Co-optimizing Integrated Transmission-Distribution Systems using PowerModelsITD.jl

Juan Ospina, *Ph.D.*
Scientist
A-1 Information Systems and Modeling Group
Los Alamos National Laboratory (LANL)

LA-UR-23-26135

Acknowledgements - Team & Funding

- David M. Fobes (A-1 LANL)
- Russell Bent (T-5 LANL)
- Andreas Wächter (Northwestern University)
- Xinyi Luo (Northwestern University)

This work was performed with the support of the **U.S. Department of Energy (DOE) Office of Electricity (OE) Advanced Grid Modeling (AGM)** Research Program under program manager **Ali Ghassemian**. We gratefully acknowledge Ali's support of this work.

Outline

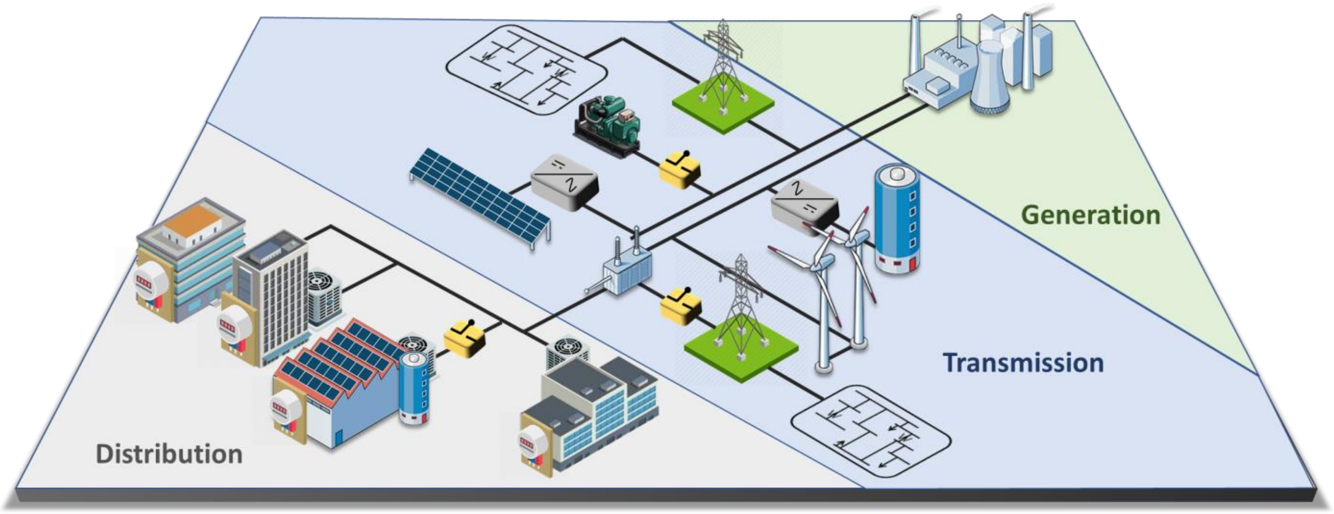
- Background & Challenges
- Introduction to **PowerModelsITD.jl**
- Using **PowerModelsITD.jl**
- Use Cases & Tests

Background & Challenges

Background

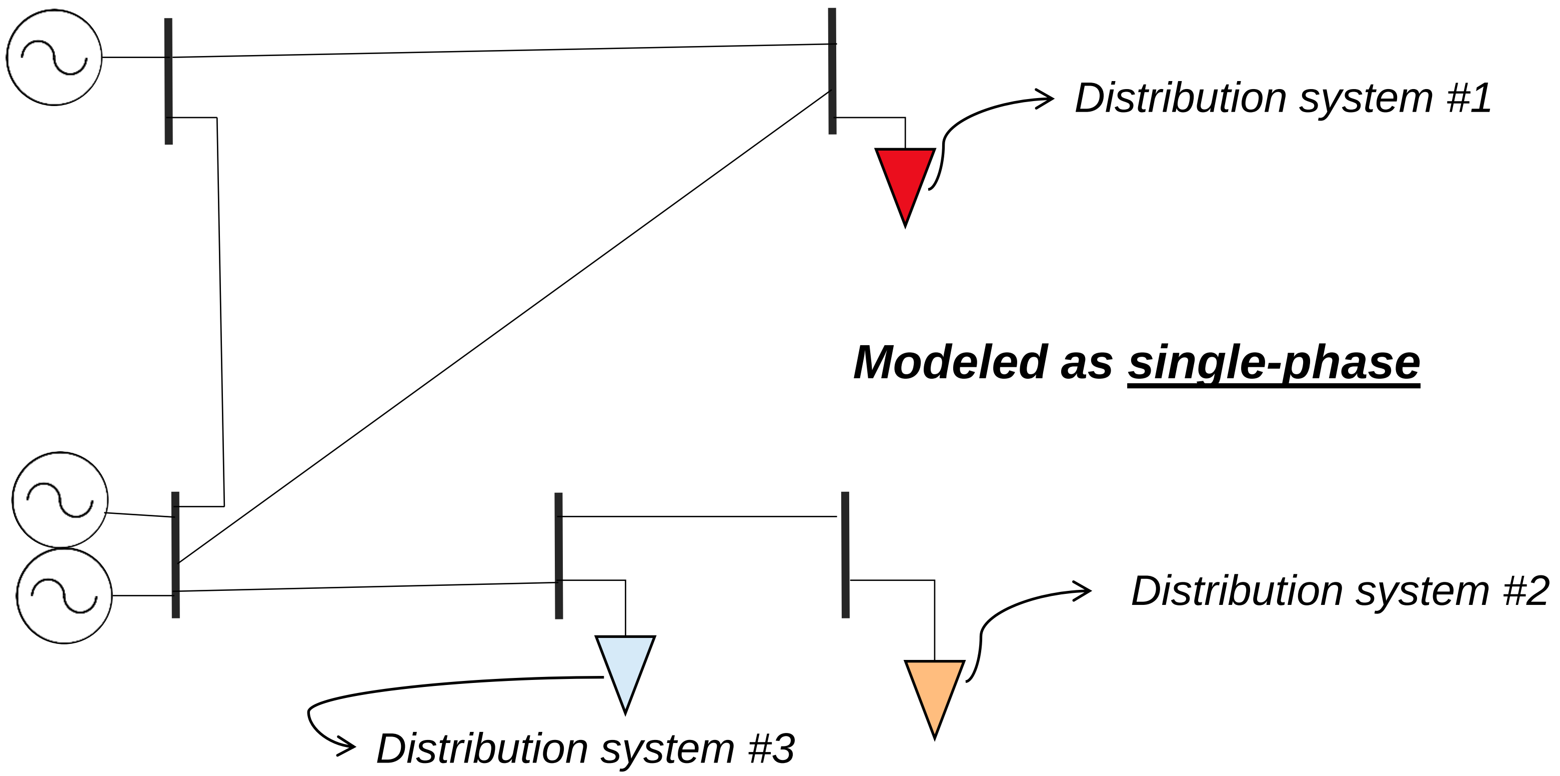
- Conventional electric power systems (EPS) are composed of:
  - **Generation**
  - **Transmission**
  - **Distribution**

- Managed independently by:
  - Transmission system (TSOs)
  - Distribution system operators (DSOs).



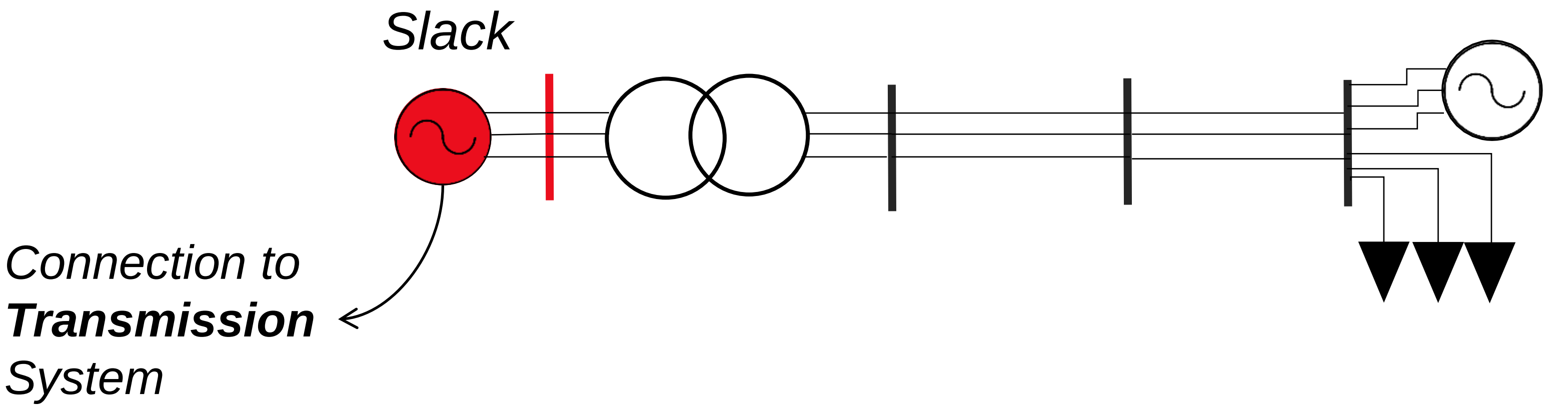
Background: TSOs

- TSOs traditionally model distribution systems as consumers (**loads**).



Background: DSOs

- DSOs traditionally regard transmission systems as slack buses with unlimited resources (often modeled as **voltage sources**).

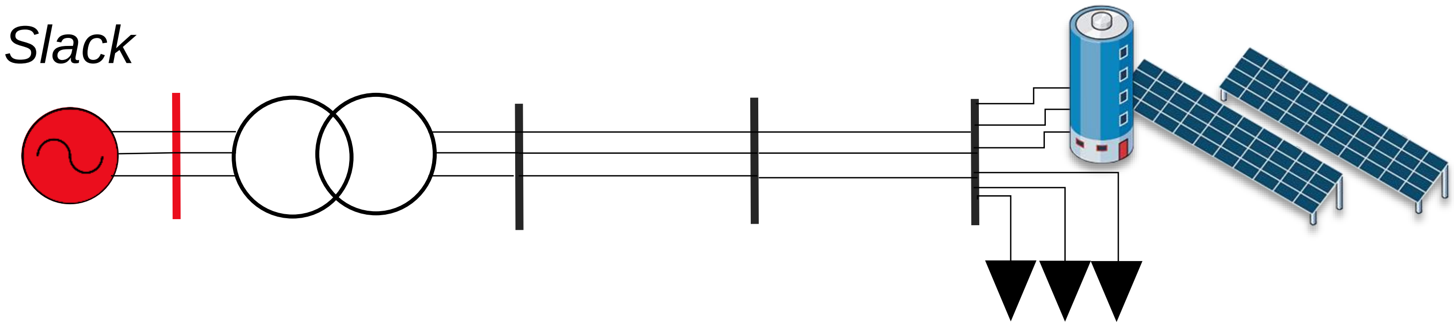


Modeled as <u>three-phase</u> (multiconductor)

Background: Integration of DERs

Distribution systems are becoming more **active**:

- Integration of **Distributed Energy Resources** (DERs)
- Integration of **Information & Communication** Technologies (ICTs).



The **assumption** of distribution being just **passive loads** is **unreasonable** for **optimal** T&D **operation**.

Challenges

- Traditionally owned and operated by **separate entities**.
- **Centralized models** may not be scalable and hard to solve. (*Assumption*)
- **Convergence issues with AC OPF (nonlinear, nonconvex formulations)**
- **Unable** to coordinate or **co-optimize** resources across T&D boundaries

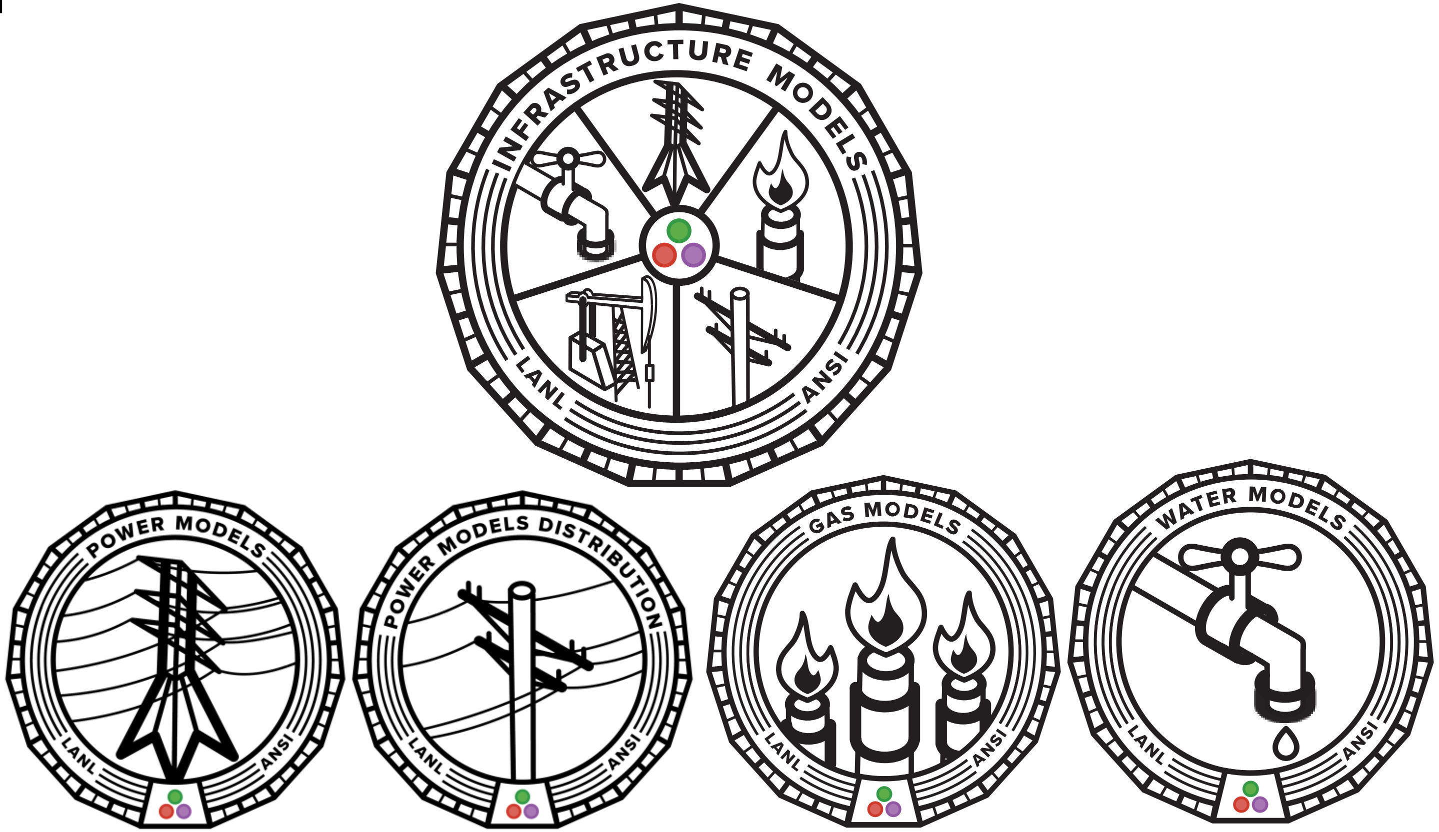
Coordination (Co-optimization) between **T&D** networks will be **imperative** for the **optimal operation** of the (future) power grid.

To **fill this gap**, we developed a **first-of-its-kind tool** that **supports** and **enables** the **Co-optimization** of **T&D systems**

Introduction to PowerModelsITD.jl

InfrastructureModels.jl

- **Core** package for **multi-infrastructure modeling** and **optimization** ecosystem



https://github.com/lanl-ansi/InfrastructureModels.jl

PowerModelsITD.jl (PMITD)

- Open-source tool (Written in **Julia**)
- Based on **LANL multi-infrastructure ecosystem**
- Used for **modeling** and **optimizing T&D systems**
- Solve steady-state **ITD Optimal Power Flow (OPF)**
- Evaluate diverse **network formulations**
- Common research platform for **emerging formulations**

GitHub

[1] https://github.com/lanl-ansi/PowerModelsITD.jl

[2] Ospina, J., et al. (2023). Modeling and Rapid Prototyping of Integrated Transmission-Distribution OPF Formulations with PowerModelsITD.jl. IEEE Transactions on Power Systems.

[3] Ospina, J., et al. (2023). On the Feasibility of Market Manipulation and Energy Storage Arbitrage via Load-Altering Attacks. Energies, 16(4), 1670.

https://github.com/lanl-ansi/PowerModelsITD.jl

PowerModelsITD.jl: Core Design

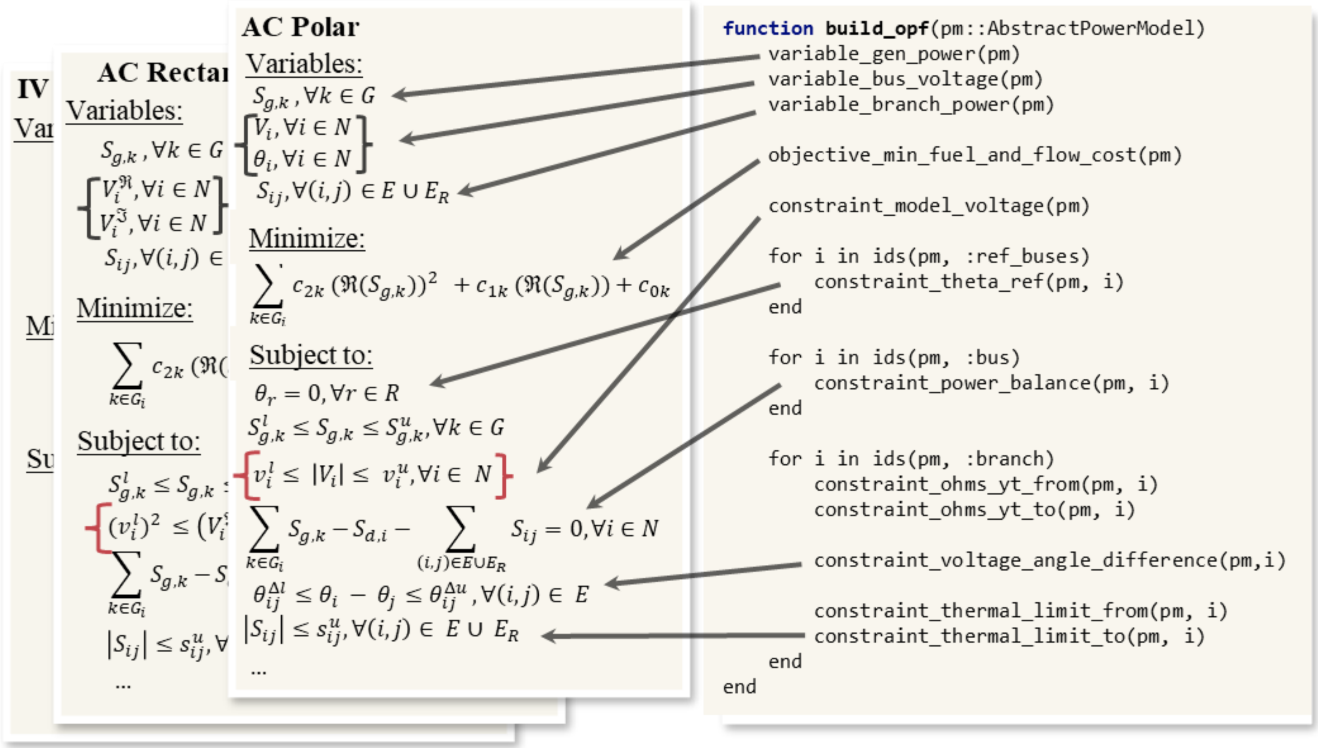
Problem Specifications

Integrated T&D Power Flow (pfitd)
Integrated T&D Optimal Power Flow (opfitd)
…

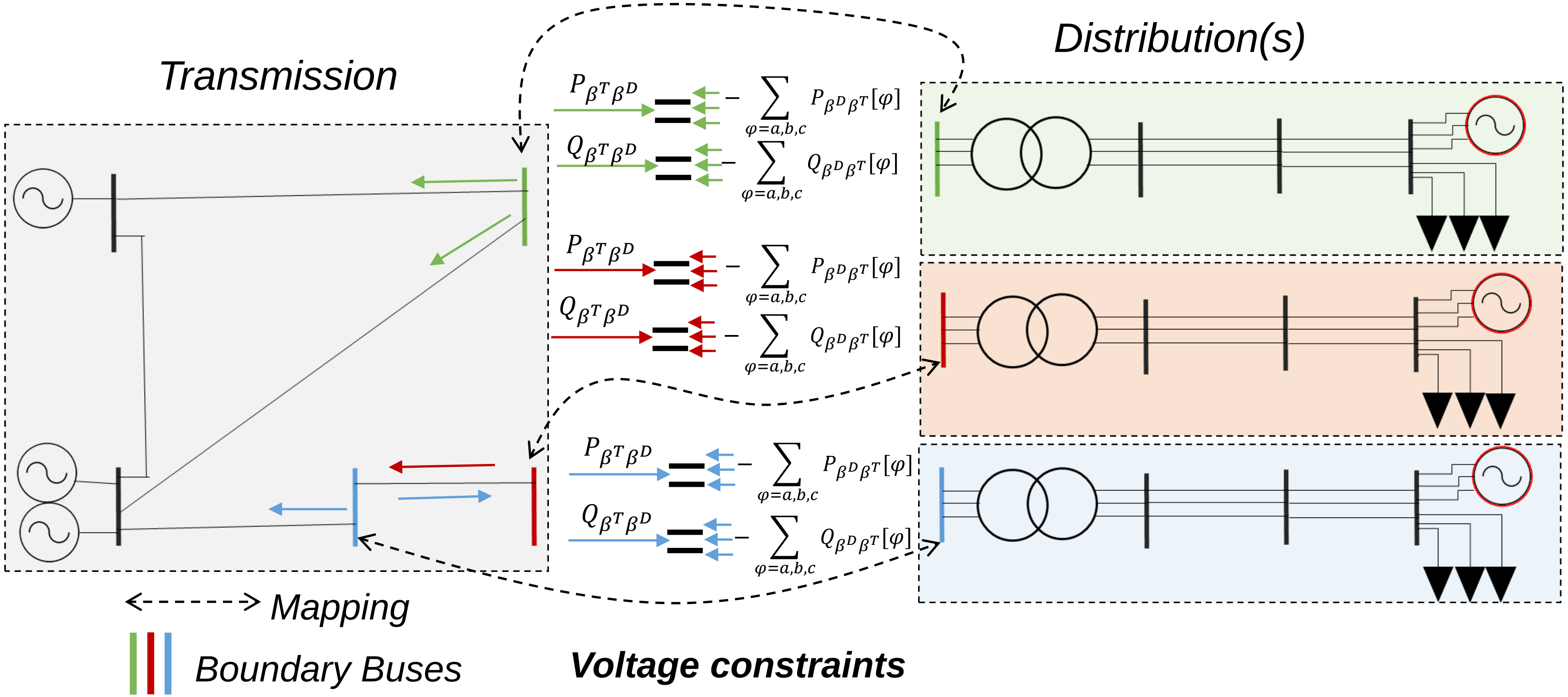
Formulations

ACP-ACPU
ACR-ACRU
IVR-IVRU
NFA-NFAU
SOCBFM-
LinDis3Flow
…

Core language feature:
<u>Multiple dispatch</u>



PowerModelsITD.jl: Boundaries



Voltage constraints

$$|V^T| = \begin{matrix} |V^D|[a] \\ |V^D|[b] \\ |V^D|[c] \end{matrix} \qquad \measuredangle V^T = \measuredangle V^D[a]$$

$$\measuredangle V^D[b] = \measuredangle V^D[a] - 120°$$

$$\measuredangle V^D[c] = \measuredangle V^D[a] + 120°$$

PowerModelsITD.jl: Formulations

NLP Formulations

- ACP-ACPU
  - Power-Voltage, polar coordinates, non-linear (NLP)
- ACR-ACRU
  - Power-Voltage, rectangular coordinates, non-linear (NLP)
- IVR-IVRU
  - Current-Voltage, rectangular coordinates, non-linear (NLP)

Linear Approximations

- NFA-NFAU
  - Network Flow Approximation
  - Active power only, lossless, linear (LP)
- BFA-LinDist3Flow
  - Branch Flow Approximation - Linear Approximation

Relaxations

- SOCBFM-SOCUBFM
  - Second Order Cone Branch Flow Model Relaxations – W-space.

Hybrid Formulations (Experimental)

- ACR-FOTRU
  - Power-Voltage NLP, rectangular coordinates, First-Order Taylor Approximation
- ACP-FOTPU
  - Power-Voltage NLP, polar coordinates, First-Order Taylor Approximation
- ACR-FBSU
  - Power-Voltage NLP, rectangular coordinates, Forward-Backward Sweep Approximation
- SOCBFM-LinDist3Flow
  - Second Order Cone Branch Flow Model Relaxation – W-space.
  - Linear Approximation.

Using PowerModelsITD.jl

Using PowerModelsITD.jl: Files

Transmission file

```
function mpc = case5
mpc.version = '2';
mpc.baseMVA = 100.0;

%% bus data
%	bus_i	type	Pd	Qd	Gs	Bs	area	Vm	Va	baseKV	zone
mpc.bus = [
	1	2	0.0	0.0	0.0	0.0	1		1.07762		2.80377
	2	1	300.0	98.61	0.0	0.0	1		1.08407		-0.73465
	3	2	300.0	98.61	0.0	0.0	1		1.10000		-0.55972
	4	3	390.0	131.47	0.0	0.0	1		1.06414		0.00000
	5	1	8.0	1.2	0.0	0.0	1		1.00000		0.00000
	10	2	0.0	0.0	0.0	0.0	1		1.06907		3.59033
];

%% generator data
%	bus	Pg	Qg	Qmax	Qmin	Vg	mBase	status	Pmax	Pmin
mpc.gen = [
	1	40.0	30.0	30.0	-30.0	1.07762		100.0	1	40.0	0.0;
	1	170.0	127.5	127.5	-127.5	1.07762		100.0	1	170.0	0.0;
	3	324.498		390.0	390.0	-390.0	1.1	100.0	1	520.0	0.0;
	4	0.0	-10.802		150.0	-150.0	1.06414		100.0	1	200.0
	10	470.694		-165.039		450.0	-450.0	1.06907		100.0	1
];

%% generator cost data
%	2	startup	shutdown	n	c(n-1)	...	c0
mpc.gencost = [
	2	0.0	0.0	3	0.000000	14.000000	0.000000	2.000
	2	0.0	0.0	3	0.000000	15.000000	0.000000	2.000
	2	0.0	0.0	3	0.000000	30.000000	0.000000	2.000
	2	0.0	0.0	3	0.000000	40.000000	0.000000	2.000
	2	0.0	0.0	3	0.000000	10.000000	0.000000	2.000
];

%% branch data
%	fbus	tbus	r	x	b	rateA	rateB	rateC	ratio	angle	status
mpc.branch = [
	1	2	0.00281		0.0281	0.00712		400.0	400.0	400.0	0.0
	1	4	0.00304		0.0304	0.00658		426	426	426	0.0
	1	10	0.00064		0.0064	0.03126		426	426	426	0.0
	2	3	0.00108		0.0108	0.01852		426	426	426	0.0
	3	4	0.00297		0.0297	0.00674		426	426	426	1.05
	4	10	0.00297		0.0297	0.00674		240.0	240.0	240.0	0.0
	2	5	0.00297		0.0297	0.00674		426	426	426	0.0
];
```

MATPOWER (“.m”)
PSS(R)E v33 specification (".raw")

(support PowerWorld for PSSE conversions)

Distribution

```
New Circuit.3bus_bal
!  define a really stiff source
~ basekv=230 pu=1.00 MVAsc3=200000 MVAsc1=210000

! Substation Transformer
New Transformer.SubXF Phases=3 Windings=2 Xhl=0.01
~ wdg=1 bus=sourcebus conn=wye kv=230   kva=25000    %r=0.0005
~ wdg=2 bus=Substation   conn=wye kv=13.8  kva=25000   %r=0.0005

!Define Linecodes
New linecode.556MCM nphases=3 basefreq=60  ! ohms per 5 mile
~ rmatrix = ( 0.1000 | 0.0400    0.1000 |  0.0400    0.0400    0.1000)
~ xmatrix = ( 0.0583 |  0.0233    0.0583 | 0.0233    0.0233    0.0583)
~ cmatrix = (50.92958178940651  | -0  50.92958178940651 | -0 -0 50.92958178940651  ) ! small cap

New linecode.4/0QUAD nphases=3 basefreq=60  ! ohms per 100ft
~ rmatrix = ( 0.1167 | 0.0467    0.1167 | 0.0467    0.0467    0.1167)
~ xmatrix = (0.0667  |  0.0267    0.0667  |  0.0267    0.0267    0.0667 )
~ cmatrix = (50.92958178940651  | -0  50.92958178940651 | -0 -0 50.92958178940651  )  ! small ca

!Define lines
New Line.OHLine  bus1=Substation.1.2.3  Primary.1.2.3  linecode = 556MCM   length=1 normamps=600
New Line.Quad    Bus1=Primary.1.2.3  loadbus.1.2.3  linecode = 4/0QUAD  length=1 normamps=6000 e

!Loads - single phase
New Load.L1 phases=1  loadbus.1.0   ( 13.8 3 sqrt / )   kW=3000   kvar=1500  model=1
New Load.L2 phases=1  loadbus.2.0   ( 13.8 3 sqrt / )   kW=3000   kvar=1500  model=1
New Load.L3 phases=1  loadbus.3.0   ( 13.8 3 sqrt / )   kW=3000   kvar=1500  model=1

!GENERATORS DEFINITIONS
New generator.gen1 Bus1=loadbus.1.2.3 Phases=3 kV=( 13.8 3 sqrt / )  kW=2000 pf=1 conn=wye Model

Set VoltageBases = "230,13.8"
Set tolerance=0.000001
set defaultbasefreq=60
```

```
)651  ) ! small cap
)651  )  ! small ca
gth=1 normamps=600
=1 normamps=6000 e

!Loads - single phase
New Load.L1 phases=1  loadbus.1.0   ( 13.8 3 sqrt / )   kW=3000   kvar=1500  model=1
New Load.L2 phases=1  loadbus.2.0   ( 13.8 3 sqrt / )   kW=3000   kvar=1500  model=1
New Load.L3 phases=1  loadbus.3.0   ( 13.8 3 sqrt / )   kW=3000   kvar=1500  model=1

!GENERATORS DEFINITIONS
New generator.gen1 Bus1=loadbus.1.2.3 Phases=3 kV=( 13.8 3 sqrt / )  kW=2000 pf=1 conn=wye Model

Set VoltageBases = "230,13.8"
Set tolerance=0.000001
set defaultbasefreq=60
```

OpenDSS (".dss")

Boundary file

```
[
    {
    "transmission_boundary": "5",
    "distribution_boundary": "3bus_unbal.voltage_source.source"
    },

    {
    "transmission_boundary": "6",
    "distribution_boundary": "3bus_bal.voltage_source.source"
    }
]
```

JSON (".json")

*other proprietary file formats supported via **DiTTo [4]***

[4] “DiTTo (Distribution Transformation Tool),” 2021, Accessed: Aug. 06, 2021. [Online]. Available: https://github.com/NREL/ditto

Using PowerModelsITD.jl: Solving

Case w/ 1 distro. system

```julia
using PowerModelsITD
import Ipopt
ipopt = Ipopt.Optimizer

# Path for the files
pmitd_path = joinpath(dirname(pathof(PowerModelsITD)), "..")

# Files
pm_file = joinpath(pmitd_path, "test/data/transmission/case5_withload.m")
pmd_file = joinpath(pmitd_path, "test/data/distribution/case3_balanced.dss")
boundary_file = joinpath(pmitd_path, "test/data/json/case5_case3_bal.json")

pmitd_type = NLPowerModelITD{ACPPowerModel, ACPUPowerModel}

result = solve_opfitd(pm_file, pmd_files, boundary_file, pmitd_type, ipopt)
```

Case w/ 2 distro. systems

```julia
using PowerModelsITD
import Ipopt
ipopt = Ipopt.Optimizer

# Path for the files
pmitd_path = joinpath(dirname(pathof(PowerModelsITD)), "..")

# Files
pm_file = joinpath(pmitd_path, "test/data/transmission/case5_with2loads.m")
pmd_file1 = joinpath(pmitd_path, "test/data/distribution/case3_unbalanced.dss")
pmd_file2 = joinpath(pmitd_path, "test/data/distribution/case3_balanced.dss")
boundary_file = joinpath(pmitd_path, "test/data/json/case5_case3x2_unbal_bal.json")

pmd_files = [pmd_file1, pmd_file2] # vector of files
pmitd_type = NLPowerModelITD{ACPPowerModel, ACPUPowerModel}

result = solve_opfitd(pm_file, pmd_files, boundary_file, pmitd_type, ipopt)
```

Simple User Interface → Easy User Adoption

Using PowerModelsITD.jl: Results

```
julia> result
Dict{String, Any} with 8 entries:
  "solve_time"         => 0.12712
  "optimizer"          => "Ipopt"
  "termination_status" => LOCALLY_SOLVED
  "dual_status"        => FEASIBLE_POINT
  "primal_status"      => FEASIBLE_POINT
  "objective"          => 18146.3
  "solution"           => Dict{String, Any}("multiinfrastructure"=>true, "it"=>Dict{String, Any}("pmd…
  "objective_lb"       => -Inf
```

Transmission

```
julia> result["solution"]["it"]["pm"]
Dict{String, Any} with 6 entries:
  "baseMVA"      => 100.0
  "branch"       => Dict{String, Any}("3"=>Dict{String, Any}("qf"=>206.656, "qt"=>-202.276, "pt"=>221.006, "pf"=>-220.308), "4"=>Dict{String, Any}("qf"=>-217.108, "qt"=>221.882, "pt"=>79.0383, "pf"=>-78.3924), "1"=…
  "gen"          => Dict{String, Any}("4"=>Dict{String, Any}("qg"=>56.3262, "pg"=>18.0328), "1"=>Dict{String, Any}("qg"=>30.0, "pg"=>40.0), "5"=>Dict{String, Any}("qg"=>-201.205, "pg"=>461.003), "2"=>Dict{String, A…
  "multinetwork" => false
  "bus"          => Dict{String, Any}("4"=>Dict{String, Any}("va"=>-1.06955e-34, "vm"=>0.9), "1"=>Dict{String, Any}("va"=>3.95367, "vm"=>0.917681), "5"=>Dict{String, Any}("va"=>-0.949629, "vm"=>0.937736), "2"=>Dict…
  "per_unit"     => false
```

Distribution

```
julia> result["solution"]["it"]["pmd"]
Dict{String, Any} with 7 entries:
  "line"        => Dict{String, Any}("3bus_unbal.quad"=>Dict{String, Any}("qf"=>[1344.85, 1503.97, 1502.46], "qt"=>[-1333.33, -1500.0, -1500.0], "pt"=>[-3333.33, -2333.33, -2333.33], "pf"=>[3351.62, 2340.39, 2344.9…
  "settings"    => Dict{String, Any}("sbase"=>100000.0)
  "transformer" => Dict{String, Any}("3bus_bal.subxf"=>Dict{String, Any}("q"=>[[1508.51, 1508.51, 1508.51], [-1508.41, -1508.41, -1508.41]], "p"=>[[2351.59, 2351.59, 2351.59], [-2351.58, -2351.58, -2351.58]]), "3bu…
  "generator"   => Dict{String, Any}("3bus_unbal.gen1"=>Dict{String, Any}("qg_bus"=>[-0.0, -0.0, -0.0], "qg"=>[-0.0, -0.0, -0.0], "pg"=>[666.668, 666.668, 666.668], "pg_bus"=>[666.668, 666.668, 666.668]), "3bus_bal…
  "load"        => Dict{String, Any}("3bus_unbal.l2"=>Dict{String, Any}("qd_bus"=>[1500.0], "pd_bus"=>[3000.0], "qd"=>[1500.0], "pd"=>[3000.0]), "3bus_bal.l3"=>Dict{String, Any}("qd_bus"=>[1500.0], "pd_bus"=>[3000.…
  "bus"         => Dict{String, Any}("3bus_unbal.loadbus"=>Dict{String, Any}("va"=>[-1.10106, -120.971, 119.172], "vm"=>[7.38801, 7.42776, 7.41273]), "3bus_bal.substation"=>Dict{String, Any}("va"=>[-1.08179, -121.0…
  "per_unit"    => false
```

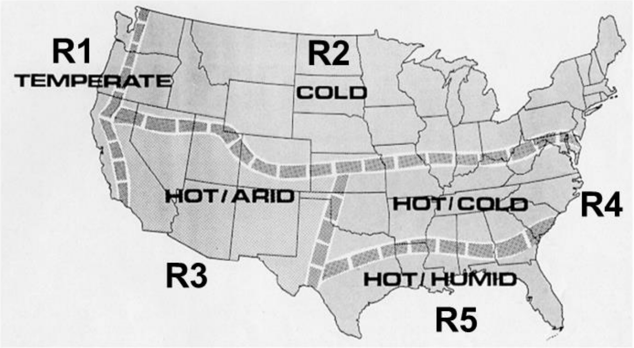
Boundary

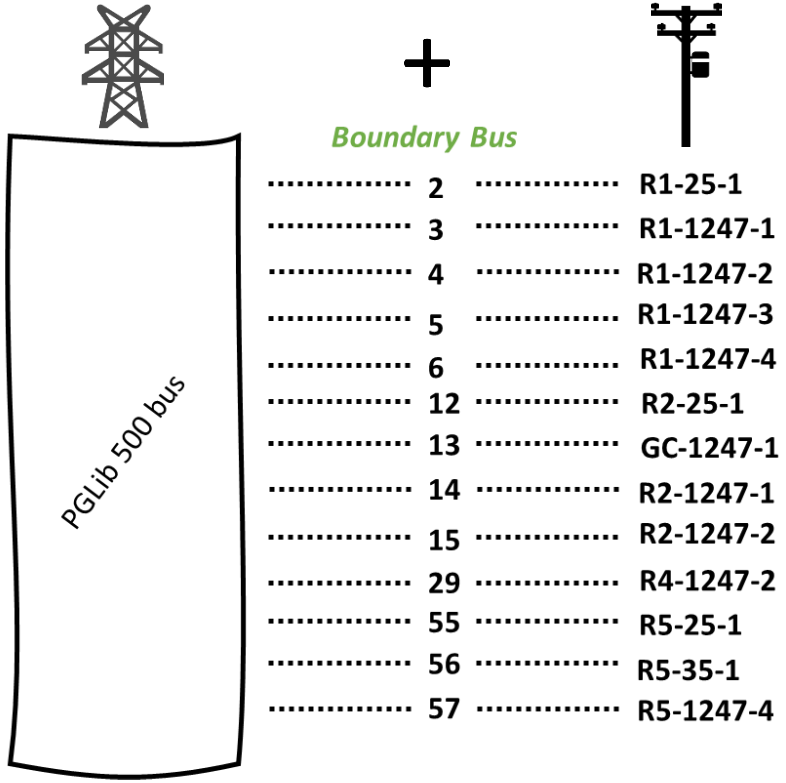
```
julia> result["solution"]["it"]["pmitd"]["boundary"]
Dict{String, Any} with 4 entries:
  "(100001, 5, voltage_source.3bus_unbal.source)" => Dict{String, Any}("pbound_fr"=>[8068.8], "qbound_fr"=>[4367.42])
  "(100001, voltage_source.3bus_unbal.source, 5)" => Dict{String, Any}("pbound_to"=>[-3367.36, -2346.47, -2354.97], "qbound_to"=>[-1355.14, -1507.53, -1504.75])
  "(100002, voltage_source.3bus_bal.source, 6)"   => Dict{String, Any}("pbound_to"=>[-2351.62, -2351.62, -2351.62], "qbound_to"=>[-1508.64, -1508.64, -1508.64])
  "(100002, 6, voltage_source.3bus_bal.source)"   => Dict{String, Any}("pbound_fr"=>[7054.87], "qbound_fr"=>[4525.93])
```

Use Cases & Tests

Use Cases & Tests: OPF

Taxonomy
PNNL Cases [5]





Test Cases	N	E
case_r1_25_1	759	762
case_r1_1247_1	3403	3583
case_r1_1247_2	1450	1527
case_r1_1247_3	168	165
case_r1_1247_4	970	981
case_r2_25_1	1617	1681
case_gc_1247_1	96	93
case_r2_1247_1	1731	1750
case_r2_1247_2	1207	1275
case_r4_1247_2	1155	1202
case_r5_25_1	3116	3250
case_r5_35_1	1435	1505
case_r5_1247_4	2030	2088

Totals: 19,137 19,862

Totals:

Buses/Nodes: 19,637 (w/ +500 from transmission)

Edges: 20,595 (w/ +733 from transmission)

PVs

55

DGs

17

[5] Schneider, Kevin P., et al. *Modern grid initiative distribution taxonomy final report*. No. PNNL-18035. Pacific Northwest National Lab.(PNNL), Richland, WA (United States), 2008.

Use Cases & Tests: OPF Results

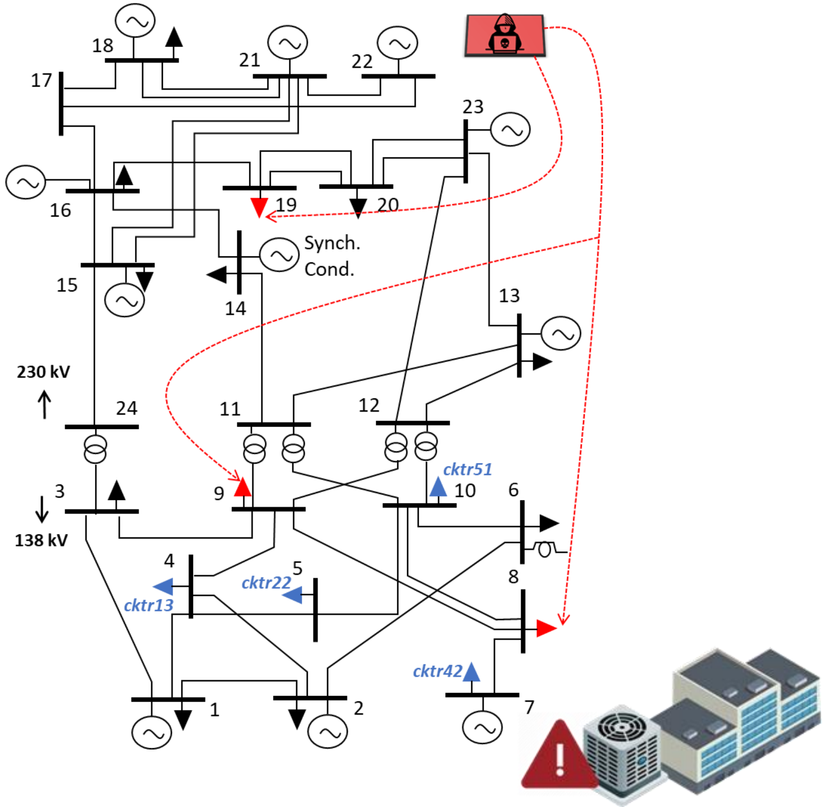
CPU: x6 Cores @ 2.80 Ghz
RAM: 128 GB

Ipopt vers.: 3.14.4
MUMPS vers.: 5.4.1

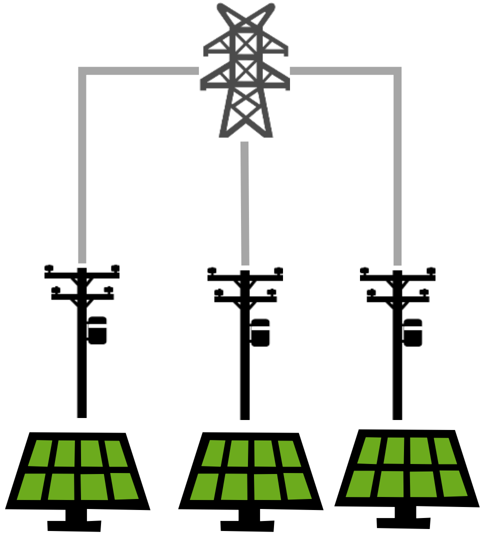
Case PNNL - All Regions			
Formulation	**$/hr**	**Time (s)**	**Iterations**
ACP-ACPU	422,095.2350	525.154	94
IVR-IVRU	422,095.2348	360.954	99
NFA-NFAU	412,286.7567	10.860	24
ACR-FBSUBF	422,074.7218	226.852	97
BFA-LinDist3	412,286.7567	146.084	45
SOCBF-LinDist3	421,529.7893	241.203	75

Use Cases & Tests: Other Use Cases

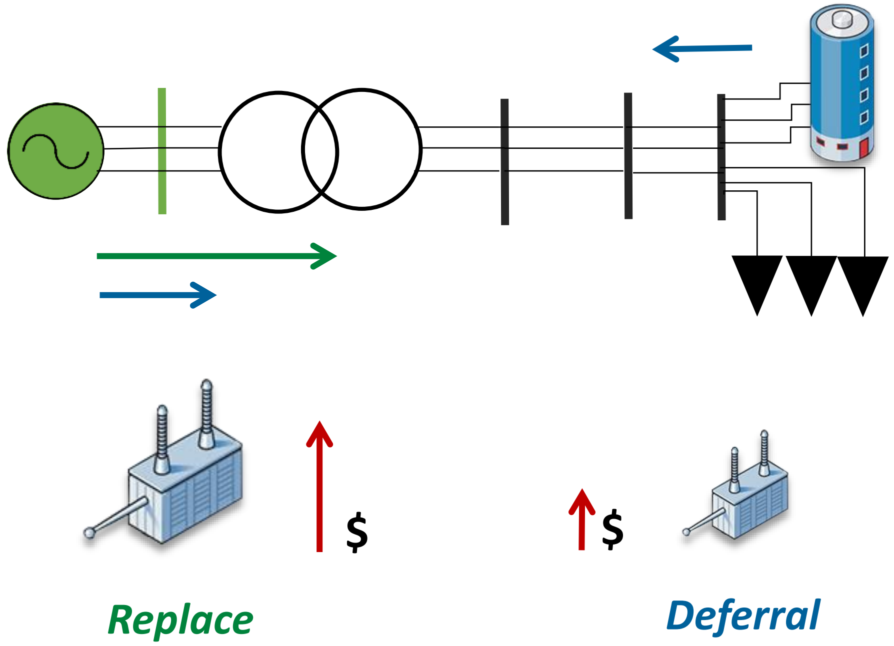
T&D
Market Manipulation via Load-Altering Attacks [6]



Hosting Limit Capacity for T&D



T&D Coordination Transformer Deferral



[6] Ospina, J., Fobes, D. M., & Bent, R. (2023). On the Feasibility of Market Manipulation and Energy Storage Arbitrage via Load-Altering Attacks. *Energies*, *16*(4), 1670. url: https://www.mdpi.com/1996-1073/16/4/1670

Thank you
Questions?

Contacts:

- Juan Ospina: jjospina@lanl.gov
- David M. Fobes: dfobes@lanl.gov