IOWA STATE UNIVERSITY

ECpE Department

GridLAB-D Tutorial

GRA: Rui Cheng

Advisor: Dr. Zhaoyu Wang

Department of Electrical and Computer Engineering

Iowa State University

Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help

- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape

- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

What is GridLAB-D

- Introduction to GridLAB-D
  - GridLAB-D is a power system simulation tool
  - It provides valuable information to users who design and operate electric power transmission and distribution systems, and to utilities that wish to take advantage of the latest smart grid technology.
  - It contains many power system models, including overhead line, HVAC, PV, battery, load and so on.

Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help

- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape

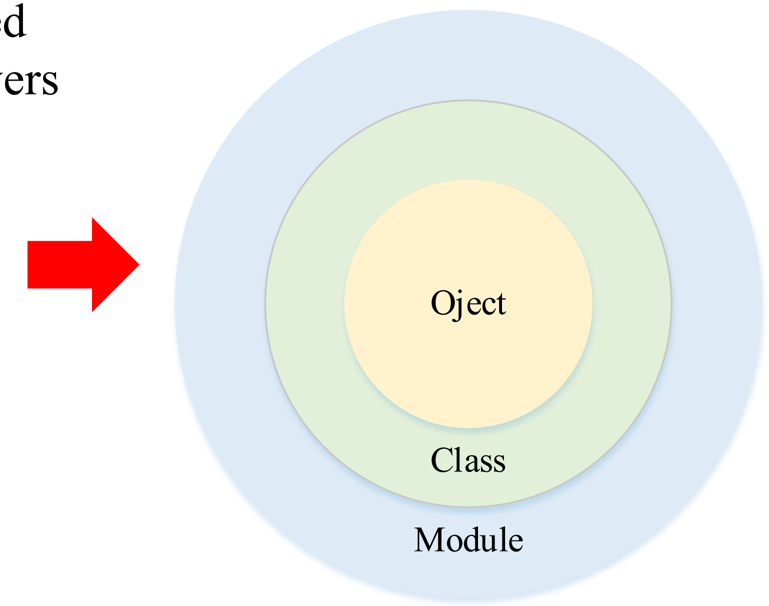
- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

What is GridLAB-D

- GridLAB-D Key Attributes
  - Open Source
    - Code is readily available for inspection
    - Correction or expansion of the code base to meet the user's particular need is possible.
  - Command-Line
    - Pure command-line program
    - No GUI for user
    - Import commands into a system terminal or command line

What is GridLAB-D

- GridLAB-D Key Attributes
  - Object-Based
    - Module:
      - ➢ The collection of classes that are related
      - ➢ It groups common parameters and solvers
    - Class:
      - ➢ It defines similar objects
      - ➢ Properties are the same for all objects, but set parametrically
      - ➢ Static classes are pre-compiled
    - Object:
      - ➢ Instances of classes
      - ➢ Used to define the different agents in the system
      - ➢ Each property is assigned an individualized value



What is GridLAB-D

- GridLAB-D Key Attributes
  - Multi-Domain Models
  - ➢ GridLAB-D is commonly represented as a distribution system and/or smart grid simulator with incredibly useful built-in classes that are more than what you might find in a traditional distribution system simulator
  - ➢ It contains classes for all may common distribution components(such as transformers, lines, voltage regulators)
  - ➢ The interactions between multi-domain models are considered.

  - Discrete Time
  - ➢ GridLAB-D is a discrete time simulator
  - ➢ These simulators are not actually continuous.
  - ➢ But the time steps used by the simulator are small enough and regularly spaced so as to closely and reasonably approximate continuous time.

Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help

- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape

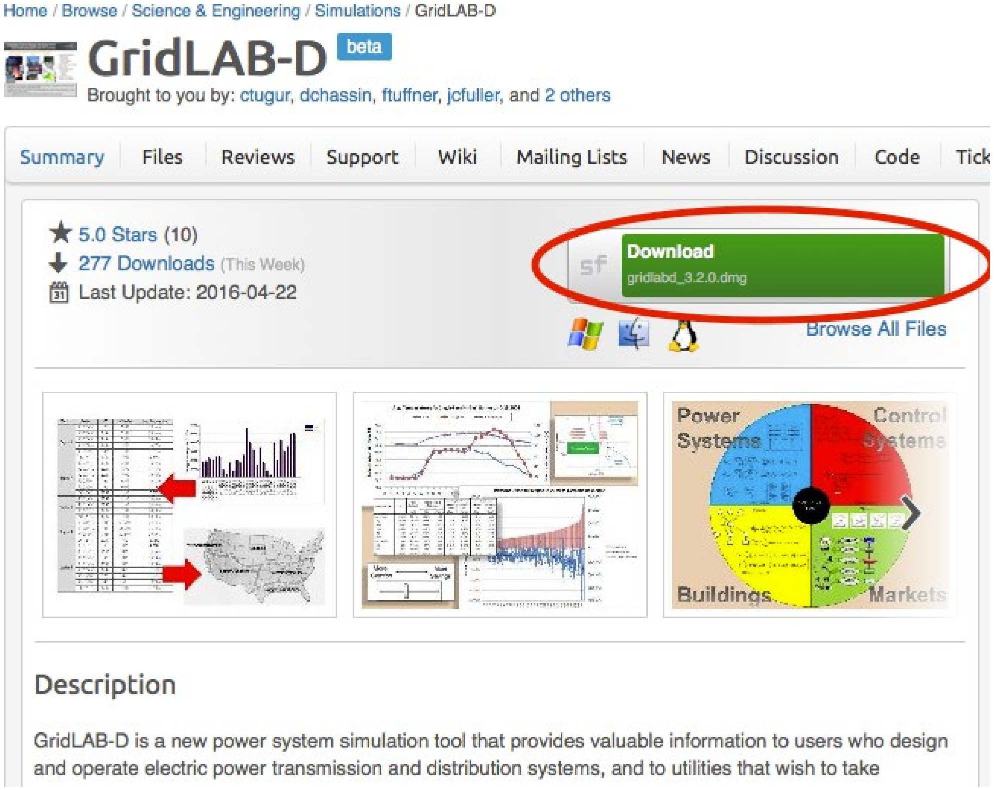
- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

What is GridLAB-D

- Installing GridLAB-D: Using a traditional installer

Using a traditional installer:

- ➢ Website: https://sourceforge.net/projects/gridlab-d/
- ➢ Once the installer has been downloaded, launch it and walk through the process of installing the software as you would for any other commercial software



What is GridLAB-D

- Installing GridLAB-D
  - File Locations (the default directory for GridLAB-D on Windows is as follows):

➢ c:\Program Files\GridLAB-D

This is the main GridLAB-D directory. It contains all the subdirectories as well as the readme and uninstall files.

➢ c:\Program Files\GridLAB-D\bin

This contains all the executables. The PATH environment variable should include this directory.

➢ c:\Program Files\GridLAB-D\etc

This contains all the runtime files. The GLPATH environment variable should include this directory.

➢ c:\Program Files\GridLAB-D\lib

This contains all the module files. Both the PATH and GLPATH environment variables should include this directory.

➢ c:\Program Files\GridLAB-D\samples

This contains sample data files.

What is GridLAB-D

- Installing GridLAB-D
  - Environment variables:

➢ PATH: The Windows PATH environment must include both the bin and lib directories.

➢ GLPATH: GridLAB-D uses the GLPATH environment to find runtime files and module files. It should include at least the etc and lib directories.

➢ GRIDLAB-D: This should contain the path where GridLAB-D is installed.

Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help

- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape

- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

What is GridLAB-D

- Getting help
  - Running resource: --help, --modhelp module[:class]

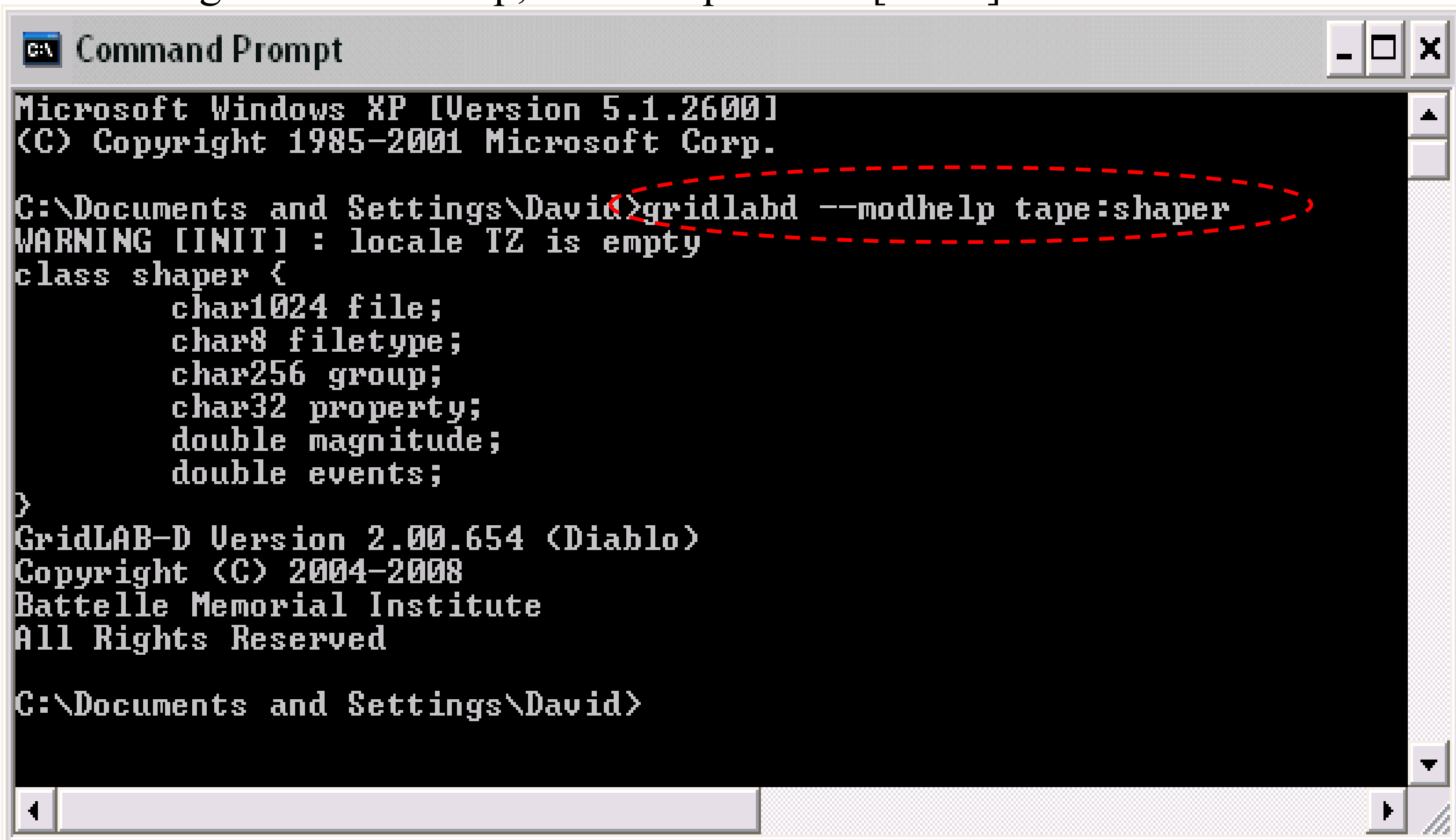
Command Prompt

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\David>gridlabd --help
WARNING [INIT] : locale TZ is empty
Syntax: gridlabd [OPTIONS ...] <file> ...
Options:
  --avlbalance               toggles AVL tree balancing
  -c|--check                 toggles module checks after model loads
  -D|--define <def>          defines a macro value
  --debug                    toggles debug output (prints internal messages)
  --debugger                 toggles debugger mode (generates internal messages)
  --dumpall                  toggles module data dump after run completes
  -e|--environment <name>    specifies user environment (default none)
  --license                  print license information
  -L|--libinfo <module>      print module library information
  -o|--output <file>         specifies model should be output after run
  --profile                  toggles profilers
  -q|--quiet                 toggles quiet mode (suppresses startup banner)
  --test                     toggles test mode (activate testing procedures)
  -T|--threadcount <n>       specifies the number of processor threads to use
  -v|--verbose               toggles verbose mode (active verbose messages)
  -V|--version               prints the GridlabD version information
  -w|--warn                  toggles warning mode (generates warning messages)
  --xmlencoding <num>        set the XML encoding (8, 16, or 32)
  --xmlstrict                toggles XML encoding to be strict
  --xsd <module>[:<object>] prints the xsd of an object
  --xsl <modlist>            prints the xsl for the modules listed

C:\Documents and Settings\David>_
```

Example: --help option

What is GridLAB-D

- Getting help:
  - Running resource: --help, --modhelp module[:class]



Example: --modhelp tape:shaper

What is GridLAB-D

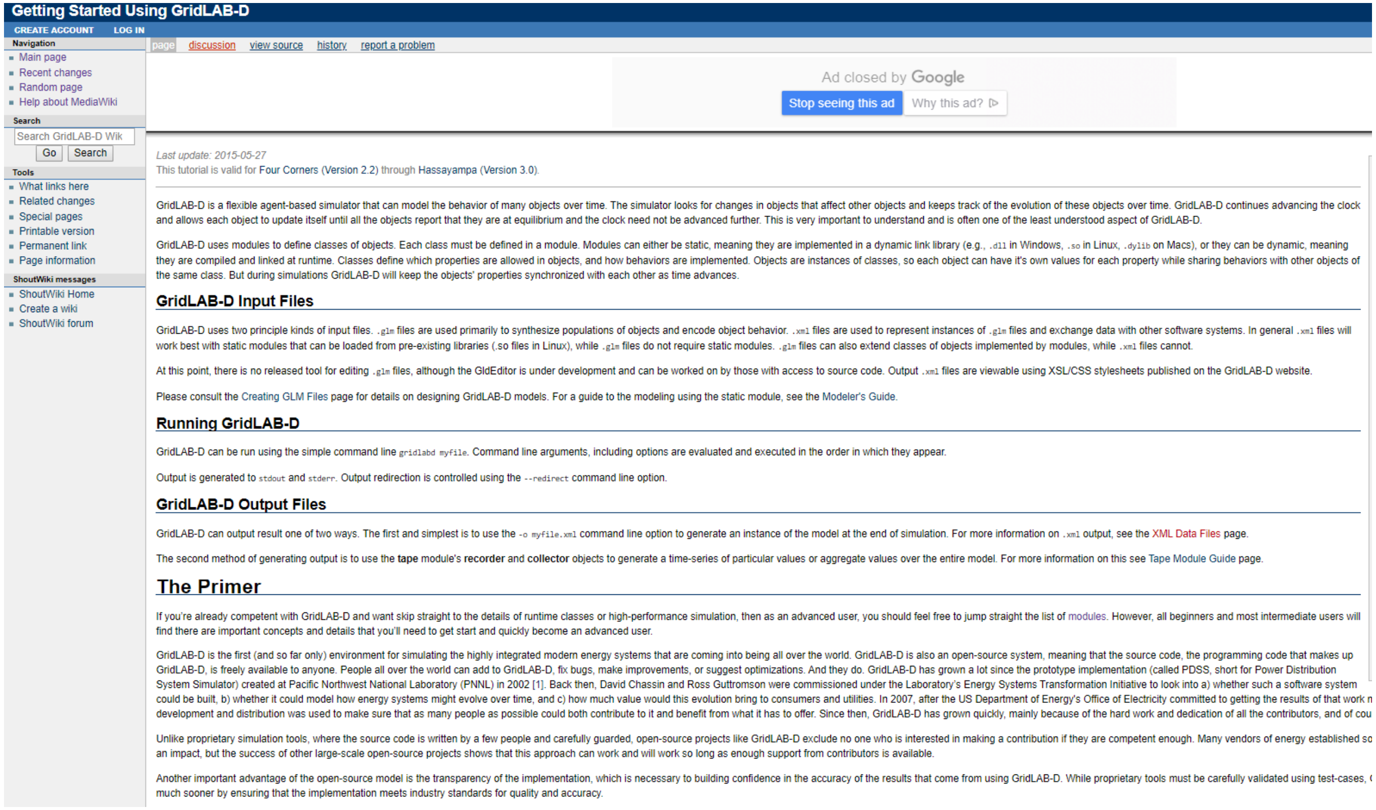
Command Prompt (2)

```
C:\Documents and Settings\d3x289\My Documents\GLD_Test\trunk\VS2005\Win32\Release>gridlabd --modhelp powerflow:line
WARNING [INIT] : locale TZ is empty
module powerflow {
        bool show_matrix_values;
        double primary_voltage_ratio;
        double nominal_frequency;
        bool require_voltage_control;
        double geographic_degree;
        complex fault_impedance;
        double warning_underfrequency;
        double warning_overfrequency;
        double warning_undervoltage;
        double warning_overvoltage;
        double warning_voltageangle;
        double maximum_voltage_error;
        enumeration {NR=2, GS=1, FBS=0} solver_method;
        double acceleration_factor;
        int32 NR_iteration_limit;
        int32 NR_superLU_procs;
        double default_maximum_voltage_error;
}
class line {
        parent link;
        class link {
                parent powerflow_object;
                class powerflow_object {
                        set {A=1, B=2, C=4, D=256, N=8, S=112, G=128} phases;
                        double nominal_voltage[V];
                }

                enumeration {OPEN=0, CLOSED=1} status;
                object from;
                object to;
                complex power_in[VA];
                complex power_out[VA];
                complex power_losses[VA];
                complex power_in_A[VA];
                complex power_in_B[VA];
                complex power_in_C[VA];
                complex power_out_A[VA];
                complex power_out_B[VA];
                complex power_out_C[VA];
                complex power_losses_A[VA];
                complex power_losses_B[VA];
                complex power_losses_C[VA];
                complex current_out_A[A];
                complex current_out_B[A];
                complex current_out_C[A];
                complex current_in_A[A];
                complex current_in_B[A];
                complex current_in_C[A];
                set {CN=768, CR=512, CF=256, BN=48, BR=32, BF=16, AN=3, AR=2, AF=1, UNKNOWN=0} flow_direction;
        }

        object configuration;
        double length[ft];
}
```

Example: --modhelp powerflow:line

What is GridLAB-D

- Getting help
  - GridLAB-D wiki pages--These two pages provide earlier attempts to help new user's getting started in GridLAB-D.



Getting Started Using GridLAB-D

CREATE ACCOUNT LOG IN

Navigation
- Main page
- Recent changes
- Random page
- Help about MediaWiki

Search

Go Search

Tools
- What links here
- Related changes
- Special pages
- Printable version
- Permanent link
- Page information

ShoutWiki messages
- ShoutWiki Home
- Create a wiki
- ShoutWiki forum

page discussion view source history report a problem

Last update: 2015-05-27
This tutorial is valid for Four Corners (Version 2.2) through Hassayampa (Version 3.0).

GridLAB-D is a flexible agent-based simulator that can model the behavior of many objects over time. The simulator looks for changes in objects that affect other objects and keeps track of the evolution of these objects over time. GridLAB-D continues advancing the clock and allows each object to update itself until all the objects report that they are at equilibrium and the clock need not be advanced further. This is very important to understand and is often one of the least understood aspect of GridLAB-D.

GridLAB-D uses modules to define classes of objects. Each class must be defined in a module. Modules can either be static, meaning they are implemented in a dynamic link library (e.g., `.dll` in Windows, `.so` in Linux, `.dylib` on Macs), or they can be dynamic, meaning they are compiled and linked at runtime. Classes define which properties are allowed in objects, and how behaviors are implemented. Objects are instances of classes, so each object can have it's own values for each property while sharing behaviors with other objects of the same class. But during simulations GridLAB-D will keep the objects' properties synchronized with each other as time advances.

GridLAB-D Input Files

GridLAB-D uses two principle kinds of input files. `.glm` files are used primarily to synthesize populations of objects and encode object behavior. `.xml` files are used to represent instances of `.glm` files and exchange data with other software systems. In general `.xml` files will work best with static modules that can be loaded from pre-existing libraries (.so files in Linux), while `.glm` files do not require static modules. `.glm` files can also extend classes of objects implemented by modules, while `.xml` files cannot.

At this point, there is no released tool for editing `.glm` files, although the GldEditor is under development and can be worked on by those with access to source code. Output `.xml` files are viewable using XSL/CSS stylesheets published on the GridLAB-D website.

Please consult the Creating GLM Files page for details on designing GridLAB-D models. For a guide to the modeling using the static module, see the Modeler's Guide.

Running GridLAB-D

GridLAB-D can be run using the simple command line `gridlabd myfile`. Command line arguments, including options are evaluated and executed in the order in which they appear.

Output is generated to `stdout` and `stderr`. Output redirection is controlled using the `--redirect` command line option.

GridLAB-D Output Files

GridLAB-D can output result one of two ways. The first and simplest is to use the `-o myfile.xml` command line option to generate an instance of the model at the end of simulation. For more information on `.xml` output, see the XML Data Files page.

The second method of generating output is to use the **tape** module's **recorder** and **collector** objects to generate a time-series of particular values or aggregate values over the entire model. For more information on this see Tape Module Guide page.

The Primer

If you're already competent with GridLAB-D and want skip straight to the details of runtime classes or high-performance simulation, then as an advanced user, you should feel free to jump straight the list of modules. However, all beginners and most intermediate users will find there are important concepts and details that you'll need to get start and quickly become an advanced user.

GridLAB-D is the first (and so far only) environment for simulating the highly integrated modern energy systems that are coming into being all over the world. GridLAB-D is also an open-source system, meaning that the source code, the programming code that makes up GridLAB-D, is freely available to anyone. People all over the world can add to GridLAB-D, fix bugs, make improvements, or suggest optimizations. And they do. GridLAB-D has grown a lot since the prototype implementation (called PDSS, short for Power Distribution System Simulator) created at Pacific Northwest National Laboratory (PNNL) in 2002 [1]. Back then, David Chassin and Ross Guttromson were commissioned under the Laboratory's Energy Systems Transformation Initiative to look into a) whether such a software system could be built, b) whether it could model how energy systems might evolve over time, and c) how much value would this evolution bring to consumers and utilities. In 2007, after the US Department of Energy's Office of Electricity committed to getting the results of that work n development and distribution was used to make sure that as many people as possible could both contribute to it and benefit from what it has to offer. Since then, GridLAB-D has grown quickly, mainly because of the hard work and dedication of all the contributors, and of cou

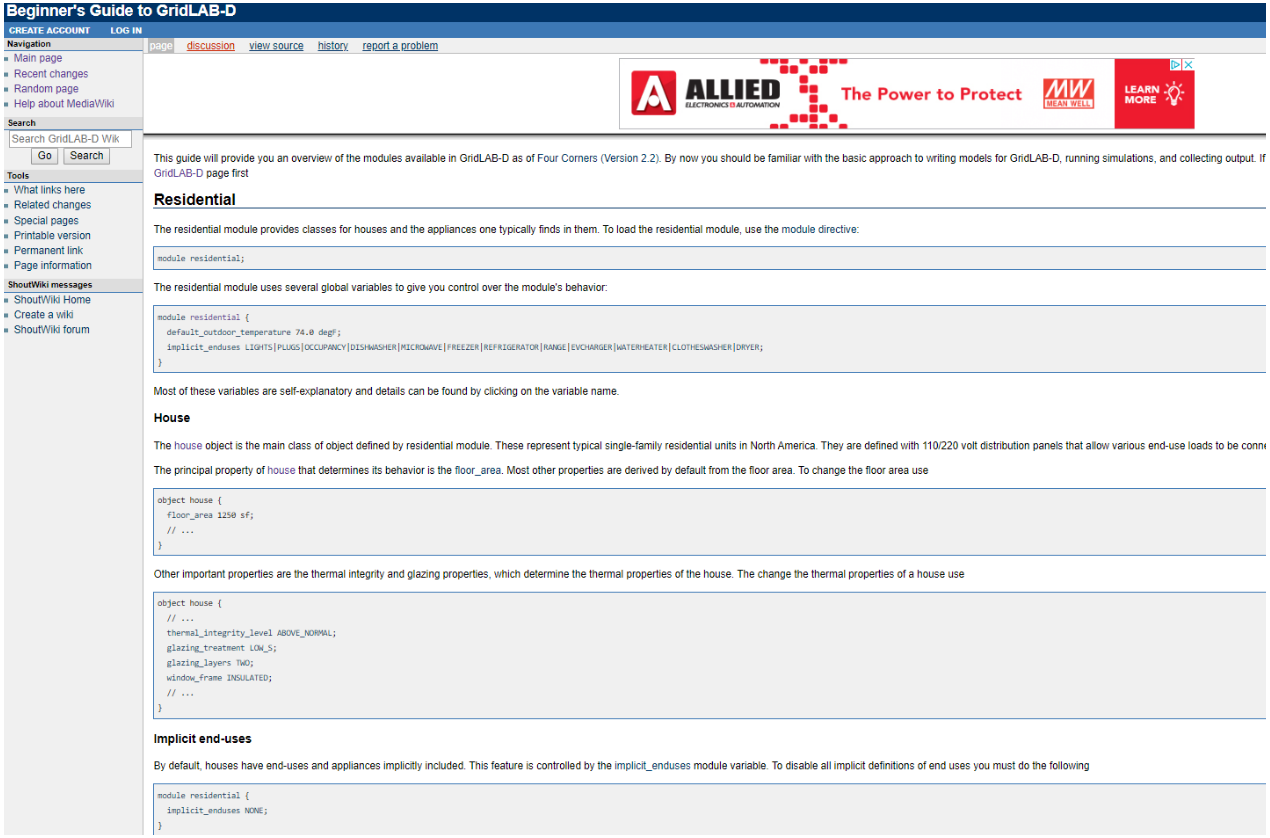
Unlike proprietary simulation tools, where the source code is written by a few people and carefully guarded, open-source projects like GridLAB-D exclude no one who is interested in making a contribution if they are competent enough. Many vendors of energy established so an impact, but the success of other large-scale open-source projects shows that this approach can work and will work so long as enough support from contributors is available.

Another important advantage of the open-source model is the transparency of the implementation, which is necessary to building confidence in the accuracy of the results that come from using GridLAB-D. While proprietary tools must be carefully validated using test-cases, much sooner by ensuring that the implementation meets industry standards for quality and accuracy.

http://gridlab-d.shoutwiki.com/wiki/Getting_Started_Using_GridLAB-D

What is GridLAB-D

- Getting help
  - GridLAB-D wiki pages--These two pages provide earlier attempts to help new user's getting started in GridLAB-D.


Beginner's Guide to GridLAB-D

CREATE ACCOUNT LOG IN

Navigation
- Main page
- Recent changes
- Random page
- Help about MediaWiki

Search

Search GridLAB-D Wik

Go Search

Tools
- What links here
- Related changes
- Special pages
- Printable version
- Permanent link
- Page information

ShoutWiki messages
- ShoutWiki Home
- Create a wiki
- ShoutWiki forum

page discussion view source history report a problem

ALLIED ELECTRONICS & AUTOMATION The Power to Protect MW MEAN WELL LEARN MORE

This guide will provide you an overview of the modules available in GridLAB-D as of Four Corners (Version 2.2). By now you should be familiar with the basic approach to writing models for GridLAB-D, running simulations, and collecting output. If GridLAB-D page first

Residential

The residential module provides classes for houses and the appliances one typically finds in them. To load the residential module, use the module directive:

```
module residential;
```

The residential module uses several global variables to give you control over the module's behavior:

```
module residential {
  default_outdoor_temperature 74.0 degF;
  implicit_enduses LIGHTS|PLUGS|OCCUPANCY|DISHWASHER|MICROWAVE|FREEZER|REFRIGERATOR|RANGE|EVCHARGER|WATERHEATER|CLOTHESWASHER|DRYER;
}
```

Most of these variables are self-explanatory and details can be found by clicking on the variable name.

House

The house object is the main class of object defined by residential module. These represent typical single-family residential units in North America. They are defined with 110/220 volt distribution panels that allow various end-use loads to be conn

The principal property of house that determines its behavior is the floor_area. Most other properties are derived by default from the floor area. To change the floor area use

```
object house {
  floor_area 1250 sf;
  // ...
}
```

Other important properties are the thermal integrity and glazing properties, which determine the thermal properties of the house. The change the thermal properties of a house use

```
object house {
  // ...
  thermal_integrity_level ABOVE_NORMAL;
  glazing_treatment LOW_S;
  glazing_layers TWO;
  window_frame INSULATED;
  // ...
}
```

Implicit end-uses

By default, houses have end-uses and appliances implicitly included. This feature is controlled by the implicit_enduses module variable. To disable all implicit definitions of end uses you must do the following

```
module residential {
  implicit_enduses NONE;
}
```

http://gridlab-d.shoutwiki.com/wiki/Beginner%27s_Guide_to_GridLAB-D

Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help
- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape
- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

GridLAB-D Module

- Power flow Module

  Classes：

  - billdump – Billing data dump on meter objects at specified times.
  - currdump – Current data dump on link object at specified times.
  - powerflow_library – Abstract class for objects the only contain data but don't synchronize.
    - emissions – Emissions library object
    - line_configuration – Line configuration library object
    - line_spacing – Link spacing library object
    - overhead_line_conductor – Overhead conductor library object
    - power_metrics – Reliability metrics container
    - regulator_configuration – Regulator configuration library object
    - restoration – Restoration control library object
    - transformer_configuration – Transformer configuration library object
    - triplex_line_configuration – Triplex line configuration library object
    - underground_line_conductor – Underground line conductor configuration library object
  - powerflow_object – Abstract class for object the are included in the flow solution
    - fault_check – Fault identification object for reliability analysis
    - frequency_gen – Frequency generation object
    - link – Abstract link (branch) object.
      - fuse – Fusable link object.
      - line – Generic line object
        - overhead_line – Overhead line object
        - triplex_line – Triplex line object
        - underground_line – Underground line object

GridLAB-D Module

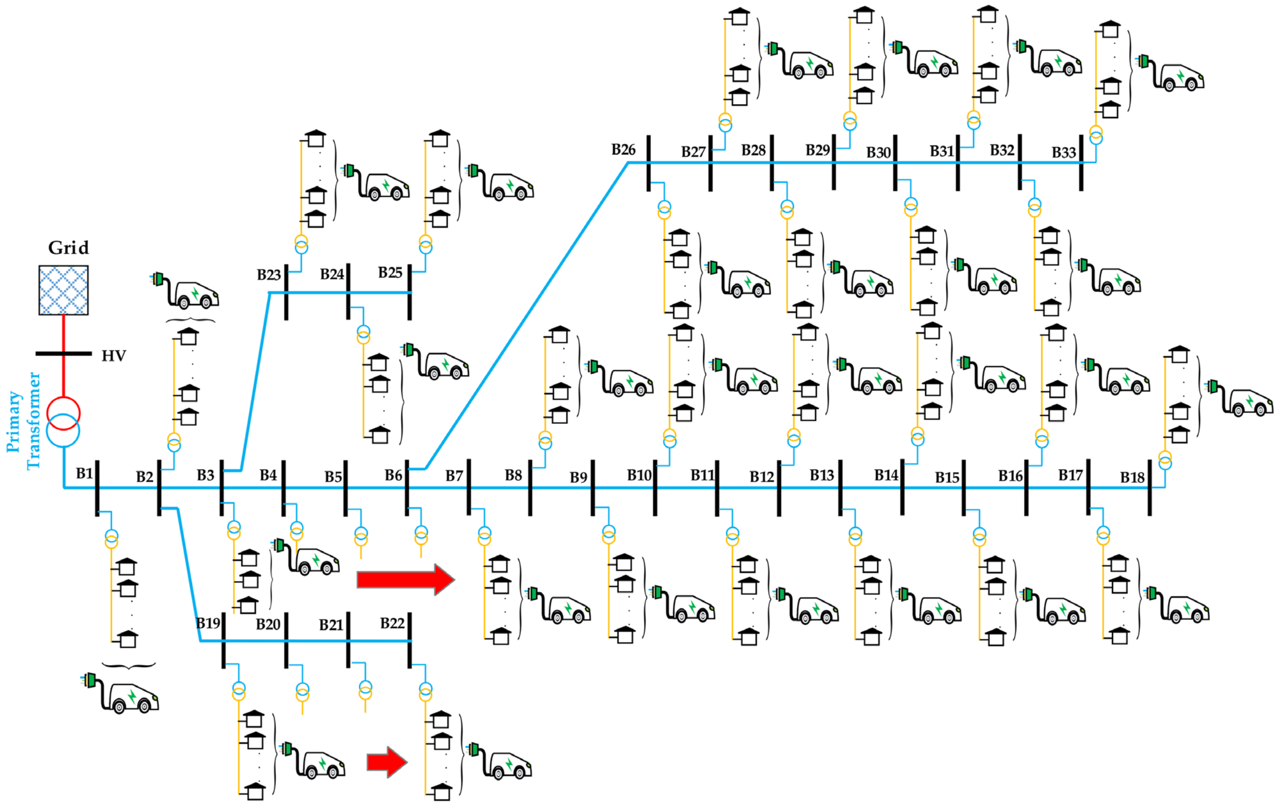
- Power flow Module

  Classes：

        - regulator – Voltage regulator object
        - relay – Relay object
        - series_reactor – Series reactor object
        - switch_object – Generic switch object
            - recloser – Recloser object
            - sectionalizer – Sectionalizer object
        - transformer – Transformer object
    - node – Generic node (bus) object.
        - capacitor – Capacity object
        - load – Generic load object
            - pqload – PQ load object
        - meter – Meter object
        - motor – Motor object
        - substation – Substation object
        - triplex_node – Triplex node object
            - triplex_meter – Triplex meter object
            - volt_var_control – Volt-var controller object
  - voltdump – Volt data dump on node objects at specified times

GridLAB-D Module

- Power flow Module
  - The power flow module performs distribution level solver methods to primarily obtain the voltage and current values in a system.
  - Nearly all objects within the power flow module are derived from two primary classes: node and link.



GridLAB-D Module

- Power flow Module

  Node Object

  - The node object is equivalent to a bus of the distribution system.

```
object node {
        name NodeOne;
        phases ABC;
        nominal_voltage 7200.0;
        voltage_A 7200.0+0d;
        voltage_B 7200.0-120.0d;
        voltage_C 7200.0+120.0d;
        bustype PQ;
        }
```

Bus type	Description
PQ	PQ for a constant power bus (default)
PV	PV for a voltage-controlled (magnitude) bus
SWING	SWING for the infinite bus of a system.

GridLAB-D Module

- Power flow Module

  Load Object

  - Load objects are derived from the node objects
  - Load objects provide a means to implement constant current, constant power, and constant impedance losses or generation into the system.
  - Loads can be a mixture of the constant current, constant impedance, and constant power types. A typical, mixed load would be implemented as

```
object load {
    phases "ABCD";
    name 841;
    constant_current_C -0.586139+9.765222j;
    constant_impedance_B 221.915014+104.430595j;
    constant_power_A 42000.000000+21000.000000j;
    nominal_voltage 4800;
}
```

GridLAB-D Module

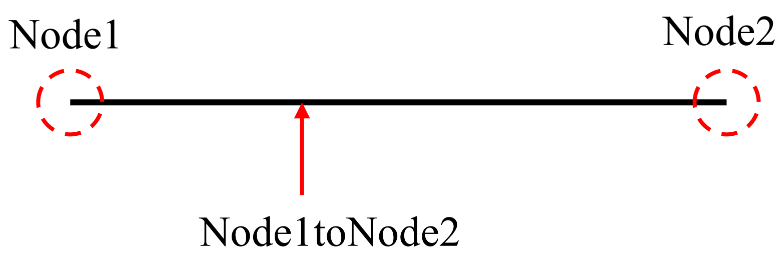
- Power flow Module

  Link Object

  - The link object is a connection between nodes in a distribution system

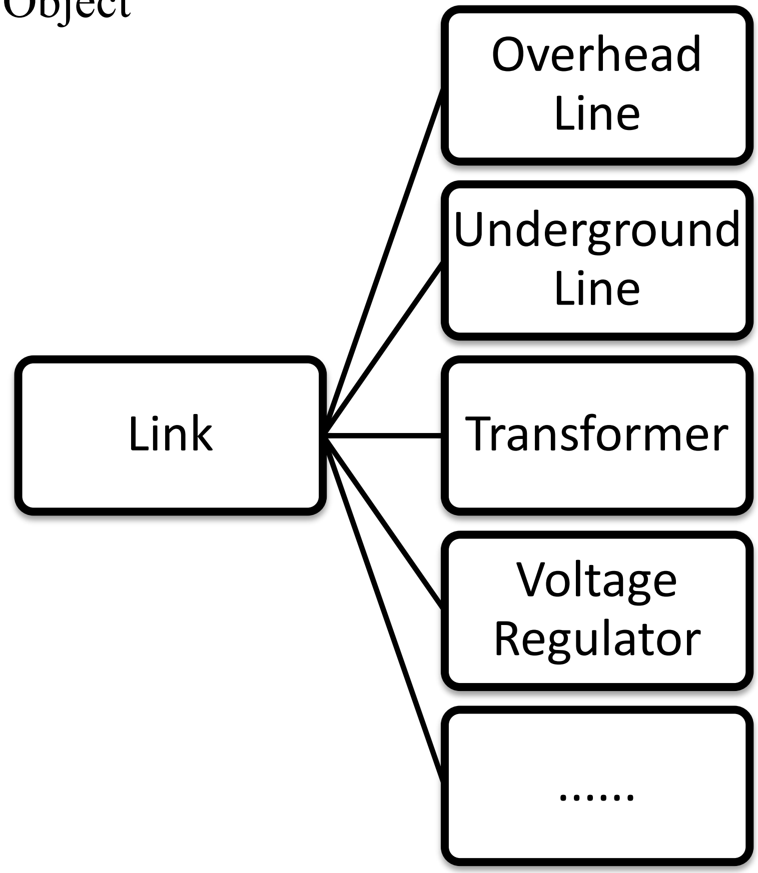
```
object link {
    name Node1toNode2;
    phases ABC;
    from Node1;
    to Node2;
}
```

Property Name	Description
from	One connecting end of the link object. This will be the name or reference to a node-based object elsewhere in the power flow model.
to	The other connecting end of the link object. This will be the name or reference to a node-based object elsewhere in the power flow model.



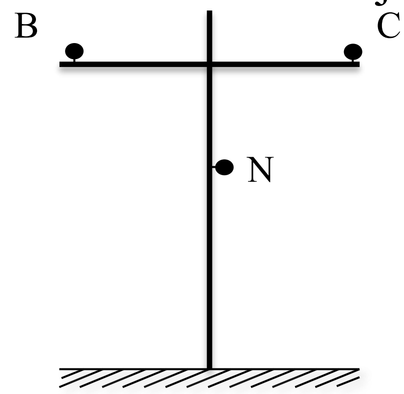
GridLAB-D Module

- Power flow Module
    - Link Object



GridLAB-D Module

- Power flow Module
  - Overhead Line Object



➢ Object Code

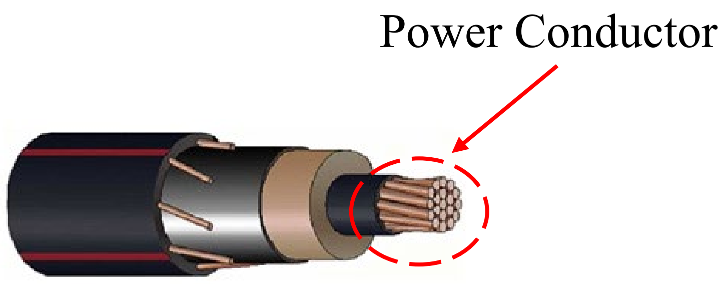
```
object overhead_line {
    phases "BCN";
    name ohl_632-645;
    from node_632;
    to load_645;
    length 500 ft;
    configuration line_configuration_603;
}
```

➢ Configuration Code

```
object line_configuration {
        name line_configuration_603;
        conductor_B overhead_line_conductor_6030;
        conductor_C overhead_line_conductor_6030;
        conductor_N overhead_line_conductor_6030;
        spacing line_spacing_505603;
}
object overhead_line_conductor {
        name overhead_line_conductor_6030;
        geometric_mean_radius 0.004460;
        resistance 1.120000;
}
object line_spacing {
        name line_spacing_505603;
        distance_BC 7.0 ft;
        distance_CN 5.657 ft;
        distance_BN 5.0 ft;
}
```

GridLAB-D Module

- Power flow Module
  - Underground Line Object



Underground Cables

➢ Object Code

```
object underground_line {
    phases "AN";
    name ugl684-652;
    from node_684;
    to load-652;
    length 800;
    configuration line_configuration_607;
}
```

➢ Configuration Code

```
object line_configuration {
        name underground_line_configuration_607;
        conductor_A underground_line_conductor_6070;
        conductor_N underground_line_conductor_6070;
        spacing line_spacing_520;
}
object underground_line_conductor {
        name underground_line_conductor_6070;
        outer_diameter 1.060000;
        conductor_gmr 0.011100;
        conductor_diameter 0.368000;
        conductor_resistance 0.970000;
        neutral_gmr 0.011100;
        neutral_resistance 0.970000;
        neutral_diameter 0.0640837;
        neutral_strands 6.000000;
        shield_gmr 0.000000;
        shield_resistance 0.000000;
}
object line_spacing {
        name line_spacing_520;
        distance_AN 0.0833 ft;
}
```

GridLAB-D Module

- Power flow Module
  - Transformer Object

➢ Object Code

```
object transformer {
        configuration tc100100B;
        name f5_T21400253B;
        from f5_L2691959;
        to f5_X2691959B;
        nominal_voltage 7200;
        phases BS;
}
```

➢ Configuration Code

```
object transformer_configuration {
        name tc100100B;
        connect_type SINGLE_PHASE_CENTER_TAPPED;
        install_type POLETOP;
        primary_voltage 7200.0;
        secondary_voltage 120.0;
        power_rating 100.0;
        powerB_rating 100.0;
        impedance 0.006+0.0136j;
        impedance1 0.012+0.0204j;
        impedance2 0.012+0.0204j;
        shunt_impedance 259200+103680j;
}
```

GridLAB-D Module

- Power flow Module
  - Voltage Regulator Object

➢ Object Code

```
object regulator {
        name reg1501491;
        phases "ABC";
        from node_150;
        to node_1491;
        configuration
regulator_configuration_15;
}
```

➢ Configuration Code

```
object regulator_configuration {
        name regulator_configuration_15;
        connect_type WYE_WYE;
        band_center 120.000;
        band_width 2.0;
        time_delay 30.0;
        raise_taps 16;
        lower_taps 16;
        current_transducer_ratio 700;
        power_transducer_ratio 20;
        compensator_r_setting_A 3.0;
        compensator_x_setting_A 7.5;
        compensator_r_setting_B 3.0;
        compensator_x_setting_B 7.5;
        compensator_r_setting_C 3.0;
        compensator_x_setting_C 7.5;
        CT_phase "ABC";
        PT_phase "ABC";
        regulation 0.10;
        Type A;
        tap_pos_A 7;
        tap_pos_B 7;
        tap_pos_C 7;
        Control MANUAL;  // OUTPUT_VOLTAGE,
                // LINE_DROP_COMP or REMOTE NODE
}
```

GridLAB-D Module

- Power flow Module

  Power flow solution methods

  - Forward/Backward Sweep method
    - Forward/Backward Sweep method is also referred to as a ladder solver.
    - This was one of the first methods developed for unbalanced power flow.
    - For radial systems this is a very efficient algorithm.
    - It is unable to solve on a networked or looped system.
    - Even though a system is radial, there may be components that present a networked appearance.
    - Overall this is a very good solution method, but it has limitations.
  - Newton-Raphson method
    - Newton-Raphson Method was originally developed for use on transmission level analysis.
    - They are based on the power injections at each node, both real and reactive power.
    - These power injections are then used to update the voltage magnitude and voltage angle.
    - The NR method requires the calculation, and inversion of a Jacobian which is computationally intensive.

GridLAB-D Module

- Power flow Module

Power Flow Solution Methods

```
module powerflow {
solver_method FBS;
//In this module, FBS method is selected to calculate power flow.
default_maximum_voltage_error 1e-6;
 //Define convergence criteria
NR_iteration_limit 50;
//Maximum iteration is 50
}
```

- ➢ As implemented, FBS is faster.
- ➢ As implemented, NR handles meshed systems, reconfiguration algorithms, and reliability.
- ➢ Method to be used depends upon application, but both methods currently work with all objects in power flow and all other modules (except reconfiguration and reliability).

Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help
- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape
- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

GridLAB-D Module

- Residential Module

  Common Classes

    - house – Single-family home model.
    - ZIPload – Generic constant impedance/current/power end-use load.

  - As of Hassayampa (Version 3.0)

  These may be available in earlier versions but they have not been validated and are not supported.

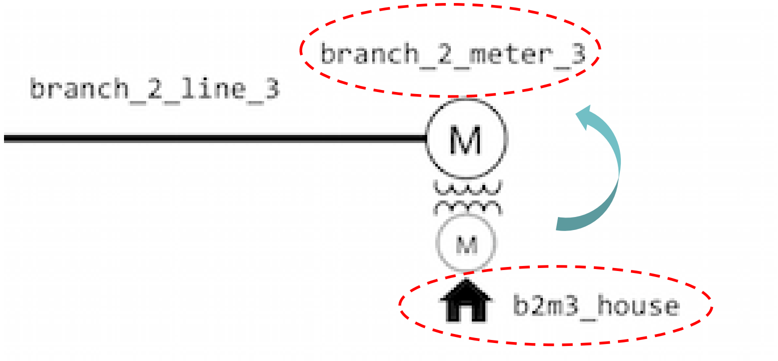
  - lights – Typical residential lights.
  - occupantload – Residential occupants (sensible and latent heat).
  - plugload – Typical residential plug loads.

GridLAB-D Module

- Residential Module

  House Object

  - It attaches to the power system via a meter. (meter is parent of house)
  - Thermal dynamics are expressed by an Equivalent Thermal Parameter (ETP) model.
  - It can simulate heat flow of the house.



➢ Object Code

```
object house {
name house_114A_3; parent triplex_meter_114A_3;
//'parent' means house_114A_3 is connected with meter_114A_3.
air_temperature 72;mass_temperature 72;cooling_COP 3.5;
over_sizing_factor 0.0;ceiling_height 8;number_of_stories 1;
aspect_ratio 1.5;floor_area 864;mass_internal_gain_fraction 0.5;
mass_solar_gain_fraction 0.5;glass_type GLASS;glazing_layers TWO;
airchange_per_hour 1.5;Rroof 19;Rdoors 3;Rfloor 4;Rwall 11;
};
```

GridLAB-D Module

- Residential Module

  House Object

  - The types of equipment that form a residential heating/ventilating/air conditioning (HVAC)system is defined in House object by the input parameters.
  - System_mode: heating/cooling system operation state

system_mode	Description
UNKNOWN	When system_mode is set to UNKNOWN, it will be set to OFF
OFF	Neither the heating, auxiliary heating nor cooling system is currently in operation.
COOL	The cooling system is in operation (if it exists).
HEAT	The heating system is in operation (if it exists).
AUX	The auxiliary heating system is operating (if it exists).
Default: If system_mode is not set, it is assumed that system_mode is OFF.	

GridLAB-D Module

- Residential Module

  House Object

  - heating_system_type: Type of heating system in the house

heating_system_type	Description
NONE	No heating system exists.
HEAT_PUMP	Specifies that heat is provided by heat pump that is a reversible vapor compression cycle
RESISTANCE	Specifies that heat is provided by an electric resistance heating coil
GAS	Specifies that heat is provided by a gas-powered furnace or boiler
Default: When heating_system_type is not set, it is assumed that the heating system of the house is HEAT_PUMP.	

GridLAB-D Module

- Residential Module
  House Object
  - cooling_system_type: Cooling system in house

cooling_system_type	Description
NONE	No air-conditioning provided.
ELECTRIC	Forced-air vapor-compression cooling, representing either a central air conditioner or a window/wall unit that cycles on and off to maintain air_temperature below the cooling thermostat set point.
HEAT_PUMP	Not used
Deafult: If no cooling_system_type is set, it is assumed to be NONE.	

GridLAB-D Module

- Residential Module

  House Object

  ➢ auxiliary_system_type：specify auxiliary heat for heat pump heating_system_type.

cooling_system_type	Description
NONE	No air-conditioning provided.
ELECTRIC	Forced-air vapor-compression cooling, representing either a central air conditioner or a window/wall unit that cycles on and off to maintain air_temperature below the cooling thermostat set point.
HEAT_PUMP	Not used

GridLAB-D Module

- Residential Module
  House Object
  - Primary Inputs

Parameter (symbol; selections)	Default Value	
Heat system type (gas, heat pump, resistance, none)	heat pump	-
Cool system type (electric, none)	none	-
Cooling COP, standard conditions[e]	3.50	-
Heating COP, standard conditions[a]	3.50	-
Latent cooling, fraction, of sensible cooling	35%	-
Thermostat set point, heat (Tset_heat; value, or a schedule)[b]	70	°F
Thermostat set point, cool (Tset_cool; value, or a schedule)[c]	75	°F
Thermostat deadband (dTdeadband)[d]	2.0	°F
Thermostat cycle time, minimum (tmin)	2.0	min

GridLAB-D Module

- Residential Module
  House Object
  - Primary Inputs

Auxiliary heat (electric, none)[e]	electric	-
Auxiliary heat deadband (dTaux; value, none)[e,f,g]	2.0	°F
Auxiliary heat outdoor lockout temperature (Taux; value, none)[e,h]	none	°F
Auxiliary heat time delay (taux)[e,h]	none	min
Fan type (1-speed, 2-speed, none)[h,i]	1-speed	-
Fan power, low-speed, fraction of hi-speed[j]	%	-
Heating COP curve (default, flat, linear, curved)	default	-
Cooling COP curve (default, flat, linear, curved)	default	-
Heating capacity curve (default, flat, linear, curved)	default	-
Cooling capacity curve (default, flat, linear, curved)	default	-
Use latent heat (true, false)	true	-
Include fan heat gain (true, false)	true	-

GridLAB-D Module

- Residential Module
  - ZIP load Object

➢ It contains a classic ZIP load model (constant impedance, current, and power)

$$P_i = \frac{|V_a^2|}{|V_n^2|} * |S_n| * Z\% * \cos(Z_\theta) + \frac{|V_a|}{|V_n|} * |S_n| * I\% * \cos(I_\theta) + |S_n| * P_\% * \cos(P_\theta)$$

$$Q_i = \frac{|V_a^2|}{|V_n^2|} * |S_n| * Z\% * \cos(Z_\theta) + \frac{|V_a|}{|V_n|} * |S_n| * I\% * \cos(I_\theta) + |S_n| * P_\% * \cos(P_\theta)$$

Where:
Pi: Real power consumption of ith load; *Qi*: Reactive power consumption of ith load.
Va: Actual terminal voltage; *Vn*: Nominal terminal voltage
Sn: Apparent Power Consumption at nominal voltage; *Z%*: Percent of load that is constant impedance
I%:Percent of load that is constant current; *P%*:Percent of load that is constant power
Z_θ:Phase angle of constant impedance fraction; I_θ:Phase angle of constant current fraction.
P_θ :Phase angle of constant power fraction.

```
object ZIPload {
        name house1_load; parent house1;
        base_power responsive_loads*1.06; heatgain_fraction 0.90;
        power_pf 1.0;current_pf 1.0;impedance_pf 1.0;
        impedance_fraction 0.20;current_fraction 0.40;power_fraction 0.40;
   };
```

Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help
- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape
- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

GridLAB-D Module

- Generator Module

  Classes

  - battery – battery model
  - diesel_dg –
    - PQ_CONSTANT mode diesel_dg – The diesel_dg with predefined P and Q output
  - dc_dc_converter –
  - energy_storage –
  - inverter – converts DC (solar, battery, etc.) to AC
    - Droop inverter – The inverter with droop mode enabled
    - Voltage source inverter (VSI) – The isochronous mode and droop mode VSI
  - microturbine –
  - power_electronics –
  - rectifier –
  - solar – solar panel
  - windturb_dg –

GridLAB-D Module

- Generator Module

Designed to model distributed energy resource (DER). Sometimes also referred as distributed generation.

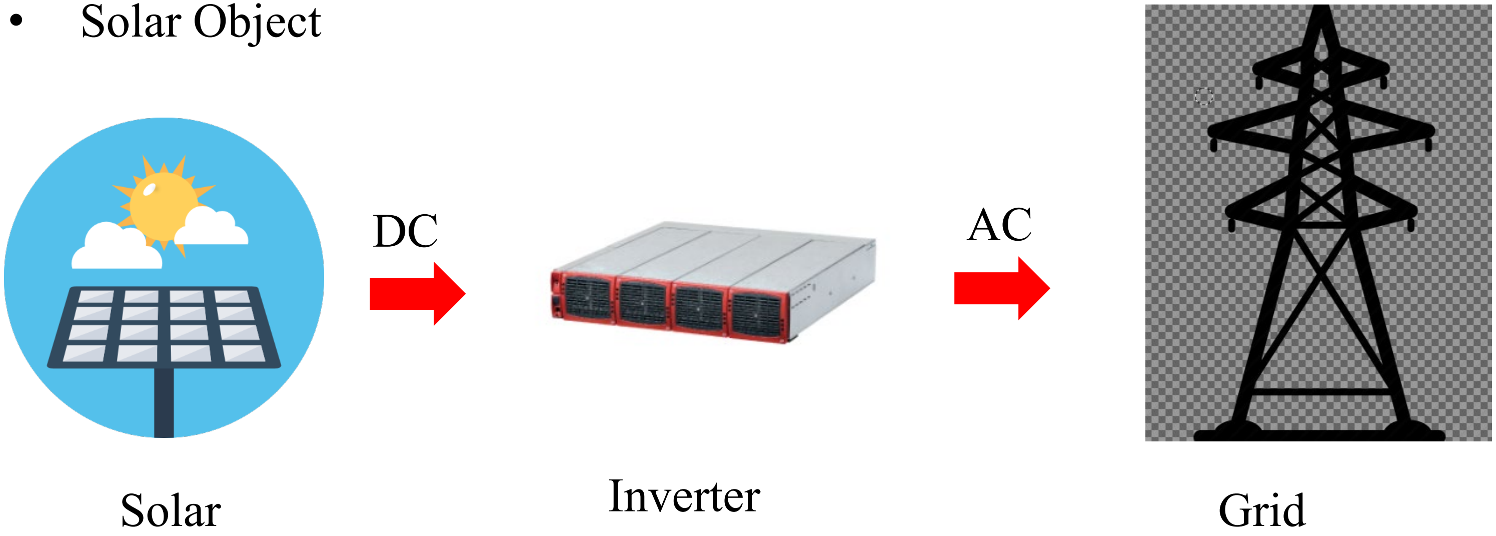
Photovoltaic

Wind

Battery

GridLAB-D Module

- Generator Module
  - Solar Object



- Solar object must have inverter as its parent.
- DC networks are not currently supported
- Power flow sees it as a current injection and negative load

GridLAB-D Module

- Generator Module
  - Solar Object

➢ Fixed axis PV object implemented in version 2.3

➢ Needed for most of the residential PV simulation or analysis

➢ User can specify the tilt needed or select a default which is often the latitude of the region selected

```
object solar {
        name trip_solar;
        phases AS;
        parent trip_inv;
        area 29.6296 m^2;
        tilt_angle 50.0;
        efficiency 0.135;
        orientation_azimuth 25.0;
        orientation FIXED_AXIS;
        };
```

GridLAB-D Module

- Generator Module
  - Inverter Object

➢ It converts DC power to AC power

➢ All the dc power generating distributed resources should have inverter as its parent object.

➢ Constant PF mode allows the user to set any power factor to regulate the reactive power within the specified inverter rating.

```
object inverter {
   name inv_B_2_645;
   phases BS;
   generator_mode CONSTANT_PF;
   generator_status ONLINE;
   inverter_type PWM;
   power_factor 1.0;
  };
```

GridLAB-D Module

- Generator Module
  - Inverter Object

Control mode	Description
CONSTANT_PQ	Sets the output of the battery to the specified constant real and reactive power levels.
CONSTANT_PF	This is the default controller mode with the power factor set to unity.
VOLT_VAR	Uses the reactive power capability of the inverter to regulate the voltage at the point of common coupling.
LOAD_FOLLOWING	Allows the inverter to use the energy from the battery to regulate the load at a user-definable point on the feeder.

GridLAB-D Module

- Generator Module
  - Solar-Inverter System Implementation

```
object inverter {
    name inv_B_2_645;
    phases BS;
    generator_mode CONSTANT_PF;
    generator_status ONLINE;
    inverter_type PWM;
    power_factor 1.0;
object solar {
        name sol_inv_B_2_645;
        generator_mode SUPPLY_DRIVEN;
        generator_status ONLINE;
        panel_type SINGLE_CRYSTAL_SILICON;
        efficiency 0.2;
        area 450;
            };
    }
```

Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help

- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape

- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

GridLAB-D Module

- Tape Module

  Classes
  - player – Play data into the model
  - shaper – Generate pulsed or modulated data from averages
  - recorder – Record data to a stream
    - multi_recorder – Record properties from multiple objects
    - group_recorder – Records properties of objects designated by class type and group id
    - violation_recorder – Records voltage and thermal limit violations as well as reverse flow through a substation
  - collector – Data aggregation recording
  - histogram – Property statistics

- The tape module implements objects that can be used to establish and change the boundary condition on a model, and observes the properties of individual objects or the aggregate properties of a group of objects.
- **Player** and **shaper** tapes are used for updating the model at specified times from a file.
- **Recorder** and **collector** tapes are used for collecting information from the model.

GridLAB-D Module

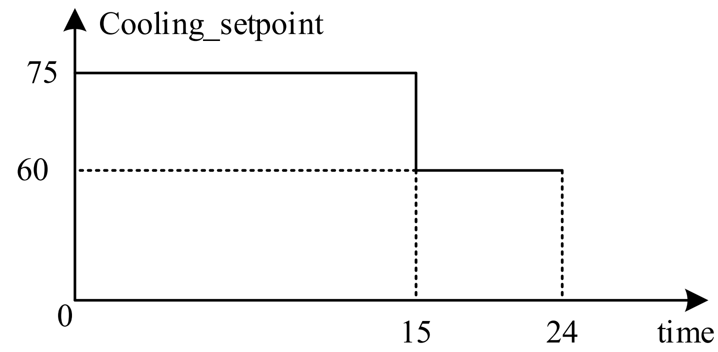
- Tape Module
  - Player Object

```
object player {
  name player-name;
  parent target-object-name;
  property target-property-name;
  file output-file-name;
}
```

```
object player {
    name player;
    parent house1;
    property cooling_setpoint;
    file “theat,csv”;
}
```

theat.csv:

```
2019-01-01 0:00:00,75
+15h,60
+9h,75
```



From 0:00 to15： 00 the cooling set point of house1 is 75F,

From 15:00 to 24:00 the cooling set point of house1 is 60F.

GridLAB-D Module

- Tape Module
  - Recorder Object

```
object recorder {
  name "recorder-name";
  parent "target-object-name";
  property "target-property-name";
  file "output-file-name";
  interval sampling-interval;
  limit sampling-limit;
}
```

```
object recorder {
  name recorder1;
  parent house1;
  property air_temperature, cooling_setpoint;
  file theat_record.csv;
  interval 1800;
  limit 48;
}
```

- Parent house1 means this recorder records data or property from house1.
- The air temperature and cooling set point are recorded in file theat_record.csv .
- Interval 4800 means the frequency is 4800 seconds each time.
- Limit 48 means the record times is 48.

GridLAB-D Module

- Recorder Object

properties

interval : 30min=1800s

# timestamp	air_temperature	cooling_setpoint
2001-01-01 00:00:00 PST	60.1805	60
2001-01-01 00:30:00 PST	61.8869	60
2001-01-01 01:00:00 PST	57.2472	60
2001-01-01 01:30:00 PST	60.7869	60
2001-01-01 02:00:00 PST	61.7786	60
2001-01-01 02:30:00 PST	57.0379	60
2001-01-01 03:00:00 PST	60.5776	60
2001-01-01 03:30:00 PST	61.5343	60
2001-01-01 04:00:00 PST	56.8017	60
2001-01-01 04:30:00 PST	60.3734	60
2001-01-01 05:00:00 PST	61.3693	60
2001-01-01 05:30:00 PST	56.8531	60
2001-01-01 06:00:00 PST	60.5573	60
2001-01-01 06:30:00 PST	62.1531	60
2001-01-01 07:00:00 PST	57.9207	60
2001-01-01 07:30:00 PST	62.4451	60
2001-01-01 08:00:00 PST	58.6043	60
2001-01-01 08:30:00 PST	62.8697	60
2001-01-01 09:00:00 PST	58.9204	60
2001-01-01 09:30:00 PST	63.035	60
2001-01-01 10:00:00 PST	58.9531	60
2001-01-01 10:30:00 PST	62.9797	60
2001-01-01 11:00:00 PST	58.8059	60

GridLAB-D Module

- Tape Module
  - Collector Object
  - Collectors are different from recorders in that they aggregate multiple object properties into a single value.
  - They do not use the parent property but instead use the group property to form a collection of objects over which the aggregate is taken.

```
object collector {
  name collector-name;
  group class=class-name;
  property aggregator(property)[,...];
  file "file-name";
}
```

➡

```
object collector {
        file exercise_2_4_2.csv;
        group "class=house";
        property “avg(air_temperature)";
        interval 3600;
}
```

➢ Collect the hourly average indoor air temperature a population of houses.

Outline

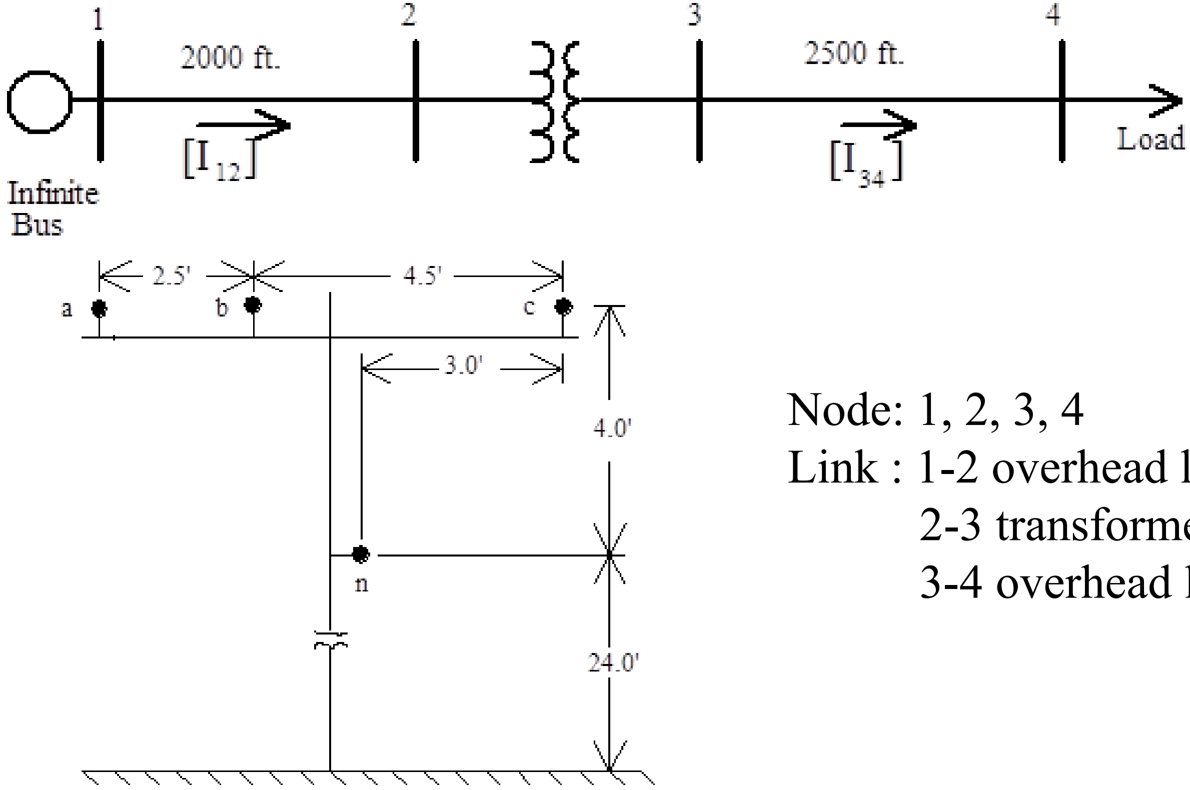
- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help

- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape

- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

Test System Modeling and Simulation

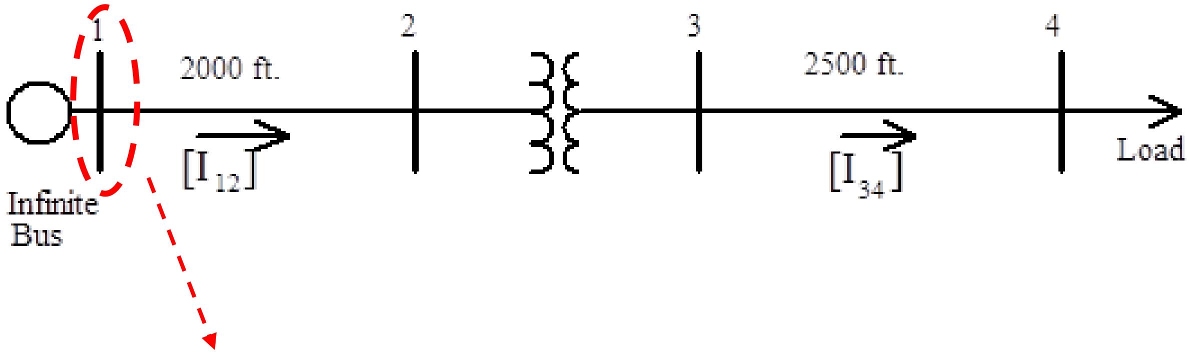
- IEEE 4 Node Test Feeder



Node: 1, 2, 3, 4
Link : 1-2 overhead line12
2-3 transformer23
3-4 overhead line34

Test System Modeling and Simulation

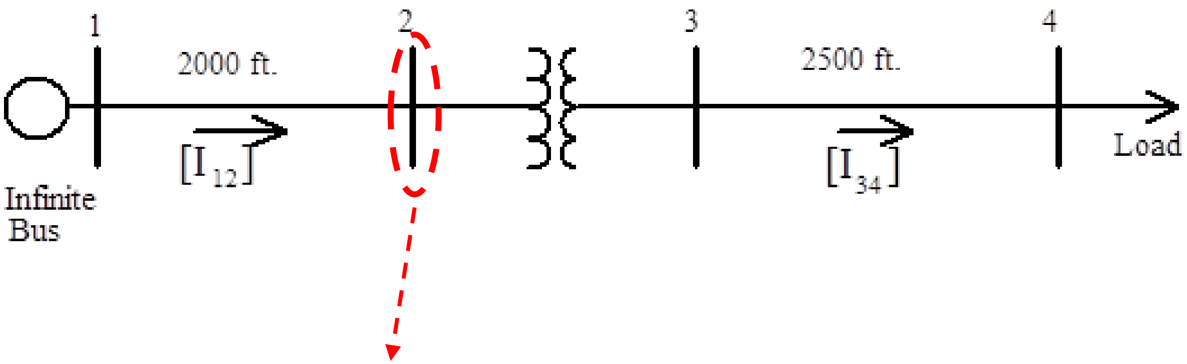
- IEEE 4 Node Test Feeder




```
object node {
        name node1;
        bustype SWING;
        phases "ABCN";
        voltage_A +7199.558+0.000j;
        voltage_B -3599.779-6235.000j;
        voltage_C -3599.779+6235.000j;
        nominal_voltage 7200;
}
```

Test System Modeling and Simulation

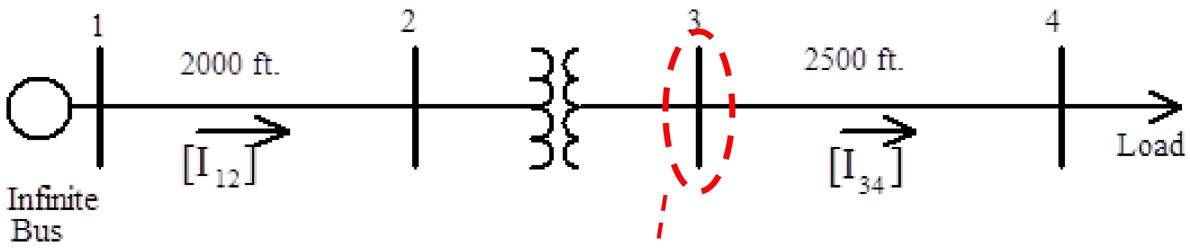
- IEEE 4 Node Test Feeder




```
object node {
        name node2;
        phases "ABCN";
        voltage_A +7199.558+0.000j;
        voltage_B -3599.779-6235.000j;
        voltage_C -3599.779+6235.000j;
        nominal_voltage 7200;
}
```

Test System Modeling and Simulation

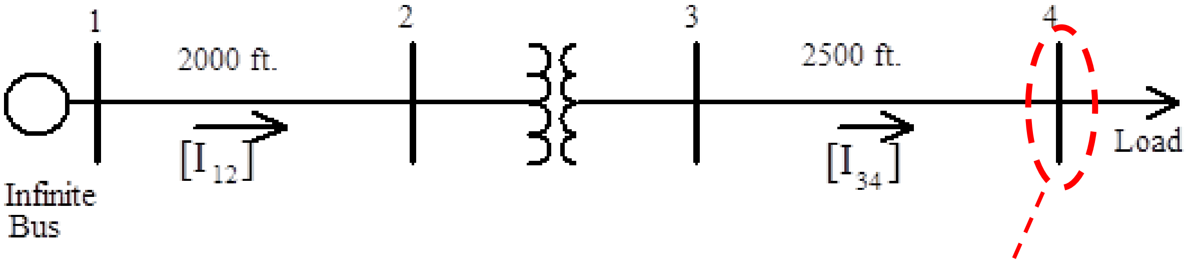
- IEEE 4 Node Test Feeder




```
object node {
        name node3;
        phases "ABCN";
        voltage_A +2401.777+0.000j;
        voltage_B -1200.889-2080.000j;
        voltage_C -1200.889+2080.000j;
        nominal_voltage 2400;
}
```

Test System Modeling and Simulation

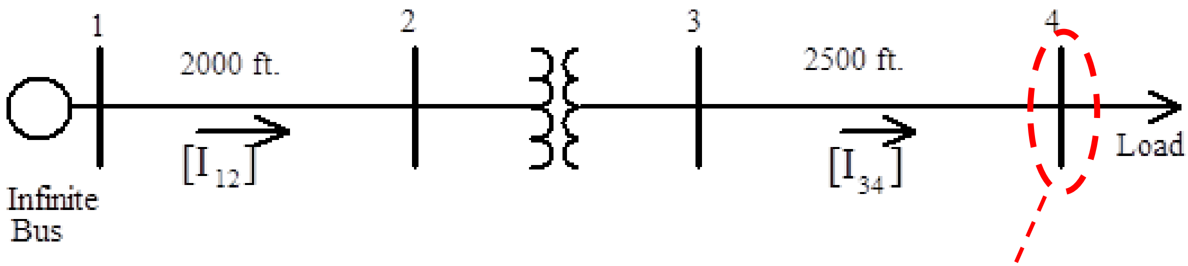
- IEEE 4 Node Test Feeder




```
object load {
        name load4;
        phases "ABCN";
        voltage_A +2401.777+0.000j;
        voltage_B -1200.889-2080.000j;
        voltage_C -1200.889+2080.000j;
        constant_power_A +1800000.000+871779.789j;
        constant_power_B +1800000.000+871779.789j;
        constant_power_C +1800000.000+871779.789j;
        nominal_voltage 2400;
}
```

Test System Modeling and Simulation

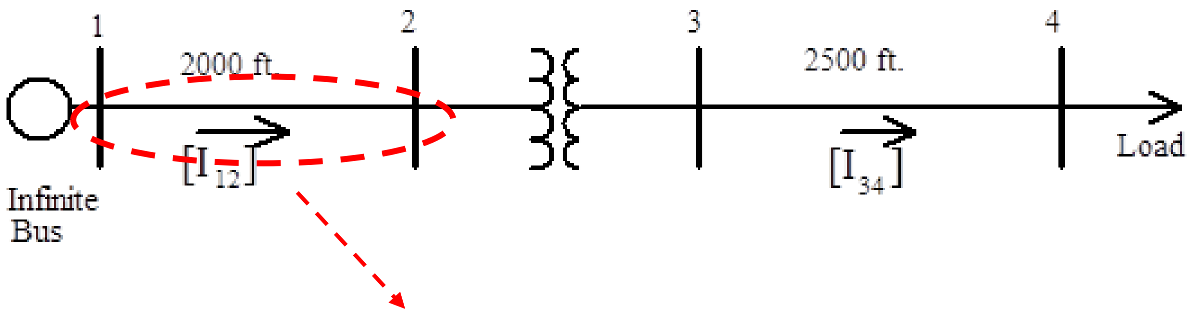
- IEEE 4 Node Test Feeder




```
object load {
        name load4;
        phases "ABCN";
        voltage_A +2401.777+0.000j;
        voltage_B -1200.889-2080.000j;
        voltage_C -1200.889+2080.000j;
        constant_power_A +1800000.000+871779.789j;
        constant_power_B +1800000.000+871779.789j;
        constant_power_C +1800000.000+871779.789j;
        nominal_voltage 2400;
}
```

Test System Modeling and Simulation

- IEEE 4 Node Test Feeder



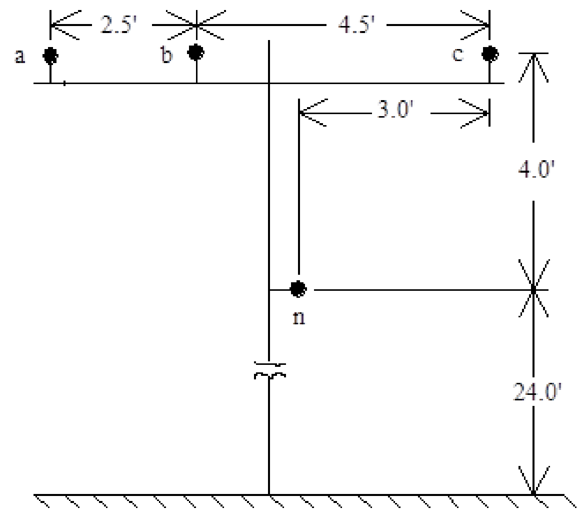

```
object overhead_line {
		name ovl_12;
		phases "ABCN";
		from node1;
		to node2;
		length 2000;
		configuration line_configuration300;
}
```

In line_configuration object define the parameters and spacing distribution of line

Test System Modeling and Simulation

- IEEE 4 Node Test Feeder

```
object line_configuration {
        name line_configuration300;
        conductor_A overhead_line_conductor100;
        conductor_B overhead_line_conductor100;
        conductor_C overhead_line_conductor100;
        conductor_N overhead_line_conductor101;
        spacing line_spacing200;
}
object line_spacing {
        name line_spacing200;
        distance_AB 2.5;
        distance_BC 4.5;
        distance_AC 7.0;
        distance_AN 5.656854;
        distance_BN 4.272002;
        distance_CN 5.0;
}
```



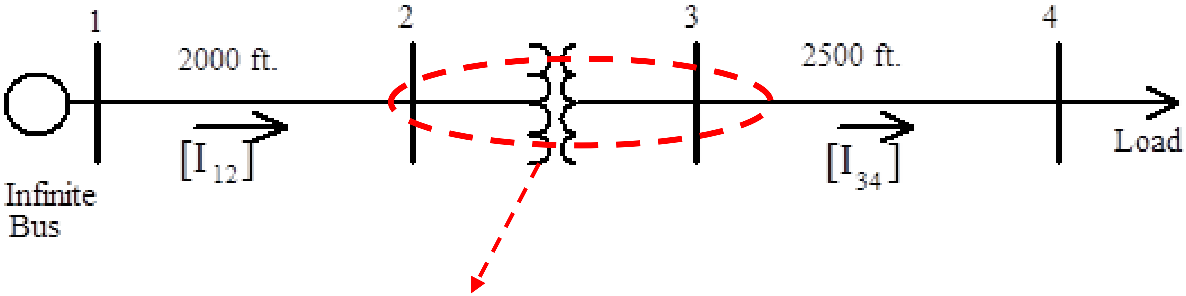
Test System Modeling and Simulation

- IEEE 4 Node Test Feeder

```
object line_configuration {
        name line_configuration300;
        conductor_A overhead_line_conductor100;
        conductor_B overhead_line_conductor100;
        conductor_C overhead_line_conductor100;
        conductor_N overhead_line_conductor101;
        spacing line_spacing200;
}
object overhead_line_conductor {
        name overhead_line_conductor101;
        geometric_mean_radius 0.00814;
        resistance 0.592;
}
object overhead_line_conductor {
        name overhead_line_conductor101;
        geometric_mean_radius 0.00814;
        resistance 0.592;
}
```

Test System Modeling and Simulation

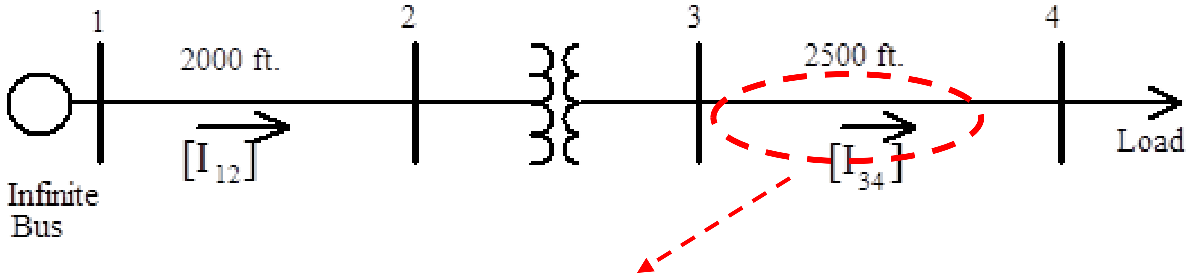
- IEEE 4 Node Test Feeder




```
object transformer_configuration {
        name transformer_configuration400;
        connect_type 1;
        power_rating 6000;
        powerA_rating 2000;
        powerB_rating 2000;
        powerC_rating 2000;
        primary_voltage 12470;
        secondary_voltage 4160;
        resistance 0.01;
        reactance 0.06;
}
```

Test System Modeling and Simulation

- IEEE 4 Node Test Feeder




```
object overhead_line {
        name ovl_34;
        phases "ABCN";
        from node3;
        to load4;
        length 2500;
        configuration line_configuration300;
}
```

Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help

- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape

- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

Test System Modeling and Simulation

- Code

```
clock {
        timestamp '2000-01-01 0:00:00';
        timezone EST+5EDT;
}
module powerflow {
        solver_method NR;
}
object overhead_line_conductor {
        name overhead_line_conductor100;
        geometric_mean_radius 0.0244;
        resistance 0.306;
}
object overhead_line_conductor {
        name overhead_line_conductor101;
        geometric_mean_radius 0.00814;
        resistance 0.592;
}
```

Simulation time
Snapshot power flow

Power flow method,
NR method is selected.

Overhead line parameters

Test System Modeling and Simulation

- Code

```
object line_spacing {
        name line_spacing200;
        distance_AB 2.5;
        distance_BC 4.5;
        distance_AC 7.0;
        distance_AN 5.656854;
        distance_BN 4.272002;
        distance_CN 5.0;
}
```

Overhead line spacing distribution

```
object line_configuration {
        name line_configuration300;
        conductor_A overhead_line_conductor100;
        conductor_B overhead_line_conductor100;
        conductor_C overhead_line_conductor100;
        conductor_N overhead_line_conductor101;
        spacing line_spacing200;
}
```

Overhead line configuration

Test System Modeling and Simulation

- Code

```
object transformer_configuration {
        name transformer_configuration400;
        connect_type 1;
        power_rating 6000;
        powerA_rating 2000;
        powerB_rating 2000;
        powerC_rating 2000;
        primary_voltage 12470;
        secondary_voltage 4160;
        resistance 0.01;
        reactance 0.06;
}
object node {
        name node1;
        bustype SWING;
        phases "ABCN";
        voltage_A +7199.558+0.000j;
        voltage_B -3599.779-6235.000j;
        voltage_C -3599.779+6235.000j;
        nominal_voltage 7200;
}
```

Transformer parameters

Node1

Test System Modeling and Simulation

- Code

```
object transformer_configuration {
        name transformer_configuration400;
        connect_type 1;
        power_rating 6000;
        powerA_rating 2000;
        powerB_rating 2000;
        powerC_rating 2000;
        primary_voltage 12470;
        secondary_voltage 4160;
        resistance 0.01;
        reactance 0.06;
}
object node {
        name node1;
        bustype SWING;
        phases "ABCN";
        voltage_A +7199.558+0.000j;
        voltage_B -3599.779-6235.000j;
        voltage_C -3599.779+6235.000j;
        nominal_voltage 7200;
}
```

Transformer parameters

Node1

Test System Modeling and Simulation

- Code

```
object overhead_line {
        phases "ABCN";
        from node1;
        to node2;
        length 2000;
        configuration line_configuration300;
}

object node {
        name node2;
        phases "ABCN";
        voltage_A +7199.558+0.000j;
        voltage_B -3599.779-6235.000j;
        voltage_C -3599.779+6235.000j;
        nominal_voltage 7200;
}

object transformer {
        name transformer23;
        phases "ABCN";
        from node2;
        to node3;
        configuration transformer_configuration400;
}

object node {
        name node3;
        phases "ABCN";
        voltage_A +2401.777+0.000j;
        voltage_B -1200.889-2080.000j;
        voltage_C -1200.889+2080.000j;
        nominal_voltage 2400;
}
```

Test System Modeling and Simulation

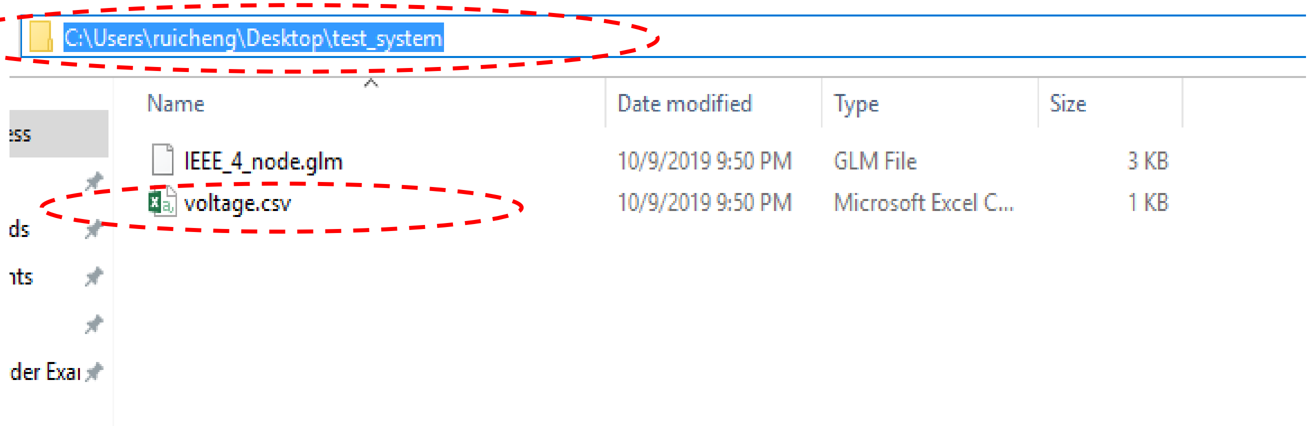
- Code

```
object overhead_line:34 {
        phases "ABCN";
        from node3;
        to load4;
        length 2500;
        configuration line_configuration300;
}

object load {
        name load4;
        phases "ABCN";
        voltage_A +2401.777+0.000j;
        voltage_B -1200.889-2080.000j;
        voltage_C -1200.889+2080.000j;
        constant_power_A +1800000.000+871779.789j;
        constant_power_B +1800000.000+871779.789j;
        constant_power_C +1800000.000+871779.789j;
        nominal_voltage 2400;
}
```

Test System Modeling and Simulation

- Code
  - The code is saved in IEEE_4_node.glm, which is in the directory: C:\Users\ruicheng\Desktop\test_system (in my computer).
  - The voltages of all nodes are recorded in the file named voltage.csv by using recorder object.

```
object multi_recorder {
    property node1:voltage_A,node1:voltage_B,node1:voltage_C,
    node2:voltage_A,node2:voltage_B,node2:voltage_C,
    node3:voltage_A,node3:voltage_B,node3:voltage_C,
    load4:voltage_A,load4:voltage_B,load4:voltage_C;
    file "voltage.csv";
}
```



Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help

- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape

- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

Test System Modeling and Simulation

- Operation Procedure
  - Open CMD, and change the directory to: C:\Users\ruicheng\Desktop\test_system
  - Input the command: “gridlabd IEEE_4_node.glm” in CMD.

C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\ruicheng\Desktop\test_system>
C:\Users\ruicheng\Desktop\test_system>gridlabd IEEE_4_node.glm
WARNING  [2000-01-01 03:00:00 EST] : transformer:transformer23 is at 113.71% of its rated power value
WARNING  [2000-01-01 03:00:00 EST] : Line:ovl_34 is at 104.29% of its continuous rating on phase A!
WARNING  [2000-01-01 03:00:00 EST] : Line:ovl_34 is at 100.97% of its continuous rating on phase C!

C:\Users\ruicheng\Desktop\test_system>
```

Outline

- What is GridLAB-D
  - Introduction to GirdLAB-D
  - GridLAB-D Key Attributes
  - Installing GridLAB-D
  - Getting Help

- GridLAB-D Module
  - Power flow
  - Residential
  - Generator
  - Tape

- Test System Modeling and Simulation
  - IEEE 4 Node Test Feeder
  - Code
  - Run snapshot power flow
  - Result

Test System Modeling and Simulation

- Result

node1:voltage_A	node1:voltage_B	node1:voltage_C
+7199.56+0j	-3599.78-6235j	-3599.78+6235j
node2:voltage_A	node2:voltage_B	node2:voltage_C
+7106.42-42.0676j	-3606.9-6161.63j	-3520.34+6189.71j
node3:voltage_A	node3:voltage_B	node3:voltage_C
+2242.74-144.808j	-1251.27-1892.2j	-1002.84+2020.69j
load4:voltage_A	load4:voltage_B	load4:voltage_C
+1893.76-302.435j	-1277.97-1617.28j	-705.201+1850.98j