



数据可视化平台搭建

李 柏 锐

北京信息科技大学
信息与通信工程学院





“目录”

C O N T E N T S

01

背 景

b a c k g r o u n d

02

设 计 思 路

d e s i g n t h o u g h t

03

代 码 展 示

C o d e p r e s e n t a t i o n

04

A P P 截 图 展 示

A P P S c r e e n s h o t d i s p l a y

05

总结以及未来想要开发的内容

S u m m a r y a n d w h a t y o u w a n t t o d e v e l o p i n t h e f u t u r e



1

“ 背景 ”

The background of this code is to build a data visualization platform designed to simplify the extraction, cleaning, analysis, and presentation of business data (such as user access data, transaction data, etc.).



背景

随着各项业务的上线，需要对业务数据（例如用户访问数据，交易数据等）进行提取，清洗，分析并形成可视化的报表/看板，各步骤可能由不同的技术/业务部门进行；在此背景下，实现一个统一的集中的数据分析平台是一个需求度较高。





2

“ 设计思路 ”

The design idea of this MATLAB App can be divided into several main parts: interface layout, data loading, data processing and data visualization

界面布局



使用 MATLAB 的 App Designer 创建用户界面（UI）。

这个 UI 包含以下组件：

UIFigure：主界面窗口，包含所有的 UI 元素。

LoadDataButton：按钮，用于加载数据文件（CSV 或 Excel）。

GenerateReportButton：按钮，用于生成数据报告。

ExportReportButton：按钮，用于导出报告为 PDF 文件。

DataTable：表格，用于显示加载的数据。

UIAxes1, UIAxes2, UIAxes3：三个图表区域，用于显示不同类型的数据可视化图表。





数据加载



通过点击 LoadDataButton 按钮，用户可以选择一个 CSV 或 Excel 文件，加载其中的数据。代码逻辑包括：

使用 `uigetfile` 打开文件选择对话框。

根据文件类型（CSV 或 Excel）使用 `readtable` 函数读取数据。

确保数据列名是有效的 MATLAB 变量名。

将数据列名统一重命名为英文，以便后续处理。

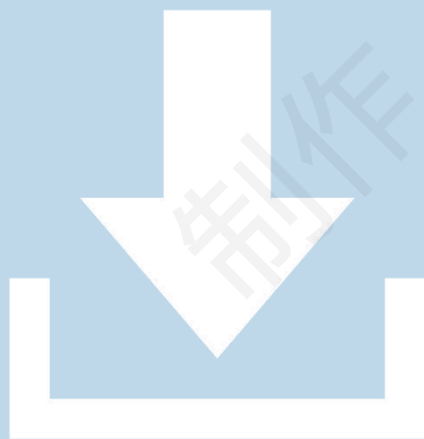
将加载的数据展示在 DataTable 表格中。

显示数据加载成功的提示信息。





数据处理



在数据加载完成后，通过点击

`GenerateReportButton` 按钮，可以对数据进行处理和分析。代码逻辑包括：

检查是否有数据被加载，如果没有则提示用户先加载数据。

对数据进行基本统计分析，并生成多个图表：

价格分布的直方图（显示在 `UIAxes1`）。

方向分布的柱状图（显示在 `UIAxes2`）。

代码类型的饼图（显示在 `UIAxes3`）。





数据可视化



在生成报告的过程中，数据会以多种图表形式展示：

直方图：显示数据的频率分布情况。

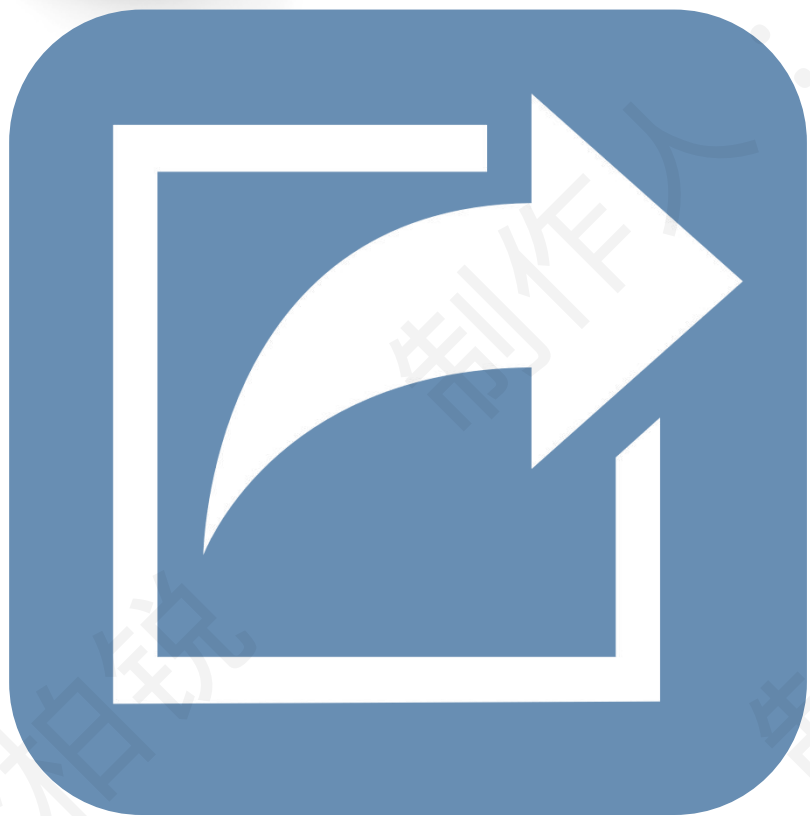
柱状图：展示不同分类的数据计数。

饼图：展示各个分类的占比情况。





导出报告



通过点击 `ExportReportButton` 按钮，可以将生成的报告导出为 PDF 文件。代码逻辑包括：

创建一个新的图窗，并使用 `tilayout` 布局。

在新的图窗中生成与 UI 中相同的图表。

添加一个文本框，用于显示数据摘要（总交易数、总数量、总费用）。

使用 `exportgraphics` 将图窗内容导出为 PDF 文件。

显示导出成功的提示信息。





设计思路总结



用户友好：提供了按钮和表格，用户可以通过简单的点击操作加载数据、生成报告和导出报告。

多样化的数据可视化：通过直方图、柱状图和饼图展示不同维度的数据分析结果，帮助用户更好地理解数据。

易于扩展：代码结构清晰，可以根据需要扩展更多的数据分析和可视化功能。

这个设计旨在提供一个直观且功能强大的数据可视化工具，适用于各种业务数据的分析和报告生成。





3

“ 代 码 展 示 ”

This code is a MatLAB-based data visualization application designed to help users load and analyze business data from CSV or Excel files.



代码展示

```
classdef EnhancedDataVisualizationApp < matlab.apps.AppBase
```

```
% 对应应用组件的属性
```

```
properties (Access = public)
```

```
    UIFigure      matlab.ui.Figure      % 主界面窗口
    LoadDataButton matlab.ui.control.Button % 加载数据按钮
    GenerateReportButton matlab.ui.control.Button % 生成报告按钮
    ExportReportButton matlab.ui.control.Button % 导出报告按钮
    DataTable      matlab.ui.control.Table % 数据表格
    UIAxes1        matlab.ui.control.UIAxes % 第一个图表区域
    UIAxes2        matlab.ui.control.UIAxes % 第二个图表区域
    UIAxes3        matlab.ui.control.UIAxes % 第三个图表区域
```

```
end
```

```
properties (Access = private)
```

```
    Data % 表格数据
```

```
end
```

```
% 处理组件事件的回调函数
```

```
methods (Access = private)
```

```
% 按钮按下函数：LoadDataButton
```

```
function LoadDataButtonPushed(app, event)
```

```
    [file, path] = uigetfile({'*.csv'; '*.xlsx'}, 'Select Data File'); % 打开文件
```

```
    选择对话框
```

```
    if isequal(file, 0)
```

```
        return;
```

```
    end
```

```
    filePath = fullfile(path, file); % 获取文件路径
```

```
    [~, ~, ext] = fileparts(filePath);
```

```
    if strcmp(ext, '.csv')
```

```
        app.Data = readtable(filePath, 'VariableNamingRule', 'preserve');
```

```
% 读取CSV文件
```

```
    elseif strcmp(ext, '.xlsx')
```

```
        app.Data = readtable(filePath, 'Sheet', 1, 'VariableNamingRule',  
'preserve'); % 读取Excel文件
```

```
    end
```

```
% 确保列名是有效的MATLAB变量名
```

```
    app.Data.Properties.VariableNames =  
    matlab.lang.makeValidName(app.Data.Properties.VariableNames);
```





代 码 展 示

```
% 统一重命名列名
app.Data.Properties.VariableNames = {'TradeDate', 'SettlementDate', 'Account',
'Code', 'Direction', 'Price', 'Quantity', 'Fee1', 'Fee2'};

% 在表格中显示数据
app.DataTable.Data = app.Data;

% 显示加载成功消息
uialet(app.UIFigure, '数据加载成功', '成功');
end

% 按钮按下函数：GenerateReportButton
function GenerateReportButtonPushed(app, event)
    if isempty(app.Data)
        uialet(app.UIFigure, '请先加载数据', '错误'); % 提示先加载数据
        return;
    end

    % 数据可视化示例
    % 价格分布的直方图
    histogram(app.UIAxes1, app.Data.Price);
    title(app.UIAxes1, '价格分布');
    xlabel(app.UIAxes1, '价格');
    ylabel(app.UIAxes1, '频次');
```


```
% 方向分布的柱状图
directionCounts = groupcounts(app.Data.Direction);
bar(app.UIAxes2, directionCounts);
title(app.UIAxes2, '方向分布');
xlabel(app.UIAxes2, '方向');
ylabel(app.UIAxes2, '数量');

% 代码类型的饼图
[codeTypes, ~, idx] = unique(app.Data.Code);
codeCounts = accumarray(idx, 1);
pie(app.UIAxes3, codeCounts);
title(app.UIAxes3, '代码类型饼状图');
end

% 按钮按下函数：ExportReportButton
function ExportReportButtonPushed(app, event)
    if isempty(app.Data)
        uialet(app.UIFigure, '请先加载数据', '错误'); % 提示先加载数据
        return;
    end

    % 创建新的图窗用于导出
    fig = figure('Visible', 'off');
    t = tiledlayout(fig, 2, 2);

    % 价格分布的直方图
    nexttile(t);
    histogram(app.Data.Price);
    title('价格分布直方图');
    xlabel('价格');
    ylabel('频次');
```





代 码 展 示

% 方向分布的柱状图

```
nexttile(t);
```

```
directionCounts = groupcounts(app.Data.Direction);
```

```
bar(directionCounts);
```

```
title('方向分布柱状图');
```

```
xlabel('方向');
```

```
ylabel('数量');
```

% 代码类型的饼图

```
nexttile(t);
```

```
[codeTypes, ~, idx] = unique(app.Data.Code);
```

```
codeCounts = accumarray(idx, 1);
```

```
pie(codeCounts);
```

```
title('代码类型饼状图');
```

% 添加文本摘要

```
nexttile(t);
```

```
axis off; % 关闭坐标轴
```

```
summaryText = sprintf('Total Trades: %d\nTotal Quantity: %d\nTotal Fee1: %.2f\nTotal Fee2: %.2f, ...
```

```
height(app.Data), sum(app.Data.Quantity), sum(app.Data.Fee1), sum(app.Data.Fee2));
```

```
text(0.5, 0.5, summaryText, 'HorizontalAlignment', 'center', 'VerticalAlignment', 'middle');
```

% 导出为PDF

```
exportgraphics(t, 'Report.pdf', 'ContentType', 'vector');
```

```
close(fig);
```

% 显示导出成功消息

```
uiaalert(app.UIFigure, '报告已导出为PDF格式请在文件中查收', '成功');
```

```
end
```

```
end
```





代码展示

```
% 组件初始化
methods (Access = private)

% 创建UIFigure和组件
function createComponents(app)

% 创建UIFigure并隐藏，直到所有组件创建完成
app.UIFigure = uifigure('Visible', 'off');
app.UIFigure.Position = [100 100 800 600];
app.UIFigure.Name = 'Enhanced Data Visualization App';

% 创建LoadDataButton
app.LoadDataButton = uibutton(app.UIFigure, 'push');
app.LoadDataButton.ButtonPushedFcn = createCallbackFcn(app,
@LoadDataButtonPushed, true);
app.LoadDataButton.Position = [20 550 100 22];
app.LoadDataButton.Text = '加载数据';

% 创建GenerateReportButton
app.GenerateReportButton = uibutton(app.UIFigure, 'push');
app.GenerateReportButton.ButtonPushedFcn = createCallbackFcn(app,
@GenerateReportButtonPushed, true);
app.GenerateReportButton.Position = [140 550 120 22];
app.GenerateReportButton.Text = '综合报告';

% 创建ExportReportButton
app.ExportReportButton = uibutton(app.UIFigure, 'push');
app.ExportReportButton.ButtonPushedFcn = createCallbackFcn(app,
@ExportReportButtonPushed, true);
app.ExportReportButton.Position = [280 550 100 22];
app.ExportReportButton.Text = '输出报表';
```





代 码 展 示

```
% 创建DataTable
app.DataTable = uitable(app.UIFigure);
app.DataTable.ColumnName = {'TradeDate', 'SettlementDate', 'Account',
'Code', 'Direction', 'Price', 'Quantity', 'Fee1', 'Fee2'};
app.DataTable.RowName = {};
app.DataTable.Position = [20 300 760 200];

% 创建用于直方图的UIAxes
app.UIAxes1 = uiaxes(app.UIFigure);
app.UIAxes1.Position = [20 50 360 200];

% 创建用于柱状图的UIAxes
app.UIAxes2 = uiaxes(app.UIFigure);
app.UIAxes2.Position = [400 50 180 200];

% 创建用于饼图的UIAxes
app.UIAxes3 = uiaxes(app.UIFigure);
app.UIAxes3.Position = [600 50 180 200];

% 显示所有组件创建完成后的窗口
app.UIFigure.Visible = 'on';
end
end
```

```
% 应用初始化和构造
methods (Access = public)

% 构造应用
function app = EnhancedDataVisualizationApp

% 创建和配置组件
createComponents(app)

% 向App Designer注册应用
registerApp(app, app.UIFigure)

if nargin == 0
    clear app
end

% 在应用删除前执行的代码
function delete(app)

% 删除应用时删除UIFigure
delete(app.UIFigure)
end
end
end
```





4

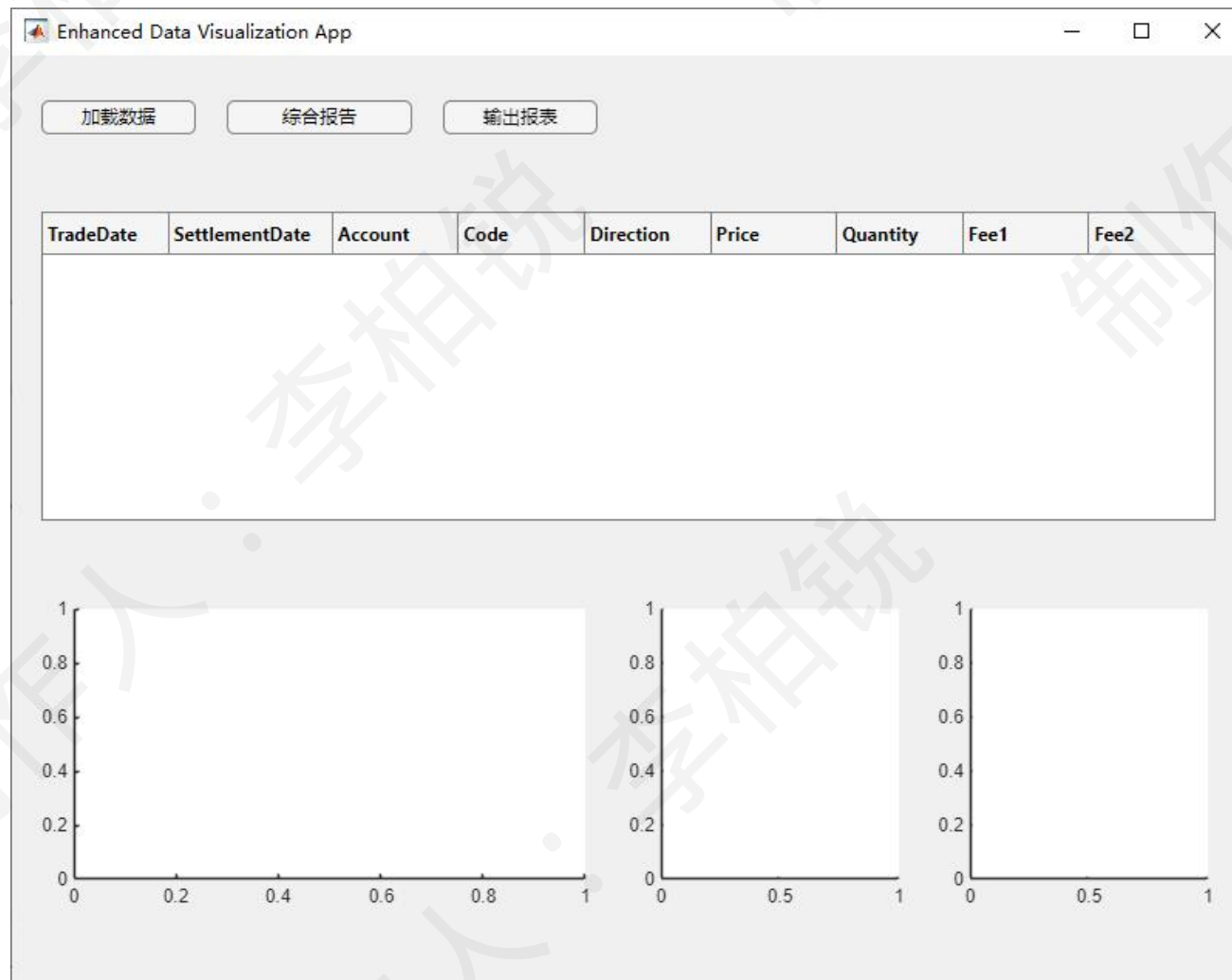
“APP截图展示”

With a simple button operation, users can load data files and generate multiple charts, including price distribution histograms, directional distribution bars, and code type pie charts. The application also supports exporting the generated visual reports as PDF files, making it easy for users to view and share analysis results.

APP 截图展示

这个基于 MATLAB 的数据可视化应用程序界面简洁明了，用户通过点击按钮即可加载 CSV 或 Excel 数据文件，并生成价格分布直方图、方向分布柱状图和代码类型饼图。数据表格实时展示加载的数据，且支持将生成的报告导出为 PDF 文件，便于分享和存档。整个过程高效直观，适用于企业的业务数据分析和报告生成。

下图是我设计的APP界面截图



APP截图展示

由于MATLAB无法加载中文字体，所以需将报表的表头（即每列的列头）改成英文具体如下：

交易日 -> TradeDate

交收日 -> SettlementDate

账户 -> Account

代码 -> Code

方向 -> Direction

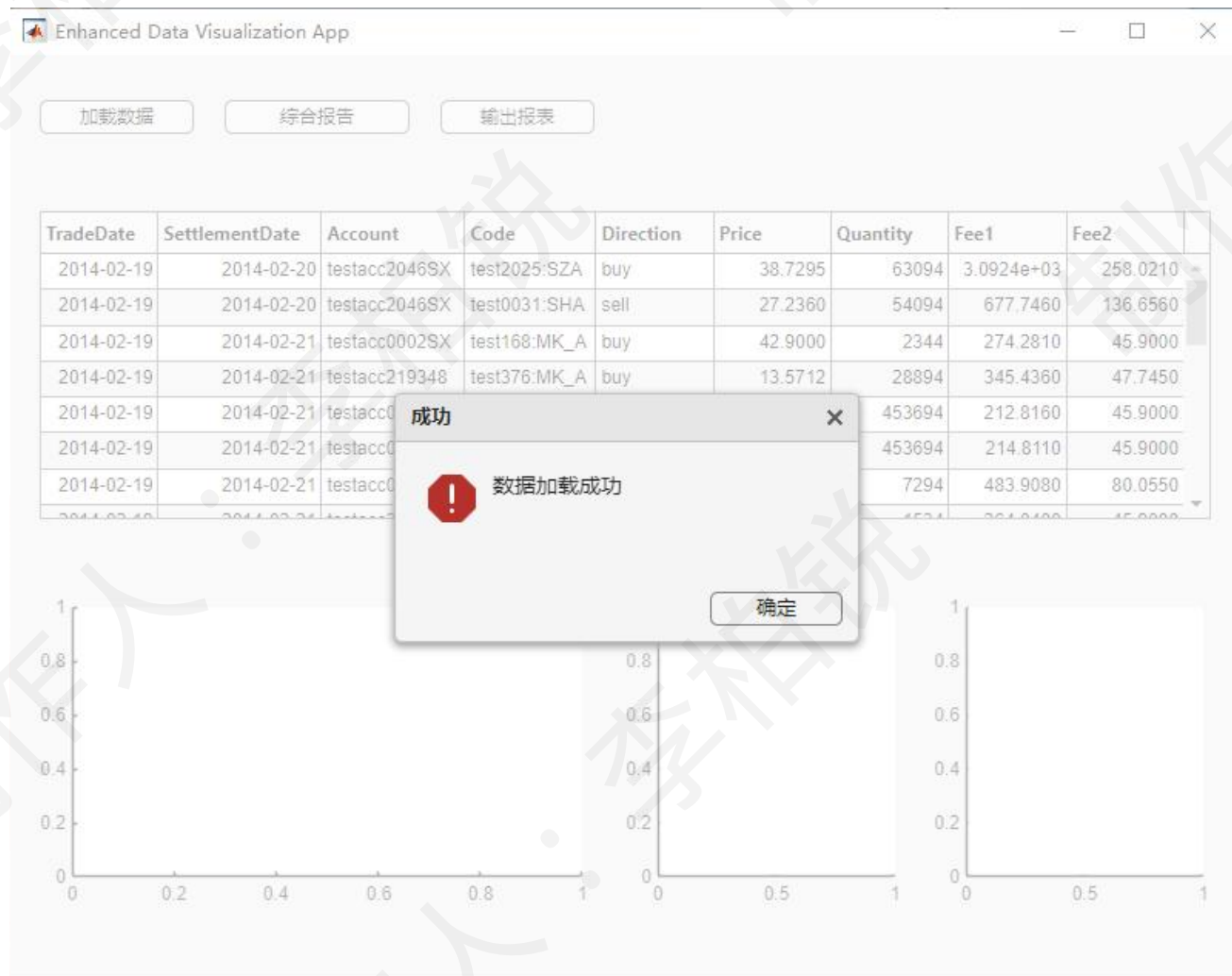
价格 -> Price

数量 -> Quantity

费用1 -> Fee1

费用2 -> Fee2

下图是加载表格数据后的界面截图



APP截图展示

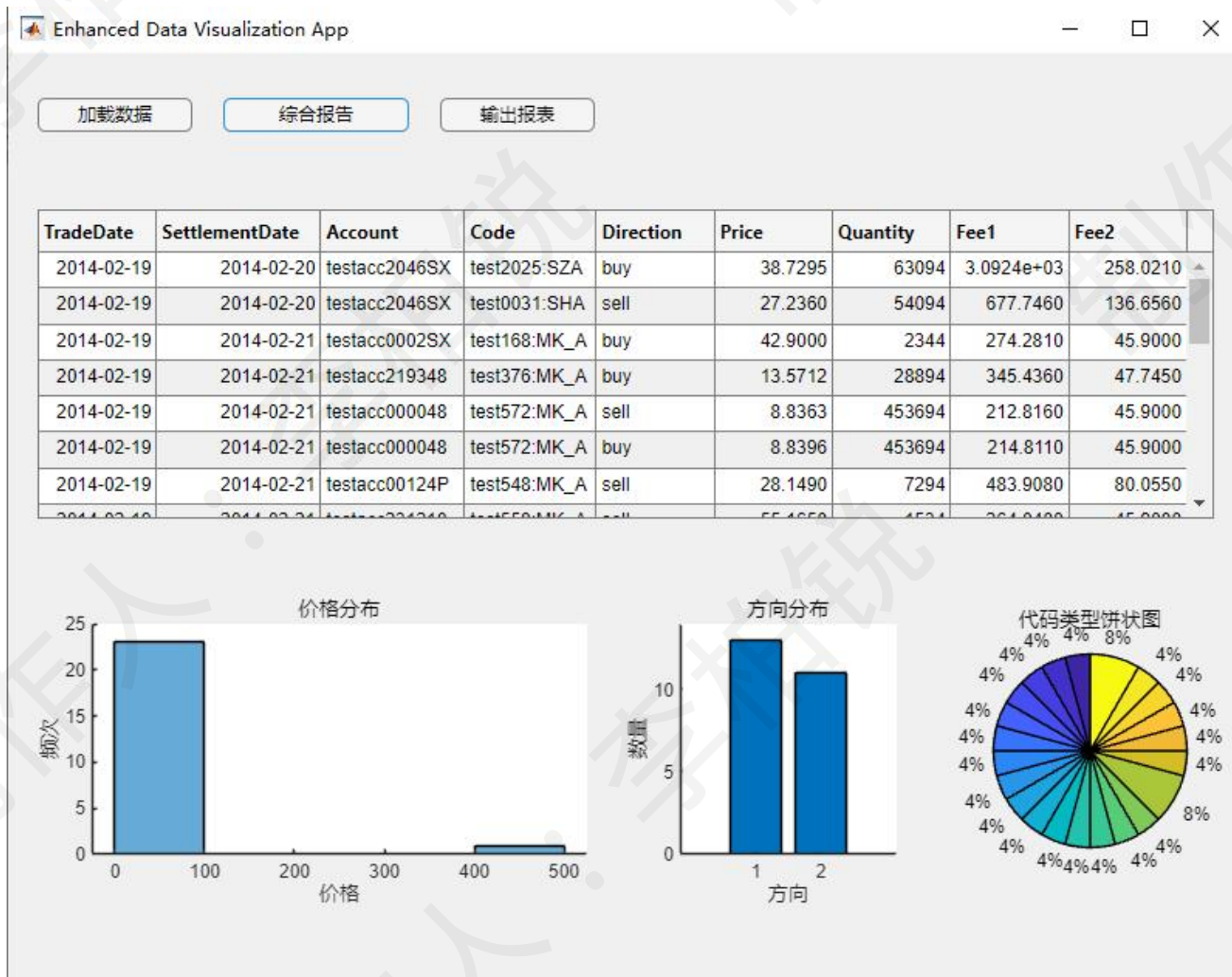
由于提供的报表太大了，为了方便并快速的分析出结果，我只截取了前25行的表格进行综合分析。

左下图表为直方图：展示价格分布，帮助用户了解价格数据的频率分布情况。

中下图为柱状图：展示交易方向分布，用户可以看到不同交易方向的数量分布。

右下图为饼图：展示交易代码类型分布，便于用户了解不同类型交易在整体中的占比。

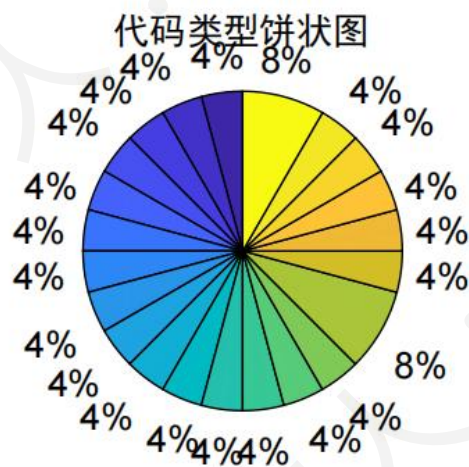
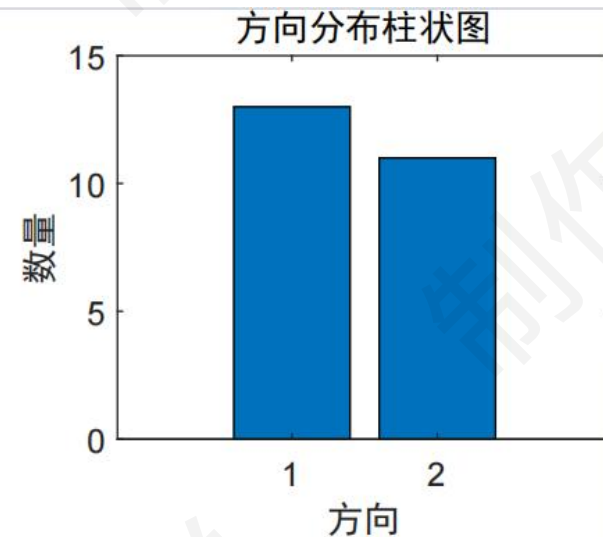
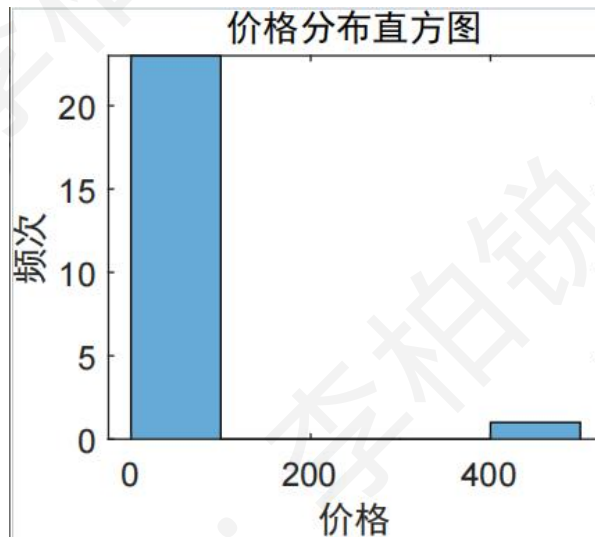
下图是点击综合报告后的界面截图





该代码输出的 PDF 报告包含价格分布直方图、方向分布柱状图和代码类型饼图，并附有关键数据统计摘要（总交易数、总数量、总费用）。所有内容采用支持中文的字体，确保在 PDF 中正确显示。报告布局清晰，导出高质量，便于打印和分享，帮助用户快速理解业务数据的主要特征和趋势。

下图是输出的 P D F 报表的界面截图



总交易数: 24
总数量: 2019426
总费用1: 18993.58
总费用2: 2526.07

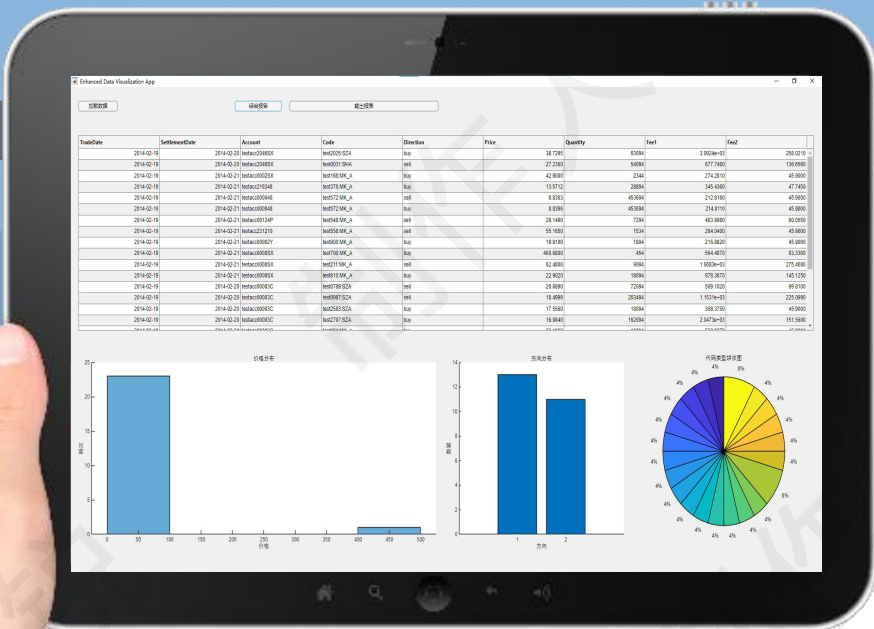


5

总结以及未来想要开发的内容

Summary and what you want to develop in the future.

总结以及未来想要开发的内容

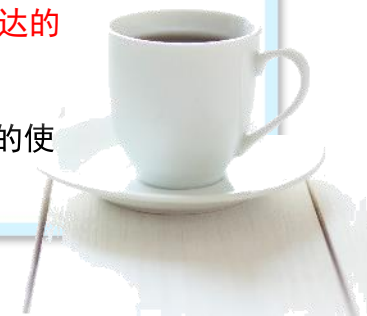


总结

这个 MATLAB 应用程序通过直观的用户界面，实现了从 CSV 或 Excel 文件加载业务数据，并生成价格分布直方图、方向分布柱状图和代码类型饼图，同时提供关键数据统计摘要。用户可以导出高质量的 PDF 报告，布局清晰，支持中文显示。整个设计思路是简化数据提取、清洗、分析和展示的过程，提供一站式的数据可视化和报告生成工具，帮助用户高效地进行业务数据分析和决策支持。

在未来我想要增加或改进的内容如下：

- 1.由于MATLAB无法加载中文字体，需将报表的表头（即每列的列头）改成英文，所以我想找到一个方法让这个APP可以同时识别中文和英文两种形式的列头。
- 2.我想增加更多的分析方式，并运行更多的统计图，统计方式来表达更多的内容，其目的均为让用户能更加清晰的看到以及感受到数据被图像所表达的魅力。
- 3.我想在输出PDF的时候增加英文版或更多语种，以此来增加外国用户的使用体验，使得此app被更多的用户所使用。





谢谢观看

APP制作人：李柏锐 制作时间：2024/6/9

最后我要感谢华泰证券出的竞赛题目。同时，感谢所有使用这款应用程序的用户（我的家人），是他们的实际操作和意见帮助我们不断改进和优化功能，使其更加符合实际需求。感谢 MATLAB 提供的强大开发平台，使得我们能够高效地实现数据提取、清洗、分析和可视化展示。希望这款应用程序能够在您们的业务数据分析和决策过程中发挥积极的作用。

