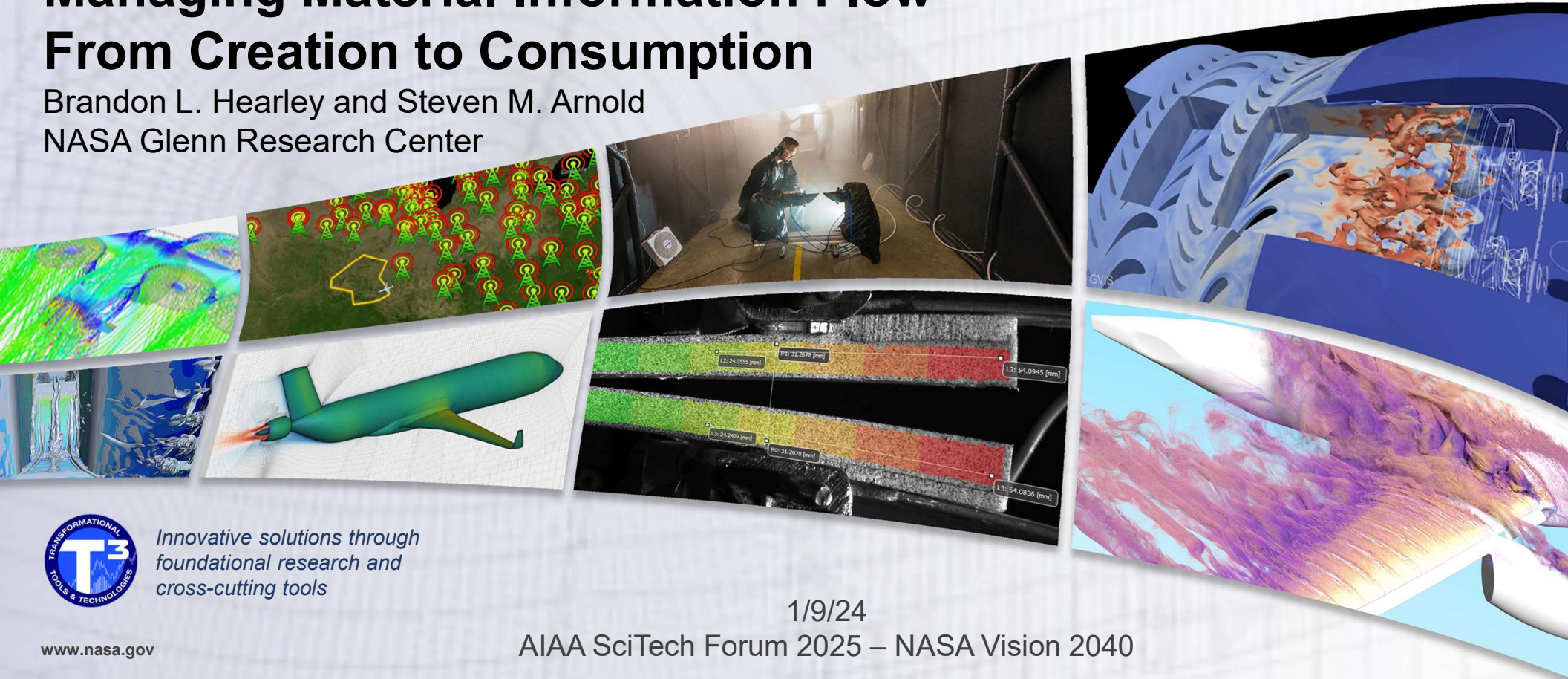


Managing Material Information Flow From Creation to Consumption

Brandon L. Hearley and Steven M. Arnold
NASA Glenn Research Center



*Innovative solutions through
foundational research and
cross-cutting tools*

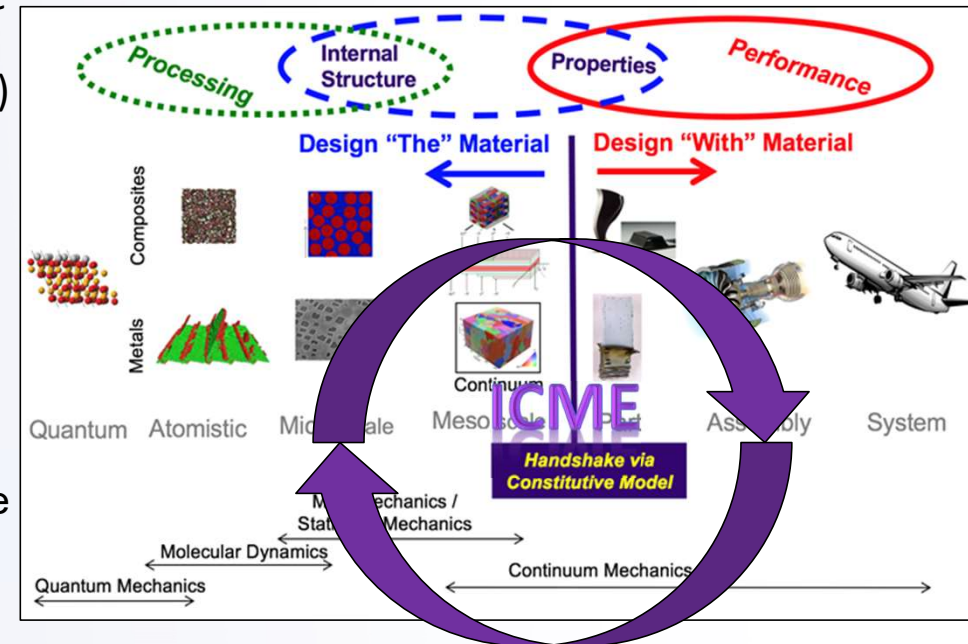
www.nasa.gov

1/9/24

AIAA SciTech Forum 2025 – NASA Vision 2040

Integrated Computational Materials Engineering (ICME) Enables Innovation

- Top performing organizations rate **New Materials** as one of **THE MOST IMPORTANT** factors in meeting their innovation goals (Historically new materials ≥ 20 years)
- Integrated Computation Materials Engineering (ICME) looks to bridge the gap between the “**Design-the-Material**” (Material Science) and “**Design-with-the-Material**” (Structural) viewpoints
 - Enables design of ‘fit-for-purpose’ materials
- Requirements for ICME
 - Experimentally validated materials models at multiple length scales
 - Understanding processing-structure-properties-performance relationships
 - Integrated framework that can automatically pass information across scales during design optimization
 - Manufacturing capability to achieve desired microstructure at any location in an application



Requires a robust information management system integrated with simulation toolsets and data pipelines

Vision 2040: A Roadmap for Integrated, Multiscale Modeling and Simulation of Materials and Systems



- In 2018, NASA Released the Vision 2040 report, outlining **technical** and **cultural** gaps preventing ICME design and suggested actions to close those gaps



Nine Identified Key Element Discipline Areas

- | | |
|--|---|
| 1. Models and Methodologies | 6. Data, Informatics, & Visualization |
| 2. Multiscale Measurement & Characterization Tools and Methods | 7. Workflows & Collaboration Frameworks |
| 3. Optimization & Optimization Methodologies | 8. Education & Training |
| 4. Decision Making and UQ | 9. Computational Infrastructure |
| 5. Verification & Validation | |

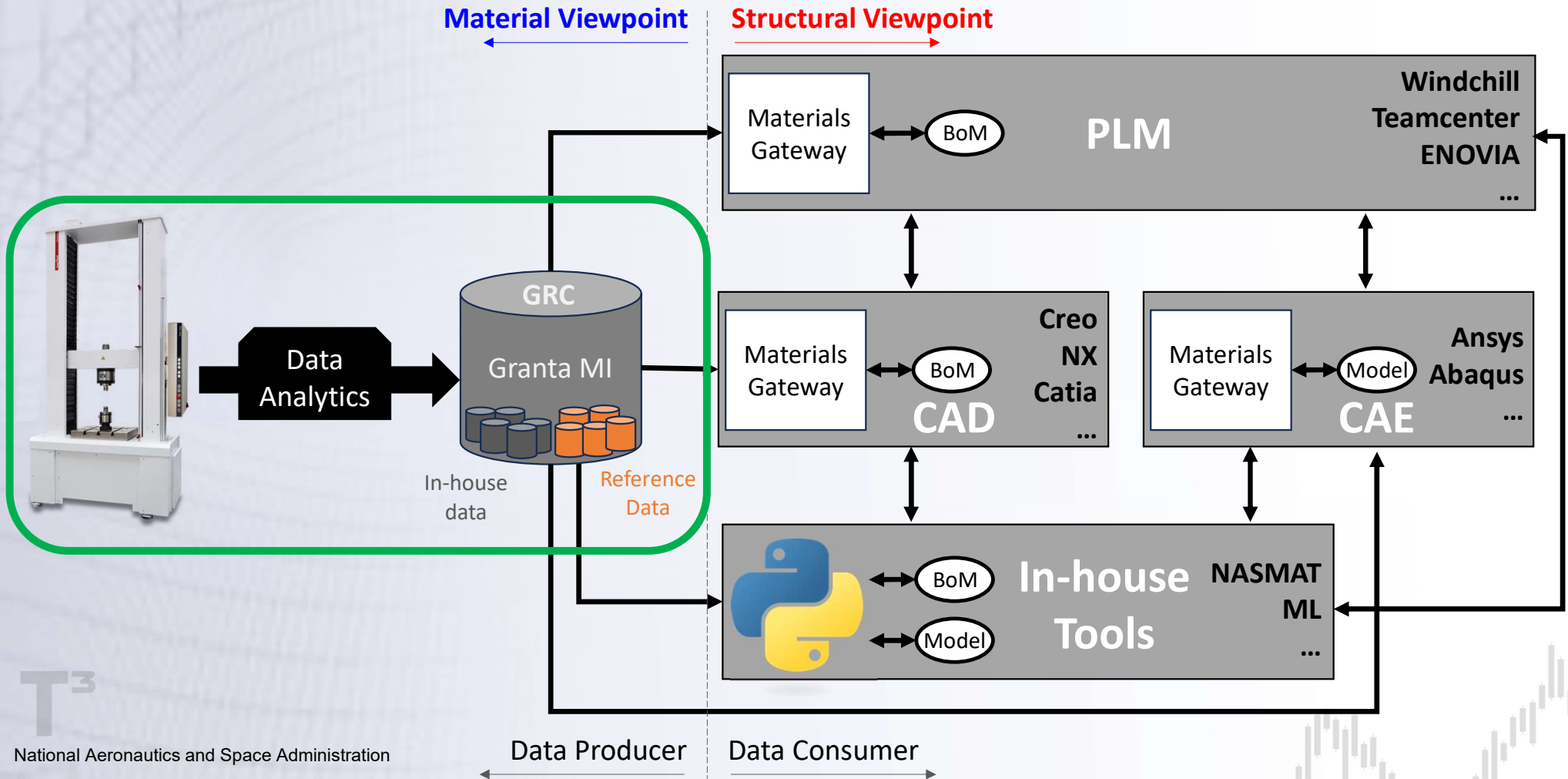
2040 Ecosystem **Revolutionizes** Design Paradigm

Today' s Design Paradigm	2040 Design Paradigm
Design Of Materials And Systems Is Disconnected	Design Of Materials And Systems Is Integrated
Stages Of The Product Development Lifecycle Are Segmented	Stages Of The Product Development Lifecycle Are Seamlessly Joined
Tools, Ontologies, And Methodologies Are Domain-specific	Tools, Ontologies, And Methodologies Are Usable Across The Community
Materials Properties Are Based On Empiricism	Materials Properties Are Virtually Determined
Product Certification Relies Heavily On Physical Testing .	Product Certification Relies Heavily On Simulation

*The cyber-physical-social ecosystem that marries “the design of materials” (material scientist viewpoint) with “the design with materials” (structural analyst viewpoint) approaches into one concurrent **transformational digital paradigm**.*



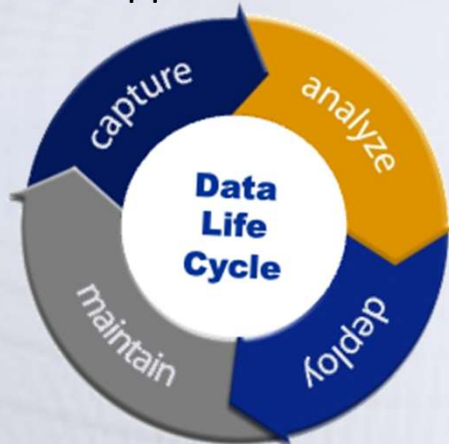
Materials Information Management at NASA GRC



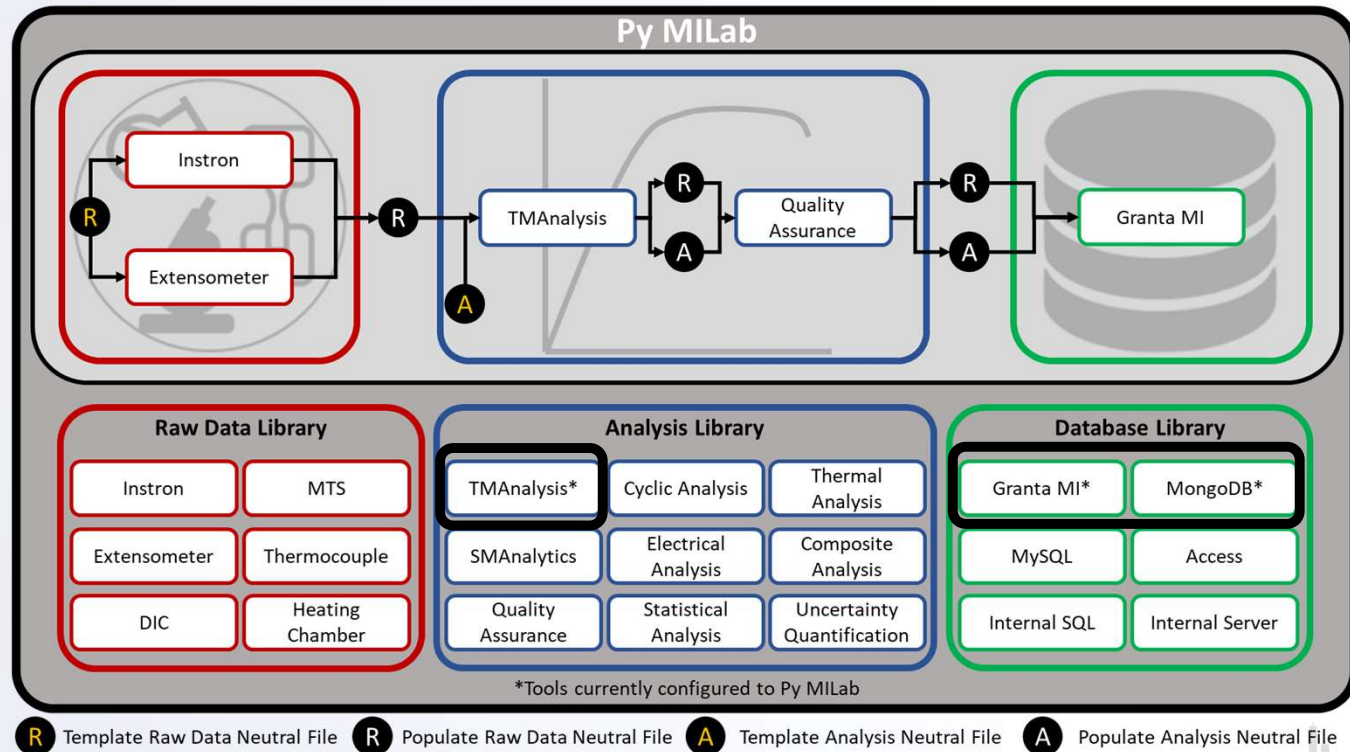
Py MILab: Automating Test Data Management Throughout the Data Life Cycle



- Py MILab look to automate test data management through data production, analysis, and storage
- Workflows established through modular approach



- Modules are interfaced via neutral file formats



Neutral File Formats: Enabling Tool Modularity and Flexibility

- Py MILab Operates on 2 Neutral Files: Raw Data File and Analysis File

- JSON Files → easily readable by most programming languages

- Each neutral file has a unique set of categories and attributes

- Each attribute contains:

- Data Type (point, string, array, image, etc.)
- Data Description
- Placeholder for value (when populated)
- Units (if applicable)
- Linking Behavior to other data (if applicable)

- New Categories can be added to the neutral files as new capabilities are added to the framework

- Cyclic Analysis, Fatigue Analysis, Shape Memory Alloys, etc.

```
{
  "General Information": {
    "Type": "point",
    "UnitsLink": "[ 'General Information' ][ 'Units' ][ 'Stress' ]",
    "Description": "Shear elastic modulus in the 12-direction",
    "Value": null
  },
  "Tensile Analysis": {
    "Type": "point",
    "UnitsLink": "[ 'General Information' ][ 'Units' ][ 'Stress' ]",
    "Description": "Shear elastic modulus in the 13-direction",
    "Value": null
  },
  "Compressive Analysis": {
    "Type": "point",
    "UnitsLink": "[ 'General Information' ][ 'Units' ][ 'Stress' ]",
    "Description": "Shear elastic modulus in the 23-direction",
    "Value": null
  },
  "Shear Analysis": {
    "Shear Modulus-12": {
      "Type": "point",
      "UnitsLink": "[ 'General Information' ][ 'Units' ][ 'Stress' ]",
      "Description": "Shear elastic modulus in the 12-direction",
      "Value": null
    },
    "Shear Modulus-13": {
      "Type": "point",
      "UnitsLink": "[ 'General Information' ][ 'Units' ][ 'Stress' ]",
      "Description": "Shear elastic modulus in the 13-direction",
      "Value": null
    },
    "Shear Modulus-23": {
      "Type": "point",
      "UnitsLink": "[ 'General Information' ][ 'Units' ][ 'Stress' ]",
      "Description": "Shear elastic modulus in the 23-direction",
      "Value": null
    },
    "Shear Proportional Limit-12": {
      "Type": "point",
      "UnitsLink": "[ 'General Information' ][ 'Units' ][ 'Stress' ]",
      "Description": "Shear proportional limit in the 12-direction",
      "Value": null
    }
  }
}
```

← *Category*

← *Attribute*



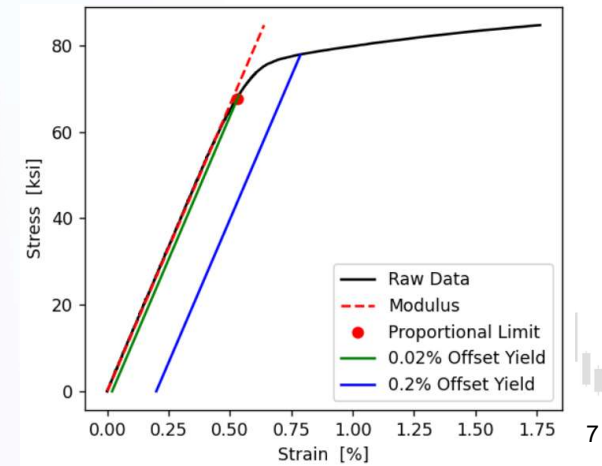
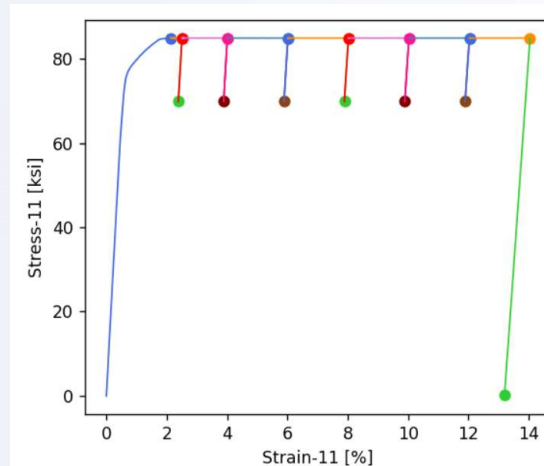
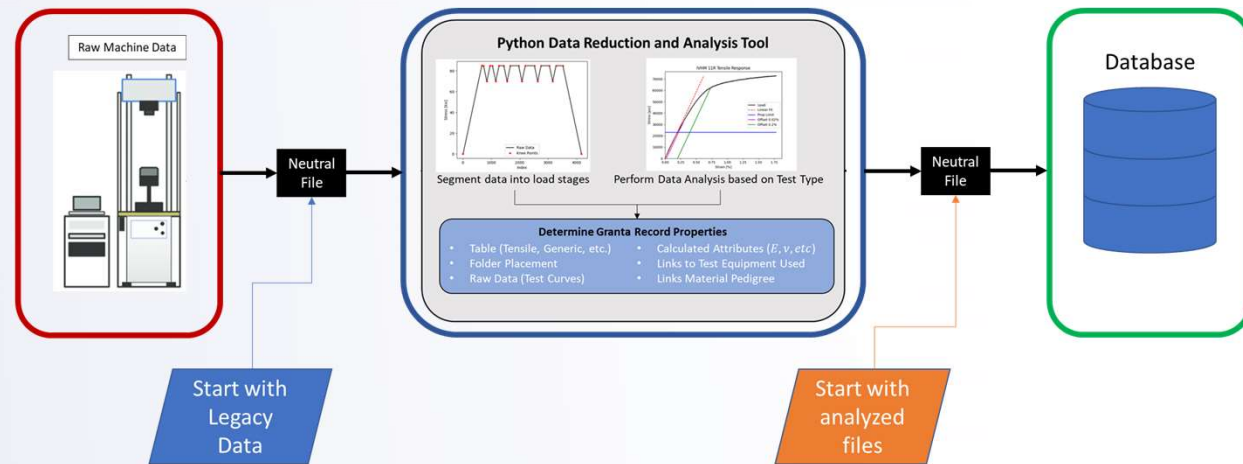
TMAAnalysis: Analyzing and Storing Test Data



- TMAAnalysis performs automatic segmentation, reduction, and analysis of thermomechanical test data
- Benefits:
 1. Reduce user effort for data management and increases throughput
 2. Consistency in data analysis and reduction
 3. Establishes traceability between raw test data and analysis for model development
- Able to perform automatic segmentation for complex load histories
 - User has ability to edit within the GUI
- Individual analysis subroutines for each load type (tension, compression, creep, relaxation)
 - Applied to each stage

NASA TM 20240012389

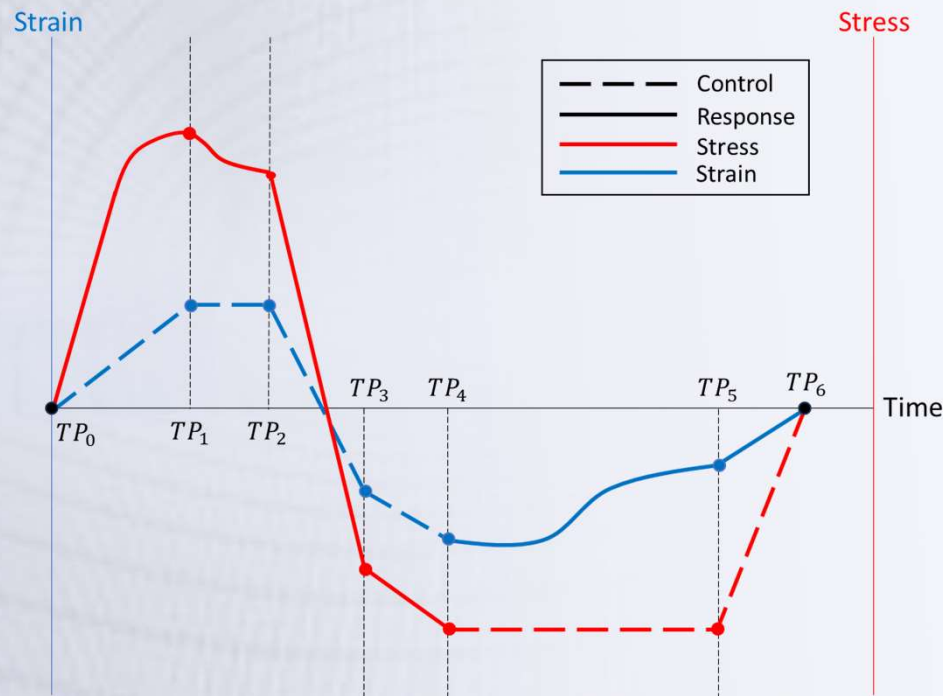
National Aeronautics and Space Administration



Generic Thermomechanical Test: Segmentation & Analysis



General Load History: Any thermomechanical test can be decomposed into discrete linear segments of known stress rate, strain rate, or temperature rate with defined turn points



N segments (Between TP_{i-1} and TP_i); Load = Temp or Mech

N=1 tensile, compressive, shear, etc.

N=2 load/unload, creep, relaxation, etc.

Load Segment Type	Description	Rule
Tensile Loading	Axial loading of the specimen in one of the three normal directions	$\dot{\epsilon}_{ij} > 0$ and $\epsilon_{ij}^{end} > 0$ or $\dot{\sigma}_{ij} > 0$ and $\sigma_{ij}^{end} > 0$ where $i = j$
Compressive Loading	Axial compressive loading of the specimen in one of the three normal directions	$\dot{\epsilon}_{ij} < 0$ and $\epsilon_{ij}^{end} < 0$ or $\dot{\sigma}_{ij} < 0$ and $\sigma_{ij}^{end} < 0$ where $i = j$
Shear Loading	Shear loading of the specimen in one of the three shear directions	$\dot{\epsilon}_{ij} > 0$ and $\epsilon_{ij}^{end} > 0$ or $\dot{\sigma}_{ij} > 0$ and $\sigma_{ij}^{end} > 0$ where $i \neq j$
Tensile Unloading	Axial unloading of the specimen in one of the three normal directions	$\dot{\epsilon}_{ij} < 0$ and $\epsilon_{ij}^{end} > 0$ or $\dot{\sigma}_{ij} < 0$ and $\sigma_{ij}^{end} > 0$ where $i = j$
Compressive Unloading	Axial compressive unloading of the specimen in one of the three normal directions	$\dot{\epsilon}_{ij} > 0$ and $\epsilon_{ij}^{end} < 0$ or $\dot{\sigma}_{ij} > 0$ and $\sigma_{ij}^{end} < 0$ where $i = j$
Shear Unloading	Shear unloading of the specimen in one of the three shear directions	$\dot{\epsilon}_{ij} < 0$ and $\epsilon_{ij}^{end} > 0$ or $\dot{\sigma}_{ij} < 0$ and $\sigma_{ij}^{end} > 0$ where $i \neq j$
Relaxation	Constant strain over a set period of time	$ \dot{\epsilon}_{ij} = 0$
Creep	Constant stress over a set period of time	$ \dot{\sigma}_{ij} = 0$

T³

TMAAnalysis: Segmentation



- Three segmentation algorithms available in TMAAnalysis 1.0

1. Raw Segmentation

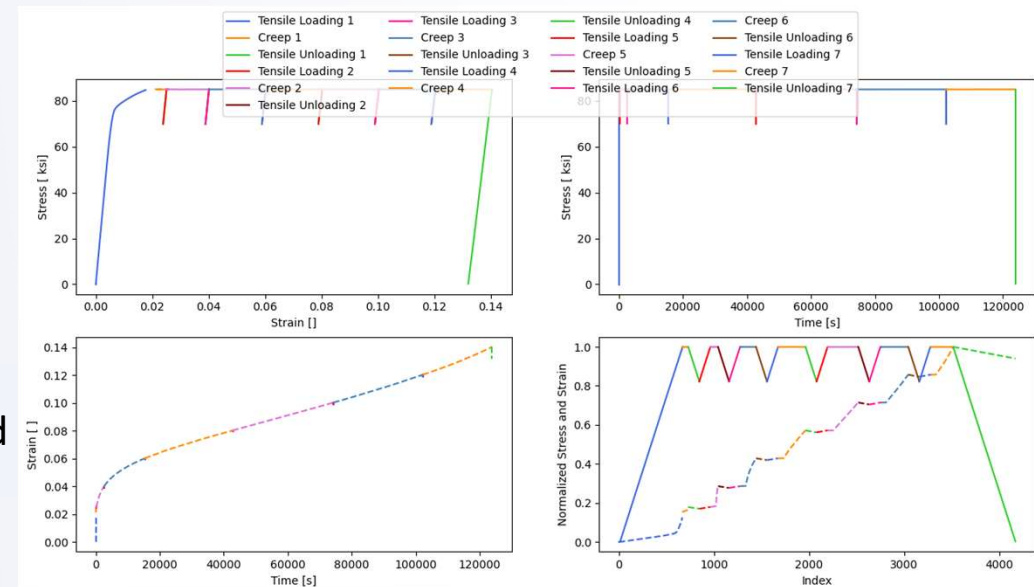
- Uses linear analysis of stress and strain vs time coupled with a knee-point algorithm to identify turn points
- Typically used for **legacy data**

2. Assisted Segmentation

- Uses target values from the test machine (e.g., target strain, target stress, stage duration) coupled with the knee-point algorithm to find turn points
- Typically used for **new test data**

3. Defined Segmentation

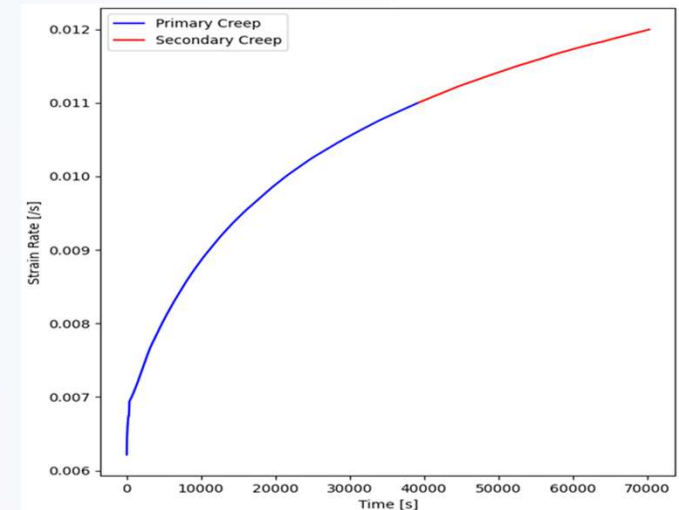
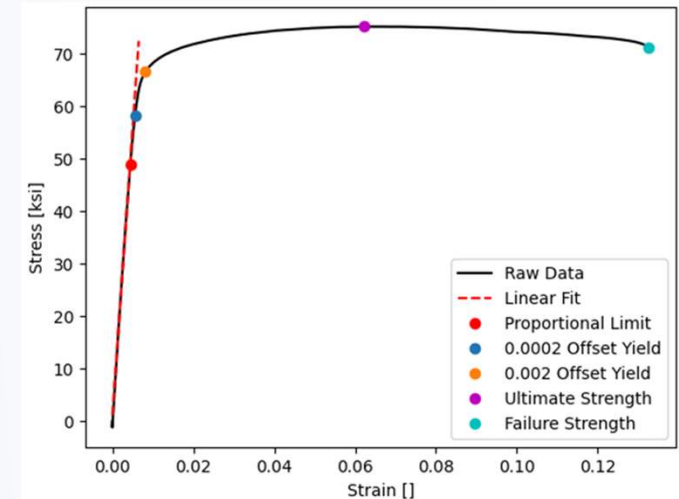
- Allows the user to specify the exact end point/data point index of the end of each stage manually
- Typically used while **editing stages in the GUI**



TMA Analysis: Stage Analysis

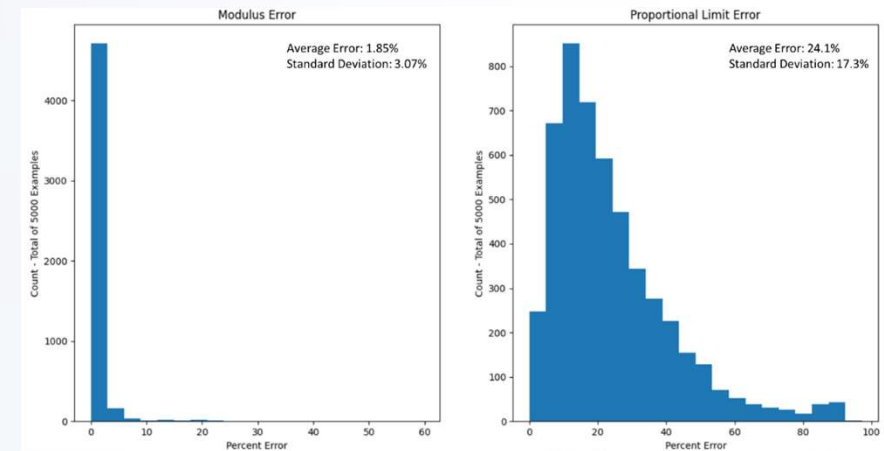
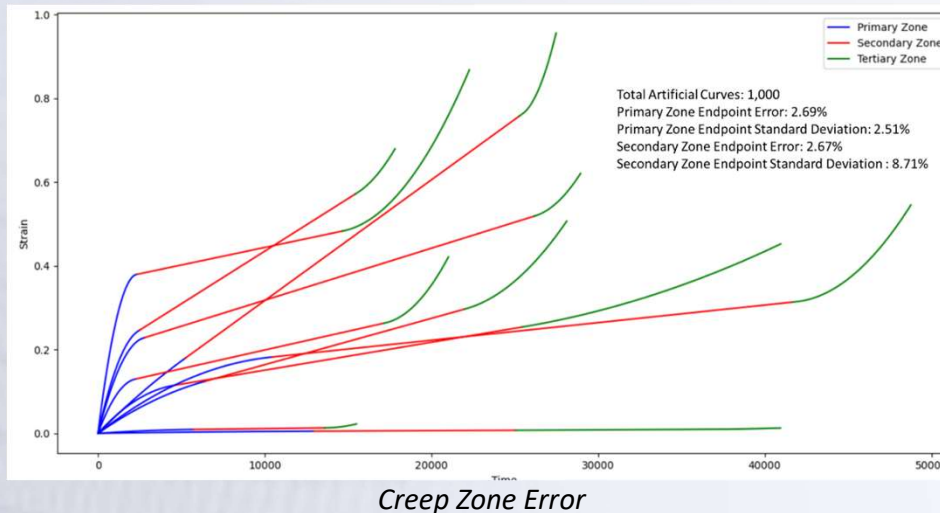
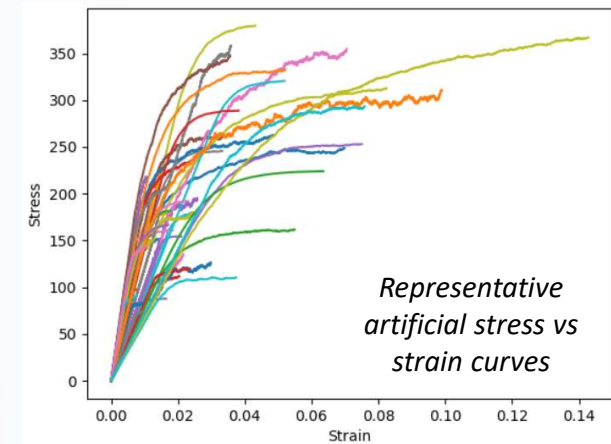


- Target time, strain, and stress and strain/stress rates calculated for each stage
 - Loading (Tension, Compression, and Shear)
 - Calculates modulus, proportionall limit, Poisson's Ratio (except for shear), yield, UTS, and failure (if applicable)
 - Unloading (Tension, Compression, and Shear)
 - Calculates unloading modulus, reversible strain, and irreversible strain
 - Relaxation
 - Calculates constant strain, hold time, and stress change
 - Creep
 - Calculates constant stress and hold time
 - Automatic identification of primary, secondary, and tertiary zones



Analysis Module Error

- Automatic algorithms for proportional limit, modulus, and creep zones error studied by generating artificial raw data with noise
 - E and $\sigma_P \rightarrow$ stress vs strain curves
 - Low error for E , higher error for σ_P
 - Creep Zones $\rightarrow$ strain vs time curves
 - Low error in prediction of primary, secondary, and tertiary zones



Modulus and Proportional Limit Error

TMAalysis: Executing the Code

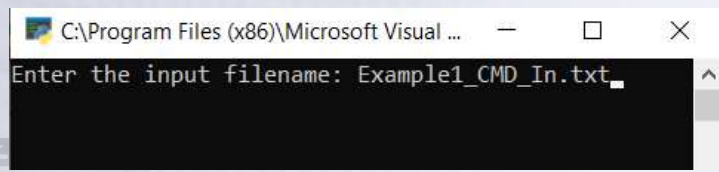
- TMAalysis has two options for executing automatic segmentation and analysis
 1. TMAalysis CMD – command line executable for rapid, batch analysis
 2. TMAalysis GUI – Graphical User Interface for viewing and editing analysis

CMD

```

TEST                                # Keyword to indicate new test
Raw_Tensile_Failure.json            # Name of Raw Data Neutral File
Fail = 1                           # Tested to Failure (0 = No, 1 = Yes)
Valid = 1                          # Valid Test (0 = No, 1 = Yes)
YieldOff = 0.0002, 0.002           # Yield offset strain values
Plot = 1                           # Plotting (0 = No, 1 = Yes)

TEST                                # Keyword to indicate new test
Raw_Tensile_Load_Unload.json        # Name of Raw Data Neutral File
Fail = 0                           # Tested to Failure (0 = No, 1 = Yes)
Valid = 1                          # Valid Test (0 = No, 1 = Yes)
YieldOff = 0.002                   # Yield offset strain values
Plot = 0                           # Plotting (0 = No, 1 = Yes)
  
```



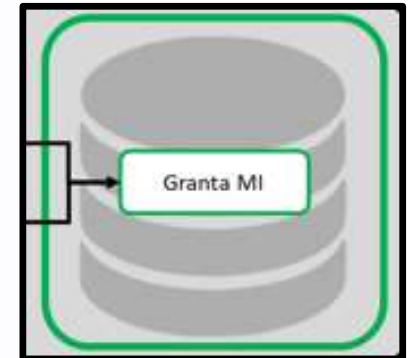
GUI

Stage Table Test Properties Reanalyze Stages

Stage Name	Stage Type	Control Mode	Target Strain []	Target Stress [ksi]	End Time [s]	End Index
1 Stage_0	Tensile Loading	✓ Strain	✓ 0.2	105.85	19960.02	4353
2 Stage_1	Tensile Unloading	✓ Strain	✓ 0.19	-0.04	21182.62	5576
3 Stage_2	Creep	✓ Stress	✓ 0.19	-0.04	22980.62	7374

Time vs. Stress-11 Plot

Py MILab Database: Overview of Capabilities



- Py MILab 1.0 Capabilities:
 - Configuring New Databases
 - Allow users to configure their in-house schema to the Py MILab Neutral Files attributes
 - Currently support **Granta MI** and **MongoDB**
 - Each platform has different data types supported, require unique sub-modules written
 - Viewing/Editing Configurations via Schema Translator
 - Users store local library of configurations → have option to view and edit as necessary
 - Allows periodic saving and adapting schemas
 - Importing Data
 - Users can import Py MILab Neutral files and select a defined database configuration for automatic or “semi-automatic” data importing
 - Automatic – connect directly to the database via API to import data
 - Semi-Automatic – write an import file unique to that database platform (offline)





Schema Translator: Selecting the User's Schema

- Granta MI users may want to configure Py MILab to *multiple* tables in their database
 - Separate schema for Tensile, Creep, Relaxation, etc.*

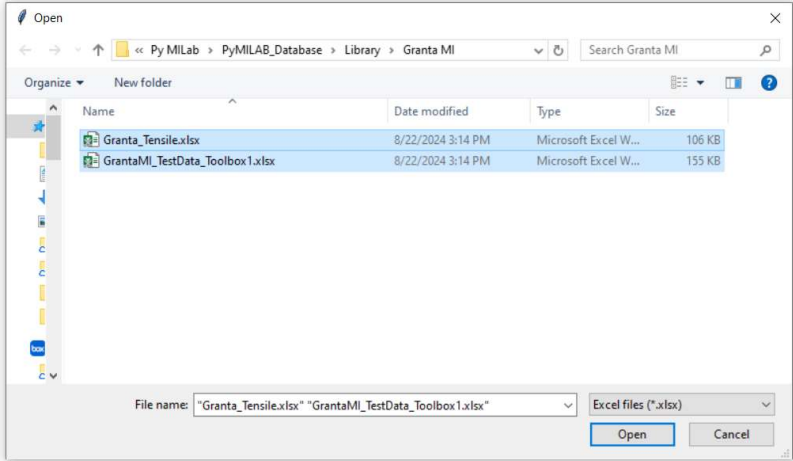
Py MILab: Database

Upload the excel files for the tables in Granta MI you want to configure to (see documentation for help):

Select Files

Back

Continue



Users can select Excel schema files for multiple tables

Schema files are generated using MI Toolbox (documentation has detailed instructions)



Schema Translator: Configuring Attributes

- Granta MI supports Single Value, Functional, and Tabular Attributes

Py MILab: Database

Test Data: Thermomechanical

Single Value

Functional Value

Tabular Value

Record Placement

Granta MI Column

Stage Name

Stage Type

Control Mode

End Time (s)

Target Strain (11 axis) (%)

Target Strain (22 axis) (%)

Target Strain (33 axis) (%)

Target Strain (12 axis) (%)

Py MILab #1

Stages - Stage Name

Stages - Stage Type

Stages - Control Mode

target

Control Information - Target Time

Control Information - Target Displacement-11

Control Information - Target Displacement-22

Control Information - Target Displacement-33

Control Information - Target Deformation-12

Control Information - Target Deformation-13

Control Information - Target Deformation-23

Control Information - Target Load-11

Control Information - Target Load-22

Control Information - Target Load-33

Test Summary

Operator(s)

Standards

Metals Material/Processing Information

Ceramics Material/Processing Information

Polymers Material/Processing Information

Reinforcements Material/Processing Information

Testing Frame

Temperature Application Equipment

Strain Measurement Device(s)

Load Measurement Device(s)

Temperature Measurement Device(s)

Yield Points (11-axis)

Yield Points (22-axis)

Yield Points (33-axis)

Compressive Yield Points (11-axis)

Compressive Yield Points (22-axis)

Compressive Yield Points (33-axis)

Shear Yield Points (12-axis)

Shear Yield Points (13-axis)

Shear Yield Points (23-axis)

Test Summary

Drop down to select tabular attribute

Configure each tabular column with a Py MILab Attribute

Home

1/5

Next

Save

Schema Translator: Setting Record Placement Rules



- For each table (if multiple are selected), records can be written to an arbitrary number of folders

Py MILab: Database

Test Data: Thermomechanical

Single Value Functional Value Tabular Value Record Placement

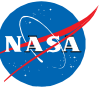
Number of Folder Levels: 2 Level 2 Number of Conditions: 5

Conditional Attribute	Conditional Value	Naming Attribute	Format
Test Information - Test Type	Tensile	Tensile Analysis - Strain Rate	[attribute]
Test Information - Test Type	Tensile Analysis - Modulus-11	Compressive Analysis - Compressive Strain Rate	[attribute] /s
Test Information - Test Type	Shear	Shear Analysis - Shear Strain Rate	[attribute] /s
Test Information - Test Type	Creep	Creep Analysis - Creep Hold Stress-11	[attribute] ksi Hold Stress
Test Information - Test Type	Relaxation	Creep Analysis - Creep Hold Stress-22	[attribute] Hold Strain
Test Information - Test Type		Creep Analysis - Creep Hold Stress-33	
Test Information - Test Type		Relaxation Analysis - Relaxation Hold Strain-11	
Test Information - Test Type		Relaxation Analysis - Relaxation Hold Strain-22	
Test Information - Test Type		Relaxation Analysis - Relaxation Hold Strain-33	

Home

[attribute] replaced by data value
(i.e., [attribute] /s → 1E-3 /s)





Importing Data

- Users can select a configured database and the Raw and Analysis (optional) Neutral Files and connect with Python API to write records

Py MILab: Database

Select the database platform:

MongoDB
Granta MI
MongoDB

Select the database:

MongoTensile

Select the Raw Data Neutral File

Raw_Tensile_Failure.json

Browse

Select the Analysis Neutral File

Raw_Tensile_Failure_Analyzed.json

Browse

Back

Continue

Data automatically written to the user's configured database from the configured neutral files

Summary



- Automating workflows for data creation, analysis, maintenance, and dissemination is critical for ICME and the NASA Vision 2040
 - Large data sets will require high throughput and consistent methods for analysis
- Py MILab looks to automate the process of capturing raw data from the test machine, analyzing the data, and storing the data in an information management system
 - Modular approaches enable more capabilities within the same framework
- TMAAnalysis performs automatic segmentation and stage-by-stage analysis of isothermal mechanical tests
 - GUI enables users to view results and edit automatic analyses if necessary
 - CMD executable enables high throughput for large number of tests
- Py MILab: Database allows users to configure their database platform and schema to the Py MILab neural files
 - Enables any user to use the Py MILab framework within their organization with the Schema Translator
 - Automates data storage process



Thank You for Your Attention



Integrate Don't Duplicate

Brandon Hearley, PhD
Digital Transformation Specialist
NASA Glenn Research Center
Multiscale and Multiphysics Modeling Branch
brandon.l.hearley@nasa.gov

Steven M. Arnold, PhD, FASM
Technical Lead: Multiscale Multiphysics Modeling
T3 Project: Material and Structures Discipline Lead
Materials and Structures Division
steven.m.arnold@nasa.gov



National Aeronautics and Space Administration

