

Seven Ways to CYA with JMP *reloaded*

D. J. Garbutt¹

¹BIOP AG
Basel, Switzerland

PhUSE, 2011 Brighton

Introduction

JMP is a very capable tool for statistical analysis and data exploration. In this presentation I claim it also makes an indispensable tool for the statistical programmer. To support this I present (roughly) seven scenarios where JMP can help with programming and debugging typical reporting programs and do a distinctly better job than common alternatives.



Outline



- 1 CYA and JMP
 - Clinical programming vs. real programming
 - JMP - history, features
- 2 SAS vs JMP in operation
- 3 Scenarios where JMP can help
 - Check consistency and feasibility
 - Check data structure
 - Errors created by programs

Some words for the acronym

If you have an interesting discovery with your data then it is most likely a data error, a measurement miscalibration or an experimental artefact. Dave, go back and check your data.

Cover Youthful Arrogance

Cover Yourself against future Aggravation

Polonius! Cover Your Arras (Hamlet)



Clinical Programming



The Basic Problem is time vs. completeness

- and that completeness means much programming around edge case in the data
- it follows that knowing your data well can substitute for complex overly general programming
- YAGNI ¹
- build (only) what you need
- learn specifics, standardise and reduce workload,
- never tackle general things and exploit the specifics in your data

¹a slogan from XP that stands for You Ain't Gonna Need It

Strategies



All these strategies have problems

Example

exploiting each issue's specificities

become fragile to changes or extensions in data or protocol

- costly reprogramming

code can be long and hard to read

- especially if you do not know all the details of the data

high specificity prevents re-use

- even by the author

the programming culture becomes blind to the similarities between projects and does see developing general tools as worthwhile

Strategies



All these strategies have problems

Example

too strong focus on commonalities

devalues rare but essential needs as too complicated

comes to regard extensions as 'scope creep' and to be avoided

leading to *cross-over studies will not be done because we can't report them*

paralysis by analysis leading to late delivery of code

History of JMP, its place now

designed as **interactive** program for data analysis in the 80's

Interactive live graphics, brushing, sub-setting, labeling with colour, size, shape

Interactive reshaping: joining, transposing, splitting, stacking and filtering of data

Now grown up and can read SAS datasets, send code to SAS and run stored processes



A question of style

- pseudo-interactive vs interactive
- thinking – pause – doing – thinking
- similar yes, but the cycle time makes it a different world



Assessing our tools

Good tool effect the time / effort we are willing to expend on a task is remarkably constant
with good tools therefore we can get more done.

- What does good mean?

operate at a higher level of abstraction

unitary pieces that can be fitted together in unexpected ways

can build diagnostics that show so unconsidered errors can still be found

clear feedback for comprehension
(graphics not tables)

exactly what you need already built in already

lithe and lean only features you need



Freq-all in JMP



In SAS

- 1 Get idea
- 2 Google
- 3 Find macro from Ron Fehd
- 4 Download
- 5 Paste into SAS window
run on my data
- 6 Print/ browse output

Total Time: 1hr

With JMP

- 1 Get idea
- 2 ctrl-A - select all variables
- 3 Analyze | distributions
- 4 see problems and find
which cases immediately

Total Time: 2 mins.

58 extra minutes for data exploration → output window with histograms etc., which can still be explored

lab data

Example

lab data

how to restructure it

how to view it



Check consistency and feasibility

Way out values

- add boxplots to distributions and QQ plots
- plot boxplots per centre, group per variable
- select any individual points - locate them in data sets
- make a plot matrix for a better MV look at the data



Check consistency and feasibility

Getting lists of discrete values

- get the lists
- Check vs data definitions



Check consistency and feasibility

Heat map for values

- use with values that can be sorted
e.g. by obs
- Not suitable if looking for outliers - the colour scale is adjusted to prevent it being dominated by them



Check data structure

Page Incidence plot per variable

- useful if you have page source variables so you can monitor CRF entry
- use the long thin DS and count data points per page



Check data structure

Percent missingness, patterns of missing values

- For checking page delivery and
- data structures of analysis datasets vs originals



Check data structure

Analyse Counts per patient, or patient visit

- similar use to %missing
- check also for extra visits, duplicated data (= incorrect pno or visit number)



Dataset meta data

- count labels per dataset
- compare lengths, formats attached, etc
- find variables in each dataset
 - using row marking
 - 1 select the VAR name, right click 'select similar cells'
 - 2 select colour rows from row dropdown
 - 3 select next cell (varname)

Check data structure

Compare datasets, count labels per dataset

- still with variable metadata



Check data structure

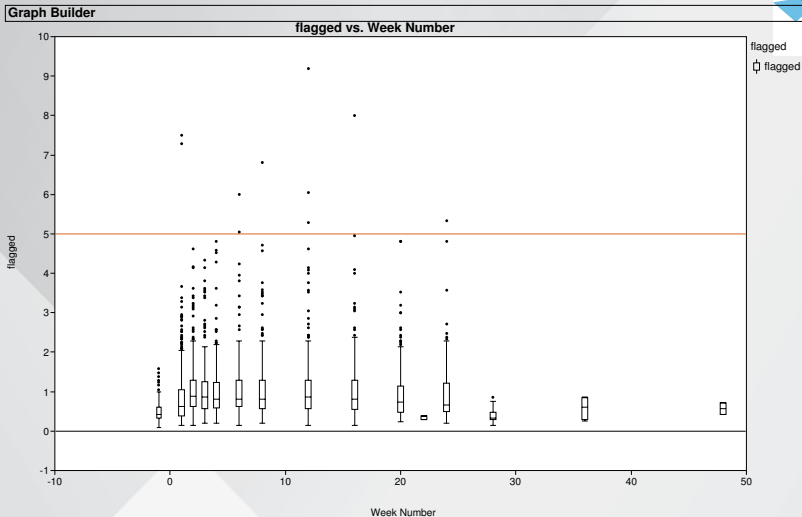
Strange patterns in Lab data

- define strange?
- bimodal
 - fit normal mixture eg height / weight
 - lab values with conversions
- plot lab ranges per centre to look for odd patterns



Check data structure

Box plots



Bilirubin values as tree map



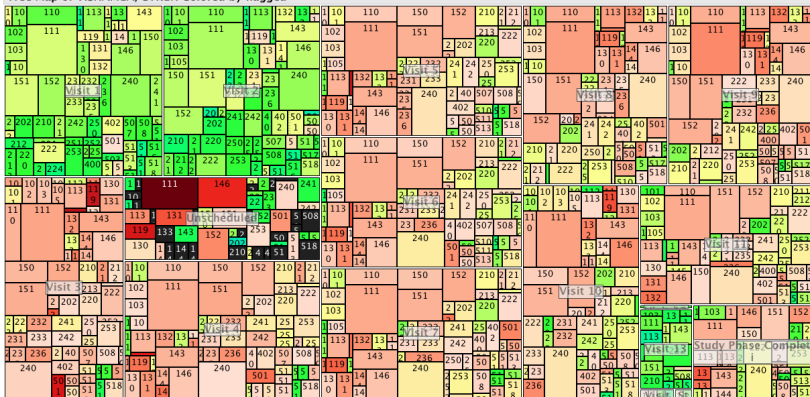
- Tree maps typically used for categorical data of counts (AEs,...)
 - space filling rectangles with proportional to count, or size
 - colouring by another factor
- In JMP they can be other things too
 - example bilirubin / upper limit, grouped by centre within visit name
 - reveals a lot - but in a very compact way.
 - Visit fill space from top left, size & colour by average bilirubin value.
- The values are converted to a colour scale automatically
- Or you can allocate them to a variable
 - Various scales can be chosen

Check data structure

Tree map of bilirubin



Tree Map of VISNAM1A, CTR1N Colored by flagged



Check data structure

Setting a colour scale

Biop

Lower Limit	Upper Limit	Lab Param	PARONR 1N	Units	Preferred Unit	sampledate	flagged
							1.476190476
							4
							1.333333333
							1
							0.619047619
							3.428571428
							3.428571428
							4.285714285
							4.238095238
							1
							6.809523809
							9.190476190
							1.190476190
							8.619047619
							11.80952381
							7.476190476
							0.857142857
							2.380952381
							1.904761904

'flagged' in Table 'LRS-TOTBil-thin'

Column Name:

☐ Lock

Data Type:

Modeling Type:

Format: Width:

☐ Use thousands separator (,)

Column Properties ▾

Formula
Control Limits
Color Gradient
optional item

Remove

Color Gradient

☒ Color Cell by Value

minimum:

maximum:

center:

OK
Cancel
Apply
Help

Check data structure

Looking for extra high values

– bubble plot

- Bilirubin values over time create with graph builder



Play the movie
open ff and play movie

[Check data structure](#)

Script for bubble plot



```
New Script(  
  "Bubble Plot 2",  
  Bubble Plot(  
    X( :VISNAM1A ), Y( :flagged ), Time( :VIS1N ),  
    Coloring( :flagged ), ID( :SID1A ), Speed( 4.32 ),  
    Bubble Size( 25.25 ), Time Index( 2.9 ),  
    Trail Bubbles( 1 ), Trail Lines( 1 ),  
    All Labels( 0 ), No Labels( 0 ),  
    Title Position( 8.16, 9 ),  
    SendToReport(  
      Dispatch(  
        {},  
        "2",  
        ScaleBox,  
        {Min( 0 ), Max( 10 ), Inc( 1 ), Minor Ticks( 0 ),  
        Add Ref Line( 5, Solid, {214, 103, 36} )})  
      ) ) )
```


Errors created by programs

Check for relational structures

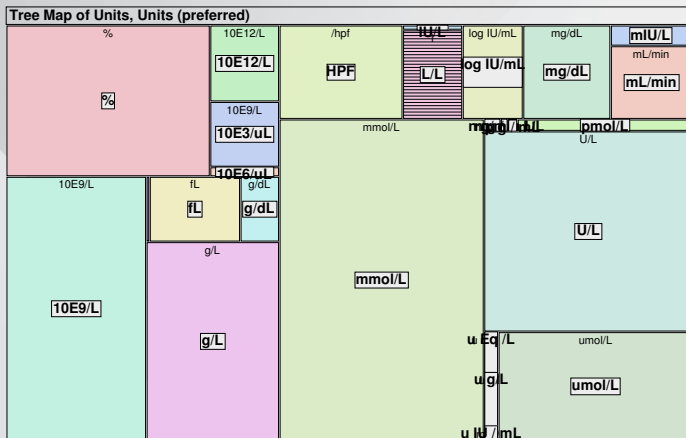
- given a set of tables find
 - 1 list of variables in common between each pair
 - 2 for those pairs table occurrence patterns to see which can be merged
 - 3 list of all shared variables
 - 4 any shared variables not same type/ format in every table
 - 5 draw an ER diagram of the tables and their relationships
 - 6 make and check inferences about data level (patient, visit, time)



Errors created by programs

Incorrect Lab units

- how to find and quickly check for them
- lbs vs. Kg ; °F vs. °C ; mg /ml



Errors created by programs

Normal ranges that aren't

- plot ranges vs centre; age, sex,...
- by centre within Lab parameter



Summary I

- Statistical programming is unlike other types of programming because it is tied to the actual data that will be analysed.
- programming is a craft
- and debugging is being a scientist (theory, experiment, refutation...)
- Because JMP is designed for analysing and viewing data it makes an excellent tool for a *data programmer* and a *data scientist*
- JMP is a *data browser on steroids*
it is therefore also useful for statisticians and data managers, anyone working daily with more data than fits on a sheet of paper



Summary II

- When reporting is driven by completely with metadata we will instead be debugging and working with metadata.



An interactive tool will be even more useful...

Questions

?

For Further Reading I

JMP guides and books

Tutorials on web site and free PDF books from help menu

