FROSch Preconditioners for Land Ice Simulations of Greenland and Antarctica

Alexander Heinlein[1] Mauro Perego[2] Sivasankaran Rajamanickam[2] Ichitaro Yamazaki[2]

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

[1]Delft University of Technology

[2]Sandia National Laboratories

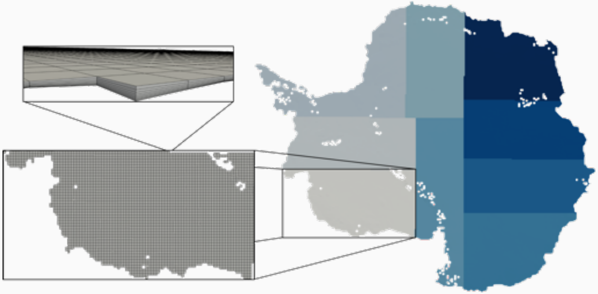
Land Ice Simulations of Greenland and Antarctica

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.

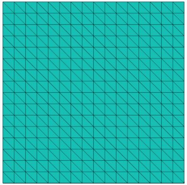
Taken from https://unsplash.com.

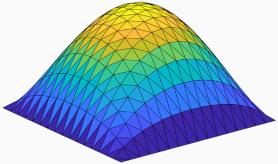


The simulations are also characterized by:

- The **mesh structure**:
  - Volume mesh is obtained by **extrusion** of the surface mesh $\Rightarrow$ **2D domain decomposition**.
  - **Highly anisotropic**.
- Specific combination of Dirichlet, Neumann, and Robin **boundary conditions**.

Model Problem & Domain Decomposition





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

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

Discretize (e.g., using finite elements)

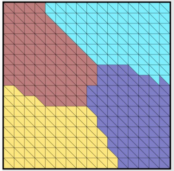
$$Kx = b.$$

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

Overlapping domain decomposition

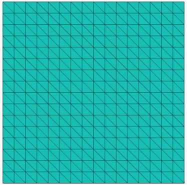
Overlapping Schwarz methods are based on **overlapping decompositions** of the computational domain Ω.

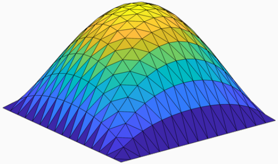
Overlapping subdomains $\Omega'_1, \ldots, \Omega'_N$ can be constructed by **recursively adding layers of elements** to nonoverlapping subdomains $\Omega_1, \ldots, \Omega_N$.



Nonoverlap. DD

Model Problem & Domain Decomposition





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

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

Discretize (e.g., using finite elements)

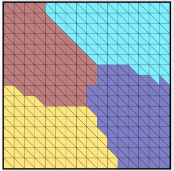
$$Kx = b.$$

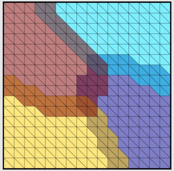
$\Rightarrow$ Construct a **parallel scalable preconditioner** M^{-1} using **overlapping Schwarz domain decomposition methods**.

Overlapping domain decomposition

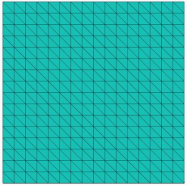
Overlapping Schwarz methods are based on **overlapping decompositions** of the computational domain Ω.

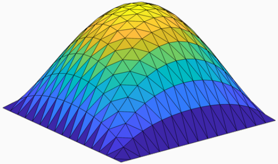
Overlapping subdomains $\Omega_1', \ldots, \Omega_N'$ can be constructed by **recursively adding layers of elements** to nonoverlapping subdomains $\Omega_1, \ldots, \Omega_N$.





Nonoverlap. DD   Overlap $\delta = 1h$





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)

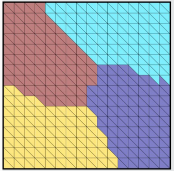
$$Kx = b.$$

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

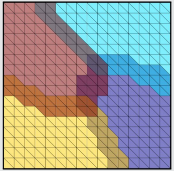
Overlapping domain decomposition

Overlapping Schwarz methods are based on **overlapping decompositions** of the computational domain Ω.

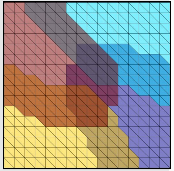
Overlapping subdomains $\Omega'_1, ..., \Omega'_N$ can be constructed by **recursively adding layers of elements** to nonoverlapping subdomains $\Omega_1, ..., \Omega_N$.



Nonoverlap. DD



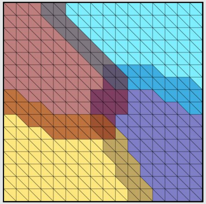
Overlap $\delta = 1h$

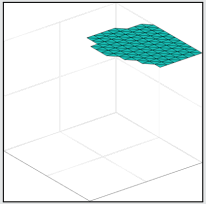


Overlap $\delta = 2h$

Two-Level Schwarz Preconditioners

One-Level Schwarz preconditioner





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

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

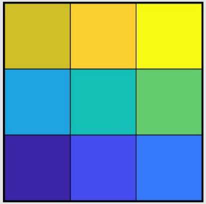
where R_i and R_i^T are restriction and prolongation operators corresponding to Ω_i', and $K_i := R_i K R_i^T$.
$\rightarrow$ algebraic

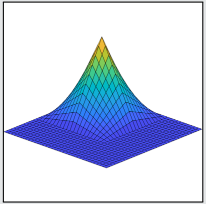
Condition number estimate:

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

with subdomain size H and the width of the overlap δ.

Adding a Lagrangian coarse space





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

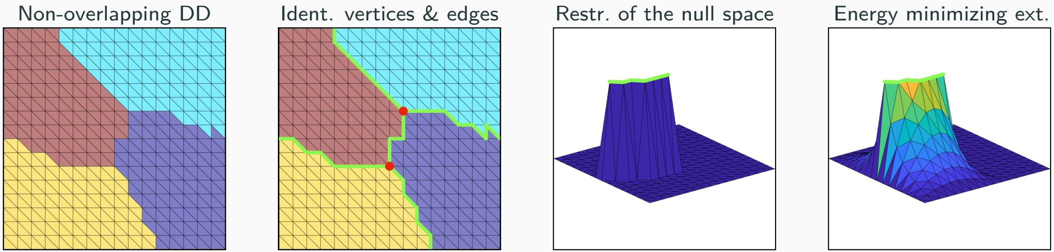
$$M_{\mathrm{OS-2}}^{-1} K = \underbrace{\Phi K_0^{-1} \Phi^T K}_{\text{coarse level – global}} + \underbrace{\sum_{i=1}^{N} R_i^T K_i^{-1} R_i K}_{\text{first level – local}},$$

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

A Lagrangian coarse basis requires a coarse triangulation (geometric information) $\rightarrow$ not algebraic

$$\Rightarrow \kappa\left(P_{\mathrm{OS-2}}\right) \leq C \left(1 + \frac{H}{\delta}\right)$$

Extension-Based GDSW Coarse Spaces



In **GDSW (Generalized–Dryja–Smith–Widlund) coarse spaces**, the coarse basis functions are chosen as **energy minimizing extensions** of functions Φ_Γ that are defined on the interface Γ:

$$\Phi = \begin{bmatrix} -K_{II}^{-1} K_{\Gamma I}^T \Phi_\Gamma \\ \Phi_\Gamma \end{bmatrix} = \begin{bmatrix} \Phi_I \\ \Phi_\Gamma \end{bmatrix}$$

The functions Φ_Γ 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

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

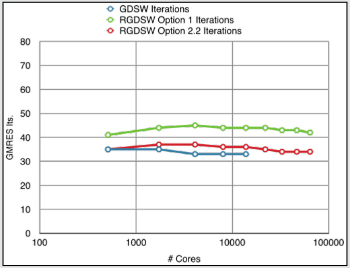
cf. **Dohrmann, Klawonn, Widlund (2008)**, **Dohrmann, Widlund (2009, 2010, 2012)**.

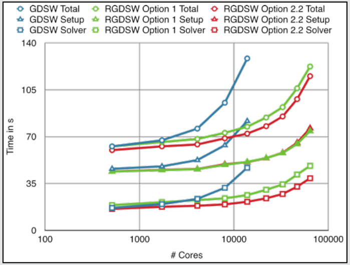
→ We only obtain the exponent 2 for very irregular subdomains.

→ **Scalable and algebraic!**

GDSW vs RGDSW (reduced dimension)

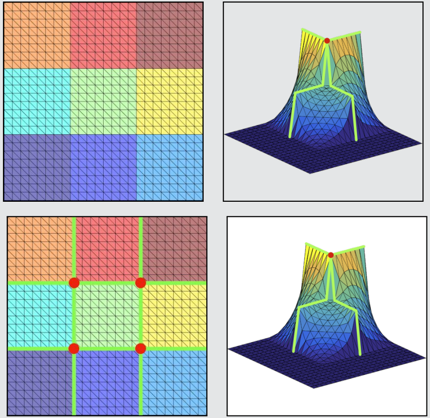
Heinlein, Klawonn, Rheinbach, Widlund (2019).





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).

https://github.com/trilinos/Trilinos

https://github.com/SNLComputation/Albany

`FROSch` (**F**ast and **R**obust **O**verlapping **Sch**warz) preconditioners through `Stratimikos`/`Thyra` interface.

https://github.com/trilinos/Trilinos

https://github.com/SNLComputation/Albany

`FROSch` (**F**ast and **R**obust **O**verlapping **Sch**warz) preconditioners through `Stratimikos/Thyra` interface.

Hardware

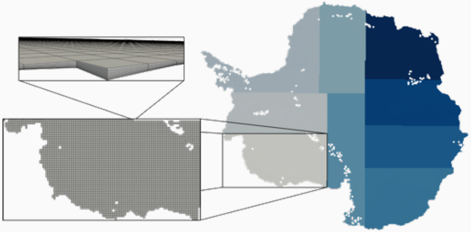
If not mentioned otherwise, the simulations were performed on the Cori supercomputer (NERSC).

Velocity Problem

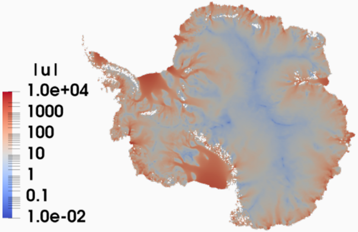
We use a **first-order (or Blatter-Pattyn) approximation of the Stokes equations**

$$\begin{cases} -\nabla \cdot (2\mu\, \dot{\epsilon}_1) &= -\rho_i\, |\boldsymbol{g}|\, \partial_x s, \\ -\nabla \cdot (2\mu\, \dot{\epsilon}_2) &= -\rho_i\, |\boldsymbol{g}|\, \partial_y s, \end{cases}$$

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



Antarctica mesh & domain decomposition.



Velocity u solution

Nonlinear viscosity model

The **ice viscosity** μ is modeled using **Glen's law**

$$\mu = \frac{1}{2} A(T)^{-\frac{1}{n}}\, \dot{\epsilon}_e^{\frac{1-n}{n}},$$

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 $\dot{\epsilon}$.

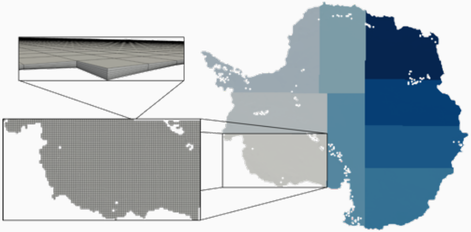
See Perego, Gunzburger, Burkardt (2012) and Tezaur, Perego, Salinger, Tuminaro, Price (2015) for more details.

Velocity Problem

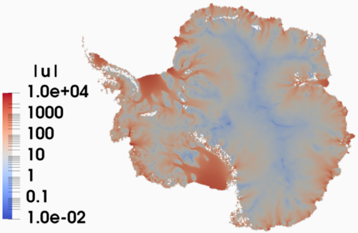
We use a **first-order (or Blatter-Pattyn) approximation of the Stokes equations**

$$\begin{cases} -\nabla \cdot (2\mu\, \dot{\epsilon}_1) &= -\rho_i\, |\boldsymbol{g}|\, \partial_x s, \\ -\nabla \cdot (2\mu\, \dot{\epsilon}_2) &= -\rho_i\, |\boldsymbol{g}|\, \partial_y s, \end{cases}$$

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



Antarctica mesh & domain decomposition.



Velocity u solution

Boundary conditions

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

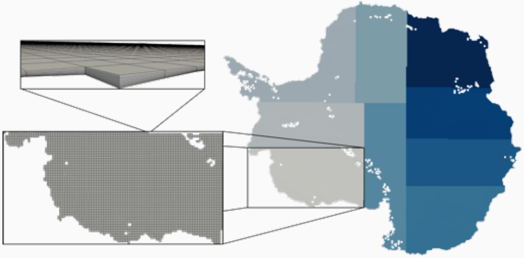
See **Perego, Gunzburger, Burkardt (2012)** and **Tezaur, Perego, Salinger, Tuminaro, Price (2015)** for more details.

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

Without rotational coarse basis functions (2 rigid body modes)								
	GDSW				RGDSW			
MPI ranks	dim V_0	avg. its (nl its)	avg. setup	avg. solve	dim V_0	avg. its (nl its)	avg. setup	avg. solve
512	4 598	**40.8** (11)	15.36 s	**12.38 s**	1 834	42.6 (11)	**14.99 s**	12.50 s
1 024	9 306	**43.3** (11)	5.80 s	6.27 s	3 740	44.5 (11)	**5.65 s**	**6.08 s**
2 048	18 634	**41.7** (11)	3.27 s	2.91 s	7 586	42.7 (11)	**3.11 s**	**2.79 s**
4 096	37 184	**41.4** (11)	2.59 s	2.07 s	15 324	42.5 (11)	**1.07 s**	**1.54 s**
8 192	72 964	**39.5** (11)	1.51 s	1.84 s	30 620	42.0 (11)	**1.20 s**	**1.16 s**
With rotational coarse basis functions (3 rigid body modes)								
	GDSW				RGDSW			
MPI ranks	dim V_0	avg. its (nl its)	avg. setup	avg. solve	dim V_0	avg. its (nl its)	avg. setup	avg. solve
512	6 897	**35.5** (11)	15.77 s	**11.21 s**	2 751	40.7 (11)	**15.23 s**	12.22 s
1 024	13 959	**35.6** (11)	6.16 s	**5.78 s**	5 610	42.9 (11)	**5.65 s**	6.04 s
2 048	27 951	**33.5** (11)	3.78 s	3.45 s	11 379	42.2 (11)	**3.17 s**	**2.81 s**
4 096	55 776	**31.8** (11)	2.21 s	3.80 s	22 986	44.3 (11)	**1.95 s**	**2.70 s**
8 192	109 446	**29.3** (11)	2.49 s	5.33 s	45 930	40.8 (11)	**1.19 s**	**3.13 s**

Problem: Velocity **Mesh:** Antarctica, 4 km hor. resolution, 20 vert. layers **Size:** 35.3 m degrees of freedom (P1 FE)

- 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.



Antarctica mesh & domain decomposition.

MPI ranks	mesh	# dofs	1 OpenMP thread avg. its (nl its)	avg. setup	avg. solve	4 OpenMP threads avg. its (nl its)	avg. setup	avg. solve
32	16 km	2.2 m	24.1 (11)	11.97 s	9.47 s	23.5 (11)	4.15 s	3.25 s
128	8 km	8.8 m	32.0 (10)	14.08 s	8.71 s	32.0 (10)	4.97 s	2.85 s
512	4 km	35.3 m	42.6 (11)	14.99 s	12.50 s	42.6 (11)	5.50 s	4.02 s
2 048	2 km	141.5 m	61.0 (11)	22.83 s	19.76 s	61.0 (11)	7.36 s	6.55 s
8 192	1 km	566.1 m	67.1 (14)	17.36 s	22.91 s	67.1 (14)	6.20 s	7.39 s

Problem: Velocity **Meshes:** Antarctica 20 vert. layers **Discretization:** P1 FE **Coarse space:** RGDSW

Temperature Problem

The **steady state enthalpy equation** reads

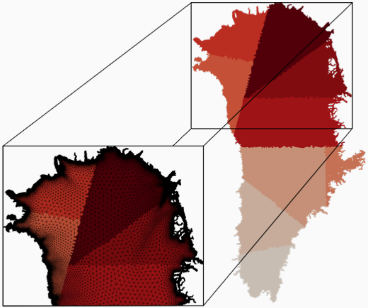
$$\nabla \cdot \boldsymbol{q}(h) + \boldsymbol{u} \cdot \nabla h = 4\mu\, \epsilon_e^2$$

with the enthalpy growing linearly with the water content ϕ

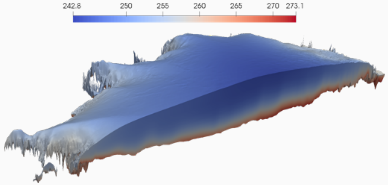
$$h = \begin{cases} \rho_i c\,(T - T_0), & \text{for cold ice } (h \le h_m), \\ h_m + \rho_w L\, \phi, & \text{for temperate ice.} \end{cases}$$

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

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



Greenland mesh & domain decomposition.



Temperature T solution

Water flux term

The **water flux term**

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

describes the percolation of water driven by gravity; cf. **Schoof and Hewitt (2016, 2017)**.

See **Perego et al. (in preparation)** and **Heinlein et. al (submitted 2021)** for more details.

The **steady state enthalpy equation** reads

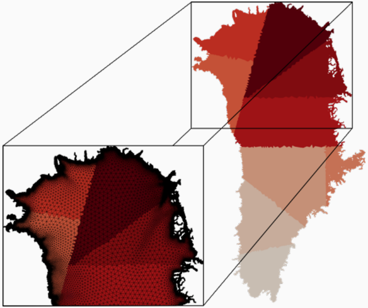
$$\nabla \cdot \boldsymbol{q}(h) + \boldsymbol{u} \cdot \nabla h = 4\mu\,\epsilon_e^2$$

with the enthalpy growing linearly with the water content ϕ

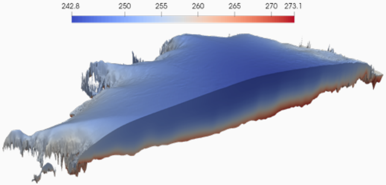
$$h = \begin{cases} \rho_i c\,(T - T_0), & \text{for cold ice } (h \le h_m), \\ h_m + \rho_w L\,\phi, & \text{for temperate ice.} \end{cases}$$

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

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



Greenland mesh & domain decomposition.



Temperature T solution

Boundary conditions

- *Upper surface*: $h = \rho_i c(T_s - T_0)$
  (**Dirichlet boundary condition**)
- *Bed*: $m = G + \beta\sqrt{u^2 + v^2} - k\nabla T \cdot \boldsymbol{n}$,
  $m\,(T - T_m) = 0,\ T_m \le 0.$
  (**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						
	one layer of algebraic overlap			two layers of algebraic overlap		
MPI ranks	avg. its	avg. setup	avg. solve	avg. its	avg. setup	avg. solve
512	18.1 (11)	**0.42 s**	**0.35 s**	**17.1** (11)	0.51 s	0.40 s
1 024	23.7 (11)	**0.25 s**	**0.25 s**	**22.1** (11)	0.27 s	0.27 s
2 048	29.6 (11)	**0.16 s**	**0.17 s**	**27.6** (11)	0.23 s	0.20 s
4 096	39.8 (11)	**0.15 s**	**0.15 s**	**35.6** (11)	0.17 s	0.17 s
RGDSW preconditioner						
	one layer of algebraic overlap			two layers of algebraic overlap		
MPI ranks	avg. avg. its	avg. setup	avg. solve	avg. avg. its	avg. setup	avg. solve
512	19.5 (11)	**0.44 s**	**0.41 s**	**18.7** (11)	0.55 s	0.46 s
1 024	25.2 (11)	**0.28 s**	**0.29 s**	**23.9** (11)	0.35 s	0.33 s
2 048	31.5 (11)	0.26 s	**0.24 s**	**29.5** (11)	**0.25 s**	0.27 s
4 096	42.2 (11)	**0.25 s**	**0.27 s**	**38.2** (11)	**0.25 s**	0.29 s

Problem: Temperature **Mesh:** Greenland
1-10 km hor. resolution
20 vert. layers **Size:** 1.9 m degrees of freedom (P1 FE)

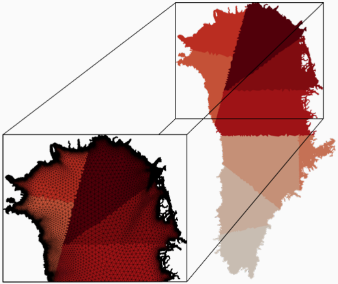
Coupled Problem

Couple the velocity and temperature problems. Therefore, compute the vertical velocity w using the incompressibility condition

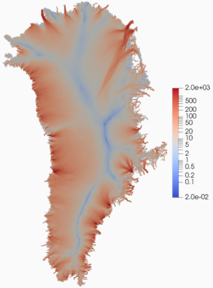
$$\partial_x u + \partial_y v + \partial_z w = 0,$$

with the **Dirichlet boundary condition at the ice lower surface**

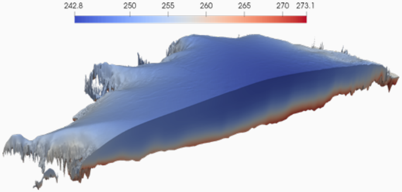
$$\boldsymbol{u} \cdot \boldsymbol{n} = \frac{m}{L\,(\rho_i - \rho_w \phi)}.$$



Greenland mesh & domain decomposition.



Velocity u solution



Temperature T solution

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

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

See **Perego et al. (in preparation)** and **Heinlein, Perego, Rajamanickam (submitted 2021)** for more details.

Monolithic (R)GDSW Preconditioners for CFD Simulations

Monolithic GDSW preconditioner

Consider the discrete saddle point problem

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

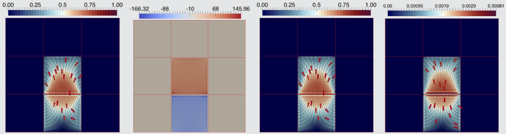
We construct a **monolithic GDSW preconditioner**

$$\mathcal{M}_{\mathrm{GDSW}}^{-1} = \phi \mathcal{A}_0^{-1} \phi^T + \sum\nolimits_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i^{-1} \mathcal{R}_i,$$

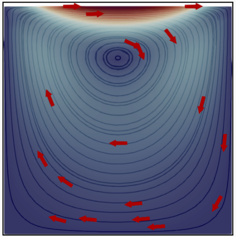
with block matrices $\mathcal{A}_0 = \phi^T \mathcal{A} \phi$, $\mathcal{A}_i = \mathcal{R}_i \mathcal{A} \mathcal{R}_i^T$, and

$$\mathcal{R}_i = \begin{bmatrix} \mathcal{R}_{u,i} & 0 \\ 0 & \mathcal{R}_{p,i} \end{bmatrix} \quad \text{and} \quad \phi = \begin{bmatrix} \Phi_{u,u_0} & \Phi_{u,p_0} \\ \Phi_{p,u_0} & \Phi_{p,p_0} \end{bmatrix}.$$

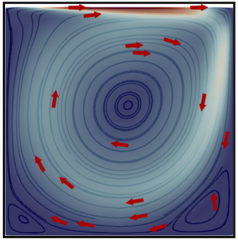
Using $\mathcal{A}$ to compute extensions: $\phi_I = -\mathcal{A}_{II}^{-1} \mathcal{A}_{I\Gamma} \phi_\Gamma$; cf. **Heinlein, Hochmuth, Klawonn (2019, 2020)**.



Φ_{u,u_0} Φ_{p,u_0} Φ_{u,p_0} Φ_{p,p_0}



Stokes flow



Navier–Stokes flow

Related work:

- Original work on monolithic Schwarz preconditioners: **Klawonn and Pavarino (1998, 2000)**
- 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

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

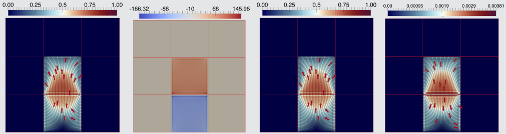
We construct a **monolithic GDSW preconditioner**

$$\mathcal{M}_{\mathrm{GDSW}}^{-1} = \phi \mathcal{A}_0^{-1} \phi^T + \sum\nolimits_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i^{-1} \mathcal{R}_i,$$

with block matrices $\mathcal{A}_0 = \phi^T \mathcal{A} \phi$, $\mathcal{A}_i = \mathcal{R}_i \mathcal{A} \mathcal{R}_i^T$, and

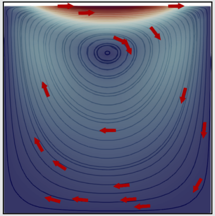
$$\mathcal{R}_i = \begin{bmatrix} \mathcal{R}_{u,i} & 0 \\ 0 & \mathcal{R}_{p,i} \end{bmatrix} \quad \text{and} \quad \phi = \begin{bmatrix} \Phi_{u,u_0} & \Phi_{u,p_0} \\ \Phi_{p,u_0} & \Phi_{p,p_0} \end{bmatrix}.$$

Using $\mathcal{A}$ to compute extensions: $\phi_I = -\mathcal{A}_{II}^{-1} \mathcal{A}_{I\Gamma} \phi_\Gamma$; cf. Heinlein, Hochmuth, Klawonn (2019, 2020).

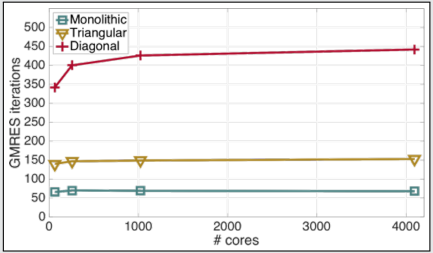


Φ_{u,u_0} Φ_{p,u_0} Φ_{u,p_0} Φ_{p,p_0}

Monolithic vs Block Preconditioners



Stokes flow



Computations performed on magnitUDE, University Duisburg-Essen.

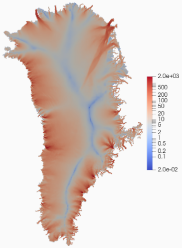
Monolithic (R)GDSW Preconditioners for Multiphysics Land Ice Simulations

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

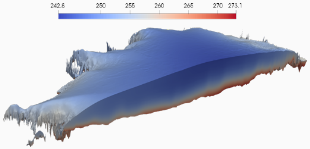
$$\mathcal{M}_{\mathrm{GDSW}}^{-1} = \phi \mathcal{A}_0^{-1} \phi^T + \sum\nolimits_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i^{-1} \mathcal{R}_i,$$

for the tangent matrix of the coupled problem

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



Velocity u solution



Temperature T solution

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:

$$r_{u,1} := \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \; r_{u,2} := \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \; r_{u,3} := \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \; r_{u,4} := \begin{bmatrix} y \\ -x \\ 0 \\ 0 \end{bmatrix}, \text{ and } r_T := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

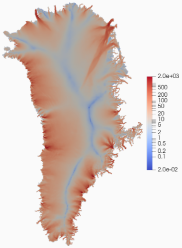
See **Heinlein, Perego, Rajamanickam (submitted 2021)** for more details.

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

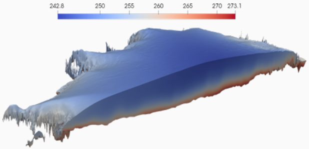
$$\mathcal{M}_{\mathrm{GDSW}}^{-1} = \phi \mathcal{A}_0^{-1} \phi^T + \sum\nolimits_{i=1}^{N} \mathcal{R}_i^T \mathcal{A}_i^{-1} \mathcal{R}_i,$$

for the tangent matrix of the coupled problem

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



Velocity u solution



Temperature T solution

Fully coupled extensions

We compute coarse basis function using extensions

$$\phi = \begin{bmatrix} -\mathcal{A}_{II}^{-1} \mathcal{A}_{\Gamma I}^T \Phi_\Gamma \\ \Phi_\Gamma \end{bmatrix} = \begin{bmatrix} \phi_I \\ \phi_\Gamma \end{bmatrix}$$

based on the coupled matrix $\mathcal{A}$.

See **Heinlein, Perego, Rajamanickam (submitted 2021)** for more details.

Decoupled extensions

We compute coarse basis function using extensions

$$\phi = \begin{bmatrix} -\tilde{\mathcal{A}}_{II}^{-1} \tilde{\mathcal{A}}_{\Gamma I}^T \Phi_\Gamma \\ \Phi_\Gamma \end{bmatrix} = \begin{bmatrix} \phi_I \\ \phi_\Gamma \end{bmatrix}$$

based on the decoupled matrix

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

Greenland Coupled Problem – Coarse Spaces

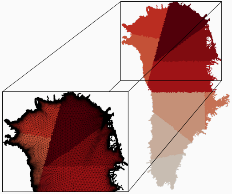
fully coupled extensions

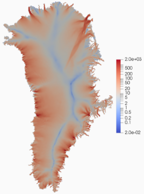
MPI ranks	dim V_0	no reuse: avg. its (nl its)	no reuse: avg. setup	no reuse: avg. solve	reuse coarse basis: avg. its (nl its)	reuse coarse basis: avg. setup	reuse coarse basis: avg. solve
256	1 400	100.1 (27)	4.10 s	6.40 s	**18.5** (70)	**2.28 s**	**1.07 s**
512	2 852	129.1 (28)	1.88 s	4.20 s	**24.6** (38)	**1.04 s**	**0.70 s**
1 024	6 036	191.2 (65)	1.21 s	4.76 s	**34.2** (32)	**0.66 s**	**0.70 s**
2 048	12 368	237.4 (30)	0.96 s	4.06 s	**37.3** (30)	**0.60 s**	**0.58 s**

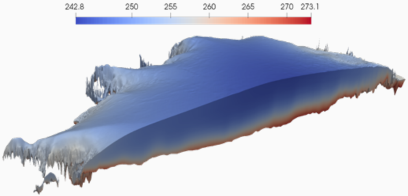
decoupled extensions

MPI ranks	dim V_0	no reuse: avg. its (nl its)	no reuse: avg. setup	no reuse: avg. solve	reuse coarse basis: avg. its (nl its)	reuse coarse basis: avg. setup	reuse coarse basis: avg. solve
256	1 400	23.6 (29)	3.90 s	1.32 s	**21.5** (34)	**2.23 s**	**1.18 s**
512	2 852	27.5 (30)	1.83 s	0.78 s	**26.4** (33)	**1.13 s**	**0.78 s**
1 024	6 036	30.1 (29)	1.19 s	**0.60 s**	**28.6** (43)	**0.66 s**	0.61 s
2 048	12 368	36.4 (30)	0.69 s	0.56 s	**31.2** (50)	**0.57 s**	**0.55 s**

Problem: Coupled **Mesh:** Greenland
3-30 km hor. resolution
20 vert. layers
Size: 7.5 m degrees of freedom (P1 FE) **Coarse space:** RGDSW







MPI ranks	**decoupled** (no reuse) avg. its. (nl its)	avg. setup	avg. solve	**fully coupled** (reuse coarse basis) avg. its (nl its)	avg. setup	avg. solve	**decoupled** (reuse 1st level symb. fact. + coarse basis) avg. its (nl its)	avg. setup	avg. solve
512	**41.3** (36)	18.78 s	**4.99** s	45.3 (32)	11.84 s	5.35 s	45.0 (35)	**10.53** s	5.36 s
1 024	53.0 (29)	8.68 s	4.22 s	**47.8** (37)	5.36 s	**3.82** s	54.3 (32)	**4.59** s	4.31 s
2 048	62.2 (86)	4.47 s	4.23 s	66.7 (38)	2.81 s	4.53 s	**59.1** (38)	**2.32** s	**3.99** s
4 096	**68.9** (40)	2.52 s	**2.86** s	79.1 (36)	1.61 s	3.30 s	78.7 (38)	**1.37** s	3.30 s

Problem: Coupled **Mesh:** Greenland 1-10 km hor. resolution 20 vert. layers **Size:** 68.6 m degrees of freedom (P1 FE) **Coarse space:** RGDSW

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

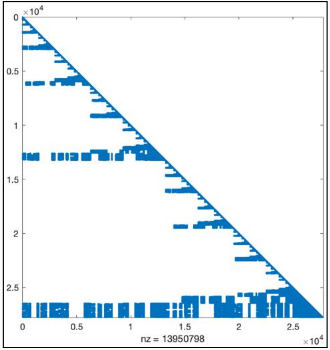
The sparse triangular solver is an **important kernel** in many codes (including `FROSch`) but is **challenging to parallelize**

- 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**

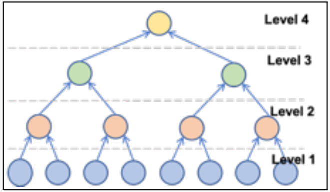
Parallel supernode-based triangular solver:

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.



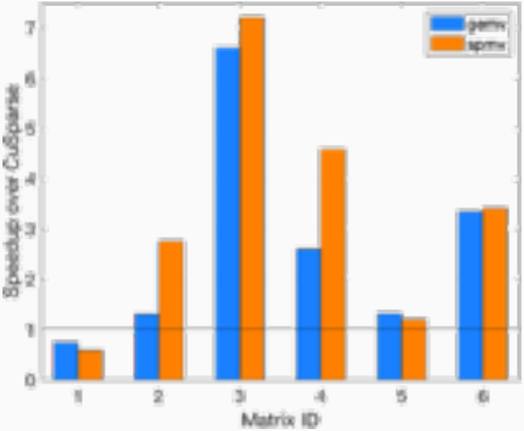
Lower-triangular matrix – SuperLU with METIS nested dissection ordering



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**.

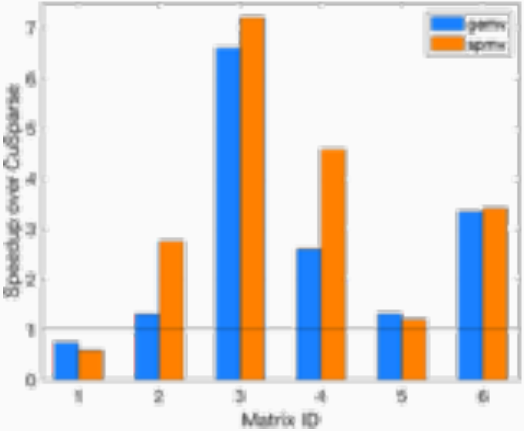
NVIDIA V100 GPU



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

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

NVIDIA V100 GPU



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

$\rightarrow$ Performance depends on **number of levels** and **sizes of supernodes**.

Preliminary Strong Scaling Results

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

		42 MPI ranks per node		6 MPI ranks, 6 GPUs per node		12 MPI ranks, 6 GPUs per node		42 MPI ranks, 6 GPUs per node	
1 node	its (nl its)	42	**171**	6	OOM	12	OOM	42	OOM
	subd. setup time		**506.7 s**						
	solve time		**29.3 s**						
2 node	its (nl its)	96	200 (9)	12	**156** (9)	24	165 (9)	96	OOM
	subd. setup time		**165.9 s**		2 892.7 s		884.8 s		
	solve time		15.3 s		5.9 s		**5.3 s**		
3 node	its (nl its)	126	208 (9)	18	**178** (9)	36	180 (9)	126	208 (9)
	subd. setup time		**88.1 s**		1 485.3 s		506.3 s		96.1 s
	solve time		9.7 s		4.0 s		**3.7 s**		10.7 s
4 node	its (nl its)	168	217 (9)	24	**165** (9)	48	197 (9)	168	217 (9)
	subd. setup time		**49.0 s**		848.0 s		307.7 s		59.9 s
	solve time		7.0 s		**3.2 s**		3.3 s		6.6 s

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

Preliminary Strong Scaling Results

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

		42 MPI ranks per node		6 MPI ranks, 6 GPUs per node		12 MPI ranks, 6 GPUs per node		42 MPI ranks, 6 GPUs per node	
1 node	its (nl its)	42	**171**	6	OOM	12	OOM	42	OOM
	subd. setup time		**506.7 s**						
	solve time		**29.3 s**						
2 node	its (nl its)	96	200 (9)	12	**156** (9)	24	165 (9)	96	OOM
	subd. setup time		**165.9 s**		2 892.7 s		884.8 s		
	solve time		15.3 s		5.9 s		**5.3 s**		
3 node	its (nl its)	126	208 (9)	18	**178** (9)	36	180 (9)	126	208 (9)
	subd. setup time		**88.1 s**		1 485.3 s		506.3 s		96.1 s
	solve time		9.7 s		4.0 s		**3.7 s**		10.7 s
4 node	its (nl its)	168	217 (9)	24	**165** (9)	48	197 (9)	168	217 (9)
	subd. setup time		**49.0 s**		848.0 s		307.7 s		59.9 s
	solve time		7.0 s		**3.2 s**		3.3 s		6.6 s

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

		42 MPI ranks per node		6 MPI ranks, 6 GPUs per node		12 MPI ranks, 6 GPUs per node		42 MPI ranks, 6 GPUs per node	
1 node	its (nl its)	42	**171**	6	OOM	12	OOM	42	OOM
	subd. setup time		**506.7 s**						
	solve time		**29.3 s**						
2 node	its (nl its)	96	200 (9)	12	**156** (9)	24	165 (9)	96	OOM
	subd. setup time		**165.9 s**		2 892.7 s		884.8 s		
	solve time		15.3 s		5.9 s		**5.3 s**		
3 node	its (nl its)	126	208 (9)	18	**178** (9)	36	180 (9)	126	208 (9)
	subd. setup time		**88.1 s**		1 485.3 s		506.3 s		96.1 s
	solve time		9.7 s		4.0 s		**3.7 s**		10.7 s
4 node	its (nl its)	168	217 (9)	24	**165** (9)	48	197 (9)	168	217 (9)
	subd. setup time		**49.0 s**		848.0 s		307.7 s		59.9 s
	solve time		7.0 s		**3.2 s**		3.3 s		6.6 s

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 × NVIDIA V100 GPUs and 2 × 21-core Power9 per node

		42 MPI ranks per node		6 MPI ranks, 6 GPUs per node		12 MPI ranks, 6 GPUs per node		42 MPI ranks, 6 GPUs per node	
1 node	its (nl its)	42	**171**	6	OOM	12	OOM	42	OOM
	subd. setup time		**506.7 s**						
	solve time		**29.3 s**						
2 node	its (nl its)	96	200 (9)	12	**156** (9)	24	165 (9)	96	OOM
	subd. setup time		**165.9 s**		2 892.7 s		884.8 s		
	solve time		15.3 s		5.9 s		**5.3 s**		
3 node	its (nl its)	126	208 (9)	18	**178** (9)	36	180 (9)	126	208 (9)
	subd. setup time		**88.1 s**		1 485.3 s		506.3 s		96.1 s
	solve time		9.7 s		4.0 s		**3.7 s**		10.7 s
4 node	its (nl its)	168	217 (9)	24	**165** (9)	48	197 (9)	168	217 (9)
	subd. setup time		**49.0 s**		848.0 s		307.7 s		59.9 s
	solve time		7.0 s		**3.2 s**		3.3 s		6.6 s

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 precondititoners**).
- 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 ressources:** Cori (NERSC), JUQUEEN (JSC), magnitUDE (UDE), Summit (ORNL)