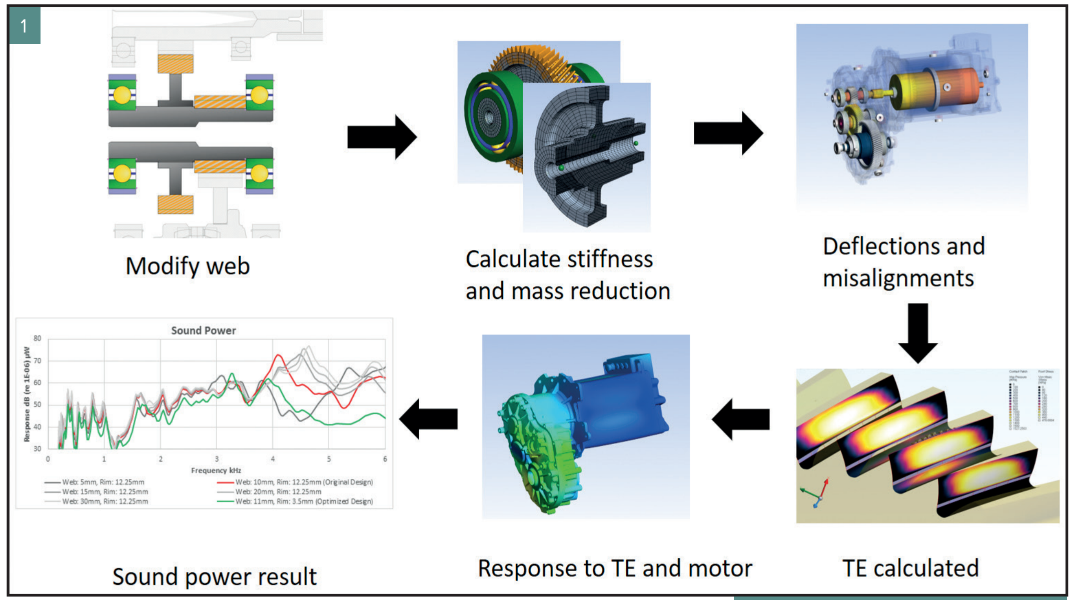
Automation via scripting

The next generation of transmission-specific software is enabling engineers to save time by automating processes such as model building and extracting results



1. Automated optimization methodology to investigate the effect of gear blank web stiffness on system dynamics

The use of software systems in transmission development, manufacture and troubleshooting processes is nowadays considered essential. With the ever-stringent requirements and decreasing lead times required to get ahead in a highly competitive market, the efficient use of software tools throughout development is critical. Software should enable engineers to optimize the use of their time in making important engineering decisions, and allow them to spend less time on manual tasks such as concept/model building, running analyses and extracting results. Further, the use of multiple tools in current design and development workflows often means duplication of effort in inputting models, which also increases the potential for error. Advanced software systems should enable data to be passed easily between tools, whether they are separate commercial packages or in-house methods.

Next-generation transmission-specific software will, via scripting, allow engineers to make efficient use of time and company IP within workflows without needing to share it externally.

Transmission design and analysis software such as MASTA have now become critical to many engineers. A full system approach is essential to capture component interactions in complex transmission systems. These tools can be used from concept design, through detailed design, to manufacture, as well as to troubleshoot existing transmission problems. Comprehensive analyses cover a broad range of fidelity from the implementation of international standards to higher fidelity system and component level analyses. Considerations including weight, durability, efficiency and NVH can be considered.

Such tools enable virtual testing iterations where the effect of design changes on targets can be analyzed with ease and 'what if' scenarios investigated quickly and efficiently. Such tools can be controlled in an outer loop that includes studies in DOE and optimization, allowing efficient investigation into the effects of manufacturing and assembly variability. Figure 1 shows this process, depicting an automated study of the NVH behavior of an EV gearbox while tuning a gear blank to minimize the dynamic response of the housing.

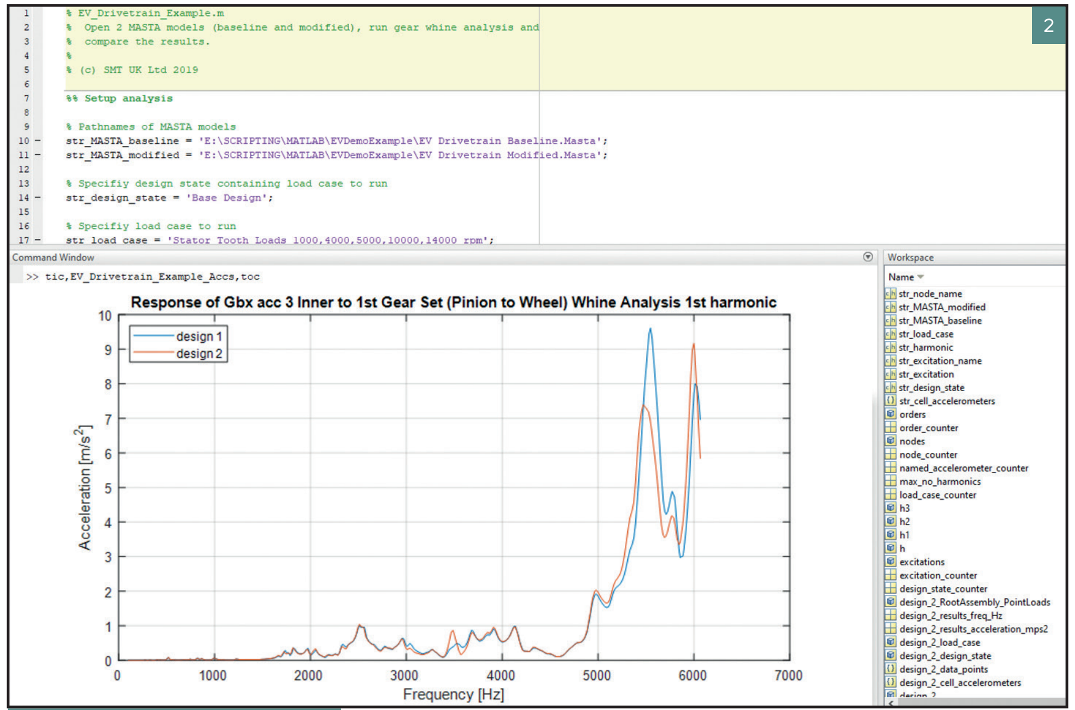
Having such software embedded into a company's design and development process can enable notable reductions in development time while producing more robust transmissions. Hardware testing iterations are also minimized.

There is, however, often a large amount of work still performed by the engineer, which could benefit from being automated. Further, companies can benefit from more tightly integrating such tools with their own IP, analysis methods, workflows, in-house tools and knowledge database. This is where scripting comes in.

In MASTA 10, SMT has introduced extensive scripting functionality. Scripting developed out of two main ideas. First, enabling engineers to spend less time on manual tasks of model building, running analyses and extracting results. And second, enabling companies to use their own IP efficiently in design and development workflows without having to share it externally. MASTA 10 scripting, however, enables much more than this.

The API allows users to write their own code outside MASTA to interact with MASTA models and analyses. The API can be used from a range of languages such as Python and MATLAB, as well as any .NET language. This flexibility allows users to write in the languages they are familiar with. It also enables code writing using common integrated development environments, including Microsoft's Visual Studio, and use debugging, IntelliSense and other development tools. Further, it allows easy interaction with in-house tools no matter what language they are written in. The use cases for such functionality are limitless and range from small utility functions to whole new analysis methods.

One such use case is a customer that has its own in-house code for synthesizing planetary system designs. Such code takes as input the boundaries under which the system is to operate, and outputs a number of suggested design topologies. The next step is to build MASTA models of those options and analyze them in more detail. Scripting allows the customer to write code to automatically create these MASTA models from the outputs. This eliminates manual work and the potential for error in building the models.

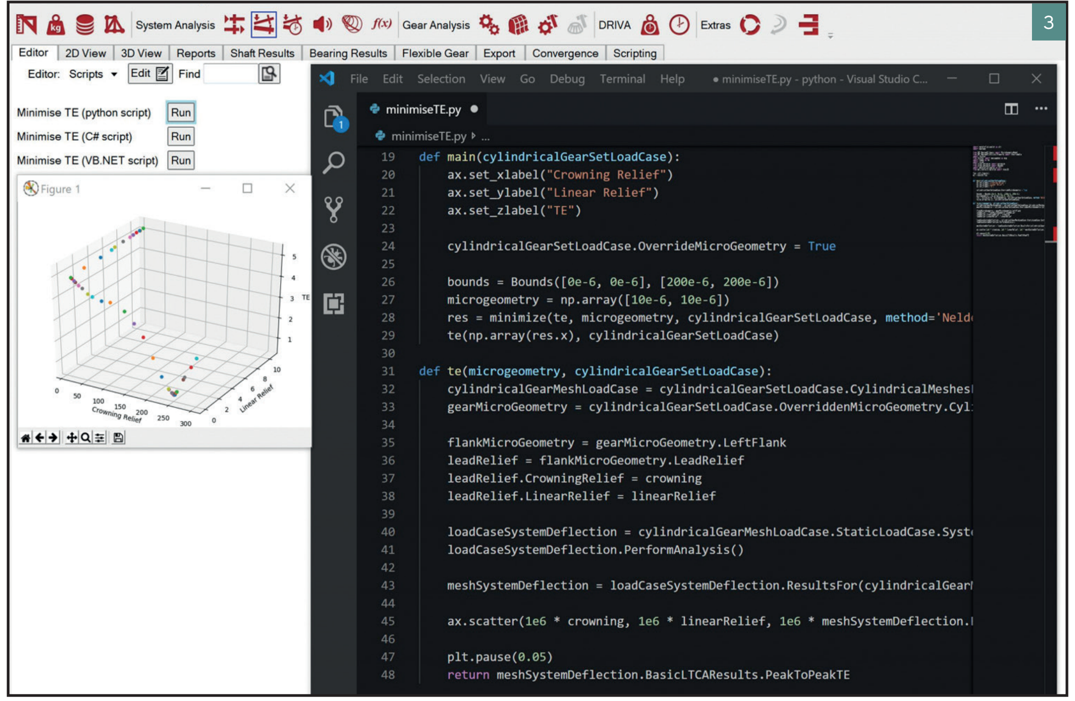


```
% EV_Drivetrain_Example.m
%  Open 2 MASTA models (baseline and modified), run gear whine analysis and
%  compare the results.
%
% (c) SMT UK Ltd 2019

%% Setup analysis

% Pathnames of MASTA models
str_MASTA_baseline = 'E:\SCRIPTING\MATLAB\EVDemoExample\EV Drivetrain Baseline.Masta';
str_MASTA_modified = 'E:\SCRIPTING\MATLAB\EVDemoExample\EV Drivetrain Modified.Masta';

% Specifiy design state containing load case to run
str_design_state = 'Base Design';

% Specifiy load case to run
str load case = 'Stator Tooth Loads 1000,4000,5000,10000,14000 rpm';
```

```
>> tic,EV_Drivetrain_Example_Accs,toc
```

2. MATLAB script using MASTA API to compare two different MASTA models

3. Script using high-level Python functions, run from SMT's MASTA, to optimize micro geometry for minimum transmission error



```
def main(cylindricalGearSetLoadCase):
    ax.set_xlabel("Crowning Relief")
    ax.set_ylabel("Linear Relief")
    ax.set_zlabel("TE")

    cylindricalGearSetLoadCase.OverrideMicroGeometry = True

    bounds = Bounds([0e-6, 0e-6], [200e-6, 200e-6])
    microgeometry = np.array([10e-6, 10e-6])
    res = minimize(te, microgeometry, cylindricalGearSetLoadCase, method='Neld
    te(np.array(res.x), cylindricalGearSetLoadCase)

def te(microgeometry, cylindricalGearSetLoadCase):
    cylindricalGearMeshLoadCase = cylindricalGearSetLoadCase.CylindricalMeshes
    gearMicroGeometry = cylindricalGearSetLoadCase.OverriddenMicroGeometry.Cyl

    flankMicroGeometry = gearMicroGeometry.LeftFlank
    leadRelief = flankMicroGeometry.LeadRelief
    leadRelief.CrowningRelief = crowning
    leadRelief.LinearRelief = linearRelief

    loadCaseSystemDeflection = cylindricalGearMeshLoadCase.StaticLoadCase.Syst
    loadCaseSystemDeflection.PerformAnalysis()

    meshSystemDeflection = loadCaseSystemDeflection.ResultsFor(cylindricalGear

    ax.scatter(1e6 * crowning, 1e6 * linearRelief, 1e6 * meshSystemDeflection.

    plt.pause(0.05)
    return meshSystemDeflection.BasicLTCAResults.PeakToPeakTE
```

A second use case is shown in Figure 2. MATLAB code was written using the MASTA API to run two variants of MASTA models and compare dynamic response to gear whine. In this case, the acceleration response at an accelerometer location for the primary gear mesh order is compared. By means of such a utility, comparison between models can be conducted quickly and efficiently.

Scripting doesn't only allow the user to interact with software from outside the system interface. It also gives greater possibility to extend the software functionality in the user interface. Figure 3 shows a Python script referenced by a button in SMT's MASTA software. When this button is pressed, the script optimizes the gear micro geometry to minimize transmission error. This simple example demonstrates the advantage of being able to use high-level Python functions and libraries such as Matplotlib and SciPy in MASTA scripts.

Scripts run from within the software can either be obtained from referenced source code libraries or from compiled dynamic link libraries (DLLs). For easy distribution of scripting tools, the SMT Store allows easy download from within the software of any apps that have been distributed by SMT.

Greater automation is needed for many tasks within transmission design and development processes using software. Software does not replace the expert engineer, but it can minimize time spent on manual tasks and maximize the time spent making more important design decisions. Scripting and the MASTA API can enable this.

Scripting further enables a company to continually improve its design and analysis processes by efficiently integrating their own technical advances as they are made and without being required to share their own IP. ◎

FREE READER INQUIRY SERVICE
To learn more about Smart Manufacturing Technology, visit:
www.ukimediaevents.com/info/tr
INQUIRY NO. 511