Collection of Power Flow models: Usage and documentation

University of Wisconsin - Madison

May 20, 2015

1 Quick Start Guide

To get started, extract the zip archive of the models and Matpower IEEE test-cases to a convenient place. Section 3 (Model Usage) provides available runtime options.

2 Model descriptions

2.1 Power Flow Models

1. DC power flow models
   (a) `dcopf.gms`
   Regular DC power flow model.
   (b) `dcopf_shift.gms`
   DC power flow model with shift matrices.
2. AC power flow models
   (a) `polar_acopf.gms`
   Polar-power voltage formulation.
   (b) `rect_acopf.gms`
   Rectangular power-voltage formulation.
   (c) `iv_acopf.gms`
   Rectangular current-voltage formulation.
3. Decoupled AC power flow models
   (a) `polar_decoupled.gms`
   Polar-power voltage formulation.
   (b) `ybus_polar_decoupled.gms`
   Ybus Polar-power voltage formulation.

4. Ybus AC power flow models
   (a) `ybus_polar_acopf.gms`
       Polar-power voltage formulation.
   (b) `ybus_rect_acopf.gms`
       Rectangular power-voltage formulation.
   (c) `ybus_iv_acopf.gms`
       Rectangular current-voltage formulation.
5. `uc_dc.gms`
   Optimal DC power flow model with unit commitment.
6. Unit commitment AC power flow models
   (a) `uc_polar.gms`
       Polar-power voltage formulation with unit commitment.
   (b) `uc_rect.gms`
       Rectangular power-voltage formulation with unit commitment.
   (c) `uc_iv.gms`
       Rectangular current-voltage formulation with unit commitment.

The GAMS models take as input a GDX file written by the `to_gdx.gms` utility. Currently, this utility supports reading from a Matpower-formatted `'.m'` structure file using an awk script and from a PSSE-formatted raw text file using an awk script for comma delimited files or a C++ application for comma or space delimited files. Additional information on data utilities can be found in the Data Utilities section of this project.

3 Power Flow Model usage

The basic model can be run from the command line with a single option:
`gams model.gms --case=/path/case.gdx`

Note that running `dcopf_shift.gms` requires an additional dataset, namely the shift matrices. Please refer to Section 4 for more details.

Sections 3.1 and 3.2 provide general options based on whether they are single period power flow or (up to) multi-period unit commitment models. Additional options specific to the DC and AC formulation are listed in Section 3.3 and 3.4 respectively.

3.1 OPF model options

These options apply to models that do not incorporate unit commitment, such as `dcopf.gms, polar_acopf.gms, iv_acopf.gms` etc. These models solve the power flow model for a single time period and rely on the dataset to provide the on/off status of generators. Do not include square braces [ ] in the options.

1. `--timeperiod=[#]`
   Select which timeperiod to solve in a specific dataset. (*Default = 1*)

2. `--obj=[objective]`
   - `pwl`: Piecewise linear objective
   - `quad`: Quadratic objective (*Default*)
   - `linear`: Simplified linear objective that uses only linear components of the quadratic objective function. Useful for finding feasibility.
   - `0`: 0 objective function. Useful for solving feasibility models.

3. `--linelimits=[datatype]`
   - `given`: Uses original source data on line limits (*Default*)
   - `uwcalc`: Inferred line limits calculated according to the System Characteristic Inference section of this study
   - `inf`: Removes all line limits, sets them to infinity

4. `--genPmin=[datatype]`
   - `0`: Removes lower bound on generator operating limits, i.e. set to 0
   - `given`: Uses original source data for minimum operating limit of generators (*Default*)
   - `uwcalc`: Inferred line limits calculated according to the System Characteristic Inference section of this study

5. `--allon=[option]`
   - `gens`: All generators are on/active and abide by operating bounds
   - `lines`: All valid lines are on/active in the network
   - `both`: All valid lines and generators are active
   - (*Default*) This option is ignored

6. `--savesol=[#]`
   - `0`: Solution is not saved (*Default*)
   - `1`: Solution is saved into `gdx` file, in similar format as input file

   ** For more a thorough save of the environment, the GAMS command line option* `gdx=out` *will save all data, variable and equation information at the end of the run into* `outfile.gdx`*. See GAMS manual for further information.*

7. `--verbose=[#]`
   - `0`: Regular listing file (`.lst`) output (*Default*)
   - `1`: Listing file output is suppressed before model solve

3.2 Unit commitment model options

These options apply to models incorporating unit commitment, `uc_dc.gms` and `uc_ac.gms`. They are able to solve the power flow model for single or multiple time periods, and include ramping constraints which are not considered in single period models. Note the difference in the default objective function used here, compared to the regular powerflow models in Section 3.1. Do not include square braces [ ] in the options.

1. `--times=[timeframe]`: Select which timeframe to solve the model.
    - `t`: Solve a single time period, t.
    - "t_1*t_n": Solve multiple time periods, from t_1 to t_n
    - (*Default*) Time frame is read in from the input datafile
2. `--obj=[objective]`
    - `pwl`: Piecewise linear objective (*Default*)
    - `quad`: Quadratic objective
3. `--demandbids=[#]`
    - `0`: Demand is fixed (*Default*)
    - `1`: Incremental elastic demand bidding is allowed, if data available
4. `--linelimits=[datatype]`
    - `given`: Uses original source data on line limits (*Default*)
    - `uwcalc`: Inferred line limits calculated according to the System Characteristic Inference section of this study
    - `inf`: Removes all line limits, sets them to infinity
5. `--genPmin=[datatype]`
    - `0`: Removes lower bound on generator operating limits, i.e. set to 0
    - `given`: Uses original source data for minimum operating limit of generators (*Default*)
    - `uwcalc`: Inferred line limits calculated according to the System Characteristic Inference section of this study
6. `--ramprates=[datatype]`
    - `given`: Uses original source data for ramp up and ramp down rates (*Default*)
    - `uwcalc`: Inferred ramping limits calculated according to the System Characteristic Inference section of this study
7. `--allon=[option]`

- `lines`: All valid lines are on/active in the network
- (*Default*) This option is ignored

8. `--savesol=[#]`

   - `0`: Solution is not saved (*Default*)
   - `1`: Solution is saved into `gdx` file, in similar format as input file

9. `--verbose=[#]`

   - `0`: Regular listing file (`.lst`) output (*Default*)
   - `1`: Listing file output is suppressed before model solve

10. `--relax=[#]`

    - `0`: Model solved regularly (*Default*)
    - `1`: Relaxed integer models, e.g. solved with rmip/rmiqcp

3.3 DC specific model options

1. `--lineloss=[#]`: Approximates line loss by increasing demand.

   - `0`: No changes to provided demand profile. (*Default*)
   - `1`: Increase active demand values by 5.5%.

3.4 AC specific model options

1. `--qlim=[#]`: Whether to enforce reactive power limits as D-curve circle constraints

   - `0`: Ignore D-curves, instead just use rectangle constraints. (*Default*)
   - `1`: Include D-curve constraints.

   *** Does not apply to polar_decoupled*

2. `--slim=[#]`: Whether to enforce apparent power limits instead of simple bound active line power constraints

   - `0`: Use simple bounds on active line power. (*Default*)
   - `1`: Use apparent power limits.

   *** Does not apply to iv_acopf, polar_decoupled or ybus models*

3. `--ic=[#]`: Choose method for generating initial conditions, i.e. NLP starting point

   - `0`: [**Midpoint**] Begin with all voltage magnitudes, voltage angles, real power, and reactive power variables at the midpoint of their bounds, calculating line flow variables from these values. (*Default*)

- `1`: **[Random]** All variables initialized using random draws between variable bounds.
- `2`: **[Flat]** Flat start, where all initial guesses for voltage magnitude and voltage angle are set to 1.0 and zero, respectively, and power flow initial guesses are set to zero.
- `3`: **[Random/AC]** Voltage magnitude and voltage angle variables are initialized using random draws between variable bounds. Real and reactive power variables are initialized using AC transmission line model (applied to each line separately) and the initialized voltage magnitude and voltage angle values.
- `4`: **[DC/AC]** Real power and voltage angle values are initialized using a DCOPF model. Voltage magnitudes are initialized at 1.0. Reactive power is initialized using relevant equations from the AC transmission line model (applied to each line separately) and the initialized voltage magnitude and voltage angle values.
- `5`: **[DC-/AC]** Voltage angle values are initialized using a DCOPF model (real power values are obtained in the DCOPF, but discarded). Voltage magnitudes are initialized at 1.0. Real and reactive power variables are initialized using the AC transmission line model (applied to each line separately) and the initialized voltage magnitude and voltage angle values.
- `6`: **[Decoupled]** Voltage magnitude, voltage angle, real power and reactive power variables are initialized using a decoupled ACOPF model.
- `7`: **[DCLoss]** Real power and voltage angle values are initialized using a DCOPF model with line loss approximation (`--lineloss=1.055`). Voltage magnitudes are initialized at 1.0. Reactive power is initialized using relevant equations from the AC transmission line model (applied to each line separately) and the initialized voltage magnitude and voltage angle values.
- `8`: **[Matpower]** Use voltage magnitude, voltage angle, real power, and reactive power values given in Matpower solutions (if available).
- `9`: **[inputFile]** Use voltage magnitude, voltage angle, real power, and reactive power values given in the GDX file.

*** Does not apply to polar_decoupled*

4 Additional Notes

1. **Shift matrices**
   In order to use `dcopf_shift.gms`, a gdx file containing the shift matrix needs to be at the same location as the input case file. In other words, `gams dcopf_shift.gms --case=/path/case.gdx` will search for

/path/case_Shift_Matrix.gdx.
If the shift matrix does not already exist, use calc_S_matrix.gms in the Data utilities section to generate the necessary file.
*** Currently, the LMP is not computed for dcopf_shift.gms.*

2. **--wind**
   For specialized datasets that define geninfo(gen, 'PrimeMover', 'pm_WT' (refer to the Appendix of datautilities.pdf for additional information), option --wind sets the generation of wind turbines to 0. This option is available for dcopf.gms, dcopf_shift.gms, uc_dc.gms, polar_acopf.gms, rect_acopf.gms, iv_acopf.gms, uc_polar.gms, uc_rect.gms, uc_iv.gms. Note that this information does not exist for any of the IEEE or RTS testcases in the testcase archive.