Saving Time and Analysis with JMP Scripting

Byron Wingerd, JMP

June 30, 2017

Tracking and Trending Manufacturing Metrics

Part 2

Using JSL to automate repetitive monitoring

Tips and tricks for working with control chart scripts

Building the Weekly Metrics Report

Parts List

1. Historical Process Data
2. Current Data
3. Control Limits Table
4. Specification Limits Table
5. Some JMP Reports

PLUS: About an hour to cobble it all together*

*Includes necessary stop for fresh coffee

This is a Technically Advanced Example

Every effort was made to keep the example simple

- Data manipulation example
- Example of loading control limits to a table
- Example of loading spec limits to a table
- Example of editing an axis box settings
- Example of saving a report as an image

Most people won’t use all these parts together, but these are all good things to know how to do.

JSL Essentials

- Help>Books>Scripting Guide
- Help>Scripting Index
- This one link: https://community.jmp.com/t5/JMPer-Cable/Do-you-want-to-learn-the-JMP-Scripting-Language/ba-p/37438
- Script and log windows

- Syntax
  - Only spelling matters, and punctuation
  - Its function based, need to glue between with a semicolon

Workflow

- Data Preparation
  - Open Historic data and Current data, Concatenate the Current data
  - Add Control Limits
  - Add Specification Limits
- Generate reports
- Make Reporting Media
  - Update PowerPoint slides
  - Save reports as images

Opening the Data Files

```
dt_old      =open("Historic Metrics Data.jmp");
dt_new      =open("Current Metrics Data.jmp");
dt_cl       =open("Control Limits.jmp");
dt_specs    =open("Specification Limits.jmp");
```

- Note variable name convention: Intentional vs. random or creative
  - Variable names have very few constraints
- File names are missing the path in this example
  - Data files are in the same folder as the script.
  - Q:\User\Data\Project\Subfolder\longlongname\obscurenumber\data\JMPFILE.jmp would work too

- `dt_update = dt_old<<concatenate(dt_new, Append to first table, create source column);`

Add New Data to the Historic Data

- Manual: Tables tab, Concatenate
  - Do this once and edit the Source script that is written to the resulting table
- Check the Scripting Index for arguments

- dt_update = dt_old<<concatenate(dt_new, Append to first table, create source column);

Load Control and Specification Limits

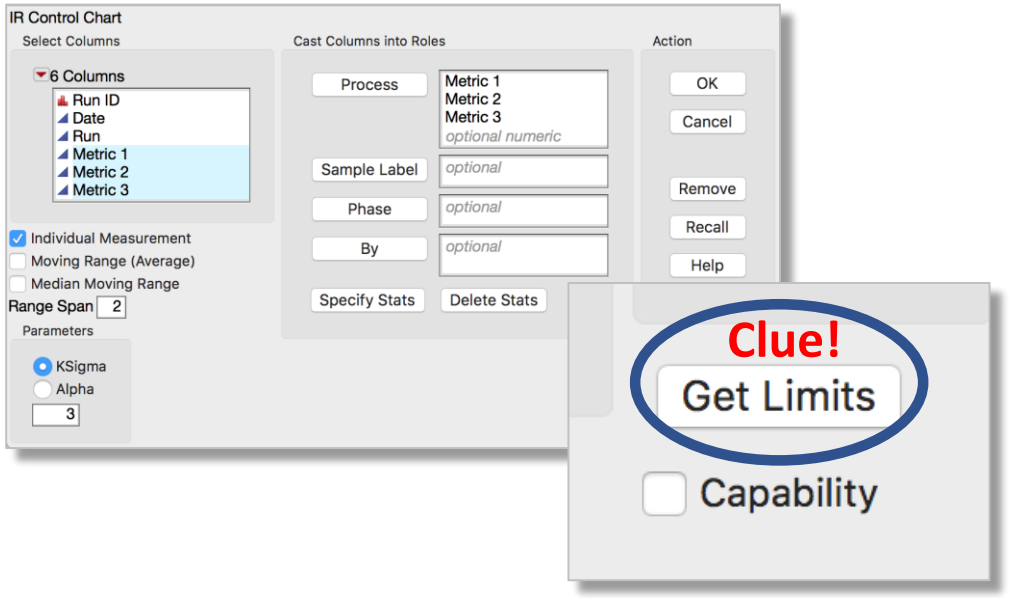
- DO NOT!
  - Go shopping for obscure ways to do this
  - Try to scrape and append column properties
  - Spend hours and hours writing complex custom code

HARD WAY

- DO
  - Script the platforms.
  - Check the scripting index
  - Look for clues in the platforms

Easy WAY

Loading Control Limits

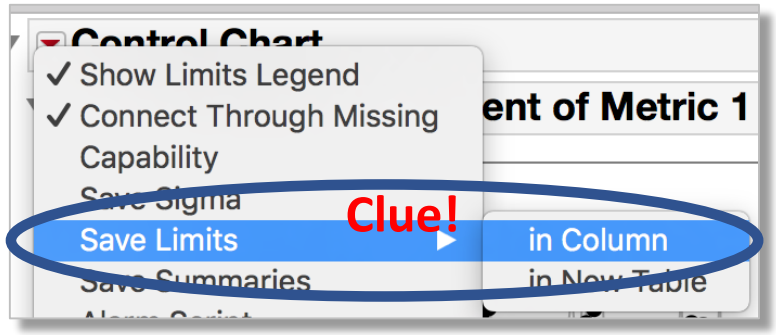


Search Scripting Index for “Get Limits”

```
Control Chart(
Chart Col( :Metric Col, Individual Measurement),
Get Limits(“Path/Control_Limits_file.jmp")
);
```

Save Control Limits to Column

Search Scripting Index for “Save Limits”

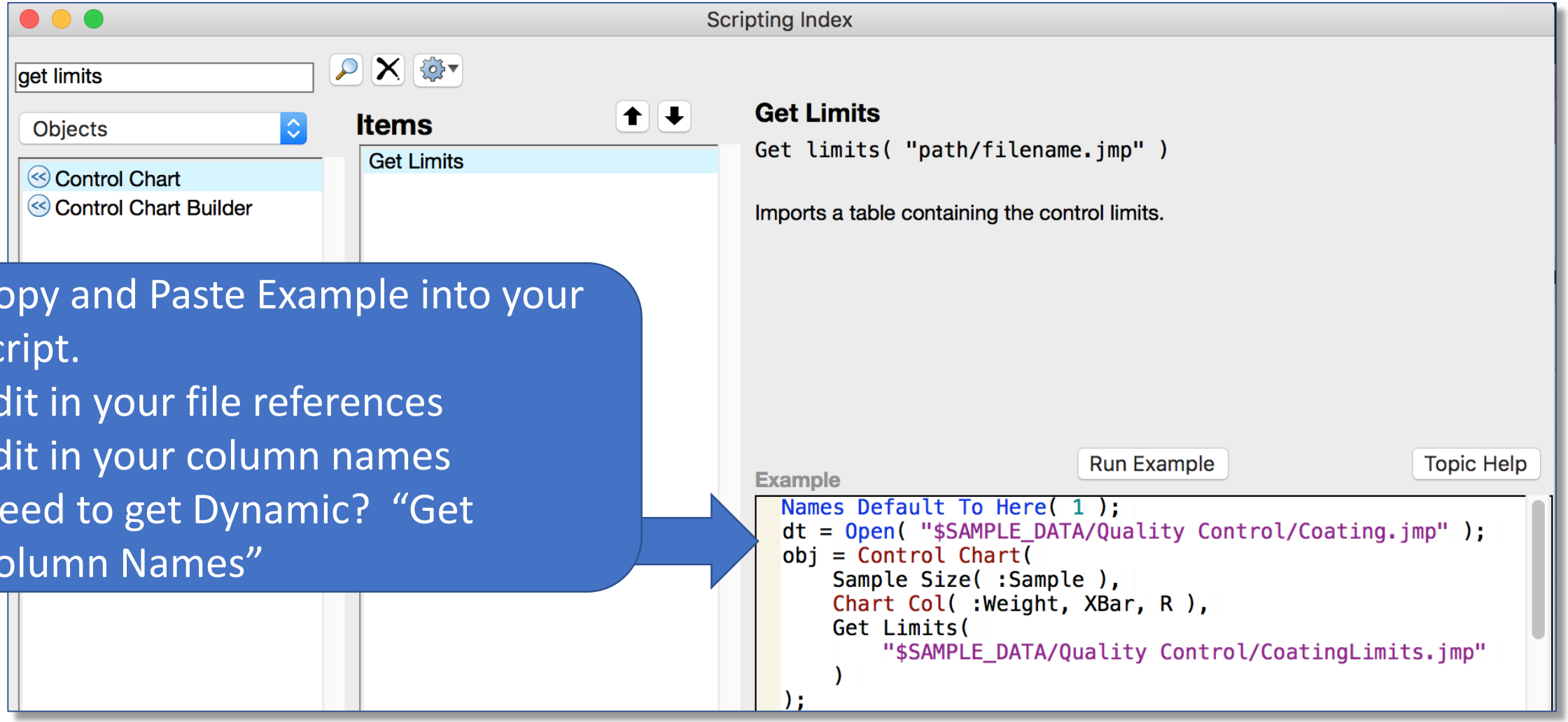



```
Obj=Control Chart(
Chart Col( :Metric Col, Individual Measurement),
Get Limits(“Path/Control_Limits_file.jmp")
);
obj << in Column;
```

Don’t write complex scripts.
Script the platform!

Load Control Limits

Search Scripting Index for “Get Limits”



- Copy and Paste Example into your script.
- Edit in your file references
- Edit in your column names
- Need to get Dynamic? “Get Column Names”

Load Control Limits

```
obj=dt_update<<Control Chart(
        Chart Col( :Metric 1, Individual Measurement ),
        Chart Col( :Metric 2, Individual Measurement ),
        Chart Col( :Metric 3, Individual Measurement ),
        get limits("Control Limits.jmp"));
obj<<in Column;
obj<<close window;
```

- Note: "**obj**" is a handle for the report.
  - Send "in Column" to it
  - Send "close" to it …because I don't want to use this report, just the platform.

Loading Specification Limits

- DO NOT!
  - Go shopping for obscure ways to do this
  - Try to scrape and append column properties
  - Spend hours and hours writing complex custom code

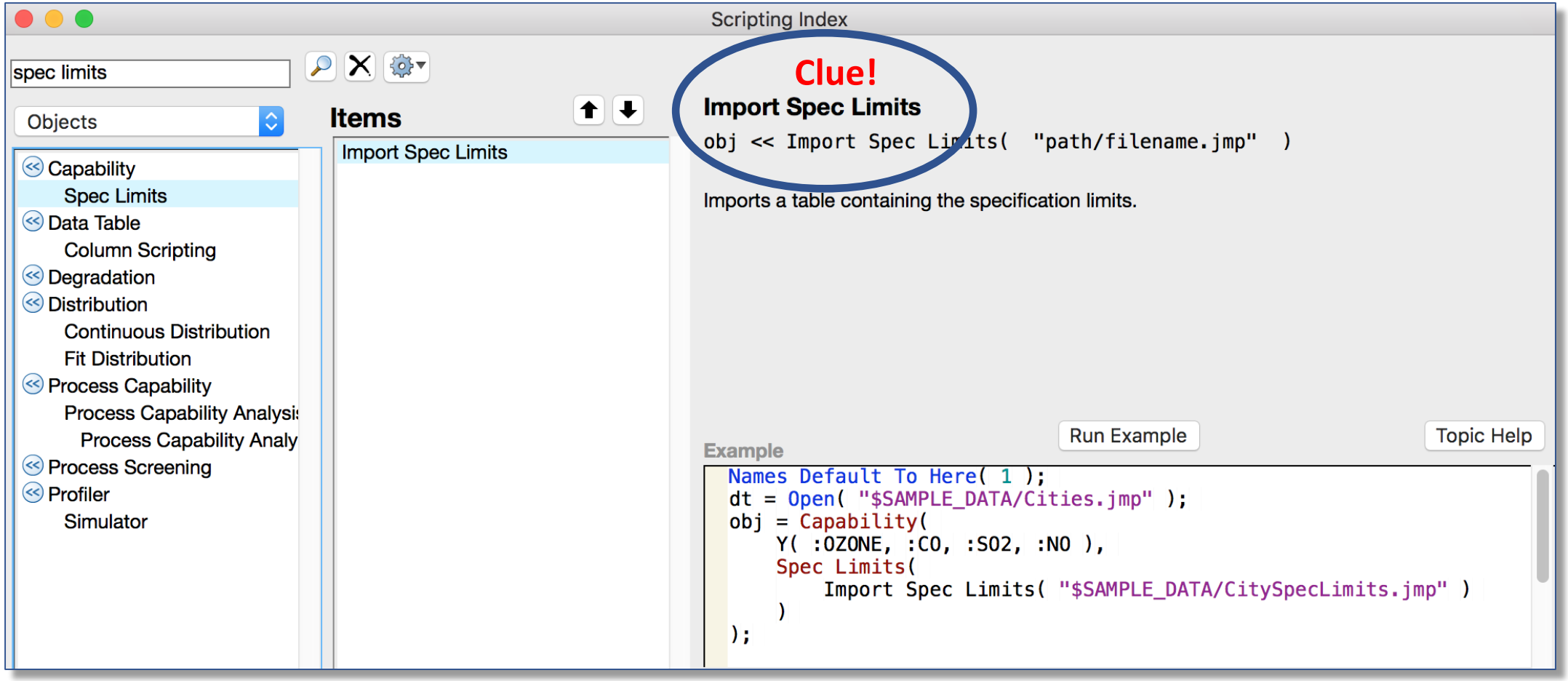
HARD WAY

- DO
  - Script the platforms.
  - Check the scripting index
  - Look for clues in the platforms

Easy WAY

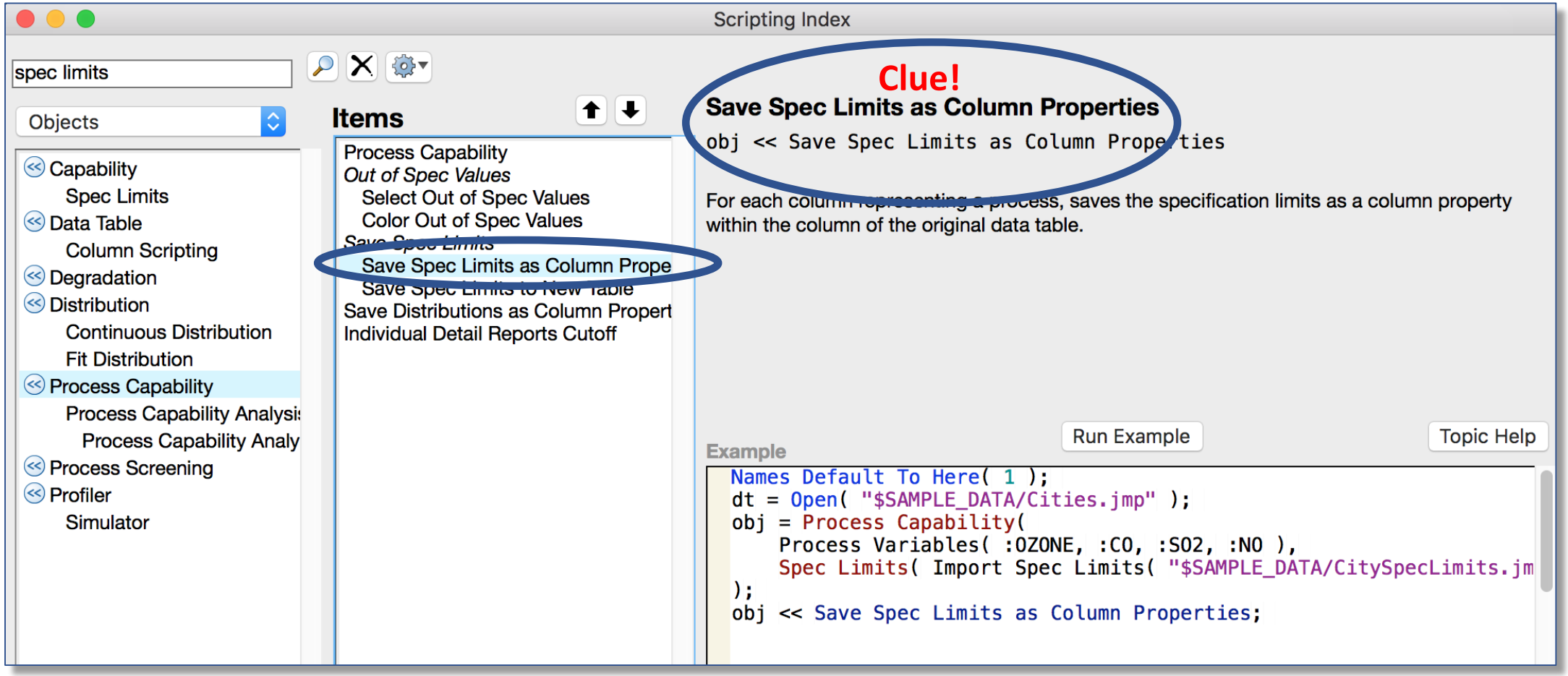
Load Specification Limits

Search Scripting Index for “Spec Limits”



Load Specification Limits

Save spec limits back to the table



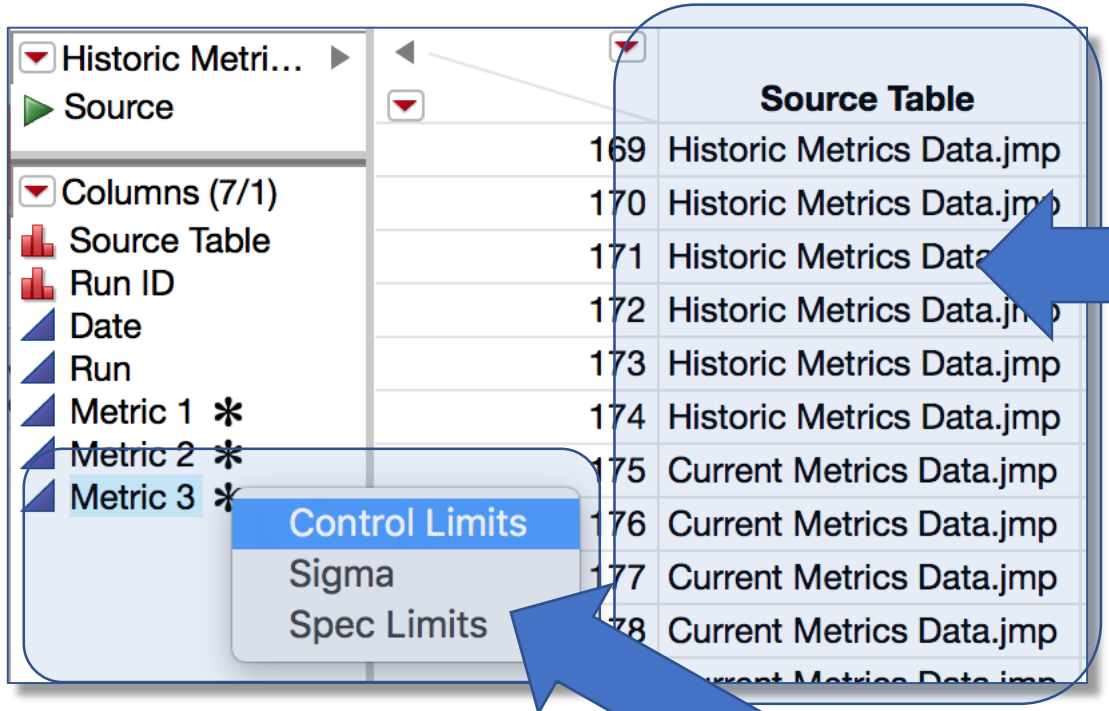
Load Specification Limits

```
obj = dt_update<<Capability(
        Y( :Metric 1, :Metric 2, :Metric 3 ),
        Spec Limits(Import Spec Limits( "Specification Limits.jmp" )));
obj << Save Spec Limits as Column Properties;
obj<<close window;
```

- Note: "**obj**" is a handle for the report.
  - Send "in Column" to it
  - Send "close" to it …because I don't want to use this report, just the platform.

Also, This step might take a few seconds with a long list of column names

Progress Check: Look at the data table



Old and New data in the same table with a column to identify the source

Control Limits and Spec Limits loaded to the Column Properties

Generate Reports

- Manually run the platform to get your report/graph
  - Manually format the graph to make it look “right”
- Get the script
- Edit the script
- String scripts together

Working with Control Chart Scripts

```
fig1 = dt_update <<
Control Chart Builder(
    Size( 533, 449 ), Show Control Panel( 0 ),
    Variables( Subgroup( :Date ), Y( :Metric 1 ) ),
    Chart( Position( 1 ) ), Chart( Position( 2 ) ),
    SendToReport(
        Dispatch({},"Date",ScaleBox,{Min( Col Maximum( :Date ) - In Days( 30 ) ), Max( Col
        Maximum( :Date ) ), Interval( "Week" ),Inc( 1 ), Minor Ticks( 6 ), Label Row( Label
        Orientation( "Angled" ) )}))
);
```

Edit 1: Be specific, send the control chart to the right data table

Working with Control Chart Scripts

```
fig1 = dt_update <<
Control Chart Builder(
    Size( 533, 449 ), Show Control Panel( 0 ),
    Variables( Subgroup( :Date ), Y( :Metric 1 ) ),
    Chart( Position( 1 ) ), Chart( Position( 2 ) ),
    SendToReport(
        Dispatch({},"Date",ScaleBox,{Min( Col Maximum( :Date ) - In Days( 30 ) ), Max( Col
        Maximum( :Date ) ), Interval( "Week" ),Inc( 1 ), Minor Ticks( 6 ), Label Row( Label
        Orientation( "Angled" ) )}))
);
```

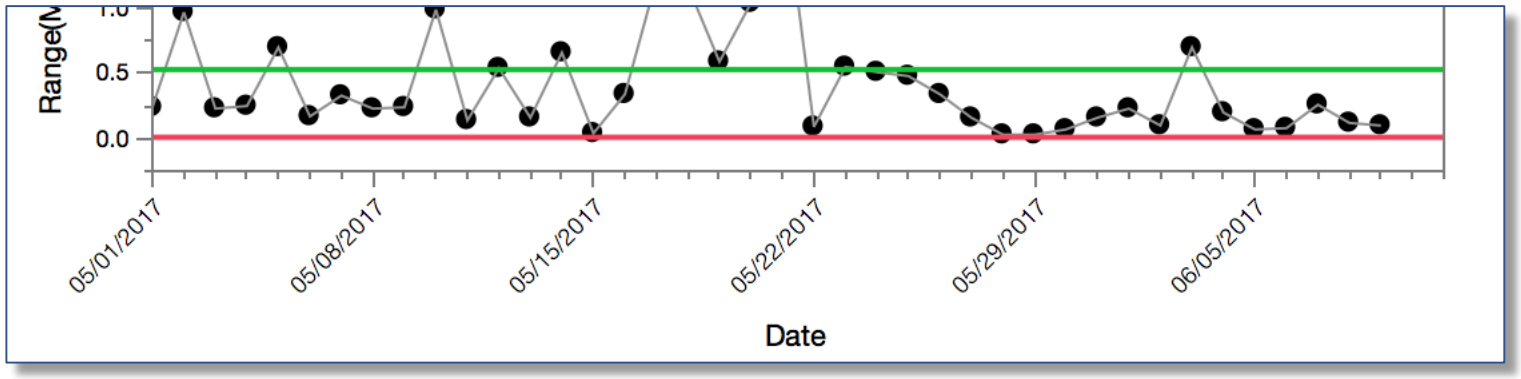
- Edit 2: Unpack and edit the “SendToReport”

Note: If your are following along, this is missing in the example

```
SendToReport(
	Dispatch(
		{},
		"Metric 1",
		ScaleBox,
		{Add Ref Line( 13, "Solid", "Blue", "LSL", 1 ),
		Add Ref Line( 17, "Solid", "Blue", "USL", 1 )}
		)
	)
```

Tip: Just delete this part, it happens automatically anyway

Formatting the Axis Box



Byron's Preference for Axis Boxes in Control Charts (today's version)

1. I like the date angled
2. I like the dates at one week intervals
3. I like 6 tick marks between the weekly intervals
4. I really only want to see the current month or so of data

Do it manually and then copy the script

Formatting the Axis Box

- Argument sent to the text edit the Axis box

```
Dispatch(
    {},
    "Date",
    ScaleBox,
    {Min( 3576441600 ), Max( 3579984000 ), Interval( "Week" ), Inc( 1 ),
    Minor Ticks( 6 ), Label Row( Label Orientation( "Angled" ) )}
),
```

Edit: I need relative Date References not specific ones

Replace “Min(n)“ and “Max(n) with this:

- Min(Col Maximum(:Date) - In Days(30))
- Max(Col Maximum(:Date)

Now the last 30 days are displayed on the chart (days not rows)

Formatting the Axis Box (alternate approach)

- Argument the edit the Axis box

```
Dispatch(
        {},
        "Date",
        ScaleBox,
        {Min( 3576441600 ), Max( 3579984000 ), Interval( "Week" ), Inc( 1 ),
        Minor Ticks( 6 ), Label Row( Label Orientation( "Angled" ) )}
),
```

In this case, the label column is Numeric, Continuous, Date Format. So the Min and Max are in units of the column, which happen to be seconds.

If there is no label, or a nominal label, this will work to show the last 50

{min(nrows()-50, max{nrows()}

nrows() refers to the number or rows in the table.

Making Reporting Media

fig1 = dt_update << Control Chart Builder(.....

1. Be specific, send the report script to the right data table
2. Get a handle
3. Do stuff with the handle

- fig1 << save picture("figure1.jpg" ,jpg); Make a jpg image
- fig1 << save picture("figure1.emf" ,emf); Make an enhance metafile
- fig1 << save picture("figure1.pdf" ,pdf); Make a PDF

In this example I need graphics for the next step

Making Reporting Media

In PowerPoint

1. Insert Picture from File
2. Insert and Link Option

- Save and Close PowerPoint File
- Run JSL to make figures and save new images.
- Open PowerPoint File

Making Reporting Media Notes

- The simple script to create the control chart automatically produces the capability report because we already saved the spec limits to the column properties.
- JMP does a lot of complex things automatically, take advantage of them

Weekly Metrics Update Example

- Make a separate PowerPoint file
- Use the insert, and link function in PowerPoint
- Six Figures
  - Three Control Chart Slides with Capability Analysis
    - One figure per page
  - One slide with some other graphs
    - Two figures on one page

Notes on Reproducing this Presentation

First, run this script: Data Script.jsl
This script will run: Reports Script.jsl

The script generates some figures that will end up in the same directory as the Data Script.jsl file.

- Ideally when you run the PowerPoint deck, you will be able to click on the bold blue triangle to launch a second PowerPoint deck. Depending on your version of MS and administrative control this might be broken.
- Ideally the second PowerPoint file would have figures that automatically update. Unfortunately the links break when I package the file.
- If you want to see the auto update feature work, you have to re-insert and link the images to the appropriate pages.