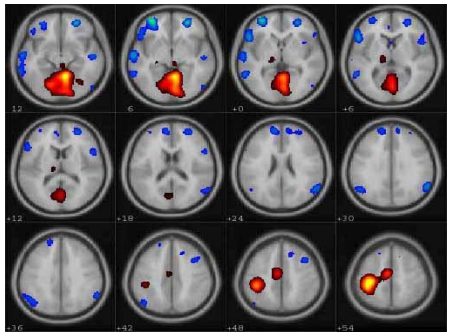
LLVM for Interactive Modeling and High Performance Simulation

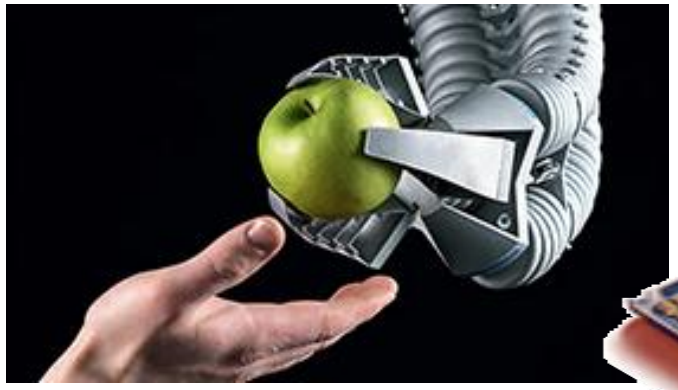
Peng Cheng Nathan Brewton Dale Martin

Model Based Design

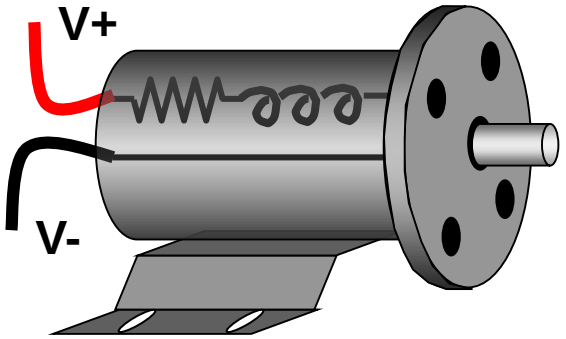
MATLAB® SIMULINK®

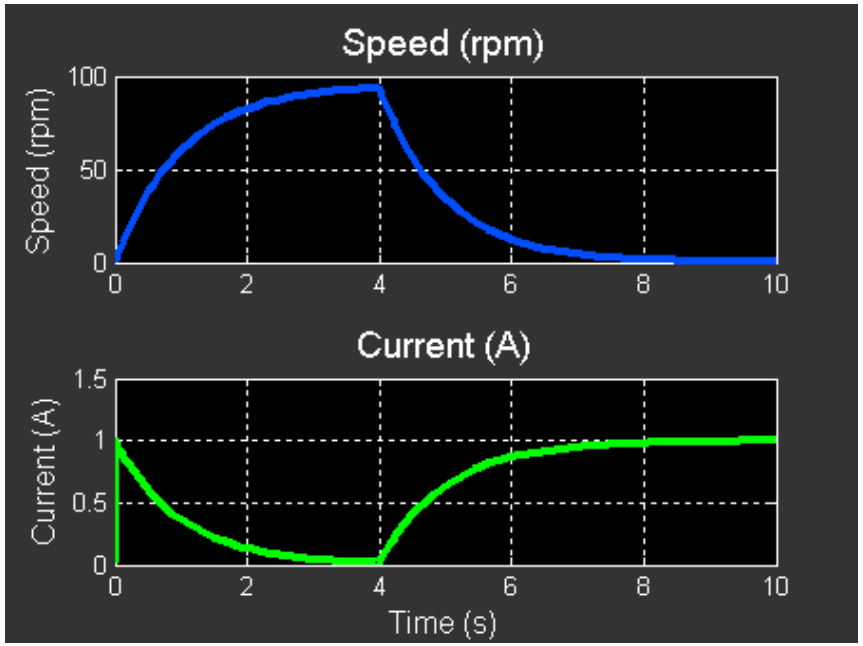
- Aerospace and defense
- Automotive
- Biotech and pharmaceutical
- Electronics and semiconductors
- Industrial automation and machinery
- Medical devices
- …

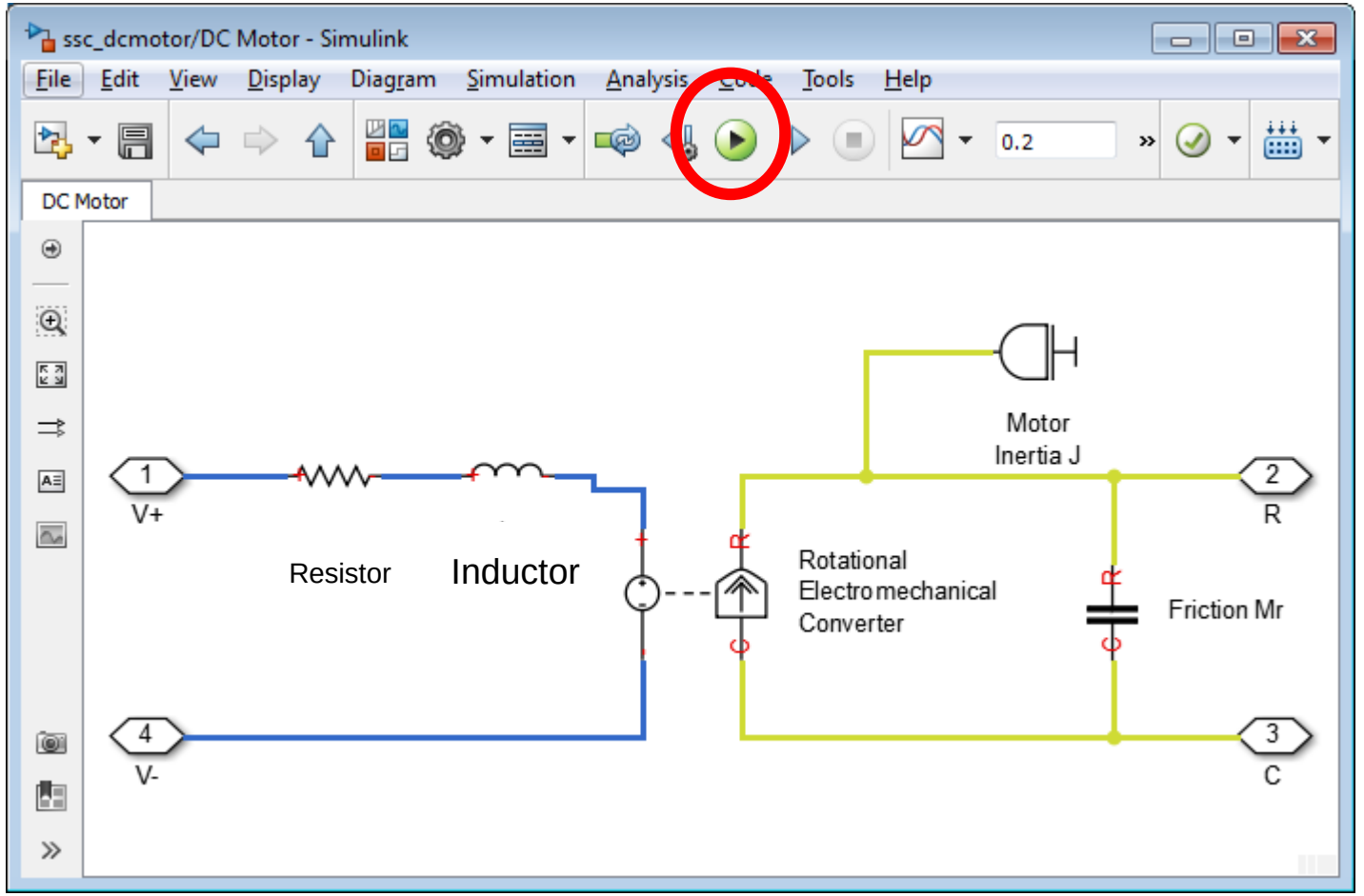




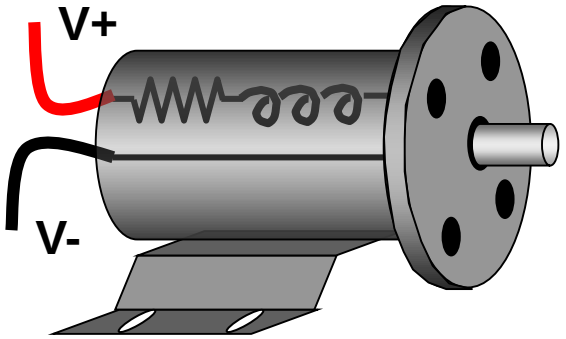
An Example of Interactive Modeling

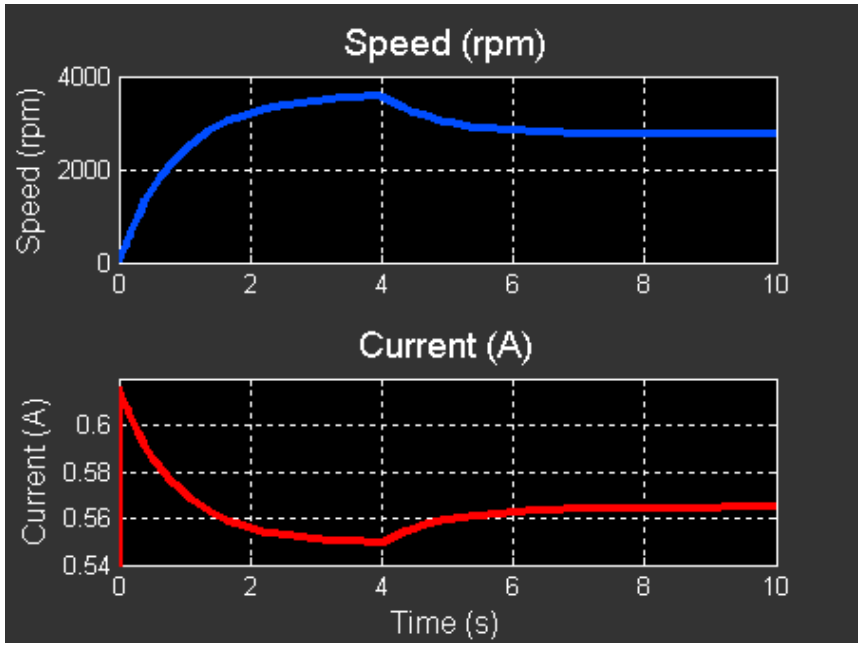


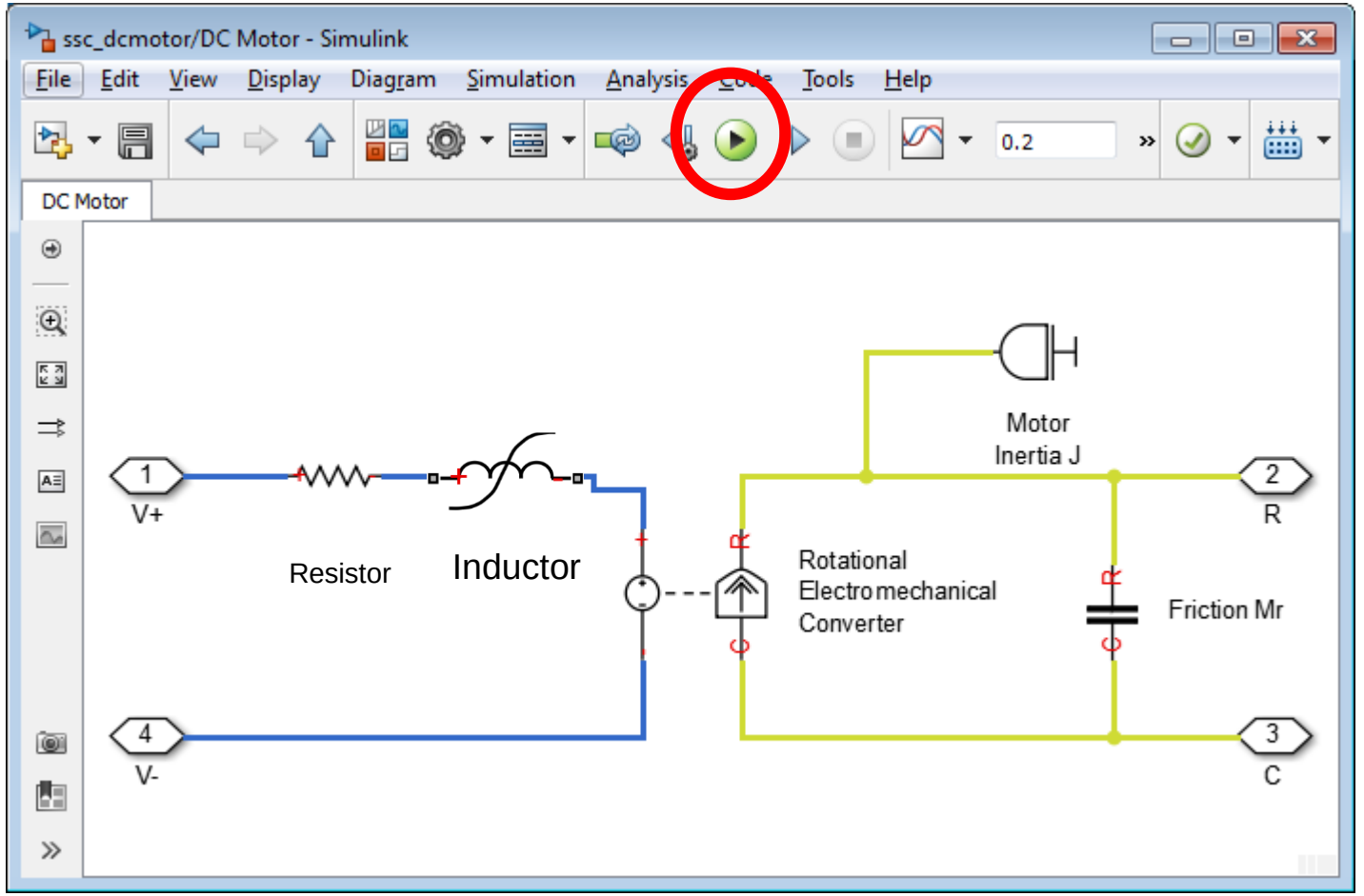




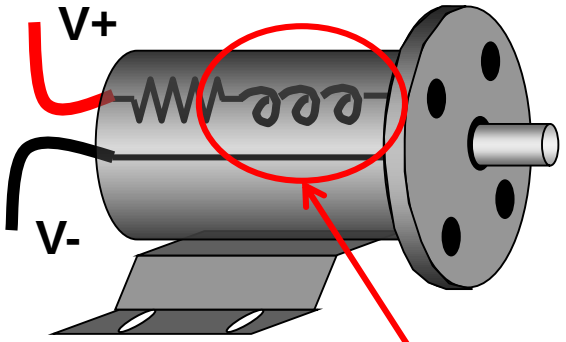
An Example of Interactive Modeling





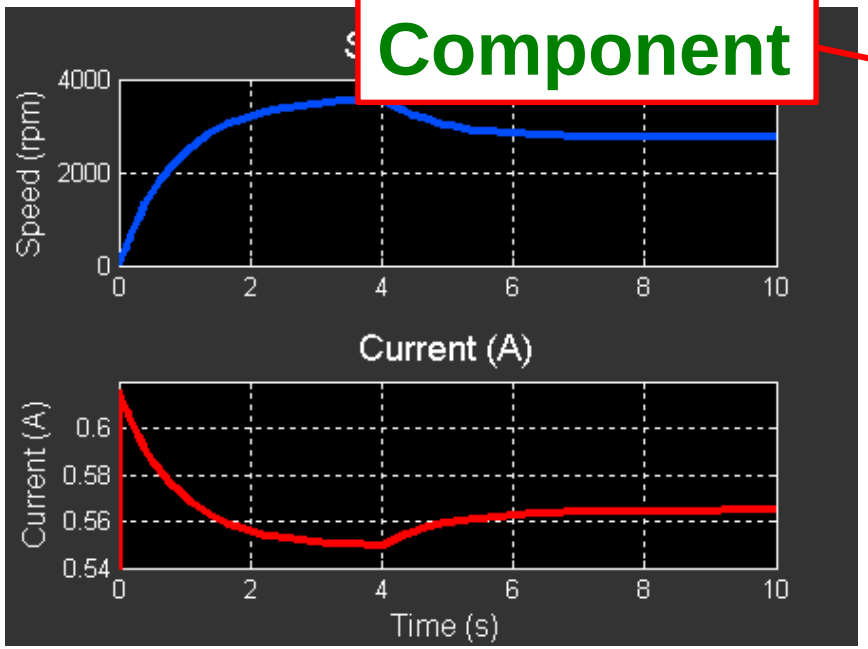


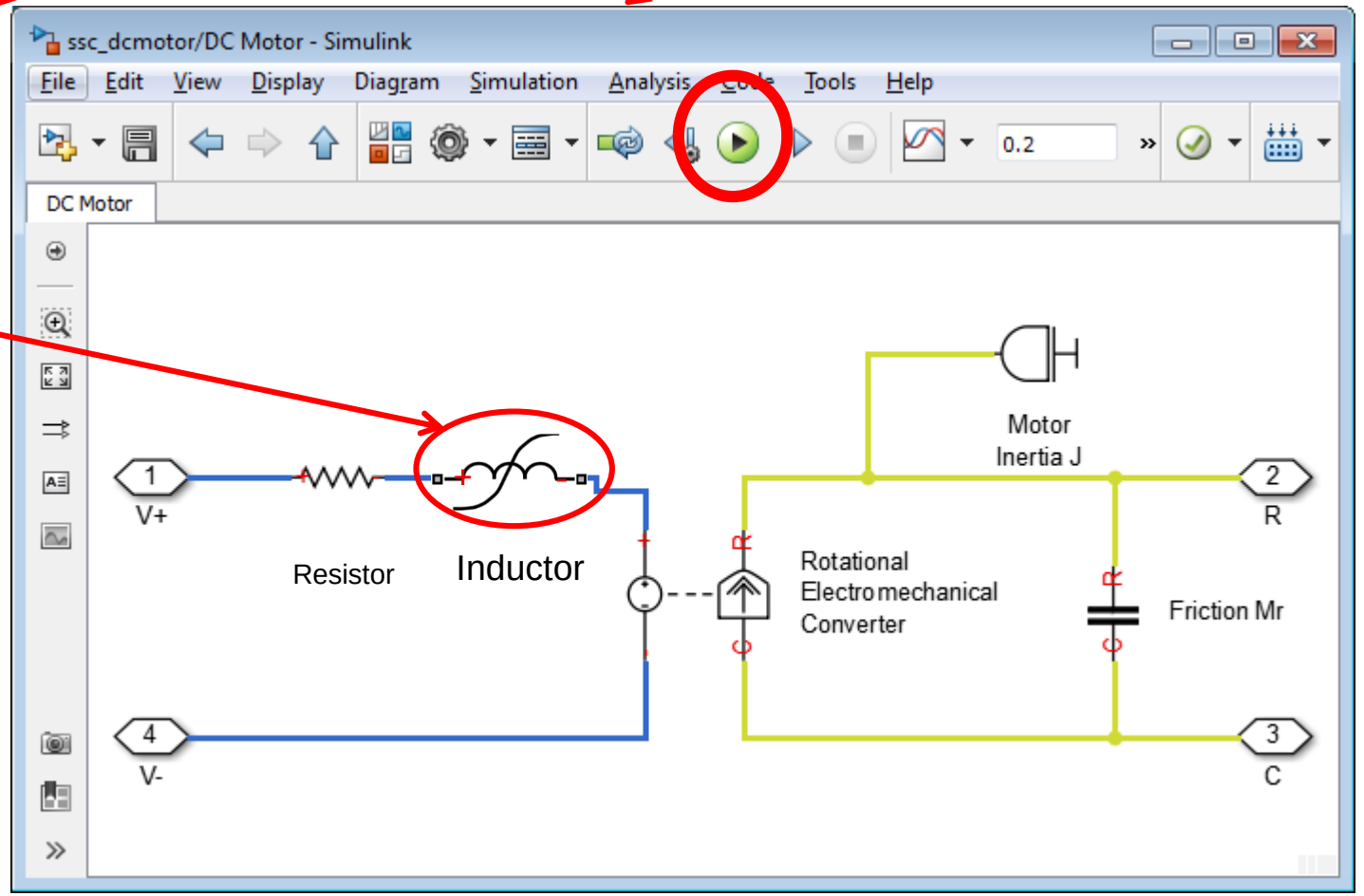
An Example of Interactive Modeling



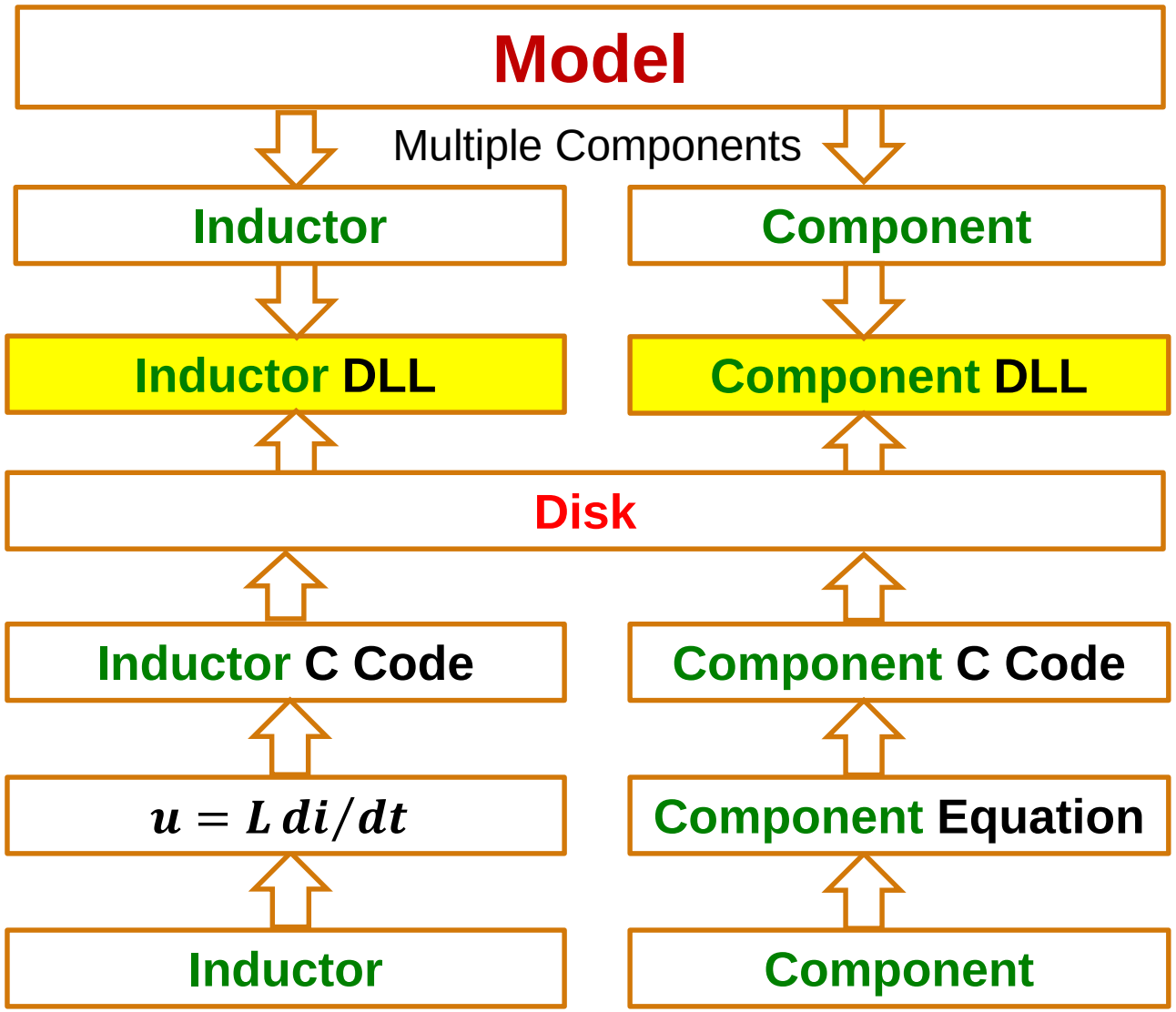
Component

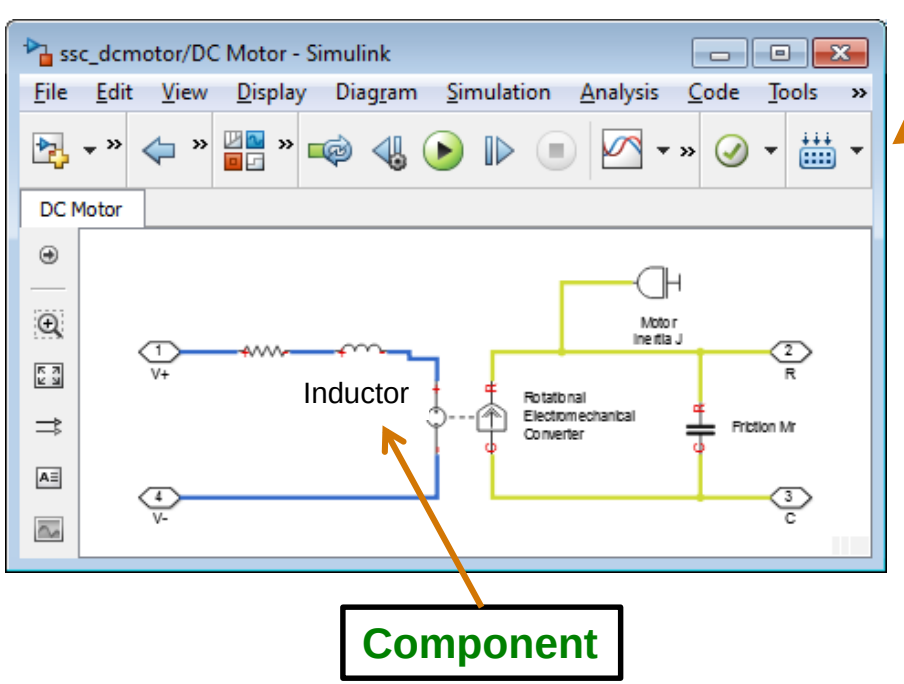
Model



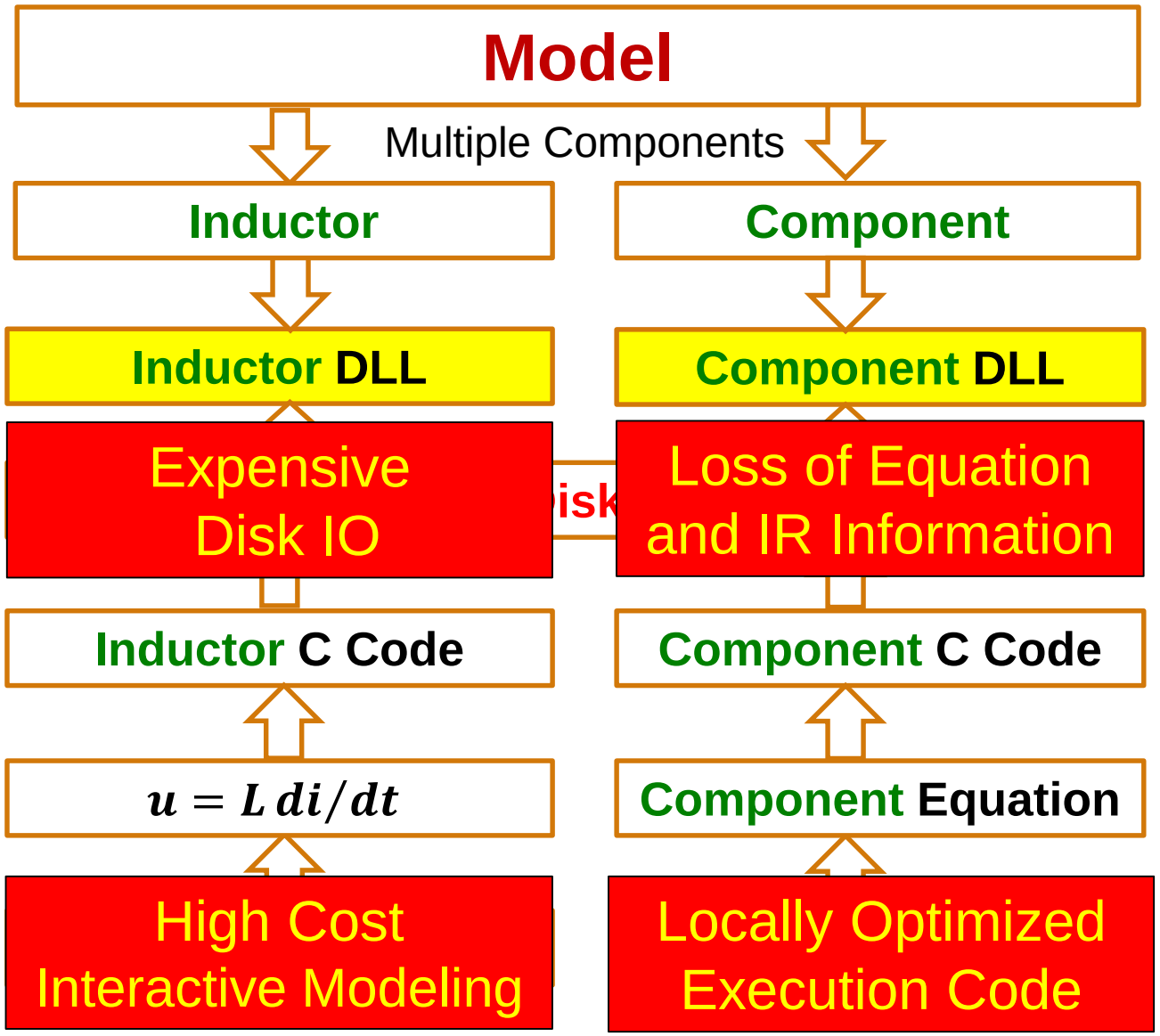


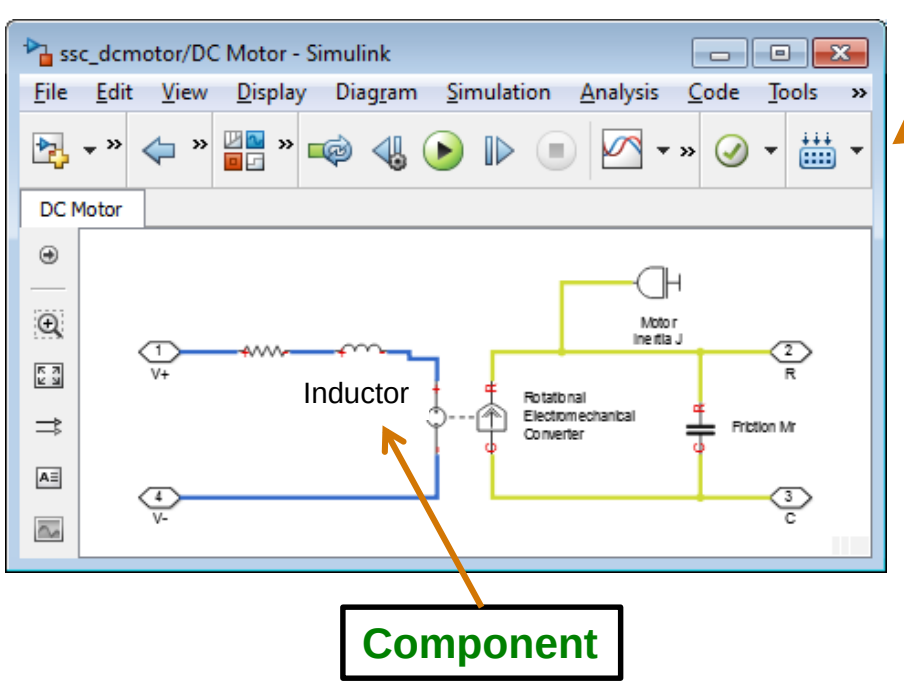
Shared Library-Based Simulation





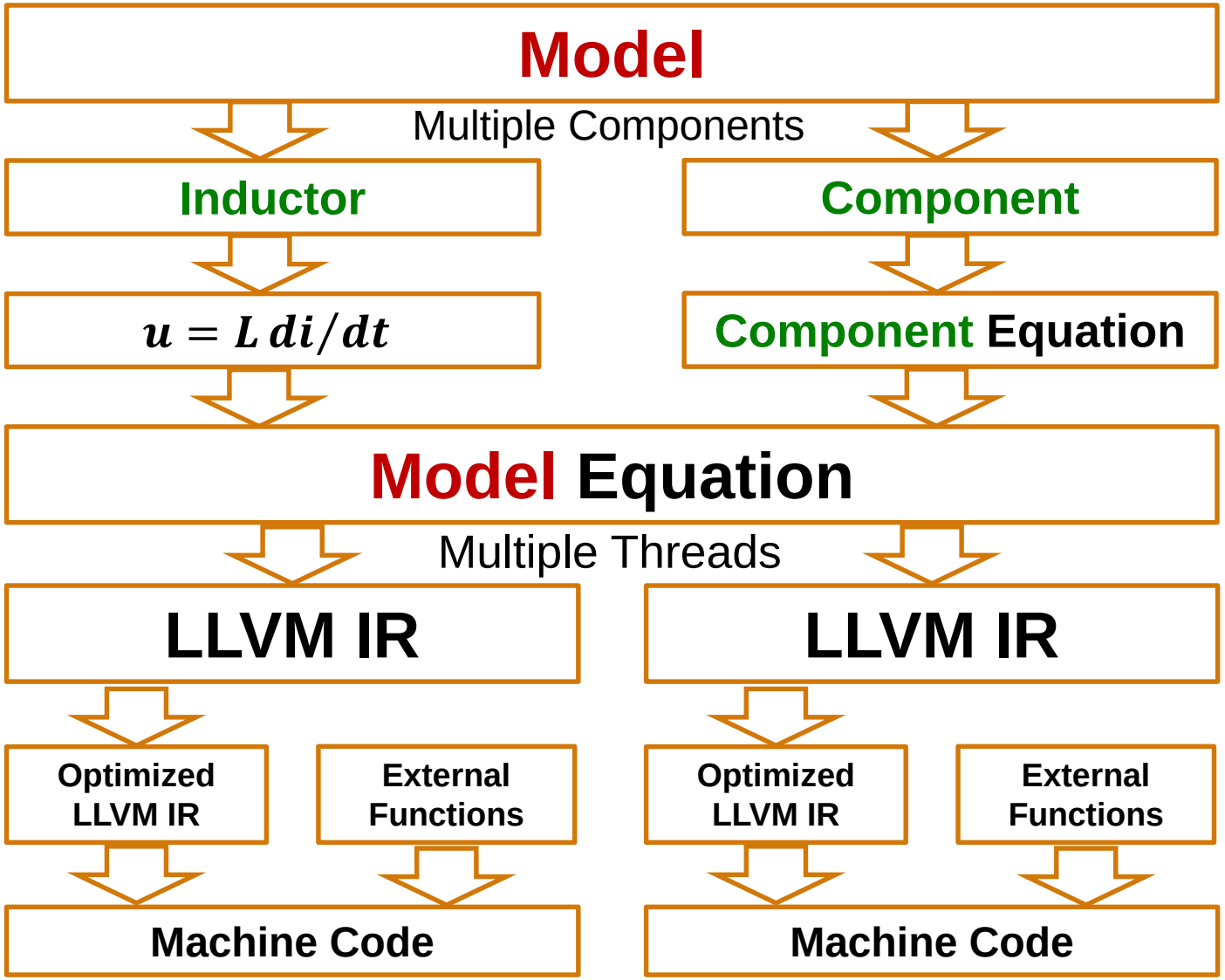
Shared Library-Based Simulation

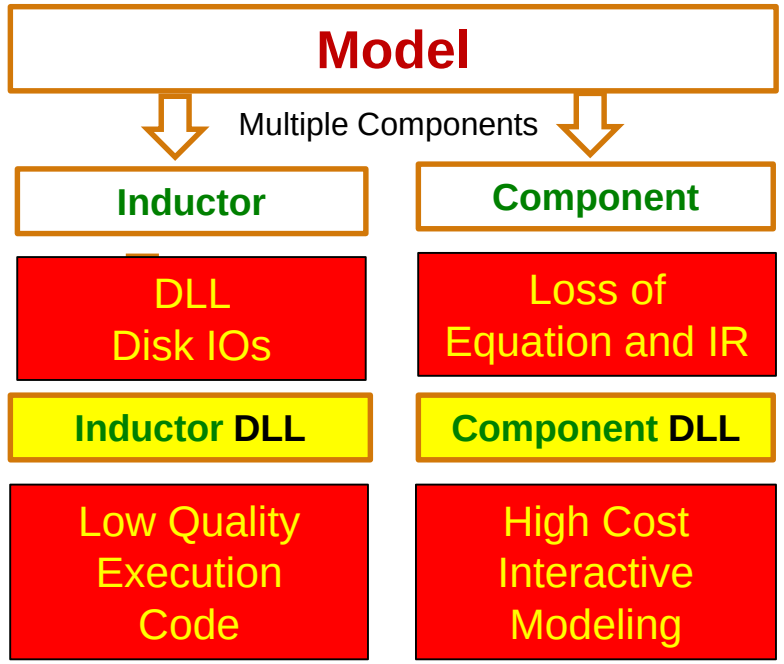




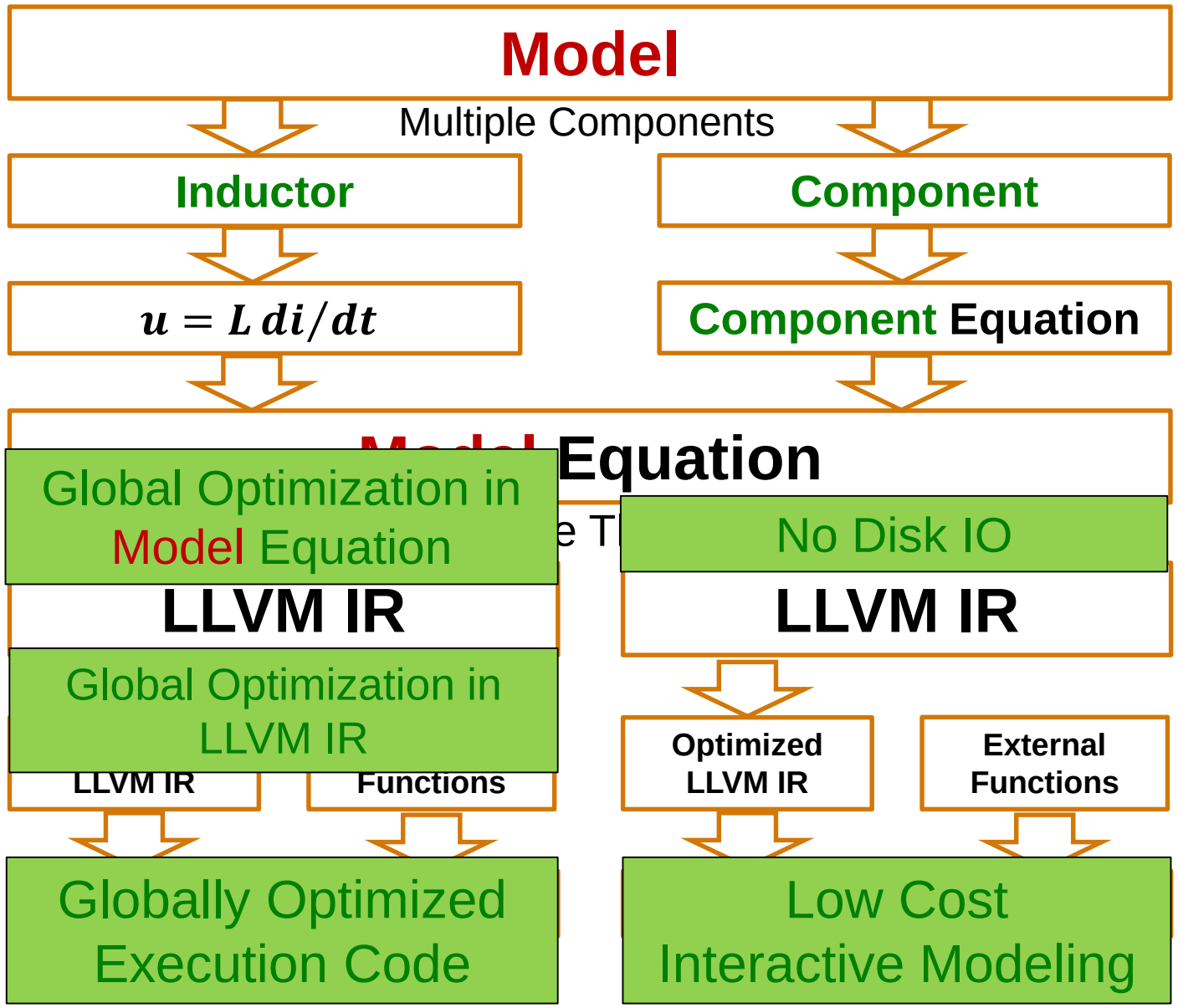
- Prevent global Model optimizations
  - Variable elimination
  - Index reduction
  - Constant folding
  - Function inline

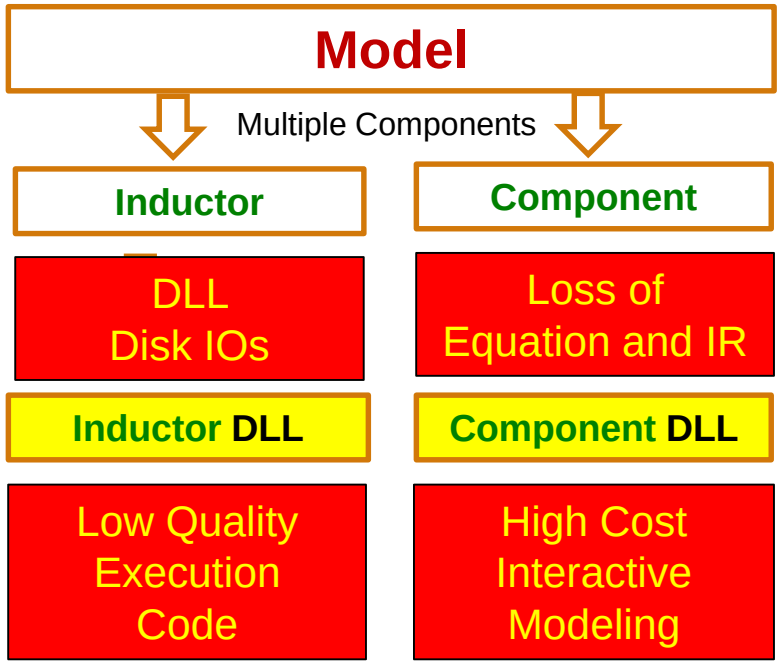
Multiple Thread LLVM-Based Simulation





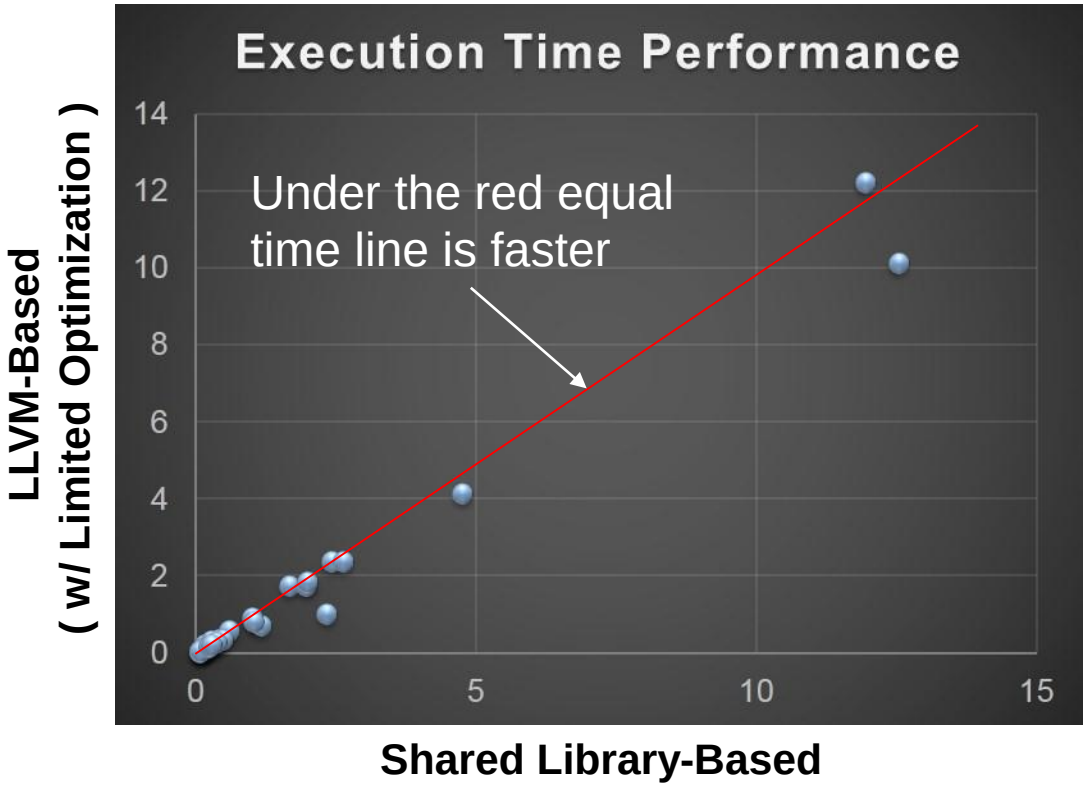
Multiple Thread LLVM-Based Simulation

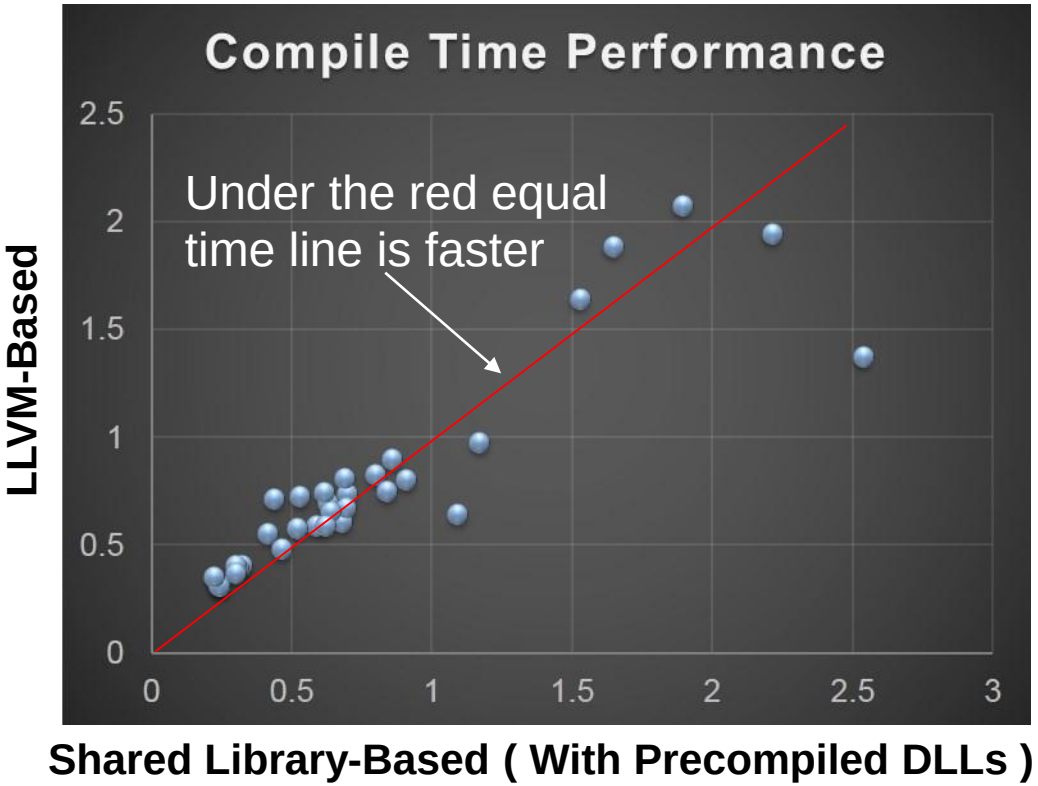




Results

- LLVM-based backend of new simulation engines in production code
  - Consistent floating point numerical computation
  - Support multiple threads on 64bit Linux, 32bit and 64bit Windows, 64bit Intel Mac
- Performance (win64 with LLVM 3.2 JIT)





Challenges

- Shared library support for large scale software
  - Multiple linkage of static LLVM library is painful
- JIT performance
  - CodeGenOpt: None, Less, Default, Aggressive (targeted for code quality)
  - New option targeted for fast JIT ?
- Exception handling on Windows
  - Propagate through JIT code the exception thrown from the external functions
- Legacy JIT to MCJIT transition

MathWorks

Thank You !