Release 2023 R1 Highlights
Ansys Twin Builder

Ansys

What is a Digital Twin?

Past, Present, Future, Simulate!

digital twin™ CONSORTIUM : “Virtual representation of real-world entities and processes, synchronized at a specified frequency and fidelity”

Customer Value: Track the past, provide deeper insights into the present, predict and influence future behavior

25 %
average margin for aftermarket services
compared to 10 % for new equipment

Increase Revenue

costs high as
$ 800 M
per year – an amount equal to as much as 3 % of productive revenues

Reduce Costs

1.65 B
industrial assets under condition monitoring by 2025

Gain Competitive Edge

Sources:
1. “Industrial aftermarket services: Growing the core.”, McKinsey.com
2. “Controlling Warranty Costs by Preventing No Fault Found”, WIKA Group
3..Total addressable market (TAM) and compound annual growth rate (CAGR) information throughout presentation is based on third party study completed by Evaluserve Inc. in 2019 commissioned by ANSYS. Study was based on customer and industry expert interviews and review of industry analyst reports and commentaries. Refer to Cautionary Statement for a discussion of factors that could impact future financial results.

Ansys Leading the Way in Global Digital Twin Initiatives

Digital Twin Consortium

digital twin™ CONSORTIUM

- Ansys is one of 8 founding members

Ansys AUTODESK. Bentley DELL Technologies

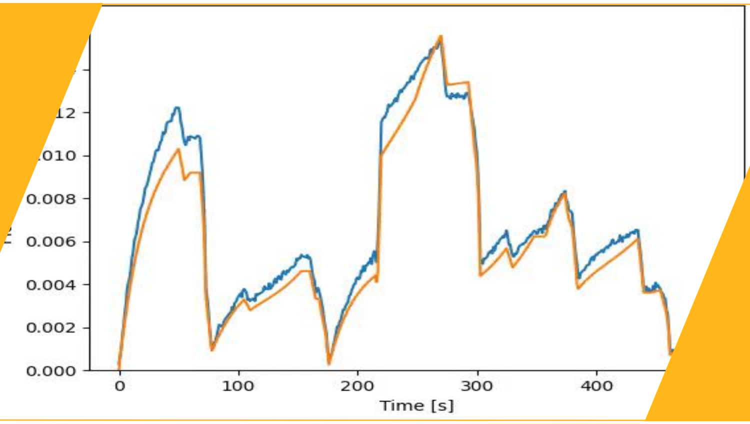
GE Digital lendlease Microsoft NORTHROP GRUMMAN

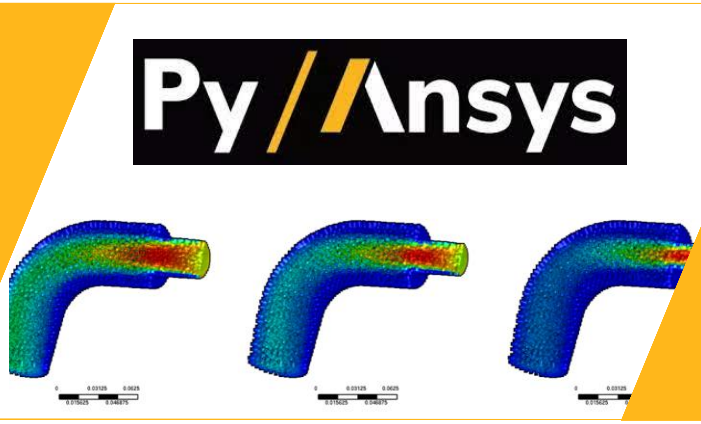
- Drives the development and adoption of digital twin technologies
- Emerging standards body (part of OMG)

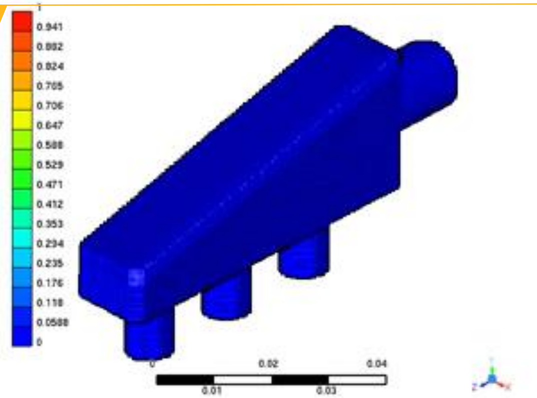
Digital Twin Definition Language (DTDL)

- Ansys and Microsoft collaborating on DTDL and on developing reusable reference architecture
- Enabling IoT solutions to provision, use, and configure IoT devices from multiple sources in a single solution

Accurate & Evolving Digital Twins with Twin Builder Hybrid Analytics







Increase Digital Twin Prediction Accuracy with Hybrid Analytics

- ✓ Model Exchange FMU support with new solver in Twin Deployer improves **interoperability and simulation stability**
- ✓ Improved documentation and API examples make it **easier to get started with deployment**
- ✓ Automate Runtimes deployment in Python with **PyTwin**

Automation, Speed & Accuracy for ROM creation

- ✓ **New Linear Static and Dynamic** ROMs offer **faster and more accurate** ROMs for a variety of applications
- ✓ Improved ROM visualization with the ability to generate ROM images with **geometry deformation**
- ✓ **Automate your ROM** creation with PyAEDT, which now includes new examples for creating static and dynamic ROMs

Enhanced Solver and Model Library and User Experience

- ✓ Improvements to **Modelica Editor Bijection** (Text-to-Diagram) support, which allows users to switch between text and diagram views for easier model creation
- ✓ **Easier workflow with "Wire Together“,** which allows for easier pin connection for components with large numbers of pins
- ✓ **Create and Export videos** from your ROM images for better visualization

Innovations in Ansys Digital Twin 2023 R1

Transform Your Operations with Data-Driven and Simulation-Based Digital Twin Software

Export and Scaled Deployment

- Support for Hierarchical Parameters in the Model Parameter Dialog
- **Model Exchange (ME) FMU Support in Twin Deployer**
- **Examples of Simpler Scaffolding Code** for Easier API Understanding
- Improved ROM Image Generation Support in the Deployment Package
- **New Improved API Documentation**
- Several Enhancements and Bug-Fixes
- **Automate Runtimes deployment in Python with PyTwin**

Reduced Order Model

- **New Linear Static ROM with Support for Large Number of Parameters**
- **New Linear Dynamic ROM**
- Cut-plane coordinate Specification by Orientation and Origin
- Improved Workflow for Renaming Input/Output Pins for ROMs
- **Image Generation with Geometry Deformation**
- Improvements on modal ROM export from Mechanical

Model Building-Modelica Enhancements

- **Modelica Editor Bijection (Text-to-Diagram) support**
- Modelica Enhanced Diagram Graphics
- Support for Hierarchical Display and Editing of Parameters

Solver and Model Library

- Usability Improvements for Device Characterization Wizard
- Support for Gradient Fitting in Device Characterization
- New Basic Dynamics MOSFET Model with Q3 control (Beta)
- Carrier Wave Options in the SVPWM Component

General Enhancements

- **Easier Pin Connection for Large Components**
- **Twin Builder ROM Animation Video Export**
- **Example of ROM creation using PyAnsys**
- Easy Debug and Error Log Access

Hybrid Digital Twin

Ansys

Challenges in Creating a Digital Twin

Accuracy

Adhere as closely as possible to physics behavior

Flexibility

Model multiple scenarios without overfitting to any one

Adaptability

Adapt to changing environments and conditions

Scalability

Rapidly build and deploy adaptable digital twins

Digital Twin Goals: Accuracy, Time, and Cost

Customers are demanding more from their products, pressuring manufacturers to produce faster, better, and more reliable designs

Data and simulation approaches each make reasonable predictions, but also both face limitations

Data-Driven Modeling

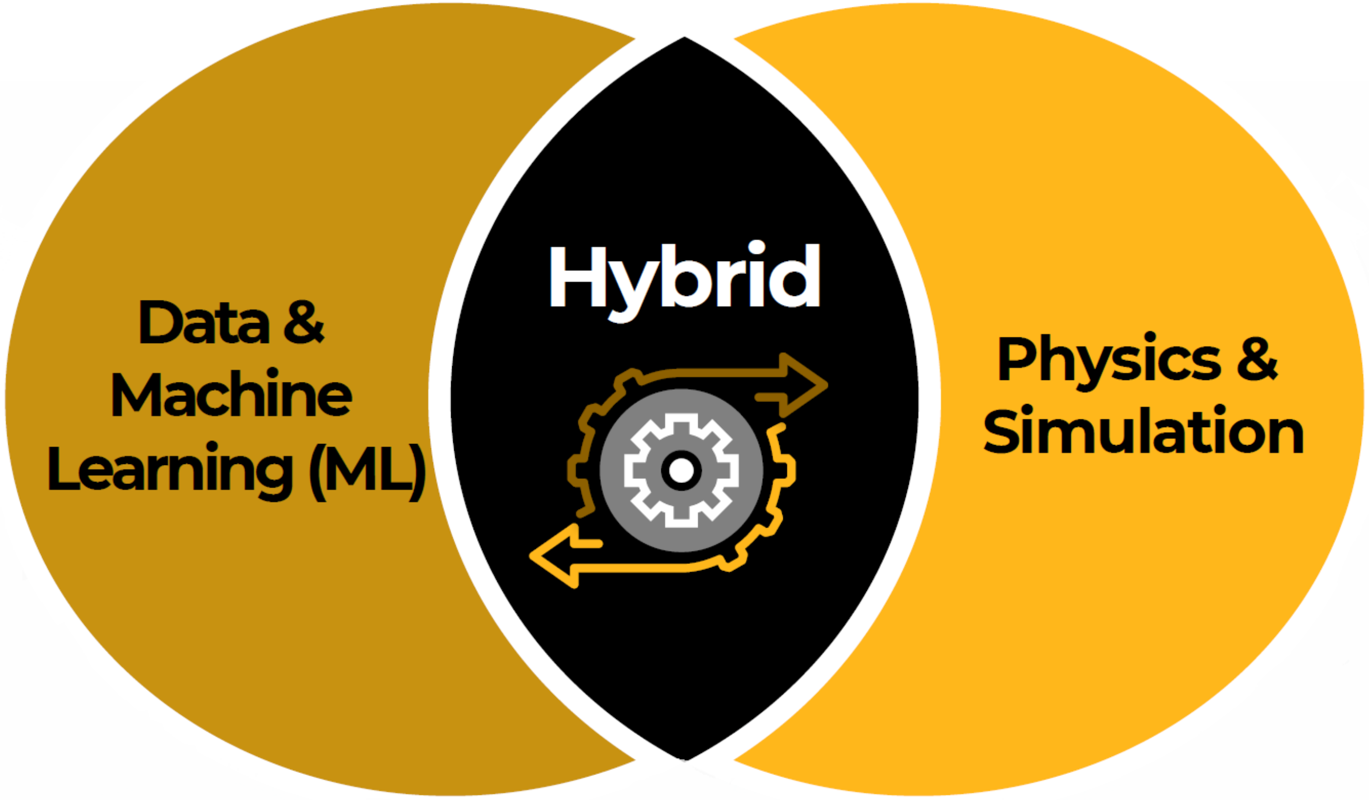
Accuracy limited by observed data

Simulation-Based Modeling

Missing real-time field insights

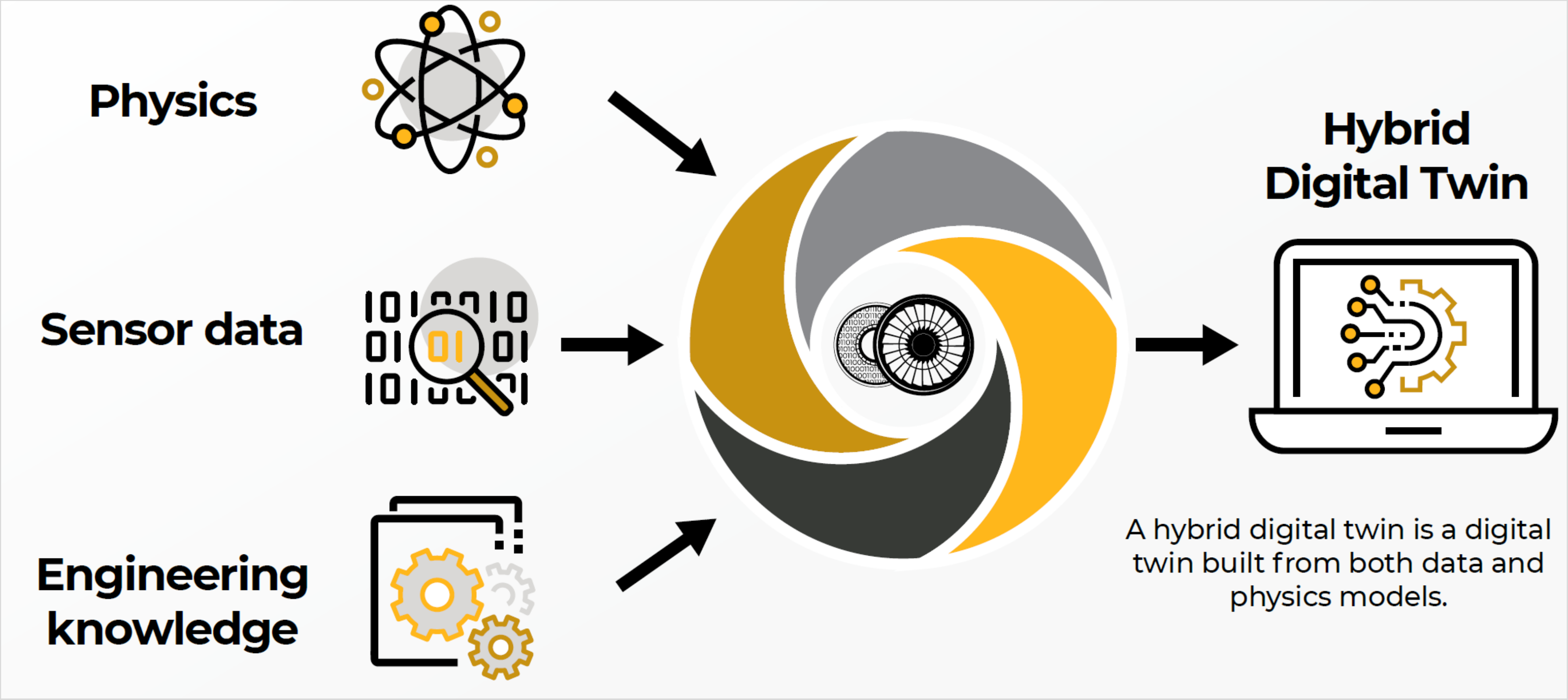
The Solution: Hybrid Digital Twins, the Best of Machine Learning and Physics

Hybrid Analytics is a toolset for combining data and physics modeling using machine learning techniques to create Hybrid Digital Twins



How Does it Work?

Combine data and physics to create the best possible digital twins



Physics

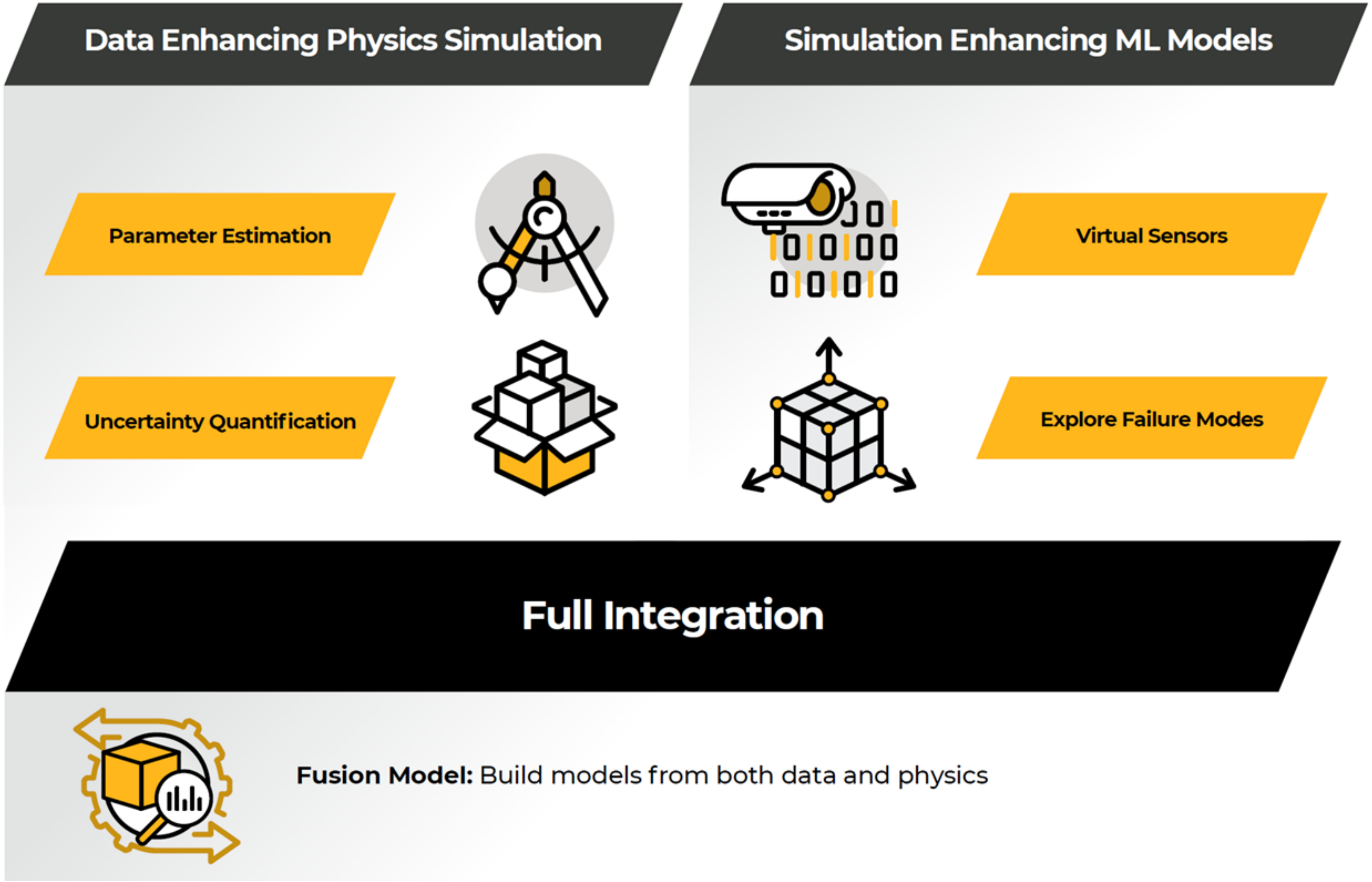
Sensor data

Engineering knowledge

Hybrid Digital Twin

A hybrid digital twin is a digital twin built from both data and physics models.

Hybrid Analytics Techniques



Data Enhancing Physics Simulation

Parameter Estimation

Uncertainty Quantification

Simulation Enhancing ML Models

Virtual Sensors

Explore Failure Modes

Full Integration

Fusion Model: Build models from both data and physics

Hybrid Analytics Capabilities

Transform your operations with data-driven and simulation-based digital twin software

Unparalleled Accuracy

Closely match simulation results with measurement data by calibrating model parameters

Uncertainty Quantification

Uncertainty quantification on parameters and outputs provides confidence in fit

Fusion Modeling

Compensate for any unmodeled physics or other effects by modeling the difference between a physics model and data

Robust Algorithms

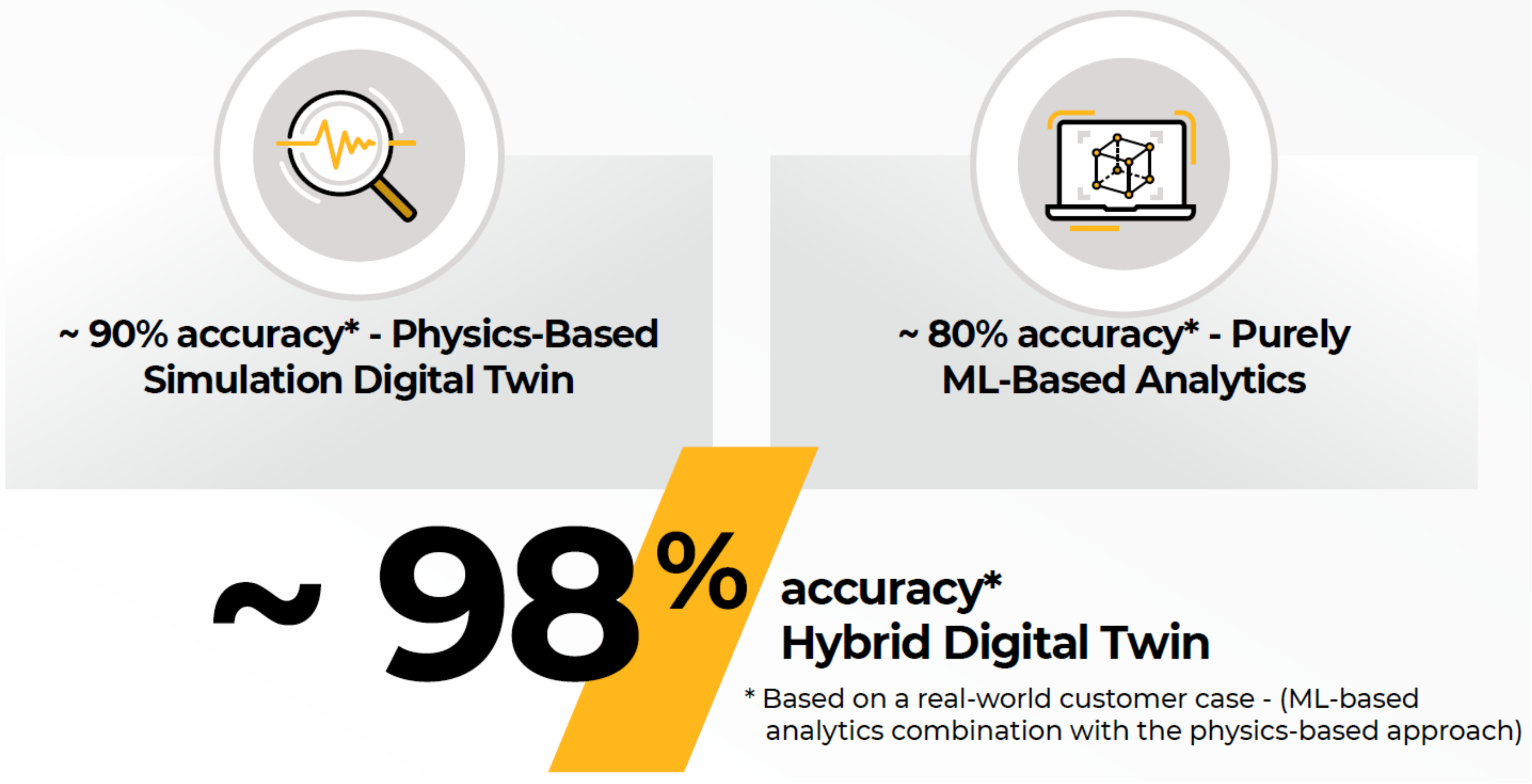
Cover diverse use cases

Easy Online Recalibration

Quickly recalibrate digital twin with online recalibration

Hybrid Digital Twin: Combine ML-Based Analytics with Physics-Based Approaches to Create Accurate and Evolving Digital Twins

Using hybrid analytics techniques, the resulting hybrid digital twin is more accurate than if using either just data and machine learning of physics simulation alone.



~ 90% accuracy* - Physics-Based Simulation Digital Twin

~ 80% accuracy* - Purely ML-Based Analytics

~ 98% accuracy* Hybrid Digital Twin

* Based on a real-world customer case - (ML-based analytics combination with the physics-based approach)

Hybrid Digital Twin Key Benefits

Create virtual sensors to measure missing data

Perform what-ifs before applying a solution

Analyze accurate and deterministic predictions based on physical principles

Explore causality and failure modes using physics

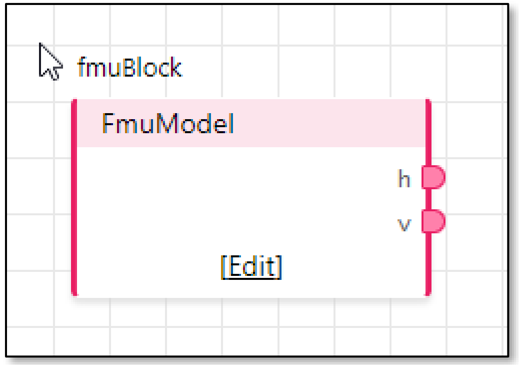
Export and Scaled Deployment

Easily Export and Deploy Digital Twins

Model Exchange (ME) FMU Support in Twin Deployer

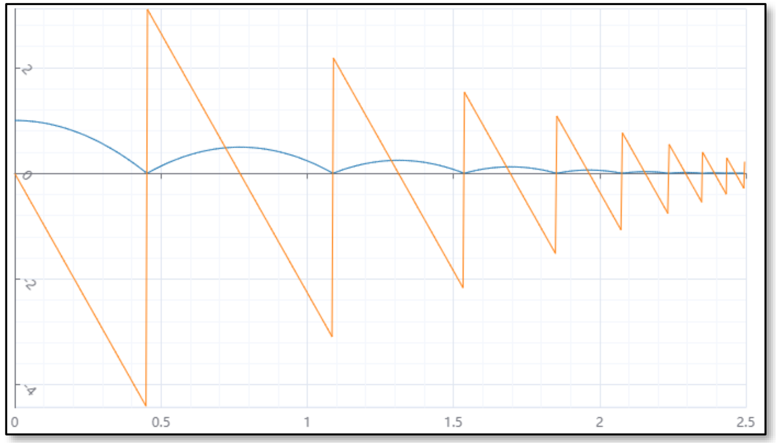
What's New

- Support for FMI 2.0 Model Exchange FMUs



User Benefits

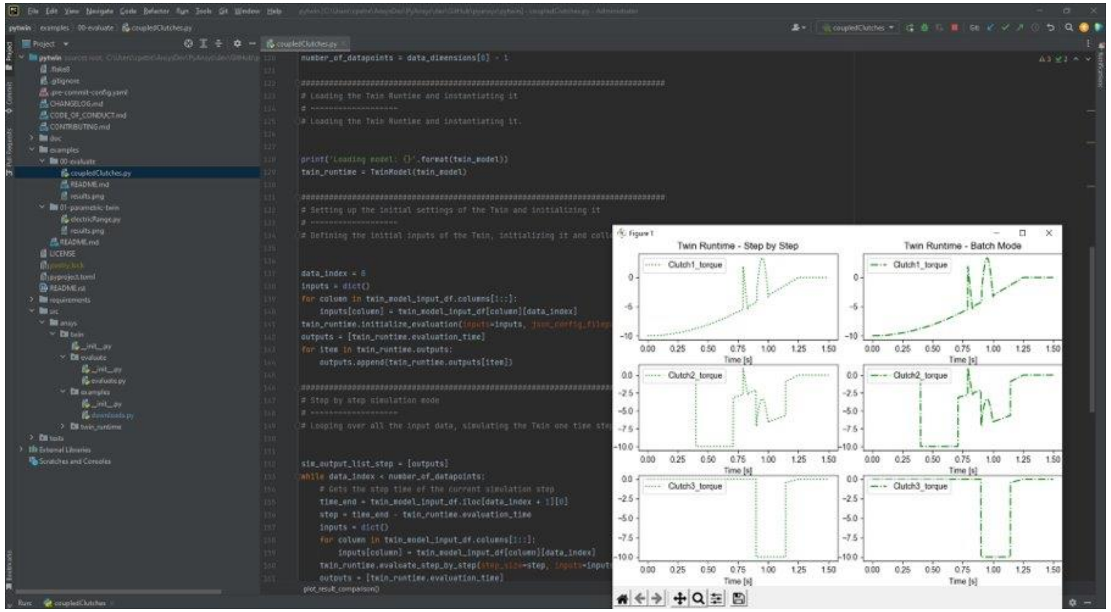
- ME simulation stability has been improved.
- For improved simulation stability, the FMU Sundials CVODE solver is used to perform numerical integration.



Extend API and Automation through PyAnsys/PyTwin

What's New

- Expose deployment workflows to PyAnsys/PyTwin
- Provide new Python workflows for Runtime integration with examples of new use cases (twin consumption, offline twin application, ROM results and connection with Ansys flagships,…)



```
number_of_datapoints = data_dimensions[0] - 1

################################################################################
# Loading the Twin Runtime and instantiating it
# ------------------
# Loading the Twin Runtime and instantiating it.


print('Loading model: {}'.format(twin_model))
twin_runtime = TwinModel(twin_model)

################################################################################
# Setting up the initial settings of the Twin and initializing it
# ------------------
# Defining the initial inputs of the Twin, initializing it and colle


data_index = 0
inputs = dict()
for column in twin_model_input_df.columns[1::]:
    inputs[column] = twin_model_input_df[column][data_index]
twin_runtime.initialize_evaluation(inputs=inputs, json_config_filep
outputs = [twin_runtime.evaluation_time]
for item in twin_runtime.outputs:
    outputs.append(twin_runtime.outputs[item])

################################################################################
# Step by step simulation mode
# ------------------
# Looping over all the input data, simulating the Twin one time step


sim_output_list_step = [outputs]
while data_index < number_of_datapoints:
    # Gets the step time of the current simulation step
    time_end = twin_model_input_df.iloc[data_index + 1][0]
    step = time_end - twin_runtime.evaluation_time
    inputs = dict()
    for column in twin_model_input_df.columns[1::]:
        inputs[column] = twin_model_input_df[column][data_index]
    twin_runtime.evaluate_step_by_step(step_size=step, inputs=input
    outputs = [twin_runtime.evaluation_time]
plot_result_comparison()
```

User Benefits

- PyTwin makes it simple to test and automate Twins deployment in Python.
- The SDK API's high-level abstraction makes it even easier to interact with Twin to Initialize/Simulate/Post process the Twins.
- Leverage new workflow opportunities when connecting with Python ecosystem (including other PyAnsys packages)

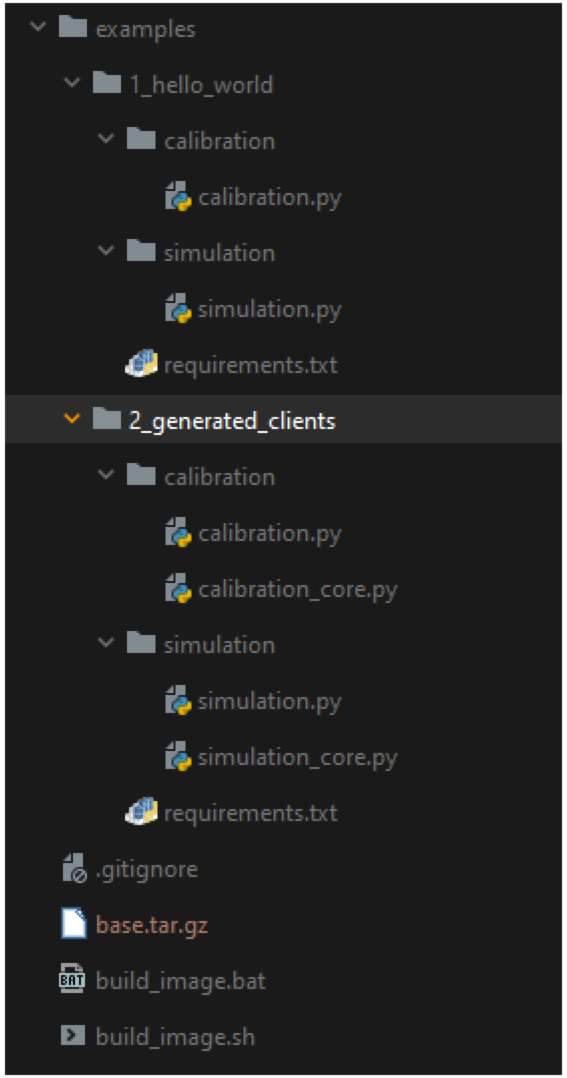
Easier Scaffolding Code Examples for Getting Started (1/2)

What’s New

- Example of auto-generated clients moved to separate folders in Twin Deployer

User Benefits

- 1_hello_world
  - Only basic error handling to keep simplicity
  - Easier to understand the API usage
- 2_generated_clients
  - Hides implementation details regarding API endpoints and payloads
  - Common API errors handled by generated clients
  - Focus on application usage rather than on the REST API behavior



```
examples
  1_hello_world
    calibration
      calibration.py
    simulation
      simulation.py
    requirements.txt
  2_generated_clients
    calibration
      calibration.py
      calibration_core.py
    simulation
      simulation.py
      simulation_core.py
    requirements.txt
  .gitignore
  base.tar.gz
  build_image.bat
  build_image.sh
```

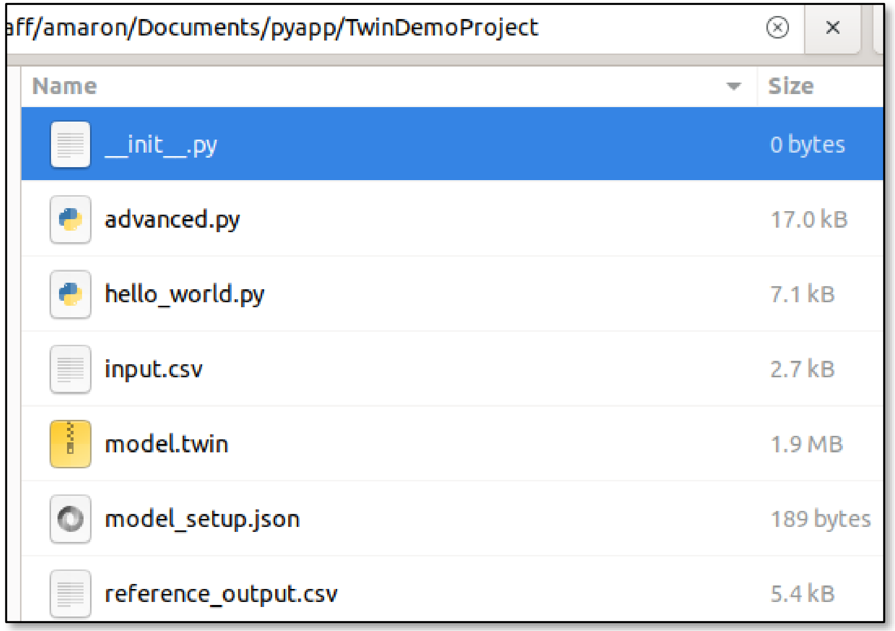
Easier Scaffolding Code Examples for Getting Started (2/2)

What’s New

- Updates on sample clients
  - A new hello_world.py example for simple and advanced.py for more advanced APIs
  - By default, hello_world.py is executed when executing the run_windows.bat/run_linux.sh scripts
  - Updated some embedded dependencies to ensure compatibility with both Ubuntu20.04 and CentOS7.8/7.9

User Benefits

- ‘hello world.py’ example contains much simpler sample client code that only runs a Batch Mode CSV simulation (fastest simulation mode)
- Easier for new user to get started with the API



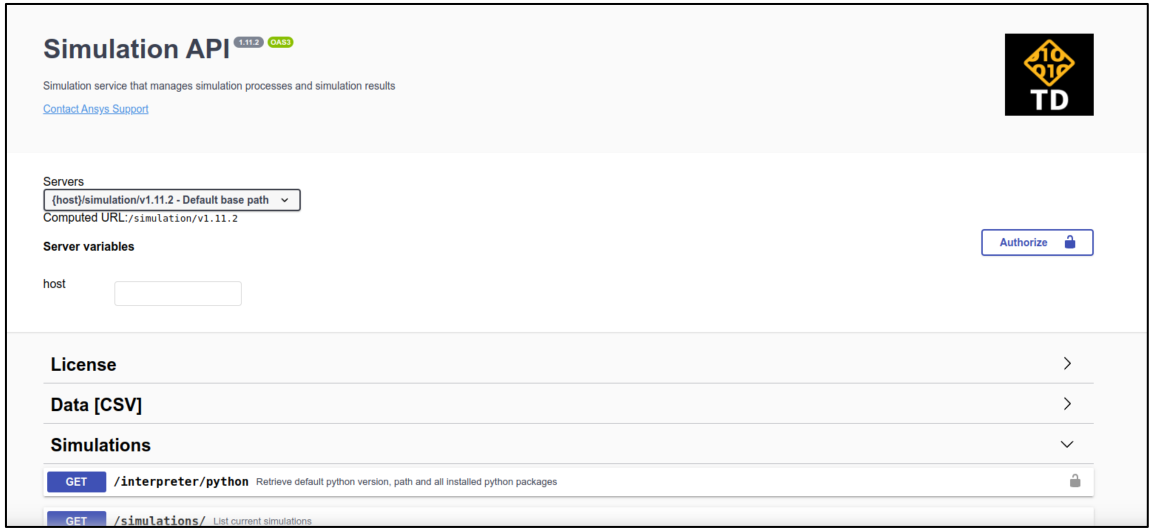
aff/amaron/Documents/pyapp/TwinDemoProject

Name	Size
__init__.py	0 bytes
advanced.py	17.0 kB
hello_world.py	7.1 kB
input.csv	2.7 kB
model.twin	1.9 MB
model_setup.json	189 bytes
reference_output.csv	5.4 kB

New Improved API Documentation for Deployment

What’s New

- New Improved API documentation is exported from Twin Deployer



Simulation API 1.11.2 OAS3

Simulation service that manages simulation processes and simulation results

Contact Ansys Support

Servers

{host}/simulation/v1.11.2 - Default base path

Computed URL:`/simulation/v1.11.2`

Server variables

host

GET `/interpreter/python` Retrieve default python version, path and all installed python packages

GET `/simulations/` List current simulations

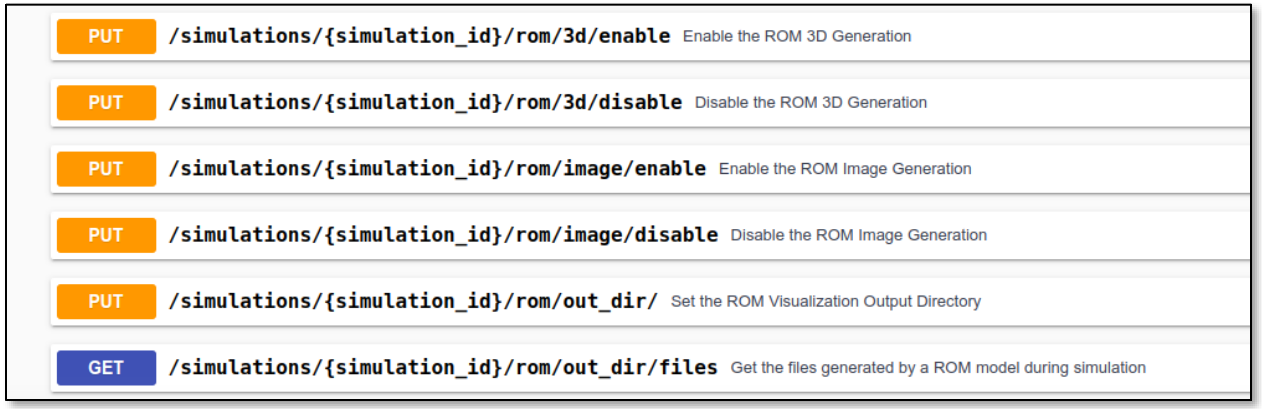
User Benefits

- The new improved style and exported sample code make the documentation more accessible to understand the different APIs and their payloads.

Support for ROM Image Generation in Deployment Package

What's New

- New API endpoints for ROM .modbin/.png file generation and retrieval



PUT `/simulations/{simulation_id}/rom/3d/enable` Enable the ROM 3D Generation

PUT `/simulations/{simulation_id}/rom/3d/disable` Disable the ROM 3D Generation

PUT `/simulations/{simulation_id}/rom/image/enable` Enable the ROM Image Generation

PUT `/simulations/{simulation_id}/rom/image/disable` Disable the ROM Image Generation

PUT `/simulations/{simulation_id}/rom/out_dir/` Set the ROM Visualization Output Directory

GET `/simulations/{simulation_id}/rom/out_dir/files` Get the files generated by a ROM model during simulation

User Benefits

- Currently, ROM models require X server and display access from within the container, making .png file generation impossible in headless environments (i.e., cloud). This capability enables it.
- Files are downloaded as .zip file

API Versioning Support in Endpoint URLs

What’s New

- API endpoint URLs have been updated to support multiple versioning options, making it easier to maintain legacy client code.

```
localhost:9080/simulation/v1.10.2/simulations
localhost:9080/simulation/v1.10/simulations
localhost:9080/simulation/v1/simulations
```

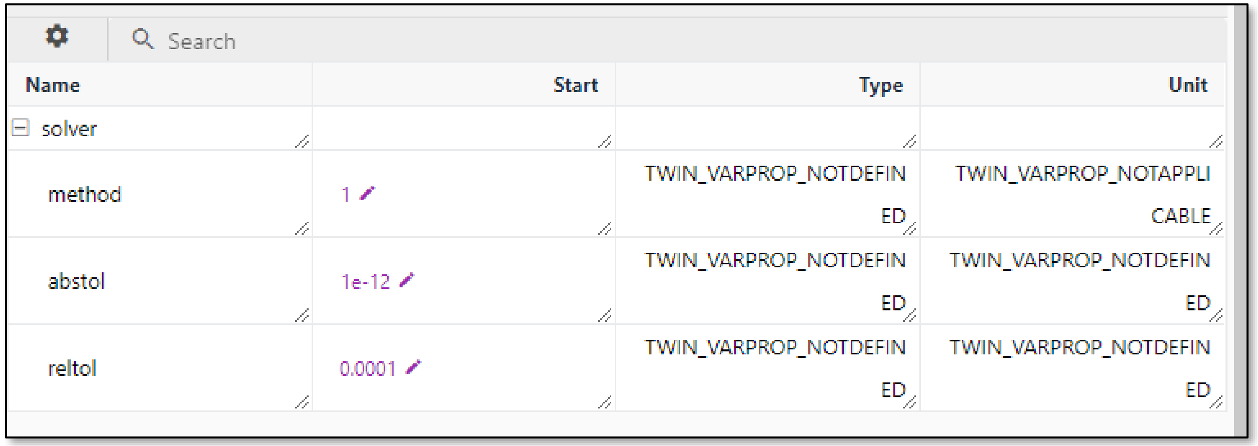
User Benefits

- Customers can deploy containers from multiple Twin Deployer versions without changing their client applications, unless the major versions changes (i.e., a breaking change was introduced)

Twin Deployer: Model Parameter Dialog Enhancements

What’s New

- New Model Parameter Dialog enhancement in Twin Deployer



Name	Start	Type	Unit
⊟ solver			
method	1	TWIN_VARPROP_NOTDEFINED	TWIN_VARPROP_NOTAPPLICABLE
abstol	1e-12	TWIN_VARPROP_NOTDEFINED	TWIN_VARPROP_NOTDEFINED
reltol	0.0001	TWIN_VARPROP_NOTDEFINED	TWIN_VARPROP_NOTDEFINED

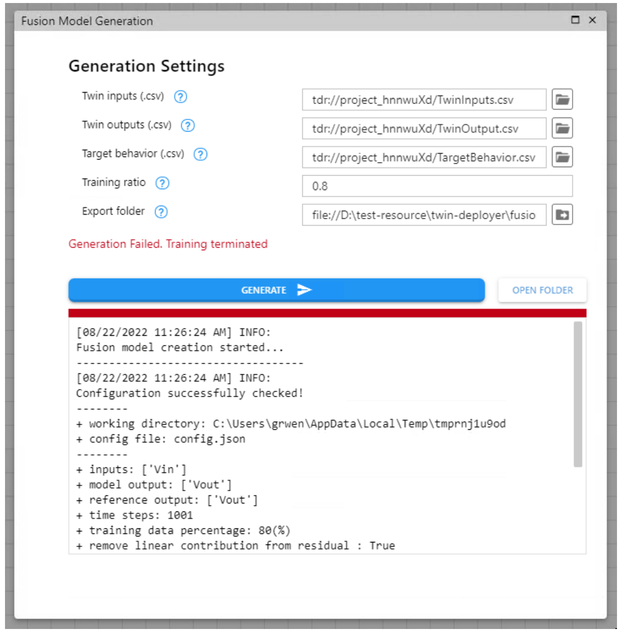
User Benefits

- Model parameter grouping (Twin, FMU) based on name
- The folding and unfolding of the parameter tree structure
- Possibility of searching the parameters
- Column selection capability

Other Enhancements

What’s New

- Updated Twin Deployer documentation in the installation
- Other Usability enhancements



Generation Settings

Twin inputs (.csv) ? tdr://project_hnnwuXd/TwinInputs.csv

Twin outputs (.csv) ? tdr://project_hnnwuXd/TwinOutput.csv

Target behavior (.csv) ? tdr://project_hnnwuXd/TargetBehavior.csv

Training ratio ? 0.8

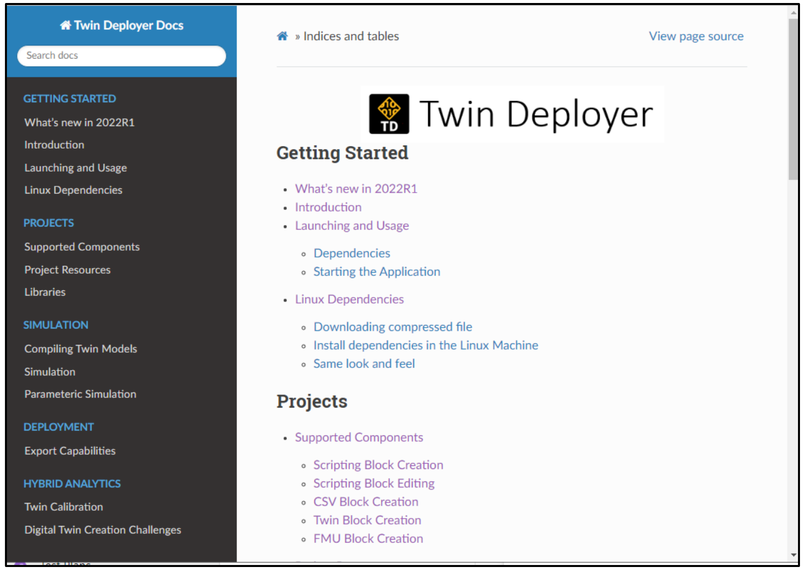
Export folder ? file://D:\test-resource\twin-deployer\fusio

Generation Failed. Training terminated

```
[08/22/2022 11:26:24 AM] INFO:
Fusion model creation started...
-----------------------------------
[08/22/2022 11:26:24 AM] INFO:
Configuration successfully checked!
--------
+ working directory: C:\Users\grwen\AppData\Local\Temp\tmprnj1u9od
+ config file: config.json
--------
+ inputs: ['Vin']
+ model output: ['Vout']
+ reference output: ['Vout']
+ time steps: 1001
+ training data percentage: 80(%)
+ remove linear contribution from residual : True
```

User Benefits

- Improved usability and stability
- The new improved style makes the documentation easier to access and consume.



Getting Started

- What’s new in 2022R1
- Introduction
- Launching and Usage
  - Dependencies
  - Starting the Application
- Linux Dependencies
  - Downloading compressed file
  - Install dependencies in the Linux Machine
  - Same look and feel

Projects

- Supported Components
  - Scripting Block Creation
  - Scripting Block Editing
  - CSV Block Creation
  - Twin Block Creation
  - FMU Block Creation

Reduced Order Model

Enhancements to Reduced Order Model and Workflows

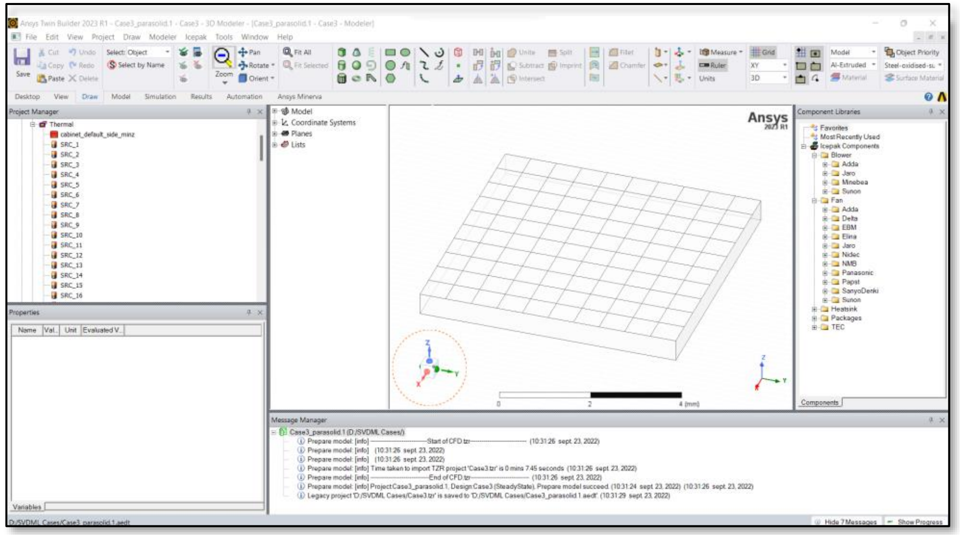
New Linear Static ROM that handles a large number of parameter

What's New

- New Linear Static ROM that handles a large number of parameter

User Benefits

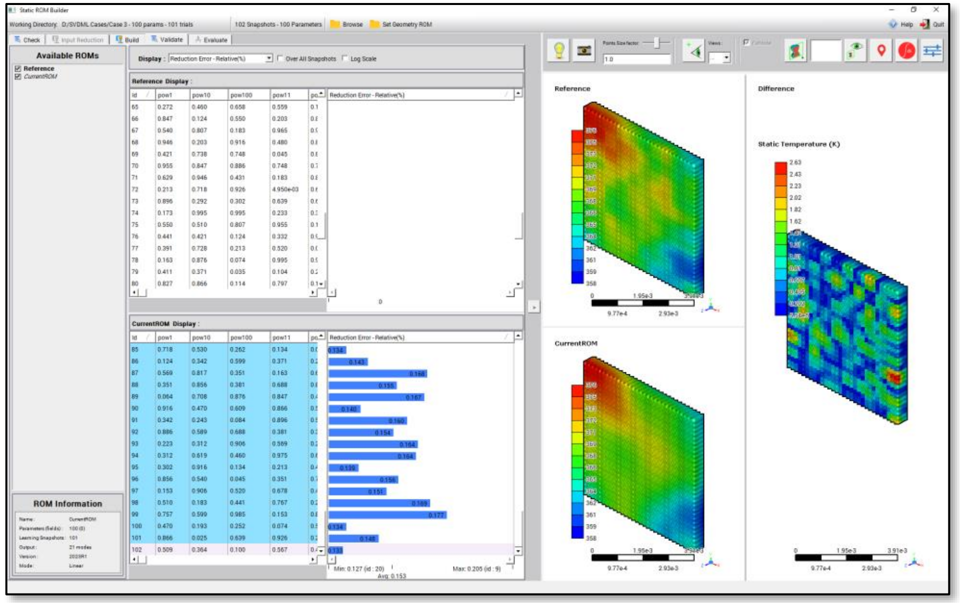
- Ideal for Electronics/Thermal applications, but also applicable to other problems.
- To characterize an n-parameter problem, a linear problem requires only n+1 snapshots.



Message Manager

```
Case3_parasolid.1 (D:/SVDML Cases/)
  Prepare model: [info] ————————Start of CFD.tzr———————— (10:31:26  sept. 23, 2022)
  Prepare model: [info]   (10:31:26  sept. 23, 2022)
  Prepare model: [info]   (10:31:26  sept. 23, 2022)
  Prepare model: [info] Time taken to import TZR project 'Case3.tzr' is 0 mins 7.45 seconds  (10:31:26  sept. 23, 2022)
  Prepare model: [info] ————————End of CFD.tzr———————— (10:31:26  sept. 23, 2022)
  Prepare model: [info] Project:Case3_parasolid.1, Design:Case3 (SteadyState), Prepare model succeed. (10:31:24  sept. 23, 2022) (10:31:26  sept. 23, 2022)
  Legacy project 'D:/SVDML Cases/Case3.tzr' is saved to 'D:/SVDML Cases/Case3_parasolid.1.aedt'. (10:31:29  sept. 23, 2022)
```

D:/SVDML Cases/Case3_parasolid.1.aedt



Working Directory: D:/SVDML Cases/Case 3 - 100 params - 101 trials

102 Snapshots - 100 Parameters

Display : Reduction Error - Relative(%) □ Over All Snapshots □ Log Scale

Reference Display :

id	pow1	pow10	pow100	pow11	po...
65	0.272	0.460	0.658	0.559	0.1
66	0.847	0.124	0.550	0.203	0.8
67	0.540	0.807	0.183	0.965	0.9
68	0.946	0.203	0.916	0.480	0.8
69	0.421	0.738	0.748	0.045	0.8
70	0.955	0.847	0.886	0.748	0.7
71	0.629	0.946	0.431	0.183	0.8
72	0.213	0.718	0.926	4.950e-03	0.6
73	0.896	0.292	0.302	0.639	0.6
74	0.173	0.995	0.995	0.233	0.3
75	0.550	0.510	0.807	0.955	0.1
76	0.441	0.421	0.124	0.332	0.5
77	0.391	0.728	0.213	0.520	0.0
78	0.163	0.876	0.074	0.995	0.5
79	0.411	0.371	0.035	0.104	0.2
80	0.827	0.866	0.114	0.797	0.1

Reduction Error - Relative(%)

0

CurrentROM Display :

id	pow1	pow10	pow100	pow11	po...	Reduction Error - Relative(%)
85	0.718	0.530	0.262	0.134	0.0	0.134
86	0.124	0.342	0.599	0.371	0.2	0.143
87	0.569	0.817	0.351	0.163	0.6	0.168
88	0.351	0.856	0.381	0.688	0.6	0.155
89	0.064	0.708	0.876	0.847	0.4	0.167
90	0.916	0.470	0.609	0.866	0.5	0.140
91	0.342	0.243	0.084	0.896	0.5	0.160
92	0.886	0.589	0.688	0.381	0.1	0.154
93	0.223	0.312	0.906	0.589	0.2	0.164
94	0.312	0.619	0.460	0.975	0.6	0.164
95	0.302	0.916	0.134	0.213	0.4	0.139
96	0.856	0.540	0.045	0.351	0.7	0.156
97	0.153	0.906	0.520	0.678	0.4	0.151
98	0.510	0.183	0.441	0.767	0.2	0.169
99	0.757	0.599	0.985	0.153	0.8	0.177
100	0.470	0.193	0.252	0.074	0.5	0.134
101	0.866	0.025	0.639	0.926	0.2	0.146
102	0.509	0.364	0.100	0.567	0.4	0.133

Min: 0.127 (id : 20) Avg: 0.153 Max: 0.205 (id : 9)

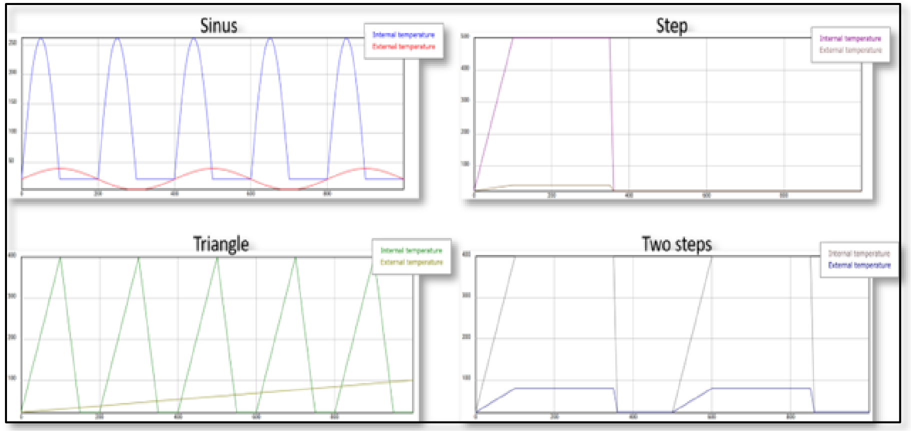
ROM Information

Name :	CurrentROM
Parameters (fields) :	100 (0)
Learning Snapshots :	101
Output :	21 modes
Version :	2023R1
Mode :	Linear

New Linear Dynamic ROM (Linear TDVF)

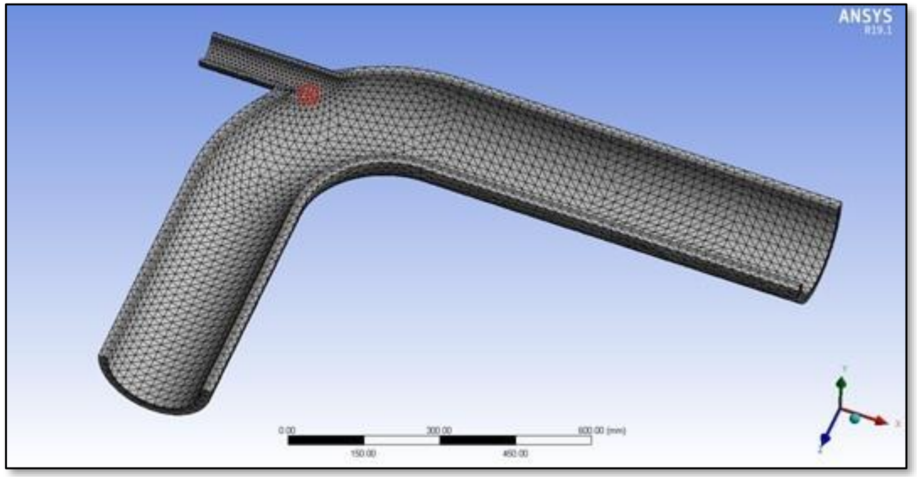
What’s New

- New Linear Dynamic ROM Linear TDVF (Linear Time Domain Vector Fitting)



User Benefits

- Linear TDVF is a very effective linear dynamic ROM algorithm making simulation much faster
- The trapezoidal rule is used for convolution making it very robust
- Field inputs are also supported.



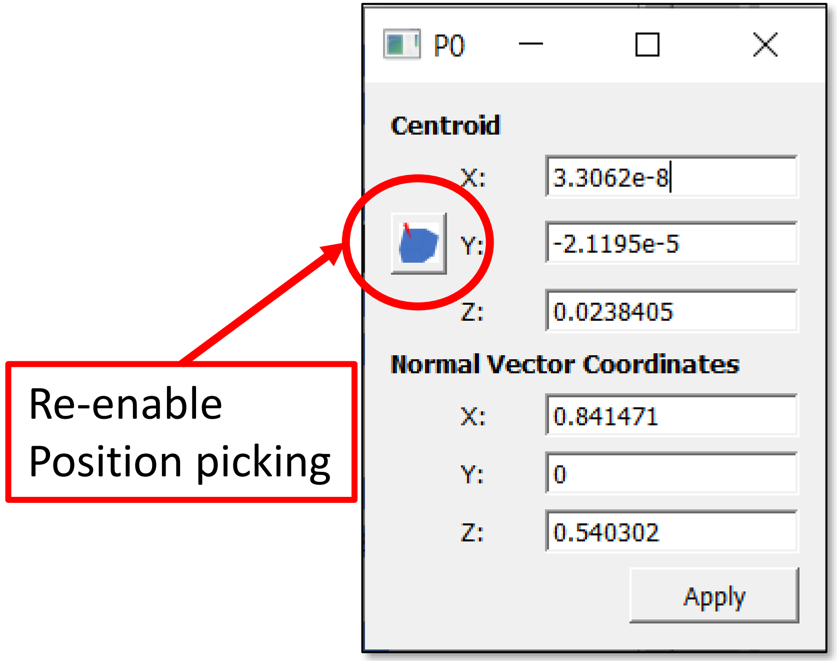
Cut-plane coordinate Specification by Orientation and Origin

What's New

- Improved Cut-plane Coordinate Specification by Orientation and Origin
- Generate a cut-plane by click (with mouse)
- Ability to edit its orientation and origin using UI

User Benefits

- Easier and better control of cut-plane coordinate specification with mouse and UI



Centroid

X: 3.3062e-8

Y: -2.1195e-5

Z: 0.0238405

Normal Vector Coordinates

X: 0.841471

Y: 0

Z: 0.540302

Re-enable Position picking

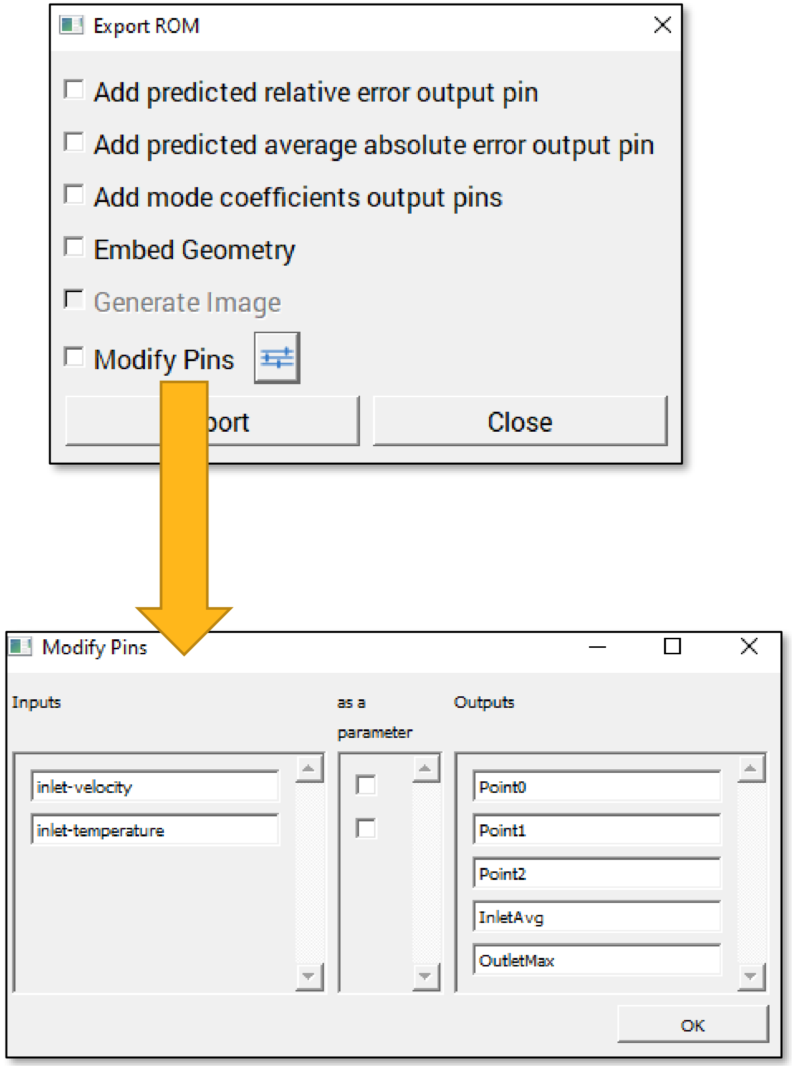
Improved Workflow for Renaming Input/Output Pins for ROMs

What's New

- Improved Workflow for renaming Input/Output Pins for ROMs

User Benefits

- The ability to rename each pin (input and output) at export-time.
- At export-time, you can specify whether a ROM input parameter will be considered as an input or parameter (e.g., for hybrid calibration) for the exported model.



Export ROM

□ Add predicted relative error output pin

□ Add predicted average absolute error output pin

□ Add mode coefficients output pins

□ Embed Geometry

□ Generate Image

□ Modify Pins

Modify Pins

Inputs	as a parameter	Outputs
inlet-velocity	□	Point0
inlet-temperature	□	Point1
		Point2
		InletAvg
		OutletMax

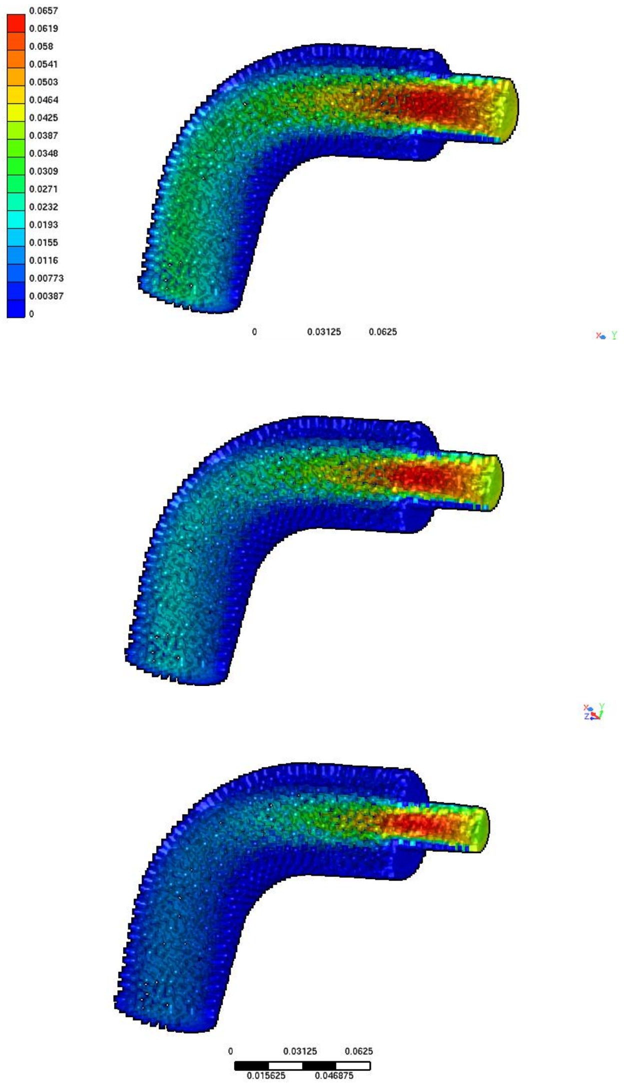
Image Generation with Geometry Deformation

What's New

- Image Generation with Geometry Deformation

User Benefits

- Geometry deformation support for image generation allows users to generate images from ROM and Twins.



Improvements on SPM ROM/Modal ROM Export from Mechanical

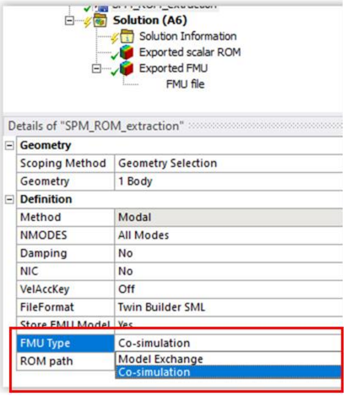
What's New

- SPM ROM/Modal Field ROM for Mechanical to Twin Builder Improvements
  - Include highest frequency value in the FMU export to suggest Max timestep in Twin Builder
  - Include solver type in the FMU/TBROM export (Model Exchange or Co-simulation)

User Benefits

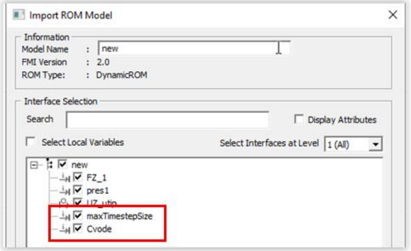
- Better usability and less error prone
- Improved stability

SPM ROM extension



Details of "SPM_ROM_extraction"

Geometry	
Scoping Method	Geometry Selection
Geometry	1 Body
Definition	
Method	Modal
NMODES	All Modes
Damping	No
NIC	No
VelAccKey	Off
FileFormat	Twin Builder SML
Store FMU Model	Yes
FMU Type	Co-simulation
ROM path	Model Exchange Co-simulation

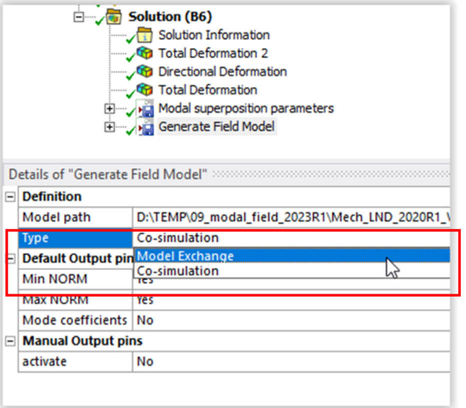


Import ROM Model

Information
Model Name : new
FMI Version : 2.0
ROM Type: : DynamicROM

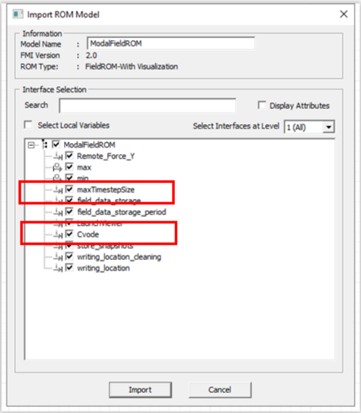
Interface Selection
Search
□ Display Attributes
□ Select Local Variables
Select Interfaces at Level 1 (All)

Modal field extension



Details of "Generate Field Model"

Definition	
Model path	D:\TEMP\09_modal_field_2023R1\Mech_LND_2020R1_\
Type	Co-simulation
Default Output pin	Model Exchange Co-simulation
Min NORM	Yes
Max NORM	Yes
Mode coefficients	No
Manual Output pins	
activate	No



Import ROM Model

Information
Model Name : ModalFieldROM
FMI Version : 2.0
ROM Type: : FieldROM-With Visualization

Interface Selection
Search
□ Display Attributes
□ Select Local Variables
Select Interfaces at Level 1 (All)

Model Building – Modelica

Enhancements to the Modelica workflow

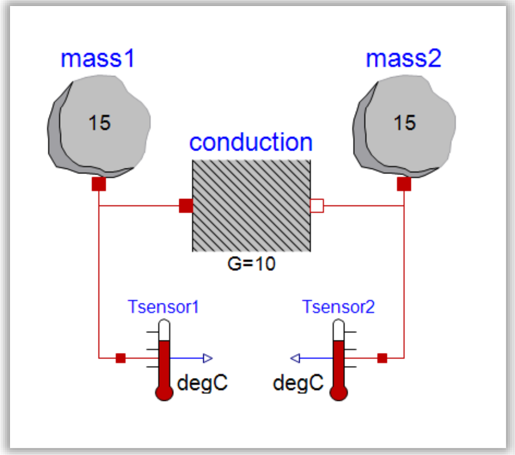
Modelica: Enhanced Diagram Graphics

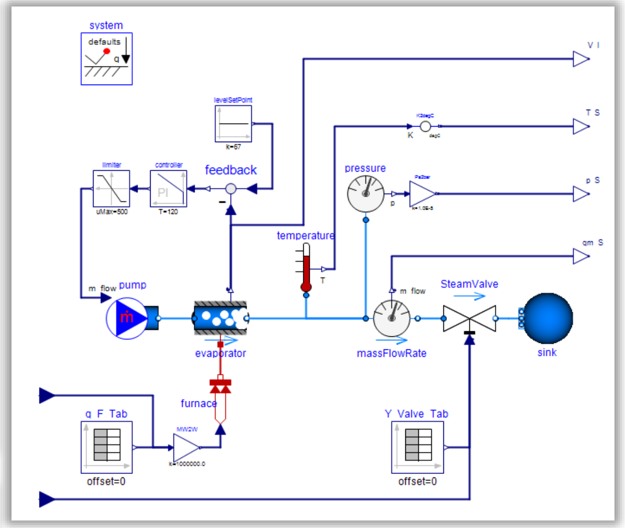
What's New

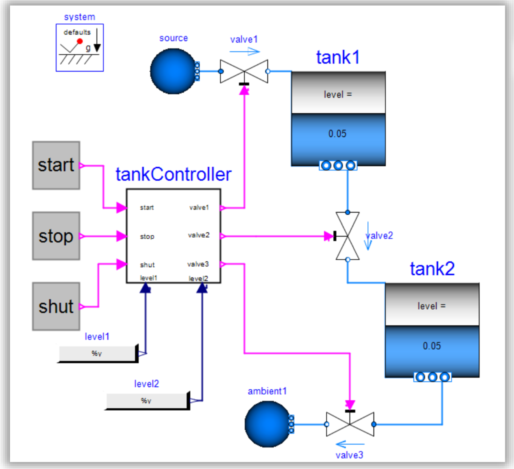
- Modern and Enhanced Graphics in Modelica Editor
  - Support for Gradient fill in component icons
  - Support for Anti-Aliasing and transparency in images
  - Support for different pattern styles and line styles
  - Improved drawing for text in icons

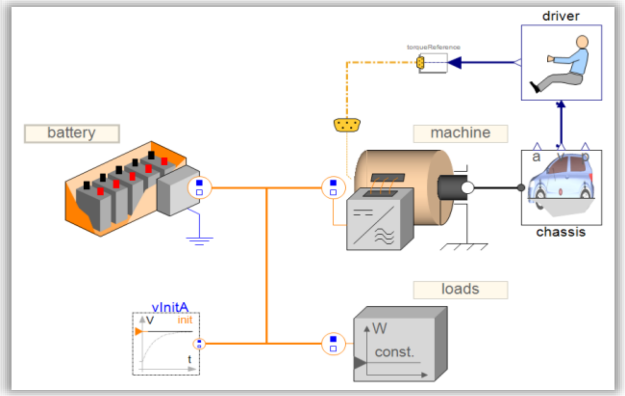
User Benefits

- Better graphics support conforms the graphics to the Modelica standard.
- It is now possible to import libraries with advanced graphics features.









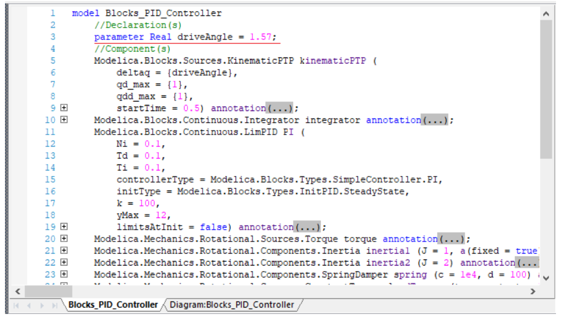
Improved Bijection in Modelica Text Editor

What’s New

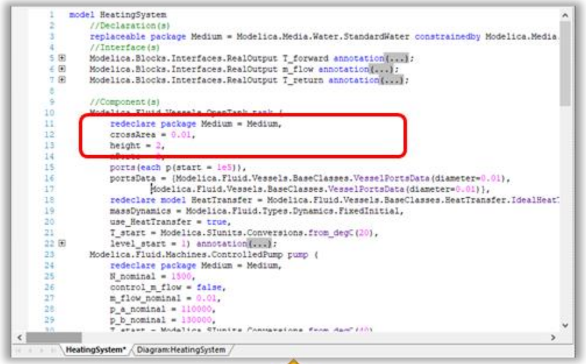
- Add new parameters or modify existing outputs in Text Editor
- Support for component parameter modification in Text Editor
- Support for “Extend” model from library in Text Editor

User Benefits

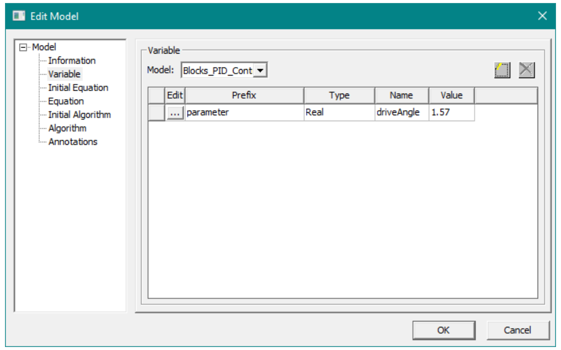
- The capability allows you to make direct design changes in either text or diagram view and have them reflected in the alternate view.
- Make Modelica model creation faster and preserver changes across both views.



```
model Blocks_PID_Controller
    //Declaration(s)
    parameter Real driveAngle = 1.57;
    //Component(s)
    Modelica.Blocks.Sources.KinematicPTP kinematicPTP (
        deltaq = {driveAngle},
        qd_max = {1},
        qdd_max = {1},
        startTime = 0.5) annotation(...);
    Modelica.Blocks.Continuous.Integrator integrator annotation(...);
    Modelica.Blocks.Continuous.LimPID PI (
        Ni = 0.1,
        Td = 0.1,
        Ti = 0.1,
        controllerType = Modelica.Blocks.Types.SimpleController.PI,
        initType = Modelica.Blocks.Types.InitPID.SteadyState,
        k = 100,
        yMax = 12,
        limitsAtInit = false) annotation(...);
    Modelica.Mechanics.Rotational.Sources.Torque torque annotation(...);
    Modelica.Mechanics.Rotational.Components.Inertia inertia1 (J = 1, a(fixed = true
    Modelica.Mechanics.Rotational.Components.Inertia inertia2 (J = 2) annotation(...
    Modelica.Mechanics.Rotational.Components.SpringDamper spring (c = 1e4, d = 100)
```



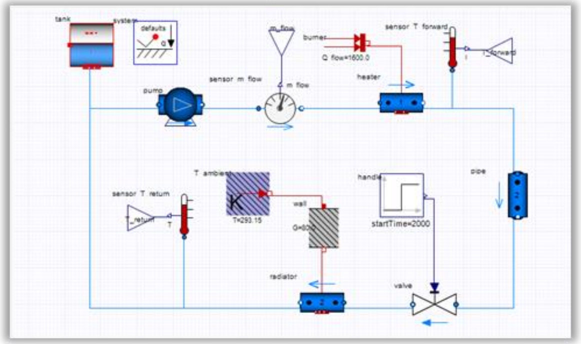
```
model HeatingSystem
    //Declaration(s)
    replaceable package Medium = Modelica.Media.Water.StandardWater constrainedby Modelica.Media.
    //Interface(s)
    Modelica.Blocks.Interfaces.RealOutput T_forward annotation(...);
    Modelica.Blocks.Interfaces.RealOutput m_flow annotation(...);
    Modelica.Blocks.Interfaces.RealOutput T_return annotation(...);

    //Component(s)
    Modelica.Fluid.Vessels.OpenTank tank (
        redeclare package Medium = Medium,
        crossArea = 0.01,
        height = 2,
        nPorts = 2,
        ports(each p(start = 1e5)),
        portsData = {Modelica.Fluid.Vessels.BaseClasses.VesselPortsData(diameter=0.01),
                Modelica.Fluid.Vessels.BaseClasses.VesselPortsData(diameter=0.01)},
        redeclare model HeatTransfer = Modelica.Fluid.Vessels.BaseClasses.HeatTransfer.IdealHeat
        massDynamics = Modelica.Fluid.Types.Dynamics.FixedInitial,
        use_HeatTransfer = true,
        T_start = Modelica.SIunits.Conversions.from_degC(20),
        level_start = 1) annotation(...);
    Modelica.Fluid.Machines.ControlledPump pump (
        redeclare package Medium = Medium,
        N_nominal = 1500,
        control_m_flow = false,
        m_flow_nominal = 0.01,
        p_a_nominal = 110000,
        p_b_nominal = 130000,
```



Variable

Model: Blocks_PID_Cont

	Edit	Prefix	Type	Name	Value
	...	parameter	Real	driveAngle	1.57



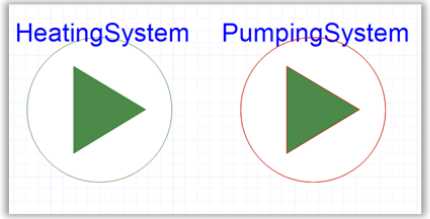
Modelica: Hierarchical Display and Editing of Parameters

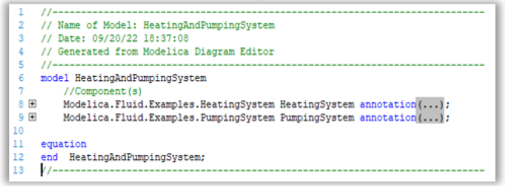
What’s New

- Hierarchical Display of Parameters
  - Internal parameters can be exposed in Modelica and other FMUs using the dot notation
  - Component dialog now supports ability to show selected internal parameters as a multi level tree control

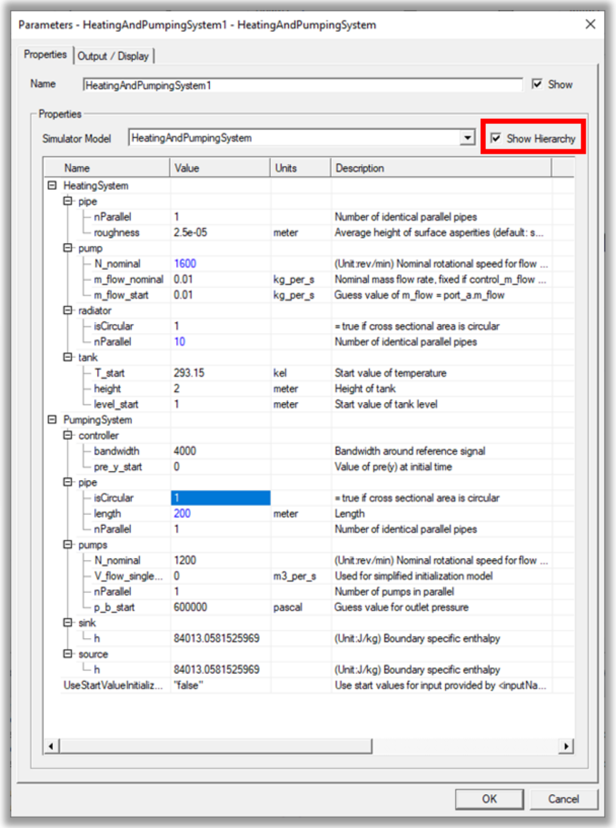
User Benefits

- Improves usability and readability





```
//-----------------------------------------------------------------------
// Name of Model: HeatingAndPumpingSystem
// Date: 09/20/22 18:37:08
// Generated from Modelica Diagram Editor
//-----------------------------------------------------------------------
model HeatingAndPumpingSystem
    //Component(s)
    Modelica.Fluid.Examples.HeatingSystem HeatingSystem annotation(...);
    Modelica.Fluid.Examples.PumpingSystem PumpingSystem annotation(...);

equation
end  HeatingAndPumpingSystem;
//-----------------------------------------------------------------------
```



Parameters - HeatingAndPumpingSystem1 - HeatingAndPumpingSystem

Properties | Output / Display

Name: HeatingAndPumpingSystem1 ☑ Show

Properties

Simulator Model: HeatingAndPumpingSystem ☑ Show Hierarchy

Name	Value	Units	Description
⊟ HeatingSystem			
⊟ pipe			
nParallel	1		Number of identical parallel pipes
roughness	2.5e-05	meter	Average height of surface asperities (default: s...
⊟ pump			
N_nominal	1600		(Unit:rev/min) Nominal rotational speed for flow ...
m_flow_nominal	0.01	kg_per_s	Nominal mass flow rate, fixed if control_m_flow ...
m_flow_start	0.01	kg_per_s	Guess value of m_flow = port_a.m_flow
⊟ radiator			
isCircular	1		= true if cross sectional area is circular
nParallel	10		Number of identical parallel pipes
⊟ tank			
T_start	293.15	kel	Start value of temperature
height	2	meter	Height of tank
level_start	1	meter	Start value of tank level
⊟ PumpingSystem			
⊟ controller			
bandwidth	4000		Bandwidth around reference signal
pre_y_start	0		Value of pre(y) at initial time
⊟ pipe			
isCircular	1		= true if cross sectional area is circular
length	200	meter	Length
nParallel	1		Number of identical parallel pipes
⊟ pumps			
N_nominal	1200		(Unit:rev/min) Nominal rotational speed for flow ...
V_flow_single...	0	m3_per_s	Used for simplified initialization model
nParallel	1		Number of pumps in parallel
p_b_start	600000	pascal	Guess value for outlet pressure
⊟ sink			
h	84013.0581525969		(Unit:J/kg) Boundary specific enthalpy
⊟ source			
h	84013.0581525969		(Unit:J/kg) Boundary specific enthalpy
UseStartValueInitializ...	"false"		Use start values for input provided by <inputNa...

Solver and Model Library

Enhancements in Solver and Model Library

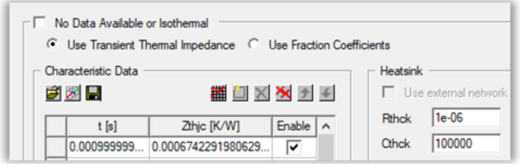
Usability Improvements for Device Characterization Wizard

What's New

- Simplified Thermal Fitting Page
- New selection of Fitting Type
- Simplified Selection of Channel Type

User Benefits

- Better usability
- Gradient fitting is a new option that work better in some cases
- More intuitive selection for choosing the channel type



□ No Data Available or Isothermal

● Use Transient Thermal Impedance ○ Use Fraction Coefficients

Characteristic Data

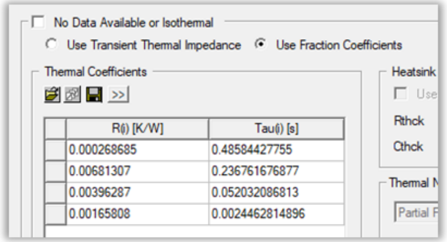
t [s]	Zthjc [K/W]	Enable
0.000999999...	0.0006742291980629...	☑

Heatsink

□ Use external network

Rthck: 1e-06

Cthck: 100000



□ No Data Available or Isothermal

○ Use Transient Thermal Impedance ● Use Fraction Coefficients

Thermal Coefficients

R(i) [K/W]	Tau(i) [s]
0.000268685	0.48584427755
0.00681307	0.236761676877
0.00396287	0.052032086813
0.00165808	0.0024462814896

Heatsink

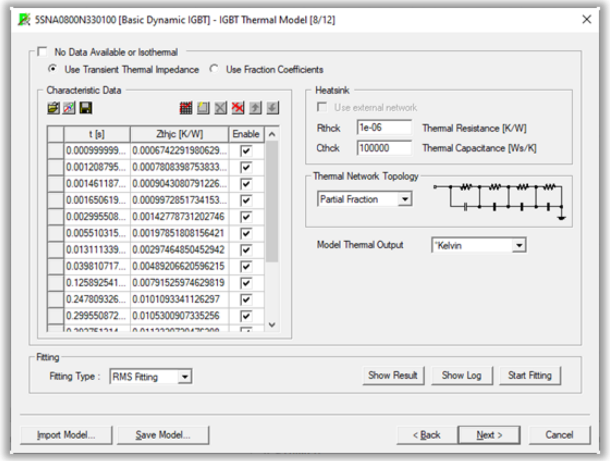
□ Use

Rthck

Cthck

Thermal N

Partial F



5SNA0800N330100 [Basic Dynamic IGBT] - IGBT Thermal Model [8/12]

□ No Data Available or Isothermal

● Use Transient Thermal Impedance ○ Use Fraction Coefficients

Characteristic Data

t [s]	Zthjc [K/W]	Enable
0.000999999...	0.0006742291980629...	☑
0.001208795...	0.0007808398753833...	☑
0.001461187...	0.0009043080791226...	☑
0.001650619...	0.0009972851734153...	☑
0.002995508...	0.00142778731202746	☑
0.005510315...	0.00197851808156421	☑
0.013111339...	0.00297464850452942	☑
0.039810717...	0.00489206620596215	☑
0.125892541...	0.00791525974629819	☑
0.247809326...	0.0101093341126297	☑
0.299550872...	0.0105300907335256	☑

Heatsink

□ Use external network

Rthck: 1e-06 Thermal Resistance [K/W]

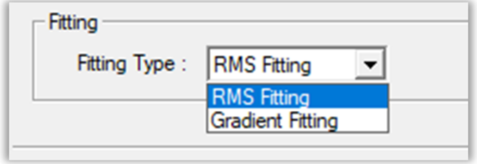
Cthck: 100000 Thermal Capacitance [Ws/K]

Thermal Network Topology: Partial Fraction

Model Thermal Output: °Kelvin

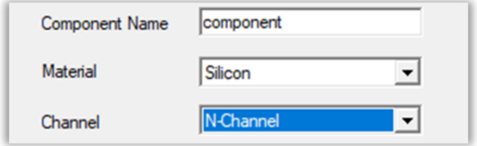
Fitting

Fitting Type : RMS Fitting



Fitting

Fitting Type : RMS Fitting



Component Name: component

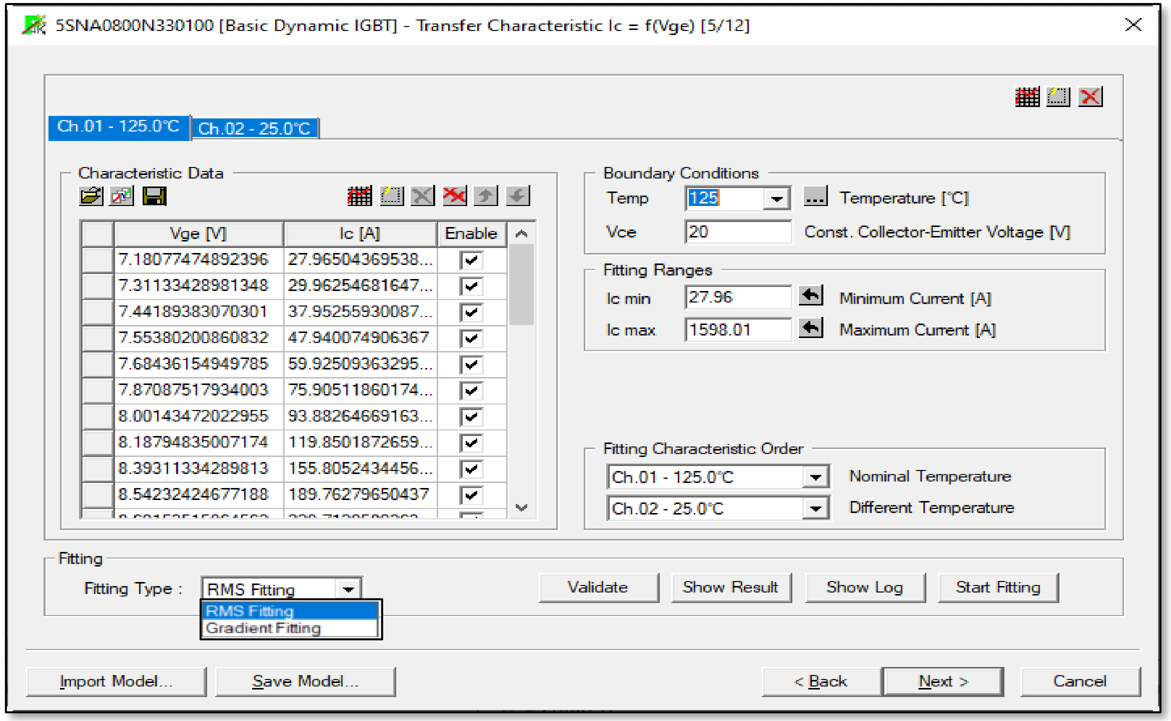
Material: Silicon

Channel: N-Channel

Support for Gradient Fitting in Device Characterization

What’s New

- Support for Gradient Fitting in Device Characterization



5SNA0800N330100 [Basic Dynamic IGBT] - Transfer Characteristic Ic = f(Vge) [5/12]

Ch.01 - 125.0°C | Ch.02 - 25.0°C

Characteristic Data

Vge [V]	Ic [A]	Enable
7.18077474892396	27.96504369538...	☑
7.31133428981348	29.96254681647...	☑
7.44189383070301	37.95255930087...	☑
7.55380200860832	47.940074906367	☑
7.68436154949785	59.92509363295...	☑
7.87087517934003	75.90511860174...	☑
8.00143472022955	93.88264669163...	☑
8.18794835007174	119.8501872659...	☑
8.39311334289813	155.8052434456...	☑
8.54232424677188	189.76279650437	☑

Boundary Conditions

Temp: 125 — Temperature [°C]

Vce: 20 — Const. Collector-Emitter Voltage [V]

Fitting Ranges

Ic min: 27.96 — Minimum Current [A]

Ic max: 1598.01 — Maximum Current [A]

Fitting Characteristic Order

Ch.01 - 125.0°C — Nominal Temperature

Ch.02 - 25.0°C — Different Temperature

Fitting

Fitting Type : RMS Fitting

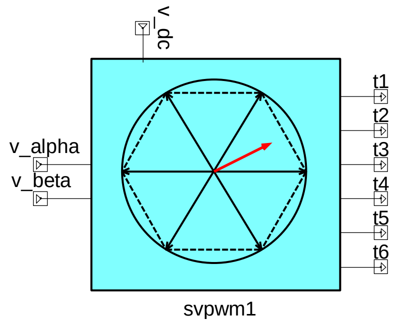
User Benefits

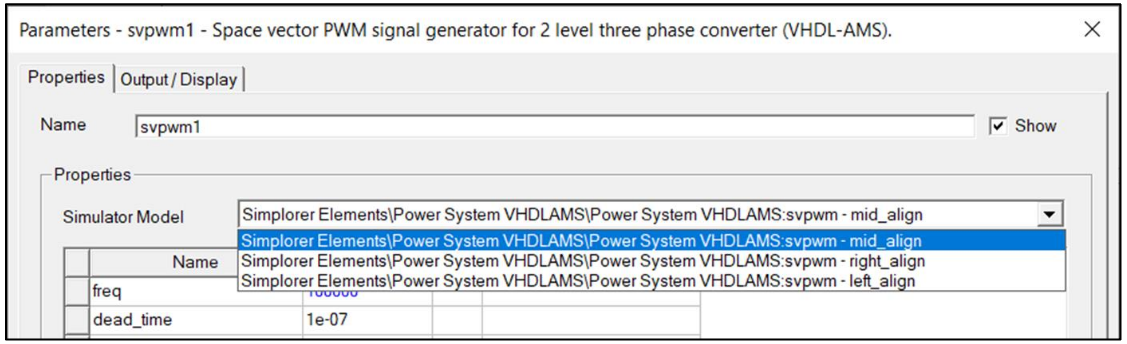
- In some cases, it is advantageous to use Gradient Fitting instead of RMS Fit:
  - Results are not satisfying, or fitting takes too much time.
  - Some model parameters run out of the expected scope.
  - Simulation using the final component shows convergence issues.
  - Find the best compromise of a general unsatisfactory fitting result.

New Carrier Wave Options in the SVPWM Component

What’s New

- New Carrier Wave Options in the SVPWM Component





Parameters - svpwm1 - Space vector PWM signal generator for 2 level three phase converter (VHDL-AMS).

Name: svpwm1 ☑ Show

Properties

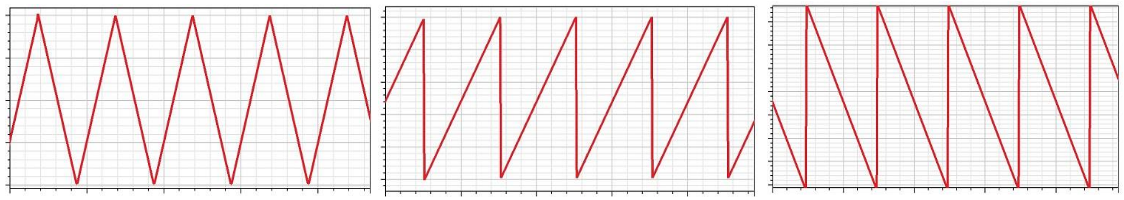
Simulator Model: Simplorer Elements\Power System VHDLAMS\Power System VHDLAMS:svpwm - mid_align

- Simplorer Elements\Power System VHDLAMS\Power System VHDLAMS:svpwm - mid_align
- Simplorer Elements\Power System VHDLAMS\Power System VHDLAMS:svpwm - right_align
- Simplorer Elements\Power System VHDLAMS\Power System VHDLAMS:svpwm - left_align

Name	
freq	100000
dead_time	1e-07

User Benefits

- This new option of carrier wave allows for all three options:
  - mid_align – triangle source for carrier wave.
  - right_align – right sawtooth source for carrier wave.
  - left_align – left sawtooth source for carrier wave.



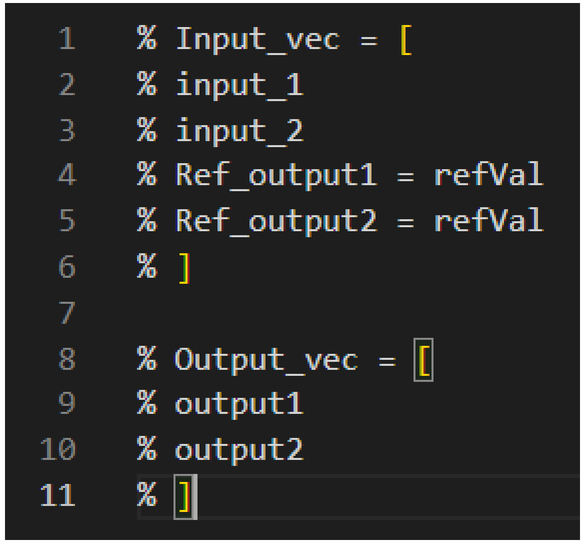
Enhancements for MATLAB Workflows

What's New

- Simulink Co-simulation update in Twin Builder
  - Added support for MATLAB versions R2021b and R2022a.
  - Removed older versions R2017b and R2018a.
- LTI toolkit export for MATLAB now contains information for input and output name and reference values

User Benefits

- Additional information has been added to the m-file to better support MATLAB users.
- Easier reuse of LTI-ROMs in MATLAB/Simulink



```
% Input_vec = [
% input_1
% input_2
% Ref_output1 = refVal
% Ref_output2 = refVal
% ]

% Output_vec = [
% output1
% output2
% ]
```

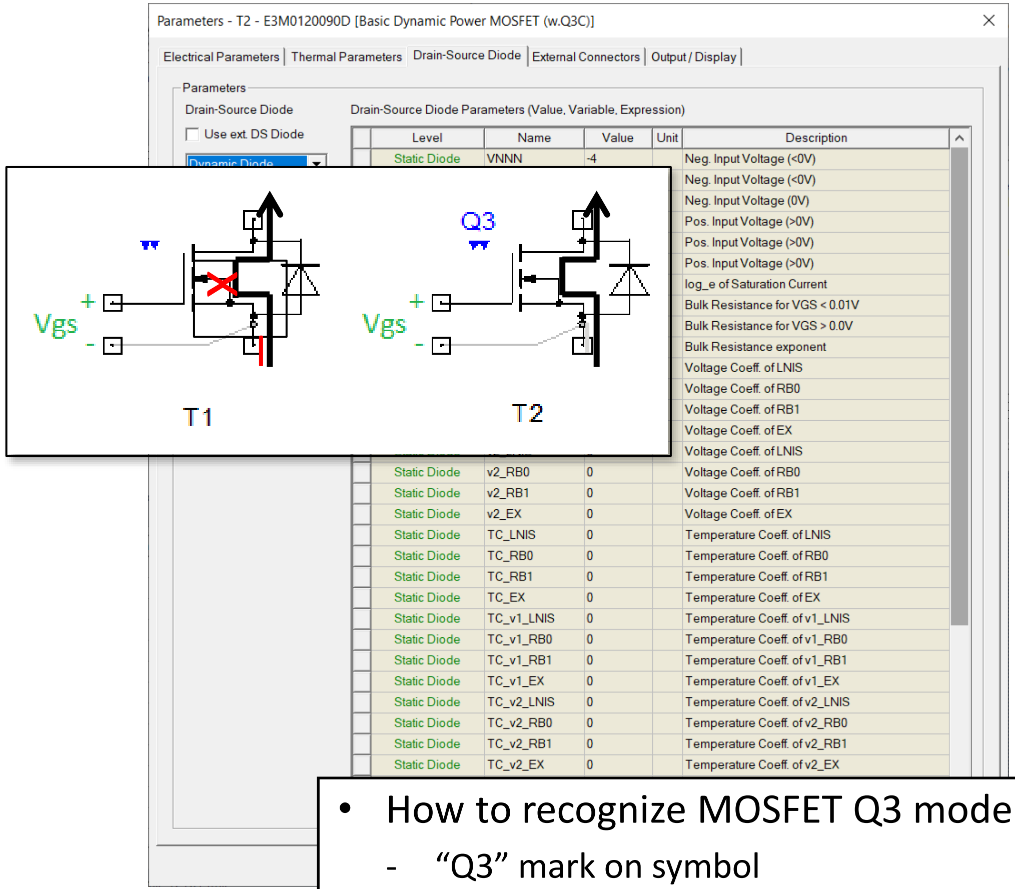
New Basic Dynamics MOSFET Model with Q3 control (Beta)

What's New

- New Basic Dynamics MOSFET Model with Q3 control (Beta)
- Q3 model can only be generated by Characterization Wizard through a special workflow
- Regular wizard workflow still generate the old MOSFET model with no reverse conduction

User Benefits

- Regular MOSFET models in TB cannot conduct reverse current through transistor and a new basic dynamic model MOSFET_Q3 is added to solve this limit



Parameters - T2 - E3M0120090D [Basic Dynamic Power MOSFET (w.Q3C)]

Parameters

Drain-Source Diode

□ Use ext. DS Diode

Dynamic Diode

Drain-Source Diode Parameters (Value, Variable, Expression)

Level	Name	Value	Unit	Description
Static Diode	VNNN	-4		Neg. Input Voltage (<0V)
				Neg. Input Voltage (<0V)
				Neg. Input Voltage (0V)
				Pos. Input Voltage (>0V)
				Pos. Input Voltage (>0V)
				Pos. Input Voltage (>0V)
				log_e of Saturation Current
				Bulk Resistance for VGS < 0.01V
				Bulk Resistance for VGS > 0.0V
				Bulk Resistance exponent
				Voltage Coeff. of LNIS
				Voltage Coeff. of RB0
				Voltage Coeff. of RB1
				Voltage Coeff. of EX
				Voltage Coeff. of LNIS
Static Diode	v2_RB0	0		Voltage Coeff. of RB0
Static Diode	v2_RB1	0		Voltage Coeff. of RB1
Static Diode	v2_EX	0		Voltage Coeff. of EX
Static Diode	TC_LNIS	0		Temperature Coeff. of LNIS
Static Diode	TC_RB0	0		Temperature Coeff. of RB0
Static Diode	TC_RB1	0		Temperature Coeff. of RB1
Static Diode	TC_EX	0		Temperature Coeff. of EX
Static Diode	TC_v1_LNIS	0		Temperature Coeff. of v1_LNIS
Static Diode	TC_v1_RB0	0		Temperature Coeff. of v1_RB0
Static Diode	TC_v1_RB1	0		Temperature Coeff. of v1_RB1
Static Diode	TC_v1_EX	0		Temperature Coeff. of v1_EX
Static Diode	TC_v2_LNIS	0		Temperature Coeff. of v2_LNIS
Static Diode	TC_v2_RB0	0		Temperature Coeff. of v2_RB0
Static Diode	TC_v2_RB1	0		Temperature Coeff. of v2_RB1
Static Diode	TC_v2_EX	0		Temperature Coeff. of v2_EX

- How to recognize MOSFET Q3 model
  - "Q3" mark on symbol
  - "(w. Q3C)" on component parameter page
  - Extended list of static parameters in tab "Drain-Source Diode"

General UI/UX Enhancements

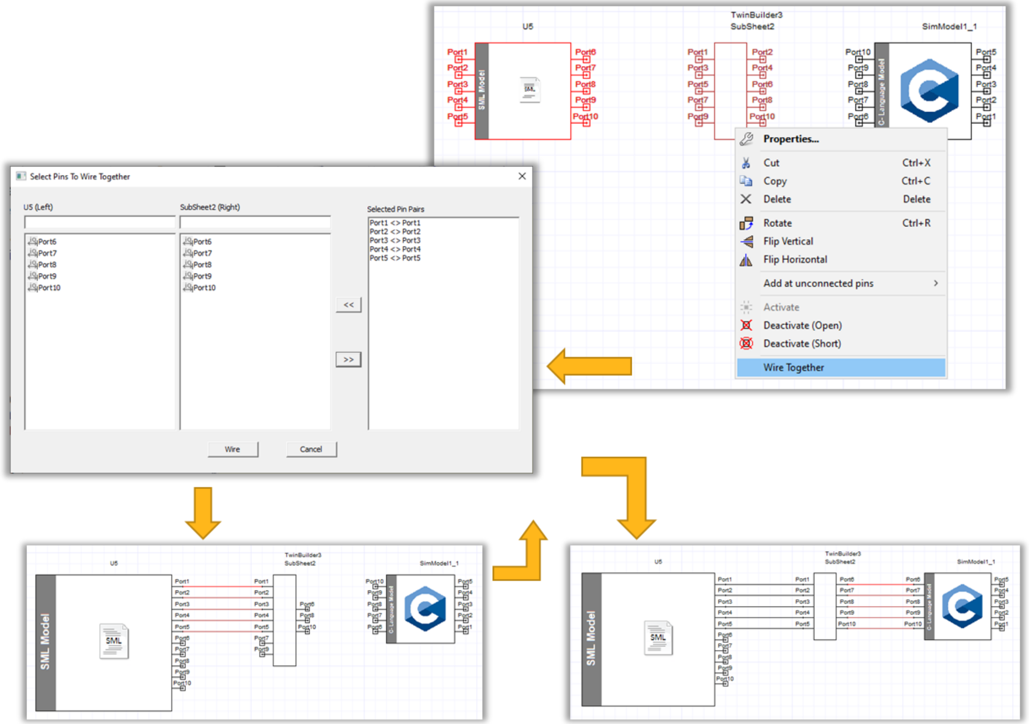
Wire Together: Easier Pin Connection for Large Components

What's New

- Wire Together Two Large Components with the help of UI
- Supports pins with direction (in, out)
- Allows Filtering on large pin list
- Works also with rectangular symbols
- Multi select to connect many pins

User Benefits

- Improved usability for connecting components, particularly those with large number of input and output pins



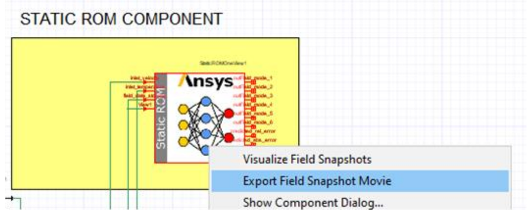
Twin Builder ROM Animation Video Export

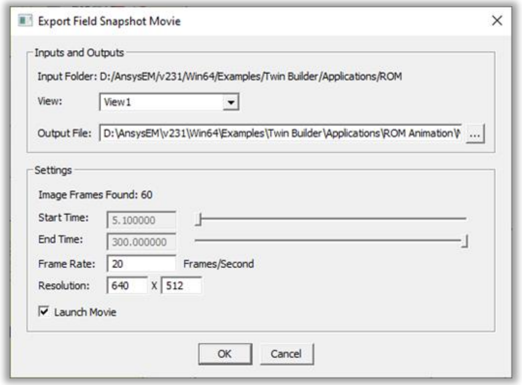
What’s New

- Export ROM Field Snapshot Animation as a movie
- New Script Command ExportAsMovie() is also available

User Benefits

- Easily create and share ROM animation from Twin Builder


STATIC ROM COMPONENT


Export Field Snapshot Movie

Inputs and Outputs

Input Folder: D:/AnsysEM/v231/Win64/Examples/Twin Builder/Applications/ROM

View: View1

Output File: D:\AnsysEM\v231\Win64\Examples\Twin Builder\Applications\ROM Animation\

Settings

Image Frames Found: 60

Start Time: 5.100000

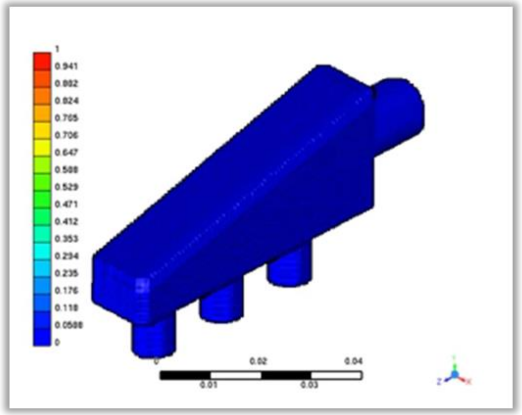
End Time: 300.000000

Frame Rate: 20 Frames/Second

Resolution: 640 X 512

☑ Launch Movie

Exported Movie File



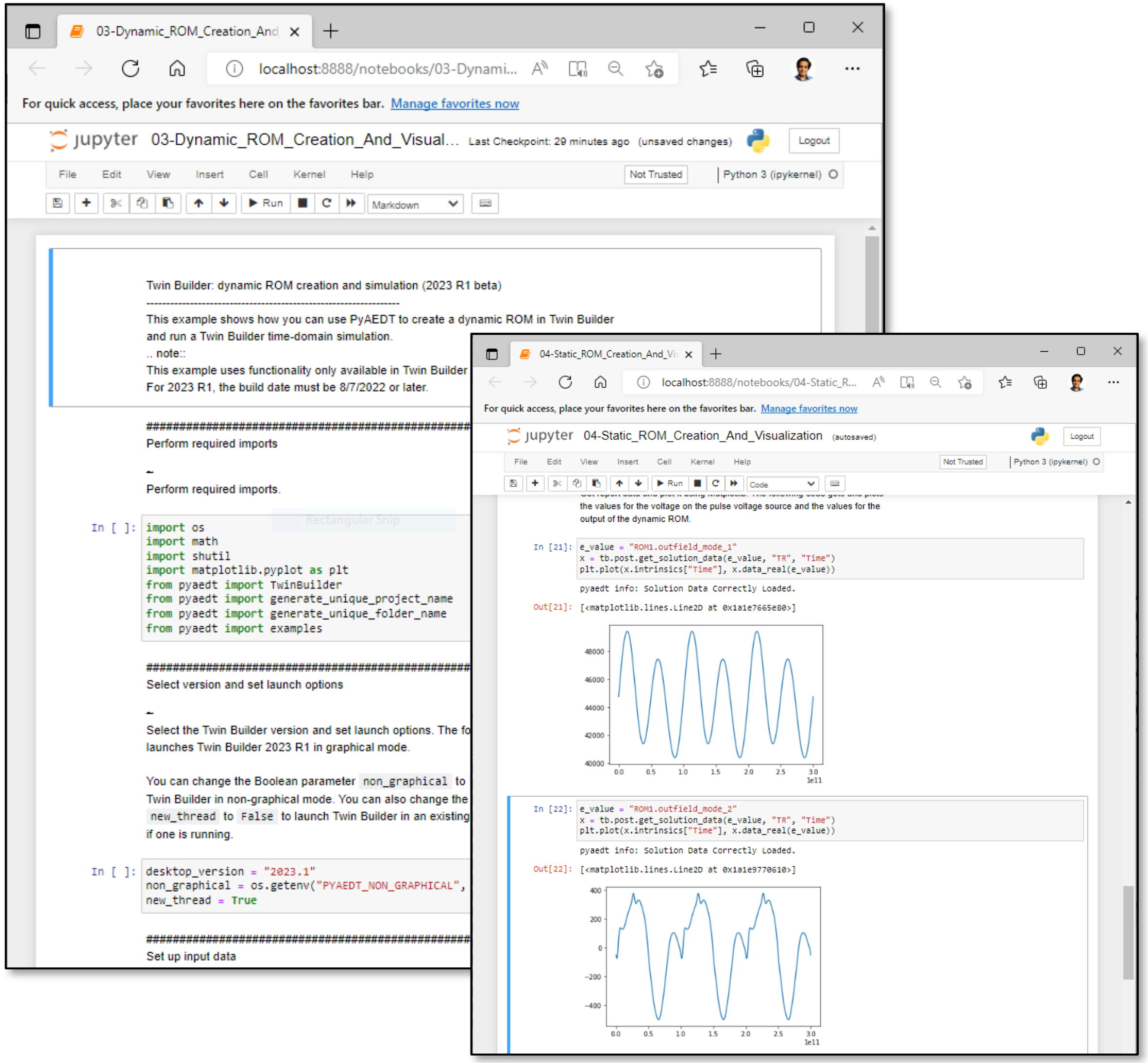
Automation with pyAnsys/pyAEDT: ROM creation using python

What's New

- New example for creating Static ROM and Dynamic ROM

User Benefits

- Scripting examples on GitHub/pyAnsys can run on command line or in a Jupyter Notebook
- Automate workflows for easier ROM creation


03-Dynamic_ROM_Creation_And
localhost:8888/notebooks/03-Dynami...
For quick access, place your favorites here on the favorites bar. Manage favorites now
jupyter 03-Dynamic_ROM_Creation_And_Visual... Last Checkpoint: 29 minutes ago (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Help
Not Trusted
Python 3 (ipykernel)
Run
Markdown
Rectangular Snip
04-Static_ROM_Creation_And_Vi
localhost:8888/notebooks/04-Static_R...
For quick access, place your favorites here on the favorites bar. Manage favorites now
jupyter 04-Static_ROM_Creation_And_Visualization (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Help
Not Trusted
Python 3 (ipykernel)
Run
Code
48000
46000
44000
42000
40000
0.0
0.5
1.0
1.5
2.0
2.5
3.0
1e11
400
200
0
−200
−400
0.0
0.5
1.0
1.5
2.0
2.5
3.0
1e11

Twin Builder: dynamic ROM creation and simulation (2023 R1 beta)

This example shows how you can use PyAEDT to create a dynamic ROM in Twin Builder and run a Twin Builder time-domain simulation.
.. note::
This example uses functionality only available in Twin Builder
For 2023 R1, the build date must be 8/7/2022 or later.

##
Perform required imports

~
Perform required imports.

```
In [ ]: import os
        import math
        import shutil
        import matplotlib.pyplot as plt
        from pyaedt import TwinBuilder
        from pyaedt import generate_unique_project_name
        from pyaedt import generate_unique_folder_name
        from pyaedt import examples
```

##
Select version and set launch options

~
Select the Twin Builder version and set launch options. The fo
launches Twin Builder 2023 R1 in graphical mode.

You can change the Boolean parameter `non_graphical` to
Twin Builder in non-graphical mode. You can also change the
`new_thread` to `False` to launch Twin Builder in an existing
if one is running.

```
In [ ]: desktop_version = "2023.1"
        non_graphical = os.getenv("PYAEDT_NON_GRAPHICAL",
        new_thread = True
```

##
Set up input data

the values for the voltage on the pulse voltage source and the values for the output of the dynamic ROM.

```
In [21]: e_value = "ROM1.outfield_mode_1"
         x = tb.post.get_solution_data(e_value, "TR", "Time")
         plt.plot(x.intrinsics["Time"], x.data_real(e_value))

         pyaedt info: Solution Data Correctly Loaded.

Out[21]: [<matplotlib.lines.Line2D at 0x1a1e7665e80>]
```

```
In [22]: e_value = "ROM1.outfield_mode_2"
         x = tb.post.get_solution_data(e_value, "TR", "Time")
         plt.plot(x.intrinsics["Time"], x.data_real(e_value))

         pyaedt info: Solution Data Correctly Loaded.

Out[22]: [<matplotlib.lines.Line2D at 0x1a1e9770610>]
```

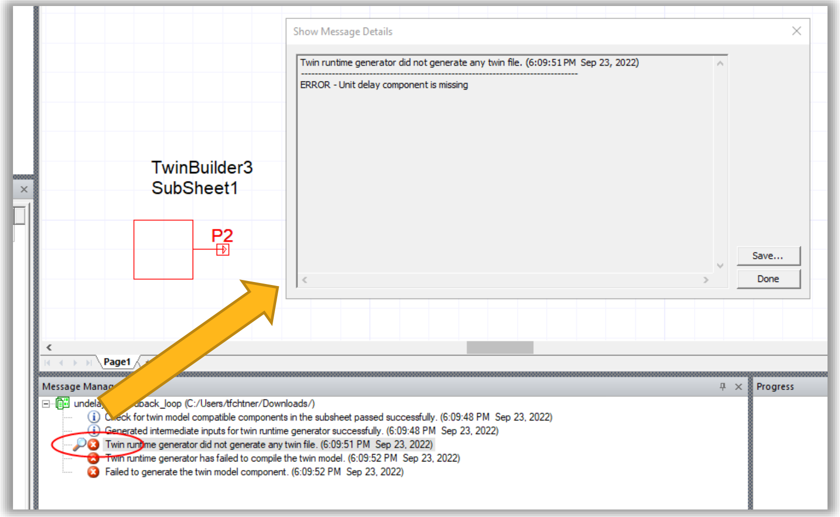
Easy Debug and Error Log Access

What's New

- Twin Generation Log
- Advanced log for Simulation

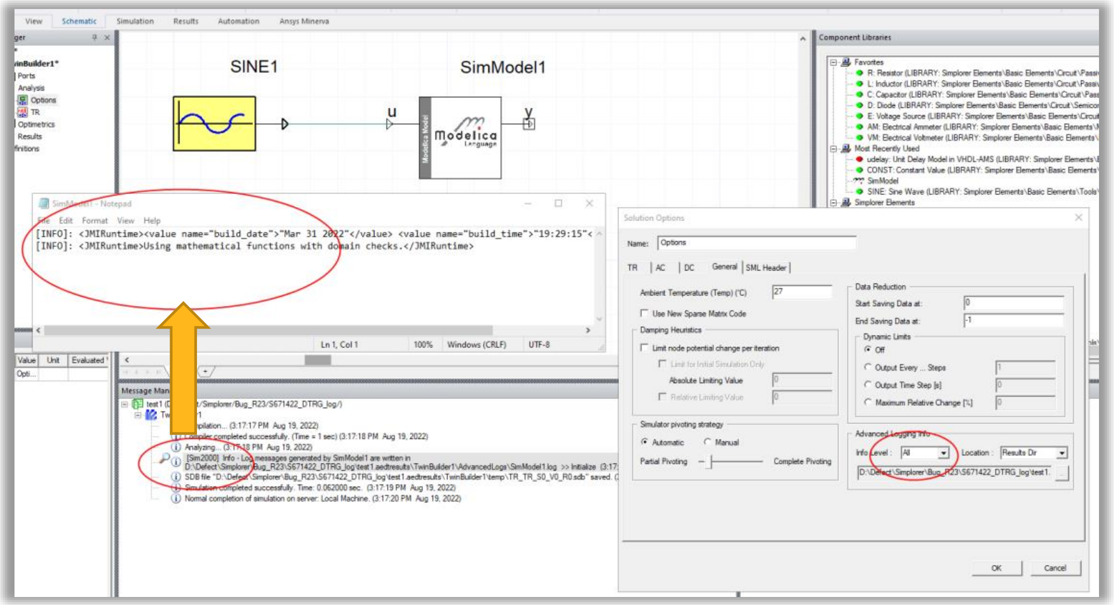
User Benefits

- Magnify icon now shows a detailed log window
- Easy access to debug and logs for easier debugging



```
Twin runtime generator did not generate any twin file. (6:09:51 PM  Sep 23, 2022)
---------------------------------------------------------------------------
ERROR - Unit delay component is missing
```

```
undela  back_loop (C:/Users/tfchtner/Downloads/)
  Check for twin model compatible components in the subsheet passed successfully. (6:09:48 PM  Sep 23, 2022)
  Generated intermediate inputs for twin runtime generator successfully. (6:09:48 PM  Sep 23, 2022)
  Twin runtime generator did not generate any twin file. (6:09:51 PM  Sep 23, 2022)
  Twin runtime generator has failed to compile the twin model. (6:09:52 PM  Sep 23, 2022)
  Failed to generate the twin model component. (6:09:52 PM  Sep 23, 2022)
```


View
Schematic
Simulation
Results
Automation
Ansys Minerva
inBuilder1*
Ports
Analysis
Options
TR
Optimetrics
Results
finitions
SINE1
SimModel1
u
y
Modelica Language
Modelica Model
Component Libraries
Favorites
R: Resistor (LIBRARY: Simplorer Elements\Basic Elements\Circuit\Passi
L: Inductor (LIBRARY: Simplorer Elements\Basic Elements\Circuit\Passi
C: Capacitor (LIBRARY: Simplorer Elements\Basic Elements\Circuit\Pass
D: Diode (LIBRARY: Simplorer Elements\Basic Elements\Circuit\Semicon
E: Voltage Source (LIBRARY: Simplorer Elements\Basic Elements\Circuit
AM: Electrical Ammeter (LIBRARY: Simplorer Elements\Basic Elements\
VM: Electrical Voltmeter (LIBRARY: Simplorer Elements\Basic Elements\
Most Recently Used
udelay: Unit Delay Model in VHDL-AMS (LIBRARY: Simplorer Elements\
CONST: Constant Value (LIBRARY: Simplorer Elements\Basic Elements\
SimModel
SINE: Sine Wave (LIBRARY: Simplorer Elements\Basic Elements\Tools\
Simplorer Elements
SimModel1 - Notepad
File Edit Format View Help
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8
Value
Unit
Evaluated
Opti...
Message Man
Solution Options
TR
AC
DC
General
SML Header
OK
Cancel

```
[INFO]: <JMIRuntime><value name="build_date">"Mar 31 2022"</value> <value name="build_time">"19:29:15"<
[INFO]: <JMIRuntime>Using mathematical functions with domain checks.</JMIRuntime>
```

```
test1 (D  t/Simplorer/Bug_R23/S671422_DTRG_log/)
  Tw     1
    pilation... (3:17:17 PM  Aug 19, 2022)
    Compiler completed successfully. (Time = 1 sec) (3:17:18 PM  Aug 19, 2022)
    Analyzing... (3:17:18 PM  Aug 19, 2022)
    [Sim2000]  Info - Log messages generated by SimModel1 are written in
    D:\Defect\Simplorer\Bug_R23\S671422_DTRG_log\test1.aedtresults\TwinBuilder1\AdvancedLogs\SimModel1.log  >> Initialize  (3:17
    SDB file "D:\Defect\Simplorer\Bug_R23\S671422_DTRG_log\test1.aedtresults\TwinBuilder1\temp\TR_TR_S0_V0_R0.sdb" saved. (
    Simulation completed successfully. Time: 0.062000 sec.  (3:17:19 PM  Aug 19, 2022)
    Normal completion of simulation on server: Local Machine. (3:17:20 PM  Aug 19, 2022)
```

Name: Options

Ambient Temperature (Temp) (°C): 27

□ Use New Sparse Matrix Code

Damping Heuristics
□ Limit node potential change per iteration
□ Limit for Initial Simulation Only
Absolute Limiting Value: 0
□ Relative Limiting Value: 0

Simulator pivoting strategy
● Automatic ○ Manual
Partial Pivoting — Complete Pivoting

Data Reduction
Start Saving Data at: 0
End Saving Data at: -1

Dynamic Limits
● Off
○ Output Every ... Steps: 1
○ Output Time Step [s]: 0
○ Maximum Relative Change [%]: 0

Advanced Logging Info
Info Level: All
Location: Results Dir
D:\Defect\Simplorer\Bug_R23\S671422_DTRG_log\test1.

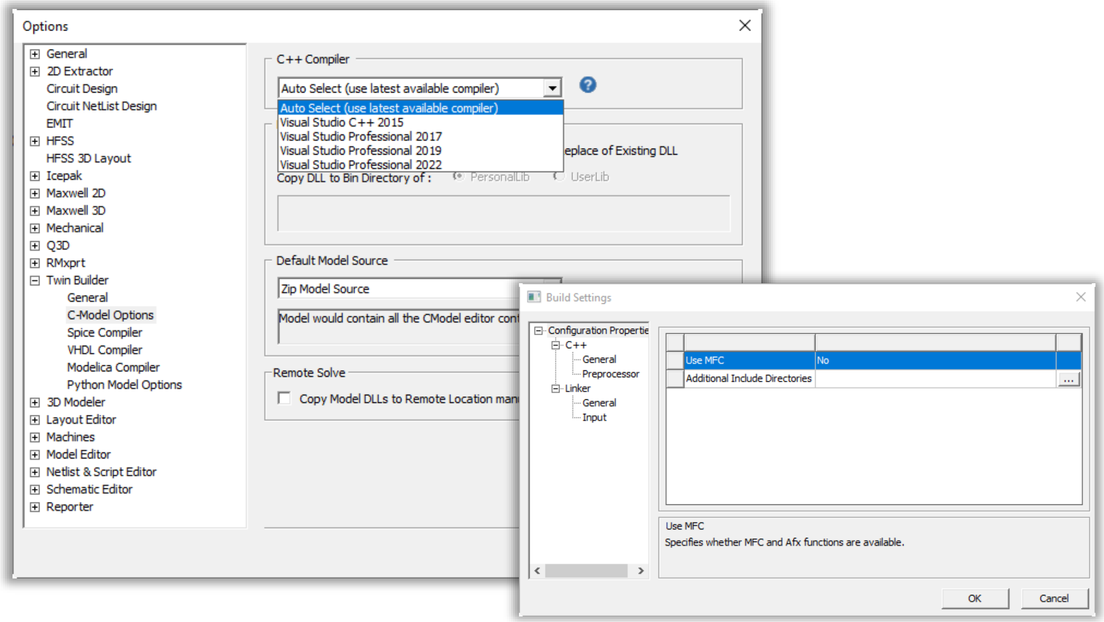
Compiler Upgrades for 2023 R1

What’s New

- Compiler Upgrades for 2023 R1 (Visual C++ 2022)
- Modelica users can now use an external version of GCC for compilation of Modelica models

User Benefits

- Added Support for Visual C++ 2022
  - Modelica support stays at Visual C++ 2019 for now
  - Support for Visual C++ 2013 dropped
  - Installation of MFC libraries now optional and off by default
  - Available for C-model, VHDL and Python models



C++ Compiler

Auto Select (use latest available compiler)

Auto Select (use latest available compiler)
Visual Studio C++ 2015
Visual Studio Professional 2017
Visual Studio Professional 2019
Visual Studio Professional 2022

eplace of Existing DLL

Copy DLL to Bin Directory of : ● PersonalLib ○ UserLib

Default Model Source

Zip Model Source

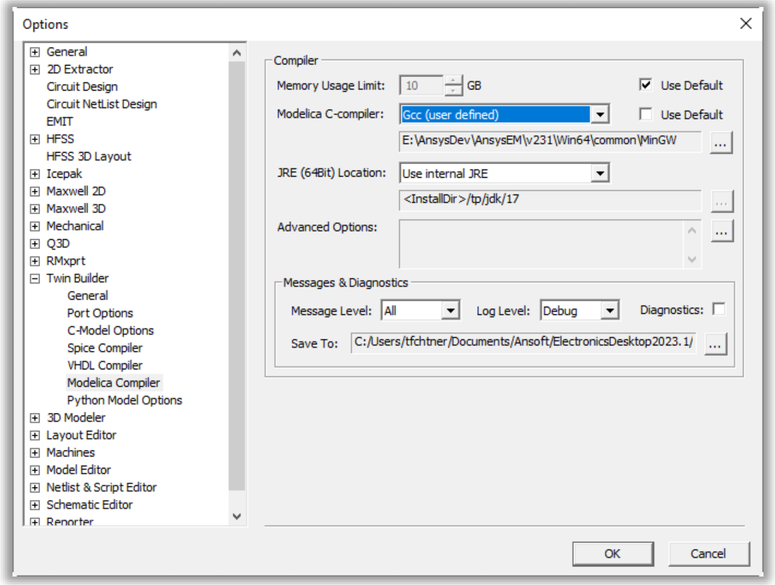
Model would contain all the CModel editor cont

Remote Solve

□ Copy Model DLLs to Remote Location man

Use MFC	No
Additional Include Directories	...

Use MFC
Specifies whether MFC and Afx functions are available.



Compiler

Memory Usage Limit: 10 GB ☑ Use Default

Modelica C-compiler: Gcc (user defined) □ Use Default

E:\AnsysDev\AnsysEM\v231\Win64\common\MinGW ...

JRE (64Bit) Location: Use internal JRE

<InstallDir>/tp/jdk/17 ...

Advanced Options: ...

Messages & Diagnostics

Message Level: All Log Level: Debug Diagnostics: □

Save To: C:/Users/tfchtner/Documents/Ansoft/ElectronicsDesktop2023.1/ ...

Ansys