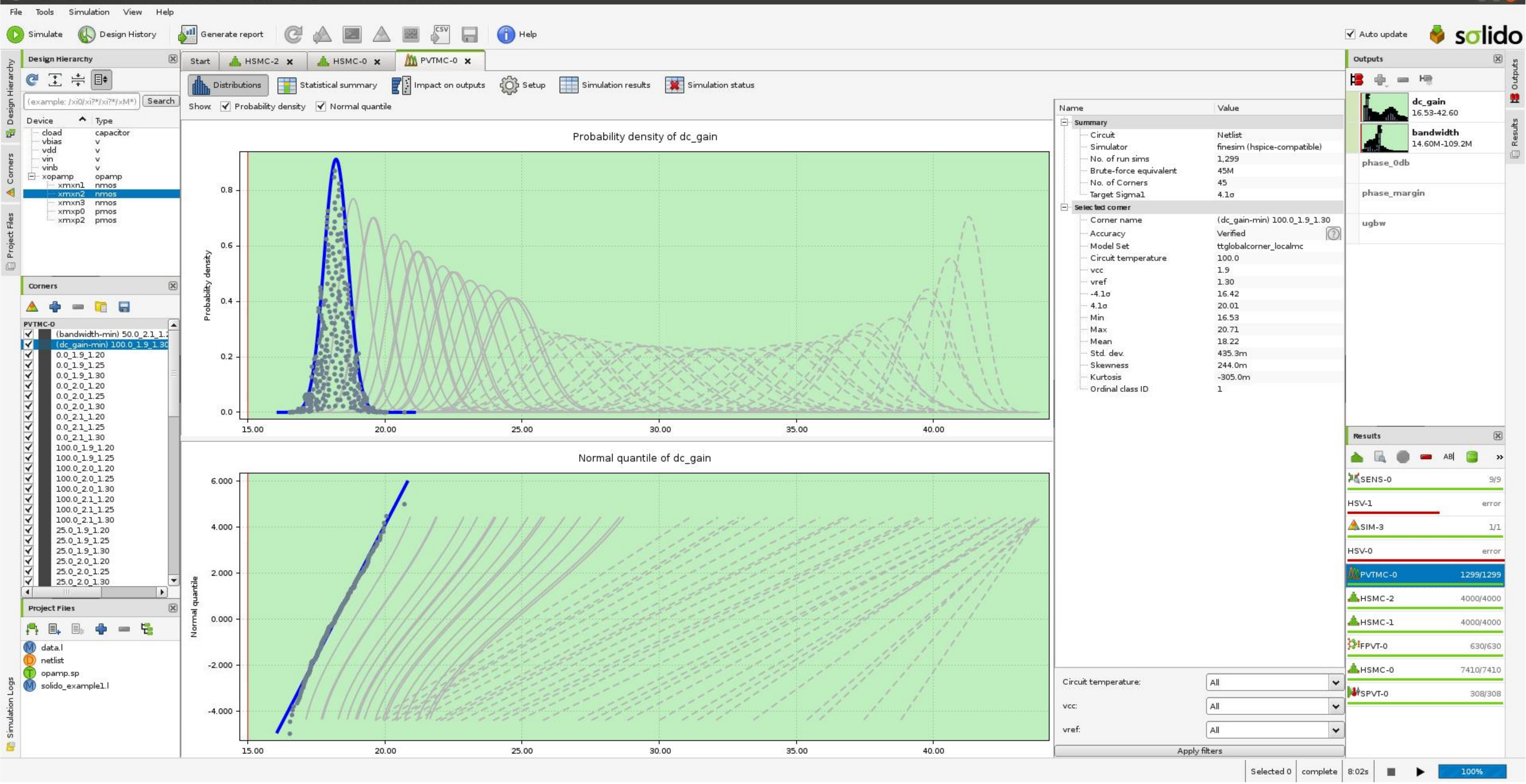
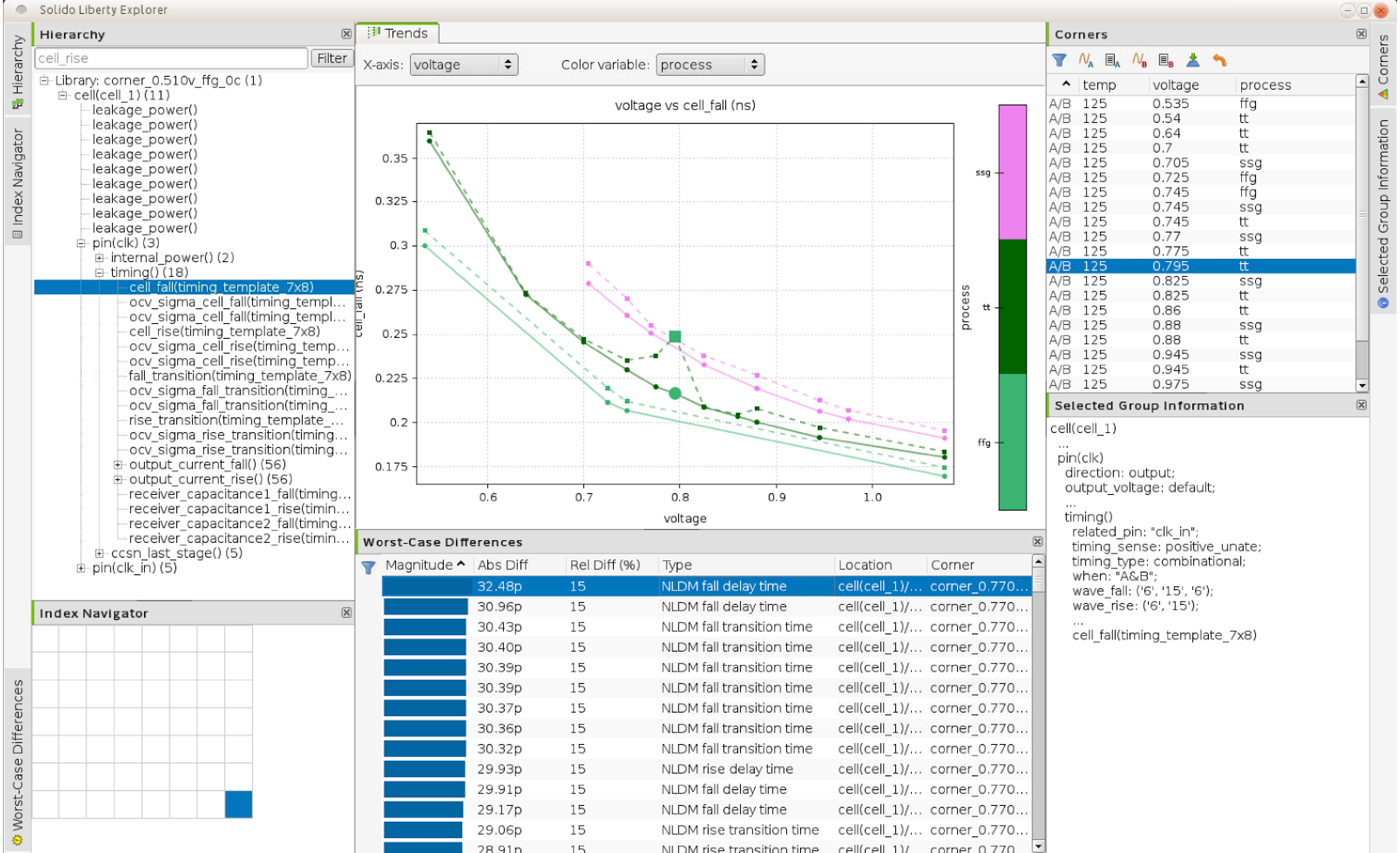
Getting The Right Answer with Machine Learning

IEEE-EDPS 2021

Jeff Dyck

SIEMENS


File Tools Simulation View Help
Simulate
Design History
Generate report
Help
Auto update
solido
Design Hierarchy
Device
Type
cload capacitor
vbias v
vdd v
vin v
vinb v
xopamp opamp
xmxn1 nmos
xmxn2 nmos
xmxn3 nmos
xmxp0 pmos
xmxp2 pmos
Corners
PVTMC-0
(bandwidth-min) 50.0_2.1_1.2
(dc_gain-min) 100.0_1.9_1.30
0.0_1.9_1.20
0.0_1.9_1.25
0.0_1.9_1.30
0.0_2.0_1.20
0.0_2.0_1.25
0.0_2.0_1.30
0.0_2.1_1.20
0.0_2.1_1.25
0.0_2.1_1.30
100.0_1.9_1.20
100.0_1.9_1.25
100.0_2.0_1.20
100.0_2.0_1.25
100.0_2.0_1.30
100.0_2.1_1.20
100.0_2.1_1.25
100.0_2.1_1.30
25.0_1.9_1.20
25.0_1.9_1.25
25.0_1.9_1.30
25.0_2.0_1.20
25.0_2.0_1.25
25.0_2.0_1.30
Project Files
data.l
netlist
opamp.sp
solido_example1.l
Start
HSMC-2
HSMC-0
PVTMC-0
Distributions
Statistical summary
Impact on outputs
Setup
Simulation results
Simulation status
Show: Probability density Normal quantile
Probability density of dc_gain
Probability density
Normal quantile of dc_gain
Normal quantile
Name Value
Summary
Circuit Netlist
Simulator finesim (hspice-compatible)
No. of run sims 1,299
Brute-force equivalent 45M
No. of Corners 45
Target Sigma1 4.1σ
Selected corner
Corner name (dc_gain-min) 100.0_1.9_1.30
Accuracy Verified
Model Set ttglobalcorner_localmc
Circuit temperature 100.0
vcc 1.9
vref 1.30
-4.1σ 16.42
4.1σ 20.01
Min 16.53
Max 20.71
Mean 18.22
Std. dev. 435.3m
Skewness 244.0m
Kurtosis -305.0m
Ordinal class ID 1
Circuit temperature: All
vcc: All
vref: All
Apply filters
Outputs
dc_gain 16.53-42.60
bandwidth 14.60M-109.2M
phase_0db
phase_margin
ugbw
Results
SENS-0 9/9
HSV-1 error
SIM-3 1/1
HSV-0 error
PVTMC-0 1299/1299
HSMC-2 4000/4000
HSMC-1 4000/4000
FPVT-0 630/630
HSMC-0 7410/7410
SPVT-0 308/308
Selected 0
complete
8:02s
100%


Solido Liberty Explorer
Hierarchy
Index Navigator
Worst-Case Differences
cell_rise
Filter
Library: corner_0.510v_ffg_0c (1)
cell(cell_1) (11)
leakage_power()
leakage_power()
leakage_power()
leakage_power()
leakage_power()
leakage_power()
leakage_power()
leakage_power()
leakage_power()
pin(clk) (3)
internal_power() (2)
timing() (18)
cell_fall(timing_template_7x8)
ocv_sigma_cell_fall(timing_templ...
ocv_sigma_cell_fall(timing_templ...
cell_rise(timing_template_7x8)
ocv_sigma_cell_rise(timing_temp...
ocv_sigma_cell_rise(timing_temp...
fall_transition(timing_template_7x8)
ocv_sigma_fall_transition(timing_...
ocv_sigma_fall_transition(timing_...
rise_transition(timing_template_...
ocv_sigma_rise_transition(timing...
ocv_sigma_rise_transition(timing...
output_current_fall() (56)
output_current_rise() (56)
receiver_capacitance1_fall(timing...
receiver_capacitance1_rise(timin...
receiver_capacitance2_fall(timing...
receiver_capacitance2_rise(timin...
ccsn_last_stage() (5)
pin(clk_in) (5)
Trends
X-axis: voltage
Color variable: process
voltage vs cell_fall (ns)
cell_fall (ns)
voltage
process
ssg
tt
ffg
Worst-Case Differences
Magnitude
Abs Diff
Rel Diff (%)
Type
Location
Corner
32.48p 15 NLDM fall delay time cell(cell_1)/... corner_0.770...
30.96p 15 NLDM fall delay time cell(cell_1)/... corner_0.770...
30.43p 15 NLDM fall transition time cell(cell_1)/... corner_0.770...
30.40p 15 NLDM fall transition time cell(cell_1)/... corner_0.770...
30.39p 15 NLDM fall transition time cell(cell_1)/... corner_0.770...
30.39p 15 NLDM fall transition time cell(cell_1)/... corner_0.770...
30.37p 15 NLDM fall transition time cell(cell_1)/... corner_0.770...
30.36p 15 NLDM fall transition time cell(cell_1)/... corner_0.770...
30.32p 15 NLDM fall transition time cell(cell_1)/... corner_0.770...
29.93p 15 NLDM rise delay time cell(cell_1)/... corner_0.770...
29.91p 15 NLDM fall delay time cell(cell_1)/... corner_0.770...
29.17p 15 NLDM fall delay time cell(cell_1)/... corner_0.770...
29.06p 15 NLDM rise transition time cell(cell_1)/... corner_0.770...
Corners
temp voltage process
A/B 125 0.535 ffg
A/B 125 0.54 tt
A/B 125 0.64 tt
A/B 125 0.7 tt
A/B 125 0.705 ssg
A/B 125 0.725 ffg
A/B 125 0.745 ffg
A/B 125 0.745 ssg
A/B 125 0.745 tt
A/B 125 0.77 ssg
A/B 125 0.775 tt
A/B 125 0.795 tt
A/B 125 0.825 ssg
A/B 125 0.825 tt
A/B 125 0.86 tt
A/B 125 0.88 ssg
A/B 125 0.88 tt
A/B 125 0.945 ssg
A/B 125 0.945 tt
A/B 125 0.975 ssg
Selected Group Information
cell(cell_1)
...
pin(clk)
direction: output;
output_voltage: default;
...
timing()
related_pin: "clk_in";
timing_sense: positive_unate;
timing_type: combinational;
when: "A&B";
wave_fall: ('6', '15', '6');
wave_rise: ('6', '15');
...
cell_fall(timing_template_7x8)
Selected Group Information

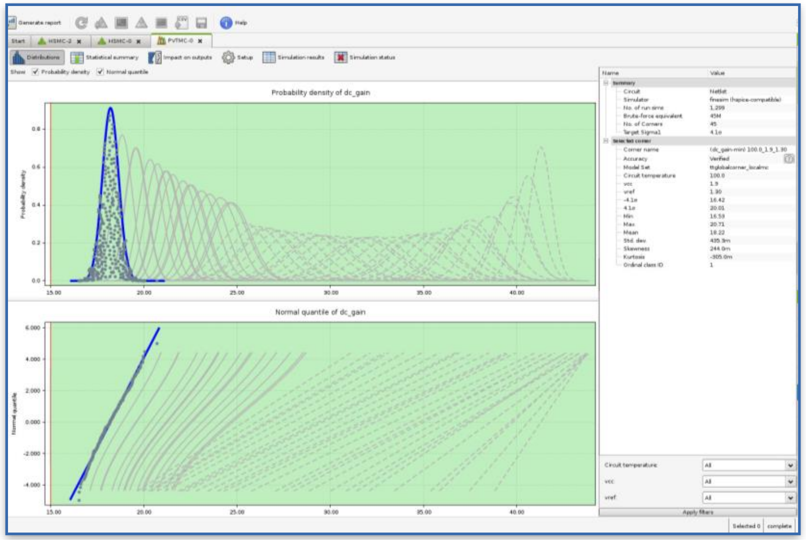
Solido ML Tools and Benefits

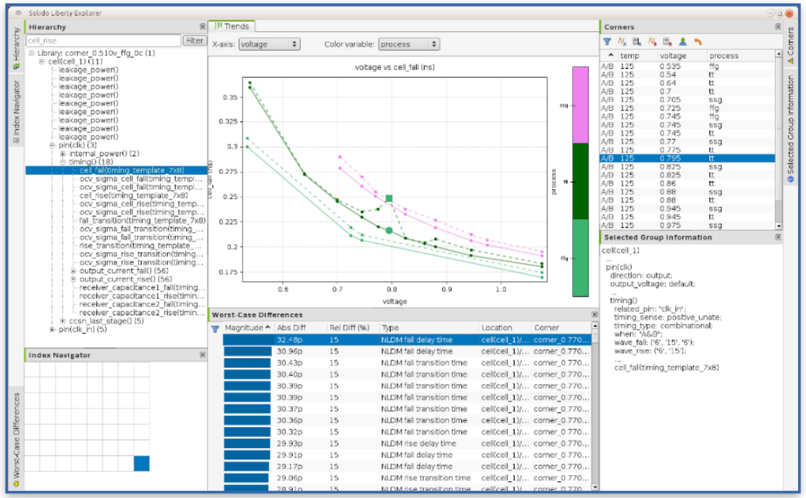
Solido Variation Designer

- 2-50X faster PVT analysis
- 10X faster 3-sigma Monte Carlo (1 PVT)
- 100X faster 3-sigma Monte Carlo (many PVTs)
- >1KX faster 4.5-sigma Monte Carlo
- >1MX 6-sigma Monte Carlo

Solido Characterization Suite

- Generates Liberty models at new PVTs 1000X faster
- Reduces overall timing characterization schedule by 50%
- Finds new classes of errors in Liberty data
- Reduces Liberty debugging time from weeks to hours





Great ML Requires More Than Just Speedups…

Verifiability: Can you tell if the ML model is right?

Accuracy: How close to perfect is the ML model?

Generality: Does the ML approach work on everything?

Robustness: Can I bet my next design schedule on it?

Usability: Does it "just work" for my team?

ML CAD Tool Capability Levels

Level 0: No ML

Gives the right answer by running all cases. Often too compute-intensive for production use.

Level 1: Partially reliable ML

Big speedups. Sometimes right, sometimes wrong, and can't tell. Cool demo! Shows promise. Not useful, yet.

Level 2: Partially reliable ML with accuracy-aware self-verification

Can tell when it's right or wrong based on accuracy criteria. Tool can't solve it automatically. Need a backup plan.

Level 3: Adaptive, accuracy-aware ML with self-verification

Meets accuracy criteria on most cases automatically. Still the odd bump. Useful, with great support.

Level 4: Fully reliable, accuracy-aware ML with self-verification

Production-ready and dependable. Production hardened and proven. Just works. Use it.

ML Levelquest and Effort

- **Level 0 to 1:** Identifying and testing ML opportunities (hours-days)
- **Level 1 to 2:** Defining what “accurate” means; making the algorithm self-verifying (weeks-months)
- **Level 2 to 3:** Developing methods for achieving accuracy dynamically and automatically (months-years)
- **Level 3 to 4:** Years of production hardening and addressing corner cases (years-decades)

- Each level is ~an order of magnitude more work

Level 2: Defining “Right” and Proving It Automatically

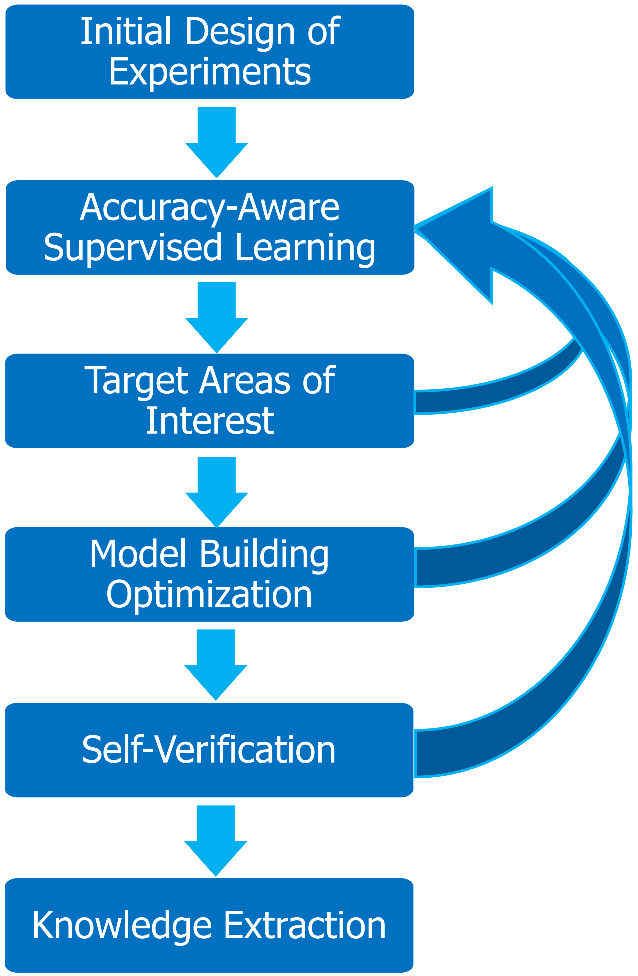
Define what “right” means for a given engineering problem

- Tolerances: Target range for a correct answer, absolute, relative, conditional
- Acceptable error rates, if applicable: Statistical affordances
- Biggest acceptable outliers: Min/max
- Regions of interest where higher accuracy is needed
- Regions of disinterest where no answer is even needed
- E.g. +/- 2 ps absolute error, <2% relative error for voltages <0.7, 99.5% of the time; 5ps, 5% tolerable for voltages >=0.7
- This is actually hard!

Designing an algorithm that can tell when ML-generated results are right, automatically

- Automated spot checking with spacefill sampling
- Automated checking in regions of highest interest
- Accuracy-aware modeling and automated checking in regions that have the widest model confidence intervals
- Self-verifying algorithms by design
- How would you prove it manually? Maybe do that?

Getting to Level 3 With Solido-Style Active Learning



Start by simulating a small amount of the total space, equally spread and covering all main and interaction effects

Build an accuracy-aware ML model that can very quickly predict outcomes for the rest of the space

Use the predicted values to decide which cases are interesting and simulate those; get simulation accuracy where it matters

Make the rest of the predictions adequately accurate; run more simulations in areas of greatest uncertainty

Prove that the areas of interest are covered and that target accuracy is achieved throughout the space

Help with insight and debugging by revealing dominant terms in ML models

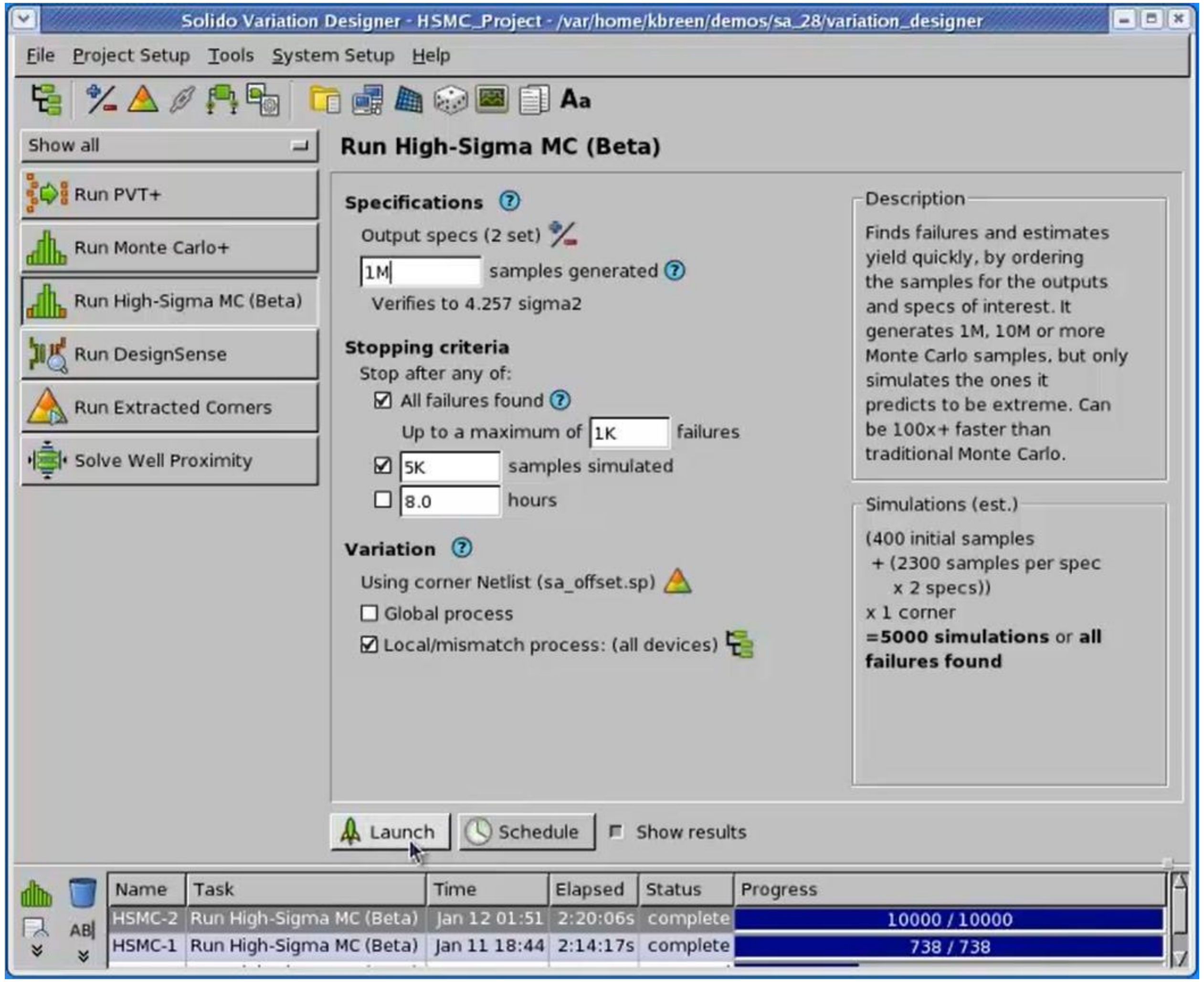
Solido's First Level 3 ML Solution: High-Sigma Monte Carlo (c.2008)

- Challenge: Monte Carlo and SPICE accurate high-sigma analysis
- Order of magnitude more simulations per 0.5 sigma!

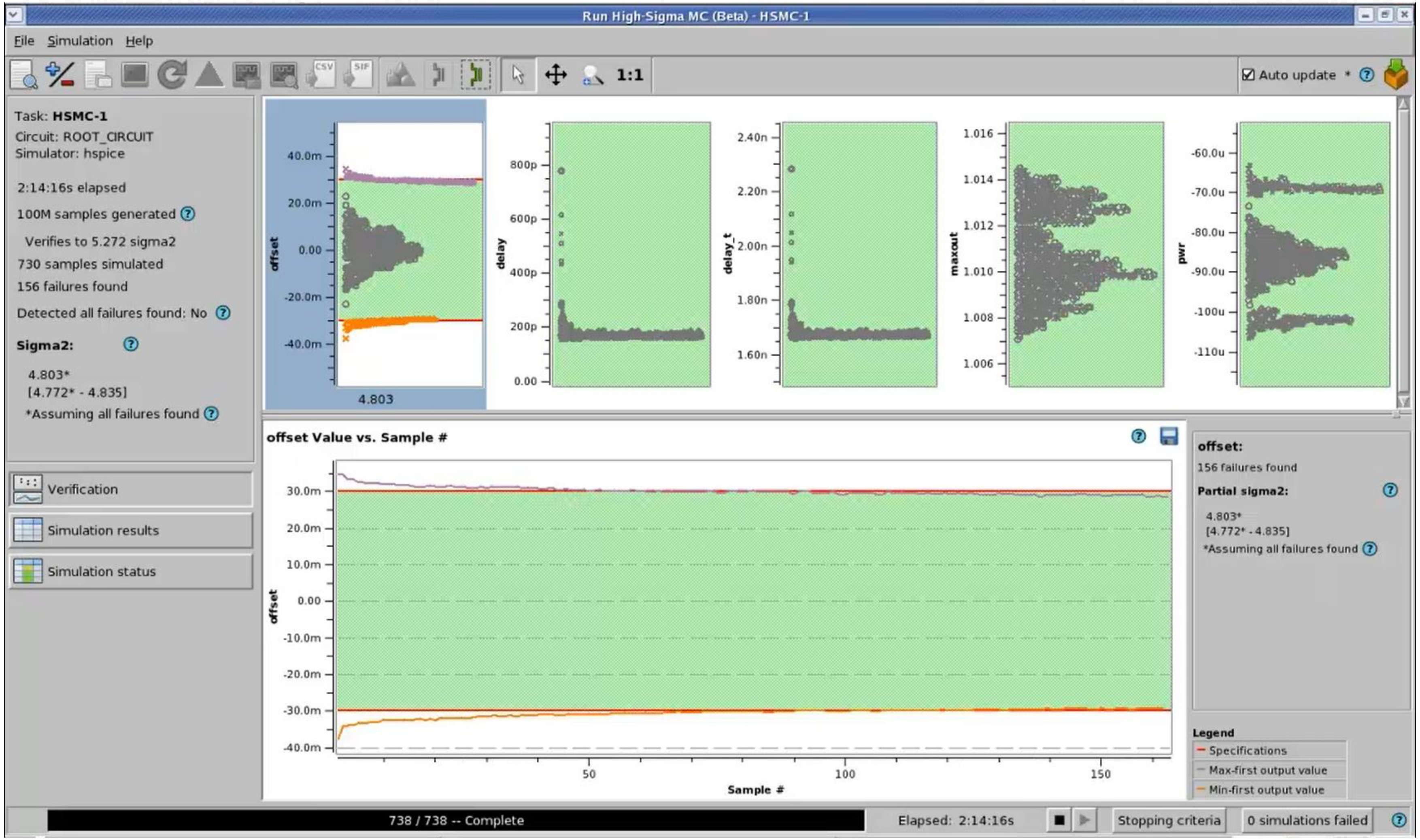
Sigma	Simulations
4	~1M
4.5	~10M
5	~100M
5.5	~1B
6	~10B

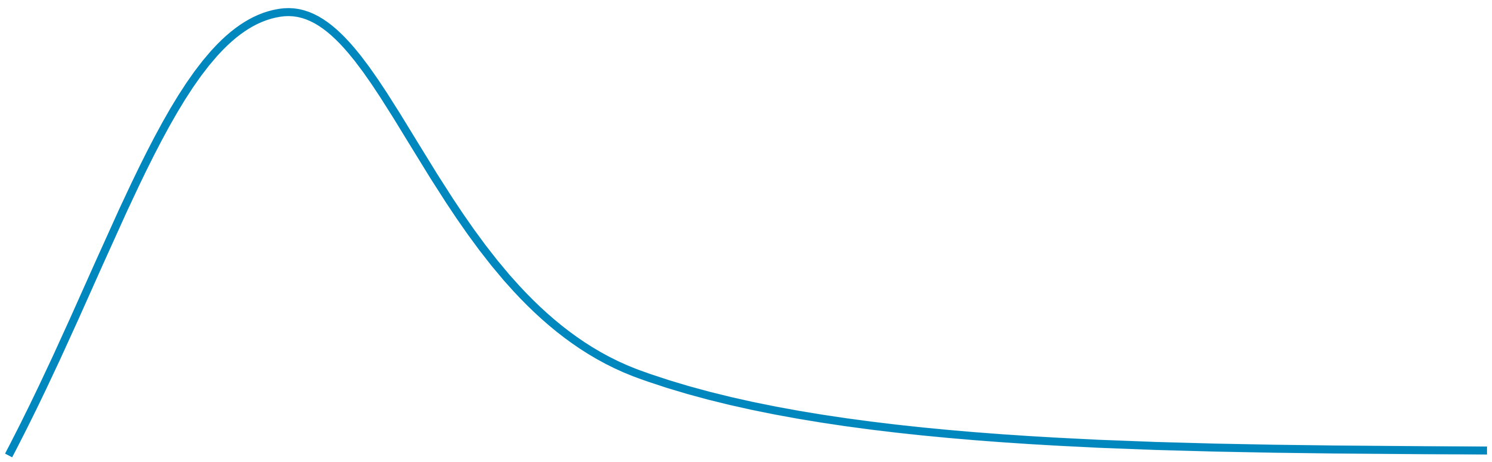
- Goal: MC and SPICE accuracy in 1000s of simulations
  (because that is all we have time for in production flows!)

Solido's First Level 3 ML Solution: High-Sigma Monte Carlo (c.2008)

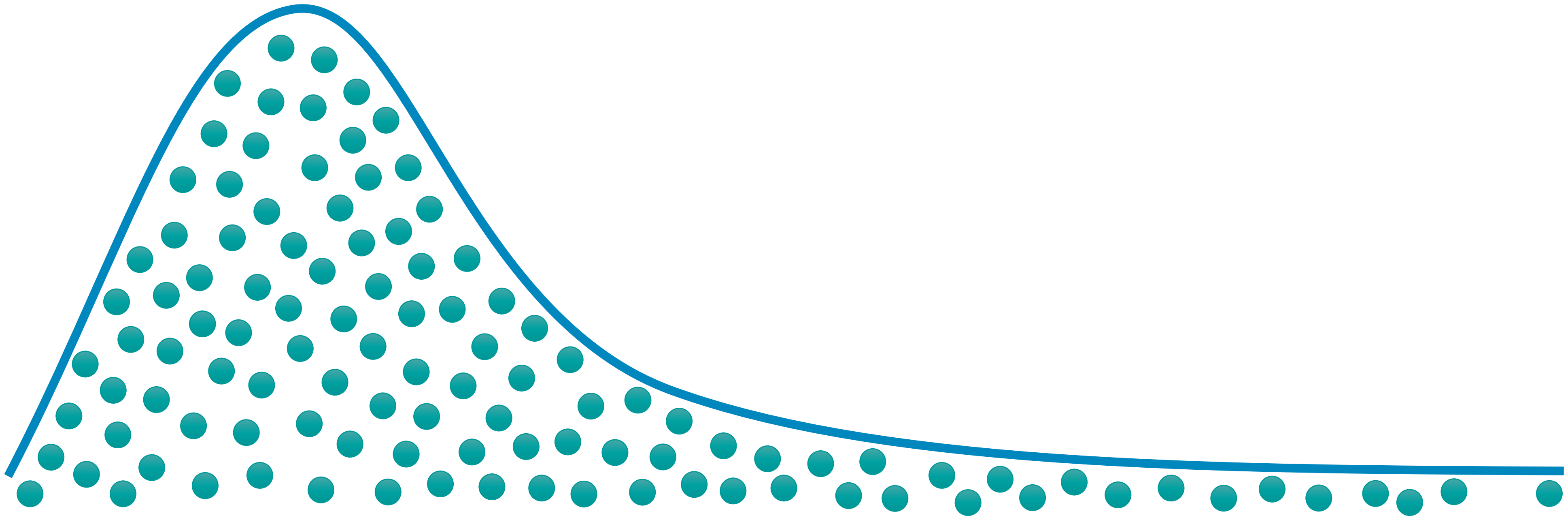


Solido's First Level 3 ML Solution: High-Sigma Monte Carlo (c.2008)



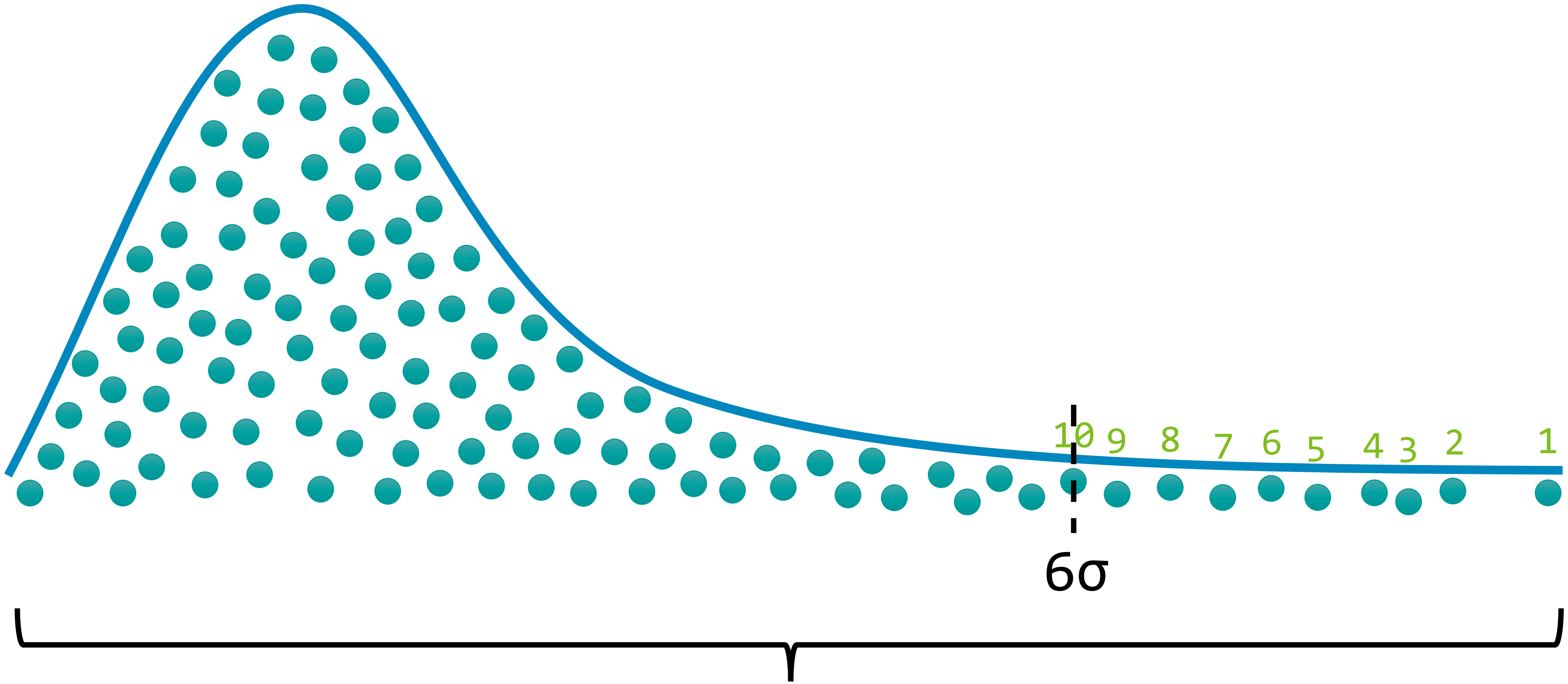


Brute Force Monte Carlo



Brute Force Monte Carlo

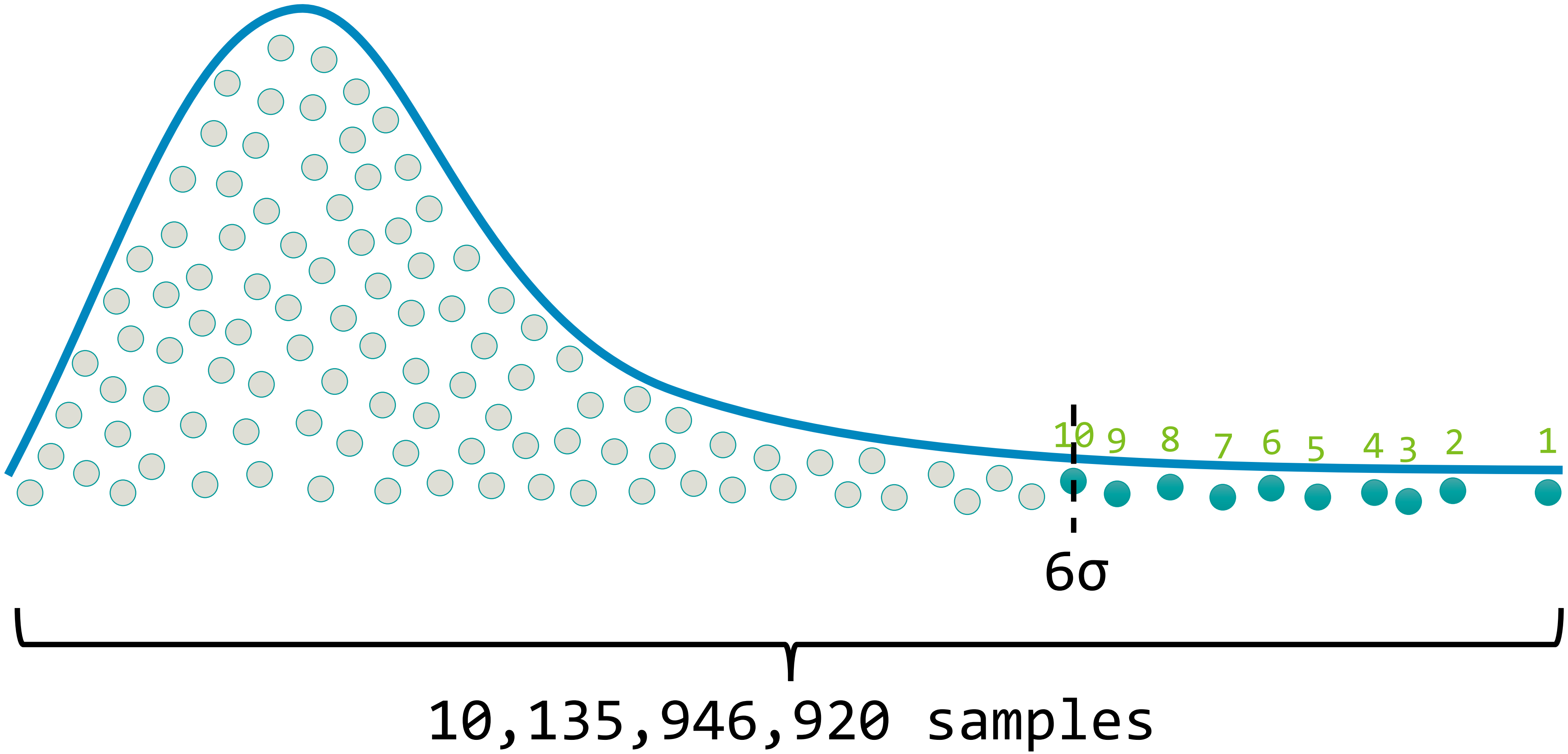
6σ = 10 failures per 10,135,946,920



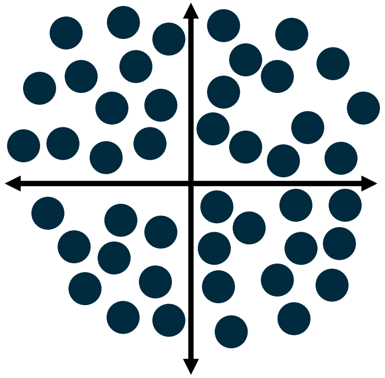
10,135,946,920 samples

What if…

6σ = 10 failures per 10,135,946,920

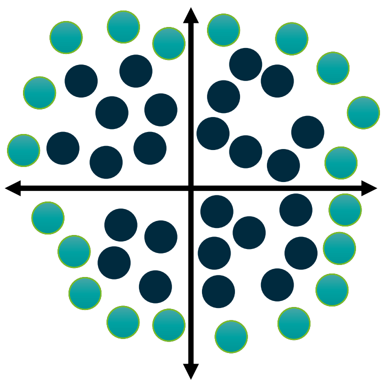


High-Sigma Monte Carlo



1. Generate Monte Carlo samples. Don’t simulate.

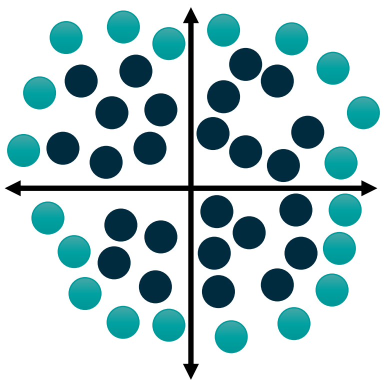
High-Sigma Monte Carlo



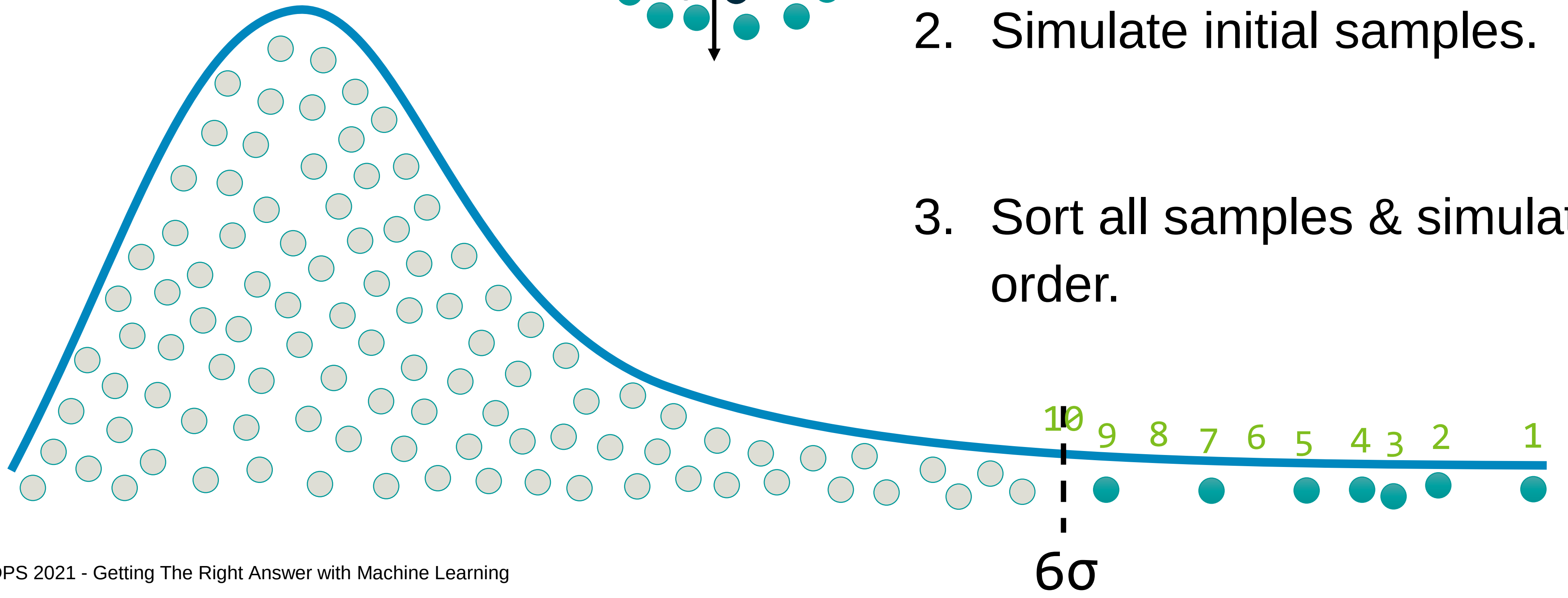
1. Generate Monte Carlo samples. Don’t simulate.

2. Simulate initial samples.

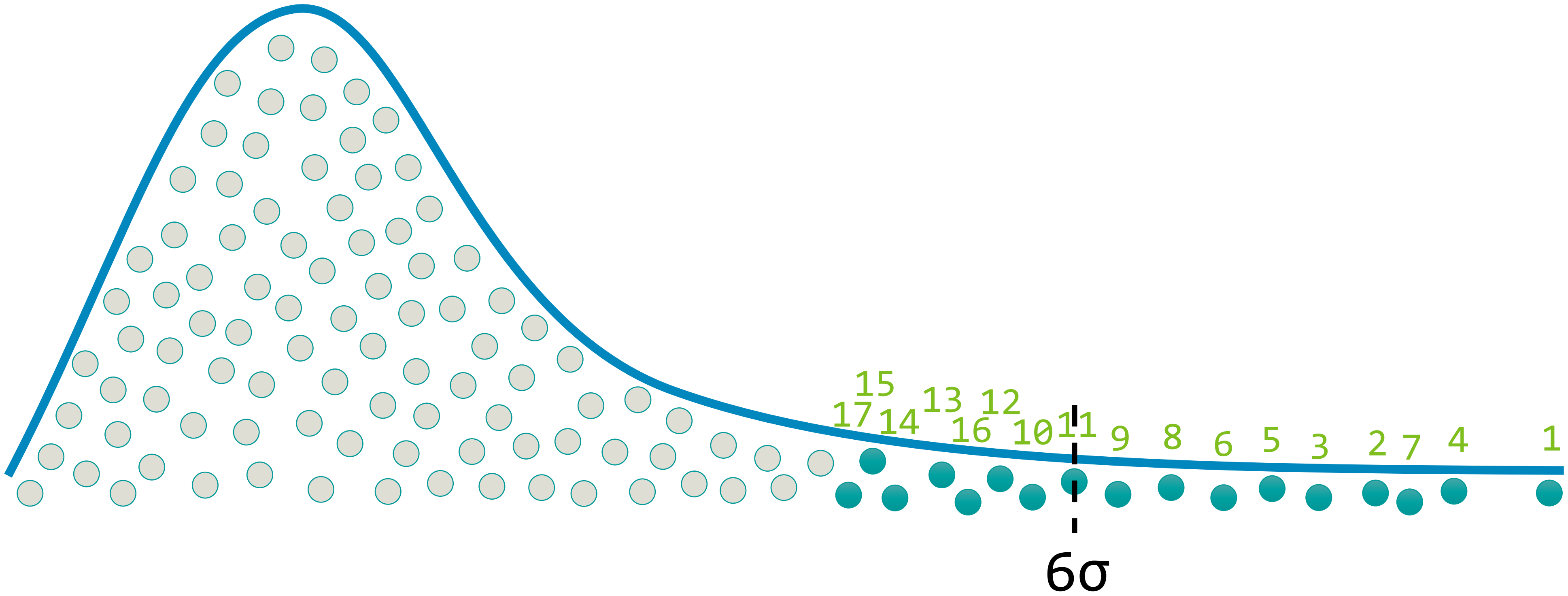
High-Sigma Monte Carlo



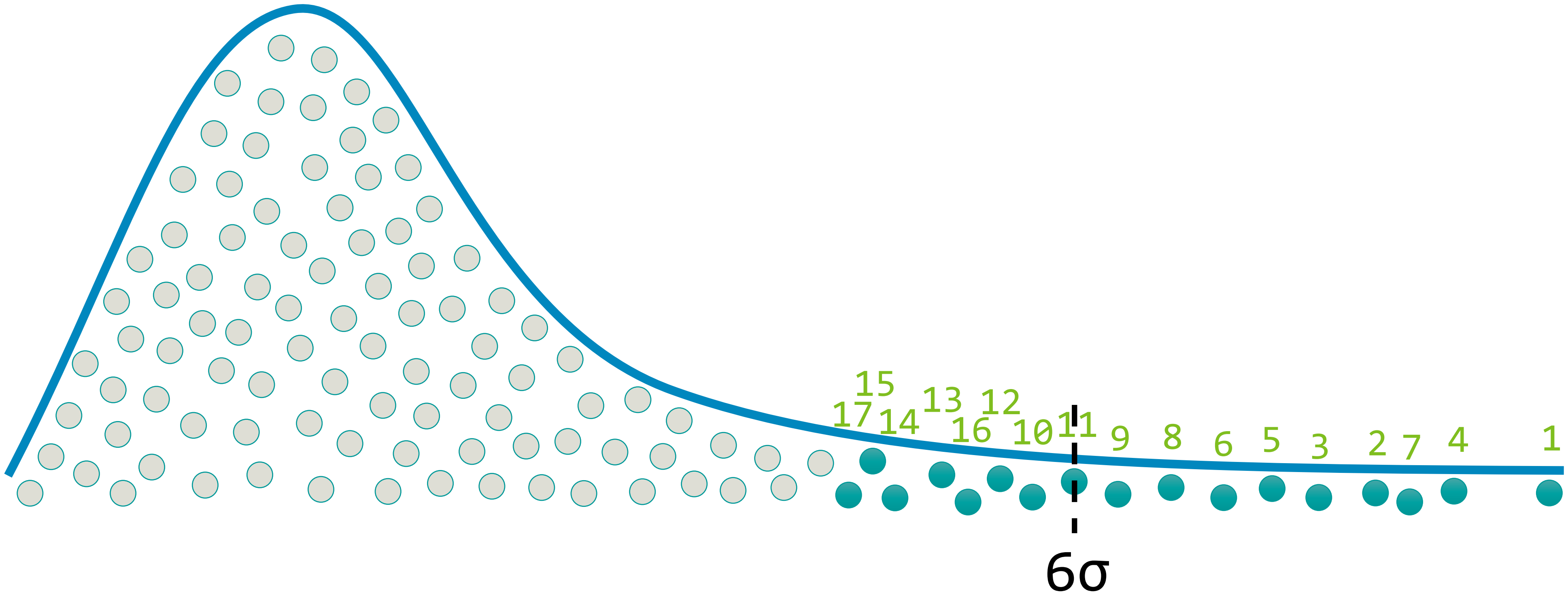
1. Generate Monte Carlo samples. Don’t simulate.
2. Simulate initial samples.
3. Sort all samples & simulate in order.



High-Sigma Monte Carlo applied



High-Sigma Monte Carlo applied



HSMC: Getting From Level 3 to Level 4

2008: Invented – level 3

- Capacity: 1M samples
- 1000 process variables
- Continuous outputs only
- 1 failure region
- Self-verification shown in logs
- Manual setup and operation
- Required stable simulation env.

2021: Person decades later – nearing level 4

- Capacity: 1T samples
- 100K process variables
- All outputs (e.g. multi-modal, n-ary)
- N failure regions
- Self-verification GUI
- Fully adaptive; no expertise needed
- Handles, recovers from, and corrects sim failures

Toward Level 4: Finding and Fixing the Corner Cases

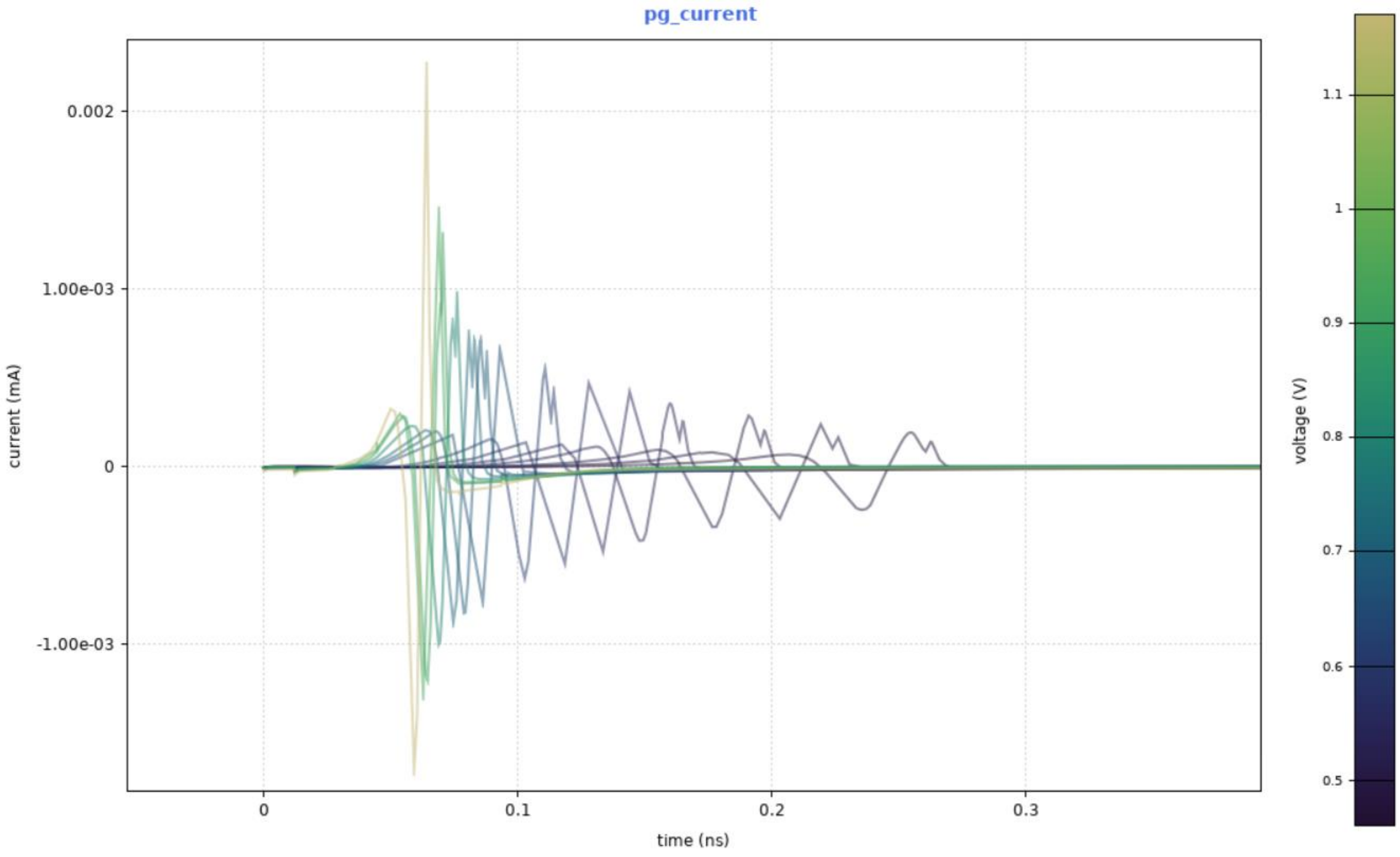
- This is the expensive part
- 100s of person years invested to date in going from level 3 to 4

Level 3 to 4 Example: Accurate ML Modeling for Jagged Waveforms Using Better Descriptive Parameters

Scott Johnston
James Marquez
Ken De Lathouwer
Megan Marsh

SIEMENS

Modeling Jagged Waveforms is Hard



Waveforms are shapes, not values

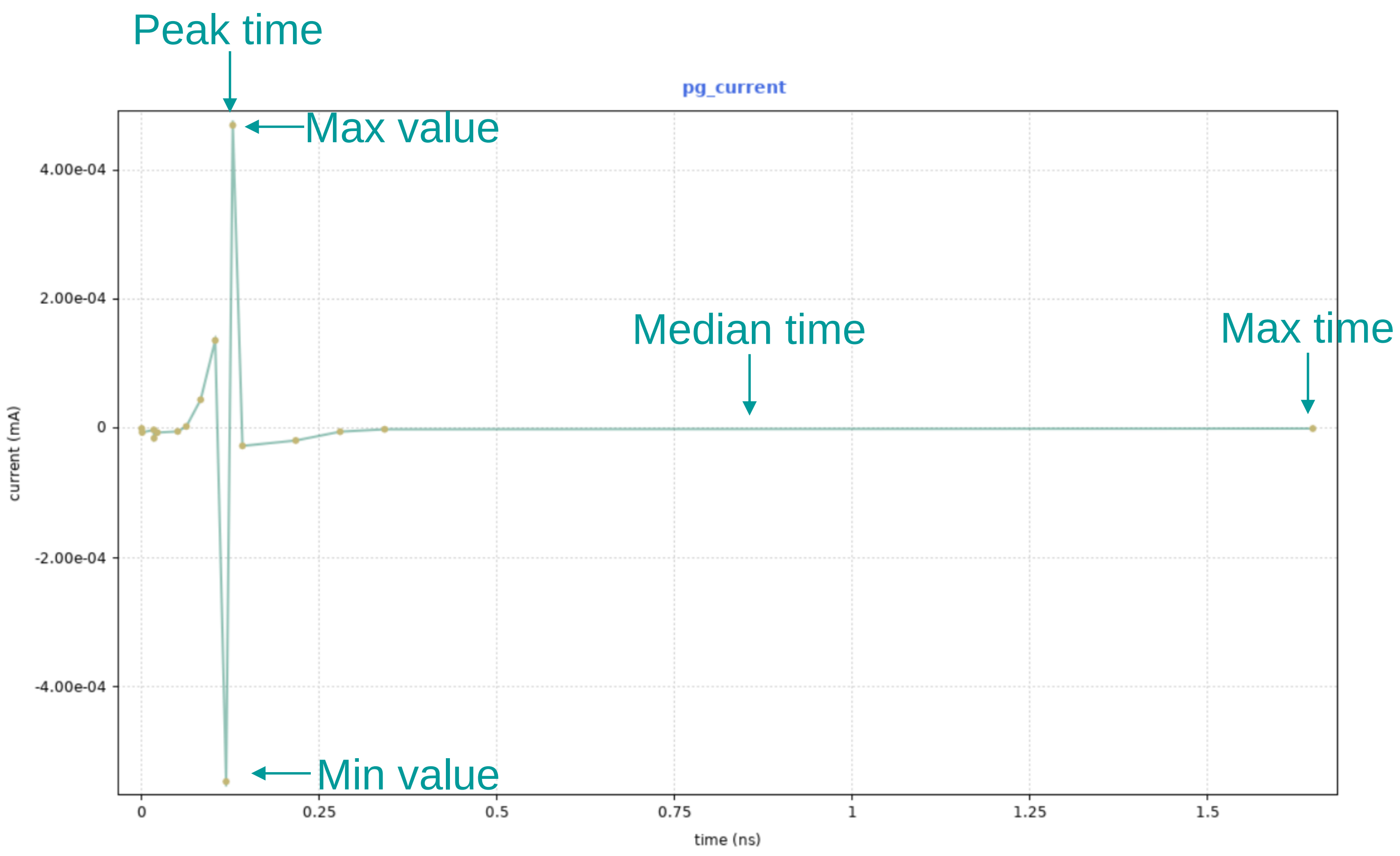
Accuracy of the shape is really important

Jagged shapes are really hard to model

Modeling the jagged shape directly fails

SIEMENS

Modeling Jagged Waveforms Using Their Parameters



Observation: The general shape holds across input dimensions

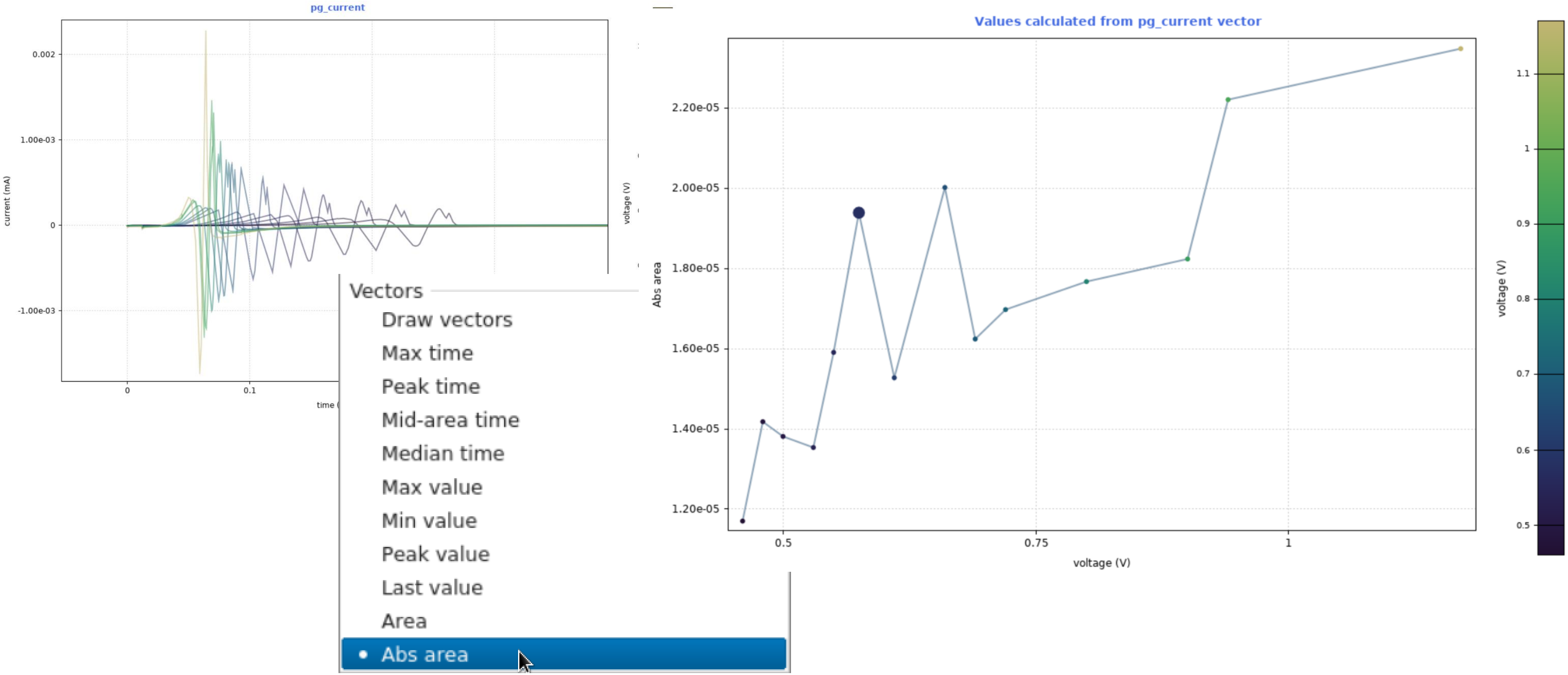
Observation 2: Waveforms are defined by continuous numeric parameters

Can we model and predict the parameters of a waveform?

Can we use those predicted parameters to morph the waveform to other input conditions?

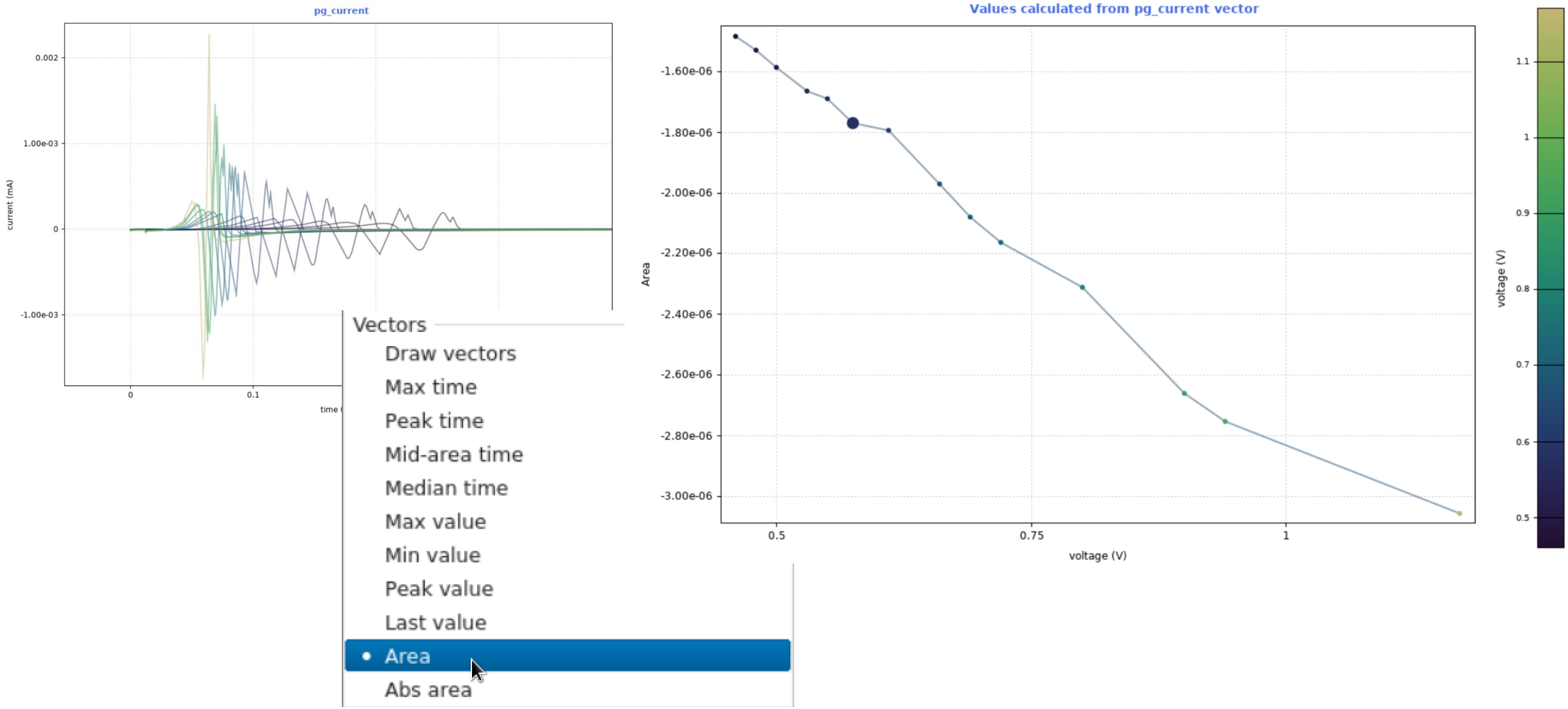
SIEMENS

Absolute Area: Too Noisy to Model



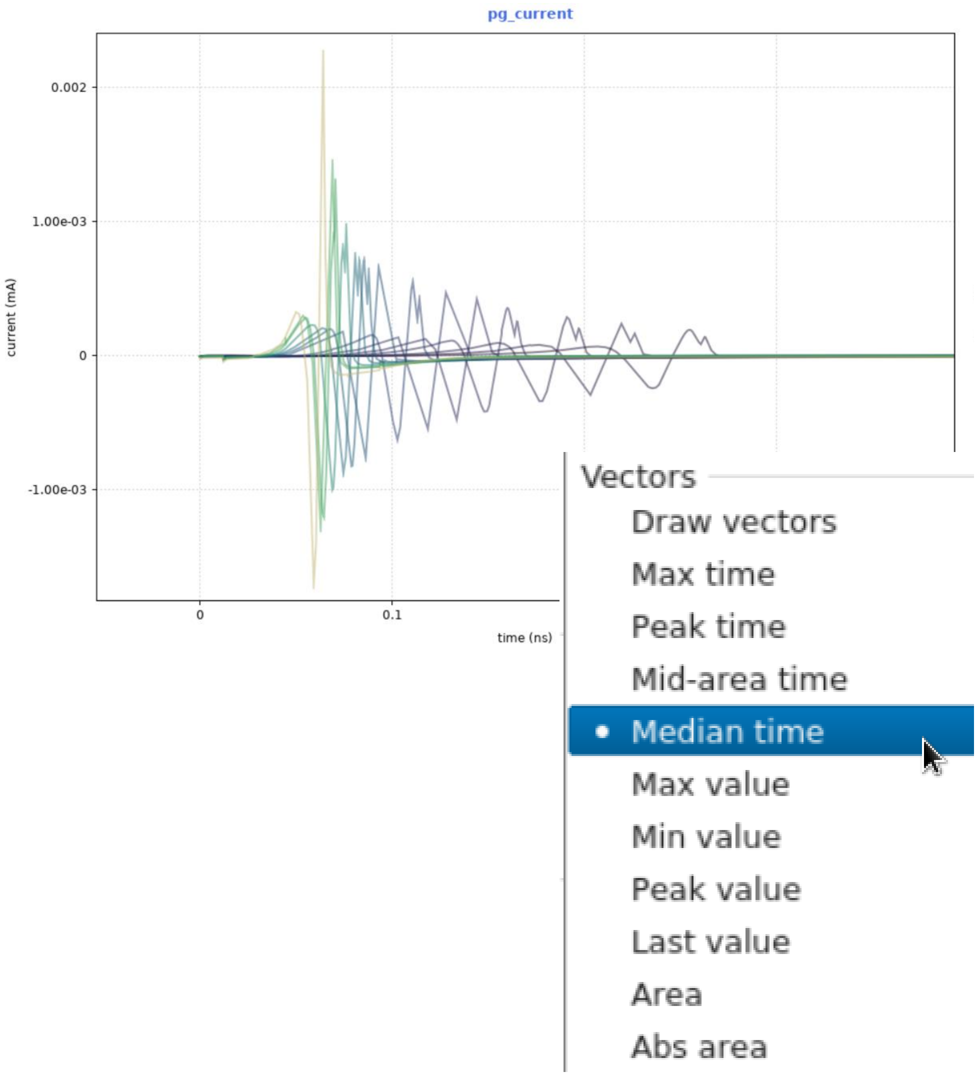
SIEMENS

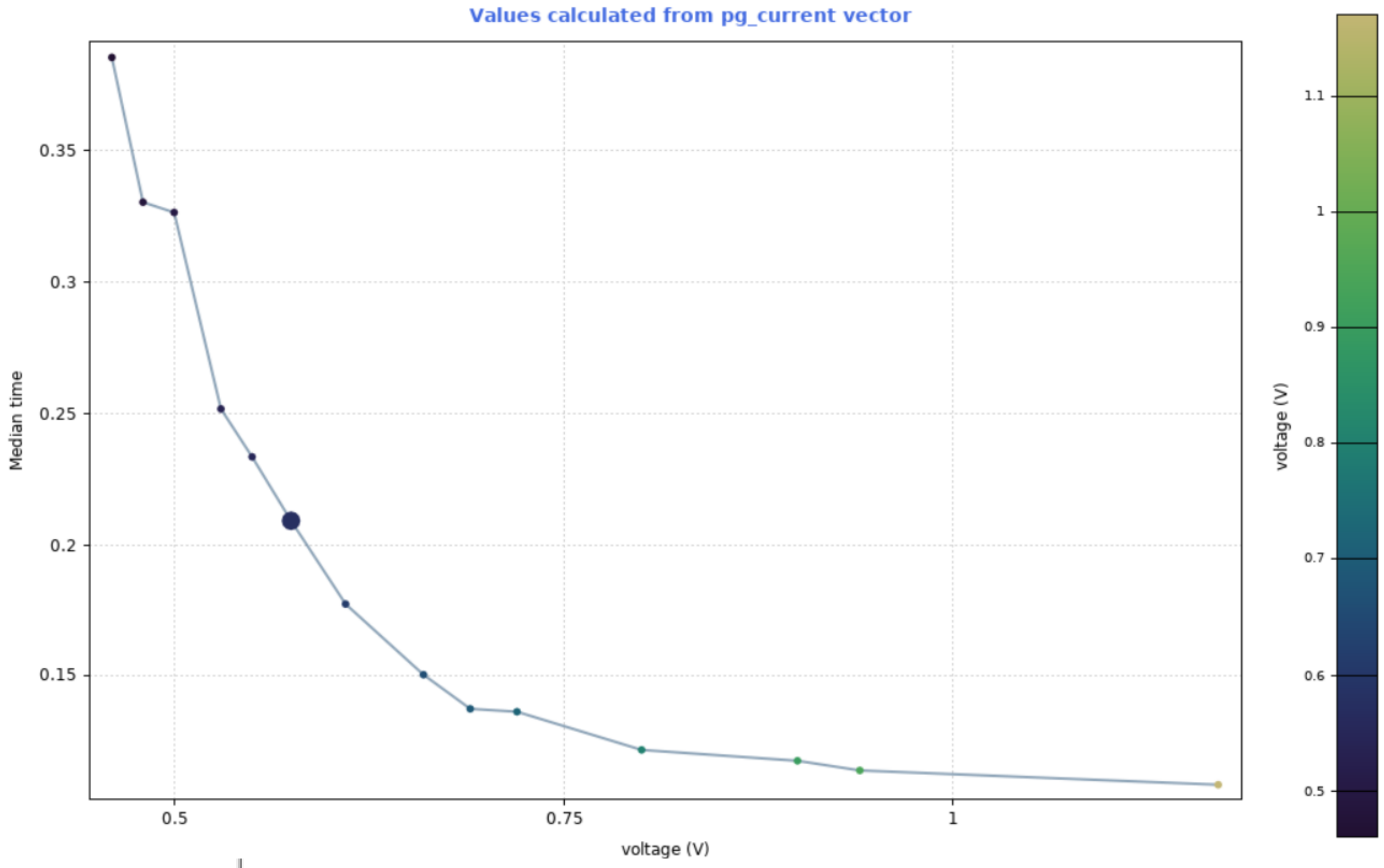
Area: Pretty Smooth – Models Well



SIEMENS

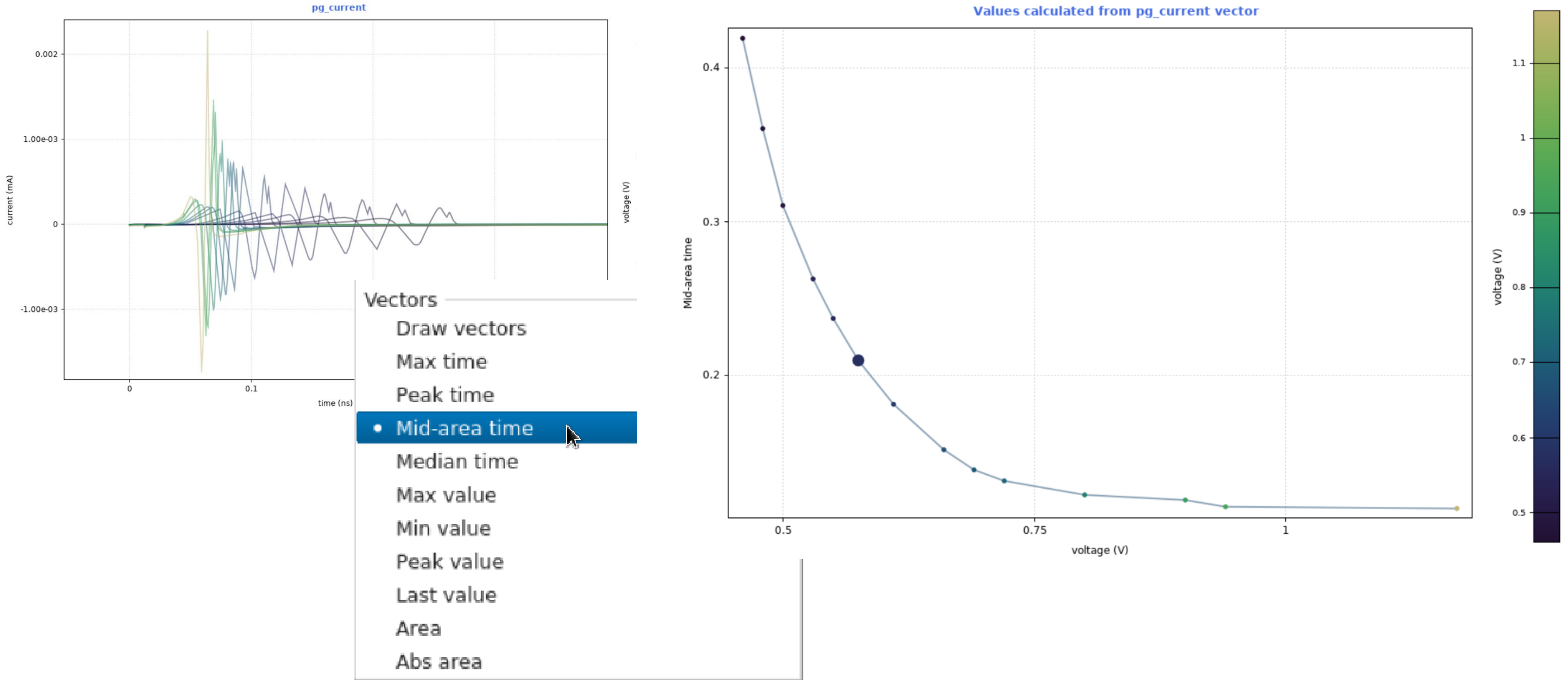
Median Time: Not Bad - A Little Noisy





SIEMENS

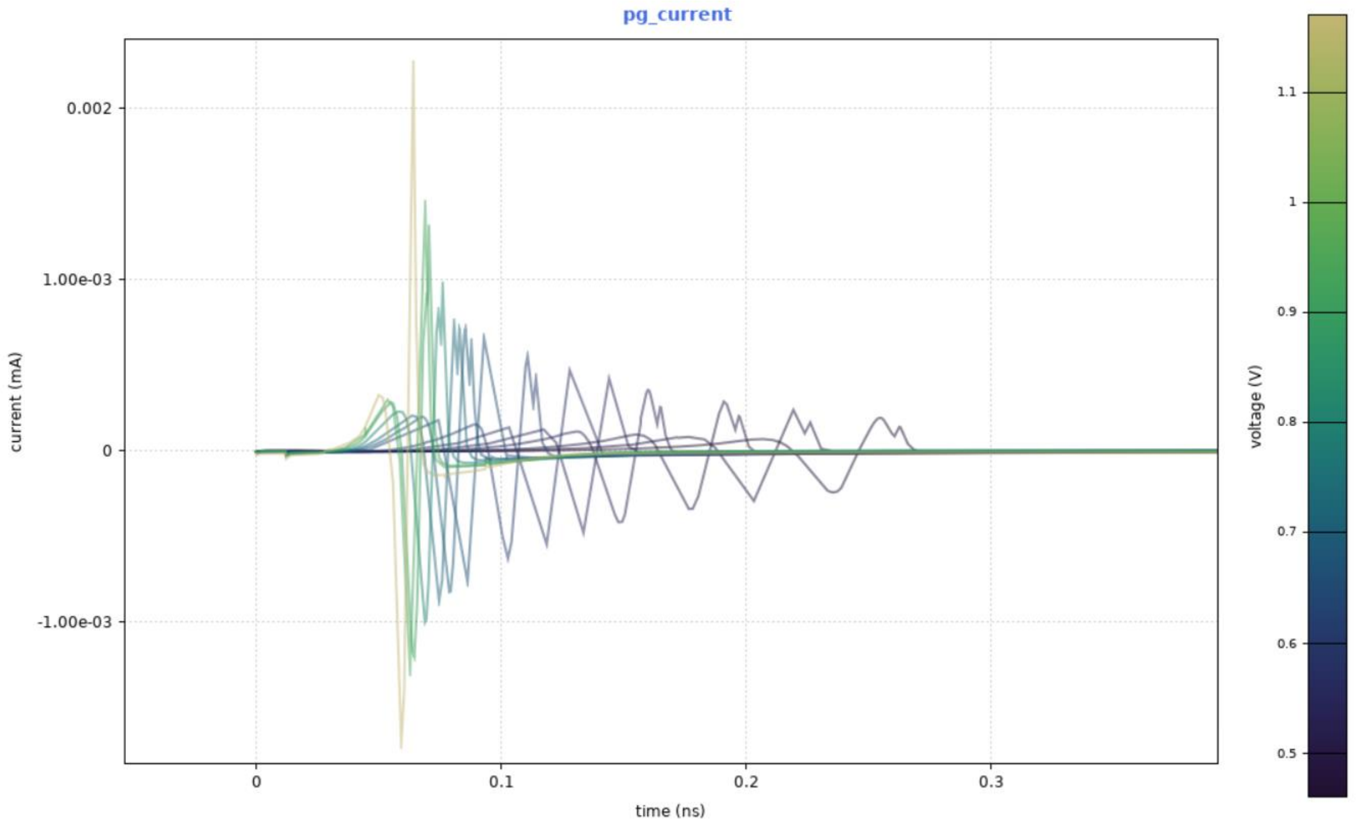
Mid-Area Time: Smoother and Models Well



SIEMENS

Improvement by Modeling Jagged Waveforms Using Better Parameters

	% In Tolerance (All corners)	% In Tolerance (WC Corners)
Original parameters	~80%	~50%
Best parameters	>99%	>99%



SIEMENS

ML CAD Tool Capability Levels

Level 0: No ML

Gives the right answer by running all cases. Often too compute-intensive for production use.

Level 1: Partially reliable ML

Big speedups. Sometimes right, sometimes wrong, and can't tell. Cool demo! Shows promise. Not useful, yet.

Level 2: Partially reliable ML with accuracy-aware self-verification

Can tell when it's right or wrong based on accuracy criteria. Tool can't solve it automatically. Need a backup plan.

Level 3: Adaptive, accuracy-aware ML with self-verification

Meets accuracy criteria on most cases automatically. Still the odd bump. Useful, with great support.

Level 4: Fully reliable, accuracy-aware ML with self-verification

Production-ready and dependable. Production hardened and proven. Just works. Use it.

SIEMENS

ML Levelquest and Effort

- **Level 0 to 1:** Identifying and testing ML opportunities (hours-days)
- **Level 1 to 2:** Defining what “accurate” means; making the algorithm self-verifying (weeks-months)
- **Level 2 to 3:** Developing methods for achieving accuracy dynamically and automatically (months-years)
- **Level 3 to 4:** Years of production hardening and addressing corner cases (years-decades)

- Each level is ~an order of magnitude more work

SIEMENS

Getting the Right Answer With Machine Learning

Levels 3 and 4 are useful for production engineering applications

Leveling up is hard, necessary, and has a ton of great research problems remaining

Thank you

SIEMENS