Maxwell: bringing cloud-powered electromagnetic simulations to Matlab

Advanced interface tutorial

- Advanced interface quick-start
- Examples: Maxwell in action
- How Maxwell uses the cloud (EC2)
- How Maxwell solves electromagnetics

Jesse Lu, Jelena Vuckovic group, Stanford University

Definitions

- What is Maxwell?
  - a Matlab toolset
  - that uses Amazon's Elastic Compute Cloud (EC2)
  - to solve 3D frequency-domain electromagnetic simulations.

- Features:
  - Cryptographically-secure communication (https)
  - Full control over all simulation parameters
  - GPU-acceleration provided by Nvidia Tesla GPUs
  - Queueing system to allow for full usage of cluster
  - Scalable to hundreds of simultaneous simulations running on hundreds of nodes.

Maxwell provides two user interfaces: Advanced and other

- advanced:

- other:

This presentation covers the advanced interface

Advanced interface quick-start

- Sign up at

```
% Download maxwell.m
>> urlwrite('m.lightlabs.co', 'maxwell.m');

% Provide AWS credentials and launch a 2-node cluster.
>> maxwell.aws_credentials('aws-access-id', 'aws-secret-key');
>> maxwell.launch('cluster-name', 2);

% Run simulation on 1 node.
>> [E, H] = maxwell.solve('cluster-name', 1, ...);

% Terminate cluster
>> maxwell.terminate('cluster-name');
```

Wait, what just happened?

- `urlwrite()` downloaded the advanced interface for Maxwell,
- `maxwell.aws_credentials()` provided the AWS credentials that
- `maxwell.launch()` needed to create a cluster on EC2.
- `maxwell.solve()` solved the electromagnetic simulation on the cluster and downloaded the resulting electromagnetic fields, and
- `maxwell.terminate()` terminated the EC2 cluster.

Examples: Maxwell in action

How Maxwell uses the cloud (EC2)

- Maxwell uses your Amazon Web Services (AWS) account to
  - Create a custom Amazon EC2 cluster, and
  - Solve electromagnetic simulations on it;

  all without leaving your local Matlab environment.

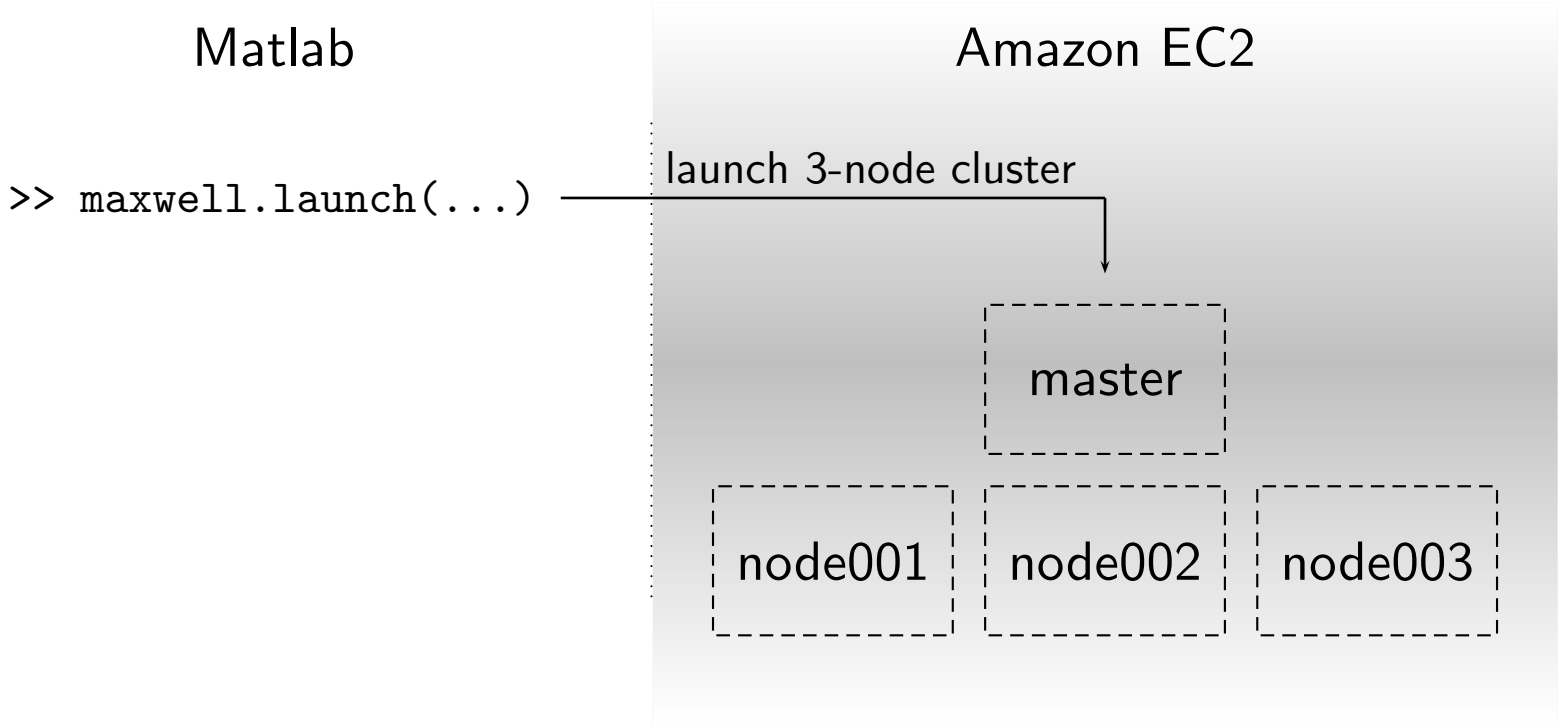
- To get started, you need to
  - sign up for an AWS account,
  - retrieve your AWS security credentials, and
  - purchase the custom Maxwell Amazon Machine Image (AMI).

  For detailed instructions see Website.

- Maxwell's advanced interface comprises of just five commands:
  - `maxwell.aws_credentials()`
  - `maxwell.launch()`
  - `maxwell.solve()`
  - `maxwell.solve_async()`
  - `maxwell.terminate()`

  for full documentation use "`doc maxwell.command`" in Matlab.

- `maxwell.aws_credentials('aws-key-id', 'aws-secret-key');`
  - Stores the security credentials linked to your AWS account locally
  - Security credentials are used to launch and terminate clusters
  - Transmitted over https and never stored on server-side
  - Tutorial on obtaining your credentials at Website.



- `maxwell.launch('cluster-name', num_nodes);`
  - Creates an EC2 cluster consisting of 1 master node and `num_nodes` worker nodes

  - `'cluster-name'` parameter allows for using multiple clusters at once.
  - The launch can be monitored manually from the EC2 Management Console at `console.aws.amazon.com/ec2`

- The master node is launched
  - with the paid Maxwell Amazon Machine Image (AMI),
  - as an on-demand instance, and
  - as an `m1.medium` instance.

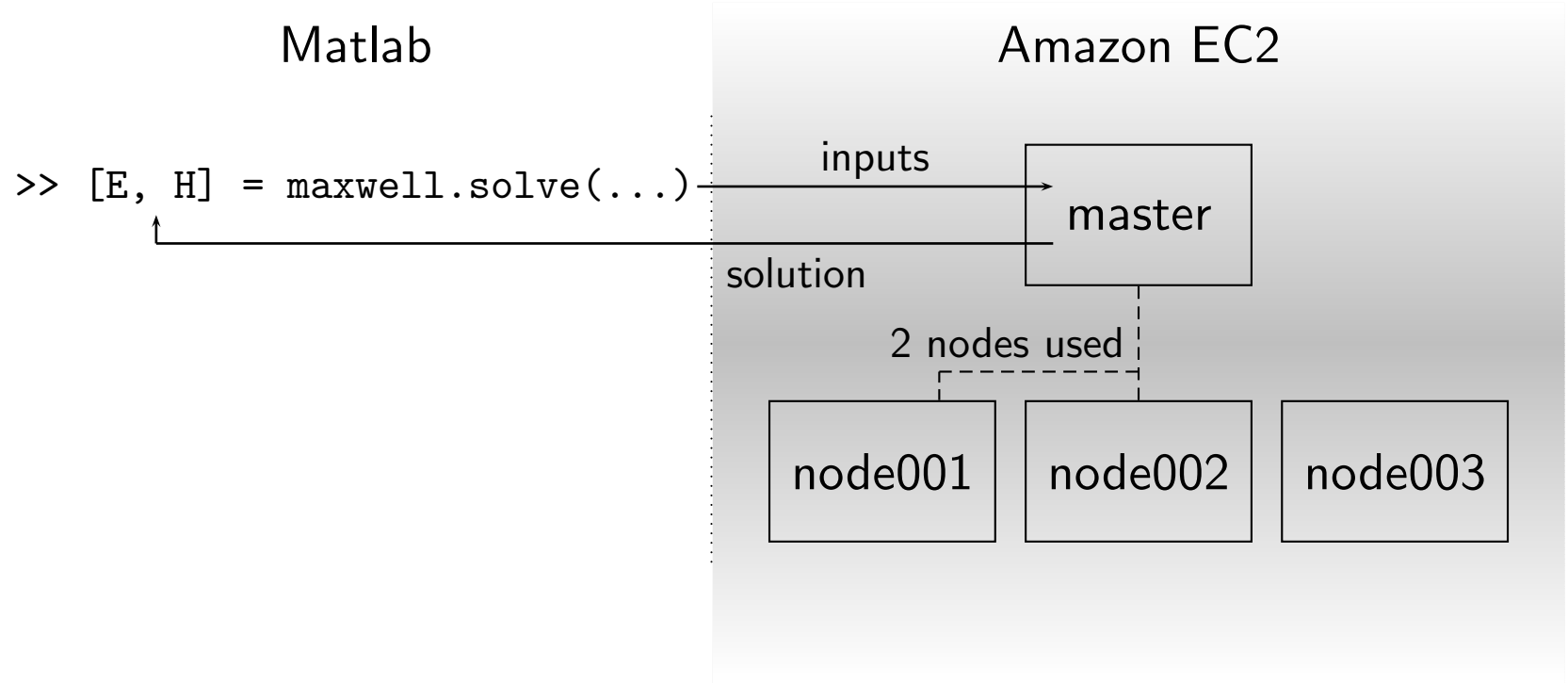
- The worker nodes are launched
  - using a public (free) AMI,
  - as spot request instances (in order to achieve up to 80% savings), and
  - as `cg1.4xlarge` instances.

Note that the use of spot requests for worker nodes may result in sudden cluster termination, in this case the cluster will need to be terminated and a new cluster should be started.

- Naturally, a cluster can only run simulations on the nodes that it contains (i.e. you can't run a 5-node simulation on a 4-node cluster)

- However, each cluster contains a built-in queueing system that allows for multiple jobs to be handled simultaneously (i.e. 10 2-node simulations on a 4-node cluster is okay)

- Lastly, Amazon, by default, caps the number of worker nodes allows at 10; to request more, go to `aws.amazon.com/contact-us/ec2-request/` and request the limit of `cg1.4xlarge` spot requests to be increased for your account.

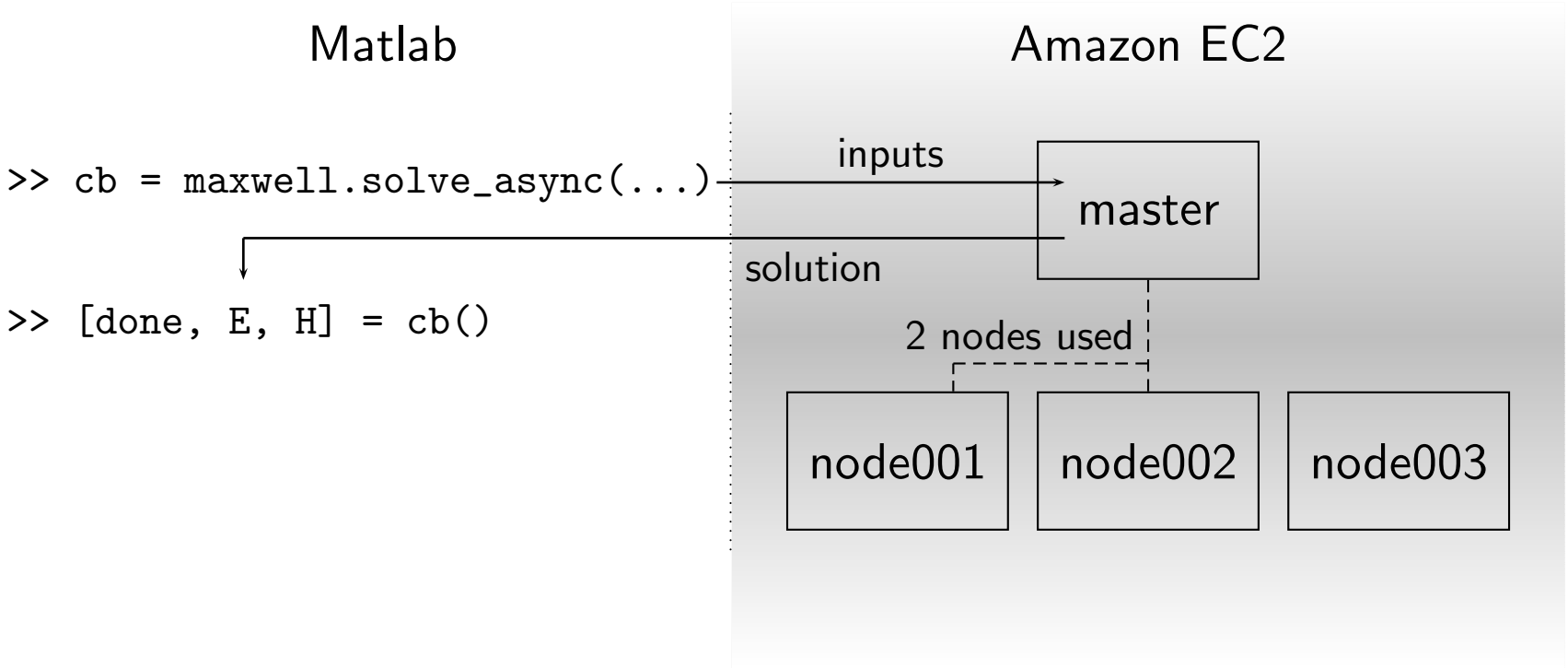


- `[E, H] = maxwell.solve('cluster-name', n, ...);`
  - Solves an electromagnetic simulation on `n` nodes of cluster `'cluster-name'`

  - Additional simulation parameters "..." described in following section
  - Returns as solution both electric and magnetic fields
  - For full documentation of this function see Website

- `maxwell.solve()` proceeds as follows:
  - Transfers simulation parameters to the specified cluster
  - Waits for worker nodes to be provisioned for the simulation
  - Continues to wait as simulation is executed on worker nodes
  - Retrieves simulation results back to Matlab

- `maxwell.solve()` features
  - complete integration within Matlab
  - no need to deal with simulations files
  - real-time plot of simulation progress



- `callback = maxwell.solve_async('cluster-name', n, ...);`
  - Asynchronous solve that returns a callback function instead of waiting for the simulation to complete

- The callback function is then used to check for solve completion and to retrieve the simulation results:

```
[is_finished, E, H] = callback();
```

- If the solve has not finished, `is_finished` is set to `false` and `E` and `H` both return empty cell arrays
- If the solve has indeed finished, `is_finished` is set to `true` and `E` and `H` contain the solution fields to the problem
- The additional simulation parameters "`...`" are identical to those used in `maxwell.solve()` and are detailed in the following section

- The purpose of `maxwell.solve_async()` is to allow even single-threaded Matlab users to simultaneously execute a virtually unlimited number of simulations.

```
% Start n simulations.
cb{1} = maxwell.solve_async(...);
...
cb{n} = maxwell.solve_async(...);

% Wait for the simulations to complete.
[is_finished, E, H] = cb{1}();
...
CHECK THIS!!
```

- `maxwell.terminate('cluster-name');`
  - Terminates the cluster `'cluster-name'`
  - Note that AWS instances are charged by the hour and that partial hours are charged the full hour.

- Terminations (and launches) can be monitored manually from the EC2 management console at `console.aws.amazon.com/ec2`

- Terminations can also be executed manually from the console, although it is recommended to perform the full termination from Matlab in order to tie up all loose ends.

How Maxwell solves electromagnetics

- In this section we detail the simulation parameters used by the `maxwell.solve()` and `maxwell.solve_async()` functions

- Along with the `'cluster-name'` and `n` parameters, both functions support the following parameters which describe the physical simulation:

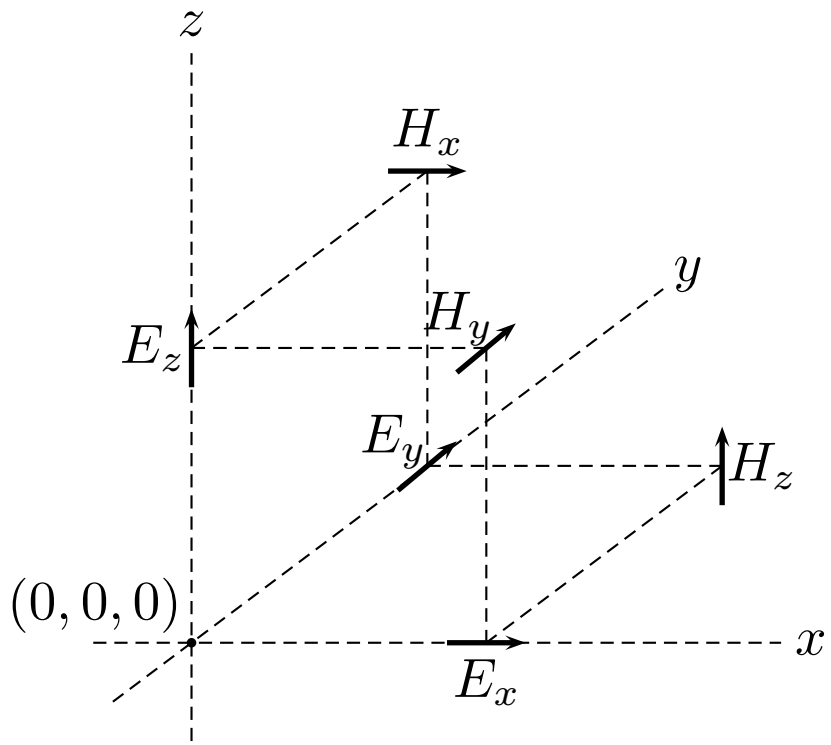
  - `omega`
  - `d_prim, d_dual, s_prim, s_dual`
  - `mu, epsilon`
  - `E, J`
  - `max_iters, err_thresh`

  and can be understood from the master equation:

$$\nabla \times \mu^{-1} \nabla \times E - \omega^2 \epsilon E = -i\omega J \tag{1}$$

- Frequency parameter: `omega`
  - The angular frequency of the simulation
  - Since Maxwell employs a frequency-domain solver, the exact frequency for the simulation can be set with this parameter
  - is equal to $2\pi f$ where f is in units of Hz

The Yee cell



- Spatial grid parameters: `d_prim, d_dual, s_prim, s_dual`
  - Controls the spatial grid of the simulation
  - `d_prim` and `d_dual` are typically used to set the spatial grid within the simulation
  - `s_prim` and `s_dual` are typically used to set the spatial grid within the absorbing layers at the edges of the simulation.
  - In the end, both `d` and `s` parameters are essentially interchangeable
  - `d_prim` and `s_prim` refer to the distances between E_w in direction w where $w = x, y, z$ (e.g. distances between E_x in the x direction)
  - `d_dual` and `s_dual` refer to the distances in directions other than w of adjacent E_w (e.g. distances between E_y in the x direction)
  - Lastly, Maxwell uses a periodic wrap-around grid, so that the first `_prim` and last `_dual` entries correspond to the the wrap-around distances from the last to the first Yee cells

- Material parameters: `mu, epsilon`
  - defined at H- and E- field points on the Yee grid respectively
- Field parameters: `E, J`
  - `E` allows you to set the initial value of the E-field for the solver.
  - `E = 0` works for most cases, but a random initial field is needed in other cases.
  - `J` describes the current source in the master equation and is situated at the E-field point on the Yee grid.
- Convergence parameters: `max_iters, err_thresh`
  - `max_iters` determines the maximum number of iterations before the solver terminates
  - `err_thresh` determines the error threshold below with the solver termiantes, typically set to `1e-6`.