SPSS Syntax Reference Sheet

By Daniel Whitten - **Questions**? d.j.whitten@fsw.leidenuniv.nl

For more detail on any command, in SPSS refer to Help > Command Syntax Reference (also found here). Most examples can be tested on this dataset (some examples require testing previous examples).

Remember:

- Open the Syntax Editor in SPSS with New > Syntax
- Save your syntax document so you can refer back and repeat your steps automatically
- Highlight the commands you want to run and press the green "play" button or ctrl+r to run
- Commands always begin with a `COMMAND` word and end with a `.` [period]
- Commands that change variables usually need to be followed by `EXECUTE.` to run
- Commands are not case-sensitive, but variable names are
- Only use quotation marks for labels and descriptions, not for numerical data or variable names

Colour Coding:

- `RED` = **Necessary syntax:** Keep these parts the same as in the example, or you could get an error
- `DARK BLUE` = **Existing variable:** Find the exact variable name in your dataset, beware of typos!
- `LIGHT BLUE` = **Existing value:** Find the appropriate values in your dataset for the given variable
- `DARK GREEN` = **New variable:** You are making this variable, it needs a name with no spaces
- `LIGHT GREEN` = **New value:** you are choosing this value, ensure it's a number and not in quotes
- `OLIVE GREEN` = **New label:** Anything goes, be descriptive and use spaces, ensure it is in quotes
- `BLACK` = **Numeric Value:** This is a specific value you must provide from elsewhere

Create New Variables

Recode Variables

Note: There are special keywords for this command, they are:

- `INTO` - the recoded values go into a new variable
- `THRU` - indicates a range of values
- `LO` - the lowest value in the variable
- `HI` - the highest value in the variable
- `ELSE` - every value not already mentioned (always put this last if you use it!)
- `COPY` - replicate the specified values exactly in the new variable
- `MISSING` - all missing values in the old variable
- `SYSMIS` - missing values in the new variable

Some additional rules:

- Recoding values must have an `=` between the old and new values
- Values are almost always numerical, use words for labelling (this means no quotation marks in `RECODE` commands)
- Variable names can never have spaces, use `_` to represent spaces if needed
- Each `RECODE` command line needs to end with a `.` [period]
- Consider declaring `MISSING VALUES` [here] before recoding a variable
- With no `ELSE` keyword, all values not explicitly mentioned will be recoded as missing values
- With an `ELSE` keyword, use `(MISSING = SYSMIS)` before the `ELSE` to avoid coding your missing values as real values

Basic

Format

```
RECODE old_variable (old_value = new_value) INTO new_variable.

EXECUTE.
```

Example

```
RECODE insurance (1 = 0)(2 = 1) INTO insurance_recoded.
EXECUTE.
```

Range

Format

```
RECODE old_variable (old_val1 THRU old_val2 = new_val) INTO new_variable.
EXECUTE.
```

Example

```
RECODE age (LO THRU 17 = 1)(18 THRU 54 = 2)(55 THRU HI = 3) INTO age_ranges.
EXECUTE.
```

Change One Value (Keep All Else the Same)

Format

```
RECODE old_var (old_val = new_val)(MISSING = SYSMIS)(ELSE = COPY) INTO new_var.
EXECUTE.
```

Example

```
RECODE age (90 THRU HI = 90)(MISSING = SYSMIS)(ELSE = COPY) INTO age_capped.
EXECUTE.
```

Dummy Variables

Format

```
RECODE categorical_variable (value1 = 1)(MISSING = SYSMIS)(ELSE = 0) INTO dummy1.
RECODE categorical_variable (value2 = 1)(MISSING = SYSMIS)(ELSE = 0) INTO dummy2.
RECODE categorical_variable (value3 = 1)(MISSING = SYSMIS)(ELSE = 0) INTO dummy3.
EXECUTE.
```

Example

```
RECODE age_ranges (1 = 1)(MISSING = SYSMIS)(ELSE = 0) INTO child.
RECODE age_ranges (2 = 1)(MISSING = SYSMIS)(ELSE = 0) INTO adult.
RECODE age_ranges (3 = 1)(MISSING = SYSMIS)(ELSE = 0) INTO senior.
EXECUTE.
```

Declare Missing Values

Note: This command uses some of the same special keywords as `RECODE`, they are explained above: `LO`, `HI`, and `THRU`.

Basic

Format

```
MISSING VALUES existing_variable (missing_value).
```

Example

```
MISSING VALUES income (-5100).
```

Multiple Missing Values - One Variable

Format

```
MISSING VALUES existing_variable (lowest_missing_value THRU highest_missing_value).
```

Example

```
MISSING VALUES income (LO THRU -100).
```

Multiple Variables (Individual Variables)

Format

```
MISSING VALUES variable1 (missing_value) variable2 (missing_value).
```

Example

```
MISSING VALUES income (LO THRU -100) age (LO THRU -1).
```

Multiple Variables (Range of Variables)

Format

```
MISSING VALUES variable1 TO variable4 (missing_value).
```

Example

```
MISSING VALUES age TO income (LO THRU -1).
```

Transform (Compute) Values

Note: This command is highly customizable, and follows mathematical order of operations. See the Command Syntax Reference for a more complete list of possible computations, below are some common examples.

Note: surrounding any part of the calculation with `ABS()` converts to absolute values (`ABS(x)` = |x|)

Basic Transformation

Formats

```
COMPUTE new_variable = existing_variable + value.
EXECUTE.
```

```
COMPUTE new_variable = existing_variable - value.
```

```
EXECUTE.
```

```
COMPUTE new_variable = existing_variable * value.
EXECUTE.
```

```
COMPUTE new_variable = existing_variable / value.
EXECUTE.
```

Example

```
COMPUTE monthly_income = income / 12.
EXECUTE.
```

Multiple Variable Transformation

Formats

```
COMPUTE new_variable = variable1 + variable2.
EXECUTE.
```

```
COMPUTE new_variable = variable1 - variable2.
EXECUTE.
```

```
COMPUTE new_variable = ABS(variable1) * variable2.
EXECUTE.
```

```
COMPUTE new_variable = variable1 / variable2.
EXECUTE.
```

Example

```
COMPUTE income_by_year_of_age = income / age.
EXECUTE.
```

Create a Square Value

Formats

```
COMPUTE variable_square = questionable_variable * questionable_variable.
EXECUTE.
```

Example

```
COMPUTE income_square = income * income.
EXECUTE.
```

Create a Natural Logarithmic Value

Formats

```
COMPUTE variable_log = LN(variable).

EXECUTE.
```

Example

```
COMPUTE income_log = LN(income).

EXECUTE.
```

Create Interaction Variables

Note: See Field (2018) [Leiden][WorldCat] 11.3 for more information in moderation and interaction variables

Formats

```
COMPUTE interaction_variable = predictor_variable * moderator_variable.

EXECUTE.
```

Example

```
COMPUTE age_income_interaction = income * age.

EXECUTE.
```

Conditional Values

Note: This command is highly customizable See the Command Syntax Reference for a more complete list of possible computations, below are some common examples.

Note: creating new variables with just an `IF` command, without a preceding `COMPUTE` command to establish a baseline value (compare basic format against other examples) means that values that do not match the condition are coding as missing values.

Special Symbols:

- `a < b` — a is less than b
- `a <= b` — a is less than or equal to b
- `a > b` — a is greater than b
- `a >= b` — a is greater than or equal to b
- `a = b` — a is equal to b
- `a <> b` — a is not equal to b
- `ABS(a)` — the absolute value of a

Basic

Format

```
IF (condition) new_variable = value.

EXECUTE.
```

Example

```
IF (income <> 0) absolute_income = ABS(income).

EXECUTE.
```

Test for One Condition (No Missing Values)

Format

```
COMPUTE new_variable = 0.
IF (true_condition) new_variable = 1.
EXECUTE.
```

Example

```
COMPUTE below_poverty_line = 0.
IF (income < 2000) below_poverty_line = 1.
EXECUTE.
```

Test for Multiple Conditions (No Missing Values)

Formats

```
COMPUTE new_variable = 0.
IF (true_condition1 AND true_condition2) new_variable = 1.
EXECUTE.

COMPUTE new_variable = 0.
IF (true_condition1 OR true_condition2) new_variable = 1.
EXECUTE.
```

Examples

```
COMPUTE uninsured_poor = 0.
IF (income <= 2000 AND insurance_recoded = 0) uninsured_poverty = 1.
EXECUTE.

j

COMPUTE insured_poor_and_uninsured_rich = 0.
IF (
   (income <= 20000 AND insurance_recoded = 1)
  OR (income > 100000 AND insurance_recoded = 0)
  ) insured_poor_and_uninsured_rich = 1.
EXECUTE.
```

Test with Missing Values

Formats

```
COMPUTE new_variable = -9.
IF (true_condition) new_variable = 1.
IF (false_condition) new_variable = 0.
MISSING VALUES new_variable (-9).
EXECUTE.
```

```
COMPUTE new_variable = -9.
IF (true_condition1 AND true_condition2) new_variable = 1.
IF (false_condition1 OR false_condition2) new_variable = 0.
MISSING VALUES new_variable (-9).
EXECUTE.

COMPUTE new_variable = -9.
IF (true_condition1 OR true_condition2) new_variable = 1.
IF (false_condition1 OR false_condition2) new_variable = 0.
MISSING VALUES new_variable (-9).
EXECUTE.
```

Examples

```
COMPUTE below_poverty_line = -9.
IF (income < 2000) below_poverty_line = 1.
IF (income >= 2000) below_poverty_line = 0.
MISSING VALUES below_poverty_line (-9).
EXECUTE.

COMPUTE uninsured_poor = -9.
IF (income <= 2000 AND insurance_recoded = 0) uninsured_poverty = 1.
IF (income > 2000 OR insurance_recoded = 1) uninsured_poverty = 0.
MISSING VALUES uninsured_poverty (-9).
EXECUTE.

COMPUTE insured_or_rich = -9.
IF (income > 100000 OR insurance_recoded = 1) insured_or_rich = 1.
IF (income < 100000 OR insurance_recoded = 0) insured_or_rich = 0.
MISSING VALUES insured_or_rich (-9).
EXECUTE.

COMPUTE insured_poor_and_uninsured_rich = -9.
IF (
 (income <= 20000 AND insurance_recoded = 1)
 OR (income > 100000 AND insurance_recoded = 0)
 ) insured_poor_and_uninsured_rich = 1.
IF (
  (income <= 20000 AND insurance_recoded = 0)
  OR (income > 100000 AND insurance_recoded = 0)
  OR income > 20000 AND income <= 100000)
 ) insured_poor_and_uninsured_rich = 0.
MISSING VALUES insured_poor_and_uninsured_rich (-9).
```

```
EXECUTE.
```

Create Binary Dummy to Check a Statistic

Note: For example to check which cases have a standardized residual > 3.29 or Cook's distance > 1.

Format

```
COMPUTE check_variable = -9.

IF (ABS(saved_variable) > cutoff) check_variable = 1.

IF (ABS(saved_variable) <= cutoff) check_variable = 0.

MISSING VALUES check_variable (-9).

EXECUTE.
```

Example

```
COMPUTE check_cooks_distance = -9.

IF (ABS(cooks_distance) > 1) check_cooks_distance = 1.

IF (ABS(cooks_distance) <= 1) check_cooks_distance = 0.

MISSING VALUES check_cooks_distance (-9).

EXECUTE.
```

Standardized Variables

Note: If a custom name is not specified (as in these two examples), the new variable will be the old variable name preceded by the letter "Z", eg. `Zage`.

Basic

Format

```
DESCRIPTIVES existing_variable
 /SAVE.
```

Example

```
DESCRIPTIVES age
 /SAVE.
```

Multiple Variables

Format

```
DESCRIPTIVES variable1 (variable1_std) variable2 (variable2_std)
 /SAVE.
```

Example

```
DESCRIPTIVES age (age_std) income (income_std)
 /SAVE.
```

All Variables

Format

```
DESCRIPTIVES ALL
 /SAVE.
```

Example

```
DESCRIPTIVES ALL
 /SAVE.
```

Rescale Variables (0-1 Scale)

Basic

Note: This requires you know the minimum and maximum values for the variable, which can be found using **DESCRIPTIVES** (here). Alternatively, the examples below include syntax to automatically find and store those values.

Format

```
COMPUTE variable1_scaled = (variable1 - minimum_value) / (maximum_value - variable1_min).
EXECUTE.
```

Example

```
COMPUTE income_scaled = (income - -5100) / (1302000 - -5100).
EXECUTE.
```

Automatically Calculate Scale

Note: This is a combination of two commands, and actually creates new variables for the minimum and maximum values for each variable using **AGGREGATE** [here], as well as the rescaled variable. As such, once you run the **AGGREGATE** command once, you can reuse the minimum and maximum variables elsewhere without re-running this command.

Format

```
AGGREGATE
 /variable1_min = MIN(variable1)
 /variable1_max = MAX(variable1).
COMPUTE variable1_scaled = (variable1 - variable1_min) / (variable1_max - variable1_min).
EXECUTE.
```

Example

```
AGGREGATE
 /age_min = MIN(income)
 /age_max = MAX(income).
```

```
COMPUTE age_scaled = (age - age_min) / (age_max - age_min).
EXECUTE.
```

Multiple Variables

Format

```
AGGREGATE
 /variable1_min = MIN(variable1)
 /variable1_max = MAX(variable1)
 /variable2_min = MIN(variable2)
 /variable2_max = MAX(variable2).
COMPUTE variable1_scaled = (variable1 - variable1_min) / (variable1_max - variable1_min).
COMPUTE variable2_scaled = (variable2 - variable2_min) / (variable2_max - variable2_min).
EXECUTE.
```

Example

```
AGGREGATE
 /age_min = MIN(age)
 /age_max = MAX(age)
 /income_min = MIN(income)
 /income_max = MAX(income).
COMPUTE age_scaled = (age - age_min) / (age_max - age_min).
COMPUTE income_scaled = (income - income_min) / (income_max - income_min).
EXECUTE.
```

Helper Variables with Data from Other Variables

Note: these examples create new variables with specific data from other variables. For example, the maximum value for `variable1` might be stored as `variable1_max` or the standard deviation for `variable2` as `variable2_sd`. The value for these new variables will be the same for all cases. These are often useful later, either for viewing data using `DESCRIPTIVES` [here] or calculating new variables using `COMPUTE` [here]. For example, you could create `variable2_mean` with the mean value of `variable2`, then `COMPUTE` another variable showing the difference between the case value for `variable2` and `variable2_mean`. For an example, see the syntax for rescaling variables above.

Basic Statistics for Specific Variables

Note: These include some basic statistics. For a full list of all possible values that can be used, see here.

Note: Delete any lines for values you do not want. Repeat lines with different variables as needed. Ensure there is only one `.` [period] in the command, at the very end.

Format

```
AGGREGATE
 /variable_mean = MEAN(variable)
 /variable_min = MIN(variable)
 /variable_max = MAX(variable)
 /variable_sd = SD(variable)
 /variable_sd = SUM(variable)
 /variable_mean = MEDIAN(variable).
```

Example

```
AGGREGATE
 /age_mean = MEAN(age)
 /age_min = MIN(age)
 /age_max = MAX(age)
 /income_median = MEDIAN(income).
```

Mode for Specific Variable

Note: Mode is not available for the `AGGREGATE` command. If you just need to calculate the value, use a frequency table [here]:

```
FREQUENCIES variable
 /STATISTICS MODE.
```

However, if you need to create a common variable for the mode of another variable, you can try this workaround format. Be sure you confirm the value of the variable against the above `FREQUENCIES` command. Repeat the entire block of code below for each variable if you need the mode for multiple variables.

Format

```
COMPUTE mode_dummy = 1.
AGGREGATE
 /BREAK = variable
 /mode_dummy_sum = SUM(mode_dummy).
SORT CASES BY mode_dummy_sum(A).
AGGREGATE
 /OUTFILE OVERWRITE = YES
 /variable_mode = LAST(variable).
DELETE VARIABLES mode_dummy mode_dummy_sum.
```

Example

```
COMPUTE mode_dummy = 1.
AGGREGATE
 /BREAK = age
 /mode_dummy_sum = SUM(mode_dummy).
SORT CASES BY mode_dummy_sum(A).
AGGREGATE
 /OUTFILE OVERWRITE = YES
 /age_mode = LAST(age).
DELETE VARIABLES mode_dummy mode_dummy_sum.
```

Add/Change Descriptive Labels

Variable Labels

Basic

Format

```
VARIABLE LABELS existing_variable "New Label".
EXECUTE.
```

Example

```
VARIABLE LABELS age_ranges "Age as a Range".
EXECUTE.
```

Multiple Variables

Format

```
VARIABLE LABELS existing_variable1 "New Label"
 existing_variable2 "New Label"
 existing_variable3 "New Label".
EXECUTE.
```

Example

```
VARIABLE LABELS child "Child Ages (0-17)"
 adult "Adult (18-54)"
 senior "Senior (55+)".
EXECUTE.
```

Value Labels

Basic

Format

```
VALUE LABELS existing_variable value "Value Description".
EXECUTE.
```

Example

```
VALUE LABELS age_capped 90 "90+".
EXECUTE.
```

Multiple Values - One Variable

Format

```
VALUE LABELS existing_variable
 value1 "Value #1"
 value2 "Value #2"
 value3 "Value #3".
```

```
EXECUTE.
```

Example

```
VALUE LABELS age_ranges
 1 "Child (0-17)"
 2 "Adult (18-54)"
 3 "Senior (55+)".
EXECUTE.
```

Multiple Variables

Format

```
VALUE LABELS existing_variable1
 value1 "Variable 1 - Value #1"
 value2 "Variable 1 - Value #2"
 value3 "Variable 1 - Value #3"
 /existing_variable2
 value1 "Variable 2- Value #1"
 value2 "Variable 2- Value #2"
 value3 "Variable 2- Value #3".
EXECUTE.
```

Example

```
VALUE LABELS child
 0 "Is Not a Child"
 1 "Is a Child"
 /adult
 0 "Is Not an Adult (18-54)"
 1 "Is an Adult (18-54)"
 /senior
 0 "Is Not a Senior"
 1 "Is a Senior".
EXECUTE.
```

Limiting Scope of Cases

Filtering

Basic (Exclude Cases Where Variable = 0)

Format

```
FILTER BY existing_variable.
[SYNTAX USING FILTER]
FILTER OFF.
```

Example

```
FILTER BY insurance_recoded.
[SYNTAX USING FILTER]
FILTER OFF.
```

Custom

Note: Here we are just using `RECODE` [here] to create a filtering variable where the values we want to include != `0` (I use `1` here), and the values to exclude = `0`. I have included two formats, one where we want to exclude specific values, and another where we want to include specific values. Both have the same result, as seen in the examples.

Formats

```
RECODE existing_variable (value_to_exclude = 0)(MISSING = SYSMIS)(ELSE = 1) INTO filter.
EXECUTE.
FILTER BY filter.
[SYNTAX USING FILTER]
FILTER OFF.

RECODE existing_variable (value_to_include = 1)(MISSING = SYSMIS)(ELSE = 0) INTO filter.
EXECUTE.
FILTER BY filter.
[SYNTAX USING FILTER]
FILTER OFF.
```

Examples

```
RECODE age (LO THRU 17 = 0)(MISSING = SYSMIS)(ELSE = 1) INTO filter_age_18plus.
EXECUTE.
FILTER BY filter_age_18plus.
[SYNTAX USING FILTER]
FILTER OFF.

RECODE age (18 THRU HI = 1)(MISSING = SYSMIS)(ELSE = 0) INTO filter_age_18plus.
EXECUTE.
FILTER BY filter_age_18plus.
[SYNTAX USING FILTER]
FILTER OFF.
```

Basic Statistics

Descriptive Statistics

Basic

Format

```
DESCRIPTIVES existing_variable.
```

Example

```
DESCRIPTIVES age.
```

Multiple Variables

Format

```
DESCRIPTIVES variable1 variable2 variable3
 /MISSING LISTWISE.
```

Example

```
DESCRIPTIVES age age_ranges insurance_recoded child adult senior
 /MISSING LISTWISE.
```

All Variables

Format

```
DESCRIPTIVES VARIABLES = ALL
 /STATISTICS ALL.
```

Example

```
DESCRIPTIVES ALL
 /STATISTICS ALL.
```

Frequency Tables

Basic

Format

```
FREQUENCIES existing_variable.
```

Example

```
FREQUENCIES age.
```

Multiple Variables

Format

```
FREQUENCIES variable1 variable2 variable3.
```

Example

```
FREQUENCIES age age_ranges insurance_recoded child adult senior.
```

Percentile Values

Format

```
FREQUENCIES variable1 variable2 variable3
```

```
 /PERCENTILES # #.
```

Example

```
FREQUENCIES age insurance_recoded
 /PERCENTILES 25 75.
```

All Statistics

Format

```
FREQUENCIES existing_variable
 /BARCHART
 /PIECHART
 /STATISTICS ALL.
```

Example

```
FREQUENCIES age
 /BARCHART
 /PIECHART
 /STATISTICS ALL.
```

Means

Basic

Format

```
MEANS dependent_variable BY independent_variable.
```

Example

```
MEANS income BY age_ranges.
```

Extra Details

Format

```
MEANS dependent_variable BY independent_variable
 /CELLS = DEFAULTS MEDIAN SEMEAN RANGE.
```

Example

```
MEANS income BY age_ranges
 /CELLS = DEFAULTS MEDIAN SEMEAN RANGE.
```

Multiple Dependent Variables (Table for Each)

Format

```
MEANS dependent_variable1 dependent_variable2 BY independent_variable
 /CELLS DEFAULTS MEDIAN SEMEAN RANGE.
```

Example

```
MEANS income insurance_recoded BY age_ranges
 /CELLS DEFAULTS MEDIAN SEMEAN RANGE.
```

Contingency Tables

Basic

Format

```
CROSSTABS variable1 BY variable2.
```

Example

```
CROSSTABS age_ranges BY age.
```

Pearson Correlations

Basic

Format

```
CORRELATIONS variable1 variable2
 /MISSING LISTWISE.
```

Example

```
CORRELATIONS income age
 /MISSING LISTWISE.
```

Extra Details

Format

```
CORRELATIONS variable1 variable2
 /MISSING LISTWISE
 /STATISTICS ALL.
```

Example

```
CORRELATIONS income age
 /MISSING LISTWISE
 /STATISTICS ALL.
```

Graphs

Scatter Plot

Format

```
GRAPH SCATTERPLOT x_axis_variable WITH y_axis_variable
 /MISSING LISTWISE
 /TITLE "Graph Title".
```

Example

```
GRAPH SCATTERPLOT age WITH income
 /MISSING LISTWISE
 /TITLE "Yearly Income by Age".
```

Analysis

Bivariate OLS Regression Analysis

Basic Bivariate Regression

Format

```
REGRESSION
 /MISSING LISTWISE
 /DEPENDENT dependent_variable
 /METHOD ENTER independent_variable.
```

Example

```
REGRESSION
 /MISSING LISTWISE
 /DEPENDENT income
 /METHOD ENTER age.
```

Bivariate Regression with Additional Statistics

Format

```
REGRESSION VARIABLES dependent_variable independent_variable
 /MISSING LISTWISE
 /DESCRIPTIVES DEFAULTS N
 /STATISTICS DEFAULTS CI
 /DEPENDENT dependent_variable
 /METHOD ENTER independent_variable.
```

Example

```
REGRESSION VARIABLES income age
 /MISSING LISTWISE
 /DESCRIPTIVES DEFAULTS N
 /STATISTICS DEFAULTS CI
 /DEPENDENT income
 /METHOD ENTER age.
```

Multiple OLS Regression Analysis

Basic Multiple Regression

Format

```
REGRESSION
 /MISSING LISTWISE
 /DEPENDENT dependent_variable
 /METHOD ENTER independent_variable1 independent_variable2.
```

Example

```
REGRESSION
 /MISSING LISTWISE
 /DEPENDENT income
 /METHOD ENTER child senior insurance_recoded.
```

Multiple Regression with Additional Statistics

Format

```
REGRESSION VARIABLES dependent_variable independent_variable1 independent_variable2
 /MISSING LISTWISE
 /DESCRIPTIVES DEFAULTS N
 /STATISTICS DEFAULTS CI TOL
 /DEPENDENT dependent_variable
 /METHOD ENTER independent_variable1 independent_variable2.
```

Example

```
REGRESSION VARIABLES income child senior insurance_recoded
 /MISSING LISTWISE
 /DESCRIPTIVES DEFAULTS N
 /STATISTICS DEFAULTS CI TOL
 /DEPENDENT income
 /METHOD ENTER child senior insurance_recoded.
```

Regression with Multiple Models

Format

```
REGRESSION VARIABLES dependent_variable ind_var1 ind_var2 ind_var3 ind_var4
 /MISSING LISTWISE
 /DESCRIPTIVES DEFAULTS N
 /STATISTICS DEFAULTS CI CHANGE TOL
 /DEPENDENT dependent_variable
 /METHOD ENTER ind_var1 ind_var2
 /METHOD ENTER ind_var2 ind_var3 ind_var4.
```

Example

```
REGRESSION VARIABLES income child senior insurance_recoded
 /MISSING LISTWISE
 /DESCRIPTIVES DEFAULTS N
 /STATISTICS DEFAULTS CI CHANGE TOL
 /DEPENDENT income
 /METHOD ENTER insurance_recoded
 /METHOD ENTER child senior.
```

With Saved Residuals and Case Diagnostics

Note: On the `/SAVE` line, you can request other values be saved as variables, using the same format of `VALUE (variable_name)`. This format only saves residuals along with their standardized and studentized versions (`RESID`, `ZRESID`, and `SRESID`). Some examples of other possible values include:

- `PRED` — Predicted value
- `ADJPRED` — Adjusted Predicted value
- `COOK` — Cook's Distance
- `DRESID` — Deleted Residuals
- `SDRESID` — Studentized Deleted Residuals
- `DFBETA` — DFBeta (Note: Creates one variable for constant + each IV, with sequential numbering from 0)
- `SDBETA` — Standardized DFBeta (Note: Creates one variable for constant + each IV, with sequential numbering from 0)

Format

```
REGRESSION VARIABLES dependent_variable independent_variable1 independent_variable2
 /MISSING LISTWISE
 /STATISTICS DEFAULTS CI TOL
 /DEPENDENT dependent_variable
 /METHOD ENTER independent_variable1 independent_variable2
 /SAVE RESID (residual) ZRESID (residual_standardized) SRESID (residual_studentized)
 /CASEWISE PLOT OUTLIERS(SD_threshold).
```

Example

```
REGRESSION VARIABLES income child senior insurance_recoded
 /MISSING LISTWISE
 /DESCRIPTIVES DEFAULTS N
 /STATISTICS DEFAULTS CI TOL
 /DEPENDENT income
 /METHOD ENTER child senior insurance_recoded
 /SAVE RESID (residual) ZRESID (residual_standardized) SRESID (residual_studentized)
 /CASEWISE PLOT OUTLIERS(2).
```

With Partial Plots, Residual/Predicted Scatterplot, Residual Plots, & Case Diagnostics

Note: On the `/SAVE` line, see the format above for some of the different measures you can save. This one saves Standardized Residuals (`ZRESID`), Cook's Distance (`COOK`), and DFBetas (`DFBETA`).

Format

```
REGRESSION VARIABLES dependent_variable independent_variable1 independent_variable2
```

```
 /MISSING LISTWISE
 /STATISTICS DEFAULTS CI TOL
 /DEPENDENT dependent_variable
 /METHOD ENTER independent_variable1 independent_variable2
 /SAVE ZRESID (residual_standardized) COOK (cooks_distance) DFBETA (dfbeta)
 /SCATTERPLOT (*ZRESID *ZPRED)
 /PARTIALPLOT
 /RESIDUALS
 /CASEWISE PLOT OUTLIERS(SD_threshold).
```

Example

```
REGRESSION VARIABLES income child senior insurance_recoded
 /MISSING LISTWISE
 /DESCRIPTIVES DEFAULTS N
 /STATISTICS DEFAULTS CI TOL
 /DEPENDENT income
 /METHOD ENTER child senior insurance_recoded
 /SAVE ZRESID (residual_standardized) COOK (cooks_distance) DFBETA (dfbeta)
 /SCATTERPLOT (*ZRESID *ZPRED)
 /PARTIALPLOT
 /RESIDUALS
 /CASEWISE PLOT OUTLIERS(2).
```

Example of Multiple Regression with Diagnostic Tools to Test Assumption

Note: This is just one example of combining many syntax commands in SPSS. This syntax performs a multiple regression and then provides certain diagnostics for identifying outliers and influential cases. It gives:

- Case diagnostics for cases with standardized residuals > 2 Standard Deviations
- Various standard residual statistics
- A scatterplot of standardized predicted values x standardised residuals
- Scatterplots of the DV with every IV ("Partial Plots")
- A histogram of residuals (Standardized)
- A normal probability plot of residuals (Standardized)
- The case number and standardized residual for the 10 cases with the largest absolute standardized residuals
- The Durbin-Watson test statistic
- An at-a-glance table of:
  - # of standardized residuals > |3.29|
  - % of standardized residuals > |2.58|
  - % of standardized residuals > |1.96|
  - Min, Max, and Mean for Cook's Distance
  - Min, Max and Mean for highest |DFBetas| across variables

This format should be adjusted for your own needs. Note that any suspicious cases flagged using these diagnostics should be investigated using more thorough methods. In particular, the Maximum DFBeta included in the final table is used to see at-a-glance the highest DFBeta values across the model, and should not be relied upon for anything else.

Note: you **MUST** change the value on the `COMPUTE` line to equal the total number of independent variables in the model

Note: you will have to delete the variables created by this command if you want to run it again, or specify different variable names under the `/SAVE` and the `COMPUTE` lines as appropriate.

Format

```
REGRESSION VARIABLES dependent_variable independent_variable1 independent_variable2
 /MISSING LISTWISE
 /DESCRIPTIVES DEFAULTS N
 /STATISTICS DEFAULTS CI TOL
 /DEPENDENT dependent_variable
 /METHOD ENTER independent_variable1 independent_variable2
 /SAVE ZRESID (residual_standardized) COOK (cooks_distance) DFBETA (dfbeta)
 /SCATTERPLOT (*ZRESID *ZPRED)
 /PARTIALPLOT
 /RESIDUALS
 /CASEWISE PLOT OUTLIERS(2).
FILTER residual_standardized.
AGGREGATE
 /residual_329_count = COUT(residual_standardized -3.29 3.29)
 /residual_258_perc = POUT(residual_standardized -2.58 2.58)
 /residual_196_perc = POUT(residual_standardized -1.96 1.96).
FILTER OFF.
COMPUTE max_dfbeta = ABS(MAX(dfbeta0 TO dfbeta[# of independent variables])).
COMPUTE min_dfbeta = ABS(MIN(dfbeta0 TO dfbeta[# of independent variables])).
COMPUTE abs_max_dfbeta = MAX(max_dfbeta TO min_dfbeta).
VARIABLE LABELS residual_329_count "Number of Cases w/ Std Residual > |3.29|"
 residual_258_perc "Percent of Cases w/ Std Residual > |2.58|"
 residual_196_perc "Percent of Cases w/ Std Residual > |1.96|"
 max_dfbeta "Maximum |DFBeta| value across all regression variables".
DESCRIPTIVES residual_329_count residual_258_perc residual_196_perc cooks_distance max_dfbeta
 /MISSING LISTWISE.
```

Example

```
REGRESSION VARIABLES income child senior insurance_recoded
 /MISSING LISTWISE
 /DESCRIPTIVES DEFAULTS N
 /STATISTICS DEFAULTS CI TOL
 /DEPENDENT income
 /METHOD ENTER child senior insurance_recoded
 /SAVE ZRESID (residual_standardized) COOK (cooks_distance) DFBETA (dfbeta)
 /SCATTERPLOT (*ZRESID *ZPRED)
 /PARTIALPLOT
 /RESIDUALS
 /CASEWISE PLOT OUTLIERS(2).
FILTER residual_standardized.
AGGREGATE
 /residual_329_count = COUT(residual_standardized -3.29 3.29)
 /residual_258_perc = POUT(residual_standardized -2.58 2.58)
 /residual_196_perc = POUT(residual_standardized -1.96 1.96).
FILTER OFF.
COMPUTE max_dfbeta = ABS(MAX(dfbeta0 TO dfbeta3)).
COMPUTE min_dfbeta = ABS(MIN(dfbeta0 TO dfbeta3)).
COMPUTE abs_max_dfbeta = MAX(max_dfbeta TO min_dfbeta).
FILTER residual_standardized.
VARIABLE LABELS residual_329_count "Number of Cases w/ Std Residual > |3.29|"
 residual_258_perc "Percent of Cases w/ Std Residual > |2.58|"
 residual_196_perc "Percent of Cases w/ Std Residual > |1.96|"
 max_dfbeta "Maximum |DFBeta| value across all regression variables".
DESCRIPTIVES residual_329_count residual_258_perc residual_196_perc cooks_distance max_dfbeta
 /MISSING LISTWISE.
```

Logistic Regression

Basic Logistic Regression

Note: Both formats below work if there is a single block being entered. For multiple model analysis (here), use the first method, with the `/METHOD ENTER` line

Format

```
LOGISTIC REGRESSION VARIABLES dependent_variable
 /METHOD ENTER variable1 variable2.


LOGISTIC REGRESSION VARIABLES dependent_variable WITH variable1 variable2.
```

Example

```
LOGISTIC REGRESSION VARIABLES insurance_recoded
 /METHOD ENTER age income.


LOGISTIC REGRESSION VARIABLES insurance_recoded WITH age income.
```

Logistic Regression with Confidence Intervals

Note: Both formats below work if there is a single block being entered. For multiple model analysis (here), use the first method, with the `/METHOD ENTER` line
Note: This requests 95% confidence intervals, change the number in `CI(95)` to adjust this.

Format

```
LOGISTIC REGRESSION VARIABLES dependent_variable
 /METHOD ENTER variable1 variable2
 /PRINT CI(95).


LOGISTIC REGRESSION VARIABLES dependent_variable WITH variable1 variable2
  /PRINT CI(95).
```

Example

```
LOGISTIC REGRESSION VARIABLES insurance_recoded
 /METHOD ENTER age income
 /PRINT CI(95).

LOGISTIC REGRESSION VARIABLES insurance_recoded WITH age income
 /PRINT CI(95).
```

Logistic Regression with Automatic Dummy Variables (Last Category as Baseline)

Note: Both formats below work if there is a single block being entered. For multiple model analysis (here), use the first method, with the `/METHOD ENTER` line.
Note: `/CATEGORICAL` sets the last category in the variable as the omitted baseline, see here to specify a category.
Note: Multiple variables can be listed on the `/CATEGORICAL` line, and separate dummy variables will be used for each one

Format

```
LOGISTIC REGRESSION VARIABLES dependent_variable
 /METHOD ENTER variable1 variable2 variable3
 /CATEGORICAL variable3.

LOGISTIC REGRESSION VARIABLES dependent_variable WITH variable1 variable2 variable3
  /CATEGORICAL variable3.
```

Example

```
LOGISTIC REGRESSION VARIABLES voted
 /METHOD ENTER income age_ranges
 /CATEGORICAL age_ranges.

LOGISTIC REGRESSION VARIABLES voted WITH income age_ranges
 /CATEGORICAL age_ranges.
```

Logistic Regression with Automatic Dummy Variables (Custom Category as Baseline)

Note: Both formats below work if there is a single block being entered. For multiple model analysis (here), use the first method, with the `/METHOD ENTER` line.
Note: The number that specifies the omitted baseline category in `INDICATOR([baseline category number])` refers to the number it is in sequence, not necessarily the value itself. Eg. if the categories for a variable have the values 3,5,7, & 15, putting `INDICATOR(2)` sets 5 (the 2nd value sequentially) as the baseline category.
Note: Only one variable can be listed on the `/CONTRAST` line, but multiple `/CONTRAST` lines can be included.

Format

```
LOGISTIC REGRESSION VARIABLES dependent_variable
 /METHOD ENTER variable1 variable2 variable3
 /CONTRAST (variable3) INDICATOR([baseline category number]).

LOGISTIC REGRESSION VARIABLES dependent_variable WITH variable1 variable2 variable3
 /CONTRAST (variable3) INDICATOR([baseline category number]).
```

Example

```
LOGISTIC REGRESSION VARIABLES voted
 /METHOD ENTER income age_ranges
 /CONTRAST (age_ranges) INDICATOR(1).

LOGISTIC REGRESSION VARIABLES voted WITH income age_ranges
 /CONTRAST (age_ranges) INDICATOR(1).
```

Logistic Regression with Multiple Models

Note: The `/METHOD ENTER` line is required when entering variables, more can be added for as many blocks as needed

Format

```
LOGISTIC REGRESSION VARIABLES dependent_variable
 /METHOD ENTER variable1
 /METHOD ENTER variable2 variable3.
```

Example

```
LOGISTIC REGRESSION VARIABLES insurance_recoded
 /METHOD ENTER age
 /METHOD ENTER income.
```

Logistic Regression with Outlier and Influential Case Diagnostics

Note: Both formats below work if there is a single block being entered. For multiple model analysis (here), use the first method, with the `/METHOD ENTER` line.
Note: you will have to delete the variables created by this command if you want to run it again, or specify different variable names under the `/SAVE` and the `COMPUTE` lines as appropriate.

Format

```
LOGISTIC REGRESSION VARIABLES dependent_variable
 /METHOD ENTER variable1 variable2 variable3
 /CASEWISE OUTLIERS
 /SAVE ZRESID (residual_standardized) LEVER (leverage) COOK (cooks_distance) DFBETA (dfbeta).

LOGISTIC REGRESSION VARIABLES dependent_variable WITH variable1 variable2 variable3
 /CASEWISE OUTLIERS
 /SAVE ZRESID (residual_standardized) LEVER (leverage) COOK (cooks_distance) DFBETA (dfbeta).
```

Example

```
LOGISTIC REGRESSION VARIABLES insurance_recoded
 /METHOD ENTER income age
 /CASEWISE OUTLIERS
 /SAVE ZRESID (residual_standardized) LEVER (leverage) COOK (cooks_distance) DFBETA (dfbeta).
```

```
LOGISTIC REGRESSION VARIABLES insurance_recoded WITH income age
 /CASEWISE OUTLIERS
 /SAVE ZRESID (residual_standardized) LEVER (leverage) COOK (cooks_distance) DFBETA (dfbeta).
```

Collinearity Diagnostics for Logistic Regression

Note: Collinearity diagnostics are not available with `LOGISTIC REGRESSION`, so instead we run an OLS `REGRESSION` and only ask SPSS to display collinearity diagnostics with `/STATISTICS TOL`.

Format

```
REGRESSION
 /MISSING LISTWISE
 /DEPENDENT dependent_variable
 /METHOD ENTER variable1 variable2 variable3
 /STATISTICS TOL.
```

Example

```
REGRESSION
 /MISSING LISTWISE
 /DEPENDENT insurance_recoded
 /METHOD ENTER income age
 /STATISTICS TOL.
```

Custom Macros

!OLSComp (Calculate Predicted Values)

Setup (Once per Session)

- Download the !OLSComp syntax file here if you have not saved it already
- Open in SPSS (double-click the file or select from File > Open > Syntax)
- Highlight all text (with cursor or ctrl+a)
- Run the syntax (green “play” button or ctrl+r)
- Repeat every time you restart SPSS to enable !OLSComp

Basic Predictions

Note: The constant values for predictor values, as well as mean(x) ± 1SD(x) can be calculated from the table generated by the first part of the command.
Note: You can only run two predictions at a time (Set1 and Set2). To run more, repeat the `!OLSCOMP` syntax.

Format

```
DESCRIPTIVES dependent_var independent_variable1 independent_variable2 independent_variable3
/MISSING LISTWISE.


!OLSCOMP
```

```
 Y = dependent_var /
 XList = independent_variable1 independent_variable2 independent_variable3 /
 Set1 = {1; IV1_value1; IV2_value_constant; IV3_value_constant} /
 Set2 = {1; IV1_value2; IV2_value_constant; IV3_value_constant} /
 Title = "FV1: Independent Variable 1 - Value 1;FV2: Independent Variable 1 - Value 2".
```

Example

```
DESCRIPTIVES income age insurance_recoded
/MISSING LISTWISE.

!OLSCOMP
 Y = income /
 XList = age insurance_recoded /
 Set1 = {1; 25; 1} /
 Set2 = {1; 75; 1} /
 Title = "FV1: Age 25;FV2: Age 75".
```

PROCESS (Moderation Analysis)

Note: Additional instructions on this command cannot be found in the Command Syntax Reference. Instead, refer to Hayes (2013) - Appendix A. [Leiden] [WorldCat]

Setup (Once per Session)

- Download the PROCESS files here if you have not saved it already
- Unzip and find "PROCESS v3.5 for SPSS/process.sps"
- Open in SPSS (double-click the file or select from File > Open > Syntax)
- Highlight all text (with cursor or ctrl+a)
- Run the syntax (green "play" button or ctrl+r)
- Repeat every time you restart SPSS to enable PROCESS
- Can also be added as a menu option as described in Field (2018) [Leiden][WorldCat] 4.13.1

Basic Model with One Moderator

Format

```
PROCESS intprobe = 1 /model = 1 /longname = 1
 /y = dependent_variable
 /x = independent_variable
 /mcx = 1 [if x is categorical, otherwise delete this line]
 /w = moderator_variable.
```

Examples

```
PROCESS intprobe = 1 /model = 1 /longname = 1
 /y = income
 /x = age_ranges
 /mcx = 1
 /w = insurance_recoded.

PROCESS /intprobe = 1 /model = 1 /longname = 1
 /y = income
 /x = age
 /w = insurance_recoded.
```

Model with Control Variables

Format

```
PROCESS intprobe = 1 /model = 1 /longname = 1
 /y = dependent_variable
 /x = independent_variable
 /mcx = 1 [if x is categorical, otherwise delete this line]
 /w = moderator_variable
 /cov = variable1 variable2.
```

Examples

```
PROCESS intprobe = 1 /model = 1 /longname = 1
 /y = income
 /x = age_ranges
 /mcx = 1
 /w = insurance_recoded
 /cov = left_right_pol_scale.

PROCESS /intprobe = 1 /model = 1 /longname = 1
 /y = income
 /x = age
 /w = insurance_recoded
 /cov = left_right_pol_scale.
```

Center Moderating Variable

Format

```
PROCESS intprobe = 1 /model = 1 /longname = 1
 /y = dependent_variable
 /x = independent_variable
 /mcx = 1 [if x is categorical, otherwise delete this line]
 /w = moderator_variable
 /cov = variable1 variable2
 /center = 1.
```

Examples

```
PROCESS intprobe = 1 /model = 1 /longname = 1
 /y = income
 /x = age_ranges
 /mcx = 1
 /w = insurance_recoded
 /cov = left_right_pol_scale
 /center = 1.
```

```
PROCESS /intprobe = 1 /model = 1 /longname = 1
 /y = income
 /x = age
 /w = insurance_recoded
 /cov = left_right_pol_scale
 /center = 1.
```

Probe at Mean ± 1SD and Center Moderating Variable

Format

```
PROCESS intprobe = 1 /model = 1 /longname = 1
 /y = dependent_variable
 /x = independent_variable
 /mcx = 1 [if x is categorical, otherwise delete this line]
 /w = moderator_variable
 /cov = variable1 variable2
 /center = 1 /moments = 1.
```

Examples

```
PROCESS intprobe = 1 /model = 1 /longname = 1
 /y = income
 /x = age_ranges
 /mcx = 1
 /w = insurance_recoded
 /cov = left_right_pol_scale
 /center = 1 / moments = 1.
```

```
PROCESS /intprobe = 1 /model = 1 /longname = 1
 /y = income
 /x = age
 /w = insurance_recoded
 /cov = left_right_pol_scale
```

```
/center = 1 /moment = 1.
```