

# Hybrid Simulation: Integration Methods

Bozidar Stojadinovic, Associate Professor

University of California, Berkeley



**nees@berkeley**

The George E. Brown, Jr. Network for Earthquake Engineering Simulation



# Solution Strategy

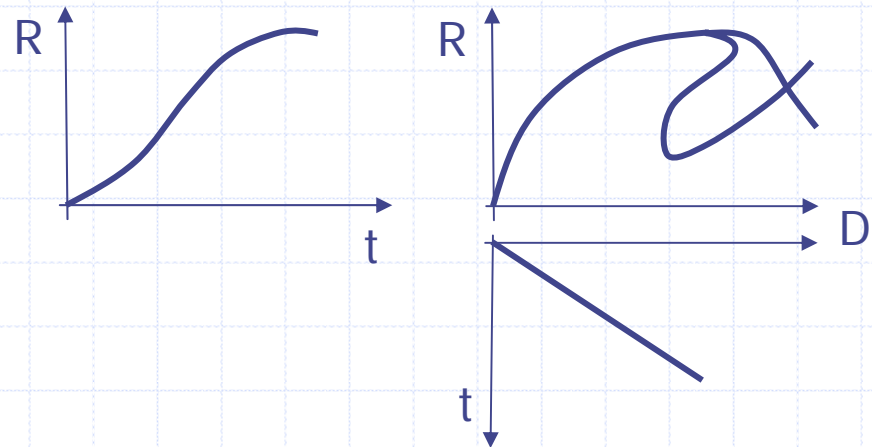
- ◆ Time-stepping integration:
  - Acquire the state
    - ◆ Assemble computed and measured state data
  - Extrapolate to the new state:
    - ◆ Explicit (knowing only the current state)
    - ◆ Implicit (iterate, assuming something about the future state and/or the way of getting there)
  - Move to the new state
    - ◆ Actuate the physical specimens
    - ◆ Iterate the computer specimens
- ◆ Communication (data transfer) is included in all states, too

# Limitations

- ◆ The physical model is alive:
  - Relaxes while under constant displacement or force
  - Sticks when starting to move
  - Develops a hysteresis when it unloads
  - Accumulates damage
- ◆ There is no way to erase an achieved state if it is found to be wrong, and go back to a previously converged state
- ◆ The computer model is fine

# Stepping Out to a New State

- ◆ Physical model moves forward in time
- ◆ Displacement or force control
  - No snap-back (displ)
  - No softening (force)
- ◆ Computer model may iterate!



# Choice of Integration Algorithm

## ◆ Explicit methods:

- Target displacement is computed using only current (or previous) state data
- No iteration required
- Conditionally stable: short time-step

## ◆ Implicit methods:

- Unconditionally stable: longer time-step
- Less sensitive to higher-mode excitation
- Require an assumption about the target state and iteration on that assumption
- Physical substructures cannot be iterated
- Require an accurate tangent stiffness matrix

# Explicit Methods

◆ Central  
Difference

$$Ma_{i+1} + Cv_{i+1} + r_i = f_{i+1}$$

◆ Newmark's  
method family:

$$d_{i+1} = d_i + \Delta tv_i + \Delta t^2 \left[ \left( \frac{1}{2} - \beta \right) a_i + \beta a_{i+1} \right]$$

$$v_{i+1} = v_i + \Delta t [(1 - \gamma)a_i + \gamma a_{i+1}]$$

■ Constant  
acceleration  
method

$$\beta = 1/4; \gamma = 1/2$$

■ Linear  
acceleration  
method

$$\beta = 1/6; \gamma = 1/2$$

# Explicit Methods

## ◆ Modified Newton method:

$$Ma_{i+1} + \left[ (1 + \alpha)K + \frac{\rho}{\Delta t^2}M \right] d_{i+1} = f_{i+1} + \left( \alpha K + \frac{\rho}{\Delta t^2}M \right) d_i$$

$$d_{i+1} = d_i + \Delta t v_i + \frac{1}{2} \Delta t^2 a_i$$

$$v_{i+1} = v_i + \frac{1}{2} \Delta t (a_i + a_{i+1})$$

- ◆ Select parameters to control numerical damping of higher modes  $\rho < 0; \alpha \geq 0$

# Implicit Methods

## ◆ Newmark alpha-method (HHT)

$$Ma_{i+1} + Cv_{i+1} + (1 + \alpha)r_{i+1} - \alpha r_i = f_{i+1}$$

$$d_{i+1} = d_i + \Delta t v_i + \Delta t^2 \left[ \left( \frac{1}{2} - \beta \right) a_i + \beta a_{i+1} \right]$$

$$v_{i+1} = v_i + \Delta t [(1 - \gamma)a_i + \gamma a_{i+1}]$$

$$\frac{1}{3} \leq \alpha \leq 0, \gamma = \left( \frac{1}{2} - \alpha \right), \beta = \frac{(1 - \alpha)^2}{4}$$

◆ Must adjust during time-step

# Variations of Newmark's Alpha-Method

## ◆ Analog/digital hybrid scheme:

- Use available force measurements during the time-step

## ◆ Iterative corrector scheme:

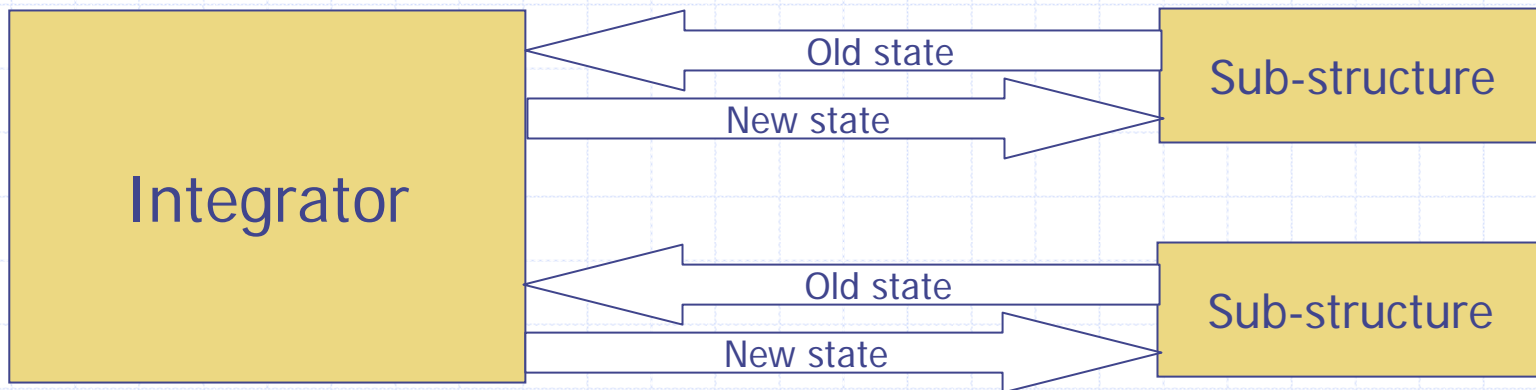
- Use a Newton-Raphson method and a tangent stiffness estimate to advance through the time-step

## ◆ Operator splitting methods:

- Explicit and implicit operators applied to different substructures

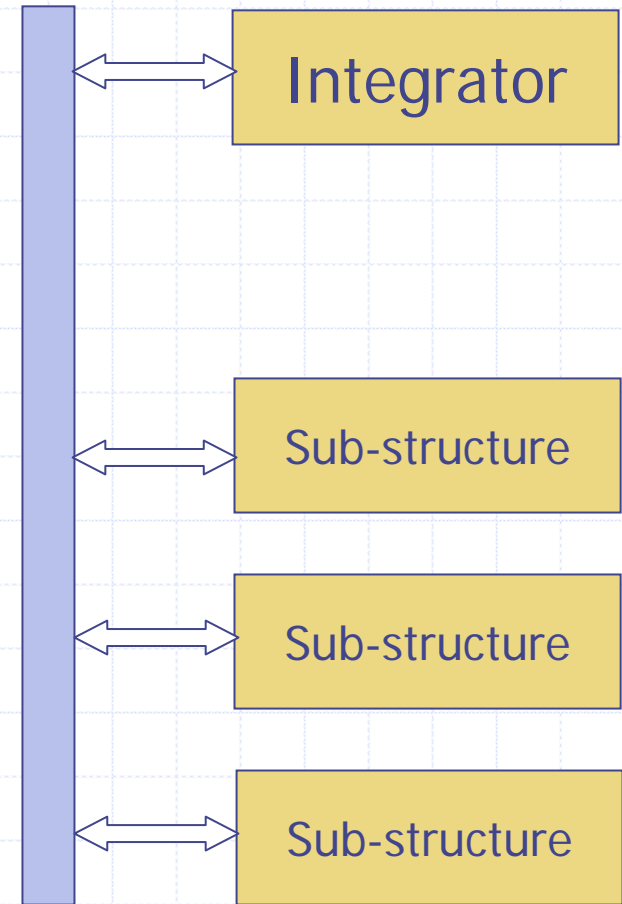
# Data Flow

- ◆ Integrator assembles state and extrapolates a new state
- ◆ Sub-structures implement the new state and report it back



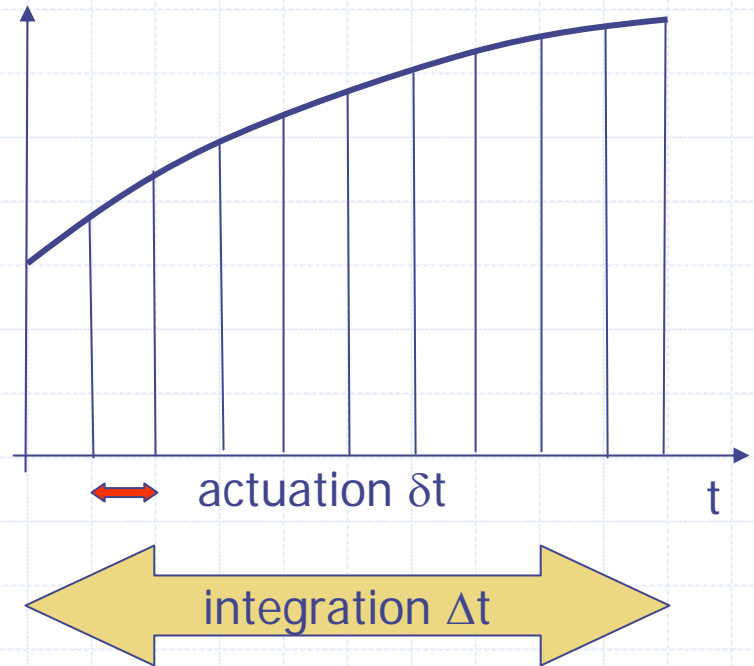
# Implementation

- ◆ Data bus links sub-structures to the integrator:
  - Local bus: local implementation
  - Use the internet as the bus: geographically distributed implementation
    - ◆ sub-structures and integrator are resources on a network, such as NEES



# Timing Constraints

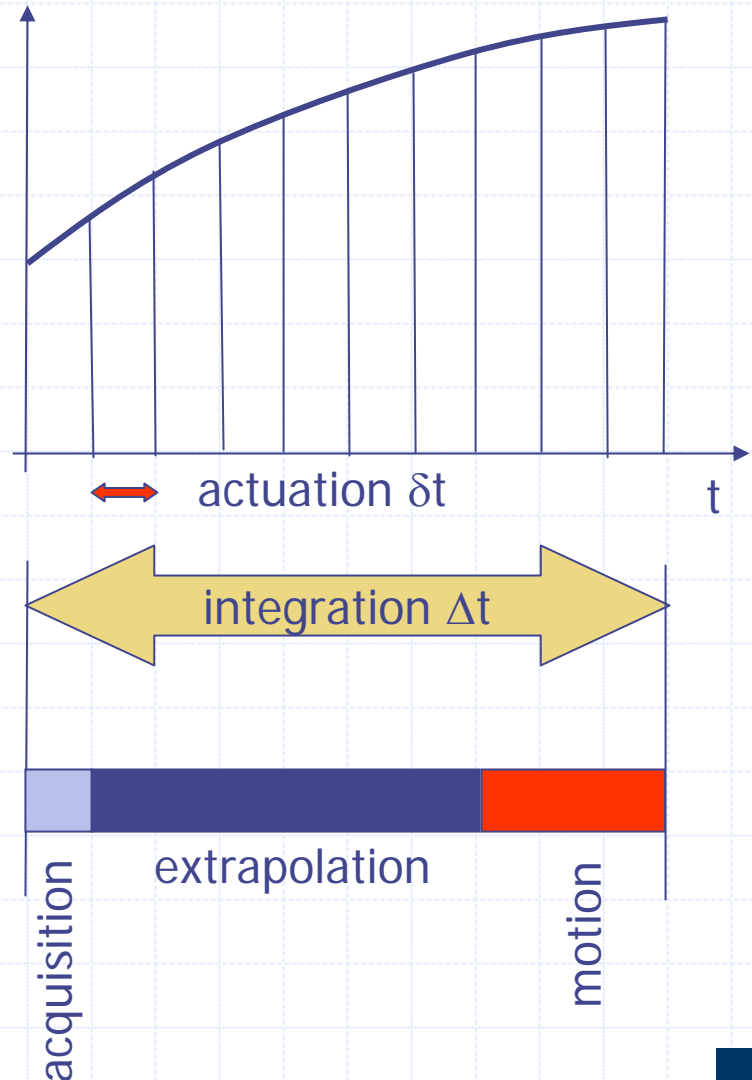
- ◆ Processes run at different rate:
  - Actuation 1000Hz
  - Integration 100Hz
  - Observation 10Hz (video?)
- ◆ To maintain continuous signal feed to the actuator we generate command signals during the integration time step



# Real-Time Simulation

◆ Integration time step governs the duration of all other activities:

- Acquisition of state (communication, assembly)
- Extrapolation of the new state (solving)
- Motion to the new state (communication, actuation, iteration)

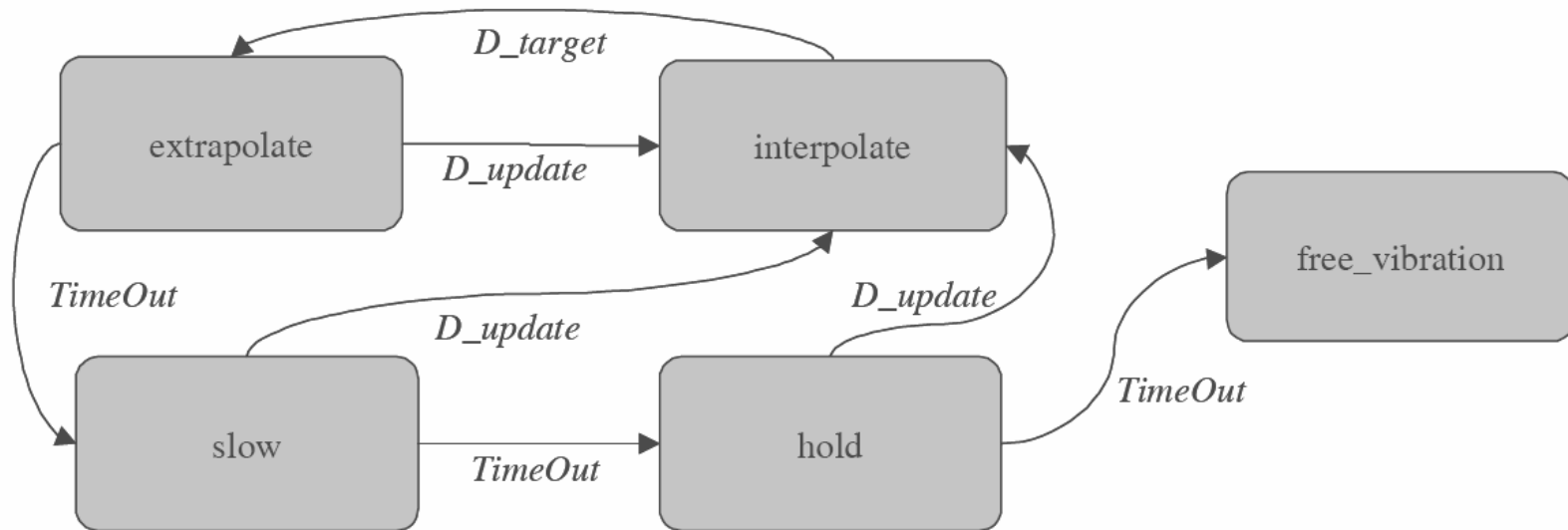


# Sources of Delays

- ◆ Communication while assembling state
- ◆ Solution while extrapolating a step
- ◆ Communication while sending a new state
- ◆ Delay while applying a new state:
  - Physical model: actuator time delay
  - Computer model: iteration
- ◆ These are random!
  - we know the distributions, but not the duration of a particular delay

# Event-Driven Simulation

- ◆ Define states of the hybrid simulation
- ◆ Transition on available information



## Legend

State:   
State Transition Path:   
Event causing State Transition: *Event/functionCall()*

# Finite State Machine

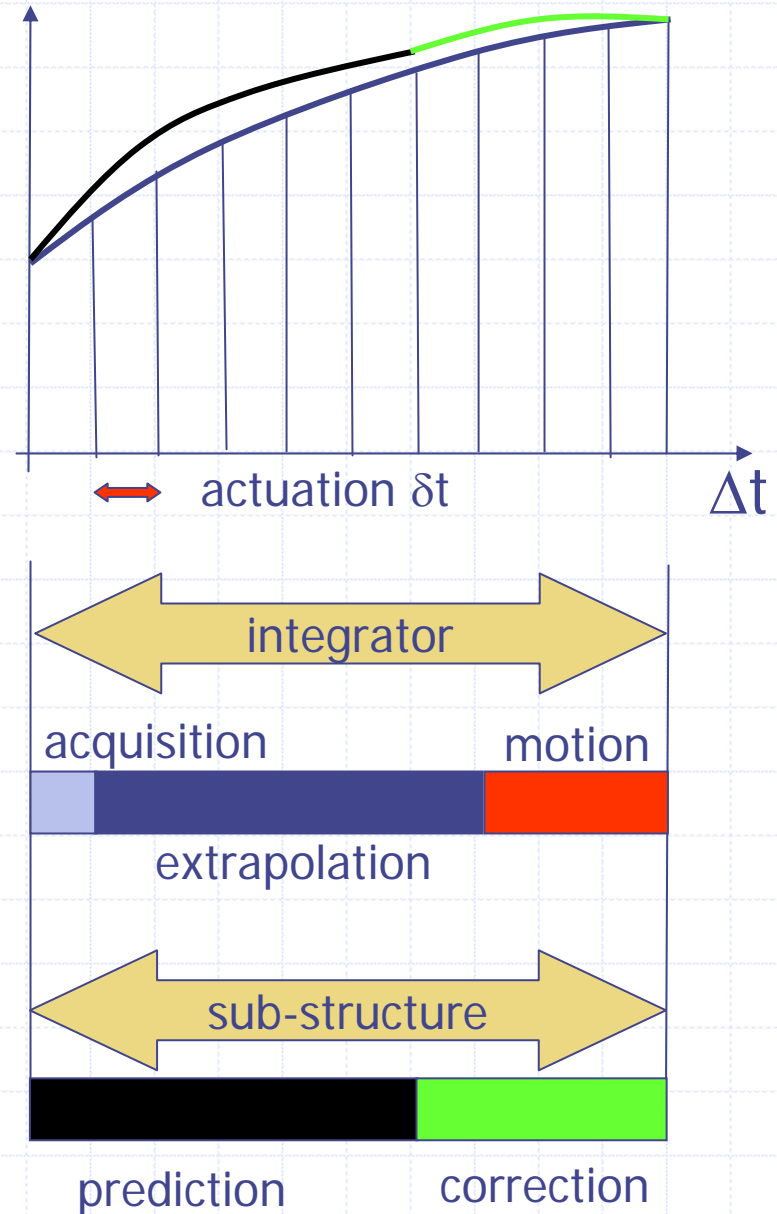
- ◆ If there are no delays, then state transitions are not an overload, and the simulation is real-time
- ◆ If delays prevail, the simulations slows down and/or halts:
  - Error is incurred due to:
    - ◆ Low velocity
    - ◆ Discontinuity (stop and go)
- ◆ Simulation may fail if data does not arrive!

# Implementation

◆ Separate integrator and sub-structure processes:

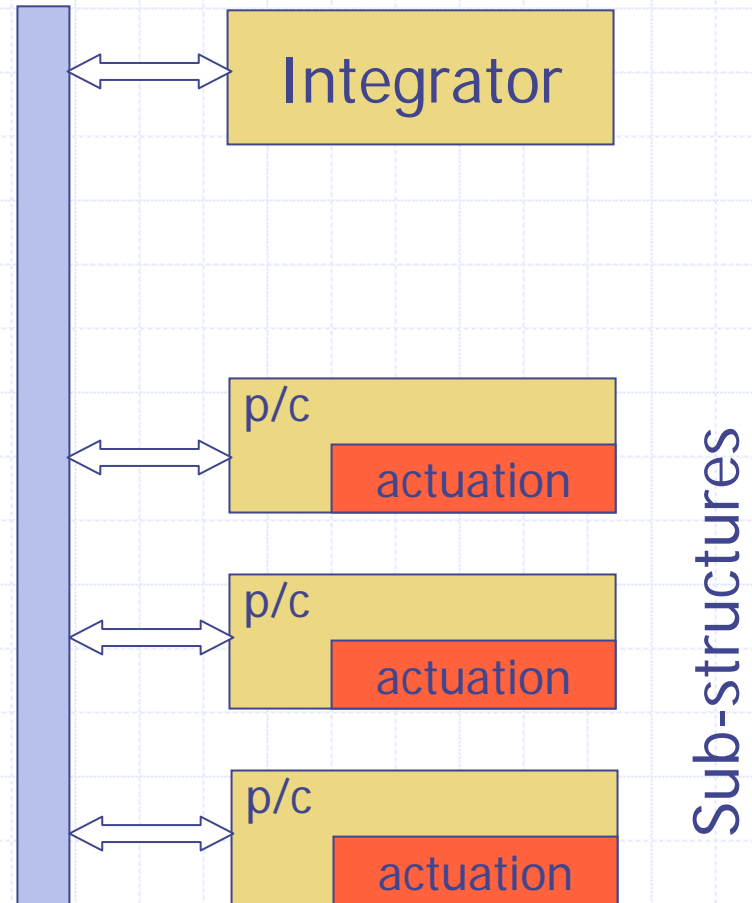
- Start issuing actuator commands using a local (fast) estimator (predictor) of the new (target) state
- Correct the trajectory when the true target state arrives from the integrator

◆ Error is incurred in the process!



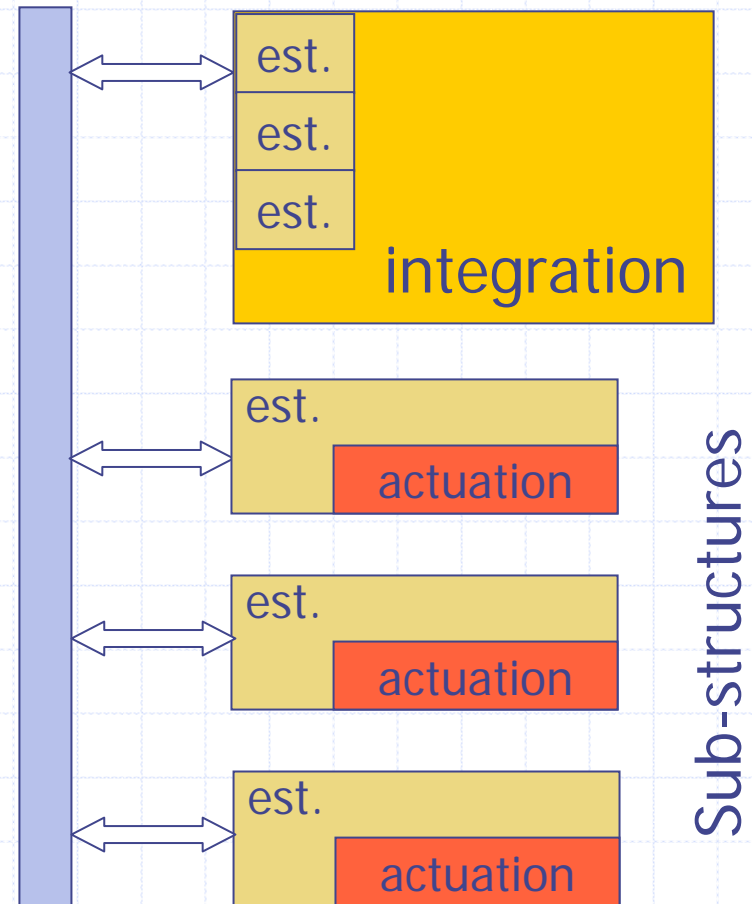
# Three-loop Architecture

- ◆ Local estimators (predictor and corrector) at sub-structure level act as buffers between the asynchronous integrator and actuation systems



# Extensions

- ◆ Use local estimators on both sides of data links:
  - They are system ID units that model of the interaction between the rest of the structure and the sub-structure in a simplified way
  - They are fast and local (no delay)



# Thank you!

Development and operation of the *nees@berkeley* equipment site is sponsored by NSF.

<http://nees.berkeley.edu>

Contributions to this presentation from Prof. Gilberto Mosqueda are gratefully acknowledged.



**nees@berkeley**

The George E. Brown, Jr. Network for Earthquake Engineering Simulation

