Berkeley MAPP and VAPP

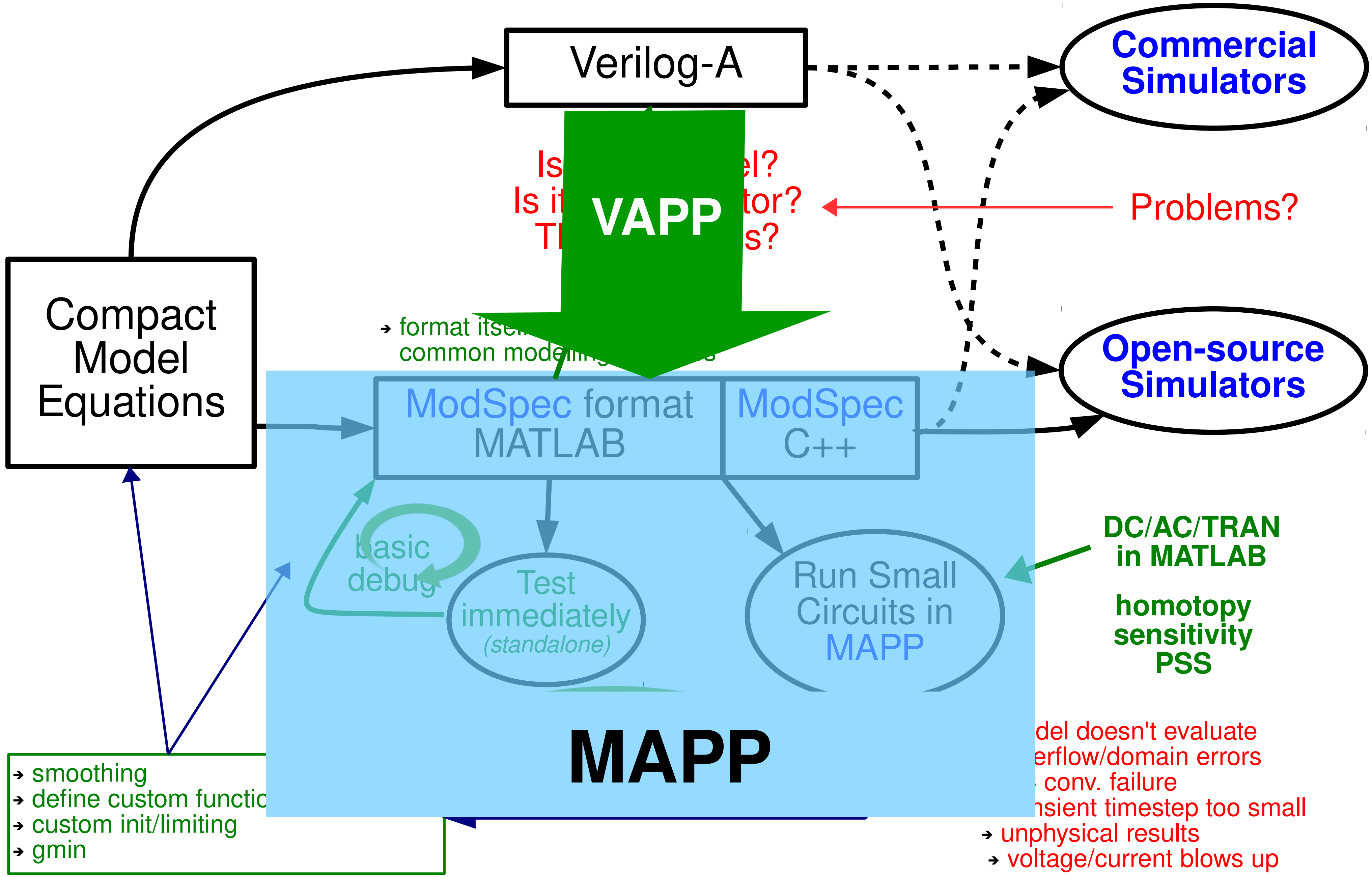
(Model and Algorithm Prototyping Platform)

(Verilog-A Parser and Processor)

Tianshi Wang, A. Gokcen Mahmutoglu, Karthik Aadithya*, Archit Gupta and Jaijeet Roychowdhury

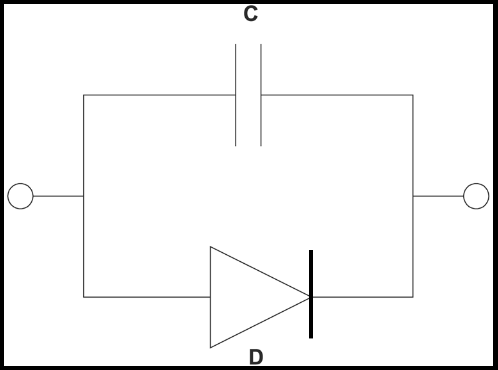
EECS Department, University of California, Berkeley
*Sandia National Laboratories

Compact Model Development



What's ModSpec: a glimpse

```
function MOD = diodeCapacitor_ModSpec_wrapper()
% ModSpec description of an ideal diode in parallel with a capacitor
   MOD = ee_model();
   MOD = add_to_ee_model(MOD, 'external_nodes', {'p', 'n'});
   MOD = add_to_ee_model(MOD, 'explicit_outs', {'ipn'});
   MOD = add_to_ee_model(MOD, 'parms', {'C',2e-12, 'Is',1e-12, 'VT',0.025});
   MOD = add_to_ee_model(MOD, 'f', @f);
   MOD = add_to_ee_model(MOD, 'q', @q);
end

function out = f(S)
    v2struct(S);
    out = Is*(exp(vpn/VT)-1);
end

function out = q(S)
    v2struct(S);
    out = C*vpn;
end
"diodeCapacitor_ModSpec_wrapper.m" 19L, 548C written          1,1          All
```



MOD.terminals
MOD.parms
MOD.explicit_outs
MOD.f: function handle
MOD.q: function handle
...

vpn

ipn

$$\vec{z} = \frac{d}{dt}\vec{q}_e(\vec{x},\vec{y}) + \vec{f}_e(\vec{x},\vec{y},\vec{u})$$

$$\vec{0} = \frac{d}{dt}\vec{q}_i(\vec{x},\vec{y}) + \vec{f}_i(\vec{x},\vec{y},\vec{u})$$

Differential Algebraic Equations

What's ModSpec: a glimpse

Executable & debuggable standalone

Easy to examine/write by hand

General: any device in any physical domain

Easily & directly usable by any simulator

Supports every analysis DC/AC/tr/PSS

Mathematically well defined, modular

MOD.terminals
MOD.parms
MOD.explicit_outs
MOD.f: function handle
MOD.q: function handle
...

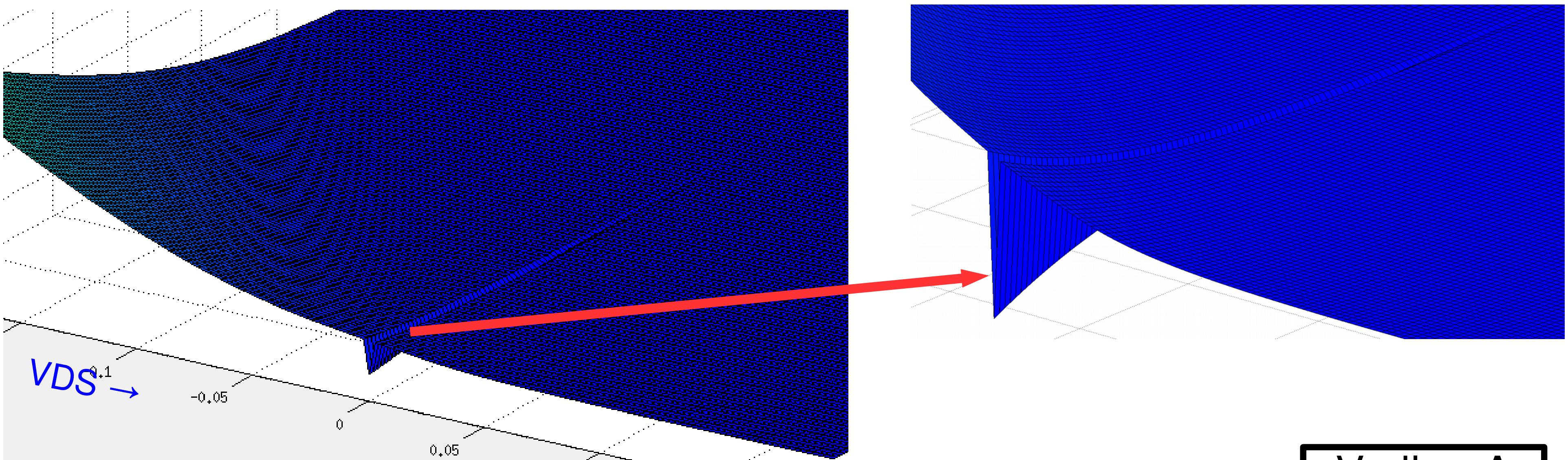
$$\vec{z} = \frac{d}{dt}\vec{q}_e(\vec{x},\vec{y}) + \vec{f}_e(\vec{x},\vec{y},\vec{u})$$

$$\vec{0} = \frac{d}{dt}\vec{q}_i(\vec{x},\vec{y}) + \vec{f}_i(\vec{x},\vec{y},\vec{u})$$

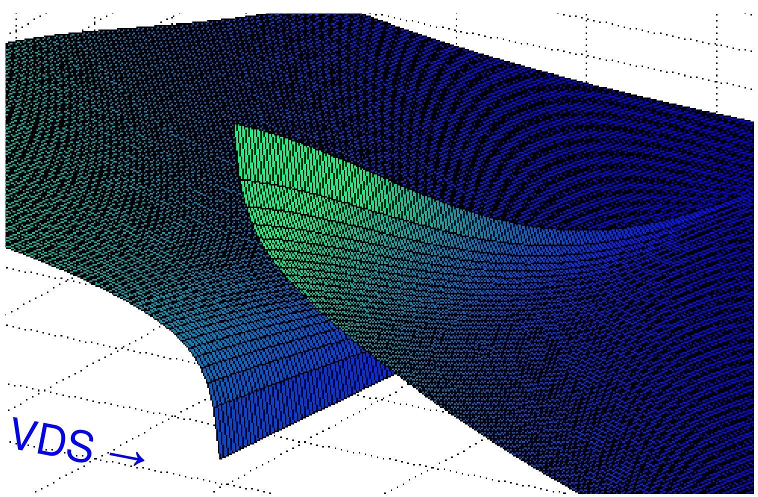
Differential Algebraic Equations

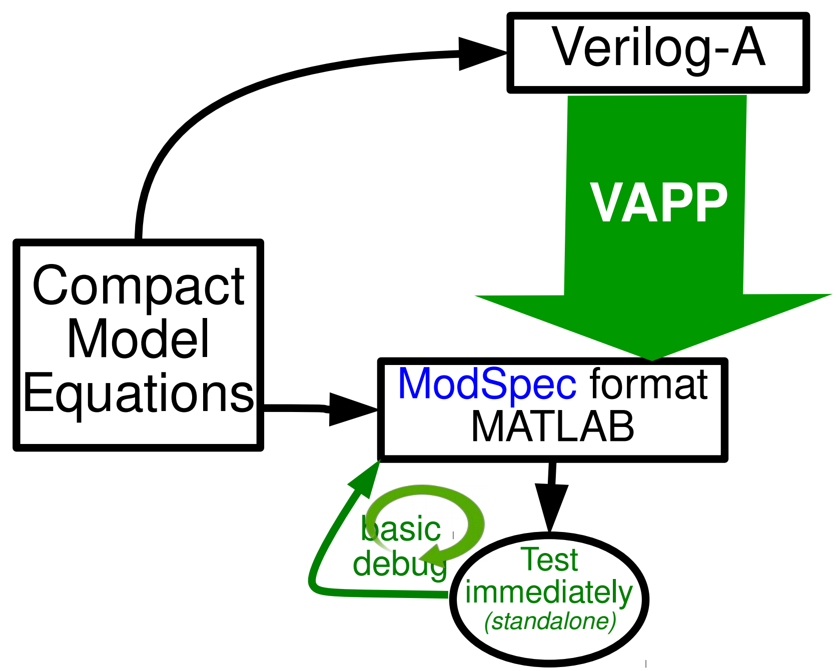
ModSpec: Model Debugging Example

MVS: “notch” in IDS at exactly VDS = zero



MVS: dIDS/dVDS





What's ModSpec: a glimpse

Executable & debuggable standalone	Easy to examine/write by hand	General: any device in any physical domain
Easily & directly usable by any simulator	Supports every analysis DC/AC/tr/PSS	Mathematically well defined, modular

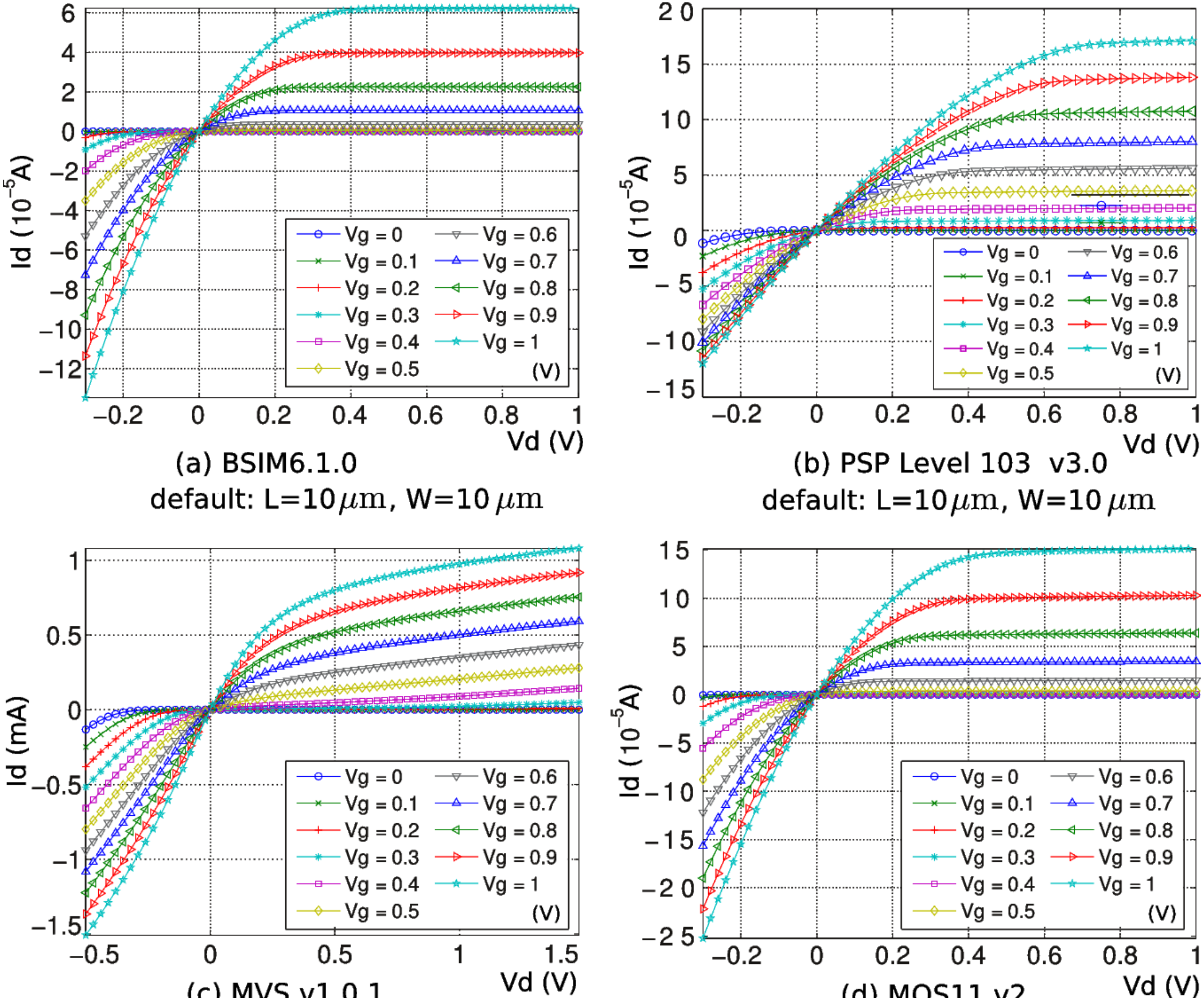
MOD.terminals
MOD.parms
MOD.explicit_outs
MOD.f: function handle
MOD.q: function handle
...

$$\vec{z} = \frac{d}{dt}\vec{q}_e(\vec{x},\vec{y}) + \vec{f}_e(\vec{x},\vec{y},\vec{u})$$

$$\vec{0} = \frac{d}{dt}\vec{q}_i(\vec{x},\vec{y}) + \vec{f}_i(\vec{x},\vec{y},\vec{u})$$

Differential Algebraic Equations

MAPP: Compact Model Prototyping



(a) BSIM6.1.0
default: L=10μm, W=10 μm

(b) PSP Level 103 v3.0
default: L=10μm, W=10 μm

(c) MVS v1.0.1
default: L=80nm, W=1μm

(d) MOS11 v2
default: L=1μm, W=1μm

What's ModSpec: a glimpse

Executable & debuggable standalone

Easy to examine/write by hand

General: any device in any physical domain

Easily & directly usable by any simulator

Supports every analysis DC/AC/tr/PSS

Mathematically well defined, modular

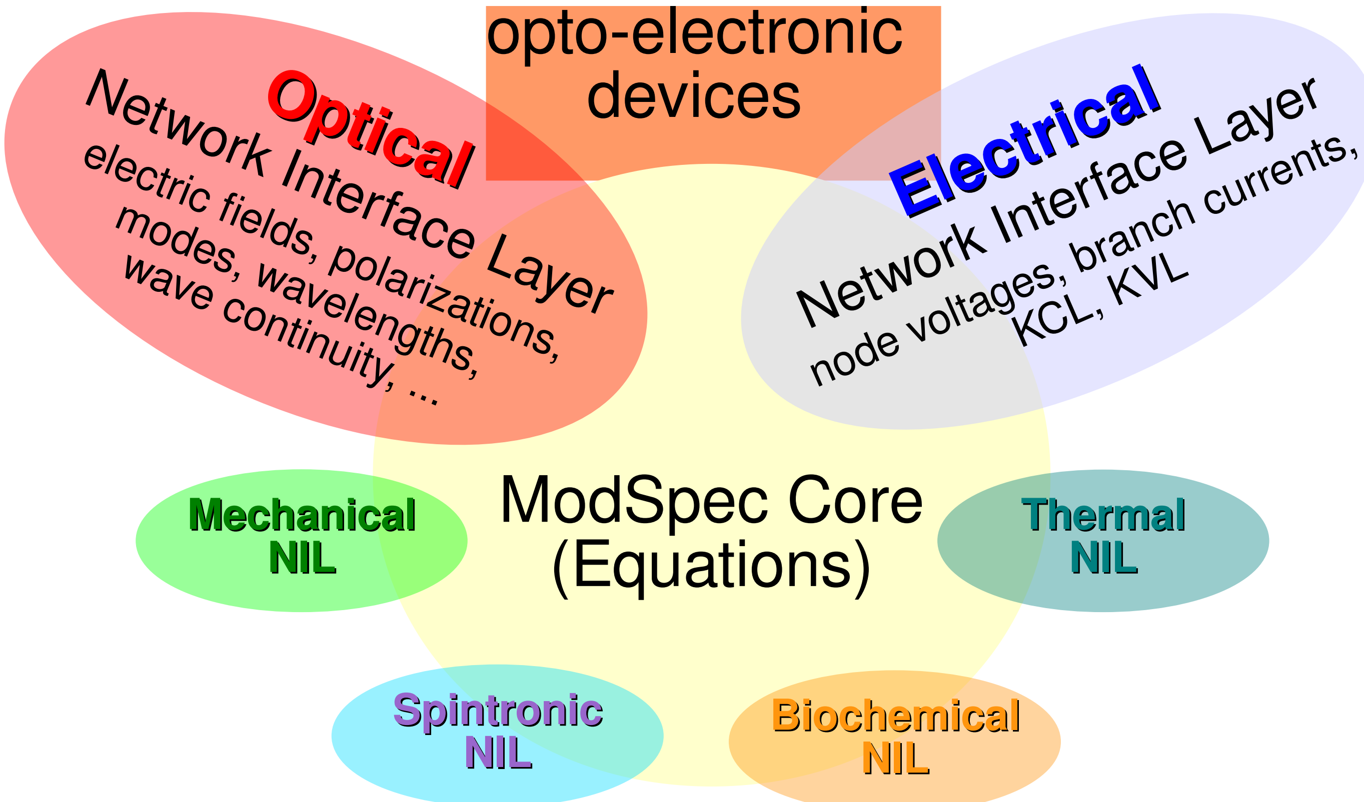
MOD.terminals
MOD.parms
MOD.explicit_outs
MOD.f: function handle
MOD.q: function handle
…

$$\vec{z} = \frac{d}{dt}\vec{q}_e(\vec{x}, \vec{y}) + \vec{f}_e(\vec{x}, \vec{y}, \vec{u})$$

$$\vec{0} = \frac{d}{dt}\vec{q}_i(\vec{x}, \vec{y}) + \vec{f}_i(\vec{x}, \vec{y}, \vec{u})$$

Differential Algebraic Equations

ModSpec: Multiphysics Support



MOD.terminals
MOD.parms
MOD.explicit_outs
MOD.f: function handle
MOD.q: function handle
...

$$\vec{z} = \frac{d}{dt}\vec{q}_e(\vec{x}, \vec{y}) + \vec{f}_e(\vec{x}, \vec{y}, \vec{u})$$

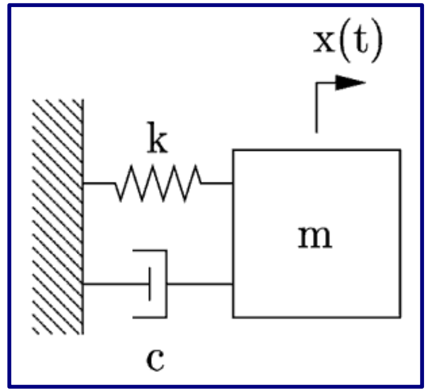
$$\vec{0} = \frac{d}{dt}\vec{q}_i(\vec{x}, \vec{y}) + \vec{f}_i(\vec{x}, \vec{y}, \vec{u})$$

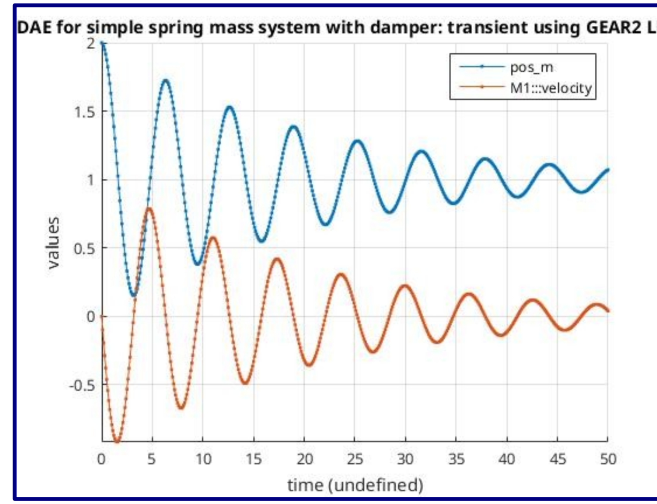
Differential Algebraic Equations

Multiphysics Systems

potential/flow systems:

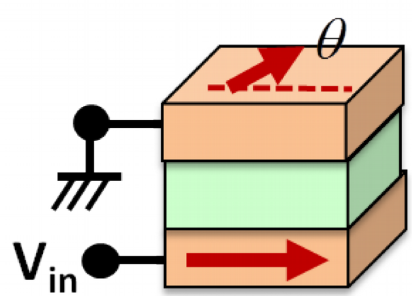
kinematic NIL:
"flow": force
"potential": position





magnetic NIL:
"flow": magnetic flux
"potential": magnetomotive force

thermal NIL:
"flow": power flow
"potential": temperature



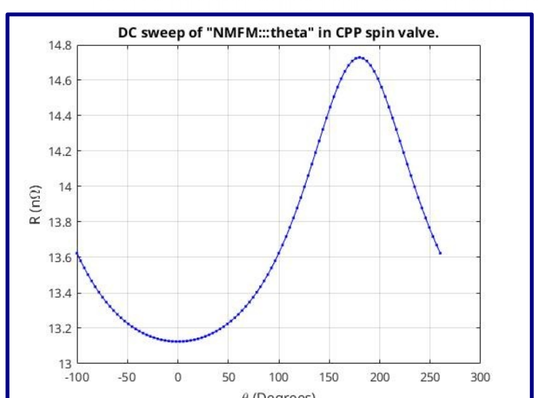
Chemical reaction networks

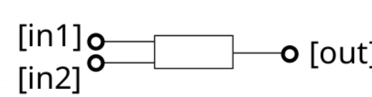
rates **and** ***concentrations***

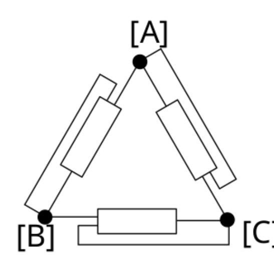
"KCLs" at nodes have d/dt terms

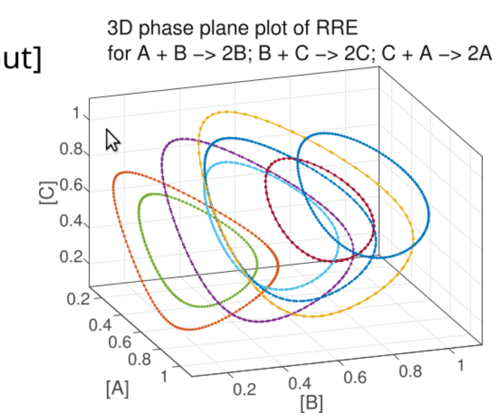
Spintronic systems:

vectorized spin currents
vectorized spin voltages









Kerem Yunus Camsari; Samiran Ganguly; Supriyo Datta (2013), "Modular Spintronics Library," https://nanohub.org/resources/17831.

What's ModSpec: a glimpse

Executable & debuggable standalone	Easy to examine/write by hand	General: any device in any physical domain
Easily & directly usable by any simulator	Supports every analysis DC/AC/tr/PSS	Mathematically well defined, modular

MOD.terminals
MOD.parms
MOD.explicit_outs
MOD.f: function handle
MOD.q: function handle
…

$$\vec{z} = \frac{d}{dt}\vec{q}_e(\vec{x},\vec{y}) + \vec{f}_e(\vec{x},\vec{y},\vec{u})$$

$$\vec{0} = \frac{d}{dt}\vec{q}_i(\vec{x},\vec{y}) + \vec{f}_i(\vec{x},\vec{y},\vec{u})$$

Differential Algebraic Equations

Glimpse: ModSpec Model in Xyce

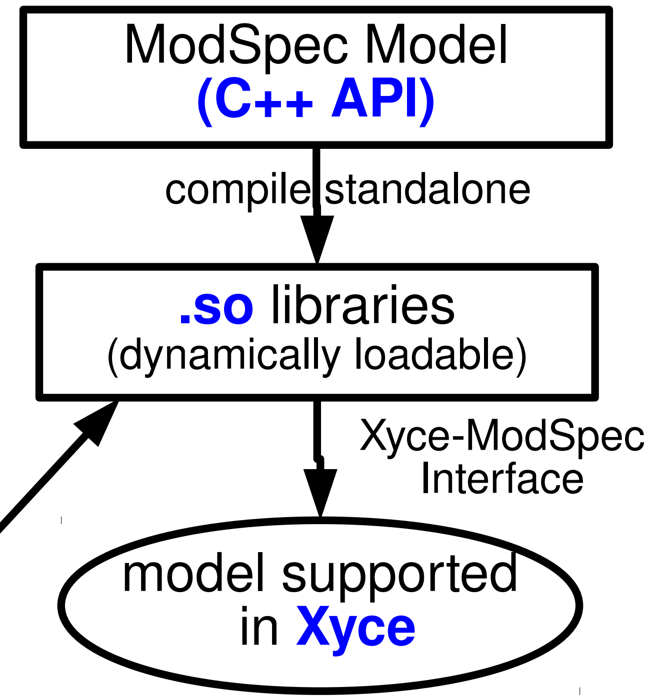
```
*** Test-bench for generting dc response of an inverter

*** Creat sub-circuit for the inverter
.subckt inverter Vin Vout Vvdd Vgnd

yModSpec_Device X1 Vvdd Vin Vout Vvdd MVSmod type=-1 W=1.0e-4
   Lgdr=32e-7 dLg=8e-7 Cg=2.57e-6 beta=1.8 alpha=3.5 Tjun=300
   Cif = 1.38e-12 Cof=1.47e-12 phib=1.2 gamma=0.1 mc=0.2
   CTM_select=1  Rs0=100 Rd0 = 100 n0=1.68 nd=0.1 vxo=7542204
   mu=165 Vt0=0.5535 delta=0.15

yModSpec_Device X0 Vout Vin Vgnd Vgnd MVSmod type=1 W=1e-4
   Lgdr=32e-7 dLg=9e-7 Cg=2.57e-6 beta=1.8 alpha=3.5 Tjun=300
   Cif=1.38e-12 Cof=1.47e-12 phib=1.2 gamma=0.1 mc=0.2
   CTM_select=1 Rs0=100 Rd0=100 n0=1.68 nd=0.1 vxo=1.2e7
   mu=200 Vt0=0.4 delta=0.15

.model MVSmod MODSPEC_DEVICE SONAME=MVS_ModSpec_Element.so

.ends

*** circuit layout
Vsup sup 0 1
Vin in 0 0
Vsource source 0 0
X2  in out sup 0 inverter

*** simulation
.dc Vin 0 1 0.01

.print dc V(in) V(out)
*** END
.end
```

.model line

model's name

model parameter: name of .so library

Xyce netlist for inverter (using MVS ModSpec/C++ model)



Updates in the last year

- **limiting correction**
- **composite parameters**
- **works in Xyce 6.5**

What's ModSpec: a glimpse

Executable & debuggable standalone

Easy to examine/write by hand

General: any device in any physical domain

Easily & directly usable by any simulator

Supports every analysis DC/AC/tr/PSS

Mathematically well defined, modular

MOD.terminals
MOD.parms
MOD.explicit_outs
MOD.f: function handle
MOD.q: function handle
...

$$\vec{z} = \frac{d}{dt}\vec{q}_e(\vec{x}, \vec{y}) + \vec{f}_e(\vec{x}, \vec{y}, \vec{u})$$

$$\vec{0} = \frac{d}{dt}\vec{q}_i(\vec{x}, \vec{y}) + \vec{f}_i(\vec{x}, \vec{y}, \vec{u})$$

Differential Algebraic Equations

STEAM: Fast, Accurate Table-Based Models

- Compact model using only tabulated i-v, q-v data?
  - » previous table-based attempts: important details unclear, poor accuracy, low speedup
  - » our goal: can we speed up existing compact models?
- Our approach: STEAM
  - » tabulate ModSpec functions fe, fi, qe, qi (one time cost)
  - » device eval: multi-dimensional cubic spline interpolation
- Initial results
  - » 150x eval speedup for BSIM3 (6-15x tran/DC)
  - » relative error as low as you like: eg, 10^{-4}
    - – but memory requirements grow with accuracy

replace with "lookup" tables

implementation details:
multi-dimensional splines,
passive extrapolation

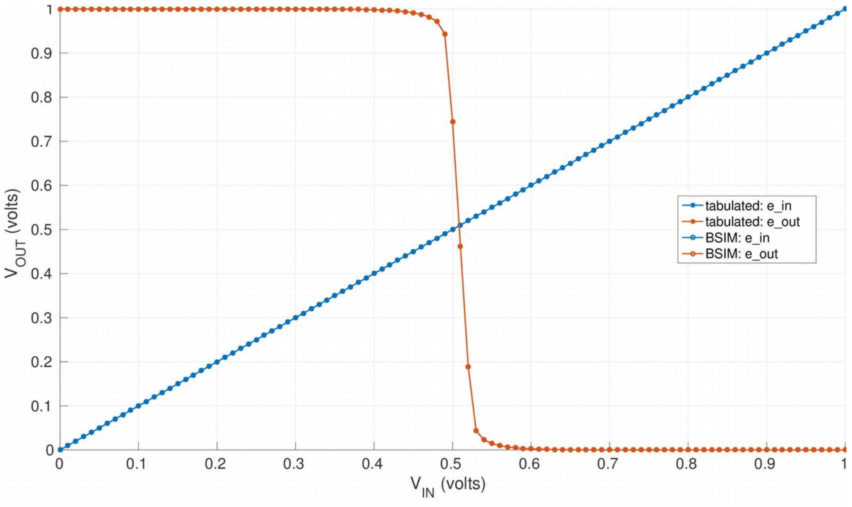
$$\vec{z} = \frac{d}{dt}\vec{q}_e(\vec{x},\vec{y}) + \vec{f}_e(\vec{x},\vec{y},\vec{u})$$

$$\vec{0} = \frac{d}{dt}\vec{q}_i(\vec{x},\vec{y}) + \vec{f}_i(\vec{x},\vec{y},\vec{u})$$

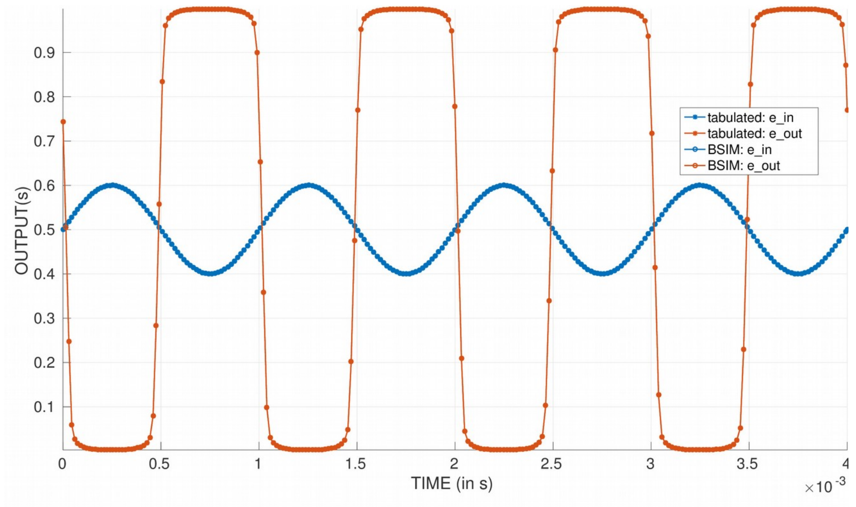
Differential Algebraic Equations

BSIM3 Inverter: STEAM vs Original

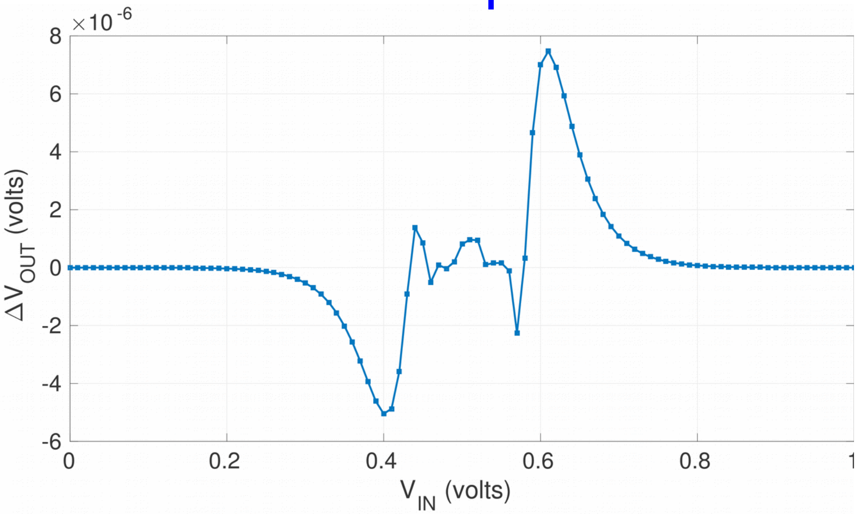
DC sweep



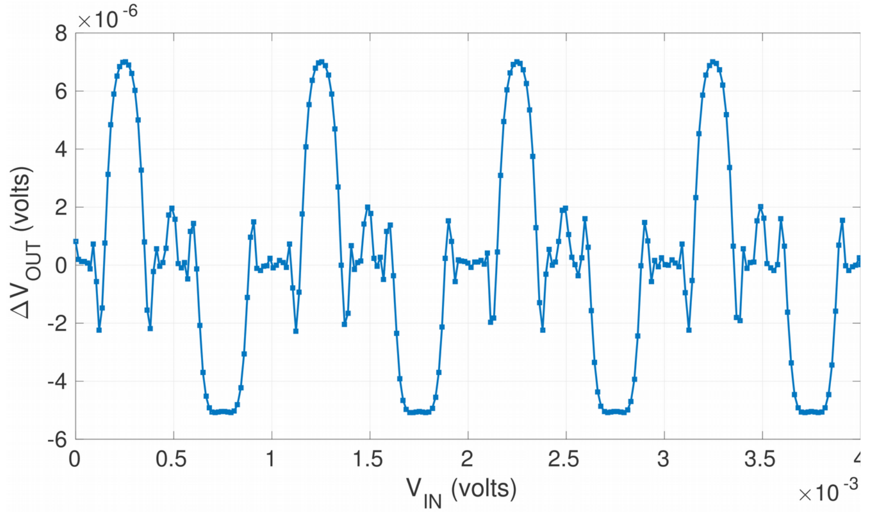
Transient



DC sweep: error



Transient: error



What's ModSpec: a glimpse

Executable & debuggable standalone

Easy to examine/write by hand

General: any device in any physical domain

Easily & directly usable by any simulator

Supports every analysis DC/AC/tr/PSS

Mathematically well defined, modular

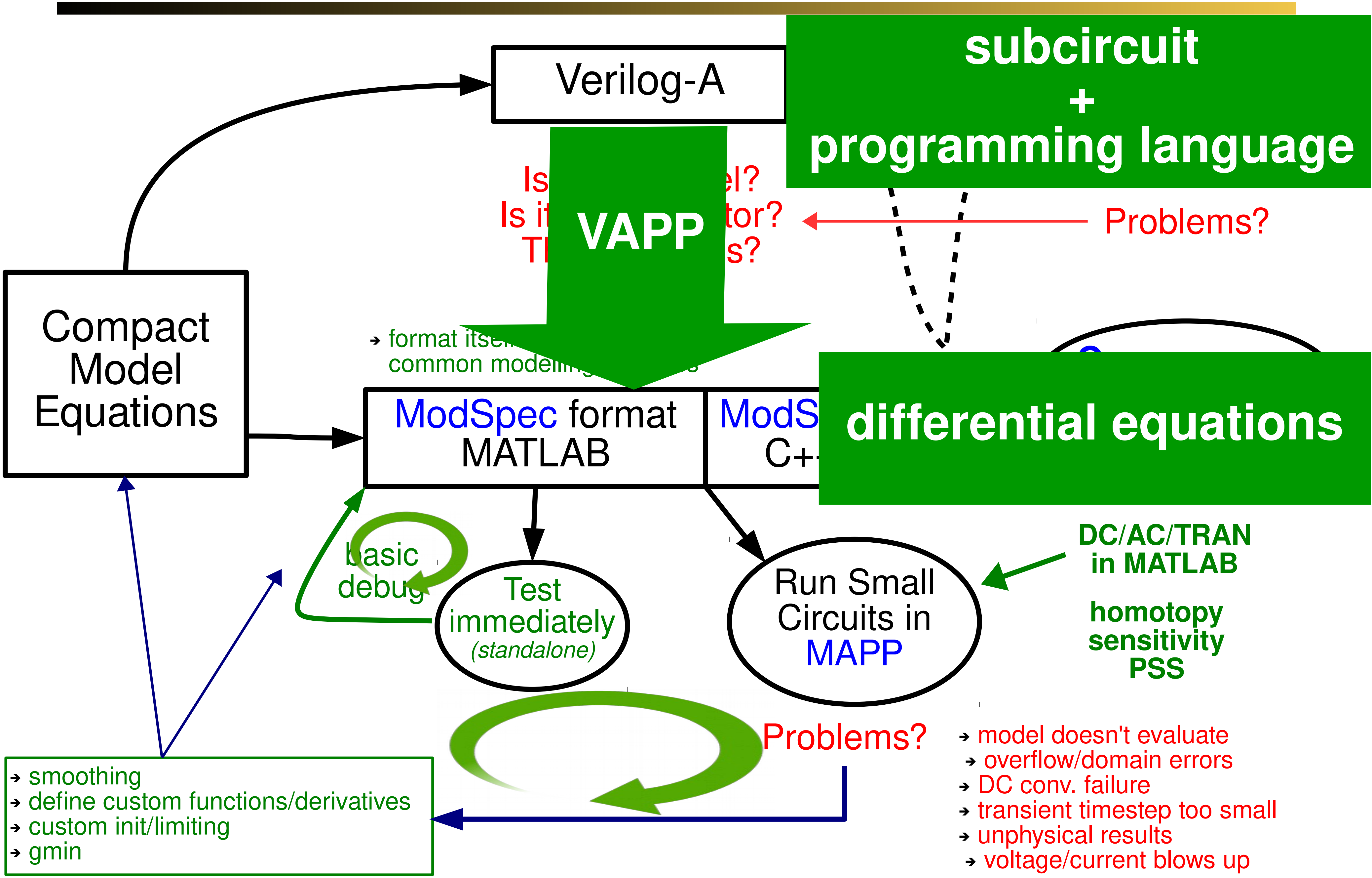
MOD.terminals
MOD.parms
MOD.explicit_outs
MOD.f: function handle
MOD.q: function handle
...

$$\vec{z} = \frac{d}{dt}\vec{q}_e(\vec{x}, \vec{y}) + \vec{f}_e(\vec{x}, \vec{y}, \vec{u})$$

$$\vec{0} = \frac{d}{dt}\vec{q}_i(\vec{x}, \vec{y}) + \vec{f}_i(\vec{x}, \vec{y}, \vec{u})$$

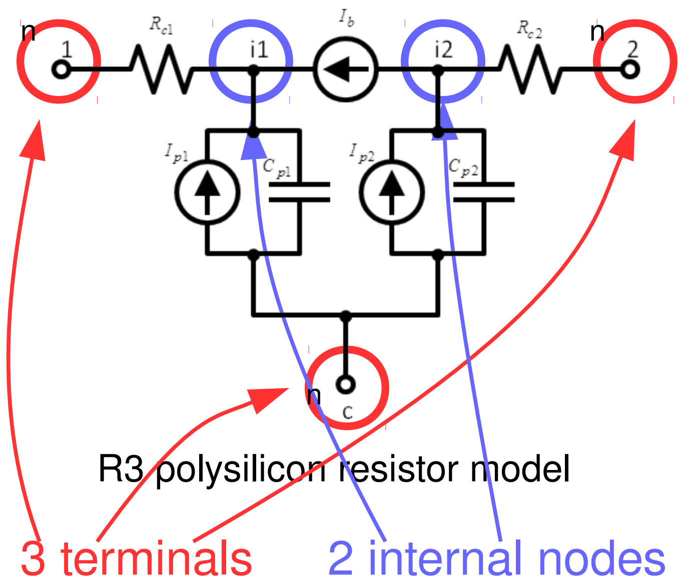
Differential Algebraic Equations

Compact Model Development

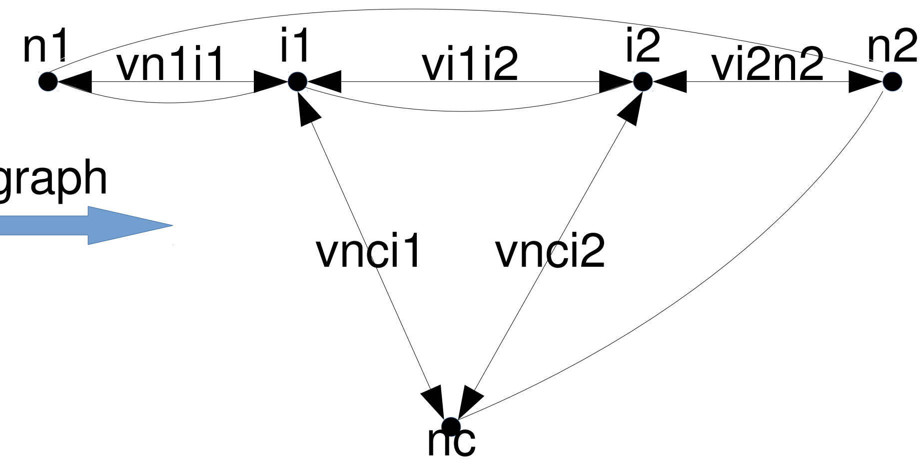


T. Wang, UC Berkeley

VAPP: New Graph Based Core



Convert to a graph



How do we know that {vn1i1, vi1i2} are internal unknowns?

And {vnci1, vnci2} dependent voltages?

Algorithm:

- Construct a spanning tree (ST)
- Designate branches in the ST as independent voltages
- Remaining branches are independent currents
- Construct loop and cutset matrices
- Express dependent quantities in terms of independent ones

VAPP: What Is Still Lacking?

- Node collapse:

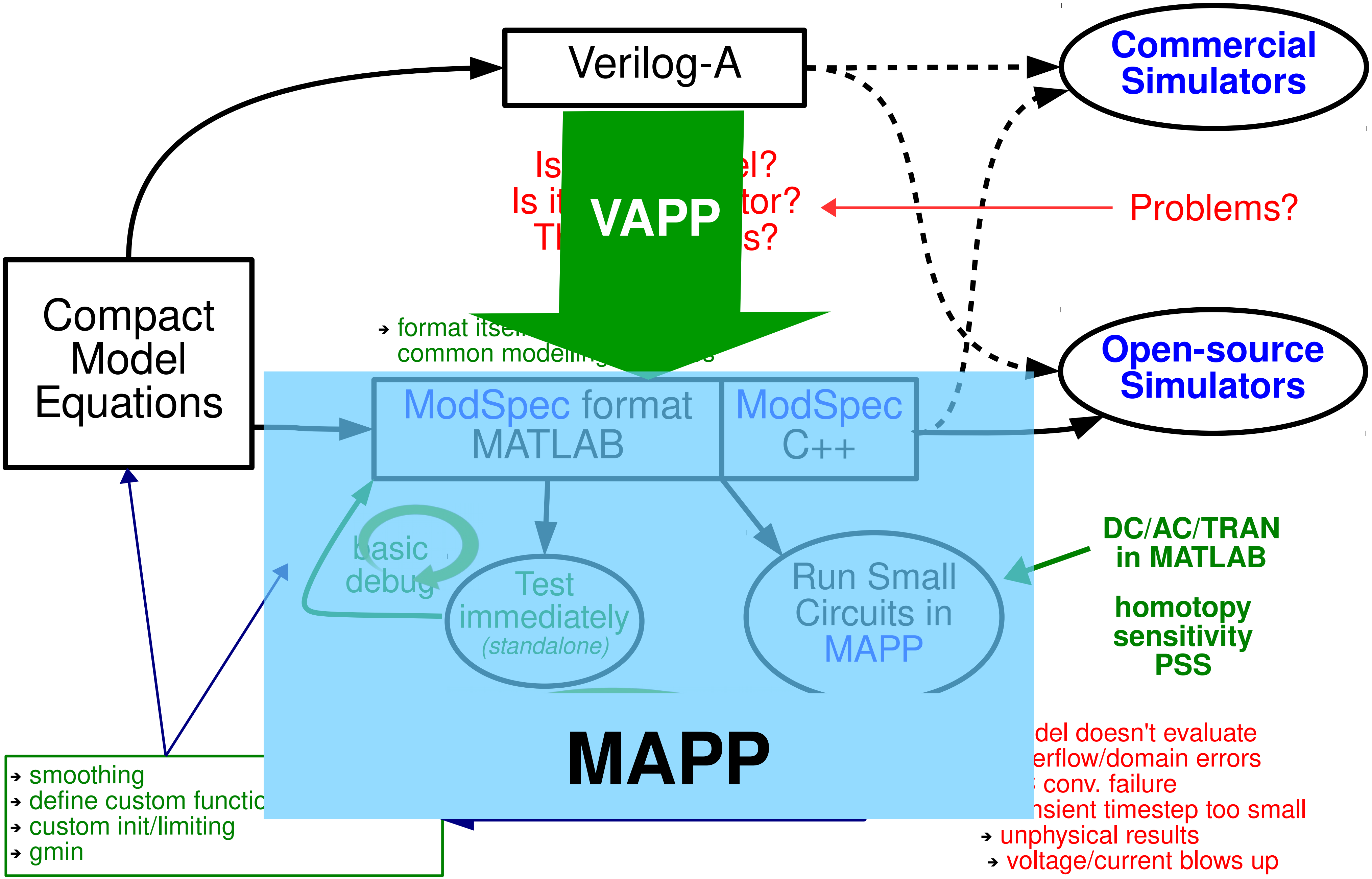
```
if (rdsmod == 0)
  begin
    V(source, sourcep) <+ 0;
    V(drainp, drain)   <+ 0;
  end
else
  begin
    I(drain, drainp)   <+ type * gdtot * vded;
    I(source, sourcep) <+ type * gstot * vses;
  end
```

⇨ Changes the number of unknowns.

- Separate networks (graphs) for different disciplines. E.g., `thermal`, `magnetic` ,...
  Important for self heating.

- Support for noise functions in MAPP. E.g., `white_noise`, `flicker_noise`

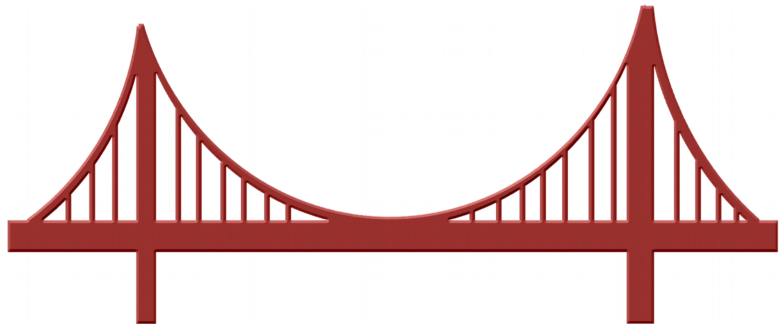
Compact Model Development



Memristive Devices & Applications

devices

UMich, Stanford,
HP, HRL Labs,
Micron, Crossbar,
Samsung, ...



applications

- nonvolatile memories
- FPAAs
- neuromorphic circuits
- oscillators

Knowm

Compact Models

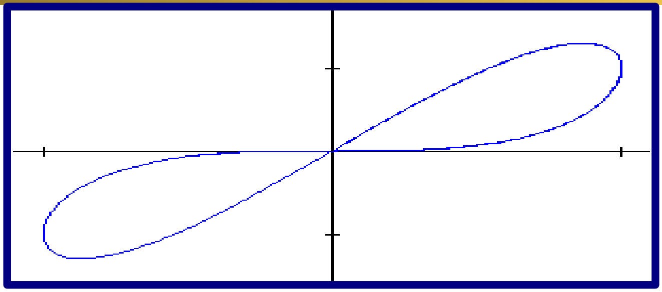
- Linear/nonlinear ion drift models
  Biolek (2009), Joglekar (2009),
  Prodromakis (2011), etc.
- UMich RRAM model (2011)
- TEAM model (2012)
- Simmons tunneling barrier model (2013)
- Yakopcic model (2013)
- Stanford/ASU RRAM model (2014)
- Knowm "probabilistic" model (2015)

not one works in DC

Verilog-A problems

idt(), $bound_step,
$abstime, @initial_step,
$rdist_normal, ...

Challenges in Memristor Modelling



- hysteresis
  - internal state variable
- model internal unks in Verilog-A
  - use potentials/flows
- upper/lower bounds of internal unks
  - filament length, tunneling tap size
  - clipping functions
- smoothness, continuity, finite precision issues, ...
  - use smooth functions, safe functions
  - GMIN
  - scaling of unks/eqns
  - SPICE-compatible limiting function (the only smooth one)

How to Model Hysteresis Properly

Template:

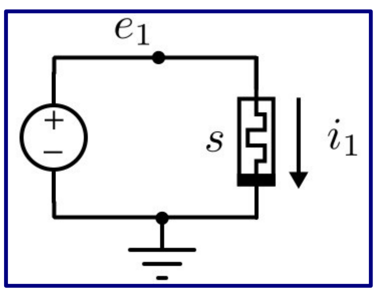
$$\mathbf{ipn} = f_1(\mathbf{vpn},\ \mathbf{s})$$

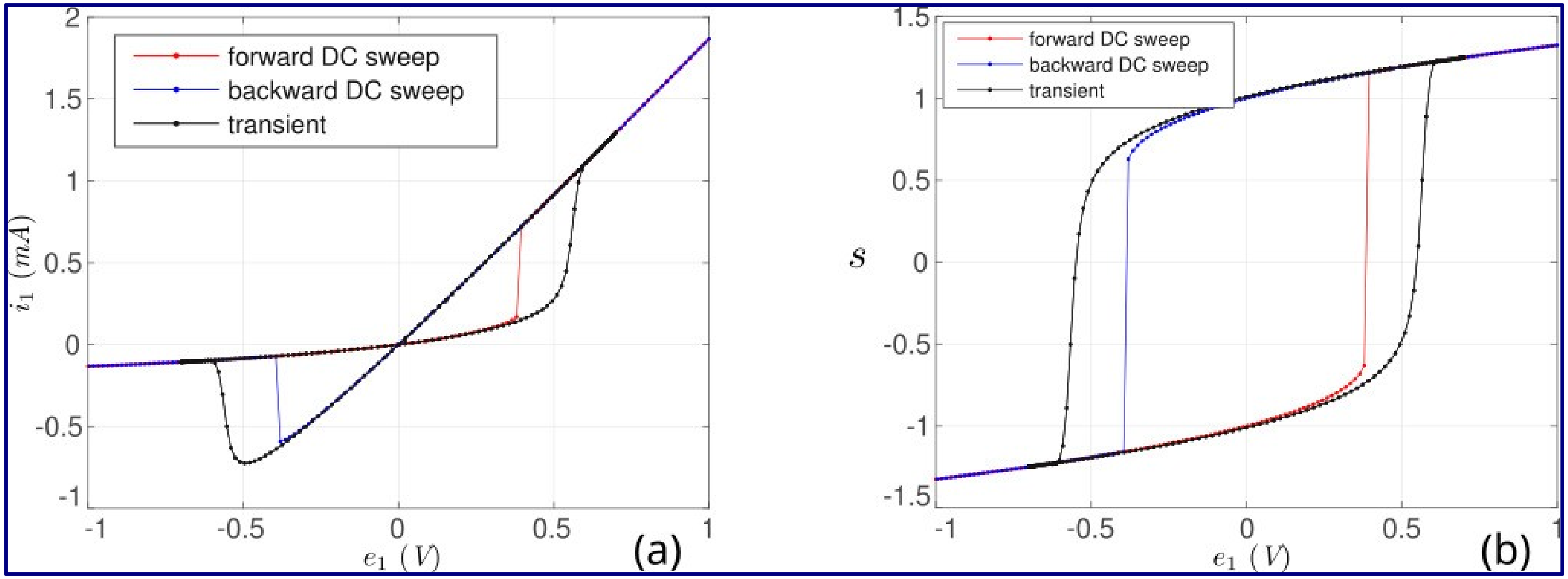
$$\frac{d}{dt}\mathbf{s} = f_2(\mathbf{vpn},\ \mathbf{s})$$

ModSpec:

$$\mathbf{ipn} = \frac{d}{dt}\underbrace{q_e(\mathbf{vpn},\ \mathbf{s})}_{\mathbf{0}} + \underbrace{f_e(\mathbf{vpn},\ \mathbf{s})}_{f_1}$$

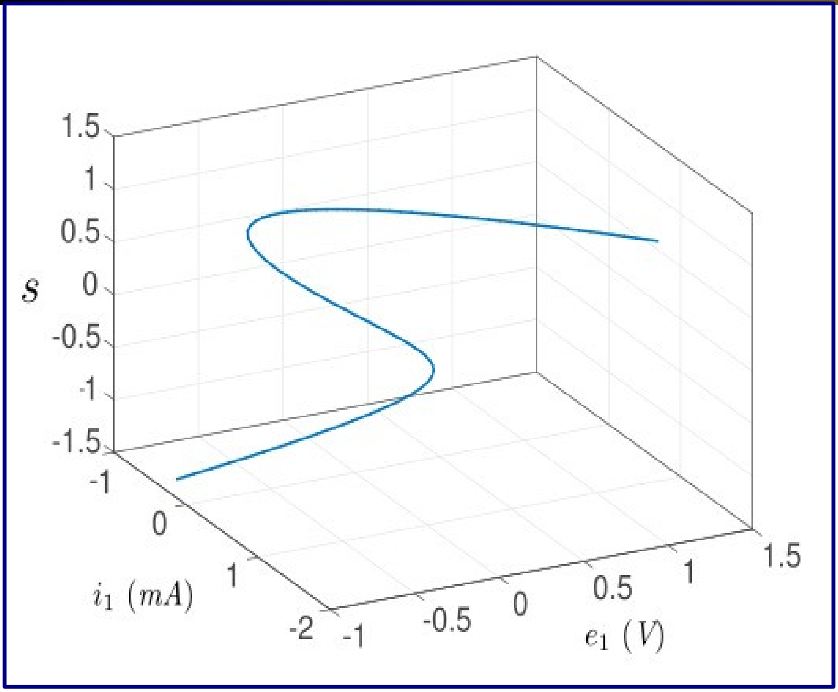
$$0 = \frac{d}{dt}\underbrace{q_i(\mathbf{vpn},\ \mathbf{s})}_{-\mathbf{s}} + \underbrace{f_i(\mathbf{vpn},\ \mathbf{s})}_{f_2}$$





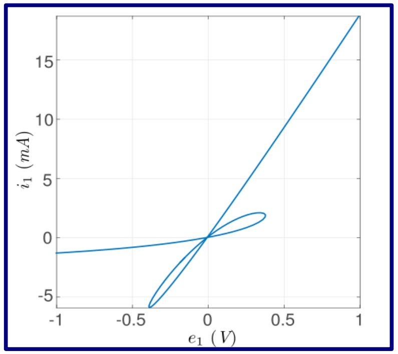
T. Wang, UC Berkeley

How to Model Hysteresis Properly

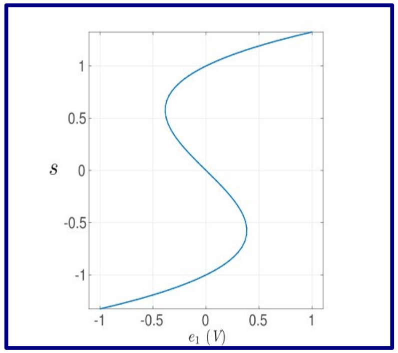


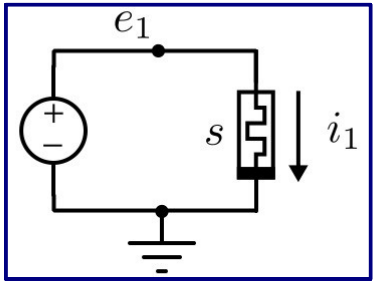
homotopy

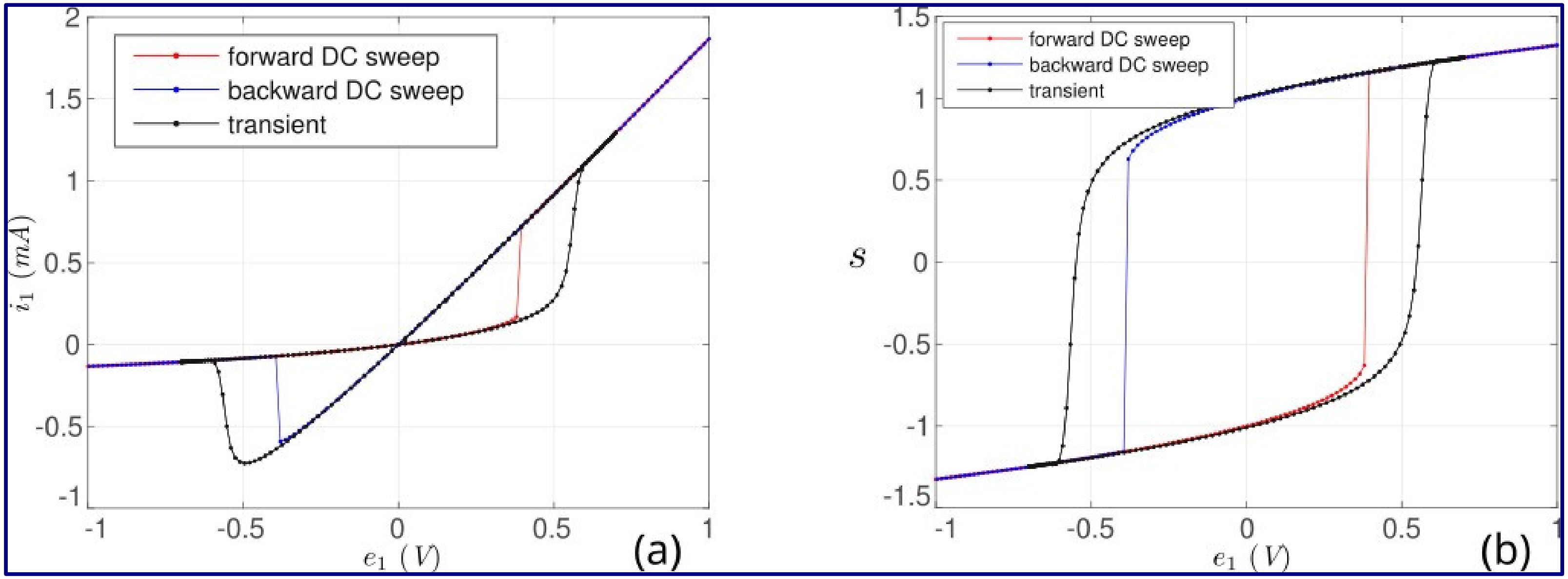
top



side







T. Wang, UC Berkeley

Memristor Models

$$\frac{d}{dt}\mathbf{s} = f_2\left(\mathbf{vpn},\ \mathbf{s}\right)$$

Available f2:

1. linear ion drift

$$f_2 = \mu_v \cdot R_{on} \cdot f_1(\mathbf{vpn},\ s)$$

2. nonlinear ion drift

$$f_2 = a \cdot \mathbf{vpn}^m$$

3. Simmons tunnelling barrier

$$f_2 = \begin{cases} c_{off} \cdot \sinh(\frac{i}{i_{off}}) \cdot \exp(-\exp(\frac{s-a_{off}}{w_c} - \frac{i}{b}) - \frac{s}{w_c}), & \text{if } i \geq 0 \\ c_{on} \cdot \sinh(\frac{i}{i_{on}}) \cdot \exp(-\exp(\frac{a_{on}-s}{w_c} + \frac{i}{b}) - \frac{s}{w_c}), & \text{otherwise,} \end{cases}$$

4. TEAM model

5. Yakopcic model

6. Stanford/ASU

$$f_2 = -v_0 \cdot \exp(-\frac{E_a}{V_T}) \cdot \sinh(\frac{\mathbf{vpn} \cdot \gamma \cdot a_0}{t_{ox} \cdot V_T})$$

$$\mathbf{ipn} = f_1\left(\mathbf{vpn},\ \mathbf{s}\right)$$

Available f1:

1. $$f_1 = (R_{on} \cdot s + R_{off} \cdot (1-s))^{-1} \cdot \mathbf{vpn}$$

2. $$f_1 = \frac{1}{R_{on}} \cdot e^{-\lambda \cdot (1-s)} \cdot \mathbf{vpn}$$

3. $$f_1 = s^n \cdot \beta \cdot \sinh(\alpha \cdot \mathbf{vpn}) + \chi \cdot (\exp(\gamma \cdot) - 1)$$

4. $$\mathrm{f}_1 = \begin{cases} A_1 \cdot s \cdot \sinh(B \cdot \mathbf{vpn}), & \text{if } \mathbf{vpn} \geq \mathbf{0} \\ A_2 \cdot s \cdot \sinh(B \cdot \mathbf{vpn}), & \text{otherwise.} \end{cases}$$

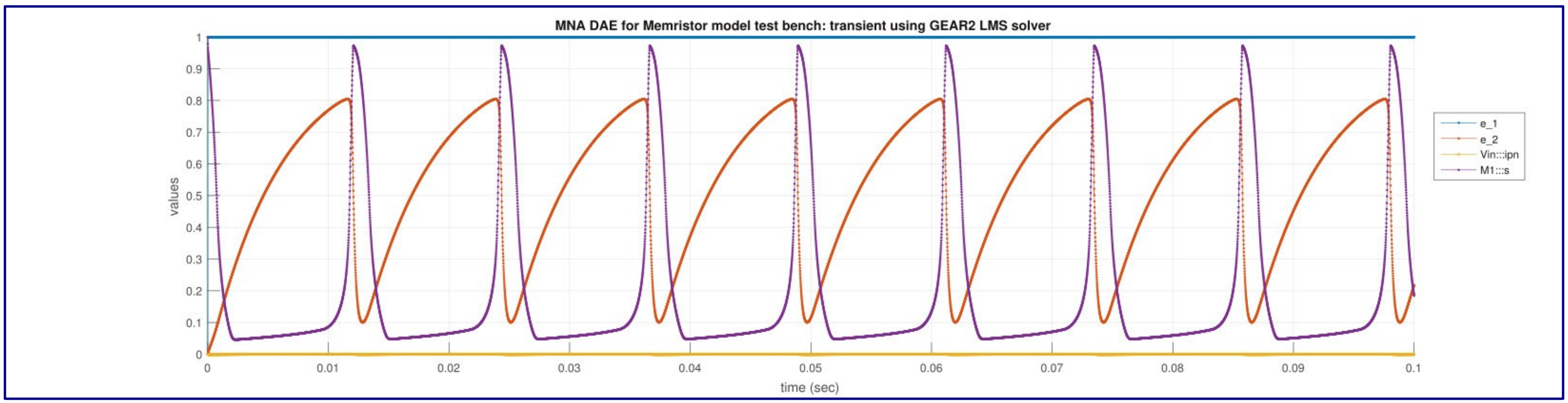
5. $$f_1 = I_0 \cdot e^{-\mathbf{Gap}/g0} \cdot \sinh(\mathbf{vpn}/V_0)$$
$$\mathbf{Gap} = s \cdot minGap + (1-s) \cdot maxGap.$$

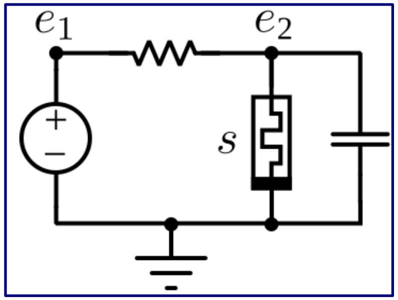
- **set up boundary**
- **fix f2 flat regions**
- **smooth, safe funcs, scaling, etc.**

Memristor Models

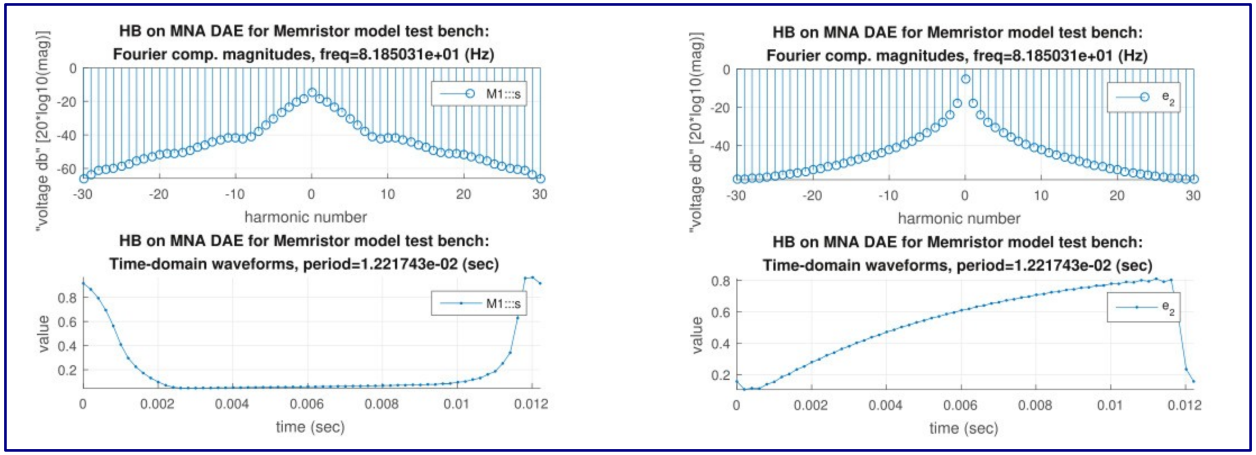
A collection of 30 models:

- all smooth, all well posed
- not just RRAM, but general memristive devices
- not just bipolar, but unipolar
- not just DC, AC, TRAN, but homotopy, PSS, ...



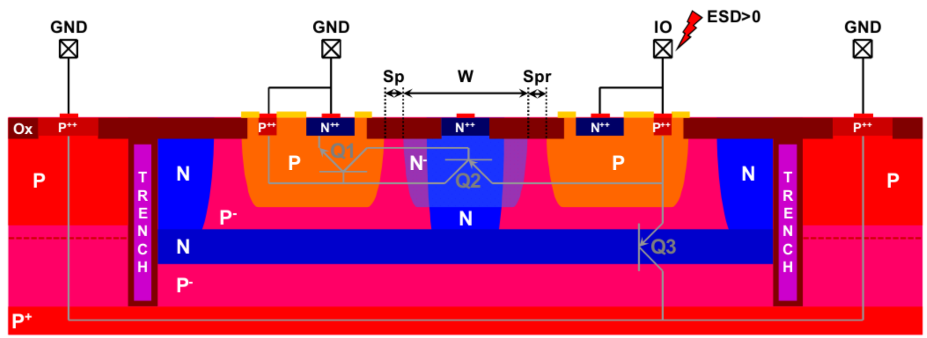


PSS using HB

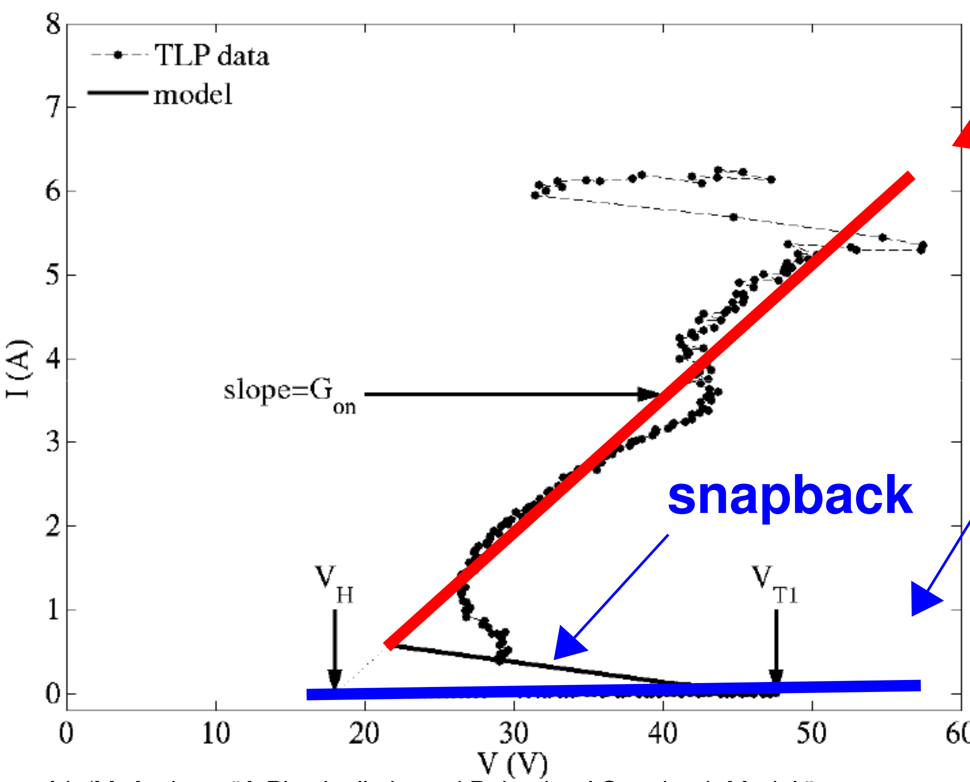


ESD Snapback Model

ESD protection device



Gendron, et al. "New High Voltage ESD Protection Devices based on Bipolar Transistors for Automotive Applications." IEEE EOS/ESD Symposium, 2011.



Ida/McAndrew. "A Physically-based Behavioral Snapback Model." IEEE EOS/ESD Symposium, 2012.

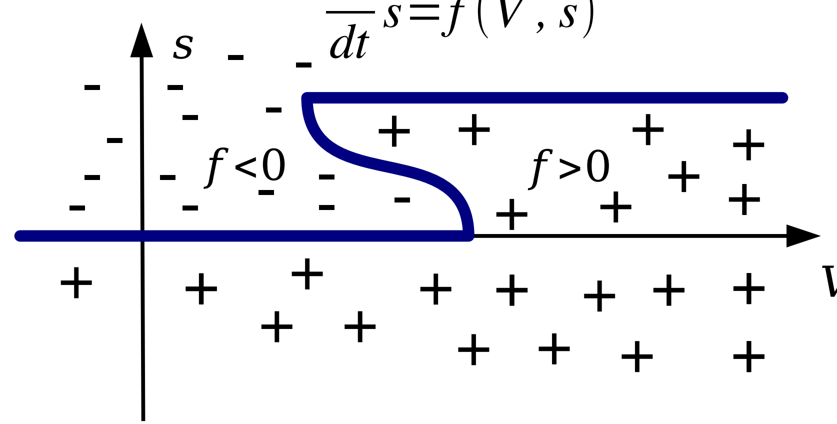
$$I_{on} = G_{on} \cdot (V - V_H)$$

$$I_{off} = I_S \cdot (1 - e^{-V/\phi_T}) \cdot \sqrt{1 + \frac{max(V, 0)}{V_D}}$$

$$I = s \cdot I_{on} + I_{off}$$

internal state: indicator of impact ionization

$$\frac{d}{dt} s = f(V, s)$$



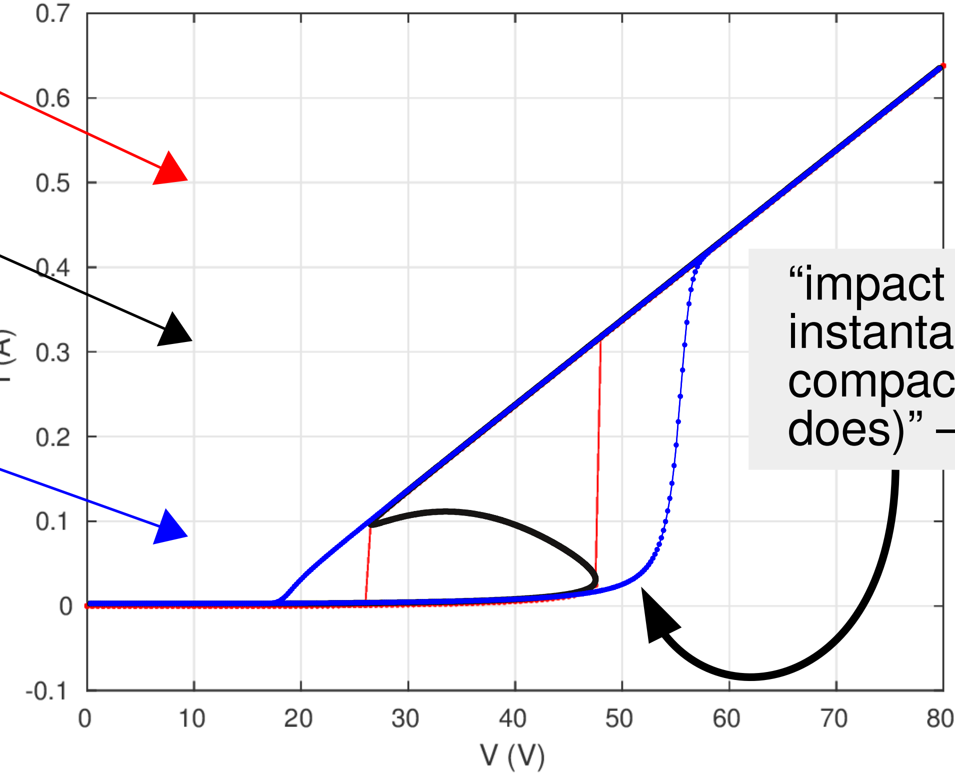
T. Wang, UC Berkeley

ESD Snapback Model

forward/backward DC sweeps

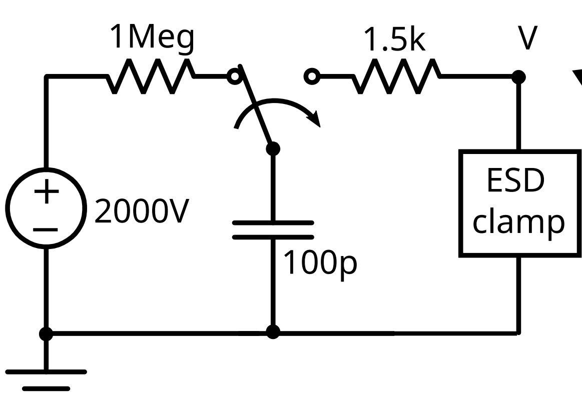
homotopy

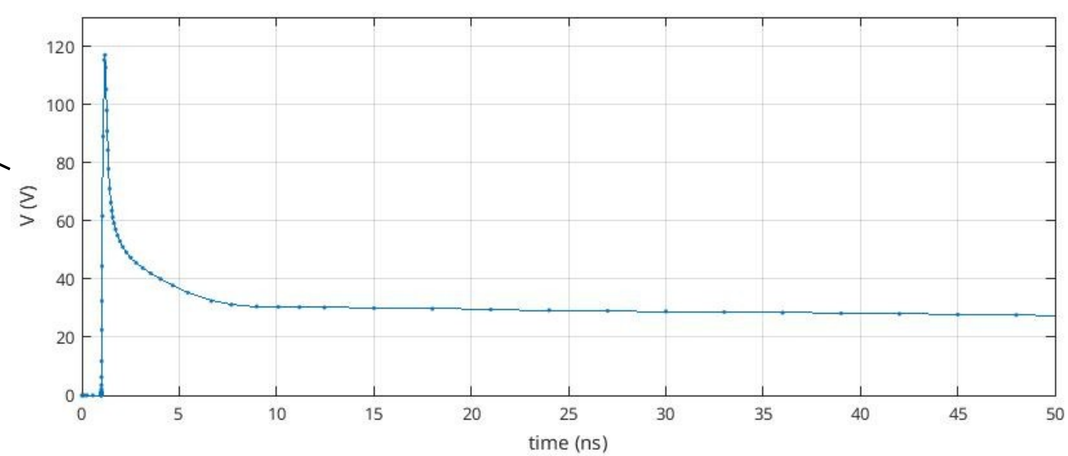
transient voltage sweeps



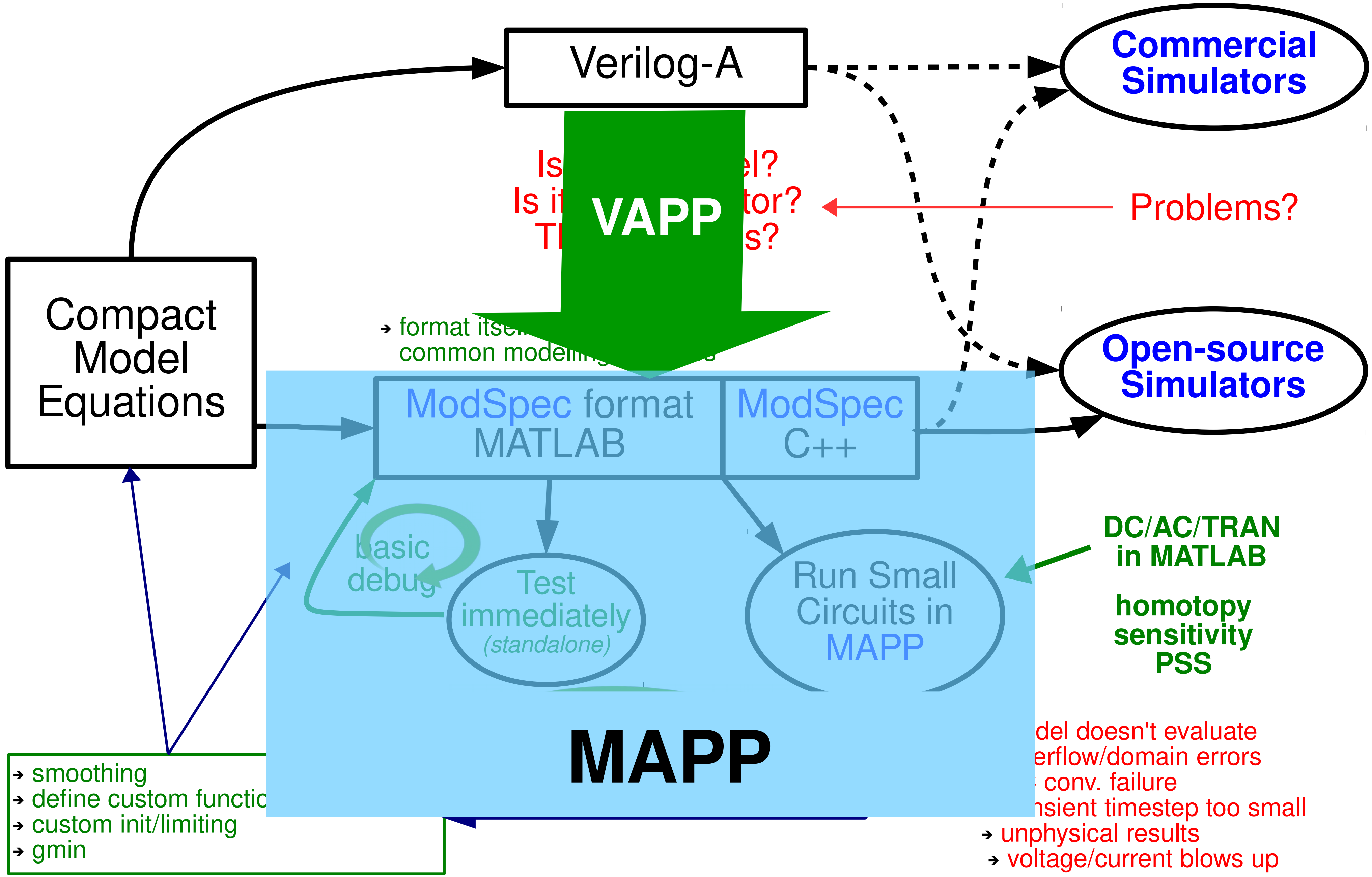
"impact ionization doesn't happen instantaneously (although all compact models assume that it does)" –- C.C. McAndrew

Human Body Mode (HBM) test





Compact Model Development



T. Wang, UC Berkeley