Bug fix Gauss Systems- Versione 22

Di seguito un elenco con le correzioni di *bug* relative alla versione Gauss 22:

1. Added new preprocessor #includedir to add current file directory to source path. If executed from the Program Input/Output Window (PGM), uses current working directory.
2. #include and #includedir statements can now be processed with F4 in the GUI.
3. **__FILE_DIR()** now works with F4 in the GUI.
4. Added new function **resetsourcepaths()** to restore source path to initial value from gauss.cfg.
5. Duplicate header prevention was added for dataframes. This can be toggled via policy in gauss.cfg with the policy_check_df_header_dupes key.
6. Added new function **asdf()** to allow automatic conversion of scalar/matrix/string/string array to a dataframe. Headers can now be specified as N additional arguments, where N is equal to the column count of the input symbol.
7. Added new function **currentprocname()** was added to return the name of the current proc. It also accepts 1 argument to return the name of the calling procedure(s) from previous stack frame(s) if desired.
8. Added new function **isunique()** to return a scalar denoting whether all rows in a matrix or dataframe are unique.
9. Added new function **isrowunique()** to return a vector denoting whether each row is unique.
10. Added new function **dropduplicates()** to return the input matrix/dataframe with all duplicate rows removed.
11. Added new function **getduplicates()** to return the input matrix/dataframe with only duplicate rows present. The original row number is prepended as the first column to the output of this function.
12. **setcolnames()** now has header duplicate checking and auto-renaming if the policy_check_df_header_dupes policy is enabled in gauss.cfg. This policy is enabled by default.
13. **dfname()** was added as an alias for **setcolnames()**.
14. **dftype()** was added as an alias for **setcoltypes()**.
15. **asdate()** was added as an alias for **setcoldateformats()**.
16. %e, %E, %f, %F, %g, %G flags were reimplemented for more consistent results with **sprintf()**.
17. **dttostr()** will now return a string instead of a 1x1 string array.
18. Formula strings now support more than one dependent variable. (fields specified before a ~ in a formula string).
19. Added %v support to **asdate()**. This is shorthand for %e-%b-%Y.
20. Added support for the automatic monthly, quarterly, and yearly date-conversions in Stata files (e.g. .dta files).
21. **sprintf()** now supports the following base conversion patterns: %b (binary), %o (octal), %x and %X (hex).
22. **aggregate()** now accepts an optional input specifying the column index or name of the variable to aggregate on.

23. Graphics: **plotScatter()**, **plotXY()** and **plotBox()** now support formula strings and automatically handle dataframe input to generate the appropriate axis and legend labels.

24. Graphics: New formula string keyword, **by** splits data to be plotted by **plotScatter()**, **plotXY()** and **plotBox()** by a specified categorical or string variable and automatically handle dataframe input to generate the appropriate legend items.

25. Graphics: Added support for legends to have their own title with **plotSetLegendTitle()**.

26. Graphics: Added new functions for vertical/horizontal lines to span the entire axis: **plotAddVLine()**, **plotAddHLine()**.

27. Graphics: Added new functions for vertical/horizontal bars to span the entire axis: **plotAddVBar()**, **plotAddHBar()**.

28. Graphics: **plotAddVBar()** and **plotAddHBar()** support FRED-style input data. (eg { 1950, 1 }, { 1951, 1 }, { 1952, 0 }, …).

29. Graphics: Added support for dates in simple string format to **plotSetXRange()**.

30. Graphics: Added support for outliers to **plotBox()**.

31. Graphics: **plotBox()** can now accept a vector of groups as the first data input. The y variable will be split by the categories in the group vector and plotted as separate boxes.

32. Graphics: Added new function **plotSetJitterRange()** to control the jitter range for **plotScatter()** and **plotBox()** outliers.

33. Graphics: Attributes for each axis can be assigned separately. The existing **plotSetAxesPen()** convenience procedure will still assign attributes to all axes simultaneously.

34. Graphics: The font can now be specified for **plotContour()** labels.

35. Graphics: Axis tics can now be displayed on the inside of the chart (as opposed to outside only) or hidden completely with the **plotSetTicPosition()** function.

36. Graphics: Added new function **plotSetOutlineEnabled()** to allow a box outline around the entire chart. Outline attributes are controlled via axis properties using **plotSetAxesPen()** or individually with **plotSetXPen()** and **plotSetYPen()**.

37. Graphics: Axes are now at a higher Z-order than series, so lines will not render on top of the axes lines.

38. **plotSetAxesPen()** has a new optional input to set the axes line style.

39. Graphics: New functions **plotSetXGridPen()** and **plotSetYGridPen()** allow the major and minor x and y axis grid lines to be enabled and styled separately.

40. Graphics: New function **plotSetGridPen()** allows the major and minor x and y axis grid lines to be enabled and styled.

41. Graphics: Graph profile settings in the preferences dialog have been fully refactored to only show properties related to the selected graph category. This should reduce confusion regarding which properties are respected when plotting a graph of the specified type.

42. Graphics: Added support for specifying the bar width (**plotSetBarWidth()**) and box width (**plotSetBoxWidth()**).

43. Graphics: **plotAddXY()** and **plotAddScatter()** now support category labels as input for X values, so data can be added to locations specified by a text label, rather than a numeric value.

44. Graphics: Contour is now a new default graph profile instead of being shared with Surface.

45. Graphics: New convenience function **plotSetLinePen()** to set the line width, color and style in one call.
46. Graphics: New function **plotCloseAll()** closes all open graphs.
47. Graphics: Performance: support was improved for plotting large numbers of points for XY and scatter series.
48. Graphics: Behavior change: the default line thickness for bar plots has been set to zero to be consistent with commonly desired styling for added spanning bars.
49. Graphics: Behavior change: the legend position can be updated with settings from a plotAdd call if no legend items already exist on the graph.
50. Graphics: Bug Fix: Outside middle legend will now always have a vertical orientation.
51. Graphics: Bug Fix: **plotOpenWindow()** now retains focus in the widget prior to the call (eg the PGM).
52. Graphics: Bug Fix: **plotAddBarH()** would calculate the X offset incorrectly if the input X values were index values instead of labels.
53. Graphics: Bug Fix: Axis properties for **plotPolar()** was applying the settings in a reversed manner. X-Axis settings now represent the azimuth with Y-Axis settings representing the radial axis. The input order for **plotPolar()** has not changed.
54. Graphics: Bug Fix: **plotTSHF()** would not allow a fixed axis range to be specified.
55. Graphics: Bug Fix: **plotTSHF()** would sometimes omit axis labels in the case of too few calculated labels. At least 1 will be rendered now.
56. Performance: **movingave()** up to 4-6x faster.
57. Performance: **unique()** was optimized and should be faster.
58. For convenience you can now assign a scalar value to multiple elements of a matrix or dataframe (eg x[1 3 5,2] = 7.3;).
59. Dataframes: All dataframe functions (**dfname()**, **dftype()**, **asdate()**, etc) can now automatically convert a non-dataframe input to a dataframe. String arrays are automatically converted to a category column.
60. Dataframes: **asdate()** now allows omission of the format argument, and will default to %Y-%m-%d %H:%M:%S.%L. All or part of this format can be specified in the input argument.
61. Dataframes: passing a format of %s to **asdate()** will automatically coerce it to a friendly format.
62. Dataframes: Behavior: Overwriting an entire column during an assign will overwrite the LHS metadata if the RHS is also a dataframe.
63. Dataframes: Behavior: Combining dataframes with string arrays using the string combine operator, $+ is now supported.
64. Dataframes: Multiple new functions now support dataframes as input arguments: **strtrim()**, **strtriml()**, **strtrimr()**, **strtrunc()**, **strtruncl()**, **strtruncr()**, **strtruncpad()**, **upper()**, **lower()**, **strindx()**, **strreplace()**, **strsect()**, **indsav()**, **indnv()**, **contains()**, **strsplit()**, **strjoin()**, **strcombine()**, **aggregate()**
65. Dataframes: A low-level function **normalizecollabels()** was added to automatically refactor string/category columns to remove duplicates and consolidate keys.
66. Dataframes: Added string/string array assignment support to existing string/category columns.
67. Dataframes: Date pattern matching has been relaxed. If a string fully matches a date format pattern completely, the calculated date up until that point is now returned instead of requiring the entire format to be satisfied. Most functions that take a date format pattern now default to allowing full/partial usage of the pattern "%Y-%m-%d %H:%M:%S.%L".

68. Dataframes: **strctoposix()** now returns a dataframe.
69. Dataframes: Symbols viewed in the symbol editor will now show up as a 'Dataframe' in the type field instead of 'Matrix'.
70. Dataframes: Improved behavior when checking for like-column types in a partial row assignment from one dataframe to another.
71. Dataframes: **outerjoin()** (left outer join) has been rewritten completely as an intrinsic with full support for dataframes with a significant performance increase.
72. Dataframes: Generated code in the file import dialog now takes advantage of new dataframe behavior to allow more concise code.
73. Dataframes: Specifying custom col labels for string/category columns now uses a **seqa()** representation for the values if they are left as their default. (Optimization)
74. Dataframes: **sortc()** now allows you to specify columns by name.
75. Dataframes: Empty date formats now default to the default date format of %Y-%m-%d.
76. Dataframes: Any function converting a symbol to a string/category will now sort the labels before generating the keys.
77. Dataframes: Bug Fix: Unsorted indices passed to dataframe functions could cause changes to be incorrectly applied.
78. Dataframes: Bug Fix: Specific cases where a program errored out could potentially remove metadata from a symbol in the workspace.
79. Dataframes: Bug Fix: Metadata was not being applied correctly in specific struct-index assignment cases.
80. Dataframes: Bug Fix: String/Category columns can now be used with the %s pattern in **sprintf()**.
81. Dataframes: Bug Fix: All dataframe and string combinations are now supported for $+ operations.
82. Behavior Change: **aggregate()** will now check for and ignore missing values by default. An optional input flag has been added to not check for missing values as in the previous version.
83. Behavior Change: Code generation for dataframe operations in the symbol editor have been optimized to be as concise as possible.
84. Behavior Change: Columns in the symbol editor will attempt to automatically resize to yield a more user-friendly display.
85. Behavior Change: Multiple equality filters of the same type in the dataframe 'Filter' tab are now grouped together to use **rowcontains()** for optimized code generation and performance.
86. Behavior Change: Policy policy_scalar_df_indexing is now enabled by default. This policy was added in 21.0.6 to control behavior for dataframe indexing operations that return a scalar. Resulting scalar will now remain a dataframe by default.
87. Bug Fix: **setcollabels()** incorrectly allowed the indices argument to be omitted. This has been fixed, but improved to allow omission of the indices argument if the input argument only has one column. The values used will be [0…N-1] where N is the number of labels.
88. Bug Fix: **move()** now makes a copy if the input symbol can't release ownership.
89. Bug Fix: Use system palette when restoring regular font color in textbox of editor/PGM find widgets.
90. Bug Fix: Custom missing values set with **msym()** was incorrectly printing the missing value backwards in **sprintf()**.

91. Bug Fix: **selif()** could return a partial dataframe if the return value was a scalar missing.
92. Bug Fix: If a tab character was the delimiter in the file import dialog, the generated code would include a literal tab character as a string. This has been fixed to escape the tab character in the string (eg ctl.delimiter = "\t").
93. Bug Fix: **seqadt()** and **seqaposix()** now correctly allow dataframes to pass through without losing their metadata.
94. Bug Fix: Formula strings that contained a : or * character in the argument field (eg date($my_date, '%Y-%m %H:%M')) were being treated as multiplier operations.
95. Bug Fix: Add date cell editing support in the symbol editor.
96. Bug Fix: In the import dialog, generated code was not updating when a custom category label or date format was specified. This bug was visual only, as the correct code was generated when the *Import* button was pressed.
97. Bug Fix: In the import dialog, the input box for the new column name was not noticeably greyed out on macOS when the widget was disabled.
98. Bug Fix: A missing/NaN in a string/category column will now display the correct value when printed, instead of an empty string.
99. Bug Fix: The symbol editor will no longer automatically open the 'Manage' panel for dataframes.
100. Bug Fix: **setcolnames()** was incorrectly allowing empty names as input.