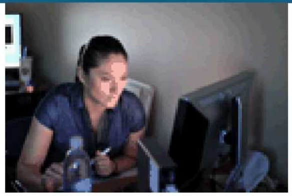
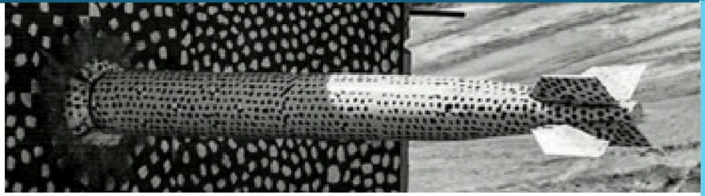
SAND2019-4533PE

Using Kokkos to Manage Memory and Parallelism for Method of Moments





PRESENTED BY

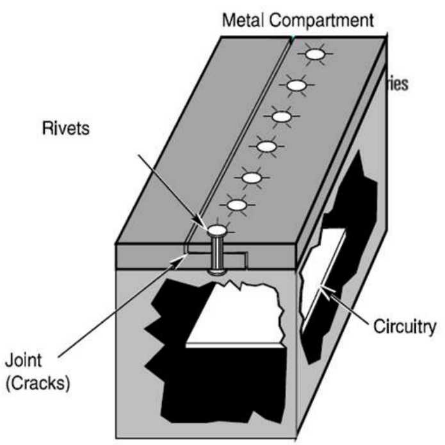
Brian Zinser

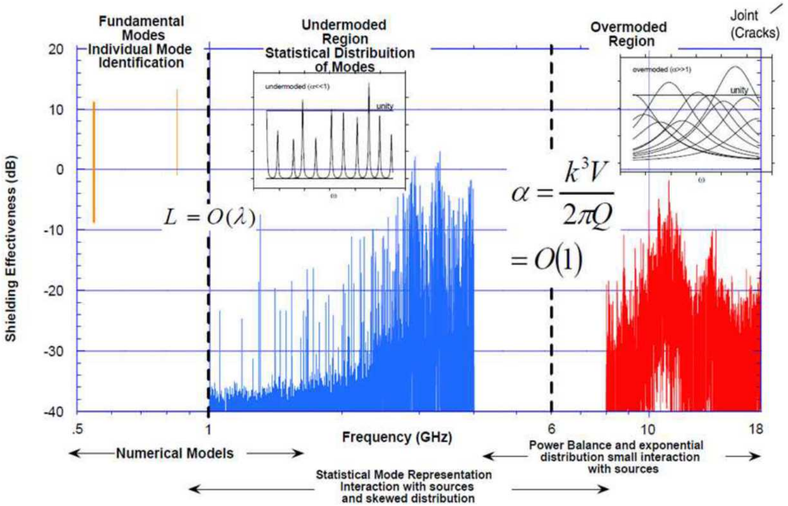
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Electromagnetic Radiation Coupling

Overview and Motivation for Cavity Resonance Dampeners

- Metallic cavities/cases are often used to protect electrical circuits and systems from harsh EM environments
  - ➢ External fields can couple into the cavity through mechanical joints, seams, and rivets
  - ➢ For high-Q resonant cavities the internal field levels can be much higher than the incoming field, which can disrupt circuit operation
  - ➢ Reducing EM coupling paths can be challenging due to often opposing mechanical and electrical requirements





L. K. Warne et al., Electromagnetics, 1-28, 2017

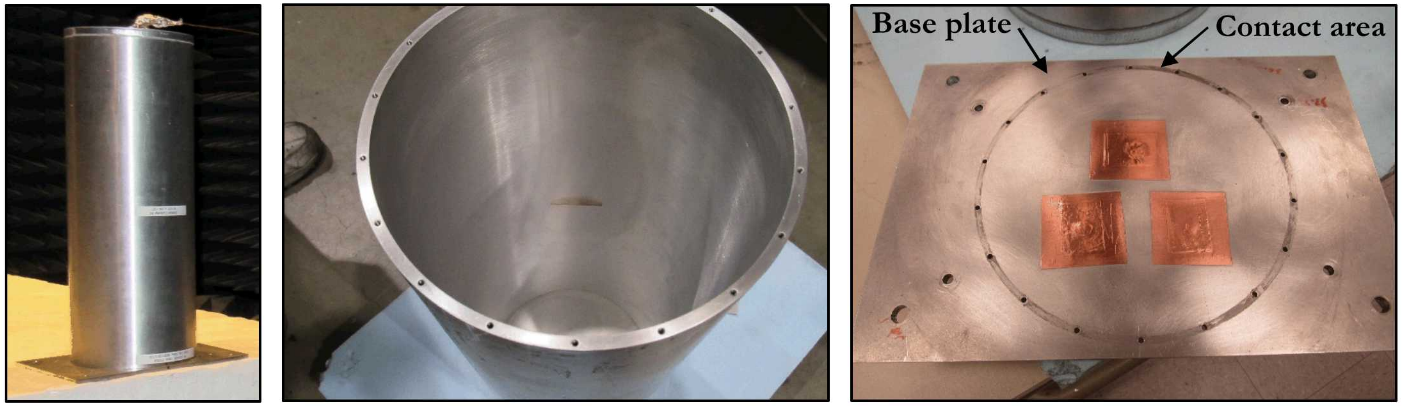
Shielding effectiveness (SE) measures the reduction of the electromagnetic field at a given point in space caused by placing a shield between the source and that point.

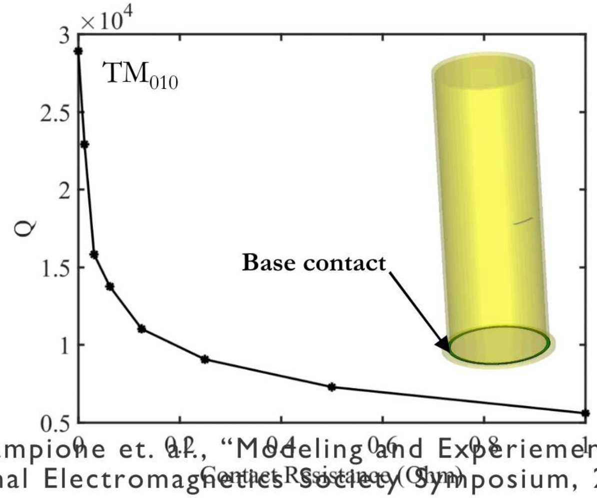
$$SE = 20\log\left(\frac{|\mathbf{E}|}{|E_i|}\right)$$

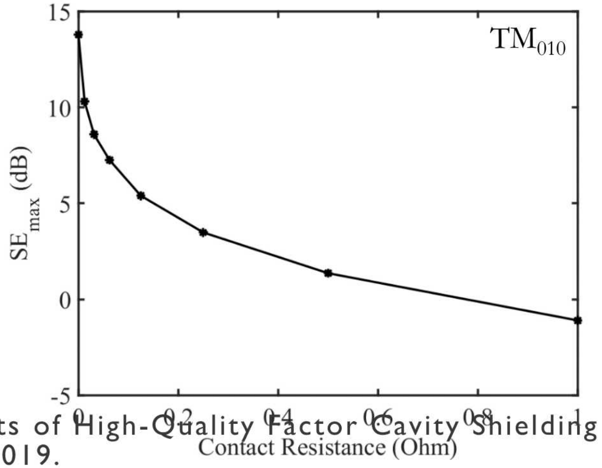
- Model resonant cavities with bounding methods and investigate the use of EM absorbing materials within a cavity to dampen the internal fields
- EM absorbers have been proven to reduce internal field levels → broadband, cavity insensitive solution
  - ➢ D. Williams, IEEE Trans. Microwave Theory Tech. **37**, 253-256 (1989)
  - ➢ P. Dixon, IEEE Microwave Magazine **6**, 74-84 (2005)
  - ➢ S. Campione et al., Sandia National Laboratories Report, SAND2018-10548, Albuquerque, NM (2018)

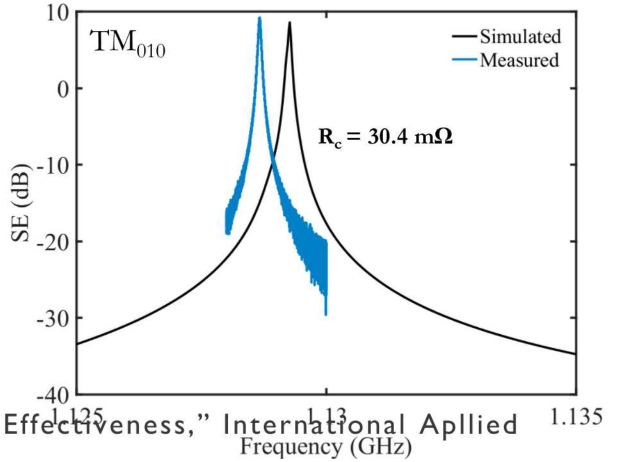
Cylinder Joint Resistance

- Fabricated cylinder has joint at the bottom where the base plate is screwed in
- This joint has some contact resistance that was not initially included in the model/simulations
- A reasonable fit with the TM_{010} measurements is achieved with only 30.4 mΩ of joint resistance
- High-Q cavities are very sensitive to additional resistance between joints









Salvatore Campione et. al., "Modeling and Experiements of High-Quality Factor Cavity Shielding Effectiveness," International Apllied Computational Electromagnetics Society Symposium, 2019.

Method of Moments(MoM) Formulation

MoM– boundary element method so # of unknowns for a 2D mesh, not a 3D mesh

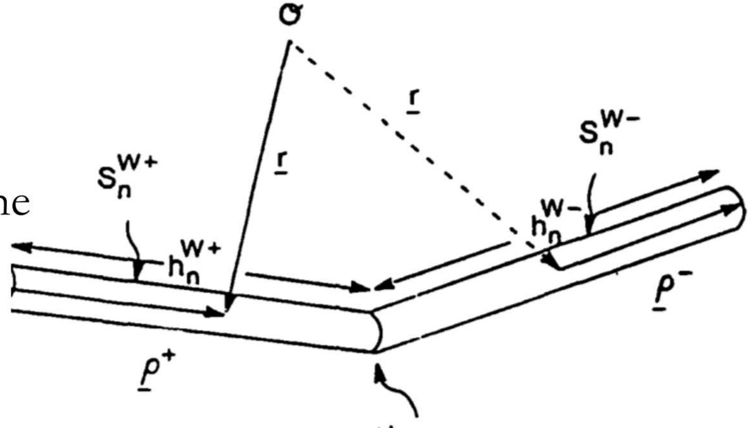
Dense, complex valued matrix – O(N^2) memory to store matrix – memory bound at system memory level

Prefer to solve by Pliris LU solver – O(N^3) computations

EFIE, MFIE, CFIE, Dielectrics (sphere from Rob's Jin-Fa slide) for different regions – requires branching or sorting

For example,

- Omitting constants, the EFIE matrix entry $A_{T,S} = \iint G(\Lambda^T \cdot \Lambda^S + \nabla \cdot \Lambda^T * \nabla \cdot \Lambda^S)$.
- The Galerkin test and source basis functions are $\Lambda(r) = \frac{r - v_0}{|h_0|}$ with divergences $\nabla \cdot \Lambda_0 = \frac{2}{|h_0|}$.
- $G(r, r') = \frac{e^{-jk|r-r'|}}{4\pi|r-r'|}$ is the Green's function, which causes the matrix to be dense and complex; it also requires branching or sorting in the algorithm to deal with the singularity.
- 3 test (first integral) and 7 source (second integral) quadrature points

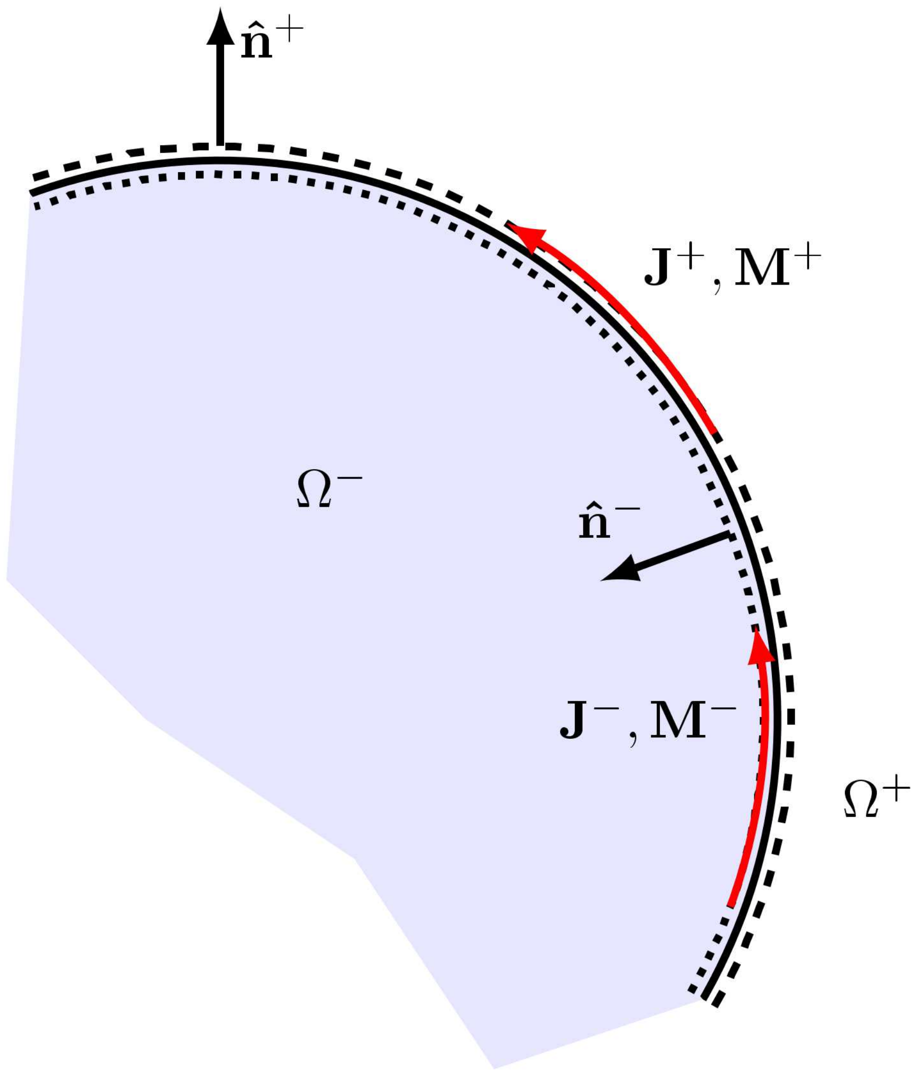


BEM Solution of Multi-Region Problems

- Enforce tangential continuity of $\mathbf{E}$ and $\mathbf{H}$ on boundary
- Convenient to define $(\mathbf{J}^+, \mathbf{M}^+) = -(\mathbf{J}^-, \mathbf{M}^-)$
- Yields combined field formulation

$$\begin{aligned} \text{EFIE}^+ + \alpha\text{EFIE}^- \\ \text{MFIE}^+ + \beta\text{MFIE}^- \end{aligned} \qquad (2)$$

- Choice of α, β affects accuracy, conditioning
  - $\alpha = \beta = 1$ gives PMCHWT formulation
  - $\alpha = \frac{\epsilon_r^-}{\epsilon_r^+}, \beta = \frac{\mu_r^-}{\mu_r^+}$ gives Müller formulation



Implementation

Avoid MPI during fill / Pliris has its own MPI implementation

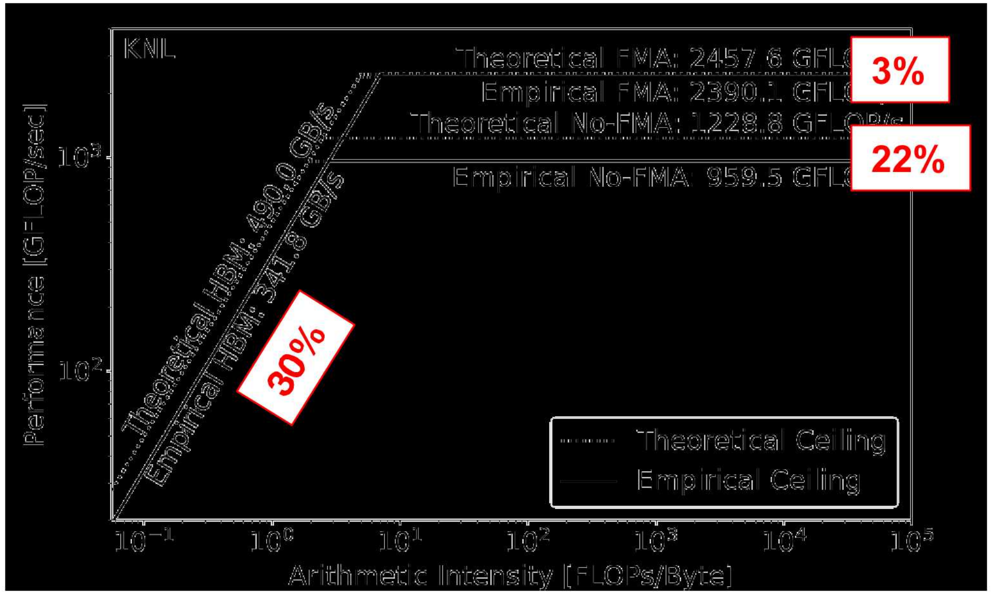
Targeting 1 MPI ranks per CPU socket or per CPU node, 4 MPI ranks per KNL, 1 MPI rank per GPU

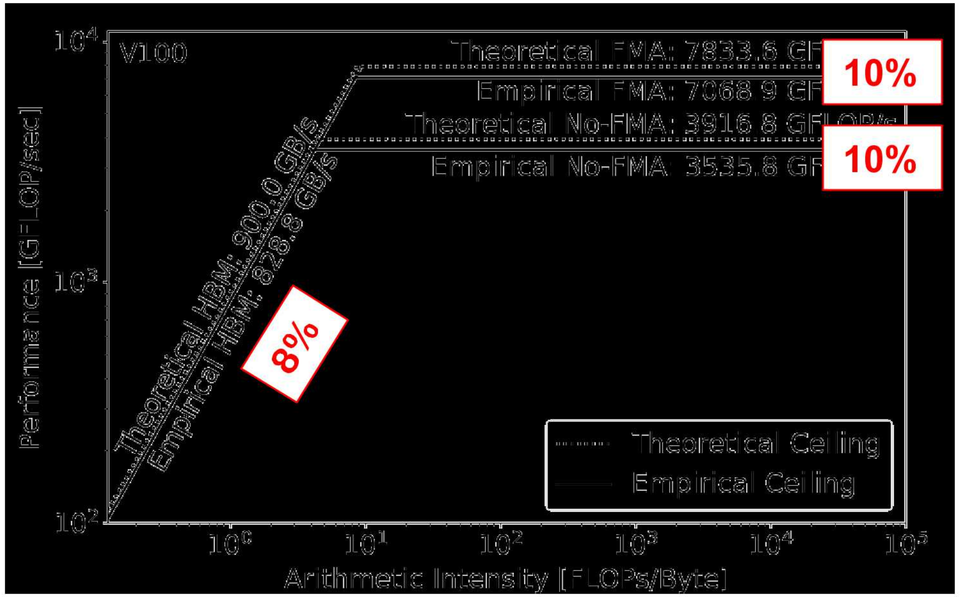
Kokkos for parallelism on a single MPI rank

- Handling multiple memory levels is very important for us. Memory spaces:
  - Kokkos scratch level 0: shared memory or L1 cache
  - Kokkos scratch level 1: L3 or HBM
  - DeviceSpace: Global memory or system memory
  - HostSpace: System memory

MoM EFIE GPU Implementation

Theoretical and Empirical Rooflines for KNLs and V100s





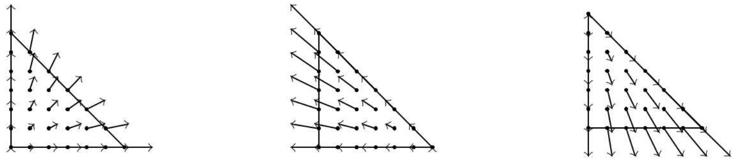
KNL FMA required arithmetic intensity:
$\frac{2390.1}{341.8}$ FLOPs/byte ≈ 7 FLOPs/byte

V100 FMA required arithmetic intensity:
$\frac{7068.9}{828.8}$ FLOPs/byte ≈ 8.5 FLOPs/byte

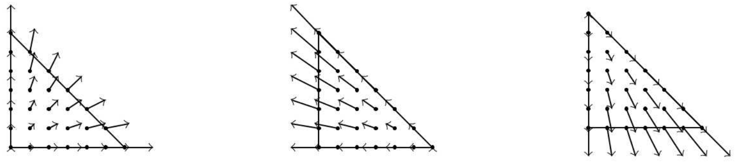
John Pennycook, Charlene Yang, and Jack Deslippe, "Quantitatively Assessing Performance Portability with Roofline", NeRSC. https://ideas-productivity.org/events/hpc-best-practices-webinars/#webinar025

MoM EFIE Arithmetic Intensity when Considering Triangle Pairs

Test triangle:



Source triangle:

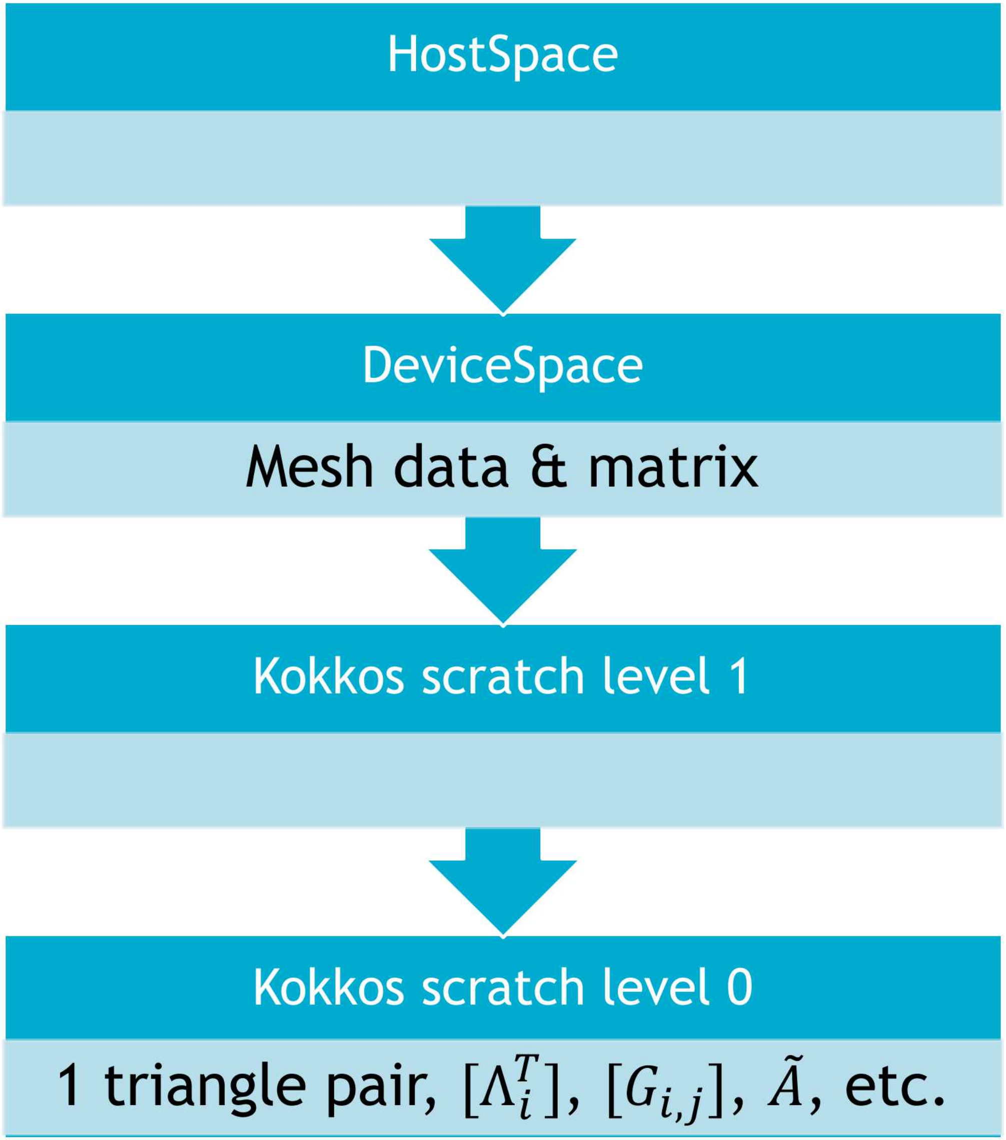


- **Fills 1/4th of 9 EFIE matrix entries:** Compute the 4D integral $\iint G(\Lambda^T \cdot \Lambda^S + \nabla \cdot \Lambda^T * \nabla \cdot \Lambda^S)$ over an triangle pair where T and S loop through the 3 half-basis functions on each triangle.
- Assume the triangle pair requires loading 376 bytes = 6 vertices (18 doubles) + eps & mu (2 complexes) + connectivity information (14 ints) + 9 matrix contributes (9 complexes, atomic scatter)
- With some reuse of data, the arithmetic intensity is $\frac{2335}{376}$ FLOPs/byte $\approx$ 6.2 FLOPs/byte

	add, sub, mul (1 FLOPs)	div (4 FLOPs)	exp, sincos, sqrt (8 FLOPs)	estimated FLOPs	Total for 3x7 points and 3 half-basis
$\Lambda(r)$	16	2		24	720 (= FLOPs x 10 x 3)
$\nabla \cdot \Lambda$		1		4	12 (= FLOPs x 3)
$G(r, r')$	7	2	3	39	819 (= FLOPs x 21)
$G(\Lambda^T \cdot \Lambda^S + \nabla \cdot \Lambda^T * \nabla \cdot \Lambda^S)$	8			8	504 (= FLOPs x 21 x 3)
Elemental mapping	28			28	280 (= FLOPs x 10)
Total	59	5	3	108	2335

Considering Triangle Pairs via Quadrature Inner Product

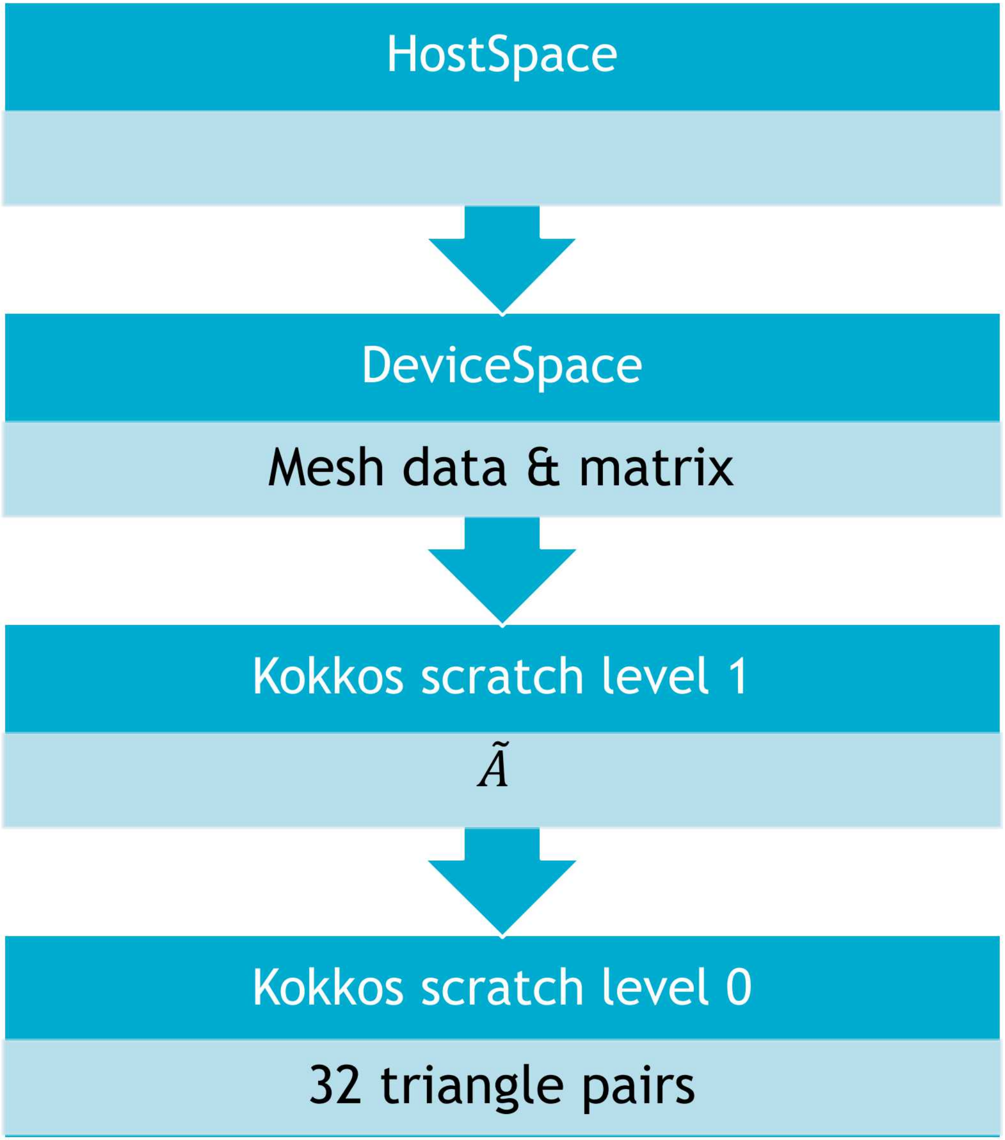
- Inner product over quadrature to form a 3x3 block of matrix contributions: $\tilde{A} = [\Lambda_1^T \quad \cdots \quad \Lambda_3^T] \begin{bmatrix} G_{1,1} & \cdots & G_{1,7} \\ \vdots & & \vdots \\ G_{3,1} & \cdots & G_{3,7} \end{bmatrix} \begin{bmatrix} \Lambda_1^S \\ \vdots \\ \Lambda_7^S \end{bmatrix}.$
- 3x3 block of complexes $\tilde{A}$ is scattered to nonconsecutive matix entries. It contains 18 doubles, i.e., 144 bytes.
- $[\Lambda_i^T]$, $[\Lambda_j^S]$, $[\nabla \cdot \Lambda^T]$, $[\nabla \cdot \Lambda^S]$, $[G_{i,j}]$ vectors and matrix contain 162 doubles, i.e., 1296 bytes.
- Requires 1528 bytes of Kokkos scratch level 0.
- GPU details: Only 1 triangle pair considered by a warp.



Considering Triangle Pairs via a Loop over Triangle Pairs

- Instead of storing the $[\Lambda_i^T]$, $[\Lambda_j^S]$, $[G_{i,j}]$ vectors and matrix required for $\tilde{A} = [\Lambda_1^T \quad \cdots \quad \Lambda_3^T] \begin{bmatrix} G_{1,1} & \cdots & G_{1,7} \\ \vdots & & \vdots \\ G_{3,1} & \cdots & G_{3,7} \end{bmatrix} \begin{bmatrix} \Lambda_1^S \\ \vdots \\ \Lambda_7^S \end{bmatrix}$, compute them on the fly.
- Requires 7424 bytes of Kokkos scratch level 0 and 4608 bytes of Kokkos scratch level 1.

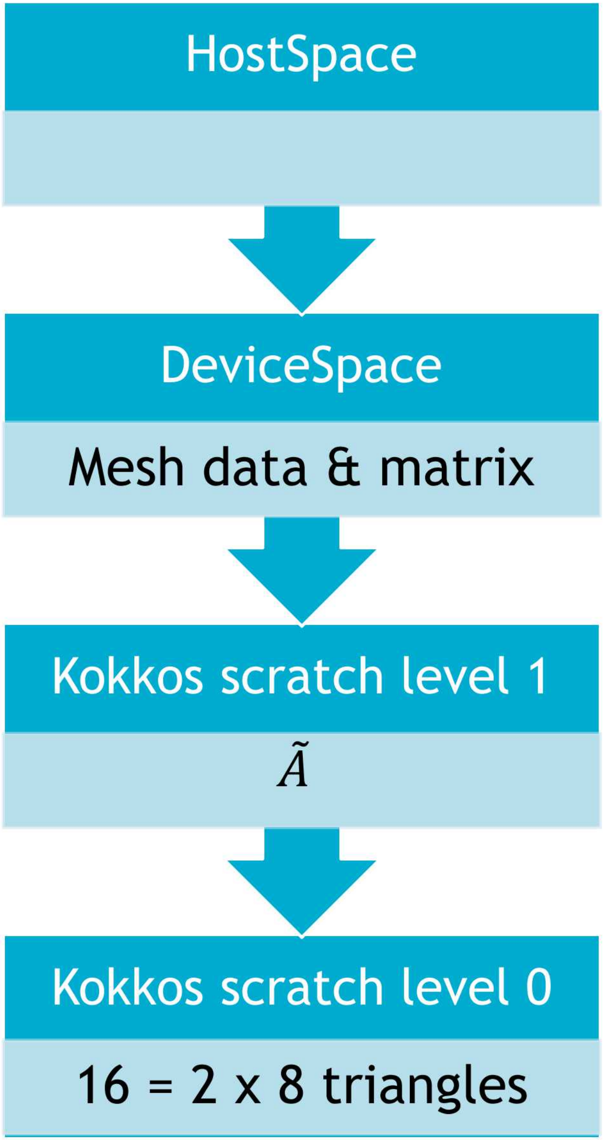
	Total FLOPs per triangle pair
$\Lambda(r)$	1512 (= FLOPs x 21 x 3)
$\nabla \cdot \Lambda$	252 (= FLOPs x 21 x 3)
$G(r, r')$	2457 (= FLOPs x 21 x 3)
$G(\Lambda^T \cdot \Lambda^S + \nabla \cdot \Lambda^T * \nabla \cdot \Lambda^S)$	504 (= FLOPs x 21 x 3)
Elemental mapping	1764 (= FLOPs x 21 x 3)
Total	6489



Considering Triangle Pairs via a Loop over Triangles

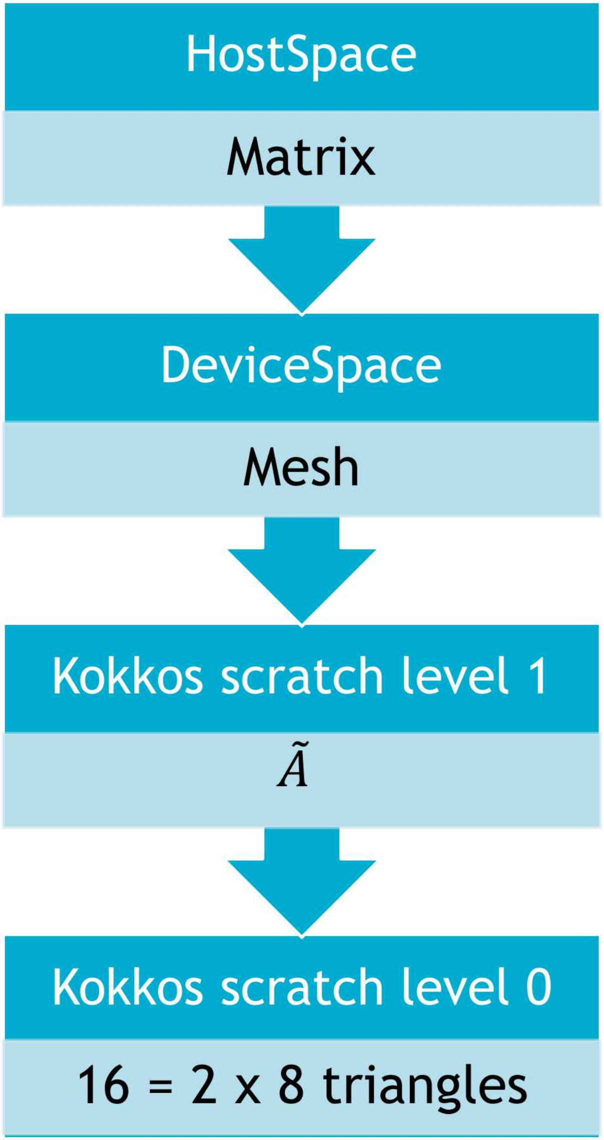
Thread	0	1	2	3	4	5	6	7	8	9	10
Test Unknown	40	41	42	43	44	45	46	47	40	41	42
Source Unknown	620	620	620	620	620	620	620	620	621	621	621

- Instead of loading 32 triangle pairs, load 8 test and 8 source triangles to make 64 triangle pairs on the fly.
- Reduces memory use in Kokkos scratch level 0 by reducing the required number of vertices, but maybe not the other information.
  - Previous slide's 32 triangle pairs require 192 vertices, which is 576 doubles or 4608 bytes.
  - This slide's 16 triangles require 48 vertices, which is 144 doubles or 1152 byes.
  - Savings of at least 3456 bytes.
- Required FLOPs largely unaffected.



Considering Triangle Pairs while Storing the Matrix in HostSpace

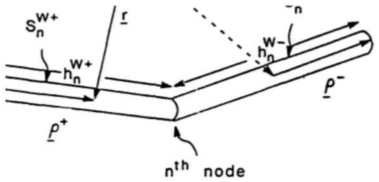
- Send $\tilde{A}$ to HostSpace with instructions on where to put it in the matrix.
- $\tilde{A}$ is $3x3$ and each entry goes to a different location in the matrix.
- Information required by the host to scatter $\tilde{A}$ is 216 bytes = 9 entries of $\tilde{A}$ (9 complexes) + 9 matrix coordinates (18 ints).
- GPU details:
  - Data reuse: the algorithm requires 2335 FLOPs. Its arithmetic intensity is $\frac{2335}{216}$ FLOPs/byte ≈ 10.8 FLOPs/byte.
  - No data reuse: the algorithm requires 6489 FLOPs. Its arithmetic intensity is $\frac{6489}{216}$ FLOPs/byte ≈ 30 FLOPs/byte.
  - Feasible if (the GPU max FLOPs divided by the GPU-HostSpace bandwidth) is less than the arithmetic intensity.
    - V100 with FMA: 8.5 FLOPs/byte
    - V100 without FMA: 4.3 FLOPs/byte



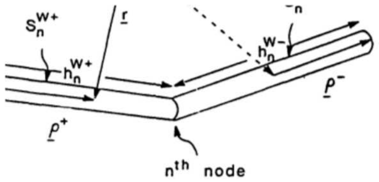
John Pennycook, Charlene Yang, and Jack Deslippe, “Quantitatively Assessing Performance Portability with Roofline”, NeRSC. https://ideas-productivity.org/events/hpc-best-practices-webinars/#webinar025

MoM EFIE Arithmetic Intensity when Considering Basis Pairs

Test basis:



Source basis:



- **Fills EFIE matrix entry $(\boldsymbol{T}, \boldsymbol{S})$:** Compute the 4D integral $\iint G(\Lambda^T \cdot \Lambda^S + \nabla \cdot \Lambda^T * \nabla \cdot \Lambda^S)$ over 4 triangle pairs, 2 of which support basis T and 2 of which support basis S.
- Assume for 4 triangle pairs, the triangle pair requires loading 652 bytes = 12 vertices (36 doubles) + eps & mu (2 complexes) + connectivity information (16 ints) + 1 matrix contributes (1 complex)
- With some reuse of data, the arithmetic intensity is $\frac{6044}{652}$ FLOPs/byte ≈ 9.3 FLOPs/byte

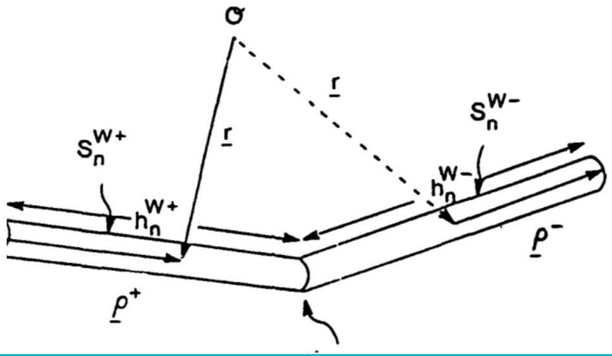
	add, sub, mul (1 FLOPs)	div (4 FLOPs)	exp, sincos, sqrt (8 FLOPs)	estimated FLOPs	Total for 3x7 points and 2x2 triangles
$\Lambda(r)$	16	2		24	960 (= FLOPs x 10 x 4)
$\nabla \cdot \Lambda$		1		4	16 (= FLOPs x 4)
$G(r, r')$	7	2	3	39	3276 (= FLOPs x 21 x 4)
$G(\Lambda^T \cdot \Lambda^S + \nabla \cdot \Lambda^T * \nabla \cdot \Lambda^S)$	8			8	672 (= FLOPs x 21 x 4)
Elemental mapping	28			28	1120 (= FLOPs x 10 x 4)
Total	59	5	3	108	6044

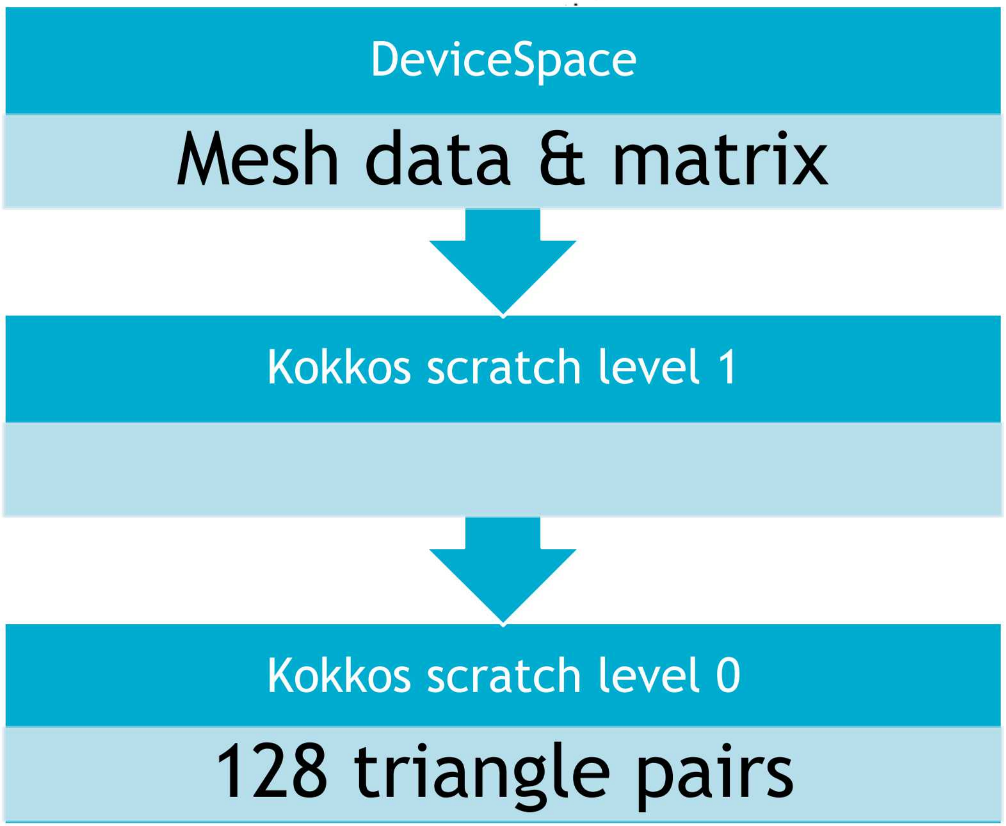
Considering Basis Pairs without Reuse

- Since each basis pair is supported by 4 triangle pairs, 32 basis pairs requires 128 triangle pairs, i.e., 20864 bytes in Kokkos scratch level 0.
- With T and S fixed for a single edge of each element, the contribution is still given by

$$[\Lambda_1^T \quad \cdots \quad \Lambda_3^T] \begin{bmatrix} G_{1,1} & \cdots & G_{1,7} \\ \vdots & & \vdots \\ G_{3,1} & \cdots & G_{3,7} \end{bmatrix} \begin{bmatrix} \Lambda_1^S \\ \vdots \\ \Lambda_7^S \end{bmatrix}$$

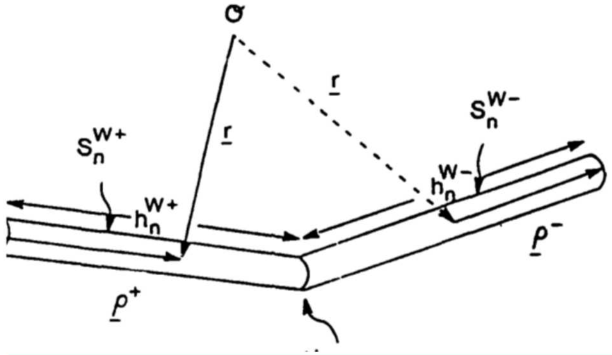
	Total FLOPs per basis pair (no reuse)
$\Lambda(r)$	2016 (= FLOPs x 21 x 4)
$\nabla \cdot \Lambda$	336 (= FLOPs x 21 x 4)
$G(r, r')$	3276 (= FLOPs x 21 x 4)
$G(\Lambda^T \cdot \Lambda^S + \nabla \cdot \Lambda^T * \nabla \cdot \Lambda^S)$	672 (= FLOPs x 21 x 4)
Elemental mapping	2352 (= FLOPs x 21 x 4)
Total	8652





Considering Basis Pairs Making Triangle Pairs on the Fly

- If not reusing data, consider the 4 triangle pairs supporting the basis pair simultaneously and make triangle pairs on the fly.
  - 16 triangles require 48 vertices = 144 doubles = 1152 byes.
  - 128 triangle pairs require 768 vertices = 2304 doubles = 18432 bytes.



Thread	0	1	2	3	4	5	6	7	8	9	10
Test Unknown	6	6	6	6	7	7	7	7	8	8	8
Test Element	+	-	+	-	+	-	+	-	+	-	+
Source Unknown	600	600	600	600	600	600	600	600	600	600	600
Source Element	+	+	-	-	+	+	-	-	+	+	-

DeviceSpace

Mesh data & matrix

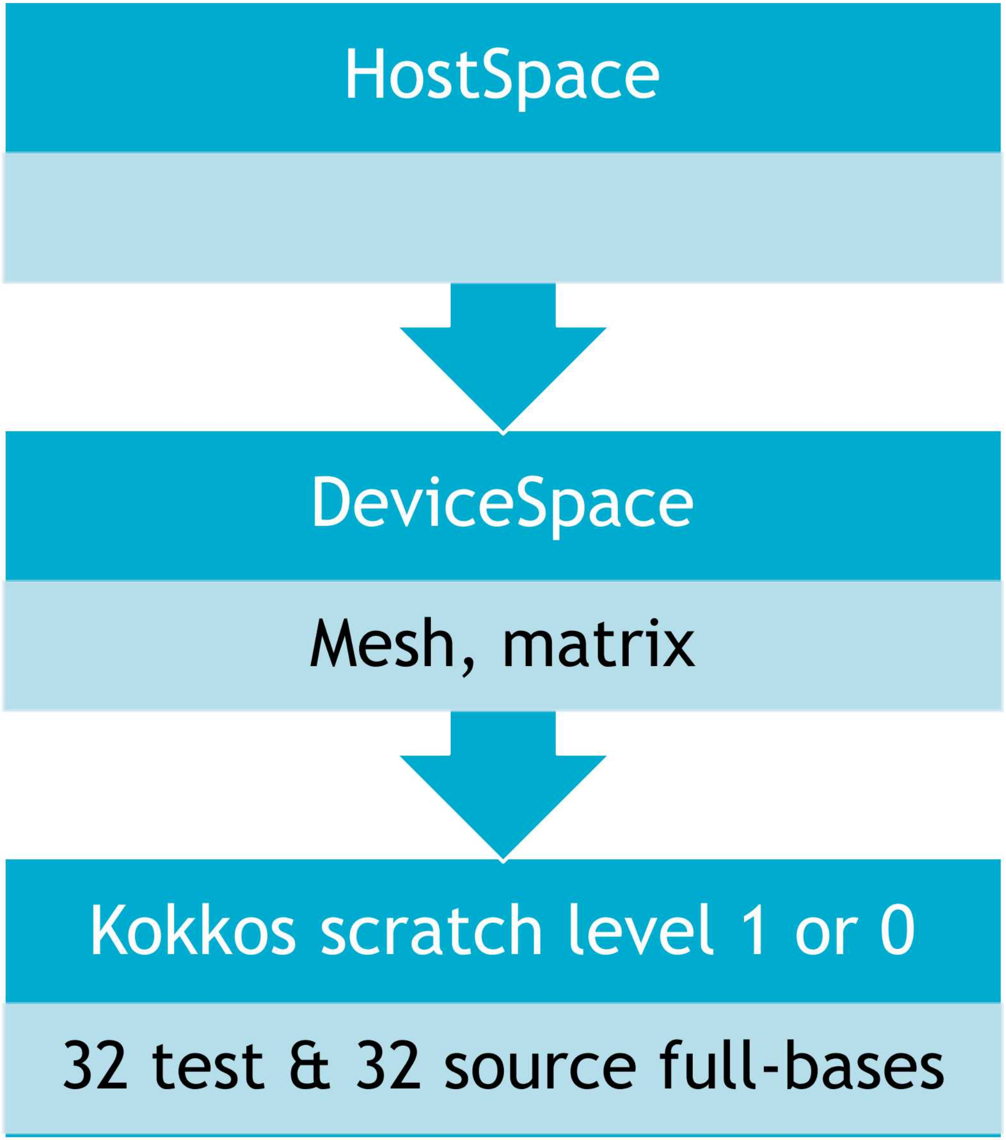
Kokkos scratch level 0

16 = 2 x 8 triangles

Considering Basis Pairs via an Outer Product of Bases

- Instead of loading Basis pairs, load 32 test and 32 source bases to make 1024 basis pairs on the fly.
  - Loading 1024 basis pairs requires 24576 vertices = 73728 doubles = 589824 bytes.
  - Loading 64 bases requires 768 vertices = 2304 doubles = 18432 byes.
- Fills a 32x32 block of the system matrix.
- GPU details: Load the information for 1 test basis and all 32 source bases for a given warp. This is broadcasting and coalesced memory access, respectively.

Thread	0	1	2	3	4	5	6	7	8	9	10
Test Unknown	6	7	8	9	10	11	12	13	14	15	16
Source Unknown	600	600	600	600	600	600	600	600	600	600	600



Considering Basis Pairs via Precomputing and Calling BLAS

- Precompute Λ & G.
- Use BLAS to perform a contraction over some of the dimensions of the following tensors:

$$\begin{bmatrix}\Lambda_1^{Ti} & \cdots & \Lambda_3^{Ti}\end{bmatrix}\begin{bmatrix}G_{1,1}^{Ti,Sj} & \cdots & G_{1,7}^{Ti,Sj} \\ \vdots & & \vdots \\ G_{3,1} & \cdots & G_{1,1}^{Ti,Sj}\end{bmatrix}\begin{bmatrix}\Lambda_1^{Sj} \\ \vdots \\ \Lambda_7^{Sj}\end{bmatrix}$$

- Λ_1^{Ti} is a matrix. 1st dimension: (x,y,z) coordinates. 2nd dimension: Full-basis $[Ti]$. Same for all other Λ. Full-bases $[Ti]$ and [Sj] .
- $G_{1,1}^{Ti,Sj}$ is a 3-dimensional tensor. 1st dimension: real and imaginary parts. 2nd and 3rd dimensions:
- Contract over the shown quadrature point dimension and the (x,y,z) coordinates of the Λs.
- For *N* test full-basis and *N* source full-basis, fills a continuous NxN block of the system matrix.
- The *G* tensor contains $41N^2$ doubles and the Λs contain 30 $30N$, where BLAS likes large *N*.
- Opting for instead precomputing quadrature points and basis requires $60N$ instead of $41N^2 + 30N$.