本周，本团队主要着手进行数据清洗。经过与李泽华学长的沟通，我们认识到了自己上周所做的内容偏离了增长的轨道。这周重回正轨，着手把数据从原始数据变为以生产批次号为标示的`X-Y`型数据。

1. 数据提取环节

进行数据处理，首先需要把excel文件中的数据提取到数据处理的工具里面去。

预定使用的软件为Matlab R2017b。准备先把各个sheet导入到Matlab的变量里面，然后再对这些变量写特定的程序脚本来处理。

数据的提取分为两个部分，一个是生产过程中的数据，一个是过程检验数据。为了制成excel表格，我们需要对数据进行提取。

1.1. 生产过程数据的提取

使用如下的代码把`1 生产过程统计数据.xlsx`中的内容提取到matlab的工作变量中：

`ExtractInfo.m`

```
%% 提取生产过程的数据
filename = "StatisticsInProduction.xlsx";
for i = 1:24
    sheetname = GenSheetname(i);
    xls = xlsread(filename,sheetname);
    endrow = length(xls) + 1;
    if i <= 6
        varname = "15" + string(i + 6);
    else
        if i <= 18
            varname = "16" + string(i - 6);
        else
            varname = "17" + string(i - 18);
        end
    end
    eval('raw_' + varname + ' = ' + 'importfilefromexcel(filename, sheetname, 2,
endrow);');
end

% 更改列名称
for i = 1:24
    if i <= 6
        varname = "15" + string(i + 6);
    else
        if i <= 18
            varname = "16" + string(i - 6);
        else
            varname = "17" + string(i - 18);
        end
    end
```

```
    eval('raw_' + varname + ".Properties.VariableNames = {'Number', 'Workorder', 'Date',
'Max', 'Min', 'Average', 'SD', 'CPK', 'Confidence', 'Varname', 'Segment',
'Category'};");
end

%% 提取制叶段的测试结果
filename = "LeafTest.xlsx";

xls = xlsread(filename,"大片率");
endrow = length(xls) + 2;
raw_bigleaf = importfilefromexcel2(filename, "大片率", 3, endrow);

xls = xlsread(filename,"中片率");
endrow = length(xls) + 2;
raw_midleaf = importfilefromexcel2(filename, "中片率", 3, endrow);

xls = xlsread(filename,"大中片率");
endrow = length(xls) + 2;
raw_bmleaf = importfilefromexcel2(filename, "大中片率", 3, endrow);

xls = xlsread(filename,"碎片率");
endrow = length(xls) + 2;
raw_brokenleaf = importfilefromexcel2(filename, "碎片率", 3, endrow);

%更改列名称
raw_bigleaf.Properties.VariableNames = {'undefined', 'Number', 'Value', 'Time',
'Workorder', 'Productionorder', 'Category'};
raw_bmleaf.Properties.VariableNames = {'undefined', 'Number', 'Value', 'Time',
'Workorder', 'Productionorder', 'Category'};
raw_midleaf.Properties.VariableNames = {'undefined', 'Number', 'Value', 'Time',
'Workorder', 'Productionorder', 'Category'};
raw_brokenleaf.Properties.VariableNames = {'undefined', 'Number', 'Value', 'Time',
'Workorder', 'Productionorder', 'Category'};

%% 提取制丝段的测试结果
filename = "CuttobaccoTest.xlsx";

xls = xlsread(filename, "长丝率");
endrow = length(xls) + 2;
raw_longsi = importfilefromexcel3(filename, "长丝率", 3, endrow);

xls = xlsread(filename, "短丝率");
endrow = length(xls) + 2;
raw_shortsi = importfilefromexcel3(filename, "短丝率", 3, endrow);

xls = xlsread(filename, "中丝率");
endrow = length(xls) + 2;
raw_midsi = importfilefromexcel3(filename, "中丝率", 3, endrow);

xls = xlsread(filename, "整丝率");
endrow = length(xls) + 2;
raw_completesi = importfilefromexcel3(filename, "整丝率", 3, endrow);
```

```
xls = xlsread(filename, "碎丝率");
endrow = length(xls) + 2;
raw_brokensi = importfilefromexcel3(filename, "碎丝率", 3, endrow);

xls = xlsread(filename, "填充值");
endrow = length(xls) + 2;
raw_fillsi = importfilefromexcel3(filename, "填充值", 3, endrow);


%更改列名称
raw_fillsi.Properties.VariableNames = {'undefined', 'Number', 'Value', 'Time',
'Workorder', 'Productionorder', 'Category'};
raw_longsi.Properties.VariableNames = {'undefined', 'Number', 'Value', 'Time',
'Workorder', 'Productionorder', 'Category'};
raw_shortsi.Properties.VariableNames = {'undefined', 'Number', 'Value', 'Time',
'Workorder', 'Productionorder', 'Category'};
raw_midsi.Properties.VariableNames = {'undefined', 'Number', 'Value', 'Time',
'Workorder', 'Productionorder', 'Category'};
raw_completesi.Properties.VariableNames = {'undefined', 'Number', 'Value', 'Time',
'Workorder', 'Productionorder', 'Category'};
raw_brokensi.Properties.VariableNames = {'undefined', 'Number', 'Value', 'Time',
'Workorder', 'Productionorder', 'Category'};
```

代码中，`function GenSheetname`是一个自定义的函数，它可以根据月份序号得到excel表中对应的sheet名。`eval`函数可以方便我们对变量名进行命名，这也是解释型语言的一个优势。核心函数是`importfilefromexcel(filename, sheetname, startrow, endrow)`，它的定义如下（由MATLAB的GUI程序辅助生成）。

`importfilefromexcel1.m`

```
function tableout = importfilefromexcel1(workbookFile,sheetName,startRow,endRow)
%% Input handling

% If no sheet is specified, read first sheet
if nargin == 1 || isempty(sheetName)
    sheetName = 1;
end

% If row start and end points are not specified, define defaults
if nargin <= 3
    startRow = 2;
    endRow = 45603;
end

%% Import the data, extracting spreadsheet dates in Excel serial date format
[~, ~, raw, dates] = xlsread(workbookFile, sheetName,
sprintf('A%d:L%d',startRow(1),endRow(1)),'' , @convertSpreadsheetExcelDates);
for block=2:length(startRow)
    [~, ~, tmpRawBlock,tmpDateNumBlock] = xlsread(workbookFile, sheetName,
sprintf('A%d:L%d',startRow(block),endRow(block)),'' , @convertSpreadsheetExcelDates);

    raw = [raw;tmpRawBlock]; %#ok<AGROW>
```

```
    dates = [dates;tmpDateNumBlock]; %#ok<AGROW>
end
raw(cellfun(@(x) ~isempty(x) && isnumeric(x) && isnan(x),raw)) = {''};
stringVectors = string(raw(:,[2,10,11,12]));
stringVectors(ismissing(stringVectors)) = '';
raw = raw(:,[1,4,5,6,7,8,9]);
dates = dates(:,3);

%% Replace non-numeric cells with NaN
R = cellfun(@(x) ~isnumeric(x) && ~islogical(x),raw); % Find non-numeric cells
raw(R) = {NaN}; % Replace non-numeric cells
R = cellfun(@(x) ~isnumeric(x) && ~islogical(x),dates); % Find non-numeric cells
dates(R) = {NaN}; % Replace non-numeric Excel dates with NaN

%% Create output variable
I = cellfun(@(x) ischar(x), raw);
raw(I) = {NaN};
data = reshape([raw{:}],size(raw));

%% Create table
tableout = table;

%% Allocate imported array to column variable names
tableout.VarName1 = data(:,1);
tableout.VarName2 = categorical(stringVectors(:,1));
dates(~cellfun(@(x) isnumeric(x) || islogical(x), dates)) = {NaN};
tableout.VarName3 = datetime([dates{:,1}].', 'ConvertFrom', 'Excel');
tableout.VarName4 = data(:,2);
tableout.VarName5 = data(:,3);
tableout.VarName6 = data(:,4);
tableout.SD = data(:,5);
tableout.CPK = data(:,6);
tableout.VarName9 = data(:,7);
tableout.VarName10 = stringVectors(:,2);
tableout.VarName11 = categorical(stringVectors(:,3));
tableout.VarName12 = categorical(stringVectors(:,4));

% For code requiring serial dates (datenum) instead of datetime, uncomment
% the following line(s) below to return the imported dates as datenum(s).

% tableout.VarName3=datenum(tableout.VarName3);
```

成功提取之后，将其保存为`.matlab`文件，方便之后的读取，从此就可以摆脱excel，直接在Matlab上面进行数据处理了。

1.2. 过程检验数据的提取

过程检验的数据分为叶片过程检验数据和叶丝过程检验数据。需要分别进行提取。这个过程中变量少，所以提取需要花的时间并不多。

利用同样的方法，对检验数据进行提取。

由于数据量大，整个数据提取过程耗费了较多时间（约几十分钟），可能是算法方面的问题。从这方面也可以看出，数据量庞大的时候一个优秀的算法的重要性。

1.3. 检验数据并生成.matlab文件保存

采用Matlab中数据和excel文档中数据抽查比对的方式确定数据提取的准确性。经过检查比对，数据无误，可以保存。

保存之后，将过程中使用的算法以及生成的`stats.matlab`文档使用`Git`加入Version Control，然后上传至GitHub。

最后，因为数据需要保密的原因，具体提取出来的数据以及excel原文档不会上传到GitHub中。

1.4. 数据格式内容备注

由于MATLAB中`cell`类型的数据占内存太大，我的PC机不能处理，所以使用`table`类型的数据。因为matlab中不支持中文作为列名，所以在列名分别为：

1. 在生产过程统计数据中

序号	工单号	日期	最大值	最小值	平均值	SD	CPK	置信值	参数	工艺段	牌号
Number	Workorder	Date	Max	Min	Average	SD	CPK	Confidence	Varname	Segment	Category

2. 在叶片和叶丝检验数据中：

检验项目	序号	检验值	录入时间	生产工单号	生产批次号	物料
undefined	Number	Value	Time	Workorder	Productionorder	Category

特此记录，以备后用。

2. 对提取的数据进行预处理

首先，尝试使用自己编的函数来遍历。最后发现效率太低，遂学习MATLAB自带函数，尝试使用它的函数来进行编程，同时学习table数据变量的用法。

2.1. excel文件的熟悉

假设原`1 生产过程统计数据`的工单号是连续的（即中间没有断开）（事实上检测了数个工单，并没有发现太大违背假设的情况，以后如果出现例外，再进行处理），继续进行处理。

在研究`1 生产过程统计数据`之后，发现不同工单号对应的检测变量并没有特定的规律。具体体现在：

> 有的一次加料34个变量，有的一次加料一次加料52个变量，有的一次加料40个变量，变量个数完全不统一

目前的解决做法是，列出所有可以检测的变量（17年1月），生成一个`string array`类型存放`1 - ?`所对应的变量。然后进行数据提取到另一个table的环节的时候，按照`string array`表的对应关系，一个批次号一个批次号地填入。

如何找到所有的变量？利用MATLAB内部的去重函数`unique`，统计有多少个对应的变量。`unique`函数会自动帮我们把对应的`string array`排序。

2.2. 数据预处理环节

对各个excel表进行熟悉了之后，编写代码进行对应的数据清洗（预处理）。具体成果是`PreProcess_Beta`函数。它的执行是基于之前获得的`raw_stats.mat`的。具体的逻辑如下：

```
1. 查找本月所有的生产批次号(因为之研究制叶段和制丝段，生产批次号中带'X'或者'Y'的被忽略);
2. 根据查到的生产批次号， 查找对应的制叶段， 制丝段的生产过程数据和检测数据， 并最终生成表格.
3. return 生成的表格
```

根据以上的逻辑，写出了如下的函数代码：

```
function [xy_stats, variables] = PreProcessData_Beta(month, year, raw_data)
%本脚本可以把中某一个月的原始数据按照x-y的关系生成一个新的table表格。
%（然后保存到一个.csv或者excel文件中（以可以方便之后的为准））。
%
%输入参数：两个数字，一个raw_data。其中，年份是两位数字，如17，raw_data如raw_171。
%建议使用本函数之前clear一下变量空间，防止干扰，产生意想不到的结果
%版本：1.1

%% 读取需要使用的数据（从文件中）
load raw_stats.mat relationship raw_fillsi raw_longsi raw_midsi raw_shortsi raw_brokensi
raw_completesi raw_bigleaf raw_bmleaf raw_midleaf raw_brokenleaf;
%% 常数性变量。
variables = unique(raw_data.Varname);
%save(%得到本月有的所有的变量。并输出，便于后续处理。
%得到 string monthyear，为之后的查找做准备（table T的处理）
if month <= 9
    monthyear = "20" + string(year) + "0" + string(month);
else
    monthyear = "20" + string(year) + string(month);
end



po_array = relationship.Productionorder;
po_array = char(po_array);
po_array = po_array(:,1:6);
po_array = string(po_array);%把po_array变成一个仅有年、月的string array

%下面的代码块可以获得绝大部分生产批次在特定时间的批次号。
startrow = find(po_array==monthyear, 1, 'first');

endrow = find(po_array==monthyear, 1, 'last');
```

```
T = relationship(startrow:endrow,:);
%删除T中的有关X、Y工序（这两个工序不做研究）的批次
po_array = T.Productionorder;
po_array = char(po_array);
po_array = po_array(:,9);%把po_array变成一个仅有一个字母的char array
rows = [find(po_array=='X');
        find(po_array=='Y')];
T(rows,:) = [];%删除操作

number_of_orders = length(unique(T.Workorder));%计算总共有几个工单，然后初始化xy_stats。

%% 初始化table：xy_stats。初始值都为NaN。
vector = zeros(number_of_orders/4,1);
for i = 1:number_of_orders/4
    vector(i) = NaN;
end
expression = "vector,";
for i = 2:length(variables)-1
    expression = expression + "vector,";
end
expression = expression + "vector";
eval("xy_stats = table(" + expression + ");");%初始化变量

%其他的列
xy_stats.Productionorder = categorical(vector);
xy_stats.Time = string(vector);
xy_stats.Fillsi = vector;
xy_stats.Longsi = vector;
xy_stats.Midsi = vector;
xy_stats.Shortsi = vector;
xy_stats.Brokensi = vector;
xy_stats.Completesi = vector;
xy_stats.Bigleaf = vector;
xy_stats.Bmleaf = vector;
xy_stats.Midleaf = vector;
xy_stats.Brokenleaf = vector;

%% 将本批次号的信息提取到相应的xy_stats中去
row = 1;
for i = 1:height(T)-3
    productionorder = T.Productionorder(i);
    %如果满足一个批次号对应4个工单，继续寻找。
    if T.Productionorder(i) == T.Productionorder(i+1) && T.Productionorder(i) ==
T.Productionorder(i+2) && T.Productionorder(i) == T.Productionorder(i+3)
        workorder = {categorical(T.Workorder(i))
            categorical(T.Workorder(i+1))
            categorical(T.Workorder(i+2))
            categorical(T.Workorder(i+3))};

        %提取出来相应的数据到smalltable中，便于处理.
        %主要提取均值，变量名。
        startrow = find(raw_data.Workorder == workorder{1}, 1, 'first');

        endrow = find(raw_data.Workorder == workorder{1}, 1, 'last');
```

```
        smalltable1 = raw_data(startrow:endrow, [2, 6, 10, 12]);
        startrow = find(raw_data.Workorder == workorder{2}, 1, 'first');
        endrow = find(raw_data.Workorder == workorder{2}, 1, 'last');
        smalltable2 = raw_data(startrow:endrow, [2, 6, 10, 12]);
        startrow = find(raw_data.Workorder == workorder{3}, 1, 'first');
        endrow = find(raw_data.Workorder == workorder{3}, 1, 'last');
        smalltable3 = raw_data(startrow:endrow, [2, 6, 10, 12]);
        startrow = find(raw_data.Workorder == workorder{4}, 1, 'first');
        endrow = find(raw_data.Workorder == workorder{4}, 1, 'last');
        smalltable4 = raw_data(startrow:endrow, [2, 6, 10, 12]);
        smalltable = [smalltable1;
                      smalltable2;
                      smalltable3;
                      smalltable4];
        if(isempty(smalltable))
            continue;
        end
        for j = 1:height(smalltable)
            k = find(variables==smalltable.Varname(j));
            xy_stats{row, k} = smalltable.Average(j);
        end
        xy_stats.Productionorder(row) = productionorder;
        xy_stats.Time(row) = T.Time(i);
        rows = find(raw_fillsi.Productionorder == productionorder);
        value = mean(raw_fillsi.Value(rows));
        xy_stats.Fillsi(row) = value;
        rows = find(raw_longsi.Productionorder == productionorder);
        value = mean(raw_longsi.Value(rows));
        xy_stats.Longsi(row) = value;
        rows = find(raw_midsi.Productionorder == productionorder);
        value = mean(raw_midsi.Value(rows));
        xy_stats.Midsi(row) = value;
        rows = find(raw_shortsi.Productionorder == productionorder);
        value = mean(raw_shortsi.Value(rows));
        xy_stats.Shortsi(row) = value;
        rows = find(raw_brokensi.Productionorder == productionorder);
        value = mean(raw_brokensi.Value(rows));
        xy_stats.Brokensi(row) = value;
        rows = find(raw_completesi.Productionorder == productionorder);
        value = mean(raw_completesi.Value(rows));
        xy_stats.Completesi(row) = value;
        rows = find(raw_bigleaf.Productionorder == productionorder);
        value = mean(raw_bigleaf.Value(rows));
        xy_stats.Bigleaf(row) = value;
        rows = find(raw_bmleaf.Productionorder == productionorder);
        value = mean(raw_bmleaf.Value(rows));
        xy_stats.Bmleaf(row) = value;
        rows = find(raw_midleaf.Productionorder == productionorder);
        value = mean(raw_midleaf.Value(rows));
        xy_stats.Midleaf(row) = value;
        rows = find(raw_brokenleaf.Productionorder == productionorder);
        value = mean(raw_brokenleaf.Value(rows));

        xy_stats.Brokenleaf(row) = value;
```

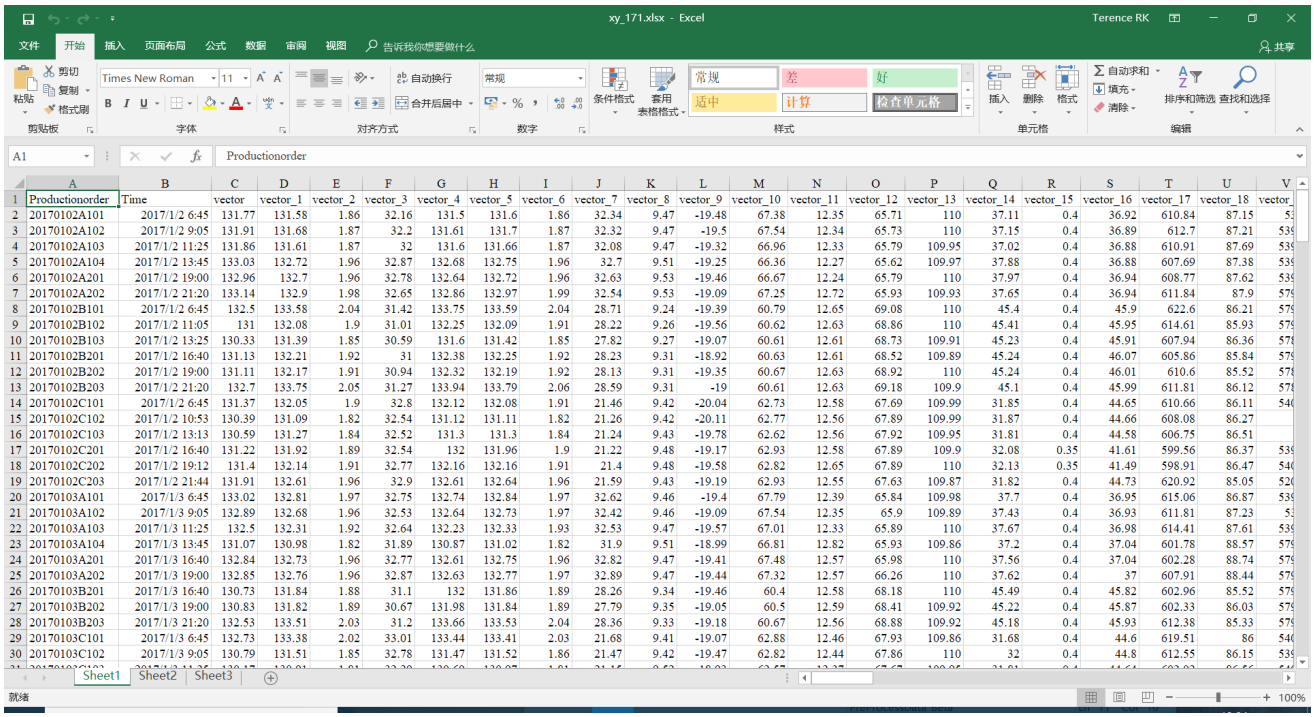
```
        row = row + 1;
    end
end

%% 对xy_stats进行最后的操作，polishing
xy_stats = xy_stats(:, [length(variables)+1, length(variables)+2, 1:length(variables),
length(variables)+3:length(variables)+12]);
%% 保存得到的table到文件
```

试验了几个月的数据，发现函数执行正常。

如果不出意外的话，本函数将适用于所有的月份的初始数据的预处理，获得数据只需要运行一下本函数即可。

下面是我们获得提取出来的matlab table变量导入到excel文件的截图：

`xy_171.xlsx`



由于matlab的table数据不支持中文格式的列名称，所以另附一个excel来标识各个变量的实际含义（顺序相同），部分截图如下：

`variables_171.xlsx`

1	1区冷凝水温度（℃）
2	1区筒壁温度（℃）
3	1区蒸汽压力（bar）
4	1区蒸汽薄膜阀开度（%）
5	2区冷凝水温度（℃）
6	2区筒壁温度（℃）
7	2区蒸汽压力（bar）
8	2区蒸汽薄膜阀开度（%）
9	KLD总蒸汽压力（bar）
10	KLD排潮负压（μbar）
11	KLD排潮风门开度（%）
12	KLD新烘后新水分（%）
13	KLD烘后温度（℃）
14	KLD热风温度（℃）
15	KLD热风蒸汽薄膜阀开度（%）
16	KLD热风风速（m/s）
17	KLD热风风门开度（%）
18	KLD除水量（l/h）
19	SIROX后温度（℃）
20	SIROX新蒸汽质量流量（Kg/h）
21	SIROX蒸汽体积流量（m3/h）
22	SIROX蒸汽温度（℃）
23	SIROX蒸汽薄膜阀开度（%）
24	SIROX蒸汽质量流量（kg/h）
25	SIROX阀前蒸汽压力（bar）
26	Sirox蒸汽体积流量（m3/h）
27	Sirox蒸汽质量流量（kg/h）
28	一次1#罐料液温度（℃）
29	一次2#罐料液温度（℃）
30	一次出口水分（%）

之后，就方便使用python对数据进行进一步的处理了。

参考文档：

1. MATLAB official document - table2cell
2. MATLAB新的统计数据类型Table
3. MATLAB official documentation - table
4. MATLAB table常用操作

下周计划

本周末考试周已经来临，考虑到下周可能需要较多的时间进行复习备考，所以下周可能没有过多的时间来实行项目，因此暂时不定计划，希望老师批准。可能会有项目的执行，具体请参考下周的周报。