文章编号:1671-4598(2025)01-0155-08 DOI:10.16526/j.cnki.11-4762/tp.2025.01.020 中图分类号:TP391.41 文献标识码:A

基于动态加载技术的试验数据分析系统研究

陆俊杰[1]，李洪普[1]，李晓峰[2]，李 锋[2]
(1. 中船奥蓝托无锡软件技术有限公司，江苏 无锡 214000；
2. 中国船舶科学研究中心，江苏 无锡 214000)

摘要：传统的试验数据分析系统通常内嵌固定数量的算法模块进行试验数据处理，系统存在扩展性差、灵活度低等问题，在面对算法更新、算法扩展的需求时，迭代开发又增加了时间与人力成本；因此提出一种基于动态加载技术的试验数据分析系统，主要基于 Python 语言进行开发，采用 PyQt5、MongoDB 数据库、importlib 库、WebService、MATLAB Engine API 等技术搭建，具备算法信息同步、算法存储、算法展示、模块动态加载以及远程通信接口等功能；采用 importlib 库和 MATLAB Engine API 两种关键技术实现多类算法文件的动态加载，采用 WebService 技术实现不同系统间的远程数据通信。经实际应用满足了系统运行期间能够远程实时获取算法文件，动态加载并调用其算法模块，最终正确展示数据分析结果，验证了设计方案的可行性，极大地提升了试验数据分析系统的可扩展性与灵活度。

关键词：试验数据分析系统；PyQt5；模块动态加载；importlib 库；远程数据通信

Research on Experimental Data Analysis System Based on Dynamic Loading Technology

LU Junjie[1], LI Hongpu[1], LI Xiaofeng[2], LI Feng[2]
(1. Wuxi Orient Software Technology Co., Ltd., Wuxi 214000, China;
2. China Ship Scientific Research Center, Wuxi 214000, China)

Abstract: Traditional experimental data analysis systems usually adopt the modules embedded a fixed number of algorithm to process experimental data, which has the problems of poor scalability and low flexibility. According to the requirements of algorithm update and expansion, iterative development increases time and labor costs. Therefore, a test data analysis system based on dynamic loading technology is proposed, which is mainly developed based on Python language and built by using PyQt5, MongoDB database, importlib library, WebService, MATLAB Engine API and other technologies. It has the functions of the algorithm information synchronization, algorithm storage, algorithm display, module dynamic loading, and remote communication interface. Two key technologies of importlib and MATLAB Engine API are used to implement the dynamic loading of multi-class algorithm files, and the WebService technology is used to achieve remote data communication between different systems. Through practical application, the system meets the remote and real-time acquisition of algorithm files during operation, and the module of the algorithm can be loaded and invoked dynamically. Finally, the system displays the experimental data analysis result correctly, verifies the feasibility of the design scheme, and greatly improves its scalability and flexibility.

Keywords: experimental data analysis system; PyQt5; module dynamic loading; importlib; remote data communication

0 引言

软件技术的不断发展使得软件内部功能模块关系越来越复杂，许多用户不再满足于功能定制单一，而是愈发希望增加软件的可扩展性。软件可扩展性是指软件系统能够方便地适应变化、增长和需求的能力。它是衡量软件质量的一个重要属性，对于应对不断变化的需求和技术来说至关重要。

在试验数据分析领域，传统的试验数据分析系统根据需求封装了特定数量的分析算法，当面对新增算法或

收稿日期:2023-09-20； **修回日期**:2024-01-03。

作者简介:陆俊杰(1995-),男,硕士,工程师。

引用格式:陆俊杰,李洪普,李晓峰,等. 基于动态加载技术的试验数据分析系统研究[J]. 计算机测量与控制,2025,33(1):155-162.

者已有算法不断更新时只能依靠程序开发人员对试验数据分析系统进行二次开发、编译以及部署，导致时间和人力成本的增加。

因此针对上述问题，对基于动态加载技术的试验数据分析系统进行了研究。经过理论研究和实际应用，在系统运行状态下，采用动态加载技术实现系统实时加载并调用算法模块进行操作，旨在解决现有试验数据分析系统中分析功能扩展周期长、成本高、代价大的问题。

1 模块导入原理

在 python 语言中，命名空间（namespace）是一个存储名称与对象对应关系的字典，其中名称就是指标识符，对象就是指一切对象，例如整数、浮点数、字符串、函数、类等等。

命名空间可以划分以下三大类：

1）内置命名空间（Built-in Namespace），是 python 解释器启动时就加载的命名空间，开发人员可以直接使用；

2）全局命名空间（Global Namespace），全局命名空间是指包含在主程序层定义任何对象，自主程序启动时创建一直存次直到解释器停止；

3）局部命名空间（Loacal Namespace），是在函数调用时创建的命名空间，用于存储函数的局部变量、参数、临时变量等。

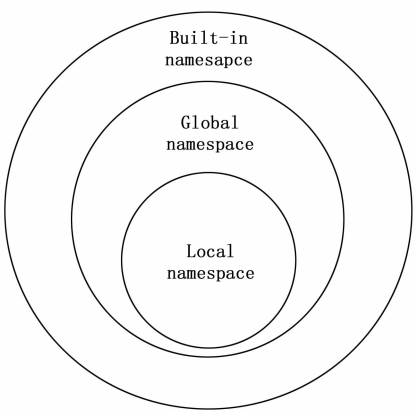

图 1 命名空间类型

为了区分不同命名空间中存在的相同对象名，引入了作用域（scope），指可以直接访问命名空间的文字区域。Python 作用域搜索遵循 LEGB 规则，即假如在程序中引入 x 解释器将依次在局部、全局和内置命名空间中搜索该对象。

模块化编程时强调将计算机程序的功能分离成独立的、可相互改变的“模块”的软件设计技术，它能够实现每个模块包含着执行预期功能的一个唯一方面所必需的所有东西。Python 使用函数（funtion）、模块（module）和包（packages）等概念来实现模块化编程，使得代码更具可维护性和可重用性。

模块文件类型主要有 *.py 文件、*.pyc、*.pyd、*.pyw、*.so、*.dll 等，也是 python 代码载体的最小单元，具体的定义种类如下所示：

1）模块可以用 python 本身编写；

2）模块可以用其他语言编写；

3）内置模块则内置在解释器中。

当使用关键字 import 导入模块或包时，python 通常提供两种导入方式：相对导入和绝对导入。相对导入例如 from. import B 或 from.. A import B，使用. 表示当前模块，使用.. 表示上层模块。绝对导入例如 import A. a 或者 from A import a。导入原理是在调用者方面，将模块或包的名称放入命名空间中，通过点表示法以＜module _ name＞作为前缀进行调用其中的对象。import 操作主要分为两个部分：

1）通过 _ import _ （name，globals＝None，local＝None，fromlist＝（），level＝0）函数搜索给定模块。函数中 name 是指要导入的模块名；globals 是字段类型，属于全局变量，用于确定如何解释包上下文中的名称；fromlist 是指要从 name 模块中导入的对象或子莫模块的名称；level 是指使用相对导入或绝对导入，0 表示绝对导入，＞0 表示相对导入（数值指示相对导入的层级）。

2）将搜索结果绑定到一个局部作用域，表现形式为使用 _ import _ （）函数搜索并创建模块，将其返回值进行 import 的 name binding 操作。

1.1 Pyhon 语言模块导入

在主程序运行中，使用系统内置函数 _ _ import _ （）即可实现模块的动态记载，但是面对非解释器默认路径下则无法实现动态加载功能，因此本文介绍 importlib 库以实现动态加载。

当导入一个特定模块时，python 首先会检查模块缓存 sys. modules，判断是否已经导入指定模块，若在 sys. modules 中找不到指定名称的模块则需要调用 python 的导入协议去实现查找和加载。此协议有两个概念性模块组成，即查找器（finder）和加载器（loader）。

查找器定义了一个模块查找机制，其原理是通过自身具备的所有策略去搜寻模块并反馈结果，其主要功能包含定位内置模块、冻结模块和指定路径下的模块，这些查找器全储存在 sys. meta_path 中，在 python3. 3 版本之后 sys. meta_path 对外公开查找器类型，不再隐藏式导入。查找器若处理成功则返回一个 ModuleSpec 对象，反之则返回 None。

加载器负责执行模块加载功能。导入机制调用 exec _ module（module）执行模块代码，加载器需要实现两个条件：1）若加载一个 python 模块，加载器必须在模

块的全局命名空间执行模块代码；2）若不能执行模块，则抛出错误。在执行加载之前，导入机制会初始化设置与导入相关的模块信息（基于 ModuleSpace 对象的信息）。ModuleSpec 主要包含有 _ name _ （模块名称）、_ loader _ （加载器对象）、_ file _ （模块路径）。

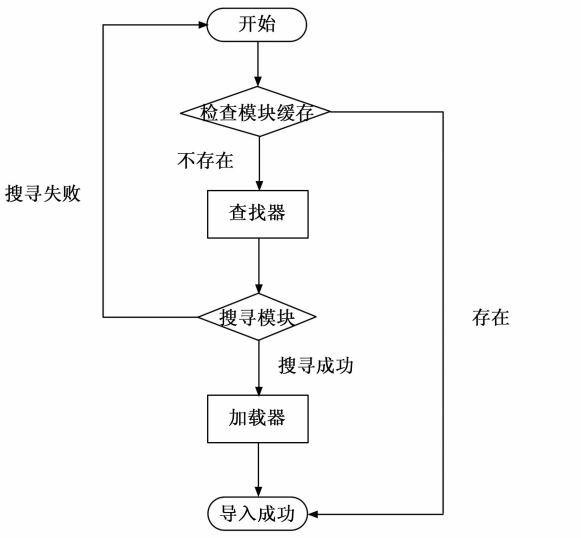

图 2　python 导入机制

1.2　Matlab 模块导入

除了 python 语言算法文件外，常用的算法文件语言还有 M 语言，即在 Matlab（Matrix Laboratory）软件中使用的编程语言，其代码文件格式主要是.m 文件。Matlab 软件作为一款全球著名的数学分析软件，对其他编程语言的交互性已经日益成熟。通过安装 MATLAB Engine API 实现在 python 中调用 matlab 功能模块。

其运行原理是在 python 中创建 matlab，engine 接口，调用函数 start _ matlab（）启动一个新 MATLAB® 进程，并返回 Python® 变量 eng，它是用于与 MATLAB 进程通信的 MatlabEngine 对象。通过 MatlabEngine 对象调用 matlab 自身或代码文件中的函数进行操作。

2　系统结构和设计

系统的主体框架使用跨平台语言 python 进行编码开发，UI 界面使用 PyQt5 开发库进行设计搭建，系统基于解释型编程语言开发，解释型编程语言是通过逐行解释执行代码来运行程序的，无需预先编译为机器码，因此，解释型编程语言通常具有一定的动态执行代码的能力，故以此为基础进行系统功能设计。

本系统包含界面展示、算法管理、远程通信接口等功能。由于试验数据和算法文件存储在远程服务器端的数据管理系统中，本系统设计新增远程通信接口和数据存储功能，用以实现数据和文件的远程传输。算法管理功能主要包含算法展示、算法同步、算法删除等子功能。算法同步功能作为本系统核心功能，细分为信息同步、模块动态加载、算法存储等。

系统设计原理是在系统运行状态下，通过管理页面导入新算法代码文件，在数据分析页面，通过动态加载机制调用新算法代码文件进行数据分析，同时利用数据库统一管理算法代码文件，不同计算机上的数据分析系统可同步调用及共享，从而降低系统功能扩展的技术门槛、开发成本与时间，提高数据分析系统可扩展性以及数据分析效率。

其目标是结合多种编程语言导入机制研制出模块动态加载技术。系统功能框架如图 3 所示。

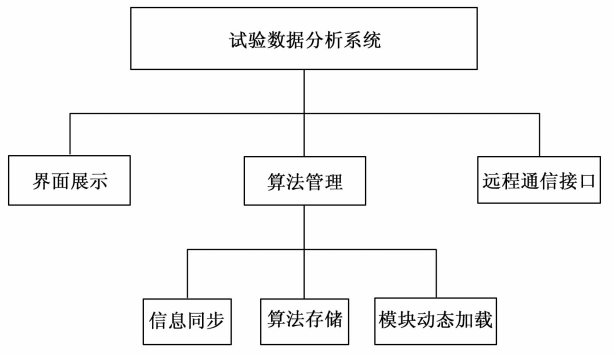

图 3　系统功能框架

远程试验数据分析算法库是管理试验信息和算法信息的统一平台。试验数据分析系统通过远程通信接口模块实现与远程试验数据分析算法库的交互，实现方式是发送请求获取算法信息和算法文件。系统将获取到的信息存储至本地数据库，算法文件解压缩存储至本地路径下。通过在线读取数据库的方式，系统将算法信息展示在界面上，并且根据算法信息和算法文件路径进行模块的动态加载从而界面呈现算法分析界面或结果。

系统数据流程如图 4 所示。

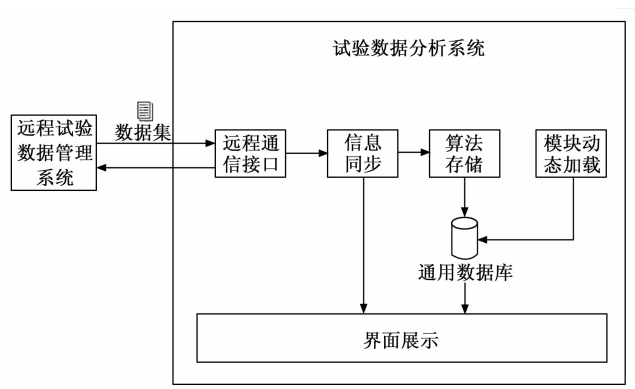

图 4　数据流程图

2.1　界面框架设计

系统采用 PyQt5 开发库进行界面搭建。PyQt 是 Qt

框架的 Python 语言实现，由 Riverbank Computing 开发。Qt 是一套 C++库和开发工具，包括平台图形用户界面、网络、线程、正则表达式、SQL 数据库、SVG、OpenGL、XML、用户和应用程序设置、定位和定位服务、网页浏览、3D 动画、图表、3D 数据可视化和与应用商店接口。PyQt5 将其中 1000 多个类实现为一组 Python 模块。PyQt5 的特性有：

1）基于高性能的 Qt 的 GUI 控件集。

2）能够跨平台运行在 Linux、Windows 系统上。

3）使用信号槽机制进行通信。

4）可以使用成熟的 IDE 界面进行界面设计，并自动生成可执行的 Python 代码。

5）提供一整套种类齐全的窗口控件。

2.2 远程通信接口

远程通信接口主要实现试验数据、算法信息、算法文件包的远程传输功能。考虑到多系统之间通常在同一局域网或公共网内，本文设计提出采用 WebService 接口实现远程通信功能。

WebService 技术是一种跨编程语言和跨操作系统平台的远程调用技术。它是基于网络的、分布式的模块化组件，遵守具体的技术规范，这些规范使得 WebService 能与其他兼容的组件进行交互操作。

WebService 遵循 SOAP 协议通过 XML 封装数据，最终通过 Http 协议传输数据。WebService 服务端首先通过 WSDL 文件来说明对外调用的服务内容。WSDL 是 Web Service Description Language 的缩写，即 web 服务描述语言，用于描述 WebService 及其方法、参数和返回值，客户端可通过一个 url 地址访问。WebService 通过 Http 协议发送请求和接收结果的内容都采用 XML 格式封装，并增加一些特定的消息头以说明内容格式。这些消息头和 XML 内容格式由 SOAP 协议规定。SOAP 协议（简单对象访问协议）是一种基于 http 的应用层协议。通过使用 http 发送的 XML 格式数据实现消息请求和发送。

系统采用 WebService 技术实现和远程试验数据分析算法库的数据交互功能。试验数据和算法信息的传输过程是系统向远程服务端发送查询请求，服务端接收请求调用方法进行查询，将结果反馈至系统中。算法文件包的传输过程是系统将算法信息生成 json 字符串作为请求参数向远程服务端发送下载请求，服务端接收请求和 json 字符串后在后台查询相关文件最终生成本地压缩包，远程服务端将下载地址反馈至系统中以便远程下载文件。

2.3 算法管理功能模块

算法管理功能模块是本系统的核心模块，主要包含等信息同步、算法存储、模块动态加载等功能。信息同步功能主要是通过远程通信接口实时获取算法信息并展示。算法存储功能主要是将信息同步获取的算法信息和算法文件通过不同的存储方式保存在本地。模块动态加载功能主要是通过 importlib 库、MATLAB Engine API 等技术调用算法文件内容，实现系统集成数据分析软件或直接调用分析函数，最终用以达到试验数据分析的目的。

2.3.1 信息同步

本文提出了通过远程通信接口实现算法信息和算法文件的传输来替代传统试验数据分析系统的本地导入功能，其优点在于提升系统自动化程度、简化操作流程、便于信息管理。

算法信息同步流程如下：

1）首先通过远程通信接口向试验数据分析算法库发出查询请求，试验数据分析算法库调用内部检索方法生成算法信息列表并返还；

2）系统接收到反馈消息，将算法信息展示；

3）在算法信息列表中选择单条或多条算法条目，系统根据选择内容生成含参消息，再次通过远程通信接口向试验数据分析算法库发送查询请求；

4）试验数据分析算法库接收含参消息，调用内部查询语句获取算法模块文件，并生成文件下载地址，以此作为含参消息返还；

5）系统再次接收到含有下载地址的消息，通过该地址进行算法文件下载，文件以字节流形式进行传输，系统将对字节流信息进行转换，在本地生成算饭模块文件；

6）系统在进行算法模块文件获取的同时，将选定的算法信息集通过内部存储接口存储至本地 MongoDB 数据库。

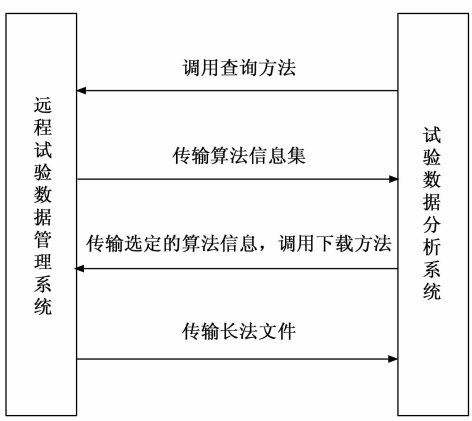

图 5 信息同步流程

当系统接收到算法信息和算法文件后，将进行算法信息存储和算法文件的本地存放。

2.3.2 算法存储

系统采用 MongoDB 数据库实现算法信息存储，相

较于 Oracle、Mysql 等数据库，其优势是：1）分布式存储数据，可以存储大容量数据和半结构化、不规则信息；2）高性能和高可用性，具备高效的数据存储和查询算法。在信息同步模块中，人为选定的算法信息将保存到 MongoDB 数据库以便后续系统查询展示。基于非关系型数据库存储算法代码文件及其属性，数据库中建立数据分析算法表，表中包含 ID、Algorithm _ Name、Function _ Description、Module _ Name、Programming _ Language、Upload _ Time、Source _ Unit、Responsible _ Person、Code _ File 列，其中 ID 选择自增序列（从 00000 - 99999）进行填充，其余列依次填充算法名称、功能描述、模块名称、编程语言、上传时间、来源单位、负责人、算法代码文件。

算法信息所关联的算法文件通常以压缩包形式下载至本地。系统配置文件中设置算法文件存放路径，系统将根据此存放路径解压缩算法文件，解压后的根文件名称与算法文件名称一致。最终在路径下形成一个文件夹就是一个算法的存放管理形式。此方法的优点在于在下次执行时，无需执行上述同步操作即可直接本地调用，节省了算法模块文件获取时间。

在算法模块文件执行前，若需要更新本地算法文件，可通过比对本地算法代码文件创建时间与远程试验数据分析算法库中算法功能文件的上传时间先后顺序判断是否进行更新。

2.3.3　模块动态加载

系统将采用 Importlib 库、MATLAB Engine API 等技术实现在系统运行中模块功能实时调用。

算法文件夹由两种内容组成：一种是软件项目式文件夹，通过对软件项目文件的调用实现系统嵌入分析软件；另一种是函数方式文件夹，由单个或多个文件组成，每个文件内容都是函数方法，系统可配合试验数据直接调用生成分析结果。

对于 Python 语言的算法文件，通常是以模块名称命名的算法代码文件集，调用文件中统一命名的类，将数据包或数据文件路径通过信号传递给该类，剩余工作由算法代码自动执行，若在数据分析过程中需要输入或者调整参数时，通过在算法代码文件执行时呈现的 UI 上操作完成。

首先将算法文件路径添加到 sys. path；其次，根据算法信息所提供的模块名称，采用 Importlib 库的 import _ module（）函数在算法文件中进行查询加载，如果查询加载成功则系统将获取到模块对象，其中包含参数属性、函数方法等内容；最终，系统可以调用模块函数进行软件界面初始化，以便后续算法分析处理。

对于 M 语言的算法文件，通常是函数方式式文件加载。首先调用 matlab. engine. start_matlab 函数启动 MATLAB® 进程；其次，根据算法文件夹路径和算法信息所提供的模块名称生成模块访问路径；系统通过 matlab. engine. cd 函数读取模块访问路径，将模块加载至 MATLAB® 进程中；最终，系统通过调用模块函数实现算法分析处理。

3　系统实现

基于动态加载技术的试验数据分析系统将根据上述系统设计进行搭建，本章将详细描述系统实现过程。

3.1　软件界面展示

系统界面布局可划分为 3 块，分别是菜单栏、算法库界面和算法管理界面。菜单栏作为顶部区域，主要显示系统名称、关闭按钮等。算法管理界面是左侧区域，主要展示算法按钮，实现通过点击操作打开算法处理界面，提供算法处理功能。算法库管理界面是中间区域，作为系统主要操作区域，初始化显示算法库管理界面，提供算法同步、算法删除和算法分析等操作按钮。

3.2　算法同步

在系统第一次运行时尚未显示有任务算法信息和处理展示，因而需要进行第一次算法同步操作，以便后续验证算法模块动态加载。

预览算法信息

在算法库界面，通过点击同步按钮进行算法信息同步操作。系统后台开始调用远程通信模块，通信地址根据系统配置文件生成，例如“http://192. 168. 0. 19:8080/OrientTDM/services/ToAlgorithm? wsdl”，其中红色字体部分为远程服务器 IP 地址和端口号，蓝色字体为试验数据服务系统对接的通信模块名称。在确保两个系统在同一网段内并且运行正常的情况下，系统调用 self. client. service. findAlgorithmall()，打开弹窗，采用 Pyqt5. QtWidget. tableWidget 控件以表格形式展示接收到的算法信息。

远程算法库信息主要包含统计算法名称、功能描述、模块名称、编程语言、上传时间、来源单位、负责人等属性，算法名称标识该算法的应用场景，功能描述详细阐述该算法的具体功能及使用方法，其他属性用于记录算法代码文件的一般描述信息。表格最左侧一列为多选复选框，通过勾选复选框获取所需算法集合。

3.3　算法存储和展示

3.3.1　下载算法文件包

在算法库信息弹窗中，完成勾选算法点击右下角更新按钮，系统将再次进行远程通信获取完整的算法信息和算法文件包。首先将算法所代表的 id 集合生成 id _ list 列表，考虑到 WebService 含参通信的特性，将 id _ list 列表转换为 id _ json 字符串并作为参数，系统调用 self. client. service. transferFiles（id _ json），在远程试

验数据分析算法库中生成算法文件集合的压缩包。

鉴于一套算法可能包含多个代码文件，因此算法文件统一以压缩包格式上传下载，压缩包格式选择 zip 格式，压缩包名称通常需要与解压后的根文件夹名称一致，根目录文件夹下可包含算法代码文件以及其他资源文件，故算法代码文件中资源调用的路径需设定为相对路径；

在系统中配置文件存放路径“\ algolib”，将算法文件压缩包通过远程下载至“algolib”文件夹并解压缩。

3.3.2 显示、存储算法信息

在系统后台接收算法信息和算法文件包时，先进行算法信息存储操作，再进行算法文件下载和解压缩，最终进行算法信息展示。

算法信息存储功能主要是从数据库中遍历算法信息集合，若检索到相应算法信息集合则进行数据覆盖；若未检索到相关信息集合则采用单项插入法将算法基础信息存储到 MongoDB 数据库。基于非关系型数据库存储算法模块文件及其属性，数据库中建立数据分析算法表，表中包含 ID、Algorithm _ Name、Function _ Description、Module _ Name、Programming _ Language、Upload _ Time、Source _ Unit、Responsible _ Person、Code _ File 列，其中 ID 选择自增序列进行填充，其余列依次填充算法名称、功能描述、模块名称、编程语言、上传时间、来源单位、负责人、算法代码文件。

算法信息展示功能主要系统后台将检索遍历 MongoDB 数据库的算法信息表，检索结果生成算法信息集合，在前端的算法信息管理界面将算法信息集合放入 Pyqt5. QtWidget. tableWidget 控件进行展示。

3.4 模块动态加载

在算法界面实现算法信息展示后，左侧算法管理区域将根据算法信息动态生成算法菜单栏，其内容是带有算法名称的按钮和图标，以便后续算法处理操作。

遍历算法信息集合 algo _ list，通过参数算法名称和算法文件名称的一致性查找到相对应的算法包。

通过参数算法语言的判断分别处理 python 语言算法包和 M 语言算法包。

考虑到系统需要动态生成图标和按钮控件，因而使用 python 代码去设计控件布局。根据算法名称使用 Pyqt5. QtWidget. QPushButton 控件创建按钮，调用 QPushButton. setStyleSheet（）设置按钮触发样式，调用 QPushButton. setIcon（）设置按钮图标。

针对 M 语言算法包，系统调用 creatEngin（）创建 Matlab 引擎 engine；根据当前算法获取算法包文件路径 file _ path，调用 engine. cd（file _ path）实现 matlab 进程打开文件路径。根据算法信息中填写的调用算法方法名称进行动态调用以便数据处理操作。

针对 python 语言算法包，系统调用 load _ python _ module（file _ path，module _ name，function）自定义函数实现动态加载算法。在自定义函数中有 3 个入参，其中 file _ path 是指当前算法包的文件路径；module _ name 是指在系统中算法库的模块名称，作为前缀名方便各算法的引用；function 是指当前算法的调用方法名。系统调用 importlib. import _ module（name）返回得到 module 实例，其中包含了详细的模块类和方法，最后调用 importlib. reload（module）实现系统能够调用模块功能。在系统调用代码模块中的函数方法时，一种类型是功能型或计算型代码功能，通过参数输入从而进行处理生成输出；另一种类型是由于某些算法分析过程中需要手动输入或调整参数，针对该类算法代码文件要求具备运行时展现 UI（User Interface，用户界面），由用户在页面上输入或调整参数，提高数据分析系统人机交互的友好性。代码如图 6 所示。

```
def load_python_module(file_path,modul_name, name):
    """ load python module.
    Args:
        file_path: python path
    Returns:
        dict: variables and functions mapping for specified python module

            {
                "variables": {},
                "functions": {}
            }
    """
    debugtalk_module = {
        "variables": {},
        "functions": {}
    }

    sys.path.insert(0, file_path)
    try:

        module = importlib.import_module(modul_name + '.' + name['name'] + '.' + name['type'][:-4] + '.' + name['function'])

        # 修复重载bug
        importlib.reload(module)

        sys.path.pop(0)

        return module
    except Exception as ex:
        logger.error('加载算法失败:'.format(ex), exc_info=True)
```

图 6 load _ python _ module 函数

在算法管理区域，点击算法名称将运行算法文件模块加载算法处理界面。

4 实验结果与分析

本文设计了基于动态加载技术的试验数据分析系统，实现了对多种算法语言文件的动态加载功能。具体操作步骤如下：

1）在系统数据分析界面，由于系统初始化因而尚未存储显示算法信息。通过点击同步按钮实现与试验数据分析算法库的通信，从而接收加载远程算法信息，如图 7 所示。

2）在上述同步窗口中勾选算法信息点击更新，系统将与试验数据分析算法库进行第二次通信。一方面在远程服务器实现算法文件打包，将下载地址以通信信息返回至系统后台以便下载操作；另一方面系统将算法信息更新存储在本地数据库。同步完成如图 8 所示。

图 7　同步算法信息

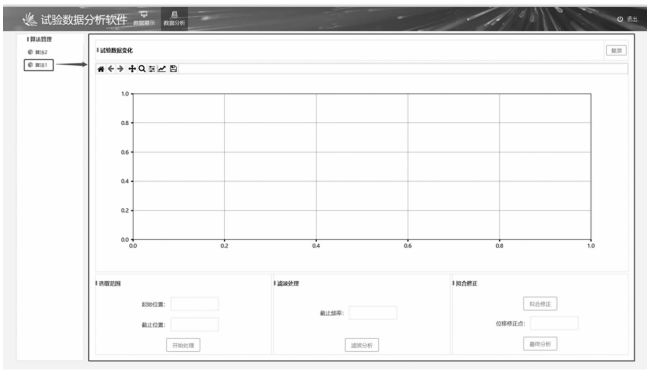

图 10　matlab 算法模块动态加载

图 8　展示算法信息

图 11　python 算法模块更新

3）在算法管理界面，当完成算法信息同步后，左侧导航树将动态加载算法名称和图标，并在后台完成页面初始化，其中将调用算法模块方法进行操作。

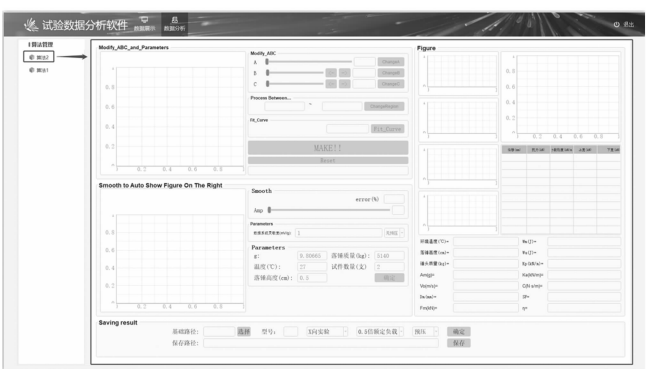

图 9　python 算法模块动态加载

上述操作完成算法信息的一次动态加载操作。若需要进行算法更新时，重复上述操作即可实现算法模块功能的动态更新加载。由图 11 可见动态更新加载后的界面相较于之前缩小了，其原因并不是因为右侧算法界面无法自适应或者软件进行了修改，而是因为界面内容来源于算法文件模块，进一步说明当算法文件更新时，软件对应算法功能模块也随之动态变化。

经过测试，通过远程通信接口实现多系统之间的数据传输，试验数据和算法信息不再仅限于本地导入。与试验数据分析算法库连接并非强关联，从而降低数据分析系统与试验数据分析算法库的耦合度，提高数据分析系统部署的灵活性。

系统实现动态获取算法信息和算法文件包，实时更新存储算法信息，界面动态展示算法信息，最终算法文件功能模块动态加载调用，整个系统运行稳定，符合系统设计要求。

5　结束语

本文介绍了基于动态加载技术的试验数据分析系统，一个利用动态加载技术调用算法文件进行界面展示和试验数据处理的 C/S 架构系统。系统主要实现了远程通信接口、算法管理、算法同步等功能。本文结合 importlib 库、WebService、MongoDB 数据库、PyQt5 图形界面框架等技术，设计并实现了试验数据分析系统，具体工作内容如下：

1）简要介绍了试验数据分析系统的功能模块和数据流程。

2）重点阐述了试验数据数据分析系统的核心功能设计。

3）详细展示整体系统的实现过程和功能演示效果。

系统成功解决了传统试验数据分析系统可扩展性差、灵活度低等问题，对于相关领域的试验数据分析系统设计具有一定的参考价值和借鉴意义。

该系统仍有改进优化的方面：

1）扩大算法文件语言，目前只适用于 Python 和 M 语言，提高系统对多语言文件的动态加载特性。

2）目前系统中对算法文件功能模块的动态调用仅限于模块层级，可以研究对模块内类、函数等动态调用。

参考文献：

[1] 张立国. 基于 Matlab/GUI 的地铁车辆试验数据分析系统 [J]. 城市轨道交通研究，2016，19 (12)：53－56.

[2] 杨建波，李 霄，任 佳. 基于 LabVIEW 的起落架落震试验数据分析系统设计 [J]. 计算机测量与控制，2016，24 (11)：261－264.

[3] 唐佑辉，许成义，赵云昌. 基于 PyQt5 的水深比对检测系统在水下地形测量检查中的应用 [J]. 山东国土资源，2023，39 (8)：58－62.

[4] 桑晓丹，郭 锐. 基于 PyQt5 的数字图像处理实验平台设计 [J]. 电子技术与软件工程，2021 (18)：129－130.

[5] 陶文玲，侯冬青. PyQt5 与 Qt 设计师在 GUI 开发中的应用 [J]. 湖南邮电职业技术学院学报，2020，19 (1)：19－21.

[6] 郭宗芝，刘 彬，邹玉龙，等. 基于模块动态加载机制的航天器软件重构方案研究 [J]. 计算机测量与控制，2018，26 (6)：126－129，148.

[7] 宁 涛，王 铮，龙 川. 面向一般应用的动态加载机制 [J]. 计算机工程，2008 (17)：79－81.

[8] 王政霞，黄大荣，孙艳红. 一般意义下动态加载机制的应用研究 [J]. 自动化技术与应用，2005 (6)：30－33.

[9] 李 威. 基于插件技术的服务开发框架设计与应用 [D]. 武汉：华中科技大学，2023.

[10] 钱宇虹. Java 动态加载与插件开发研究 [J]. 中小企业管理与科技 (下旬刊)，2015 (10)：156－157.

[11] 李俊晨. 基于 Android 平台的动态加载技术的研究与实现 [D]. 北京：北京邮电大学，2018.

[12] 董 杰. 基于动态加载技术的无人作战平台设备数据安全设计 [J]. 电子技术与软件工程，2018 (22)：76－78.

[13] 王 铮，杨志祥. 基于 .COM 文件的一种动态加载机制的研究 [J]. 计算机技术与发展，2007 (4)：45－48.

[14] 任 杰. 面向多终端的数据提供系统的研究与实现 [D]. 北京：北京邮电大学，2018.

[15] 焦圣明，严明良，郭 静，等. 基于 WebService 的分布式交通气象信息共享技术研究 [J]. 计算机工程与科学，2012，34 (3)：196－200.

[16] 王文哲. 多源异构科技共享资源管理系统的研究与实现 [D]. 武汉：武汉理工大学，2012.

[17] 刘卫国，陈 斌. 基于 Python 与 MATLAB 混合编程的文本分类应用案例设计 [J]. 软件导刊，2021，20 (7)：71－75.

[18] 谢树平，毛源豪. 基于 FPGA 的 DDR SDRAM 测试平台设计 [J/OL]. 计算机测量与控制：1－11 [2023-09-20]. http://kns.cnki.net/kcms/detail/11.4762.TP.20230719.1026.046.html.

[19] 李文韬，安加权. 基于 Web Service 技术的 PDM 与 TDM 集成应用研究 [J]. 计算机测量与控制，2012，20 (7)：1924－1927.

[20] 侯志勇. 固体火箭发动机试验数据分析处理软件设计 [J]. 计算机测量与控制，2010，18 (8)：1934－1936.

(上接第 154 页)

[20] LI J F，CHEN C X，WANG L. Fusion algorithm of multispectral images based on dual-tree complex wavelet transform and frequency-domain U-Net [J]. Journal of Biomedical Engineering Research，2020，39 (2)：145－150.

[21] YIN M，LIU X，LIU Y. Medical image fusion with parameter-adaptive pulse coupled neural network in nonsubsampled shearlet transform domain [J]. IEEE Transactions on Instrumentation and Measurement，2018，68 (1)：49－64.

[22] ZHAO Z X，XU S，ZHANG C X. DIDFuse：deep image decomposition for infrared and visible image fusion [C]. Twenty-Ninth International Joint Conference on Artificial Intelligence，Yokohama：IJCAI，2020，1：970－976.

[23] LIU Y，CHEN X，CHENG J. A medical image fusion method based on convolutional neural networks [C]. 2017 20th International Conference on Information Fusion (Fusion)，2017，1：1－7.

[24] LIU Z，BLASCH E，XUE Z. Objective assessment of multiresolution image fusion algorithms for context enhancement in night vision：a comparative study [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence，2011，34 (1)：94－109.

[25] ASLANTAS V，BENDES E. A new image quality metric for image fusion：the sum of the correlations of differences [J]. International Journal of Electronics and Communications，2015，69 (12)：1890－1896.

[26] WANG Z，BOVIK A C，SHEIKH H R. Image quality assessment：from error visibility to structural similarity [J]. IEEE Transactions on Image Processing，2004，13 (4)：600－612.

[27] SHEIKH H R，BOVIK A C. Image information and visual quality [J]. IEEE Transactions on Image Processing，2006，15 (2)：430－444.

[28] WANG Z，BOVIK A C，SHEIKH H R. Image quality assessment：from error visibility to structural similarity [J]. IEEE Transactions on Image Processing，2004，13 (4)：600－612.