Hybrid Symbolic-Numeric Framework for Power System Modeling and Simulation

2020 CURENT NSF/DOE Site Visit and Industry Conference
Virtual
Nov. 2020

Hantao Cui, Fangxing Li, Kevin Tomsovic
The University of Tennessee, Knoxville
Hybrid Symbolic-Numeric Framework for Power System Modeling and Simulation

Background and motivation:
- Reduce the programming efforts for complex models such as renewable energy generators
- Readily interfaced with data analytics and machine learning tools
- Implemented in the ANDES software and distributed free and open-source

Technical approach:
- Modeling using both symbolic descriptive approach and traditional approach
- Symbolically derive partial derivatives
- Generates vectorized numerical code
- Perform fast numerical simulations

Conclusion:
- Descriptive symbolic modeling can significantly simplify the modeling of complex devices
- Implemented over 50 models in the proposed framework, including synchronous generators, generator controllers, solar PVs and wind generators
- Simulation results are verified with PSS/E and TSAT
Hybrid Symbolic-Numeric Framework - Modeling

• An Example of Developing Models
  ◦ The IEEEST stabilizer model can be implemented in a few lines

```python
from andes.core.block import Lag2ndOrd, LeadLag2ndOrd, LeadLag, Gain, WashoutOrLag, Algeb
from andes.core.discrete import Limiter
from andes.core.model import Model

class IEEESTModel(Model):
    def __init__(self, system, config):
        Model.__init__(self, system, config)
        # omitted: setup input signal `sig`

        self.F1 = Lag2ndOrd(u=self.sig, K=1, T1=self.A1, T2=self.A2)  # filter 1
        self.F2 = LeadLag2ndOrd(u=self.F1_y, T1=self.A3, T2=self.A4,
                                T3=self.A5, T4=self.A6, zero_out=True)  # filter 2

        self.LL1 = LeadLag(u=self.F2_y, T1=self.T1, T2=self.T2)  # filter 3
        self.LL2 = LeadLag(u=self.LL1_y, T1=self.T3, T2=self.T4)  # filter 4
        self.VKs = Gain(u=self.LL2_y, K=self.KS)  # Gain
        self.WO = WashoutOrLag(u=self.VKs_y, T=self.T6, F=self.T5, name='WO')
        self.VLIM = Limiter(u=self.WO_y, lower=self.CMIN, upper=self.CMAX)  # Vss limiter
        self.Vss = Algeb(e_str='VLIM_zi * WO_y + VLIM_zu * LMAX + VLIM_z1 * LMIN - Vss')
        self.OLIM = Limiter(u=self.v, lower=self.VCLR, upper=self.VCUR)  # output limiter
        self.v.out, e_str = 'OLIM_zi * Vss - vsout'  # output equation
```

• IEEE 14-bus system and NPCC test system
  ○ Using models GENROU, ESST3A, EXDC2, SEXS, IEEEG1, TGOV1, IEEEST and ST2CUT
  ○ Perfect match with TSAT results
Acknowledgements

This work was supported primarily by the ERC Program of the National Science Foundation and DOE under NSF Award Number EEC-1041877 and the CURENT Industry Partnership Program.

Other US government and industrial sponsors of CURENT research are also gratefully acknowledged.