Accelerating HFSS Simulations Using HPC Resources

Mark Jones
Electronics Manager
Ozen Engineering, Inc.

Ansys / ELITE CHANNEL PARTNER

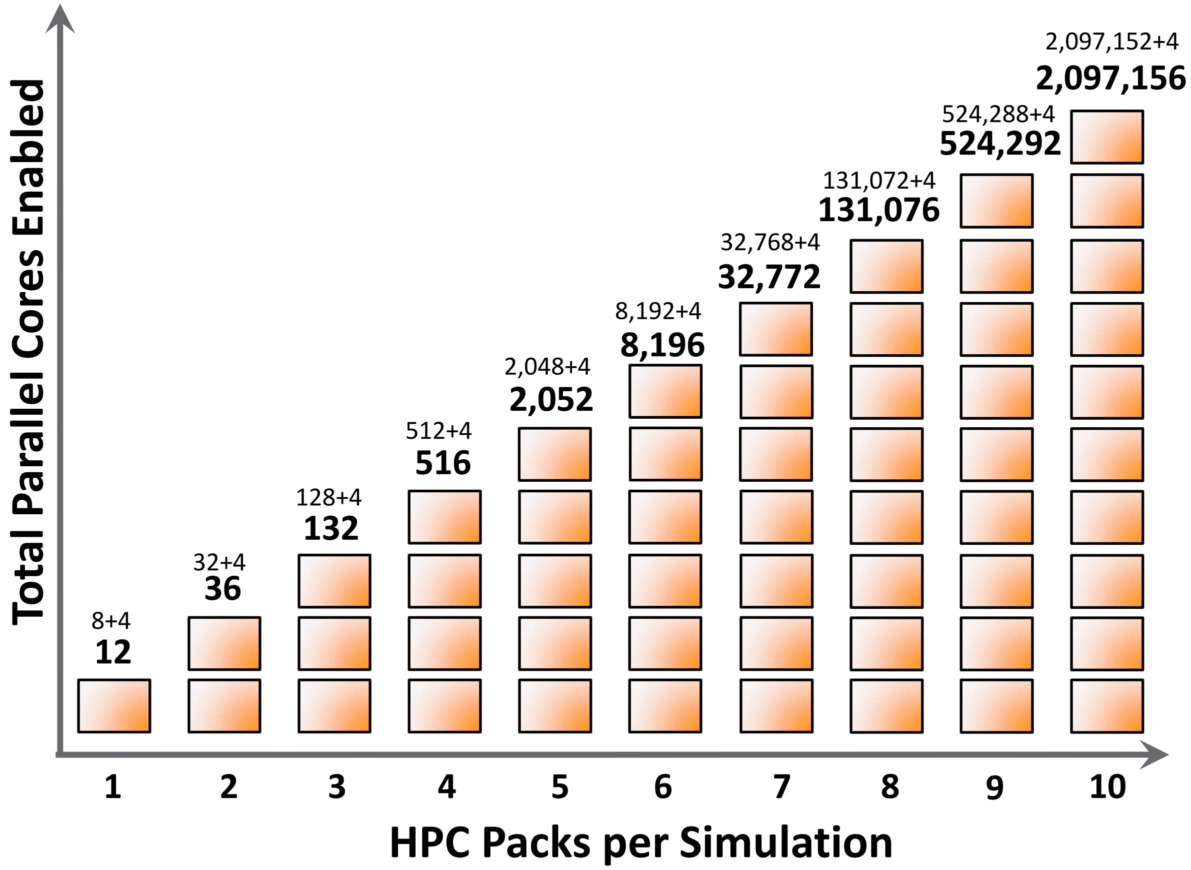
Outline

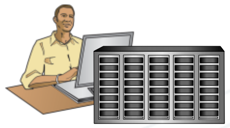
- Overview of HPC Pack licensing
  - Scaling behavior
  - Example usage
  - Parametric simulations

- HFSS HPC capabilities
  - Iterative and direct solvers
  - Shared and distributed memory matrix multi-processing
  - Domain decomposition solver
  - Distributed frequency sweeps
  - Distributed parametric sweeps and optimizations
  - HPC settings

HPC Pack Licenses

- Provide large-scale parallel capabilities for individual simulations
- Packs add cores to the 4 cores included with base solver license
  - Additional cores increase as geometric sequence ar^{n-1}
    - First term a=8
    - Common ratio r=4
  - Total cores for n Packs = $8*(4^{n-1})+4$
- HPC tasks from one Pack cannot be split across multiple simulations
  - Different from HPC Workgroup (Pool)

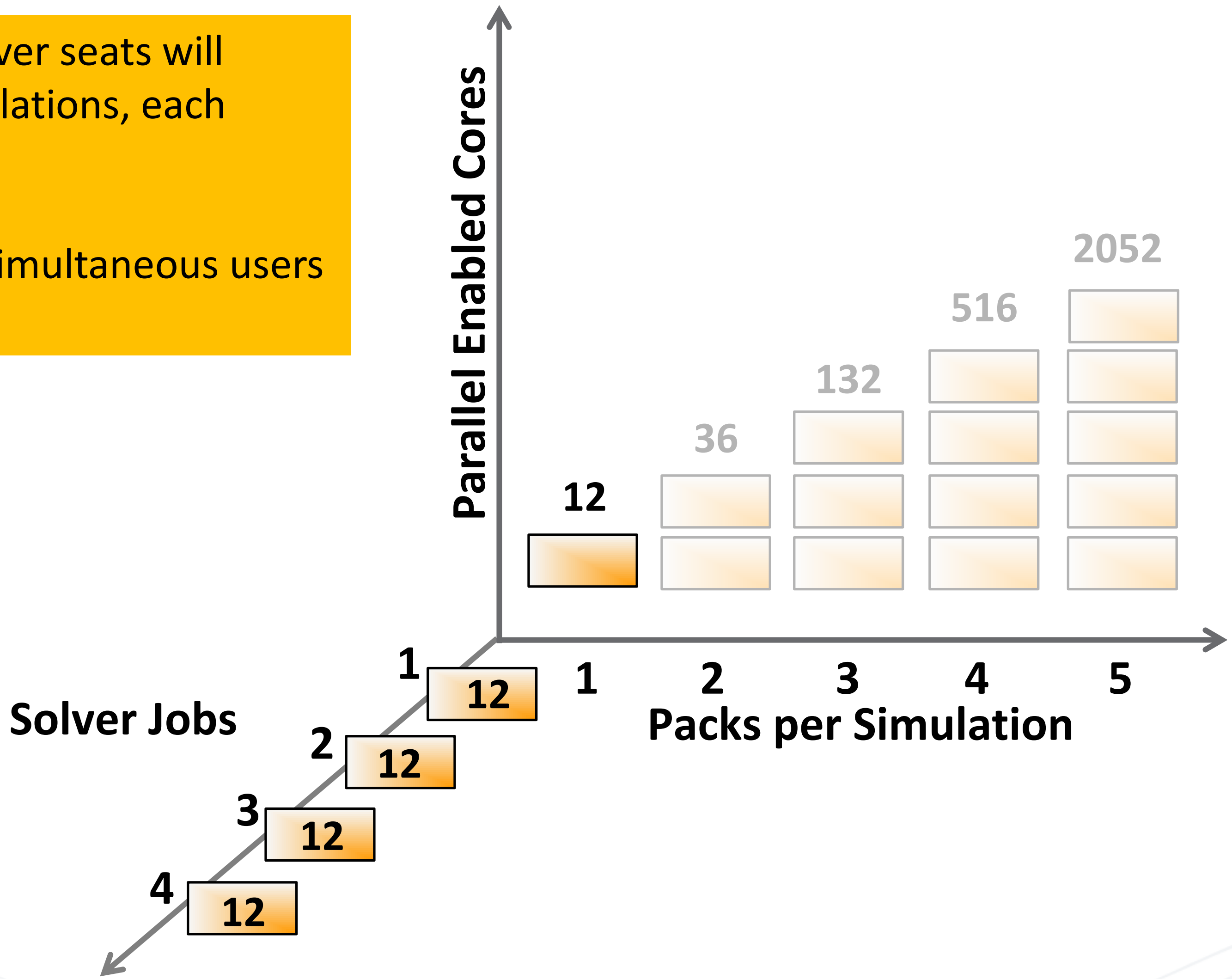




HPC Pack Usage: Example 1

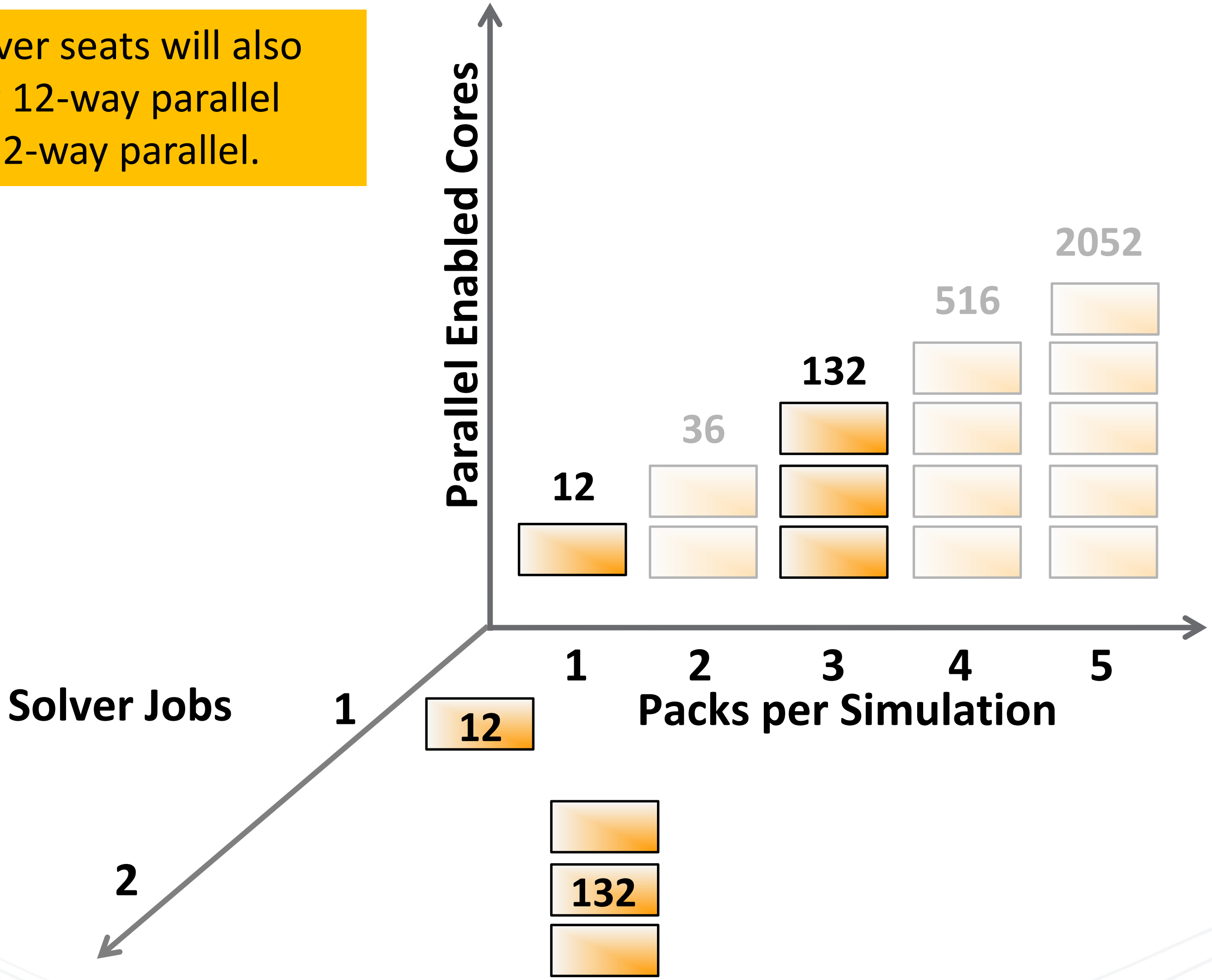
4 HPC Packs used with 4 solver seats will enable 4 simultaneous simulations, each running 12-way parallel.

This means there can be 4 simultaneous users with each using 1 HPC Pack.



HPC Pack Usage: Example 2

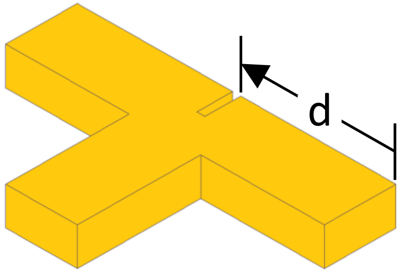
4 HPC Packs used with 2 solver seats will also enable 1 simulation running 12-way parallel and 1 simulation running 132-way parallel.



Parametric Simulations via HPC Packs

- Two-step calculation
    1. Standard calculation for use of total parallel cores
        - Subtract 4 cores per simulation included with base license
    2. Each parametric simulation after the first also uses a Pack
        - Each parallel simulation only requires HPC Pack license instead of HFSS solver license (lowers cost)

Example: Position of slot in waveguide tee. Solve 6 simultaneous parametric simulations with 24 cores / variation



Variation	d
1	2 cm
2	3 cm
3	4 cm
4	5 cm
5	6 cm
6	7 cm

HPC Pack License Calculation		
(1) Calculate Packs needed for total parallel cores	6 variations x (24-4) cores / variation	120 cores = 3 HPC Packs
(2) Calculate Packs needed for additional variations	(6-1) = 5 additional variations	5 HPC Packs
Add Packs for cores and variations		**8 HPC Packs**

HPC Parametric Licensing Calculator

Ansys HPC Licensing Calculator

Use this form to calculate the amount of HPC licensing required for your simulation using Ansys solvers like Fluent, HFSS, and Mechanical.

For example, six concurrent parametric analyses on eight cores each.

https://www.ansys.com/parametric

How many concurrent parametric analyses do you wish to run? 6

How many cores per analysis? 24

Which version are you using? 2021 R2 and later

Are you using Ansys optiSLang? No

CALCULATE ▸ CLEAR RESULTS ▸

160 HPC Workgroup increments OR 8 HPC Pack increments

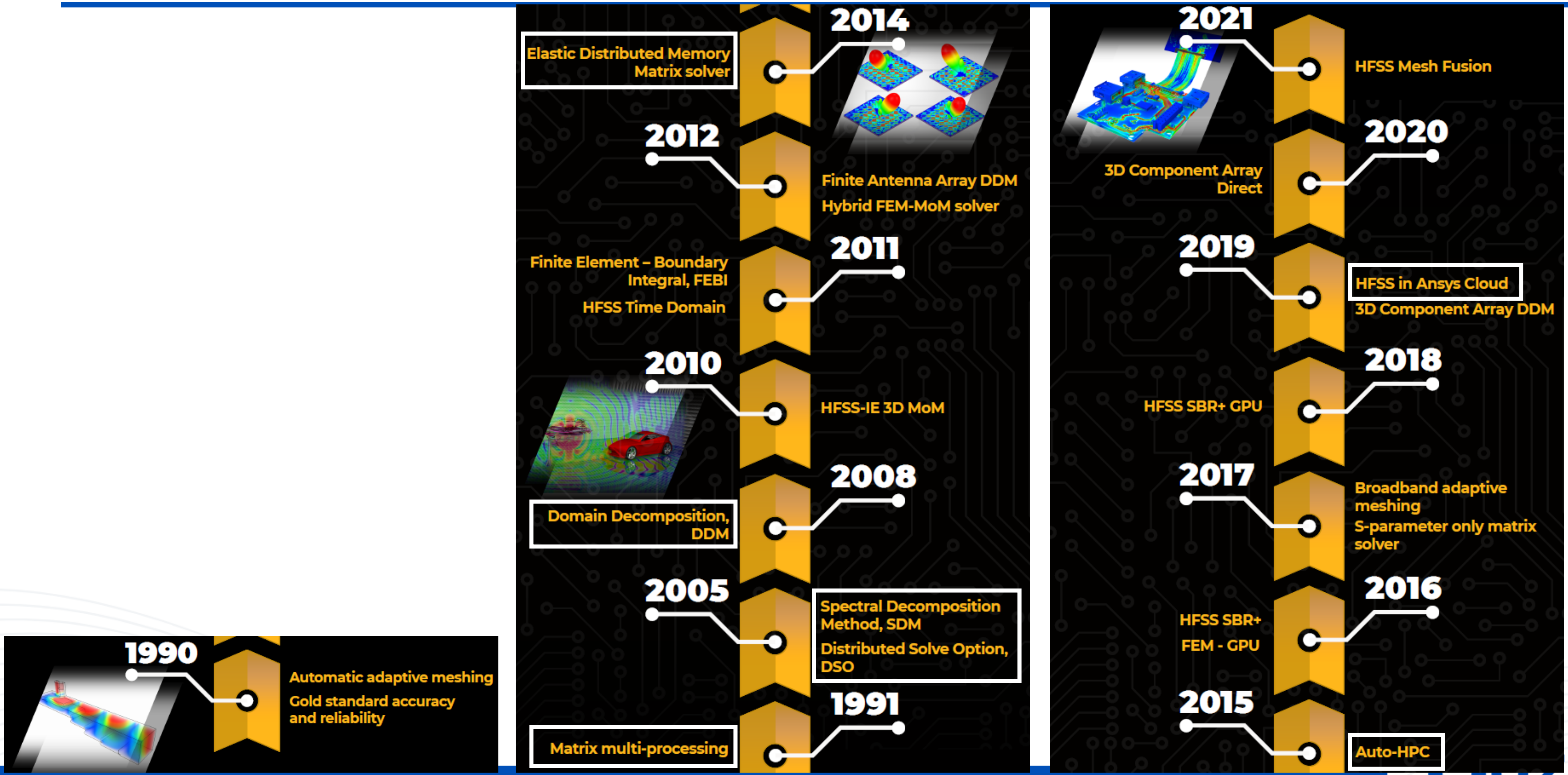
OZEN

Outline

- Overview of HPC Pack licensing
  - Scaling behavior
  - Example usage
  - Parametric simulations

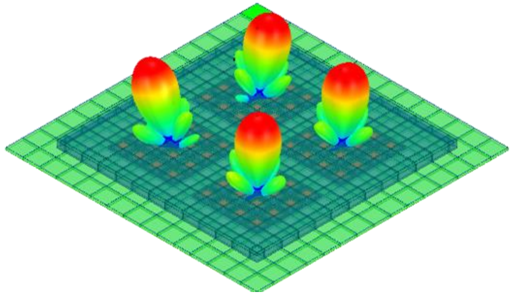
- HFSS HPC capabilities
  - Iterative and direct solvers
  - Shared and distributed memory matrix multi-processing
  - Domain decomposition solver
  - Distributed frequency sweeps
  - Distributed parametric sweeps and optimizations
  - HPC settings

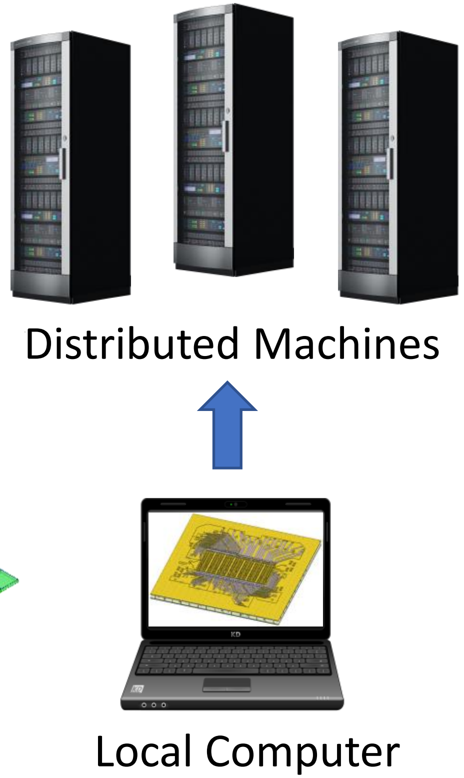
Over 30 Years of HFSS Innovation



HFSS and HPC

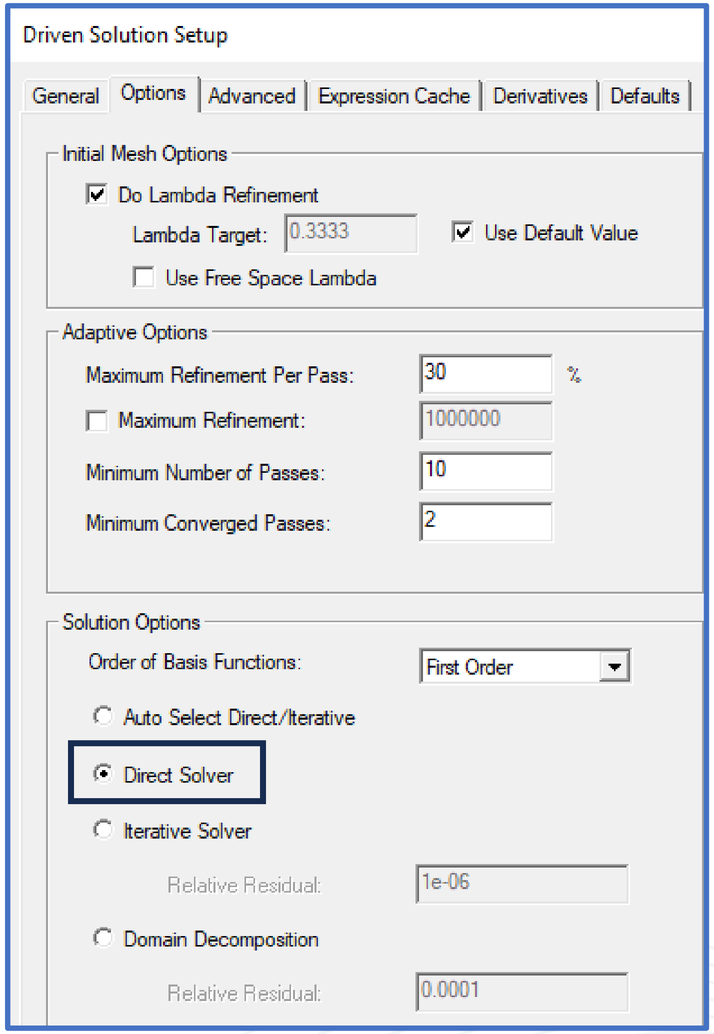
- HPC enables **faster** and **larger** simulations
  - Matrix multi-processing
  - Distributed memory matrix multi-processing
  - Distributed frequency sweeps
  - Distributed parametric sweeps and optimizations
  - Domain decomposition method
- FEM solver supports distributed matrix assembly, matrix solver, and field recovery
- Automated hierarchical HPC
  - Multi-tier HPC combining variations, frequencies, and solvers into scalable hierarchy





Direct Matrix Solver

- Mature technology developed by Ansys with decades of proven usage
  - Default solver setting
- Efficient multifrontal implementation to solve Ax = b
  - Find A = LU where L is lower triangular matrix and U is upper triangular matrix
  - Solve $Ax = b$ using forward-backward substitution
  - $LUx = b \rightarrow Ly = b \rightarrow y(1) = b(1)/L(1,1)$, $y(2) = [b(2)-L(2,1)y(1)]/L(2,2)$, etc.
  - $Ux = y \rightarrow x(N) = y(N)/U(N,N)$, $x(N-1) = [y(N-1)-U(N-1,N)x(N)]/U(N-1,N-1)$, etc.
- Direct sparse-matrix solver: solution guaranteed within a finite set of operations
  - Efficient for very large number of excitations (solves for all excitations b simultaneously)
  - Supports novel implementation of mixed precision to save memory

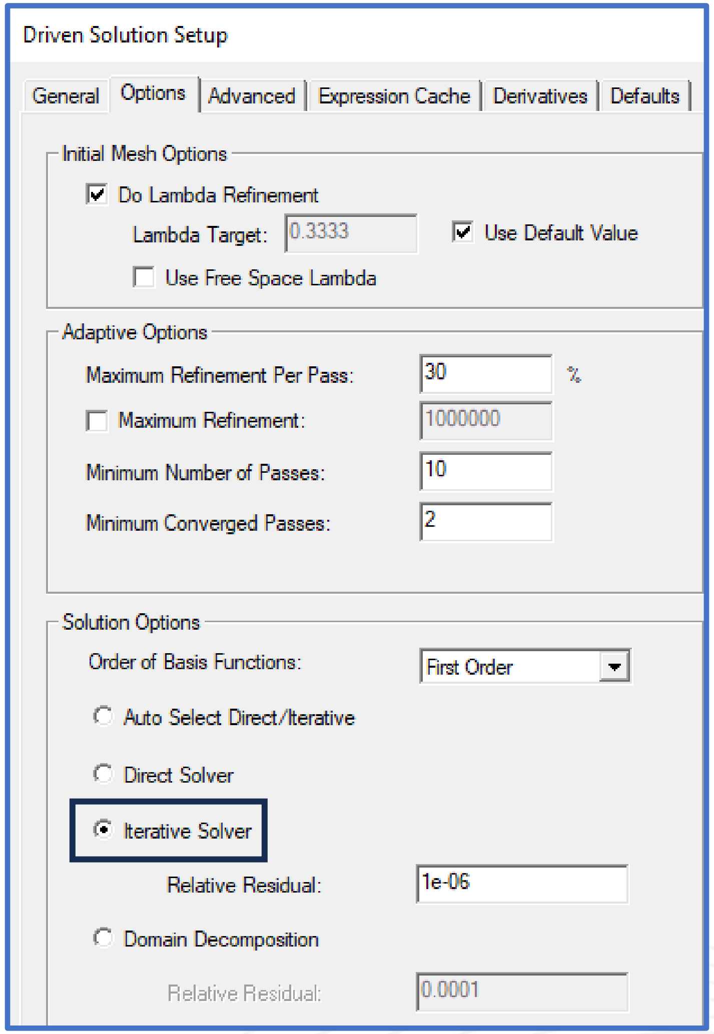


OZEN

Iterative Matrix Solver

- Introduced in 2007 as optional technique for driven solutions
  - Multi-level preconditioned conjugate gradient solver
- Great alternative for memory-limited situations
  - A*x* = *b* is solved iteratively by starting with initial guess $x = x_0$
  - $x_N = x_{N-1} + F\, r_{N-1}$ where $r_{N-1} = b - Ax_{n-1}$
  - Iterations stop when r_N is below desired error tolerance
  - Recommended for moderate number of excitations (typically fewer than 2*cores)
- Solver uses a multi-level preconditioner M to reduce the number of iterations
  - MAx = Mb
  - Multi-level preconditioning enabled via hierarchical vector basis functions

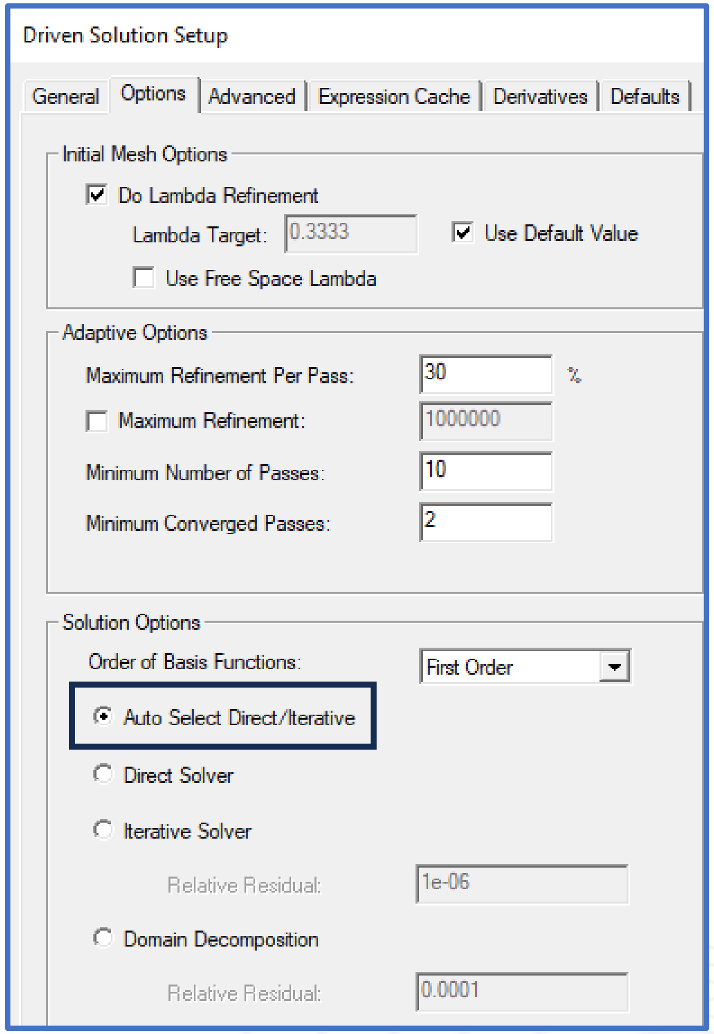
Solver	Time	RAM
Direct	$N^{1.7}$	$N^{1.3}$
Iterative	$N^{1.2}$	$N^{1.0}$



OZEN

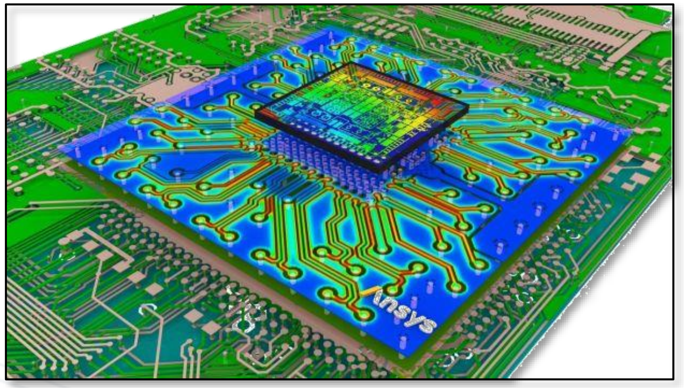
Auto Select of Direct / Iterative Solver

- Introduced in 2022 as optional setting for driven solutions
- Algorithm uses set of information about model complexity:
  - Number of mesh elements, matrix unknowns
  - Matrix bandwidth
  - Number of excitations
  - Solution frequency
  - Mesh quality
  - Computing resources (RAM, cores)
- For each adaptive pass, algorithm determines which solver to use



HFSS Supports AMD Math Libraries

- Fully supported in 2023 R1 release
- AMD math library for direct matrix solver
  - Automatic detection of CPU vendor and math library
  - AMD Optimizing CPU Libraries (AOCL) for numerical linear algebra (AOCL-BLAS, AOCL-LAPACK, etc.)



Complex IC Package (representative image)

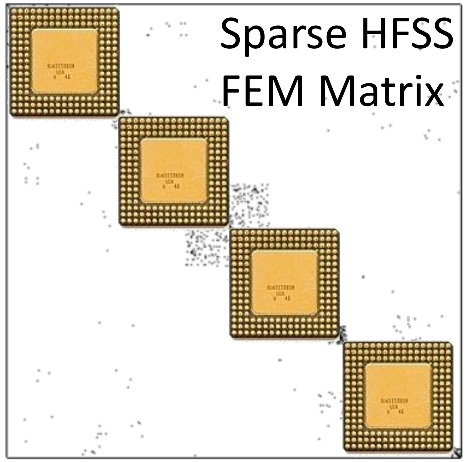
Number of Cores	Without AMD Math Library	With AMD Math Library
8	0:17:46	0:09:45 **(1.8X)**
16	0:11:04	0:06:59 **(1.6X)**
32	0:08:13	0:05:42 **(1.4X)**

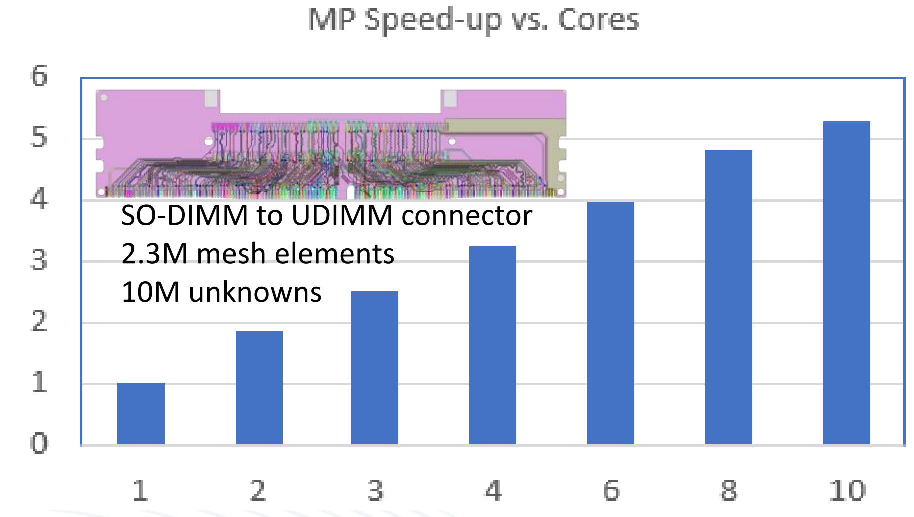
Up to 80% faster solutions

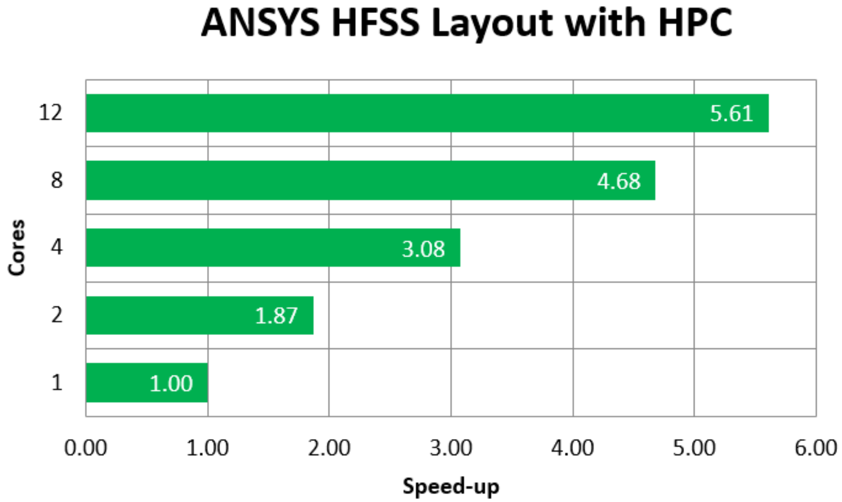
OZEN

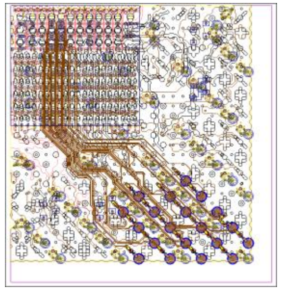
Matrix Multi-Processing

- Uses multiple CPU cores to solve dense frontal matrices along matrix
- Each core is assigned its own frontal matrix, allowing this portion of the solution process to speed up through parallelization



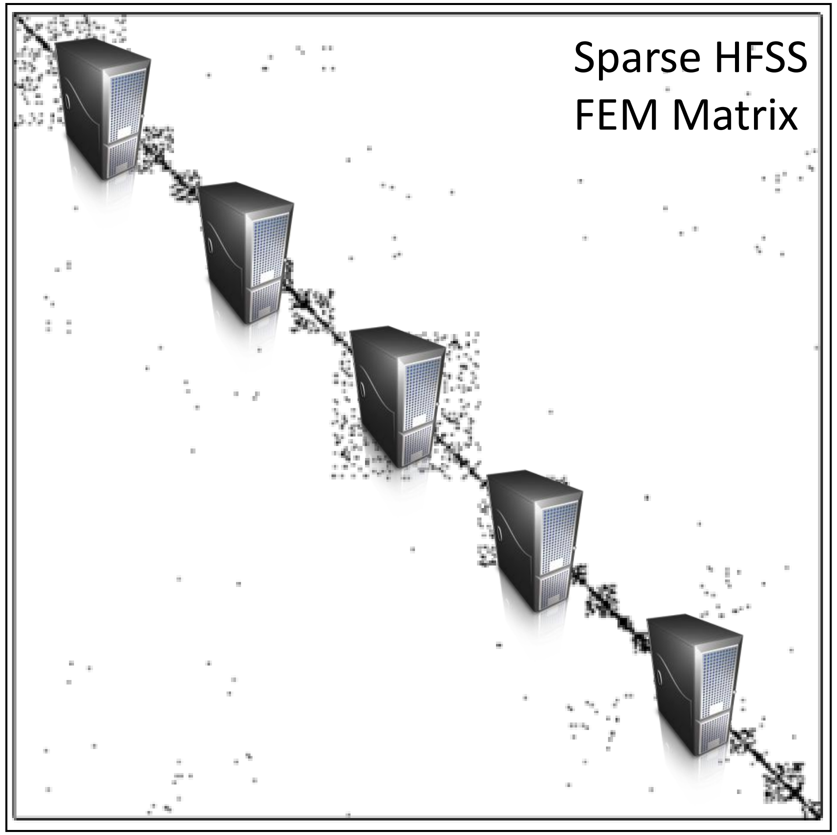






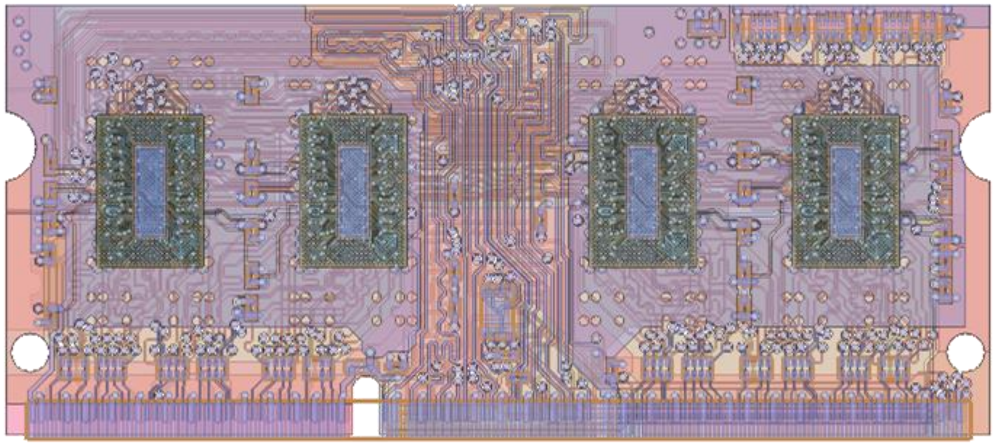
Distributed Memory Matrix Solver

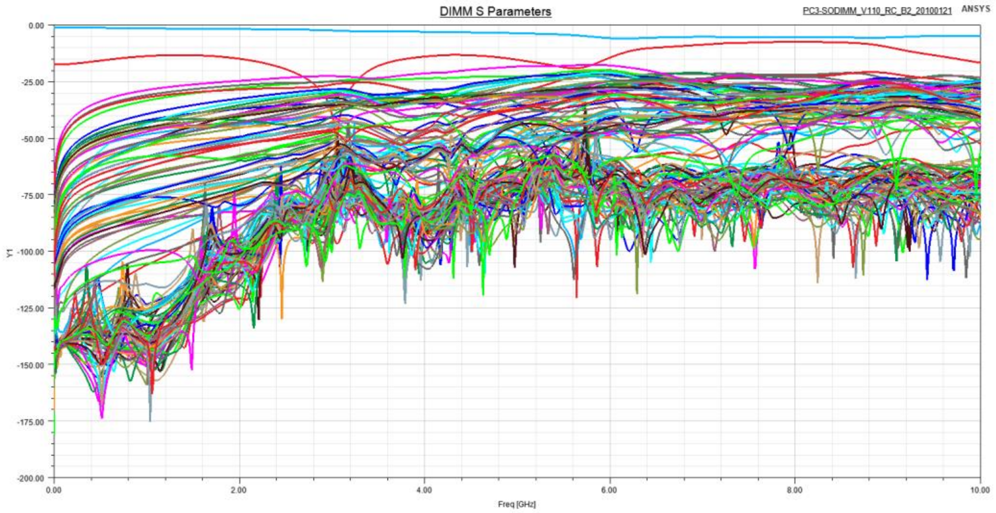
- Matrix multi-processing is not limited to single computer
- Introduced in 2014
  - Provides access to memory and cores across networked computers
  - Greatest accuracy, lowest noise floor, best efficiency for many ports
  - Solves fully-coupled electromagnetic matrix
- Significantly increases simulation capacity



Distributed Memory Matrix Example

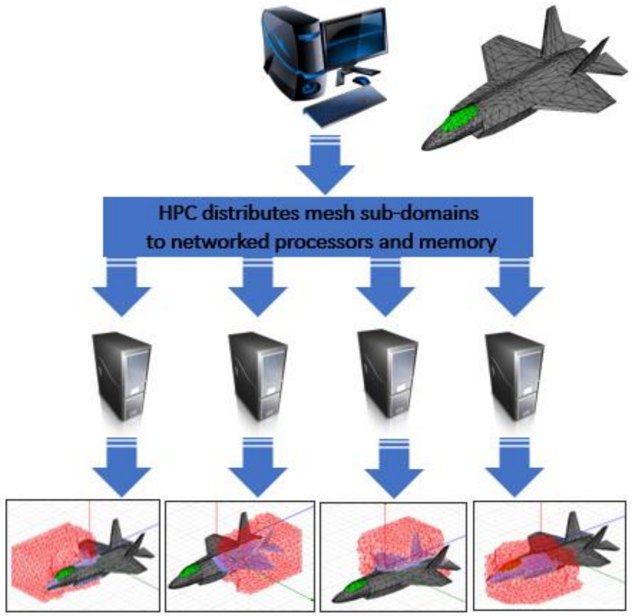
- Full 8-layer SO-DIMM module
  - 5.2M mesh elements
  - 21M matrix unknowns
  - 128 ports
  - Distributed frequency sweep
- Solved on Ansys Cloud
  - 128 total cores across 8 nodes
  - Maximum RAM / node: 32.75 GB
  - Total RAM: 262 GB
  - Total runtime: 7 hours

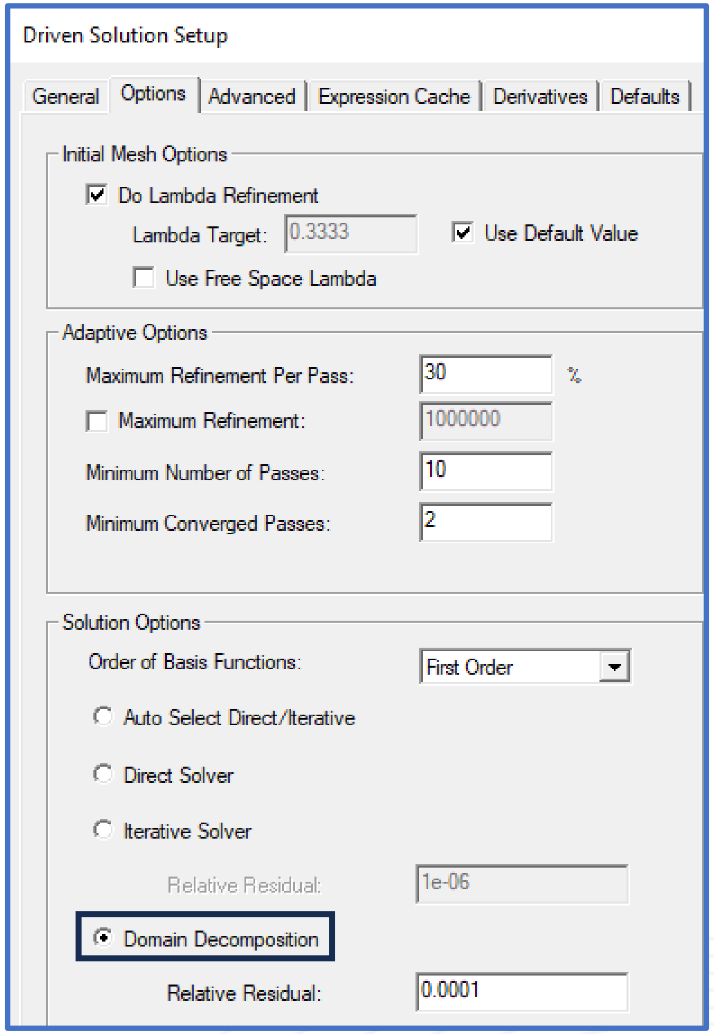




Domain Decomposition Method Solver

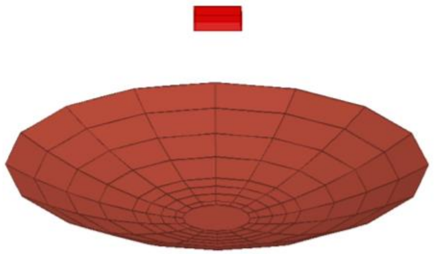
- Solver selection introduced in 2008
- Distributes mesh sub-domains to network of processors or nodes
  - Divide and conquer strategy
  - FEM volume automatically divided into multiple balanced domains
  - Full electromagnetic coupling across domains via Robin's transmission conditions
- Significantly increases simulation capacity



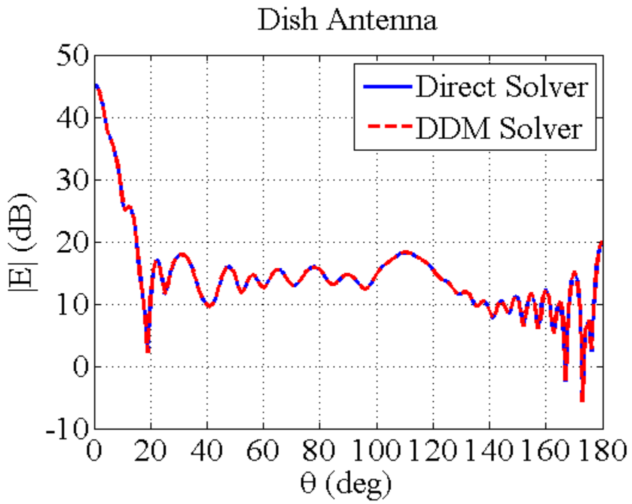


OZEN

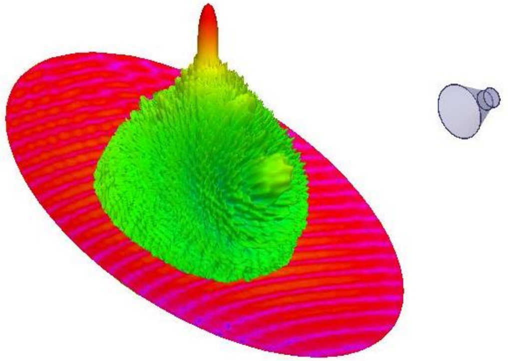
Domain Decomposition Method Examples



Reflector Antenna



Method	RAM	Runtime	Speed-up Factor
Direct Solver (8 cores)	46 GB	230 min	1.0
DDM Solver (8 domains)	**16 GB**	**20 min**	**11.5**



Reflector Antenna

Method	RAM	Runtime	Speed-up Factor
Direct Solver (8 cores)	36 GB	132 min	1.0
DDM Solver (7 domains)	**19 GB**	**20 min**	**6.6**

Summary of Distributed Memory Matrix and Domain Decomposition Solvers

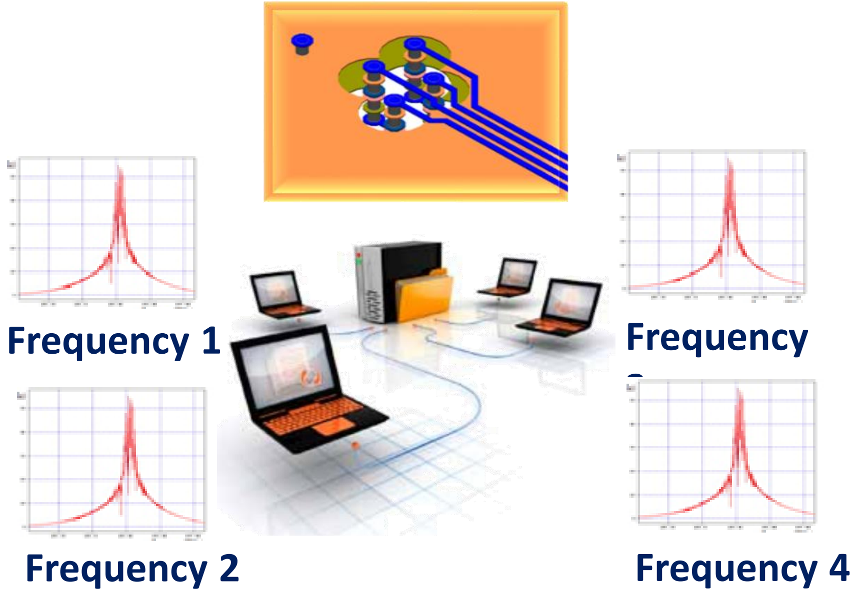
- Both methods can obtain full-fidelity solutions for large model that cannot be solved on single shared-memory machine
  - Enable use of distributed memory across multiple computers

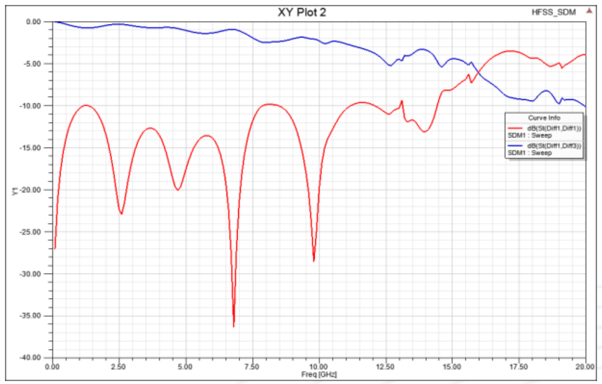
- Distributed memory matrix (DMM) solver
  - Direct matrix solver
  - Automatically invoked when needed to distribute matrix solution across multiple computers (based on RAM needs)
  - Recommended for IC, packages, and PCB simulations – many ports

- Domain decomposition (DDM) solver
  - Matrix solver selection explicitly enabled by user
  - Recommended for antenna simulations – fewer ports
  - Inefficient for models with slowly decaying spatial fields

Distributed Frequency Sweeps

- Introduced in 2005
  - Spectral Decomposition Method (SDM)
  - Discrete and interpolating frequency sweeps
- Distributes frequency points across network of processors or on single computer
  - Scalable to many cores





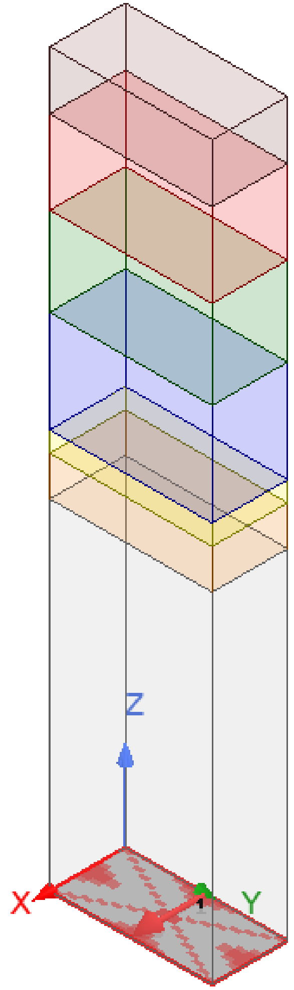
OZEN

Example Speed-up Factors

- Small model with Auto-HPC
  - ~5k tetrahedra
- Solved on Linux server
  - 64 cores
  - 256 GB RAM
- Model includes distributed interpolating frequency sweep
  - Frequency sweep dominates runtime
- Obtained linear speed-up vs number of cores
  - Solving more frequencies in parallel required fewer frequency groups for sweep to converge

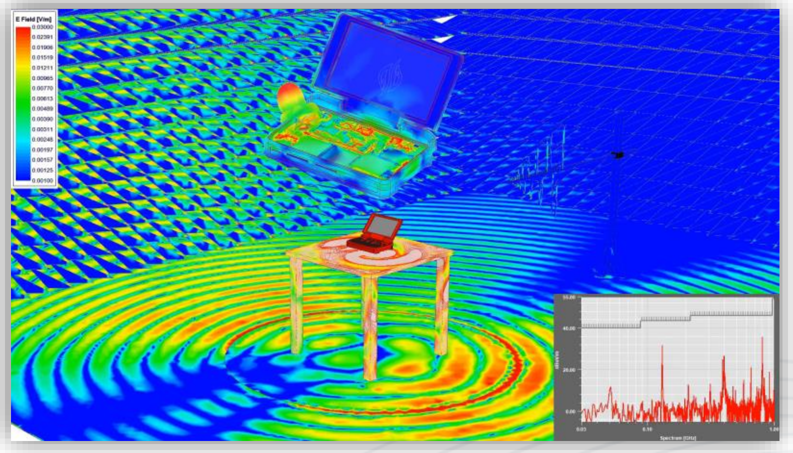
Total Cores	Frequency Groups Auto-Solved in Series	Speed-up Factor
4	4	**1.0**
12	3	**2.9**
16	2	**4.0**
20	1	**5.2**

Simulation results courtesy of Greg Le Sage at SLAC



Distributed Parametric Variations

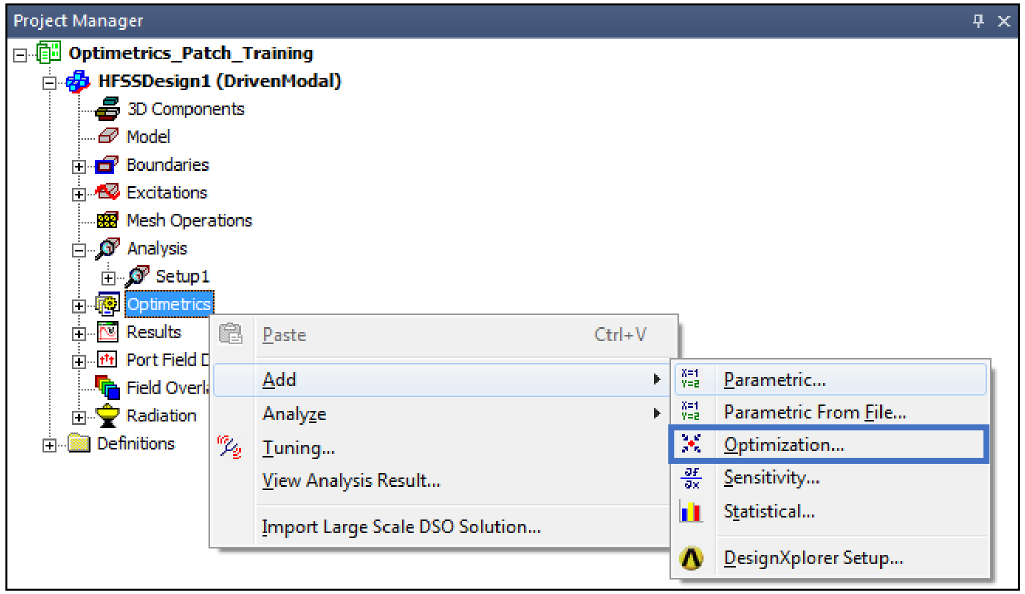
- Introduced in 2005
  - Originally called Distributed Solve Option (DSO)
- Distributes design variations across multiple cores/compute nodes
  - Each variation can be accelerated with matrix multi-processing and distributed frequency sweeps
- Scalable to many cores and nodes
  - Near-linear increase in throughput

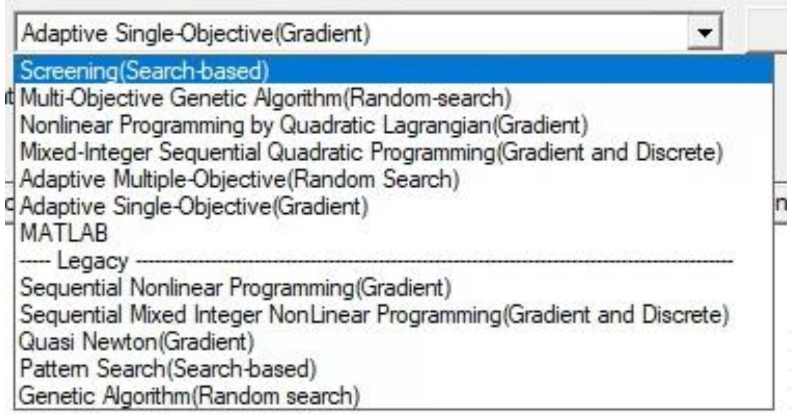


OZEN

Distributed Optimization

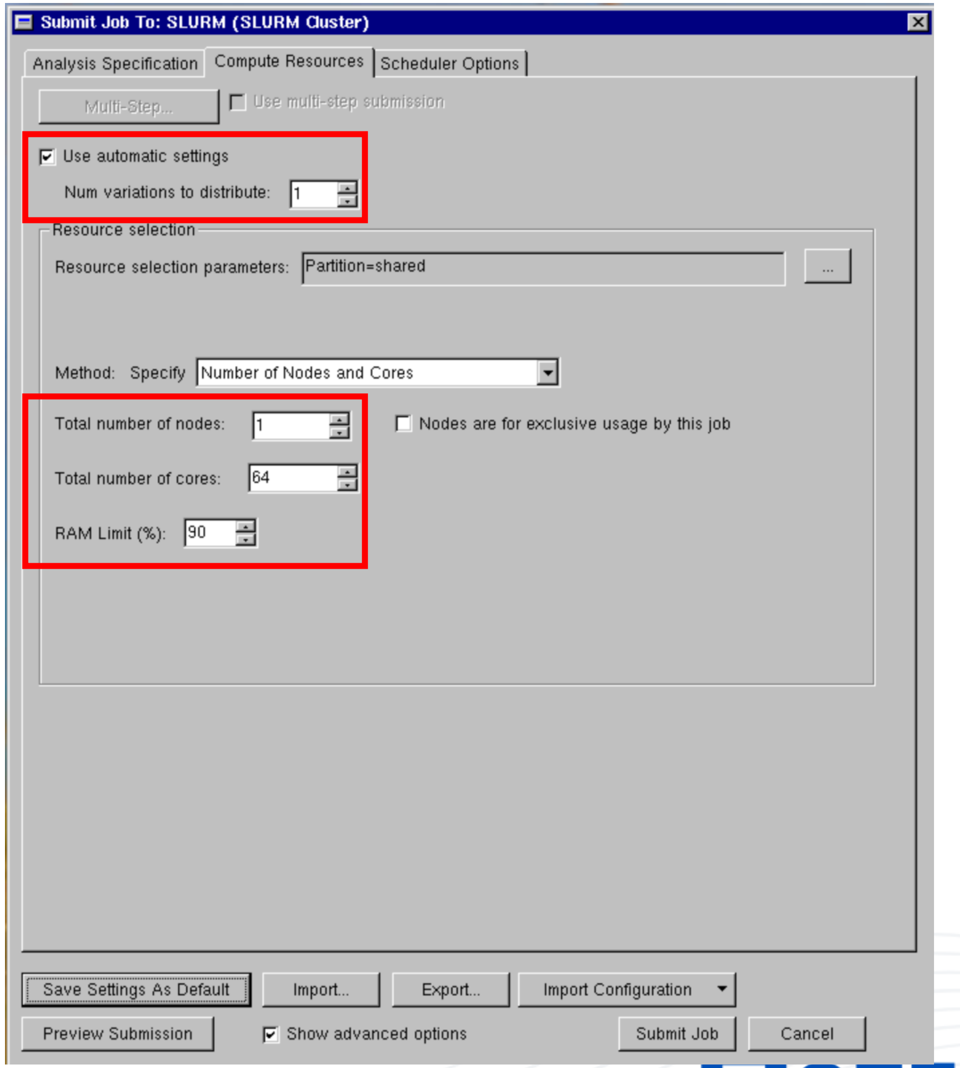
- All optimizers use matrix multi-processing and distributed frequency sweeps for each iteration
- Five algorithms support parallel iterations
    1. Screening (global search)
    2. Multi-objective genetic algorithm (global search)
    3. Legacy genetic algorithm (global search)
    4. Adaptive multiple objective (global search)
    5. Adaptive single objective (gradient search)





HPC Settings

- For manual HPC settings, “tasks” are single processes that may run in parallel
    - Frequency point
    - Parametric variation
    - Optimizer iteration
    - DDM subdomain
    - Iterative solver excitation
    - Transient solver excitation
- Auto-HPC option introduced in 2015
    - Eliminates need to specify tasks
    - Simply specify hardware resources (number of nodes and cores) and number of variations to distribute
    - Automated hierarchical HPC
    - Ansys recommends Auto-HPC setting



Summary

- HPC capabilities are continually pushing limits for high-fidelity simulations
  - On-premises and/or cloud computing clusters
  - Ansys HPC licenses

- HFSS simulations can be greatly accelerated by HPC technologies
  - Distributed parametric sweeps and optimizations
  - Distributed interpolating and discrete frequency sweeps
  - Shared and distributed memory matrix multi-processing
  - Domain decomposition solver and other DDM-enabled capabilities

- Ozen Engineering and Ansys are here to support you
  - support@ozeninc.com

Additional Resources

- Ansys webinar: “Overview of Ansys HFSS Solver Technologies,” Rickard Petersson.
  - https://www.youtube.com/watch?v=N7v4fgDyxB4
- Ansys webinar: “A Review of HPC Technologies in Ansys HFSS,” Matt Commens.
  - https://www.youtube.com/watch?v=DC-SA4hloHQ
- Ansys webinar: “Best Practices for Maximizing HFSS Performance During Package, PCB and System Simulations,” Dennis Soldo
  - https://www.youtube.com/watch?v=sQ-KEohvMEo
- Ansys webinar: “The Foundation of Domain Decomposition Technologies in HFSS,” Kezhong Zhao.
  - https://www.ansys.com/resource-center/webinar/the-foundation-of-domain-decomposition-technologies-in-ansys-hfss

THANK YOU!

Any questions?

Ansys / ELITE CHANNEL PARTNER