Volume mesh is obtained by **extrusion** of the surface mesh
  ⇒ **2D domain decomposition**.
- **Highly anisotropic.**
- Specific combination of Dirichlet, Neumann, and Robin **boundary
conditions**.
Consider a **Poisson model problem** on $[0, 1]^2$:

$$-\Delta u = f \quad \text{in } \Omega, \quad u = 0 \quad \text{on } \partial \Omega.$$ 

Discretize (e.g., using finite elements)

$$Kx = b.$$ 

$\Rightarrow$ Construct a **parallel scalable**

**preconditioner** $M^{-1}$ using **overlapping**

**Schwarz domain decomposition methods.**

**Overlapping domain decomposition**

**Overlapping Schwarz methods** are based on **overlapping decompositions** of the computational domain $\Omega$.

Overlapping subdomains $\Omega'_1, \ldots, \Omega'_N$ can be constructed by **recursively adding layers of elements** to nonoverlapping subdomains $\Omega_1, \ldots, \Omega_N$.
Consider a **Poisson model problem** on $[0, 1]^2$:

$$-\Delta u = f \quad \text{in } \Omega,$$

$$u = 0 \quad \text{on } \partial \Omega.$$

Discretize (e.g., using finite elements)

$$Kx = b.$$ 

**⇒ Construct a parallel scalable preconditioner** $M^{-1}$ **using overlapping Schwarz domain decomposition methods.**

**Overlapping domain decomposition**

**Overlapping Schwarz methods** are based on **overlapping decompositions** of the computational domain $\Omega$.

Overlapping subdomains $\Omega'_1, \ldots, \Omega'_N$ can be constructed by **recursively adding layers of elements** to nonoverlapping subdomains $\Omega_1, \ldots, \Omega_N$. 

Nonoverlap. DD  
Overlap $\delta = 1h$
Model Problem & Domain Decomposition

Consider a Poisson model problem on $[0,1]^2$:

$$-\Delta u = f \quad \text{in } \Omega,$$

$$u = 0 \quad \text{on } \partial\Omega.$$ 

Discretize (e.g., using finite elements)

$$Kx = b.$$ 

⇒ Construct a parallel scalable preconditioner $M^{-1}$ using overlapping Schwarz domain decomposition methods.

Overlapping domain decomposition

Overlapping Schwarz methods are based on overlapping decompositions of the computational domain $\Omega$.

Overlapping subdomains $\Omega'_1, \ldots, \Omega'_N$ can be constructed by recursively adding layers of elements to nonoverlapping subdomains $\Omega_1, \ldots, \Omega_N$.

Nonoverlap DD  Overlap $\delta = 1h$  Overlap $\delta = 2h$
Two-Level Schwarz Preconditioners

### One-Level Schwarz preconditioner

- **Overlap** $\delta = 1h$
- **Restriction** $R_i$ to $\Omega_i'$

Based on an **overlapping domain decomposition**, we define a **one-level Schwarz operator**

$$M_{OS-1}^{-1} K = \sum_{i=1}^{N} R_i^T K_i^{-1} R_i K,$$

where $R_i$ and $R_i^T$ are restriction and prolongation operators corresponding to $\Omega_i'$, and $K_i := R_i K R_i^T$.

**Condition number estimate:**

$$\kappa(P_{OS-1}) \leq C \left( 1 + \frac{1}{H\delta} \right)$$

with subdomain size $H$ and the width of the overlap $\delta$.

### Adding a Lagrangian coarse space

- **Coarse triangulation**
- **Q1 basis function**

The **two-level overlapping Schwarz operator** reads

$$M_{OS-2}^{-1} K = \Phi K_0^{-1} \Phi^T K + \sum_{i=1}^{N} R_i^T K_i^{-1} R_i K,$$

where $\Phi$ contains the coarse basis functions and $K_0 := \Phi^T \Phi$; cf., e.g., Toselli, Widlund (2005).

A Lagrangian coarse basis requires a coarse triangulation (geometric information) → **not algebraic**

$$\Rightarrow \kappa(P_{OS-2}) \leq C \left( 1 + \frac{H}{\delta} \right)$$
In GDSW (Generalized–Dryja–Smith–Widlund) coarse spaces, the coarse basis functions are chosen as energy minimizing extensions of functions $\Phi_\Gamma$ that are defined on the interface $\Gamma$:

$$
\Phi = \begin{bmatrix} -K_{ll}^{-1}K_{\Gamma l}^T \Phi_\Gamma \\ \Phi_\Gamma \end{bmatrix} = \begin{bmatrix} \Phi_l \\ \Phi_\Gamma \end{bmatrix}
$$

The functions $\Phi_\Gamma$ are restrictions of the null space of global Neumann matrix to the edges, vertices, and, in 3D, faces (partition of unity) of the non-overlapping decomposition.

The condition number of the GDSW operator is bounded by

$$
\kappa \left( M_{GDSW}^{-1} K \right) \leq C \left( 1 + \frac{H}{\delta} \right) \left( 1 + \log \left( \frac{H}{h} \right) \right)^2;
$$


→ We only obtain the exponent 2 for very irregular subdomains.

→ Scalable and algebraic!
Weak Scalability up to $64k$ MPI Ranks / $1.7b$ Unknowns (3D Poisson; Juqueen)

**GDSW vs RGDSW (reduced dimension)**


![Graphs showing comparison between GDSW and RGDSW](image)

**RGDSW (Reduced dimension GDSW)**

Non-overlapping DD  
Ident. vertices & edges

RGDSW option 1  
RGDSW option 2.2

Reduced dimension GDSW coarse spaces are constructed from **nodal interface functions (different partition of unity)**; cf. Dohrmann, Widlund (2017).
Software Framework for the Land Ice Simulations

FROSch (Fast and Robust Overlapping Schwarz) preconditioners through Stratimikos/Thyra interface.

https://github.com/trilinos/Trilinos

https://github.com/SNLComputation/Albany

If not mentioned otherwise, the simulations were performed on the Cori supercomputer (NERSC).
Software Framework for the Land Ice Simulations

ФROSch (Fast and Robust Overlapping Schwarz) preconditioners through Stratimikos/Thyra interface.

If not mentioned otherwise, the simulations were performed on the Cori supercomputer (NERSC).

Hardware

https://github.com/trilinos/Trilinos

https://github.com/SNLComputation/Albany
We use a first-order (or Blatter-Pattyn) approximation of the Stokes equations

\[
\begin{align*}
-\nabla \cdot (2\mu \dot{e}_1) &= -\rho_i |g| \partial_x s, \\
-\nabla \cdot (2\mu \dot{e}_2) &= -\rho_i |g| \partial_y s,
\end{align*}
\]

with the \(\rho_i\) the ice density, the ice surface elevation \(s(x, y)\), the gravity acceleration \(g\), and strain rates \(\dot{e}_1\) and \(\dot{e}_2\); cf. Blatter (1995) and Pattyn (2003).

Nonlinear viscosity model

The ice viscosity \(\mu\) is modeled using Glen’s law

\[
\mu = \frac{1}{2}A(T)\left(\frac{1}{n}\right)^{1-n} \dot{e}^n,
\]

where \(A(T) = \alpha_1 e^{\alpha_2 T}\) is a temperature-dependent rate factor, \(n = 3\) is the power-law exponent, and the effective strain rate \(\dot{e}\).

Velocity Problem

We use a first-order (or Blatter-Pattyn) approximation of the Stokes equations

\[
\begin{align*}
-\nabla \cdot (2\mu \dot{\varepsilon}_1) &= -\rho_i |g| \partial_x s, \\
-\nabla \cdot (2\mu \dot{\varepsilon}_2) &= -\rho_i |g| \partial_y s,
\end{align*}
\]

with the \(\rho_i\) the ice density, the ice surface elevation \(s(x, y)\), the gravity acceleration \(g\), and strain rates \(\dot{\varepsilon}_1\) and \(\dot{\varepsilon}_2\); cf. Blatter (1995) and Pattyn (2003).

Boundary conditions

- **Upper surface**: \(\dot{\varepsilon}_j = 0, j = 1, 2\) (stress-free Neumann condition)
- **Lower surface**: \(2\mu \varepsilon \dot{\varepsilon}_j \cdot n + \beta u = 0, j = 1, 2\) (sliding Robin condition with friction coefficient \(\beta\))
- **Lateral boundary**: \(2\mu \varepsilon \dot{\varepsilon}_j \cdot n = \frac{1}{2} gH (\rho_i - \rho_w r^2) n_1, j = 1, 2\) (open-ocean Neumann condition with density of ocean water \(\rho_w\) and ratio of submerged ice thickness \(r\))

## Antarctica Velocity Problem – Comparison of Coarse Spaces (Strong Scaling)

### Without rotational coarse basis functions (2 rigid body modes)

<table>
<thead>
<tr>
<th>MPI ranks</th>
<th>GDSW</th>
<th>RGDSW</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>dim $V_0$</td>
<td>avg. its</td>
</tr>
<tr>
<td>512</td>
<td>4 598</td>
<td>40.8 (11)</td>
</tr>
<tr>
<td>1 024</td>
<td>9 306</td>
<td>43.3 (11)</td>
</tr>
<tr>
<td>2 048</td>
<td>18 634</td>
<td>41.7 (11)</td>
</tr>
<tr>
<td>4 096</td>
<td>37 184</td>
<td>41.4 (11)</td>
</tr>
<tr>
<td>8 192</td>
<td>72 964</td>
<td>39.5 (11)</td>
</tr>
</tbody>
</table>

### With rotational coarse basis functions (3 rigid body modes)

<table>
<thead>
<tr>
<th>MPI ranks</th>
<th>GDSW</th>
<th>RGDSW</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>dim $V_0$</td>
<td>avg. its</td>
</tr>
<tr>
<td>512</td>
<td>6 897</td>
<td>35.5 (11)</td>
</tr>
<tr>
<td>1 024</td>
<td>13 959</td>
<td>35.6 (11)</td>
</tr>
<tr>
<td>2 048</td>
<td>27 951</td>
<td>33.5 (11)</td>
</tr>
<tr>
<td>4 096</td>
<td>55 776</td>
<td>31.8 (11)</td>
</tr>
<tr>
<td>8 192</td>
<td>109 446</td>
<td>29.3 (11)</td>
</tr>
</tbody>
</table>

### Problem: Velocity

### Mesh: Antarctica

### Size: 35.3 m degrees

4 km hor. resolution

20 vert. layers
Antarctica Velocity Problem – Weak Scalability

- Weak scalability study for an **increasing** horizontal mesh resolution.
  - **1 OpenMP thread**: From 32 to 8,192 processor cores
  - **4 OpenMP threads**: From 128 to 32,768 processor cores
- The number of vertical layers is fixed to 20.
- P1 FEM spatial discretization.

### Problem: Velocity
### Meshes: Antarctica
### Discretization: P1 FE
### Coarse space: RGDSW

<table>
<thead>
<tr>
<th>MPI ranks</th>
<th>mesh</th>
<th># dofs</th>
<th>1 OpenMP thread</th>
<th>4 OpenMP threads</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>avg. its (nl its)</td>
<td>avg. setup</td>
</tr>
<tr>
<td>32</td>
<td>16 km</td>
<td>2.2 m</td>
<td>24.1 (11)</td>
<td>11.97 s</td>
</tr>
<tr>
<td>128</td>
<td>8 km</td>
<td>8.8 m</td>
<td>32.0 (10)</td>
<td>14.08 s</td>
</tr>
<tr>
<td>512</td>
<td>4 km</td>
<td>35.3 m</td>
<td>42.6 (11)</td>
<td>14.99 s</td>
</tr>
<tr>
<td>2048</td>
<td>2 km</td>
<td>141.5 m</td>
<td>61.0 (11)</td>
<td>22.83 s</td>
</tr>
<tr>
<td>8,192</td>
<td>1 km</td>
<td>566.1 m</td>
<td>67.1 (14)</td>
<td>17.36 s</td>
</tr>
</tbody>
</table>

Antarctica mesh & domain decomposition.
Temperature Problem

The **steady state enthalpy equation** reads

\[ \nabla \cdot q(h) + u \cdot \nabla h = 4\mu \epsilon^2 \]

with the enthalpy growing linearly with the water content \( \phi \)

\[ h = \begin{cases} 
\rho_i c_i (T - T_0), & \text{for cold ice } (h \leq h_m), \\
h_m + \rho_w L \phi, & \text{for temperate ice}.
\end{cases} \]

the **melting enthalpy** \( h_m := \rho_w c(T_m - T_0) \), the **uniform reference temperature** \( T_0 \), and the **enthalpy flux**

\[ q(h) = \begin{cases} 
\frac{k}{\rho_i c_i} \nabla h, & \text{for cold ice } (h \leq h_m), \\
\frac{k}{\rho_i c_i} \nabla h_m + \rho_w L j(h), & \text{for temperate ice}.
\end{cases} \]

The **water flux term**

\[ j(h) := \frac{1}{\eta_w} (\rho_w - \rho_i) k_0 \phi^\gamma g \]


See Perego et al. (in preparation) and Heinlein et. al (submitted 2021) for more details.
The **steady state enthalpy equation** reads

\[ \nabla \cdot q(h) + u \cdot \nabla h = 4\mu \epsilon_e^2 \]

with the enthalpy growing linearly with the water content \( \phi \)

\[ h = \begin{cases} 
\rho_i c_i (T - T_0), & \text{for cold ice } (h \leq h_m), \\
h_m + \rho_w L \phi, & \text{for temperate ice.} 
\end{cases} \]

the **melting enthalpy** \( h_m := \rho_w c_i (T_m - T_0) \), the **uniform reference temperature** \( T_0 \), and the **enthalpy flux**

\[ q(h) = \begin{cases} 
\frac{k}{\rho_i c_i} \nabla h, & \text{for cold ice } (h \leq h_m), \\
\frac{k}{\rho_i c_i} \nabla h_m + \rho_w L j(h), & \text{for temperate ice.} 
\end{cases} \]

**Boundary conditions**

- **Upper surface**: \( h = \rho_i c_i (T_s - T_0) \)
  (Dirichlet boundary condition)

- **Bed**: \( m = G + \beta \sqrt{u^2 + v^2} - k \nabla T \cdot n \),
  \( m (T - T_m) = 0, \ T_m \leq 0 \).
  (Stefan boundary condition with melting rate \( m \))

See Perego et al. (in preparation) and Heinlein et. al (submitted 2021) for more details.
# Greenland Temperature Problem – One-Level Schwarz VS Two-Level Schwarz

**one-level Schwarz preconditioner**

<table>
<thead>
<tr>
<th>MPI ranks</th>
<th>one layer of algebraic overlap</th>
<th>two layers of algebraic overlap</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>avg.</td>
<td>avg.</td>
</tr>
<tr>
<td></td>
<td>its</td>
<td>setup</td>
</tr>
<tr>
<td>512</td>
<td>18.1 (11)</td>
<td>0.42 s</td>
</tr>
<tr>
<td>1024</td>
<td>23.7 (11)</td>
<td>0.25 s</td>
</tr>
<tr>
<td>2048</td>
<td>29.6 (11)</td>
<td>0.16 s</td>
</tr>
<tr>
<td>4096</td>
<td>39.8 (11)</td>
<td>0.15 s</td>
</tr>
</tbody>
</table>

**RGDSW preconditioner**

<table>
<thead>
<tr>
<th>MPI ranks</th>
<th>one layer of algebraic overlap</th>
<th>two layers of algebraic overlap</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>avg.</td>
<td>avg.</td>
</tr>
<tr>
<td></td>
<td>avg.</td>
<td>its</td>
</tr>
<tr>
<td>512</td>
<td>19.5 (11)</td>
<td>0.44 s</td>
</tr>
<tr>
<td>1024</td>
<td>25.2 (11)</td>
<td>0.28 s</td>
</tr>
<tr>
<td>2048</td>
<td>31.5 (11)</td>
<td>0.26 s</td>
</tr>
<tr>
<td>4096</td>
<td>42.2 (11)</td>
<td>0.25 s</td>
</tr>
</tbody>
</table>

### Problem
- Temperature

### Mesh
- Greenland
- 1-10 km hor. resolution
- 20 vert. layers

### Size
- 1.9 m degrees of freedom (P1 FE)
Couple the velocity and temperature problems. Therefore, compute the vertical velocity $w$ using the incompressibility condition

$$\partial_x u + \partial_y v + \partial_z w = 0,$$

with the Dirichlet boundary condition at the ice lower surface

$$u \cdot n = \frac{m}{L (\rho_i - \rho_w \phi)}.$$

Then, the tangent matrix of the coupled problem has the structure

$$\begin{bmatrix} A_u & C_{uT} \\ C_{Tu} & A_T \end{bmatrix} \begin{bmatrix} x_u \\ x_T \end{bmatrix} = \begin{bmatrix} \tilde{r}_u \\ \tilde{r}_T \end{bmatrix}.$$
Monolithic (R)GDSW Preconditioners for CFD Simulations

Consider the discrete saddle point problem

\[ \mathcal{A} x = \begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix} = b. \]

We construct a monolithic GDSW preconditioner

\[ \mathcal{M}^{-1}_{\text{GDSW}} = \phi A_0^{-1} \phi^T + \sum_{i=1}^N \mathcal{R}_i^T A_i^{-1} \mathcal{R}_i, \]

with block matrices \( A_0 = \phi^T A \phi \), \( A_i = \mathcal{R}_i A \mathcal{R}_i^T \), and

\[ \mathcal{R}_i = \begin{bmatrix} \mathcal{R}_{u,i} & 0 \\ 0 & \mathcal{R}_{p,i} \end{bmatrix} \quad \text{and} \quad \phi = \begin{bmatrix} \Phi_{u,u0} & \Phi_{u,p0} \\ \Phi_{p,u0} & \Phi_{p,p0} \end{bmatrix}. \]

Using \( \mathcal{A} \) to compute extensions: \( \phi_I = -A_I^{-1} A_{II} \phi_I \); cf. Heinlein, Hochmuth, Klawonn (2019, 2020).

Related work:

- Other publications on monolithic Schwarz preconditioners: e.g., Hwang and Cai (2006), Barker and Cai (2010), Wu and Cai (2014), and the presentation Dohrmann (2010) at the Workshop on Adaptive Finite Elements and Domain Decomposition Methods in Milan.
Monolithic GDSW Preconditioner

Consider the discrete saddle point problem

\[ Ax = \begin{bmatrix} A & B^T \\ B & 0 \end{bmatrix} \begin{bmatrix} u \\ p \end{bmatrix} = \begin{bmatrix} f \\ 0 \end{bmatrix} = b. \]

We construct a **monolithic GDSW preconditioner**

\[ M_{\text{GDSW}}^{-1} = \phi A_0^{-1} \phi^T + \sum_{i=1}^{N} R_i^T A_i^{-1} R_i, \]

with block matrices \( A_0 = \phi^T A \phi \), \( A_i = R_i A R_i^T \), and

\[ R_i = \begin{bmatrix} R_{u,i} & 0 \\ 0 & R_{p,i} \end{bmatrix} \quad \text{and} \quad \phi = \begin{bmatrix} \Phi_{u,u_0} & \Phi_{u,p_0} \\ \Phi_{p,u_0} & \Phi_{p,p_0} \end{bmatrix}. \]

Using \( A \) to compute extensions: \( \phi_I = -A_I^{-1} A_{II} \phi_I \); cf. Heinlein, Hochmuth, Klawonn (2019, 2020).

Monolithic vs Block Preconditioners

**Stokes flow**

Computations performed on magnitUDE, University Duisburg-Essen.
We construct a **monolithic two-level (R)GDSW preconditioner** \cite{HeinleinHochmuthKlawonn2019, HeinleinHochmuthKlawonn2020} for the tangent matrix of the coupled problem

\[
M^{-1}_{GDSW} = \phi A_0^{-1} \phi^T + \sum_{i=1}^{N} R_i^T A_i^{-1} R_i,
\]

for the tangent matrix of the coupled problem

\[
Ax := \begin{bmatrix} A_u & C_{uT} \\ C_{Tu} & A_T \end{bmatrix} \begin{bmatrix} x_u \\ x_T \end{bmatrix} = \begin{bmatrix} \tilde{r}_u \\ \tilde{r}_T \end{bmatrix} =: r.
\]

### Null space

We use an **equal-order P1 finite element discretization in space** for all variables. Therefore, the null space in each finite element node is spanned by:

- \( r_{u,1} := \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \),
- \( r_{u,2} := \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \),
- \( r_{u,3} := \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \),
- \( r_{u,4} := \begin{bmatrix} y \\ -x \\ 0 \\ 0 \end{bmatrix} \),
- \( r_T := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \).

See \cite{HeinleinPeregoRajamanickamSubmitted2021} for more details.
Monolithic (R)GDSW Preconditioners for Multiphysics Land Ice Simulations

We construct a monolithic two-level (R)GDSW preconditioner (Heinlein, Hochmuth, Klawonn (2019, 2020))

\[
\mathcal{M}_{\text{GDSW}}^{-1} = \phi A_0^{-1} \phi^T + \sum_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i^{-1} \mathcal{R}_i,
\]

for the tangent matrix of the coupled problem

\[
\mathcal{A} x := \begin{bmatrix} A_u & C_{uT} \\ C_{Tu} & A_T \end{bmatrix} \begin{bmatrix} x_u \\ x_T \end{bmatrix} = \begin{bmatrix} \tilde{r}_u \\ \tilde{r}_T \end{bmatrix} =: r.
\]

Fully coupled extensions
We compute coarse basis function using extensions

\[
\phi = \begin{bmatrix} -A_{ll}^{-1} A_{l_1 l}^T \Phi_l \\ \Phi_l \end{bmatrix} = \begin{bmatrix} \phi_l \\ \phi_l \end{bmatrix}
\]

based on the coupled matrix \( \mathcal{A} \).

Decoupled extensions
We compute coarse basis function using extensions

\[
\tilde{\phi} = \begin{bmatrix} -\tilde{A}_{ll}^{-1} \tilde{A}_{l_1 l}^T \Phi_l \\ \Phi_l \end{bmatrix} = \begin{bmatrix} \phi_l \\ \phi_l \end{bmatrix}
\]

based on the decoupled matrix

\[
\tilde{\mathcal{A}} = \begin{bmatrix} A_u & 0 \\ 0 & A_T \end{bmatrix}.
\]

See Heinlein, Perego, Rajamanickam (submitted 2021) for more details.
### Greenland Coupled Problem – Coarse Spaces

| MPI ranks | dim V₀ | fully coupled extensions | |  |  | decoupled extensions | |  |  |
|---|---|---|---|---|---|---|---|---|
|  |  | no reuse | reuse coarse basis | no reuse | reuse coarse basis | no reuse | reuse coarse basis | no reuse | reuse coarse basis |
| avg. its | avg. its | avg. avg. | avg. its | avg. avg. | avg. avg. | avg. its | avg. avg. | avg. avg. | avg. its | avg. avg. | avg. avg. |
| (nl its) | setup | solve | (nl its) | setup | solve | (nl its) | setup | solve | (nl its) | setup | solve |
| 256 | 1 400 | 100.1 (27) | 4.10 s | 6.40 s | 18.5 (70) | 2.28 s | 1.07 s | 23.6 (29) | 3.90 s | 1.32 s | 21.5 (34) | 2.23 s | 1.18 s |
| 512 | 2 852 | 129.1 (28) | 1.88 s | 4.20 s | 24.6 (38) | 1.04 s | 0.70 s | 27.5 (30) | 1.83 s | 0.78 s | 26.4 (33) | 1.13 s | 0.78 s |
| 1 024 | 6 036 | 191.2 (65) | 1.21 s | 4.76 s | 34.2 (32) | 0.66 s | 0.70 s | 30.1 (29) | 1.19 s | 0.60 s | 28.6 (43) | 0.66 s | 0.61 s |
| 2 048 | 12 368 | 237.4 (30) | 0.96 s | 4.06 s | 37.3 (30) | 0.60 s | 0.58 s | 36.4 (30) | 0.69 s | 0.56 s | 31.2 (50) | 0.57 s | 0.55 s |

**Problem:** Coupled  
**Mesh:** Greenland  
**Size:** 7.5 m degrees  
**Coarse space:** RGDSW

TUG 2021  
15/20
Greenland Coupled Problem – Large Problem

<table>
<thead>
<tr>
<th>MPI ranks</th>
<th>decoupled (no reuse)</th>
<th>fully coupled (reuse coarse basis)</th>
<th>decoupled (reuse 1st level symb. fact. + coarse basis)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>avg. its. (nl its)</td>
<td>avg. setup avg. solve</td>
<td>avg. its (nl its) avg. setup avg. solve</td>
</tr>
<tr>
<td>512</td>
<td>41.3 (36) 18.78 s</td>
<td>45.3 (32) 11.84 s 5.35 s</td>
<td>45.0 (35) 10.53 s 5.36 s</td>
</tr>
<tr>
<td>1024</td>
<td>53.0 (29)  8.68 s</td>
<td>47.8 (37)  5.36 s 3.82 s</td>
<td>54.3 (32)  4.59 s 4.31 s</td>
</tr>
<tr>
<td>2048</td>
<td>62.2 (86)  4.47 s</td>
<td>66.7 (38)  2.81 s 4.53 s</td>
<td>59.1 (38)  2.32 s 3.99 s</td>
</tr>
<tr>
<td>4096</td>
<td>68.9 (40)  2.52 s</td>
<td>79.1 (36)  1.61 s 3.30 s</td>
<td>78.7 (38)  1.37 s 3.30 s</td>
</tr>
</tbody>
</table>

Problem: Coupled  Mesh: Greenland  Size: 68.6 m degrees of freedom (P1 FE)

Sparse Triangular Solver in Kokkos-Kernels (Amesos2 – SuperLU/Cholmod)

The sparse triangular solver is an important kernel in many codes (including FROSch) but is challenging to parallelize

- Factorization using a sparse direct solver typically leads to triangular matrices with dense blocks called supernodes
- In supernodal triangular solver, rows/columns with a similar sparsity pattern are merged into a supernodal block, and the solve is then performed block-wise
- The parallelization potential for the triangular solver is determined by the sparsity pattern

Parallel supernode-based triangular solver:

1. Supernode-based level-set scheduling, where all leaf-supernodes within one level are solved in parallel (batched kernels for hierarchical parallelism)
2. Partitioned inverse of the submatrix associated with each level: SpTRSV is transformed into a sequence of SpMVs

See Yamazaki, Rajamanickam, Ellingwood (2020) for more details.
Performance Results With Matrices From the SuiteSparse Matrix Collection

Comparison of the **sparse-triangular solve in GEMV and SpMV mode** for the matrix vector multiplications against **CuSparse sparse triangular solver**.

**NVIDIA V100 GPU**

![Comparison graph showing performance results]

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>type</th>
<th>n</th>
<th>nh x</th>
<th>n_f</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ACTIVSg70K</td>
<td>power system grid</td>
<td>69.999</td>
<td>12.6</td>
<td>83</td>
<td>0.003</td>
</tr>
<tr>
<td>2</td>
<td>dawson5</td>
<td>structural problem</td>
<td>51.537</td>
<td>770.4</td>
<td>1277</td>
<td>3.512</td>
</tr>
<tr>
<td>3</td>
<td>qa8fK</td>
<td>acoustic problem</td>
<td>66.127</td>
<td>653.3</td>
<td>22</td>
<td>0.006</td>
</tr>
<tr>
<td>4</td>
<td>FEM3Dtherm</td>
<td>thermal problem</td>
<td>17.880</td>
<td>324.6</td>
<td>15</td>
<td>0.008</td>
</tr>
<tr>
<td>5</td>
<td>thermal1</td>
<td>thermal problem</td>
<td>82.654</td>
<td>58.7</td>
<td>27</td>
<td>0.002</td>
</tr>
<tr>
<td>6</td>
<td>apache1</td>
<td>3D finite difference</td>
<td>80.800</td>
<td>240.2</td>
<td>25</td>
<td>0.002</td>
</tr>
<tr>
<td>7</td>
<td>apache2</td>
<td>3D finite difference</td>
<td>715.176</td>
<td>53.6</td>
<td>32</td>
<td>0.001</td>
</tr>
<tr>
<td>8</td>
<td>helm2c03</td>
<td>2D problem</td>
<td>392.257</td>
<td>14.9</td>
<td>109</td>
<td>0.018</td>
</tr>
</tbody>
</table>
Comparison of the **sparse-triangular solve** in **GEMV and SpMV mode** for the matrix vector multiplications against **CuSparse sparse triangular solver**.

NVIDIA V100 GPU

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>type</th>
<th>$n$</th>
<th>$n_{h,k}$</th>
<th>$n_f$</th>
<th>error</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ACTIVSg70K</td>
<td>power system grid</td>
<td>69,999</td>
<td>12.6</td>
<td>83</td>
<td>0.003</td>
</tr>
<tr>
<td>2</td>
<td>dawson5</td>
<td>structural problem</td>
<td>51,537</td>
<td>770.4</td>
<td>1277</td>
<td>3.512</td>
</tr>
<tr>
<td>3</td>
<td>qa8fk</td>
<td>acoustic problem</td>
<td>66,127</td>
<td>653.3</td>
<td>22</td>
<td>0.006</td>
</tr>
<tr>
<td>4</td>
<td>FEM3Dtherm</td>
<td>thermal problem</td>
<td>17,880</td>
<td>324.6</td>
<td>15</td>
<td>0.008</td>
</tr>
<tr>
<td>5</td>
<td>thermal1</td>
<td>thermal problem</td>
<td>82,654</td>
<td>58.7</td>
<td>27</td>
<td>0.002</td>
</tr>
<tr>
<td>6</td>
<td>apache1</td>
<td>3D finite difference</td>
<td>80,800</td>
<td>240.2</td>
<td>25</td>
<td>0.002</td>
</tr>
<tr>
<td>7</td>
<td>apache2</td>
<td>3D finite difference</td>
<td>715,176</td>
<td>53.6</td>
<td>32</td>
<td>0.001</td>
</tr>
<tr>
<td>8</td>
<td>helm2d03</td>
<td>2D problem</td>
<td>392,257</td>
<td>14.9</td>
<td>109</td>
<td>0.018</td>
</tr>
</tbody>
</table>

→ Performance depends on **number of levels** and **sizes of supernodes**.
### Preliminary Strong Scaling Results

**Computations on Summit (ORNL):** $6 \times$ NVIDIA V100 GPUs and $2 \times 21$-core Power9 per node

<table>
<thead>
<tr>
<th></th>
<th>42 MPI ranks per node</th>
<th>6 MPI ranks, 6 GPUs per node</th>
<th>12 MPI ranks, 6 GPUs per node</th>
<th>42 MPI ranks, 6 GPUs per node</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1 node</strong></td>
<td>its (nl its) subd. setup time solve time</td>
<td>42</td>
<td>171</td>
<td>506.7 s</td>
</tr>
<tr>
<td><strong>2 node</strong></td>
<td>its (nl its) subd. setup time solve time</td>
<td>96</td>
<td>200 (9)</td>
<td>165.9 s</td>
</tr>
<tr>
<td><strong>3 node</strong></td>
<td>its (nl its) subd. setup time solve time</td>
<td>126</td>
<td>208 (9)</td>
<td>88.1 s</td>
</tr>
<tr>
<td><strong>4 node</strong></td>
<td>its (nl its) subd. setup time solve time</td>
<td>168</td>
<td>217 (9)</td>
<td>49.0 s</td>
</tr>
</tbody>
</table>

**Problem:** Coupled

**Mesh:** Greenland (structured)

**Size:** 16 km hor. resolution

**Prec.:** One-level Schwarz with alg. overlap 0

**Setup time:**

- 1 node: 506.7 s
- 2 node: 2892.7 s
- 3 node: 1485.3 s
- 4 node: 848.0 s
Preliminary Strong Scaling Results

**Computations on Summit (ORNL):** 6 × NVIDIA V100 GPUs and 2 × 21-core Power9 per node

<table>
<thead>
<tr>
<th></th>
<th>42 MPI ranks per node</th>
<th>6 MPI ranks, 6 GPUs per node</th>
<th>12 MPI ranks, 6 GPUs per node</th>
<th>42 MPI ranks, 6 GPUs per node</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 node</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>its (nl its) subd. setup time</td>
<td>42</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>solve time</td>
<td>171</td>
<td>OOM</td>
<td>OOM</td>
</tr>
<tr>
<td></td>
<td></td>
<td>506.7 s</td>
<td>29.3 s</td>
<td>156 (9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15.3 s</td>
<td></td>
<td>2892.7 s</td>
</tr>
<tr>
<td>2 node</td>
<td></td>
<td>96</td>
<td>12</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>its (nl its) subd. setup time</td>
<td>200 (9)</td>
<td>165.9 s</td>
<td>156 (9)</td>
</tr>
<tr>
<td></td>
<td>solve time</td>
<td>15.3 s</td>
<td>156.9 s</td>
<td>2892.7 s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>29.3 s</td>
<td></td>
<td>5.9 s</td>
</tr>
<tr>
<td>3 node</td>
<td></td>
<td>126</td>
<td>18</td>
<td>36</td>
</tr>
<tr>
<td></td>
<td>its (nl its) subd. setup time</td>
<td>208 (9)</td>
<td>178 (9)</td>
<td>180 (9)</td>
</tr>
<tr>
<td></td>
<td>solve time</td>
<td>9.7 s</td>
<td>88.1 s</td>
<td>1485.3 s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15.3 s</td>
<td></td>
<td>4.0 s</td>
</tr>
<tr>
<td>4 node</td>
<td></td>
<td>168</td>
<td>24</td>
<td>48</td>
</tr>
<tr>
<td></td>
<td>its (nl its) subd. setup time</td>
<td>217 (9)</td>
<td>165 (9)</td>
<td>197 (9)</td>
</tr>
<tr>
<td></td>
<td>solve time</td>
<td>7.0 s</td>
<td>49.0 s</td>
<td>848.0 s</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15.3 s</td>
<td></td>
<td>3.2 s</td>
</tr>
</tbody>
</table>

**Problem:** Coupled  
**Mesh:** Greenland (structured)  
16 km hor. resolution  
20 vert. layers  

**Size:** 2.0 m degrees of freedom  
(P1 FE)  

**Prec.:** One-level Schwarz with alg. overlap 0

- Generally, **higher setup times** for the GPU configurations
Preliminary Strong Scaling Results

Computations on Summit (ORNL): 6 × NVIDIA V100 GPUs and 2 × 21-core Power9 per node

<table>
<thead>
<tr>
<th>1 node</th>
<th>2 node</th>
<th>3 node</th>
<th>4 node</th>
</tr>
</thead>
<tbody>
<tr>
<td>it's (\text{nl its}) subd. setup time solve time</td>
<td>it's (\text{nl its}) subd. setup time solve time</td>
<td>it's (\text{nl its}) subd. setup time solve time</td>
<td>it's (\text{nl its}) subd. setup time solve time</td>
</tr>
<tr>
<td>42</td>
<td>96</td>
<td>126</td>
<td>168</td>
</tr>
<tr>
<td>171</td>
<td>200 (9)</td>
<td>208 (9)</td>
<td>217 (9)</td>
</tr>
<tr>
<td>506.7 s</td>
<td>165.9 s</td>
<td>88.1 s</td>
<td>49.0 s</td>
</tr>
<tr>
<td>29.3 s</td>
<td>15.3 s</td>
<td>9.7 s</td>
<td>7.0 s</td>
</tr>
<tr>
<td>42 MPI ranks per node</td>
<td>6 MPI ranks, 6 GPUs per node</td>
<td>12 MPI ranks, 6 GPUs per node</td>
<td>42 MPI ranks, 6 GPUs per node</td>
</tr>
<tr>
<td>42</td>
<td>6</td>
<td>12</td>
<td>42</td>
</tr>
<tr>
<td>OOM</td>
<td>156 (9)</td>
<td>178 (9)</td>
<td>165 (9)</td>
</tr>
<tr>
<td>292.7 s</td>
<td>4.0 s</td>
<td>3.7 s</td>
<td>884.8 s</td>
</tr>
<tr>
<td>5.9 s</td>
<td>4.0 s</td>
<td>3.7 s</td>
<td>5.3 s</td>
</tr>
<tr>
<td>setup time</td>
<td>solve time</td>
<td>setup time</td>
<td>solve time</td>
</tr>
<tr>
<td>506.7 s</td>
<td>29.3 s</td>
<td>OOM</td>
<td>12</td>
</tr>
<tr>
<td>88.1 s</td>
<td>165.9 s</td>
<td>178 (9)</td>
<td>165 (9)</td>
</tr>
<tr>
<td>24</td>
<td>24</td>
<td>36</td>
<td>48</td>
</tr>
<tr>
<td>setup time</td>
<td>solve time</td>
<td>setup time</td>
<td>solve time</td>
</tr>
<tr>
<td>88.1 s</td>
<td>178 (9)</td>
<td>180 (9)</td>
<td>197 (9)</td>
</tr>
<tr>
<td>9.7 s</td>
<td>4.0 s</td>
<td>4.0 s</td>
<td>3.2 s</td>
</tr>
<tr>
<td>406 s</td>
<td>208.7 s</td>
<td>168</td>
<td>217 (9)</td>
</tr>
<tr>
<td>10.7 s</td>
<td>59.9 s</td>
<td>10.7 s</td>
<td>6.6 s</td>
</tr>
</tbody>
</table>

Problem: Coupled Mesh: Greenland (structured) 16 km hor. resolution 20 vert. layers
Size: 2.0 m degrees of freedom (P1 FE) Prec.: One-level Schwarz with alg. overlap 0

- Generally, higher setup times for the GPU configurations
- For many configurations, we obtain a significant speedup in the solve times
### Preliminary Strong Scaling Results

**Computations on Summit (ORNL)**: $6 \times$ NVIDIA V100 GPUs and $2 \times 21$-core Power9 per node

<table>
<thead>
<tr>
<th></th>
<th>42 MPI ranks per node</th>
<th>6 MPI ranks, 6 GPUs per node</th>
<th>12 MPI ranks, 6 GPUs per node</th>
<th>42 MPI ranks, 6 GPUs per node</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 node</td>
<td>its (nl its) setup time</td>
<td>42 171 506.7 s 29.3 s</td>
<td>6 OOM</td>
<td>12 OOM</td>
</tr>
<tr>
<td></td>
<td>subd. setup time solve time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 node</td>
<td>its (nl its) setup time</td>
<td>96 200 (9) 165.9 s 15.3 s</td>
<td>12 156 (9) 2892.7 s 5.9 s</td>
<td>24 165 (9) 884.8 s 5.3 s</td>
</tr>
<tr>
<td></td>
<td>subd. setup time solve time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 node</td>
<td>its (nl its) setup time</td>
<td>126 208 (9) 88.1 s 9.7 s</td>
<td>18 178 (9) 1485.3 s 4.0 s</td>
<td>36 180 (9) 506.3 s 3.7 s</td>
</tr>
<tr>
<td></td>
<td>subd. setup time solve time</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 node</td>
<td>its (nl its) setup time</td>
<td>168 217 (9) 49.0 s 7.0 s</td>
<td>24 165 (9) 848.0 s 3.2 s</td>
<td>48 197 (9) 307.7 s 3.3 s</td>
</tr>
<tr>
<td></td>
<td>subd. setup time solve time</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Problem: Coupled Mesh: Greenland (structured) 16 km hor. resolution 20 vert. layers | Size: 2.0 m degrees of freedom (P1 FE) | Prec.: One-level Schwarz with alg. overlap 0 |

- Generally, **higher setup times** for the GPU configurations
- For many configurations, we obtain a significant **speedup in the solve times**
  - Removing UVM dependency and **better parallelization** on the GPUs: **MPS** and **SuperLU_DIST**
Thank you for your attention!

### Summary
- Scalable FROSch preconditioners
  - for the single physics **velocity and temperature problems** and
  - for the **coupled multi physics problem** *(monolithic (R)GDSW preconditioners)*.
- Preliminary results on GPUs using **parallel sparse triangular solver**.

### Outlook
- Improving the robustness of the nonlinear convergence.
- Improving the setup times on GPUs.

### Acknowledgements
- **Financial support:** DFG (KL2094/3-1, RH122/4-1), DOE (SciDAC projects FASTMath, ProSPect)
- **Computing resources:** Cori (NERSC), JUQUEEN (JSC), magnitUDE (UDE), Summit (ORNL)

Disclaimer: This presentation describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy’s National Nuclear Security Administration under contract DE-NA-0003525.